Alex Milliard et Frédérik Boutin

Travail présenté à M. Yacine Benahmed

Université du Québec à Rimouski

Travail pratique #2

INF26207 - Téléinformatique

Table des matières

[Transmission d’un fichier entre deux sockets en python 2](#_Toc100616724)

[Three-way handshake 2](#_Toc100616725)

[Envoi et réception du fichier 2](#_Toc100616726)

[Retrait de l’entête 2](#_Toc100616727)

[Fiabilité du réseau 2](#_Toc100616728)

[Références 3](#_Toc100616729)

# Transmission d’un fichier entre deux sockets en python

## Three-way handshake

Le serveur est lancé en premier et reste ainsi en mode attente de demande de connexion de la part d’un client. Le client, de son côté, tente de se connecter à l’adresse IP et au numéro de port désignée (127.0.0.1 et 22222) et, s’il y parvient, envoie un message « Hello Serveur » accompagné de sa propre adresse IP et numéro de port (127.0.0.1 et 5653). À la réception de ce message, le serveur répond avec son propre message de bienvenue « Hello Client » accompagnée lui aussi de sa propre adresse IP et numéro de port. Finalement à la réception de ce dernier message, le client envoi « connexion » au serveur et ce dernier indique « Connexion établie entre le Serveur et le Client » sur la console à la réception même de ce dernier message.

## Envoi et réception du fichier

Premièrement, la grosseur du fichier à envoyer est déterminer en utilisant la fonction « filesize » de la librairie « os ». Cette information nous permet d’évaluer le nombre total de datagramme qui seront à envoyer pour couvrir l’entièreté du fichier. Le programme ouvre ensuite le document au format lecture binaire (fonction « open ») selon un buffer prédéfinit de bytes (fonction « read » à raison de 1000 bytes dans ce cas). À cette information est ajoutée un entête contenant le numéro du datagramme transmis et le nombre total de datagramme. Le datagramme est ensuite envoyé au client.

Du côté client, le datagramme est réceptionné et l’entête retiré (voir section retrait de l’entête ci-dessous). Une validation séquentielle est effectuée pour valider que le datagramme reçu est bel et bien celui qui est attendu. Dans quel cas, un accusé réception est envoyé au serveur lui indiquant qu’il peut procéder à l’envoi du datagramme suivant. Dans l’éventualité, où le datagramme n’est pas reçu, le serveur attend 3 secondes (via la fonction « settimeout ») et renvoi le même datagramme s’il ne reçoit aucun accusé-réception. Pour le datagramme final, on ajoute un bloc -END- afin d’informer le client que c’est la fin de l’envoi des informations. Cette information nous sert aussi de condition de fin pour les boucles d’envoi.

Le cycle d’envoi et réception ( « send » et « recvfrom ») ne peut être combiné, car cela provoque une erreur au sein du programme. Nous avons rencontré ce problème lors de l’envoi du datagramme et la réception de l’accusé du client, il faut s’assurer de bien séparer ces deux étapes dans la boucle pour que le programme se déroule bien.

## Retrait de l’entête

Une fonction « find » est utilisé afin d’identifier le header envoyé par le serveur via deux variables (« start » et « end »), ces variables sont ensuite enregistrées sous une seule variable « substring ». Cette variable est ensuite retirée du datagramme via l’utilisation de la fonction replace où elles sont remplacées par rien. Une autre variable (« checksum ») accumule, tant qu’à elle, l’ensemble des numéros des entêtes envoyées (XXXX) suivit d’un « ; » en retirant le reste de l’entête en utilisant la même logique que précédemment. Cette variable sera ensuite utilisée pour des fins de vérification.

## Fiabilité du réseau

Une fonction random génère un nombre entre 0 et 100 et, si ce nombre est égale à 20, 40, 60, 80 ou 100, un message vide est envoyé.

# Références

1. <https://docs.python.org/3/howto/sockets.html>, consulté le 9 avril 2022
2. <https://docs.python.org/3/library/os.html>, consulté le 9 avril 2022
3. <https://docs.python.org/3/library/random.html>, consulté le 11 avril 2022
4. <https://docs.python.org/3/library/math.html>, consulté le 11 avril 2022