T2
Protocols de les aplicacions tradicionals de la xarxa

Xarxes de Computadors i Aplicacions

PAU ARTIGAS, DAVID CARRERA i JORDI TORRES
Departament d'Arquitectura de Computadors
UPC, setembre - 2009

Contingut

Intercanvi de documents
- HTTP, FTP

Correu electrònic
- SMTP, mime, POP3, IMAP

Servei de directori a internet
- DNS

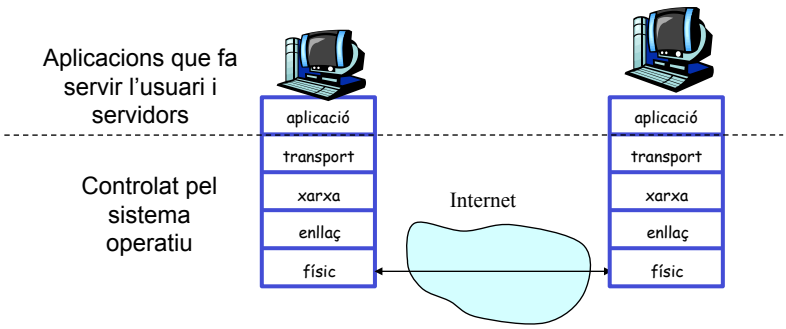
Algunes d'aquestes transparències són una adaptació de les transp. del llibre Computer Networking: A Top Down Approach Featuring the Internet, 2nd edition.

Jim Kurose, Keith Ross

Addison-Wesley, July 2002.

Introducció: Protocols Nivell Aplicació

- Són els usats per les aplicacions dels usuaris i pels servidors
  - Usen els serveis oferts pel sistema operatiu
    - a través de sockets



Introducció: Protocols Nivell Aplicació

Intercanvi de documents
- HTTP, FTP, "p2p", NFS

Correu electrònic
- SMTP, POP3, IMAP

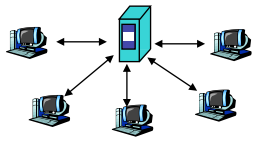
Servei de directori
- DNS, LDAP

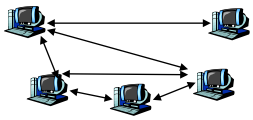
Execució remota
- RDP (Terminal Services), SSH, VNC, telnet, X-Windows

Comunicació
- RTP, VoIP (veu sobre IP), Missatgeria Instantània, IRC

Introducció: Models comunicació de les aplicacions distribuïdes

- Una arquitectura client/servidor és aquella en què una màquina o procés té unes dades o funcionalitats i un conjunt de clients fan peticions per accedir a aquestes dades o funcionalitats
  - Els servidors generalment, però no sempre, estan a màquines potents i dedicades exclusivament per aquests serveis
    - Són passius en quant esperen peticions
  - Els clients per altra banda estan als equips dels usuaris
    - Són actius en quant envien peticions
- Un altre model de comunicació és l'entre parells (peer-to-peer), en que cada màquina o procés és alhora un client i un servidor i tothom té les mateixes responsabilitats i les dades i funcionalitats estan repartides

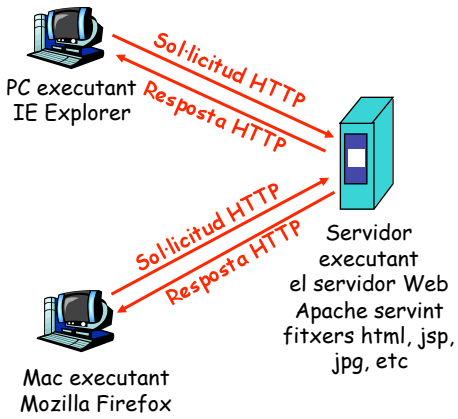




HTTP: Introducció

HTTP: protocol de transferència d'hipertext

- És el protocol de la capa d'aplicació per la Web
- Utilitza el model client/servidor
  - *client:* navegador a la màquina d'un usuari que sol·licita, rep i mostra els objectes Web
  - *servidor:* Servidor www que té els continguts i respon a les sol·licituds del client
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068



HTTP: Introducció

Alguns Termes

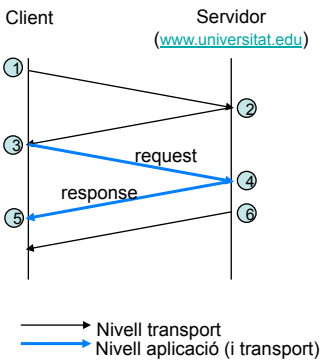
- Una pàgina Web típicament consta d'un arxiu HTML base que inclou diversos objectes referenciats
  - Una pàgina web consta d'objectes
- Els objectes poden ser un arxiu HTML, una imatge JPEG, un applet Java, un arxiu de flash, ...
- Cada objecte es direcciona amb una URL
  - Exemple d'una URL:

`http://www.ac.upc.es/XCA/logo.gif`

Protocol — Nom del host — Path

HTTP: Funcionament bàsic

Suposem que algú escriu al navegador la URL
http://www.universitat.edu/facultat/index.html *(conté text i referència a 10 imatges jpeg)*



1. El client HTTP inicia la connexió TCP al servidor HTTP
2. El servidor HTTP al host www.universitat.edu espera connexions TCP. Quan "accepta" una conexió, notifica al client
3. El client HTTP envia un *request message* (que conté el URL) on indica que desitja l'objecte facultat/index.html
4. El servidor HTTP rep el missatge de sol·licitud, i li respon amb un *response message*
5. El client HTTP rep el missatge de resposta que conté l'arxiu html i el mostra.
6. El servidor HTTP tanca la connexió TCP.

Al recórrer l'arxiu html troba 10 objectes jpeg referenciats, aleshores els passos 1 a 6 es repeteixen per cada un dels 10 objectes jpeg.

HTTP: Persistència

HTTP no persistent

Un sol objecte és enviat per cada connexió TCP
- El SO ha de treballar i assignar els recursos del host per cada connexió TCP
- HTTP/1.0 utilitza HTTP no persistent

HTTP persistent

Múltiples objectes poden ser enviats sobre una mateixa connexió TCP
- El servidor deixa la connexió oberta després d'enviar el missatge de resposta i els missatges que posteriorment s'envien entre el client i el servidor són enviats sobre aquesta mateixa connexió
- HTTP/1.1, per omissió, usa connexions persistents

Una optimització que es fa servir (en ambdós casos) és que el navegador obre varies connexions paral·leles per descarregar els objectes referenciats

HTTP: Persistència amb pipelining

Persistència sense pipelining:

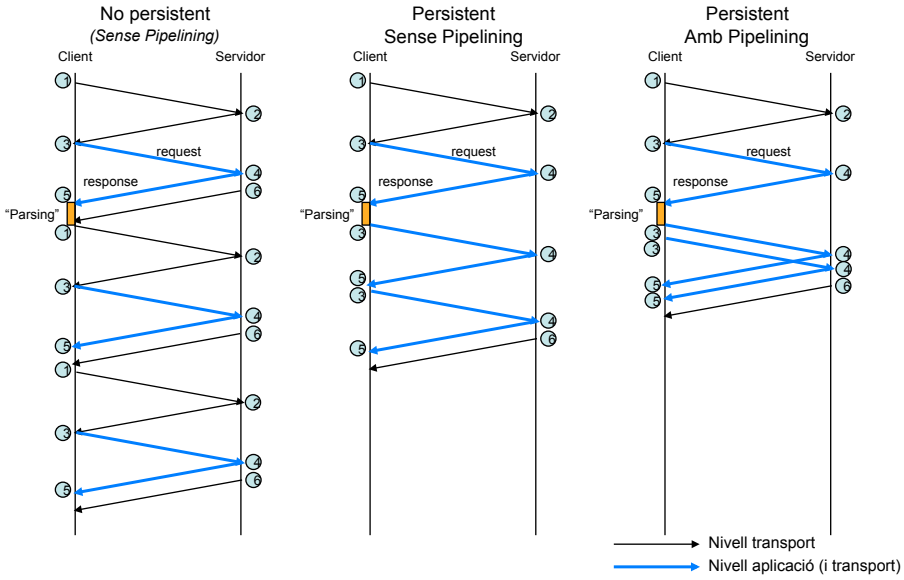
- El client emet una nova sol·licitud només quan la resposta anterior ha estat rebuda
  - *Es requereix aproximadament un RTT per cada objecte referenciat*

Persistència amb pipelining:

- El client envia una sol·licitud tan aviat com troba un objecte referenciat
  - *Es solapen els RTT dels objectes referenciats*
  - Per omissió a HTTP/1.1

HTTP: Exemple

Descàrrega d'un html que referencia dues imatges.



HTTP: Missatge sol·licitud

- HTTP té dos tipus de missatges: ***request*, *response***
- Missatge de sol·licitud (request):
  - ASCII (format llegible per humans)

Línia de sol·licitud (comandes GET, POST, HEAD)

Línies de capçalera

"Carriage return, line feed" Indica el final

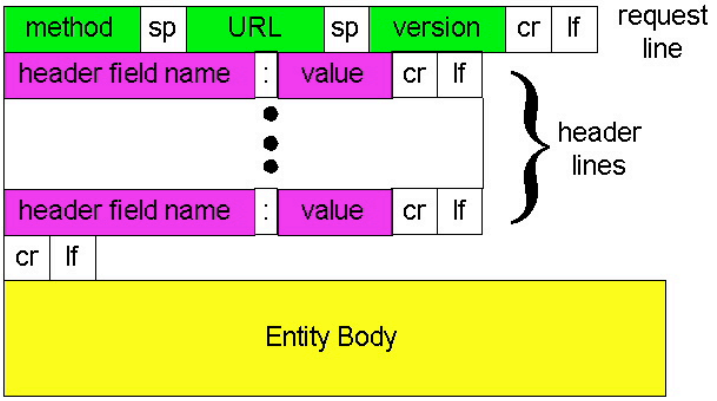
```
GET /algundir/pagina.html HTTP/1.1
Host: www.algunsitio.edu
User-agent: Mozilla/4.0
Connection: close
Accept-language:fr
```

("carriage return, line feed" addicional)

CR/LF

HTTP: Missatge sol·licitud

Missatge de sol·licitud HTTP: format general



HTTP: Caching

- Que passa quan visitem una web que ja havíem visitat i no ha canviat?
  - No caldria descarregar-se altra vegada el contingut si ens l'hem guardat el primer cop que hi hem accedit
- Una *cache* (cau) és una col·lecció de dades duplicades a partir d'uns originals, de forma que accessos futurs a aquestes dades poden ser fets sobre les copies locals amb un cost inferior a l'accés a l'original
  - Però cal controlar que poden haver estat modificades!
    - Comprovar-ho
    - "Temps de vida"
    - ...

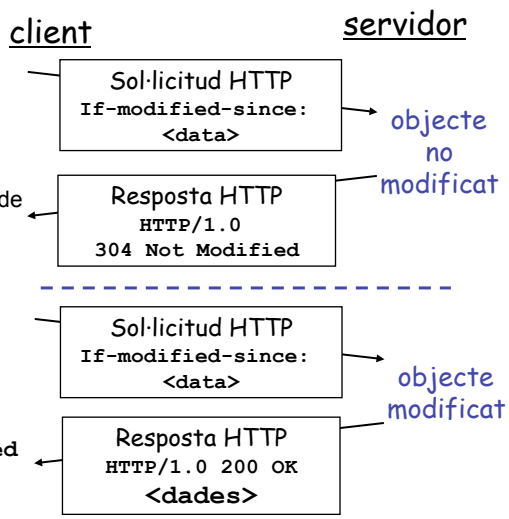
HTTP: GET condicional

(caching del costat del client)

- **Objectiu:** no enviar objectes si el client té una versió actualitzada en cache
- **client**: El client especifica la data de la copia en cache al missatge de sol·licitud HTTP

  `If-modified-since: <fecha>`
- **servidor**: La resposta no porta l'objecte si la copia en cache està actualitzada:

  `HTTP/1.0 304 Not Modified`



HTTP: Enviament de dades

- Les pàgines web sovint inclouen formularis per a introduir dades o poder "pujar arxius"

Usant el método GET:

- Los dades són enviades en el camp del URL de la línia de sol·licitud

  www.algunlloc.com/cerca?nom=arcesio&cognom=net
  - Per tant no es poden enviar dades binàries!!!

Usant el mètode POST:

- Los dades són enviades al servidor a través del cos del missatge (Entity Body)

HTTP: Mètodes

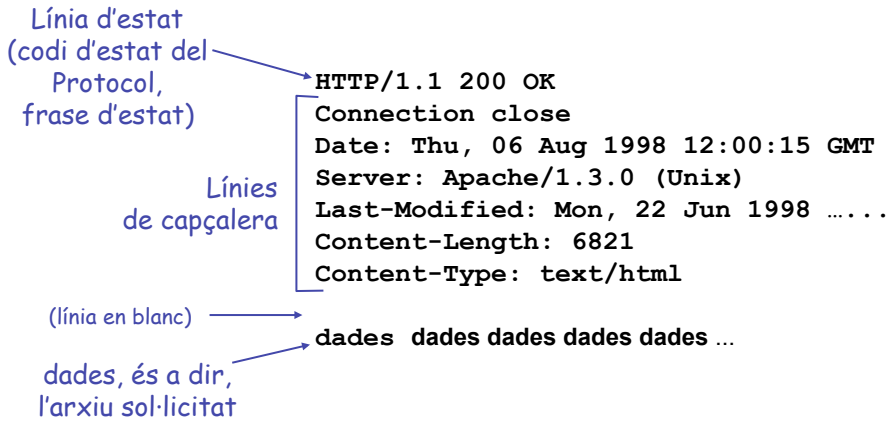
HTTP/1.0

- GET
- POST
- HEAD
  - Fa una consulta al servidor sobre les característiques de l'objecte però no el transfereix

HTTP/1.1

- GET, POST, HEAD
- PUT
  - Envia un arxiu en el cos del missatge al path especificat a la URL
- DELETE
  - Borra l'arxiu especificat a la URL
- I altres ...

HTTP: Missatge de resposta



HTTP: Codis d'estat

S'usen a la primera línia del missatge de resposta del servidor → client.

Alguns Exemples:

2** - Èxit

200 OK
- Sol·licitud exitosa, l'objecte sol·licitat va en aquest missatge

3** - Redirecció

301 Moved Permanently
- L'objecte sol·licitat ha estat mogut, la nova ubicació s'especifica posteriorment en aquest missatge (Location:)

4** - Error del client

400 Bad Request
- El missatge de sol·licitud no ha estat entès pel servidor

404 Not Found
- El document sol·licitat no s'ha trobat en aquest servidor

5** - Error del servidor

505 HTTP Version Not Supported
- Versió d'HTTP no suportada

HTTP: Connexió sense navegador

El `telnet` és un programa senzill que es connecta al servidor i port indicats i aleshores:
- Envia tot el que escrivim
- Ens mostra tot el que el servidor ens envia

1. Connectar-se, usant el telnet, al port 80 d'algun servidor web

```
telnet www.fib.upc.edu 80
```

Obre una connexió TCP al port 80 (port "ben conegut" de HTTP) a www.fib.upc.edu.

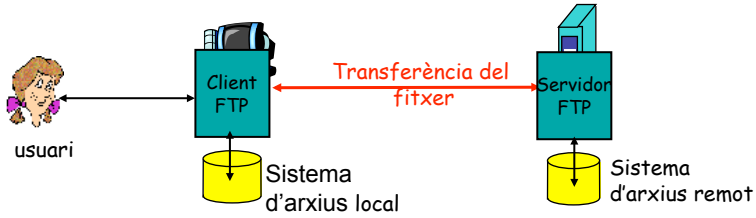
2. Escriure una sol·licitud de HTTP amb el mètode GET:

```
GET /index.html HTTP/1.0
```

A l'escriure això (i prémer <ENTER> dos vegades), s'enviarà aquesta sol·licitud HTTP mínima (però completa) al servidor HTTP

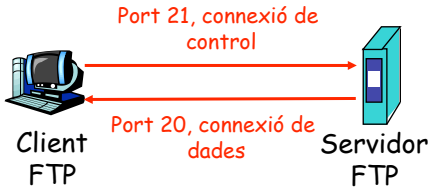
3. ¡Observa el missatge de resposta enviat pel servidor HTTP!

FTP: Protocol de transferència d'arxius



- Transfereix arxius cap a i des de el host remot
- Usa el model client/servidor
  - *client:* qui inicia la sessió (per transferir cap a/des de el host remot)
  - *servidor:* host remot
- ftp: RFC 959, RFC1123
- Servidor ftp: ports 21 y 20 (per defecte)

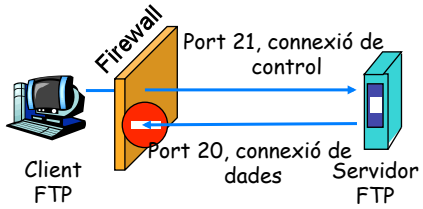
FTP: Connexions



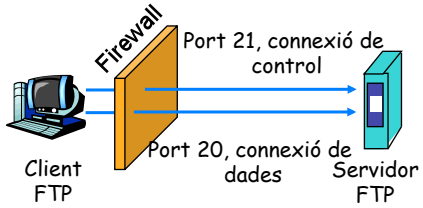
- El client FTP contacta el servidor FTP en el port 21, usant TCP com a protocol de transport
- El client envia comandes sobre la connexió de control.
- Quan el servidor rep una comanda per transferir un arxiu, el servidor obre una connexió TCP per dades amb el client
  - Després de transferir l'arxiu, el servidor tanca la connexió.
  - El servidor obre una segona connexió de dades per a transferir un altre arxiu.
- El servidor FTP manté **informació "d'estat":** directori actual, la autenticació inicial, etcètera

FTP: Mode passiu

- Si el client té un firewall, l'ftp no funciona!



- Per això existeix l'ftp en mode "passiu", en que és el client el que obre ambdues connexions
  - Es fa servir la comanda PASV



FTP: Comandes

Algunes comandes:

- Enviats com text ASCII sobre el canal de control
- **`USER`** ***`username`***
- **`PASS`** ***`password`***
- **`LIST`** retorna una llista dels arxius en el directori actual
- **`RETR`** ***`filename`*** recupera (porta) l'arxiu
- **`STOR`** ***`filename`*** desa (col·loca) l'arxiu en el host remot
- …

FTP: Respostes

- Usa un codi d'estat i una frase (com a HTTP)
  - El codi és per l'ús dels programes i el missatge per l'ús de les persones (pot variar)

Exemple de codis de retorn

- `331 Username OK, password required`
- `125 data connection already open; transfer starting`
- `425 Can't open data connection`
- `452 Error writing file`
- …

Correu Electrònic

Dos components principals:

- Agents d'usuari
- Servidors de correu

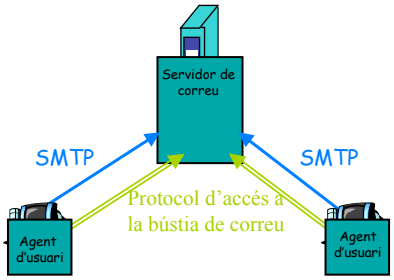
Dos protocols principals:

- Protocol simple de transferència de correu (SMTP)
- Protocol d'accés a la bústia de correu

Correu Electrònic: Introducció

Agent d'usuari

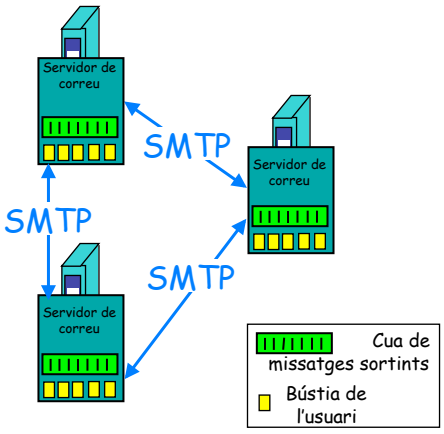
- Conegut com "lector de correu"
  - Exemples: Eudora, Outlook, elm, Thunderbird
- Recupera els missatges col·locats al servidor
  - Usant el protocol d'accés a la bústia de correu
- Permet enviar missatges
  - Usant SMTP



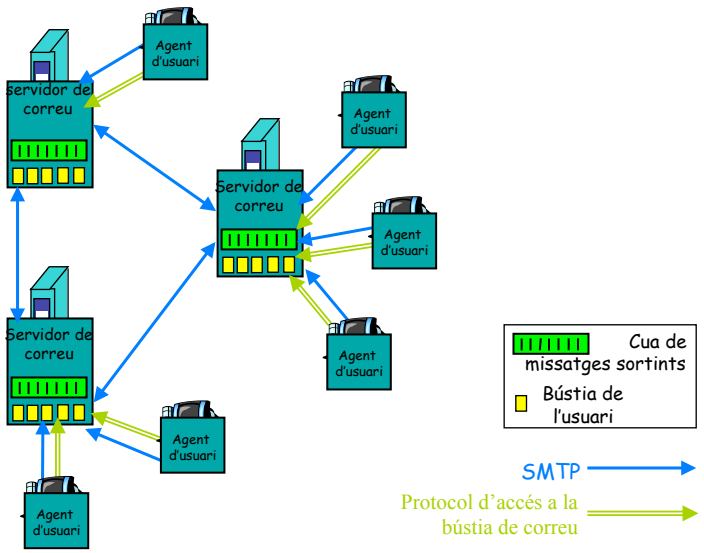
Correu Electrònic: Introducció

Servidors de correu

- La bústia conté els missatges que han arribat per l'usuari
- Cua de missatges de correu sortints (per a ser enviats)
- Protocol SMTP usat entre els servidors de correu per enviar els missatges
  - Es comporta com un client SMTP quan envia correu a un altre servidor de correu
  - Es comporta com "servidor" quan rep correu d'un altre equip



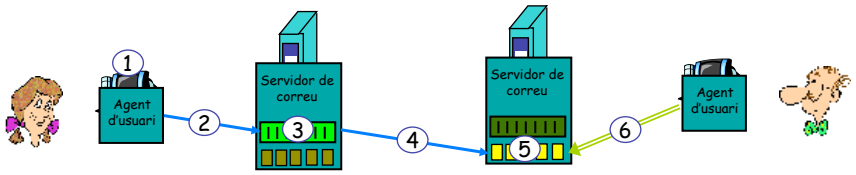
Correu Electrònic: Introducció



Correu Electrònic: Exemple

Escenari: Alicia envia un missatge a Beto

1) Alicia usa el seu agent d'usuari per elaborar un missatge per a `beto@algunsitio.ar`
2) L'agent d'usuari d'Alicia envia el missatge al seu servidor de correu sortint; el missatge és col·locat a la cua de missatges sortints del servidor
3) El costat "client" del servidor SMTP obre una connexió TCP amb el servidor de correu de Beto
4) El costat "client" del servidor SMTP envia el missatge d'Alicia sobre la connexió TCP
5) El servidor de correu de Beto col·loca el missatge a la bústia de Beto
6) Beto invoca el seu agent d'usuari per llegir els missatges usant el protocol d'accés a la bústia de correu



Correu Electrònic: SMTP [RFC 2821]

- Utilitza TCP per a transferir missatges de correu des del client al servidor, utilitza el port 25
  - Utilitza connexions persistents (com l'HTTP persistent)
- Transferència directa: entre el servidor que envia i el servidor que rep
- La transferència té tres fases
  - Obertura (handshaking)
  - Transferència dels/del missatge/s
  - Tancament
- Interacció comanda/resposta
  - comandes: text ASCII
  - resposta: codis d'estat i frase
- Els missatges han d'estar en ASCII de 7 bits

Correu Electrònic: Exemple SMTP

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alicia@crepes.fr>
S: 250 alicia@crepes.fr... Sender ok
C: RCPT TO: <beto@hamburger.edu>
S: 250 beto@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself

            MISSATGE

C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Correu Electrònic: Format del missatge

(Recordem que SMTP és el protocol per l'intercanvi de missatges)

- RFC 822: estàndard per el format de missatges de text:
  - Línies de *header* (capçaleres), és a dir,
    - `To:`, `From:`, `Subject:`
    - *¡Són diferents a les dades de les comandes SMTP!*
  - Cos (*body*)
    - És el "missatge", només permet caràcters ASCII

```
…
C: From: alicia@crepes.fr
C: To: beto@hamburger.edu
C: Subject: Pregunta.
C: MIME-Version: 1.0
C: Content-Type: text/plain
C:
C: ¿Te gusta la salsa de tomate?
C:   ¿y los pepinillos?
C:
C: Saludos
…
```

Capçaleres

(línia en blanc)

Cos

Correu Electrònic: MIME

- MIME: Multimedia Internet Mail Extension, RFC 2045, 2056
  - Extensions per Multimedia
- Línies addicionals a la capçalera del missatge declaren contingut de tipus MIME

Versió de MIME

Mètode usat para codificar dades

Tipus de dades multimedia, subtipus, paràmetre de declaració

Dades codificades

```
From: alicia@crepes.fr
To: beto@hamburger.edu
Subject: Imagen de un crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.........................
......base64 encoded data
```

Correu Electrònic: Tipus MIME

`Content-Type: tipus/subtipus; paràmetres`

Text
- Exemple de subtipus: **`plain, html`**

Image
- Exemple de subtipus: **`jpeg, gif`**

Audio
- Exemple de subtipus: **`basic`** (codificació 8-bit mu-law), **`32kadpcm`** (codificació 32 kbps)

Video
- Exemple de subtipus: **`mpeg, quicktime`**

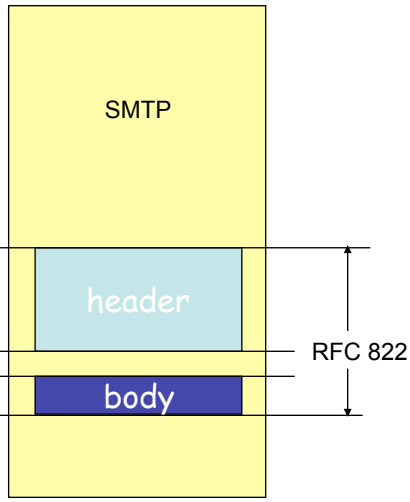
Application
- Dades que han de ser processades pel client abans de "poder-se veure"
- Exemple de subtipus: **`msword, octet-stream`**

Correu Electrònic: Tipus Multipart

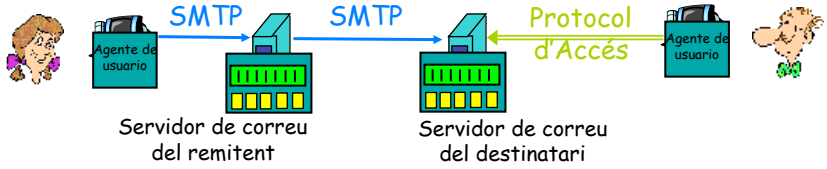
```
From: alicia@crepes.fr
To: beto@hamburger.edu
Subject: Imagen de un crepe.
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=StartOfNextPart

--StartOfNextPart
Content-Type: text/plain
Hola Beto, por favor encuentra la imagen de un crepe.
--StartOfNextPart
Content-Transfer-Encoding: base64
Content-Type: image/jpeg
base64 encoded data .....
.........................
......base64 encoded data
--StartOfNextPart
Content-Type: text/plain
¿te gustaría tener la receta?
```

Correu Electrònic: Visió global

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to
    meet you
C: MAIL FROM: <alicia@crepes.fr>
S: 250 alicia@crepes.fr... Sender ok
C: RCPT TO: <beto@hamburger.edu>
S: 250 beto@hamburger.edu ...
    Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a
    line by itself
C: From: alicia@crepes.fr
C: To: beto@hamburger.edu
C: Subject: Pregunta.
C: MIME-Version: 1.0
C: Content-Type: text/plain
C:
C: ¿Te gusta la salsa de tomate?
C:   ¿y los pepinillos?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing
    connection
```



Correu Electrònic: Protocols d'accés a la bústia de correu



- Protocol d'accés a la bústia de correu: recupera els missatges del servidor
  - POP: Post Office Protocol [RFC 1939]
    - autorització (agent <-->servidor) i descàrrega dels missatges
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - Més característiques (més complex)
    - manipulació dels missatges emmagatzemats al servidor
  - HTTP: Hotmail , Yahoo! Mail, etc.

Correu Electrònic: POP3 i IMAP

POP3	IMAP
• 2 Mètodes de funcionament – "Descarrega i borra": un cop llegeix els missatges els elimina del servidor • No es pot tornar a llegir un missatge si es canvia de client – "Descarrega i guarda": no els elimina al llegir-los • Permet tenir copies dels missatges en diferents clients	• Manté tots els missatges al mateix lloc: el servidor. • Permet a l'usuari que organitzi els seus missatges en carpetes
• POP3 no manté informació de sessions anteriors	• IMAP manté informació d'estat de sessions anteriors: –Noms de carpetes i mapeig entre la identificació dels missatges i el nom de les carpetes

Correu Electrònic: Exemple POP3

Fase d'autorització

- Comandes del client:
  - **`user:`** nom de l'usuari
  - **`pass:`** la clau
- Respostes del servidor
  - **`+OK`**
  - **`-ERR`**

Fase de transacció

- Comandes del client:
  - **`list:`** llista els números dels missatges
  - **`retr:`** recupera el missatge pel número
  - **`dele:`** borra el missatge
  - **`quit:`** **`acaba`** la sessió

```
S: +OK POP3 server ready
C: user beto
S: +OK
C: pass goloso
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

DNS: Domain Name System

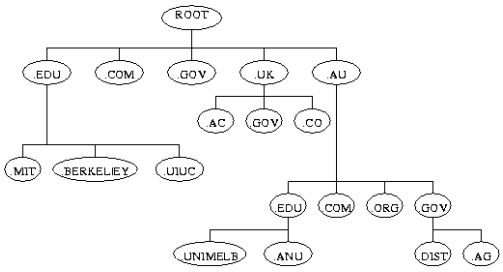
Les persones: tenen molts identificadors:
- DNI, nom, passaport

Identificadors de host i routers de Internet:
- El "nom", per exemple, gaia.cs.umass.edu – utilitzat pels humans
- La direcció IP (32 bits)

Pregunta: ¿qui associa les direccions IP i els noms?

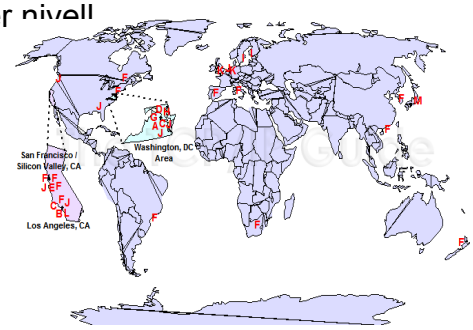
DNS: Introducció

- ***Protocol de la capa d'aplicació*** per *resoldre* noms (traducció entre direcció i nom)
  - nota: funció central de Internet, implementada com un protocol de la capa d'aplicació
  - RFCs 1034 I 1035
- ***Base de dades distribuïda*** implementada com una jerarquia de molts *servidors de noms*
  - ¿Perquè no un DNS centralitzat?



DNS: Servidors arrel (root servers)

- Arrel de la jerarquia DNS
  - N'hi ha varis per fiabilitat i rendiment
  - Coneixen els servidors DNS de tots els dominis de primer nivell



- Dominis de primer nivell
  - Genèrics: .com, .edu, .gov, .net, and .org, ...
  - Per país: .es, .uk, .fr, ...

DNS: Servidors de noms

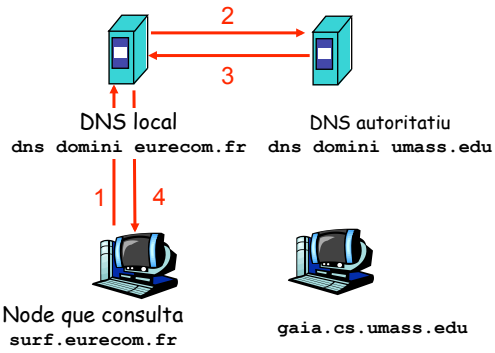
Servidors de noms locals:
- Servidor local a cada xarxa d'accés a Internet
- Encarregat de rebre peticions dels equips de la xarxa a la que es troba connectat (p.ex. un ISP)

Servidor de noms autoritatiu:
- Servidor que conté la informació primària d'una zona de domini
- La seva informació és la "bona"

DNS: Exemple simplificat

`surf.eurecom.fr` desitja saber la direcció IP de **`gaia.cs.umass.edu`**

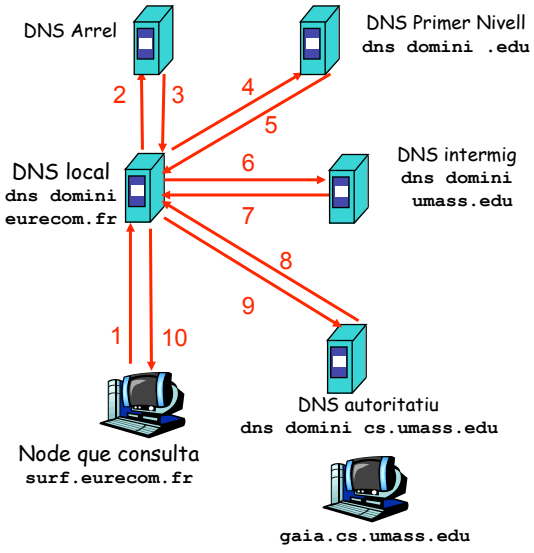
1. El client contacta el seu servidor DNS local (**`dns.eurecom.fr`**)
2. El servidor local contacta el servidor de noms autoritatiu (**`dns.umass.edu`**) …
3. … aquest li respon
4. Aleshores envia la resposta al client



DNS: Exemple

DNS arrel:

- Podria no conèixer el servidor de noms autoritatiu
- Podria conèixer el *DNS intermedi:* el que es contacta per trobar el DNS autoritatiu



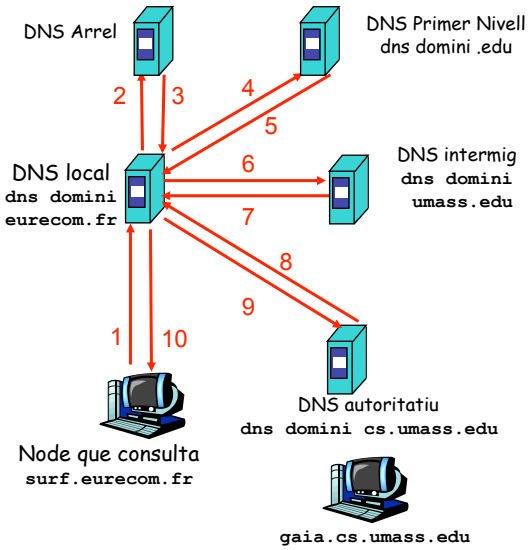
DNS: Tipus de consultes

Consulta recursiva:

- Espera que el servidor respongui amb l'adreça que respon a la consulta o que retorni un codi d'error
  - Generalment només el servidor DNS local ho permet

Consulta iterada:

- El servidor contesta la pregunta si la coneix directament, o indica l'adreça d'un servidor que ens donarà més informació



DNS: Registres

DNS: és una Base de Dades distribuïda que guarda registres de recursos (RR)

Format RR: `(name, value, type, ttl)`

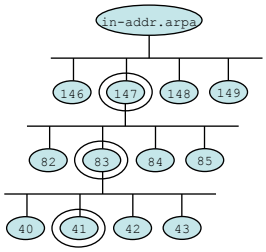
- **Type=A**
  - **`name`** es un nom de host
  - **`value`** es una direcció IP
- **Type=MX**
  - **`name`** es un nom de domini
  - **`value`** es el nom d'un servidor de correu associat amb **`name`**
- **Type=NS**
  - **`name`** es un domini (per exemple. indret.com)
  - **`value`** es una direcció IP de un DNS autoritatiu per aquest domini
- **Type=CNAME**
  - **`name`** és un àlies para algun nom "canònic" (el nom real)
  - **`value`** és el nom canònic
    `www.yahoo.com` és realmente `www.yahoo.akadns.net`

DNS: Registres

- **Resolució inversa**
  - **Type=PTR**
    - **`name`** es un domini de l'estil d.c.b.a.in-addr.arpa
    - **`value`** és el nom DNS de la ip a.b.c.d

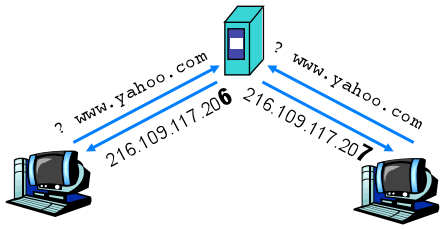
    - exemple de consulta usant la comanda "nslookup" o "dig"
      `nslookup -type=PTR 7.41.83.147.in-addr.arpa`
      `dig 7.41.83.147.in-addr.arpa PTR` (`+trace` si es vol fer iteratiu i es vol veure la jerarquia que es segueix)

  - Es tracta d'una jerarquia diferent
    - Si bé es poden reaprofitar alguns dels servidors



DNS: Round Robin

- Permet fer balanceig de càrrega distribuint les peticions entre diferents respostes vàlides

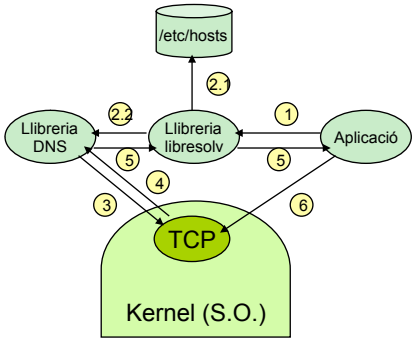


- Aplicable sobretot a RR de tipus:
  - A
  - CNAME

DNS: Caching i actualització de registres

- Quan el DNS aprèn el mapeig, fa una copia en *cache*
  - Les dades col·locades en la *cache* tenen un temps de vigència (fixat en el registre guardat pel servidor autoritatiu), al passar aquest temps les dades es treuen de la *cache*
- Els registres habitualment els actualitza l'administrador del servidor
- Com funciona el DNS dinàmic?

DNS: com ho fan servir les aplicacions



1. L'aplicació es vol connectar a www.google.com, però per a poder-ho fer necessita saber la IP, per això crida la funció `gethostbyname()`
2. La funció `gethostbyname()` fa, per ordre:
   1. consulta al fitxer de hosts local
   2. crida a la llibreria de DNS
3. La llibreria de DNS fa una petició al servidor DNS local
4. El servidor DNS local respon
   - Fent les peticions que siguin necessàries
5. La llibreria dóna el resultat a l'aplicació
6. L'aplicació usa la IP resultant per a connectar-se

DNS: Comanda “nslookup”/”dig”

- Permeten fer **consultes** a la base de dades DNS
  - Ens donen més informació que la resposta
    - Per exemple: qui ens ha respost
- Exemples

Preguntar per l’adreça (el registre de tipus A) de www.fib.upc.edu

```
dig www.fib.upc.edu
nslookup www.fib.upc.edu
```

Preguntar pels servidors de correu (registre de tipus MX) del domini fib.upc.edu

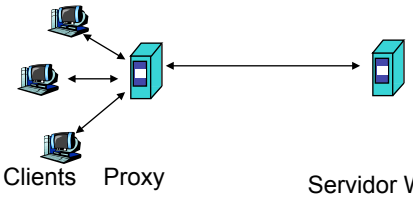
```
dig fib.upc.edu MX
nslookup -type=MX www.fib.upc.edu
```

DNS: Gestió en l’assignació de Dominis

- ICANN s’encarrega d’assignar @IP o blocs de @IP
  - Ara és privat (encara que es tracta d’una organització sense ànim de lucre) que va recollir les activitats de la IANA. S’encarrega de donar tant dominis genèrics com @IP
- Els **dominis genèrics** són registrats per companyies a les que ICANN dóna el dret a que actuïn com a tals sota certes restriccions (Accredited Registrars)
  - Per exemple Nominalia (http://www.nominalia.com)
- El domini territorial **.es** l’assigna el Ministeri de Foment: servei ES-NIC (http://www.nic.es)

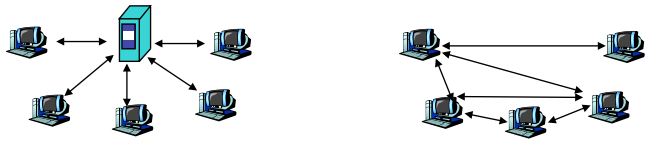
Servidors Proxy

- Un servidor proxy és un servei que accepta peticions dels clients i les realitza en nom seu
  - Són específics de cada aplicació
  - Un cas típic són els proxy-cache web, en que un cop reben una petició web guarden una copia en cache de la resposta i la següent petició la responen a partir de la copia en cache, estalviant temps i ample de banda



Peer-to-peer

- Les aplicacions entre parells segueixen un model diferent al client-servidor
  - Client-servidor: el contingut està ubicat en un “gran” servidor i els clients hi accedeixen
  - Entre parells: tothom és alhora client i servidor
    - Tothom aporta i tothom rep



Peer-to-peer

- Presenten nous reptes:
  - Gran quantitat de màquines presents
  - Si les dades estan repartides entre tots com trobar-les?
    - A vegades es fan servir servidors per a les cerques
  - Es connecten i es desconnecten amb facilitat (no com un servidor)
- Obren noves possibilitats
  - Ús dels recursos als “extrems de la xarxa”
    - Disc, ample de banda i temps de processament
- Funcionalitats
  - Compartició de fitxers
  - Transferència de video i audio *multicast*
  - Proxy-cache web
  - ...