TP 2 Couche Transport

TP Réseaux SIC-IA

Ilyas Bambrik

2024

Table des matières

Objectifs

- Analyser l'entête TCP avec Wireshark ;
- Travail à rendre : pour chaque question/étape prenez une capture d'écran *complète (les captures d'écran partielles ne seront comptabilisés)*.

Analyse du trafic avec Wireshark

I

La réalisation de ce TP nécessite l'installation de Wireshark :

https://www.wireshark.org/download.html

Lors du lancement de Wireshark soyez sûre que le logiciel est lancé en mode administrateur .

Ouvrez Wireshark et sélectionnez l'interface Npcap Loopback *(1)*. Ensuite, insérez le filtre de port [*tcp.port == 9500*] en haut *(2)* afin d'observer seulement les paquets destinés / générés par le port TCP 9500.

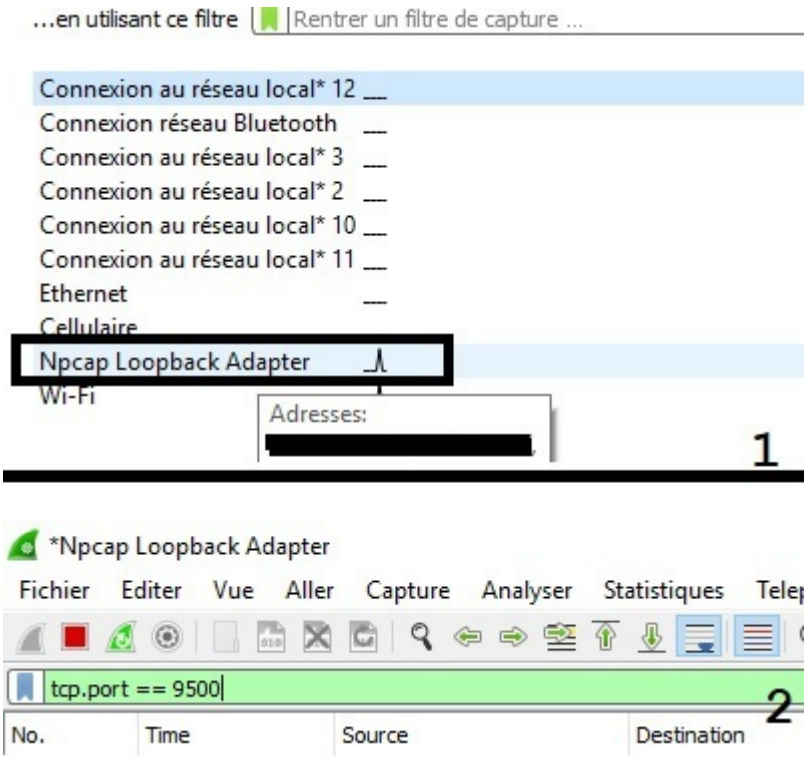


Figure 1. Interface loopback

Ci-dessous, les boutons permettant de a) d'arrêter la capture, b) redémarrer la capture, c) changer l'interface de capture.

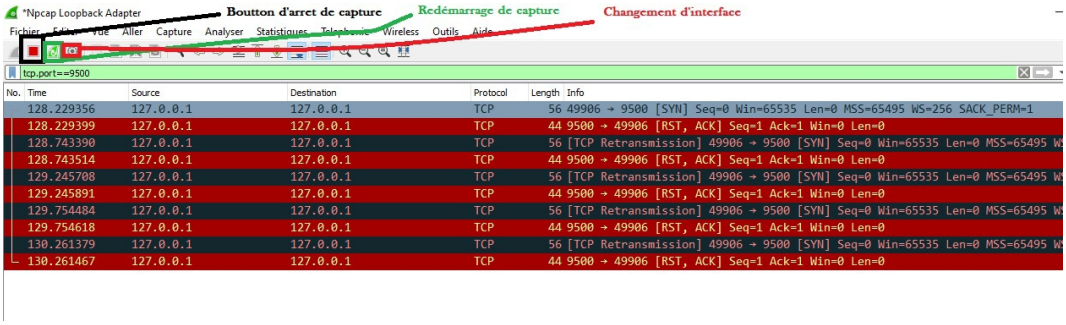


Figure 2. Interface de capture

Exercice : Three way handshake et flags

II

Question

1. Exécutez le client du TP1 (Client.py) sans lancer le serveur et observez le résultat sur Wireshark. Quels sont les flags affichés dans la capture Wireshark ?
2. Exécutez le serveur (Serveur.py du TP 1) et ensuit le programme Client et observez le résultat sur Wireshark. Repérez les flags utilisés pour l'échange threeway-handshake dans la capture (les trois premiers segments échangés entre le client et serveur).
3. Transmettez un message du client vers le serveur et *repérez le contenu de la transmission sur Wireshark* (voir *Figure 3*).
4. Quels sont les flags utilisés dans le segment contenant les données envoyées par client ?
5. Quelles sont les options négociées initialement entre le serveur et client (sélectionnez les deux premiers échanges du three-way handshake afin de visualiser ces options)?
6. Transmettez un message "Fin" depuis le client vers le serveur afin de clôturer la connexion entre serveur et client. *Repérez les flags utilisés pour clôturer la connexion.*

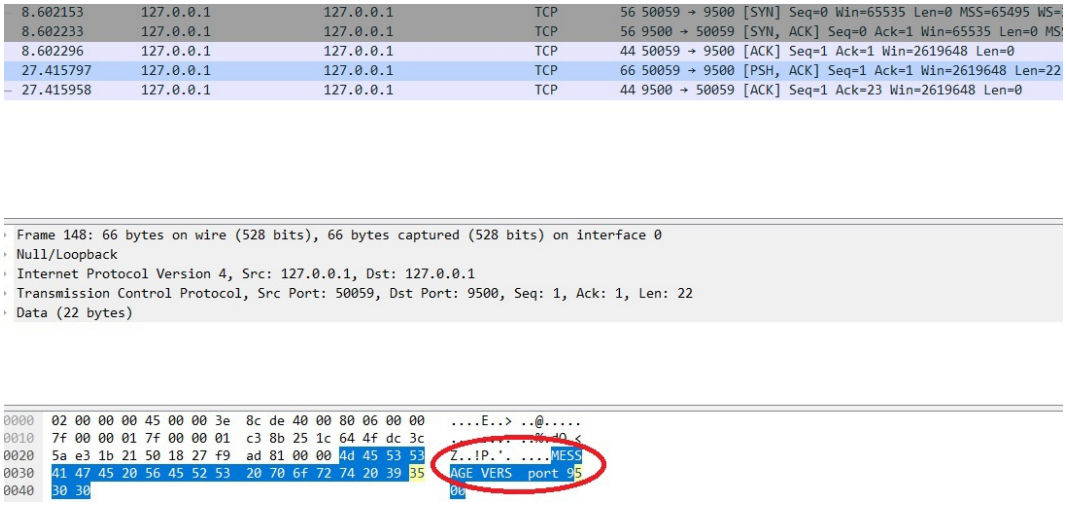


Figure 3. Capture du contenu du message

Exercice : UDP broadcast

III

Dans cet exercice, vous devez prendre une capture d'écran complète de votre écran d'un broadcast UDP sur le port 5000.

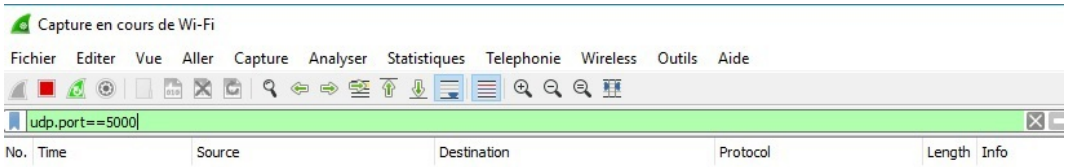


Figure 4. Filtre UDP

Question

UDP est moins utilisé que TCP mais présente d'autres avantages. Par exemple, puisque le TWH n'est pas nécessaire dans UDP, une application peut diffuser périodiquement sur un numéro de port spécifique afin que d'autres programmes/appareils puissent détecter l'existence de celle-ci dans le réseaux local. C'est le cas des jeux vidéos multijoueur dans le réseau local.

1. Connectez au réseau local.
2. Lancez Wireshark *sur la machine de votre camarade* ou d'une machine quelconque dans le réseau local (votre camarade doit être connecté au même réseau local aussi).
3. Filtrez pour obtenir les messages UDP entrants du port 5000 sur la machine de votre camarade (*udp. port==5000*);
4. Exécutez le *programme suivant sur votre machine* afin de diffusez un message dans le réseau (le code est téléchargeable depuis le lien suivant *https://drive.google.com/drive/folders/1jgkUyXKXEsXb-3nhJT-gDt2GRHmZelNh?usp=sharing*) :

```
1 import socket
2 SocketUDP= socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # ceration socket UDP
3 SocketUDP.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1) # rendre le
  socket UDP broadcast
4 Numero_Port=5000
5 AdresseDiffusion='192.168.1.255'
6 MessageADiffuser=b'Votre message a diffuser dans le reseau' # contenu du message
  diffusion
7 SocketUDP.sendto(MessageADiffuser, (AdresseDiffusion, Numero_Port)) #
  transmettre le message
```

Exercice : Client jeux textuel Partie 1

IV

Le code du serveur suivant, prend un mot aléatoire depuis le dictionnaire, réarrange les lettres de ce dernier aléatoirement, et envoie le résultat au client.

```
1 import random, time
2 from socket import socket
3 List_mots=open("dictionnaire.txt").read().split("\n")
4 socket_serveur=socket()
5 socket_serveur.bind(("localhost",900))
6 socket_serveur.listen(10)
7 while True:
8   connexion,_=socket_serveur.accept()
9   start=time.time()
10  mot_aleatoire=List_mots[int(random.random()*len(List_mots))]
11  lettres=list(mot_aleatoire)
12  random.shuffle(lettres)
13  connexion.send(("".join(lettres)+"#GAMESTART\r\n").encode("ascii"))
14  guess=""
15  while "#FINISH\r\n" not in guess:
16    guess=""
17    while True:
18      guess+=connexion.recv(1024).decode("ascii")
19      if "#TRY\r\n" in guess:break
20    guess=guess.split("#")[0]
21    if guess.lower()==mot_aleatoire.lower():
22      connexion.send(("#YOUWON "+str(time.time()-start)+"\r\n").encode("ascii"))
23    connexion.send(b"#WRONG\r\n")
24  connexion.close()
25
```

Question

- Quelle séquence de caractères marque la fin de la réponse client?
- Quels sont les codes de réponse serveur?
- Rédigez un code client qui permet de récupérer et afficher les lettres transmis par le serveur, lire et transmettre le mot saisie par le client et afficher la réponse du serveur à la fin.

Exercice : Client jeux textuel Partie 2

V

Le code suivant est un serveur de jeux textuel qui sélectionne un mot aléatoirement à partir d'un fichier dictionnaire.txt et l'envoie au client. Le client doit taper le mot correctement et le transmettre au serveur. Si le mot envoyé par le client est correct le serveur revoie le temps accomplie par le client :

```
1 from socket import socket;from random import random;import time
2 List_mots=open("dictionnaire.txt").read().split("\n")
3 socket_serveur=socket()
4 socket_serveur.bind(("localhost",700))
5 socket_serveur.listen(10)
6 while True:
7   connexion,_=socket_serveur.accept()
8   start=time.time()
9   mot_aleatoire=List_mots[int(random()*len(List_mots))]
10  connexion.send((mot_aleatoire+"#ENDOFWORD").encode("ascii"))
11  client_typed=connexion.recv(1024).decode("ascii")
12  while "#END" not in client_typed:
13    client_typed+=connexion.recv(1024).decode("ascii")
14  if client_typed.split("#")[0]==mot_aleatoire:
15    connexion.send(("Time to type ="+str(time.time()-start)+"#ENDOFGAME").encode(
  "ascii"))
16  else:
17    connexion.send(b"#TYPINGERROR")
18 connexion.close()
19
```

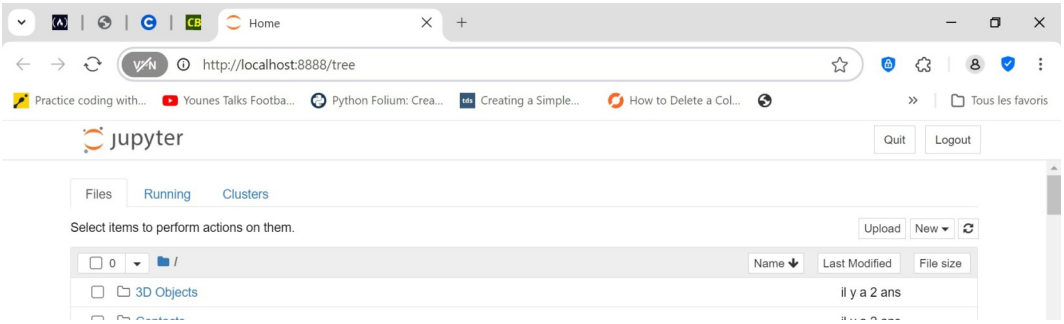
Question

- Quels sont le codes de réponse serveur ?
- Quelle séquence de caractères exprime la fin du mot tapé par le client ?

Exercice : Analyse du trafic Jupyter Notebook

VI

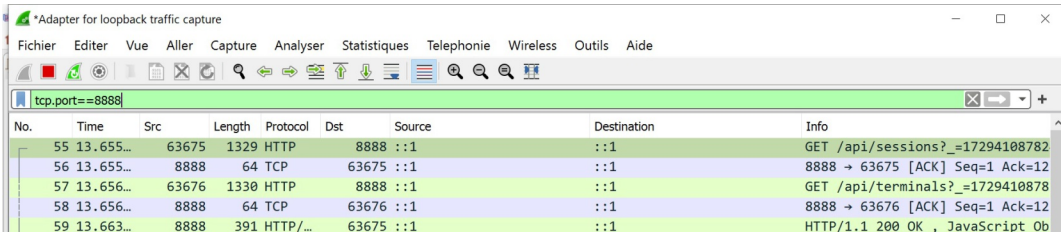
Ouvrez Jupyter Notebook dans votre PC afin de suivre cet exercice.



Question 1

Quel est le numéro de port utilisé pour connecter à l'application Jupiter Notebook ? (regardez l'URL)

Lancez Wireshark et filtrez pour obtenir seulement les messages envoyés du navigateur web vers le serveur Jupiter Notebook (*tcp.dstport==8888 pour éliminer les messages dans le sans inverse*).

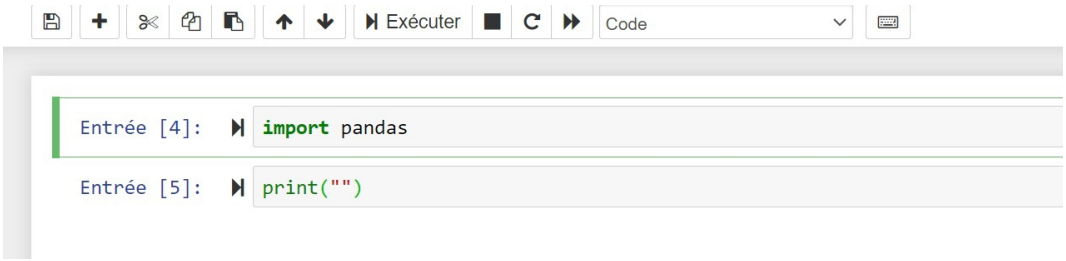


Question 2

Créez un nouveau Notebook python et repérez les messages échangés lors de cette opération sur Wireshark. Utilisez le filtre *tcp.port==8888 && (http||json)*

Question 3

Ajoutez du contenu à votre Notebook et enregistrez le. Repérez les message client avec le même filtre précédant.



Question 4

Exécutez une cellule de code et repérez la réponse serveur. Utilisez le filtre *tcp.srcport==8888.*