

M2-CCI Enseignement de Réseaux – Compte-Rendu du TP n.2

Groupe BAJA Sara – BOLLOLI Marco

Introduction :

Ce compte-rendu a pour objet les manipulations effectuées dans le TP n.2, et les observations et les calculs qui y étaient associés. De suite, les étapes et les questions seront présentées avec la numérotation adoptée dans l'énoncé du TP pour des raisons de clarté.

Etude des protocoles de niveau 3 (2.1) :

De façon très similaire à celle du TP1, le réseau est mis en place à partir de 4 ordinateurs et un Hub tel que décrit dans la figure suivante.

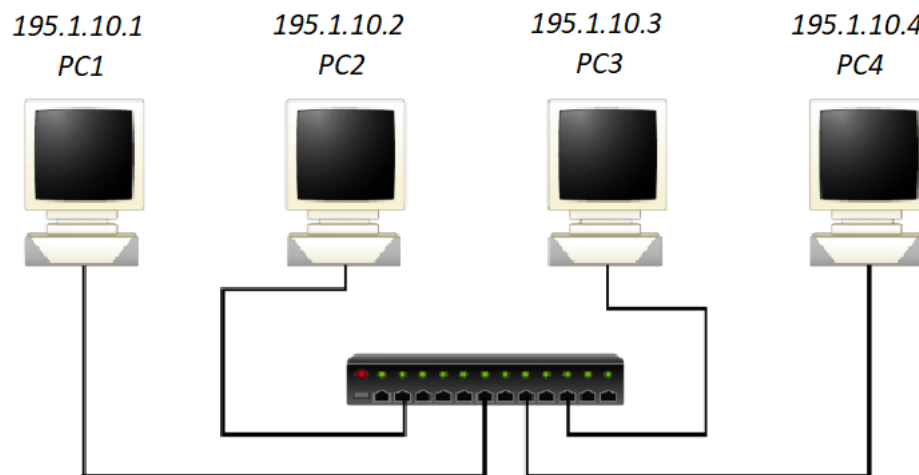


Figure 1

Dans chaque ordinateur, nous n'utilisons pas l'interface réseau de la carte mère (*em0*) mais plutôt l'interface *bge0* qui est installée en plus et qui permet d'observer des collisions entre paquets.

L'interface n'étant pas associée à une adresse IP v4, nous avons procédé à le faire pour chaque ordinateur à l'aide de la commande **ifconfig <nom interface> <adresse>**. De suite le paramètres choisis :

Machine 1 : 195.1.10.1

Machine 2 : 195.1.10.2

Machine 3 : 195.1.10.3

Machine 4 : 195.1.10.4

Nous avons bien vérifié, à l'aide de la même commande ifconfig, que bge0 était correctement configurée sur chaque ordinateur (Cf. Figure 2).

Les informations qui nous intéressent les plus ici sont celles données par le **status** et le **media**.

Le fait que **status** est en mode active et qu'on a 10base/UTP <half-duplex> au niveau de **média** indique que l'interface est en marche.

Sauf où autrement spécifié, le PC3 est la machine utilisée pour suivre les manipulations et enregistrer le passage des paquet parmi les autres 3 machines du réseau.

2.1.1 Le protocole ARP

Après avoir vidé les tables ARP, nous lançons un Ping de PC1 à PC2 et PC2 à PC4.

Nous ouvrons WireShark sur PC3 pour enregistrer les échanges.

Question 1 : Qu'est ce qui permet d'identifier les paquets comme étant de type ARP?

Dans la Figure 1, nous voyons la capture correspondante à un Ping entre PC2 et PC4. Nous voyons que d'abord PC2 envoie un paquet ARP à tout le réseau à travers l'adresse *broadcast* (ff : ff : ff : ff : ff : ff), pour trouver PC4. Le fait qu'il agit bien d'un ARP est indiqué par le EtherType x0806.

Question 2 : Où se situe le paquet ARP dans la trame Ethernet ?

Le paquet se situe après le EtherType, dans la zone Data. Dans la Figure 1 en bas, nous pouvons voir le paquet ARP (sur fond blanc) et le padding (sur fond bleu) juste après.

Question 3 : De la même manière, faites le schéma correspondant au paquet ARP. Indiquez le rôle de chacun des champs du paquet ARP.

Le schéma suivant correspondant au paquet ARP. Le nombre d'octets des champs sont indiqués entre parenthèses. AM: adresse machine, IP: adresse internet.

Type matériel (2)	Type de protocole (2)	longueur adresse matériel (1)	longueur adresse de protocole (1)	type d'opération (2)	Source AM (6)	Source IP (4)	Destination AM (6)	Destination IP (4)
----------------------	--------------------------	----------------------------------	--------------------------------------	-------------------------	------------------	------------------	-----------------------	-----------------------

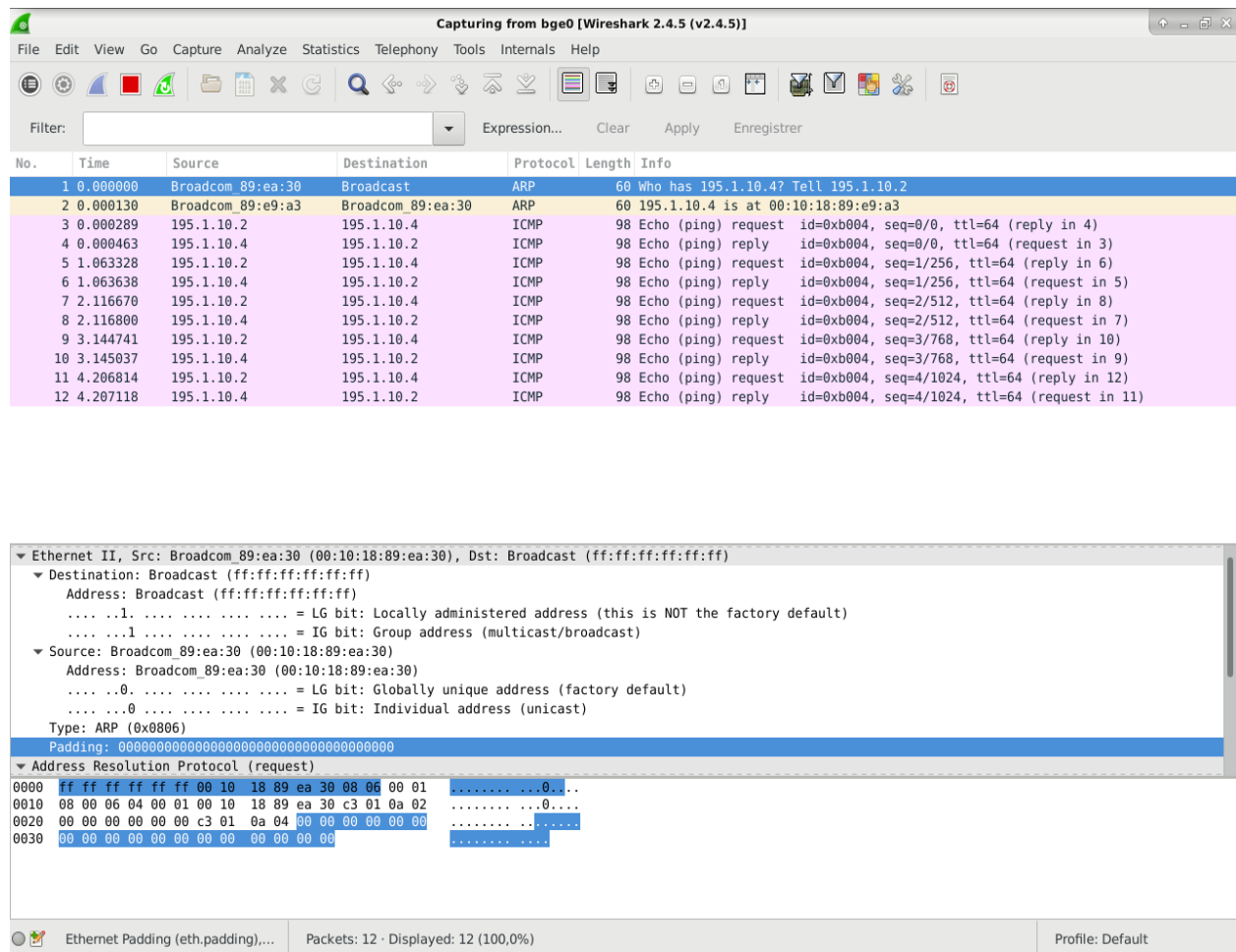


Figure 1 (questions 2 et 4)

Question 4 : Comment le niveau Ethernet connaît-il la taille des trames qu'il reçoit ? Comment détermine-t-il la fin de la trame ? Retrouvez les octets de bourrage d'Ethernet dans la trame.

Le niveau Ethernet ne connaît pas la taille des paquets envoyés. A ce niveau, le paquet est identifié par le début du signal qui lui est associé. Le début du paquet est indiqué par un marqueur particulier et la fin par un silence.

Lorsque la taille du paquet est inférieure à 64 octets, on ajoute des octets de bourrage qui sont constitués par des zéros. Sur la figure 1, on voit bien les octets de bourrage (en bas, sur fond bleu) à la fin du paquet.

Question 5 : Qu'est ce qui est transmis à la couche ARP par la couche Ethernet (à la réception d'un paquet) ? Ethernet connaît-il l'existence d'un bourrage à la réception d'un paquet ?

La couche Ethernet transmet à la couche ARP tout le paquet ARP + le bourrage, donc une réponse qui contient l'adresse de la machine demandée. À la réception d'un paquet, le protocole Ethernet ne connaît pas des octets de bourrage. Ils sont connus par les protocoles des couches supérieures. Après un ping de A vers B. Nous observons des paquets de type ARP. A envoie un paquet de type ARP-request en demandant l'adresse de la machine B. B envoie un paquet de type ARP-reply pour donner son adresse machine à A.

Question 6 : compléter de façon informelle le corps de l'algorithme suivant

Tant que(1) Faire

 Attendre un évènement x

 Si x = Envoi d'un paquet vers une station Alors

 Si son adresse est connue Alors

 Envoyer le message à cette adresse

 Sinon

 Envoyer un paquet ARP Broadcast pour récupérer son adresse

 Fin si

 Fin si

 Si x = Arrivée d'une requête ARP Alors

 Si l'émetteur est dans la table ARP Alors

 Remettre à 0 le compteur de son entrée

 Sinon

 L'ajouter à la table

 Fin si

 Si la requête m'est adressée Alors

 Envoyer une réponse ARP

 Sinon Si je connais le destinataire et que son entrée est published Alors

 Envoyer une réponse ARP avec son adresse MAC

 Fin si

 Fin si

 Si x = Arrivée d'une réponse ARP Alors

 Ajouter l'entrée dans la table ARP

 Fin si

 Si x = Le timer d'une entrée arrive à expiration Alors

 Supprimer l'entrée de la table ARP

 Fin si

Fin Tant que

Question 7 : Observez les tables ARP de A et B. Pourquoi B apparait-il dans la table de A?

Comment B a-t-il pu apprendre l'adresse de A alors qu'il n'a pas envoyé de ARP-request à A?

Lors de l'envoi d'un ARP-request, A insère dans le message son adresse (l'adresse source) : cela permet à la machine réceptrice de connaître la machine à laquelle envoyer le paquet ARP-reply.

Question 8 : Effacez la table ARP de B. Recommencez le ping de A vers B. Notez les paquets engendrés. Expliquez.

A maintenant connaît l'adresse de B, donc il lui envoie directement un paquet ICMP. Par contre, B ne connaît pas de qui ce paquet arrive, donc il est contraint d'envoyer un paquet ARP en broadcast pour l'adresse ethernet de la machine émettrice. (Cf. Figure 2)

Après réception de la réponse (paquet ARP-reply), B peut enfin renvoyer son paquet ICMP.

The image shows a Wireshark 2.4.5 capture on interface bge0. The packet list shows a series of ICMP Echo (ping) requests and replies between 195.1.10.1 and 195.1.10.2. Before the first ping, there is an ARP broadcast from 195.1.10.1 to the broadcast address 89:ea:30. The packet details pane for the selected packet (No. 1) shows the Ethernet II header with source 89:ea:10 and destination 89:ea:30, and the Internet Protocol Version 4 header with source 195.1.10.1 and destination 195.1.10.2. The packet bytes pane shows the raw data of the packet.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=0/0, ttl=64 (reply in 4)
2	0.000120	Broadcast	Broadcast	ARP	60	Who has 195.1.10.1? Tell 195.1.10.2
3	0.000277	Broadcast	Broadcast	ARP	60	195.1.10.1 is at 00:10:18:89:ea:10
4	0.000433	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=0/0, ttl=64 (request in 1)
5	1.044756	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=1/256, ttl=64 (reply in 6)
6	1.044890	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=1/256, ttl=64 (request in 5)
7	2.091393	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=2/512, ttl=64 (reply in 8)
8	2.091528	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=2/512, ttl=64 (request in 7)
9	3.101368	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=3/768, ttl=64 (reply in 10)
10	3.101499	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=3/768, ttl=64 (request in 9)
11	4.164379	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=4/1024, ttl=64 (reply in 12)
12	4.164526	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=4/1024, ttl=64 (request in 11)
13	5.227377	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=5/1280, ttl=64 (reply in 14)
14	5.227684	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=5/1280, ttl=64 (request in 13)
15	6.290408	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=6/1536, ttl=64 (reply in 16)
16	6.290542	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=6/1536, ttl=64 (request in 15)
17	7.353268	195.1.10.1	195.1.10.2	ICMP	98	Echo (ping) request id=0x3204, seq=7/1792, ttl=64 (reply in 18)
18	7.353400	195.1.10.2	195.1.10.1	ICMP	98	Echo (ping) reply id=0x3204, seq=7/1792, ttl=64 (request in 17)

Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
 Ethernet II, Src: Broadcast 89:ea:30 (00:10:18:89:ea:30), Dst: Broadcast 89:ea:30 (00:10:18:89:ea:30)
 Destination: Broadcast 89:ea:30 (00:10:18:89:ea:30)
 Address: Broadcast 89:ea:30 (00:10:18:89:ea:30)
 ...0... = LG bit: Globally unique address (factory default)
 ...0... = IG bit: Individual address (unicast)
 Source: Broadcast 89:ea:10 (00:10:18:89:ea:10)
 Address: Broadcast 89:ea:10 (00:10:18:89:ea:10)
 ...0... = LG bit: Globally unique address (factory default)
 ...0... = IG bit: Individual address (unicast)
 Type: IPv4 (0x0800)
 Internet Protocol Version 4, Src: 195.1.10.1, Dst: 195.1.10.2
 0000 00 10 18 89 ea 30 00 10 18 89 ea 10 00 00 45 00E.
 0010 00 54 8d fa 00 00 40 01 52 a9 c3 01 0a 01 c3 01 .T...@. R.....
 0020 0a 02 08 00 6f 74 32 04 00 00 4e 6b 5a 76 00 0cot2. .NKZv..
 0030 c2 96 08 09 0a 0b 0c 0d 0e 0f 10 11 12 13 14 15 !"#%&
 0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25&'()*+,-./012345
 0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35
 0060 36 37 67

Figure 2 (question 8)

Question 9 : Est ce que c'est le timer de ARP ou de Ping qui déclenche les réémissions des paquets ARP-Request ? Quelle est la durée de ce timer de réémission ?

Comme A ne connaît pas B, mais B est débranché (donc ne peut pas répondre) le ping de A vers B après avoir débranché la machine B déclenche des émissions des paquets ARP-request. Ces

émissions sont déclenchées par le timer du ping non par celui de ARP. Le timer de ARP est de 1200 s (20 minutes), mais comme nous observons un timer plus rapide, celui-ci ne peut correspondre qu'à celui de ping (environ 1s).

Question 10 : Observez le réseau au moment où vous configurer une interface (if-config). À quoi peut servir le ARP gratuit ?

Les Paquets ARP servent à signaler l'adresse IP d'un nouvel ordinateur sur le réseau. L'ARP gratuit request est envoyé à toutes les machines du réseau pour savoir s'il existe une machine qui a la même adresse internet. Si l'adresse n'existe pas encore, on n'a pas d'ARP gratuit reply.

Question 11 : Pourquoi deux paquets de type ARP reply sont-ils observés ? Expliquez ce qu'il s'est passé (demandez une analyse détaillée de ces paquets ; étudiez en particulier les adresses Ethernet et Internet).

Les adresses IP published sont "publiques", donc si une machine connaît une adresse published elle répondra à une ARP-request, même si ce n'est pas son propre adresse Ethernet. Cela car la correspondance à l'adresse Ethernet est gérée à bas niveau, alors que la correspondance avec l'adresse IP est gérée au niveau ARP (donc la machine répond "ne sachant pas" que ce n'est pas son "vrai" IP). Suite à plusieurs ARP-reply avec la même IP, ce sera la dernière réponse émise qui sera retenue dans la table ARP du récepteur.

Donc si PC1 a mémorisé les informations (Ethernet et IP) de PC2, suite à l'envoi d'une ARP-request de la part de PC4, pour demander qui a l'IP 190.1.10.2 (IP de PC2), il y aura 2 paquets ARP-reply qui seront envoyés, de la part de PC1 et PC2. (Cf Figure 11) Les entêtes Ethernet seront différents au niveau des octets de l'adresse source, par contre le paquet ARP contiendra les mêmes informations, donc PC4, dans ce cas, mémorisera la bonne information dans sa table ARP.

Question 12 : Dans quels cas la déclaration d'une adresse en Publish d'une autre machine peut-elle être intéressante ? (avec une autre adresse Ethernet par exemple).

Pour des raisons de sécurité ou administration réseau, une machine pourrait prendre l'adresse IP d'une autre et donc intercepter ou bloquer l'envoi des données vers celle-ci.

Question 13 : Au moment où vous configurez la deuxième machine, vous devriez avoir un message du système à l'écran (si vous sortez de l'environnement "xinit") sur la première. À quoi sert-il ? Qu'est ce qui l'a déclenché?

Nous avons bien visualisé le message. Le message indique que l'adresse IP est utilisée par une autre machine et il est déclenché après l'envoi d'un paquet ARP gratuit après la nouvelle configuration de la machine. Ceci sert à éviter que plusieurs machines aient le même IP sur un même réseau.

2.1.2 Le protocole ICMP

Question 15 : Que remarquez-vous pour le champ Identifier et le champ Sequence Number ? Observez ces deux champs dans les paquets générés par un ping ? A quoi servent-ils ?

Dans notre fichier pour l'envoi ICMP, nous écrivons (en hexadécimale) :

Type: Echo request = 08

Code-type (pour une demande ou une réquête): 00

checksum : 00 00 (pour faciliter le calcul au départ)

identifier ID: 01 01

sequence Number: 02 02

Donc la séquence se présente comme 08 00 00 00 01 01 02 02

Après envoi du fichier de *PC1* à *PC4* à l'aide de la commande **send_icmp**, le fichier reçu par *PC4* présente une structure légèrement différente, car le checksum a été modifié par le protocole CRC pour la détection d'erreur (Cf. Figure 4).

Concernant les champs Identifier et Sequence Number, le premier correspond au numéro de série du ping qui servira à identifier si c'est un request ou reply, tandis que le deuxième correspond au numéro du ping dans la série.

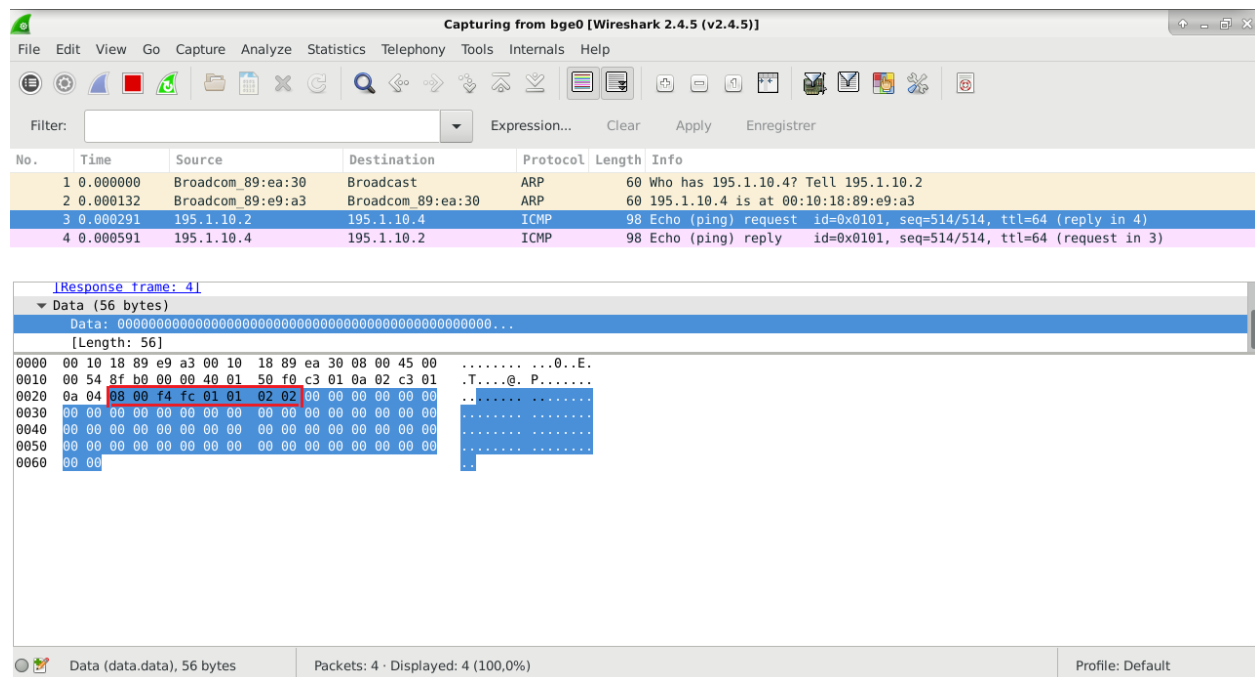


Figure 4 (questions 15 et 16)

Question 16 : Donnez l'algorithme de calcul des deux octets de Checksum. Vérifiez sur un de vos paquets ICMP capturés ce calcul. Quels types d'erreurs ce type de vérification peut-elle détecter et ne pas détecter ?

Un checksum est un algorithme qui permet de donner dans le message une indication d'erreurs de transmission de message. On ajoute les éléments de la trame, deux octets à deux octets, et ensuite nous calculons le complément à 1

Dans notre exemple : 08 00 00 00 01 01 02 02.

```
08 00
01 01
02 02
-----
```

0B 03 → f4 fc checksum qui effectivement apparait dans le paquet ICMP.

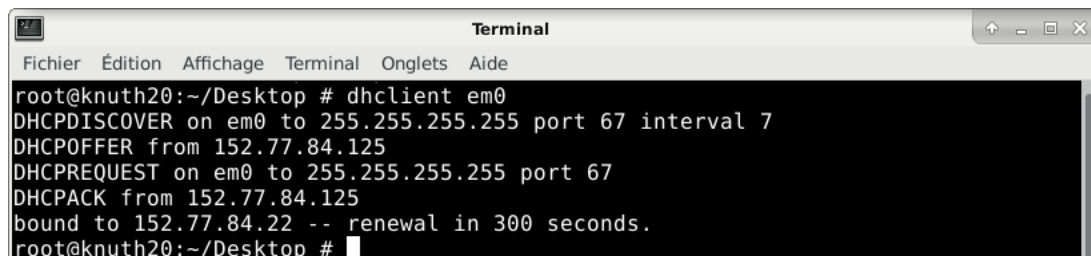
L'avantage de cette méthode, surtout pour un checksum sur des lignes longues (16 ou 32 bits) est la possibilité d'identifier des erreurs en rafales. La limite est que deux erreurs isolés peuvent se compenser, ne pas modifier le checksum, donc passer la detection.

2.2 Etude du protocole DHCP

Question 17 : Observez et commentez les paquets capturés alors. Expliquez en particulier les informations contenues dans le paquet DHCP ACK. Aidez vous du man de dhclient et du cours pour expliquer le contenu de ces paquets. Expliquez la nouvelle configuration de l'interface em0.

On connecte PC2 à la prise murale via l'interface em0. Ensuite on utilise la commande **dhclient em0**. La machine envoie d'abord en broadcast une trame DHCPDISCOVER pour trouver les serveurs disponibles. Un ou plusieurs serveurs répondent avec une trame DHCPOFFER, un proposant une adresse libre dans leur réseau. Le client renvoie un paquet DHCPREQUEST pour confirmer le serveur duquel il accepte l'offre, voire pour demander d'autres paramètres, et le serveur lui envoie un DHCP ACK qui confirme l'adresse IP exacte.

Dans notre cas, l'adresse IP attribué est le 152.77.84.22



```
Terminal
Fichier  Édition  Affichage  Terminal  Onglets  Aide
root@knuth20:~/Desktop # dhclient em0
DHCPDISCOVER on em0 to 255.255.255.255 port 67 interval 7
DHCPOFFER from 152.77.84.125
DHCPREQUEST on em0 to 255.255.255.255 port 67
DHCPACK from 152.77.84.125
bound to 152.77.84.22 -- renewal in 300 seconds.
root@knuth20:~/Desktop #
```

Figure 5 (question 17)

2.3 Étude de la fragmentation des paquets IP

Question 18 : Analysez les paquets engendrés sur le réseau. En particulier, étudiez les entêtes IP et expliquez les rôles des champs IDENTIFICATION, TOTAL LENGTH, FRAGMENT OFFSET, et du flag MF ("More Fragment"). Répéter si nécessaire la manipulation avec d'autres tailles de paquets. Pourquoi l'entête UDP n'apparaît que dans un seul paquet ? Dans quel ordre sont envoyés les fragments ?

Retrouvez dans l'hexadécimal des paquets l'entête UDP. L'entête UDP est-elle dans le premier ou le dernier fragment ? Retrouvez précisément dans ces paquets les 4600 octets de données de l'application. Faites une figure pour expliquer le découpage effectué par IP.

Nous procédons à un envoi à travers le protocole UDP d'un paquet de 4600 octets, ce qui force le protocole IP à le fragmenter en 4 paquets.

L'entête UDP apparaît dans un seul fragment car le découpage est effectué après avoir mis l'entête. Elle est envoyée donc une seule fois, avec le premier fragment (les fragments sont envoyés en ordre de réassemblage, sauf si l'utilisateur le réordonne).

Concernant les différents champs :

Identification : identifie le paquet par tous les paquets, pour le réassembler à la fin ;

More Fragment : MF=1 veut dire qu'il y a encore des paquets à venir, si MF=0 alors c'est le dernier paquet ;

Total length : longueur du fragment ;

Fragment Offset : le numéro du premier octet du fragment par rapport à la taille du paquet original (ex. dans la Figure 6 nous pouvons voir que le paquet original a été fragmenté aux octets 1480 et 2960)

Si nous voulons faire un petit schéma de l'envoi

entete IP	Entete UDP	Data
--------------	---------------	------

Entete UDP	Data
---------------	------

Entete UDP	Data
---------------	------

Entete UDP	Data
---------------	------

