

Exercices dirigés

UTC505/USRS4D

-IP-

E. Gressier-Soudan

2021-2022

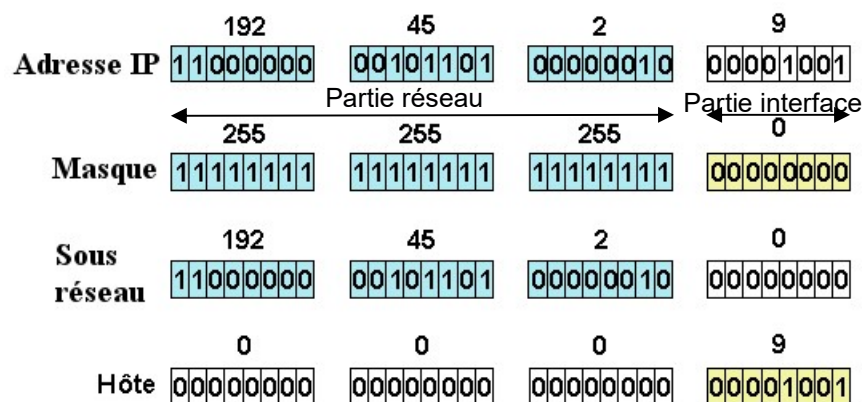
Ce polycopié a été élaboré par l'équipe enseignante "Réseaux et protocoles" à partir d'exercices rédigés par MM. Florin, Gressier-Soudan qu'ils en soient ici remerciés.

ED•Adressage IP & Forwarding Adresses, réseau et sous-réseau, tables de routage

L'objectif de ce chapitre d'exercices est de faire manipuler les adresses IP, les masques, en binaire, en hexadécimal, et en décimal pour pouvoir passer de l'un à l'autre aisément.

Son deuxième objectif est de comprendre le processus de traversée d'un routeur.

Les concepts à comprendre sont l'adresse IP (ici IPv4 mais c'est extensible à IPv6) et le masque. Le masque délimite la partie adresse de réseau¹ de la partie numéro d'interface. C'est très clair quand on manipule les bits d'une adresse et les bits d'un masque. Le tout est résumé dans la figure ci-dessous avec un masque de longueur 24 bits :



source https://sti2d.ecolelamache.org/v_le_masque_de_sous_rseau.html (21/05/2021)

L'adresse IP désigne un coupleur de communication au niveau de la couche de communication réseau (3 dans les modèles OSI et Internet) IP, et l'adresse de réseau désigne un ensemble d'interfaces. L'adresse IP d'interface est utilisée pour désigner une destination à atteindre particulière. L'adresse de réseau est utilisée dans les routeurs pour qu'un datagramme IP envoyé vers une destination puisse atteindre cette destination.

¹ Ici réseau est à prendre au sens large. La partie réseau peut parfois être découpée ou redécoupée en réseau puis sous-réseau(x).

L'opération "masque" sur les adresses correspond à un ET logique appliqué donc bit par bit suivant la longueur du masque sur une adresse de réseaux IPv4 ou IPv6.

Rappels sur l'univers binaires :

Puissance de 2	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Valeur décimale	128	64	32	16	8	4	2	1
Somme des colonnes de gauche à droite	128	192	224	240	248	252	254	255

Table des valeurs des groupements de chiffres binaires

Binaire	Décimal	Octal	Hexadécimal	Binaire	Décimal	Octal	Hexadécimal
0000	0	0	0	1000	8	10	8
0001	1	1	1	1001	9	11	9
0010	2	2	2	1010	10	12	A
0011	3	3	3	1011	11	13	B
0100	4	4	4	1100	12	14	C
0101	5	5	5	1101	13	15	D
0110	6	6	6	1110	14	16	E
0111	7	7	7	1111	15	17	F

source : https://fr.wikipedia.org/wiki/Syst%C3%A8me_binaire (24/04/2021)

Here is the bitwise equivalent operations of two bits P and Q:

p	q	F^0	NOR^1	Xq^2	$\neg p^3$	\rightarrow^4	$\neg q^5$	XOR^6	$NAND^7$	AND^8	$XNOR^9$	q^{10}	If/then ¹¹	p^{12}	Then/if ¹³	OR^{14}	T^{15}
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Bitwise equivalents		0	NOT (p OR q)	(NOT p) AND q	NOT p	p AND (NOT q)	NOT q	p XOR q	NOT (p AND q)	p AND q	NOT (p XOR q)	q	(NOT p) OR q	p	p OR (NOT q)	p OR q	1

source : https://en.wikipedia.org/wiki/Bitwise_operation (24/04/2021)

Exemple :

```

00001010
ET (logique)
11111111
=
00001010

```

Plus si cela vous intéresse [ici](#) (origine Masque de réseau et sous-réseaux par le LYCEE CFA-CFC JEANNE D'ARC, consulté le 15/03/2020).

Cours et exercices sur les conversions en binaire, hexadécimal et décimal (consultés le 13/10/2019) :

<https://www.apprendre-en-ligne.net/info/codage/codage.pdf>

http://www.scientillula.net/MPI/fex6_conversions/fex6_conversions.html

<https://lipn.univ-paris13.fr/~manzonetto/~M1101/M1101-td-01-correction.pdf>

Mise en perspective du Masque, de l'adresse IPv4 (de longueur 32 bits), et de l'adresse de réseau:

Soit une adresse IPv4 : 10.168.0.0, en fonction de la valeur du masque, essayons de déterminer sa nature.

1. Hypothèse : le masque du réseau auquel 10.168.0.0 appartient vaut /18

Effectuons "10.168.0.0" ETlogique "/18".

Le masque /18 est la notation compacte de 11111111.11111111.11 000000.00000000 et exprimé en décimal c'est : 255.255.192.0.

"10.168.0.0" ETlogique "/18" = 10.168.0.0 ETlogique 11111111.11111111.11 000000.00000000

Un octet tout à 1 appliqué à un nombre sur un octet, ne change rien au nombre (ET équivalent à la multiplication et 1 est neutre pour la multiplication).

Un octet tout à 0 appliqué à un nombre sur un octet, le transforme en 0 (ET équivalent à la multiplication et 0 est absorbant pour la multiplication)

Donc on peut calculer partiellement rapidement : "10.168.0.0" ETlogique "/18" = 10.168.qqch.0 ! Il reste à trouver ce qqch. En fait c'est simple car dans l'adresse l'octet correspondant au 3^{ème} octet du masque vaut 0. On a donc ce qqch qui vaut 0.

D'où "10.168.0.0" ETlogique "/18" = 10.168.0.0

Par la suite, on notera "10.168.0.0" ETlogique "/18" par 10.168.0.0/18

Définition : Une adresse IP est une adresse de réseau (et pas d'interface ou d'host) si la partie qui correspond au numéro d'interface est à 0.

Des calculs ci-dessus, peut-on déduire que 10.168.0.0 est une adresse de réseau ou de host ?

Comme 10.168.0.0/18 = 10.168.0.0, on en conclut que 10.168.0.0 est une adresse de réseau. En effet :

10.168.0.0/18 => 10.10101000.00000000.0/18 => 10.10101000.00 000000.0 = 10.168.0.0

2. Hypothèse : le masque auquel 10.168.0.0 appartient vaut maintenant /10, est-ce que 10.168.0.0 est une adresse de réseau ou de host ?

10.168.0.0/10 => 10.10101000.00000000.0/10 => 10.10 000000.00000000.0 = 10.128.0.0 est différent de 10.168.0.0 donc 10.168.0.0 n'est pas une adresse de réseau quand le masque est /10

Le concept à maîtriser ensuite correspond à l'adresse de diffusion associée à un réseau, adresse IPv4 bien sûr dans notre contexte. Là encore, il faut se servir du masque du réseau. On détecte la partie réservée à numéroté les interfaces dans l'adresse IP d'un réseau, grâce au masque, puis on met que des 1 dans cette zone.

3. On cherche l'adresse de diffusion limitée au réseau, ou encore appelée broadcast, de 10.128.0.0/10 ?

10.128.0.0 s'écrit 10.10 000000.00000000.00000000 sachant que le masque est 11111111.11 000000.00000000.00000000

adresse de réseau : 10 .10 000000.00000000.00000000

masque : 11111111.11 000000.00000000.00000000

broadcast : 10 .10 111111.11111111.11111111

broadcast décimal : 10 . 191 . 255 . 255 (128+63=191)



Exercice 1 : Calcul d'adresses IPv4.

Question 1 :

Sur Internet, on trouve des outils pour calculer des adresses, l'un de ceux-ci donne pour résultat si on lui fournit l'adresse d'interface 10.168.0.1 et le masque /20:

```
Address: 10.168.0.1          00001010.10100100.0000 0000.00000001
Netmask: 255.255.240.0 = 20   11111111.11111111.1101 0000.00000000
=>
Network: 10.168.0.0/20       00001010.10101000.0000 0000.00000000
Broadcast: 10.168.15.255     00001010.10101000.0000 1111.11111111
HostMin: 10.168.0.1          00001010.10101000.0000 0000.00000001
HostMax: 10.168.15.254       00001010.10101000.0000 1111.11111110
Hosts/Net: 4094               (Private Internet)
```

Est-ce que le résultat est correct ou contient-il une ou plusieurs erreurs ? Justifiez votre réponse.

Correction :

On peut résoudre ce type de question avec un outil en ligne comme <http://jodies.de/ipcalc>

- L'adresse IP 10.168.0.1 en binaire n'est pas correcte :

Le premier octet en binaire correspond bien à la valeur de 10 en binaire.

Le deuxième octet 168 n'est pas bien représenté en binaire : $168 = 128 + 32 + 8$, en binaire cela donne 1010 1000

Les 3^{ème} et 4^{ème} octets de l'adresse IP sont corrects.

L'adresse IP de l'interface en binaire devrait être

00001010.10101000.0000 0000.00000001

- Il n'est pas possible d'avoir dans la partie gauche d'un masque un 0 intercalé au milieu d'une suite de 1.

11111111.11111111.1101 0000.00000000

devrait être

11111111.11111111.1111 0000.00000000 on a bien maintenant une suite de 20 "1" consécutifs

L'adresse IP du réseau est bien 10.168.0.0/20

<http://jodies.de/ipcalc> avec 10.168.0.0 avec un masque de taille 20 donne le résultat suivant :

```
Address: 10.168.0.1          00001010.10101000.0000 0000.00000001
Netmask: 255.255.240.0 = 20   11111111.11111111.1111 0000.00000000
Wildcard: 0.0.15.255         00000000.00000000.0000 1111.11111111
=>
Network: 10.168.0.0/20       00001010.10101000.0000 0000.00000000
Broadcast: 10.168.15.255     00001010.10101000.0000 1111.11111111
HostMin: 10.168.0.1          00001010.10101000.0000 0000.00000001
HostMax: 10.168.15.254       00001010.10101000.0000 1111.11111110
Hosts/Net: 4094               (Private Internet)
```

Qui confirme la réponse donnée au dessus.

Question 2 :

L'outil peut aussi proposer un calcul si on veut découper son réseau en sous réseaux. Si on indique un masque /21 voila les propositions qu'il fait pour 2 sous-réseaux :

```

Netmask: 255.255.248.0 = 21 11111111.11111111.11011 000.00000000

Network: 10.168.0.0/21 00001010.10101000.00000 000.00000000
Broadcast: 10.168.7.255 00001010.10101000.00000 111.11111111
HostMin: 10.168.0.1 00001010.10101000.00000 000.00000001
HostMax: 10.168.7.254 00001010.10101000.00000 111.11111110
Hosts/Net: 2046 (Private Internet)

Network: 10.168.8.0/21 00001010.10101000.00001 000.00000000
Broadcast: 10.168.15.255 00001010.10101000.00001 111.11111111
HostMin: 10.168.8.1 00001010.10101000.00001 000.00000001
HostMax: 10.168.15.254 00001010.10101000.00001 111.11111110
Hosts/Net: 2046 (Private Internet)

Subnets: 2, Hosts: 4092

```

Est-ce que le résultat est correct ou contient-il une ou plusieurs erreurs ? Pourquoi lorsqu'on passe à deux sous-réseaux on perd deux hosts ? Justifiez votre réponse.

Correction :

Là encore le masque /21 n'est pas correct dans son format binaire. Il devrait être :

11111111.11111111.11111 000.00000000

On perd deux adresses dans chaque sous-réseau. C'est normal. On perd l'adresse du réseau lui-même qui se termine par une partie interface tout en 0 et le broadcast qui se termine par une partie interface tout en 1, et pour chaque réseau créé. On n'en perd que 2 car l'adresse de diffusion du second réseau est celle du réseau avant décomposition, et l'adresse du premier sous-réseau correspond à l'adresse réseau du réseau initial.

<http://jodies.de/ipcalc> donne :

```

Address: 10.168.0.1 00001010.10101000.00000 0000.00000001
Netmask: 255.255.240.0 = 20 11111111.11111111.1111 0000.00000000
Wildcard: 0.0.15.255 00000000.00000000.0000 1111.11111111
=>
Network: 10.168.0.0/20 00001010.10101000.00000 0000.00000000
Broadcast: 10.168.15.255 00001010.10101000.00000 1111.11111111
HostMin: 10.168.0.1 00001010.10101000.00000 0000.00000001
HostMax: 10.168.15.254 00001010.10101000.00000 1111.11111110
Hosts/Net: 4094 (Private Internet)

```

Subnets

```

Netmask: 255.255.248.0 = 21 11111111.11111111.11111 000.00000000
Wildcard: 0.0.7.255 00000000.00000000.00000 111.11111111

Network: 10.168.0.0/21 00001010.10101000.00000 000.00000000
Broadcast: 10.168.7.255 00001010.10101000.00000 111.11111111
HostMin: 10.168.0.1 00001010.10101000.00000 000.00000001
HostMax: 10.168.7.254 00001010.10101000.00000 111.11111110
Hosts/Net: 2046 (Private Internet)

Network: 10.168.8.0/21 00001010.10101000.00001 000.00000000
Broadcast: 10.168.15.255 00001010.10101000.00001 111.11111111
HostMin: 10.168.8.1 00001010.10101000.00001 000.00000001
HostMax: 10.168.15.254 00001010.10101000.00001 111.11111110
Hosts/Net: 2046 (Private Internet)

Subnets: 2
Hosts: 4092

```



Question 3 :

L'outil peut aussi fournir des réponses quand on propose de rechercher une adresse englobante, "supernet". Si on propose un masque /18 on obtient la réponse suivante :

```

Netmask: 255.255.192.0 = 18      11111111.11111111.11 000000.00000000
Network: 10.168.0.0/18           00001010.10101000.00 000000.00000000
Broadcast: 10.168.63.255         00001010.10101000.00 111111.11111111
HostMin: 10.168.0.1              00001010.10101000.00 000000.00000001
HostMax: 10.168.63.254           00001010.10101000.00 111111.11111110
Hosts/Net: 16382                  (Private Internet)

```

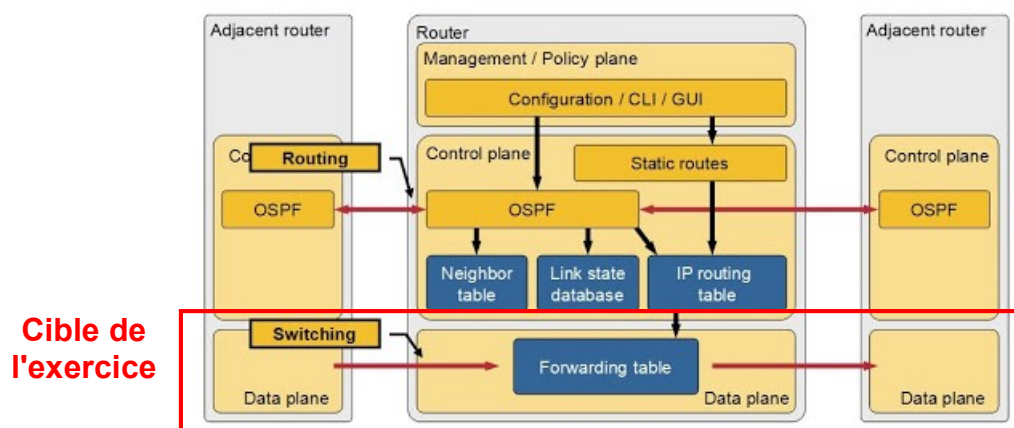
Pourquoi peut-on disposer de 16382 hosts maintenant ? Justifiez votre réponse.

Correction :

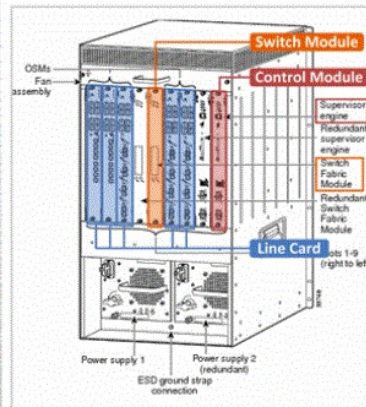
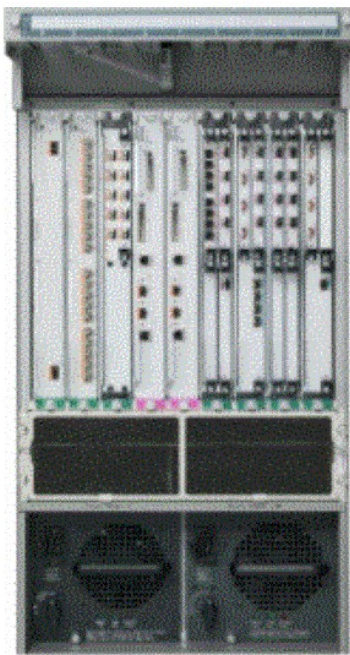
Si le masque est /18, il reste 14 bits pour affecter des numéros d'interface et fabriquer des adresses IP. 2^{14} vaut $2^4 * 1024$ soit $16 * 1024 = 16\,384$ valeurs, comme on doit retirer la valeur 0 et la valeur correspondant à tous les bits à 1, il ne reste que $16\,384 - 2 = 16\,382$ adresses d'interfaces (hosts) possibles.

Exercice 2 : Principe du "forwarding" dans un routeur IP

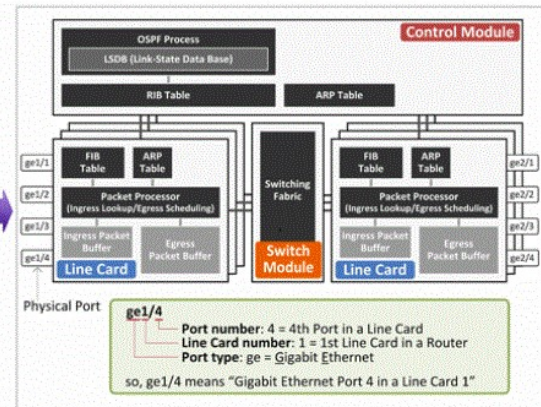
Pour introduire l'exercice, il faut s'imaginer l'organisation d'un routeur. La figure ci-après aide à se représenter les fonctions mise en œuvre dans cet équipement.

Management, Control and Data Planes

source : <https://blog.ipospace.net/2013/08/management-control-and-data-planes-in.html> (consultée le 29/10/2020)



Cisco 7600 Router

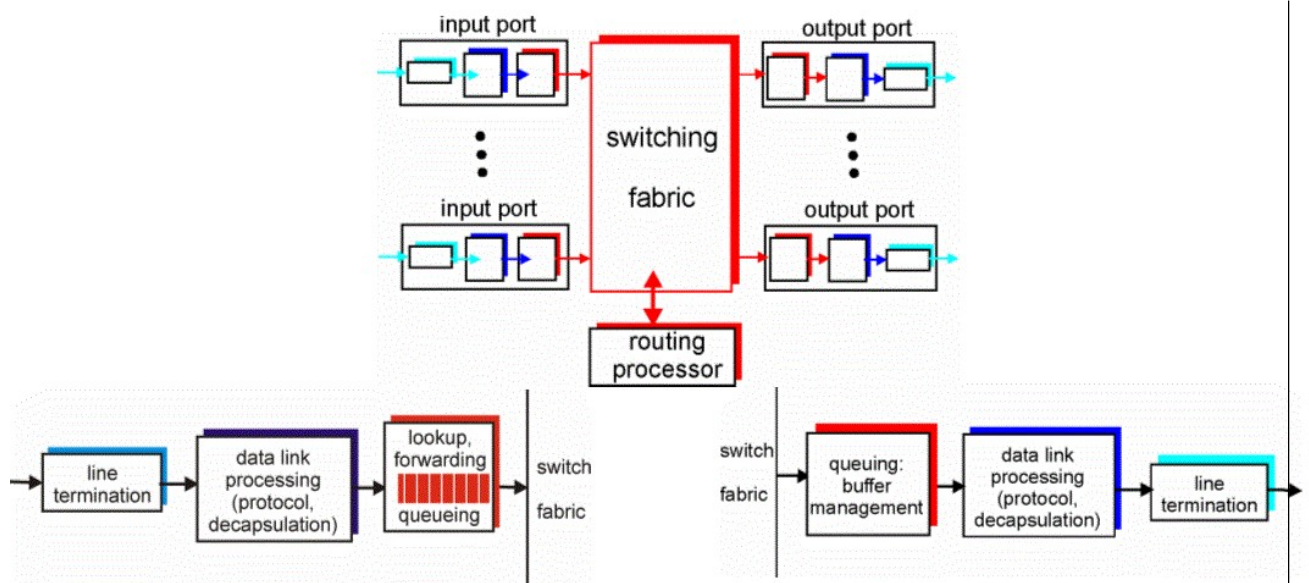


General Router Architecture

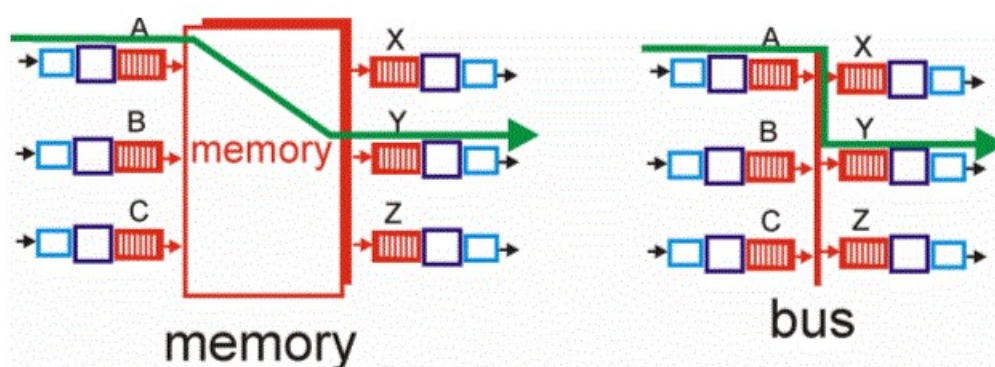
Source : <https://www.netmanias.com/en/post/blog/6338/ip-routing-network-protocol-switching/switching-and-routing-part-1-router-architecture> (29/04/2021)

Source : https://www.cisco.com/c/fr_ca/support/routers/7600-series-routers/series.html (29/04/2021)

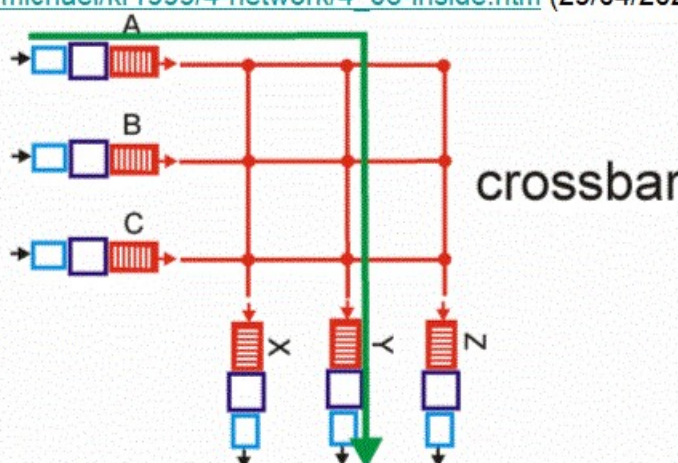
Mise en correspondance des vues logique et physique d'un routeur et de son organisation interne



Représentation schématique d'un routeur et de ses "pattes"



Source : http://www2.ic.uff.br/~michael/kr1999/4-network/4_06-inside.htm (29/04/2021)



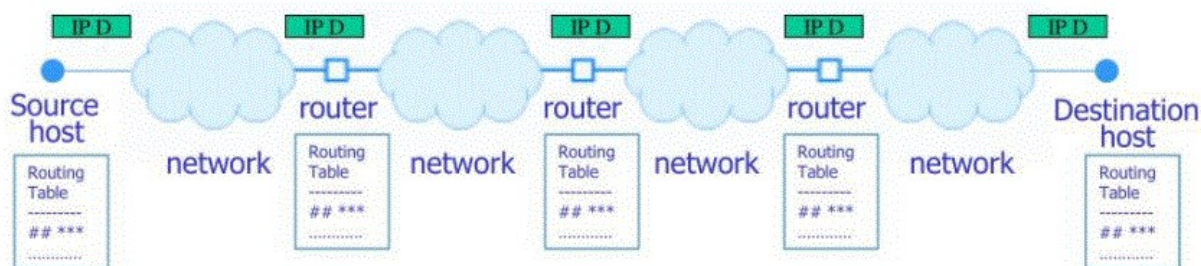
Différentes mises en œuvre de la partie commutation

Complément :

La nouvelle vision de l'architecture introduit 3 plans : Management Plane, Control Plane et Data Plane. Le management plane s'occupe essentiellement de configuration et des informations liées à l'administration à distance, c'est le dernier plan qui s'est formalisé. Le control plane est plus ancien et c'est là qu'on trouve toute l'intelligence du routeur : calcul des routes, génération des tables de routages, tables de gestion de la QoS, tables de gestion des groupes multicast... Le plan forwarding est aussi appelé plan de données ou Data Plane existe depuis toujours.

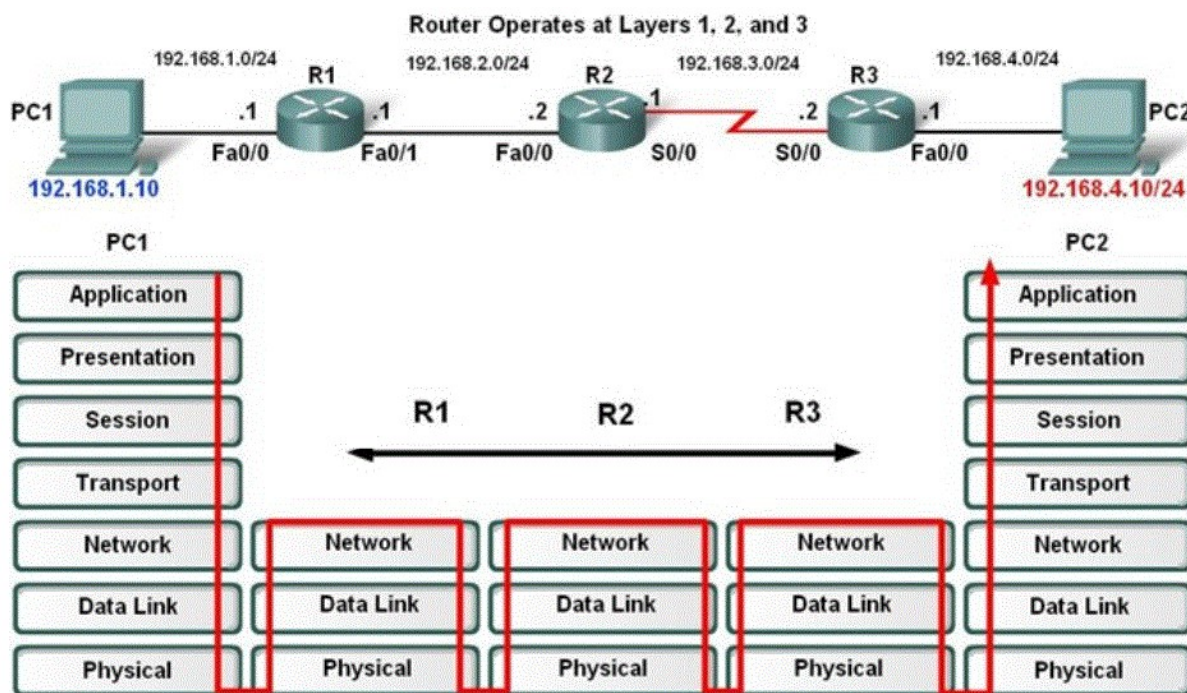
Cette nouvelle terminologie correspond à l'émergence de nouvelles technologies en relation avec l'approche SDN, Software Defined Network qui offre une vision logicielle des technologies réseaux et qui correspond à une démarche de virtualisation des équipements réseaux à l'instar de la virtualisation de l'exécution avec les machines virtuelles (VM pour Virtual Machine en anglais).

L'équipement routeur a pour rôle de diriger un datagramme (l'unité de message gérée par la couche IP) d'une voie d'entrée vers une voie de sortie qui le rapproche de sa destination. Au niveau macroscopique cela donne :



source : <https://line.17qq.com/articles/snqnpnwvy.html> (22/04/2021)

Le schéma ci-après représente le chemin de traversée des couches en relation avec le modèle ISO d'une succession de routeurs :



source : <https://slideplayer.fr/slide/4259942/> (22/04/2021)

Et parcourir le chemin d'un point à un autre de l'Internet en traversant une succession de routeur se fait à l'aide des tables de routage.

Dans un routeur IP la table de routage comporte pour chaque entrée (ligne) une "route". On trouve dans l'exemple de la table suivante:

- un numéro d'entrée pour se repérer facilement,
- l'adresse IP d'un réseau de destination,
- le masque associé à cette adresse réseau de destination (en notation /n),
- l'adresse IP du prochain routeur ou du prochain hôte à visiter sur la route,
- le coût (la métrique),
- l'interface de sortie coupleur/carte réseau qui va donner la liaison à emprunter.

D'autres informations sont prévues dans les tables de routage IP mais n'apparaissent pas dans l'exemple de l'exercice, comme par exemple l'adresse du port à utiliser en sortie ou le type de la route pour définir si le datagramme IP doit atteindre une destination distante ou s'il est à délivrer à sa destination directement par le présent routeur. Ici, toutes

les destinations à atteindre sont distantes.

N°	Réseau Destination	Masque du réseau Dest	Prochain Routeur	Métrique	Interface associée
1	0.0.0.0	/0	10.1.3.65	1	eth1
2	10.1.0.0	/16	10.1.3.65	1	eth1
3	10.1.3.0	/24	10.1.3.1	0	eth1
4	10.1.3.64	/26	10.1.3.126	0	eth0
5	10.1.3.128	/26	10.1.3.190	0	eth0
6	10.1.3.192	/26	10.1.3.254	0	eth0
7	10.1.4.0	/24	10.1.3.4	11	eth1
8	10.1.4.64	/26	10.1.3.4	9	eth1
9	10.1.4.128	/26	10.1.3.4	10	eth1
10	10.1.16.64	/26	10.1.3.65	5	eth1
11	10.1.8.0	/24	10.1.3.65	6	eth1
12	10.1.8.0	/26	10.1.3.65	9	wlan0
13	10.1.8.64	/26	10.1.3.65	17	ppp2
14	10.1.8.64	/26	10.1.3.62	22	ppp0
15	10.1.8.128	/26	10.1.3.65	25	ppp1

Question 1

Une fois extraite l'adresse IP de destination du datagramme, on va chercher quel est le réseau de destination le plus approprié connu par le routeur. Chaque réseau de destination potentiel correspond à une ligne de la table de routage. Chaque ligne pour identifier un réseau de destination contient l'adresse de réseau et le masque associé.

Pour établir une correspondance entre le réseau mentionné sur une ligne de la table de routage et l'adresse IP de destination contenue dans le datagramme, on applique le masque à l'adresse de destination.

- Si le résultat de l'application du masque à l'adresse IP de destination correspond à l'adresse du réseau dans la ligne de la table de routage, on garde la ligne. On dit que la ligne est candidate à la propagation du datagramme. On dit aussi ligne candidate à l'acheminement du datagramme vers le prochain routeur
- Si le résultat ne correspond pas, la ligne est tout simplement éliminée.

On fait ce calcul de masque pour toutes les lignes dans la table de routage. A la fin de cette phase il reste, normalement une ou plusieurs lignes candidates. S'il n'y en a pas, on se trouve en présence d'une erreur de routage, le datagramme est éliminé et le routeur prévient la source (via l'@IP source contenue dans le datagramme) comme quoi la destination n'est pas atteignable (HOST UNREACHABLE).

On comprend la nécessité d'avoir une table de routage la moins longue possible car le temps pour déterminer une interface de sortie est proportionnel à son nombre d'entrées. Les routeurs peuvent bénéficier d'optimisations hardware qui permettent de traiter plusieurs entrées de la table de routage en même temps, par exemple avec



l'utilisation de mémoires associatives.

Application à la destination **10.1.8.66**. Quelles sont les routes qui passent positivement ce test ?

Correction :

On exécute l'algorithme ci-dessus pour déterminer les lignes de sortie candidates, on l'effectue sur les 15 lignes de la table de routage :

- **Ligne 1** : 0.0.0.0/0, on effectue $10.1.8.66/0 = 0.0.0.0$, c'est la même adresse que l'adresse de réseau de cette ligne, c'est une ligne candidate.
- **Ligne 2** : 10.1.0.0/16, on effectue $10.1.8.66/16 = 10.1.0.0$, c'est la même adresse que l'adresse de réseau de cette ligne, c'est une ligne candidate.
- **Ligne 3** : 10.1.3.0/26 on effectue $10.1.8.66/26 = 10.1.8.64$, ce n'est pas la même adresse que l'adresse de réseau de cette ligne, cette ligne n'est pas retenue
- **Lignes 4,5,6** : 10.1.3.64/26, 10.1.3.128/26, 10.1.3.192/26, on a $10.1.8.66/26 = 10.1.8.64$, cela ne correspond pas donc ces trois lignes ne sont pas candidates.
- **Lignes 7,8,9,10** : 10.1.4.0/26, 10.1.4.64/26, 10.1.4.128/26, 10.1.16.64/26, on a $10.1.8.66/26 = 10.1.8.64$, cela ne correspond pas donc ces quatre lignes ne sont pas candidates.
- **Ligne 11** : 10.1.8.0/24, on a $10.1.8.66/24 = 10.1.8.0$, on a égalité, cette ligne est candidate
- **Ligne 12** : 10.1.8.0/26, on a $10.1.8.66/26 = 10.1.8.64$, on n'a pas correspondance, la ligne n'est pas candidate
- **Lignes 13 et 14** : 10.1.8.64/26, on a $10.1.8.66/26 = 10.1.8.64$, on a correspondance, ces deux lignes sont candidates
- **Ligne 15** : 10.1.8.128/26, on a $10.1.8.66/26 = 10.1.8.64$, on n'a pas correspondance, ce n'est pas une ligne candidate.

On a fini de parcourir les 15 entrées de la table de routage. On a obtenu la liste des lignes de sortie candidates suivantes : 1, 2, 11, 13 et 14.

N°	Réseau Destination	Mask réseau Dest	Prochain Routeur	Métrique	lrf associée
1	0.0.0.0	/0	10.1.3.65	1	eth1
2	10.1.0.0	/16	10.1.3.65	1	eth1
3	10.1.3.0	/24	10.1.3.1	0	eth1
4	10.1.3.64	/26	10.1.3.126	0	eth0
5	10.1.3.128	/26	10.1.3.190	0	eth0
6	10.1.3.192	/26	10.1.3.254	0	eth0
7	10.1.4.0	/24	10.1.3.4	11	eth1
8	10.1.4.64	/26	10.1.3.4	9	eth1
9	10.1.4.128	/26	10.1.3.4	10	eth1
10	10.1.16.64	/26	10.1.3.65	5	eth1
11	10.1.8.0	/24	10.1.3.65	6	eth1
12	10.1.8.0	/26	10.1.3.65	9	wlan0
13	10.1.8.64	/26	10.1.3.65	17	ppp2
14	10.1.8.64	/26	10.1.3.62	22	ppp0
15	10.1.8.128	/26	10.1.3.65	25	ppp1

Il faut progresser dans notre démarche pour sélectionner la "meilleure" ligne de sortie. C'est l'objet de la question suivante.

Question 2

Lorsque l'on a opéré ce premier élagage on réalise un second élagage conduisant à choisir les routes qui sont associées aux masques les plus longs. Pourquoi les routeurs doivent-ils réaliser une recherche de "correspondance la plus longue" ('Longest Match based **forwarding** algorithm') ?

Donner le numéro des entrées qui passent ce second filtrage.

Correction :

Plus le masque est long plus la correspondance avec la partie réseau contenue dans l'adresse de destination est longue et donc plus on a de chance d'être proche du réseau de destination si on emprunte la ligne de sortie correspondante...

Ca s' imagine bien si on raisonne en terme de similitude. Plus la correspondance est longue, plus il y a de similitude entre les parties réseaux des deux adresses (adresse réseau dans la table, adresse de destination dans le datagramme).

Mais attention, c'est une promesse pas une assurance.

Dans les routeurs, il y a souvent une ligne contenant 0.0.0.0/0, c'est la "route par défaut". On est sûr que cette ligne sera toujours candidate puisqu'on applique un masque /0. Implicitement, le routeur qui est au bout de cette ligne de sortie a une meilleure connaissance des routes de l'Internet ou a lui-même un routeur par défaut. Ce type d'organisation pour résoudre des problèmes de chemin, d'accès à une ressource est hiérarchique et se retrouve souvent, le DNS (Domain Name Service) en est un autre exemple.

Dans notre cas, les lignes qui restent par rapport au critère de la plus longue correspondance sont les 13 et 14, le masque étant /26.

Question 3

Les routeurs peuvent ensuite arbitrer entre différentes routes selon un routage de la pomme de terre chaude. Dans le paquet IP il existe dans l'entête une zone dédiée à la qualité de service du paquet (TOS 'type of service'), qui contient deux informations.

Rappelez la signification de ces deux informations de qualité de service.

Que peut faire un routeur pour utiliser cette zone et filtrer entre les différentes routes encore jugées équivalentes?

Question 4

Un quatrième filtrage peut utiliser la métrique.

Rappeler la signification de cette valeur.

Comment utiliser cette entrée dans la table de routage pour filtrer entre les différentes routes qui seraient encore jugées équivalentes ?

Question 5

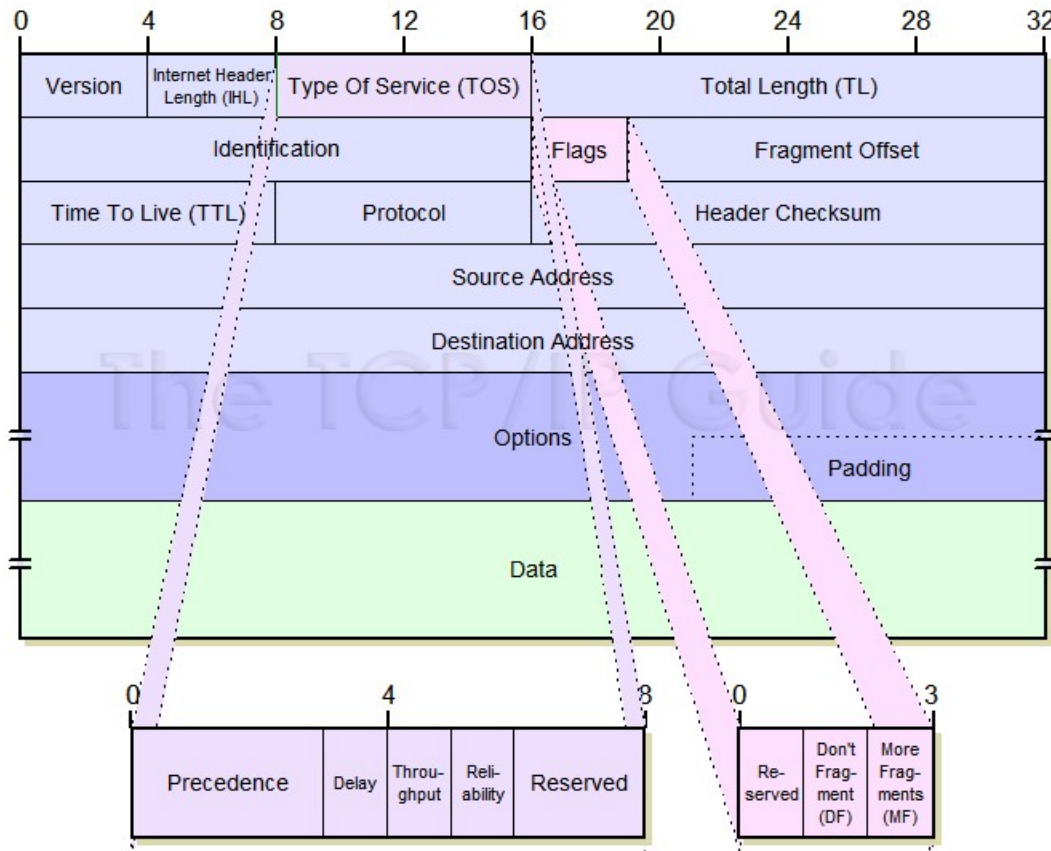
Un cinquième filtrage peut mettre en avant des techniques propres à chaque fabricant de routeur.

Quelle optimisation plus globale au réseau peut-on encore réaliser si l'on a encore

plusieurs routes jugées équivalentes ?

Correction 3, 4 et 5 :

On reprend l'entête d'un datagramme IP, et on extrait le champ ToS, Type of Service en anglais, ou Type de Service en français.



Le champ gestion du datagramme est découpé comme suit :

0	1	2	3	4	5	6	7
Préséance			Type de Service				
			D	T	R	C	

- La préséance est l'équivalent d'une priorité ou d'une marque d'importance qui est codée sur 3 bits. Celui qui a la préséance la plus élevée est transmis en premier :
 - 000 : normal
 - 001 : prioritaire
 - 010 : immédiat
 - 011 : urgent
- Le champ TOS (Type Of Service)"Type De Service" est lié à la métrique utilisée pour le routage du datagramme :
 - bit **T**: Débit (Throughput) -> demande le plus grand débit
 - bit **R**: Fiabilité (Reliability/Error Rate) -> demande le plus faible taux d'erreur

- bit **C**: Coût minimal (Cost) -> demande un coût minimal
- bit **D**: Délais courts (Delay) -> demande le plus court délai (évite les satellites)

On peut combiner plusieurs métriques pour router un datagramme, mais dans ce cas, on doit certainement paramétrer le routeur pour indiquer quelle métrique est prépondérante sur quelle autre.

Si deux routes sont jugées équivalentes, le routeur peut choisir la sortie qui "coûte" le moins cher. Ici la ligne 13 a un coût de 17 et la ligne 14 un coût de 22. On va donc choisir la ligne 13 ?

Si il n'y a qu'une métrique utilisée et qu'il reste des routes équivalentes, il faut choisir d'autres critères.

S'il y a plusieurs métriques associées aux lignes de sorties, on peut appliquer un nouveau critère de coût. Il peut toujours rester des routes équivalentes et on est ramené à la situation du paragraphe ci-dessus.

On peut imaginer que les équipements suivant les fournisseurs proposent des critères spécifiques. Par exemple, on peut imaginer une stratégie d'équilibrage de charge (load balancing) entre deux sorties équivalentes. Le routeur aussi s'adapte en fonction du taux de remplissage des files d'attente des interfaces de sorties, et il choisit la moins chargée/pleine. Il peut aussi choisir aléatoirement une des sorties, cette politique s'apparente à la technique de la "patate chaude". On peut considérer aussi qu'il y a une route primaire, et une route backup qui sert si la route primaire tombe en panne. Les possibilités sont aussi nombreuses que les offres imaginées par les équipementiers.

Question 6

S'il n'existe plus lors de l'une des étapes précédentes de route possible pour atteindre le destinataire, que se passe-t'il ?

Correction :

On a donné la réponse dans la correction de la question 1. S'il n'y a pas de lignes candidates à l'issue du parcours de la table de routage, on se trouve en présence d'une erreur de routage, le datagramme est éliminé et le routeur prévient la source (via l'@IP source contenue dans le datagramme) comme quoi la destination n'est pas atteignable (HOST UNREACHABLE).

Exercice 3 : Routage IP sur un hôte

On vous donne l'extrait du résultat de la commande ipconfig sur une machine quelconque.

```
Carte Ethernet Connexion au réseau local filaire eth0 :
  Suffixe DNS propre à la connexion. . . : cnam.fr
  Adresse IPv4. . . . . : 163.173.231.107
  Masque de sous-réseau. . . . . : 255.255.255.0
  Passerelle par défaut. . . . . : 163.173.231.2
```

```
Carte réseau sans fil Connexion réseau sans fil wif0 :
  Suffixe DNS propre à la connexion. . . : cnam.fr
  Adresse IPv4. . . . . : 163.173.112.23
  Masque de sous-réseau. . . . . : 255.255.255.0
  Passerelle par défaut. . . . . : 163.173.112.2
```

On en déduit que la machine possède 2 interfaces.



Question 1

Paramètres de configuration

- Donner la notation compacte pour le masque (/n)

Correction :

Le masque est 255.255.255.0, ce qui correspond à 24 bits à 1 sur la partie gauche du masque. La notation compacte est donc /24.

- Donner l'adresse de réseau IP associée correspondant à chacune des interfaces

Correction :

L'interface Ethernet a l'adresse IP : 163.173.231.107

L'interface Wifi a l'adresse IP : 163.173.112.23

- Donner l'adresse IP du routeur associé pour chacun des réseaux IP auxquels ces interfaces appartiennent.

Correction :

Routeur pour la carte Ethernet : 163.173.231.2

Routeur pour la carte Wifi : 163.173.112.2

- Donner l'adresse de diffusion associée à chacun des réseaux IP apparaissant dans les résultats ci-dessus

Correction :

Adresse de diffusion pour le réseau IP de l'interface Ethernet : 163.173.231.255

Adresse de diffusion pour le réseau IP de l'interface Wifi : 163.173.112.255

Suite du problème. En fait, le réseau IP a été mal paramétré, on conçoit un nouveau plan d'adressage.

Sur la machine, on veut envoyer le datagramme IP d'adresse de destination 163.173.228.26.

Question 2

Quand elle met en fonctionnement que sa carte Ethernet on a la table de routage locale suivante active:

	Réseau/mask	Next hop	métrique	accessibilité	interface
L1	0.0.0.0/0	163.173.231.2	10	distant	eth0
L2	127.0.0.0/8	0.0.0.0	0	direct	lo0
L3	163.173.228.0/24	0.0.0.0	0	direct	eth0

Quelle ligne de la table de routage emprunte le datagramme d'adresse IP destination 163.173.228.26 pour sortir du routeur et atteindre cette destination ? Estimer combien de routeurs au minimum sont traversés et pourquoi ? Expliquez brièvement votre réponse.

Correction :

On passe le masque associé à chacune des lignes de la table de routage sur 163.173.228.26. soit les masques : /0 /8 /24. 163.173.228.26/0 donne pour résultat 0.0.0.0, la ligne 1 de la table de routage est donc candidate pour la sortie du datagramme. 163.173.228.26/8 donne pour résultat 163.0.0.0 ce qui est bien différent de 127.0.0.0, la

2^{ème} ligne n'est donc pas une ligne de sortie candidate. 163.173.228.26/24 donne pour résultat 163.173.228.0 ce qui correspond exactement à l'adresse de réseau de la ligne 3. La ligne 3 est la ligne pour laquelle la correspondance est la longueur la plus longue pour le masque de l'adresse réseau (24 bits), 163.173.228.26 va donc être émis par l'interface qui figure sur la ligne 3. Le datagramme va être délivré directement à l'interface destinatrice sans passer par un routeur.

Question 3

Quand la station E ne met en fonctionnement que sa carte Wifi, on a la table de routage locale suivante active :

	Réseau/mask	Next hop	métrique	accessibilité	Interface
L1	0.0.0.0/0	163.173.112.2	10	distant	wif0
L2	127.0.0.0/8	0.0.0.0	0	direct	lo0
L3	163.173.112.0/24	0.0.0.0	0	direct	wif0

Quelle ligne de la table de routage emprunte le datagramme d'adresse IP destination 163.173.228.26 pour sortir du routeur et atteindre cette destination ? Estimer combien de routeurs au minimum sont traversés et pourquoi ? Expliquez brièvement votre réponse.

Correction :

Le datagramme avec la destination 163.173.228.26 va emprunter la ligne 1 pour sortir de la station. C'est la seule ligne dont le résultat après application du masque correspondant sur l'adresse destination donne un résultat identique. 163.173.228.26/0 donne 0.0.0.0 !

Du coup il va passer par le routeur dont la carte interface relais est 163.173.112.2. On va au moins traverser un routeur pour atteindre la destination. Comme la destination est dans le même réseau de campus 163.173.0.0/16, il y a de fortes chances que le datagramme ne traverse pas plus de deux routeurs : un routeur de département, et un routeur de campus. Pour faire une réponse exacte, on ne dispose pas de suffisamment d'éléments dans le texte de la question.

Question 4

La station E active son interface Wifi. On effectue la commande tracert et on obtient le résultat suivant :

Détermination de l'itinéraire vers 163.173.228.26 avec un maximum de 30 sauts :

```

1      5 ms      2 ms      6 ms    163.173.112.2
2      3 ms      9 ms     47 ms    163.173.228.26
```

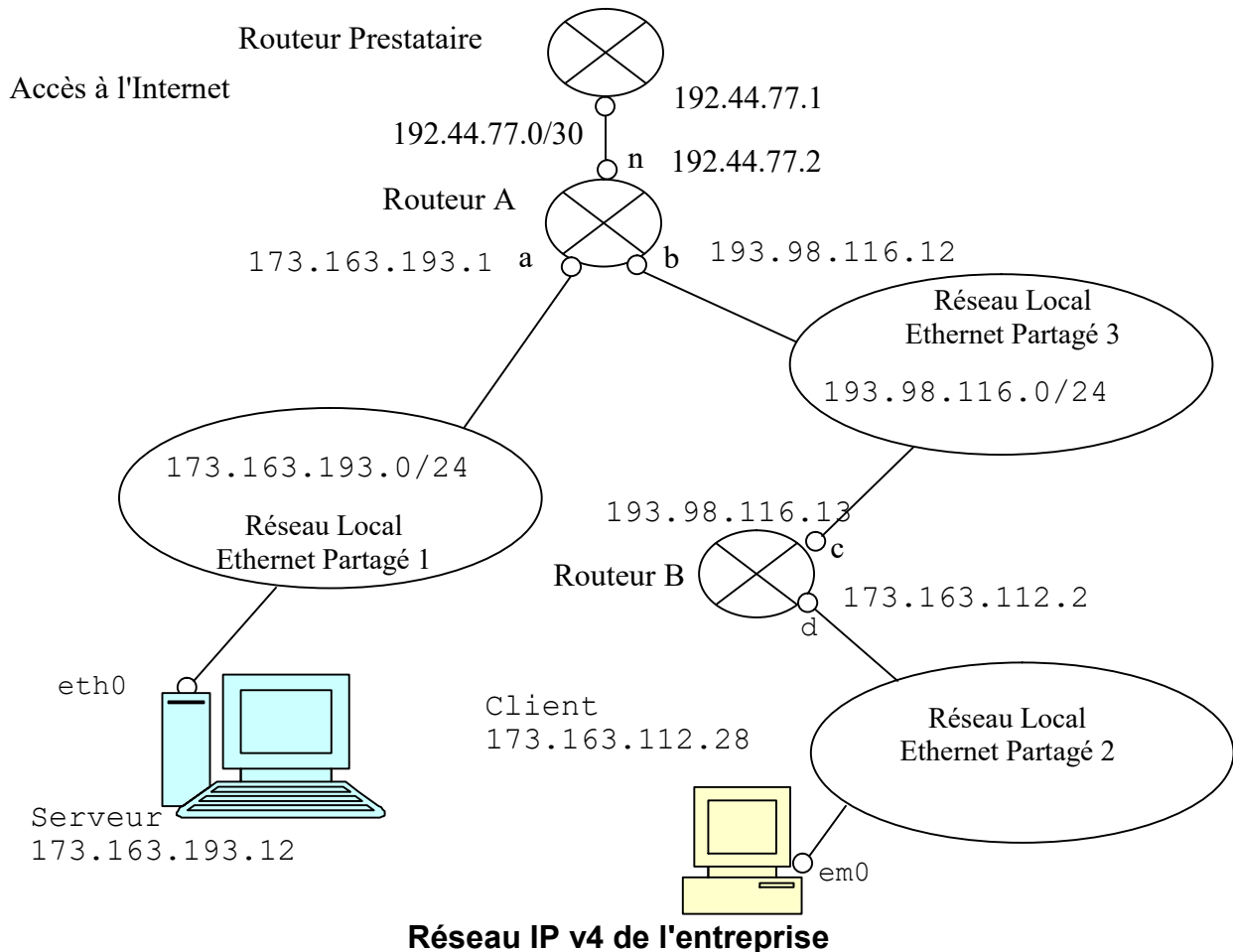
Ce choix est-il cohérent avec votre réponse à la question 3 ? Pourquoi ?

Correction :

Le résultat de tracert montre qu'on traverse un seul routeur pour aller à la destination 163.173.228.26. C'est cohérent avec la réponse à la question précédente où on indiquait qu'on traversait au moins un routeur.

Exercice 4 : Fonctionnement du Routage IP V4 dans un trajet Client-Serveur, illustration de l'acheminement d'un datagramme à travers un ensemble de sous-réseaux

Dans le réseau de la figure ci-après on raisonne suivant un adressage de type CIDR (Classless Inter Domain Routing). On travaille sur le réseau suivant :



Vous observerez que certaines informations sont manquantes, par exemple pour le réseau local Ethernet partagé 2. Les questions de l'exercice vont vous permettre de compléter les informations manquantes.

On s'intéresse à tout ce qui peut être lié à l'acheminement d'une source vers une destination IP. Les routeurs sont configurés manuellement (routage statique).

Soit une station qu'on désignera par Client. Son système d'exploitation est de type Windows. On applique la commande `ipconfig/all`, et on obtient l'affichage suivant :

```
Suffixe DNS propre ... la connexion. : manc.fr
Description. . . . . : Ethernet LAN 802.3 NIC
Adresse physique . . . . . : 4C-ED-DE-E5-A8-3A
Adresse IPv4 . . . . . : 173.163.112.28
Masque de sous-réseau. . . . . : 255.255.255.0
Passerelle par défaut. . . . . : 173.163.112.2
Serveurs DNS. . . . . : 173.163.128.6
```

Question 1

Donner le masque de sous-réseau auquel appartient Client en notation compacte à partir des résultats de la commande `ipconfig` ci-dessus.

Correction :

Le masque donné est 255.255.255.0, en notation compacte c'est /24.

Question 2

Quelle est l'adresse IPv4 correspondant à l'interface Ethernet d'adresse MAC : 4C-ED-DE-E5-A8-3A ?

Correction :

D'après le résultat de la commande `ipconfig`, c'est : 173.163.112.28.

Question 3

Quelle est l'adresse IP du réseau auquel appartient cette interface ? Expliquez très brièvement comment vous la trouvez.

Correction :

L'adresse IP du réseau auquel appartient cette interface est calculée en appliquant le masque /24 à l'adresse IP 173.163.112.28 soit 173.163.112.28/24. On obtient la valeur 173.163.112.0. Le réseau a donc l'adresse IP 173.163.112.0/24.

Question 4

Quelle est l'adresse de broadcast IP associée à ce réseau IP ? Expliquez très brièvement comment vous la trouvez.

Correction :

Pour obtenir l'adresse de broadcast du réseau, on remplace les bits de l'adresse de réseau non couverts par le masque /24 soit les 8 derniers par des 1. Cette partie correspond à la numérotation des interfaces. L'adresse de broadcast associée à ce réseau est donc 173.163.112.255.

Question 5

Combien d'adresses IPv4 sont disponibles pour être affectées à des interfaces dans ce sous-réseau ? On supposera que chaque machine n'a qu'une seule interface réseau. Expliquez brièvement votre résultat. Observez bien le réseau local Ethernet partagé 2 avant de répondre.

Correction :

On a deux adresses interdites : l'adresse du réseau, et l'adresse de broadcast. D'après les hypothèses de la question, Client utilise déjà une adresse, et, on peut supposer, même si on ne la connaît pas, que la carte réseau Ethernet du routeur B a une adresse IP. Sachant que le masque est /24, cela laisse 2^8 soit 256 adresses IP potentielles sur ce réseau IP. Il faut en enlever 2 (les deux adresses IP réservées : le réseau lui-même et le broadcast sur ce réseau) et 2 autres (173.163.112.28-em0, 173.163.112.2-interface d du routeur B). On a 256-4 adresses encore disponibles pour numéroté des interfaces dans ce sous-réseau. Soit 252 adresses IP peuvent être affectées dans ce sous réseau IP à des interfaces.

Question 6

L'adresse IP de l'interface du routeur associée à ce réseau IP V4 est 173.163.112.2. Complétez les cases vides de la table de routage de la machine Client ci-après.

On indique que le nom de l'interface réseau de la machine Client est em0.



Réseau IP/mask	Next-Hop	Commentaire	Inter-face	Accessibilité
0.0.0.0/0		Route par défaut	em0	distant
127.0.0.0/8	0.0.0.0	Loopback, on ne passe par la carte NIC	lo0	direct
	0.0.0.0	Le réseau IP où je suis connecté, on passe par la carte NIC	em0	direct

Correction :

Réseau IP/mask	Next-Hop	Commentaire	Inter-face	Accessibilité
0.0.0.0/0	173.163.112.2	Route par défaut	em0	distant
127.0.0.0/8	0.0.0.0	Loopback, on ne passe par la carte NIC	lo0	direct
173.163.112.0/24	0.0.0.0	Le réseau IP où je suis connecté, on passe par la carte NIC	em0	direct

Question 7

Si une trame Ethernet part de l'interface em0. Quelle sera l'adresse MAC dans le champ source de cette trame ?

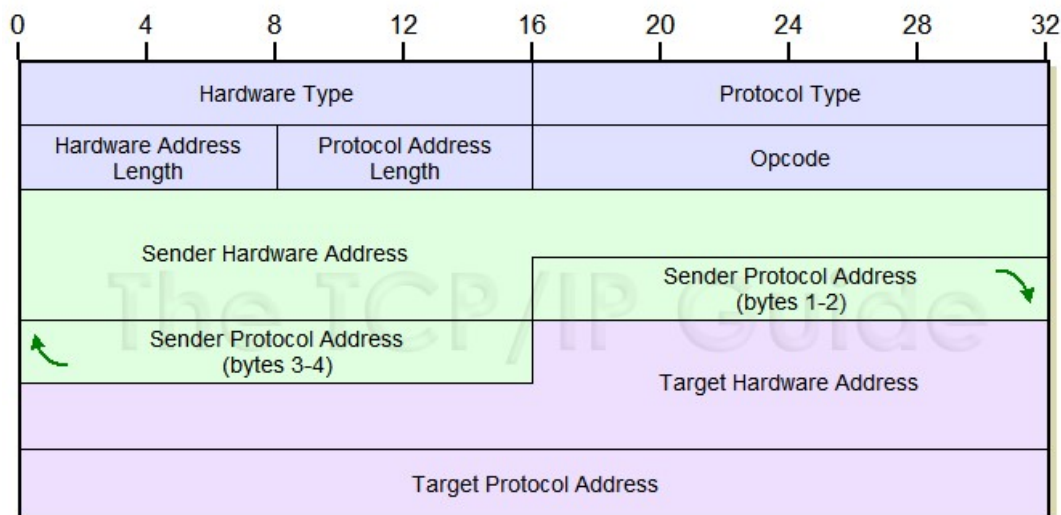
Correction :

L'adresse MAC dans le champ source de la trame sera 4C-ED-DE-E5-A8-3A.

Question 8

La station Client ne connaît pas l'adresse Ethernet de l'interface du routeur reliée à son sous-réseau IP. Elle effectue donc une requête ARP (Address Resolution Protocol) pour la connaître.

On donne le format d'une requête ARP telle qu'elle est définie dans le document RFC826 de l'IETF et dessinée au lien http://www.tcpipguide.com/free/t_ARPMessageFormat.htm (consulté le 30/11/2016).



Format d'une ARP (demande ou réponse)

- Le champ `Hardware Type` vaut 1 pour des adresses Ethernet. Pour ARP "Hardware" est synonyme de Ethernet.
- Le champ `Protocol Type` vaut 0800 en hexadécimal pour Internet. Pour ARP "Protocol" est synonyme de Internet.
- `Opcode` vaut 1 pour une requête de résolution d'adresse IP ARP, et, 2 pour la réponse à une résolution d'adresse.
- `Sender` correspond à l'émetteur du message ARP.
- `Target` correspond au destinataire du message ARP.

A partir de la figure "Format d'une unité de données de protocole ARP", compléter la requête ARP ci-dessous en remplissant la partie manquante.

Correction :

0001		0800	
06	04	0001	
4C	-	ED	- DE - E5
-A8	-	3A	173 . 163 .
112	.	28	00 00
00	00	00	00
173	.	163	. 112 . 2

Question 9

Comme Client ne connaît pas l'adresse Ethernet qui correspond à l'adresse IP de destination 173.163.112.2 pour atteindre le routeur, va-t-il utiliser une adresse destination MAC d'Ethernet de type multicast (diffusion sur un groupe d'adresses MAC, par exemple un groupe de routeurs) ou broadcast (diffusion à toutes les adresses MAC du réseau local ff:ff:ff:ff:ff:ff) ?

Correction :

Pour faire parvenir la requête ARP à la bonne machine destinatrice dont il ne connaît pas l'adresse MAC, une seule solution envoyer en diffusion sur tout le réseau local Ethernet. Il utilise donc l'adresse MAC dédiée : ff:ff:ff:ff:ff:ff

On rappelle la structure d'une trame Ethernet :

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
6 octets	6 octets	2 octets	46 à 1500 octets	4 octets

Question 10

Remplissez les champs Adresses MAC de la trame Ethernet envoyée par la carte Ethernet de Client (em0) et qui contient la requête ARP ci-dessus.

On vous indique que le champ type de la trame Ethernet transportant une requête ARP est 0x0806.

Correction :

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
FF:FF:FF:FF:FF:FF	4C-ED-DE-E5-A8-3A	0x0806	DDDDDDDD...DDD	XXXXXXXX

Question 11

Va-t-il être nécessaire d'ajouter du bourrage dans la partie données de la trame Ethernet encore appelée charge utile ci-dessus. Si oui, combien d'octets de bourrage sont ajoutés.

Correction :

La requête ARP fait 28 octets de longueur. Elle doit s'insérer dans la partie consacrée aux données de la trame qui fait une taille minimum de 46 octets de données. $28 < 46$, donc il faut ajouter du bourrage. On ajoute la différence soit 18 octets de bourrage à la requête ARP.

Question 12

Donnez la réponse ARP du routeur si l'adresse Ethernet de son interface d'adresse IP 173.163.112.2 est 4C-ED-DE-E5-09-4B.

Remplissez la partie manquante de la réponse ARP ci-après.

0001		0800	
06	04	0002	
		173	163
.112	2	4C - ED	-
DE - E5	-	A8 - 3A	
173	163	112	28

Correction :

0001		0800	
06	04	0002	
4C - ED	-	DE - E5	
- 09 - 4B		173	163
.112	2	4C - ED	-
DE - E5	-	A8 - 3A	
173	163	112	28

On rappelle que le datagramme qui va du client vers le serveur est acheminé dans une trame qui le contient comme charge utile. Ce datagramme applicatif a l'entête ci-dessous. Seuls les champs qui sont les plus significatifs pour le problème à résoudre dans la question ci-après sont explicités.

0	1	2	3
0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9	0 1 2 3 4 5 6 7 8 9 0 1
+	+	+	+
4	5	0 0	Total Length
+	+	+	+
	Identification	Flags	Fragment Offset
+	+	+	+
Time to Live	06		Header Checksum
+	+	+	+
	173.163.112.28 (source client)		
+	+	+	+
	173.163.193.12 (destination serveur)		
+	+	+	+

Entête du Datagramme IP envoyé de la source 173.163.112.28 à la destination 173.163.193.12

Question 13

Remplir tous les champs d'adresse MAC de la trame Ethernet qui va du Client au routeur B et qui contient le datagramme IP à émettre de la figure ci-après.

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
			DDDDDDDD...DDD	XXXXXXXX

Correction :

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
4C-ED-DE-E5-09-4B	4C-ED-DE-E5-A8-3A	0x0800	DDDDDDDD...DDD	XXXXXXXX

Question 14

Est-ce que le datagramme contenu dans la trame répondue à la question 13 va s'arrêter sur le prochain routeur ou va poursuivre son chemin ? Expliquer brièvement pourquoi.

Correction :

Le datagramme va poursuivre sa route à travers le réseau mais la trame qui le contient va s'arrêter à l'interface Ethernet qui l'aura reçue.

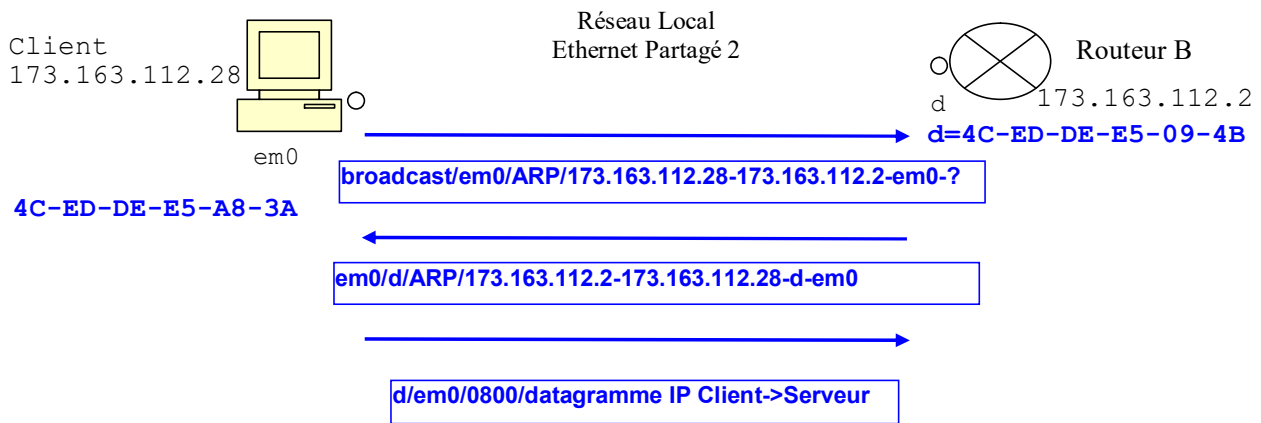
Question 15

Pour le datagramme ci-dessus, qu'elle est la taille de l'entête IP en nombre d'octets ? Y a-t-il des options dans l'entête du datagramme ?

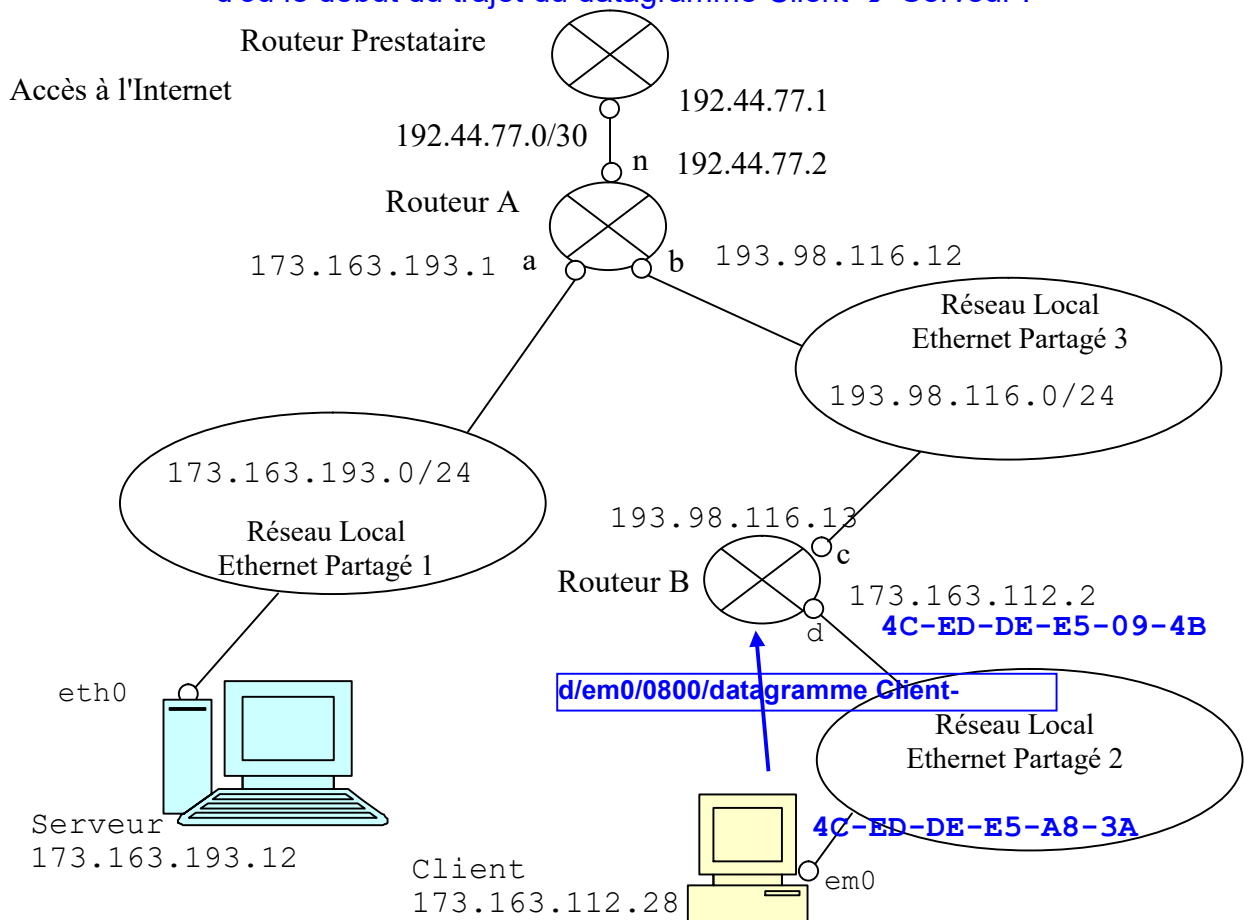
Correction :

L'entête fait 5 mots de 4 octets, donc 20 octets. C'est la taille minimale d'une entête IP. Il n'y a donc pas d'options dans l'entête du datagramme IP.

Ci-après, un schéma récapitulatif de ce qui s'est passé jusqu'à maintenant.



d'où le début du trajet du datagramme Client → Serveur :



Le routeur destinataire de la trame est le routeur B. La table de routage du routeur B est la suivante :

lig	Destination	Next hop	Port	Type
1	0.0.0.0/0	193.98.116.12	c	distant
2	193.98.116.0/24	0.0.0.0	c	direct
3	173.163.112.0/24	0.0.0.0	d	direct

Question 16

Une fois le datagramme arrivé sur le routeur B. Quelle est la ligne de la table de routage de B qui va être sélectionnée pour le faire sortir et atteindre la destination Serveur qui a l'adresse IPv4 173.163.193.12. Expliquez brièvement pourquoi.

Correction :

La destination du datagramme est l'adresse IP 173.163.193.12. On applique successivement chaque masque contenu dans les lignes de la table de routage.

- 173.163.193.12/0 = 0.0.0.0, ligne 1 candidate, réponse identique à la colonne Next hop.
- 173.163.193.12/24 = 173.163.193.0, ligne 2 pas candidate, car adresse différente
- 173.163.193.12/24 = 173.163.193.0, ligne 3 pas candidate, car adresse différente

Il n'y a que la ligne 1. le next hop est 193.98.116.12 et le datagramme passe par le port c via le réseau local Ethernet 3.

Question 17

Est-ce que les adresses IP source et/ou destination contenues dans le datagramme envoyé par Client vont être modifiées par le routeur B pour que le datagramme puisse atteindre sa destination finale qui est la machine Serveur ?

Correction :

Non, un routeur ne touche jamais les adresses IP à l'intérieur d'un datagramme sinon comment le routage vers la destination pourrait-il se faire. Le champ qu'il peut toucher, c'est le champ DSCP quand il gère de la QoS.

Question 18

Si l'adresse MAC de l'interface c du routeur B est : 4C-ED-DE-E5-08-5C, et, si l'adresse MAC de l'interface b du routeur A est : 4C-ED-DE-E5-07-6D.

Donner les adresses MAC source et destination de la trame qui circule entre les routeurs A et B. Vous remplirez le dessin ci-dessous.

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
		0x0800	DDDDDDDD...DDD	XXXXXXXX

Correction :

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
4C-ED-DE-E5-07-6D	4C-ED-DE-E5-08-5C	0x0800	DDDDDDDD...DDD	XXXXXXXX

Soit la table du routeur A :

lig	Destination	Next hop	Port	Type
1	0.0.0.0/0	192.44.77.1	n	distant
2	192.44.77.0/30	0.0.0.0	n	direct
3	173.163.193.0/24	0.0.0.0	a	direct
4	193.98.116.0/24	0.0.0.0	b	direct
5	173.163.112.0/24	193.98.116.13	b	distant

Question 19

Une fois le datagramme arrivé sur le routeur A. Quelle est la ligne de la table de routage de A qui va être sélectionnée pour le faire sortir et atteindre la destination Serveur qui est d'adresse IPv4 173.163.193.12. Expliquez très brièvement pourquoi.

Correction :

- 173.163.193.12/0 = 0.0.0.0, ligne 1 candidate résultat identique
- 173.163.193.12/30 = 173.163.193.12, ligne 2 pas candidate résultat différent
- 173.163.193.12/24 = 173.163.193.0, ligne 3 candidate
- 173.163.193.12/24 = 173.163.193.0, ligne 4 pas candidate résultat différent
- 173.163.193.12/24 = 173.163.193.0, ligne 5 pas candidate résultat différent

La ligne 3 a un masque plus long que la ligne 1, c'est donc la ligne retenue

Le datagramme poursuit sa route en étant émis par l'interface a du routeur A sur le réseau local Ethernet 1.

Question 20

Est-ce que la trame qui sera générée par le routeur A va atteindre l'hôte Serveur sans aucun relaiage par un routeur ? Pourquoi ?

Correction :

La ligne 3 indique un next hop qui vaut 0.0.0.0 et un type d'acheminement direct. La trame va permettre de mener le datagramme à la destination IP directement.

Question 21

Si l'adresse MAC de l'interface a du routeur A est : 4C-ED-DE-E5-06-5E. Et si l'adresse MAC de Serveur est 4C-ED-DE-E5-05-6F. Donner les adresses MAC source et destination de la trame qui est envoyée par le routeur A.

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
		0x0800	DDDDDDDD...DDD	XXXXXXXX

Correction :

Adresse MAC destination	Adresse MAC source	Type	Charge utile - Données	FCS- contrôle d'erreur
4C-ED-DE-E5-05-6F	4C-ED-DE-E5-06-5E	0x0800	DDDDDDDD...DDD	XXXXXXXX

Question 22

Si on effectue la commande `tracert Serveur` sur la machine Client, donner la liste des adresses ip que va afficher le résultat de la commande.

Correction :

Si on effectue depuis Client la commande `tracert Serveur`. On obtient la liste d'adresses IP suivantes :

173.163.112.2 puis 193.98.116.12 puis 173.163.193.12

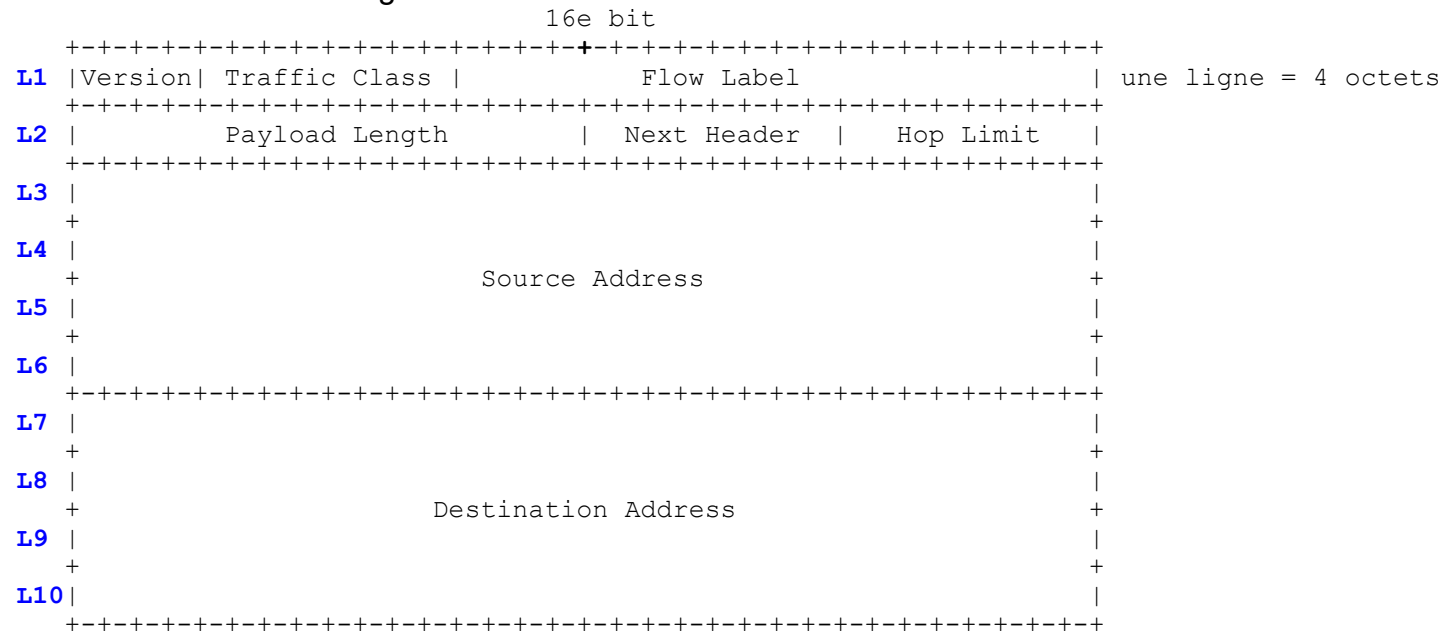
Le résultat est différent en sens inverse... chercher pourquoi.

Exercice 5 : Découverte d'un des services d'ICMPv6 : Neighbor Discovery équivalent à ARP pour IPv4 – facultatif mais recommandé pour tous ceux qui feront du réseau que ça soit un peu, beaucoup, passionnément ou à la folie !!!

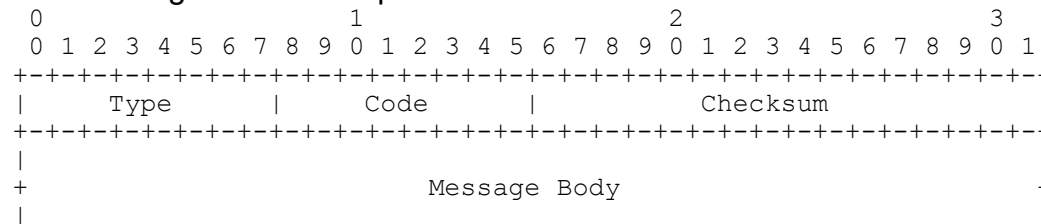
On donne la structure globale d'une trame Ethernet :

Ethernet II				
Destination MAC 6 Bytes	Source MAC 6 Bytes	Type 2 Bytes	Data 46 – 1500 Bytes	Frame Check Sequence 4 Bytes

On donne la structure de l'entête d'un datagramme IPv6 suivant la RFC8200 :



On donne le format d'un message ICMPv6 d'après la RFC 4443 :



On effectue la capture de trames suivante avec un analyseur de protocole, ici Wireshark :

Time	Source	Destir	Proto	Lengt	Info
1 0.000000	fe80::c001:2f...	ff...	IC...	86	Neighbor Solicitation for fe80::c002:3ff:fee4:0 from c2:01:02:40:00:00
2 0.024297	fe80::c002:3f...	fe...	IC...	86	Neighbor Advertisement fe80::c002:3ff:fee4:0 (sol, ovr) is at c2:02:03:e4:00:00
3 5.028119	fe80::c002:3f...	fe...	IC...	86	Neighbor Solicitation for fe80::c001:2ff:fe40:0 from c2:02:03:e4:00:00
4 5.055978	fe80::c001:2f...	fe...	IC...	78	Neighbor Advertisement fe80::c001:2ff:fe40:0 (sol)

Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)
 Ethernet II, Src: c2:01:02:40:00:00 (c2:01:02:40:00:00), Dst: IPv6mcast_ff:e4:00:00 (33:33:ff:e4:00:00)
 > Destination: IPv6mcast_ff:e4:00:00 (33:33:ff:e4:00:00)
 > Source: c2:01:02:40:00:00 (c2:01:02:40:00:00)
 Type: IPv6 (0x86dd)
 Internet Protocol Version 6, Src: fe80::c001:2ff:fe40:0, Dst: ff02::1:ffe4:0
 0110 = Version: 6
 ▾ 1110 0000 = Traffic Class: 0xe0 (DSCP: CS7, ECN: Not-ECT)
 1110 00.. = Differentiated Services Codepoint: Class Selector 7 (56)
 00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
 0000 0000 0000 0000 0000 = Flow Label: 0x000000
 Payload Length: 32
 Next Header: ICMPv6 (58)
 Hop Limit: 255
 Source: fe80::c001:2ff:fe40:0
 Destination: ff02::1:ffe4:0
 Internet Control Message Protocol v6
 Type: Neighbor Solicitation (135)
 Code: 0
 Checksum: 0x334f [correct]
 [Checksum Status: Good]
 Reserved: 00000000
 Target Address: fe80::c002:3ff:fee4:0
 ▾ ICMPv6 Option (Source link-layer address : c2:01:02:40:00:00)
 Type: Source link-layer address (1)
 Length: 1 (8 bytes)
 Link-layer address: c2:01:02:40:00:00 (c2:01:02:40:00:00)

00	33 33 ff e4 00 00 c2 01 02 40 00 00 86 dd 6e 00	33-----@----n-
10	00 00 00 20 3a ff fe 80 00 00 00 00 00 00 c0 01	... :...
20	02 ff fe 40 00 00 ff 02 00 00 00 00 00 00 00 00	...@-----
30	00 01 ff e4 00 00 87 00 33 4f 00 00 00 00 fe 80 30....
40	00 00 00 00 00 00 c0 02 03 ff fe e4 00 00 01 01
50	c2 01 02 40 00 00	...@--

Question 1 : On s'intéresse pour l'instant à la trame 1.

- Délimiter l'entête de la trame Ethernet dans la capture en hexadécimal ci-dessous.
 - Délimiter l'entête du datagramme IPv6 dans la capture en hexadécimal ci-dessous.
 - Délimiter le message ICMPv6 dans la capture en hexadécimal ci-dessous.
 - A quelle couche appartient ICMPv6 d'après l'observation de la trace ? Pourquoi ?
- Ne pas hésiter à utiliser des couleurs différentes pour que votre réponse soit facile à lire.

0000	33 33 ff e4 00 00 c2 01	02 40 00 00 86 dd 6e 00
0010	00 00 00 20 3a ff fe 80	00 00 00 00 00 00 c0 01
0020	02 ff fe 40 00 00 ff 02	00 00 00 00 00 00 00 00
0030	00 01 ff e4 00 00 87 00	33 4f 00 00 00 00 fe 80
0040	00 00 00 00 00 00 c0 02	03 ff fe e4 00 00 01 01
0050	c2 01 02 40 00 00	

Attention la colonne la plus à gauche numérote les lignes et cette numérotation est hexadécimale, pour ne pas confondre la police est différente.

Correction :

- Délimiter l'entête de la trame Ethernet dans la capture en hexadécimal ci-dessous.
- Délimiter l'entête du datagramme IPv6 dans la capture en hexadécimal ci-dessous.
- Délimiter le message ICMPv6 dans la capture en hexadécimal ci-dessous.

```

0000  33 33 ff e4 00 00 c2 01 02 40 00 00 86 dd 6e 00
          MAC dest          MAC source      Type version IP=> 6
          entête Ethernet

0010  00 00 00 20 3a ff fe 80 00 00 00 00 00 00 c0 01
                                adresse IPv6 source
          entête IP

0020  02 ff fe 40 00 00 ff 02 00 00 00 00 00 00 00 00
                                adresse IPv6 dest

0030  00 01 ff e4 00 00 87 00 33 4f 00 00 00 00 fe 80

0040  00 00 00 00 00 00 c0 02 03 ff fe e4 00 00 01 01
                                message ICMPv6

0050  c2 01 02 40 00 00

```

- A quelle couche appartient ICMPv6 d'après l'observation de la trace ? Pourquoi ? ICMP est au moins au dessus de la couche 2 car il s'appuie directement sur IP qui est en couche 3, ICMP est-il en couche 4 ou en couche 3 ? Internet Control Message Protocol est en fait un protocole de service compagnon d'IP, donc on le range en couche 3 même s'il est encapsulé dans 3 couches inférieures (Physique, Ethernet, IPV6).

Retrouver les champs suivants dans la trace hexadécimale ci-dessus et entourez les :

- Quelle est l'adresse Ethernet destination en **hexadécimal** ? 33 33 ff e4 00 00
- Quelle est l'adresse Ethernet source en **hexadécimal** ? c2 01 02 40 00 00
- Quel est le type de la trame en **hexadécimal** ? 86 dd, 0x86DD correspond à une charge utile de type IPv6, ce qui est confirmé par le demi octet qui suit qui lui est dans l'entête du datagramme IPv6

- Quelle est la version du protocole IP en **décimal** ? **6** est ce qu'on récupère dans la trace hexadécimale, mais c'est aussi la même valeur en décimal.
- Quelle est la longueur de l'entête IP en **décimal** ? Elle est fixe, il n'y a pas d'option en IPv6 puisqu'aucun champ ne porte ce nom dans le format. Elle fait 40 octets comme on peut le compter sur le format donné en début d'exercice : 10 lignes de 4 octets. Les options en IPv6 sont gérées comme des protocoles de niveau supérieur à l'aide du champ "Next Header".
- Quelle est l'adresse IPv6 source en **hexadécimal** ? **fe 80 00 00 00 00 00 00 c0 01 02 ff fe 40 00 00** ou encore **fe80:0000:0000:0000:c001:02ff:fe40:0000** qui se note aussi **fe80::c001:02ff:fe40:0** comme on peut le voir dans la fenêtre du milieu affichée par Wireshark
- Quelle est l'adresse IPv6 destination en **hexadécimal** ? **ff 02 00 00 00 00 00 00 00 00 00 01 ff e4 00 00** ou encore **ff02:0000:0000:0000:0000:0001:ffe4:0000** qui se note aussi **ff02::1:ffe4:0** comme on peut le voir dans la fenêtre du milieu affichée par Wireshark
- La longueur de la charge utile (PAYLOAD) est de 32 octets que comptabilise-t-elle exactement ? Il faut noter que "PAYLOAD" fait référence à la trace Wireshark et donc au contenu acheminé par le datagramme IPv6. Cela correspond au message ICMP et rien d'autre.

Question 2 : Adresse Multicast MAC/Ethernet pour IPv6.

Wikipedia indique "Au niveau ethernet, un préfixe OUI est réservé aux adresses IPv6 multicast (33:33:xx)". "xx" figure les octets restants d'une adresse MAC sur 48 bits. L'OUI (Organizationally Unique Identifier) identifie un fabricant de carte Ethernet. Combien d'adresses MAC peut-on associer à une OUI, justifiez votre réponse² ?

Correction :

On a au total 48 bits pour une adresse MAC, soit 6 octets. L'OUI "33:33" utilise 2 octets d'une adresse MAC, il reste "://://://://" soit 4 octets. $4 \times 8 = 32$ bits disponibles pour définir des adresses MAC par rapport à un fabricant de cartes MAC particulier identifié par les 2 premiers octets ou à un usage particulier comme le multicast IPv6 qui se décline de façon différente suivant qu'on est en IPv4 ou en IPv6. $2^{32} = 4 \times 1024 \times 1024 \times 1024 = 4$ Giga d'adresses ($4 \times 1\,073\,741\,824$) soit 4 294 967 296. Dans la pratique, autant que j'ai pu l'observer, le 3^{ème} octet à partir du début est aussi fixé par le constructeur. Il ne reste donc plus que 24 bits pour numéroter des adresses MAC différentes pour un constructeur particulier. $2^{24} = 16 \times 1024 \times 1024$ soit 16 Méga d'adresses ($16 \times 1\,048\,576$ ou 16 777 216).

Enfin dans la RFC7042, paragraphe : "2.3.1 Identifiers Prefixed "33-33"

All MAC-48 multicast identifiers prefixed "33-33" (that is, the 2^{32} multicast MAC identifiers in the range from 33-33-00-00-00-00 to 33-33-FF-FF-FF-FF) are used as specified in [RFC2464] for IPv6 multicast. In all of these

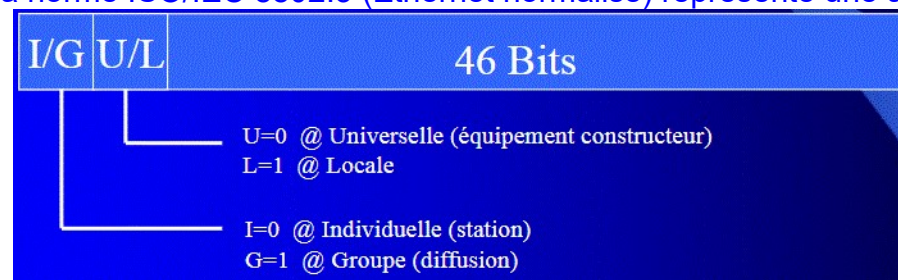
² Sans justification, même si la réponse est juste, elle n'est pas comptée et c'est 0.

identifiers, the Group bit (the bottom bit of the first octet) is on, as is required to work properly with existing hardware as a multicast identifier. They also have the Local bit on and are used for this purpose in IPv6 networks."

L'ordre des bits dans les documents des organismes de l'Internet, IETF (Internet Engineering Task Force) et les Request For Comments (RFC) par exemple, est différent de l'ordre des bits dans les documents IEEE (Institut of Electrical and Electronics Engineers) norme IEEE802.3 Ethernet. Quel ordre sur les bits d'un octet a été choisi par l'IETF, et plus généralement l'ordre sur les octets d'un mot (IPv4 par exemple) ?

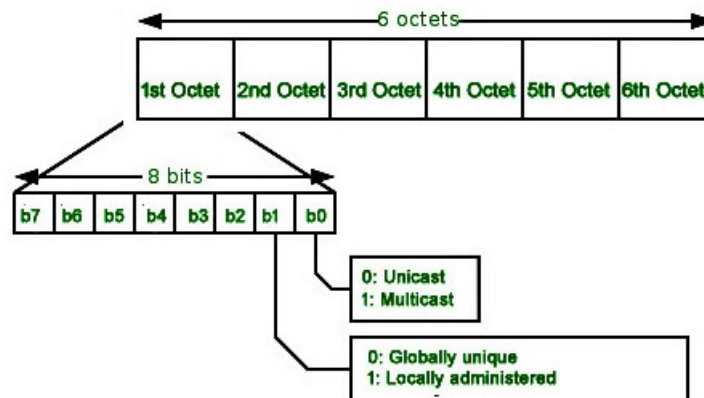
Correction :

Pour information voilà comment la norme ISO/IEC 8802.3 (Ethernet normalisé) représente une adresse MAC :



Pour l'ISO ou l'IEEE, on a le premier bit qui indique si l'adresse MAC est une adresse unicast (0) ou broadcast/multicast (1). Le deuxième bit indique si c'est une adresse universelle (0), ou encore équipement constructeur, ou une adresse locale (1), ou encore administrée localement.

L'IETF numérote les octets dans un mot suivant la représentation BigEndian, et numérote les bits dans un octet de la même façon en BigEndian. Le bit le plus significatif est rangé dans les bits de poids faible du mot. Cela donne pour les adresses MAC une représentation différente :



Une description sur Internet qui décrit la situation posée par ces représentations différentes, <https://www.askapache.com/online-tools/mac-lookup/> (05/02/2021), le texte a été reformaté p/r à l'origine mais les mots sont les mêmes :

"Bit-reversed notation

The standard notation, also called canonical format, for MAC addresses is written in transmission bit order with the least significant bit transmitted first, as seen in the output of the `iproute2/ifconfig/ipconfig` command, for example.

However, since IEEE 802.3 (Ethernet) and IEEE 802.4 (Token Bus) send the bytes (octets) over the wire, left-to-right, with least significant bit in each byte first, while IEEE 802.5 (Token Ring) and IEEE 802.6 send the bytes over the wire with the most significant bit first, confusion may arise when an address in the latter scenario is represented with bits reversed from the canonical representation.

For example:

- An address in canonical form³ 12-34-56-78-9A-BC would be transmitted over the wire as bits: 01001000 00101100 01101010 00011110 01011001 00111101 in the standard transmission order (least significant bit first).
- But for Token Ring networks, it would be transmitted as bits 00010010 00110100 01010110 01111000 10011010 10111100 in most-significant-bit first order.

The latter might be incorrectly displayed as 48-2C-6A-1E-59-3D. This is referred to as *bit-reversed order*, *non-canonical form*, *MSB format*, *IBM format*, or *Token Ring format*, as explained in RFC 2469. Canonical form is generally preferred, and used by all modern implementations."

Dans une adresse MAC, comme 33:33:FF:E4:00:00⁴, à quel bit reconnaît-on que c'est une adresse multicast, groupe de diffusion en français, et non une adresse broadcast (c'est très différent), quelle est sa position dans l'octet concerné ?

³ Canonical form, c'est la façon d'écrire les nombres des humains, de la gauche vers la droite depuis la numération arabe.

⁴ Le préfixe 33:33:FF d'une adresse MAC est dédié à l'usage du protocole ICMPv6 Neighbor Discovery



Correction :

On le reconnaît à travers le premier octet (cf les schémas donnés avant). C'est le premier bit le plus à droite de cet octet, en notation IETF, qui donne cette information. S'il vaut 1, c'est une adresse de diffusion (broadcast, à tous, ou multicast, à un groupe). 33 en hexadécimal, s'écrit 0011 0011 en binaire. Le bit le plus à droite étant à 1, c'est une adresse de diffusion. Ce n'est pas un broadcast sinon l'adresse serait ff:ff:ff:ff:ff:ff.

A votre avis, brièvement, pour quel usage a-t-on un préfixe OUI qui identifie toutes les adresses multicast MAC pour le multicast IPv6 ?

Correction :

Un OUI qui permet d'associer la famille des adresses multicast IPv6 à un sous-ensemble d'adresses MAC c'est pratique, au niveau de l'interface entre la carte LAN et la couche réseau, le pilote d'interface peut plus rapidement passer le datagramme au bon traitement de la couche réseau IPv6 ici. Donc ça permet de mieux discriminer les adresses MAC.

Question 3 : Format des adresses IPv6.

Une adresse IPv6 fait 128 bits. Combien cela représente d'octets et combien de mots de 4 octets ?

Correction :

On le voit dans le format de l'entête : une adresse IPv6 fait 4 lignes, et une ligne fait 4 octets. Donc une adresse IPv6 fait 16 octets, c'est aussi ce qu'on trouve dans les valeurs de la trace hexadécimale Wireshark. Et, $16 \times 8 = 128$... On a bien, de différentes manières, recalculer les 128bits d'une adresse IPv6.

Les adresses IPv6 montrées dans la fenêtre explicative de Wireshark (celle du milieu) sont différentes de celles que vous avez trouvées dans la trace hexadécimale (fenêtre du bas). Expliciter complètement l'adresse fe80::c001:2ff:fe40:0 de telle façon qu'il y ait 4 chiffres hexadécimaux séparés par ":".

Correction :

A l'aide de la trace Wireshark on voit que fe80::c001:2ff:fe40:0 c'est fe 80 00 00 00 00 00 00 c0 01 02 ff fe 40 00 00. On peut encore le réécrire : fe80:0000:0000:0000:c001:02ff:fe40:0000.

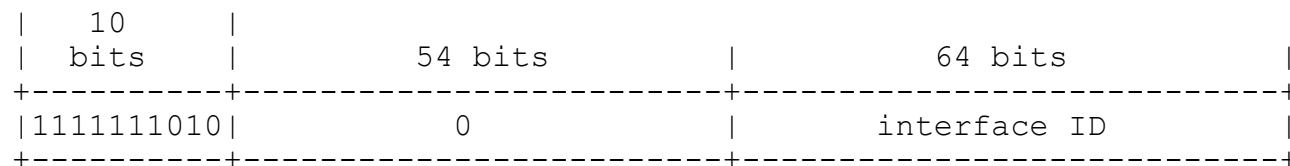
Si on applique le masque en notation compacte /10 sur l'adresse fe80::c001:2ff:fe40:0, quelle est l'adresse IPv6 résultante ?

Correction :

Le masque /10 va s'appliquer sur les 10 premiers bits les plus à gauche de l'adresse IPv6 soit fe8. "fe" va être récupéré. 8 s'écrit 1000, donc la fin du masque /10 va récupérer le 10 et mettre le reste à 0 mais qui est déjà à 0. Tout ce qui suit va passer à 0 une fois l'opération de masquage complétée. Donc fe80::c001:2ff:fe40:0/10 = fe80::

La RFC4291 section "2.5.6. Link-Local IPv6 Unicast Addresses" dit :

"Link-Local addresses are for use on a single link. Link-Local addresses have the following format:

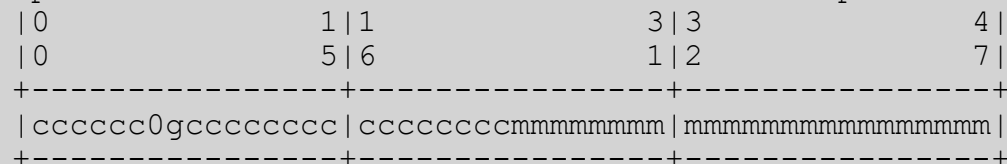


Link-Local addresses are designed to be used for addressing on a single link for purposes such as automatic address configuration, neighbor discovery, or when no routers are present.

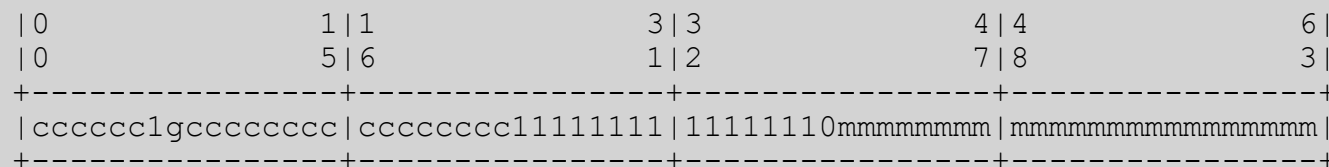
Routers must not forward any packets with Link-Local source or destination addresses to other links."

Puis dans son annexe A, y est écrit :

"[EUI64] defines a method to create an IEEE EUI-64 identifier from an IEEE 48-bit MAC identifier. This is to insert two octets, with hexadecimal values of 0xFF and 0xFE (see the Note at the end of appendix), in the middle of the 48-bit MAC (between the company_id and vendor-supplied id). An example is the 48-bit IEEE MAC with Global scope:



where "c" is the bits of the assigned company_id, "0" is the value of the universal/local bit to indicate Global scope, "g" is individual/group bit, and "m" is the bits of the manufacturer-selected extension identifier. The interface identifier would be of the form:



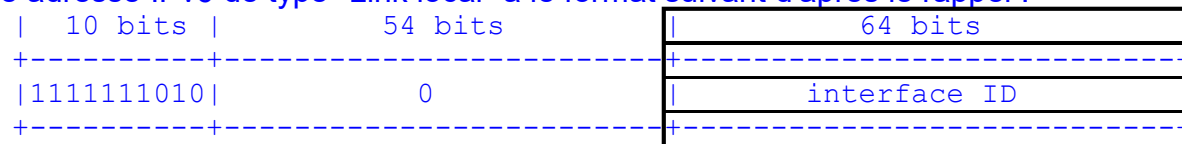
When IEEE 802 48-bit MAC addresses are available (on an interface or a node), an implementation may use them to create interface identifiers due to their availability and uniqueness properties."

Donner les 64 derniers bits de l'adresse IPv6 multicast `fe80::c001:2ff:fe40:0` qui correspondent à l'identificateur d'interface EUI-64 associé. On y retrouve l'adresse MAC Ethernet `c2:01:02:40:00:00`. mais pourquoi a-t-on `c0` dans l'adresse IPv6 au lieu de `c2` qui est dans l'adresse MAC ?

Correction : La question est donnée à tout le monde compte tenu d'un manque de précision dans son énoncé.

Il manquait une phrase dans l'extrait de la RFC : "The only change needed to transform an IEEE EUI-64 identifier to an interface identifier is to invert the "u" universal/local bit.", qui précise tout le sens de la modification à effectuer

Une adresse IPv6 de type "Link local" a le format suivant d'après le rappel :



C'est la partie encadrée d'une adresse IPv6 à laquelle on s'intéresse. Les 64 bits de fin de l'adresse IPv6 étudiée `fe80::c001:2ff:fe40:0` forment l'identificateur d'interface EUI-64 soit `c001:2ff:fe40:0`, et de façon extensive : `c0 01 02 ff fe 40 00 00`. L'adresse MAC est `c2 01 02 40 00 00`.

On trouve plusieurs correspondances :

- les 3 octets de la première moitié de l'adresse MAC, sauf le 7^{ème} bit en partant de la gauche, sont identiques aux 3 premiers octets de la partie interface ID de l'adresse IPv6 : `cx 01 02`
- le premier octet de l'adresse MAC `c2` est : `1100 0010`. Dans le texte de l'IETF, il est dit que le 7^{ème} bit doit être inversé (cf ajout ci-dessus) dans l'interface ID dans l'adresse IPv6, cet octet devient `1100 0000` soit `c0`. Après cette transformation on trouve bien les 3 premiers octets de la partie interface ID de l'adresse IPv6 : `c0 01 02`.
- au milieu de l'interface ID de 64 bits on a bien les octets "0xFF and 0xFE" précaunisés par le RFC : `c0 01 02 ff fe 40 00 00`
- les 3 octets de la moitié de fin de l'adresse MAC sont identiques aux 3 octets de fin de l'adresse IPv6 : `40 00 00`

La figure ci-dessous éclaire un peu mieux ce qu'il se passe en sens inverse de la réponse ci-dessus, comme ça vous avez deux raisonnements. C'est un exemple de conversion IEEE802 MAC vers identificateur IPv6 Modifié EUI-64 :

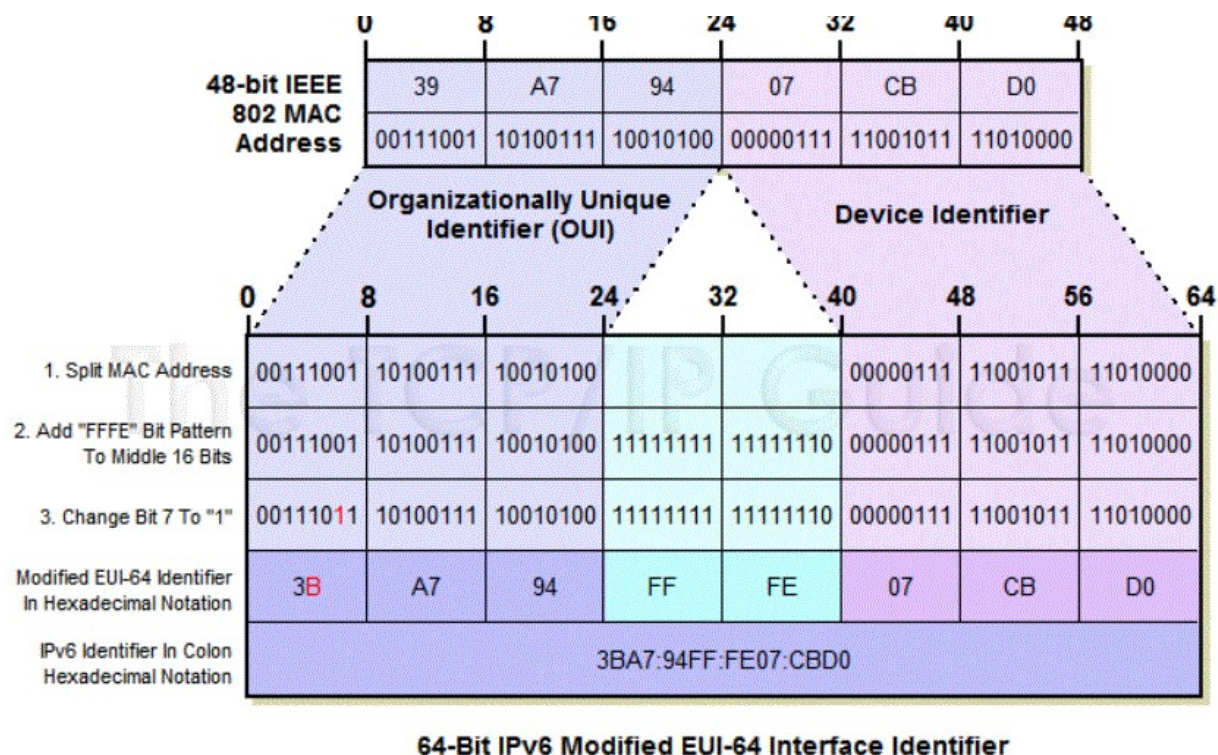


Figure 98: Converting IEEE 802 MAC Addresses To IPv6 Modified EUI-64 Identifiers

Source : http://www.tcpipguide.com/free/t_IPv6InterfaceIdentifiersandPhysicalAddressMapping-2.htm consultée le (12/10/2020).

Question 4 : ICMPV6 avec le service Neighbor Discovery a un rôle équivalent à ARP (Address Resolution Protocol) qui est utilisé en combinaison avec IPv4.

C2:01:02:E4:00

1 0.000000 fe80::c001:2f... ff... IC... 86 Neighbor Solicitation for fe80::c002:3ff:fee4:0 from c2:01:02:40:00:00

Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)

Ethernet II, Src: c2:01:02:40:00:00 (c2:01:02:40:00:00), Dst: IPv6mcast_ff:e4:00:00 (33:33:ff:e4:00:00)

Internet Protocol Version 6, Src: fe80::c001:2ff:fe40:0, Dst: ff02::1:ffe4:0

Internet Control Message Protocol v6

Type: Neighbor Solicitation (135)

Code: 0

Checksum: 0x334f [correct]

[Checksum Status: Good]

Reserved: 00000000

Target Address: fe80::c002:3ff:fee4:0

✓ ICMPv6 Option (Source link-layer address : c2:01:02:40:00:00)

Type: Source link-layer address (1)

Length: 1 (8 bytes)

Link-layer address: c2:01:02:40:00:00 (c2:01:02:40:00:00)

33:33:FF:E4:00

2 0.024297 fe80::c002:3f... fe... IC... 86 Neighbor Advertisement fe80::c002:3ff:fee4:0 (sol, ovr) is at c2:02:03:e4:00:00

Frame 2: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)

Ethernet II, Src: c2:02:03:e4:00:00 (c2:02:03:e4:00:00), Dst: c2:01:02:40:00:00 (c2:01:02:40:00:00)

Internet Protocol Version 6, Src: fe80::c002:3ff:fee4:0, Dst: fe80::c001:2ff:fe40:0

Internet Control Message Protocol v6

Type: Neighbor Advertisement (136)

Code: 0

Checksum: 0x0d2b [correct]

[Checksum Status: Good]

✓ Flags: 0x60000000, Solicited, Override

0 = Router: Not set

..... = Solicited: Set

...1..... = Override: Set

...0 0000 0000 0000 0000 0000 0000 0000 = Reserved: 0

Target Address: fe80::c002:3ff:fee4:0

✓ ICMPv6 Option (Target link-layer address : c2:02:03:e4:00:00)

Type: Target link-layer address (2)

Length: 1 (8 bytes)

Link-layer address: c2:02:03:e4:00:00 (c2:02:03:e4:00:00)

C2:02:03:E4:00

C2:01:02:E4:00

```

3 5.028119  fe80::c002:3f... fe... IC... 86 Neighbor Solicitation for fe80::c001:2ff:fe40:0 from c2:02:03:e4:00:00
Frame 3: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)
Ethernet II, Src: c2:02:03:e4:00:00 (c2:02:03:e4:00:00), Dst: c2:01:02:40:00:00 (c2:01:02:40:00:00)
Internet Protocol Version 6, Src: fe80::c002:3ff:fee4:0, Dst: fe80::c001:2ff:fe40:0
Internet Control Message Protocol v6
  Type: Neighbor Solicitation (135)
  Code: 0
  Checksum: 0x70d0 [correct]
  [Checksum Status: Good]
  Reserved: 00000000
  Target Address: fe80::c001:2ff:fe40:0
  ▾ ICMPv6 Option (Source link-layer address : c2:02:03:e4:00:00)
    Type: Source link-layer address (1)
    Length: 1 (8 bytes)
    Link-layer address: c2:02:03:e4:00:00 (c2:02:03:e4:00:00)

```

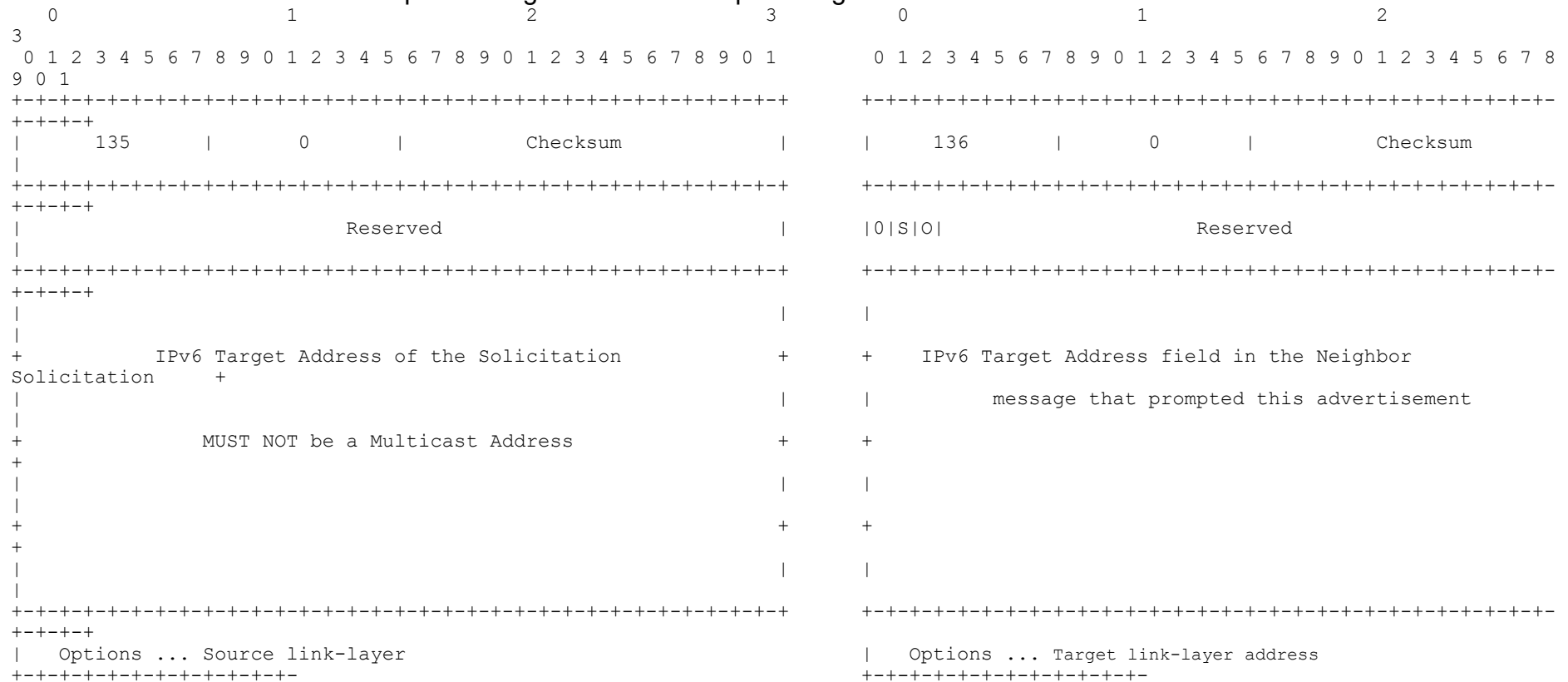
```

4 5.055978  fe80::c001:2f... fe... IC... 78 Neighbor Advertisement fe80::c001:2ff:fe40:0 (sol)
Frame 4: 78 bytes on wire (624 bits), 78 bytes captured (624 bits)
Ethernet II, Src: c2:01:02:40:00:00 (c2:01:02:40:00:00), Dst: c2:02:03:e4:00:00 (c2:02:03:e4:00:00)
Internet Protocol Version 6, Src: fe80::c001:2ff:fe40:0, Dst: fe80::c002:3ff:fee4:0
Internet Control Message Protocol v6
  Type: Neighbor Advertisement (136)
  Code: 0
  Checksum: 0xf6bf [correct]
  [Checksum Status: Good]
  ▾ Flags: 0x40000000, Solicited
    0... .. = Router: Not set
    .1.. .. = Solicited: Set
    ..0. .... = Override: Not set
    ...0 0000 0000 0000 0000 0000 0000 0000 = Reserved: 0
  Target Address: fe80::c001:2ff:fe40:0

```

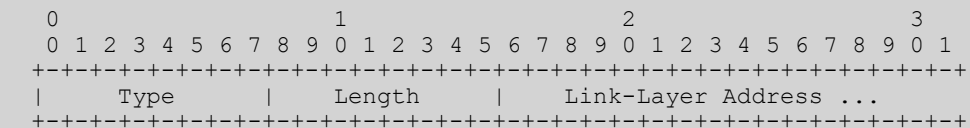
Ci-dessus l'enchaînement de 4 trames qui portent les messages ICMP v6 avec les options liées à la découverte de voisins sur un lien.

On vous donne la structure des options Neighbor Solicitation puis Neighbor Advertisement :



avec le format pour le champ option qui nous intéresse, on vous donne l'extrait :

"4.6.1. Source/Target Link-layer Address



Fields: /* Type; Longueur; Valeur */

Type

1 for Source Link-layer Address

2 for Target Link-layer Address

Length The length of the option (including the type and length fields) in units of 8 octets.
 For example, the length for IEEE 802 addresses is 1

Link-Layer Address The variable length link-layer address.

Description
 The Source Link-Layer Address option contains the link-layer address of the sender of the packet. It is used in the Neighbor Solicitation, Router Solicitation, and Router Advertisement packets.
 The Target Link-Layer Address option contains the link-layer address of the target. It is used in Neighbor Advertisement and Redirect packets."

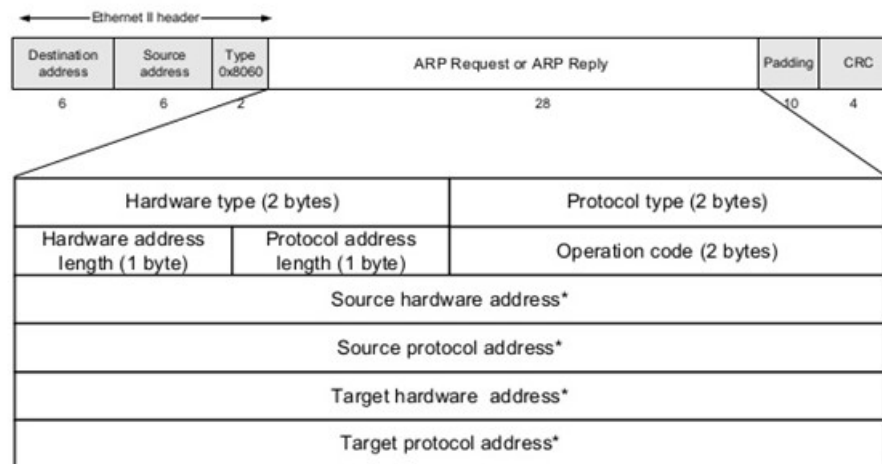
On vous demande d'établir quelques points de comparaison entre les protocoles ARP/IPv4 et ICMPv6-Neighbor Discovery/IPv6.
 On vous demande pour cela de remplir le tableau ci-dessous (0,25 point par case juste) :

	ARP	ICMPv6-Neighbor Discovery
Protocole d'acheminement sous-jacent des messages		
Type d'adresse MAC pour l'envoi de la requête		
Un message est-il bloqué par un routeur ?		
Pourquoi ?		
Un message est-il bloqué par un commutateur ?		
Pourquoi ?		
Charge induite dans la partie information de la trame Ethernet		
Est-il nécessaire de faire du bourrage dans la trame Ethernet		

Correction :

Cette question prend le point de vue de l'encapsulation des couches, et des équipements d'interconnexion, qui se situent donc au niveau 2 (Liaison) et 3 (Réseau) du modèle en couches.

Pour resituer le discours, le schéma ci-après décrit une requête ARP, qui n'existe que dans un univers IPv4 mais qui n'est pas encapsulée dans un datagramme IP et qui donc est incapable de traverser un routeur IPv4 :



	ARP	ICMPv6-Neighbor Discovery
Protocole d'acheminement sous-jacent des messages	Ethernet	IPv6
Type d'adresse MAC pour l'envoi de la requête	Broadcast (à tous)	Multicast (à un groupe)
Un message est-il bloqué par un routeur ?	oui	Dans l'absolu Oui. Mais le TTL(hop Limit dans l'entête IPv6)=255. Cette astuce est fait pour être certain que le datagramme vient bien d'une source locale, et donc n'a traversé aucun routeur. En effet, sinon il ne pourrait pas être à 255 puisqu'il aurait été décrémenté. C'est pour se protéger des attaques de type ICMP spoofing.
Pourquoi ?	On résout sur le LAN et pas plus loin	On résout sur le lien, un LAN par ex, pas plus loin.
Un message est-il bloqué par un commutateur ?	Non	Non a priori
Pourquoi ?	Ca n'est pas géré	Le commutateur, généralement, n'a pas les capacités à gérer des groupes de diffusion multicast MAC dans ses tables de commutation, mais certains équipements savent le faire.
Charge induite dans la partie information de la trame Ethernet	28 octets	40+32 = 72 octets
Est-il nécessaire de faire du bourrage dans la trame Ethernet	Oui, 28 octets plus petit que la taille minimum de la partie charge utile de la trame Ethernet II	Non, elle est supérieure à 46 octets

Question 5 : Dans l'entête IPv6 qui achemine la requête IPv6, le champ associé à la Qualité de Service contient la valeur CS7. Cette valeur est associée à la gestion du réseau et attribue une priorité très élevée au datagramme qui la porte. Sachant que les datagrammes échangés le sont entre deux extrémités en liaison directe, à quoi cette information pourrait-elle servir sur chacun des hôtes ? Pour trouver une réponse, il faut raisonner par rapport à la congestion des files de messages dans la couche IP qui offre un acheminement en mode datagramme donc non fiable. Est-ce qu'on peut associer une qualité de service CS7 pour une requête ou une réponse ARP ? (1 point).

Correction :

La requête ICMPv6 ND (Neighbor Discovery) est une requête de service qui participe à fluidifier la mise en œuvre d'IPv6 sur le lien (connaissance des adresses MAC de voisins). Elle est donc prioritaire. Mais comme elle est échangée sur le lien elle ne traverse pas de routeur.

Toutefois, les "hosts" sur le lien peuvent très bien mettre en œuvre de la gestion de QoS pour éviter les saturations dans leurs files de datagrammes à l'émission ou à la réception. Si tel est le cas, quand elles sont saturées, elles éliminent en fonction de la classe de services les datagrammes dans leurs files. CS7 évite que les requêtes ICMPv6 ND soient éliminées en cas de saturation.

Exercice 6 : Fragmentation IP lors de la traversée d'un routeur.

Question 1

Dans un réseau qu'est ce que la fragmentation ?

Correction :

Tous les types de technologies de liaison sur lesquelles s'appuient les réseaux définissent une taille maximum des messages transportés. En anglais on parle de MTU, Maximum Transfer Unit ou parfois Maximum Transmission Unit. C'est le nombre maximum d'octets de données qui peuvent être transportés dans une seule trame. Le concept de MTU peut s'étendre à d'autres niveaux dans la pile OSI mais pas au-delà de la couche Transport, ça ne dépasse pas la couche réseau généralement.

Le MTU le plus populaire est celui d'Ethernet, il vaut 1500 octets. C'est celui que nous avons vu dans le premier exercice.

Le problème posé est celui des messages trop longs pour être transportés en une seule unité d'information par un réseau de capacité d'acheminement MTU donnée. Ces messages doivent être découpés en unités de taille maximum égale au MTU du réseau utilisé (opération de fragmentation) et le message d'origine doit être reconstruit à partir de ses fragments (réassemblage).

Question 2

Comment fonctionne la fragmentation en IP V4 (expliquez en les principes généraux) ?

Correction :

- **Pourquoi ça marche :** La fragmentation IPV4, est un mécanisme bien identifié dans IPV4 car plusieurs champs y contribuent très explicitement dans l'entête. Ci-dessous le format de l'entête tel que décrit dans la RFC791 (septembre 1981, Jon Postel):

0	1	2	3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1			
+++++	+++++	+++++	+++++
Version	IHL	Type of Service	Total Length
+++++	+++++	+++++	+++++
	Identification	Flags	Fragment Offset
+++++	+++++	+++++	+++++
	Time to Live	Protocol	
+++++	+++++	+++++	+++++
	Source Address		
+++++	+++++	+++++	+++++
	Destination Address		
+++++	+++++	+++++	+++++
	Options		
+++++	+++++	+++++	+++++
			Padding
+++++	+++++	+++++	+++++

Il n'a pas bougé depuis, certains champs, comme ToS, ont évolué, mais c'est marginal. Pour la fragmentation on utilise principalement les champs : Identification, Flags, Fragment Offset, et implicitement Total Length du fait de la nature même de la fragmentation.

- **Identification** : il est sur 16 bits, la valeur choisie par l'émetteur doit être unique, tous les fragments issus d'un même datagramme portent la même valeur dans le champ Identification, il permet donc d'identifier tous les fragments d'un même datagramme source.
- **Flags** : il se découpe en 3 bits mais seulement 2 bits sont utilisés, comme c'est écrit dans la RFC :

Bit 0: reserved, must be zero
 Bit 1: (DF) 0 = May Fragment, 1 = Don't Fragment.
 Bit 2: (MF) 0 = Last Fragment, 1 = More Fragments.

0	1	2
+	+	+
	D	M
	0	F
+	+	+

- Il indique si on peut fragmenter le datagramme (DF oui = 0, non = 1). Si on ne veut pas qu'un datagramme soit fragmenté, on fait DF = 1. Cela générera une erreur dès que le datagramme trouvera un MTU trop petit sur son chemin vers sa destination.
- Si d'autres fragments suivent (MF oui = 1, non = 0, s'interprète comme "encore").
- **Fragment Offset** : Position du fragment par rapport au datagramme fabriqué sur la source.

Fragment Offset: 13 bits

This field indicates where in the datagram this fragment belongs.

The fragment offset is measured in units of 8 octets (64 bits). The



`first fragment has offset zero.`

Cela veut dire que les blocs sont découpés en multiple de 8 octets. Les fragments sont considérés comme une suite de blocs de 8 octets (leur taille est donc un multiple de 8) sauf le dernier fragment. La position d'un fragment par rapport au datagramme d'origine, donc avant le premier envoi et bien sûr avant la première fragmentation, est déterminée par ce nombre, le premier fragment est à la position 0. Ce champ fait une longueur de 13 bits, il peut y avoir 2^{13} , soit 8192 blocs différents.

- **Total Length** : Ce champ ne joue pas un rôle explicite dans la fragmentation, mais il en tient compte. En effet, on ajoute à chaque fragment une entête IP, donc on rajoute 20 octets (en faisant l'hypothèse qu'il n'y a pas d'option). Ceci réduit l'espace disponible dans la partie donnée de la trame et ampute le MTU de 20 octets dont il faut tenir compte lors de la fragmentation.

Avec ces informations on voit qu'on peut fragmenter un datagramme à tout moment : au départ ou dans chaque routeur traversé qui le nécessite.

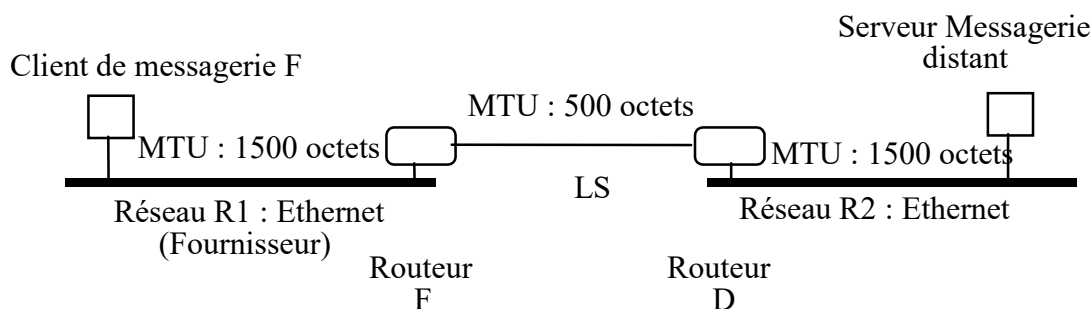
- **Qui fragmente et quand** : La fragmentation IPV4 se produit au fil de l'eau, sur certains routeurs quand le MTU du lien à traverser est trop petit par rapport à la taille du datagramme. On ne réassemble jamais en route seulement à l'arrivée sur la machine destinataire. Tanenbaum baptise cette fragmentation de "fragmentation non transparente".

Chaque interface, ou contrôleur, de communication d'un routeur connaît la taille du MTU de la voie à laquelle il est associé.

Un datagramme pendant son périple vers la destination peut subir plusieurs fragmentations successives. Le réassemblage s'effectue par le destinataire du datagramme. Internet aurait pu choisir une autre stratégie : réassembler une fois la liaison nécessitant la fragmentation traversée. On aurait une fragmentation transparente dans ce cas. Mais serait ce bien efficace ? En effet, un peu plus loin sur le chemin, on pourrait être amené à fragmenter de nouveau... l'approche suivie par IPV4 est plus pragmatique.

Pour ne fragmenter qu'au départ, il faut que la source connaisse le Path MTU. C'est le plus petit MTU entre la source et la destination. C'est possible de le faire en envoyant un datagramme IP avec le bit DF positionné, ICMP (RFC 1191) répond alors par un message d'erreur "type 3 / code 4, "Destination Unreachable" / "Fragmentation Needed and Don't Fragment was Set". On découvre le MTU minimum par ajustements successifs de la taille du datagramme.

Un client de messagerie F transfère un message électronique de 4000 octets de données vers un serveur distant en utilisant trois voies de communication successives selon la figure ci-après. On considère que toutes les entêtes ajoutées par les différentes couches de protocoles traversées au dessus de la couche IP font partie des 4000 octets. Dans les datagrammes IP l'entête est une entête standard de 20 octets (il n'y a pas d'options rajoutées en extensions dans les entêtes IP).



Question 3

Expliquez dans le fonctionnement de la fragmentation et décrivez précisément les entêtes des datagrammes IP échangés. Vous ne décrivez que les champs associés à la fragmentation et la longueur du datagramme ?

Correction :

On voit d'après l'architecture présentée dans la figure ci-dessus qu'on va devoir fragmenter au départ, sur le routeur F et réassembler sur le serveur. Pour être plus précis il faut tenir compte des différents MTU, des entêtes qui se rajoutent, de la taille initiale du datagramme. Regardons étape par étape ce qu'il se passe :

a) A l'origine. Pour la couche IP on a des données de 4000 octets, en fait un segment TCP qui contient le message Mail, les entêtes SMTP (simple Mail Transfer Protocol), et les entêtes TCP. Il faut y ajouter les 20 octets d'entête. On a donc un datagramme d'une taille de 4020 octets. On peut représenter le datagramme d'origine ainsi :

Datagramme fabriqué par IP	LG=4020	ID=X	MF=0	DF=0	Offset = 0	
-------------------------------	---------	------	------	------	------------	--

L'identificateur est attribué, il n'y a pas de fragments (pas encore) donc More Follow (MF) vaut 0, la fragmentation est autorisée Don't Fragment vaut 0 (DF=0). Il n'y a qu'un seul datagramme donc Offset = 0. Ce qui nous fait un datagramme original de 4020 octets avec l'entête IP sachant qu'il n'y a pas d'options.

b) Au départ sur le premier réseau. Cette taille dépasse le MTU de 1500 octets du réseau de départ qui est un Ethernet. On va donc fragmenter en 3 ce datagramme original : 1500 + 1500 + 1000. Ce n'est pas bon en fait car on n'a pas tenu compte des entêtes IP de 20 octets que chaque fragment doit comporter. Il faut découper en $(20_{IP} + 1480_{TCP+Données Mail}) + (20_{IP} + 1480_{TCP+DMail}) + (20_{IP} + 1040_{TCP+DMail})$. La fragmentation du segment TCP à encapsuler dans le datagramme donne :

0	1480	2960	4000
Fragment 1	Fragment 2	Fragment 3	

Il faut transformer cette fragmentation en unité de blocs de 8 octets. C'est comme cela que la couche IP calcule la position par rapport à l'origine de chaque fragment :

Offset 0	185 blocs de 8 octets	370 blocs	
Fragment 1	Fragment 2	Fragment 3	

En effet, 1480 octets divisé par 8 donne 185 blocs de 8 octets, le deuxième fragment démarre à la 185^{ème} position dans le datagramme d'origine. 2960 octets divisé par 8 donne 370 blocs de 8 octets, le Troisième fragment démarre à la 370^{ème} position dans le datagramme d'origine. Ce qui part du Client de Messagerie est donc :

- Le premier fragment a une entête IP qui contient les informations suivantes (celles qui nous intéressent par rapport à la fragmentation) :

Fragment 1		LG=1500	ID=X	MF=1	DF=0	Offset = 0	
------------	--	---------	------	------	------	------------	--

Puisque le datagramme est fragmenté : MF = 1, c'est le premier fragment donc Offset = 0 comme nous l'avons dit plus haut. Le MTU de la trame est complètement occupé, le datagramme a une longueur maximale de 1500 octets (LG)

- On raisonne de la même façon pour le fragment 2 :

Fragment 2		LG=1500	ID=X	MF=1	DF=0	Offset = 185	
------------	--	---------	------	------	------	--------------	--

Cette fois l'Offset = 185

- Et pour le fragment 3 :

Fragment 3		LG=1060	ID=X	MF=0	DF=0	Offset = 370	
------------	--	---------	------	------	------	--------------	--

Cette fois l'Offset = 370, il n'y a plus d'autre fragment à suivre donc MF = 0, et ce datagramme est plus petit donc LG = 1060.

c) Sur le routeur F. Le lien suivant a un MTU de 500 octets. Il est plus petit que le précédent. On va devoir fragmenter à nouveau sur le routeur F.

Un fragment de 1480 octets va être découpé en 3 : un sous-fragment de 480 octets, deux autres de 480 octets, et un dernier de 40 octets sans compter les entêtes IP qu'il faut rajouter ensuite quand on fabrique le datagramme prêt à l'envoi. 480 divisé par 8 donne 60 blocs de 8 octets. Les sous-fragments vont être positionnés à 0, 60, 120 et 180. Le dernier sous-fragment sera un peu plus court que les autres car il ne contient que 40 octets du datagramme d'origine contenant le segment TCP à acheminer.

- Ceci donne après fragmentation du fragment 1 :

Fragment 1.1		LG=500	ID=X	MF=1	DF=0	Offset = 0	
--------------	--	--------	------	------	------	------------	--

Fragment 1.2		LG=500	ID=X	MF=1	DF=0	Offset = 60	
--------------	--	--------	------	------	------	-------------	--

Fragment 1.3		LG=500	ID=X	MF=1	DF=0	Offset = 120	
--------------	--	--------	------	------	------	--------------	--

Fragment 1.4		LG=60	ID=X	MF=1	DF=0	Offset = 180	
--------------	--	-------	------	------	------	-----------------	--

- Pour la fragmentation du fragment 1 cela donne presque la même chose, le premier sous-fragment démarre à un Offset de 185, ceci n'a pas changé :

Fragment 2.1		LG=500	ID=X	MF=1	DF=0	Offset = 185	
--------------	--	--------	------	------	------	-----------------	--

Fragment 2.2		LG=500	ID=X	MF=1	DF=0	Offset = 245	
--------------	--	--------	------	------	------	-----------------	--

Fragment 2.3		LG=500	ID=X	MF=1	DF=0	Offset = 305	
--------------	--	--------	------	------	------	-----------------	--

Fragment 2.4		LG=60	ID=X	MF=1	DF=0	Offset = 365	
--------------	--	-------	------	------	------	-----------------	--

- Pour le fragment 3, on procède de la même manière. Mais on va générer 3 sous-fragments : deux de 480 octets et un dernier de 80 octets. L'Offset du premier sous-fragment est identique à celui du fragment 3, pour les autres on procède par pas de 60 blocs. On obtient :

Fragment 3.1		LG=500	ID=X	MF=1	DF=0	Offset = 370	
--------------	--	--------	------	------	------	-----------------	--

Fragment 3.2		LG=500	ID=X	MF=1	DF=0	Offset = 430	
--------------	--	--------	------	------	------	-----------------	--

Fragment 3.3		LG=100	ID=X	MF=0	DF=0	Offset = 490	
--------------	--	--------	------	------	------	-----------------	--

Pour le fragment 3.3, MF=0, c'est le dernier fragment du datagramme d'origine.

d) Sur le routeur D. Rien ne se passe les fragments poursuivent leur chemin, IP applique une fragmentation non transparente. C'est-à-dire qu'il ne réassemble pas en cours de route. L'assemblage complet est laissé au destinataire final.

e) Sur le serveur de Messagerie. La couche IP du serveur de messagerie réassemble tous les fragments du datagramme en un seul morceau. Et livre un segment TCP complet à la couche supérieure. S'il manque un fragment, le datagramme est éliminé.

Question 4

Pourquoi la fragmentation est incompatible avec des flux de données nécessitant une QoS temps réel ? Quels paramètres de QoS : latence, gigue, débit, taux de perte vont être

influencés par la fragmentation ?

Correction :

On voit que la fragmentation introduit des temps de traitement quand on traverse un routeur qui doit l'appliquer. Ce temps de traitement ajoute du délai à la transmission donc augmente le paramètre de QoS Latence.

Les fragments vont aussi peupler les files de messages/mémoires tampons/buffers des interfaces de sorties liaison ou du routeur. Donc ils vont provoquer plus facilement de la congestion, qui a pour conséquence éventuelle des suppressions de fragments. Potentiellement cela risque d'augmenter le Taux de perte.

Si on suppose que le trafic est mixte, c'est-à-dire qu'il y a des petits datagrammes (qui ne seront pas fragmentés) et d'autres plus gros (qui le seront), on peut considérer la fragmentation comme provoquant un effet de gigue. Le paramètre de QoS Gigue d'un flux peut donc aussi être impacté.