Couche Réseau

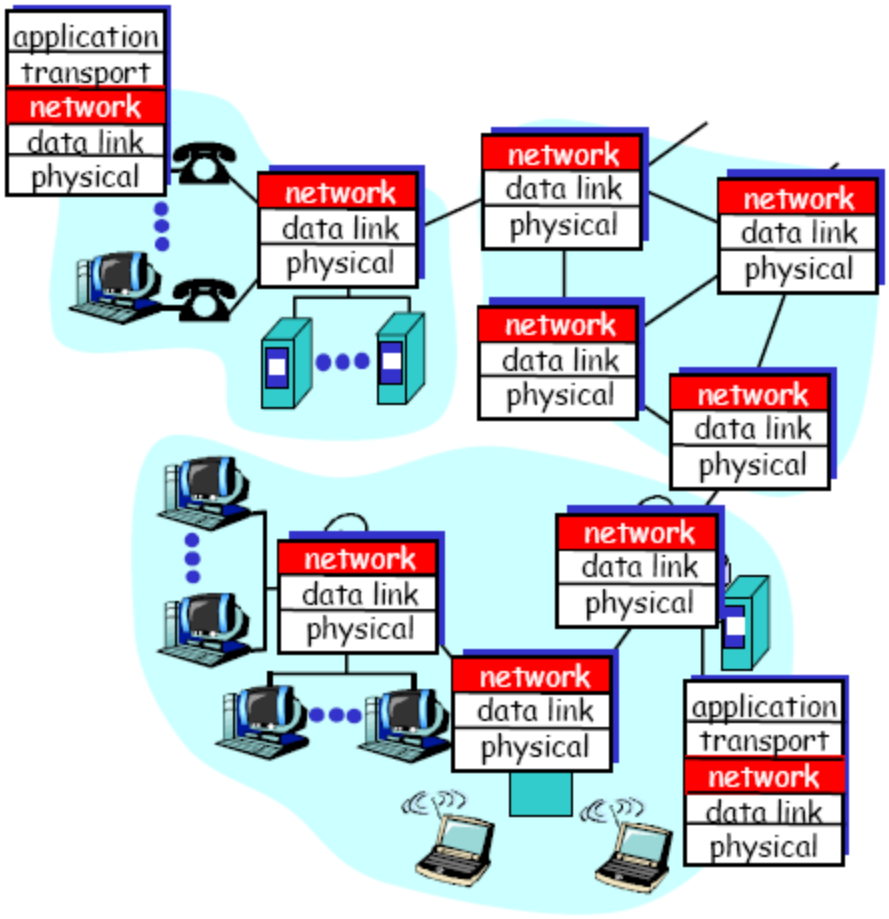
Prof. Rami Langar
LIGM/UPEM
Rami.Langar@u-pem.fr
http://perso.u-pem.fr/~langar

Plan

- Fonctionnalités de la couche réseau
- Modèle de service : Circuit virtuel/Datagramme
- Service IP
- Datagramme IP
- Adressage IP et CIDR
- Network Address Translation: NAT
- Protocole ICMP
- Quelques exemples: Ping et Traceroute
- Algorithmes de routage:
  - Vecteur de distance
  - Etat de liens

Fonctionnalités de la couche réseau

- Transporter des paquets de l'émetteur vers le récepteur
- Les protocoles de couche réseau s'exécutent dans chaque hôte et routeur.
- Trois fonctions principales :
  - Choix du chemin : route suivie par les paquets de la source à la dest. Algorithmes de routage
  - Commutation : transporter les paquets du port d'entrée vers le bon port de sortie.
  - Mise en place de l'appel : Dans les réseaux à commutation de circuits, la mise en place du circuit est effectuée par la couche réseau.



Service IP

- Utilise un service minimum
  - envoi d'une unité de transfert d'un point à ses voisins
  - voisin : partage la même connexion physique
- Rend un service minimum
  - service en mode non connecté
    - absence d'états dans les routeurs
    - transmission de datagrammes
  - remise *best effort*
    - service non fiable
  - service de connectivité
- Avantages
  - ☺ robustesse
  - ☺ efficace pour les échanges brefs
  - ☺ simplicité d'utilisation

Que fait un routeur ?

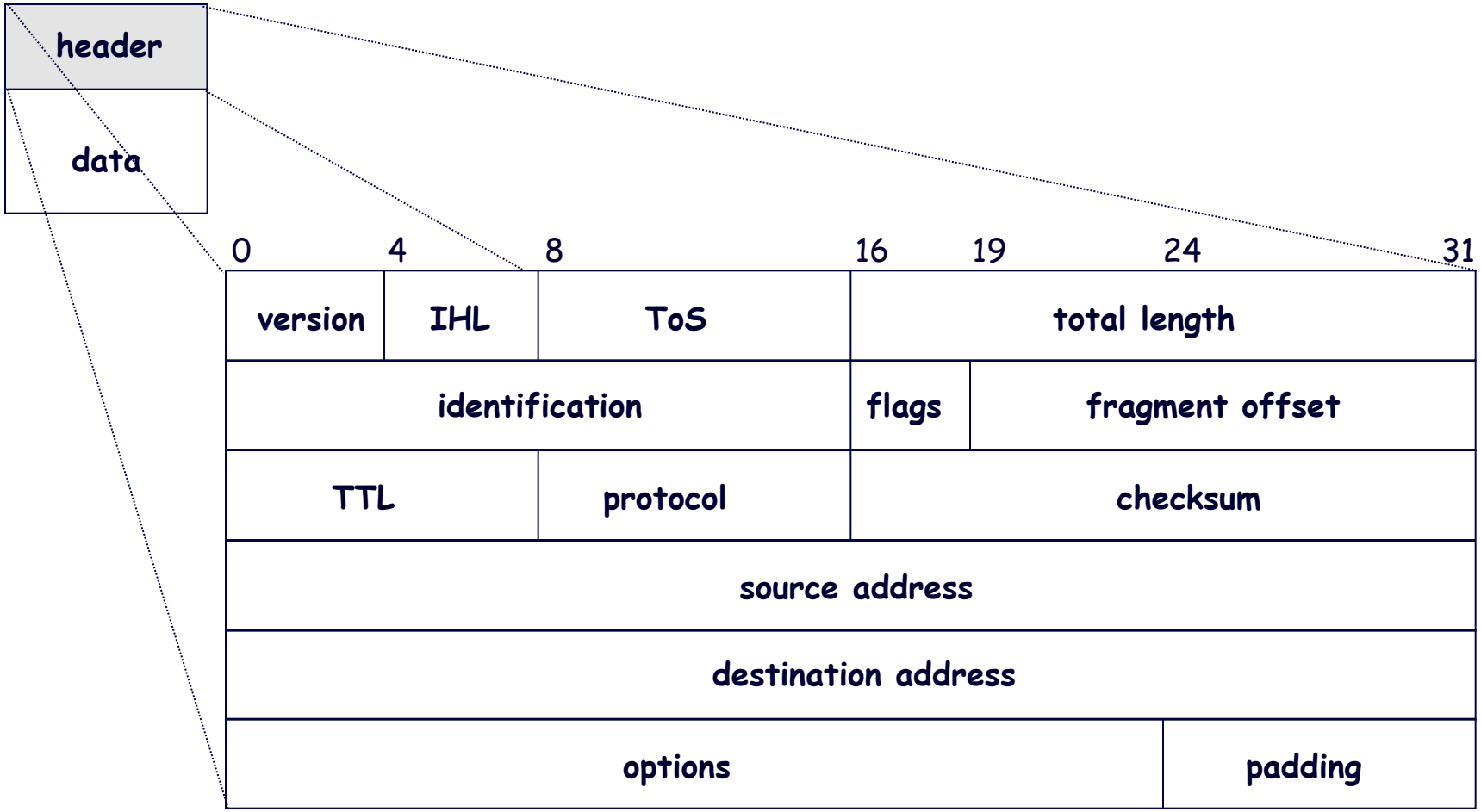
Pour chaque datagramme IP qui traverse le routeur, IP :

- vérifie le checksum, si faux ➔destruction du datagramme
- détermine si ce sont des données *utilisateur* ou de *contrôle* destinées au routeur
- décrémente la durée de vie, si nulle ➔ destruction du datagramme
- **décide du routage**
- **fragmente** le datagramme si nécessaire
- **reconstruit l'en-tête IP** avec les champs mis à jour
- transmet le(s) datagramme(s) au protocole d'accès de l'interface réseau de sortie avec **l'adresse de sous-réseau correspondante**

A réception dans l'hôte destinataire, IP :

- vérifie le checksum
- s'il y a eu fragmentation, mémorise puis **réassemble**
- **délivre au niveau supérieur** les données

Le datagramme IP

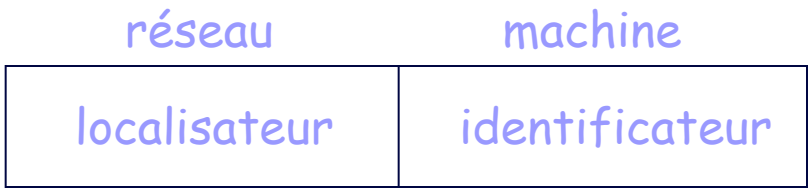


Les champs de l'en-tête IP

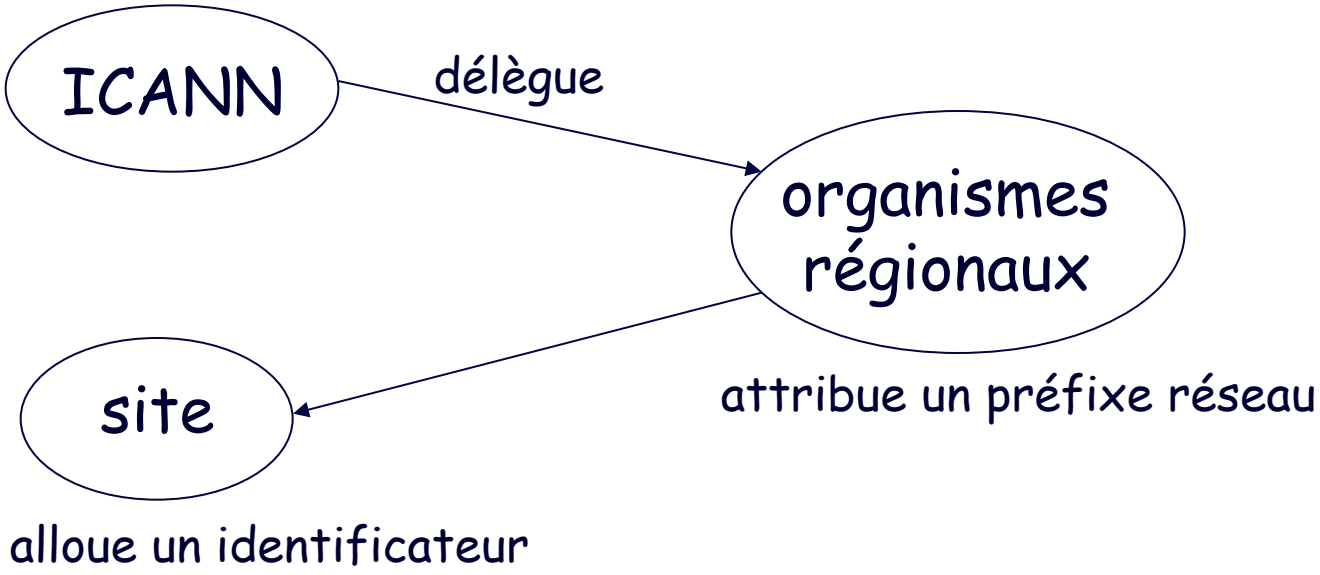
- version : identification de la version courante du protocole (4 pour IPv4)
- IHL (*IP Header Length*) : longueur de l'en-tête IP (en mots de 32 bits)
- TOS (*Type Of Service*) : type de service à appliquer au paquet en fonction de certains paramètres comme le délai de transit, la sécurité
- total length : longueur totale du datagramme (en octets)
- identification : valeur fournie par la source aidant la destination au réassemblage des différents fragments du datagramme
- flags : utilisé par la fragmentation et composé de
  - DF (*Don't Fragment*)
  - MF (*More Fragment*)
  - réservé
- offset : déplacement par rapport au datagramme initial (en multiple de 8 octets)
- TTL (*Time To Live*) : limite supérieure du temps de vie d'un datagramme
- protocol : protocole utilisé pour le champ de données
  - 1 pour ICMP
  - 6 pour TCP
  - 17 pour UDP
- checksum : zone de contrôle d'erreur portant uniquement sur l'en-tête du datagramme
- source address : @ IP de la source du datagramme
- destination address : @ IP de la destination du datagramme
- options : fonctions de contrôle utiles dans certaines situations (estampillage temporel, sécurité, routage particulier, etc.)
- padding : pour aligner l'en-tête sur 32 bits

Adressage IP

- adressage
  - pour l'identification d'un équipement réseau
  - pour le routage
- plan d'adressage homogène
  - format : 4 octets ➔ 4,3 milliards d'adresses ???
  - notation décimale pointée : x1.x2.x3.x4
- adresse globalement unique et hiérarchique
- format : <réseau> <machine>
  - localisateur ou préfixe réseau : identificateur de réseau
  - identificateur : identificateur de machine



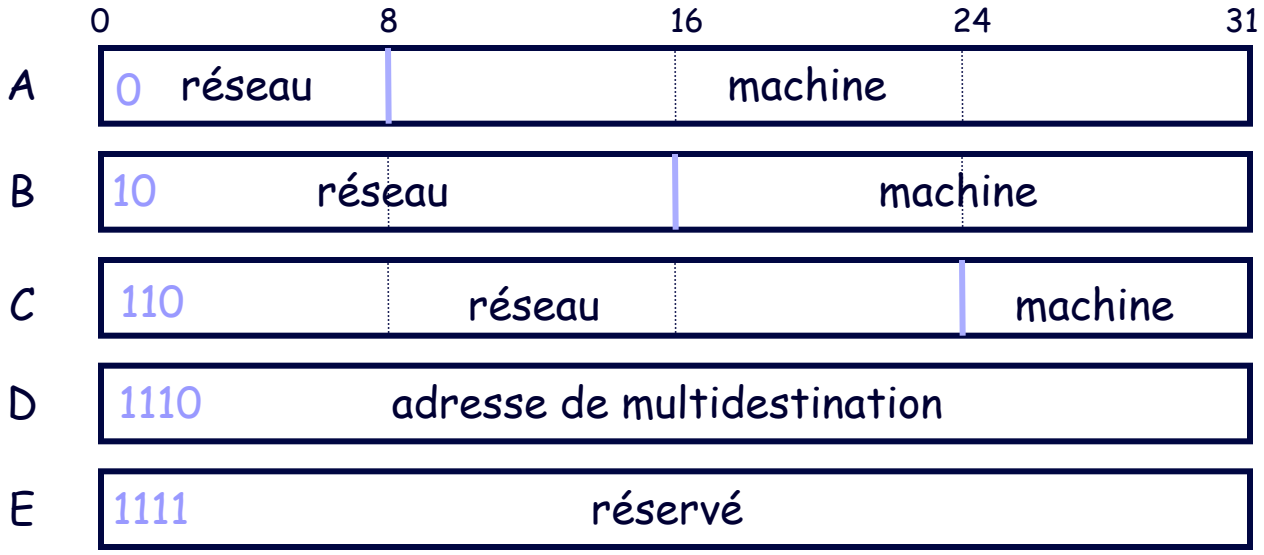
Attribution des adresses



ICANN : Internet Corporation for Assigned Names and Numbers

Classes d'adresses

- le découpage <réseau> / <machine> n'est pas fixe
- 5 classes d'adresses

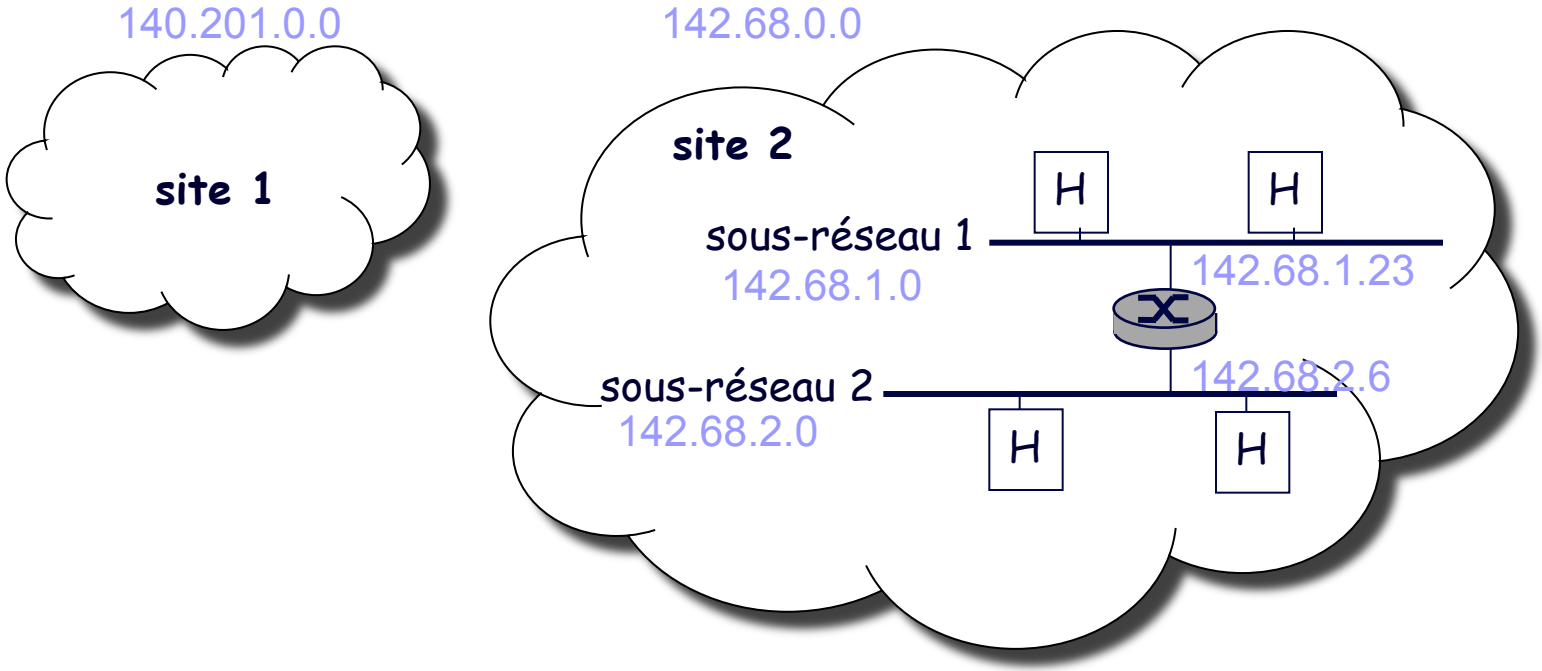


Classes d'adresses

- classe A : 2^7 réseaux (128)
  - réservé: 0.0.0.0 et 127.0.0.0
  - disponible: 1.0.0.0 à 126.0.0.0
    - 126 réseaux classe A et 16 777 214 machines/réseau
- classe B : 2^{14} réseaux (16 384)
  - réservé: 128.0.0.0 et 191.255.0.0
  - disponible 128.1.0.0 à 191.254.0.0
    - 16 382 réseaux classe B et 65 534 machines/réseau
- classe C : 2^{21} réseaux (2 097 152)
  - réservé 192.0.0.0 et 223.255.255.0
  - disponible 192.0.1.0 à 223.255.254.0
    - 2 097 150 réseaux classe C et 254 machines/réseau

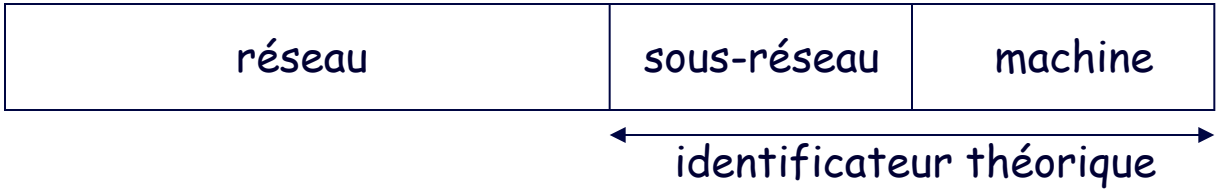
Subnetting

- Problème
  - distinction <réseau> / <hôte> insuffisante en pratique



Sous-adressage

- Principe
  - ajout d'un niveau hiérarchique dans l'adressage
    - adresse de sous-réseau
  - subdivision de la partie <hôte>



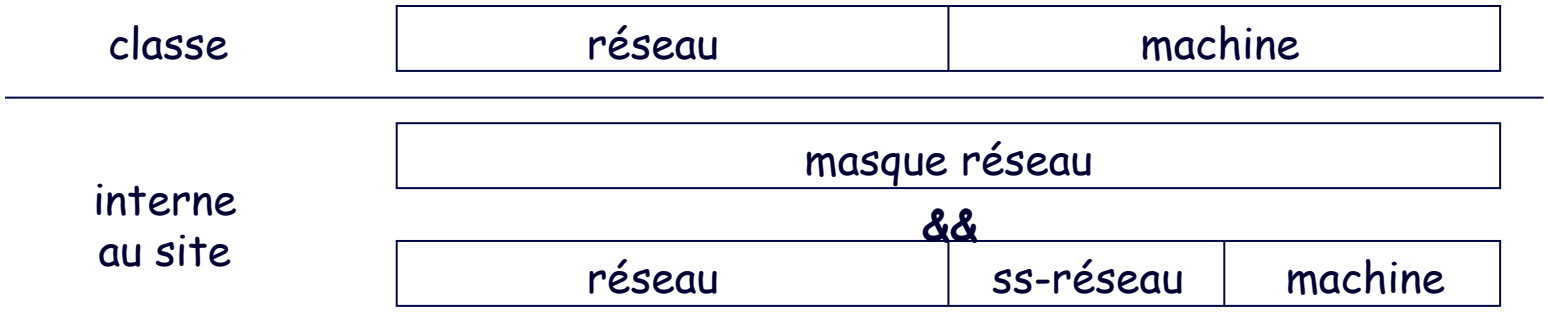
- le sous-réseau
  - est un réseau physique (i.e. un réseau IP connexe) du réseau de site
  - a une visibilité purement interne (transparent vis à vis de l'extérieur).

Le masque de sous-réseau

- Le masque indique la frontière entre la partie <sous-réseau> et la partie <machine>
- Le masque est propre au site et il est de 32 bits
- Bits du masque de sous-réseau (*subnet mask*)
  - positionnés à 1 ➔ partie réseau
  - positionnés à 0 ➔ partie machine
- Exemple
  - `11111111 11111111 11111111 00000000`
  - 3 octets pour le champ réseau, 1 octet pour le champ machine
- Notations
  - décimale pointée
    - exemple : 255.255.255.0
  - adresse réseau/masque
    - exemple : 193.49.60.0/27 (27 = nombre de bits contigus du masque)

Masque de sous-réseau

- Utilisation :



- Exemple :
  - le réseau 142.68.0.0 (classe B!) a comme masque 255.255.255.0
  - soit l'hôte d'@IP 142.68.2.6

	10001110.01000100.00000010.00000110	142.68.2.6
&&	11111111.11111111.11111111.00000000	255.255.255.0
=	10001110.01000100.00000010.00000000	142.68.2.0

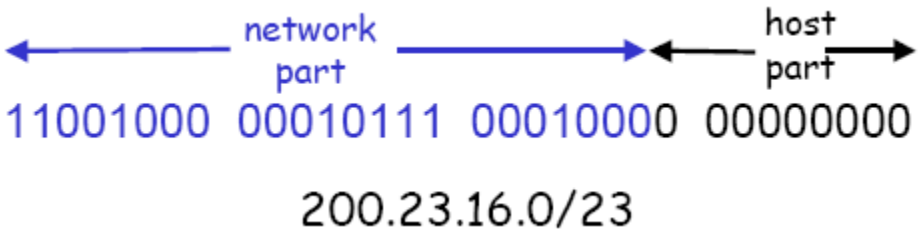
  - l'hôte est sur le sous-réseau numéro 2, et a comme identificateur 6
  - Le netmask permet de savoir si la machine source et destination sont sur le même sous-réseau.

Le masque de sous-réseau

- Le choix du découpage <réseau> / <hôte> dépend des perspectives d'évolution du site
  - exemple classe B :
    - 8 bits pour la partie sous réseau ➔ 256 sous réseaux de 254 machines
    - 3 bits pour la partie sous réseau ➔ 8 sous-réseaux de 8190 machines
  - exemple classe C :
    - 4 bits pour la partie sous-réseau ➔ 16 sous-réseaux de 14 machines

Adresse IP : CIDR

- Adressage par classe :
  - utilisation inefficace de l'espace d'adressage
  - Ex : une adresse de classe B a assez de place pour pour 65K hôtes, même si il n'y a que 2K hôtes dans ce réseau
- CIDR : Classless InterDomain Routing
  - La taille de la partie réseau est arbitraire
  - Format de l'adresse : a.b.c.d/x, où x est le # de bits dans la partie réseau de l'adresse
  - Ex: 128.96.0.0/16 : regroupe les numéros de 128.96.0.0 à 128.96.255.255
    => équivalent d'une classe B en notation classique



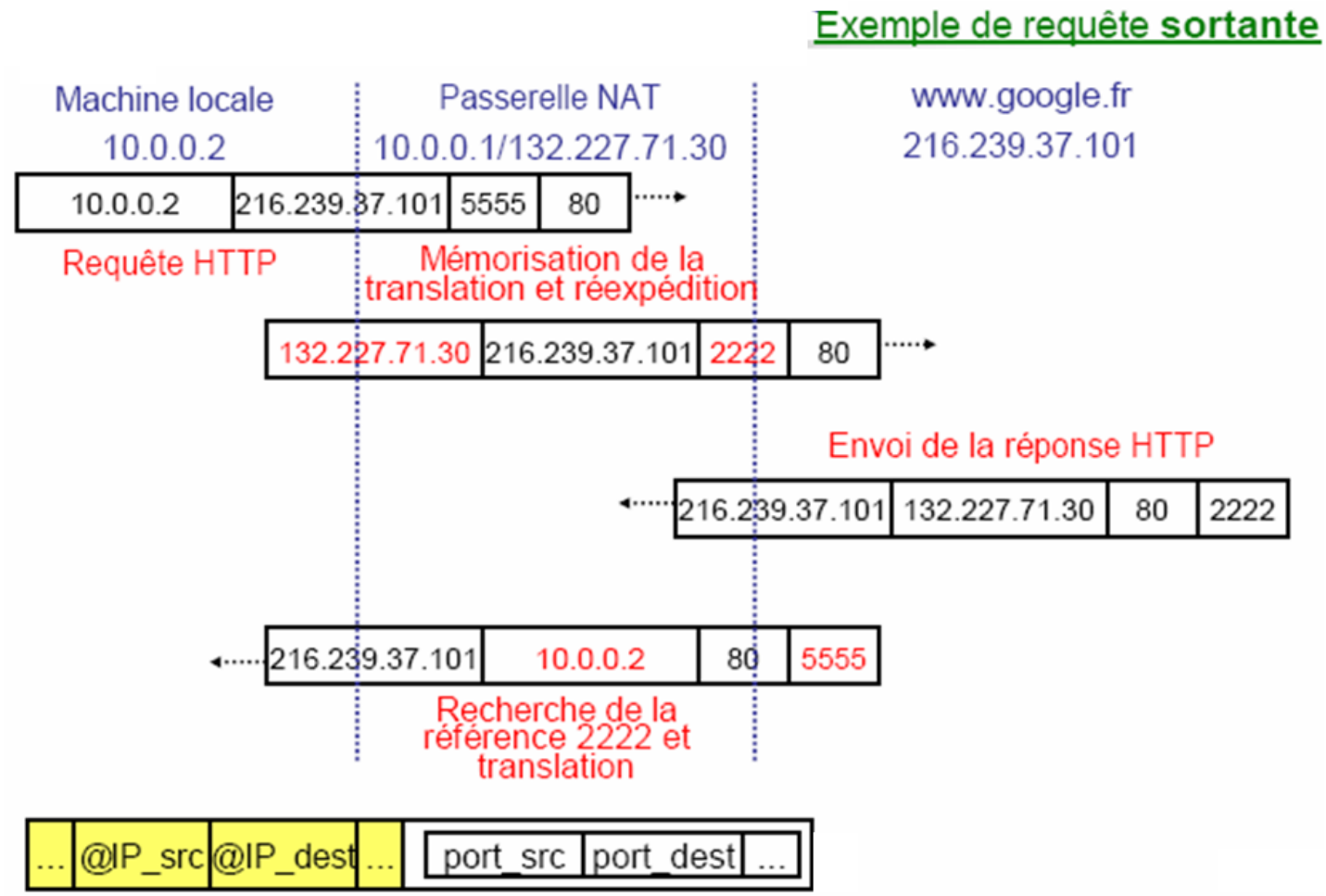
Les adresses privées et le NAT

- Adresses privées (RFC 1918)
  - des adresses qui ne seront jamais attribuées (adresses illégales) et qui ne sont pas routables sur l'Internet
  - classe A : de 10.0.0.0 à 10.255.255.255
  - classe B : de 172.16.0.0 à 172.31.255.255
  - classe C : de 192.168.0.0 à 192.168.255.255
- Si une entreprise qui utilise des adresses privées souhaitent tout de même disposer d'une connexion à l'Internet, il faut
  - demander une adresse publique
  - faire des conversions adresse privée <--> adresse publique

Les adresses privées et le NAT

- NAT (RFC 3022) - Network Address Translator
  - mise en correspondance d'une adresse privée et d'une adresse publique
  - traduction statique ou dynamique (lors de la connexion)
  - une solution au manque d'adresses IP publiques : quelques adresses IP publiques pour beaucoup d'adresses IP privées mais le NAT est coûteux en perf.
- Fonctionnement du NAT
  - une table stockée dans le NAT fait la correspondance entre (@IP_src privée, port_src) et une @IP_publique
  - quand le paquet part : @IP_src devient @IP_publique, port_src devient la référence de l'entrée dans la table
  - quand la réponse revient : port_dest du paquet permet de retrouver dans la table @IP et port src

NAT


Exemple de requête sortante
Machine locale
10.0.0.2
Passerelle NAT
10.0.0.1/132.227.71.30
www.google.fr
216.239.37.101
10.0.0.2
216.239.37.101
5555
80
Requête HTTP
Mémorisation de la translation et réexpédition
132.227.71.30
216.239.37.101
2222
80
Envoi de la réponse HTTP
216.239.37.101
132.227.71.30
80
2222
216.239.37.101
10.0.0.2
80
5555
Recherche de la référence 2222 et translation
...
@IP_src
@IP_dest
...
port_src
port_dest
...

Routage

- fonction déterminant un chemin vers une adresse destinataire
- table de routage
  - informations nécessaires pour atteindre le prochain nœud
  - Contient 3 informations:
    - Destination, Chemin, cout
- algorithme de routage
  - calcul d'un chemin optimal pour atteindre une adresse destinataire
- protocole de routage
  - échange d'informations de routage
  - dépend du domaine dans lequel se trouve le routeur
  - ex : RIP, OSPF, …

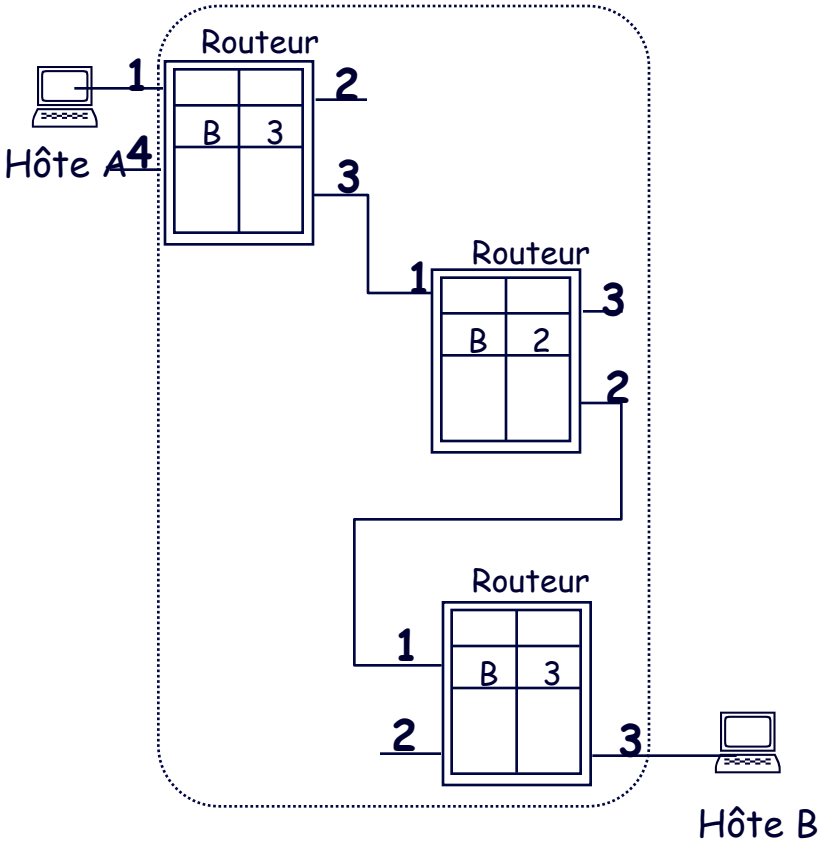
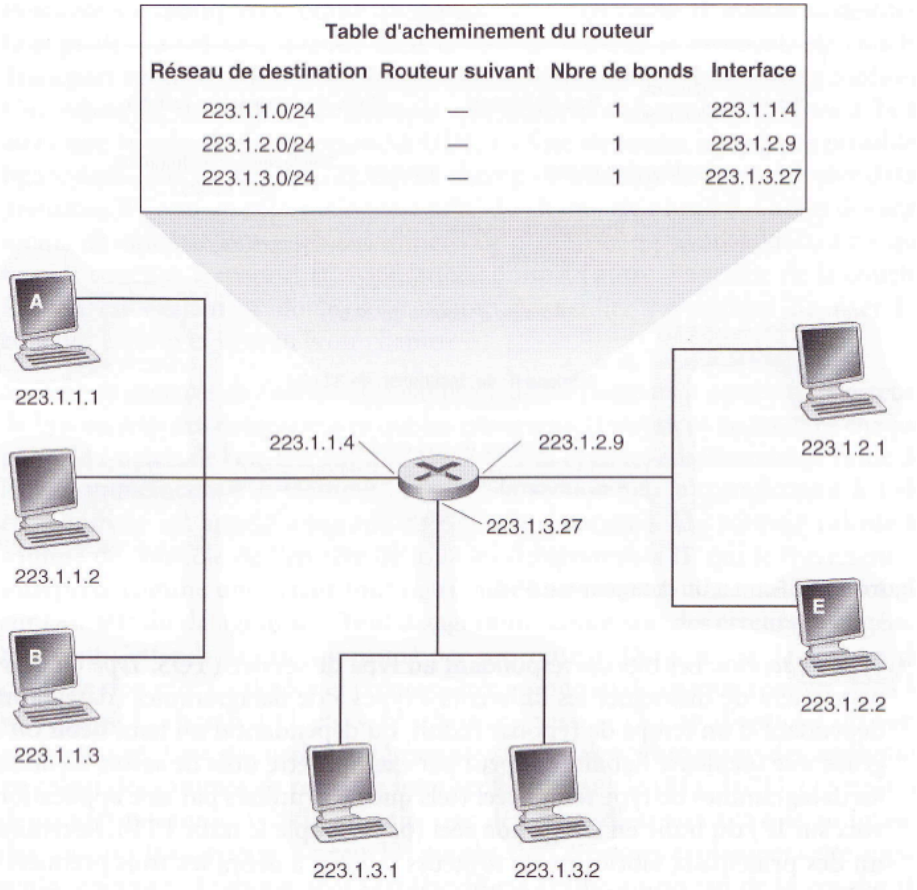


Table de routage



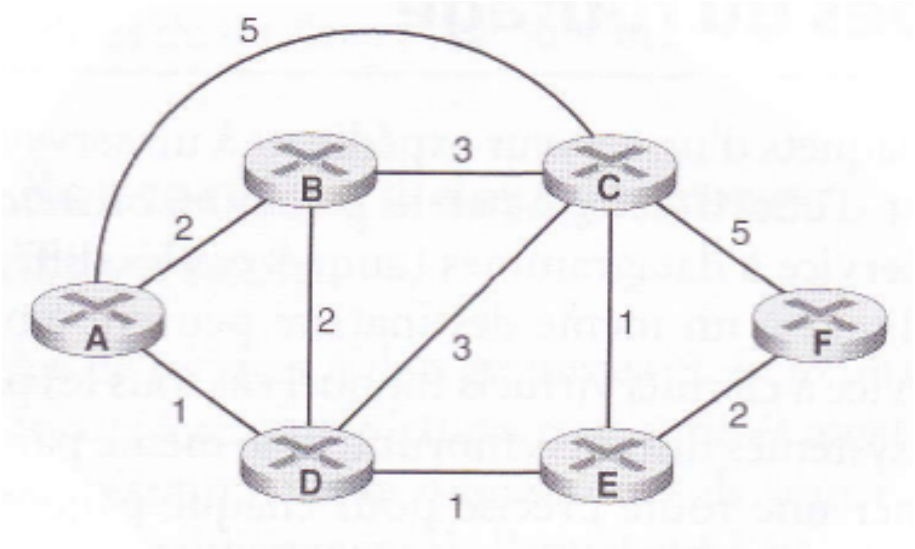
Protocole et algorithmes de routage

- Deux classes de protocole de routage
  - Les protocoles de routage Intra-domaine
  - Les protocoles de routage Inter-domaine

- Deux types d'algorithme de routage
  - État de lien
  - Vecteur de distance

Modèle théorique de réseau

- Un réseau est représenté par un graphe
  - Les nœuds représentent les routeurs
  - Les liens représentent les liaisons
- Un algorithme de routage consiste à identifier le parcours entre un expéditeur et un destinataire dont la somme des liaisons est la plus faible



Algorithme de routage par état de lien

- Chaque nœud du réseau diffuse l'état des liaisons auxquelles il est rattaché
- Avec les informations sur les états des liens venant des autres nœuds dans le réseau, chaque nœud possède une connaissance sur la topologie complète du réseau
- Chaque nœud applique l'algorithme de Dijkstra sur cette topologie pour trouver les chemins les plus courts vers chaque destination

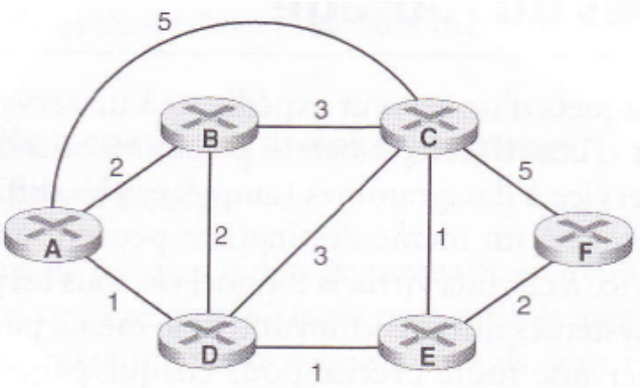
Algorithme de Dijkstra (1)

- Pour un nœud source A
  - C(i,j) : coût du lien direct (i,j)
  - D(v) : coût du chemin du nœud source au destinataire v
  - P(v) : Nœud précédant du destinataire v
  - N : groupe du nœud source et des nœuds destinataires dont les chemins les plus courts sont définitivement trouvés

Algorithme de Dijkstra (2)

```
1 Initialisation:
2    N = {A}
3    pour tous les noeuds v
4      si v adjacent à A
5        alors D(v) = c(A,v)
6        ou bien D(v) = 8
7
8 Effectuer une boucle
9    trouver w à l'extérieur de N tel que D(w) soit minimum
10    ajouter w à N
11    mettre à jour D(v) pour tout v adjacent à w et à l'extérieur de N:
12      D(v) = min( D(v), D(w) + c(w,v) )
13    /* le nouveau coût du parcours vers new v est soit l'ancien coût,
         ➥soit
14      celui calculé vers w additionné du coût de w à v */
15 jusqu'à tous les noeuds de N
```

Algorithme de Dijkstra (3)



Étape	N	D(B), p(B)	D(C), p(C)	D(D), p(D)	D(E), p(E)	D(F), p(F)
0	A	2,A	5,A	1,A	∞	∞
1	AD	2,A	4,D		2,D	∞
2	ADE	2,A	3,E			4,E
3	ADEB		3,E			4,E
4	ADEBC					4,E
5	ADEBCF					

Afficher la table de routage

- Windows
  - C> route PRINT

```
IPv4 Table de routage
===========================================================================
Itinéraires actifs :
Destination réseau    Masque réseau  Adr. passerelle   Adr. interface Métrique
          0.0.0.0          0.0.0.0    192.168.0.254     192.168.0.4     25
        127.0.0.0        255.0.0.0          On-link       127.0.0.1    306
        127.0.0.1  255.255.255.255          On-link       127.0.0.1    306
  127.255.255.255  255.255.255.255          On-link       127.0.0.1    306
      192.168.0.0    255.255.255.0          On-link     192.168.0.4    281
      192.168.0.4  255.255.255.255          On-link     192.168.0.4    281
    192.168.0.255  255.255.255.255          On-link     192.168.0.4    281
        224.0.0.0        240.0.0.0          On-link       127.0.0.1    306
        224.0.0.0        240.0.0.0          On-link     192.168.0.4    281
  255.255.255.255  255.255.255.255          On-link       127.0.0.1    306
  255.255.255.255  255.255.255.255          On-link     192.168.0.4    281
===========================================================================
```

  - C> ipconfig /all
    - Voir la passerelle par défaut

Utilitaires TCP/IP

- host
  - Interrogation dns
  - host [-l][-v][-w][-r][-d][-t types] [-a] machine [serveur]
  - Machine = @IP ou nom + domaine local (extrait via hostname)
  - Serveur : nom ou @IP d'un serveur DNS spécifique
  - -v: mode verbeux
  - -r: supprime la recherche récursive
  - -l: liste d'un domaine complet
  - -t: précise un type d'enregistrement (filtre sur recherche sur l'ensemble d'un domaine) -t mx

Utilitaires TCP/IP

- [langar@localhost ~]$ host www.lpi.org
  - www.lpi.org has address 24.215.7.162
- [langar@localhost ~]$ host www.ipsl.jussieu.fr
  - www.ipsl.jussieu.fr is an alias for weberie.ipsl.jussieu.fr.
  - weberie.ipsl.jussieu.fr has address 192.168.1.56

Configuration réseaux

- Configurer le réseau sur un poste Linux c'est définir :
  - une adresse IP + un masque de réseau + une adresse de diffusion
  - la route par défaut
  - la méthode de résolution de noms (serveur DNS)
  - un nom de machine
- Nommage des cartes réseau sous Linux
  - eth0, eth1, eth2,...
  - Commande dmesg renseigne entre autres sur les interfaces présentes et leur alias (ethx)
  - Fichiers de configuration

    /etc/sysconfig/network-scripts/ifcfg-eth0

Configuration réseaux

- /sbin/ifconfig
  - Utilitaire de configuration des interfaces réseau
  - ifconfig -a : affiche l'état de la configuration de toutes les interfaces

```
# ifconfig -a
eth0      Lien encap:Ethernet  HWaddr 00:A0:C9:DD:F2:B3 (1)
          inet adr:134.157.45.218 Bcast:134.157.45.255
Masque:255.255.255.128 (2)
          adr inet6: fe80::2a0:c9ff:fedd:f2b3/64 Scope:Lien
        (3)UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        (4)RX packets:2282 errors:0 dropped:0 overruns:0 frame:0
           TX packets:67 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 lg file transmission:1000
           RX bytes:197746 (193.1 Kb)  TX bytes:6357 (6.2 Kb)
```

Configuration réseaux

- Modifier dynamiquement sa configuration
  - Commande ifconfig

```
$ /sbin/ifconfig eth0 192.168.45.1 netmask 255.255.255.0
broadcast 192.168.1.255 up
```

(1) eth0 : nom de l'interface
(2)192.168.45.1 : adresse IP de l'hôte
(3) netmask 255.255.255.0 : masque de réseau
(4) broadcast 192.168.45.255 : adresse de diffusion du réseau
(5) up : active l'interface

- Démarrer manuellement une interface
  - Commande ifup interface
- Stopper manuellement une interface
  - Commande ifdown interface

Configuration réseaux

- netstat
  - Commande très utilisée qui affiche les connexions réseau, les tables de routage, les statistiques d'interface, les connexions masquées, les messages netlink, les membres multicast
  - - i : liste des interfaces
  - - n : mode numérique. Affiche les adresses plutôt que des noms
  - - a : affiche également les sockets d'écoute des serveurs
  - - p : affiche les processus réseau (PID, user)
  - - r : affiche les informations de route
  - - v : verbose
- Gestion des routes
  - route : affiche, ajoute et supprime les routes de la table de routage locale
  - route add [options] cible
  - route del [options] cible

Configuration réseaux

- La table de routage locale indique le chemin que doit emprunter le trafic hors réseau local
  - L'utilitaire route permet de contrôler et de modifier les routes locales
  - **#route add default gw <@ de la passerelle>**
  - **#route del default**
  - **#route** ou **#netstat -rn** pour afficher les routes définies

```
# netstat -rn
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic   MSS Fenêtre
irtt Iface
134.157.45.128  0.0.0.0         255.255.255.128 U         0 0
 0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U         0 0
 0 eth0
0.0.0.0         134.157.45.254  0.0.0.0         UG        0 0
 0 eth0
```

UG: identification de la passerelle

Protocole ICMP

- Internet Control Message Protocol), RFC 792
- Objectif
  - Rapport d'erreurs de la couche Réseau
- Exemple
  - Au cours d'une session HTTP, un message d'erreur « Réseau de destination inaccessible » est affiché
  - Quand un routeur ne peut pas trouver le chemin spécifié par l'application, il génère un message d'erreur à la source
  - Le terminal reçoit le message ICMP et renvoie le code d'erreur à la couche TCP qui à son tour le renvoie à l'application

Protocole ICMP

- Basé sur IP, Protocol ID = 1
- Chaque message ICMP est doté d'un champ de code et de catégorie qui signifie le contenu du message
  - Exemples
    - Type = 0, code = 0 : message d'écho
    - Type = 8, code = 0 : demande d'écho
    - Type = 11, code = 0 : expiration de la durée de vie

Ping

- Ping
  - Le programme *ping* envoie à une destination un message ICMP avec type = 8, code = 0 (demande d'écho)
  - La machine destination répond par un message d'écho ICMP (type = 0, code = 0)
  - Le programme affiche le délai du parcours à l'utilisateur

```
C:\Users\Univ P & M Curie>ping www.lip6.fr

Envoi d'une requête 'ping' sur w.lip6.fr [132.227.73.20] avec 32 octets de donné
es :
Réponse de 132.227.73.20 : octets=32 temps=22 ms TTL=53
Réponse de 132.227.73.20 : octets=32 temps=22 ms TTL=53
Réponse de 132.227.73.20 : octets=32 temps=23 ms TTL=53
Réponse de 132.227.73.20 : octets=32 temps=21 ms TTL=53

Statistiques Ping pour 132.227.73.20:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 21ms, Maximum = 23ms, Moyenne = 22ms
```

Traceroute (1)

- Traceroute
  - Le programme *traceroute* envoie des paquets IP destinés à une destination avec les valeurs TTL incrémentées (1, 2, 3, …)
  - Le N^{ième} routeur va recevoir le N^{ième} paquet dont sa durée de vie expire à ce routeur
  - Les routeurs qui écartent le paquet IP envoie un message ICMP (type=11, code = 0) à la source lui permettant de déterminer les noms et adresses IP de tous les routeurs jalonnant le parcours du paquet ainsi que les délais correspondants

Traceroute (2)

```
C:\Users\Univ P & M Curie>tracert www.lip6.fr

Détermination de l'itinéraire vers w.lip6.fr [132.227.73.20]
avec un maximum de 30 sauts :

  1     1 ms     1 ms     1 ms  192.168.0.254
  2    24 ms    21 ms    21 ms  diderot-3-81-57-106-254.fbx.proxad.net [81.57.10
6.254]
  3    24 ms     *        *     th2-6k-2-a5.routers.proxad.net [213.228.7.254]
  4    23 ms    22 ms    21 ms  bzn-crs16-1-be1004.intf.routers.proxad.net [212.
27.50.173]
  5    21 ms    22 ms    21 ms  aub-6k-1-po20.intf.routers.proxad.net [212.27.51
.82]
  6    22 ms    21 ms    22 ms  renater.routers.proxad.net [212.27.38.206]
  7    21 ms    22 ms    22 ms  te1-1-jussieu-rtr-021.noc.renater.fr [193.51.189
.229]
  8    22 ms    22 ms    22 ms  rap-ipv4-vl165-te3-2-jussieu-rtr-021.noc.renater
.fr [193.51.181.101]
  9    21 ms    22 ms    22 ms  site-6.01-jussieu.rap.prd.fr [195.221.127.182]
 10    23 ms    22 ms    23 ms  r-kennedy.reseau.jussieu.fr [134.157.254.29]
 11    23 ms    22 ms    22 ms  lip6-routeur.lip6.fr [132.227.106.31]
 12    23 ms    23 ms    22 ms  w.lip6.fr [132.227.73.20]

Itinéraire déterminé.
```