

TEMA 2

SERVICIOS Y PROTOCOLOS

DE APLICACIÓN EN INTERNET

Fundamentos de Redes

2017/2018

GRUPO C

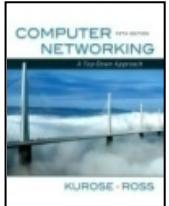
Sandra Sendra Compte (ssendra@ugr.es)



ugr

Universidad
de Granada

➤ Bibliografía Básica:

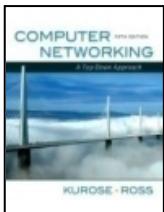


Capítulo 2 (2.1, 2.2, 2.4, 2.5), James F. Kurose y Keith W. Ross.
COMPUTER NETWORKING. A TOP-DOWN APPROACH, 5^a Edición, Addison-Wesley, 2010, ISBN: 9780136079675.



Capítulo 11, Pedro García Teodoro, Jesús Díaz Verdejo y Juan Manuel López Soler. **TRANSMISIÓN DE DATOS Y REDES DE COMPUTADORES**, Ed. Pearson, 2016, ISBN: 978-0-273-76896-8.

➤ Para saber más...



Capítulos 7 y 8, James F. Kurose y Keith W. Ross. **COMPUTER NETWORKING. A TOP-DOWN APPROACH**, 5^a Edición, Addison-Wesley, 2010, ISBN: 9780136079675.

➤ Agradecimientos:

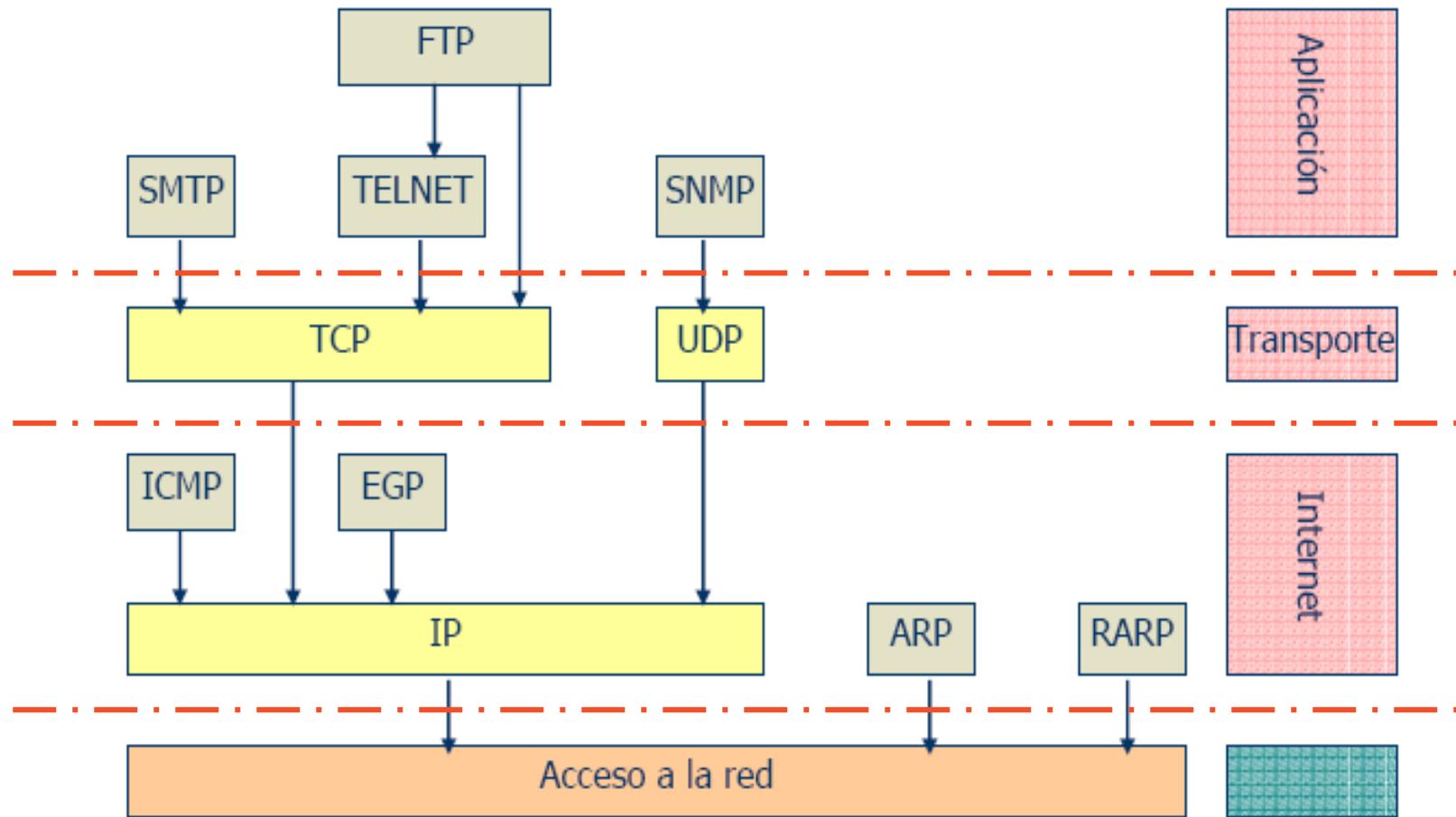
Estas transparencias están inspiradas en las transparencias utilizadas por Kurose y Ross en la Universidad de Massachusetts.



Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. **Introducción a las aplicaciones de red**
2. Servicio de Nombres de Dominio (DNS)
3. La navegación Web
4. El Correo electrónico
5. Protocolos seguros
6. Aplicaciones multimedia
7. Aplicaciones para interconectividad de redes locales
8. Cuestiones y ejercicios

Estructura de protocolos





➤ Protocolos TCP/IP:

- El origen de esta familia de protocolos fue la red ARPANET (en ella se desarrollaron los conceptos fundamentales de diseño y gestión de redes)
- Los niveles más bajos (**enlace y físico**) no están implementados ya que se diseñó para no depender de una red física concreta:
 - Los protocolos **ARP** (Adress Resolution Protocol) y **RARP** (Reverse Adress Resolution Protocol) se encargan de enlazar los sistemas de direccionamiento IP y el de la red física utilizada.



INTRODUCCIÓN A LAS APLICACIONES DE RED: PROTOCOLOS TCP/IP

➤ Protocolos TCP/IP:

Capa de Red:

- La base de la familia de protocolos es el nivel de Red (**IP, Internet Protocol**). Es un protocolo de conmutación de paquetes muy sencillo, de tipo datagrama, de forma que se pueda implementar en cualquier tipo de máquina.
- Existen actualmente dos versiones IPv4, IPv6
- Protocolos de apoyo:
 - **ICMP** (Internet Control Message Protocol): comunicación de mensajes entre nodos de la red
 - **IGMP** (Internet Group Management Protocol): envío de mensajes a grupos de usuarios



INTRODUCCIÓN A LAS APLICACIONES DE RED: PROTOCOLOS TCP/IP

➤ Protocolos TCP/IP:

La capa de transporte:

- Implementa protocolos extremo a extremo(entre nodo origen y destino de la información). Se definen dos protocolos:
 - **TCP(Transmission Control Protocol):**
 - Es un protocolo orientado a la conexión con control de errores
 - Se encarga también del control de flujo
 - Fragmentado y reensamblado de flujos (garantiza el secuenciamiento)
 - **UDP (User Datagram Protocol) :**
 - Es un protocolo sin conexión (datagrama)
 - No realiza control de errores
 - No garantiza el secuenciamiento de la información
 - Es muy rápido
 - Útil para peticiones aisladas, o transmisión de audio o vídeo



INTRODUCCIÓN A LAS APLICACIONES DE RED: PROTOCOLOS TCP/IP

➤ Protocolos TCP/IP:

Capa Aplicación (protocolos de alto nivel):

- Protocolos basados en ICMP:
 - **PING**: solicitud de eco
- Protocolos basados en TCP:
 - **TELNET**: terminal remoto
 - **FTP(File Transfer Protocol)**: transmisión de ficheros
 - **SMTP(Simple Mail Transfer Protocol)**: correo electrónico
 - **HTTP(HyperText Transfer Protocol**: páginas web?
 - **RPC (Remote Procedure Call)**: ejecución de procesos remotos
- Protocolos basados en UDP:
 - **SNMP(Simple Network Management Protocol)**: gestión de red
 - **BOOTP**: arranque remoto
 - **DNS(Domain Name System)**?
 - **RPC(Remote Procedure Call)**: ejecución de procesos remotos
 - **NFS/RPC (Network File System)** (gestión de ficheros en red)

INTRODUCCIÓN A LAS APLICACIONES DE RED: PROTOCOLOS DE TRANSPORTE

Servicio TCP:

Orientado a conexión
Transporte fiable
Control de flujo
Control de congestión

Servicio UDP:

No orientado a conexión
Transporte no fiable
Sin control de flujo
Sin control de congestión,
¿Para qué existe UDP?

TCP y UDP (capa de transporte) al ser usuarios del protocolo IP (capa de red) **no garantizan**:

- Retardo acotado
- Fluctuaciones acotadas
- Mínimo *throughput*
- Seguridad



ARQUITECTURA CLIENTE/SERVIDOR

INTRODUCCIÓN A LAS APLICACIONES DE RED: ARQUITECTURA CLIENTE/SERVIDOR

➤ Arquitectura cliente-servidor:

- La arquitectura **cliente-servidor** es una forma específica de diseño de aplicaciones, aunque también se conoce con este nombre a las computadoras en las que estas aplicaciones son ejecutadas.
- El **cliente** es la computadora que se encarga de efectuar una petición o solicitar un servicio. El cliente no posee control sobre los recursos, sino que es el servidor el encargado de manejarlos.
- La computadora remota que actúa como **servidor** evalúa la petición del cliente y decide aceptarla o rechazarla consecuentemente.
- Una vez que el servidor acepta el pedido la información requerida es suministrada al cliente que efectuó la petición, siendo este ultimo el responsable de proporcionar los datos al usuario con el formato adecuado.
- Finalmente debemos precisar que cliente y servidor no tienen que estar necesariamente en computadoras separadas, sino que pueden ser programas diferentes que se ejecuten en la misma computadora.

INTRODUCCIÓN A LAS APLICACIONES DE RED: ARQUITECTURA CLIENTE/SERVIDOR

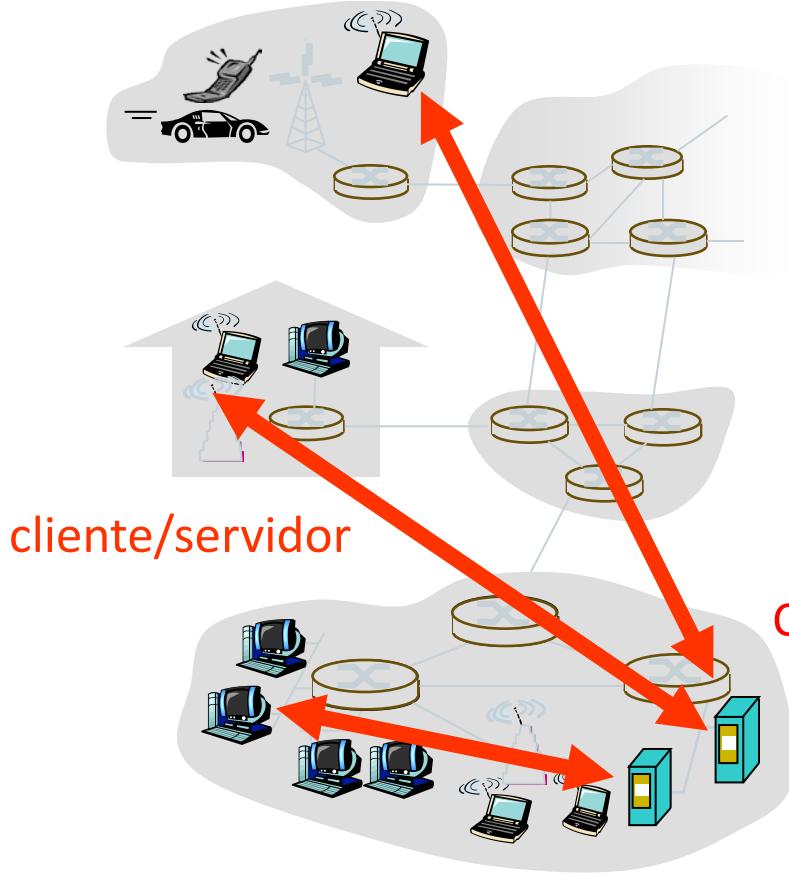
➤ Ventajas de la arquitectura cliente/servidor:

- El modelo cliente/servidor se recomienda, en particular, para redes que requieran un alto grado de fiabilidad. Las principales ventajas son:
 - **Recursos centralizados:** debido a que el servidor es el centro de la red, puede administrar los recursos que son comunes a todos los usuarios.
 - **Seguridad mejorada:** la cantidad de puntos de entrada que permite el acceso a los datos no es importante.
 - **Administración al nivel del servidor:** ya que los clientes no juegan un papel importante en este modelo, requieren menos administración.
 - **Red escalable:** es posible quitar o agregar clientes sin afectar el funcionamiento de la red y sin la necesidad de realizar mayores modificaciones.

INTRODUCCIÓN A LAS APLICACIONES DE RED: ARQUITECTURA CLIENTE/SERVIDOR

- **Desventajas de la arquitectura cliente/servidor:**
 - La arquitectura cliente/servidor también tiene las siguientes desventajas:
 - **Costo elevado:** debido a la complejidad técnica del servidor.
 - **Servidor es el eslabón débil:** el servidor es el único eslabón débil en la red de cliente/servidor, debido a que toda la red está construida en torno a él. Afortunadamente, el servidor es altamente tolerante a los fallos (principalmente gracias al sistema RAID).

INTRODUCCIÓN A LAS APLICACIONES DE RED: ARQUITECTURA CLIENTE/SERVIDOR



Servidor:

- Siempre en funcionamiento
- IP permanente & pública
- Agrupados en “granjas”
- <http://www.xatakandroid.com/mundo-android/la-imagen-de-la-semana-google-muestra-el-corazon-de-internet>
- <https://www.youtube.com/watch?v=zRwPSFpLX8I>

Clientes:

- Funcionando intermitentemente
- Pueden tener IP dinámica & privada
- Se comunican con el servidor
- No se comunican entre sí



PROCESOS CLIENTE Y SERVIDOR

INTRODUCCIÓN A LAS APLICACIONES DE RED: PROCESOS CLIENTE Y SERVIDOR

Proceso Cliente : proceso que inicia la comunicación

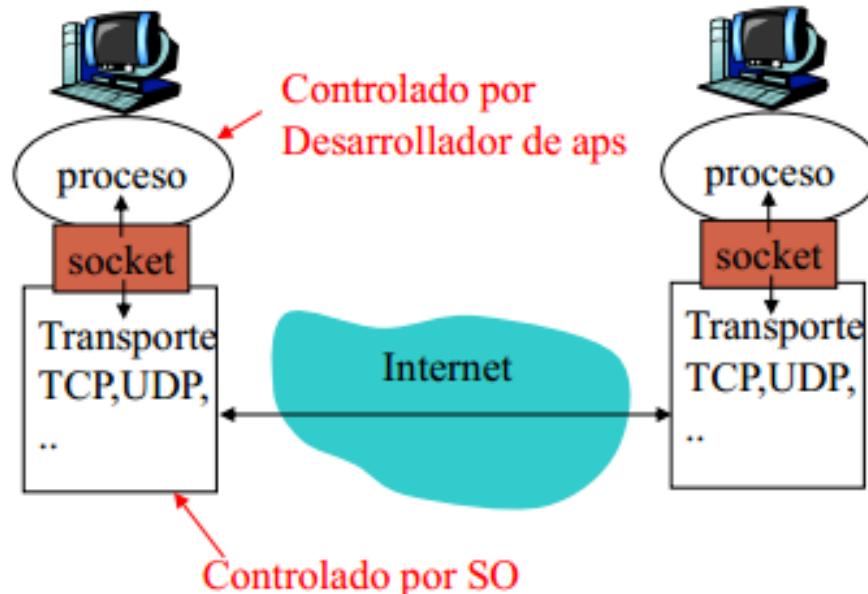
Proceso Servidor: proceso que espera a ser contactado
→IP permanente & pública

➤ Proceso envía/recibe mensajes
a/desde su **socket**

➤ Para recibir mensajes un proceso
debe tener un **identificador**
(IP + puerto)

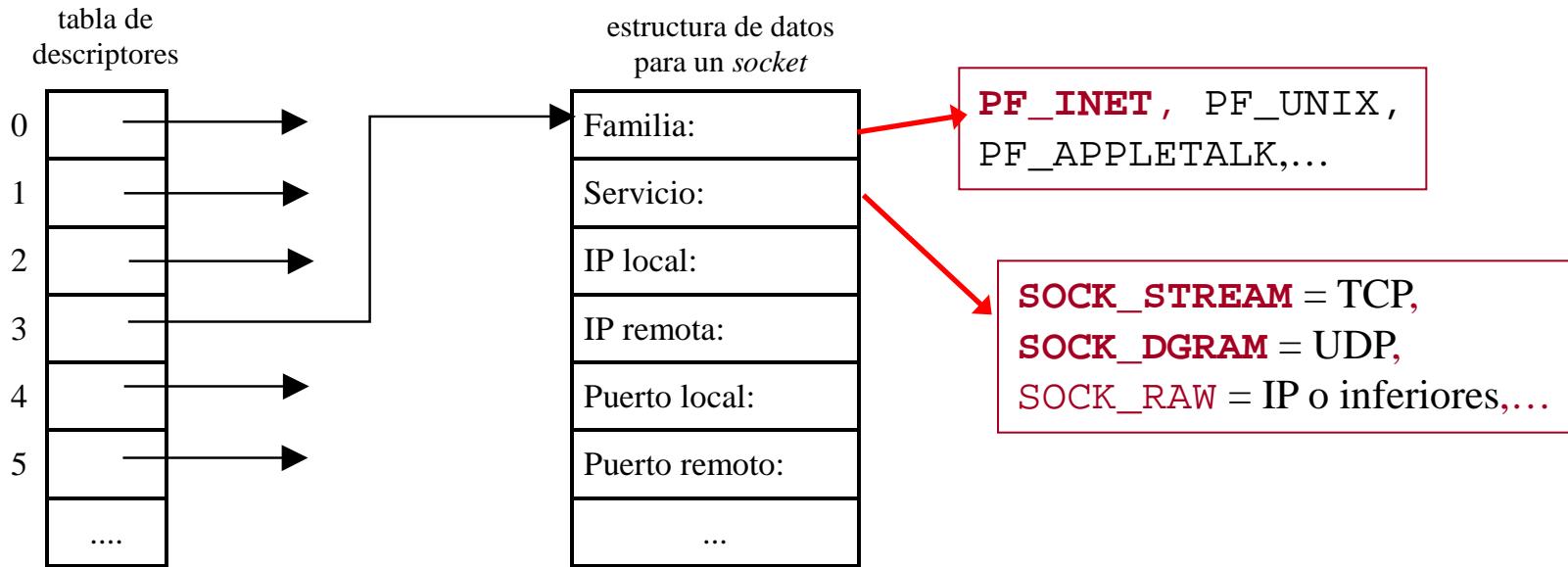
Ej: servidor web **gaia.cs.umass.edu**:
Dirección IP: 128.119.245.12
Número de puerto: 80

Cliente/servidor Cliente/servidor



INTRODUCCIÓN A LAS APLICACIONES EN RED: LA INTERFAZ SOCKET

- Definimos **SOCKET** como un **descriptor** de una transmisión a través del cual la aplicación puede enviar y/o recibir información hacia y/o desde otro proceso de aplicación.
- Es una “**puerta**” de acceso entre la **aplicación** y los servicios de **transporte**.
- En la práctica un **socket** es un **puntero** a una estructura:



INTRODUCCIÓN A LAS APLICACIONES EN RED: LA INTERFAZ SOCKET

Tipos de sockets

- Cada tipo de socket va a definir una serie de propiedades en función de las comunicaciones en las cuales está implicado:
1. La fiabilidad de la transmisión. Ningún dato transmitido se pierde.
 2. La conservación del orden de los datos. Los datos llegan en el orden en el que han sido emitidos.
 3. La no duplicación de datos. Sólo llega a destino un ejemplar de cada dato emitido.
 4. La comunicación en modo conectado. Se establece una conexión entre dos puntos antes del principio de la comunicación (es decir, se establece un circuito virtual). A partir de entonces, una emisión desde un extremo está implícitamente destinada al otro extremo conectado.
 5. La conservación de los límites de los mensajes. Los límites de los mensajes emitidos se pueden encontrar en el destino.
 6. El envío de mensajes (urgentes). Corresponde a la posibilidad de enviar datos fuera del flujo normal, y por consecuencia accesibles inmediatamente (datos fuera de flujo).

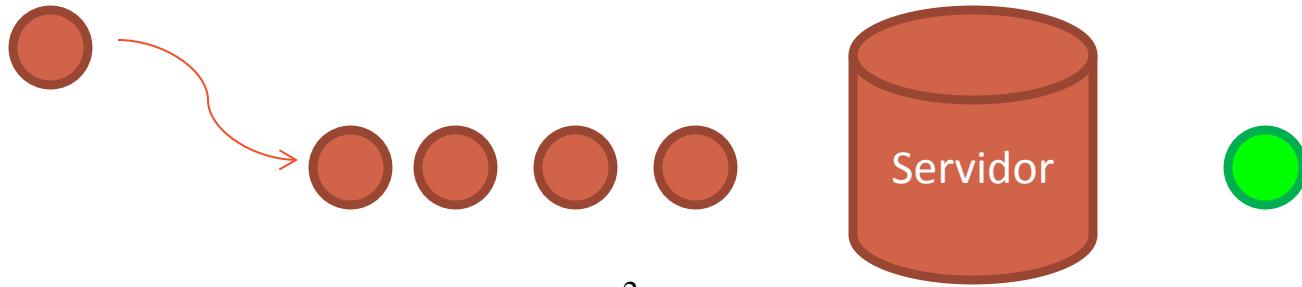
INTRODUCCIÓN A LAS APLICACIONES EN RED: LA INTERFAZ SOCKET

Tipos de sockets

- En cuanto a los tipos de sockets disponibles, se pueden considerar:
- **SOCK_STREAM:** Los sockets de este tipo permiten comunicaciones fiables en modo conectado (propiedades 1, 2, 3 y 4) y eventualmente autorizan, según el protocolo aplicado los mensajes fuera de flujo (propiedad 6).
 - **SOCK_DGRAM:** Corresponde a los sockets destinados a la comunicación en modo no conectado para el envío de datagramas de tamaño limitado. Los datagramas no trabajan con conexiones permanentes (Prot. UDP). La transmisión por los datagramas es a nivel de paquetes, donde cada paquete puede seguir una ruta distinta, no garantizándose una recepción secuencial de la información.
 - **SOCK_RAW:** Permite el acceso a los protocolos de más bajo nivel (por ejemplo, el protocolo IP en el dominio Internet). Su uso está reservado al superusuario.
 - **SOCK_SEQPACKET:** Corresponde a las comunicaciones que poseen las propiedades 1, 2, 3, 4 y 5.
- Los dos tipos de sockets más utilizados son los dos primeros.

INTRODUCCIÓN A LAS APLICACIONES DE RED: RETARDO EN COLA

- Para estimar los retardos (tiempos) en cola se usa la teoría de colas:
- El uso de un servidor se modela con un sistema M/M/1 (ver *TRANSMISIÓN DE DATOS Y REDES DE COMPUTADORES*)



➤ El retardo en cola es:

$$R = \frac{\lambda \cdot (T_s)^2}{1 - \lambda \cdot T_s}$$

donde T_s (distribución exponencial) es el tiempo de servicio y λ (Poisson) la ratio de llegada de solicitudes.

- Esta misma expresión se puede utilizar para calcular el retardo en cola en un router.

INTRODUCCIÓN A LAS APLICACIONES DE RED: ¿QUÉ DEFINEN LOS PROTOCOLOS DE APLICACIÓN?

➤ ¿Qué es y qué define un protocolo?

➤ El tipo de servicio

- Orientado o no orientado a conexión
- Realimentado o no

➤ El tipo de mensaje

ej., request, response,

➤ La sintaxis:

Definición y estructura de “campos” en el mensaje

En aplicación generalmente son orientados a texto (HTTP)

Aunque hay excepciones (DNS)

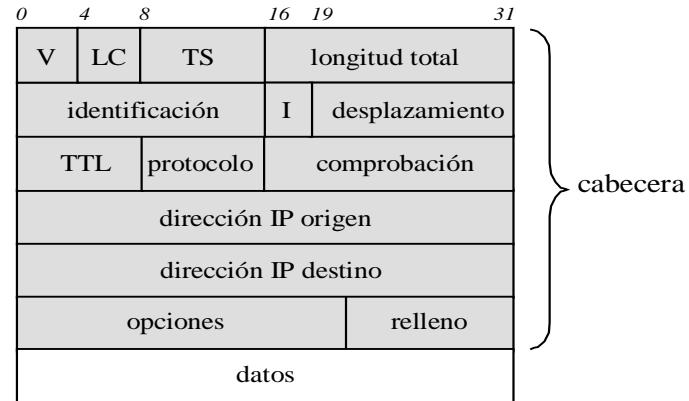
Tendencia para otras capas: usar formato Type-Length-Value

➤ La semántica:

Significado de los “campos”

➤ Las reglas:

Cuándo los procesos envian mensajes/responden a mensajes



INTRODUCCIÓN A LAS APLICACIONES DE RED: ¿QUÉ DEFINEN LOS PROTOCOLOS DE APLICACIÓN?

➤ Tipos de protocolos:

➤ **Protocolos de dominio público** (Definidos en RFCs (ej., HTTP, SMTP)) **vs.** **propietarios** ➤ (ej., Skype, IGRP)

➤ Protocolos in-band vs. out-of-band

➤ **In-band Protocols:** protocolos de red con la que se regula el control de datos.

➤ **Out-of-band protocols: Llamado “Urgent Data” en TCP**) Útil para la separación de dos tipos diferentes de datos. No significa que será entregado más rápido o con mayor prioridad que los datos en el flujo de datos dentro de la banda. (Ej. Protocolos FTP, puertos 20 (Datos) y 21 (Control))

➤ Protocolos stateless vs. state-full:

➤ **Protocolos stateless:** es un protocolo de comunicaciones que trata cada petición como una transacción independiente que no tiene relación con cualquier solicitud anterior, de modo que la comunicación se compone de pares independientes de *solicitud* y *respuesta*. Un protocolo sin estado no requiere que el servidor retenga información de la sesión o de estado acerca de cada socio de las comunicaciones durante la duración de múltiples peticiones.

➤ **Protocolos stateful:** un protocolo que requiere el mantenimiento del estado interno en el servidor se conoce como un protocolo con estado.

➤ **Protocolos persistentes versus no-persistentes:** En una conexión persistente solo se hará una conexión TCP, mientras que en una conexión no persistente se utilizarán múltiples conexiones TCP, una por cada objeto solicitado.

INTRODUCCIÓN A LAS APLICACIONES DE RED: ¿QUÉ DEFINEN LOS PROTOCOLOS DE APLICACIÓN?

- Tendencia: hacer los protocolos **flexibles** con
 - Una cabecera fija
 - Una serie de “trozos” (obligatorios y opcionales)



INTRODUCCIÓN A LAS APLICACIONES DE RED: ¿QUÉ DEFINEN LOS PROTOCOLOS DE APLICACIÓN?

- **Tendencia: hacer los protocolos flexibles con:**
- **Una cabecera fija**
 - **Una serie de “trozos” (obligatorios y opcionales)**
 - Los trozos pueden incluir una cabecera específica más una serie de datos en forma de parámetros:
 - Parámetros fijos: en orden
 - Parámetros de longitud variable u opcionales.
 - Para los parámetros se usa Formato TLV (*Type-Length-Variable*)

0	8	16	31
Tipo de parámetro		Longitud del parámetro	
Valor del parámetro			

INTRODUCCIÓN A LAS APLICACIONES DE RED: CARACTERÍSTICAS

Pérdida de datos (errores)

Algunas aps (ej., audio) pueden tolerar algunas pérdida de datos; otras (ej. FTP, telnet) requieren transferencia 100% fiable

Requisitos temporales

Algunas aps denominadas *inelásticas* (ej., telefonía Internet, juegos interactivos) requieren retardo acotado (delay) para ser efectivas

Ancho de banda (tasa de transmisión o *throughput*)

Algunas aps requieren envío de datos a una tasa determinada

Seguridad

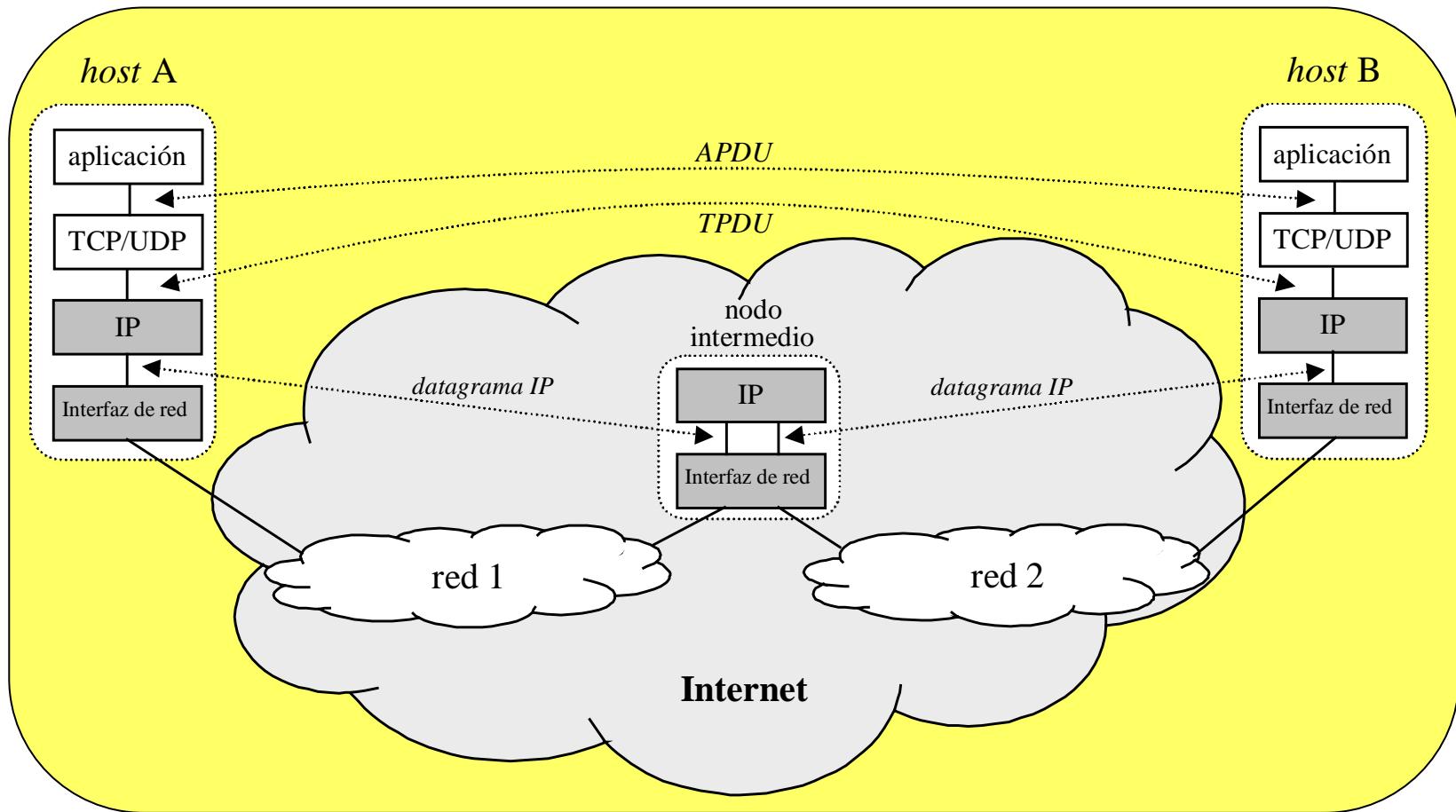
Encriptación, autenticación, no repudio, ...



INTRODUCCIÓN A LAS APLICACIONES DE RED: REQUERIMIENTOS DE ALGUNAS APLICACIONES.

Application	Data loss	Throughput	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web documents	no loss	elastic	no
real-time audio/video	loss-tolerant	audio: 5kbps-1Mbps video:10kbps-5Mbps	yes, 100's ms
stored audio/video	loss-tolerant	same as above	yes, few s
interactive games	loss-tolerant	few kbps up	yes, 100's ms
instant messaging	no loss	elastic	yes and no

INTRODUCCIÓN A LAS APLICACIONES DE RED: PROTOCOLOS DE TRANSPORTE



INTRODUCCIÓN A LAS APLICACIONES DE RED

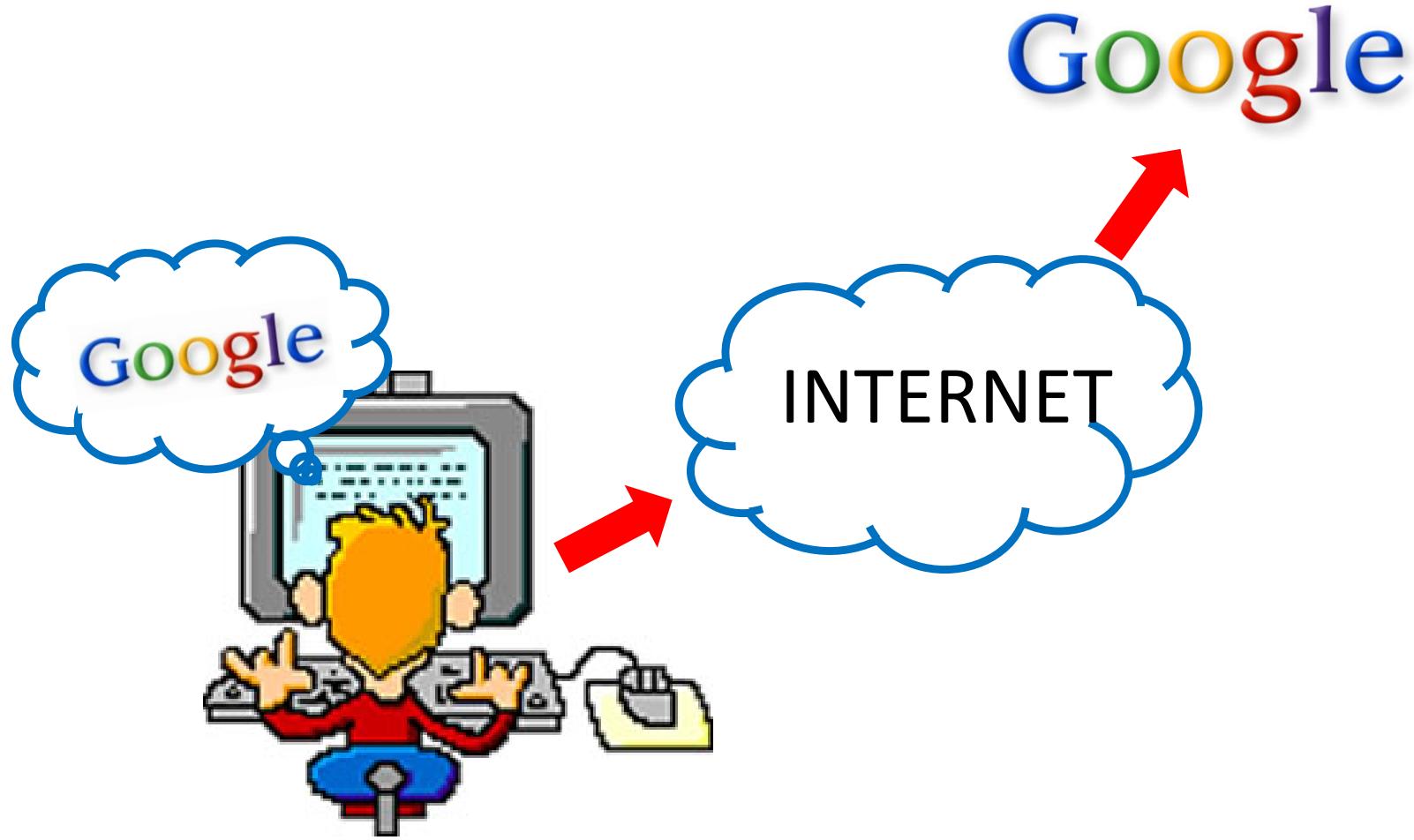
Application	Application layer protocol	Underlying transport protocol
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	HTTP (eg Youtube), RTP [RFC 1889]	TCP or UDP
Internet telephony	SIP, RTP, proprietary (e.g., Skype)	typically UDP



Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red
2. **Servicio de Nombres de Dominio (DNS)**
3. La navegación Web
4. El Correo electrónico
5. Protocolos seguros
6. Aplicaciones multimedia
7. Aplicaciones para interconectividad de redes locales
8. Cuestiones y ejercicios

SERVICIO DE NOMBRES DE DOMINIO (DNS)



SERVICIO DE NOMBRES DE DOMINIO (DNS)

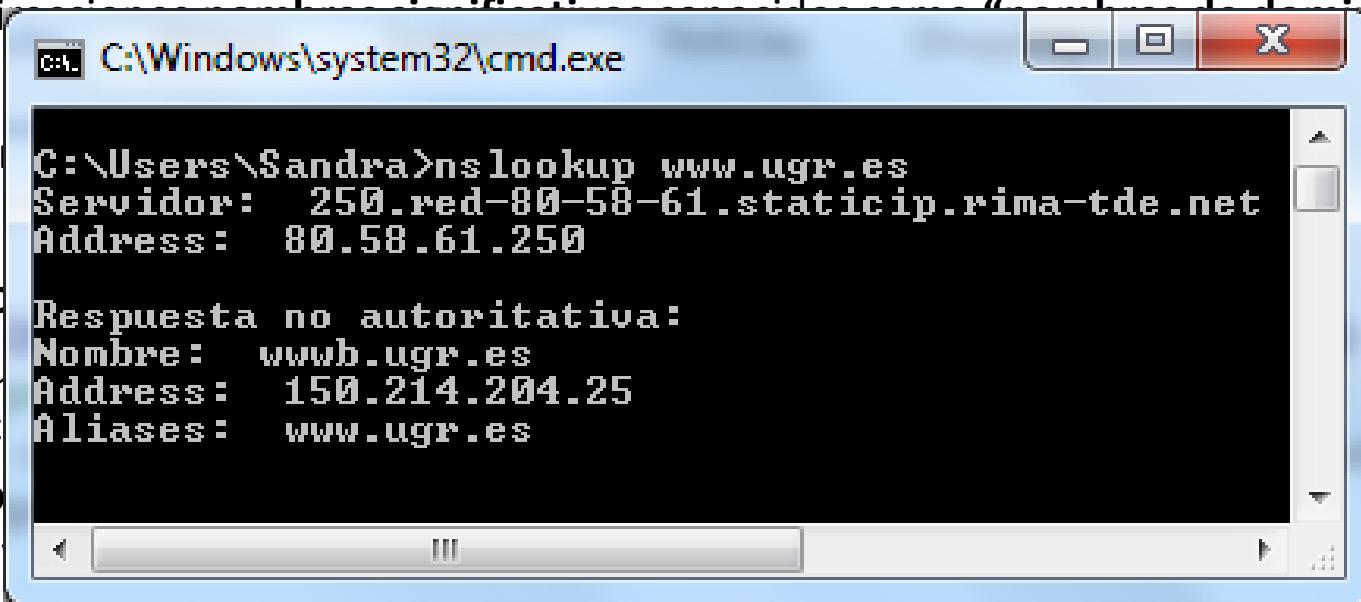
- La comunicación en Internet precisa de direcciones IP.
- Las direcciones IP son difíciles de memorizar o recordar necesitamos asignar a las direcciones **nombres significativos** conocidos como “**nombres de dominio**”.

www.ugr.es <-----> 150.214.204.25

- Este método es inviable y engorroso.
- **En los comienzos de Internet....**
- Se utilizaba una única tabla centralizada de traducción de nombres a direcciones.
- En los 70 el ARPANET estaba formada por unos cientos de máquinas y un sólo archivo, **HOSTS.TXT** que contenía toda la información que se necesitaba sobre esas máquinas.
- El centro de información de red del Departamento de defensa disponía de la versión maestra de la tabla y otros sistemas realizaban una copia regularmente.

SERVICIO DE NOMBRES DE DOMINIO (DNS)

- La comunicación en Internet precisa de direcciones IP.
- Las direcciones IP son difíciles de memorizar o recordar necesitamos asignar a las direcciones un nombre más fácil de recordar “**Nombre**”.
- Este nombre se traduce en una dirección IP.
- **En los servidores de DNS** se almacenan las direcciones IP correspondientes a los nombres de dominio.
- Se usan los servidores para traducir los nombres de dominio en direcciones IP.
- En los servidores de DNS se almacenan las direcciones IP de esas máquinas.
- El centro de información de red del Departamento de defensa disponía de la versión maestra de la tabla y otros sistemas realizaban una copia regularmente.



```
C:\Windows\system32\cmd.exe
C:\Users\Sandra>nslookup www.ugr.es
Servidor: 250.red-80-58-61.staticip.rima-tde.net
Address: 80.58.61.250

Respueta no autoritativa:
Nombre: wwwb.ugr.es
Address: 150.214.204.25
Aliases: www.ugr.es
```



SERVICIO DE NOMBRES DE DOMINIO (DNS)

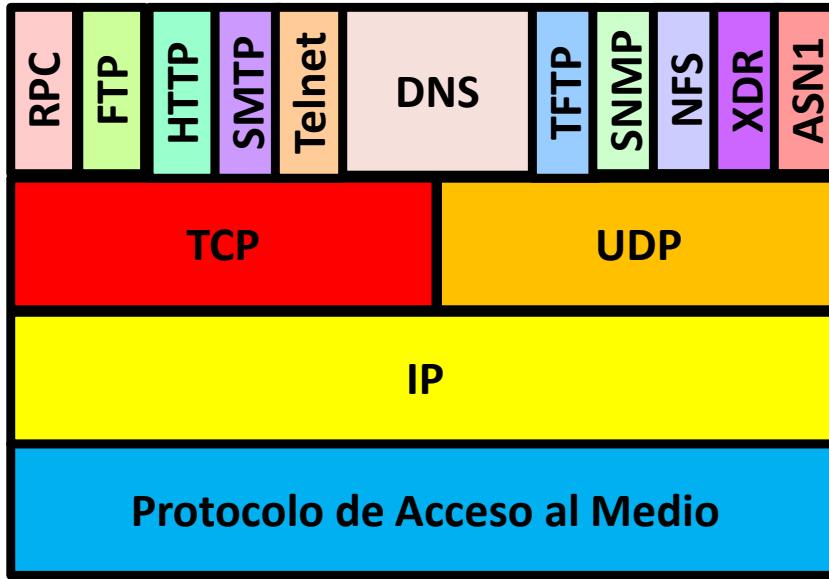
- Cuando Internet creció, con los protocolos TCP/IP este método 'explotó' por diversos motivos:
 - El tráfico y la carga de red para la máquina que contenía las tablas que hacía posible el mapeo era desbordante.
 - La consistencia del archivo era muy difícil de mantener, cuando el *HOST.TXT* llegaba a una maquina muy lejana era ya obsoleto.
 - No se podía garantizar la no duplicidad de nombres (mantener una administración central en una red Internacional era muy complicado).
 - El método **no** era **Escalable**.
- **Conclusión:**
 - El método estaba obsoleto y era muy ineficiente a medida que aumentaron la cantidad de máquinas.
 - Surgió un nuevo sistema de resolución de nombres, **DNS (Domain Name System)** que solucionó los problemas anteriores.

SERVICIO DE NOMBRES DE DOMINIO (DNS)

- El Sistema de Nombres de Dominio es un esquema jerárquico que permite asignar nombres significativos de alto nivel a grandes conjuntos de máquinas y direcciones IP.
- Utiliza una Base de Datos Distribuida para implementarlo.
- Se utiliza un mecanismo Cliente/Servidor, donde unos programas llamados servidores de nombres contienen información acerca de un segmento de la base de datos y la ponen a disposición de los clientes, llamados **resolvers**.
- Los **resolvers** son rutinas que crean preguntas y las envían en forma de paquete UDP a un servidor DNS local. Tipos de nombres en Internet:
 - **Nombres totalmente calificados** (anakena.dcc.uchile.cl)
 - **Nombres parciales** (anakena).
 - Es ilegal usar nombres intermedios (como anakena.dcc) porque si existiera el dominio de primer nivel .dcc ese nombre sería ambiguo.

SERVICIO DE NOMBRES DE DOMINIO (DNS)

- El servicio DNS los podemos situar en la arquitectura de TCP/IP como protocolo de aplicación, tanto sobre UDP como TCP en el puerto 53 de la capa de transporte.

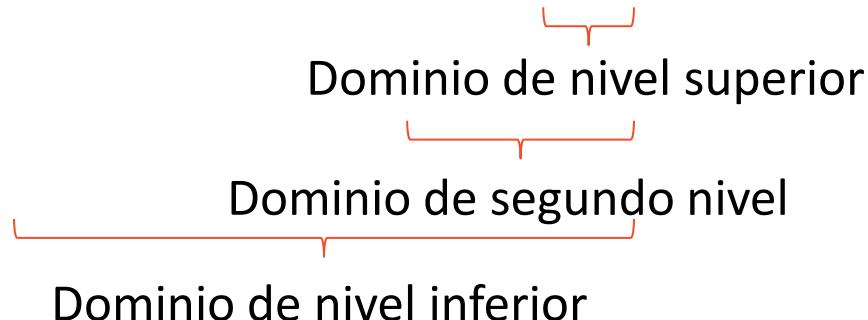


SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ **Nombres de Dominio**

- El DNS se basa en un esquema jerárquico de nombres denominado ***nombre de dominio***.
- Un ***nombre de dominio*** consiste en una secuencia de etiquetas separadas por un punto.

robotica.ugr.es



SERVICIO DE NOMBRES DE DOMINIO (DNS)

Inicialmente fueron definidos los siguientes 9 dominios genéricos (RFC 1591):

.com -> organizaciones comerciales

.edu -> instituciones educativas, como universidades, de EEUU.

.gov -> instituciones gubernamentales estadounidenses

.mil -> grupos militares de estados unidos

.net -> proveedores de Internet

.org -> organizaciones diversas diferentes de las anteriores

.arpa-> propósitos exclusivos de infraestructura de Internet

.int -> organizaciones establecidas por tratados internacionales entre gobiernos

.xy -> indicativos de la zona geográfica (ej. es (España); pt (portugal); jp (Japón)...



➤ Elementos del DNS

1. La sintaxis del nombre
2. La implementación de la base de datos

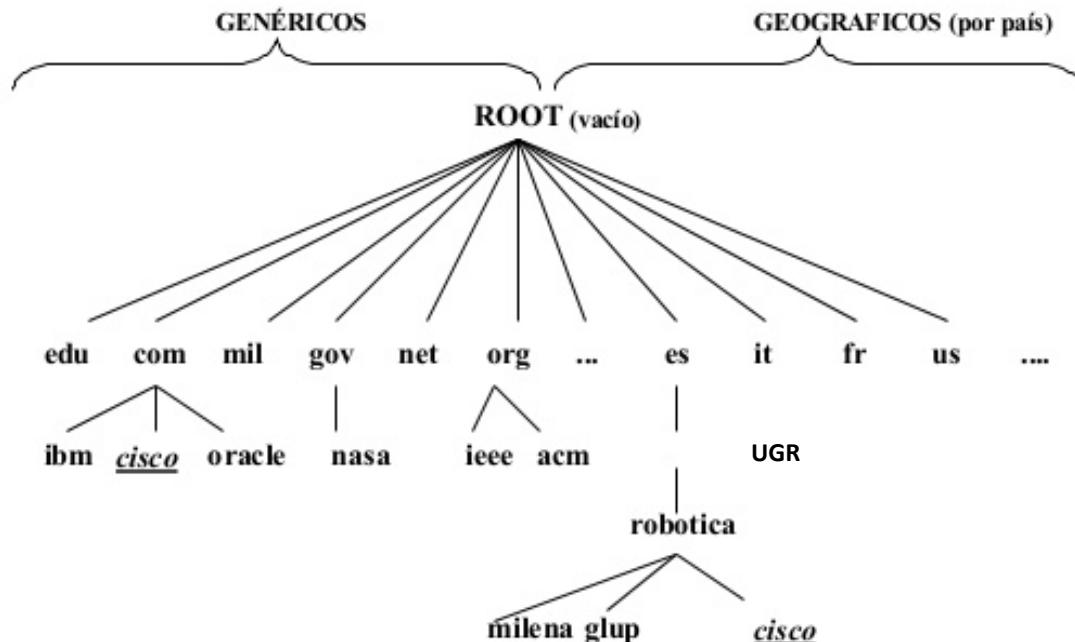
➤ La sintaxis del nombre:

- **Nombre de dominio:** cadena de hasta 255 caracteres, formada por etiquetas separadas por puntos (long. Etiqueta <64 caracteres) de forma jerárquica o por niveles.
- Cada dominio es un índice en la BBDD del DNS.
- No se distinguen mayúsculas de minúsculas.
- Tipos de nombres de dominio:
 - **Absolutos**, terminados con “.” (ej. “ugr.es.”)
 - **Relativos**, no terminados con “.”
- En el nivel raíz, los dominios se clasifican en:
 - **Geográficos**: división por países (o regiones).
 - **Genéricos**: en función del tipo de organización.

SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ La sintaxis del nombre:

- Todos los dominios en Internet pueden representarse mediante un árbol. Las hojas del árbol serían los dominios que ya no contiene más dominios.



SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ **Delegación de la autoridad**

- La organización que posee un nombre de dominio, es **responsable** del funcionamiento y mantenimiento de los servidores de nombres (**zona de autoridad**)
- La solicitud de registro se realiza a una autoridad competente, por ejemplo InterNIC (<http://www.internic.net/>) es una autoridad de registro. Necesario identificar al menos 2 DNS.

MOVISTAR: DNS1: 80.58.0.33 y DNS 2:80.58.32.97

- Solicitar un dominio a una empresa (ej. www.arsys.es) y/o ISP.
- Cada país dispone de autoridades de registro.

SERVICIO DE NOMBRES DE DOMINIO (DNS)



InterNIC

[Home](#) [Registrars](#) [Whois](#) [FAQ](#)

InterNIC—Public Information Regarding Internet Domain Name Registration Services

Do you have a complaint or dispute?

Your Registrar or Domain Name:

- Domain Name Transfer Dispute
- Unsolicited Renewal or Transfer Solicitation
- Your Registrar is Not on the Accredited List
- Unauthorized Transfer of Your Domain Name
- Trademark Infringement
- Registrar Services Dispute
 - Failure to answer phones or respond to email messages
 - Financial Transaction Issues
- Uniform Domain Name Dispute Resolution (UDRP) Intake Report System

Inaccurate Whois Information

Spam or Viruses

IP Address Issues

Content on a Website

Information about Registrars

- Search Accredited Registrars
 - Alphabetical List
 - List by Location
 - List by Language Suffix
- Have a Problem with a Registrar?
 - Complaint Form
 - Helpful Hints

Information about Whois

- Search Whois
- Report Inaccurate Whois

FAQs and Information

- FAQs (ICANN)
- Domain Transfer FAQs
- Explanation of Domain Name Disputes
- Glossary of Terms

The screenshot shows the Arsys website interface. At the top, there's a navigation bar with links for ATENCIÓN 24/7, EMAIL, CHAT, ES, EN, PROFESIONALES, SOPORTE, and ÁREA DE CLIENTE. Below the navigation is a banner featuring a smiling man in an office setting with the text "20 AÑOS DE INNOVACIÓN EN DOMINIOS, HOSTING Y CLOUD" and "DALE LA VUELTA A TU NEGOCIO". To the right of the banner is a sidebar with a "Registra tu dominio" section listing services like "Protege tu marca", "1.500 nuevas extensiones", "Página de bienvenida", and "Gestión DNS". Below the banner is a search bar with the placeholder "www.indicatudominio.com" and a "BUSCAR" button. To the right of the search bar are buttons for ".com" (10€), ".es" (10€), and ".org" (10€). The bottom of the page features a footer with links for DOMINIOS, HOSTING, SERVIDORES CLOUD, and SOLUCIONES CLOUD.



SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ Delegación de la autoridad:

- En una zona existe un **administrador local** que *puede delegar* en otros administradores.

(Ej.) “ugr.es.” puede delegar en el Dep. de TSTC (“tstc.ugr.es.”) para gestionar este dominio inferior.

- Un mismo recurso puede tener asignados varios dominios o nombres registrados, formando *servidores virtuales*.

(Ej.) <http://web1.ugr.es> y <http://www.universidades.org> son 2 servidores de 2 dominios diferentes pero que se pueden asociar a la misma IP.

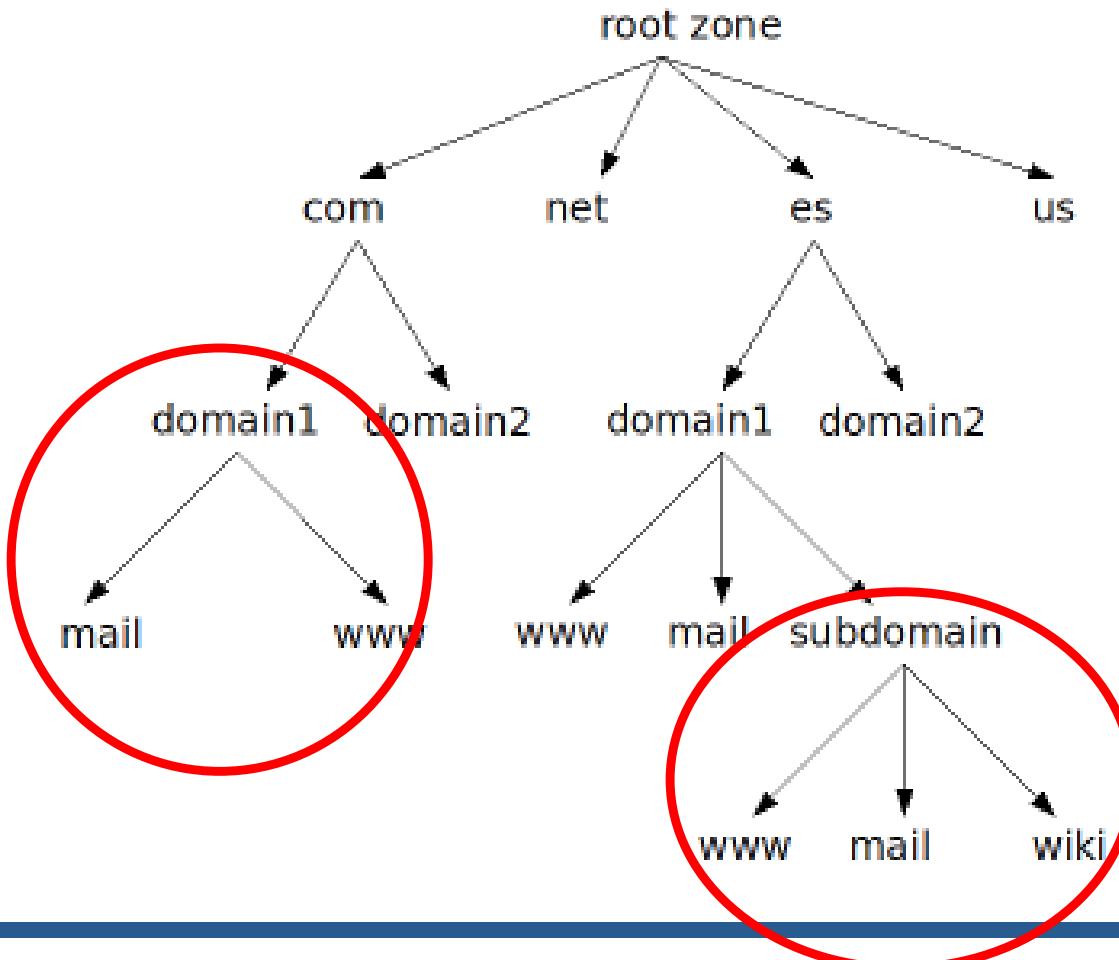
SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ **Delegación de la autoridad:**

- Generalmente, los servidores tienen información completa acerca de una parte del espacio de dominio de nombres (**zona de autoridad**).
- Una **zona** es la porción del espacio de dominio de la que es responsable un determinado servidor DNS.
- **La diferencia entre una zona y un dominio** es que la primera contiene los nombres de dominio y datos que representan a un dominio y un **dominio** es un nombre a que agrupa a otras máquinas o dominios inferiores.

SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ Delegación de la autoridad:





¿Cómo funciona el servicio de DNS?

SERVICIO DE NOMBRES DE DOMINIO (DNS)

- El DNS opera mediante tres componentes principales:
- **Los Clientes DNS:** Programas en los ordenadores de los usuarios que hacen peticiones de resolución de nombres (un navegador web)
 - **Los Servidores DNS:** Son servidores que contestan las consultas realizadas por los Clientes DNS. Los servidores recursivos tienen la capacidad de reenviar la petición a otro servidor si no disponen de la dirección solicitada.
 - **Zonas de Autoridad:** Almacenan datos en los servidores DNS de los dominios. Según las características de la zona, los servidores DNS se pueden clasificar en: **primarios, secundarios, maestros y locales.**

SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ **Tipos de servidores**

- **Primarios (Primary Name Servers):** Almacenan la información de su zona en una base de datos local. Son responsables de mantener la información actualizada y cualquier cambio debe ser notificado a este servidor
- **Secundarios (Secondary Name Servers):** Son aquellos que obtienen los datos de su zona desde otro servidor que tenga autoridad para esa zona. El proceso de copia de la información se denomina *transferencia de zona*.
- **Maestros (Master Name Servers):** son los que transfieren las zonas a los servidores secundarios. Cuando un servidor secundario arranca busca un servidor maestro y realiza la transferencia de zona. Un servidor maestro para una zona puede ser a la vez un servidor primario o secundario de esa zona. Estos servidores extraen la información desde el servidor primario de la zona. Así se evita que los servidores secundarios sobrecargen al servidor primario con transferencias de zonas.

SERVICIO DE NOMBRES DE DOMINIO (DNS)

➤ **Tipos de servidores**

➤ **Locales (*Caching-only servers*)**: no tienen autoridad sobre ningún dominio: se limitan a contactar con otros servidores para resolver las peticiones de los clientes DNS. Estos servidores mantienen una *memoria caché* con las últimas preguntas contestadas. Cada vez que un cliente DNS le formula una pregunta, primero consulta en su memoria caché. Si encuentra la dirección IP solicitada, se la devuelve al cliente; si no, consulta a otros servidores, apuntando la respuesta en su memoria caché y comunicando la respuesta al cliente.



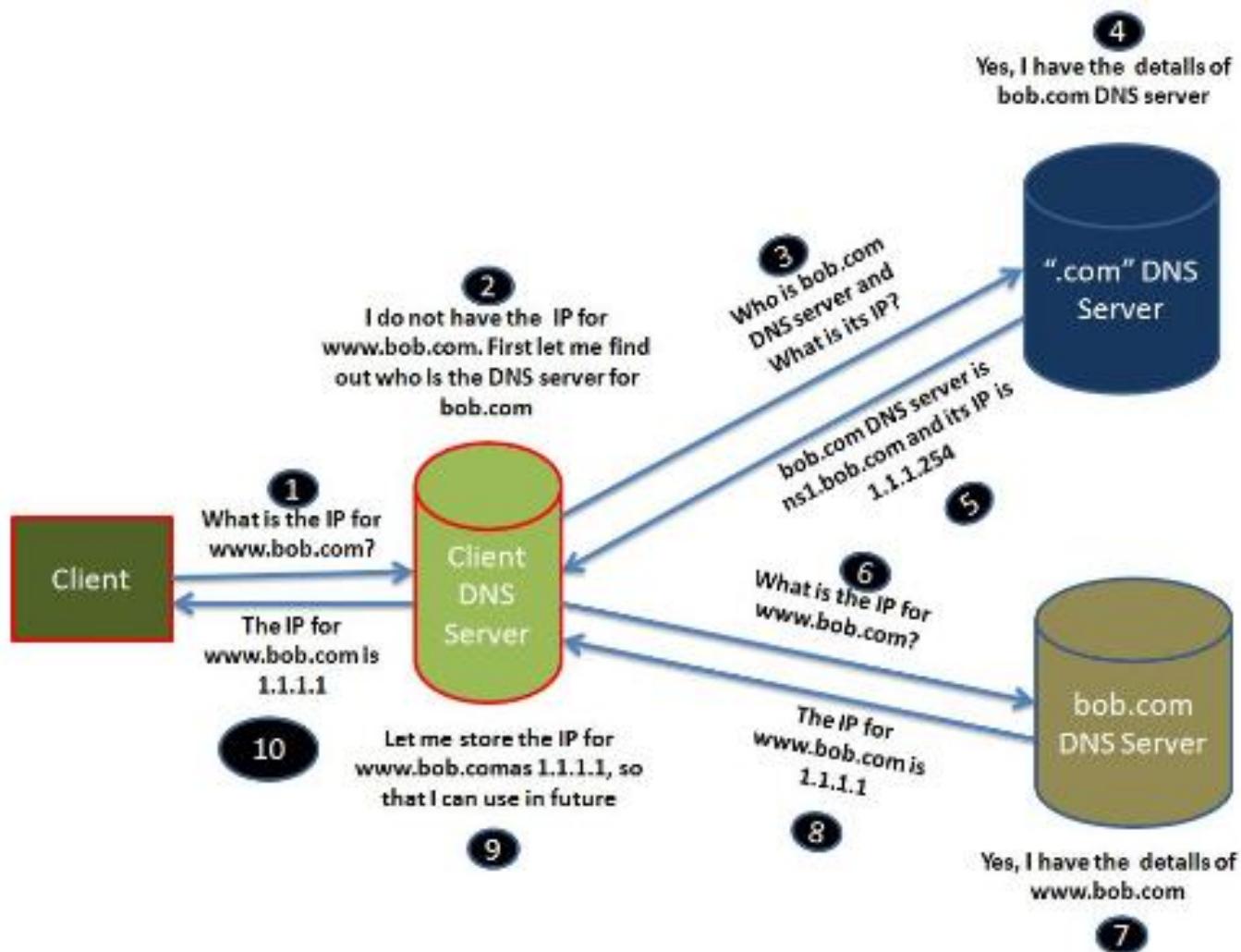
➤ Proceso de resolución de nombres de dominio

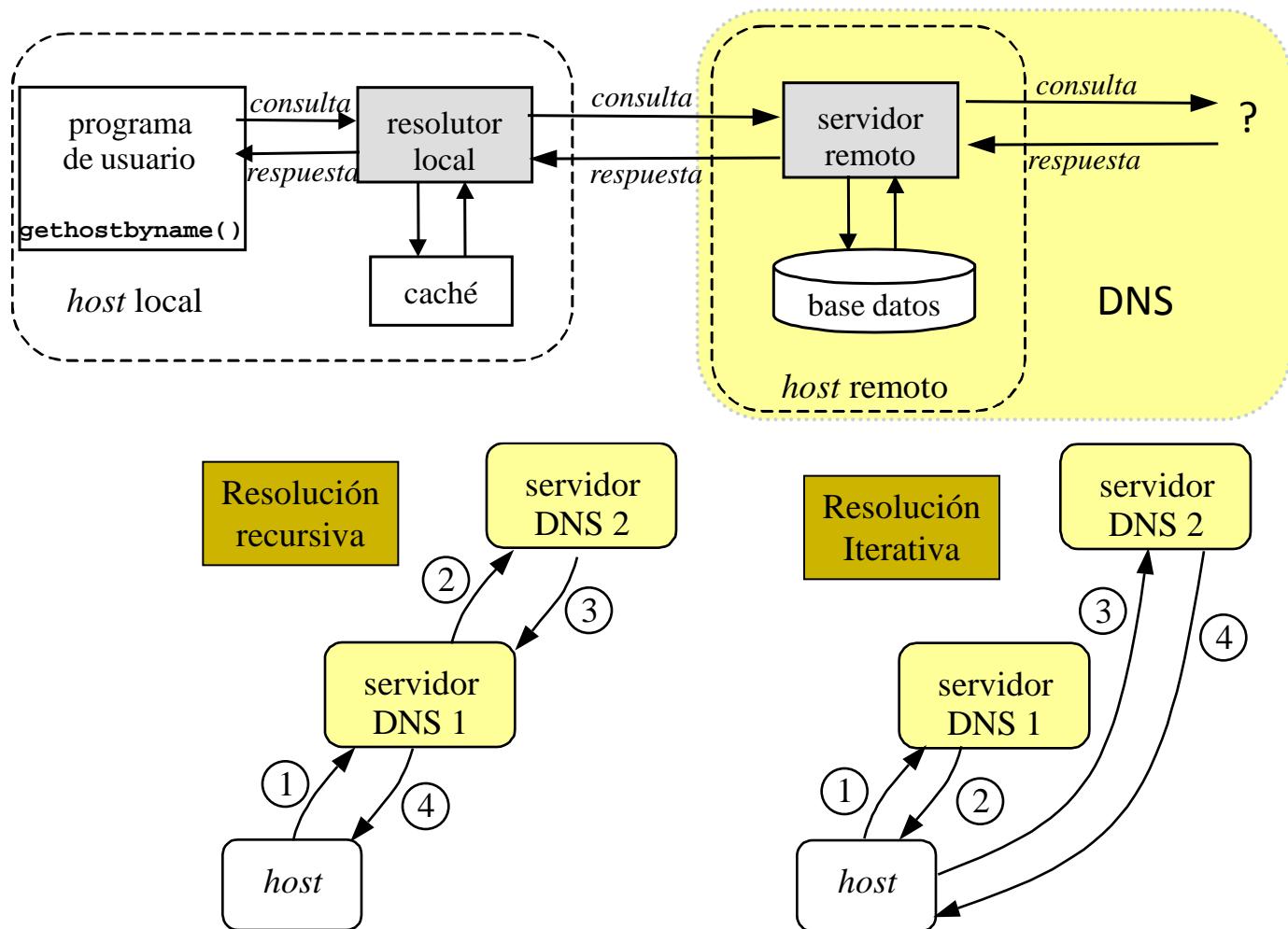
- En este proceso hay dos partes el cliente “resolver” y el servidor que es el DNS.
- Las funciones del cliente son:
 - Interrogar al servidor DNS
 - Interpretar las respuestas que pueden ser registros de recursos (RR) o errores
 - Devolver la información al programa que realiza la petición al cliente DNS
- Para este proceso de traducción los clientes pueden formular dos tipos de preguntas: recursivas o iterativas.

➤ **Tipo de preguntas formuladas por los clientes DNS**

- **Recursiva:** obliga al servidor DNS a que responda aunque tenga que consultar a otros servidores. *Esta opción es más frecuente.*
- **Iterativa:** el servidor contesta si tiene la información y si no, le remite la dirección de otro servidor capaz de resolver. De esta forma el cliente tiene mayor control sobre el proceso de búsqueda. *Esta opción es menos frecuente.*
- **Híbrido:** es una combinación de las 2 anteriores.

SERVICIO DE NOMBRES DE DOMINIO (DNS)





➤ Gestión de la base de datos DNS:

- Cada zona debe tener **al menos** un servidor de autoridad.
- En cada zona hay servidores **primarios** (copia master de la db) y **secundarios** (obtiene la db por transferencia)
- Además, existe un servicio de **cache** para mejorar prestaciones.
- La **topología real** de servidores es complicada: existe **13 servidores raíz (A-M)** (ver <http://www.root-servers.org>)
- El root-server F (y otros) tiene un servidor en Madrid (**Espanix: punto neutro**)

➤ Gestión de la base de datos DNS:

- Cuando un cliente (a través de un resolver local) solicita una resolución de nombres a su servidor puede ocurrir:
 - **Respuesta CON autoridad:** el servidor tiene autoridad sobre la zona en la que se encuentra el nombre solicitado y devuelve la dirección IP.
 - **Respuesta SIN autoridad:** el servidor no tiene autoridad sobre la zona en la que se encuentra el nombre solicitado, pero lo tiene en la cache.
 - **No conoce la respuesta:** el servidor preguntará a otros servidores de forma recursiva o iterativa. Normalmente se “eleva” la petición a uno de los servidores raíz.

Root-servers <http://www.root-servers.org/>

Servidor A: Network Solutions, Herndon, Virginia, USA.

Servidor B: Instituto de Ciencias de la Información de la Universidad del Sur de California, USA.

Servidor C: PSINet, Virginia, USA.

Servidor D: Universidad de Maryland, USA.

Servidor E: NASA, en Mountain View, California, USA.

Servidor F: Internet Software Consortium, Palo Alto, California, USA.

Servidor G: Agencia de Sistemas de Información de Defensa, California, USA.

Servidor H: Laboratorio de Investigación del Ejercito, Maryland, USA.

Servidor I: NORDUnet, Estocolmo, Suecia.

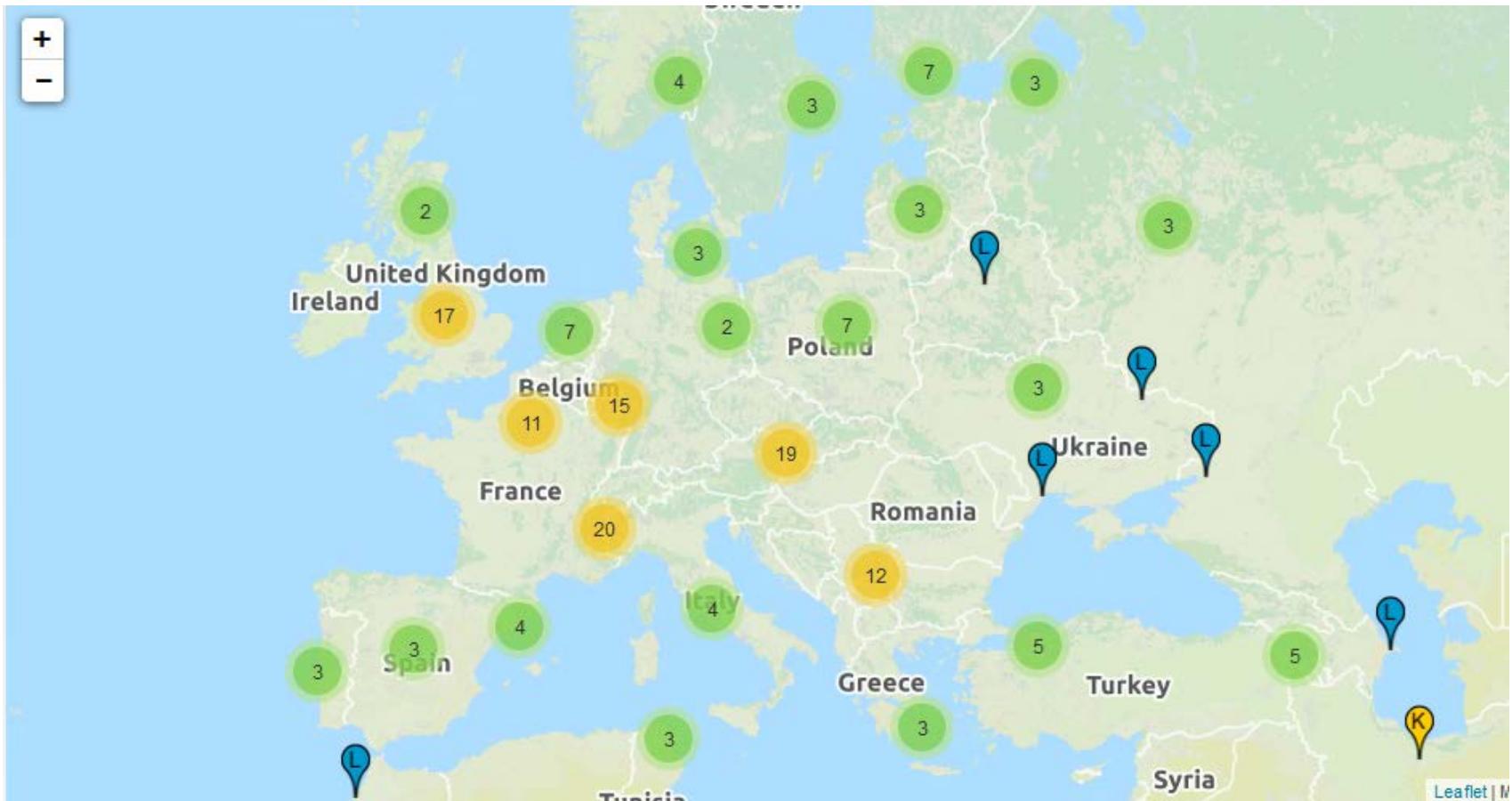
Servidor J: (TBD), Virginia, USA.

Servidor K: RIPE-NCC, Londres, Inglaterra.

Servidor L: (TBD), California, USA.

Servidor M: Wide Project, Universidad de Tokyo, Japón.







¿Cómo son las bases de datos que manejan esta información?

➤ ¿Cómo es la base de datos DNS?

- Todo dominio está asociado al menos a un registro **Resource Record (Registro de recursos)**
- El sistema DNS es una base de datos distribuida en la que cada servidor guarda una serie de registros. El formato de los registros es de la siguiente forma:

[Nombre_dominio] [TTL] [Clase] Tipo Dato_Registro(Valor)

- Cuando un cliente (*resolver*) da un nombre de dominio al DNS, lo que recibe son los RR asociados a ese nombre y por tanto la función real del DNS es relacionar los dominios de nombres con los RR.
- Normalmente existen muchos RR por dominio.

➤ ¿Cómo es la base de datos DNS?

- ***Nombre_dominio***: puede haber más de un registro por dominio, también conocido como recurso. Este campo a veces puede omitirse, tomando por defecto el último nombre de dominio indicado con anterioridad.
- ***TTL***: tiempo de vida para almacenarse, indicando la estabilidad del registro. Información altamente estable tiene un valor grande (86400 seg. o un día), mientras que la volátil recibe un valor pequeño (60 seg.).
- ***Clase***: Actualmente sólo se utiliza *IN*, para información de Internet. Este campo si se omite, se toma el último valor indicado con anterioridad.
- ***Dato_Registro(valor)*** es un número o texto ascii dependiendo del tipo de registro.

➤ Los tipo de registros más utilizados son:

Tipo de Registro	Descripción
SOA <i>Start Of Authority</i>	Inicio de autoridad, identificando el dominio o la zona. Fija una serie de parámetros para esta zona.
NS <i>Name Server</i>	El nombre de dominio se hace corresponder con el nombre de una computadora de confianza para el dominio o servidor de nombres.
A <i>Address</i>	Dirección IP de un host en 32 bits. Si este tiene varias direcciones IP, multihomed, habrá un registro diferente por cada una de ellas.
CNAME	Es un alias que se corresponde con el nombre canónico verdadero.
MX	Se trata de un intercambiador de correo (Mail eXchanger), es decir, un dominio dispuesto a aceptar solo correo electrónico.
TXT	Texto, es una forma de añadir comentarios a la Base de Datos. Por Ej., para dar la dirección postal del dominio.
PTR	Apuntador, hace corresponder una dirección IP con el nombre de un sistema. Usado en archivos dirección-nombre, la inversa del tipo A.
HINFO	Información del Host, tipo y modelo de computadora.
WKS	Servicios públicos (Well-Known Services). Puede listar los servicios de las aplicaciones disponibles en el ordenador.

➤ Los tipo de registros más utilizados son:

Registros MX

- ***Mail Exchanger***: son servidores de correo ordenados por prioridad en un dominio y registrados en el DNS, de forma que en caso de fallo del principal, generalmente el que tendrá información de todas las cuentas de correo de los usuarios, el cliente de correo (quien quiere realizar la entrega) averiguará a través del DNS el MX del dominio, quien recibirá el correo en nombre del principal.
- Este MX intermedio, no requiere tener configuradas las cuentas de correo y en el momento que el principal se reponga, el MX hará entrega de los correos.

➤ Preguntas inversas:

- Existe una base de datos asociada de **resolución inversa** para traducir direcciones IP en nombres de dominio. (*in-addr.arpa*)
- Los clientes DNS también pueden formular *preguntas inversas*, es decir, a partir de una IP conocer un nombre de dominio.
- Para evitar una búsqueda exhaustiva por todo el espacio de nombres de dominio, se ha creado un dominio especial llamado *in-addr.arpa*.
- Cuando un cliente DNS desea conocer el nombre de dominio asociado a la dirección IP *w.x.y.z* formula una pregunta inversa a *z.y.x.w.in-addr.arpa*.
- La inversión de los bytes es necesaria debido a que los nombres de dominio son más genéricos por la derecha, al contrario que ocurre con las direcciones.

➤ Formato de los mensajes DNS

- Los mensajes de consulta y respuesta intercambiados entre los clientes y los servidores del DNS tienen un formato muy sencillo. Un servidor añade la información requerida a la consulta original y la envía de vuelta. El formato general del mensaje se muestra a continuación:

Cabecera.
Consulta (o consultas).
(En la respuesta) RR de respuesta.
(En la respuesta) RR que identifican servidores con autorización.
(En la respuesta) RR con información adicional.

- **Formato de los mensajes DNS**
- La cabecera contiene los siguientes campos:

Campo	Descripción
ID	Identificador para hacer corresponder una respuesta con su petición
Parámetros	Consulta o respuesta. Consulta normal o inversa. En respuestas, si es de confianza. En respuestas, si está truncada. Recursivo o no. En respuestas, si la recursión está disponible. En respuestas, código de error.
Num. De consultas	Proporcionado en una consulta y en una respuesta.
Num. de respuestas	Proporcionado en una respuesta
Num. De registros de autoridad	Proporcionado en una respuesta. La información de los registros de autoridad incluye los nombres de los servidores que contienen los datos de confianza.
Num. De registros adicionales	Proporcionado en una respuesta. La información incluye las direcciones de los servidores de confianza

➤ Formato de los mensajes DNS

- La sección de consulta contiene los campos que se muestran en la tabla siguiente. Normalmente, un mensaje contiene una única consulta, pero se permite concatenar varias peticiones en la sección de consulta:

Campo	Descripción
Nombre	Nombre de dominio o dirección de IP en el subárbol IN-ADDR.ARPA
Tipo	Tipo de consulta, por ejemplo A o NS.
Clase	IN para Internet, se representa como 1.

➤ Formato de los mensajes DNS

- Las secciones de respuesta, información de autoridad e información adicional se estructuran todas ellas de la misma forma. Consisten en una secuencia de registros de recurso que contienen los campos que se muestran en la tabla siguiente:

Campo	Descripción
Nombre	Nombre del nodo para este registro.
Tipo	Tipo de registro, como SOA o A, indicado por un código numérico.
Clase	IN, se representa como 1.
TTL	Tiempo de vida, un entero con signo de 32 bits que indica cuánto tiempo el registro puede permanecer en la caché.
RDLENGTH	Tamaño del campo de datos de recursos
RDATA	La información, por ejemplo, para un registro de direcciones, es la dirección IP. Para un registro SOA, incluye información más extensa



Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red
2. Servicio de Nombres de Dominio (DNS)
- 3. La navegación Web**
4. El Correo electrónico
5. Protocolos seguros
6. Aplicaciones multimedia
7. Aplicaciones para interconectividad de redes locales
8. Cuestiones y ejercicios



Introducción

LA NAVEGACIÓN WEB

- La WWW (World Wide Web) es la aplicación más importante en Internet.
- WWW es un sistema de distribución de información basado en hipertexto o hipermedios enlazados y accesibles a través de Internet.
- Con un navegador web, un usuario visualiza sitios web compuestos de páginas web que pueden contener texto, imágenes, videos u otros contenidos multimedia, y navega a través de ellas usando hiperenlaces.
- Ha contribuido a su espectacular crecimiento los siguientes factores:
 - Presentación atractiva
 - Fácil de usar
 - Interface unificado para todos los servicios
 - Permite de manera flexible e interactiva acceder a grandes cantidades de información
- Las interfaces de los clientes Web son muy intuitivas, y cualquier persona se hace con su funcionamiento rápidamente.



LA NAVEGACIÓN WEB

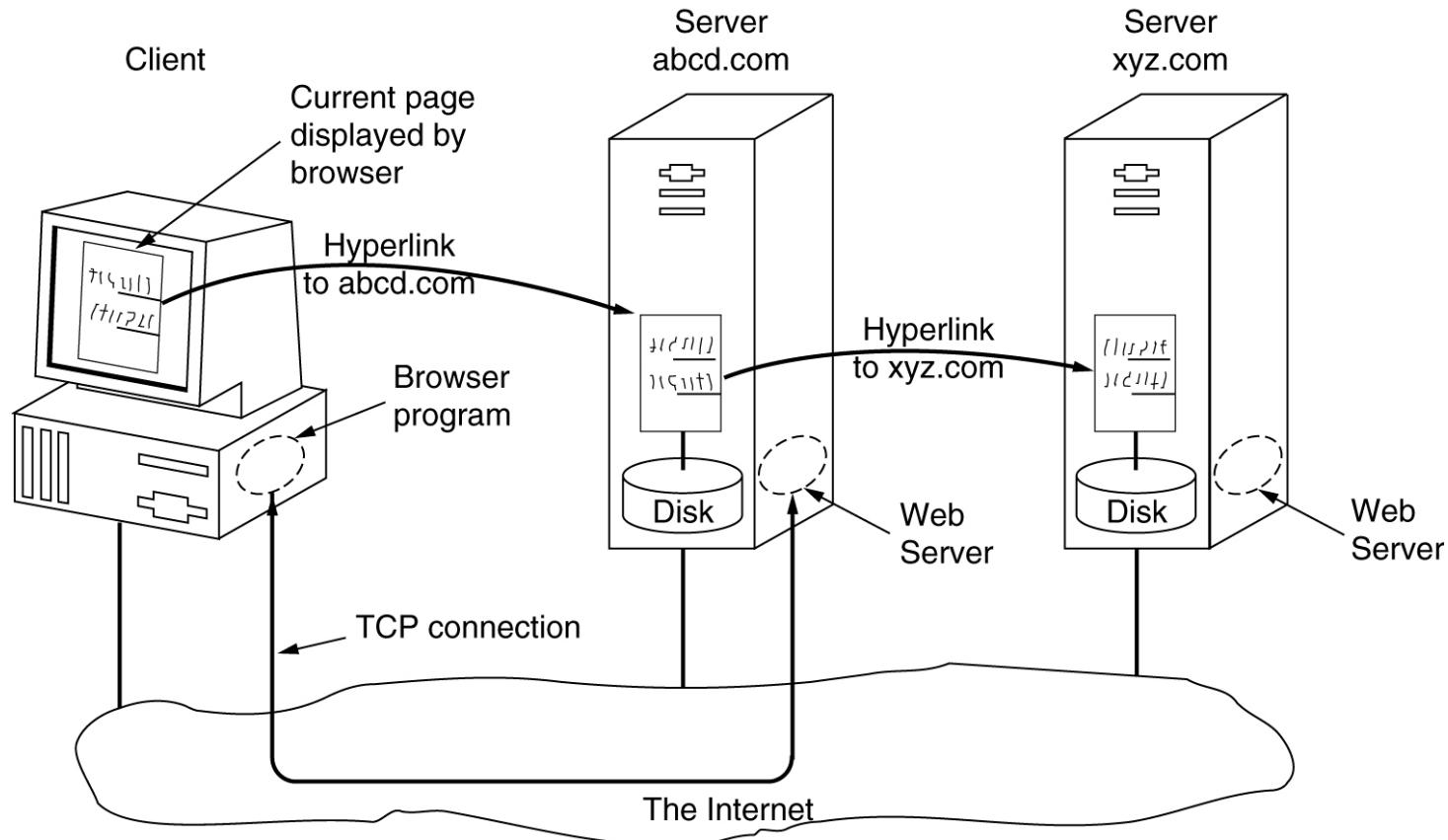
- Por su flexibilidad, puede dar soporte a multitud de servicios diferentes, por lo que resulta interesante para empresas, centros de enseñanza, etc. A través del Web se puede acceder a servicios comerciales y publicitarios, cursos y guías, bases de datos,...
- Es muy fácil publicar nueva información, así como incorporar información en formato electrónico de la que se disponía previamente, para hacerla accesible a todo el mundo.
- Está en continua evolución, y cada día sus capacidades de acceso y representación de información se vuelven más sofisticadas.
- Ha sido la principal causa del espectacular crecimiento que Internet ha tenido en los últimos cuatro años, tanto en el número de usuarios como en el volumen de información disponible.
- También sirve como soporte para las denominadas "Intranets".

LA NAVEGACIÓN WEB

- Tim Berners-Lee dirige el World Wide Web Consortium (W3C), la organización que coordina estándares y añade nuevas funcionalidades y desarrollos a la Web.
- W3C se fundó en 1994, a raíz de las negociaciones entre los países que conformaban el CERN y las instituciones estadounidenses interesadas en el proyecto web.
- W3C tiene su sede en Boston (Massachusetts) y en él participan el CERN, el Institut National de Recherche en Informatique et en Automatique de Francia (INRIA) y el Massachusetts Institute of Technology (MIT).

- Se destacan los siguientes estándares:
 - el Identificador de Recurso Uniforme (URI), que es un sistema universal para referenciar recursos en la Web, como páginas web,
 - el Protocolo de Transferencia de Hipertexto (HTTP), que especifica cómo se comunican el navegador y el servidor entre ellos,
 - el Lenguaje de Marcado de Hipertexto (HTML), usado para definir la estructura y contenido de documentos de hipertexto,
 - el Lenguaje de Marcado Extensible (XML), usado para describir la estructura de los documentos de texto.
- Tim Berners-Lee dirige desde 2007 el World Wide Web Consortium (W3C), el cual desarrolla y mantiene esos y otros estándares que permiten a los ordenadores de la Web almacenar y comunicar efectivamente diferentes formas de información.

LA NAVEGACIÓN WEB

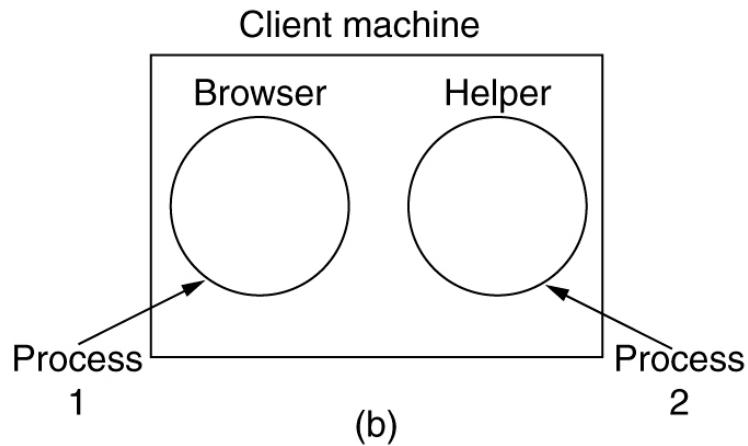
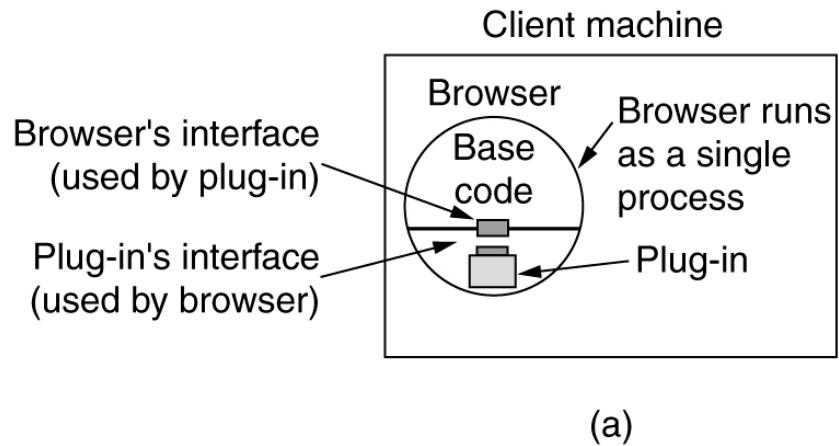


Partes del modelo web.



Cliente web

- Un cliente web (también llamado navegador) es esencialmente un programa que puede abrir páginas Web.
- Sirve para acceder a la www y “navegar” por ella a través de los enlaces.
- Cuando seleccionamos un elemento el navegador sigue el hipervínculo y obtiene la página seleccionada.
- Tiene soporte para imágenes, sonidos y videos.
- No todas las páginas contienen HTML, las hay que pueden tener un documento PDF, un icono GIF, un vídeo en MPEG... El servidor indica el tipo MIME. Si el tipo MIME no es de los integrados hay dos posibilidades:
 - Plug-in
 - Aplicaciones auxiliares.



(a) plug-in del navegador.

(b) Aplicación ayuda.

LA NAVEGACIÓN WEB

- ✓ Los navegadores son sofisticadas aplicaciones que se encargan de recoger y mostrar la información de los servidores Web.
- Ofrecen al usuario la posibilidad de activar una URL, seleccionando un enlace de un documento o introduciéndola directamente.
 - Descodifican los campos de una URL.
 - Se conectan con el servidor HTTP correspondiente para recoger el contenido de la URL.
 - Interpretan el hipertexto y lo muestran, adecuándose a las características y limitaciones del entorno en que se ejecuta el cliente.
 - Recogen el resto de los elementos que componen una página Web, como imágenes, sonidos, aplicaciones Java, objetos insertados,...
 - Además, suelen disponer de utilidades que simplifican muchas operaciones comunes: copias temporales de las páginas y direcciones visitadas recientemente, agendas de URLs, clientes de correo electrónico, etc..

`http://www.epsg.upv.es/historia.php?modo=presentacion&titulo=Hist%F2ria`

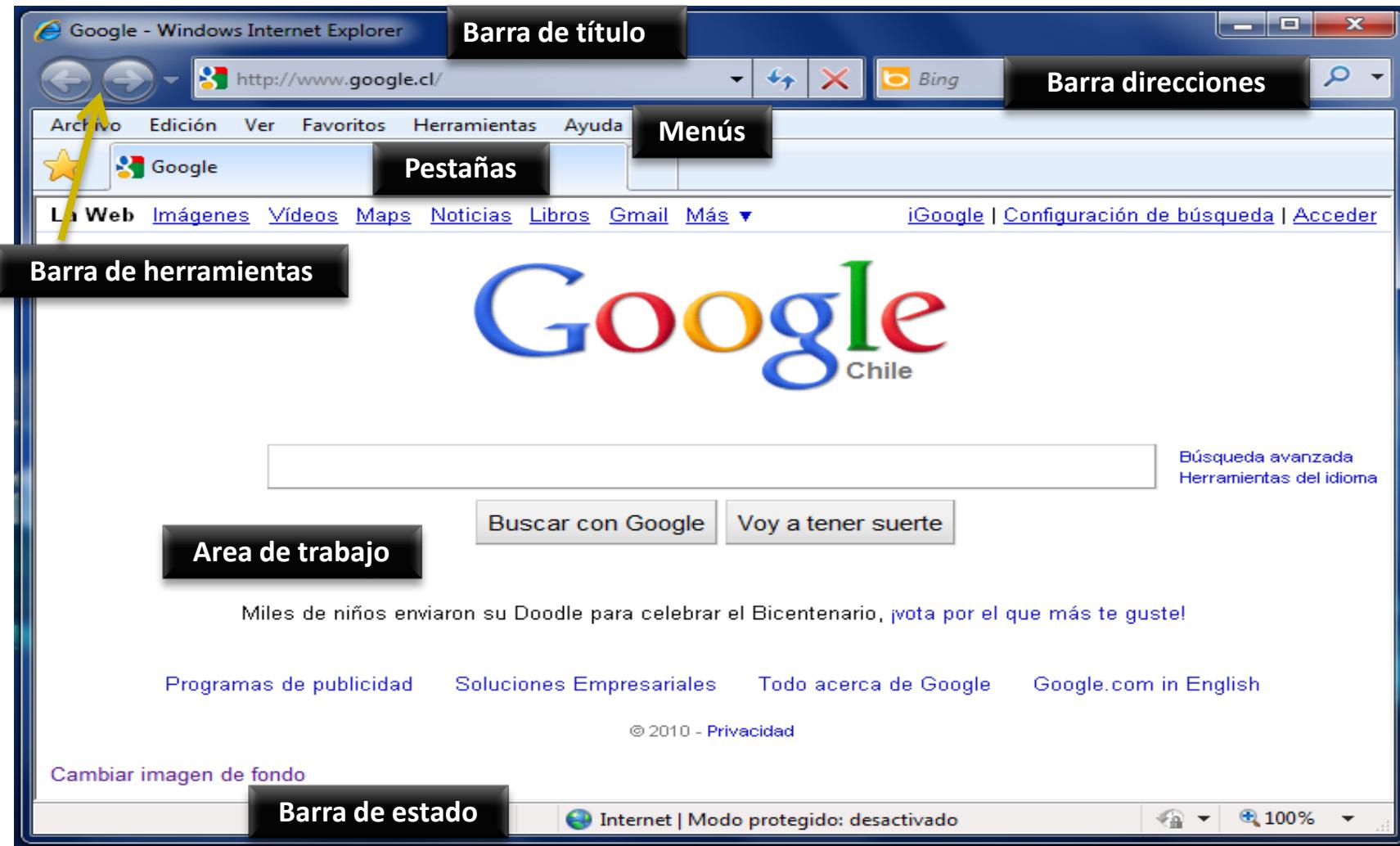
1. El navegador determina el URL
2. Accede al servicio DNS para averiguar la dirección IP de www.epsg.upv.es
3. DNS contesta 158.42.144.1
4. El navegador se conecta al puerto TCP, 80 de 158.42.144.1
5. Y envía “GET /historia.php?modo=presentacion&titulo=Hist%F2ria”
6. El servidor manda el fichero *historia.php*
7. Se cierra la conexión
8. El navegador visualiza el contenido de *historia.php*
9. Si existen imágenes, el navegador las obtiene y las presenta en pantalla.

LA NAVEGACIÓN WEB

- ✓ Permiten acceder a servicios ftp, correo, etc.
- ✓ Permiten almacenar la información en el disco o crear marcadores (bookmarks) de las páginas web mas visitadas.

Name	Used for	Example
http	Hypertext (HTML)	http://www.cs.vu.nl/~ast/
ftp	FTP	ftp://ftp.cs.vu.nl/pub/minix/README
file	Local file	file:///usr/suzanne/prog.c
news	Newsgroup	news:comp.os.minix
news	News article	news:AA0134223112@cs.utah.edu
gopher	Gopher	gopher://gopher.tc.umn.edu/11/Libraries
mailto	Sending e-mail	mailto:JohnUser@acm.org
telnet	Remote login	telnet://www.w3.org:80

LA NAVEGACIÓN WEB



Navegadores web más conocidos:

- **Mozilla Firefox**
- **Google Chrome**
- **Amaya**
- **Epiphany**
- **Galeon**
- **Internet Explorer sobre Windows**
- **Konqueror sobre linux**
- **Lynx sobre linux**
- **Netscape Navigator**
- **Opera**
- **Safari**
- **Seamonkey**
- **Shiira**
- **Flock**
- **Arora**
- **K-Meleon**
- **Orca Browser**
- **Avant Browser**



Servidor web

LA NAVEGACIÓN WEB

- ✓ Es un ordenador que provee los datos solicitados por parte de los navegadores de otras computadoras.
- ✓ Se entiende como el software que configura un PC como servidor para facilitar el acceso a la red y sus recursos.
- ✓ Los Servidores almacenan información en forma de páginas web y a través del protocolo HTTP lo entregan a petición de los clientes (navegadores web) en formato HTML.
- ✓ Un servidor www (web site) espera conexiones en el puerto 80 TCP.
- ✓ Tras establecer una conexión espera una petición por parte del cliente y contesta con la información solicitada (protocolo HTTP).



LA NAVEGACIÓN WEB

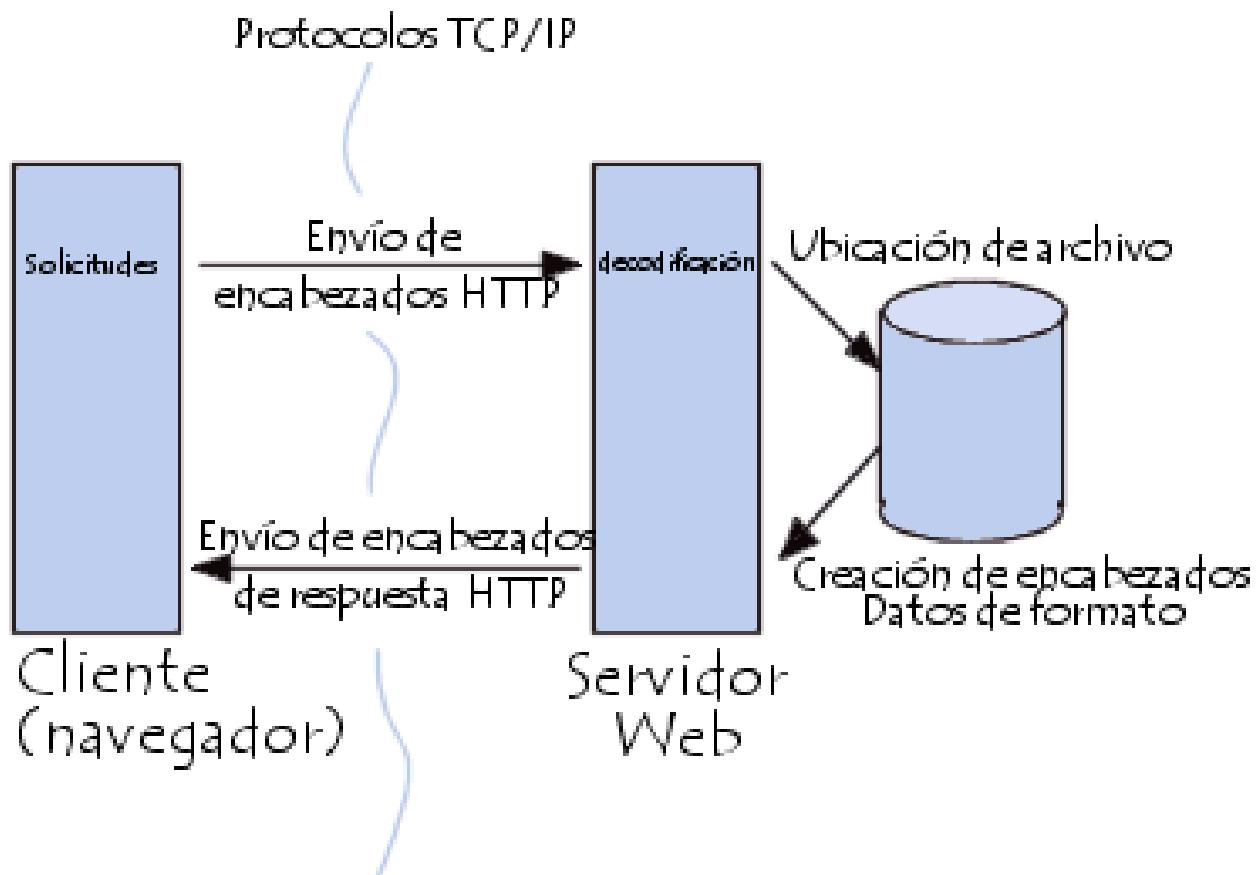
- ✓ Definición completa: *Un servidor web o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando o cediendo una respuesta en cualquier lenguaje o aplicación del lado del cliente.*
- ✓ Los pasos que da el servidor en su ciclo inicial son:
 1. Acepta una conexión TCP de un cliente (un navegador).
 2. Obtiene el nombre del archivo solicitado por el cliente
 3. Obtiene el archivo (del disco)
 4. Devuelve el archivo al cliente
 5. Libera la conexión TCP.

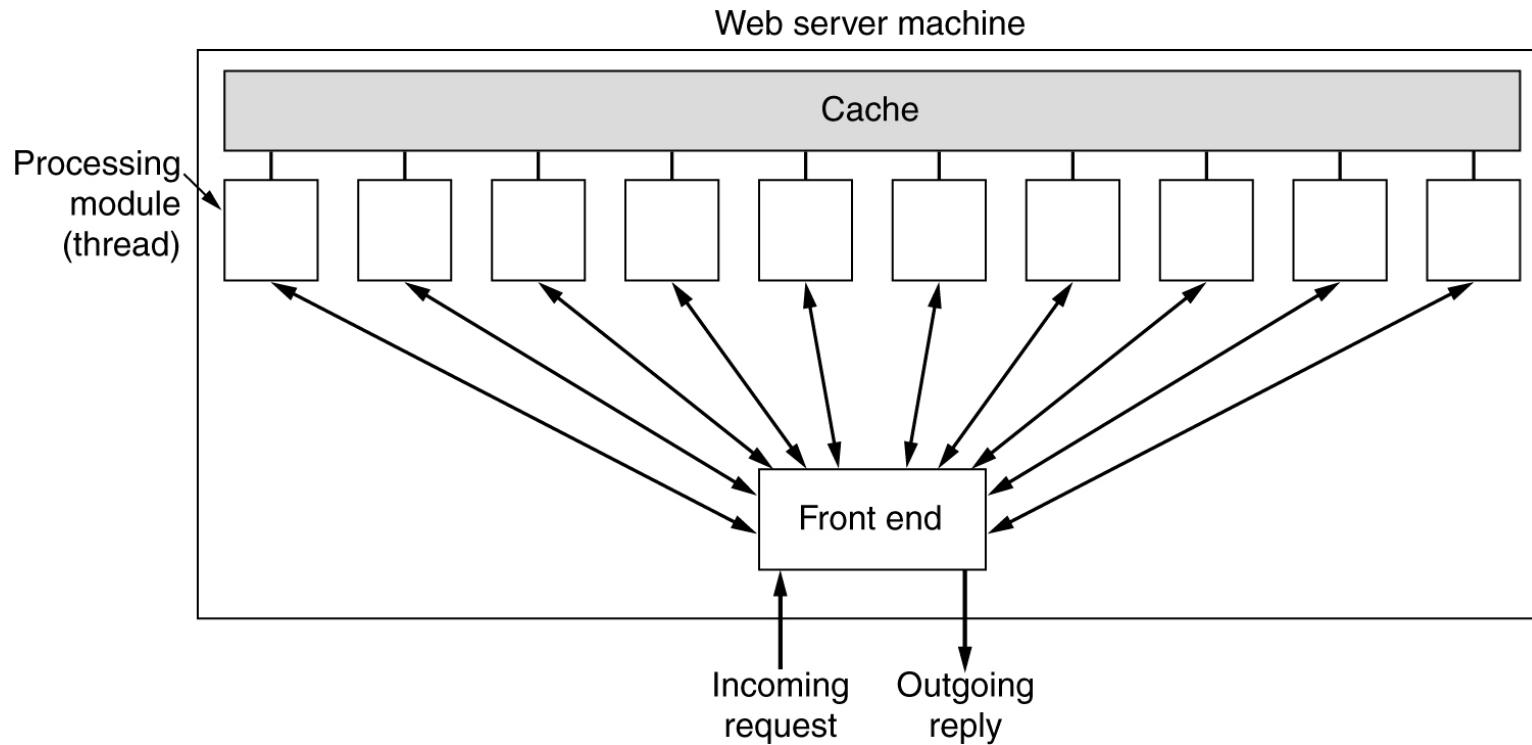


- ✓ Los servidores modernos hacen más que sólo aceptar y devolver nombres de archivos
 - 1. Resuelve el nombre de la página Web solicitado
 - 2. Autentica al cliente
 - 3. Realiza control de acceso en el cliente
 - 4. Realiza control de acceso en la página Web
 - 5. Verifica la caché
 - 6. Obtiene del disco la página solicitada
 - 7. Determina el tipo MIME que se incluirá en la respuesta
 - 8. Se encarga de diversos detalles
 - 9. Devuelve la respuesta al cliente
 - 10. Realiza una entrada en el registro del servidor.

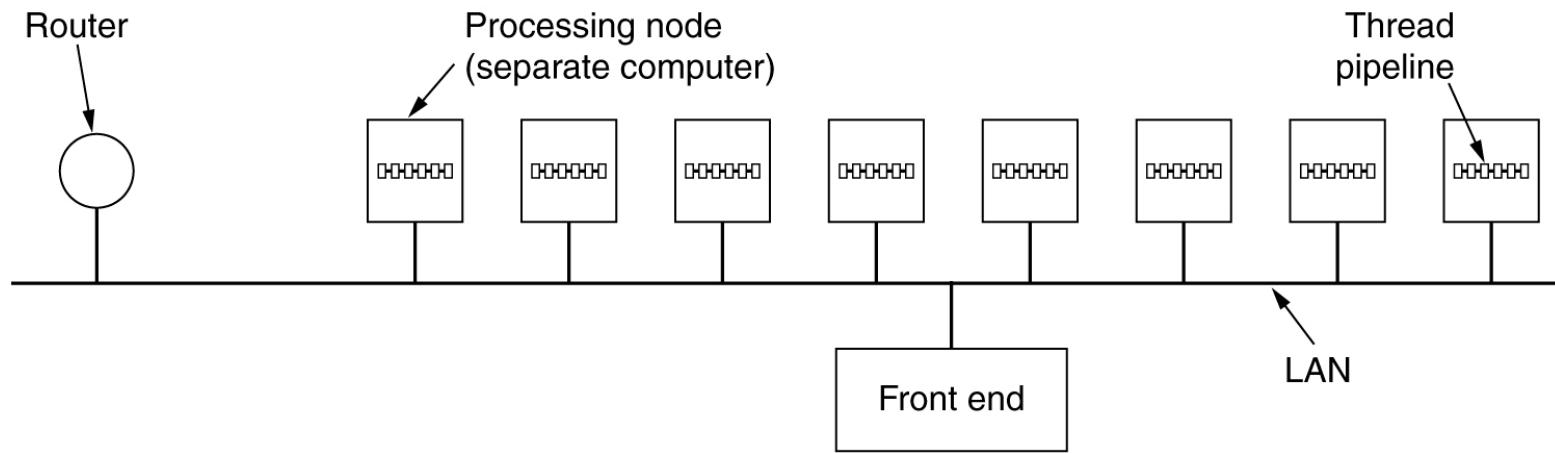
Problemas de velocidad

- ✓ La frustración sobre los problemas de congestión en la infraestructura de Internet y la alta latencia que provoca la lenta navegación, ha llevado a crear un nombre alternativo para la World Wide Web: la World Wide Wait (en castellano, la Gran Espera Mundial). Aumentar la velocidad de Internet es una discusión latente sobre el uso de tecnologías de peering y QoS. Otras soluciones para reducir las esperas de la Web se pueden encontrar en W3C.
- ✓ Las guías estándar para los tiempos de respuesta ideales de las páginas web son (Nielsen 1999, página 42):
 - 0,1 segundos (una décima de segundo). Tiempo de respuesta ideal. El usuario no percibe ninguna interrupción.
 - 1 segundo. Tiempo de respuesta más alto que es aceptable. Los tiempos de descarga superiores a 1 segundo interrumpen la experiencia del usuario.
 - 10 segundos. Tiempo de respuesta inaceptable. La experiencia de usuario es interrumpida y el usuario puede marcharse del sitio web o sistema.
- ✓ Estos tiempos son útiles para planificar la capacidad de los servidores web.





Servidor web multihilo con los módulos de acceso (front end) y procesamiento.



Granja de servidores.



Servidores web:

- Servidor HTTP Apache (libre, servidor más usado del mundo)
- Servidor HTTP Cherokee
- Internet Information Server



Protocolo HTTP

LA NAVEGACIÓN WEB

- FUNDAMENTOS DE REDES 2017/2018. TEMA 2
- ✓ El protocolo de Transferencia de HiperTexto es un sencillo protocolo Cliente/Servidor que articula los intercambios de información entre los clientes web y los servidores http.
 - ✓ Está soportado sobre los servicios de conexión TCP/IP .
 - ✓ Un proceso servidor escucha en un puerto de comunicaciones TCP (80) y espera solicitudes de los clientes Web.
 - ✓ Una vez establecida la conexión se encarga de mantener la comunicación y garantizar un intercambio de datos libres de errores.



- ✓ Se basa en sencillas operaciones de solicitud/respuesta.
- ✓ Un cliente establece una conexión con un servidor y envía un mensaje con los datos de la solicitud.
- ✓ El servidor responde con un mensaje similar que contiene el estado de la operación y su posible resultado.

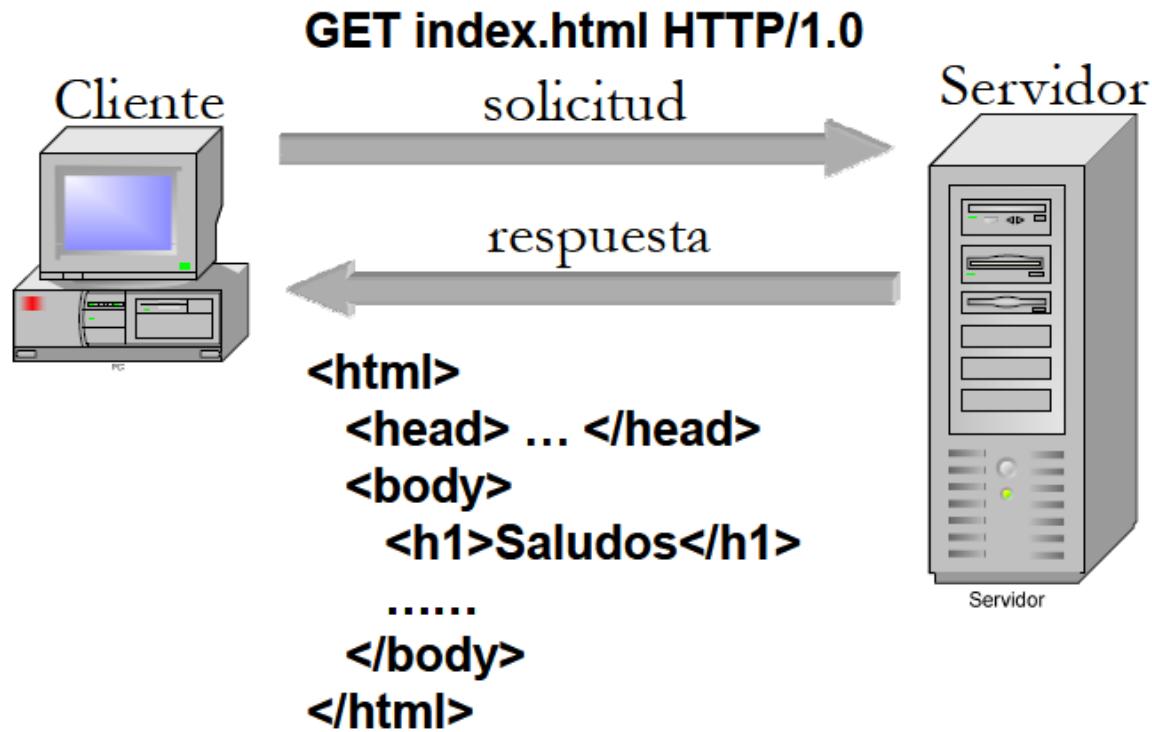
Características del HTTP

- ✓ Protocolo basado en ASCII. De esta forma se puede transmitir cualquier tipo de documento: texto, binario, etc, respetando su formato original.
- ✓ Permite la transferencia de objetos multimedia. El contenido de cada objeto intercambiado está identificado por su clasificación MIME.
- ✓ Existen tres funciones básicas (hay más, pero por lo general no se utilizan) que un cliente puede utilizar para dialogar con el servidor: **GET**, para recoger un objeto, **POST**, para enviar información al servidor y **HEAD**, para solicitar las características de un objeto (por ejemplo, la fecha de modificación de un documento HTML).

LA NAVEGACIÓN WEB

- ✓ Cada operación HTTP implica una conexión con el servidor, que es liberada al término de la misma. Es decir, en una operación se puede recoger un único objeto.
- ✓ No mantiene estado. Cada petición de un cliente a un servidor no es influida por las transacciones anteriores. El servidor trata cada petición como una operación totalmente independiente del resto.
- ✓ Cada objeto al que se aplican los comandos del protocolo está identificado a través de la información de situación del final de la URL.
- ✓ HTTP se diseñó específicamente para el World Wide Web: es un protocolo rápido y sencillo que permite la transferencia de múltiples tipos de información de forma eficiente y rápida. Se puede comparar, por ejemplo, con FTP, que es también un protocolo de transferencia de ficheros, pero tiene un conjunto muy amplio de comandos, y no se integra demasiado bien en las transferencias multimedia.

ETAPAS DE UNA CONEXIÓN HTTP



Etapas de una conexión HTTP:

1. Un usuario accede a una URL, seleccionando un enlace de un documento HTML o introduciéndola directamente en el campo *Location* del cliente Web.
2. El cliente Web decodifica la URL, separando sus diferentes partes. Así identifica el protocolo de acceso, la dirección DNS o IP del servidor, el posible puerto opcional (el valor por defecto es 80) y el objeto requerido del servidor.
3. Se abre una conexión TCP/IP con el servidor, llamando al puerto TCP correspondiente, (80).
4. Se realiza la petición. Para ello, se envía el comando necesario (GET, POST, HEAD,...), la dirección del objeto requerido (el contenido de la URL que sigue a la dirección del servidor), la versión del protocolo HTTP empleada (casi siempre HTTP/1.0) y un conjunto variable de información, que incluye datos sobre las capacidades del navegador y datos opcionales para el servidor.



LA NAVEGACIÓN WEB

5. El servidor devuelve la respuesta al cliente. Consiste en un código de estado y el tipo de dato MIME de la información de retorno, seguido de la propia información.
6. Se cierra la conexión TCP.

NOTA: Este proceso se repite en cada acceso al servidor HTTP.

Estructura de los mensajes HTTP:

- ✓ El diálogo con los servidores HTTP se establece a través de mensajes formados por líneas de texto, cada una de las cuales contiene los diferentes comandos y opciones del protocolo.
- ✓ Sólo existen dos tipos de mensajes, uno para realizar peticiones y otro para devolver la correspondiente respuesta. La estructura general de los dos tipos de mensajes se puede ver en el siguiente esquema:

Mensaje de solicitud		Mensaje de respuesta
Comando HTTP + parámetros		Resultado de la solicitud
Cabeceras del requerimiento (línea en blanco)		Cabeceras de la respuesta (línea en blanco)
Información opcional		Información opcional



LA NAVEGACIÓN WEB

- ✓ La primera línea del mensaje de solicitud contiene el comando que se solicita al servidor HTTP, mientras que en la respuesta contiene el resultado de la operación, un código numérico que permite conocer el éxito o fracaso de la operación.
- ✓ La separación entre cada línea del mensaje se realiza con un par CR-LF (retorno de carro más nueva línea). El final de las cabeceras se indica con una línea en blanco, tras la cual se pueden incluir los datos transportados por el protocolo, por ejemplo, el documento HTML que devuelve un servidor o el contenido de un formulario que envía un cliente .

Formato mensaje de solicitud

línea de petición
(comandos GET,
POST, HEAD)

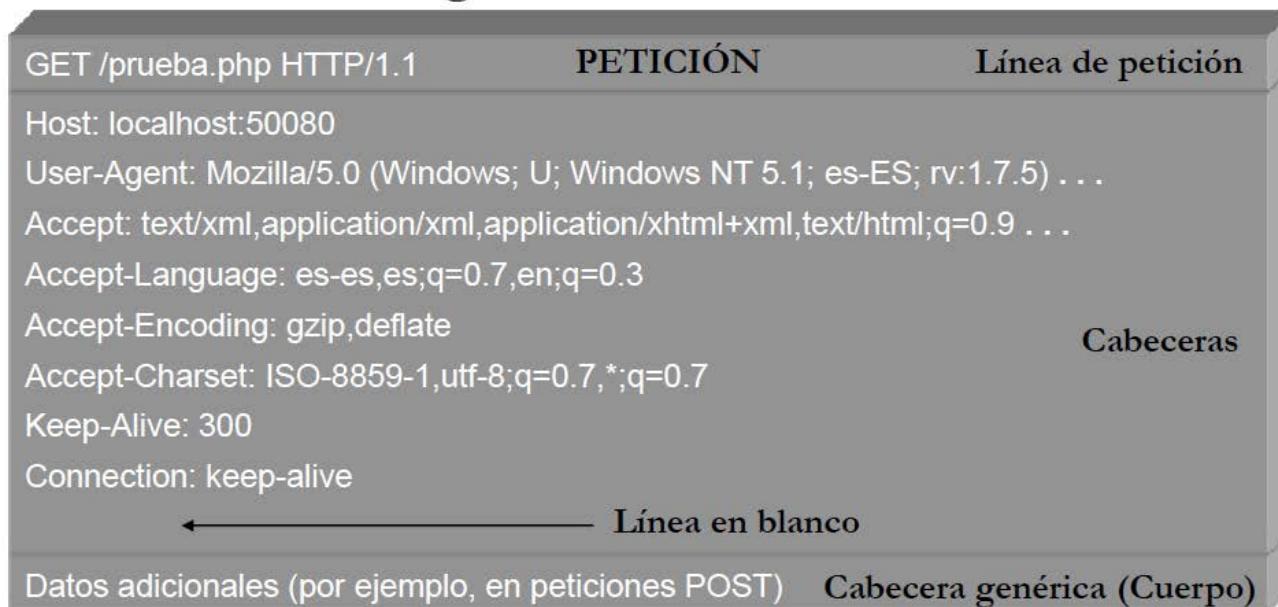
líneas de cabecera

<CR><LF>
indica final
de la cabecera

```
GET /itel/orla2000.html HTTP/1.1
Connection: close
User-agent: Mozilla/4.0
Accept: text/html, image/gif,image/jpeg
Accept-language: sp
```

LA NAVEGACIÓN WEB

- Petición de navegador



Formato mensaje respuesta

línea de estado
(versión, código, cod. verbal)

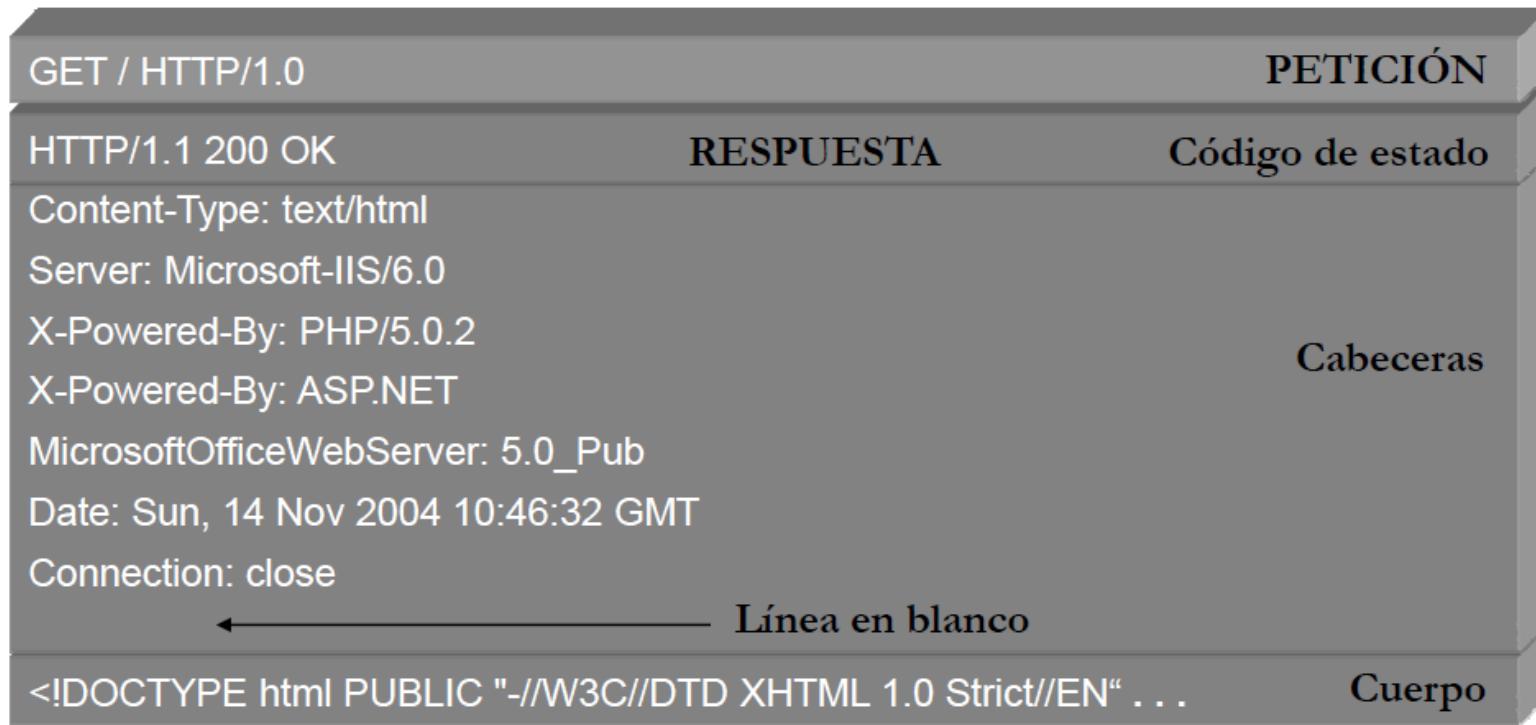
```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 1998 GMT
Content-Length: 6821
Content-Type: text/html
datos datos datos datos ...
```

líneas de cabecera

datos, e.g.,
fichero html solicitado

LA NAVEGACIÓN WEB

- Petición-respuesta





Comandos del protocolo HTTP.

- ✓ Los comandos de HTTP representan las diferentes operaciones que se pueden solicitar a un servidor HTTP. El formato general de un comando es:

Nombre del comando Objeto sobre el que se aplica Versión de HTTP utilizada

- ✓ Cada comando actúa sobre un objeto del servidor, normalmente un fichero o aplicación, que se toma de la URL de activación. La última parte de esta URL, que representa la dirección de un objeto dentro de un servidor HTTP, es el parámetro sobre el que se aplica el comando.



Principales Métodos HTTP.

1- **GET** que se utiliza para recoger cualquier tipo de información del servidor (realizar peticiones).

- Se utiliza siempre que se pulsa sobre un enlace o se teclea WWW directamente una URL.
- Como resultado, el servidor HTTP envía el documento correspondiente a la URL seleccionada, o bien activa un módulo CGI, que generará a su vez la información de retorno.
- En algunos casos se envían datos al servidor en la URL.

LA NAVEGACIÓN WEB

Principales Métodos HTTP.

2 – **HEAD** permite solicitar información sobre un objeto (fichero): tamaño, tipo, fecha de modificación.

- Es utilizado por los gestores de cachés de páginas o los servidores proxy, para conocer cuándo es necesario actualizar la copia que se mantiene de un fichero.

3 – **POST** Sirve para enviar información al servidor (p.e., los datos contenidos en un formulario).

- El servidor pasará esta información a un proceso encargado de su tratamiento

4 – **PUT** permite actualizar información sobre un objeto del servidor.

- Es similar a POST, pero en este caso, la información enviada al servidor debe almacenarse en la URL que acompaña al comando.



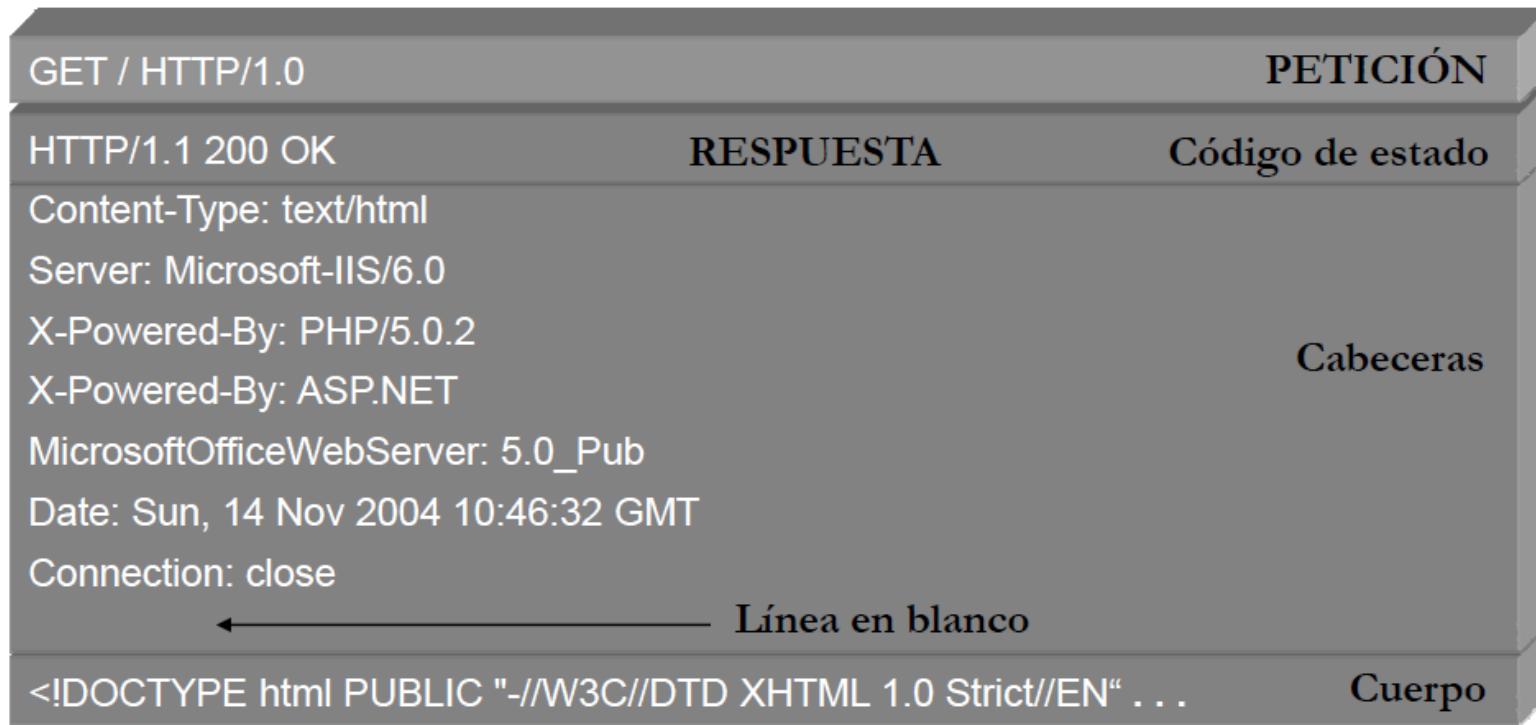
LA NAVEGACIÓN WEB

Principales Métodos HTTP.

- 5– **DELETE** permite eliminar el documento especificado del servidor.
- 6– **TRACE** solicita al servidor que envíe de vuelta en un mensaje de respuesta, en la sección del cuerpo de entidad, toda los datos que reciba del mensaje de solicitud. Se utiliza con fines de comprobación y diagnóstico.
- 7– **OPTIONS** devuelve los métodos HTTP que el servidor soporta para una URL específico. Esto puede ser utilizado para comprobar la funcionalidad de un servidor web mediante petición en lugar de un recurso específico
- 8– **CONNECT** se utiliza para saber si se tiene acceso a un host, no necesariamente la petición llega al servidor, este método se utiliza principalmente para saber si un proxy nos da acceso a un host bajo condiciones especiales, como por ejemplo "corrientes" de datos bidireccionales encriptadas (como lo requiere SSL).

LA NAVEGACIÓN WEB

- Petición-respuesta





Comandos del protocolo HTTP.

- ✓ Los comandos de HTTP representan las diferentes operaciones que se pueden solicitar a un servidor HTTP. El formato general de un comando es:

Nombre del comando Objeto sobre el que se aplica Versión de HTTP utilizada

- ✓ Cada comando actúa sobre un objeto del servidor, normalmente un fichero o aplicación, que se toma de la URL de activación. La última parte de esta URL, que representa la dirección de un objeto dentro de un servidor HTTP, es el parámetro sobre el que se aplica el comando.



Principales Métodos HTTP.

1- **GET** que se utiliza para recoger cualquier tipo de información del servidor (realizar peticiones).

- Se utiliza siempre que se pulsa sobre un enlace o se teclea WWW directamente una URL.
- Como resultado, el servidor HTTP envía el documento correspondiente a la URL seleccionada, o bien activa un módulo CGI, que generará a su vez la información de retorno.
- En algunos casos se envían datos al servidor en la URL.



LA NAVEGACIÓN WEB

Principales Métodos HTTP.

2 – **HEAD** permite solicitar información sobre un objeto (fichero): tamaño, tipo, fecha de modificación.

- Es utilizado por los gestores de cachés de páginas o los servidores proxy, para conocer cuándo es necesario actualizar la copia que se mantiene de un fichero.

3 – **POST** Sirve para enviar información al servidor (p.e., los datos contenidos en un formulario).

- El servidor pasará esta información a un proceso encargado de su tratamiento

4 – **PUT** permite actualizar información sobre un objeto del servidor.

- Es similar a POST, pero en este caso, la información enviada al servidor debe almacenarse en la URL que acompaña al comando.

LA NAVEGACIÓN WEB

Principales Métodos HTTP.

- 5– **DELETE** permite eliminar el documento especificado del servidor.
- 6– **TRACE** solicita al servidor que envíe de vuelta en un mensaje de respuesta, en la sección del cuerpo de entidad, toda los datos que reciba del mensaje de solicitud. Se utiliza con fines de comprobación y diagnóstico.
- 7– **OPTIONS** devuelve los métodos HTTP que el servidor soporta para una URL específico. Esto puede ser utilizado para comprobar la funcionalidad de un servidor web mediante petición en lugar de un recurso específico
- 8– **CONNECT** se utiliza para saber si se tiene acceso a un host, no necesariamente la petición llega al servidor, este método se utiliza principalmente para saber si un proxy nos da acceso a un host bajo condiciones especiales, como por ejemplo "corrientes" de datos bidireccionales encriptadas (como lo requiere SSL).



Códigos de estado del servidor HTTP.

- **1xx:** mensajes informativos.
- **2xx:** mensajes asociados con operaciones realizadas correctamente.
- **3xx:** mensajes de redirección, que informan de operaciones complementarias que se deben realizar para finalizar la operación.
- **4xx:** errores del cliente; el requerimiento contiene algún error, o no puede ser realizado.
- **5xx:** errores del servidor, que no se puede llevar a cabo una solicitud.



Las cabeceras.

- ✓ Son un conjunto de variables que se incluyen en los mensajes HTTP, para modificar su comportamiento o incluir información de interés. En función de su nombre, pueden aparecer en los requerimientos de un cliente, en las respuestas del servidor o en ambos tipos de mensajes. El formato general de una cabecera es:

Nombre de la variable: Cadena ASCII con su valor

- ✓ Los nombres de variables se pueden escribir con cualquier combinación de mayúsculas y minúsculas. Además, se debe incluir un espacio en blanco entre el signo : y su valor. En caso de que el valor de una variable ocupe varias líneas, éstas deberán comenzar, al menos, con un espacio en blanco o un tabulador.



- ✓ **Cabeceras comunes para peticiones y respuestas**
 - **Content-Type:** descripción MIME de la información contenida en este mensaje.
 - **Content-Length:** longitud en bytes de los datos enviados, expresado en base decimal.
 - **Content-Encoding:** formato de codificación de los datos enviados en este mensaje. Sirve, por ejemplo, para enviar datos comprimidos o encriptados.
 - **Date:** fecha local de la operación. Las fechas deben incluir la zona horaria en que reside el sistema que genera la operación. Por ejemplo: Sunday, 12-Dec-96 12:21:22 GMT+01. No existe un formato único en las fechas.

LA NAVEGACIÓN WEB

✓ **Cabeceras sólo para peticiones del cliente**

- **Accept:** campo opcional que contiene una lista de tipos MIME aceptados por el cliente.
- **Authorization:** clave de acceso que envía un cliente para acceder a un recurso de uso protegido o limitado. La información incluye el formato de autorización empleado, seguido de la clave de acceso propiamente dicha. La explicación se incluye más adelante.
- **From:** campo opcional que contiene la dirección de correo electrónico del usuario del cliente Web que realiza el acceso.



- **If-Modified-Since:** permite realizar operaciones GET condicionales, en función de si la fecha de modificación del objeto requerido es anterior o posterior a la fecha proporcionada. Puede ser utilizada por los sistemas de almacenamiento temporal de páginas. Es equivalente a realizar un HEAD seguido de un GET normal.
- **Referer:** contiene la URL del documento desde donde se ha activado este enlace. De esta forma, un servidor puede informar al creador de ese documento de cambios o actualizaciones en los enlaces que contiene. No todos los clientes lo envían.
- **User-agent:** cadena que identifica el tipo y versión del cliente que realiza la petición. Por ejemplo, los *browsers* de Netscape envían cadenas del tipo User-Agent: Mozilla/3.0 (WinNT; I)

✓ Cabeceras sólo para respuestas del servidor HTTP

- **Allow:** informa de los comandos HTTP opcionales que se pueden aplicar sobre el objeto al que se refiere esta respuesta. Por ejemplo, Allow: GET, POST.
- **Expires:** fecha de expiración del objeto enviado. Los sistemas de cache deben descartar las posibles copias del objeto pasada esta fecha. Por ejemplo, Expires: Thu, 12 Jan 97 00:00:00 GMT+1. No todos los sistemas lo envían.
- **Last-modified:** fecha local de modificación del objeto devuelto. Se puede corresponder con la fecha de modificación de un fichero en disco, o, para información generada dinámicamente desde una base de datos, con la fecha de modificación del registro de datos correspondiente.



- ✓ HTTP ha pasado por múltiples versiones del protocolo, muchas de las cuales son compatibles con las anteriores.
- ✓ El RFC 2145 describe el uso de los números de versión de HTTP. El cliente le dice al servidor al principio de la petición la versión que usa, y el servidor usa la misma o una anterior en su respuesta.
- ✓ 0.9 => Obsoleta. Soporta sólo un comando, GET, y además no especifica el número de versión HTTP. No soporta cabeceras. Como esta versión no soporta POST, el cliente no puede enviarle mucha información al servidor.
- ✓ HTTP/1.0 (mayo de 1996) => Esta es la primera revisión del protocolo que especifica su versión en las comunicaciones, y todavía se usa ampliamente, sobre todo en servidores proxy. RFC 1945.



- ✓ HTTP/1.1 (junio de 1999) => Versión actual; las conexiones persistentes están activadas por defecto y funcionan bien con los proxies. También permite al cliente enviar múltiples peticiones a la vez por la misma conexión (pipelining) lo que hace posible eliminar el tiempo de Round-Trip delay por cada petición. RFC 2616
- ✓ HTTP/1.2 => Los primeros borradores de 1995 están en documento PEP (Protocolo de Extensión de Protocolo), un mecanismo de extensión para HTTP. Los hizo el World Wide Web Consortium y se envió al Internet Engineering Task Force. El PEP inicialmente estaba destinado a convertirse en un rango distintivo de HTTP/1.2. En borradores posteriores, sin embargo, se eliminó la referencia a HTTP/1.2. El RFC 2774, HTTP Extension Framework, incluye en gran medida a PEP. Se publicó en febrero de 2000.



Arquitectura HTTP

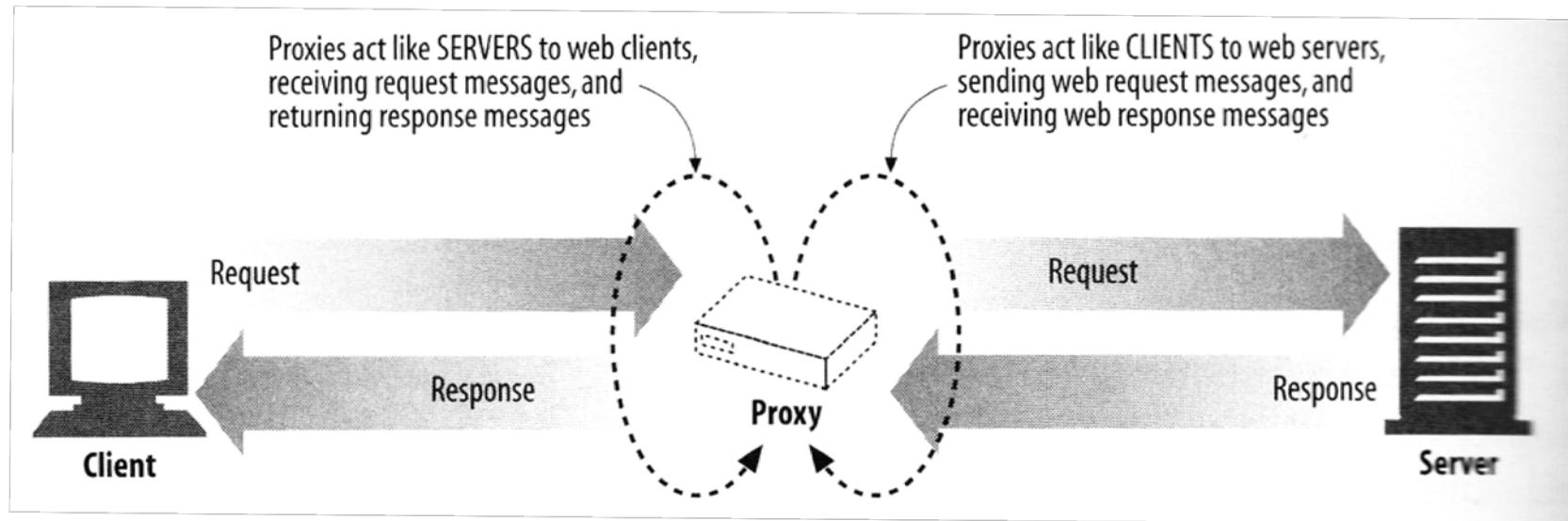


LA NAVEGACIÓN WEB

- ✓ Los requerimientos de los sistemas y aplicaciones web actuales hacen que no baste con una arquitectura basada únicamente en un cliente y un servidor que intercambian información.
- ✓ El gran crecimiento de las aplicaciones web hace necesario completar el sistema básico con nuevos elementos como:
 - Proxies,
 - Caches
 - Gateways

✓ Servidor Proxy

Servidor intermedio entre el cliente y el servidor que hace el papel de cliente y de servidor

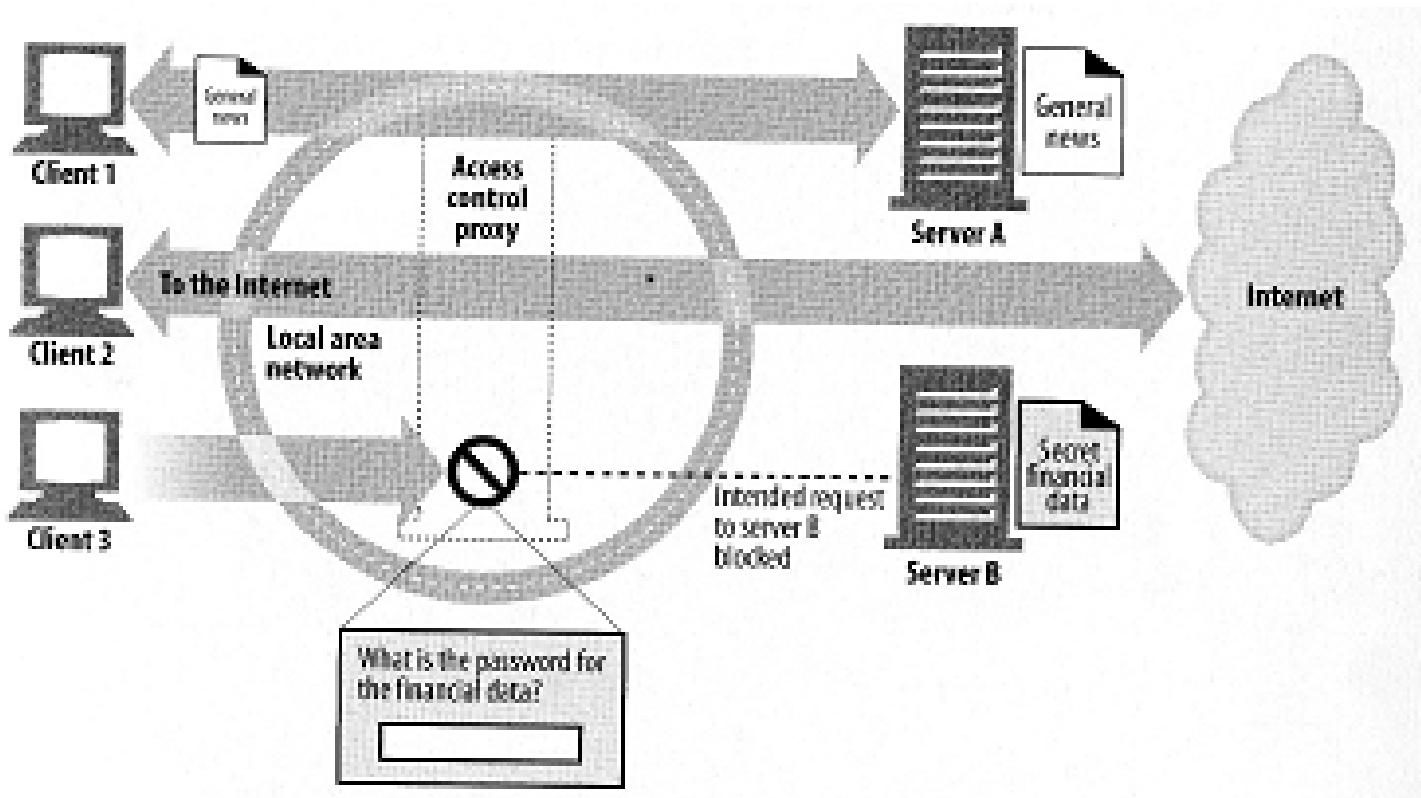


✓ Ventajas del Servidor Proxy

- **Control.** Sólo el intermediario hace el trabajo real, por tanto se pueden limitar y restringir los derechos de los usuarios, y dar permisos sólo al proxy.
- **Ahorro.** Por tanto, sólo uno de los usuarios (el proxy) ha de estar equipado para hacer el trabajo real.
- **Velocidad.** Si varios clientes van a pedir el mismo recurso, el proxy puede hacer caché: guardar la respuesta de una petición para darla directamente cuando otro usuario la pida. Así no tiene que volver a contactar con el destino, y acaba más rápido.
- **Filtrado.** El proxy puede negarse a responder algunas peticiones si detecta que están prohibidas.

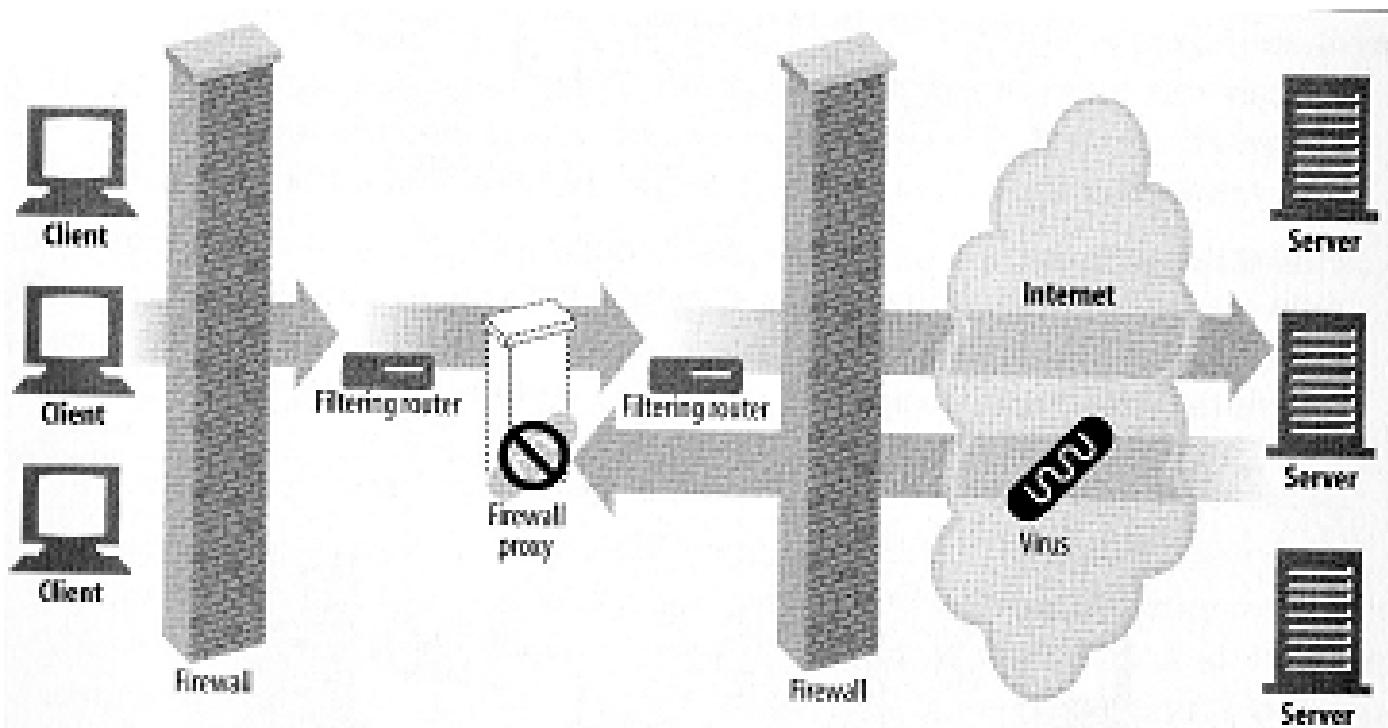
LA NAVEGACIÓN WEB

Ej.1 Acceso centralizado a documentos



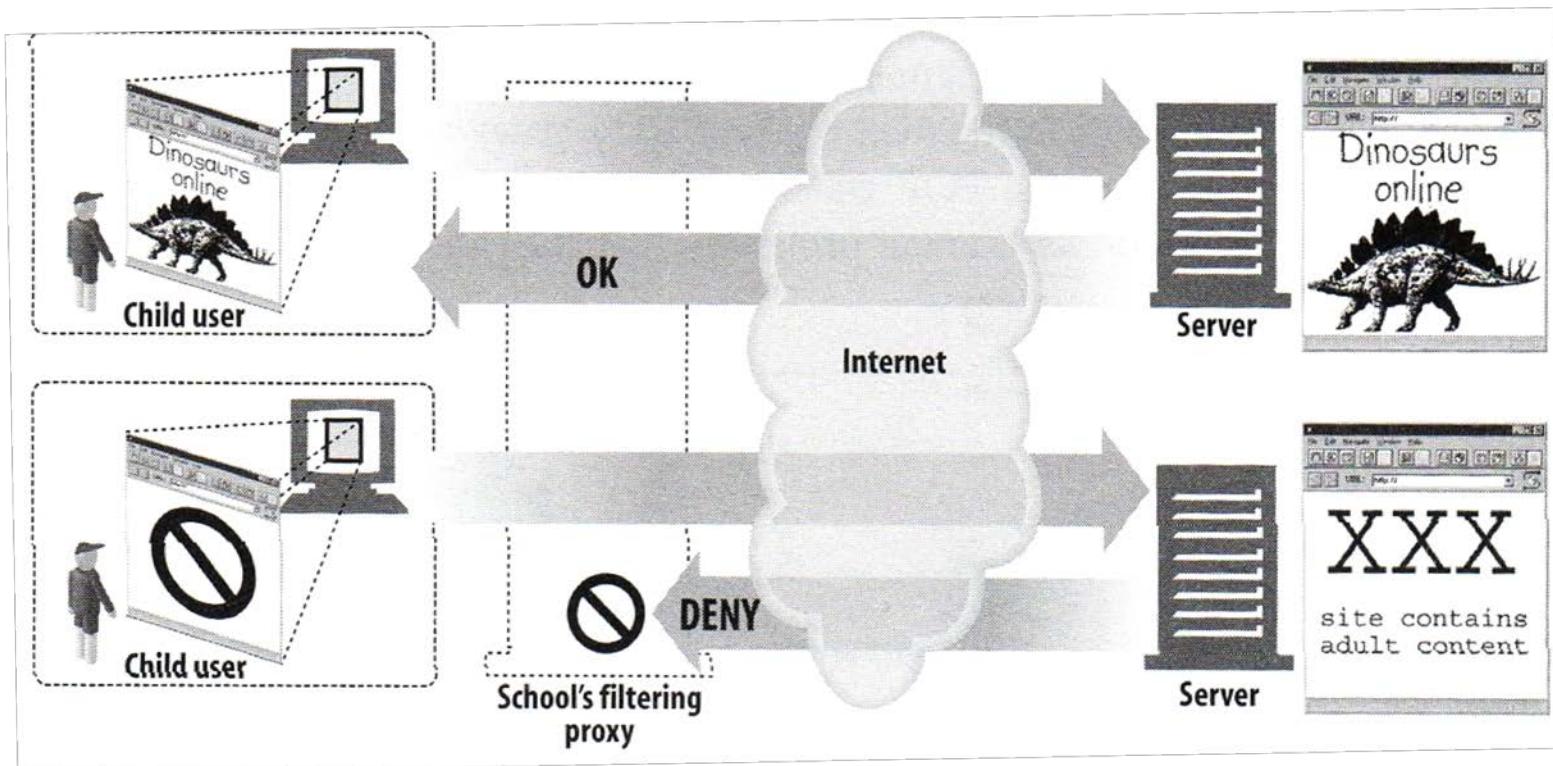
LA NAVEGACIÓN WEB

Ej.2 Seguridad/Firewall



LA NAVEGACIÓN WEB

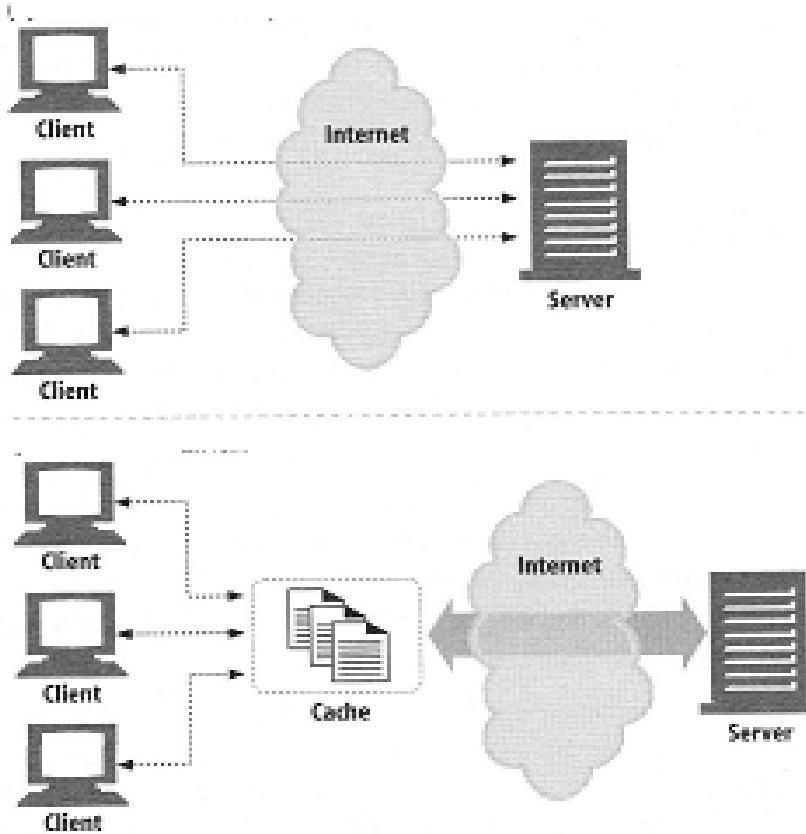
Ej.3 Filtros de protección a menores



LA NAVEGACIÓN WEB

✓ Servidor Caché.

Servidor que almacena en memoria las páginas que se han consultado más recientemente





✓ Ventajas del Servidor Caché.

- Reduce las transferencias de datos redundantes.
- Reduce los “cuellos de botella” en los servidores web.
- Optimiza el rendimiento de los servidores web.
- Reduce los retardos en los tiempos de respuesta.

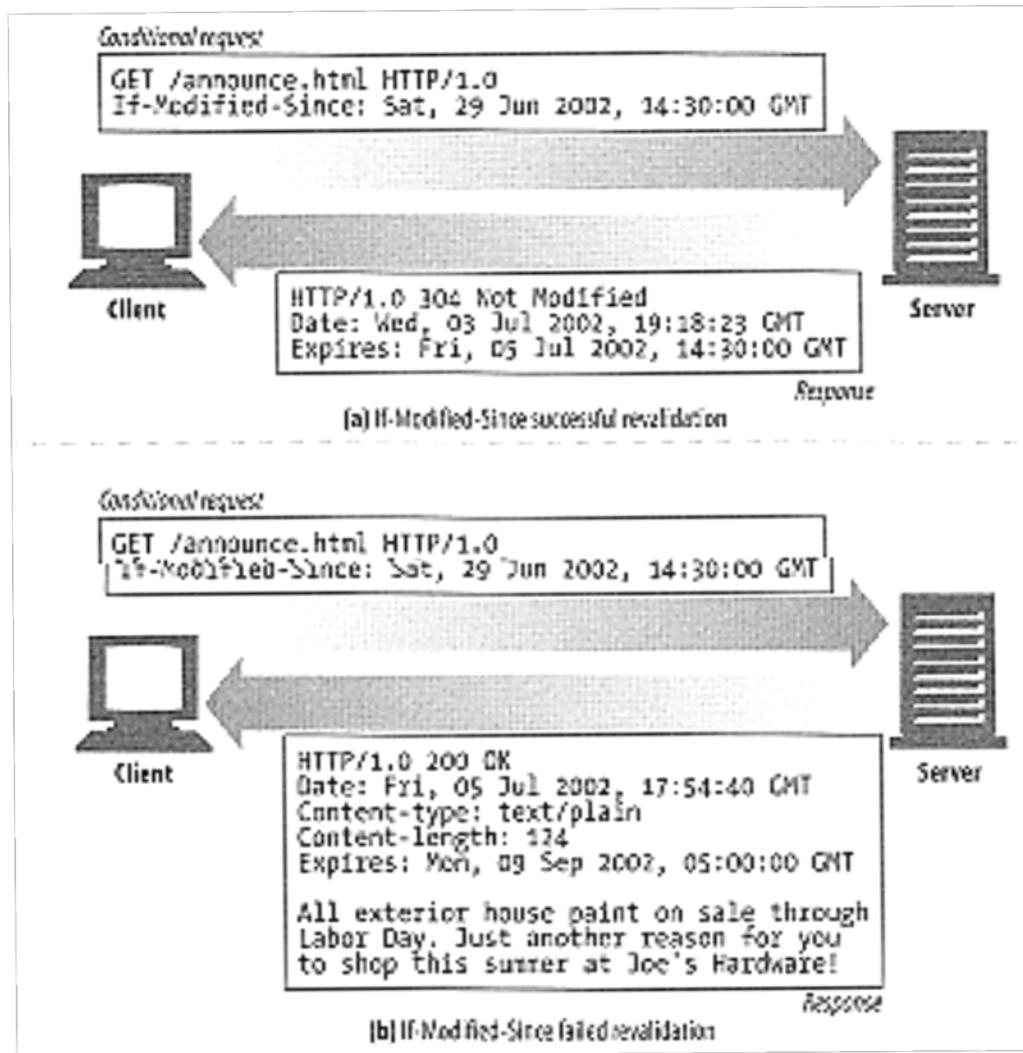


✓ Etapas del proceso de caché.

(ejemplo válido para una solicitud con comando GET)

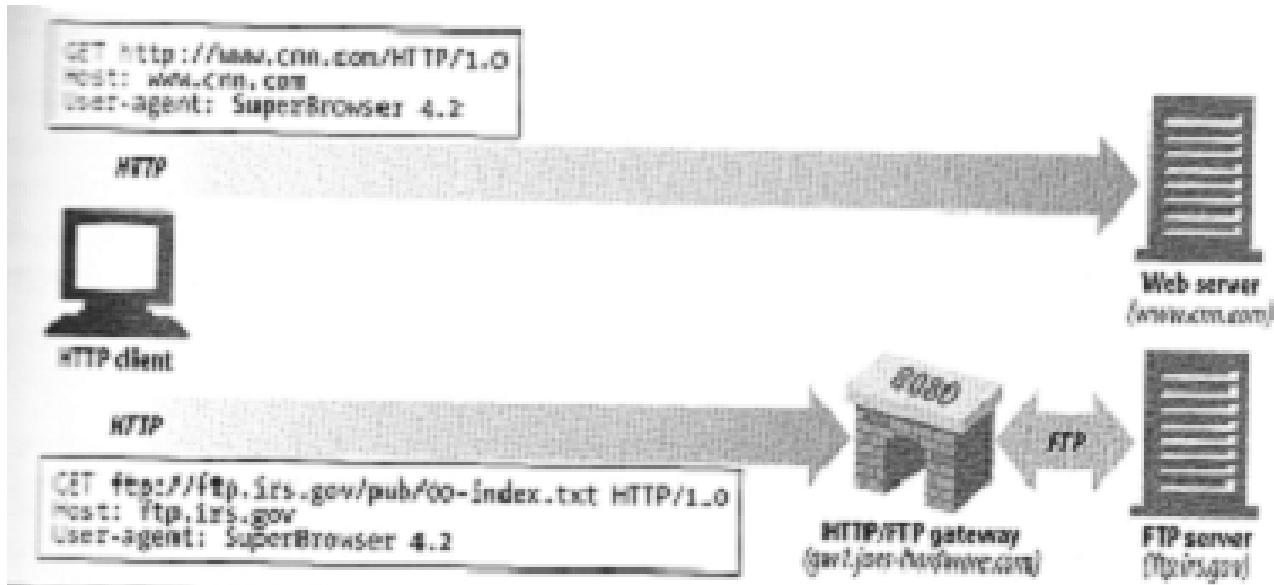
- 1. Recepción :** el servidor Caché lee el mensaje que le llega.
- 2. Análisis:** el servidor Caché analiza el mensaje, extrae la URL y las cabeceras.
- 3. Búsqueda:** El servidor caché verifica si tiene una copia disponible en local del documento solicitado.
- 4. Validación:** El servidor verifica si la copia es lo suficientemente actual, de no ser así solicita al servidor una nueva versión.
- 5. Respuesta:** El servidor caché genera la respuesta y la envía al cliente web.

LA NAVEGACIÓN WEB

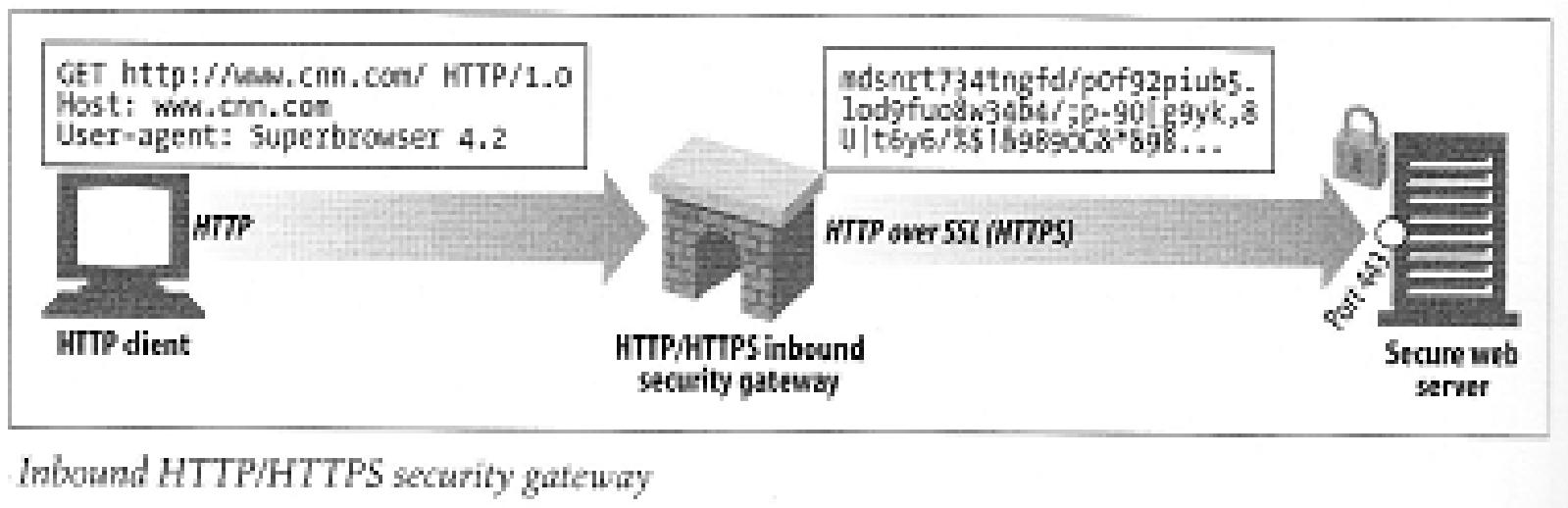


✓ Gateways.

Servidor intermedio que actúa como interfaz entre el protocolo HTTP y otros protocolos o aplicaciones



LA NAVEGACIÓN WEB





Identificación, Autorización y Seguridad



✓ Identificación y Cookies

HTTP es un protocolo que **no mantiene estado**, es decir que trata todas las conexiones de forma anónima e independiente.

Con los actuales sistemas y aplicaciones web se hace necesario subsanar esta carencia del HTTP. Existen unas cuantas técnicas para identificar a los clientes en HTTP.

- Algunas cabeceras contiene información sobre el cliente
- Usando usuario y Password podremos también identificar a los clientes.
- Cookies.

LA NAVEGACIÓN WEB

- ✓ Las ***cookies*** son pequeños ficheros de texto que se intercambian los clientes y servidores HTTP, para solucionar una de las principales deficiencias del protocolo: la falta de información de estado entre dos transacciones. Fueron introducidas por Netscape, y ahora han sido estandarizadas en el RFC 2109.
- La primera vez que un usuario accede a un determinado documento de un servidor, éste proporciona una *cookie* que contiene datos que relacionarán posteriores operaciones.
 - El cliente almacena la *cookie* en su sistema para usarla después. En los futuros accesos a este servidor, el *navegador* podrá proporcionar la *cookie* original, que servirá de nexo entre este acceso y los anteriores.
 - Todo este proceso se realiza automáticamente, sin intervención del usuario.



- ✓ Su aplicación más inmediata son los sistemas de **compra electrónica**. Estos supermercados virtuales necesitan relacionar el contenido de un pedido con el cliente que lo ha solicitado.
- ✓ Otro uso muy interesante son los **sistemas personalizados de recepción de información**, en los que es posible construir una página a medida, con información procedente de fuentes muy diversas. En accesos sucesivos, el cliente enviará la *cookie*, y el servidor podrá generar una página personalizada con las preferencias del usuario.
- ✓ Por último, algunas compañías emplean las *cookies* para realizar un seguimiento de los accesos a sus servidores WWW, identificando las páginas más visitadas, la manera en que se pasa de una a otra sección, etc.



LA NAVEGACIÓN WEB

Uso de las *cookies*

- ✓ Una *cookie* es simplemente una serie de líneas de texto, con pares variable/valor. Existe un conjunto predefinido de nombres de variable, necesarias para el correcto funcionamiento de las *cookies*.
 - **Domain**= conjunto de direcciones Internet para el que es válida la *cookie*. Se puede dar una dirección única (www.mitienda.es) o un rango (.netscape.com).
 - **Path**= fija el subconjunto de URLs para las que sirve esta *cookie*.
 - **Version**= Permite seleccionar entre diferentes versiones del modelo de *cookies*.
 - **Expires**= Fecha de expiración de la información. Si no se incluye, los datos son descartados al finalizar la sesión con el cliente Web.



LA NAVEGACIÓN WEB

- ✓ Un servidor HTTP envía los diferentes campos de una *cookie* con la nueva cabecera HTTP Set-Cookie:

Set-Cookie: Domain=www.unican.es; Path=/; Nombre=Luis; Expires Fri, 15-Jul-97 12:00:00 GMT

- Cuando se accede a una URL que verifica el par dominio/path registrado, el cliente enviará automáticamente la información de los diferentes campos de la cookie con la nueva cabecera HTTP Cookie:

Cookie: Domain=www.unican.es; Path=/; Nombre=Luis



LA NAVEGACIÓN WEB

- ✓ Ejemplo de una cookie

EGSOFT_ID
191.46.211.13-655193640.29148285
www.rollingstone.com/
0
2867435528
30124157
4206779936
291478284

- ✓ La información dentro de una *cookie* se organiza a partir de líneas de texto.
- ✓ El campo más interesante es la tercera línea.
- ✓ Las *cookies* pueden tener asociada una fecha de expiración, a partir de la cual el cliente Web tratará de obtener una nueva versión.



FUNDAMENTOS DE REDES 2017/2018. TEMA 2

Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red
2. Servicio de Nombres de Dominio (DNS)
3. La navegación Web
- 4. El Correo electrónico**
5. Protocolos seguros
6. Aplicaciones multimedia
7. Aplicaciones para interconectividad de redes locales
8. Cuestiones y ejercicios



Introducción



Introducción:

- El correo electrónico es un servicio de red que permite a los usuarios enviar y recibir mensajes y archivos rápidamente mediante sistemas de comunicación electrónicos.
- El nacimiento del correo electrónico (*email*) ocurrió a principios de los años 60. En este sistema un usuario sólo era capaz de enviar mensajes a usuarios del mismo sistema.
- Una dirección de correo electrónico es un conjunto de palabras que identifican a una persona que puede enviar y recibir correo.
- A la dirección se puede acceder mediante un nombre de usuario y una Contraseña.



Introducción:

- En 1971, Ray Tomlinson incorporó el uso de la arroba (@). Eligió la arroba como divisor entre el usuario y la computadora en la que se aloja el correo (dominio del *proveedor* que da el correo) porque no existía la arroba en ningún nombre ni apellido.
- En inglés la arroba se lee «at» (en). Así, destinatario@destino.es se lee destinatario en destino punto es.
- El destinatario puede tener letras, números, y algunos signos.
- La dirección la tiene que dar un proveedor de correo, que son quienes ofrecen el servicio de envío y recepción.
- Es indiferente que las letras que integran la dirección estén escritas en mayúscula o minúscula

Introducción:

- La primera transferencia verdadera de correo electrónico en la red se llevó a cabo en 1971. Se envió un mensaje de prueba entre dos máquinas a través de ARPANET.
- Los primeros sistemas de correo electrónico simplemente consistían en protocolos de transferencia de archivos, con la convención de que la primera línea de cada mensaje (es decir del archivo) contenía la dirección del destinatario.
- Limitaciones de este sistema: envío a grupos, sin notificación.
- En 1982 se publicaron las propuestas de correo electrónico del ARPANET como RFC 821 (protocolo de transmisión SMTP) y RFC 822 (formato de mensaje).
- Dos años después, el CCITT (*Comité Consultivo Internacional Telegráfico y Telefónico*) elaboró su recomendación X.400, pero su excesiva complejidad, hace que no se utilice, como la mayoría de aplicaciones OSI.



Arquitectura del sistema de correo

Arquitectura del sistema de correo:

- Generalmente, el cliente de correo electrónico permite configurar diferentes tipos de conexión.
- El correo electrónico se entrega usando una arquitectura cliente/servidor:
 1. Un mensaje de correo electrónico se crea usando un programa de correo cliente.
 2. Este programa envía el mensaje a un servidor.
 3. El servidor lo redirige al servidor de correo del destinatario y allí se le suministra al cliente de correo del destinatario.



Arquitectura del sistema de correo:

- Para permitir todo este proceso, existe una variedad de **protocolos de red estándar** que permiten que diferentes máquinas, a menudo ejecutando sistemas operativos diferentes y usando diferentes programas de correo, envíen y reciban correo electrónico o email.

- Los sistemas de correo actuales se basan en la distinción entre la envoltura (RFC 821) y su contenido (RFC 822).

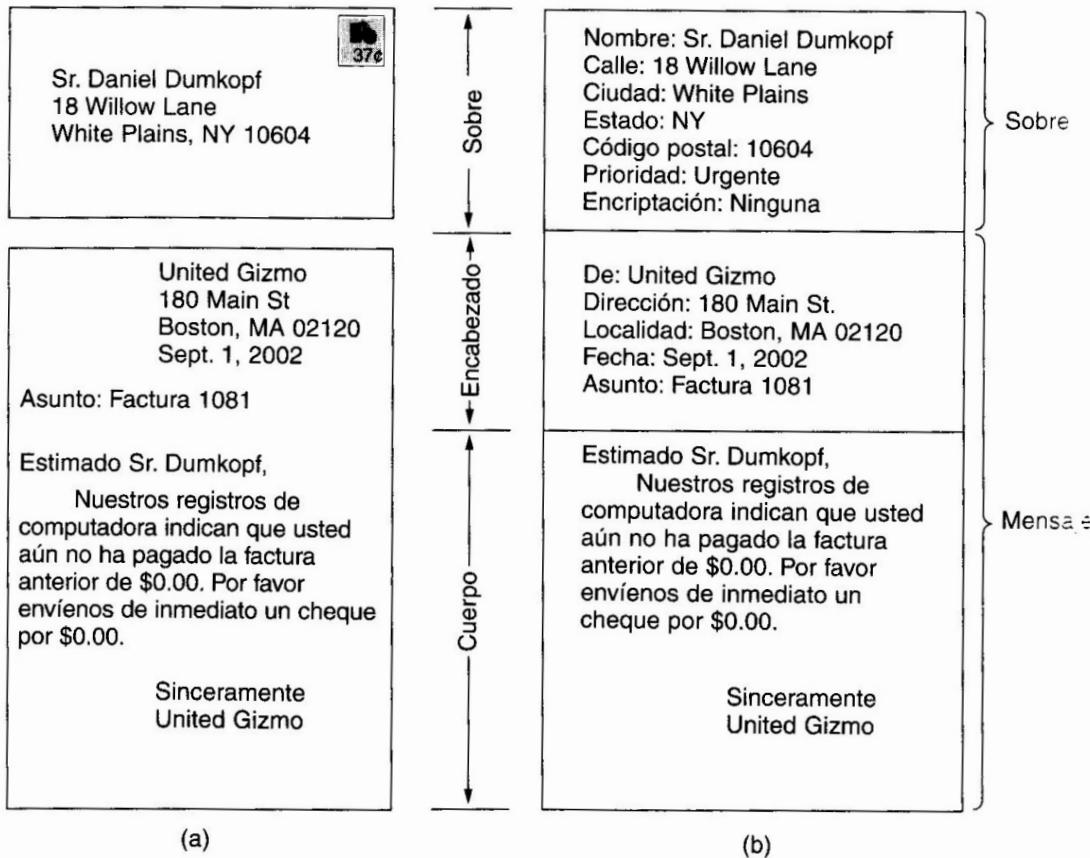
Arquitectura del sistema de correo:

- Para poder escribir un correo electrónico, el programa pide como mínimo tres cosas:
 - **Destinatario:** una o varias direcciones de correo a las que ha de llegar el mensaje. Si el destino son varias personas, normalmente se puede usar una lista con todas las direcciones, separadas por comas o punto y coma.
 - **Asunto:** una descripción corta que verá la persona que lo reciba antes de abrir el correo
 - **El propio mensaje:** Puede ser sólo texto, o incluir formato, y no hay límite de tamaño
- Además, se puede incluir archivos adjuntos al mensaje.

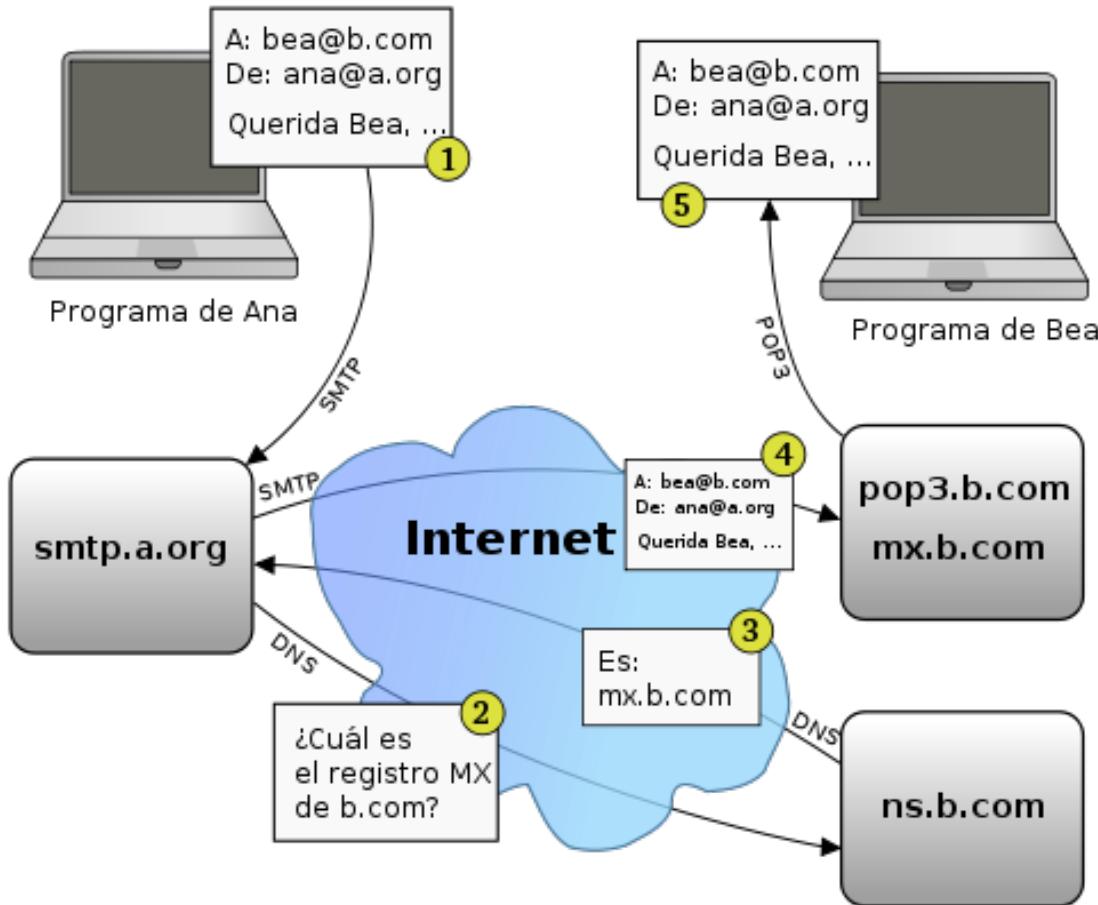
Arquitectura del sistema de correo:

- También existen los campos CC y CCO, que son opcionales y sirven para hacer llegar copias del mensaje a otras personas:
- **Campo CC (Copia de Carbón):** quienes estén en esta lista recibirán también el mensaje, pero verán que no va dirigido a ellos. Tanto el destinatario principal como los del campo CC pueden ver la lista completa.
- **Campo CCO (Copia de Carbón Oculta):** una variante del CC, que hace que los destinatarios reciban el mensaje sin aparecer en ninguna lista. Por tanto, el campo CCO nunca lo ve ningún destinatario

Arquitectura del sistema de correo:



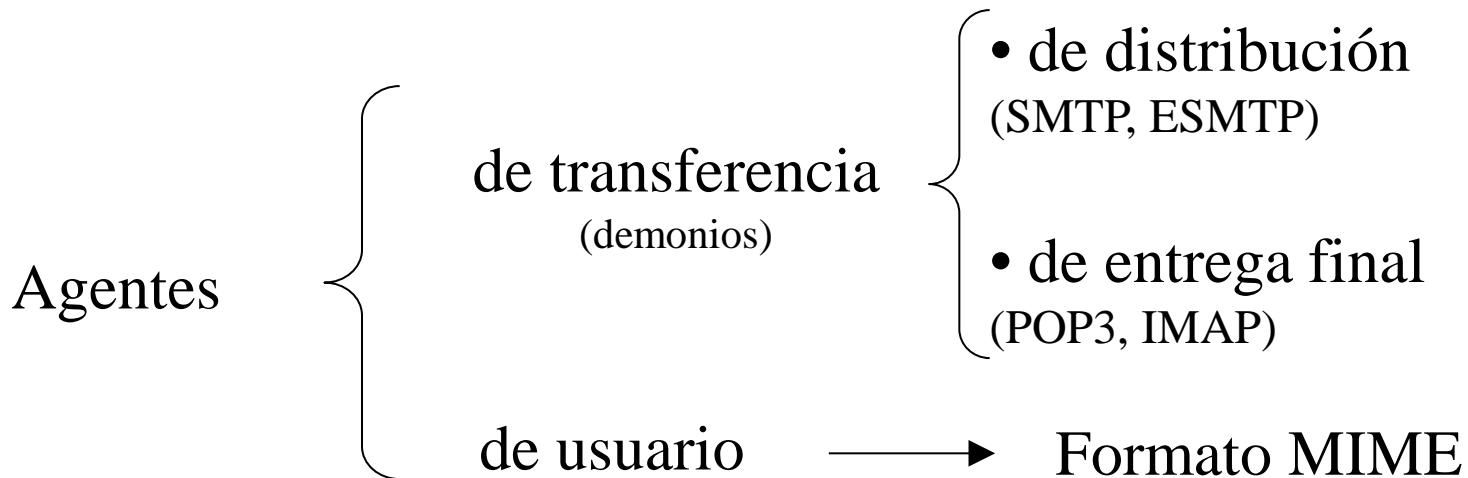
Arquitectura del sistema de correo:



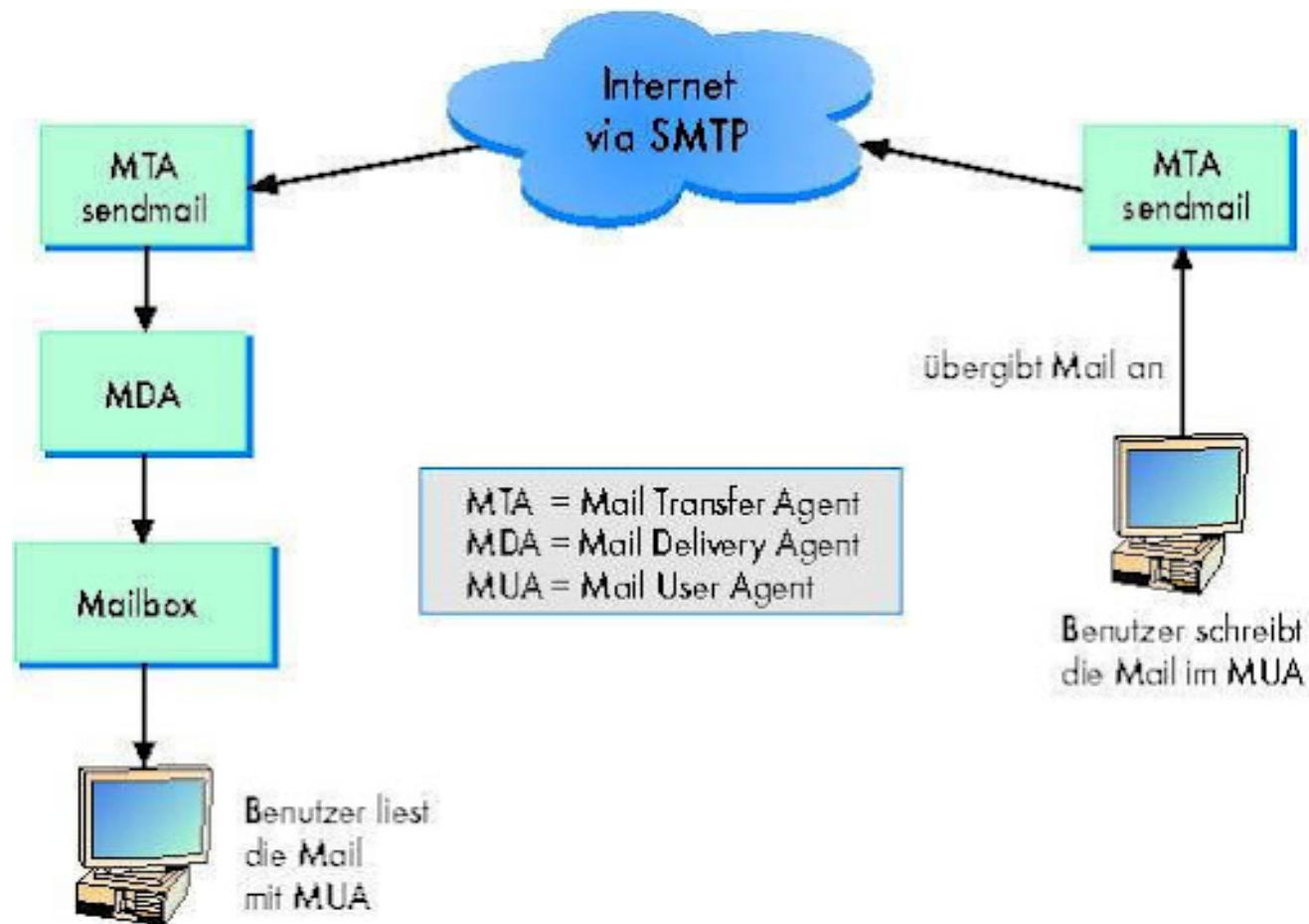
Arquitectura del sistema de correo:

➤ Subsistemas que lo forman:

- **Agentes de transferencia (MTA):** mueven los mensajes del origen al destino, son por lo general *demonios (daemons)*
- **Agentes de usuario (MDA):** Permiten leer y enviar correo. Son programas locales que proporcionan un método basado en comandos, menús o interfaces gráficas.



Arquitectura del sistema de correo:





Agentes de transferencia



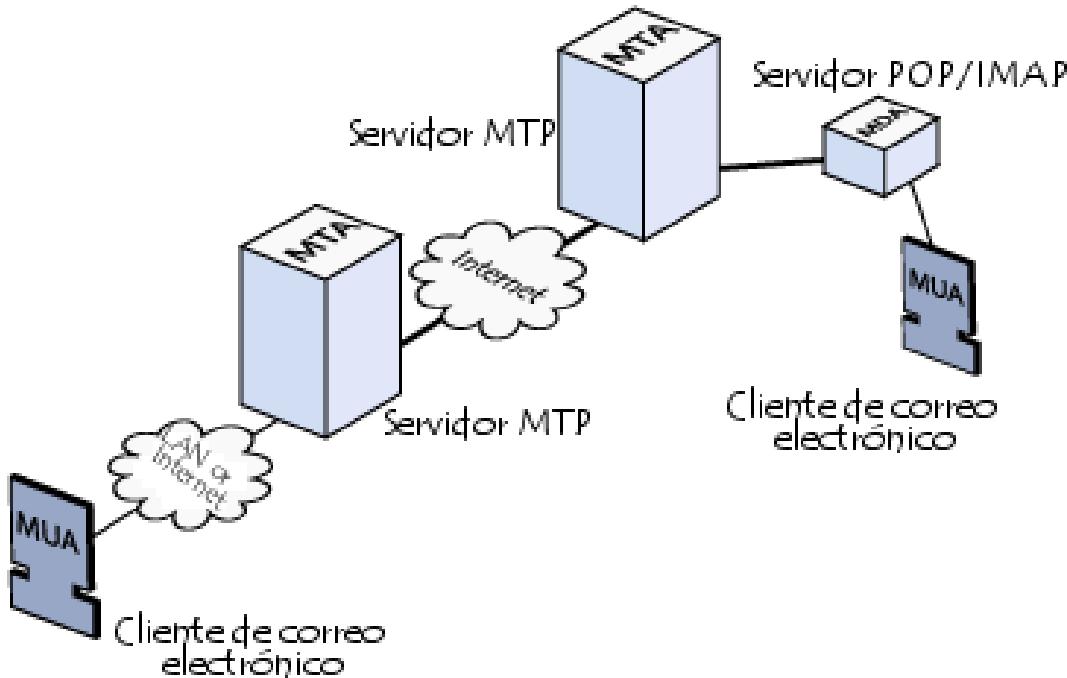
Agentes de transferencia:

Se clasifican en:

- Agentes de **distribución**: utilizando para ello el protocolo SMTP (Simple Mail Transfer Protocol) RFC 821.
- Agentes de **entrega final**: que permita al usuario gestionar su correo a través de una máquina remota, utilizando los protocolos POP3 (Post Office Protocol) RFC 1939 e IMAP4 (Interactive Mail Access Protocol) RFC 2060

Agentes de transferencia de distribución (SMTP)

- El SMTP es un sencillo protocolo **cliente/servidor**. Establecida una comunicación **TCP** entre la computadora transmisora del correo, que opera como cliente, y el **puerto 25** del servidor de correo destino, el cliente permanece a la espera de recibir un mensaje del servidor.





Agentes de transferencia de distribución (SMTP):

- Las especificaciones básicas del protocolo SMTP indican que todos los caracteres enviados están codificados mediante el código ASCII de 7 bits y que el 8º bit sea explícitamente cero.
- El tamaño máximo permitido para estas líneas era de 1000 caracteres.
- Por lo tanto, para enviar caracteres acentuados es necesario recurrir a algoritmos que se encuentren dentro de las especificaciones MIME (se verá más adelante en este mismo tema).

Agentes de transferencia de distribución (SMTP):

- El objetivo principal del SMTP es transmitir correo entre servidores de correo mediante una conexión punto a punto..
- El servidor comienza por enviar una línea de texto que proporciona su identidad e indica si está preparado o no para recibir correo:
 - **a.-Si no lo está**, el cliente libera la conexión y lo intenta después.
 - **b.-Si está dispuesto** a aceptar correo electrónico, el cliente anuncia de quién viene el mensaje, y a quién está dirigido. Si existe tal destinatario en el destino, el servidor da al cliente permiso para enviar el mensaje. Entonces el cliente envía el mensaje y el servidor acusa su recibo. Una vez que todo el correo ha sido intercambiado **en ambas direcciones**, se libera la conexión.

Problemas en SMTP:

- Uno de los mayores problemas del protocolo SMTP es que **no requiere autenticación**. Esto permite que cualquiera en la Internet puede enviar correo a cualquiera otra persona o a grandes grupos de personas.
- Esta característica de SMTP es lo que **hace posible el correo basura o spam**.
- Los servidores SMTP modernos intentan minimizar este comportamiento permitiendo que sólo los hosts conocidos accedan al servidor SMTP.
- Los servidores que no ponen tales restricciones son llamados servidores open relay.



➤ Comandos SMTP: cliente

Comando	Descripción
HELO (ahora EHLO)	Identifica el remitente al destinatario.
MAIL FROM	Identifica una transacción de correo e identifica al emisor.
RCPT TO	Se utiliza para identificar un destinatario individual . Si se necesita identificar múltiples destinatarios es necesario repetir el comando.
DATA	Permite enviar una serie de líneas de texto. El tamaño máximo de una línea es de 1.000 caracteres. Cada línea va seguida de un retorno de carro y avance de línea <CR><LF>. La última línea debe llevar únicamente el carácter punto ":" seguido de <CR><LF>.
RSET	Aborta la transacción de correo actual.
NOOP	No operación. Indica al extremo que envíe una respuesta positiva. Keepalives
QUIT	Pide al otro extremo que envíe una respuesta positiva y cierre la conexión.
VRFY	Pide al receptor que confirme que un nombre identifica a un destinatario valido.
EXPN	Pide al receptor la confirmación de una lista de correo y que devuelva los nombres de los usuarios de dicha lista.
HELP	Pide al otro extremo información sobre los comandos disponibles.
TURN	El emisor pide que se inviertan los papeles , para poder actuar como receptor. El receptor puede negarse a dicha petición.
SOML	Si el destinatario está conectado, entrega el mensaje directamente al terminal, en caso contrario lo entrega como correo convencional.
SAML	Entrega del mensaje en el buzón del destinatario. En caso de estar conectado también lo hace al terminal.
SEND	Si el destinatario está conectado, entrega el mensaje directamente al terminal.



➤ Códigos de respuesta SMTP: servidor

<u>Código</u>	<u>Descripción</u>
211	Estado del sistema.
214	Mensaje de ayuda.
220	Servicio preparado.
221	Servicio cerrando el canal de transmisión.
250	Solicitud completada con éxito.
251	Usuario no local, se enviará a <dirección de reenvío>
354	Introduzca el texto, finalice con <CR><LF>.<CR><LF>.
421	Servicio no disponible.
450	Solicitud de correo no ejecutada, servicio no disponible (buzón ocupado).
451	Acción no ejecutada, error local de procesamiento.
452	Acción no ejecutada, insuficiente espacio de almacenamiento en el sistema.
500	Error de sintaxis, comando no reconocido.
501	Error de sintaxis. P.ej contestación de SMTP a ESMTP
502	Comando no implementado.
503	Secuencia de comandos errónea.
504	Parámetro no implementado.
550	Solicitud no ejecutada, buzón no disponible.
551	Usuario no local, pruebe <dirección de reenvío>. Si no se tiene cuenta
552	Acción de correo solicitada abortada.
553	Solicitud no realizada (error de sintaxis).
554	Fallo en la transacción.



Ejemplo SMTP – Captura Wireshark

EL CORREO ELECTRÓNICO

```
cliente -> serv_correo TCP 3612 > 25 [SYN] Seq=0 Win=64512 Len=0 MSS=1460
serv_correo -> cliente TCP 25 > 3612 [SYN, ACK] seq=0 Ack=1 win=65535 Len=0 MSS=1460
cliente -> serv_correo TCP 3612 > 25 [ACK] Seq=1 Ack=1 win=64512 Len=0
serv_correo -> cliente SMTP Response: 220 SERV-CORREO Spectrum Server 1.0 ESMTP ready
cliente -> serv_correo SMTP Command: EHLO alfon
serv_correo -> cliente SMTP Response: 250-SERV-CORREO
cliente -> serv_correo SMTP Command: AUTH LOGIN
serv_correo -> cliente SMTP Response: 334 VXNlcmShbwU6
cliente -> serv_correo SMTP Command: XXXzdGVtYXNlXXxhbmx1LmVz
serv_correo -> cliente SMTP Response: 334 UGFzc3dvcmQ6
cliente -> serv_correo SMTP Command: XXXETUFGXXXW3Q==_
serv_correo -> cliente TCP 25 > 3612 [ACK] Seq=243 Ack=71 win=65465 Len=0
serv_correo -> cliente SMTP Response: 235 2.0.0 Authentication successful
cliente -> serv_correo SMTP Command: MAIL FROM: <alfon@miservidor.com>
serv_correo -> cliente SMTP Response: 250 2.1.0 Sender <alfon@miservidor.com> ok
cliente -> serv_correo SMTP Command: RCPT TO: <destinatario@otroservidor.com>
serv_correo -> cliente SMTP Response: 250 2.1.5 Recipient <alfon@miservidor.com> ok (local)
cliente -> serv_correo SMTP Command: DATA
serv_correo -> cliente SMTP Response: 354 Enter mail, end with CRLF.CRLF
cliente -> serv_correo SMTP DATA fragment, 1460 bytes
cliente -> serv_correo SMTP DATA fragment, 740 bytes
serv_correo -> cliente TCP 25 > 3612 [ACK] Seq=411 Ack=3061 win=65535 Len=0
cliente -> serv_correo SMTP EOM:
serv_correo -> cliente TCP 25 > 3612 [ACK] Seq=411 Ack=4521 win=65535 Len=0
serv_correo -> cliente SMTP Response: 250 2.0.0 48456cd9-0001e6ea Message accepted for delivery
cliente -> serv_correo SMTP Command: QUIT
serv_correo -> cliente SMTP Response: 221 2.0.0 SMTP closing connection
serv_correo -> cliente TCP 25 > 3612 [FIN, ACK] Seq=505 Ack=6732 win=65524 Len=0
cliente -> serv_correo TCP 3612 > 25 [ACK] Seq=6732 Ack=506 win=64008 Len=0
cliente -> serv_correo TCP 3612 > 25 [FIN, ACK] Seq=6732 Ack=506 win=64008 Len=0
serv_correo -> cliente TCP 25 > 3612 [ACK] Seq=506 Ack=6733 win=65524 Len=0
```

EL CORREO ELECTRÓNICO

- Las tres primeras líneas corresponden al inicio de la comunicación con el servidor de correo.
- Establecimiento de conexión en tres pasos.
 - El número de secuencia inicial es relativo y por tanto en este caso **0**.

Traza con número de secuencia inicial aleatorio:

cliente -> servidor TCP 1125 > smtp [SYN] Seq=13766490 Ack=0 Win=16384 Len=0

servidor -> cliente TCP smtp > 1125 [SYN, ACK] Seq=472370892 Ack=13766491 Win=8736 Len=0

cliente -> servidor TCP 1125 > smtp [ACK] Seq=13766491 Ack=472370893 Win=17472 Len=0

1. **cliente** envia **SYN** con un número aleatorio Seq=13766490 y Ack=0
2. **servidor** lo recibe, envia **SYN-ACK** y responde con su propio número de secuencia y con un ACK = al número de secuencia anterior + 1, es decir: Seq=472370892 Ack=13766491
3. **cliente** a su vez responde con **ACK** y número de secuencia inicial (Seq=13766490) +1 y ACK = número de secuencia anterior (472370892) +1, es decir: Seq=13766491 Ack=472370893



EL CORREO ELECTRÓNICO

Inmediatamente el servidor de correo responde con un código **220** que está listo:

SMTP Response: 220 SERV-CORREO Spectrum Server 1.0 ESMTP ready

Responde el cliente iniciando el diálogo y sesión **SMTP** con el servidor:

SMTP Command: EHLO alfon

Responde el servidor de correo aceptando con un código **250** y el nombre del servidor:

SMTP Response: 250-SERV-CORREO

El cliente inicia la autenticación con el servidor. (Servidor autenticado):

SMTP Command: AUTH LOGIN



EL CORREO ELECTRÓNICO

Responde el servidor pidiendo **Username** y un código **334**, en este caso **codificado** (Base64). En caso de fallo de autentificación se responde con un código **535**. el cliente envia el usuario;

SMTP Response: 334 VXNlXXXhbWU6

SMTP Command: XXXzdGVtYXNAYWJhbmNILmVz

El servidor pide **Password** el el cliente responde. Todo ello codificado en Base64.:

SMTP Response: 334 UGFzc3dvcmQ6

SMTP Command: XXXETUFOXXXwJQ==

Todo salió bien:

SMTP Response: 235 2.0.0 Authentication successfuk



EL CORREO ELECTRÓNICO

Indicamos al servidor el inicio del mensaje de correo con **MAIL FROM**. Indicamos remitente y destinatario (**RCPT TO**):

SMTP Command: MAIL FROM:

SMTP Response: 250 2.1.0 Sender ok

SMTP Command: RCPT TO:

SMTP Response: 250 2.1.5 Recipient ok (local)

Con **DATA** indicamos al servidor que lo que viene a continuación es el texto del mensaje. El servidor responde aceptando con un código **354** indicando que podemos introducir el mensaje y enviamos los paquetes correspondientes al texto de mensaje:

SMTP Response: 354 Enter mail, end with CRLF.CRLF

SMTP DATA fragment, 1460 bytes

SMTP DATA fragment, 740 bytes



EL CORREO ELECTRÓNICO

Una vez que finalizamos la transmisión de los paquetes del mensaje y con el comando **EOM** indica el cliente el fin del mensaje:

SMTP EOM

El servidor se da por enterado y acepta el mensaje:

SMTP Response: 250 2.0.0 48456cd9-0001a6ae Message accepted for delivery

El cliente indica ahora que no hay más operaciones a realizar que se puede cerrar la conexión y el servidor se da por enterado y cierra conexión con el cliente:

SMTP Command: QUIT

SMTP Response: 221 2.0.0 SMTP closing connection

EL CORREO ELECTRÓNICO

Cierre de conexión TCP. Al igual que al principio se inicia una petición de conexión por parte del cliente con un diálogo en tres fases, ahora se da por terminada la conexión pero en cuatro pasos. Esto es debido a que cada una de las “direcciones” debe ser cerrada independientemente:

serv_correo -> cliente TCP 25 > 3612 [FIN, ACK] Seq=505 Ack=6732 Win=65524 Len=0

cliente -> serv_correo TCP 3612 > 25 [ACK] Seq=6732 Ack=506 Win=64008 Len=0

cliente -> serv_correo TCP 3612 > 25 [FIN, ACK] Seq=6732 Ack=506 Win=64008 Len=0

serv_correo -> cliente TCP 25 > 3612 [ACK] Seq=506 Ack=6733 Win=65524 Len=0

A tener en cuenta en SMTP:

- La sintaxis de los comandos del cliente se especifica con rigidez.
- La sintaxis de las respuestas del servidor es menos rígida, sólo cuenta el código numérico, pudiendo cada implementación del protocolo SMTP poner la cadena de texto que desee después del código numérico.
- Algunas implementaciones más viejas de SMTP no pueden manejar mensajes mayores de 64 Kbytes.
- Si el cliente y el servidor tienen temporizaciones distintas, uno de ellos puede terminar mientras que el otro continúa trabajando, terminando inesperadamente la conexión.
- **Solución**, un nuevo protocolo extendido: **SMTP extendido (ESMTP)** en el RFC 1425. Los clientes que deseen usarlo deben enviar un mensaje **EHLO**, en lugar de HELO. Si el saludo se rechaza, **código 500**, esto indica que el servidor es un servidor SMTP normal (basado en el RFC 821) y el cliente debe proceder de la manera normal

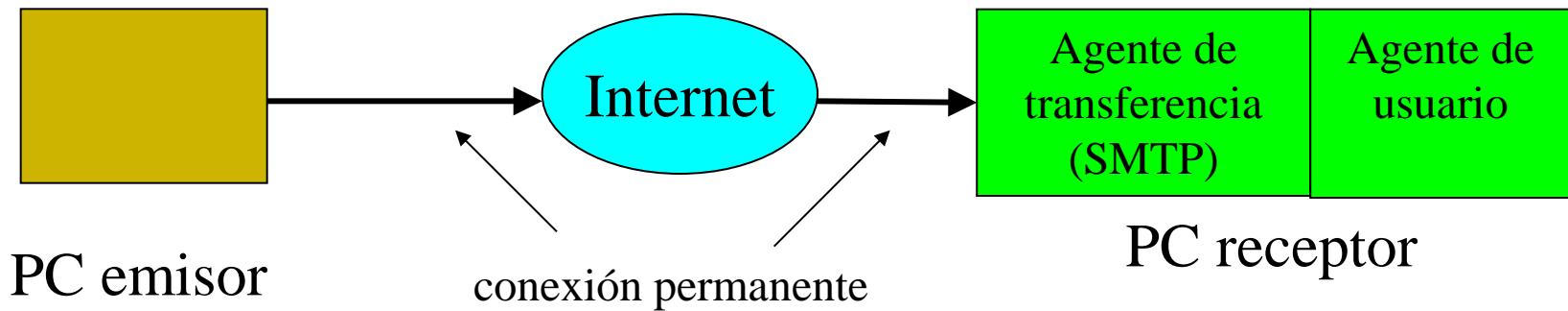
Agentes de transferencia de entrega final de usuario:

- En muchas compañías los usuarios trabajan en un PC de escritorio que no está en Internet (no tienen su dominio propio) y que es incapaz de enviar o recibir correo. Sin embargo, la compañía puede tener uno o más servidores SMTP que pueden enviar y recibir correo electrónico.
- Para poder recibir el correo electrónico, un PC debe comunicarse con un servidor de correo electrónico usando algún protocolo de entrega. Esta comunicación se realiza explícitamente bajo indicación del usuario.
- Para enviar un correo, se debe conectar al servidor de correo local (se realiza habitualmente por SMTP).

Agentes de transferencia de entrega final de usuario:

- El correo entrante en un cliente se puede obtener básicamente a través de los siguientes protocolos:
 - **POP3** (Post Office Protocol) RFC 1939 objetivo del POP3 es obtener correo electrónico del **buzón remoto y almacenarlo en la máquina local del usuario** para su lectura posterior (por defecto se borra automáticamente el mensaje en el servidor de correo).
 - **IMAP** (Interactive Mail Access Protocol) RFC 2060 **No copia el correo electrónico en la máquina personal del usuario dado que el usuario puede tener varias computadoras para consultar el correo**, y observa si sus correos han sido leídos con anterioridad

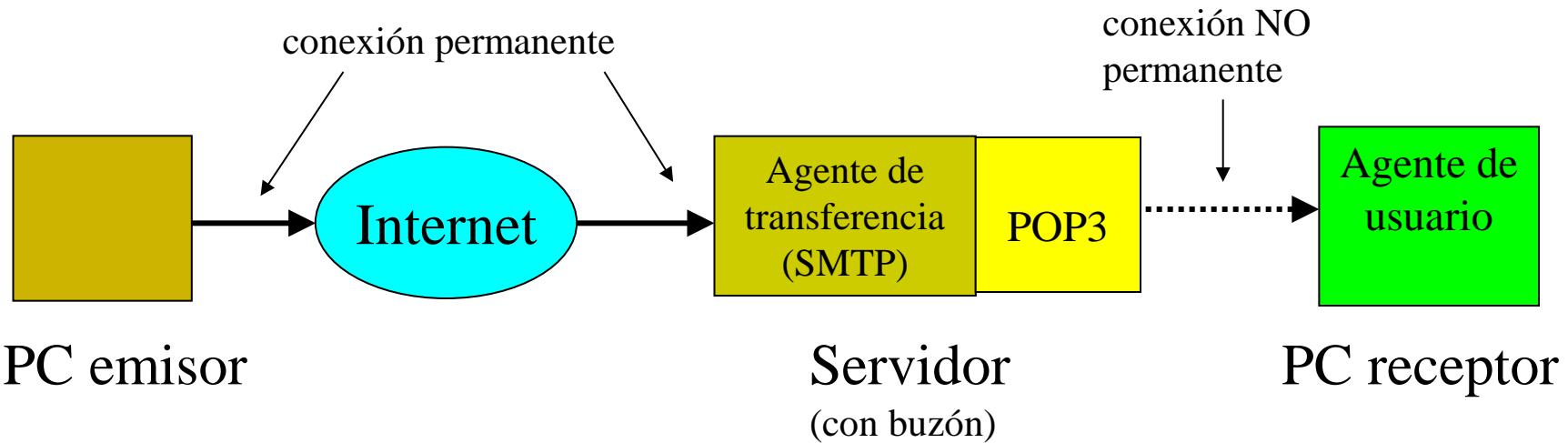
Agentes de transferencia de entrega final de usuario:



Problema: acceso no permanente a Internet

- a través de un ISP
- PC servidor

Agentes de transferencia de entrega final de usuario:



Problema: obtener correo del buzón

Solución: POP3



POP3:

- **POP3** (Post Office Protocol) permite recoger el correo electrónico en un servidor remoto (servidor POP).
- Existen dos versiones principales de este protocolo, POP2 y POP3, a los que se le asignan los puertos 109 y 110 respectivamente, y que funcionan utilizando comandos de texto radicalmente diferentes.
- POP3 se define en la **RFC 1939**.
- Tiene comandos para que un usuario establezca una sesión (USER y PASS), la termine (QUIT), obtenga mensajes (RETR) y los borre (DELE).
- El protocolo mismo consiste en texto ASCII y se asemeja a SMTP.
- El objetivo del POP3 es obtener correo electrónico del **buzón remoto y almacenarlo en la máquina local del usuario** para su lectura posterior. *Existen versiones actualmente, que ya permiten no descargar el correo del buzón como IMAP*



EL CORREO ELECTRÓNICO

Comandos POP3

Comando	Descripción
USER identification	Este comando permite la autenticación. Debe estar seguido del nombre de usuario, es decir, una cadena de caracteres que identifique al usuario en el servidor. El comando <i>USER</i> debe preceder al comando <i>PASS</i> .
PASS password	El comando <i>PASS</i> permite especificar la contraseña del usuario cuyo nombre ha sido especificado por un comando <i>USER</i> previo.
STAT	Información acerca de los mensajes del servidor
RETR	Número del mensaje que se va a recoger
DELE	Número del mensaje que se va a eliminar
LIST [msg]	Número del mensaje que se va a mostrar
NOOP	Permite mantener la conexión abierta en caso de inactividad
TOP <messageID> <n>	Comando que muestra <i>n</i> líneas del mensaje, cuyo número se da en el argumento. En el caso de una respuesta positiva del servidor, éste enviará de vuelta los encabezados del mensaje, después una línea en blanco y finalmente las primeras <i>n</i> líneas del mensaje.
UIDL [msg]	Solicitud al servidor para que envíe una línea que contenga información sobre el mensaje que eventualmente se dará en el argumento. Esta línea contiene una cadena de caracteres denominada <i>unique identifier listing (lista de identificadores únicos)</i> que permite identificar de manera única el mensaje en el servidor, independientemente de la sesión. El argumento opcional es un número relacionado con un mensaje existente en el servidor POP, es decir, un mensaje que no se ha borrado.
QUIT	El comando <i>QUIT</i> solicita la salida del servidor POP3. Lleva a la eliminación de todos los mensajes marcados como eliminados y envía el estado de esta acción.



Ejemplo POP3

```
S: +OK servidor POP3 listo
C: USER carolina
                S: +OK
C: PASS vegetales
                S: +OK inicio de sesión exitoso

C: LIST
                S: 1 2505
                S: 2 14302
                S: 3 8122
                S: .

C: RETR 1
                S: (envía mensaje 1)
C: DELE 1
C: RETR 2
                S: (envía mensaje 2)

C: DELE 2
C: RETR 3
                S: (envía mensaje 3)
C: DELE 3
C: QUIT
                S: +OK servidor POP3 desconectándose
```

EL CORREO ELECTRÓNICO

POP3:

- POP3 inicia cuando el usuario arranca el lector de correo. Éste llama al servidor y establece una conexión TCP con el agente de transferencia de mensajes en el **puerto 110**. Una vez se ha establecido la conexión el protocolo POP3 pasa por tres estados:
 - Autorización
 - Transacciones
 - Actualización
- El protocolo POP3 administra la autenticación utilizando el nombre de usuario y la contraseña. Sin embargo, esto no es seguro, ya que las contraseñas, al igual que los correos electrónicos, circulan por la red como texto sin codificar (de manera no cifrada).
- Según **RFC 1939**, es posible cifrar la contraseña utilizando un algoritmo **MD5** y beneficiarse de una autenticación segura. Sin embargo, debido a que este comando es opcional, pocos servidores lo implementan.



POP3:

- Además, el protocolo POP3 bloquea las bandejas de entrada durante el acceso, lo que significa que es imposible que dos usuarios accedan de manera simultánea a la misma bandeja de entrada.

- Para añadir seguridad a POP3, es posible utilizar la encriptación Secure Socket Layer (SSL) para la autenticación del cliente y las sesiones de transferencias de datos.



IMAP:

- IMAP (Internet Message Access Protocol) es un protocolo alternativo al de POP3, pero que ofrece más posibilidades.
- Fue diseñado por Mark Crispin en el año 1986 como protocolo de correo electrónico de acceso remoto.
- La idea en la que se basa IMAP es que el servidor de correo electrónico mantenga un depósito central al que puede accederse desde cualquier máquina. Por tanto, a diferencia del POP3, **no copia el correo electrónico en la máquina personal del usuario dado que el usuario puede tener varias computadoras para consultar el correo**, y observa si sus correos han sido leídos con anterioridad
- IMAP supone que todo el correo electrónico permanecerá en el servidor de manera indefinida.



IMAP:

- IMAP2 (Interactive Mail Access Protocol) se definió en la RFC 1064 (en 1988) y posteriormente fué actualizado por RFC 1176 (en 1990).
- IMAP2 introdujo el etiquetado comando/respuesta y fué la primera versión que se distribuyó públicamente.
- IMAP3 es una variante rara extinguida de IMAP que se publicó en la RFC 1203 (en 1991).
- El IETF formó un grupo de trabajo a principios de 1990 para crear IMAP4.
- IMAP4 se definió en la RFC 2060. La última revisión está en la RFC 3501 (IMAP, 2003)
- IMAP permite administrar diversos accesos de manera simultánea.
- IMAP permite administrar diversas bandejas de entrada



IMAP:

- IMAP brinda más criterios que pueden utilizarse para ordenar los correos electrónicos
- IMAP proporciona los mecanismos para crear, borrar, renombrar y manipular buzones en el servidor.
- El protocolo es conveniente también para usuarios que se estén conectando al servidor de correo a través de una conexión lenta, porque sólo la información de la cabecera del correo es descargada para los mensajes, hasta que son abiertos, ahorrando de esta forma ancho de banda.
- Para seguridad adicional, es posible utilizar la encriptación SSL para la autenticación de clientes y para las sesiones de transferencia de datos.
- Un proceso cliente IMAP se comunica con el proceso servidor IMAP identificado con el número de **puerto 220 TCP**.



Diferencias importantes entre IMAP y POP3:

- Respaldo para los modos de operación en línea y fuera de línea
 - Al utilizar POP3, los clientes se conectan brevemente al servidor de correo, solamente el tiempo que les tome descargar los nuevos mensajes. Al utilizar IMAP4, los clientes permanecen conectados el tiempo que su interfaz permanezca activa y descargan los mensajes bajo demanda. Esta manera de trabajar de IMAP4 puede dar tiempos de respuesta más rápidos para usuarios que tienen una gran cantidad de mensajes o mensajes grandes.
- Respaldo para la conexión de múltiples clientes simultáneos a un mismo destinatario
 - El protocolo POP3 supone que el cliente conectado es el único dueño de una cuenta de correo. En contraste, el protocolo IMAP4 permite accesos simultáneos a múltiples clientes y proporciona ciertos mecanismos a los clientes para que se detecten los cambios hechos a un buzón de correo por otro cliente concurrentemente conectado.



Diferencias importantes entre IMAP y POP3:

- Respaldo para acceso a partes MIME de los mensajes y obtención parcial
 - Casi todo el correo electrónico de Internet es transmitido en formato MIME. El protocolo IMAP4 les permite a los clientes obtener separadamente cualquier parte MIME individual, así como obtener porciones de las partes individuales o los mensajes completos. Es más seguro.
- Respaldo para que la información de estado del mensaje se mantenga en el servidor
 - A través de la utilización de señales definidas en el protocolo IMAP4 de los clientes, se puede vigilar el estado del mensaje, por ejemplo, si el mensaje ha sido o no leído, respondido o eliminado. Estas señales se almacenan en el servidor, de manera que varios clientes conectados al mismo correo en diferente tiempo pueden detectar los cambios hechos por otros clientes.



Diferencias importantes entre IMAP y POP3:

- Respaldo para accesos múltiples a los buzones de correo en el servidor
 - Los clientes de IMAP4 pueden crear, renombrar o eliminar correo (por lo general presentado como carpetas al usuario) del servidor, y mover mensajes entre cuentas de correo. El soporte para múltiples buzones de correo también le permite al servidor proporcionar acceso a los directorios públicos y compartidos.
- Respaldo para búsquedas de parte del servidor
 - IMAP4 proporciona un mecanismo para que los clientes pidan al servidor que busque mensajes de acuerdo a una cierta variedad de criterios. Este mecanismo evita que los clientes descarguen todos los mensajes de su buzón de correo, agilizando, de esta manera, las búsquedas.

Diferencias importantes entre IMAP y POP3:

- Respaldo para un mecanismo de extensión definido
 - Como reflejo de la experiencia en versiones anteriores de los protocolos de Internet, IMAP4 define un mecanismo explícito mediante el cual puede ser extendido. Se han propuesto muchas extensiones de IMAP4 y son de uso común.
 - Un ejemplo de extensión es el IMAP IDLE, que sirve para que el servidor avise al cliente cuando ha llegado un nuevo mensaje de correo y éstos se sincronicen. Sin esta extensión, para realizar la misma tarea, el cliente debería contactar periódicamente al servidor para ver si hay mensajes nuevos.



Comparación entre POP3 e IMAP4:

Características	POP3	IMAP4
RFC	RFC 1939	RFC 2060
Puerto TCP utilizado	110	143
Dónde se almacena el correo electrónico	PC del usuario	Servidor
Tiempo de conexión requerido	Poco	Mucho
Uso de recursos del servidor	Mínimo	Amplio
Quién mantiene los buzones	Usuario	ISP
Bueno para usuarios móviles	No	Si
Descargas parciales de mensajes	No	Si
Sencillo de implementar Soporte amplio	Si	No



Agentes de Usuario



Agentes de usuario:

- Un agente de usuario es una aplicación informática que funciona como cliente en un protocolo de red; el nombre se aplica generalmente para referirse a aquellas aplicaciones que acceden a la WWW.
- Está diseñado para recoger y enviar correo electrónico
- Un agente de usuario acepta una variedad de comandos para componer, recibir y contestar los mensajes, así como para manipular los buzones de correo.
- Las labores principales de los agentes de usuario pueden dividirse en su capacidad para procesar y enviar correo electrónico y su capacidad para recibir el mismo.
- Existen diversos tipos de agentes de usuario para el correo electrónico:
 - Una aplicación cliente de correo electrónico o **MUA** (Mail User Agent)
 - **Una interfaz web**, a la que se accede con un navegador web.

Agentes de usuario:

- El correo web es cómodo porque permite ver y almacenar los mensajes desde cualquier sitio en vez de en un ordenador personal concreto.
- El problema del correo web es que es difícil de ampliar con otras funcionalidades, porque el sitio ofrece un conjunto de servicios concretos y no permite cambiarlos.
- Suele ser más lento que un programa de correo, ya que debe estar continuamente conectado al sitio web y leer los correos de uno en uno.
- Sin embargo, el cliente de correo electrónico es más rápido (permite almacenar localmente los correos electrónicos) no teniendo que estar conectado continuamente para leerlos, y permite incorporar potentes filtros anti-correo no deseado.

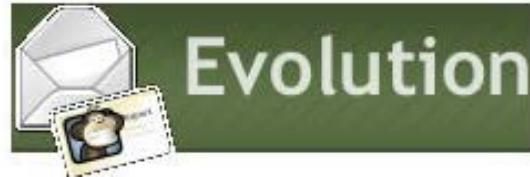
EL CORREO ELECTRÓNICO

Ejemplos de clientes de correo electrónico:

- Windows Live Mail (Windows)
- Evolution (Linux)
- AirMail (Mac OS X e iOS)
- Outlook Express (Windows)
- Thunderbird (Windows, Linux, Mac OS X).



Windows Live™



AirMail



Thunderbird



Outlook



Formato de los mensajes. RFC 822

EL CORREO ELECTRÓNICO

Formato de los mensajes. RFC 822

- ✓ Estándar para mensajes de texto creado en agosto de 1982.
- ✓ Los mensajes con formato RFC 822 están formados por una **envoltura primitiva** (descrita en el **RFC 821**), *algunos campos de cabecera, una línea en blanco, y el cuerpo del mensaje*. Cada campo de cabecera consiste en una sola línea de texto ASCII que contiene el nombre del campo, dos puntos (:) y, para la mayoría de los campos un valor.
- ✓ Algunos campos del encabezado del mensaje se completan de forma automática por **el User Agent**: From, Date, Received. Otros son completados por **el usuario**: To, Subject, Bcc..
- ✓ Los campos Received sólo aparecen en el mail al llegar al destino, pues contiene el nombre, IP y otros datos que son completados por los MTA por los que pasó el mensaje.
- ✓ El campo Bcc sólo se ve en el mail emisor, o sea en el mail que queda copiado en la carpeta “Mensajes Enviados”.

EL CORREO ELECTRÓNICO

- ✓ El RFC 822 en principio sólo soportaba TEXTO en el cuerpo del mensaje. Para poder enviar otro tipo de formatos se extendió dicho RFC con las **extensiones MIME**.

Cabecera
del
mensaje



RFC-822

Cuerpo
del
mensaje



MIME RFC-1345



Campos principales del RFC 822:

<u>Cabecera</u>	<u>Descripción</u>
To:	Direcciones de email de los destinatarios primarios.
Cc:	Direcciones de email de los destinatarios secundarios. En términos de entrega no existe diferencia con los destinatarios primarios.
Bcc:	Direcciones de email de las copias al carbón ciegas. Es como el campo anterior excepto que esta línea se borra de todas las copias enviadas a los destinatarios primarios y secundarios.
From:	Persona o personas que crearon el mensaje.
Sender:	Dirección de correo del remitente. <i>Puede omitirse si es igual al campo anterior.</i>
Received:	Línea agregada por cada agente de transferencia en la ruta . La línea contiene la identidad del agente, la fecha y hora de recepción del mensaje y otra información que puede servir para detectar fallos en el sistema de enrutamiento. Se añaden apiladas en la cabecera, a medida que se intercambia el email.
Return-Path:	Puede usarse para identificar una trayectoria de regreso al remitente.

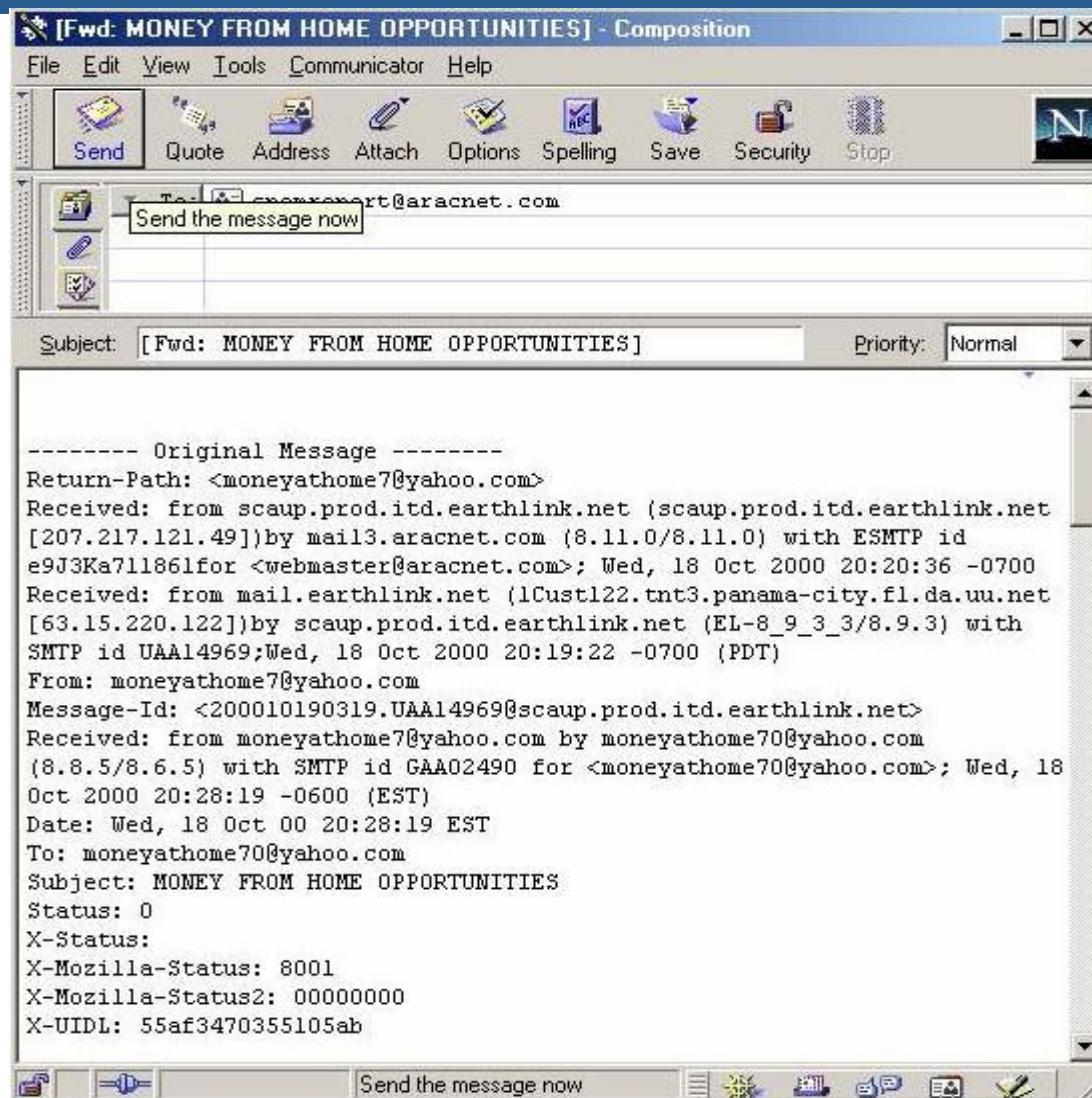
EL CORREO ELECTRÓNICO

Los mensajes RFC 822 pueden contener una variedad de campos auxiliares de **cabecera** usados por los agentes de usuario o los destinatarios.

Cabecera	Descripción
Date:	Fecha y hora de envío del mensaje.
Reply-To:	Se usa cuando la persona que escribió el mensaje y la que lo envió no desean ver la respuesta.
Message-Id:	Número único para referencia posterior a este mensaje. Suele estar compuesto por un número y la dirección de email completa del usuario que lo manda.
In-Reply-To:	Identificador del mensaje al que éste corresponde.
References:	Otros identificadores de mensaje.
Keywords:	Claves seleccionadas por el usuario.
Subject:	Resumen corto del mensaje para exhibir en una línea.

- ✓ El RFC 822 explícitamente indica que los usuarios pueden inventar cabeceras nuevas para uso privado siempre y cuando **comiencen con la cadena X-**.

EL CORREO ELECTRÓNICO





Multipurpose Internet Mail Protocol Extensions (MIME):

- En 1991 la IETF comenzó a desarrollar esta norma y desde 1994 todas las extensiones MIME están especificadas de forma detallada en diversos documentos oficiales disponibles en Internet.
- El RFC 822 estaba pensado inicialmente para texto en ASCII 7 bits pero este sistema presenta problemas
 - Mensajes en idiomas con acentos (español, francés y alemán).
 - Mensajes en alfabetos no latinos (hebreo y ruso).
 - Mensajes en idiomas sin alfabetos (chino y japonés).
 - Mensajes que no contienen texto (audio y vídeo).
- MIME (Multipurpose Internet Mail Extensions) RFC 1341 y RFC 1521 mantiene la idea básica de continuar usando el RFC 822, pero permite agregar una estructura al cuerpo del mensaje y definir reglas de codificación para los mensajes no ASCII.
- MIME sólo afecta a los agentes de usuario, ya que para SMTP es totalmente transparente.

Multipurpose Internet Mail Protocol Extensions (MIME):

- Nada cambia respecto a la arquitectura de correo anterior.
- Las extensiones de MIME van encaminadas a soportar:
 - Texto en conjuntos de caracteres distintos de US-ASCII;
 - Adjuntos que no son de tipo texto;
 - Cuerpos de mensajes con múltiples partes (multi-part);
 - Información de encabezados con conjuntos de caracteres distintos de ASCII.
- MIME está especificado en seis RFCs: RFC 2045, RFC 2046, RFC 2047, RFC 4288, RFC 4289 y RFC 2077.



➤ Cabeceras de mensajes MIME

Cabecera	Descripción
MIME-Version:	Identifica la versión de MIME. Si no existe se considera que el mensaje es texto normal en inglés.
Content-Description:	Cadena de texto que describe el contenido. Esta cadena es necesaria para que el destinatario sepa si desea descodificar y leer el mensaje o no.
Content-Id:	Identificador único, usa el mismo formato que la cabecera estándar Message-Id.
Content-Transfer-Encoding:	Indica la manera en que está envuelto el cuerpo del mensaje.
Content-Type:	Especifica la naturaleza del cuerpo del mensaje.

➤ Content-Transfer-Encoding

- Indica la manera en que está envuelto el cuerpo para su transmisión, ya que podría haber problemas con la mayoría de los caracteres distintos de letras, números y signos de puntuación.
- Existen 5 tipos de codificación (RFC1521) : *ASCII 7, ASCII 8, codificación binaria, base64 y entrecomillada-imprimible.7.2*

EL CORREO ELECTRÓNICO

MIME: Content-Transfer-Encoding. Esquemas de codificación:

- 1.-**ASCII de 7 bits** y ninguna línea excede de 1000 caracteres.
- 2.-**ASCII de 8 bits**. Este esquema viola el protocolo original del correo electrónico. Ninguna línea debe exceder de 1000 caracteres.
- 3.- **Binaria**. Utilizan los 8 bits y no respetan el límite de 1000 caracteres por línea. Los programas ejecutables caen en esta categoría. No se da ninguna garantía de que los mensajes en binario llegarán correctamente.
- 4.- **Base64** o armadura ASCII. En este esquema, se dividen grupos de 24 bits en unidades de 6 bits (**26 mayúsculas, 26 minúsculas, 10 dígitos y + y / de forma 'A' es 0, 'B' es 1, ..., 'a' es 26, ...**), enviándose cada unidad como carácter ASCII legal. Las secuencias == y = se usan para indicar que el último grupo contenía solo 6 o 12 bits, respectivamente.

Los retornos de carro y avances de línea se ignoran, por lo que pueden introducirse a voluntad para mantener la línea lo suficientemente corta



MIME: Content-Transfer-Encoding. Esquemas de codificación:

5.- **Entrecomiñada-imprimible (QUOTED-PRINTABLE)**. Esta codificación es ASCII de 7 bits, con todos los caracteres por encima de 127 codificados como un signo de igual seguido del valor del carácter en dos dígitos hexadecimales. Se utiliza en el caso de mensajes que son casi completamente ASCII, pero con algunos caracteres no ASCII. En este caso la codificación base64 es algo ineficiente.

MIME. Content-Type

- *Content-Type* especifica la forma del cuerpo del mensaje. Existen 7 tipos definidos en el RFC 1521, cada uno de los cuales tiene uno o más subtipos. El tipo y el subtipo se separan mediante un carácter diagonal (/), ej:
Content-Type: video/mpeg



EL CORREO ELECTRÓNICO

MIME: Content-Type: tipos y subtipos

- La lista inicial de tipos y subtipos especificada por el RFC 1521 es:

Tipo	Subtipo	Descripción
Text	Plain	Texto sin formato.
	Richtext	Texto con comandos de formato sencillos.
Image	Gif	Imagen fija en formato GIF.
	Jpeg	Imagen fija en formato JPEG.
Audio	Basic	Sonido.
Video	Mpeg	Película en formato MPEG.
Application	Octet-stream	Secuencia de bytes no interpretada.
	Postscript	Documento imprimible PostScript.
Message	Rfc822	Mensaje MIME RFC 822.
	Partial	Mensaje dividido para su transmisión.
	External-body	El mensaje mismo debe obtenerse de la red.
Multipart	Mixed	Partes independientes en el orden especificado.
	Alternative	Mismo mensaje en diferentes formatos.
	Parallel	Las partes deben verse simultáneamente.
	Digest	Cada parte es un mensaje RFC 822 completo.

MIME: Content-Type: tipo text:

- *text/plain* es para mensajes ordinarios que pueden visualizarse como se reciben, sin codificación ni ningún procesamiento posterior.
- *text/richtext* permite la inclusión de un lenguaje de marcación sencillo en el texto (*para indicar negritas, cursivas, tamaños ... Independiente del sistema*). Utiliza el lenguaje SGML (Standard Generalized Markup Language), también usado como base del HTML



MIME: Content-Type: tipo application:

- El tipo ***application*** es un tipo general para los formatos que requieren procesamiento externo no cubierto por ninguno de los otros tipos.
- El subtipo ***octet-stream*** simplemente es una secuencia de bytes no interpretados, tal que a su recepción, un agente de usuario debería *presentarla en la pantalla sugiriendo al usuario que se copie en un archivo y solicitando un nombre de archivo*.
- El subtipo ***postscript***, se refiere al lenguaje PostScript de Adobe Systems. Aunque un agente de usuario puede llamar a un intérprete PostScript externo para visualizarlo, hacerlo no está exento de riesgos al ser PostScript un lenguaje de programación completo.

EL CORREO ELECTRÓNICO

MIME: Content-Type: tipo message:

- El tipo ***message*** permite que un mensaje esté encapsulado por completo dentro de otro. Este esquema es útil para reenviar, correo electrónico.
- El subtipo ***rfc822*** se utiliza cuando se encapsula un mensaje RFC 822 completo en un mensaje exterior.
- El subtipo ***partial*** hace posible dividir un mensaje encapsulado en pedazos y enviarlos por separado. **Los parámetros hacen posible ensamblar correctamente todas las partes en el destino.** Ej: 1/3, 2/3, 3/3.
- El subtipo ***external-body*** puede usarse para mensajes muy grandes, por ejemplo películas de vídeo. En lugar de incluir el archivo mpeg en el mensaje, se da una dirección de FTP y el agente de usuario del receptor puede obtenerlo a través de la red cuando se requiera.



MIME: Content-Type: tipo multipart:

- El tipo es ***multipart***, que permite que un mensaje contenga más de una parte, con el comienzo y el fin de cada parte claramente delimitados.
- El subtipo ***mixed*** permite que cada parte sea diferente.
- El subtipo ***alternative*** indica que cada parte contiene el mismo mensaje, pero expresado en un medio o codificación diferente.
- El subtipo ***parallel*** se usa cuando todas las partes deben “verse” simultáneamente, por ejemplo, en los canales de audio y vídeo de las películas.
- El subtipo ***digest*** se usa cuando se juntan muchos mensajes en un mensaje compuesto.



➤ Listado de puertos relacionados con e-mail:

POP3 - port 110

IMAP - port 143

SMTP - port 25

HTTP - port 80

Secure SMTP (SSMTP) - port 465

Secure IMAP (IMAP4-SSL) - port 585

IMAP4 over SSL (IMAPS) - port 993

Secure POP3 (SSL-POP) - port 995



Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red
2. Servicio de Nombres de Dominio (DNS)
3. La navegación Web
4. El Correo electrónico
5. **Seguridad & protocolos seguros**
6. Aplicaciones multimedia
7. Aplicaciones para interconectividad de redes locales
8. Cuestiones y ejercicios



- **Seguridad, ¿para qué?**
- **Compartición** → muchos usuarios involucrados, más atacantes potenciales
- **Complejidad del sistema** → complejidad de los controles de seguridad
- **Límite desconocido** → identidad incierta de usuarios
- **Múltiples puntos de ataque** → mecanismos de protección en todo el camino de la información

➤ Primitivas de seguridad

- **Confidencialidad**
 - Sólo accede a la información quien debe hacerlo.
- **Responsabilidad**
 - Autenticación: Los agentes de la comunicación son quien dicen ser.
 - No repudio: No se puede negar el autor de una determinada acción.
 - Control de accesos: Garantía de identidad para el acceso.
- **Integridad**
 - Detección de alteraciones (intencionadas o no) de la información.
- **Disponibilidad**
 - Mantener las prestaciones de los servicios con independencia de la demanda.

- Ataques más comunes
- Rastreadores o sniffers
- Suplantaciones de IP o spoofing
- Ataques de contraseñas
- Control de salida ilegal de información sensible desde una fuente interna
- Ataques de hombre en el medio (o man-in-the-middle attacks)
- Ataques de denegación de servicio, Denial of Service o ataques DoS.
- Ataques a nivel de aplicación para explotar vulnerabilidades conocidas
- Caballos de Troya (Trojan Horses), virus y otros códigos maliciosos



- **Mecanismos de seguridad**
- **De prevención:**
 - mecanismos de autenticación e identificación
 - mecanismos de control de acceso
 - mecanismos de separación(física, temporal, lógica, criptográfica y fragmentación)
 - mecanismos de seguridad en las comunicaciones (cifrado de la información)
- **De detección:**
 - IDS (Intruder Detected System)
- **De recuperación:**
 - copias de seguridad (backup)
 - mecanismos de análisis forense: averiguar alcance, las actividades del intruso en el sistema y cómo entró

PROTOCOLOS SEGUROS

- Seguridad:
 - Seguridad (criptográfica) en protocolos:
 - Capa de aplicación
 - Pretty Good Privacy (PGP)
 - Secure Shell (SSH)
 - Capa de sesión (entre aplicación y transporte)
 - Secure Socket Layer (SSL) → HTTPS, IMAPS, SSL-POP, VPN
 - Transport Secure Layer (TSL)
<http://heartbleed.com/>
 - Capa de Red → IPSec (VPN)
 - Seguridad Perimetral:
 - *Firewalls*, sistemas de detección de intrusiones (IDS) y de respuesta (IRS)





Protocolos Seguros



➤ Resumen de esta sección:

- Conceptos
- Relación entre criptografía y redes
- Métodos básicos de cifrado en redes
 - Cifrado de enlace
 - Cifrado extremo a extremo

➤ Conceptos:

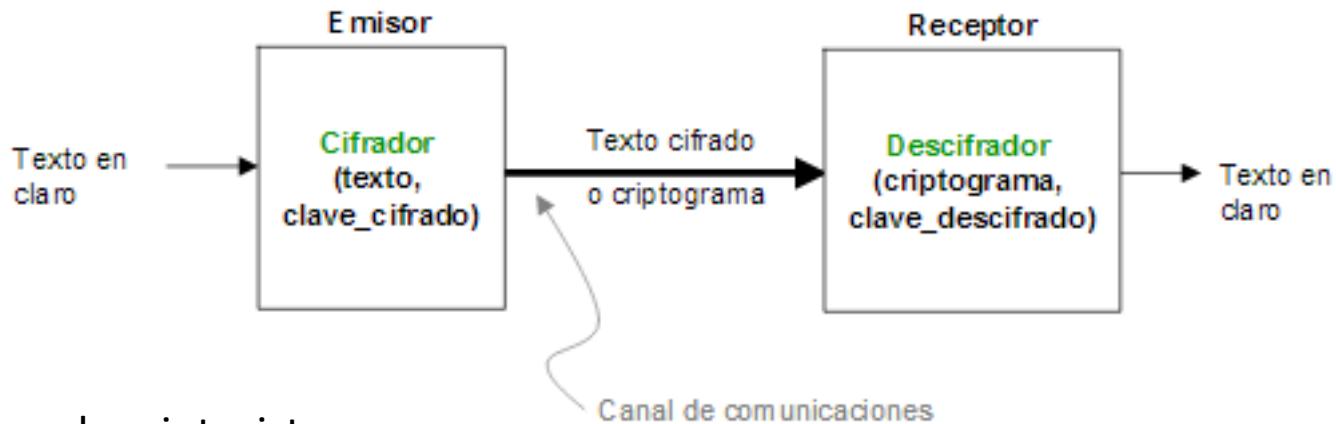
Criptología: (del griego krypto y logos, estudio de lo oculto, lo escondido) es la ciencia que trata los problemas teóricos relacionados con la seguridad en el intercambio de mensajes en clave entre un emisor y un receptor a través de un canal de comunicaciones (en términos informáticos, ese canal suele ser una red de computadoras). Esta ciencia está dividida en dos grandes ramas:

- la **criptografía**, ocupada del cifrado de mensajes en clave y del diseño de **Criptosistemas**, y
- el **criptoanálisis**, que trata de descifrar los mensajes en clave, rompiendo así el criptosistema

PROTOCOLOS SEGUROS

➤ Conceptos:

- Criptosistema, formado por:
 - un alfabeto
 - un espacio de claves
 - un conjunto de transformaciones de cifrado
 - un conjunto de transformaciones de descifrado



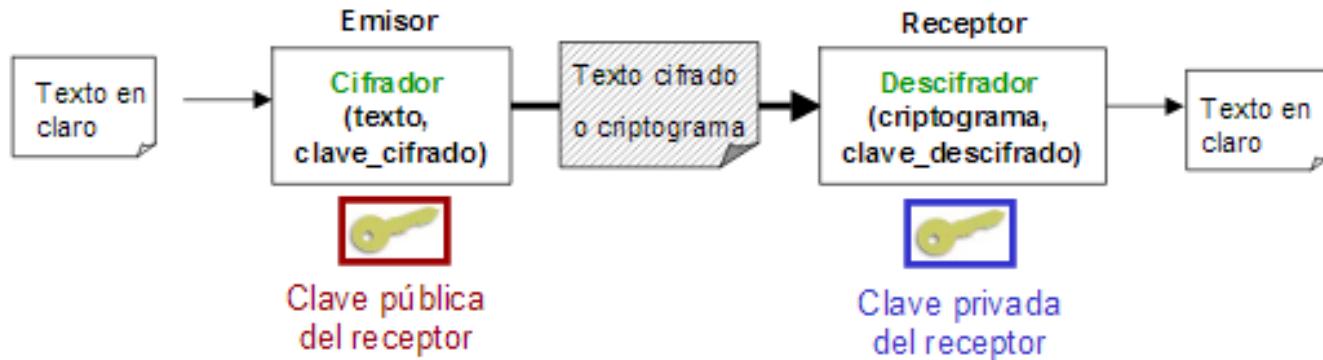
➤ Tipos de criptosistema:

- de clave privada o secreta (simétricos): DES (Data Encryption Standard)
- de clave pública (asimétricos): RSA (Rivest-Shamir-Adleman)

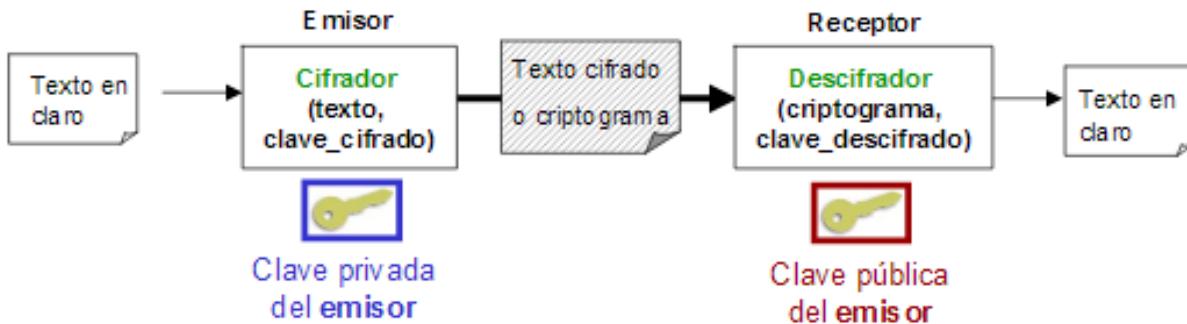
PROTOCOLOS SEGUROS

➤ Conceptos:

- Cifrado/Descifrado asimétrico con clave pública:



- Firma digital y autenticación





➤ Conceptos:

- **Esteganografía (del griego stegos (cubierta)):** ciencia que estudia los procedimientos encaminados a ocultar la existencia de un mensaje en lugar de ocultar su contenido:
- El **objetivo de la criptografía** es que un atacante que consigue un mensaje no sea capaz de averiguar su contenido y el **objetivo de la esteganografía** es ocultar ese mensaje dentro de otro sin información importante, de forma que el atacante ni siquiera se entere de la existencia de dicha información oculta
- No sustituye al cifrado convencional sino que lo complementa, pues ocultar un mensaje reduce las posibilidades de que sea descubierto. Sin embargo, en caso de ser descubierto, el que ese mensaje haya sido cifrado introduce un nivel adicional de seguridad

➤ **Relación entre criptografía y redes:**

- La criptografía es el mecanismo más utilizado para proporcionar seguridad en redes
- Permite crear conexiones seguras sobre canales inseguros
- La Criptografía podrá entonces ser empleada en diferentes niveles de abstracción (protocolos de distintos niveles)
- Según el tipo de red, puede ser más o menos necesaria:
 - Redes internas (LAN): la red es propietaria de la empresa → control total sobre su seguridad
 - Redes externas: no se controlan las infraestructuras públicas → no controlamos la seguridad → Criptografía
 - Intranet (redes externas que se comportan de cara a los usuarios como
 - redes privadas internas) → no controlamos la seguridad → Criptografía

PROTOCOLOS SEGUROS

➤ Métodos básicos de cifrado de redes:

Cifrado de enlace:

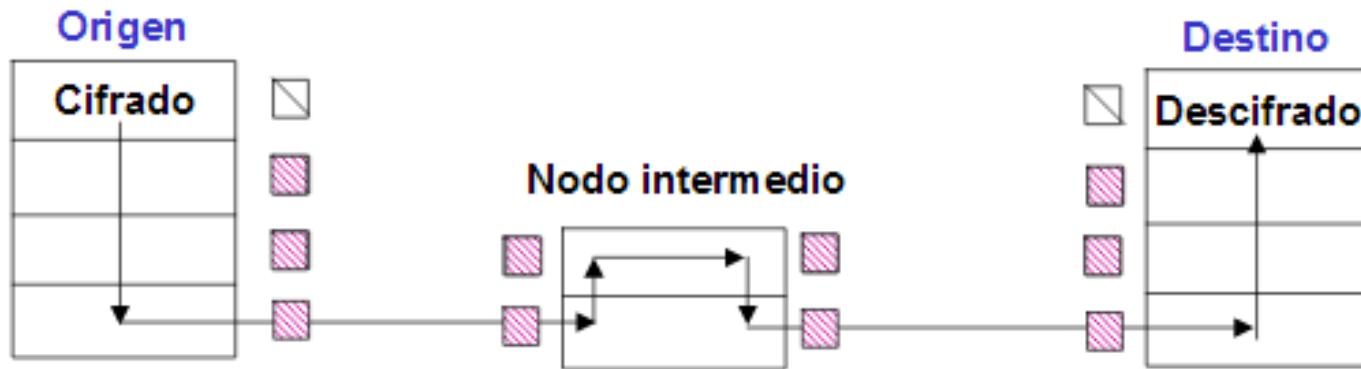
- De capa 2 de OSI
- Cifra todo el mensaje, incluidas las cabeceras de niveles superiores
- Requiere nodos intermedios con capacidades de cifrado/descifrado
- La información está protegida entre cada par de nodos consecutivos (distintas claves para cada par)
- Es necesario descifrarla, aunque sea parcialmente, para procesos de encaminamiento, control de errores...



➤ **Métodos básicos de cifrado:**

Cifrado extremo a extremo:

- De capa 7 de OSI
- Sólo se cifran los datos, las cabeceras se añaden y se transmiten sin cifrar
- El cifrado de datos se mantiene desde origen hasta destino



PROTOCOLOS SEGUROS

➤ Métodos básicos de cifrado

Cifrado de enlace	Cifrado extremo a extremo
Seguridad en los nodos	
El mensaje queda expuesto en el nodo emisor y receptor, y en todos los nodos intermedios	El mensaje sólo queda expuesto en el nodo emisor y en el receptor
Papel del usuario	
El cifrado se aplica en el nodo emisor y en todos los intermedios	El cifrado es aplicado sólo por el proceso emisor
El cifrado es transparente al usuario	El usuario aplica el cifrado (puede decidir qué partes de la información quiere cifrar)
El nodo se encarga del cifrado	El usuario debe encontrar la aplicación de cifrado y preocuparse de usarla
Un solo servicio para todos los usuarios	Cada usuario selecciona su criptosistema
Suele realizarse por hardware	Suele realizarse por software
Se cifran todos los mensajes o ninguno	El usuario elige qué mensajes quiere cifrar y qué claves usar en cada caso
Aspectos de implementación	
Se necesita una clave para par de nodos. Si se compromete uno de los nodos, sólo se comprometen las claves relacionadas con sus adyacentes, y no toda la red	Si se usa cifrado simétrico se necesita una clave para cada par de usuarios, si se usa cifrado asimétrico se necesitan dos claves para cada par de usuarios
Proporciona autenticación de nodos (protege información de cabeceras potencialmente útiles para ataques)	Proporciona autenticación de usuarios (las cabeceras se transmiten sin cifrar, luego se pueden interceptar fácilmente)
En resumen	
Más rápido y más fácil para el usuario	Más flexible y no requiere que los nodos intermedios tengan capacidad de cifrado/descifrado



➤ **Protocolos seguros:**

➤ **Servicios de seguridad:**

- Manejo de claves (negociación y almacenamiento de claves)
- Confidencialidad / Privacidad
- No repudio
- Integridad
- Autenticación
- Autorización

PROTOCOLOS SEGUROS

➤ Protocolos seguros . Niveles

Seguridad de Nivel de	Ventajas	Desventajas	Ejemplos de protocolos
Aplicación	<ul style="list-style-type: none"> - Se puede extender la aplicación para brindar servicios de seguridad sin tener que depender del SO - Facilita el servicio de no repudio 	<ul style="list-style-type: none"> - Los mecanismos de seguridad deben ser diseñados de forma independiente para cada aplicación - Mayores probabilidades de cometer errores 	<ul style="list-style-type: none"> Kerberos PGP SSH SMIME SET IPSec (ISAKMP) RADIUS TACACS
Transporte	<ul style="list-style-type: none"> - En teoría, no se requieren modificaciones por aplicación 	<ul style="list-style-type: none"> - Mantener el contexto del usuario es complicado - TLS requiere que las aplicaciones sean modificadas 	<ul style="list-style-type: none"> SSL (Netscape Corp.) TLS (IETF)
Red	<ul style="list-style-type: none"> - Disminuye el flujo excesivo de negociación de claves - Las aplicaciones no requieren modificación alguna - Permite crear VPNs e intranets 	<ul style="list-style-type: none"> - Difícil manejar el no repudio 	<ul style="list-style-type: none"> IPSec (AH, ESP)(IETF) NLSP (ISO) Protocolos de tunneling: PPTP, L2TP
Enlace de datos	<ul style="list-style-type: none"> - Más rápido 	<ul style="list-style-type: none"> - No son soluciones estables y funcionan bien sólo para enlaces dedicados - Los dispositivos deben estar físicamente conectados 	<ul style="list-style-type: none"> ATMs (IEEE) SILS (IEEE) CHAP PAP MS-CHAP, EAP, LEAP, PEAP



Protocolos seguros - Nivel de Aplicación

PROTOCOLOS SEGUROS

➤ **SSH (Secure SHell):**

- SSH es un protocolo de nivel de aplicación para crear conexiones seguras entre dos sistemas sobre redes no seguras (SSH2)
- Alternativa a programas de acceso remoto no seguros, como telnet, ftp, rlogin, rsh y rcp (slogin, ssh y scp)
- Proporciona terminal de sesión cifrada con autenticación fuerte del servidor y el cliente, usando criptografía de clave pública
- Incluye características como:
 - una variedad de mecanismos de autenticación de usuarios
 - conexiones TCP arbitrarias de tunneling a través de la sesión SSH, protegiendo protocolos inseguros como IMAP y permitiendo el pasoseguro a través de cortafuegos
 - reenvío automático de conexiones X windows
 - soporte para métodos de autenticación externa, incluyendo Kerberos
 - transferencias seguras de ficheros
- SSH está basado en protocolos documentados por el IETF

➤ **SSH (Secure SHell):**

Otros tipos de protección que proporciona SSH:

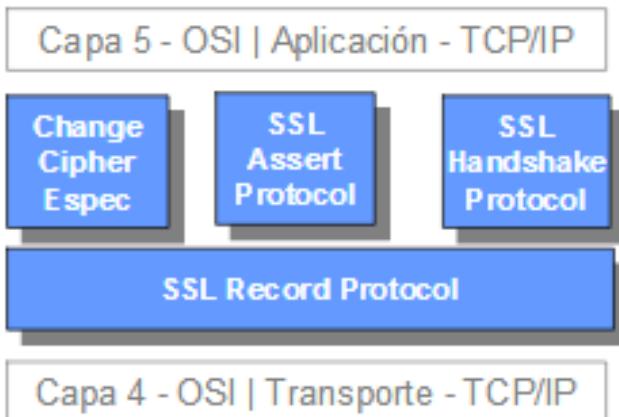
- Después de la conexión inicial, el cliente puede verificar que se está conectando al mismo servidor durante sesiones posteriores
- El cliente puede transmitir su información de autentificación al servidor, como el nombre de usuario y la contraseña, en formato cifrado
- El cliente tiene la posibilidad de usar X11 en aplicaciones lanzadas desde el indicador de comandos de la shell. Esta técnica proporciona una interfaz gráfica segura (llamada reenvío por X11)
- Si el servidor usa la técnica del reenvío de puerto, los protocolos considerados como inseguros (POP, IMAP...), se pueden cifrar para garantizar una comunicación segura



➤ Secuencia de eventos de una conexión SSH:

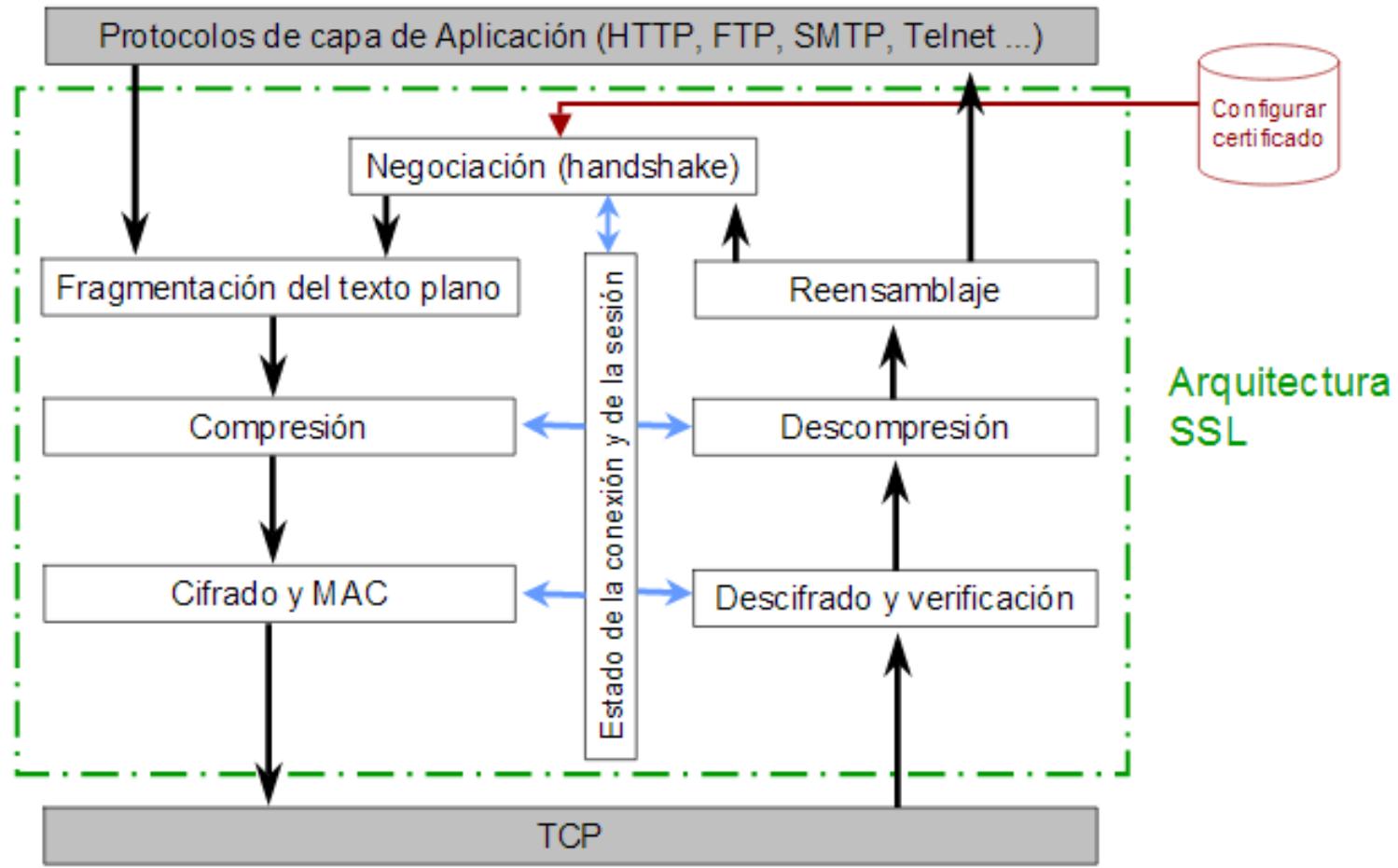
1. Se crea una capa de transporte segura para que el cliente sepa que está efectivamente comunicando con el servidor correcto. Luego se cifra la comunicación entre el cliente y el servidor por medio de un código simétrico
2. Con la conexión segura al servidor en su lugar, el cliente se autentifica ante el servidor sin preocuparse de que la información de autenticación pudiese exponerse a peligro. OpenSSH usa claves DSA o RSA y la versión 2.0 del protocolo SSH para autenticaciones predeterminadas
3. Con el cliente autenticado ante el servidor, se pueden usar varios servicios diferentes con seguridad a través de la conexión, como una sesión shell interactiva, aplicaciones X11 y túneles TCP/IP

PROTOCOLOS SEGUROS

- **Protocolos seguros - Nivel de Transporte → SSL (Secure Socket Layer)**
- El protocolo SSL fue desarrollado por Netscape en 1994 y puesto en dominio público para la definición de canales seguros sobre TCP. Su objetivo es la realización de conexiones seguras a los servidores independientemente del SO de los extremos del canal.
 - Está compuesto por dos capas:
 - La primera capa (**SSL Record Protocol**), encapsula los protocolos de nivel más alto y construye el canal de comunicaciones seguro
 - La segunda capa está formada por tres protocolos:
 - **SSL Handshake protocol** se encarga de gestionar la negociación de los algoritmos de cifrado, y la autenticación entre el cliente y el servidor
 - **SSL Alert Protocol** señaliza errores y problemas en la sesión establecida
 - **Change Cipher Espec Protocol** consiste en un solo mensaje de 1 byte que sirve para notificar cambios en la estrategia de cifrado
- 
- Capa 5 - OSI | Aplicación - TCP/IP
- Capa 4 - OSI | Transporte - TCP/IP
- Diagrama jerárquico de los protocolos SSL:
- SSL Record Protocol (en la Capa 5 - OSI | Aplicación - TCP/IP)
 - SSL Record Protocol encapsula los siguientes protocolos (en la Capa 4 - OSI | Transporte - TCP/IP):
 - Change Cipher Espec
 - SSL Alert Protocol
 - SSL Handshake Protocol

PROTOCOLOS SEGUROS

➤ SSL. Arquitectura



PROTOCOLOS SEGUROS

➤ **SSL. Funcionamiento**

Su funcionamiento es el siguiente:

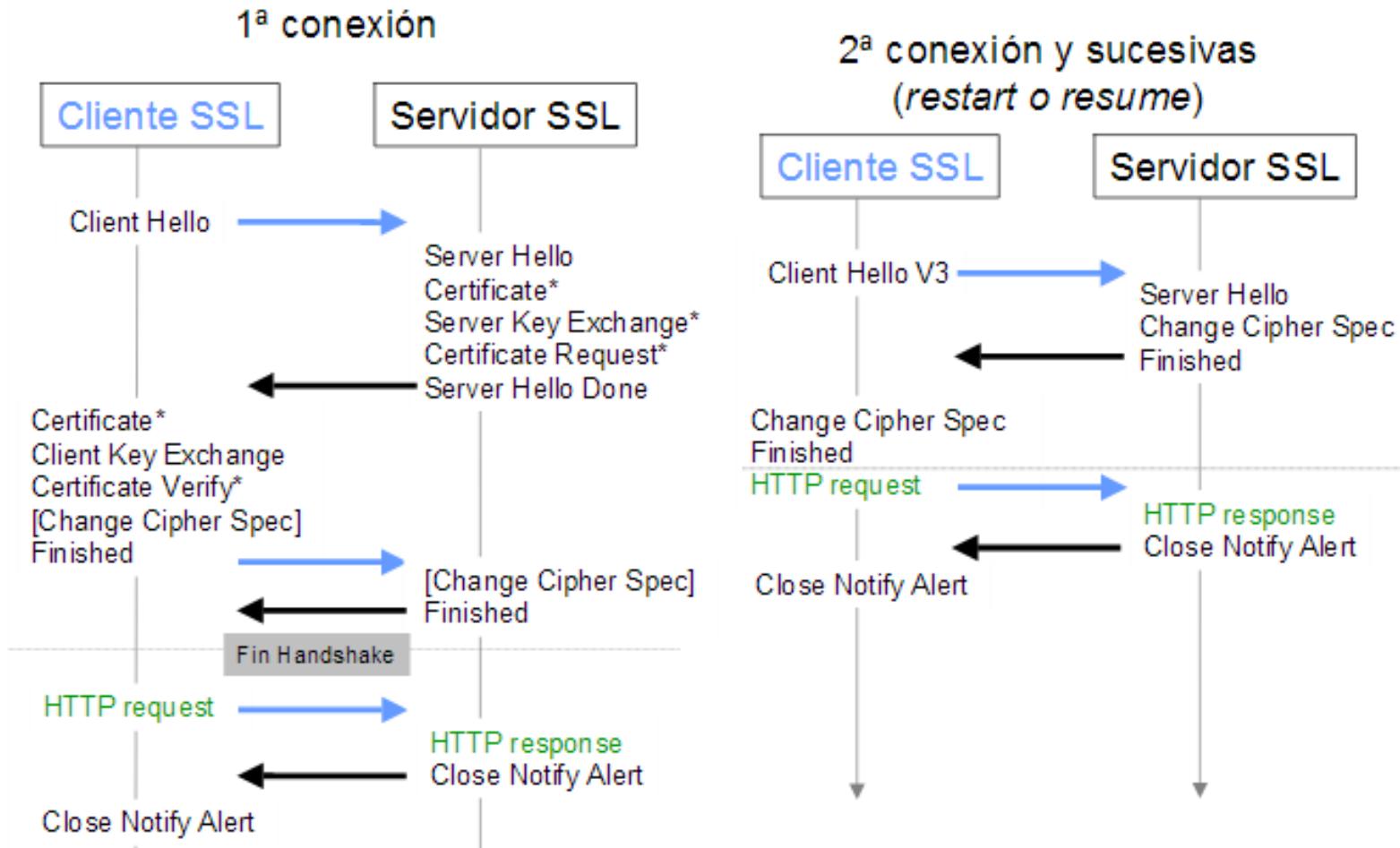
- El cliente al hacer la conexión informa sobre los sistemas criptográficos que tiene disponibles, y el servidor responde con un identificador de la conexión, su clave certificada e información sobre los sistemas criptográficos que soporta
- El cliente deberá elegir un sistema criptográfico, verificará la clave pública del servidor. Entonces se genera una clave cifrada con la clave del servidor
- Este es uno de los puntos importantes del protocolo SSL, porque si alguien pudiese descifrar la información, sólo conseguiría romper esa conexión, y una conexión posterior requeriría una clave criptográfica diferente
- Una vez finalizado este proceso, los protocolos toman el control de nivel de aplicación, de modo que SSL nos asegura que:
 - los mensajes que enviamos o recibimos no han sido modificados
 - ninguna persona sin autorización puede leer la información transmitida
 - efectivamente recibe la información quien debe recibirla



➤ SSL Handshake Protocol

- • Genera los parámetros criptográficos del estado de la sesión
- • Opera sobre el SSL Record Layer Protocol
- • Tiene dos mecanismos de negociación de sesión:
 - Full Handshake (1^a conexión)
 - Abbreviated Handshake (conexiones posteriores)

➤ SSL Handshake Protocol



PROTOCOLOS SEGUROS

➤ **SSL. Protocolos de capa superior**

- Versión actual SSL 3.0
- SSL es capaz de trabajar de forma transparente con todos los protocolos que trabajan sobre TCP
- Para ello el IANA tiene asignado un número de puerto por defecto a cada uno de ellos:

Identificador de protocolo	Puerto TCP	Descripción
https	443	HTTP sobre SSL
smt�	465	SMTP sobre SSL
ntt�	563	NTTP sobre SSL
ladps	646	LDAP sobre SSL
telnets	992	TELNET sobre SSL
imaps	993	IMAP sobre SSL
ircs	994	IRC sobre SSL
pop3s	995	POP3 sobre SSL
ftps-data	989	FTP-Datos sobre SSL
ftps-control	990	FTP-Control sobre SSL

SSL. Problemas

- Sólo trabaja sobre TCP (no UDP ni IPX)
 - crear sesión SSL sobre TCP y cifrar los paquetes UDP con el fruto de esa negociación. Requiere que cada paquete UDP pueda descifrarse por separado y se cifre con claves distintas
- No repudio de transacciones
 - SSL lo implementa si ambos extremos tienen certificados
 - Usar S/MIME sobre SSL
- Ineficiencia debido al handshake inicial
 - Cachear las sesiones (válido para HTTP, pero no para otros)
 - Uso de hardware especializado que acelere el tráfico SSL
 - Tarjeta aceleradora integrada en cada servidor SSL
 - Dispositivo externo y autónomo dedicado exclusivamente al cifrado y descifrado SSL (compartido por todos los servidores SSL)
 - Dispositivo que integra el balanceo de carga con el cifrado SSL



Protocolos seguros - Nivel de Red



IPSec (IP Security):

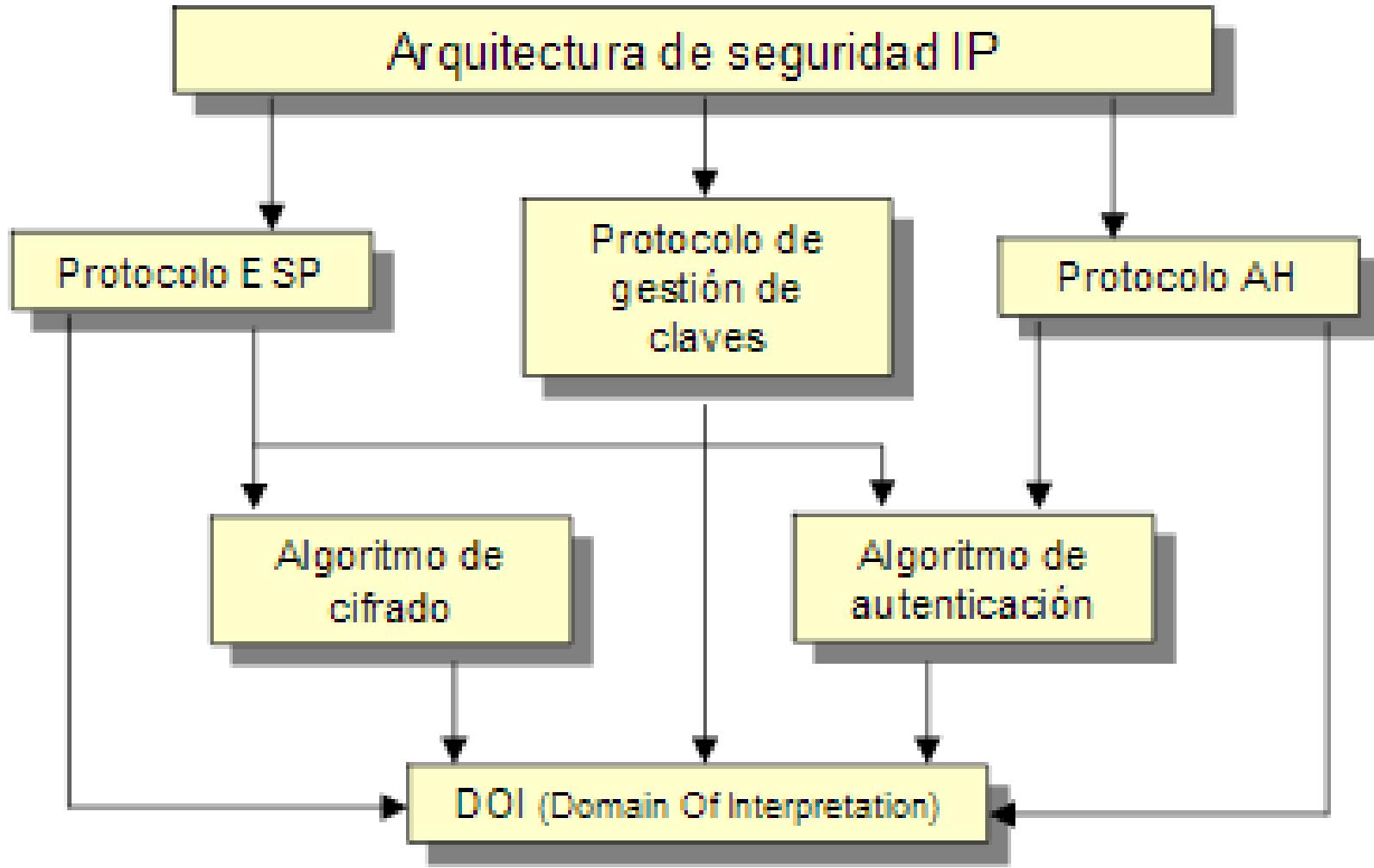
- Definido en la RFC 2401, estándar IETF desde 1999
- Suministra seguridad a nivel de red, proporcionando seguridad para IP y los protocolos de capas superiores
- Provee:
 - Control de accesos
 - Integridad no orientada a la conexión
 - Autenticación del origen de datos
 - Rechazo o reenvío de paquetes
 - Confidencialidad
 - Negociación de compresión IP
- Independiente de los algoritmos criptográficos actuales
- Contempla su implementación con IPv4 e IPv6
- Es un componente obligado en IPv6



➤ IPSec. Arquitectura

- Componentes fundamentales de esta arquitectura:
 - Protocolos de seguridad:
 - AH (Authentication Header), RFC 2402
 - ESP (Encapsulation Security Payload), RFC 2406
 - Asociaciones de seguridad: SA (Security Association)
 - IKE (Internet Key Exchange) RFC 2409
 - Algoritmos de autenticación y cifrado

PROTOCOLOS SEGUROS





➤ IPSec. Modos de funcionamiento

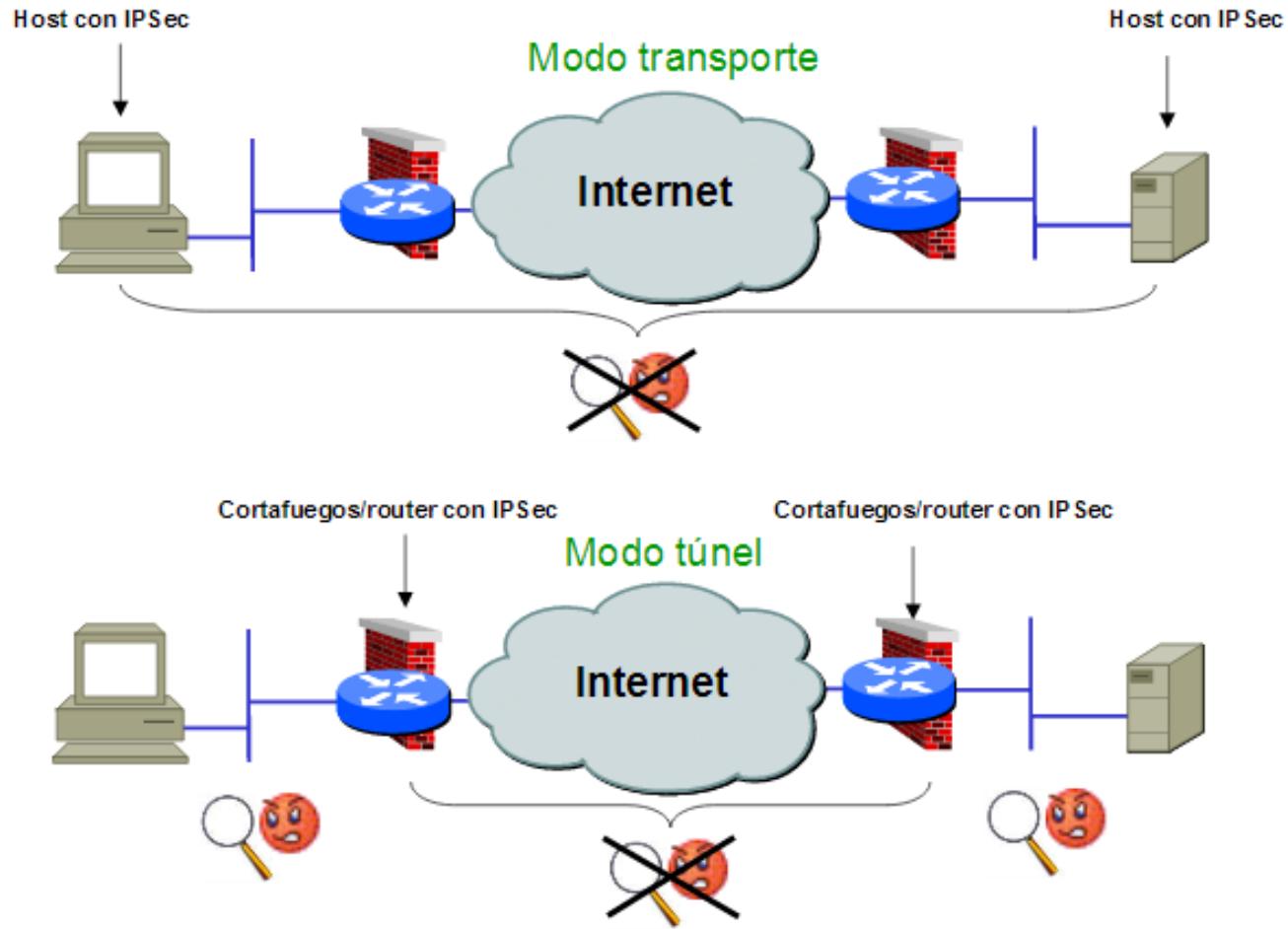
➤ Modo transporte (IP seguro):

- se protege la carga útil IP (payload) (capa de transporte)
- comunicación segura extremo a extremo
- requiere implementación de IPSec en ambos hosts

➤ Modo túnel (IP seguro dentro de IP estándar):

- se protegen paquetes IP (capa de red)
- para la comunicación segura entre routers/gateways de seguridad sólo se puede usar este modo
- permite incorporar IPSec sin afectar a los hosts
- se integra cómodamente con VPNs

➤ SSL. Modos de funcionamiento





PROTOCOLOS SEGUROS

➤ IPSec vs SSL/TLS

	SSL/TLS	IPSec
Control de accesos		
Conexiones permanentes		✓
Conexiones efímeras o puestos móviles	✓	
Ambos tipos de acceso	✓	✓
Usuarios		
Todos los usuarios son empleados de la compañía		✓
No todos los usuarios son empleados de la compañía	✓	
No todos los usuarios son empleados de la compañía y, además, algunos trabajan con sus propios sistemas	✓	✓
Software cliente		
Todos los usuarios han de tener acceso a todos los recursos de la red		✓
Deseamos controlar el acceso a determinadas aplicaciones	✓	
Necesitamos niveles variables de control de acceso en las diferentes aplicaciones	✓	✓

Confidencialidad y Autenticidad		
Precisamos de un alto nivel de seguridad en el cifrado y autenticación		✓
La confidencialidad y autenticidad no son especialmente críticas en nuestros sistemas	✓	
Precisamos de niveles moderados de confidencialidad e integridad		
Criticidad de los recursos accedidos		
Alta		✓
Moderada	✓	
Variable	✓	✓
Criticidad de las funciones realizadas		
Alta		✓
Moderada	✓	
Variable	✓	✓
Nivel técnico de los usuarios		
Entre moderado y alto		✓
Entre moderado y bajo	✓	
Implantación, flexibilidad y escalabilidad		
Deseamos una implantación rápida y facilidad de mantenimiento	✓	
Deseamos flexibilidad en las modificaciones futuras		✓
Ambas consideraciones son importantes	✓	✓



VPNs (Virtual Private Networks)



➤ VPN

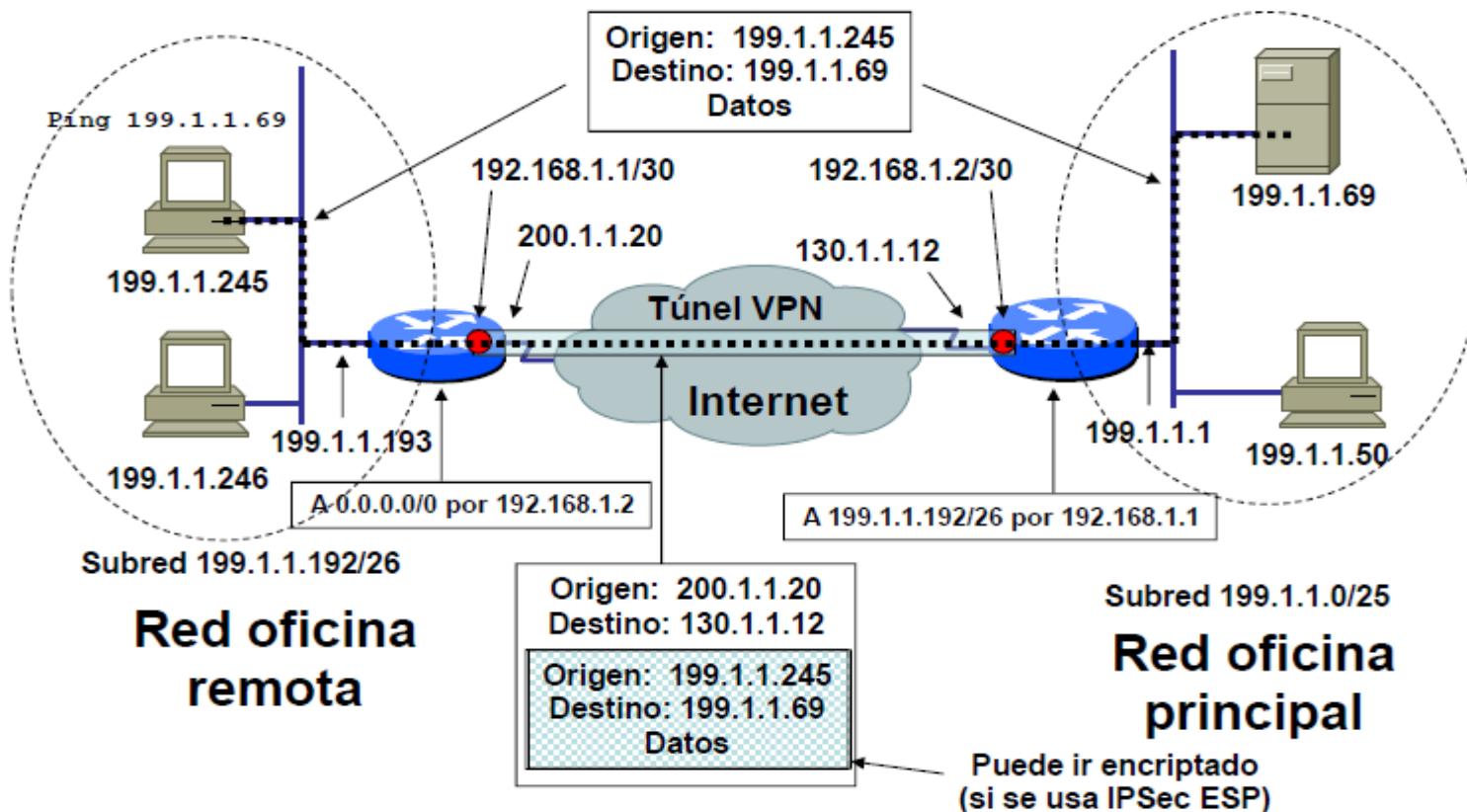
- ¿Qué es tunneling?
- Protocolos de tunneling
- Comparativa
- ¿Qué es una VPN?
- Clasificación general de las VPNs
- Elementos de una VPN
- Implementación de una VPN
- Funcionamiento básico
- ¿Para qué sirven las VPNs?
- Ventajas e inconvenientes de usar VPNs
- Evolución de los estándares de VPNs
- Alternativas a las VPNs



- **VPNs. ¿Qué es tunneling?**
- La transmisión de paquetes de datos de un determinado protocolo encapsulados en otro, de manera que el contenido del paquete original puede llegar inalterado a su destino, creando algo así como una conexión punto a punto virtual a través de una red
- También permiten crear redes privadas virtuales o VPNs (Virtual Private Networks)

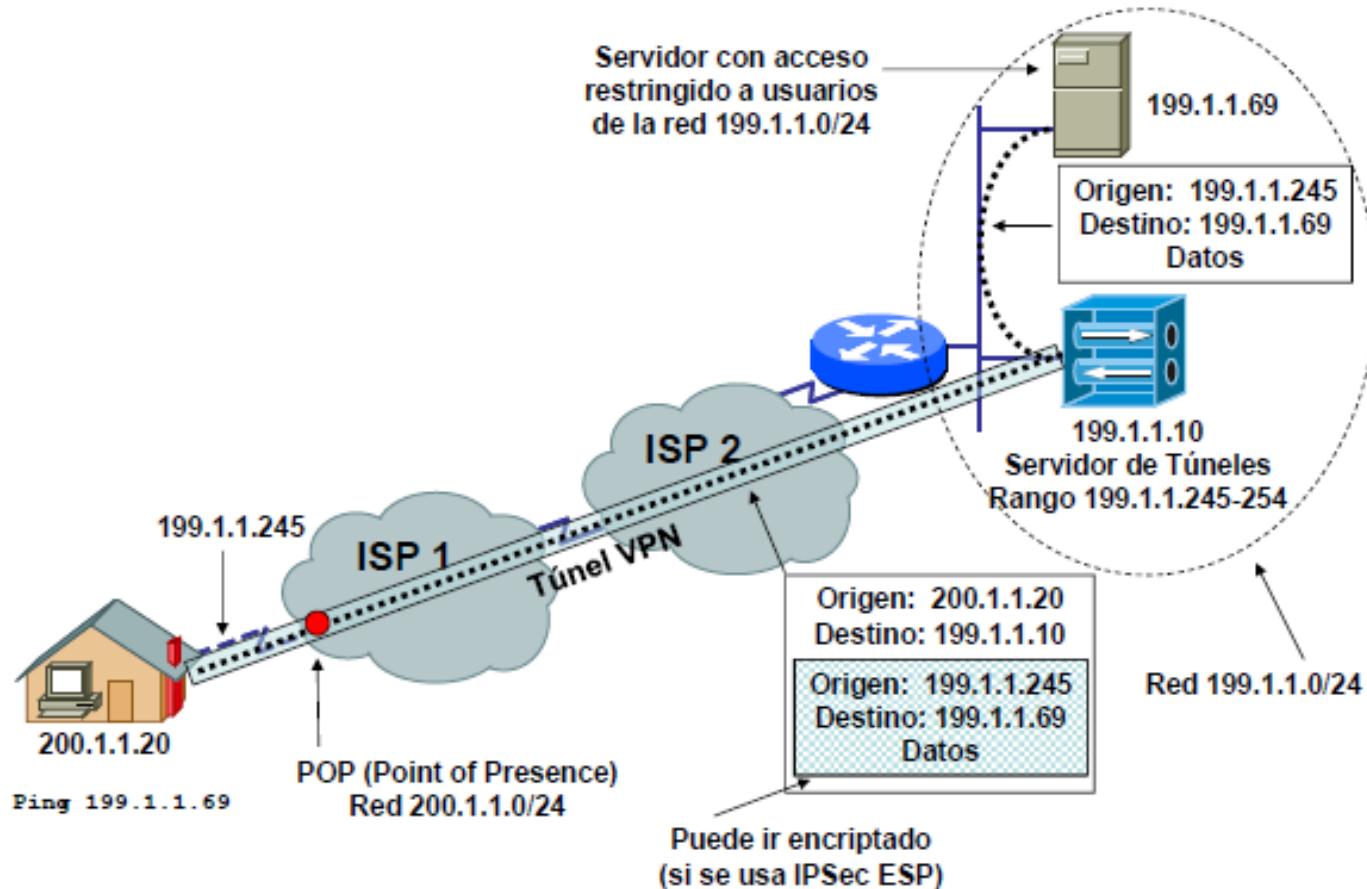
PROTOCOLOS SEGUROS

➤ VPNs. Ejemplo



PROTOCOLOS SEGUROS

➤ VPNs. Ejemplo



PROTOCOLOS SEGUROS

- Hay múltiples protocolos de tunneling:
 - **IPSec:** IP Secure (RFC 2401) [capa 3]
 - **L2F:** Layer 2 Forwarding Protocol (Cisco) (RFC 2341) [capa 2]
 - **PPTP:** Point-to-Point Tunneling Protocol (RFC 2637) [capa 2]
 - **L2TP:** Layer 2 Tunneling Protocol (RFC 2661) [capa 2 y 3]
 - **GRE:** Generic Routing Encapsulation (RFC 1701) [capa 3]
 - **IP/IP:** IP over IP (RFC 2003) [capa 3]
 - **IPSec:** IP Secure (RFC 2475) [capa 3]
 - **MPLS:** Multi-Protocol Label Switching (RFC 2917) [capas 2 y 3]
 - **MPOA:** Multi-Protocol Over ATM [capa 3]
- **PPTP:** orientado al usuario → permite establecer un túnel de forma transparente al proveedor de Internet
- **L2TP:** orientado al proveedor → permite establecer un túnel de forma transparente al usuario (generalmente se utiliza junto con IPSec)



➤ **PPTP (Point-to-Point Tunneling Protocol):**

- Protocolo desarrollado por Microsoft y normalizado por la IETF (RFC 2637)
- Permite el tráfico seguro de datos desde un cliente remoto a un servidor corporativo privado
- PPTP soporta múltiples protocolos de red (IP, IPX, NetBEUI...)
- Tiene una mala reputación en seguridad
- Muy usado en entornos Microsoft

➤ **L2F (Layer 2 Forwarding):**

- Protocolo desarrollado por Cisco Systems
- Precursor del L2TP
- Ofrece métodos de autenticación de usuarios remotos
- Carece de cifrado de datos



➤ **L2TP (Layer 2 Tunneling Protocol):**

- Estándar aprobado por la IETF (RFC 2661)
- Mejora combinada de PPTP y L2F
- No posee cifrado o autenticación por paquete, por lo que ha decombinarse con otro protocolo, como IPSec
- Combinado con IPSec ofrece la integridad de datos y confidencialidad exigidos para una solución VPN
- Permite el encapsulado de distintos protocolos (IP, IPX, NetBEUI...)

➤ VPNs. ¿Qué es una VPN?

- Proporciona el medio para usar una infraestructura de red pública como un canal apropiado para comunicaciones privadas de datos
- Con las tecnologías de cifrado y encapsulación adecuadas, una VPN constituye un túnel (generalmente túnel IP) cifrado y/o encapsulado a través de Internet
- Utiliza protocolos de tunneling
- Proporciona los servicios de las redes privadas (confianza)
- Utiliza conexiones temporales (virtuales)
- Es una combinación de hardware y/o software que:
 - Extiende una intranet o red corporativa a través de la insegura y pública Internet
 - Permite comunicación segura con las oficinas sucursales, usuarios móviles o remotos y clientes

➤ **VPNs. Clasificación general de VPNs**

➤ **VPNs de Intranets:**

- Proporcionan conectividad interna entre distintos emplazamientos de una misma empresa

➤ **VPNs de Acceso Remoto**

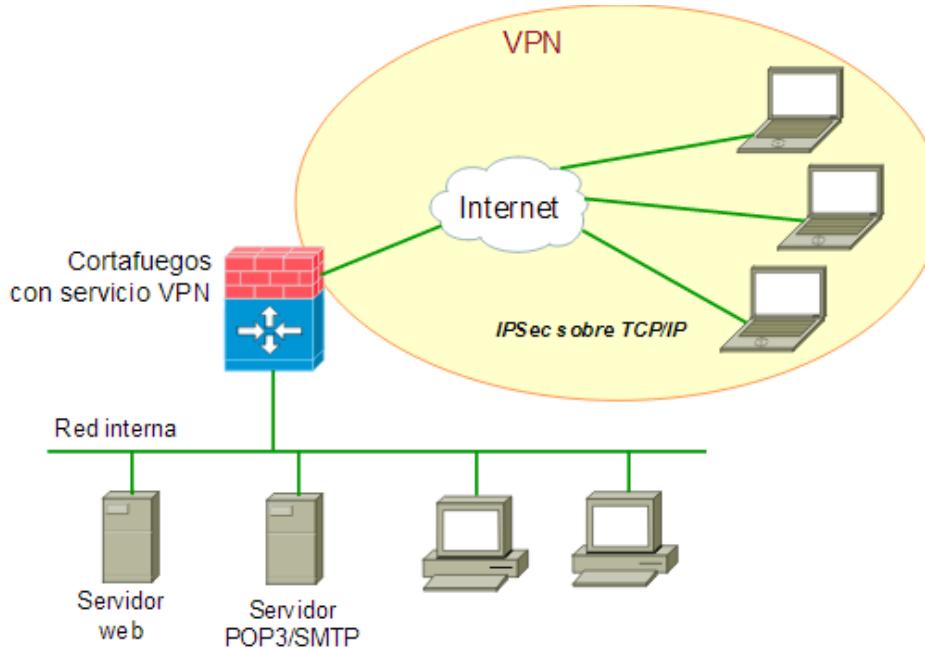
- Amplían la red interna a los tele-trabajadores, trabajadores itinerantes y a las oficinas remotas

➤ **VPNs de Extranets**

- Amplían la red de las empresas e incluyen proveedores, empresas asociadas y/o clientes

➤ VPNs. Elementos de una VPN

- Las VPNs se basan en las siguientes tecnologías
 - Firewalls: Como mecanismo de protección adicional
 - Autenticación: Para dar acceso sólo a sistemas permitidos
 - Cifrado: Para asegurar confidencialidad e integridad
 - Tunneling: Como mecanismo de intercambio de información





➤ VPNs. Implementación de una VPN

Hay que realizar las siguientes operaciones:

- Diseñar una topología de red y firewalls:
 - Teniendo en cuenta los costes y la protección
- Escoger un protocolo para los túneles
 - Teniendo en cuenta los equipos finales
 - Teniendo en cuenta las aplicaciones finales
- Diseñar una PKI (Public Key Infrastructure)
 - Teniendo en cuenta las necesidades del protocolo



➤ VPNs. Funcionamiento básico de una VPN

1. El usuario remoto marca a su ISP local y se conecta a la red del ISP de forma normal
2. Cuando desea conectarse a la red corporativa, el usuario inicia el túnel mandando una petición a un servidor VPN de la red corporativa
3. El servidor VPN autentica al usuario y crea el otro extremo del túnel
4. El usuario comienza a enviar datos a través del túnel, que son cifrados por el software VPN (del cliente) antes de ser enviados sobre la conexión del ISP
5. En el destino, el servidor VPN recibe los datos y los descifra, propagando los datos hacia la red corporativa. Cualquier información enviada de vuelta al usuario remoto también es cifrada antes de enviarse por Internet

- **VPNs. ¿Para qué sirven las VPNs?**
- Permiten conectar diferentes delegaciones de una empresa, simulando una red local de una manera transparente y económica
- Proporcionan acceso a los diferentes recursos de la red de forma remota a todos los usuarios de la red corporativa (clientes, socios, consultores...)
- Las VPNs seguras proporcionan:
 - Confidencialidad (cifrado de los datos)
 - Integridad (IPSec, asegura que los datos no son modificados en tránsito)
 - Autenticación de usuarios (certificados X.509; identificación de usuarios y protección contra ataques de suplantación)
 - Control de acceso a la red (políticas VPN)
 - No repudio



- VPNs. Ventajas
- Ahorro en costes
- No se compromete la seguridad de la red empresarial
- El cliente remoto adquiere la condición de miembro de la LAN (permisos, directivas de seguridad)
- El cliente tiene acceso a todos los recursos ofrecidos en la LAN (impresoras, correo electrónico, base de datos, ...)
- Acceso desde cualquier punto del mundo (siempre y cuando se tenga acceso a Internet)



➤ VPNs. Problemas

- No se garantiza disponibilidad (NO Internet → NO VPN)
- No se garantiza el caudal (red pública)
- Gestión de claves de acceso y autenticación delicada y laboriosa
- La fiabilidad es menor que en una línea dedicada
- Mayor carga en el cliente VPN (encapsulación y cifrado)
- Mayor complejidad en la configuración del cliente (proxy, Servidor de correo, ...)
- Una VPN se considera segura, pero no hay que olvidar que la información sigue viajando por Internet (no seguro y expuestos a ataques)



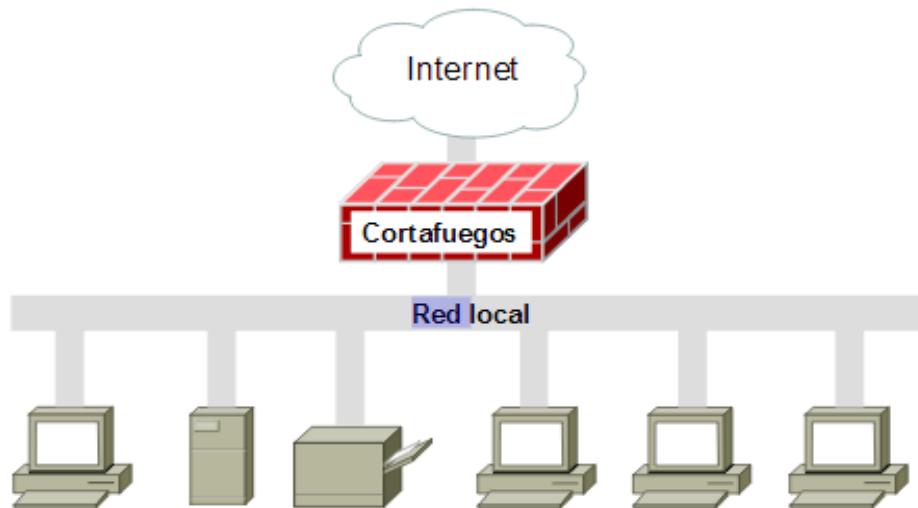
Cortafuegos



➤ **Qué veremos en esta sección:**

- ¿Qué es un cortafuegos?
- ¿Hasta qué nivel nos protegen?
- Funciones de los cortafuegos
- Componentes de los cortafuegos
- Técnicas aplicadas en los cortafuegos
- Arquitecturas de los cortafuegos
- Tipos de cortafuegos
- Servicios adicionales de un cortafuegos

- ¿Qué es un cortafuegos?
- Combinación de técnicas, políticas de seguridad y tecnologías (hardware y software) encaminadas a proporcionar seguridad en la red, controlando el tráfico que circula entre dos o más redes (y más concretamente, entre una red privada e Internet)
- Cortafuegos = firewall o gateway de seguridad



- **¿Hasta qué nivel nos protege un cortafuegos?**
- El nivel de protección que ofrece un cortafuegos depende de las necesidades concretas
- Un cortafuegos proporciona un único punto de acceso donde centralizar las medidas de seguridad y auditoría de la red
- No puede protegernos de:
 - amenazas que no pasan a través del cortafuegos
 - amenazas que provienen de nuestra propia red
 - clientes o servicios que admitimos como válidos pero que resultan vulnerables (tunneling sobre HTTP, SMTP...)
- **Los cortafuegos deben combinarse con otras medidas de seguridad en redes → protocolos seguros**

➤ Funciones de un cortafuegos

- Controlar, permitiendo o denegando, los accesos desde la red local hacia el exterior y los accesos desde el exterior hacia la red local (redes, subredes o nodos específicos y servicios)
- Filtrar los paquetes que circulan, de modo que sólo los servicios permitidos puedan pasar
- Monitorizar el tráfico, supervisando destino, origen y cantidad de información recibida y/o enviada
- Almacenar total o parcialmente los paquetes que circulan a través de él para analizarlos en caso de problemas
- Establecer un punto de cifrado de la información si se pretende comunicar dos redes locales a través de Internet

PROTOCOLOS SEGUROS

➤ **Componentes de un cortafuegos**➤ **Filtros**

- dispositivos que permiten bloquear selectivamente determinados paquetes
- normalmente se trata de routers con capacidad de filtrado o computadoras con utilidades de filtrado

➤ **Nodos bastión(bastion host o gate)**

- son computadoras altamente seguras que sirven como punto de contacto entre la red local e Internet
- se trata de máquinas vulnerables por estar expuestas directamente a Internet
- generalmente máquinas UNIX en las que se han extremado las medidas de seguridad (sólo se instalan los servicios absolutamente imprescindibles)

➤ Técnicas aplicadas a cortafuegos

➤ Filtrado de paquetes

- se controla selectivamente el tráfico de la red definiendo una serie de reglas que especifican qué tipos de paquetes pueden circular en cada sentido y cuáles deben bloquearse
- las reglas para definir qué paquetes se permiten o no se basan en las cabeceras de los paquetes

➤ Servicios delegados(proxy service)

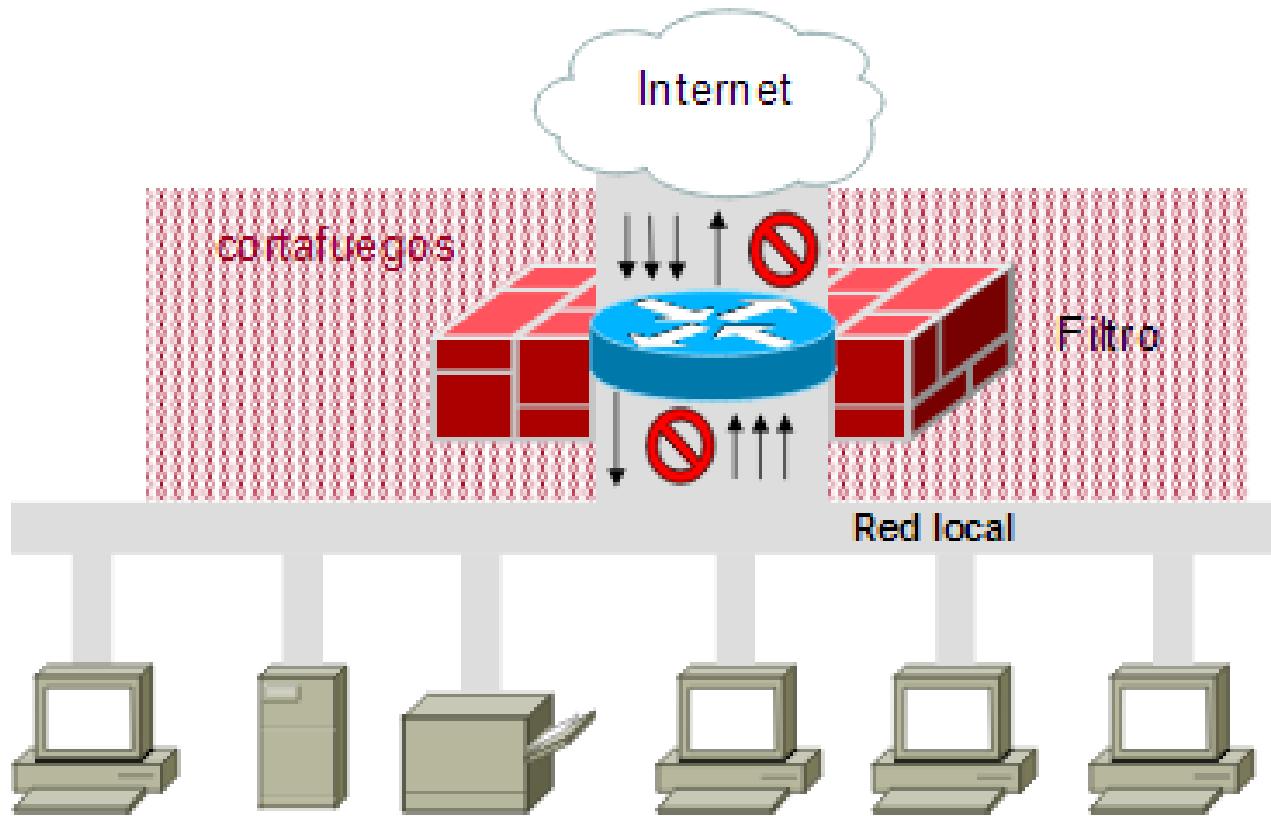
- son aplicaciones especializadas que funcionan en un cortafuegos (normalmente en el nodo bastión) y que hacen de intermediarios entre los servidores y los clientes reales
- estas aplicaciones reciben las peticiones de servicios de los usuarios, las analiza y en su caso modifica, y las transmiten a los servidores reales
- es transparente al usuario y al servidor real

➤ Arquitecturas de cortafuegos

La combinación de los dos componentes básicos, filtro y nodo bastión, y las técnicas de filtrado y delegación, permite definir múltiples arquitecturas para los cortafuegos:

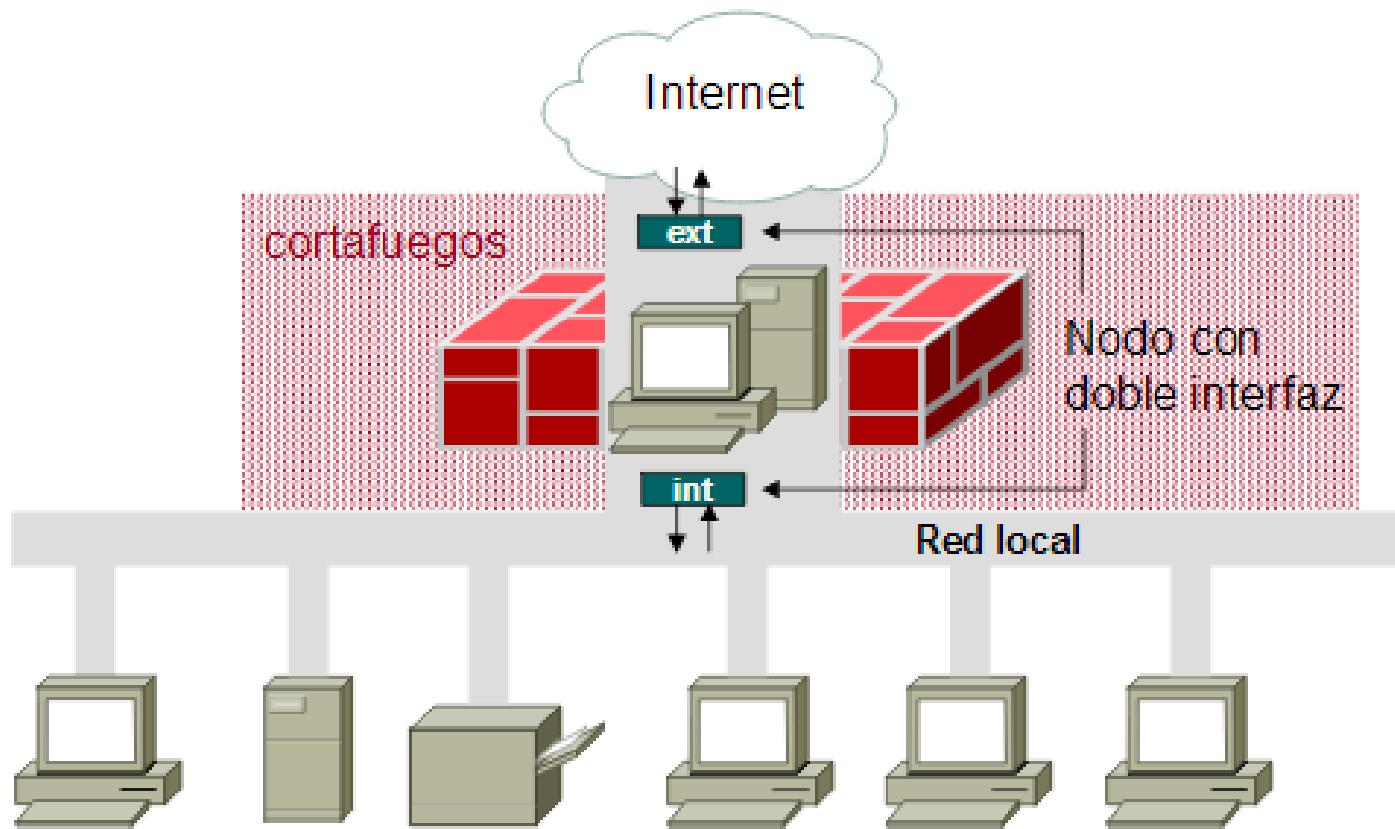
- Cortafuegos de filtrado de paquetes (Screening Router)
- Cortafuegos con nodo de doble interfaz (Dual-Homed host architecture)
- Cortafuegos con nodo pantalla
- Cortafuegos con red pantalla (DMZ, DeMilitarized Zone)

PROTOCOLOS SEGUROS



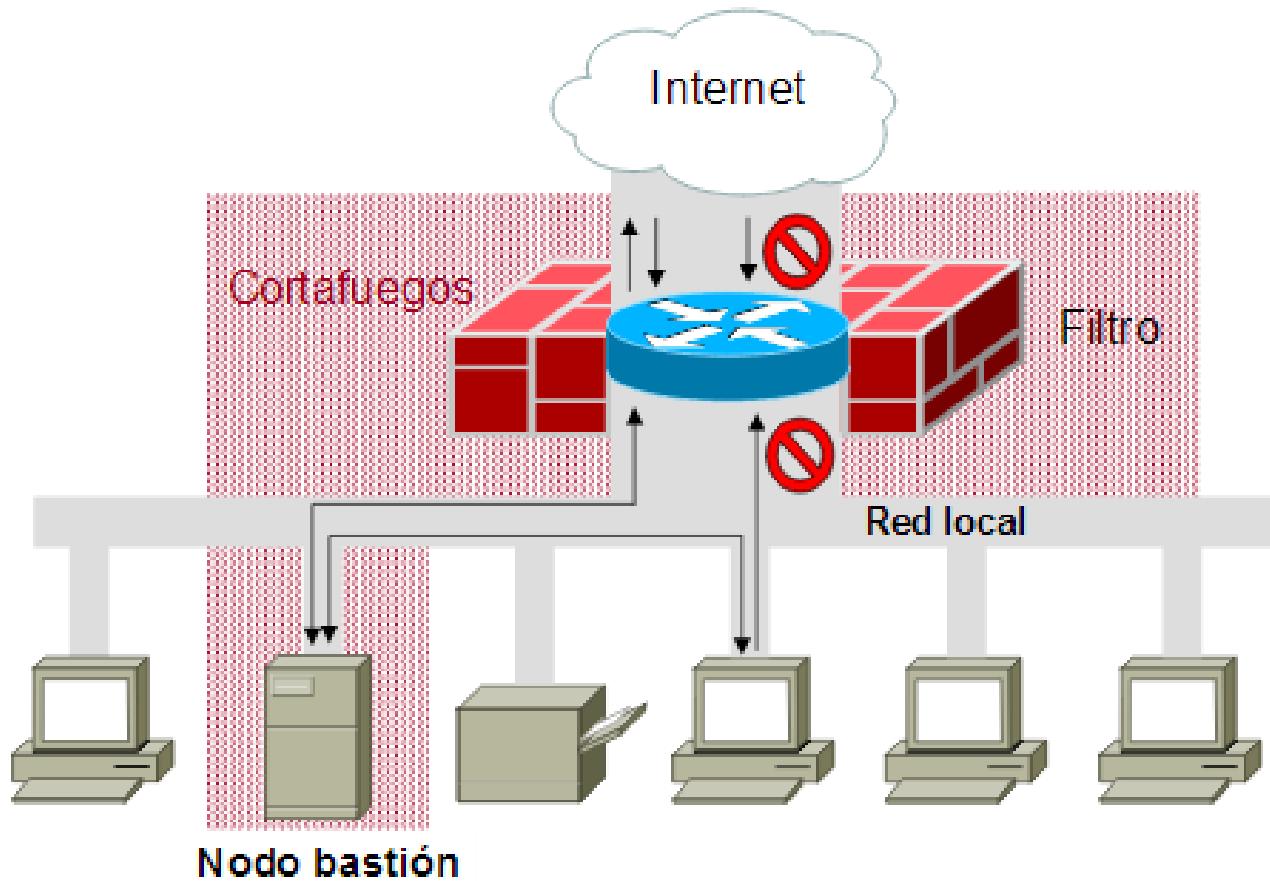
Cortafuegos con filtrado de paquetes
(*Screening Router*)

PROTOCOLOS SEGUROS



Cortafuegos con nodo con doble interfaz
(Dual-homed host arquitectura)

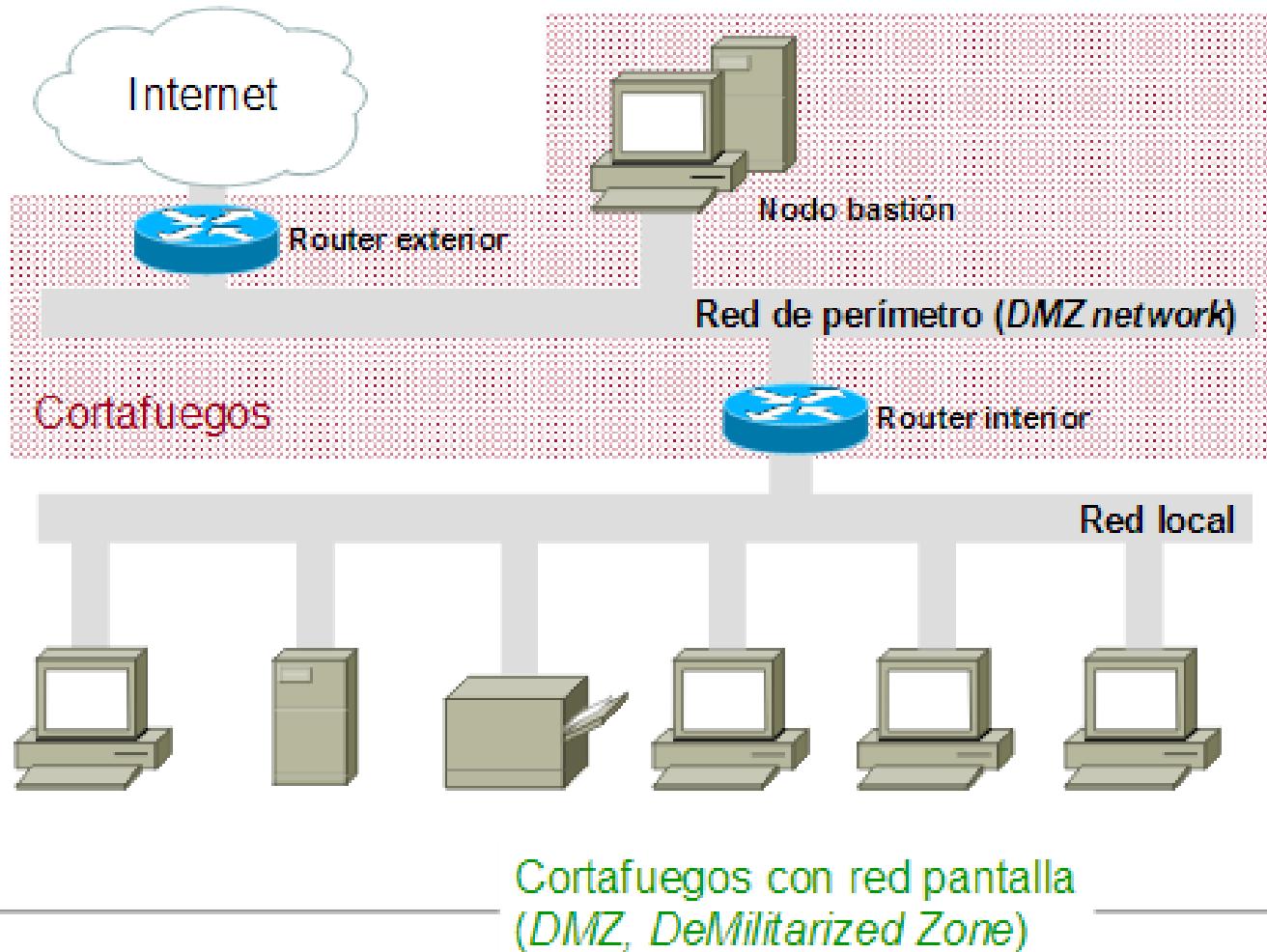
PROTOCOLOS SEGUROS



Nodo bastión

Cortafuegos con nodo pantalla

PROTOCOLOS SEGUROS



PROTOCOLOS SEGUROS

➤ **Cortafuegos de filtrado de paquetes**➤ **Ventajas:**

- Rapidez, transparencia, flexibilidad
- Alto rendimiento y escalabilidad a bajo coste
- Útiles para bloquear ataques DoS

➤ **Inconvenientes:**

- funcionalidad limitada
- complejidad de su configuración (por expertos) → susceptibles a error en la implementación de las reglas
- fácilmente vulnerables mediante técnicas de spoofing
- no previenen contra ataques que exploten vulnerabilidades de aplicaciones
- históricos de accesos imprecisos

➤ **Muy efectivos, como primera barrera, si se combinan con más medidas de seguridad**



Mecanismos de Seguridad

PROTOCOLOS SEGUROS

Recordemos:

La seguridad se deben de garantizar las siguientes características:

- **Confidencialidad:** el contenido de la comunicación sólo puede ser comprendido por la persona/entidad a la que va dirigida o está autorizada para su visualización.
- **Integridad:** garantiza que el documento recibido no ha sufrido ninguna alteración con respecto al original y además, esto puede ser comprobado.
- **Autenticación:** característica que permite comprobar que el contenido recibido ha sido elaborado por la persona que consta como autor. Es decir, podemos aportar alguna prueba de que somos quien decimos ser.
- **Disponibilidad:** los servicios deben de estar accesibles para los usuarios autorizados cuando estos lo requieran.
- **No repudio:** consiste en que nadie que haya realizado una determinada operación pueda negar que realmente lo hizo. En este caso tenemos dos posibilidades:
 - **No repudio en origen:** el emisor no puede negar que realizó el envío.
 - **No repudio en destino:** el receptor no puede negar que recibió la comunicación

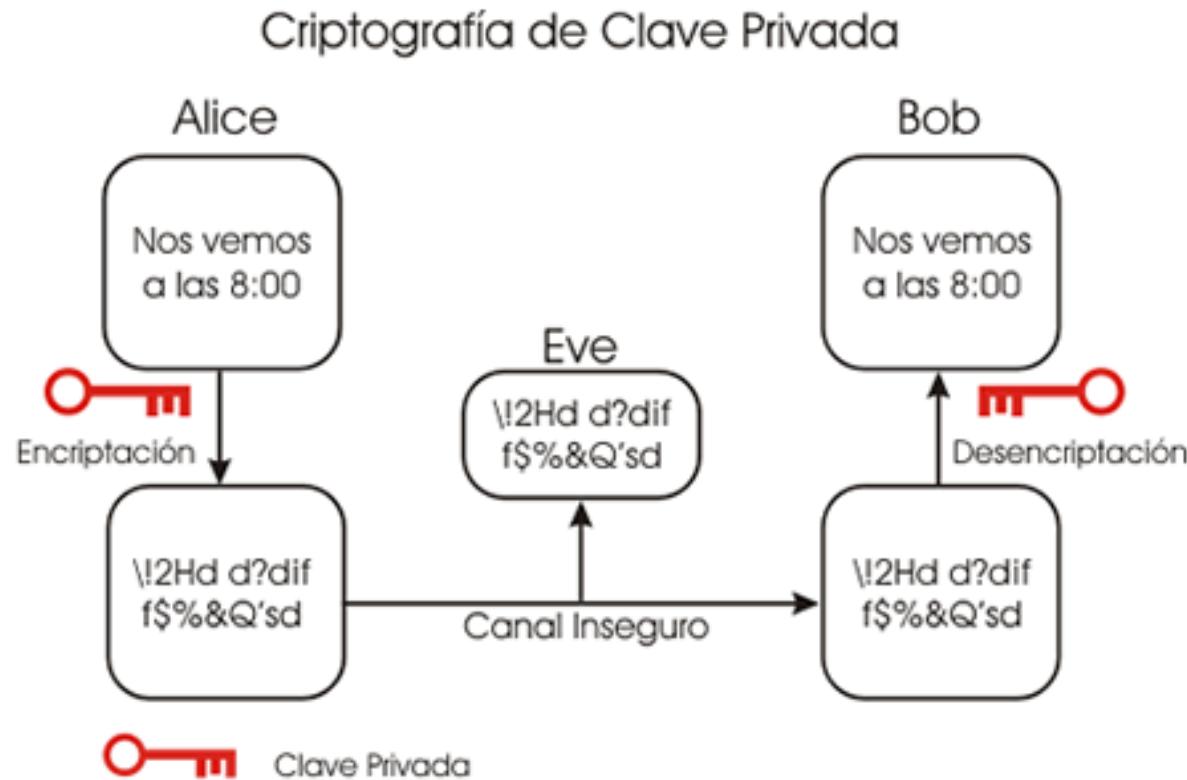


Cifrado de clave Pública y Privada

- **Mecanismos de Seguridad**
- En el **cifrado de clave privada** las claves de cifrado y descifrado son la misma (o bien se deriva de forma directa una de la otra), debiendo mantenerse en secreto dicha clave.
- El cifrado de clave privada, es más rápido que el de clave pública (de 100 a 1000 veces), y por tanto se utiliza generalmente en el intercambio de información una vez establecida una sesión. Estas claves también son conocidas como claves de sesión o de cifrado simétricas, ya que en ambos extremos se posee la misma clave.
 - DES (Data Encryption Standard) y T-DES (o 3DES)
 - IDEA (International Data Encryption Algorithm)
 - AES (Advanced Encryption Standard) o Rijndael

PROTOCOLOS SEGUROS

- **Cifrado de clave privada:** Utiliza una clave para el cifrado y descifrado del mensaje. Esta clave se debe intercambiar entre los equipos por medio de un canal seguro. Ambos extremos deben tener la misma clave para cumplir con el proceso



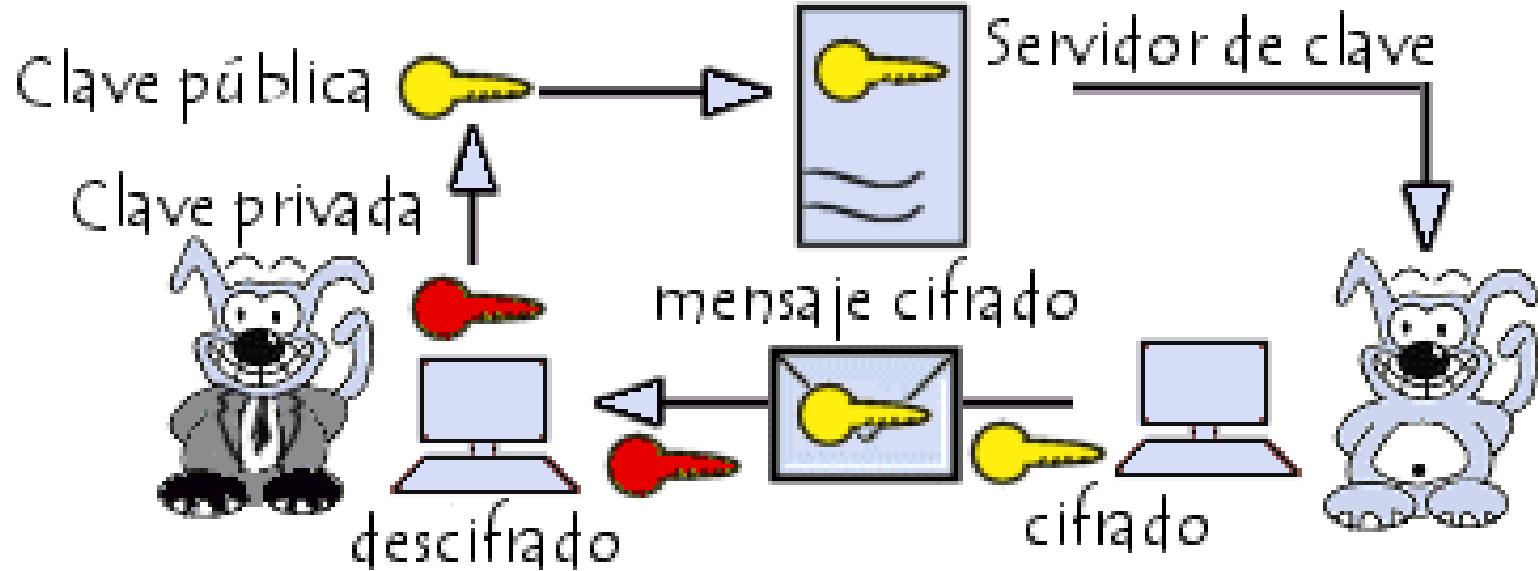


PROTOCOLOS SEGUROS

- Para que un algoritmo de este tipo sea considerado fiable debe cumplir algunos requisitos básicos:
 - Conocido el criptograma (texto cifrado) no se pueden obtener de él ni el texto en claro ni la clave.
 - Conocidos el texto en claro y el texto cifrado debe resultar más caro en tiempo o dinero descifrar la clave que el valor posible de la información obtenida por terceros.
 - Las principales desventajas de los métodos simétricos son la distribución de las claves, el peligro de que muchas personas deban conocer una misma clave y la dificultad de almacenar y proteger muchas claves diferentes.

PROTOCOLOS SEGUROS

- **Cifrado de clave pública:** se basa en el uso de dos claves diferentes, claves que poseen una propiedad fundamental: una clave puede descifrar lo que la otra ha cifrado. Una de las claves de la pareja, llamada clave privada, es usada por el propietario para cifrar los mensajes, mientras que la otra, llamada clave pública, es usada para descifrar el mensaje





PROTOCOLOS SEGUROS

- Las claves pública y privada tienen características matemáticas especiales, de tal forma que se generan siempre a la vez, por parejas, estando cada una de ellas ligada intrínsecamente a la otra.
- Mientras que la clave privada debe mantenerla en secreto su propietario, ya que es la base de la seguridad del sistema, la clave pública es difundida, para que esté al alcance del mayor número posible de personas, existiendo servidores que guardan, administran y difunden dichas claves.

PROTOCOLOS SEGUROS

- Para que un algoritmo de clave pública sea considerado seguro debe cumplir con los siguientes puntos:
 - Conocido el texto cifrado no debe ser posible encontrar el texto en claro ni la clave privada.
 - Conocido el texto cifrado (criptograma) y el texto en claro debe resultar más caro en tiempo o dinero descifrar la clave que el valor posible de la información obtenida por terceros.
 - Conocida la clave pública y el texto en claro no se puede generar un criptograma correcto cifrado con la clave privada.
 - Dado un texto cifrado con una clave privada sólo existe una pública capaz de descifrarlo, y viceversa.
 - El primer sistema de clave pública que apareció fue el de Diffie-Hellman, en 1976, y fue la base para el desarrollo de los que después aparecieron, entre los que cabe destacar el RSA (el más utilizado en la actualidad).

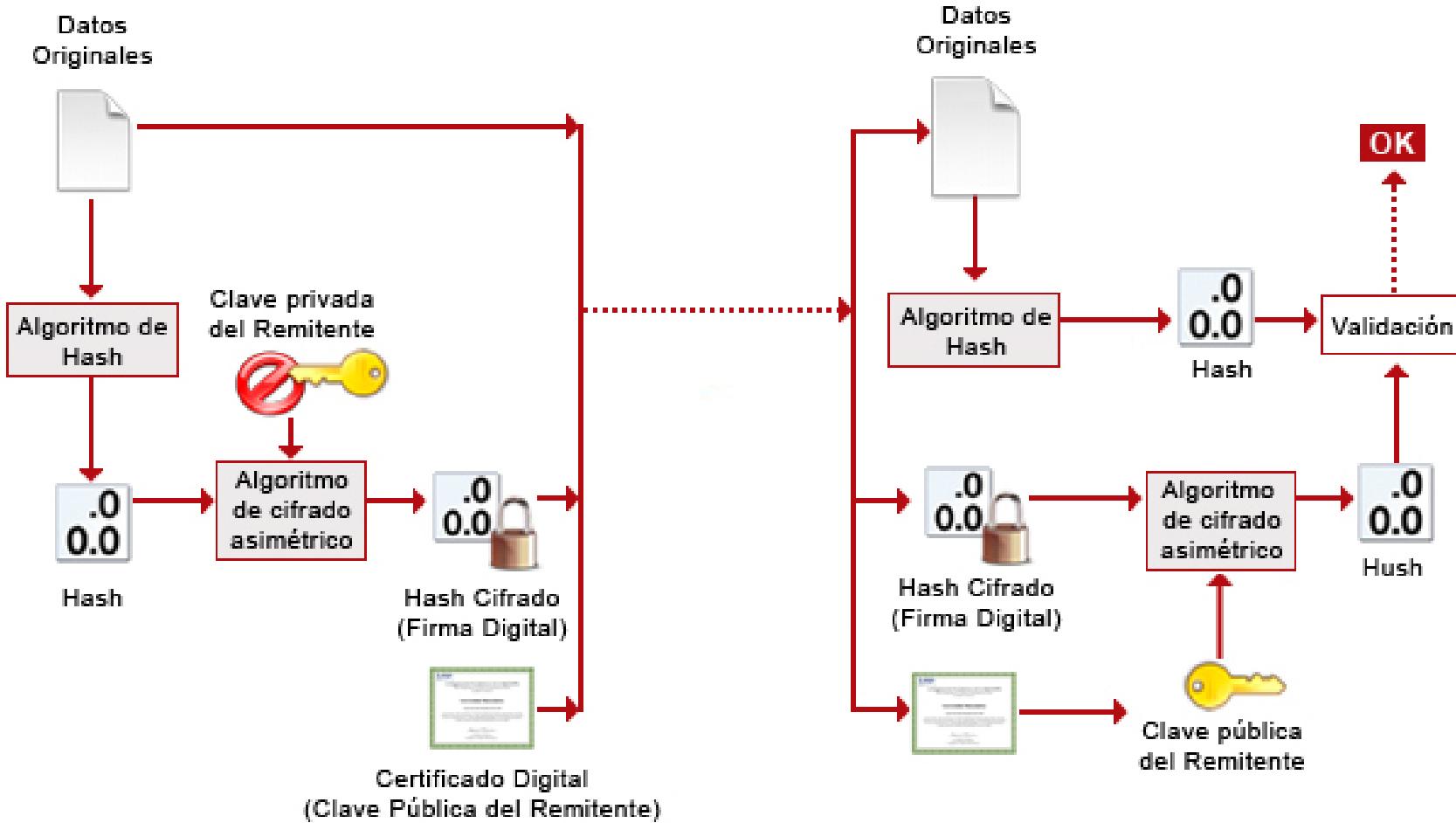


Firma Digital

PROTOCOLOS SEGUROS

- **Las Firmas electrónicas** (también llamadas *firmas digitales*) es un proceso que hace posible garantizar la autenticidad del remitente (función de *autenticación*) y verificar la integridad del mensaje recibido.
- Las firmas electrónicas también poseen una función de reconocimiento de autoría, es decir, hacen posible garantizar que el remitente ha enviado verdaderamente el mensaje (en otras palabras, se aseguran de que el remitente no pueda negar el envío del mensaje).
- La firma electrónica asegura que nuestras transacciones electrónicas cumplen estas garantías.
- Una firma electrónica es un **conjunto de datos, en forma electrónica que, consignados junto a otros o asociados con ellos, pueden ser utilizados como medio de identificación del firmante.**

PROTOCOLOS SEGUROS





Certificado Digital

PROTOCOLOS SEGUROS

- Los algoritmos de cifrado asimétrico se basan en el hecho de compartir una clave pública entre varios usuarios.
- Sin embargo, **no garantizan que la clave pertenezca al usuario con el que está asociada**. Un hacker puede corromper la clave pública que aparece en el directorio remplazándola con su propia clave pública.
- Un certificado permite asociar una clave pública con una entidad (una persona, un equipo, etc.) para garantizar su validez.
- El certificado es como la tarjeta de identificación de la clave, emitido por una entidad llamada *Entidad de certificación (CA)*.
- La entidad de certificación es responsable de emitir los certificados, de asignarles una fecha de validez (similar a la fecha de vencimiento de los alimentos) y de revocarlos antes de esta fecha en caso de que la clave (o su dueño) estén en una situación de riesgo.

PROTOCOLOS SEGUROS

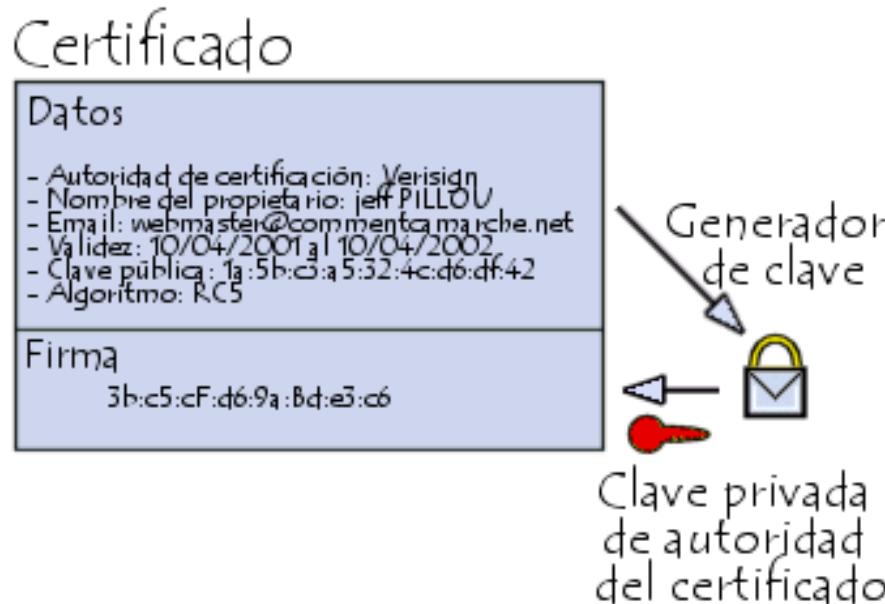
➤ Estructura de los certificados

- Los certificados son pequeños archivos divididos en dos partes:
 - La parte que contiene la información
 - La parte que contiene la firma de la entidad de certificación
- La estructura de los certificados está estandarizada por la norma **X.509** (más precisamente, X.509v3) de la [UIT](#), que define la información que contiene el certificado:
 - La versión de X.509 a la que corresponde el certificado;
 - El número de serie del certificado;
 - El algoritmo de cifrado utilizado para firmar el certificado;
 - El nombre (DN, siglas en inglés de *Nombre distinguido*) de la entidad de certificación que lo emite;
 - La fecha en que entra en vigencia el certificado;
 - La fecha en que finaliza el período de validez del certificado;
 - El objeto de utilización de la clave pública;
 - La clave pública del dueño del certificado;
 - La firma del emisor del certificado (*huella digital*).

PROTOCOLOS SEGUROS

➤ **Estructura de los certificados**

- La entidad de certificación firma toda esta información (información + clave pública del solicitante) y esto implica que una función hash crea una huella digital de esta información y luego este hash se cifra con la clave privada de la entidad de certificación.
- La clave pública se distribuye antes de tiempo para permitir a los usuarios verificar la firma de la *entidad de certificación* con su clave pública.





PROTOCOLOS SEGUROS

➤ Firmas del certificado

- Existen varios tipos de certificados en función del nivel de sus firmas:
 - **Los certificados firmados localmente** son certificados de uso interno. Al estar firmados por un servidor local, este tipo de certificados permiten garantizar los intercambios confidenciales dentro de una organización, por ejemplo, en una Intranet. Los certificados firmados localmente se pueden usar para autenticar usuarios.
 - **Los certificados firmados por una entidad de certificación** son necesarios cuando se deben garantizar los intercambios seguros con usuarios anónimos, por ejemplo, en el caso de una página Web segura al que pueda acceder el público general. La certificación de un tercero garantiza al usuario que el certificado pertenece efectivamente a la organización a la que dice pertenecer.

PROTOCOLOS SEGUROS

➤ **Tipos de uso**

- Los certificados se utilizan principalmente en tres tipos de contextos:
 - **Los certificados de cliente** se almacenan en la estación de trabajo del usuario o se integran en un contenedor como una tarjeta inteligente, y permiten identificar a un usuario y asociarlo con ciertos privilegios. En la mayoría de los casos, se transmiten al servidor cuando se establece una conexión y el servidor asigna privilegios en función de la acreditación del usuario. Son verdaderas tarjetas de identificación digitales que usan un par de claves asimétricas con una longitud de 512 a 1024 bits.
 - **Los certificados de servidor** se instalan en un servidor Web y permiten conectar un servicio con el dueño del servicio. En el caso de página Web, permiten garantizar que la dirección URL de la página Web y especialmente su dominio pertenecen realmente a tal o cual compañía. También permiten proteger las transacciones con usuarios gracias al protocolo SSL.
 - **Los certificados VPN** se instalan en un equipo de red y permiten cifrar flujos de comunicación de extremo a extremo entre dos puntos (p. ej., dos ubicaciones de una compañía). En este tipo de escenario, los usuarios tienen un certificado cliente, los servidores aplican un certificado de servidor y el equipo de comunicación usa un certificado especial (generalmente un certificado IPSec).



Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red
2. Servicio de Nombres de Dominio (DNS)
3. La navegación Web
4. El Correo electrónico
5. Seguridad & protocolos seguros
6. **Aplicaciones multimedia**
7. Aplicaciones para interconectividad de redes locales
8. Cuestiones y ejercicios



➤ Definiciones:

Para A. Bartolomé (1994) :

“Los sistemas Multimedia, en el sentido que hoy se da al término, son básicamente sistemas interactivos con múltiples códigos”

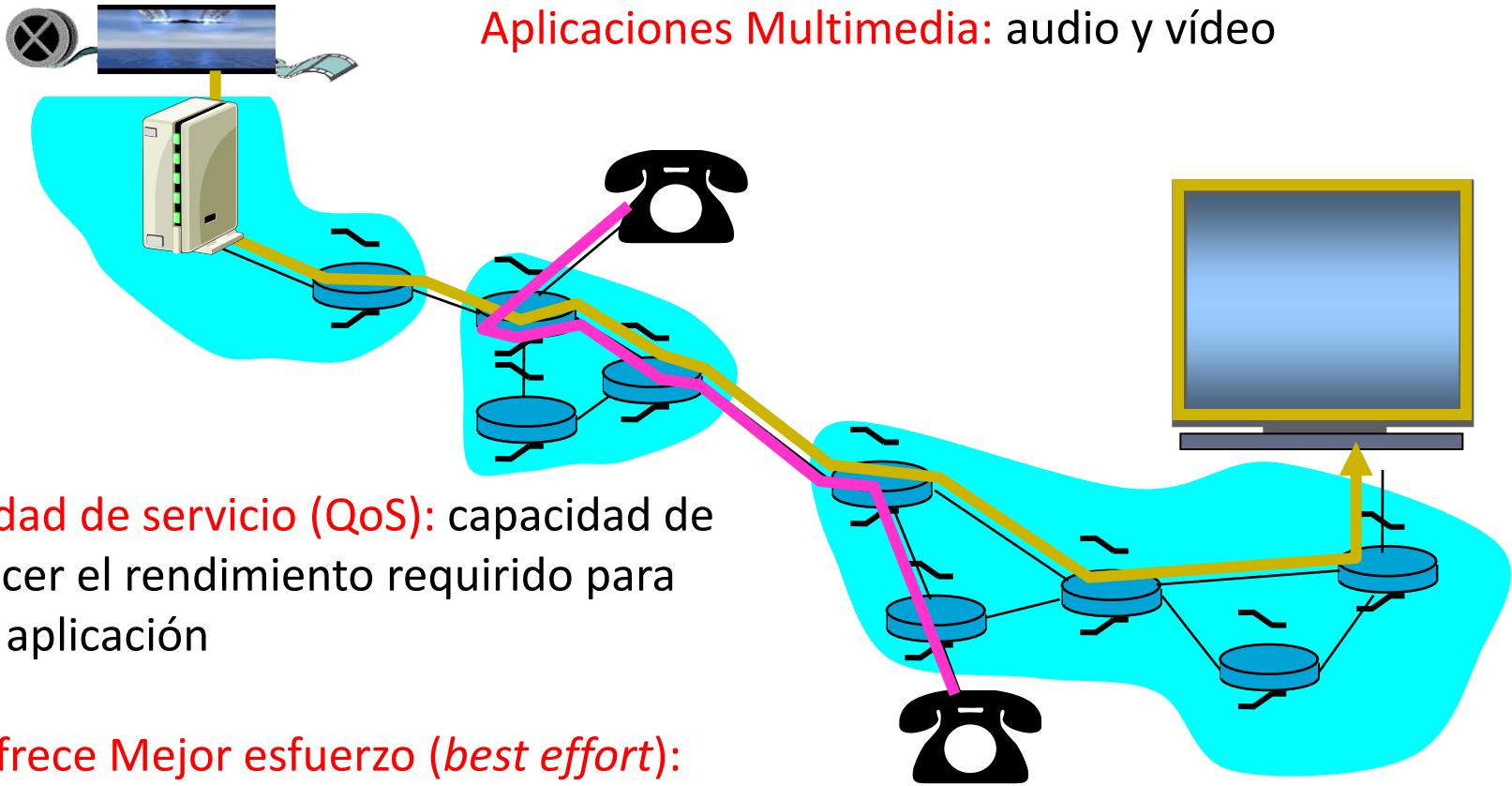
Según Fred Hoffstetter:

“Multimedia es el uso del ordenador para presentar y combinar: texto, gráficos, audio y vídeo con enlaces que permitan al usuario navegar, interactuar, crear y comunicarse”.

➤ Concepto de Aplicación multimedia:

- El término multimedia hace referencia al uso combinado de diferentes medios de comunicación: texto, imagen, sonido, animación y video.
- Los programas informáticos que utilizan de forma combinada y coherente con sus objetivos diferentes medios, y permiten la interacción con el usuario son aplicaciones multimedia interactivas.
- La evolución producida en los sistemas de comunicación ha dado lugar a este tipo heterogéneo de aplicaciones o programas que tienen dos características básicas:
 - **Multimedia:** Uso de múltiples tipos de información (textos, gráficos, sonidos, animaciones, videos, etc.) integrados coherentemente.
 - **Hipertexto:** Interactividad basada en los sistemas de hipertexto, que permiten decidir y seleccionar la tarea que deseamos realizar, rompiendo la estructura lineal de la información.

➤ Conceptos: IP = “*tecnología de convergencia*” →



➤ Tipos de aplicaciones

- Flujo de audio y vídeo (*streaming*) almacenado ➔ Ej. YouTube
- Flujo de audio y vídeo en vivo ➔ Ej. emisoras de radio o IPTV
- Audio y vídeo interactivo ➔ Ej. Skype

➤ Características fundamentales

- Elevado ancho de banda
- Tolerantes relativamente a la pérdida de datos
- Exigen *Delay* acotado
- Exigen *Jitter* acotado
- Se pueden beneficiar de usar de *multicast*

- **Jitter:** la variabilidad temporal durante el envío de señales digitales, una ligera desviación de la exactitud de la señal de reloj. El jitter suele considerarse como una señal de ruido no deseada. En general se denomina jitter a un cambio indeseado y abrupto de la propiedad de una señal. El jitter es la primera consecuencia de un retraso de la señal.
- **Delay:** medida de tiempo en la que un paquete tarda en viajar desde un origen hasta un destino. Esta unidad de medida no es fácil de obtener con exactitud en una red debido a la dificultad de la sincronización del reloj de los puntos extremos, que, normalmente, están lejanos entre sí. A fin de superar esta dificultad, la latencia se mide como el tiempo de ida y vuelta dividido en dos, cuyo nombre en inglés es Round Trip Time (RTT).

Network Delivery Issues



Stalling

Low Quality Source or Overly Aggressive Optimization



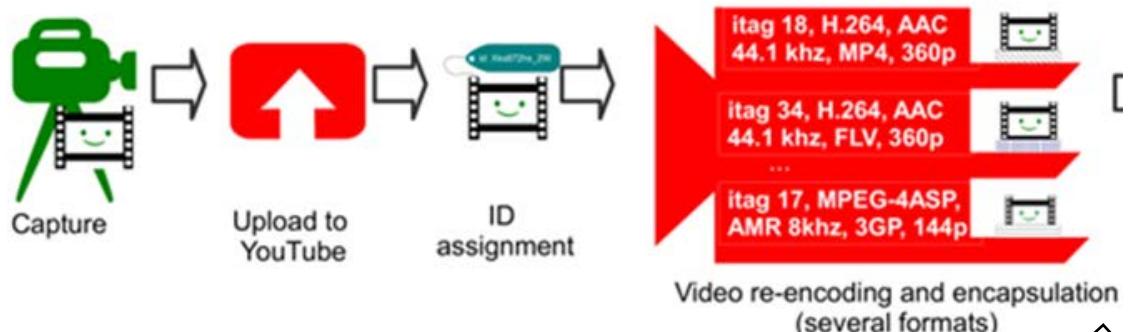
Blurriness



Macroblocking (loss of packets)

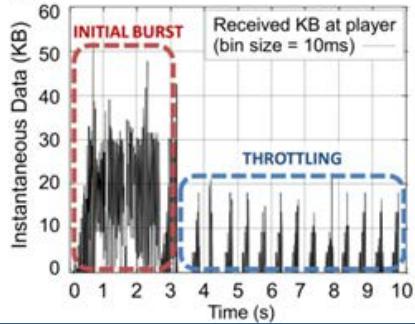
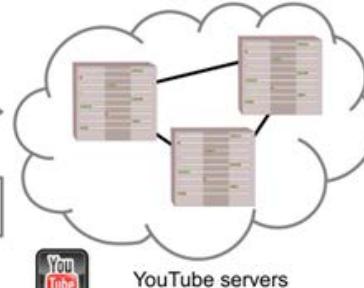


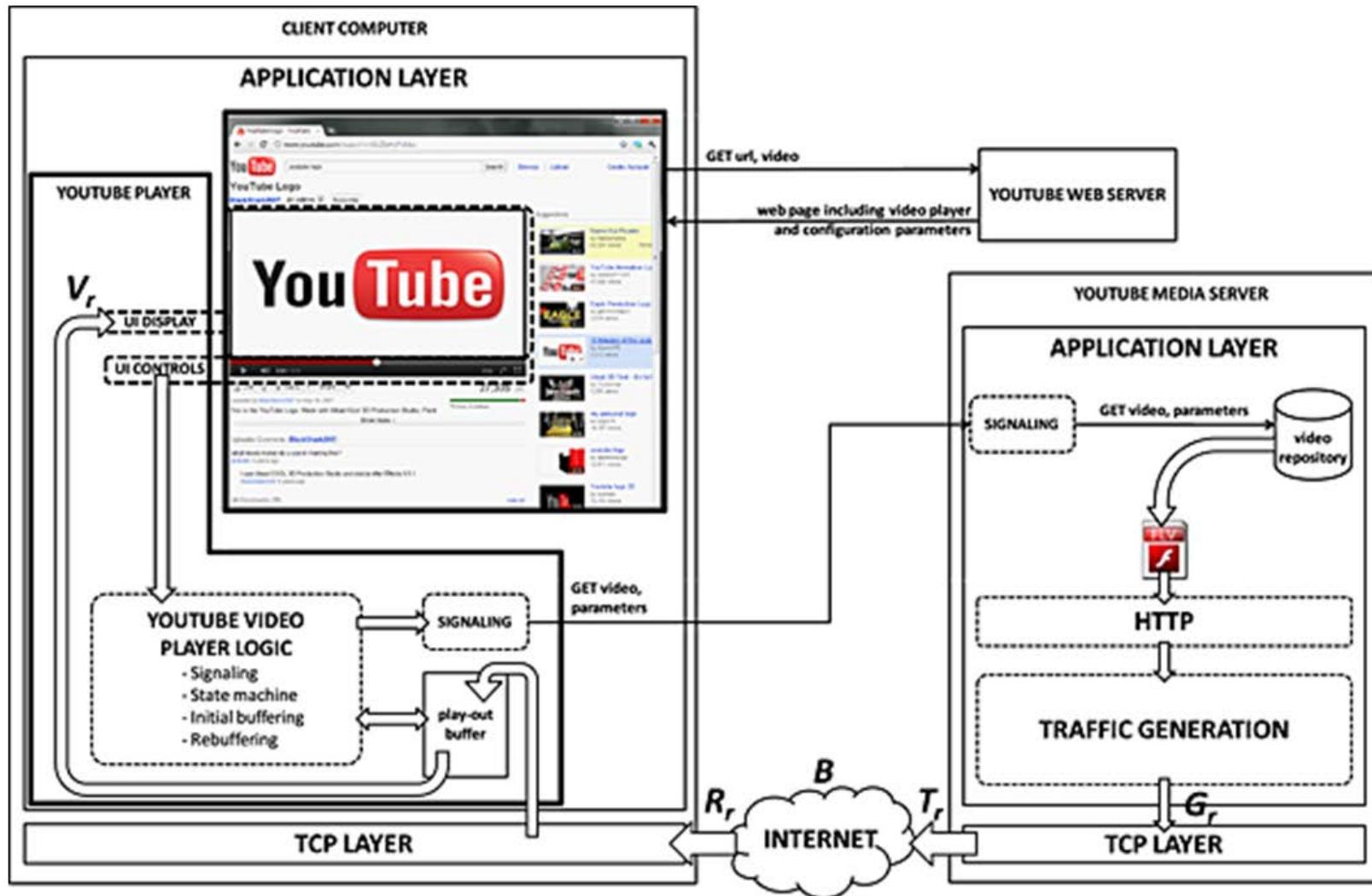
Blockiness

Uploading**Downloading**

Video request (video id, itag, Range)

Media content







Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red
2. Servicio de Nombres de Dominio (DNS)
3. La navegación Web
4. El Correo electrónico
5. Protocolos seguros
6. Aplicaciones multimedia
- 7. Aplicaciones para interconectividad de redes locales**
8. Cuestiones y ejercicios

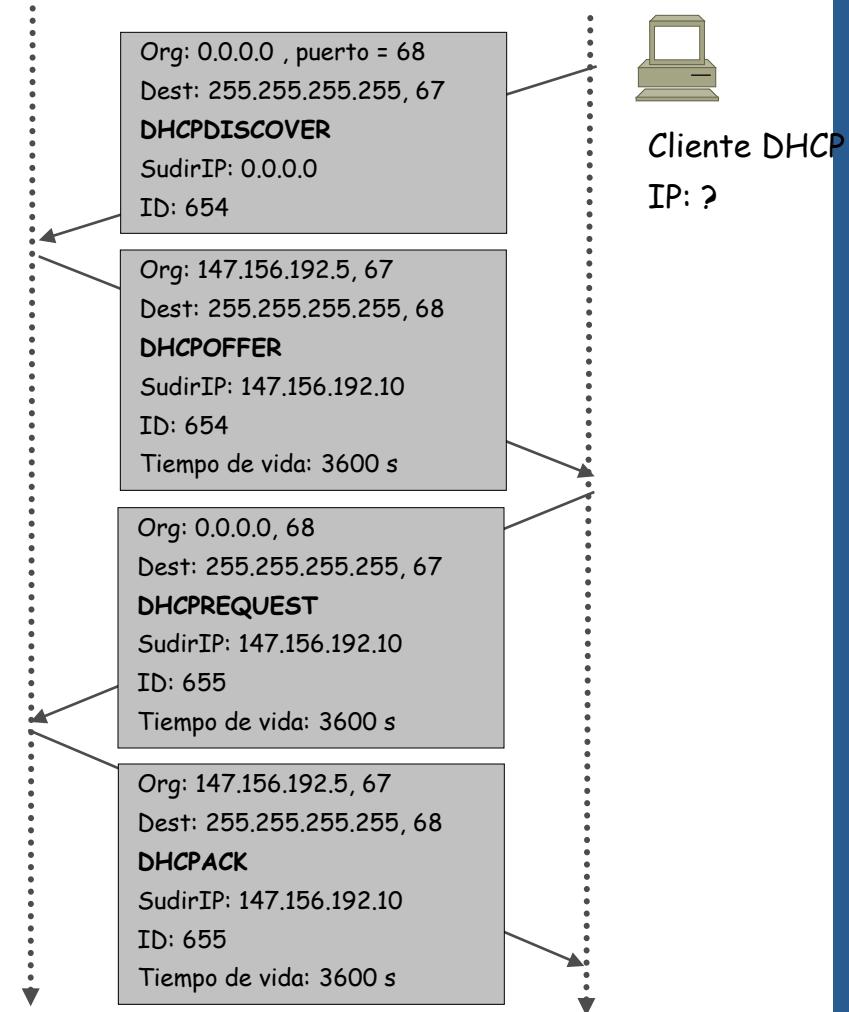
APLICACIONES PARA INTERCONECTIVIDAD DE REDES LOCALES: DHCP

➤ DHCP (Dynamic Host Configuration Protocol)


Servidor DHCP
147.156.192.5

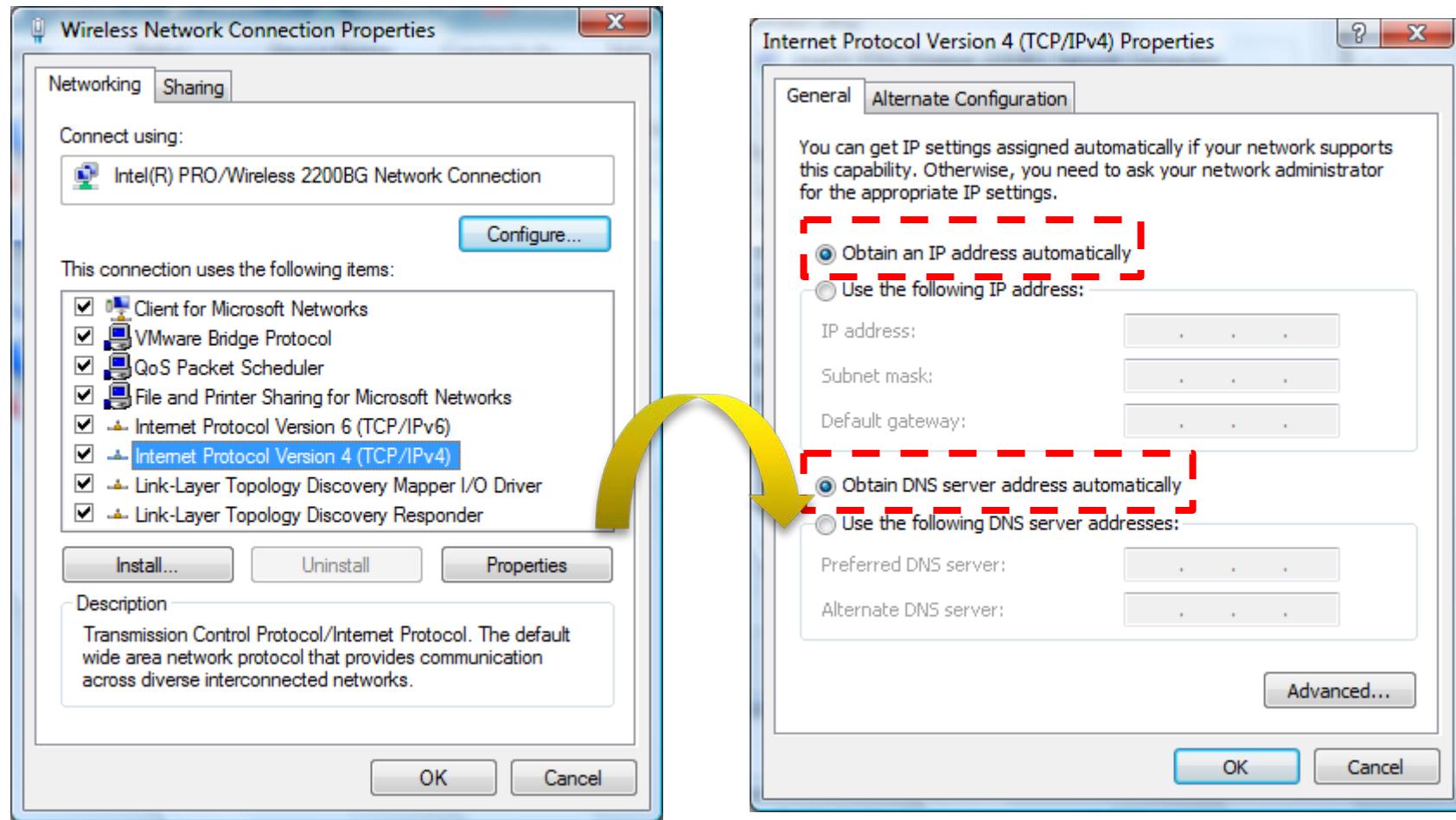
Para asignar las direcciones se usa **DHCP** (RFC 2131-3396), protocolo usuario de UDP (**puerto 67**)

- El host (cliente) envía un mensaje *broadcast*: "DHCP discover"
- El server DHCP responde con un mensaje "DHCP offer"
- El host solicita una dirección IP, mensaje "DHCP request"
- El server DHCP envía la dirección IP: mensaje "DHCP ack"



APLICACIONES PARA INTERCONECTIVIDAD DE REDES LOCALES: DHCP

Configuración de un cliente MS Windows:





APLICACIONES PARA INTERCONECTIVIDAD DE REDES LOCALES: DHCP

Configuración de un cliente Linux (Fedora Core dist.):

```
# Sample /etc/sysconfig/network-scripts/ifcfg-eth0 :  
  
DEVICE=eth0  
BOOTPROTO=dhcp  
HWADDR=00:0C:29:CE:63:E3  
ONBOOT=yes  
TYPE=Ethernet
```

Configuración de un servidor de Linux (*dhcpd*):

```
# Sample /etc/dhcpd.conf  
  
default-lease-time 600;max-lease-time 7200;  
option subnet-mask 255.255.255.0;  
option broadcast-address 192.168.1.255;  
option routers 192.168.1.254;  
option domain-name-servers 192.168.1.1, 192.168.1.2;  
option domain-name "mydomain.org";  
subnet 192.168.1.0 netmask 255.255.255.0 {  
    range 192.168.1.10 192.168.1.100;  
    range 192.168.1.150 192.168.1.200;  
}  
  
# Static IP address assignment  
host haagen {  
    hardware ethernet 08:00:2b:4c:59:23;  
    fixed-address 192.168.1.222;  
}
```



Tema 2. SERVICIOS Y PROTOCOLOS DE APLICACIÓN EN INTERNET

1. Introducción a las aplicaciones de red
2. Servicio de Nombres de Dominio (DNS)
3. La navegación Web
4. El Correo electrónico
5. Protocolos seguros
6. Aplicaciones multimedia
7. Aplicaciones para interconectividad de redes locales
- 8. Cuestiones y ejercicios**



CUESTIONES Y EJERCICIOS

1. Discuta las características de las siguientes aplicaciones en términos de su tolerancia a la pérdida de datos, los requisitos temporales, la necesidad de rendimiento mínimo y la seguridad.

La telefonía móvil

WhatsApp

YouTube

Spotify

Comercio electrónico

TEMA 2

SERVICIOS Y PROTOCOLOS

DE APLICACIÓN EN INTERNET

Fundamentos de Redes

2017/2018



ugr
Universidad
de Granada