|  |
| --- |
| UNIVERSITÉ du QuÉbec en Outaouais |
| Dissecteur de chaine pour la détection des maliciels avec Wireshark |
| Projet de session |
|  |
| INF1443 – Sécurité des réseaux informatiques  Prof. : Mathieu Couture |
|  |
| **Guillaume Plouffe** |
|  |
| **11/17/2014** |

Table des matières

[Introduction 4](#_Toc404454117)

[Environnement de développement 5](#_Toc404454118)

[Exemple d'un dissecteur de chaine 6](#_Toc404454119)

[Détection des comportements malicieux 9](#_Toc404454120)

[Asprox 9](#_Toc404454121)

[Pushdo 9](#_Toc404454122)

[Fonctionnement du dissecteur de chaine pour Asprox 12](#_Toc404454123)

[Fonctionnement du dissecteur de chaine pour Pushdo 15](#_Toc404454124)

[Conclusion 19](#_Toc404454125)

[Référence 20](#_Toc404454126)

Table des figures

[Figure 1 : Exemple d'un simple dissecteur de chaine 7](#_Toc404454127)

[Figure 2 : Résultat de l'application de notre simple protocole dans Wireshark 8](#_Toc404454128)

[Figure 3 : Requete initial de Asprox pour ce connecter au serveur C&C 9](#_Toc404454129)

[Figure 4 : Segment TCP contenant les deux clés (entête et donnée) de Pushdo 10](#_Toc404454130)

[Figure 5 : Segment TCP de Pushdo contenant les trois paquets encrypter selon la clé d'entête ou de donnée [6] 10](#_Toc404454131)

[Figure 6 : Décryptions XOR du message Pushdo avec les deux clés 11](#_Toc404454132)

[Figure 7 : Expression créer par l'ajout de l'arbre d'analyse du protocole de détection Asprox 12](#_Toc404454133)

[Figure 8 : Détection d'un URL encrypté par Asprox 13](#_Toc404454134)

[Figure 9 : Notre protocole Asprox dans la hiérarchie des protocoles 13](#_Toc404454135)

[Figure 10 : Dissecteur de chaine pour détecter Asprox 14](#_Toc404454136)

[Figure 11 : Expression créer par l'ajout de l'arbre d'analyse du protocole de détection Pushdo 15](#_Toc404454137)

[Figure 12 : Détection d'un message encrypté par Pusdo 16](#_Toc404454138)

[Figure 13 : Dissecteur de chaine permettant de détecter Pushdo 18](#_Toc404454139)

# Introduction

Wireshark est un analyseur de trafic réseaux très populaire qui permet de réaliser une inspection et une analyse selon plusieurs protocoles réseaux. Le mécanisme d’inspection est réaliser a l’aide de décodeurs de protocole appelé « dissecteur ». Lors de l’installation, Wireshark vient avec plusieurs dissecteurs « buildin », mais il est également possible d’en ajouter. Pour ce faire, il existe différentes techniques : l’intégrer au logiciel, l’ajouter au plug-in, le charger par un appel dans le fichier init.lua, le charger depuis la ligner de commande. Dans le premier cas, le développement risque d’être alourdit lors de la recompilation car c’est tout le logiciel qui sera recompilé lors des modifications apporter au dissecteur. Dans le deuxième cas, il s’agit tout simplement de l’ajouter dans le dossier Plugin. Pour ce projet, nous avons plutôt utilisé les deux dernière technique.

Il existe trois types de dissecteur : Les post-dissecteurs, les dissecteurs de chaine et les dissecteurs conventionnels. Les post-dissecteurs est un décodeur qui sera exécuté sur un paquet lorsque toute les autres dissecteurs auront terminé. Il peut ainsi avoir accès aux données extraites par tous les autres dissecteurs. Un peu similaire au post-dissecteur, le dissecteur de chaine peut avoir accès à un dissecteur en particulier. Il possède l’avantage de ne pas devoir s’exécuté sur tout les paquets. Le dissecteur conventionnel est le plus communément utiliser dans la réalisation de décodeur pour nouveau protocole. Il n’a donc pas accès au autre dissecteur et peut s’utiliser seule.

Pour ce projet, nous avons réalisé deux dissecteurs de chaine écrit en Lua dans le but de détecter des activités provenant des maliciels Asprox et Pushdo.

Dans les sections qui suivent, nous allons expliquer plus en détail l’environnement de développement et le fonctionnement d’un dissecteur de chaine. Nous allons ensuite donner un brève description du comportement que l’on vise a détecter selon le maliciel et donner un présenter les fonctionnalité du dissecteur pour Asprox et Pushdu. Nous allons terminer sur une récapitulation des résultats et une inspection de notre dissecteur sur d'autre pcap susceptible de contenir Asprox ou Pushdo.

# Environnement de développement

Nous avons développé dans un environnement Windows (win7) avec Wireshark 1.2.1. Nous avons opté pour la libraire de dissecteur pour Lua plutôt que C, puisqu’elle est, en comparaison, moins difficile et moins longue à implémenter pour Wireshark. Ce fut également une occasion personnelle d’apprendre un nouveau langage. Apparu en 1993, Lua est un langage léger (Taille de la version 5.0.2 = 95 a 185 Ko) conçu pour être embarquer au sein d’autre application. Il est relativement populaire dans le développement de jeux tel que World of Warcraft, Far Cry, MineCraft, etc.

Le IDE utiliser pour le développement des programmes a été Notepad++ avec sa définition du langage Lua. Puisqu’il s’agissait de réaliser des scripts relativement court, nous n’avons pas juger nécessaire d’utiliser un logiciel de développement complet comme Eclipse ou Microsoft Visual Studio.

Lors du développement, nous avons abondement utiliser la ligne de commande ci-dessous afin de rapidement exécuter un pcap avec un dissecteur particulier :

Wireshark -r C:\Users\Guillaume\Desktop\SecuriteReseau\Project\pushdo2.pcap -X lua\_script:myDissector\httpExtraPushdo3.lua

Nous l’avons éventuellement intégrer dans un batch file pour accélérer le processus durant le développement. Ce n’est que lorsque les deux dissecteurs furent terminer que nous avons ajouté les lignes suivante au fichier *init.lua* de Wireshark afin qu’elles puissent être chargé automatique lors de chaque exécution de Wireshark :

dofile(DATA\_DIR.."myDissector/httpExtraAsprox.lua")

dofile(DATA\_DIR.."myDissector/tcpExtraPushdo.lua")

# Exemple d'un dissecteur de chaine

Dans cette section nous allons expliquer les différents blocs nécessaire a la construction d'un dissecteur de chaine. Les autres types de dissecteur sont constituer de manière légèrement différente et ne seront pas couvert dans ce document.

Dans la Figure 1, un exemple d'un simple dissecteur de chaine est présenté. Son rôle est de lire la valeur du champ du port de destination décodé par le protocole TCP et le mettre dans l'arbre du nouveau protocole intitulé *Extra analysis of the tcp protocol*. Ce script sera lu par Wireshark lors de sont chargement. A ce moment, il va initialiser le nouveau protocole, son champ *tcp\_extra.dstPort* et le champ à lire du protocole TCP. Il va ensuite sauvegarder le protocole TCP originalement assigné dans la table de dissection afin d'analyser les paquets partant vers le port 443. Cette sauvegarde est important afin de pouvoir l'exécuter, par la suite, dans la fonction de notre protocole. Le but de cette exécution est de récupérer les champs d'intérêt (*tcp.dstport*) produit par son décodage. Pour les dissecteurs conventionnel et post-dissecteur, l'exécution des fonctions de protocoles s'effectue sur chaque paquet lors du chargement d'un pcap (ou lors d'un balayage réseau). Puisque, dans l'exemple, nous avons uniquement remplacé le protocole TCP sur le port 443 dans la table de dissection, notre protocole est alors uniquement lancé pour les paquets utilisant ce port. Le résultat dans Wireshark de l'application de notre simple protocole est illustré dans la Figure 2.



Figure : Exemple d'un simple dissecteur de chaine

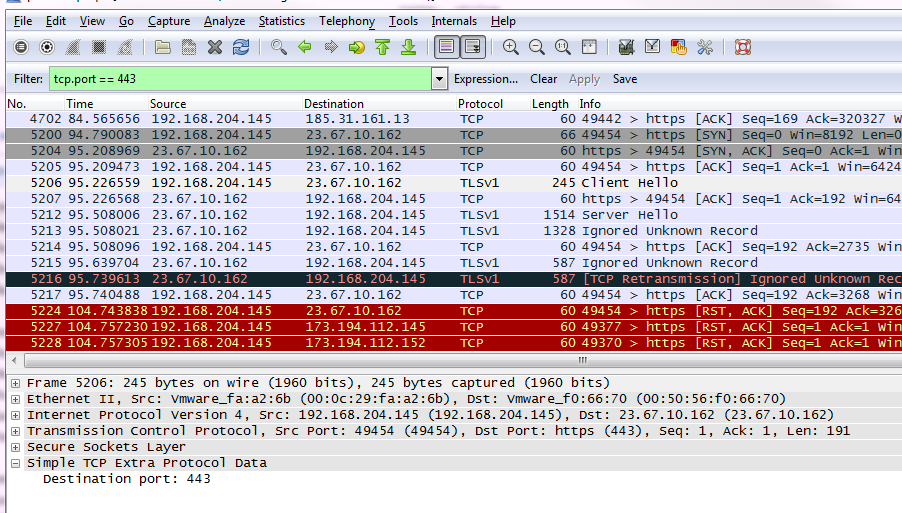


Figure : Résultat de l'application de notre simple protocole dans Wireshark

# Détection des comportements malicieux

Les programmes malicieux Asprox et Pushdo ont été vu brièvement en classe et font partie des lectures obligatoire du cours. Par conséquent, nous supposons que le lecteur possède déjà une certaine e connaissance de ces deux maliciels.

## Asprox

Le pcap sur lequel nous avons basé notre dissecteur du botnet Asprox est disponible a l’adresse *http://malware-traffic-analysis.net/2014/07/08/index2.html*. Il utilise, dans ce cas-ci, la technique de reconnaissance par hameçonnage via les courriels avec de faux E-Zpass (Carte de crédit pour payer les autoroutes a péage).

Tel que vu en cours, lorsque l’exécutable de ce malicieux tente de se connecter à un des serveur de command et contrôle (C&C), il va envoyer un paquet, le plus souvent à destination du port 8080. L’adresse URL de la requête est normalement encrypter RC4 avec comme clé les premier 8 caractère (excluant le « / »).

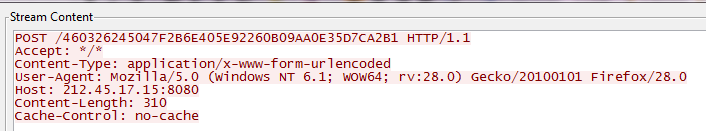


Figure : Requete initial de Asprox pour ce connecter au serveur C&C

Dans notre pcap, nous avons une seule requête vers le serveur C&C et la clé est *46032624*. Le message décrypter est */index.php?r=gate*.

Afin de confirmer la présence d’Asprox, notre dissecteur va analysé tout les paquet envoyer vers un port destinataire 8080 (TCP). De ceux-ci, il va rechercher les paquets contenant une requête URI non vide dans le protocole http et vérifier si les mots clés « php », « index » ou « gate » peuvent être obtenu suite a la décryptions par RC4 du URL. Si c’est le cas, alors nous avons détecté la présence d’Asprox. Nous pouvons alors afficher une alerte et le url décrypter.

## Pushdo

Le pcap sur lequel nous avons basé notre dissecteur du botnet Pushdo est disponible a l’adresse [*http://malware-traffic-analysis.net/2014/10/09/*](http://malware-traffic-analysis.net/2014/10/09/). Contrairement a Asprox, lors de l’envoie de requête vers le serveur C&C, Pushdo ne conserve pas la clé et le message encrypter dans le même paquet. Les données de la requête sont plutôt segmenté sur plusieurs paquet TCP. Cela rend notre travail un peu plus difficile. Le premier paquet contient la version de la clé (généralement 02000000), suivis de la clé Xor pour les entêtes (8 octets) et de la clé Xor pour leur donnée (8 octets). Dans le premier message envoyé par Pushdo dans notre pcap, les clés sont *dafa47b7afa37df5* et *dbfd1af5893d8e38*.

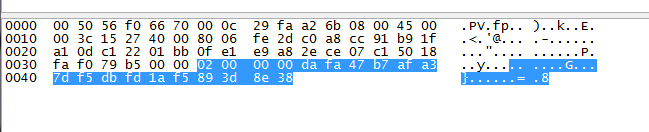


Figure : Segment TCP contenant les deux clés (entête et donnée) de Pushdo

Le segment suivant contient les données envoyées par *Pushdo* au C&C. Ils sont envoyer sont forme de trois paquet *winver*, info et end. Tel qu’illustré a la Figure 5, chaque paquet contient donc sont propre entête et ses propres données. Afin de décrypter rapidement ces trois paquet on peut simplement déchiffrer le tout avec la clé d`entête et la clé de donnée plutôt que selon leur section. De cette manière, on obtient tout de même les mots clés que l’on recherche. La Figure 6 montre les trois paquets décryptés avec la clé d’entête (haut) et la clé de donnée (bas).

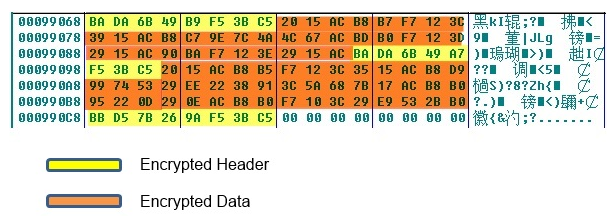


Figure : Segment TCP de Pushdo contenant les trois paquets encrypter selon la clé d'entête ou de donnée [6]

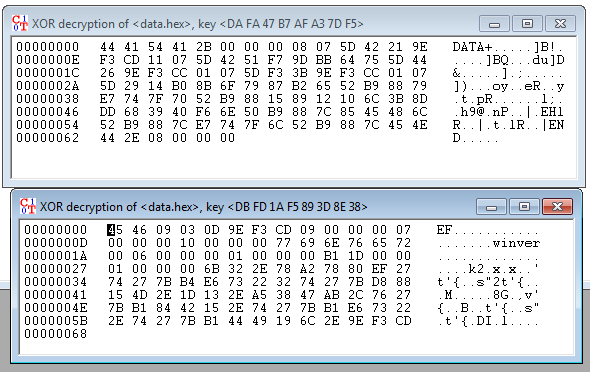


Figure : Décryptions XOR du message Pushdo avec les deux clés

Comme les paquets sont envoyés sur le port 443 (TCP), notre dissecteur va donc se concentrer sur ceux-ci. Nous allons ensuite rechercher les paquets avec un segment TCP commençant avec « 02000000 » et recueillir les clés d’entête et de donnée. La présence de Pushdo sera confirmée si les paquets décryptés sur le segment suivant contiennent l'un des mots-clés suivant: « DATA », « winter », etc.

# Fonctionnement du dissecteur de chaine pour Asprox

Suite au chargement du pcap contenant Asprox, nous allons dans la boite de dialogue contenant les expressions (Figure 7) et sélectionnons le titre de notre protocole, tel que nous l'avons défini lors de sa déclaration.

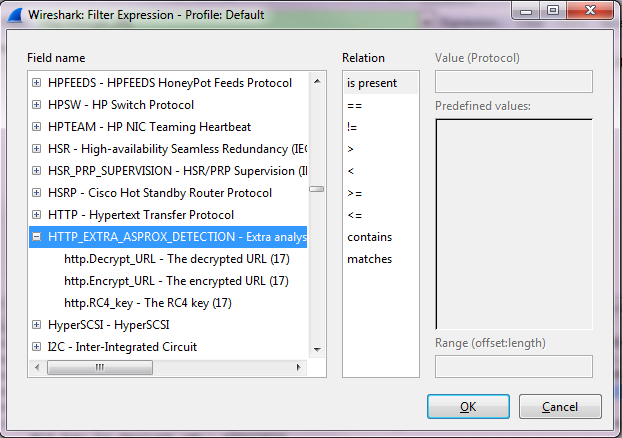


Figure : Expression créer par l'ajout de l'arbre d'analyse du protocole de détection Asprox

Nous allons filtre les paquets avons de trouver les paquets contenant un URL décrypter dans lequel les mots clés décrit dans la dernière section sont inclus. Tel qu'illustré a la Figure 8, nous trouvons un paquet répondant a ce critères, confirmant la présence du maliciel Asprox.

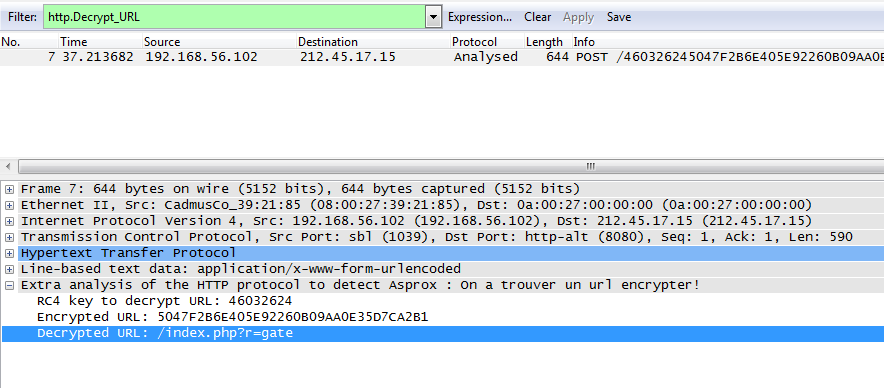


Figure : Détection d'un URL encrypté par Asprox

On peut voir dans la boite de hiérarchie des protocoles (Figure 9) que seul un paquet contient l'arbre de dissection de notre protocole de détection d'Asprox. Il ne fut donc repérer que dans un seul paquet. A noter que le protocole de détection de Pushdo est également activé et, tel que prévu, il ne retourne aucun résultat. Le code du dissecteur est présenté à la Figure 10.

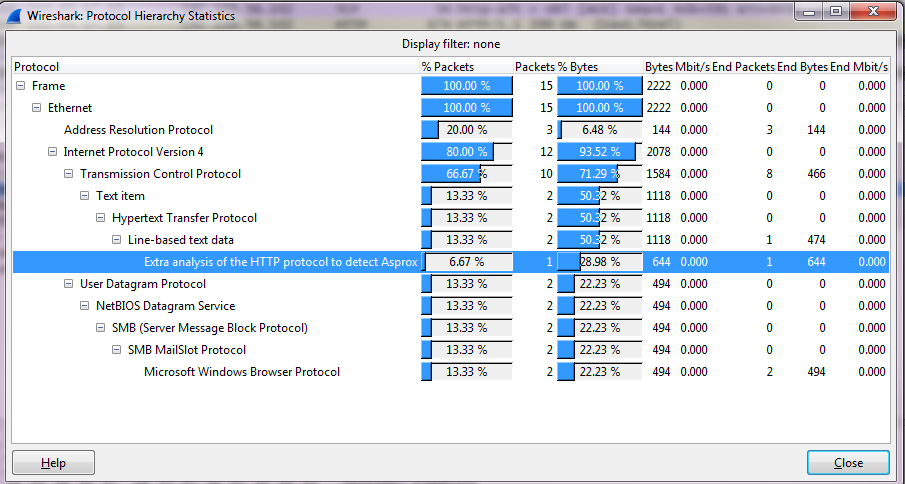


Figure : Notre protocole Asprox dans la hiérarchie des protocoles



Figure : Dissecteur de chaine pour détecter Asprox

# Fonctionnement du dissecteur de chaine pour Pushdo

De même que pour Asprox, nous allons charger le pcpap de Pushdo et vérifier la présence de notre protocole à l'aide de la boite d'expressions.

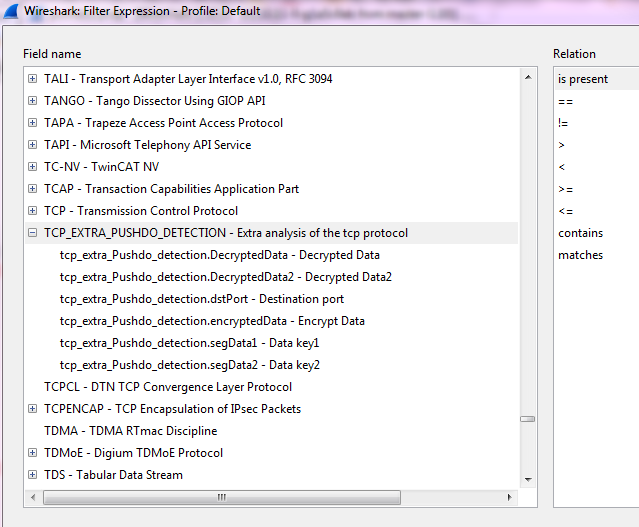


Figure : Expression créer par l'ajout de l'arbre d'analyse du protocole de détection Pushdo

Nous allons filtre les paquets afin de trouver celles contenant un ensemble de donnée décrypter avec la clé d`entête et la clé de donnée pour lesquels certain mots clés sont inclus (voir section précédente). Tel qu'illustré à la Figure 12, nous trouvons deux paquet répondant a ces critères, confirmant la présence du maliciel Pushdo.

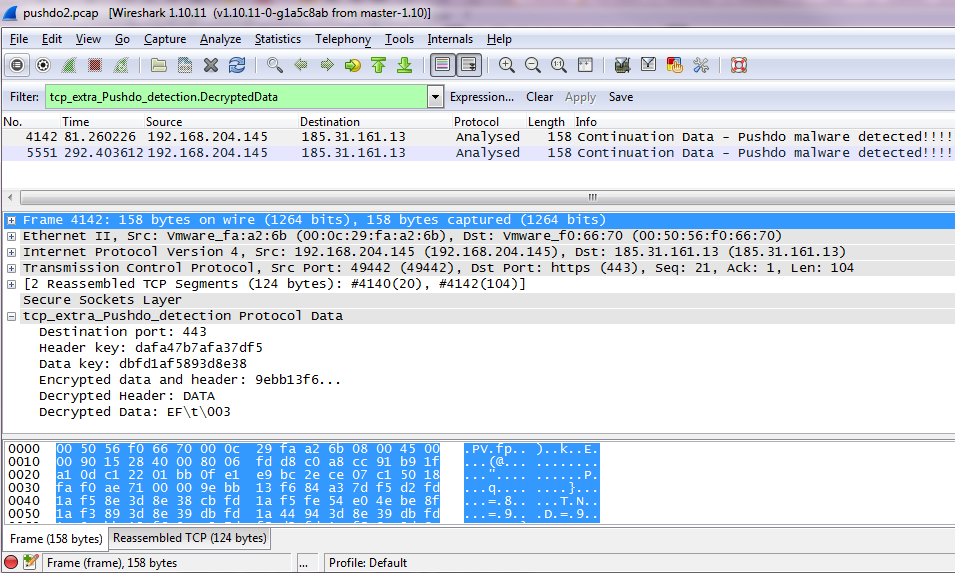


Figure : Détection d'un message encrypté par Pusdo

La fonction du protocole permettant d'analysé les paquets accumule les segments TCP d'un message afin d'obtenir les deux premiers. Ainsi, on peut extraire les deux clé dans le premier et les données dans le deuxième. Malheureusement, en raison d'un bogue présumé dans la libraire des dissecteurs Lua pour Wireshark, seul les 8 premiers octets du second segment nous est retourné. Cela suffit néanmoins à décrypter le mot DATA dans la première entête, confirmant la présence de Pushdo. La Figure 13 présente le script du dissecteur de chaine utilisé à cet effet.



Figure : Dissecteur de chaine permettant de détecter Pushdo

# Conclusion

En résumé, nous avons conçu deux dissecteur de chaine pour Wireshark pouvant être appliquer pour détecter la présence des maliciel Asprox et Pushdo dans un pcap ou lors d'un balayage réseaux. L'utilisation d'un dissecteur de chaine permet d'effectuer un balayage rapide en analysant uniquement les paquets envoyé sur le port TCP communément utiliser par le maliciel. Cette performance n'aurait pas été possible par l'utilisation d'un post-dissecteur et encore moins avec un dissecteur conventionnel puisque, dans les deux cas, tout les paquets aurait été analysé. La détection de mot clé dans les données se révèle être une méthode efficace pour révéler la présence de ses deux maliciel. Il serait également facile de modifier les points de repère tel que la version de la clé d'Asprox afin de s'adapter a d'éventuelles variantes du maliciel. Par curiosité, nous avons analysé au hasard d'autre pcap provenant de *malware-traffic-analysis.net*, soit cinqprésumant contenir Asprox et cinq présumant contenir Pushdo. Ainsi, pour Asprox, deux furent détecté par notre dissecteur dont un possède la même clé que notre pcap de référence:

1. http://malware-traffic-analysis.net/2014/07/02/index2.html
2. http://malware-traffic-analysis.net/2014/08/29/index2.html

Concernant Pushdo, aucun des cinq fichiers analysés ne fut détecté par notre dissecteur.

# Référence

1. http://malware-traffic-analysis.net/index.html
2. http://wiki.Wireshark.org/Lua/Dissectors
3. http://www.cse.sc.edu/~okeefe/tutorials/Wireshark/wsluarm\_modules.html
4. http://wiki.Wireshark.org/LuaAPI/Dissector
5. https://www.Wireshark.org/docs/dfref/h/http.html
6. https://blog.fortinet.com/post/a-new-version-of-the-3rd-generation-of-pushdo