TLS-SEC / Master SSIR

Detection d'un botnet par l'analyse des données de requête DNS

Rapport du projet long

Omar ANSER | Yacine ANSER

Résumé

Le domain name system (DNS) est l'un des protocoles les plus utilisés dans l'Internet. L'objectif principal de ce protocole est de traduire les noms de domaine (ND) en adresse IP. Malheureusement, de nombreux cybercriminels déploient ce protocole à des fins malveillantes, par exemple pour permettre de récupérer le nom de domaine d'un C&C serveur (Command And Control) ou la communication avec un botnet. Dans le premier cas, la solution la plus simple serait de blacklister les noms de domaine suspects sachant qu'un botnet peut se défendre en utilisant plusieurs adresses IP pour un seul nom de domaine ou en implémentant un algorithme de génération de nom de domaine (DGA). Dans le second cas, les Botmasters établissent un tunnel de communication entre le C&C et les machines infectées grâce à des requêtes et des réponses DNS. Il est possible de réaliser une approche efficace pour la détection d'un botnet utilisant ce type de communication en analysant les données de requête DNS présente au sein d'un réseau. La première partie de ce rapport traitera sur la détection d'un botnet lors de sa résolution de nom appliquant les méthodes de défenses énoncés. La deuxième abordera la détection d'un tunnel de communication basé sur le protocole DNS.

Table des matières

1 Introduction

Les cyberattaques, y compris l'injection de malware, n'ont jamais cessé de menacer les ordinateurs des réseaux et les systèmes d'information. L'une des formes prédominantes infectent les systèmes avec des programmes malveillants les forçant à agir comme des botnets.

Le Domain Name System (DNS) est l'équivalent d'un annuaire téléphonique fournissant un mécanisme pour nommer les ressources de telle sorte que les noms soient utilisables dans différents hôtes, réseaux, familles de protocoles et organismes administratifs [1]. En d'autres termes, le DNS traduit les noms de domaine plus facilement mémorisables en adresses IP numériques nécessaires pour localiser et identifier les dispositifs et services informatiques.

Un botnet, ou réseau de robots, est un ensemble d'ordinateurs connectés à Internet (bots) qui sont infectés par un programme malveillant spécifique afin d'être contrôlés à distance par un serveur de commande et de contrôle (C&C).

Un botnet peut être utilisé contre tout appareil connecté à l'Internet pour exécuter un large éventail d'actions malveillantes. Ces actions comprennent le lancement d'un déni de service distribué (DDoS), le phishing, la génération et l'envoi de messages de spam, la propagation de logiciels malveillants.

Le protocole DNS peut être utilisé de manière illicite à différents stades du processus de communication avec le botnet, tels que la recherche du serveur C&C, la transmission de données, et/ou le contrôle des bots. Au stade initial de la communication avec le botnet, un bot essaie de trouver son C&C en envoyant des requêtes DNS pour résoudre le nom de domaine du serveur. Après l'avoir résolu, la communication entre le C&C et le bot peut commencer. Des méthodes de détection existent [2], mais les botnets modernes peuvent mettre en œuvre certaines techniques pour les contourner.

Domain Generation Algorithm (DGA), Fast-Flux Service Network (FFSN), Domain Flux, Double Flux, et le DNS tunneling [3] sont les techniques les plus utilisées par les botnets modernes pour contourner les méthodes de détection et ainsi pouvoir communiquer avec chaque bot.

Grace à l'algorithme DGA, un grand nombre de noms de domaine avec une courte durée de vie sont générés, et avec le DNS tunneling, les communications des bots avec le serveur de C&C peuvent être **enveloppées** et **tunnelées** par le biais de paquets DNS.

L'objectif principal de ce projet est de concevoir et mettre en œuvre une technique de détection efficace de botnets basés sur le DNS et s'appuyant sur la détection d'anomalies en sein d'un réseau. Le projet est en deux parties. La première partie implémente une solution pour surveiller une activité de groupe dans le trafic DNS et ainsi déduire la présence d'un botnet même si ce dernier utilise un DGA. La deuxième partie met en œuvre un mécanisme basé sur les signatures pour détecter le DNS tunneling. La section 2 propose un état de l'art non-exhaustif sur le sujet. La section 3 décrit la méthodologie suivi pour réaliser notre solution. Par la suite, dans la section 4, nous présenterons plus en détails notre solution en analysant les résultats obtenus. Enfin la section 5 conclura ce rapport.

2 Etat de l'art

2.1 Premier partie du projet

Pour la détection des botnets, différentes architectures et techniques ont été proposées. En outre, les chercheurs ont réalisé différentes classifications pour mieux comprendre la structure d'un botnet [4] [5] [6] [7]. Les techniques de détection des botnets ont été principalement classées en deux types d'approches : celles qui sont basées sur l'installation et la configuration d'un honeynet au sein du réseau surveillé et le système de détection des intrusions (IDS) [8] [9] [10] [11]. La figure 1 illustre la classification des techniques de détection des botnets basées sur l'analyse du trafic DNS.

Weimer [12] a proposé un système passif qui collecte les noms de domaine à partir du trafic DNS et les stocke dans une base de données pour les analyser et déduire des comportements malveillants. De même, Zdrnja et ses collaborateurs [13] ont appliqué le concept de surveillance passive pour détecter les anomalies du trafic DNS. Dans leur travail, les auteurs ont contesté la possibilité de différencier les comportements DNS inhabituels ou anormaux des comportements DNS légitimes. Toutefois, les auteurs n'ont pas mentionné les caractéristiques du DNS qui doivent être capturées et stockées [13]. La détection des botnets basée sur l'analyse du trafic DNS a le potentiel de repérer des botnets du monde réel sans une connaissance préalable de leurs protocoles de communication et de leurs structures [14].

Notre solution se rapproche d'une détection passive utilisant le clustering (Figure 1). Ainsi, Nous ne détaillerons pas dans ce rapport l'état de l'art des autres méthodes de détection.

La détection des botnets basée sur le clustering, consiste à regrouper un ensemble de **nœuds** en fonction de certains paramètres (par exemple les caractéristiques du trafic DNS) de manière à ce que les nœuds groupés au sein d'un même groupe expriment une certaine similarité qu'avec ceux des autres clusters à un niveau qui permet la conclusion de l'existence d'un botnet.

Dans ce cadre, Perdisci et al.[15] ont proposé une approche de détection d'anomalie pour détecter un fast-flux (FFSN). Leur méthode s'appuie sur l'analyse passive des traces de requêtes DNS récursives (RDNS) obtenues sur un réseau important. Toutefois, la portée de cette solution est limitée aux botnets qui génèrent des spam-emails et adoptent la technologie fast-flux pour récupérer l'adresse IP du serveur C&C [16].

BotGad (Botnet Group Activity Detector) est une solution de détection qui a été proposée par Choi et al [17]. Leur méthode se base sur le fait qu'un botnet a comme caractéristique particulière d'agir dans la majorité des cas en groupe. Ainsi, BotGAD extrait depuis le réseau surveillé certaines caractéristiques du trafic DNS pour distinguer les requêtes légitimes et illégitimes qui pourraient faire partie du trafic du botnet. Selon les auteurs, le trafic illégitime apparaît comme un groupe d'hôtes présentant le même comportement. Par exemple, le bot essaie de rechercher un serveur C&C. Ce comportement apparaîtra comme un groupe d'hôtes essayant de récupérer certains noms de domaine à différents intervalles de temps. L'inconvénient de cette approche est son incapacité à identifier les botnet qui utilisent la technologie fast-flux et peut produire un nombre élevé de faux positifs [18].

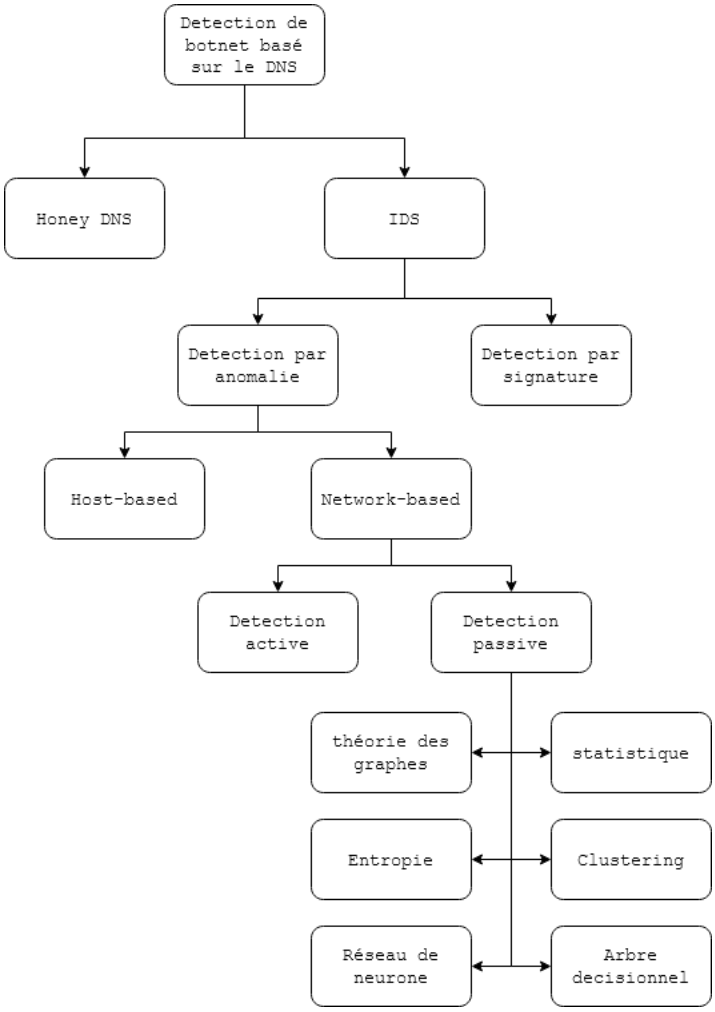

FIGURE 1 – Classification des techniques de détection des botnets basées sur les caractéristiques du trafic DNS.

2.2 Deuxième partie du projet

De nombreux papiers scientifiques ont abordé la question de l'extraction de caractéristiques pour le DNS tunneling. Celles ci permettraient la détection d'un botnet utilisant un tunnel au sein d'un réseau local. L'étude la plus significatif est celle réalisé par [19].Les auteurs ont présenté une approche statistique pour établir le profil d'un tunnel DNS. La méthode proposée vise à utiliser un mécanisme d'extraction de `features` en exploitant des caractéristiques telles que le temps et le type d'enregistrement des messages DNS. Les auteurs ont utilisé un trafic réseau simulé avec l'outil **dns2tcp** afin de saisir ces caractéristiques. Une analyse est faite sur la base des statistiques collectées à partir du trafic réseau. D'autres auteurs [20] ont proposé une approche statistique qui utilise le **Machine Learning** afin de détecter la présence du tunnel DNS. Les auteurs ont utilisé

des caractéristiques statistiques liées à l'aspect du trafic, telles que les suites de requête et réponse pour maintenir la connexion . En utilisant une méthode de classification basée sur des règles, les auteurs ont démontré l'efficacité de la méthode qu'ils proposent pour la détection des tunnels.

De façon similaire, d'autres auteurs [21] ont proposé une solution qui utilise aussi le machine learning, cette technique vise à identifier le tunnel DNS à l'aide de caractéristiques se trouvant dans la couche Application de la couche OSI. La technique proposée vise à exploiter des caractéristiques qui distinguent le comportement légitime du DNS au sein d'un réseau et le comportement illégitimes. Ces caractéristiques sont construites statistiquement à partir des sessions TCP et du trafic réseau, on y trouve la taille des paquets et le temps entre les requêtes DNS et les résponses. Par la suite, un arbre de décision est utilisé pour la phase d'apprentissage. Au terme de la phase d'apprentissage, le classificateur est testé à l'aide d'une série de requêtes DNS afin d'évaluer ses performances.

Homem et al [22] ont proposé un système de détection fondé sur la technique de classification de l'entropie. Ils ont examiné la structure interne des paquets encapsulés dans les paquet DNS [13]. Ainsi, les auteurs proposent une technique qui utilise la distribution moyenne de l'entropie.

3 Méthodologie

3.1 Introduction

La mise en œuvre d'une approche globale et complète pour la détection des botnets est une tâche non trivial. Les botnets utilisent différents protocoles, différentes architectures et peuvent échapper aux méthodes de détection de différentes manières. Dans ce qui suit nous allons vous présentez la méthodologie suivit lors de notre projet.

3.2 Conception de l'étude

Afin de réaliser nos expérimentations, nous avons d'abord développé une plateforme d'expérimentation de botnet, cette plateforme est composée d'un émulateur et d'un simulateur, l'émulateur va créer l'architecture réseau qui correspond à celle se trouvant dans la figure 3 , quand au simulateur, il va simuler le trafic réseau à l'intérieur de l'émulateur. Ces deux composants nous aideront à créer un ensemble de données (Dataset). cela représente une partie critique dans notre projet car le dataset doit être réaliste ce qui n'est pas une tâche facile.

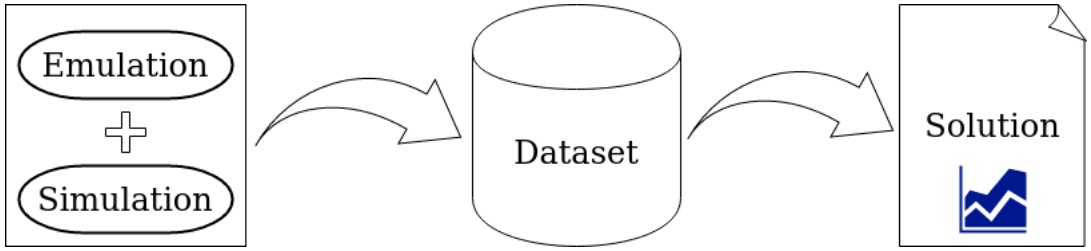

FIGURE 2 – Démarche suivie

La période de simulation doit être suffisamment longue pour permettre aux bots **d'imprégner** les traces du réseau de leurs caractéristiques. Lorsque nous supposons que nous disposons d'un bon dataset, nous appliquons notre solution.

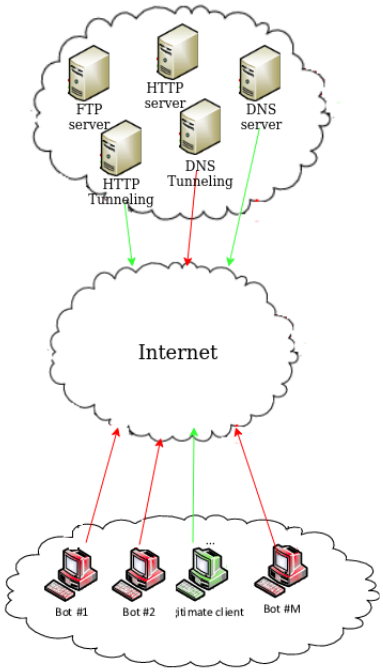

FIGURE 3 – Architecture réseau

3.3 Génération du dataset

3.3.1 Emulbot

Emulbot est émulateur de botnets en temps réél entièrement virtualisé avec Docker qui est un logciel libre permettant de lancer des applications dans des conteneurs logiciels. Emulbot offre la possibilité de tester un botnet de le vie réelle sous différentes propriétés réseau. L'emulateut est codé en Python et utilise l'API Docker. Il est simple d'utilisation et permet de générer un dataset réaliste sans avoir à utiliserr un composant matériel supplémentaire. Dans notre cas, nous avons pu

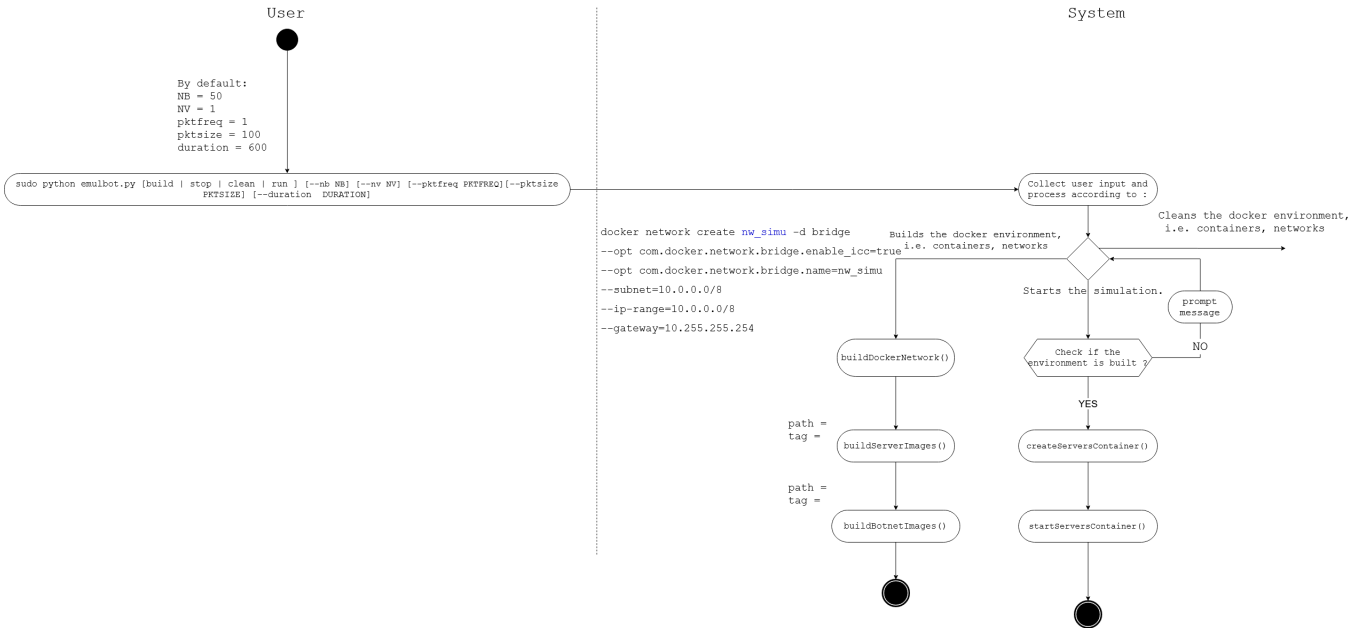

FIGURE 4 – Diagramme d'activité Emulbot

lancer 300 bots sans aucune difficulté. La Figure 4 présente son diagramme d'activité. Emulbot se concentre principalement sur le canal de communication entre les bots et le C&C. Il ignore les autres serveurs intermédiaires (Reporter, Loader...).

Lien Github vers emulbot

3.3.2 Simulateur

Afin de simuler le trafic au sein d'Emulbot ils nous a fallu développer un Scheduler. Ce Scheduler prend en paramètre : une liste de requêtes, il peut s'agir de requêtes HTTP ou FTP, on attribue à chaque requête une probabilité. il prend aussi une liste de machines (Adresses IPs). Ainsi le Scheduler fonctionnera comme cela, il prendra une requête de la liste des requêtes et prend une machine au hasard dans la file d'attente. La machine choisi exécutera la requête. Nous répétons cela suffisamment longtemps pour avoir un bon Dataset. Nous estimons qu'avec cette méthode, nous avons un trafic réaliste et représentatif de la réalité et nous évitons la famine.

3.3.3 Dataset

Nous avons décidé de stocker le dataset dans une base de donnée afin de faliciter la manipulation des données et de réduire la complexité de calcul. Nous avons utilisé pour cela Mongodb qui est une base de donnée NoSQL.

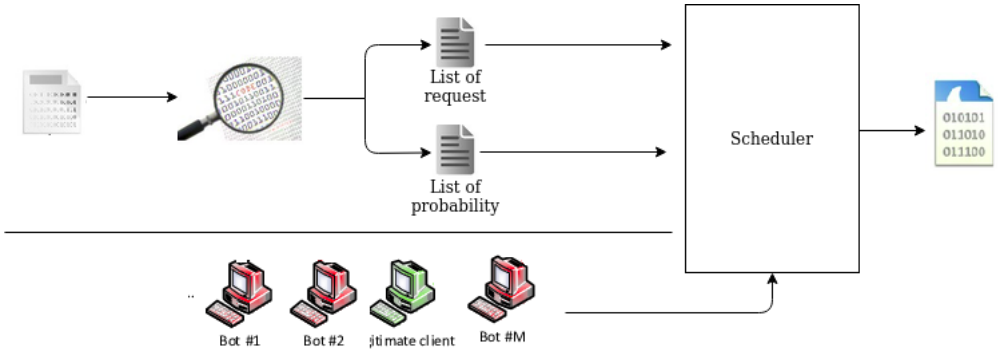

FIGURE 5 – Simulateur

4 Méthode de detection d'un botnet par groupe d'activité basé sur le protocole DNS

4.1 Caractéristiques du DNS dans un botnet

Comme mentionné précédemment, les hôtes infectés accèdent automatiquement au serveur C&C avec son nom de domaine. Par conséquent, une requête DNS RR (resource record) est utilisée. Une telle requête apparaît également dans d'autres situations. Les 4 cas suivants montrent les situations dans lesquelles les requêtes DNS sont utilisées dans un botnet. (1) Lors de la procédure de ralliement : les hôtes infectés doivent se regrouper. Le botmaster veut que son botnet soit invisible et portable, c'est pourquoi les bots utilisent le DNS pour se retrouver. (2) Lors des comportements malveillants d'un botnet : Plusieurs types d'activités malveillantes telles que les attaques DDoS et les envois de spam sont accompagnés de requêtes DNS. (3) Lors de la migration du serveur C&C : le botnet peut migrer d'un serveur C&C à l'autre. À ce moment, le DNS est également utilisé. (4) Dans le cas où le serveur C&C change d'adresse IP : Si un serveur C&C utilise un DHCP (Dynamic Allocation IP), l'adresse IP correspondante peut être modifiée à tout moment et un botmaster peut également changer l'adresse IP du serveur C&C de manière intentionnelle. Les bots envoient donc des requêtes DNS pour récuprer la nouvelle adresse IP du serveur C&C.

Les requêtes DNS d'un botnet peuvent être distinguable des requêtes DNS légitimes en se basant sur les caractéristiques uniques des requêtes DNS d'un botnet. Le tableau 1 montre quelques différences entre les requêtes DNS d'un botnets et les requêtes DNS légitimes sur un réseau.

Premièrement, seuls les membres du botnet envoient des requêtes avec le nom de domaine du serveur C&C dans la query du paquet DNS (taille fixe), les utilisateurs légitimes n'ont pas la connaissance de ce nom. Par conséquent, le nombre d'adresses IP qui questionne le domaine d'un botnet est normalement fixe. Deuxièmement, les membres fixes d'un botnet agissent et migrent ensemble en même temps. L'activité de groupe d'un botnet est une propriété propre à lui et indissociable.

	Adresses IPs sources des requête DNS	Modèles d'activité	Type du DNS
DNS d'un botnet	Taille fixe	Activité en groupe à un temps spécifique	DDNS
DNS légitime	Anonymes (utilisateurs légitimes)	Aucune activité en groupe. Random et en continue	DNS

TABLE 1 – Différences entre `botnet` et DNS légitime

La plupart des requêtes DNS légitimes sont cependant continues et discontinu. Troisièmement, un botnet utilise généralement le DDNS comme type de DNS alors qu'un utilisateur légitime utilise du DNS classique.

4.2 Méthode de détection

On s'inspirant de notre état de l'art, Nous avons développé une méthode de détection de botnet basée sur les requêtes DNS en s'aidant de l'analyse expliquée dans la sous section précédente.

La méthode est divisée en 3 algorithmes : Insert-DNQ-Query, Delete-DNS-Query, et Detect-BotDNS-Query. La Figure 6 montre le premier algorithme. Une base de données est employé pour stocker les données des requêtes DNS est comprend l'adresse IP source de la requête, le nom de domaine de la requête et le timestamp de la requête envoyé. Nous regroupons les données DNS par nom de domaine et timestamp. Ci dessous un pseudo code de l'algorithme Insert-DNS-Query.

```
Inset-DNS-Query(Qt) // Qt = requête DNS entre deux timestamp t-1 et t
{
  A; // Tableau de Q
  ND; // Noms de domaine de Qt
  if (ND is not in A){
    insert(DN,A);
    IP, IPList = Adresse IP de Qt, liste des adresses IP du ND
    if (IP is not in IPList){
        insert(IP, IPList)
        cnt = len(IPList)
        cnt ++
    } } }
```

L'algorithme Delete-DNS-Query permet de supprimer un nom de domaine redondant. Si la taille de la liste IP ne dépasse pas un seuil fixé ou si le nom de domaine est légitime et qu'il existe déjà dans une Whitelist, ce nom de domaine ne doit pas être traité. Cet étape est indispensable afin de réduire la complexité de calcul et économiser de la mémoire. Finalement, l'algorithme Detect-BotDNS-Query détecte la présence d'un botnet. La Figure 7 présente son principe. Nous définissons

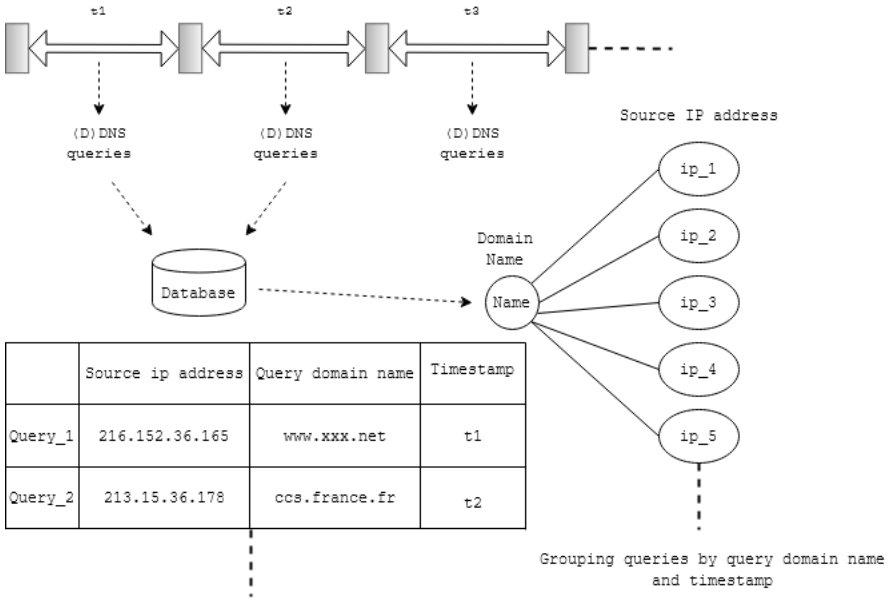

FIGURE 6 – Insert-DNS-Query

et calculons la valeur numérique de **l'activité de groupe du DNS** d'un botnet, appelée similarité. S'il existe deux listes d'adresses IP qui sont sollicitées aux temps t1 et t2 et qui possède le même nom de domaine, supposons que la taille des deux listes soit égale à respectivement A et B, et que la taille des adresses IP dupliquées entre eux soit égale à C. Nous calculons la similarité S des deux listes avec cette simple formule :

$$S = (\frac{C}{A} + \frac{C}{B}) \times \frac{1}{2}$$

Si A = 0 ou B = 0, alors nous définissons la similarité comme -1. Si la similarité est proche de 0, il faut mettre le nom de domaine dans la Whitelist et supprimer la liste d'adresses IP du domaine. Si la valeur de la similarité est proche de 1 alors le nom de domaine de la requête peut être celui du C&C. Il faut donc l'insérer dans la blacklist et le surveiller après cette période.

4.3 Résultats

Afin d'appliquer la méthode décrite ci-dessus, nous avons décidé de lancer Emulbot avec son simulateur sur une période de 2 voir 3 heures avec 4 à 6 timestamps. à un t choisi, les bots envoient automatiquement des requêtes DNS pour récupérer le nom de domaine du C&C. Cette action peut être effectué autant de fois souhaiter durant une simulation. L'algorithme Insert-DNS-Query créé en temps réel le dataset. la détection se fait en appliquant le troisième algorithme sur le dataset après sa création. La Figure 8 présente le résultat le plus pertinent que nous avons eu.

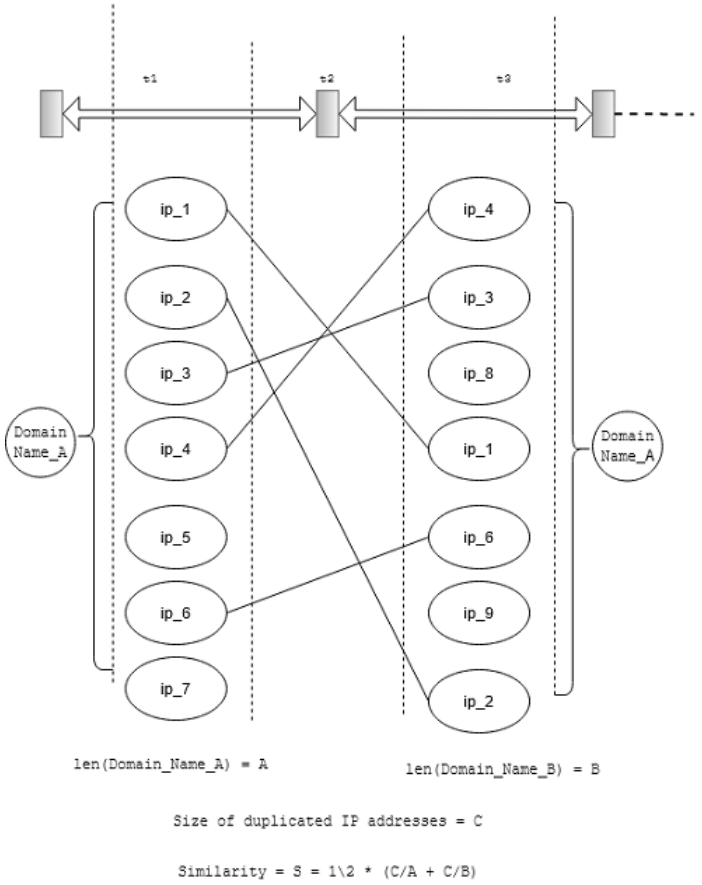

FIGURE 7 – Detect-BotDNS-Query

Ce graphe présente la valeur de S (la similarité) pour chaque nom de domaine présent dans Emulbot incluant le nom de domaine du C&C. Nous remarquons qu'il y a un nombre élevé de nom de domaine dont la valeur de silimarité est en dessous de 0.4, ces valeurs sont en réalité biaisés et devraient se rapprocher de 0. Ce manque de précision est dû au mauvais choix du seuil dans l'algorithme Delete-DNS-Query. Ainsi, la plupart des valeurs de S (80%) sont, dans une situation plus réaliste, égales à 0 ou -1. Par contre il est clair que, dans le cas où un groupe de bots souhaite récupérer le nom de domaine du C&C, la valeur de S est nettement plus important que les autres, ce qui est logique vu les caractéristiques du protocole DNS dans un botnet cités dans la partie qui précéde. Pour ce qui est des valeur qui dépasse 0.4, nous pouvons conclure qu'il s'agit d'un nom de domaine fréquemment utilisé par plusieurs groupes de machine à des intervalles de temps distincts au sein d'un réseau.

Le résultat présenté ci-dessous dépend fortement de la partie simulation d'Emulbot.

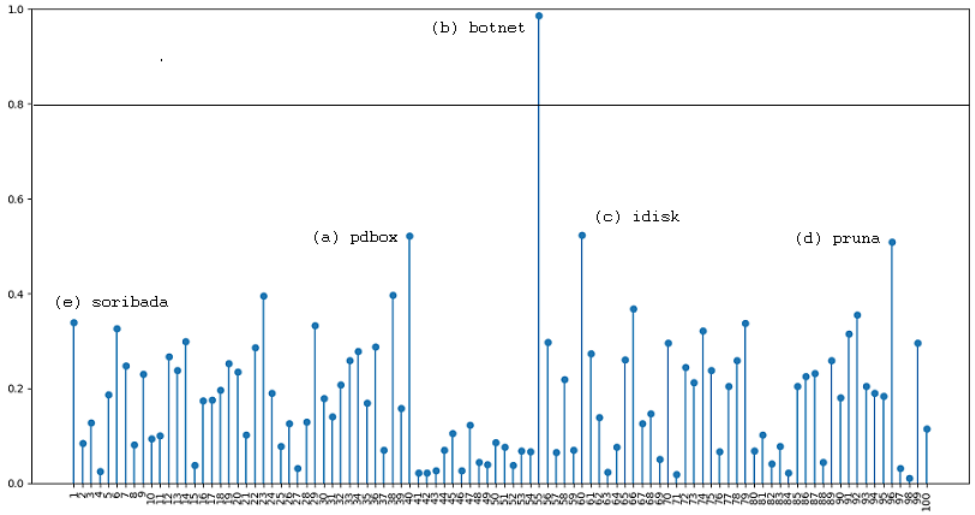

FIGURE 8 – Resultat de `Detect-BotDNS-Query(dataset)`

4.4 Conclusion

L'analyse du trafic DNS fait dans notre solution a le potentiel de repérer un botnet sans une connaissance préalable des protocoles de communication et de sa structure. La mise en place est simple et la création du dataset est inclue dans la solution, ce qui évite de recourir à une autre méthode de création de dataset. De plus aucune communication avec les bots ou les serveurs intermédiaires du botnet n'est nécessaire. Cette solution n'affecte en aucun cas l'activité d'un réseau. La complexité de calcul est réduite à son minimum.

Cette solution n'est toutefois applicable qu'après la génération du dataset, elle n'est donc pas en temps réel, et malgré notre volonté de réduire la complexité de calcul, son utilisation dans un réseau très important est problématique.

5 Méthode de détection d'un botnet utilisant le DNS tunneling

5.1 Introduction

Les botnets utilisent le DNS pour faire bien plus que de résoudre des noms de domaine, ils l'utilisent pour masquer leurs communications et pour envoyer et recevoir des commandes et des données en toute confiance . pour se faire le botmaster encapsule les commandes dans les paquet DNS. Notre objectif est donc d'essayer de détecter les communications entre les botmasters et les bots qui traversent ce tunnel DNS.

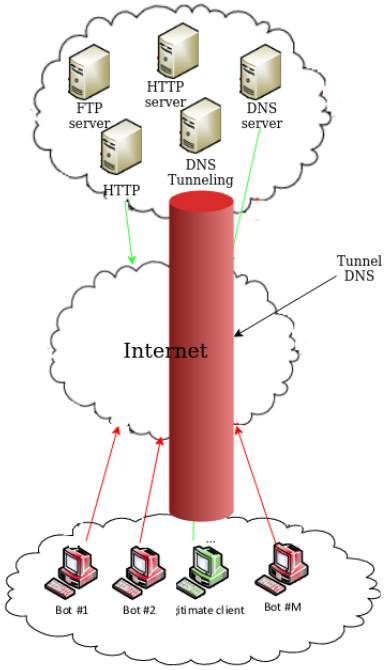

FIGURE 9 – DNS tunneling

Afin de mieux comprendre la nature du trafic DNS généré par le Tunnel, nous avons crée deux scénario, le premier légitime (sans bots) et le deuxième illégitime (avec bots).Notre hypothèse est que nous devrions voir une corrélation entre la fréquence et la quantité de paquets HTTP et DNS.

5.2 Profil du traffic DNS

Le standard DNS définit plus de 80 types d'enregistrement, les plus communs sont l'enregistrement A ; qui est utilisé pour trouver l'adresse IP du nom de domaine, l'enregistrement MX, l'enregistrement CNAM et l'enregistrement TXT, le plus utilisé dans le tunneling DNS est l'enregistrement TXT car ils offrent la structure de charge utile la plus grande et la plus diverse. La figure 8 représente une capture d'écran de notre botnet qui utilise le tunneling pour envoyer des commandes, nous pouvons voir qu'il utilise l'enregistrement TXT et envoye des requettes et des

réponses continuellement pour maintenir la communication.

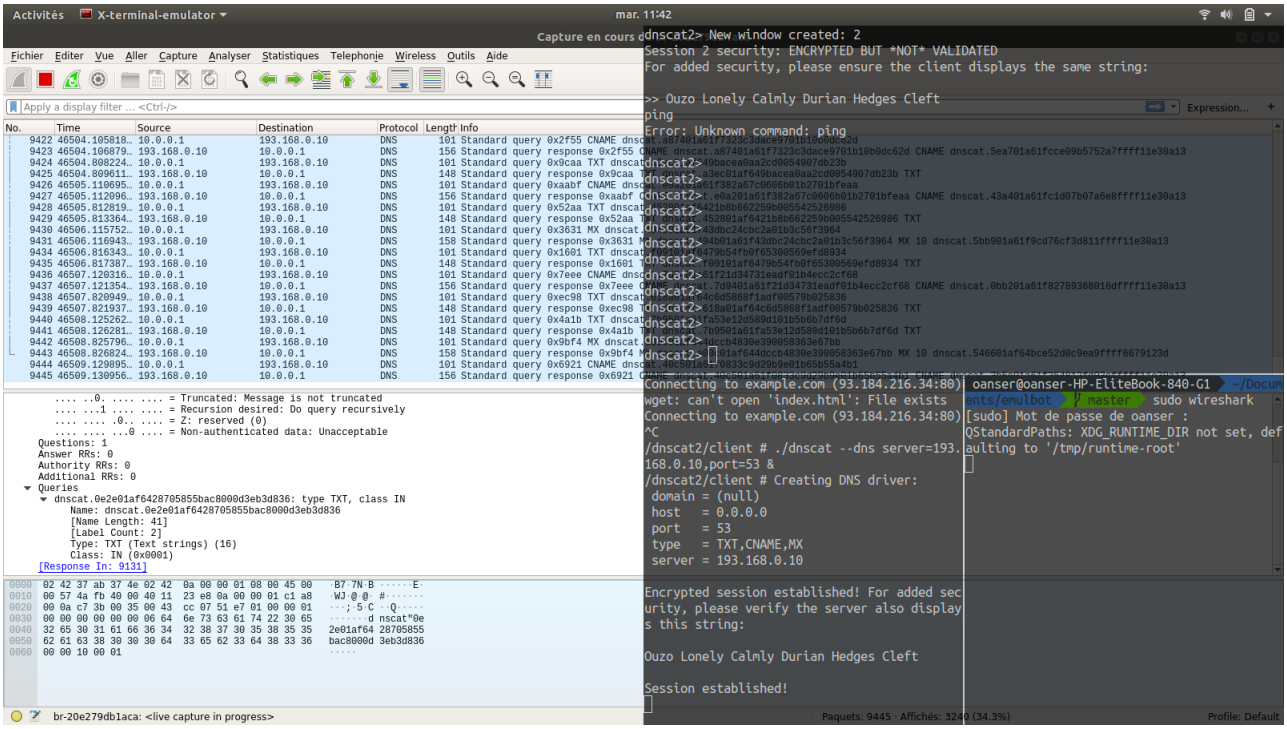

FIGURE 10 – Capture WireShark

5.3 Resultat

Nous simulons d'abord un trafic ne contenant pas de botnet qui utilise le tunnel DNS pour communiquer. Comme le montre la Figure 11, Nous pouvons voir une forte corrélation entre HTTP et DNS en terme de fréquence et de nombre de paquets. Cela est logique car les requêtes DNS sont formées lorsqu'un utilisateur génère des requêtes HTTP en naviguant sur le web.

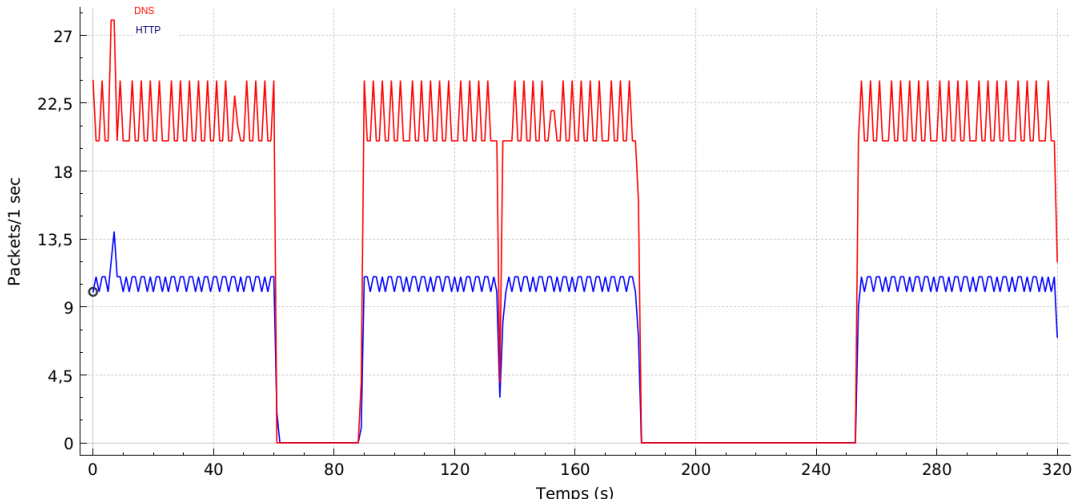

FIGURE 11 – Traffic DNS et HTTP 1

Dans notre deuxième simulation, nous avons établi un tunnel DNS entre le botmaster et les bots. Comme le DNS est basé sur UDP, qui contrairement à TCP est un protocole sans connexion, le botmaster et les bots doivent continuellement s'interroger l'un l'autre pour maintenir une connexion. Cela signifie que même si le tunnel DNS n'envoie pas de données ou de commande, nous devrions tout de même constater un niveau élevé d'activité DNS grâce aux interrogations fréquentes. La Figure 12 montre bien que même s'il n'y a pas de trafic HTTP, nous voyons quand même un certain trafic DNS. A partir de là, nous pouvons supposer qu'il y a un tunnel DNS.

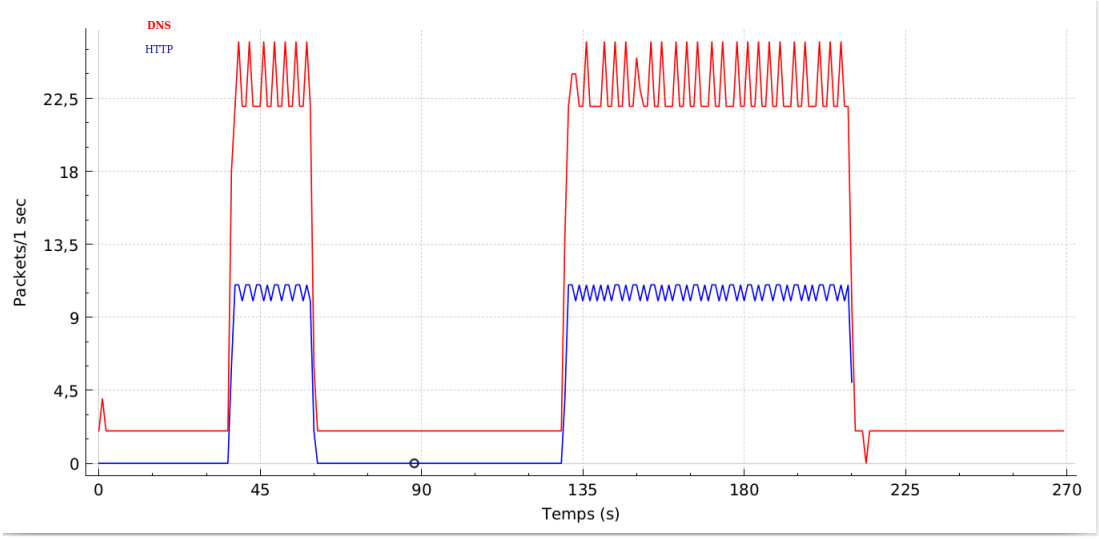

FIGURE 12 – Traffic DNS et HTTP 2

6 Consclusion

Notre travail a consisté à étudier la détection des botnets à l'aide des requêtes DNS, nous sommes parvenus à développer une plateforme d'expérimentation de botnets qui nous à permis de créer un Dataset. Par la suite, nous avons proposé deux méthodes, basé sur les requetes DNS, afin de détécter un botnet dans un réseau local. La premier solution est basé sur une caractéristique propre d'un botnet qui est de procéder dans la majorité des cas en groupe. La deuxiéme sur la correlation qui existe entre le traffic HTTP et le traffic DNS. Les solutions proposées ont permis de détecter les bots dans le contexte de l'étude. Des amélioration peuvent etre faite au niveau des deux méthodes comme par exemple réduire le temps de calcul de la premier solution ou bien ameliore notre simulateur afin d'avoir un meilleur dataset.

Références

[1] RFC 883

[2] A Survey of Botnet and Botnet Detection, Maryam Feily, Alireza Shahrestani et Sureswaran Ramadass ; 2009 Third International Conference on Emerging Security Information, Systems and Technologies

[3] Dietrich et al., 2011 ; Farnham & Atlasis, 2013 ; Lysenko, Pomorova, Savenko, Kryshchuk, & Bobrovnikova, 2015.

[4] Karim A, Salleh RB, Shiraz M, Shah SAA, Awan I, Anuar NB (2014) Botnet detection techniques : review, future trends, and issues. J Zhejiang Univ Sci C 15(11) :943–983

[5] Rodrı´guez-Go´mez RA, Macia´-Ferna´ndez G, Garcı´a-Teodoro P (2013) Survey and taxonomy of botnet research through lifecycle. ACM Comput Surv (CSUR) 45(4) :45

[6] Feily M, Shahrestani A, Ramadass S (2009) A survey of botnet and botnet detection. In : Third international conference on emerging security information, systems and technologies, 2009 (SECURWARE'09). IEEE, pp 268–273

[7] Silva SS, Silva RM, Pinto RC, Salles RM (2013) Botnets : a survey. Comput Netw 57(2) :378–403

[8] Zeidanloo HR, Shooshtari MJZ, Amoli PV, Safari M, Zamani M (2010) A taxonomy of botnet detection techniques. In : 2010 3rd IEEE international conference on computer science and information technology (ICCSIT). IEEE, pp 158–162

[9] Abdullah RS, Abdollah MF, Noh ZAM, Mas'ud MZ, Selamat SR, Yusof R, Melaka UTM (2013) Revealing the criterion on botnet detection technique. IJCSI Int J Comput Sci Issues 10(2) :208–215

[10] Jing L, Yang X, Kaveh G, Hongmei D, Jingyuan Z (2009) Botnet : classification, attacks, detection, tracing, and preventive measures. EURASIP journal on wireless communications and networking, IEEE Computer Society, Vol. 2009, pp 1184–1187

[11] Silva SS, Silva RM, Pinto RC, Salles RM (2013) Botnets : a survey. Comput Netw 57(2) :378–403

[12] Weimer F (2005) Passive DNS replication. In : FIRST conference on computer security incident, p 98

[13] Zdrnja B, Brownlee N, Wessels D (2007) Passive monitoring of DNS anomalies. In : Sommer R, Hammerli B (eds) Detection of intrusions and malware, and vulnerability assessment. Springer, Berlin, Heidelberg, pp 129–139

[14] Feily M, Shahrestani A, Ramadass S (2009) A survey of botnet and botnet detection. In : Third international conference on emerging security information, systems and technologies, 2009 (SECURWARE'09). IEEE, pp 268–273

[15] Perdisci R, Corona I, Dagon D, Lee W (2009) Detecting malicious flux service networks through passive analysis of recursive DNS traces. In : Annual computer security applications conference, 2009 (ACSAC'09). IEEE, pp 311–320

[16] Karim A, Salleh RB, Shiraz M, Shah SAA, Awan I, Anuar NB (2014) Botnet detection techniques : review, future trends, and issues. J Zhejiang Univ Sci C 15(11) :943–983

[17] Choi H, Lee H, Kim H (2009) BotGAD : detecting botnets by capturing group activities in network traffic. In : Proceedings of the fourth international ICST conference on COMmunication system softWAre and middlewaRE. ACM, p 2

[18] Feily M, Shahrestani A, Ramadass S (2009) A survey of botnet and botnet detection. In : Third international conference on emerging security information, systems and technologies, 2009 (SECURWARE'09). IEEE, pp 268–273

[19] E. Cambiaso, M. Aiello, M. Mongelli, and G. Papaleo, "Feature transformation and Mutual Information for DNS tunneling analysis," in 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN),2016.

[20] Maurizio Dusi, Manuel Crotti, Francesco Gringoli, and Luca Salgarelli, "Detection of encrypted tunnels across network boundaries," in Communications, 2008. ICC'08. IEEE International Conference on, 2008.

[21] Maurizio Dusi, Manuel Crotti, Francesco Gringoli, and Luca Salgarelli, "Tunnel hunter : Detecting application-layer tunnels with statistical fingerprinting,",2009.

[22] Irvin Homem, Panagiotis Papapetrou, and Spyridon Dosis, "Entropy-based Prediction of Network Protocols in the Forensic Analysis of DNS Tunnels," 2016.