

Tutoriel : découverte de l'analyseur de protocoles Wireshark

Tutoriel réalisé avec la version 1.6.2 de Wireshark.

Présentation

Un **analyseur de trame** est un outil **de base** de l'informaticien car il permet de comprendre ce qu'il se passe à un niveau très bas. Il permet aussi de mettre en évidence de nombreux concepts théoriques du cours.

Wireshark (anciennement Ethereal) est un logiciel libre d'**analyse de protocole**, ou **packet sniffer**, utilisé dans le **dépannage** et l'**analyse du fonctionnement** des réseaux informatiques. Il est utilisé pour **diagnostiquer** des dysfonctionnements dans un réseau informatique.

Un analyseur de protocoles (ou analyseur de réseaux ou de paquets) est un logiciel permettant **d'intercepter** et de consigner le trafic des données transférées sur un réseau de données. L'analyseur **capture** chaque **PDU** (protocol data unit - unité de données de protocole) des flux de données circulant sur le réseau.

Il permet de décoder et d'analyser leur contenu conformément aux spécifications **RFC** ou autres appropriées.

Wireshark est programmé pour reconnaître la structure de différents protocoles réseau.

Installation de Wireshark (si nécessaire)

Installez Wireshark. Durant l'installation, acceptez d'installer WinPcap 4.1.2

Prise de contact

Une fois installé, le logiciel se présente ainsi :

Vous pouvez très facilement démarrer une capture en cliquant tout simplement sur l'interface réseau qui vous intéresse. Vous ne verrez donc que le trafic réseau vu par cette carte réseau. Démarrez une capture, au bout de quelques instants vous verrez des paquets réseau apparaître dans la fenêtre, ce qui montre que même si vous ne faites rien, il y a des informations qui circulent sur le réseau ! Arrêtez la capture des trames :

Nous allons voir maintenant que ce logiciel donne des informations très détaillées. Examinons l'écran principal du logiciel :

On observe en trois parties : 1. La liste des trames Ethernet capturées. Elles sont chacune numérotées et horodatées par Wireshark (ces données ne figurent donc pas dans la trame d'origine). 2. Pour chaque trame, sa structure est présentée sous une forme hiérarchique (ainsi ce que nous voyons dans le volet 2 est le détail de la trame numéro 2) 3. Le volet 3 est la même chose que le volet 2 mais sous une forme brute non structurée avec une présentation ASCII et hexadécimale. Vous pouvez vous amuser à la présenter en binaire (clic droit dans le volet 3).

1. Examen détaillé Examinons en détail le volet 2. Vous pouvez cliquer sur les croix pour développer les contenus. Ce volet met en évidence le phénomène d'encapsulation : Le premier élément concerne la trame proprement dite (taille, temps, etc.) :

Ensuite, nous montons d'un cran et abordons la partie liée à Ethernet. Nous retrouvons les adresses physiques de destination et de source, également le type trame de niveau supérieur (ici IP 0x0800) :

À la couche supérieure, c'est la partie IP :

Nous retrouvons les adresses IP source et destination du paquet. De plus, nous découvrons que certaines données correspondent à des bits d'un octet particulier (« differentiated services field »). Des données techniques comme la longueur du paquet, le numéro de séquence, le temps à vivre (TTL ou Time To Live), l'identité du protocole supérieur (UDP) sont nécessaires au fonctionnement de cette couche. Montons encore d'un niveau pour observer la partie « transport ». Ici il s'agit de UDP qui est un protocole simple sans gestion des erreurs, son contenu est beaucoup plus simple que TCP :

Comme à chaque fois, une information concernant le protocole de niveau supérieur (ici SSDP pour Simple Service Discovery Protocol) est intégré. Nous retrouvons également la notion de port source et de destination mais aussi de « checksum » qui permet le contrôle d'erreur.

Et enfin, abordons la partie « application ». Nous remarquons que Wireshark sait mettre en relation les données structurées et les données brutes. Ainsi, sur n'importe quelle couche du paquet, si vous sélectionnez un élément, celui-ci est mis en évidence dans le dernier volet :

Pour information, le paquet présenté ici correspond au protocole de découverte UPnP basé sur le protocole SSDP (Simple Service Discovery Protocol). Il s'agit d'un lecteur Philips MCI730 qui diffuse sur le réseau (adresse multicast 239.255.255.250) pour prévenir de ses services (diffuser de la musique).

1. Filtres Lorsque vous capturez des trames sur un réseau, vous pouvez avoir beaucoup de « bruit » puisque nous avons dit que toutes les machines parlent en permanence. Il est donc important de pouvoir filtrer une capture sur différents critères. Parmi les plus fréquents, nous avons les adresses source ou destination de niveau 2 ou 3 et le protocole. Si vous reprenez la figure 1, vous voyez qu'il existe une zone « filter » qui permet justement de saisir des requêtes avec une syntaxe propre à Wireshark (mais qui s'inspire du langage C). Par exemple, filtrons sur l'adresse MAC source : 00:90:3e:90:1a:23 (à adapter à ce que vous avez réellement dans votre capture) Dans le champ Filter saisissez : eth.src == 00:90:3e:90:1a:23 et cliquez sur le bouton Apply Lorsque vous avez commencé à taper eth. Vous avez vu que de nombreux autres champs sont disponibles.

Vous pouvez faire de même avec les adresses IP, par exemple l'adresses source 192.168.1.27. Dans le champ Filter nous saisissons : ip.src == 192.168.1.27

Pour un filtre basé sur les protocoles, nous saisissons tout simplement : Filter : http ou ssh ou dns

Bien sûr, les filtres peuvent être cumulés, par exemple protocole et adresse source : Filter : dns && ip.src == 192.168.1.200

1. A retenir Wireshark est l'outil « universel » de la capture de trames sur un réseau local. Les informations affichées sont très nombreuses et de bas niveau, ce qui permet de comprendre dans le détail le fonctionnement du réseau et de détecter des anomalies. Le logiciel capture toutes les trames « vues » par la carte réseau concernée. Rapidement, cela peut faire de très nombreux paquets puisque les machines parlent en permanence. Il est donc important de pouvoir filtrer une capture sur la base de différents critères (bien souvent les adresses ou les protocoles). Une capture peut être enregistrée afin d'en garder une trace.
2. Configurer Wireshark Vous pouvez vérifier quelle interface réseau est configurée. A partir du menu Capture puis option, utilisez la liste déroulante Interface. Vérifiez la carte réseau utilisée. Pour un ordinateur, il n'y a en général qu'une seule carte réseau.

Vous pouvez ensuite définir les options de capture. Examinez les deux options mises en relief ci-dessous.

Configuration de Wireshark permettant de capturer des paquets en mode de proximité Si vous ne sélectionnez pas l'option Capture packets in promiscuous mode, seules les PDU destinées à l'ordinateur sont capturées. Si vous la sélectionnez, toutes les PDU destinées à l'ordinateur et toutes celles détectées par la carte réseau de l'ordinateur sur le même segment de réseau (c'est-à-dire les PDU transitant par la carte réseau non destinées à l'ordinateur) sont capturées. Configuration de Wireshark permettant de résoudre les noms réseau Utilisez l'option Enable transport name resolution pour indiquer si Wireshark doit convertir les adresses réseau détectées dans les PDU en noms. Bien que cette fonction soit utile, notez que le processus de résolution des noms risque d'ajouter des PDU aux données capturées et ainsi peut-être de fausser l'analyse.

1. Enregistrer des fichiers de capture Vous pouvez enregistrer les informations capturées pour les PDU de données dans un fichier. Le fichier peut ensuite être ouvert dans Wireshark en vue d'une analyse ultérieure sans qu'il soit nécessaire de capturer à nouveau le trafic de données. Les informations qui s'affichent lorsque vous ouvrez un fichier de capture sont identiques à celles de la capture d'origine. Vous êtes invité à enregistrer les PDU capturées lorsque vous fermez un écran de capture ou que vous quittez Wireshark.

From:
/ - Les cours du BTS SIO

Permanent link:
</doku.php/si2/twireshark>

Last update: 2014/11/04 12:18

