Section 7

C) 1.

2. Protocole TCP.

3. voir screenshots **total de donnée – voir len = …**

4. Oui de 5894 octets et un de 5818 octets.

5. Les paquets du ls commence à paquet #21021

De l’information sur les paquets (protocoles, headers, data, etc) qui circule entre le client et le serveur. Comme c’est du TCP, il y a également les confirmations de réceptions.

6. a. On filtre la conversation pour ne laisser que client vers serbeur.

b. On sauvegarde le fichier en raw et on l’ouvre dans winHex.

c. On enleve le bloc de donnees avant FF D8 FF E0 et apres FF D9

d. On enregistre le résultat avec l’extension .jpg

7. a faire

Section 8

Mode1

1. TCP. C’est l’échange de synchronisation. UDP est un protocole sans connection.
2. 5000 du cote serveur et 49811 du cote client.
3. On utilise le filtre pour n’avoir que la partie destination = serveur. On utilise len = … dans la section info pour le nombre d’octets de donnée. Même chose pour serveur vers client, cette fois avec filtre de ip.dst == addr client

(aucune données envoyé de serveur vers client)

1. Le client envoit la même ligne plusieurs fois. Il le fait en 6 itérations.

Mode2

1. TCP
2. 49824 du cote client et 5000 du cote serveur.
3. On met le filtre a dst = serveur.addr et tcp.len plus grand que 0. On obtient ainsi le nombre de paquet dans une direction. On remarque aussi (voir image) que tous les paquets ont 40 octets de données. Même chose pour serveur vers client.
4. Même chose mais en 100 itérations

Mode3

1. UDP et non.
2. 49612 et 5010 pour le serveur.
3. 1 seul paquet client vers serveur de 12000 octets de données. Ne se rend pas.

Mode4

1. UPD et non.
2. 49613 = source et 5010 = dest