

Exercices dirigés

UTC505/USRS4D

-Transport TCP-

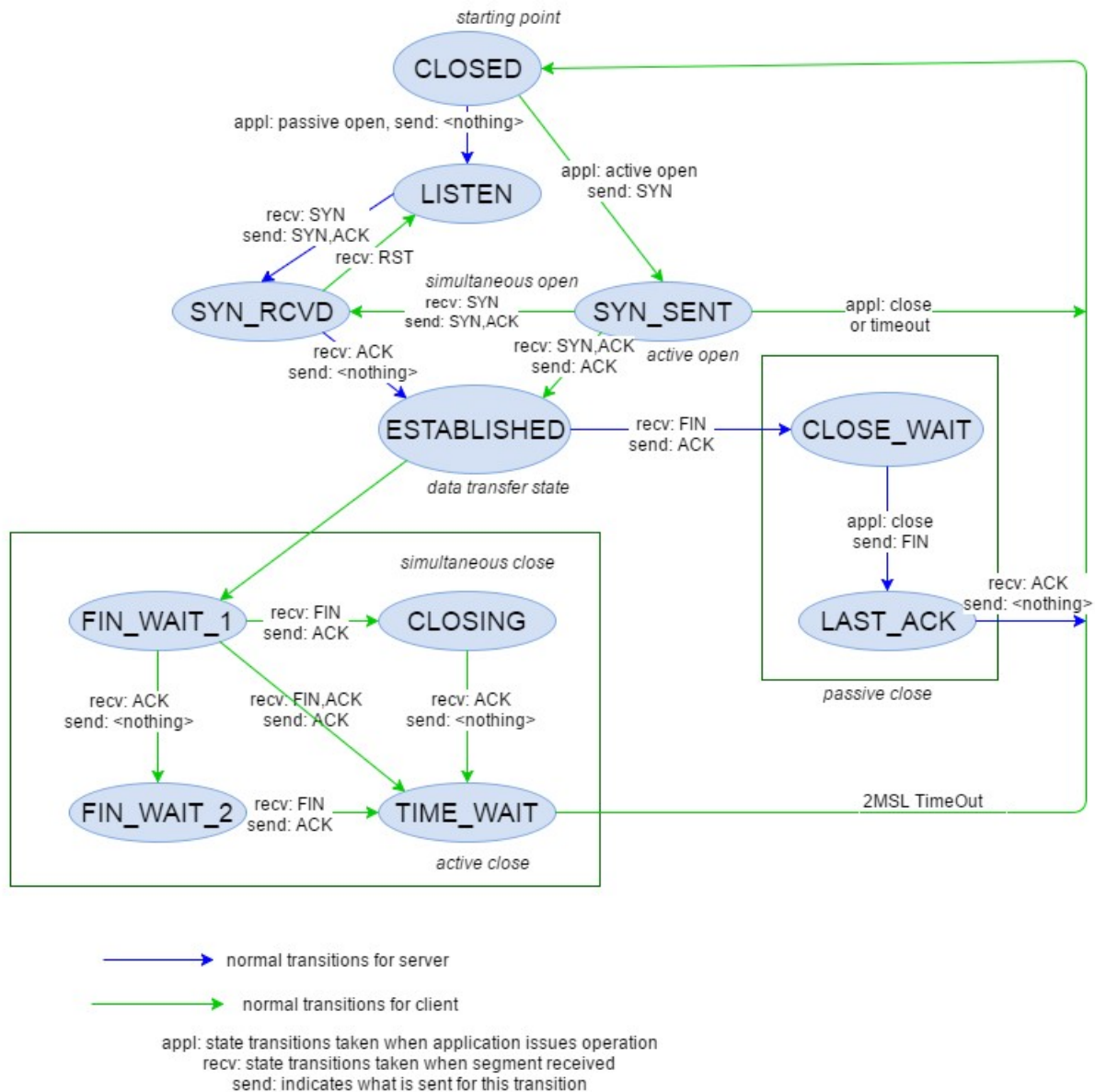
E. Gressier-Soudan

2021-2022

Ce polycopié a été élaboré par l'équipe enseignante "Réseaux et protocoles" à partir d'exercices rédigés par MM. Florin, Gressier-Soudan qu'ils en soient ici remerciés.

ED•TCP

Le diagramme d'état du protocole TCP (Transmission Control Protocol) va servir pendant tout l'ED. Il est issu de la RFC793.



TCP Protocol State Diagram

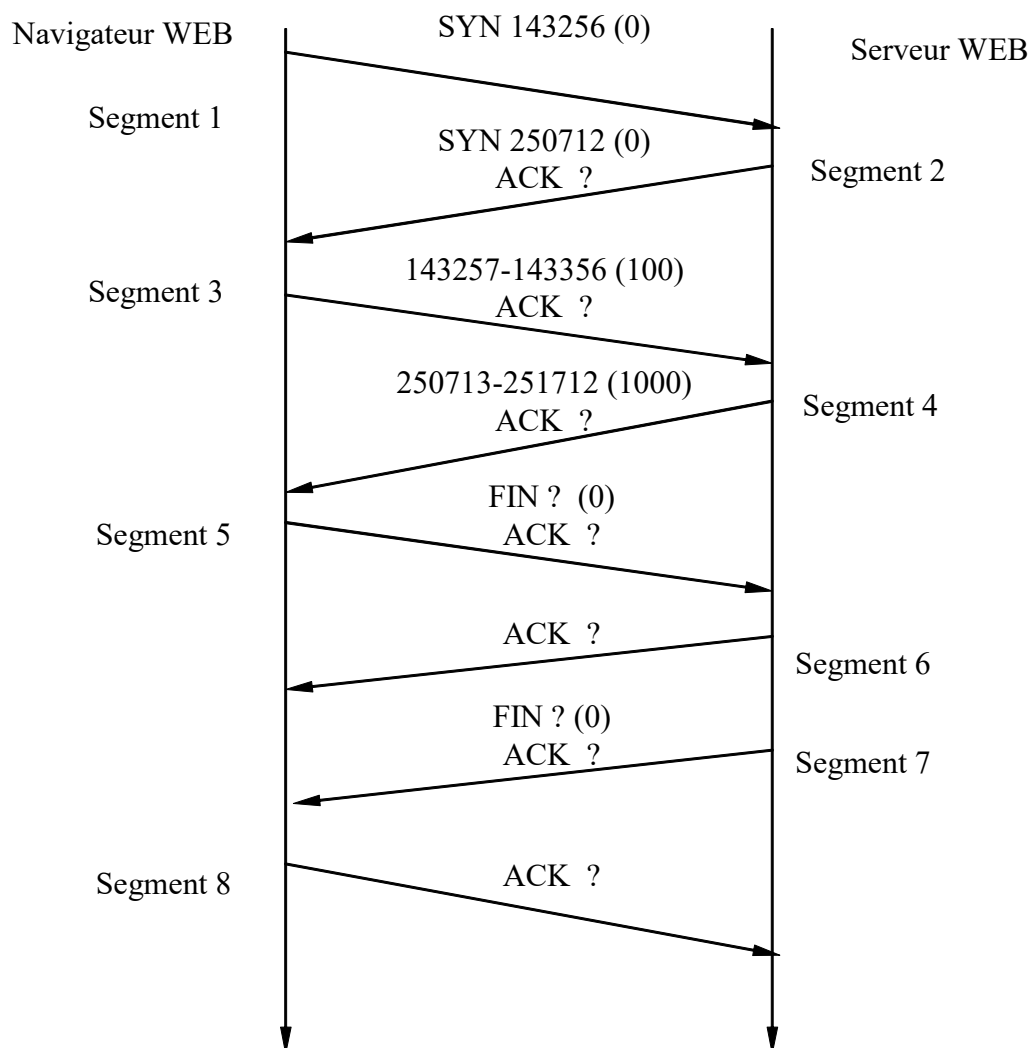
Source : <https://www.thedailyprogrammer.com/2016/09/tcp-state-transition-diagram.html> (consulté le 20/08/2020)

Exercice 1 : Numérotation des octets dans un échange TCP

L'échange TCP de la figure suivante correspond au transfert d'une page WEB entre un navigateur WEB et un serveur WEB. On fait l'hypothèse que la requête à la page WEB fait 100 octets et que la page WEB retournée fait 1000 octets. Il n'y a pas d'erreurs de transmission.

Pour chaque segment de données, différentes informations apparaissent. D'une part la présence d'un ou plusieurs des différents indicateurs comme SYN, FIN, ACK (contrôle). Par ailleurs sur la première ligne deux chiffres sont portés. Le premier chiffre correspond au numéro de séquence du premier octet du segment, le deuxième chiffre correspond au numéro du premier octet du prochain segment à envoyer. Le chiffre entre parenthèses correspond au nombre total d'octets transmis dans le segment. Si le segment est porteur d'un acquittement positif, l'indicateur ACK est mentionné et à côté de lui doit figurer la valeur du champ acquittement du segment TCP.

Complétez les numéros de séquence et les numéros d'acquittement qui manquent sur la figure (qui apparaissent sous forme de point d'interrogation). Indiquez à quoi correspondent les différents segments numérotés de 1 à 8.



Exercice 2 : Echange de données entre un client et un serveur FTP.

Question 1

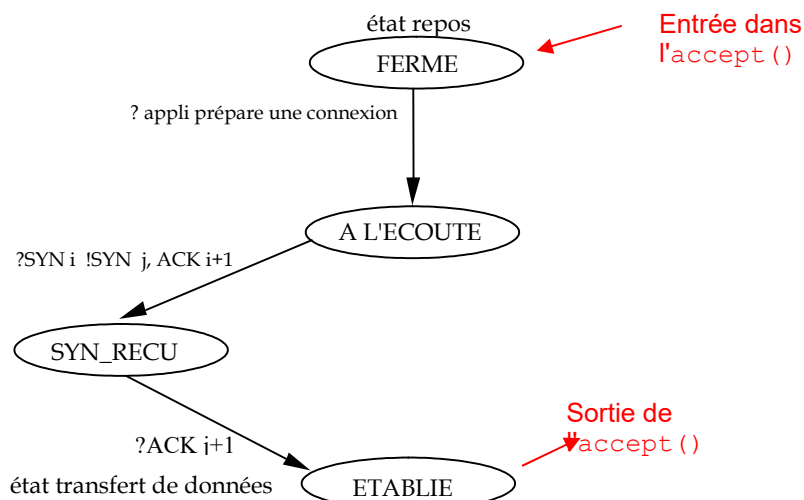
Le protocole FTP, File Transfer Protocol appartient à quelle couche du modèle ISO-OSI ? Justifiez brièvement votre réponse.

Question 2

Expliquez pourquoi le protocole FTP nécessite l'utilisation de TCP, Transmission Control Protocol.

Question 3

Soit la partie d'automate TCP suivante :

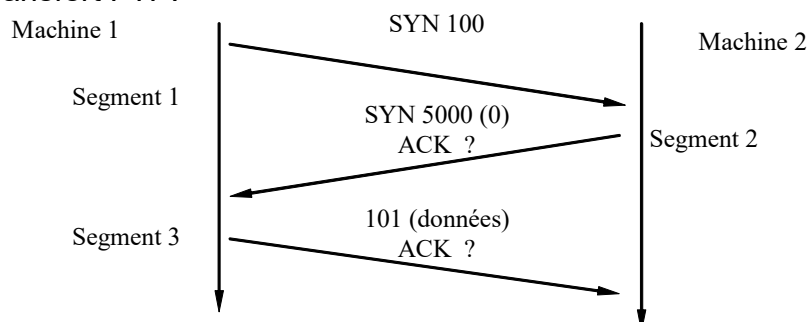


On rappelle que "?" représente une condition, et "!" une action, ici ce sont des envois de messages.

Est-ce que cette partie de l'automate protocolaire correspond à l'ouverture de connexion active ou à l'ouverture de connexion passive ? Justifiez votre réponse.

Question 4

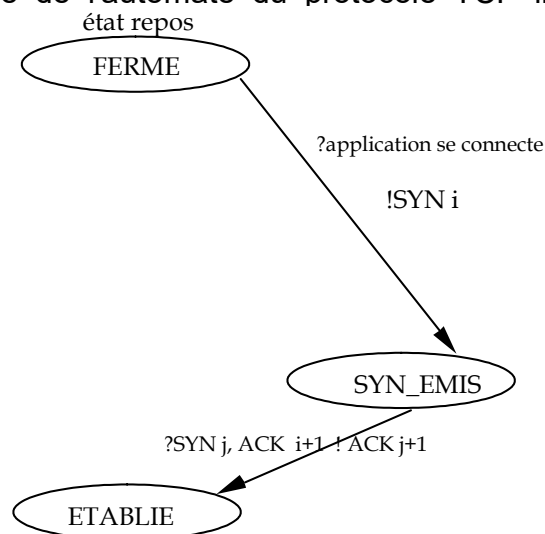
Avec un espion de ligne, on constate l'échange suivant lors d'une ouverture de connexion TCP pour un transfert FTP.



Complétez le schéma ci-dessus en donnant les numéros portés par les acquittements des segments échangés lors de l'ouverture de connexion TCP.

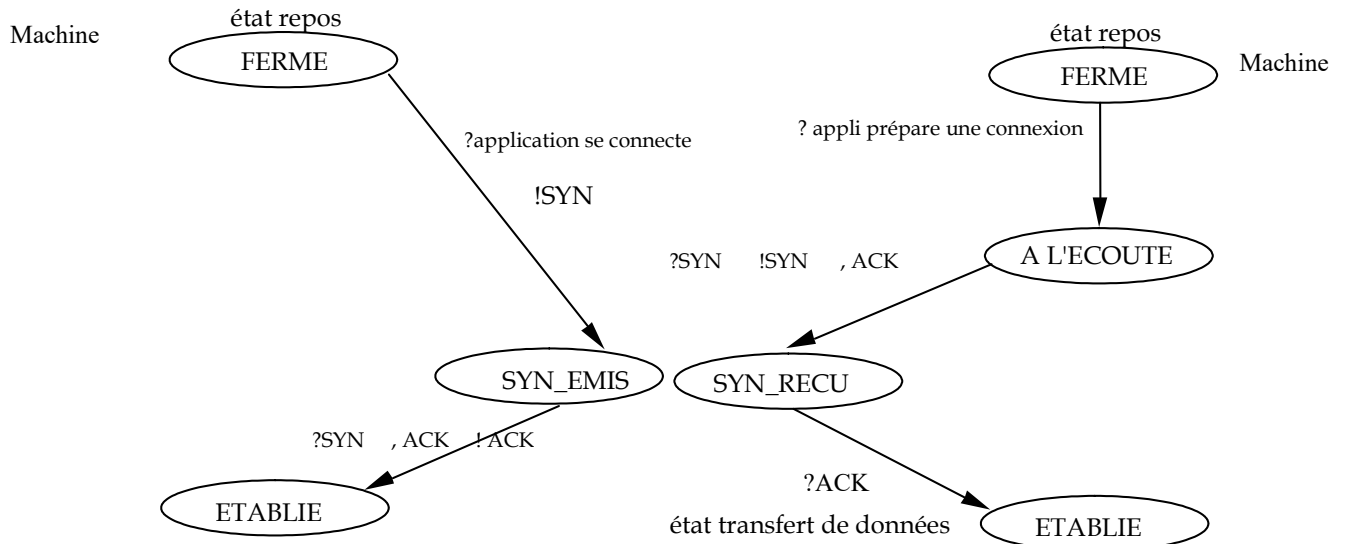
Question 5

Voici une autre partie de l'automate du protocole TCP impliquée dans l'ouverture de connexion.



Sur le schéma ci-après, portez le type des machines (client ou serveur) pour chaque partie d'automate qu'elles exécutent, et les différents numéros de séquence issus de la trace des messages échangés indiquée en question 4. Il faut donner les valeurs de l'échange qui doivent remplacer les variables i et j dans les figures ci-dessus.

Puis, dans l'API Socket, à quels appels système de l'API socket (`listen()`, `close()`, `shutdown()`, `connect()`, `select()`, `accept()`, `read()`, `write()`, `recvfrom()`, `sendto()`...) correspondent les parties d'automates du protocole TCP dans les questions précédentes.

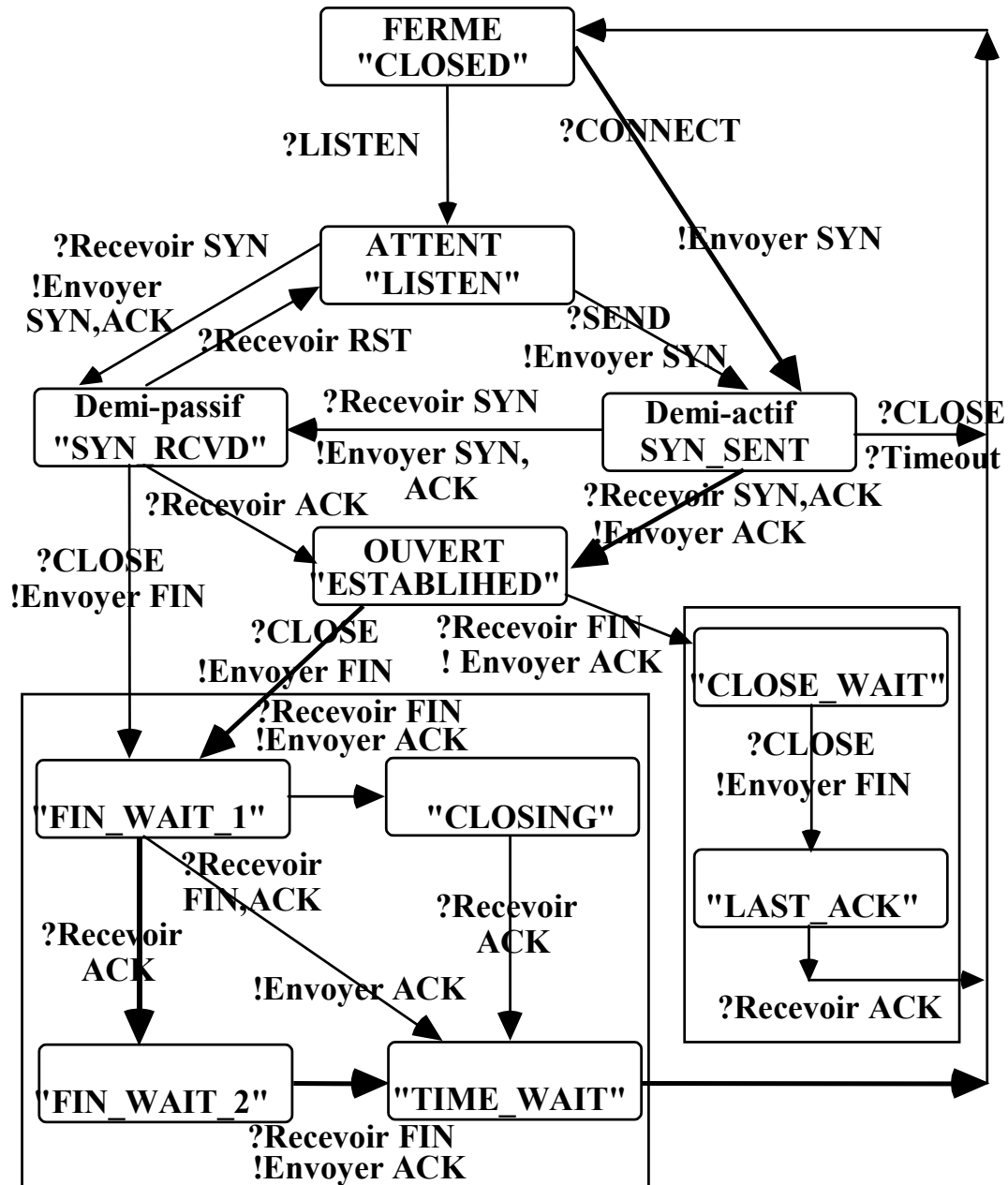


Exercice 3 : Automate protocolaire TCP – fermeture de connexion – Exercice cours (et pas court en longueur)

Rappelez les principales fonctions réalisées par TCP?

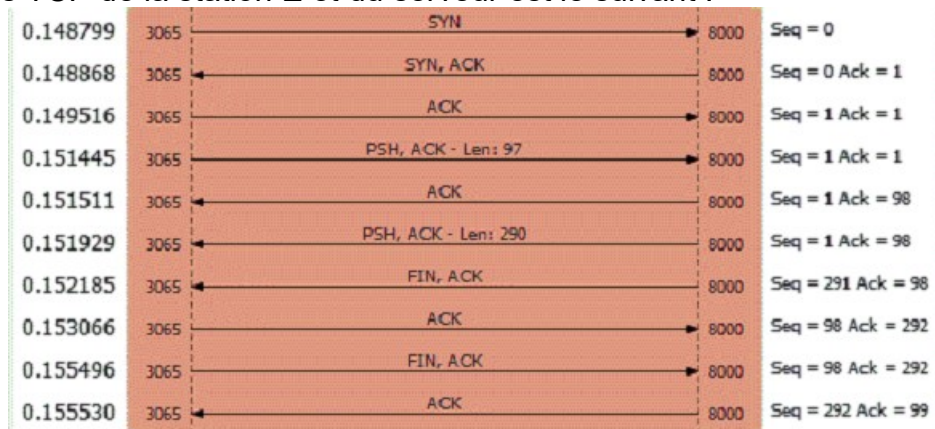
Examinez l'automate en suivant les transitions portées en traits plus épais. Il s'agit d'une ouverture de connexion, un état de transfert en régime connecté, suivi d'une fermeture. Analysez les différentes étapes d'une ouverture de connexion entre un client et un serveur.

Que se passe-t-il lors de la fermeture de connexion?



Exercice 4 : Fonctionnement TCP à travers un échange Web

Un navigateur E, client, 192.168.0.1 (à gauche) demande à un serveur web distant, 192.168.0.2 (à droite) une page via HTTP. Le graphe des échanges entre les automates protocolaires TCP de la station E et du serveur est le suivant :



Ce graphe est obtenu dans Wireshark avec la commande "Graphique des Flux" dans le menu "Statistiques". Quand il y a des données, le champ "Len : valeur" apparaît, sinon, c'est qu'il n'y a aucune donnée dans le segment.

L'identifiant de l'association entre E et le serveur si le numéro de port TCP de E est 3065 et si le numéro de port TCP du serveur est 8000 est :

(192.168.0.1, 3065, TCP, 192.168.0.2, 8000)

Cette association est mise en œuvre par une connexion TCP, comme indiquée. Pour nommer l'instance de la connexion TCP qui porte cette association, il faut ajouter les numéros de séquence initiaux (ISN, Initial Sequence Number) de chacune des entités communicantes aux informations de l'association.

La différence entre la connexion TCP, et l'instance de connexion TCP correspond à la différence : relation entre client et serveur, et, relation effectivement active entre le client et le serveur matérialisée par l'état "ESTABLISHED" des automates protocolaires respectifs.

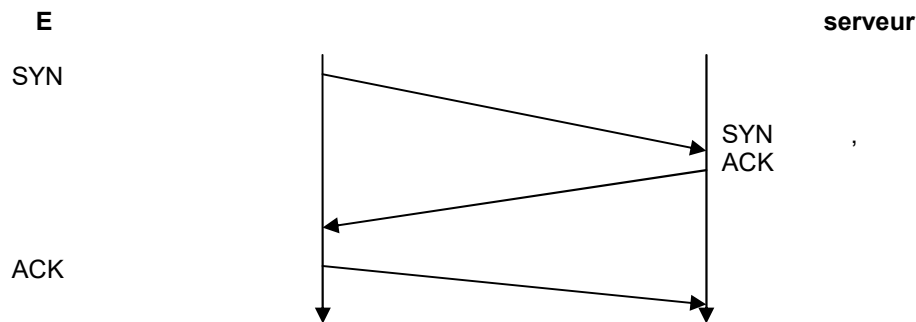
Question 1

Les octets sont comptés en absolu. La valeur du numéro du premier octet envoyé par E pour ouvrir la connexion est : 3753C517 qui vaut en décimal 928236823. La valeur du numéro du premier octet envoyé par le serveur lors de l'ouverture de la connexion est : EC1E0D86 qui vaut en décimal 3961392518.

Donner l'identifiant complet de l'instance de cette connexion entre E et le serveur.

Question 2

Ajouter les numéros d'octets portés par les segments d'ouverture de connexion échangés entre E et le serveur sur le dessin ci-dessous :

**Question 3**

La requête HTTP est portée par un segment de type PSH, ACK. Combien d'octets de données de charge utile pour TCP ont été envoyés de E vers le serveur pour cette requête ? Expliquez brièvement votre résultat. Aidez vous du dessin ci-dessous pour justifier votre réponse.

Question 4

A combien d'octets de charge utile pour TCP correspond la page Web ? Expliquez brièvement votre résultat. Aidez vous du schéma ci-dessous pour justifier votre réponse.

Question 5

Expliquer pourquoi on peut affirmer que la connexion TCP est complètement fermée entre E et le serveur d'après le graphe d'échanges ci-dessus ?

La trace de la trame 4 est découpée en 3 parties pour être complètement visible avec l'entête IP, l'entête TCP, la partie HTTP, et son ensemble en Hexadécimal.

On remarquera que c'est une nouvelle instance de connexion TCP qui est mise en œuvre mais c'est toujours pour la même association ou presque... en effet, le numéro du port client change.

4 0.003290	192.168.0.1	192.168.0.2	HTTP	133 GET / HTTP/1.0
Frame 4: 133 bytes on wire (1064 bits), 133 bytes captured (1064 bits)				
Ethernet II, Src: SmcEther_68:8b:fb (00:e0:29:68:8b:fb), Dst: 3Com_1b:07:fa (00:20:af:1b:07:fa)				
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.2				
0100 = Version: 4				
.... 0101 = Header Length: 20 bytes (5)				
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)				
Total Length: 119				
Identification: 0xc1c7 (49607)				
> Flags: 0x4000, Don't fragment				
Fragment offset: 0				
Time to live: 64				
Protocol: TCP (6)				
Header checksum: 0xf765 [validation disabled]				
[Header checksum status: Unverified]				
Source: 192.168.0.1				
Destination: 192.168.0.2				

Transmission Control Protocol, Src Port: 3064, Dst Port: 8000, Seq: 1, Ack: 1, Len: 79

Source Port: 3064

Destination Port: 8000

[Stream index: 0]

[TCP Segment Len: 79]

Sequence number: 1 (relative sequence number)

Sequence number (raw): 3929668180

[Next sequence number: 80 (relative sequence number)]

Acknowledgment number: 1 (relative ack number)

Acknowledgment number (raw): 137897116

0101 = Header Length: 20 bytes (5)

▼ Flags: 0x018 (PSH, ACK)

000. = Reserved: Not set

...0 = Nonce: Not set

.... 0... = Congestion Window Reduced (CWR): Not set

.... .0.. = ECN-Echo: Not set

.... ..0. = Urgent: Not set

.... ...1 = Acknowledgment: Set

.... 1... = Push: Set

....0.. = Reset: Not set

....0. = Syn: Not set

....0 = Fin: Not set

[TCP Flags:AP...]

Window size value: 17520

[Calculated window size: 17520]

[Window size scaling factor: -2 (no window scaling used)]

Checksum: 0x3b9f [unverified]

[Checksum Status: Unverified]

Urgent pointer: 0

> [SEQ/ACK analysis]

> [Timestamps]

TCP payload (79 bytes)

▼ Hypertext Transfer Protocol

▼ GET / HTTP/1.0\r\n

> [Expert Info (Chat/Sequence): GET / HTTP/1.0\r\n]

Request Method: GET

Request URI: /

Request Version: HTTP/1.0

User-Agent: Wget/1.5.3\r\n

Host: 192.168.0.2:8000\r\n

Accept: */*\r\n

\r\n

[Full request URI: <http://192.168.0.2:8000/>]

[HTTP request 1/1]

0000	00 20 af 1b 07 fa 00 e0 29 68 8b fb 08 00 45 00	-)h E .
0010	00 77 c1 c7 40 00 40 06 f7 65 c0 a8 00 01 c0 a8	-w . . @ . @ . -e
0020	00 02 0b f8 1f 40 ea 39 fa 54 08 38 24 9c 50 18	- @ . 9 -T . 8 \$. P .
0030	44 70 3b 9f 00 00 47 45 54 20 2f 20 48 54 50	Dp ; . . . G E T / H T T P
0040	2f 31 2e 30 0d 0a 55 73 65 72 2d 41 67 65 6e 74	/ 1 . 0 . . U s e r - A g e n t
0050	3a 20 57 67 65 74 2f 31 2e 35 2e 33 0d 0a 48 6f	: W g e t / 1 . 5 . 3 . - H o
0060	73 74 3a 20 31 39 32 2e 31 36 38 2e 30 2e 32 3a	s t : 1 9 2 . 1 6 8 . 0 . 2 :
0070	38 30 30 30 0d 0a 41 63 63 65 70 74 3a 20 2a 2f	8 0 0 0 . . A c c e p t : * /
0080	2a 0d 0a 0d 0a	*

Question 6 :

Est-ce que l'URL contenue dans cette requête est correctement formée ? Est-ce que la requête HTTP est correcte ?

Question 7 : Puzzle sur une communication TCP

Remettre les échanges listés sur chaque ligne dans le bon ordre. La suite correcte représente l'enchaînement des trames qui montre le déroulement d'une connexion TCP entre (192.168.0.1, 3064), navigateur Web, et (192.168.0.2, 8000), serveur Web. On doit retrouver dans l'ordre les 3 phases : ouverture de connexion-transfert de données-fermeture de connexion. On vous donne uniquement l'ordre du premier SYN et la position de la requête GET dans l'enchaînement.

Attention, dans les échanges Wireshark les numéros de séquence sont en relatif (ils démarrent à 0), ça facilite.

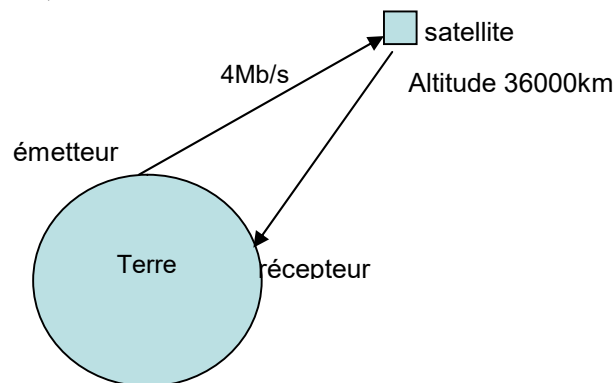
1 -	192.168.0.1	192.168.0.2	TCP	60 3064 → 8000 [SYN] Seq=0 Win=16384 Len=0 MSS=1460
	192.168.0.2	192.168.0.1	TCP	54 8000 → 3064 [ACK] Seq=296 Ack=81 Win=32120 Len=0
	192.168.0.1	192.168.0.2	TCP	60 3064 → 8000 [ACK] Seq=1 Ack=1 Win=17520 Len=0
	192.168.0.1	192.168.0.2	TCP	60 3064 → 8000 [ACK] Seq=80 Ack=296 Win=17520 Len=0
	192.168.0.2	192.168.0.1	TCP	348 8000 → 3064 [PSH, ACK] Seq=1 Ack=80 Win=32120 Len=294 [TCP se
	192.168.0.2	192.168.0.1	TCP	54 8000 → 3064 [ACK] Seq=1 Ack=80 Win=32120 Len=0
4 -	192.168.0.1	192.168.0.2	HTTP	133 GET / HTTP/1.0
	192.168.0.2	192.168.0.1	TCP	54 8000 → 3064 [FIN, ACK] Seq=295 Ack=80 Win=32120 Len=0
	192.168.0.2	192.168.0.1	TCP	58 8000 → 3064 [SYN, ACK] Seq=0 Ack=1 Win=32120 Len=0 MSS=1460
	192.168.0.1	192.168.0.2	TCP	60 3064 → 8000 [FIN, ACK] Seq=80 Ack=296 Win=17520 Len=0

Expliquer brièvement comment vous trouvez votre réponse.

Exercice 5 : Fenêtre d'Anticipation et communication TCP satellite. A faire soi-même, facile.

Soit un satellite en orbite géo-stationnaire. Il est situé à 36000 km de la terre. Les messages font 1500 octets et le débit nominal de la liaison satellite est de 4 Mb/s ($4 \cdot 10^6 \text{ b/s}$)¹ en montée (upload) et en descente (download)².

On rappelle que le délai de propagation d'une onde électromagnétique dans l'espace est voisin de la vitesse de la lumière, soit 300000 km/s.



Les 1500 octets comptent pour : l'entête IP (20 octets), l'entête TCP (20 octets), et les données (1460 octets).

Pour construire une trame, on ajoute aux 1500 octets les informations de gestion : l'entête et le CRC de la trame qui comptent pour 20 octets (Longueur de la trame, Type de la trame, Adresse destination de liaison, Type de la charge acheminée, CRC).

On néglige dans les calculs la prise en compte du préambule.

1. Quelle est la taille d'une trame envoyée sur la liaison satellite ? Expliquez très brièvement votre calcul.
2. Quel est le délai de propagation de la terre au satellite en millisecondes (ms) ? Expliquez très brièvement votre calcul, il compte pour le calcul de la latence entre émetteur et récepteur.
3. Quel est le temps, en ms, nécessaire pour émettre une trame ?
4. Combien de temps met la trame pour arriver complètement au destinataire en passant par le satellite ? A peine arrivée sur le satellite, elle est renvoyée vers le récepteur sur la terre. Le satellite, malgré toute l'intelligence qu'il peut contenir n'est considéré qu'un élément de la couche physique, c'est si on veut résumer un répéteur. Justifiez très brièvement votre calcul.

¹ Attention, on utilise de façon systématique le Mo qui exprime une puissance 1024 , alors qu'officiellement on devrait utiliser dans cette circonstance 1024^2 . Pour plus d'information consulter : https://fr.wikipedia.org/wiki/Pr%C3%A9fixe_binaire, merci à Loïc Cartier, auditeur d'UTC505 1^{er} semestre 2020-2021 pour ce lien. Dans l'exercice il est bien question de mégabit et pas de Mibi.

² En réalité, le débit de montée est largement inférieur au débit de descente. Mais on fait cette hypothèse pour simplifier l'exercice.

On utilise TCP pour envoyer les données. On suppose qu'on envoie un segment puis le récepteur l'acquitte. Tant que l'émetteur n'a pas reçu l'acquittement, il n'envoie pas de nouveau segment. L'acquittement est un segment qui ne contient aucune donnée. On suppose qu'il n'y a pas de perte de segment lors des transmissions, ce qui est une hypothèse optimiste.

5. Quelle est la taille d'une trame qui contient un segment d'acquittement ?

6. Quel est le temps, en ms, nécessaire pour transmettre ce segment d'acquittement vers la base source ?

7. Combien de temps se passe-t-il entre le début d'émission du segment de données et la réception de son acquittement par la source ? Justifiez très brièvement votre calcul.

8. Quel est le débit utile avec cette solution qui consiste à envoyer segment par segment, et à ne pas envoyer le segment suivant tant que l'acquittement du précédent n'est pas arrivé à l'émetteur.

On rappelle que le débit utile se calcule en faisant le rapport taille d'un message en bits sur temps pour le transmettre en secondes.

9. Quelle est la valeur de l'efficacité de cette liaison ? Qu'en concluez vous ?

On rappelle que l'efficacité d'une ligne est le rapport débit utile sur débit nominal.

On propose une autre solution. On met en place une fenêtre d'anticipation à l'émission dans TCP. Ce qui veut dire qu'on envoie plusieurs segments avant de recevoir le premier acquittement.

10. Quelle pourrait la valeur K de cette fenêtre d'anticipation, exprimée en nombre de messages, qui maximise l'efficacité de la communication satellitaire pour qu'elle s'approche des 100%.

11. TCP gère une fenêtre d'anticipation, mais exprimée en octets. Quelle valeur proposeriez vous pour configurer la fenêtre d'émission de TCP à partir de la réponse à la question précédente.

Attention, la taille de la fenêtre doit s'exprimer sous la forme d'un chiffre qui est une puissance de 2, on la comptera en Kilo octets (1024 octets).

12. La taille d'une fenêtre TCP est au maximum de 65536 octets (numérotés de 0 à 65535) car c'est un champ sur 16 bits (et au minimum de 2 octets). Est-ce suffisant ? Quel mécanisme imagineriez vous pour augmenter la taille de la fenêtre sans modifier l'entête d'un segment TCP.

Exercice 6 : Etude de Trace TCP portant un échange http (à faire en autonomie)

On s'intéresse maintenant à TCP qui porte des échanges HTTP. La trace ci-après est relevée au niveau de l'outil Wireshark :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.00000000	Inventec_9a:4d:aa	Broadcast	ARP	42	who has 163.173.228.17? Tell 163.173.231.107
2	0.00060600	DellEsgP_f9:ad:5a	Inventec_9a:4d:aa	ARP	60	163.173.228.17 is at 00:0f:1f:f9:ad:5a
3	0.00062700	163.173.231.107	163.173.228.17	TCP	66	50074 > http [SYN] Seq=0 win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
4	0.00186700	163.173.228.17	163.173.231.107	TCP	66	http > 50074 [SYN, ACK] Seq=0 Ack=1 win=5840 Len=0 MSS=1460 SACK_PERM=1 WS=4
5	0.00191200	163.173.231.107	163.173.228.17	TCP	54	50074 > http [ACK] Seq=1 Ack=1 win=65700 Len=0
6	0.00241400	163.173.231.107	163.173.228.17	HTTP	544	GET / HTTP/1.1
7	0.00313900	163.173.228.17	163.173.231.107	TCP	60	http > 50074 [ACK] Seq=1 Ack=491 win=6912 Len=0
8	0.00360700	163.173.228.17	163.173.231.107	TCP	353	[TCP segment of a reassembled PDU]
9	0.00388500	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]
10	0.00416400	163.173.231.107	163.173.228.17	TCP	54	50074 > http [ACK] Seq=491 Ack=1760 win=65700 Len=0
11	0.00500200	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]
12	0.00503000	163.173.228.17	163.173.231.107	HTTP	74	HTTP/1.1 200 OK (text/html)
13	0.00528500	163.173.231.107	163.173.228.17	TCP	54	50074 > http [ACK] Seq=491 Ack=3240 win=65700 Len=0
14	0.03563900	163.173.231.107	163.173.228.17	HTTP	559	GET /logo_cnam.gif HTTP/1.1
15	0.03661300	163.173.228.17	163.173.231.107	TCP	353	[TCP segment of a reassembled PDU]
16	0.03679700	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]
17	0.03691400	163.173.231.107	163.173.228.17	TCP	54	50074 > http [ACK] Seq=996 Ack=4999 win=65700 Len=0
18	0.03703200	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]
19	0.03787100	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]
20	0.03791500	163.173.231.107	163.173.228.17	TCP	54	50074 > http [ACK] Seq=996 Ack=7919 win=65700 Len=0
21	0.03792400	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]
22	0.03797300	163.173.228.17	163.173.231.107	HTTP	1275	HTTP/1.1 200 OK (GIF87a)
23	0.03800800	163.173.231.107	163.173.228.17	TCP	54	50074 > http [ACK] Seq=996 Ack=10600 win=65700 Len=0
24	15.0372060	163.173.228.17	163.173.231.107	TCP	60	http > 50074 [FIN, ACK] Seq=10600 Ack=996 win=7984 Len=0
25	15.0372830	163.173.231.107	163.173.228.17	TCP	54	50074 > http [ACK] Seq=996 Ack=10601 win=65700 Len=0
26	15.0375270	163.173.231.107	163.173.228.17	TCP	54	50074 > http [FIN, ACK] Seq=996 Ack=10601 win=65700 Len=0
27	15.0379470	163.173.228.17	163.173.231.107	TCP	60	http > 50074 [ACK] Seq=10601 Ack=997 win=7984 Len=0

Attention, les numéros de séquence affichés sont relatifs, le véritable numéro de séquence dans le segment est transformé pour numéroté les octets d'une connexion à partir de 0. Cette règle est appliquée dans les deux sens.

Pour vous aider dans votre analyse des échanges, vous pouvez vous appuyer sur la trace WireShark correspondante obtenue avec la commande "Follow TCP stream" (en **violet** la requête client, et en **turquoise** la réponse du serveur) :

```
GET / HTTP/1.1
Host: lmil7.cnam.fr
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:14.0) Gecko/20100101 Firefox/14.0.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Connection: keep-alive
HTTP/1.1 200 OK
Date: Wed, 27 Mar 2013 16:12:20 GMT
Server: Apache/2.0.53 (Linux/SUSE)
Last-Modified: Tue, 20 Sep 2005 16:16:03 GMT
ETag: "1734016-b7c-401366056b6c0"
Accept-Ranges: bytes
Content-Length: 2940
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html
<HTML>
<HEAD>
  <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
  <META NAME="GENERATOR" CONTENT="Mozilla/4.03 [fr] (Win95; I) [Netscape]">
  <TITLE>Laboratoire de Micro-Informatique du CNAM-PARIS</TITLE>
</HEAD>
<BODY TEXT="#330000" BGCOLOR="#FFFFBB" LINK="#0000EE" VLINK="#FF0000" ALINK="#FF0000">
&nbsp;
... une partie de cette page a été retirée de la trace car elle ne présentait aucun intérêt pour répondre...
<HR>Pour vos commentaires et questions : <B><A HREF="mailto:jbb@cnam.fr">Web
Master</A></B>
<BR>Dernière modification : Monday, November 03 1997 10:58:19
</BODY>
</HTML>
GET /logo_cnam.gif HTTP/1.1
Host: lmil7.cnam.fr
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:14.0) Gecko/20100101 Firefox/14.0.1
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Connection: keep-alive
Referer: http://lmil7.cnam.fr/
HTTP/1.1 200 OK
Date: Wed, 27 Mar 2013 16:12:20 GMT
Server: Apache/2.0.53 (Linux/SUSE)
Last-Modified: Tue, 04 Nov 1997 08:46:02 GMT
```



```
ETag: "1734010-1b95-31f1cb9824680"
Accept-Ranges: bytes
Content-Length: 7061
Keep-Alive: timeout=15, max=99
Connection: Keep-Alive
Content-Type: image/gif
GIF87a.....}}}}{yyywwuuu
...la suite de la trace a été enlevée elle ne présente pas d'intérêt pour l'exercice
```

Question 1

Question 1.1

Les lignes 3, 4, 5 matérialisent l'ouverture de connexion, expliquer :

- pourquoi ces lignes correspondent à l'ouverture de connexion,
- qui fait l'ouverture de connexion passive et qui fait donc l'ouverture de connexion active
- pourquoi, selon vous, le MSS (Maximum Segment Size) vaut 1460,

Question 1.2

Les lignes 24, 25, 26, 27 matérialisent la fermeture de connexion, expliquer :

- pourquoi ces lignes correspondent à la fermeture de connexion,
- combien de messages pour fermer la connexion des deux côtés,
- finalement pour obtenir la page qui s'affiche sur le navigateur, combien de connexions ont été ouvertes

Question 1.3

Combien de données ont été échangées :

- du client vers le serveur
- du serveur vers le client

Question 1.4

Dans le segment ci-dessous, trame 3 de la trace, retrouvez le numéro de séquence réel du SYN dans la trame en octets au format hexadécimal.

```
Ethernet II, Src: Inventec 9a:4d:aa (00:26:6c:9a:4d:aa), Dst: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a)
  Destination: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a)
    Address: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a)
      .... ..0. .... .. = LG bit: Globally unique address (factory default)
```




```

.....0 ..... = IG bit: Individual address (unicast)
Source: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)
Address: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)
.....0 ..... = LG bit: Globally unique address (factory default)
.....0 ..... = IG bit: Individual address (unicast)
Type: IP (0x0800)
Internet Protocol Version 4, Src: 163.173.231.107 (163.173.231.107), Dst: 163.173.228.17 (163.173.228.17)
Version: 4
Header length: 20 bytes
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
Total Length: 52
Identification: 0x4d92 (19858)
Flags: 0x02 (Don't Fragment)
    0... .... = Reserved bit: Not set
    .1.. .... = Don't fragment: Set
    ..0. .... = More fragments: Not set
Fragment offset: 0
Time to live: 128
Protocol: TCP (6)
Header checksum: 0x9a59 [correct]
    [Good: True]
    [Bad: False]
Source: 163.173.231.107 (163.173.231.107)
Destination: 163.173.228.17 (163.173.228.17)
Transmission Control Protocol, Src Port: 50074 (50074), Dst Port: http (80), Seq: 0, Len: 0
Source port: 50074 (50074)
Destination port: http (80)
[Stream index: 0]
Sequence number: 0 (relative sequence number)
Header length: 32 bytes
Flags: 0x002 (SYN)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    .... 0... = Congestion Window Reduced (CWR): Not set
    .... .0.. = ECN-Echo: Not set
    .... ..0. = Urgent: Not set
    .... ...0 = Acknowledgment: Not set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
    .... .... ..1. = Syn: Set
    .... .... ...0 = Fin: Not set
Window size value: 8192

```

```

[Calculated window size: 8192]
Checksum: 0xcfa1 [validation disabled]
Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP),
SACK permitted
  Maximum segment size: 1460 bytes
    Kind: MSS size (2)
    Length: 4
    MSS Value: 1460
  No-Operation (NOP)
    Type: 1
    0... .... = Copy on fragmentation: No
    .00. .... = Class: Control (0)
    ...0 0001 = Number: No-Operation (NOP) (1)
  Window scale: 2 (multiply by 4)
    Kind: Window Scale (3)
    Length: 3
    Shift count: 2
    [Multiplier: 4]
  No-Operation (NOP)
    Type: 1
    0... .... = Copy on fragmentation: No
    .00. .... = Class: Control (0)
    ...0 0001 = Number: No-Operation (NOP) (1)
  No-Operation (NOP)
    Type: 1
    0... .... = Copy on fragmentation: No
    .00. .... = Class: Control (0)
    ...0 0001 = Number: No-Operation (NOP) (1)
  TCP SACK Permitted Option: True
    Kind: SACK Permission (4)
    Length: 2

```

Les entêtes ont été colorées (MAC, IP, TCP).

0000	00 0f 1f f9 ad 5a 00 26 6c 9a 4d aa 08 00 45 00Z.&l.M...E.
0016	00 34 4d 92 40 00 80 06 9a 59 a3 ad e7 6b a3 ad	.4M.@....Y...k..
0032	e4 11 c3 9a 00 50 3c ac 6c 05 00 00 00 00 80 02P<.l.....
0048	20 00 cf a1 00 00 02 04 05 b4 01 03 03 02 01 01
0064	04 02	..

On rappelle la grille d'analyse d'une entête TCP :

identifiant émetteur				identifiant récepteur				20 octets				
no de séquence du premier octet émis contenu dans ce segment												
no d'acquittement : no de séquence du prochain octet à recevoir par celui qui envoie ce segment												
bits indicateurs												
longueur entête + options		Réserve	E	C	U	A	P		R	S	F	taille de la fenêtre
			C	W	R	C	S		S	Y	I	
			N	R	G	K	H		T	N		
contrôle d'erreur sur l'entête						fin des données urgentes placées en début des données utilisateur dans le segment						
options s'il y en a												
données s'il y en a												

20octets

Question 2

On s'intéresse à l'état des connexions sur une des deux machines. On lance la commande netstat -a dont une partie du résultat est affiché ci-dessous. On obtient un ensemble de connexions actives.

Proto	Adresse locale	Adresse distante	État
TCP	163.173.231.107:139	netinfo67:0	LISTENING
TCP	163.173.231.107:2869	wi225:59696	TIME_WAIT
TCP	163.173.231.107:2869	wi225:59699	ESTABLISHED
TCP	163.173.231.107:2869	wi225:59704	ESTABLISHED
TCP	163.173.231.107:2869	bering:32927	TIME_WAIT
TCP	163.173.231.107:49997	wpad:http	CLOSE_WAIT
TCP	163.173.231.107:50074	lmi17:http	ESTABLISHED
TCP	169.254.8.118:139	netinfo67:0	LISTENING

L'automate TCP dispose de 3 types d'états : les états liés à l'ouverture de connexion, les états liés au transfert de données, les états liés à la fermeture de connexion.

Question 2.1

A quels types d'états correspondent les états TIME_WAIT, CLOSE_WAIT, ESTABLISHED, LISTENING, suivants relevés dans les résultats de la commande netstat.

Question 2.2

Dans quel état est la connexion TCP qui achemine le trafic http que nous étudions d'après les résultats de netstat.

Question 2.3

Quel appel système de l'API sockets déclenche l'ouverture de connexion active sur le client.

Question 3

Fragmentation IP ou fragmentation TCP ?

On s'intéresse à la fragmentation de segments TCP. Ligne 8 et 9 on voit apparaître dans la ligne :

8	0.00360700	163.173.228.17	163.173.231.107	TCP	353	[TCP segment of a reassembled PDU]
9	0.00388500	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]

La trace détaillée correspondante pour ces deux lignes donne :

No.	Time	Source	Destination	Protocol	Length	Info
8	0.003607000	163.173.228.17	163.173.231.107	TCP	353	[TCP segment of a reassembled PDU]

Frame 8: 353 bytes on wire (2824 bits), 353 bytes captured (2824 bits) on interface 0

Interface id: 0

Frame Number: 8

Frame Length: 353 bytes (2824 bits)

Capture Length: 353 bytes (2824 bits)

[Frame is marked: True]

[Frame is ignored: False]

[Protocols in frame: eth:ip:tcp]

Ethernet II, Src: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a), Dst: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)

Destination: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)

Address: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)

....0. = LG bit: Globally unique address (factory default)

....0. = IG bit: Individual address (unicast)

Source: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a)

Address: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a)

....0. = LG bit: Globally unique address (factory default)

....0. = IG bit: Individual address (unicast)

Type: IP (0x0800)

Internet Protocol Version 4, Src: 163.173.228.17 (163.173.228.17), Dst: 163.173.231.107 (163.173.231.107)

Version: 4

Header length: 20 bytes



```
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
Total Length: 339
Identification: 0x72eb (29419)
Flags: 0x02 (Don't Fragment)
    0... .... = Reserved bit: Not set
    .1... .... = Don't fragment: Set
    ..0. .... = More fragments: Not set
Fragment offset: 0
Time to live: 64
Protocol: TCP (6)
Header checksum: 0xb3e1 [correct]
    [Good: True]
    [Bad: False]
Source: 163.173.228.17 (163.173.228.17)
Destination: 163.173.231.107 (163.173.231.107)
Transmission Control Protocol, Src Port: http (80), Dst Port: 50074 (50074), Seq: 1, Ack: 491, Len: 299
Source port: http (80)
Destination port: 50074 (50074)
[Stream index: 0]
Sequence number: 1      (relative sequence number)
[Next sequence number: 300      (relative sequence number)]
Acknowledgment number: 491      (relative ack number)
Header length: 20 bytes
Flags: 0x018 (PSH, ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (CWR): Not set
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...1 .... = Acknowledgment: Set
    .... .... 1... = Push: Set
    .... .... .0.. = Reset: Not set
    .... .... ..0. = Syn: Not set
    .... .... ...0 = Fin: Not set
Window size value: 1728
[Calculated window size: 6912]
[Window size scaling factor: 4]
```

Checksum: 0xe4e3 [validation disabled]
 [SEQ/ACK analysis]
 [Bytes in flight: 299]
 TCP segment data (299 bytes)

No.	Time	Source	Destination	Protocol	Length	Info
9	0.003885000	163.173.228.17	163.173.231.107	TCP	1514	[TCP segment of a reassembled PDU]

Frame 9: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface 0

Interface id: 0

Frame Number: 9

Frame Length: 1514 bytes (12112 bits)

Capture Length: 1514 bytes (12112 bits)

[Frame is marked: True]

[Frame is ignored: False]

[Protocols in frame: eth:ip:tcp]

[Coloring Rule Name: HTTP]

[Coloring Rule String: http || tcp.port == 80]

Ethernet II, Src: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a), Dst: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)

Destination: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)

Address: Inventec_9a:4d:aa (00:26:6c:9a:4d:aa)

.... ..0. = LG bit: Globally unique address (factory default)

.... ...0 = IG bit: Individual address (unicast)

Source: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a)

Address: DellEsgP_f9:ad:5a (00:0f:1f:f9:ad:5a)

.... ..0. = LG bit: Globally unique address (factory default)

.... ...0 = IG bit: Individual address (unicast)

Type: IP (0x0800)

Internet Protocol Version 4, Src: 163.173.228.17 (163.173.228.17), Dst: 163.173.231.107 (163.173.231.107)

Version: 4

Header length: 20 bytes

0000 00.. = Differentiated Services Codepoint: Default (0x00)

.... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)

Total Length: 1500

Identification: 0x72ed (29421)

Flags: 0x02 (Don't Fragment)

0... = Reserved bit: Not set

.1.. = Don't fragment: Set




```
..0. .... = More fragments: Not set
Fragment offset: 0
Time to live: 64
Protocol: TCP (6)
Header checksum: 0xaf56 [correct]
  [Good: True]
  [Bad: False]
Source: 163.173.228.17 (163.173.228.17)
Destination: 163.173.231.107 (163.173.231.107)
Transmission Control Protocol, Src Port: http (80), Dst Port: 50074 (50074), Seq: 300, Ack: 491, Len: 1460
Source port: http (80)
Destination port: 50074 (50074)
[Stream index: 0]
Sequence number: 300      (relative sequence number)
[Next sequence number: 1760      (relative sequence number)]
Acknowledgment number: 491      (relative ack number)
Header length: 20 bytes
Flags: 0x010 (ACK)
  000. .... .... = Reserved: Not set
  ...0 .... .... = Nonce: Not set
  .... 0... .... = Congestion Window Reduced (CWR): Not set
  .... .0.. .... = ECN-Echo: Not set
  .... ..0. .... = Urgent: Not set
  .... ...1 .... = Acknowledgment: Set
  .... .... 0... = Push: Not set
  .... .... .0.. = Reset: Not set
  .... .... ..0. = Syn: Not set
  .... .... ...0 = Fin: Not set
Window size value: 1728
[Calculated window size: 6912]
[Window size scaling factor: 4]
Checksum: 0xd169 [validation disabled]
[SEQ/ACK analysis]
  [Bytes in flight: 1759]
[Reassembled PDU in frame: 12]
TCP segment data (1460 bytes)
```

Question 3.1

- Est-ce que le paramètre MSS (Maximum Segment Size) a un impact sur la taille des segments au niveau TCP.
- Quelle est la valeur du MTU (Maximum Transfer Unit) pour les datagrammes IP sur un réseau local Ethernet ?
- Est-ce que le MTU a un impact sur la fragmentation au niveau de la couche IP ?

Question 3.2

Est-ce IP ou TCP ou les deux qui fragmentent dans le cas de l'observation ci-dessus ? Expliquez comment vous comprenez les mécanismes mis en œuvre pour que le transfert de données soit le plus efficace possible ?

Question 3.3

Quelle QoS est associée à chacun des datagrammes 8 et 9 ? Pensez vous que c'est un bon choix, pourquoi ?

Question 4

Une commande WireShark permet de dessiner le flot TCP à partir de la capture faite. L'outil WireShark affiche le graphique ci-après. Mettre en correspondance les différentes primitives de l'API Sockets AFINET, SOCKSTREAM qui correspondent à la mise en œuvre de TCP (ouverture/fermeture de connexion, envoi/réception de données) avec les différentes parties de la trace.

Time	163.173.231.107 163.173.228.17	Comment
0.000627000	(30074) → (80) SYN	Seq = 0
0.001867000	(30074) ← (80) SYN, ACK	Seq = 0 Ack = 1
0.001912000	(30074) → (80) ACK	Seq = 1 Ack = 1
0.002414000	(30074) → (80) PSH, ACK - Len: 490	Seq = 1 Ack = 1
0.003139000	(30074) ← (80) ACK	Seq = 1 Ack = 491
0.003607000	(30074) → (80) PSH, ACK - Len: 299	Seq = 1 Ack = 491
0.003885000	(30074) ← (80) ACK - Len: 1460	Seq = 300 Ack = 491
0.004164000	(30074) → (80) ACK	Seq = 491 Ack = 1760
0.005002000	(30074) → (80) ACK - Len: 1460	Seq = 1760 Ack = 491
0.005030000	(30074) → (80) PSH, ACK - Len: 20	Seq = 3220 Ack = 491
0.005285000	(30074) → (80) ACK	Seq = 491 Ack = 3240
0.035639000	(30074) → (80) PSH, ACK - Len: 505	Seq = 491 Ack = 3240
0.036613000	(30074) → (80) PSH, ACK - Len: 299	Seq = 3240 Ack = 996
0.036797000	(30074) ← (80) ACK - Len: 1460	Seq = 3539 Ack = 996
0.036914000	(30074) → (80) ACK	Seq = 996 Ack = 4999
0.037032000	(30074) ← (80) ACK - Len: 1460	Seq = 4999 Ack = 996
0.037871000	(30074) ← (80) ACK - Len: 1460	Seq = 6459 Ack = 996
0.037915000	(30074) → (80) ACK	Seq = 996 Ack = 7919
0.037924000	(30074) ← (80) ACK - Len: 1460	Seq = 7919 Ack = 996
0.037973000	(30074) → (80) PSH, ACK - Len: 1221	Seq = 9379 Ack = 996
0.038008000	(30074) → (80) ACK	Seq = 996 Ack = 10600
15.037206000	(30074) ← (80) FIN, ACK	Seq = 10600 Ack = 996
15.037283000	(30074) → (80) ACK	Seq = 996 Ack = 10601
15.037527000	(30074) → (80) FIN, ACK	Seq = 996 Ack = 10601
15.037947000	(30074) ← (80) ACK	Seq = 10601 Ack = 997

- Questions avancées - pour travailler votre côté Geek -

Question 5

On s'intéresse à la gestion de la fenêtre de contrôle de flux, à la taille de la fenêtre (Window Size) et à l'option Window Scale. Ils interviennent lors de l'ouverture de connexion de part et d'autre, au moment de l'échange des segments avec l'indicateur SYN.

L'option Window Scale est nécessaire pour que l'échange de données puisse remplir au maximum la connexion sur le réseau sous-jacent (il faut prendre la métaphore d'un tuyau, pipe en anglais). Cette option est utilisée quand le produit débit x délai A/R qui matérialise la capacité de remplissage maximale d'une connexion TCP est supérieur à 64 K octets.

Le réseau local utilisé dans le problème est un réseau Ethernet 1Gb/s (10^9 b/s). La proximité réseau du client et du serveur fait que le délai A/R est estimé à 1,5ms (0.0015 s) environ.

Question 5.1

Calculez le produit débit x délai A/R (BDP, bandwidth-delay product) pour la connexion que nous étudions.

Question 5.2

Comparez la capacité de remplissage théorique au remplissage visé par la connexion TCP entre notre client et notre serveur. D'après vous pourquoi cette connexion ne spécifie pas des paramètres pour mieux remplir le réseau ?

Je n'ai pas les réponses aux questions subsidiaires ci-dessous... plutôt des pistes de réponses dont je ne suis pas absolument certain.

Question subsidiaire 6 :

A partir de la trame 5 la taille de la fenêtre annoncée par le navigateur est augmentée : elle passe à 16425×4 soit 65 700 octets. Elle ne changera plus pendant tout le reste de l'échange. Pourquoi ?

Question subsidiaire 7

Appliquez le même raisonnement dans le sens inverse quand le serveur http est récepteur et le navigateur est émetteur. Il faut prendre la trame 4 dans l'échange donné en question 4, les paramètres window size et window scale sont annoncés dans la partie option du segment TCP.