Cours Internet

Gilles MENEZ

12 décembre 2001

Table des matières

Cours Réseau : Internet

Gilles Menez

C.N.R.S. – I.3.S.

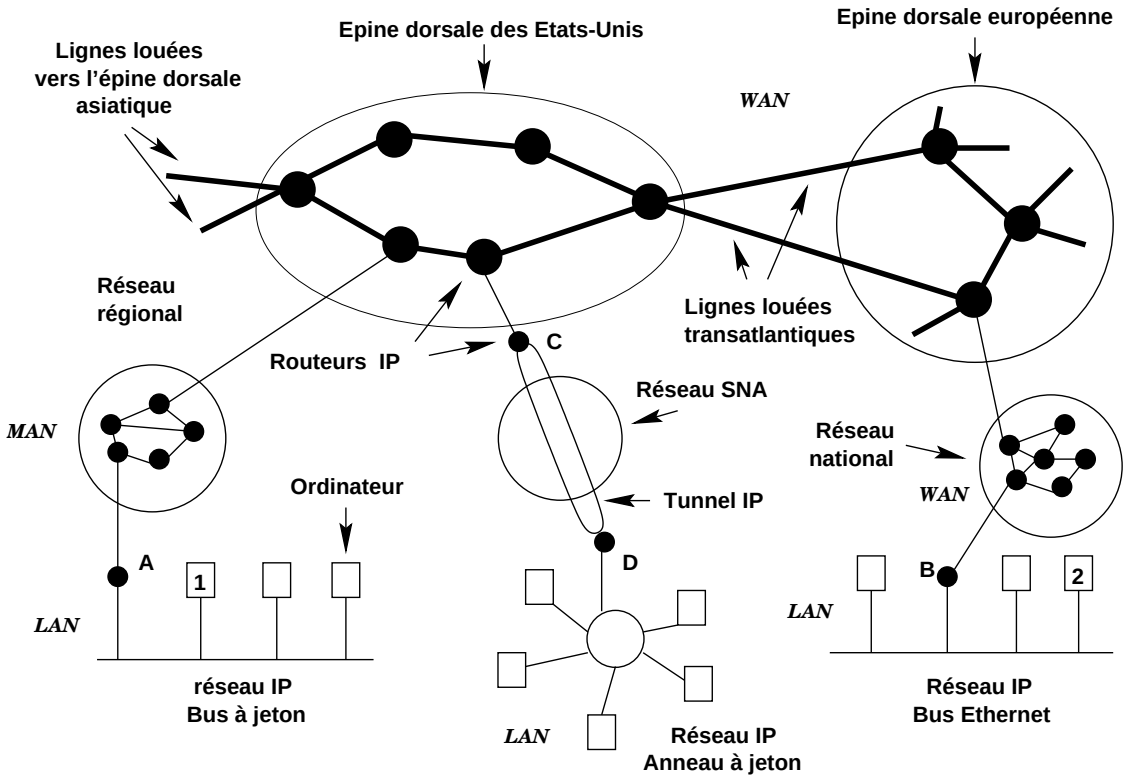
Université de Nice-Sophia Antipolis

2000, Route des Lucioles,

06410 Biot

email : menez@unice.fr

Des réseaux . . . trés différents . . .



Item	Quelques possibilités
Service offert	Connecté ou non connecté
Protocole réseau	IP, IPX, CLNP, Appletalk, DECnet, etc
Adressage	Uniforme(802) ou hiérarchique(IP)
Diffusion	possible ou non
Taille du paquet	chaque réseau définit sa propre taille
Qualité de service	Oui ou non ; divers types différents
Gestion des erreurs	Transmission : fiable, ordonnée ou désordonnée
Contrôle de flux	Fenêtre coulissante, contrôle de débit, . . .
Contrôle de congestion	Seau percé, contrôle d'engorgement, . . .
Sécurité	Stratégie de routage, chiffrement, . . .
Paramètres	Temporisations diverses, caractéristiques des flux, . . .
Coût des communications	Temps de connexion, nombre de paquets, . . .

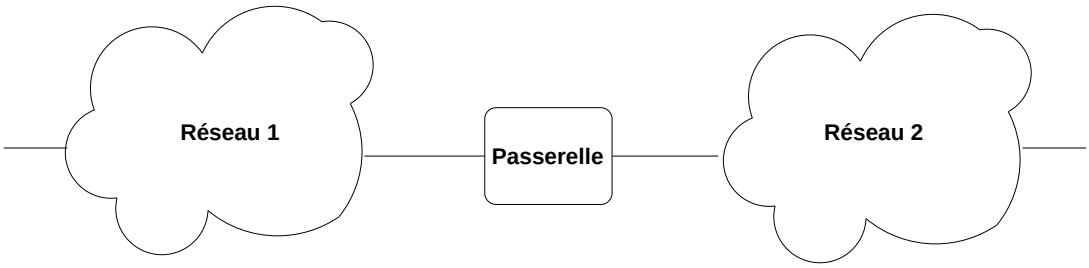
1 Interconnexion de réseaux

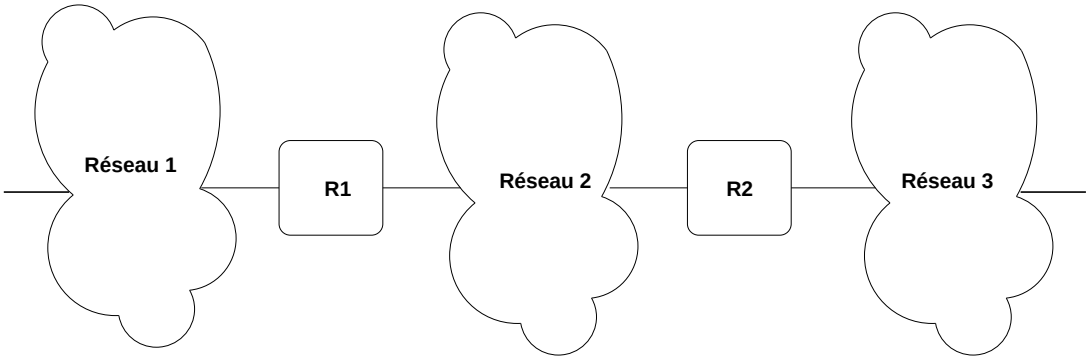
Les différents exemples de réseaux cités et étudiés (LAN, Ethernet, Token Ring, WAN, ATM, . . .) ont montré la grande diversité d'architecture, de protocoles, de méthodes d'adressage, ...

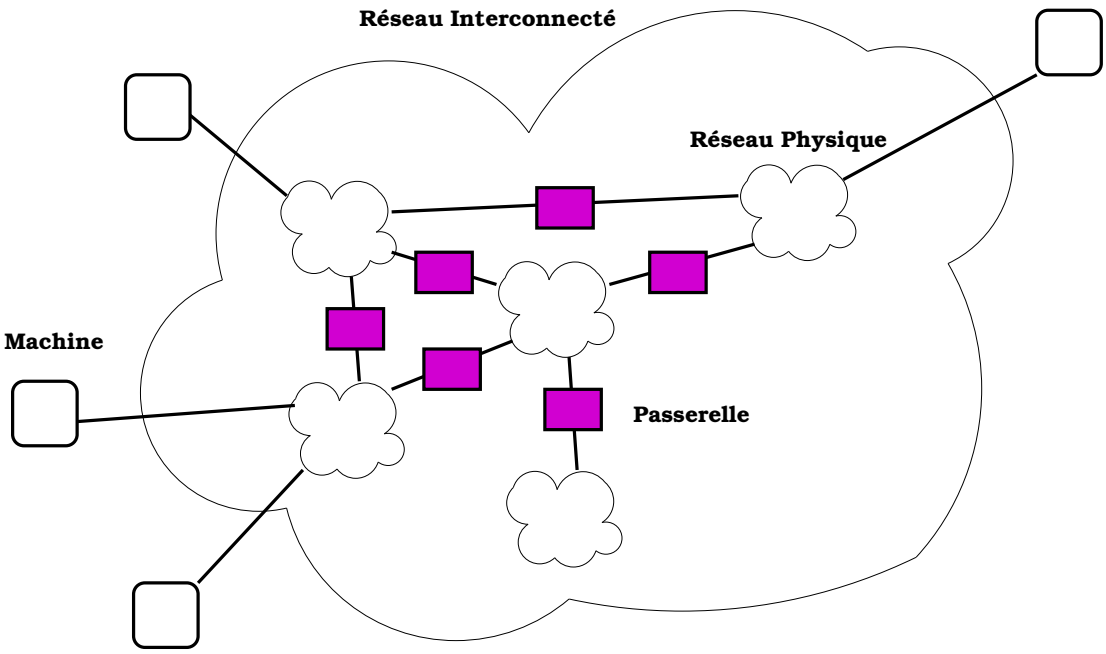
➢ Ce constat étant fait, on peut néanmoins imaginer que ces **réseaux fondamentalement trés différents soient capables d'échanger** des informations.

Ceci est la problématique d'Internet !

Interconnexion







1.1 Généralités sur l'interconnexion

Définition :

Physiquement, deux réseaux **ne peuvent être reliés** que par l'intermédiaire d'une machine connectée à chacun d'eux et qui sait acheminer des paquets d'un réseau à l'autre.

De telles machines sont appelées **passerelles** .

- La passerelle doit **accepter**, sur le réseau 1 les paquets destinés aux machines du réseau 2 et transmettre les paquets correspondants.
- De façon analogue, elle doit **accepter**, sur le réseau 2 les paquets destinés aux machines du réseau 1 et transmettre les paquets correspondants.

Passerelles et Routage :

① Les passerelles doivent bien évidemment savoir **comment router** les paquets vers leur destination.

➢ Ce sont souvent des mini-ordinateurs qui conservent des informations relatives à chacune des machines de l'interconnexion à laquelle ils sont reliés.

② Ainsi, lorsque **les interconnexion de réseaux deviennent plus complexes**,

➢ les passerelles doivent connaître des informations relatives à la topologie de l'interconnexion, au-delà du réseau auquel elles sont connectées.

③ Pour **minimiser la taille des passerelles**, les paquets sont acheminés en fonction du réseau destination et non en fonction de la machine destination.

➢ La quantité d'information gérée par une passerelle devient alors proportionnelle au nombre de réseaux de l'interconnexion et non au nombre de machines.

Réseau Virtuel Unique :

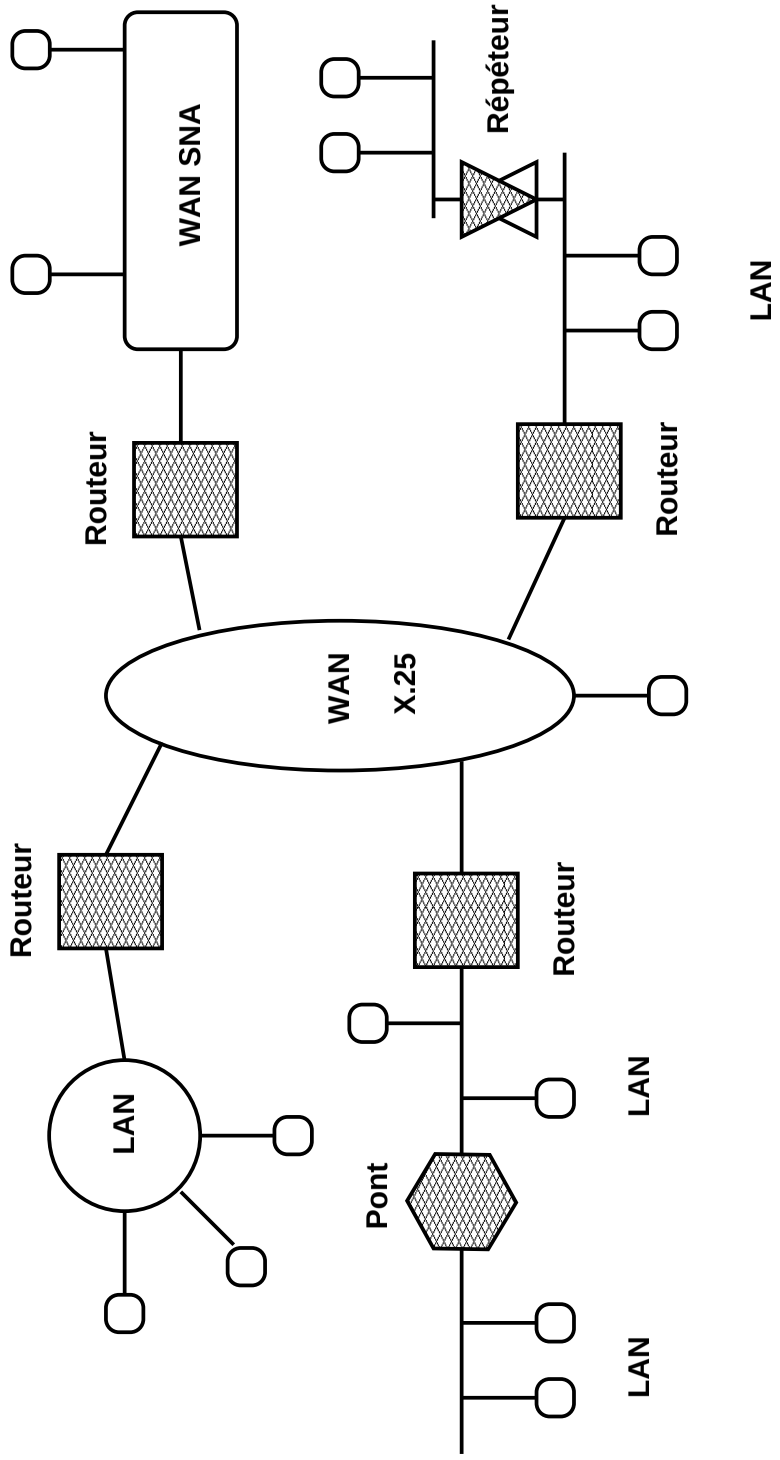
L'utilisateur voit une interconnexion comme un **réseau virtuel unique** auquel les machines sont connectées.

➢ **Pour acheminer des paquets entre des paires quelconques de réseaux**, les passerelles peuvent être obligées de leur faire traverser plusieurs **réseaux intermédiaires**.

Les réseaux de l'interconnexion **doivent accepter** que des données extérieures les traversent.

➢ Les utilisateurs ordinaires ignorent l'existence du trafic supplémentaire acheminé par leur réseau.

Interconnexion



1.2 Unités fonctionnelles d'interconnexion de réseau

On ne conçoit plus désormais un réseau local sans une ouverture vers le monde extérieur.

> Il devient nécessaire d'**interconnecter les réseaux entre eux et de pouvoir les raccorder** aux moyens de communication publics ou privés à grande distance.

Les unités fonctionnelles d'interconnexion de réseau sont les boîtes noires qui vont permettre d'interconnecter les réseaux.

Le nom et la fonction des unités d'interconnexion dépend étroitement de la couche dans laquelle s'opère le transfert :

- ➢ le répéteur,
- ➢ le pont ou **bridge** ,
- ➢ le routeur,
- ➢ la passerelle de transport,
- ➢ la passerelle d'application : elle réalise l'interconnexion entre applications de couches supérieures, elle agit dans la couche application.

1.2.1 Les répéteurs

Définition :

Les **répéteurs** sont des unités de bas niveau qui :

- amplifient ou
- régénèrent

les signaux électriques affaiblis afin qu'ils puissent être reçus dans de bonnes conditions par un récepteur distant.

> Un répéteur prolonge le support physique.

Par exemple, c'est ce qui permet de relier les segments Ethernet : on se rappelle qu'un segment Ethernet **mesure au maximum** (pour des problèmes de puissance électrique des transceivers) 500 mètres mais que la **vitesse de propagation** autorise une longueur maximum de 2.5 Km.

① Le répéteur ne regarde pas le contenu de la trame, ni les adresses de destination.

② Il n'a pas d'existence logique donc pas d'adresse Ethernet.

③ Il ne diminue pas la charge : Les deux segments ne font qu'un !

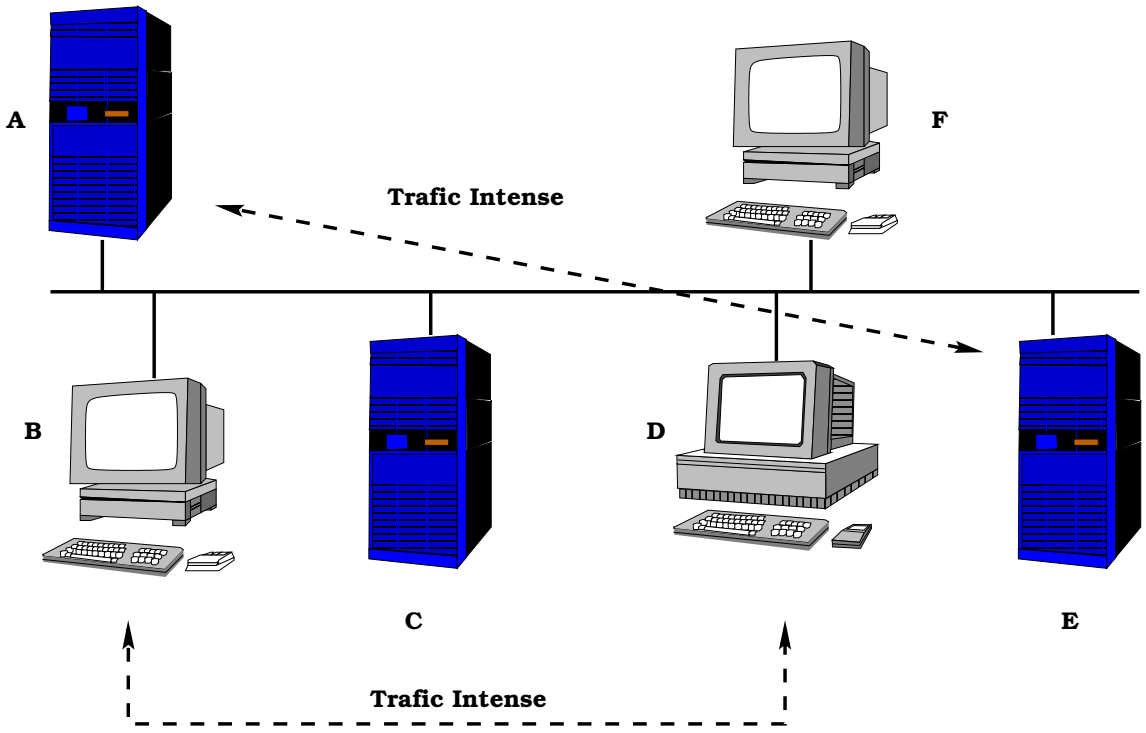
④ Il ne filtre pas les collisions !

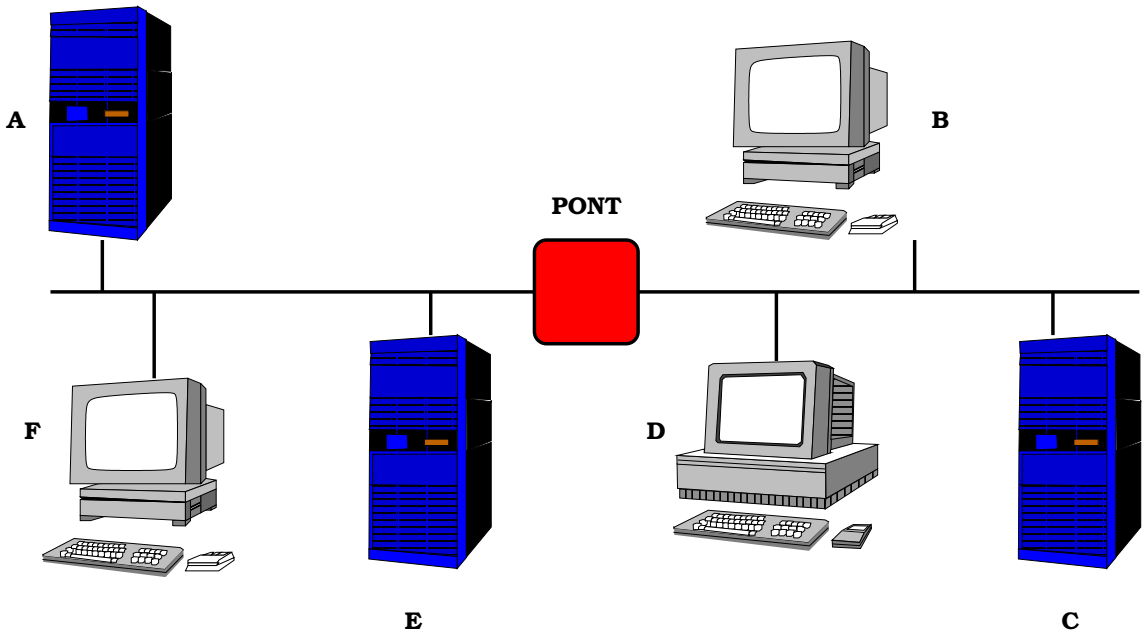
1.2.2 Les multirépéteurs

Aussi appelé **HUB** , ils assurent la même fonction de répéteur dans le cadre d'une structure en étoile. Ils sont souvent un point de rencontre pour différentes technologies de média :

- coax
- paire torsadée
- fibre

Ponts





1.2.3 Les ponts

Définition :

Les **ponts** , sont conçus pour construire un réseau local logique à partir de plusieurs réseaux locaux homogènes distants.

Les ponts recoivent les trames, **les analysent et les font suivre** vers leur destination.

① **Le pont mémorise et retransmet** : il n'y a plus de limitation en distance pour Ethernet.

On dit qu'ils **travaillent au niveau liaison** (couche MAC).

① L'analyse et la vérification portent par exemple sur le total de contrôle.

② Un pont peut filtrer, c'est à dire ne pas retransmettre, des trames érronées. Il s'agit de **pont filtrant** .

③ Il permet aussi de localiser les collisions sur un segment.

④ Le pont a une adresse Ethernet (MAC).

⑤ Il est néanmoins ignoré des couches supérieures et d'ailleurs lui même n'interprete pas le contenu des trames : il ne peut donc pas filtrer un broadcast IP.

⑥ Il peut à la rigueur faire quelques modifications sur l'entête de la trame.

Le pont est un entité « intelligente » qui doit savoir où sont ces interlocuteurs pour retransmettre.

- Auto learning (écoute et construit)
- Table figée
- Mixte

Ponts et performances :

Les ponts permettent essentiellement de **segmenter** un réseau local en deux sous-réseaux **pour améliorer** les performances.

Dans un réseau Ethernet par exemple qui approche de la saturation, on peut chercher les couples de machines qui ont un gros trafic entre elles et les isoler de chaque coté du pont.

➢ Le pont permet ainsi de **localiser le trafic** donc améliore le comportement local qui n'a plus a subir des trafics qui ne lui étaient pas destinés.

Trés utilisé avant, il est aujourd'hui supplanté par le routeur . . . encore plus intelligent et guère plus cher.

1.2.4 Les routeurs

Définition :

Les **routeurs** sont conceptuellement semblables aux ponts, sauf qu'ils **travaillent dans la couche réseau**.

- ➢ Ils sont destinés à relier plusieurs réseaux de technologies différentes.
- ➢ Ils effectuent le routage des informations à travers l'ensemble des réseaux inter-connectés.

Ils travaillent sur l'entête IP, qu'ils utilisent pour :

- router
- filtrer les paquets (datagrammes) inutiles, les broadcasts
- filtrer les trames (Ethernet)

➢ Les routeurs nécessitent beaucoup d'intelligence (CPU, mémoire), ils peuvent constituer un matériel dédié ou non (PC ou SUN).

Le routeur peut aussi effectuer des conversions entre réseaux différents :

$$\text{IP} < --- > \text{X.25}$$
$$\text{IP} < --- > \text{IPX}$$
$$\ldots$$

Certains (B-router) ou pont-routeurs sont une évolution des routeurs :

- ➢ ils routent les protocoles qu'ils connaissent et
- ➢ font office de ponts pour les autres.

On peut aussi parler de passerelle dans le cas des routeurs ... même si ... voir ci-aprés.

1.2.5 Les passerelles de transport

Définition :

Les **passerelles** , ou **gateways** , entrent en scène dans les cas les plus complexes pour assurer une **compatibilité au niveau des protocoles de couches hautes entre réseaux hétérogènes**.

Ces passerelles travaillent dans les couches transport, et permettent à deux architectures de discuter :

$$\text{DECNET} < - - - > \text{IP}$$
$$\text{Localtalk Appletalk} < - - - > \text{Ethernet IP}$$
$$\ldots$$

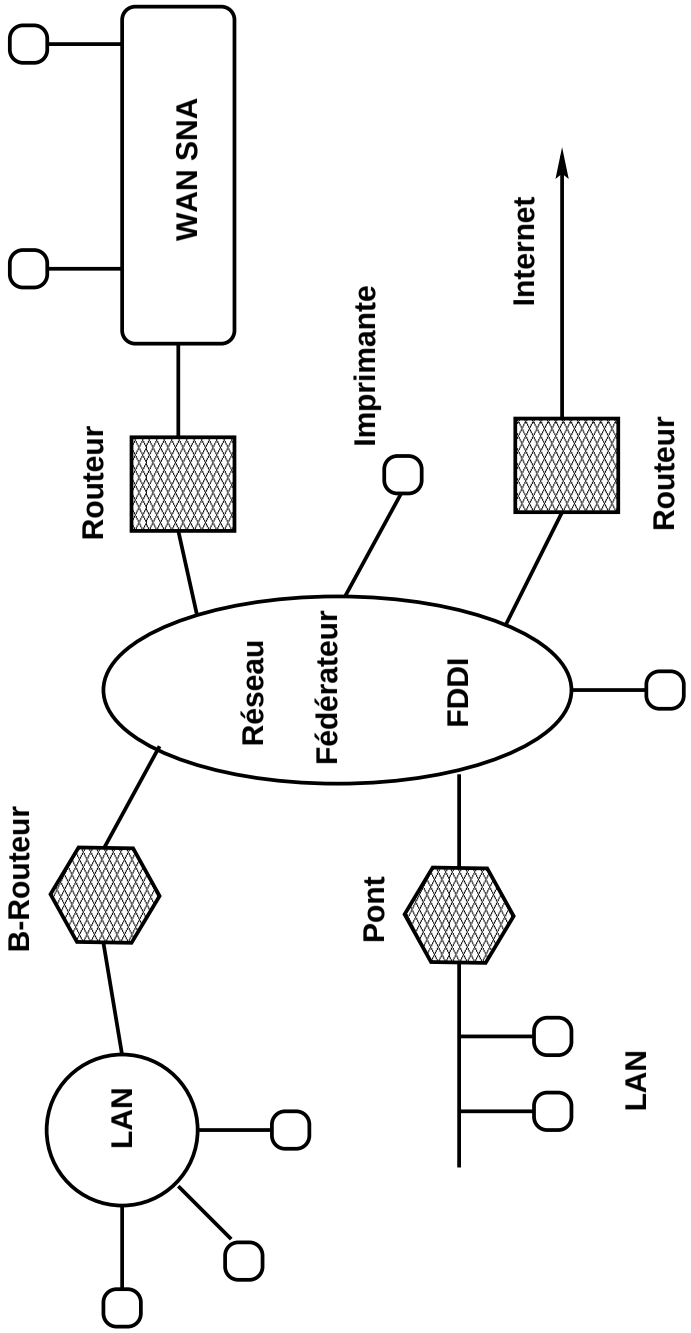
1.2.6 Les passerelles d'applications

Définition :

Ces passerelles mettent en relation deux parties d'une application globale répartie :

$$\text{Telnet} < - - - > \text{SETHOST}$$
$$\text{Telnet} < - - - > \text{PAD}$$
$$\text{SMTP} < - - - > \text{X400}$$

Backbone


WAN SNA
Routeur
Imprimante
Internet
Routeur
Réseau
Fédérateur
FDDI
B-Routeur
Pont
LAN
LAN

1.2.7 Réseau fédérateur

Les grandes entreprises sont généralement **structurés en départements** qui sont dans des étages différents d'un bâtiment voire sur plusieurs bâtiments d'un même site.

① Le trafic global généré peut être important avec de gros échanges de données au sein des départements.

② Il est fréquent de constituer des réseaux pour chaque département (ou groupe de départements)

③ et de **relier tous ces réseaux par un réseau à haut débit** qui fonctionne en **réseau fédérateur** ou **backbone** .

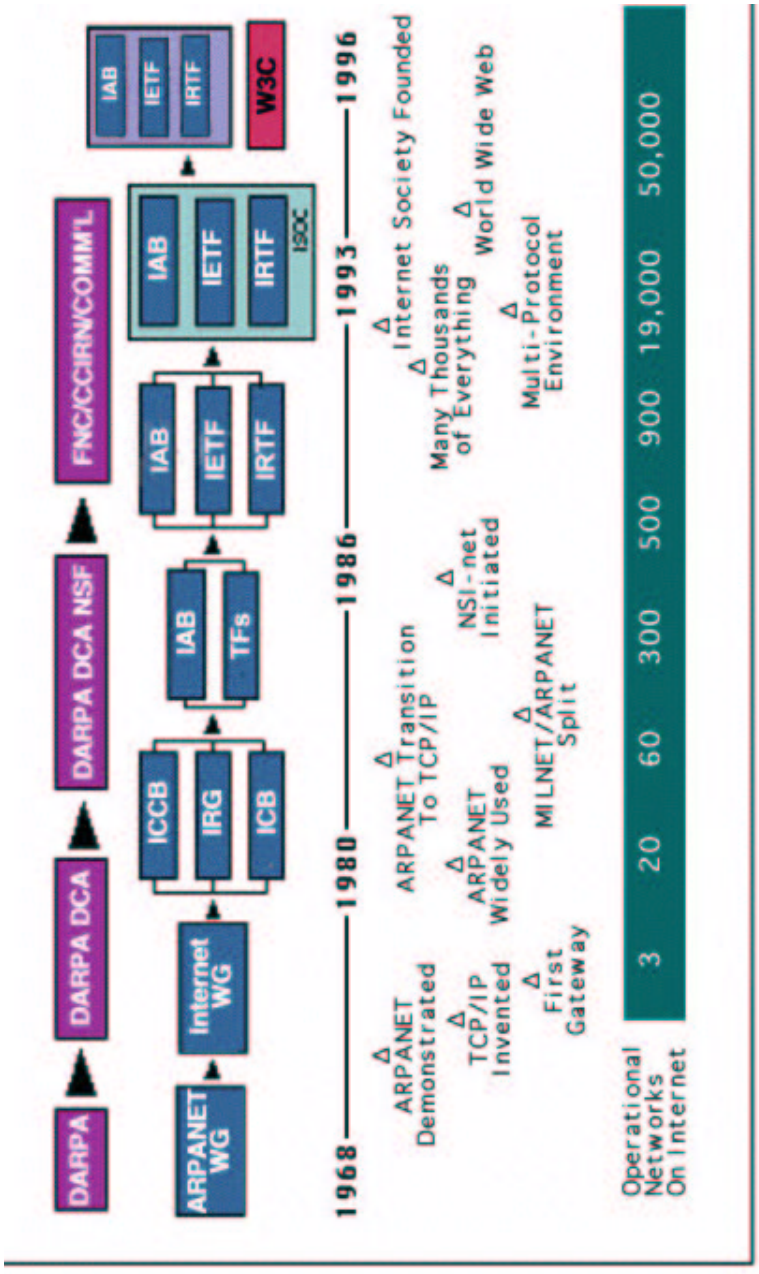
La liaison de chaque réseau au réseau fédérateur se fait au minimum par un pont.

➢ Certains moyens communs à l'ensemble de l'entreprise peuvent être mis sur le réseau fédérateurs

➢ et devenir accessibles à tous de même que les accés à des réseaux externes (par exemple Internet).

Internet : L'histoire

http ://www.isoc.org/zakon/Internet/History/HIT.html



2 Internet

2.1 Historique

① En 1969, fut créé aux Etats-Unis le réseau Arpanet sous l'impulsion du DARPA (Defense Advanced Research Projects Agency).

Ce réseau, initialement destiné à la recherche, avait un double objectif :

(a) permettre aux université, aux militaires et à certains centre de recherche d'échanger des informations

(b) et d'expérimenter les techniques de commutation par paquets.

➢ Il permettrait notamment d'étudier comment les communications pouvaient être maintenues en cas d'attaque nucléaires.

② Devant le déploiement parallèle d'autres réseaux et des réseaux locaux d'entreprise, il apparut intéressant de pouvoir les relier entre eux, indépendamment de leurs technologies respectives pour offrir un service de réseau global.

➢ Deux protocoles furent alors développés et prirent leur forme définitive dans les années 77-79 :

(a) TCP : Transport Control Protocol

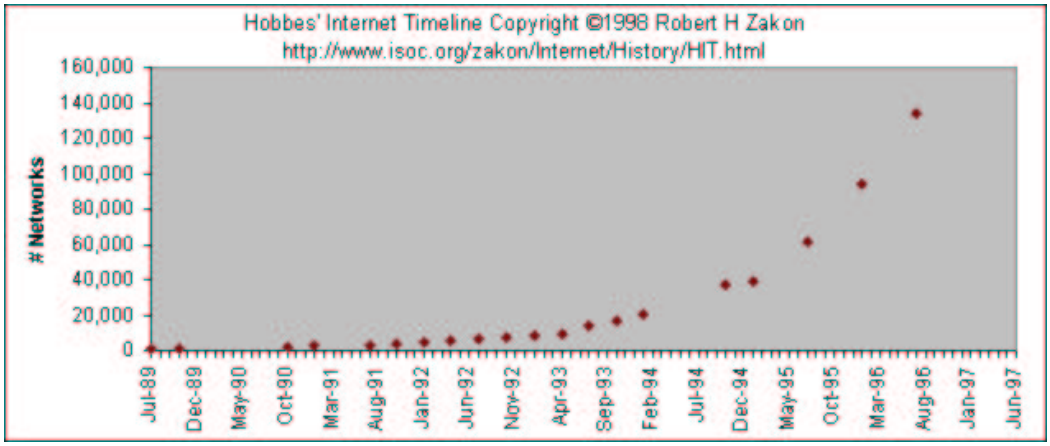
(b) IP : Internet Protocol

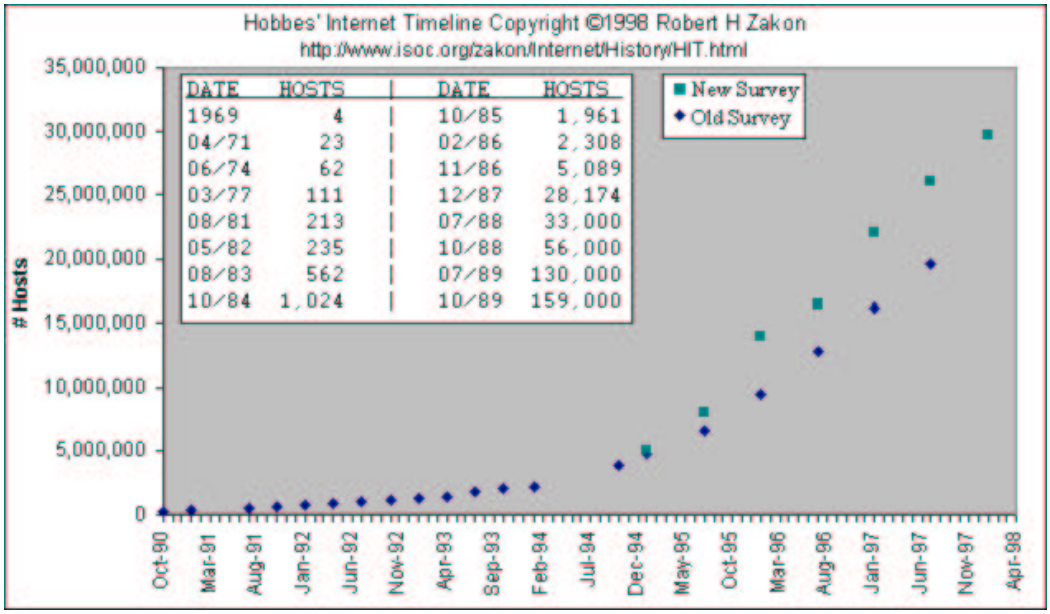
③ Ces protocoles furent implantés sur le réseau Arpanet qui devint la base du réseau Internet au début des années 80.

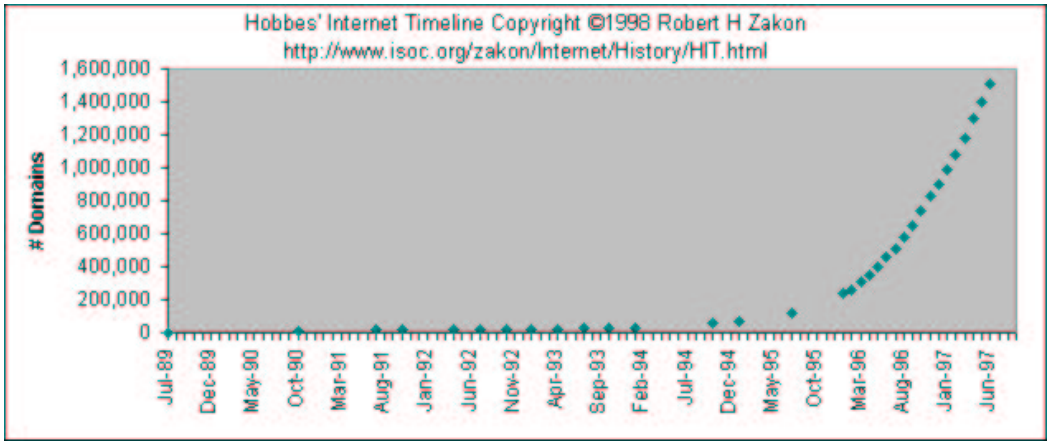
④ La DARPA sépara d'Arpanet le réseau militaire qui prit le nom de Milnet.

⑤ Pour favoriser l'adoption des nouveaux protocoles, la DARPA créa une société chargée de les développer et subventionna l'université de Berkely pour qu'elle les intègre à son système d'exploitation Unix, lui-même distribué à bas prix aux universités.

Internet : La croissance







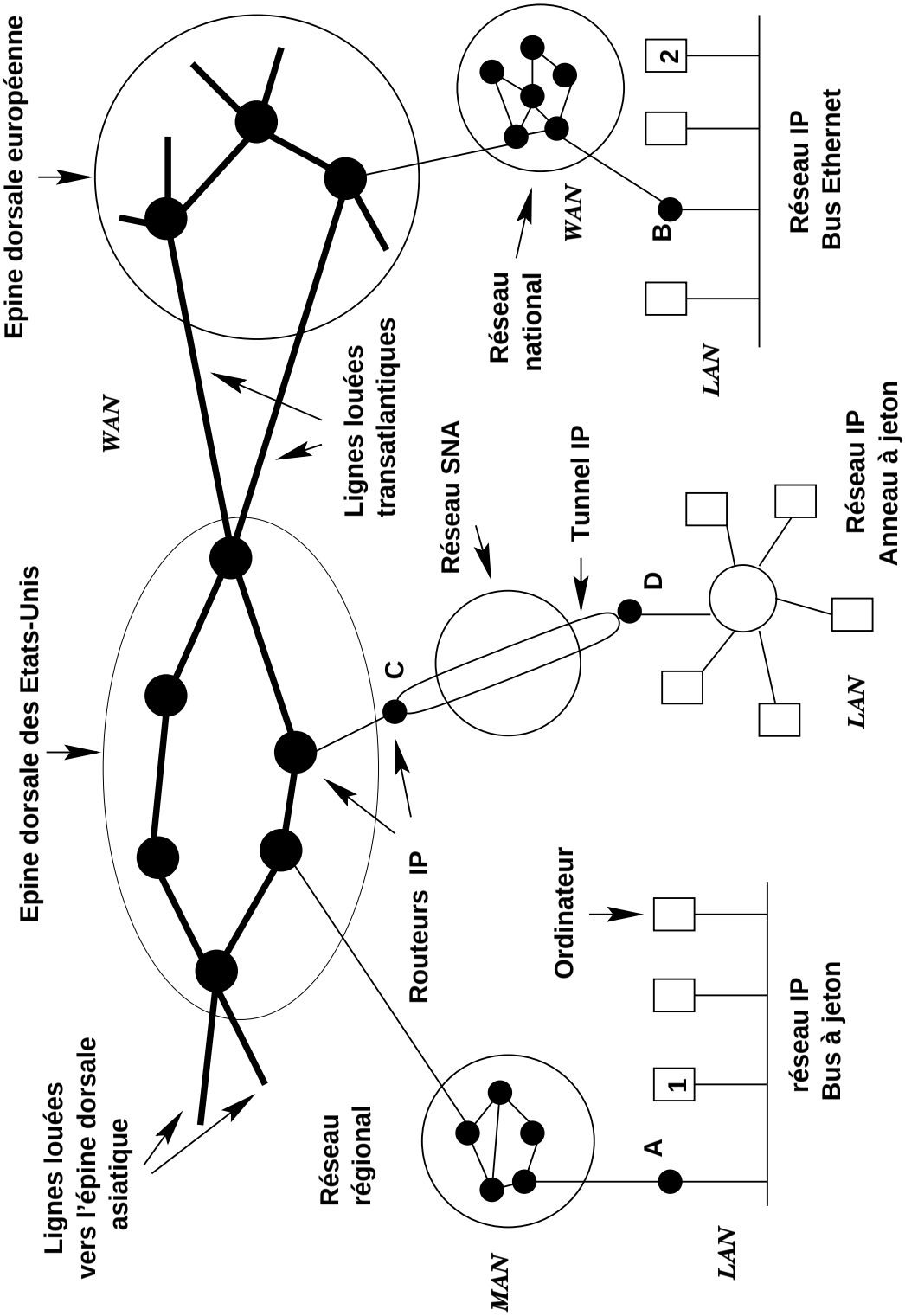
2.2 La croissance

Le réseau **Internet** est le plus connu et le plus grand.

- En 1993, il connectait déjà 1.3 million d'ordinateurs,
- Et surtout, il avait une croissance annuelle de 80%.

Autre paramètre significatif, le nombre de réseaux conectés, on n'avait pas prévu une telle croissance !

Organisation d'Internet



2.3 Plaidoyer pour l'interconnexion des systèmes

L'objectif initial consiste à développer des technologies permettant d'**interconnecter** les systèmes informatiques pour :

① échanger des informations

② et partager des moyens

Définition :

Internet veut donc répondre au problème de la coopération de ces multiples réseaux dans le but de satisfaire un **besoin de communication global** :

- Internet = un réseau global
- Internetworking = techniques d'interconnexion des réseaux

Internet fait le constat qu'il **n'y a pas qu'un réseau**, mais plusieurs différents et indépendants : LAN, WAN, Backbone, lignes louées . . . :

➢ Si chaque réseau accepte d'acheminer un trafic supplémentaire au sien propre et qui correspond à du trafic en transit

➢ alors en contrepartie, il a le droit d'écouler du trafic sur l'intégralité de l'Internet.

En ce sens Internet est un **réseau coopératif** .

Bien entendu,

- les interfaces entre les réseaux seront réalisées par les unités fonctionnelles d'interconnexion de réseau
- et tout ceci doit être transparent à l'utilisateur.

2.3.1 Contraintes soumises et remplies par Internet et l'architecture de ses protocoles TCP/IP

① Indépendance technique vis à vis réseaux matériels

② Connectivité Universelle

③ Gestion possible d'accusé de réception de bout en bout (service de transport fiable)

④ Fournir des protocoles d'application standard : Les services de base (mail, ftp, . . .) sont assurés par les protocoles de TCP/IP (SMTP, FTP, . . .).

⑤ La technique Internet est dite **ouverte** puisqu'elle traite de l'interconnexion des systèmes dont les spécifications sont publiques et disponibles :

➢ l'enrichissement est possible !

2.4 Objectifs et hypothèses de bases d'Internet

Internet **ne constitue pas** un nouveau type de réseau physique.

- ➢ Il offre, par l'interconnexion de multiples réseaux, **un service de réseau virtuel mondial** permettant la mise en relation de plusieurs centaines de millions d'ordinateurs.
- ➢ Les divers réseaux sont interconnectés par des routeurs.

Pour Internet, tous les réseaux qui le constitue sont **équivalents**.

Il repose sur des solutions pragmatiques :

① Ce réseau virtuel repose sur un **adressage global** se plaçant au-dessus des différents réseaux utilisés.

② Lorsque des données empruntent plusieurs réseaux, la qualité de l'échange est globalement donnéee par le réseau le plus faible : il suffit qu'un seul des réseaux empruntés perde des paquets pour que l'échange ne soit pas fiable.

- ➢ Internet offre donc un **service non fiable de remise de paquet en mode sans connexion**. C'est le principe du **datagramme** développé par le protocole IP.

③ La **fiabilisation** des dialogues, lorsqu'elle est nécessaire, est réalisée **aux niveaux des extrémités** par TCP.

④ D'autres applications qui n'ont pas besoin de cette fiabilité peuvent utiliser un autre protocole de transport : UDP (User Datagram Protocol).

⑤ Les protocoles d'Internet (TCP/IP) traitent **tous les réseaux de la même façon** !

Aux Etats-Unis, la NSF (National Science Foundation) a installé le réseau NFSNET.
En France le réseau RENATER sert de backbone à l'enseignement et à la recherche.

2.4.1 Intranet

Internet est un réseau global, mais l'internetworking constitue l'ensemble des techniques d'interconnexion des réseaux sous-jacentes.
L'industrie a largement adopté ces techniques.

Définition :

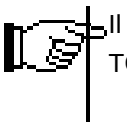
Intranet est un réseau privé d'entreprise fondé sur les protocoles et les applications du réseau Internet.

2.5 Les services d'un Internet

Les services d'interconnexion sont présentés comme des protocoles de communication.
On peut distinguer 2 classes de services :

1. Services de niveau application
2. Services de niveau réseau

2.5.1 Services de niveau application



Il s'agit des services ne nécessitant pas la connaissance des aspects techniques de TCP/IP.

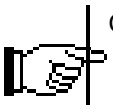
Ces services font preuve d'un haut niveau d'interopérabilité de l'internet :

tous les systèmes informatiques vont collaborer à la réalisation du service.

On peut citer à titre d'exemple :

- mail
- ftp
- telnet

2.5.2 Services de réseau



Ceux sont les services spécifiques à l'infrastructure du réseau.

➢ Ils sont de deux types.

Service de remise en mode non connecté :

C'est l'échange de blocs de données, les paquets, entre 2 parties SANS ENTENTE préalable.

➢ Internet va router ce bloc.

Internet est donc un réseau de paquets. Plusieurs raisons à cela :

① La taille réduite d'un paquet (par rapport à un message) permet d'être plus facilement en **correspondance avec les capacité du matériel** (buffers, ...) d'où une gestion plus efficace.

② Les routeurs vont gérer des paquets.
Ils n'ont pas besoin de comprendre les informations utilisées par les programmes d'applications et qui donnent un sens à un ensemble de paquets.
Les activités de communications sont donc **découplées** des programmes d'application. Elles vont pouvoir évoluées facilement ... exemple IP6.

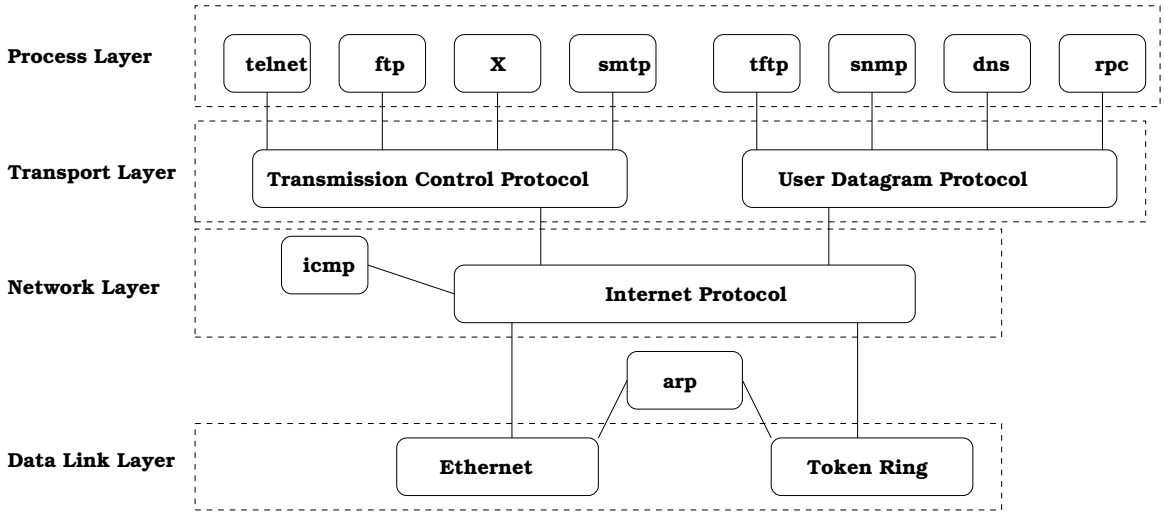
③ Le paquet, en soi, est un abstraction assez souple pour permettre des **évolutions** au niveau des protocoles.

Service de transport fiable :

Ce service induit :

- Un notion de connexion : une mise en relation préalable
- notion de circuit
- reprise sur erreur

Architecture Internet : Protocoles



An addressing scheme to logically identify devices and to group them into networks Internet Protocol	IP
The ability to resolve physical addresses when a logical IP address is known Address Resolution Protocol	ARP
A redundant storage system for network address information Domain Name Service	DNS
The ability to re-route to an alternative path if a current path becomes unusable ...through router configuration, later to be Routing Information Protocol	RIP
The notification of error conditions on the network Internet Control Message Protocol	ICMP
A simple mechanism for exchanging data messages User Datagram Protocol	UDP
The reliable transfer of data as required Transmission Control Protocol	TCP
The ability to transfer files File Transfer Protocol	FTP
Trivial File Transfer Protocol	TFTP
A mail messaging system Simple Main Transfer Protocol	SMTP
The ability to act as a terminal to a host computer	Telnet
Enhanced routing capabilities Open Shortest Path First	OSPF
Cisco's Internet Gateway Routing Protocol	IGRP
Enhanced Internet Gateway Routing Protocol	EIGRP
Expanded capabilities for storing network information other than addresses Network Information Service	NIS
Remote router management Simple Network Management Protocol	SNMP
The ability to mount a remote file volume and treat it as if it were local Network File System protocol	NFS
A programming interface to facilitate the programming of network services Remote Procedure Call protocol	RPC
Enhanced capabilities to provide bootstrap information for network clients Boot Protocol	BOOTP
Dynamic Host Configuration Protocol	DHCP
The acquisition of network performance data The Remote Monitoring MIB	RMON

2.6 Architecture en couche d'Internet

L'architecture du protocole Internet représente **l'évolution du consensus** entre plusieurs

- ① constructeurs,
- ① universitaires,
- ① vendeurs,
- ① chercheurs,
- ① organismes gouvernementaux

aussi bien dans le domaine du logiciel que du matériel.

Le processus de standardisation est géré par l' **Internet Architecture Board**

L'architecture globale d'Internet définit 4 couches :

1. La couche **application** :
   C'est la couche la plus élevée. On y fait une distinction entre :
   - ➢ les protocoles "user application" : FTP (File Transfert Protocols), SMTP (Simple Mail Transfert Protocols)
   - ➢ et les protocoles "support application" : SNMP (Simple Network Management Protcols), DNS (Domain Name System)
2. La couche **transport** :
   Conformément au modèle OSI, la couche transport fournit des services de communication "end–to–end" aux applications (acheminement de messages entre applications, entre processus).
   - ➢ Les deux protocoles majeurs sont TCP (Transmission Control Protocol)
   - ➢ et UDP (User Datagram Protocol).
3. La couche **internet** :
   Une des caractéristiques définissant Internet est l'utilisation de IP (Internet Protocol) pour fournir des services de "niveau réseau"(acheminement de messages entre systèmes : routage et contrôle de flux, ...)).
   - ➢ Plusieurs protocoles de contrôle et de gestion (ICMP Internet Control Message Protocol) sont utilisés pour accéder aux services fournis par IP.
   - ➢ IP est un protocole de type "connectionless".
4. La couche **liaison** :
   C'est la couche la plus basse.
   - ➢ C'est le moyen par lequel les hôtes s'interfacent avec le réseau local. On peut y trouver différents protocoles (Ethernet, Token Ring, ...).
   - ➢ C'est la couche de la gestion des erreurs, ...

La suite de protocoles (reposant sur IP) utilise par conséquent une technologie en **mode non connecté**.

- ➢ Les informations transférées au niveau IP sont structurées en **datagrammes** indépendants les uns des autres : chaque datagramme est transmis endépendamment des autres.

2.7 Les protocoles d'Internet

Une famille de protocoles de communications ont été developpés pour satisfaire les objectfis initiaux d'ARPANET.

➢ Les objectifs originels continuent d'être remplis par ces protocoles désormais connus comme l'**architecture TCP/IP** ou encore les **protocoles IP**.

Ces objectifs sont les suivants :

① An addressing scheme to logically identify devices and to group them into networks Internet Protocol (IP)

② The ability to resolve physical addresses when a logical (IP) address is known
Address Resolution Protocol (ARP)

③ A redundant storage system for network address information
Domain Name Service (DNS)

④ The ability to re-route to an alternative path if a current path becomes unusable
...through router configuration, later to be Routing Information Protocol (RIP)

⑤ The notification of error conditions on the network
Internet Control Message Protocol (ICMP)

⑥ A simple mechanism for exchanging data messages
User Datagram Protocol (UDP)

⑦ The reliable transfer of data as required
Transmission Control Protocol (TCP)

⑧ The ability to transfer files
File Transfer Protocol (FTP)
Trivial File Transfer Protocol (TFTP)

⑨ A mail messaging system
Simple Main Transfer Protocol (SMTP)

⑩ The ability to act as a terminal to a host computer
Telnet

➢ Internet est une technologie ouverte, et avec les années des ajouts ont été faits pour améliorer les protocoles (performances et fonctionnalités) initiaux.

Les évolutions majeures sont :

① Enhanced routing capabilities
Open Shortest Path First (OSPF)
Cisco's Internet Gateway Routing Protocol (IGRP)
Enhanced Internet Gateway Routing Protocol (EIGRP)

② Expanded capabilities for storing network information other than addresses
Network Information Service (NIS)

③ Remote router management
Simple Network Management Protocol (SNMP)

④ The ability to mount a remote file volume and treat it as if it were local
Network File System protocol (NFS)

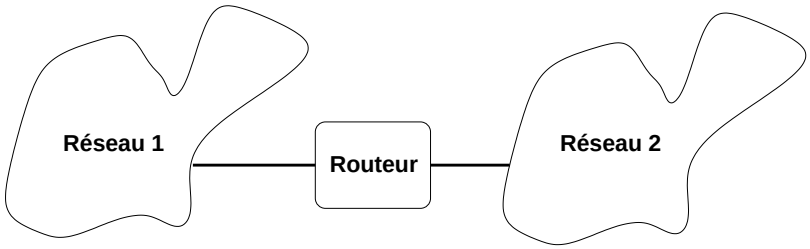
⑤ A programming interface to facilitate the programming of network services
Remote Procedure Call protocol (RPC)

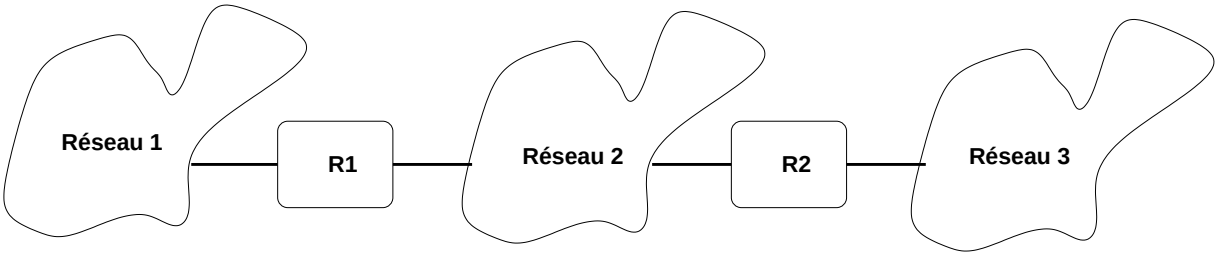
⑥ Enhanced capabilities to provide bootstrap information for network clients
Boot Protocol (BOOTP)
Dynamic Host Configuration Protocol (DHCP)

⑦ The acquisition of network performance data
The Remote Monitoring MIB (RMON)

Adresses et routeurs

Adresse IP	8bits		8bits		8bits		8bits
Adresse IP	1	.	2	.	255.		4
0x	01		02		FF		04





3 Adressage

Internet veut fournir un **service de communication « universel »** qui permette à n'importe quel type d'ordinateur, sur n'importe quel type de réseau de communiquer.

➢ La première chose à faire c'est établir un mécanisme d'identification universel qui soit « reconnaissable » et « acceptable » par tous ceux qui veulent faire partie d'Internet.

Définir un ensemble de noms ou d'adresses aptent à connecter « tout le monde ».

Définition :

Cette adresse constitue une **adresse logique** .

3.1 Adresse Internet

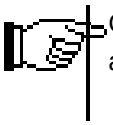
Les concepteurs de TCP/IP ont choisi une méthode analogue à celle de l'adressage physique.

- Ils auraient pu associer un nom ou une route à une machine
- Ils ont choisi une adresse binaire, c'est à dire un nombre sur 32 bits.

Autrement dit, en IPV4, il y a 2^{32} adresses possibles : 4,2 milliards d'adresses possibles.

Cette adresse est souvent représentée sous une forme « pointée ».

8bits	.	8bits	.	8bits	.	8bits
0x01		02		FF		04
1	.	2	.	255	.	4



Chaque ordinateur et chaque routeur du réseau Internet possède au moins une adresse IP qui l'identifie.

3.2 Adresses et routage

Les routeurs sont les premiers éléments du réseau Internet concernés par les adresses :

➢ c'est eux qui assurent le transit,

➢ c'est donc qui analysent les adresses et les exploitent.

① Lorsqu'un Internet devient complexe, il comporte de **nombreux réseaux** et donc de nombreux routeurs.

② Or les routeurs doivent connaitre des informations relatives à la **topologie globale** de l'Internet bien au dela des réseaux qu'ils raccordent :

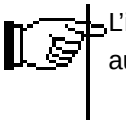
Le routeur R1 doit acheminer vers le réseau 2 tous les blocs de données émis sur le réseau 1 destinés au réseaux 2 **et** 3.

③ Au fur et à mesure que la dimension de l'Internet s'accroît, la tâche du routeur qui consiste à un faire un choix de route devient plus complexe car il doit gérer de **plus en plus d'entrée** dans ces tables de routage.

L'ingéniosité des adresses IP vient de leur efficacité dans la tâche de routage.

En effet, pour simplifier la compléxité de l'opération de routage, les routeurs vont acheminer les blocs de données

- ➢ En fonction du réseau de destination
- ➢ et NON en fonction de l'ordinateur de destination



L'information manipulée est proportionnelle au nombre de réseaux de l'Internet et non au nombre de ses ordinateurs.

3.3 Adresse IP (suite)

Une adresse IP est formée de 4 octets et contient donc :

1. L' **identification du réseau** sur lequel elle est connectée : id_res
2. L' **identification de l'ordinateur** sur ce réseau : id_ord

#IP	8bits	.	8bits	.	8bits	.	8bits
#IP en hexa	C0		29		06		14
#IP en dot	192	.	41	.	6	.	20

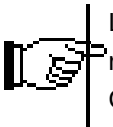
Cette adresse est unique sur Internet.

Les ordinateurs connectés simultanément sur plusieurs réseaux possèdent **une adresse IP différentes sur chacun des réseaux**.

- ➢ C'est le cas des routeurs.

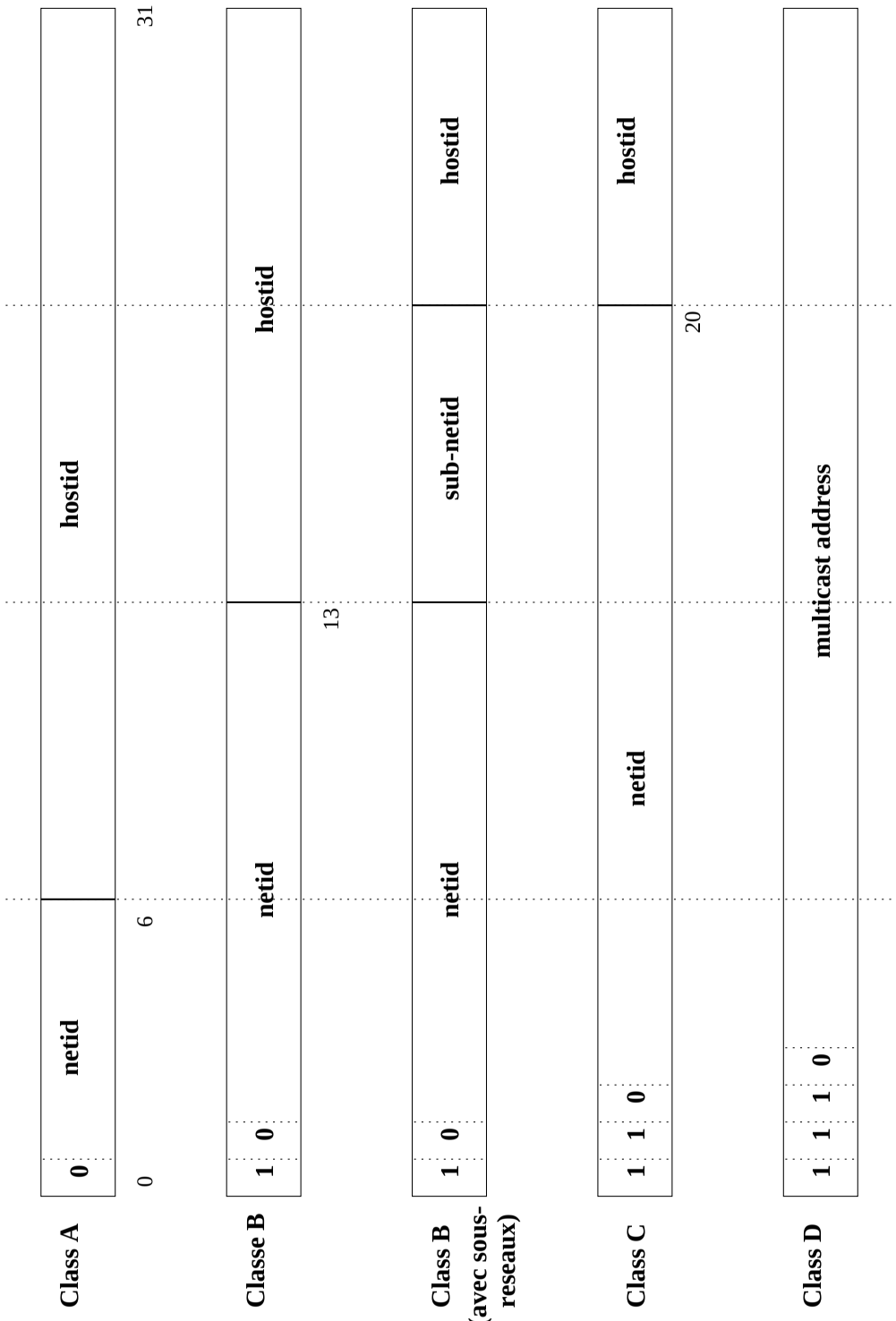
  Ce type d'ordinateur est dit « multi-homed ».
- ➢ C'est l'occasion de préciser que l'adresse **ne spécifie pas un ordinateur ou une machine** mais le **point d'accés** de cet ordinateur au réseau.

  On parle de son **interface** .



L'attribution d'une adresse dépend toujours (sauf si le réseau est fermé) d'une arborescence d'organismes centralisateurs dont la racine est le NIC (Network Information Center).

Classe des adresses Internet



3.4 Classes d'adresses

Il ne servirait à rien de constater que l'adresse doit être composée d'une partie réseau et d'une partie ordinateur si on ne savait de combien d'éléments chaque partie se compose.

Ceci est pour le moins difficile à définir a priori.

➢ C'est pourquoi Internet a prévu 5 classes d'adresses.

> Les 3 premières classes permettent de moduler l'occupation de id_res grâce au sous-adressage.

La classe d'une adresse peut être déterminée à partir de ces 5 premiers bits . . . efficacité oblige.

Class	Class bits	#bits id_res	first	last	comment
A	0	7	0.1.0.0	126.0.0.0	0.0.0.0 et 127.0.0.0 sont réservées 16.0.0.0 (DEC) 18 (MIT)
B	10	14	128.0.0.0	191.255.0.0	IMAG : 129.88, Jussieu :134.157
C	110	21	192.0.1.0	223.255.255.0	Classe des LANs
D	1110	-	224.0.0.0	239.255.255.255	Multicast
E	11110	-	240.0.0.0	247.255.255.255	Martiens à venir

① Adresses de classe A :

Elles correspondent à des adresses de machines appartenant à de grands réseaux : 126 réseaux de 16 Méga - 2 machines.

② Adresses de classe B :

Elles correspondent à des adresses de machines appartenant à des réseaux de taille moyenne : (16K-2) réseaux de (64K-2) machines.

③ Adresses de classe C :

Elles correspondent à des adresses de machines appartenant à de petits réseaux : (2M-2) réseaux d'au maximum 254 machines.

④ Adresses de classe D :

Ces adresses sont réservées pour permettre la définition d'ensembles de machines dans le but de réaliser des envois à destinations multiples (notion de "multicast").

⑤ Adresses de classe E :

Martiens à venir . . .

Adresses réservées

Si vous voyez :	Cela signifie :
0.0.0.0	Comme adresse source : « Cet » ordinateur Comme adresse destination : Ce type d'adresse est utilisé par une machine qui ne connait pas son adresse IP (diskless). Elle n'est autorisée qu'au démarrage et ne constitue pas une adresse valide. (cf RARP)
255.255.255.255	Cette adresse désigne une diffusion limitée au réseau d'attachement (limité par les routeurs) (cf routed et rwho). Cette adresse n'est jamais retransmise par un routeur.
127.X.Y.Z	Ceux sont des adresses de rebouclage. Ces adresses sont destinées à permettre les communications interprocessus sur un même ordinateur. Ceci permet de réaliser des tests de logiciels TCP/IP. Ce trafic n'apparait JAMAIS sur le réseau ! Le cas échéant le routeur doit éliminer ce type d'adresse.
130.190.0.0	Adresse de réseau (id_ord = 0) désigne le réseau, de classe B en l'occurence.
130.190.255.255	Adresse de diffusion dirigée vers ce réseau. Le routeur va re-émettre cette trame vers le réseau/ss-réseau désigné.
0.0.X.Y	Adresse signifiant « un ordinateur sur ce réseau ». La machine ne connait pas l'adresse du réseau elle connait la classe.
Les adresses de classe A de 10.0.0.0 à 10.255.255.255, de classe B de 172.16.0.0 à 172.31.255.255 et de classe C de 192.168.0.0 à 192.168.255.255	Adresses réservées à la constitution de réseaux privés autrement appelés intranet.
0.x.x.x, 128.0.x.x, 191.255.x.x, 129.0.0.x, 223.255.255.x	Adresses réservées par le NIC (Network Information Center).

3.5 Adresses particulières

L'adressage IP permet quelques fonctions :

① 0.0.0.0 :

Ce type d'adresse est utilisé par une machine qui ne connait pas son adresse (diskless). Elle n'est autorisée qu'au démarrage et ne constitue pas une adresse valide. (cf RARP)

② 255.255.255.255 : désigne une **diffusion limitée** au réseau d'attachement (limité par les routeurs) (cf routed et rwho)

Cette adresse n'est jamais retransmise par un routeur.

③ 127.X.Y.Z : **loopback localhost**

Ceux sont des adresses de rebouclage. Ces adresses sont destinées à permettre les communications interprocessus sur un même ordinateur.

Ceci permet de réaliser des tests de logiciels TCP/IP.

Ce trafic n'apparait JAMAIS sur le réseau ! Le cas échéant le routour doit éliminer ce type d'adresse.

④ 130.190.0.0 : **adresse de réseau** (id_ord = 0)

désigne le réseau, de classe B en l'occurence.

⑤ 130.190.255.255 : **diffusion dirigée** vers ce réseau

⑥ 0.0.X.Y : un ordinateur sur ce réseau. La machine ne connait pas l'adresse du réseau elle connait la classe.

⑦ Les adresses de classe A de 10.0.0.0 à 10.255.255.255, de classe B de 172.16.0.0 à 172.31.255.255 et de classe C de 192.168.0.0 à 192.168.255.255 sont réservées à la constitution de **réseaux privés autrement appelés intranet**.

3.6 Faiblesse de l'adressage IP

Le fait de coder l'information relative au réseau dans l'adresse présente certains inconvénients

① La **mobilité** :

L'ordinateur « portable » doit changer d'adresse chaque fois qu'il change de réseau. C'est donc un frein à la mobilité !

② **Le changement de classe** (lié à l'extension du réseau) :

le passage d'une classe C à une classe B nécessite de changer tous les numéros. C'est donc un frein à l'évolution

③ Le **choix des routes** se fait sur la base de l'identificateur réseau ce qui a plusieurs conséquences :

- Tout le trafic relayé vers un réseau particulier emprunte le même chemin (notamment dommage dans le cas multi-homed)
- Tous les types de trafic empruntent le même chemin sans considération de débit ou de délai des réseaux empruntés.
- Seul le dernier routeur sait si l'ordinateur destinaire existe ou est opérationnel. (Il faut donc prévoir un mécanisme de gestion des comptes rendus.

4 ARP : Address Resolution Protocol

Pour acheminer les datagrammes, le protocole IP utilise obligatoirement les services d'un réseau physique.

- ➢ Ces datagrammes sont encapsulés dans des trames physiques dont ils constituent la partie données.
- ➢ Au niveau des couches inférieures à IP, c'est à dire des couches liaison de données et physique, **l'acheminement suppose donc la connaissance de l'adresse physique du destinataire**.

Dit autrement, il faut que l'adresse physique du destinataire figure dans le datagramme IP que l'émetteur souhaite transmettre :

> Il est donc indispensable de disposer d'un mécanisme permettant de convertir une adresse logique (INTERNET) en l'adresse physique (ETHERNET ou Token Ring ou ...) correspondante.

Problème :

La variété des formats d'adresses physiques exclut la possibilité de définir mathématiquement la fonction de conversion correspondante et il est nécessaire de disposer d'un **mécanisme dynamique** permettant cette résolution.

- ➢ C'est l'objet du protocole **ARP (Address Resolution Protocol)** de réaliser cette association en masquant les mécanismes d'adressage propres au réseau physique utilisé.

On peut considérer que **ce protocole ne fait pas partie intégrante** de la suite des protocoles INTERNET et constitue **une interface entre cette suite et les couches plus basses**,

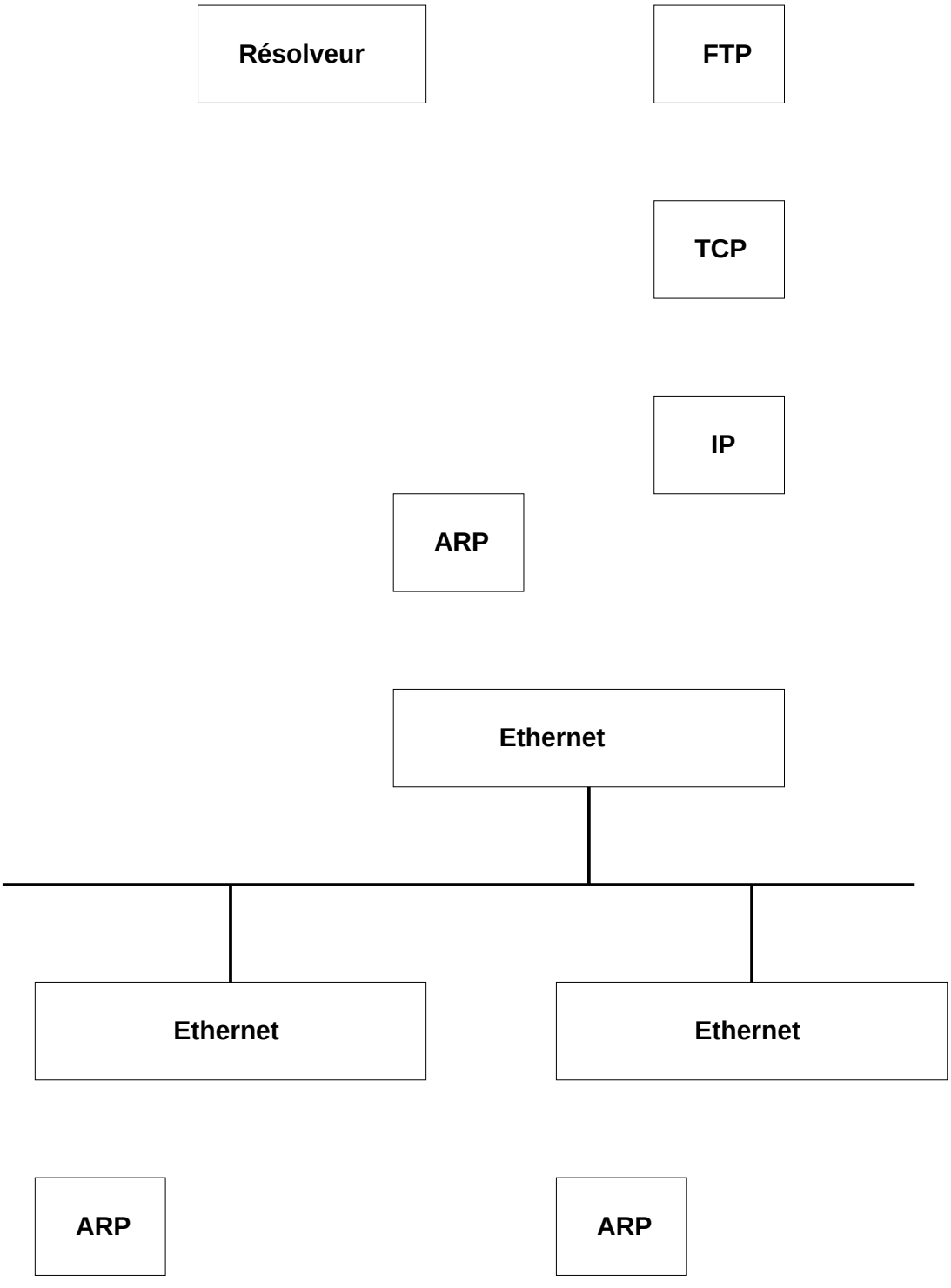
- ➢ mais il joue néanmoins un rôle fondamental.

4.1 La solution

Solution :

La solution choisie consiste à faire connaître à tous (tous les hôtes présents sur le réseau physique) l'adresse INTERNET recherchée et celui qui y reconnait son adresse renvoit son identité ETHERNET.

Service ARP


Résolveur
FTP
TCP
IP
ARP
Ethernet
Ethernet
Ethernet
ARP
ARP

4.2 Service ARP

Nous nous plaçons dans le cas d'une correspondance à établir entre IP et Ethernet et la nécessité de la résolution d'adresse fournie par ARP apparaît dans l'exemple ci-dessous **décrivant le début d'une connexion FTP** (p55 [5]).

- Résolution de nom :

  Le client FTP convertit l'adresse du serveur FTP (ex : eea.unice.fr) en une adresse IP (134.59.3.111) à l'aide du fichiers /etc/hosts ou d'un serveur de noms (DNS).

- Demande de connexion :

  Le client FTP demande à la couche TCP d'établir une connexion avec cette adresse.

  Le logiciel TCP envoie une requête de connexion à ce serveur en émettant un datagramme IP contenant l'adresse IP

- Résolution d'adresse physique :

  En supposant que les machines client et serveur sont sur le même réseau local Ethernet, la **machine émettrice doit convertir l'adresse IP** sur 4 octets en une adresse Ethernet sur 6 octets avant d'émettre la trame Ethernet contenant le paquet IP. C'est ce que va faire ARP.

  ① Le module ARP envoie une requête ARP dans une trame Ethernet avec une adresse de destination multicast. Ainsi, toutes les machines du réseau local reçoivent cette requête contenant l'adresse IP à résoudre.

  ② La couche ARP de la machine visée (ici eea.unice.fr) reconnaît que cette requête lui est destinée et répond par une réponse ARP contenant son adresse matérielle 00 :20 :AF :AB :42 :43. Les autres machines du réseau ignorent la requête.

  ③ La réponse ARP est reçue par l'émetteur de la requête. Pour ce retour, il n'y a pas de problème de résolution puisque l'adresse physique de l'émetteur étant envoyée dans la requête elle est connue de la machine qui répond.

  ④ La réponse ARP est reçue par la couche ARP du serveur FTP, et le driver Ethernet peut alors émettre le paquet IP avec la bonne adresse Ethernet de destination.

- Emission :

  Le datagramme IP est envoyé à la machine destination.

La trame ARP

16 bits	type de réseau physique (1 pour ETHERNET)
16 bits	type de réseau logique (0800_{16} si IP)
8 bits	longueur de l'adresse physique (6 si Ethernet)
8 bits	longueur de l'adresse logique (4 si IP)
16 bits	type de l'opération
	Requête ARP (1) Réponse ARP (2) Requête RARP (3) Réponse RARP (4)
48 bits	adresse physique de l'émetteur
32 bits	adresse logique de l'émetteur
48 bits	adresse physique du destinataire
32 bits	adresse logique du destinataire

4.3 La trame ARP

- ➢ Dans la trame Ethernet encapsulant le datagramme ARP :
  - Les deux premiers champs d'une trame Ethernet émise par ARP sont conformes à l'en-tête d'une trame Ethernet habituelle et **l'adresse de destination sera ff :ff :ff :ff :ff :ff**, l'adresse multicast désignant toutes les machines du réseau à la fois.
  - La valeur du champ type de trame est 0806_{16} indiquant le protocole ARP.
- ➢ Le champ **type de matériel** est égal à 1 pour un réseau Ethernet
- ➢ et celui **type de protocole** est égal est 0x800 pour IP.
- ➢ Les **tailles en octets** spécifiées ensuite sont 6 (6 octets pour une adresse Ethernet) et 4 (4 octets pour une adresse IP).
- ➢ Le champ **op** vaut :
  - 1 pour une requête ARP et
  - 2 pour une réponse ARP.
- ➢ Les quatre champs suivants contiennent des adresses
  - l'adresse Ethernet émetteur : Lors de la requête ARP, elle est **redondante** puisqu'elle est déjà dans la trame Ethernet.
  - l'adresse Ethernet cible : Lors d'une requête ARP, elle est **non renseignée** puisque c'est ce que l'on cherche.
  - La machine qui reconnaît son numéro IP à l'intérieur d'une requête ARP qu'elle reçoit la renvoie en y intervertissant
    - ➢ les adresses IP cible et émetteur,
    - ➢ ainsi que les adresses Ethernet cible (après l'avoir substituée à l'adresse de diffusion dans l'en-tête et renseignée dans le corps de la trame) et Ethernet émetteur.

4.4 Le cache ARP

Pour éviter la multiplication des requêtes ARP, chaque machine gère un cache dans lequel elle mémorise les correspondances adresses IP/adresses Ethernet déjà résolues préalablement.

- ➢ Ainsi, le module ARP ne lancera une requête (broadcast) que lorsqu'il ne trouvera pas cette correspondance dans le cache,
- ➢ sinon il se contentera d'émettre les données qu'il reçoit d'IP en ayant fixer correctement l'adresse physique de destination.

Cependant, les correspondances ne sont pas conservées indéfiniment car cela pourrait provoquer des erreurs lorsque l'on change un ordinateur (ou une carte réseau) sur le réseau en conservant un même numéro IP pour cet ordinateur mais évidemment pas la même adresse physique.

- ➢ La durée de validité d'une entrée est par défaut limitée à 20 minutes.

4.5 Le protocole RARP

Définition :

Quant à lui, le protocole RARP joue le rôle inverse de ARP **en permettant de déterminer l'adresse IP d'un équipement dont on connaît l'adresse physique**.

➢ Ceci est notamment utile pour amorcer une station sans disques, ou un TX, qui n'a pas en mémoire son adresse IP mais seulement son adresse matérielle.

Le format d'une trame RARP est identique à celui d'une trame ARP où :

- le champ type de trame vaut 0x8035 et
- le champ op vaut 3 pour une requête RARP
- et 4 pour une réponse.

Une requête RARP est diffusée sous forme de broadcast,

➢ donc toutes les machines du réseau la reçoivent et la traitent.

➢ Mais la plupart des machines ignorent simplement cette demande,

➢ seuls, le ou les **serveurs RARP** du réseau vont traiter la requête grâce à un ou plusieurs fichiers et vont retourner une réponse contenant l'adresse IP demandée.

4.6 Le protocole Proxy ARP

Définition :

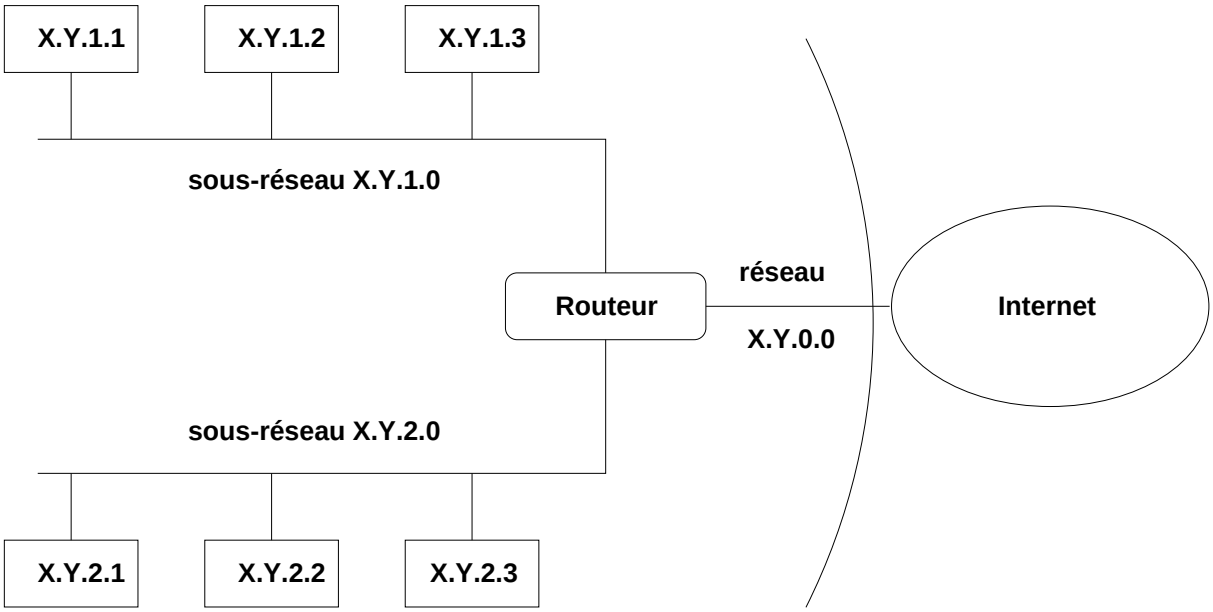
Le protocole **Proxy ARP** permet de **réaliser cette association** (adresse physique/adresse logique) pour une **machine n'appartenant pas au même réseau physique**.

➢ Il suppose la propagation de la requête via les routeurs, lequel fournira son adresse logique.

D'un point de vue de la requète de l'émetteur, tout se passe comme si le routeur était la machine de destination.

On va tout de suite voir à quoi ca sert dans le cadre du sous adressage.

Sous-Réseaux et Sous-Adressage



- Le sous adressage est un moyen de « couper » un gros réseau en plus petits réseaux, plus rapides et plus facilement gérables.
  1. autonomie de gestion et maximisation de l'utilisation des adresses (utilisant le principe de classes).
  2. gestion facilitée (évolution, sécurité, . . .).
  3. réduction de la congestion (séparation des trafics)

4.7 Notion de sous-réseaux

4.7.1 Motivation

La vue originelle de l'univers d'Internet est celle d'une hiérarchie à deux niveaux :

① the « top level » : Internet comme un tout.

➢ Le réseau peut être traité comme un boîte noire à laquelle un ensemble d'hôte sont connectés.

② le « niveau au dessous » : les réseaux individuels, **chacun** avec son propre numéro de réseau.

L'Internet **n'a pas de topologie hiérarchique** (même si l'interprétation des adresses est hiérarchique) :

➢ il n'y a pas de réseau dans un réseau !

Dans ce modèle à deux niveaux, chaque hôte voit **son** réseau comme une entité unique :

➢ id-res ne désigne qu'**un** réseau !

Bien que cette vision ait prouvé sa simplicité et son efficacité, un certain nombre d'organisations l'ont trouvé inadéquate et ont **ajouté un troisième niveau** dans l'interprétation de l'adresse Internet.

Dans cette vision, un réseau Internet est divisé en un ensemble de sous-réseaux.

Le modèle a trois niveaux, où l'on introduit **les réseaux dans les réseaux**, est utile pour des réseaux appartenant à des organisations modérément grosses (e.g., Universities or companies with more than one building), où il y a souvent **besoin de plus d'un cable LAN** pour couvrir « la zone ».

➢ Chaque LAN peut alors être traités comme un **sous réseau**.

Plus d'un réseau physique . . . :

Les motivations, d'une organisation, pour utiliser plus d'un cable LAN pour couvrir un campus sont nombreuses :

1. Utilisations de **technologies LAN différentes** :
   Une organisation peut avoir différents types d'équipements : Ethernet, Token Ring
2. **Limites « physiques »** de ces technologies :
   Most **LAN technologies impose limits**, based on electrical parameters, on the number of hosts connected, and on the total length of the cable.

   ➢ Il est facile de dépasser ces limites, spécialement celles sur la longueur des cables.

Parfois, un même domaine (par exemple un campus universitaire) est constitué de deux localisations trop éloignées pour utiliser un technologie de connexion de type LAN.

➢ Dans ce cas, des connexions rapide point-à-point permettent de connecter ces différents LANs.

3. Résolution de **congestion réseau** :
   Lorsque la bande passante d'un réseau est limitée, et c'est toujours le cas, on a toujours intérét à regrouper les machines qui parlent beaucoup entre elles de façon à ce qu'elles ne monopolisent pas la bande passante.

   ➢ En définissant des sous-réseaux, chacun séparés par des « passerelles », on évite le mélange des flux et le dialogue « local » ne va pas polluer le reste du réseau.

4. **Administration** :
   Imaginez une companie qui souhaite que ses adresses IP soient administrées localement.
   Chacun de ses départements (engineering, marketing, sales) se verrait attribuer un morceau du gateau global et l' administrateur du département serait responsable de sa zone d'adresse.

   ➢ Au passage, morceller un réseau, c'est lorsqu'il y a un problème par exemple de routage, pouvoir localiser plus facilement la cause.

   ➢ C'est aussi un gain de sécurité. Les trafics ne se mélangent plus !

Politiques de gestion des adresses logiques :

Une organisation, qui a été forcé d'utiliser plus d'un LAN, a trois choix pour affecter ces adresses Internet :

① Acquérir un numéro de réseau Internet distinct pour chaque cable :

➢ Il y a **Non-utilisation des sous-réseaux**.

② Utiliser un unique numéro de réseau pour toute l'organisation et affecté les numéros IP sans regarder sur quel LAN se trouve la machine :

➢ **Les sous réseaux sont transparents**.

③ Utiliser un numéro de réseau unique et **partitionner l'espace des adresses** en affectant des numéros de sous réseaux aux différents LAN :

➢ On parle de **sous réseaux explicites**.

Analyse des politiques de gestion :

Chacune de ces trois approches a ses désavantages :

① **Non-utilisation des sous-réseaux**

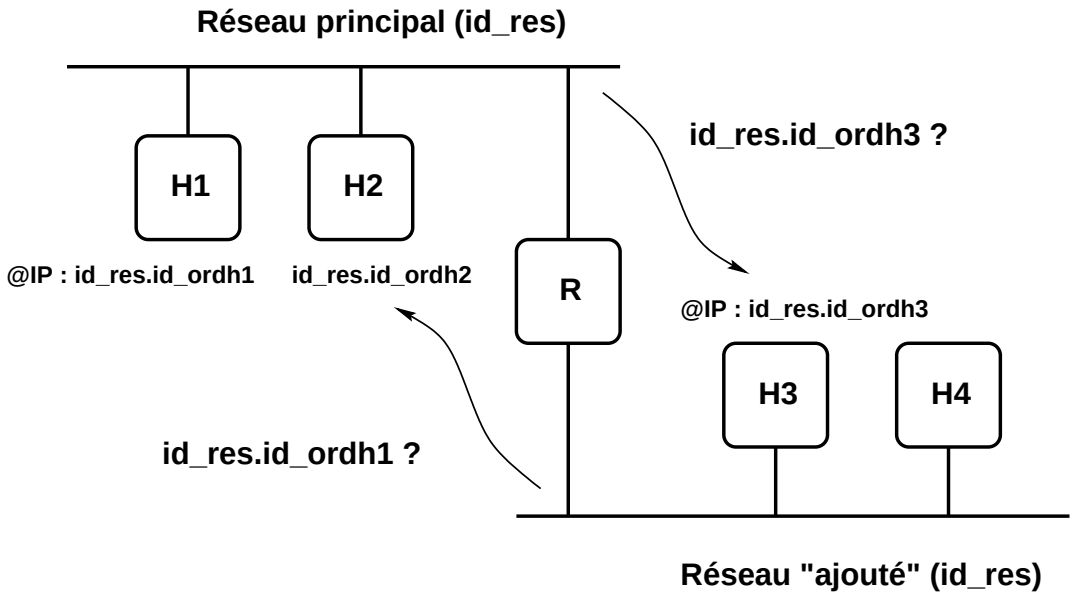
Bien que ne nécéssitant pas de nouveaux protocoles, ni de modification de procotoles déjà existants, cette approche provoque l'**explosion de la taille des tables de routage Internet**.

➢ L'information sur les détails internes de la connectivité locale à l'organisation est propagée sur tout Internet .. c'est pas super !

② **Sous réseaux transparents** :

La seconde approche requiert quelques conventions ou protocoles qui fasse **apparaître un ensemble de LANs comme un réseau Internet unique**.

Ceci est le cas du schéma qui suit :



Les LANs sont connectés entre eux par un routeur.

- Ce routeur doit permettre **de partager une adresse réseau (id_res)**, fournie par le NIC, **entre deux sous réseaux**.
- Néanmoins, ces sous réseaux ne sont pas identifier pas un id_res différent.

En utilisant une technique **proxy ARP** ,

- le routeur répond, sur chaque sous-réseau, aux requêtes destinées à des ordinateurs situés sur l'autre sous réseau en indiquant sa propre adresse physique.
- R route ensuite correctement les paquets qu'il reçoit.

R ment donc au sujet de l'association entre les adresses physiques et IP. Il fait comme si toutes les machines étaient sur le même réseau.

- see RFC-925 [2] et 10.5 dans [12].

Cependant, cette solution n'est pas toujours implémentable : il faut que la technologie LAN supporte ARP et tout particulièrment les broadcast.

Un problème plus fondamental, dans ce cas de figure, est que les routeurs doivent découvrir sur quel LAN est un hôte.

- Il va, pour se faire, utiliser un algorithme à base de broadcast.
- Comme le nombre de LANs croît, le coût du broadcasting croît.

Enfin la taille des caches de translation dans les routeurs croît avec le nombre total d'hôtes dans le réseau.

③ **Sous réseaux explicites** :
L'utilisation des sous réseaux explicites nécéssite une modification, certes mineure, de IP et notamment de l'algorithme de routage (10.11 [12]) qui doit savoir gérer des masques de sous-réseaux..

➢ Mais on garde une compatibilité ascendante : si ca marche avec les sous-réseaux, ca marche sans.

4.7.2 Le sous-adressage explicite

Ces sous-réseaux sont un moyen de couper un gros réseau en plus petits réseaux, plus rapides et plus facilement gérables.

➢ En **local**, un même réseau (id_res) va donner naissance à plusieurs réseaux (id_res1, id_res2, id_res3, . . .).

➢ Pour le reste du monde, il reste (id_res).

Définition :

La définition d'adresses de **sous-réseaux** est obtenue grâce aux système des adresses IP.

On a vu qu'IP développe les 3 classes A, B, C d'adresses.

➢ Chaque classe définit un morcellement différents entre la partie réseau et la partie ordinateur d'une adresse IP.

La création de sous-réseaux est obtenue **en découpant la partie réservée à l'adresse des machines** sur un réseau en deux parties dont la première sera un identificateur de sous-réseau.

➢ Ainsi un seul réseau de classe B, sur lequel on pourrait nommer 65 536 machines pourra être décomposé en 254 sous-réseaux de 254 machines, de la manière décrite ci-dessous :

<id_rés sur 16 bits>.<id. de sous-réseau sur 8 bits>.<id_ord sur 8 bits>

➢ L'administrateur d'un réseau peut décider de découper où il veut la zone des identificateurs de machines, mais le découpage autour du « . » facilite le travail des routeurs.

On peut également adopter le même principe pour un réseau de classe C.

Cette technique a pour effet de provoquer un routage hiérarchique sur 3 niveaux (network, subnetwork, host).

Définition :

These are **classless addresses** .

La figure illustre :

- le cas d'un réseau X.Y.0.0 découpé en deux
- sous-réseaux X.Y.1.0 et X.Y.2.0.

Routage :

D'un point de vue du routage, et sans tenir compte du sous adressage, on sait que :

① **Quand un paquet n'est pas destiné au réseau local**, toutes les décisions de routage pour ce paquet seront basées sur la portion « réseau » de l'adresse.

② Inversement, **tous les ordinateurs qui partagent un numéro de réseau** peuvent communiquer (broadcast compris) entre eux sans nécessiter un routeur.

➢ Par exemple, l'interface d'adresse IP 207.86.28.48 peut théoriquement communiquer, et sans routeur, avec toute machine dans le réseau 207.86.28.X (classe C).

De la même façon, **en utilisant le sous-adressage**, pour tout le reste d'Internet,

➢ il n'existe qu'un seul réseau X.Y.0.0 et tous les routeurs traitent les datagrammes à destination de ce réseau de la même façon.

➢ Par contre, le routeur R (interne à l'organisation) se sert du troisième octet (égal à 1 ou 2) de l'adresse contenue dans les datagrammes qui lui proviennent pour les diriger vers le sous-réseau auquel ils sont destinés assurant ainsi un routage hiérarchique.

Définition :

Outre son'adresse IP, une machine doit également connaître le nombre de bits attribués à l'identificateur du sous-réseau et à celui de la machine afin de savoir **calculer l'adresse réseau** (tenant compte de la notion de sous réseau).

Cette information est rendue disponible grâce à un **masque de sous-réseau** ou **subnet netmask** qui est un mot de 32 bits contenant

➢ des bits à 1 au lieu et place de **l'identificateur de réseau et de sous-réseau**

➢ et des bits à 0 au lieu et place de **l'identificateur de machines**.

- Ainsi le masque 255.255.255.0 indique que les 24 premiers bits d'une adresse désignent le sous-réseau et les 8 derniers une machine.
- Le masque 255.255.255.192 ((192) 10 = (11000000) 2) indique que les 26 premiers bits désignent le sous-réseau et les 6 derniers une machine.

De cette manière **à partir de l'adresse d'un datagramme et de son masque de sous-réseau**, une machine peut déterminer si le datagramme est destiné :

① à une machine sur son propre sous-réseau,

② à une machine sur un autre sous-réseau de son réseau,

③ ou à une machine extérieure à son sous-réseau.

Par exemple, dans le cadre du réseau de la figure où le masque de sous-réseau est 255.255.255.0 et en supposant que notre machine soit celle identifiée par l'adresse IP X.Y.1.2 (elle est sur le sous-réseau X.Y.1.0) et qu'elle souhaite émettre un datagramme :

① Si l'adresse de destination est X.Y.1.1, un « et » entre la représentation binaire de cette adresse et celle du masque de sous-réseau donne X.Y.1.0 à savoir l'adresse du sous-réseau de notre machine,

➢ donc le **datagramme est destiné à une machine de ce même sous-réseau**.

② Si l'adresse de destination est X.Y.2.1, un calcul du même genre donne X.Y.2.0

➢ c'est-à-dire **l'adresse d'un autre sous-réseau du même réseau**.

③ Si l'adresse de destination est S.T.U.V (avec S.T $!=$ X.Y) le résultat sera

➢ **l'adresse d'un réseau différent de celui auquel appartient notre machine**.

4.7.3 Netmasks en Unix

La commande ifconfig permet d'assigner un masque de sous-réseau (netmask) is à une interface. Lorsque l'interface est configurée, elle possède une adresse et un masque. Par exemple :

```
ifconfig le0 inet 134.59.131.23 netmask 255.255.255.0.
```

4.7.4 Sous-réseaux : règles de constitution

Il a des règles pour constituer des sous-réseaux :

- Adressage/subnet.ps ....
- mini-howto linux
- ....

Adressage/subnet.ps ....

4.7.5 Sous-réseaux : Avantages

On retiendra les deux avantages majeurs de cette approche :

① Méthode efficace de découpage de réseau :

Un des problèmes majeurs des classes d'adresse IP est le manque d'une classe permettant de gérer une organisation de taille moyenne pour laquelle la classe B est trop grosse (65534 hosts) et la classe C trop petite (254 hosts).

➢ On va, avec la notion de sous-réseaux pouvoir « découper », des réseaux irréalistes comme par exemple les classes A.

② Masquage de la structure interne :

De plus, toujours dans l'esprit d'une « malformation » initiale du système d'adressage d'Internet.

Lorsqu'une organisation reçoit un réseau de l'IANA (Internet Assigned Numbers Authority) via l'InterNic, elle peut l'architecturer comme elle le souhaite par exemple en constituant différents sous-réseaux correspondants à des réseaux physiques différents (TM, FDDI, ...) . Et ceci avec un même numéro de réseau !

➢ La notion de sous-réseau est donc un moyen de **cacher** la topologie locale au reste du monde.

Vis à vis du reste du monde, et de son routage, l'organisation est géré comme un réseau unique.

Les tables de routages externes n'ont donc pas à tenir compte des différents réseaux.

L'opération de routage en est donc facilitée.

4.7.6 Sous-réseaux : Evolution du concept d'adresse

Le concept de classe d'adresse évolue, sous la pression de l'approche de sous-réseaux.

➢ Cette évolution vise à engendrer une nouvelle forme de spécification du nombre de bits réservés à une adresse réseau.

Cette nouvelle terminologie se réferre à une adresse IP suivie par le nombre de bits réservés pour le numéro réseau.

Par exemple :

207.215.121.0/24

est la formalisation de qui, avant, faisait référence à une adresse de Class C.

Définition :

Cette méthode est une généralisation de l'approche des sous-masques réseaux.

➢ Les nouveaux routeurs peuvent reconnaître cette notation : ils sont alors capables de réaliser une routage **classless** (cf routage IP).

5 Le protocole Internet (IP) RFC 791

Un utilisateur percoit un Internet comme un réseau virtuel unique et homogène qui interconnecte de nombreux ordinateurs et sur lequel ceux-ci peuvent communiquer :

➢ l'architecture réelle d'un Internet est masquée.

Définition :

C'est le protocole de couche réseau **IP** qui est au coeur du fonctionnement d'un Internet :

Il gère les communications de **machine à machine**.

5.1 Les services offerts

L'objectif est de permettre la connexion d'un grand nombre de réseaux.

- Un protocole d'une couche supérieure, inter–agit avec IP grâce à deux primitives :
  1. SEND : pour transmettre à IP ce que l'on souhaite émettre et spécifiant les caractéristiques (via les paramètres de SEND) de l'émission.
  2. et DELIVER : pour récupérer depuis IP l'information en provenance d'un hôte distant. Là encore, des paramètres permettent de particulariser les réceptions.

  Le service de transmission fourni n'est pas trés **fiable** : les datagrammes peuvent être perdus, dupliqués, retardés, altérés ou remis dans la désordre.

  ➢ On parle de **remise au mieux** (**best effort delivery**) et ni l'émetteur ni le récepteur ne sont informés directement par IP des problèmes rencontrés.

- Les unités de données sur lesquelles travaille IP sont appelées les **IP datagram** .
  Les **IP datagrams** réalisent une **encapsulation** des données en provenance de la couche supérieure avec un **IP header** .
  IP fournit une communication de type "connectionless" ou "datagram".

  ➢ Chaque datagramme est traité indépendamment des autres. Ainsi en théorie, au moins, deux datagrammes IP issus de la même machine et ayant la même destination peuvent ne pas suivre obligatoirement le même chemin.

- IP fournit **une durée de vie garantie** pour tous les paquets qu'il transmet.
  Ceci implique pour l'utilisateur que si un datagramme IP n'est pas livré avant un certain laps de temps, on peut en déduire qu'il est perdu.
  Ce paramètre (**'time–to–live period**) est un des arguments de l'opération de SEND.
- IP permet, enfin, de **gérer des messages ICMP d'erreur et de supervision**.

5.2 Fonctionnalités IP

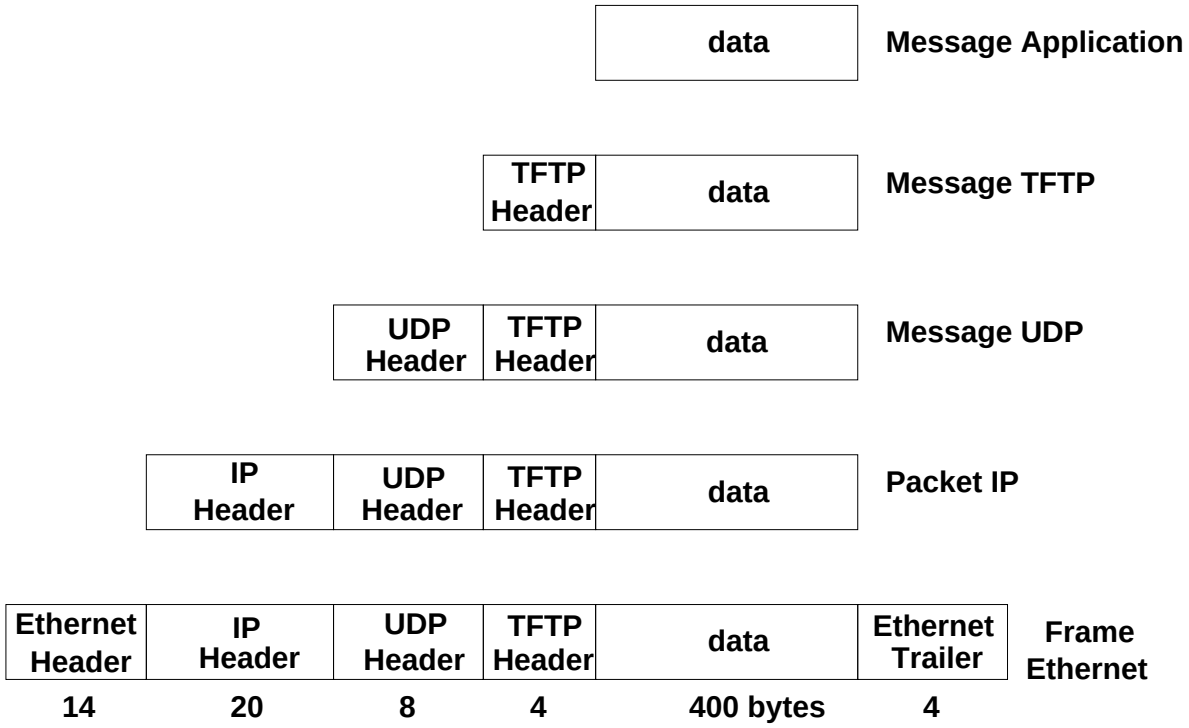
Le rôle du protocole IP est centré autour des trois fonctionnalités suivantes chacune étant décrite dans une des sous-sections à venir.

① Définir le format du datagramme IP qui est l'unité de base des données circulant sur Internet.

② Définir le routage dans Internet.

③ Définir la gestion de la remise non fiable des datagrammes (cf ToS).

Encapsulation

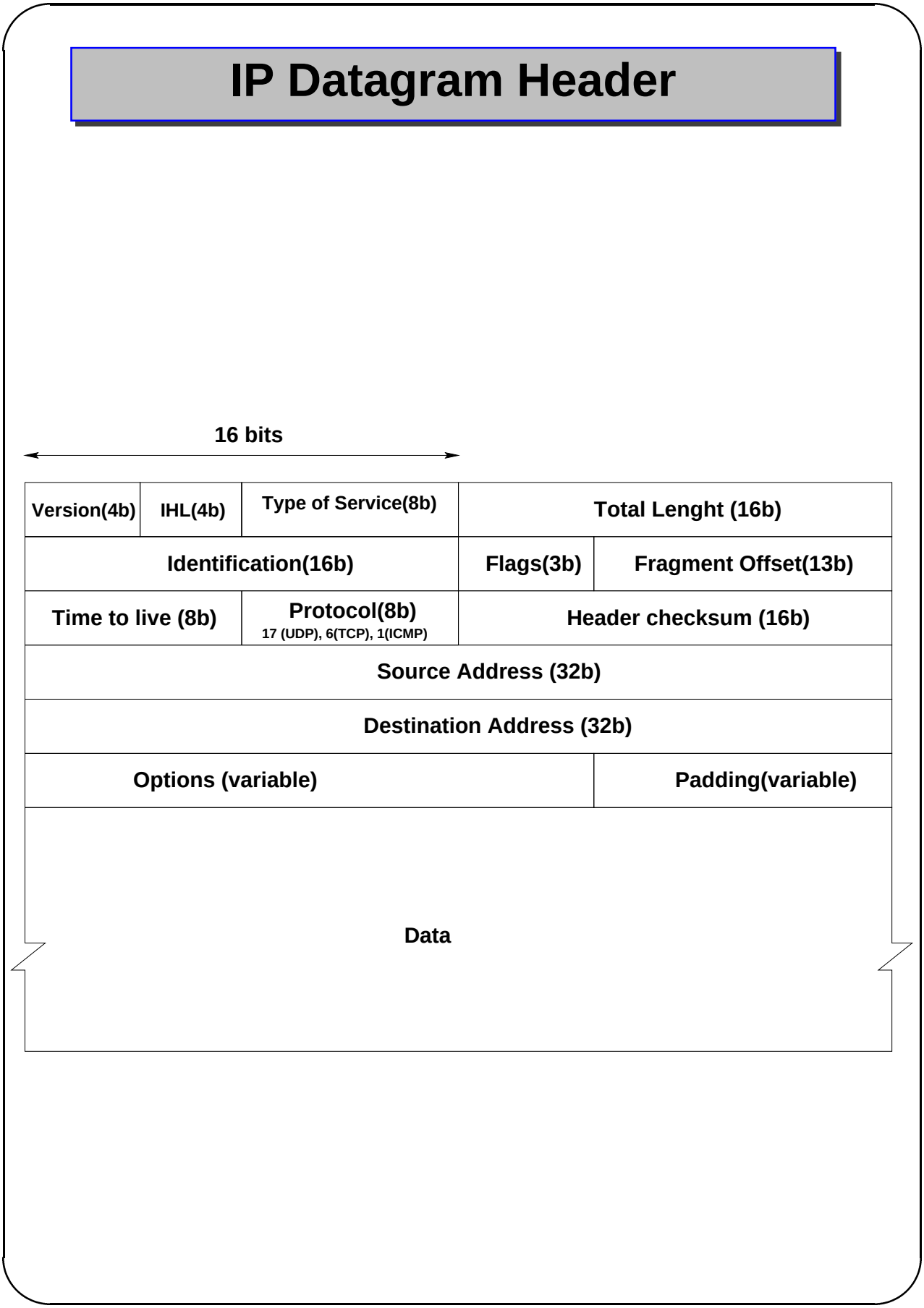


5.3 Encapsulation et Unités de données

Supposons que l'on regarde les unités de données manipulées par chaque couche dans le cadre d'une application TFTP.

- On s'apercoit que chaque couche rajoute, encapsule, l'information qu'elle doit gérer avant de le transmettre à la couche suivante.
- Ceci dans le but que l'entité correspondante puisse à son tour gérer les données.
- C'est une des concrétisations du protocole.

IP Datagram Header



5.4 Format d'un datagramme IP

Le datagramme constitue l'unité d'information échangée par des entités IP et constitue la partie données d'une trame physique.

On y retrouve les données nécessaires à sa gestion :

1. **Version** : indique la version d'IP qui a généré cet entête.

   Actuellement c'est 4, d'où son nom IPv4.

   Tout logiciel IP doit d'abord vérifier que le numéro de version du datagramme qu'il reçoit est en accord avec lui-même. si ce n'est pas le cas le datagramme est tout simplement rejeté.

   Ceci permet de tester des nouveaux protocoles sans interférer avec la bonne marche du réseau.

2. **Internet Header Lenght** (IHL) : longueur de l'entête en nombre de mots de 32 bits.

   C'est au moins 5 mais ca peut varier à cause du champ "option".

3. **Type of Service** (TOS) : permet de spécifier le type de service souhaité qui sera utilisé au bon vouloir des "routers".

   Cela comporte des indicateurs

   - ➢ d'importance du datagramme,
   - ➢ de délivrance rapide,
   - ➢ de débit important,
   - ➢ de demande de fiabilité.

   Les 8 bits du TOS se répartissent comme suit :

0 1 2	3	4	5	6	7
priorité	D	T	R	C	inutilisé

① Le champ priorité varie de 0 (priorité normale, valeur par défaut) à 7 (priorité maximale pour la supervision du réseau) et permet d'indiquer l'importance de chaque datagramme.

Même si ce champ n'est pas pris en compte par tous les routeurs, il permettrait d'envisager des méthodes de contrôle de congestion du réseau qui ne soient pas affectées par le problème qu'elles cherchent à résoudre.

② Les 4 bits D, T, R et C permettent de spécifier ce que l'on veut privilégier pour la transmission de ce datagramme (nouvel RFC 1455).

- D est mis à 1 pour essayer de minimiser le délai d'acheminement (par exemple choisir un câble sous-marin plutôt qu'une liaison satellite),
- T est mis à 1 pour maximiser le débit de transmission,
- R est mis à 1 pour assurer une plus grande fiabilité et
- C est mis à 1 pour minimiser les coûts de transmission.

Si les quatre bits sont à 1, alors c'est la sécurité de la transmission qui doit être maximisée. Les valeurs recommandées pour ces 4 bits sont données dans la table qui suit.

Ces 4 bits servent à améliorer la qualité du routage et ne sont pas des exigences incontournables. Simplement, si un routeur connaît plusieurs voies de sortie pour une même destination il pourra choisir celle qui correspond le mieux à la demande.

application	minimise le délai	maximise le débit	maximise la fiabilité	minimise le coût
telnet/rlogin	1	0	0	0
FTP				
contrôle	1	0	0	0
transfert	0	1	0	0
SMTP				
commandes	1	0	0	0
données	0	1	0	0
NNTP	0	0	0	1

4. **Total length** : spécifie la longueur (en octets) du datagramme IP incluant l'entête.

   Comme ce champ est de 2 octets on en déduit que la taille complète d'un datagramme ne peut dépasser 65535 octets.

   Utilisée avec la longueur de l'en-tête elle permet de déterminer où commencent exactement les données transportées.

5. **Flags** : contient les indicateurs "More" et "Do not fragment"

   Si le datagramme doit être fragmenté en contradiction avec cet indicateur, le datagramme est perdu.

6. **Fragment Offset** : offset du fragment.

   C'est la distance en unités de 64 bits depuis le premier bit du datagramme.

7. **Checksum** : contient un code détecteur d'erreur **qui ne s'applique qu'à l'entête**.

   - ➢ L'objectif est de se pémunir contre les "mauvaises manipulations" des routeurs.
   - ➢ L'intègrité des données est supposée assurée par les protocoles inférieurs (CSMA/CD) et supérieurs ICMP, IGMP, TCP et UDP.

   Le calcul est illustré juste aprés !

   Pour calculer cette somme de contrôle,

   ① on commence par la mettre à zéro.

   ② Puis, en considérant la totalité de l'en-tête comme une suite d'entiers de 16 bits, on fait la somme de ces entiers en complément à 1.

   ③ On complémente à 1 cette somme et cela donne le total de contrôle que l'on insère dans le champ prévu.

   A la réception du datagramme, il suffit d'additionner tous les nombres de l'en-tête (cheksum compris) et si l'on obtient un nombre avec tous ses bits à 1, c'est que la transmission s'est passée sans problème.

8. **Source Address** :

9. **Destination Address** : contient le code binaire de l'adresse Internet des correspondants.

   Les 32 bits sont aujourd'hui insuffisants, d'où des propositions d'évolution.

10. **Time to live** : contient la durée de vie restante.

    En fait, cette durée initialement exprimée en secondes, est devenue, compte tenu des performances des routeurs (traitant un paquet en moins d'une seconde), le nombre maximal de routeurs que peut traverser le datagramme.

    Elle est initialisée à N (souvent 32 ou 64) par la station émettrice et décrémenté de 1 par chaque routeur qui le reçoit et le réexpédie.

    Lorsqu'un routeur reçoit un datagramme dont la durée de vie est nulle, il le détruit et envoie à l'expéditeur un message ICMP.

    Ainsi, **il est impossible qu'un datagramme tourne indéfiniment dans un internet**.

11. **Protocol** : contient l'identification du protocole client.

    - 17 pour UDP,
    - 6 pour TCP,
    - 1 pour ICMP ...

    Ainsi, la station destinatrice qui reçoit un datagramme IP pourra diriger les données qu'il contient vers le protocole adéquat : c'est le **démultiplexage**.

12. **Options** : spécifie des options telles que la sécurité, l'enregistrement de la route, l'estampille horaire, routage strict, etc..

13. **Padding** : Pour être sûr que l'entête occupe un nombre entier de mots de 32 bits.

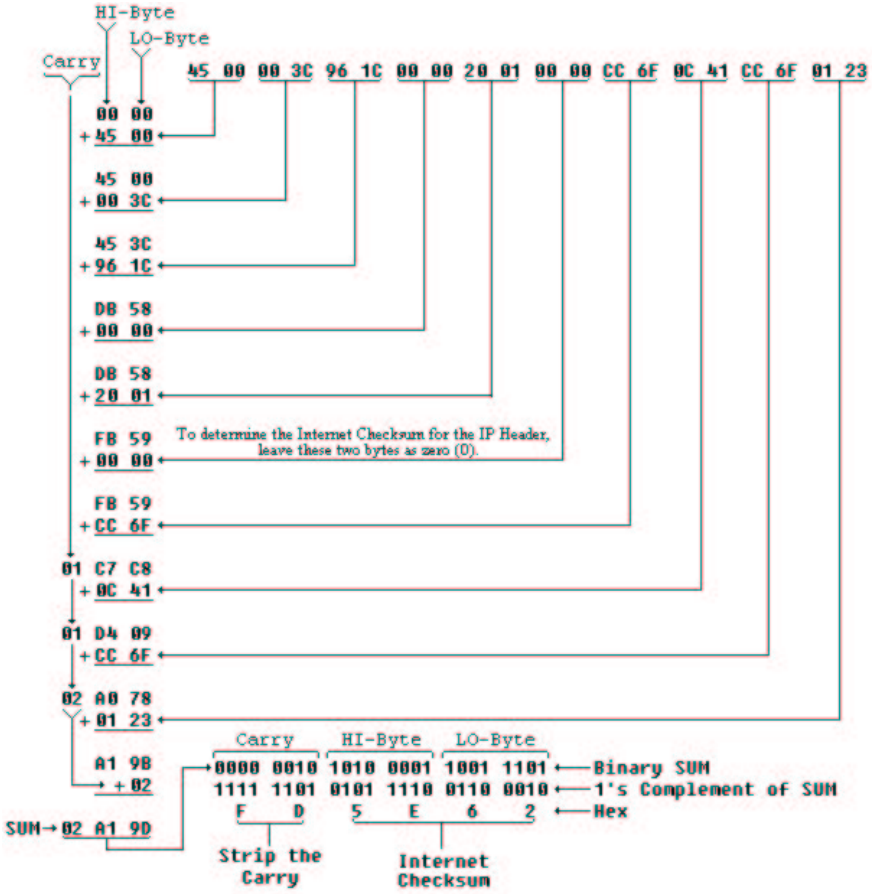
Calcul du Checksum pour IP

Sample IP Header

45	IP Version & IP Header Length
00	Type Of Service
00 3C	Total Packet Length
96 1C	Identification
00 00	Flags & Fragmentation Offset
20	Time To Live (TTL)
01	Protocol (ICMP)
00 00	Header Checksum (To Be Determined)
CC 6F 0C 41	Source IP Address (204.111.12.65)
CC 6F 01 23	Destination IP Address (204.111.1.35)

45 00 00 3C 96 1C 00 00 20 01 00 00 CC 6F 01 23 CC 6F 01 23

IP Header Represented as an Array of Bytes

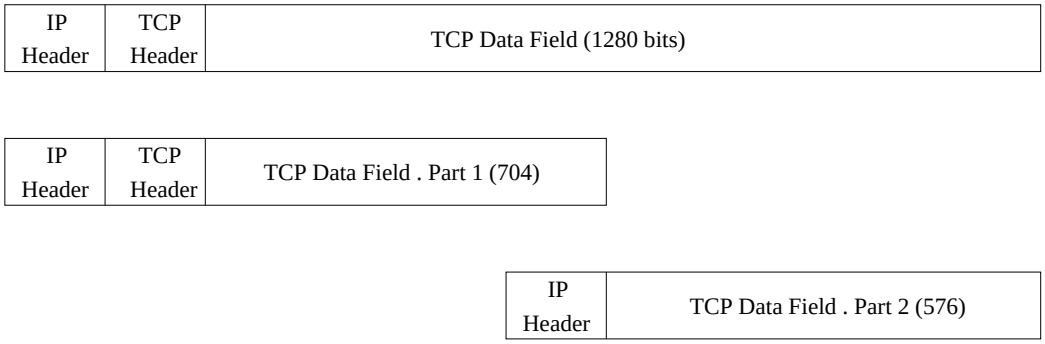


5.4.1 Calcul du checksum

Le calcul du checksum est une opération fréquente en TCP/IP.

- Les octets adjacents sont regroupés pour former des entiers de 16 bits.
  La somme va être calculée sur les octets concernés. Si le nombre d'octets était impair on sait aussi s'en sortir ...
- Le checksum est initialisé à 0.
- Les entiers de 16 bits sont additionnés ensembles (ainsi que le carry)
- et le complément à 1 de la somme constitue le **checksum field** .

Fragmentation IP



Field IP Header	Original Datagram	Fragment 1	Fragment 2
Source	I_A	I_A	I_A
Destination	I_B	I_B	I_B
Protocol	TCP	TCP	TCP
Identif.	225	225	225
More Flag	0	1	0
Offset	0	0	11

5.5 Fragmentation and Reassembly

Une des différences entre les différents standards réseaux est la taille maximum d'un paquet (**MTU : Maximum Transfer Unit**) dont la valeur varie suivant la nature du réseau physique :

type de réseau	valeur de la MTU
Ethernet	1500
Token Ring	4464 ou 17914
Token Bus	8182
Aloha	80
X25	de 16 à 576
FDDI	4352

Plutôt que d'imposer une taille qui pourrait en pénaliser certains(CSMA/CD si trop gros, encapsulation peu rentable si trop petits), parce que trop petite ou trop grande, le protocole IP **se charge de fragmenter les données** afin d'adapter leur taille aux contingences du réseau physique utilisé.

➢ En fait c'est les routeurs qui réalisent cette opération et qui fragmentent les "IP datagrams" qui ne correspondent pas aux caractéristiques de taille du réseau destinataire.

Définition :

La **fragmentation** donne lieu à la constitution d'un ensemble d'informations permettant **le réassemblage des différents fragments *sur le site destinaire***.

Nous décrivons ci-dessous les grandes lignes de la technique de fragmentation utilisée :

① **tous les fragments issus d'un même datagramme portent la même identité** : champ Identification identique.

Ce numéro permet d'identifier différents fragments comme parties d'un même datagramme ;

② Le champ Flags :

- le premier bit du champ Flags (qui en comporte 3) n'est pas utilisé (il doit être nul).
- Le second permet de demander (s'il est égal à 1) à ce que le datagramme ne puisse en aucun cas être fragmenté.
- Enfin le troisième (*More Flag* **indique, lorsqu'il est égal à 1, que le fragment n'est pas le dernier du datagramme**.
  Ce bit n'est donc nul que dans le dernier fragment du datagramme.

③ **le datagramme originel n'est reconstitué (sauf négociation particulière) que sur le site destinataire final** : si le routage nécessite le passage par différentes passerelles, les différents fragments seront, en principe, acheminés individuellement. Les passerelles sont par ailleurs susceptibles de procéder à une nouvelle fragmentation plus fine si les contraintes physiques l'imposent ;

④ Si l'un des fragments constituant un datagramme est perdu, alors le datagramme complet est considéré comme perdu.

⑤ La taille des fragments doit être un multiple de 64 bits

⑥ Le champ *Offset Field* du "IP header" indique la distance en unités de 64 bits depuis le premier bit du datagramme.

⑦ A l'arrivée, le processus de "reassembly" commence. Sa durée est temporellement bornée car le succés de cette opération n'est pas garanti.

5.5.1 Critique de la fragmentation

Le processus de fragmentation est couteux en temps de calcul pour les routeurs.

➢ Il tend donc à accroître le délai de transmission.

Il peut aussi entraîner une certaine inefficacité.

➢ La fragmentation engendre des datagrammes de taille plus petite adaptée au MTU du réseau physique sous-jacent.

Dans le cas où se MTU viendrait à croître de nouveau (changement de réseau), les petits datagrammes n'exploiteraient pas cette nouvelle caractéristique (puisqu'ils ne sont rassemblés qu'à destination) et du coup le rendement de la transmission serait non-optimal.

Afin, d'un point de vue de la machine destinatrice, cette tâche (réassemblage) vient s'ajouter à sa charge initiale.

5.6 Time to Live

Dans le but d'éviter l'existence de datagrammes fantômes n'arrivant pas à atteindre leur destination, IP tente de garantir un temps butoir à partir duquel les datagrammes qui n'ont pas été livrés peuvent être considérés comme perdus.

Ceci est réalisé grâce au champ « **time–to–live** » de l'entête.

Ce champ est initialisé avec la valeur maximum à la source et décrémenté au fur et à mesure que le datagramme traverse le réseau.

- Si le champ vaut 0 avant d'avoir atteint la source alors il est éliminé.

Idéalement, l'unité de temps manipulée devrait être les secondes ou millisecondes.

➢ Ceci est en pratique trés dur à réaliser car cela nécessite de maintenir une horloge globale au réseau.

➢ On travaille donc en "hops" c'est à dire le nombre de router traversés.

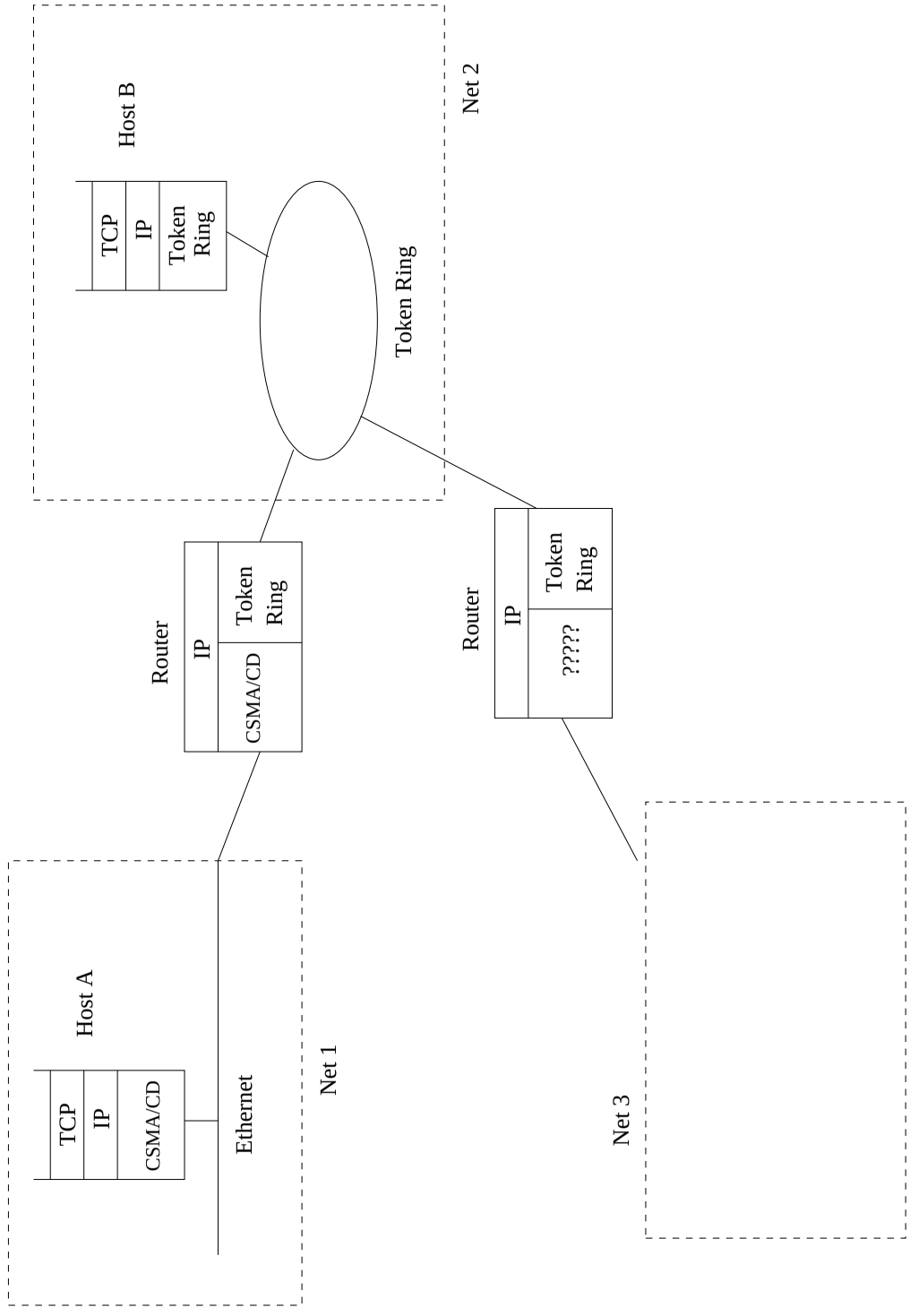
Chaque router décrémentant la valeur de 1.

Celui qui produit une valeur 0 ayant en charge de détruire le datagramme et du coup envoie un message ICMP à la source.

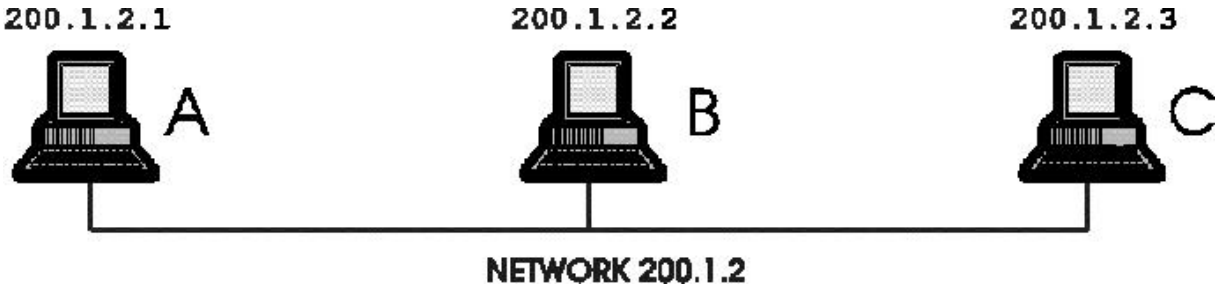
Remarque :

Il est intéressant de noter que ce champ continue d'être décrémenté au cours du réassemblage des différents fragments d'un datagramme et permet de fixer une limite à l'attente de fragments manquants sur le site destinataire.

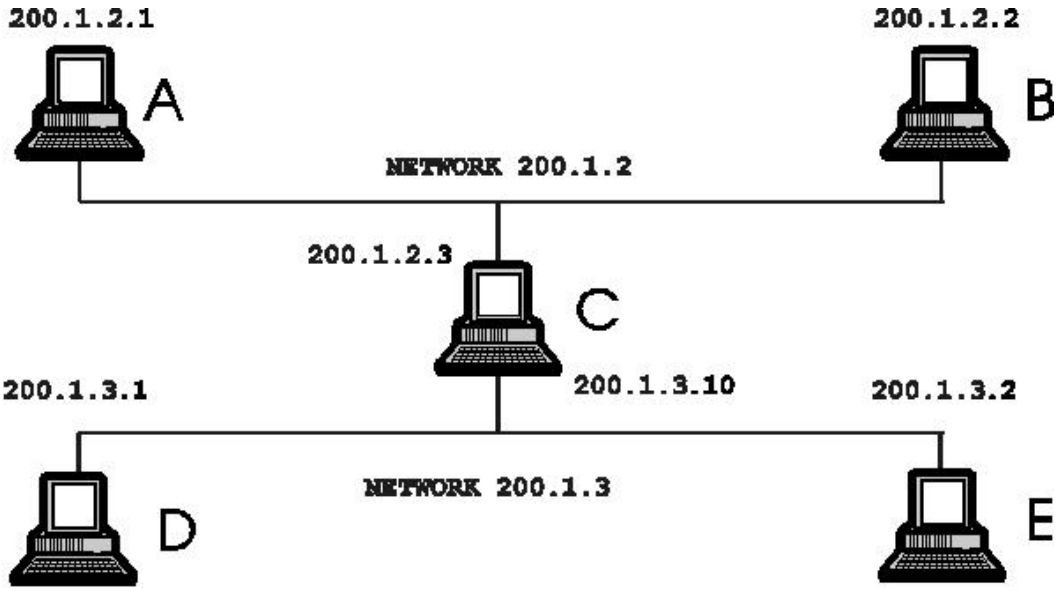
Interconnexions de réseaux



Routage : Remise directe



Routage : Remise indirecte



5.7 Routage IP

Le format des datagrammes caractérise les aspects statiques de IP.

➢ Le routage caractérise les aspects opérationnels.

Conceptuellement, le routage IP (cf RFC 1812) est simple. Il met en œuvre deux types de remises :

① Remise directe.

② Remise indirecte.

5.7.1 Routage direct

Considerons un petit réseau interne TCP/IP consistant d'un segment Ethernet et de 3 noeuds.

- Le numéro IP de ce segment Ethernet est 200.1.2.
- Les numéros IP des machines A, B et C sont respectivement 1, 2 et 3.
- Ceux sont des adresses de classes C et par conséquent il peut y avoir jusqu'à 254 noeuds sur ce segment de réseau.
- Chacun de ces noeuds a une adresse physique Ethernet de 6 octets que l'on écrira par exemple 02-FE-87-4A-8C-A9 (avec un notation à tirets.

Le **routage direct** est celui mis en place dans le premier exemple lorsque A veut communiquer avec C.

Définition :

En somme, si le paquet n'a pas besoin d'être « forwardé », i.e. que **la source et la destination sont sur le même réseau**, alors c'est le routage direct (ou **direct routing**) qui est utilisé.

Même en présence de routeur, le dernier routeur aura toujours à faire une remise directe :

➢ Il est donc aussi utilisé dans le deuxième exemple lorsque A communique avec E, ou **plus précisément** lorsque **A communique avec C et C communique avec E**.

Etre sur le même réseau . . .

C'est quelque chose qui se vérifie facilement. Il suffit de comparer les identificateurs réseaux des adresses IP de l'émetteur et du destinataire.

- On vu que cela se complique un tout petit peu dès que l'on utilise le sous adressage.

N'oublions pas ARP . . .

Supposons que A souhaite envoyer un paquet à C pour la première fois et qu'il connaisse l'adresse IP de C grâce à un appel DNS.

- Pour envoyer ce paquet via Ethernet, A a besoin de l'adresse Ethernet (adresse physique) de C.
- A va donc utiliser l'Address Resolution Protocol (ARP) !

Ceci me permet d'enchaîner sur une caractéristique essentielle de la remise directe :

➢ Dans ce cas, l'adresse de lien (Ethernet) correspond à la machine de destination du datagramme IP.

On verra que cette caractéristique n'est plus vraie lors d'une remise indirecte.

5.7.2 Routage indirect

Définition :

Le **routage indirect** est utilisé dés lors que le **numéro de réseau** de la source et de la destination ne correspondent pas et qu'il faut par conséquent mettre en œuvre un ou plusieurs routeurs.

➢ C'est donc le cas du deuxième exemple, lorsque A veut communiquer avec E.

En revenant à l'exemple, supposons maintenant que deux réseaux Ethernet sont joints par un C sur la figure, agissant comme un routeur.

D'un point de vue de la structure du réseau, on est en présence de deux segments Ethernet séparés et, pour l'instant, les concepteurs n'ont pas choisi d'utiliser un sous-adressage explicite donc :

➢ **chaque réseau a son propre numéro IP (identificateur réseau) de classe C**.

Par conséquent, soit deux machines A et E désirant communiquer. Ces deux machines appartiennent à des réseaux différents :

➢ A appartient au réseau 200.1.2 où elle est identifiable par son adresse IP : I_A.

➢ B appartient au réseau 200.1.3 où elle est identifiable par son adresse IP : I_B.

La communication de A vers E est initiée lorsque l'utilisateur en A demande à l'entité locale IP d'envoyer des données à l'adresse Internet I_E.

Pour ce faire, l'entité IP en A **encapsule** les données dans un **IP datagram** et détermine l'adresse de destination.

Le remise indirecte est caractérisée par le fait que l'adresse IP de destination et l'adresse de lien (Ethernet) ne correspondent pas à la même machine.

➢ L'adresse de lien est celle de la machine intermédiaire,

➢ L'adresse IP est celle de la machine au bout de la route . . .

5.7.3 Algorithme

Il y a deux provenances potentielles de datagrammes pour un module IP d'une machine :

① **Provenance locale** : le datagramme provient des modules UDP, TCP, ICMP ou IGMP.

② **Provenance externe** : l'interface réseau transmet au module IP un datagramme que ce dernier est, par exemple, censé router.

Lorsqu'un datagramme est reçu via une interface réseau, IP vérifie d'abord si l'adresse de destination est une de ses propres adresses ou une adresse IP de broadcast.

① Si c'est le cas, le datagramme est démultiplexé grâce à son champ protocole et donc confié au module de protocole spécifié (dans l'entête IP).

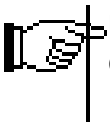
② Si ce n'est pas le cas (i.e ce datagramme n'est pas destiné à ce module IP),

➢ soit la machine est configurée pour réaliser une opération de routage (activation de l' **ipforwarding** , et dans ce cas le paquet va être réémis comme suit,

➢ soit la machine n'est un routeur et le datagramme est rejeté silencieusement.

Plaçons nous maintenant dans la cas, où le module IP se voit confier la gestion du datagramme.

Table de routage :



La couche IP gère une table de routage en mémoire qu'elle consulte **à chaque fois qu'on lui confie un datagramme à envoyer**.

Cette table contient des entrées et peut être visualisée avec la commande `netstat -rn` :

```
~> hostname
ellington
~> netstat -rn

Routing Table:
  Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
127.0.0.1            127.0.0.1             UH       0    173  lo0
134.59.2.0           134.59.2.216          U        3    546  le0
224.0.0.0            134.59.2.216          U        3      0  le0
default              134.59.2.254          UG       0    375
```

Chaque entrée dans la table de routage contient les informations suivantes :

① Une **adresse IP de destination** : cette adresse peut être

➢ une adresse de machine identifiant une machine particulière (id_ord != 0).

➢ une adresse de réseau identifiant toutes les machines du réseau (id_ord == 0)

Les flags contribuent à la différentiation.

② Une **adresse IP de passerelle** (gateway) : cette adresse peut être

➢ l'adresse IP d'un routeur de saut suivant (**next-hop-router**).

➢ l'adresse IP d'une interface (de la machine) au réseau connectée directement.

Ce type de gateway est utilisée pour décrire « l'Ethernet » (cf bas de page 122 [6]). L'entrée dans la table de routage prend alors la forme :

- L'adresse de destination est une adresse de réseau, celui directement connecté à la machine devant router le datagramme.
- L'adresse de gateway est une adresse d'interface de la machine devant router le datagramme.
- Seul le flag U(up), est levé. (La destination est un réseau donc pas de flag H et la gateway n'est pas un routeur donc pas de flag G).

La sémantique est : «Le trafic pour ce réseau doit passer par l'interface de la machine»

L'adresse IP du routeur de saut suivant doit nécessairement être sur le même réseau de façon à pouvoir réaliser une remise directe.

③ **Des flags** : Ces flags indiquent :

➢ si l'adresse de destination est celle d'un réseau ou d'une machine.

➢ si la gateway est un routeur ou un machine directement connectée au réseau.

④ La **spécification d'une interface** : par laquelle le datagramme doit passer.

Le routage IP est effectué sur la base du saut à saut.

➢ Comme on peut le voir d'aprés l'information de la table de routage, IP ne connaît la route complète d'aucune destination indirecte.

Le routage IP va fournir l'adresse du routeur de saut suivant,
- Ce routeur est supposé plus proche de la destination,
- Ce routeur est directement connecté à cette dernière.

Routage :

Le routage IP effectue les traitements suivants :

① Recherche dans la table d'une entrée associée à l'adresse IP de destination (intégrale) : **on cherche une machine** (id_ord != 0).

Si on trouve, envoi du paquet à la gateway (i.e. routeur de saut suivant ou réseau directement connecté).

② Recherche dans la table d'une entrée correspondant à **un numéro de réseau identique** à celui de l'adresse de destination.

Si on trouve, envoi du paquet à la gateway (i.e. routeur de saut suivant ou réseau directement connecté).

Cette unique entrée suffit à atteindre toutes les machines situées sur le réseau de destination.

➢ Toutes les machines reliées sur un Ethernet local, par exemple, sont gérées à l'aide d'une entrée de ce type.

Ce processus de vérification doit pouvoir prendre en compte un éventuel masque de sous-réseau.

On note que la recherche d'une machine est toujours prioritaire.

③ Recherche dans la table de routage d'une **entrée dénommée `default`**

Si on trouve, envoi du paquet à la gateway (i.e. routeur de saut suivant ou réseau directement connecté).

Quelques fois, cette entrée correpond à l'adresse de destination 0.0.0.0.

④ Si **aucune recherche n'a aboutit**, génération d'un message ICMP (`host unreachable` ou `unreachable network`) à destination de l'application qui est à l'origine de ce datagramme.

5.7.4 Mise en oeuvre Unix

Une méthode Unix-style pour ajouter une entrée de routage à la machine A est :

```
route add [destination_ip] [gateway] [metric]
```

où la **metric value** est le nombre de sauts jursqu'à la destination.

Dans ce cas :

```
route  add   200.1.3.2   200.1.2.3 1
```

dira à A d'utiliser C comme « gateway » pour atteindre E.

La plupart du temps, il n'est pas nécessaire de préciser explicitement (par une commande route) la route pour toutes les destinations.

Il est suffisant de dire que C est la **passerelle par defaut** (¡¡ **default gateway** ¿¿).

➢ La **default gateway** est l'adresse IP de la machine où envoyer tous les paquets destiné à une machine qui n'est pas directement connectée au réseau.

La table de routage de la default gateway doit être correctement configurée pour re-émettre (forward) les paquets correctement.

```
~/TeX/INFO/Cours_Internet> route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref
134.59.131.0    *               255.255.255.0   U     0      0
127.0.0.0       *               255.0.0.0       U     0      0
default         134.59.131.254  0.0.0.0         UG    0      0
```

5.7.5 Politique de routage

La politique de routage consiste à décider quelles routes définir dans la table de routage.

Routage statique :

Le routage est dit **routage statique** lorsqu'il met en oeuvre une table de routage qui reste opérationnelle indéfiniment, jusqu'à elle soit changée manuellement par l'utilisateur.

➢ C'est la forme la plus basique de routage, et elle peut tout à fait convenir si la topologie du réseau, et des machines qui y sont connectées, varie peu dans le temps.

Routage dynamique :

Le **routage dynamique** utilise des protocoles spéciaux permettant de collecter des informations visant à gérer (au mieux) le routage.

Ces protocoles permettent de réactualiser automatiquement et périodiquement les tables de routage avec des routes sur lesquelles il y a un conscensus entre les différents routeurs.

Il y a une classification de ces protocoles suivant qu'ils sont :

- **Interior Gateway Protocols** (IGPs) :
  Les protocoles IGP sont utilisés pour distribuer l'information de routage à l'intérieur d'un **Autonomous System (AS)** .
  Un AS est un ensemble de routeurs **dans** un domaine administré par une autorité.
  Exemples de protocoles IGP :
  - ➢ OSPF (Open Shortest Path First)
  - ➢ RIP (Routing Information Protocol)
- **Exterior Gateway Protocols** (EGPs) :
  Les protocoles EGP sont utilisés pour le routage inter-AS, ainsi chaque AS sera informé sur les chemins menant aux autres AS à travers l'Internet.
  Exemples de protocoles EGP :
  - ➢ EGP
  - ➢ BGP

  See RFC 1716 for more information on IP router operations.

5.7.6 Routage Classless (RFC1519)

So far, we have been discussing conventional routing.

One recent change to IP routing has a significant impact on subnet routing : **Classless Interdomain Routing** (CIDR).

In classless routing, address classes become meaningless.

➢ The boundary between the network and host portions of the address is no longer implied but rather is always explicitly stated.

➢ Instead of assigning a specific class of network, the IANA assigns a block of addresses as a network-netmask pair.

Routing advertisements and tables carry bothnetwork and netmask.

Routing algorithms get the netmask from the routing table instead of the interface definition.

- ➢ This allows hosts to store and track routes for distant subnets as well as local ones.
- ➢ This means subnets no longer need to be contiguous and subnet routes can be advertised to ensure optimal routing. It also provides for supernetting (coalescing of contiguous networks).

The original routing information protocol (RIP) could not support classless routing. RIP version 2 includes extra information in the routing advertisement to support classless routing. Open Shortest Path First (OSPF) has been similarly enhanced.

Although CIDR has been a standard since 1993, most UNIX and router vendors are just now integrating CIDR routing into their products.

Before you design a network that relies on classless routing, make sure you are ready to commit to upgrad ing all your hosts and routers to revisions that are capable ofsupporting it. As long as you have legacy systems on your network, it is best to stick with the old design rules.

Systems known to support CIDR include AIX 4 .2, HP-UX 10.20, Solaris 2.6, and some versionsof Linux.

Aussi :

Dans `http://www.rjsmith.com/ip_addr.html`

Classless Internet Domain Routing, or CIDR (pronounced like ”cider”), is an attempt by the IETF to address the explosion in the proliferation of IP addresses on the Internet. Most people have been led to believe that CIDR addressing is an attempt to conserve IP addresses. The actual benefit is to allow ISPs to aggregate the number of routes in their router’s routing tables. While CIDR addressing does allow ISPs to assign addresses based on the actual size of a customer’s network, it actually wastes some otherwise usable addresses. Howvever, this loss of address space is offset by the need to aggregate routes, since this allows routers to operate more efficiently, resulting in the faster transmission of traffic on the Internet.

6 Le protocole ICMP

Internet est un réseau décentralisé.

- ➢ Il n'y a donc pas de superviseur global du réseau.
- ➢ Chaque routeur fonctionne de manière autonome.

Définition :

Des anomalies, dues à des pannes d'équipement ou à une surcharge temporaire, peuvent intervenir.

- ➢ Afin de réagir correctement à ces défaillances, le protocole de diagnostic ICMP a été développé.

Le **protocole ICMP (Internet Control Message Protocol)** organise un échange d'information **permettant aux routeurs d'envoyer des messages d'erreurs** à d'autres ordinateurs ou routeurs.

Bien qu'ICMP tourne au-dessus de IP, il est requis dans tous les routeurs c'est pourquoi on le place dans la couche IP :

- ➢ Strictement parlant, **c'est une violation du mécanisme de couche** car c'est un protocole au dessus de IP qui sert à IP !

Le but d'ICMP **n'est pas de fiabiliser** le protocole IP, **mais de fournir** à une autre couche IP, ou à une couche supérieure de protocole (TCP ou UDP), **le compte-rendu** d'une erreur détectée dans un routeur.

Un message ICMP étant acheminé à l'intérieur d'un datagramme IP, il est susceptible, lui aussi, de souffrir d'erreurs de transmission.

- ➢ Mais la règle est qu'aucun message ICMP ne doit être délivré pour signaler une erreur relative à un message ICMP.
- ➢ On évite ainsi une avalanche de messages d'erreurs quand le fonctionnement d'un réseau se détériore.

Messages ICMP

1. Annonce d'erreurs réseau :
   - *Destination unreachable*
   - *Parameter problem*
   - *Redirect*
2. Annonce de timeouts :
   - *Time exceeded*
3. Annonce de congestion réseau :
   - *Source quench*
4. Assistance de problèmes :
   - *Echo et echo reply*
   - *Timestamp et Timestamp reply*

6.1 Messages du protocole ICMP

ICMP fournit les messages suivants :

① **Destination unreachable** :

Ce message est retourné à la source d'un datagramme IP. Il contient alors l'entête et les premiers 64 bits du datagramme qui a généré l'erreur.

- ➢ Ce peut être un router qui n'a pas trouvé la destination (TCP or UDP packet directed at a port number with no receiver attached ?)
- ➢ ou qui nécessitait une fragmentation interdite par l'indicateur "Not–Fragment".

② **Parameter problem** :

Ce message est retourné à la source d'un datagramme IP. Il contient alors l'entête et les premiers 64 bits du datagramme qui a généré l'erreur.

- ➢ Il signale qu'une erreur syntaxique ou sémantique a été trouvée dans l'entête du datagramme.

③ **Redirect** :

Ce message est retourné à la source d'un datagramme IP. Il contient alors l'entête et les premiers 64 bits du datagramme qui a généré l'erreur.

- ➢ Il permet de prévenir la source, que lors de son prochain envoie, elle choisisse un autre router, si ce dernier permet un chemin plus "court".

④ **Time exceeded** :

Ce message est retourné à la source d'un datagramme IP. Il contient alors l'entête et les premiers 64 bits du datagramme qui a généré l'erreur.

- ➢ Il indique qu'une valeur "time–to–live" a expiré.

⑤ **Source quench** :

Ce message est retourné à la source d'un datagramme IP. Il contient alors l'entête et les premiers 64 bits du datagramme qui a généré l'erreur.

- ➢ Il permet d'exercer un contrôle de congestion.
  Il équivaut à une demande de réduction de débit de la source.

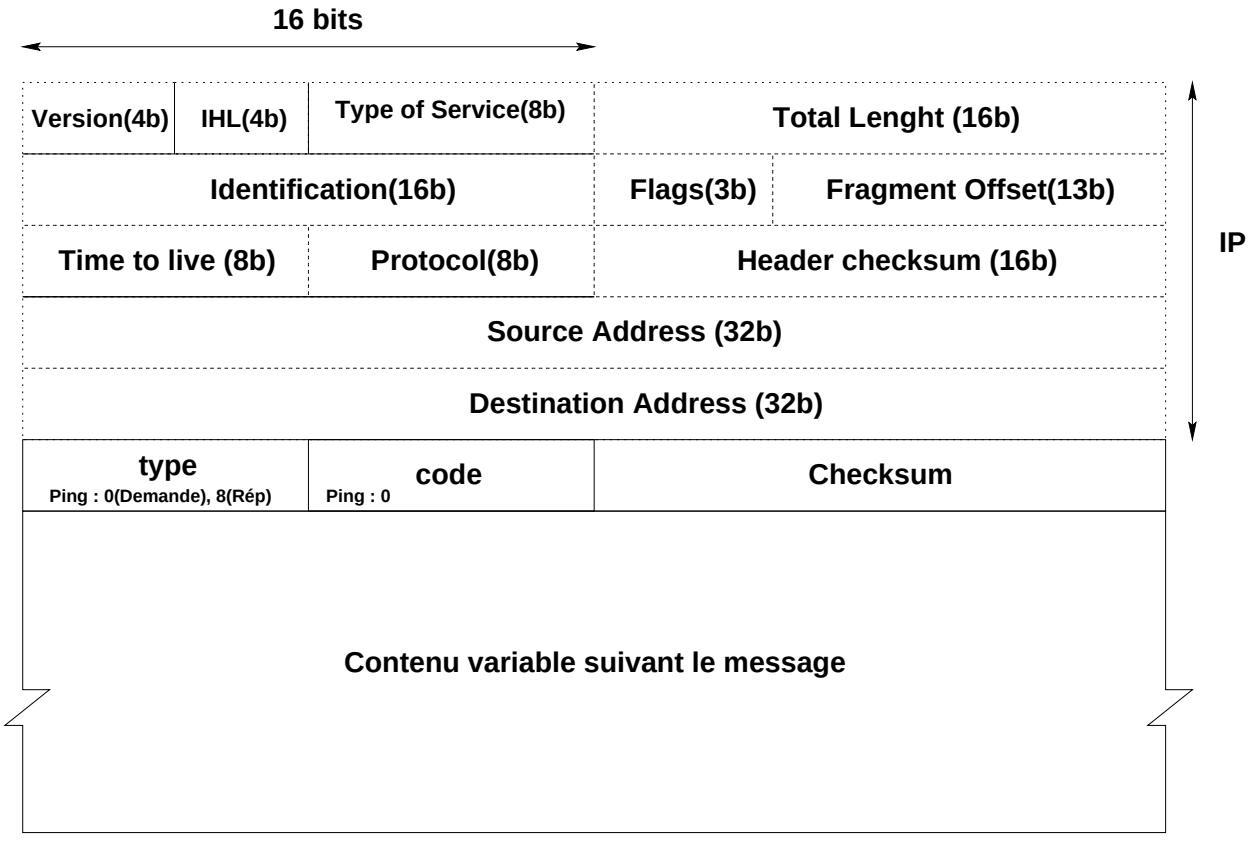
⑥ **Echo et echo reply** :

Une paire de messages échangés entre deux nœuds pour vérifier que la communication est possible.

⑦ **Timestamp et timestamp reply** :

Une paire de messages échangés entre deux nœuds pour échantillonner les caractéristiques temporelles d'INTERNET (**round trip delay** (RTT)).

Datagramme ICMP



6.2 Le datagramme ICMP

La sémantique du datagramme ICMP est résumée dans deux champs :

① Le **champ type** peut prendre 15 valeurs différentes spécifiant de quelle nature est le message envoyé.

② Pour certains types, le **champ code** sert à préciser encore plus le contexte d'émission du message.

Le **checksum** est une **somme de contrôle de tout** le message ICMP calculée comme dans le cas de l'en-tête d'un datagramme IP.

Détails :

Le détail des différentes catégories de messages est donné dans la liste ci-dessous où **chaque alinéa commence par le couple (type,code)** de la catégorie décrite :

- (0,0) ou (8,0) : Demande (type 8) ou réponse (type 0) d'écho dans le cadre de la commande ping.
- (3,0-13) : Compte-rendu de destination inaccessible délivré quand un routeur ne peut délivrer un datagramme.
  Le routeur génère et envoie ce message ICMP à l'expéditeur de ce datagramme.
  Il obtient l'adresse de cet expéditeur en l'extrayant de l'en-tête du datagramme, il insère dans les données du message ICMP toute l'en-tête ainsi que les 8 premiers octets du datagramme en cause.
  Une liste non exhaustive des différents codes d'erreurs possibles est :
  - 0 Le réseau est inaccessible.
  - 1 La machine est inaccessible.
  - 2 Le protocole est inaccessible.
  - 3 Le port est inaccessible.
  - 4 Fragmentation nécessaire mais bit de non fragmentation positionné à 1.
  - 5 Échec de routage de source.
  - 6 Réseau de destination inconnu.
  - 7 Machine destinataire inconnue.
  - 8 Machine source isolée (obsolète)
  - 9 Communication avec le réseau de destination administrativement interdite.
  - 10 Communication avec la machine de destination administrativement interdite.
  - 11 Réseau inaccessible pour ce type de service.
  - 12 Machine inaccessible pour ce type de service.
  - 13 Communication administrativement interdite par filtrage.
- (4,0) : Demande de limitation de production pour éviter la congestion du routeur qui envoie ce message.
- (5,0-3) : Demande de modification de route expédiée lorsqu'un routeur détecte qu'un ordinateur utilise une route non optimale, ce qui peut arriver lorsqu'un ordinateur est ajouté au réseau avec une table de routage minimale.
  Le message ICMP généré contient l'adresse IP du routeur à rajouter dans la table de routage de l'ordinateur.
  Les différents codes possibles ci-après expliquent le type de redirection à opérer par l'ordinateur.
  - 0 Redirection pour un réseau.
  - 1 Redirection pour une machine.
  - 2 Redirection pour un type de service et réseau.

  - 3 Redirection pour un type de service et machine.
- (9,0) : Avertissement de routeur expédié par un routeur.
- (10,0) : Sollicitation de routeur diffusé par une machine pour initialiser sa table de routage.
- (11,0) : TTL détecté à 0 pendant le transit du datagramme IP, lorsqu'il y a une route circulaire ou lors de l'utilisation de la commande traceroute (voir section 2.8.2).
- (11,1) : TTL détecté à 0 pendant le réassemblage d'un datagramme.
- (12,0) : Mauvaise en-tête IP.
- (12,1) : Option requise manquante.
- (13-14,0) : Requête (13) ou réponse (14) timestamp, d'estampillage horaire.
- (15,0) et (16,0) : devenues obsolètes.
- (17-18,0) : Requête (17) ou réponse (18) de masque de sous-réseau.

7 Le protocole UDP

Le **protocole UDP** (RFC 768) est un des deux protocoles de niveau transport proposé par TCP/IP.

Il offre ce service à des applications utilisateurs telles que NFS (Network File System) et SNMP (Simple Network Management Protocol).

Le service proposé est un peu plus qu'une interface à IP.

- Le protocole UDP (rfc 768) utilise IP pour acheminer, d'un ordinateur à un autre, **en mode non fiable des datagrammes** qui lui sont transmis par une application.
- UDP ne gère pas de connexions et n'utilise pas d'accusé de réception et ne peut donc pas garantir que les données ont bien été reçues.
- Il ne réordonne pas les messages si ceux-ci n'arrivent pas dans l'ordre dans lequel ils ont été émis.
- Il n'assure pas non plus de contrôle de flux. Il se peut donc que le récepteur ne soit pas apte à faire face au flux de datagrammes qui lui arrivent.

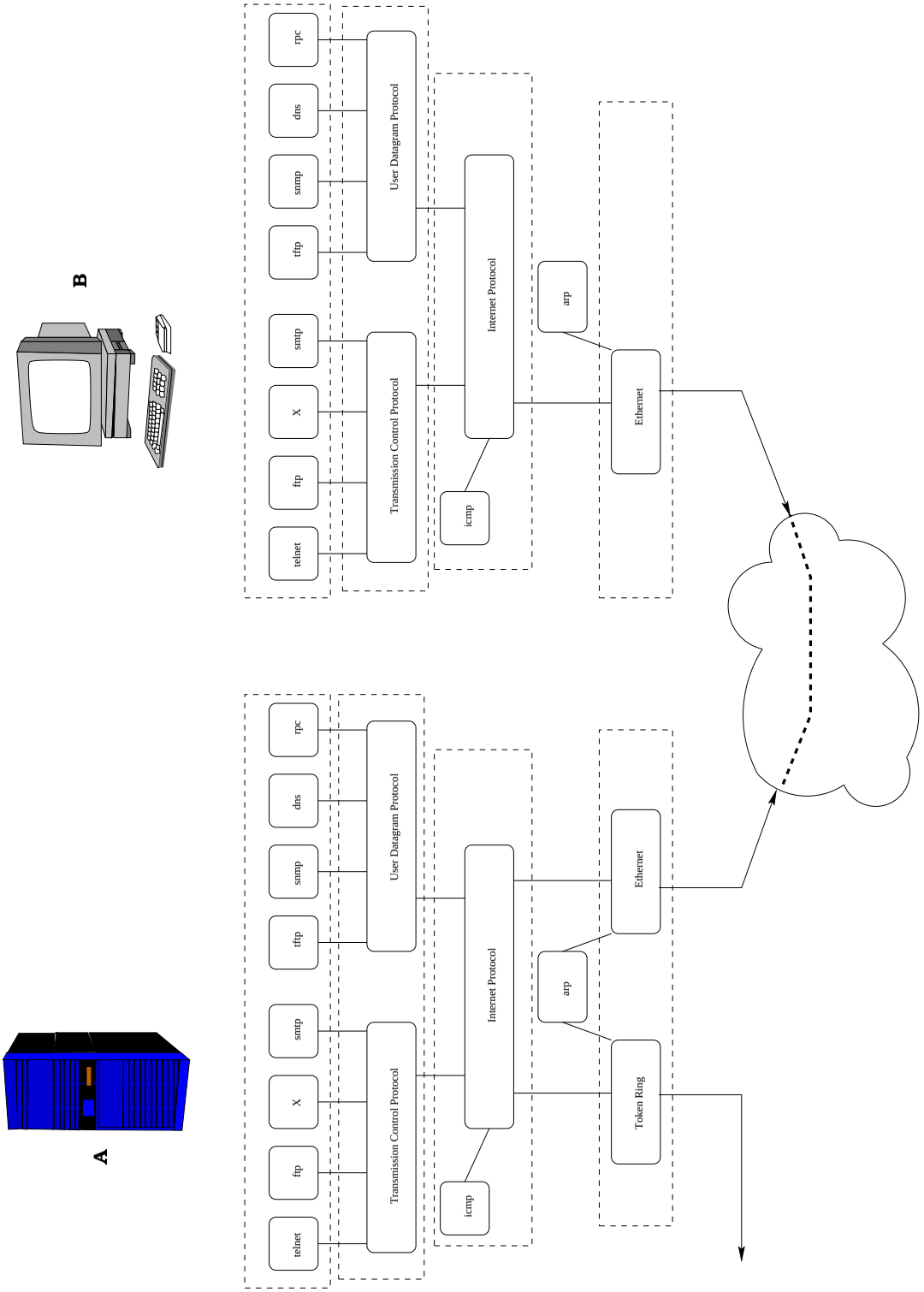
C'est donc à l'application qui utilise UDP de gérer les problèmes de perte de messages, duplications, retards, déséquencement, . . .

Cependant, UDP fournit **deux services supplémentaires par rapport à IP** :

① il permet de distinguer plusieurs applications destinatrices sur la même machine par l'intermédiaire des ports : c'est la notion de **multiplexage-démultiplexage** appliquée à un pile de protocoles.

② il fournit (de façon optionnelle) **un checksum** qui permet de vérifier l'intégrité des données.

Multiplexage - Démultiplexage



7.1 Principes de base du multiplexage et du démultiplexage

Les protocoles de communication utilisent souvent dans l'empilement de couches le multiplexage et le démultiplexage.

L'émetteur :

Un ordinateur qui émet un message y inclut des bits supplémentaires qui indiquent :

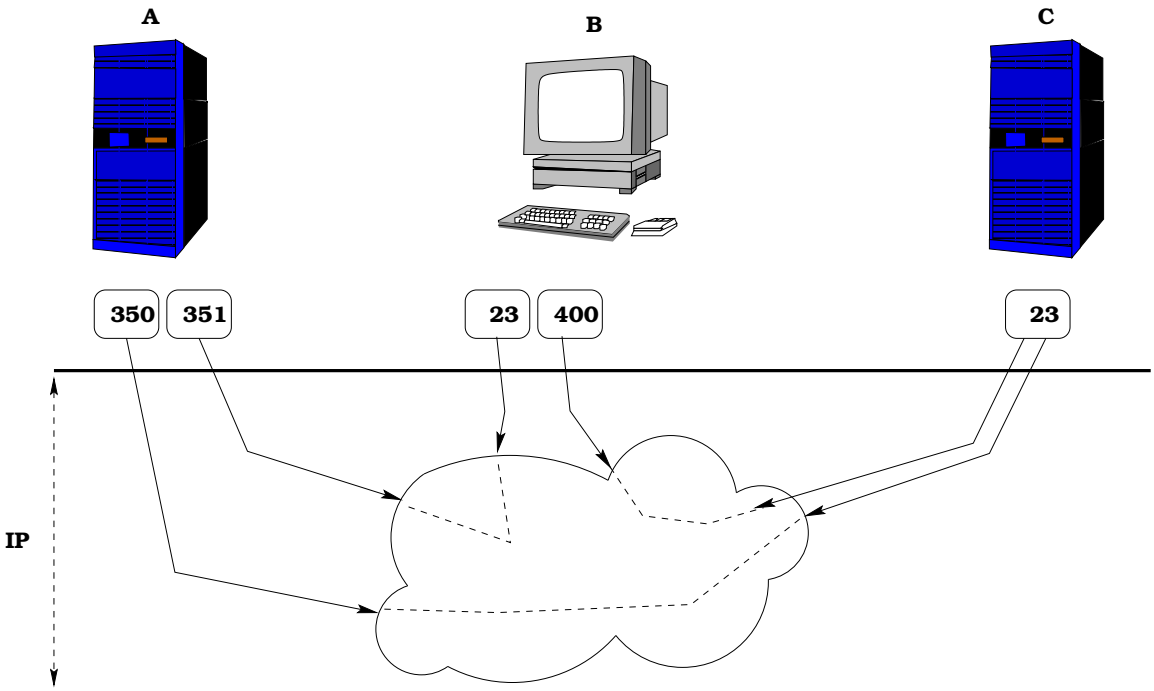
- le type de message
- le programme d'origine
- et les protocoles utilisés.

Puis les messages sont mis dans des datagrammes (UDP ou IP) pour être transférés et placés dans une suite de trames (Ethernet).

Le récepteur :

Côté récepteur, l'ordinateur destinataire utilise ces informations annexes pour orienter les données vers le bon module de traitement.

Ports de communication




```
~> cat /etc/services
netstat         15/tcp
                ....
ftp-data        20/tcp
ftp             21/tcp
telnet          23/tcp
                ....
http            80/tcp
www             80/tcp
                ....
login           513/tcp
                ....
talk            517/udp
```

7.1.1 Les ports UDP

Au niveau de la couche IP, une adresse de destination permet d'identifier un ordinateur.

Mais on ne fait pas de distinction sur l'utilisateur ou le programme d'application qui doit recevoir le datagramme.

Comment identifer la destination finale : machine **et** application ?

➢ La notion de **processus est trop peu abstraite** pour être retenue.

➢ C'est la notion de port qui s'impose !

Définition :

Un port est une destination abstraite sur une machine identifié par un numéro qui sert d'interface à l'application (et éventuellement aux rpocessus sous-jacents) pour recevoir et émettre des données.

Par exemple,

```
~> cat /etc/services

tcpmux          1/tcp                           # rfc-1078
                ....
netstat         15/tcp
                ....
ftp-data        20/tcp
ftp             21/tcp
telnet          23/tcp
                ....
whois           43/tcp          nicname  # usually to sri-nic
domain          53/tcp
domain          53/udp
                ....
http            80/tcp                  # www is used by some broken
www             80/tcp                 # progs, http is more correct
                ....
snmp            161/udp
                ....
login           513/tcp                 # BSD rlogind(8)
                ....
talk            517/udp                 # BSD talkd(8)
                ....
```

est un court extrait du fichier /etc/services dans lequel sont enregistrés les numéros de port utilisés par chaque application.

On y voit que l'application http utilise le port 80 et que l'application snmp utilise le port 161.

➢ Pour communiquer avec un port distant, l'émetteur à besoin de connaitre à la fois l'adresse IP du destinataire et le numéro de port ciblé chez le destinataire.

7.1.2 Gestion des ports

Chaque message contiendra le numéro de port destination et le numéro de port source (à qui les réponses sont adressées).

Les machines émettrice et le réceptrice maintiennent **une table des ports** qui liste les numéros de ports actifs.

La plupart des systèmes d'exploitation assurent un **accés synchrone** aux ports.

➢ Du point de vue d'un processus particulier, l'accés synchrone signifie que le traitement est interrompu pendant l'accés à un port.

Par exemple, si un processus tente d'extraire des données d'un port avant que celles-ci y soient arrivées, le système d'exploitation bloque le processus jusqu'à ce que les données arrivent.

➢ Une fois les données arrivées, le système d'exploitation passe les données au processus et le réactive.

En général, les opérations sur les ports sont **tamponnées** pour que les informations ne soient pas perdues si, lorsqu'elles arrivent, le processus n'est pas prêt à les lire.

➢ Au niveau du système, le moyen le plus simple de se représenter un port est de penser à une file d'attente de taille **finie** !

Lorsque UDP reçoit un datagramme, il vérifie que le numéro de port destination correspond bien à un port actif. Si ce n'est pas le cas un message ICMP de port inaccessible est émis et le datagramme est détruit.

S'il y a correspondance, la datagramme UDP est placé (sauf si la file est pleine) dans la file d'attente associée au port UDP : un programme d'application pourra alors y accéder.

7.1.3 Les numéros de port

Comment affecter les numéros de ports ?

➢ Le problème est important car deux machines doivent convenir des numéros de port utilisés avant de pouvoir interopérer.

Ainsi, lorsqu'une machine A souhaite obtenir un fichier d'une machine B, elle doit connaître le port utilisé par le programme de transfert de fichiers.

Il existe deux approches bien différentes pour réaliser l'affectation de ces ports :

① La première consiste à faire appel à une **autorité centrale**, qui en l'occurence fixe les numéros de ports de 0 à 1024.

Ensuite, les logiciels sont réalisés conformément aux indications de la liste (dite d' **affectation universelle**)

② La seconde approche utilise l'**association dynamique**. Chaque fois qu'un programme d'application a besoin d'un port, les logiciels de communications (incluts dans le système) lui en affecte un.

7.1.4 Gestion d'un paquet UDP

Un paquet entrant IP avec un type UDP est passé au module UDP par IP.

① Analyse du checksum :

➢ Si le checksum vaut 0, cela signifie que le checksum n'a pas été calculé par l'émetteur et il peut être ignoré. Thus the sending computer's UDP module may or may not generate checksums.

If Ethernet is the only network between the 2 UDP modules communicating, then you may not need checksumming.

However, it is recommended that checksum generation always be enabled because at some point in the future a route table change may send the data across less reliable media.

② Analyse du numéro de port :

➢ Si le checksum est valide (il porte sur **l'en-tête et les données** Le numéro de port de destination est examiné et si une application est liée à ce port, un message (pour cette application) est mémorisé dans une queue.

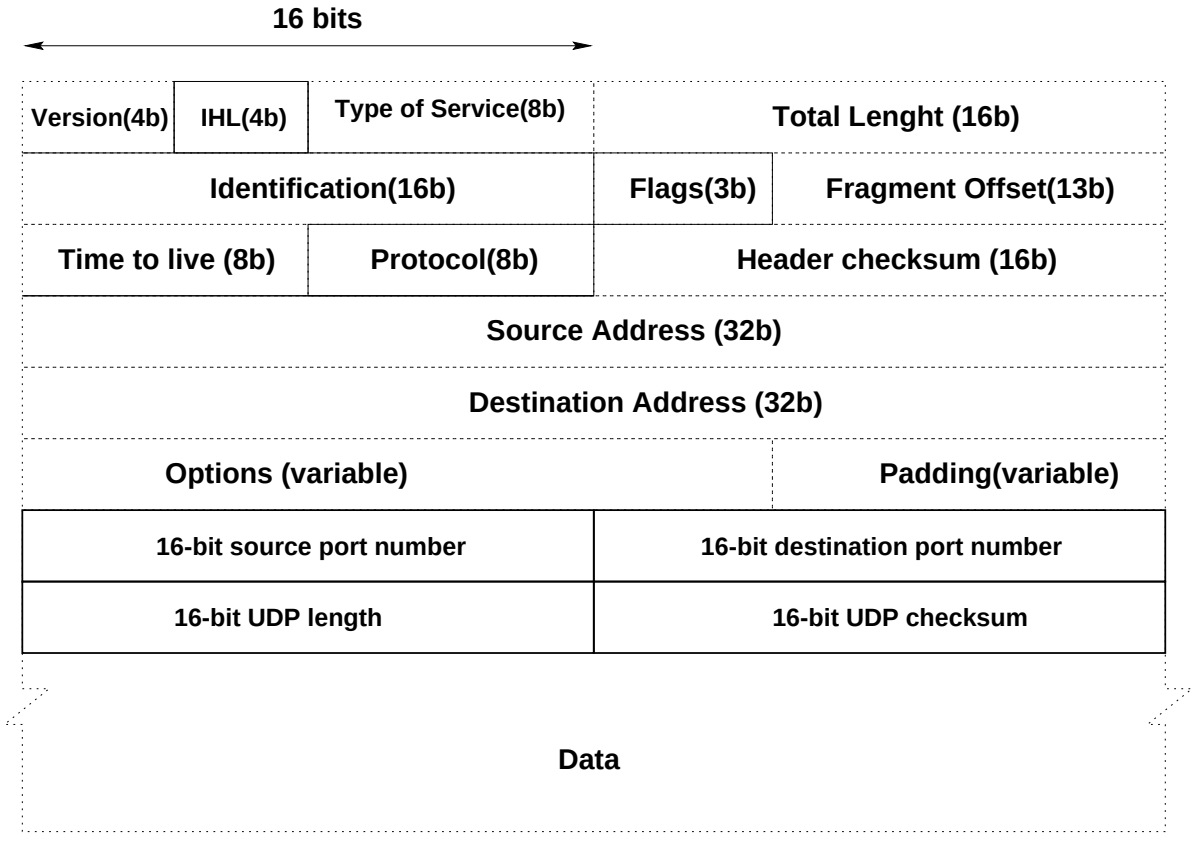
➢ Sinon le datagramme UDP est éliminé

Congestion :

- Si les datagrammes UDP entrants arrivent plus vite que ce que l'application ne peut les lire, et si la queue est pleine
- Alors les datagrammes UDP sont éliminés.

➢ UDP continuera d'éliminer les datagrammes UDP jusqu'à ce que de la place apparaisse dans la queue.

Datagramme UDP



7.2 Le datagramme UDP

- Les **numéros de port** (chacun sur 16 bits) identifient les applications (éventuellement les processus sous-jacents) émettrices et réceptrices.
- Le **champ longueur** contient sur 2 octets la taille de l'en-tête et des données transmises.
  Puisqu'un datagramme UDP peut ne transmettre aucune donnée, la valeur minimale de la longueur est 8 :soit la taille de l'entête du datagramme UDP.
- Le **checksum** est un total de contrôle qui est optionnel car il n'est pas indispensable lorsque UDP est utilisé sur un réseau très fiable.
  S'il est fixé à 0 c'est qu'en fait il n'a pas été calculé.
  De manière précise, UDP utilise **l'en-tête et les données mais également une pseudo-entête** (p.145 [5]) pour aboutir à un checksum.

  On peut s'interroger sur ce qui se passe lorsqu'on a un message UDP dont le total de contrôle aprés calcul est nul :

  ➢ Aussi surprenant que cela puisse paraître, la valeur zéro n'est pas un problème car en complément à un le 0 a deux représentations : tous les bits à un et tous les bits à zéro.

  ➢ Lorsque le total de contrôle vaut zéro, UDP utilise la représentation dans laquelle tous les bits sont à un.

7.2.1 L'encaspulation UDP

Chaque datagramme émis par UDP est encapsulé dans un datagramme IP en y fixant à 17 la valeur du protocole.

7.2.2 C'est IP qui fragmente !

Quand une application envoie des données via UDP, elle arrive à l'autre bout comme une seule unité.

➢ Par exemple, si l'application effectue 5 écritures sur le port UDP,

➢ l'application distante devra faire 5 lectures sur le port UDP.

De plus, la taille de chaque écriture correspond parfaitement à la taille de chaque lecture.

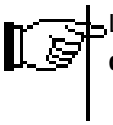
➢ UDP préserve les tailles de message définies par l'application.
Il ne réunit jamais deux messages en un, ni de divise un message d'application en plusieurs.

Il n'en reste pas moins que si UDP ne fragmente pas ces messages (c'est IP qui va le faire), le logiciel UDP n'autorise pas n'importe quelle taille de message d'application.

Extrait du manuel de sendto :

... If the message is too long to pass atomically through the underlying protocol, the error EMSGSIZE is returned, and the message is not transmitted. ...

8 Le protocole TCP



Le **protocole TCP (Transmission Control Protocol)** procure **un service de flux d'octets orienté connexion et fiable**.

8.1 Nécessité de ce type de service

On a vu dans ce qui précède, qu'à divers niveaux des réseaux, des unités de données peuvent être perdues ou détruites ... notamment parce qu'IP ne fournit pas un service sans perte.

- ➢ Les réseaux à routage dynamique peuvent dupliquer des paquets, les ré-émettre dans le désordre, ou avec des retards importants.

Ce service était nécessaire, sous peine de voir chaque programme d'application développer ses propres mécanismes de contrôle !

8.2 Propriétés du service de remise fiable TCP

L'interface entre les programmes d'application et le service de remise fiable TCP/IP possède cinq propriétés :

8.2.1 Orientation Connexion

Les données transmises par TCP sont encapsulées dans des datagrammes IP en y fixant la valeur du protocole à 6.

Le terme **orienté connexion** signifie que les applications dialoguant à travers TCP sont considérées l'une comme un serveur, l'autre comme un client, et qu'elles doivent établir une connexion avant de pouvoir dialoguer.

- ➢ Les ordinateurs vérifient donc préalablement que le transfert est autorisé, que les deux machines sont prêtes en s'échangeant des messages spécifiques.
- ➢ Une fois que tous les détails ont été précisés, les applications sont informées qu'une connexion a été établie et qu'elles peuvent commencer leurs échanges d'informations.

Il y a donc exactement deux extrémités communiquant l'une avec l'autre sur une connexion TCP.

8.2.2 Circuits virtuels

Effectuer un transfert en mode connecté équivaut à effectuer un appel téléphonique.

- ➢ Pendant le transfert, les logiciels des deux ordinateurs continuent à communiquer pour vérifier que les données sont correctement reçues.
- ➢ Si pour une raison quelconque, la communication s'interrompt les deux ordinateurs détectent l'incident et en rendent compte aux programmes d'applications concernés.

Nous utilisons le terme de circuit viruel pour décrire une telle connexion car bien que les programmes d'application la voient comme un circuit physique dédié, sa fiabilité est en réalité assurée par le protocole de remise fiable TCP.

8.2.3 Transferts tamponnés

Les programmes d'application émettent des flots de données sur le circuit virtuel en transmettant des octets au logiciel de communication.

- ➢ Lors du transfert des données, les applications utilisent toute taille de blocs qui leur paraît adaptée (celle-ci peut descendre jusqúau simple octet comme dans le cas de telnet).
- ➢ A l'extrémité réceptrice, le logiciel de communication remet aux programmes d'application les octets dans **l'ordre exact** dans lequel il les a reçus sur la connexion.

Le logiciel de communication est **libre de décomposer le flot de données en paquets indépendants** des blocs de données transmis par les programmes d'application.

- ➢ Il peut, par exemple, améliorer l'efficacité du transfert et minimiser le trafic en regroupant des données pour pouvoir constituer un datagramme de taille suffisante avant de le transférer.
- ➢ De façon analogue, si les programmes d'applications engendrent de trés grands blocs de données, les logiciels de communications peuvent choisir de fragmenter chaque bloc avant de les transmettre.

Dans la cas d'applications où les données doivent être absolument remises même si le tampon n'est pas plein, les applications utilisent la commande **push** pour forcer le transfert :

- Côté émetteur : cette commande oblige le logiciel de communication à émettre toutes le données qui ont été générées sans attndre que le tampon d'émission soit plein.
- Côté récepteur : elle provoque la remise immédiate des données au programme d'application.

La subdivision du flôt reste à la discretion de TCP.

8.2.4 Connexions non structurées

Il est important de comprendre que le service de TCP **ne traite pas des flots structurés**.

- ➢ Le contenu des octets n'est pas du tout interprété par TCP.

Par exemple, il n'y a aucun moyen d'identifer le contenu d'un flot comme étant des variables de types différents (matrices, ...).

- ➢ **Les programmes d'application** utilisant le service de flot **doivent connaître la structure des données échangées** et se mettre d'accord sur le structure du flot avant d'établir la connexion.

8.2.5 Connexions bidirectionnelles simultanées.

Les connexions assurées par TCP permettent les échanges simultanés d'informations dans les deux sens, en mode bidirectionnel simultané (**full duplex**).

- ➢ Du point de vue d'un processus d'application, toute connexion bidirectionnelle simultanée est composée de deux flots indépendants, de sens contraire, sans interaction apparente.

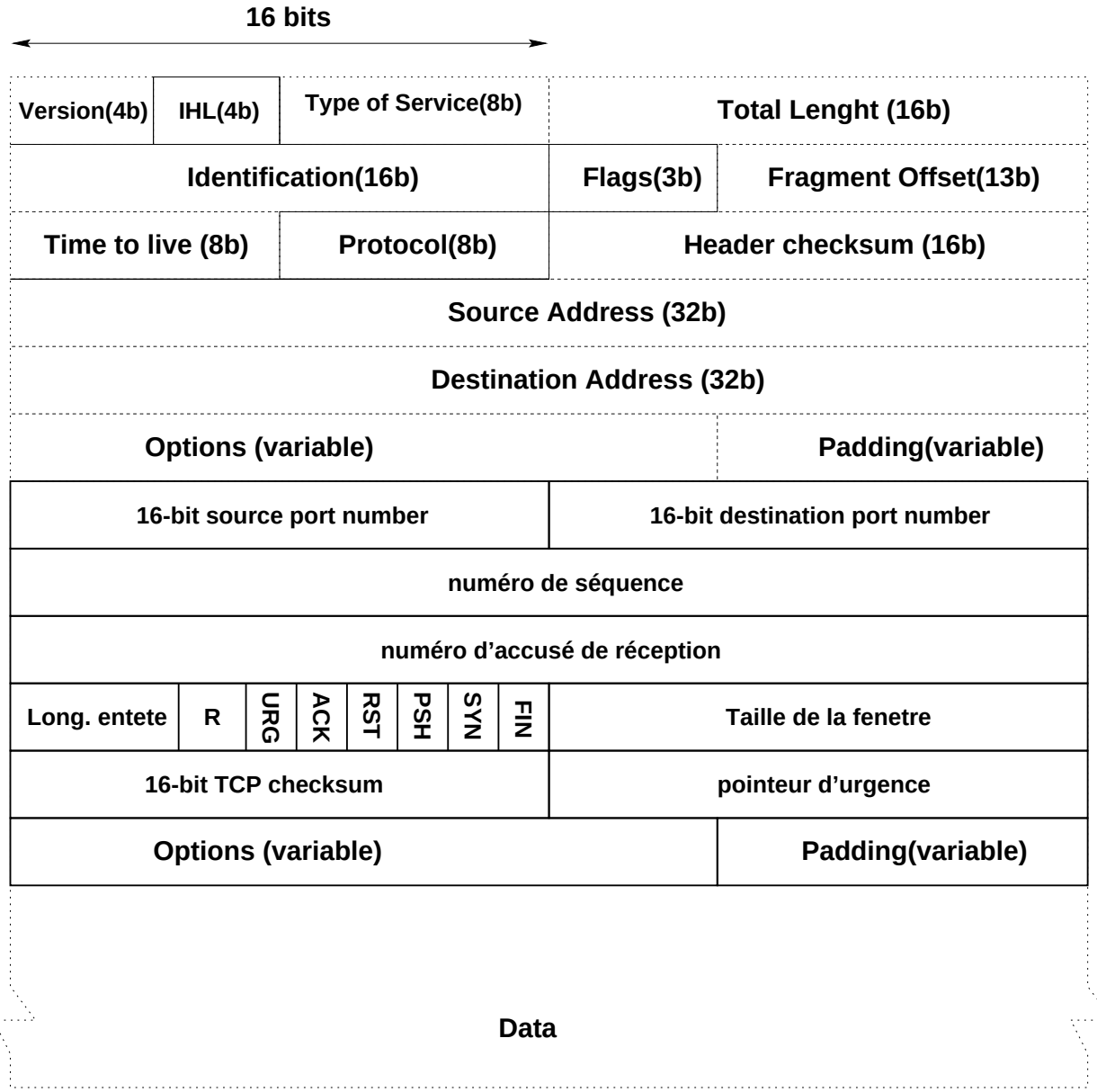
Le service de flot TCP permet qu'un processus d'application libère un sens de transmission alors même que des données continuent d'être transférées en sens inverse.

➢ La connexion devient alors bidirectionnelle à l'alternat (**half duplex**)

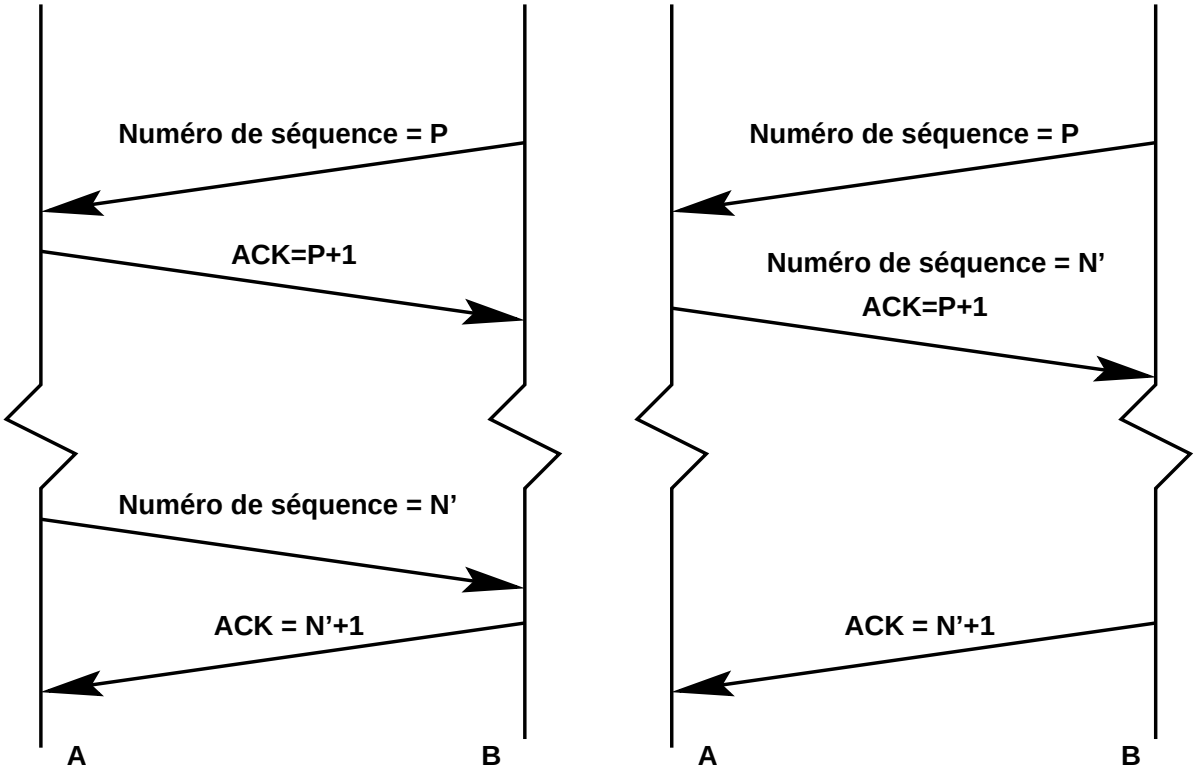
L'avantage d'une connexion bidirectionnelle simultanée est que le protocole sous-jacent peut inclure dans l'en-tête de segments TCP d'une communication de A vers B des informations relatives à la communication de B vers A.

➢ Cette technique de superposition (**piggybacking**) permet de réduire le trafic sur le réseau.

Segment TCP


16 bits
Version(4b)
IHL(4b)
Type of Service(8b)
Total Lenght (16b)
Identification(16b)
Flags(3b)
Fragment Offset(13b)
Time to live (8b)
Protocol(8b)
Header checksum (16b)
Source Address (32b)
Destination Address (32b)
Options (variable)
Padding(variable)
16-bit source port number
16-bit destination port number
numéro de séquence
numéro d'accusé de réception
Long. entete
R
URG
ACK
RST
PSH
SYN
FIN
Taille de la fenetre
16-bit TCP checksum
pointeur d'urgence
Options (variable)
Padding(variable)
Data

Numéro de Séquence et Numéro d'ACK



Avec piggybacking

8.3 Notion de segment

L'unité de données échangées entre les logiciels TCP de deux ordinateurs est le **segment** .
On échange des segments pour :

① établir une connexion,

② transférer des données et

③ libérer une connexion.

Comme TCP utilise la superposition (**piggybacking**), un accusé de réception qui transite d'un ordinateur A vers un ordinateur B peut être acheminé par un segment transportant des données qui transitent de B vers A, même si l'accusé de réception fait référence à des données qui transitent de B vers A.

Chaque segment est divisé en deux partie :
- un en-tête (**TCP header**)
- et des données.

L'en-tête : L'en-tête, sans option, d'un segment TCP a une taille totale de 20 octets.

Il comporte des informations d'identification et de contrôle et se compose des champs suivants :
- Le **port source et le port destination** identifient les applications émettrice et réceptrice (aux extrémités de la connexion) via un numéro de port.
  En les associant avec les numéros IP source et destination du datagramme IP qui transporte un segment TCP, **on identifie de manière unique chaque connexion**.
- Le **numéro de séquence** (non signé sur 32 bits) donne **la position** du segment dans le flux de données **envoyées par l'émetteur** ;
  c'est-à-dire la place dans ce flux du premier octet de données transmis dans ce segment.
- Le **numéro d'accusé de réception** contient le numéro de séquence du prochain octet **attendu par le récepteur** ;
  c'est-à-dire le numéro de séquence du dernier octet reçu avec succès plus 1.

  Notons que le numéro de séquence fait réference au flot qui s'écoule dans le même sens que le segment. Le numéro d'accusé de réception fait référence au flot de sens contraire.

  De manière précise, TCP n'acquitte pas un à un chaque segment qu'il reçoit, mais acquitte l'ensemble du flot de données jusqu'à l'octet k-1 en envoyant un acquittement de valeur k.
  Par exemple, dans une transmission de 3 segments de A vers B,

  ① Si les octets de 1 à 1024 sont reçus correctement, alors B envoie un ACK avec la valeur 1025.

  ② Puis, si le segment suivant contenant les octets de 1025 à 2048 se perd et que B reçoit d'abord correctement le segment des octets de 2049 à 3072, B n'enverra pas d'accusé de réception positif pour ce troisième segment.

  ③ Ce n'est que lorsque B recevra le deuxième segment, qu'il pourra envoyer un ACK avec la valeur 3073, que A interprétera comme l'acquittement des deux derniers segments qu'il a envoyés.

  On appelle cela un **acquittement cumulatif** .
- La **longueur d'en-tête** contient (sur 4 bits) la taille de l'en-tête en mots de 32 bits (on dit aussi multiple de 4 octets), y compris les options présentes.
  Ainsi une en-tête peut avoir une taille variant de

➢ 5 mots de 4 octets = 20 octets (aucune option) à

➢ 15 mots de 4 octets = 60 octets (maximum d'options).

- Le champ **réservé** comporte 6 bits réservés à un usage ultérieur.
- Les 6 champs **bits de code** qui suivent permettent de spécifier le rôle et le contenu du segment TCP pour pouvoir interpréter correctement certains champs de l'en-tête.

  La signification de chaque bit, quand il est fixé à 1 est la suivante.
  – URG, le pointeur de données urgentes est valide.
  – ACK, le champ d'accusé de réception est valide.
  – PSH, ce segment requiert un push.
  – RST, réinitialiser la connexion.
  – SYN, synchroniser les numéros de séquence pour initialiser une connexion.
  – FIN, l'émetteur a atteint la fin de son flot de données.
- La **taille de fenêtre** est un champ de 16 bits qui sert au contrôle de flux selon la méthode de la fenêtre glissante.

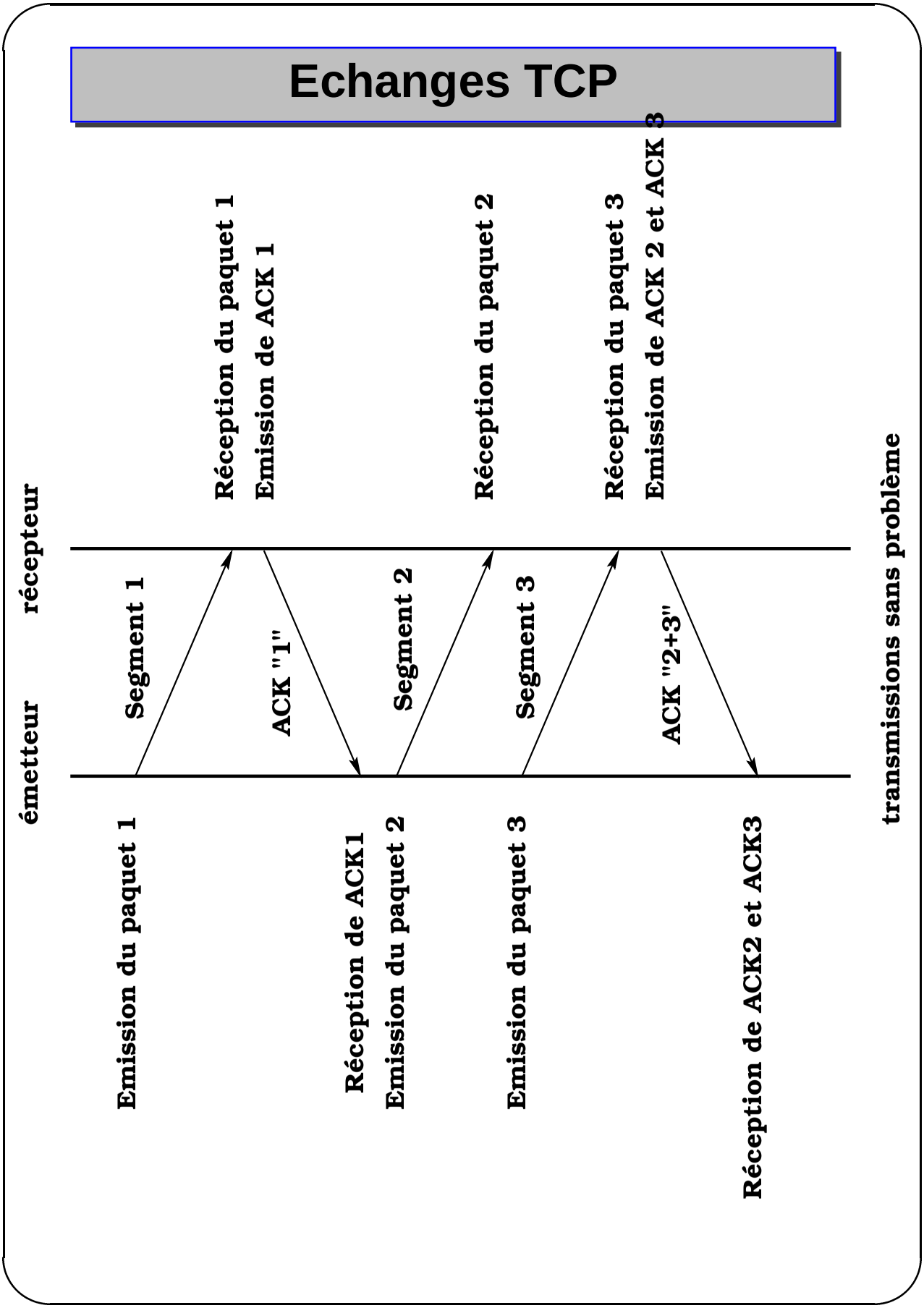
  ➢ Il indique le nombre d'octets (moins de 65535) que le récepteur est prêt à accepter.

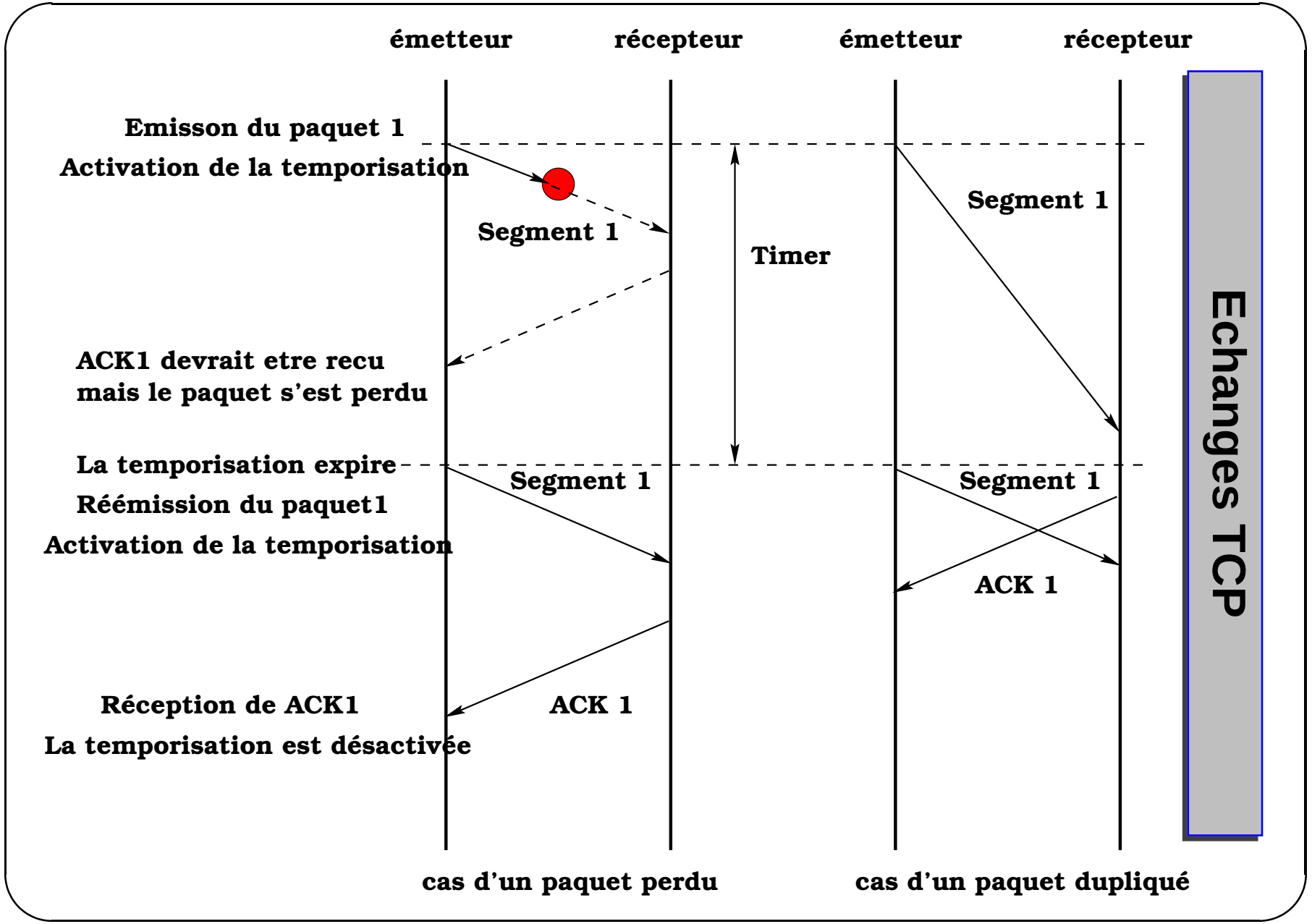
  Ainsi l'émetteur augmente ou diminue son flux de données en fonction de la valeur de cette fenêtre qu'il reçoit.
- Le **checksum** est un total de contrôle sur 16 bits utilisé pour vérifier la validité de **l'en-tête et des données transmises**.
  Il est **obligatoirement** calculé par l'émetteur et vérifié par le récepteur. Le calcul utilise une pseudo-entête analogue à celle d'UDP.
- Le **pointeur d'urgence** est un offset positif qui, ajouté au numéro de séquence du segment, indique le numéro du dernier octet de donnée urgente.

  Il faut également que le bit URG soit positionné à 1 pour indiquer des données urgentes que le récepteur TCP doit passer le plus rapidement possible à l'application associée à la connexion.
- **L'option** la plus couramment utilisée est celle de la taille maximale du segment TCP qu'une extrémité de la connexion souhaite recevoir (**MSS**).

  Ainsi, lors de l'établissement de la connexion il est possible d'optimiser le transfert de deux manières.
  – Sur un réseau à haut débit, il s'agit de remplir au mieux les paquets, par exemple en fixant une taille qui soit telle que le datagramme IP ait la taille du MTU du réseau.
  – Sinon, sur un réseau à petit MTU, il faut éviter d'envoyer des grands datagrammes IP qui seront fragmentés, car la fragmentation augmente la probabilité de pertes de messages.

Echanges TCP




émetteur
récepteur
émetteur
récepteur
Emisson du paquet 1
Activation de la temporisation
Segment 1
Timer
Segment 1
ACK1 devrait etre recu
mais le paquet s'est perdu
La temporisation expire
Réémission du paquet1
Activation de la temporisation
Segment 1
Segment 1
ACK 1
Réception de ACK1
La temporisation est désactivée
ACK 1
Echanges TCP
cas d'un paquet perdu
cas d'un paquet dupliqué

8.4 Echanges TCP : assurer la fiabilité !

Le service de remise fiable garantit la remise des datagrammes,

- sans perte,
- ni duplication,

alors même qu'il utilise IP qui lui est un protocole de remise non fiable.

Comment un protocole peut-il assurer un **service de remise fiable** si les protocoles sous-jacents n'assurent qu'un service de remise non-fiable ?

Ceci est réalisé à l'aide de la technique générale de l'accusé de bonne réception et retransmission (**ACK : positive acknowledgment with retransmission**) présentée de manière simplifiée.

① Chaque **segment est émis avec un numéro** qui va servir au récepteur pour envoyer un accusé de réception (ACK). Ainsi l'émetteur sait si l'information qu'il voulait transmettre est bien parvenue à destination.

De plus, à chaque envoi de segment, **l'émetteur arme une temporisation qui lui sert de délai d'attente de l'accusé de réception** correspondant à ce segment.

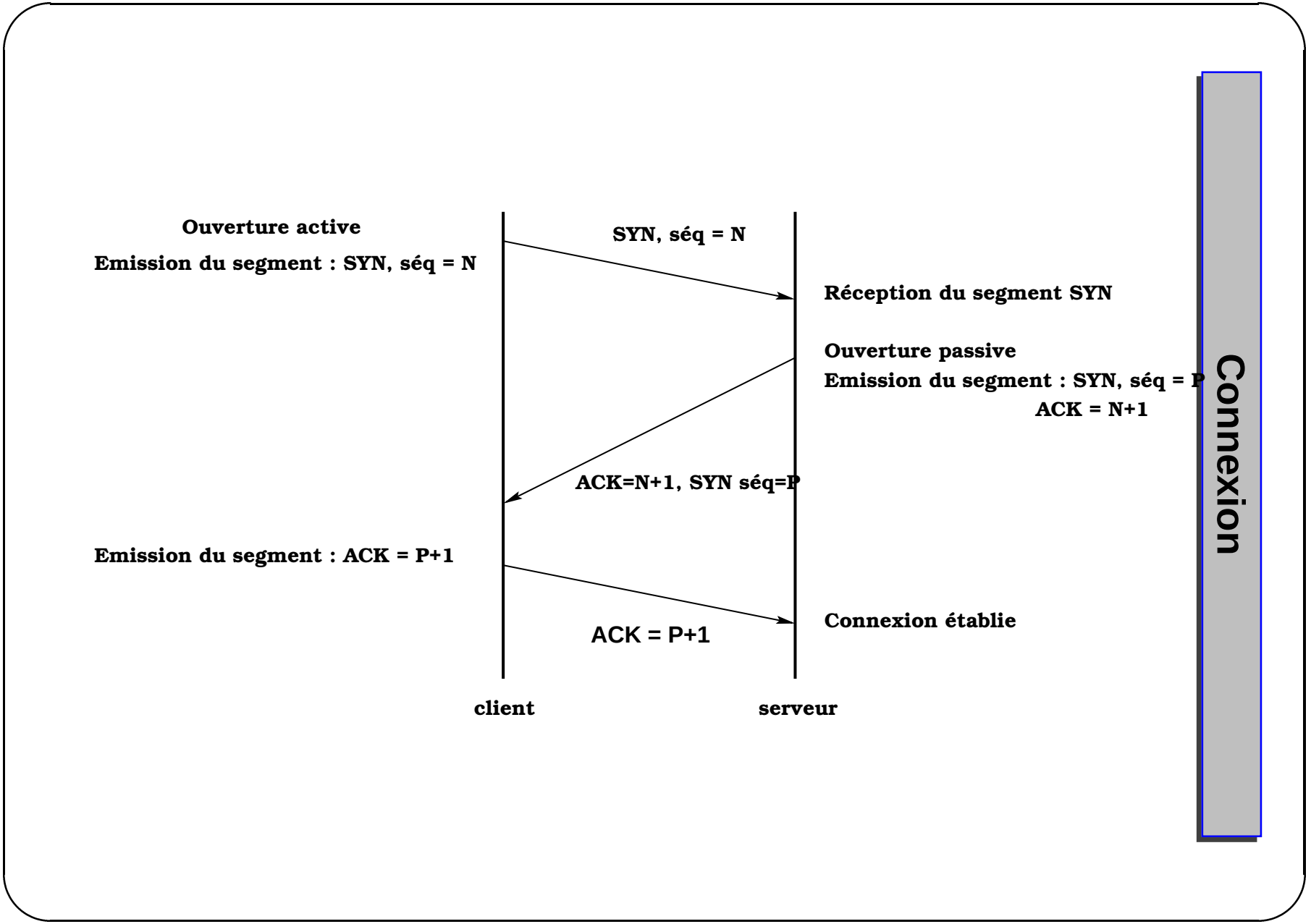
Lorsque la temporisation expire sans qu'il n'ait reçu de ACK, l'émetteur considère que le segment s'est perdu et il le réexpédie.

② Mais **il se peut que la temporisation expire alors que le segment a été transmis sans problème**, par exemple suite à un engorgement de réseau ou à une perte de l'accusé de réception correspondant.

Dans ce cas, l'émetteur réémet un segment alors que c'est inutile : c'est un **doublon** .

Pour résoudre ce problème, les protocoles affectent à chaque paquet(i.e segment) un numéro séquentiel et éxigent du récepteur qu'il garde trace des numéros recus. Le récepteur est donc apte à faire la distinction et peut éliminer les doublons.

Cette technique est aussi étendue aux ACK qui font figurer le numéro de paquet qui est ainsi « accusé de réception ».


Connexion
Ouverture active
Emission du segment : SYN, séq = N
SYN, séq = N
Réception du segment SYN
Ouverture passive
Emission du segment : SYN, séq = P
ACK = N+1
ACK=N+1, SYN séq=P
Emission du segment : ACK = P+1
Connexion établie
ACK = P+1
client
serveur

8.5 Connexion TCP

Bien que la norme veuille qu'il y ait un schéma « client (connexion active) - serveur (attente passive) » , la **demande de connexion** est conçue pour fonctionner dans toutes les circonstances :

- connexion simultanée
- connexion par l'une ou l'autre des extrémités
- il n'y a ni maître, ni esclave.

L'établissement d'une connexion se fait en trois temps et réalise deux fonctions importantes :

① Il garantit que les deux extrémités sont prêtes à transférer des données (et qu'elles le savent).

② Il leur permet de s'accorder sur les numéros de séquences initiaux.

L'extrémité demandant l'ouverture de la connexion, est assimilée au client.

① Il émet un segment SYN (où le bit SYN est fixé à 1) spécifiant le numéro de port du serveur avec lequel il veut se connecter.

Il expédie également un numéro de séquence initial N . Cette phase est appelée ouverture active et « consomme » un numéro de séquence.

② Le serveur répond en envoyant un segment dont les bits ACK et SYN sont fixés à 1.

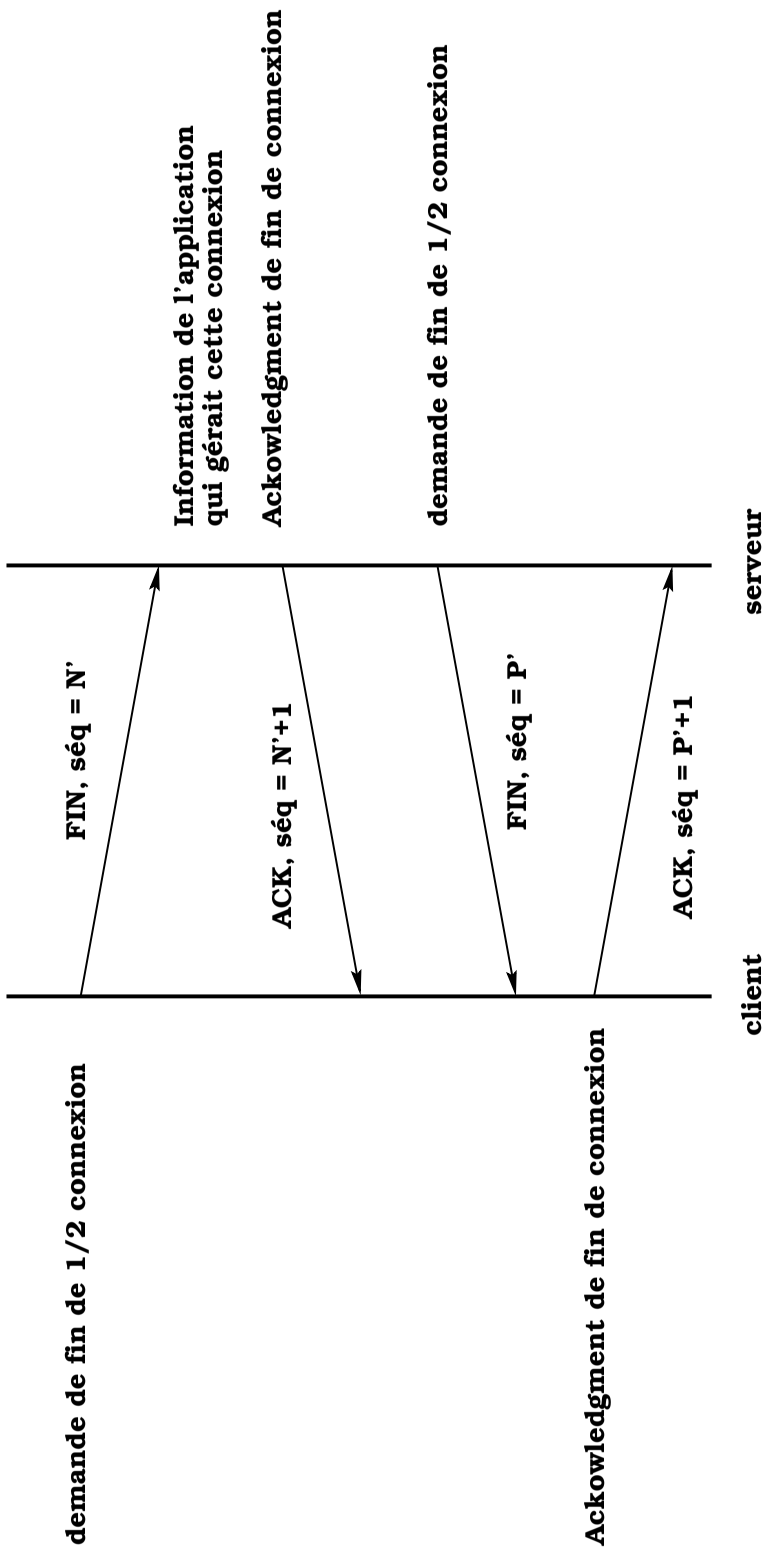
Ainsi, dans un même segment il acquitte le premier segment reçu avec une valeur de ACK=N + 1 et il indique un numéro de séquence initial. Cette phase est appelée ouverture passive.

③ Le client TCP doit évidemment acquitter ce deuxième segment en renvoyant un segment avec ACK=P + 1.

Ces trois segments complètent l'établissement de la connexion.

- Ceci est souvent appelé **la poignée de mains à trois voies** (**three-way handshake**)

Déconnexion



8.6 Déconnexion TCP

La terminaison d'une connexion peut être demandée par n'importe quelle extrémité et se compose de deux demi-fermetures puisque des flots de données peuvent s'écouler simultanément dans les deux sens.

① L'extrémité qui demande la fermeture (le client dans l'exemple) achève la transmission des données restantes à envoyer et présentes dans le tampon,

② Elle attend que le récepteur en accuse réception,

③ Elle émet un segment où le bit FIN est fixé à 1 et où le numéro de séquence vaut N' ,

④ Le récepteur du segment l'acquitte en retournant un ACK=N' + 1 et informe l'application (locale à la récpetion) de la demi-fermeture de la connexion.

⑤ À partir de là, les données **ne peuvent plus transiter que dans un sens** (de l'extrémité ayant acceptée la fermeture vers l'extrémité l'ayant demandée), et dans l'autre seuls des accusés de réception sont transmis (TCP refuse d'accepter des données pour ce sens de transmission)

⑥ Quand l'autre extrémité veut fermer sa demi-connexion, elle agit de même que précédemment ce qui entraîne la terminaison complète de la connexion et la destruction dans le programme de communication de tous les enregistrements relatifs à la connexion.

8.7 Réinitialisation d'une connexion TCP

TCP fournit une procédure de réinitialisation qui permet de rompre une connexion.

Par exemple, lorsqu'un client souhaite une connexion et que le serveur n'a pas de processus souhaitant gérer cette demande de connexion sur le port demandé, le serveur va répondre par une **réinitialisation de connexion** (cf p89 [4].

① Une des extrémités réinitialise la connexion en émettant un segment dans lequel le **bit RST** est positionné.

② L'autre extrémité répond immédiatement en libérant la connexion

La réinitialisation correspond à une cessation immédiate du trafic dans les deux sens.

8.8 Remise forcée des données

On a vu que TCP est libre de décomposer le flot de données en segments de transmission.

Il ne tient donc pas compte de la taille de l'unité de transfert utilisées par les programmes d'application.

- Un des arguments à cela est d'accroître l'efficacité en constituant des segments suffisamment longs pour réduire le coût des **overhead** .

Pour prendre en compte les besoins des utilisateurs **interactifs**, TCP permet à un programme d'application de faire un **push** pour **forcer la remise** des octets présents dans le flot sans avoir à attendre qu'un tampon se remplisse.

➢ Le ¡¡push¿¿ fait plus que contraindre TCP à éméttre un segment.

➢ Il positionne le **bit PSH** dans le segment de façon à ce que les données soient remises, sans délai, au programme application destinataire.

Remarque :
On peut être amené à « pusher » aprés chaque caractère.

8.9 Les données hors bande

Si TCP est un protocole orienté connexion (les données se suivent dans un tube ...), il est parfois important pour le programme qui se trouve à l'autre extrémité de **pouvoir émettre** ce qu'on appelle des données **hors bande** , **sans avoir à attendre** que le programme distant consomme la totalité des octets déjà présents dans le flot.

En ce sens, on modifie le fonctionnement classique du tube qui propose un accés séquentiel aux données.

De telles commandes sont souvent nécessaires si un programme ne fonctionne pas correctement. Elles doivent être transmises sans attendre que le programme lise les données du flot TCP, sinon il serait impossible de tuer des programmes qui ne lisent pas de données.

Pour permettre cette signalisation hors bande, TCP autorise l'émetteur à indiquer que certaines données sont urgentes, ce qui signifie que le programme récepteur doit se voir notifier l'arrivée de ces données aussi rapidement que possible, indépendamment de leur position dans le flot.

Le mécanisme utilisé pour marquer les données urgentes lors de leur transmission dans un segmentest constitué du **bit URG** et du champ **pointeur d'urgence** .

- Lorsque le bit URG est positionné, ce champ repère, dans la fenêtre, la position où les données urgentes se terminent (cf [4]).

8.10 Option de taille maximale de segment

Tous les segments émis sur une connexion ne sont pas de la même taille.
Les deux extrémités vont se mettre d'accord sur une taille maximale de segment :

➢ La négociation utilise le champ **options**

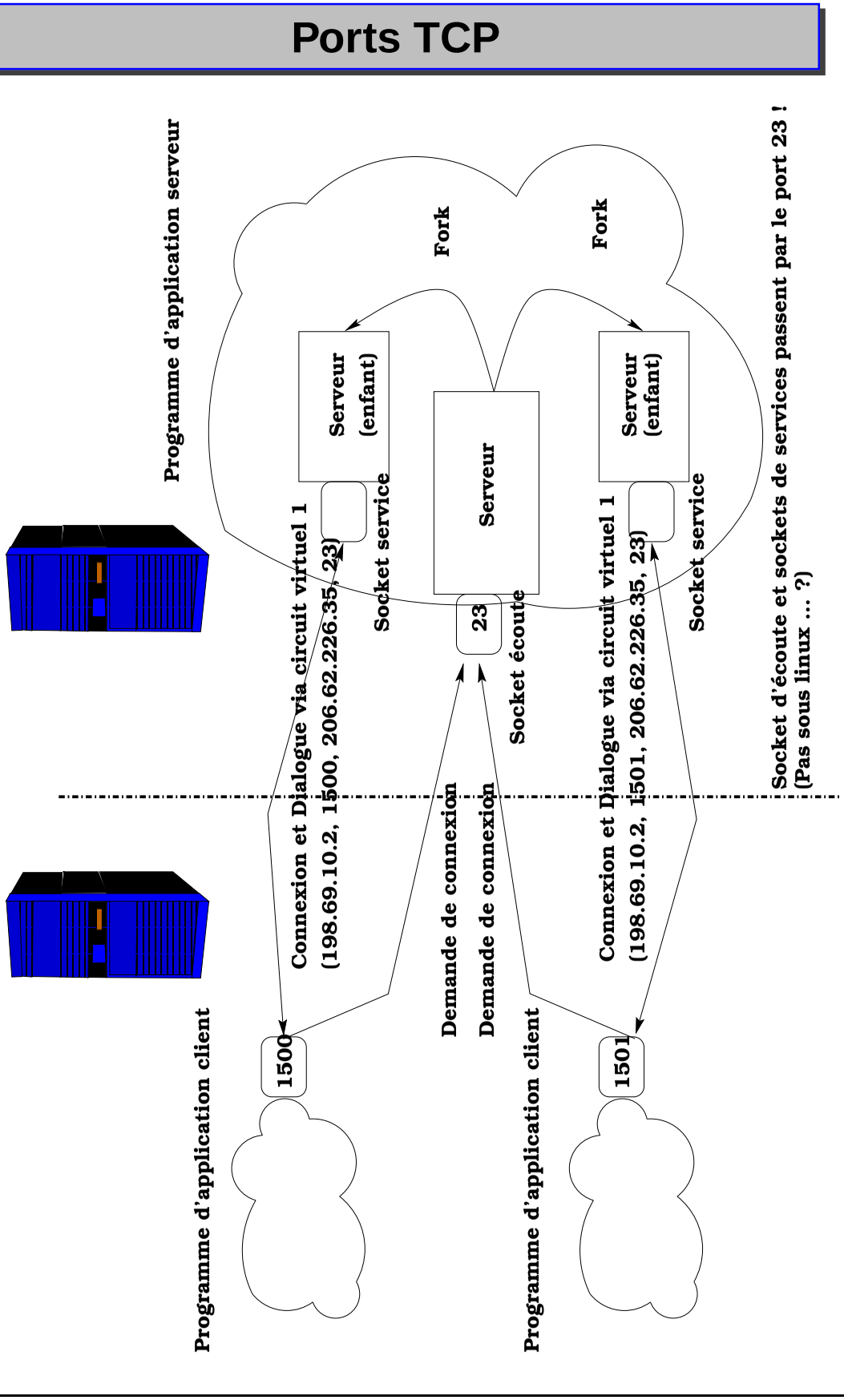
Les critères de négociations peuvent être divers :

- Un petit ordinateur peut négocier une **MSS (Maximum Segment Size)** qui permette aux segments d'être contenus dans le tampon de réception.
- Il est aussi judicieux de choisir une taille de segment qui permette de remplir un paquet sans dépasser le **MTU (Maximum Transmit Unit)** du réseau ... fragmentation ...

8.11 Calcul du total de contrôle TCP

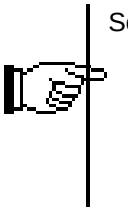
Ca ressemble à celui de UDP .. rien de trés spécial.

Ports TCP


206.62.226.35 : Machine du serveur
198.69.10.2
Programme d'application serveur
Fork
Fork
Serveur (enfant)
Serveur
Serveur (enfant)
Socket service
Socket service
Socket écoute
23
Connexion et Dialogue via circuit virtuel 1
(198.69.10.2, 1500, 206.62.226.35, 23)
Connexion et Dialogue via circuit virtuel 1
(198.69.10.2, 1501, 206.62.226.35, 23)
Demande de connexion
Demande de connexion
Socket d'écoute et sockets de services passent par le port 23 !
(Pas sous linux ... ?)
Programme d'application client
Programme d'application client
1500
1501

8.12 Port, connexion et extrémités de connexion

Comme UDP, TCP se trouve dans la couche transport.



Son rôle est donc de permettre à plusieurs programmes d'application de communiquer.

➢ Il démultiplexe les infos recues vers différents programmes d'application en utilisant les numéros de port.

Le client :

Généralement, que ce soit en TCP ou en UDP, le programme d'application client ne se préoccupe pas du numéro de port qu'il utilise pour émettre ou recevoir.

➢ Seule la garantie de l'unicité du numéro de port, qu'il utilise, est important pour lui.

➢ C'est la condition nécessaire à un démultiplexage cohérent.

Le serveur :

Pour ce qui du programme d'application serveur, il correspond à un service, on doit pouvoir le trouver facilement.

➢ Son numéro de port doit être connu et relativement stable.

Numéro de port UDP :

Dans le cadre d'UDP, un port peut être représenté par une file d'attente dans laquelle le logiciel de communication met les datagrammes à destination de ce port.

➢ Le logiciel serveur n'a qu'à examiner l'entête pour savoir à qui (numéro IP et port) il doit répondre.

Numéro de port TCP :

L'utilisation des ports en TCP est un peu différente par exemple parce que l'application cliente veut pouvoir utiliser un même numéro de port pendant toute la durée d'une connexion.

Or un serveur veut pouvoir gérer en parallèle plusieurs clients (on appelle cela un serveur concurrent).

➢ La conclusion c'est, qu'en TCP, un même numéro de port doit pouvoir servir au démultiplexage de plusieurs connexions. (cf p45 [4])

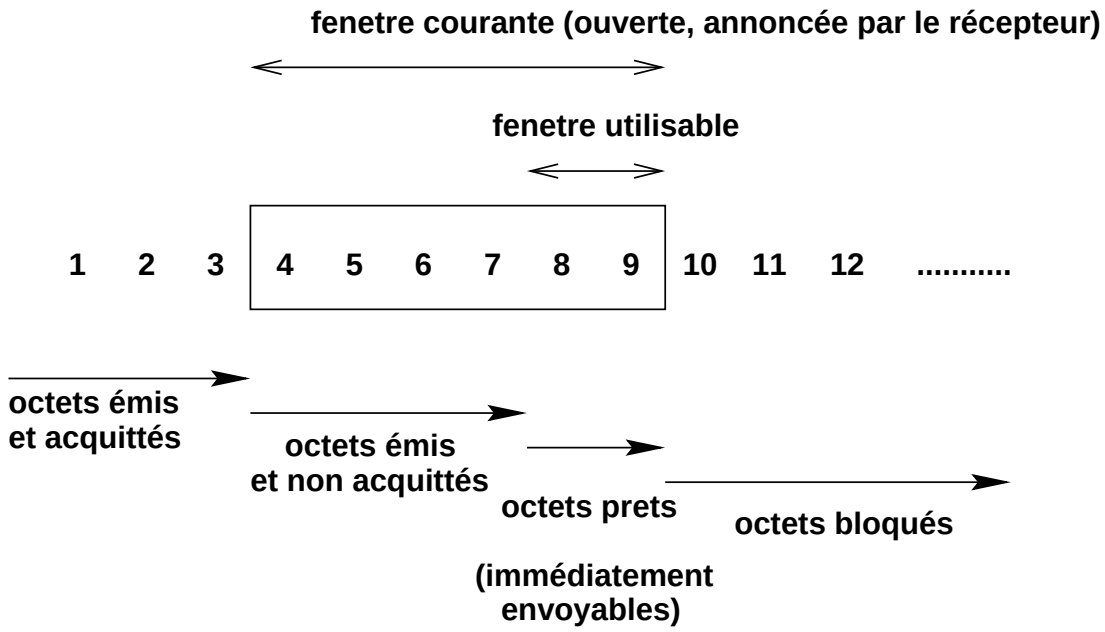
Les ports TCP sont donc plus complexes car un numéro de port donné ne correspond pas à un objet unique (la file d'attente).

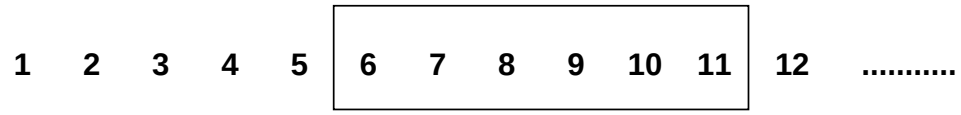
TCP s'appuie sur la notion de connexion dans laquelle les objets sont des **circuits virtuels**, et non des ports individuels.

- Une connexion est identifiée par deux extrémités de connexion.
- L'extrémité est une paire de nombres entiers : (adresse IP d'un ordinateur, numéro de port)

Ceci étant, il est alors possible que sur un même ordinateur plusieurs connexions partagent un même numéro de port.

Contrôle de flux : Fenêtre glissante





La fenetre a glisse apres que 5 ait été acquitté

8.13 Contrôle de flux : la fenêtre glissante

L'objectif du mécanisme que l'on va maintenant étudier est double :

① **améliorer l'efficacité de la transmission de flots** :

Un protocole simple utilisant un mécanisme d'accusé de bonne réception **gaspille une quantité significative de capacité** car la transmission d'un nouveau paquet est **retardée jusqu'à ce que l'accusé de réception** correspondant au paquet précédent ait été reçu par l'émetteur.

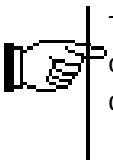
② **résoudre le problème du contrôle de flux de bout en bout** :

Le contrôle de flux est primordial quand un micro ordinateur communique avec un gros ordinateur (cas où les vitesses de dialogue sont probablement déséquilibrées), sinon le tampon d'entrée du micro sera très vite saturé.

Ceci consiste en un contrôle de flux de bout en bout : **il s'agit de permettre au récepteur de réguler l'arrivée des données**.

On pourra aussi permettre aux systèmes intermédiaires de réguler une source, en faisant variés les délais de transmissions sur la connexions.

➢ Ces délais sont utilisés pour le calcul de temporisation de retransmission (cf p215 [12] (qui sont utilisées pour la réémission des segments ... plus la temporisation est grande ... moins on charge le réseau)



TCP utilise la technique de la **fenêtre glissante** (aussi appelée **fenêtre d'anticipation** ou **sliding window**) pour améliorer l'efficacité de la transmission et contrôler le flux des échanges.[1]

La gestion de fenêtre glissante peut aussi être présentée comme une forme plus élaborée d'accusé de bonne réception et transmission qui doit permettre **d'augmenter le taux d'utilisation du réseau**.

Elle permet aussi de réguler le trafic en fonction de la charge des routeurs et du débit des réseaux traversés.

8.13.1 La fenêtre glissante : le principe

On rappelle que l'ensemble d'un flux de données unidirectionnel d'une machine A vers une machine B est constitué d'une séquence d'octets tous numérotés individuellement.

Le mécanisme TCP de fenêtre **opère au niveau de l'octet** et non au niveau du segment ou du paquet.

① La **fenêtre glissante** va consister à fixer quels sont les octets appartenant à ce flux que A peut émettre immédiatement.

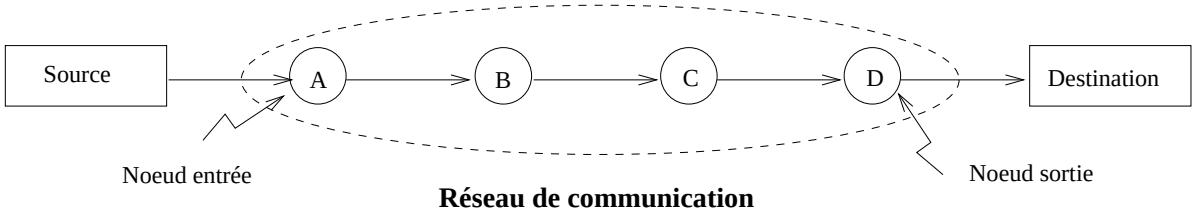
Dans cet exemple la fenêtre couvre les octets de 4 à 9, car la taille de la fenêtre courante est 6 et que tous les octets jusqu'au troisième inclus ont été émis et acquittés.

② A tout instant TCP calcule sa **fenêtre utilisable** qui est constituée des octets présents dans la fenêtre et non encore envoyés. Ces octets sont généralement immédiatement transmis.

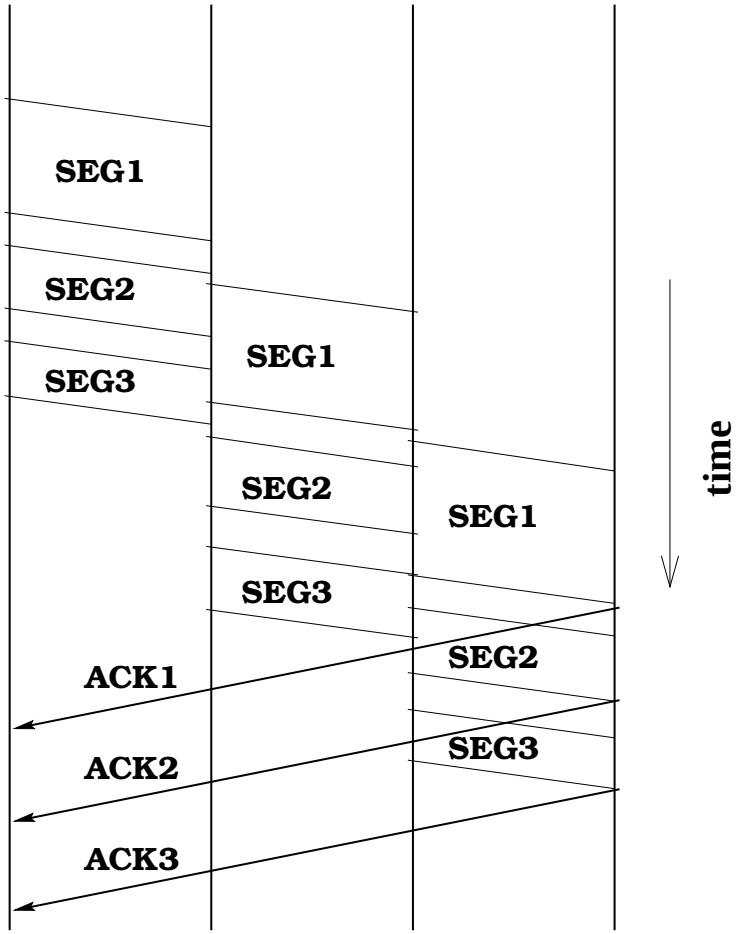
③ La transmission de données fait donc évoluer les bords de la fenêtre : **La fenêtre glisse**. Lorsque l'octet 4 est acquitté la fenêtre glisse pour permettre l'émission des octets suivants.

- La **fenêtre se ferme** quand les données sont émises et acquittées : le bord gauche avance vers la droite.
- La **fenêtre s'ouvre** quand les données peuvent être émises : le bord droit s'avance vers la droite.

Saturation du réseau



Réseau de communication

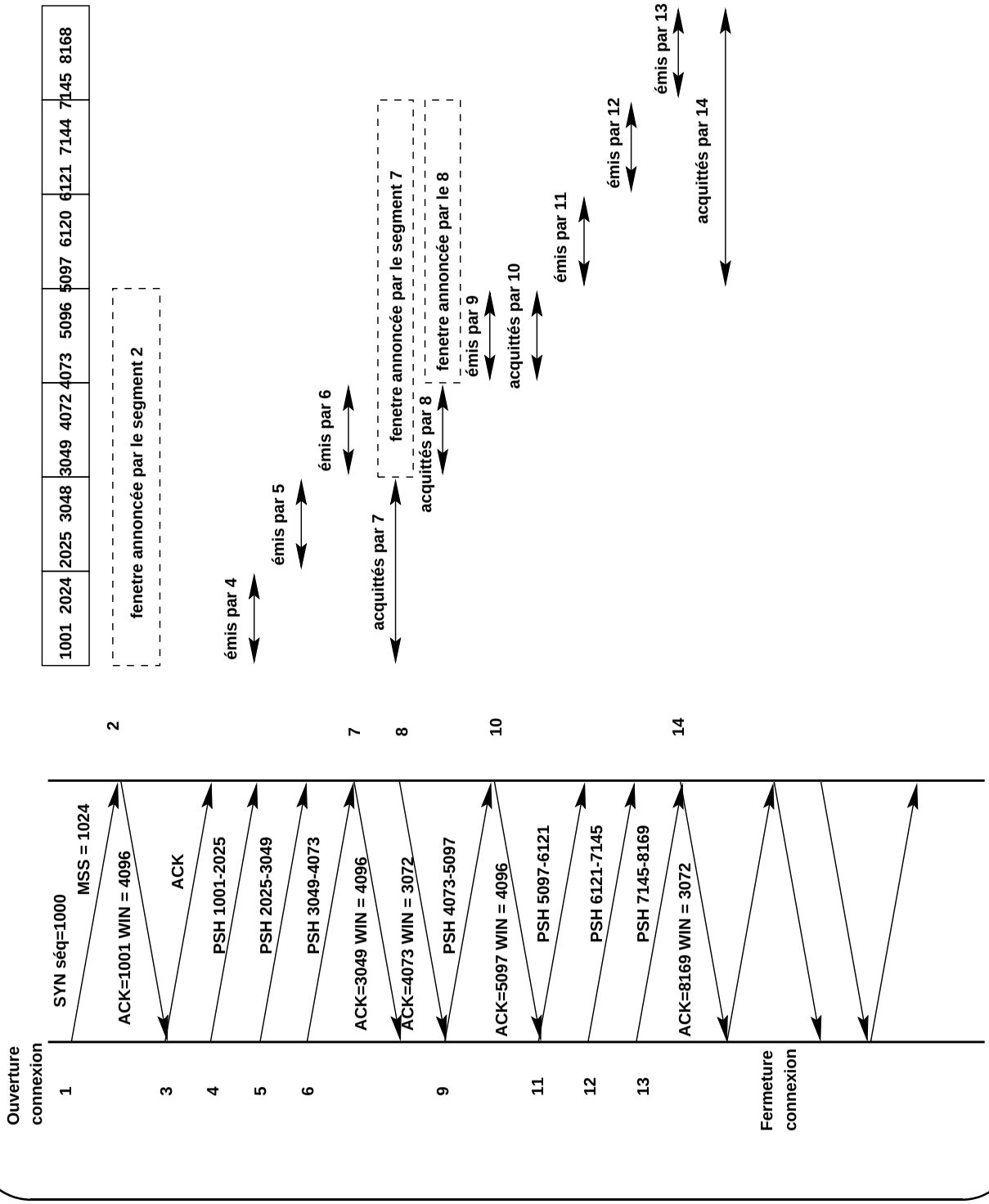


8.13.2 La fenêtre glissante : l'impact au niveau du réseau

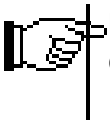
Puisqu'un protocole de fenêtre glissante bien configuré conduit à saturer le réseau de paquets, il permet d'obtenir un débit sensiblement plus élevé qu'un protocole simple d'ACK.

➢ L'émetteur peut transmettre tous les paquets qui sont dans la fenêtre sans attendre d'ACK.

Gestion de la fenêtre



8.13.3 La fenêtre glissante : la taille



Pour le flot de A vers B, la taille de **la fenêtre est contrôlée par B** qui envoie dans chacun de ses accusés de réception la taille de la fenêtre qu'il désire voir utiliser par A.

En ce sens le mécanisme de fenêtre glissante joue son rôle de contrôle de flux.

- ➢ Si la demande exprime **une augmentation, A déplace le bord droit de sa fenêtre courante et émet immédiatement** les octets qui viennent d'y entrer :
- ➢ Si la demande exprime **une diminution**, il est déconseillé de déplacer réellement le bord droit de la fenêtre vers la gauche.

  Ce rétrécissement est opéré lors des glissements de la fenêtre vers la droite avec l'arrivée des accusés de réception.

La figure qui précède illustre les échanges de segments entre un émetteur A et un récepteur B et l'évolution de la fenêtre glissante de A suivant les indications de B.

8.13.4 La fenêtre glissante : ARQ

On a vu que, dans les échanges TCP, les accusés de réception indiquent toujours le numéro du prochain octet attendu par le récepteur.

Le mécanisme d'ACK de TCP est dit cumulatif parce qu'il indique combien d'octets du flot ont été accumulés.

- L'**avantage** c'est qu'un tel mécanisme est **facile à générer et non ambigus**.
- L'**inconvénient** c'est que **l'émetteur n'a pas d'indication**, sur le segment associé à l'emplacement repére dans le flot.

Faisons l'hypothèse que la taille de la fenêtre corresponde à 5 segments et que le début de la fenêtre(de 5000 octets) corresponde à l'octet 101.

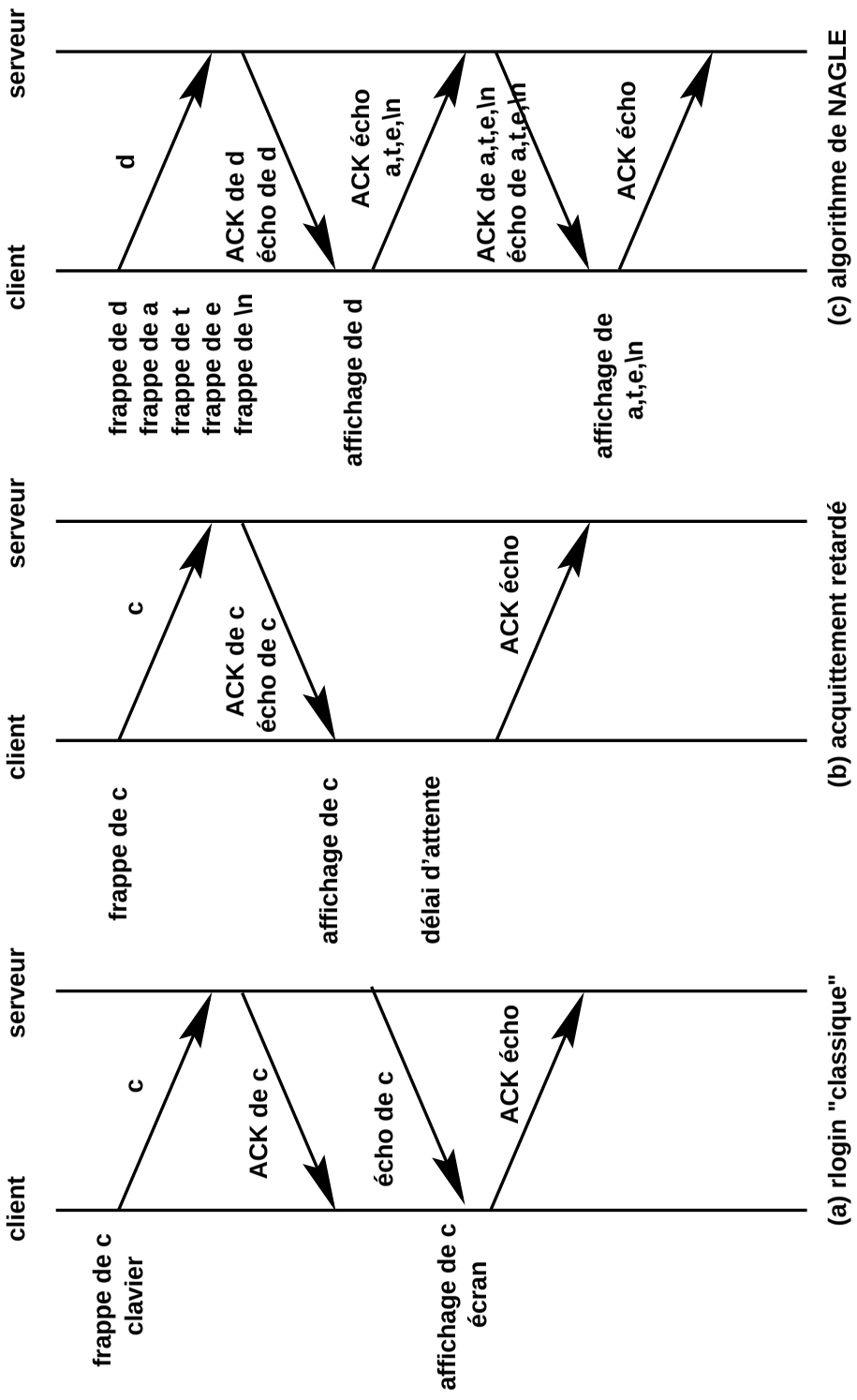
- ➢ Si le premier segment se perd et que les autres arrivent, il n'y a aucun moyen pour le récepteur de signifier à l'émetteur que 4 segments sur 5 sont bien arrivés ... il ne peut envoyer d'ACK 5001.

Au bout de la temporisation,

- soit l'émetteur renvoit les 5 segments et même si il s'arreçte d'émettre dés que le récepteur lui envoie un ACK 5001 on a quand même trés probablement surchargé sur le réseau.
- soit l'émetteur choisi de retransmettre séquentiellement (« en attendant un ACK pour ») chaque segment. Là encore on sous utilise probablement le réseau.

Aucune des solutions n'est satisfaisante !

Petits Paquets



8.14 Petits Paquets

Le transfert de données de TCP est de deux types :

① le transfert interactif dans lequel chaque segment transporte très peu d'octets, voire un seul,

② et le transfert en masse où chaque segment transporte un maximum d'octets.

8.14.1 Les données

Cette distinction est confortée par une étude de 1991 qui indique que la moitié des paquets TCP contient des données en masse (ftp, mail, news, ...) et l'autre moitié des données interactives (telnet, rlogin, ...).[1]

➢ Mais, 90% des octets transmis proviennent de données en masse et 10% seulement de données interactives car il apparaît que 90% des paquets de telnet et rlogin comportent moins de 10 octets.

Autrement dit, **la moitié des paquets émis en TCP servent à transporter 10% des données** !

➢ Ils sont donc souvent vides !

8.14.2 Nous n'avons pas les mêmes valeurs ...

En essayant d'améliorer TCP, on s'est trés vite aperçu qu'il y avait un sérieux problème de performances dans le cas où l'émétteur et le récepteur travaillaient à des vitesses trés différentes.

➢ A l'établissement de la connexion le logiciel TCP du récepteur (le serveur) alloue un tampon de K octets et communique cette valeur de fenêtre à l'émetteur via le ACK.

➢ Si l'émetteur envoie rapidement (du moins plus rapidement que le récepteur ne peut les traiter) des données, le tampon à la réception va se remplir.

➢ Pas de problème le récepteur va communiquer à l'émetteur une taille de fenêtre plus petite ... jusqu'à ce qu'elle atteigne 0 pour dire que son tampon est plein.

Le problème commence maintenant, dés que le récepteur consomme un (ou quelques) octets, il « s'en va tout fier » envoyer un ACK pour dire à l'émetteur qu'il dispose de place dans son tampon pour accueillir des octets.

➢ L'émetteur, plus rapide lui fournit aussitôt de quoi remplir son tampon.

➢ Et le dialogue prend ainsi la forme de petits échanges où seulement quelques octets sont échangés.

C'est le **syndrome de la fenêtre stupide** (SWS : Silly Window Syndrome).

On vient de montrer que le récepteur peut provoquer un SWS. Ce syndrome peut aussi être la conséquence de l'autorité d'un émetteur :

- une application souhaite génère son message octet par octet, et le logiciel TCP envoie dés que les données sont prêtes (attention l'application n'utilise pas de PSH) ... encore des petits paquets !

[1] Cette statistique a dû largement évoluer avec l'apparition du Web

8.14.3 L'addition est salée ...

Dans les deux cas de figure, **on transmet de nombreux petits paquets**.

D'ailleurs ces paquets, à y regarder de plus prés ne sont pas si petits. Certes ils contiennent peu d'information mais il nécessite tous les entêtes des protocoles assurant leur transport (Ethernet, IP, TCP).

➢ Les spécifications de TCP comportent maintenant des heuristiques qui permettent d'éviter ce syndrome.

Il s'agit essentiellement :

- pour l'émetteur d'éviter de transmettre de faibles quantités de données.
- pour le récepteur d'éviter d'annoncer des fenêtres trop petites.

8.14.4 Du coté du récepteur : Acquittement retardé

Règle :

Dans le cas de figure évoqué plus haut (récepteur trop lent) la règle suivante est appliquée :

Si on a envoyé une fenêtre d'indication nulle, attendre que l'espace disponible soit au moins égal à 50% de la taille du tampon ou de la taille maximum de segment pour envoyer une nouvelle indication de fenêtre.

Il y a deux façons d'appliquer cette règle :

① TCP envoie un ACK chaque fois qu'un segment arrive, mais **sans mettre de nouvelle indication de fenêtre**.

② TCP attend que la fenêtre que l'on souhaite proposer à l'émetteur soit suffisamment large, pour envoyer un ACK.

Le norme recommande d'utiliser la deuxième façon.

Attendre pour envoyer un ACK présente des avantages et des inconvénients :

Avantages :

Si d'autres données arrive pendant l'attente (le tampon n'est pas plein mais pas encore assez vide) on groupe l'ACK.

- On diminue le trafic ... c'est bon pour le débit
- Un petit retard peut même permettre d'exploiter à fond la superposition des données (piggybacking). Un exemple de transfert interactif (petite taille de paquets) qui exploite particulièrement bien le mécanisme d'acquittement retardé est celui généré par la commande rlogin lancée depuis un client vers un serveur.

  Dans ce cas de figure, tous les caractères tapés par l'utilisateur sur le client sont envoyés vers le serveur en utilisant un caractère par segment, et **ils sont ensuite renvoyés en sens inverse** par le serveur pour un écho sur l'écran du client.

  Tous les segments échangés dans ce cas là ont leur bit PSH fixé à 1.

  Dans la cas de cette application, tout segment doit être acquitté dans un sens comme dans l'autre ; ce qui devrait amener au diagramme d'échanges illustré dans le a) de la figure.

En fait, TCP gère ce type d'échange avec la procédure d'acquittement retardé qui consiste à envoyer l'acquittement en l'incluant dans un segment qui transporte également des données comme illustré dans le b) de la figure.
Pour cela l'acquittement est généralement retardé de 200ms dans le but d'attendre d'éventuelles données à transmettre, dont on pourra profiter pour transporter le ACK.

Inconvénients :

- Attendre trop c'est s'exposer à une retransmission de l'émetteur.
- Autre inconvénient (que l'on passera rapidement), TCP utilise l'arrivée des ACK pour estimer les temps de boucles qui interviennent dans le calcul des temporisations. Si on fausse les ACK .... on modifie les temporisations.
  La norme prévoit un retard max de 500ms et préconise que le récepteur renvoie un ACK (sans indication de fenêtre) aprés l'arrivée de chaque nouveau segment pour ne pas fausser le calcul des tempos.

8.14.5 Du coté de l'émetteur : Algorithme de Nagle

Côté émetteur, TCP peut retarder l'envoi d'un segment jusqú à ce qu'il ait atteint une taille raisonnable : c'est le **groupage** (ou **clumping**)
Mais retarder de combien ?

- pas assez : génération de petits paquets
- trop : on perd en intéractivité

Une solution à ce problème est l'algorithme de Nagle (RFC 896) :

- ➢ Il spécifie qu'une connexion TCP ne peut avoir qu'un petit segment non encore acquitté.
- ➢ Aucun petit segment supplémentaire ne peut être envoyé tant que l'acquittement n'est pas recu.

Si l'on reprend l'exemple du rlogin, dans ce contexte la frappe de la commande date sur le client provoquerait quand même l'échange de 15 datagrammes IP de 41 octets :

- 15 caractères car date contient 4 caractères + le newline et donc $5 * 3$ segments (utilisation du schéma d'acquittement retardé)
- 41 octets car 1 octet pour le caractère + 20 octets d'en-tête TCP + 20 octets d'en-tête IP.

Ce type de situation (prolifération de **tinygrams** à l'emétteur) est parfaitement gérée par l'algorithme de NAGLE.
Ainsi, si un réseau a un fort débit et n'est pas du tout chargé la procédure sera celle du b) de la figure.

Par contre, dès que le temps de transfert (RTT) est non négligeable, les acquittements vont arriver plus lentement que la frappe des caractères par l'utilisateur du poste client.

- ➢ Dans ce cas, la couche TCP du client va accumuler de petits volumes de données (quelques caractères) et les envoyer dans le même segment TCP diminuant ainsi considérablement le nombre

d'échanges comme illustré dans le c) de la figure ... on réduit alors considérablement le nombre de datagrammes.

La beauté de cet algorithme est qu'il a une horloge autonome : plus les ACKs reviennent vite, plus les données sont envoyées vite.

Gestion socket :

Dans les cas où de petits messages doivent être remis sans délai (mouvement de souris dans le cas d'un serveur X) l'algorithme de Nagle doit être invalidé pour donner une impression de temps réel.

Les API sockets utilisent l'option TCP_NODELAY pour dévalider l'algorithme de NAGLE.

Table des figures

Références

[1] Auteurs : C.Macchi, J.-F.Guilbert
Editeur : Dunod Informatique,
Titre : Téléinformatique,
Année : 1979

[2] Auteurs : T.N. Saadawi, M.H Ammar, A.E. Hakeem
Editeur : Wiley Series,
Titre : Telecommunication Networks,
Année : 1994

[3] Auteur : W.Richard Stevens
Editeur : Prentice Hall,
Titre : Unix Network Programming,
Année : 1990

[4] Auteur : W.Richard Stevens
Editeur : Prentice Hall,
Titre : Unix Network Programming (Vol 1 : Networking APIs),
Année : 19980

[5] Auteur : W.Richard Stevens
Editeur : Addison Wesley,
Titre : TCP/IP Illustrated Vol 1,
Année : 1996

[6] Auteur : W.Richard Stevens
Editeur : Thomson Publishing,
Titre : TCP/IP illustré Vol 1,
Année : 1996

[7] Auteur : J.-M. Rifflet
Editeur : Ediscience (International),
Titre : La communication sous Unix, Applications réparties,
Année : 1995

[8] Auteur : G. Pujolle
Editeur : Eyrolles,
Titre : Les réseaux,
Année : 1995

[9] Auteur : A. Ferréro
Editeur : InterEditions,

Titre : Les réseaux locaux commutés et ATM,
Année : 1998

[10] Auteur : A. Tanenbaum
Editeur : InterEditions,
Titre : Réseaux 3ème édition,
Année : 1997

[11] Auteur : G. Bouyer
Editeur : Dunod,
Titre : Transmission et réseaux de données,
Année : 1995

[12] Auteur : Douglas Comer
Editeur : InterEditions,
Titre : TCP/IP Architecture, protocoles, applications.
Année : 1996 (traduction)