Université de Limoges | FACULTÉ DES SCIENCES ET TECHNIQUES

Master 1ère année

UE Réseaux avancés I

Corrections – nov. 2010

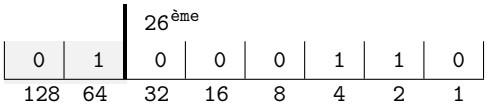
Durée : 1h30 — Documents non autorisés

VLSM & CIDR — 3 points

1 – 2pts Soit le réseau `137.204.212.0/23` :

a. Une allocation des adresses est la suivante :
- LAN_1 doit contenir 126 postes : `137.204.212.0/25` ;
- LAN_2 doit contenir 126 postes : `137.204.212.128/25` ;
- LAN_3 doit contenir 62 postes ; `137.204.213.0/26` ;
- $Res_{R_1\text{-}R_2}$: `137.213.64.0/31` ;
- $Res_{R_2\text{-}R_3}$: `137.213.66.0/31` ;
- $Res_{R_1\text{-}R_3}$: `137.213.68.0/31`.

b. l'adresse réseau `137.204.213.70/26` est incorrecte :
La décomposition binaire du 4ème octet, 70, où se trouve la coupure au 26ème bit :



On remarque que des bits positionnés à 1 sont situés à droite du préfixe `/26`, ce qui est incorrect vis à vis de la définition d'un préfixe réseau.
La solution consiste à utiliser l'adresse réseau : `137.204.213.192.0/26`.

2 – 1pt *On peut simplifier ces adresses destination à l'aide du « supernetting » :*

Destination	next-hop
143.189.208.0/21	R_4

ou comme réponse acceptable, en utilisant la route par défaut vers R4.

Datagramme IP & Analyse de trame — 5 points

3 – 2pts Questions sur la fragmentation :

a. la fragmentation du datagramme IP est nécessaire lorsque le réseau dans lequel doit être transmis le datagramme possède une MTU, « Maximum Transmission Unit », de taille inférieure ;

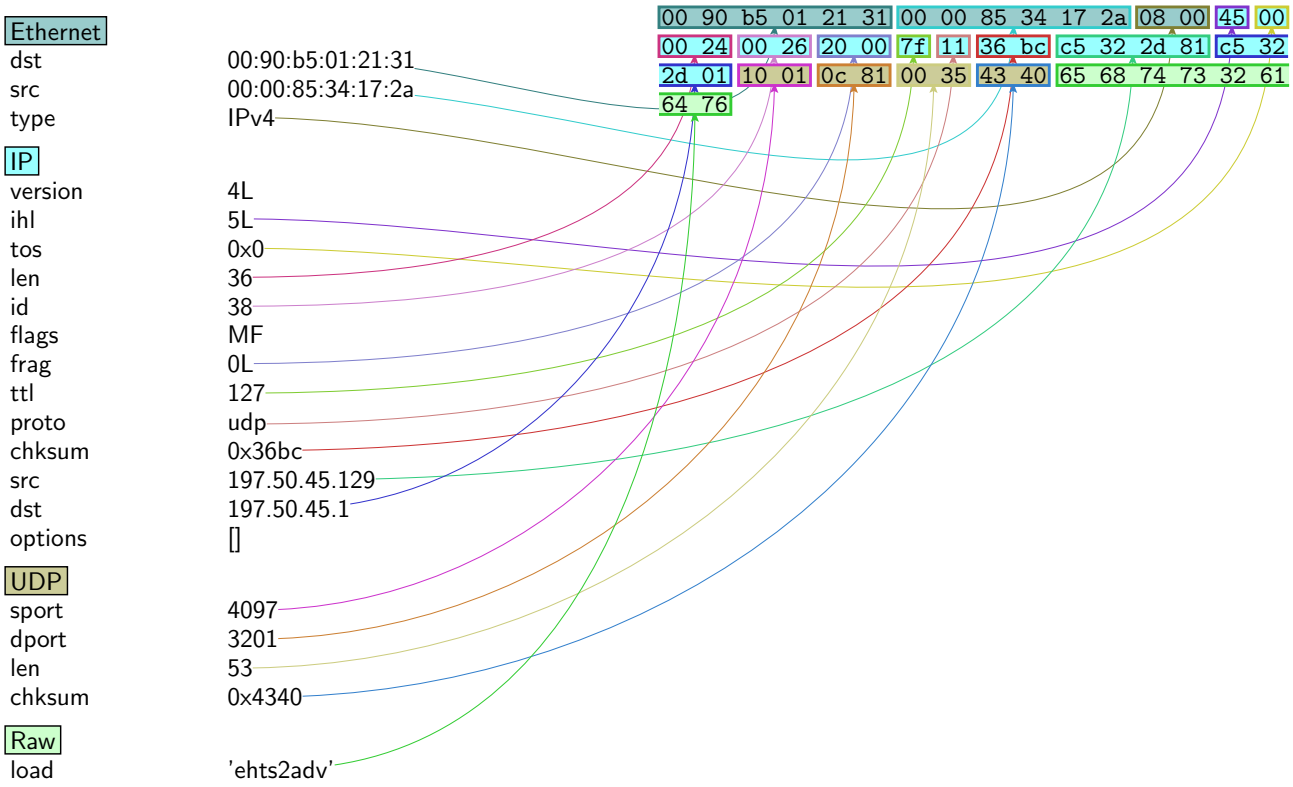
b. les champs du datagramme IP modifiés par la fragmentation sont : le drapeau MF qui va valoir 1 pour tous les fragments et 0 pour le dernier, le champs « offset » et le « checksum » calculé sur l'en-tête.

c. Le dernier fragment est identifié par le drapeau MF à zéro.

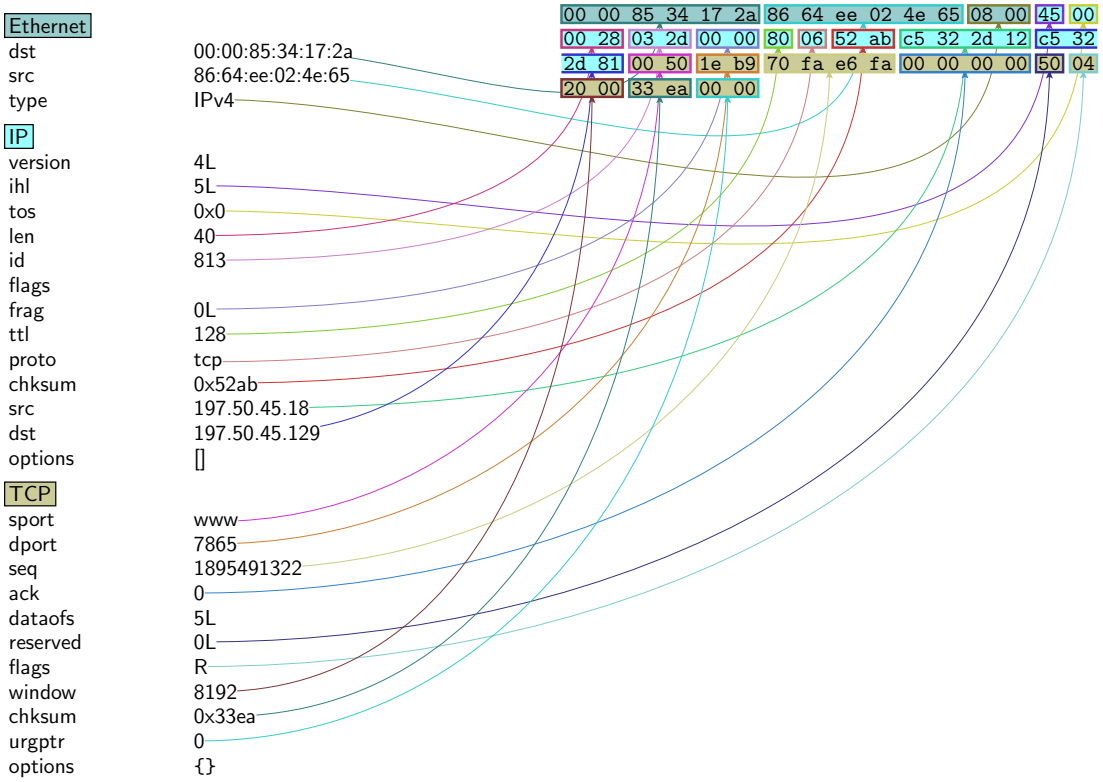
d. Le matériel qui réalise la recombinaison des fragments : la machine destinataire.

4 – Soit la trame n°1 :

3pts



Et la trame n°2 :



Analyse :

– la trame 1 contient le premier fragment (offset à zéro, et MF positionné à 1) d'un datagramme UDP. Le TTL est de 127 ce qui pourrait indiquer que le datagramme provient d'un routeur dont l'@MAC serait `00:00:85:34:17:2A`.
D'autre part, l'@IP source `197.50.45.129` ne ferait pas parti du même réseau que le destinataire d'@IP `197.50.45.1` du fait du passage par un routeur : le réseau destination et source sont différents suivant le préfixe `/25`.

– la trame 2 contient un segment TCP en provenance du port 80. Ce segment contient un drapeau Reset indiquant un échec de communication. L'@MAC de destination de la trame

est `00:00:85:34:17:2A`, ce qui indiquerait un envoi vers le routeur (TTL à 128). Le réseau est encore différencié entre expéditeur et destinataire suivant le préfixe `/25`.

En conclusion, on a appris l'existence d'un routeur et de deux réseaux : `197.50.45.0/25` et `197.50.45.128/25`.

TCP – (7 points)

5 – 2pts Le « SYN-COOKIE » sert à éviter un DoS, « Denial of Service », causé par un épuisement des ressources du serveur pour la gestion des connexions TCP :

- chaque connexion est associée à un TCB, « Transmission Control Block » ;
- le nombre de TCB utilisables est limité ;
- le DoS consiste à envoyer un grand nombre de demandes de connexions (segment TCP avec le drapeau SYN positionné) sans finir le « handshake » d'établissement de la connexion (ainsi, il est possible d'envoyer ces segments d'une adresse IP autorisée dont on aura pris frauduleusement l'identité).
  On parle dans ce cas de « SYN flood ».

On parle de « cookie » comme dans le cas du Web, car le serveur dissimule des données permettant de créer le TCB dans son numéro de séquence TCP lors de sa réponse « SYN-ACK » :

- Si le client poursuit le « handshake » alors il renverra ces données avec son « ACK », et ces données permettront de créer le TCB sur le serveur.
- Dans le cas contraire, aucun TCB n'est créé sur le serveur, et on évite le DoS.

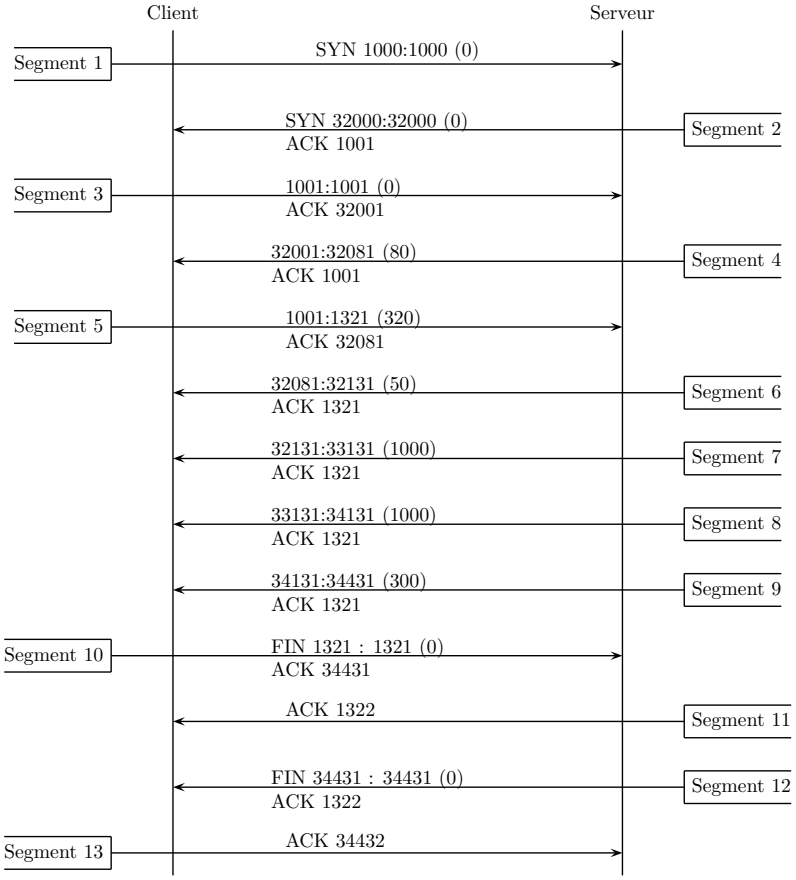
6 – 2pts Questions sur le protocole TCP :

Le contrôle de flux correspond au contrôle de l'émetteur par le récepteur.
Ce contrôle est réalisé par la transmission, du récepteur à l'émetteur, de la taille d'une fenêtre d'autorisation d'envoi : l'émetteur peut envoyer autant d'octets que la taille de la fenêtre lui permet. La taille de la fenêtre est transmise avec chaque segment et peut varier lors de la communication en fonction des besoins et limitations du récepteur.
Le contrôle de congestion permet de tenir compte de l'état du réseau qui est influencé par l'ensemble des machines connectées à ce réseau. Si le réseau entre en congestion, il est nécessaire de limiter la communication afin de ne pas contribuer à cette congestion.
Chaque interlocuteur dispose d'une fenêtre de congestion dont la taille limite en nombre d'octets les données qui peuvent être envoyées dans le réseau.
La taille de la fenêtre de congestion évolue en fonction des réémissions de segments : si un segment se perd, alors on suppose que le réseau entre en congestion et la taille de la fenêtre est diminuée, si au contraire les segments sont transmis sans perte alors le réseau est libre et la taille de la fenêtre est agrandie.

7 – 3pts On remarque que :

- la taille maximale de segment que ce soit pour le client ou pour le serveur est de 1000 octets ;
- la taille de la fenêtre définie par le serveur est de 800 octets :
  - ◇ le client ne peut envoyer au serveur des segments de taille supérieure à 800 octets ;
  - ◇ le serveur peut envoyer au client des segments de 1000 octets car la taille de la fenêtre est supérieure (1200) ;
  - ◇ l'envoi des 2300 octets du serveur vers le client va être décomposé en 3 segments : 1000, 1000 et 300 octets.

msc Échanges TCP



Programmation Socket – (5 points)

8 – 5pts La société Hardwarehouse offre un hébergement de serveurs matériels dans le but de fournir et de mutualiser des salles réfrigérés, des alimentations protégées, des accès physiques protégés, *etc*.

a. Est-ce qu'il est possible de réduire le contenu du message échangé ? Et comment ?
Il est possible d'enlever l'indication de l'@IP de la machine client car on peut la déterminer à l'aide du TSAP du client.
Le message devient alors :

Identifiant service	numéro de port

b. Dans le cas où l'on voudrait pouvoir avoir plusieurs BBServer, comment faire pour que tous les BBClient échangent avec tous les BBServer ?
On peut utiliser le protocole UDP et du multicast : chaque serveur fera partie du même groupe indiqué par la même @IP de classe D.
Chaque client diffusera son information à destination de ce groupe.

c. Indiquez quelles sont les instructions de la programmation Socket et leurs paramètres, qui sont nécessaires à l'écriture du BBServer réalisant la réception d'un message de la part d'un BBClient.
Voir la correction du TP N°2.