

Architecture des Réseaux (ARES)

2/5 : **Application**

Olivier Fourmaux (olivier.fourmaux@upmc.fr)

Version 6.2

ARES : plan du cours 2/5

1 Applications historiques

- Introduction
- Connexion à distance
- Transfert de fichiers

2 Applications principales

- World Wide Web
- Messagerie électronique
- Peer-to-peer

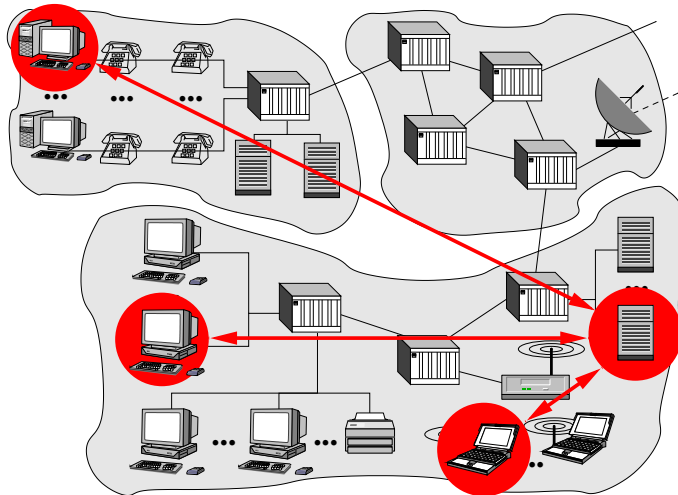
3 Applications support

- Annuaire (DNS)
- Administration de réseau

ARES : plan du cours 2/5

- 1 Applications historiques
 - Introduction
 - Connexion à distance
 - Tranfert de fichiers
- 2 Applications principales
 - World Wide Web
 - Messagerie électronique
 - Peer-to-peer
- 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 :

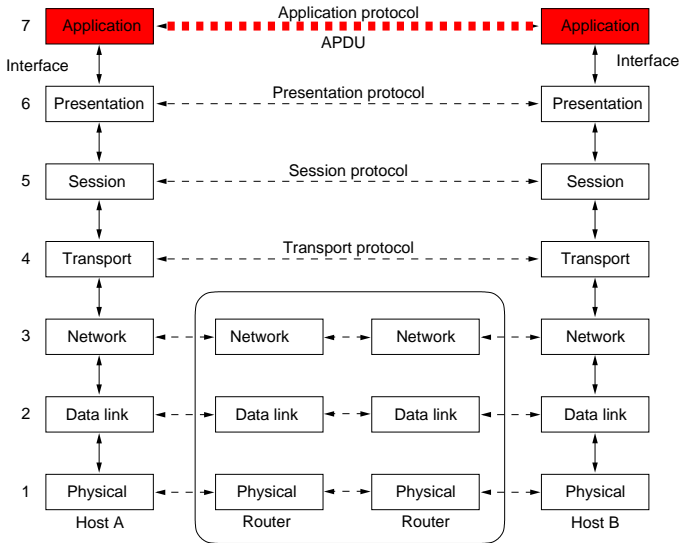
- standardise les échanges entre les applications les plus courantes
 - accès au web (HTTP), envoi d'*e-mail* (SMTP, POP, IMAP) ...



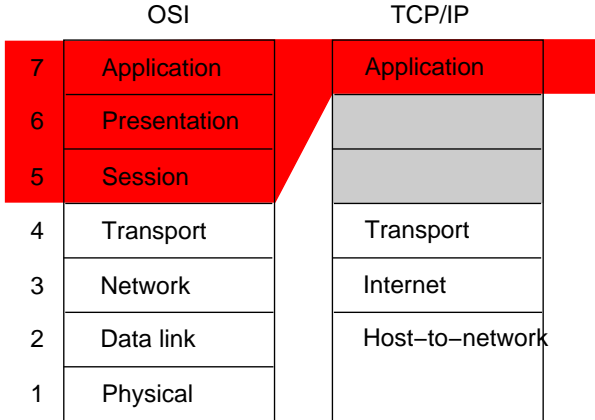
applications \neq protocoles de la couche application

- 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

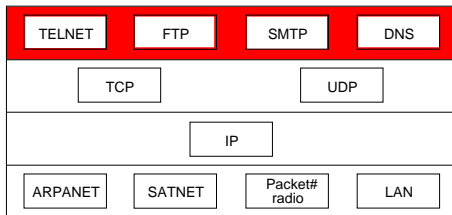
Couche Application : modèle OSI



Couche Application : modèle TCP/IP (1)



Couche Application : modèle TCP/IP (2)



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

TELNET pour contrôler une machine à distance

FTP pour transférer des données

SMTP pour échanger du courrier électronique

HTTP pour surfer sur la toile

DNS pour convertir les noms de l'Internet

SNMP pour administrer le réseau...

ARES : plan du cours 2/5

- 1 Applications historiques
 - Introduction
 - Connexion à distance
 - Tranfert de fichiers
- 2 Applications principales
 - World Wide Web
 - Messagerie électronique
 - Peer-to-peer
- 3 Applications support
 - Annuaire (DNS)
 - Administration de réseau

Applications de connexion à distance

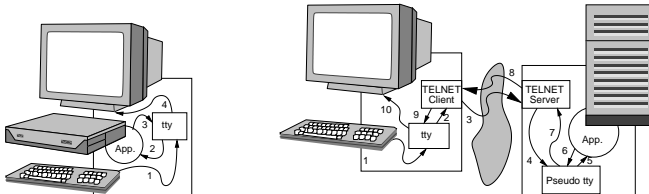
A partir d'un terminal ouvert sur une machine locale, connexion sur une machine distante

- 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 (*TE*Lecommunication *NE*Twork 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**...)



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

terminal local réel \Leftrightarrow terminal réseau virtuel

- 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**

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

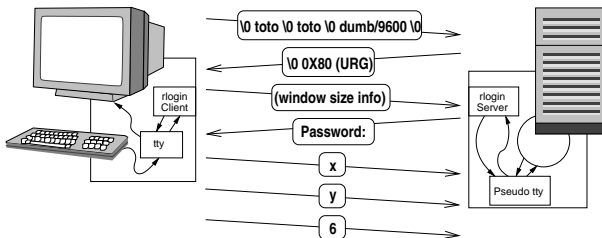
      <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
      <HTML>
      ...
      </HTML>
      Connection closed by foreign host.
```

➡ connexion TCP brute (limitation 7 bits)

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**, assure :
 - 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
- Peer-to-peer

3 Applications support

- Annuaire (DNS)
- Administration de réseau

Application de transfert de fichiers

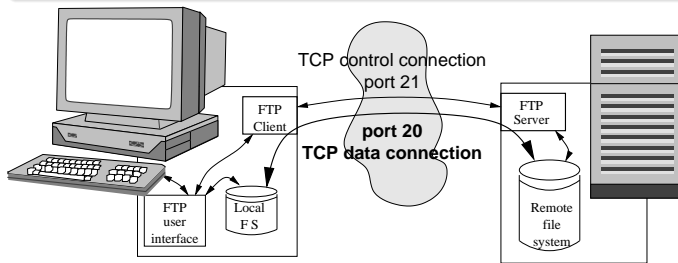
Copie d'un fichier d'un système vers un autre en environnement hétérogène

- 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 (*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**)



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**

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

Commandes utilisateur du programme ftp

```
Unix> ftp
```

```
ftp> help
```

Commands may be abbreviated. Commands are:

!	debug	mkdir	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	rhelph	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 : Réponses

Codage usuel : *status + description textuelle*

<i>status</i>	<i>description</i>	<i>status</i>	<i>description</i>
		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 AB]Ga!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.
```

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 (*Trivial File Transfer Protocol*)

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

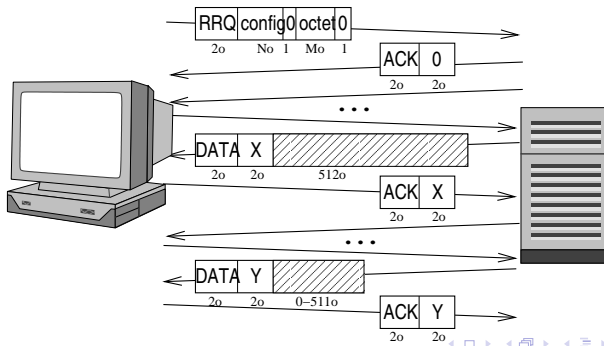
- 5 messages :

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)

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*)

ARES : plan du cours 2/5

- 1 Applications historiques
 - Introduction
 - Connexion à distance
 - Transfert de fichiers
- 2 Applications principales
 - World Wide Web
 - Messagerie électronique
 - Peer-to-peer
- 3 Applications support
 - Annuaire (DNS)
 - Administration de réseau

World Wide Web

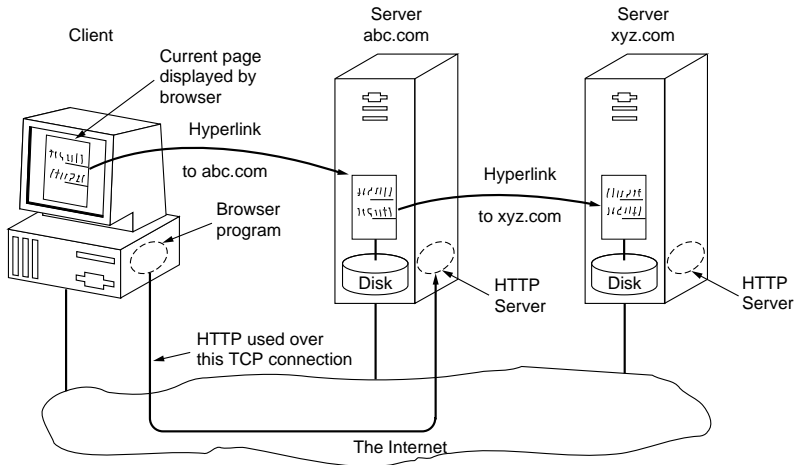
→ 90' : Internet = réseau académique

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-Champagne)
 - 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 objet 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

Format général d'un message :

- Request line -

Method	sp	URL	sp	Version	cr	If
Header field name	:	Value	cr	If		
Header field name	:	Value	cr	If		

...
Header lines

Header field name	:	Value	cr	If
cr	If			
Entity body				

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
```

• Method

- GET
- POST (formulaires)
- HEAD (test de pages)

• Connection

- Close

HTTP : Format réponse

Format général d'un message :

- Status line -

Version	sp	Status code	sp	Phrase	cr	If
Header field name	:	Value	cr	If		
Header field name	:	Value	cr	If		

...
Header lines

Header field name	:	Value	cr	If
cr	If			
Entity body				

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>
```

• status + description :

- 200 OK
- 301 Move permanently
- 400 Bad Request
- 404 Not Found
- 505 HTTP Version Not Supported ...

HTTP : Identifier les utilisateurs (1)

Authentification (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^{re} requête HTTP :

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

1^{re} 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.....
.....
.....
.....
.....

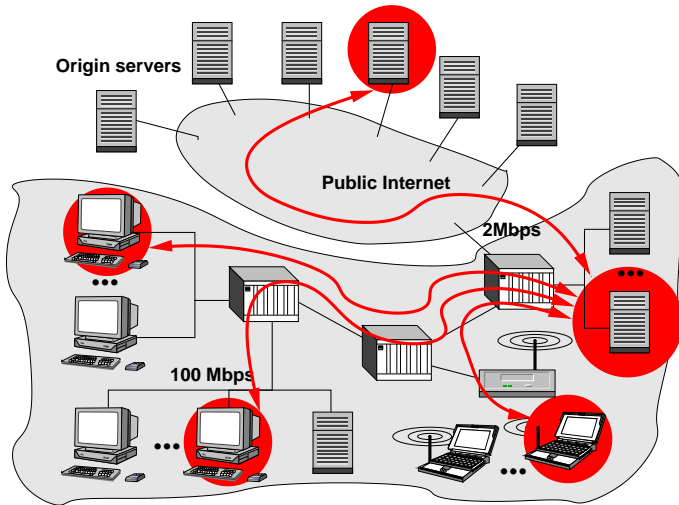
2^{me} requête HTTP :

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

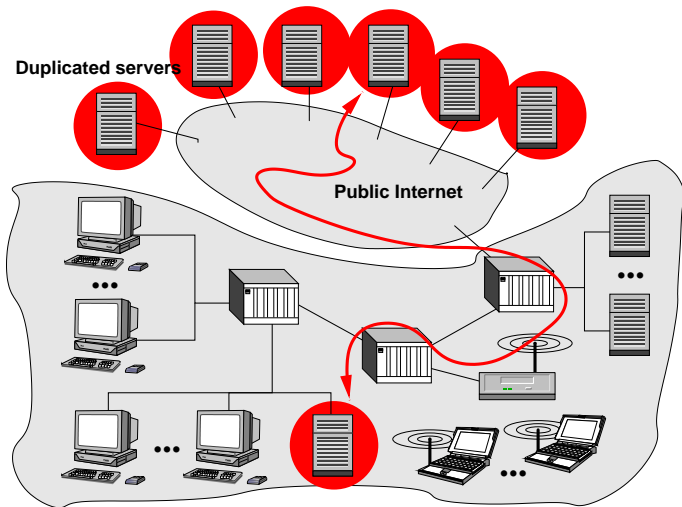
2^{me} réponse HTTP :

```
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...

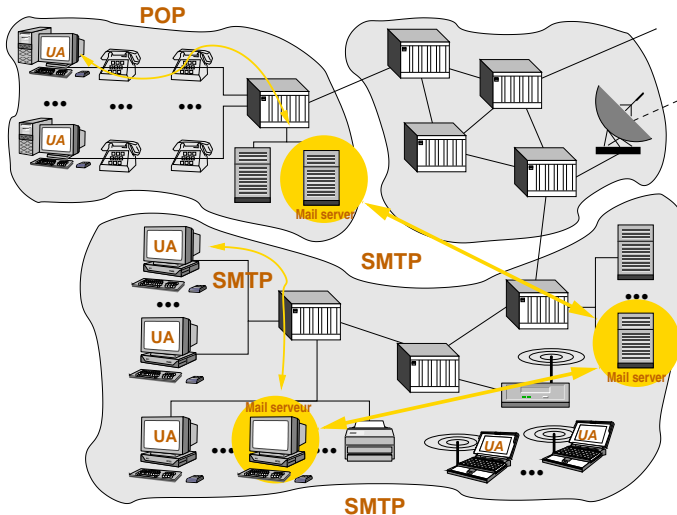
Protocole de transport générique

- 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
 - Peer-to-peer
- 3 Applications support
 - Annuaire (DNS)
 - Administration de réseau

Application de messagerie électronique



SMTP : introduction

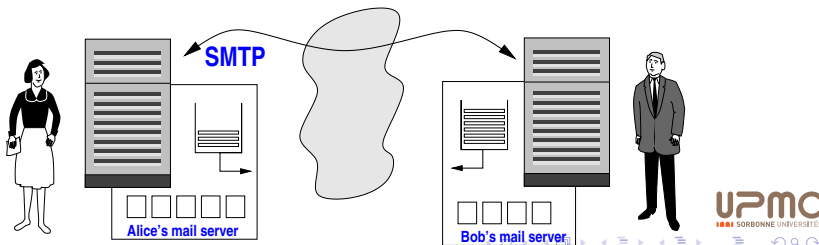
Echange de messages asynchrones à travers l'Internet

- 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**

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 +0
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 (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 20
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 : 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
VERFY	Vérification de l'adresse du destinataire
NOOP	Pas d'opération, force le serveur à répondre
RSET	Annule la transaction

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) :

EHLO	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 :

EHLO 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
```

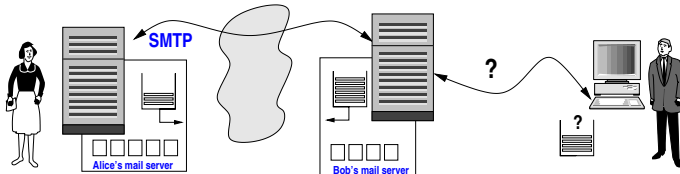
```
JVBERi0xLjIKJcfsj6IKNSAwIG9iago8PC9MZW5ndGggNiAwIFlvdG9yIC9GbG90ZURl  
Y29kZT4+CnN0cmVhbQp4n01dy7YdtRGd3684Mx6L07T63ZkBgdhgXvYlJF1MHNsYm+sHhkCS...
```

Remise finale des messages

Machine accédant sporadiquement au réseau ?

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

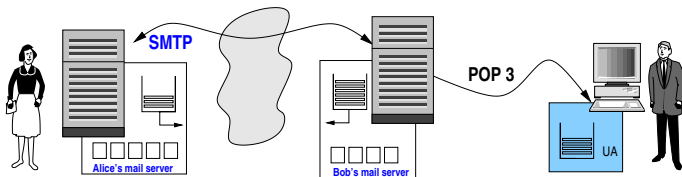
- plusieurs alternatives combinables :
 - accès direct au serveur (montage NFS ou SMB)
 - POP
 - IMAP
 - HTTP



POP3

Post Office Protocol – Version 3 (RFC 1939)

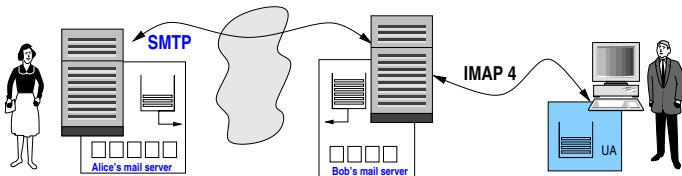
- simple
- connexion TCP sur le port 110
- trois phases :
 - autorisation (identification)
 - transaction (récupération et marquage)
 - mise à jour (suppression effective du serveur)



IMAP4

Internet Mail Access Protocol – version 4 (RFC 2060)

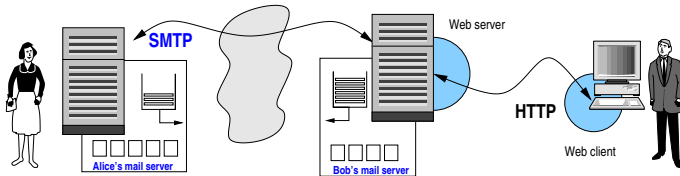
- complexe
- connexion TCP sur le port 143
- même fonctionnalité que POP avec :
 - accès par attribut (12^{ème} e-mail d'Alice)
 - récupération de partie de message (3^{ème} pièce jointe)
 - **synchronisation** de boîtes aux lettres



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

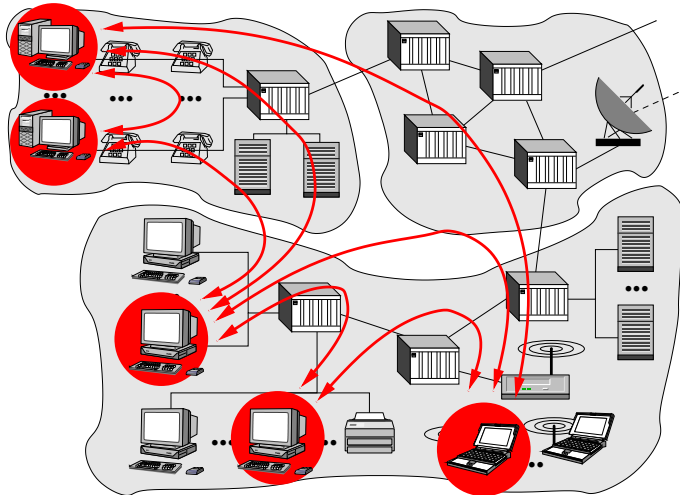
2 Applications principales

- World Wide Web
- Messagerie électronique
- Peer-to-peer

3 Applications support

- Annuaire (DNS)
- Administration de réseau

Application de partage de fichier *Peer-to-peer*



De nombreuses applications *Peer-to-peer*

Applications *peer-to-peer* :

- **partage de fichiers**

- Napster, eDonkey, BitTorrent...
- FastTrack (KaZaA, Grokster et Imesh), Gnutella2...
- Gnutella...
- BitTorrent

- **stockage anonyme**

- Freenet, Entropy...

- **distribution de flux audio/vidéo**

- VoD : Peercast, Joost...
- P2PTV : Coolstreaming, TVants, PPLive...

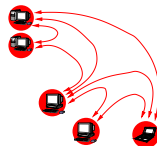
- autres services réseau "remontés" au niveau applicatif

- protocoles de routage IP
- réseaux ad-hoc
- communications multipoints...

P2P : questions

Principes fondamentaux

- éléments de base générique (ni client, ni serveur)
- agrégation des ressources réseaux/processeur/stockage
- protocoles de niveau applicatif



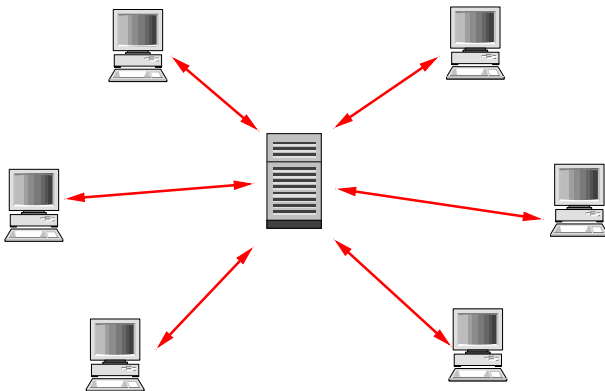
Standards ?

Ne peut-on pas tout faire en client/serveur ?

- est-ce juste du "réseau au niveau applicatif" ?
- quels nouveaux types de services ? d'applications ?
- quels sont les nouveaux challenges techniques ?

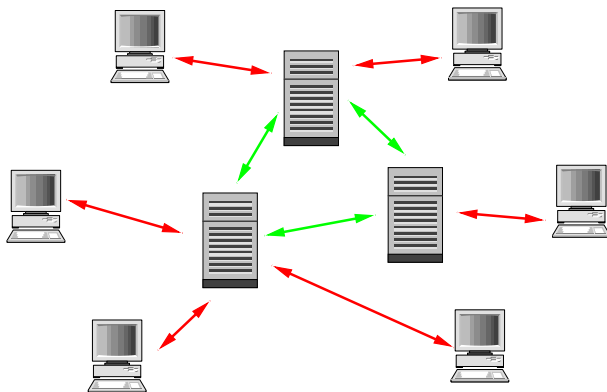
P2P : architectures (1)

Client/serveur centralisé classique



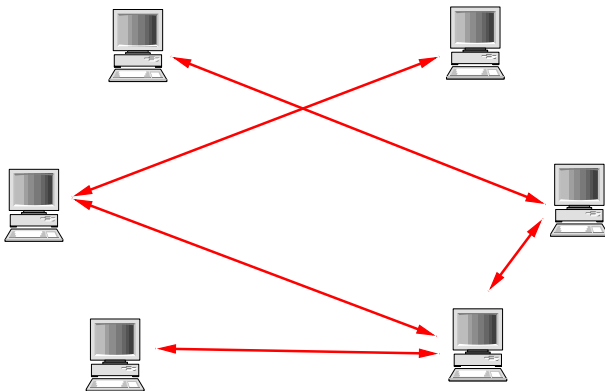
P2P : architectures (2)

Client/serveur avec réplication des serveurs



P2P : architectures (3)

Peer-to-peer classique



P2P : Comparaison Client/serveur

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

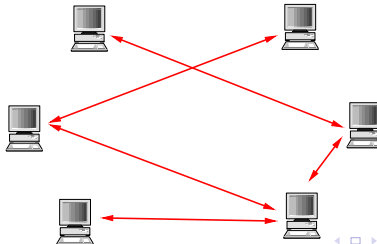
```
Peer_main_loop()
```

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

P2P : Fonctionnalités

Caractéristiques des systèmes *peer-to-peer*

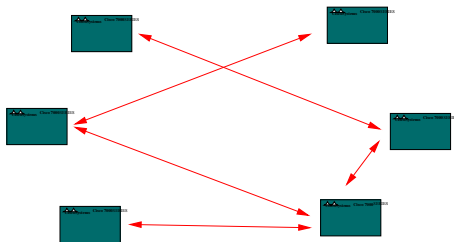
- pas de rôle distinct
 - évite les points de congestion ou les nœuds défaillants
 - besoin d'algorithmes distribués
 - découverte de service (nommage, adressage, calcul de métriques)
 - maintien d'un état du voisinage
 - routage au niveau applicatif (lié au contenu)
 - robustesse, gestion de perte de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



P2P : Applications existantes

Le *peer-to-peer* n'est pas nouveau :

- Routeurs IP
 - découverte de la topologie
 - maintien de l'état du voisinage
 - autonome et tolérance aux fautes
 - algorithme distribué de routage ...



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...

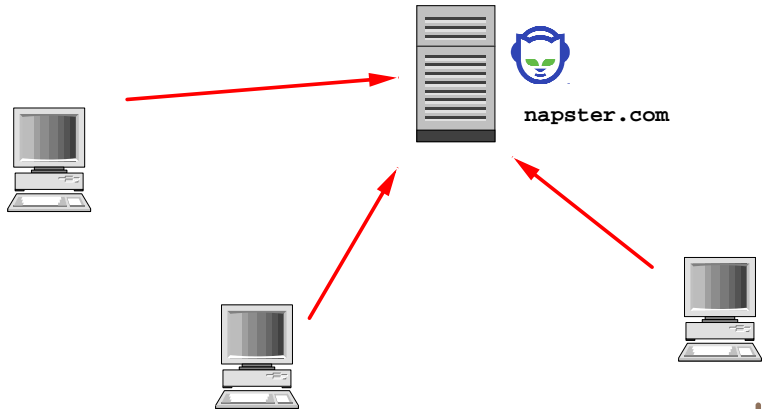
Napster : Principe

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
 - Selection du meilleur pair (*pings*)

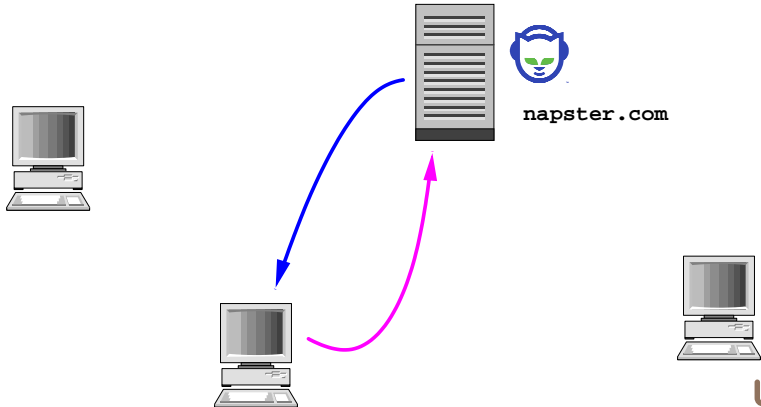
Napster : *Upload*

Les utilisateurs chargent la liste des fichiers à partager



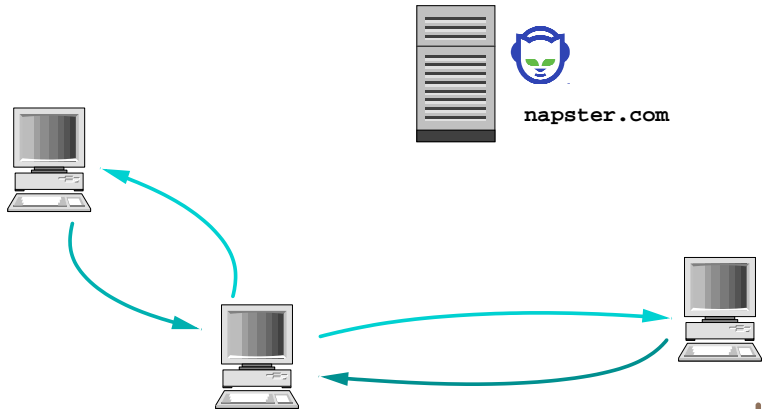
Napster : *Search*

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



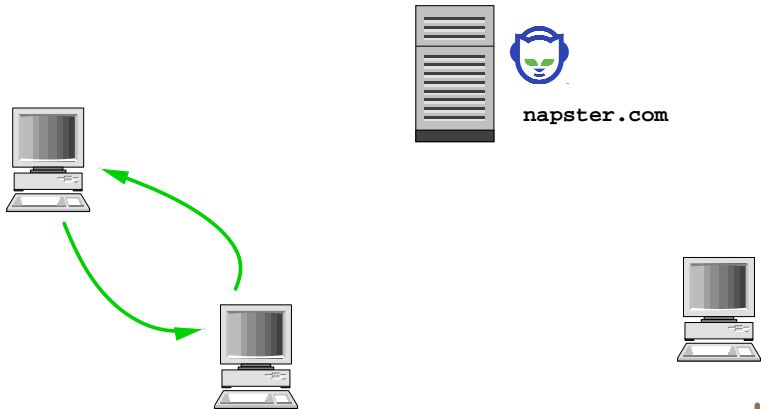
Napster : *Pings*

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



Napster : *Download*

L'utilisateur récupère directement le fichier chez le pair choisi



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
- évolution :
 - **OpenNap** :
 - *open source*
 - communications entre serveurs
 - tous types de données

Gnutella : motivations (1)



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 !

Gnutella : motivations (2)

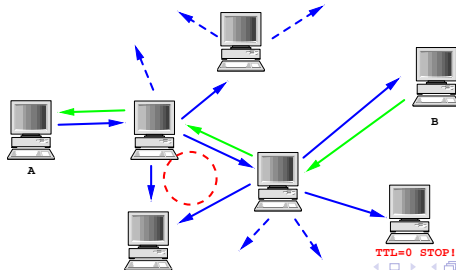
Peer-to-peer Networking

- connexion exclusive entre les applications des pairs
- problème :
 - recherche de fichiers **décentralisée**
- chaque application :
 - stocke une sélection de fichiers
 - oriente les requêtes de recherche (*route query*) de et vers ses pairs
 - répond aux demandes de transfert de fichiers
- historique
 - 03/00 abandon du projet *freelance* au bout de qqs jours après son lancement par AOL (Nullsoft)
 - trop tard : déjà plus de 20K utilisateurs...

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 : messages

Structure du message de contrôle Gnutella :

Gnode ID (16 octets)	Type (1 octet)	TTL (1 octet)	Sauts (1 octet)	Longueur (4 octets)	Données...
-------------------------	-------------------	------------------	--------------------	------------------------	------------

Gnode ID : identification du nœud dans le réseau gnutella

Type : action à réaliser

- Ping (recherche de pair)
- Pong (réponse à un Ping, adresse IP)
- Query (recherche de fichier selon des critères)
- Query-Hit (réponse avec liste des fichiers et IP)
- Push (mécanisme de passage de *firewall*)

TTL : nombre de sauts encore réalisables

Sauts : nombre de sauts réalisés

$$\bullet \text{ } TTL_n + Sauts_n = TTL_{initial}$$

Longueur : taille du bloc **données** en octets

Données : peuvent être vides

Gnutella : identification des pairs (1)

Détection active des pairs

- requête Ping
 - pas de données
 - limitations des envois pour ne pas saturer le réseau
 - crée un état dans la table de routage (retour des Pong)
 - répondre et relayer aux pairs connectés (limite du TTL)

- réponse Pong

- données :

Port (2 octets)	Adresse IP (4 octets)	Nb de fichiers (4 octets)	Nb de Ko partagés (4 octets)
--------------------	--------------------------	------------------------------	---------------------------------

Port : port sur lequel le pair est en attente

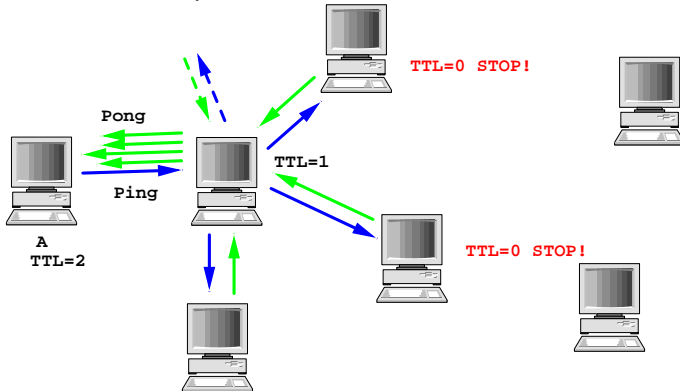
Adresse IP : adresse à laquelle le pair est atteignable

Nb fichiers : nombre de fichiers partagés par le pair

Ko partagés : quantité de données partagées par le pair

Gnutella : identification des pairs (2)

Détection active des pairs



Gnutella : recherche de fichier (1)

Requête pour trouver une information

- requête Query :

Vitesse minimum (2 octets)	Critères de recherche (N octets)
-------------------------------	-------------------------------------

Vitesse : débit minimum pour qu'un pair réponde (Ko/s)

Critères : chaîne de caractères terminée par 0x00

- crée un état dans la table de routage (retour des Query-Hit)
 - relayer aux pairs connectés (limite du TTL)
- réponse Query-Hit...

Gnutella : recherche de fichier (2)

- réponse Query-Hit

Nb Hit (1 octet)	Port (2 octets)	Adresse IP (4 octets)	Vitesse (4 octets)	Résultats (N octets)	GID Pair (16 octets)
---------------------	--------------------	--------------------------	-----------------------	-------------------------	-------------------------

Nb Hit : indique le nombre de champs des **Résultats**

Port : port sur lequel le pair est en attente

Adresse IP : adresse à laquelle le pair est atteignable

Vitesse : débit minimum demandé (Ko/s)

Résultats : ensemble de **Nb Hit** champs :

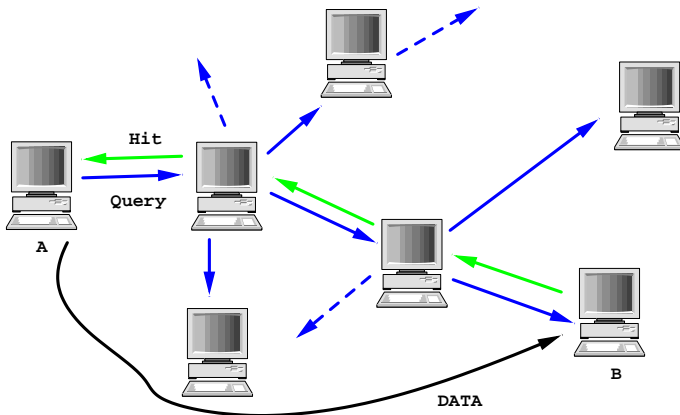
Index du fichier (4 octets)	Taille du fichier (4 octets)	Nom du fichier (chaîne terminant par 0x0000)
--------------------------------	---------------------------------	---

GID Pair : identification pour un Push

- sont routées selon le chemin inverse suivi par requêtes Query

Gnutella : recherche de fichier (3)

Requête pour trouver une information



Gnutella : *Firewall* (1)

Retourner la connexion des données

- requête Push

- données :

GID Pair (16 octets)	Index du fichier (4 octets)	Adresse IP (4 octets)	Port (2 octets)
-------------------------	--------------------------------	--------------------------	--------------------

GID Pair : identification du pair

Index : identifiant unique du fichier sur le pair

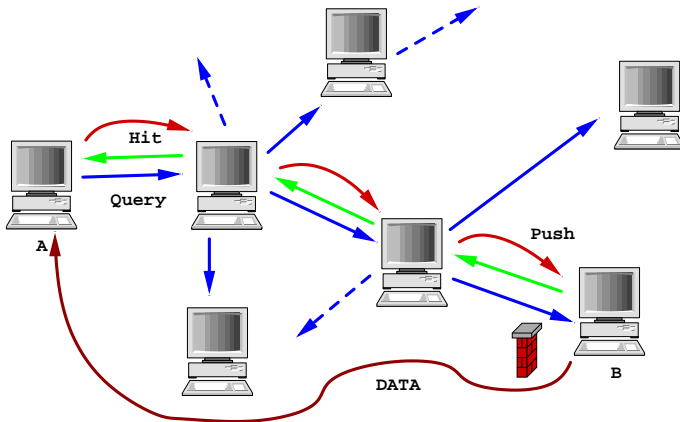
Adresse IP : adresse à laquelle le fichier doit être
envoyé

Port : port sur lequel le destinataire est en
attente

- sont routées selon le chemin inverse suivi par réponses
Query-Hit
- permet la création de la connexion donnée à partir du pair

Gnutella : *Firewall* (2)

Retourner la connexion des données



Gnutella : Gestion des connexions

Connexion de contrôle sur TCP

- demande de connexion :

GNUTELLA CONNECT/0.6

Node: 201.33.182.178:6346

User-Agent: gtk-gnutella/0.80 beta2 - 22/01/2002

- réponse du pair :

GNUTELLA/0.6 200 OK

User-Agent: Morpheus 2.0.1.7

Remote-IP: 181.185.36.178

- confirmation :

GNUTELLA/0.6 200 OK

Récupération des données par **HTTP**

- séparée du réseau Gnutella :

- connexion directe entre pairs et envoi d'un GET

Gnutella : Remarques

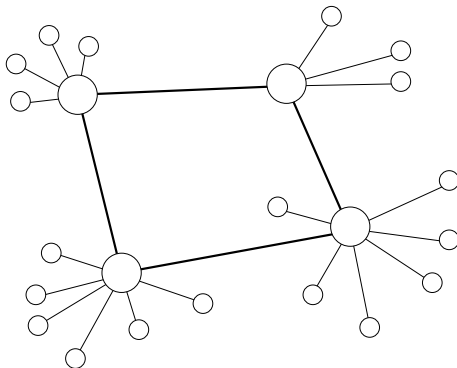
Leçons retenues :

- 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
 - \exists protocoles permettant de ne pas connaître le destinataire

Evolutions P2P

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

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



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*

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) :
 - <http://thepiratebay.org/>
 - <http://www.mininova.com/>
 - <http://www.demonoid.com/> (abonnement)
 - ...
 - *tracker* distribué (table de hachage distribuée)
 - basée sur Kademlia
- *multitracker*
 - redondance
 - surcoute en signalisation
- cryptage des échanges
 - *Protocol header encrypt (PHE)*
 - *Message stream encryption/Protocol encryption (MSE/PE)*
- distribution de contenus (*streaming A/V*)
 - nombreux projets...

P2P : autres

Partage de fichier **complètement anonyme**

- **Freenet**

- *peer-to-peer* décentralisé (comme Gnutella)
 - connaissance locale seulement
 - accès aux ressources de proche en proche (routage)
 - producteur anonyme
 - consommateur anonyme
 - résistance aux tentatives de limitation d'accès

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

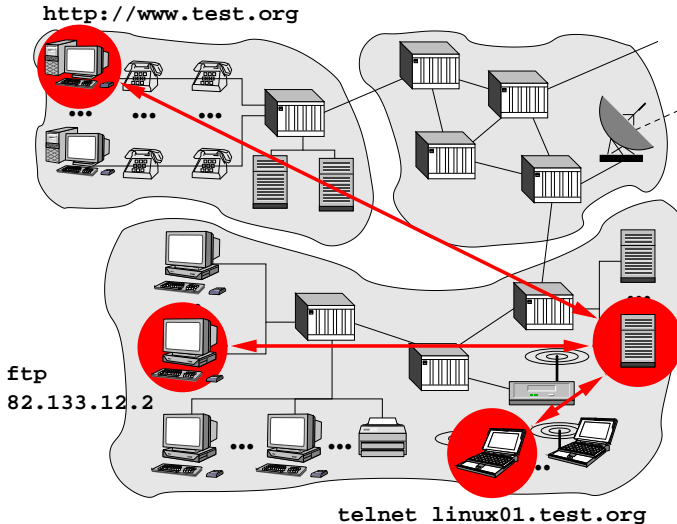
- **CAN** (*Content Addressable Network*)

- système de coordonnées cartésiennes virtuelles ...

ARES : plan du cours 2/5

- 1 Applications historiques
 - Introduction
 - Connexion à distance
 - Transfert de fichiers
- 2 Applications principales
 - World Wide Web
 - Messagerie électronique
 - Peer-to-peer
- 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*)

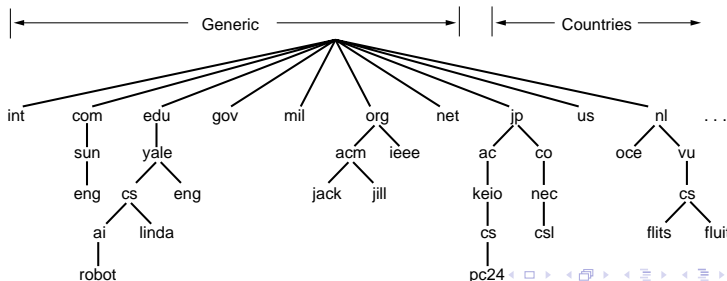
Annuaire standard de l'Internet (RFC 1034 et RFC 1035)

- 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)



DNS : gTLD (*generic Top Level Domain*)

gTLD	intro.	description	operator
.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.

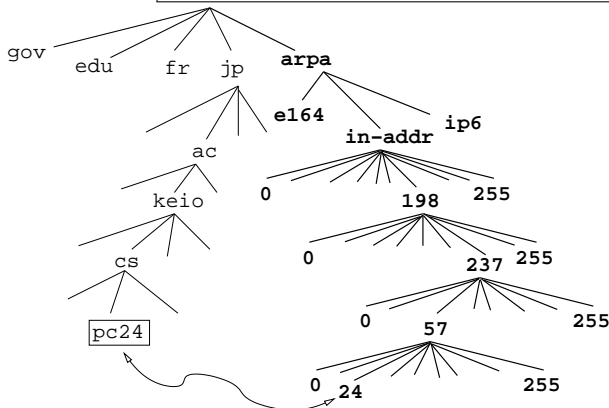
DNS : ccTLD (*country code Top Level Domain*)

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
.za	South Africa
.zw	Zimbabwe

DNS : Domaine .arpa

Résolution : pc24.cs.keio.ac.jp. ➡ ?

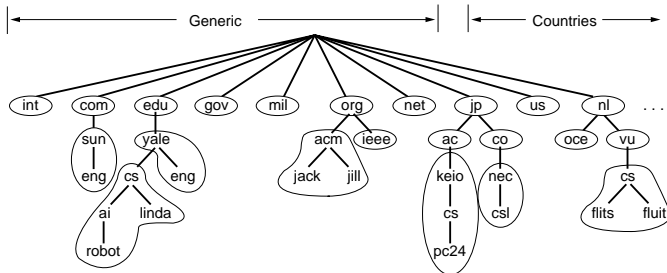
Résolution **inverse** : 24.57.237.198.in-addr.arpa. ➡ ?



DNS : zones (1)

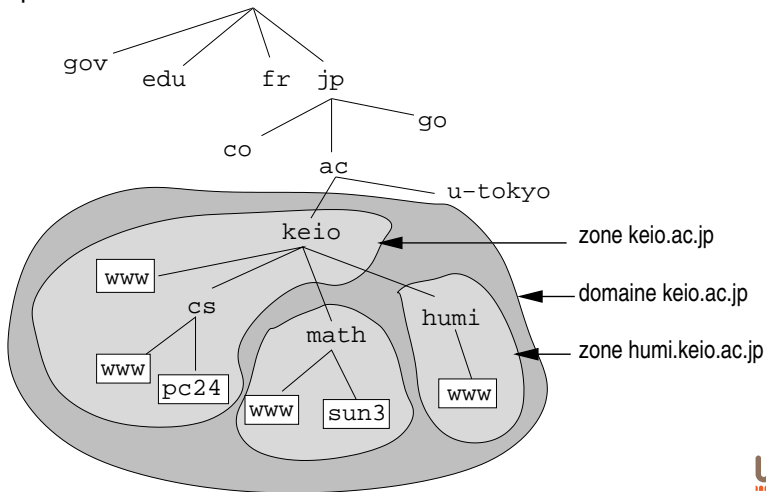
ICANN gère la racine et délègue les TLDs à des *domain name registry*

- 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 !

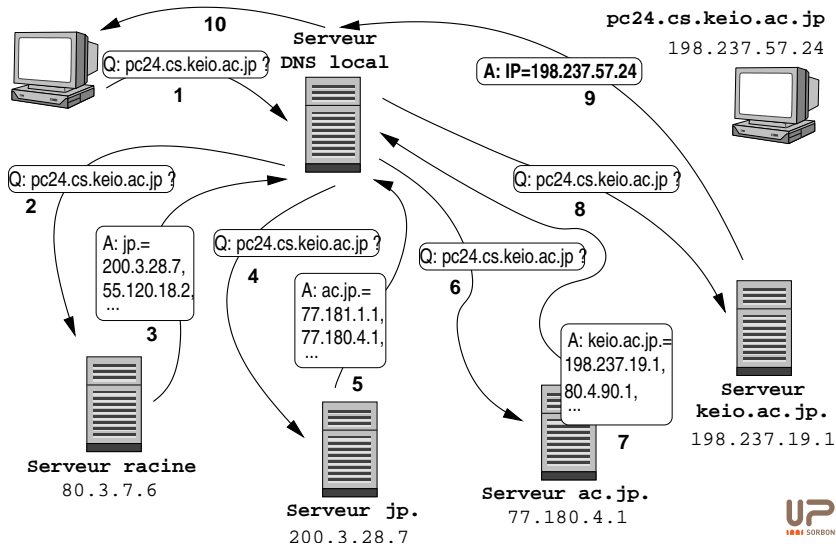


DNS : serveurs de noms

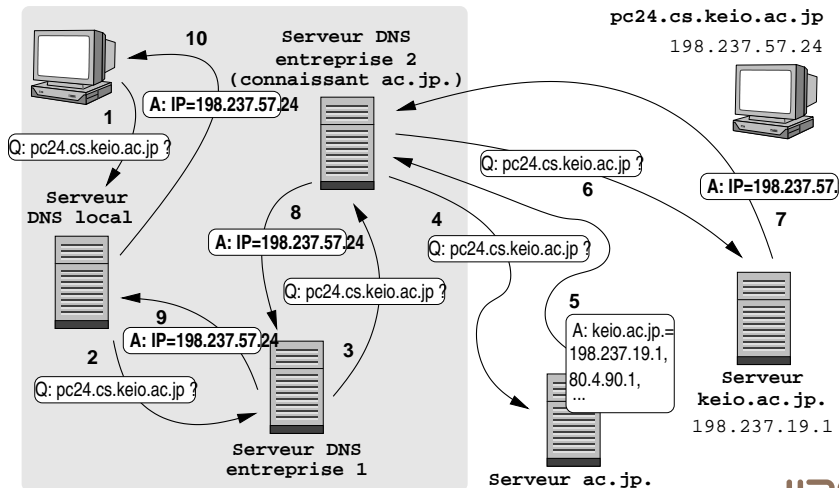
Différents types de serveurs de noms

- **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 **récurives** ou **itératives**

DNS : requête itérative



DNS : requête récursive



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)

DNS : format général du message

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

flags :

- **QR** (1 bit) : 0 = question, 1 = réponse
- **opcode** (4 bit) 0 = standard ...
- **AA** (1 bit) : 1 = réponse autoritaire
- **TC** (1 bit) : 1 = tronqué (datagramme UDP < 512o)
- **RD** (1 bit) : 1 = demande récursion (indiqué par le client)
- **RA** (1 bit) : 1 = récursion disponible (indiqué par le serveur)
- **réservé** (3 bits) : 000
- **rcode** (4 bits) : 0 = pas d'erreur... 3 = erreur de nom...

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

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

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...

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

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

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 : 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...

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
```

ARES : plan du cours 2/5

- 1 Applications historiques
 - Introduction
 - Connexion à distance
 - Transfert de fichiers
- 2 Applications principales
 - World Wide Web
 - Messagerie électronique
 - Peer-to-peer
- 3 Applications support
 - Annuaire (DNS)
 - Administration de réseau

administration de réseau

Développement du réseau (nombreux équipements et machines à gérer)

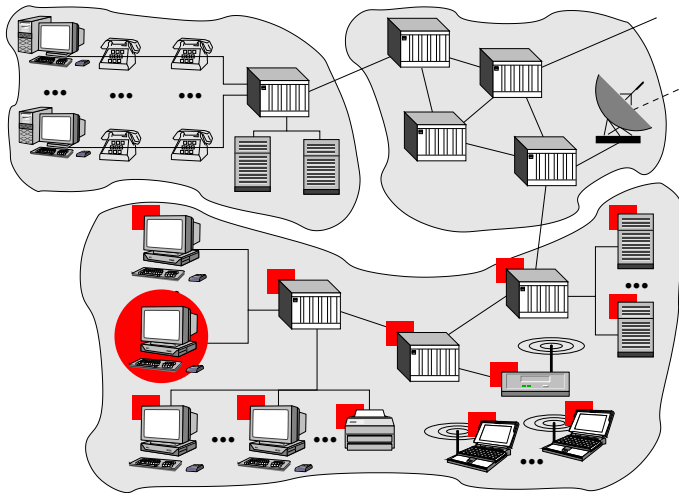
Besoins :

- 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, switchs...
 - ordinateurs, imprimantes, sondes...
- constructeurs multiples
- localisation géographique distante

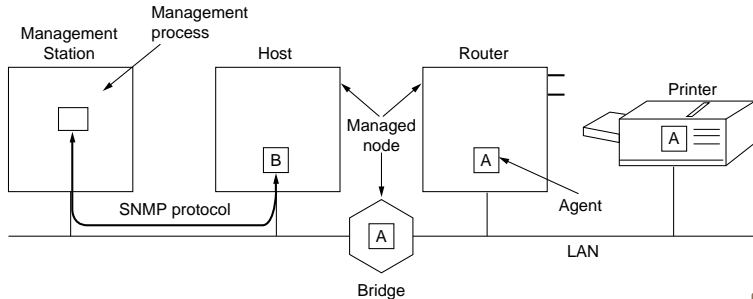
Equipements administrables



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



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

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)

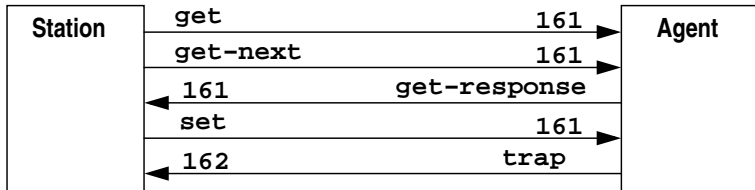
- désignation et type d'information définis par des **MIB**
- structures communes et nomenclature définies dans la **SMI**
- représentation des données en **ASN.1**
- protocole **SNMP** entre la station et les agents permettant :
 - **lecture/écriture** de variables sur des éléments gérés
 - **alarmes**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

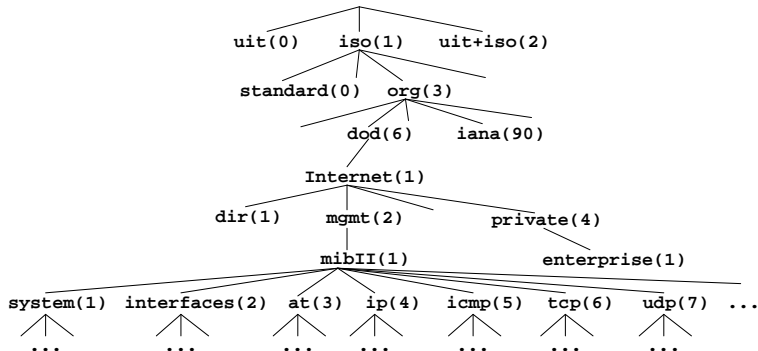
SNMP : SMI (*Structure for Management Information*)

- 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

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 ...

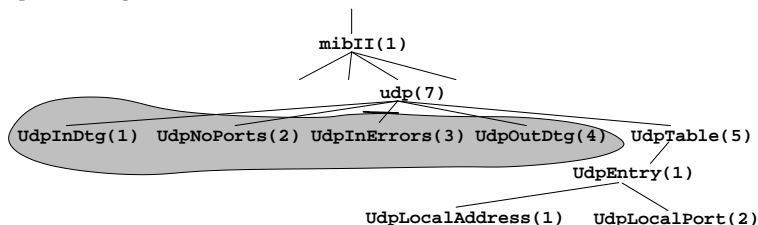
Ces groupes contiennent des variables **simples** ou **tables**

MIB : variable 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

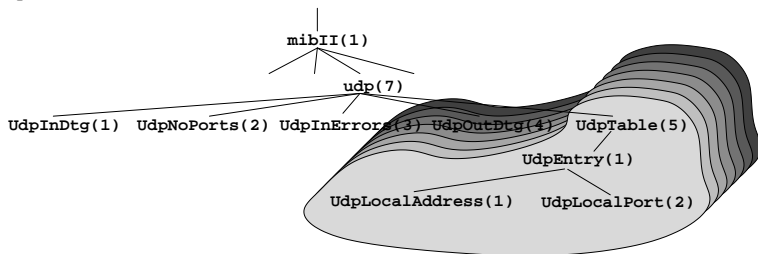


MIB : variable table

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

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

- 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
    }...
END
```

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

- UNIVERSAL (00)

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

- APPLICATION (01)

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

- CONTEXT (10)

- PRIVATE (11)

- longueur des données (1 octet si $< 0x80$, sinon norme X.208)
 - longueur 49 \Rightarrow 0x31, longueur 242 \Rightarrow 0x8200F2...
- données (valeur)
 - les OID (avec les valeurs entières successives A.B.C.D...) sont codés en octets avec les 2 premiers agrégés : $A*40+B$, C , D

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...
```

MIB RMON

Remote MONitoring (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) ...

Autres MIB IETF (1)

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



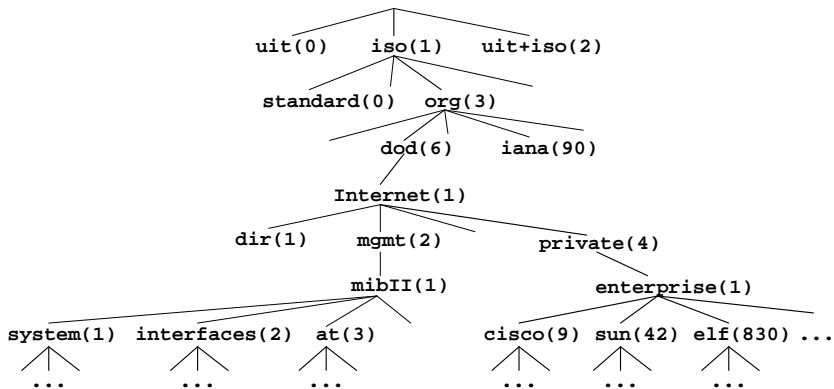
- 274 Objets (228 OID dont 16 tables)
 - 20 groupes :
 - groupe général
 - groupe des entrées
 - groupe des sorties
 - groupe des dimensions de sortie
 - groupe de la couverture
 - groupe des fournitures
 - groupe des colorants ...

Autres MIB IETF (2)

RFC1230 : IEEE 802.4 Token Bus MIB
RFC1381 : MIB Extension for X.25 LAPB
RFC1559 : DECnet Phase IV MIB Extensions
RFC1593 : SNA APPN Node MIB
RFC1611 : DNS Server MIB Extensions
RFC1612 : DNS Resolver MIB Extensions
RFC1696 : Modem MIB
RFC1697 : Relational DB Mngmnt System MIB
RFC1724 : RIP Version 2 MIB
RFC1748 : IEEE 802.5 MIB
RFC2020 : IEEE 802.12 Interface MIB
RFC2320 : Classical IP and ARP Over ATM MIB
RFC2564 : Application Management MIB
RFC1792 : TCP/IPX Connection MIB
RFC2605 : Directory Server Monitoring MIB
RFC2707 : Job Monitoring MIB
RFC2720 : Traffic Flow Measurement : Meter MIB
RFC2788 : Network Services Monitoring MIB
RFC2789 : Mail Monitoring MIB
RFC2790 : Host Resources MIB
RFC2863 : The Interfaces Group MIB
RFC2922 : Physical Topology MIB
RFC2932 : IPv4 Multicast Routing MIB
RFC2933 : IGMP MIB
RFC2934 : PIM MIB for IPv4
RFC2981 : Event MIB
RFC2982 : Distributed Management Expression MIB
RFC3014 : Notification Log MIB
RFC3144 : RMon MIB Extensions for Interface
RFC3287 : RMon MIB Extensions for DiffServ.....

... **RFC4672** : RADIUS Dynamic Authoriz. Client MIB
RFC4673 : RADIUS Dynamic Authoriz. Server MIB
RFC4711 : Real-time Application QoS Monit. MIB
RFC4747 : The Virtual Fabrics MIB
RFC4807 : IPsec Security Policy DB Conf. MIB
RFC4898 : TCP Extended Statistics MIB
RFC4935 : Fibre Channel Fabric Conf. Server MIB
RFC4936 : Fibre Channel Zone Server MIB
RFC4983 : Fibre Channel RSCN MIB
RFC5017 : MIB Textual Conventions for URIs
RFC5060 : Protocol Independent Multicast MIB
RFC5066 : EFMcu Interface MIB
RFC5097 : MIB for the UDP-Lite protocol
RFC5098 : Signaling MIB for PacketCable MTAs
RFC5131 : A MIB Textual Convention for Language Tags
RFC5132 : IP Multicast MIB
RFC5240 : PIM Bootstrap Router MIB
RFC5324 : MIB for Fibre-Channel Security Protocols
RFC5428 : Management Event MIB for PacketCable
RFC5519 : Multicast Group Membership Discovery MIB
RFC5525 : Reliable Server Pooling MIB Module Definition
RFC5601 : Pseudowire (PW) MIB
RFC5602 : Pseudowire (PW) over MPLS PSN MIB
RFC5603 : Ethernet Pseudowire (PW) MIB
RFC5728 : The SatLabs Group DVB-RCS MIB
RFC5813 : ForCES MIB
RFC5833 : CAPWAP Protocol Base MIB
RFC5834 : CAPWAP Protocol Binding MIB for IEEE 802.11
RFC6240 : SONET/SDH Circuit Emulation over Packet MIB
RFC6639 : MPLS-TP MIB-Based Management Overview

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*) facilité par UDP

➡ améliorations avec **SNMPv3**