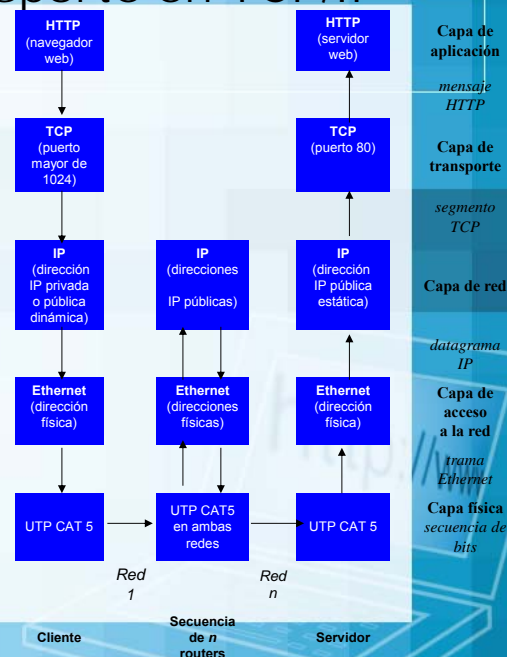


Nivel de Transporte en Internet



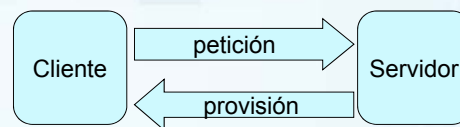
Nivel de Transporte en TCP/IP

- La capa de transporte transmite mensajes entre las **aplicaciones** de dos ordenadores.
- La programación de aplicaciones sobre el nivel de transporte se realiza de acuerdo al **paradigma cliente/servidor**.



Paradigma cliente/servidor

- Los procesos servidores:
 - ofrecen un servicio concreto el cual esta siempre disponible para los usuarios del servicio
- Los procesos clientes
 - piden un servicio en un momento dado.
- Cliente y servidor se intercambiarán mensajes utilizando alguno de los dos protocolos de transporte de Internet: TCP o UDP.
- Concepto de PUERTO
- La estructura general de estas aplicaciones es:



Cliente: ... Petición_servicio() Servicio(...)	Servidor: WHILE TRUE DO Espera_petición(...) Servicio(...) END
--	---

Protocolo UDP

- User Datagram Protocol
- Proporciona un servicio de envío y recepción de datagramas
 - No orientado a la conexión
 - No fiable
- Primitivas de servicio
 - Send (origen, destino, datos)
 - Recv (origen, destino, datos)

0										10										20										30	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Puerto UDP origen															Puerto UDP destino																
Longitud del datagrama															Checksum UDP																
Datos																															
...																															

Longitud mínima: 8 bytes
Longitud máxima: 65515 (65535 – 20)

Longitud mínima: 8 bytes
 Longitud máxima: 65515 (65535 – 20)

Protocolo TCP

- Transmission Control Protocol
 - Transporta y regula el control de flujo de extremo a extremo
 - Orientado a la conexión
 - Fiable
 - Flujo no estructurado: flujo de bytes
 - Memoria intermedia → buffer
 - Conexiones punto a punto y full-duplex
 - Extremo de una conexión → <TCP, DIRIP, PUERTO>
 - Conexión →
- <TCP, dir IP origen, puerto origen, dirIP destino, puerto destino >**
- Protocolo con tres fases:
 - Establecimiento de la conexión
 - Transferencia de Datos – Control de flujo ventana deslizante
 - Terminación de la conexión

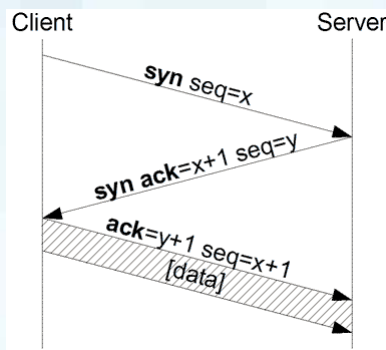
Segmento TCP

- Números de secuencia → primer bytes de datos del segmento
- Indicadores (6 bits) → Funciones de Control:
 - URG. Datos urgentes (+NS: donde + marcador: cuantos)
 - ACK. acuse de recibo
 - PSH. Enviar datos existentes sin esperar a completar segmento
 - RST. Reset de la conexión
 - SYN. Sincronización número de secuencia
 - FIN. Cierre de conexión

0										10										20										30									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Puerto TCP origen															Puerto TCP destino																								
Número de secuencia																																							
Número de acuse de recibo																																							
HLEN				Reservado						Indicadores								Ventana																					
Checksum TCP															Marcador urgente																								
Opciones																								Relleno															
Datos																																							
...																																							

Protocolo TCP

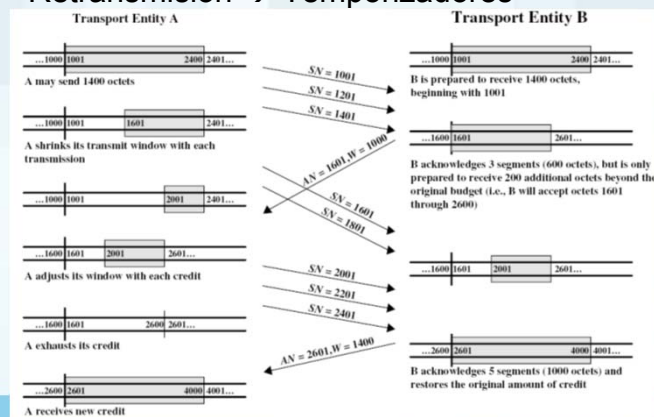
- Fase de Conexión →
 - Negociación en 3 pasos (3-way handshake)
 - Configuración de número de secuencia (32 bits)



- Fase de Desconexión →
 - Negociación en 4 pasos (4-way handshake)
 - Desde cualquier extremo

Protocolo TCP

- Control de congestión →
 - Tamaño ventana adaptable →
 - la cantidad de datos recibidos (en bytes) que pueden ser metidos en el [buffer](#) de recepción durante la conexión.
 - Retransmisión → Temporizadores



Diferencias de uso entre UDP y TCP

- El protocolo TCP
 - Ventajas + Robusto y fiable
 - Desventajas + complejo + lento
- USO de UDP ventajoso
- Aplicaciones tiempo real como audio o video
- Consultas a servidores en que se requiere el envío de uno o dos mensajes únicamente
- Transmisión en modo multicast (a muchos destinos) o en modo broadcast (a todos los destinos)

Interfaz Socket

- SOCKET →
 - Dirección IP + puerto de comunicación + protocolo
- primitivas de llamada a los sockets
 - **SOCKET**: que crea un nuevo punto terminal de conexión.
 - **BIND**: que conecta una dirección local a un socket.
 - **CLOSE**: que cierra socket
- Primitivas para sockets TCP
 - **LISTEN**: que anuncia la disposición de aceptar conexiones, indicando también el tamaño de cola.
 - **ACCEPT**: que bloquea al invocador hasta la llegada de un intento de conexión TCP.
 - **CONNECT**: que intenta establecer activamente una conexión TCP.
 - **SEND**: que envía datos a través de la conexión TCP.
 - **RECEIVE**: que recibe los datos de la conexión TCP.
- Primitivas para sockets UDP
 - **SENDTO**: que envía datos a un socket UDP.
 - **RCVFROM**: que recibe los datos de un socket UDP.

