M2PCCI

Cheick CISSOKO

Gaëtan LAGIER

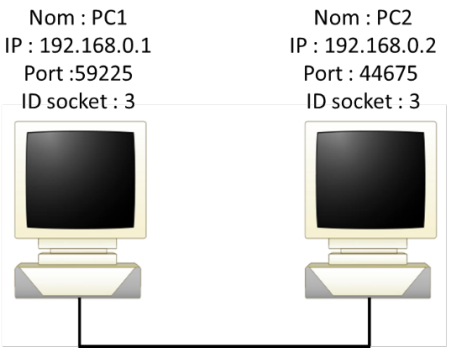
Eric THIERRY

RAPPORT TP3

2016/2017

Etude de la couche transport, le protocole UDP

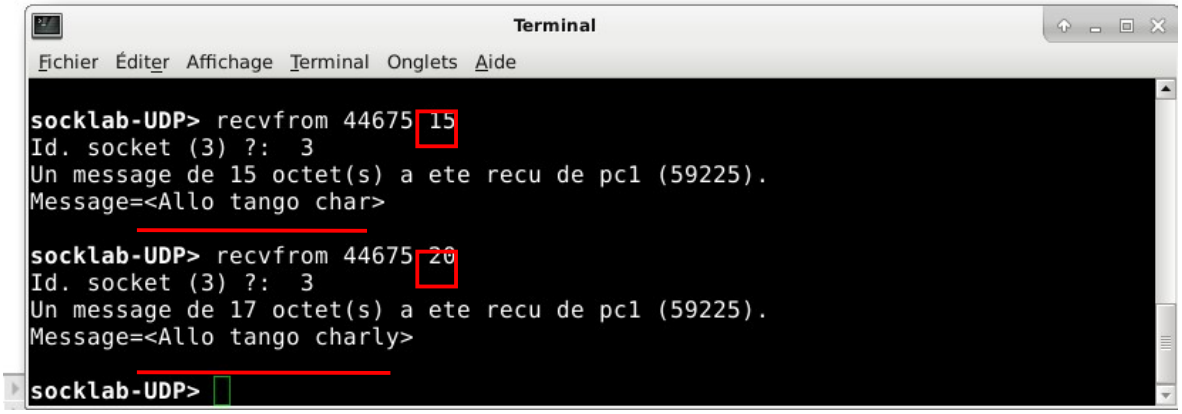
Configuration des machines pour le protocole UDP :



Q2) Le message envoyé par pc1 est : "Allo tango charly". L'émission de ce message engendre un seul paquet UDP. Le paquet est composé des entêtes Ethernet IP et UDP et la zone data contient le message envoyé. Ce dernier est entièrement reçu par le PC2, mais seule la partie correspondante à la taille du buffer de réception de la machine réceptrice est affiché à l'écran.

Message envoyé du PC1 au PC2 : "Allo tango charly" :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.1	192.168.0.2	UDP	60	59225 → 44675 Len=17
2	23.543350	192.168.0.1	192.168.0.2	UDP	60	59225 → 44675 Len=17



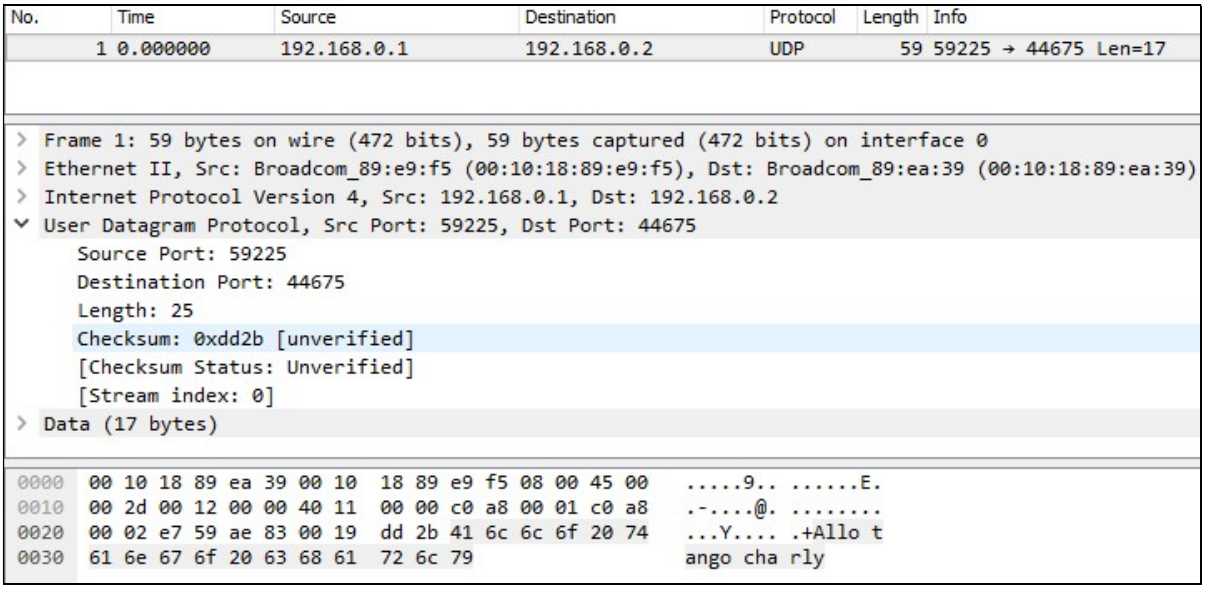
Avec un buffer de 15 octets le message est tronqué (« Allo tango char »). L'augmentation du buffer de réception à 20 octets permet de réceptionner le message complet.

Q3) L'identificateur de socket permet de lister une association locale entre un port et une adresse IP. On peut voir sur l'image suivante que deux ports sont en écoutes pour le protocole UDP sur la machine serveur. Aucune information relative à la socket distante n'est connue.

```
socklab-UDP> =
 Id  Proto   Adresse                  Connexion                  RWX ?
 ------------------------------------------------------------------------
 3   UDP U   *(44675)                 -                          .W.
>4   UDP U   *(12338)                 -                          .W.
```

Q4) UDP donne les paramètres de services à IP, mais ces informations ne sont jamais affichées. Les champs composant l'entête UDP lors de l'envoie d'un paquet au serveur sont les suivants:

- Port source (2bytes) : port de la machine cliente
- Port destination (2bytes) : port de la machine serveur
- Length (2bytes) : Taille de l'entête UDP (8 bytes) + taille data
- Cheksum (2bytes) : représente la validité du paquet UDP



Q5) Lorsque la réception est demandée avant l'émission des données, la machine réceptrice se met en écoute et attend l'arrivée d'un paquet.

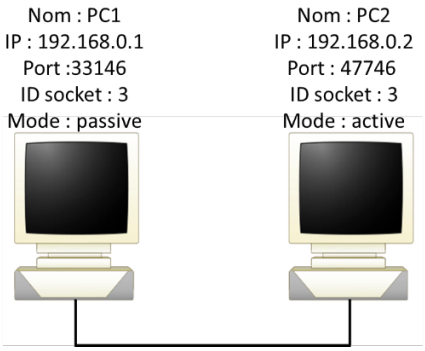
Si plusieurs paquets sont envoyés avant la demande de réception, ils sont stockés dans le buffer de réception de la machine réceptrice tant que ce dernier n'est pas plein. Si les paquets envoyés sont plus grand que la place restante dans le buffer, ces paquets sont ignorés et donc perdus.

Lorsque l'on croise l'émission des paquets chacune des machines reçoit le message qui lui est dédié. La taille du paquet envoyé doit être inférieure ou égal à la taille du buffer de réception. En effet, si sa taille est supérieure, le paquet est ignoré. En plus du buffer de réception, il existe un buffer de lecture. Ce deuxième buffer limite la taille des données lues par paquet. Si la taille des données est supérieure au buffer de lecture les données sont tronquées.

Lorsque le message est envoyé vers une adresse IP qui n'existe pas matériellement sur le réseau, le message se perd. Si le message est envoyé vers un port inexistant sur une machine identifiée, la machine réceptrice renvoie un paquet ICMP de type **Destination Unreachable** et de code **Port** Unreachable. La partie data du paquet ICMP contient les entêtes IP et UDP du paquet précédent.

Etude de la couche transport, le protocole TCP

Configuration des machines pour le protocole TCP :



Remarque : les numéros de ports et ID socket des deux ordinateurs utilisés vont changer au cours du TP car ils sont modifiés à chaque connexion nouvellement réalisée.

Q6) Lors d'une connexion dite TCP entre deux machines, chacune possède un rôle propre. La socket du serveur est dite « passive » car elle est en attente de réception. La seconde socket est dite « active » car c'est elle qui va initier la connexion et le dialogue. Lors de l'établissement de la connexion, la socket « active » envoie un message de demande de connexion à la socket « passive ». La socket « passive » répond à ce message en indiquant qu'elle est prête à dialoguer. La socket « active » confirme à son tour la connexion, il est alors possible de communiquer.

Q7) Les étapes ci dessous détaillent les paquets échangés :

La machine cliente envoie un paquet TPC contenant le flag SYN au serveur afin de demander une connexion.

3 0.000335	192.168.0.2	192.168.0.1	TCP	74 45746 → 33146 [SYN]	Seq=0 Win=65535 Len=0 M
4 0.000659	192.168.0.1	192.168.0.2	TCP	74 33146 → 45746 [SYN, ACK]	Seq=0 Ack=1 Win=65
5 0.000675	192.168.0.2	192.168.0.1	TCP	66 45746 → 33146 [ACK]	Seq=1 Ack=1 Win=66560 L

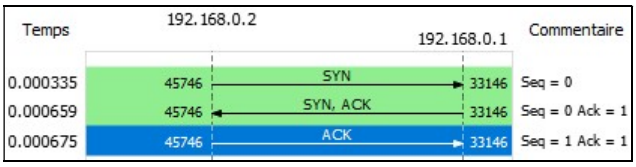
Si le serveur est à l'écoute, il envoie en réponse un paquet à la socket cliente (192.168.0.2 :47546) avec les flags SYN et ACK. Ce paquet informera la machine que le serveur accepte la connexion et qu'il a bien reçu le paquet de demande de connexion.

3 0.000335	192.168.0.2	192.168.0.1	TCP	74 45746 → 33146 [SYN]	Seq=0 Win=65535 Len=0 M
4 0.000659	192.168.0.1	192.168.0.2	TCP	74 33146 → 45746 [SYN, ACK]	Seq=0 Ack=1 Win=65
5 0.000675	192.168.0.2	192.168.0.1	TCP	66 45746 → 33146 [ACK]	Seq=1 Ack=1 Win=66560 L

Enfin le client émet un dernier paquet à destination du serveur avec le flag ACK. Ce paquet indique au serveur que le client a bien reçu la confirmation de connexion.

3 0.000335	192.168.0.2	192.168.0.1	TCP	74 45746 → 33146 [SYN]	Seq=0 Win=65535 Len=0 M
4 0.000659	192.168.0.1	192.168.0.2	TCP	74 33146 → 45746 [SYN, ACK]	Seq=0 Ack=1 Win=65
5 0.000675	192.168.0.2	192.168.0.1	TCP	66 45746 → 33146 [ACK]	Seq=1 Ack=1 Win=66560 L

Résumé de la validation de la connexion en trois étapes, aussi appelée "Three-way handshake".



Q8) Le flag SYN permet d'identifier le paquet comme étant une demande de connexion.

Le tableau ci-dessous décrit les champs sequence number et acknowledgment number lors de l'établissement d'une connexion.

NUMERO ET TYPES DE PAQUETS	SEQUENCE NUMBER	ACKNOWLEDGMENT NUMBER
Paquet 1, SYN (client)	89 be a3 5e (relatif 0)	0
Paquet 2, SYN, ACK (serveur)	70 7d 59 87 (relatif 0)	89 be a3 5f (relatif 1)
Paquet 3, ACK (client)	89 be a3 5f (relatif 1)	70 7d 59 88(relatif 1)

Les sequence number des premiers paquets (1 et 2) envoyés par les deux machines sont déterminés aléatoirement. L'acknowledgment number du premier paquet du client (1) est à zéro car aucun paquet ne le précède. L'acknowledgment number du deuxième paquet (2) est égale au sequence number du premier paquet + 1. Le sequence number du troisième paquet est égale au sequence number du premier paquet + 1 car il s'agit du deuxième paquet envoyé par le client lors de cet échange. Son acknowledgment number correspond au sequence number du deuxième paquet + 1 car il acquitte ce paquet.

Les deux premiers paquets de l'échange possèdent les options maximum segment size à 1460 bytes et window scale à 6 (*64). L'échange de ces paramètres permet de fixer les modalités des futurs dialogues.

Q9) La commande "***accept***" peut être réalisée avant ou après la commande "***connect***". Dans le cas où la commande "***connect***" est réalisée en premier, la connexion est établie sur la socket passive du serveur. La commande "***accept***" entraine chez le serveur la création d'une nouvelle socket spécifique à ce client. Dans le cas inverse, il y a directement création d'une socket spécifique entre le prochain client et le serveur.

Q10) Une connexion est réellement identifiée par les **adresses IP** et les **numéros de port** des machines connectées, d'où un minimum de quatre informations pour deux machines connectées entre elles.

Q11) L'état de la connexion (**state**) définit s'il existe une connexion entre deux sockets. La connexion entre deux sockets est dite : ***ESTABLISHED***. La machine jouant le rôle serveur possède une socket en écoute : ***LISTEN***.

Q12) La machine émettrice crée une socket puis demande la connexion. La machine réceptrice répond avec un paquet contenant les flags "**RST, ACK"**. La machine émettrice supprime alors sa socket à la réception de ce paquet car il lui indique qu'aucun port ne correspond à sa demande.

Etude du séquencement et de la récupération d'erreurs.

Q13) Paramètres de la manipulation :

- Socket Active : pc1 port 55006
- Socket Passive : pc2 port 45362
- Données envoyées : 5000 octets

Ci-dessous le tableau représentant les paquets capturés lors d'un envoi de 5000 octets :

Paquet (numero)	Source	Seq number	Ack number
1	Pc1	1 (0153 1C7B)	1 (ABDD 495E)
2	Pc1	1449 (0153 2223)	1 (ABDD 495E)
3	Pc2	1 (ABDD495E)	2897 (0153 27CB)
4	Pc1	2897 (0153 27CB)	1 (ABDD 495E)
5	Pc2	1 (ABDD 495E)	4345 (0153 2D73)
6	Pc1	4345 (0153 2D73)	1 (ABDD 495E)
7	Pc2	1 (ABDD 495E)	5001 (0153 3003)

Le champ ACK number est utilisé par la machine émettrice pour identifier les paquets correspondant au même message. Sur le tableau ci-dessus l'ACK number des paquets envoyés par pc1 est toujours ABDD 495E. Cet identifiant sur 2 octets est utilisé par la machine réceptrice (pc2) dans le champ SEQ number des paquets qu'elle envoie pour accuser la bonne réception. Le champ SEQ number est utilisé par la machine émettrice pour identifier la position du premier byte de données dans le message. La machine réceptrice met dans ACK number le numéro des derniers bytes +1 qu'elle a reçu. (Uniquement si elle a tout reçu du premier byte au byte en « cours »)

Q14) Il n'y a pas toujours d'acquittement pour chaque paquet de donnée envoyé. En effet un acquittement peut acquitter plusieurs paquets. Dans le tableau précédent on voit que le paquet 3 acquitte les deux paquets précédents. Ce mécanisme permet d'optimiser le débit applicatif.

Q15) Le protocole TCP utilise un timer pour l'envoie de donné. Quand ce timer sonne, un paquet est réémis. Dans le cas ou la machine réceptrice n'est pas à l'écoute, l'envoie d'un nouveau paquet se fait à chaque sonnerie de timer. Pour éviter de saturer le réseau le timer est doublé à chaque réémission d'un même paquet. En comparaison, le protocole UDP ne réclame aucun acquittement des paquets émis. Un paquet émis vers une machine débranchée du réseau n'engendrera aucun message d'erreur, il sera donc définitivement perdu.

Buffer d'émission utilisé pour la récupération d'erreurs :

Q16) L'envoi de paquet de petite taille diminue le débit applicatif du réseau. Un faible buffer d'émission oblige l'émetteur à n'envoyer que des paquets de taille peu importante. Comme il doit attendre l'acquittement de ces derniers pour réémettre, cela réduit considérablement le débit applicatif.

Q17) Le buffer d'émission permet de stocker les paquets émis en attente d'acquittement. Lors de l'acquittement d'un paquet présent dans le buffer, ce dernier est effacé. Dans le cas ou aucun acquittement n'est reçu pour un paquet donné, ce dernier est réémis après un certain temps (sonnerie timer). Si les timers des paquets suivants sonnent à leur tour, les paquets correspondants ne sont pas réémis car le serveur acquitte une séquence complète de byte reçu. En d'autre terme, pour une séquence de n bytes, si le serveur a reçut tous les bytes sauf le $x^{ème}$ (x<n), il acquittera la réception du premier au $x\text{-}1^{ème}$ byte uniquement. Il ne précise pas avoir reçut du $x+1^{ème}$ au $n^{ème}$. Ainsi, il se peut que seul le premier paquet réémis manquait et que sa réception entraine l'acquittement de tous les autres.

Q18) Influence du buffer d'émission sur le débit applicatif :

RTT = 2 * latence = 20 ms

Débit applicatif = taille buffer/RTT = 1000 octets/ 0,02 s = 8kbit / 0,02 s = 400kbit/s
2000 octets/ 0,02 s = 16kbit / 0,02 s = 800kbit/s
10 000 octets/ 0,02 s = 80kbit / 0,02 s = 4Mbit/s

Q19) Pour réaliser les tests, nous avons utilisés un HUB qui possède un débit théorique de 10Mbit/s.
Buffer optimal = débit applicatif théorique * RTT = 10 000 000 * 0,02 = 200 000bits = 25 000 octets

Q20) Si le buffer est trop petit, l'émetteur sera obligé d'attendre l'acquittement de ses paquets avant de pouvoir continuer à émettre. Si la taille du buffer est trop grande, une partie de la mémoire allouée à ce buffer ne sera jamais utilisée.

Q21) Algorithme de mise à jour des champs SEQ et ACK :

Tant que (vrai) faire :
Attendre (évènement) :

Si Emetteur :

- Si événement est envoi d'un paquet, SEQ représente le premier byte de donnée envoyé. ACK représente l'identifiant du message envoyé. Tous les paquets envoyé correspondant au même message possèderont le même ACK.
- Si événement est sonnerie timer, réémission du paquet (dans le cas où il s'agit du premier timer d'une séquence de paquet) avec les mêmes ACK et SEQ.

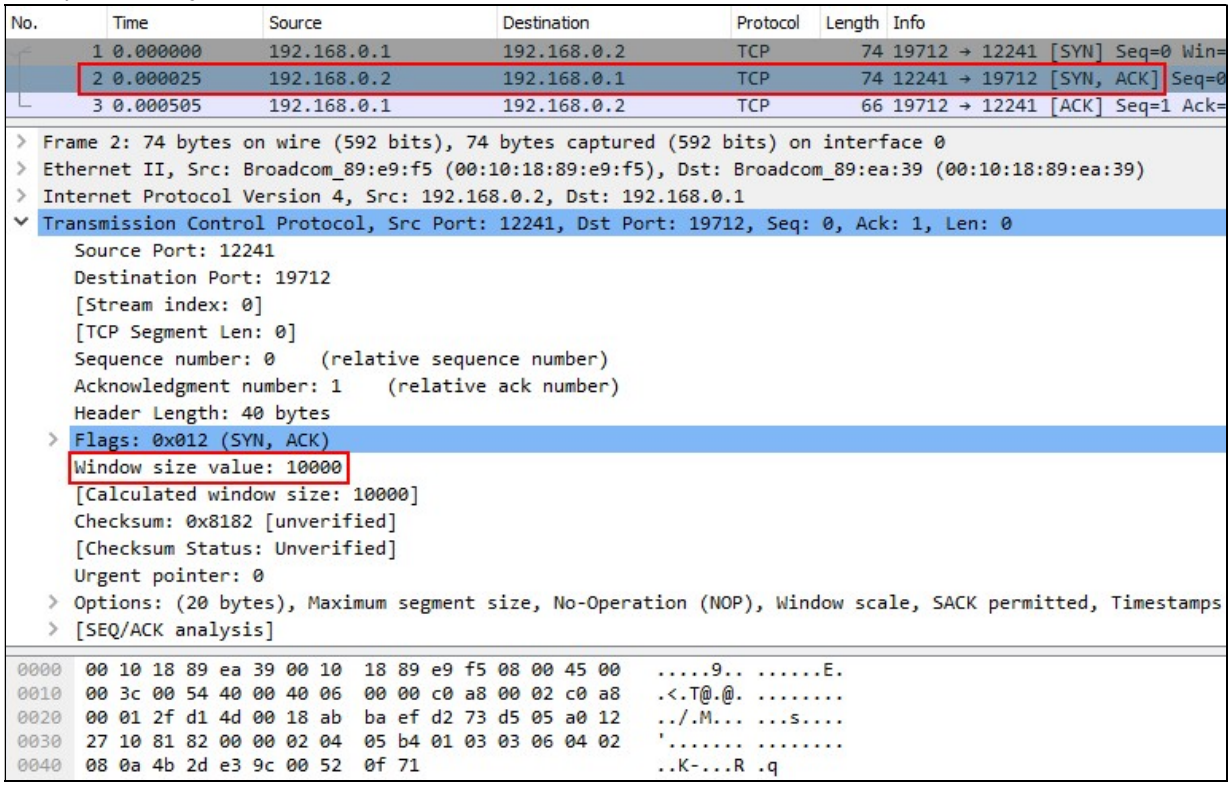
Si Récepteur :

- Si événement est réception d'un paquet. Envoi d'un acquittement avec son champ SEQ représentant l'identifiant du message égal au ACK du paquet reçu. Le champ ACK peut varier selon les circonstances :
  1. Aucune perte de paquet : le champs ACK représente le dernier byte reçut + 1.
  2. Perte de paquet : le dernier byte reçu + 1 (appelé x) du paquet précédent ne correspond pas au premier byte de ce paquet (il manque le paquet intermédiaire). ACK représente alors le dernier byte reçu + 1 (x).
  3. Réception d'un paquet manquant. L'ACK contient l'acquittement du dernier byte + 1 de la séquence complète reçu jusqu'à présent.

Fin tant que ;

Contrôle de flux :

Q22) Les paquets échangés sont les trois paquets habituels (three-ways handshake), issus de l'ouverture de la connexion (voir Question 7). Cependant, le champ window du paquet envoyé par la machine réceptrice (passive) contient la taille modifiée de son buffer de réception. Dans notre cas 10 000 bytes (voir image, paquet n°2).
L'émetteur ne peut pas envoyer plus d'octet que la taille du buffer de réception car le protocole TCP ne permet pas de perte de données. Ainsi, l'émetteur doit s'assurer que l'ensemble des informations envoyées est reçu.



Q23) Le champ window envoyé dans les paquets d'acquittement de la machine réceptrice permet à l'émetteur de ne pas envoyer plus d'octets que la place disponible dans le buffer de réception.

Q24) En théorie, dès la libération d'un byte dans le buffer de réception, la réémission d'un nouveau byte de donnée serait possible. Cependant, le récepteur est effectivement débloqué lorsque la place d'un paquet complet est libérée dans le buffer de réception (1448 bytes). Cela évite de surcharger le réseau avec des paquets de petites tailles qui ont pour effet de réduire le débit applicatif.

Q25) A la réception d'un flux d'octet, la machine réceptrice informe l'émetteur de la fenêtre de réception possible restante. L'émetteur envoie alors autant d'information que cette fenêtre est capable de recevoir. Une fois la fenêtre nulle, l'envoi de bytes supplémentaires n'est possible que si la machine réceptrice informe l'émetteur d'un changement de taille de la fenêtre de réception.
Il se peut que le paquet indiquant le changement de la taille de la fenêtre de réception se perde. La machine émettrice serait alors en attente et le dialogue serait interrompu. Pour éviter ce phénomène, l'émetteur envoie en continu des paquets d'un octet, obligeant l'acquittement de la machine réceptrice qui transmet en même temps la taille de sa fenêtre de réception.

Q26) Il n'est pas intéressant d'avoir un buffer d'émission plus grand que le buffer de réception car le contrôle de flux empêche l'émetteur d'émettre plus que la taille du buffer de réception. Une partie du buffer d'émission sera inutilisée.

Q27) Quel que soit l'ordre de fermeture de la connexion (serveur puis client ou client puis serveur), les séquences de paquets échangées sont identiques :

Un paquet TCP avec les flags FIN et ACK et envoyé par la machine qui ferme la connexion à destination de l'autre machine qui envoie alors un paquet TCP avec flag ACK en réponse :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.2	192.168.0.1	TCP	66	31714 → 12688 [FIN, ACK]
2	0.000032	192.168.0.1	192.168.0.2	TCP	66	12688 → 31714 [ACK] Seq=1
3	8.872753	192.168.0.1	192.168.0.2	TCP	66	12688 → 31714 [FIN, ACK]
4	8.872928	192.168.0.2	192.168.0.1	TCP	66	31714 → 12688 [ACK] Seq=2

Le même échange de paquet est réalisé lors de la fermeture de la connexion par l'autre machine :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.2	192.168.0.1	TCP	66	31714 → 12688 [FIN, ACK]
2	0.000032	192.168.0.1	192.168.0.2	TCP	66	12688 → 31714 [ACK] Seq=1
3	8.872753	192.168.0.1	192.168.0.2	TCP	66	12688 → 31714 [FIN, ACK]
4	8.872928	192.168.0.2	192.168.0.1	TCP	66	31714 → 12688 [ACK] Seq=2

Le flag FIN indique que l'émetteur du paquet a fermé sa connexion.

Dans le cas où la socket d'écoute est fermée par le serveur, un paquet TCP avec les flags RST, ACK est envoyé. Ce paquet informe les machines clientes que le serveur n'est plus à l'écoute.

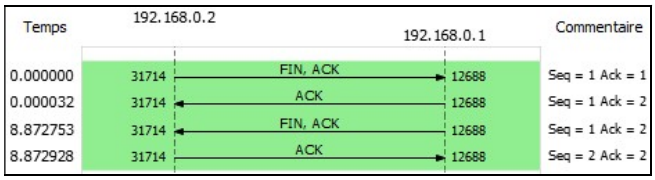
Q28) Ci-dessous un résumé des champs des paquets échangés :

Numéro paquet	Source	Seq number	ACK number
1	Pc2	1 (149E 92E4)	1 (9C64 33A5)
2	Pc1	1 (9C64 33A5)	2 (149E 92E5)
3	Pc1	1 (9C64 33A5)	2 (149E 92E5)
4	Pc2	2 (149E 92E5)	2 (9C64 33A6)

Le premier paquet possède un champ SEQ x et un champ ACK y. L'acquittement de ce paquet (paquet 2) possède un champ SEQ = y et un champ ACK = x+1.

Lors du deuxième dialogue, le principe est le même mais les protagoniste sont inversés.

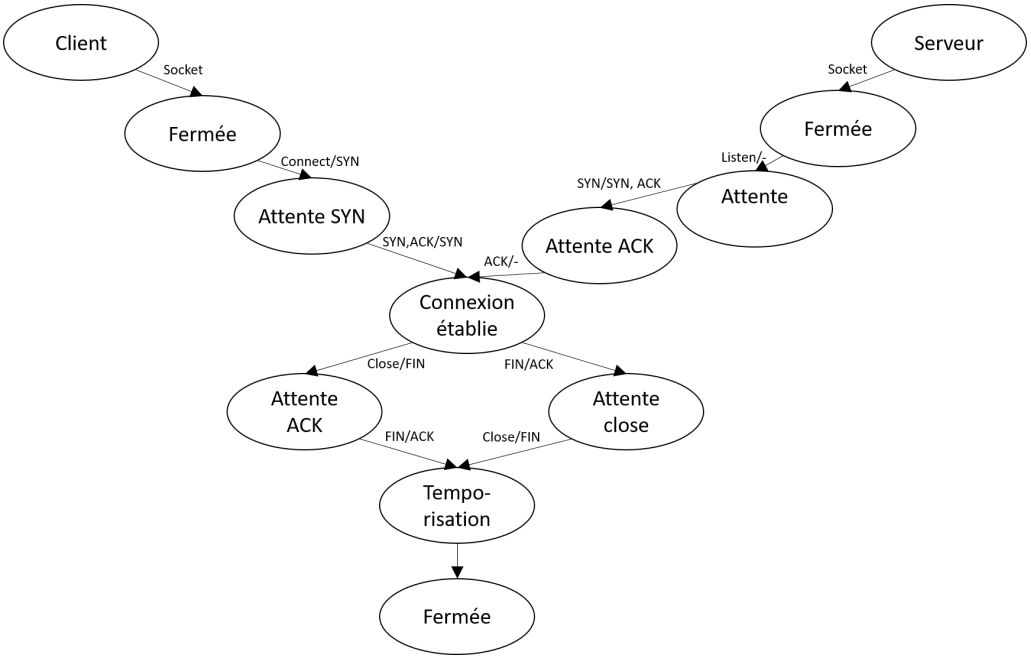
Résumé de la fermeture séquentielle d'une connexion TCP



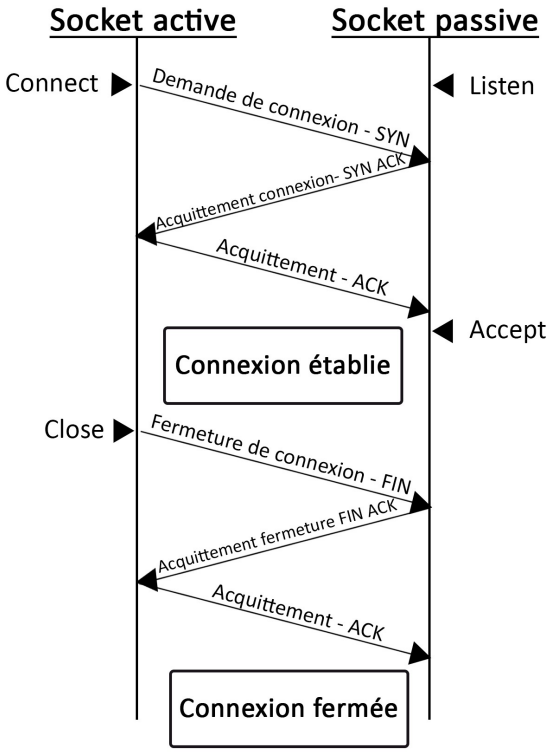
Q29) Après avoir fermé la connexion, si l'autre machine essaye d'envoyer un paquet, la machine ayant fermée sa connexion envoie un paquet TCP avec les flags RST ACK. Cela a pour effet de supprimer l'association port-IP destination dans la socket correspondante.

Remarque : la procédure est la même quel que soit l'ordre de fermeture des connexions

Q30) Automate de Mealy ouverture et fermeture connexion TCP :



Q31) Graph de l'automate :



Q32) Lors d'un dialogue avec la commande talk, les protocoles utilisés sont les protocoles UDP et TCP.

- Lors de l'initialisation de la connexion, un échange de paquet UDP et TCP est observé entre les deux machines. L'appel à la commande talk sur une machine engendre l'échange de 4 paquets UDP (utilisant le port standard de la commande talk: 518) : Voir photo zone 1. L'appel à la commande talk sur la deuxième machine engendre le même échange : Voir photo zones 2 + 2Bis. A la suite de ces échanges, un échange de paquets TCP est réalisé afin de mettre en place la connexion (et mise en place des sockets des deux côtés) : Voir photo zones 3 + 3Bis :

No.	Time	Source	Destination	Protocol	Length	Info	
1	0.000000	192.168.0.1	192.168.0.2	UDP	126	12909 → 518 Len=84	1
2	0.000150	192.168.0.2	192.168.0.1	UDP	66	518 → 12909 Len=24	
3	0.000463	192.168.0.1	192.168.0.2	UDP	126	12909 → 518 Len=84	
4	0.000620	192.168.0.2	192.168.0.1	UDP	66	518 → 12909 Len=24	
5	11.743639	192.168.0.2	192.168.0.1	UDP	126	37269 → 518 Len=84	2
6	11.743946	192.168.0.1	192.168.0.2	UDP	66	518 → 37269 Len=24	
7	11.744103	192.168.0.2	192.168.0.1	TCP	74	46799 → 53031 [SYN]	
8	11.744415	192.168.0.1	192.168.0.2	TCP	74	53031 → 46799 [SYN,	3
9	11.744729	192.168.0.2	192.168.0.1	TCP	66	46799 → 53031 [ACK]	
10	11.744730	192.168.0.2	192.168.0.1	TCP	69	46799 → 53031 [PSH,	
11	11.744886	192.168.0.1	192.168.0.2	UDP	126	12909 → 518 Len=84	2Bis
12	11.745199	192.168.0.2	192.168.0.1	UDP	66	518 → 12909 Len=24	
13	11.745356	192.168.0.1	192.168.0.2	TCP	69	53031 → 46799 [PSH,	3Bis
14	11.845354	192.168.0.2	192.168.0.1	TCP	66	46799 → 53031 [ACK]	

- Lors du dialogue, chaque frappe au clavier engendre un paquet TCP contenant le caractère en zone data. Ce paquet TCP est acquitté une fois reçu par l'autre machine :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.2	192.168.0.1	TCP	67	46799 → 53031 [PSH, ACK] S
2	0.105607	192.168.0.1	192.168.0.2	TCP	66	53031 → 46799 [ACK] Seq=1
3	0.423852	192.168.0.2	192.168.0.1	TCP	67	46799 → 53031 [PSH, ACK] S
4	0.529615	192.168.0.1	192.168.0.2	TCP	66	53031 → 46799 [ACK] Seq=1
5	0.559879	192.168.0.2	192.168.0.1	TCP	67	46799 → 53031 [PSH, ACK] S
6	0.664401	192.168.0.1	192.168.0.2	TCP	66	53031 → 46799 [ACK] Seq=1

Sur cette photo, les 6 paquets correspondent à un échange de trois caractères de PC2 vers PC1.

- Lors de la fermeture de la connexion, seul le protocole TCP est utilisé. Quatre paquets sont échangés. Cet échange correspond à une fermeture séquentielle des deux protagonistes :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.1	192.168.0.2	TCP	66	53031 → 46799 [FIN, ACK] Seq=1 Ack=1
2	0.000307	192.168.0.2	192.168.0.1	TCP	66	46799 → 53031 [ACK] Seq=1 Ack=2 Win=1
3	0.000464	192.168.0.2	192.168.0.1	TCP	66	46799 → 53031 [FIN, ACK] Seq=1 Ack=2
4	0.000621	192.168.0.1	192.168.0.2	TCP	66	53031 → 46799 [ACK] Seq=2 Ack=2 Win=1

Q33)

- La connexion est réalisée par un échange standard de paquets TCP, ainsi qu'un échange de paquets TELNET spécifiques à cette application :

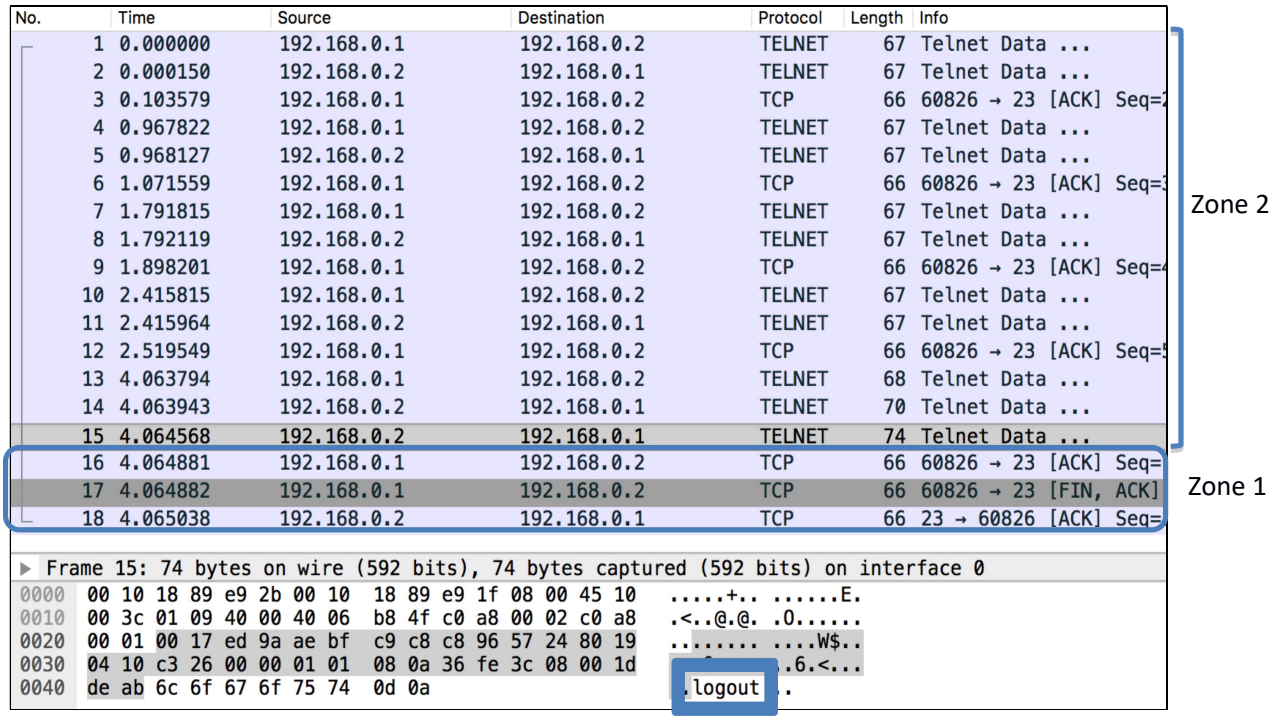
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.1	192.168.0.2	TCP	74	18795 → 23 [SYN] Seq=
2	0.000306	192.168.0.2	192.168.0.1	TCP	74	23 → 18795 [SYN, ACK]
3	0.000307	192.168.0.1	192.168.0.2	TCP	66	18795 → 23 [ACK] Seq=
4	0.001400	192.168.0.1	192.168.0.2	TELNET	99	Telnet Data ...
5	0.004990	192.168.0.2	192.168.0.1	TELNET	69	Telnet Data ...

- Une fois la connexion établie, nous constatons que chaque caractère fait l'objet d'un échange de trois paquets. Un premier paquet TELNET contenant le caractère est envoyé par la machine connectée. Un paquet TELNET est envoyé par la machine source pour afficher le caractère sur la console de la machine connectée. La machine connectée acquitte avec un paquet TCP :

No.	Time	Source	Destination	Protocol	Length	Info	
1	0.000000	192.168.0.1	192.168.0.2	TELNET	67	Telnet Data ...	"l"
2	0.000150	192.168.0.2	192.168.0.1	TELNET	67	Telnet Data ...	
3	0.105925	192.168.0.1	192.168.0.2	TCP	66	60826 → 23 [ACK]	
4	0.295932	192.168.0.1	192.168.0.2	TELNET	67	Telnet Data ...	"s"
5	0.296083	192.168.0.2	192.168.0.1	TELNET	67	Telnet Data ...	
6	0.401853	192.168.0.1	192.168.0.2	TCP	66	60826 → 23 [ACK]	
7	1.543773	192.168.0.1	192.168.0.2	TELNET	68	Telnet Data ...	Entrée
8	1.544081	192.168.0.2	192.168.0.1	TELNET	70	Telnet Data ...	
9	1.649861	192.168.0.1	192.168.0.2	TCP	66	60826 → 23 [ACK]	
10	1.650162	192.168.0.2	192.168.0.1	TELNET	202	Telnet Data ...	Texte à afficher
11	1.755935	192.168.0.1	192.168.0.2	TCP	66	60826 → 23 [ACK]	

Ici, le caractère « l » envoyé par la machine cliente permet à la machine source de stocker la commande en cours de frappe. La machine source renvoie alors ce que la machine cliente doit afficher sur la console. La machine cliente acquitte (TCP). Idem pour le caractère « s » et la frappe « entrée ». Après la frappe « entrée », la machine source envoie le résultat de la commande à afficher sur la console cliente.

- La déconnexion de TELNET : la fermeture de la connexion est similaire à la fermeture d'une connexion TCP (voir photo zone 1). Ces paquets sont cependant précédés par des paquets TELNET dont un contient le message « logout » (voir photo zone 2).



Ci-après le contenu du fichier /etc/services :

- Port telnet 23 protocoles UDP et TCP
- Port talk 518 UDP et TCP
- Port rlogin 541 udp et TCP