

Exercice 1. Questions / discussions de cours

Q1.1 Le serveur DNS d'un site web est en panne. L'administrateur me dit qu'en attendant, je peux me connecter à `http://127.0.0.1/` Donnez deux raisons pour lesquelles cela ne fonctionnera probablement pas.

Q1.2 Pourquoi dit-on que HTTP est acentré alors que DNS est décentralisé, quelle est la différence ?

Q1.3 J'ai configuré le réseau `10.0.1.0/24` sur le vlan 1, et le réseau `10.0.2.0/24` sur le vlan 2, avec un routeur qui a les adresses `10.0.1.1` et `10.0.2.1` et qui route correctement les paquets de l'un à l'autre. La machine ayant l'adresse `10.0.1.2` n'arrive cependant pas à pinguer la machine ayant l'adresse `10.0.2.2`. Je me rends compte que cette dernière a comme adresse configurée `10.0.2.2/16`. Que se passe-t-il exactement pour ce ping qui échoue ? (détaillez tous les paquets émis)

Q1.4 Pourquoi cela n'a pas de sens d'appeler `accept` pour une socket UDP ?

Q1.5 Quel sens cela a-t-il d'appeler `connect` pour une socket UDP ?

Exercice 2. Adresses

Un serveur de VPN fournit à ses clients des IPs au sein d'un réseau `80.67.2.0/23`.

Q2.1 Précisez quelles adresses font partie de cette plage, et combien il y en a.

Q2.2 Le business-plan prévoit que l'on aie prochainement 1500 clients en tout, il faut donc que l'on étende ce réseau. Les adresses voisines de `80.67.2.0/23` sont effectivement disponibles, proposez la nouvelle plage d'adresses à utiliser.

On souhaite également attribuer à chaque VPN un préfixe IPv6 /48.

Q2.3 L'attribution se fera au sein de la plage `2001:910::/32`, proposez une numérotation des préfixes /48.

Q2.4 De combien d'adresses IPv6 chaque VPN disposera-t-il ainsi ?

Exercice 3. Calculs (on pourra faire des arrondis de calculs grossiers si l'on n'a pas de calculatrice). Précisez bien le déroulement de votre calcul.

On a un serveur de fichiers qui fournit à des postes de travail légers une image disque contenant tous les logiciels nécessaires. L'image pèse 3Go, et est récupérée par chaque poste de travail lorsque l'on l'allume. Puisque c'est la même pour tous les postes, elle est en permanence en mémoire du serveur.

Q3.1 Le réseau étant câblé entièrement à 1Gbp/s et 100µs de latence A/R, combien de temps faut-il en théorie pour qu'un poste récupère l'image ?

Q3.2 Pour quelle raison la plus évidente y aurait-il parfois besoin plus de temps ?

En pratique, on se rend compte que cela prend bien plus de temps. On réalise que c'est parce que le protocole utilisé est TFTP, basé sur UDP plutôt que TCP, et qui est très limité : il attend l'acquittement d'un paquet de données avant d'envoyer le paquet de données suivant.

Q3.3 Combien de temps le transfert prend-t-il en pratique ?

Exercice 4. Analyse de paquet

Voici une trame Ethernet capturée par *wireshark*, reçue par la carte réseau de mon ordinateur portable :

```
0000| 44 a3 bb 5b fb 08 ae 92 df f8 d8 8b 08 00 45 00 | D..[.....E.
0010| 00 28 00 00 40 00 40 06 1b e0 0a 00 04 03 0a 00 | .(..@.@.....
0020| 04 07 00 17 04 05 00 00 00 00 fa c0 da ce 50 14 | .....P.
0030| 00 00 f6 76 00 00 00 00 00 00 00 00 00 00 00 | ...v.....
```

On rappelle le format de l'en-tête Ethernet :

48bits destination	48bits source	16bits protocol	data...
--------------------	---------------	-----------------	---------

le format des en-têtes IPv4 et TCP :

0	4	8	16	18	32	0	4	8	16	32	
Ver	hdrl	TOS	length			source		destination			
identification			flags	offset		sequence number					
TTL		protocol	acknowledgment number								
source						offs	res	flags	window size		
destination						checksum		urgent pointer			
data...						data...					

Q4.1 Quelle est l'adresse destination du paquet IPv4 ?

Q4.2 Quelle est la taille totale du paquet IPv4 ?

Q4.3 Quel est le port utilisé du côté de l'émetteur, et du récepteur ?

Q4.4 Quel côté est serveur, pourquoi ?

Exercice 5. HTTP/1.0 ; HTTP/1.1 ; QUIC (HTTP/3)

En HTTP/1.0, un client ne peut envoyer qu'une seule requête GET : le serveur ferme la connexion à la fin du fichier retourné. HTTP/1.1 permet au client d'envoyer plusieurs requêtes GET l'une après l'autre avec la même connexion TCP. Il peut même en envoyer plusieurs à l'avance, sans même attendre que le serveur envoie des données (Pipelining HTTP)

Q5.1 Faites un chronogramme de ce qui se passe quand un client récupère ainsi deux fichiers en HTTP/1.1.

Q5.2 Qu'est-ce que l'on y gagne exactement dans le cas où le client récupère de nombreux petits fichiers ? Pourquoi on y gagne encore bien plus dans le cas où l'on utilise HTTPS ?

Q5.3 Qu'est-ce que le serveur doit faire pour que le client puisse récupérer correctement les différents fichiers ainsi envoyés par le serveur ?

J'ai un site web dont le fichier html commence par indiquer d'utiliser une grosse image en fond de page qui n'apporte pas d'information utile, puis contient du texte avec des petites images qui apportent de l'information utile.

Q5.4 Pourquoi HTTP/1.1 pose problème dans ce cas-ci par rapport à utiliser plusieurs connexions HTTP/1.0 en parallèle ?

Le protocole QUIC est une réimplémentation de TCP, par-dessus UDP, et est utilisé par HTTP/3 pour résoudre ces problèmes ensemble : avec une seule connexion QUIC on peut lancer plusieurs flux en parallèle.

Q5.5 Pourquoi QUIC est-il implémenté au-dessus d'UDP, plutôt que sans UDP à côté de TCP et UDP ?