Sniff'eirb, sniffer intelligent

Rapport final

Nicolas Verdier Nicolas Retrain Gabrielle Schmidt
Benjamin Vandenberghe
Fatima Iddir

Responsable : M. Aymeric Vincent

8 janvier 2013

Table des matières

Introduction

Chapitre 1

Les besoins et l'état de l'art

1.1 L'état de l'art

Il existe un grand nombre de logiciels et de librairies qui proposent des fonctionnalités de captures réseaux de bas niveau et d'analyse de paquets (d'un point de vu utilisateur final). Nous pouvons notamment citer libpcap, Scapy, Winpcap ou encore tcpdump. Ces outils ont été un modèle d'inspiration pour tenter de réaliser notre propre *Sniffer*[1] réseau.

1.2 Nos objectifs

1.2.1 Premières étapes

L'objectif premier était tout d'abord de retrouver les fonctionnalités principales de Wireshark, comme par exemple afficher une liste des paquets circulant sur le réseaux ainsi que les informations s'y rapportant (timestamp, adresse IP source, adresse IP destination, port, protocole...), tout en apportant une solution différente et plus simple d'utilisation à un utilisateur néophyte. Ainsi nous avons défini trois premiers objectifs :

- Récupérer les communications[2] en clair d'un réseau.
- Stocker les paquets de cette communication.
- Afficher les différents flux[3] possibles de cette communication.

1. Terme anglicisé désginant un dispositif permettant d'analyser le trafic d'un réseau
2. Une communication est définie par un échange réseau entre deux paires (Adresse IP ; Port).
3. Un flux est défini comme un assemblage possible des paquets d'une communication.

1.2.2 La valeur ajoutée de Sniff'eirb

Le but de ce projet n'étant pas de reproduire un Wireshark, nous devions y apporter des atouts significatifs. Plusieurs fonctionnalités ont donc été implémentées :

L'arbre des possibilités pour les flux TCP : Lors de la capture des paquets circulant sur le réseau pour une même communication, des événements inatendus, comme la répétition de paquets ou encore la perte de certains, pouvaient survenir. La capture de tous les paquets permet donc de connaître les différentes façons de les réassembler pour obtenir un flux décrivant la communication.

La gestion du protocole HTTP : La reconstruction des pages webs visitées par un utilisateur sur le réseau était le fil conducteur du projet. L'analyse des paquets circulant sur le réseau est le principal outil pour arriver à cette fin.

1.2.3 Les idées abandonées ou non implémentées

En parallèle, beaucoup d'idées durent être abandonnée du fait d'un manque de temps et d'énergie pour les traiter. Voici une liste non exhaustive de ces idées :

Chapitre 2

L'architecture

2.1 L'architecture générale

2.2 La base du sniffer : la récupération des paquets et leur stockage

2.3 La reconstruction...

2.3.1 ... des flux

2.3.2 ... puis des documents

2.4 L'affichage

2.5 Choix des outils

-python -javascript

Chapitre 3

L'implémentation

3.1 La base du sniffer : la récupération des paquets et leur stockage

-patch de scapy pour pouvoir sniffer infiniment (jusqu'à l'appel d'une fonction)
-mongodb, no sql, index, -pymongo

3.2 La reconstruction

3.2.1 Recontruction d'un flux

Reconnaissance des protocoles : ARP TCP UDP .. qu est ce qu'un flux : pourquoi il peut y en voir plusieurs : comment reconstruire tous les flux possibles : lianatree, algo, problèmes rencontrées comment choisir le meilleur :

3.2.2 Recontruction d'un Document

Reconnaissance des protocoles : HTTP, FTP, HTTPS, IMAP, SNMP, ... -reconstruction des pages html (.html et .gzip !) -reconstruction des images

3.3 L'affichage

-création d'un serveur web minimaliste -template html -jquery datatables bootstrap -Ajax

Conclusion

Annexes