

Servicios de Internet (y2)

Area de Ingeniería Telemática
<http://www.tlm.unavarra.es>

Arquitectura de Redes, Sistemas y Servicios
3º Ingeniería de Telecomunicación

Temario

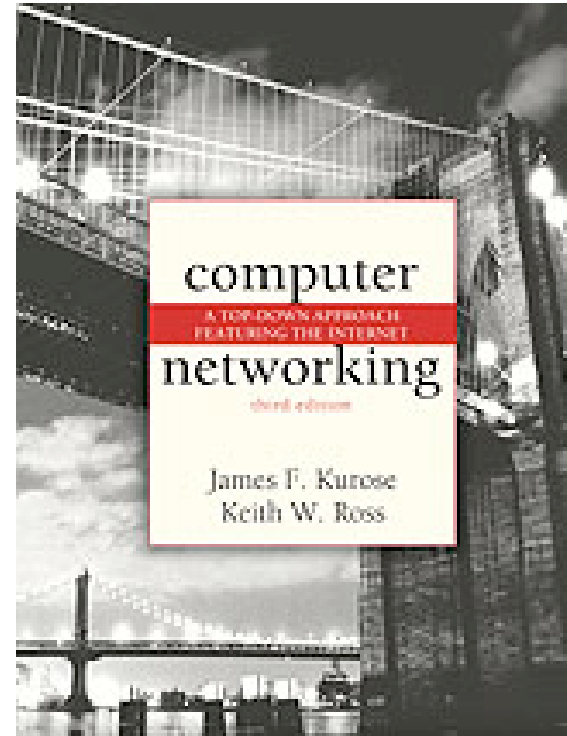
1. Introducción
2. Arquitecturas, protocolos y estándares
3. Conmutación de paquetes
4. Conmutación de circuitos
5. Tecnologías
6. Control de acceso al medio en redes de área local
7. Servicios de Internet

Temario

1. Introducción
2. Arquitecturas, protocolos y estándares
3. Conmutación de paquetes
4. Conmutación de circuitos
5. Tecnologías
6. Control de acceso al medio en redes de área local
7. Servicios de Internet
 - La Web
 - E-Mail.
 - **FTP. Telnet**
 - **Otros**
 - Desarrollo de clientes y servidores

Material

Del Capítulo 2 de
Kurose & Ross,
**“Computer Networking a top-down approach
featuring the Internet”**
Addison Wesley



Recuerde...

- Las aplicaciones de Internet se construyen utilizando protocolos de nivel de aplicación
 - Los protocolos de nivel de aplicación son poco homogéneos (diferentes para cada servicio)
 - Estamos viendo ejemplos: Web, Mail, FTP...
 - Los protocolos de nivel de aplicación usan los servicios del nivel de transporte
- Generalmente usan los servicios de TCP (Transport Control Protocol)
- Pero...
 - Pero **TCP no es el único nivel de transporte** disponible
 - También existe **UDP : User Datagram Protocol**

Nivel de transporte

UDP

Area de Ingeniería Telemática
<http://www.tlm.unavarra.es>

Arquitectura de Redes, Sistemas y Servicios
3º Ingeniería de Telecomunicación

Objetivos

- ¿Qué servicios ofrece el protocolo de transporte UDP?
- ¿Cómo?

Contenido

- Introducción
- Nivel de transporte
- UDP
 - Características
 - Formato
 - Demultiplexación
- Errores ICMP asociados

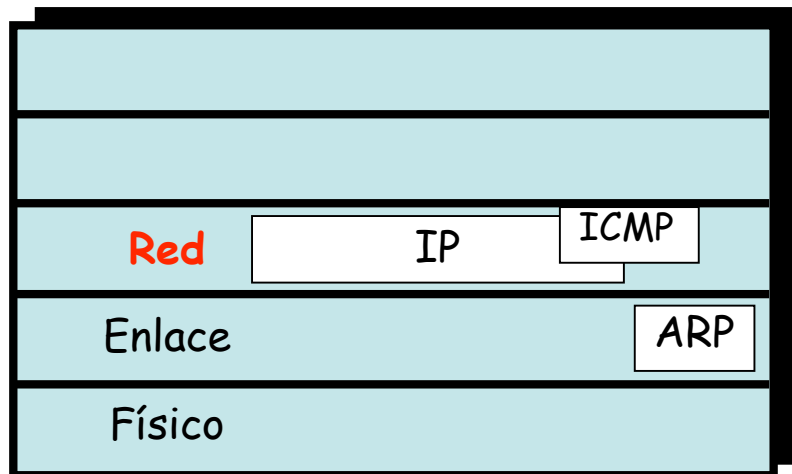
Contenido

- **Introducción**
- **Nivel de transporte**
- **UDP**
 - Características
 - Formato
 - Demultiplexación
- **Errores ICMP asociados**

Nivel de red

IP

- Ofrece un servicio best-effort
- Los paquetes se pueden retrasar, perder, desordenar, duplicar, etc.
- Van dirigidos a un host, pero ¿a qué aplicación?
- ¿Cómo debería mandar el host?
 - Demasiado rápido: congestión
 - Demasiado lento: ineficiente

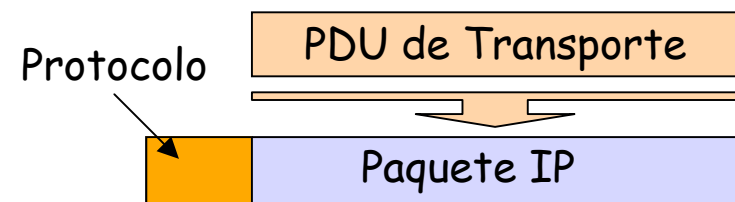
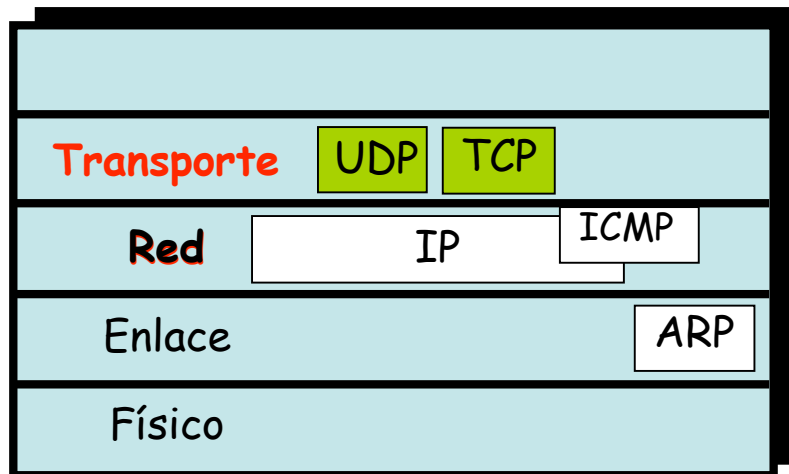
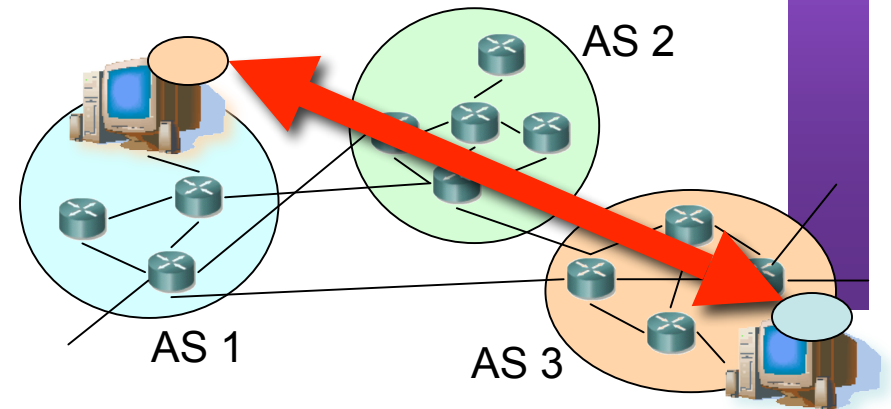


Nivel de transporte

Nivel de transporte (...)

- *Comunicación lógica* extremo a extremo entre procesos (...)
- Puede ofrecer fiabilidad, orden
- Mensajes de mayor tamaño:
 - Emisor segmenta
 - Receptor reensambla
- Inteligencia en los extremos

- TCP/IP ofrece 2 protocolos (...)
- Emplean los servicios del nivel de red (...)
- PDU del nivel de transporte: segmento



Multiplexación/Demultiplexación

Multiplexación en emisor

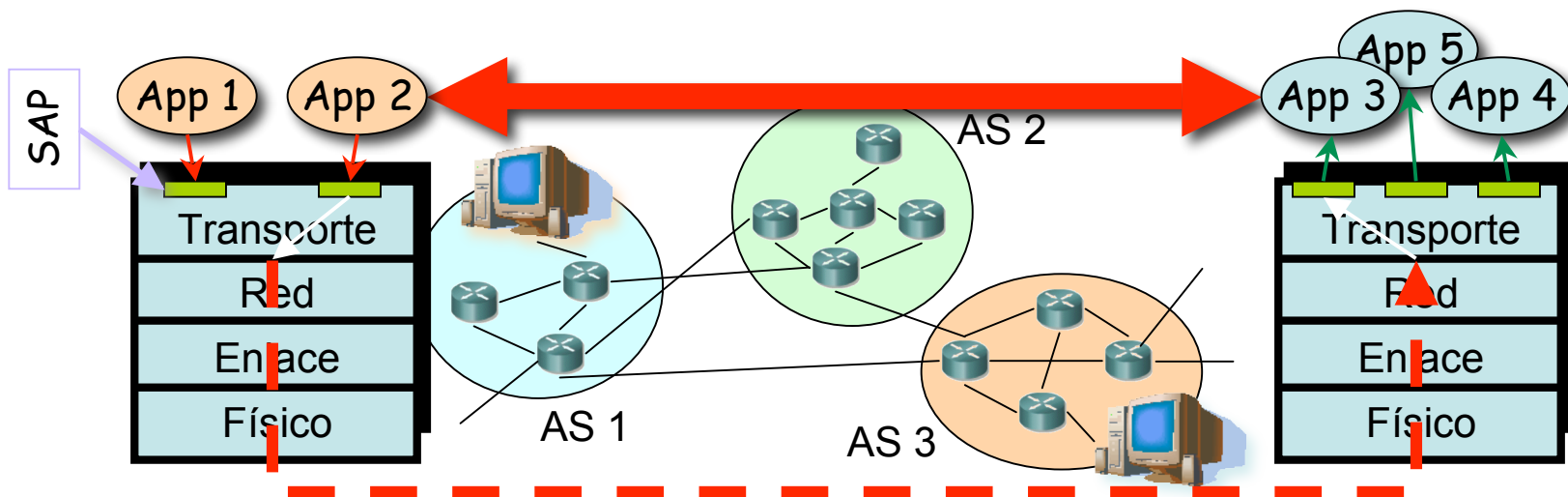
- Recoger datos de varias aplicaciones
- Añadir cabecera de transporte
- Incluye un identificador de la aplicación origen y la destino (puerto)

Enrutamiento

- Hace llegar los paquetes al host (dirección IP) correcto

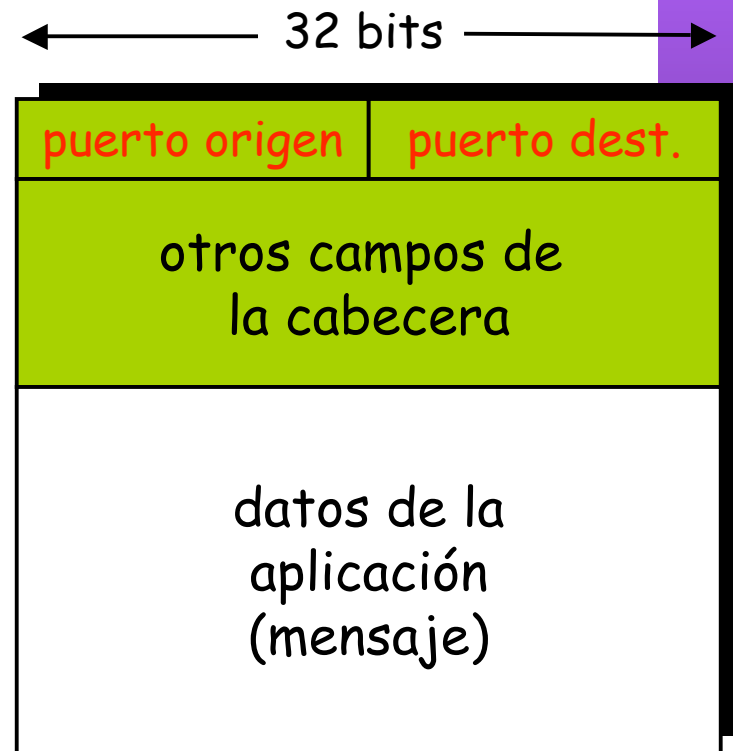
Demultiplexación en receptor

- Cada datagrama IP lleva un segmento del nivel de transporte
- Según el puerto destino y tal vez mirando también el origen decide la aplicación destino



Formato de la PDU de transporte

- TDP o UDP
- **Puerto origen**
 - Identifica a la aplicación emisora en el host
- **Puerto destino**
 - Identifica a la aplicación receptora en el host
- En el sentido contrario irán al revés
- El emisor debe conocer el puerto del receptor
- Puertos
 - [0,1023] *Well known*
 - [1024,49151] *Registered*
 - [49152,65535] Dinámicos, privados o *efímeros*



Contenido

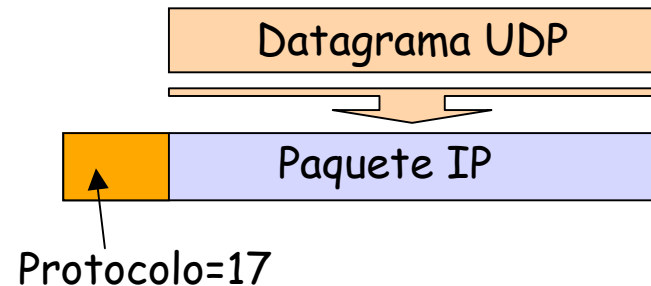
- Introducción
- Nivel de transporte
- **UDP**
 - **Características**
 - **Formato**
 - **Demultiplexación**
- Errores ICMP asociados

UDP: User Datagram Protocol

- RFC 768
- Protocolo de transporte **simple**, sin gran inteligencia
- Servicio “best effort”
- Datagramas
- Los datagramas UDP se pueden:
 - Perder
 - Llegar desordenados a la aplicación
- ¿Transferencia fiable sobre UDP?
 - Añadir fiabilidad en el nivel de aplicación
 - ¡Recuperación ante errores específica de cada aplicación!
- Sin conexión:
 - No hay handshaking entre emisor y receptor
 - Cada datagrama UDP es procesado de forma independiente a los demás
- Empleado frecuentemente para aplicaciones de streaming multimedia
 - Soportan pérdidas
 - Sensibles a la tasa de envío
- Otros usos de UDP:
 - DNS
 - SNMP

UDP: User Datagram Protocol

- ¿Por qué existe UDP?
 - Es simple: no hay que mantener estado
 - Un establecimiento de conexión añadiría retardo no deseado
 - Cabecera pequeña
 - No hay control de congestión: puede enviar tan rápido como desee
- Encapsulado en paquetes IP, protocolo 17
- Cuando un host recibe un datagrama UDP :
 - Comprueba el puerto destino en el mismo
 - Dirige el segmento a la aplicación que está esperando datos a ese puerto
- Diferentes IP origen o puertos origen van al mismo punto de acceso al servicio (SAP)



Cabecera UDP

Puerto origen

- Normalmente lo escoge el sistema operativo
- Suele ser un puerto efímero

Puerto destino

- Puerto del servidor
- *Well known* o se debe conocer por algún medio

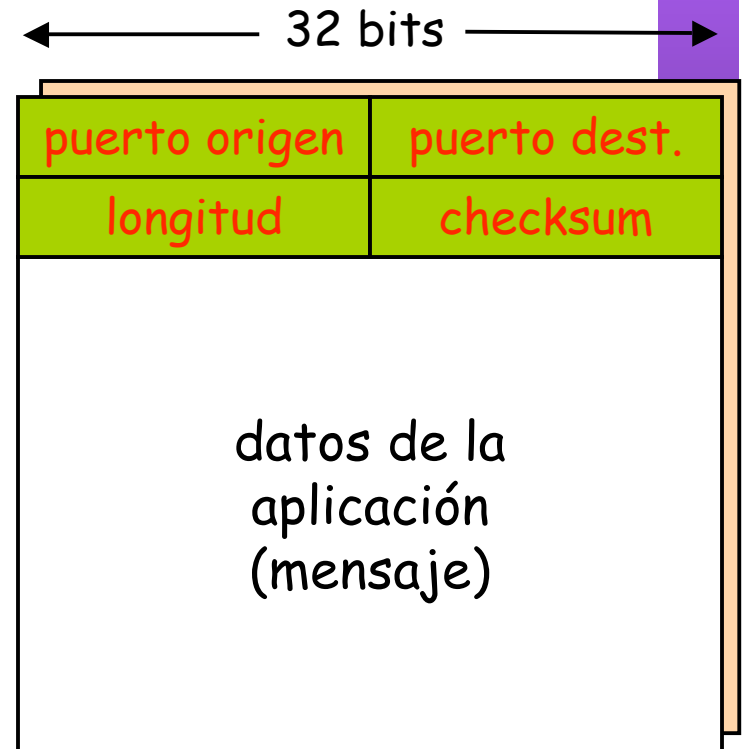
Respuesta servidor→cliente

- Sentido contrario
- Puerto origen es el del servidor (*well known*)
- Puerto destino el efímero del cliente

Longitud

- Bytes del datagrama UDP

Checksum (...)



Checksum UDP

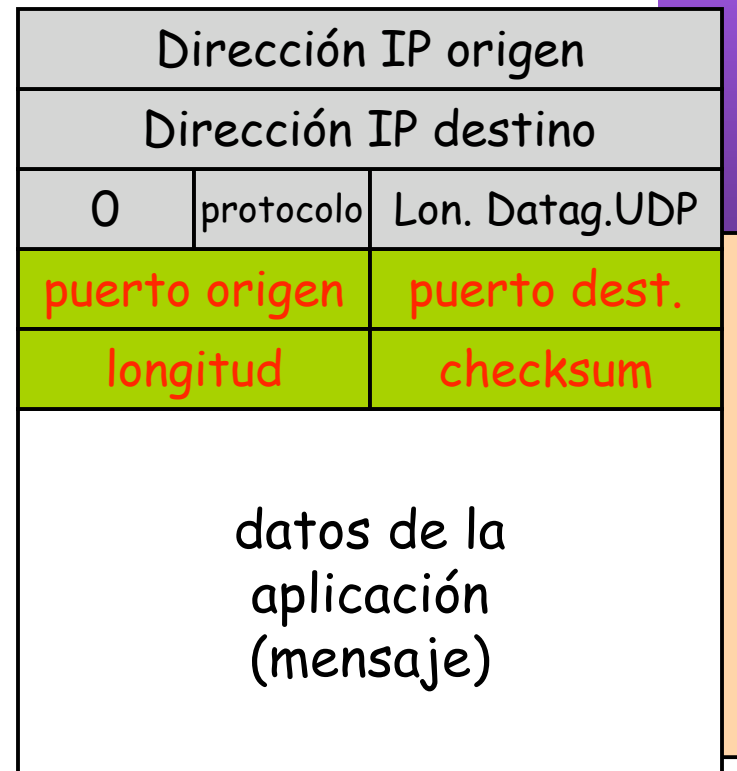
Objetivo: detectar “errores” (ej., bits cambiados) en un datagrama
 Cubre a la cabecera y los datos (y parte de la cabecera IP)

Emisor:

- Trata el datagrama como una secuencia de enteros de 16 bits
- Complemento a 1 de la suma (en complemento a 1) del datagrama y *pseudocabecera*
- Coloca el checksum en el campo

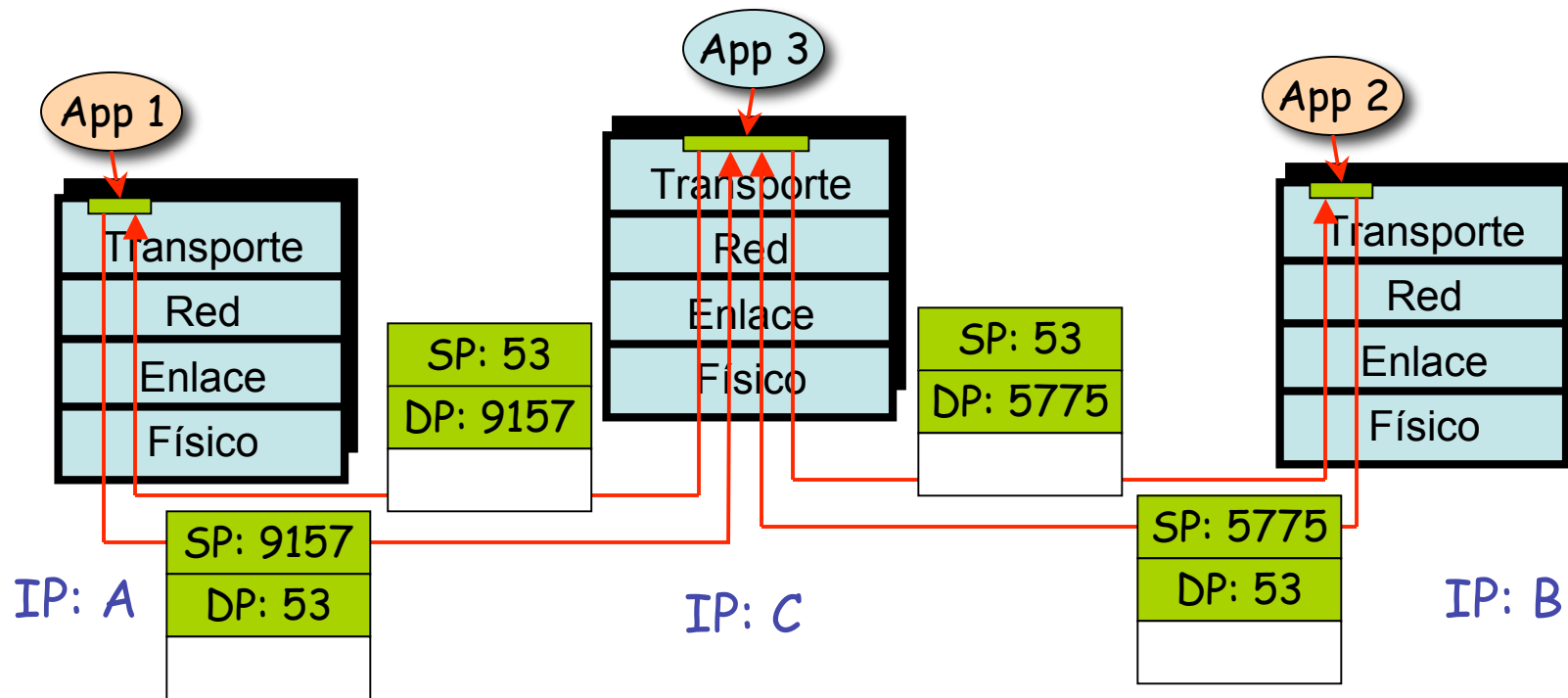
Receptor:

- Hace la suma en complemento a 1 de todo el datagrama
- ¿Da 0?
 - NO - error detectado
 - Sí - no hay errores detectados
 ¡Pero aún así puede haberlos!



Opcional

Demultiplexación: Ejemplo

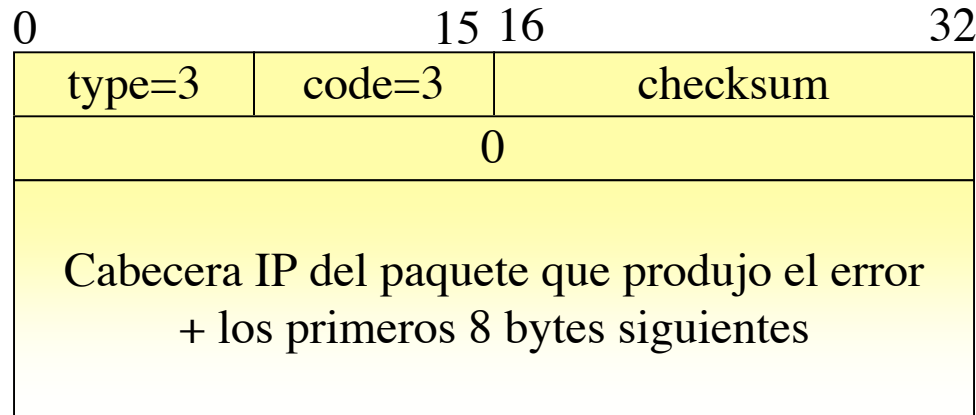


Contenido

- Introducción
- Nivel de transporte
- UDP
 - Características
 - Formato
 - Demultiplexación
- **Errores ICMP asociados**

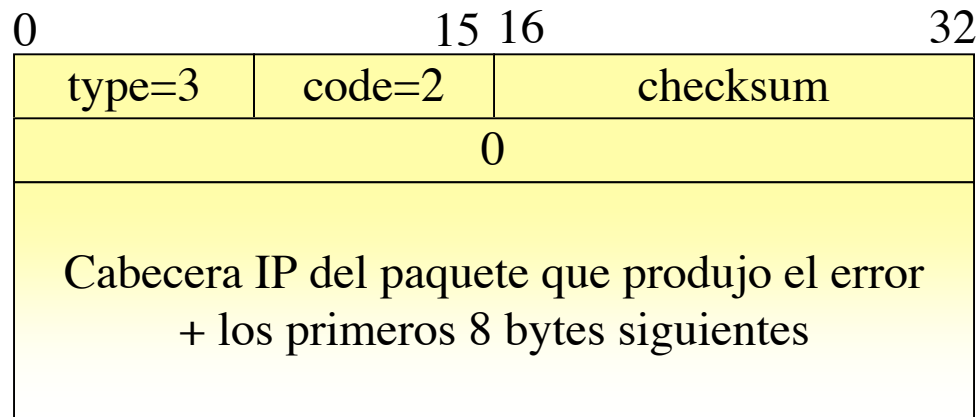
Mensajes ICMP

- Puerto destino inalcanzable (*destination port unreachable*)
 - Generado por un host que recibe un datagrama UDP para cuyo puerto destino no espera mensajes ninguna aplicación
 - tipo=3 (destino inalcanzable), código=3



Mensajes ICMP

- Protocolo inalcanzable
 - Generado cuando el host receptor del paquete IP no conoce el protocolo que viene indicado en la cabecera del mismo
 - tipo=3 (destino inalcanzable), código=2



Resumen

- Dos protocolos de nivel de transporte a elegir

UDP da pocos más servicios que IP

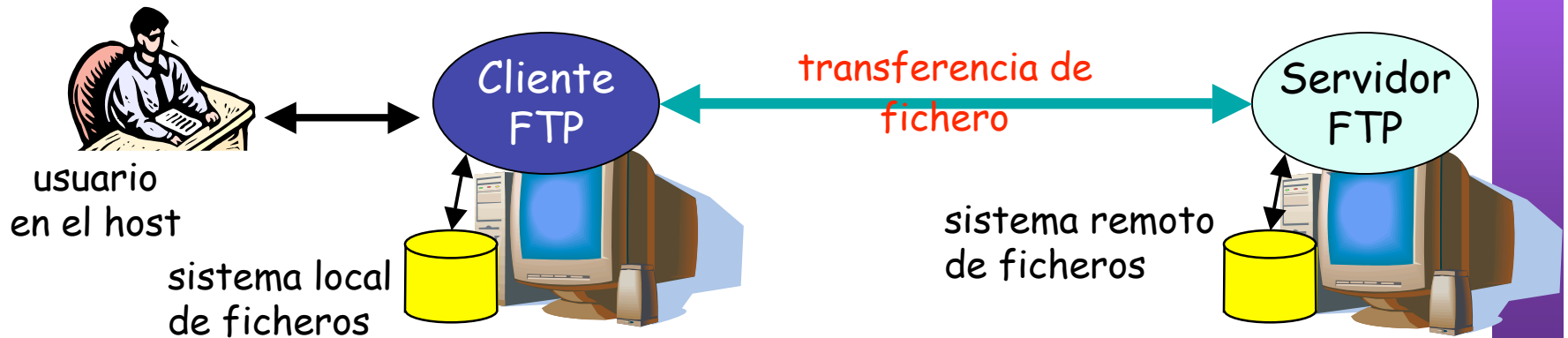
- Principalmente la multiplexación por puertos
- Pero es simple

TCP ofrece más servicios

- También multiplexación por puertos basada en conexiones
- Transporte fiable
- Control de flujo
- Control de congestión

Servicios: FTP y Telnet

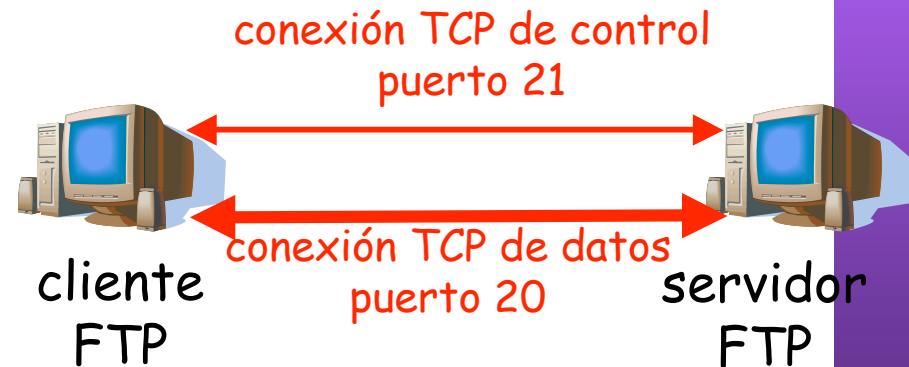
FTP: File Transfer Protocol



- Transferencia de fichero hacia/desde host remoto
- modelo cliente-servidor
 - *cliente*: extremo que inicia la transferencia (bien sea desde o hacia el extremo remoto)
 - *servidor*: host remoto
- FTP: RFC 959
- Servidor FTP: TCP puerto 21

FTP: conexiones de datos y control separadas

- El **cliente FTP** **contacta** con el servidor en el puerto 21
- **Se autentifica** a través de esta **conexión de control**
- Puede explorar los directorios remotos enviando comandos por la conexión de control
- Conexión de control “out of band”
- Cuando el **servidor** recibe un comando para una transferencia de fichero **abre una conexión TCP con el cliente**



- Servidor emplea el puerto 20 en esa conexión
- Tras transferir el fichero cierra esa conexión de datos
- El servidor FTP mantiene el “estado”: directorio actual, autenticación

Comandos y respuestas FTP

Comandos de ejemplo:

- Enviados como texto ASCII por el canal de control
- **USER username**
- **PASS password**
- **LIST** devuelve una lista de los ficheros en el directorio actual
- **RETR filename** Obtiene el fichero
- **STOR filename** Almacena el fichero en el host remoto

Códigos de respuesta:

- Código de estado y frase (como en HTTP)
- **331 Username OK, password required**
- **125 data connection already open; transfer starting**
- **425 Can't open data connection**
- **452 Error writing file**

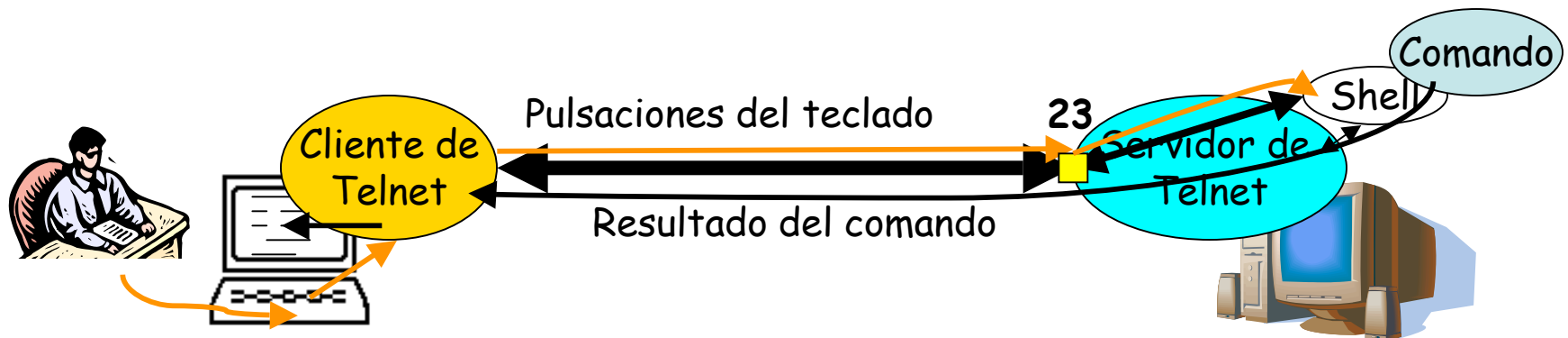
Login remoto (Telnet)

Uso interactivo remoto de una computadora

Funcionamiento:

- Usuario ejecuta un **cliente de Telnet** especificando la máquina servidor (...)
 - Conexión TCP con el servidor (**puerto servidor 23**) (...)
 - El servidor crea una **Shell** que queda conectada a la conexión TCP (...)
- Las pulsaciones del teclado del usuario se transmiten por la conexión a la Shell (...)
 - La **shell ejecuta los comandos** que escribe el usuario (...)
 - Resultado del comando a pantalla vuelve por la conexión TCP y sale en la pantalla del cliente (...)

Otros servicios similares:
 rlogin, rsh, ssh



Ejemplo de Telnet

```
$ telnet 10.1.11.1
Trying 10.1.11.1...
Connected to 10.1.11.1.
Escape character is '^]'.

Red Hat Linux release 6.0 (Hedwig)
Kernel 2.2.5-15 on an i586
login: ro
Password:
Last login: Fri Nov  9 09:30:27 from lucas.net.tlm.unavarra.es
[ro@pclr11 ro]$ ls -al
total 3
drwxr-xr-x  2 ro      users      1024 Oct 31 20:10 .
drwxr-xr-x  5 root    root       1024 Sep 25 19:25 ..
-rw-----  1 ro      users      482  Nov  9 09:30 .bash_history
[ro@pclr11 ro]$ date
Fri Nov  9 09:50:57 CET 2001
[ro@pclr11 ro]$ ls
[ro@pclr11 ro]$ exit
logout
Connection closed by foreign host.
```

Ejemplo de Telnet

```
$ /opt3/ro/ficheros/bin/tcpdump_ro -ttnlS tcp and host 10.1.11.1
Kernel filter, protocol ALL, datagram packet socket
tcpdump: listening on all devices
154.171 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: S 1145:1145(0)
154.175 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: S 2026:2026(0) ack 1146
154.175 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1146:1146(0) ack 2027
154.177 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1146:1173(27) ack 2027
154.178 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2027:2027(0) ack 1173
154.215 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2027:2039(12) ack 1173
154.215 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1173:1173(0) ack 2039
154.216 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2039:2078(39) ack 1173
154.218 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1173:1291(118) ack 2078
154.222 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2078:2081(3) ack 1291
154.222 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1291:1294(3) ack 2081
154.241 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2081:2081(0) ack 1294
154.242 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2081:2150(69) ack 1294
154.243 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1294:1297(3) ack 2150
154.261 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2150:2150(0) ack 1297
154.275 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2150:2157(7) ack 1297
154.292 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1297:1297(0) ack 2157
155.980 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1297:1298(1) ack 2157
155.980 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2157:2158(1) ack 1298
```

Ejemplo de Telnet

```
155.992 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1298:1298(0) ack 2158
156.111 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1298:1299(1) ack 2158
156.112 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2158:2159(1) ack 1299
156.132 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1299:1299(0) ack 2159
156.279 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1299:1301(2) ack 2159
156.280 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2159:2161(2) ack 1301
156.292 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1301:1301(0) ack 2161
156.298 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2161:2171(10) ack 1301
156.312 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1301:1301(0) ack 2171
156.847 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1301:1302(1) ack 2171
156.861 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2171:2171(0) ack 1302
156.991 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1302:1303(1) ack 2171
157.011 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2171:2171(0) ack 1303
157.167 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1303:1304(1) ack 2171
157.181 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2171:2171(0) ack 1304
157.303 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1304:1305(1) ack 2171
157.321 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2171:2171(0) ack 1305
157.483 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1305:1306(1) ack 2171
157.501 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2171:2171(0) ack 1306
157.643 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1306:1307(1) ack 2171
157.661 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2171:2171(0) ack 1307
157.823 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1307:1309(2) ack 2171
157.847 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2171:2171(0) ack 1309
157.871 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2171:2173(2) ack 1309
```


Ejemplo de Telnet

```
157.882 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1309:1309(0) ack 2173
157.883 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2173:2237(64) ack 1309
157.902 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1309:1309(0) ack 2237
158.011 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2237:2253(16) ack 1309
158.022 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1309:1309(0) ack 2253
158.907 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1309:1310(1) ack 2253
158.908 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2253:2254(1) ack 1310
158.922 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1310:1310(0) ack 2254
159.007 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1310:1311(1) ack 2254
159.008 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2254:2255(1) ack 1311
159.022 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1311:1311(0) ack 2255
159.119 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1311:1312(1) ack 2255
159.120 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2255:2256(1) ack 1312
159.132 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1312:1312(0) ack 2256
159.327 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1312:1313(1) ack 2256
159.328 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2256:2257(1) ack 1313
159.342 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1313:1313(0) ack 2257
159.707 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1313:1314(1) ack 2257
159.708 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2257:2258(1) ack 1314
159.722 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1314:1314(0) ack 2258
159.775 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1314:1315(1) ack 2258
159.776 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2258:2259(1) ack 1315
159.792 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1315:1315(0) ack 2259
160.119 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1315:1317(2) ack 2259
```

Ejemplo de Telnet

```
160.120 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2259:2261(2) ack 1317
160.132 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1317:1317(0) ack 2261
160.133 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2261:2270(9) ack 1317
160.152 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1317:1317(0) ack 2270
160.153 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2270:2473(203) ack 1317
160.172 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1317:1317(0) ack 2473
162.031 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1317:1318(1) ack 2473
162.032 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2473:2474(1) ack 1318
162.052 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1318:1318(0) ack 2474
162.128 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1318:1319(1) ack 2474
162.129 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2474:2475(1) ack 1319
162.142 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1319:1319(0) ack 2475
162.355 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1319:1320(1) ack 2475
162.356 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2475:2476(1) ack 1320
162.372 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1320:1320(0) ack 2476
162.423 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1320:1321(1) ack 2476
162.424 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2476:2477(1) ack 1321
162.442 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1321:1321(0) ack 2477
162.611 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1321:1323(2) ack 2477
162.612 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2477:2479(2) ack 1323
162.622 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1323:1323(0) ack 2479
162.623 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2479:2509(30) ack 1323
162.642 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1323:1323(0) ack 2509
162.643 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2509:2525(16) ack 1323
```

Ejemplo de Telnet

```
162.662 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1323:1323(0) ack 2525
165.247 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1323:1324(1) ack 2525
165.248 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2525:2526(1) ack 1324
165.262 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1324:1324(0) ack 2526
165.306 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1324:1325(1) ack 2526
165.307 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2526:2527(1) ack 1325
165.322 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1325:1325(0) ack 2527
165.406 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1325:1327(2) ack 2527
165.407 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2527:2529(2) ack 1327
165.422 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1327:1327(0) ack 2529
165.423 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2529:2545(16) ack 1327
165.442 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1327:1327(0) ack 2545
165.998 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1327:1328(1) ack 2545
165.999 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2545:2546(1) ack 1328
166.012 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1328:1328(0) ack 2546
166.254 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1328:1329(1) ack 2546
166.256 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2546:2547(1) ack 1329
166.272 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1329:1329(0) ack 2547
166.351 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1329:1330(1) ack 2547
166.352 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2547:2548(1) ack 1330
166.372 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1330:1330(0) ack 2548
166.490 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1330:1331(1) ack 2548
166.491 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2548:2549(1) ack 1331
166.502 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1331:1331(0) ack 2549
```

Ejemplo de Telnet

```
166.807 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: P 1331:1333(2) ack 2549
166.808 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: P 2549:2551(2) ack 1333
166.816 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: FP 2551:2559(8) ack 1333
166.816 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: . 1333:1333(0) ack 2560
166.817 eth0 P 1.1.1.12.1798 > 10.1.11.1.telnet: F 1333:1333(0) ack 2560
166.818 eth0 P 10.1.11.1.telnet > 1.1.1.12.1798: . 2560:2560(0) ack 1334
```

Otros servicios...

Mas servicios...

- DNS
- P2P
- Mensajería

Contenido

- **DNS**
- **P2P**
- **Mensajería**

El problema de los nombres

- Las direcciones IP, que identifican a los interfaces de los hosts, son números de 32 bits
- Sencillas de manejar para las máquinas, complicado para los humanos
- Más sencillo memorizar nombres textuales
- Hace falta “traducir” el nombre textual en la dirección numérica para que se pueda realizar la comunicación. Esto se llama “resolver el nombre”
- La traducción se realiza mediante el Sistema de Nombres de Dominio o DNS (Domain Name System)

Domain Name System

- Es una **base de datos distribuida**
- Servidores de nombres organizados **jerárquicamente**
- Es un **protocolo de aplicación**
- Permite a los hosts traducir entre nombres y direcciones
 - Funcionalidad vital
 - Implementada como protocolo a nivel de aplicación
 - Complejidad en los extremos de la red

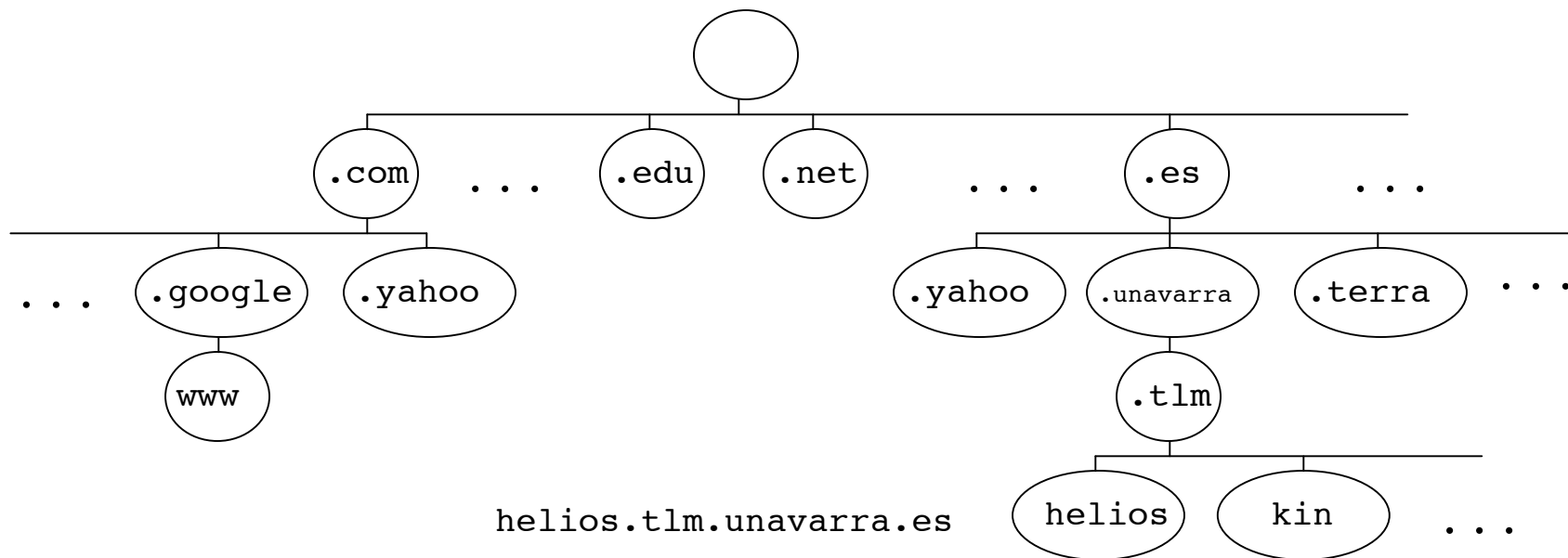
¿Por qué no centralizado?

- Punto de fallo
- Volumen de tráfico
- Base de datos centralizada lejana
- Mantenimiento

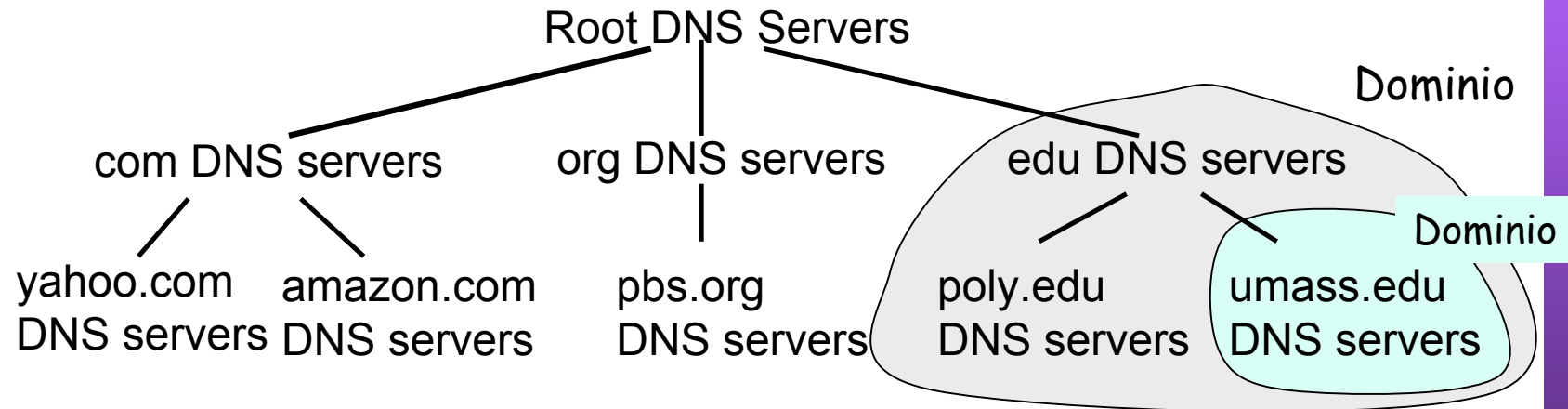
¡ No escala !

Jerarquía de nombres

- Los nombres están formados por segmentos alfanuméricos separados por puntos (no distingue mayúsculas)
helios.tlm.unavarra.es
www.google.com
- Estructura jerárquica (. . .)



B.D. jerárquica distribuida

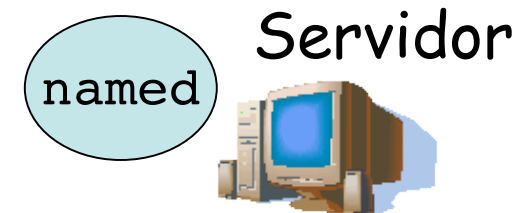
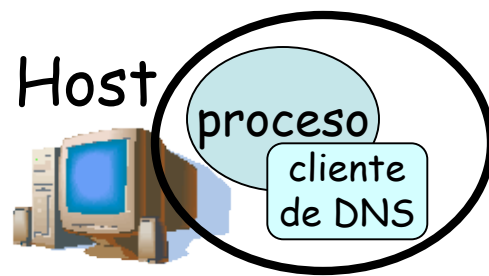


El cliente busca la IP de **www.amazon.com**:

- El cliente pregunta a un **servidor Root** para encontrar el servidor de DNS del dominio **com**
- El cliente pregunta al **servidor del dominio com** para obtener el servidor del dominio **amazon.com**
- El cliente pregunta al servidor DNS del dominio **amazon.com** para obtener la IP de **www.amazon.com**.

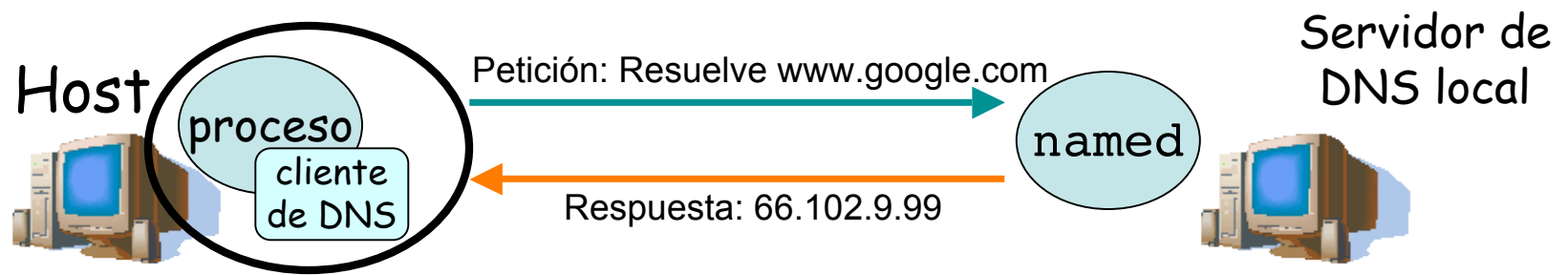
Implementación

- El servidor es un programa específico pero el cliente es generalmente solo unas funciones en una librería (*resolver*) (...)
- La aplicación cliente de DNS es la propia aplicación del usuario (...)
- El software típico que lo implementa es BIND (Berkeley Internet Name Domain) (el programa servidor se llama *named*) (...)
- Emplea UDP (puerto servidor 53) o TCP si el mensaje de respuesta es de más de 512 Bytes.



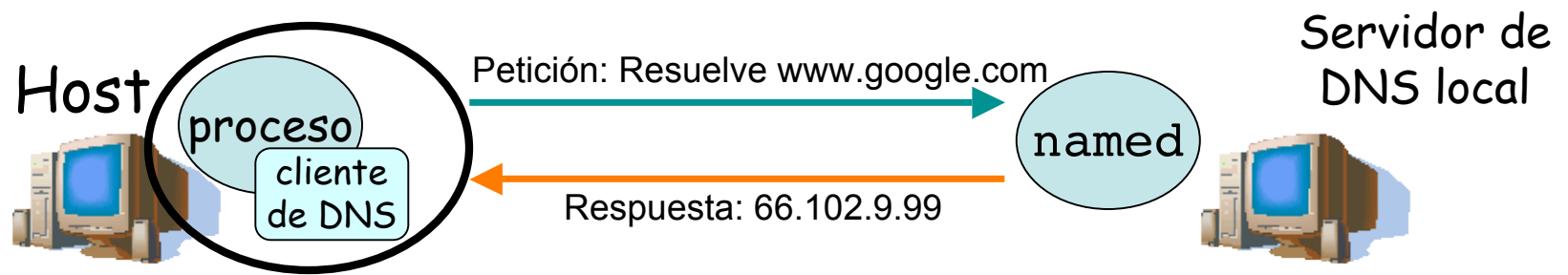
Funcionamiento

- Cada ISP posee un servidor de nombres local (...)
- Los hosts tienen configurado a su servidor local
- Cuando un host desea resolver un nombre hace la petición a su servidor local el cual le devuelve la respuesta (... ..)



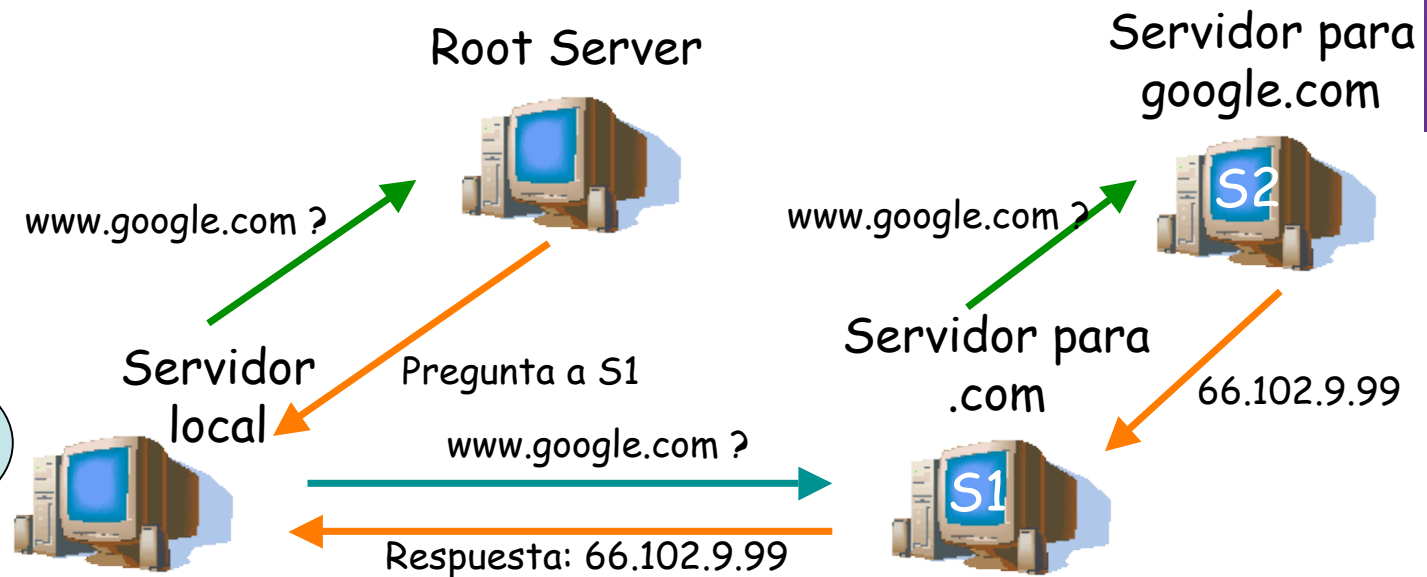
Funcionamiento

- ¿Cómo conoce la respuesta el servidor local?
 - Si es el servidor autoritario (***authoritative server***) para el dominio en el que está esa máquina él tiene la porción de la base de datos distribuida en la que está el mapeo (***zone file***)
 - Si no lo es preguntará a un ***Root Server***



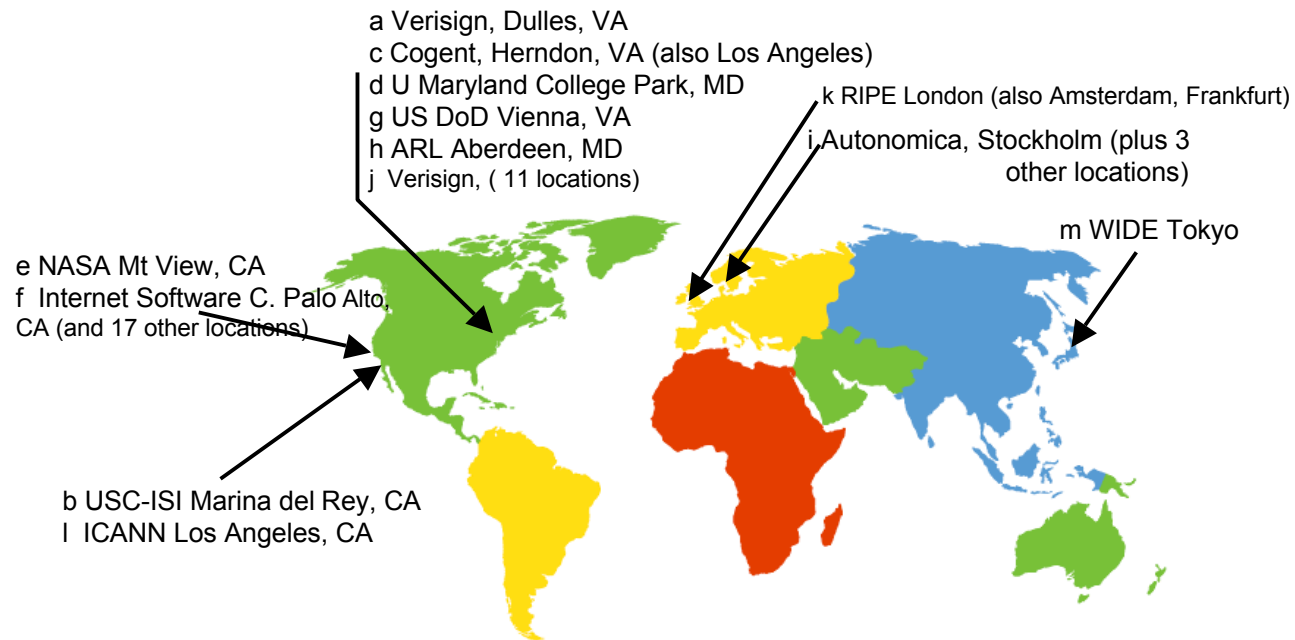
Funcionamiento

- El servidor local pregunta a un **Root Server** (...)
- Éste le devuelve la dirección de un servidor intermedio (petición **iterativa**) (...)
- El Servidor local hace una petición recursiva a ese servidor (...)
- Continuará haciendo la petición (**recursiva**) hasta que llegue un servidor autoritario (. . .)
- Todas las peticiones son recursivas menos la petición al Root Server para reducir la carga sobre los Root



DNS: Root name servers

- 13 en el mundo
- En el fichero de configuración de cada servidor de DNS



TLDs, Authoritative Servers, cache

Servidores de Top-level domains (TLD):

- Responsables de *com, org, net, edu*, (etc) y de los dominios raíz de países (*es, uk, fr, ca, jp, etc*)
- ESNIC para el TLD *.es* (<http://www.nic.es>)

Authoritative DNS servers:

- Servidores DNS de organizaciones
- Mantienen el mapeo autorizado para los nombres dentro del dominio de la organización

Fully Qualified Domain Name (FQDN)

- En realidad la raíz del árbol tiene también “nombre” pero es nulo
- Un FQDN incluye el nombre hasta la raíz, o sea, termina en un “.”
[www.tlm.unavarra.es.](http://www.tlm.unavarra.es)

Una vez que un servidor de DNS aprende un mapeo lo cachea

- Las entradas en la cache caducan tras un tiempo
- Normalmente los servidores de los TLD van a estar cacheados en los servidores locales
 - Así que los Root no se suelen visitar

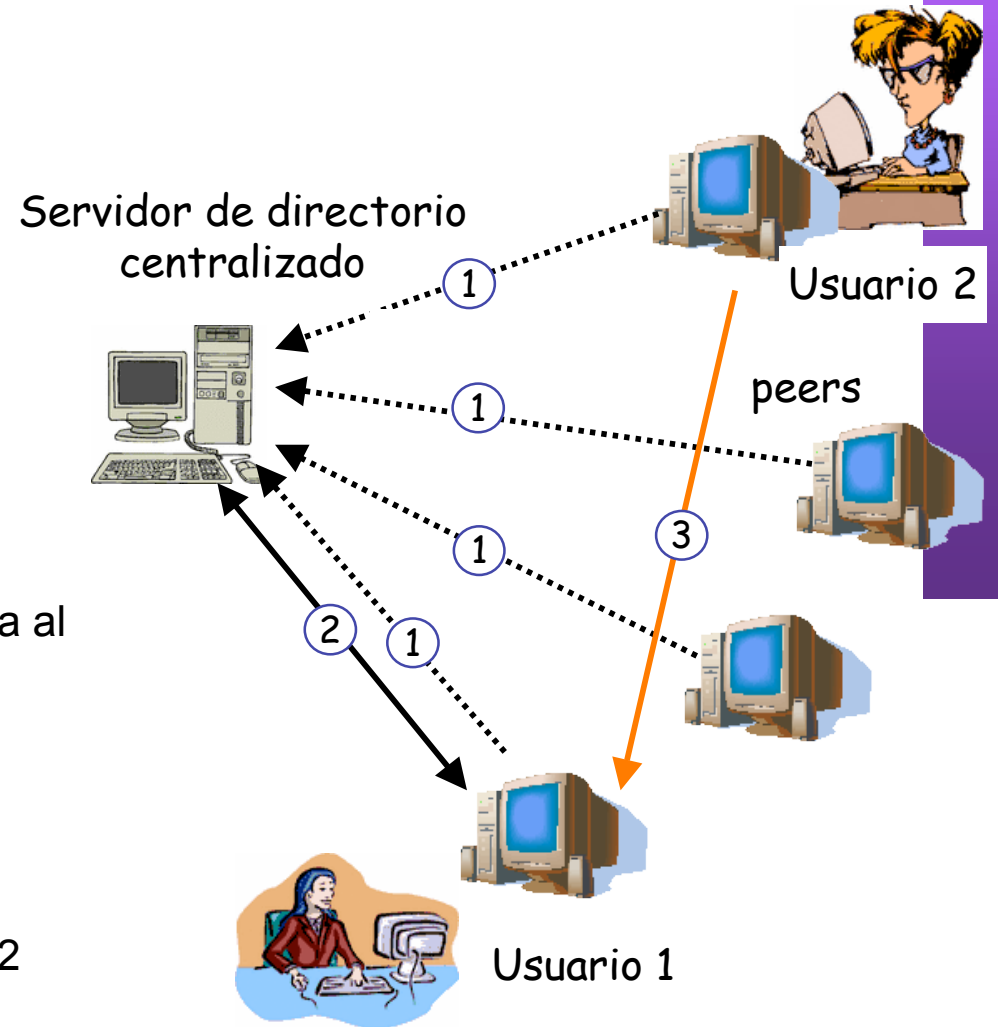
Contenido

- DNS
- **P2P**
- Mensajería

P2P: directorio centralizado

Diseño original de "Napster"

- 1) Cuando un peer se conecta, informa al servidor central:
 - Dirección IP
 - contenido
- 2) Usuario 1 hace una búsqueda de "Requiem"
- 3) Usuario 1 pide el fichero a Usuario 2



Ventajas e inconvenientes

Ventajas

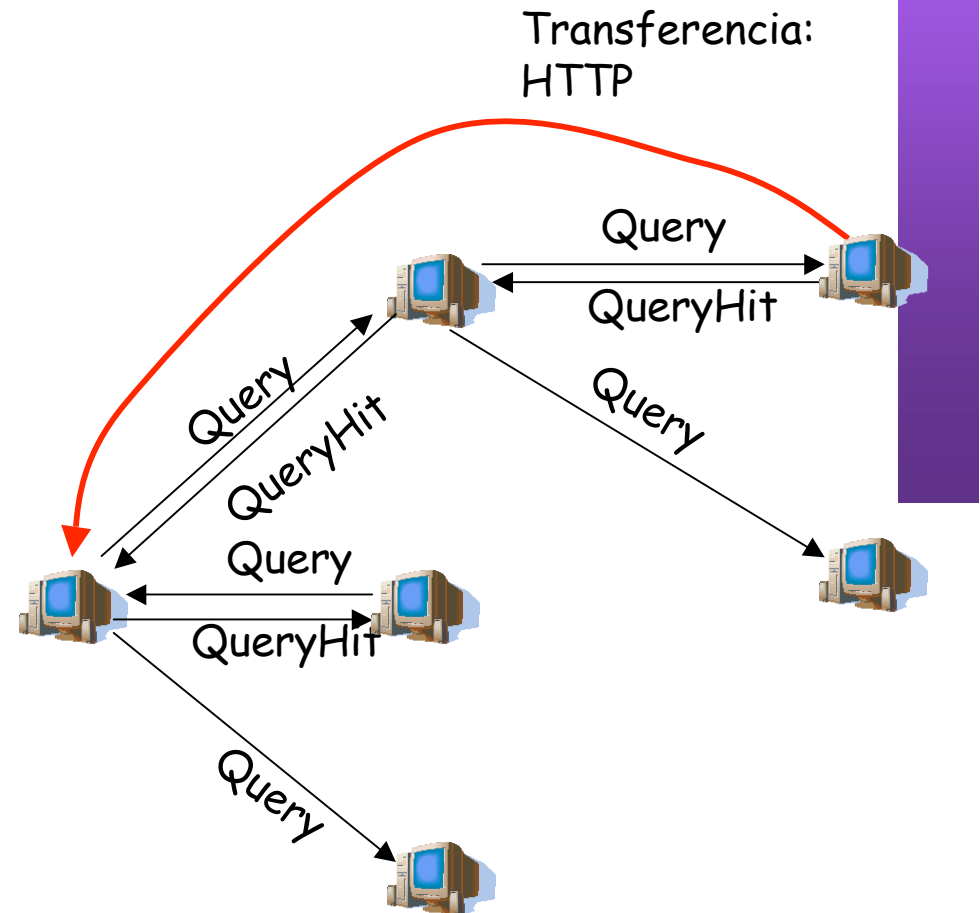
- Todos los peers son servidores
- **Altamente escalable**

Inconvenientes

- **Un punto de fallo central**
- Impone un límite de prestaciones
- Infracción de copyrights!

Gnutella

- Completamente distribuido
- Dominio público
- Overlay network
 - Grafo
 - Cada conexión un enlace
- Petición de búsqueda enviada sobre las conexiones TCP
- peers reenvían la petición
- Respuesta enviada por el camino inverso
- Escalabilidad: limitar el alcance de la inundación



Contenido

- DNS
- P2P
- **Mensajería**

Servicios de conversación

- Conversación a líneas en máquinas UNIX: **write**

```
[daniel@t1m13 daniel]$ write lir  
Hola tu  
[daniel@t1m13 daniel]$
```

```
[lir@t1m13 lir]$  
Message from daniel@t1m13.net.t1m.unavarra.es  
on pts/0 at 18:39 ...  
Hola tu  
EOF
```

- Conversación en terminal UNIX completo: **talk**

```
[daniel@t1m13 daniel]$ talk lir@t1m21
```

```
[Connection established]  
Hola
```

```
[-----]  
Pues hola
```

```
Message from Talk_Daemon@t1m13.net.t1m.unavarra.es  
at 18:30 ...  
talk: connection requested by daniel@t1m13.  
talk: respond with: talk daniel@t1m13.  
[lir@t1m13 lir]$ talk daniel@t1m13
```

```
[Connection established]  
Pues hola
```

```
[-----]  
Hola
```

Evolución de los servicios clásicos de conversación

- Internet Relay Chat (IRC):
 - Los clientes se conectan a un servidor central
 - Existen “habitaciones”. Todos los usuarios que ejecuten el comando para “unirse” a una habitación podrán leer lo que cualquier otro en esa habitación escriba (... ..)

```

otilio[9]% irc dmorato irc.arrakis.es

*** Connecting to port 6667 of server
-atreides.irc-hispano.org- *** Looking
-atreides.irc-hispano.org- *** Found y
*** As2gcu.DMOyqd.virtual is now your
+atreides.irc-hispano.org)
*** Welcome to the IRC-Hispano Interne
+atreides.irc-hispano.org)

% /join #pamplona
*** dmorato (daniel@As2gcu.DMOyqd.virt
*** Topic for #pamplona: Pamplona.cjb.
*** #pamplona CHaN 1065118105
*** #pamplona 1006939646
-CHaN- Bienvenido a #Pamplona,ciudad s
  
```

The screenshot shows an IRC client window with the following content:

- Window title: X-Chat
- Channel: #pamplona
- Server: Pamplona.cjb.net
- Chat log:
 - *** Current local users: 1942 Max: 3208 (Jueves, 2 de Octubre de 2003 -- 22:53 +02:00, since 20030919-13:34)
 - *** Current global users: 18864 Max: 31203 (Martes, 30 de Septiembre de 2003 -- 23:02 +02:00, since 20030919-13:34)
 - *** - atreides.irc-hispano.org Message of the Day -
 - *** - 19/9/2003 13:34
 - *** -
 - *** -
 - *** - Bienvenidos al servidor de IRC de Arrakis
 - *** -
 - *** - Puedes consultar normas de convivencia en:
 - *** - <http://www.irc-hispano.org>
 - *** -
 - *** -
 - *** End of /MOTD command
 - *** -atreides.irc-hispano.org- on 1 ca 1(2) ft 10(10) tr
 - *** dmorato sets mode +x dmorato
 - *** dmorato (daniel@As2gcu.DMOyqd.virtual) has joined #pamplona
 - *** Topic for #pamplona is Pamplona.cjb.net
 - *** Topic for #pamplona set by CHaN at Thu Oct 2 20:08:25 2003
 - *** -CHaN- Bienvenido a #Pamplona,ciudad sin ley
 - *** <|[THC]|> paz y amor y mucho rap!
- User input: dmorato

Evolución de los servicios clásicos de conversación Messengers (...)

