# TP7: Analyse de traces réseau

## Module ArcSys

**Objectifs pédagogiques :**

- Comprendre l'encapsulation de données et la notion de couche en réseau (E1) ;
- Apprendre à analyser une trace réseau (E1, E2) ;
- Savoir utiliser des outils pour manipuler des paquets réseaux (E1, E2).
- Sensibilation aux problématiques de sécurité, d'anonymat et d'éthique en réseau (E2).
- Savoir enregistrer une trace réseau (E3).
- Découvrir la rétro-ingénierie de protocoles réseaux (E3).

### ★ Introduction

Dans ce TD nous allons nous intéresser à l'analyse de traces réseau. Une trace est constituée de l'ensemble des trames et paquets qui ont été envoyés et reçus sur un réseau capturés grâce à un *packet sniffer* qui va écouter sur ce réseau. Comme vu en cours, un paquet est constitué d'une en-tête et de données, où l'en-tête contient les informations nécessaires pour acheminer les données et comprendre ce qu'elles contiennent (i.e. le protocole)

L'analyse de traces est nécessaire pour de multiples raisons : comprendre un problème en cours sur un réseau (e.g. congestion), détection d'intrusion, surveillance de l'utilisation du réseau ou encore rétro-ingénierie d'un protocole.

Pour l'analyse, plusieurs outils sont à notre disposition. Dans notre cas nous utiliserons Wireshark et Scapy.

Téléchargez le fichier de traces [1] : `http://people.irisa.fr/Martin.Quinson/Teaching/ArcSys/trace.pcap`

### ★ Exercice 1: Wireshark

Dans un premier temps nous allons utiliser Wireshark pour analyser manuellement la trace. Wireshark va permettre de faciliter l'analyse, car il permet un découpage automatique des paquets en fonction des différentes couches. Il permet aussi un affichage clair des différents champs des en-têtes et de filtrer en fonction de ces champs (e.g. adresse IP ou port).

Par exemple, pour ne voir que les paquets liés à l'adresse IP `10.3.0.1` on peut utiliser le filtre
`ip.src == 10.3.0.1 or ip.dst == 10.3.0.1`

▷ **Question 1:** Chargez la trace dans Wireshark et expérimentez avec l'interface. Par exemple, combien de paquets ont été chargés ? et de paquets TCP ? ou encore en cliquant sur un paquet, observez le découpage en différentes couches de celui-ci par Wireshark.

▷ **Question 2:** Quel protocole s'occupe de la résolution d'un nom de domaine en une adresse IP ? À quelle adresse IPv4 a été résolu le nom de domaine `www.bing.com` ?

▷ **Question 3:** Quand `10.3.0.12` envoie un paquet à `10.3.0.1` et `63.235.37.191`, quelle est l'adresse MAC de destination ? Que remarquez-vous ?

▷ **Question 4:** En observant un paquet TCP, on remarque que les ports source et destination sont dans l'en-tête TCP et non IP, pourquoi ? quelle aurait été la conséquence à la création de IP de mettre les ports dans son en-tête ? pensez-vous que cela serait-souhaitable ?

▷ **Question 5:** Quand `10.3.0.12` initie une connexion TCP à `www.bing.com` pourquoi 3 paquets sont nécessaires (SYN, SYN-ACK et SYN) ? pourquoi pas 4 ? et avec 2 cela serait-il possible ?

▷ **Question 6:** À quoi correspond le paquet 37 ? Quel est le résultat attendu après avoir envoyé ce paquet ? pourquoi le serveur envoie plusieurs paquets en réponse ?

▷ **Question 7:** Comment se calcule la taille maximale des *données* pouvant être envoyées par TCP en un seul paquet ? Pourquoi une taille maximale est-elle nécessaire ?

### ★ Exercice 2: Scapy

Nous allons maintenant utiliser Scapy qui permet de manipuler des paquets via Python (en créer ou en décoder). Pour ce faire, nous vous donnons un début de fichier Python à l'adresse suivante : `http://people.irisa.fr/Martin.Quinson/Teaching/ArcSys/parser.py`

Le but de cet exercice sera de trouver le plus d'informations concernant les personnes connectés au réseau durant la capture, toujours en utilisant la capture précédente.

Pour mettre un peu de contexte, la capture a été effectuée après s'être connecté au Wi-Fi gratuit d'un café en ville. Plusieurs personnes y été connectées, à vous de trouver qui et ce qu'elles y faisaient.

▷ **Question 1:** Complétez la fonction `run_find_ips` qui a pour but de lister l'ensemble des adresses IPs locales des machines connectées au réseau

1. Merci à Suzanne Matthews et David Raymond pour l'idée du TP et du fichier de traces original

▷ **Question 2:** Complétez la fonction `run_dump_emails` qui affiche sur la sortie standard l'ensemble des communications avec un serveur mail pour une adresse IP donnée. Que constatez-vous ?

▷ **Question 3:** Complétez la fonction `run_list_gets` qui affiche sur la sortie standard l'ensemble des requêtes HTTP GET effectuées par une adresse IP donnée (bonus : grouper par nom de domaine). Qu'apprenez vous sur les différentes personnes ?

▷ **Question 4:** Grâce aux informations collectées, pouvez vous donner le nom, le travail, l'adresse e-mail et mot de passe, et autre information intéressante sur chaque personne ?

▷ **Question 5:** Comment se prémunir de ce type d'écoute passive du réseau ? Ne serait-il pas possible d'apprendre de l'information sur les différentes personnes malgré cette solution ?

▷ **Question 6:** Que pensez-vous de ce type d'écoute ?

### ★ Exercice 3: Écoute du réseau (bonus)

Maintenant que vous avez vu comment utiliser des outils pour analyser une trace réseau, c'est à votre tour d'écouter le réseau pour en générer une.

Pour ce faire on utilisera l'interface de Wireshark, mais attention, des privilèges sont nécessaires pour mettre votre interface réseau en mode *promiscuous* (i.e. qu'elle puisse capturer tout les paquets même ceux qui ne lui sont pas destinés). Vous avez deux possibilités :

1. **Facile** Lancer Wireshark en root (mais devrait faire convulser toute personne ayant suivi le module de sécurité)
2. **Moins facile** Suivez les instructions du wiki de Wireshark pour séparer le privilège de capture du réseau : `https://wiki.wireshark.org/CapturePrivileges`

▷ **Question 1:** Après avoir lancé la capture, connectez vous à différents sites que vous utilisez régulièrement et observez les paquets émis et reçus (n'oubliez pas qu'il est possible de filtrer la sortie pour limiter le bruit). Êtes-vous capable de voir le contenu des paquets ? Si oui, sur combien de sites est-ce le cas ? Que pouvez-vous en conclure ?

▷ **Question 2:** Pouvez-vous voir les paquets émis/reçus des autres ordinateurs de la salle ?

▷ **Question 3:** Si c'est le cas, demandez à un autre groupe de lancer le jeu 7colors en réseau et essayez de déterminer le protocole utilisé (il s'agit ici de rétro-ingénierie).

Les curieux de ce genre de pratique découvriront avec intérêt les projets suivants : `https://github.com/ClitherProject/Slither.io-Protocol` et `https://github.com/firebolt55439/Diep.io-Protocol`.