tutoría Redes de Datos

Capítulo 5: Capa de transporte

udp Escuela de Informática y Telecomunicaciones
FACULTAD DE INGENIERÍA Y CIENCIAS

Ph.D. Nicolás Boettcher

Problemas con los segmentos

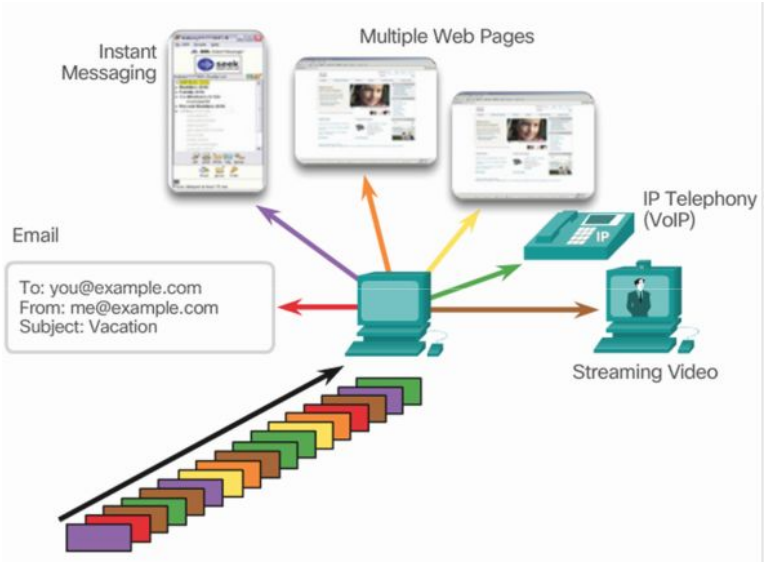
- Recordar PDU de capa 4.
- Pérdidas de segmentos por rutas congestionadas o enlaces caídos.
- Segmentos llegan fuera de orden.
- Segmentos se duplican por retardos que obligan a la retransmisión.

¿Cómo evitar estos problemas?

Segmentación

Segmentación de datos y re-ensamblado

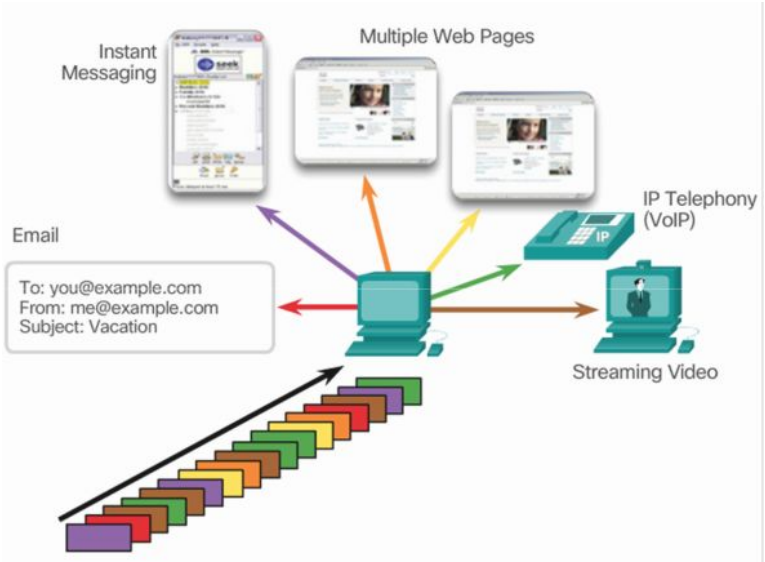
- Re-ensambla segmentos para pasarlos a la aplicación de manera coherente.
- Re-ensamblado es posible gracias a la enumeración y secuenciación de los segmentos.
- Sin segmentación solo una aplicación podría comunicarse a la vez.



Aplicaciones

Identificar las aplicaciones

- Asegurar que cada aplicación en ejecución reciba los datos correctamente
- Para ello utiliza puertos específicos para cada aplicación



Maximum Segment Size

MSS se define en la conexión TCP y permite saber el tamaño máximo de los paquetes a segmentar.

MTU: 1500 bytes

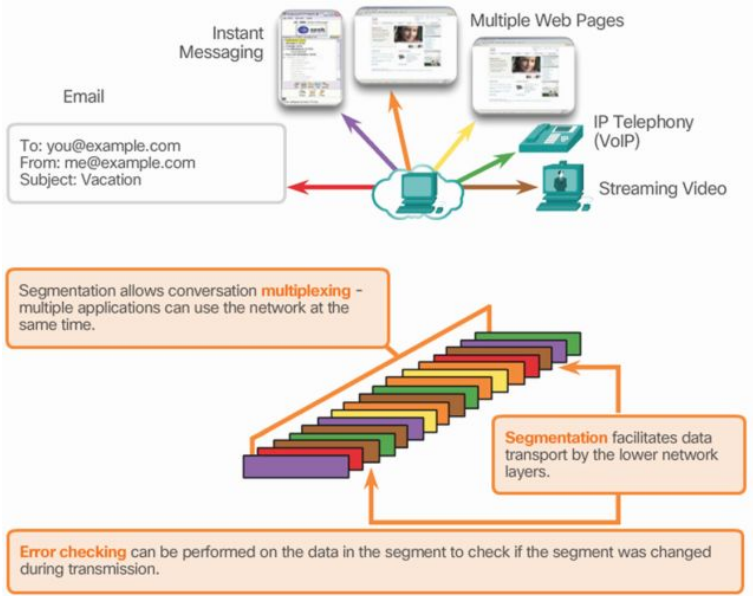
```
~ ifconfig wlp0s20f3 | grep -o "mtu [0-9]*"
mtu 1500
~ ifconfig lo | grep -o "mtu [0-9]*"
mtu 65536
```

MSS: 1460 bytes (40 bytes header IP/TCP)

ip route add 192.168.1.0/24 dev eth0 advmss 1460

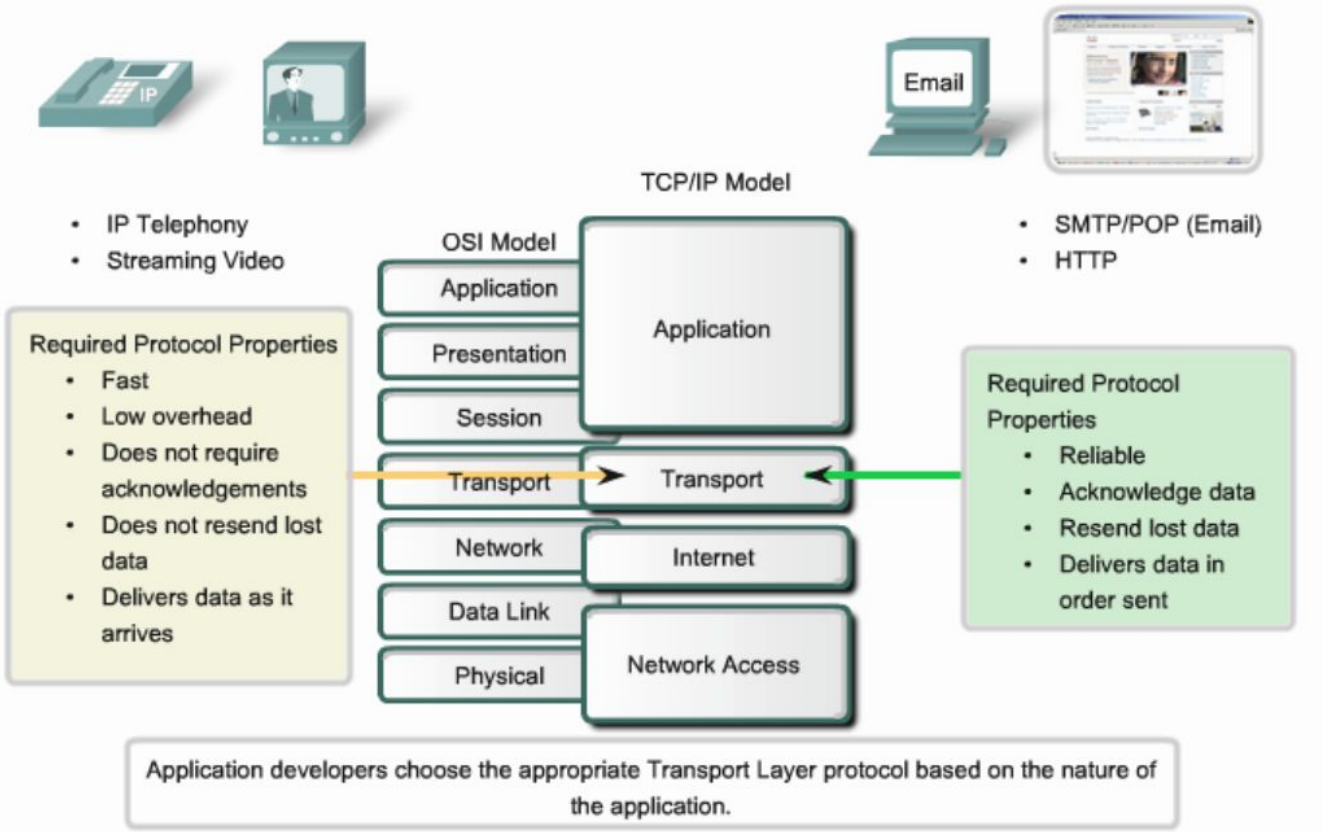
Multiplexing

- La segmentación de los datos permite que múltiples conexiones de diferentes usuarios sean multiplexadas en la misma red.
- La capa de transporte agrega un header (o encabezado) que contiene información para identificar cada segmento de datos.



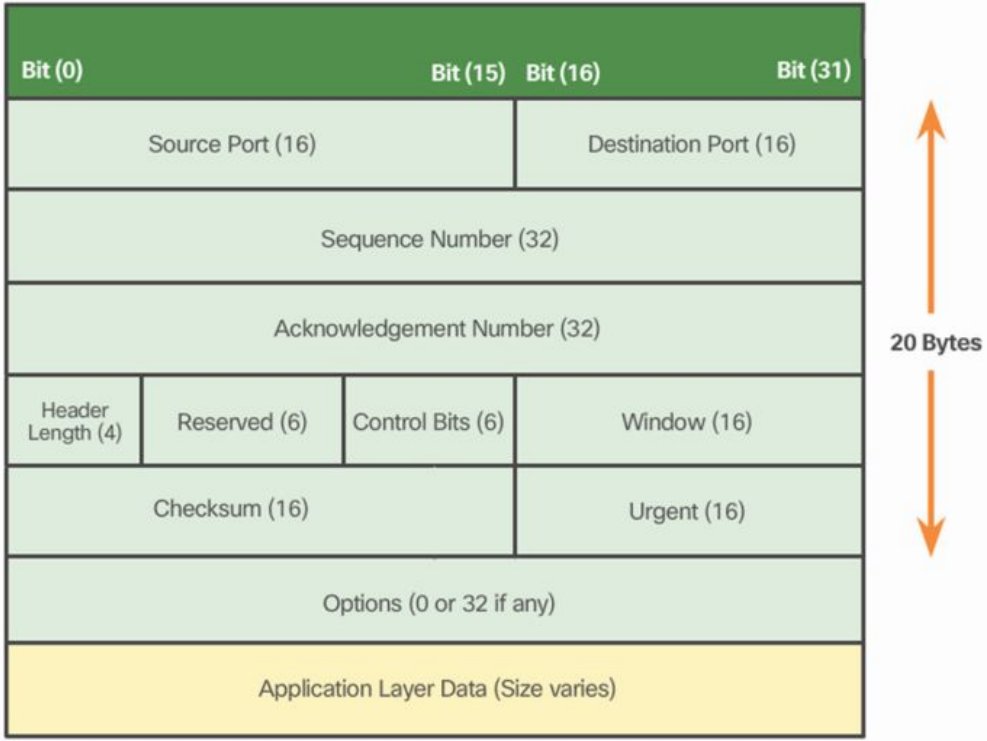
Protocolos de capa de transporte

UDP vs TCP


IP
Email
TCP/IP Model
OSI Model
Application
Presentation
Session
Transport
Network
Data Link
Physical
Application
Transport
Internet
Network Access
IP Telephony
Streaming Video
SMTP/POP (Email)
HTTP
Required Protocol Properties
Fast
Low overhead
Does not require acknowledgements
Does not resend lost data
Delivers data as it arrives
Required Protocol Properties
Reliable
Acknowledge data
Resend lost data
Delivers data in order sent
Application developers choose the appropriate Transport Layer protocol based on the nature of the application.

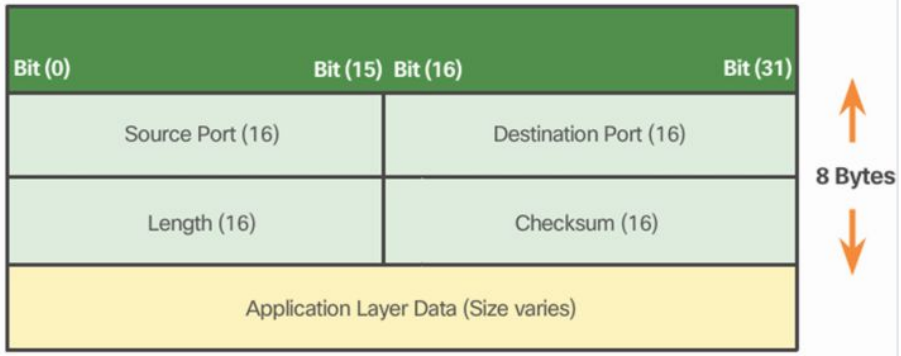
Header TCP

- TCP es un protocolo stateful, vale decir mantiene información acerca del estado de la comunicación, lo enviado y lo recibido.
- El encabezado TCP (header) es de 20 bytes y encapsula la información de la capa de aplicación.



Header UDP

- UDP es un protocolo stateless, vale decir ni el receptor ni el origen deben llevar un seguimiento del estado de la comunicación
- Si se requiere de confiabilidad debe ser manejada por la aplicación
- Los paquetes enviados por UDP son conocidos bajo el nombre de datagramas
- UDP tiene una cabecera de 8 bytes



Puertos

Para poder realizar varias conexiones de forma simultánea, la IP tiene asignados 65536 puntos de salida y entrada de datos, algunos de ellos asignados por IANA (RFC 1700)

Port Numbers

Port Number Range	Port Group
0 to 1023	Well-known Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

```
[redacted] ~ cat /etc/services | grep "netstat" -A30
netstat         15/tcp
qotd            17/tcp          quote
chargen         19/tcp          ttytst source
chargen         19/udp          ttytst source
ftp-data        20/tcp
ftp             21/tcp
fsp             21/udp          fspd
ssh             22/tcp                          # SSH Remote Login Protocol
telnet          23/tcp
smtp            25/tcp          mail
time            37/tcp          timserver
time            37/udp          timserver
whois           43/tcp          nicname
tacacs          49/tcp                          # Login Host Protocol (TACACS)
tacacs          49/udp
domain          53/tcp                          # Domain Name Server
domain          53/udp
```

¿Puerto TCP o UDP?

El puerto puede ser TCP o UDP

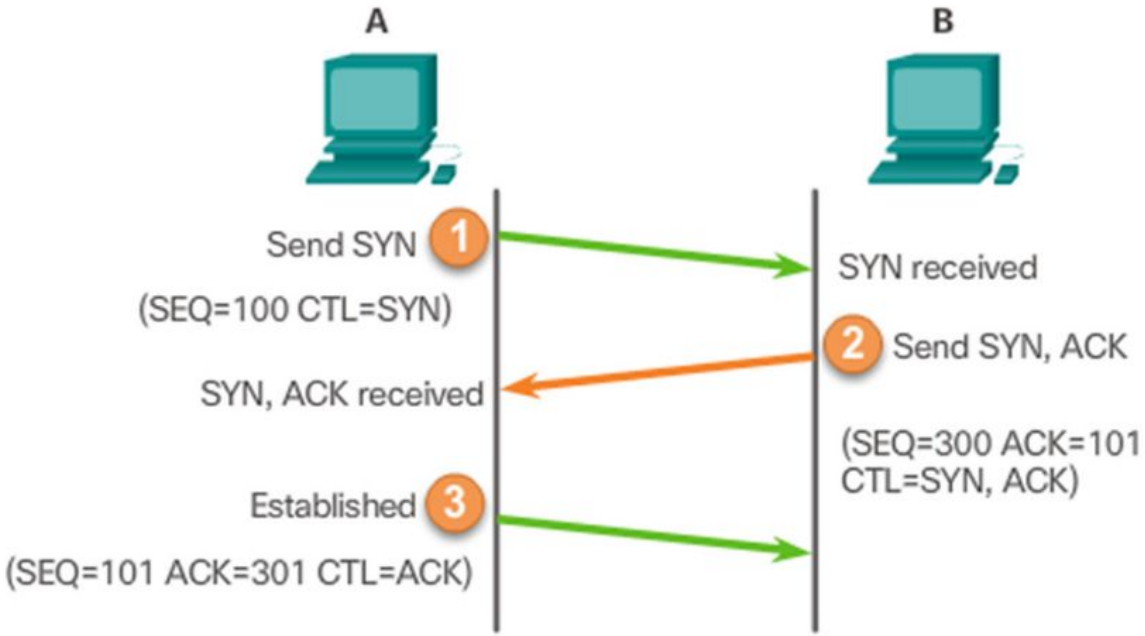
```
~ grep domain /etc/services
domain          53/tcp                  # Domain Name Server
domain          53/udp
domain-s        853/tcp                 # DNS over TLS [RFC7858]
domain-s        853/udp                 # DNS over DTLS [RFC8094]
```

Netstat en UNIX

- Conexiones desconocidas pueden ser significar amenazas importantes para nuestros sistemas
- Netstat permite listar todos los protocolos en uso, la dirección local y los puertos asociados, la dirección remota y sus puertos y el estado de la conexión.

```
[redacted] netstat -n | sort | more
Active Internet connections (w/o servers)
Active UNIX domain sockets (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
Proto RefCnt Flags       Type       State         I-Node   Path
tcp        0      0 127.0.0.1:35686         127.0.0.1:40771         TIME_WAIT
tcp        0      0 127.0.0.1:36600         127.0.0.1:40329         ESTABLISHED
tcp        0      0 127.0.0.1:39876         127.0.0.1:40823         TIME_WAIT
tcp        0      0 127.0.0.1:40329         127.0.0.1:36600         ESTABLISHED
tcp        0      0 127.0.0.1:41125         127.0.0.1:47760         ESTABLISHED
tcp        0      0 127.0.0.1:41548         127.0.0.1:45251         ESTABLISHED
tcp        0      0 127.0.0.1:42815         127.0.0.1:43784         ESTABLISHED
tcp        0      0 127.0.0.1:43784         127.0.0.1:42815         ESTABLISHED
tcp        0      0 127.0.0.1:45251         127.0.0.1:41548         ESTABLISHED
tcp        0      0 127.0.0.1:46846         127.0.1.1:139           ESTABLISHED
tcp        0      0 127.0.0.1:47760         127.0.0.1:41125         ESTABLISHED
tcp        0      0 127.0.1.1:139           127.0.0.1:33858         ESTABLISHED
tcp        0      0 127.0.1.1:139           127.0.0.1:46846         ESTABLISHED
tcp        0      0 192.168.0.5:32954       64.233.186.95:443       ESTABLISHED
tcp        0      0 192.168.0.5:33244       104.22.22.88:443        ESTABLISHED
tcp        0      0 192.168.0.5:33572       64.233.186.121:443      ESTABLISHED
tcp        0      0 192.168.0.5:34572       64.233.186.189:443      ESTABLISHED
tcp        0      0 192.168.0.5:34854       52.42.50.159:443        ESTABLISHED
```

3-way Handshake



CTL = Which control bits in the TCP header are set to 1
A sends ACK response to B?

Validando valores

```
54673 > http [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=992105 TSecr=0 WS=64
http > 54673 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 MSS=1380
54673 > http [ACK] Seq=1 Ack=1 Win=5840 Len=0
```

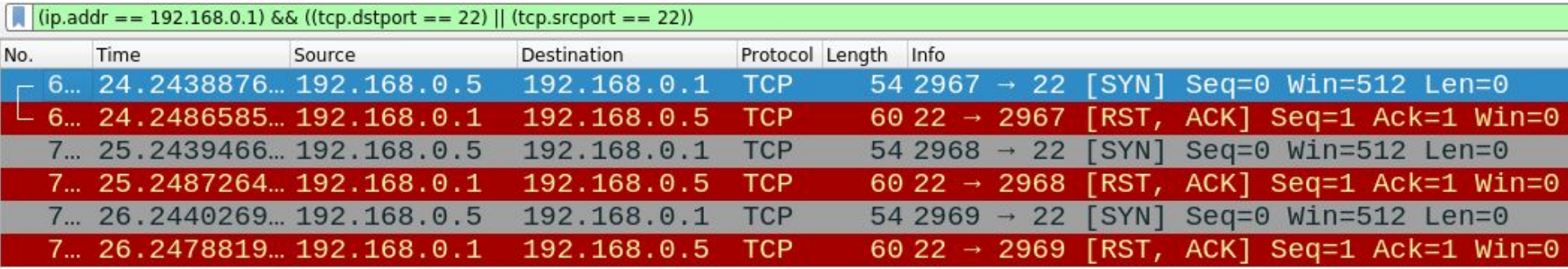
SYN -> seq=x

SYN,ACK -> ack=x+1, seq=y

ACK -> ack=y+1, seq=x+1

Servicio no disponible

¿Qué pasa si hay una petición y el puerto no está disponible?



Finalizando la conexión

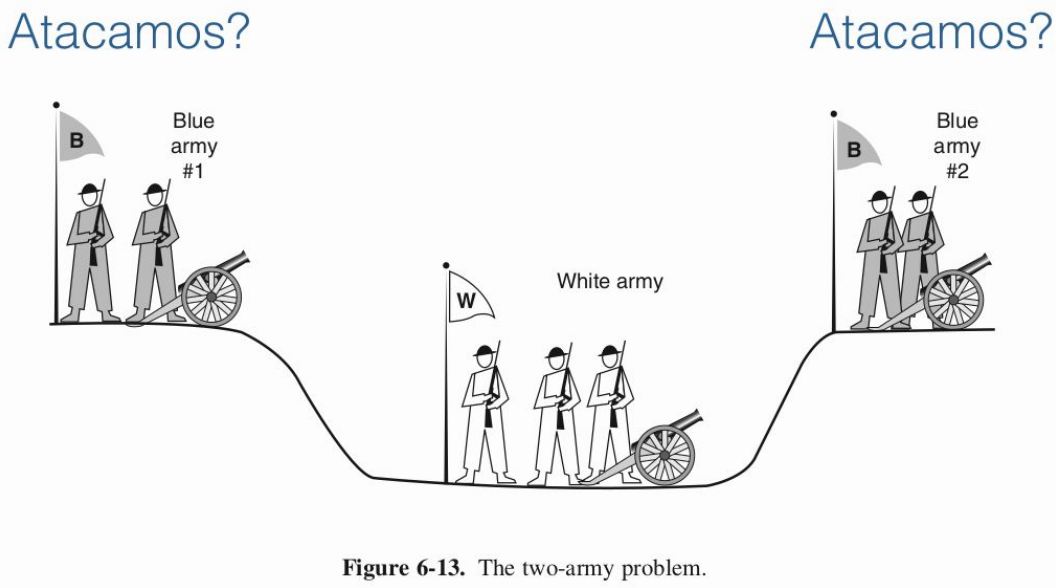
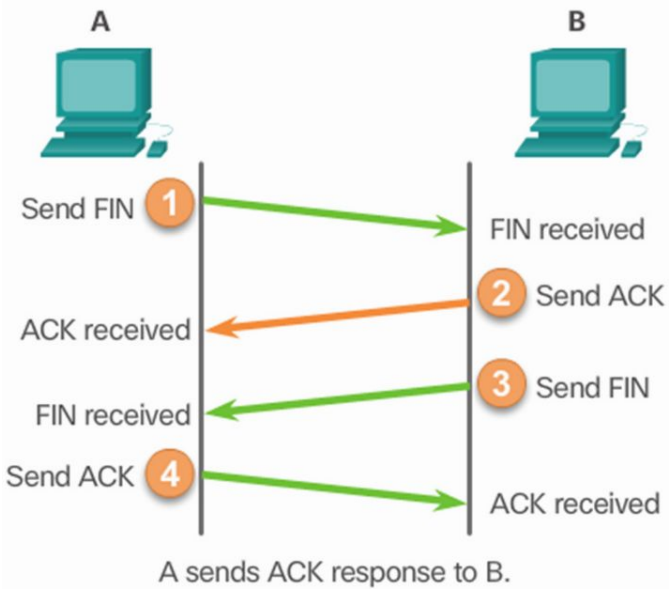


Figure 6-13. The two-army problem.



A sends ACK response to B.

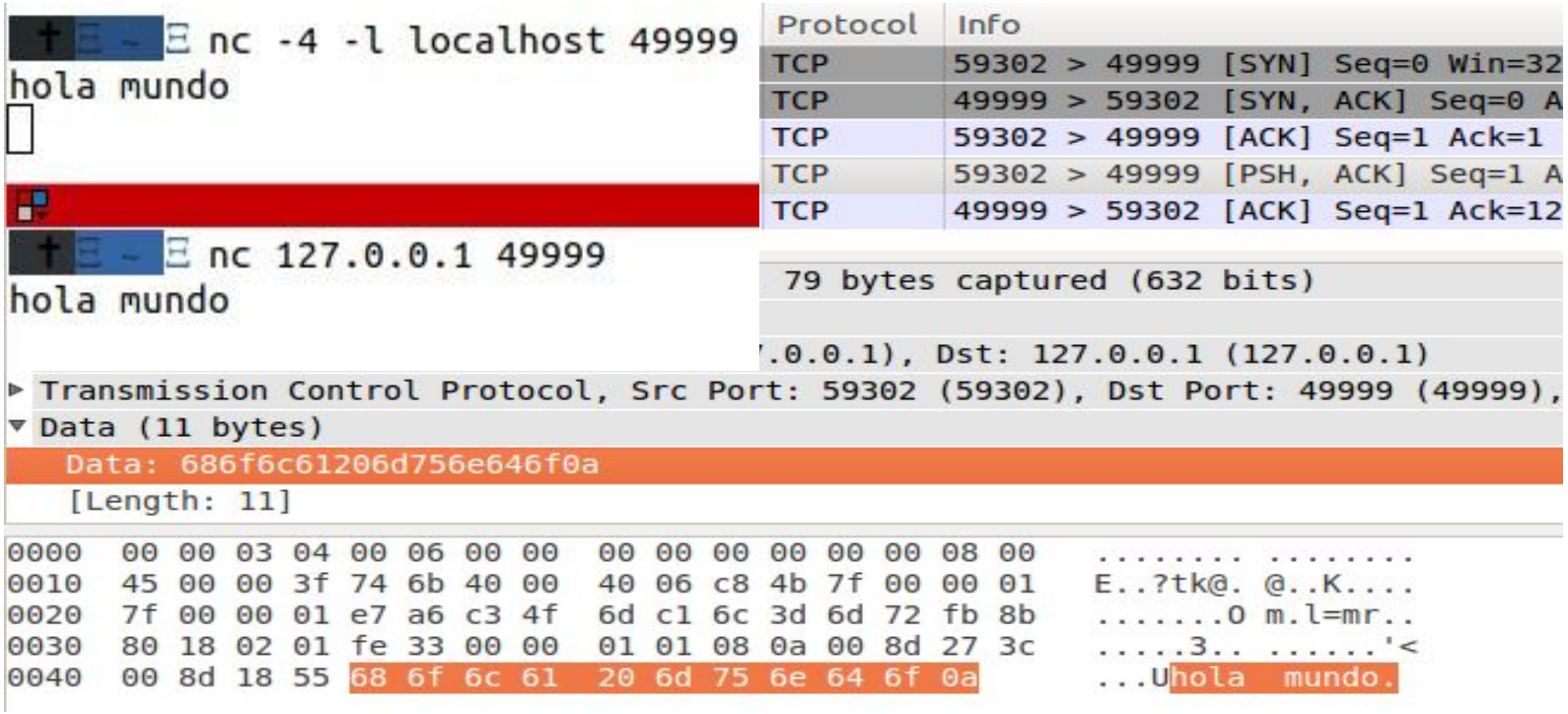

```
nc -4 -l localhost 49999
a
b
c
1
2
3
```

```
nc localhost 49999
a
b
c
1
2
3
```

```
nc -lu localhost 49999
a
b
c
1
2
3
```

```
nc -u localhost 49999
a
b
c
1
2
3
```

Server/Client



SPEEDTEST by OOKLA

¿Se medirá con tráfico TCP o UDP?

Mismo enlace, distinto server. ¿Por qué tanta diferencia?