Détection de botnets domain-flux dans un réseau à large échelle

Hachem Guerid, Karel Mittig
Orange Labs
42 rue des Coutures, 14000, Caen, France
hachem.guerid@orange.com
karel.mittig@orange.com

Ahmed Serhrouchni
Infres, TELECOM ParisTech
46, rue Barrault, 75013, Paris, France
ahmed@telecom-paristech.fr

***Abstract*—Différentes approches ont été proposées pour détecter les noms de domaine de botnets, mais ces dernières n'abordent pas les problématiques de volumétrie et de confidentialité requises pour une implémentation à large échelle. Nous proposons une nouvelle approche de détection de noms de domaine de botnets de type domain-flux. Nous utilisons dans notre analyse les filtres de Bloom, une structure de donnée probabiliste, pour minimiser l'espace de stockage et respecter la confidentialité des utilisateurs infectés. Notre implémentation démontre que notre approche détecte un nombre important de noms malveillants tout en respectant les contraintes d'un réseau à large échelle.**

Keywords-component; Botnets; Domain Flux; DGA; Bloom Filters; DNS

I. INTRODUCTION

Face à la prolifération de la cybercriminalité, la sécurité des réseaux et des services est devenue un enjeu majeur pour les opérateurs télécom et leurs utilisateurs. Dans ce contexte, la problématique des botnets, réseau de machines infectées par des logiciels malveillants permettant à des cybercriminels d'en prendre le contrôle, est devenu une préoccupation majeure du fait du nombre de machines infectées et des menaces associées : attaques par dénis de service distribué (DDoS), spam, vol de données bancaires, etc.

Ces machines infectées sont sous le contrôle de machines maîtres, qui sont, à leur tour, sous le contrôle de cybercriminels. Les machines maîtres envoient des commandes en permanence vers les machines corrompues afin de recueillir des informations dérobées ou pour déclencher des attaques. Les botnets utilisent pour leurs communications des protocoles légitimes afin de dissimuler leur présence dans le réseau. Ces communications diffèrent d'un botnet à un autre, ce qui rend la tâche de détection difficile.

Pour que les machines infectées se connectent avec le serveur de contrôle et commandes (C&C), elles doivent posséder son adresse IP ou son nom de domaine. Le nom de domaine se base sur le protocole DNS (Domain Name System), qui permet d'associer une adresse IP à un nom de domaine.

L'identification des noms de domaine d'un botnet et de leurs adresses IP associées permettent d'alimenter des listes noires de serveurs malveillants, comme Zeus Tracker [1] ou Malware Domain List [16]. Ces listes sont maintenues par une communauté d'experts ou par des entités privées. Ils sont utilisés par les administrateurs réseaux afin de bloquer l'accès à des sites malveillants référencés et de détecter les machines infectées de leur parc.

Pour contourner ces contre-mesures, certains botnets (Torpig, Conficker, Kraken) [2] [3] génèrent des centaines, voire des milliers de noms de domaine chaque jour. Les machines infectées génèrent une liste de noms de domaine en se basant sur des paramètres communs à toutes les machines. Cette technique est appelée Domain Flux [1], en référence au Fast-Flux [4] où l'adresse IP rattachée à un nom de domaine est alternée à chaque unité de temps.

Dans cette liste de noms de domaine générés, l'opérateur du botnet ne va en réserver qu'un (ou quelques uns), le reste des noms de domaine restant indéfinis et générant ainsi des erreurs lors de la résolution. Les machines infectées sollicitent successivement sans succès les noms de domaines potentiels jusqu'au moment où elles obtiendront une résolution DNS valide leur indiquant la présence d'un serveur malveillant.

Dans la méthode de détection que nous proposons, nous nous basons sur le fait que les machines infectées par un botnet de type domain-flux vont présenter un taux d'erreurs de résolution DNS anormalement élevé avant de trouver le serveur de contrôle et commande (C&C). Par ailleurs, les noms de domaine demandés par les bots d'un même botnet (contrôlées par la même entité) appartiennent au même ensemble. Ceci implique un phénomène de convergence du trafic DNS tant au niveau des erreurs générées que des résolutions valides. C'est précisément cette caractéristique que nous proposons d'exploiter dans cet article, afin d'arriver à une nouvelle approche permettant de détecter les botnets.

Afin d'établir les communautés des machines infectées par le même botnet, nous nous appuyons sur des filtres de Bloom [5], structures de données probabilistes, pour représenter les noms de domaine inexistants demandés.

On utilise les filtres de Bloom pour atteindre les objectifs de volumétrie et de confidentialité requis pour envisager une implémentation à large échelle. En effet, les filtres de Bloom permettent de minimiser l'espace de stockage tout en étant non réversibles, c'est-à-dire qu'ils ne permettent pas (ou très

difficilement) de reconstituer l'information qui a servi à les remplir [6].

La première étape de notre approche consiste ainsi à établir des communautés de sources appartenant à un même botnet par analyse des erreurs de résolution. Une fois qu'une communauté est construite, nous analysons dans une deuxième étape les requêtes valides de cette communauté afin d'identifier le nom de domaine du C&C.

En appliquant notre méthode de détection sur des captures de trafic DNS d'une heure, nous avons pu identifier des serveurs de contrôle et de commande liés à différents botnets de types domain-flux et associés à des centaines de noms de domaine malveillants.

La suite de la publication est organisée comme suit : Dans la section II, nous introduisons l'utilisation des filtres de Bloom. Dans la section III, nous comparons notre approche avec les méthodes de détection de noms de domaine malveillants existantes. Cette méthode de détection de botnet de type domain-flux est ensuite détaillée dans les sections IV et V. Les résultats de notre implémentation sont présentés dans la section VI, suivis par une conclusion en section VII.

II. BLOOM FILTERS

Un filtre de bloom [4] est une structure de données probabiliste qui permet de représenter un ensemble d'éléments. La particularité principale de cette structure de données est d'optimiser l'espace mémoire avec un faible taux de faux positifs sans générer de faux négatifs.

Un filtre de Bloom correspond à un tableau de *m* bits, tous initialisés à zéro, et *k* fonctions de hachage indépendantes entre elles (voir figure 1). Les fonctions de hachage sont définies sur l'ensemble d'entiers *[1,m]*. Ces fonctions de hachages sont uniformément distribuées dans l'ensemble considéré.

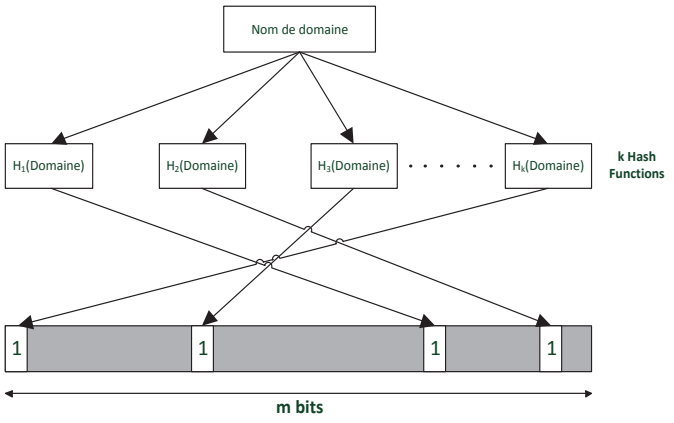

Figure 1. Pour chaque fonction de hachage, positionnner le bit correspondant à 1.

Pour chaque élément que l'on ajoute à l'ensemble, on applique les k fonctions de hachage. La sortie de chaque fonction de hachage est définie sur l'ensemble *[1,m]* correspondant à un emplacement spécifique dans le tableau [7]. Le bit de cet emplacement est positionné à 1.

Pour savoir si un élément appartient à l'ensemble représenté par un filtre de Bloom, on vérifie les emplacements correspondant aux k fonctions de hachage. Si tous les bits sont à 1 alors l'élément appartient à l'ensemble avec un certain taux de faux positifs. Par contre si un emplacement du tableau est à *0* alors l'élément n'appartient pas à l'ensemble (puisqu'il n'y a pas de faux négatifs).

Une des propriétés du filtre de Bloom est qu'il permet de tester si un élément appartient à l'ensemble, mais ne permet pas de retrouver les éléments ayant permis de construire cet ensemble. Cette propriété, ajoutée à la notion probabiliste des filtres de Bloom, permet ainsi de garantir l'anonymat des informations collectées et donc de protéger la vie privée des utilisateurs du réseau dans lequel notre solution de détection serait déployée.

III. ETAT DE L'ART

Plusieurs approches visent à détecter les botnets en utilisant le trafic DNS. Parmi celles-ci, les plus notables sont Exposure [8] et Notos [9]. L'approche proposée par Exposure consiste à évaluer un ensemble de paramètres temporels ainsi que les champs de réponses DNS. Notos propose un système de calcul permettant de calculer dynamiquement une réputation pour les adresses IP associées aux noms de domaine.

Contrairement aux deux approches citées précédemment, notre approche de détection ne nécessite qu'un historique très limité, ne dépassant pas 30 minutes de trafic, et ne portant que sur le trafic en erreur. Par ailleurs, la méthode que nous proposons ne s'appuie pas sur des valeurs spécifiques (informations sur les adresses IP, valeurs de TTL). Ceci permet d'empêcher certaines stratégies d'échappement observées sur certains botnets : utilisation de sites web légitimes piratés comme C&C, …

D'autres approches [10] [11] se basent sur un analyseur alphanumérique pour détecter des noms de domaine générés de façon automatique. Ils affirment que ces derniers n'interférent pas avec les noms de domaine légitimes. Ils génèrent pour cela des mots imprononçables. L'approche [11] utilise les erreurs de résolution DNS pour raffiner son analyse. Notre méthode de détection détecte les noms de domaine utilisés dans les botnets de type domain-flux quelque soit leur nature syntaxique.

Jiang N. et al. [12] utilisent les noms de domaine erronés pour détecter les membres d'un botnets. Ils représentent la correspondance entre les noms de domaine erronés avec les hôtes du réseau avec une matrice. Notre approche diffère de cette dernière par notre utilisation de filtres de Bloom ce qui minimise l'espace de stockage et le temps de calcul nécessaire, ainsi que l'analyse des requêtes abouties pour l'identification automatique du nom de domaine lié au botnet détecté.

Dans notre approche de détection, nous comparons entre les vecteurs de différents filtres de Bloom pour identifier les membres d'une même communauté de bots. Cette méthode est utilisée dans les moteurs de recherches [13] pour affiner les résultats présentés aux utilisateurs.

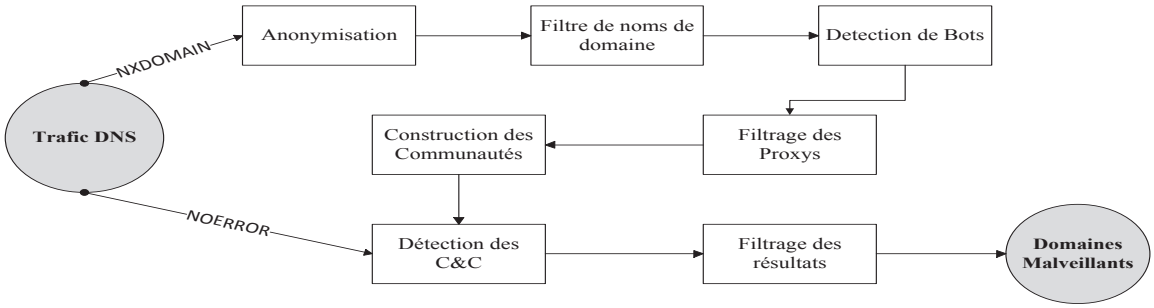

Figure 2. Etapes de détection de noms de domaine Domain Flux

IV. LA METHODE DE DETECTION

Afin d'échapper aux contre-mesures et de rendre l'infrastructure de leurs botnets plus flexible, les opérateurs de botnets ont implémenté une technique qui consiste en la génération de noms de domaine à intervalle de temps déterminé, ce qui leur permet de contourner les méthodes de détection classiques s'appuyant sur des listes noires.

Les machines infectées utilisent des algorithmes de type DGA [2] (Domain Generation Algorithm) pour générer une liste de noms de domaine. Ces algorithmes se synchronisent en utilisant des paramètres partagés ou accessibles à toutes les machines infectées, comme la date actuelle. Ceci permet à l'ensemble des machines du botnet de générer une liste de noms de domaine équivalente à un instant t. On peut citer par exemple le botnet Torpig qui a utilisé Twitter [10] afin de synchroniser les fonctions de génération de noms de domaine de ses bots, ou encore le botnet Conficker qui a utilisé la date courante de sites connus populaires afin de synchroniser les DGA.

Une fois qu'un bot a généré une liste de noms de domaine, il effectue des résolutions DNS sur ces noms jusqu'à ce qu'il trouve une correspondance avec une adresse IP. Cette adresse IP correspond dans la majorité des cas à l'adresse IP du serveur de contrôle et commande du botnet.

Une machine infectée va ainsi générer un nombre important d'erreurs de résolution DNS spécifiques (NXDOMAIN). Notre approche de détection se base sur le fait que : (i) un bot génère un nombre plus important d'erreurs de résolution DNS qu'une machines non infectée ; et que (ii) les erreurs de résolutions générées par les bots d'un même botnet convergent sur les mêmes noms de domaine, contrairement aux erreurs humaines [12].

L'ensemble des noms de domaine inexistants sollicités par chaque machine sont représentés dans une structure de données probabiliste (Filtre de Bloom).

Ces filtres sont utilisés pour construire des communautés de bots appartenant au même botnet. Les machines effectuant le même type d'erreurs DNS dans la même unité de temps sont susceptibles d'appartenir au même botnet. Une fois que la taille de la communauté atteint un certain nombre, on bascule cette dernière vers une deuxième phase d'analyse.

Dans cette deuxième étape, on analyse les requêtes DNS réussies de chaque communauté construite dans la première étape. Si un nom de domaine est demandé par la majorité des membres d'une même communauté, il est classifié comme étant malveillant ou très populaire.

Pour différencier les noms de domaine malveillants des noms de domaine très populaires, on utilise une liste blanche reprenant les 100 noms de domaine les plus populaires référencés par Alexa [14]

V. DETAILS

Pour pouvoir détecter les bots, nous avons besoin d'avoir un identifiant qui nous permet d'identifier les machines infectées, le nom de domaine demandé, ainsi que l'existence du nom de domaine.

Les informations nécessaires à la détection sont disponibles dans les paquets de réponses DNS. Nous faisons correspondre les adresses IP avec l'identifiant de la machine infectée. Les informations concernant le nom de domaine ainsi que son existence sont aussi disponibles dans les réponses DNS.

Notre approche de détection comprend plusieurs étapes (voir figure 2) : Anonymisation, filtrage des noms de domaine non valides, détection des bots, filtrage des proxys, construction des communautés, identification des C&C, et le filtrage des noms de domaine populaires.

A. Anonymisation :

L'objectif de notre approche vise à identifier les serveurs de contrôle et commande et non les machines infectées, une première étape consiste à anonymiser les adresses IP à l'origine des résolutions DNS. Pour cela, une fonction de hachage non réversible avec sel est appliquée sur les adresses IP source.

Cette opération permet ainsi de renforcer la confidentialité du trafic, les données utilisées pour la suite de l'analyse n'étant alors plus constituées que d'un identifiant unique associé à un filtre de Bloom.

B. Filtrage des noms de domaine non valides

Les requêtes DNS qui sont générées par les machines infectées afin de retrouver leur serveur de contrôle reposent sur une syntaxe que nous qualifions de valide. Nous définissions une syntaxe valide comme étant constituée d'un domaine de

premier niveau (TLD) existant, et au minimum d'un domaine de second niveau.

Lors de l'analyse d'une réponse DNS NX, il n'y a pas de différence entre un nom de domaine invalide et un nom de domaine valide mais inexistant. Pour cette raison, nous effectuons un pré-filtrage des réponses DNS de type NX. Les noms de domaine de premier niveau qui n'appartiennent pas à la liste des TLD connus sont ainsi filtrés et exclus de l'analyse.

Ce filtrage permet d'exclure de l'analyse un grand nombre d'erreurs de configuration logicielles (« localhost », « .home », …).

C. Détection des bots:

On associe à chaque identifiant source un filtre de Bloom. Ce filtre contient l'ensemble des noms de domaine inexistants sollicités par la machine associée à l'identifiant.

Nous utilisons une longueur de vecteur pour les filtres de Bloom de 800 bits. Cette longueur nous permet d'identifier les identifiants effectuant un nombre élevé de requêtes DNS, tout en minimisant l'espace nécessaire pour conserver cette information.

Lors de l'ajout d'une entrée dans le filtre de Bloom, on effectue en même temps un test d'appartenance afin de déterminer si le nom de domaine en question a déjà été sollicité. Si le nom de domaine n'a jamais été sollicité auparavant, un compteur est incrémenté, représentant le nombre de noms de domaine inexistants différents sollicités par ce même identifiant.

Ce compteur est utilisé pour déterminer la probabilité qu'un identifiant corresponde à une machine infectée. Lorsque le compteur dépasse une valeur seuil (indiquant que la machine associée à l'identifiant a généré un nombre élevé de requêtes vers des noms de domaine inexistants), l'enregistrement associé est basculé dans la liste des identifiants suspects.

D. Filtrage des Proxys

On reçoit dans cette étape une liste d'identifiants correspondant à des bots potentiels, chaque identifiant étant associé à son filtre de Bloom.

Afin de corréler correctement le trafic entre plusieurs bots d'un même botnet, il est important à ce stade d'identifier et d'exclure de l'analyse les identifiants correspondant à des serveurs proxy. En effet, ces serveurs proxy, qui correspondent en général à des réseaux d'entreprises, génèrent un biais dans l'analyse car ils agrègent le trafic d'un nombre indéfini de machines. Ils génèrent ainsi un nombre très important de requêtes DNS, et ne présentent pas le phénomène de convergence remarqué pour les machines individuelles.

On identifie les serveurs proxy en se basant sur le taux de remplissage du filtre de Bloom. Un taux de remplissage élevé correspond à une machine qui a effectué un nombre très important de requêtes DNS. Au dessus d'un certain seuil, la source est considérée comme représentant un proxy et elle est ajoutée à une liste d'identifiants exclus de l'analyse.

E. Construction des communautés:

Cette étape de détection sert à identifier les communautés d'identifiants appartenant au même botnet. Nous utilisons en entrée la liste des identifiants considérés comme probablement infectés, minorée des serveurs proxy retirés par le filtre précédent.

Les machines infectées d'un même botnet vont générer des erreurs de résolution DNS vers le même ensemble de noms de domaine.

Pour comparer le trafic généré par chaque identifiant, il suffit de comparer les filtres de Bloom associés. Si deux identifiants sont représentés par des filtres de Bloom équivalents (mêmes fonctions de hachages et même taille de vecteur), alors leurs vecteurs seront identiques.

Cependant, dans la majorité des cas et pour un ensemble d'identifiants infectés par le même botnet, le recoupement entre ces erreurs de résolution n'est que partiel. Ceci s'explique par l'absence de synchronisation globale entre les machines infectées, ainsi que par une composante aléatoire souvent incorporée dans les algorithmes DGA. De plus, d'autres erreurs de résolution DNS viennent se mêler aux requêtes des bots : logiciels mal configurés, comportement normal des utilisateurs.

Afin de regrouper les identifiants appartenant au même botnet, et qui n'ont pas forcement fait des requêtes vers les mêmes noms de domaine, nous proposons une méthode de comparaison des vecteurs des filtres de Bloom.

Pour cela nous allons comparer entre eux chaque élément de l'ensemble contenant les identifiants potentiellement infectés. Si le taux de ressemblance entre les filtres de Bloom de deux éléments atteint un certain seuil (80%), les deux éléments sont fusionnés, et ils verront leurs filtres de Bloom fusionnés également. L'opération de comparaison et de fusion est répétée pour tous les éléments de l'ensemble.

Le taux de ressemblance entre deux filtres de Bloom correspond au résultat de l'opération de division entre le max des taux de remplissage des filtres entrants et le taux de remplissage du vecteur résultant du ET logique entre les deux vecteurs. La valeur du taux de remplissage se rapproche de 100% si les deux vecteurs sont équivalents.

La comparaison entre les filtres de Bloom s'effectue en parallèle à l'opération de remplissage de ces derniers à cause du temps de calcul nécessaire pour comparer les filtres des identifiants suspects.

Pour fusionner les filtres de Bloom de deux éléments, nous faisons un ET logique entre tous les bits des vecteurs de ces derniers, ce qui se rapproche d'une intersection entre les deux ensembles représentés. Le vecteur résultant correspond à un nouveau filtre de Bloom qui sera associé avec l'union des identifiants des vecteurs d'entrée.

A la fin de l'opération de comparaison, chaque macroélément ainsi constitué va correspondre à une communauté. Si une communauté contient un nombre suffisant d'adresses IP, cette communauté sera envoyée vers l'étape suivante de l'analyse.

F. *Détection des C&C*

Dans cette phase, après la construction d'une communauté contenant des identifiants d'un même botnet, notre objectif est de détecter les noms de domaine associés au C&C. Cette étape de l'analyse ne s'appuie donc plus sur le trafic DNS en erreur (NX), mais sur le trafic DNS aboutissant à une résolution (NOERROR).

On analyse ainsi les réponses DNS réussies des membres d'une communauté. Un deuxième filtre de Bloom est ainsi rattaché à chaque identifiant concerné de manière à stocker cette information tout en préservant l'anonymat des données.

La taille des filtres de Bloom de cette étape est supérieure à celle utilisée dans les réponses NX. Ceci se justifie par le fait que le nombre d'identifiants concernés par cette étape est très réduit, et aussi parce que, dans cette étape, les filtres de Bloom sont utilisés pour représenter les noms de domaine des réponses NOERROR, dont le nombre est supérieur à celui des réponses DNS NX.

Afin d'éviter les collisions nombreuses lorsque le filtre de Bloom est rempli, les vecteurs des filtres de Bloom sont réinitialisés s'ils sont remplis à plus de 70%.

Pour chaque réponse DNS NOERROR, le nom de domaine est ajouté au filtre de Bloom de l'identifiant qui a effectué la requête. Un test d'appartenance est effectué pour les autres membres de la communauté. Lorsqu'un nom de domaine est présent dans la majorité des filtres de Bloom des membres d'une communauté alors ce dernier est extrait pour analyse.

Le nom de domaine identifié dans cette étape peut correspondre à trois cas de figure :

(i) le serveur de contrôle et commande du botnet ;

(ii) un nom de domaine malveillant en relation avec les activités malveillantes du botnet (click fraud, spams, etc.) ;

(iii) un nom de domaine populaire (Facebook, Google, …).

En limitant la phase de détection des C&C à un intervalle de temps court, la probabilité que des noms de domaine légitimes mais non populaires apparaissent à cette étape est quasi-nulle. En revanche, la probabilité de retrouver les noms de domaine populaires à ce stade est élevée car ils présentent le même phénomène de convergence que pour les C&C.

G. *Filtrage des résultats*

Les noms de domaine identifiés dans l'étape précédente sont comparés avec la liste des noms de domaine les plus populaires fournis par Alexa [13]. Ceci nous permet de filtrer les noms de domaine qui sont sollicités de manière légitime par l'ensemble d'une communauté.

Finalement, les noms de domaine en sortie de cette étape sont caractérisés comme étant malveillants.

VI. RESULTATS

Pour évaluer notre approche de détection de botnets de type domain-flux, nous avons appliqué notre algorithme sur un flux DNS anonymisé d'une heure de trafic représentative d'un réseau à large échelle, et représentant de l'ordre de 600 millions de requêtes et réponses DNS.

Dans notre trace de trafic, les réponses NX représentent 15% du nombre de réponses DNS. Dans ces 15%, seules 41% des réponses sont associées à des requêtes syntaxiquement valides. Ainsi, seulement 3% du trafic total est utilisé pour construire les communautés de bots.

Pour avoir des communautés de bots homogènes, nous avons besoins d'éviter un taux de collision élevé au niveau des filtres de Bloom. Ceci survient lorsque le taux de remplissage des filtres est élevé. Pour éviter cela, on exclue de l'analyse les filtres de Bloom dont le taux de remplissage atteint un certain seuil. Afin d'évaluer ce seuil, nous avons fait varier le nombre de faux positifs détecté en fonction de ce seuil (voir figure 3).

On remarque que jusqu'à un seuil de 90%, le nombre de faux positifs ne dépasse pas 3, ces derniers étant filtrés lors de la dernière étape puisque leur TLD fait partie des 100 noms de domaine les plus populaires. Lorsque le taux de remplissage s'approche de 100%, des communautés de serveurs proxys sont interprétées comme étant des bots et le nombre de faux positifs évolue alors de manière exponentielle.

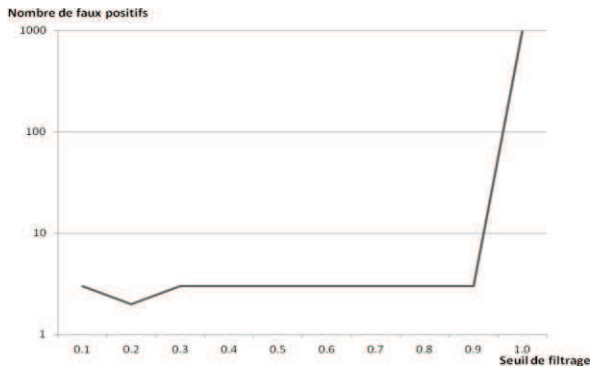

Figure 3. Nombre de faux positifs par rapport au seuil de filtrage des proxys

Lors de notre analyse, les premiers domaines malveillants sont détectés à partir de 15 minutes d'analyse. Sur l'échantillon considéré, nous avons détecté un total de 345 domaines malveillants (voir tableau 1). Parmi ceux-ci, 305 sont des domaines générés par DGA. Ces derniers correspondent à des serveurs de contrôle et commandes de botnets, ou à des serveurs utilisés dans d'autres activités malveillantes.

Nous avons aussi détecté 10 de noms de domaine liés au botnet Bagle [15] qui n'utilise pas de DGA, mais contacte plusieurs noms de domaine pour fonctionner. Cette détection est due au fait que plusieurs des domaines concernés n'étaient pas accessibles, ce qui aboutit à un comportement comparable à celui d'un botnet de type domain-flux.

30 noms de domaine ne contenant que des annonces publicitaires ont été détectées par notre approche, et correspondent après analyse à une activité de fraude au click.

Classes de noms de domaine	Nombre
DGA	305
Bagle botnet	10
Fraude au click	30

Tableau 1. Classification des noms de domaine détectés

Le nombre de noms de domaine détectés dans un échantillon d'une heure est encourageant. En effet, notre approche se base sur le phénomène de génération synchrone de noms de domaine par les membres d'un même botnet, et cette génération peut se produire moins d'une fois par jour.

Il est difficile de mesurer le taux de faux négatifs de notre approche de détection dans un échantillon de trafic réel non qualifié d'une heure étant donné que notre approche ne détecte que les botnets de type domain-flux ou ayant un comportement similaire.

Nous avons mesuré les performances de notre approche de détection lors de l'analyse des traces DNS anonymisées. Elle permet le traitement moyen de 130 000 paquets DNS par seconde. Cette valeur se situe dans un ordre de grandeur acceptable pour un réseau opérateur, et est donc encourageante pour une première implémentation.

Il est important de noter que par défaut le nombre de paquets DNS traités par seconde décroit progressivement en fonction du taux de remplissage des filtres de Bloom. Pour éviter cela, un mécanisme de rafraichissement périodique de ces filtres a été ajouté, et permet de réinitialiser périodiquement les filtres de Bloom. Ce mécanisme permet également de s'assurer que ces filtres représentent uniquement les dernières requêtes.

De plus, un deuxième mécanisme permet de supprimer l'ensemble des informations relatives à un identifiant après une durée fixée.

Nous avons fait varier ces deux paramètres et avons pu constater que la durée optimale de ces cycles de rafraichissement était de 45 minutes pour les filtres de Bloom et de 20 minutes pour la conservation d'un identifiant inactif. En dessous de ces valeurs, le nombre de domaines détectés est impacté. Au dessus de ces valeurs, la détection n'est pas améliorée mais les performances se dégradent.

Ce mécanisme de rafraichissement permet par ailleurs de s'affranchir de tout stockage sur disque en conservant l'ensemble des données en mémoire. De plus, il nous permet de nous assurer qu'aucune information relative à un identifiant inactif ne sera conservée au delà de ce délai de 20 minutes.

VII. CONCLUSIONS

Nous avons présenté une nouvelle approche de détection de botnets de type domain-flux visant à répondre aux problématiques de charge et de confidentialité d'un réseau à large échelle. Nous avons testé notre approche avec un trafic anonymisé d'une heure, qui nous a permis de valider cette approche avec la détection de 345 noms de domaine liés à des activités malveillantes.

Notre approche respecte les contraintes de volumétrie et de respect de la confidentialité d'un réseau à large échelle.

Les travaux futurs se concentrent actuellement sur les possibilités d'adapter cette méthode à un environnement distribué, ainsi qu'à l'amélioration des différentes méthodes de filtrage de résultats pour qu'elles ne soient plus dépendantes de listes blanches.

VIII. REFERENCES

[1] Zeus Tracker. https://zeustracker.abuse.ch/faq.php.

[2] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your Botnet is My Botnet: Analysis of a Botnet Takeover," in Conference on Computer and Communication Security (CCS), 2009.

[3] F. Leder and T. Werner, "Know Your Enemy: Containing Conficker," https://www. honeynet.org/papers/conficker/, April 2009.

[4] J. Nazario and T. Holz, "As the Net Churns: Fast-Flux Botnet Observations," Proc. Int'l Conf. Malicious and Unwanted Software (Malware), IEEE Press, 2008, pp. 24–31.

[5] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," Communications of the ACM, 13(7):422–426, July 1970.

[6] L. Qiu, Y. Li, X.Wu, , "Preserving privacy in association rule mining with bloom filters," J. Intell. Inf. Syst. 29(3), 253–278 (2007).

[7] A. Broder and M. Mitzenmacher, "Network applications of Bloom filters: A survey," Internet Mathematics, vol. 1, no. 4, pp. 485–509, 2004.

[8] L. Bilge, E. Kirda, C. Kruegel, and M. Balduzzi, "Exposure : Finding malicious domains using passive dns analysis," in Symposium on Network and Distributed System Security (NDSS), 2011.

[9] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster, "Building a Dynamic Reputation System for DNS," in Proc. USENIX Security Symposium, 2010, pp.273-290.

[10] S. Yadav, A. K. K. Reddy, A. N. Reddy, and S. Ranjan, "Detecting algorithmically generated malicious domain names," in Proceedings of the 10th annual conference on Internet measurement, ser. IMC '10. New York, NY, USA: ACM, 2010, pp. 48–61.

[11] S. Yadav, A. K. K. Reddy, "Winning with DNS Failures: Strategies for Faster Botnet Detection," in 7th International ICST Conference on Security and Privacy in Communication Networks (SecureComm), London, United Kingdom, September 2011.

[12] N. Jiang, J. Cao, Y. Jin, L. E. Li, and Z.-l. Zhang, "Identifying suspicious activities through DNS failure graph analysis," in IEEE International Conference on Network Protocols (IEEE ICNP'10), 2010, pp. 144–153.

[13] N. Jain, M. Dahlin, and R. Tewari (2005), "Using Bloom Filters to Refine Web Search Results," Proceedings of the Eight International Workshop on the Web & Databases, pp. 25-30.

[14] Alexa Internet Inc. http://www.alexa.com.

[15] M86 Security Labs, "A little Spam With Your Bagle?," http://www.m86security.com/labs/i/A-Little-Spam-With-Your-Bagle-,trace.999~.asp, June 2009.

[16] Malware Domain List. http://www.malwaredomainlist.com/mdl.php