PROGRES - Mini-Projet 1

Sébastien Tixeuil - Sebastien.Tixeuil@lip6.fr

Exercice 1 - Relai TCP

On cherche à programmer en Python un mécanisme de relai entre un client et un serveur utilisant le protocole TCP. Un relai est un programme qui agit comme un serveur vis à vis du client, et comme un client vis à vis du serveur. Il retransmet au serveur toutes les données que le client auraient normalement transmises au serveur, et il retransmet au client toutes les données que le serveur auraient normalement transmises au client.

1. A partir des exemples de code vus en cours pour le client TCP et le serveur TCP, programmer le mécanisme de relai en Python en utilisant la bibliothèque `socket`; On considèrera que l'adresse IP et le numéro de port du serveur sont connues du relai.
2. Tester le relai avec des programmes clients et serveurs utilisant TCP, sur la même machine et sur des machines différentes ; s'assurer que tout fonctionne correctement.
3. Tester le relai avec plusieurs clients (ou plusieurs instances du même client) qui effectuent des requêtes simultanées sur le serveur (via le relai) ; s'assurer que tout fonctionne correctement.

Exercice 2 - Relai HTTP

On cherche à programmer en Python un relai dédié au protocole HTTP. On considère donc que tous les échanges entre le client et le serveur (via le relai) utilisent le protocole HTTP. Le client est donc typiquement un navigateur web, et le serveur est donc typiquement un serveur web.

1. Modifier le relai TCP développé pour l'exercice 1 pour qu'il fasse office de *cache* HTTP : la première fois qu'une URI est fournie en argument de GET dans une requête HTTP, le relai retransmet la requête au serveur, et stocke localement sa réponse dans un fichier ; si la même URI lui est demandée par la suite par le même client ou un autre client, il envoie directement au client le fichier qu'il a stocké localement sans faire de nouvelle requête au serveur.
2. Modifier le relai TCP développé pour l'exercice 1 pour qu'il fasse office de *sniffeur* HTTP. Plus précisément, toutes les URI des requêtes GET du client sont archivées dans un fichier de log, toutes les réponses non vides du serveur aux requêtes GET sont archivées dans le même fichier de log. Le fichier de log doit contenir suffisamment d'informations pour effectuer des audits : étant donnée une URI (ou une partie d'une URI), on veut pouvoir retrouver les adresses IP des clients qui ont obtenu une réponse non vide d'un serveur concernant cette URI. Ecrire un programme en Python qui permette une telle recherche.
3. Modifier le relai TCP pour qu'il fasse office de censeur HTTP. Le relai dispose maintenant d'une liste de sites interdits (fournie au relai, par exemple dans un fichier de configuration). Si l'URI demandée dans une requête GET du client contient un lien qui renvoie vers un site interdit, ce lien est remplacé par un message « Interdit » dans le corps de la réponse faite au client, et le serveur correspondant n'est pas sollicité. L'adresse IP du client ayant sollicité un site interdit doit être stockée dans un fichier de log sur le relai.
4. Tester les relais HTTP réalisés, d'abord individuellement (un proxy entre un client et un serveur), puis en les enchaînant (par exemple, un client est connecté à un cache HTTP qui est connecté à un loggeur HTTP, qui lui même est connecté au serveur HTTP. Tester également le bon fonctionnement quand l'enchainement des relais fait que plusieurs relais sont clients d'un autre relai.