Protocolos de Transporte: UDP, TCP

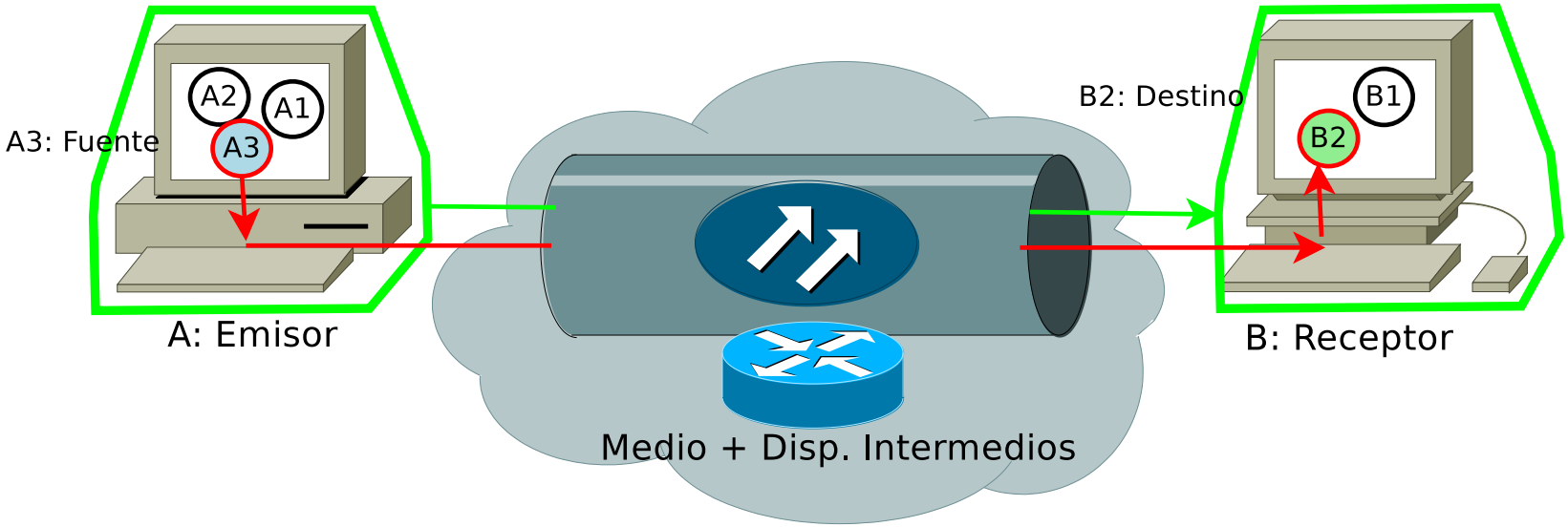
(parte I) 2020

UNIVERSIDAD
NACIONAL
DE LA PLATA

Introducción

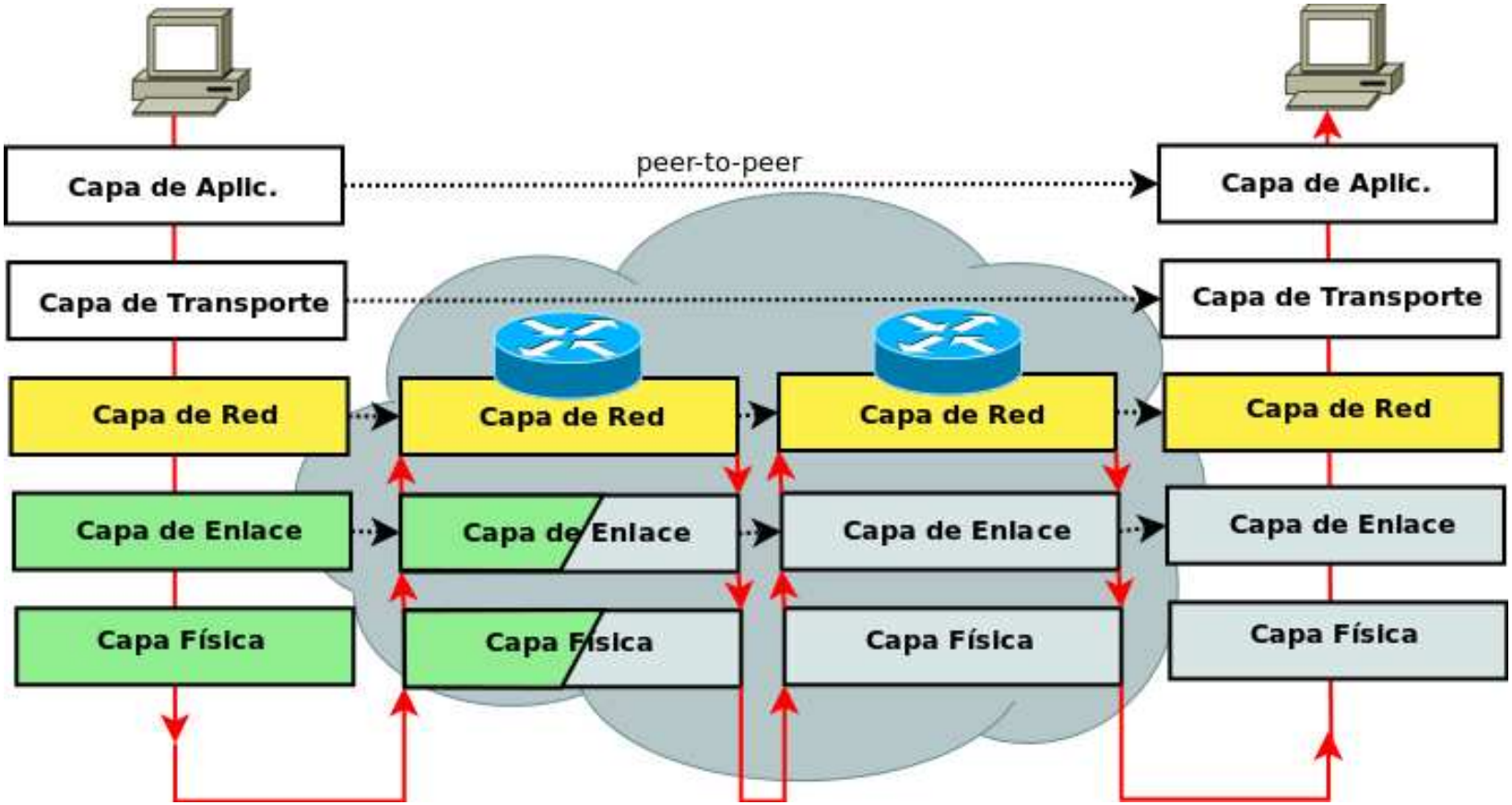
- IP provee un servicio “débil”, pero eficiente: “Best-effort”.
- En IP los paquetes pueden ser descartados, des-ordenados, retardados duplicados o corrompidos.
- Paquetes IP solo dirección DST y SRC, ¿Como elegir la App.?
- IP corre en **todos los nodos de la red** (routers y host), protocolos de transporte solo necesario en **end-points** (hosts).
- IP: comunicación lógica HOP-BY-HOP, comunica **hosts**.
- Transporte: comunicación lógica **HOST-TO-HOST (END-TO-END)**, comunica **procesos**.
- Aplicación: comunicación lógica **PROCESS-TO-PROCESS (END-TO-END)** comunica usuarios, agentes, etc.

Direccionamiento a nivel de Transporte



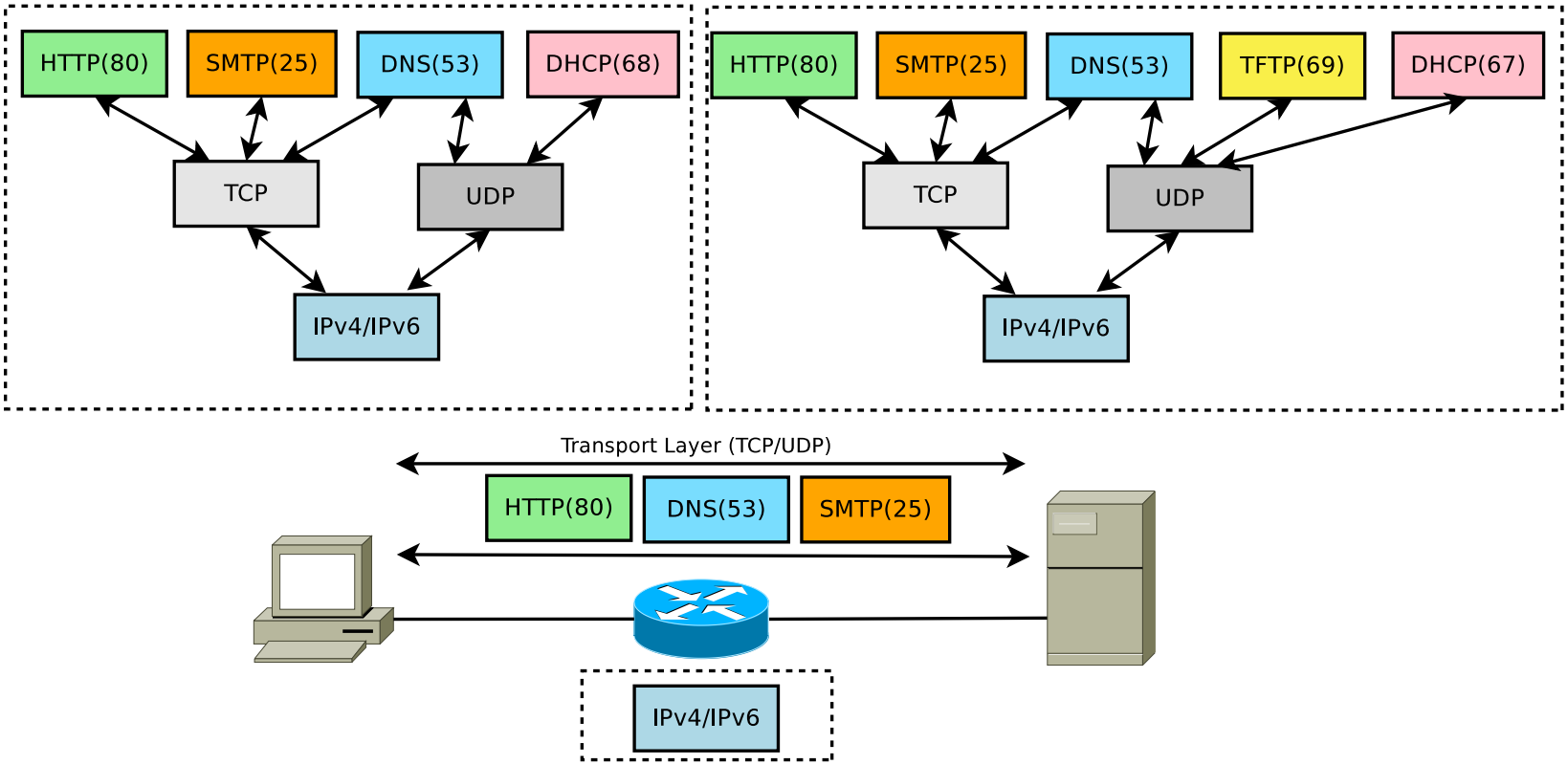
- Servicio Internet, IP, puerta a puerta, host-to-host.
- Servicio Transporte, persona a persona, process-to-process.
- Identificador de proceso independiente de plataforma, nro. de puerto.

Comunicación en Capas

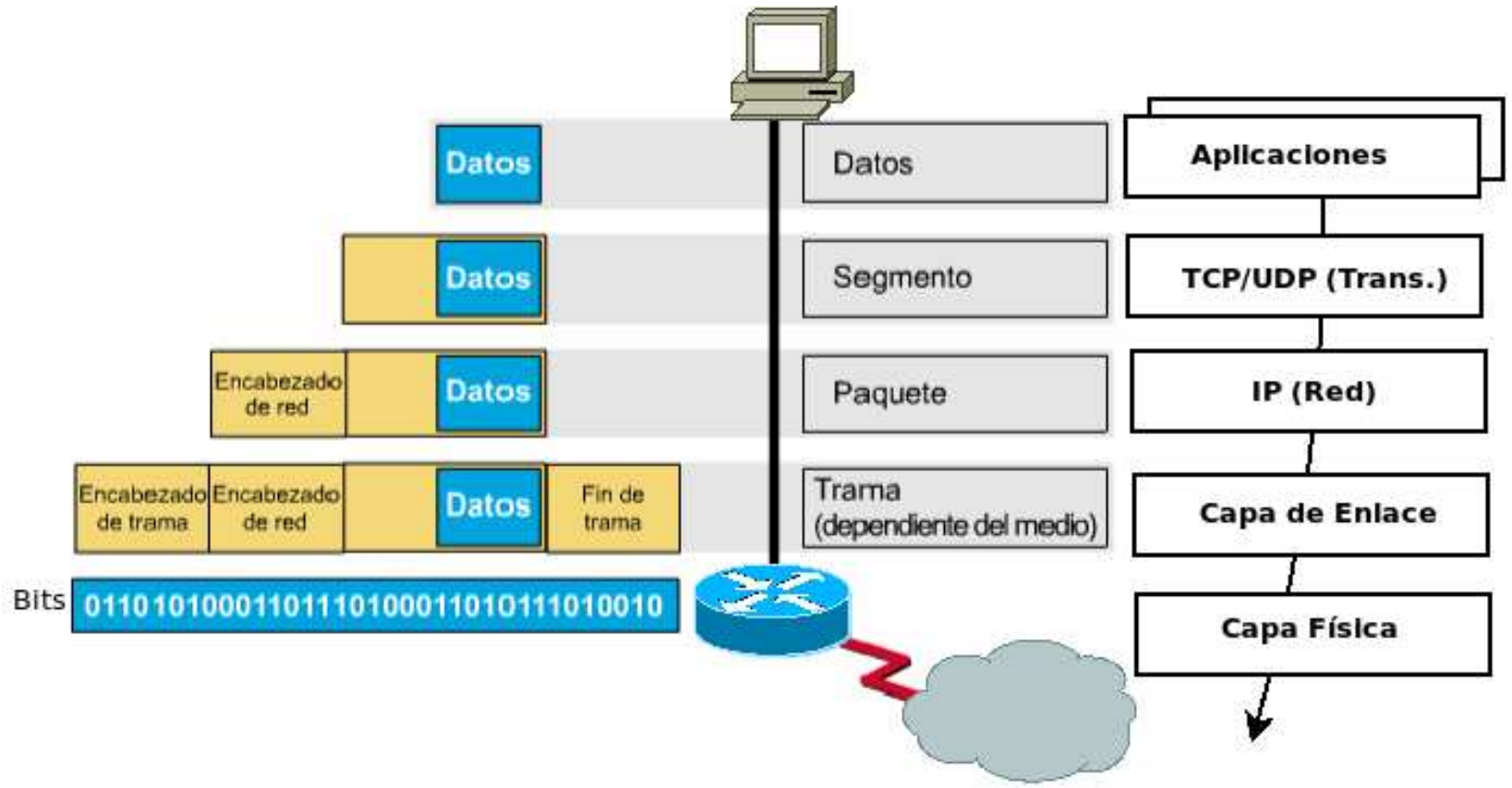


Funcionalidad de Capa de Transporte

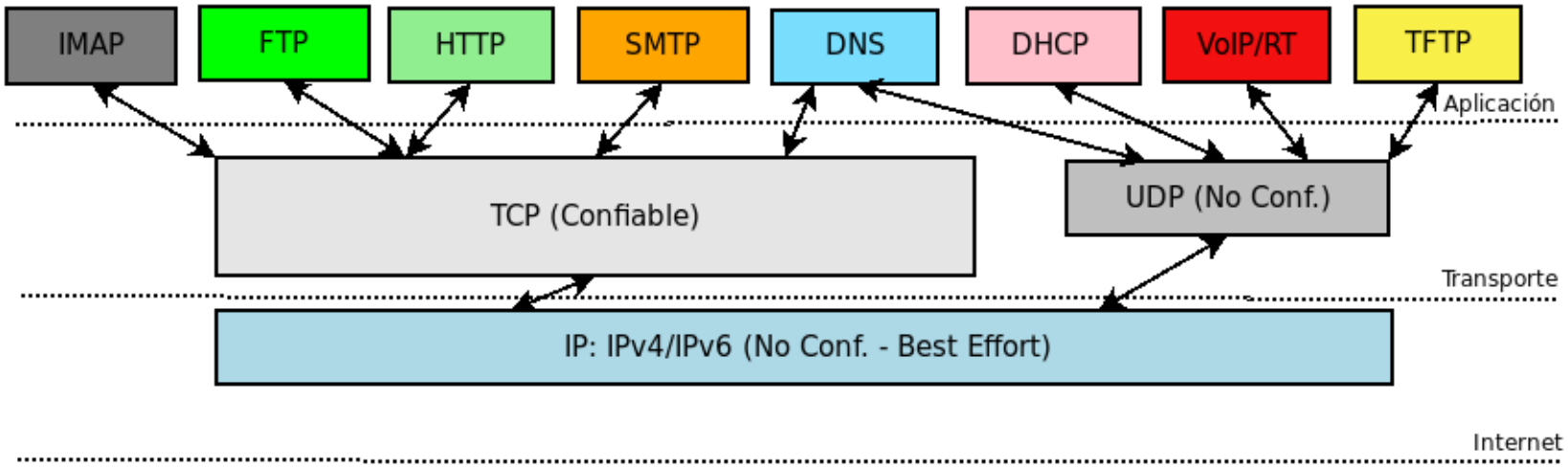
- MUX/DEMUX App. to App. (puertos, Ports).
- Soporte de datos de tamaños arbitrarios.
- Control y Detección de Errores, pérdida, duplicación, se corrompen.
- ¿Cómo enviar info sobre la red de acuerdo al estado de la misma? ¿Cuándo y Cómo una App. debe enviar datos?
  - Control de Flujo.
  - Control de Congestión.
- Dos modelos:
  - Modelo Confiable: TCP.
  - Modelo NO Confiable: UDP.


HTTP(80)
SMTP(25)
DNS(53)
DHCP(68)
TCP
UDP
IPv4/IPv6
HTTP(80)
SMTP(25)
DNS(53)
TFTP(69)
DHCP(67)
TCP
UDP
IPv4/IPv6
Transport Layer (TCP/UDP)
HTTP(80)
DNS(53)
SMTP(25)
IPv4/IPv6

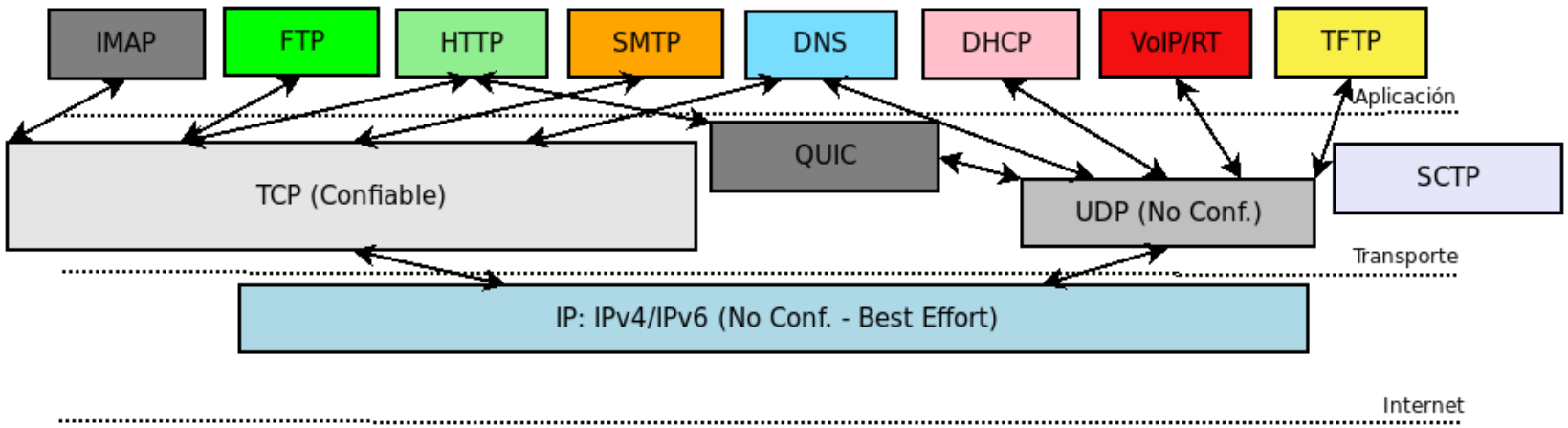
Protocolos de Transporte, Encapsulación



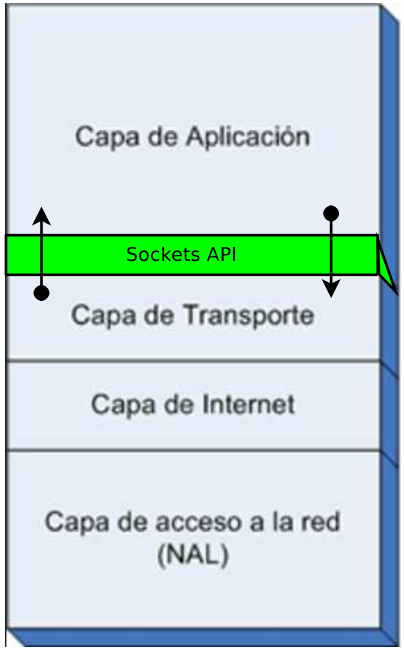
Protocolos de Transporte

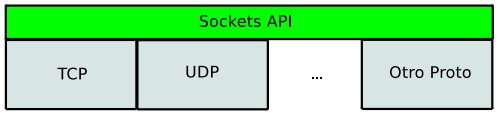


Protocolos de Transporte, Alt.



Selección Protocolo de Transporte





- La aplicación de acuerdo a como esta programada selecciona el transporte.
- El acceso a los servicios de transporte se hace mediante API:**Network socket**.

UDP

- User Datagram Protocol (RFC-768).
- Protocolo Minimalista. Menor Overhead.
- Características de IP: best-effort.
- Orientado a Packets/Datagramas (mensajes auto-contenidos).
- PDU: Datagrama (Por coherencia con nivel Transporte se suele llamar Segmento).
- Solo provee MUX/DEMUX.
- No incrementa Overhead end-to-end.
- No requiere establecimiento de conexión.
- Servicio FDX.
- Aplicaciones: video/voz streaming/TFTP/DNS/Bcast/Mcast.

TCP

- Transport Control Protocol (RFC-793)
- Protocolo confiable, ordenado, buffering, control de flujo y de congestión.
- Orientado a Streams (secuencia de bytes, $\equiv$ archivo).
- PDU: Segmento.
- Provee MUX/DEMUX.
- Incrementa Overhead end-to-end para ofrecer confiabilidad.
- Requiere establecimiento de conexión (y cierre).
- Servicio FDX.
- Aplicaciones: FTP/HTTP/SMTP/acceso remoto(SSH, telnet,...)/Unicast.

SCTP

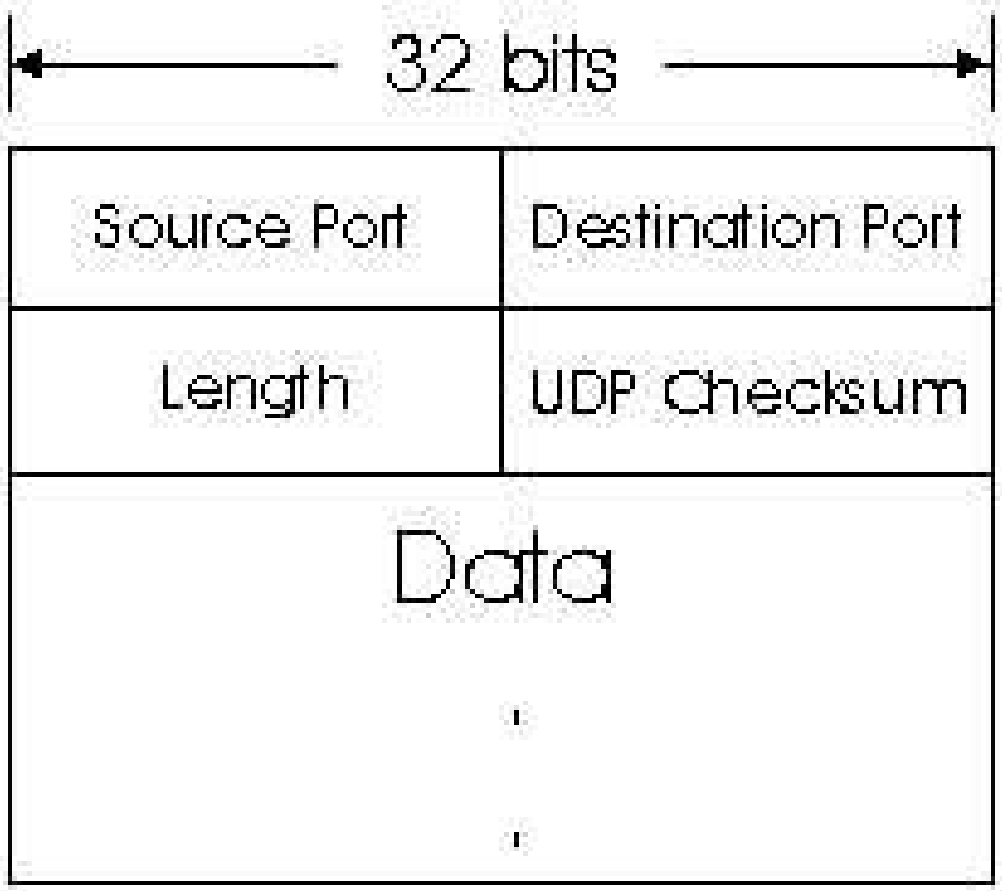
- Stream Control Transmission Protocol (RFC-4960).
- Protocolo con menor Overhead que TCP y más servicios que UDP.
- Orientado a Packets/Datagramas/Mensajes.
- Incrementa Overhead end-to-end.
- Asegura orden, secuencia con control de congestión.
- Servicio FDX.
- Aplicaciones: WebRTC y ssh lo podría utilizar (no es lo más común).

Headers/Encabezados Servicios

- El encabezado IP provee: Ruteo, Fragmentación, Detección de algunos errores.
- El encabezado UDP provee: MUX/DEMUX de aplic. , Detección de errores (no obligatorio).
- El encabezado TCP provee: MUX/DEMUX, Detección de errores, Sesiones, Control de Errores, Control de Flujo y Control de Congestión.

```
? grep udp /etc/protocols
udp     17      UDP # user datagram protocol
? grep tcp /etc/protocols
tcp     6       TCP # transmission control protocol
```

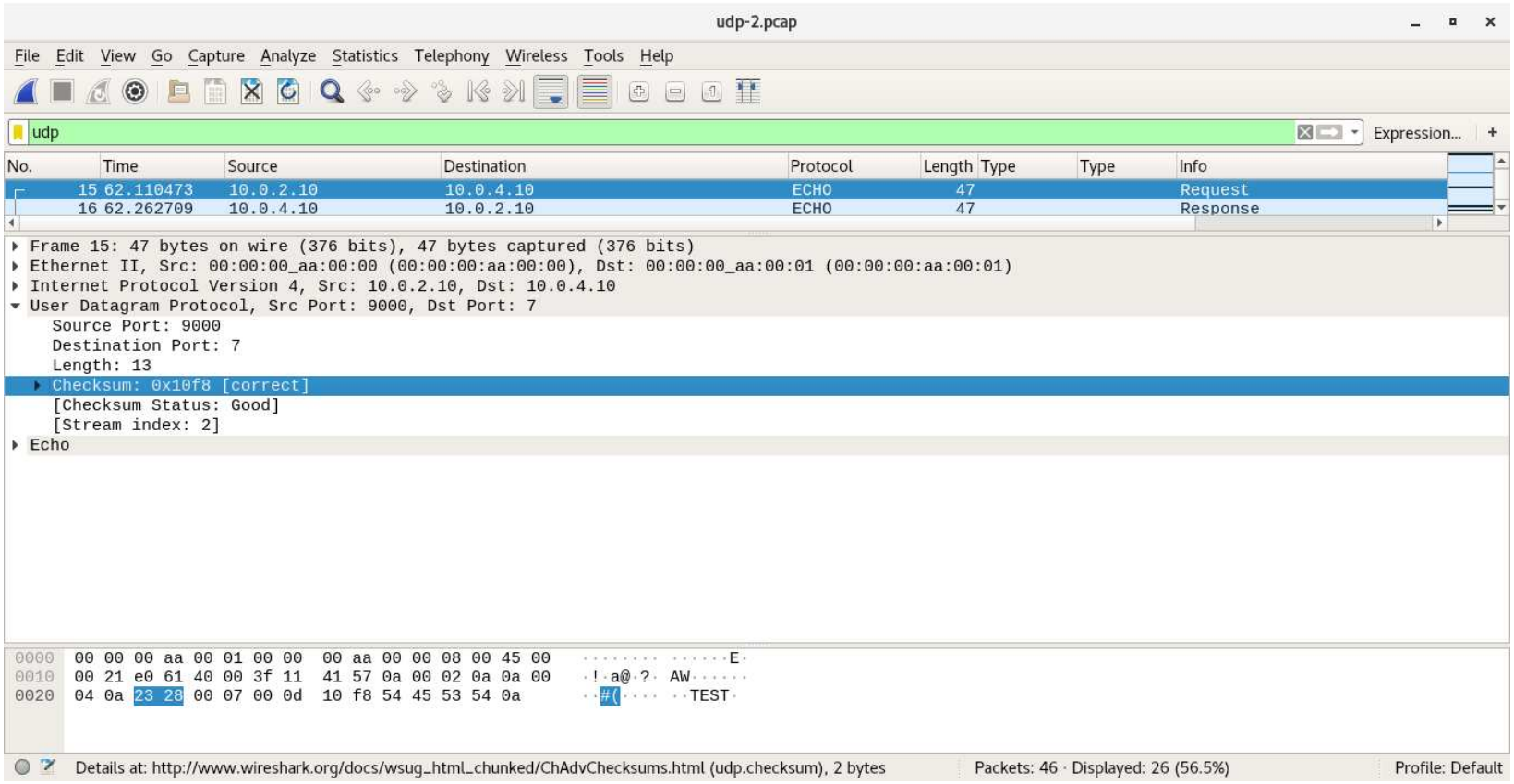
Datagrama UDP



Datagrama (UDP)

- Puertos: MUX/DEMUX.
- Longitud: UDP HDR + Payload.
- Checksum
  - Cálculo Ca1, Opcional. 0 = Sin checksum.
  - Calculado HDR + PseudoHDR + Payload.
  - PseudoHDR: IP.SRC + IP.DST + Zero + IP.PROTO + UDP.LENGTH.
  - PseudoHDR: protección contra paquetes mal enrutados.
  - Aplicaciones de LAN por eficiencia lo podrían deshabilitar.
  - Si tiene error se descarta silenciosamente.

Cálculo de Checksum



Cálculo de Checksum (Cont.)

```
Pseudo header SRC=10.0.2.10, DST=10.0.4.10, PROTO=17
              0A00 020A 0A00 040A 0011

UDP header SRCP=9000, DSTP=7, LEN=13, LEN-PH=13, CKSUM=0
           2328 0007 000D 000D

DATA       5445 5354 0A00

   00001010 00000000 = 10.0
   00000010 00001010 = 2.10
   00001010 00000000 = 10.0
   00000100 00001010 = 4.10
   00000000 00010001 =   17
   00100011 00101000 = 9000
   00000000 00000111 =    7
   00000000 00001101 =   13
   00000000 00001101 =   13
   00101010 01000101
   00101001 01010100
   00001010 00000000
  ------------------
  ~11101111 00000111     EF07 ~EF07 = 10F8
   00010000 11111000
```

Cálculo de Checksum (ejemplo con Carry(C))

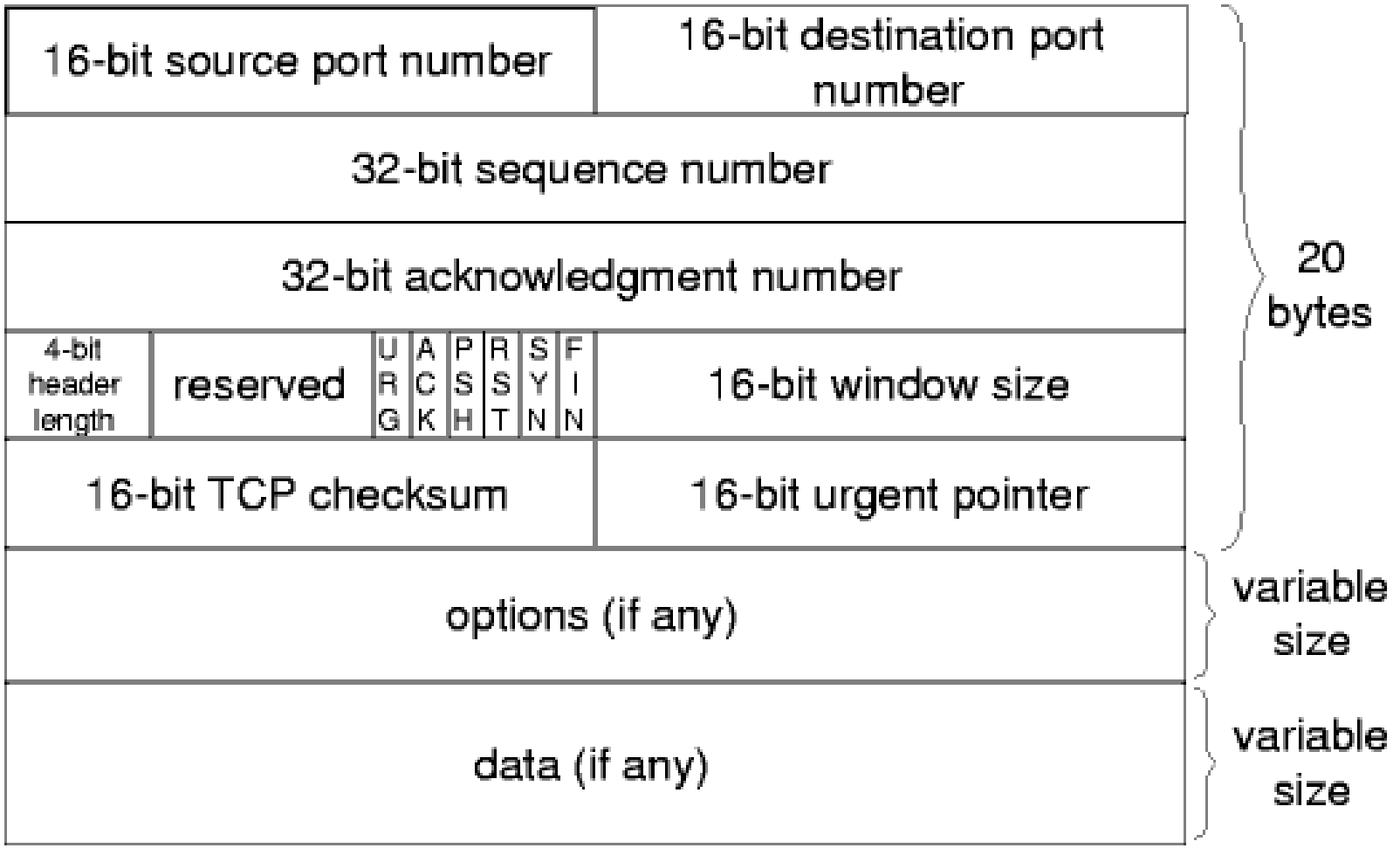
```
Pseudo header SRC=10.0.2.10, DST=10.0.4.10, PROTO=17
              0A00 020A 0A00 040A 0011

UDP header SRCP=9000, DSTP=7, LEN=15, LEN-PH=15, CKSUM=0
           2328 0007 000F 000F

DATA       5445 5354 0A00 FF[00] [00] = v. padding

   00001010 00000000 = 10.0
   00000010 00001010 = 2.10
   00001010 00000000 = 10.0
   00000100 00001010 = 4.10
   00000000 00010001 =   17
   00100011 00101000 = 9000
   00000000 00000111 =    7
   00000000 00001111 =   15
   00000000 00001111 =   15
   01010100 01000101
   01010011 01010100
   00001010 00000000
   11111111 00000000
  ------------------
{1}11101110 00010111
                  {1}
  ------------------
  ~11101110 00011000
   00010001 11100111       EE18 ~EE18 = 11E7
```

Segmento TCP



Segmento TCP (Cont.)

- Puertos: MUX/DEMUX.
- No tiene Longitud total, si de HDR LEN (variable, max 60B Unit=4B).
- Total LEN se computa para PseudoHDR, no viaja en el segmento.
- Checksum:
  - Cálculo Ca1. Obligatorio, calculado, igual que UDP.
  - Si tiene error podría pedir retransmisión, implementación de TCP descarta y espera RTO (Retransmission Timer).
- Necesidad de manejar Timers, RTO (tmout. por cada segmento). (implementaciones lo manejan más eficiente).

Segmento TCP (Cont.)

- Campos de Sesiones: Flags: SYN, FIN.
- Campo de Detección de Errores: Checksum.
- Campos de Control de Errores: ACK, Nro. Sec (#Seq), Nro. Ack (#Ack).
- Campo de Control de Flujo: a los de errores se agrega, Win.
- Campos de Congestión: agrega flags si participa la red.
- Máquina de estado finita por cada conexión.

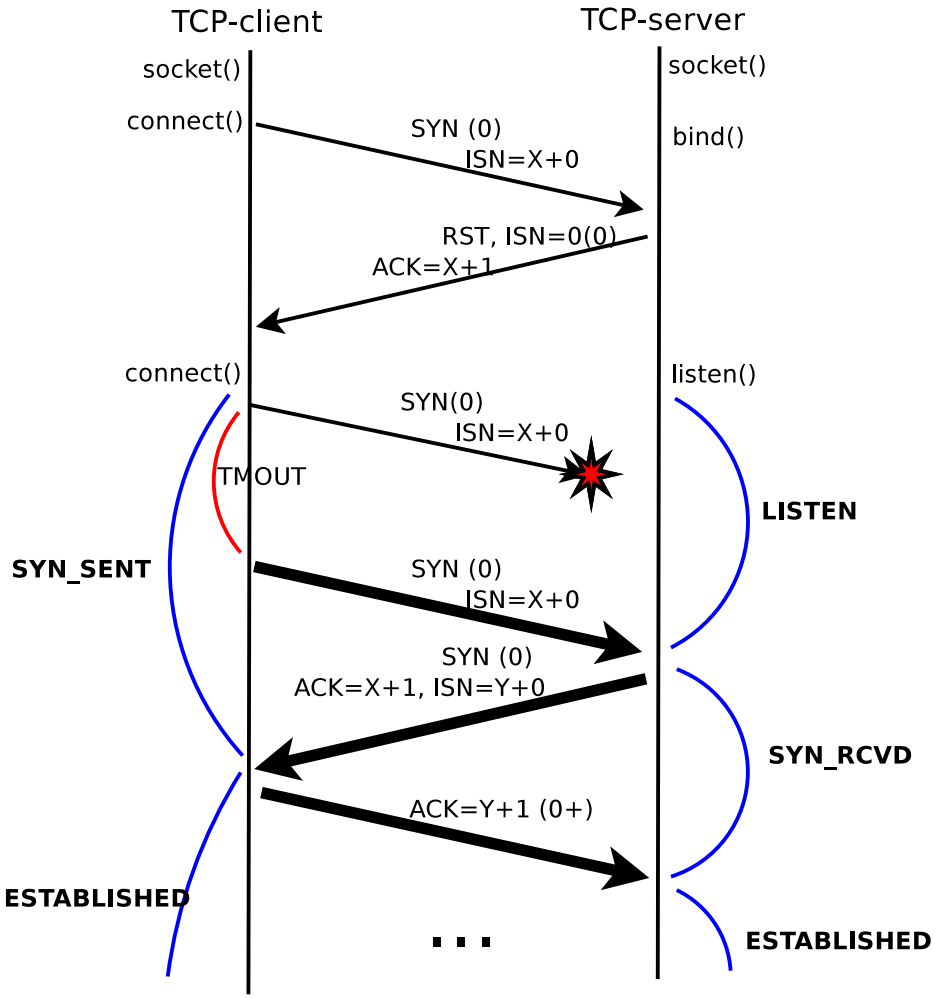
Segmento TCP (Cont.)

- Permite Opciones y Negociación.
- TCP entrega y envía lo datos agrupados o separados de forma dis-asociada de la aplicación:
  - La aplicación puede enviar 300 bytes en un write y TCP lo podría enviar en 3 segmentos separados de 100 bytes c/u.
  - La aplicación puede enviar 100 bytes y luego otros 200 y TCP esperar para enviarlos todos juntos.
  - La aplicación puede intentar leer 200 bytes del buffer y TCP solo entregar 150 bytes y luego el resto.

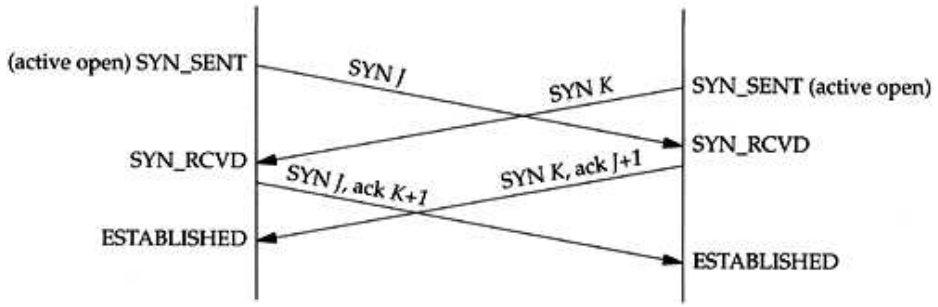
TCP Establecimiento de Conexión

- Flags: SYN (Synchronize), ACK (Acknowledge) y RST (Reset).
- 3Way-Handsake (3WH).
- En el 3 segmento se puede enviar info.
- el ISN (Initial Sequence Number), se utiliza un contador que se incrementa cada 4 mseg.
- RST si no hay proceso en estado LISTEN.
- Open Pasivo (servidor) y Activo (cliente).
- Open simultáneo.

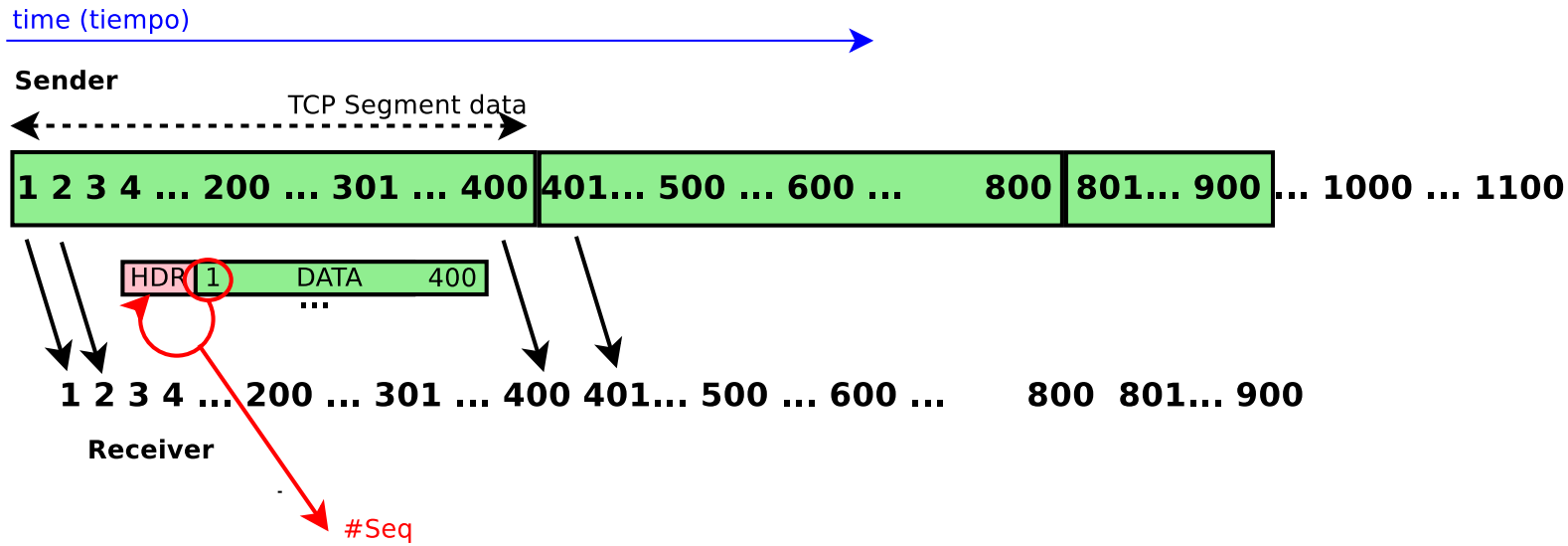
3 Way Handshake



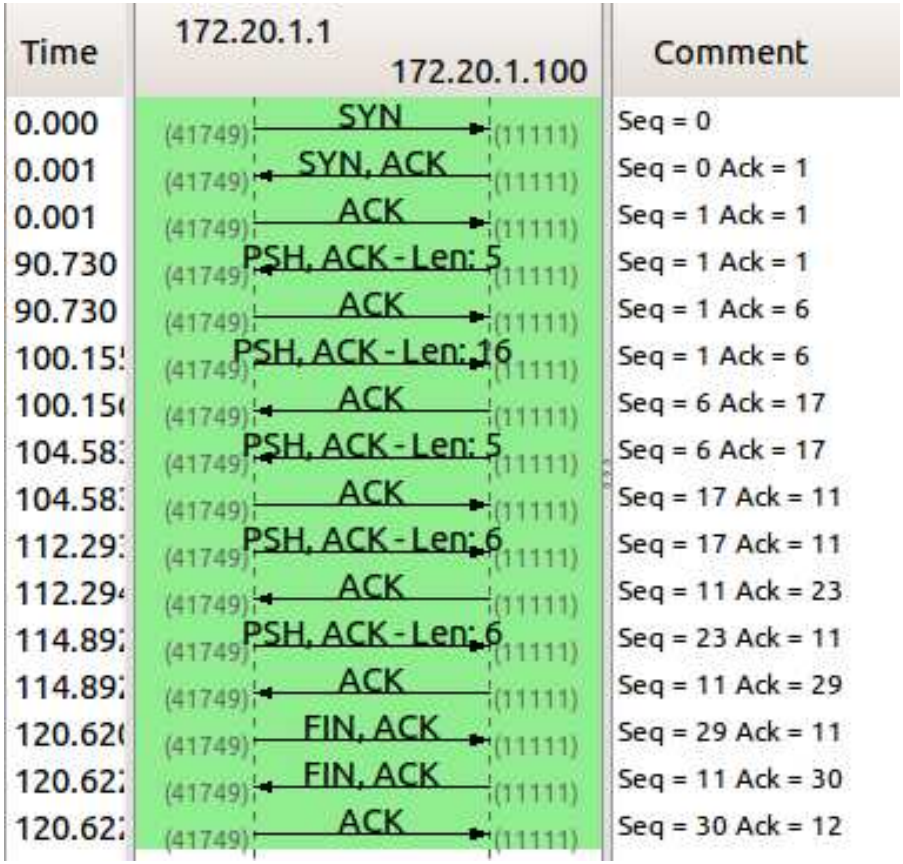
3 Way Handshake (Open Simultáneo)



Orientado a streams (secuencia en orden de bytes)



Nros. de Secuencia/ACK TCP



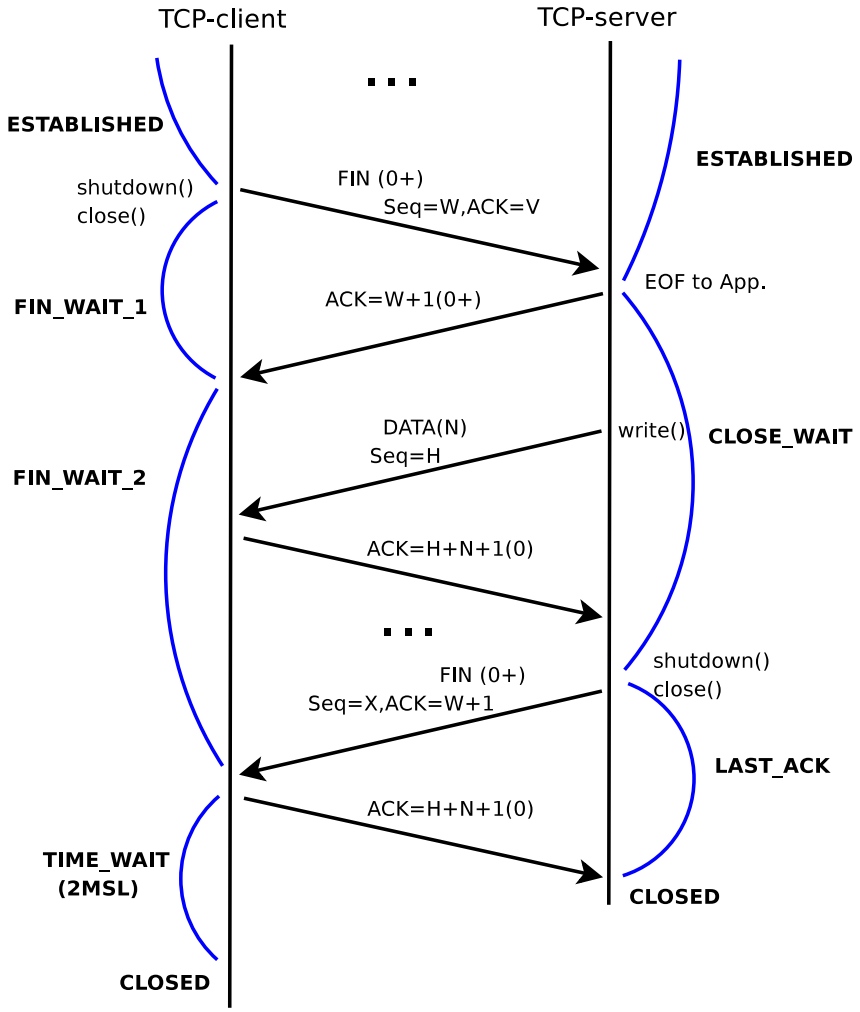
Control de Errores TCP

- Errores que pueden existir en IP:
  - Pérdida de paquetes: descartados.
  - Des-ordenados, retardados.
  - Duplicados.
  - Corrompidos.
- TCP intenta solucionarlos con el control de Errores que implementa.

TCP Cierre de Conexión

- Flags: FIN (Finish), ACK y RST.
- 4Way-Close (4WC).
- Posibilidad de Half-Close.
- Podría cerrarse en 3WC.
- Espera en TIME_WAIT, 2MSL (aprox. 2*2min).
- Evitar con `SO_REUSEADDR`.
- Cierre incorrecto con RST.
- Close simultáneo.

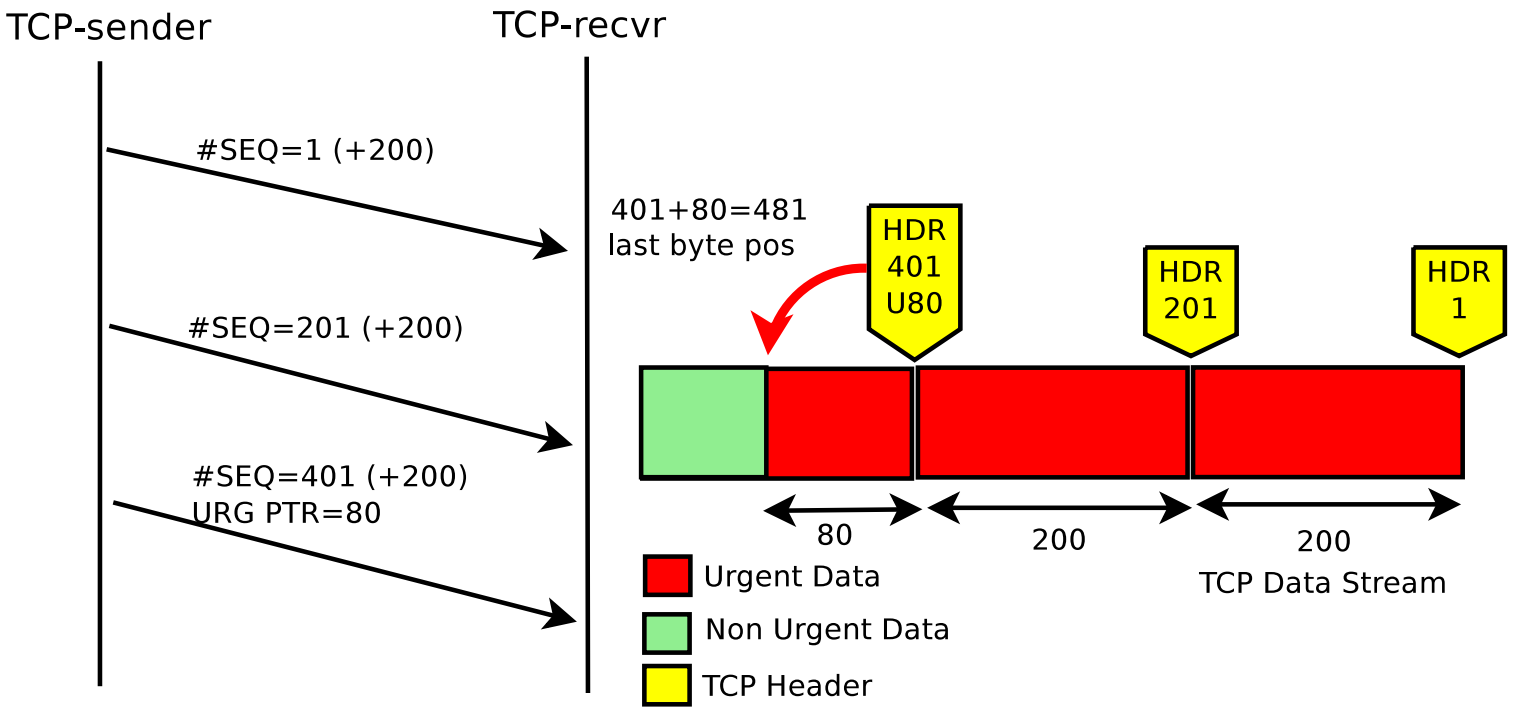
TCP Close


TCP-client
TCP-server
ESTABLISHED
shutdown()
close()
FIN (0+)
Seq=W,ACK=V
ESTABLISHED
EOF to App.
ACK=W+1(0+)
FIN_WAIT_1
DATA(N)
Seq=H
write()
CLOSE_WAIT
FIN_WAIT_2
ACK=H+N+1(0)
shutdown()
close()
FIN (0+)
Seq=X,ACK=W+1
LAST_ACK
ACK=H+N+1(0)
TIME_WAIT
(2MSL)
CLOSED
CLOSED

Otros campos TCP

- TCP entrega y envía lo datos agrupados o separados de forma dis-asociada de la aplicación.
- Datos Urgentes: URG.
  - Urgent Pointer válido su URG=1.
  - Indica: offset positivo + #Seq = last Data Urgent byte.
  - Indicar a la App. datos urgentes, debe leer.
  - Debería combinarse con PSH. Habitualmente llamado OOB data (TCP no soporta OOB!!!).
- Pushear datos: PSH.
  - Fuerza a TCP a pasar datos a la App.
  - No lo deja “Bufferear” los datos recibidos (Input).

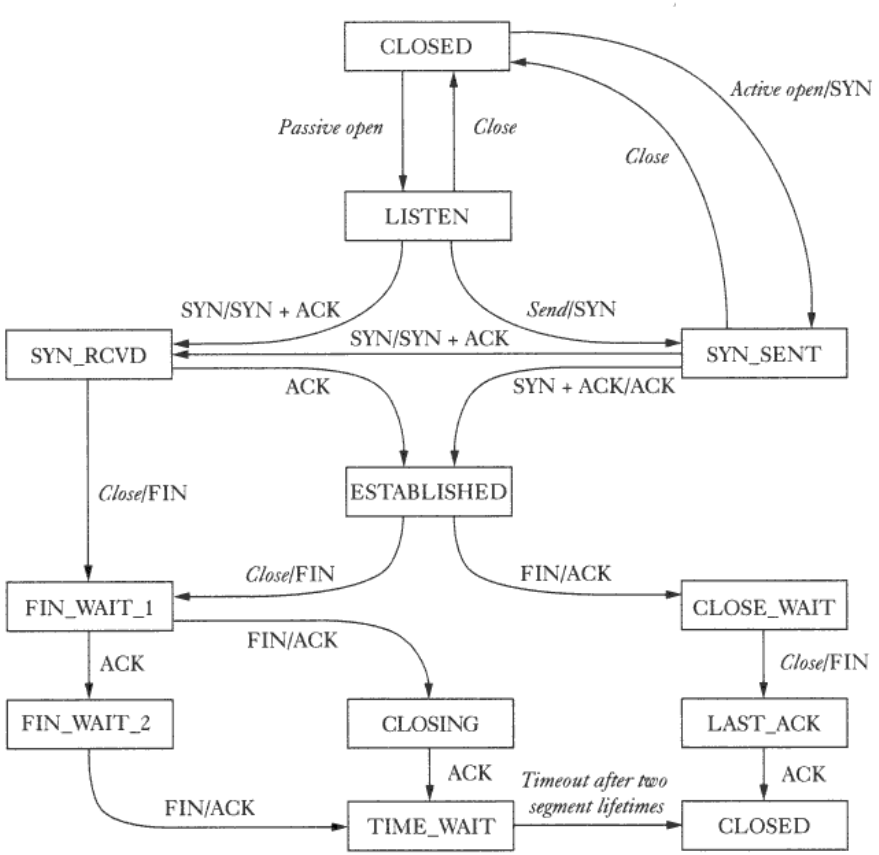
Urgent Pointer



Opciones TCP

- Maximum Segment Size (MSS), min. recomendado 536B, RFC-879 (basado en MTU=576, TCP+IP=40). Aclaraciones en RFC-6691.
- Window Scaling.
- Selective Acknowledgements (SACK).
- Timestamps.
- NOP.
- Otras.

TCP Diagrama de Estados, Reducido



Fuentes de Información

- Kurose/Ross: Computer Networking (6th Edition).
- TCP/IP Illustrated, Volume 1: The Protocols, W. Richard Stevens, K. Fall (2nd. ed).
- RFCs: http://www.faqs.org/rfcs/rfc793, rfc798, ...
- Wikipedia http://www.wikipedia.org.
- CS 144: Introduction to Computer Networking, Stanford Course.
- TCP/IP Guide: http://www.tcpipguide.com/.
- Internet ...