

**INF3405 – Réseaux informatiques**

**Automne 2019**

**TP 3 : Analyse d’applications client-serveur avec WireShark**

**Groupe 3**

**2038408 – Clément Prime**

**1879536 – Jacob Dorais**

**Soumis à : Bilal Itani**

**3 Décembre 2019**

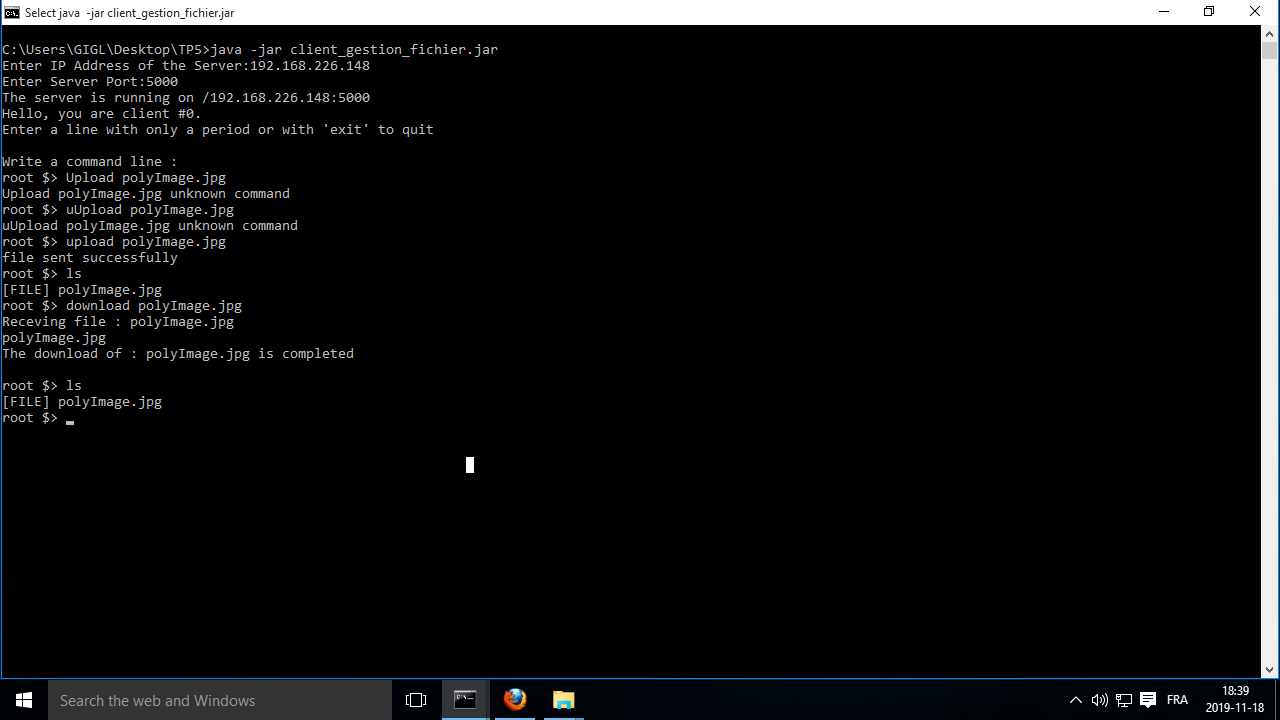
**Introduction**

L’objectif de ce laboratoire était decomprendre les divers types de paquets qui circulent dans un réseau, particulièrement dans les applications client-serveur. Pour cela nous avons dû préparer un environnement de travail composé de client virtuel. Nous avons le client A virtuel avec l’adresse IP 192.168.226.136 et un client virtuel B avec l’adresse IP 192.168.226.137. La première application client-serveur que nous allons évaluer est celle développé lors du TP1 du cours de réseau informatique INF3405. La deuxième application est une application mystère.

Par la suite, on va utiliser l’analyseur de protocoles WireShark pour analyser la transmission de différents messages sur le réseau ainsi que Winhex pour extraire les messages.

1. **Partie gestionnaire de fichier**

Voici une impression d’écran du côté client après avoir exécuté les commandes évalué.



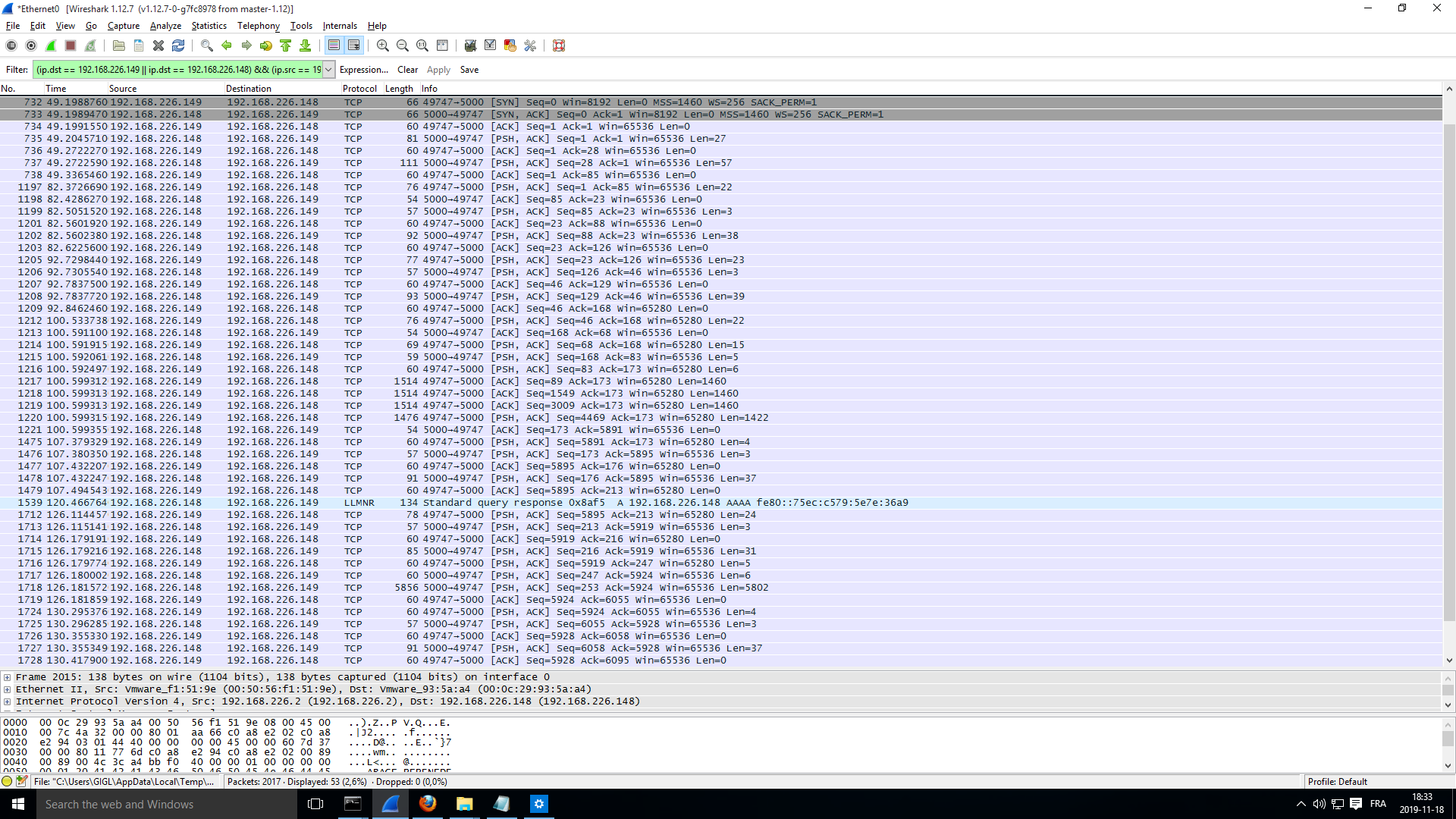
**figure 1 :** Commandes exécuté sur le client

*Question 1 :*

Le filtre appliqué sur wireshark pour avoir uniquement les échanges client-serveur est le suivant ;

(ip.dst == 192.168.226.149 || ip.dst == 192.168.226.148) &&   
(ip.src == 192.168.226.149 || ip.src == 192.168.226.148)

voici le résultat



**figure 2 :** application du filtre pour obtenir la conversation

*Question 2 :*

Tel que vu dans la figure 2, le protocole utilisé lors de la communication client-serveur est TCP

*Question 3 :*

Voici le nombre de paquets et d’octets qui ont été envoyé entre le client et le serveur :

serveur vers client: 20 paquets-7266 octets

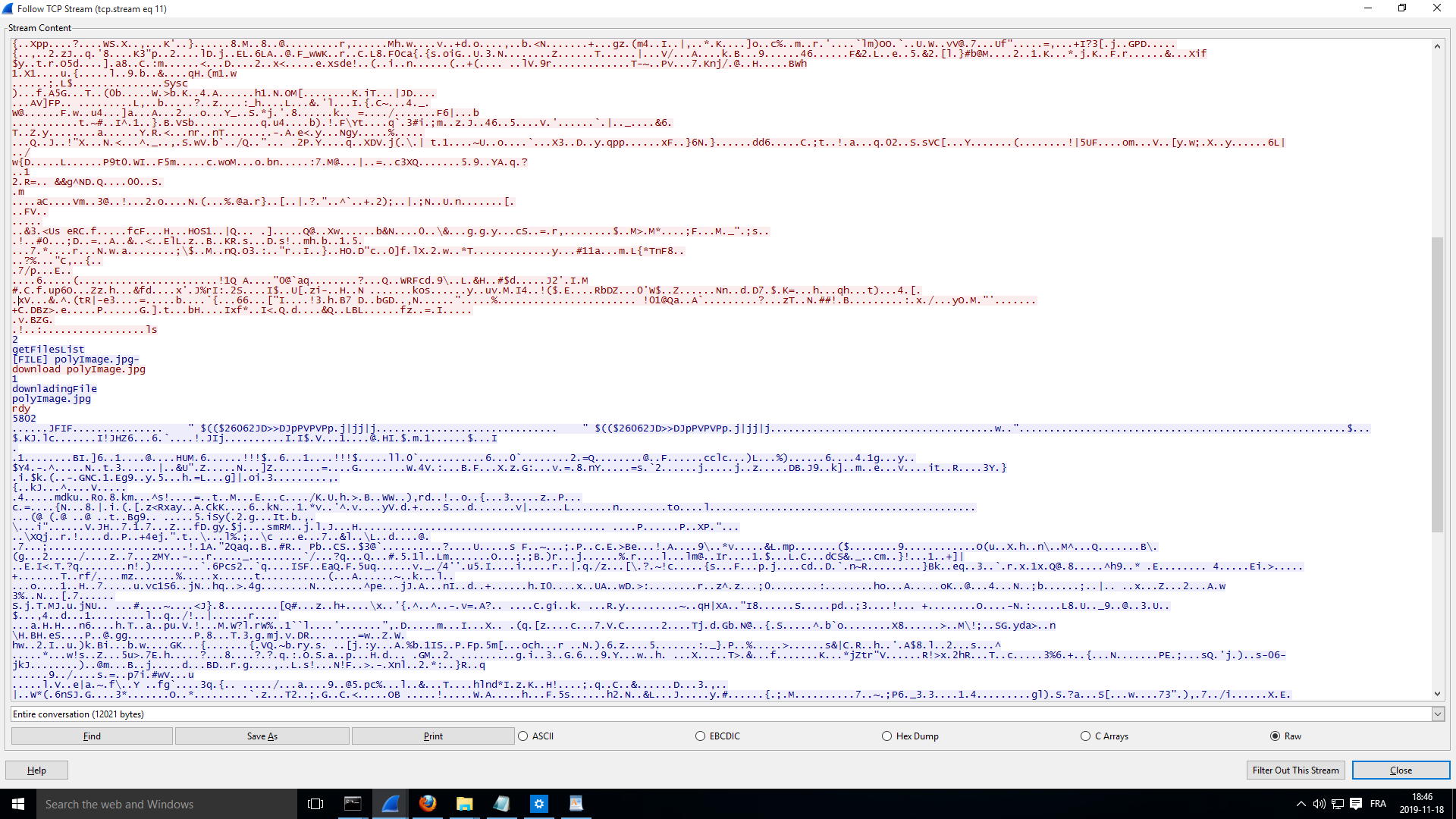
clients vers serveur: 27 paquets-7480 octets

*Question 4 :*

oui, lorsque l’image est envoyé du serveur vers le client. Comme wireshark a fait la capture du côté du serveur, lorsque l'application a envoyé son bloc de 5000 octets, la capture s'est fait avant que la carte réseau le découpe en plus petits paquets.

*Question 5 :*

Sur la figure 3, nous trouvons la conversation où la commande ls a été envoyé.

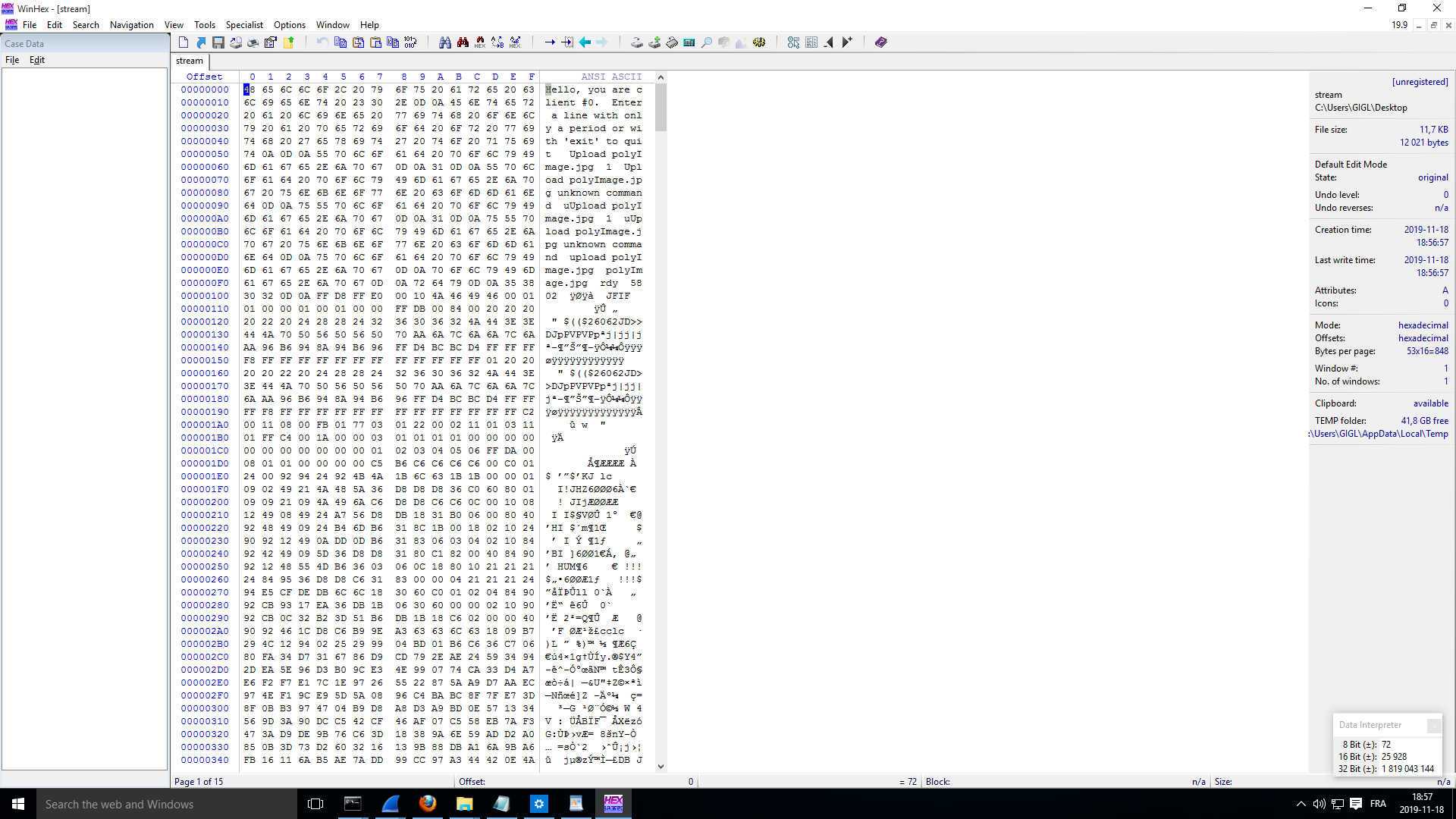


**Figure 3 :** détails de la commande ls

Nous pouvons comprendre le fonctionnement de ls. Une fois ls envoyé par le client, le serveur envoie d'abords un "2" pour indiquer le nombre de message, ensuite, il envoie un message pour signifier qu'il vas commencer à énumerer les fichier, son 2emm message étant le nom du premier fichier. La conversation est donc terminé, car le nombre de message était connue d'avance.

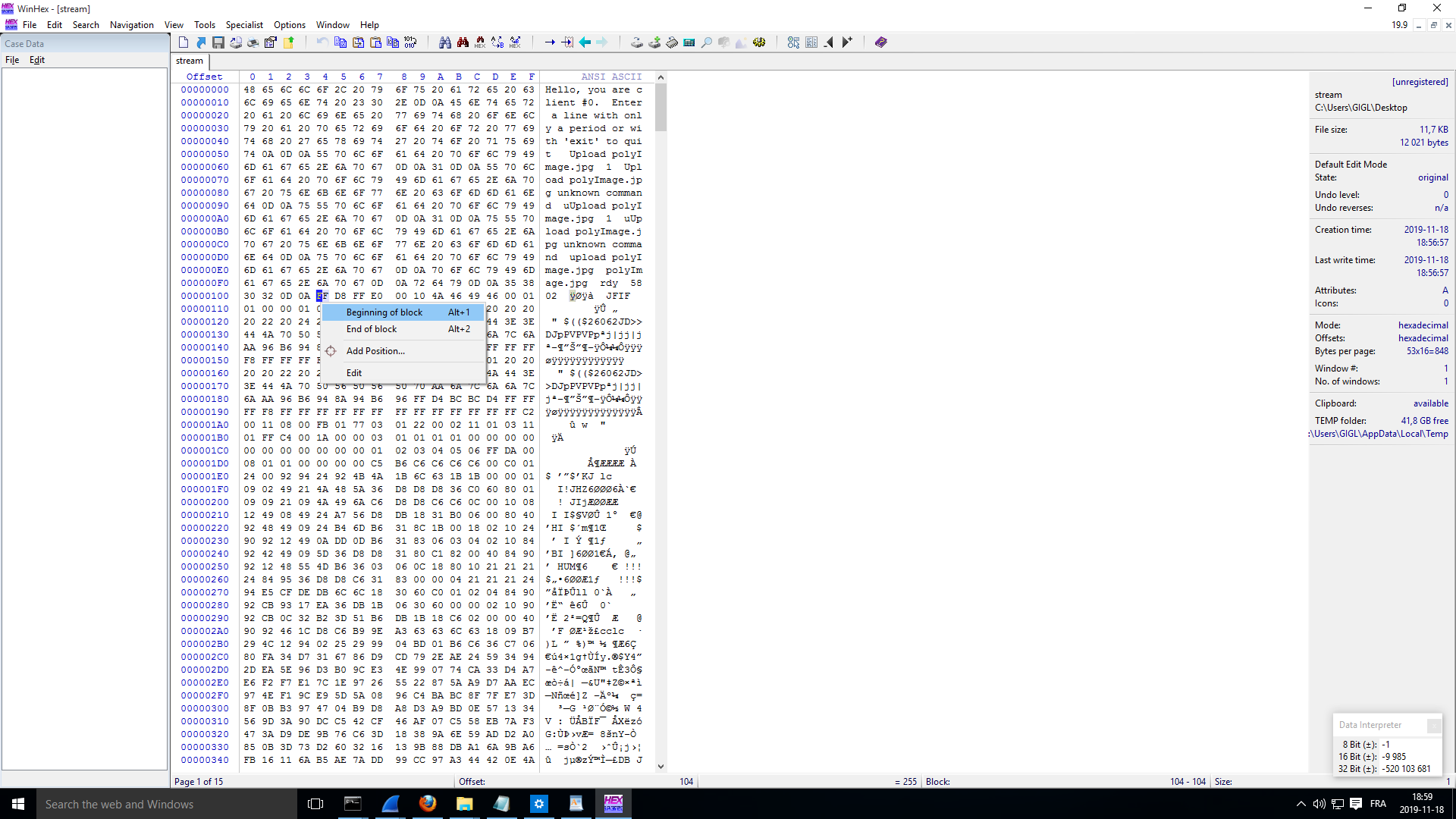
*Question 6 :*

oui, voici, sur les figures 4 à 8, les étapes à suivre pour recouvrir l’image à partir de la conversation.



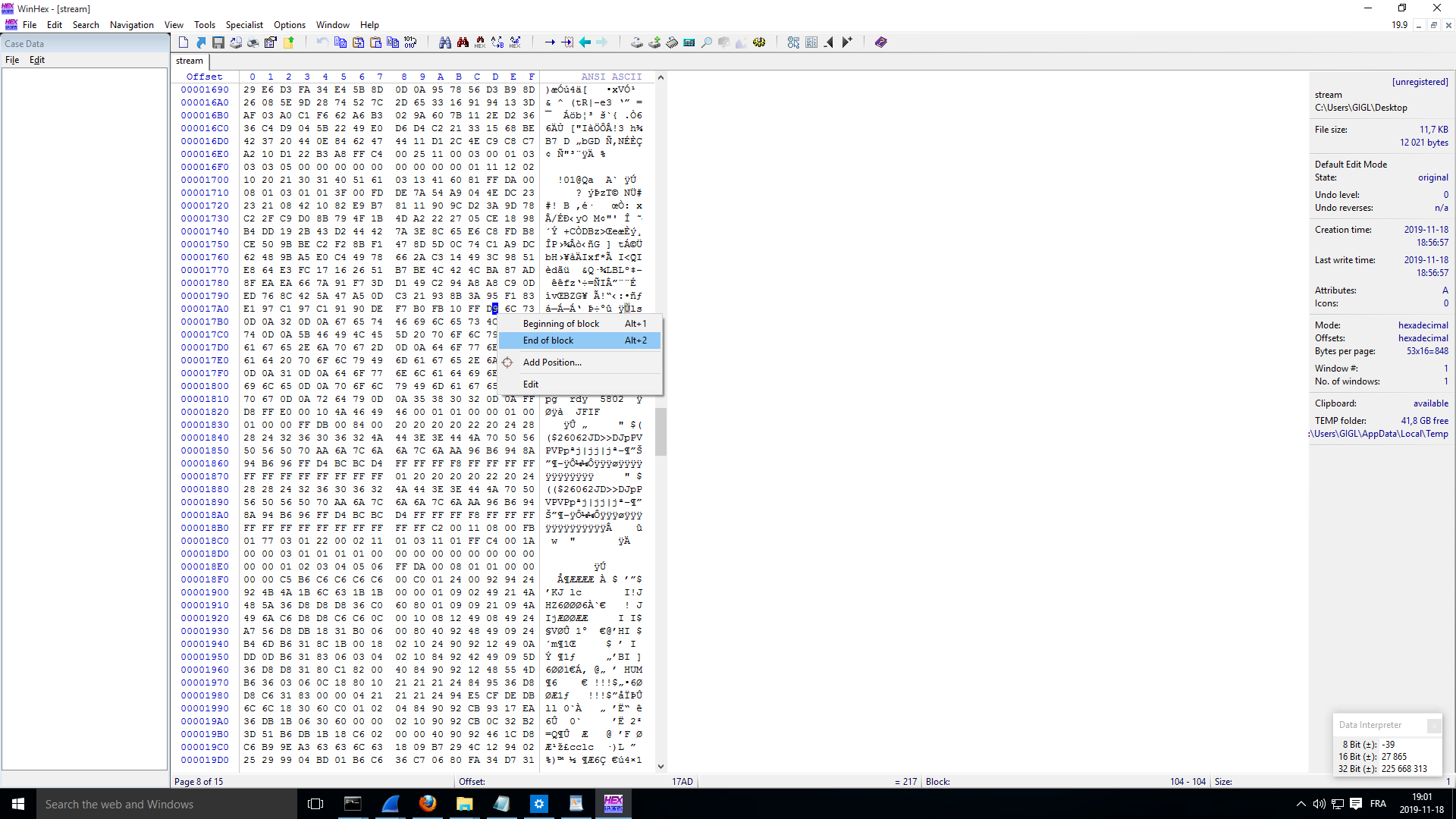
**Figure 4 :** Conversation dans winhex

étape 1. mettre la toute la conversation brute dans winhex



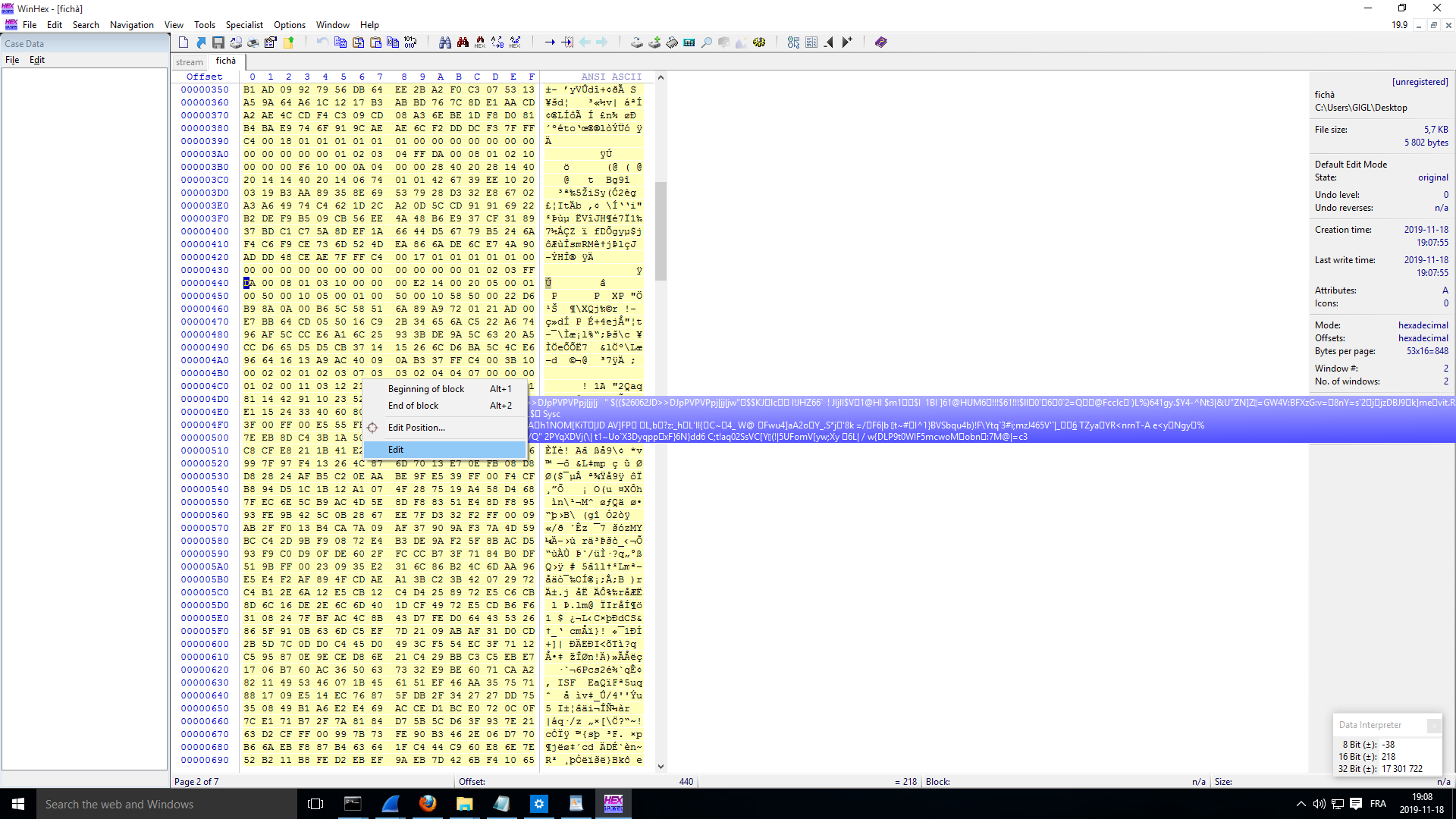
**Figure 5 :** selection de l’entête du fichier

étape 2. chercher l'entête de fichier et désigner comme début de block



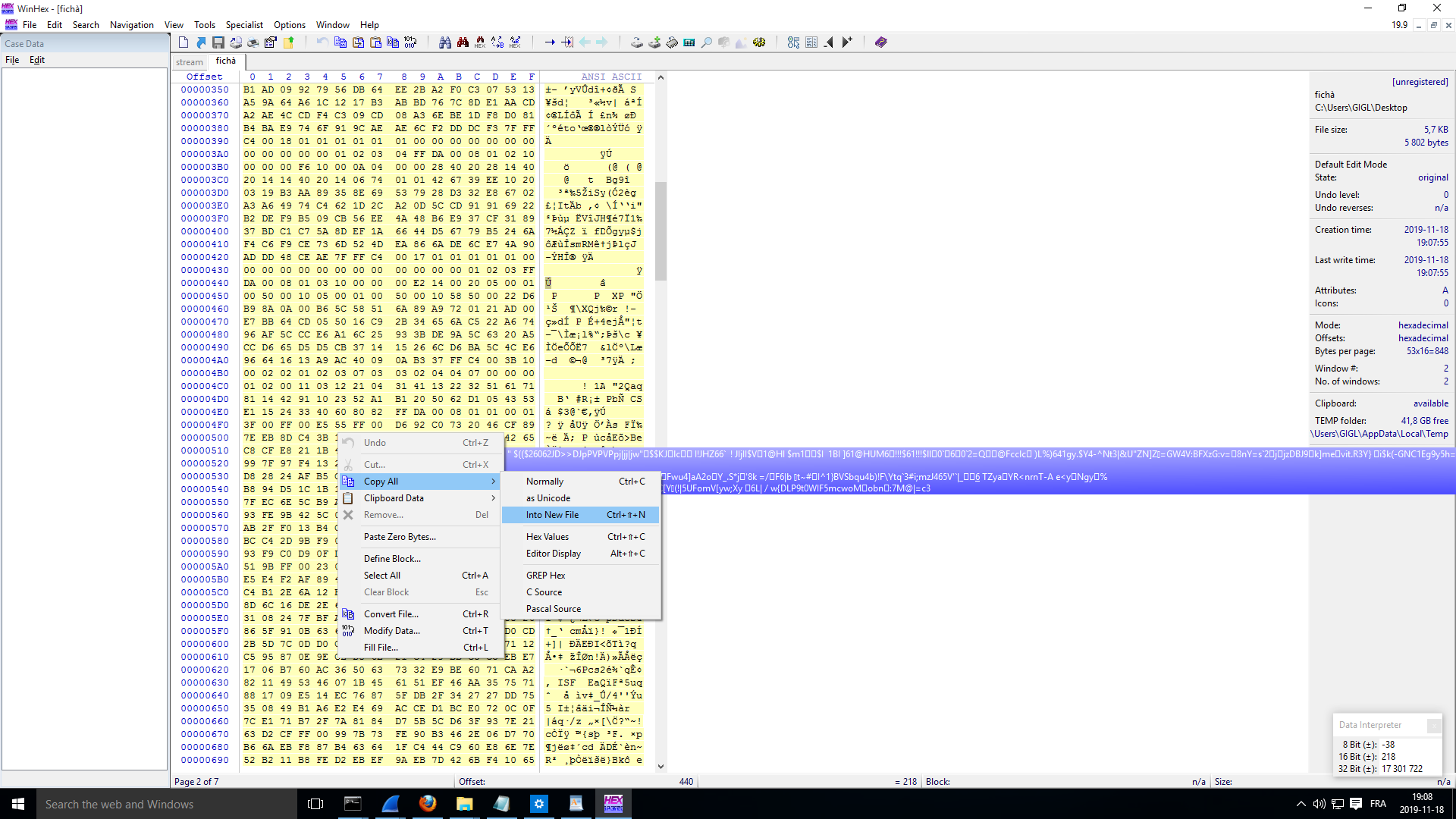
**Figure 6:** selectionner la fin du fichier

étape 3.chercher la fin du fichier et designer comme fin de block



**Figure 7 :** édition du block

étape 4 . clique droit, edit



**Figure 8 :** exportation du fichier

étape 5. all, into new file

étape 6. enregistrer avec jpg comme extension. Le fichier est l’image et ont peux l’ouvrir.

*Question 7 :*

Le gestionnaire de fichier n’est pas sécuritaire, car comme les communication sont envoyé sur le réseau, n’importe qui qui enregistre les paquets passant, peux retrouver tout les fichiers envoyé et reçu de la même manière que nous avons procédé en Q6.

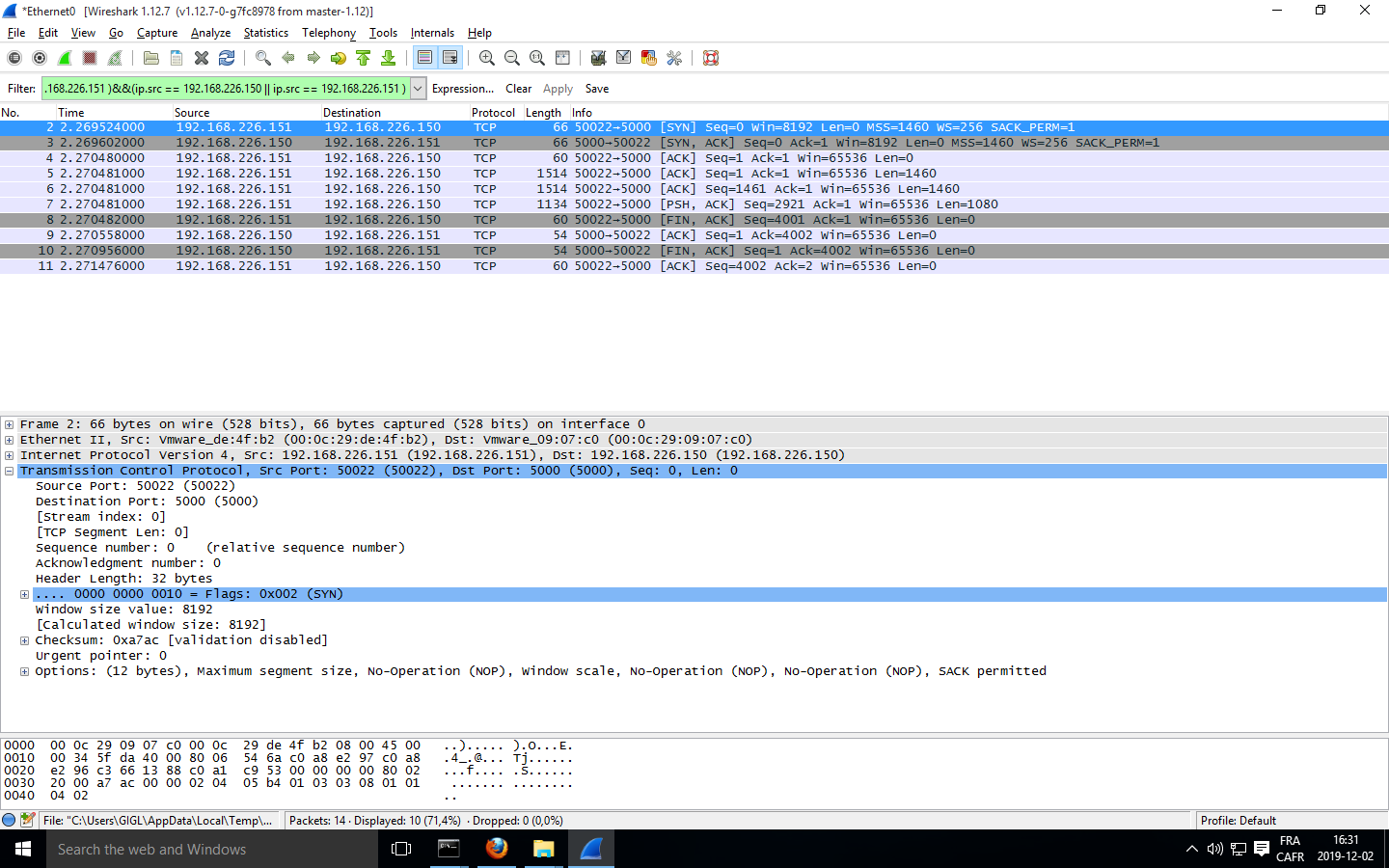
1. **Partie application secrète**

Pour la prochaine partie du laboratoire, il est à noter que nous avons changé les adresses ipv4 du serveur et du client. Voici les nouvelles adresses:

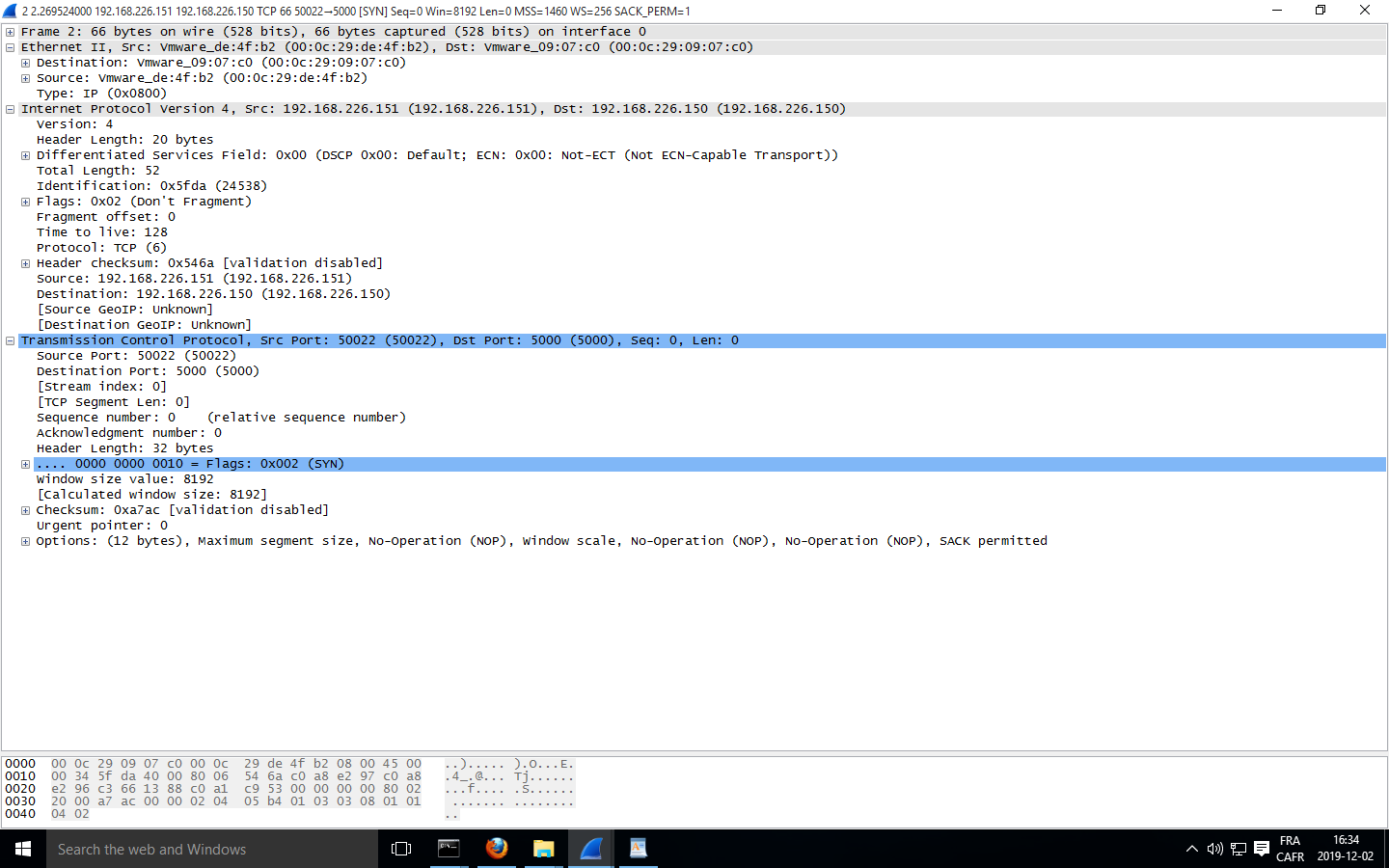
Serveur: 192.168.226.150

Client: 192.168.226.150

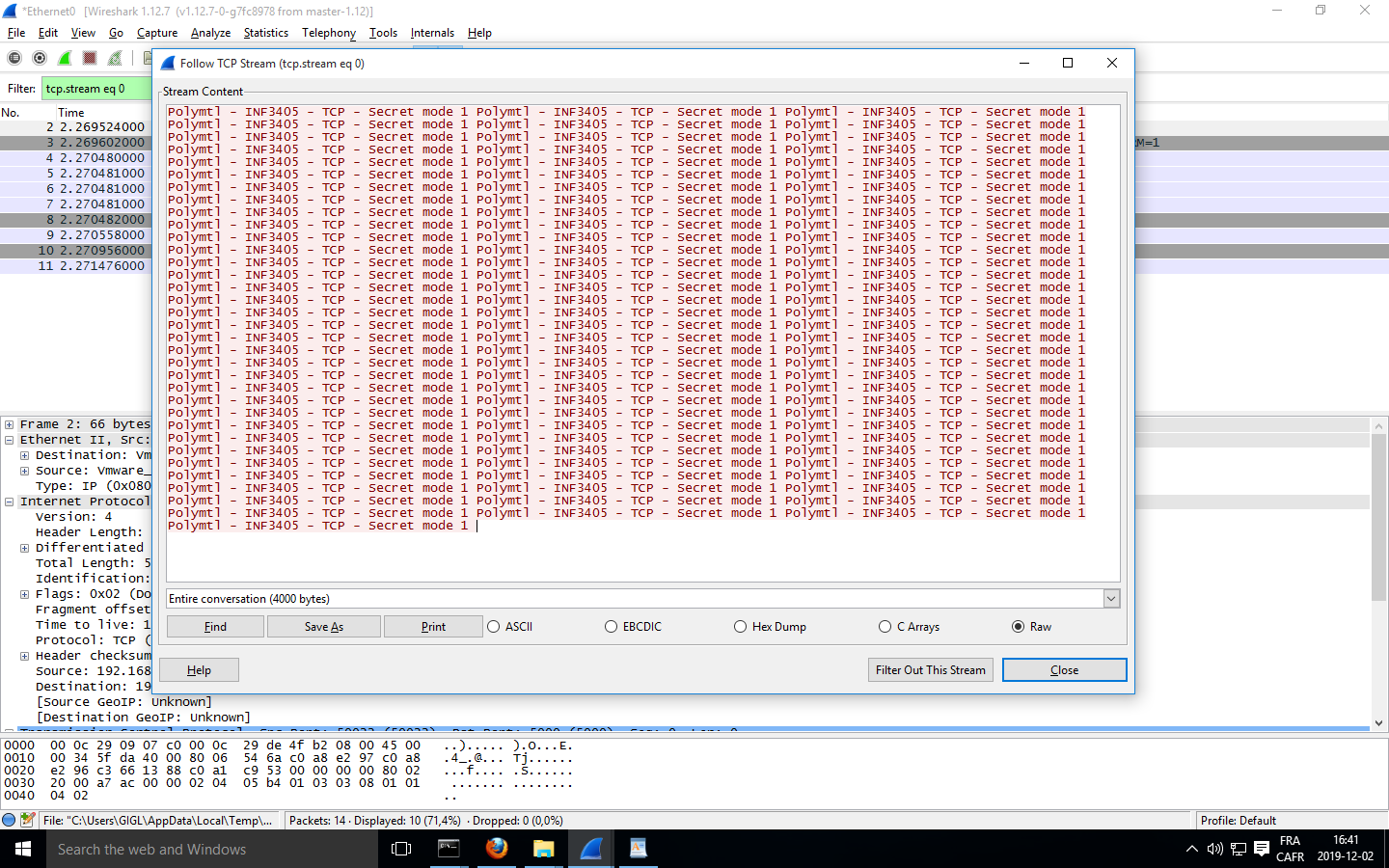
1. **mode secret 1**



**Figure 9 :** conversation du mode secret 1



**Figure 10 :** premier échange du mode secret 1



**Figure 11 :** flot du mode secret 1

*Question 1 : protocole de couche transport*

Le protocole utilisé pour la couche de transport est le protocole TCP. Comme vu à la figure 10, la première communication est de type SYN, pour synchronisation. Le client envoie un message au serveur et attend une réponse du serveur pour confirmer qu'il est prêt à recevoir le message.

*Question 2 : port de communication*

src : 50022

dst : 5000

*Question 3 : nombre de paquets et octets contenants les données*

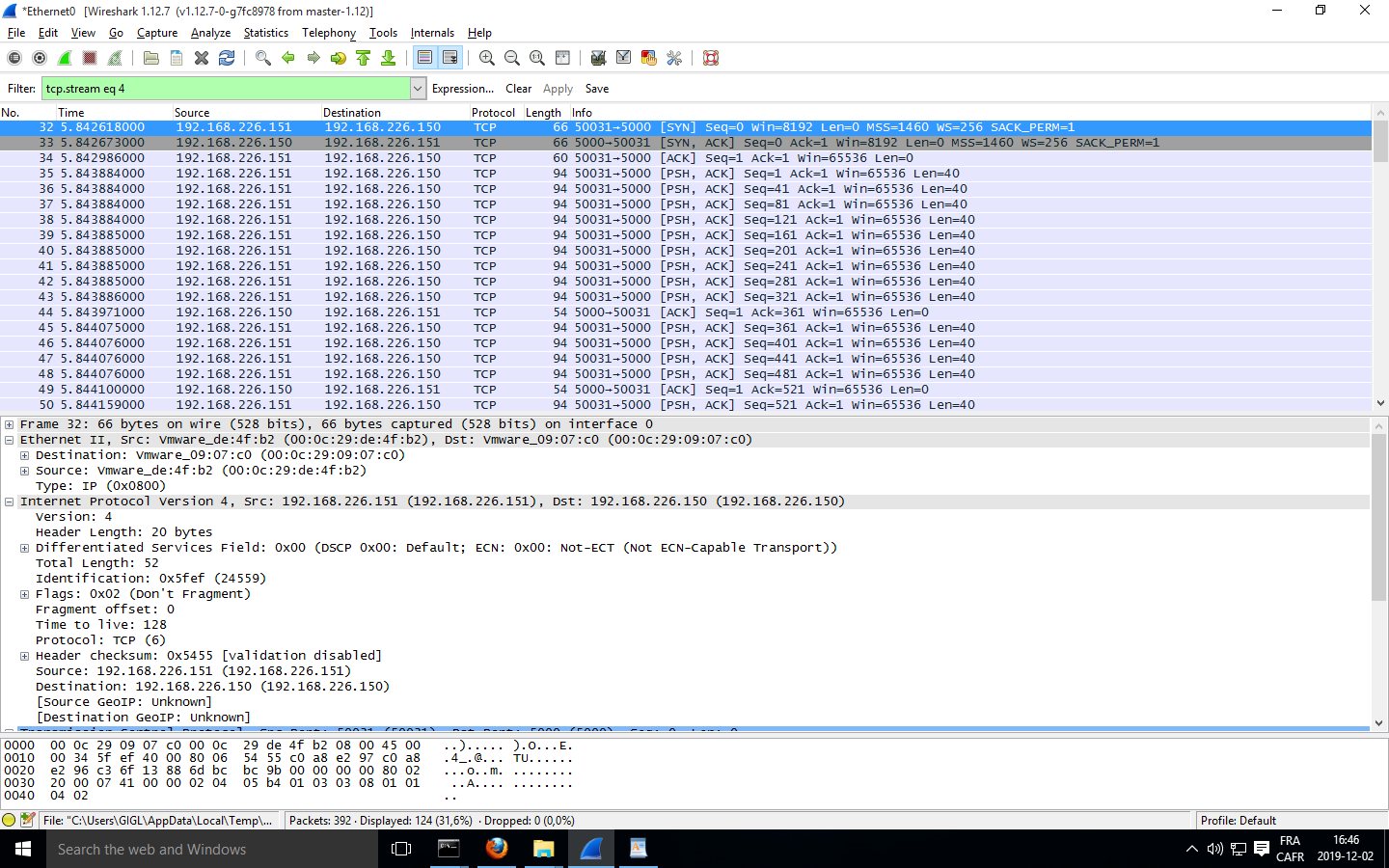
client vers serveur : 7 paquets, 4000 octets de données

serveur vers client : 3 paquets, 0 octets de données

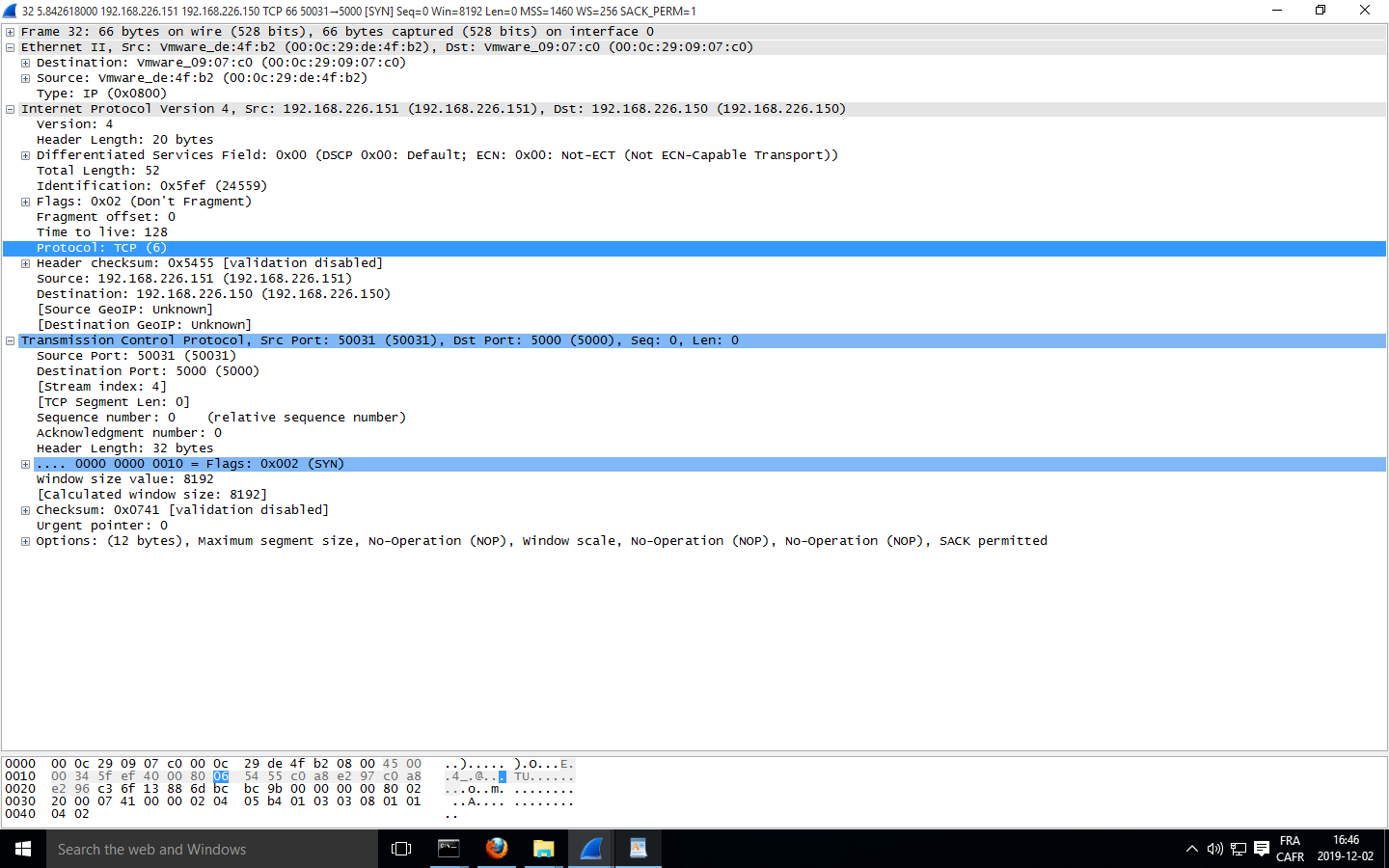
*Question 4 : que fait le client*

Le client envoie un message constitué de 100 fois : « Polymtl – INF3405 – TCP – Secret mode 1 » car il a envoyé 4000 octets.

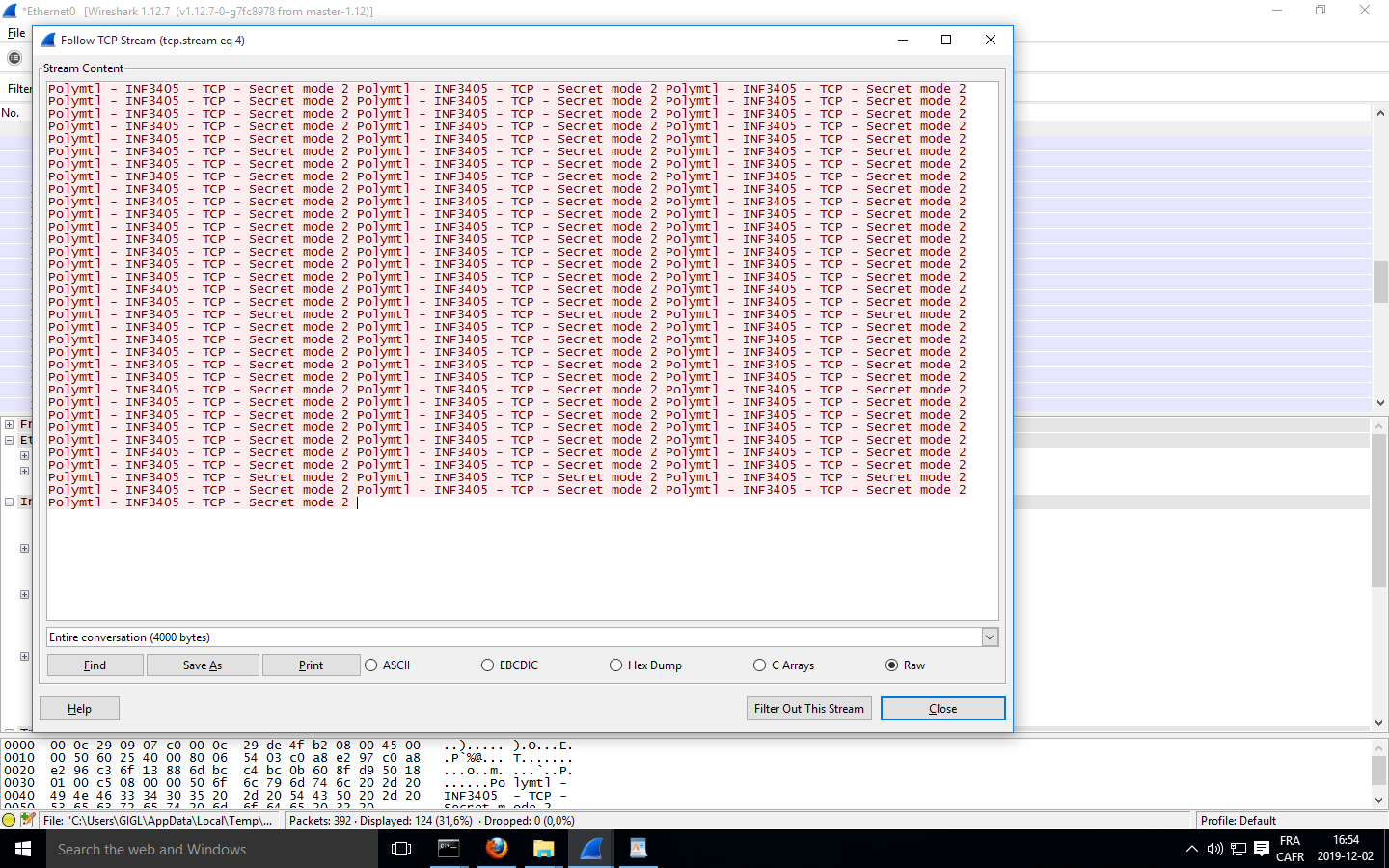
1. **mode secret 2**



**Figure 12 :** conversation du mode secret 2



**Figure 13 :** premier échange du mode secret 2



**Figure 14 :** flot du mode secret 2

*Question 1 : protocole de couche transport*

Comme au mode secret 1, le protocole utilisé pour la couche de transport est le protocole TCP. Comme vu à la figure 13, la première communication est de type SYN, pour synchronisation. Le client envoie un message au serveur et attend une réponse du serveur pour confirmer qu'il est prêt à recevoir le message.

*Question 2 : port de communication*src : 500031

dst : 5000

*Question 3 : nombre de paquets et octets contenants les données*

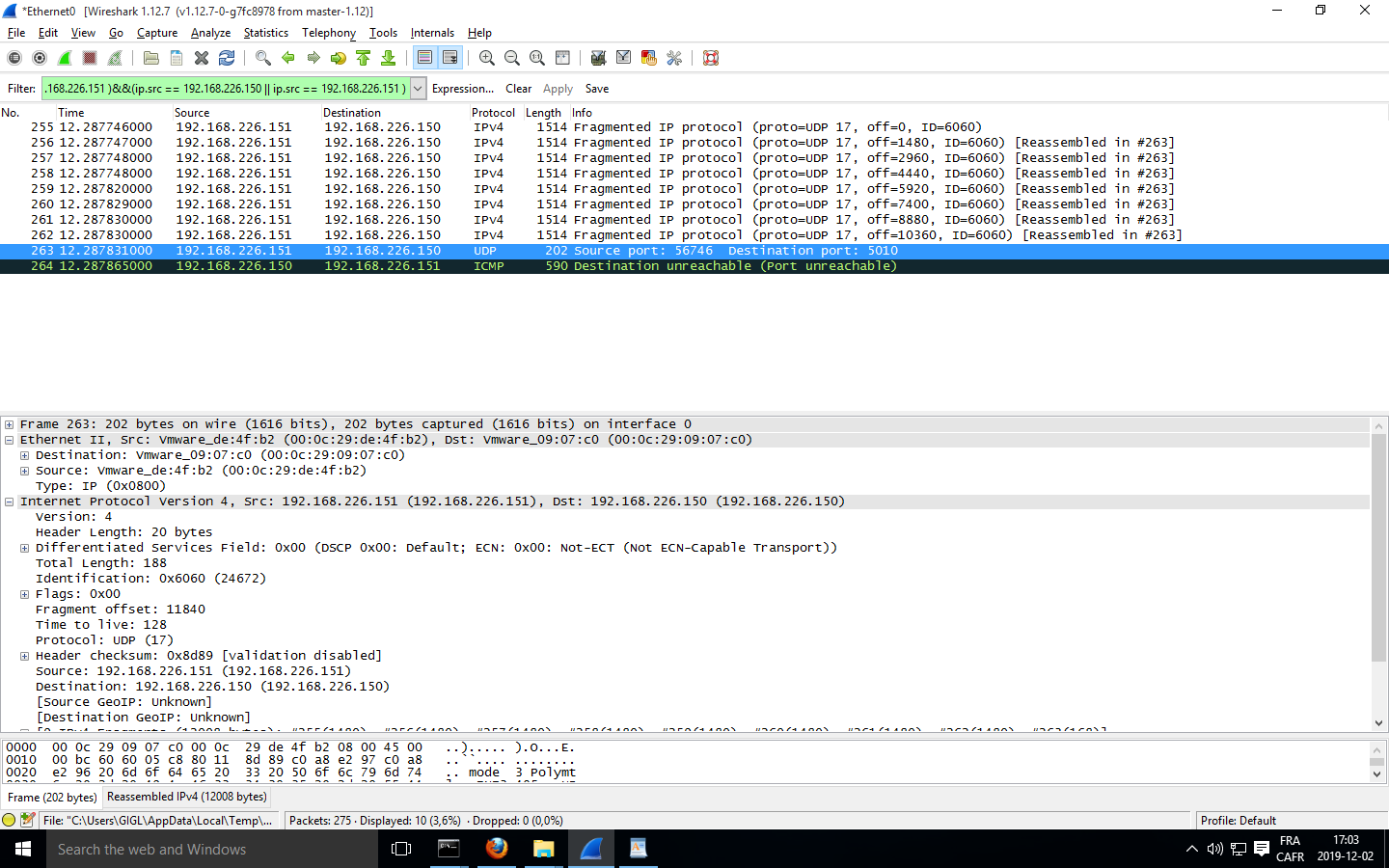
client vers serveur : 136 paquets, 4000 octets de données

serveur vers client : 19 paquets, 0 octets de données

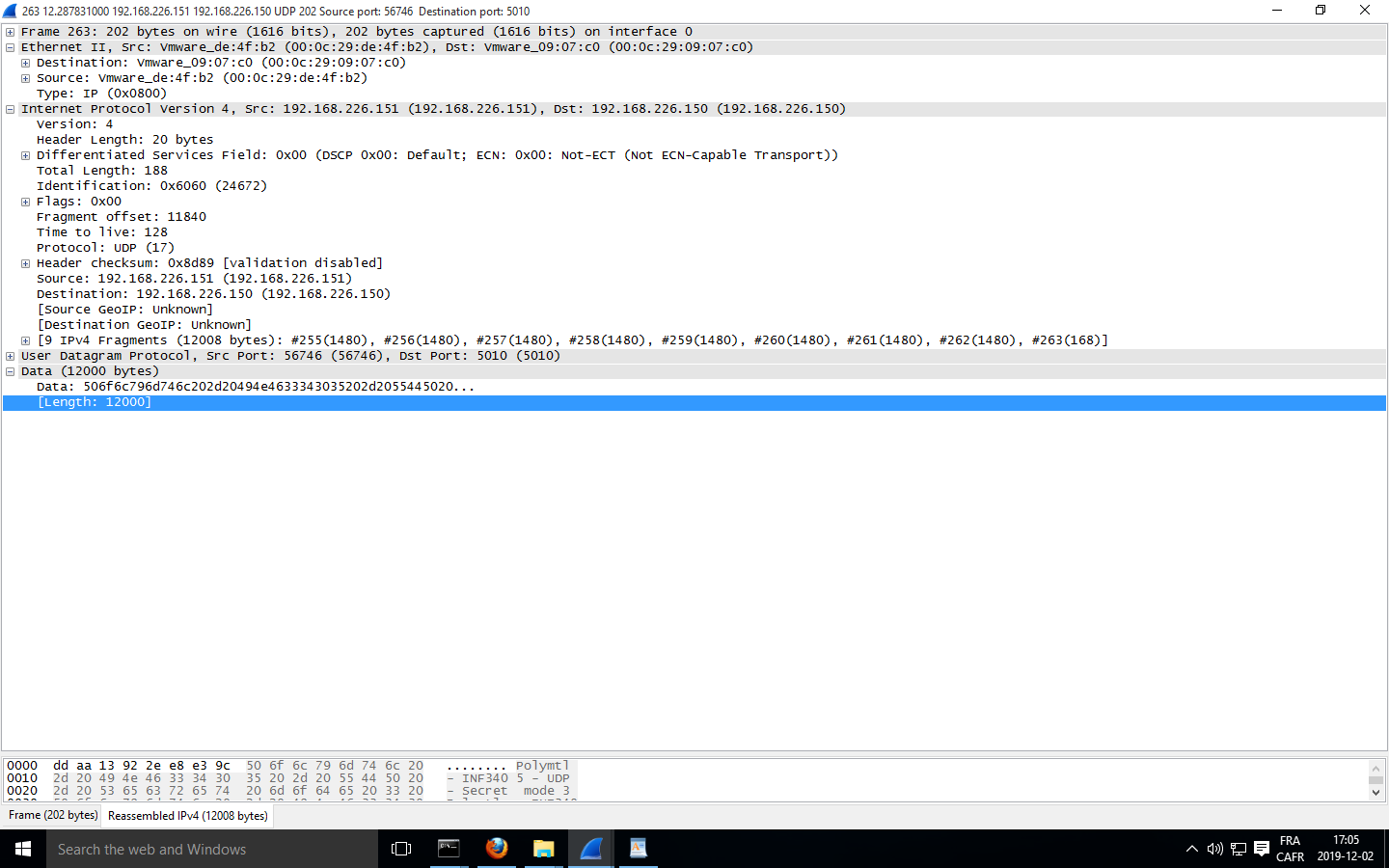
*Question 4 : que fait le client*

envoie le message « Polymtl – INF3405 – TCP – Secret mode 2 » 100 fois, car le message est de 40 octets et il a envoyé 4000 octets. 136 paquets ont été envoyé, à cause des messages de confirmation (acknowledge).

1. **mode secret 3**



**Figure 15 :** conversation du mode secret 3



**Figure 16 :** un des messages du mode secret 3

*Question 1 : protocole de couche transport*

Le protocole de la couche de transport utilisé est UDP. Il n'y a pas de premier échange tel que dans le cas du TCP, car UDP ne s'assure pas de la synchronisation, il envoie les paquets et c'est tout.

*Question 2 : port de communication*

src : 56746

dst : 5010

*Question 3 : nombre de paquets et octets contenants les données*

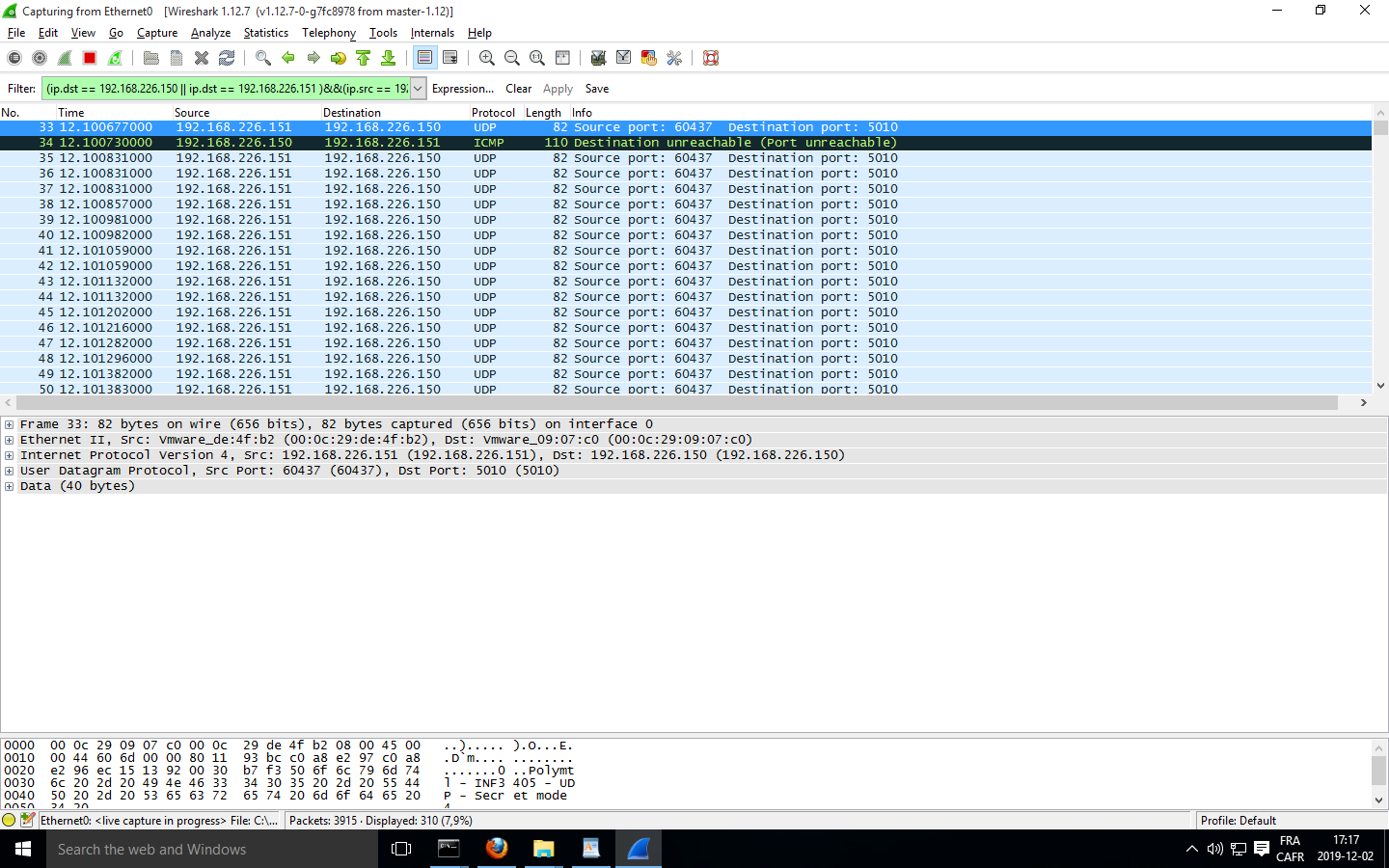
client vers serveur : 9 paquets 12000 octets de données

serveur vers client : 1 paquet 520 octets de données

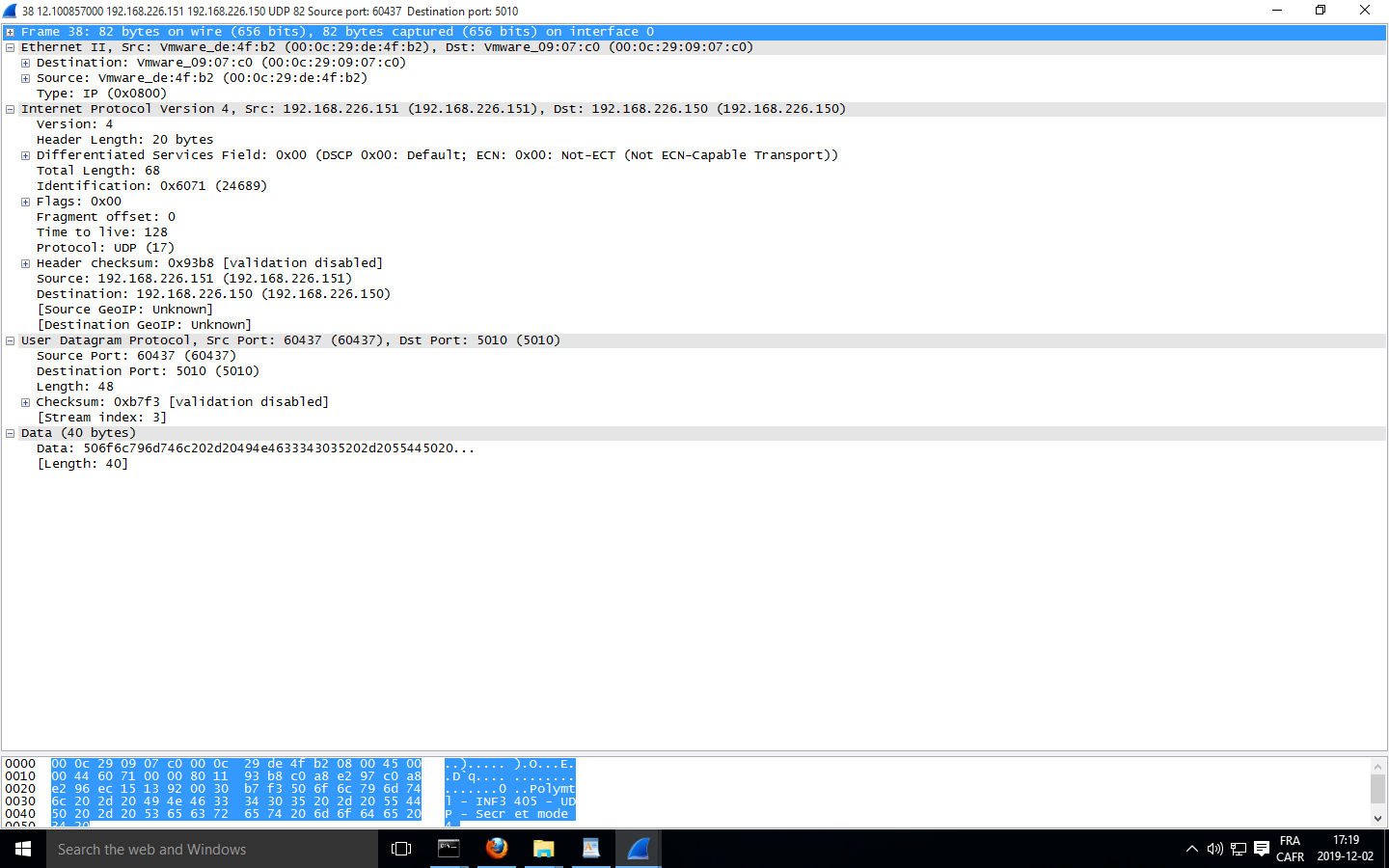
*Question 4 : que fait le client*

envoie un message constitué de 300 fois : « Polymtl – INF3405 – UDP – Secret mode 3 » car il a envoyé 12000 octets. Il reçoit aussi une confirmation fait au niveau de l'application (pas par la couche UDP) où le serveur renvoie 520 des octets de données pour confirmer.

1. **mode secret 4**



**Figure 17 :** conversation du mode secret 4



**Figure 18 :** un des messages du mode secret 4

*Question 1 : protocole de couche transport*

Comme le mode secret 3, le protocole de la couche de transport utilisé est UDP. Il n'y a pas de premier échange tel que dans le cas du TCP, car UDP ne s'assure pas de la synchronisation, il envoie les paquets et c'est tout.

*Question 2 : port de communication*

src : 60437

dst : 5010

*Question 3 : nombre de paquets et octets contenants les données*

client vers serveur : 300 paquets : 12000 octets de données

serveur vers client : 1 paquet : 40 octets de données

*Question 4 : que fait le client*

envoie le message : « Polymtl – INF3405 – UDP – Secret mode 4 » 300 fois, ce qui donne un total de 12000 octets. Il reçoit aussi une confirmation fait au niveau de l'application (pas par la couche UDP) où le serveur renvoie 40 des octets de données pour confirmer.

1. **Analyse des performances et protocol TCP**

*Question 1 : comparer performance de mode 1 et 2*

La différence entre le mode secret 1 et 2 est que le mode 1 envoie un message avec tout les données tandis que dans le mode 2, il envoie une centaine de messages avec plein de confirmation tout au long de l'échange. Le mode 1 est plus efficace puisque moins d'octets doivent être envoyé qu'au mode 2. Au mode 2, comme 155 messages ont été envoyé et reçu, pour le même nombre de données envoyés, beaucoup plus d'entêtes ont été envoyé. Le pourcentage d'efficacité est donc moindre. Cependant, si un paquet venais à être perdu dans le mode 1, un gros paquet devrais être ré-envoyé. Si un paquet est perdu dans le mode 2, seulement un petit paquet devrais être ré-envoyé.

*Question 2 : comparer performance de mode 3 et 4*

La différence entre le mode 3 et 4 est la même qu'entre le mode 1 et 2, c'est-à-dire que dans le mode 3, ont envoie 9 paquets pour les 12000 données, ils ont donc été envoyé d'un coup et séparé par la carte réseau pour former des paquets de la grosseur maximale tandis que dans le mode 4, les données ont été envoyé au compte goûte pour faire des petits paquets de 40 octets. Pour la même raison mentionnée à la question 1, le mode 3 est plus efficace, car beaucoup moins d'entêtes doivent être envoyé dans le mode 3, donc le pourcentage d'efficacité est plus élevé. Cependant, Nous ne pouvons pas amener l'argument d'un paquet perdu. Il est à noter que le protocol UDP est plus efficace que le mode TCP, car comme il n'y a pas de confirmation et les messages sont envoyé unilatéralement, il est donc plus rapide.

*Question 3 : fiabilité de chaque mode*

Évaluons d'abord la différence entre les modes 1,2 et 3,4. Les modes 1 et 2 utilisent le protocole TCP et 3,4 utilisent le protocole UDP. Dans le mode TCP, il y a synchronisation au début de la conversation et les paquets sont vérifié, contrairement à UDP où il n'y a pas de synchronisation et de vérification de paquets. Si un paquet est manquant, personne ne seras au courant et il n'y auras pas de manière de redemander le paquet. Pour ces raisons, les modes 1 et 2 sont plus fiable que 3 et 4. Les modes 1 et 2 sont similairement fiable, mais entre les modes 3 et 4, il y a une différence. Dans ces modes, les applications font une confirmation des paquets au niveau de l'application, cependant, ce n'est pas une confirmation totale, dans le mode 3, 520 octets sur les 12000 ont été confirmé, mais dans le mode 4, seulement 40 ont été confirmé. Nous avons donc 4.3% des octets vérifié dans le mode 3 et 0.3% des octets vérifié dans le mode 4. Pour cette raison, le mode 3 est un peu plus fiable que le mode 4, mais il ne faut pas oublier que les deux sont loin des modes 1 et 2 avec 100% de vérification avec le checksum et s'assurer des paquets envoyés.

*Question 4 : nécéssité du ACK lors de TCP*

Le dernier échange FIN ACK est pour signifier que la conversation est terminé et que le receveur peux arrêter de s'attendre à recevoir d'autres messages. Le protocol TCP imite la manière dont deux personnes parlerais dans la vrai vie, d'abord, un "salut" pour signifier qu'on souhaite communiquer, si l'autre n'as pas entendu, la première personne ne vas pas commencer à parler avant d'avoir la confirmation qu'il écoute. À la fin, un "à la prochaine" pour signifier que la conversation est terminé et tout le monde est au courant

**conclusion**

Nous avons d’abords créé un environnement de travail virtuel avec deux machine dans le même sous-réseau, nous avons testé et compris les différentes manières de communiquer lors de notre travail de gestionnaire de fichier à l’aide du logiciel WireShark qui nous a permis d’analyser tout les paquets entrant et sortant. Nous avons aussi analysé et comparer une application secrète. Cela nous a permis de mieux comprendre les avnatages et inconvénients du choix de protocol tel que TCP ou UDP.