Operating Systems

Rohini Naik

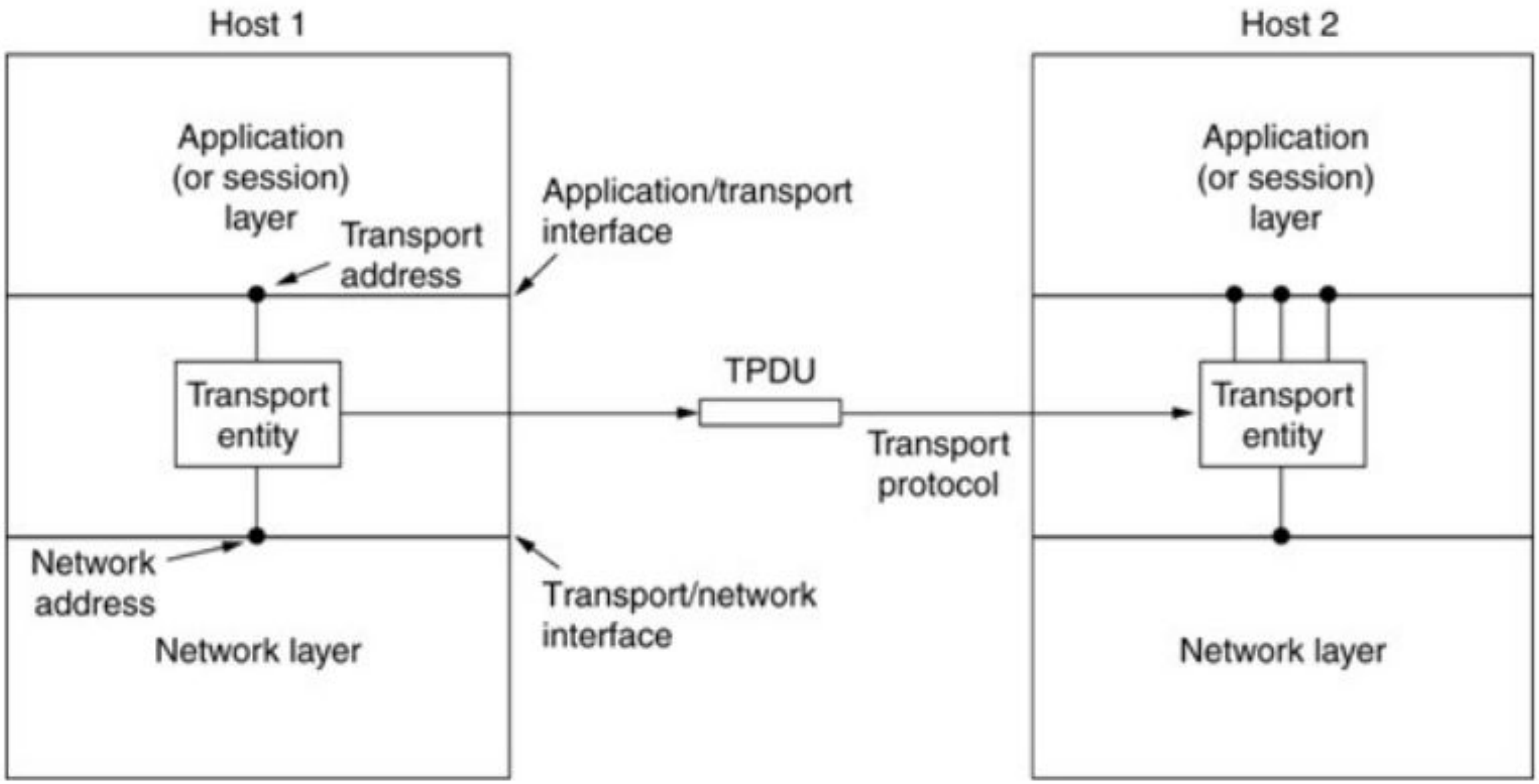
Unit 4

Transport Layer

The Transport Service

a) Services Provided to the Upper Layers
b) Transport Service Primitives
c) Berkeley Sockets

Services Provided to the Upper Layers



The network, transport, and application layers.

Why the transport layer ?

1. The network layer exists on end hosts ***and routers in the network.*** The end-user cannot control what is in the network. So the end-user establishes another layer, only at end hosts, to provide a transport service that is more reliable than the underlying network service.

2. While the network layer deals with only a few transport entities, the transport layer allows several concurrent applications to use the transport service.

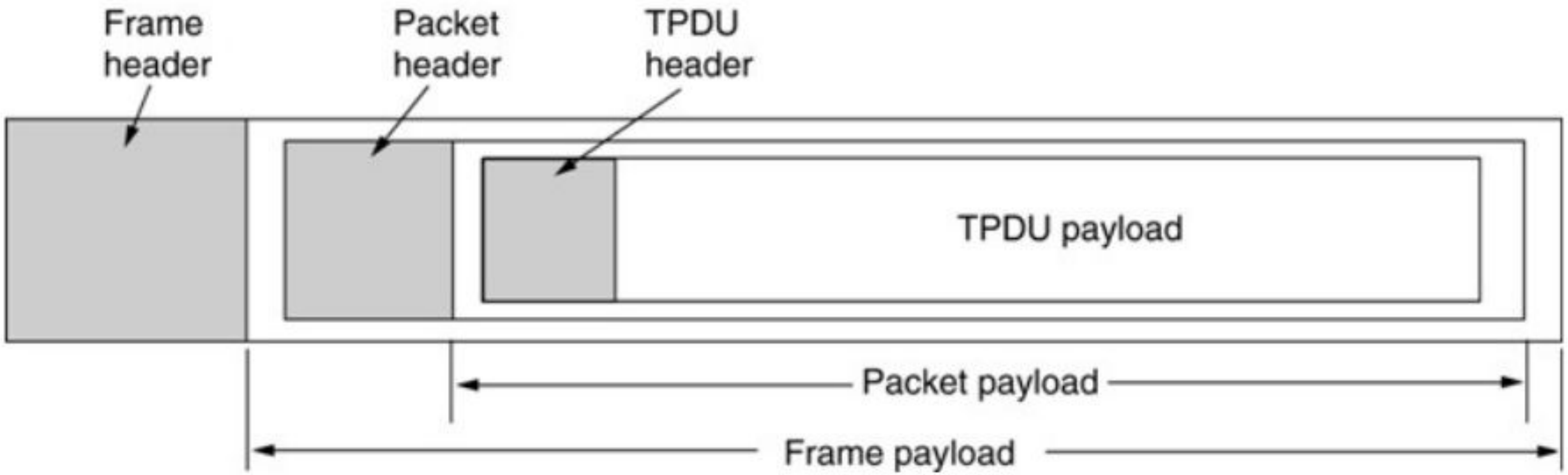
3. It provides a common interface to application writers, regardless of the underlying network layer. In essence, an application writer can write code once using the transport layer primitive and use it on different networks (but with the same transport layer).

Transport Service Primitives

Primitive	Packet sent	Meaning
LISTEN	(none)	Block until some process tries to connect
CONNECT	CONNECTION REQ.	Actively attempt to establish a connection
SEND	DATA	Send information
RECEIVE	(none)	Block until a DATA packet arrives
DISCONNECT	DISCONNECTION REQ.	This side wants to release the connection

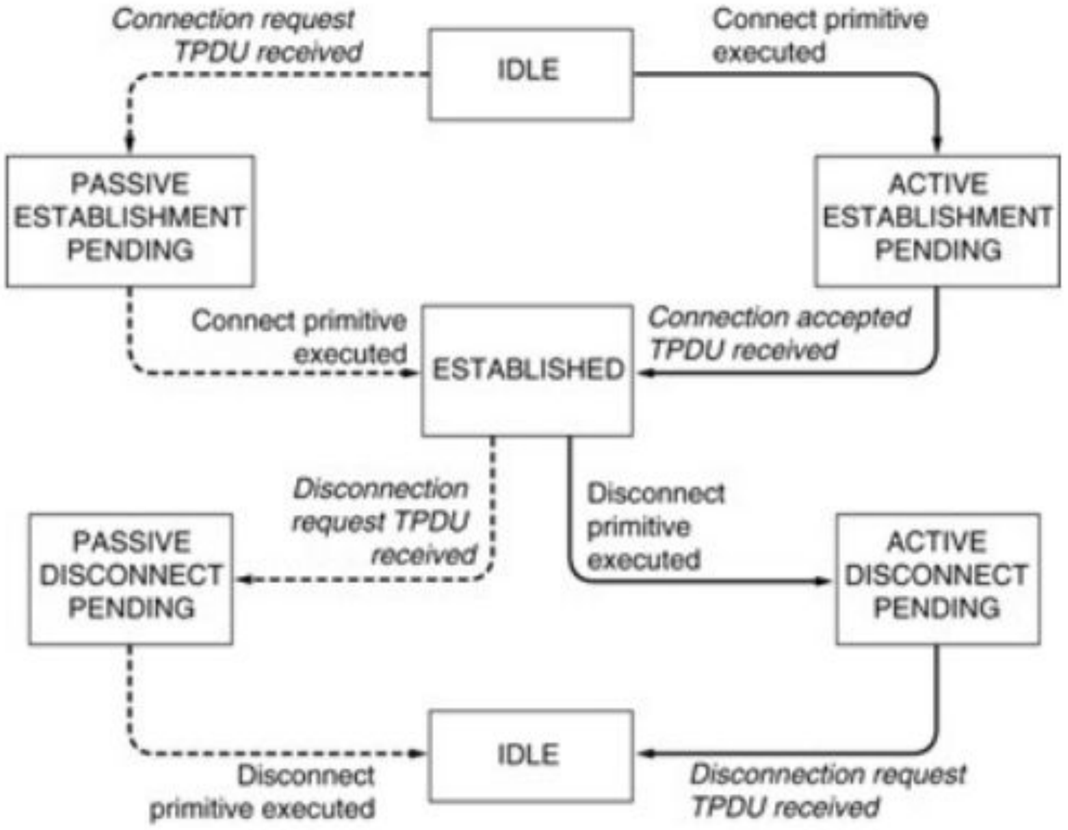
The primitives for a simple transport service.

Transport Service Primitives (2)



The nesting of TPDUs, packets, and frames.

Transport Service Primitives (3)



A state diagram for a simple connection management scheme. Transitions labelled in italics are caused by packet arrivals. The solid lines show the client's state sequence. The dashed lines show the server's state sequence.

Berkeley Sockets

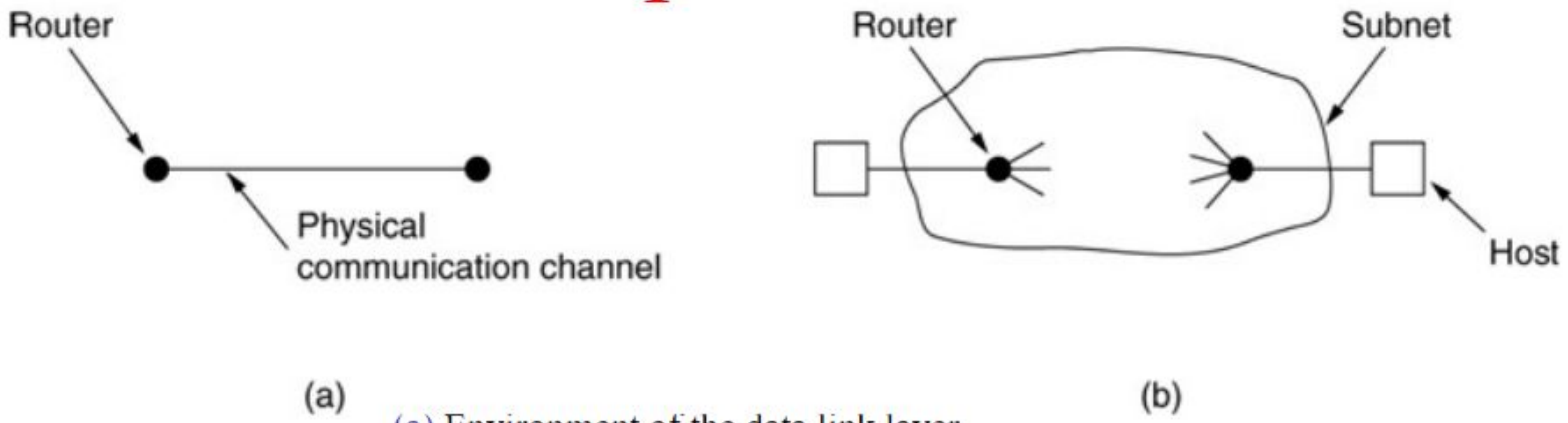
Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

The socket primitives for TCP.

Elements of Transport Protocols

a) Addressing
b) Connection Establishment
c) Connection Release
d) Flow Control and Buffering
e) Multiplexing
f) Crash Recovery

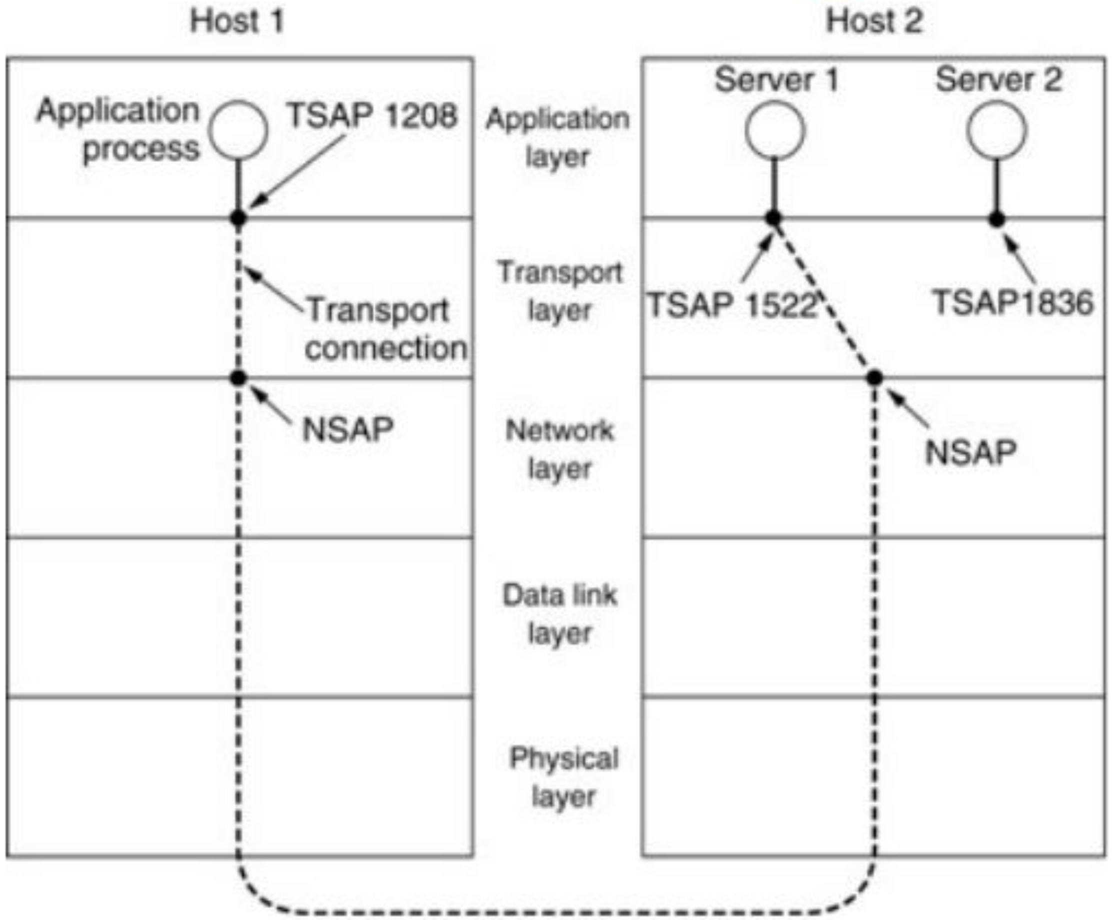
Transport Protocol



(a) Environment of the data link layer.
(b) Environment of the transport layer.

Both data link layer and transport layer do error control, flow control, sequencing. The differences are:
1. Storage capacity in subnet. Frames must arrive sequentially, TPDUs can arrive in any sequence.
2. Frames are delivered to hosts, TPDUs need to be delivered to users, so per user addressing and flow control within the hosts is necessary.

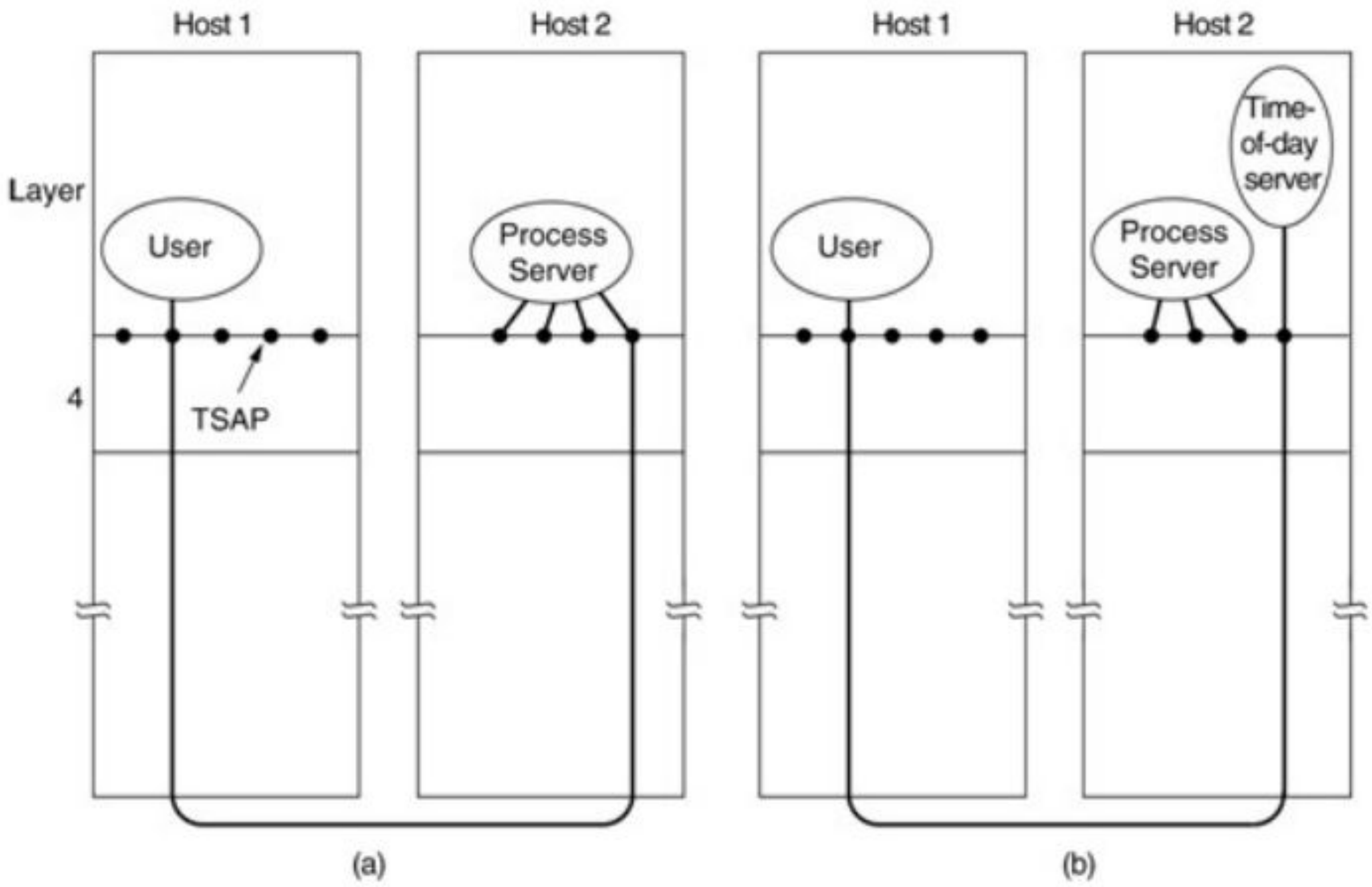
Addressing



TSAPs (Transport Service Access Point) , NSAPs (Network SAP).

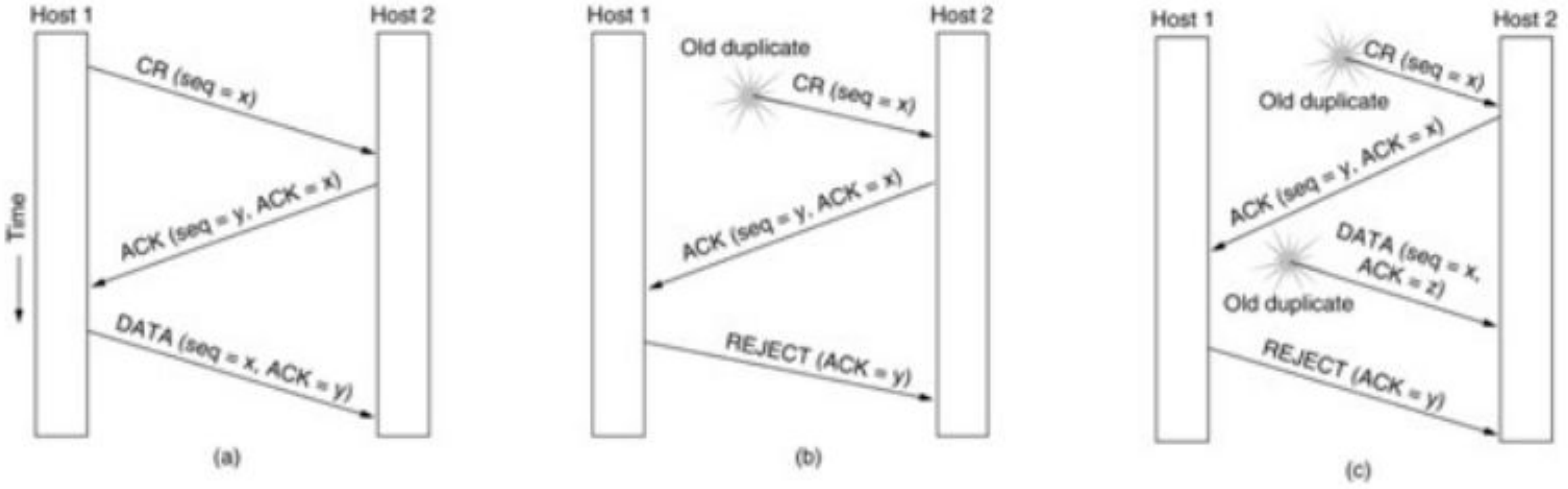
TCP calls TSAP s ... ports
ATM calls TSAPs ... AAL-SAP

Connection Establishment (1)



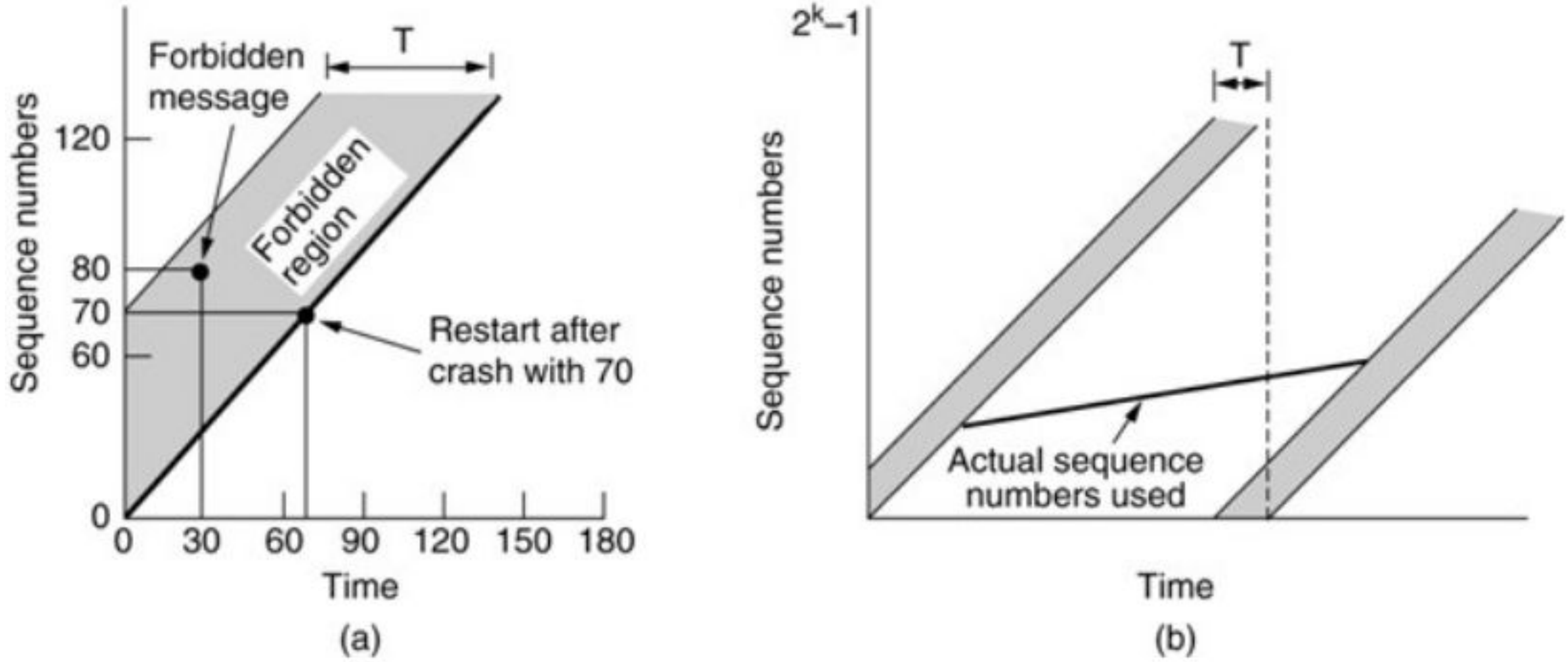
How a user process in host 1 establishes a connection with a time-of-day server in host 2.

Connection Establishment (2)



Three protocol scenarios for establishing a connection using a three-way handshake. CR denotes CONNECTION REQUEST.
(a) Normal operation,
(b) Old CONNECTION REQUEST appearing out of nowhere.
(c) Duplicate CONNECTION REQUEST and duplicate ACK.

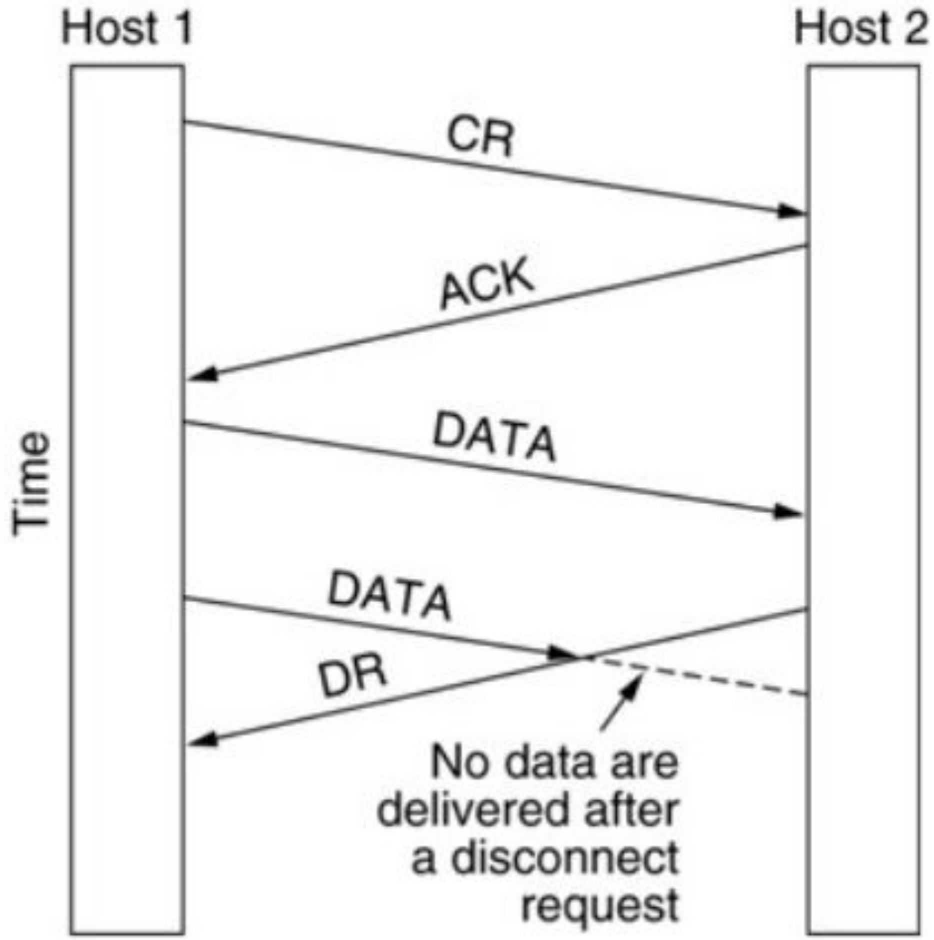
Connection Establishment (3)



(a) TPDUs may not enter the forbidden region.

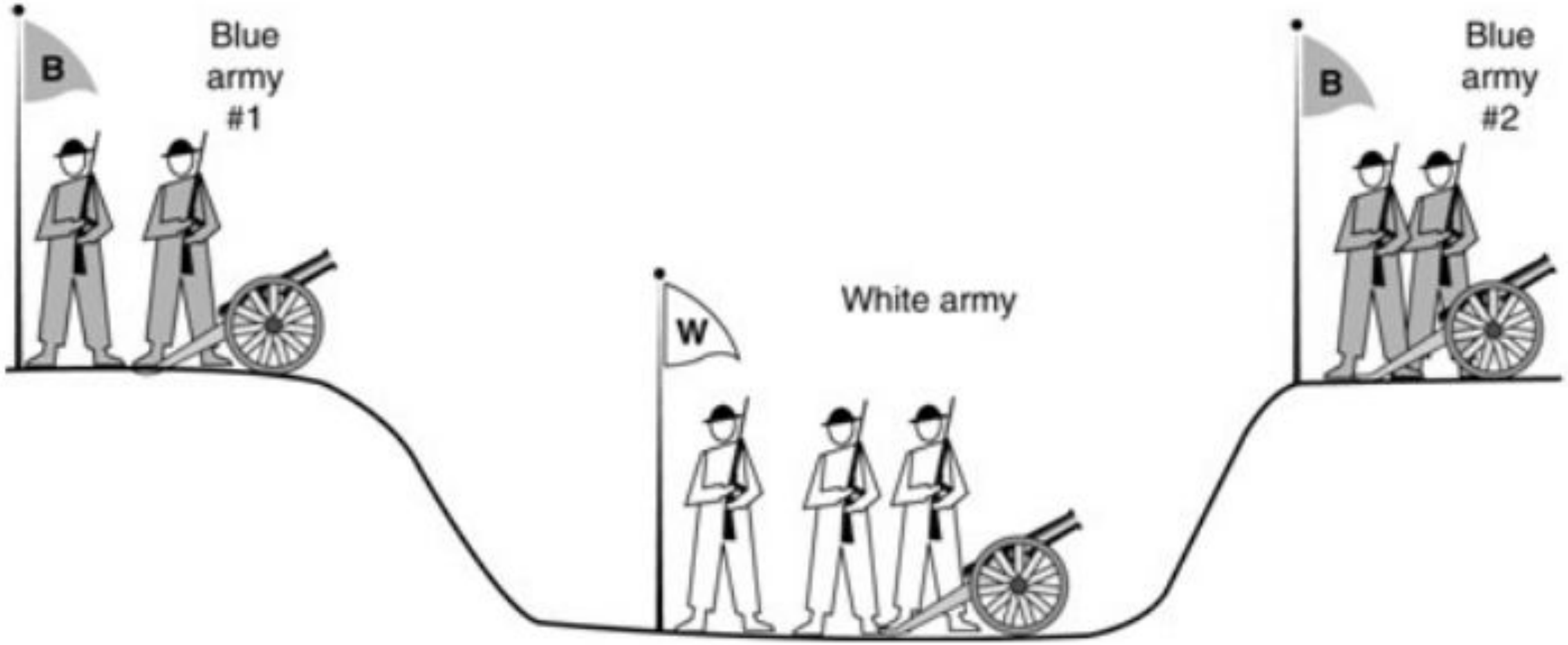
(b) The resynchronization problem.

Connection Release



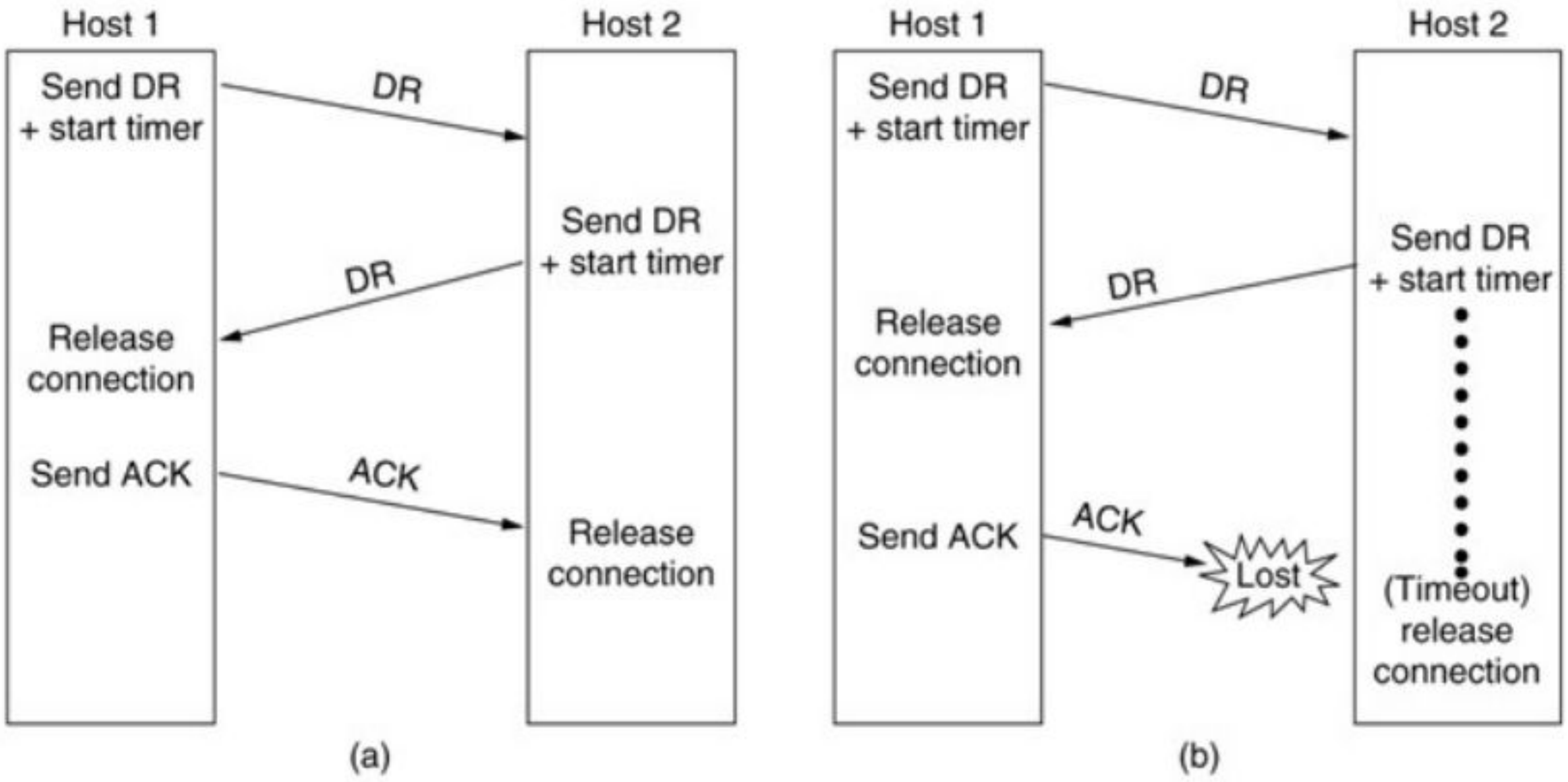
Abrupt disconnection with loss of data.

Connection Release (2)



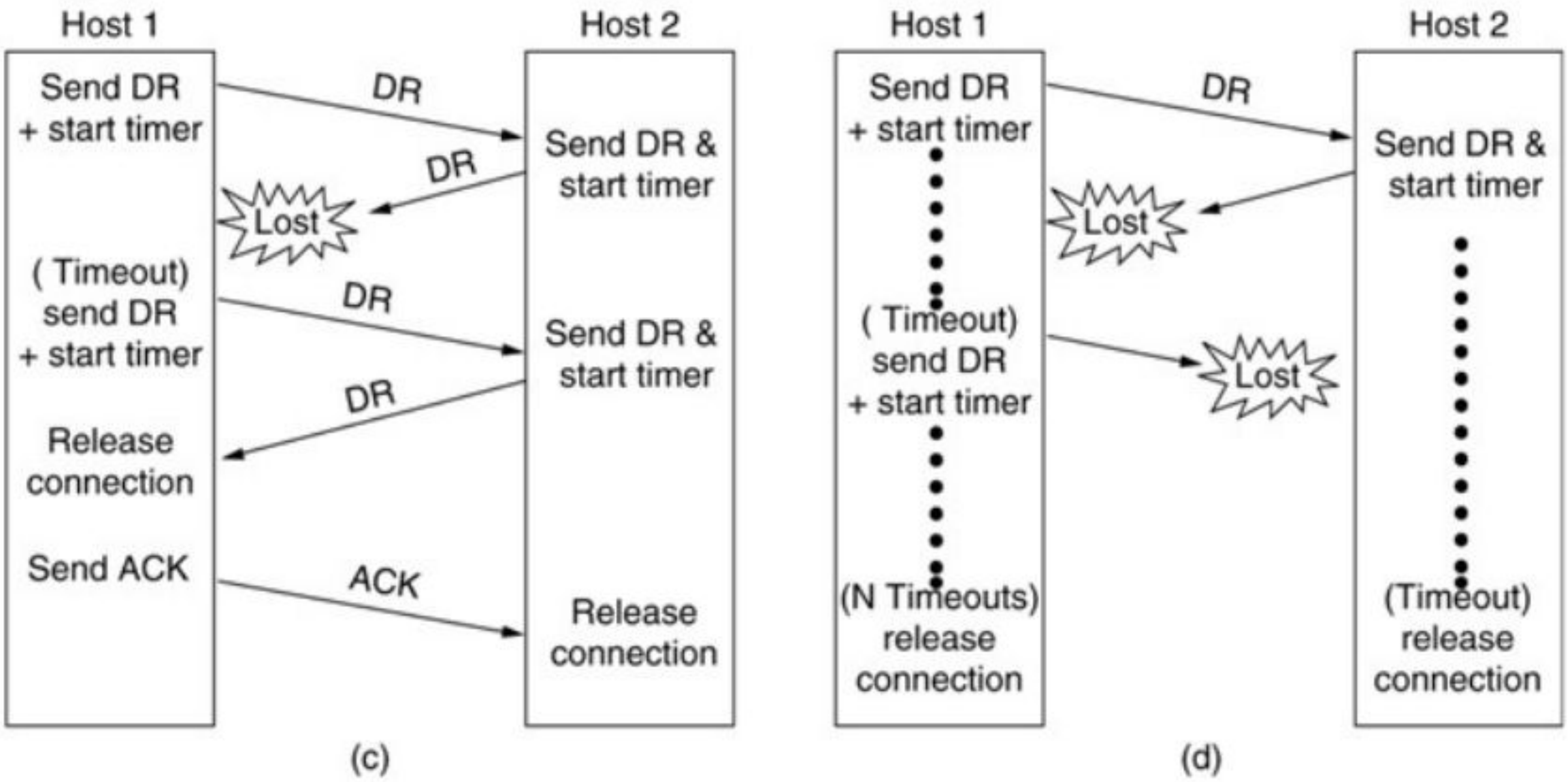
The two-army problem.

Connection Release (3)



Four protocol scenarios for releasing a connection. (a) Normal case of a three-way handshake. (b) final ACK lost.

Connection Release (4)



(c) Response lost. (d) Response lost and subsequent DRs lost.

Flow Control and Buffering

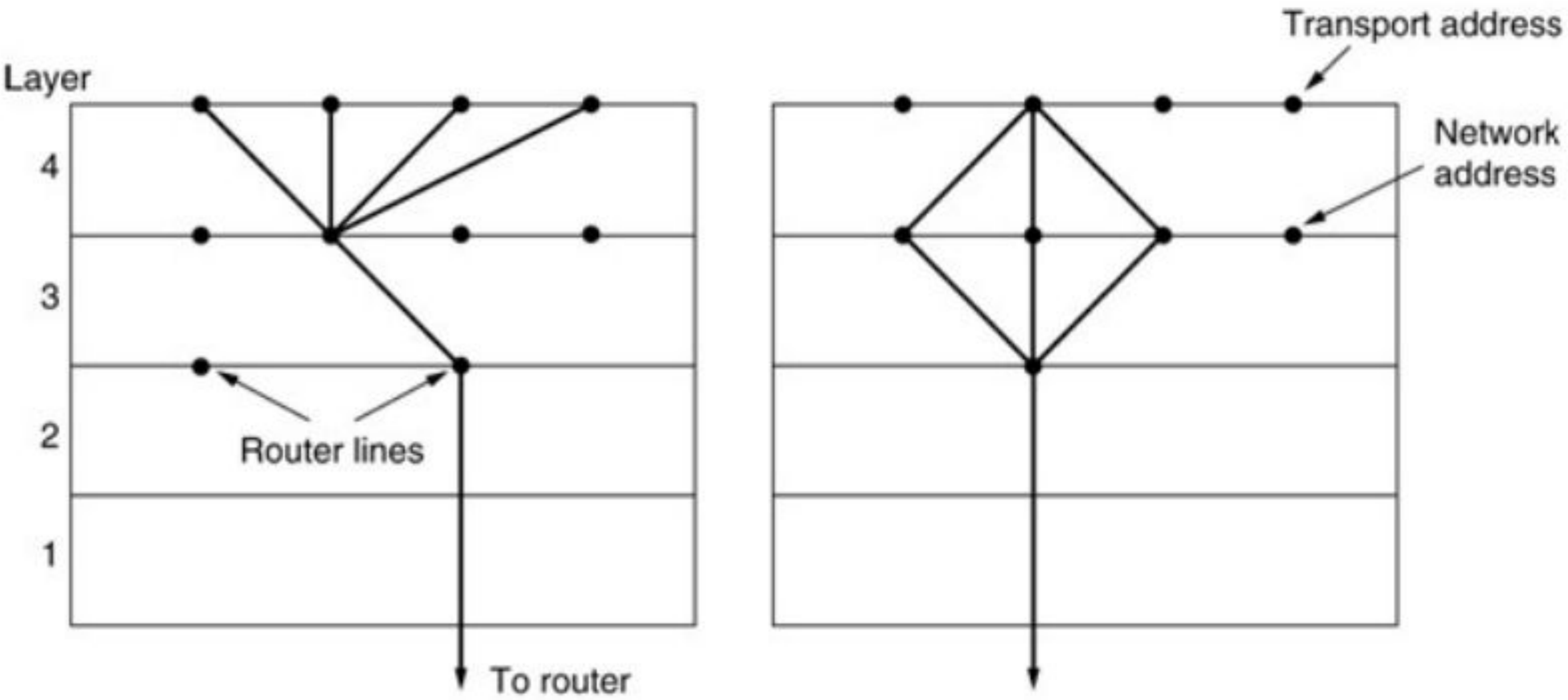
	A	Message	B	Comments
1	→	< request 8 buffers>	→	A wants 8 buffers
2	←	<ack = 15, buf = 4>	←	B grants messages 0-3 only
3	→	<seq = 0, data = m0>	→	A has 3 buffers left now
4	→	<seq = 1, data = m1>	→	A has 2 buffers left now
5	→	<seq = 2, data = m2>	• • •	Message lost but A thinks it has 1 left
6	←	<ack = 1, buf = 3>	←	B acknowledges 0 and 1, permits 2-4
7	→	<seq = 3, data = m3>	→	A has 1 buffer left
8	→	<seq = 4, data = m4>	→	A has 0 buffers left, and must stop
9	→	<seq = 2, data = m2>	→	A times out and retransmits
10	←	<ack = 4, buf = 0>	←	Everything acknowledged, but A still blocked
11	←	<ack = 4, buf = 1>	←	A may now send 5
12	←	<ack = 4, buf = 2>	←	B found a new buffer somewhere
13	→	<seq = 5, data = m5>	→	A has 1 buffer left
14	→	<seq = 6, data = m6>	→	A is now blocked again
15	←	<ack = 6, buf = 0>	←	A is still blocked
16	• • •	<ack = 6, buf = 4>	←	Potential deadlock

Dynamic buffer allocation. Buffer allocation info travels in separate TPDUs

The arrows show the direction of transmission. '…' indicates a lost TPDU.

Potential deadlock if control TPDUs are not sequenced or timed out

Multiplexing



(a) (b)

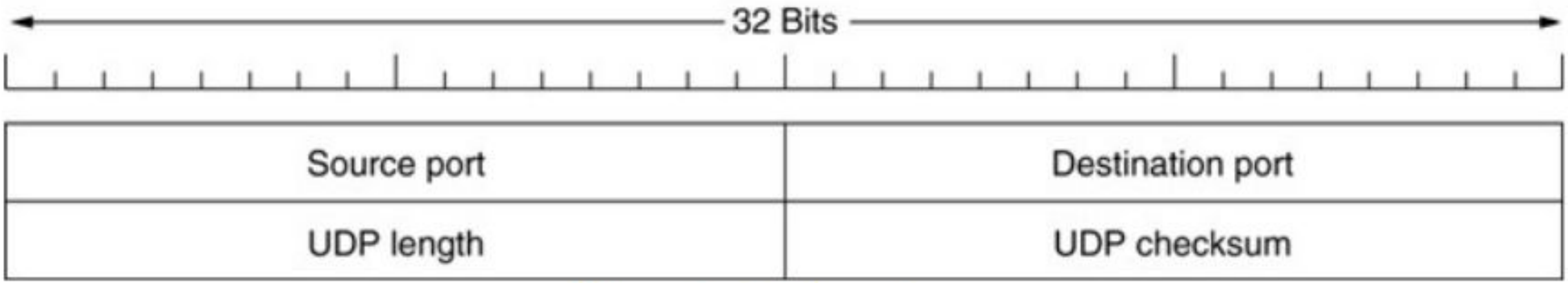
Upward multiplexing.

Downward multiplexing. Used to increase the bandwidth, e.g., two ISDN connections of 64 kbps each yield 128 kbps bandwidth.

The Internet Transport Protocols: UDP

- Introduction to UDP
- Remote Procedure Call
- The Real-Time Transport Protocol

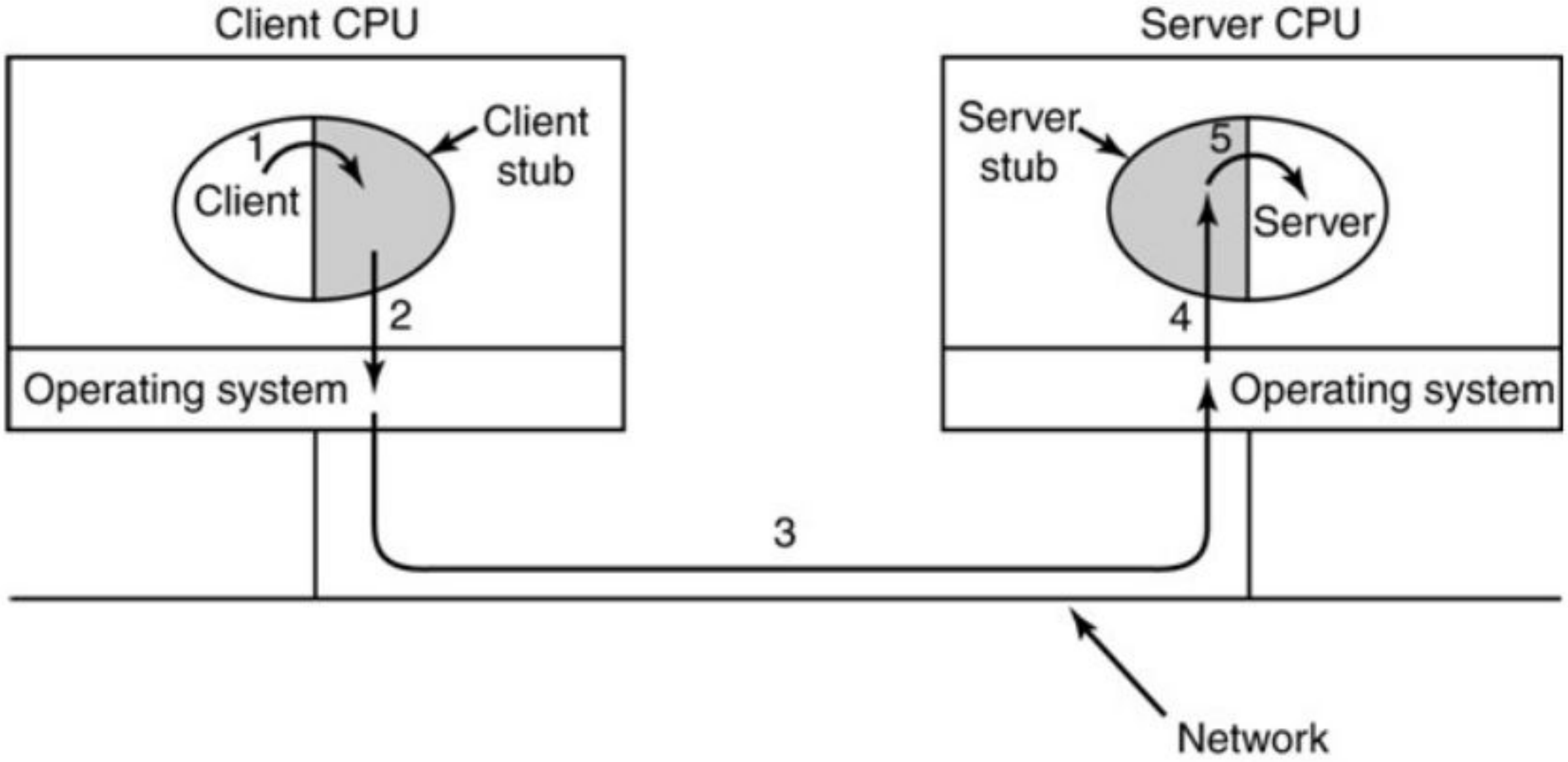
Introduction to UDP



The UDP header.

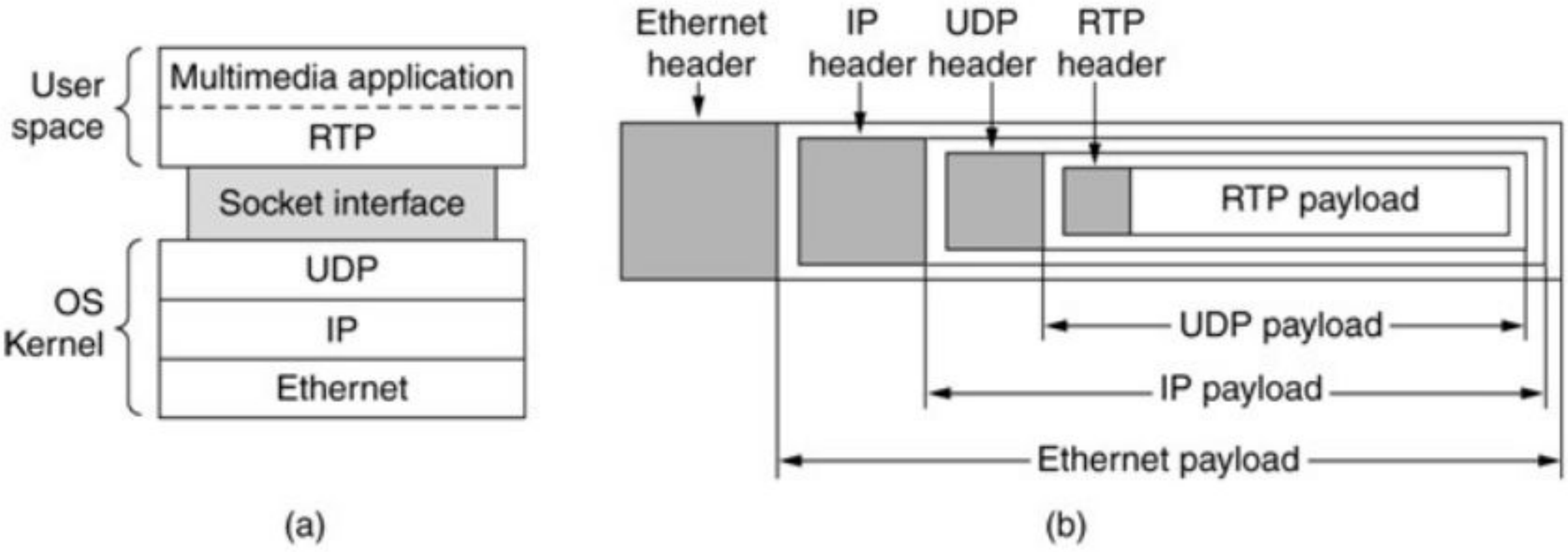
UDP only provides TSAPs (ports) for applications to bind to. UDP does not provide reliable or ordered service. The checksum is optional.

Remote Procedure Call



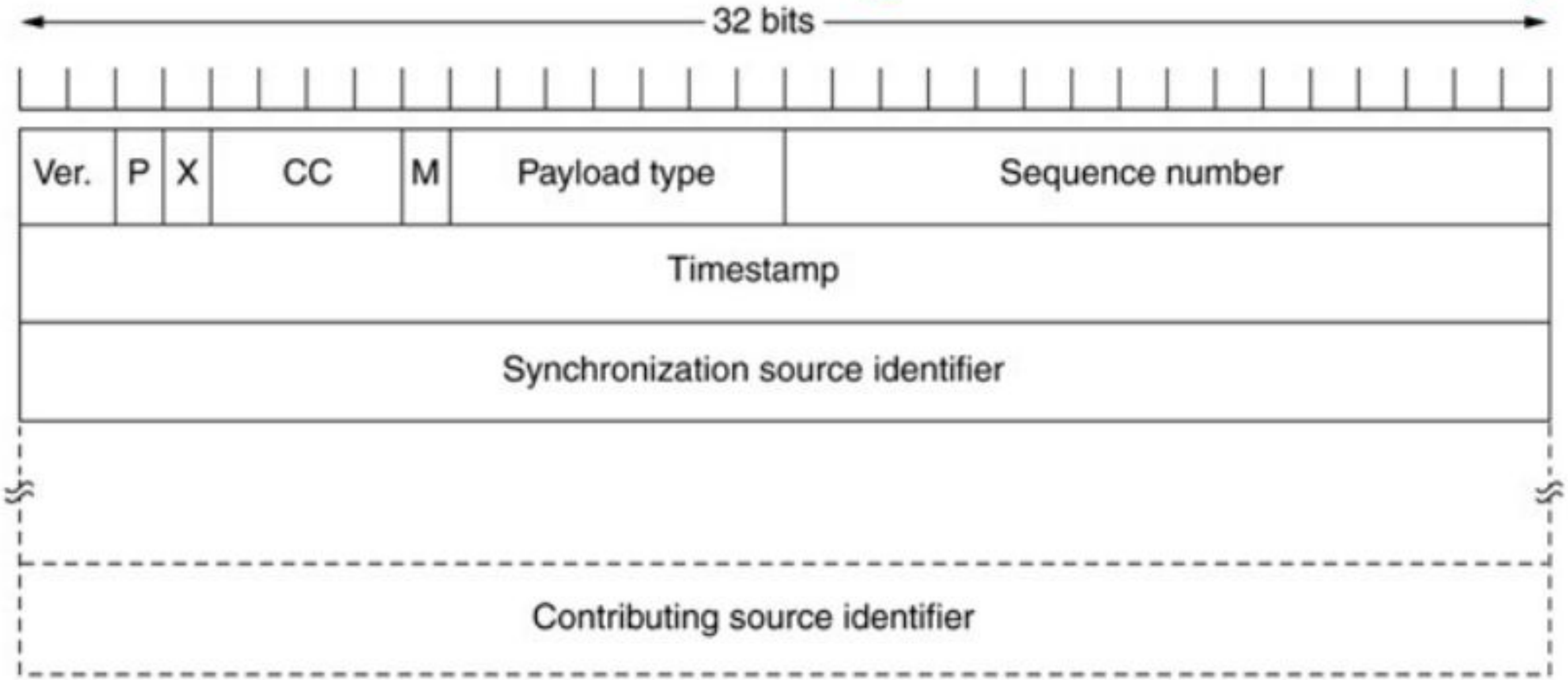
Steps in making a remote procedure call. The stubs are shaded.

The Real-Time Transport Protocol



(a) The position of RTP in the protocol stack. (b) Packet nesting.

The Real-Time Transport Protocol (2)



The RTP header. X indicated the presence of an extension header.

CC says how many contributing sources are present (0 to 15).

Syn. Source Id. tells which stream the packet belongs to.

For feedback information is used an associated protocol called RTCP (Real Time Control Protocol)

The Internet Transport Protocols: TCP

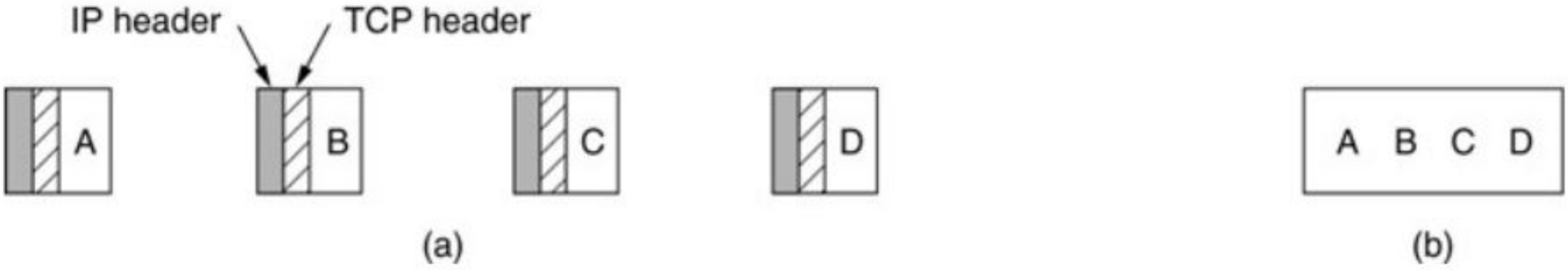
a) Introduction to TCP
b) The TCP Service Model
c) The TCP Protocol
d) The TCP Segment Header
e) TCP Connection Establishment
f) TCP Connection Release
g) TCP Connection Management Modeling
h) TCP Transmission Policy
i) TCP Congestion Control
j) TCP Timer Management
k) Wireless TCP and UDP
l) Transactional TCP

The TCP Service Model

Port	Protocol	Use
21	FTP	File transfer
23	Telnet	Remote login
25	SMTP	E-mail
69	TFTP	Trivial File Transfer Protocol
79	Finger	Lookup info about a user
80	HTTP	World Wide Web
110	POP-3	Remote e-mail access
119	NNTP	USENET news

Some assigned ports.

The TCP Service Model (2)



(a) Four 512-byte segments sent as separate IP datagrams.

(b) The 2048 bytes of data delivered to the application in a single READ CALL.

TCP Service Model (3)

All TCP connections are full-duplex and point-to-point.

TCP provides a byte stream. i.e it does not preserve message boundaries

At sender TCP may immediately send or buffer data at its discretion.

Sender can use a PUSH flag to instruct TCP not to buffer the send.

Sender can use URGENT flag to have TCP send data immediately and have the receiver TCP signal the receiver application that there is data to be read.

Some TCP features

Every byte has its own 32 bit sequence number.

Sending and receiving entities exchange data in segments

Each segment is the 20 byte header and data (total up to 64K)

TCP may aggregate multiple writes into one segment or split one write into several segments.

A segment size if the smaller of either 64K or the MTU of the network layer (MTU of Ethernet is about 1500 bytes)

A segment must fit in a single IP payload.