

*Telefónica*

---



**HOSTED IP**

Protocolo TCP



# Protocolo TCP

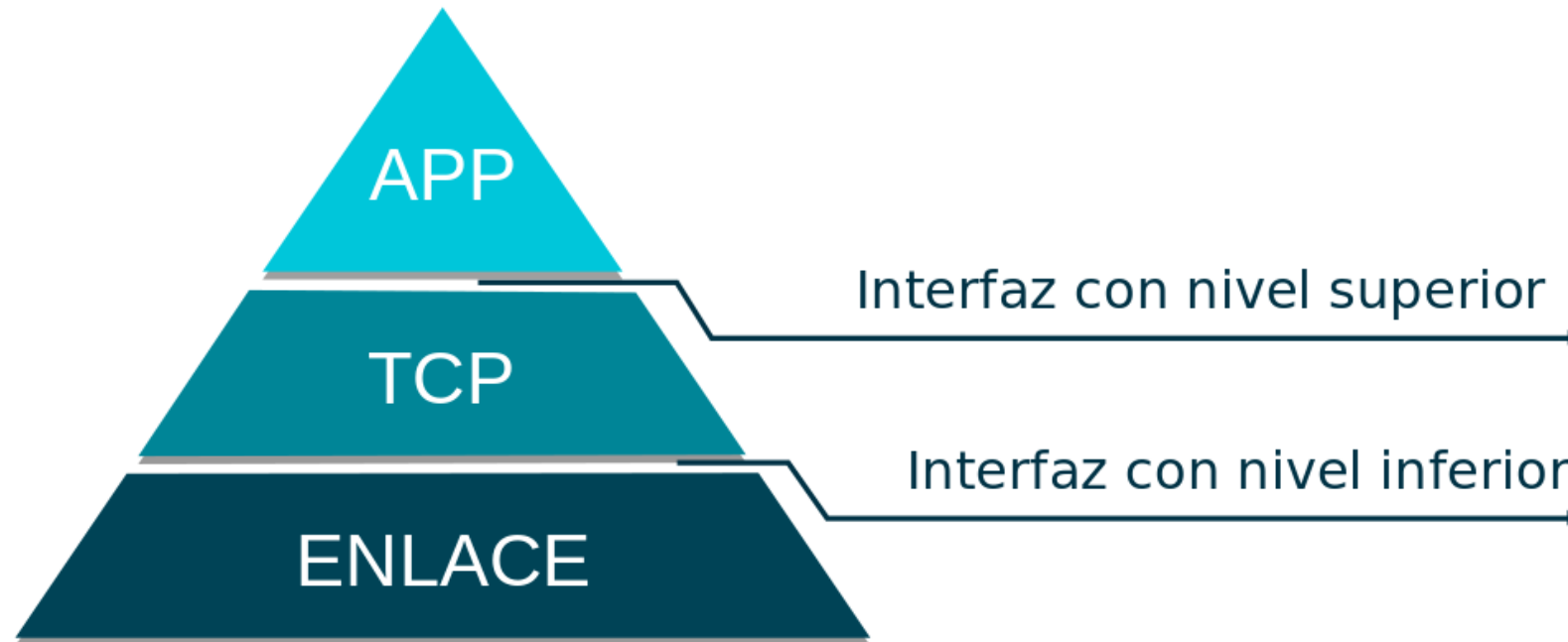
## Introducción



# Protocolo TCP

## Introducción

TCP es un protocolo orientado a la conexión entre dos extremos, fiable y diseñado para encajar en una jerarquía de protocolos



# Protocolo TCP

## Introducción

Es tarea de TCP, asegurar que los datos, se entregan bien, en secuencia y sin errores. Para poder operar sobre un entorno de Internet menos fiable, ofrece:

- Transferencia básica de datos
- Fiabilidad
- Control de flujo
- Multiplexamiento
- Conexiones
- Prioridad y seguridad

# Protocolo TCP

## Introducción

Un flujo de datos enviado sobre una conexión TCP, se entrega de forma fiable y ordenada al destino, para ello, necesita

- Números de secuencia
- Acuses de recibo
- Retransmisiones
- Control de flujo

# Protocolo TCP

## Introducción

Una conexión, empieza llamando a la función OPEN con los argumentos de un puerto local y una dirección de conector remoto

Los procedimientos para establecer conexiones, utilizan el bit de control SYN e involucran un intercambio de tres mensajes

La conexión queda establecida, cuando los números de secuencia quedan sincronizados

# Protocolo TCP

## Introducción

TCP, almacena los datos del usuario emisor y los envía en segmentos según su propia conveniencia, siempre que no se invoque la función **PUSH**

- PUSH, envía al destino, todos los datos que tiene hasta ese momento

Cuando el receptor, ve el indicador PUSH, no debe esperar más datos antes de pasarlos a la aplicación receptora

# Protocolo TCP

Especificación funcional

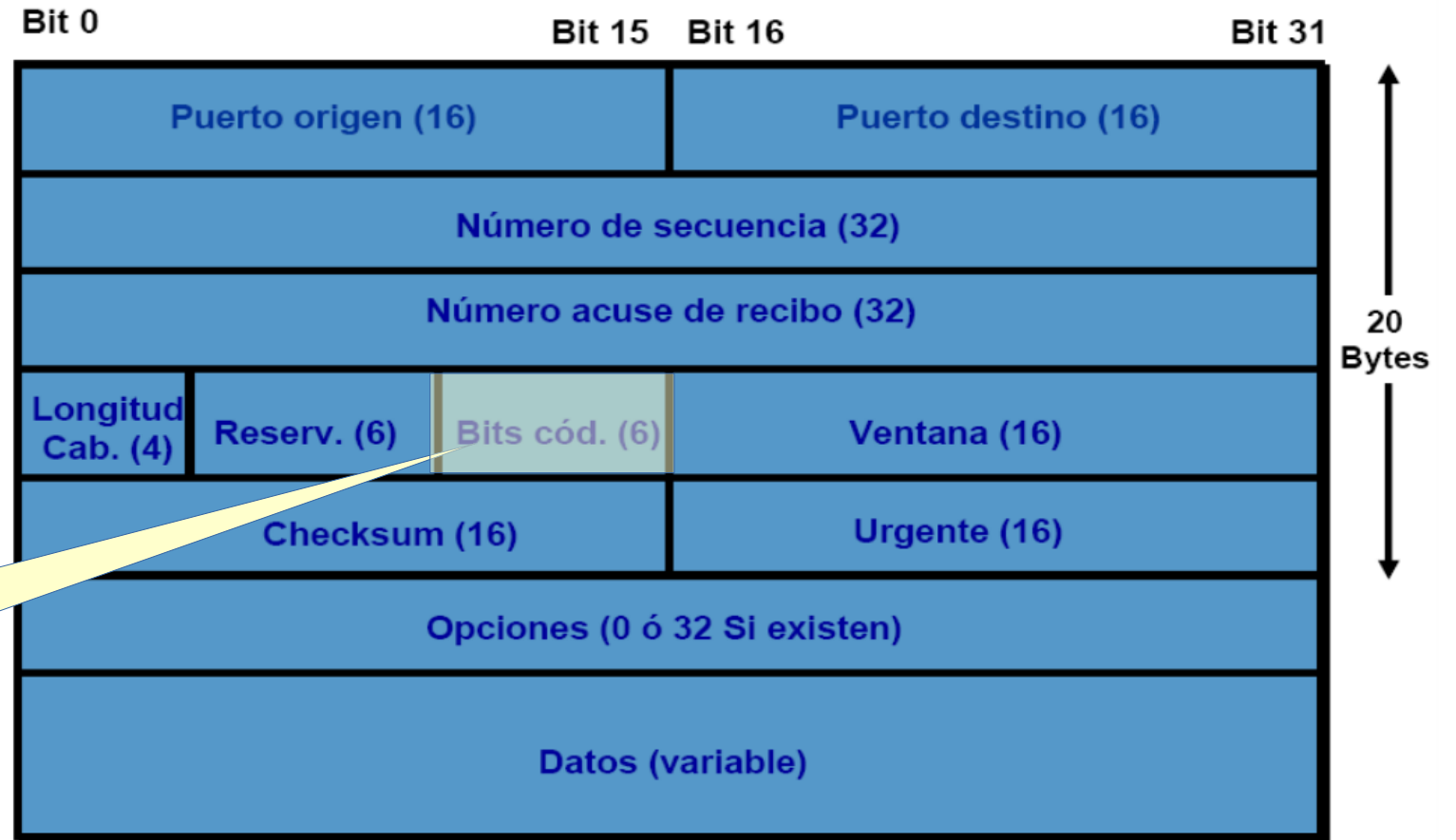
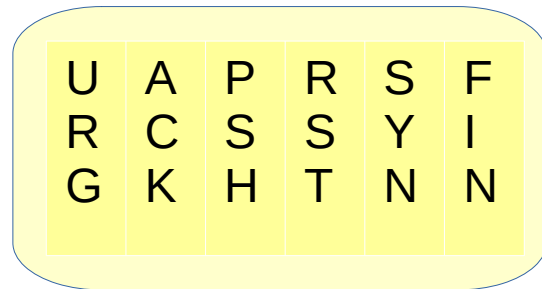




# Protocolo TCP

## Especificación funcional

### Cabecera TCP



# Protocolo TCP

## Especificación funcional

El mantenimiento de una conexión TCP, requiere el almacenamiento y seguimiento de varias variables, almacenadas en un registro de conexión

- Dirección de conector local y remoto
- Valores de seguridad y prioridad de la conexión
- Puntero al bufer de envío y recepción del usuario
- Puntero a la cola de retransmisión
- Variables relacionadas con los números de secuencia de envío y recepción

# Protocolo TCP

## Especificación funcional

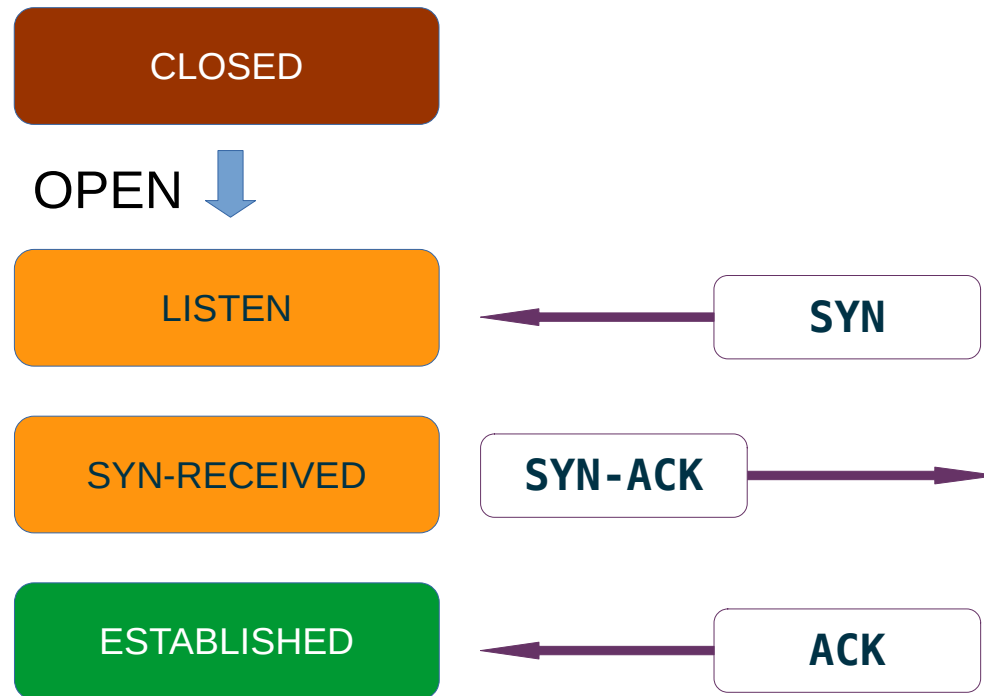
Una conexión TCP, progresa de acuerdo a una serie de estados y en función de respuesta a eventos

- OPEN, CLOSE
- SEND, RECEIVE
- ABORT
- STATUS
- Llegadas de segmentos con indicadores (SYN, ACK,, RST, FIN)
- La expiración de plazos de tiempo

# Protocolo TCP

## Especificación funcional

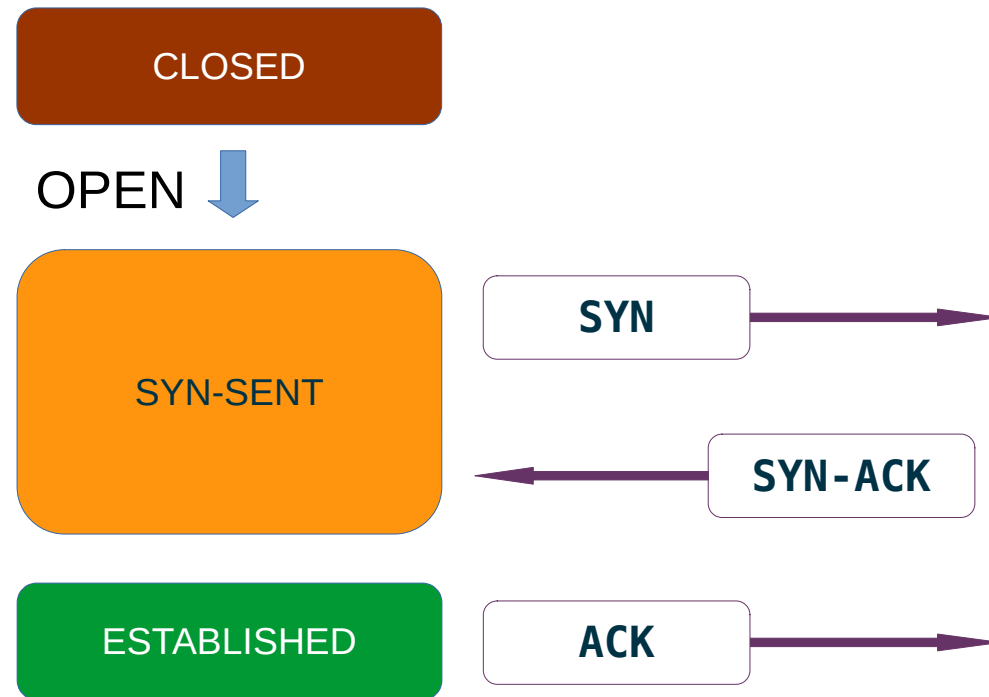
### Estados en el establecimiento de una conexión **PASIVA**



# Protocolo TCP

## Especificación funcional

### Estados en el establecimiento de una conexión **ACTIVA**



# Protocolo TCP

## Especificación funcional

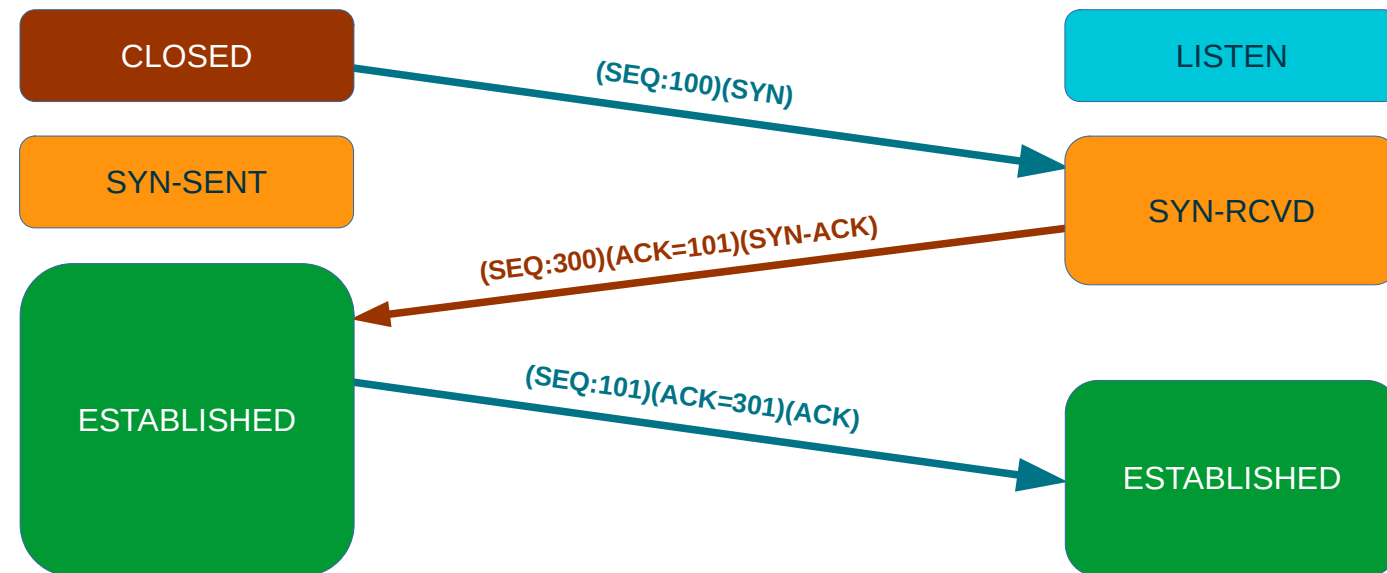
Recordamos que:

- Todo octeto de datos enviado por una conexión TCP, tiene un número de secuencia y debe ser validado por un acuse de recibo
- El acuse de recibo, es acumulativo, y una vez un segmento ha sido validado, es eliminado del bufer de retransmisión
- Para que una conexión quede establecida, los dos TCP deben sincronizar sus números de secuencia

# Protocolo TCP

## Especificación funcional

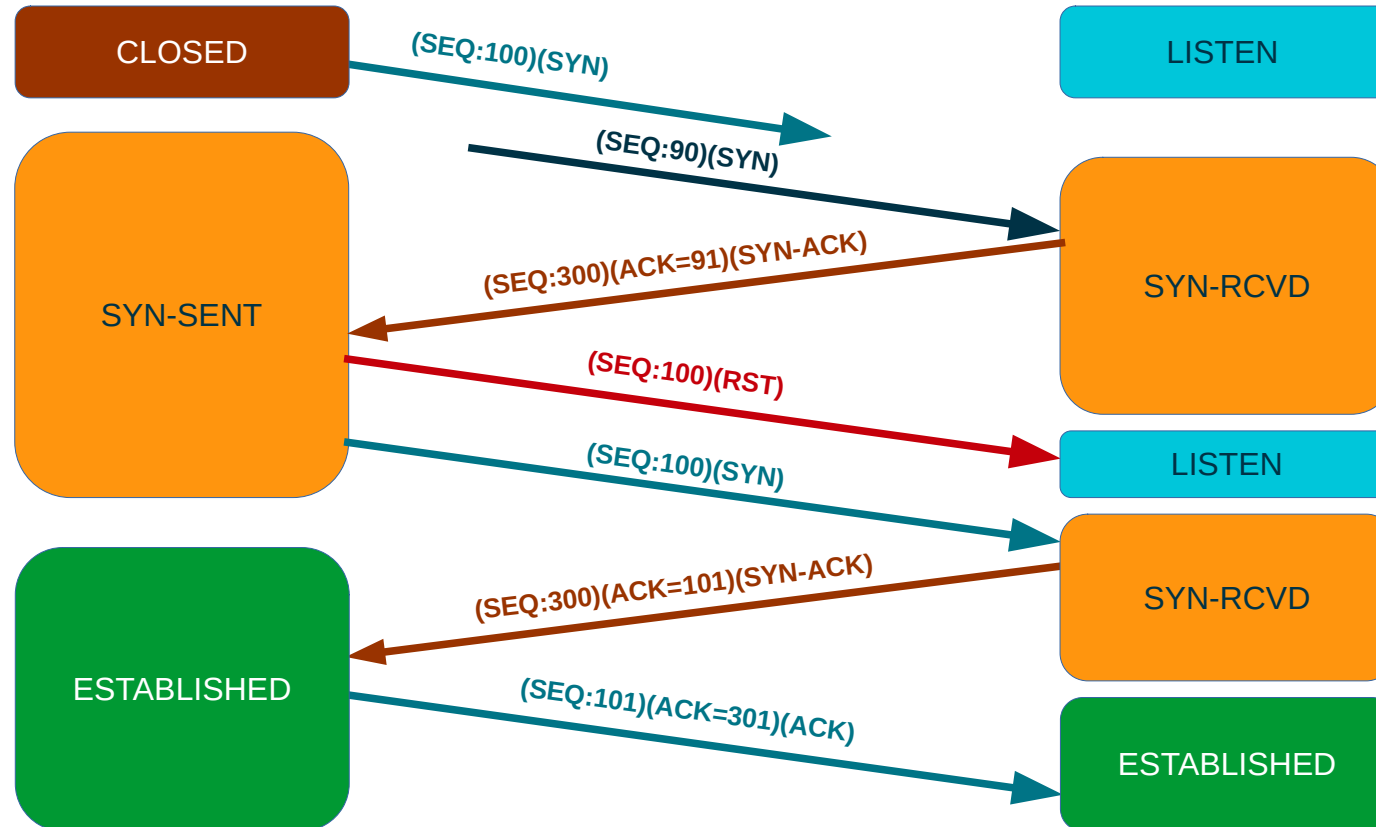
Establecimiento normal de una conexión en tres pasos, en caso de caída, existe un tiempo de silencio



# Protocolo TCP

## Especificación funcional

La llegada de un segmento SYN duplicado, hace creer al receptor, que está en un proceso de una conexión simultánea

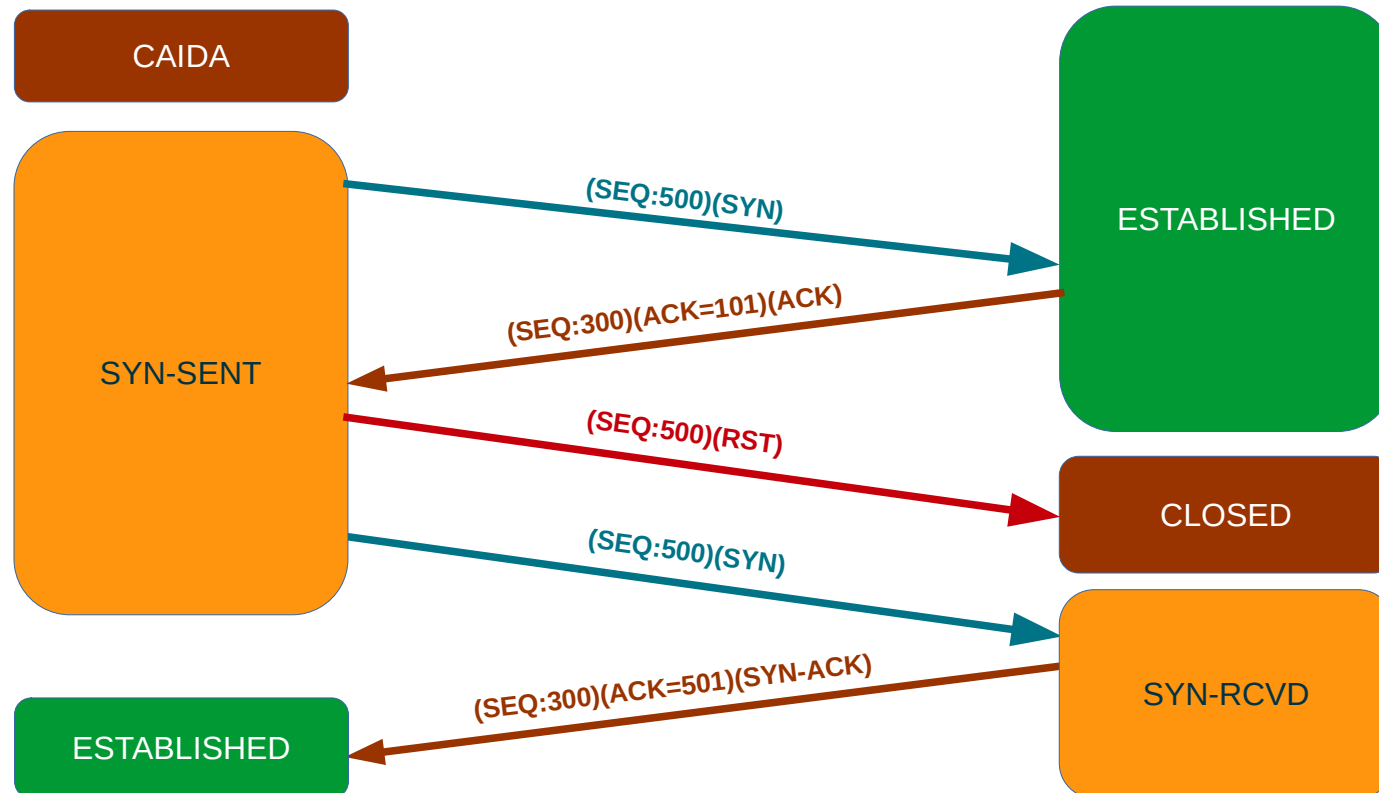




# Protocolo TCP

## Especificación funcional

Una conexión, se dice que está medio abierta, si uno de los TCP ha cerrado la conexión por una caída



# Protocolo TCP

## Especificación funcional

Un RESET, es una señal TCP que se envía con el fin de comunicar al receptor, que queremos romper la comunicación

Como regla general, se envía un RESET siempre que llegue un segmento que no está destinado a la conexión en curso

- Si la conexión no existe y se recibe cualquier segmento
- En estado no sincronizado, se recibe un segmento que confirma uno no enviado
- En estado sincronizado, cualquier segmento inaceptable, provoca el envío de un segmento sin datos, con el número de secuencia y ACK, que espera recibir

# Protocolo TCP

## Especificación funcional

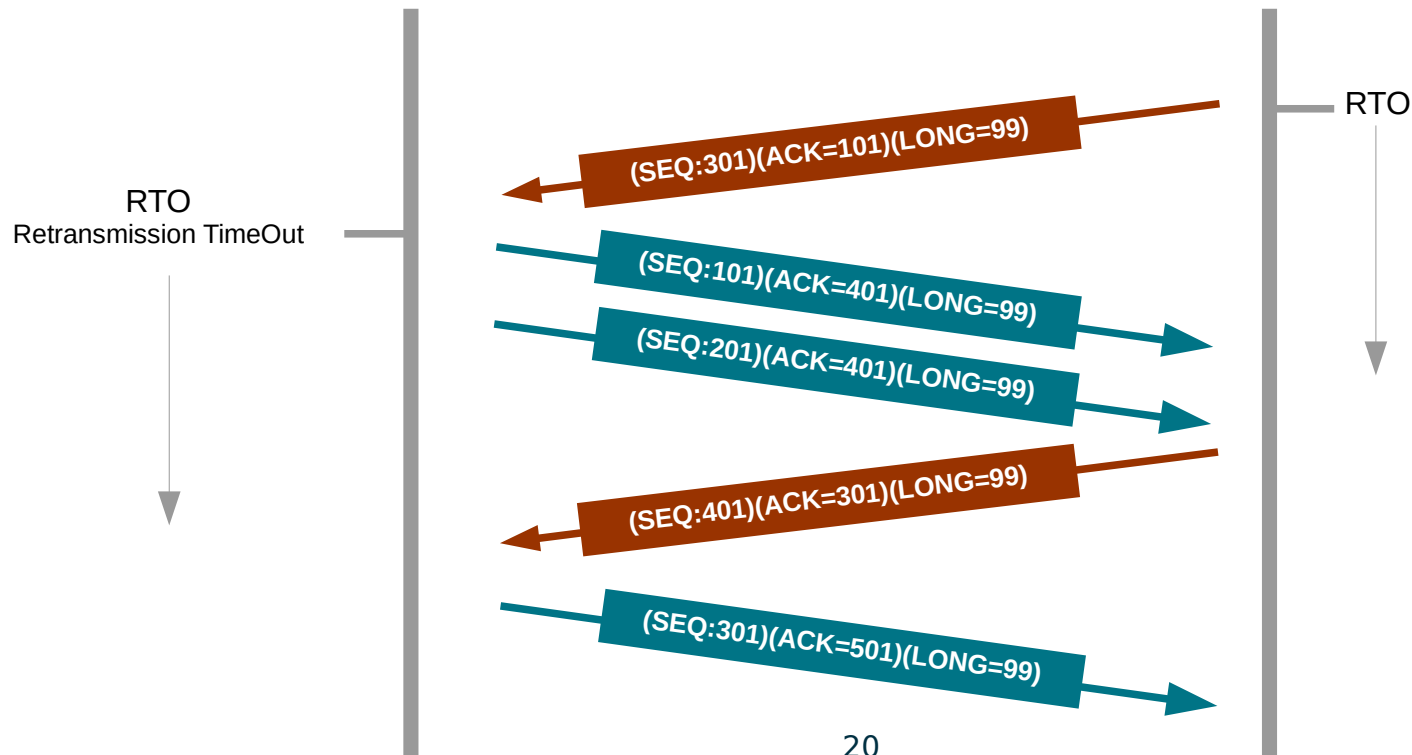
### Problemas en el establecimiento de una conexión

- Un SYN respondido con RST, busque problemas en el servidor
- Triple SYN sin respuesta, problema con un firewall o una aplicación que no responde

# Protocolo TCP

## Especificación funcional

Una vez establecida la comunicación, los datos se transmiten mediante el intercambio de segmentos



# Protocolo TCP

## Especificación funcional

Una conexión se cierra, cuando un usuario llama a la función CLOSE, puede continuar recibiendo, hasta que el otro extremo cierre con otro CLOSE

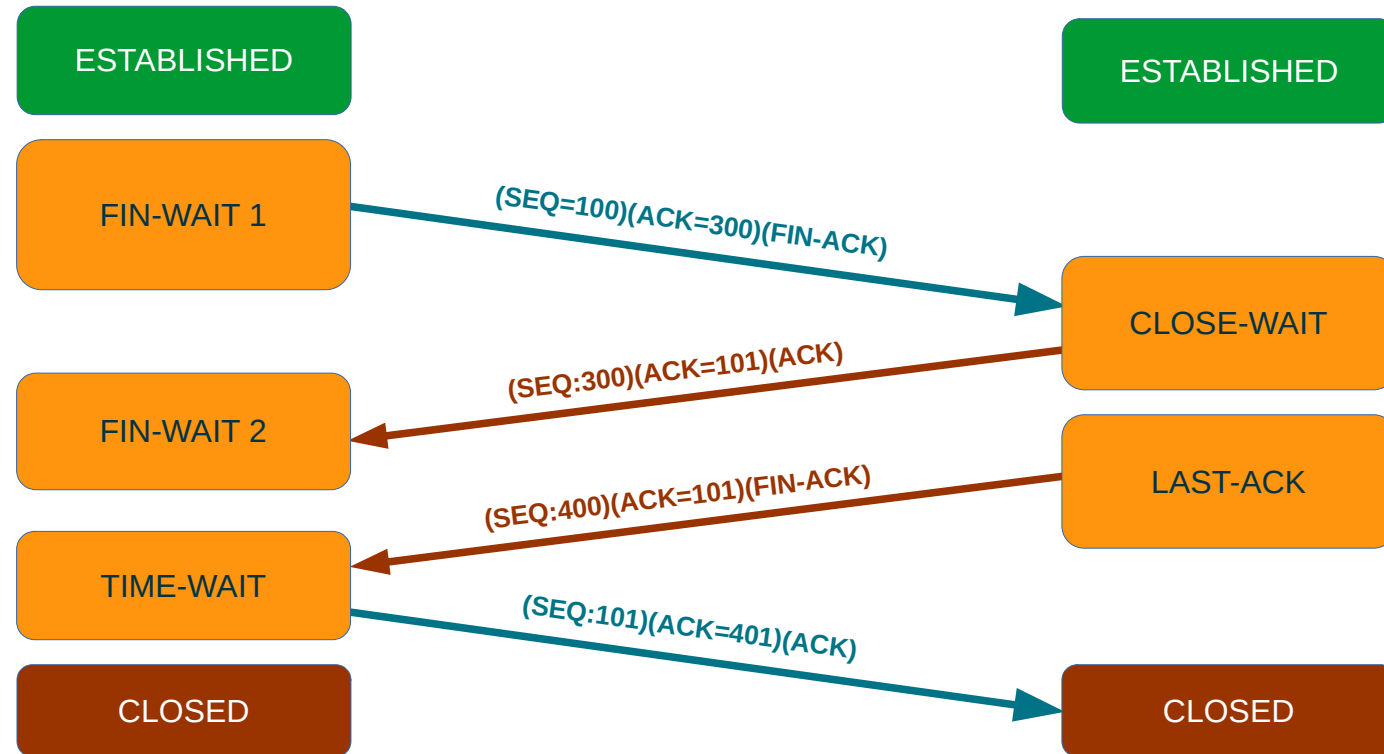
Existen tres casos

- El usuario inicia el cierre de la conexión con CLOSE
- El TCP remoto inicia el cierre enviando un FIN
- Ambos usuarios cierran simultáneamente

# Protocolo TCP

## Especificación funcional

### Cierre normal de una conexión



# Protocolo TCP

TCP y Wireshark



# Protocolo TCP

## TCP y Wireshark

### Parámetros del protocolo TCP

- Los números de secuencia, son elegidos por el proceso y son difíciles de seguir (el estándar TCP no establece ninguna regla de elección)
- El Timestamp, intenta resolver la sensibilidad de TCP a las variaciones del retardo. El rfc1323, adopta la solución
  - El remitente, pone un timestamp en cada segmento que envía
  - El receptor, refleja estas marcas en segmentos ACK



# Protocolo TCP

Control de flujo



# Protocolo TCP

## Control de flujo

El mecanismo de control de flujo, establece durante la conexión el espacio de recepción disponible y durante la comunicación, anuncia los datos que puede aceptar en ese momento (ventana de recepción)

- Si llegan más datos de los que pueden ser aceptados, serán descartados
- Cuando el receptor tiene una ventana de tamaño 0 y llega un segmento, debe enviar un ACK con su próximo número de secuencia y su tamaño de ventana actual

# Protocolo TCP

## Control de flujo

Ventana de emisión



Ventana de recepción



Ventana de emisión



ACK 1001

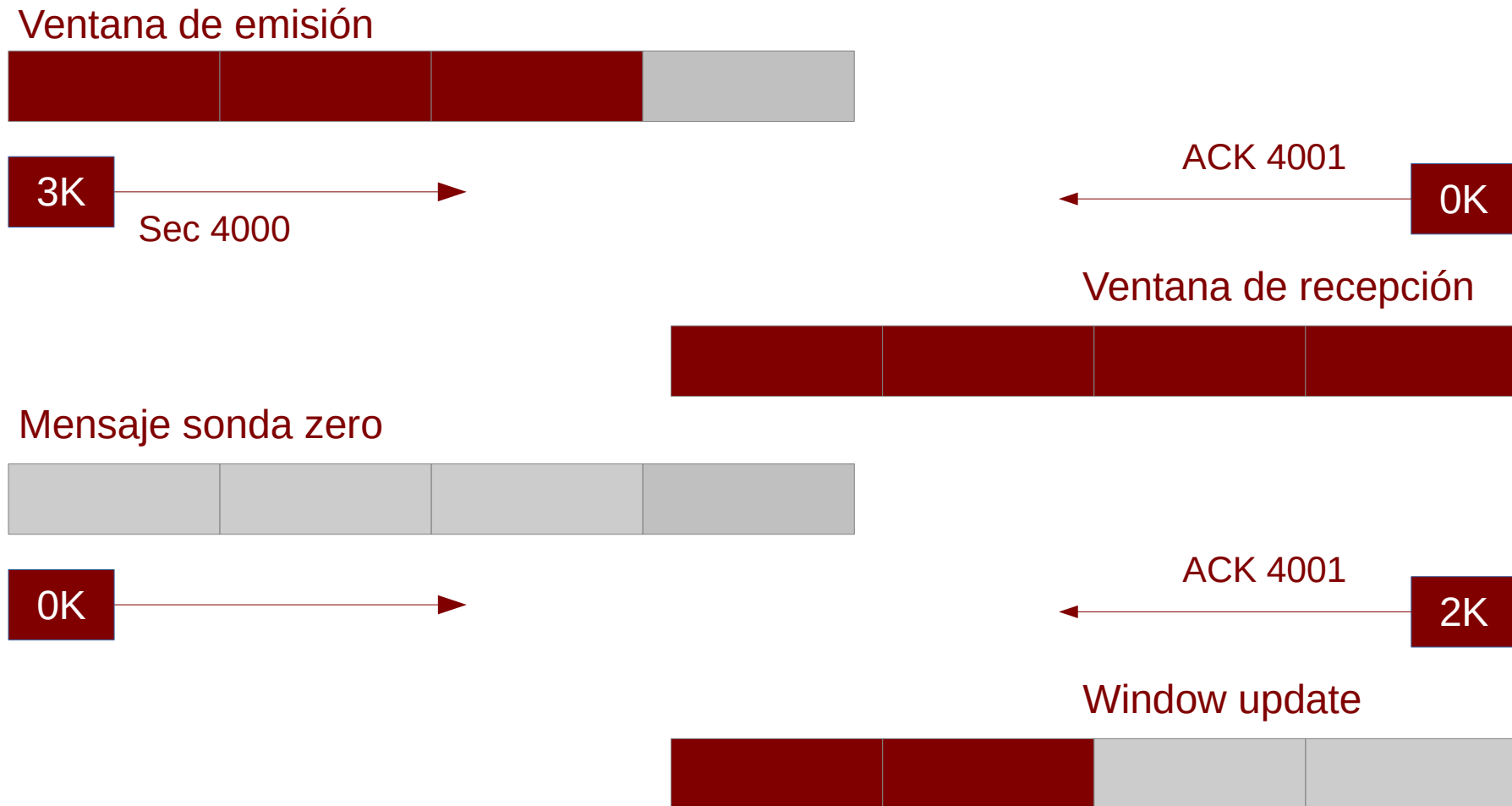


Ventana de recepción



# Protocolo TCP

## Control de flujo



# Protocolo TCP

## Control de flujo

Disponer de una ventana pequeña, provoca que los datos sean transmitidos en segmentos pequeños

Para mejorar la transmisión, hay clientes que realizan ciertas técnicas

- Que el receptor retrase la actualización hasta que el tamaño de la ventana sea de al menos el 20% o 40% del máximo posible
- Que el emisor evite el envío de mensajes pequeños hasta que la ventana sea lo bastante grande para enviar datos

# Protocolo TCP

## Control de flujo

**TCP ventana zero**, se produce cuando un receptor, anuncia un tamaño de ventana 0, que le indica al remitente que deje de enviar datos

- Puede ser debido a un equipo con problemas de memoria o capacidad de proceso
- Una aplicación que consume demasiada memoria

# Protocolo TCP

## Control de flujo

Mensaje **TCP sonda zero**, es enviado por el emisor para ver si todavía existe condición de ventana zero en recepción, si sigue igual, el emisor, doble el temporizador antes de enviar una sonda de nuevo

- Podemos utilizar gráficos TCP para ver problemas de rendimiento

# Protocolo TCP

## Control de flujo

El tamaño de ventana es de 16 bits, lo que permite anunciar un tamaño máximo de 65535 bytes

Se puede calcular el throughput para un tamaño de ventana TCP con la fórmula

$$\text{Throughput} = \frac{\text{TCP WindowSize}}{\text{RTT}}$$

Para una ventana de 65535 bytes en una ruta con RTT de 100 ms, el throughput, es de 5,24 Mbps



# Protocolo TCP

## Control de flujo

Para que tamaños de ventana más grandes alojen rutas de alta velocidad, la especificación **rfc1323**, define en las opciones de cabecera TCP, un ajuste de escala

- **Window size**, indica un multiplicador con el tamaño de ventana, para notificar el receptor un bufer de mayor tamaño
- Esta opción, solo se envía en segmentos SYN durante el proceso de conexión

# Protocolo TCP

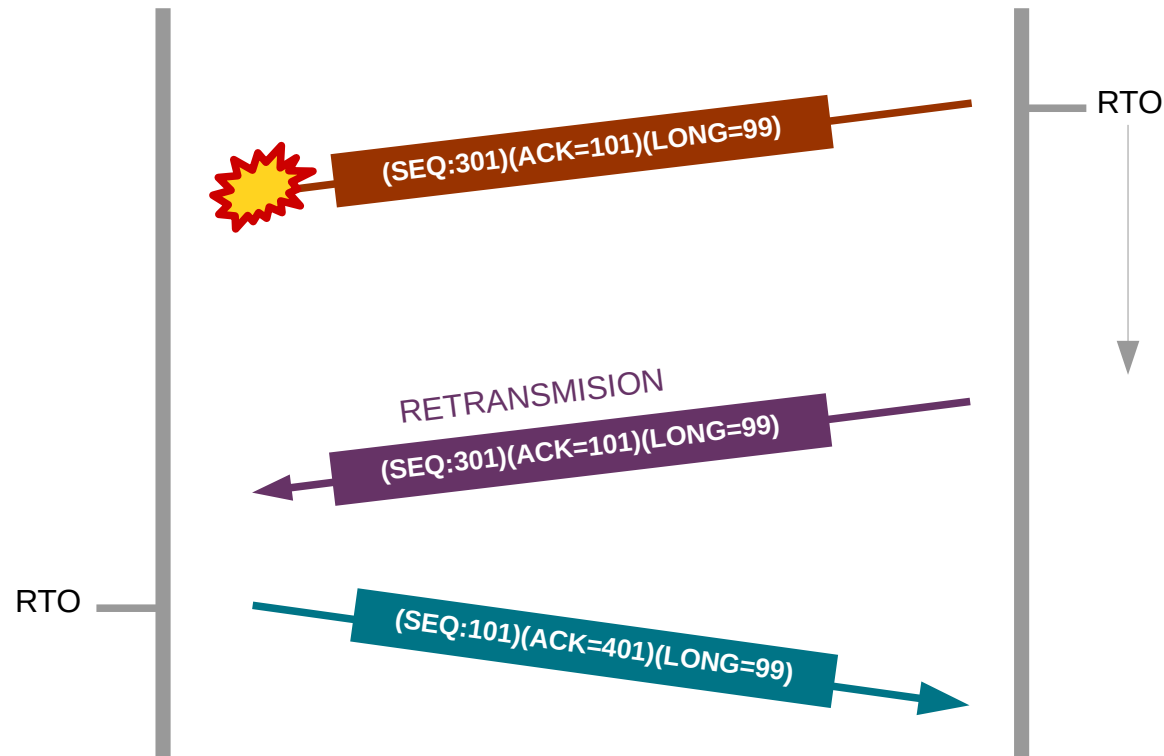
Retransmisiones



# Protocolo TCP

## Retransmisiones

TCP utiliza la retransmisión (tras un tiempo de espera) para asegurar la entrega de cada segmento



# Protocolo TCP

## Retransmisiones

Control de errores, retransmisiones. Un exceso de retransmisiones puede volver lenta la red

- Las retransmisiones ocurren cuando un paquete no ha llegado o un reconocimiento no ha llegado a tiempo
- La primera retransmisión, se produce al expirar un temporizador (RTO), las siguientes, duplican el tiempo hasta alcanzar un máximo de 5 retransmisiones
- Si se recibe 3 ACKs duplicados, no hace falta que expire RTO

# Protocolo TCP

## Retransmisiones

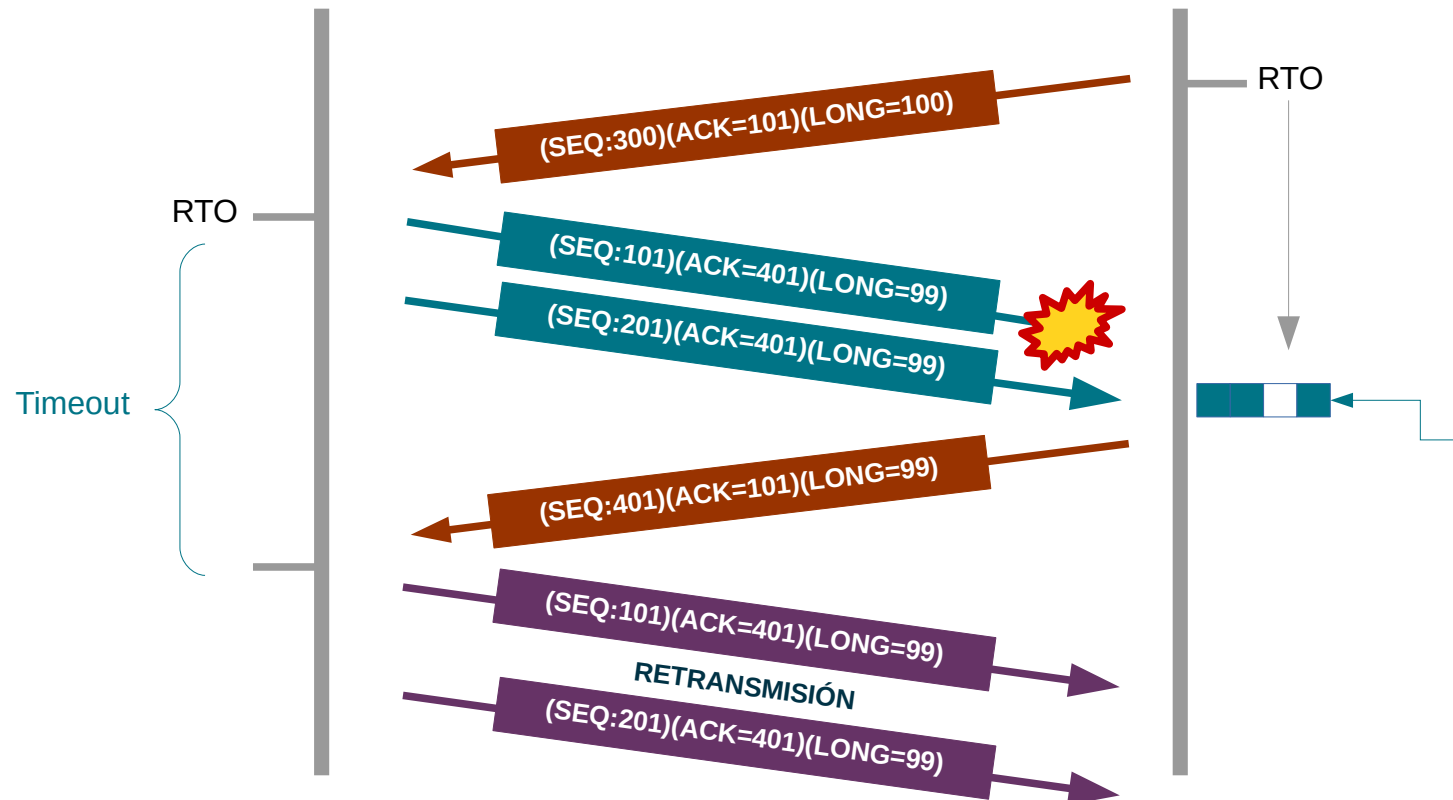
La elección del tiempo de vencimiento del temporizador de retransmisión, está basada en los retardos observados en la red

Los retardos, pueden variar dinámicamente, por tanto, los temporizadores, deben adaptarse a esta situación y se recalculan mediante diversos algoritmos

# Protocolo TCP

## Retransmisiones

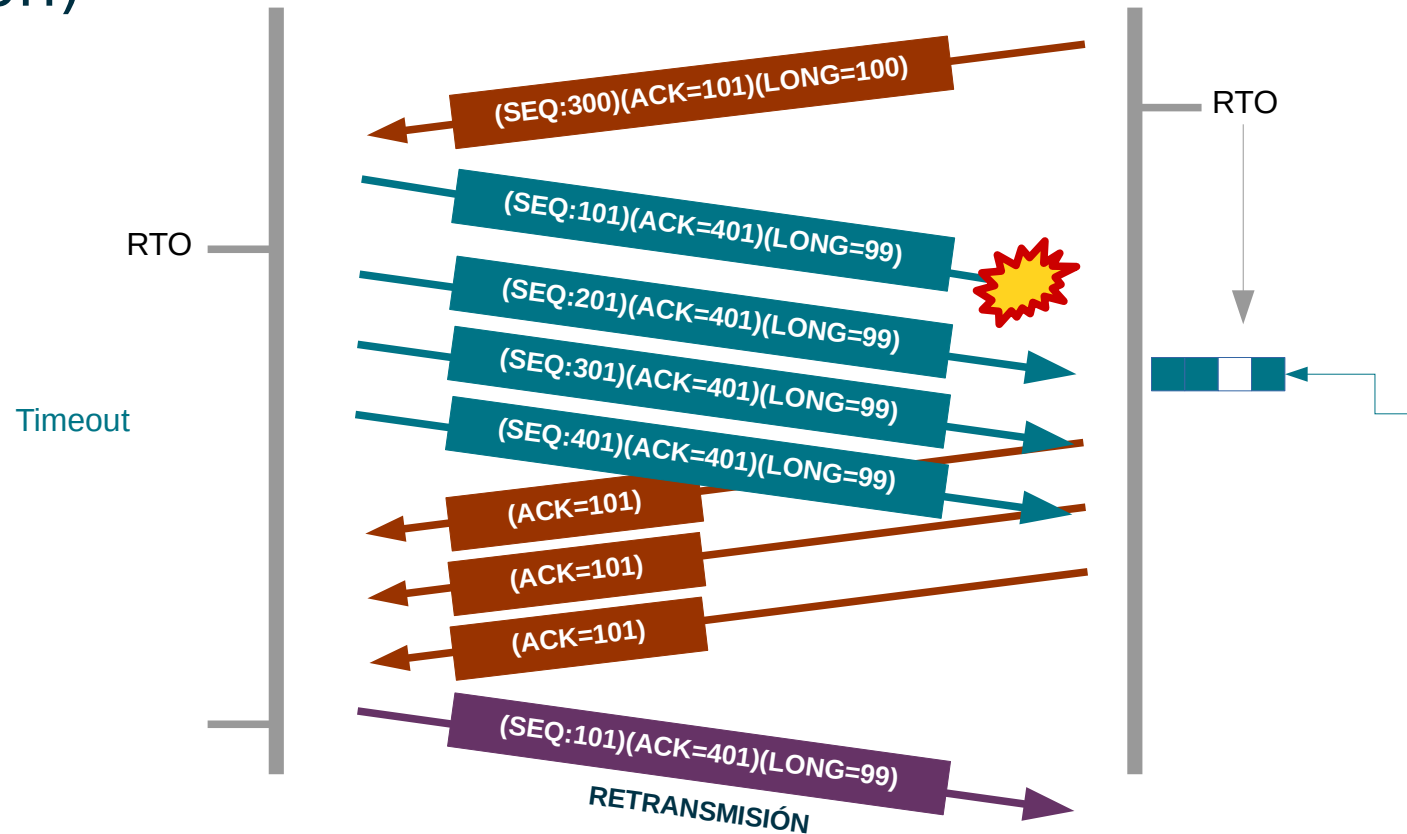
### Perdida de un segmento y vencimiento de RTO



# Protocolo TCP

## Retransmisiones

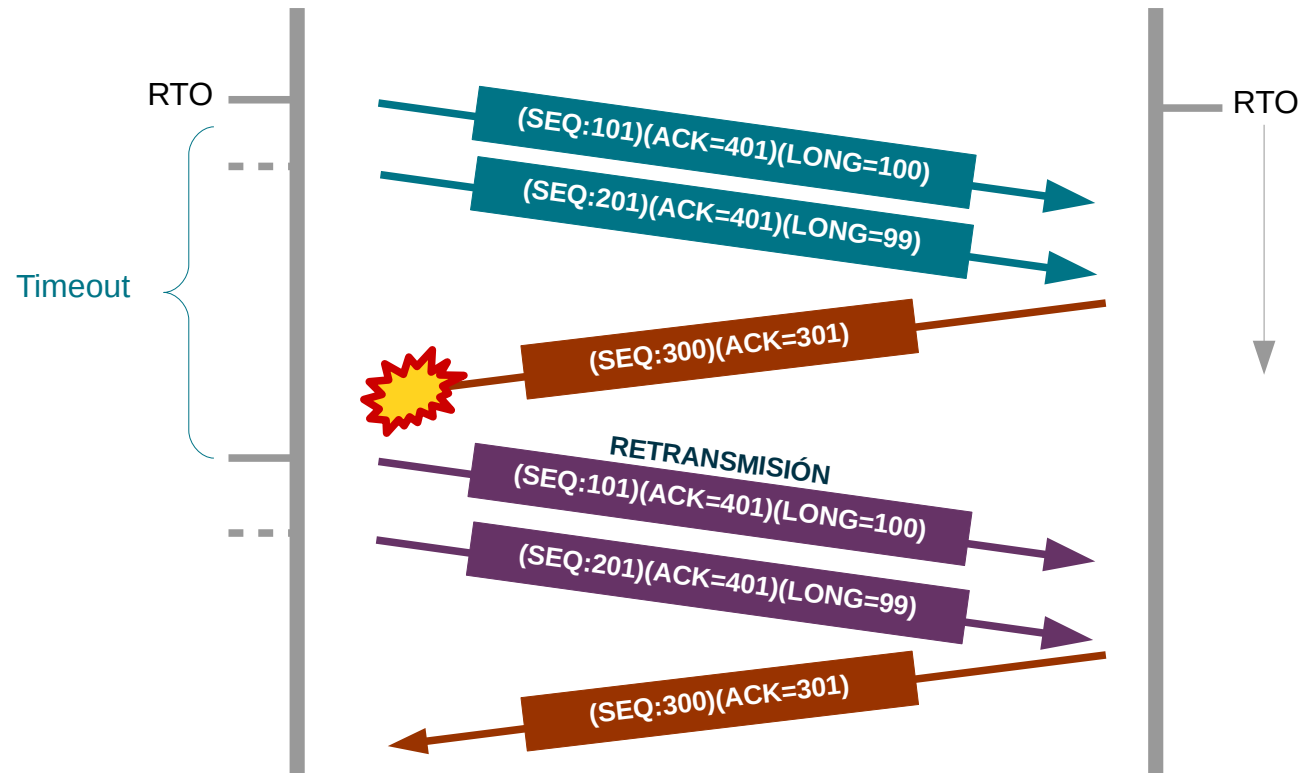
Perdida de un segmento y recepción de 3 ACKs seguidos (Fast retransmission)



# Protocolo TCP

## Retransmisiones

### Perdida de un ACK y vencimiento de RTO





# Protocolo TCP

## Retransmisiones

Después de 5 retransmisiones seguidas, la conexión se considera perdida

- Se envía un SYN intentando establecer una nueva conexión
- No se envía SYN y el usuario, deberá volver a correr la aplicación

# Protocolo TCP

## Retransmisiones

En consecuencia, las retransmisiones son un comportamiento natural, siempre que no hayan demasiadas

- Ubicar el problema en la dirección IP, conexión o aplicación concreta
- Comprobar si es debido a la pérdida de paquetes, a un servidor o a una aplicación lenta
- Comprobar si hay variaciones en el retardo

# Protocollo TCP

Selective ACK



# Protocolo TCP

## Selective ACK

El diseño original del proceso de retransmisiones es ineficiente, aunque solo se pierda un paquete, es necesario retransmitir los siguientes

En las opciones de la cabecera TCP, puede haber la siguiente opción

- SACK (selective ACK), permite reconocer paquetes específicos, ambas partes tienen que ponerse de acuerdo

# Protocolo TCP

## Selective ACK

TCP SACK, usa los reconocimientos para que el emisor, tome decisiones inteligentes respecto a las retransmisiones. Utiliza dos opciones

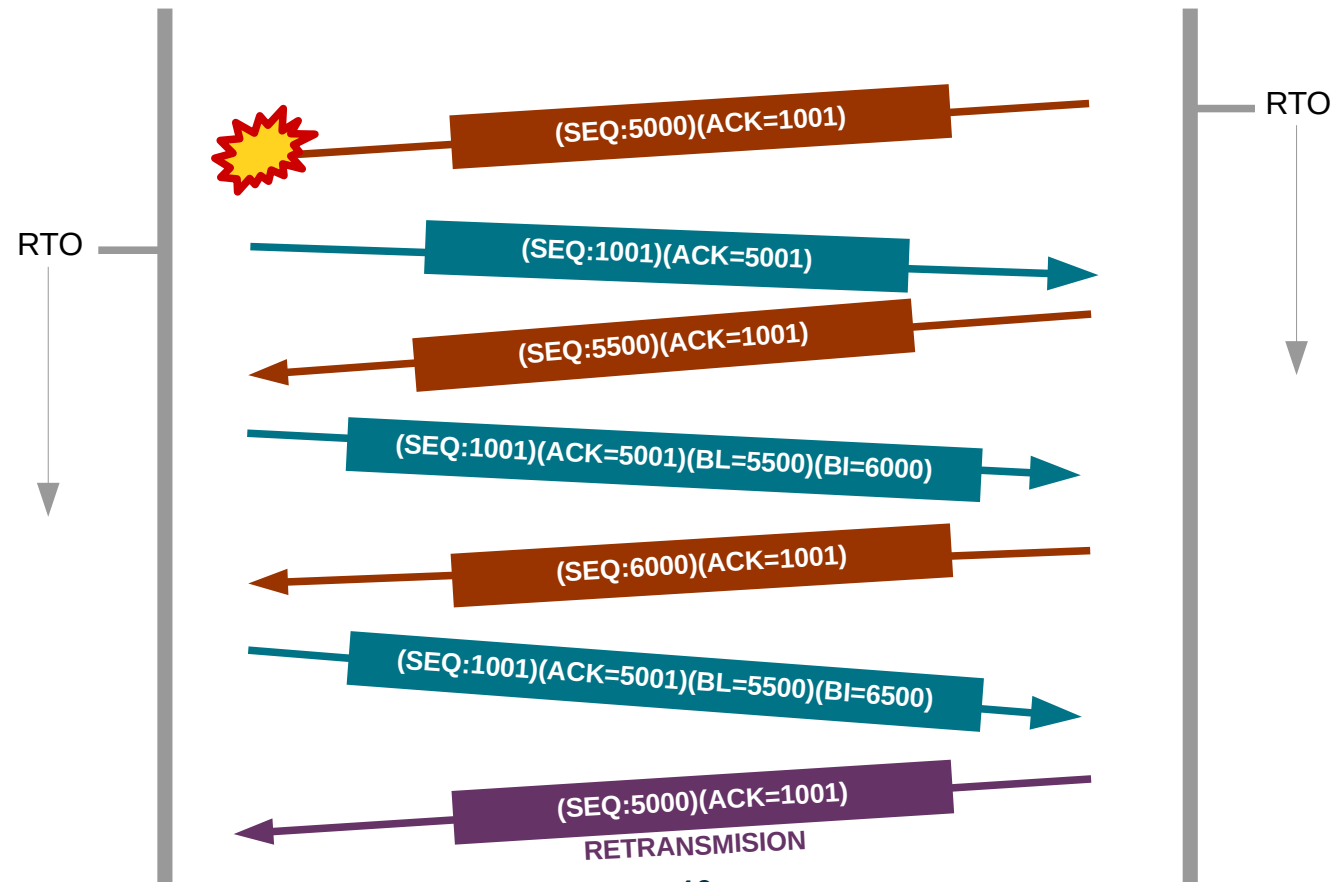
- **SACK** permitted, utilizada para habilitar la opción SACK en el establecimiento de la conexión
- SACK, utilizado para informar de los bloques de datos recibidos que no son contiguos



# Protocolo TCP

## Selective ACK

### Funcionamiento de SACK



# Protocolo TCP

ACKs duplicados



# Protocolo TCP

## ACKs duplicados

Cuando tenemos variaciones en el retardo, también se pueden esperar retransmisiones

Una demora, puede ocurrir debido a

- La inestabilidad de una línea
- Una aplicación saturada o ineficiente
- Un equipo sobrecargado



# Protocolo TCP

## ACKs duplicados

En la mayoría de los casos, ACKs duplicados, se deben a la alta latencia, las variaciones del retardo o un host lento

- Uno o dos ACKs duplicados, implican desorden en la recepción
- Tres ACKs duplicados, implica la pérdida del paquete y el emisor, vuelve a enviar el paquete (retransmisión rápida)

# Protocolo TCP

Latencia y retardos



# Protocolo TCP

## Latencia y retardos

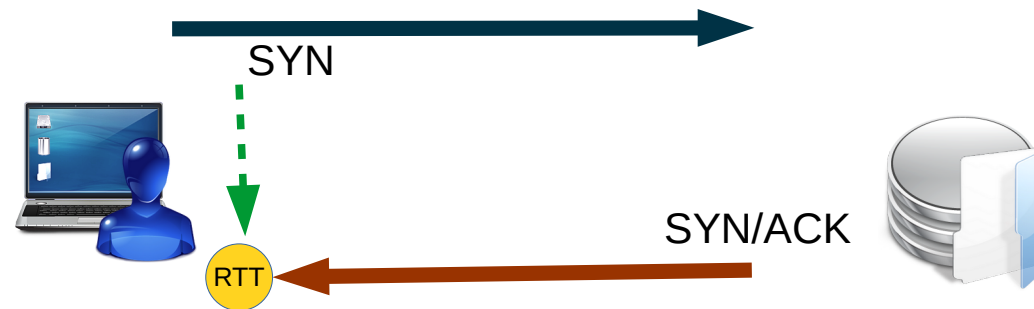
La latencia, es una medida utilizada para definir retardo de tiempo, esta, puede ser causada por

- Problemas a lo largo de una ruta
- Problemas en el cliente
- Problemas en el servidor

# Protocolo TCP

## Latencia y retardos

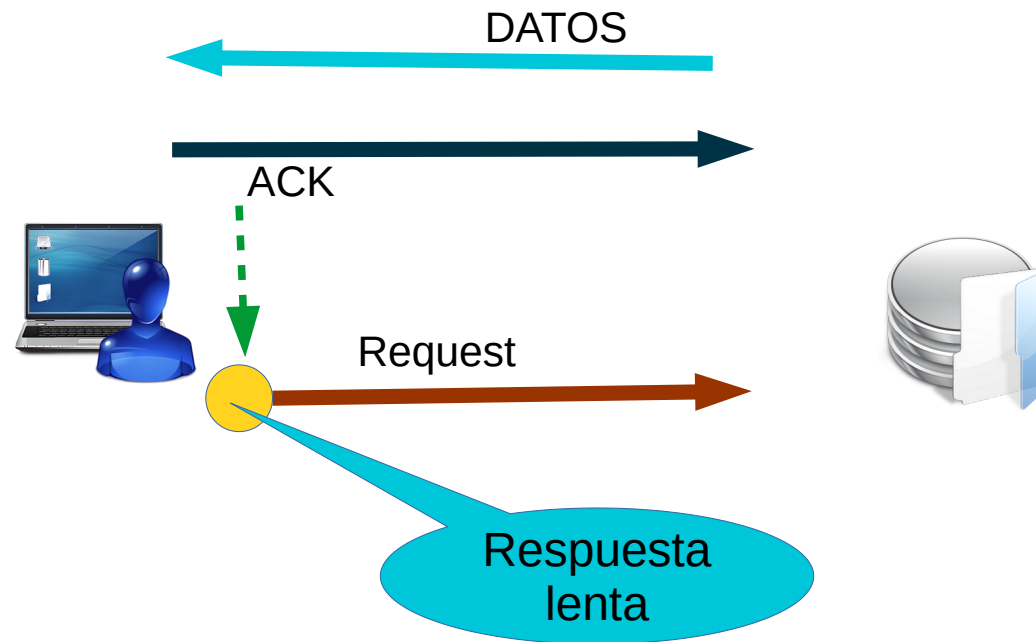
La latencia en ruta, se conoce como **Round Trip Time** (RTT, tiempo de ida y vuelta)



# Protocolo TCP

## Latencia y retardos

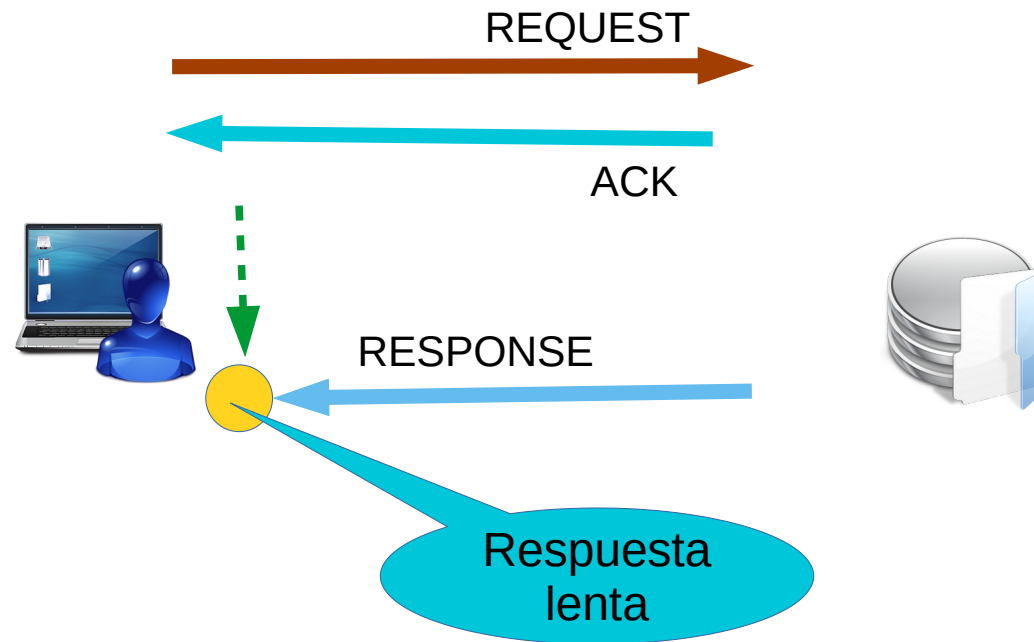
La latencia en ruta, se conoce como **Round Trip Time** (RTT, tiempo de ida y vuelta)



# Protocolo TCP

## Latencia y retardos

La latencia de servidor, se produce cuando tarda en responder a la recepción de una solicitud



# Protocolo TCP

## Latencia y retardos

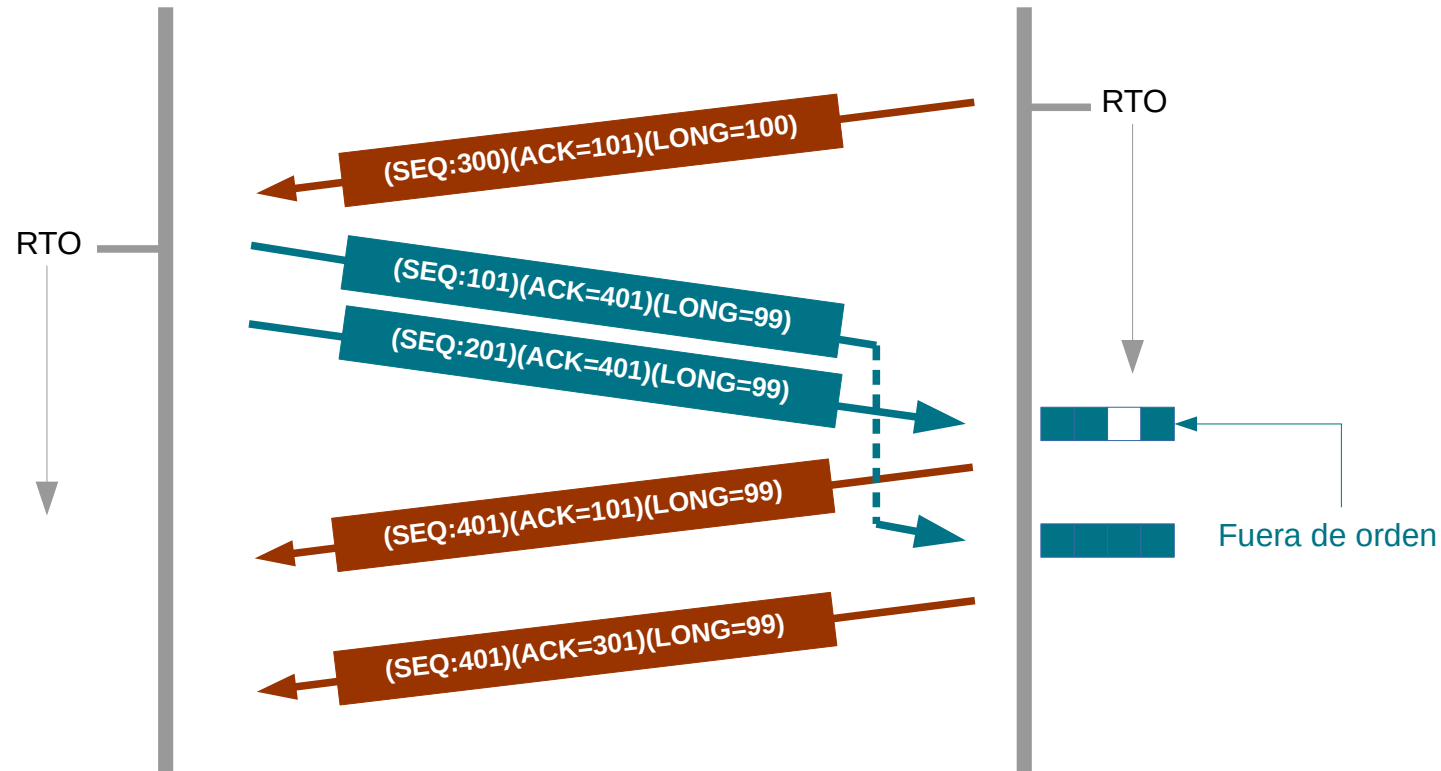
Podemos encontrarnos perdidas de paquetes y segmentos fuera de orden

- Puede ser debido a diferentes rutas de origen al destino o a perdida de paquetes
- Puede ser que Wireshark, no las haya capturado
  - Tráfico muy pesado
  - La máquina desde la que se toman las capturas no es muy potente
  - Los switchs pueden descartar paquetes
  - Captura de datos en una red inalámbrica

# Protocolo TCP

Latencia y retardos

Recepción fuera de orden





# Protocolo TCP

## Latencia y retardos

### Los motivos para perder un paquete

- Congestión en un router
- Error en una línea
- Error en un paquete

# Protocolo TCP

## Latencia y retardos

El ajuste por defecto de la columna Time, es desde el comienzo de la captura y en segundos

Para seleccionar el tiempo desde el final de un paquete hasta el final del siguiente, seleccione

- **View → Time Display Format → Seconds since previous displayed packet**

# Protocolo TCP

## Latencia y retardos

Detectar la latencia con una columna **TCP Delta**, una vez que se ha habilitado la opción “**Calculate Conversation Timestamp**”

Creamos una columna desde el parámetro “**Time since previous frame**” de TCP

# Protocolo TCP

## Latencia y retardos

¿Qué retrasos se pueden considerar como normales?

- Solicitudes de archivos “ico”
- Paquetes SYN
- Paquetes FIN, FIN-ACK, RST
- Paquetes GET
- Consultas DNS

No.	Time	TCP Delta ▲	Source	Destination	Protocol	Info
471	20.675453000	18.305302000	24.6.173.220	173.194.79.82	HTTP	GET /svn/trunk/image/p-expand.gif HTTP/1.1
470	20.675046000	18.303400000	24.6.173.220	173.194.79.82	HTTP	GET /svn/trunk/image/16x16/User.png HTTP/1.1
458	20.611551000	18.229698000	24.6.173.220	173.194.79.82	HTTP	GET /svn/trunk/image/throbber.gif HTTP/1.1

# Protocolo UDP

## Introducción



# Protocolo UDP

## Introducción

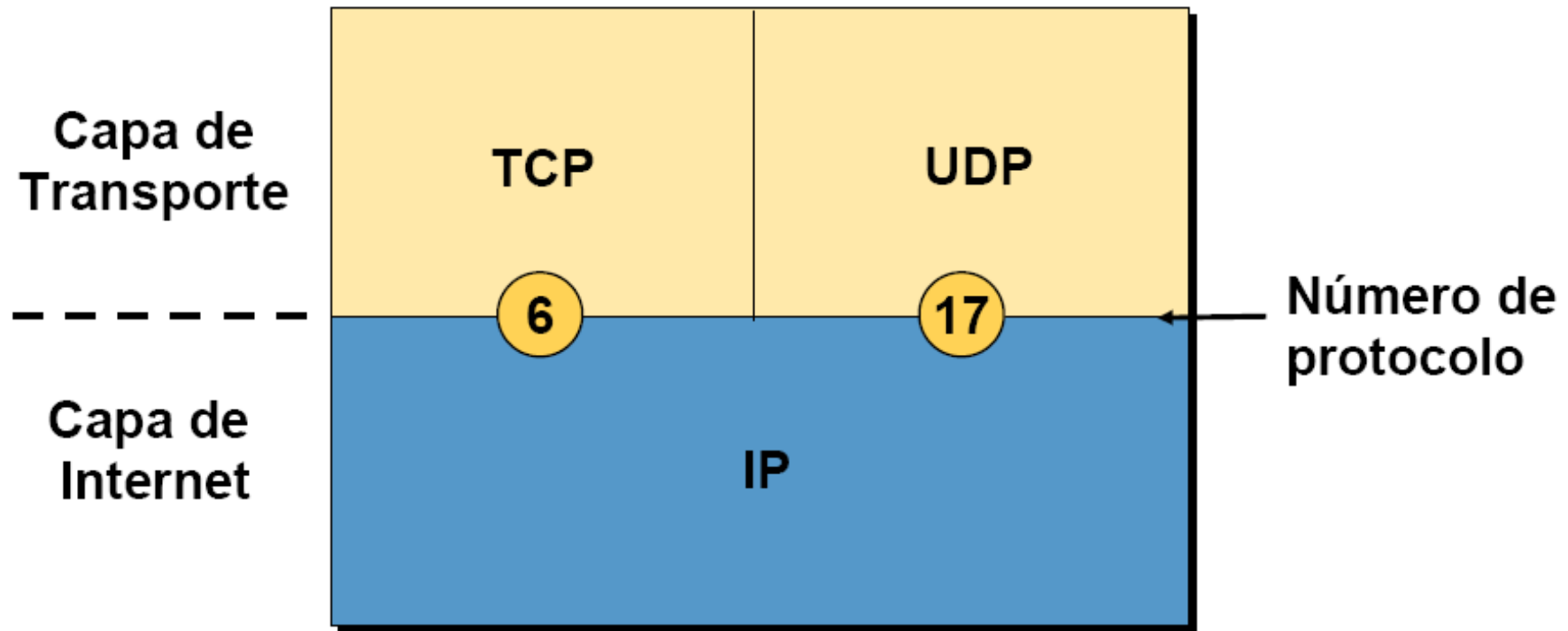
UDP aporta un procedimiento para que las aplicaciones, puedan enviar mensajes, con un mecanismo mínimo de control, buscando evitar la sobrecarga creada por el control de la comunicación

- Consultas a bases de datos
- Aplicaciones en tiempo real
- Aplicaciones de monitorización

# Protocolo UDP

## Introducción

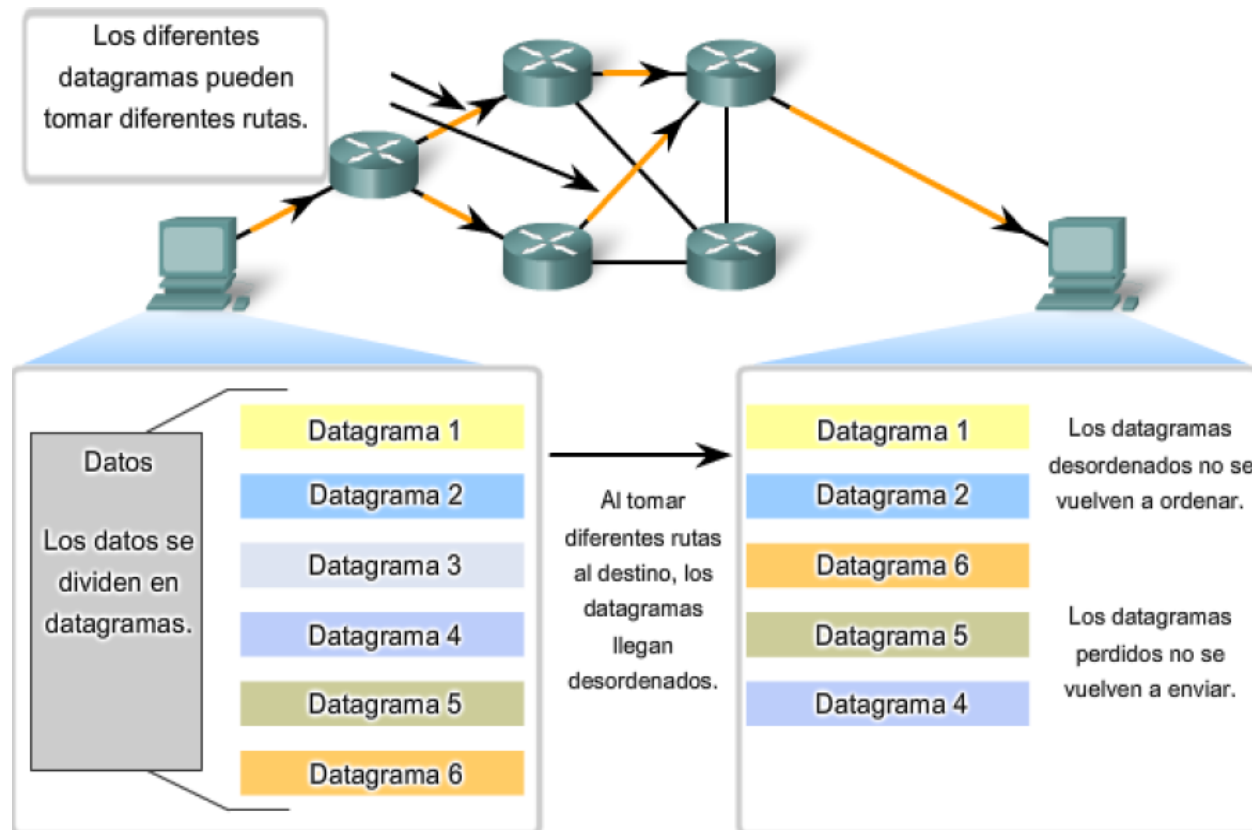
Como reconoce IP a los protocolos de transporte



# Protocolo UDP

## Introducción

La entrega de datagramas, puede ser desordenada





# Protocolo UDP

## Descripción del protocolo

UDP crea un mensaje añadiendo una cabecera simple a los datos, contiene puerto de origen y destino para enviar el mensaje al destinatario correcto

Puerto origen (16)	Puerto destino (16)
Longitud (16)	Checksum (16)
Datos (variable)	

# Protocolo UDP

## Descripción del protocolo

El campo suma de control, se calcula sobre una combinación de una pseudocabecera, construida especialmente con cierta información de la cabecera IP y UDP

Dirección IP origen		
Dirección IP destino		
0	17	Longitud UDP

# Protocolo UDP

## Descripción del protocolo

Un servicio UDP, no tiene forma de predecir o controlar cuantos datagramas se enviarán en un momento dado, si se bombardea el servicio con más datagramas de los que puede manejar, se descartan

*Telefónica*

---