F2B002C : Harmonisation Mastères Systèmes et Réseaux

Christophe Lohr

Automne 2018

Sommaire

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

F2B002C - Introduction à Unix/Linux

Christophe Lohr

Automne 2018

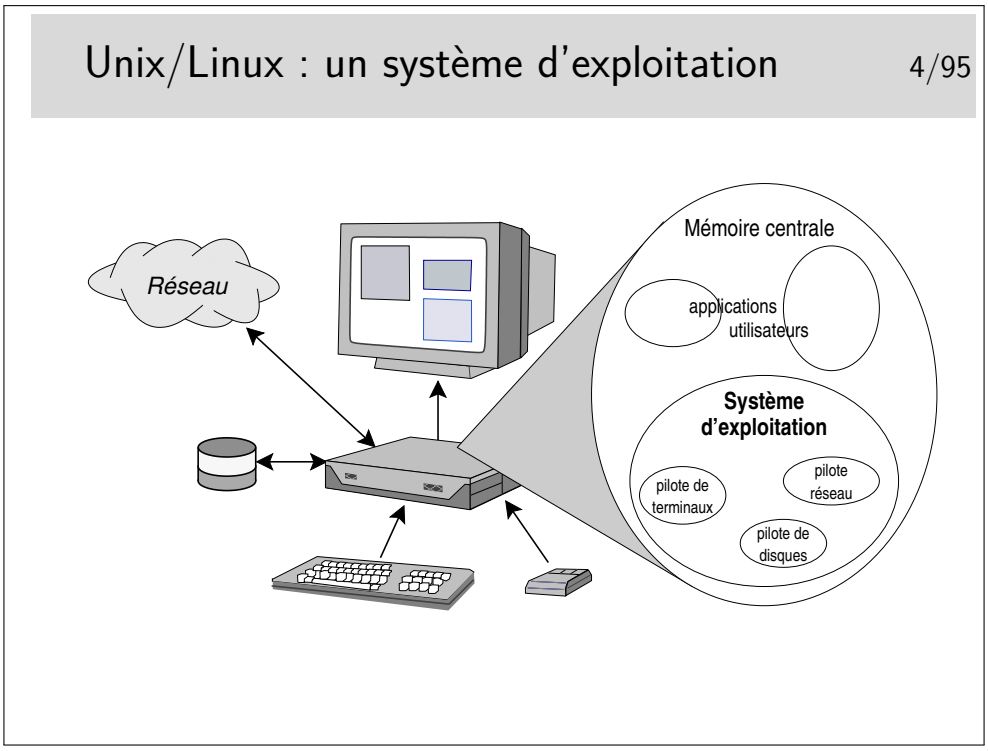
Grace Hopper - 1960

Dennis Ritchie & Ken Thompson - 1972

Sommaire

1 Introduction

1.1 Le système d'exploitation



Le système d'exploitation peut être vu comme une application particulière, chargée en mémoire centrale, très rapidement après la mise sous tension de la machine (juste après que que programme résidant en mémoire EEPROM se soit exécuté, ce qu'on appelle le BIOS dans l'architecture PC). Cette application permet d'utiliser l'ordinateur, elle gère les applications qui s'y déroulent et sert d'interface entre elles et les périphériques matériels. La mémoire centrale est chargée avec le Système d'exploitation d'une part et avec les programmes applicatifs lancés par les utilisateurs d'autre part. Le système d'exploitation contient des modules spécifiques aux périphériques qu'il sait contrôler, ce sont les pilotes (*drivers* en anglais)

1.2 Historique

Historique 6/95

- Unix naît officiellement le 1er janvier 1970 dans les laboratoires Bell AT&T : Ken Thompson et Dennis Ritchie
- Années 70 : développement d'Unix : 1973 langage C...
- Années 80 : deux filières
  - Univ. Berkeley : système Unix BSD (Berkeley Software Development)
  - AT&T : Unix Système V (déjà...), version commerciale standard
  - Sun (création 1983) et Digital (Dec) choisissent BSD : SunOS (jusqu'à la version 4) et Ultrix (Dec)
  - 1984 : Richard Stallman crée la Free Software Foundation et la Licence Publique Générale (GNU GPL)

Historique 7/95

- Années 90 : La version Système V s'impose, Sun s'y rallie (Solaris ou encore SunOS-5), Digital adopte une autre version développée par le consortium OSF, HP-UX (HP), AIX (IBM) sont des systèmes V.
- Un trublion apparaît : Linus Torvalds qui écrit le noyau Linux
- Les versions BSD continuent en logiciel libre : FreeBSD, OpenBSD...

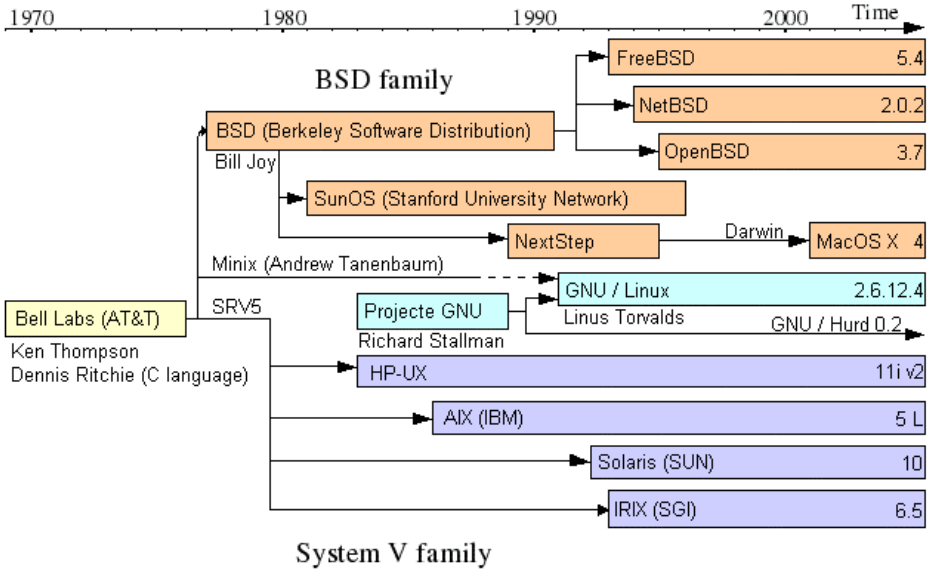
Pointeurs :

— `http://en.wikipedia.org/wiki/Unix`

— `http://virtual.park.uga.edu/hc/unixhistory.html`

— `http://www.princeton.edu/~hos/mike/transcripts/thompson.htm` (interview de Ken Thompson)

— `http://www.unix.org/what_is_unix/history_timeline.html`



Historique 8/95

- Le bazar ou la cathédrale (livre d'Eric S. Raymond)
  - Cathédrale : développement encadré
  - Bazar : on part dans toutes les directions et on choisit la meilleure (Linus Torvalds)
- Le développement de Linux appartient au deuxième style
  - Tout le monde peut participer
  - Les codes sont libres, ouverts à tous et lisibles par tous et donc sûrs...
- Résultats
  - De nombreuses distributions
  - Les logiciels évoluent très rapidement
  - Il faut suivre ! ! ! ! !

Mentionnons au passage l'une des premières (mais pas encore obsolète) documentation en français (et sous licence libre) sur Linux : *Le Guide du Routard de Linux*, de 1998, par Éric Dumas (et bien d'autres), `ftp://ftp.lip6.fr/pub/linux/french/docs/GRL/` `http://www.linux-france.org/article/grl/`.

1.3 De Unix à Linux

Linux : un système Unix 10/95

- Philosophie d'Unix :
  - (presque) tout s'utilise comme un fichier
  - "Do one thing, do it well" (Doug McIlroy, l'inventeur des *pipes Unix*) :
    - Write programs that do one thing and do it well.
    - Write programs to work together.
    - Write programs that handle text streams, because that is a universal interface.
- Caractéristiques d'un système d'exploitation Unix :
  - Multitâche (multi processus)
  - Multi utilisateurs
- Spécificités (de Linux et de tous les Unix) :
  - Son Système de Gestion de Fichiers
  - La gestion des processus
- $\rightarrow$ Linux c'est (une implémentation) Unix

...vocabulaire... 11/95

Linux (p.ex. Linux 3.2.0)

Le noyau, uniquement !

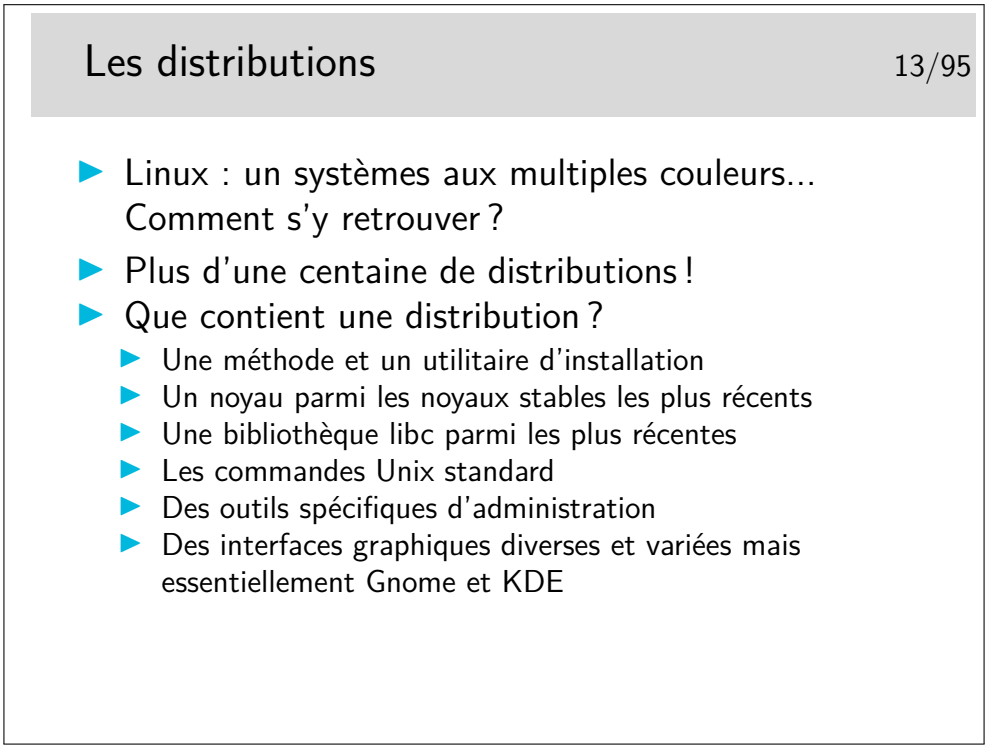
GNU/Linux

+ commandes Unix de base (implémentation de GNU) copier un fichier, répertoire, permissions utilisateur..

une distribution Linux (p.ex. Ubuntu 12.04, Debian Wheezy)

+ organisation des fichiers, outils d'administration, applications

1.4 Les distributions de Linux



Comment choisir sa distribution ?

— Liste la plus à jour : `http://lwn.net/Distributions/`

— Le catalogue proposé sur Wikipedia
`http://fr.wikipedia.org/wiki/Distribution_Linux`
`http://en.wikipedia.org/wiki/Category:Linux_distributions`

— Un bon lien pour une comparaison plutôt exhaustive des diverses distributions :
`http://www.distrowatch.com`.

— Un autre, intéressant aussi : `http://www.linux.org/dist/index.html`

Compatibilité des distributions 14/95

- Noyau *interchangeables*
  - On doit pouvoir charger un nouveau noyau sans qu'il y ait d'impact sur les commandes du système et ses outils
- Organisation des fichiers *standardisée*
  - *Filesystem Hierarchy Standard* pour l'arborescence de base
- Commandes de base *standard*
  - Outils GNU, pas d'impact
- Administration *souvent spécifique !*
  - Installation
  - Outils d'administration courante
  - Gestion des paquetage (mais il existe des familles de distribution utilisant les mêmes outils)
  - Cependant les procédures d'initialisation sont généralement basées sur Unix Système-V (/etc/inittab), il est donc relativement aisé de comprendre «comment ça marche»

«Comprendre comment ça marche» à partir de `/etc/inittab`. L'idée est que `inittab` contient les opérations à lancer à l'initialisation, ce fichier est comme l'extrémité de départ d'un *fil d'Ariane* qu'il suffit de tirer pour comprendre ce qui se passe. Et commencer à *tirer* sur ce fil revient à lire le fichier en question.

Distributions 15/95

- Stations de travail
  - Debian, Ubuntu, RedHat, Mandrake, Suse, etc.
- Spéciales temps réel
  - RT-Linux, RTLinux
- Mini distributions
  - Tiennent sur quelques disquettes voire une seule
  - FloppyFW, Tomsrtbt, Linux Router, LOAF, ...
- Embarqué
  - Uclinux, μLinux, PeeWeeLinux, ...
- Live-CD
  - Knoppix, Morphix, Kanotix, ...

Sur clef USB :

— LinuxLive USB Creator `http://www.linuxliveusb.com/` (sous Windows)

— PenDriveLinux YUMI `http://www.pendrivelinux.com/`

`yumi-multiboot-usb-creator/` (sous Windows)

— Xboot `https://sites.google.com/site/shamurxboot/` (sous Windows)

— MultiBoot `LiveUSBhttp://liveusb.info/dotclear/` (sous Linux)

— etc.

Quelques sites intéressants :

— Informations sur Linux Temps réel
`http://www.linuxdevices.com/articles/AT8073314981.html`

— Liste des minidistributions
`http://dilbert.physast.uga.edu/~andy/minilinux.html`

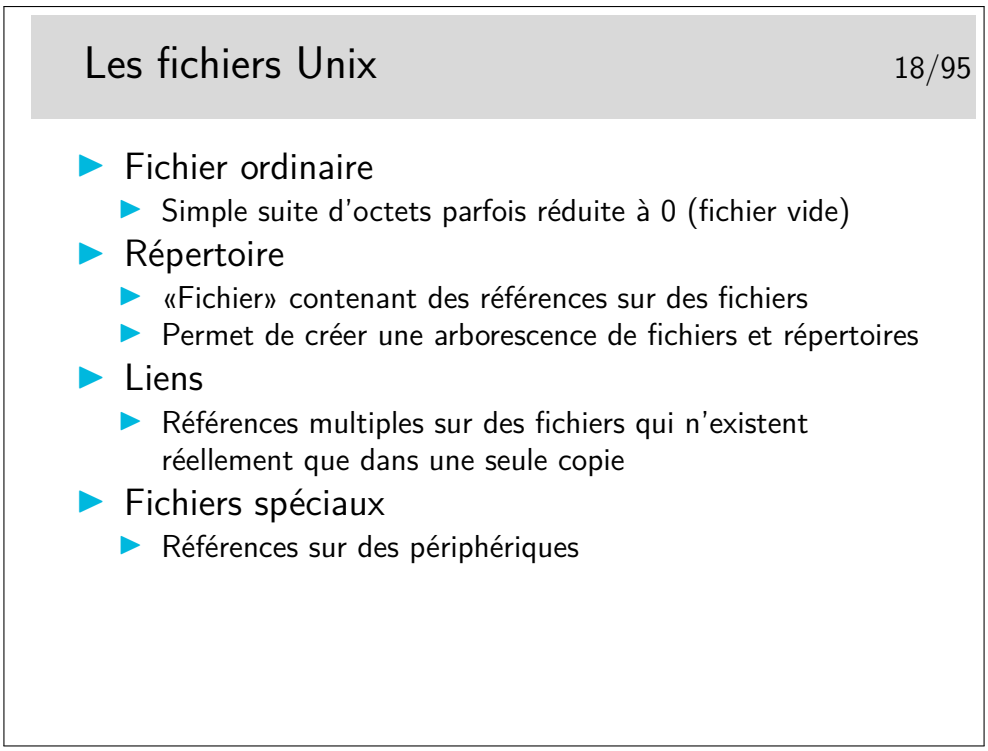
— Linux embarqué
`http://www.embedded-linux.org`
`http://www.linuxdevices.com`
`http://www.linuxdevices.com/articles/AT4525882120.html`

Les familles de gestionnaires de paquetages 16/95

- Famille Redhat
  - le gestionnaire de paquetages `rpm`
  - **Red Hat Enterprise**, **Fedora**, **CentOS**, **Mandriva/Mandrake**, **Suse**, ...
- Famille Debian
  - Gestionnaire de paquetage `dpkg` / `apt` (complémentaires)
  - **Debian**, **Ubuntu**, **Knoppix**, **Morphix**, ...
- Paquetage au format tar compressé (`.tar.gz`, `.tgz`)
  - L'utilisateur final compile les *sources*
  - Format compatible avec toutes les distributions
  - Certaines n'acceptent que cela
  - Distribution **Slackware**
  - Variante **Gentoo** : paquetages `ebuild` (outil `emerge`)

2 Le système de fichiers

2.1 Structure, nommage, droits

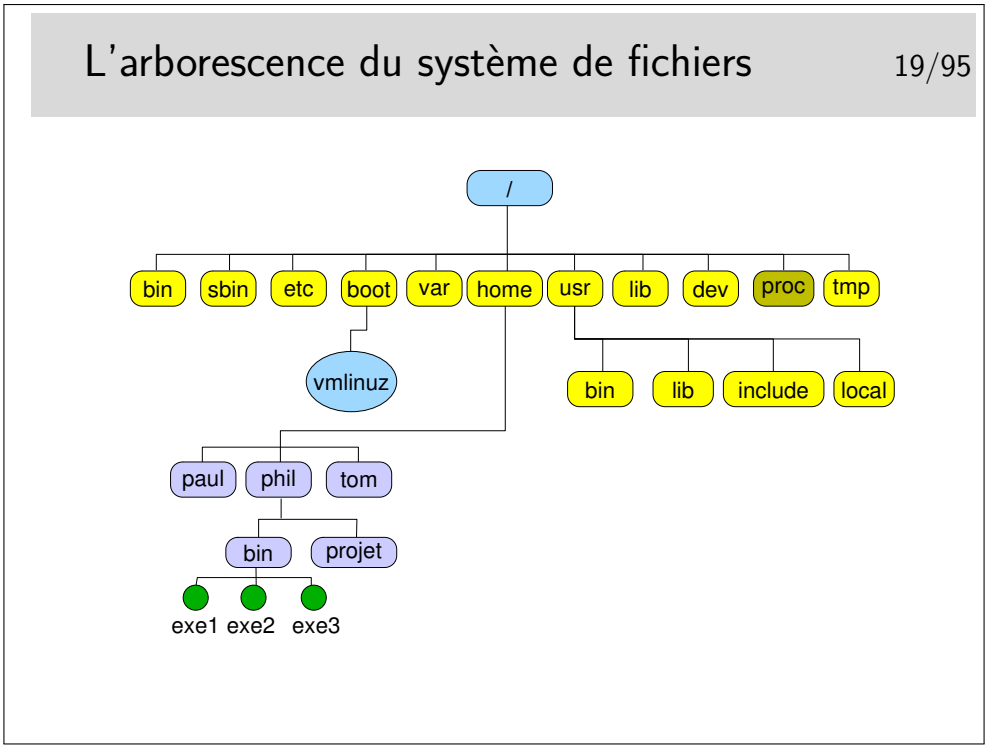


Il existe quelques autres types de fichiers pouvant apparaître dans l'arborescence :

— les sockets (type `s`) permettant la communication entre processus

— les tubes nommés (type `p` comme *pipe*) permettant aussi la communication entre processus

Les liens peuvent être de deux types : liens durs (version originale des liens sur Unix) et les liens symboliques (type l), version apportée par l'Unix de Berkeley (BSD) pour augmenter la portée des liens natifs (durs). Les liens durs ne peuvent être réalisés qu'entre références (nom dans un répertoire, on dit aussi entrée dans un répertoire) sur un même système de gestion de fichiers. Les liens symboliques peuvent passer les frontières physiques des système de fichiers.



— `/bin` et `/usr/bin` les commandes standards

— `/etc` fichiers d'administration

— `/sbin` commandes d'administration

— `/lib` et `/usr/lib` les bibliothèques (fichiers contenant les fonctions appelées par les commandes pour réaliser les opérations avec le système, exemple libc.a est la bibliothèque standard du langage C).

— `/dev` les fichiers spéciaux, référencent les périphériques du système (les *devices*).

— `/tmp` répertoire où sont créés les fichiers temporaires, il peut exister aussi /usr/tmp et /var/tmp.

— `/var/spool` répertoire où sont créées les files d'attente pour différents services tels que l'impression, le courrier électronique, etc.

— `/var/spool/cron` contient les travaux du service *cron* (voir cette référence dans le manuel de référence, faire man cron).

— `/usr/include` contient les fichiers d'entête standards des programmes en langage C.

— `/boot` répertoire contenant les composants du noyau du système.

— `/boot/vmlinuz` le noyau (le système-lui même en quelque sorte).

— etc.

Une organisation standardisée sous Unix : Filesystem Hierarchy Standard (`http://www.pathname.com/fhs/`).

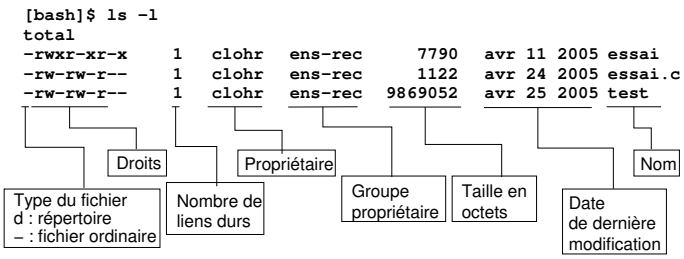
Les fichiers - Le nommage 20/95

- Nommage abolu
  - par rapport à la racine, le nom commence par /
  - /home/phil/bin/exe1
- Nommage relatif
  - relatif au répertoire dans lequel on est :

home/phil/bin/exe1	si on est dans /
phil/bin/exe1	si on est dans /home
bin/exe1	si on est dans /home/phil
exe1	si on est dans /home/phil/bin

Visualiser le contenu d'un répertoire 21/95

- La commande ls
  - Exemple :



Les fichiers cachés 22/95

- Ce sont les fichiers dont le nom commence par un point
  - Exemple : .bashrc
- Par défaut les outils d'affichage du contenu des répertoires n'affichent pas les fichiers cachés
- Ce sont généralement des fichiers de configuration d'applications
- On peut les comparer à la base de registres sur des systèmes concurrents à Unix/Linux
- Il existe aussi des répertoires cachés, leur nom commence aussi par un point

Pour visualiser la liste des fichiers cachés il faut utiliser l'option `-a` de `ls`. Pour les outils graphiques de gestion de fichiers (les «explorateurs» dirions nous sous un autre système d'exploitation bien connu), il existe une option de paramétrage dans les menus de configuration (parfois appelés «*préférences*»).

Les répertoires «.» et «..» 23/95

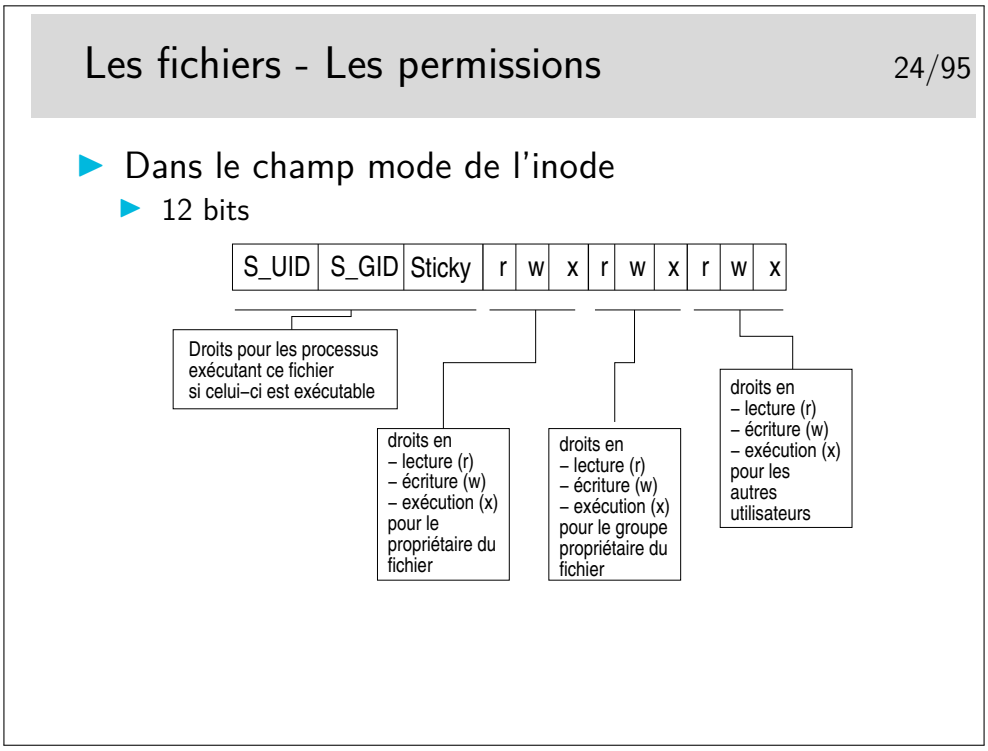
- Un répertoire n'est jamais vide, même à sa création, il contient déjà deux références sur des répertoires de nom «.» et «..»
  - Le répertoire «.» (point) constitue une référence sur le répertoire lui-même
  - Le répertoire «..» constitue une référence sur le répertoire immédiatement au dessus dans l'arborescence (le répertoire *père* en quelque sorte)
- Utilisation
  - Nommage sans ambiguïté d'un fichier local : `./test` par exemple
  - Nommage rapide d'un fichier au dessus : `../fichier` par exemple

Il arrive à beaucoup de gens d'écrire un petit fichier d'essai et de le nommer `test`. Après qu'il soit rendu exécutable, il est essayé... Et...Surprise ! le résultat n'est pas du tout celui attendu... Car la commande lancée, de nom `test`, n'est pas celle qu'on croit, il

s'agit de `/bin/test` et non le test du répertoire courant (si la variable d'environnement `PATH` ne contient pas le caractère « `.` » dans sa liste avant `/bin`).

Il suffit pour remédier à cela d'entrer alors la ligne de commande `./test` et le tour est joué.

En général, par défaut, le « `.` » ne figure pas dans la variable `PATH` pour des raisons de sécurité élémentaire et toute commande se situant dans le répertoire de travail (celui dans lequel on se trouve) doit se lancer avec la séquence `./nomFichier` (à moins que le répertoire de travail ne soit listé naturellement dans la variable `PATH`).



Si les droits en écriture/lecture/exécution sont assez évidents à comprendre il n'en va pas de même pour les trois premiers : `S_UID`, `S_GID` et le `Sticky bit` :

— Ces droits concernent les processus d'exécution du fichier, ils ne sont actifs qu'au moment où le fichier est en exécution. Ils précisent ce que le fichier (si l'on peut dire) a le droit de faire lorqu'il s'exécute.

— Lorsqu'une commande est lancée, si celle-ci correspond à un fichier exécutable, ces droits agissent comme suit :

  — si le bit *S_UID* est positionné, le processus d'exécution a les droits du propriétaire du fichier et non ceux de l'utilisateur qui a lancé la commande,

  — si le bit *S_GID* est positionné, le processus a les droits du groupe propriétaire du fichier

  — si le *sticky bit* est positionné, le processus ne sera pas purement et simplement effacé de la mémoire, il sera recopié en zone de swap et si la commande correspondant est rappelée, son rechargement en mémoire sera plus rapide. Ce n'est plus beaucoup utilisé.

Remarque importante : le *sticky bit* est maintenant utilisé sur des répertoires ouverts en lecture/écriture à tous pour restreindre le droit d'effacement des fichiers qui s'y trouvent

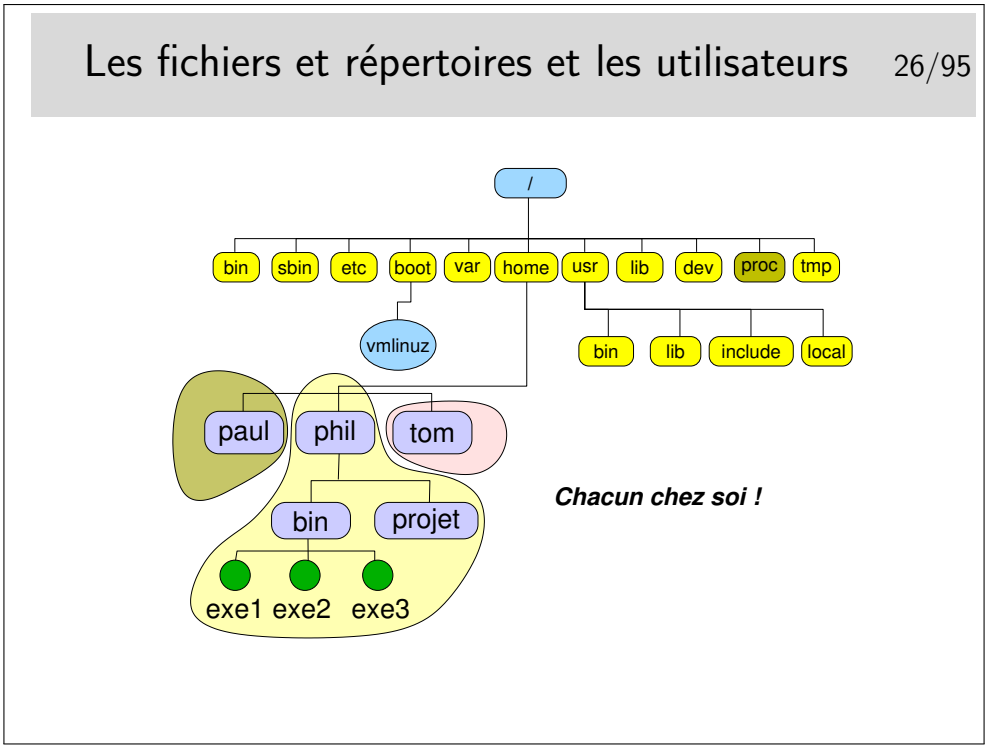
au seul propriétaire de ces fichiers. Cas des répertoires `/tmp`, `/var/tmp`, `/usr/tmp`. Voir aussi la notion d'attribut avec la commande `chattr`.

Les droits sur les répertoires 25/95

- Ce sont les mêmes types de droits que pour les fichiers : `r`, `w` et `x` et même `t`
- La sémantique associée est toutefois différente
- Droits :
  - `r` : le répertoire est lisible, on peut lister son contenu
  - `w` : le répertoire est «écrivable», on peut y créer des fichiers ou des répertoires
  - `x` : le répertoire n'est pas exécutable, il est accessible : on peut aller dedans, ou le traverser pour accéder à ce qu'il contient (fichiers, sous-répertoires)
  - `t` : le *sticky bit*, valable pour un répertoire ouvert en `w` à tout le monde, indique que seul un propriétaire de fichier peut supprimer ce fichier.

Si un répertoire est interdit de passage (i.e. le droit `x` est ôté), on ne peut pas s'y déplacer, on ne peut pas non plus descendre dans ses sous-répertoires, même si ceux-ci sont autorisés. Un droit `x` ôté est comme un obstacle infranchissable.

Si un répertoire à le droit `w` pour les utilisateurs non propriétaires, n'importe qui peut y créer un fichier, mais n'importe qui peut aussi supprimer ce fichier. Ce serait le cas pour les répertoires contenant les fichiers temporaires (`/tmp`, `/var/tmp`) si ceux-ci n'étaient pas munis du droit `t`.



Chacun chez soi... Et chacun maître chez soi. Si un utilisateur veut ouvrir son répertoire à tout le monde il en a le droit. Il a aussi le droit de fermer son répertoire à tous (même à lui même, ce qui est embêtant pour lui sur le moment, mais il peut modifier à nouveau les droits pour se ré-autoriser...).

Les répertoires ouverts en lecture sont visitables pour autant qu'ils aient le droit `x` positionné. Les répertoires `/tmp`, `/var/tmp`, `/usr/tmp` sont ouverts à tous et tous peuvent y créer des fichiers et les supprimer. Généralement ces répertoires sont munis du *sticky bit* afin de restreindre le droit d'effacement aux seuls propriétaires des fichiers.

Il est donc possible de «se promener» dans la plus grande partie du système de fichiers. Seuls certains répertoires et fichiers sensibles sont interdits.

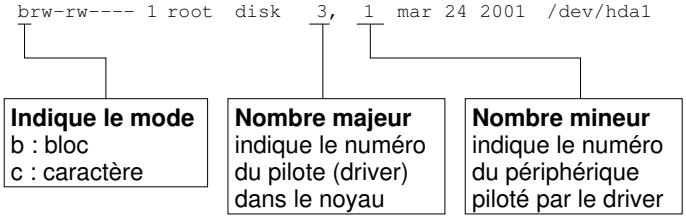
Les fichiers spéciaux 27/95

- Référencent des périphériques
  - Permettent les échanges (lectures/écritures) avec les pilotes des périphériques
  - Permettent le contrôle de ceux-ci
- Deux types
  - Les périphériques en mode bloc
    - les échanges se font par bloc d'octets (par «pages»)
  - Les périphériques en mode caractère appelé encore mode transparent (on dit plutôt mode *raw*)
    - les échanges se font octet par octet
  - Les disques sont plutôt en mode bloc, les terminaux en mode *raw*

Les fichiers spéciaux - le répertoire /dev 28/95

- Exemple d'entrée dans /dev

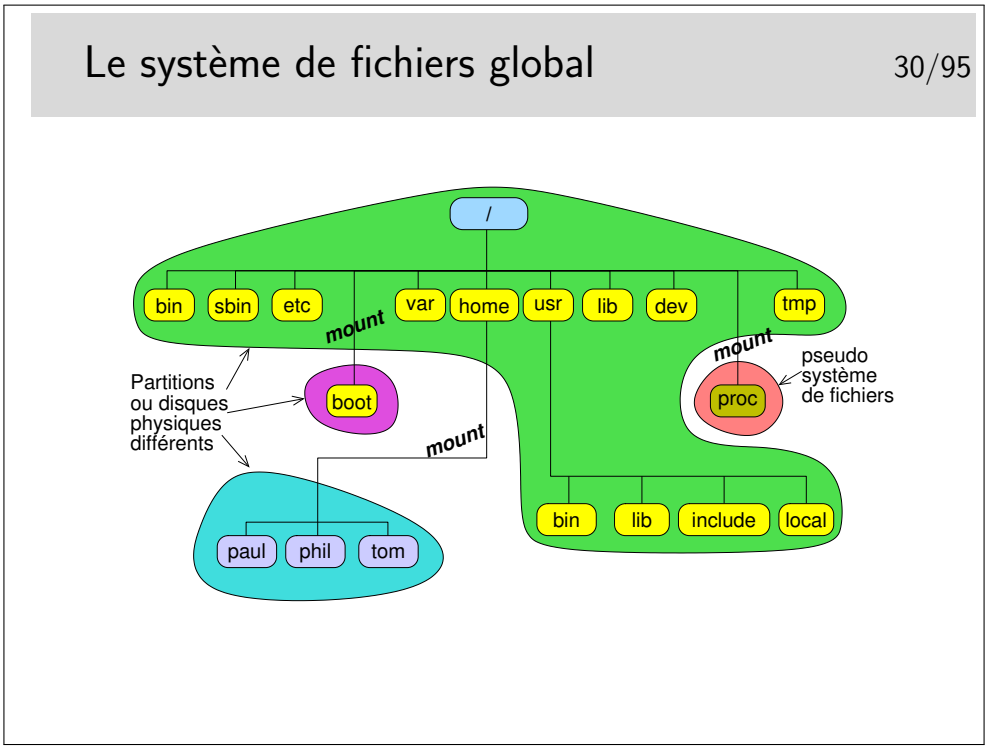
```
[bash]$ cd /dev; ls -l hda1
brw-rw---- 1 root  disk   3,  1  mar 24 2001  /dev/hda1
```



Ces fichiers sont créés par la commande `mknod`, ils peuvent être créés n'importe où mais en général on les trouve dans `/dev`.

Cette manière de voir les périphériques permet de considérer ceux-ci comme des fichiers (certes spéciaux, mais fichiers quand même), ainsi les échanges entre les applications et les périphériques se font essentiellement par des écritures et des lectures, comme pour des fichiers normaux.

2.2 Organisation sur disques



Le système de fichier global peut être constitué de plusieurs systèmes de fichiers de natures identiques ou différentes.

Dans le schéma ci-dessus on a choisi de créer 3 partitions formatées en système de fichier Linux (`ext2` ou `ext3`) :

- la partition racine avec `/bin`, `/etc`, `/var`, `/usr`, `/lib`, `/dev`, `/tmp`, ...
- une partition pour `/boot` (fait par défaut sous RedHat ou Debian par exemple)
- une partition pour les utilisateurs
- le système de fichiers proc est un pseudo système de fichiers, il n'existe pas sur disque mais est créé en mémoire vive. C'est en quelque sorte un point d'accès au noyau interne depuis le système de fichiers.
- à noter aussi l'existence du pseudo-système de fichiers `/sys` avec le noyau 2.6, donnant accès aux informations sur les «objets noyaux» tels que les bus et les drivers.

Les systèmes de fichiers sont raccrochés à la racine lors du lancement du système. D'abord le système crée la racine virtuelle qu'il nomme «`/`». Il raccroche dessus immédiatement le système de fichiers racine, puis ensuite il raccrochera les autres systèmes de fichiers selon l'ordre indiqué dans le fichier `/etc/fstab`.

L'utilitaire permettant de raccrocher un système de fichiers à l'arborescence globale est la commande `mount`.

La question pertinente est de savoir combien de partitions il faut pour installer le système. On peut laisser la procédure d'installation faire seule mais les choix automatiques ne sont pas toujours satisfaisants.

Notons que lorsqu'un utilisateur insère un CD ou une clé USB, le volume apparaît alors comme par magie sur son bureau. Techniquement, c'est bien `mount` qui est utilisé (de façon cachée) par le gestionnaire de fichier.

3 Utilisation courante

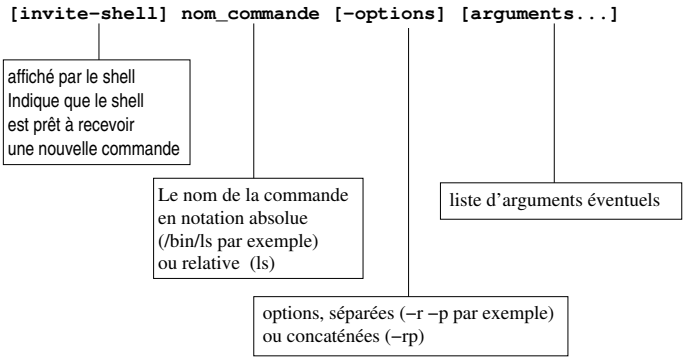
3.1 Les commandes et leur syntaxe

Comment lancer une commande 32/95

- Manière moderne
  - Via l'interface graphique
- Manière moins moderne... mais ô combien efficace !
  - Via un émulateur de terminal (xterm, gnome-terminal, ...)
  - Dans un terminal virtuel de console (accessible via `Ctrl-Alt-F1` à `F6` depuis l'interface graphique)
  - Ces méthodes lancent un processus interpréteur de commandes associé au terminal, appelé *Shell*
  - Le *Shell* est une interface entre le clavier et le système, il offre des facilités pour entrer les commandes mais aussi un vrai langage de programmation

Les commandes 33/95

- Syntaxe générale



Les commandes - La variable PATH 34/95

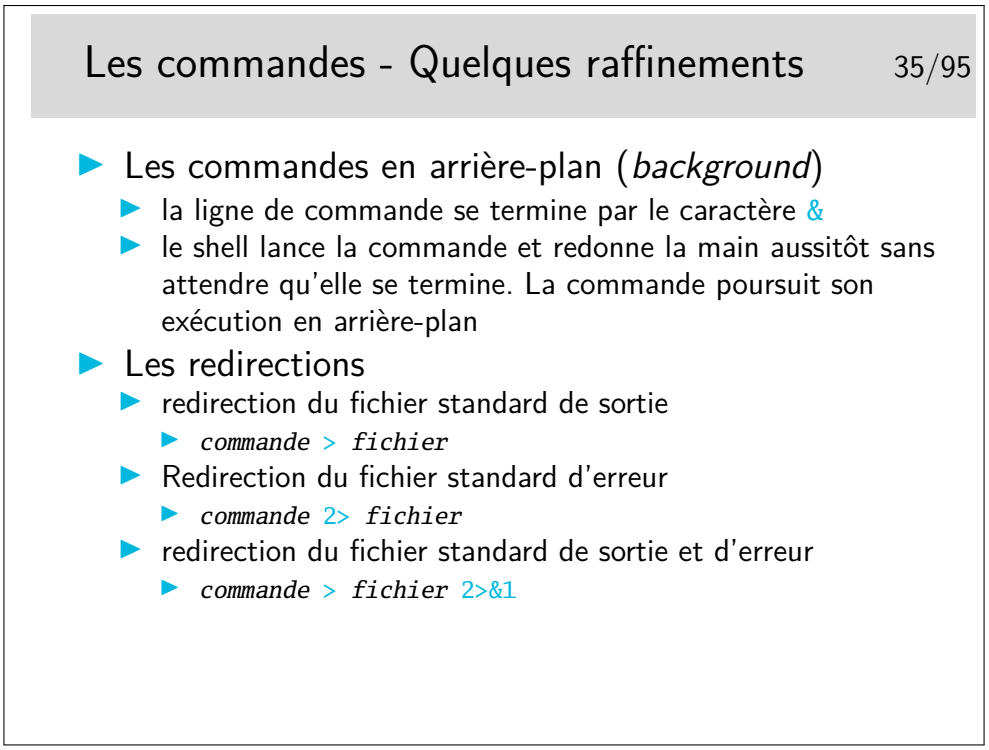
- Le PATH est une variable d'environnement
- Contient la liste des répertoires où se trouvent les commandes que l'on peut appeler par leur nom relatif le plus court : `ls` au lieu de `/bin/ls` par exemple
- Ne pas croire le système lorsqu'il vous dit : «*command not found*», il se peut que votre variable `PATH` soit mal configurée...
- Configuration dans le shell directement ou dans `$HOME/.bashrc` :
  - `export PATH=$PATH:/usr/local/bin` (par exemple)

La variable `PATH` peut aussi être configurée dans `$HOME/.bash_profile` mais ce fichier n'est pas lu systématiquement quand on ouvre une fenêtre terminale. Il n'est lu que lors de la connexion (le *login*) dans une console virtuelle (que l'on accède à partir de l'interface graphique avec le jeu de touches clavier `<Ctrl-Alt-F1>`.

Si on modifie la variable `PATH` directement dans le Shell par la commande `export` ci-dessus, la modification est prise en compte dans le shell, évidemment, mais dans CE shell seulement... Pas dans le shell de la fenêtre terminal d'à côté par exemple, et c'est très

déconcertant pour le néophyte. Pour comprendre pourquoi une telle modification n'est pas vue immédiatement de tous les shells il faut savoir qu'un shell en cours d'exécution est avant tout un processus qui hérite des propriétés du processus «terminal» lui-même (le processus qui «dessine» la fenêtre à l'écran). Ces processus forment une arborescence et héritent leurs propriétés les uns des autres d'une manière descendante. Le processus «terminal» lance le processus Shell. Le processus Shell hérite de son «père», le terminal. Si on modifie les propriétés d'un processus, les modifications pourront être vue par des sous-processus mais pas par les processus au dessus dans l'arborescence. Ni à fortiori par ceux «à coté», ou dans une autre arborescence...

Pour qu'une modification d'environnement soit permanente il suffit qu'elle soit écrite dans un fichier de configuration pris en compte systématiquement lors du lancement du Shell. Le fichier personnel `.bashrc` est le plus approprié pour le shell `bash` ; et le fichier `.tcshrc` (ou `.tcsh.PERSO`) pour le shell `tcsh`.



Les commandes - Tubes de communication 36/95

- Pour réaliser des filtres
  - *commande1* | *commande2* | *commande3* ...
  - le texte normalement affiché par *commande1* est redirigé vers *commande2* (il est lu par *commande2* et n'apparaît pas à l'écran). Le résultat est envoyé vers *commande3* et ainsi de suite
  - Exercice :
    - afficher le contenu du répertoire `/usr` à l'aide de la commande `ls -l`
    - en utilisant un tube de communication et la commande `head`, n'afficher que les 5 première lignes du résultat précédent
    - en utilisant un autre tube et la commande `tail`, n'afficher que la dernière ligne du résultat précédent
    - Voir le manuel en ligne pour savoir comment utiliser ces commandes

C'est en fait la philosophie de base de l'utilisation d'Unix : une pléiade de petits utilitaires que l'on assemble au moyen de tubes pour réaliser une grande tâche. Le contraire des approches monolithiques.

Note : lors de l'exécution de `commande1 | commande2`, les deux programmes sont exécutés en parallèle. Sur d'autres systèmes mono-tâche comme MS-DOS, il y a d'abord l'exécution du premier programme en sauvegardant sa sortie dans un fichier temporaire, puis ensuite l'exécution du second programme en lisant le fichier temporaire...

Les commandes – Les jokers (*wildcards*) 37/95

- Un caractère spécial qui en remplace plusieurs...
  - Pour les noms de fichiers
  - Un genre d'expression rationnelle (mais non POSIX)
- Exemple : `rm *` (efface les fichiers du répertoire)
- Reconnus par le shell :
  - `?` : n'importe quel caractère
  - `*` : zéro ou plusieurs caractères
  - `[A1x]` : le caractère `A` ou `1` ou `x`
  - `[a-z]` : les caractère de `a` à `z` (code ASCII)
  - `{ab,ac}` : la chaîne `ab` ou `ac`

Les commandes Unix classiques 38/95

(liste non exhaustive...)

- Créer, naviguer parmi les fichiers et répertoires
  - `ls cd pwd cp mv rm mkdir rmdir`
- Afficher — éditer des fichiers
  - `more less` — `vi emacs touch`
- Filtres texte
  - `echo cat grep sort uniq sed tail tee head cut tr split paste printf`
- Comparaison de fichiers
  - `comm cmp diff patch`

Les commandes Unix classiques 39/95

...

- Administration basique (niveau utilisateur)
  - `chmod chown ps su w who`
- Communication
  - `mail telnet ftp finger ssh`
- Shells
  - `sh csh ksh zsh bash tcsh`

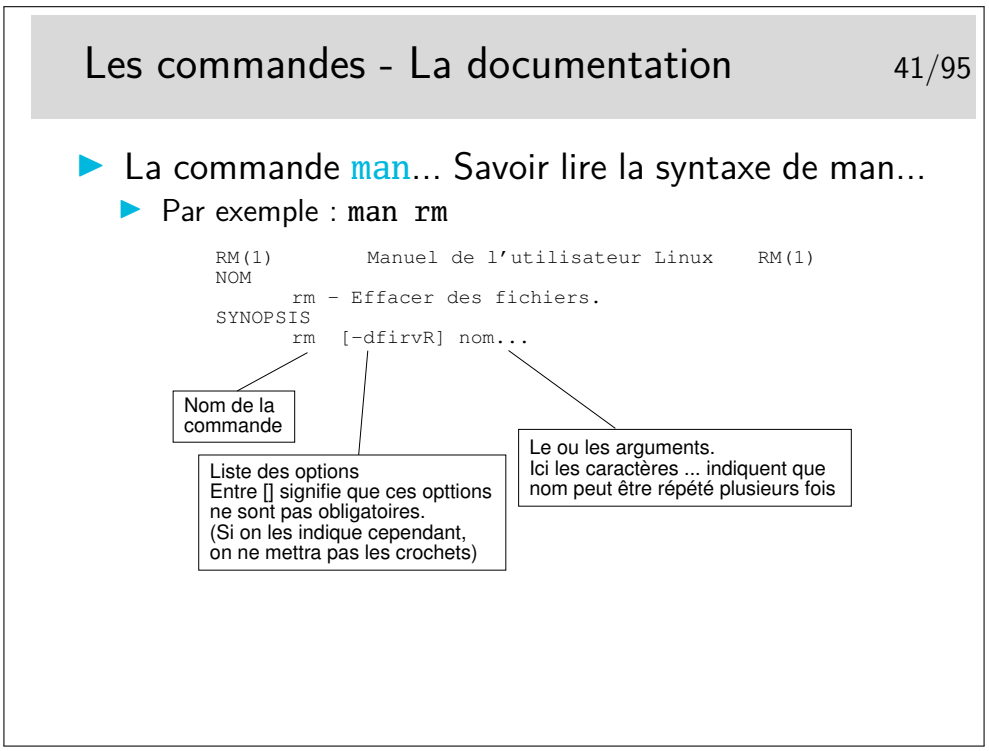
Ce ne sont quelques commandes classiques que tout utilisateur d'Unix finit par connaître par coeur au bout de quelque temps de pratique...

Même si il est évidemment difficile de les connaître toutes dans le détail, il est bon de savoir qu'elles existent et de savoir retrouver leur documentation en temps utile.

Ces commandes sont rangées typiquement dans `/bin` et `/usr/bin`

`http://en.wikipedia.org/wiki/List_of_Unix_programs`

3.2 La documentation



Les pages du «manuel» Unix, en ligne (Sun fournissait un gros classeur du man imprimé...).

La source d'information de référence incontournable. Ne pas poser une question dans les forums de discussion dont la réponse est dans le man... sous peine d'être bien mal accueilli :-)

Attention : les pages du man ne sont pas des cours ni des tutoriels, mais du condensé d'information. Chaque mot est important. Il est parfois difficile de se plonger dedans, mais les pages de man sont toutes rédigées selon la même manière, tant est si bien que lorsque l'on a pris le plis, il est relativement facile de s'y retrouver rapidement.

Le man est organisé en sections :

1. Commandes utilisateur
2. Appels système
3. Fonctions de bibliothèque
4. Fichiers spéciaux
5. Formats de fichier
6. Jeux
7. Divers
8. Administration système
9. Interface du noyau Linux

Chaque section possède une page d'introduction qui présente la section, disponible en tapant `man <num_section> intro`.

Les commandes - La documentation 42/95

- La commande `info`
  - alternative à `man`
  - généralement plus à jour
  - interface texte «à la emacs» avec menu
- La commande `apropos`
  - recherche les commandes «à propos de *xxxx*» (recherche dans les mots clefs des pages de man)

Note : les commandes *internes* au shell (p.ex. `cd pwd history` etc.) n'ont pas de page de `man` car ce ne sont pas des programmes, mais juste des mots clefs reconnus et interprétés directement par le shell. Ces commandes sont donc expliquées dans la documentation du shell lui-même. Par exemple, si vous cherchez de la documentation sur la commande `cd` et que vous avez *bash* comme shell, regardez dans le `man bash`. Une autre façon de faire est de taper `help cd` (En effet, `help` est une autre commande interne de *bash* qui affiche de la documentation sur les commandes internes de *bash*.)

Les outils de recherche de fichiers 43/95

- La commande `locate`
  - recherche les occurrences de la chaîne de caractères qui lui est passée en arguments dans une base de données mise à jour via la commande `updatedb`
- La commande `whereis`
  - Recherche le nom de commande passé en argument dans un certain nombre de répertoires standards
- La commande `which`
  - Recherche dans le `PATH` où se trouve la commande indiquée en argument
- La commande `find`
  - Recherche n'importe quoi n'importe où

4 Les processus

4.1 Environnement, cycle de vie

Les processus 45/95

- Le concept de processus
  - Dans une première approche on peut dire qu'un processus est un programme en cours d'exécution, dans un environnement donné, dans la mémoire centrale
- Notion d'environnement
  - Ensemble d'informations complémentaires au programme en exécution qui viennent paramétrer l'exécution
  - Essentiellement trois types d'information d'environnement
    - Les variables d'environnement
    - L'identité de l'utilisateur au nom duquel s'exécute le processus
    - Les fichiers ouverts (notamment ceux d'entrée/sortie standard)

Création d'un processus 46/95

- Un processus est toujours créé par le noyau, à la demande d'un autre processus
- Le processus qui demande la création est appelé le père, le processus créé est appelé le fils
- Le processus fils est créé en mémoire centrale dans une zone mémoire distincte du processus père
- Le processus fils est la copie intégrale du processus père. Mais un détail technique permet au développeur de différencier les instructions exécutées par le père de celles exécutées par le fils

Le fils est un clone du père mais il y a une mutation génétique... Le père demande la création du fils via une fonction de bas niveau (un appel système). Cette fonction rend 0 dans le processus fils et une valeur strictement positive dans le processus père. Cette

valeur dans le père est en fait le numéro de processus du fils. Le programmeur profite de cette différence pour différentier le code exécuté par les deux processus. Sinon les deux exécuteraient strictement la même chose.

Les variables d'environnement 47/95

- Chaînes de caractères, en majuscules par coutume
  - Syntaxe `NOM=valeur`
  - Regroupées dans un espace mémoire appelé «tableau des variables d'environnement»
  - Ce tableau est hérité par les processus fils et résiste à l'exécution d'une commande (voir plus loin)
- Quelques variables standard
  - `PATH`, `HOME`, `USER`, `LOGNAME`, `DISPLAY`, ...

Identité utilisateur associé au processus 48/95

- Deux identités d'utilisateur !
  - L'utilisateur réel : celui qui a lancé le processus, identifié par son numéro d'utilisateur dans `/etc/passwd` (*ruid : real user ID*)
  - L'utilisateur effectif :
    - Dans la majorité des cas il s'agit de l'utilisateur réel
    - Si le fichier exécuté (qui a donné naissance au processus) a le bit *S_UID* positionné (une lettre s apparaît à la place du x des droits du propriétaire du fichier) alors l'utilisateur effectif est le propriétaire du fichier exécuté (*euid : effective user ID*)
    - Attention : trou de sécurité potentiel, surtout si le fichier appartient à root
- Deux identités de groupe
  - Concept identique à ci-dessus : groupe réel, groupe effectif (bit *S_GID*)

Pendant l'exécution d'un fichier ayant le bit *S_UID* positionné on a les droits du propriétaire du fichier, les droits d'un autre utilisateur, sans lui demander... Mais si le fichier exécuté possède ce droit c'est que son propriétaire l'y a mis, car lui seul peut le faire, ou l'administrateur, ou une application malicieuse...

Tout utilisateur peut avoir chez lui de tels fichiers pour des raisons qui lui sont propres et pour que des applications qui lui sont personnelles puissent fonctionner pour tout le monde. L'administrateur peut toutefois demander aux utilisateurs de justifier la présence de tels fichiers, c'est son droit et peut être même son devoir.

De tels fichiers existent et appartiennent à root, l'administrateur. S'ils sont vulnérables à des attaques de type débordement de tampon (buffer overflow) alors la sécurité du système entier est gravement compromise.

Les fichiers standards d'entrée-sortie 49/95

- Trois fichiers standards
  - Le fichier standard d'entrée (descripteur 0, FILE Pointer stdin)
  - Le fichier standard de sortie (descripteur 1, FILE Pointer stdout)
  - Le fichier standard de sortie d'erreur (descripteur 2, FILE Pointer stderr)
- Ouverts par défaut lors du lancement d'un exécutable
- Associés virtuellement au clavier pour l'entrée standard et à l'écran pour les deux autres
- Ils peuvent être redirigés vers des fichiers réels ou des tubes de communication

Contrôle sur les processus 50/95

- Lister les processus
  - La commande `ps`
    - Nombreuses options : `ax`, `axl`, `axf`, `-ef`, etc.
  - La commande `top`
    - Comme `ps axu`, avec réaffichage régulier, plus des informations sur la charge et l'occupation mémoire
- Arrêter un processus
  - Si on a le contrôle (processus en premier plan dans un terminal)
    - Sans le tuer (arrêt momentané) : `<Ctrl-Z>`
    - En le supprimant : `<Ctrl-C>`
  - La commande `kill` (voir pages suivantes)

Sur certains systèmes les paramétrages des terminaux ou des émulateurs de terminaux sont tels que les associations de touches `<Ctrl-Z>` ou `<Ctrl-C>` ne fonctionnent pas. On peut le vérifier avec la commande `stty -a` qui affiche le paramétrage du terminal.

Exemple :

```
[bash]$ stty -a
speed 38400 baud; rows 25; columns 80; line = 0;
intr = ^C; quit = ^\; erase = ^H; kill = ^U; eof = ^D; eol = <undef>;
eol2 = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W;
lnext = ^V; flush = ^O; min = 1; time = 0;
...
```

Remarquer le paramètre `intr = ^C`, l'accent circonflexe indique la touche `<Ctrl>`. En fait, `intr` est le paramètre permettant de tuer rapidement un processus. Voir aussi `susp`. Et faire `man stty`.

On peut changer le paramétrage `intr` avec : `[bash]$ stty intr ^F`

Contrôle sur les processus : les signaux 51/95

- Un signal est une sorte d'interruption logicielle envoyée à un processus par le noyau après qu'un événement particulier soit intervenu
- L'événement peut être :
  - Une faute logicielle (division par 0, manipulation d'une adresse mémoire interdite, erreur d'alignement de donnée)
  - Terminaison d'un processus fils : par défaut (mais paramétrable) le père est prévenu
  - Intervention de l'utilisateur via le shell ou l'interface graphique pour tuer le processus ou le stopper ou autre (modification de la taille d'une fenêtre par exemple)
- Dans la plupart des cas le signal est fatal au processus

Les principaux signaux 52/95

Signal	Numéro	Fonction
HUP	1	Signal envoyé au processus en premier plan associé à un terminal lorsque celui-ci est fermé
INT	2	Envoyé depuis le clavier avec la combinaison de touches <CTRL-C> (par défaut)
QUIT	3	Envoyé depuis le clavier avec la combinaison de touches <CTRL-|> (par défaut)
KILL	9	Ne peut être intercepté, envoyé depuis le clavier via la commande `kill` (`kill -9` ou `kill -KILL`)
TERM	15	Envoyé via le clavier par la commande `kill` simple
SEGV		Erreur de segmentation, accès à une zone mémoire interdite
CLD		Terminaison d'un fils
WINCH		Modification de la taille de la fenêtre associée à l'application
STOP		Arrêt du processus sans le terminer. Envoyé via la combinaison de touches <CTRL-Z>
URG		Une données urgente a été reçue via le protocole TCP et est en attente de lecture (voir le cours sur la programmation réseau)
IO		Des données réseau sont arrivées et sont en attente de lecture. (voir le cours sur la programmation réseau)
USR1		Nom de signal utilisable par le développeur, à son gré
USR2		Nom de signal utilisable par le développeur, à son gré

Les raccourcis claviers (p.ex. <CTRL-C>) pour envoyer les signaux au processus en cours dans le shell sont paramétrés au niveau du terminal : `stty -a`.

La commande `kill` 53/95

- `kill` *[numéro_ou_nom_de_signal] numéro_processus | numéro_job*
- le numéro ou le nom de signal sera en général omis sauf si le résultat est négatif, auquel cas on pourra essayer le signal `KILL` (-9) qui ne peut pas être intercepté par le processus
- Le numéro de processus sera obtenu par `ps`
- Le numéro de job n'est valable que pour les processus en arrière plan (background) ou les processus stoppés. On peut le connaître avec la commande `jobs`
- Le programmeur d'application peut gérer l'arrivée des signaux (sauf le signal 9) et les ignorer ou les traiter de manière à ce que le processus se termine proprement ou ne se termine pas

Les processus zombies 54/95

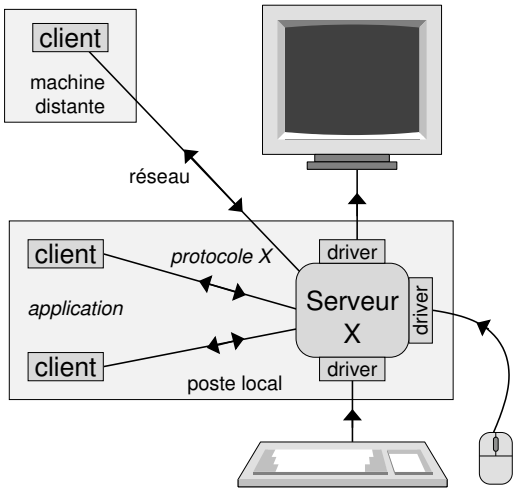
- Lorsqu'un processus fils se termine, son père doit acquitter la terminaison. C'est un problème de programmeur, pas d'utilisateur. Si l'acquittement n'est pas fait, le processus terminé reste dans la liste des processus (état `Z`), il est appelé «zombie»
- Un processus zombie est un processus fils pour lequel son père n'a pas acquitté la terminaison
- Le processus zombie est vidé de sa substance mais reste dans la liste des processus de la machine et peut être listé par `ps`
  - On ne peut plus le supprimer, il faut supprimer le père pour que le zombie disparaisse
  - Il est généralement dû à une erreur de programmation

En Shell (`sh`, `bash`, `ksh`) on peut récupérer le code de retour du exit du fils dans la variable `$?`.

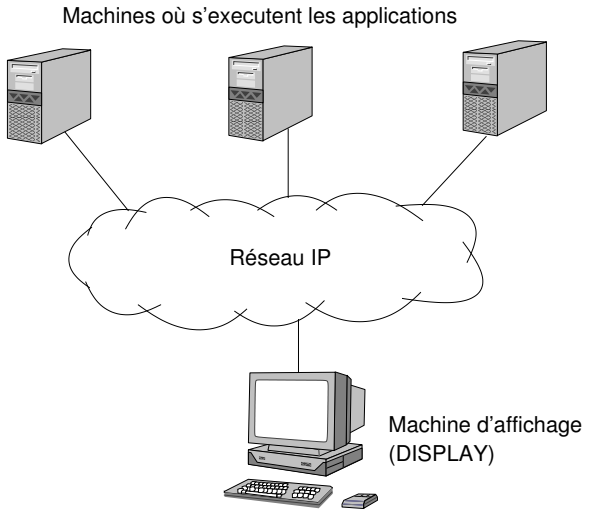
5 L'interface graphique X-Window

5.1 Client-serveur, authentification, bureau

X-Window - Principes 56/95



X-Window - Intrinsèquement Réseau 57/95



X-Window - Applications en réseau 58/95

- Toute application X peut s'exécuter sur une machine et s'afficher sur une autre
- La machine d'affichage est indiquée dans une option (`-display`) ou une variable d'environnement (`DISPLAY`)
  - Exemples :
  - `$ xterm -display lune:0.0`
  - `$ export DISPLAY=lune:0.0`
  - `$ xterm`
  - format du `display` : *adresseOuNomMachine* :*numServ*.*numEcran*

X-Window - Réseau et sécurité 59/95

- Par défaut il n'est pas possible d'afficher des fenêtres sur un «display» utilisé par un autre utilisateur si celui-ci n'a pas donné l'autorisation
- Les autorisations sont possibles avec la commande `xhost`
  - `xhost +terre`
  - autorise les «connexion graphiques» depuis la machine `terre`
  - authentification par adresse IP
- Autorisations avec la commande `xauth`
  - plus complexe mais plus *sûr*
  - fichier `.Xauthority`
  - autentifie un utilisateur (qui doit posséder le bon *cookie* de 128 bits)

X-Window - Le serveur et les applications 60/95

- Le serveur X sait gérer l'affichage mais il ne sait pas quoi ni comment afficher !
  - ce sont les applications qui l'informent via le protocole X
  - des librairies applicatives pour dessiner boutons, menu, assenceurs, etc. (*graphic toolkits*)
- Le serveur X reçoit tous les événements clavier et souris
  - il informe alors les applications de l'événement si celles-ci ont demandé qu'il leur soit envoyé
  - les applications traitent l'événement et décident de ce qu'il faut faire, elles demandent éventuellement des modifications d'affichage au serveur

X-Window - Une application très particulière 61/95

- Le gestionnaire de fenêtre ou *Window Manager*
  - le serveur X **ne sait pas gérer** les fenêtres ! ! ! !
  - une application spécifique est nécessaire pour
    - offrir des menus de fond d'écran
    - permettre de déplacer, iconifier, restaurer, supprimer les fenêtres et aussi modifier leur taille
    - C'est le **Window Manager**
  - Les Window Managers existent en grand nombre, il y en a pour tous les goûts
  - Les applications sont normalement compatibles avec tous les Window Managers (vœu pieux !)

6 Les scripts shell

6.1 Historique, fonctionnement, programmation

Les scripts shell 63/95

- Deux familles : Bourne Shell et C-Shell
- Standard (Bourne) : `/bin/sh` (voir `man sh (1)`)
- Interpréteur de commandes (terminal ou fichier)
- Possède
  - des commandes internes
  - des structures de contrôle
- Manipule des variables

Deux familles qui se distinguent essentiellement par leur syntaxe de programmation.

— Bourne Shell, le standard de System V, et finalement présent sur tous les Unix.

— C-Shell, syntaxe «à la C», développé pour BSD, apporte le «job control» (i.e. la capacité à arrêter et relancer un processus de manière interactive), et aussi l'historique des commandes tapées.

Ensuite, d'autres variantes apportent leur lot de fonctionnalités pour le confort de l'utilisateur, ou des extensions du langage et de la syntaxe de programmation.

— Korn Shell (par David Korn), entre les deux, reprend la syntaxe du Bourne Shell, ajoute le job control et l'historique.

— `tcsh`, de la famille c-shell et devenu standard sur BSD, apporte la complétion (i.e. proposer la fin d'une commande/fichier lorsque l'on a tapé le début).

— `bash` (Bourne Again Shell), de la famille de Bourne-Shell, est le shell GNU. Il est présent en standard sur tous les systèmes Linux. Apporte le job control, l'historique, la complétion, et enrichie la programmation (structures de contrôles, substitutions de variables, évaluation, tableaux, etc.). Bref, parmis les plus aboutits des shell.

— Certains puristes lui préfèrerons `dash` (Debian Almquist Shell), conforme au Bourne-Shell sans extensions du language et plus rappide.

— Des shell exotiques comme zsh qui appartiennet plus ou moins aux deux familles (mais avec une forte coloration Bourne)

Ne manquez pas de lire *"Top Ten reasons not to use the C shell"* (`http://www.grymoire.com/Unix/CshTop10.txt`). Gardez tout de même un œuil critique. Notez que si le shell par défaut qui vous est proposé ne vous convient pas vous pouvez en changer par la commande `chsh` (ou `ypchsh` si votre compte utilisateur est géré par NIS). Lire le `man` avant, et pensez qu'il y aura peut être quelques effets de bord avec des scripts de configuration.

Fonctionnement du shell 64/95

Attente d'une commande, lecture, analyse de la commande
Si la commande est une ***commande interne***
    Alors exécution la commande
Sinon Si ***fonction***
    Alors exécution la suite de commandes de la **fonction** (dans le shell courant)
Sinon
    Création d'un processus fils
    Lancement de la commande par le processus fils
    Attendre la fin
Finsi

Commandes internes (ex. en Bash) 65/95

- `cd` : changer de répertoire
- `pwd` : afficher le répertoire courant
- `echo` : afficher à l'écran
- `read` : lecture
- `export` : exportation de variables
- `eval` : évaluation de commande
- `exec` : exécution de commande
- `wait` *n* : attente de la fin du processus n
- `exit` *n* : sortie en renvoyant le code n
- `trap` *commande n* : détection et exécution au signal n
- `break` : sortie de boucle
- `set` : positionnement de variable
- `shift` : décalage des arguments
- `unset` : annulation de variable
- `help` *commande* ...

Fonctions et alias du shell 66/95

- Exécutent des suites de commandes dans le **shell courant** (pas de création de processus fils)
- Syntaxe des fonctions :
  - `function () { liste de commandes ... }`
- Utiles pour
  - configurer l'environnement interactifs
  - l'écriture de script
- Les alias
  - Permettent de renommer des commandes en les paramétrant
  - Création de nouvelles commandes
  - Exemple : `alias rm='rm -i'`

Les variables du shell 67/95

- Syntaxe : 1er caractère : une lettre ou un souligné, puis lettre(s), chiffre(s) ou souligné(s)
- Affectation
  - `VAR=‘pwd‘`
  - `VAR=`*`chaîne`*
  - `VAR="`*`chaîne de caracteres`*`"`
  - `VAR=`*`12`*
- Les guillemets (*double-quote*) masquent l'interprétation des caractères blanc (espace) et Tab
- Le caractère «\» masque l'interprétation du caractère qui le suit
- Les accents aigus (*single quote*) masquent l'interprétation de tous les caractères

Les variables du shell 68/95

- Accès : `$VAR`
- Variables d'environnement
  - `PATH` : liste des chemins accessibles
  - `HOME` : répertoire par défaut (avant-dernier champ de `/etc/passwd`
  - `PS1` : 1er prompt (généralement le caractère `$`)
  - `PS2` : 2ème prompt (généralement le caractère >)
  - `IFS` : séparateur de champ (blanc par défaut)

Variables systèmes 69/95

- ? : code de retour de la dernière commande
- # : nombre de paramètres passées à la commande en cours
- $: numéro du PID du shell courant
- ! : numéro du PID du dernier shell lancé en background

Transmission de variables 70/95

- Affectation à l'appel de la procédure
  - exemple : `prompt> TERM=vt100 nom_de_la_procédure`
- Paramètres positionnels (arguments)
  - `prompt> nom_de_la_procédure ARG1 ARG2 ARG3...` accessibles depuis l'intérieur du script par les variables définies par leur position $1, $2, $3, ...
    $0 est le *nom_de_la_procédure*

Visibilité des variables 71/95

- Les variables sont locales au script sauf si elles ont été explicitement exportées
- `export VAR`
- N.B. `VAR` et non `$VAR` : c'est la variable qui est exportée et non son contenu

La boucle `for` 72/95

- Syntaxe :
  `for` `VAR` **`in`** `w1 w2 w3 ... wn`
  `do`
  `liste de commandes`
  `done`
- autre écriture :
  `for i do ... done`
  `$i` prendra les valeurs des paramètres positionnels

La structure `case` 73/95

- Syntaxe :

```
case $VAR in
 cas1 ) liste de commandes ;;
 cas2 ) liste de commandes ;;
 ...
 casn ) liste de commandes ;;
esac
```

- Intégrations des cas :
  - **[c1-c2]** branchement si la variable est composée d'un seul caractère compris entre c1 et c2
  - **[c1c2c3]** branchement si la variable testée est composée d'un seul caractère égal à c1 c2 ou c3
  - **[xx|yy|zz]** branchement si la variable testée est composée de 2 caractères xx, yy ou zz

Structure `if-then-else` 74/95

- Syntaxe :

if *liste de commandes*
then
liste de commandes
else
liste de commandes
fi

- le test du if porte **sur le code retour de la dernière commande**

La commande `test` 75/95

- Test numérique :
  - `test X` *`option`* `Y`
  - *option* : `-eq`, `-ne`, `-gt`, `-ge`, `-le`
- Test sur les fichiers et répertoires
  - `test` *`option`* `fichier`
  - *option* : `-s` fichier existe et n'est pas vide, `-d` fichier est un répertoire, `-f` fichier est ordinaire, `-w` fichier a le droit en écriture, `-r` fichier a le droit en lecture
- Test sur les chaînes de caractères
  - `test S1` *`option`* `S2`
  - *option* : = ou `!=`
  - `test` *`option`* `S`
  - `-z` : longueur 0 ; `-n` vérifie que la chaîne a une longueur non nulle

La boucle `while` 76/95

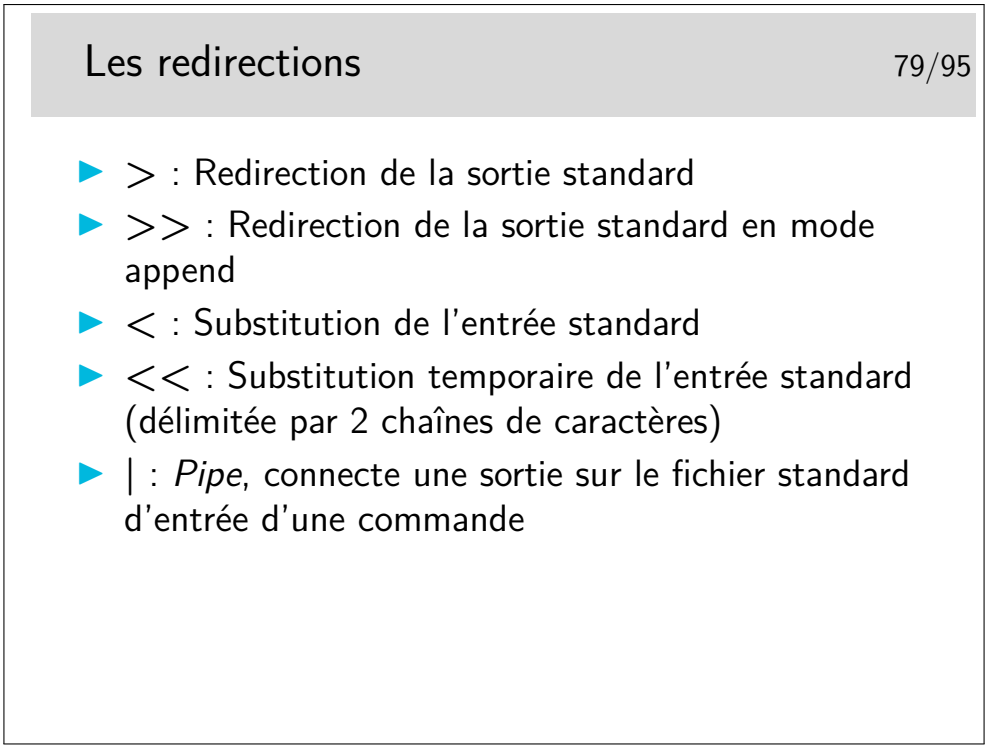
- Syntaxe :
  while *liste de commandes* **do**
  liste de commandes
  done
- NB : La boucle **while** est exécutée tant que le code retour de la dernière commande de la condition est vrai

Les substitutions 77/95

- **$*** ou **$@** tous les paramètres positionnels
- **${var}** identique à `$var` (sert à la concaténation)
- **${var-chose}** si var est définie rend `var` sinon `chose`
- **${var=chose}** si var est définie rend `var` sinon affectation de `chose` à `var`
- **${var ?message}** si `var` est définie rend `var` sinon apparition du message
- **‘accent grave‘** : `var = ‘date‘`

Méta caractères du shell 78/95

- ***** : remplacé par un nb quelconque de caractères pris dans les fichiers du répertoire
- **?** : remplacé par 1 seul caractère
- **[...]** : tout caractère compris entre les crochets sera pris en compte s'il existe dans les noms des fichiers du répertoire de travail
- **"** : le guillemet permet de passer plusieurs arguments mais n'empêche pas l'interprétation des caractères spéciaux
- **'** : la quote interdit au shell d'interpréter les caractères spéciaux
- \ : interdit l'interprétation du caractère suivant
- **()** : permet de grouper des commandes (nouveau shell)



Pour apprendre à faire des scripts shell :

— S'inspirer de scripts exisants (p-ex. dans `/etc/init.d/`), couvre 90% des besoins

— *Bash Guide for Beginners*
`http://www.tldp.org/LDP/Bash-Beginners-Guide/html/Bash-Beginners-Guide.html`

— *Advanced Bash-Scripting Guide*
`http://www.tldp.org/LDP/abs/html/abs-guide.html`

7 Paquetages logiciels : rpm, debian, Gnu tar

7.1 gnu tar, debian, red hat, etc.

Logiciels sources au format général GNU 89/95

- Téléchargeables sous forme de fichier de type archives `tar` compressées avec `gzip` (`.tgz`, `.tar.gz`) ou `bzip2` (`.bz2`)
  - `tar xvf` *paquetage*
- Contiennent un script de configuration et de création des `Makefiles` adaptés à l'architecture et à la version du système : `configure`
- Configuration, compilation, installation
  - `[bash]$ ./configure [--options]`
  - `[bash]$ make`
  - `[bash]$ make install`

Les paquetages logiciels Debian 90/95

- Trois niveaux d'utilitaires : `aptitude`, `apt`, `dpkg`
  - `dselect`/`aptitude`/`synaptic` offrent une interface texte ou graphique et permet de configurer les moyens de recherche des paquetages, de faire des suggestions, de les installer, les mettre à jour et les désintaller
  - lorsque l'on connait très exactement ce que l'on veut installe/désintaller il est plus rapide d'utiliser les commandes apt : `apt-get`, `apt-cache`, ...
  - `dpkg` pour manipuler un fichier de paquetage déjà sur le disque, ex. : lister le contenu d'un paquetage : `dpkg -L` *nomDuPackage*

Les paquetages logiciels Debian 91/95

- Exemples :

```
linux# apt-cache search linuxconf
linuxconf - a powerful Linux administration kit
linuxconf-x - X11 GUI for Linuxconf
linuxconf-dev - Development files for Linuxconf
linuxconf-i18n - international language files for Linuxcon
linux#
```

Packages Debian - Installation d'un logiciel 92/95

```
linux# apt-get install linuxconf-x
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
libwxxt1
The following NEW packages will be installed:
libwxxt1 linuxconf-x
0 packages upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 532kB of archives. After unpacking 1438kB will be used.
Do you want to continue? [Y/n] Y
Get:1 ftp://172.16.19.2 stable/main libwxxt1 1.67c-6 [486kB]
Get:2 ftp://172.16.19.2 stable/main linuxconf-x 1.17r5-2 [45.8kB]
Fetched 532kB in 1s (499kB/s)
Selecting previously deselected package libwxxt1.
(Reading database ... 30948 files and directories currently installed.)
Unpacking libwxxt1 (from .../libwxxt1_1.67c-6_i386.deb) ...
Selecting previously deselected package linuxconf-x.
Unpacking linuxconf-x (from .../linuxconf-x_1.17r5-2_i386.deb) ...
Setting up libwxxt1 (1.67c-6) ...
Setting up linuxconf-x (1.17r5-2) ...
linux#
```

Redhat Pakages Manager 93/95

- La commande `rpm`
- Permet d'installer (`-i`) ou de supprimer (`-e`) des logiciels :
  - `rpm -ivh nom_du_package`
  - Le nom du package peut être une URL
- Gère les dépendances entre logiciels (entre bibliothèques) : refuse d'installer si une dépendances n'existe pas (forçage possible mais dangereux)
- Permet de s'informer sur un *package*, savoir ce qu'il contient, de retrouver à quel *package* appartient tel fichier, de connaître les *packages* installés

Redhat Pakages Manager 94/95

...

- Permet de créer un *package* à partit d'une arborescence source compilée
- gestion de la base installée : `/var/lib/{rpm | rpm.rpmsave}`
- Commande `yum` recherche, télécharge et installe un paquetage

Exemples :

— À quel paquetage appartient la commande `ls` ?

```
[linux]# rpm -qf /bin/ls

fileutils-4.0-1
```

— Le package NFS est-il installé ?

```
[linux]# rpm -q nfs

package nfs is not installed
```

En fait, le paquetage NFS porte un nom plus complexe et est peut être installé malgré tout. Essayons avec l'option `-a` qui permet, en mode *query* (option `-q`) de lister tous les paquetages installés.

```
[linux] rpm -qa | grep nfs

nfs-utils-clients-0.2.1-2mdk

nfs-utils-0.2.1-2mdk
```

— Listons le contenu du paquetage `nfs-utils-clients-0.2.1-2mdk`

```
[linux]# rpm -ql nfs-utils-clients-0.2.1-2mdk

/etc/rc.d/init.d/nfslock

/usr/sbin/rpc.lockd

...
```

Où trouver les paquetages RedHat 95/95

- Sur les CD-ROM d'installation
  - si monté à l'endroit standard :
    - `/mnt/cdrom/Redhat/RPMS`
- Sur le web
  - `http://www.rpmfind.com`
- Outils systèmes
  - `gnorpm`
  - `yum`
  - ...

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

TP - Unix/Linux

Harmonisation Mastères

Automne 2018

1 Exercice sur les commandes en réseau

Cette série de questions est également optionnelle : vous ne serez pas évalués dessus, et elles ne sont pas requises pour mener à bien vos enseignements. Cependant, l'expérience montre qu'il est bon d'avoir quelques compétences sur le sujet (si vous souhaitez travailler sur les PC de l'école depuis chez vous, transférer vos fichiers, etc.).

Remarque : traditionnellement on utilisait sous Unix les commandes `telnet` et `rlogin` pour les connections à distance. Ces commandes sont maintenant à éviter car les mots de passe sont véhiculés en clair sur le réseau. Sous Linux, le serveur pour `rlogin` n'est d'ailleurs plus installé de manière standard sur les distributions courantes. Il faut préférer `ssh` que nous allons voir maintenant.

	Travail	Réponse
1.1	Repérez le nom d'une autre machine que la vôtre dans la salle. Connectez-vous dessus par `ssh`. Vérifiez que vous êtes bien sur la machine en question grâce à la commande `uname -n`.	- -
1.2	Dans quel répertoire êtes-vous arrivé avec `ssh` ? Listez son contenu. Est-il semblable au contenu de votre répertoire d'accueil sur la machine physique sur laquelle vous travaillez ? Pourquoi ?	- -
1.3	Déconnectez-vous en fermant votre session `ssh` par `exit` (ou `logout`). Un répertoire caché `.ssh` est créé ; que contient-il ?	-
1.4	Après avoir regardé sa page de `man`, lancez la commande `ssh-keygen`, et contentez-vous de faire `Entree` aux diverses questions posées (vous avez confiance ou vous avez lu le `man` ?). Que s'est-il passé, et que contient le répertoire `.ssh` ?	- -
1.5	Ajoutez votre clé publique ainsi généré dans votre liste de clés autorisées (`cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys`) [1]. Reconnectez-vous à la machine voisine. Que se passe-t-il ? Pourquoi ?	-
1.6	Exécutez la commande `ssh -X` *`machine_voisine`*. Dans la fenêtre de connexion sur la machine distante lancez la commande `xterm`. Que constatez-vous sur la machine locale ? (Ce résultat est obtenu grâce à l'option `-X`. [2])	-

1. Le script `ssh-copy-id` permet de copier sa clef publique sur un compte distant.
2. Cela peut être une bonne idée d'utiliser l'option `-C` en complément.

1.7	Si vous disposez d'un compte différent sur un autre machine Unix (p.ex. au RésEl ?), copiez chez vous un fichier depuis cette autre machine avec la commande `scp`. Si non, copiez le fichier `/etc/hostname` de la machine voisine (cela a moins d'intérêt vu que vous avez des comptes partagés en réseau, c'est juste pédagogique...).	-

Remarques (à prendre en compte en dehors des séances de TP)

- Sous environnement MS-Windows il existe également des implémentations des commandes `ssh` et `scp` avec le logiciel *putty*. Ce logiciel est disponible librement à l'adresse suivante : `http://www.chiark.greenend.org.uk/~sgtatham/putty/`.
- Si vous voulez obtenir directement l'affichage des fenêtres graphiques Unix sur votre écran MS-Windows c'est un peu plus compliqué... Pour cela il y a plusieurs niveaux de solutions suivant que le serveur graphique et/ou les applications graphiques se trouvent en local sur votre PC MS-Windows ou bien sur un machine Unix distante. Voici donc trois cas de figure distincts :
  - VNC : un système open-source de bureau à distance. Cette solution est disponible sur les machines MS-Windows de l'école. Un client sur votre machine MS-Windows se connecte à une machine Unix sur laquelle se trouvent des applications graphiques et un serveur graphique qui envoie au client VNC les graphismes des applications par le protocole VNC (un genre de flux jpeg). En retour le client VNC envoie au serveur graphique les événements clavier et souris (excepté parfois le caractère `~`...).
  - Une implémentation d'un serveur X-Window sous MS-Windows. Par exemple `Xming` `http://www.straightrunning.com/XmingNotes/`. Dans cette solution, seul le serveur X est présent sur le PC MS-Windows ; et il faut se connecter sur une machine Unix pour y exécuter des applications graphiques. Celles-ci, grâce au protocole X (un peu gourmand en bande passante mais qui se compresse très bien) dialogueront avec le serveur X sur la machine MS-Windows. Les communications graphiques pouvant se faire également au travers d'un tunnel ssh (un petit tutoriel au RéSel `http://resel.fr/configuration/xming/`).
  - *Cygwin* `http://www.cygwin.com/`. C'est une implémentation sous MS-Windows d'un environnement POSIX qui peut ainsi exécuter toutes les commandes et applications classiques Unix ; et notamment un serveur X-Window, des applications graphiques, un shell, etc. Il faut télécharger un programme de «setup», le lancer et chercher l'item *X11* puis les items *xorg*. Avec Cygwin vous pouvez aussi obtenir les mêmes commandes ssh et scp que sous Unix.

Ceci décrit comment faire une connexion graphique à distance, reste la question de *«comment passer le firewall ?»*

- Tapez la commande suivante depuis votre machine locale (Linux ou Windows, équipée des commandes `ssh`), et dites ce qu'elle fait :

```
ssh -L 5901:srv-disi-vnc-04.priv.enst-bretagne.fr:5950 \
                              srv-disi-vnc-04.priv.telecom-bretagne.eu
```

- Toujours sur votre machine locale (équipé d'un client VNC), et sans fermer la connexion précédente, tapez la commande suivante et dites ce qu'elle fait : `vncviewer :1`

2 Le Shell de Bourne, initiation à la programmation

Le shell est nous offre un langage de programmation avec des structures de contrôle classiques (*if*, *while*, *for*, etc.). Il permet l'utilisation de variables, il possède des commandes qui lui sont propre (commandes internes). Les commandes Unix deviennent de véritables fonctions du shell. Ainsi nous pouvons écrire des fichiers exécutables qui enchaînent des commandes ou des programmes, ces fichiers sont en quelque sorte des commandes de plus haut niveau.

Dans un certain nombre d'établissements, pour des raisons historiques (typiquement un héritage de Sun Solaris), le shell par défaut proposé aux utilisateurs est le shell `tcsh` (c'est encore un peu le cas à Télécom Bretagne). Pour connaître votre shell faites par exemple `echo $0`.)

Pour différentes raisons [3] nous ne nous attarderons pas sur l'apprentissage de ce shell. Aussi, si vous êtes sous un shell `tcsh`, il serait préférable que vous travailliez directement en Bourne Shell, sous `bash` par exemple. Vous disposez de deux façons d'obtenir `bash` :

3. `http://www.grymoire.com/Unix/CshTop10.txt`

— De manière ponctuelle. Vous pouvez vous contenter de lancer la commande `bash` dans votre shell courant. Le prompt de `bash` a été paramétré par les administrateurs système pour ressembler à celui de `tcsh`. C'est un peu troublant (vous pourrez le changer en modifiant la variable d'environnement `PS1`), mais vous avez bien un shell `bash`.
— De manière permanente (mais non nécessairement définitive, vous pourrez rechanger pour `tcsh` si le cœur vous en dit). La commande `ypchsh` (lisez le `man`) vous permet de changer votre shell par défaut par l'un de ceux listés dans le fichier `/etc/shell` (du serveur NIS, mais c'est globalement le même que celui de la machine locale). Notez qu'il y aura un délai le temps que se propagent les pages NIS depuis le serveur et que votre cache local se mette à jours (si vous êtes pressés, connectez-vous sur une autre machine qui ne vous a pas vu depuis longtemps).

2.1 Les variables

— Créez la variable `var` et donnez-lui la valeur `10`. Vérifiez avec `echo $var` que la variable existe et possède la valeur indiquée.
— Créez la variable `VAR` et fixez-lui la valeur `VAL1`.
— Ajoutez au contenu de `VAR` le contenu `VAL2` en séparant les deux valeurs par le caractère « `:` ». Cela revient à concaténer `VAL1` avec `:VAL2`. On doit obtenir le résultat `VAL1:VAL2` dans `VAR`.

2.2 Les paramètres positionnels

— À l'aide d'un éditeur, créez le fichier `exo22` et écrivez dedans les lignes suivantes :
```
#!/bin/sh
echo $1 $2 $4 $3 $0
echo "Nombre de paramètres: $x"
echo "Numéro du processus: $y"
```
Vous remplacerez les *`$x`* et *`$y`* ci-dessus par les expressions convenables.
— Sauvegardez et rendez le fichier exécutable avec la commande `chmod`.
— Exécutez ce script de la manière suivante : `./exo22 un deux trois quatre cinq`
— Que constatez-vous ? Expliquez.
— En fin de script, ajoutez les lignes qui suivent, ré-exécutez et expliquez la commande interne `shift` :
```
shift
echo $1 $2 $4 $3 $0
```
— En fin de script, ajoutez les lignes qui suivent, ré-exécutez et expliquez la commande interne `set` :
```
set assez dit la baleine
echo $1 $2 $4 $3 $0
```

2.3 Les paramètres positionnels (suite)

— Au prompt de votre Shell (`bash`!), tapez les commandes suivantes :
```
date
d=`date`
```
(attention, ce sont des accents graves, des apostrophes inversées ou backquotes)
```
set $d
echo $1 $3
echo $d
```
— Attendez quelques instants (au moins une minute) et refaites `echo $d`.
— En déduire le rôle des accents graves et le rôle de la commande `set`.

2.4 Les commandes UNIX sont des fonctions du Shell

Ce sont des fonctions au sens où elles rendent une valeur qui peut être utilisée et testée grâce aux structures de contrôle du Shell comme `if`. La valeur rendue est stockée dans la variable «`?`». Elle est donc accessible par l'expression `$?`.

Essayez les commandes suivantes, et dites quelle est la valeur rendue :
—
```
grep root /etc/passwd
echo $?
```
—
```
grep truc /etc/passwd
echo $?
```
—
```
grep root /etc/pwd
echo $?
```

— `touch truc` *(création d'un fichier vide de nom* ***truc****)*
```
rm truc
echo $?
```
— `rm truc`
```
echo $?
```
— `rm  clohr/.cshrc`
```
echo $?
```

Vérifiez pour chacune de ces commandes si les valeurs rendues sont conformes à ce qu'annonce le manuel de référence dans les sections `EXIT STATUS` (faites `man grep`, `man rm`).

2.5 Les commandes Unix sont des fonctions du Shell (suite)

— Écrivez ceci dans un fichier `exo25` :
```
#!/bin/sh
grep $1 /etc/passwd > /dev/null
if test $? -eq 0
then
        echo "l'utilisateur $1 existe sur cette machine"
else
        echo "$1 n'est pas un utilisateur enregistré sur cette machine"
fi
```
— Rendez ce fichier exécutable (`chmod`). Et exécutez-le avec divers arguments (`root`, `truc`,...)

Attention, les utilisateurs des machines du réseau de l'école sont gérés par un service centralisé appelé NIS (Network Information Service) ou encore *Yellow Pages*. Vous trouverez donc curieux que la commande ci-dessus vous indique que vous n'êtes pas enregistrés (si vous cherchez votre nom de login). Remplacez alors la première ligne utile (`grep`...) par `ypmatch $1 passwd`).

2.6 Structures de contrôle `if` et `case`

La commande `date` vous donne normalement la date en français. Écrivez un script (`exo26`) qui vous l'affiche en anglais. (Sans jouer avec la `locale ;-)`) Vous indiquerez en particulier l'heure au format suivant : `it's 5 past 15` ou `it's 20 to 16` selon que les minutes seront inférieures ou supérieures à 30.

Si votre environnement est tel que la commande date affiche son résultat en anglais alors faites un programme de traduction anglais vers français.

Les commandes à utiliser (éventuellement) sont : `date` pour obtenir la date, `set` pour fractionner la réponse de date en mots (récupérés dans `$1`, `$2`, etc.) [Note : on peut également utiliser `read` pour ça.] La variable `IFS` contient les séparateurs permettant à `set` de fractionner les chaînes de caractères qu'on lui passe en paramètre. Pour fractionner l'heure il faudra rajouter le caractère « `:` » dans `IFS` (ou positionner `IFS` à la valeur `":"`).

La principale structure de contrôle du Shell à utiliser est `case`.

2.7 Scripts interactifs

Créez un fichier `exo27` qui intègre une séquence interactive qui pourrait être la suivante :

```
$ exo27
 Veuillez entrer votre nom et votre prénom: nom prenom
 Bienvenue prenom nom
$ _
```

Ce qui est souligné ci-dessus est entré par l'utilisateur. Ce qui est en italique est le résultat émis par votre programme. Veuillez noter l'inversion de l'affichage, le nom est affiché après le prénom alors qu'ils ont été entrés dans l'ordre inverse. Cette inversion est bien entendu le fait du programme, ce n'est pas *magique*.

Commandes à utiliser : `read`, `set`, `echo`.

Un plus serait de vérifier que deux mots ont bien été rentrés par l'utilisateur et dans le cas contraire reprendre la séquence en son début (utilisation de `test` et de la structure de contrôle `while`).

Nota : si on veut supprimer l'écho de chaque caractère renvoyé par le terminal (par exemple lorsqu'il y a un mot de passe à entrer) il faut utiliser la commande `stty -echo`. La commande `stty echo` rétablira l'écho.

2.8 Programmez en shell au standard Unix

Nous avons vu aux 2.4 et 2.5 que les commandes Unix rendaient une valeur lors de leur terminaison et que cette valeur était stockée dans la variable « `?` » du Shell. Dans cet exercice nous allons voir comment écrire un script qui se comporte de la même façon.

Reprenez le fichier de la question 2.5 et complétez-le avec la commande `exit` à laquelle vous fournirez en paramètre une valeur correspondant au résultat, à savoir : *utilisateur trouvé* → résultat VRAI, valeur rendue `0`, *utilisateur non trouvé* → résulta FAUX, valeur rendue différente de `0`.

Testez différentes valeurs de retour. Quelle est la valeur maximum (autour de 256...) ?

2.9 Comprendre les mécanismes des processus et ce qu'est une commande interne du Shell

À l'aide d'un éditeur créez le fichier `exo29` et écrivez dedans ceci : `cd /usr; pwd`

— Rendez le fichier exécutable et exécutez-le.

— Faites alors "à la main" `pwd` et notez ce que cette commande affiche, vous constatez que vous n'êtes pas dans `/usr` alors que votre commande `exo29` semble vous y avoir conduit. Expliquez pourquoi vous n'êtes pas restés dans `/usr`.

— Exécutez ce fichier à la manière d'une commande interne du Shell (en `sh` ou `bash` vous faites `.<`*espace*`>nom_fichier`, en C-Shell vous faites `source nom_fichier`) et vérifiez son résultat avec `pwd`. Constat ?

— Qu'en déduisez-vous sur la nature de cd en tant que commande ?

F2B002C - Introduction aux Réseaux

Christophe Lohr

Automne 2018

Sommaire

Cadre du cours

Objectifs

— Appréhender les concepts fondamentaux des Réseaux
  — Multiplexage, paquetisation, délais, protocoles, modélisation, ...

- Montrer comment s'appliquent ces concepts à quelques exemples de réseaux fortement utilisés en entreprise
  - Les réseaux locaux et la technologie Ethernet
    - Concepts et réalité
  - Les réseaux IP
    - Concepts, principes de fonctionnement, routage...
    - Vous devriez savoir, en fin de cours, ce qu'est un routeur, un commutateur et connaître les principes de l'architecture d'un réseau d'entreprise

Plan

- Première partie : Concepts fondamentaux des réseaux
  - Notion de protocole et modélisation
  - Les couches 1, 2 et 3 dans le contexte connecté
- Deuxième partie : les réseaux locaux
  - La standardistaion
  - La technologie Ethernet
    - Topologie et supports physique, Adressage, Notion de VLAN
- Troisième partie : Les réseaux IP
  - IP :concepts, adressage, routage
  - TCP,UDP : concepts, notion de port, contrôle de flux
  - Quelques protocoles applicatifs

Bibliographie sommaire

Références

[1] A. Tanenbaum. *Réseaux*. Eyrolles, 2003

[2] G. Pujolle. *Les réseaux*. Eyrolles, 2003

[3] P. Rolin, G. Martineau, L. Toutain, A. Leroy. *Les réseaux*. Hermes, 1997

[4] P. Toutain. *Réseaux locaux et Internet*. Hermes, 2003

[5] `http://www.iec.org/online/tutorials`

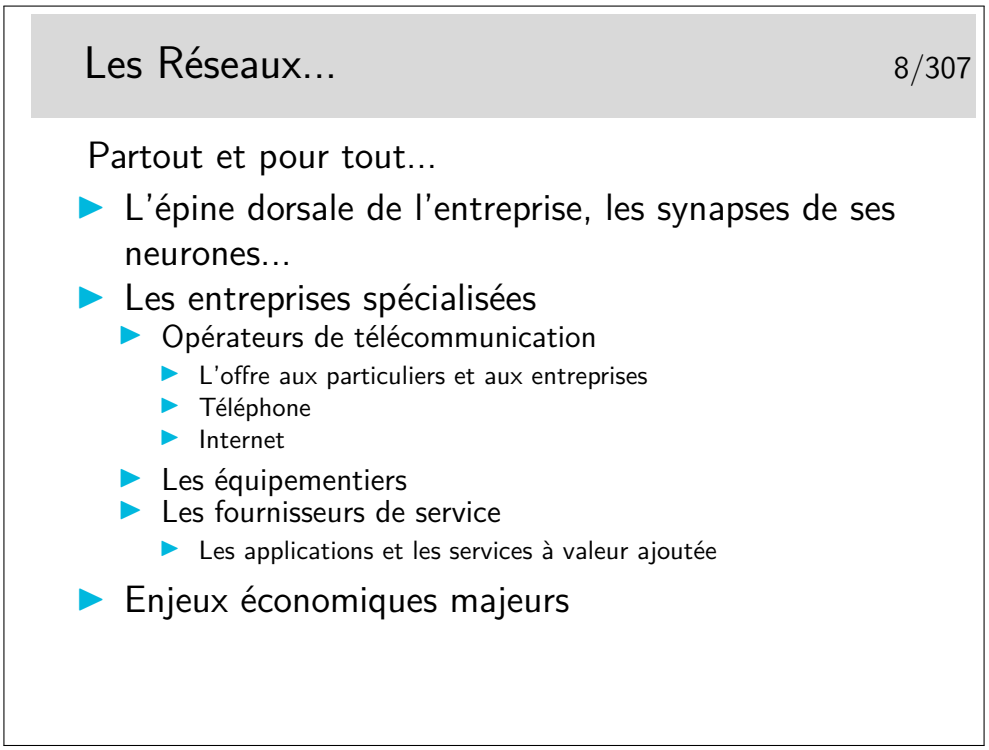
[6] `http://www.techfest.com/networking/index.htm`

Et bien d'autres...
(Voir D. Comer, W.R. Stevens,...)

58

Première partie
Introduction aux réseaux

1 Introduction



synapse : *nom féminin* (grec *sun,* avec, et *aptein,* joindre) Neurol. Région de contact entre deux neurones. (c)Larousse.

Les réseaux pour l'entreprise 9/307

- Le Système d'Information (le SI) : le cœur de l'entreprise
  - Des bases de données
  - Le réseau permet l'accès à ces bases
    - Par l'infrastructure : câbles, matériels, machines...
    - Par les applications
      - Ne pas oublier les applications
      - Les logiciels de groupware (travail en groupe)
      - Courrier, agenda partagé, workflow (suivi de documents électroniques), ...
- L'entreprise répartie
  - Agences dispersées, il faut les relier au siège
    - Dispersion à l'échelle mondiale parfois

Réseaux : le mariage des Télécommunications et de l'Informatique 10/307

- Le monde des télécommunications
  - Histoire plus que centenaire
  - Une culture : le téléphone
  - Des principes forts, une normalisation très structurée
- Le monde de l'informatique communicante
  - Histoire récente
  - Au départ pour l'entreprise, en interne, pas de facturation
  - Des acteurs divers
    - Fournisseurs de matériels et de logiciels, Utilisateurs, Chercheurs
  - Une réactivité et innovation fortes
  - Standardisation rapide et légère
- Les deux mondes convergent

Un choc des cultures...

On sait facturer dans un monde, dans l'autre on ne se pose même pas le problème... Mais vient l'heure des comptes... C'est un exemple de différence des cultures, il y en a d'autres, portant sur les principes mêmes du fonctionnement des réseaux : le mode connecté du monde des télécommunications et le mode datagramme du monde de l'informatique (spécialement dans les réseaux locaux et Internet). Nous aborderons ces concepts dans le cours.

Les deux mondes ont eu du mal à coopérer.

L'objectivité qui caractérise les scientifiques cède parfois la place à une «légère» subjectivité...

Réseaux : l'explosion du nombre de services 11/307

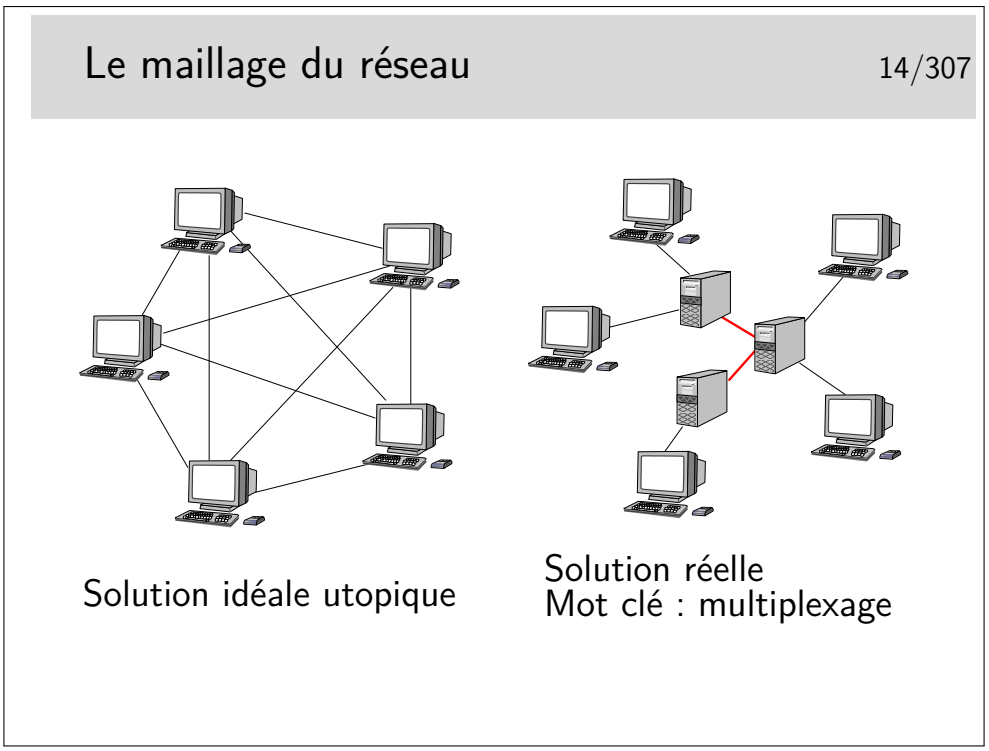
- Les services logiciels : ce sont les applications
  - Logiciels pour ordinateurs de bureau (portables ou non)
  - Logiciels embarqués
    - Sur les portables de type téléphone
    - Sur les portables de type PDA
    - La combinaison des appareils et des services
      . Le téléphone portable qui devient appareil photo et PDA,
      . média-center / média-renderer, domotique, M2M, etc.
- Les services matériels
  - La technologie va très vite

    (à peine avons nous le temps de nous habituer au modem V92 que l'ADSL est arrivé, et maintenant le FTTH pointe son nez)
  - Les technologies sans fil s'imposent rapidement

    (IEEE-802.11{abg}, Bluetooth, Zigbee)

2 Les concepts fondamentaux

2.1 Multiplexage



Les terminaux ne peuvent pas être tous interconnectés. Cette solution serait trop coûteuse en nombre de liaisons et ces dernières seraient la plupart du temps sous utilisées.

Les terminaux sont reliés à des machines intermédiaires, des relais, qui concentrent le trafic et acheminent les divers flux d'information sur des supports qui les relient.

Les divers flux de trafic sont acheminés sur les liens inter-noeuds de manière simultanée ou quasi simultanée. On dit qu'ils sont *multiplexés* et les liens entre les noeuds sont appelés de *multiplex*. Une des fonction des noeuds est d'assurer le *multiplexage* des informations sur les liens.

Un multiplex est donc une voie de communication sur laquelle on véhicule plusieurs «communications» à la fois. (notez les guillemets, il reste à définir ce qu'est une «communication», ce n'est pas si simple)

Multiplexage et multiplexes 15/307

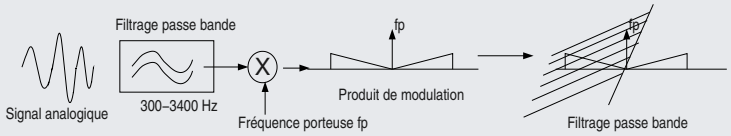
Différents types de multiplexes

- Le multiplexage en fréquence
- Le multiplexage temporel analogique
  - Par échantillonnage du signal origine
- Le multiplexage temporel numérique
  - Par échantillonnage et numérisation
- Le multiplexage statistique
  - Par acheminement sur canal commun de segments d'informations appartenant à diverses communications

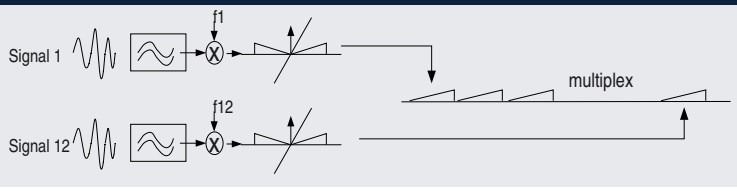
Multiplexage en fréquence 16/307

Exemple du multiplexage de canaux téléphoniques : Modulation en amplitude

Modulation d'un canal



Multiplexage



Cas du mutilpexage de canaux téléphoniques :

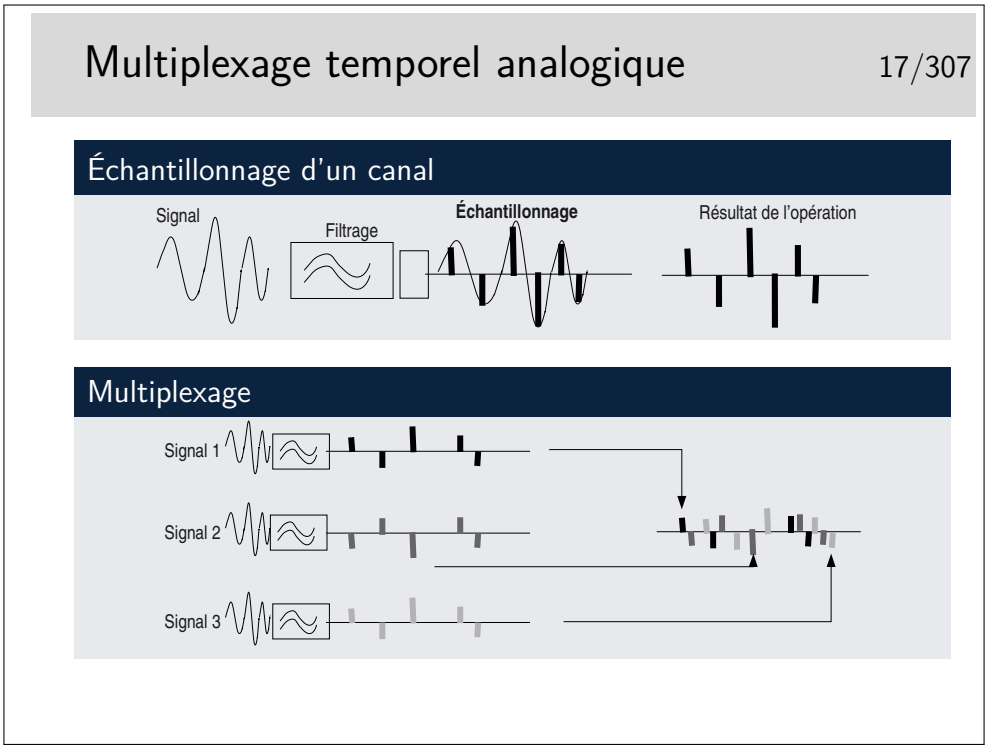
— Chaque canal est d'abord filtré dans la bande standard 300-3400 Hz. Le signal filtré vient moduler une porteuse en amplitude. Le résultat de cette modulation produit deux bandes de fréquences contenant l'information utile. Ces deux bandes sont centrées autour de la valeur de la porteuse : de fp-3400 à fp-300 puis de fp+300 à fp+3400.

— Ce résultat de l'opération de modulation est filtré pour ne garder qu'une bande

inférieure ou supérieure (ici supérieure) de largeur 4000 Hz (de fp à fp+4000).

— Le multiplexage consiste à placer sur le même support le résultat de la modulation et filtrage de 12 canaux avec 12 fréquences porteuses différentes espacées chacune de 4000 Hz.

— On constitue ainsi ce qu'on appelle un groupe primaire qui à son tour peut être considéré comme un signal analogique et qui peut venir moduler une fréquence porteuse supérieure. Plusieurs groupes primaires peuvent ainsi être multiplexés pour constituer un groupe secondaire et ainsi de suite jusqu'à quatre niveaux.

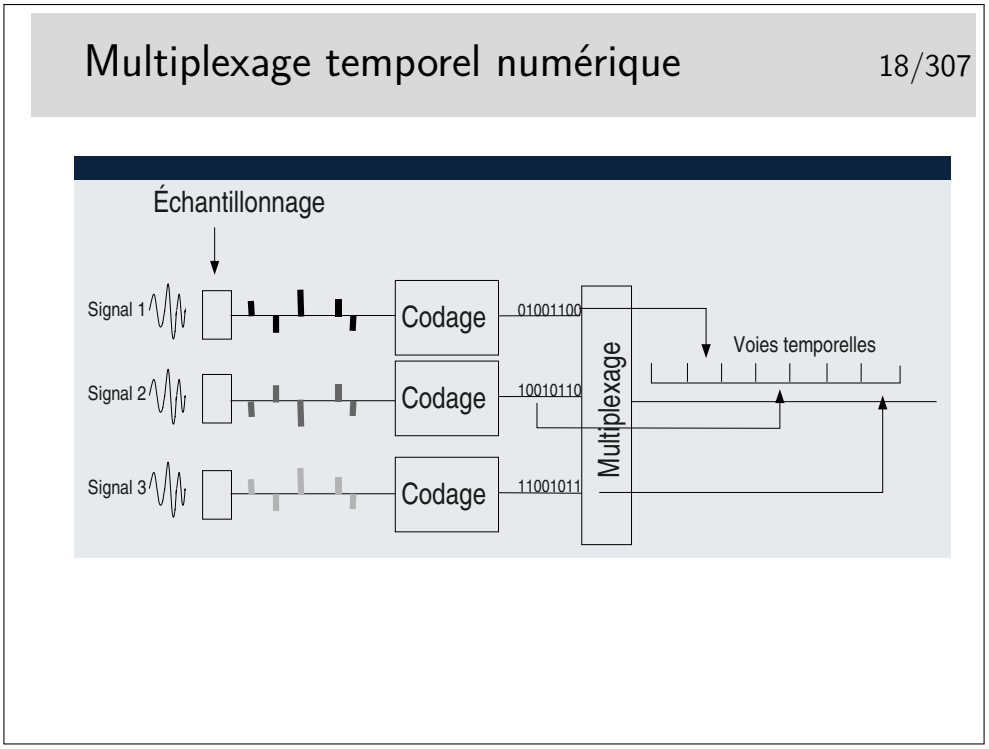


Chaque signal est échantillonné un certain nombre de fois par secondes, chacun avec un décalage dans le temps par rapport aux autres de telle manière qu'ils peuvent être véhiculés sur le même support sans se mélanger après multipexage.

La fréquence d'échantillonnage est telle que le signal doit pouvoir etre reconstitué sans déformations majeures. Shannon a montré que la fréquence d'échantillonnage devait être être le double de la fréquence maximum du signal à échantillonner. Ainsi, en téléphonie, la bande de fréquence est de 300-3400 Hz, on prend 0-4000 Hz par excès, donc la fréquence d'échantillonnage doit être de 8000 Hz.

Le multiplexage par échantillonnage analogique seul ne permet pas de bonnes performances. Le mutiplex ne doit pas être très long, quelques dizaines de centimètres par exemple dans un tout petit autocommutateur téléphonique. Si le support est plus long les échantillons se détériorent à cause des caractéristiques capacitives et selfiques du support, ils sont bruités et déformés et risquent de se mélanger.

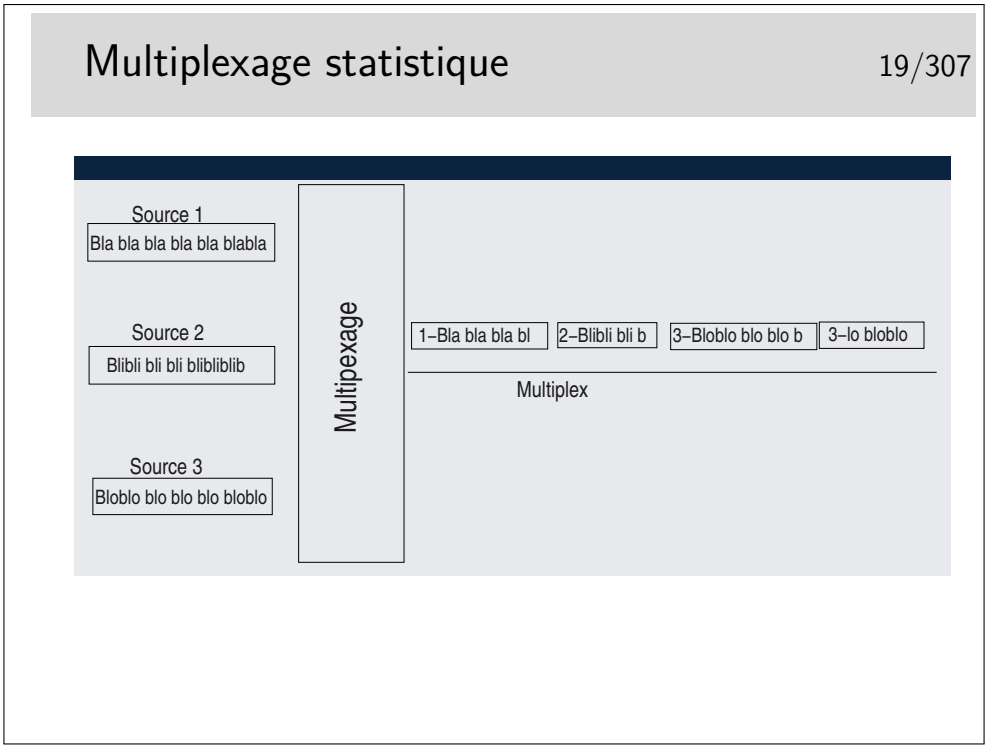
La solution réside dans la numérisation.



Chaque échantillon est transformé en un nombre binaire pouvant être de 12 bits par exemple, ramenés par des techniques de compression à 8 bits.

En téléphonie par il existe deux techniques principales de codage standardisées par le document ITU-T G711, toutes deux respectent la loi de Shannon édictant qu'il faut échantillonner à 8KHz, donc un échantillon toutes les 125μs. Elles divergent sur les points suivants :

- Loi européene dite loi A (Alaw) :
  - méthode de codage spécifique
  - 3,9μs par voie, (8 bits, 488ns par bit)
  - chaque canal a un débit de 64 Kb/s (calculez...)
  - 32 voies par multiplex à 2,048 Mb/s
  - une voie pour la synchronisation du multiplex
  - une voie pour la signalisation des voies téléphoniques
  - 30 voies utiles (ou 31 si la voie de signalisation n'est pas utilisée)
  - multiplex appelé E1 pour le téléphone
- Loi nord américaine et japonaise dite loi μ (μLaw)
  - méthode de codage spécifique
  - 24 voies par mutiplex
  - 24 voies utiles, certains bits pris sur les voies téléphoniques servent à la synchronisation et à la signalisation $\rightarrow$ 56 Kb/s par voie
  - multiplex appelé T1



Les données sont découpées en séquences de *longueur variable* qui sont ensuite injectées sur le même support, les unes à la suite des autres. Il n'y a pas d'ordre à priori. Si une seule source émet elle a toute la bande passante du support à sa disposition. Si n sources émettent simultanément elles disposent chacune de 1/n de la bande passante (si leurs données sont de taille égales).

Les séquences ne doivent pas être trop longues car alors le support est dédié trop longtemps à une seule source.

Les séquences doivent pouvoir être identifiées pour pouvoir les reconnaître à la réception et les remettre à leur bon destinataire. Elles sont munies d'une «étiquette» d'identification qui peut être un numéro de canal ou une adresse de réception.

Le canal est virtuel, il n'existe que si des données sont présentes.

Ce mode de multiplexage est utilisé pour le transfert de données. Il est universellement répandu : ethernet, IP (donc Internet), etc...

Les séquences de données sont appelées «paquets».

Parfois les paquets de données sont de taille constante, on les appelle alors des cellules (réseau de type ATM par exemple) (terme à ne pas confondre avec les cellules des réseaux de téléphonie de type GSM où les cellules sont des zones géographiques desservies par une antenne émettrice/réceptrice).

Signal analogique 20/307

Analogique

Une variation d'un signal physique, la pression d'air générée par un son, l'intensité de lumière et de couleur générée par la lumière éclairant une scène, un paysage, est transformée en un signal électrique représentatif du signal physique. Le signal électrique varie avec les variations du signal physique. Ses variations sont «analogues» à celles du signal physique, c'est un signal analogique.

Signal numérique 21/307

Numérique

Un signal électrique analogique est échantillonné (on considère de courts instants du signal analogique, des «échantillons»). Les échantillons sont alors mesurés sur une certaine échelle et à chaque pas de cette échelle correspond un nombre. L'échantillon est remplacé par le nombre, nombre lui même codé en base 2 c'est à dire une une suite de 0 et 1. L'échantillon est numérisé, il devient un élément d'un signal numérique.

Pour qu'un signal échantillonné puisse être reconstitué dans de bonnes conditions de restitution il faut qu'il ait été échantillonné à une fréquence deux fois plus élevée que sa fréquence maximale. Ainsi pour un signal de fréquence f, il faut qu'il soit échantillonné à une fréquence $2 \times f$. Intuitivement on comprend qu'on doit au moins prendre un échantillon de chaque alternance (Shannon à montré cela de manière beaucoup plus rigoureuse que la simple intuition).

Quelques remarques sur l'échantillonnage 22/307

- Nyquist (1889-1976) a montré que pour restituer correctement un signal après échantillonnage il faut que la fréquence de celui-ci soit au moins le double de la fréquence maximale à échantillonner
- Exemple en téléphonie :
  - Bande analogique : 300Hz-3400Hz
  - Fréquence d'échantillonnage minimale : 6800Hz
    - Par précaution on prend 8000Hz

Questions :

— Pour un échantillonnage de type téléphonique, quelle sera la période de l'échantillonnage ?

— Quelle sera la durée d'un échantillon si on désire multiplexer 32 signaux sur un même support physique ?

— On code numériquement chaque échantillon sur un mot de 8 bits, quel débit numérique en résulte par voie ?

— Le codage du son en mode wav (origine Microsoft, fichiers avec l'extension .wav) correspond typiquement à un échantillonnage à la fréquence de 44100Hz, chaque échantillon étant codé sur 16 bits. Quelle est la bande passante nécessaire en b/s pour transmettre un fichier .wav contenant un son stéréophonique (2 canaux) ?

Sur Nyquist : `http://www.geocities.com/bioelectrochemistry/nyquist.htm`

Avantages et inconvénients des différents types de multiplexage I 23/307

- Multiplexage temporel
  - ressource (la voie temporelle) réservée pour la durée de la communication, même si celle-ci est silencieuse : mauvaise utilisation de la ressource
  - bande passante et délais de transfert garantis
- Multiplexage en fréquence
  - mêmes avantages et inconvénients

Avantages et inconvénients des différents types de multiplexage II 24/307

- Multiplexage statistique
  - utilisation optimale du canal, si des sources sont silencieuses, le canal peut être utilisé par d'autres
  - bande passante globale partagée entre toutes les sources, pas de garantie de réservation (sauf Frame Relay et ATM au prix de complexité supplémentaire pour la réservation et le contrôle de l'utilisation)
  - pas de délai de transfert garanti
  - gigue (variation des délais) pouvant être importante
  - C'est actuellement le moyen le plus utilisé pour les données.
  - La voix et la vidéo sont mal adaptées à cette technique car les délais et la bande passante ne sont pas garantis

La paquetisation dans les réseaux de données 25/307

- Multiplexage statistique
  - Pendant qu'une source émet sur un multiplex, les autres sources doivent être silencieuses
  - Pour rendre équitable l'utilisation du multiplex, une source ne peut pas le monopoliser trop longtemps, il faut limiter sa durée d'émission
  - Les données sont segmentées en unités appelées «paquets»
  - Les paquets ont une taille maximale et parfois une taille minimale
  - Les paquets doivent être munis d'une entête, sorte d'étiquette, qui permet de les reconnaître et ainsi de savoir en réception vers quel destinataire acheminer le paquet

2.2 Notion de connexion

Communications avec ou sans connexions 27/307

- Avec connexion, comme le téléphone ?
  - Il faut chercher un chemin dans le réseau entre la source et la destination puis le réserver et l'établir
  - Lorsque la communication est terminée il faut libérer le chemin
  - Ces opération nécessitent des échanges d'informations spécifiques que l'on appelle la signalisation
  - Le chemin est appelé circuit
- Sans connexion, comme à la poste ?
  - On munit les données «d'enveloppes» contenant l'adresse de la destination et le réseau achemine ces données en les routant dans chaque nœud en fonction de cette adresse
  - Les unités de données véhiculées sont appelées des datagrammes

Avantages et inconvénients des modes avec et sans connexion I 28/307

- Mode orienté connexion
  - Avantages
    - Le chemin est toujours le même pour la durée de la connexion, les données sont reçues dans l'ordre ou elles ont été émises
    - La signalisation nécessaire à l'établissement de la connexion peut permettre de véhiculer des informations de demande de qualité de service
    - Les délais de traitement dans les nœuds sont généralement courts
  - Inconvénients
    - Il faut un certain délai d'établissement et de rupture de la connexion
    - Le chemin est préétabli et si une maille du réseau devient inutilisable (panne d'un nœud, rupture de la maille) la communication est rompue

Avantages et inconvénients des modes avec et sans connexion II 29/307

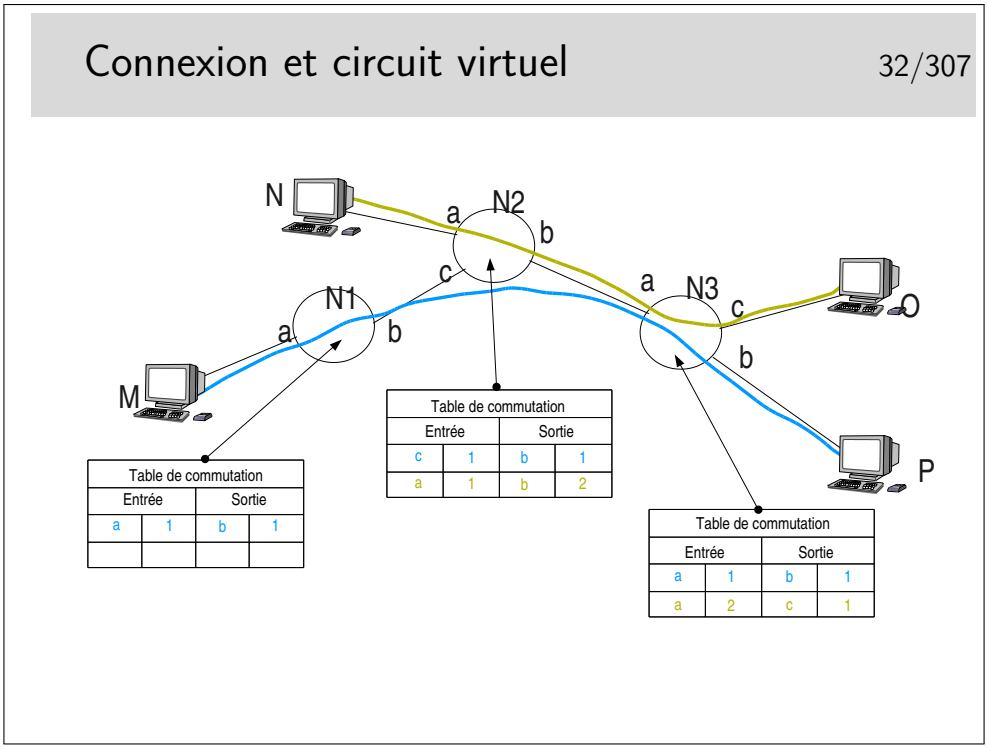
- Mode sans connexion
  - Avantages
    - Pas de nécessité de signalisation pour établir des chemins
    - Reroutage facilité des données en cas de rupture d'un lien dans le réseau
  - Inconvénients
    - Il n'y a pas de chemin préétabli, deux unités de données successives peuvent arriver dans le désordre si le chemin de la seconde a été plus court que le chemin de la première en cas de modification de parcours entre les deux unités
    - On peut envoyer des données vers des destinations inexistantes

Réseaux actuels avec et sans connexion 30/307

- Orientés connexion
  - X25 : le réseau de données des années 80 (né vers 1976)
  - Frame Relay, très utilisé aujourd'hui pour interconnecter des unités dispersées de mêmes entreprises
  - ATM : dans le cœur de réseau des opérateurs (mais aussi dans votre modem ADSL...)
  - MPLS (MultiProtocol Label Switching) : dans le cœur des réseaux d'opérateurs (au service des entreprises)
- Orientés sans connexion
  - Les réseaux locaux d'entreprises : Ethernet, Token-Ring
  - IP : Internet
  - Réseau Cyclades (l'ancètre de IP, abandonné en 1978 par choix politique...)

Caractéristiques des réseaux orientés connexion 31/307

- Les nœuds acheminent les unités de données (les paquets) entre les entrées et les sorties en effectuant des opération de commutation
  - Les nœuds sont des commutateurs
  - Les paquets sont munis d'étiquettes qui les identifient
  - Les étiquettes sont attribuées lors de la phase d'établissement de la connexion. Elles identifient les paquets sur chaque lien.
  - Elles identifient aussi la communication, le canal, le circuit.
  - La série d'étiquette réservées pour une communication sur chaque lien constitue un circuit virtuel



Les paquets issus de M et à destination de P sont munis d'une étiquette 1 en M. Elle reste 1 après passage dans tous les noeuds. Sans doute la communication M-P a-t'elle été établie la première.

Les paquets issus de N à destination de O sont munis d'une étiquette 1. En sortie du premier noeud traversé cette étiquette devient 2 car l'étiquette 1 est déjà attribuée à une communication.

La commutation de circuit virtuel consiste à échanger des étiquettes dans les noeuds et à orienter les unités de données (les paquets) vers les bonnes interfaces de sortie. Le chemin est un chemin d'étiquettes. Celles-ci (les étiquettes) sont réservées lors de l'établissement de la communication.

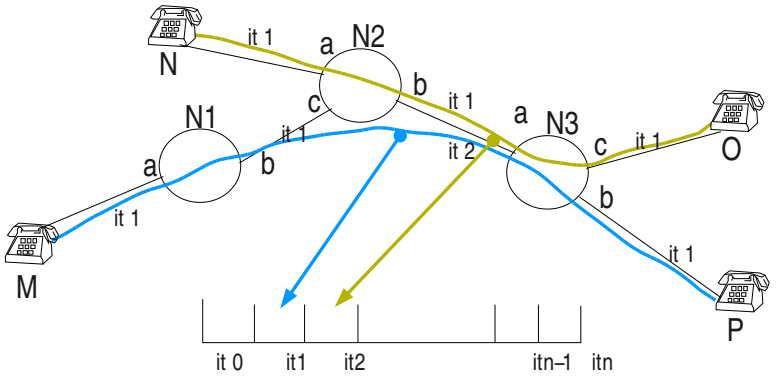
Connexion et circuit réel I

33/307

- Il n'y a pas d'étiquette associée aux données
- les données sont véhiculées dans des canaux spécifiques, réservé lors de la phase d'appel, établis lors de l'établissement de la connexion
- les canaux peuvent être des intervalles de temps, régulièrement espacés (en téléphonie numérique européenne : 32 intervalles de temps tous les 125μs)
- les noeuds du réseaux sont des commutateurs, ils commutent les données en fonction des informations contenues dans leurs tables de commutation
- les tables de commutation donnent la correspondance entre le canal entrant et le canal sortant

Connexion et circuit réel II

34/307



- Entre $N2$ et $N3$, les intervalles de temps 1 et 2 sont attribués aux communications $M-P$ et $N-O$, respectivement, et ce, pour la durée des communications
- Si la communication $M-P$ se termine, la ressource $it1$ est libérée, la communication $N-O$ n'en profite pas
- Si la communication $M-N$ est silencieuse, la bande passante non utilisée est perdue

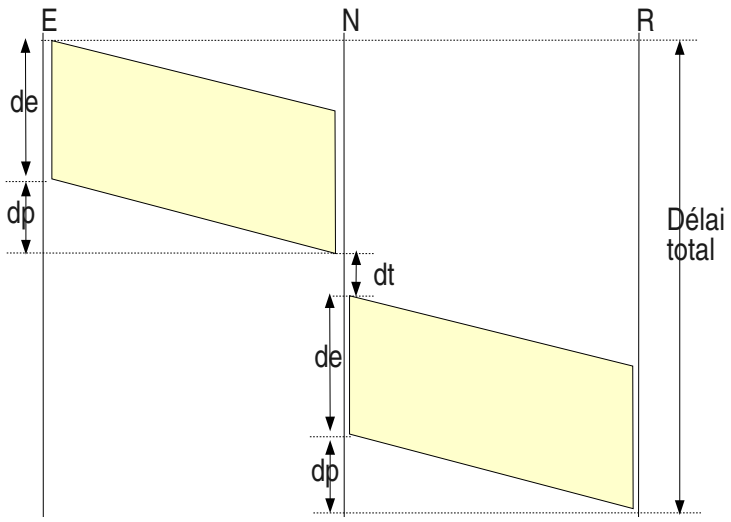
2.3 Délais et QoS

Les délais 36/307

- Entre un émetteur et un récepteur, les données vont subir différents délais
  - Durée d'émission : inversement proportionnelle au débit, proportionnelle à la longueur des segments de donnée
  - Le délai de propagation : au minimum le délai de la lumière dans le vide ($c \approx 300000 Km/s$), dépend par ailleurs du support ($2/3c$ dans le verre et le cuivre).
    Exemple : 250μs pour une liaison par satellite géostationnaire (terre - terre)
  - Le délai de traitement dans les nœuds : délai de traitement réel plus temps passé dans les files d'attente en attente de traitement et en attente de ré-émission

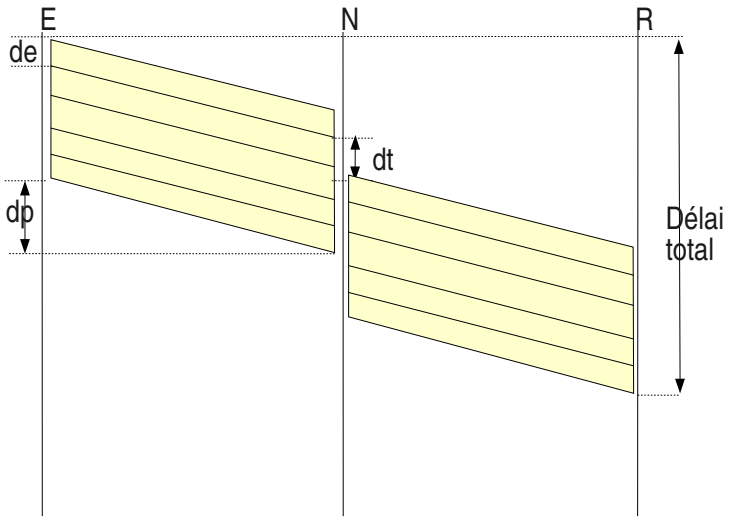
Relation célérité de la lumière dans le vide (c), fréquence (f) et longueur d'onde (λ) : $\lambda f = c$

Les délais 37/307



Délai total $= 2de + 2dp + dt$ si un seul nœud intermédiaire et si les débits sont les mêmes sur les tous les liens.

Réduisons les délais avec de petits paquets 38/307



Les interfaces des nœuds fonctionnent en parallèle. Une interface peut recevoir pendant qu'une autre émet. Il reste cependant le temps de traitement, c'est à dire le temps mis par l'unité centrale du nœud pour déterminer vers quelle interface de sortie il faudra acheminer le paquet, plus le temps passé dans les mémoires files d'attente.

Notion de qualité de service liée aux délais 39/307

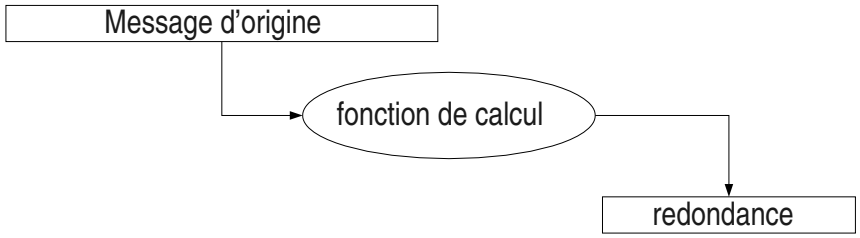
- Pour le téléphone
  - Délai de 50ms : bon confort de communication
  - Au delà de 200ms (communications par satellites geostationnaires) : qualité très moyene, nécessité d'annullation d'écho
  - 400ms : toute dernière extrémité à ne pas dépasser
- Pour les données
  - Dépend fortement du type d'application
- Variation des délais
  - Inhérente aux réseaux de transmission de données
  - Inadapté aux services de type téléphone ou vidéo (mais on tente quand même avec succès, p.ex. : VoIP)

2.4 Détection et correction d'erreur

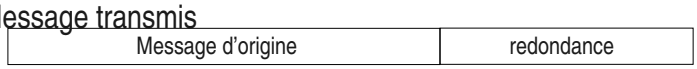
Détection et correction d'erreur 41/307

- Une transmission binaire ne se fait pas sans erreur : des «1» peuvent devenir des «0» et réciproquement
  - Comment détecter des erreurs ?
  - Comment les corriger ?
- Détection (principe général)
  - transmet l'information par bloc plus une séquence de redondance
  - la séquence de redondance est calculée par une fonction spécifique
  - le même calcul est fait à l'arrivée et le résultat est comparé
- Correction
  - directe si la redondance est suffisante et possède des propriétés de correction
  - par retransmission

Détection d'erreur : par séquence de redondance 42/307



Message transmis



À la réception l'opération est reconduite, le résultat est comparé à ce qui est reçu et le message est accepté ou non.

Dans les cas très répandus où les opérations de détection d'erreur sont simples et où les algorithmes ne permettent pas la correction immédiate, on ne peut pas dire si les erreurs de transmission portent sur le message lui même ou la redondance, ou les deux. On jette tout simplement le message (on l'ignore).

On ne se pose pas, à ce stade, le problème de la correction. Ce n'est pas l'affaire de l'al-

gorithme, on s'en remet pour cela à des mécanismes situés dans des couches protocolaires supérieures (si on en a besoin).

Détection d'erreur : la méthode du bit de parité 43/307

- Un bit de parité est rajouté à une séquence de bits
  - le nombre de bits à 1 résultant doit être pair (parité paire) ou impair (parité impaire)
  - Exemples :
    - parité paire

1	0	0	1	0	0	1	1
0	0	0	0	1	1	0	0

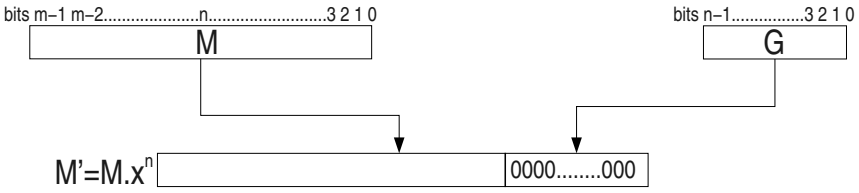
    - parité impaire

1	0	0	1	0	0	1	0
0	0	0	0	1	1	0	1

  - Ne permet pas de détecter des erreurs doubles
  - Méthode utilisée par exemple dans les interfaces séries de nos PC (interfaces de type ANSI RS232-C (ITU-T V24))

Détection d'erreur : la méthode de la division par polynôme I 44/307

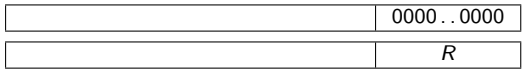
- Un message M est une suite de bits, donc un nombre. On peut l'assimiler à un polynôme. Exemple : la suite de bits 101101 peut être représentée par le polynôme $x^5 + x^3 + x^2 + x^0$
- Le message M a au plus m bits, il est de degré $m - 1$
- Considérons un polynôme G de n bits (degré n-1) avec $n < m$
- On multiplie M par x^n (2^n). Cela revient à «décaler» M de n bits vers la gauche et à ménager ainsi n bits vides à droite



Cette méthode est appelée CRC (*Cyclical Redundancy Code*)

Détection d'erreur : la méthode de la division par polynôme II 45/307

- Le polynôme M' est divisé par G (division modulo 2)
- Alors : $M' = G \times Q + R$ (Q polynôme quotient, R reste)
- Le polynôme $T = M' - R$ est construit
  - le reste R vient se placer dans l'espace droit de M' (la soustraction binaire modulo 2 est équivalent à une addition)



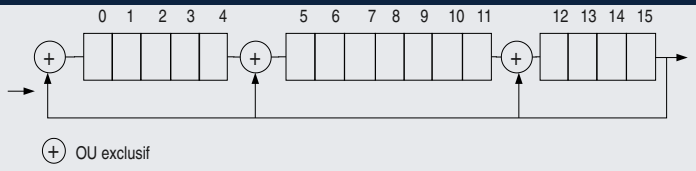
  - $T = M' - R = G \times Q$ donc T est divisible par G
- On transmet T, à la réception on divise le polynôme reçu par G, si on a reçu T, pas d'erreur, alors $R = 0$

Le reste de la division est appelé CRC du nom de la méthode.

Quelques CRC types et le calcul par registres à décalage 46/307

- CRC-12 $= x^{12} + x^{11} + x^3 + x^2 + x + 1$
- CRC-16 $= x^{16} + x^{15} + x^2 + 1$
- CRC-CCITT $= x^{16} + x^{12} + x^5 + 1$
- CRC-32 $= x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

Codage du CRC-16 à l'aide d'un registre à décalage



Facile à implémenter matériellement...

Dans l'histoire des télécomunications des réseaux et de l'informatique, un certain nombre de polynôme de CRC ont été standardisés, offrant oist une *bonne capacité* à détecter les erreurs, soit étant faciles à implémenter (tout est affaire de compromis). Voir
.

Existe aussi en solutions logicielles, mais plus le message est long plus il y a de temps

de calcul. Voir les implémentations en C du CRC-32, par exemple ici :
`http://www.cl.cam.ac.uk/Research/SRG/bluebook/21/crc/crc.html`
Des fonctions existent toutes faites, par exemple en PHP : `int crc32(string str)`
voir : `http://fr2.php.net/crc32`

Détection d'erreur : la méthode de la somme de contrôle 47/307

(ou *Checksum*)

- Soit un message formé par la suite de mots A, B, C, D, ..., R, S, T
  - Les mots sont des ensemble de 8 ou 16 bits ou plus
  - Un mot contiendra la somme de contrôle dans le message, soit S ce mot. Il est mis à 0 pour le calcul
  - Calcul : $Z = A + B + C + D + \ldots + R + 0 + T$
  - Le résultat Z est inversé, on obtient $\bar{Z}$
  - On transmet $ABCD \ldots R\bar{Z}T$
  - À la réception on fait la somme $Z' = A + B + C + D + \ldots + R + \bar{Z} + T$
  - S'il n'y a pas d'erreur alors $Z' = Z + \bar{Z} = 1111 \ldots 1111$
  - le résultat est inversé comme à l'émission et donc $\bar{Z'} = 0$

Rappel (s'il en est besoin) : si un mot Z vaut 1010, alors son inverse (on dit aussi son complément à 1) vaut 0101 et peut être noté $\bar{Z}$ (Z barre).

Détection et correction d'erreur 48/307

- Si la redondance est suffisante et l'algorithme sufisamment puissant il est possible de détecter les erreurs et de les corriger.
  - On peut le montrer sur un exemple simple avec le mécanisme du bit de parité.
    Soit le bloc suivant, chaque ligne et chaque colonne est munie d'un bit de parité paire (dernier bit).

	1	0	1	0	1	0	1	0
	1	0	1	0	0	0	0	1
Cherchez l'erreur :	0	0	0	1	0	1	0	0
	1	1	1	1	0	0	0	0
	1	1	1	0	0	1	1	1

  - Autres techniques : codes de Reed-Solomon, Turbo-codes (origine ENST Bretagne), etc.

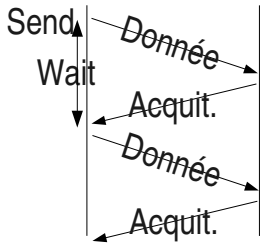
— Reed-Solomon :

`http://en.wikipedia.org/wiki/Reed%E2%80%93Solomon_error_correction`

— Turbo-codes : `http://en.wikipedia.org/wiki/Turbo_codes`

Correction d'erreur par retransmission I 49/307

- Un récepteur détecte une erreur, il jette le segment de données erroné
- Comment l'émetteur peut-il détecter qu'il y a eu une erreur ?
  - Impossible sans l'utilisation d'un mécanisme d'acquittement

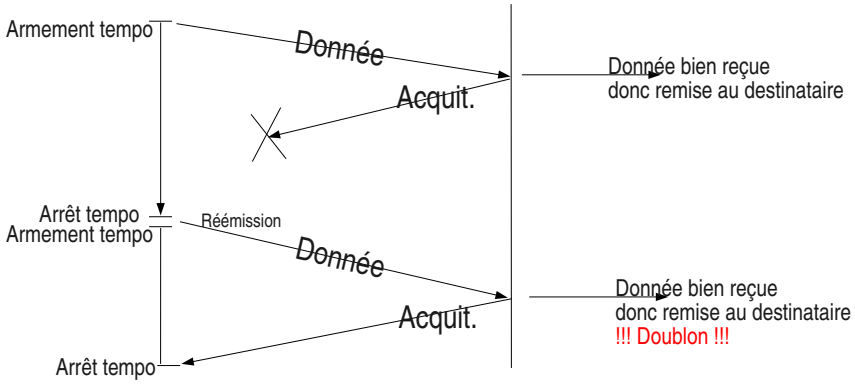


Ce mécanisme est appelé «Send & wait»

Que se passe-t'il si la donnée se perd ?
Si l'acquittement se perd ?

Correction d'erreur par retransmission II 50/307

- Utilisation de temporisateur (timer)

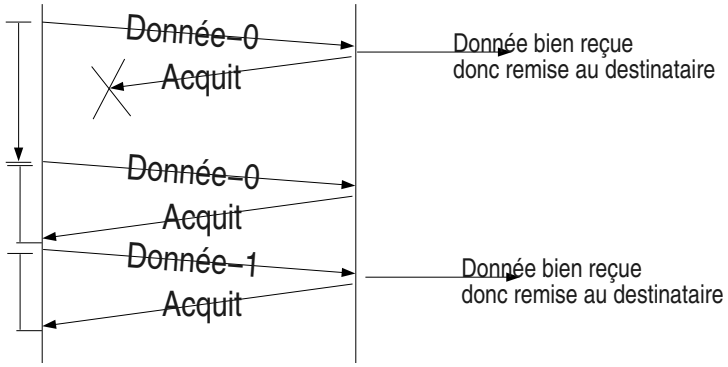


Où comment se créer d'autres problèmes en voulant en régler un... Et maintenant comment on fait pour éviter les doublons ?

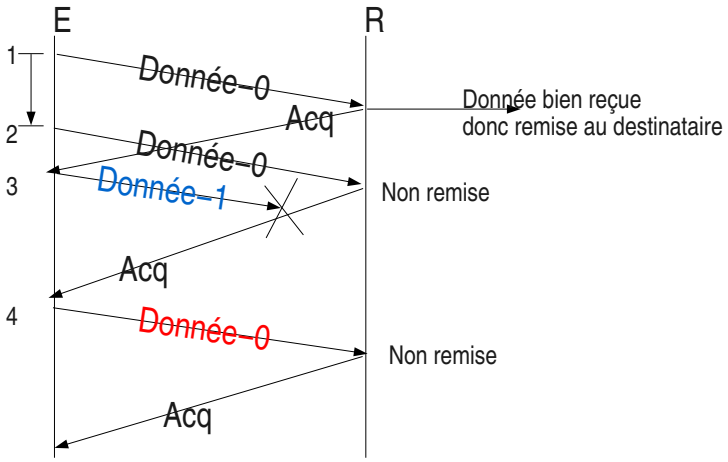
C'est une règle, hélas, très générale, en Réseaux, que de se créer de nouveaux problèmes en apportant des solutions à d'autres...

Retransmission : le problème des doublons 51/307

- On peut éviter les doublons en numérotant les données
  - Exemple : les données sont munies d'un numéro 0 et 1 alternativement



Timer trop court et malchance... 52/307



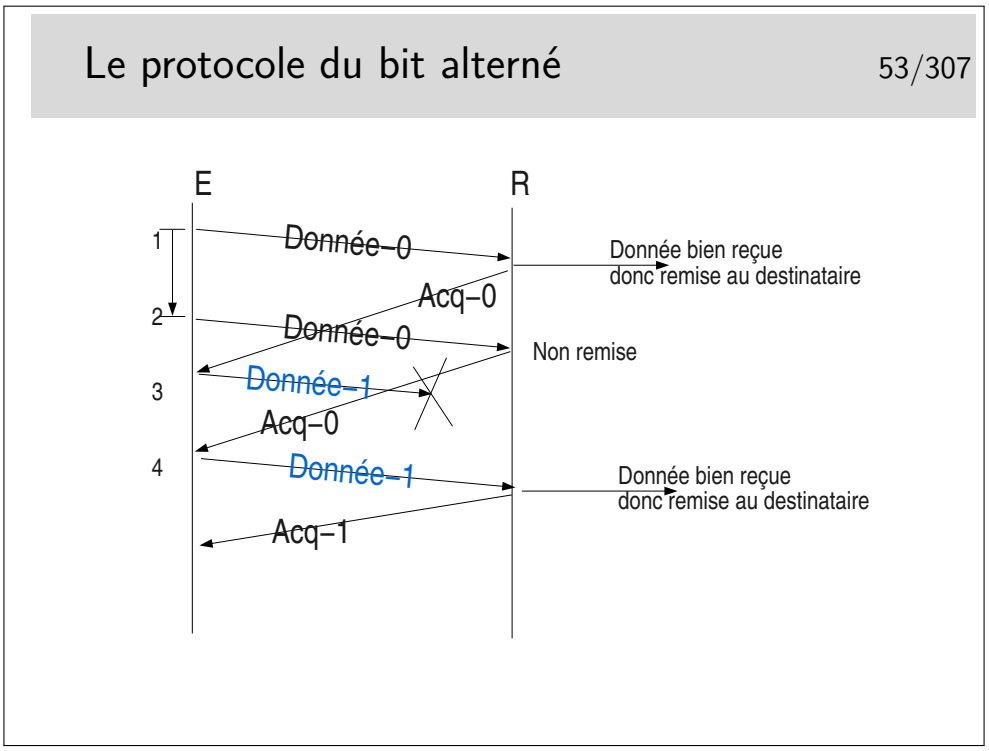
1. émission de `Donnée-0`, bien reçue, elle est remise à l'entité destinatrice
2. le timer expire trop tôt, `Donnée-0` est réémise et est bien reçue, elle n'est pas remise car elle porte le même numéro que la donnée précédente. Cependant, le récepteur renvoie un acquittement car il pense que s'il a reçu à nouveau `Donnée-0` c'est que son acquittement précédent s'est perdu..
3. un acquittement arrive, l'émetteur peut alors émettre une nouvelle donnée : `Donnée-1`

4. un acquittement arrive, l'émetteur pense qu'il s'agit de l'acquittement de `Donnée-1`.

À chaque fois qu'il reçoit un acquittement il pense que celui-ci concerne les données précédemment émises, il vide les tampons mémoires qui les contenaient.

Ainsi, en 4, s'il y a une nouvelle donnée à envoyer ce sera `Donnée-0` (pas le même contenu qu'au début du scénario), or le récepteur voyant arriver à nouveau un numéro 0 le rejettera. Le numéro est censé représenter les données, on ne compare pas celle qu'on reçoit avec les précédentes. D'ailleurs celles-ci ont été livrée à l'entité de destination, elles ne sont pas gardées en mémoire.

Dans ce scénario le segment `Donnée-1` n'est pas reçu mais il est considéré par l'émetteur comme bien reçu. Le segment `Donnée-0` suivant est bien émis et bien reçu mais il n'est pas remis à son destinataire final... Tout va mal dans ce scénario !



Les acquittements portent les numéro des données qu'ils acquittent. Ainsi, en 4, on reçoit à nouveau `Acq-0`, alors qu'on attendait `Acq-1`, il y a un problème, le contenu de `Donnée-1` précédent est toujours en mémoire (on ne le vidra que sur réception de `Acq-1`), on peut donc réémettre `Donnée-1` (le même message que précédemment).

2.5 Des protocoles

Notion de protocole I

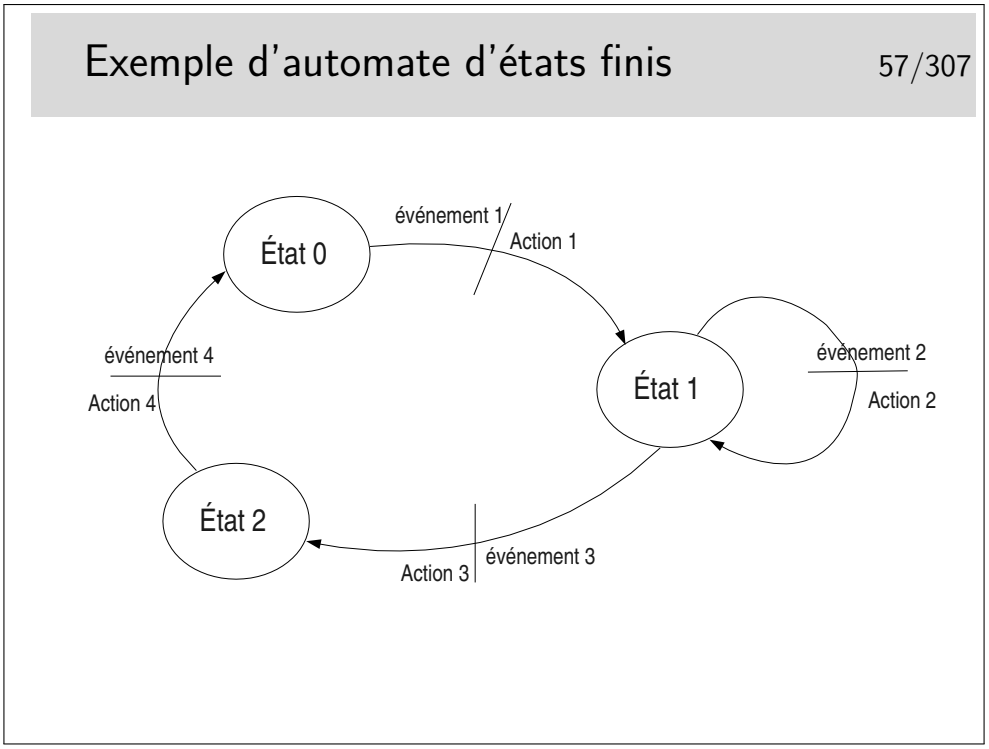
55/307

- Dans ce qui précède nous avons développé des moyens de transmettre des données sans erreurs en rajoutant de l'information et des messages spécifiques
  - À chaque ajout de mécanisme on corrige un problème, on en rajout un autre
  - Il faut que la série correction/nouveau problème converge

Notion de protocole II

56/307

- En rajoutant un protocole au dessus d'une voie de communication on modifie les propriétés de cette voie de telle manière que l'on obtient une nouvelle voie de comunication
- Pour mettre en œuvre et gérer les mécanimes il faut construire un automate logiciel comportant des états et des transitions d'états sur événements



Au départ l'automate est dans un certain état, disons l'état 0 par exemple. Seul l'arrivée de l'événement 1 peut faire que l'automate passe dans l'état 2. Au passage dans ce nouvel état l'action 1 sera effectuée. Un événement ne fait pas toujours changer d'état, on l'illustre ici par l'état 1, dans lequel on reste après que soit survenu l'événement 2 et que l'action 2 ait été effectuée.

Cette représentation est proche de la formalisation mathématique des automates par les «réseaux de Petri».

La représentation schématique est intéressante car elle est souvent plus facile à comprendre (lorsqu'il n'y a pas trop d'états ni d'événements). Cependant elle ne permet pas de s'assurer que tous les cas possibles de relation événement/transition soient envisagés. Il faut alors recourir à la représentation de l'automate sous forme de tableau.

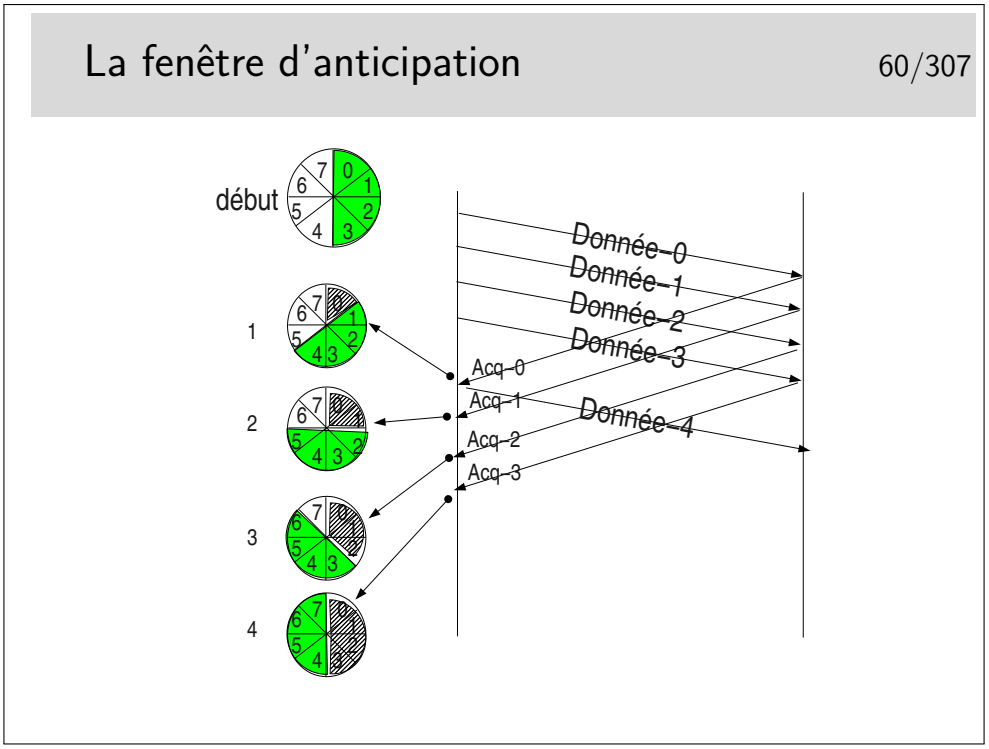
L'automate du protocole du bit alterné 58/307

Événement État	Demande d'envoi de donnée	Arrivée Ack 0	Arrivée Ack 1	Expiration Timer
État 0 nouvel état	Envoyer Data-0 Armer Timer Attente Ack 0			
Attente Ack 0 nouvel état		Arrêt timer État 1		Envoyer Data-0 Armer Timer Même état
État 1 nouvel état	Envoyer Data-1 Armer Timer Attente Ack 1			
Attente Ack 1 nouvel état			Arrêt timer Etat 0	Envoyer Data-1 Armer Timer Même état

Les protocoles à anticipation 59/307

- le mécanisme du *Send & Wait* conduit à une mauvaise utilisation de la bande passante, quand on attend on n'utilise pas la bande passante alors disponible
- Il est judicieux d'anticiper l'émission sans attendre les acquittements
  - Dans une certaine mesure...
  - Une limite, appelée fenêtre d'anticipation, permet de ne pas trop anticiper et bloque l'émission si les acquittements n'arrivent pas
  - Les acquittements existent toujours
  - La fenêtre «tourne» ou «glisse» lorsqu'un acquittement arrive

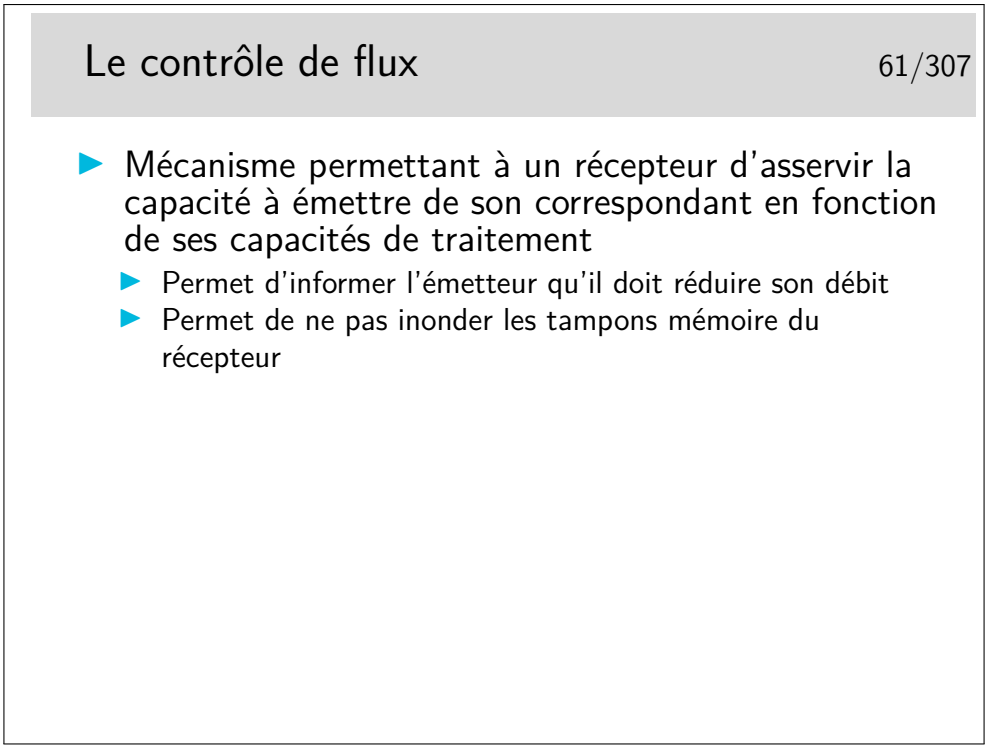
On dit que la fenêtre est «tournante» ou «glissante» selon la représentation qu'on en fait. Voir le transparent suivant.



Paramètres : les segments de données sont numérotés modulo 8. La fenêtre d'anticipation est de 4.

On ne peut pas émettre plus de 4 segments à la suite si on ne reçoit pas pendant ce temps un acquittement. Lorsqu'un acquittement arrive, la fenêtre «tourne» d'un pas, le tampon mémoire contenant la donnée correspondant à l'acquittement est libéré, la donnée est considérée comme bien envoyée, on l'efface. En pratique on ne l'efface pas, on considère son emplacement mémoire comme libre. Sur le schéma ci-dessus, on indique cette «libération mémoire» par une zone grisée dans la fenêtre.

Départ : la fenêtre englobe les numéros 0 à 4. On commence l'envoi. L'acquittement pour la donnée 0 arrive en 1. La fenêtre tourne et englobe 1 à 4. Le segment 4 devient éligible à l'émission. Il est émis si une donnée est à émettre. Le tampon mémoire contenant Donnée-0 est effacé. En 2 l'acquittement pour Donnée-1 arrive, la fenêtre tourne et englobe maintenant 2 à 5. Etc.



Un émetteur et un récepteurs ne fonctionnent pas obligatoirement à la même vitesse (au même débit). Les tampons mémoire de réception se vident lorsque les applications destinatrices viennent y puiser les données reçues. Si la machine de réception est lente, si l'application réceptrice prend trop de temps à traiter les données et ne vient pas les retirer suffisamment rapidement des tampons de réception, ceux-ci se remplissent dangereusement. Lorsque les tampons sont pleins les données à recevoir seront perdues. Le contrôle de flux permet d'éviter ces pertes en évitant que les données ne soient envoyées.

Le contrôle de flux pourra être couplé à des mécanismes d'acquittement, ce sont les trames RR et RNR du protocole HDLC-LAPB que nous verrons plus loin : RR pour *Receiver Ready* ou encore «tout va bien, envoyez !», RNR pour *Receiver Not Ready* ou encore «OK, j'ai bien reçu vos données mais arrêtez vous quelque temps».

3 Modélisation et standardiasation

3.1 Standardisation

Les organismes de standardisation 64/307

- Les organismes officiels nationaux et internationaux
  - OSI : Organisation de Standardisation Internationale (ISO en anglais), et ses branches nationales : AFNOR en France, DIN en Allemagne, ANSI aux USA
  - UIT-T : Union Internationale des Télécommunications, secteur des Télécommunications (il y a aussi le secteur Radio)
  - ETSI : European Telecommunications Standards Institute
- Les organismes de l'industrie et de la recherche
  - IEEE : Institut of Electrical and Electronics Engineers
- L'Internet
  - IETF : Internet Engineering Task Force, étudie et développe les protocoles et services au dessus du protocole IP

3.2 Principe de la modélisation

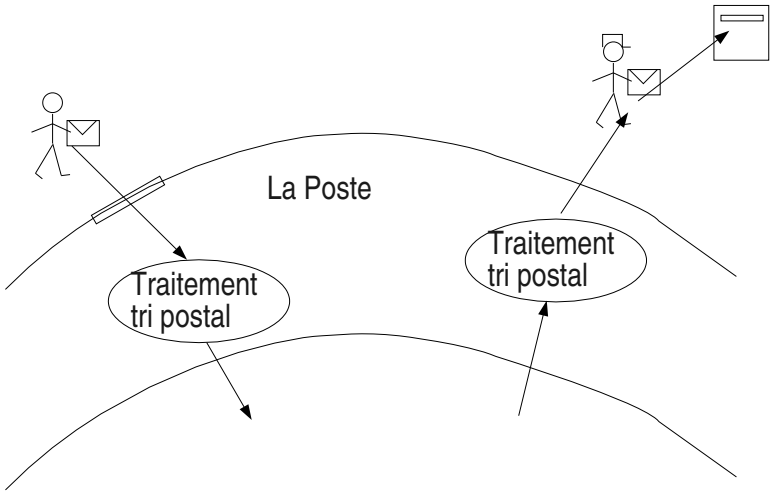
La modélisation de la problématique réseau 66/307

- Comment «voir» le réseau ?
  - Comme un utilisateur : «*Je me connecte, je ne sais pas comment ça marche, je ne veux pas savoir comment ça marche, mais ça marche ! Et j'utilise.*»
  - Comme un développeur d'application communicantes : «*Par quel moyen programmatique mes applications peuvent-t'elles communiquer ? Dois-je savoir comment fonctionne le réseau ? Tout le réseau ? Une partie du réseau ?*»
  - Comme le gestionnaire du réseau : «*Dois envisager le réseau dans son ensemble, du câble aux applications ?*»

Exemple d'un réseau existant : La poste 67/307

- En tant qu'utilisateur :
  - *Je dois savoir où est situé le bureau de poste, l'adresse de mon correspondant et avoir une boite aux lettres. Je dois pouvoir interagir avec le service offert par La Poste. La poste doit me fournir un service et des moyens d'accès à ce service.*
  - *Je n'ai pas à savoir comment fonctionne le service de La Poste de l'intérieur*
- En tant que postier :
  - *Je dois savoir traiter les lettres, les orienter vers les bons sacs postaux, placer les sacs postaux dans les bons camions, voitures trains, avions...*
  - *Je n'ai pas à savoir comment fonctionne le service de transport qui achemine physiquement les sacs postaux.*

Services et interfaces d'utilisation 68/307



La Poste offre un *SERVICE*

On accède à ce service par des moyens appelés, en terme «réseaux», des *PRIMITIVES DE SERVICE*.

On interagit avec le service via les primitives de service dont le paramètre principal est une sorte d'adresse : l'adresse où est situé le bâtiment de La Poste pour aller «poster» sa lettre, l'adresse du destinataire de la lettre pour que le postier sache dans quelle boite aux lettres déposer celle-ci. Cette sorte d'adresse est appelée en termes réseaux le *POINT*

D'ACCÈS AU SERVICE (le Service Access Point ou SAP).
Notion de couche, d'interface entre couches et d'indépendance entre couche.

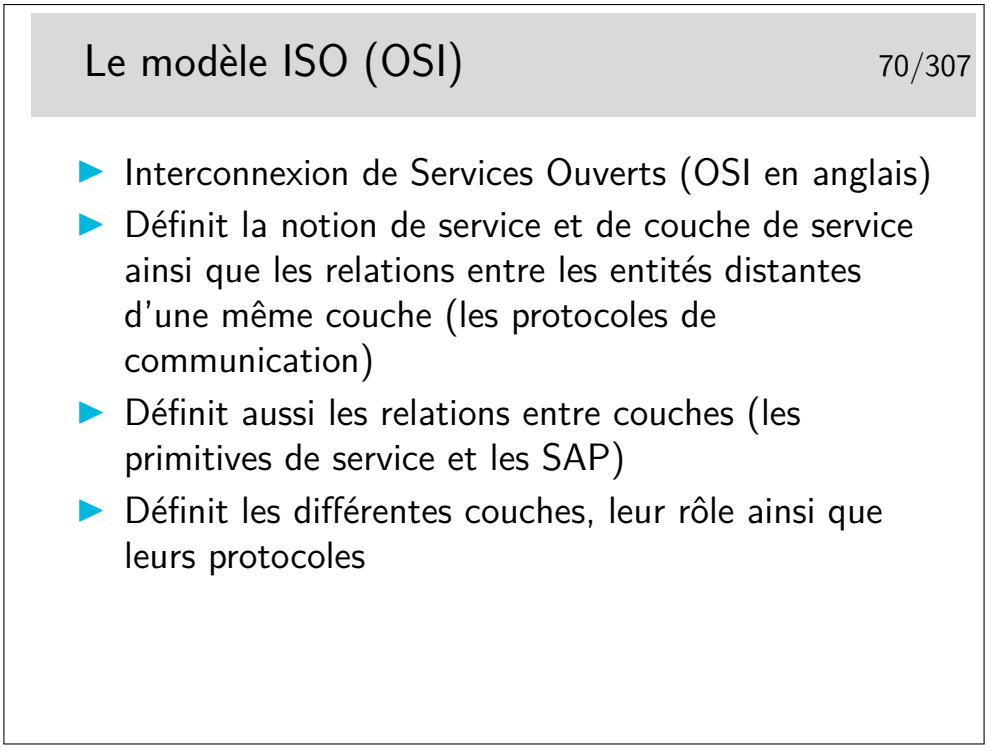
Question : le travail d'acheminement de la lettre est-il terminé lorsque le postier distributeur (le facteur) a déposé la lettre dans la boite aux lettres du destinataire ?

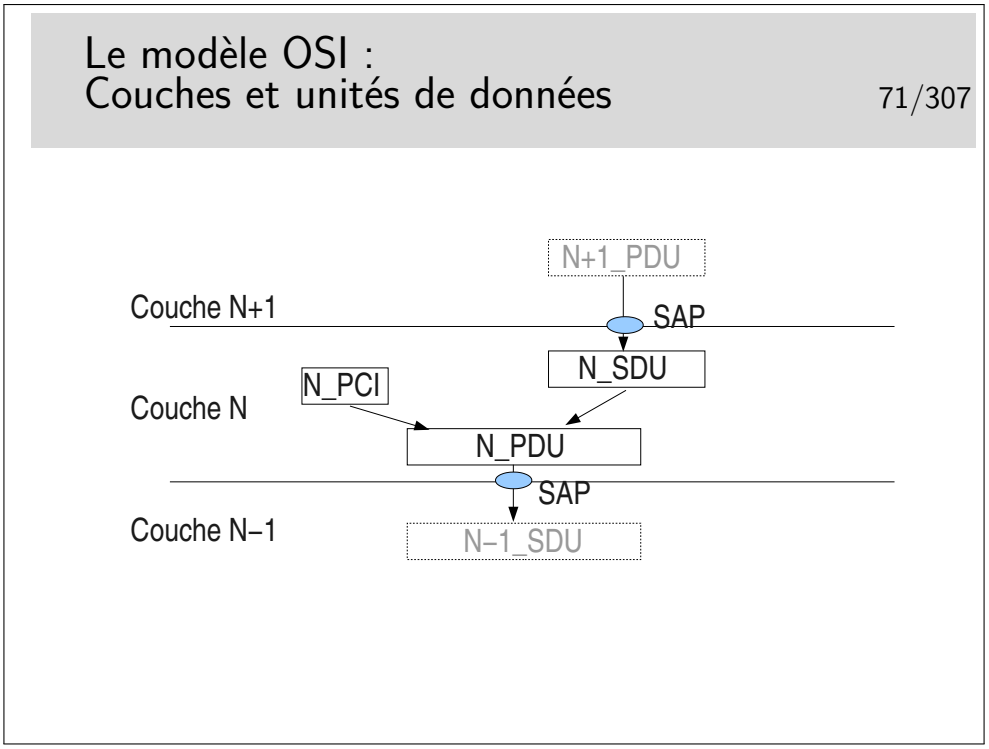
Réponse : oui et non !!!

— Oui pour le service postal.

— Non pour le destinataire final. Il peut s'agir de la boite aux lettres de la famille Dupont. La lettre peut être pour Philippe Dupont et non pour Jacques Dupont. Le SAP «adresse des Dupont» n'est pas le seul SAP à considérer. Le SAP «Jacques» et le SAP «Phillipe» sont aussi à prendre en compte, mais pas au niveau du service postal (on dira plus tard : «pas dans la même couche»).

3.3 Le modèle ISO (OSI)





L'unité de donnée fournie par l'utilisateur est une unité de donnée à SERVIR. C'est une UNITE de DONNEE de SERVICE : une *SERVICE DATA UNIT* (SDU) en anglais.

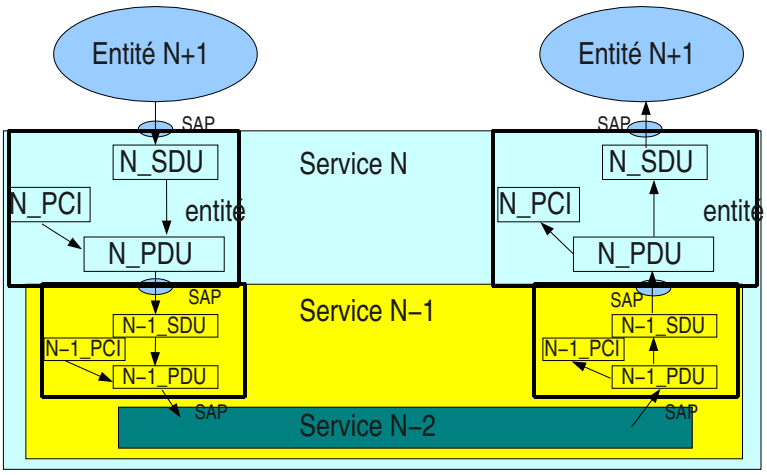
La couche assurant le service utilise un certain mécanisme qui lui est propre, un *protocole* particulier, nécessitant un échange de données spécifiques. Ces données protocolaires n'ont rien à voir avec les données utiles, elles servent à la gestion du transfert de celles-ci.

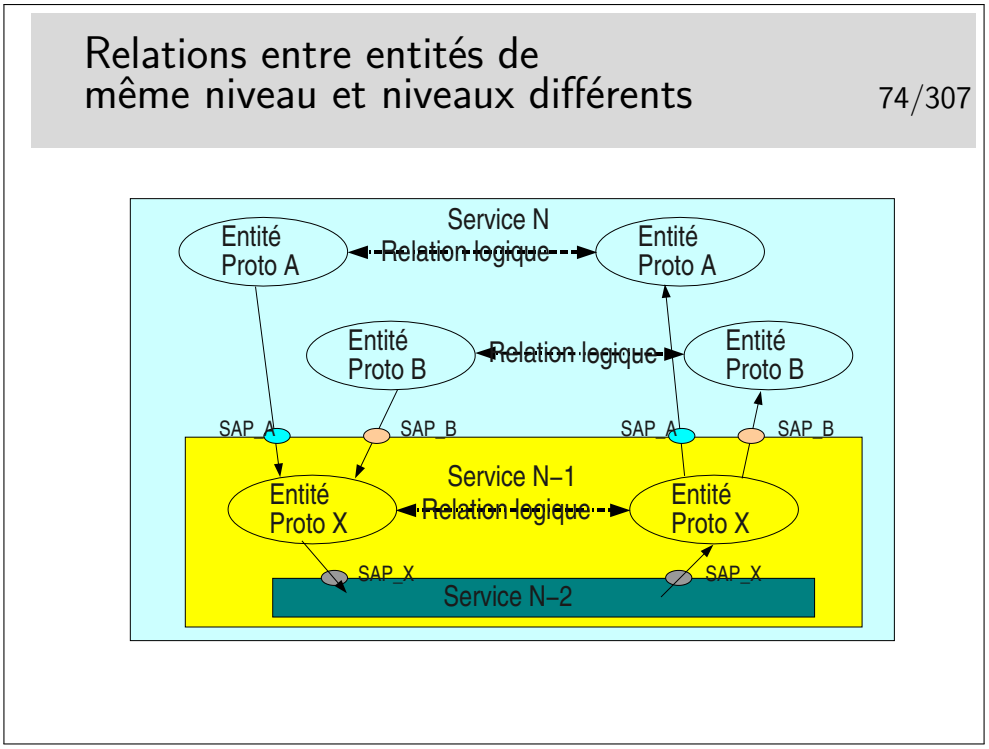
Les données protocolaires (Protocol Control Information) sont ajoutées au segment de données utiles (les données de service, la SDU) pour former une nouvelle unité : l'unité de données de protocole ou *PROTOCOL DATA UNIT* (PDU).

Notion d'entité protocolaire 72/307

- Une entité protocolaire est un «programme informatique», une application ou un module opérationnel du système d'exploitation mettant en œuvre un protocole
- Une même couche peut être composée de plusieurs entités mettant en œuvre des protocoles différents pour assurer un même service
  - Exemples :
    - Le téléchargement de fichier peut être réalisé via le protocole ftp ou le protocole du web http
    - Des machines peuvent partager des fichiers via les protocoles NFS (monde Unix), SMB (monde Windows) ou AppleTalk (monde MacIntosh)

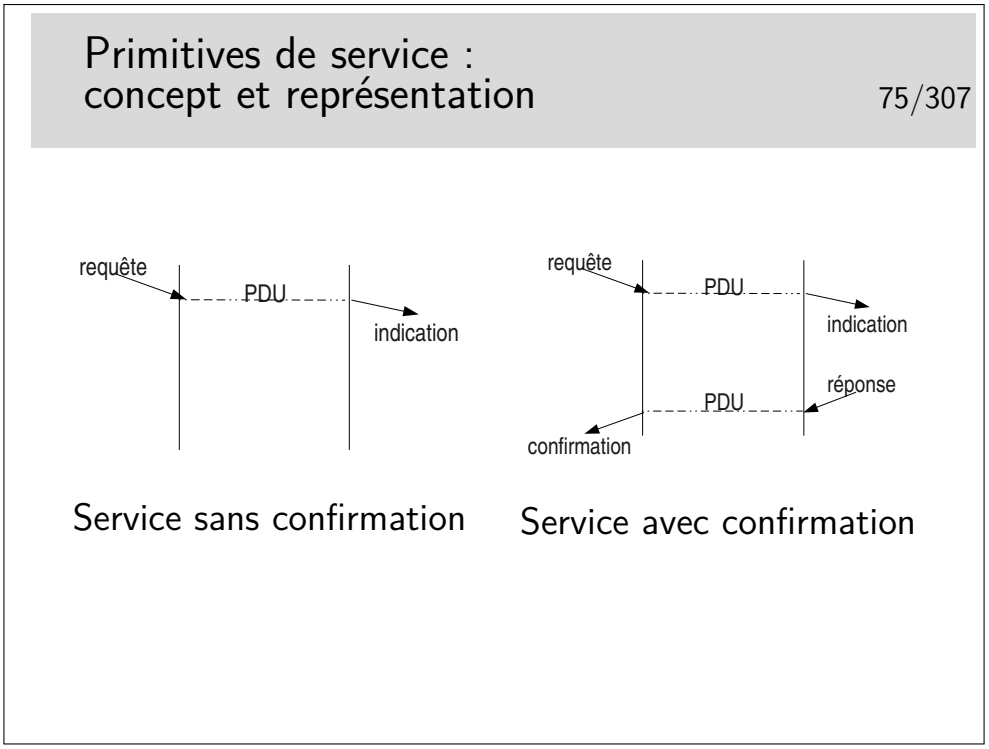
Les services et les couches 73/307





La figure ci-dessus est un exemple de ce qui est possible. Un service N met en œuvre deux types de protocoles différents pour assurer un service. Les entités protocolaires A peuvent communiquer entre elles. De même pour les entités protocolaires B. Les entités A ne peuvent communiquer avec les entités B car elles ne «parlent pas» le même langage (le même protocole).

Il est possible que les entités A et les entités B utilisent le même protocole X sous jacent (de couche $N-1$). Lorsque l'entité A de gauche émettra un PDU vers l'entité A de droite, il faudra que ce PDU soit muni de l'identité du SAP entre l'entité A (niveau N) et l'entité X de niveau $N-1$ (donc SAP A). Sinon l'entité X de droite ne saurait pas vers quelle entité réceptrice A ou B envoyer le PDU. De même pour les entités B.



Le concept de primitive de service peut s'apparenter au concept de fonction de programme informatique ou mieux encore d'objet au sens «langage orienté objet» tel C++ ou Java.

Mais nous sommes face à un problème car si les concepts sont voisins, ils n'en ont pas moins des représentations différentes. Les primitives définies pour une couche protocolaire sont indépendantes de tout langage informatique. Fort heureusement d'un coté. Sous un autre angle de vue cela pose problème. En effet, il faut pouvoir, à la fin, implémenter les protocole dans un langage et sur un système d'exploitation. Comment traduire en C/C++ sous Unix/Linux, Windows ou MacOS le paradigme «Requête/indication» présenté ci-dessus ? Il faut noter en outre que la requête peut être codée en C et l'indication en Java.

Dans la pratique, il y a une réelle différence entre les outils à notre disposition pour développer des applications et les définitions formelles des primitives. C'est vrai pour les applications standards, c'est encore plus différent pour les entités protocolaires développées dans les noyaux des systèmes d'exploitation.

Lorsqu'on étudie les documents de spécification d'un protocole on est toutefois confronté au concept de primitive. Il faut alors absolument savoir ce que cela représente, en faisant abstraction, en première lecture, de toute idée d'implémentation. Par la suite peut se poser le problème de l'implémentation.

Les 7 couches du modèle OSI
La couche 1 (*Physical Layer*)

1 - La couche physique

- Définit les moyens pour transformer les bits constituant les données en information analogique transportable sur les liens physiques entre la machine et son nœud de raccordement au réseau
- Définit les caractéristiques matérielles et électriques (ou optiques) des supports physique

Les 7 couches du modèle OSI
La couche 2 (*Data Link layer*)

2 - La couche liaison

- Définit les moyens d'acheminer des données structurées au dessus du niveau physique, entre la machine et son nœud de raccordement. La structure de base est la trame
- Définit les moyens de contrôler la fiabilité de la trame en réception
- Peut définir des mécanismes de contrôle de flux et de récupération d'erreurs

Niveau physique et niveau liaison : on ne traverse pas le réseau.

Les 7 couches du modèle OSI
La couche 3 (*Network Layer*)

78/307

3 - La couche Réseau

- Définit les moyens pour acheminer l'information entre machines d'extrémités en traversant les nœuds du réseau
- Implique une nécessité d'identification des machines terminales : un adressage
- Implique de mettre en œuvre des mécanismes de routage dans les nœuds
- Les unités de données de ce niveau sont appelées des paquets

Rappel de l'épisode précédent (niveau physique et niveau liaison) : on ne traverse pas le réseau !

Épisode présent (niveau 3) : on traverse le Réseau !

Les 7 couches du modèle OSI
La couche 4 (*Transport Layer*)

79/307

4 - La couche transport

- Dernière des couches basses
- Interface entre les applications et la couche réseau
- Fournit une abstraction du réseau
  - Corrige les imperfections de la couche réseau
    Dernière chance pour que les applications soient assurées du bon transfert de leurs données
  - Le modèle OSI fournit 5 classes de fonctionnalités différentes pour corriger les imperfections du réseau. De la classe 0 pour le très bon réseau à la classe 4 pour le mauvais réseau.

Tout ceci est très théorique. En pratique, il faut considérer ce niveau transport par type d'architecture : IP, Novell IPX, AppleTalk, IBM SNA, Decnet de Digital (racheté par Compaq, racheté par HP).

Les 7 couches du modèle OSI
Les couche 5 (*Session Layer*)

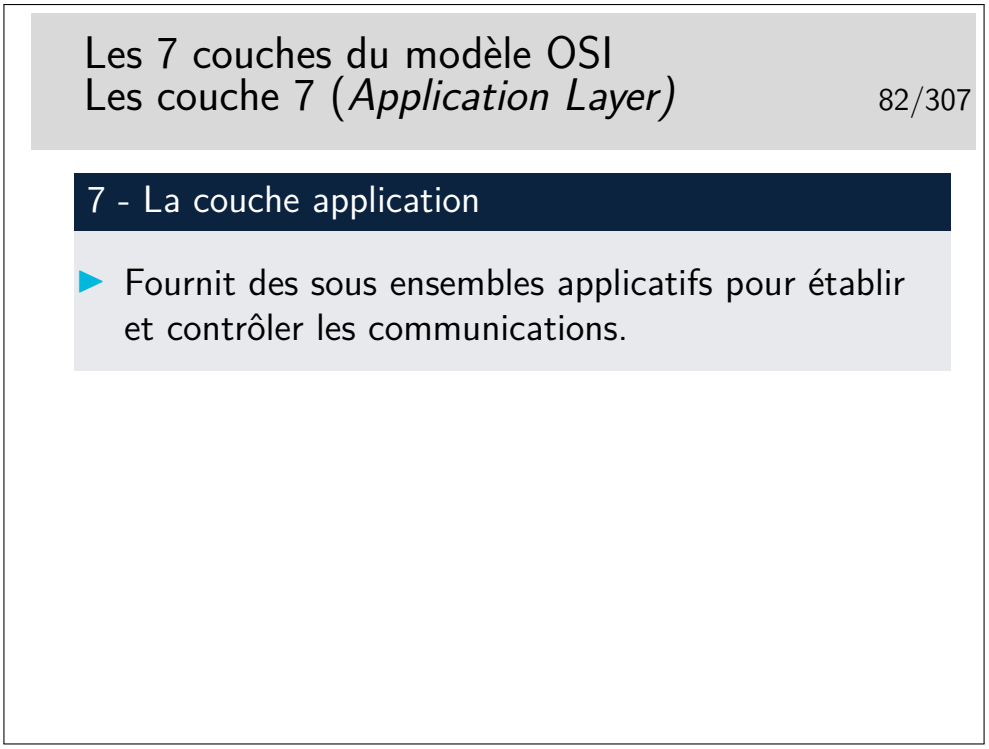
5 - La couche session

- Une communication entre deux applications est considérée comme une session
- La session est organisée, contrôlée
  - Il est prévu des points de synchronisation du dialogue

Les 7 couches du modèle OSI
Les couche 6 (*Presentation Layer)*

6 - La couche Présentation

- Fournit les moyens de décrire les types d'information échangés entre les application : langage de description de types : ASN.1
- Fournit les moyens de coder ces types de données dans un format indépendant de celui des machines (problème de la représentation *big* ou *little endian*)



Tout ceci est à nouveau très théorique. Il faut le replacer dans le contexte de chaque architecture (IP, IPX, etc.)

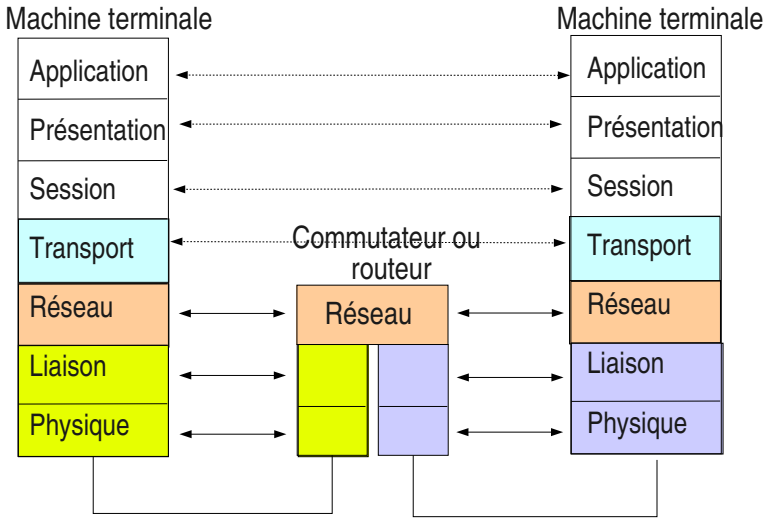
La couche application, qu'on «résume», ici, d'une seule phrase est la plus complexe de toutes dans la réalité des recommandations ISO et ITU-T. Même si ses mérites sont grands, sa complexité et le peu d'outils de développements (API) ont fait qu'il n'existe pas beaucoup de réalisations pratiques (courrier électronique X400, annuaire X500 et quelques autres). Ces applications ont vu le jour vers la fin des années 80, à une époque où l'Internet se répandait déjà beaucoup dans les réseaux du monde de l'enseignement et de la recherche. Le début des années 90 a été décisif, après ne courte période d'incertitude, l'ouverture des standards autour de IP, face à au monde OSI plus fermé à fait pencher la balance en faveur de IP.

Aujourd'hui, il reste cependant le modèle, incontournable, tout au moins pour les couches bases. Il nous aide à placer les concepts et les fonctionnalités des différents réseaux qui existent.

Les 7 couches du modèle OSI : pour résumer... 83/307

- Couches physique (1) et liaison (2) : de la machine au réseau
- Couche réseau (3) : de la machine à la machine en traversant le réseau
- Couche transport (4) : une abstraction du réseau (couches basses) pour l'applicatif
- Couches hautes (5 6 7) : des services, géré au niveau applicatif ou middleware

Où sont implémentées les couches ? 84/307



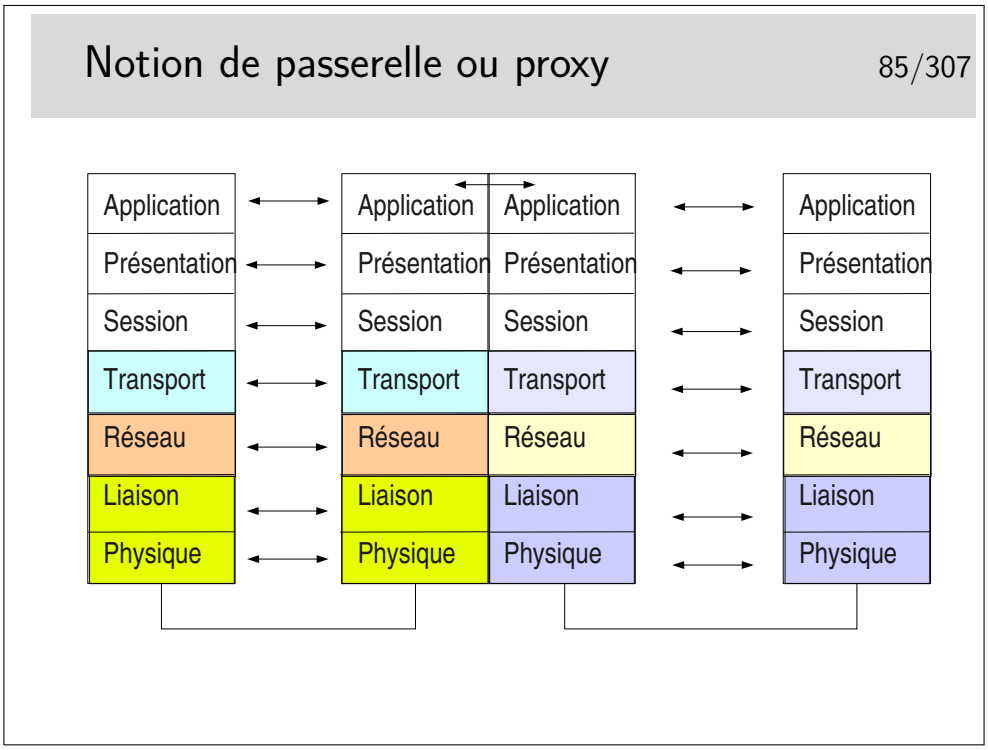
Toutes les couches sont implémentées dans les machines terminales.

Seules les couches 1 à 3 sont nécessaires dans les machines du cœur de réseau.

Il est très important de noter l'indépendance entre les couches. Voyez ici les couches physique et liaison concernant le lien coté gauche et le lien coté droit. Ces niveaux peuvent être totalement différents du point de vue physique mais aussi du point de vue couche liaison. On pourrait avoir, coté gauche un lien Ethernet sur paire torsadée et de l'autre un lien sur fibre optique employant la technologie Sonet/SDH.

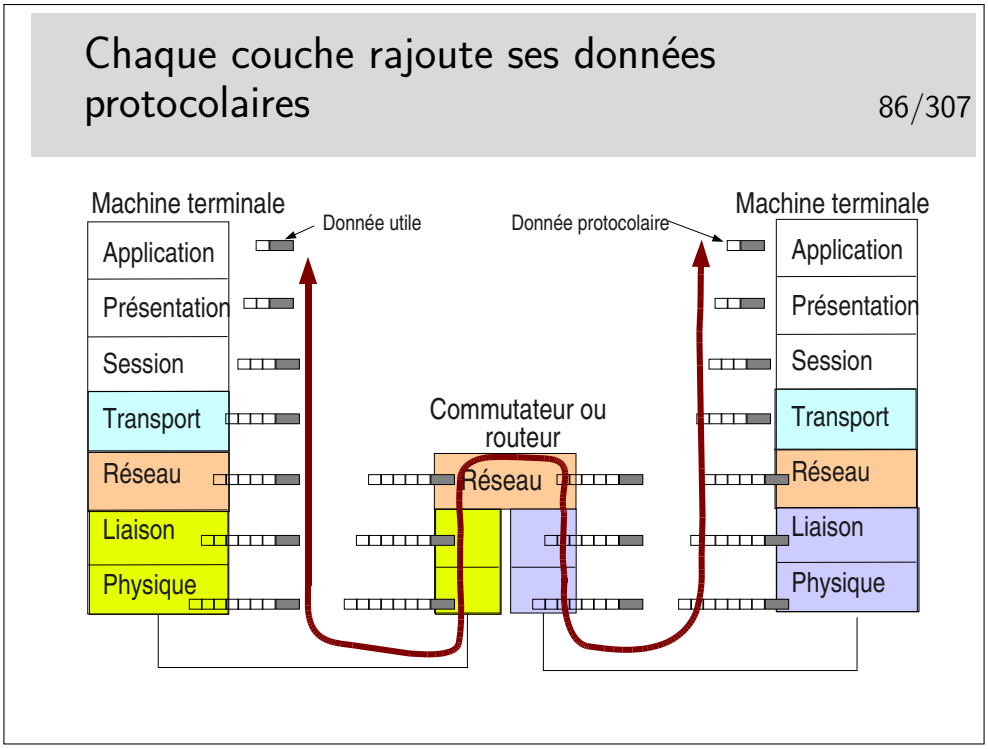
Les débits peuvent aussi être différents. On pourrait avoir 1GB/s coté gauche et 155Mb/s coté droit.

Les couches Physique et liaison sont en général implémentées sur un même support matériel, une carte d'interface du type de celle que vous pouvez avoir sur votre ordinateur personnel



Toutes les couches sont implémentées dans la machine intermédiaires. La couche application réalise une traduction des données pour les adapter à l'application destinatrice. Les protocoles mis en oeuvre à gauche peuvent être tous différents de ceux de droite.

Dans le cas d'un proxy de type Web, les protocoles sont les mêmes à gauche et à droite, mais on oblige toutes les requêtes à passer par l'application intermédiaire (on imagine par exemple un client Web à gauche et un serveur à droite). Cela permet de réaliser des caches permettant de mémoriser les documents déjà téléchargés, afin de les renvoyer plus rapidement sans encombrer le réseau à nouveau.



Notion de rendement : rapport entre le volume des données utiles et le volume total de données véhiculées sur un niveau.

Par exemple, pour une application cliente telnet, sur TCP IP sur Ethernet (nous verrons ces protocoles plus loin dans le cours) il est courant d'envoyer les données octet par octet dans le sens client vers serveur (au rythme où sont frappées les touches du clavier). Pour chaque octet envoyé, telnet ne rajoute rien mais TCP (niveau 4) rajoute au moins 20 octets (plus 12 avec les options des implémentations TCP d'aujourd'hui), IP (niveau 3) rajoute 20 octets. On a donc au moins $1 + 20 + 20 = 41$ octets et au plus $1 + 20 + 12 + 20 = 53$ octets. Dans le cas 41 octets, Ethernet rajoute 5 octets de bourrage pour arriver à la taille minimale requise, donc 46 octets. Par ailleurs Ethernet rajoute 18 octets d'informations protocolaires donc $46 + 18 = 64$ octets dans un cas et $53 + 18 = 71$ octets dans l'autre.

Le rendement est de 1/64 dans un cas et 1/71 dans l'autre. Ce n'est pas très efficace mais c'est ainsi !

4 Les couches basses

4.1 Problèmes de la couche physique

Problématiques 89/307

- Fonction première : transport des bits sur un support analogique
  - Modulation de signal sinusoïdal (fréquence porteuse) en amplitude et en phase par modems
  - Codage en tension : le plus simple mais limité en distance et en débit. Fragile face aux perturbations électromagnétiques
  - Codage en courant : 2 fils par signal, plus résistant aux perturbations électromagnétiques. Permet de monter très haut en débit (100Mb/s et plus)
  - Codage photonique : sur fibre optique, des impulsions de lumière
- Problème connexe : synchronisation parfaite des horloges des récepteurs sur les horloges des émetteurs

Synchronisation des récepteurs I 90/307

- L'idéal :
  - l'horloge du récepteur doit fonctionner à la même fréquence que celle de l'émeteur
  - l'horloge du récepteur doit fonctionner avec un décalage de phase constant avec celle de l'émetteur (décalage de π pour que le front montant coté recepteur ait lieu en milieu de bit reçu)

Synchronisation des récepteurs II 91/307

- La réalité physique
  - les conditions énoncées ci-dessus sont impossibles à réaliser
  - le récepteur fonctionne avec une horloge qui lui est propre. Il est impossible physiquement de fournir une horloge unique pour tous les émetteurs et récepteurs du réseau. La fréquence de l'horloge récepteur sera naturellement légèrement différente de celle de l'émetteur et en plus la différence variera en fonction du temps et des conditions physiques (température du milieu, perturbations électromagnétiques du milieu).
  - même si les horloges étaient parfaitement à la même fréquence, le problème de la phase demeurerait

Synchronisation des récepteurs III 92/307

- Le principe fondamental pour remédier au problème :
  - La suite des bits de données est transcodée pour permettre de véhiculer simultanément un signal en phase avec l'horloge d'émission.
  - L'électronique de réception est capable de déduire l'horloge émission du flux binaire reçu. Elle génère alors un signal qui vient synchroniser parfaitement l'horloge du récepteur.

Les mécanismes de codage 93/307

- Forts divers. Ils doivent permettre de respecter les règles suivantes :
  - pas de longues suites de 1 ou de 0, sinon on perd la synchronisation,
  - pas de composante continue dans le flux émis (la décompositon spectrale du signal fait apparaître un courant ou une tension à la fréquence 0 : courant continu). Ce point est finalement identique au précédent, des suites de 0 ou des suites de 1 sont analogues à des séquences de signal continu.

Quelques types de codage I 94/307

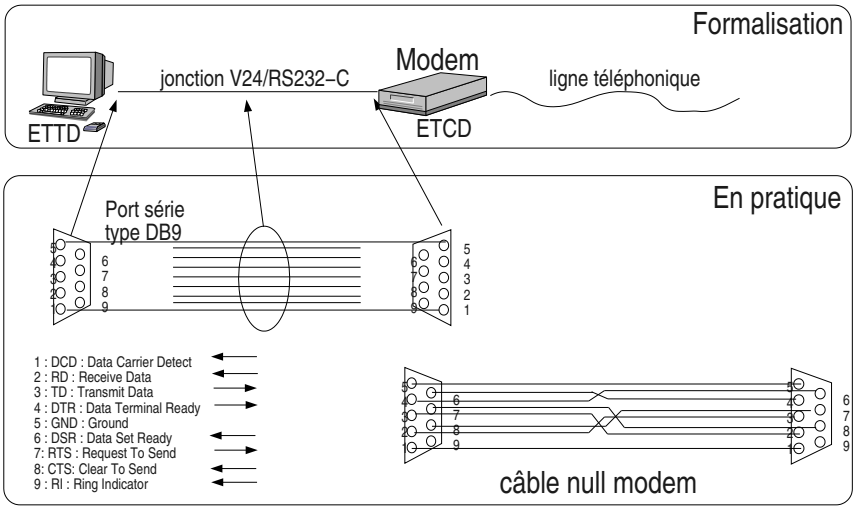
- HDB3 : (haute densité binaire d'ordre 3), les UNs sont codés alternativement +V et -V. Les ZEROs sont codés 0. Si une suite de 3 ZEROs arrive, un UN est inséré à leur suite. C'est un faux UN, pour le reconnaitre on lui donne la même polarité que le UN normal qui a précédé (viol de bi-polarité). Multiplex téléphoniques à 32 voies (2,048 Mb/s).
- AMI : Alternate Mark Inversion, même principe sauf que le bit de poids fort est toujours à 1 et qu'il n'y a pas de longue suite de 0 à priori car on porte de la voix en PCM
- bipolaire (manchester) : chaque bit est codé par une transition +V-V ou -V+V suivant la valeur du bit. Il ne faut pas croiser les fils car les bits seraient décodés inversés (les UN seraient vus commes des ZEROs et réciproquement). Réseau Ethernet.

Quelques types de codage II 95/307

- bipolaire différentiel (manchester diférentiel) : même principe que ci-dessus mais le codage d'un bit (une transition) dépend de la valeur du bit précédent. Le problème précédent n'existe alors plus. Réseau Token Ring.
- 4B/5B : on transmet 5 bits pour 4 bits utiles. À toute configuration de 4 bits de données on fait correspondre un mot de 5 bits (16 combinaisons, 32 mots possibles). Certains mots de 5 bits sont interdits de par leur configuration. Certains mots sont réservés pour un usage particulier de signalisation sur la liaison
- 8B/10B : principe analogue au précédent

4.2 Exemples de couches physique

Jonction ITU-T V24 – ANSI RS232-C Exemple de couche physique 97/307



ETTD : Equipement Terminal de Circuit de Données (en anglais DTE : Data terminal Equipment). Désigne le coté terminal de la jonction, en pratique, très souvent, l'ordinateur.

ETCD : Equipement Terminal de Circuit de Données (en anglais DCE : Data Circuit

Equipment). Désigne le coté Réseau de la jonction. Souvent en pratique il s'agit du modem. Mais cela peut être aussi une interface série d'une machine.

Ce sont de vieux termes, encore utilisés dans le cas des liaisons de type série, à bas débit.

En pratique votre ordinateur est relié à un modem standard (non ADSL), via le port COM1 ou COM2 (appelés encore «ports série»), par une jonction conforme à la recommandation V24/RS232-C.

Le câble *NULL Modem* permet de raccorder deux ordinateurs entre eux via leur port série. On dit encore de ce câble qu'il est «croisé» car, en particulier, la broche 2 de l'un est reliée à la broche 3 de l'autre et réciproquement.

Voir : `http://www.loop-back.com/null-mod.html`

Dans ce type de jonction on trouve les supports pour la transmission de l'information utile mais aussi des supports pour de l'information de service («présence porteuse», «terminal prêt», etc.). Ce type d'information est aussi appelé «signalisation».

Le codage des bits est réalisé par des niveaux de tension. Un seul fil est nécessaire par sens de transmission (plus une «terre» commune, c'est d'ailleurs plutôt un «retour commun» qu'une terre).

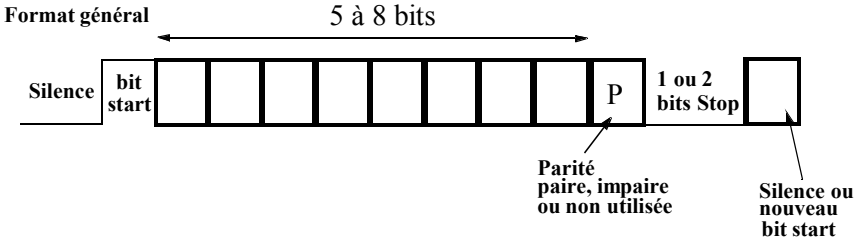
Ceci est un exemple, il en existe d'autres. Le codage par niveau de tension est simple à réaliser mais il est peu performant en termes de distance (quelques dizaines de mètres au plus) et de débit (environ 100Kb/s max).

Pour les plus grandes distances et les plus forts débits on privilégie les niveaux de courants. Pour cela il faut deux fils par jonction (2 fils par signal). Ce mode de codage est plus résistant aux perturbations électromagnétiques.

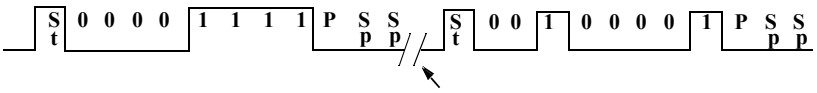
Jonction ITU-T V24 – ANSI RS232-C
Structuration de l'information

99/307

Structure orientée octet



Exemple : transmission de 2 caractères en mode parité paire avec 2 bis Stop

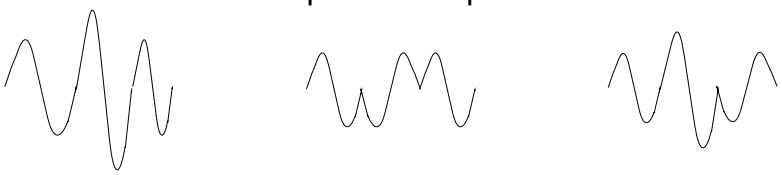


Les schémas ci-dessus présentent des «1» sous formes d'impulsions positives (crénaux vers le haut), contrairement à la réalité V24/V28 où les «1» sont codés par des tensions négatives.

Modems RTC
Exemple de couche physique

100/307

- Modulation en amplitude et phase

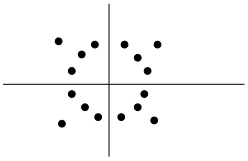


Modulation en amplitude | Modulation de phase | Modulation en amplitude et en phase

- Codage des bits
  - une fréquence,
  - une amplitude et une phase est un état
  - x amplitudes, y sauts de phase, N états
  - un état code $\log_2 N$ bits (exemple : pour 16 états on peut coder 4 bits par état)

Modems RTC : bits et bauds
Codage de l'information
101/307

- Le nombre de changement d'états par seconde s'appelle la rapidité de modulation, elle s'exprime en bauds
- Le nombre de bits par seconde s'appelle le débit
- La relation entre le débit et la rapidité de modulation s'exprime par la relation suivante :
  - $D = R \log_2 N$ (N est le nombre d'états)
- Ce type de modulation se nomme «modulation d'amplitude à quadrature de phase» ou QAM (Quadrature Amplitude Modulation) et peut se représenter ainsi :



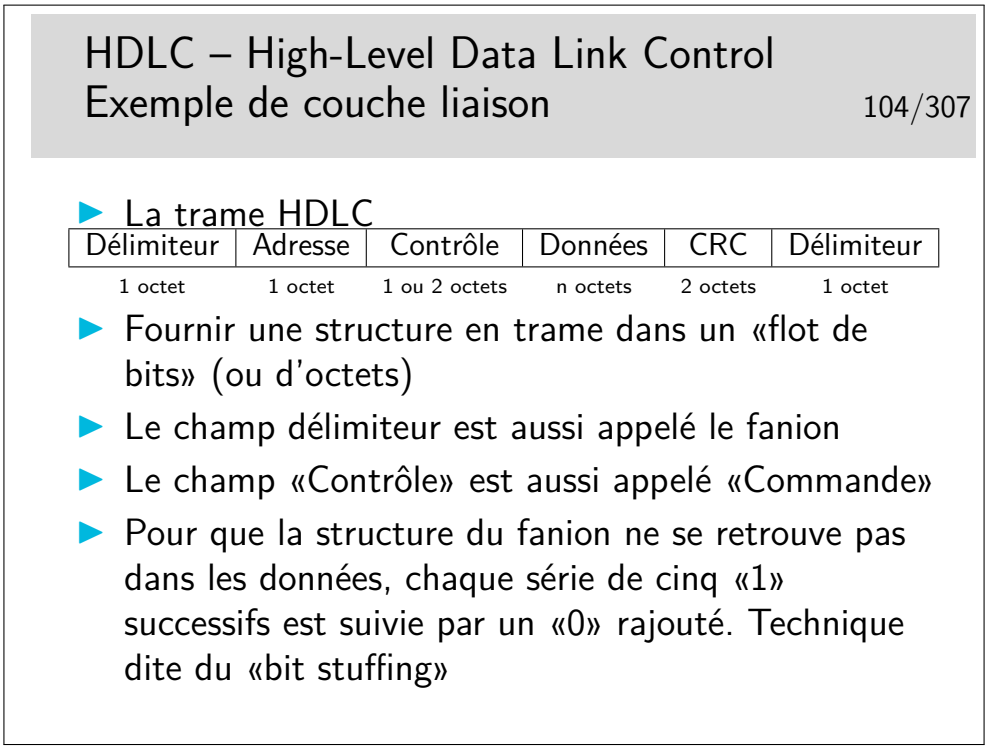
Phases espacées de 30° (12 phases)
Une amplitudes pour toutes les phases sauf 45°, 135°, 225° et 315° qui ont deux amplitudes.
16 états, 4 bits par états.
Si R = 2400 bauds, D = 9600 b/s

Une horreur à ne pas dire : «des bauds par seconde !» car alors ce n'est plus un débit mais une accélération.

Modems RTC : modulation et débit
Codage de l'information
102/307

- Le débit est exprimé en bits par secondes
- Sa valeur maximum dépend des caractéristiques physiques du canal et de la capacité de l'électronique de réception de reconnaître le signal utile dans le signal composite formé par le signal lui même et les perturbations électromagnétiques incontournables (ce qu'on appelle le bruit)
- Les caractéristiques physiques du canal permettent de définir une bande passante maximum : W (coupure à 3db)
- La puissance du bruit est estimée par rapport à la puissance du signal lui même dans le rapport Signal sur Bruit (S/B)
- Résultat fondamental (Shannon 1948) $D_{max} = W \log_2(1 + \frac{S}{B})$

4.3 Exemple de couche liaison de données



HDLC : High-level Data Link Control (ne pas trop chercher le sens profond de cet acronyme... Se souvenir cependant fermement de son nom et de ce qu'il représente...).

Le champ adresse est aujourd'hui très mal nommé car ce type de structure de données n'est généralement utilisé que pour des liaisons point-à-point. Donc, ici, l'adresse ne sert pas à l'adressage... Il sert à distinguer des trames de commandes et des trames de réponses. Un trame de commande est une trame émise sans qu'elle soit demandée, l'émetteur décide seul de l'émettre. Une trame de réponse est une trame émise en «réponse» à une trame de commande.

Les protocoles des niveaux supérieurs ne prévoient pas de délimitation de leurs PDUs. Chaque fois qu'on ne sait pas comment faire pour véhiculer de telles unités de données on a recours à HDLC, en le simplifiant parfois. C'est par exemple le cas du protocole PPP (Point to Point Protocol) utilisé pour véhiculer les datagrammes IP sur les liaisons de type série, telles que celle utilisées par les particuliers pour se connecter à Internet via des modems.

Dans le cas de PPP sur lignes séries de type PC (entre le PC et le modem et ensuite via la modulation du modem) les trames HDLC sont découpées octet par octet et émises en mode asynchrone avec, par octet, un bit START et un ou deux bits STOP (plus, éventuellement, un bit de parité). Le fanion est constitué par l'octet 0x7E, donc comme dans le cas synchrone. Si cet octet se retrouve dans les données, il est remplacé par deux octets.

Protocole HDLC-LAPB
Exemple de couche liaison

105/307

- Une spécialisation des trames HDLC (porter X.25)
- Adresse : 0xC0 ou 0x80 suivant le type Commande ou Réponse et le sens de la trame (ETTD→ETCD ou l'inverse)
- Différents types de trames :
  - I (*Information*) : trames portant des numéros et des acquittements
    - le champ contrôle est sur 1 octet (numérotation modulo 8) ou sur 2 octets (numérotation modulo 128)
  - S (*Supervision*) : trames ne portant que des acquittements
    - le champ contrôle est sur 1 ou 2 octets suivant le modulo de numérotation
  - U (*Unumbered*) : trames de commande
    - le champ contrôle est sur 1 octet
    - trames servant à établir et rompre la relation (communication) entre les deux extrémités de la liaison

LAPB signife : «*Link Access Protocol, Balanced*», ce qui traduit en français donne «protocole d'accès au lien en mode équilibré». En fait il utilise le Asynchronous Balanced Mode (ABM) de HDLC, mode dans lequel il n'y a pas de maître/esclave, les deux parties peuvent accéder indifféremment au lien.

On ne sait pas trop comment ces appellations ont été définies et par quel normalisateur (il ne s'en vante pas celui-là). Ce n'est pas très important... Il est, par contre, très important de savoir que cela existe et se nomme ainsi et aussi comment cela fonctionne, ce que nous allons voir dans ce qui suit.

La numérotation des trames et les acquittements permettent d'utiliser le mécanisme de la fenêtre d'anticipation.

L'utilisation du champ adresse est parfaitement défini dans la recommandation ITU-T X25, le lecteur se référera à ce document pour les détails techniques si nécessaire. Ces détails n'apportent rien de fondamental au concept général, on ne fournira pas, ici, d'explications complémentaires.

Protocole HDLC-LAPB
Structure du champ «contrôle» 106/307

TABLEAU 2-7/X.25

Commandes et réponses LAPB – Fonctionnement de base (modulo 8)

			1	2	3	4	5	6	7	8
Format	Commandes	Réponses	Codage							
Transfert d'information	I (Information)		0	N(S)			P	N(R)		
Supervision	RR (Prêt à recevoir)	RR (Prêt à recevoir)	1	0	0	0	P/F	N(R)		
	RNR (Non prêt à recevoir)	RNR (Non prêt à recevoir)	1	0	1	0	P/F	N(R)		
	REJ (Rejet)	REJ (Rejet)	1	0	0	1	P/F	N(R)		
Non numéroté	SABM (Mise en mode asynchrones symétrique)		1	1	1	1	P	1	0	0
	DISC (Déconnexion)		1	1	0	0	P	0	1	0
		DM (Mode déconnecté)	1	1	1	1	F	0	0	0
		UA (Accusé de réception non numéroté)	1	1	0	0	F	1	1	0
		FRMR (Rejet de trame)	1	1	1	0	F	0	0	1

Les trames `I` sont aussi dites «trames numérotées». Elles sont munies, à cet effet, d'un champ `N(S)` appelé «numéro de séquence en émission», en pratique c'est le numéro de la trame (ici modulo 8, mais on peut numéroter modulo 128 ou 32768, le champ «contrôle» est alors codé sur 2 ou 4 octets).

Le champ `N(R)` est appelé «numéro de séquence en réception», en pratique il s'agit de l'acquittement. Pour une valeur x, il signifie : «vous pouvez m'envoyer la trame $n°x$».

Le fait que les trames d'informations peuvent porter des acquittements s'appelle «piggy backing»

Les trames de supervision servent à porter uniquement des acquittements et à réaliser la fonction de contrôle de flux :

RR : Receiver Ready avec `N(R) = x` : «je suis prêt, vous pouvez m'envoyer la trame x», sous-entendu : «j'ai bien reçu toutes les trames jusqu'à $x-1$»

RNR : Receiver Not Ready avec `N(R) = x` : «j'ai bien reçu la trame $x-1$, mais je ne peux plus recevoir de trames momentanément, arrêtez l'envoi»

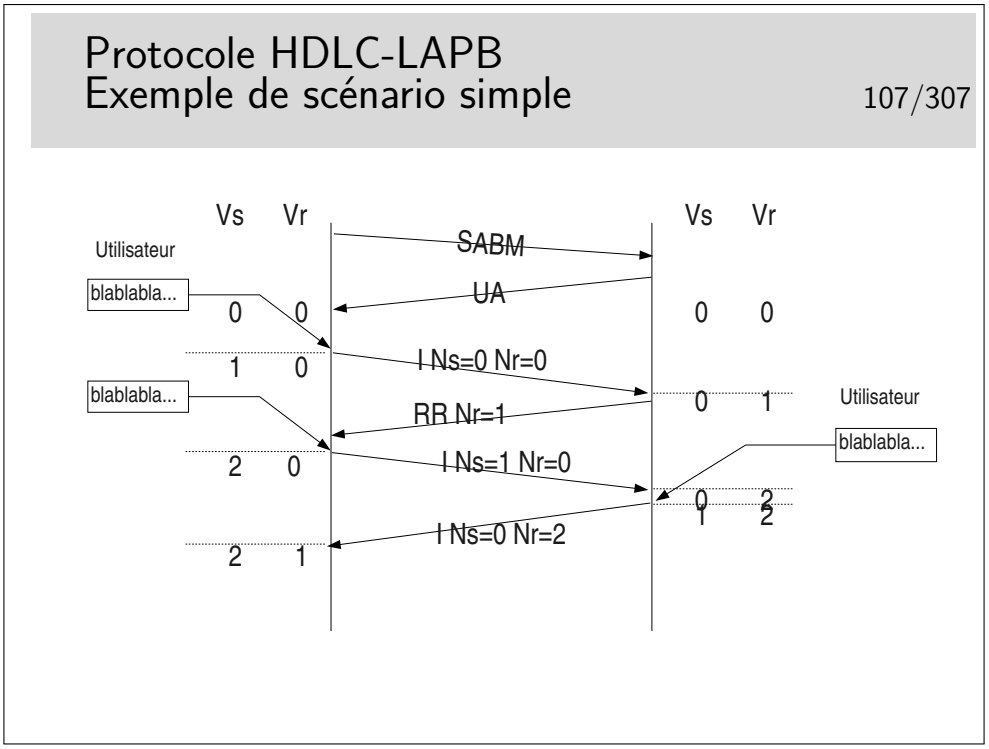
REJ : Reject avec `N(R) = x` : «Hop là !!! Erreur, vous m'envoyez n'importe quoi alors que j'attends la trame $n°x$!»

SABM : Set Asynchronous Balanced Mode (inutile de chercher à traduire) : établissement de la connexion

UA : Unumbered Acknowledge (acquittement non numéroté)

DISC : Disconnect (déconnexion)

On ne s'attardera pas ici sur le rôle du bit P/F...

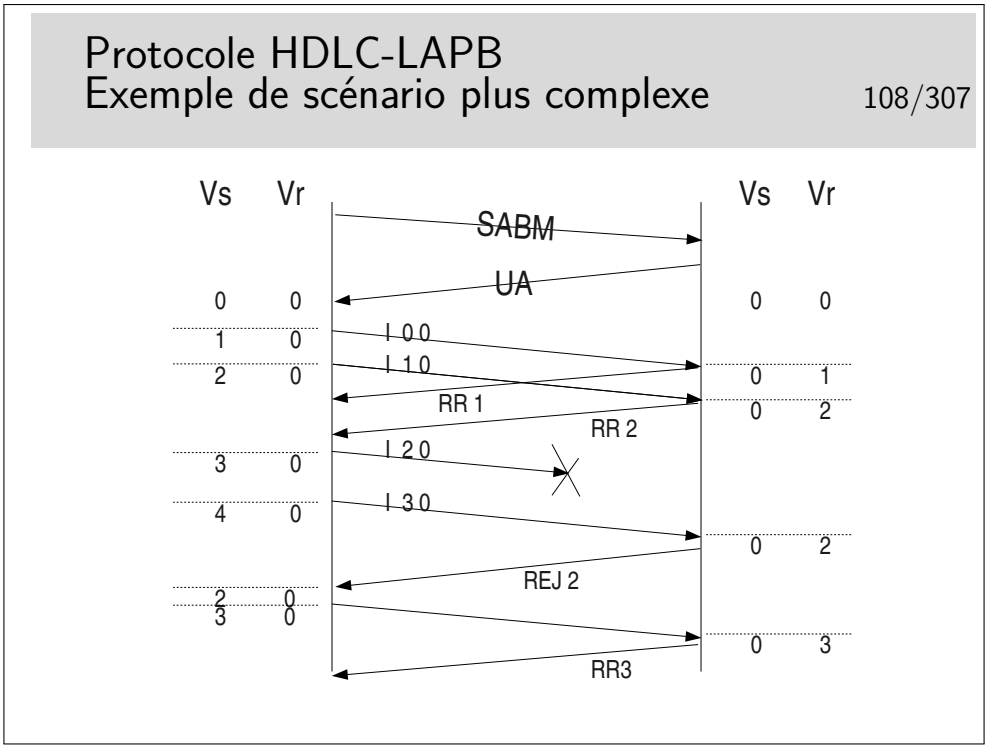


Les entités protocolaires d'extrémités (donc des logiciels qui implémentent le protocole LAPB) sont munies de deux compteurs `Vs` et `Vr`. `Vs` sert à numéroter les trames à émettre, `Vr` sert à vérifier que les trames que l'on reçoit portent le bon numéro.

Au départ, lors de l'initialisation de la connexion par l'échange `SABM/UA`, les compteurs sont mis à 0. L'utilisateur coté gauche désire envoyer des données. Celles-ci sont encapsulées dans une trame. Le champ `Ns` de celle-ci va être rempli par la valeur de `Vs`, le champ `Nr` par la valeur de `Vr`. On recopie tout simplement `Vs` et `Vr` dans `Ns` et `Nr`. Un fois la trame envoyée on se préparera à envoyer une trame suivante si nécessaire. Celle-ci verra son numéro progresser de 1. On prépare donc cela en faisant progresser `Vs` de 1. Lors de la seconde émission on recopie `Vs` et `Vr` dans `Ns` et `Nr` de cette seconde trame, puis on l'émet et on incrémente `Vs` de 1.

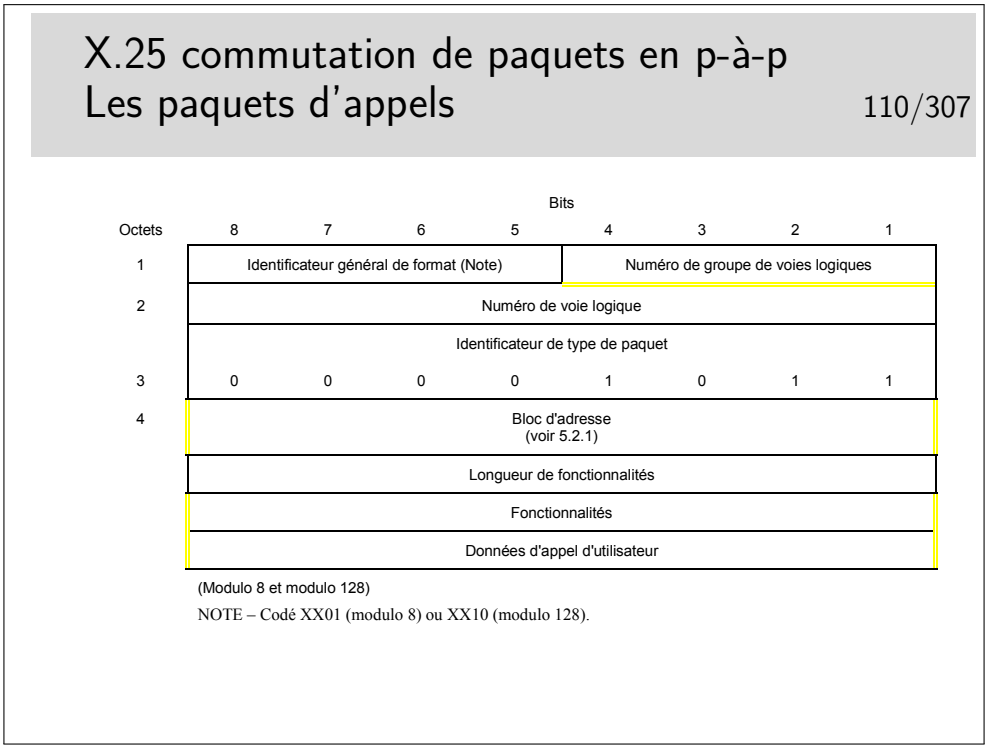
Lorsque le récepteur (droite) reçoit la première trame il vérifie que son numéro de séquence `Ns` est bon en le comparant avec son compteur `Nr`. Les deux valeurs étant égales, la trame est acceptée. Le compteur `Vr` droit progresse de 1, une trame d'acquittement est envoyée avec son `Nr = 1` (recopie du `Vr`).

Plus tard, le coté droit désirera parler à son tour et cet événement surviendra juste au moment de la réception d'une trame (dans notre scénario). Il émettra donc directement une trame `I` portant le bon `Ns` et bien sur le bon `Nr`. (Piggy backing)



Pour quelle raison la trame numéro 2 (I 2 0) peut-elle ne pas être reçue ?

4.4 Exemple de couche réseau

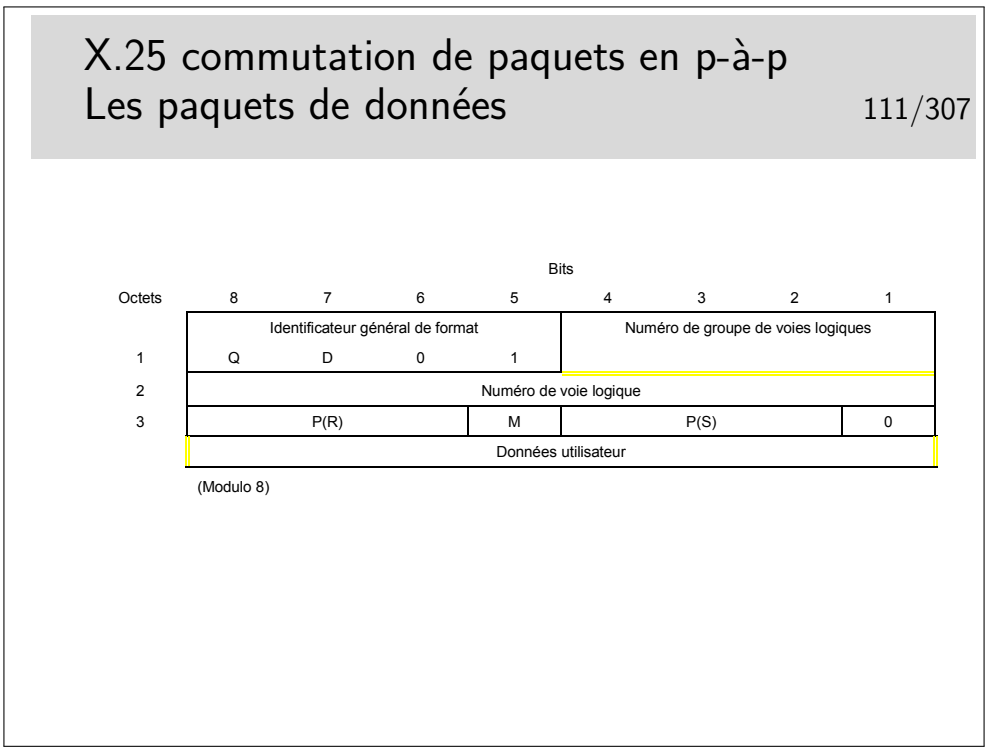


La recommandation X.25 «suggère» que pour un appel sortant (coté demandeur), l'ETTD choisisse le numéro de voie logique le plus haut disponible. Pour un appel entrant (coté demandé) il est suggéré que l'ETCD choisisse le numéro de voie logique le plus bas

disponible.

On rappelle que l'ETTD (Équipement Terminal de Traitement de Données, ou DTE en anglais) est l'équipement «terminal» (l'ordinateur) et que l'ETCD (Équipement terminal de circuit de données, ou DCE en anglais) est l'équipement «réseau» (le commutateur de raccordement).

Le paquet d'appel est muni d'un bloc d'adresse contenant l'adresse de destination et possiblement l'adresse source. Il se comporte dans le réseau comme un datagramme. Il n'y a pas de système de signalisation (protocole pour l'établissement, la rupture et la supervision des communications) comme dans d'autres types de réseau à commutation (téléphone, ATM, Frame Relay, MPLS)



Les paquets de données sont numérotés (champ `P(S)` : numéro de séquence en émission). Ils contiennent aussi un champ `P(R)` permettant de transporter les acquittements (piggy backing)

X.25 commutation de paquets en p-à-p
Les paquets de supervision 112/307

RR

Octets	Bits 8	7	6	5	4	3	2	1
1	Identificateur général de format				Numéro de groupe de voies logiques			
	Q	D	0	1				
2	Numéro de voie logique							
3	P(R)			M	P(S)			0
	Données utilisateur							

(Modulo 8)

RNR

Octets	Bits 8	7	6	5	4	3	2	1
1	Identificateur général de format				Numéro de groupe de voies logiques			
	Q	D	0	1				
2	Numéro de voie logique							
3	P(R)			M	P(S)			0
	Données utilisateur							

(Modulo 8)

REJ

Octets	Bits 8	7	6	5	4	3	2	1
1	Identificateur général de format				Numéro de groupe de voies logiques			
	Q	D	0	1				
2	Numéro de voie logique							
3	P(R)			M	P(S)			0
	Données utilisateur							

(Modulo 8)

Comme au niveau 2 (LAPB), nous avons des paquets **RR** (Receiver Ready), **RNR** (Receiver Not Ready), **REJ** (Reject) permettant de transporter les acquittements, de réaliser le contrôle de flux et la reprise sur erreur.

X.25 commutation de paquets en p-à-p
Tous les types de paquets 113/307

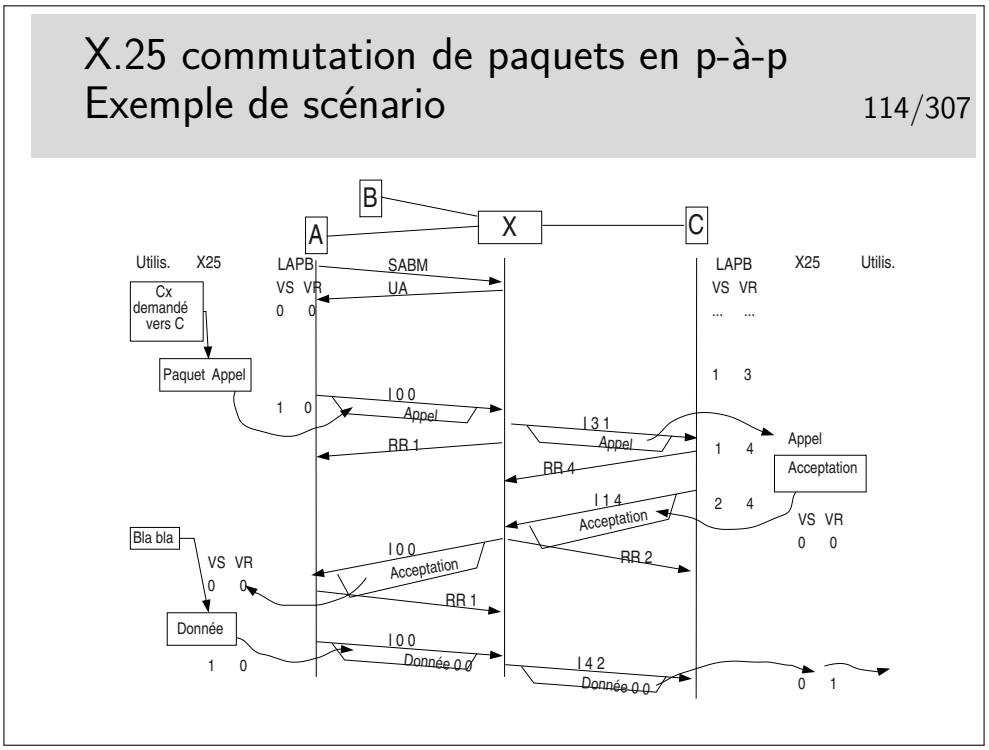
Type de paquet		Service	
de l'ETCD vers l'ETTD	de l'ETTD vers l'ETCD	VC	PVC
Etablissement et libération des communications (Note 1)			
Appel entrant	Demande d'appel	X	
Communication établie	Communication acceptée	X	
Indication de libération	Demande de libération	X	
Confirmation de libération par l'ETCD	Confirmation de libération par l'ETTD	X	
Données et interruption (Note 2)			
Données de l'ETCD	Données de l'ETTD	X	X
Interruption par l'ETCD	Interruption par l'ETTD	X	X
Confirmation d'interruption par l'ETCD	Confirmation d'interruption par l'ETTD	X	X
Contrôle de flux et réinitialisation (Note 3)			
RR par l'ETCD	RR par l'ETTD	X	X
RNR par l'ETCD	RNR par l'ETTD	X	X
	REJ par l'ETTD[a)]	X	X
Indication de réinitialisation	Demande de réinitialisation	X	X
Confirmation de réinitialisation par l'ETCD	Confirmation de réinitialisation par l'ETTD	X	X
Reprise (Note 4)			
Indication de reprise	Demande de reprise	X	X
Confirmation de reprise par l'ETCD	Confirmation de reprise par l'ETTD	X	X
Diagnostic (Note 5)			
Diagnostic[a)]		X	X

VC Communication virtuelle *(virtual call)*
PVC Circuit virtuel permanent *(permanent virtual circuit)*
a) N'est pas nécessairement disponible dans tous les réseaux.
NOTES
1 Voir 4.1 et 6.16 pour les procédures, et 5.2 pour les formats.
2 Voir 4.3 pour les procédures, et 5.3 pour les formats.
3 Voir 4.4 et 6.4 pour les procédures, et 5.4 et 5.7.1 pour les formats.
4 Voir 3.3 pour les procédures, et 5.5 pour les formats.
5 Voir 3.4 pour les procédures, et 5.6 pour les formats.

VC : Virtual Channel (canal virtuel ou circuit virtuel), établi à la demande par un paquet d'appel.

PVC : Permanent Virtual Channel : circuit virtuel permanent, établi par les gestionnaire du réseau à l'aide de procédures propres (commandes dans les commutateurs).

La liste ci-dessus est placée ici pour votre «culture générale», vous n'avez absolument pas besoin de connaître cette liste parfaitement.



Machines terminales (ETTD) : A, B et C.

Commutateur (ETCD) : X

On suppose dans ce scénario qu'une communication est déjà établie entre B et C. On ne s'en préoccupe pas mais le niveau 2 X-C est établi pour supporter cette communication. Des trames ont donc déjà été échangées entre X et C. Au moment où nous arrivons, les compteurs Vs et Vr du contexte LAPB en C sont, respectivement à 1 et 3.

Entre A et X, le niveau 2 n'est pas établi au début du scénario.

118

Deuxième partie
Réseaux locaux Ethernet et 802.3

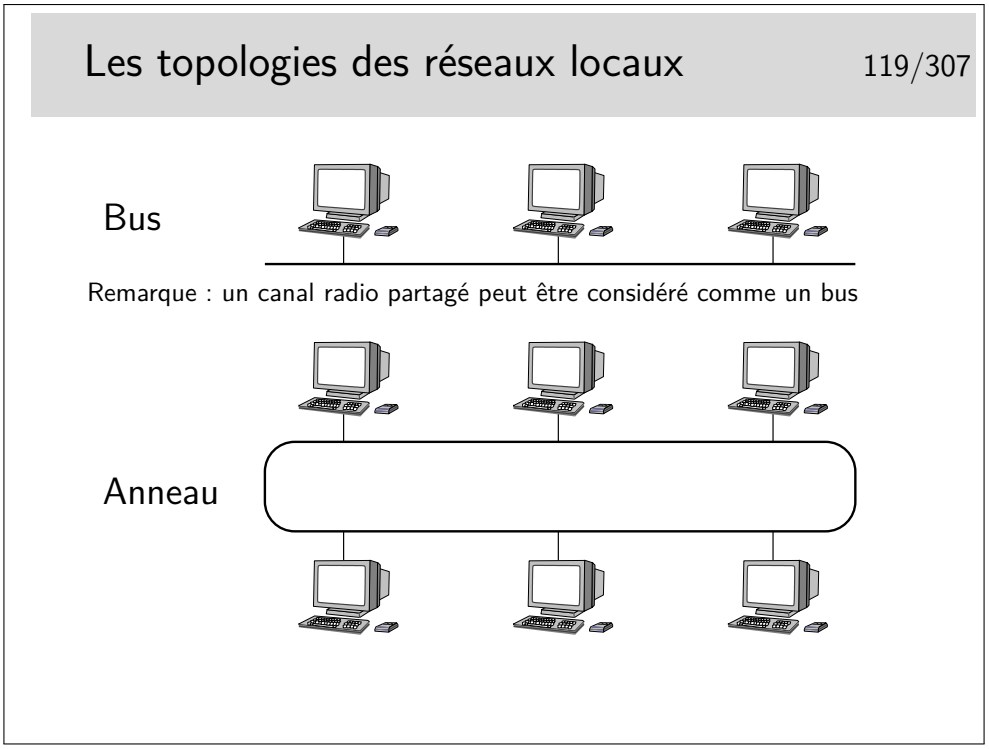
5 Introduction

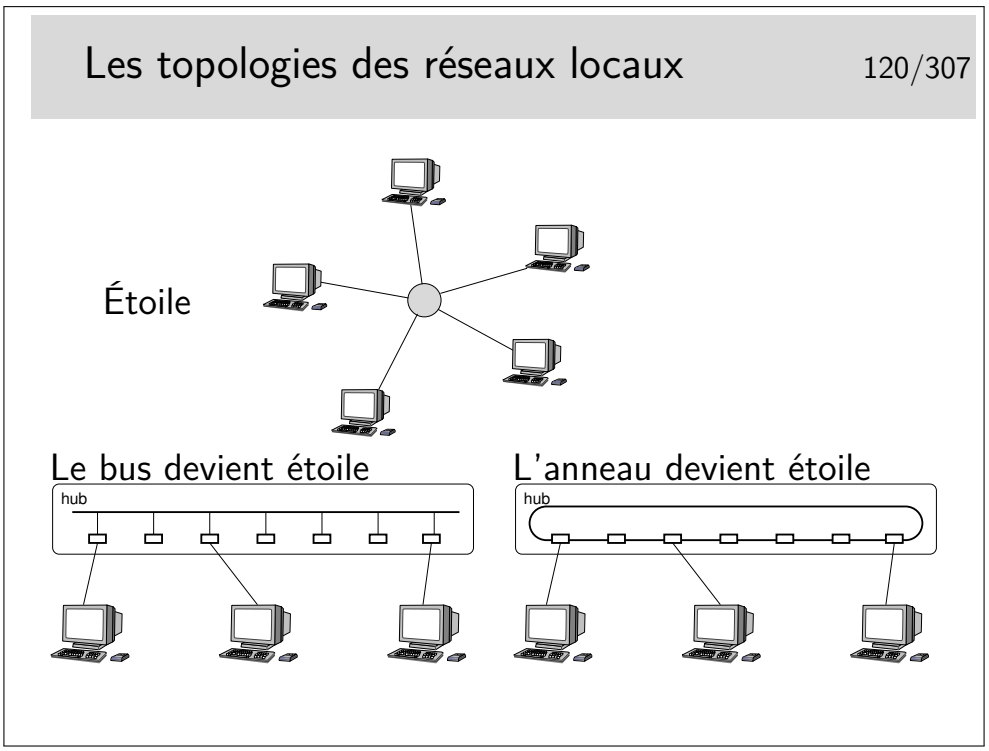
Les réseaux locaux 117/307

- Les réseaux de l'entreprise
- Caractéristiques :
  - Topologie
    - Bus, avec ou sans fil
    - Anneau
    - Étoile
    - Bande de base ou large bande
  - Caractéristiques physiques des supports (les média)
    - Cuivre
    - Fibre optique
    - radio
  - Méthode d'accès au médium
    - Comment une station peut-elle émettre ses données sur le réseau

Les différents types de réseaux 118/307

- Les réseaux locaux : LAN : Local Area Network
  - Réseaux d'entreprise
    - Courtes distances : de quelques centaines de mètres à quelques kilomètres
- Les réseaux métropolitains : MAN : Metropolitan Area Network
  - Interconnexion de réseaux locaux
  - Quelques dizaines à quelques centaines de kilomètres
- Les réseaux grande distance : WAN : Wide Area Network
  - Les réseaux nationaux et internationaux
  - Les réseaux d'opérateurs





La topologie en étoile repose sur un nœud actif central qui contrôle le «droit de parole» des stations et achemine l'information. Cette structure n'a jamais vraiment été fortement déployée à grande échelle, on peut noter cependant les réseaux locaux basées sur le protocole X25, au début des années 80 et les ceux basés sur la technologie ATM dans les années 90 (mais c'était pour émuler Ethernet, c'est à dire un bus).

Les topologies à succès sont le bus (Ethernet, qui a gagné la bataille) et l'anneau (Token Ring, d'IBM, en cours d'abandon).

Au cours du temps, le réseau en bus Ethernet a, cependant, évolué vers une structure en étoile, où les nœuds (appelés «hubs») sont des boîtiers électroniques qui émulent un bus. Le «hub» fonctionne comme un bus.

La technologie Token Ring a, elle, été très rapidement implémentée dans des boîtiers électronique renfermant la structure d'anneau.

Aujourd'hui, donc, la topologie est en étoile, mais l'ensemble fonctionne comme un bus (ou encore parfois comme un anneau).

Caractéristiques des canaux 121/307

- Transmission en bande de base
  - Un seul canal physique pour toutes les stations
  - Problème d'accès concurrent
- Transmission en large bande
  - Plusieurs canaux sur le médium
  - Un canal est caractérisé par une bande de fréquence
  - Les signaux émis par les stations sont transposés dans la bande de fréquence qui est assignée aux stations
  - Un canal donné peut être vu comme un canal en bande de base (émulation d'un bus Ethernet par exemple)

Les méthodes d'accès au médium 122/307

Très liées à la topologie (topologie logique)

- Bus
  - Tout le monde partage le canal sans priorité
  - Chaque station peut, à priori, émettre lorsqu'elle le désire
    - Il en résulte des collisions
    - Il faut gérer le problème des collisions
      - Algorithme spécifique de contention (CSMA CD ou CA...)
      - Émulation d'un anneau logique
- Anneau
  - Le droit d'émettre est transmis sur l'anneau dans une trame spécifique contenant un bit spécial appelé «jeton»
  - Les trames circulent dans un sens donné

La standardisation des Réseaux Locaux 123/307

- Menée à bien par l'organisme IEEE
  - Plus particulièrement le comité 802 de l'IEEE
    - Chaque topologie et les divers protocoles et caractéristiques sont étudiés et standardisés par un sous-comité 802
    - Exemples : 802.3 pour Ethernet, 802.5 pour Token Ring
- Modèle en couches spécifique
  - Comparable au modèle OSI pour ses deux premières couches
  - Trois couches
    - La couche physique (comme pour OSI)
    - La couche «Medium Access Control» (MAC)
    - La couche «Logical Link Control» (LLC)

Quelques sous-comités IEEE-802 124/307

- 802.1 : architecture des réseaux locaux
  - Architecture générale, interconnexion (niveau 2), QoS, etc...
- 802.2 : la couche LLC
- 802.3 : Ethernet
  - 802.3u : Ethernet 100Mb/s
  - 802.3ab, z : Ethernet 1Gb/s
  - 802.3ae : Ethernet 10Gb/s
- 802.4 : le bus à jeton
- 802.5 : l'anneau à jeton (Token Ring)
- 802.11 : les réseaux sans fils (WiFi)
- 802.15 : les WPAN : Wireless Personal Area Network

Beaucoup de sous comités :

— 802.1d : techniques de pontage

— 802.1p/q : classes de service, techniques des VLANs (Virtual LAN)

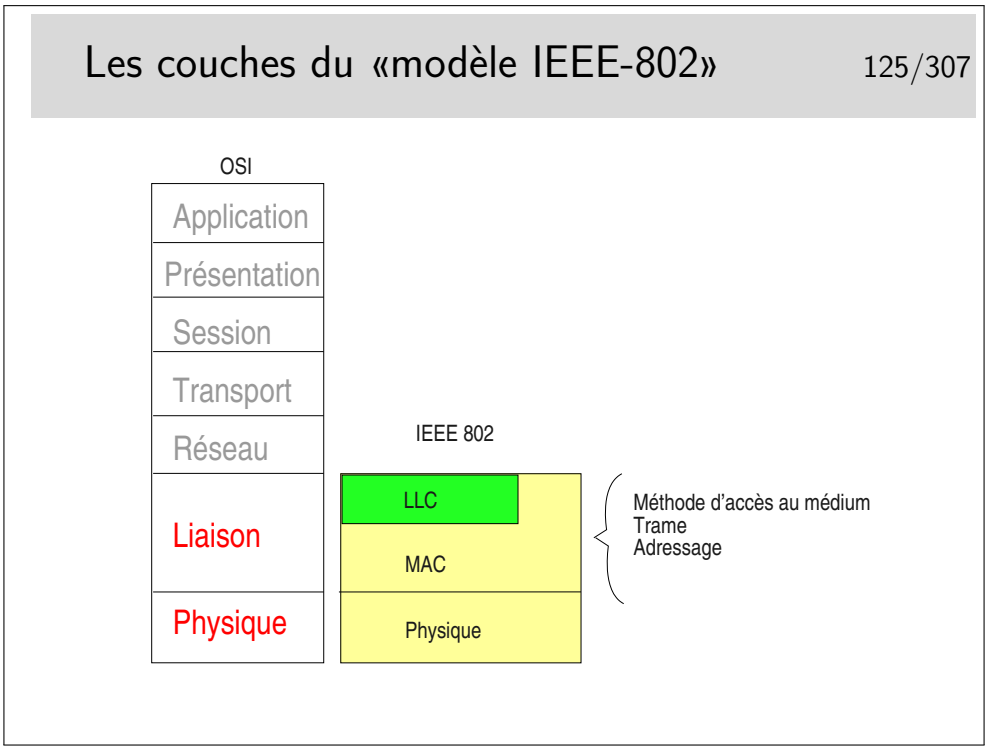
— 802.3u : Ethernet

— 100Mb/s 802.3z : Ethernet 1Gb/s

— 802.3ab : Ethernet 10Gb/s

— 802.11a : sans fils, 5Gz (projet)
— 802.11b : sans fils, 11Mb/s
— 802.11g : sans fils, 54Mb/s

Note : WiFi (Wireless Fidelity) est une dénomination commerciale pour le 802.11

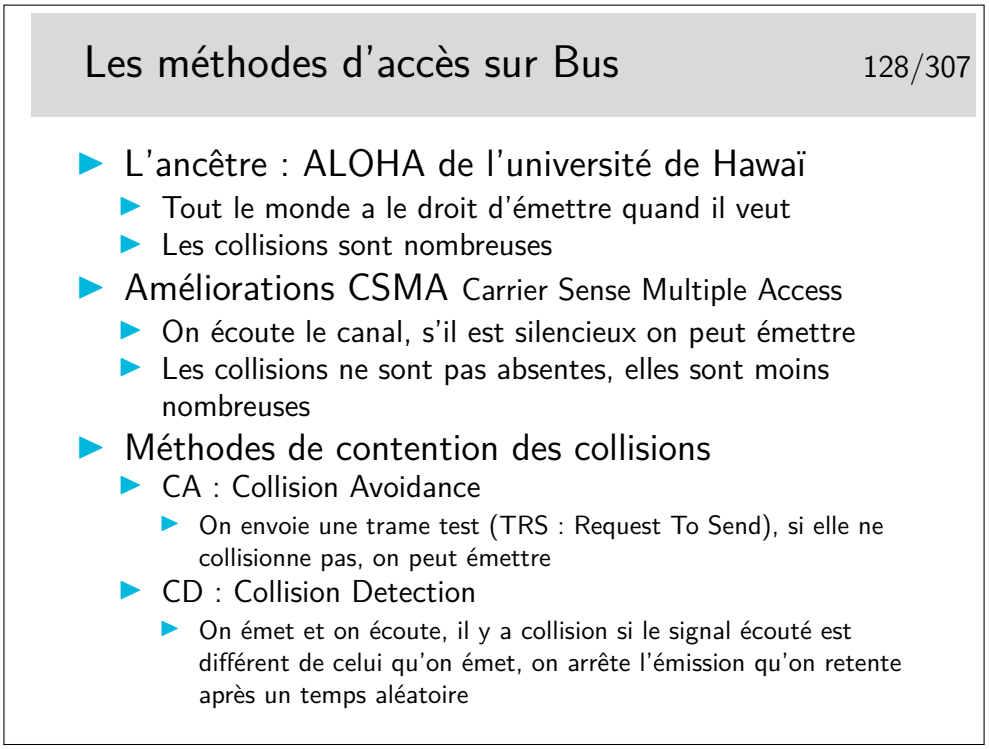


La couche LLC permet de définir différents types de liaisons (avec/sans connexion et avec/sans acquittement). Elle n'est pas toujours obligatoire, en Ethernet elle est optionnelle, certains protocoles de niveau 3 l'utilisent, d'autre pas. Par exemple IP, sur Ethernet, n'emploie pas, par défaut, la couche LLC (mais peut le faire, le choix se fait par paramétrage au niveau du système d'exploitation).

La couche MAC est centrale, elle définit l'algorithme gérant l'accès concurrent au support. Elle définit aussi une structure de trame ainsi qu'un mécanisme d'adressage (les adresses MAC, nous verrons cela plus loin).

6 La technologie Ethernet

6.1 Fondements d'Ethernet



Une autre méthode consiste à créer un anneau virtuel. Les stations s'échangent un jeton dans un ordre donné (la station A passe le jeton à la station B qui le passe à C, etc. La station qui désire émettre doit attendre de voir arriver le jeton). Ce type de réseau a été développé par des industriels des mondes de l'automobile et de l'aviation américains. Il fait partie des «réseaux locaux industriels». Il a été standardisé sous le numéro 802.4 par l'IEEE. Il ne s'est pas développé de manière importante.

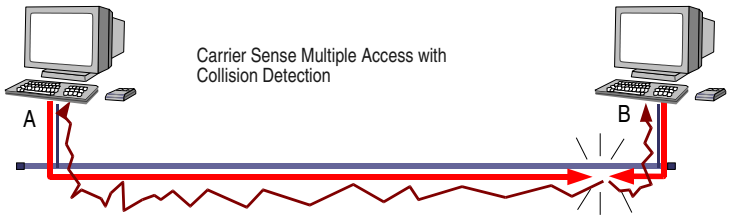
La technologie Ethernet 129/307

- La technologie impérialiste, elle a écrasé toutes les autres (ou les autres s'adaptent à elle, exemple le WiFi)
- Topologie logique : le bus
  - Aujourd'hui, topologie physique en étoile avec hubs et commutateurs (switches)
- Méthode d'accès
  - CSMA-CD : Carrier Sense Multiple Access with Collision Detection
- Origine : Intel, Xeros, Digital, première idée en 1976 (Bob Metcalfe)
- Standardisation générale : IEEE-802.3
- Des débits divers : 10, 100, 1Gb/s, voir 10Gb/s

La technologie WiFi (IEEE-802.11) n'a rien à voir, à priori, avec Ethernet, mais on commence à l'appeler couramment «l'ethernet sans fil», c'est dire... Elle s'adapte, en effet, très bien à Ethernet via des ponts simples à mettre en place.

La technologie Ethernet évolue de manière souple où chaque étape d'évolution ne remet pas en cause les versions antérieures. On peut raccorder une interface 10Mb/s sur une interface 100 ou 1000Mb/s. Les matériels d'interconnexion sont compatibles avec les trois débits.

La méthode d'accès CSMA-CD I 130/307



- La station A écoute le réseau, il n'y a pas de signal, elle peut émettre
- Le signal se propage
- La station B écoute le réseau, le signal de A ne lui est pas encore parvenu, elle décide d'émettre
- Les deux signaux vont collisionner, le signal résultant va se propager de part et d'autre et va parvenir aux deux stations qui continuent d'écouter
- Chaque station continue à émettre quelques instants pour renforcer la collision et s'arrêtent
- Chacune tire un temps aléatoire au bout duquel elles tentent une ré-émission

La méthode d'accès CSMA-CD II 131/307

- Si une station émet pendant au moins 1 RTT alors il n'y aura plus de collision non détectée
- Si une collision est détectée, chaque station en cause arrête son émission
  - Chaque station considère alors 2 intervalles de temps de valeur RTT et tire aléatoirement 1 ou 2 et ré-émet tout de suite (si 1 est tiré) ou un RTT plus tard (si 2 est tiré)
  - Si une nouvelle collision intervient, on considère alors 4 RTT, et on tire aléatoirement entre 1 et 4. On tente une émission au début de l'intervalle de temps tiré
  - En Ethernet, on peut tenter jusqu'à 16 réémissions mais on ne multiplie par 2 que jusqu'à 10 fois le nombre de RTT

Bus Ethernet : quel est le diamètre du réseau ? 132/307

Ou encore : quelle est la distance maximale entre deux stations ?

- Pour détecter une collision il faut que les deux stations soient encore en émission lorsque le signal de collision leur revient
- Cas le plus défavorable : elles sont situées aux deux extrémités du réseau et la station B émet un court instant avant que le signal de A ne lui parvienne. La collision a lieu près de B
  - Pour que A détecte la collision il lui faut attendre un temps égal à la durée de propagation de A jusqu'à B et retour
  - On appelle ce temps le Round Trip Delay, le Round Trip Time (RTT), la «tranche canal» ou encore la fenêtre de collision
  - Ce temps dépend : de la taille du segment de données (trame), de la durée d'émission de cette trame et de la distance entre les deux stations les plus éloignées

Quelle est la dimension du réseau (son diamètre ou encore la distance maximale entre deux stations) si la taille minimale des trames est de 512 bits (64 octets), le débit de 10Mb/s et la célérité du signal sur le câble de 200Km/s (c'est sous évalué...) ?

Vous devez trouver 5,12Km, ce qui est trop. Dans la réalité il faut tenir compte de l'affaiblissement en ligne qui ne permet pas de propager un signal sur une telle distance sans pertes de puissance et distorsion en ligne. Ces phénomènes obligent à placer des or-

ganes régénérateurs à des distances bien plus courtes. On appelle ces organes de répéteurs car ils «répètent» les bits, ils les régénèrent. Ce ne sont pas des amplificateurs car alors les distorsions seraient elle-mêmes amplifiées.

Les premiers réseau Ethernet pouvaient compter des segments de 500m maximum, reliés par des répéteurs.

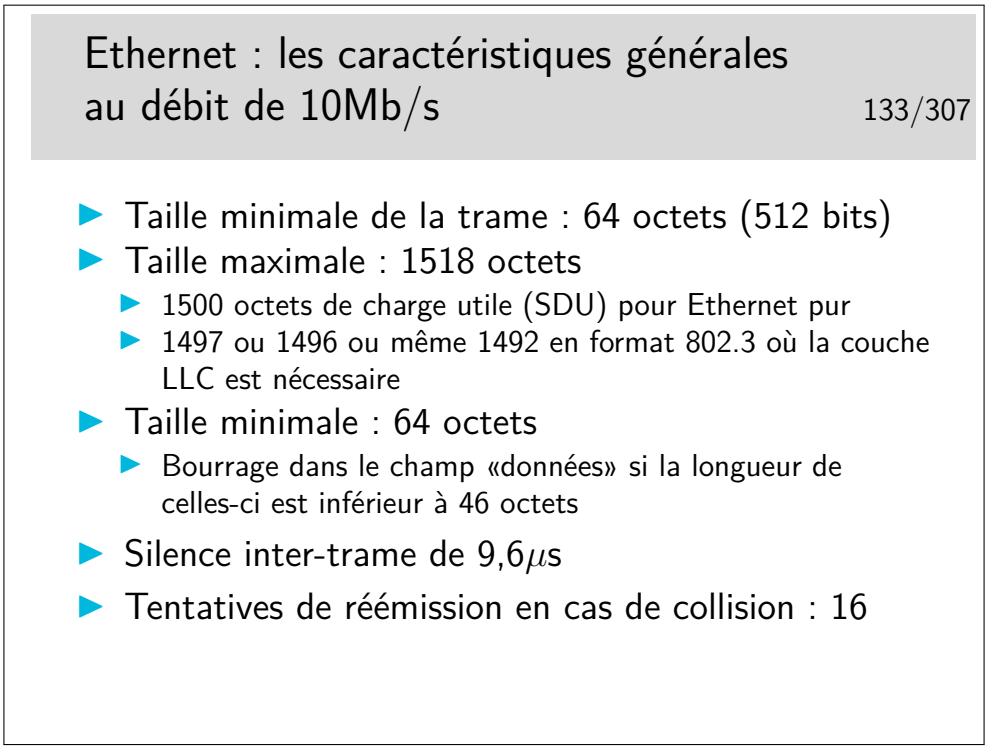
Les répéteurs apportent un retard, les câbles de raccordement de ces répéteurs au câble principal aussi. Le bilan des délais est tel que la taille maximale est de 2,5km pour les valeurs suivantes :

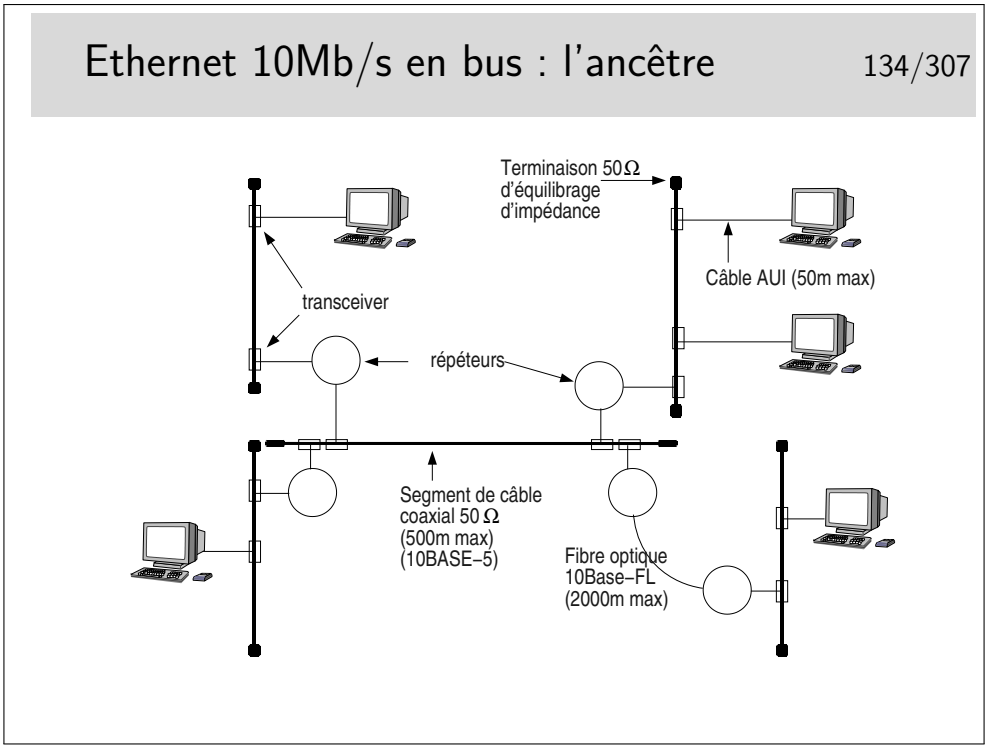
— taille minimale des trames : 512 bits

— débit d'émission : 10 Mb/s

— célérité sur les segments de câble principaux : 0,77c

— 5 segments de 500 m maximum et 4 répéteurs maximum entre deux stations.





Le médium principal est constitué par des segments de câble 50Ω de 500m maximum, reliés entre eux par des répéteurs. Il ne peut y avoir plus de 5 segments entre deux stations (4 répéteurs). Les câbles sont terminés par un «bouchon» constitué par une résistance de 50Ω permettant d'équilibrer l'impédance caractéristique du support.

Un répéteur est un organe qui «répète» sur tous les ports les bits qui arrivent sur un port. Ce n'est pas une amplification car si cela était les altérations du signal seraient amplifiées. Les bits entrant dans un port sont reconnus et régénérés sur tous les autres ports et même sur les fils émission du port sur lesquels ils arrivent.

Les stations et les répéteurs sont raccordés au câble principal via des câbles de 50m maximum. Le câble arrive sur un organe de raccordement directement fixé sur le médium appelé le «transceiver».

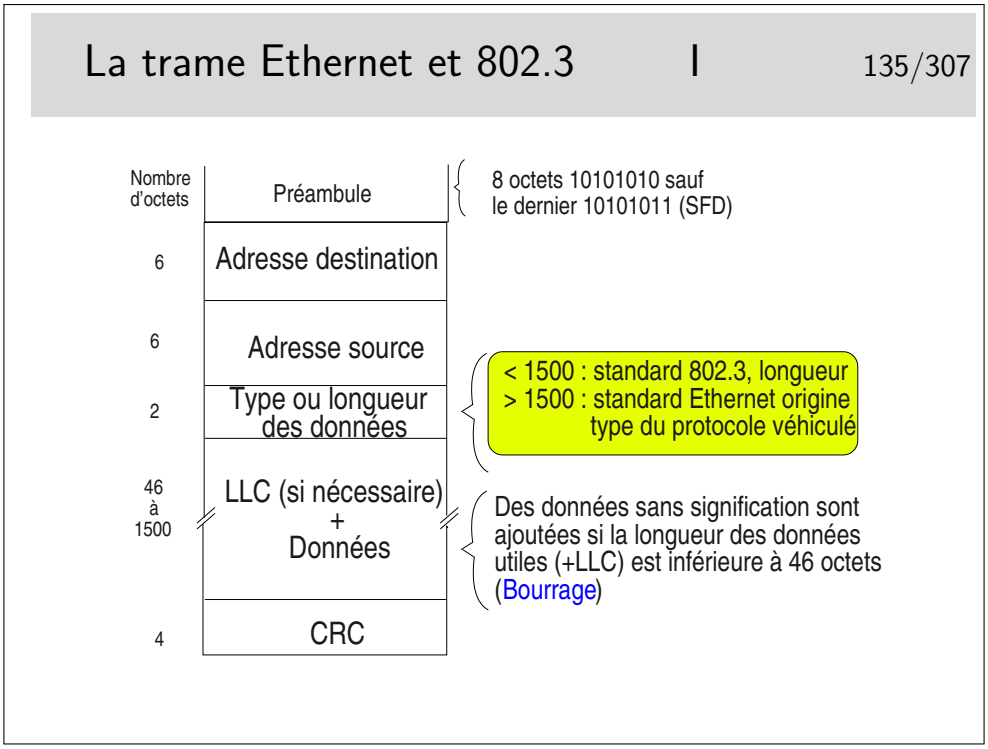
Les transceivers doivent être placés à des distances multiples de 2,5m pour des contrer le phénomène des ondes stationnaires. Un point ou un anneau de couleur noire sur le câble (qui lui est jaune) indique les emplacements possibles.

Deux segments principaux peuvent être reliés par une fibre optique de longueur 2000m maximum. Les deux répéteurs aux extrémités de la fibre ne comptent alors que pour 1.

Ce type de réseau est aujourd'hui abandonné. Les segments de câble sont remplacés par des hubs (ou des switches) et l'architecture physique est devenue ainsi une étoile tout en continuant à fonctionner avec une technique de bus. Le hub peut être comparé à ne boite dans laquelle on aurait enfermé le câble et les transceivers.

Les dénominations 10BASE-5 et 10BASE-FL sont explicitées plus loin.

Il existe une version «cheaper net» comportant des câbles coaxiaux fins (de couleur noire), avec un raccordement par prises de type BNC. Ces câbles font 185m max (10BASE-2).



SFD : Start Frame Delimitor

Les créateurs d'Ethernet (Bob Metcalfe et Intel/Xeros/Digital)) ont défini le champ «Type» pour porter l'identité du protocole véhiculé dans les données (le champ type est le SAP). Pour des soucis d'interopérabilité avec des réseaux locaux dont les trames n'ont pas ce champ type (Token Ring 802.5), le comité 802 a décidé qu'il serait obligatoire d'utiliser la couche LLC pour porter l'identité du protocole véhiculé et a transformé le rôle de ce champ en indicateur de longueur des données. Ce n'est pas une mauvaise idée en soi.

Et en pratique ?...

En pratique les deux coexistent sur les mêmes supports. Une machine peut émettre en 802.3 pour certaines applications et en Ethernet pur pour d'autres.

Par défaut IP est véhiculé en Ethernet pur (type 0800hexa), il peut être véhiculé en mode 802.3 en paramétrant l'interface.

La trame Ethernet II 136/307

- C'est un datagramme
  - Elle contient l'adresse de la station destinatrice ainsi que l'adresse de la station qui émet
  - Elle contient un CRC, on peut donc vérifier son intégrité en réception
    - Si cette vérification montre une altération, la trame est jetée
- Il n'est pas prévu à ce niveau (MAC) d'échange préalable pour établir une relation entre l'émetteur et le récepteur
  - Il n'y a pas de connexion
  - Il 'y a pas de contrôle de flux
  - Il n'y a pas de récupération sur erreur

6.2 Les adresses MAC

Les adresses Ethernet 138/307

- Définies dans la couche MAC (adresses MAC)
  - 6 octets
- 3 types d'adresses
  - Les adresses de stations, dites aussi «unicast» :
    - une adresse unique par interface matérielle (une machine peut avoir plusieurs interfaces matérielles)
    - L'adresse d'une interface est affectée par son constructeur
  - L'adresses globale, dite aussi «broadcast»
    - Permet d'envoyer une trame à toutes les stations du réseau, en une seule opération
  - Les adresses de groupes, dites aussi «multicast»
    - Permettent d'adresser un groupe de stations

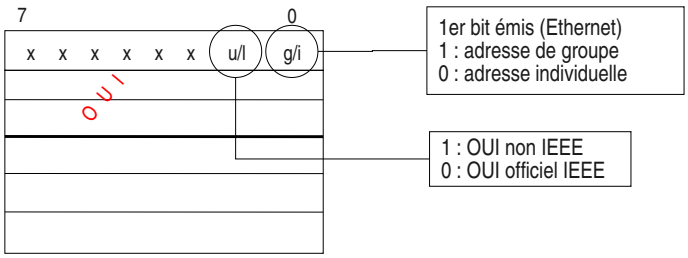
En terminologie anglo-saxonne les interfaces matérielles sont appelées NIC pour *Network Interface Card*.

Une interface voyant passer une trame de broadcast (diffusion) doit prendre en compte cette trame.

Une interface voyant passer une trame de multicast ne la prend en compte que si elle a été paramétrée pour cela.

Format des adresses Ethernet 139/307

- Format IEEE-48
  - 6 octets
    - Les 3 premiers affectés au constructeur par l'IEEE (OUI : Organisation Unit Identifier)
    - Les 3 derniers affectés par le constructeur



Les adresses de braodcast et de multicast 140/307

- Le groupe total : la diffusion ou broadcast
  - `ff:ff:ff:ff:ff:ff`: les 48 bits à 1
- Les groupes restreints : le multicast
  - Le bit de poids faible du premier octet est à 1
  - Quelques exemples :
    - `01:00:5E:xx:xx:xx`: multicast IP
    - `01:80:C2:00:00:00`: spanning tree (protocole de gestion automatique des ponts et commutateurs)
    - `09:00:4E:00:00:02`: multicast IPX

Information sur les OUI et adresses multicast : `http://standards.ieee.org/regauth/oui/oui.txt`

Exemples d'OUI, d'adresses multicats, de types

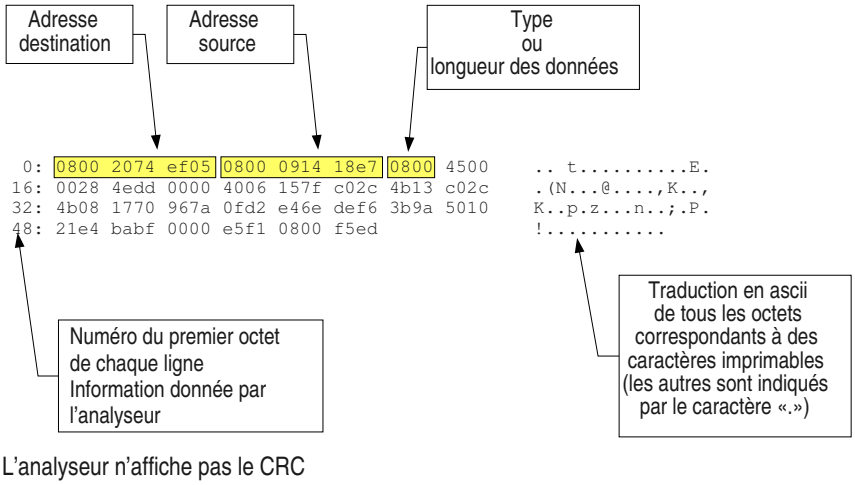
141/307

- Quelques OUI
  - 08-00-07 Apple
  - 00-00-0C Cisco
  - 08-00-08 HP
  - 08-00-20 Sun
- Quelques adresses multicast
  - 01-00-5E-xx-xx-xx Multicast internet
  - 01-80-C2-00-00-00 spanning tree

`http://standards.ieee.org/regauth/oui/oui.txt`

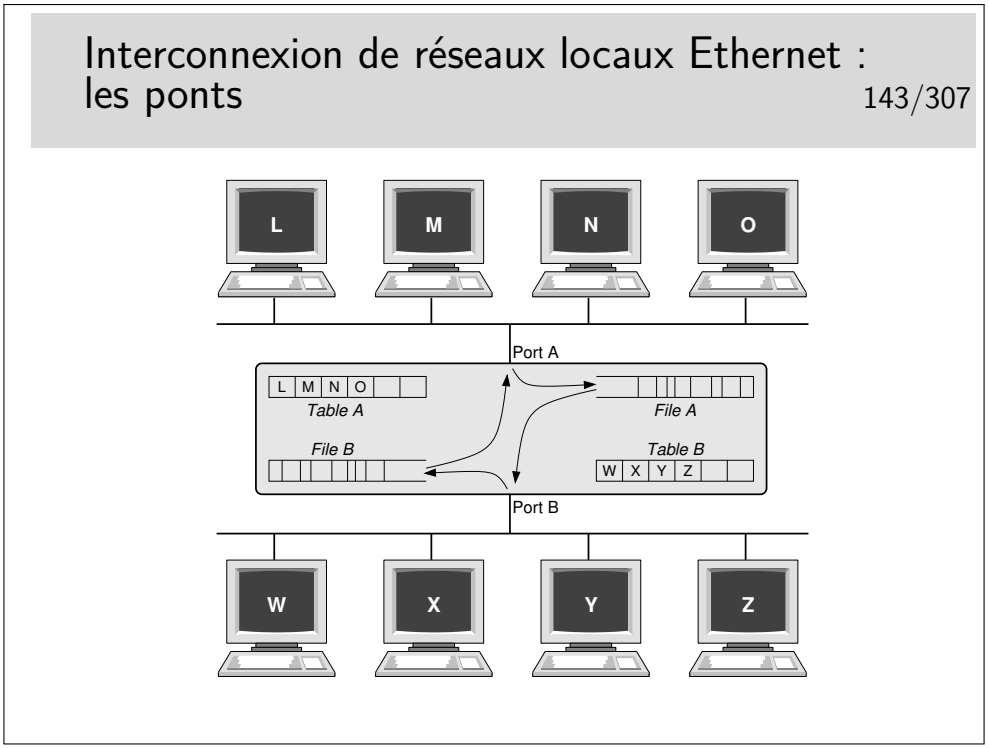
Une trame Ethernet capturée par un analyseur

142/307



L'analyseur n'affiche pas le CRC

Cette trame est au format 802.3 ou Ethernet pur ?



Le pont fonctionne tout seul sans paramétrage préalable. Il écoute le réseau sur son port A et son port B et «voit» passer des trames. Il enregistre les adresses sources de celles-ci et détermine ainsi que L, M, N et O sont du coté A. Ces adresses sont «apprises» par le pont et enregistrées dans sa table A. De même il «apprend» que, coté B, existent les stations W, X, Y et Z. Ces dernières adresses sont stockées dans la table B.

Si une trame est émise par L à destination de O, il reconnaît que ces deux stations sont du coté A, il ne fait rien. De même pour des trames de M vers N ou X vers Z par exemple.

Par contre si une trame est émise de A à destination de Z, alors cette trame est enregistrée dans la file A puis une tentative d'émission de celle-ci sera effectuée coté B. La trame ne sera mémorisée dans la file A que si elle n'a pas collisionné. Elle pourra peut-être collisionner coté B lors de sa ré-émission de ce coté mais la collision ne sera pas détectée coté A.

Si le pont ne connaît pas la station destinatrice (elle n'a rien émis encore) il retransmet les trames vers cette destination sur le port opposé de celui sur lequel il reçoit ces trames.

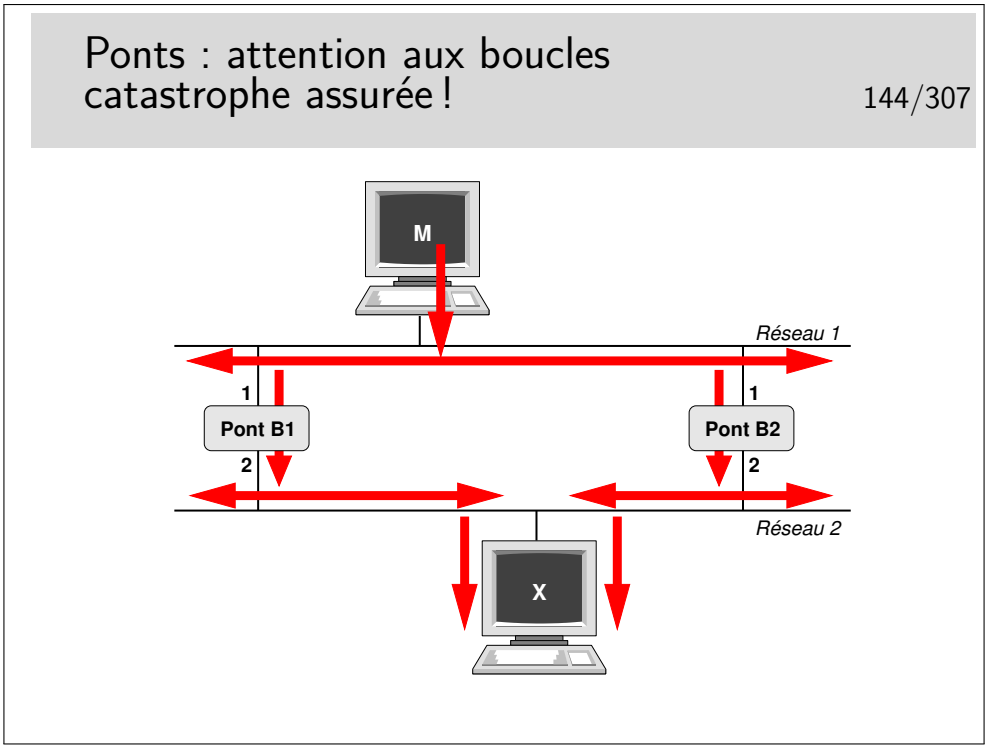
Avantages :

— le trafic est segmenté

— la connectivité totale est est maintenue

— le réseau est divisé en «domaines de collisions»

— le diamètre du réseau peut être doublé



Pour des raisons de fiabilité on peut être amené à doubler les ponts entre deux réseaux. Se pose alors le problème de boucles comme l'exprime l'exemple présenté ici.

La machine M émet une trame à destination de la machine X.

Premier point négatif : multiplication des copies de trames...

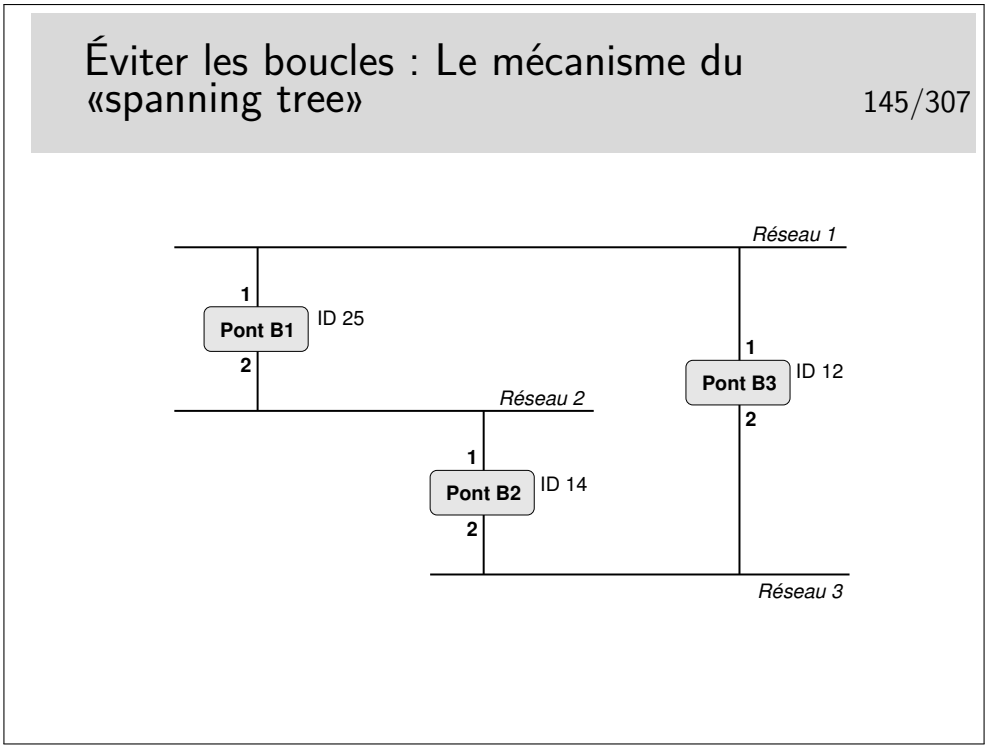
Le pont B1 enregistre la trame et la réémet sur son port 2. La trame arrive à destination. Le pont B2 fait de même. Une seconde copie arrive à destination.

Second point négatif : les trames font des boucles...

Si la station X n'a rien émis encore les ponts ne la connaissent pas. La station M émet une trame vers X. Les ponts ne connaissant pas X retransmettent cette trame comme précédemment. Il y a donc deux copies. Mais...

La copie faite par B1 arrive à destination et aussi en B2 port 2. B2 croit alors que la machine M a changé de réseau, elle est maintenant en bas. Comme il ne connaît pas X il recopie cette trame sur son port 1. Cette trame sera vue par B1 port 1 qui va la relayer vers son port 2 et va à nouveau arriver en X puis en B2 port 2 et ainsi de suite. Mais une trame tournera aussi dans le sens contraire car au début la copie faite par B2 arrive à destination.. etc... etc... Une seule trame émise suffit alors à saturer le réseau.

Pour éviter ce phénomène il faut transformer le réseau. De graphe avec des boucles il faut le transformer en arbre complet, en anglais en «spanning tree».

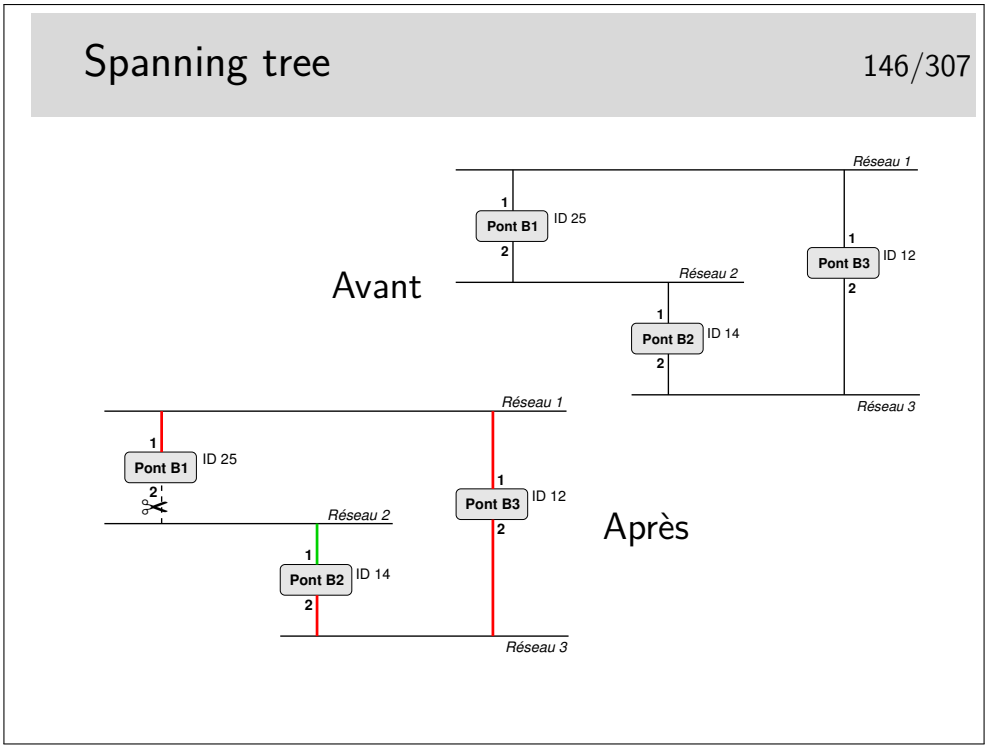


Les ponts (ou les commutateurs Ethernet) administrables peuvent mettre en œuvre cet algorithme. Sur chacun de ses ports, chaque pont annonce un identificateur qui lui est propre via une trame de type multicast (seuls les ponts sont configurés pour prendre en compte ce type de trame). Cet identificateur signifie en quelque sorte «c'est moi le pont racine». Tous les ponts commencent par annoncer qu'ils sont racine. Lorsqu'un pont reçoit un message comportant un identificateur «inférieur» au sien (on dira «meilleur»), il s'aperçoit qu'il n'est pas «racine» mais qu'un autre pont l'est plus certainement et que celui-ci est accessible via l'interface par laquelle est parvenue ce message «meilleur».

L'ensemble des ponts exécute le même algorithme, celui-ci converge finalement car un seul pont se trouve être le «meilleur», la racine (celui dont l'identificateur est le plus petit). Chaque pont sait par quelle interface atteindre le pont racine par le «plus court chemin», seule cette interface reste active pour le trafic de données, les autres interfaces permettant, elles aussi d'atteindre la racine, mais avec un chemin plus long sont inhibées pour le trafic de données. Les boucles sont ainsi évitées.

Les messages échangés s'appellent des BPDU (Bridge_PDU) et comportent les informations suivantes : `<id_pont_supposé_racine, coût_supposé, id_pont_émetteur, port_émission>`.

Les messages du protocole Spanning tree sont portés par des trames en adressage multicast. Leur adresse destination est (en standard) `01:80:c2:00:00:00`. Ce sont des trames de type 802.3. Les DSAP et SSAP de la couche LLC supérieure (voir plus loin) sont égaux à `0x42`. Les messages sont émis toutes les 2 secondes par défaut. Les identificateurs, les délais sont généralement paramétrables.



B1 émet : <25,0,25,1> sur son port 1 et <25,0,25,2> sur son port 2.

Il reçoit <12,0,12,1> par son port 1 et <14,0,14,1> par son port 2. Il constate qu'il n'est pas racine et que la racine doit être 12 (B2) accessible par son port 1. B1 considère que son port 1 est «root port».

Il émet alors : <12,1, 25,1> sur son port 1 et <12,1,25,2> sur son port 2.

B2 émet <14,0,14,1> sur son port 1 et <14,0,14,2> sur son port 2.

Il reçoit : <12,0,12,2> port 2 et <25,0,25,2> port 2 dans une première phase indiquant ainsi que le pont racine doit être accessible par son port 2 plutôt que par son port 1. B2 considère que son port 2 est «root port».

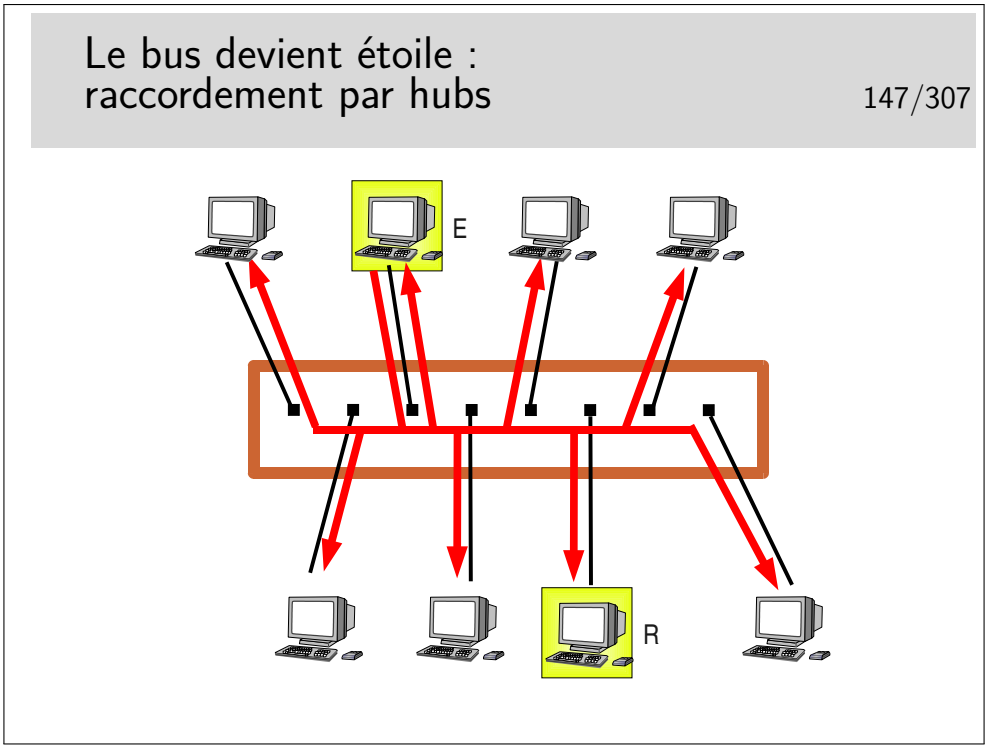
Il émet alors : <12,1,14,2> sur son port 2 et <12,1,14,1> sur son port 1.

B3 ne reçoit pas de «meilleure» configuration que celles qu'il émet. Il est donc la racine.

Comment vont alors se départager B1 (port 2) et B2 (port 1) ?

B1 reçoit port 2 : <12,1,14,1> ce qui est meilleur que ce qu'il émet sur ce même port <12,1,25,2>. Il inhibe son port 2 (pour le trafic de données, pas pour les messages de spanning tree). B2 reçoit sur son port 1 un message «moins bon» que ce qu'il émet sur ce même port, donc B2 port 1 reste actif.

Excellent tutorial animé à :
http://www.cisco.com/warp/public/473/spanning_tree1.swf



On prend le câble et les transceivers, on ramasse le tout dans un petit boîtier et le tour est joué. D'une topologie en bus nous passons à une topologie en étoile.

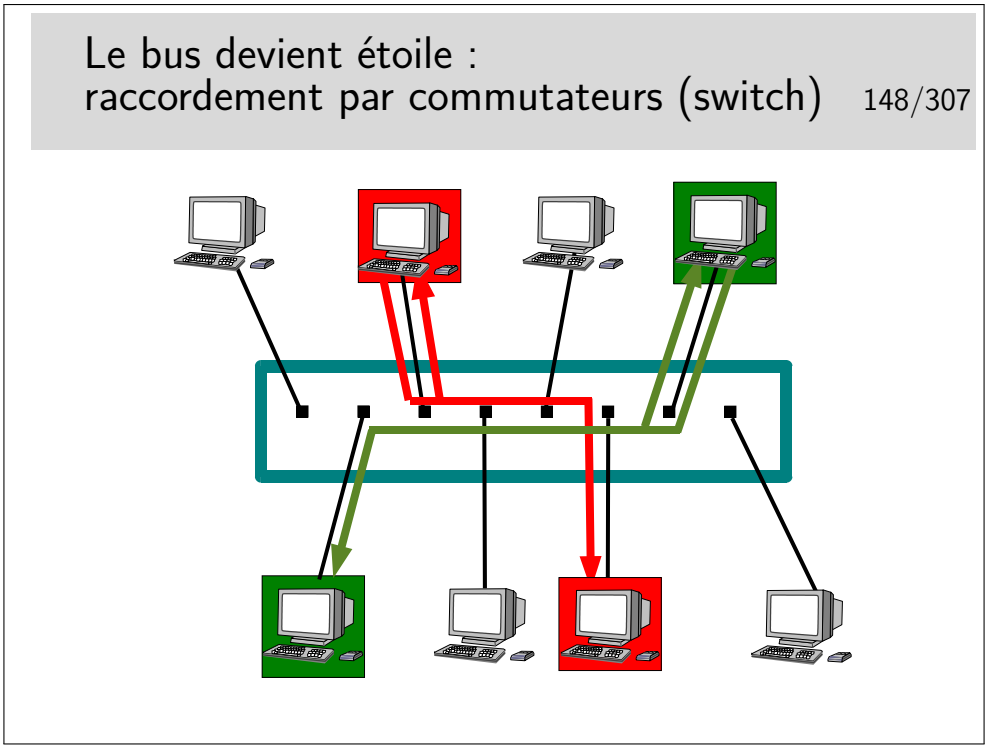
En fait ce n'est pas si simple car les raccordement changent, ce ne sont plus des transceivers mais des prises de type RJ45 et les câbles sont de type 10/100 BASE-T, à paire torsadée, de longueur max 100m.

Exemple de fonctionnement : La station E émet vers la station R. Toutes les stations reçoivent le signal, même E pour des raisons de détection de collision.

Un hub se comporte donc comme un bus.

Il est très facile d'espionner tout ce qui passe sur le hub à partir d'une machine raccordée sur un des ports.

Les hubs sont des **répéteurs**. Ce sont des organes de la **couche physique** (selon le modèle OSI), ils ne s'intéressent qu'au niveau bit.



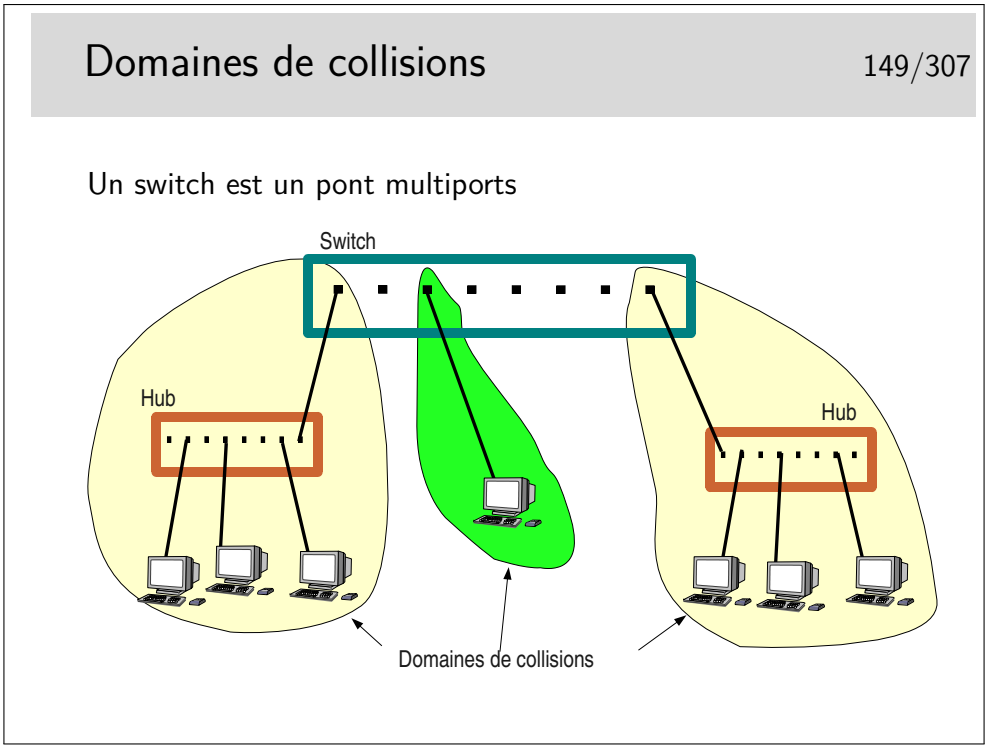
Le commutateur ressemble à un hub mais se comporte comme un pont. Il connaît les adresses (MAC) des machines qui sont raccordées sur chacun de ses ports (raccordées directement ou via d'autres d'autres commutateurs ou hubs). C'est un pont multiport.

Il ne recopie une trame que sur le port qui mène vers la destination.

Il peut faire ce travail pour plusieurs trames simultanément à condition qu'elles n'aillent pas vers le même port de sortie, auquel cas elles sont traitées les unes après les autres. Il se comporte donc comme s'il était capable d'établir des circuits entre deux ports pour de cours instants, d'on son nom de commutateur (switch en anglais).

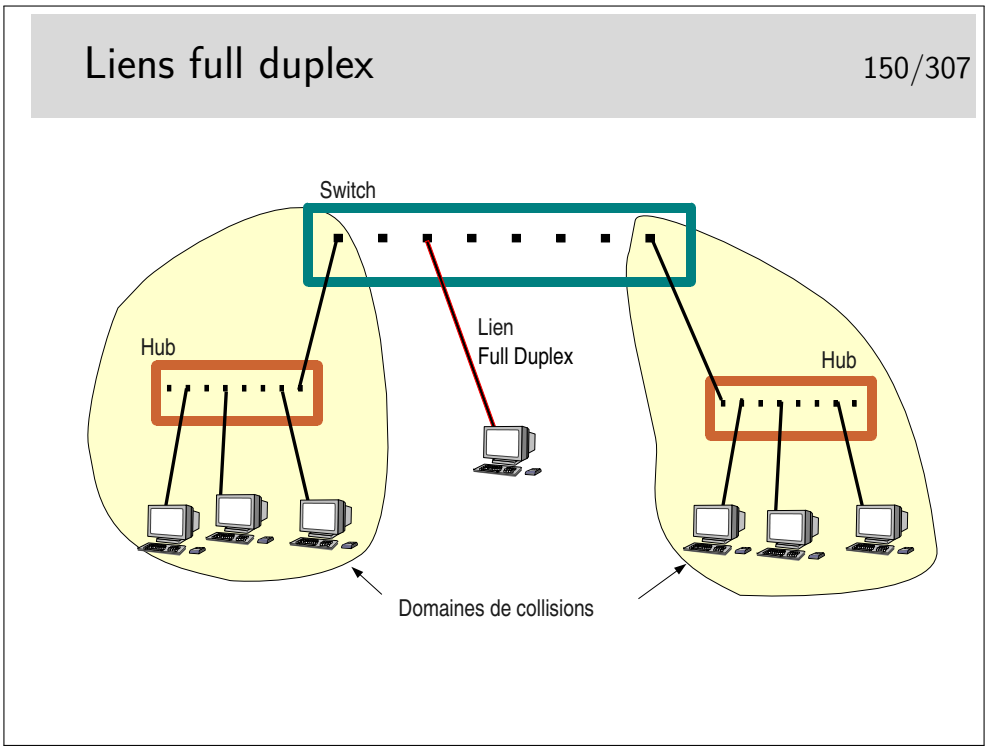
C'est une **commutation de niveau 2** (plus exactement de niveau MAC), à ne pas confondre avec la commutation de niveau 3 qu'on rencontre dans d'autres réseaux comme X25 ou ATM.

L'espionnage sur ce type d'appareil est plus difficile, sur un port donné on ne peut voir que les trames destinées à ce port ou les trames de *broadcast*, l'intérêt pour «l'espion» est donc très limité. Le «malfaisant» persévérant et cultivé en Réseaux peut cependant trouver des solutions...



Un switch se comporte comme un pont. Sur chacun de ses ports il enregistre les adresses sources des trames qu'il voit passer. Chaque port se comporte comme un port de pont. Chaque port délimite donc un domaine de collision.

On voit sur la figure ci-dessus qu'un de ces domaines ne comporte que deux machines (un port du switch et une machine seule), Voir le transparent suivant pour une utilisation plus efficace du lien en question.



Lorsqu'une machine terminale est reliée directement à un switch le domaine de col-

lision est restreint à la machine et au port du switch sur lequel elle est raccordée. Si on considère en plus que le lien physique de raccordement est un câble comportant un circuit différent pour chaque sens (deux fils par pour l'émission, deux autres fils pour la réception, ou deux fibres optiques), il devient alors intéressant d'inhiber le mécanisme de détection et contention de collision pour tirer partie pleinement de la paire émission et de la paire réception présente dans le câble de raccordement en les utilisant simultanément pour véhiculer des données utiles.

On peut ainsi passer de 10/100/1000 Mb/s à l'alternat à 10/100/1000 Mb/s dans chaque sens simultanément (10 ou 100 ou 1000 MB/s selon le matériel).

Attention, en général il faut gérer le full duplex, c'est à dire qu'il faut se connecter au switch pour l'administrer (sur son port console via un PC et l'application HyperTerminal ou via telnet en IP) et configurer les ports qu'on désire voir fonctionner dans ce mode. Il faut aussi vérifier sur la machine terminale qu'elle est bien en full duplex et, éventuellement, la forcer dans ce mode. Si le full duplex n'est pas positionné des deux cotés il en résultera un fonctionnement très ralenti, une des deux extrémités détectant alors des collisions qui n'en sont pas.

6.3 Les VLANs

Le concept de LAN virtuel ou VLAN | 152/307

- Un VLAN est construit à l'aide des commutateurs dont on restreint les possibilités de commutation à des groupes de ports, la commutation peut être totale entre les membres d'un groupe mais devient impossible entre les membres de groupes différents

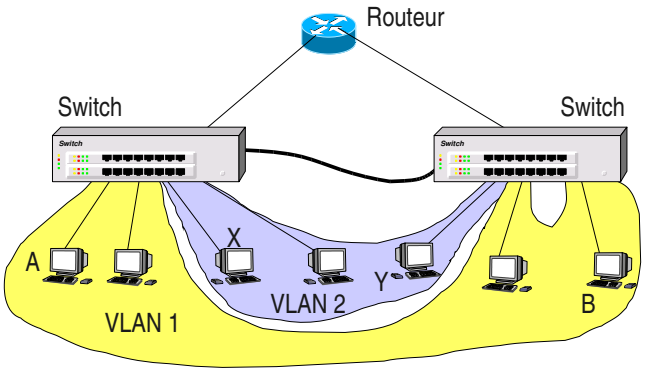
Un groupe définit un domaine de broadcast
domaine de broadcast ⇔ VLAN (plus petite commune définition)

- suivant les possibilités matérielles des switches, un VLAN peut être défini par port, par adresse MAC, par adresse IP (dans ce dernier cas, ce n'est toutefois pas assimilé à de la commutation niveau 3)

Le concept de LAN virtuel ou VLAN II 153/307

- un VLAN peut être réparti sur plusieurs commutateurs reliés entre eux
- routeur obligatoire entre VLANs, même s'il n'y a qu'un commutateur
- l'attribution d'un élément (port, adresse MAC, adresse IP) à un VLAN est réalisée par une opération de gestion
- l'interconnexion de switchs impose de marquer les trames sur les liens d'interconnexion afin de les associer à un VLAN. Exemple figure suivante : les trames échangées entre A et B ou entre X et Y doivent être différentiées sur le lien entre les deux switches. Solution : étiquetage des trames ; normes IEEE et solutions propriétaires

Les LANs virtuels : VLANs 154/307

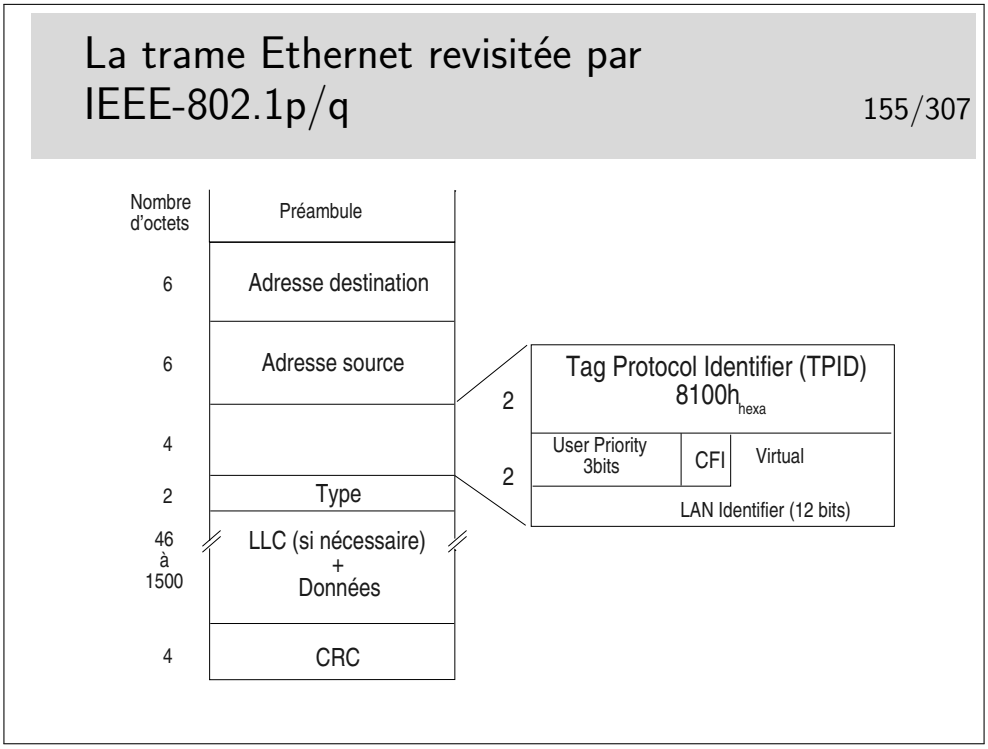


- Sur un commutateur ou plusieurs interconnectés
  - On subdivise l'espace d'interconnexion en sous espaces étanches
  - On construit ainsi une architecture logique sur une topologie physique
  - Standardisation : IEEE-802.1p et 1q

Imaginons que la machine A veuille émettre vers toutes les machines de son VLAN, elle doit émettre une trame de *broadcast* qui devra être relayée par le switch de gauche vers le switch de droite. Cependant, le switch de droite devra connaître l'identité du VLAN vers lequel diffuser la trame. Il ne faudra pas, par exemple, émettre vers la machine Y du VLAN2. Pour cela, sur le lien entre les switches, la trame devra être complémentée par un champ porteur de l' identité du VLAN.

Le transparent suivant indique comment est modifiée la trame ethernet de base.

Le lien d'interconnexion des commutateurs est parfois un «trunk». Selon les constructeurs et les types de switches, des ports spécifiques peuvent être réservés pour le «trunking».



Le champ TPID est tout simplement une valeur de champ Type spécifique. Le champ «User Priority» apporte une possibilité de privilégier certains flux.

Le bit CFI (Canonical Format Indicator), s'il est à 1, indique que le champ information comporte des indications de routage par la source.

Le champ «Virtual LAN Identifier» indique à quel VLAN appartient la trame.

Compléments du standard 802.1p/q 156/307

- GARP : Generic Attribute Registration Protocol
  - mécanisme de signalisation permettant aux stations de fournir des indications (par des valeurs d'attributs) affectant les paramètres de filtrage des switches.
  - 3 attributs déjà définis :
    - groupe d'adresses MAC, facilite les mécanismes du multicast,
    - mode de filtrage des ports,
    - VLAN
- GMRP : Garp Multicast Registration Protocol
  - mécanisme permettant aux stations terminales et aux switches de s'enregistrer (et se retirer) comme participants à un groupe multicast et de diffuser cette information à l'ensemble de switches du réseau,
  - les switches utilisent cette information comme paramètre de filtrage,
  - utilise GARP comme protocole support.

6.4 Évolutions

Les évolutions d'Ethernet 158/307

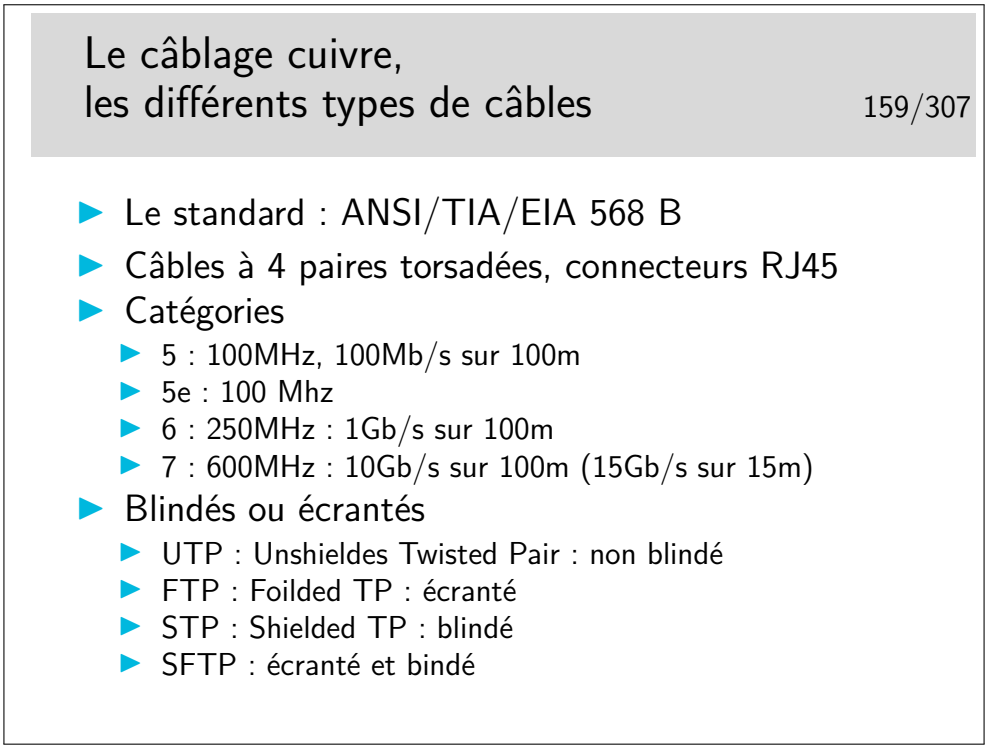
- Années 80 : le 10Mb/s, 802.3, en bus sur câbles coaxiaux
- Années 90 :
  - 10Mb/s sur paires torsadées, raccordement sur hub
  - 100Mb/s
- Fin des années 90 : le 1000Mb/s
- Années 2000 : la 10Gb/s arrive
  - Support physique revisité, utilisation du support Sonet (OC192)
  - Distances visées : 40Km et plus
- Interopérabilité totale ascendante

Sonet (Synchronous Optical Network) est un type de multiplexage mis au point par la société américaine Bellcore et standardisée par l'ITU-T sous l'appellation SDH (Synchronous Digital Hierarchy).

À l'origine, Sonet/SDH est destiné au transport des voies téléphoniques numériques.

Les concepts mis en œuvre permettent un multiplexage et un démultiplexage facilité par rapport aux techniques plus anciennes. C'est le support privilégié pour les réseaux de type ATM.

Les débits courants sont de 155Mb/s (Sonet : 0C3, SDH : STM1) sur cuivre (100m) ou fibre optique et 622Mb/s. Le débit de 1,2 Gb/s est possible. Un bond technologique est réalisé pour l'Ethernet 10Gb/s.



Un lien : `http://cablingdb.com/GlossaryPages/GlossaryC/CGlossary.asp`

6.5 Dénomination

Les dénominations des différentes versions d'Ethernet I 161/307

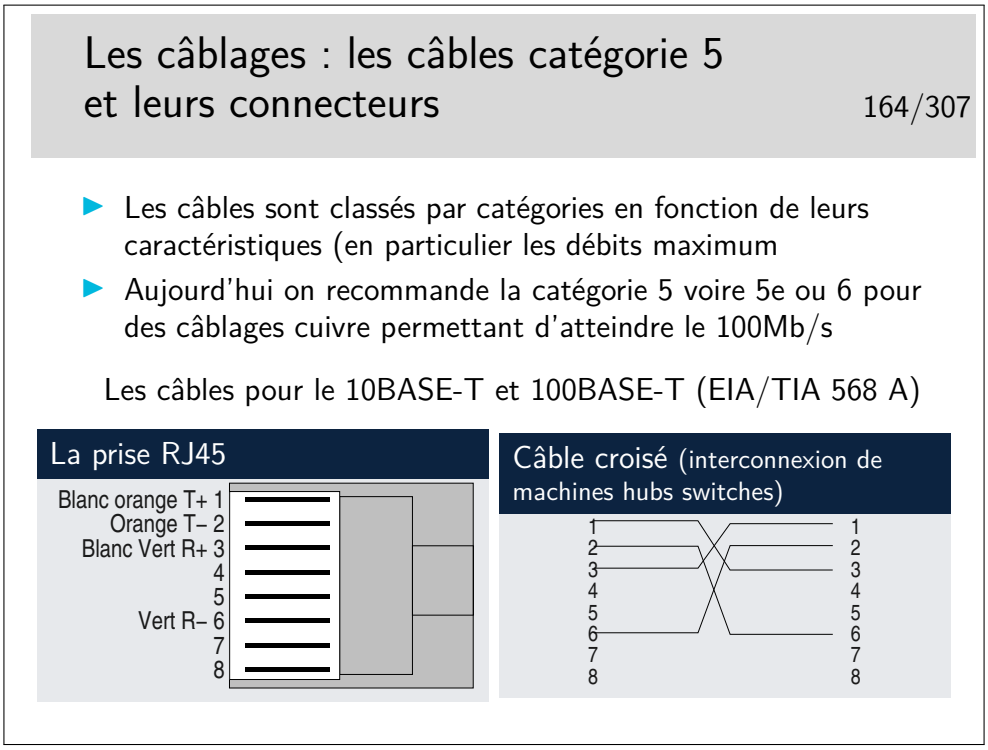
- Format : x BASE/BROAD y
  - x : le débit
  - BASE : bande de base, BROAD : large bande
  - y : indication sur la topologie (et/ou la longueur du câble)
- Bande de base :
  - Le 10 Mb/s
    - 10 BASE-5 : 10 Mb/s, topologie en bus constitué de segments de 500m
    - 10 BASE-2 : 10Mb/s, topologie en bus constitué de segments de 185m
    - 10 BASE-T : 10Mb/s, topologie en étoile, câbles en paires torsadées (T pour *Twisted pair*), longueur 100m

Les matériels à 10Mb/s tendent à disparaître aujourd'hui (2004). Les cartes interfaces sont toutes en 100-BASE-T, même en premier prix.

Les dénominations des différentes versions d'Ethernet II 162/307

- Le 100Mb/s
  - 100BASE-T4 : 4 paires utilisées, catégorie 3 à 5
  - 100BASE-TX : 2 paires torsadée, catégorie 5
  - 100BASE-FX : 2 fibre optique
- Le 1000Mb/s
  - 1000BASE-LX : fibre optique, grande (Long) longueur d'onde
  - 1000BASE-SX : fibre optique, courte (Short) longueur d'onde
  - 1000BASE-CX : paire torsadée, 25m max
  - 1000BASE-TX : 4 paires torsadées de catégorie 5

6.6 Câblage



Un document bien illustré sur le câblage... En italien, mais Mhz et nm et Km s'écrivent comme en français... `http://www.garr.it/ws4/pdf/Montessoro.pdf`

Un câble croisé permettra de raccorder :

— deux machines directement entre-elles

— deux hubs ou deux swiches entre-eux

— un hub et un switch

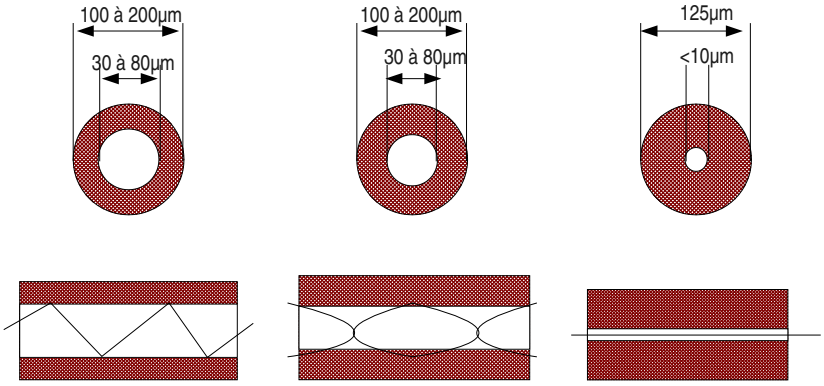
Les hubs et les switches peuvent être munis de ports directement croisés, on les repère généralement par la mention «Up Link». Un port peut aussi avoir deux prises RJ45, und roite et une croisée «up-link».

Un port peut être muni d'un petit commutateur appelé **MDI-MDIX** permettant de positionner le port en mode normal (MDI) ou en mode croisé (MDIX).

Et enfin, la fonction MDI-MDIX peut être à auto-détection : le switch détecte automatiquement s'il a à faire à un câble droit ou croisé.

Les câblages : la fibre optique 165/307

Multimode Gradient d'indice Monomode



La fibre et Ethernet gigabit 166/307

Transceiver	Fiber Diameter (microns)	Bandwidth (MHz*km)	Minimum Range (meters)
1000BASE-SX	MM 62.5 MM 62.5 MM 50 MM 50	160 200 400 500	2-220 2-275 2-500 2-550
1000BASE-LX	MM 62.5 MM 50 MM 50 SM 9	500 400 500 NA	2-550 2-550 2-550 2-5000

Transceiver	Fiber Diameter (microns)	Wavelength (nm)	Minimum Range (meters)
1000BASE-LH (Extended distance)	SM 9	1310	1 km - 49 km
1000BASE-LH (Extended distance)	SM 9	1550	50 km - 100 km

- 1000 BASE-SX : (S pour Short), $\lambda = 850nm$, codage en ligne 8B/10B
- 1000 BASE-LX : (L pour long), $1270nm < \lambda < 1355nm$, codage en ligne 8B/10B
- 1000 BASE-LH (Long Haul), fibre mono mode (SM : Single Mode), 9μ

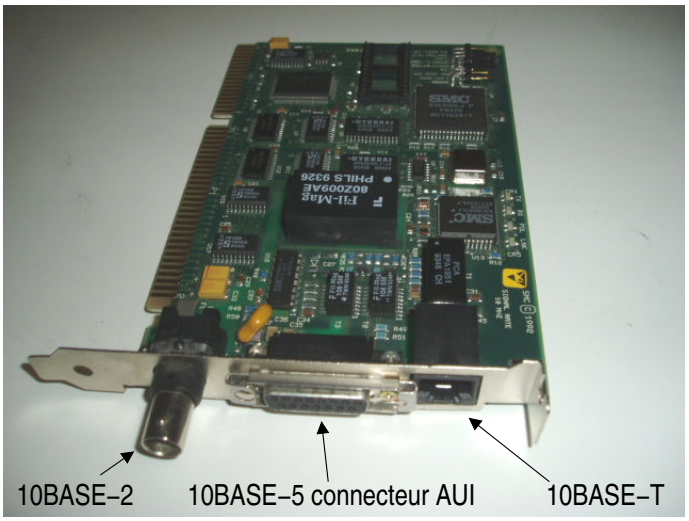
Interopérabilité des débits 167/307

- Les hubs et les switches acceptent aujourd'hui les deux débits de base 10Mb/s et 100Mb/s
- Les switches comportant des ports à 1Gb/s se généralisent
  - Un mécanisme d'autonégociation entre les extrémités dce la liaison permettent de détecter le débit maximal accepté (10 ou 100, 1000)
  - L'autonégociation permet aussi de détecter la possibilité du full-duplex et de contrôle de flux

La fonction contrôle de flux 168/307

- Disponible sur certains commutateurs
- Trames «pause»
  - Adresse multicast 01-80-c2-00-00-01
  - Type 0x8808
  - Information : sur deux octets (<65536), contient le nombre de «pause_times» (périodes de 512 bits)
  - Indiquent la durée pendant laquelle l'extrémité ne désire pas recevoir de nouvelle trame

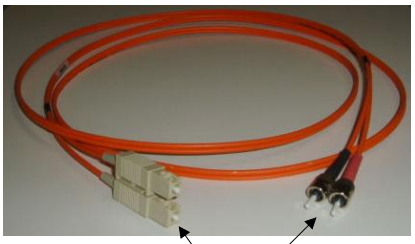
Une carte interface «multimedium» 169/307



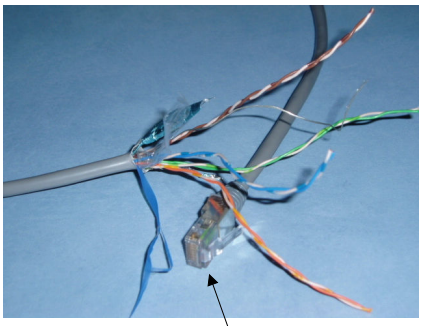
Technologie des débuts 90.
Aujourd'hui on ne trouve plus que du 10/100/1000 BASE-T, à quelques Euros...
Et en plus il y a beaucoup moins de composants...

Câbles et connecteurs – Cuivre et fibre 170/307

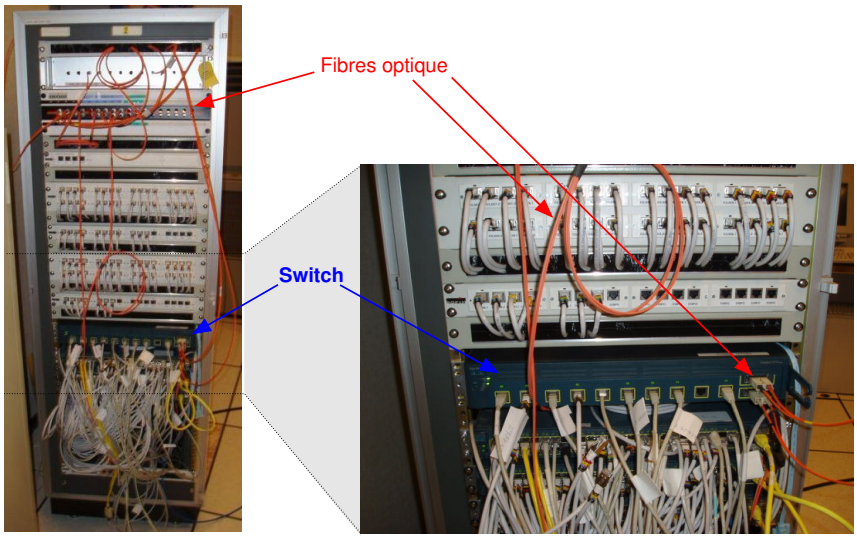
Fibre optique



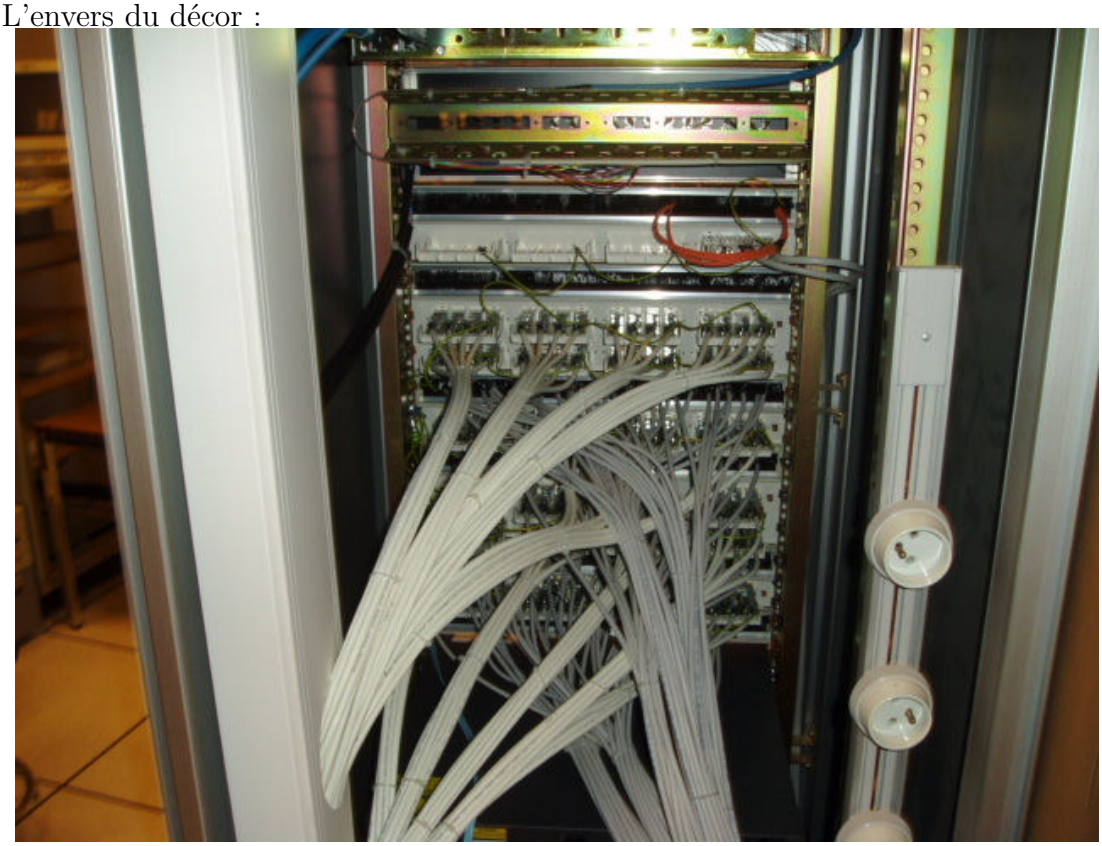
Paires torsadées



Une baie de brassage de câbles 171/307



L'envers du décor :



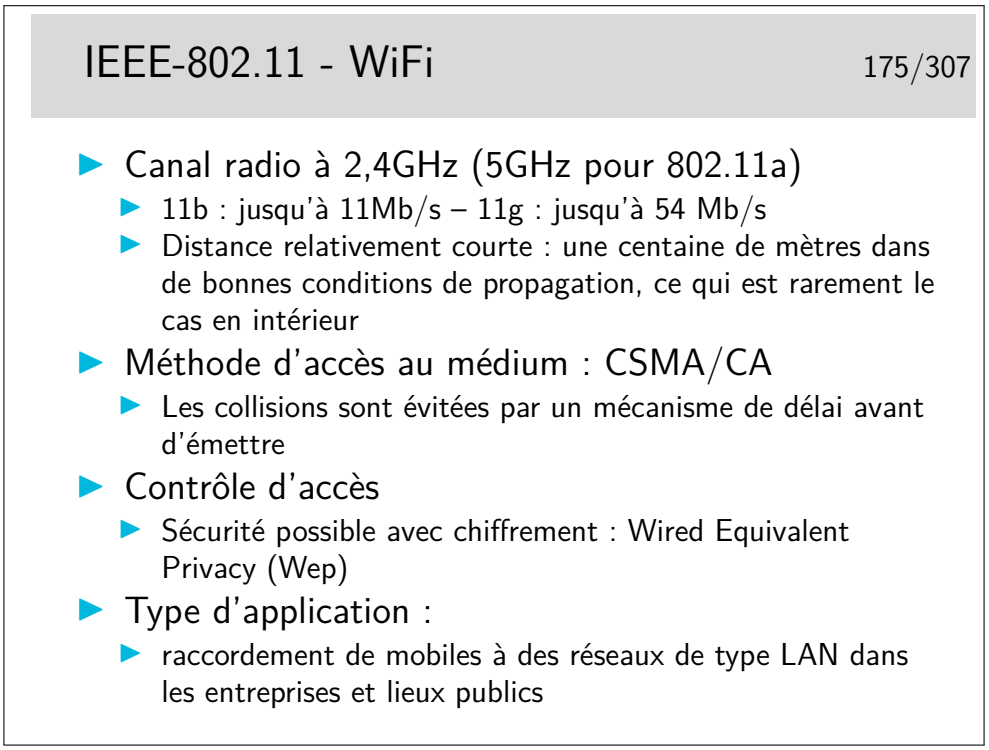
Administration des commutateurs 172/307

- Les commutateurs de bas de gamme ne s'administrent pas
- On peut se connecter sur les autres
  - Par port série dédié (et hyper-terminal sous Windows ou minicom sous linux)
  - Par telnet sur une adresse IP configurée dans l'appareil
  - Par page Web sur une adresse IP configurée dans l'appareil
  - Par SNMP
- Les actions d'administration peuvent être
  - Mettre en place ou inhiber la fonction spanning-tree et gérer cette fonction (priorités du switch, QoS, ...)
  - Mettre en place et gérer des VLANs
  - Monitorer des ports
  - Surveiller les adresses MACs enregistrées
  - Surveiller le trafic

La fonction «monitoring» consiste à rediriger le trafic normal d'un port vers un port dédié à la fonction et sur lequel on place une machine munie d'un analyseur de flux. Toutes les trames entrantes et sortantes du port surveillé sont recopiées sur le port de monitoring.

7 Les autres techniques autour des LANs

7.1 Le sans fil IEEE-802.11 (WiFi : Wireless Fidelity)



Un tutorial correct :
`http://www.intelligraphics.com/articles/80211.article.html`
Un lien sur la sécurité en 802.11 :
`http://www.iss.net/wireless/WLAN.FAQ.php`
Autre lien intéressant :
`http://wireless.ictp.trieste.it/school.2002/lectures/ermanno/HTML/802.11_Architecture.pdf#search='802%2011%20ssid'`

7.2 Les courants porteurs

Technique Courant Porteur 177/307

- CPL : courant porteur en ligne
  - Le réseau d'alimentation électrique est le medium
  - Interfaçage simple avec Ethernet ou USB : le câble catégorie 5/RJ45 est relié à une prise spéciale enfich une prise de courant
  - Débits annoncés : 5-20 Mb/s
    - Des produits arrivent annonçant 200Mb/s (théorique) !
  - Modulation en ligne OFDM (certains utilisent CDMA)
  - Solutions propriétaires, pas de compatibilité entre produits
    - Pourtant un standard : Home Plug
    - Des travaux à l'ETSI
  - Applications : «émulation Ethernet», raccordement de quartiers résidentiels, d'entreprises, etc...

Un lien intéressant `http://vlan.org/breve123.html`

7.3 Autres

Autres techniques Réseau 179/307

- Les PANs : Personnal Area Network
  - Très courtes distances
  - Typiquement : Bluetooth ZigBee
    - Interconnexion de petits portables (téléphones, PDAs, etc.), domotique (capteurs, actionneurs)
  - Standardisation en cours : IEEE-802.15
- Les bus spécialisés
  - USB
  - IEEE-1394 (FireWire ou iLink selon les constructeurs)
    - Raccordement d'appareils multimédia (caméra vidéo numériques, audio)
    - Canaux synchrones et asynchrones
    - 800Mb/s
    - Distance : 100m annoncée entre nœuds et hubs (4,5m en standard)
    - En standard sur les PCs aujourd'hui

Les canaux synchrones de IEEE-1394 (FireWire/iLink) sont adaptés à la transmission de flux sons et images.

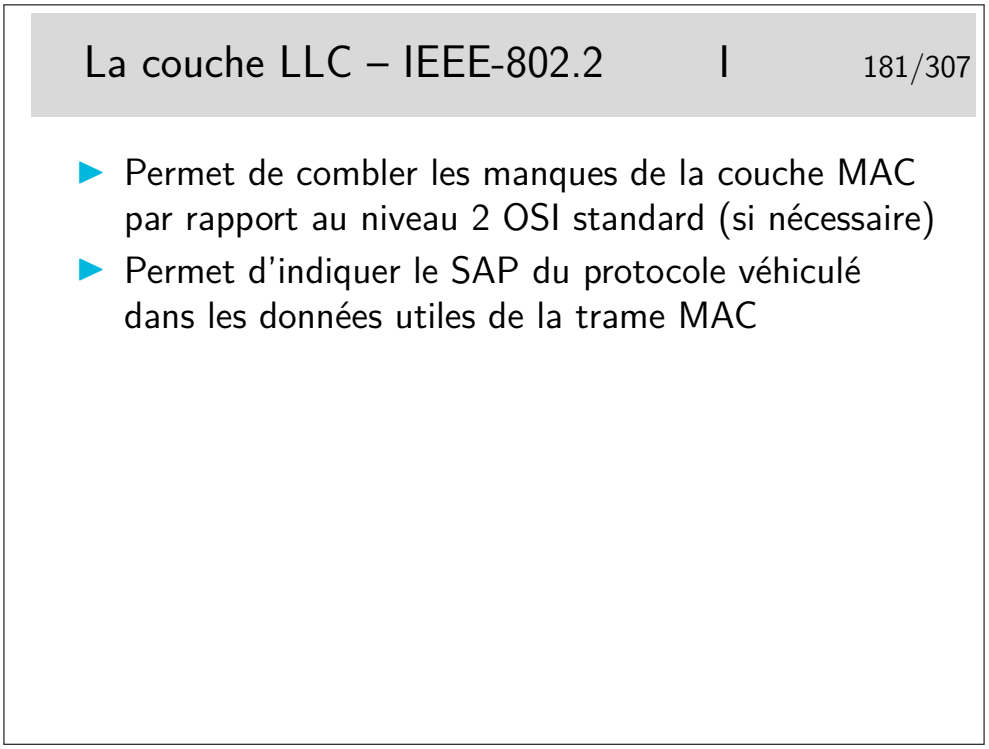
Ces réseaux utilisent des protocoles spécifiques pour les couches supérieures, par exemple AMDTP (Audio and Music Data Transmission Protocol. IEC61883-6) pour IEEE-1394. IP n'est pas implémenté en standard sur ces réseaux.

Une émulation Ethernet existe pour IEEE-1394 en utilisant les canaux asynchrone.

FireWire : IEEE-1394 chez Apple.

ilink chez Intel

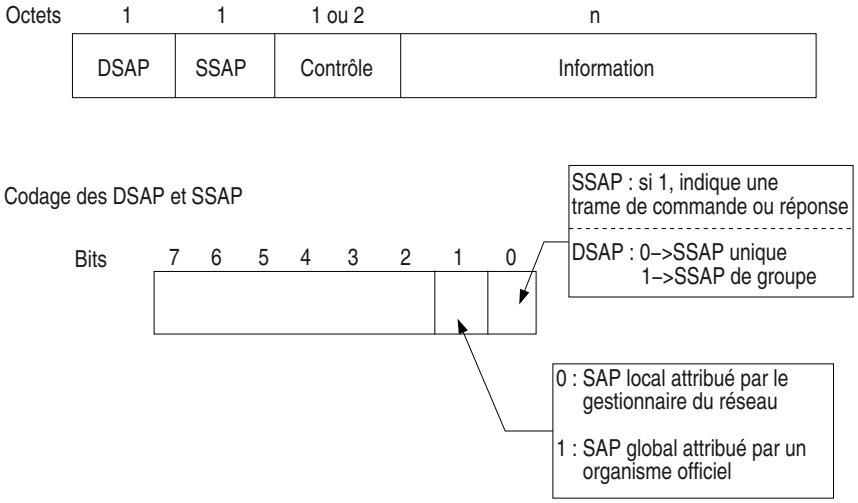
7.4 La couche LLC



La couche LLC – IEEE-802.2 II 182/307

- Trois types LLC
  - LLC type 1 : mode datagramme, sert uniquement à véhiculer le SAP des données utiles
  - LLC type 2 : mode connecté, en plus de la fonctionnalité du type 1 on assure des contrôles identiques au HDLC LAPB (avec numérotation des trames modulo 128 : champ contrôle sur 2 octets), sert à véhiculer des paquets X25 par exemple
  - LLC type 3 : mode datagramme avec acquittement. Prévu pour les réseaux industriels

Format du PDU LLC 183/307



— DSAP : Destination Service Access Point

— SSAP : Source Service Access Point (i.e. *comment* interpréter le champ contrôle)

Et donc ... il reste combien de SAP possibles, donc de protocoles identifiables ? 64 ! C'est bien peu et c'est un problème...

LLC : quelques SAP 184/307

- 0x06 : IP
- 0x42 : Spanning Tree
- 0x7E : X25
- 0xE0 : Novell IPX
- 0xAA : SNAP (voir plus loin)

La notation 0x... est celle du langage C ou Java pour les nombres en hexadécimal.
Le SAP 0x7E ne vous rappelle rien concernant X25 et surtout sa couche 2 standard ?

LLC : le champ contrôle 185/307

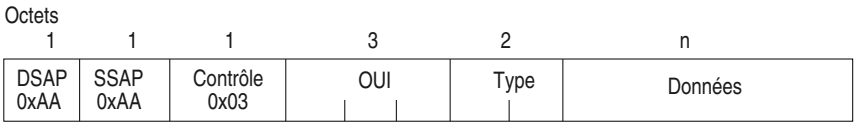
- LLC type 1
  - Le champ contrôle est codé sur un octet et vaut 0x03
- LLC type 2
  - Format identique au champ contrôle (ou commande) de HDLC LAPB
  - Trames numérotées modulo 128
    - Longueur du champ : 2 octets pour les rames I, RR, RNR et REJ
    - La trame d'établissement de la connexion s'appelle SABME (Set Asynchronous Balanced Mode *Extended*)

LLC SNAP 186/307

- SNAP : Sub Network Access Protocol
- Les champs DSAP et SSAP standards sont trop courts
  - Le champ type de la trame Ethernet pur est très bien...
  - Si on le réutilisait... En le plaçant après le champ contrôle ?
    - Oui mais sa longueur n'est que de deux octets !
    - Le tout ferait donc 5 octets (Contrôle sur 1 octet), ce n'est pas optimum pour une architecture 32 bits (l'architecture de la plupart de nos machines encore pour l'instant)
    - Et si on complétait à 8 octets en rajoutant l'OUI du concepteur du protocole ?

SNAP : même traduit un français, ça ne veut rien dire (humour (?!!!))
OUI : Organizational Unit Identifier

LLC SNAP : le format 187/307



- DSAP = SSAP = 0xAA
- Contrôle = 0x03 (trame UI : Unumbered Information)
- OUI : code attribué par l'IEEE à la société créatrice du protocole. Identique aux 3 octets de début des adresses Ethernet. Souvent à 0
- Type : identifie le protocole : identique au champ Type de la trame Ethernet

Mais non !... Ce n'est pas si compliqué !...

Et en plus tout peut être justifié... Il y a une bonne raison pour qu'il en soit ainsi. Simplement il faut chercher cette raison parfois profondément...

Exemples de trames LLC 188/307

```
 0: 0900 07ff ffff 0040 9c00 0294 0022 aaaa .......@....."..
16: 0308 0007 809b 001a 0000 0000 03e8 ffe1 ................
32: 0101 0103 e808 e103 e880 03e8 8201 9400 ................


 0: 0180 c200 0000 0000 1d07 2c63 0026 4242 ..........,c.&BB
16: 0300 0000 8000 0000 0000 0000 0000 0000 ................
32: 0000 0000 0000 00c0 0f00 0050 0000 0000 ...........P....
48: 0000 0000 ....
```

Décodons... Et répondons aux questions :

— pourquoi peut-on dire que ces trames sont de type multicast ?

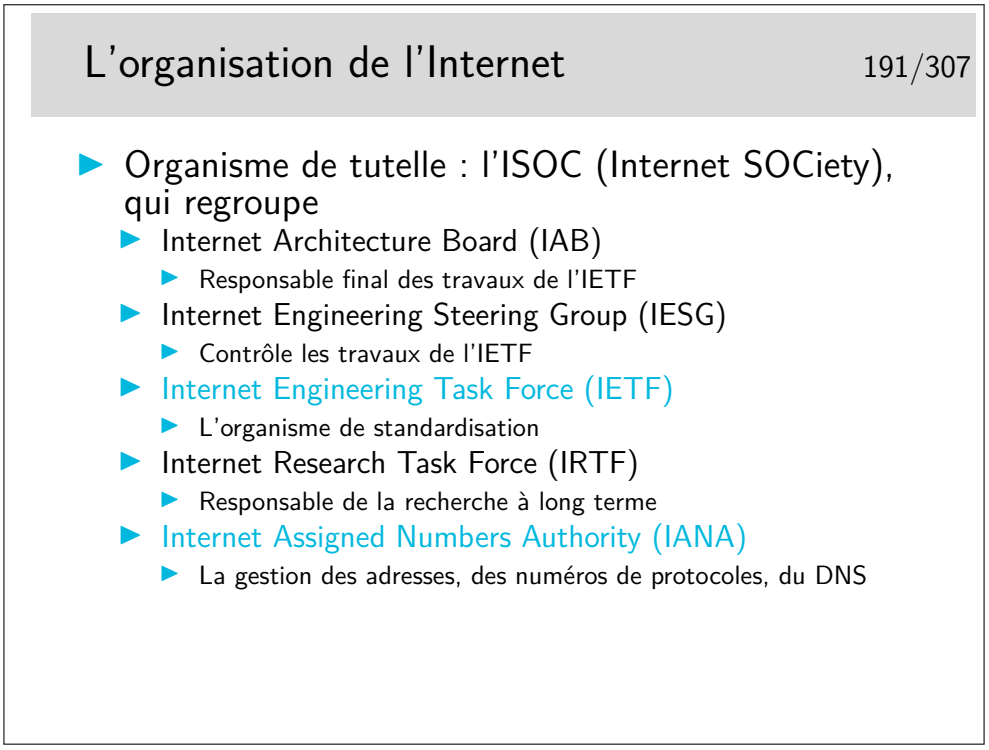
— que nous enseignent les adresses destination utilisées dans ces trames ?

— pourquoi peut-on dire qu'elles sont au format 802.3 et non Ethernet pur ?

— quels sont les protocoles véhiculés ?

Troisième partie
Le protocole IP et les protocoles associés

8 Standardisation, modélisation



Il existe aussi l'ICANN (The Internet Corporation for Assigned Names and Numbers) société de droit privé destinée à remplacer IANA. L'ICANN a été fondée en 1998 et on a parfois du mal à faire la distinction entre ses responsabilité et celles de IANA qui continue d'exister.

Voir un article de synthèse sur ce lien : `http://www.renater.fr/Projets/ICANN/`

La standardisation Internet 192/307

- Les membres des groupes de l'IETF travaillent par échange de courriers électroniques, sur des documents appelés *drafts*, dont la validité est de 6 mois
- Périodiquement ils se rencontrent dans des «meetings» pour valider des choix techniques
- Un *draft* peut évoluer vers une nouvelle version, valable 6 mois de plus
- Lorsque que le groupe de travail arrive à un consensus, le *draft* est promu au rang de RFC (Request For Comment)
  - Les RFCs sont les «normes» Internet, l'équivalent des recommandations ISO et ITU-T
  - Mais pas seulement...
- RFC et drafts sont en publications gratuite sur différents sites ftp et web (`www.ietf.org`)

Les RFC 193/307

- Format standard : ASCII pur et dur (voir rfc 2223)
- Plusieurs types forts
  - Proposed standard : draft présentant un fort consensus
  - Draft standard : pour un protocole dans cet état de standardisation, il existe au moins deux versions interopérantes
  - Standard : LE document final (mais pas obligatoirement figé pour l'éternité)
- Autres types
  - Expérimental : protocole encore en développement
  - Informatif : comme son nom l'indique
  - Historique
  - Best Current Practice (BCP)
  - Les rfc du premier avril... rfc1084, 1149, etc...

Les RFC sont en ASCII 7bits pur... Quelques activistes à l'IETF osent militer pour y introduire un peu de HTML (au moins pour le renvoie vers les sections ou la table des matières) : `http://www.catb.org/~esr/rfcs-in-html/index.html`

Les RFCs sont classés dans leur ordre de parution, il n'y a pas de classement thématique (hors les std).

Parmi les RFCs du premier avril citons par exemple celui spécifiant le protocole permettant d'envoyer des messages subliminaux (rfc1097), celui spécifiant comment utiliser

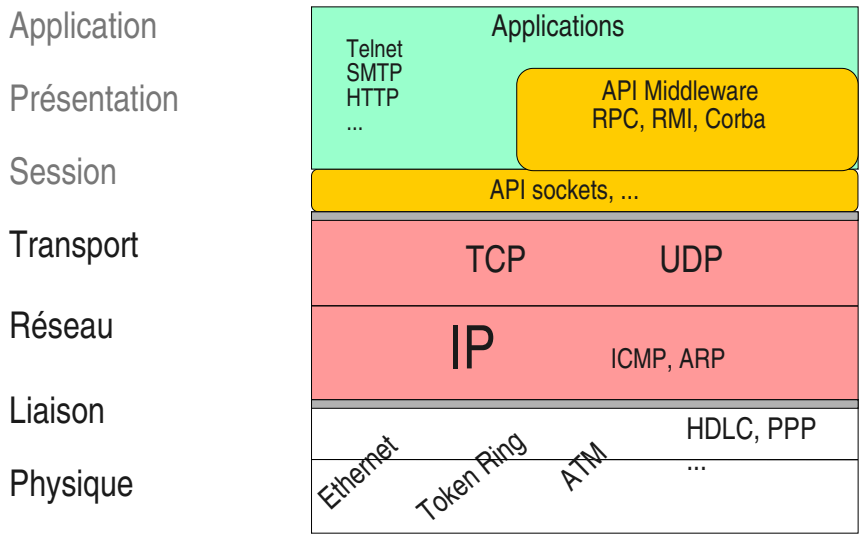
IP sur une couche liaison de type «pigeons voyageurs» ou plus exactement des «avian carriers» (rfc1149) le MTU des messages étant proportionnel à la longueur de la patte de l'oiseau...

Mais tous les RFC «premier avril» ne sont pas des plaisanteries... Le 777 est par exemple celui qui spécifie ICMP (Internet Message Control Protocol), et ce protocole est loin d'être humoristique...

Les classifications des RFCs 194/307

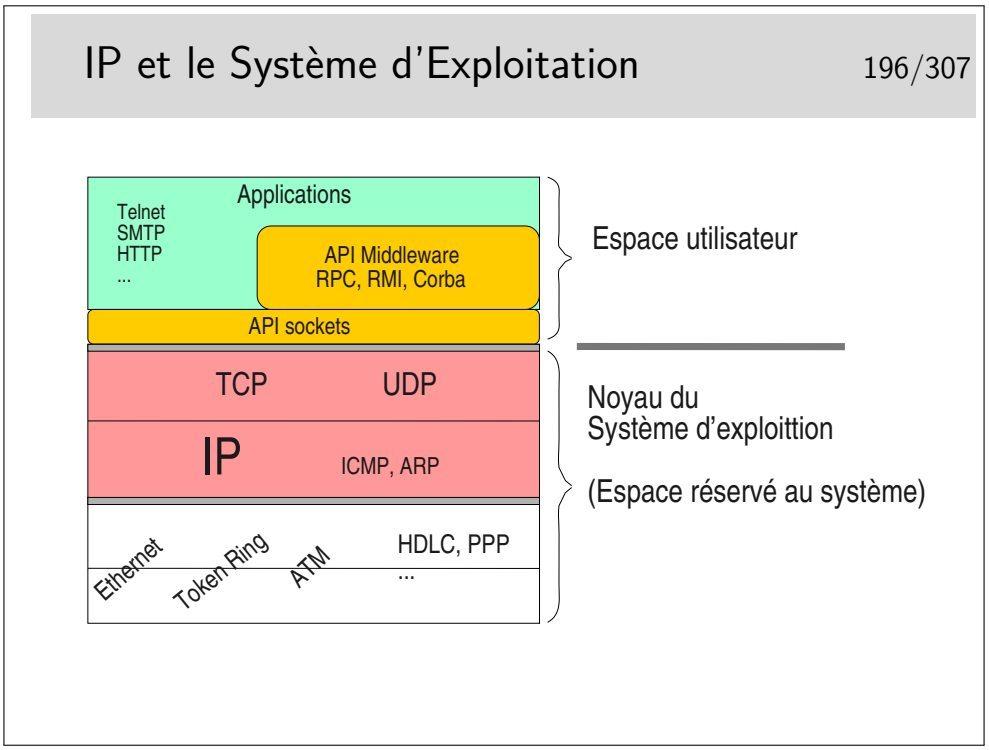
- Certains RFCs standards sont classées sous l'appellation std, le std 5 par exemple correspond au rfc791 définissant IP
- Certains sont classées sous l'appellation FYI : For Your Information
- Quelques RFCs :
  - 791 : IP (std5)
  - 793 : TCP (std7)
  - 763 : UDP (std6)

IP et le modèle OSI 195/307



Clairement, IP est au niveau 3 et TCP/UDP au niveau 4.

Quand bien même ces protocoles ne sont pas conformes à ceux qui ont été spécifiés pour le modèle OSI, on peut leur trouver ces places dans le modèle.



Les couches protocolaires jusqu'au niveau 4 sont inclues dans le noyau. Les applications (les programmes des utilisateurs) y accèdent via des interfaces qui les masquent.

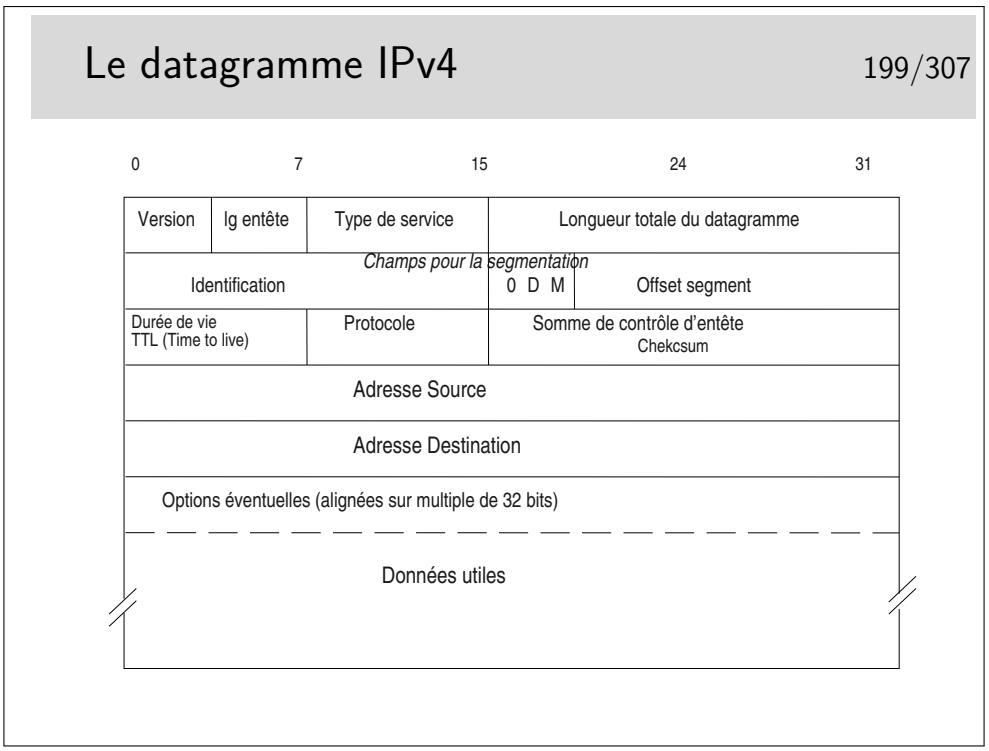
Ces interfaces sont constituées par des bibliothèques de fonctions formant des APIs utilisables par les programmeurs d'applications. La plus répandue est la bibliothèque socket tout à fait bien adaptée aux protocoles Internet (TCP, UDP, IP). Les couches protocolaires sont vues comme des fichiers qu'on écrit ou qu'on lit. Le transfert de données est aisé puisqu'il se fait comme avec des fichiers. Le portage est aisé vers le monde Windows qui implémente aussi cette interface programmatique issue des Unix de Berkeley (les sockets BSD).

Un autre type d'API est lui aussi très répandu puisqu'il est utilisé pour NFS et NIS. Au départ spécifié par SUN dans les années 80 il est depuis universel sous Unix. Il s'agit des Remote Procedure Calls ou RPC. La couche RPC offre une abstraction du réseau et permet d'appeler des fonctions en local alors qu'elles s'exécutent à distance.

Le concept a été poussé encore plus loin et adapté aux langages orientés objet comme C++ avec CORBA (Common Object Broker Architecture) qui permet d'appeler des méthodes sur des objets situés sur des machines distantes. Cette interface n'est cependant pas standard (du point de vue API) et plusieurs implémentations existent, la plupart commerciales et d'autre libres comme ORBit qui est utilisé par GNOME sous Linux.

9 Le datagramme IP

9.1 Le datagramme IPv4



Longueur d'entête : en nombre de mots de 32 bits ; si supérieur à 5, indique la présence d'options (40 octets max d'options : 10 mots max de 4 octets)

Type de service : peu utilisé par le passé, permet aujourd'hui, entre autres, de coder le DSCP (DiffServ Code Point). Voir pages suivantes.

Champs pour la segmentation : on recommande d'éviter la segmentation avec TCP. Elle n'existe plus que pour UDP si les unités de données fournies ont une taille supérieure au MTU de l'interface

bit D : *Don't fragment* : interdit la fragmentation du segment s'il est à 1

bit M : *More* : indique la présence de fragments complémentaires, si le segment d'origine a été fragmenté.

Si le champ Offset est à 0 ainsi que le bit M, le segment est celui d'origine

TTL : durée de vie ; décrémenté de 1 par chaque routeur. Si le résulat passe à 0, le datagramme est jeté et un message ICMP est envoyé à la machine émettrice

Protocole : indique le protocole supérieur véhiculé (TCP, UDP, ICMP voire même IP en cas d'encapsulation IP dans IP piour tunnelling)

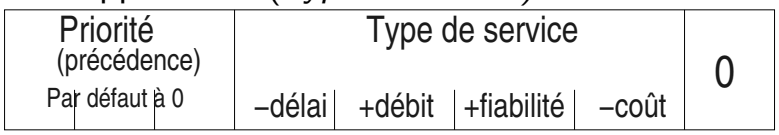
Somme de contrôle d'entête : contient la somme des mots de 16 bits constituant l'entête. Permet à l'arrivée de vérifier l'intégrité de celle-ci. Si une erreur est détectée le datagramme est jeté sans autre forme d'action. On ne peut même pas envoyer un message d'erreur à l'émetteur car l'adresse source qui indique ce dernier peut être corrompue. Comme l'indique le nom de ce champ le contrôle d'intégrité n'est fait que sur l'entête.

Adresses : source et destination, sur 32 bits ; information fondamentale, au moins en ce qui concerne la destination car elle sert au routage.

Le champ Type de Service

200/307

- Encore appelé ToS (*Type of Service) rfc1349*



- Utilisation non généralisée
  - Précédence utilisable pour marquer des flux dans les nœuds du réseau (routeurs)
  - Les bits Type de Service peuvent être positionnés par les applications terminales (API socket sous Windows et linux)
  - Sous Linux ils peuvent être positionnés via la commande iptables selon divers critères

111 - Network Control (protocole de type HELLO)

110 - Internetwork Control (protocoles de routage)

101 - CRITIC/ECP

100 - Flash Override

011 - Flash

010 - Immediate

001 - Priority

000 – Routine (priorité la plus faible)

RFC 791 et 795

Le champ Type de Service en version DiffServ

201/307

- Le champ DSCP sert à coder le *PHB (Per Hop Behavior), paramètre fondamental de DiffServ (rfc 2474)*
- Exemples de PHBs :
  - Expedited forwarding (pertes faibles, latence faible, gigue faible, bande passante assurée)
  - Assured forwarding (groupe de PHBs)
  - Best effort
  - Network control

DiffServ signifie Différentiation de Service (en anglais aussi). C'est un mécanisme relativement récent permettant de marquer les paquets afin qu'ils soient traités de manière différentiée dans les routeurs. C'est un mécanisme de qualité de service.

Les options IPv4

202/307

- Format :
- Les options :
  - LSR : *Loose Source Route* : permet d'indiquer la route
  - SSR : *Strict Source Route*, comme précédemment en plus rigoureux
  - RR : *Record Route* : les routeurs traversés rajoutent leur adresse
  - RTALT : *Router Alert*, permet de passer le paquets aux couches hautes des routeurs traversés
  - etc.

L'option LSR permet d'indiquer la route à suivre mais si un routeur intermédiaire ne sait pas comment utiliser une des indications contenue dans l'option, il peut alors utiliser sa table de routage normale.

L'option SSR est plus stricte car le paquet est rejeté si un routeur ne sait pas l'orienter

vers une direction indiquée.

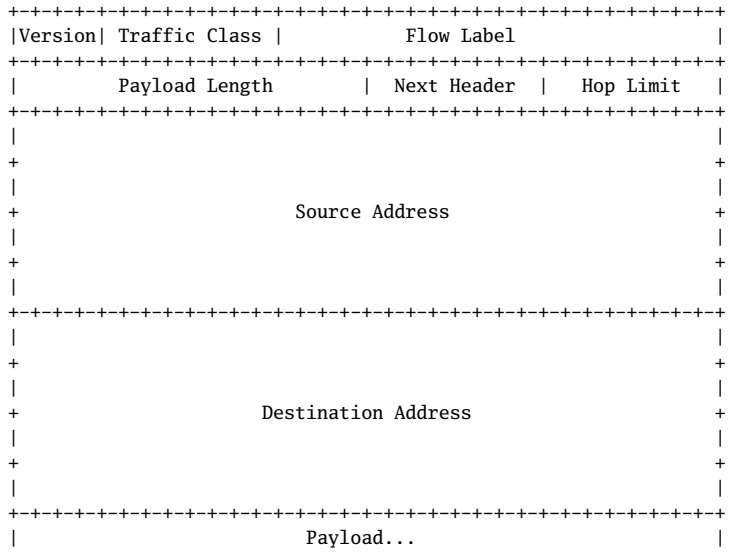
Il y a encore l'option Traceroute, obsolète car dangereuse, EEOL et NOP pour indiquer la fin de la liste et du bourrage d'alignement sur 32 bits.

L'option RTALT est utilisée par IGMP et par RSVP, ces protocoles nécessitent des traitements particuliers dans les routeurs. Ces traitements sont effectués par des entités applicatives des routeurs. Ces derniers ne doivent donc pas effectuer le routage rapide des paquets munis de cette options mais ils doivent les diriger vers les entités applicatives.

`http://www.iana.org/assignments/ip-parameters`

9.2 Le datagramme IPv6

Le datagramme IPv6 – RFC 8200 | 204/307

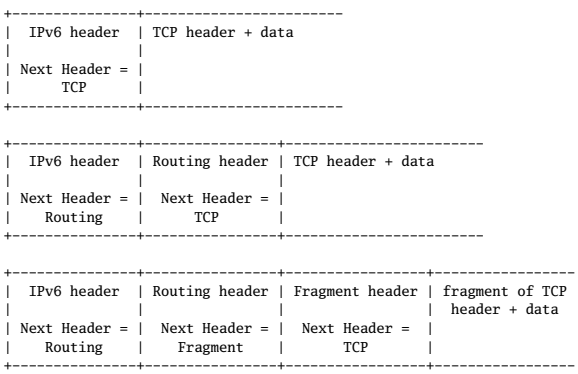


Le datagramme IPv6 II 205/307

- Version : 6 (!)
- Traffic Class : typiquement Differentiated Services et Explicit Congestion Notification, comme le TOS IPv4
- Flow Label : pour tagger les paquets d'un même flux ; visiblement peu utilisé...
- Payload Length : comme son nom l'indique... (Note : dans IPv4 on a la taille *totale* du datagramme)
- Next Header : remplace le champ Protocole IPv4 et ajoute un mécanisme un peu novateur pour gérer les options (cf. slides suivants)
- Hop Limit : l'équivalent du champ TTL IPv4 (Note : contrairement à IPv4 qui compte en secondes, ici c'est bien en nombre de sauts)
- Source & Destination Addresses : sur 128 bits (Ouf !)

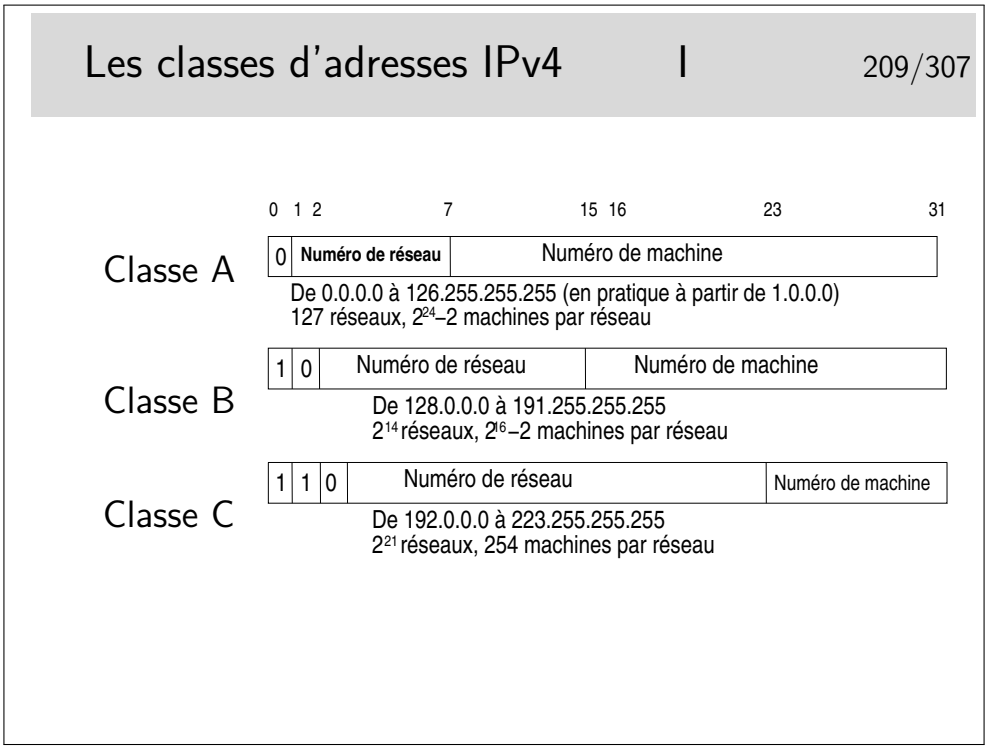
Next Header 206/307

- Les headers s'enchaînent dans le payload
- ...les options IPv6 diverses, puis le protocole transporté effectivement (TCP, UDP, ...)



10 Adressage

10.1 Les adresses IPv4

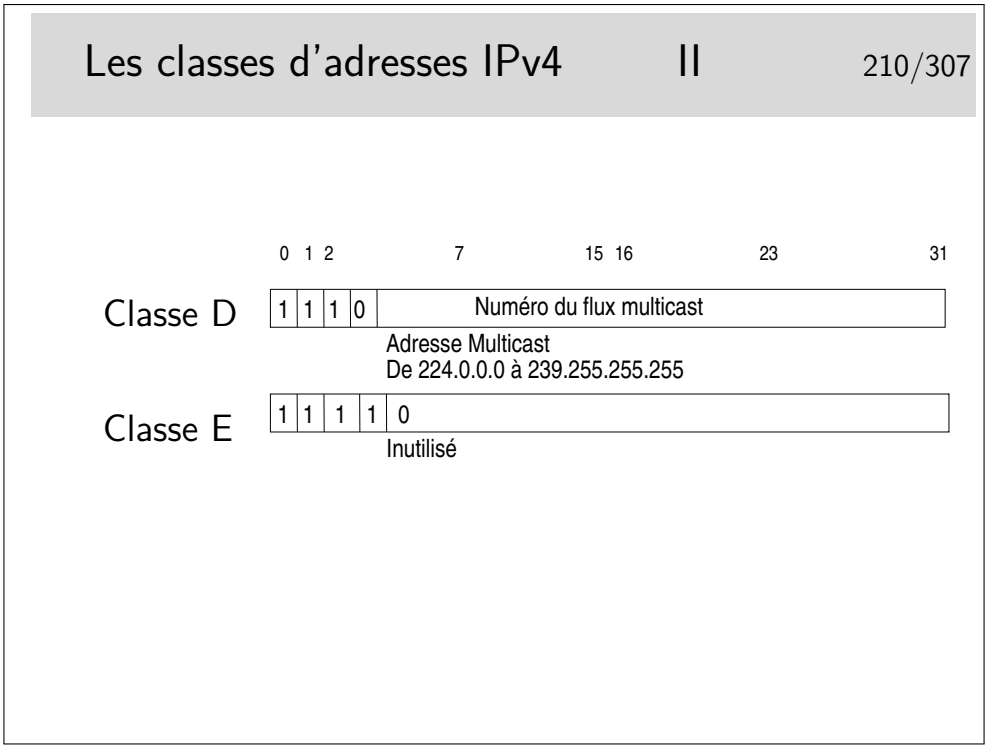


Une interface IP doit être munie d'une adresse pour pouvoir fonctionner. En général, pour des machines de bureau standard en réseau local, l'interface IP correspond à une carte réseau de type Ethernet.

L'interface IP peut aussi être correspondre à une couche complètement logicielle, c'est le cas des interfaces de niveau 2 réalisées avec le protocole PPP qui s'interpose entre IP et une interface physique comme un port série ou un port USB relié à un modem ADSL (l'empilement des couches protocolaires est alors très complexe).

Une interface matérielle peut être munie de plusieurs adresses IP, sous Unix et Linux en particulier. On a alors, par exemple, les interfaces eth0 :0, eth0 :1, eth0 :2, etc.

A noter que les adresses de classe A de la gamme 0.0.0.0 à 0.255.255.255 ne peuvent pas être utilisées, bien qu'elles ne correspondent pas à une fonction particulière.



Le multicast...

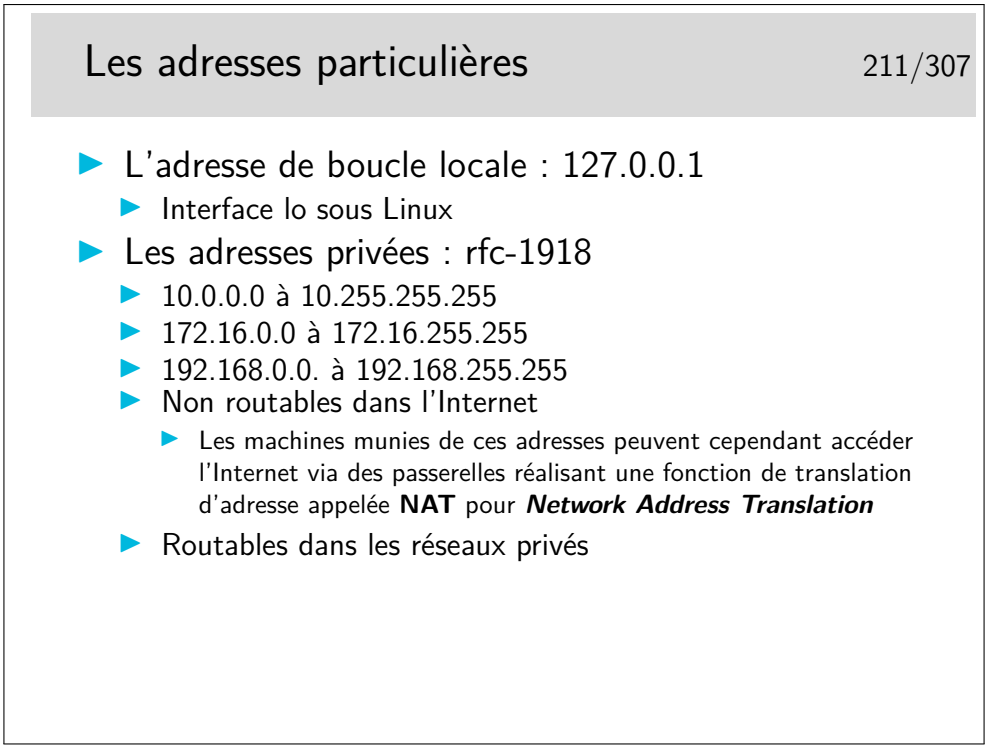
Permet d'envoyer des paquets IP à un certain nombre de machines mais pas à toutes. Des machines sur le réseau peuvent être sources de trafic multicast, elles peuvent émette par exemple à l'adresse 224.5.6.7. Il suffit que d'autres machines soient «au courant» de ce fait pour se mettre en écoute en lançant une application spécifique. Cette application demande explicitement à écouter le flux en envoyant un message du protocole IGMP (RFC-3376 pour la version 3) vers l'Internet. Ce message est relayé par les routeurs vers les sources, chaque routeur rencontré s'insère alors dans un arbre de diffusion multicast (préalablement configuré). Les paquets du flux applicatif peuvent arriver ainsi aux récepteurs ayant fait la requête.

Il faut que les routeurs soient configurés, ce n'est pas automatique.

Il existe un réseau mondial multicast, appelé le MBONE. Les sources potentielles diffusent des annonces de session à l'adresse SAP.MCAST.NET (224.2.127.254) (en UDP, port 9875).

L'outil sdr (pour unix/linux ou windows) permet d'écouter ce flux et afficher les informations de sessions.

(mot clé MBONE ; un lien : `http://www-itg.lbl.gov/mbone/`)



La fonction de NAT est mise en œuvre dans un routeur muni d'une interface configurée avec une adresse officielle (pouvant donc être routée dans l'Internet). Les paquets sortant sont modifiés, le champ «adresse source» est remplacé par l'adresse officielle de l'interface de sortie du routeur. Ce dernier mémorise l'opération pour effectuer la modification inverse pour les paquets en retour.

Des extensions du concept peuvent affecter aussi les ports TCP ou UDP.

Le mécanisme pose un problème pour les protocoles tels que ftp. En effet, ftp demande la création d'une connexion «entrante» pour réaliser les transferts de fichiers, les paquets de demande de connexion doivent être corrélés avec la connexion sortante existante. Les traitements sont alors plus complexes que pour les connexions simples comme celle pour le Web par exemple). Ces concepts demandent de comprendre ce qu'est une connexion au niveau supérieur (il n'y a pas de connexion IP). Ce point sera abordé plus loin dans le chapitre sur TCP.

Sous linux, les mécanismes de translation d'adresse sont directement intégrés au noyau. Une commande permet d'en paramétrer les caractéristiques, il s'agit de iptables qui peut bien plus encore en ce qui concerne les opérations de filtrage pour la sécurité.

Les adresses particulières pour la diffusion 212/307

Différentes possibilités :

- *Broadcast* sur le réseau local : 255.255.255.255
  - Peu utilisée
- Partie réseau normale, partie machine à 255
  - Généralement l'adresse de broadcast mise en œuvre
    - Exemples :
    - 192.168.100.255
    - 172.16.255.255
  - Partie réseau normale, partie machine à 0
  - Ancienne adresse de broadcast pouvant encore être utilisées sur des machines SUN dont l'OS est SUNOS-4

Les adresses de réseaux 213/307

- Les réseaux avec un netmask standard
  - On indique les 4 octets (entre 0 et 255), comme pour une adresse normale, les derniers octets étant à 0 (le dernier pour une classe C, les deux derniers pour ne classe B, les trois derniers pour ne classe A)
    - Exemple : 192.168.100.0
- Les réseaux «subnettés»
  - On fait figurer tous les octets (entre 0 et 255), y compris les bits de l'extension
    - Exemple : 192.168.100.32 (netmask 255.255.255.224 par exemple)
    - Voir page suivante

Le *subnetting* 214/307

- Extension de la partie «Réseau» de l'adresse en empruntant quelques bits de poids forts de la partie machine
- Par exemple 255.255.255.224 pour une classe C
  - Tout à 1 sauf la partie machine (224d = 1110 0000b)
- Le masque indique quels bits
- Permet de créer des «sous» réseaux
  - Les sous réseaux sont raccordés entre eux via des routeurs, comme des réseaux «normaux»
- Il y a toujours un netmask
  - Il est standard s'il ne comporte pas d'extension de bits par rapport à la classe d'adresses : 255.255.255.0 pour une lasse C par exemple

Pardon pour l'Académie, mais «subnetting» est plus joli que «sous-réseautage»

Avec le masque 255.255.255.224 on prend 3 bits de la partie machine. On pourra créer 8 sous réseaux à partir du numéro 192.168.100.0 :

— `192.168.100.0`
— `192.168.100.32`
— `192.168.100.64`
— `192.168.100.96`
— `192.168.100.128`
— `192.168.100.160`
— `192.168.100.192`
— `192.168.100.224`

Question : à quel sous réseau appartient la machine de numéro `192.168.100.72` ?

Le netmask général et ses notations 215/307

- Les adresses sans classe
  - Concept CIDR (RFC-1519) : *Classless InterDomain Routing*
  - La frontière de l'adresse réseau n'est plus figée selon la loi des classes
  - Utile pour agréger des routes dans les tables de routage des routeurs
  - Utile pour les fournisseurs de service pour affecter un sous ensemble d'adresses à un client...
  - Le netmask doit être précisé avec les adresses
- Notation
  - Classique : 255.255.255.128 (25 bits de masque)
  - Notation CIDR : /25 : exemple 192.168.100.128/25

Dans le dernier exemple, l'adresse donnée est une adresse de réseau et non une adresse de machine. Avec une telle adresse (et son masque) on pourra adresser deux fois 128-2 machines : de 192.168.100.1 à 192.168.100.126 (broadcast 192.168.100.127) et de 192.168.100.129 à 192.168.100.254 (broadcast 192.168.100.255).

La notation classique (255.255...) reste très répandue. La notation CIDR peut se rencontrer sous quelques Unix (BSD, Linux, ? ? ?), elles est plus commode d'utilisation.

Affectation d'adresse à une interface IP 216/307

- «À la main»
  - Selon les outils offerts par le système d'exploitation
    - À l'aide d'interfaces graphiques d'administration
    - Via des commandes spécifiques
- Dynamiquement
  - Via le protocole DHCP (*Dynamic Host Control Protocol*)
    - Un serveur est configuré pour donner l'information
  - Sur connexion via liaison point à point et le protocole PPP (Point to Point Protocol). La machine à configurer contacte un serveur situé à l'autre extrémité de la liaison. Le serveur peut lui fournir son adresse

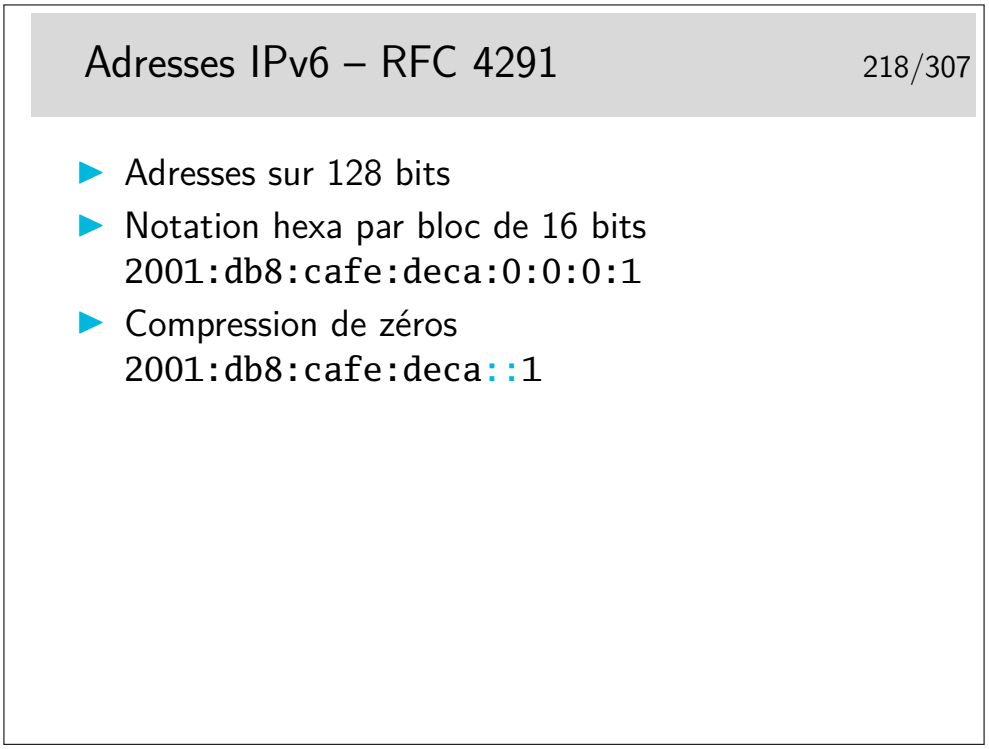
Le protocole PPP est utilisé par exemple dans les connexions aux fournisseurs de

service IP via des modems et le réseaux téléphonique général.

Dans les connexions via ADSL, c'est aussi PPP qui est utilisé pour affecter l'adresse à la machine qui se connecte mais le mécanisme est beaucoup plus complexe.

Le protocole DHCP permet de fournir une adresse IP, le netmask associé, le routeur par défaut pour l'interface en cours de configuration, le nom de domaine DNS ainsi que le ou les serveurs DNS. Tout ceci pour une période de temps configurée par le gestionnaire du serveur DNS. Les clients doivent renouveler leur demande d'adresse à la fin du temps alloué.

10.2 Les adresses IPv6



Sous-réseaux 219/307

- Deux parties :
  - **préfixe** ou **identifiant de sous-réseau** (subnet ID) : partie gauche de l'adresse
  - **identifiant de machine** (host ID) : partie droite
- notation CIDR :
  - `2001:db8:cafe:deca:a9e:1ff:fe6b:25c9/64`
  - subnet correspondant: `2001:db8:cafe:deca::`

Exemples de préfixes 220/307

- Préfixe de documentation `2001:db8::/32`
- Préfixe link-local `fe80::/10`
- Préfixe multicast `ff02::/10`
- Exemple de préfixe «end-user» `2001:db8:fada:ba00:/56`

Exemples d'adresses 221/307

- Unicast
  `2001:db8:cafe:deca:a9e:1ff:fe6b:25c9/64`
- Link-local
  `fe80::a9e:1ff:fe6b:25c9/64`
- Notation IPv4
  `2001:db8::cafe:192.168.0.1` égal à
  `2001:db8::cafe:c0a8:1`
- Localhost `::1`
- Tout les bits à 0 `::`

Affectation d'adresses à une interface IP 222/307

- Il est *normal* d'avoir plusieurs adresses IPv6 à une interface (adresse de portée *lien*, de portée *globale*)
- Configuration «à la main» : plutôt rare
- Dynamiquement
  - Auto-configuration, grâce au préfixe annoncé par le router
  - Via DHCPv6

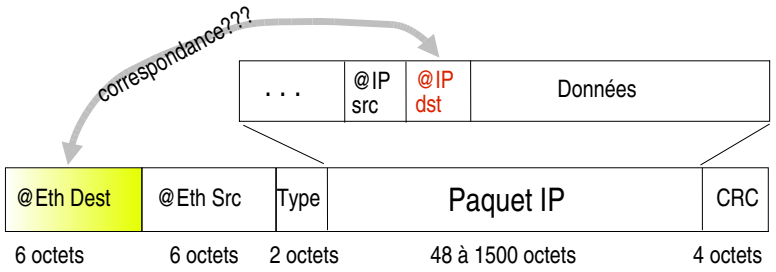
11 Protocole ARP / NDP

Relation entre adresse IP et adresse MAC dans les réseaux locaux 224/307

- Concernent les machines reliées à un réseau local de type Ethernet ou 802.11
- Les adresse IP se gèrent
  - Elles sont affectées «à la main» via des outils spécifiques du système d'exploitation des machines (interfaces graphiques ou commandes telles que ifconfig sous Unix/Linux)
  - Elles peuvent être affectées automatiquement via le protocole DHCP, mais cette possibilité est configurée elle aussi à la main
- Les adresses MAC ne se gèrent pas
  - Elles sont préaffectées par le constructeur de la carte interface que l'on achète ou qui est fournie avec la machine
  - Parfois le pilote (drivers) de la carte permet que cette adresse puisse être modifiée (via ifconfig)

...Adresses MAC et adresses IP... 225/307

- Problème : une machine M doit émettre un paquet IP vers une machine N voisine dont on ne connaît que le numéro IP (sur le même LAN)
- Si on est sur Ethernet le paquet IP sera véhiculé par une trame telle que celle-ci

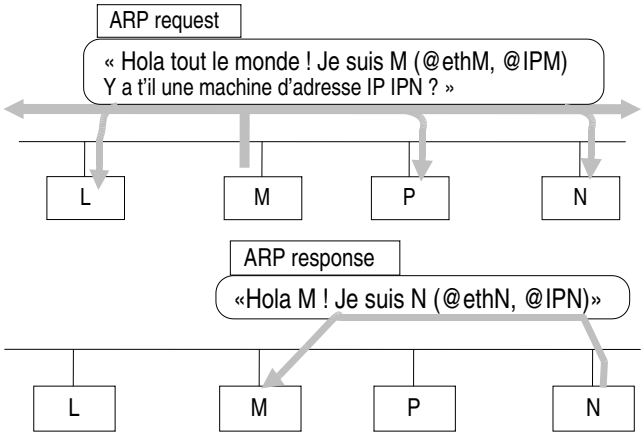


Comment la machine M peut elle retrouver l'adresse Ethernet de la machine N en ne connaissant que l'adresse IP de N ?

Adresses MAC et adresses IP : la solution ARP (IPv4)

226/307

- Adress Resolution Protocol - rfc826



Soyez curieux : la commande `arp`

Sur un réseau local, sous Windows ou Unix/Linux, ouvrez une fenêtre de commande et tapez : `arp -a` pour voir la table de résolution arp instantanée. Si vous ne voyez pas la machine de votre voisin et que vous connaissez son adresse IP (ou son nom), faites un «ping» dessus : ping 192.168.100.5 par exemple.

Si la réponse du ping est positive, refaites alors immédiatement `arp -a`, vous verrez apparaître la résolution concernant la machine voisine.

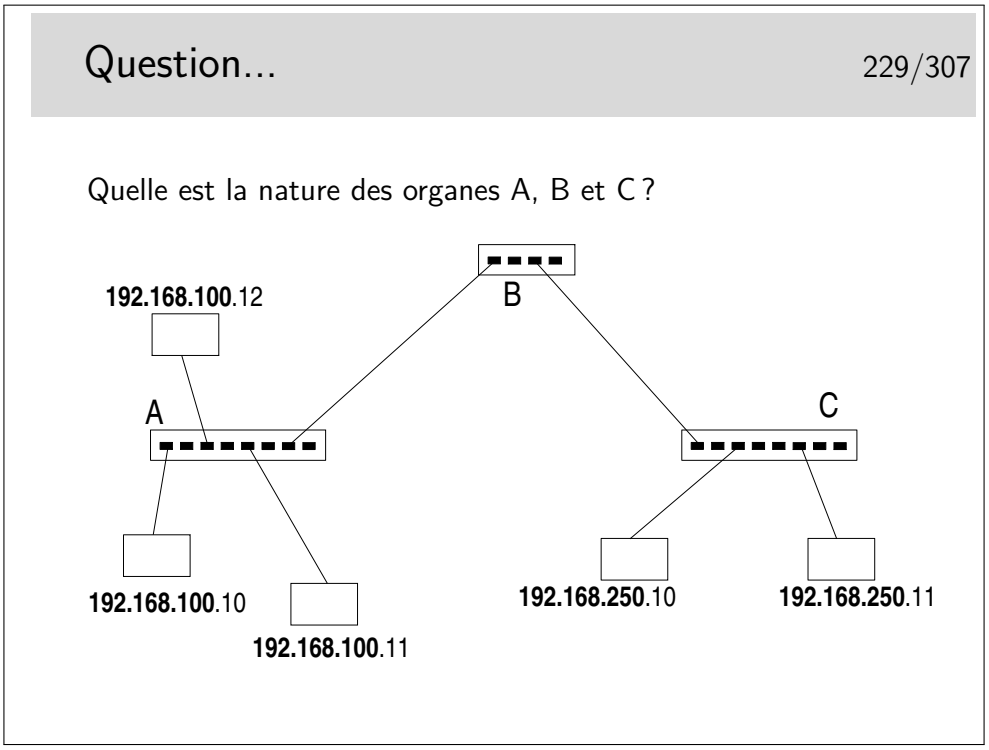
Adresses MAC et adresses IP : Neighbor Discovery Protocol (IPv6)

227/307

- Neighbor Discovery Protocol (NDP)
- Intégré dans Internet Control Message Protocol version 6 (ICMPv6) – rfc4443
- Fonctionne de manière similaire à ARP (IPv4)

Soyez curieux : la commande `ip neigh`

12 Principe du routage



Vous avez le choix entre trois types d'organes :

— des hubs,

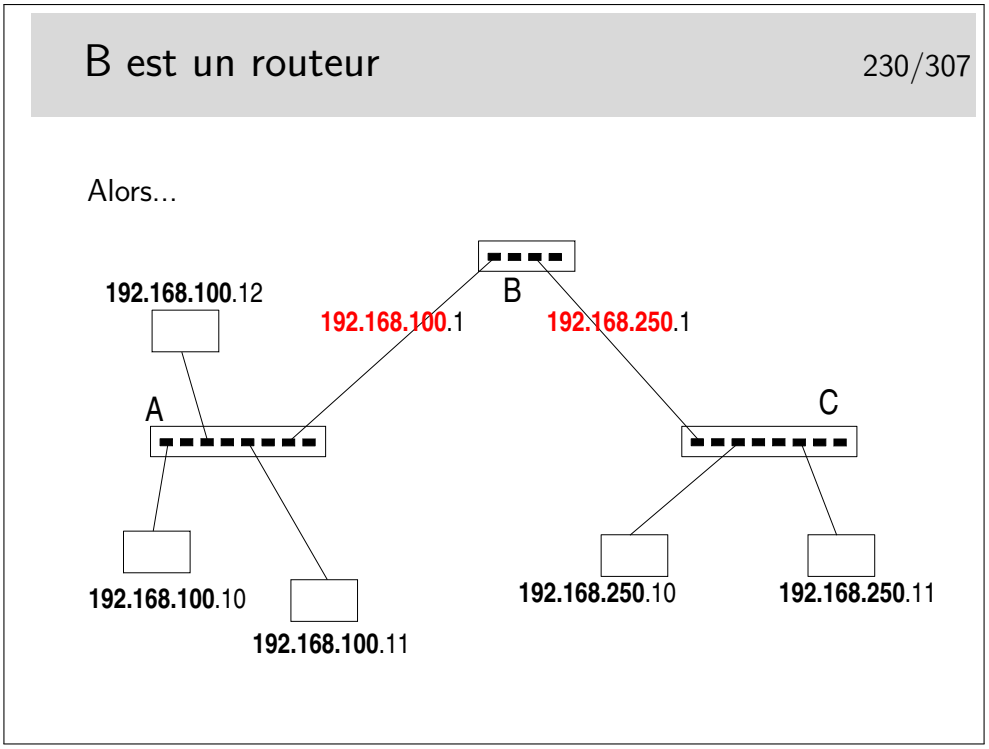
— des switches ou commutateurs ou pont multiports

— des routeurs

Aide technique : le netmask est standard pour la classe d'adresses IP utilisée...
Et cette classe est la classe ? ? ? ? ?....
? ? ?
? ? ?
...C !
Ce sont des adresses de classe C...

Or les parties «Réseau» ne sont pas identiques entre la partie droite du schéma et la partie gauche...

Donc... Il ne s'agit pas de mêmes «Réseaux». Donc, forcément B est un... et A et C sont des ... ou des ...



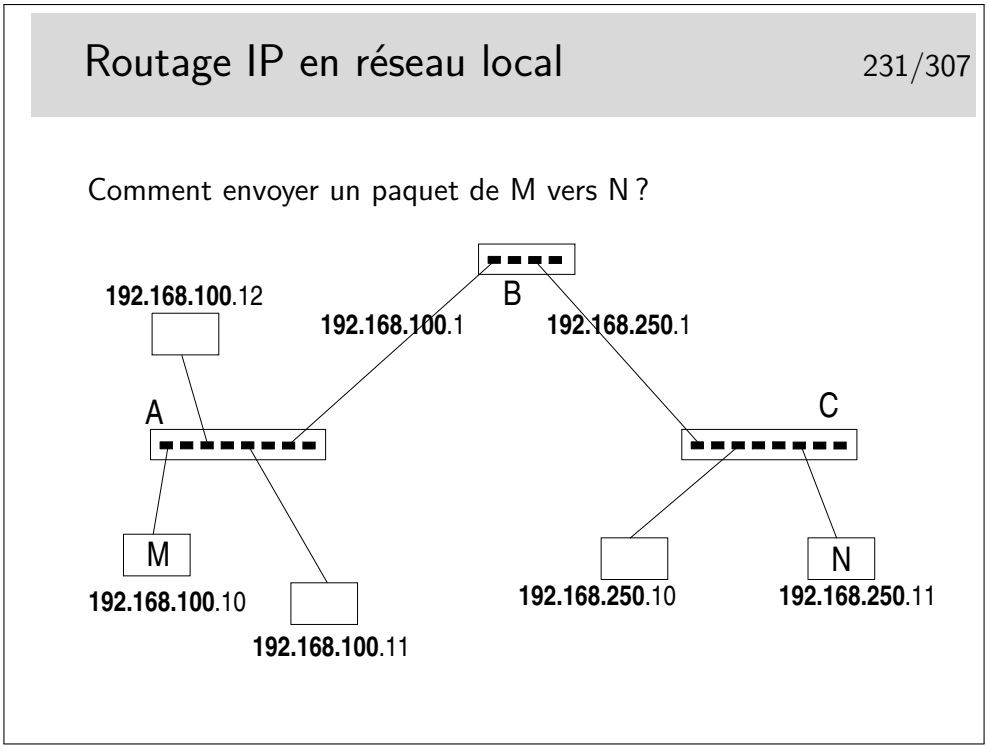
Réponse à la question... A et B sont des hubs ou des commutateurs (ou switches ou ponts multiports). Même si fondamentalement les fonctionnalités des hubs et des commutateurs sont différentes (hubs : répéteurs, organes physiques, niveau 1 ISO – switches : niveau 2 ISO) on ne peut pas faire la différence sur le schéma.

B est un routeur. Il raccorde des réseaux dont les machines sont identifiées par des numéros IP de classe C dont la partie réseau est différente. Il raccorde dont des réseaux. Il fonctionne au niveau 3 ISO.

B est un routeur, donc chacun de ses ports est identifié par une adresse IP dont la partie réseau identifie le réseau auquel ce port est raccordé.

Le premier port de B est raccordé sur le réseau 192.168.100.0, on lui donnera donc une adresse libre de ce réseau (ici .1). Un autre port est sur le réseau 192.168.250.0, on lui donnera par exemple l'adresse .1 sur ce réseau.

Ce n'est pas pour faire joli, c'est vraiment fonctionnel. Si on ne le fait pas le réseau ne peut pas fonctionner.



Rien n'est magique... L'application sur M qui désire envoyer un message à une application sur N doit savoir que N existe. Sur M on doit connaître l'adresse IP de N.

On doit aussi savoir par où on passe. En M on doit savoir que pour atteindre N il faut passer par le routeur B.

Mais B est identifié par deux adresses IP !!!

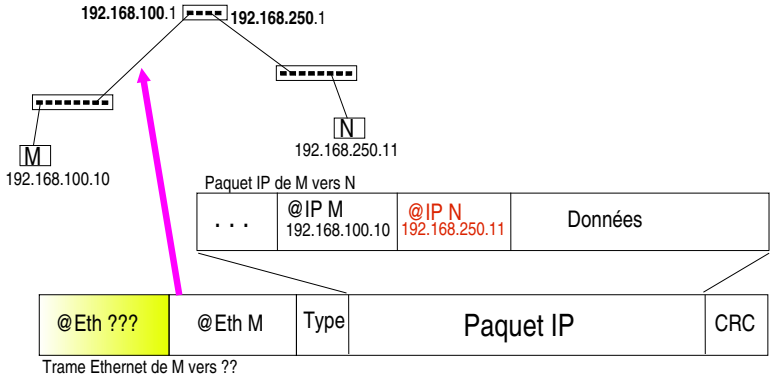
En M, nous sommes sur un réseau local, nous ne pouvons atteindre que des machines se trouvant sur le même réseau local que nous. Nous ne pouvons donc atteindre le routeur que par son port qui est situé sur le même réseau physique, celui identifié par le préfixe IP 192.168.100.

...Comment faire ?...

À suivre...

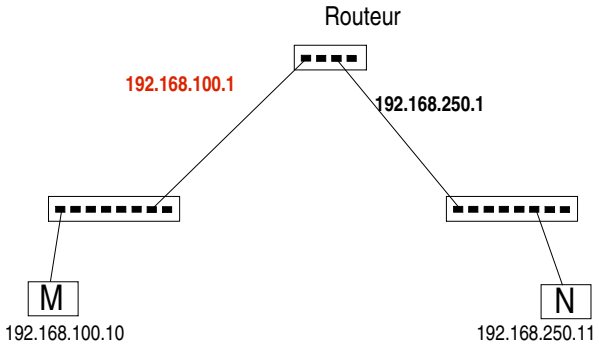
Paquet IP de M vers N, et son porteur... I 232/307

En réseau local



Comment, en M, déterminer l'adresse Mac de la trame Ethernet qui va emporter le paquet vers sa destination ?

Table de routage I 233/307

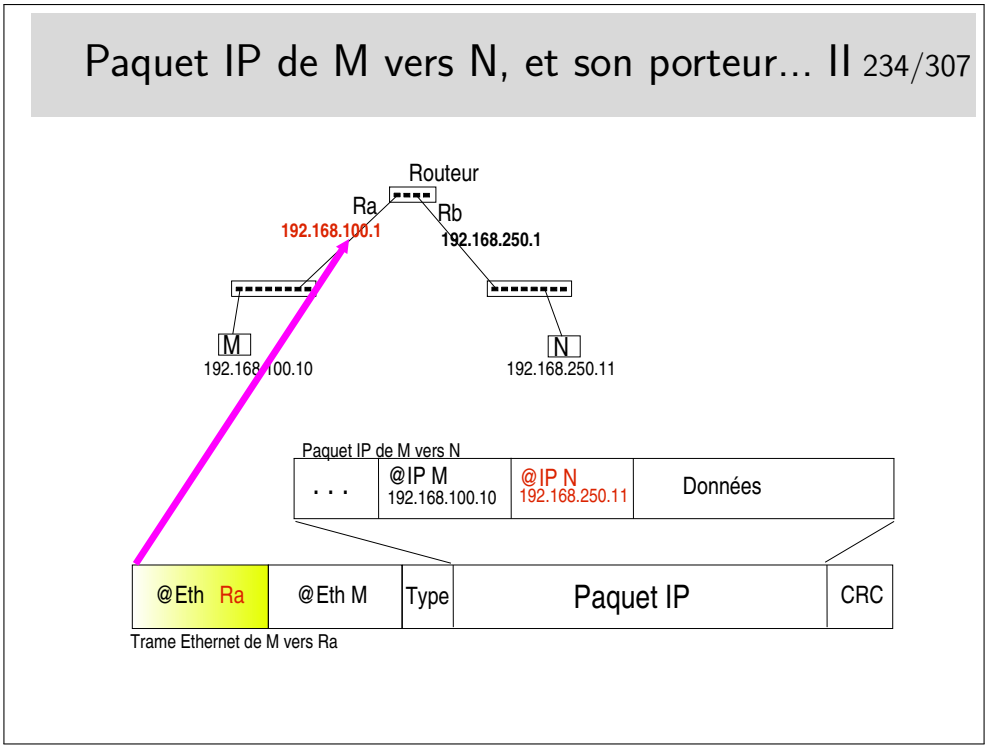


En M il faut une table de routage qui dit : « pour aller en 192.168.250.11 passer par 192.168.100.1 »
À l'aide de cette table il sera facile de faire une résolution ARP pour trouver l'adresse MAC du routeur. L'adresse destination de la trame Ethernet sera celle du routeur

Il faut une table de routage en M. Donc dans la machine terminale !!!

Chaque machine, quelle soit terminale ou intermédiaire, travaillant en IP incorpore des fonctions de routage.

Cela ne signifie pas que chaque machine terminale soit un routeur... Pour qu'une machine puisse joue le rôle de routeur il faut qu'elle possède au moins deux interfaces munies d'adresses IP (autres que 127.0.0.1, correspondant à l'interface boucle locale) et que son module IP soit autorisé à effectuer la fonction de relayage (forwarding).



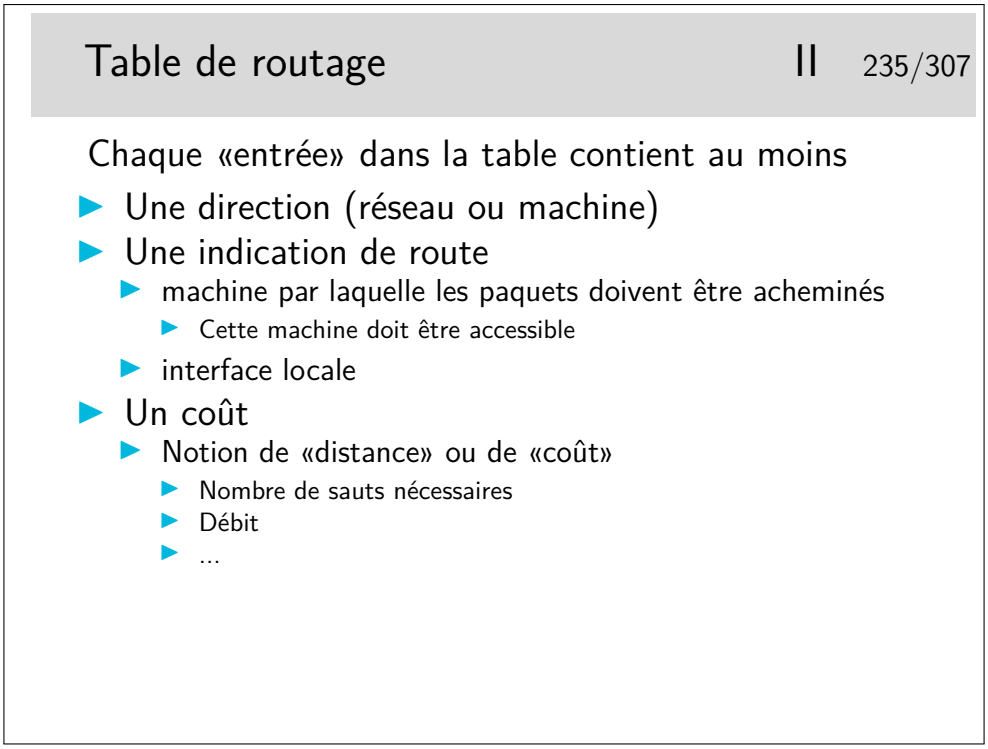
Quand on disait qu'affecter une adresse à chaque interface de routeur n'était pas un effet de cosmétique...

Si en M on ne connaît pas l'adresse IP de l'interface du routeur qui est du même coté que M (sur le même réseau) alors la résolution ARP ne peut se faire.

Il ne suffit pas que l'interface du routeur et M soient «du même coté», il faut aussi qu'ils soient sur le même réseau local pour que ARP fonctionne (on rappelle que la requête ARP est transmise par broadcast Ethernet et que ce type de message ne passe pas les routeurs).

Ici la notion de réseau local est celle qui a été vue dans la partie précédente du cours, à savoir un réseau où la diffusion est possible vers toutes les machines. Ce pourrait être un VLAN car ce type de topologie définit des domaines de broadcast.

Lorsque des machines sont reliées entre-elles via des liaison point-à-point, il n'y a pas de résolution ARP.



Les directions sont spécifiées par des numéros de réseaux indiqués sur 4 octets et accompagnés de leur netmask, par exemple 192.168.10.0 255.255.255.0, ou en notation CIDR 192.168.10.0/24. Attention, la notation CIDR peut ne pas être comprise par tous les routeurs. Elle l'est pour les machines Unix Linux ou BSD.

Une direction peut être une machine, dans ce cas l'adresse IP de la machine est indiqué directement et le netmask vaut 255.255.255.255.

Les indications de route peuvent être données en indiquant l'adresse IP du routeur par lequel il faut envoyer les paquets pour la direction correspondante. Plus précisément il s'agit de l'adresse IP de l'interface du routeur immédiatement accessible. C'est absolument obligatoire dans le cas ou l'interface du routeur en question est sur un réseau local classique.

Dans le cas ou l'interface du routeur est sur une liaison point-à-point, on peut ne donner que le nom de l'interface locale. Voir transparent suivant.

Le coût correspond à la notion de «plus court chemin». Moins le coût est grand meilleur semble le chemin (ce n'est pas toujours vrai, un chemin peut être plus court en vraie distance (moins de sauts par exemple) et cependant moins efficace en débit).

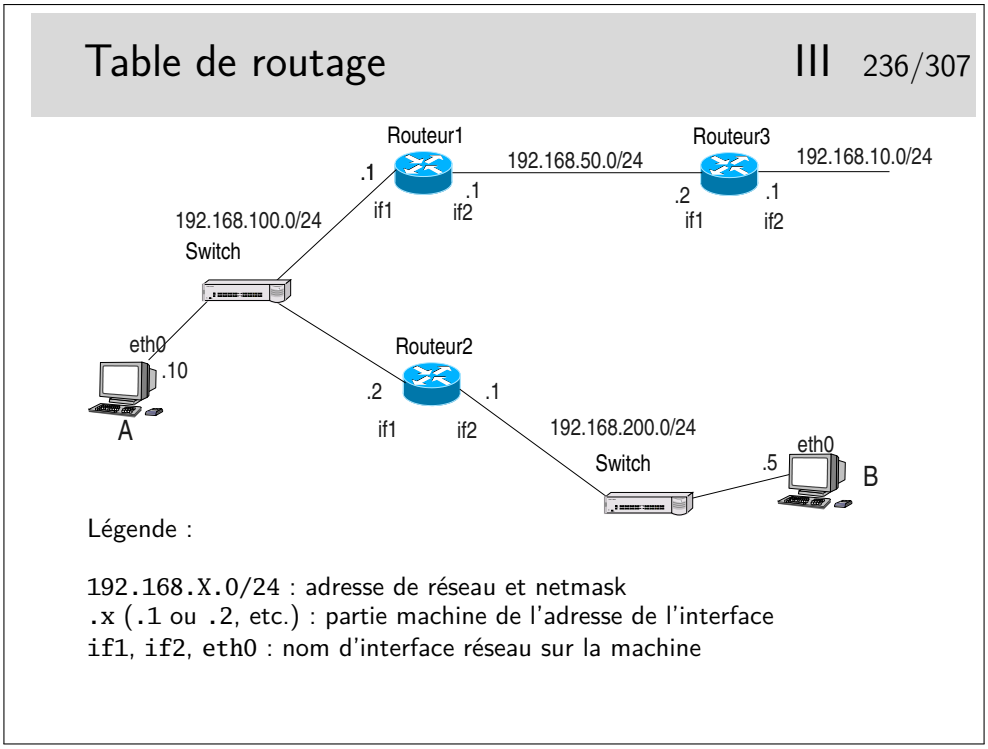


Table de routage en A

direction	netmask	gateway	interface
192.168.100.0	255.255.255.0	192.168.100.10	eth0
192.168.200.0	255.255.255.0	192.168.100.2	eth0
default	0.0.0.0	192.168.100.1	eth0

Rq : ici la mention de l'interface n'est pas très importante, elle est redondante car il n'y a qu'une seule interface physique (il y a quand même l'interface boucle locale qu'on n'a pas fait figurer dans la table de routage, mais qui exite et provoque des entrées spécifiques dans la table)

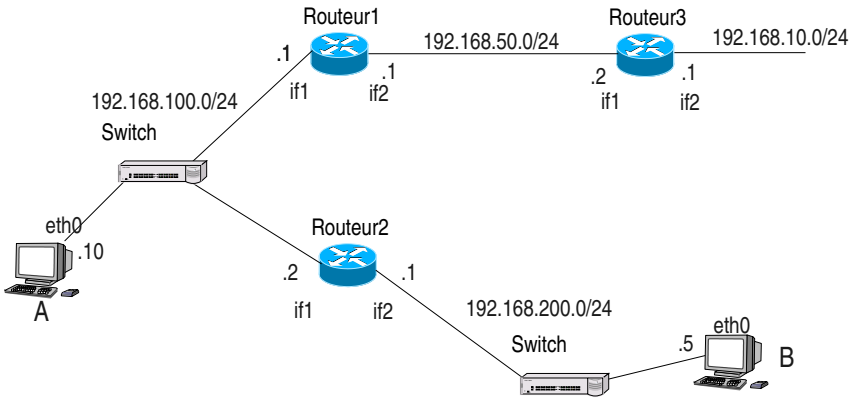
Table de routage en Routeur1

direction	netmask	gateway	interface
192.168.50.0	255.255.255.0	192.168.50.1	if2
192.168.100.0	255.255.255.0	192.168.100.1	if1
192.168.200.0	255.255.255.0	192.168.100.2	if1
default	0.0.0.0	192.168.50.2	if2

Rq : dans la ligne par défaut, la seule mention de l'interface suffirait car le lien entre les routeurs 1 et 2 est de type point-à-point.

Compléter pour le routeur 2 et la machine B...

Table de routage IV 237/307



Considérons les tables de routage correctement servies partout sauf en A
A peut atteindre if1 de Routeur3 (192.168.50.2), mais pas if2 (192.168.10.1)
Pourquoi ne peut-on pas dire en A : pour aller en 192.168.10.0/24 passer par 192.168.50.2 ? *Que faut-il dire ?*

On suppose la table de routage en A partiellement remplie. L'interface if1 du routeur2 est accessible, une commande ping, depuis A, vers l'adresse de cette interface fonctionne. Donc on peut supposer (et on a raison) que cette interface est «visible» depuis A. Cependant si on tente de créer en A une route via cette interface la commande doit échouer. Pourquoi ?

Exemple de table de routage sous Windows 238/307

```
C:\>route print
===========================================================
Liste d'Interfaces
0x1 ........................... MS TCP Loopback interface
0x1000003 ...00 01 02 6e 7c 46 ...... 3Com EtherLink PCI
===========================================================
Itinéraires actifs:
  Destination réseau     Masque réseau   Adr. passerelle   Adr. interface   Métrique
            0.0.0.0           0.0.0.0      192.44.75.1    192.44.75.184      1
          127.0.0.0         255.0.0.0        127.0.0.1        127.0.0.1      1
        192.44.75.0     255.255.255.0     192.44.75.184    192.44.75.184      1
      192.44.75.184   255.255.255.255        127.0.0.1        127.0.0.1      1
      192.44.75.255   255.255.255.255     192.44.75.184    192.44.75.184      1
          224.0.0.0         224.0.0.0     192.44.75.184    192.44.75.184      1
    255.255.255.255   255.255.255.255     192.44.75.184    192.44.75.184      1
Passerelle par défaut: 192.44.75.1
===========================================================
```

Quelle est le numéro de l'interface de boucle locale (interface interne non raccordée à un réseau physique) ?

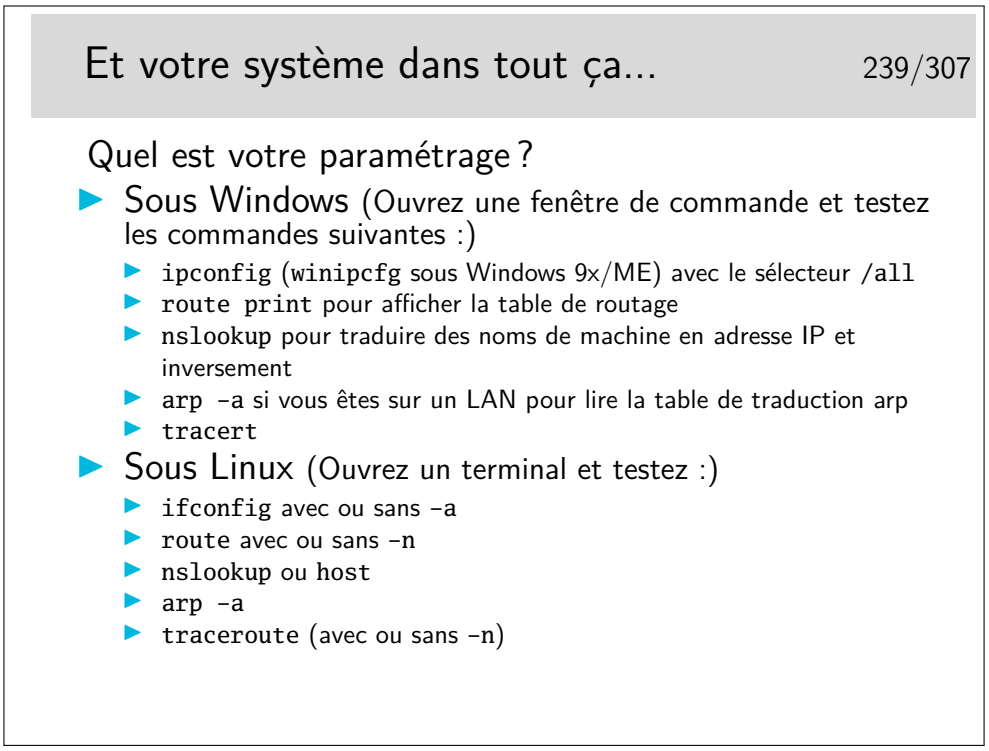
Quelle sont les adresses de broadcast possibles ?

Par quelle interface sont acheminés les paquets de broadcast ?

Que pouvez vous déduire de tout ceci concernant l'adresse de la machine en question ?

Que se passe-t'il si une application de cette machine envoie un paquet IP vers une autre application de la même machine ? Quel chemin est emprunté par le paquet ?

Quelle adresse code la destination par défaut ?



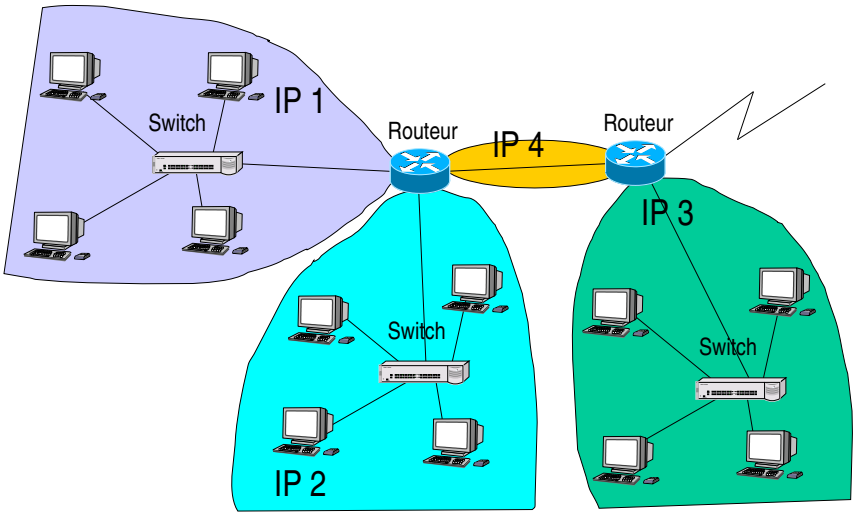
Et n'oubliez pas la commande ping... C'est la première commande à utiliser quand le réseau ne va pas très bien... «Mon voisin est-il joignable ? Alors :

`ping mon_voisin` (plutôt `adresse_IP_de_mon_voisin`)

Note : Vous remarquerez peut être que Windows affiche plus de routes que Linux. En fait Linux gère plusieurs tables de routage et a pour habitude de n'afficher que la table `main`, sauf si on lui demande gentiment. (Pour avoir la liste des tables de routage : `cat /etc/iproute2/rt_tables` ; pour afficher la table principale : `ip route show table main` ; pour afficher la table avec les règles pour le multicast et le broadcast : `ip route show table local` ; etc.)

13 Les routeurs

Le réseau IP de l'entreprise 241/307



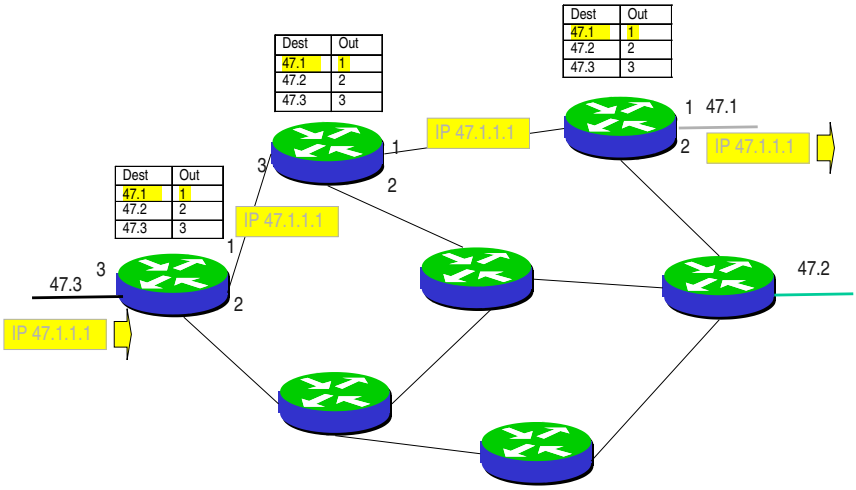
Ce que l'on va appeler le «réseau local» de l'entreprise est en réalité un réseau de niveau 3 composé de routeurs interconnectant des réseaux locaux séparés physiquement ou logiquement (VLANs).

L'adressage pourra être de type «privé», avec une seule adresse officielle en sortie et une fonction de type NAT dans le routeur de sortie.

Fonctions des routeurs 242/307

- Chaque paquet IP entrant dans le routeur voit son adresse destination examinée et comparée sur un certain nombre de bits avec le contenu d'une table (table de routage) associant des directions avec une interface de sortie
  - Cette fonction est aussi mise en œuvre dans les machines terminales
  - Le champ TTL est décrémenté par chaque routeur, le champ *cheksum* doit être recalculé à chaque fois

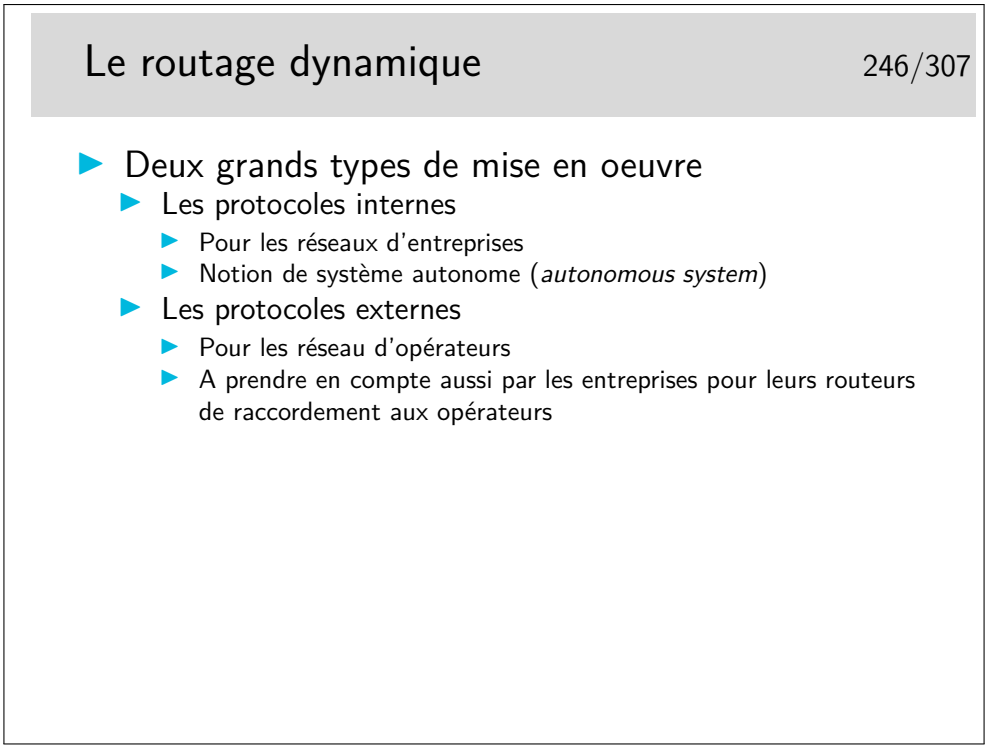
Acheminement bond par bond 243/307



14 Protocoles de routage

L'établissement de la table de routage 245/307

- Routage statique
  - La table est remplie «à la main» via une commande spécifique ou une interface graphique
- Routage dynamique
  - Via des protocoles appropriés
  - Permettent de détecter des ruptures de liens et de reconfigurer les tables de routage de manière dynamique



Un AS (*Autonomous System*) est un grand système réseau (typiquement un fournisseur d'accès) qui gère de manière cohérente et autonome son routage. Les numéro d'AS (utilisé notemment dans le routage BGP, cf. plus loin) sont attribués par les RIR (*Regional Internet Registry*) sous l'autorité de l'IANA. Le RIP européen est RIPE-NCC (*Réseaux IP Européens - Network Coordination Centre*).

Par exemple en 2015 on dénombre 1208 "opérateurs" qui offrent des services sur le territoire français : `http://www.ripe.net/membership/indices/FR.html`. (Ne pas croire que les seuls opérateurs sont Orange SFR Bouygues...)

Ces grands systèmes ont un numéro d'AS : France Télécom (AS3215), Proxad (AS12322), LDCOM Networks (AS15557 gaoland.net), Cegetel (AS8228), 9TELECOM(AS12626 9tel.net), AOL France customers , SIRIS (AS3305 siris.fr), KAPTECH(AS6760 kaptech.net), BELGACOM France (AS6771), Nerim (AS13193), etc.

Ces AS sont interconnectés via des points dits de *peering* : France-IX, SFINX, PARIX, FreeIX, PanaP, LyonIX, LINX, AMS-IX, DE-CIX, etc. Voir `http://en.wikipedia.org/wiki/List_of_Internet_exchange_points`.

Par exemple, à quoi est raccordé SFR ?

Tapez `whois as15557`

`http://www.robtex.com/as/as15557.html`

Quelles sont les blocks d'addresse alloués à Free :
`whois -r -i org ORG-PISP1-RIPE`

À quoi est raccordé Télécom Bretagne ? Voyez `http://www.robtex.com/dns/telecom-bretagne.eu.html#graph`

Un autre concept important est la séparation (conceptuelle) entre cœur et bordure d'Internet. On parle alors de la *Default-Free Zone* (DFZ). C'est l'ensemble des routeurs qui n'ont pas de *route par défaut*. Autrement-dit, ce sont les routeurs qui constituent *le*

cœur du réseau Internet (et ceci indépendemment du découpage d'Internet en AS). Par opposition, tous les autres routeurs qui ont à priori une route par défaut vers le reste d'Internet, constituent la *la bordure* d'Internet.

Le principe de fonctionnement du routage dynamique I 247/307

- La configuration des interfaces d'un routeur «apprend» à celui-ci les numéros des réseaux sur lesquels il est raccordé
  - Le routeur connaît donc déjà quelques réseaux
  - Il peut communiquer cette connaissance aux autres autres routeurs situés sur les mêmes segments de réseau que lui, via un protocole approprié
  - Les autres routeurs peuvent faire de même

../..

Le principe de fonctionnement du routage dynamique II 248/307

../..
- Au bout d'un certain temps tous les routeurs connaissent tout sur tout les autres
- Les routeurs continuent périodiquement à s'envoyer des informations. Si l'un d'eux tombe en panne, les autres, ne recevant plus ses informations, le considèrent comme hors service et inhibent les routes qui passaient par lui. Ils peuvent aussi déterminer des routes alternatives

Les protocoles de routage 249/307

- Deux grands types d'algorithmes
  - Le vecteur de distance
    - Les routeurs s'échangent leur table de routage (destination et distance ou coût) entre voisins
    - Protocole de type interne RIP (Routing Information Protocol), le plus répandu mais peu performant
  - L'état de liens
    - Tous les routeurs d'un domaine s'échangent leurs informations de routage
    - Protocole de type interne OSPF (Open Shortest Path First), plus performant que RIP mais complexe

Routage interne et externe 250/307

- Tous les routeurs du monde ne doivent pas avoir la copie des tables de routage de tous les autres routeurs
- Concept de domaines
  - Un domaine est une entreprise, un campus, une entité restreinte
  - Des protocoles de routage sont adaptés au routage intra domaine
    - RIP, OSPF
- Routage inter domaines
  - Les informations sont agrégées
  - Notion d'autonomous système
  - Protocole fortement utilisé : BGP (Border Gateway Protocol)

Protocoles de routage les plus répandus I

251/307

- RIP : Routing Information Protocol
  - Routage intra-domaine
  - Type vecteur de distance
  - Peu performant
  - Pauvre notion de coût (nombre de saut)
  - Facile à mettre en œuvre
- OSPF : Open Shortest Path First
  - Routage intra-domaine
  - Famille état des liens (link state)
  - Notion de coût plus riche (par défaut : 108/bande_passante)
  - Routage hiérarchique avec délimitation d'aires

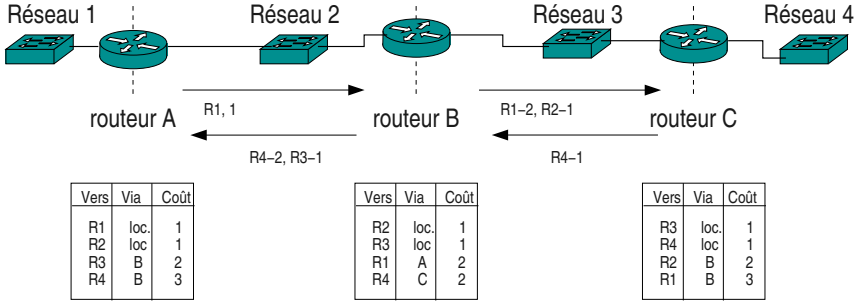
Protocoles de routage les plus répandus II

252/307

- BGP : Border Gateway Protocol (rfc1771)
  - Routage inter-domaines
  - Prend en compte le routage sans classe (CIDR rfc1466)
  - Véhiculé par TCP

Le protocole RIP 253/307

- Vecteur de distance : *direction,coût*



Vers	Via	Coût
R1	loc.	1
R2	loc	1
R3	B	2
R4	B	3

Vers	Via	Coût
R2	loc.	1
R3	loc	1
R1	A	2
R4	C	2

Vers	Via	Coût
R3	loc.	1
R4	loc	1
R2	B	2
R1	B	3

- Pourquoi les routeurs n'annoncent-ils pas tout le contenu de leur table ?

Chaque routeur annonce, systématiquement toutes les 30s, sa table de routage à ses voisins (seulement les réseaux connus et le coût pour les atteindre, ils n'annoncent pas le paramètre «via»). Chaque routeur prend donc «connaissance» de la «connaissance» des ses voisins.

Lorsque l'algorithme a convergé chaque routeur doit connaître l'existence des tous les réseaux. Si une modification du routage apparaît dans un routeur (nouvelle destination, routeur voisin silencieux), le routeur en question recalcule sa table. S'il en résulte une modification (comme dans le cas de l'apparition d'une nouvelle route) il en informe immédiatement ses voisins sans attendre les 30s habituelles.

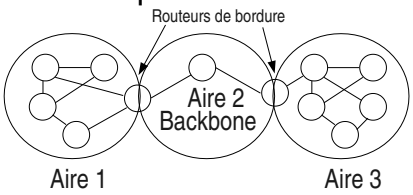
Le schéma ci-dessus montre les tables pour un réseau simple une fois atteinte la convergence. Pourquoi les routeurs n'annoncent-ils pas toute leur table ? Par exemple, pourquoi A n'annonce-t'il pas R4 ? L'algorithme serait plus simple à implémenter.

En fait A n'annonce pas vers B qu'il connaît R4 car il apprend l'existence de R4 par B lui-même. Et si C tombait en panne, B ne recevant plus d'annonce de C placerait R4 à une distance infinie et le supprimerait de sa table. Puis il indiquerait vers A ce fait nouveau. Mais peut-être que A aurait le temps de communiquer à B sa connaissance de R4 avec un coût de 2. B pourrait alors croire que A connaît une route vers R4.

Cette non annonce, dans une direction, des routes que l'on apprend par elle s'appelle la technique de l'**horizon coupé** (*splitted horizon*). Elle n'est pas complètement satisfaisante car s'il existe plusieurs autres routeurs du coté de A et que ceux-ci forment des boucles, il se peut que l'algorithme ne puisse pas converger.

Le protocole OSPF 254/307

- rfc1583, 1587, 2328
- Routage hiérarchique : notion d'aires



  - Chaque aire doit être reliée à l'aire backbone qui assure le transit entre deux aires
  - Dans chaque aire les routeurs ont une connaissance complète du routage dans leur aire
  - Les routeurs de bordure d'aires ont une connaissance des aires qu'ils raccordent, ils propagent dans chaque aire une connaissance agrégée des autres aires

Un exemple de découverte de route 255/307

Sous Windows :

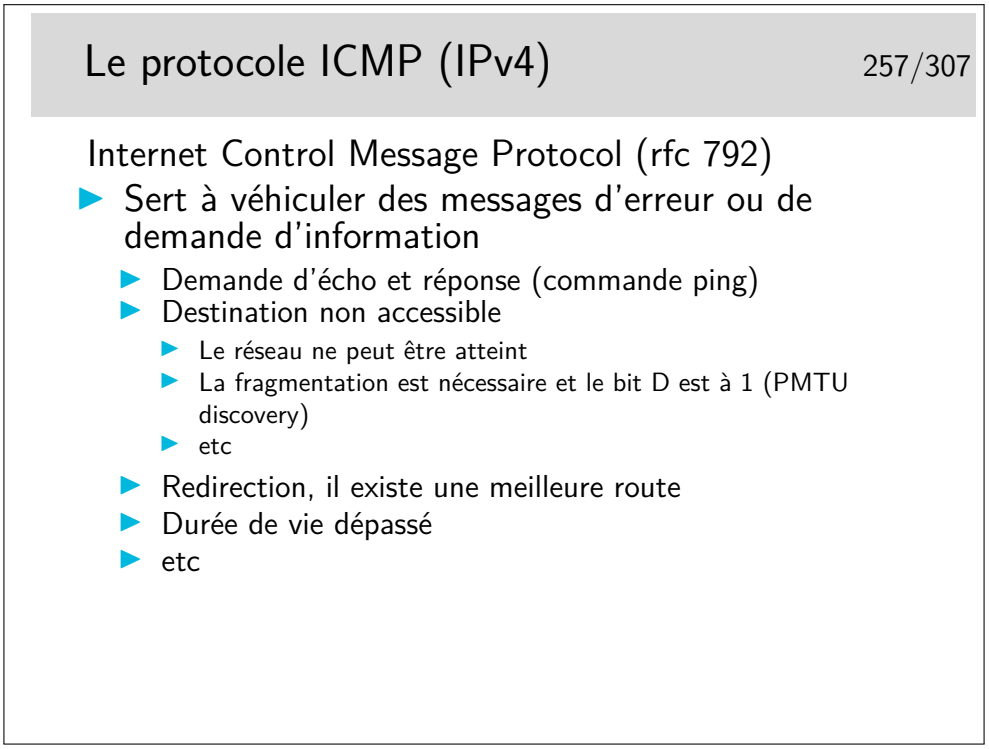
```
C:\>tracert www.redhat.com
Détermination de l'itinéraire vers www.redhat.com [209.132.177.50]
avec un maximum de 30 sauts :
 1 <10 ms  16 ms <10 ms galaxie-75.enst-bretagne.fr [192.44.75.1]
 2 <10 ms <10 ms <10 ms 193.50.69.89
 3 <10 ms <10 ms <10 ms 193.48.78.17
 4 <10 ms  15 ms <10 ms PAO-Rennes2-TR.rrb.ft.net [195.101.145.25]
 5  16 ms <10 ms  16 ms peering-GIP.rrb.ft.net [195.101.145.6]
 6 <10 ms  16 ms <10 ms rennes-a0-0-2.cssi.renater.fr [193.51.181.126]
 7  16 ms  16 ms  15 ms caen-pos1-0.cssi.renater.fr [193.51.180.17]
 8  15 ms  16 ms  15 ms rouen-pos1-0.cssi.renater.fr [193.51.180.22]
 9  31 ms  16 ms  16 ms nri-a-pos6-0.cssi.renater.fr [193.51.179.21]
10  32 ms  15 ms  16 ms 193.51.185.1
11  15 ms  16 ms  15 ms P11-0.PASCR1.Pastourelle.opentransit.net [193.251.241.97]
12  93 ms  94 ms  94 ms P12-0.NYKCR3.New-york.opentransit.net [193.251.241.134]
13  94 ms  94 ms  94 ms ft-gw.n54ny.ip.att.net [192.205.32.137]
14  93 ms  94 ms  94 ms tbr1-p010401.n54ny.ip.att.net [12.123.3.57]
15 110 ms 125 ms 109 ms tbr1-cl1.cgcil.ip.att.net [12.122.10.2]
16 110 ms 109 ms 125 ms tbr2-cl2.cgcil.ip.att.net [12.122.9.134]
17 172 ms 172 ms 172 ms tbr2-cl7.sl9mo.ip.att.net [12.122.10.46]
18 125 ms 140 ms 125 ms tbr2-cl6.dlstx.ip.att.net [12.122.10.90]
19 172 ms 172 ms 172 ms gbr2-p40.phmaz.ip.att.net [12.122.10.86]
20 172 ms 172 ms 172 ms gar2-p370.phmaz.ip.att.net [12.123.142.49]
21 172 ms 172 ms 172 ms mdf1-gsr12-1-pos-7-0.phx1.attens.net [12.122.255230]
22 141 ms 156 ms 141 ms mdf1-bi8k-2-eth-2-1.phx1.attens.net [63.241.128.158]
23  *   *   *  Délai d'attente de la demande dépassé.
```

La recherche de chemin n'aboutit pas, des filtrages sur certains routeurs en sont la cause.

Cette commande peut être rapide si on lui ajoute l'option -d qui empêche la traduction des adresses IP en noms.

La commande existe sous Unix/Linux sous le nom traceroute.

15 Gestion des erreurs



La commande `ping` n'utilise que ICMP (porté par IP).

La commande `traceroute` (`tracert` sous Windows) joue sur le dépassement de la durée de vie. Un premier paquet est créé «à la main» et son champ ttl est mis à 1. Il contient un paquet UDP à destination d'un port inconnu. Le paquet est routé, il atteint le premier routeur qui décrémente alors le ttl. Le résultat valant 0 le paquet est jeté et un message ICMP est émis vers la source du paquet jeté. Le message est véhiculé par un paquet IP comportant l'adresse du routeur, ce qu'on attend dans traceroute. On peut alors afficher l'identité de ce routeur ainsi que le temps mesuré entre l'envoi du paquet initial et le l'arrivée du message ICMP. On fait cela trois fois pour avoir une estimation du temps moyen d'aller et retour puis on recommence en mettant cette fois le ttl à 2.

Le protocole ICMPv6 (IPv6) 258/307

Internet Control Message Protocol v6 (rfc 4443)

- Mêmes fonctions que ICMP (IPv4)
  - Erreurs diverses, ping, redirections, etc.
- En plus :
  - Neighbor Discovery Protocol (équivalent du ARP pour IPv4)
  - Router Solicitation, Router Advertisement : Auto-configuration de l'adresse. Au démarrage, un host recherche s'il y a un routeur présent sur LAN ; le routeur diffuse le préfixe IPv6 à utiliser ; le host se choisit une adresse dans ce préfixe.

16 Services pour IP

Les services pour IP 260/307

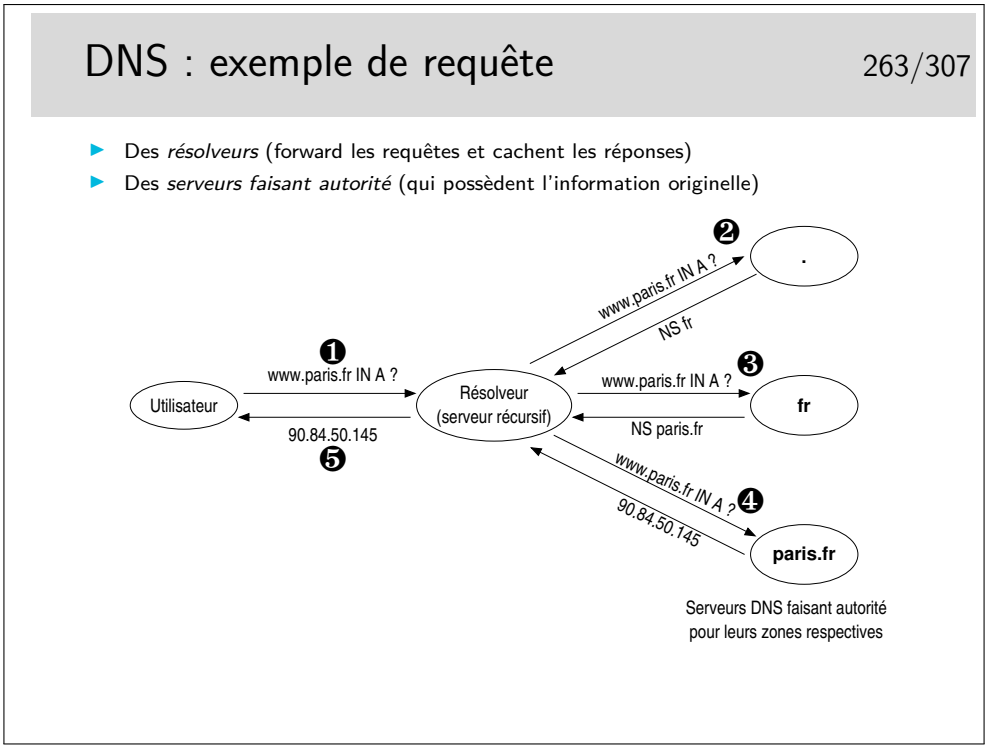
- Le protocole DHCP : Dynamic Host Control Protocol (DHCPv4 rfc2131, DHCPv6 rfc3315)
  - Utilisé pour obtenir une adresse IP de manière dynamique ainsi que des informations diverses : serveur DNS, domaine, netmask, routeur par défaut
- Le service de nommage : le DNS
  - Domain Name Service (rfc 1034 et 1035)
    - Permet d'associer un nom à une adresse IP
    - Offre un protocole pour retrouver ce nom en partant de l'adresse ou l'adresse en partant du nom

Le Domain Name Service - DNS

- Service réparti de nommage
- Répartition mondiale
- Gestion indépendante de chaque domaine, pas d'autorité absolue de référence, chaque administrateur est libre de ses choix
- Permet de connaître :
  - les numéros de machine à partir de leur nom et inversement
  - les machines gestionnaires de courrier électronique pour les domaines
  - et bien d'autres informations

DNS : notion de domaine

- Un domaine DNS peut être composé de plusieurs réseaux IP (identifiés par des numéros IP différents)
- Un domaine peut être subdivisé en sous domaines
- Les domaines forment une arborescence
- La racine de l'arborescence est subdivisée en «top level domains» : .edu, .org, .gov, .mil, .net, .com, .us pour les USA, .fr, .ca, .uk, .de, .jp, etc pour le reste du monde
- De nouveaux «top level» apparaissent ou vont apparaître : .int, ...



Soyez curieux...

Essayez `nslookup` sous Windows ou Unix/Linux.

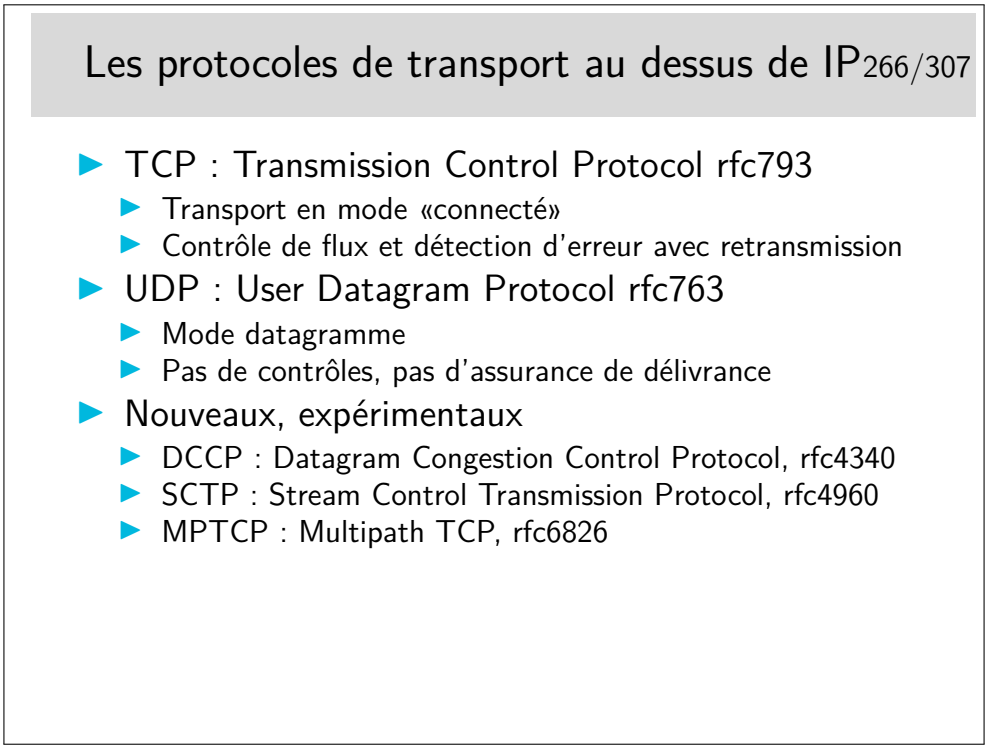
Sous Linux il y a aussi `host` tout simplement ou `dig` qui est beaucoup moins simple.

Le DNS peut faire l'objet d'attaques (certaines effectives, d'autres fantasmées...) : voir la présentation de Stéphane Bortzmeyer (Afnic) aux journées JCSA 2012 `http://www.afnic.fr/fr/l-afnic-en-bref/actualites/actualites-generales/6171/show/succes-pour-la-journee-du-conseil-scientifique-sous-le-signe-de-la-resilience-8.html`

200

Quatrième partie
Les protocoles TCP et UDP Quelques protocoles applicatifs

17 Introduction



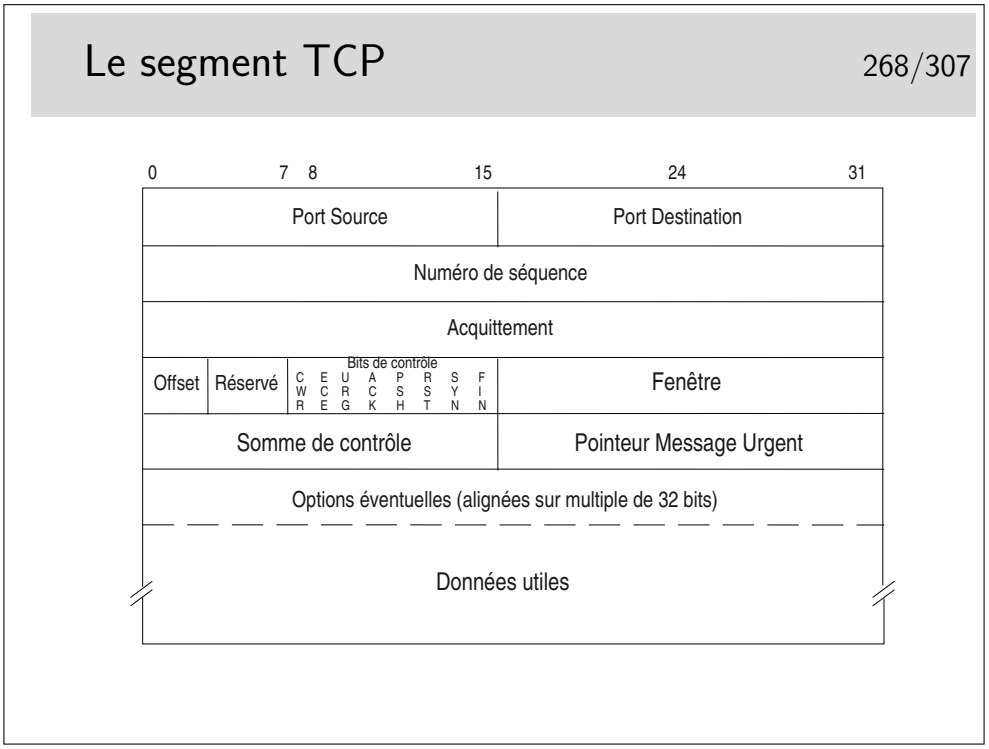
TCP et UDP, deux besoins extrèmes, mais relativement faciles à utiliser. De nombreux protocoles "universitaires" proposent des compromis, des services intermédiaires (p-ex. le protocole POC : Partial Order Connection), mais certains commencent à émerger sérieusement comme DCCP et SCTP.

DCCP : un genre d'UDP mais en mode connecté et avec contrôle de congestion. Pas de préservation de l'ordre des message ni de garantie de transmission des donnée, mais garantie sur les acquittement.

SCTP : un genre de TCP mais orienté *flux de messages* et non pas flux d'octets (flux : préserve l'ordre et guarantie la transmission) ; gère le multi-flux au sein d'une même connection, et le *multihoming* (annoncer que l'on va changer d'adresse IP pour la suite de la session).

MPTCP : compatible avec le TCP classique (programme avec des sockets TCP) ; définit des nouvelles *options* pour annoncer des sous-flux, en parallèle ou en secours, rejoindre une autre IP sans clore la session (*multihoming*), etc.

18 TCP



Ports source et destination : identifient les applications en relation, généralement l'une d'elle est serveur, son port correspond alors au numéro du protocole (exemple : 80 pour les serveurs Web)

Numéro de séquence : numéro du premier octet des données. C'est le rang du premier octet véhiculé par ce segment en comptant depuis le début de l'échange. Le numéro de début est tiré aléatoirement entre 0 et 232-1.

Acquittement : numéro du prochain octet attendu

Offset : indique la longueur de l'entête en mots de 32 bits (si égal à 5 alors pas d'option)

Bits de contrôle (CWR, ECE, Urg, Ack, Psh, Rst, Syn, Fin)

Fenêtre : permet le contrôle de flux, le récepteur indique avec ce champ combien d'octets il est prêt à recevoir (une valeur de 0 indique que ses tampons mémoire sont pleins)

Somme de contrôle : permet de savoir si le segment a été altéré ou non pendant sa transmission

Pointeur de données urgentes : indique l'emplacement de ces données dans le segment (si le bit Urg est positionné, sinon ce champ est ignoré, bien qu'il existe toujours)

Les bits de contrôle I

269/307

CWR	ECE	URG	ACK	PSH	RST	SYN	FIN

- ECE Explicit Congestion Notification-Echo (rfc 3168), l'entité TCP recevant un segment contenant ce bit à 1 doit réduire son débit
- CWR Congestion Window Reduced (rfc 3168) réponse à la réception du bit ECE pour indiquer que la requête a bien été prise en compte
- URG indique que le champ *Urgent Pointer* est significatif.
- ACK indique que le champ *Acknowledgement Number* est significatif.
- PSH fonction *PUSH*. Les données doivent être immédiatement remises à la couche supérieure.
- RST *reset*, la connexion est rompue.
- SYN synchronisation des indicateurs numéro de séquence. Segments d'initialisation de connexion.
- FIN terminaison de connexion.

Les bits de contrôle II

270/307

Exemple : segment IP et TCP dont le bit ACK est à 1

```
 0: 0800 2074 ef05 0800 0914 18e7 0800 4500
16: 0028 8954 0000 4006 db07 c02c 4b13 c02c
32: 4b08 1770 fdbe 0162 6e86 8e21 a873 5010 /* 0001 0000 */
48: 2210 bba3 0000 023a b3a1 1829
```

Les données «urgentes» 271/307

- Données appelées parfois "hors bande" ou *Out of Band* (OOB)
- Données à traiter en priorité par la couche réceptrice
- Elles sont véhiculées dans le flux normal en suivant le chemin normal. IP n'est pas sensible à ces données, leur caractère "urgent" est significatif seulement aux extrémités
- L'arrivée de ces données a un caractère aléatoire pour les applications destinatrices
- Les applications ne lisent pas ces données dans le flux normal
- Une application devant pouvoir accepter de telles données doit avertir le système pour que celui-ci lui envoie une interruption (un signal logiciel) afin qu'elle puisse traiter en priorité la donnée. L'application doit prévoir une routine spéciale de traitement pour la lecture de ces données
- Sémantique mal définie : le RFC6093 recommande aux nouvelles application ne ne plus l'utiliser...

Le bit de contrôle RST 272/307

- Utilisé par une entité TCP connectée avec une autre entité TCP distante pour avertir d'un problème
- Une application se terminant normalement fait une fermeture sur le port TCP utilisé (souvent une *socket*), ceci se concrétise par un échange de segments avec le bit FIN positionné et la connexion est rompue
- Si l'application se termine brutalement sans fermer la connexion, l'entité TCP associé envoie vers l'autre extrémité un segment avec le bit RST positionné. L'autre extrémité est ainsi prévenue de la terminaison de la connexion

TCP est orienté connexion 273/307

- La connexion TCP :
  - Relation établie point à point entre les deux extrémités
  - Normalement transparente aux routeurs (sauf si ceux-ci mettent en œuvre du filtrage où des mécanismes de QoS)
  - Caractérisée par un contexte dans les machines d'extrémité (numéros IP local et distant, ports local et distant)

La «connexion» TCP I 274/307

- La connexion est établie par un échange de paquets initiaux : le *three way handshake*
  - Ce n'est pas une connexion au sens strict du terme, il n'y a pas de chemin virtuel établi dans le réseau, les segments TCP sont portés par IP, protocole orienté datagramme
  - La connexion TCP se traduit par un contexte mémorisé dans les machines d'extrémité (le client et le serveur), ce contexte est le quintuplet :

```
[protocole, port local, port distant, @IP locale, @IP distante]
```

La «connexion» TCP II 275/307

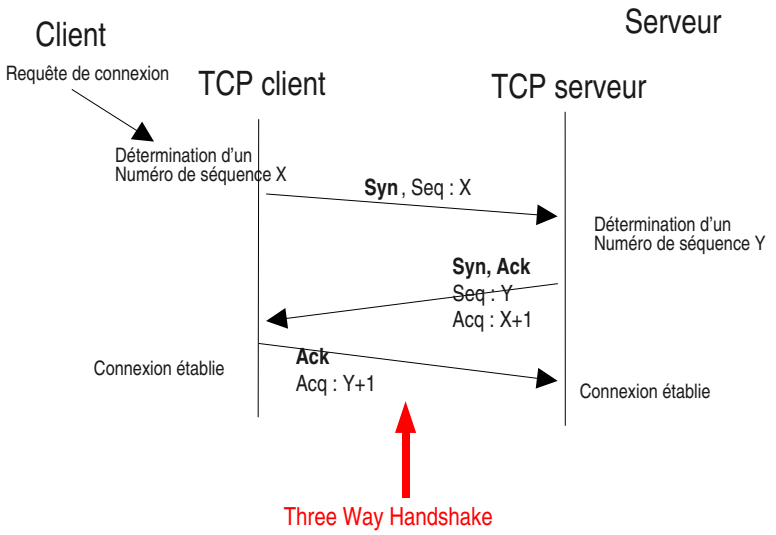
- Si les applications connectées n'ont rien à se dire, les entités TCP correspondantes ne s'envoient aucun segment, il n'y a plus alors de trafic
- Il est possible de faire en sorte que les entités s'échangent des segments d'alerte (sans données utiles puisqu'il n'y en a pas à envoyer) toutes les deux heures en positionnant une option dans TCP
- Si l'entité TCP distante ne répond pas au segment d'alerte ceci signifie que l'application associée s'est terminée sans prévenir ou que la machine s'est arrêtée. L'application locale sera prévenue lors de la prochaine lecture du port TCP

Cette option n'est pas une option au sens dans lequel ce mot sera vu plus loin. Il s'agit simplement d'un paramètre interne à l'entité TCP, positionné par l'application à l'aide d'une fonction spécifique (`setsockopt(SO_KEEPALIVE)`).

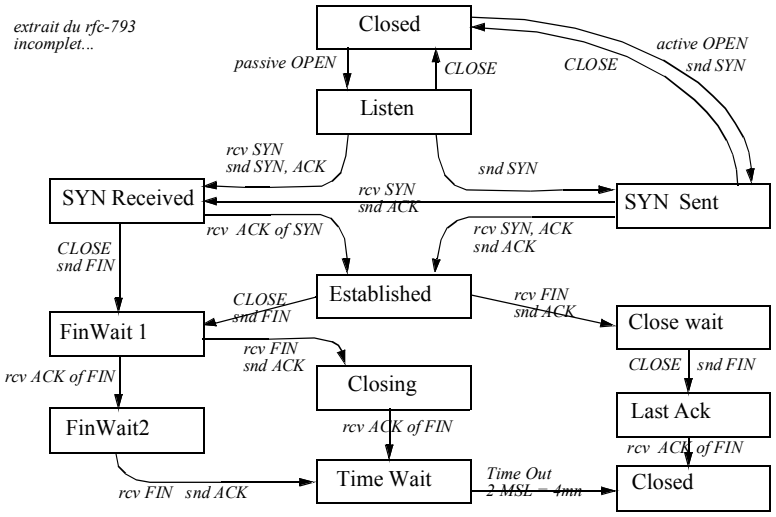
Connexions actives et passives 276/307

- Connexion passive
  - Ce n'est pas une connexion mais un point d'accès TCP ouvert par une application en fonction de serveur
  - Le contexte TCP créé attend une requête de connexion émanant d'un client
- Connexion active
  - La connexion réelle, initialisée par une application en mode client

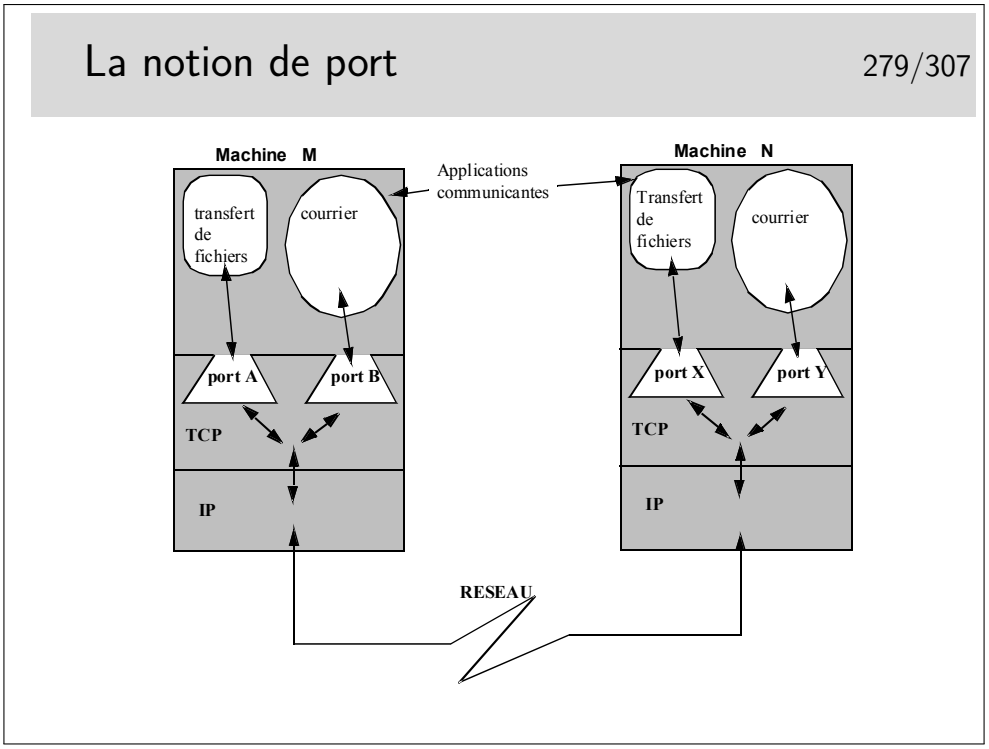
Établissement de la connexion 277/307



La machine d'états TCP 278/307



Ces états sont visualisables sous Windows dans une fenêtre de commande avec la commande `netstat -p tcp`. Sous Linux on fera `netstat -at`.



Les ports sont, en quelque sorte, les adresses des applications à l'intérieur des machines. Selon la terminologie OSI, un port est un SAP.

Quelques ports bien connus (voir aussi le fichier `/etc/services` sous Unix/Linux, ou `C:\WINNT\system32\drivers\etc\services` sous Windows)

21 : ftp (le port du canal de commande de ftp)

20 : ftp-data (le port pour la phase effective du transfert de fichiers par ftp)

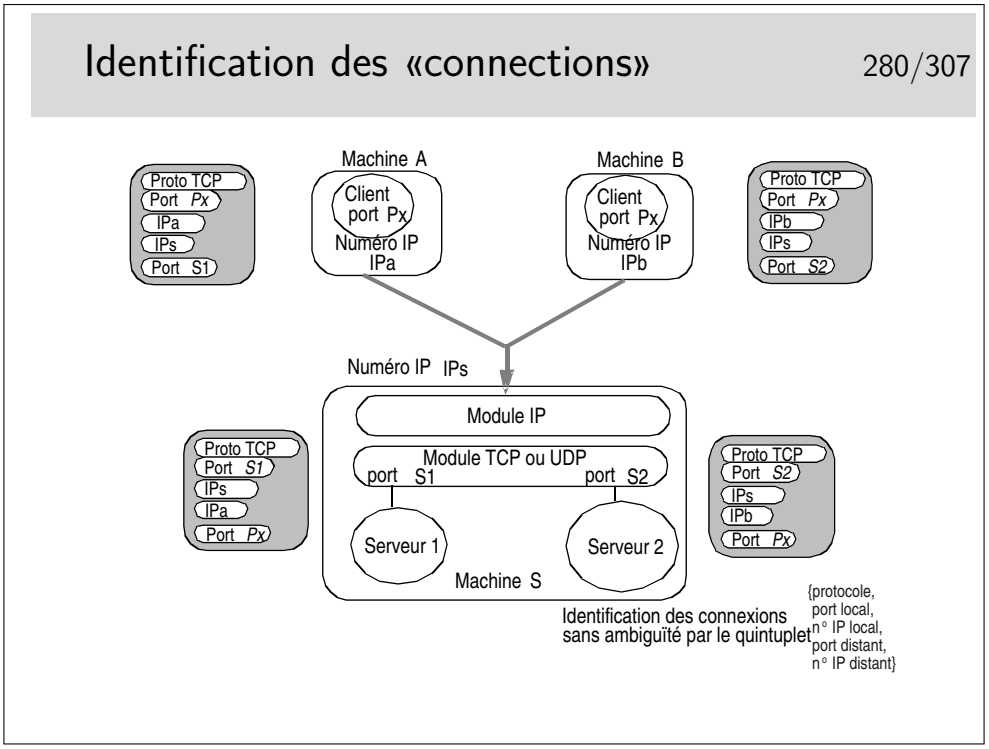
22 : ssh (les connexion à distance sécurisée par chiffrement)

23 : telnet : la même chose mais sans chiffrement

25 : le protocole d'envoie du courrier électronique Internet, SMTP

80 : le protocole http, donc le WEB

etc...



Pour résumer ce qu'est une «connexion TCP» :

— c'est une relation entre deux entités TCP résidant sur des machines différentes (ou entre deux contextes de la même couche TCP d'une machine où s'exécutent les applications en communication). En terminologie OSI, dans la couche Application on utilise le mot «association» plutôt que «connexion». Il serait plus approprié ici aussi.

— il n'y a pas de connexion associée dans le réseau

— dans chaque entité TCP en relation un contexte est créé et est associé à l'application ne cause de son coté

— ce contexte comprend :

  — une machine d'état finis gérant les états de la connexion

  — un quintuplet d'identification de connexion

  — des tampons mémoire d'émission et de réception

  — diverses variables complémentaires comme, entre-autre, le compteur associé à la gestion des congestions

— C'est l'application qui demande la création de ce contexte, à l'aide de fonctions d'une API spécifique telle que les Sockets (API issue du monde Unix, BSD au départ et ensuite reprise par tous les Unix et portée sous Windows)

Les options TCP I

- Caractéristiques
  - Le champ offset de l'entête étant codé sur 4 bits (il indique le nombre de mots de 32 bits de l'entête), la longueur max de l'entête est de 15 mots de 32 bits (15dec = 1111bin). L'entête minimum étant de 5 mots de 32 bits (20 octets), la longueur du champ option est de 10 mots de 32 bits max.

Les options TCP II

- Options standards
  - mss : *maximum segment size*, permet de négocier la taille maximale des segments TCP afin d'éviter des segmentations coûteuses. cette option est utilisées au moment de la connexion. Longueur 4 octets.
  - pas d'opération (NOP) : option nulle, permet d'aligner la prochaine option sur un début de mot de 32 bits. Plusieurs NOP peuvent se suivre si besoin. (lg : 1 octet)
  - fin de liste : permet d'aligner la fin des options sur un mot de 32 bits (lg 1 octet)

Les options TCP III 283/307

- Multiplicateur du champ fenêtre (window scale)
  - valeur permettant de multiplier la fenêtre par un multiple d'une puissance de 2 (jusqu'à 2^{14}) (rfc 1323). Ceci permet d'utiliser plus efficacement la bande passante pour des sources à haut débit et/ou des communications à grande distance en permettant un flux le plus continu possible
- Horodatage des données (rfc 1323)
  - L'option contient deux valeurs : un indicateur d'heure d'émission et un acquittement. Une source peut ainsi calculer le temps d'aller et retour sur un chemin en plaçant sa valeur d'horloge dans le premier champ. Lorsque l'acquittement du segment correspondant arrivera, cette valeur sera contenue dans le second champ de cette option. Il sera alors facile de retrancher cette valeur de la valeur courante de l'horloge.

Les options TCP IV 284/307

- Négociation de l'acquittement sélectif (rfc 2018)
- Acquittement sélectif : la première option permet d'indiquer qu'une source TCP est capable d'utiliser cette fonctionnalité, la seconde met en œuvre ce type d'acquittement permettant de demander uniquement la retransmission de données perdues en évitant de retransmettre des données suivantes bien reçues

Capture d'une séquence TCP 285/307

rfc-1323 et rfc 2018 (relevé avec tcpdump lors d'un début de requête web)

```
linux1.1082 > 206.132.41.203.www: S 2047350264:2047350264(0) win 16060 <mss
1460,sackOK,timestamp 43244276>
206.132.41.203.www > linux1.1082: S 2319802134:2319802134(0) ack 2047350265 win
32120 <mss 1460, sackOK, timestamp 190973015>
linux1.1082 > 206.132.41.203.www: . ack 2319802135 win 16060 <nop,nop,timestamp
43244311 190973015>
linux1.1082 > 206.132.41.203.www: P 2047350265:2047350896(631) ack 2319802135 win
16060 <nop, nop, timestamp 43244311 190973015>
206.132.41.203.www > linux1.1082: . ack 2047350896 win 31856 <nop, nop, timestamp
190973051 43244311>)
206.132.41.203.www > linux1.1082: P 2319802135:2319803583(1448) ack 2047350896 win
31856 <nop, nop, timestamp 190973063 43244311>
linux1.1082 > 206.132.41.203.www: . ack 2319803583 win 14612 <nop,nop,timestamp
43244367 190973063>
206.132.41.203.www > linux1.1082: P 2319803583:2319805031(1448) ack 2047350896 win
31856 <nop, nop, timestamp 190973063 43244311>
```

S : bit Syn

ack : bit ack

P : bit Push

. : pas de flag (autre que ack)

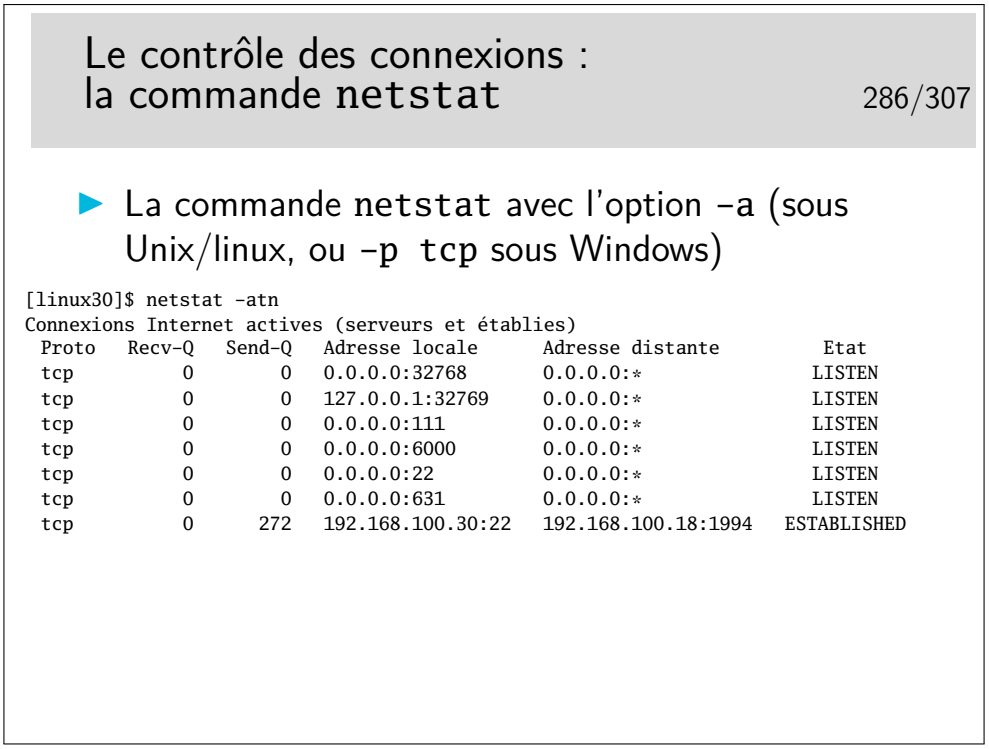
sackOK : acquittements sélectifs en fonction

mss : maximum segment size

en italique : complément d'information fourni par tcpdump mais ne figurant pas réellement dans le paquet

numéro de séquence du dernier octet, correspondant à l'acquittement alors attendu. Entre parenthèses : nombre d'octets du paquet

Les trois premières trames correspondent à une ouverture de connexion (Three Way Handshake)



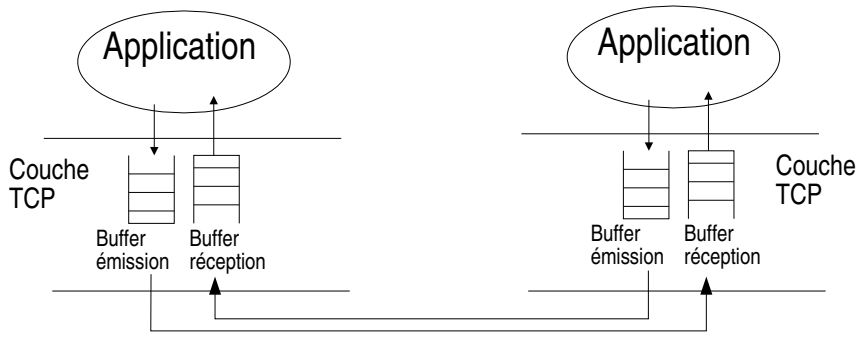
L'option `-n` de netstat permet de ne pas traduire les adresses et numéros de ports (les services). L'affichage est direct et on ne perd pas de temps en requêtes DNS pour afficher les noms des machines.

L'option `-t` restreint l'affichage au seul protocole TCP. Avec l'option `-u` on affiche les services UDP en cours (les applications serveur UDP). Il n'y a pas de connexion en UDP.

Avec la seule option `-a` on affiche toutes les connexions (ou serveurs lancées pour UDP), y compris les services n'utilisant que la communication Unix (par exemple le serveur d'affichage X-Window local et ses applications clientes).

Voir `-p tcp` ou `-p udp` sous Windows

TCP et les applications 287/307



Les données sont bufferisées et TCP les émet selon ses propres règles de contrôle de flux. Les applications n'ont pas le contrôle.

Selon la bibliothèque de développement (sockets sous Unix/Linux ou Windows par exemple) il peut être possible de dimensionner les tampons mémoire d'émission et de réception.

Le contrôle de flux TCP I 288/307

- Contrôle de flux d'extrémité à extrémité : le champ window
  - Les routeurs ne mémorisent pas "la connexion" TCP, ils ne peuvent donc pas participer au contrôle de flux. Celui-ci ne peut être réalisé que par les extrémités via le champ fenêtre.
  - Les entités TCP d'extrémités peuvent enregistrer les données reçues dans un tampon mémoire de taille limitée (8, 16, 32 Ko, ...)
  - Si l'application réceptrice ne lit pas suffisamment vite les octets s'accumulent dans le tampon mémoire de réception.

../..

Le contrôle de flux TCP II 289/307

../..
- L'entité TCP réceptrice envoie des acquittements avec une valeur de fenêtre qui diminue pour venir à 0 s'il le faut.
- 5s après avoir reçu un tel acquittement (`win=0`), l'entité émettrice teste le récepteur en lui envoyant un octet et continue ainsi en doublant l'intervalle (jusqu'à une borne de 1 min). Ce moyen permet de forcer le récepteur à renvoyer un acquittement et donner ainsi sa fenêtre. La valeur de 5s est en réalité variable selon les implémentations.
- Dès que les octets reçus par l'entité TCP réceptrice seront consommés par l'application réceptrice, la fenêtre pourra revenir à une valeur différente de 0.

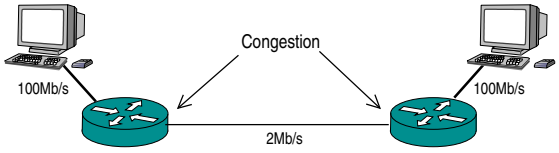
Sur Linux (noyau 2.6.xx), les paramètres liées à la gestion de la taille de la fenêtre de réception dans la pile TCP sont accessibles par la commande :

```
$ cat /proc/sys/net/ipv4/tcp_rmem
4096 87380 2076672
```

Les trois nombres donnent respectivement la valeur minimale (4096 octets), par défaut (87380 octets) et maximale (2076672 octets) de la fenêtre d'une session TCP.

Le problème de la congestion 290/307

- Il n'y a pas que les entités d'extrémités en jeu, il y a aussi le réseau. Le mécanisme de fenêtre ne permet pas d'adapter le contrôle de flux aux congestions du réseau.
- S'il y a congestion dans le réseau, des paquets peuvent être perdus, les retransmissions qui en découleront participeront au renforcement de la surcharge totale.
- Le congestion peut être due à des liens saturés mais aussi à des différences de débits entre segments de réseau.



Comment adapter TCP ?

Adaptation de TCP pour le contrôle de congestion I

291/307

- Détermination du RTT (*Round Trip Time*)
  - Avec les options spécifiques
  - Permet de «régler» un temporisateur de retransmission

Adaptation de TCP pour le contrôle de congestion II

292/307

- Le mécanisme du «*slow start*»
  - Une fenêtre de congestion est définie : `cwnd` (*congestion window*). On ne peut pas émettre plus que `cwnd` octets.
  - Juste après la connexion `cwnd` vaut 1mss (*max segment size*).
  - Un RTT plus tard, ce nombre double, et double à chaque RTT jusqu'à un seuil fixé à l'origine à 65535 (on ne peut cependant émettre que `min(cwnd, w)`, `w` étant la valeur de la fenêtre de l'entête TCP).
  - Lorsqu'une congestion a lieu, une perte se produit, le temporisateur expire et provoque une retransmission, le seuil est divisé par 2 et `cwnd` est remis à 1mss.
  - Si le seuil est dépassé, la progression de `cwnd` devient linéaire.

Les valeurs de RTT sont typiquement de :

10 ms entre sites intra Renater,

35 ms entre des sites intra Europe,

90 ms entre des sites situés en France et aux USA

210 ms entre des sites situés en Europe et en Asie

Adaptation de TCP pour le contrôle de congestion III 293/307

- Le mécanisme du «*fast recovery*» (entre autres) vient compléter le low start
  - Si on reçoit un même acquittement plusieurs fois, cela signifie deux choses : d'une part c'est qu'on a bien envoyé des segments et que ceux-ci ont été bien reçus (sinon on n'aurait pas reçu d'acquittement du tout) et d'autre part il doit y avoir un «trou» dans la séquence de segments transmis. Il suffit alors de ne retransmettre que le segment qui semble perdu.
  - Lorsque que plusieurs pertes successives ont lieu, ce mécanisme ne suffit plus car il ne permet que de récupérer le premier segment perdu. Il faut alors utiliser le mécanisme des acquittements sélectifs.

Comment éviter la segmentation avec TCP 294/307

- Le mécanisme du *Path MTU Discovery (rfc 1191)*
- le problème
  - la couche TCP émission connaît le MTU de l'interface IP de sortie (1500 par défaut sur Ethernet). Sur le chemin vers le récepteur un lien peut avoir un MTU inférieur (700 par exemple), le routeur amont sur ce lien devra segmenter.
- la solution
  - les routeurs sont interdits de segmentation TCP (bit D à 1). Si un paquet se présente de taille trop grande il est rejeté et le routeur qui le rejette émet un paquet ICMP vers l'émetteur. Ce paquet contient une information significative. Voir exemple :

Exemple de PATH MTU Discovery 295/307

Exemple : extrait d'un relevé avec tcpdump

- émission du premier paquet (seq = 1, taille 1448=MTU(1500)-TCPhead(20)-TCPoptions(12)-IPhead(20))

```
Em > Rec P 1:1449(1448) ack 1 win 16060 <nop,nop,timestamp 830140
827243>
```

- réception du mesage ICMP émis par un routeur intermédaire (celui qui devrait segmenter)

```
192.168.200.17 > Em: icmp: Rec unreachable - need to frag (mtu 700)
```

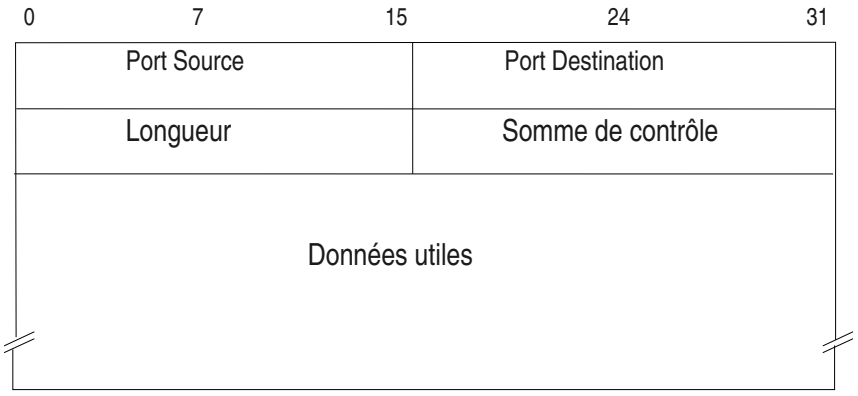
- On a bien compris... On émet des paquets de 648 octets (700-12-20-20=648)

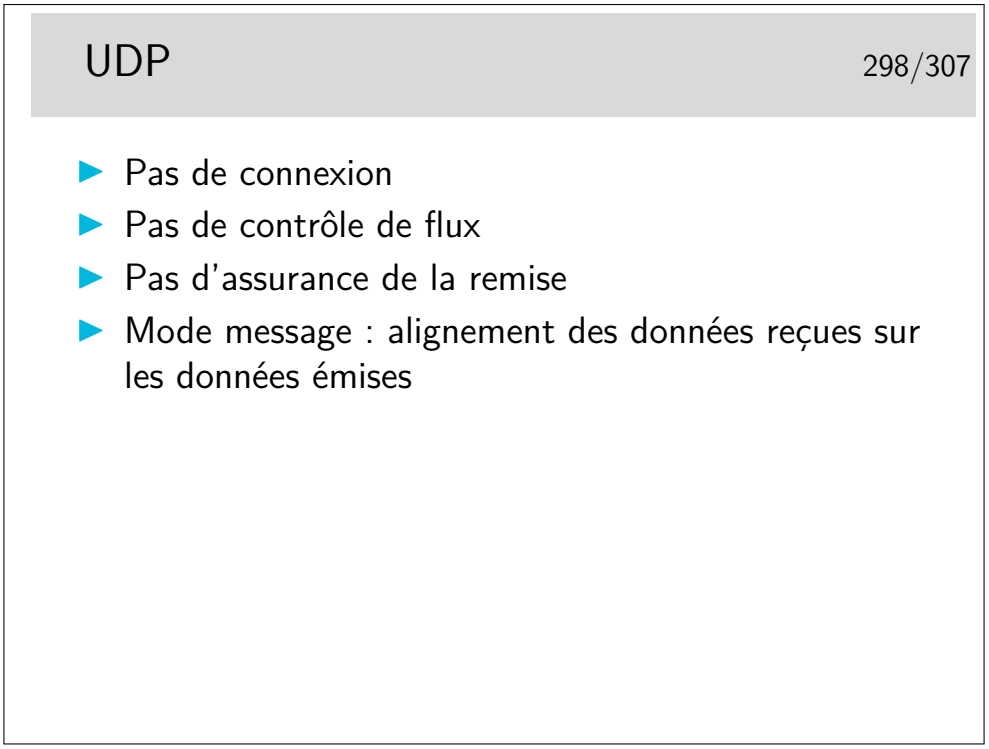
```
Em > Rec . 1:649(648) ack 1 win 16060 <nop,nop,timestamp 830140
827243>
```

En italique : ce que tcpdump nous indique concernant la longueur du SDU véhiculé.

19 UDP

Le datagramme UDP 297/307





UDP n'apporte que peu de choses par rapport à IP. Il garde de TCP la notion de port qui permet d'identifier les applications.

(A noter le champ longueur qui n'existe pas dans TCP)

Il n'y a pas de contrôle de flux, pas d'assurance de remise.

Il n'y a pas de «connexion».

On a cependant l'assurance que, si un datagramme arrive, il correspond très exactement à ce qui a été envoyé en ce qui concerne l'alignement des données. En d'autre termes, on peut dire que UDP est orienté «message». Si un message est reçu, le début correspond au début envoyé, la fin à la fin...

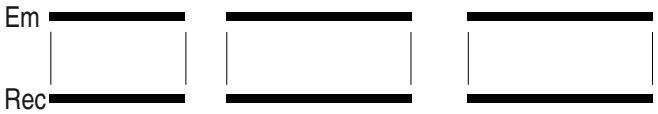
Ce n'est pas vrai en TCP. Lorsqu'une application reçoit un buffer de données via TCP elle ne peut être certaine que cette réception correspond bien à l'émission. Certes les données sont les mêmes, elles ne sont pas alignées. TCP est orienté «flot d'octets», il assure que les octets sont bien transmis.

TCP, UDP et l'alignement des données 299/307

- TCP est orienté «flot d'octets», il ne permet pas d'assurer le «cadrage» des données reçues sur celui des données émises

  Exemple de possibilité :

- UDP est orienté «Message»



TCP garantit que les octets sont bien transmis, il ne permet pas l'alignement de la réception sur l'émission. Autrement dit, un premier «message» peut voir son début partir dans un segment TCP et arriver dans un autre. Il faut pouvoir rassembler les octets appartenant à un message donné à l'arrivée. Pour cela on peut utiliser la technique TPKT recommandée dans le RFC 1006 qui consiste à commencer les messages par un octet de version (3), un octet inutilisé (sans doute pour faire joli, ou plus sûrement pour que l'ensemble soit aligné sur 4 octets), et deux octets de longueur de message. En réception il suffit de se synchroniser sur le premier message...

UDP ne garantit pas que les messages arrivent mais s'ils sont reçus, alors, on trouve tout d'un coup (à condition toutefois que la lecture prévoit de de lire un nombre suffisant d'octets).

20 Protocoles applicatifs

Les protocoles applicatifs I 301/307

- Mis en œuvre par les applications
- Des API existent
  - Sockets : le réseau est vu comme un fichier (on écrit et on lit le réseau comme un fichier), le protocole de communication est à implémenter «à la main»
  - RPC : l'API masque les aspects réseau, on appelle des procédure distantes de la même manière que des procédures locales
  - CORBA : *Common Request Broker Architecture*, extension aux applications réparties du concept «programmation objet»

Les protocoles applicatifs II 302/307

- Codage des données
  - Protocoles «texte» : smtp, http (1.x), sip, sdp
  - Protocoles codés :
    - ASN.1/BER/PER : H323, snmp
    - XDR : NFS, NIS
- Notions de «session»
  - p.ex. protocole sip, sdp, beep, et même les *cookies* sur le web, etc.

smtp : simple mail transfert protocol (rfc-821)

http : hyper text transfert protocol (le protocole du Web, rfc-1945)

sip : session initiation protocol (un des protocoles pour la téléphonie sur IP, rfc-2543)

sdp : session description protocol (descripteur de flux multimedia, inspirateur de sip : rfc-2327)

beep : blocks extensible exchange protocol, un framework pour créer des protocoles applicatifs, incluant des mécanismes d'authentification et de gestion de session, rfc 3080 3081 3117

ASN.1 : Abstract Notation, version 1 (on espère qu'il n'y aura pas de version 2, ou alors plus simple...) : langage de spécification de types de données. Comme un langage informatique ou on ne définirait que des types et des variables de ces types.

On lui associe une syntaxe de transfert (en clair : des mécanismes de codage en ligne) tels que le BER (Basic Encoding Rules) ou le PER (Packet encoding Rules).

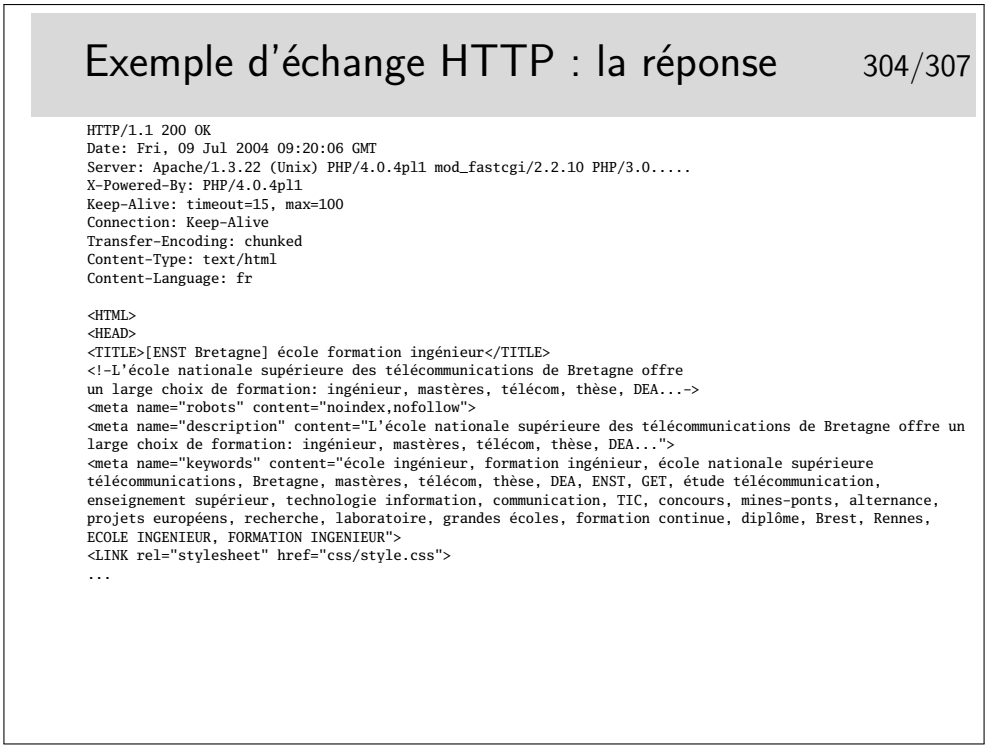
ASN.1/BER : utilisé pour snmp (simple network management protocol, rfc 1155 à 1158, 1212, 1213, etc)

ASN.1/PER est utilisé pour définir les données de signalisation pour la téléphonie sur IP dans l'architecture H323 (définition des PDUs H225 et H245, plusieurs centaines de pages, avec rien que du ASN.1 goûteux et savoureux).

On retrouve aussi ASN.1 dans les spécifications de la structure des certificats de sécurité X509.

Exemple d'échange HTTP : la requête 303/307

```
GET /index.fr.php HTTP/1.1
Host: www.enst-bretagne.fr
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.
Gecko/20030624 Netscape/7.1 (ax)
Accept: text/xml,application/xml,.....
Accept-Language: fr,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://www.enst-bretagne.fr/
```



Coté client, c'est au programmeur de l'application de préparer «à la main» les chaînes de caractères constituant les messages de requêtes du protocole. Coté serveur il faut savoir interpréter ces messages (le mot «savoir» étant traduisible en «implémentation de code»), il faut savoir aussi constituer les messages de réponse. Ces messages sont donc des suites de caractères ascii, placés dans des tampons mémoire qu'il suffit ensuite d'envoyer sur des sockets, coté émission. Coté réception il suffit de lire les sockets (comme on lirait des fichiers) et d'analyser en,suite ce qui est mémorisé dans les tampons de lecture.

Le langage est simple, il obéit à une grammaire (au sens informatique du terme) parfaitement spécifiée. Il est possible, à partir des spécifications de générer facilement des fonctions de traitement des messages. Il existe des outils adaptés aux traitements lexicaux, voire sémantiques, de telles grammaires : lex et yacc sous unix, flex et bison en logiciels libres.

Exemple d'échange smtp 305/307

(Exemple extrait du rfc-821)

```
S: MAIL FROM:<Smith@Alpha.ARPA>
R: 250 OK

S: RCPT TO:<Jones@Beta.ARPA>
R: 250 OK

S: RCPT TO:<Green@Beta.ARPA>
R: 550 No such user here

S: RCPT TO:<Brown@Beta.ARPA>
R: 250 OK

S: DATA
R: 354 Start mail input; end with <CRLF>.<CRLF>
S: Blah blah blah...
S: ...etc. etc. etc.
S: <CRLF>.<CRLF>
R: 250 OK
```

`S:` indique l'émetteur, `R:` indique le récepteur. (`S:` et `R:` ne font pas partie du protocole.)

`MAIL FROM:`, `RCPT TO:`, `DATA` sont des entêtes du protocole, de même que les séquences `<CRLF>` qui signifient simplement *retour à la ligne* pour `CR` (Carriage Return) et *saute de ligne* pour `LF` (Line Feed).

On peut noter que le cacatère `.` (point) est aussi partie intégrante du protocole.

Voici donc le protocole de base du courrier Internet si utilisé aujourd'hui. Ses spécification datent d'août 1982 !

Que pensez-vous des aspects sécurité d'un tel protocole ?

Fonctionnement du courrier électronique I 306/307

- Les utilisateurs ont une adresse de format standard
  - Par ex. :
    - `nom_utilisateur@nom-du-serveur.domaine`
  - ou plus concis :
    - `nom_utilisateur@domaine`
- Les outils mis en œuvre sont
  - Le client (MUA : *Mail User Agent* en langage OSI) : netscape, outlook, lotusNotes, etc...
    - Parle SMTP lors de l'envoi
    - Parle POP ou IMAP (sécurisé ou non) pour recevoir
  - Les serveurs (MTA : *Message Transfert Agent*) : ISS, sendmail, Postfix, etc...
    - Intermédiaires et final
  - Le DNS : requêtes MX (*Mail Exchanger*)

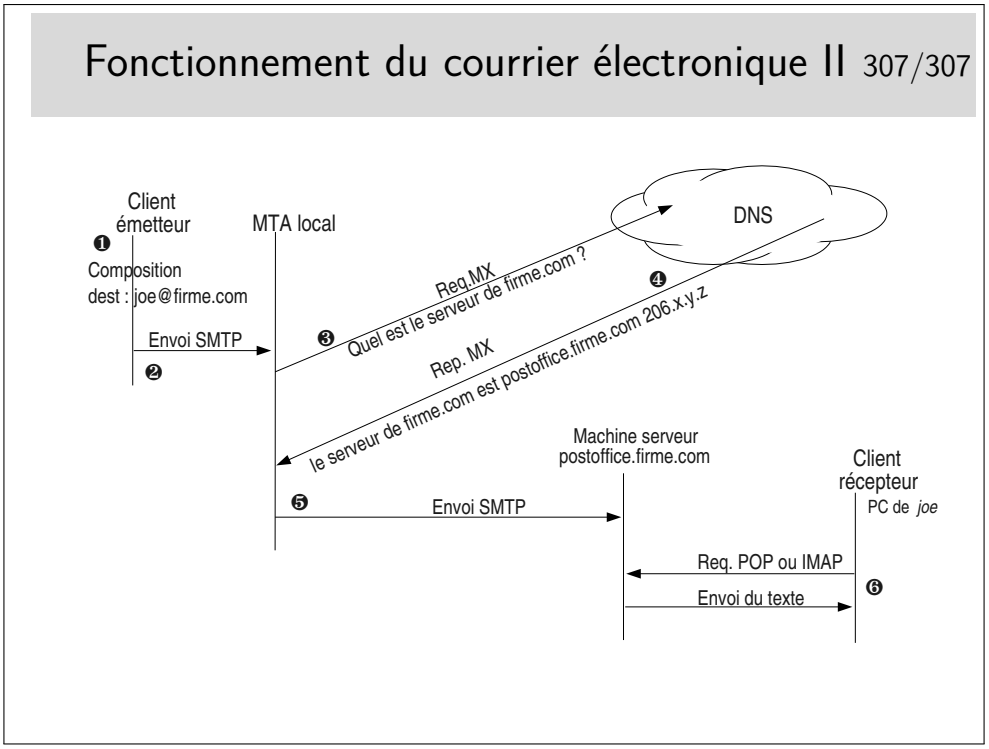
POP : Post Office Protocol

IMAP : Internet Message Access Protocol

Les protocoles POP et IMAP sont par défaut sans chiffrement et nécessitent l'échange de mots de passe utilisateur. Ces échanges se font par défaut en clair pour POP. IMAP supporte différents mécanismes d'authentification, notemment un système de challange dans lequel le mot de passe est chiffré (pour autant, les messages ne le sont pas...) Les outils clients ainsi que les serveurs peuvent être configurés pour fonctionner avec chiffrement (POPS ou IMAPS : POP/IMAP + SSL).

POP permet de télécharger sur son poste client les courriers reçus sur le serveur final. Les messages sont totalement transférés. Ils peuvent cependant rester sur le serveur.

IMAP permet de ne charger que les entêtes, il permet aussi de rechercher les messages par mot clé, les messages peuvent rester stockés sur le serveur, et de gérer des sous-répertoires (*folders*) sur le serveur.



Il peut y avoir plus de machines intermédiaires que ne le montre la figure ci-dessus. En particulier, coté réception, la machine serveur (appelée ici `postoffice.firme.com`) peut renvoyer vers un serveur interne déservant le département de l'utilisateur joe, ce denier paramètre alors son client pour qu'il se connecte par POP ou IMAP sur ce serveur particulier.

Il peut y avoir plusieurs serveurs d'arrivée, classés selon des priorités dans le DNS.

Il peut être intéressant de regarder l'entête complète des messages que l'on reçoit pour avoir la liste des MTA traversées par ces messages. Cette curiosité étant encore plus intéressante en cas de problèmes d'acheminement.

Les MTAs peuvent filtrer les messages, au départ comme à l'arrivée. Ils peuvent être munis de détecteurs de SPAMs et de virus.

Vous pouvez tester le DNS avec des requêtes MX de la manière suivante (sous Windows) :

```
c:\nslookup
> set type=MX
> microsoft.com
```

Ou sous Unix : `host -t MX microsoft.com` (la commande `nslookup`, bien qu'encore présente, est maintenant considérée comme rendue obsolète par la commande `host`).

TELECOM Bretagne

Atelier

Découverte du micro-ordinateur

Le PC et le réseau

Version 4.5

06/08/15

Table des matières

I Configuration matérielle sous Linux

☞ *Avant tout, vérifier votre PC, que tout est bien en place et bien branché ! Cependant, on n'utilisera que le disque SSD et on débranchera l'alimentation du disque HDD 7200rpm (c.à.d. le disque mécanique).*

☞ *Vous avez oublié de brancher le câble Ethernet. Branchez-le !*

Mettez votre PC sous tension et insérez-le CDROM Linux. Faites « booter » à partir de ce CD (voir le bios pour trouver la méthode de boot sur CD).

Note : Si, malgré vos efforts, le PC démarre sur Windows, laissez finir la phase de démarrage et arrêtez proprement le système avant de recommencer vos manipulations.

Un premier écran vous propose un menu de sélection d'opération. Sélectionner « `Live (amd64)` », et appuyez la touche `<Entrée>` .

Lorsque Linux est chargé, nous nous retrouvons avec un écran graphique de l'interface utilisateur GNOME. Cependant nous avons la possibilité de permuter en mode console texte : tapez `<Ctrl-Alt-F1>` (touches `Ctrl`, `Alt` et `F1` appuyées en même temps). La console de base du système apparaît, remplaçant la console graphique (qui n'est pas perdue). Revenez à l'écran graphique en tapant `<Alt-F7>`.

Nous aimerions connaître de nombreuses informations concernant notre machine et notre système. Beaucoup de celles-ci sont contenues dans des fichiers du répertoire `/proc`. Ouvrez une fenêtre terminal (menu `Applications -> Accessoires`). Notez que nous avons à notre disposition deux types de terminaux : le terminal pour simple utilisateur, et le terminal administrateur (pour lancer des commandes avec les droits d'administrateur). Dans un premier temps nous nous contenterons du terminal utilisateur.

Dans la fenêtre qui apparaît vous avez accès aux commandes standard de Linux (qui sont des commandes Unix),. Nous allons les utiliser car elles sont universelles Unix et ne dépendent pas des interfaces graphiques propres à chaque distribution Linux. Allez dans le répertoire `/proc` (commande `cd`) ; c'est un répertoire virtuel, qui n'existe pas sur le disque dur ou le CDROM, mais dont le contenu est généré à la volée par le système d'exploitation pour interagir avec le monde utilisateur. Listez les fichiers présents dans ce répertoire (`ls`). La plupart des fichiers ont, en principe, un nom assez explicite. Regardez le nom et le contenu de ces fichiers pour répondre aux questions suivantes (commande `cat` *`nom_fichier`*), et n'hésitez pas à faire preuve de curiosité :

- Infos version du noyau ?
- Infos cpu ? (modèle, constructeur, fréquence, nombre d'unités de calcul[1], ...)
- Infos mémoire ? (total, ...)

1 Il y a quelques subtilités à intuiter entre les paramètres `processor`, `physical ip`, `core ip`, `cpu cores`… Avez-vous deviné de quoi il est question ? Vous pouvez vous aider des commandes `lscpu` et `nproc`

L'utilitaire spécifique `lshw` nous liste la configuration matérielle de notre machine. (Cet utilitaire collecte les informations de nombreuses manières, `/proc` en est une, nous en verrons d'autres par la suite.)

Tapez la commande (en mode administrateur) : `sudo lshw-gtk`

Faites `Actualiser`. Note : un double-clic sur un élément des colonnes de gauche donne des détails dans les colonnes de droite...

Relevez :

- Type et nombre de processeurs (cœurs, threads) ?
- Taille des caches ?
- Type de carte vidéo ?

II Configuration de la carte réseau

II.1 La commande ifconfig

Cette commande permet de visualiser les interfaces réseau présentes sur votre machine (matérielles ou purement logicielles). Elle permet aussi de configurer l'interface en lui assignant une adresse IP, un netmask, etc.

Tapez la commande `/sbin/ifconfig -a` et regardez son résultat. Vous devez avoir deux interfaces sur votre configuration :

- `lo` : interface logicielle, implémentant une boucle locale (`lo` pour `loop` ou `local`)
- `eth0` : correspondant à l'interface physique

Répondez aux questions suivantes :

- L'interface `lo` est configurée, quels sont son adresse IP et le *netmask* associé ?
- Quelle est l'adresse MAC de la carte associée à `eth0` ?
- Quelle est l'adresse IP affectée à l'interface `eth0`, quel est le *netmask* associé ?

Remarque : si la machine est équipée de plusieurs cartes réseau, les interfaces Ethernet seront respectivement `eth0`, `eth1`, `eth2`, … et les interfaces WiFi `wlan0, wlan1,` ...

Le paramétrage de ces interfaces est généralement réalisé grâce à des outils graphiques et mémorisé dans des fichiers afin qui la machine puisse se configurer automatiquement au démarrage. Vous trouverez en annexe un récapitulatif des commandes réseau usuelles ainsi que des indications sur les fichiers permettant l'auto-configuration pour les distributions Linux standards.

II.2 Pilote (driver) et module noyau

L'interface `eth0`[2] est configurée... Comme par magie... En fait lorsque le système s'est chargé et initialisé, les interfaces matérielles ont été reconnues, les modules pilotes ont été chargés (les drivers). C'est une chance car il n'en est pas toujours de même avec Linux et parfois les matériels très récents ne sont pas reconnus.

Il serait intéressant que nous en sachions un peu plus sur notre interface, par exemple quel est son type, quel module noyau constitue son driver, etc. Voici quelques pistes pour y arriver.

Ouvrez un terminal et répondez aux questions suivantes :

- Quelle est le type de votre carte Ethernet (modèle, constructeur, etc.) ? Faites pour cela la commande `lspci -vv` (*very verbose*).

  Cette commande liste les périphériques présents sur le bus PCI (qu'il y ait ou non un driver dans l'OS pour les gérer).

On se demande maintenant quel est le pilote capable de prendre en charge cette carte ? Il s'agit d'un module du noyau, chargé au besoin, et figurant dans le système de fichier sous la forme d'un fichier binaire, exécutable par le noyau. Mais quel est ce fichier ? Nous allons enquêter de deux manières : (1) en cherchant par nous-mêmes à la main, puis (2) en s'appuyant sur le fait que le système l'a trouvé tout seul au démarrage (une chance !).

La carte a été détectée au démarrage et le driver a été chargé automatiquement. Regardons dans les *logs* du système ce qu'il s'est passé :

```
dmesg | grep eth0
```

La commande `dmesg` redonne les messages du noyau Linux, et `grep` filtre les lignes contenant `eth0.` (Mais lisez donc le `man` de ces commandes...)

- On peut également interroger l'interface `eth0` pour lui demander quel est son driver :

  `/sbin/ethtool -i eth0`

  Note : si le pilote est un peu vieux, cette commande ne renverra malheureusement rien.
- On peut aussi retrouver le nom du pilote grâce à la commande `lspci -vv` précédente.
- Cherchons un peu de documentation de notre pilote : `/sbin/modinfo` *`nom_du_pilote`*

  - Où réside ce module (fichier binaire) ?

  - Est-ce qu'il est compatible avec la version de notre noyau Linux ?
- Le module est-il chargé ? (Utilisez la commande `lsmod`)

2 Notez que l'on prononce `eth0` : « E – T – H – Zéro » (et non pas « éto », ou autre variante fleurie)

III Installation de Linux

Vous avez pu constater que le Linux chargé directement depuis le CDROM est déjà parfaitement utilisable. Cependant, pour une utilisation optimale du système il est préférable de l'installer sur une partition du disque dur.

III.1 Partitionnement

☞ *Rappel : l'installation se fera sur le disque SSD, le disque HDD 7200rpm doit être débranché. (Il serait tout à fait possible de gérer les deux disques en même temps mais cela requiert quelques précautions qui dépassent le cadre de ce TP.)*

Le disque dur de votre machine ne contient qu'une seule partition, dédiée à Windows. Celle-ci est formatée en système de fichier NTFS. Nous allons commencer par modifier ce partitionnement à l'aide de l'éditeur de partitionnement. Lancez cet outil dans le menu `Applications -> System Tools -> Administration -> Gparted Partition Editor` (il s'agit de la commande `gparted`)

Une fois l'examen des périphériques effectué, exécutez les opérations suivantes :

- Sélectionnez la partition unique `ntfs` puis `Redimensionner`
- Une nouvelle fenêtre apparaît qui vous permet de redimensionner à la main, de manière graphique : avec la souris attrapez le bord **droit** de la partition, puis diminuez la taille de moitié environ et validez. (Autre méthode : ajustez les valeurs numériques dans les boîtes de dialogue, mais surtout laissez à 0 la quantité d'`Espace libre précédent` (cela déplacerait la partition))

  Note : `gparted` refusera cette modification s'il a décelé un problème dans la partition NTFS, notamment un arrêt brutal de Windows (redémarrerez alors le PC sous Windows et arrêtez-le correctement).
- Sélectionnez la nouvelle partition de type `free ou non alloué`, cliquez-droit et sélectionnez `Nouveau`
- Gardez la taille maximale de la partition et sélectionnez `ext3` comme système de fichiers puis sélectionnez `Ajouter`

  Attention : Avant de poursuivre, vérifiez bien que vous ne déplacez pas la partition NTFS (il ne doit apparaître d'espace gris en tête de l'espace NTFS), l'opération prendrait plusieurs heures...
- Dans le menu `Édition` faites `Appliquez toutes les opérations`
- Vérifiez le nom de la partition que vous venez de créer
- Et quittez `gparted`

III.2 Installation

L'assistant d'installation est disponible ici : `Applications -> System Tools -> Install Debian wheezy`

- Choisissez le Français comme langue pour le système (la langue utilisée officiellement à l'école)
- Choisissez comme nom d'ordinateur `tp-`*`n`* où *n* est le numéro de votre PC (c'est inscrit dessus)
- Pour le nom de domaine, laissez ce que le système a découvert tout seul
- Pour le mot de passe de `root`, choisissez `azerty` (dans la vraie vie on choisirait évidemment un mot de passe plus robuste)
- Comme nom d'utilisateur nous choisirons `tp`, comme mot de passe `azerty`
- Choisissez le mode de partitionnement `Manual`
- Sélectionnez la partition prévue et modifiez celle-ci :
    1. Utilisez comme « `système de fichiers` `ext3` »
    2. Demandez qu'elle soit formatée,
    3. Choisissez le « `point de montage /` »
- Nous ne créerons pas de partition de partage (swap).
- Vous choisirez un « miroir » pour les paquets (le premier de la liste convient très bien).
- Il n'y a pas de proxy réseau (`mandataire`).
- Installez `Grub` sur le secteur d'amorçage du disque.

Vous pouvez lancer l'installation.

III.3 Vérification de l'installation

L'installation peut prendre plusieurs minutes, à son terme, rebootez la machine.
Au démarrage, le chargeur de système `grub` permet maintenant de choisir Linux ou Windows. Faites `<Entrée>` pour choisir directement Linux.

Une fois chargé et initialisé, le système vous propose une fenêtre de connexion. Connectez-vous sous le nom `tp` et donnez le mot de passe (azerty)

Lancez une fenêtre de type terminal.

Le résultat des opérations de partitionnement peut être visible grâce à la commande

```
cat /proc/partitions
```

- Quelles sont vos partitions et quelle est leur taille ?

TELECOM Bretagne	**Atelier découverte du micro-ordinateur** **Le PC et le réseau**	06/08/15, v4.5 page : 7/10

IV Caractérisation du réseau

En principe, le système s'est configuré pour utiliser la liaison Ethernet.

Vérifiez la configuration réseau : `ifconfig`, `route`, etc.

IV.1 Débit théorique

- Utilisez la commande `/sbin/ethtool eth0`
- Quel est le débit théorique sur notre lien Ethernet ?

IV.2 Mesures passives

Il s'agit ici d'effectuer des mesures sur le réseau sans perturber celui-ci, c-à-d. sans injecter de paquets, seulement en regardant ceux qui passent déjà.

- Utilisez la commande `/sbin/ifconfig eth0`

  Quels sont les compteurs qui vous paraissent intéressant pour apprécier la qualité du réseau ?

Le noyau Linux tient à jour un certain nombre de compteurs. Ils sont accessibles via des fichiers spéciaux (`/proc/net/dev /sys/class/net/eth0/` … utilisez la commande `cat` pour les afficher) ou des commandes systèmes (`netstat` … à essayer également)

- Une interface graphique pour afficher ces valeurs : menu `Applications -> Outils Systèmes -> Moniteur Système` ou parfois dans le menu `Applications -> Utilitaires -> Moniteur Système` (C'est la commande `gnome-system-monitor`)

  Note : s'il n'y a pas de trafic, il n'y aura pas grand-chose à afficher... Faites donc un peu de web ou de mail, mais pas longtemps hein !
- Relevez le débit annoncé (ordre de grandeur).
- Qu'est-ce qui est mesuré effectivement ? (La capacité réelle du lien ? Ce qui est passé dans la salle ? Juste votre PC ? Autre chose ?)

IV.3 Mesure active du débit

Il s'agit d'envoyer une grande quantité de données dans le réseau vers une autre machine, et de chronométrer. On peut faire l'expérience dans un sens, puis dans l'autre, de manière à mesurer le débit montant et le débit descendant. (Suivant les technologies réseau, ce n'est pas nécessairement symétrique. Mais dans le cas d'Ethernet ?)

Notez que si l'on fait ce type d'expérience sur un réseau de type bus, ou partagé, il convient de s'assurer que l'on est seul sur le réseau au moment de l'expérience pour que la mesure soit

pertinente... sans quoi on ne mesure pas le *débit total* du lien réseau (capacité), mais le *débit disponible*.

- Utilisez la commande `bw-client` (elle est pré-configurée pour faire une mesure vers le routeur de la salle, et attend son tour pour s'assurer qu'il n'y a qu'un seul client à la fois).

  Note : si cette commande n'est pas présente sur votre système, vous pouvez en récupérer une version à http://192.168.200.1/seance2/ Mais pensez à rendre le fichier exécutable ensuite (`chmod a+x bw-client`), et à l'appeler avec `./bw-client`

- Relevez le débit annoncé. Faites plusieurs essais pour évaluer la précision de la mesure.
- Est-ce cohérent avec le débit théorique du lien ?

IV.4 Mesures active de la latence

Il s'agit d'envoyer des paquets dans le réseau vers une autre machine, qui nous les renvoie aussitôt (protocole ICMP-Echo). On mesure ainsi le RTT (temps d'aller-retour) :

- Dans une fenêtre terminal faites un `ping` vers le routeur et surveillez la valeur du RTT.

  Note : pour l'arrêter, faites `Ctrl-C`

- Notez la valeur moyenne (`avg`) et la variance (`mdev`) de ce RTT pour l'Ethernet.
- Quel rapport voyez-vous entre le phénomène de latence que l'on évalue ainsi, et le débit que l'on a mesuré précédemment ?

IV.5 Estimation du débit

Une mesure directe du débit est assez invasive, typiquement par le téléchargement d'un gros fichier. On peut rechercher des méthodes moins invasives et perturbantes, des méthodes indirectes. On pré-suppose qu'il y a alors une corrélation entre le débit et un autre phénomène que l'on peut étudier. Il s'agit donc cette fois-ci non pas de mesurer directement le débit, mais de l'estimer en envoyant seulement quelques *paquets sonde*. Voyez l'annexe B pour des explications détaillées sur ces techniques.

IV.5.a Un estimateur qui étudie le temps d'aller-retour

Intuitivement : si l'on mesure le RTT de paquets de tailles différentes, le différentiel nous donnera une certaine estimation du débit total du lien. Cette technique, bien qu'active (envoi de paquets supplémentaires), est beaucoup moins invasive et contraignante qu'une mesure directe et effective du débit (elle peut être faite même si l'on n'est pas seul sur le réseau). De plus on mesure la capacité du lien, et non pas la bande passante disponible.

- Dans une fenêtre terminal faites :

```
sudo bing -c 1 -e 100 -s 18 -S 1472 localhost 192.168.200.1
```

Consultez la documentation pour comprendre ces paramètres (`man bing`)

- Notez les valeurs. Est-ce cohérent avec le débit théorique et les précédentes mesures de débit ? Que pensez-vous de cet estimateur ?

IV.5.b Un estimateur qui étude le temps inter-paquets

Intuitivement : lorsque l'on envoie des suites de paquets, à l'arrivée le temps qui les sépare est caractéristique du lien. Après de savants calculs statistiques, on peut en déduire une estimation du débit. Selon la façon de générer les paires de paquets, on peut caractériser soit la bande passante disponible, soit la capacité du lien. L'outil `pathrate` estime la capacité du lien.

- Dans une fenêtre terminal faites :

```
pathrate_rcv -Q -s 192.168.200.1
```

- Notez les valeurs. Est-ce cohérent avec le débit théorique et les précédentes mesures de débit ? Que pensez-vous de cet estimateur ?

Note : Pour cet test on se contente du test rapide (option `-Q` : *quick termination mode*), qui nous fournit un premier résultat en moins d'une minute. Le test plus complet, et donc plus précis, prend environ 18 mn.

V Configuration de l'interface WiFi sous Linux

Vous disposez d'une carte Wireless PCI Express.

Fermez vos sessions Linux et éteignez la machine. Débranchez le câble Ethernet. Insérez la carte WiFi dans le PC. Relancer le système et bootez sur Linux.

La machine possède maintenant 2 cartes réseau (Ethernet débranché et WiFi).

V.1 Installation du pilote (driver)

Vérifiez que la carte et le pilote sont bien installés avec les commandes :

`lspci | grep -i Wireless` (reprenez les commandes des sections II.1 et II.2).

Quel est le pilote correspondant à cette carte ?

Vérifier l'adresse MAC de la carte et notez le nom d'interface réseau.

V.2 Paramétrage

Le paramétrage WiFi est réalisé grâce au `network-manager`: lancez l'applet en cliquant sur l'icône en haut à droite représentant une prise réseau. Choisissez le réseau appelé « *`atelierpc`* ». La « *`clé wep`* » vous sera donnée au cours du TP.

Vérifiez le paramétrage WiFi avec la commande `/sbin/iwconfig` (`man iwconfig`) .

Relevez-le `essid`, le `bitrate`, ...

Vérifiez le paramétrage IP avec les commandes `/sbin/ifconfig` et `/sbin/route -n`

Note : en cas d'interférence avec la carte Ethernet, faites `sudo ifdown eth0`

V.3 Caractérisation du réseau WiFi

Refaites toutes les manipulations indiquées à la section « IV Caractérisation du réseau » page 7.

- Débit théorique, mesure du débit, mesure du RTT, estimation du débit, etc.

V.4 Commentez vos résultats

- Expliquez ce que représente le « débit » et ce que représente le « RTT ». Pour quels types d'utilisation sont-ils importants (vidéo, email, web, jeux en ligne, blog, VoIP, peer-to-peer, etc.)
- Comparez les performances d'une liaison Ethernet et d'une liaison WiFi.

Harmonisation Mastères

Atelier n° 2 – Le PC et le réseau

Annexes

A. Quelques notions préliminaires sur les concepts réseau

A.1. Les adresses Ethernet

Ethernet est la technologie prédominante aujourd'hui pour la liaison locale entre équipements. C'est un bus. Chaque équipement sur ce bus est identifié par une adresse dite MAC (*Media Access Control*), appelée en l'occurrence adresse Ethernet, ou aussi adresse physique. Cette adresse est unique, et est configurée par le constructeur de la carte Ethernet. A priori on ne la change pas, et pour communiquer les applications IP utilisent des adresses IP, dont la portée permet d'aller d'une liaison Ethernent à une autre liaison Ethernent en franchissant des routeurs IP.

Exemples : 90:2b:34:36:5a:c5 b0:48:7a:91:60:aa 74:86:7a:71:62:c4

A.2. Les adresses IP

Pour que votre PC puisse communiquer avec les autres machines du réseau ou avec l'Internet il doit posséder une **adresse** qui lui est propre. Le protocole de communication « réseau » utilisé est IP (Internet Protocol), l'adresse affectée à votre PC est donc appelée « **adresse IP** ».

Les adresses IP sont des nombres de 32 bits, donc 4 octets (en IPv4, et 128 bits en IPv6). Pour faciliter la lisibilité de ces adresses pour les utilisateurs elles sont représentées sous la forme de quatre nombres compris entre 0 et 255 (inclus), séparés par le caractère « . » (point).

Exemples : 192.44.75.10 192.168.100.2 172.16.0.4

A.3. Adresse Réseau, adresse machine, netmask

L'adresse affectée à une machine comporte deux parties : la première (à gauche) indique le numéro du réseau (l'adresse du réseau), la seconde (à droite) identifie la machine sur le réseau. Chaque partie comporte un certain nombre de bits. Historiquement ce nombre était fixé par la classe de l'adresse, mais est précisé par le *masque réseau*.

A.3.1. Les classes d'adresses

Il existe 5 classes comme l'indique le tableau suivant :

Nom	*Plage d'adresse Réseau*	*Nombre de machines par réseau*
A	0 à 127	2^{24} - 2
B	128.0 à 191.255	2^{16} - 2
C	192.0.0 à 223.255.255.0	254
D	224.0.0 à 239.255.255	Adresses réservées pour les applications multicast
E	240.0.0 à 255.255.255	Plage inutilisée

Une machine reliées à un réseau IP est munie d'au moins une adresse de classe A, B ou C.

Exemples :

192.44.75.10 : adresse de classe C, sur le réseau 192.44.75.0

172.16.10.5 : adresse de classe B, sur le réseau 172.16.0.0

Notez que les adresses réseaux sont identifiées par leur partie utile (suivant leur classe) suivie par des 0.

On ne donne pas l'adresse machine 255 car elle est réservée pour les messages en diffusion générale (broadcast). Pour la classe A, c'est l'adresse x.255.255.255, pour la classe B x.y.255.255. Pour la classe C il s'agit de x.y.z.255[1].

De même on n'utilise pas l'adresse 0 car elle fut, il y a longtemps, utilisée aussi pour le broadcast et peut-être subsiste-il des machines fonctionnant encore selon ce mode. Ceci explique pourquoi il n'y a par exemple que 254 adresses machines possibles pour un réseau de classe C.

Chaque machine fonctionnant sous IP est aussi munie d'une interface logicielle dite « boucle locale » permettant de réaliser des communications entre applications locales sans envoyer de paquets inutilement sur le réseau. Cette « boucle locale » est munie d'une adresse, toujours la même, à savoir 127.0.0.1 (réseau 127.0.0.0, masque 255.0.0.0, voir ci-après).

A.3.2. Le masque d'adresse ou *netmask*

La répartition des adresses en classe pose un problème de gaspillage d'adresses potentielles qu'il serait trop long d'expliquer ici (vous reverrez ces notions en cours de Réseaux). Les tables de routage dans les routeurs sont aussi affectées par cette classification. Afin de pallier ces problèmes, les membres actifs du développement de l'Internet (regroupés dans l'*Internet Engineering Task Force* ou IETF) ont décidé de briser le carcan des classes et de répartir la partie réseau et la partie machine des adresses de manière variable[2]. Il est alors nécessaire d'associer un indicateur complémentaire aux adresses : le masque.

Internet, comme son nom l'indique, est une interconnexion de réseaux. Mais comment savoir que deux adresses internet appartiennent au même réseau ? Il suffit de regarder le masque réseau ! Cette question, les routeurs et même les machines terminales se la posent à chaque instant, chaque fois qu'il y a des paquets à envoyer et que l'on a besoin de savoir quelle direction prendre. Deux adresses internet qui ont une même partie commune appartiennent au même réseau. Le masque réseau indique justement quelle est cette partie commune que l'on doit regarder.

Le masque est un ensemble de 32 bits avec la partie gauche, identifiant la partie réseau, à 1 et la partie droite (machine) à 0. Le masque est indiqué, de manière classique, en notation pointée comme pour les adresses, voyez les exemples suivants.

Pour des adresses standards, qui obéissent à la règle des classes, nous aurions par exemple

192.44.75.10 netmask 255.255.255.0 (une adresse de classe C)

172.16.45.78 netmask 255.255.0.0 (une adresse de classe B)

10.56.78.234 netmask 255.0.0.0 (une adresse de classe A)

On pourrait dire que les netmasks 255.0.0.0, 255.255.0.0 et 255.255.255.0 sont standards, respectivement pour les classes A, B et C.

Il est aussi possible de créer des sous-réseaux à partir d'une adresse réseau. On peut par

1 Il existe quelques autres subtilités sur ce point avec les possibilités de « subnetting » qu'il serait trop long de développer ici.

2 Spécifications dans le document RFC-1519 CIDR : Classless InterDomain Routing

exemple avoir une adresse de classe B comme celle-ci : 172.16.0.0, netmask 255.255.0.0, donc une adresse réseau standard. On peut alors étendre cette adresse réseau su 8 bits, pris dans la partie « adresse machine ». On peut alors avoir les réseaux 172.16.10.0, 172.16.20.0, etc, avec pour netmask 255.255.255.0.

Le netmask peut aussi être indiqué directement par le nombre de bits à 1. Par exemple le netmask 255.255.255.0 peut être indiqué par le nombre 24. On peut ainsi écrire 192.168.100.0/24 pour indiquer une adresse de réseau.

Sous Unix/Linux les deux écritures peuvent coexister dans les commandes en ligne. Cependant les interfaces graphiques de configuration demandent la notation de type 255.255.255.0 (sous Windows également).

A.3.3. Adresse et nom de machine

Vous ne devez pas ignorer l'existence du serveur web de l'école : www.telecom-bretagne.eu. Vous devez vous douter que derrière ce nom se cache une machine. Mais le nom de cette machine www.telecom-bretagne.eu doit bien correspondre à une adresse IP, sans quoi cette machine serait inaccessible par Internet (rien n'est magique, soyez-en certain). Cette adresse (au moins en 2013 il en était ainsi) est 192.108.117.241 (une classe C si le netmask est standard).

Il est plus facile de se souvenir du nom de la machine plutôt que de son adresse IP. Il existe un service, de portée mondiale, appelé DNS (Domain Name Service) qui gère l'association entre les noms et les adresses réelles. Ce service est abrité sur des serveurs gérés par les entreprises (ou dont la gestion est déléguée aux FAI). Les serveurs se connaissent entre eux dans une arborescence mondiale[3].

La commande *nslookup* sous Windows ou *host* sous Unix/Linux permet de consulter le DNS et d'afficher la correspondance entre une adresse et un nom en lui fournissant l'un ou l'autre.

Sous Unix/Linux il existe un mécanisme plus simple : le fichier /etc/hosts (parfois centralisé via le service réseau NIS : Network Information Service). Une machine Unix/Linux peut utiliser conjointement le fichier /etc/host (avec ou sans NIS) et le service DNS.

A.3.4. Attribution des adresses

On ne peut pas attribuer au hasard une adresse IP à une machine. Surtout si on est relié à l'Internet. Le fournisseur d'accès (FAI) doit fournir un préfixe (une adresse réseau) et un netmask à l'entreprise qui désire être raccordée à Internet.

Pour des machines personnelles, chez soi par exemple, le FAI attribue dynamiquement une adresse lors de la connexion de la machine via un modem téléphonique ou ADSL.

Un réseau local d'entreprise est relié à l'Internet via un routeur d'accès qui possède une interface raccordée au FAI et au moins une interface coté intérieur de l'entreprise. Le gestionnaire du réseau de l'entreprise définit son plan d'adressage et fournit aux utilisateurs l'adresse de leur machine et le netmask associé, ainsi que l'adresse machine du routeur (pour que chaque machine puisse accéder à l'Internet).

Un utilisateur et administrateur de sa machine doit alors configurer son interface réseau pour lui affecter l'adresse et le masque **indiqués par le gestionnaire du réseau**.

En aucun cas l'utilisateur ne doit attribuer une adresse de son cru, il risquerait d'y avoir conflit avec une autre machine. Pour pallier ce genre de problème il existe une méthode d'attribution dynamique et contrôlée car centralisée : la méthode **DHCP** (Dynamic Host Control Protocol) gérée par le gestionnaire du réseau. L'utilisateur/administrateur de sa machine ne précise rien d'autre que

3 Sans gouvernance réelle. Les serveurs doivent simplement utiliser le protocole de communication ad-hoc. Leur gestion n'est pas du tout supervisée par une autorité supérieure.

« Obtenir une adresse automatiquement » (selon le système d'exploitation) et le tour est joué.

A.3.5. Les adresses privées

L'Internet manque d'adresses. Depuis longtemps il n'y a plus d'adresses de classe A et B attribuées. Les adresses disponibles en classe C ont disparues en avril 2011. Une nouvelle version du protocole IP est en attente de déploiement (IP version 6 ou IPv6, la version actuelle de IP étant IPv4). Elle offre des adresses sur 16 octets. Son déploiement effectif tarde... Pour pallier le déficit d'adresses IPv4, l'IETF avait décidé la création de plages d'adresses privées (pouvant être attribuées plusieurs fois à certaines conditions). Ces plages sont les suivantes :

10.0.0.0 à 10.255.255.255, masque à partir de 255.0.0.0, pour la classe A

172.16.0.0 à 172.16.255.255, masque à partir de 255.255.0.0 pour la classe B

192.168.0.0 à 192.168.255.255, masque à partir de 255.255.0.0 pour la classe C

Ces adresses ne peuvent pas être routées dans Internet, elles peuvent l'être à l'intérieur des réseaux privés des entreprises ou des particuliers.

Si malgré tout un réseau privé doit être raccordé à Internet il est nécessaire que le routeur de raccordement soit muni d'une adresse officielle coté Internet et qu'il mette en œuvre la fonction **NAT** (Network Address Translation).

A.4. Les organes du réseau

Les machines des utilisateurs sont reliées à des organes actifs appelés hubs ou commutateurs (on dit encore switches). Ces organes pouvant être à leur tour reliés entre eux pour former une architecture physique plus vaste et couvrir ainsi l'ensemble de l'entreprise.

La technologie employée s'appelle Ethernet. Elle est apparue au tout début des années 80 (concepts définis en 1976). Elle a supplanté toutes ses concurrentes.

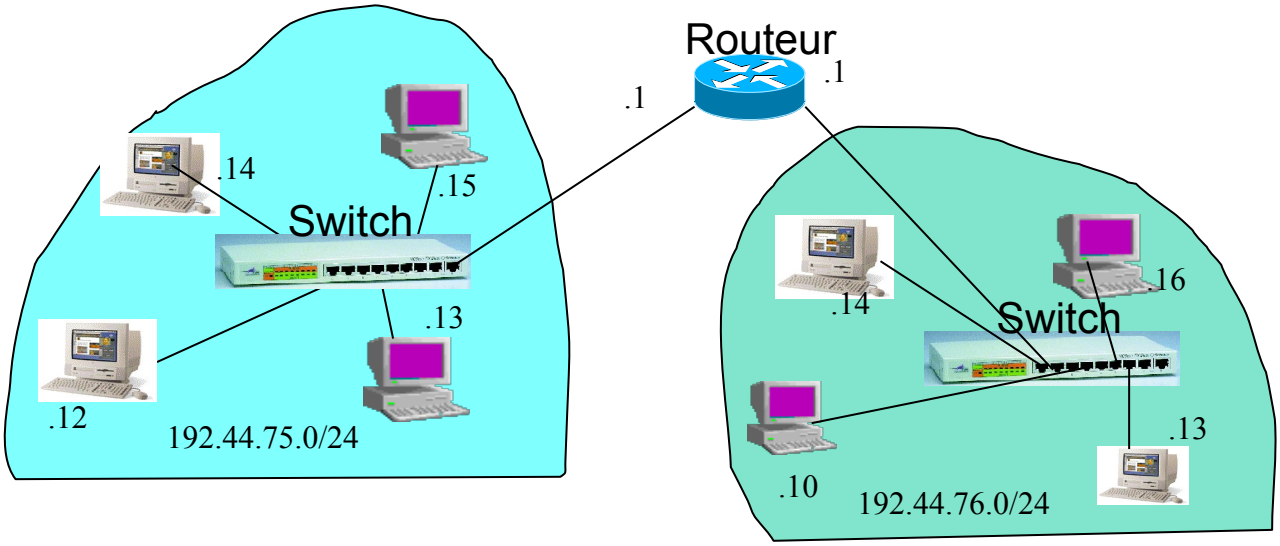
Un réseau ainsi constitué va être muni d'un adressage IP constitué d'un seul préfixe, par exemple 192.168.100.0, netmask 255.255.255.0.

Pour des raisons diverses l'entreprise crée généralement plusieurs réseaux séparés par des moyens physiques ou logiciels. Chacun de ces réseaux est muni d'un préfixe différent. Il ne peut pas y avoir de communication entre eux directement. Pour que les communications soient possibles entre ces réseaux il est nécessaires qu'ils soient chacun raccordés à un organe appelé « routeur ».

La figure suivante présente une vue simplifiée d'une partie du réseau de l'école.

Questions :

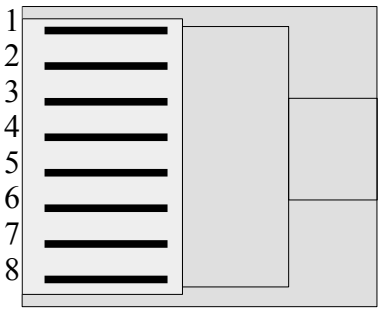
Indiquez les adresses complètes de chacune des machines ainsi que de chaque interface du routeur sous la forme `A.B.C.D - netmask x.x.x.x` ou x vaut 255 ou 0.



A.5. Le câblage

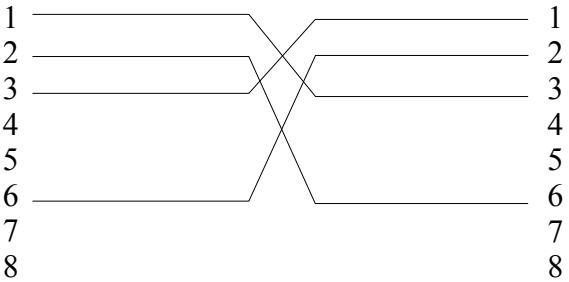
Les câbles sont généralement en cuivre. Ils ont une longueur inférieure à 100m et le débit supporté est typiquement de 100Mb/s (historiquement du 10Mb/s, et maintenant du 1Gb/s).

Le schéma suivant fournit des indications sur le type de prise utilisé ainsi que sur le brochage



1 T+ Blanc orange
2 T- Orange
3 R+ Blanc Vert
4
5
6 R- Vert
7
8

Il est possible de relier des machines directement entre elles, sans l'intermédiaire de hub ou de switch. Il faut alors utiliser un câble dit « croisé », on comprendra pourquoi en regardant le schéma suivant :



B. Notions de métrologie réseau

...et les limites du recours à un estimateur...

Pour qualifier une liaison réseau, il convient d'évaluer un certain nombre de choses : capacité du lien, bande passante disponible, latence, gigue, taux de pertes, etc. Ces mots ne vous sont pas inconnus, mais concrètement, comment on fait pour savoir ?

● Les techniques de *mesure passive* consistent à regarder ce qu'il advient pour les paquets circulant déjà dans le réseau et générés les applications en cours. L'avantage est que l'on ne perturbe pas le réseau que l'on mesure. L'inconvénient est que l'on ne sait pas trop si les paramètres que l'on observe sont ceux inhérents au lien réseau ou bien aux applications en cours...

● Les techniques de *mesure active* consistent au contraire à regarder ce qu'il se passe pour des paquets que l'on injecte exprès dans le réseau. L'avantage est que l'on identifie mieux le phénomène que l'on observe (puisqu'on l'a provoqué exprès), mais que du coup cela perturbe le lien réseau que l'on veut observer...

L'outil *ping* fait de la mesure active (chronomètre le temps d'aller-retour sur un paquet qu'il a envoyé). Cependant les paquets sont peu nombreux, et par défaut de petite taille. C'est un compromis acceptable. C'est aujourd'hui l'outil de prédilection pour évaluer la latence, la gigue, le taux de pertes.

Pour évaluer la bande passante, la stratégie la plus simple consiste à télécharger un gros fichier et à chronométrer. On est indéniablement dans de la mesure active, et qui plus est un peu perturbante pour le réseau observé. Mais c'est efficace. Cependant, cela ne mesure que la bande passante *restante*. Si l'on veut mesurer la *capacité totale* du lien, il faut s'assurer que l'on est tout seul sur le réseau au moment de la mesure.

● Une autre façon de faire consiste à avoir recours à un *estimateur*. On cherche à faire le même compromis que pour ping : injecter des paquets, certes, mais en petit nombre pour ne pas perturber le réseau, et de regarder ce qu'il advient de ces paquets sonde pour en déduire une estimation du débit.

▪ Une première stratégie consiste à regarder le délai sur des paquets de taille variable. (On peut citer par exemple les outils `bing`, `pchar`, etc.)[4] L'idée est la suivante : considérons la fonction $f(t)$ qui représente la quantité de données transmises sur le lien au cours du temps. En première approximation, f est une droite. Le débit du lien est simplement la pente $d=f'$. La latence l sur le lien est définie par $f(l)=0$. Comme avec ping, on mesure le temps d'aller-retour sur quelques paquets seulement, mais de taille différente. Le débit du lien est donc simplement le différentiel sur deux mesures, a priori. Et s'il y a d'autres applications réseau en parallèle, cela n'a pas d'impact tant que nos paquets sonde peuvent passer. Bien entendu, cela ne donne qu'une estimation, reste à savoir dans quels cas elle est pertinente.

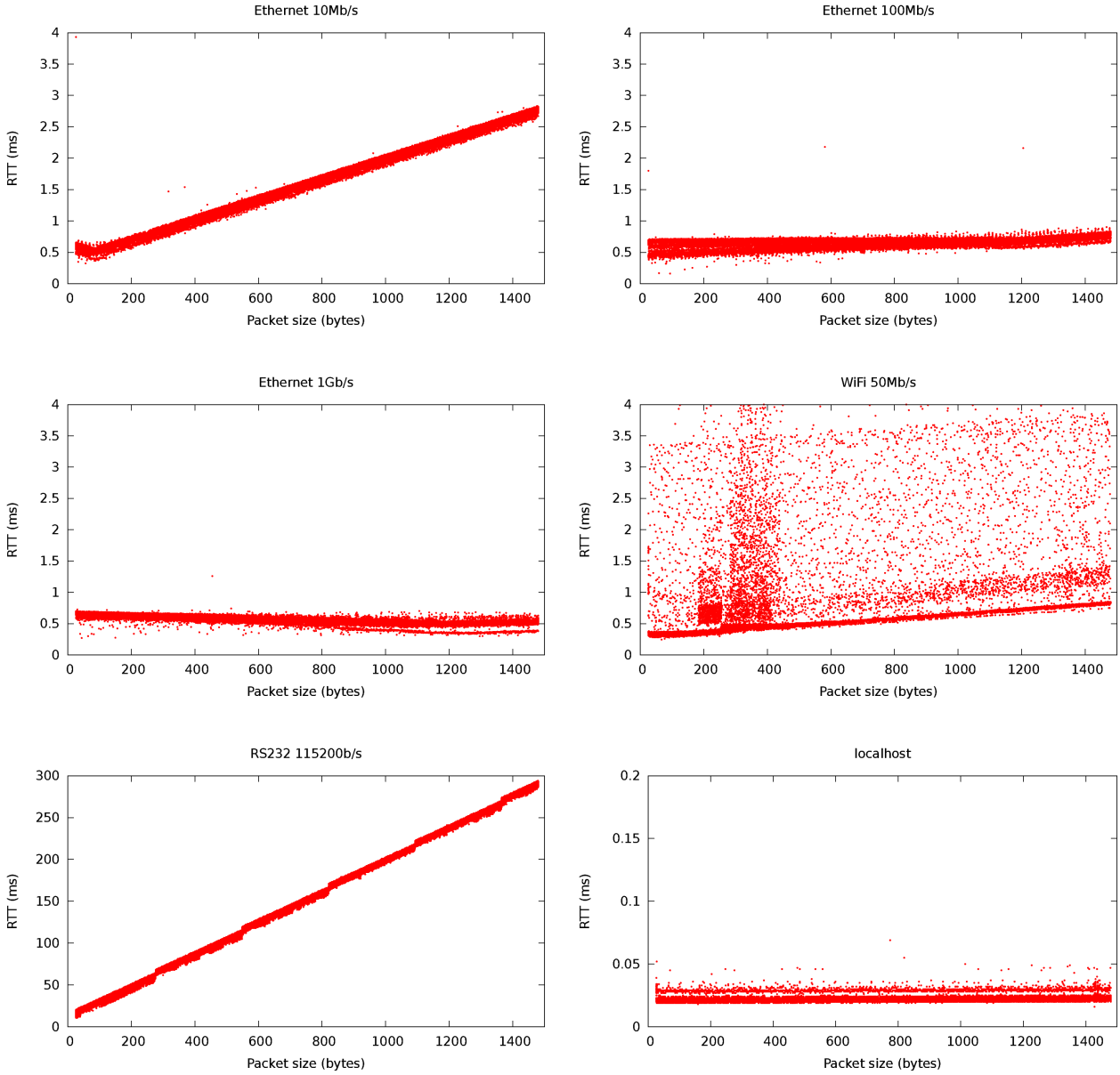
Les figures suivantes représentent des mesures de ping sur des liaisons de nature différentes entre deux PC, et en fonction de la taille des paquets (compter 24h pour obtenir une courbe à peu près propre). En abscisse la taille des paquets variant de 18 à 1472 octets (tiens, pourquoi ces valeurs ?). En ordonnée le temps en ms (attention à l'échelle d'une courbe à l'autre). Notez tout de même qu'une mesure de ping est un temps aller-retour pour une taille donnée. C'est donc le double du temps de transfert, et la fonction f précédente n'est pas le temps de transfert par quantité de données, mais la quantité de données transférées par unité de temps. Un ping est très facile à obtenir concrètement, mais ce n'est pas la courbe $f(t)$ précédente ; c'est plutôt $ping(data)=2*f^{-1}(data)$. Plus la courbe est « horizontale » plus le débit est élevé. Plus la courbe est « haute », plus la latence est

4 Outils de métrologie ; http://www.caida.org/tools/taxonomy/perftaxonomy.xml http://www.icir.org/models/tools.html
Quelques études scientifiques comparatives : http://www.caida.org/research/performance/bandwidth/

importante. Plus le trait est « épais », plus la gigue est importante (dispersion autour d'une valeur nominale).

Ces figures donnent la caractérisation de liens Ethernet 10Mb/s, 100Mb/s, 1Gb/s. On a aussi la caractérisation d'une liaison WiFi 54Mb/s. Vous constaterez que la courbe est assez similaire à celle de la liaison Ethernet 100Mb/s, mais que la dispersion des mesures autour de la courbe nominale est très importante. Il y a beaucoup plus de gigue en radio qu'en filaire, même à débit comparable. Pour confirmer cette hypothèse, on a également mesuré une liaison de type RS232 à 115200b/s qui a un débit très faible, mais avec une dispersion moindre comparativement. Finalement, voyant une telle dispersion en WiFi, on peut se demander s'il est bien judicieux de modéliser la fonction de transfert par une simple droite.

À titre de comparaison on a aussi caractérisé la liaison `localhost`, c'est-à-dire l'interface virtuelle locale d'un PC. Cela permet d'estimer en quelque sorte l'overhead de la couche réseau du système d'exploitation.



Mais revenons à notre hypothèse de départ pour notre estimateur : le phénomène à observer peut être approximé par une droite. Mais est-ce vraiment le cas ? Pour s'en assurer, on a tenté une régression linéaire simple sur les nuages de point précédents. Le tableau ci-après donne les paramètres ainsi calculés.

	100Mb	10Mb	1Gb	WiFi	RS232	localhost
Pente	0,0001091097	0,0016065393	-0,000119068	0,0001195564	0,1888760777	1,002896421E-006
Ordonnée	0,5685228356	0,3910439754	0,6567719657	0,8118460001	10,3418502878	0,0218204204
R^2	0,3128210787	0,9961655507	0,6374526684	0,0003175807	0,999275246	0,0759206327

Observons plus particulièrement le coefficient de détermination R^2. (Rappel : plus ce coefficient est proche de 1, plus le nuage de points se rapproche de la droite approximée ; une valeur faible indique au contraire que les points sont très dispersés autour de la droite sensée les modéliser.) Comme on l'avait deviné précédemment, R^2 est particulièrement mauvais pour un lien WiFi. Mais ce qui peut être plus fâcheu est qu'il n'est pas si bon que ça pour les liens 100Mb et 1Gb : le débit est tel que le temps mesuré est trop court et se trouve finalement noyé dans la gigue.

En conclusion, cet estimateur qui reste une bonne idée pour les liaisons bas débit est malheureusement peu représentatif pour les liens haut débit que l'on connaît aujourd'hui... Il faudra chercher une autre idée...

▪ Une seconde stratégie consiste à regarder le temps inter-paquets, lorsque l'on envoie des paires de paquets ou des trains de paquets. (On peut citer par exemple les outils `pathload`, `pathrate`, `SProbe`, `scriptroute`, etc.) L'idée est la suivante : en première approximation, le temps qui sépare deux paquets successifs de taille identique est déterminé par la taille d'un paquet et du débit du lien, auquel s'ajoute le silence entre trames imposé par la technologie du lien, plus l'impact des files d'attentes et des stratégies d'accès au média de la couche liaison. Bref, le délai inter-paquet est caractéristique du lien. C'est un peu plus compliqué que cela car on constate que sur des trains de paquets, le temps inter-paquets augmente progressivement puis se stabilise. Toujours est-il que ce temps inter-paquets est quelque chose d'intéressant à regarder. Avec l'aide d'une étude statistique, on arrive à en déduire une estimation du débit [1-5].

Ces estimateurs basés sur la dispersion temporelle inter-paquets sont plus complexes que les précédents basés simplement sur le RTT. Par contre ils offrent une meilleure stabilité. Un bémol tout de même : s'ils ne sont tout de même pas aussi invasifs qu'une mesure directe (télécharger un gros fichier et chronométrer), le train de paquets à envoyer peut être non négligeable (p.ex. Comptez environ 10Mb pour l'outil pathload).

[1] M. Jain and C. Dovrolis, *Pathload: A measurement tool for end-to-end available bandwidth*, PAM 2002. http://www.cc.gatech.edu/~dovrolis/Papers/pam02.ps

[2] M. Jain and C. Dovrolis, *End-to-End Available Bandwidth: Measurement methodology, Dynamics, and Relation with TCP Throughput*, ACM SIGCOMM 2002. http://www.cc.gatech.edu/~dovrolis/Papers/sigcomm02.ps.gz

[3] C. Dovrolis, P. Ramanathan, D.Mo*ore, What do packet dispersion techniques me*asure?, IEEE Infocom 2001, http://www.cc.gatech.edu/~dovrolis/Papers/infocom01.ps

[4] C. Dovrolis, P. Ramanathan, and D. Moore, *Packet Dispersion Techniques and Capacity Estimation*, IEEE/ACM Transactions on Networking 2004. http://www.cc.gatech.edu/~dovrolis/Papers/ton_dispersion.ps

[5] N. Spring, D. Wetherall, and T. Anderson, Scriptroute: *A facility for distributed Internet measurement*, North American Network Operator's Group 202 (NANOG 26). http://www.cs.umd.edu/~nspring/talks/nanog-scriptroute.ps

Décompresser une archive

Extension	Commande
.tar.gz	tar zxvf <nom_du_programme>.tar.gz
.tar.bz2	tar jxvf <nom_du_programme>.tar.bz2
.gz	gunzip <nom_du_programme>.gz
.bz2	bunzip2 <nom_du_programme>.bz2
.zip	unzip <nom_du_programme>.zip

Je n'ai plus de souris

Raccourci	Fonction dans KDE
Alt + F12	Démarre l'émulation du déplacement de la souris avec les touches du pavé numérique
2, 4, 6, et 8, (ou Bas, Gauche, Haut et Droite)	Déplace le curseur de la souris dans les quatre directions
Entrée (ou Espace)	Clic gauche et sortie du mode émulation

Autre possibilité, plus riche mais plus contraignante, lancer le programme `xorgcfg` avec Alt + F2.

Touches du pavé numérique	Fonction dans xorgcfg
1 2 3 4 6 7 8 9	Déplace le curseur de la souris dans les huit directions
/ (ou * ou -)	Sélection du bouton gauche (ou milieu ou droit)
5 (ou +)	Clic (ou double clic) avec le bouton sélectionné
0 ou .	Verouillage (ou relâchement) de l'appui du bouton sélectionné

Par ailleurs dans une boîte de dialogue on peut se déplacer entre les différentes zones ou boutons avec Tab ou Maj + Tab, dans les zones ou listes avec Haut, Bas, Gauche ou Droite, et cliquer avec Espace. Les listes déroulantes s'ouvrent et se ferment souvent avec F4.

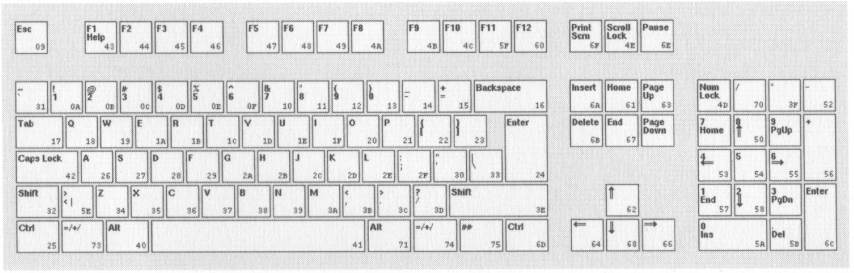
Touches magiques

Si plus rien ne marche vous pouvez redémarrer votre machine avec les raccourcis suivants (attention il faut utiliser les trois raccourcis successivement).

Raccourci	Fonction
Alt + ImprEcran + S	Vide le cache disque
Alt + ImprEcran + U	Remonte le(s) disque(s) en lecture seule
Alt + ImprEcran + B	Redémarre la machine

Toutes les notions contenues dans cet aide-mémoire sont présentées de façon détaillée dans le livre, et sont donc décrites ici sommairement.

Clavier américain



Aide-mémoire du débutant sous Linux

Raccourcis généraux

Raccourci	Fonction
Ctrl + Alt + Plus (du pavé numérique)	Passe à la résolution d'écran supérieure
Ctrl + Alt + Moins (du pavé numérique)	Passe à la résolution d'écran inférieure
Ctrl + Alt + Retour	Redémarre le serveur X
Ctrl + Alt + Suppr	Redémarre l'ordinateur
Ctrl + Alt + F1 à F6	Accès aux consoles texte
Ctrl + Alt + F7 (F8 etc.)	Accès aux consoles graphiques
Maj + Précédent (ou Suivant)	Remonte (ou redescend) dans l'affichage d'une console

KDE

Raccourci	Fonction
F1	Ouvre le centre d'aide de KDE
Ctrl + F1 à F12	Accès direct à un bureau (s'il existe)
Ctrl (+ Maj) + Tab	Déplacement circulaire entre les bureaux (en arrière)
Alt (+ Maj) + Tab	Déplacement circulaire entre les fenêtres du bureau courant (en arrière)
Alt + F2	Ouvre la mini-ligne de commande
Alt + F3	Menu des opérations de fenêtre
Alt + F4	Ferme une fenêtre
Alt + F5 ou clic milieu sur le fond du bureau	Affiche la liste des fenêtres
Ctrl + Alt + Echap	Permet de tuer une application avec un clic gauche (Échap pour revenir au curseur normal)
Ctrl (+ Maj) + A	Tout (dé)sélectionner
Ctrl + O (ou W ou S)	Ouvrir (ou fermer ou enregistrer) un fichier
F4	Ouvre ou referme une liste déroulante
Haut ou Bas	Navigue dans une liste déroulante même non ouverte
Ctrl (ou Maj) pendant un glisser-déposer	Copie (ou déplace) le(s) fichier(s) sans demander confirmation
Ctrl + Maj pendant un glisser-déposer	Crée un lien sans demander confirmation
Ctrl + Alt + L	Efface l'écran et le verrouille
Alt + Bouton gauche (ou droit) de la souris	Déplace (ou redimensionne) la fenêtre
Ctrl + Molette de la souris	Zoom avant ou arrière dans le document ou la page web
Alt + Molette de la souris	Défilement horizontal des ascenceurs
Clic droit sur le fond du bureau	Menu contextuel de création/configuration du bureau/déconnexion

Édition

Raccourci	Fonction
Ctrl + C	Copier
Ctrl + V	Coller
Ctrl + X	Couper
Ctrl + Droite (ou Gauche)	Se déplacer d'un mot vers la droite (ou vers la gauche)
Ctrl + Maj + Droite (ou Gauche)	Sélectionner un mot vers la droite (ou vers la gauche)
Ctrl + Supr (ou Efface)	Supprimer le mot de droite (ou de gauche) *
Ctrl (+ Maj) + Z	Annuler (rétablir)
Ctrl + F	Rechercher
(Maj +) F3	Rechercher le suivant en avant (en arrière)
Ctrl + R	Remplacer
Mal + Début (ou Fin)	Sélectionne du début (ou de la fin) de la ligne jusqu'au curseur
Ctrl + Début (ou Fin)	Aller au début (ou à la fin) du document

* Dans OpenOffice.org 1.1 cette combinaison de touches efface la phrase complète.

Navigation

Raccourci	Fonction
Alt + Gauche (ou Droite)	Affiche la page précédente (ou suivante) de l'historique de la navigation
F5	Recharge la page
Maj + Clic gauche	Enregistre le lien sous...
Ctrl + Plus (ou Moins) du pavé numérique	Zoom avant (ou arrière) dans une page

Le Shell – Ergonomie

Raccourcis clavier

Raccourci	Fonction
Tab	Active la saisie assistée
Ctrl + A (ou E)	Déplace vers le début (ou la fin) de la ligne de commande
Ctrl (ou Alt) + B (ou F)	Déplace d'un caractère (ou d'un mot) en arrière (ou en avant)
Ctrl (ou Alt) + D	Efface le caractère (ou le mot) suivant le curseur
Ctrl + K (ou Y)	Efface la fin de la ligne (ou rapelle le dernier effacement)
Ctrl (ou Alt) + T	Inverse les caractères (ou les mots) précédant le curseur puis déplace le curseur après le dernier caractère (ou mot) inversé
Ctrl + C	Stoppe la commande en cours
Ctrl + D	Ferme le terminal en cours
Ctrl + L	Nettoie l'écran

Utilisation de l'historique

Nom	Fonction
history | grep `texte1`	Affiche toutes les commandes commençant par `texte1` et leur numéro d'historique
! `nombre`	Exécute la commande ayant le numéro `nombre` dans l'historique
`texte`!!	Exécute une commande composée de `texte` concaténé avec la dernière commande tapée
`texte1`!`texte2`	Exécute une commande composée de `texte1` concaténé avec la dernière commande tapée commençant par `texte2`

Le Shell – Commandes

Options et notations générales

Nom	Fonction
. (un point) .. (deux points) ou ~	Répertoire courant, répertoire parent ou répertoire utilisateur
Caractères jokers ? et *	? remplace un caractère et * un nombre indéfini de caractères dans le nom d'un fichier (* seul signifie tous les fichiers)
commande - -help	Affiche une aide sommaire sur la commande
commande - -version (ou -v)	Affiche le numéro de version de la commande
commande &	Exécute la commande en arrière-plan et rend la main

Commandes

Les différentes options (précédées d'un tiret) peuvent être chainées. Ex : `ls -al`, `rm -rf`.
Un fichier peut être désigné par son nom seul s'il est dans le répertoire courant, sinon par son nom complet comprenant aussi le chemin d'accès relatif (à partir du répertoire courant) ou absolu (à partir de la racine /).
Exemple de chemin absolu : `/home/perrine/repertoire/fichier` (autre syntaxe : `~/repertoire/fichier`) et de chemin relatif à partir de `/home/perrine` : `/repertoire/fichier`. Le nom d'un répertoire commence toujours par /.
On peut chaîner deux commandes avec le caractère | (tube) de la façon suivante : `commande1 | commande2`. La sortie de la `commande1` est utilisée comme entrée de la `commande2`. Le caractère | s'obtient avec Alt Gr + 6 (du pavé alphabétique).
On peut envoyer la sortie d'une commande dans un fichier texte ainsi : `commande>fichier.txt` pour créer ou écraser le fichier, et `commande>>fichier.txt` pour compléter le fichier existant.

Fichiers et répertoires

Nom	Fonction
cd <repertoire> cd ..	Déplace vers le <repertoire> Déplace vers le répertoire parent
ls (<repertoire>) (| more) ls -a, ou ls -l	Liste les fichiers du répertoire courant (ou de <repertoire>) (page par page) Liste aussi les fichiers cachés, ou de façon détaillée
cp <fichier1> <fichier2> cp <fichier> <repertoire> cp -R <repertoire1> <repertoire2>	Copie le <fichier1> sur le <fichier2> Copie le <fichier> dans le <repertoire> Copie <repertoire1> et tout ce qu'il contient dans <repertoire2>
mv <fichier1> <fichier2> mv <fichier> <repertoire>	Renomme <fichier1> en <fichier2> Déplace <fichier> du répertoire courant vers <repertoire>
rm (-f) <fichier> rm (-r) <repertoire>	Supprime <fichier> (sans demander confirmation) Supprime <repertoire> (et tout ce qu'il contient)
mkdir <repertoire> rmdir <repertoire>	Crée le <repertoire> Supprime le <repertoire> (uniquement s'il est vide)
ln (-s) <cible> <lien>	Crée un lien (symbolique) vers <cible> nommé<lien>
chmod <droits> <fichier> (Lecture = 4, Écriture = 2, Exécution = 1)	Affecte de nouveaux droits au <fichier>
chown, chgrp <fichier>	Affecte un nouveau propriétaire, ou groupe au <fichier>
mount	Affiche les systèmes de fichiers montés
mount (ou umount) <peripherique> (ou <point_de_montage>)	Monte (ou démonte) le périphérique (de la forme `/dev/xxxx`) ou le point de montage (de la forme `/repertoire`)

Processus

Nom	Fonction
ps -A (PS aux) (| grep <nom>)	Affiche la liste des processus et leur PID (affichage plus complet) (uniquement le programme ou la commande <nom>)
kill (-9) PID	Tue proprement (ou sauvagement) le processus ayant le numéro PID
killall (-9) <nom>	Tue proprement (ou sauvagement) le(s) processus nommés <nom>

Recherche et information

Nom	Fonction
find <repertoire> -critere find <repertoire> -name "*chaine*"	Trouve un fichier correspondant au critère dans <repertoire> Trouve tous les fichiers de <repertoire> dont le nom contient chaine
grep <chaine> <*nom*>	Affiche les fichiers contenant <chaine> dont le nom contient <nom>
locate <fichier>	Affiche le(s) répertoire(s) contenant <fichier>
file <fichier>	Affiche le type de <fichier>
cat <fichier> (| more)	Affiche le contenu de <fichier> (page par page)
which <commande>	Affiche le chemin du répertoire où se trouve <commande>
df (ou du) <repertoire>	Affiche le taux d'occupation des partitions montées et points de montage (ou de tous les sous-répertoires) de <repertoire>
pwd	Affiche le nom du répertoire courant
whoami	(Qui suis-je?) Affiche le nom de l'utilisateur courant
id <utilisateur>	Affiche les numéros de l'utilisateur et de ses groupes
free	Affiche l'occupation de la mémoire physique et du swap
top	Affiche la liste des tâches qui mobilisent le plus le processeur
tail (-25 ou -f) <fichier>	Affiche les 10 (ou 25, ou réactualise) dernières lignes de <fichier>
(/sbin/)ifconfig	Affiche les informations réseau (avec le préfixe si simple utilisateur)
uname	Affiche la version du noyau
halt, reboot	Stoppe ou redémarre la machine