

## Architecture des Réseaux (ARes) 2/5 : Application

Olivier Fourmaux  
(olivier.fourmaux@upmc.fr)

Version 7.0

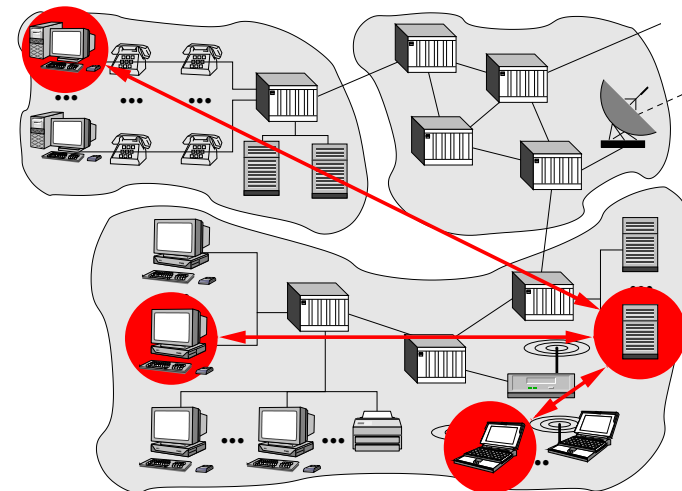
## ARes : plan du cours 2/5

- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## ARes : plan du cours 2/5

- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## Applications




## Couche Application

### Definition

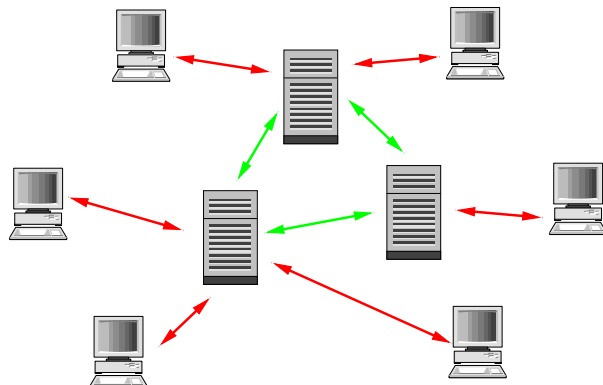
**La couche application** Ensemble des programmes et protocoles de haut niveau qui permettent aux utilisateurs de communiquer

### Remarques :

- standardise les échanges entre les applications les plus courantes
  - accès au web (HTTP), envoi d'e-mail (SMTP, POP, IMAP) ...
-  **applications  $\neq$  protocoles de la couche application**
- définit l'interface réseau avec les utilisateurs
  - s'appuie sur les services de bout-en-bout définis dans les couches inférieures
- supporte les environnements hétérogènes

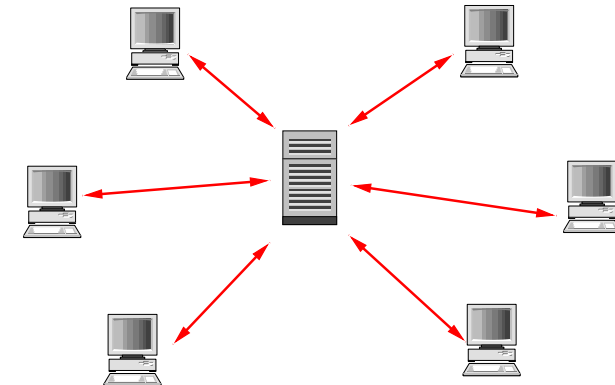
## Architectures (2)

### Client/serveur avec réplication des serveurs



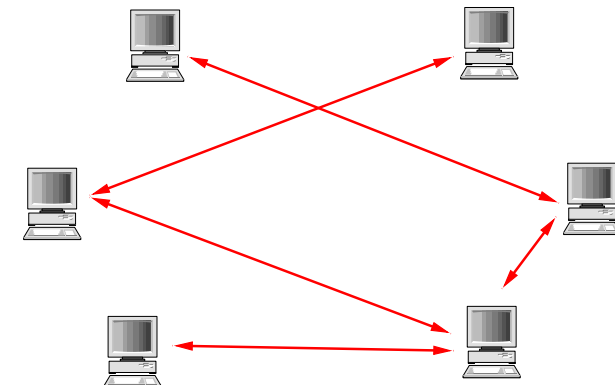
## Architectures (1)

### Client/serveur centralisé classique



## Architectures (3)

### Peer-to-peer classique



## Comparaison Client/serveur et P2P

### RPC/RMI

- synchrones
- asymétriques
- orientés langage
- identification
- authentification

Client\_call(args)

Server\_main\_loop()

```
while (true)
  await(call)
  switch(call.procid)
    case 0: call.ret=proc0(call.arg)
    case 1: call.ret=proc1(call.arg)
    ...
```

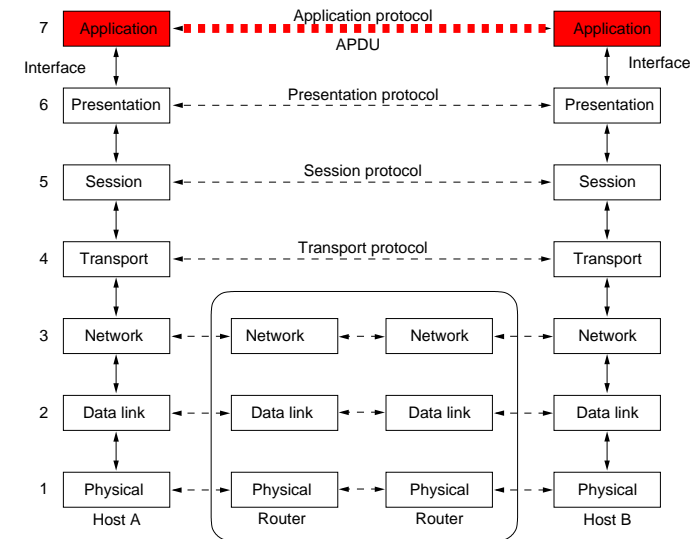
### Messages P2P

- asynchrones
- symétriques
- orientés service
- anonymat
- haute disponibilité

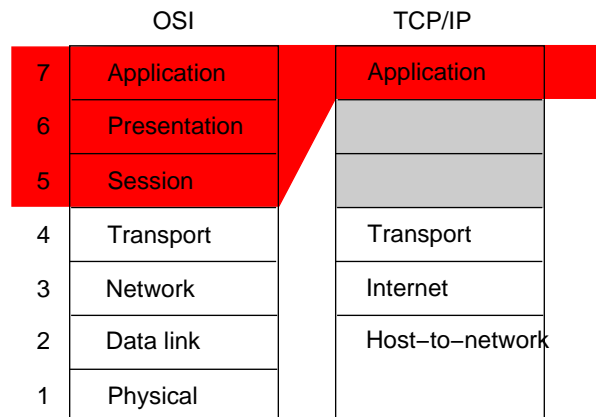
Peer\_main\_loop()

```
while (true)
  await(event)
  switch(event.type)
    case timer_expire:
      do_some_P2P_work()
      randomize_timers()
    case inbound_mesg:
      handle_mesg()
```

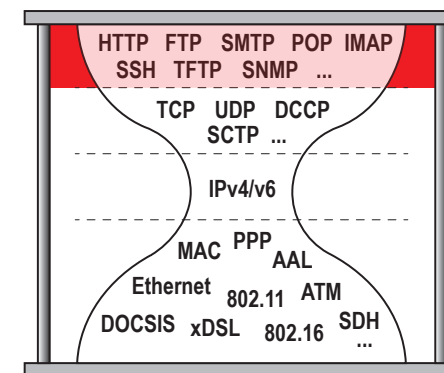
## Couche Application : modèle OSI



## Couche Application : modèle TCP/IP (1)



## Couche Application : modèle TCP/IP (2)



Dans l'Internet, des centaines de protocoles applicatifs existent !

HTTP pour surfer sur la toile

FTP pour transférer des données

SMTP pour échanger du courrier électronique...

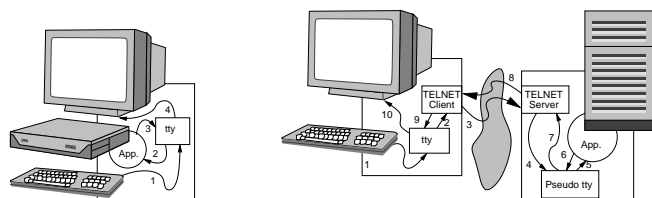
## ARes : plan du cours 2/5

- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## TELNET (TELecommunication NETwork protocol)

Application développée dès 1969 (RFC 15) et standardisé à l'IETF en 1983 (RFC 854 et Internet Standard STD 8)

- repose sur une connexion **TCP** (port serveur = 23)
- mécanisme de négociation d'options
- service de terminal virtuel
- pas de confidentialité (mot de passe en clair...)



## Applications de connexion à distance

A partir d'un terminal ouvert sur une machine locale, connexion sur une machine distante

- plusieurs protocoles :
  - TELNET
  - RLOGIN
  - SSH...
- application de type client/serveur
  - **client** : interagit avec l'utilisateur et les protocoles réseaux
  - **serveur** : idem au niveau de l'application distante
- besoin d'interactivité
  - tout ce qui tapé sur le clavier local est envoyé **rapidement** sur la connexion
  - tout ce qui est reçu de la connexion est affiché **rapidement** sur l'écran local

## TELNET : options

Plusieurs échanges initiaux pour les options (RFC 855) :

- le client émet des **requêtes** (WILL WON'T DO DON'T)
  - Command: Do Suppress Go Ahead
  - Command: Will Terminal Type
  - Command: Will Negotiate About Window Size
  - Command: Will Terminal Speed...
- le serveur renvoie des **réponses** (DO DON'T WILL WON'T)
  - Command: Do Terminal Type
  - Command: Will Suppress Go Ahead
  - Command: Dont Negotiate About Window Size
  - Command: Do Terminal Speed...
- chaque extrémité implémente une version minimale du NVT
  - négociation d'options pour les machines plus évoluées

## TELNET : NVT

### Définition d'un terminal virtuel (*Network Virtual Terminal*)

- pas de format de message, mais un en codage des données
- codage vers un système de représentation commun : **NVT**
  - chaque système peut transcoder
- Exemple :
  - `local : cc maa<bs>x.c`
  - `NVT : [c] [ ] [x] [IAC EC] [a] [a] [m] [ ] [c] [c]` →  
IAC = *Interpret As Command* (octet valeur 255)
  - il n'est pas nécessaire de connaître la conversion vers chaque type de machine
- communication dans les **environnement hétérogènes**
- contrôle **in-band**

**terminal local réel** ⇔ **terminal réseau virtuel**

## TELNET : Accès à d'autres serveurs

### Exemple d'accès à un serveur web avec TELNET :

```
Unix> telnet hobbes.lip6.fr 80
Trying 137.86.111.77...
Connected to hobbes.lip6.fr.
Escape character is '^]'.
GET /index.html HTTP/1.0
HTTP/1.1 200 OK
Date: Tue, 24 Sep 2002 15:33:07 GMT
Server: Apache/1.3.9 (Unix) Debian/GNU
Connection: close
Content-Type: text/html; charset=iso-8859-1

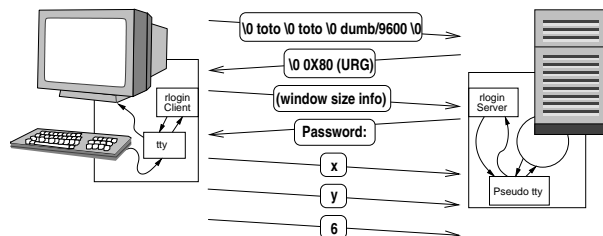
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
...
Connection closed by foreign host.
```

### → connexion TCP brute (limitation NVT)

## RLOGIN (*Remote LOGIN*)

### Application standard d'Unix BSD (RFC 1282)

- beaucoup plus simple que TELNET, pas de négociation
- repose sur une connexion **TCP** (port serveur = **513**)
- quelques commandes **in-band** en données urgentes
- pas de confidentialité (mot-de-passe **en clair**) et confiance (.rhost)



## SSH (*Secure SHell*)

- communications cryptées**, assurant :
  - authentification
  - confidentialité
  - intégrité
- repose sur une connexion **TCP** (port serveur = **22**)
  - rajoute une couche transport intermédiaire
  - authentification cryptée
  - négociation des algorithmes
  - (mux. de sessions, tunnels : X11, relayage de port, SOCKS...)
- standardisation tardive (janvier 2006) : RFCs 4251 à 4254
- nombreuses implémentations
  - OpenSSH (natif sur BSDs, GNU/Linux, MacOSX, Cygwin...)
  - PuTTY (Windows et Unixes)...

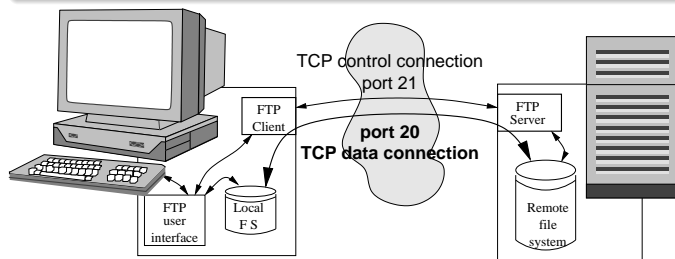
## ARes : plan du cours 2/5

- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## FTP (File Transfer Protocol)

### Standard TCP/IP pour le transfert de fichiers (RFC 959)

- signalisation **out-of-band**, deux connexions TCP
- accès **interactif**
- contrôle d'accès (mais mot de passe en clair)



## Applications de transfert de fichiers usuelles

Copie d'un fichier d'un système vers un autre en environnement hétérogène

- plusieurs protocoles :
  - FTP
  - TFTP
  - RCP, SCP, SFTP...
- application de type client/serveur
  - **client** : interagit avec l'utilisateur, le système de fichier local et les protocoles réseaux
  - **serveur** : interagit avec les protocoles réseaux et le système de fichier distant
- ne pas confondre avec les systèmes de fichiers distants
  - NFS (Sun, TCP/IP), SMB (Microsoft)...

## FTP : Connexions

Deux connexions TCP sont utilisées en parallèle :

- connexion de **contrôle**
  - **permanente** (créée à l'ouverture de la session FTP)
  - full duplex initiée par le client (port serveur = 21)
  - utilisée uniquement pour échanger les **requêtes** et **réponses**
  - besoin d'**interactivité** (et de fiabilité)
- connexion de transfert de **données**
  - **temporaire** (créée à chaque transfert de fichier)
  - full duplex (initiée par le **serveur**)
    - transmission préalable du **port client** à utiliser
  - envoi de fichiers et de liste de fichiers/répertoires
  - besoin de **débit** (et de fiabilité)
  - libérée à la fin de chaque transfert de fichier

## FTP : Données

Nombreuses représentations des données (hôtes hétérogènes) :

- type de fichiers :
  - **non structurés**
  - enregistrements
  - pages
- encodage des données :
  - **ASCII** (*American Standard Code for Information Interchange*)
  - EBCDIC (*Extended Binary-Coded Decimal Interchange Code*)
  - **binaire**
- type de transmission :
  - **flux**
  - bloc
  - compressé

➡ vérifier le type de données transférées



## Commandes utilisateur du programme ftp

```
Unix> ftp
ftp> help
```

Commands may be abbreviated. Commands are:

!	debug	mdir	sendport	site
\$	dir	mget	put	size
account	disconnect	mkdir	pwd	status
append	exit	mls	quit	struct
ascii	form	mode	quote	system
bell	get	modtime	recv	sunique
binary	glob	mput	reget	tenex
bye	hash	newer	rstatus	tick
case	help	nmap	rhel	trace
cd	idle	nlist	rename	type
cdup	image	ntrans	reset	user
chmod	lcd	open	restart	umask
close	ls	prompt	rmdir	verbose
cr	macdef	passive	runique	?
delete	mdelete	proxy	send	



## FTP : Requêtes

Codage ASCII NVT ➡ Mode interactif possible (si lisible)

```
Unix> telnet galion.ufr-info-p6.jussieu.fr 21
Trying 197.18.176.12...
Connected to localhost.
Escape character is '^]'.
220 ProFTPD 1.2.0pre10 Server (Debian) [galion.ufr-info-p6.jussieu.fr]
help
214-The following commands are recognized (* =>'s unimplemented).
214-USER      PASS      ACCT*    CWD      XCWD     CDUP     XCUP     SMNT*
214-QUIT      REIN*    PORT     PASV     TYPE     STRU*    MODE*    RETR
214-STOR      STOU*    APPE     ALLO*    REST     RNFR     RNT0     ABOR
214-DELE      MDTM     RMD      XRMD     MKD      XMKD     PWD      XPWD
214-SIZE      LIST     NLST     SITE     SYST     STAT     HELP     NOOP
214 Direct comments to root@galion.ufr-info-p6.jussieu.fr.
```



Ne pas confondre avec les commandes de l'interface utilisateur de ftp



## FTP : Réponses

Codage usuel : *status + description textuelle*

status	description	status	description
		x0z	syntaxe
1yz	réponse positive préliminaire	x1z	information
2yz	réponse positive complète	x2z	connexions
3yz	réponse positive intermédiaire	x3z	authentification
4yz	réponse négative transitoire		
5yz	réponse négative définitive	x5z	système de fichier

- 150 Opening BINARY mode data connection
- 200 Command successful
- 220 ProFTPD 1.2.0pre10 Server (Debian)
- 226 Transfer complete
- 230 User toto logged in
- 331 Username OK, Password required
- 425 Can't open data connection
- 500 Syntax error (Unknown command)...



## FTP : Exemple

## Programme ftp (interface utilisateur)

```
[toto@hobbes]$ ftp calvin.lip6.fr
Connected to calvin.lip6.fr.
220 FTPD 1.2pre8 Server (Debian)
Name (calvin.lip6.fr):toto
331 Password required for toto.
Password:
230 User toto logged in.
ftp> get toinst.txt
local: toinst.txt remote: toinst.txt

200 PORT command successful.

150 Opening BINARY mode data connection
for toinst.txt (1 bytes).
226 Transfer complete.
1 bytes received in 0.377s (0.0026 KB/s)
ftp> quit
221 Goodbye.
[toto@hobbes]$
```

## Protocole FTP (connexion de contrôle)

```
220 FTPD 1.2pre8 Server (Debian)
USER toto
331 Password required for toto.
PASS ABjGa!9F
230 User toto logged in.

PORT 192,33,82,12,4,15
200 PORT command successful.
RETR toinst.txt
150 Opening BINARY mode data connection
for toinst.txt (1 bytes).
226 Transfer complete.

QUIT
221 Goodbye.
```

## TFTP (Trivial File Transfer Protocol)

- protocole léger pour le transfert de fichiers (version 2 : RFC 1350)
- datagrammes UDP sur le port 69

- 5 messages :

opcode	nom	description
1	RRQ	requête de lecture
2	WRQ	requête d'écriture
3	DATA	données numérotées
4	ACK	acquiescement
5	ERREUR	message d'erreur

- messages DATA avec 512 octets (sauf le dernier de taille inférieure ou éventuellement nulle)
- protocole *stop-and-wait*
  - numérotation des messages DATA
  - acquiescement immédiat avec ACK
- pas de contrôle d'accès (sous Unix, souvent limité à /tftpboot)

## FTP : Divers

## Anonymous : compte invité sur certains serveurs FTP :

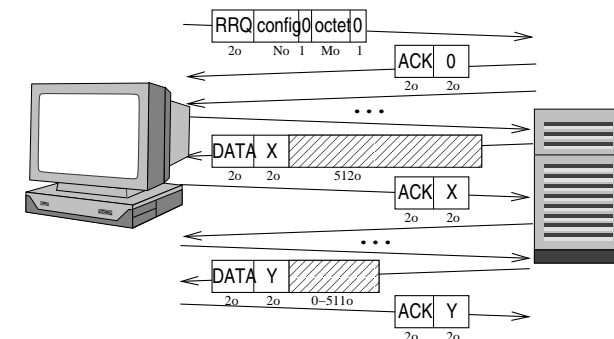
- username: anonymous
- password: *adresse@electronique.org*

## Mode passif : ouverture de la connexion donnée en sens inverse

- si l'ouverture usuelle de la **connexion donnée** impossible
  - filtrage des adresses (*firewall*)
  - translation d'adresses (NAT)
- commande PASV (RFC 1579)
  - le client envoie la commande PASV au serveur a.b.c.d
  - le serveur alloue le port 256x+y, fait une ouverture passive sur ce port et en informe le client avec une réponse
    - 227 Entering passive mode (a,b,c,d,x,y)
  - le client fait une ouverture active vers le port 256x+y

## TFTP : Exemple

```
[toto@hobbes]$ tftp calvin.lip6.fr
tftp> get config
Received 5220 bytes in 0.377 secs
tftp> quit
[toto@hobbes]$
```





## RCP, SCP, SFTP

### Protocole R\* : RCP

- spécifique à Unix et associé aux `r*` commandes (dont `rcp`)
  - le client `rcp` fonctionne avec un serveur `rshd`
  - idem `rlogin` : obsolète, problèmes de sécurités...

### Protocoles sécurisés : SCP, SFTP

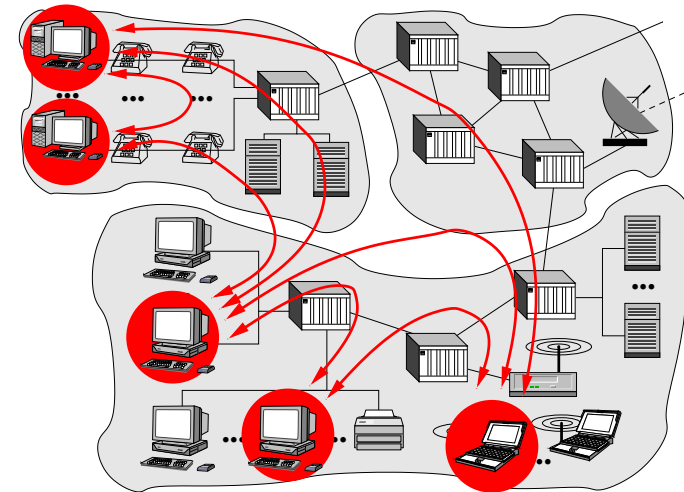
- `scp` : copie simple similaire à `rcp` encapsulé dans SSH
- `sftp` : idem FTP mais facilement encapsulable
  - SFTP est un nouveau protocole (groupe IPSEC de l'IETF)
  - SFTP peut être utilisé avec SSH (par défaut avec de nombreux clients `sftp`)
  - SFTP est différent de FTPS qui introduit la sécurisation au niveau des connexions avec SSL/TLS (*Secure Socket Layer/Transport Layer Security*)

## De nombreuses applications *Peer-to-peer*...

... mais peu de standards

- partage de fichiers**
  - Napster, eDonkey, BitTorrent...
  - FastTrack (KaZaA, Grokster et Imesh), Gnutella...
  - Gnutella...
  - BitTorrent
- distribution de flux audio/vidéo**
  - VoD : Peercast, Joost...
  - P2PTV : Coolstreaming, TVants, PPLive...
- stockage anonyme**
  - Freenet, Entropy...
- distributed computing**

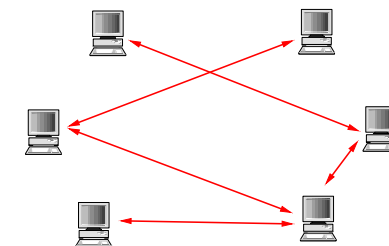
## Applications de partage de fichier *Peer-to-peer*



## Fonctionnalités P2P

### Caractéristiques des systèmes *peer-to-peer*

- pas de rôle distinct
  - évite les points de congestion ou les nœuds défaillants
- besoin d'algorithmes distribués
  - découverte de service (nommage, adressage, calcul de métriques)
  - maintien d'un état du voisinage
  - routage au niveau applicatif (lié au contenu)
  - robustesse, gestion de perte de liens ou de nœuds ...



## Napster



### Programme de partage de fichiers MP3

- pas le premier, mais le plus connu.
  - très informatif sur l'intérêt du *peer-to-peer*...  
... sur les problèmes techniques, légaux, politiques, économiques
- une technologie de rupture ?
  - historique
    - fin 98 : Shawn Fanning (19 ans) débute le développement
    - 05/99 : création de *Napster Online Music Service*
    - 06/99 : premiers tests du logiciel
    - 12/99 : premières poursuites juridiques (Metallica, RIAA...)
    - mi 00 : plus de **60M** d'utilisateurs  
importante part du trafic universitaire (30% à 50%)
    - 02/01 : jugement par la Cour d'Appel des US
    - mi 01 : 160K utilisateurs...

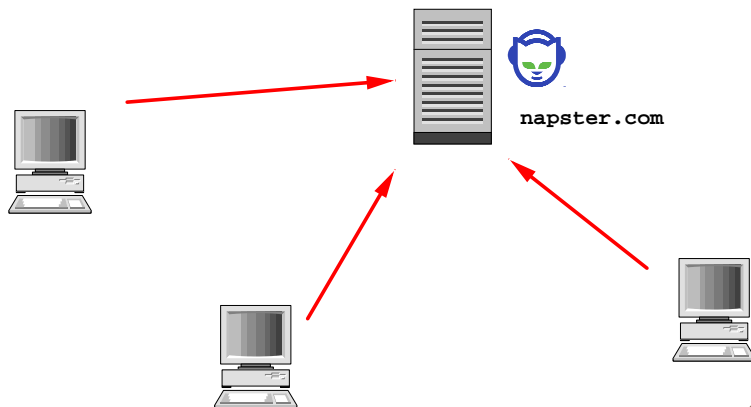
## Napster : Principe

### Approche mixte :

- recherche client/serveur avec liste centralisée
- échange direct des données recherchées entre pairs
- connexions point à point TCP (port 7777 ou 8888)
- 4 étapes :
  - Connexion au serveur Napster
  - Envoi de sa liste de fichiers au serveur (*push*)
  - Envoi des mots recherchés et récupération d'une liste de pairs
  - Sélection du meilleur pair (*pings*)

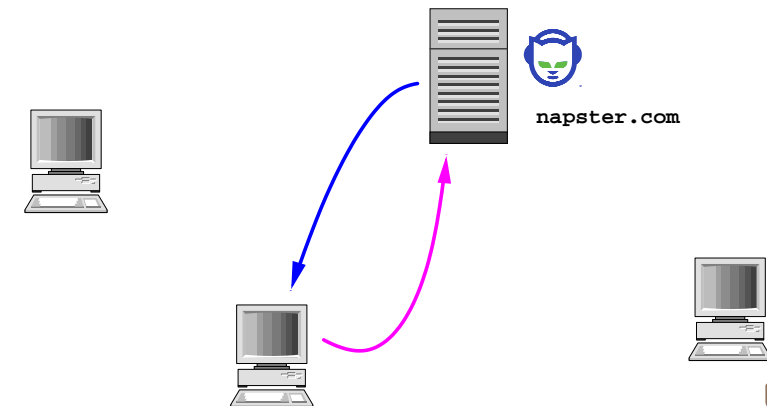
## Napster : Upload

Les utilisateurs chargent la liste des fichiers à partager



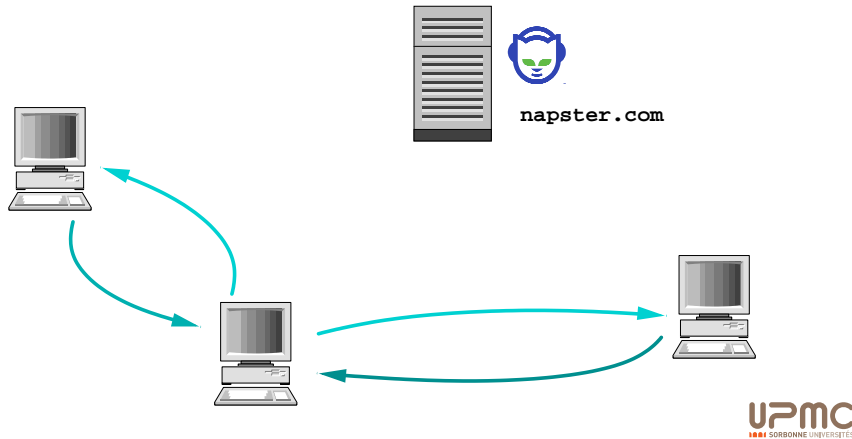
## Napster : Search

Un utilisateur émet une requête de recherche  
Le serveur indique les localisations potentielles



## Napster : Pings

Ping des pairs potentiels (recherche du meilleur débit de transfert)



Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## Napster : remarques

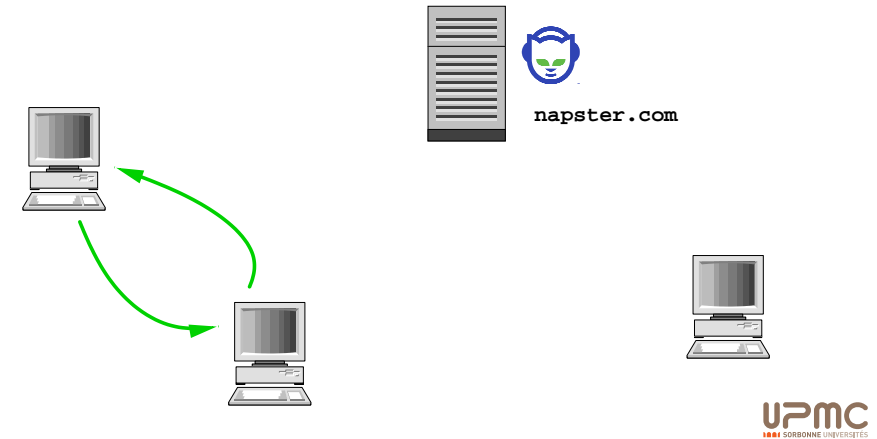
- serveur **centralisé**
  - un point de défaillance
  - risque de congestion
  - partage de charge en utilisant la rotation DNS
  - contrôlé par une entreprise
- absence de **sécurité**
  - mot de passe en clair
  - pas d'authentification
  - pas d'anonymat
  - code propriétaire
  - téléchargement de mises-à-jour

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## Napster : Download

L'utilisateur récupère directement le fichier chez le pair choisi



Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## Gnutella : motivations



Partage de fichiers complètement distribué

- corrige les défauts majeurs de Napster :
  - *Open source*
  - pas de serveurs - pas d'index global
  - connaissance locale seulement
- mais toujours les mêmes problèmes juridiques, économiques...
  - pas de responsable direct du service
  - absence d'anonymat
    - le RIAA attaque directement des utilisateurs !

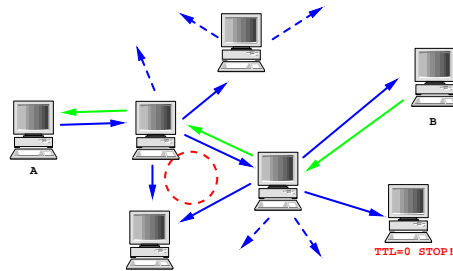
Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## Gnutella : principe

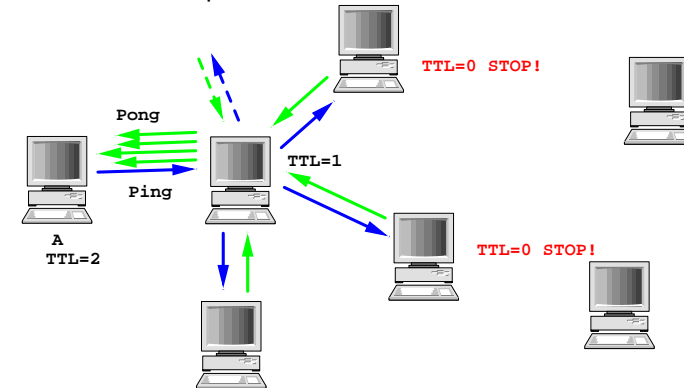
### Recherche par inondation (*flooding*)

- si je n'ai pas le fichier recherché :
  - je demande à 7 pairs s'ils ont ce fichier
- s'ils ne l'ont pas, ils contactent à leur tour 7 pairs voisins
  - recherche récursive limitée à N sauts
- détection de boucle par mémorisation temporaire des requêtes
  - les messages peuvent être reçus deux fois



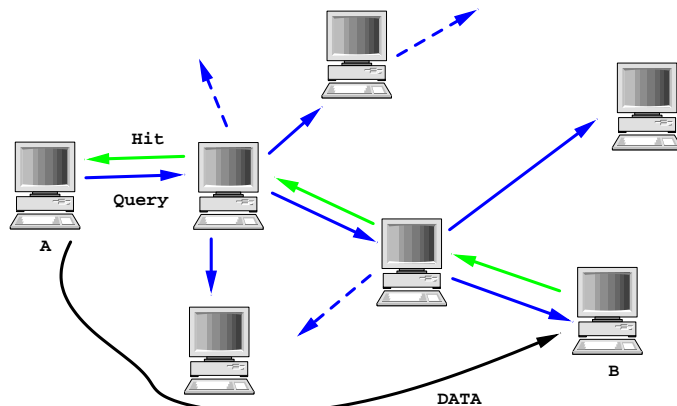
## Gnutella : identification des pairs

### Détection active des pairs



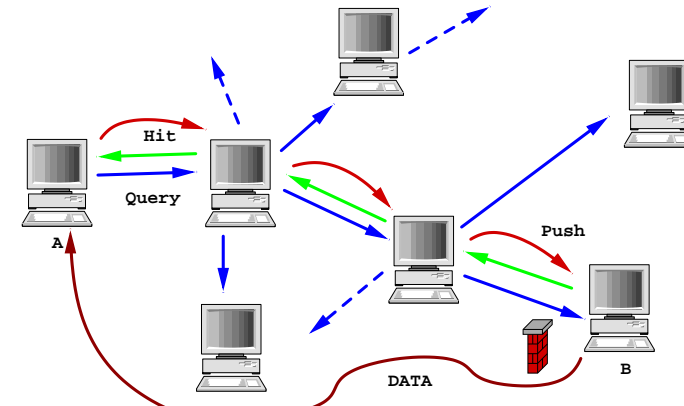
## Gnutella : recherche de fichier

### Requête pour trouver une information



## Gnutella : Firewall (2)

### Retourner la connexion des données



## Gnutella : Remarques

### Leçons retenues :

- saturation des petits pairs (modems)
  - possibilité d'indiquer que l'on a un fichier mais que l'on est saturé
- taille du réseau atteignable limitée (rupture de connectivité liée aux modems)
  - création d'une hiérarchie de pairs
- **anonymat ?**
  - le pair où l'on récupère le fichier nous connaît
    - ∃ protocoles permettant de ne pas connaître le destinataire

## BitTorrent (1)

### Partage d'un fichier :

- découpage en bloc de 64Ko à 1Mo (*Chunk*)
- création d'un *.torrent*
  - méta-données
  - signature pour chacun des *chunks*
- mise en place d'un *tracker*
  - machine qui supervise la distribution
- échange de données entre tous les demandeurs (*leechers*)
  - la source (*seed*) n'est sollicitée que pour amorcer

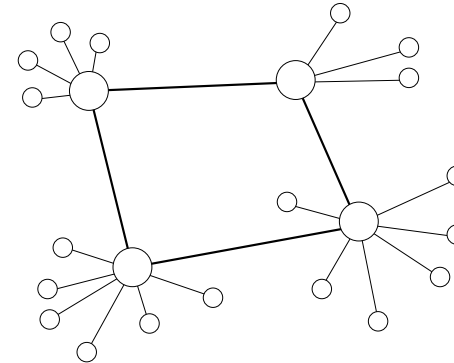
### Spécificité :

- pas de système de recherche
- pas de téléchargement direct (type HTTP)
- avantages :
  - économique
  - redondant
  - résistance aux *flash-crowd*

## Evolutions P2P

### Gnutella2, KaZaA (réseau FastTrack)...

- hôtes hétérogènes
- topologie hiérarchique
  - *Super-Nodes*



## BitTorrent (2)

### Stratégies :

- sélection de pair
  - *tit-for-tat* + *choking*
    - encourage la coopération et diminue les *free-riders*
    - sélectionne les meilleurs contributeurs, étouffe les autres
    - mécanisme périodique (10 s)
  - *optimistic unchoke*
    - découverte de nouveaux pairs
    - alimente un nouveau pair aléatoirement
    - mécanisme périodique (30 s)
- sélection de *chunk* :
  - *rarest first*
    - donne le *chunk* le plus rare en premier
    - maximise l'entropie de chaque *chunk*
  - *random first*
    - pour accélérer le démarrage des nouveaux

## BitTorrent (3)

### Evolutions :

- Indexage/recherche
  - initialement moteurs de recherche spécialisés (web)
  - *tracker* distribué (table de hachage distribuée)
    - basée sur Kademlia
- *multitracker*
  - redondance
  - surcout en signalisation
- cryptage des échanges
  - *Protocol header encrypt (PHE)*
  - *Message stream encryption/Protocol encryption (MSE/PE)*
- distribution de contenus (*streaming A/V*)
  - nombreux projets...

## ARes : plan du cours 2/5

- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## P2P : autres

### Systèmes *peer-to-peer* structurés de recherche par le contenu :

- **Chord**
  - identification par clé (SHA-1 sur une chaîne)
  - localisation par clé (SHA-1 sur l'adresse du nœud)
    - positionnement sur le nœud successeur le plus proche
- **Tapestry**
  - routage des identificateurs (hash) selon le suffixe des nœuds
- **CAN (Content Addressable Network)**
  - système de coordonnées cartésiennes virtuelles ...

## World Wide Web

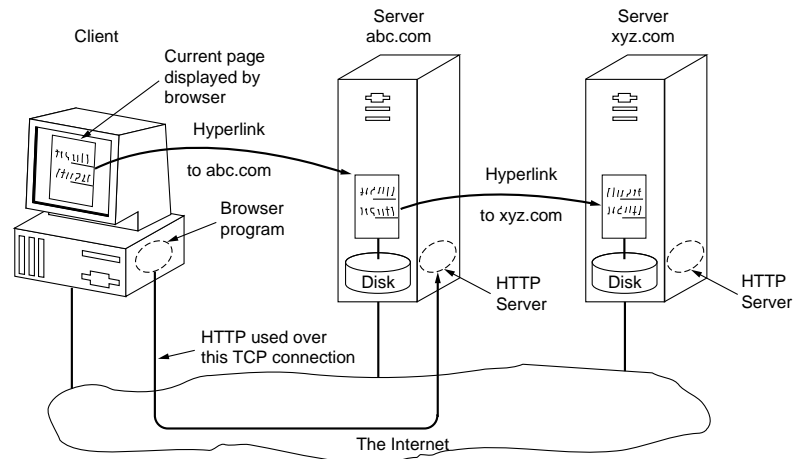
→ 90' : Internet = réseau académique

90' → : **World Wide Web**

- système d'accès aux données convivial et intuitif (graphique)
- développé au CERN par Tim Berners-Lee à partir de 1990
- première "*killer app*." grand public
- client (*browser*) :
  - NCSA Mosaic en 1993 (U. of Illinois Urbana-Champaign)
    - le WWW ne compte que 200 sites
    - première intégration d'images
    - gain de popularité exponentiel !
  - Netscape Navigator en 1994 (→ **Mozilla** en 1998)
  - Microsoft Internet Explorer en 1995 (début de la *browser wars*)
  - et beaucoup d'autres (voir le site du W3C)
- serveur (*web server*) :
  - NCSA httpd Web Server (→ **Apache** en 1998)
  - Microsoft IIS (*Internet Information Service*) en 1995

→ un protocole : **HTTP**

## HTTP : Principe



pictures from Tanenbaum A. S. *Computer Networks 3rd edition*

## HTTP : Terminologie

- une **page web** ou un **document** est composé d'objets
  - fichiers texte au format HTML
  - images GIF, JPEG...
  - applets JAVA
  - ...
- un document consiste généralement en un **fichier HTML de base** avec des références vers d'autres objets désignés par des URL
  - HTML** (*HyperText Markup Language*) est un langage à balises pour la description de documents contenant des hyper-liens identifiés par des URL
  - une **URL** (*Uniform Resource Locator*) indique un protocole pour récupérer sur une machine un objets à travers le réseau
    - `http://www.lip6.fr/info/linux.html`
    - `ftp://ftp.lip6.fr/pub/linux/disrib/debian/ls-lR.txt`
    - `file:/public/image/penguin.jpeg`
    - `mailto:olivier.fourmaux@lip6.fr`

## HTTP : Protocole

### HyperText Transfer Protocol

- connexion TCP sur le port 80
- échanges définis :
  - les **requêtes** de demande d'objets (client  $\Rightarrow$  serveur)
  - les **transferts** d'objets demandés (serveur  $\Rightarrow$  client)
- versions HTTP :
  - $\rightarrow$  97 **HTTP/1.0** (RFC1945)
    - connexions **non persistantes**, une connexion créée par objet, charge et latence importantes (TCP *three-way handshake* et *slowstart*)
  - 98  $\rightarrow$  **HTTP/1.1** (RFC2616)
    - compatibilité ascendante, connexions **persistantes**, possibilité de requêtes parallèles (**pipelining**)
- pas d'état dans le serveur (**stateless protocol**)

## HTTP : Exemple

### Browser :

```
GET /index.html HTTP/1.1
Connection: Keep-Alive
User-Agent: Mozilla/4 [en] (X11; I; Linux 0.99 i486)
Host: calvin.lip6.fr
Accept: image/gif, image/jpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: fr-FR, fr, en
Accept-Charset: iso-8859-1,*,utf-8
```

### Web server :

```
HTTP/1.1 200 OK
Date: Tue, 24 Sep 2002 12:59:28 GMT
Server: Apache/1.3.9 (Unix) Debian/GNU
Last-Modified: Sat, 29 Apr 2000 07:07:45 GMT
ETag: "1382c-ffe-390a8a41"
Accept-Ranges: bytes
Content-Length: 4094
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
<HEAD>
  <META HTTP-EQUIV="Content-Type" CONTENT="tex...
  <META NAME="GENERATOR" CONTENT="Mozilla/4.05...
  <META NAME="Author" CONTENT="johnie@debian.o...
  <META NAME="Description" CONTENT="The initia...
  <TITLE>Welcome to Your New Home Page!</TITLE>
</HEAD>
<BODY TEXT="#000000" BGCOLOR="#FFFFFF" LINK="#0...
<BR>
<H1>Welcome to Your New Home in Cyberspace!</H1>
<BR>
<IMG SRC="icons/openlogo-25.jpg" ALT="Debian">
<IMG SRC="icons/apache_pb.gif" ALT="Apache"></P>

<P>This is a placeholder page installed by the
<A HREF="http://www.debian.org/">Debian</A>
release of the
<A HREF="http://www.apache.org/">Apache</A> Web
server package, because no home page was installed
on this host. You may want to replace this as soon
as possible with your own web pages, of course....

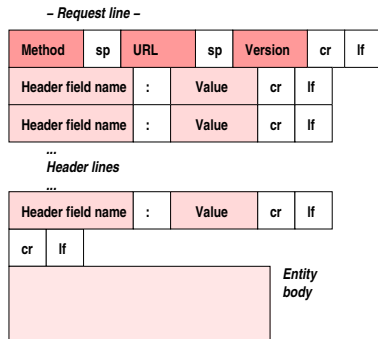
<BLOCKQUOTE>
This computer has installed the Debian GNU/Linux
operating system but has nothing to do with the
Debian GNU/Linux project. If you want to report
something about this hosts behaviour or domain,
please contact the ISPs involved directly,
<strong>not</strong> the Debian Project. <P>
</BLOCKQUOTE>
.....
</HTML>
```

## HTTP : Format requête

Exemple :

```
GET /index.html HTTP/1.1
Connection: Keep-Alive
User-Agent: Mozilla/4 [en] (X11;...)
Host: calvin.lip6.fr
Accept: image/gif, image/jpeg, */*
Accept-Encoding: gzip
Accept-Language: fr-FR, fr, en
Accept-Charset: iso-8859-1,*,utf-8
```

Format général d'un message :



- Method
  - GET
  - POST (formulaires)
  - HEAD (test de pages)
- Connection
  - Close

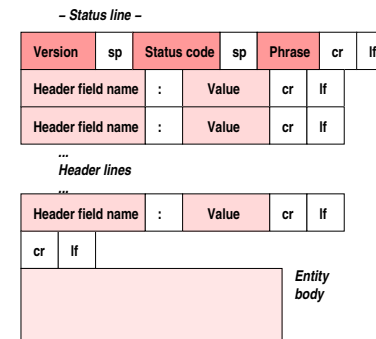
## HTTP : Format réponse

Exemple :

```
HTTP/1.1 200 OK
Date: Tue, 24 Sep 2002 12:59:28 GMT
Server: Apache/1.3.9 (Unix) Debian/GNU
Last-Modified: Sat, 29 Apr 2000 07:07:45 GMT
Content-Length: 4094
...
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
.....
</HTML>
```

Format général d'un message :



- status + description :
  - 200 OK
  - 301 Move permanently
  - 400 Bad Request
  - 404 Not Found
  - 505 HTTP Version Not Supported

## HTTP : Identifier les utilisateurs (1)

## Authentication (RFC 2617)

- 2 méthodes : simple (*Basic*) ou par signature MD5 (*Digest*)
- requête du client sur une page avec procédure d'authentification basique :
  - réponse du serveur page vide avec entête :
    - 401 Authorisation Required
    - WWW-Authenticate: *détails\_méthode\_d'autorisation*
  - requête du client sur la même page avec entête :
    - Authorization: *nom\_utilisateur mot\_de\_passe*
  - réponse du serveur :
    - si Ok ➡ la page demandée
    - sinon 401 Authorisation Required...

## HTTP : Identifier les utilisateurs (2)

## Cookies (RFC 2109)

- identifiant associé à un utilisateur sur sa machine :
- le serveur indique un cookie avec l'entête :
  - Set-cookie: *nombre\_identifiant*
- le cookie est stocké chez le client qui, lorsqu'il demandera la même page sur le même serveur, l'intégrera grâce à l'entête :
  - Cookie: *nombre\_identifiant*



## HTTP : GET conditionnel

1<sup>re</sup> requête HTTP :

GET /carte/france.jpg HTTP/1.1  
Host: www.atlas.org

1<sup>re</sup> réponse HTTP :

HTTP/1.1 200 OK  
Date: Mon, 2 Oct 2005 23:56:18  
Server: Apache/1.3.9 (Unix)  
Last-Modified: Sat, 29 Apr 2005 ...  
Content-Type: image/jpeg

Données.....  
.....  
.....  
.....

2<sup>me</sup> requête HTTP :

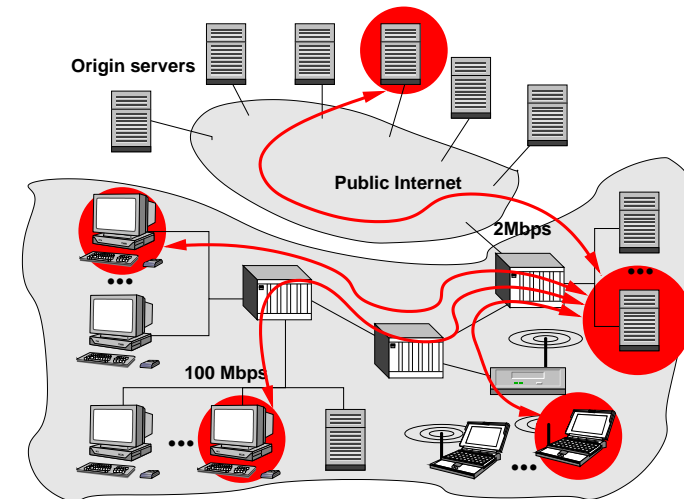
GET /carte/france.jpg HTTP/1.1  
Host: www.atlas.org  
If-modified-since: Sat, 29 Apr 2005 ...

2<sup>me</sup> réponse HTTP :

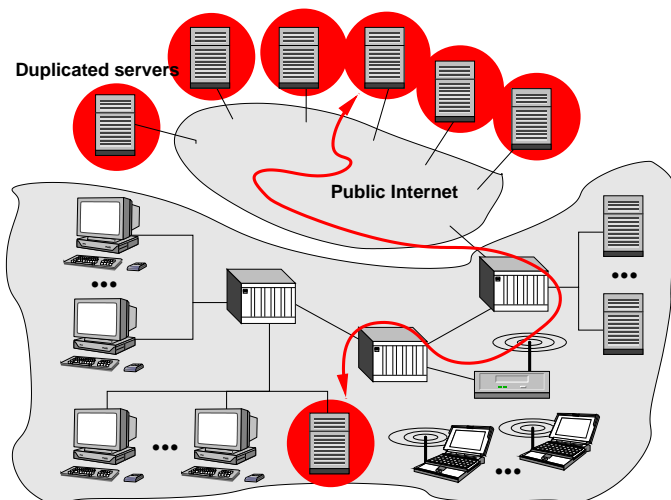
HTTP/1.1 304 Not Modified  
Date: Mon, 3 Oct 2005 00:06:43  
Server: Apache/1.3.9 (Unix) Debian/GNU



## HTTP : Cache et proxy



## HTTP : CDN



## Autour d'HTTP

Optimisation de l'accès aux ressources

- hiérarchie de caches
- répartition de charge
  - domaine des systèmes répartis

Contenu transféré

- génération automatique : PHP, ASP, Servlet...
  - programmation événementielle
- couplage aux bases d'information
  - domaine des bases de données et de la structuration de l'information type XML

Sécurité

- HTTPS (RFC 2818) : utilise SSL sur le port 443 (ou TLS)
- Applets...

Protocole de transport générique

- XML, SOAP...
- encapsulation (firewall...)



## ARes : plan du cours 2/5

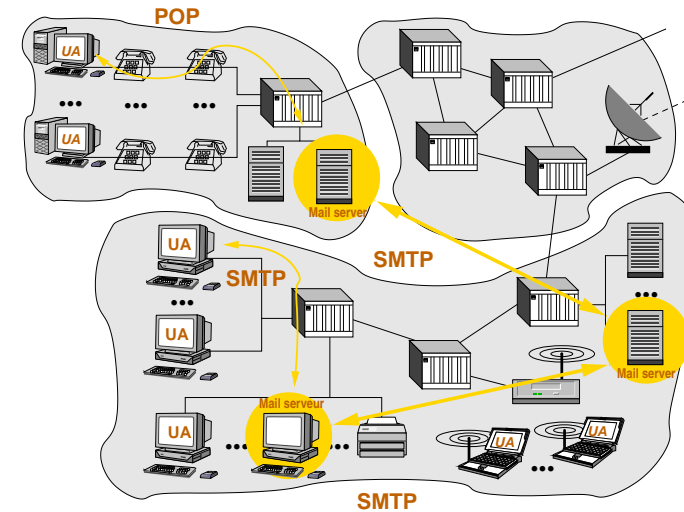
- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## SMTP : introduction

Echange de messages asynchrones à travers l'Internet

- l'ancienne "killer app."
- trois éléments de base :
  - UA (*User Agent*)
    - mail, elm, pine, mutt...
    - Eudora, Outlook et MS Mail, Mail.app, Mozilla Thunderbird...
  - serveurs de mail ou MTA (*Mail Transfer Agent*)
    - sendmail...
    - compose l'**infrastructure** du système de distribution
    - **boîtes aux lettres** des utilisateurs locaux
    - file d'attente des messages au départ ou en transit
    - temporisation et **reprise** si destinataire inaccessible
- un protocole : **SMTP**

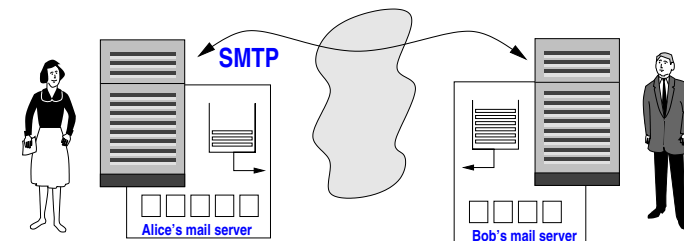
## Application de messagerie électronique



## SMTP : principes

*Simple Mail Transfer Protocol* (RFC 821 - STD 10, m.-à-j. RFC 5321)

- application client/serveur
- repose sur le service fiable des connexions **TCP**
- ancien
  - ✓ largement répandu
  - ✗ messages encodées en ASCII NVT
- connexion aux serveurs mail sur le port 25



## SMTP : exemple

```
220 hobbes.lip6.fr SMTP Sendmail 8.9.3; Wed, 22 Sep 2008 00:59:49 +0200
HELO calvin.lip6.fr
250 hobbes.lip6.fr Hello calvin.lip6.fr, pleased to meet you
MAIL FROM: pere-noel@hobbes.lip6.fr
250 pere-noel@hobbes.lip6.fr... Sender ok
RCPT TO: totu@hobbes.lip6.fr
550 totu@hobbes.lip6.fr... User unknown
RCPT TO: toto@hobbes.lip6.fr
250 toto@hobbes.lip6.fr... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Cher Toto,
N'oubliez pas de m'envoyer votre liste de cadeaux
Le Pere Noel.
.
250 BAA01090 Message accepted for delivery
QUIT
221 hobbes.lip6.fr closing connection
```



## SMTP : commandes (2)

Commandes SMTP de base (RFC 821), ensemble minimal :

HELO	Présentation du nom de domaine du client
MAIL	Identification de l'expéditeur du message
RCPT	Identification du destinataire du message
DATA	Envoi du contenu jusqu'à une ligne avec seulement un "."
QUIT	Termine l'échange de courrier
VRFY	Vérification de l'adresse du destinataire
NOOP	Pas d'opération, force le serveur à répondre
RSET	Annule la transaction



## SMTP : commandes (1)

Serveur SMTP en mode interactif :

```
Unix> telnet galion.ufr-info-p6.jussieu.fr 25
Trying 192.133.82.123...
Connected to galion.ufr-info-p6.jussieu.fr
Escape character is '^]'.
220 galion.ufr-info-p6.jussieu.fr SMTP Sendmail 8.9.3; Wed, 25 Sep 2002 00:
help
214-This is Sendmail version 8.9.3
214-Topics:
214-  HELO    MAIL    RCPT    DATA
214-  QUIT    VRFY    NOOP    RSET
214-  HELP
214-For more info use "HELP <topic>".
214-To report bugs in the implementation send email to
214-  sendmail-bugs@sendmail.org.
214-For local information send email to Postmaster at your site.
214 End of HELP info
```



## SMTP : réponses

Codage lisible usuel :

- *status + description* :
  - 220 SMTP Sendmail 8.9.3
  - 221 Closing connection
  - 250 Command successful
  - 354 Enter mail, end with "." on a line by itself
  - 550 User Unknown



## SMTP : format des messages initiaux

## Messages codés en ASCII NVT (RFC 822)

## ● l'enveloppe

- modifiée par entités SMTP successives
- commandes MAIL FROM: et RCPT TO:

## ● le message

- principalement inséré par l'agent utilisateur
- commande DATA

## ● entête

- chaque champ sur une ligne ➡ *nom: valeur*

From: Toto at Paris 13 <toto@galere.univ-paris13.fr>

Date: Mon, 22 Sep 2003 01:13:20 +0200

To: Titi at Paris 6 <titi@hypnos.lip6.fr>

Subject: rapport TER

X-Scanned-By: isis.lip6.fr

- une ligne vide

## ● corps

- terminaison par une ligne avec seulement ". "



## Evolution de l'enveloppe : ESMTP

## Quelques commandes ESMTP (RFC 1425) :

EHL0	Utilisation de ESMTP et présentation du client
SIZE	Taille maximum de message acceptée par le serveur
8BITMIME	Possibilité d'envoyer le corps encodé sur 8 bits
X???	Extension SMTP locale

## Négociation des extensions ESMTP :

EHL0 hobbes.lip6.fr.

250-hobbes.lip6.fr Hello [62.62.169.227], pleased to meet you

250-ENHANCEDSTATUSCODES

250-PIPELINING

250-EXPN

250-VERB

250-8BITMIME

250-SIZE

250-DSN

250-DELIVERBY

250 HELP



## Evolution du format des entêtes

## Caractères non ASCII dans les entêtes :

=?charset?encode?encoded-text?=

- *charset* : us-ascii, iso-8859-x, ...

- *encode* : le texte encodé doit rester en ASCII NVT

- **Quoted-printable** (Q) pour les jeux de caractères latins :

- caractères > 128 ➡ encodé sur 3 caractères (= et code hexa.)
- caractère espace ➡ toujours =20

- **Base64** (B) :

- trois octets de texte (24 bits) ➡ encodée sur 4 car. ASCII
- valeur sur 6 bits (0, 1, 2... 63) ➡ ABC...YZab...yz01...9+/  
● bourrage avec "=" si non aligné sur 4 caractères.

- *encoded-text* :

=?iso-8859-2?Q?Igen,=20k=F6sz=F6n=F6m?=

=?iso-8859-1?B?QnJhdm8sIHZvdXMgYXZleiBy6XVzc2kgIQo=?=

## MIME (Multipurpose Internet Mail Extensions)

## Nouveaux entêtes MIME (RFC 2045 et RFC 2046)

- **Mime-Version: 1.0**

- **Content-Type: type/sous-type;parametres**

- **simples** : text/plain; charset="ISO-8859-1"

- text/html, image/jpeg...

- **structurés** : multipart/mixed; Boundary=hjfdskjhfdshf

- chaque bloc du message débute par : hjfdskjhfdshf

- imbrication possible

- **Content-Disposition**: présentation du bloc (RFC 2183)

- **Content-Transfer-Encoding**: encodage du bloc

- 7 bits compatible avec les anciens MTA RFC 821

- 7bit (ASCII NVT)

- quoted-printable (recommandé pour tout texte)

- base64 (recommandé pour les flux d'octets)

- 8 bits si la commande 8BITMIME est acceptée

- 8bit et Binary (lignes ou bloc de données sur 8 bits)



## MIME : types et sous-types

/etc/mime.types	audio/midi	multipart/mixed
	audio/mpeg	multipart/parallel
	audio/x-wav	multipart/signed
application/mac-binhex40		
application/msword		text/html
application/octet-stream	image/jpeg	text/plain
application/postscript	image/png	text/richtext
application/vnd.hp-PCL	image/tiff	text/rtf
application/vnd.ms-excel		text/xml
application/x-debian-package	message/delivery-status	text/x-java
application/x-doom	message/external-body	text/x-tex
application/x-gnumeric	message/http	text/x-vcard
application/x-java-applet	message/partial	
application/x-javascript	message/rfc822	video/mpeg
application/x-msdos-program		video/quicktime
application/x-tar	multipart/alternative	video/x-msvideo
	multipart/digest	
audio/basic	multipart/encrypted	

## ESMTP : exemple de message MIME

```

From: Olivier Fourmaux <olivier.fourmaux@lip6.fr>
Date: Wed, 20 Feb 2002 01:21:01 +0100
To: Toto <toto@free.fr>
Subject: Document no 3.02
Mime-Version: 1.0
Content-Type: multipart/mixed; boundary="/9DWx/yDrRhgmJTB"
Content-Disposition: inline
Content-Transfer-Encoding: 8bit
User-Agent: Mutt/1.2.5i

--/9DWx/yDrRhgmJTB
Content-Type: text/plain; charset=iso-8859-1
Content-Disposition: inline
Content-Transfer-Encoding: 8bit

Voici le document secret que vous m'avez demandé...

--/9DWx/yDrRhgmJTB
Content-Type: application/pdf
Content-Disposition: attachment; filename="sujet-exam-RES.pdf"
Content-Transfer-Encoding: base64

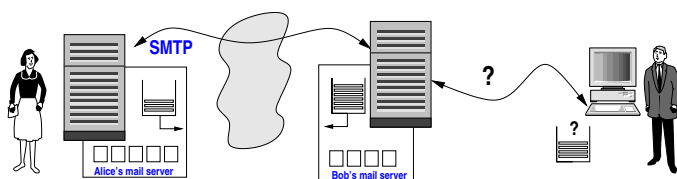
JVBERi0xLjIKJcfsj6IKNSAwIG9iago8PC9MZWN5dGggNiAwIFIvRmlsdGVCyIC9GbGF0ZUR1
Y29kZT4+CnN0cmVhbnQp4n01dy7YdtRGd3684Mx6L07T63ZkBgghgXvY1JF1MHNsYm+sHhkCS...
```

## Remise finale des messages

Machine accédant sporadiquement au réseau ?

➡ Messages stockés sur le dernier MTA (celui de l'ISP par exemple)

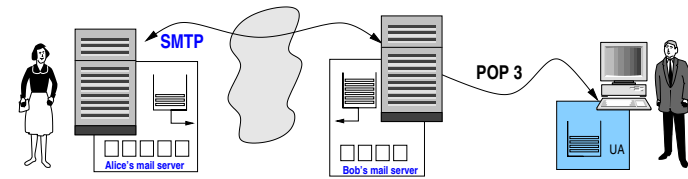
- plusieurs alternatives combinables :
  - accès direct au serveur (montage NFS ou SMB)
  - POP
  - IMAP
  - HTTP



## POP3

Post Office Protocol – Version 3 (RFC 1939)

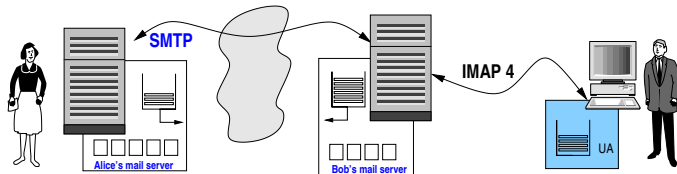
- simple
- connexion TCP sur le port 110
- trois phases :
  - autorisation (identification)
  - transaction (récupération et marquage)
  - mise à jour (suppression effective du serveur)



## IMAP4

### Internet Mail Access Protocol – version 4 (RFC 2060)

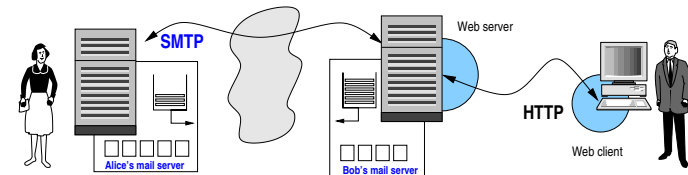
- complexe
- connexion TCP sur le port 143
- même fonctionnalité que POP avec :
  - accès par attribut (12<sup>ème</sup> e-mail d'Alice)
  - récupération de partie de message (3<sup>ème</sup> pièce jointe)
  - **synchronisation** de boîtes aux lettres



## Web-mail

### UA sur le serveur SMTP et interface Web

- comptes web spécifiques :
  - Hotmail, Yahoo!, Gmail...
- autre moyen d'accès au serveur d'entreprise ou de l'ISP :
  - horde/IMP, Squirrelmail, Zimbra...



## Messagerie et sécurité

### Les protocoles de base ne sont pas sécurisés

- échange textuel non confidentiel (contrôle et données)
- aucune authentification avec SMTP
- identifiant et mot de passe en clair avec POP et IMAP

### Quelques solutions :

- PGP (*Pretty good privacy*) en environnement hostile :
  - authentification, intégrité et confidentialité (données signées et/ou cryptées)
  - **OpenPGP** (RFC 2440) : GPG (*Gnu Privacy Guard*)
- si confiance dans le site distant, sécurisation des connexions :
  - si le site distant est accessible via SSH
    - accès à distance sur le serveur via SSH (UA textuels)
    - **tunnels SSH**
  - si clients et **serveurs** avec SSL (ou TLS)
    - POP3S (RFC 2595) : port 995
    - IMAPS (RFC 2595) : port 993
    - HTTPS pour sécuriser le Web-Mail...

## ARes : plan du cours 2/5

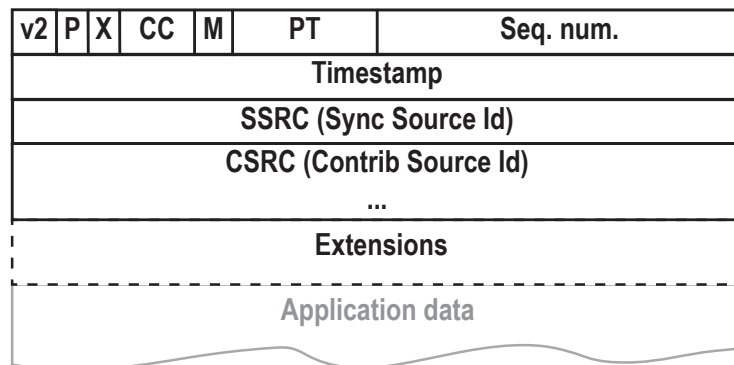
- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## Applications multimédia

Nombreux mécanismes sont associés à ces applications (ex : pour une vidéo-conférence) :

- établissement d'une session
    - préparation (médias, codecs, protocoles...)
    - annonce et invitation (news, web, e-mail...)
    - initiation (signalisation ou connexion)
  - participation
    - **émission** : codage/compression, paquets, transmission...
    - **réception** : décodage/décompression, lecture audio/vidéo...
- ➡ Besoin de protocoles spécifiques pour :
- encapsuler les données multimédia
  - gérer les participants
  - ...

## Format d'un message RTP



## RTP

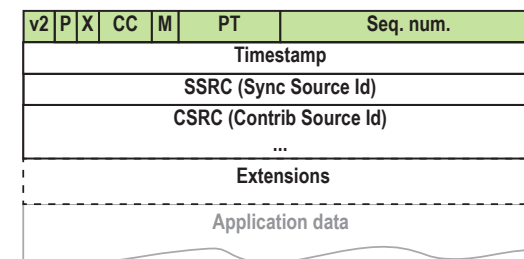
## Real-Time Transport Protocol (RFC 3550 pour la v2)

- distribution de données multimédia de bout-en-bout
- nombreux paramètres pour les applications
  - identification (type de données, participants)
  - numérotation spatiale et temporelle des paquets
  - synchronisation de flux
  - surveillance de qualité
  - communication vers un ou plusieurs destinataires



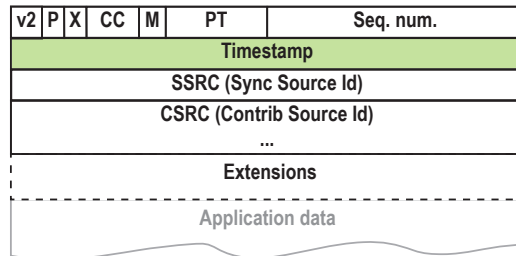
les algorithmes de codage, de synchronisation, de lecture sont implantés au niveau de l'application

## En-tête RTP



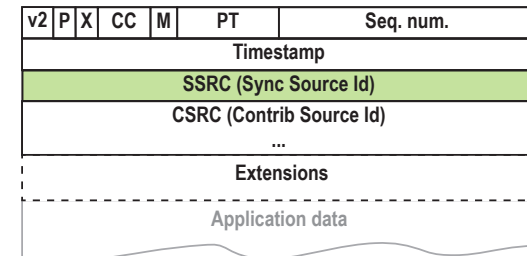
- P=1 si bourrage (padding), X=1 si extension présente
- CC : nombre d'identifiants CSRC qui suivent l'entête
- M : marqueur définit et interprété par un profil
- PT : type de données du paquet RTP (défini par un profil)
- Seq. num. : incrémenté de 1 pour chaque paquet RTP (valeur initiale aléatoire)

## En-tête RTP : Estampille temporelle



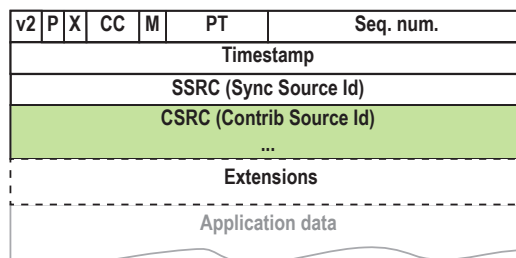
- instants d'échantillonnage du premier octet du paquet RTP
  - sont liées à une horloge qui s'incrémente de façon monotone et linéaire
  - fréquence de l'horloge dépendant du type des données (spécifiée par le profil)
- instants de présentation
  - calculés à partir d'une origine temporelle, dépend du profil
- valeur initiale aléatoire

## En-tête RTP : SSRC



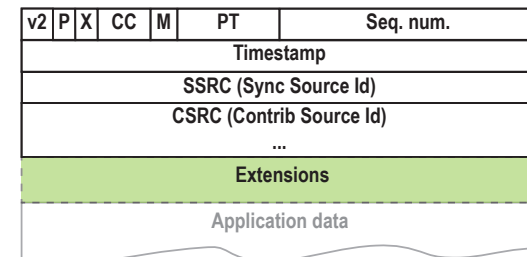
- identifie la source sur laquelle les données du paquet sont synchronisées
- à chaque SSRC correspond un interval de numéro de séquence
- choisi de manière aléatoire pour ne pas avoir 2 SSRC identiques dans la même session RTP (les implémentations de RTP doivent pouvoir gérer les collisions)
- chaque application peut avoir plusieurs SSRC

## En-tête RTP : CSRC



- liste de 0 à 15 items de 32 bits
- identifie les sources qui ont des données dans le paquet RTP.
- les CSRC sont insérés par les "mixers"
- plusieurs utilisations possibles, exemple :
  - identifier les interlocuteurs dans une session de télé-conférence (le mixer indique son SSRC et les SSRC des sources initiales deviendront des CSRC)

## En-tête RTP : CSRC



- permet de réaliser une implementation propriétaire
  - dans un cadre expérimental par exemple (pour de tester de nouveaux formats de données)
  - paramètres d'extension définis par l'implementation



## Profiles RTP audio et vidéo

Le RFC 3551 définit un profil de base sans négociation de paramètres

PT	encodage	A/V	Clock (Hz)
0	PCMU	A	8000
2	G721	A	8000
3	GSM	A	8000
5	DVI4	A	8000
7	LPC	A	8000
8	PCMA	A	8000
9	G722	A	8000
15	G728	A	8000
26	JPEG	V	90000
31	H261	V	90000...



## Autres formats d'encapsulation RTP vidéo

RFC 2435 RTP for JPEG-compressed Video  
RFC 2250 RTP for MPEG1/MPEG2  
RFC 3640 RTP for Transport of MPEG-4 Elementary Streams  
RFC 4425 RTP for Video Codec 1 (VC-1)  
RFC 4587 RTP for H.261 Video Streams  
RFC 4628 RTP for H.263 Video Streams  
RFC 4629 RTP for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)  
RFC 5371 RTP for JPEG 2000 Video Streams  
RFC 6184 RTP for H.264 Video  
RFC 6190 RTP for Scalable Video Coding (extension H264 AVC)  
RFC 6416 RTP for MPEG-4 Audio/Visual Streams  
RFC 6469 RTP for DV (IEC 61834) Video



## Autres formats d'encapsulation RTP audio

RFC 2658 RTP PureVoice(tm) Audio  
RFC 3389 RTP Comfort Noise (CN)  
RFC 3558 RTP for Enhanced Variable Rate Codecs (EVRG)  
RFC 4184 RTP for AC-3 Audio  
RFC 4298 RTP for BroadVoice Speech Codecs  
RFC 4348 RTP for VMR-WB Audio Codec  
RFC 4352 RTP for AMR-WB+ Audio Codec  
RFC 4867 RTP for AMR (Adaptive Multi-Rate) and AMR-WB Audio Codecs  
RFC 4598 RTP for Enhanced AC-3 (E-AC-3) Audio  
RFC 5188 RTP for the Enhanced Variable Rate Wideband Codec (EVRG-WB)  
RFC 5215 RTP for Vorbis Encoded Audio  
RFC 5219 A More Loss-Tolerant RTP Payload Format for MP3 Audio  
RFC 5391 RTP for ITU-T Recommendation G.711.1  
RFC 5404 RTP for G.719  
RFC 5574 RTP for the Speex Codec  
RFC 5577 RTP for ITU-T Recommendation G.722.1 (wideband)  
RFC 5584 RTP for the Adaptive TRansform Acoustic Coding (ATRAC) Family  
RFC 5993 RTP for GSM Communications Half Rate (GSM-HR)  
RFC 6884 RTP for the Enh. Var. Rate Narrowband-Wideband Codec



## RTCP

*Real-Time Transport Control Protocol (RFC 3550)*

- totalement **intégré à RTP**
  - même RFC 3550
- transmet les **informations de session**
  - synchronisation
  - participants
  - ...
- fournit des **statistiques** sur la qualité de la session
- transmet des informations de contrôle sur la session
  - ex : identifier un participant sur les écran des participants



## RTSP

## Real Time Streaming Protocol (RFC 2326)

- contrôle **hors bande** de la diffusion (habituellement TCP port 554)
- identification de la ressource via URL (ex : rtsp://media.upmc.fr:554/videofile)
- fonctionnalités typiques d'un lecteur vidéo :
  - lecture/pause
  - avance rapide
  - accès à une position temporelle...
- ne définit aucun mécanismes de codage pour la vidéo/audio
- ne définit pas la méthode d'encapsulation des données
- n'impose aucun mode de mise en mémoire tampon du lecteur média



## RTSP : exemple

```

C->W: GET /concert.sdp HTTP/1.1
Host: www.upmc.fr

W->C: HTTP/1.1 200 OK
Content-Type: application/x-rtsp

<session>
  <track src="rtsp://live.upmc.fr/concert/audio">
</session>

C->M: DESCRIBE rtsp://live.upmc.fr/concert/audio RTSP/1.0
CSeq: 1

M->C: RTSP/1.0 200 OK
CSeq: 1
Content-Type: application/sdp
Content-Length: 44

v=0
o=- 2890844526 2890842807 IN IP4 132.227.24.202
s=RTSP Session
m=audio 3456 RTP/AVP 0
a=control:rtsp://live.upmc.fr/concert/audio
c=IN IP4 224.2.0.1/16

C->M: SETUP rtsp://live.upmc.fr/concert/audio RTSP/1.0
CSeq: 2
Transport: RTP/AVP;multicast

M->C: RTSP/1.0 200 OK
CSeq: 2
Transport: RTP/AVP;multicast;destination=224.2.0.1;
port=3456-3457;ttl=16
Session: 0456804596

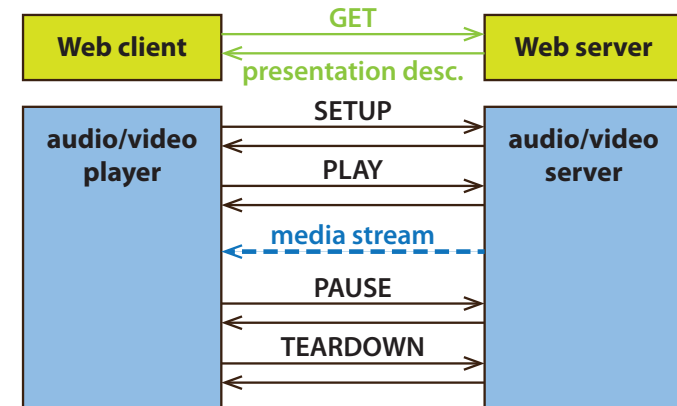
C->M: PLAY rtsp://live.upmc.fr/concert/audio RTSP/1.0
CSeq: 3
Session: 0456804596

M->C: RTSP/1.0 200 OK
CSeq: 3
Session: 0456804596

```



## RTSP : fonctionnement



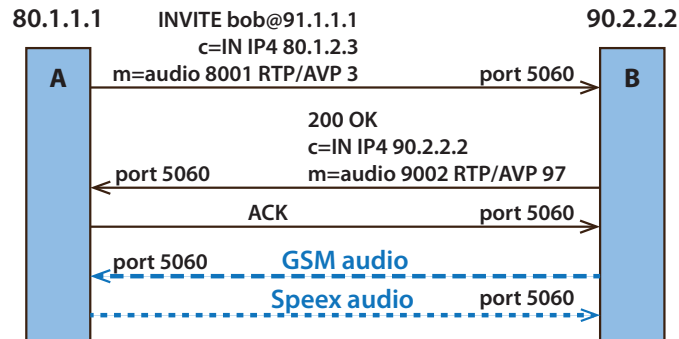
## SIP

## Session Initiation Protocol (RFC 3261)

- mise en place d'appel
  - **notification** (intention d'établir un appel)
  - **négociation** (codages, média...)
  - **terminaison** (fin de l'appel/session)
- utilisateurs identifiés par noms et/ou e-mails (pas de numéros de téléphone)
- correspondance entre identifiants et adresses IP
- gestion des appels
  - **ajout de nouveaux média** pendant l'appel
  - **changement de codage** en cours
  - **invitation** de participants
- protocole de base de l'IMS (IP Multimedia Subsystem du 3GPP)



## SIP : fonctionnement



- **négoce** du codec (réponse 606 not acceptable avec la liste des codec supportés)
- **rejet** d'appel (réponse busy, gone, payment, forbidden...)
- **transmission** (données multimédia envoyées avec RTP ou un autre protocole)

## SIP : exemple

```

INVITE sip:auto@localhost SIP/2.0
Via: SIP/2.0/UDP 127.0.0.1:13764;rport
Max-Forwards: 70
Contact: <sip:matthew@127.0.0.1:13764>
To: "sip:auto@localhost" <sip:auto@localhost>
From: "Olivier" <sip:olivier@upmc.fr>;tag=5c7cdb68
Call-ID: NmNhYWNhMjY0Y2M0OTc4YTl2MzgZTNlYTRhZTMxNTE.
CSeq: 1 INVITE
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
MESSAGE, SUBSCRIBE, INFO
Content-Type: application/sdp
User-Agent: eyeBeam release 1014g stamp 44944
Content-Length: 389

v=0
o=- 7 2 IN IP4 132.227.61.199
...

SIP/2.0 200 OK
Via: SIP/2.0/UDP 127.0.0.1:13764;rport=13764;received=127.0.0.1
To: "sip:auto@localhost" <sip:auto@localhost>;tag=tCAED3F5A
From: "Olivier" <sip:olivier@upmc.fr>;tag=5c7cdb68
Call-ID: NmNhYWNhMjY0Y2M0OTc4YTl2MzgZTNlYTRhZTMxNTE.
CSeq: 1 INVITE
Contact: <sip:127.0.0.1:5060>
Content-Type: application/sdp
Content-Length: 271

v=0
o=olivier 1208261984604 1208261984604 IN IP4 127.0.0.1
...

ACK sip:127.0.0.1:5060 SIP/2.0
Via: SIP/2.0/UDP 127.0.0.1:13764;rport
Max-Forwards: 70
Contact: <sip:matthew@127.0.0.1:13764>
To: "sip:auto@localhost" <sip:auto@localhost>;tag=tCAED3F5A
From: "Olivier" <sip:olivier@upmc.fr>;tag=5c7cdb68
Call-ID: NmNhYWNhMjY0Y2M0OTc4YTl2MzgZTNlYTRhZTMxNTE.
CSeq: 1 ACK
Content-Length: 0
  
```

## SIP : conversion de nom et localisation

Comment faire correspondre l'identifiant SIP (nom/email) à l'adresse IP de l'appelé ?

- l'appelé peut être mobile
- il peut obtenir une adresse IP dynamique/privée
- il peut avoir plusieurs appareils IP (ordinateur, tablette, smartphone... etc.)

Avec quelle flexibilité ?

- en fonction du temps et du lieu
- en fonction de l'état de l'appel et/ou de l'appelée (transfert d'appel, etc.)

➡ ce service est assuré par des serveurs SIP :

- serveur **SIP Registrar**
- serveur **SIP Proxy**

## SIP : Registrar

Au démarrage du client SIP de l'utilisateur :

➡ émission d'un message REGISTER vers le **serveur SIP Registrar** de l'utilisateur.

```

REGISTER sip:upmc.fr SIP/2.0
Via: SIP/2.0/UDP 132.227.61.205:5061;rport;branch=z9hG4bKDC8595CD770E4317ACBC3
From: Ivan <sip:olivier@upmc.fr>;tag=1516659370
To: Ivan <sip:olivier@upmc.fr>
Contact: "Ivan" <sip:olivier@132.227.61.205:5061>
Call-ID: 46E1C3CB36304F84A020CF6DD3F96461@Verso.com
CSeq: 37764 REGISTER
Expires: 1800
Max-Forwards: 70
User-Agent: LIP6-SIP-Phone v0.9
Content-Length: 0
  
```

## SIP : Proxy

L'utilisateur envoie un message INVITE vers son **Proxy SIP**

- indique l'adresse SIP du destinataire (sip:olivier@upmc.fr)

Le serveur Proxy SIP est responsable de l'acheminement des messages SIP

- éventuellement à travers plusieurs proxys.
- l'appelé envoie la réponse à travers le(s) même proxy(s)

ensuite, le proxy de l'appelé renvoie une réponse SIP à l'appelant contenant son l'adresse IP

Le service fournis par le proxy SIP est similaire à celui d'un serveur DNS local



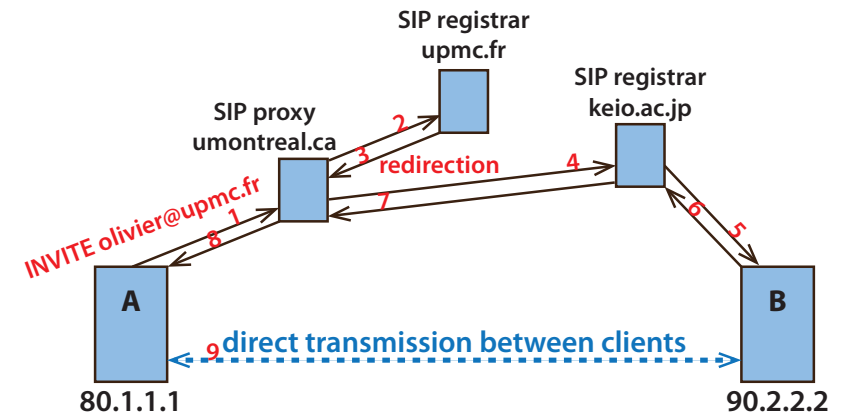
## SDP

*Session Description Protocol (RFC 4566)*

- description des sessions multimedia** pour leur initialisation
- présente les détails du média à transmettre/recevoir aux participants
  - adresses, identifiants, codecs, métadatas, etc.
- ce n'est pas un protocole de transmission/transport
  - seulement de la description**
- peut être utilisé sur un protocole de transport quelconque
- peut être intégré à tout protocoles d'initialisation/signalisation
  - RTSP, SIP, etc.
- format standard** de présentation
  - indication de contenu d'une description SDP avec Content-Type: application/sdp
  - la description SDP est une suite de ligne de texte de type <type>=<value>



## SIP : exemple



## SDP : exemple

```
v=0
o=fourmaux 2990844526 2990842807
    IN IP4 132.227.63.51
s=SDP Seminar
i=A Seminar on SDP
u=http://www.upmc.fr/sem/sdp.pdf
e=olivier.fourmaux@upmc.fr
c=IN IP4 224.2.17.12/127
t=2973397496 2973404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 99
a=rtpmap:99 h263-1998/90000
```

Types SDP possibles :

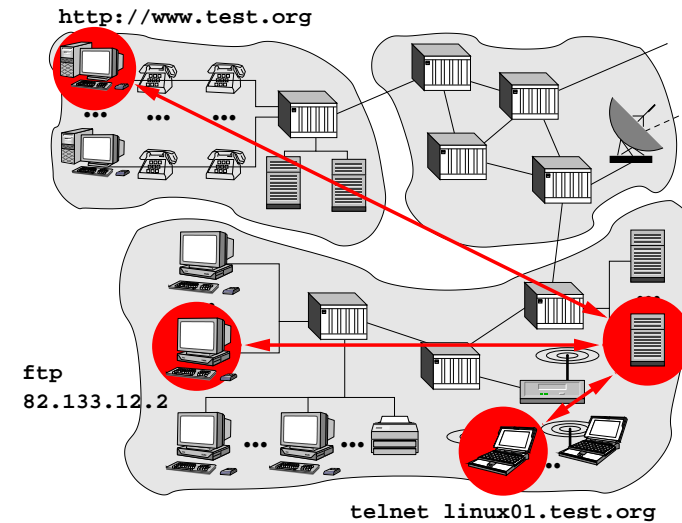
```
v= (protocol version)
o= (originator and session identifier)
s= (session name)
i= (session information)
u= (URI of description)
e= (email address)
p= (phone number)
c= (connection information)
b= (0 or more bandwidth information lines)
t= (time the session is active)
r= (0 or more repeat times)
z= (time zone adjustments)
k= (encryption key)
a= (0 or more session attribute lines)
0 or more media descriptions
m= (media name and transport address)
i= (media title)
c= (connection information)
b= (0 or more b.w. information lines)
k= (encryption key)
a= (0 or more media attribute lines)
```



## ARes : plan du cours 2/5

- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau

## Correspondance noms – adresses



## Annuaire

Conversion des noms littéraux des hôtes de l'Internet en adresses numériques

- initialement
  - un fichier
  - espace de "nommage" à plat
  - gestion centralisée par un *NIC (Network Information Center)*
- actuellement : **DNS**
  - base de données **distribuée**
  - espace de "nommage" **hiérarchique**
    - décorrélé de la topologie physique
  - système contrôlé par l'*InterNIC* (1992-1998) et puis l'*ICANN (Internet Corporation for Assigned Names and Numbers)* et ses nombreux délégués
    - délégation hiérarchique (proche de celle du "nommage")
    - taille des délégations raisonnables
  - protocole d'échange...

## DNS (*Domain Name System*)

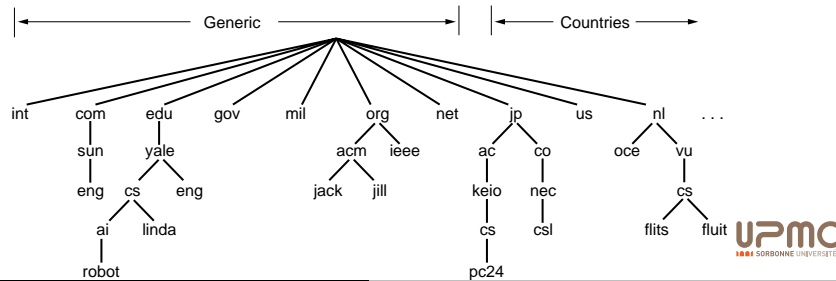
Annuaire standard de l'Internet (RFC 1034 et RFC 1035)

- espace de "nommage" hiérarchique et système de délégation
- **serveurs de noms** (serveurs DNS)
  - composants physiques de la **hiérarchie** supportant la base distribuée
  - gèrent les requêtes DNS
  - transport sur **UDP** ou TCP, port **53**
  - les applications y accèdent à travers le **resolver** (UNIX) :
    - gethostbyname (3), gethostbyaddr (3)
- services :
  - *name resolving*
  - *host aliasing*
  - *mail server aliasing*
  - *load distribution...*
- exemple :
  - BIND (*Berkeley Internet Name Domain*)
  - named (UNIX)

## DNS : Espace de "nommage"

Système de "nommage" hiérarchique

- structure arborescente (~ système de fichier Unix)
- label d'un nœud : 63 car. max. (A..Za..z- insensible à la casse)
- **domain name** = liste des labels en parcourant l'arbre vers la racine (255 car. max. au total et "." séparateur de label) :
  - absolu (**FQDN**) : pc24.cs.keio.ac.jp.
  - les noms relatifs sont gérés localement (hôte)



Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

DNS : ccTLD (*country code Top Level Domain*)

ccTLD (ISO 3166)	240 countries and external territories
.ac	Ascension Island
.af	Afghanistan
.aq	Antarctica (-60°S)
.eu	European Union
.fr	France
.gf	French Guiana
.gp	Guadeloupe
.mq	Martinique
.pf	French Polynesia + Clipperton
.pm	Saint-Pierre and Miquelon
.re	Réunion
.tf	TAAF
.ru	Russia (+.su)
.tv	Tuvalu
.uk	United Kingdom (+.gb)
.us	United States
.zw	Zimbabwe

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

DNS : gTLD (*generic Top Level Domain*)

.aero	2001	Air-transport industry *	SITA
.asia	2006	Asia-Pacific region *	Afilias
.biz	2001	Unrestricted	NeuLevel
.cat	2005	Catalan lingu. & cult.*	Asso. puntCAT
.com/.net	1985	Unrestricted	VeriSign
.coop	2001	Cooperative *	DotCooperation
.edu	1985	(US) educational inst. *	VeriSign
.gov	1985	US government *	US Admin.
.info/.org	01/85	Unrestricted	Afilias
.int	1988	Internat. organisations	ICANN
.job	2005	Human resrc. managment*	Employ Media
.mil	1985	US military *	US DoD NIC
.mobi	2005	Mobile device use *	Mobi JV
.museum	2001	Museums *	MuseDoma
.name	2001	Individuals	VeriSign
.pro	2001	Professionals	RegistryPro
.tel	2005	Internet Tel. serv.*	Telnic Limited
.travel	2005	Travel industry*	Tralliance Corp.

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : Nouveau TLD depuis 2012

En 2012, l'ICANN, a autorisé ~2000 nouveaux TLD pour 2013-14

- Grande diversité de TLD :
  - marques (.google, .danone, etc.)
  - communautés (.archi, .immo, etc.)
  - zones géographiques (.paris, .nyc, etc.)
  - langues non ASCII (.移动(.xn-6frz82g, idéogramme pour "mobile"), etc.)
  - inclassable (.photo, .wtf, etc.)

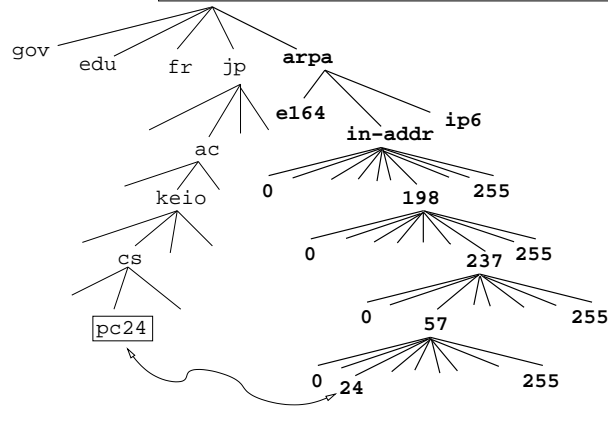
Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

DNS : Domaine .arpa

Résolution : `pc24.cs.keio.ac.jp. ➡ ?`

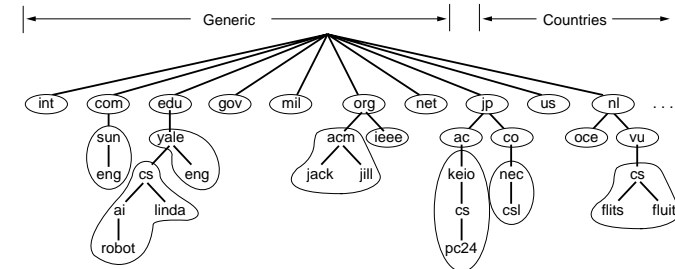
Résolution **inverse** : 24.57.237.198.in-addr.arpa. ➡ ?



## DNS : zones (1)

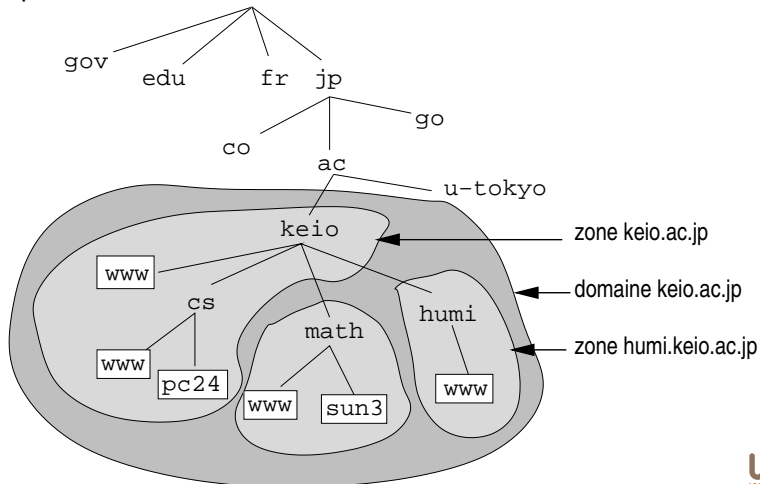
ICANN gère la racine et délègue les TLDs à des *domain name registry*

- zones (sous-arbres de l'arbre DNS) administrés séparément
  - (~ partitions physiques d'un système de fichier Unix)
  - délégation des noms de sous-domaines correspondants
  - exemple : `keio.ac.jp`.
- des **serveurs de noms** y sont associés




## DNS : zones (2)

Ne pas confondre zone et domaine !



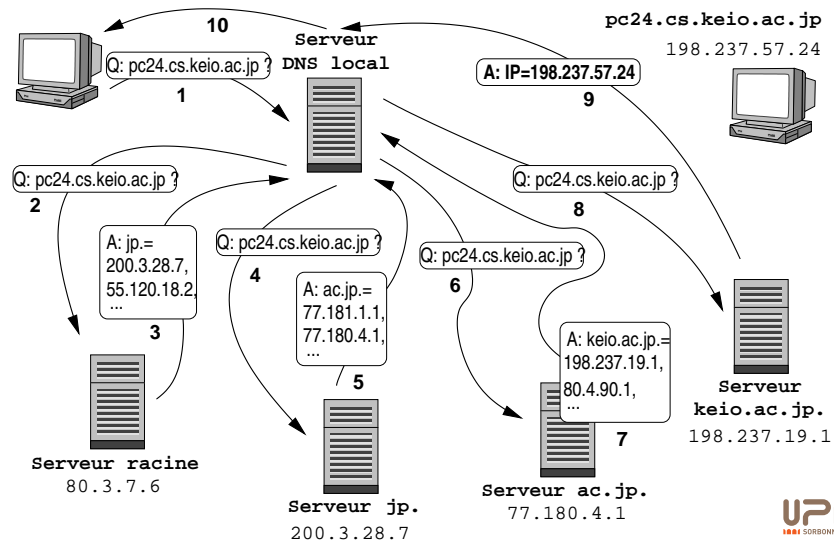
## DNS : serveurs de noms

## Différents types de serveurs de noms

- serveurs de référence d'une zone :
    - un **primaire** (*primary name server*)
      - informations de référence (*authoritative records*)
      - connaissance de ses descendants (délégations)
      - initialisation locale (disque)
    - un ou plusieurs **secondaire** (*secondary name server*)
      - redondance : complètement séparé du primaire
      - initialisation et m-à-j. à partir du primaire (**transfert de zone**)
    - physiquement indépendant de la zone
  - serveurs locaux (accès au service)
    - résolution *top-down* (des TLD vers les sous-domaines)
    - connaissance des serveurs racines (*root name server*)
      - 1 primaire et 12 secondaires, haute disponibilité (anycast)
      - config. en dur (<ftp.rs.internic.net/domain/named.root>)
    - requêtes **récurrentes** ou **itératives**
- 



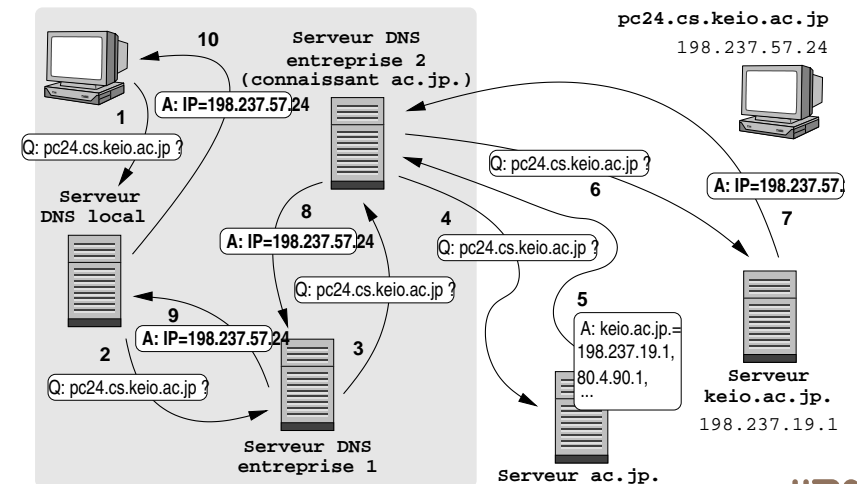
## DNS : requête itérative



Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : requête récursive



Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : performances

Capacité du système DNS à supporter la charge ?

- problèmes liés à la consultation systématique de la racine
  - ne tient pas compte de la localité des requêtes
    - serveur local généralement distinct du serveur de référence
  - charge sur les serveurs racines
    - combien de requêtes pour tout l'Internet ?
  - disponibilité des serveurs racines
    - passage obligé pour toute requête
- utilisation de **cache**
  - informations de seconde main (*non-authoritative records*)
  - réponses d'un serveur de référence inclue un délai de validité (TTL)
    - réponses pour les TLD sur les serveurs racines valide 48h
      - 100.000 requêtes par secondes (2005)

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : format général du message

0	15	16	bit 31	flags :
identificateur		flags		● (
nombre de questions		nombre de réponses		● -
nombre de serveurs		nombre d'info. add.		● (
				● R
Questions				● (
				● R
Champs des réponses				● (
				● R
Champs des serveurs de référence				● (
				● R
Champs des informations additionnelles				● R
				● R

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application



## DNS : format d'une question

0	15	16	bit 31
Nom (non aligné sur 32bits)			
Type		Classe	

- **Nom** : N octets, chaque nom de label est précédé par un octet indiquant le nombre de caractères (si >0x3F alors si 0xC0ZZ = renvoi à ZZ octets du début du message). Terminé par 0x00.

4, 'p', 'c', '2', '4', 2, 'c', 's', 4, 'k', 'e', 'i', 'o', 2, 'a', 'c', 2, 'j', 'p', 0

- **Type** (16 bits) :

val	nom	description	val	nom	description
1	<b>A</b>	adr. IPv4	13	<b>HINFO</b>	info sur l'équip.
2	<b>NS</b>	nom serv.	15	<b>MX</b>	serveur messag.
5	<b>CNAME</b>	alias	28	<b>AAAA</b>	adresse IPv6
6	<b>SOA</b>	zone gérée		...	
12	<b>PTR</b>	point. nom	255	<b>*</b>	tt types (quest.)

- **Classe** (16 bits) : 1 = Internet

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : format d'un champ réponse

0	15	16	bit 31
Nom (non aligné sur 32bits)			
Type		Classe	
TTL			
Taille des données (o.)			
Données			

- **Nom, Type, Classe** : idem
- **TTL** (32 bits) : validité en secondes
- **Taille des données** (16 bits) : en octets
- **Données** (N octets sans bourrage) :
  - Nom (chaîne codée comme pour une question) NS, CNAME
  - Adresses (valeur numérique) A sur 4 octets, AAAA sur 16...

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : annuaire inversé

Conversion des adresses numériques en noms littéraux

- requêtes de type **pointeur de nom** (PTR)
  - adresse IPv4
    - 198.237.57.24
  - conversion dans le domaine **in-addr.arpa**
    - 24.57.237.198.in-addr.arpa
    - souvent utilisé pour vérifier les droits d'accès

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : obtention d'une délégation

Pour être référence pour un sous domaine officiel :

- réservation du nom du domaine auprès d'un *domain name registrar*
- mise en place de serveurs conformes à la norme DNS
  - information de référence de la zone
    - réplication dans au moins un serveur secondaire
  - si sous délégations :
    - connaissance des serveurs descendants
  - si gestion des adresses IP correspondantes :
    - information de référence des pointeurs de nom

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Architecture des Réseaux (ARes) 2/5 : Application

## DNS : modification dynamique

## Dynamique DNS (RFC 2136)

- pour fonctionner avec l'auto-conf. des hôtes (DNS local) :
  - *update*
  - *notification*

- problèmes de sécurité...

## Service DNS dynamique (prestataire externe)

- pour fonctionner avec une adresse dynamique (accès résidentiels) :
  - serveur : dyndns.org, no-ip.org...
  - client spécifique indiquant le changement d'adresse (*host/setupbox*)
  - délégation virtuelle (sous domaine de 3ème niveau)
    - toto123.myftp.biz
    - toto123.blogspot.org
    - toto123.homelinux.org
    - toto123.dyn-o-saur.com
    - toto123.endofinternet.net...



## DNS : exemple

```
Unix> dig www.math.keio.ac.jp

;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 11895
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 3, ADDITIONAL: 4

;; QUESTION SECTION:
;www.math.keio.ac.jp.      IN      A

;; ANSWER SECTION:
www.math.keio.ac.jp.      3600    IN      CNAME   sun3.math.keio.ac.jp.
sun3.math.keio.ac.jp.     3600    IN      A        131.113.70.3

;; AUTHORITY SECTION:
math.keio.ac.jp.          3600    IN      NS       relay.math.keio.ac.jp.
math.keio.ac.jp.          3600    IN      NS       ns.st.keio.ac.jp.
math.keio.ac.jp.          3600    IN      NS       ns0.sfc.keio.ac.jp.

;; ADDITIONAL SECTION:
relay.math.keio.ac.jp.    3600    IN      A        131.113.70.1
ns.st.keio.ac.jp.        127     IN      A        131.113.1.8
ns0.sfc.keio.ac.jp.      1199    IN      AAAA     3ffe:501:1085:8001::121
ns0.sfc.keio.ac.jp.      2358    IN      A        133.27.4.121

;; Query time: 577 msec MSG SIZE rcvd: 206
```



## DNS : sécurité

## Pas de sécurité dans le protocole de base (RFC 3833)

- interception / modification de message DNS
- faux messages (*DNS cache poisoning*)
- déni de service...

## DNSSEC (RFC 4033 à 4035 + RFC 4310 + RFC 4641)

- extension du système DNS permettant :
  - authentification de l'origine des données
  - authentification du déni d'existence
  - intégrité des données
- obligatoire pour sécuriser les *DNS update*
  - attention aux extensions propriétaires...



## ARes : plan du cours 2/5

- 1 Applications historiques
  - introduction
  - connexion à distance
  - transfert de fichiers
- 2 Applications principales
  - World Wide Web
  - messagerie électronique
  - multimedia
- 3 Applications support
  - annuaire (DNS)
  - administration de réseau



## administration de réseau

Développement du réseau (nombreux équipements et machines à gérer)

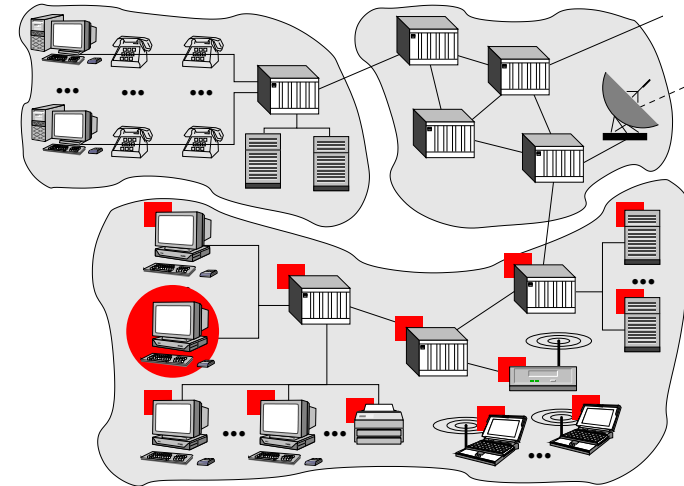
### Besoins :

- surveillance du réseau
  - détection de pannes
  - mesure de performance
- intervention sur le matériel
  - activation (interface...)
  - configuration (table de routage...)
- poste de contrôle centralisé

### Contraintes :

- matériels hétérogènes
  - routeurs, hubs, switches...
  - ordinateurs, imprimantes, sondes...
- constructeurs multiples
- localisation géographique distante

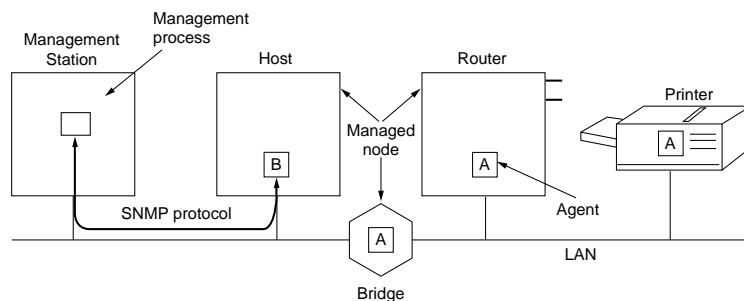
## Equipements administrables



## Administration TCP/IP

Comment gérer les machines en environnement TCP/IP ?

- instrumentation des équipements (**agents**)
- logiciels de supervision (HP Openview, Cisco Works, Nagios...)
- protocole de gestion ➡ SNMP



## SNMP : principe

Informations réseau stockées dans deux types de bases :

- **bases agents** (dans les équipements) : Les valeurs sont directement couplées avec les registres internes
- **base centralisée** (plateforme de supervision) : dernières valeurs transmises et historique (statistiques)

Standardisation (pour échange en milieu hétérogène)

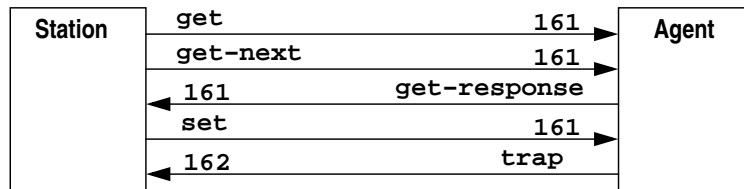
- désignation et type d'information définis par des **MIB**
- structures communes et nomenclature définies dans la **SMI**
- représentation des données en **ASN.1**
- protocole **SNMP** entre la station et les agents permettant :
  - **lecture/écriture** de variables sur des éléments gérés
  - **alarmes** non sollicitées
  - **parcours** de listes de variables dans les éléments gérés

➡ vision agrégée globale

## SNMP : commandes

La richesse est dans la MIB !

- seulement 5 commandes **simples**
- utilisation sur **UDP** port **161** et **162**

SNMP : SMI (*Structure for Management Information*)

- les info. respectent les types de la SMIv1 (RFC 1155 et 1212)

NULL	pas de valeur
INTEGER	entier signé non limité
Counter	entier positif (0 à $2^{32} - 1$ ) bouclant
Gauge	entier positif (0 à $2^{32} - 1$ ) borné
TimeTicks	durée en centième de secondes
OCTET STRING	chaîne d'octets non limitée
DisplayString	chaîne codée en NVT de 255 car. max.
IpAddress	chaîne de 4 octets
PhyAddress	chaîne de 6 octets
OBJECT ID.	identifiant numérique...
SEQUENCE	structure d'éléments nommés
SEQUENCE OF	vecteur d'éléments identiques

## SNMP : format des messages

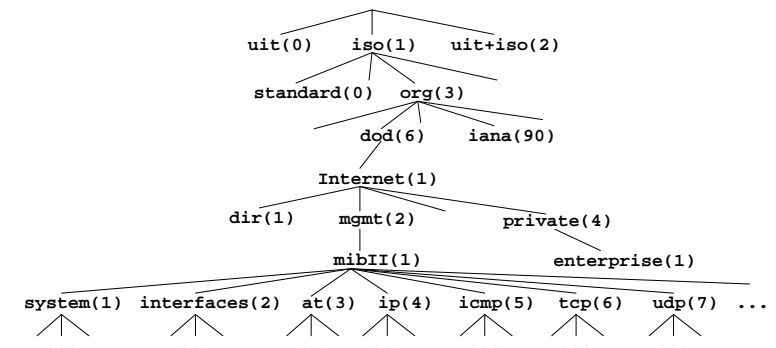
version	communauté	type PDU	ident req.	erreur status	erreur index	nom	valeur	nom	valeur	...
---------	------------	----------	------------	---------------	--------------	-----	--------	-----	--------	-----

- **version** : version SNMP - 1 (0 ~ SNMPv1)
- **communauté** : chaîne de caractères autorisant l'accès
  - généralement "public"
- **type PDU** : 0 (get), 1 (get-next), 2 (set), 3 (get-response)
  - le message de type 4 (trap) sera présenté dans la suite...
- **ident. req.** : fait correspondre requêtes et réponses
- **erreur status** et **erreur index** : type d'erreur concernant la variable référencée par l'indexage (0 ~ pas d'erreur)
- **nom** et **valeur** : variables transportées

Les tailles des champs ne sont pas précisées car la structure du message est décrite en ASN.1 avec encodage BER.

OID (*Object Identifier*)

- arbre de "nommage" (référencement **unique** d'un objet)
  - les objets de l'Internet commencent par 1.3.6.1.



## SNMP : MIB (Management Information Base)

- les groupes d'objets définis dans la MIB II (RFC 1213) :

1.3.6.1.2.1.1 system  
1.3.6.1.2.1.2 interfaces  
1.3.6.1.2.1.3 at  
1.3.6.1.2.1.4 ip  
1.3.6.1.2.1.5 icmp  
1.3.6.1.2.1.6 tcp  
1.3.6.1.2.1.7 udp  
1.3.6.1.2.1.8 egp  
1.3.6.1.2.1.10 transmission  
1.3.6.1.2.1.11 snmp

- d'autres groupes, ou sous-groupes sont définis (autres RFC) :

1.3.6.1.2.1.17 bridge  
1.3.6.1.2.1.43 printer ...

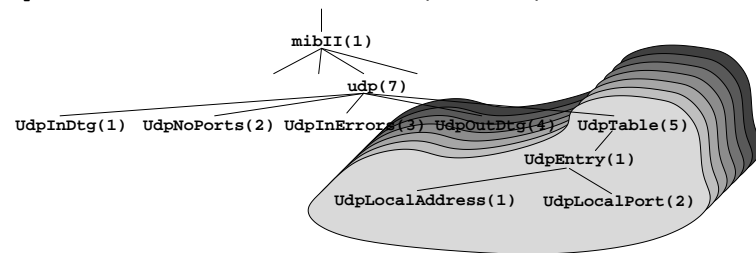
Ces groupes contiennent des variables simples ou tables

## MIB : variable table

Dans le groupe UDP, 1 variable table :

- udpTable indique les ports scrutés sur l'équipement
- udpTable est un **vecteur** de structures udpEntry

udpLocalAddress IpAddress ro adresse IP locale  
udpLocalPorts [0..65535] ro port correspondant



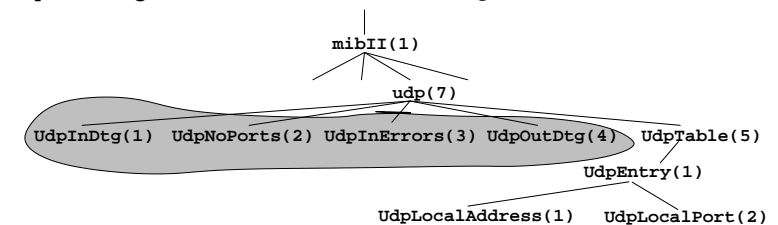
- l'**index** dans la table est ici  
udpLocalAddress.udpLocalPorts
  - l'index est précisé à la conception de la MIB

## MIB : variable simple

Dans le groupe UDP, 4 variables simples :

- la MIB II fait correspondre des types SMI

udpInDatagrams Counter ro nb datagrammes délivrés aux applications  
udpNoPorts Counter ro nb datagrammes sans application en attente  
udpInErrors Counter ro nb datagrammes non délivrables  
udpOutDatagrams Counter ro nb datagrammes émis



## SNMP : référencement des variables

Référencement des variables :

- simples : ajout de ".0" à la fin
- tables : ajout des valeurs des champs index
  - parcours des OID de la table dans l'ordre **lexicographique**

nom abrégé	OID	valeur
udpInDatagrams.0	1.3.6.1.2.1.7.1.0	17625
udpLocalAddress.0.0.0.0.53	1.3.6.1.2.1.7.5.1.1.0.0.0.0.53	0.0.0.0
udpLocalAddress.0.0.0.0.161	1.3.6.1.2.1.7.5.1.1.0.0.0.0.161	0.0.0.0
udpLocalPort.0.0.0.0.53	1.3.6.1.2.1.7.5.1.2.0.0.0.0.53	53
udpLocalPort.0.0.0.0.161	1.3.6.1.2.1.7.5.1.2.0.0.0.0.161	161

- le référencement permet de spécifier les objets dans les messages UDP
  - seuls les OID et les valeurs sont transportées

## SNMP : commande get-next

Opérateur de parcours dans l'ordre **lexicographique** des OIDs :

- renvoie la prochaine référence terminale
  - get-next udp ➡ udpInDatagrams.0 = 17625
- permet le parcours des variables...
  - get-next udpInDatagrams.0 ➡ udpNoPorts.0 = 0
- ... et des tables
  - get-next udpTable
    - ➡ udpLocalAddress.0.0.0.0.53 = 0.0.0.0
  - get-next udpLocalAddress.0.0.0.0.53
    - ➡ udpLocalAddress.0.0.0.0.161 = 0.0.0.0
  - get-next udpLocalAddress.0.0.0.0.161
    - ➡ udpLocalPort.0.0.0.0.53 = 53 ...
- fin du tableau lors du changement de nom :
  - get-next udpLocalPort.0.0.0.0.161
    - ➡ snmpInPkts.0 = 12



## SNMP : Trap

Envoi d'un message SNMP de l'agent vers l'admin. sur le **port 162**

version	communauté	type	entreprise	adr. agent	type trap	code entr.	estamp. temp.	nom	valeur	...
		= 4								

- entreprise : identificateur du créateur de l'agent
  - OID débutant par 1.3.6.1.4.1.
- adr. agent : adresse IP de l'agent
- type trap :
 

0	coldStart	agent initialisé
1	warmStart	agent réinitialisé
2	linkDown	interface désactivée
3	linkUp	interface activée
...		
6	entr. specific	voir le champ code entr.
- code entr. : sous-code du trap spécifique à l'entreprise
- estamp. temp. : valeur indiquant le nombre de centièmes de secondes depuis le démarrage de l'agent



## Syntaxe abstraite ASN.1

Couche 6 de l'OSI (définie par l'UIT, recommandation X.680)

- propriétés :
  - représentation universelle d'informations
  - type associé aux données
  - désignation par un identificateur unique (OID)
  - notation de type BNF
- description des informations échangées par SNMP :

```

RFC1157-SNMP DEFINITIONS ::= BEGIN
  Message ::= SEQUENCE {
    version      INTEGER {version-1(0)},
    community    OCTET STRING,
    data         ANY
  }
  PDUs ::= CHOICE {
    get-request   GetRequest-PDU,
    get-next-request GetNextRequest-PDU,
    get-response  GetResponse-PDU,
    set-request   SetRequest-PDU,
    trap         Trap-PDU
  }
  ...

```



## ASN.1 : PDU

Message get écrit en ASN.1 :

```

getRequest-PDU ::= [0]
  IMPLICIT SEQUENCE {
    request-id    INTEGER,
    error-status  INTEGER {
      noError(0), tooBig(1),
      noSuchName(2), badValue(3),
      readOnly(4), genErr(5), -- always 0
    }
    error-index   INTEGER, -- always 0
    variable-bindings SEQUENCE OF
      SEQUENCE {
        name      ObjectName,
        value     ObjectSyntax
      }
  }

```



## SNMP : encodage BER

Encodage **TLV** (Type, Longueur, Valeur)

- types (1o) : les 2 bits de poids fort déterminent la catégorie

0x02	INTEGER
0x04	OCTET STRING
0x05	NULL
0x06	OBJECT IDENTIFIER
0x30	SEQUENCE

- UNIVERSAL (00)

0x40	IpAddress
0x41	Counter
0x42	Gauge
0x43	TimeTicks

- APPLICATION (01)

- CONTEXT (10)

- PRIVATE (11)

- longueur des données (1 octet si < 0x80, sinon norme X.208)
  - longueur 49 ➡ 0x31, longueur 242 ➡ 0x8200F2...
- données (valeur)
  - les OID (avec les valeurs entières successives A.B.C.D...) sont



## MIB RMON

Remote **MON**itoring (RFC 2819 - STD 59)

Sonde pour obtenir des **statistiques** sur un réseau administré

- 9 groupes :
  - statistiques sur Ethernet (table de 21 attributs)
  - équipements du réseau (adresses observées...)
  - matrice de statistiques (entre deux stations)
  - capture de trames
  - ...
- nombreuses extensions
  - identification de protocoles pour RMON (RFC 2895, 2896)
  - RMON pour réseaux commutés (SMON : RFC 2613)
  - gestion des interface pour RMON (IFTOPN : RFC 3144)
  - RMON pour les services différenciés (DSMON : RFC 3287) ...



## SNMP : exemple

```

0020                                30 82 00 f2 02 01  J...D... ..0....
0030 00 04 06 70 75 62 6c 69 63 a2 82 00 e3 02 01 01  ...publi c.....
0040 02 01 00 02 01 00 30 82 00 d6 30 82 00 0d 06 08  .....0. ..0....
0050 2b 06 01 02 01 02 01 00 02 01 03 30 82 00 0f 06  +..... ..0....
0060 0a 2b 06 01 02 01 02 02 01 08 01 02 01 01 30 82  .+..... .....0.
0070 00 0f 06 0a 2b 06 01 02 01 02 02 01 08 02 02 01  ....+... .....
0080 02 .. ..
0100                                .. .. 30 82 00 10  .... C...,0...
0110 06 0a 2b 06 01 02 01 02 02 01 09 01 43 02 01 2c  ..+..... ...C...

```



## Autres MIB IETF (1)

MIB Imprimante *Printer MIB* (RFC 1759 - RFC 3805)

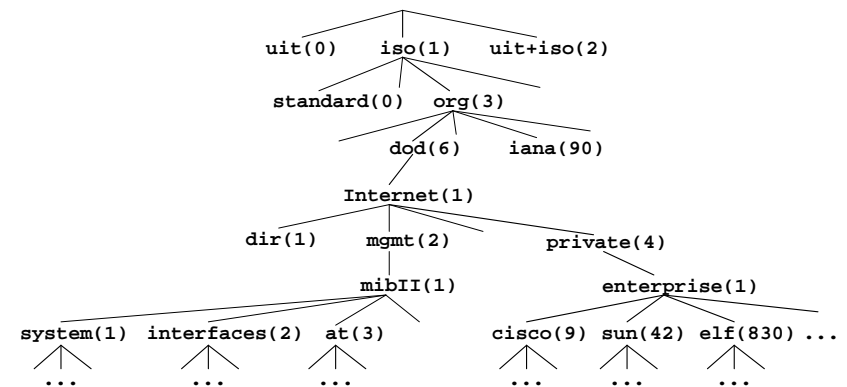
- 274 Objets (228 OID dont 16 tables)
  - 20 groupes :
    - groupe général
    - groupe des entrées
    - groupe des sorties
    - groupe des dimensions de sortie
    - groupe de la couverture
    - groupe des fournitures
    - groupe des colorants ...



## Autres MIB IETF (2)

RFC1230 : IEEE 802.4 Token Bus MIB	RFC5240 : PIM Bootstrap Router MIB
RFC1381 : MIB Extension for X.25 LAPB	RFC5324 : MIB for Fibre-Channel Security Protocols
RFC1559 : DECnet Phase IV MIB Extensions	RFC5428 : Management Event MIB for PacketCable
RFC1593 : SNA APPN Node MIB	RFC5519 : Multicast Group Membership Discovery MIB
RFC1611 : DNS Server MIB Extensions	RFC5525 : Reliable Server Pooling MIB Module Definition
RFC1612 : DNS Resolver MIB Extensions	RFC5601 : Pseudowire (PW) MIB
RFC1696 : Modem MIB	RFC5602 : Pseudowire (PW) over MPLS PSN MIB
RFC1697 : Relational DB Mngmnt System MIB	RFC5603 : Ethernet Pseudowire (PW) MIB
RFC1724 : RIP Version 2 MIB	RFC5728 : The SatLabs Group DVB-RCS MIB
RFC1748 : IEEE 802.5 MIB	RFC5813 : ForCES MIB
RFC2020 : IEEE 802.12 Interface MIB	RFC5833 : CAPWAP Protocol Base MIB
RFC2320 : Classical IP and ARP Over ATM MIB	RFC5834 : CAPWAP Proto. Binding MIB for IEEE 802.11
RFC2564 : Application Management MIB	RFC6240 : SONET/SDH Circuit Emul. over Pck. MIB
RFC1792 : TCP/IPX Connection MIB	RFC6639 : MPLS-TP MIB-Based Management Overview
RFC2605 : Directory Server Monitoring MIB	RFC6643 : Translation of SMIv2 MIB to YANG Modules
RFC2707 : Job Monitoring MIB	RFC6727 : Definitions of M. O. for Packet Sampling
RFC2720 : Traffic Flow Measurement : Meter MIB	RFC6765 : xDSL Multi-Pair Bonding (G.Bond) MIB
RFC2788 : Network Services Monitoring MIB	RFC6766 : xDSL Multi-Pair Bonding TDIM MIB
RFC2789 : Mail Monitoring MIB	RFC6767 : Ethernet-Based xDSL Multi-Pair Bonding MIB
RFC2790 : Host Resources MIB	RFC6768 : ATM-Based xDSL Bonded Interfaces MIB
RFC2863 : The Interfaces Group MIB	RFC6779 : Definition of Managed Objects for the NDP
RFC2922 : Physical Topology MIB	RFC6825 : TE Database MIB for MPLS-TE/GMPLS
RFC2932 : IPv4 Multicast Routing MIB	RFC6933 : Entity MIB (Version 4)
RFC2933 : IGMP MIB	RFC7052 : Locator/ID Separation Protocol (LISP) MIB
RFC2934 : PIM MIB for IPv4	RFC7124 : Ethernet in the 1st Mile Copper (EFMCu) MIB
RFC2981 : Event MIB	RFC7147 : Definitions of Managed Objects for iSCSI
RFC2982 : Distributed Management Expression MIB	RFC7184 : Def. of Managed Objects for the OLSRv2
RFC3014 : Notification Log MIB	RFC7330 : Def. of TCs for BFD
RFC3144 : RMon MIB Extensions for Interface	RFC7331 : BFD MIB
RFC3287 : RMon MIB Extensions for DiffServ...	...

## MIB constructeur



## SNMP : versions

Plusieurs versions ont été standardisées :

- **SNMPv1** définie dans le RFC 1157 (1990) simple et non sécurisée ➡ encore très utilisée
- **SNMPv2** définie dans les RFC 1901 à 1908 avec extensions (requêtes get-bulk et inform, MIB SNMPv2 et SNMPv2-M2M) et sécurisation mais pas de consensus des industriels
  - **SNMPv2c** réduite aux nouvelles fonctionnalités mais sans la sécurité (*Community-Based*)
  - **SNMPv2u** nouveau mécanisme de sécurité simplifié (*User-Based*)
- **SNMPv3** définie dans les RFC 3410 à 3418, réintègre la sécurité
  - seule la v3 est un standard IETF (STD-62)
  - Utilisation de multi-version : RFC 3584

## SNMP : limitations

- la mesure ne doit pas perturber le réseau
- latence
- MIB propriétaires
- sécurité
  - écoute sur le réseau (*packet sniffing*) pour connaître la communauté
  - usurpation d'identité (*IP spoofing*) facilitée par UDP

➡ améliorations avec **SNMPv3**