

Couche application

Adlen Ksentini



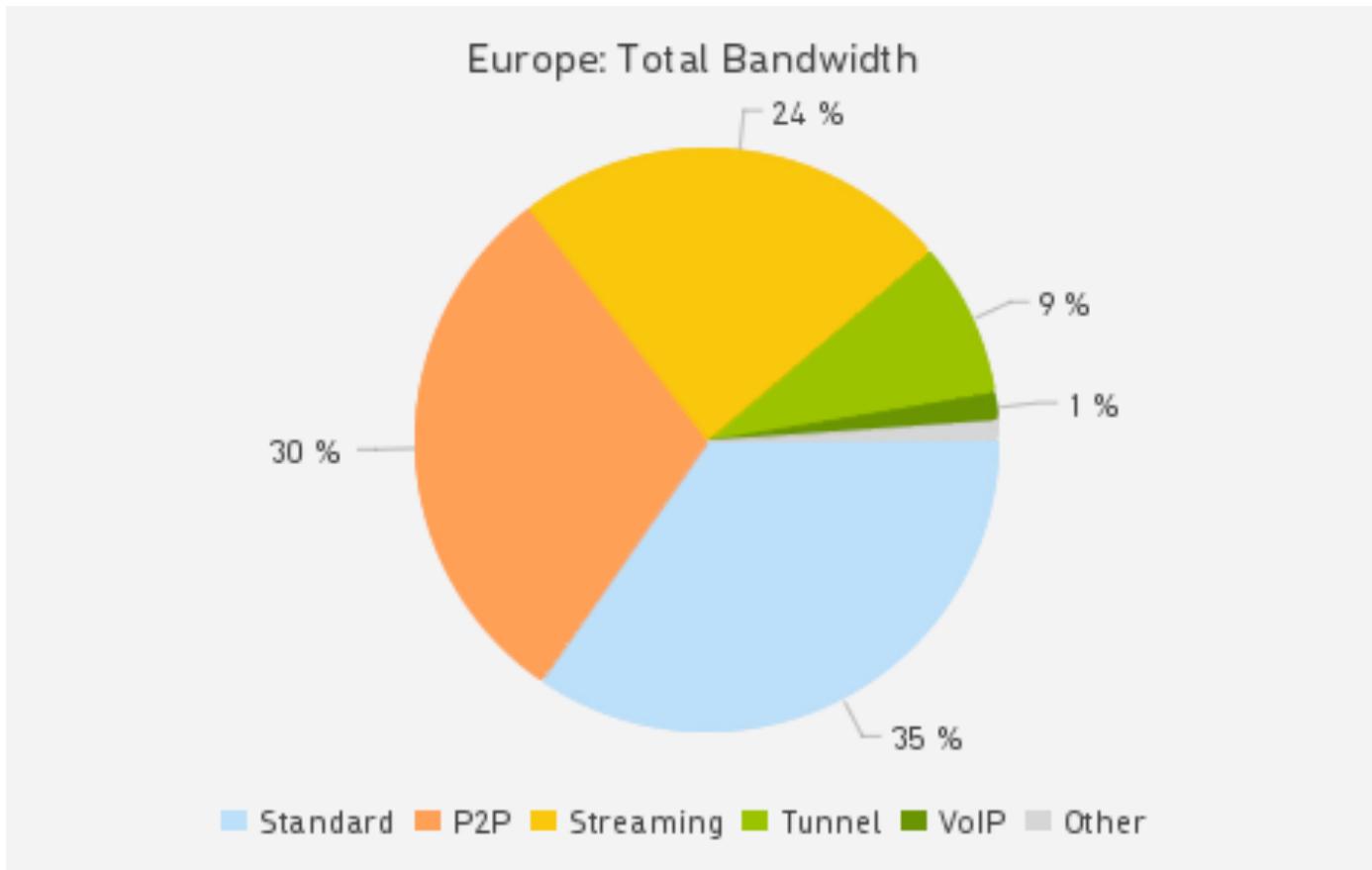
Introduction

- But :
- Connaitre les concepts et l'implémentation des protocoles applicatifs communiquant sur un réseau
 - Services offerts par la couche transport
 - Le concept Client/Serveur
 - Le concept Peer-to-Peer (p2p)
- Etude des protocoles applicatifs les plus populaires
 - HTTP
 - FTP
 - TELNET
 - SMTP / POP3 / IMAP
 - DNS
- Programmation des applications réseau
 - API Socket

Quelques applications réseau

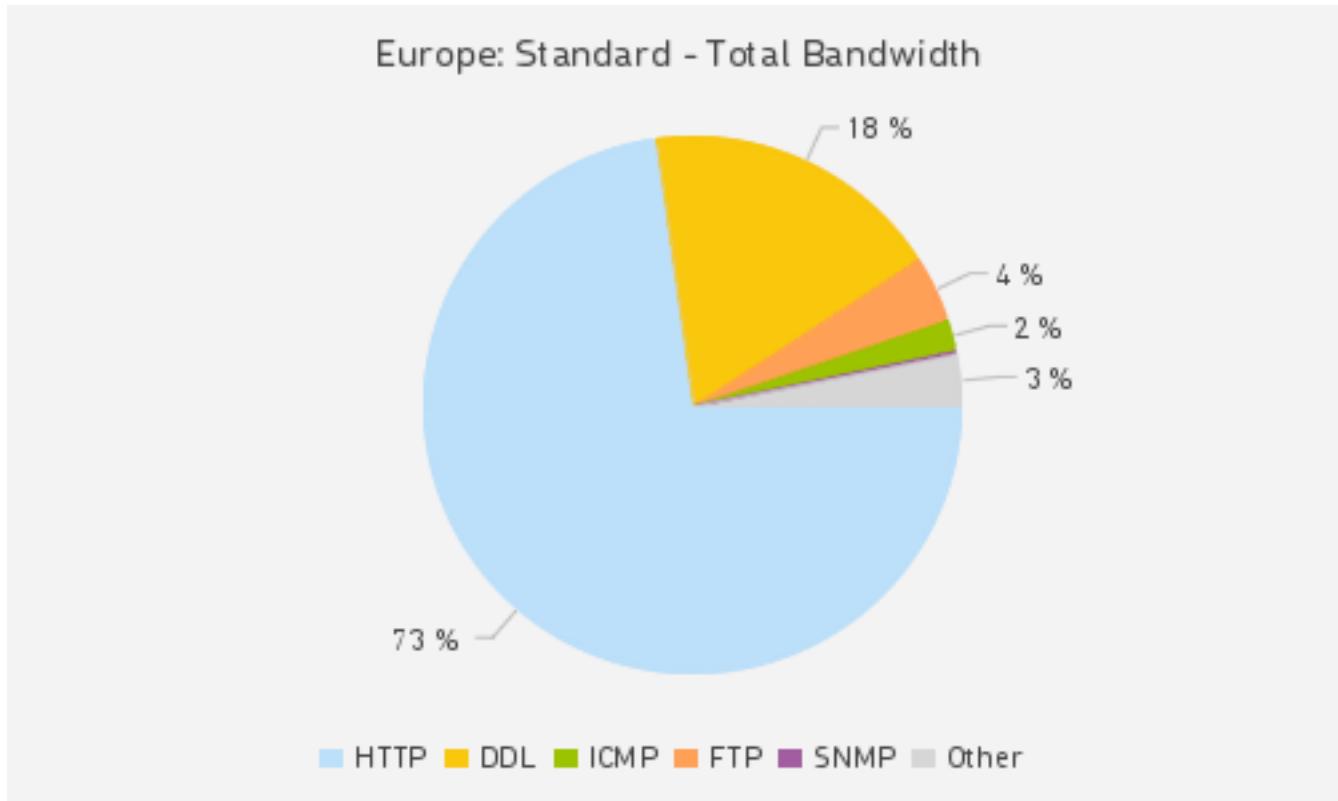
- E-mail
- Web
- Messagerie instantanée
- Contrôle à distance
- Partage de fichiers P2P
- Jeux en réseau (Second life, Warcraft)
- Streaming de vidéo (Youtube, Dailymotion)
- Téléphone sur Internet (Skype)
- Réseaux sociaux (twitter, facebook, googl+)

Distribution du trafic : Europe



Source : <http://www.internetobservatory.net/europe>

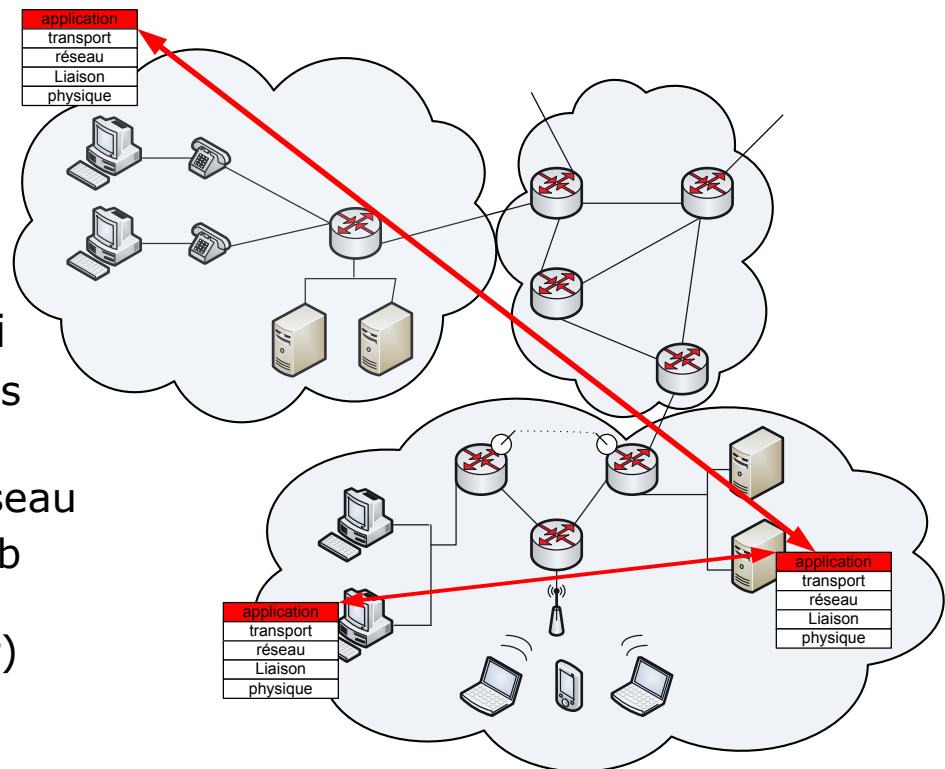
Distribution du trafic



Source : <http://www.internetobservatory.net/europe>

Création d'une application réseau

- Ecrire un programme qui
 - Fonctionne sur différents systèmes
 - Communique sur un réseau
 - Ex: Web, le serveur Web communique avec un programme (navigateur) client



Applications réseau : architectures

- Client-Serveur
- Peer-to-Peer
- Hybride entre Client/Serveur et P2P

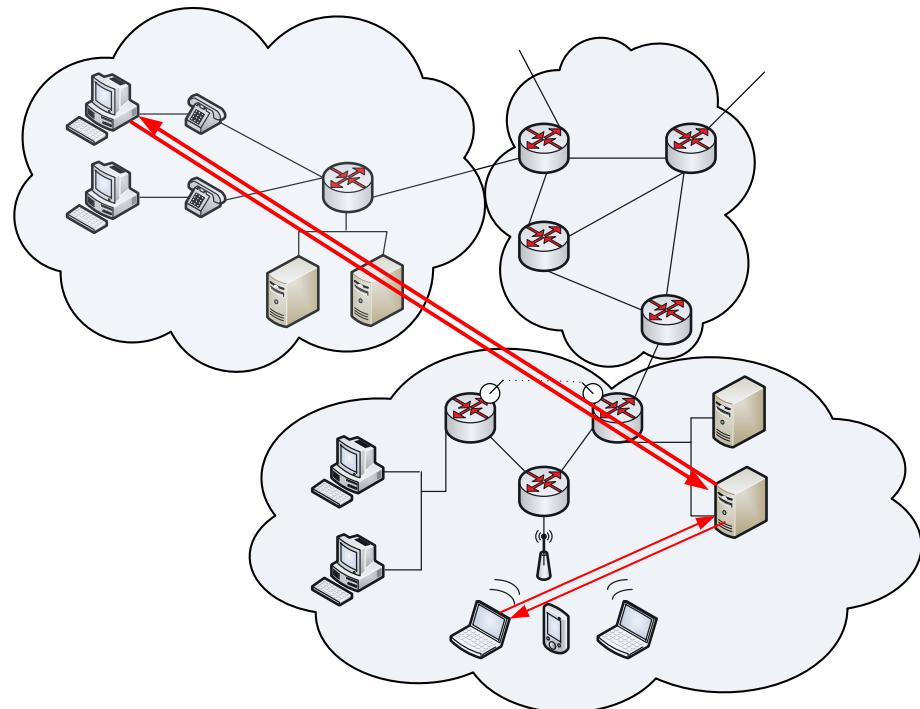
Architecture Client/Serveur

- Serveur

- En écoute sur un hôte
- Adresse IP fixe

- Client

- Communique avec un serveur
- Peut se connecter par intermittence
- Ne communique pas avec un autre client
- Peut avoir une adresse IP dynamique (non fixe)



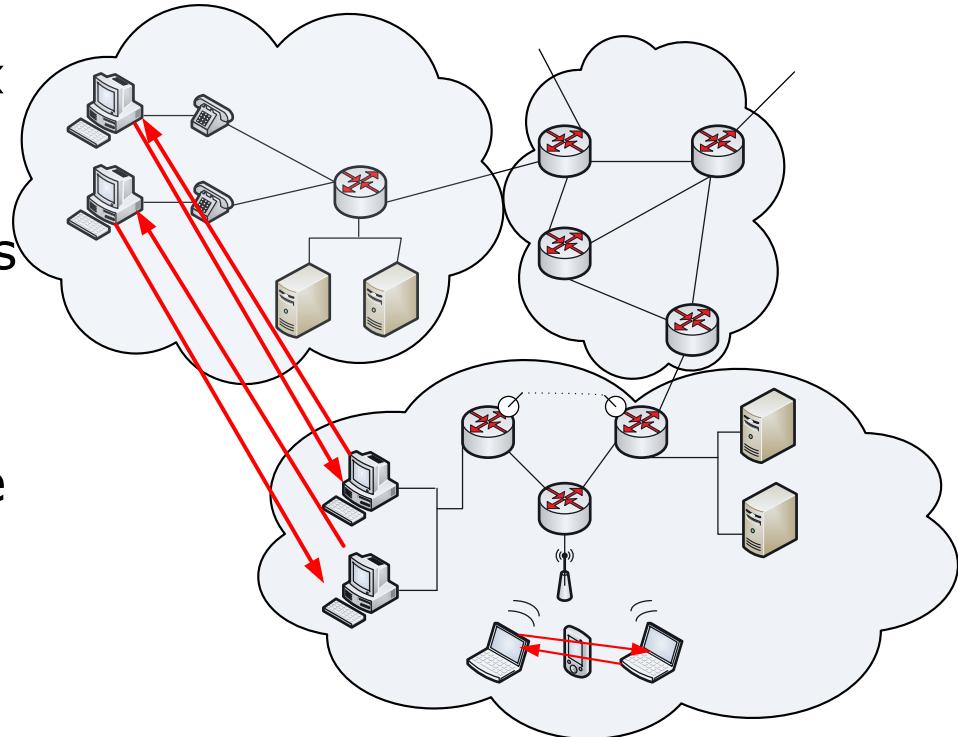
Architecture Client/Serveur (2)

- Un serveur populaire peut rapidement être submergé par les requêtes
 - Ex. Serveur de recherche sur Internet (google) ou serveur réseau social (facebook).
- Data center (ou firme de serveur)
 - D'une centaine à un millier de serveurs
 - Ex. Amazon, Google, Youtube,...



Architecture pure P2P

- Pas de serveur
- Des systèmes terminaux communiquent directement
- Les peers sont connectés par intermittence et changent souvent d'adresse IP
- Ex: Gnutella (partage de fichier) ou PPStream (IPTV)
- Av. : scalable à volonté
- Inc.: difficile à gérer



Hybride entre Client/Serveur et P2P

○ Napster

- Transfert de fichiers P2P
- La recherche de fichier est centralisée
 - Chaque Peer enregistre un fichier au niveau d'un serveur
 - Chaque Peer interroge le même serveur pour la recherche de contenu (fichier)

○ Messagerie instantanée (Skype)

- Chat (discussion) entre deux Peers
- Détection de présence et la redirection vers un Peer est centralisée
 - Chaque utilisateur s'enregistre au près d'un serveur central
 - Un utilisateur contacte le serveur central pour récupérer une adresse IP d'un autre utilisateur

Les processus de communication

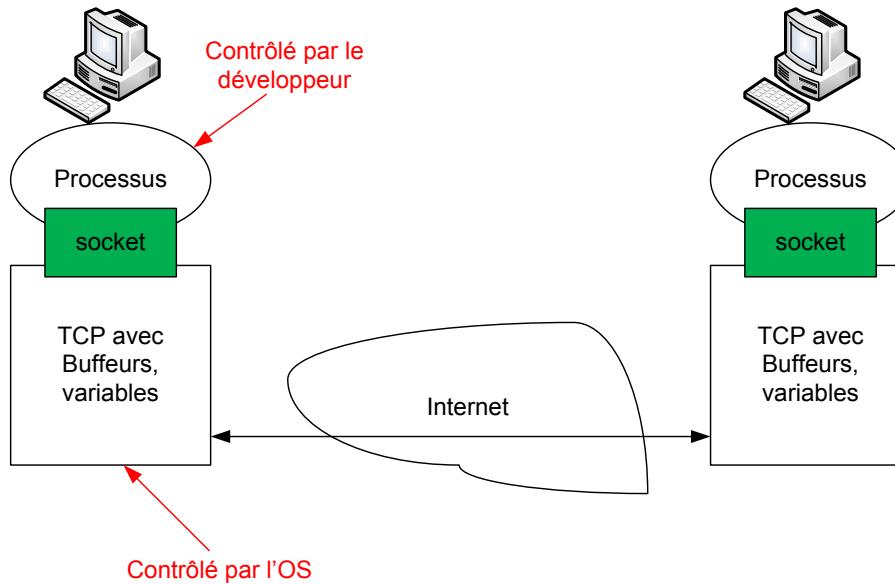
- Un **processus** est un programme qui s'exécute sur un hôte
- Deux processus communiquent dans un même hôte avec des **communications interprocessus** définis par l'OS
- Les processus sur différents hôtes communiquent en utilisant un **protocole applicatif** (messages)

Processus Client : le processus qui initialise la communication

Processus Serveur : le processus en attente d'une demande de connexion

Sockets

- Défini l' interface entre l' application et la couche transport, deux processus communiquent en émettant et en recevant des données via les sockets
- API :
 - Choix du protocole de transport
 - La possibilité de fixer les paramètres de transport



Comment identifier un processus distant ?

- **Adresse IP** de l' hôte distant (32 bits IPv4 et 128 bits IPv6)
- **Numéro de port** – permet de différencier les différents processus locaux auquel le message doit être transmis
- Ex. :
 - Serveur HTTP : 80
 - Serveur de mail : 25
 - Liste des numéros de ports connus (well known)
<http://www.iana.org>

Quel est le service de transport nécessaire à une application?

Perte de données

- Certaines applications (ex., voix) peuvent tolérer des pertes
- D'autres applications (ex., ftp, telnet) nécessitent une fiabilité à 100%

Bandé passante

- Certaines applications (ex., multimedia) nécessitent une bande passante minimale
- D'autres applications ("applis. élastique") utilisent la bande passante disponible

Délais (Timing)

- Certaines applications (e.g., voix sur IP, jeux interactifs) nécessitent des faibles délais

Besoin en service de transport de quelques applications connues

Application	Pertes	Bandé passante	Sensibilité temp.
Transfert de fichier	Sans perte	élastique	Non
e-mail	Sans perte	élastique	Non
Web	tolérant	élastique	Non
Audio/video Temps/réel	tolérant	audio: 5Kb-1Mb video: 10Kb-5Mb	oui, 100's msec
Audio/video enregistré	tolérant	similaire	oui, quelques secs
Jeux interactifs	tolérant	Quelques kbps	oui, 100's msec
Appli financière	Sans perte	élastique	Oui et non

Services offerts par les protocoles de transport Internet

Service TCP:

- *Orienté connexion:* connexion nécessaire entre le client et le serveur
- *Transport fiable* entre le processus émetteur et récepteur
- *Contrôle de flux :* l'émetteur ne submerge pas le récepteur
- *Contrôle de Congestion :* réduit le débit de l'émetteur quand le réseau est congestionné
- *Ne propose pas* de garanties de délai ou de bande passante minimale

Service UDP:

- Transfert de données non fiable
- Ne propose pas de connexion, de fiabilité, de contrôle de flot, de contrôle de congestion, de garantie temporel ou de bande passante

Aucun des deux protocoles ne chiffrent les messages

Applications Internet : application, protocoles de transport

Application	Protocole applicatif	Protocole de transport
e-mail	SMTP [RFC 821]	TCP
Accès distant	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2068]	TCP
Transfert de fichier	FTP [RFC 959]	TCP
streaming multimedia	HTTP	TCP or UDP
Voix sur IP	SIP ou skype (propriétaire)	UDP

Protocole applicatif

- Type des messages à échanger
 - Ex. : requêtes et réponses
- Syntaxe du message
 - Les champs composant un message
- Sémantique du message
 - Le sens des infos. contenues dans le message
- Règles
 - Quand et comment envoyer et répondre à un message
- Protocoles du domaine public
 - Définis dans des RFCs
 - Ex. : HTTP, SMTP
- Protocoles propriétaires
 - Ex. : Kazaa, Skype

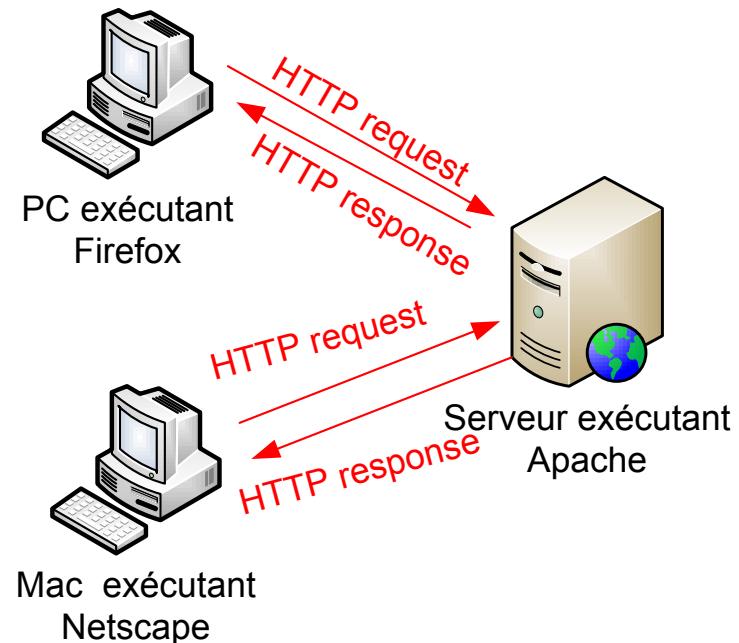
World Wide Web et HTTP

- World Wide Web (WWW) : « au cœur de l'Internet »
- Page Web:
 - Contient des “objects” (objets)
 - Un simple fichier HTML, une image JPEG ou GIF, une applet JAVA, une animation FLASH
 - Adressé par une URL (Uniform Resources Locator)
- La plupart des pages Web pages contiennent :
 - Du texte HTML
 - Objets référencés par le biais de leurs URLs
- L'URL a deux composant : nom d' hôte du serveur et le chemin d' accès à l' objet
www.ifsic.univ-rennes1.fr/L3/pic.gif

HTTP

HTTP: HyperText Transfer Protocol

- Couche applicative Web
- Définit le modèle client/serveur
 - *client*: le navigateur qui demande, reçoit, affiche les objets Web
 - *serveur*: serveur Web envoie les réponses aux requêtes
- HTTP 1.0: RFC 1945
 - Jusqu'à 1997
- HTTP 1.1: RFC 2068
 - À partir de 1998
 - Compatible avec la version 1.0



HTTP (suite)

- Le navigateur web => Agent de l'utilisateur (UserAgent)
 - Client HTTP
 - Ex. Chrome, Internet Explorer, Firefox/Mozilla
- Le serveur web
 - Serveur HTTP
 - Ex. Apache, Microsoft Internet Information Server (IIS), Netscape Enterprise Server

HTTP (suite)

Utilise les services de transport TCP :

- Le client initialise une connexion TCP (crée une socket) avec le serveur, port 80
- Le serveur accepte la demande de connexion TCP du client
- Les messages HTTP (protocole applicatif) sont échangés entre le navigateur web (client HTTP) et le serveur Web
- La connexion TCP est fermée

HTTP est « sans état »

- Le serveur ne maintient aucunes informations au sujets des requêtes précédentes des clients

Les protocoles gardant un “état” sont complexe!

- L'historique doit être gardée
- Si le serveur ou le client s'arrête de fonctionner les états peuvent être inconsistants

Connexions HTTP

- Connexion HTTP non-persistante
 - Au plus, un objet est transmis avec une connexion TCP
 - HTTP/1.0 utilise ce type de connexion
- Connexion HTTP persistante
 - Plusieurs objets peuvent être transmis avec une seule connexion TCP (entre le client et le serveur)
 - HTTP/1.1 utilise ce type de connexion par défaut

HTTP non-persistant

Si un utilisateur entre l' URL

www.ifsic.univ-rennes1.fr/L3/home.index

- 1a. Le client HTTP initialise une connexion TCP avec le serveur HTTP sur le site www.ifsic.univ-rennes1.fr. Le port 80 est choisi par défaut
- 1b. Le serveur HTTP du site www.ifsic.univ-rennes1.fr attend une connexion TCP sur le port 80. "accepte" la connexion, et l' annonce au client
2. Le client HTTP envoie les *requêtes HTTP* (contenant des URLs) par le biais d'une socket TCP. Le message indique que le client demande l' objet (L3/home.index)
3. Le serveur HTTP reçoit le message de requêtes, forme le *message de réponse* contenant l' objet requis, et l' envoie sur une (sa) socket

temps

HTTP non-persistant (suite)

- temps ↓
4. Le serveur HTTP du site ferme la connexion TCP
 5. Le client HTTP reçoit le message de réponse du serveur qui contient un fichier html, et affiche ce dernier. En parsant le fichier html, le client est informé que 10 autres objets jpeg sont référencés par ce fichier.
 6. Les étapes de 1-5 seront répétées pour chacun des 10 objets jpeg

- La page affichée peut être différente suivant le navigateur
- HTTP spécifie uniquement le format et les messages à échanger !

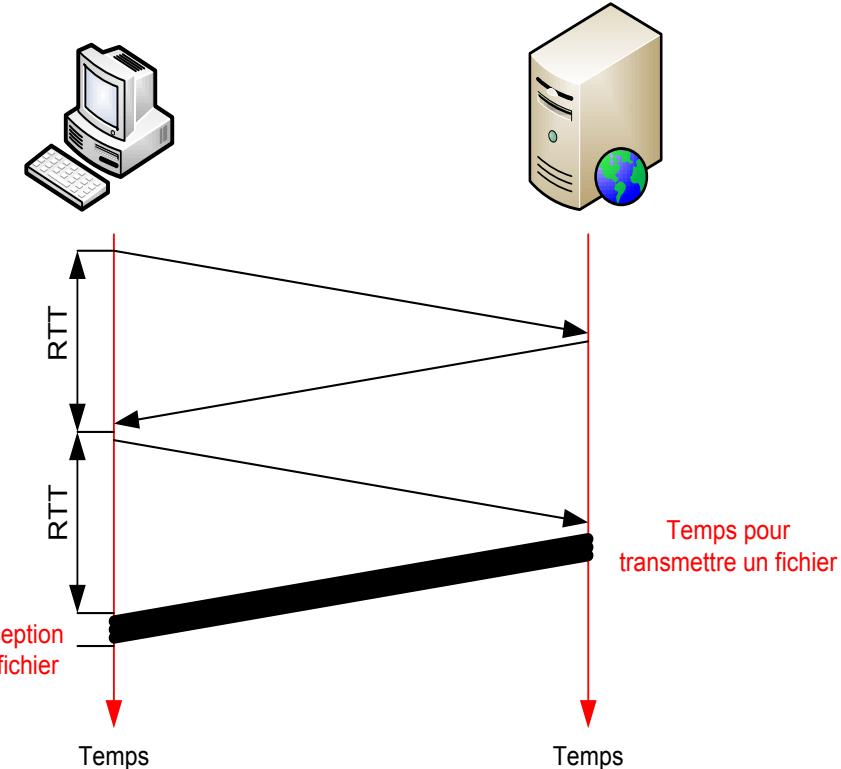
Modélisation du temps de réponse

RTT (Round Time Trip) : le temps nécessaire pour un paquet afin de faire un aller retour entre un client et un serveur

Temps de réponse :

- 1 RTT pour initialiser la connexion TCP
- 1 RTT pour la requête HTTP et quelques octets de la réponse HTTP reçue
- Temps de transmission du fichier

Total = 2RTT +
Temps_transmission_fich



HTTP persistant

HTTP non-persistant

- HTTP/1.0
- 2 RTTs sont nécessaire pour lire chaque objet
- Le serveur interprète les requêtes, répond et ferme la connexion TCP

HTTP persistant

- HTTP/1.1 (par défaut)
- Une seule connexion TCP est ouverte vers le serveur
- Le client envoie une requête pour tous les objets requis dès qu'ils sont référencés dans le fichier HTML

Persistant sans pipelining

- Le client envoie une nouvelle requête qu'après la réception de la réponse de la précédente requête
- Un RTT pour chaque objet

Persistant avec pipelining

- HTTP/1.1 (par défaut)
- Le client envoie une nouvelle requête sans attendre la réponse de la requête précédente
- Les requêtes sont (peuvent) envoyées en même temps
- Les réponses sont envoyées (peuvent) en même temps

Les messages HTTP : *request*

- Deux types de messages HTTP : *request*, *response*
- Message de requête HTTP :
 - ASCII (compréhensible par les humains)

Ligne de requête
(commandes
GET, POST, HEAD)

GET /somedir/page.html HTTP/1.1

Connection: close

User-agent: Mozilla/4.0

Accept: text/html, image/gif, image/jpeg

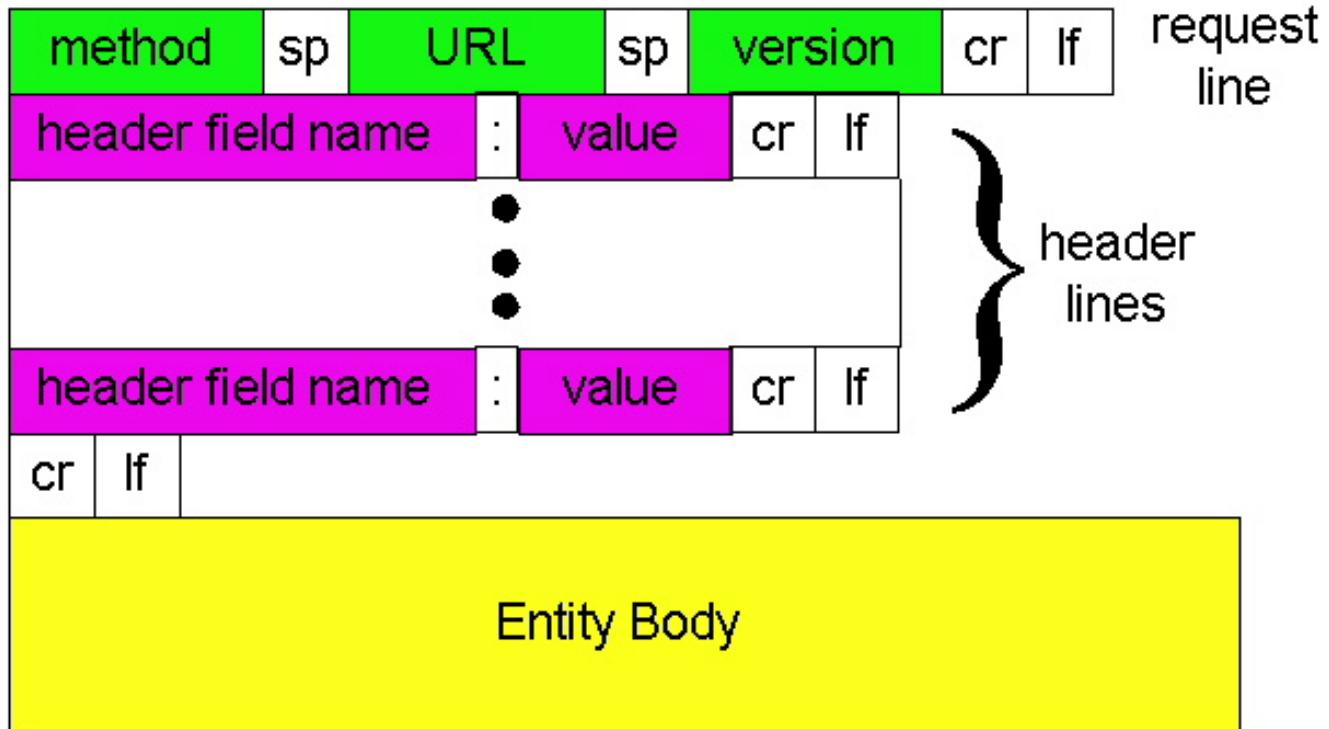
Accept-language: fr

Lignes d'entête

Le retour chariot,
indique la fin
du message

retour chariot, saut de ligne

Format du message Request HTTP



Transfert de formulaire

La méthode POST

- Page web avec formulaire
 - Envoie des données de recherche dans le Entity Body
- Avec la méthode GET
 - Envoie des données de recherche dans l' URL

Types de méthodes

HTTP/1.0

- GET
- POST
- HEAD
 - Demande les informations sur l'objet et non l'objet lui-même
 - L'en-tête uniquement

HTTP/1.1

- GET, POST, HEAD
- PUT
 - Upload le fichier spécifié dans Entity Body vers le lien spécifié dans le champ URL
- DELETE
 - Supprime le fichier spécifié dans le champ URL

Les messages HTTP : *response*

Ligne de statut
(protocole
Code de statut
Etat)

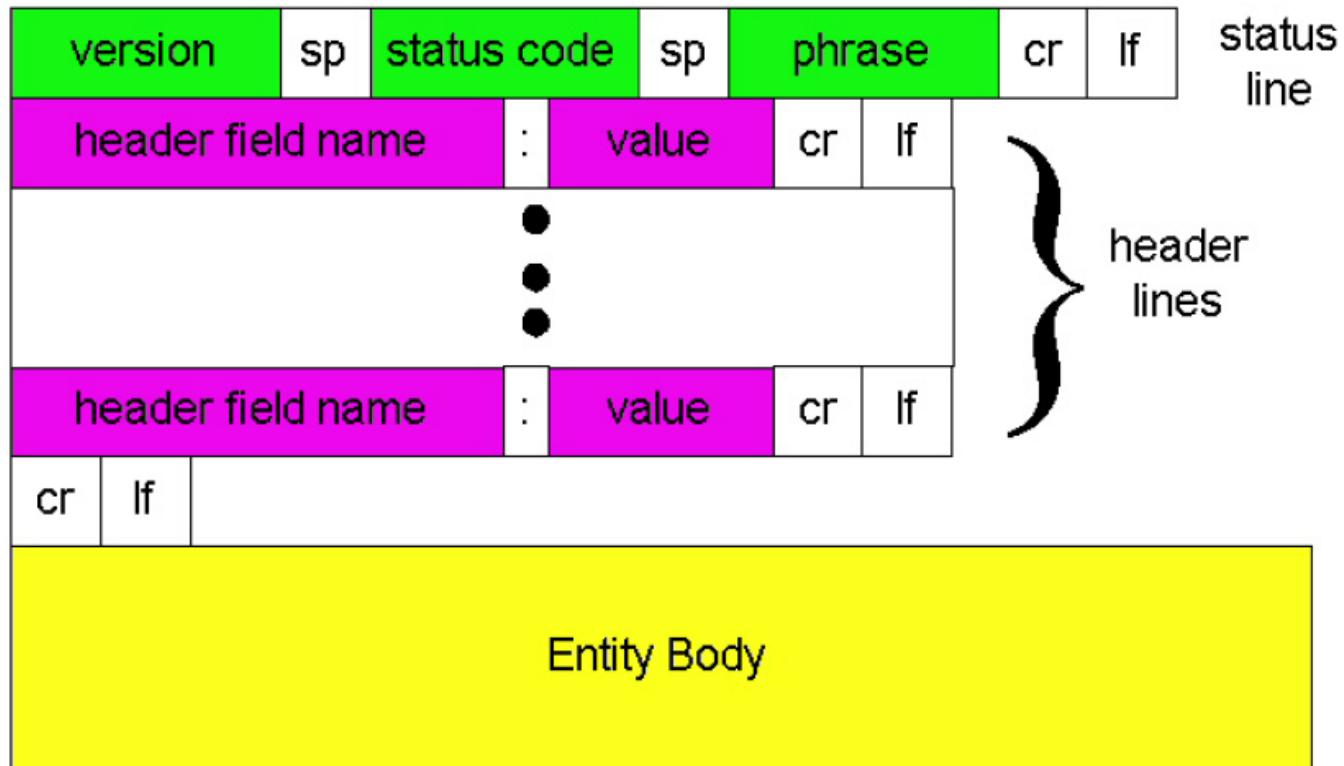
Lignes
d'entêtes

données, Ex.,
Le fichier
html

HTTP/1.1 200 OK
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 1998
Content-Length: 6821
Content-Type: text/html

data data data data data ...

Format du message Response HTTP



Code de réponse HTTP

- Dans la première ligne de la réponse

200 OK

- La requête a réussi et l' objet demandé est à la suite

301 Moved Permanently

- L' objet demandé a changé définitivement de place, son nouvel emplacement est donné dans la suite du message

400 Bad Request

- La requête est erronée

404 Not Found

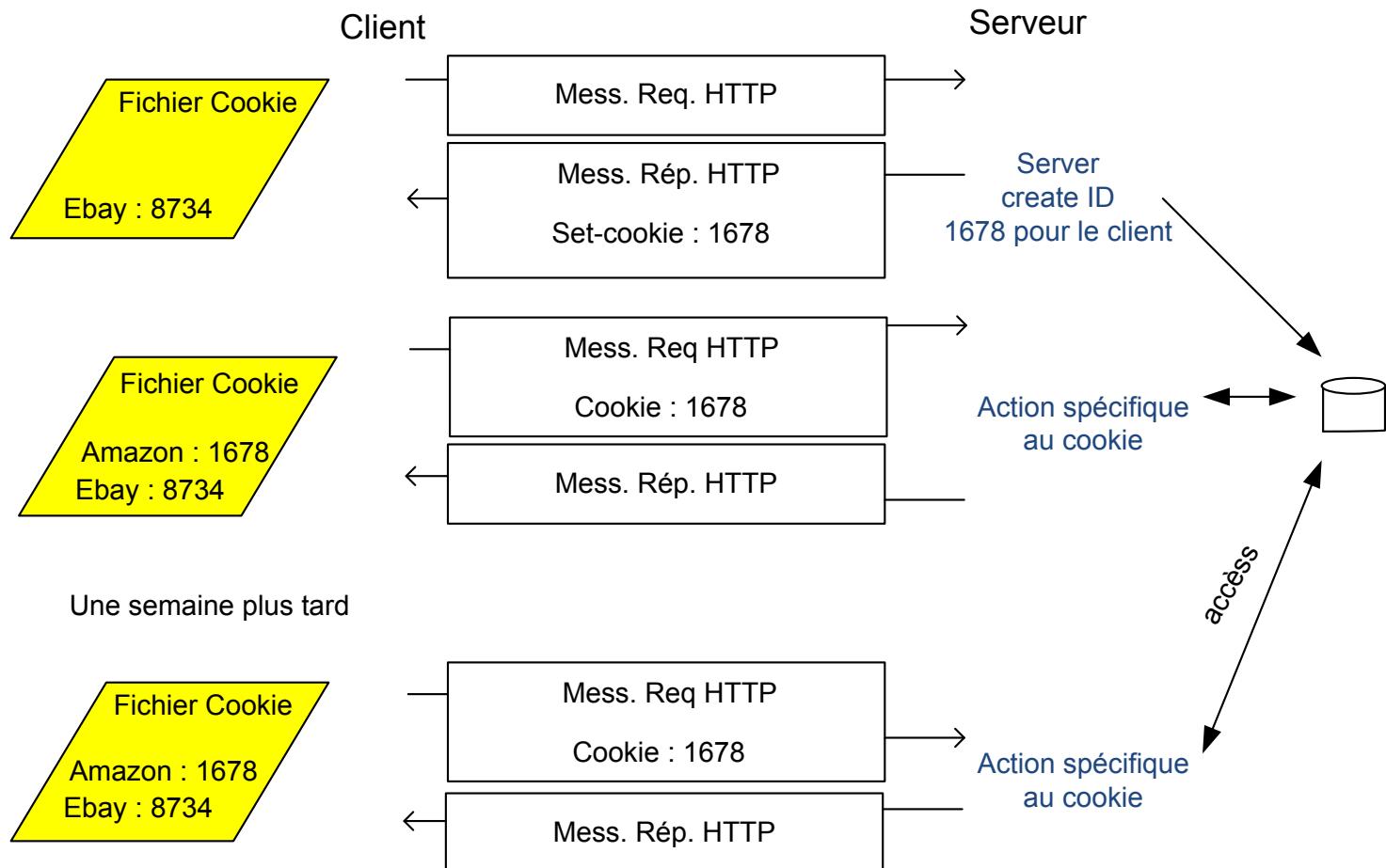
- Le document demandé n' est pas disponible sur le serveur

505 HTTP Version Not Supported

Sauvegarde d' états entre clients et serveurs - cookies

- Beaucoup de sites emploient les cookies
 - Identifier les clients
 - Personnaliser le contenu aux clients
- Quatre composants
 - Une ligne d' en-tête cookie dans la réponse HTTP
 - Une ligne d' en-tête cookie dans la requête
 - Le fichier cookie est sauvegardé sur la machine du client et il géré par le navigateur du client
- Ex. :
 - Toto a accès à Internet toujours du même PC
 - Il visite un site de e-commerce pour la première fois
 - A l' initialisation de la requête HTTP, le site créé un unique identifiant ID et l' insère dans une base de donnée pour une future identification de Toto

Cookies (suite)



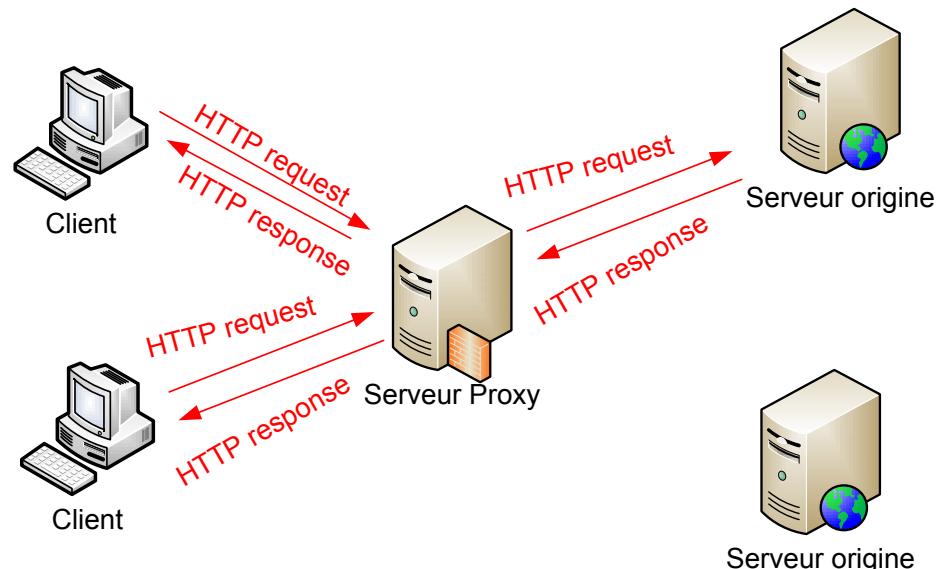
Cookies (suite)

- Fonctionnalités associées à l' utilisation de cookies
 - Autorisation
 - Achats en ligne (l' exemple du panier)
 - L' état d' une session Web
- Par contre
 - Les cookies permettent aux sites web de connaître d' avantage sur vos habitudes sur le net
 - Les moteurs de recherche
 - Les publicitaires

Cache Web – Serveur Proxy

But: satisfaire la requête du client sans utiliser le serveur origine

- L'utilisateur pointe son navigateur vers le cache :
- Le client envoie toutes ses requête HTTP vers le cache web
 - Si l'objet est dans le cache, on le retourne
 - Sinon on demande au serveur initial et on répond ensuite à la requête
- Le cache web a double fonctions : client et serveur



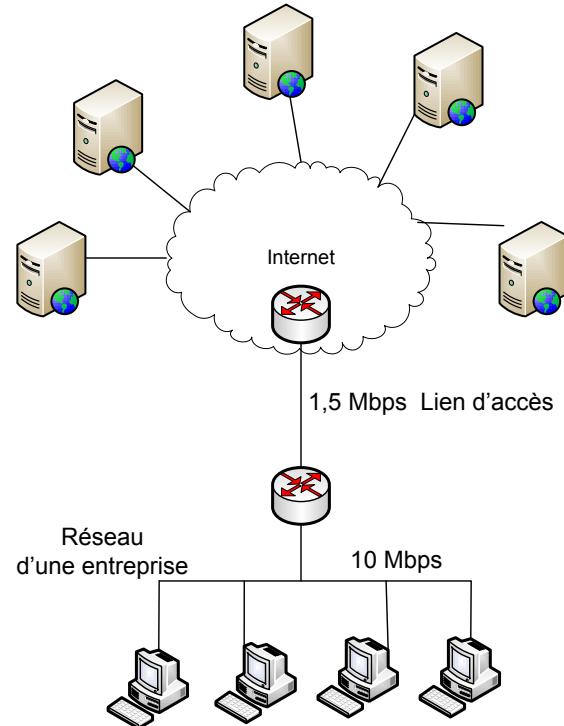
Cache Web : exemple

○ Suppositions

- Taille moyenne d'un objet = 100000 bits
- Moy. des requêtes HTTP émise du réseau d'entreprise vers les serveurs web = 15 req/sec
- RTT du routeur local vers n'importe quel serveur = 2 sec

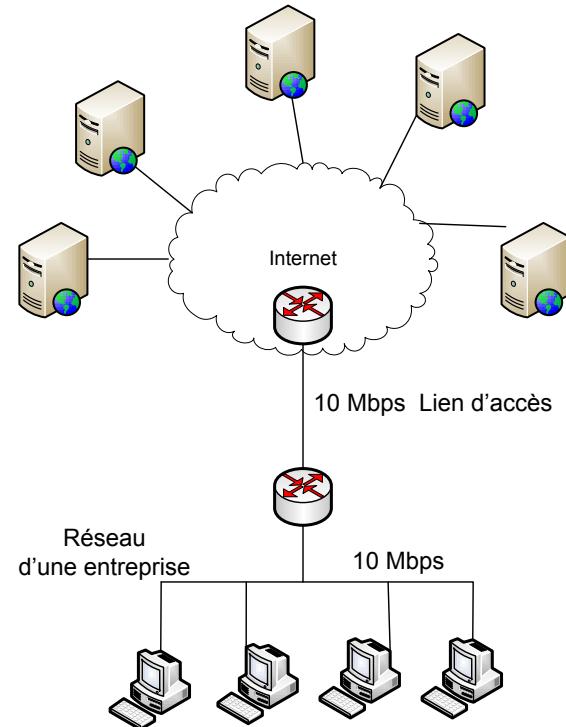
○ Conséquences

- Utilisation du LAN = 15%
- Utilisation du lien d'accès = 100%
- Délai total = délai sur Internet + délai d'accès + délai sur le LAN = 2 sec + minutes + millisecondes



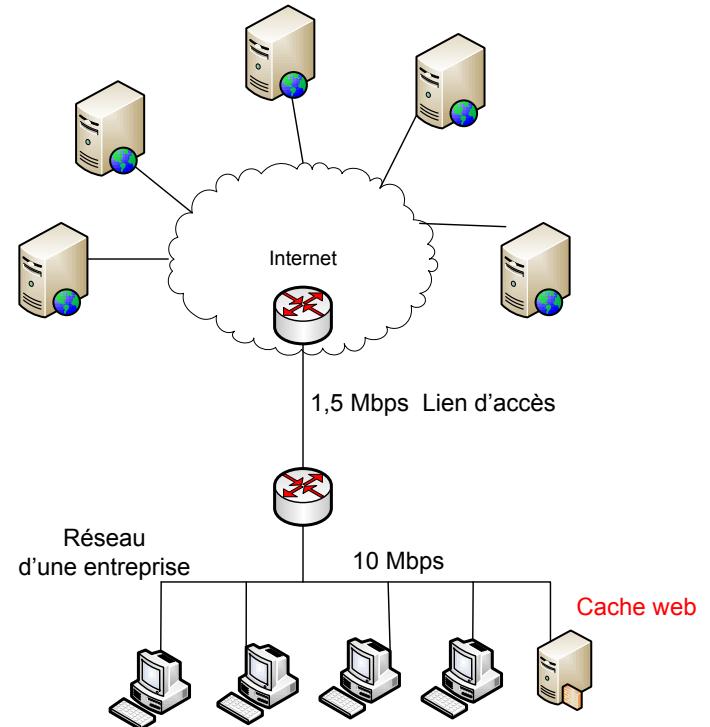
Cache Web : exemple (suite)

- Une solution
 - Augmenter le débit du lien d'accès, 10 Mbps.
- Conséquences
 - Utilisation du LAN = 15%
 - Utilisation du lien d'accès = 15%
 - Délai total = délai sur Internet + délai d'accès + délai sur le LAN = 2 sec + millisecondes + millisecondes
- Solution très coutante



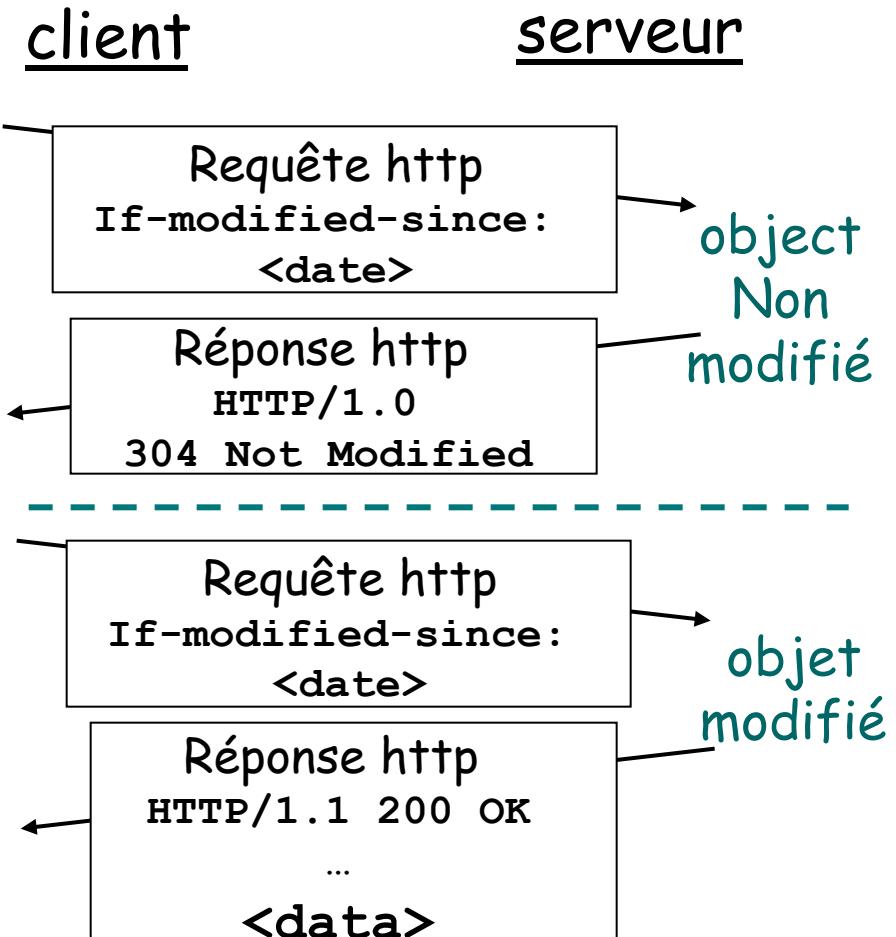
Cache Web : exemple (suite)

- Installer un Cache Web
 - On suppose que le cache satisfait 40% des requêtes
- Conséquences
 - 60% des requêtes sont traitées par le serveur origine
 - Utilisation du lien d'accès réduite à 60%, impliquant des délais un peu plus faible (on suppose 10 msec)
 - Moy. Délai total = délai sur Internet + délai d'accès + délai sur le LAN = $0,6 * 2 \text{ sec} + \text{millisecondes} < 1,4 \text{ secondes}$

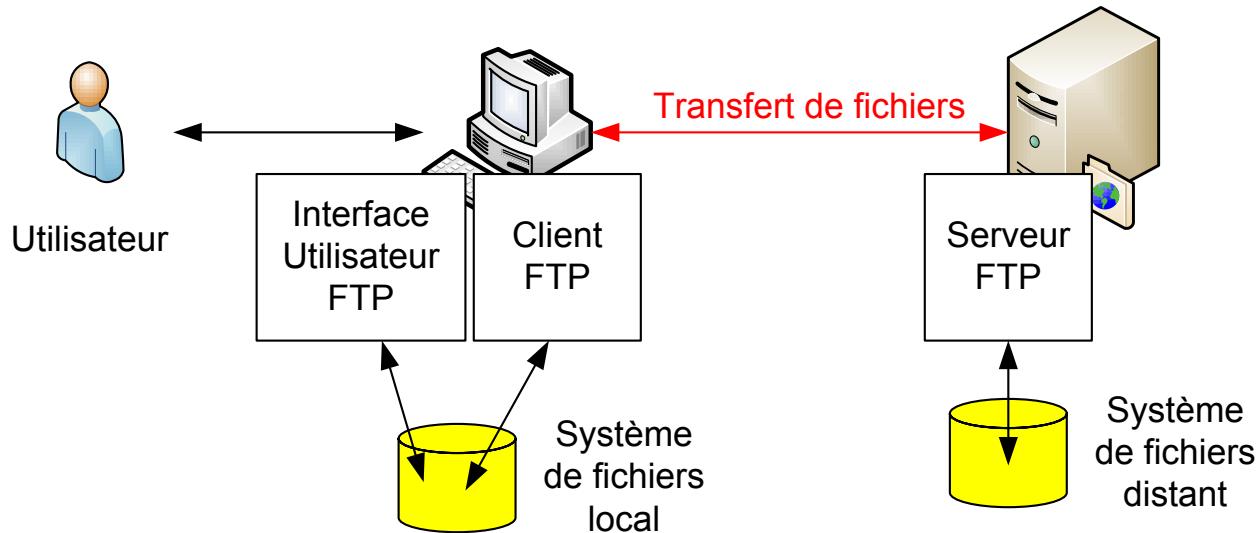


GET conditionnel

- Objectif : ne pas envoyer un objet que le client a déjà dans son cache
- client: spécifie la date de la copie cachée dans la requête http
 - If-modified-since: <date>**
- serveur: la réponse est vide si la copie cachée est à jour **HTTP/1.0 304 Not Modified**



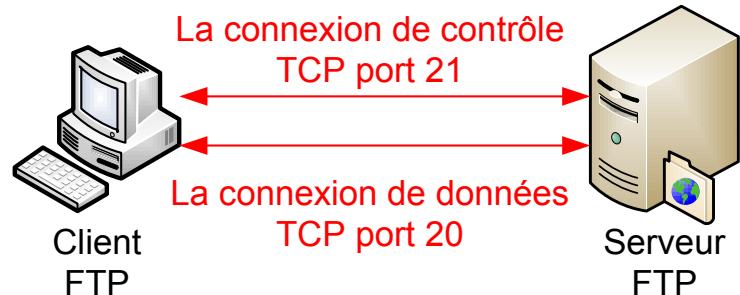
FTP : le protocole de transfert de fichier



- Transfert de fichiers du/vers une machine
- Basé sur le modèle Client/Serveur
 - Le client : la partie initialisant la connexion
 - Le serveur : la machine distante
- FTP : RFC 959
- Le serveur FTP écoute sur le port 21

FTP: la connexion de contrôle et la connexion de données

- Le client FTP contacte le serveur sur le port 21
 - Utilisation de TCP
- Le client obtient les autorisations à travers le canal (connexion) de contrôle
- Le client navigue dans le système distant en envoyant des commandes sur le canal de contrôle
- Dès que le serveur reçoit une demande de transfert de fichier il ouvre une connexion pour les données sur un nouveau port TCP
- Dès que le fichier est téléchargé le serveur clos le canal de données



- Le serveur ouvrira une nouvelle connexion de donnée pour le transfert d'un nouveau fichier
- La connexion de contrôle : **Hors bande**
- Le serveur FTP maintient l'état de la connexion :
 - Le répertoire courant, la connexion précédente

FTP en pratique

```
CLI$ ftp ftp://ftp.free.fr
...
220 Welcome to ProXad FTP server
230 Login successful.
...
ftp> cd mirrors/ftp.ubuntu.com/dvd/current
250 Directory successfully changed.
ftp> ls
...
-rw-rw-r-- 1 ftp     ftp    1468051456 Jun 12 2012 quantal-dvd-i386.iso
...
226 Directory send OK.
ftp> get quantal-dvd-amd64.iso
local: quantal-dvd-amd64.iso remote: quantal-dvd-amd64.iso
150 Opening BINARY mode data connection for quantal-dvd-amd64.iso (1476702208 bytes).
100% |*****| 1408 MiB 4.04 MiB/s 00:00 ETA
226 File send OK.
1476702208 bytes received in 05:48 (4.04 MiB/s)
ftp> quit
```

FTP : commandes et réponses

Ex. de commandes

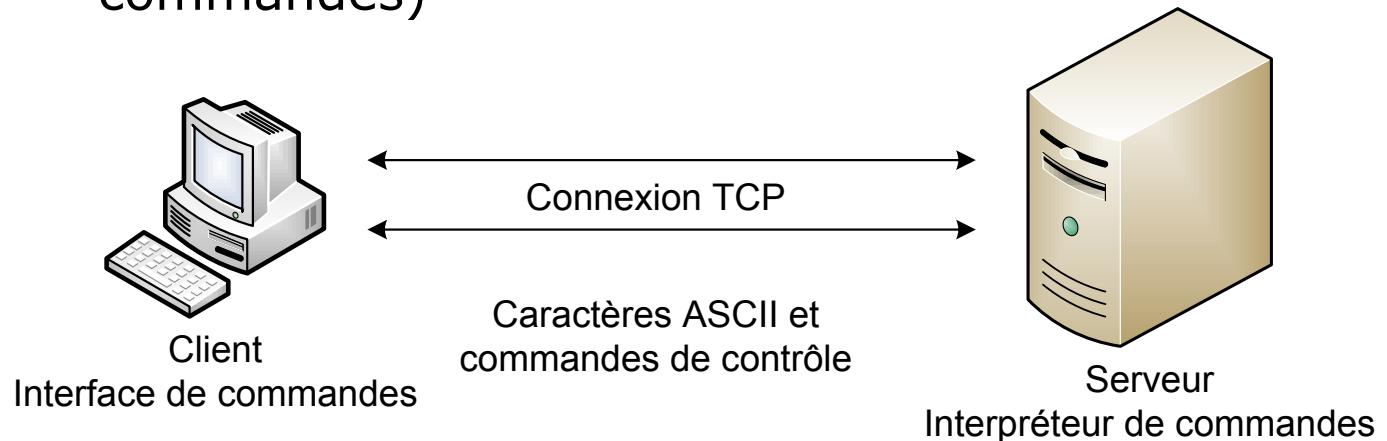
- Envoie des commandes (en code ASCII, 7-bits) sur le canal de contrôle
- **USER login**
- **PASS mot_passe**
- **LIST** renvoie la liste des fichiers présents dans le répertoire courant
- **RETR** fichier récupère un fichier
- **STOR** fichier sauvegarde un fichier sur le hôte distant

Ex. de réponses

- Statut et phrases (comme HTTP)
- **331 Username OK, password required**
- **125 data connection already open; transfer starting**
- **425 Can't open data connection**
- **452 Error writing file**

TELNET

- TELNET : protocole réseau pour le contrôle de machines à distance
- telnet => logiciel qui utilise le protocole TELNET
- TELNET
 - permet de connecter un client (système composé d'un affichage et d'un clavier) à un serveur (interpréteur de commandes)



TELNET

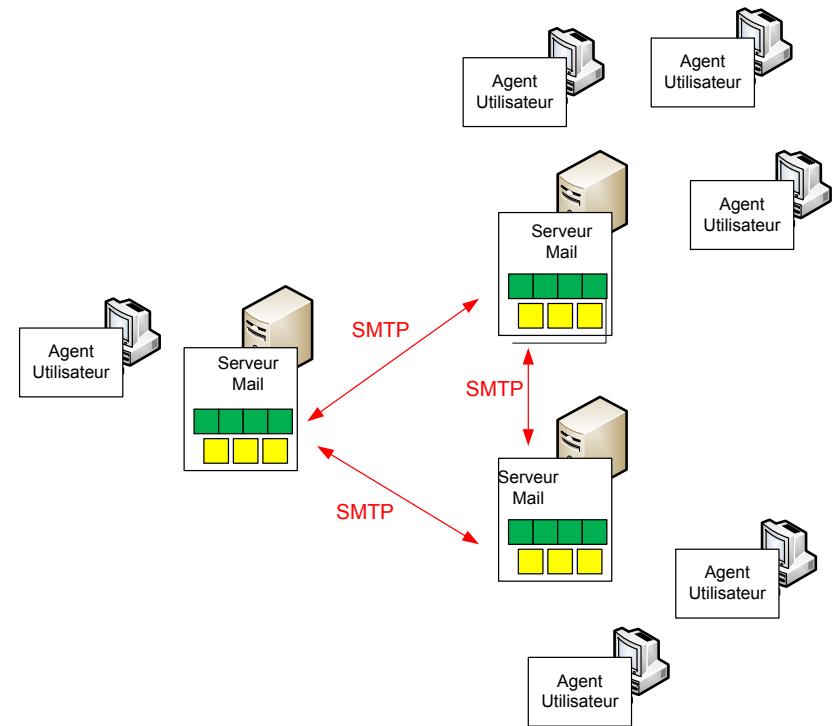
- Port 23
- Données et contrôle sur la même connexion TCP
- Network Virtual Terminal (NVT)
- Négociation des options entre le client et le serveur (jeu des caractères, mode ligne ou caractère, etc.)

SSH

- Secure Shell
- Alternative à TELNET
- Protocole permettant une communication sécurisée entre le client et le serveur - les données sont cryptées
- Port 22

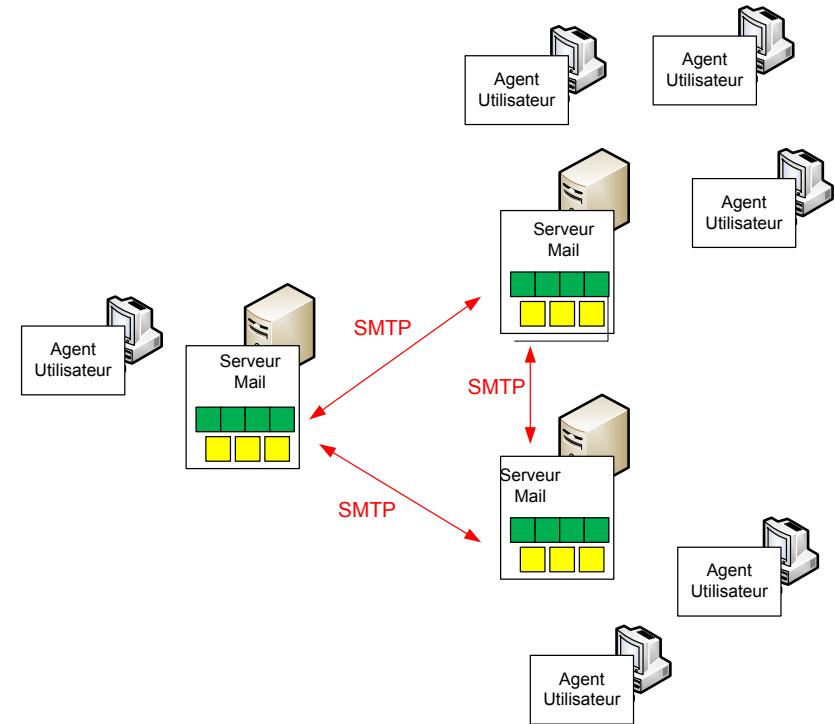
Messagerie électronique

- Trois composants majeurs
 - L' agent de l' utilisateur (user agent)
 - Le serveur de messagerie
 - Simple Mail Transfer Protocol : SMTP
- Agent de l' utilisateur (User Agent)
 - Composer, édition, lecture des messages mail
 - Ex. : Mail, Outlook, Thunderbird



Messagerie électronique

- Serveur de messagerie
 - Boite email : contient les messages à destination d'un utilisateur
 - File d'attente des messages : contient les messages en attente d'émission
 - Le protocole SMTP : est utilisé entre serveurs de messagerie pour l'envoi de mails
 - Client : serveur SMTP envoyant un mail
 - Serveur : serveur SMTP destinataire du mail

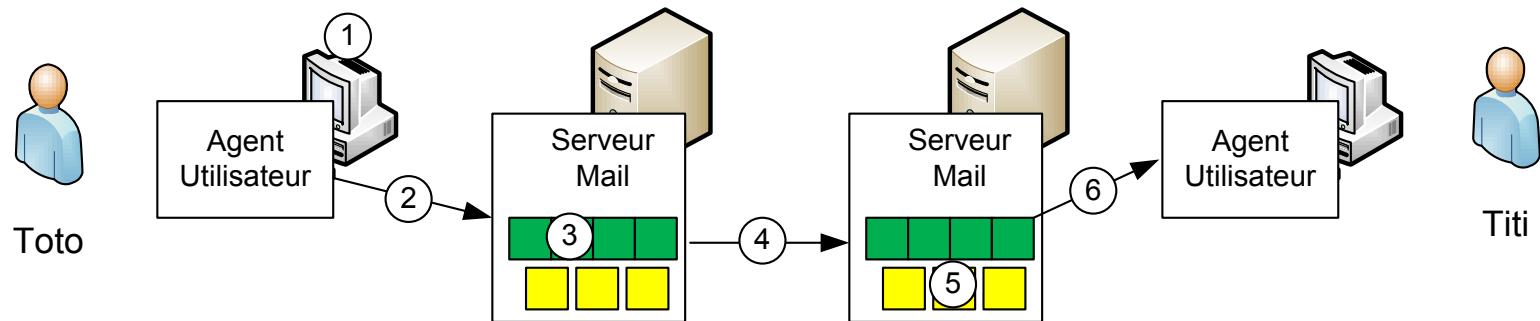


Messagerie électronique : SMTP

- Basé sur TCP pour l'envoie fiable de message électronique du client vers le serveur, sur le port 25
- Transfert directe entre serveurs
- Trois phases pour le transfert
 - Etablissement de la connexion
 - Transfert de message
 - Clôture de la connexion
- Interaction commandes / réponse
 - **Commandes** : ASCII texte
 - **Réponses** : Status de la commande
- Les messages sont codés en ASCII sur 7 bits

Scénario : message de toto vers titi

- 1- Toto utilise son UA pour composer un message à titi@univ-rennes1.fr
- 2- L' UA de toto envoie un message vers son serveur de messagerie
- 3- Le serveur de messagerie utilisé par toto ouvre une connexion SMTP/TCP avec le serveur de messagerie de titi
- 4- Le serveur de messagerie de toto envoie le message à travers la connexion TCP
- 5- Le serveur de messagerie de titi place le message dans la boite mail de titi
- 6- titi demande à son UA de lire le message



Exemple d'interaction SMTP

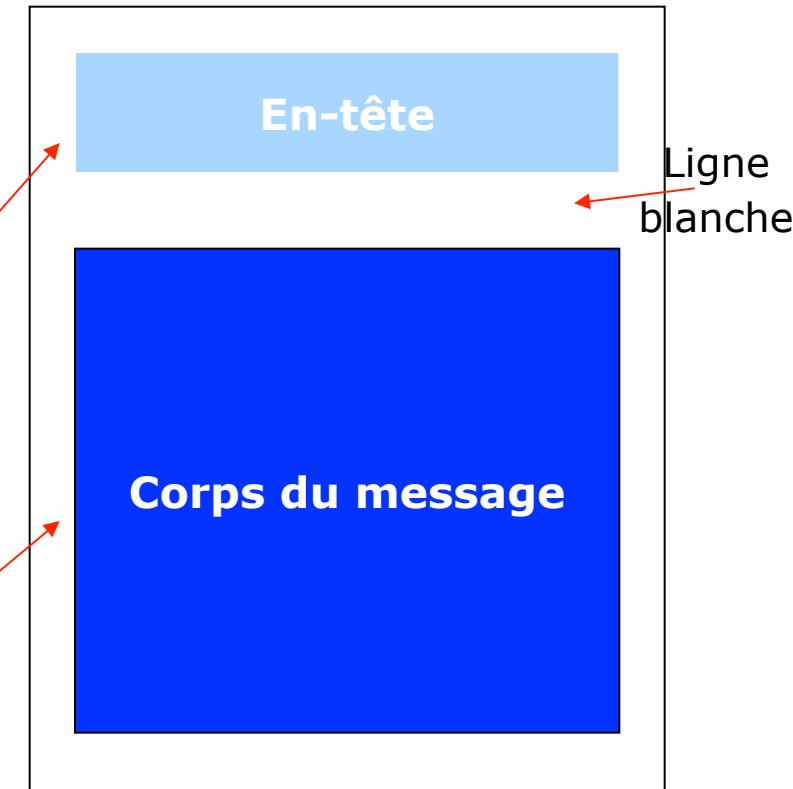
S: 220 univ-rennes1.fr
C: HELO quelquepart.fr
S: 250 hello quelquepart.fr, pleased to meet you
C: MAIL FROM: toto@quelquepart.fr
S: 250 toto@quelquepart.fr... Sender ok
C: RCPT TO: titi@univ-rennes1.fr
S: 250 titi@univ-rennes1.fr
C: DATA
S: 354 Enter mail, end with "." on a line itself
C: Blablablablabla
C: Blablabla
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221univ-rennes1.fr closing connection

SMTP (fin)

- SMTP utilise une connexion persistante
- SMTP impose l' utilisation de caractères ASCII sur 7 bits (message et En-tête)
- Le serveur détermine la fin du message grâce à la suite CRLF.CRLF
- Comparaison avec HTTP :
 - HTTP : technologie **Pull**, le client demande l'objet
 - SMTP : technologie **Push**, le client envoie l'objet
 - Les deux utilisent des commandes/réponses codés en ASCII, et des codes retours (Status)

Format du message électronique

- SMTP => protocole pour l'échange de message électronique (RFC 822 : format des messages)
- Lignes des en-têtes
 - To : vers
 - From : de
 - Subject : objet **(différent d'une commande SMTP)**
- Corps du message
 - Le message uniquement en caractères ASCII



Format du message : l'extension multimédia

- MIME : Extension multimédia, RFC 2045, 2056
- Lignes en plus dans l'en-tête pour déclarer le contenu MIME

Version MIME

Méthode utilisée pour encoder les données

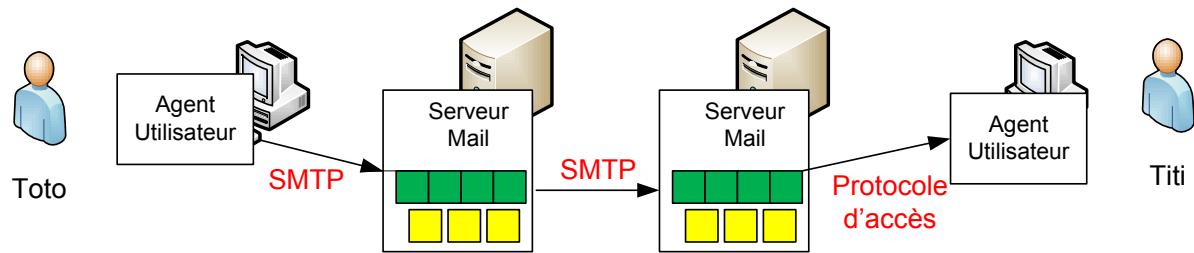
Type et sous-type des données multimédia

Données encodées

From : toto@crepes.fr
To : titi@hamburger.edu
Subject : Image d'une crepe
MIME-Version : 1.0
Content-Transfer-Encoding : base64
Content-Type : image/jpeg

Base64 encoded data
.....
.....
..... **base64 encoded data**

Protocoles d' accès à la messagerie



- SMTP : Livraison et sauvegarde vers le serveur de messagerie du récepteur
- Protocole d' accès à la messagerie : récupérer les messages du serveur
 - POP : Post Office Protocol [RFC 1939]
 - Authentification (agent-> serveur) et téléchargement
 - IMAP : Internet Mail Access Protocol [RFC 1730]
 - Manipulation des messages sur le serveur
 - HTTP : Hotmail, Yahoo Mail, Gmail, etc.

POP3 et IMAP

POP3

- Mode téléchargement et suppression
 - Pas de possibilité de retrouver le message si on change de UA
- Mode téléchargement et pas de suppression
 - Possibilité d'avoir une copie d'email sur chaque client utilisé
- POP3 est sans état

IMAP

- Maintenir les messages sur le serveur
- Permettre aux utilisateurs d'organiser leur boite aux lettres en répertoire
- IMAP maintient des états

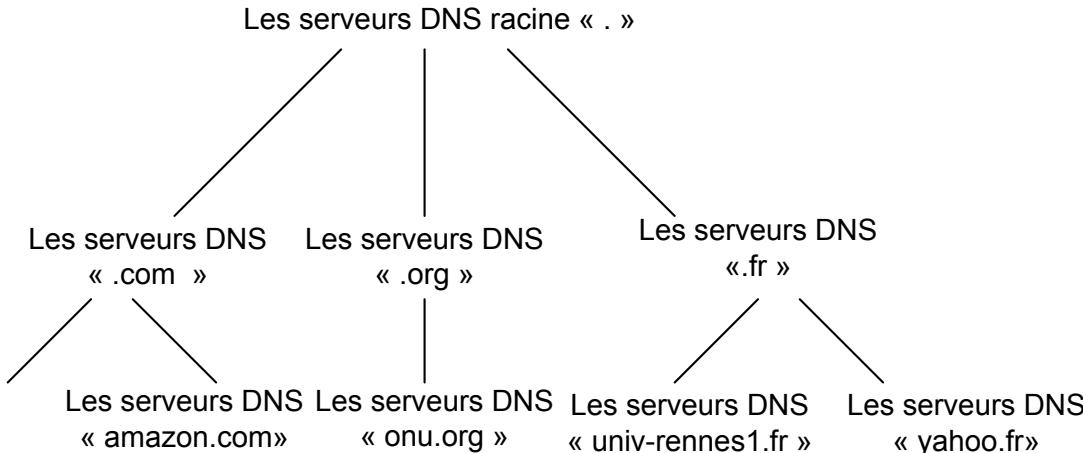
DNS : Domain Name System

- Personnes => plusieurs identifiants
 - Numéro de Sécurité Social, nom, passeport ..
- Hôtes et routeurs sur Internet
 - Une adresse IP (32 bits) pour identifier le destinataire d'un paquet
 - « nom », ex., www.yahoo.fr , utilisé par les humains
- Q : comment trouver une correspondance entre un nom et une adresse IP ?
- Système de résolution de nom ou DNS
 - Une BD distribuée et hiérarchique
 - Plusieurs serveurs de nom existent
- Protocole niveau applicatif
 - Hôtes et serveurs de nom communiquent pour résoudre des noms de hôtes (Nom/Adresse IP correspondance)

Services DNS

- Résolution
 - Nom de hôte → **Adresse IP**
text.esams.wikimedia.org. IN A 91.198.174.232
- Alias
 - **Nom d'alias canonique**
fr.wikipedia.org. IN **CNAME** text.wikimedia.org.
 - Alias pour les **serveurs d'email**
wikimedia.org. IN **MX** 10 mchenry.wikimedia.org.
- Dispersion de charge (load balancing)
 - Plusieurs adresses IP correspondant à un nom

DNS hiérarchique



Un client veut récupérer l' @IP de www.univ-rennes1.fr

- Interrogation d'un des serveurs *racines* pour trouver le serveur DNS du *.fr*
- Le client interroge le serveur DNS de *.fr* afin de trouver le serveur DNS de *univ-rennes1.fr*
- Le client interroge le serveur DNS de *univ-rennes1.fr* pour obtenir l' @IP de www.univ-rennes1.fr

DNS Racines

- Ils sont contactés par des serveurs DNS, qui n' arrivent pas à résoudre une requête
- Le serveur de nom racine
 - Contacte un serveur de nom faisant autorité sur un domaine, s' il n' arrive pas à résoudre une requête
 - Obtient une association (@IP/ nom)
 - Renvoie le résultat vers le serveur de nom local

13 serveurs dans
le monde



Les TLD (Top Level Domain) et les serveurs authoritative (autorité) et

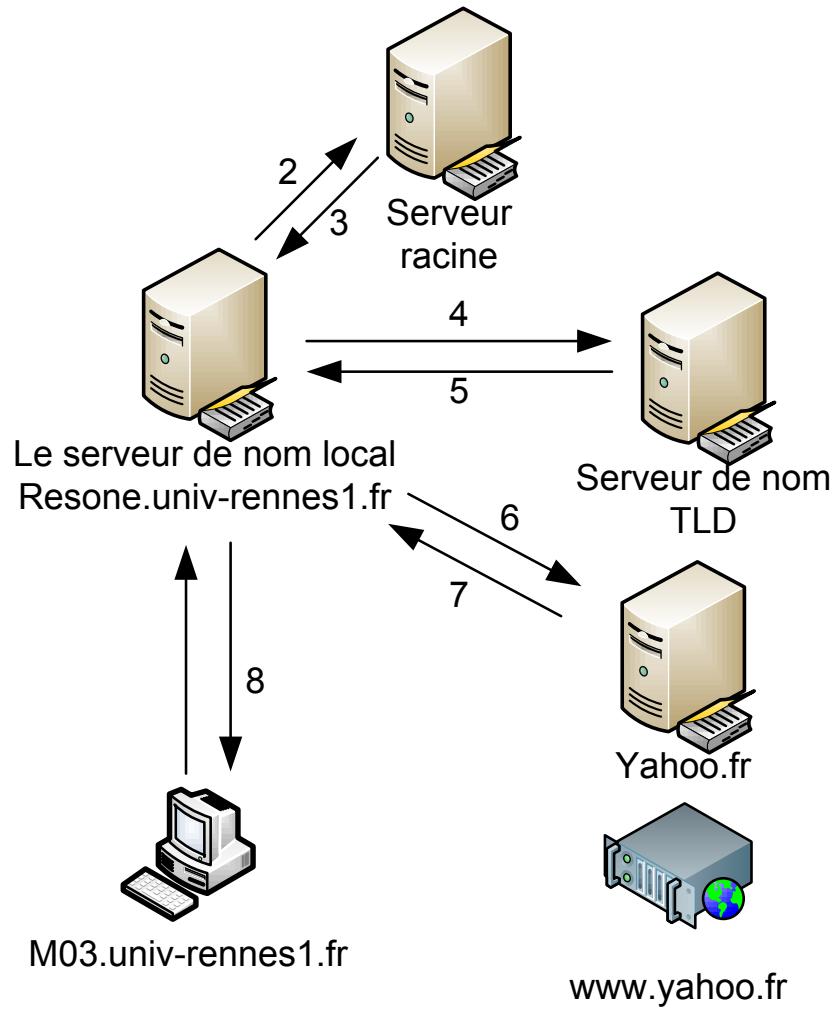
- Les serveurs de nom TLD : Responsables des domaines com, org, net, edu, tv, .., et ceux représentant les pays fr, es, uk, dz
 - Ex., le .fr est géré par l' AFNIC
- Les serveurs de nom faisant autorité : les serveurs de nom des entreprises, d'instituts, et organisation (Ex. Web et mail)
 - Peuvent être gérés par l'entreprise ou l'organisation

Serveur de nom local

- Il n'appartient pas forcément à la hiérarchie
- Chaque FAI ou Institution (Université) a son serveur local
 - Le serveur de nom par défaut
- Il reçoit les requêtes de résolution de nom émises par les hôtes
 - Il fonctionne comme un proxy, il transfert les requêtes vers le système DNS hiérarchique

Exemple

- Une machine (m03.univ-rennes1.fr) veut résoudre le nom www.yahoo.fr



DNS : le système de cache

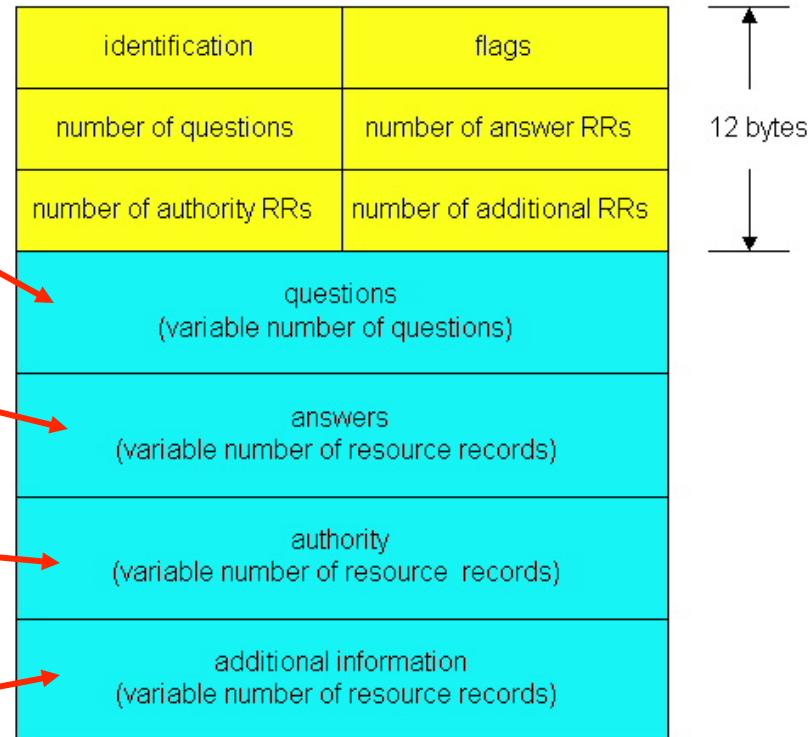
- Chaque résolution est mis en cache
 - Cette entrée du cache a une durée de vie
 - TLD serveurs sont mis en cache au niveau des serveurs locaux
 - Eviter les requêtes vers les serveurs racines
- RFC 2136

DNS

- Sauvegarde distribuée des données : Resource records (RR)
 - Format RR : (nom, valeur, type, ttl)
- Type A
 - Nom : nom de la machine
 - Valeur : adresse IP
- Type CNAME
 - Nom est un alias
 - Valeur : un nom canonical
- Type MX
 - Valeur est le nom d'un serveur de mail associé à un nom de domaine

DNS format message

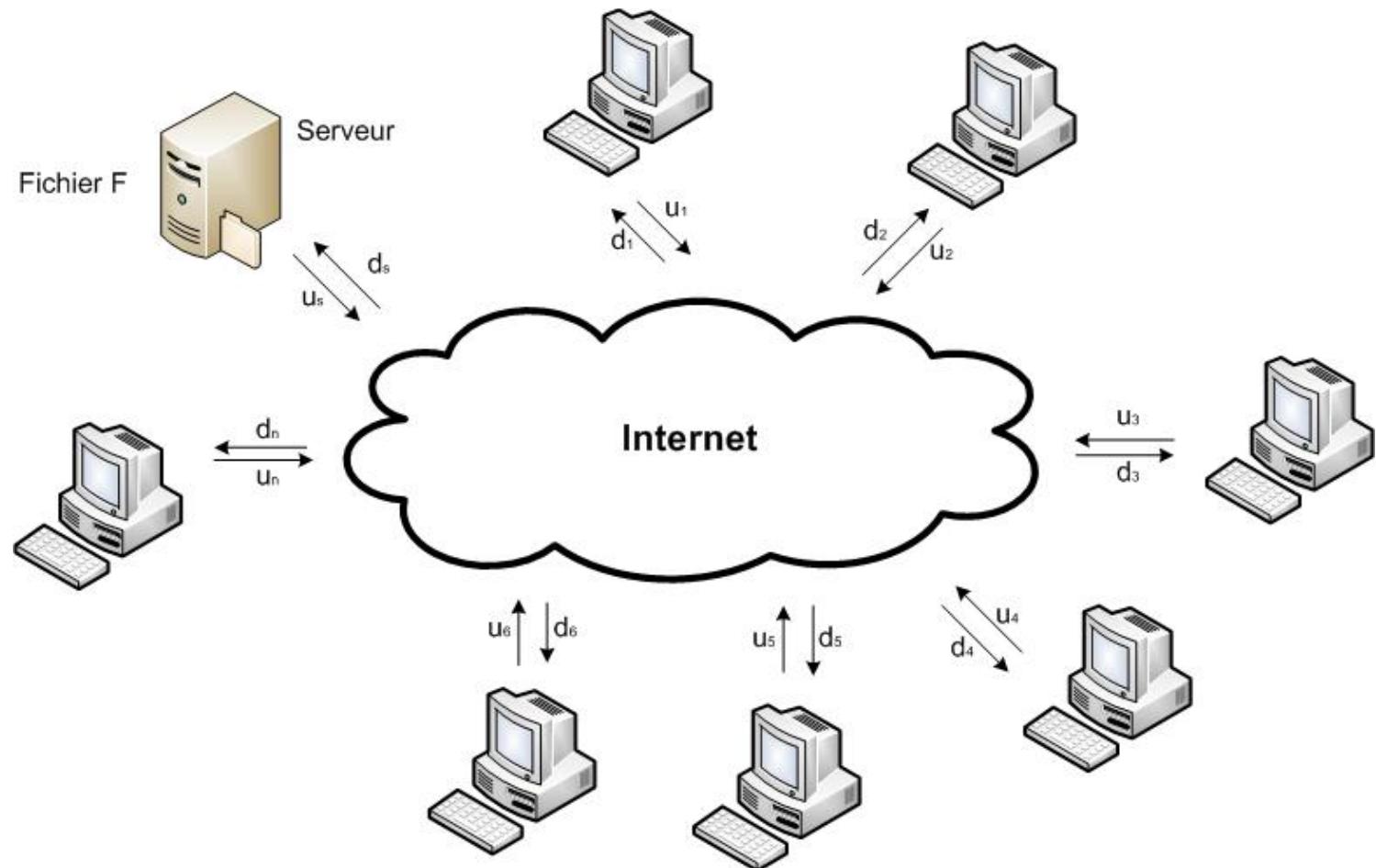
Nom, champs pour la requête
RRs en réponse à une requête
Enregistrement authoritative servers
Information complémentaire



Architecture P2P

- Partage de fichier : BitTorrent,
Gnutella
- Téléphonie sur IP : Skype

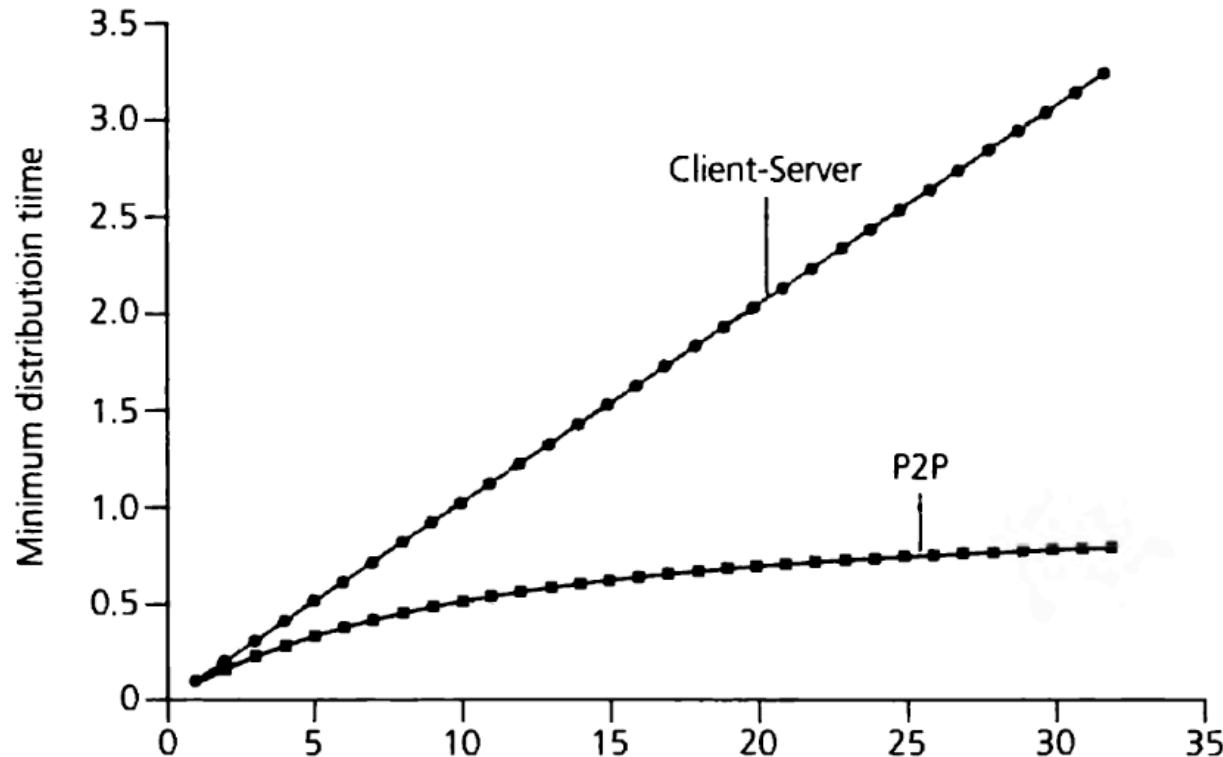
Partage de fichier : client/serveur vs p2p



Partage de fichiers : client/serveur vs p2p

- Architecture client/serveur
 - Temps pour satisfaire la demande de N clients (minimum) => $D_{cs} \geq \text{Max } (NF/u_s, F/d_{\min})$
- Architecture p2p
 - Initialement, uniquement le serveur a le fichier
 - Temps pour satisfaire la demande de N clients (minimum) => $D_{cs} \geq \text{Max } (F/u_s, F/d_{\min}, NF/u_s+u_1+\dots+u_N)$

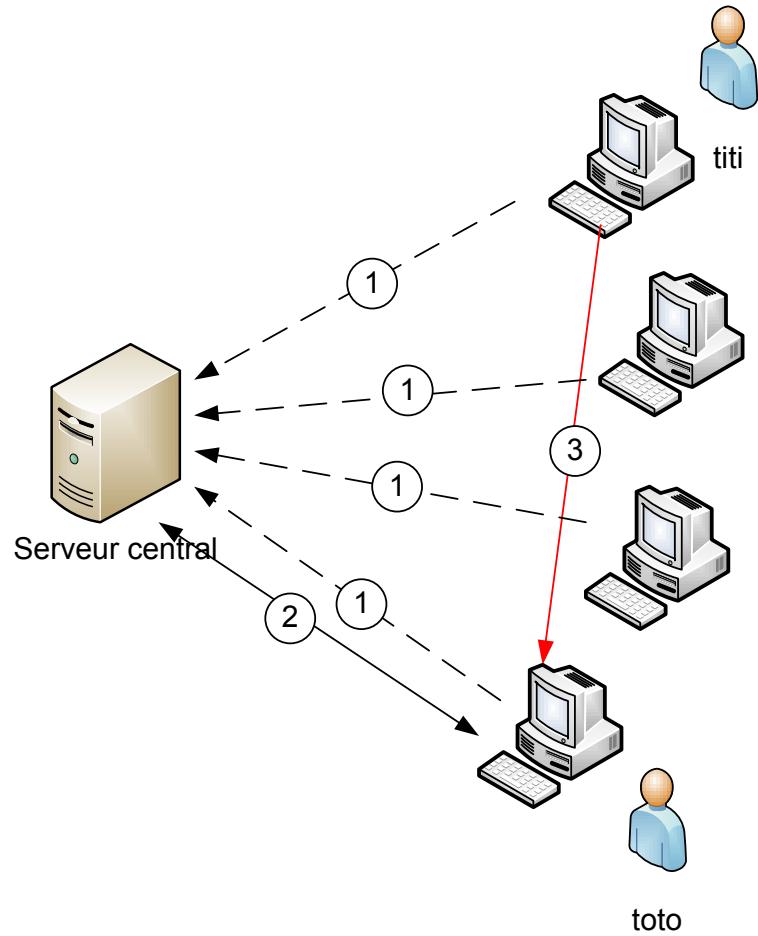
Partage de fichiers : client/serveur vs p2p



- On suppose que tous les peers ont la même capacité d'envoi
 - $F/u = 1$ heure, $u_s = 10u$.

P2P centralisé

- Le fonctionnement de « Napster »
 - 1) lorsqu' un peer se connecte, il envoie au serveur central, son IP, et le contenu à partager avec les autres peers
 - 2) Toto demande le « fich1 »
 - 3) Toto demande le fichier « fich1 » à titi



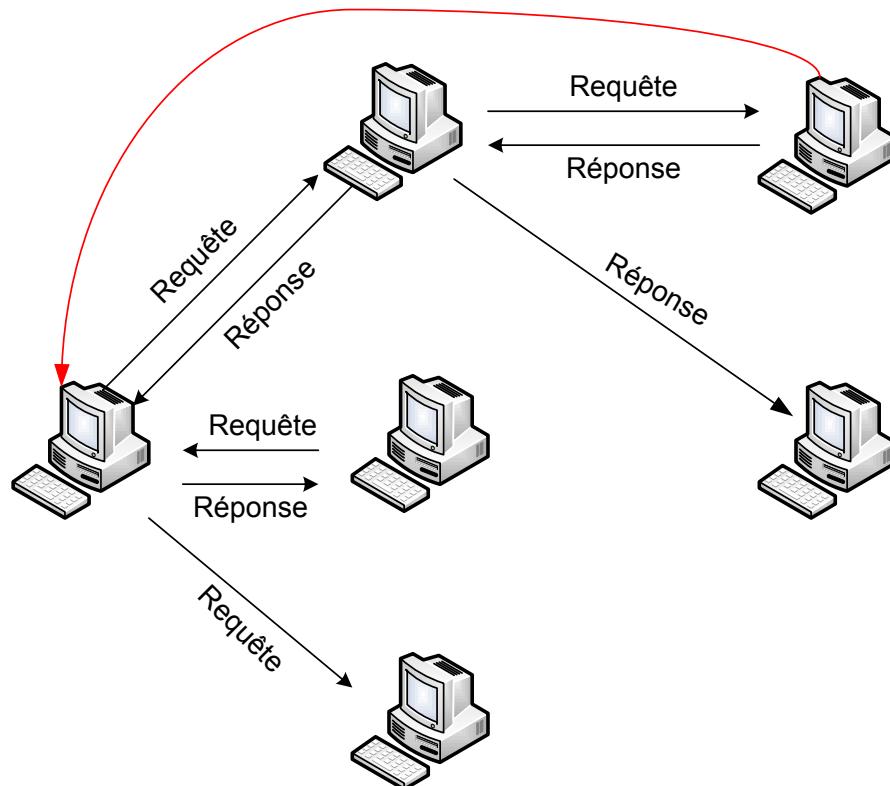
P2P centralisé : problèmes

- Point central : pas de tolérance aux pannes
 - Congestion autour du serveur
- ⇒ Le transfert de fichier est décentralisé, mais la localisation du contenu est centralisée

P2P décentralisé : Gnutella

- Complètement distribué
 - Pas de serveur central
- Protocole public
- Le réseau overlay: graphe
 - L'existence d'une connexion TCP entre un peer X et Y est représentée par un arc dans le réseau overlay
 - Le réseau overlay : les arcs et les peers
 - Un arc n'est pas un lien physique
 - En moyenne, chaque peer a moins de 10 voisins

Le protocole Gnutella



- Requêtes envoyées via les connexions TCP
- Chaque peer transfert la requête
- La réponse (@IP) est renvoyé sur le même chemin que la requête

Recherche du contenu ?

- Organisation en base de données (BD) : nom du contenu, adresse IP du détenteur de ce contenu
- Difficile à mettre en place dans un environnement décentralisé (p2p)
- Solution : distribuer la BD sur l'ensemble des peers
 - Chaque peer contiendra une partie de la BD
 - Chaque peer interroge la BD avec une paire de clé particulière

Exemple de recherche sur la BD distribuée

- Toto recherche la dernière distribution Linux, il interroge la BD distribuée.
- La DB sait que Titi est responsable de la clé « Linux ».
- Alors la DB demande à Titi d'envoyer l'adresse des détenteurs du contenu « Linux »

Solution DB distribué

- Solution naïve : répartir aléatoirement les clés sur les peers.
 - Chaque peer interroge l'ensemble des peers pour connaître les détenteurs du contenu.
 - Solution très couteuse.
-
- ✓ Distributed Hash Table (DHT)

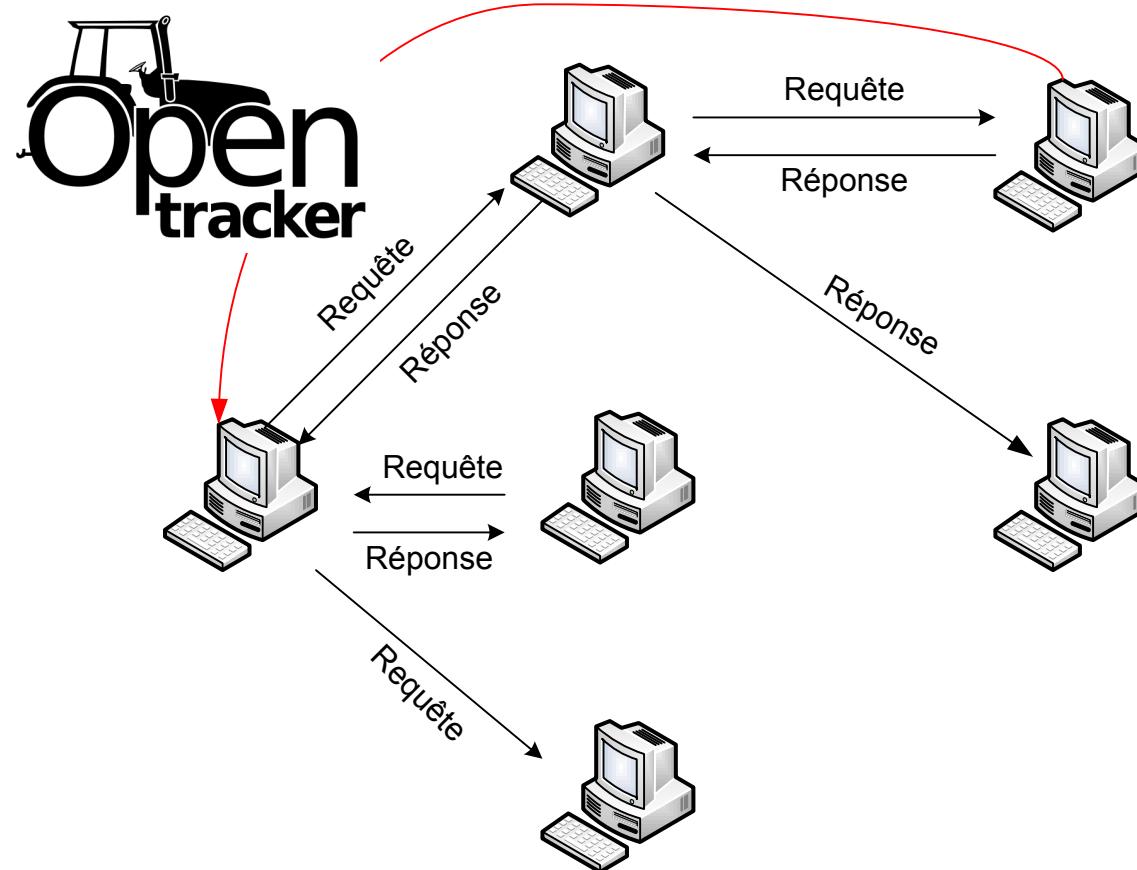
Distributed Hash Table (DHT)

- Affecter un identifiant à chaque peer, entre 0 et 2^{n-1}
- Utiliser une fonction spéciale qui associe un nom de contenu avec une valeur
 - Fonction de hashage : ex. MD5, SHA
- Associer le contenu avec l'identifiant le plus proche.
- Ex.
 - On suppose que $n=4$, donc les identifiants ainsi que les valeurs (clés) sont dans l'intervalle [0,15].
 - On suppose que 7 peers sont dans le réseau ayant comme identifiant 1,3,5,7,8,10,13 et on cherche le contenu le contenu (12)
 - Le peer le plus proche est 13.

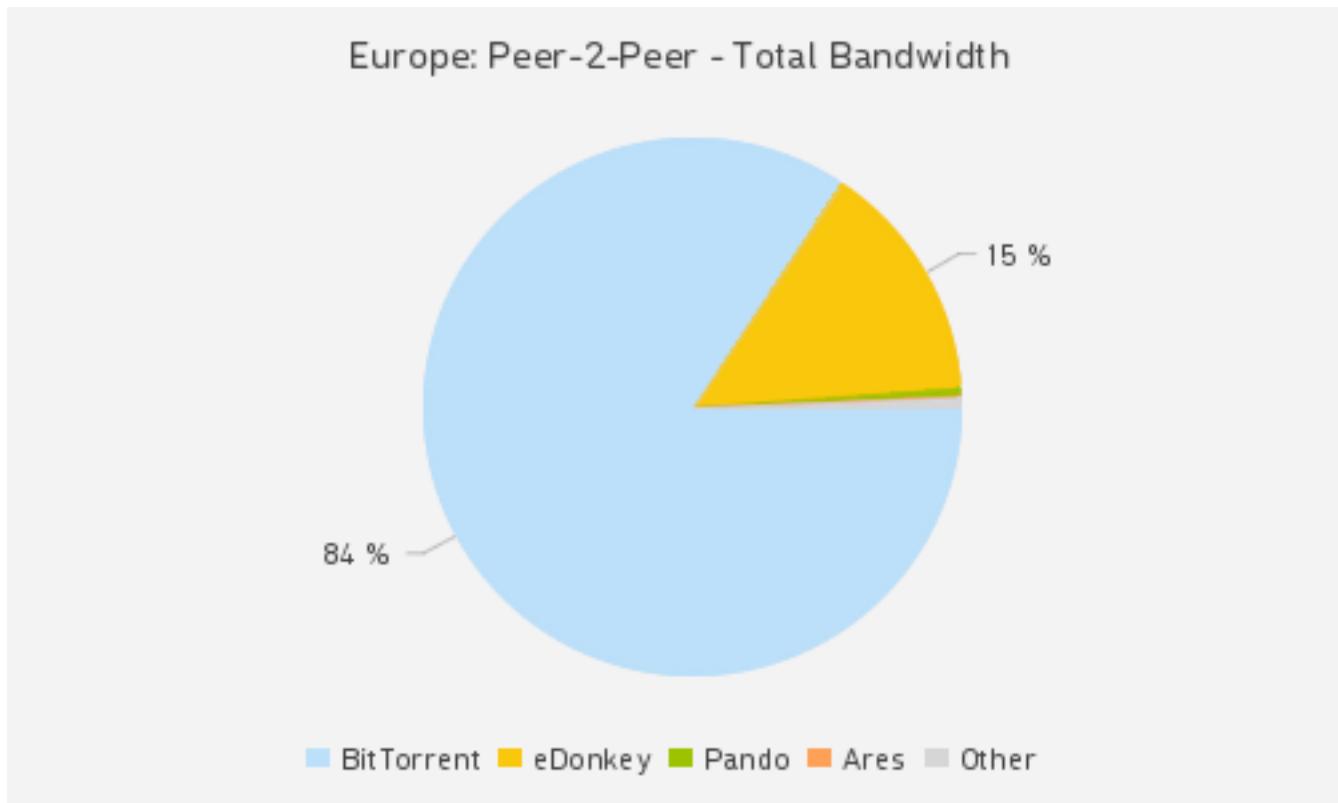
P2P : BitTorrent

- Protocole et client (représente 30% du trafic du cœur du réseau)
- Pas de recherche de contenu
 - Un fichier .torrent, contient un ensemble d'informations (tracker) pour le téléchargement du contenu
 - Obtenu à partir d'un serveur web
- Un torrent => une session de transfert d'un seul contenu entre peers
- Deux types de peers
 - Seeds :
 - proposent le téléchargement du contenu en entier
 - restent connectés
 - Leechs :
 - proposent le téléchargement d'un chunk (morceau)
 - téléchargent un chunk qui n'ont pas encore
 - volatiles

BitTorrent



P2P stat.

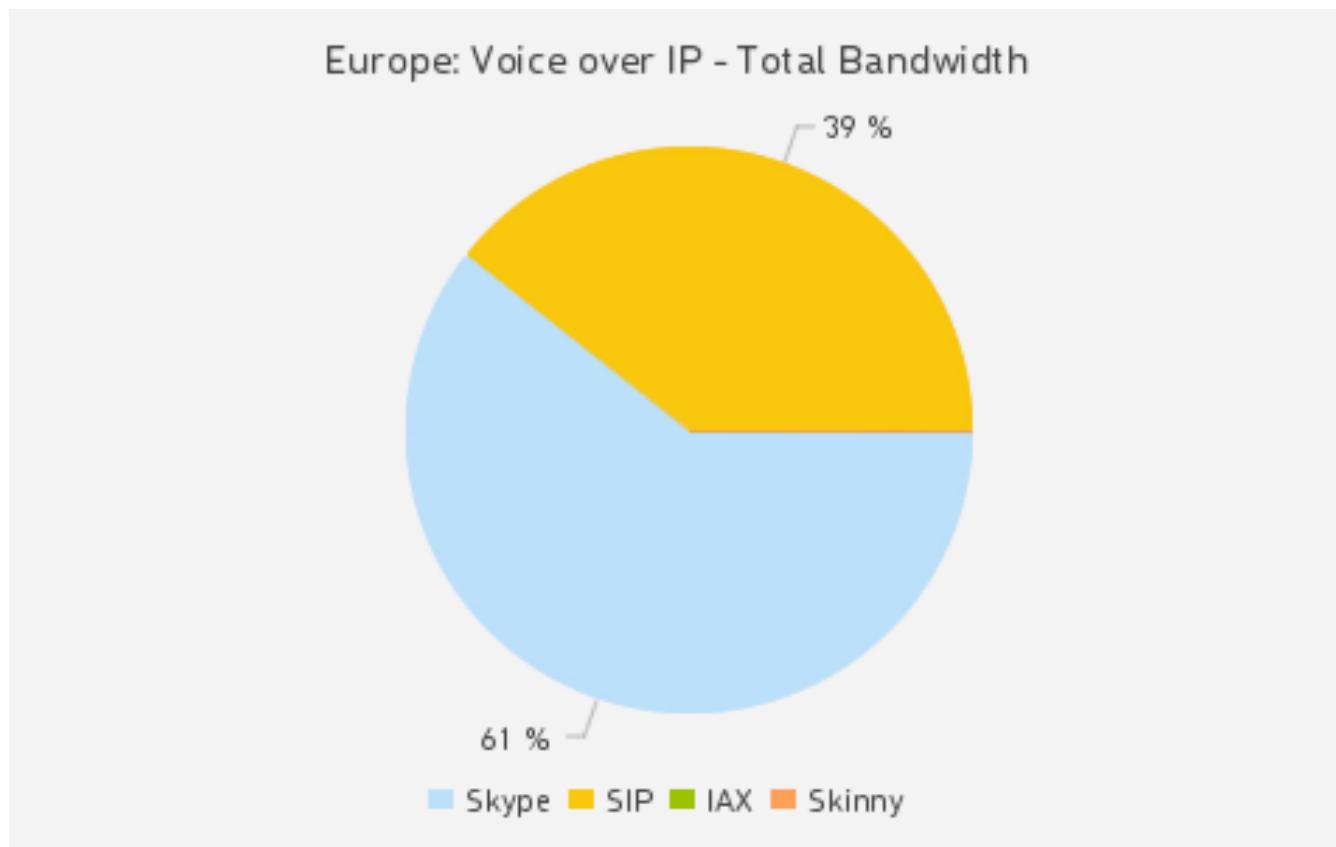


Source : <http://www.internetobservatory.net/europe>

Skype VoIP basé sur P2P

- Crée par le concepteur de KaZaa
- Racheté par eBay à 2,6 milliards de dollars et récemment par Microsoft
- Protocole propriétaire, tous les messages sont chiffrés
- Notion de superNode (super nœud)
 - Elu à cause de leur capacité
 - Permet de traverser le NAT
 - Maintient la liste des connectés

Skype vs VoIP



Source : <http://www.internetobservatory.net/europe>

Couche application : sommaire

- Concernant les protocoles
- Echange Requête/Réponse
 - Le client demande un service ou des informations
 - Le serveur répond avec les données, un statut (code)
- Le format des messages :
 - En-tête : des champs contenant des informations sur les données
 - Données
- Contrôle et les messages de données
 - En bande ou hors bande
- Transfert fiable versus transfert non fiable