Chapter 6

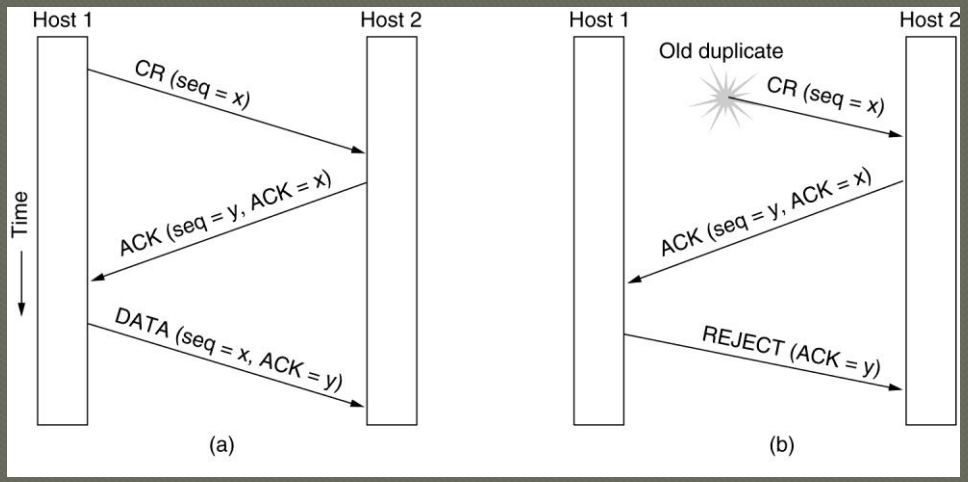
The Transport Layer

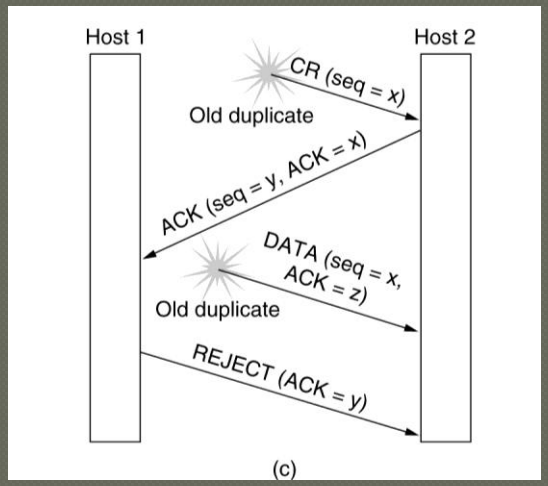
The Transport Layer

- Services Provided to the Upper Layers
- Transport Service Primitives
- Berkeley Sockets
- The elements of the Transport Protocol
  - Addressing
  - Connection Establishment
    - 3-way handshaking

  (Today)
  - Connection Release
  - Flow Control and Buffering
  - Multiplexing
  - Crash Recovery
- The UDP Protocol

Connection Establishment (review)





Sequence Number and clock time are used to avoid that old requests might be processed as new.

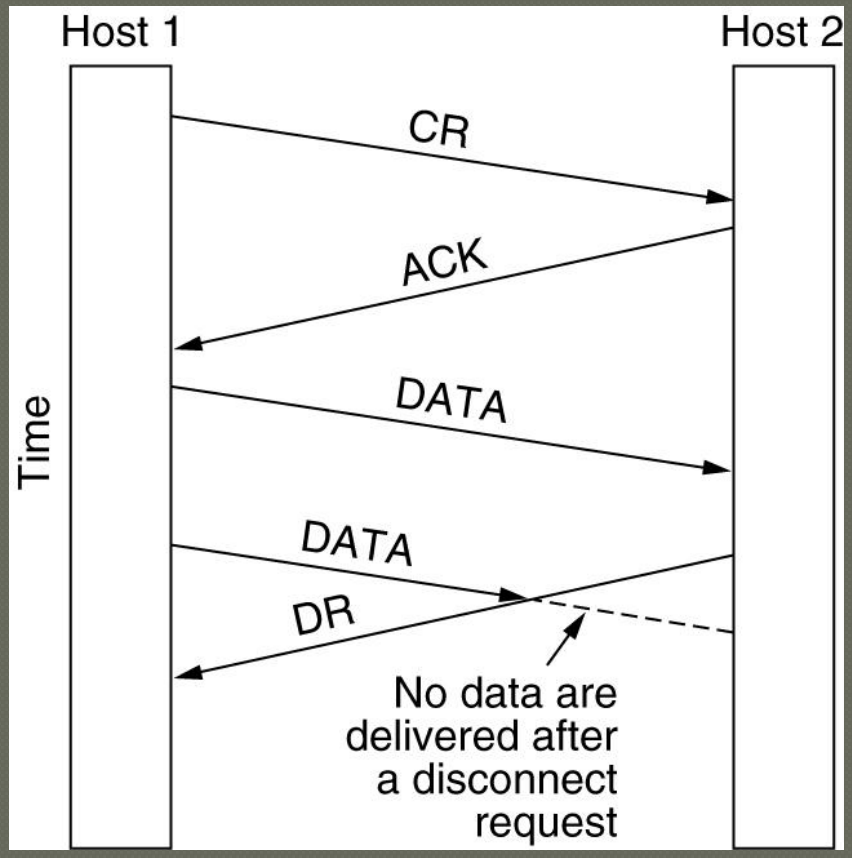
Three protocol scenarios for establishing a connection using a three-way handshake. CR denotes CONNECTION REQUEST.

(a) Normal operation,

(b) Old CONNECTION REQUEST appearing out of nowhere.

(c) Duplicate CONNECTION REQUEST and duplicate ACK.

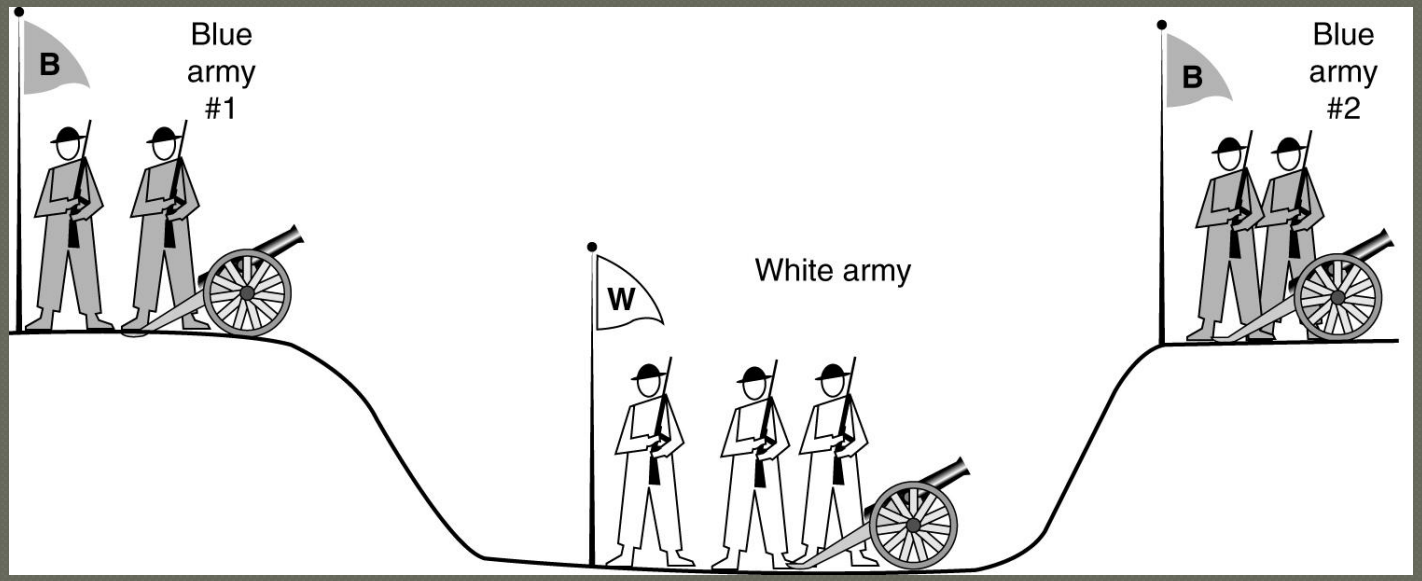
Connection Release



Abrupt disconnection with loss of data.

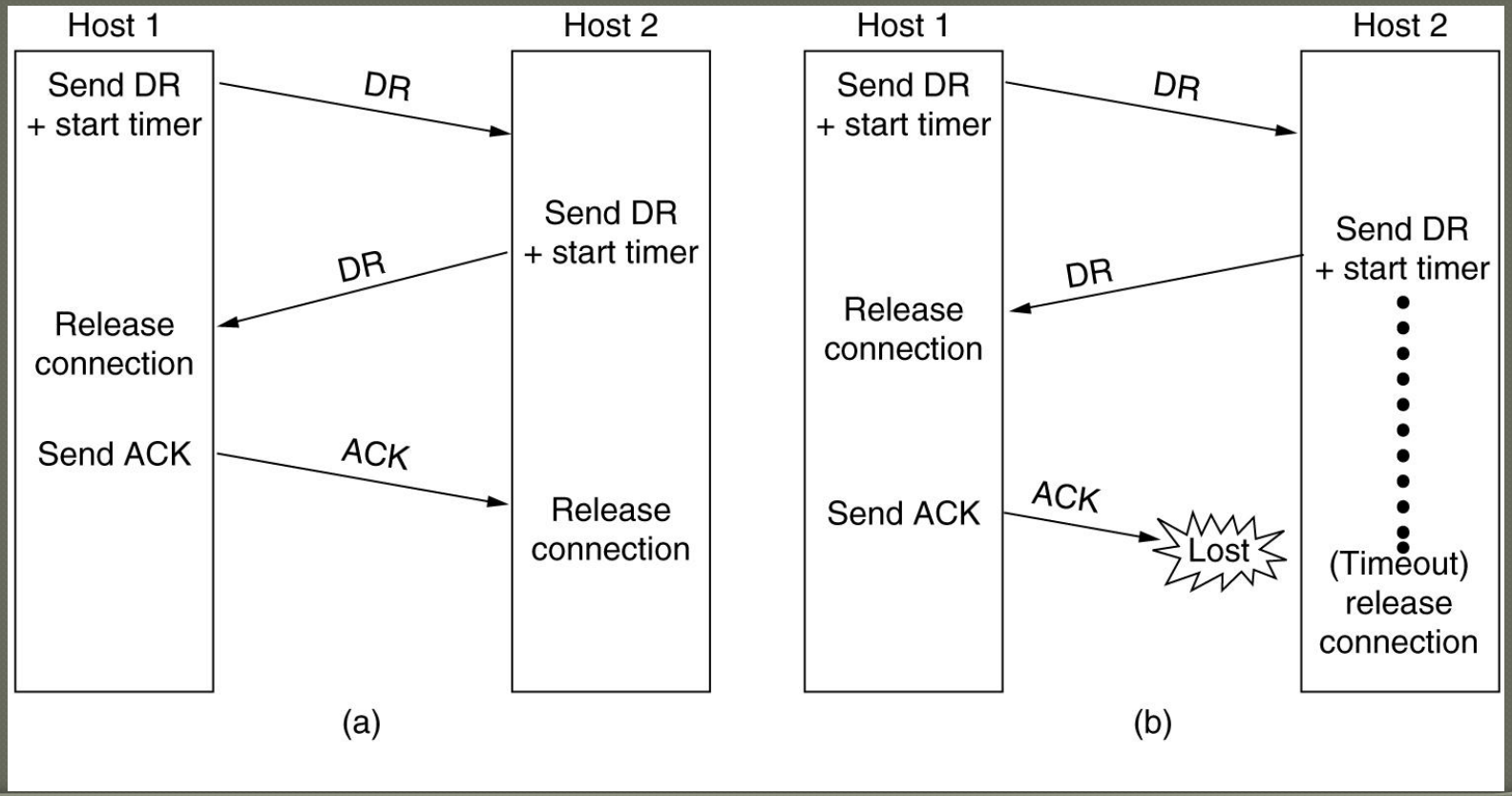
Connection Release (2)

The two-army problem.

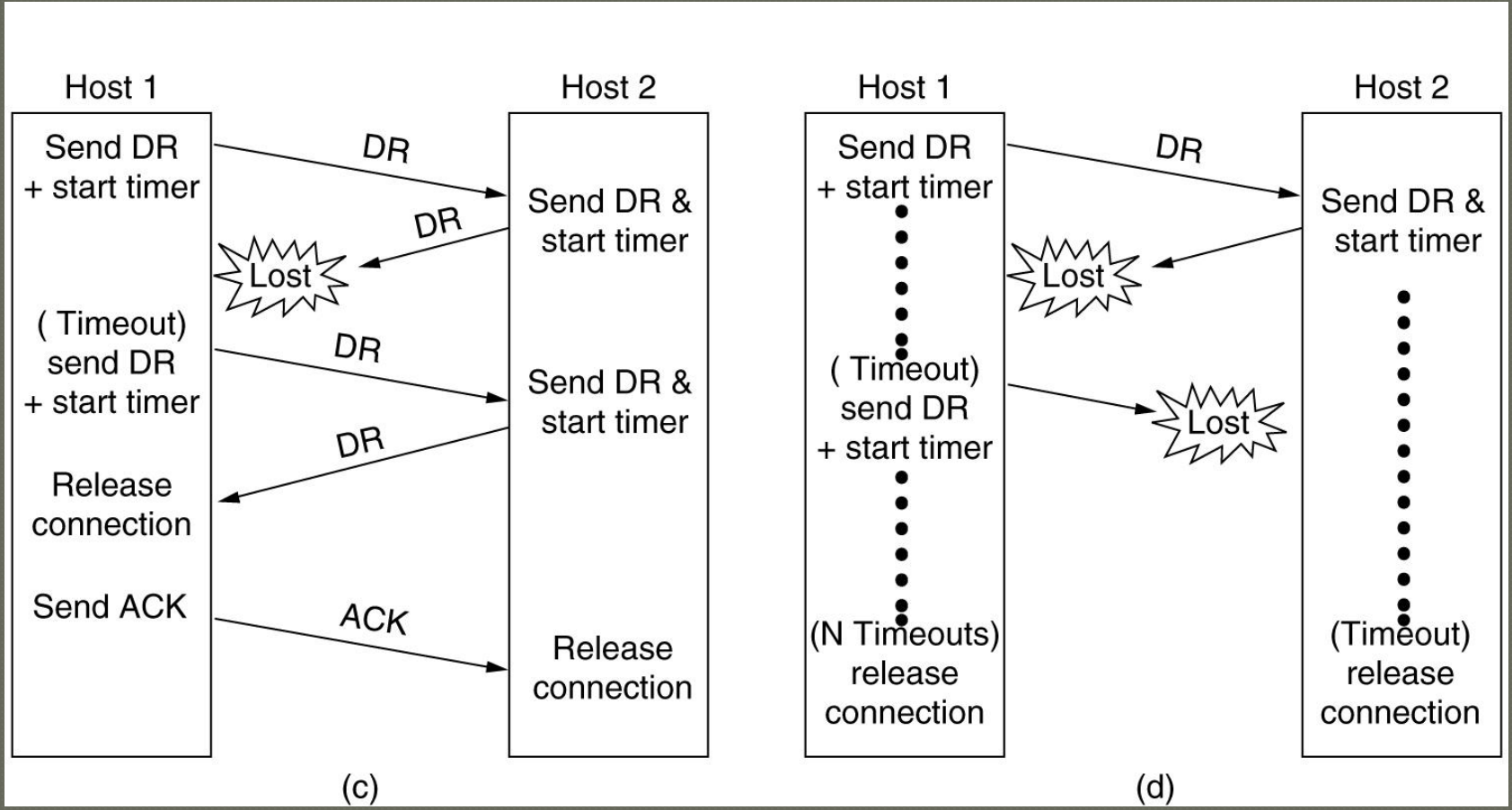


Connection Release (3)

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

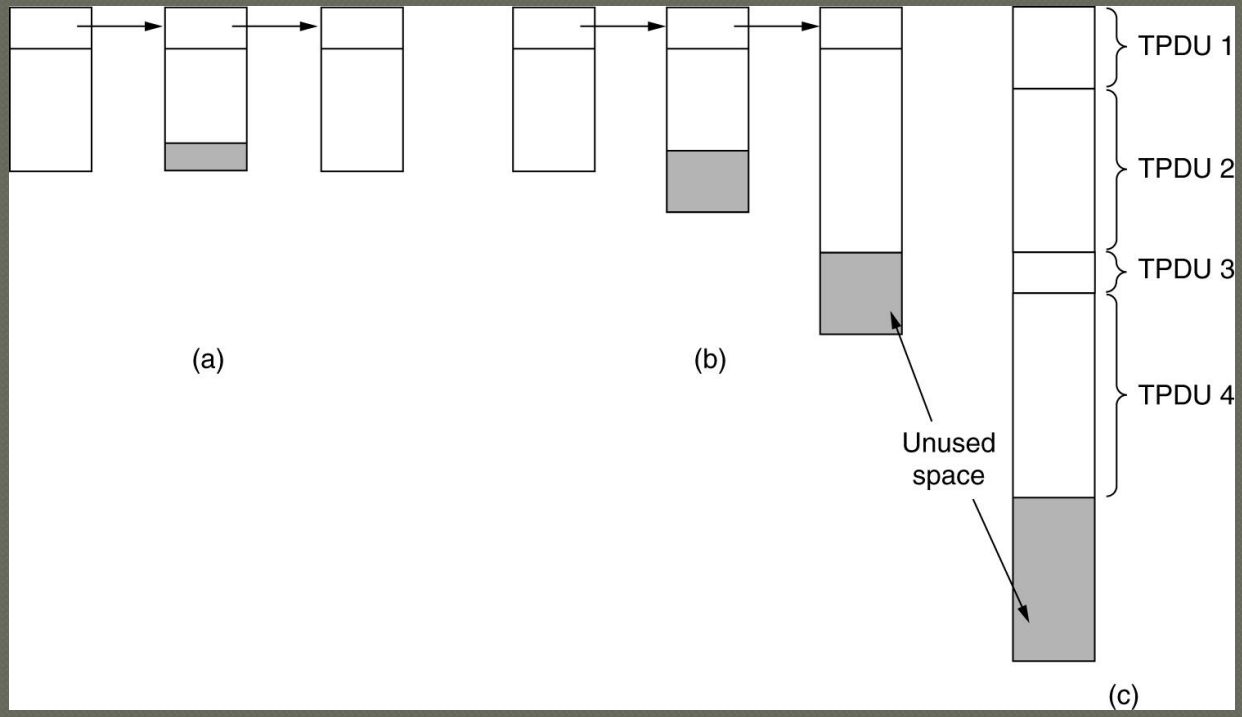


Connection Release (4)



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

Flow Control and Buffering



(a) Chained fixed-size buffers. (b) Chained variable-sized buffers. (c) One large circular buffer per connection.

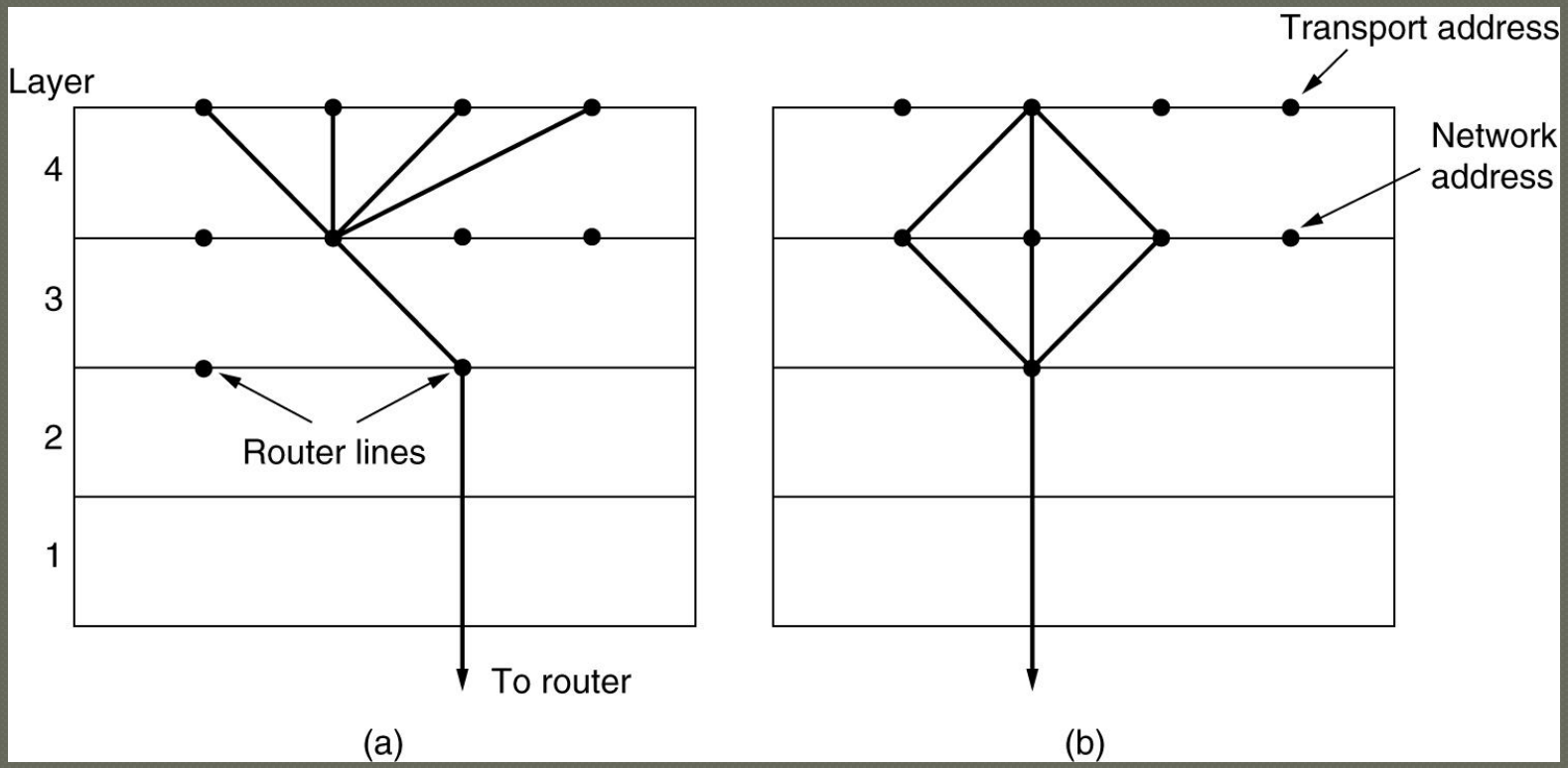
Flow Control and Buffering (2)

Dynamic buffer allocation. The arrows show the direction of transmission. A "…" indicates a lost TPDU.

	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

Multiplexing

(a) Upward multiplexing. (b) Downward multiplexing.



Crash Recovery

Different combinations of client and server strategy.

Strategy used by sending host	Strategy used by receiving host: First ACK, then write — AC(W)	AWC	C(AW)	First write, then ACK — C(WA)	W AC	WC(A)
Always retransmit	OK	DUP	OK	OK	DUP	DUP
Never retransmit	LOST	OK	LOST	LOST	OK	OK
Retransmit in S0	OK	DUP	LOST	LOST	DUP	OK
Retransmit in S1	LOST	OK	OK	OK	OK	DUP

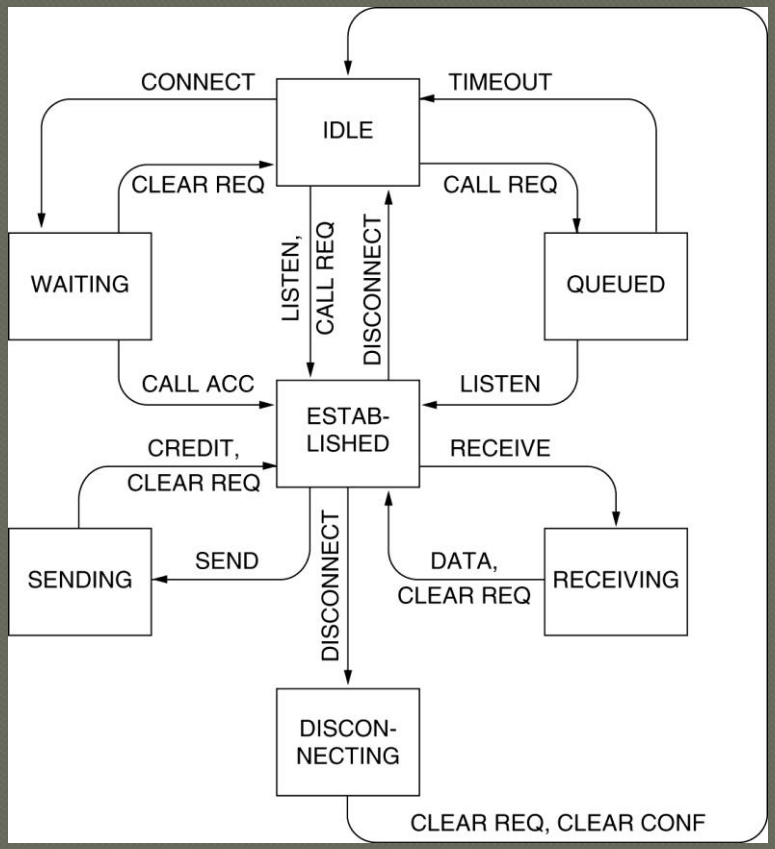
OK = Protocol functions correctly
DUP = Protocol generates a duplicate message
LOST = Protocol loses a message

The Example Transport Entity (2)

Each connection is in one of seven states:

1. Idle – Connection not established yet.
2. Waiting – CONNECT has been executed, CALL REQUEST sent.
3. Queued – A CALL REQUEST has arrived; no LISTEN yet.
4. Established – The connection has been established.
5. Sending – The user is waiting for permission to send a packet.
6. Receiving – A RECEIVE has been done.
7. DISCONNECTING – a DISCONNECT has been done locally.

The Example as a Finite State Machine (2)



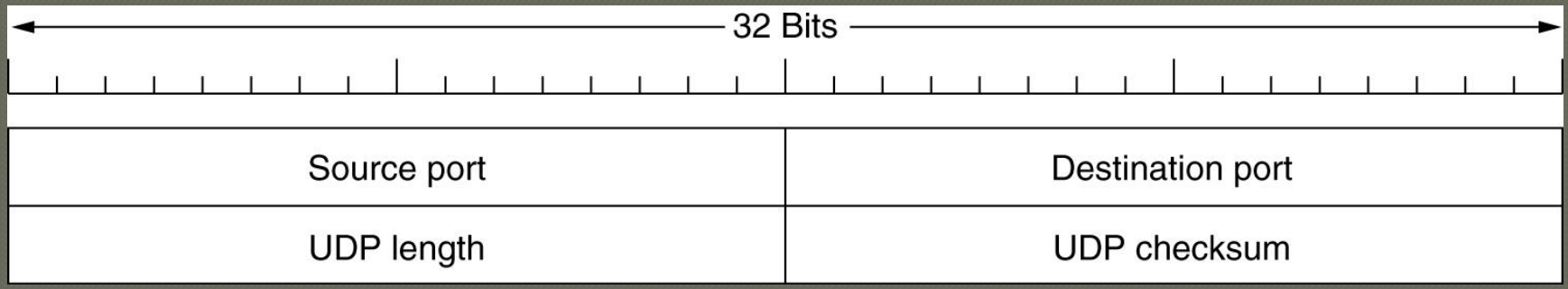
The example protocol in graphical form. Transitions that leave the connection state unchanged have been omitted for simplicity.

The Internet Transport Protocols: UDP

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

Introduction to UDP

The UDP header.



What does UDP **not** do?

1. No flow control
2. No error control or retransmission upon receipt of bad segment

Who does that? The user processes!

Introduction to UDP

What does UDP do?

1. Provides interface to the IP Protocol
2. Demultiplexes processes using ports

That's all!

Who uses it?

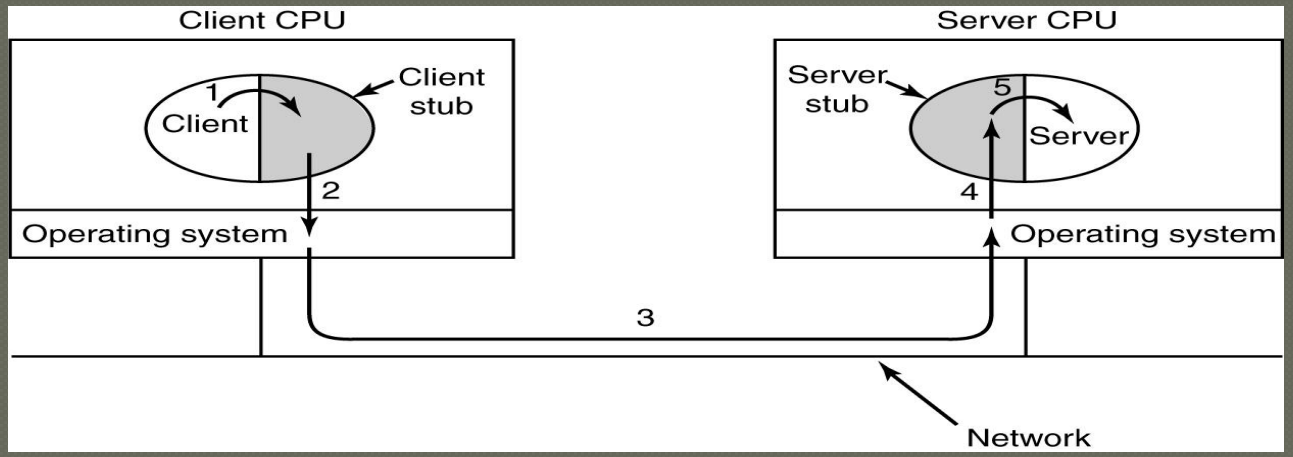
- Ex. DNS uses it to look up the IP address of a host sends a UPD packet to the DNS server which replies with the host's IP address.
- No connection – just two small messages.

Where is this useful?

1. In client-server situations for short requests
   - If request or reply is lost the client tries again.
2. In client-server Remote Procedure Calls
3. In Real-Time Multimedia Applications

Remote Procedure Call

- Request-Reply interaction implemented via Remote Procedure Call
  - RPC
  1. offers an alternative to socket (it hides them)
  2. makes program easier and more familiar to deal with.



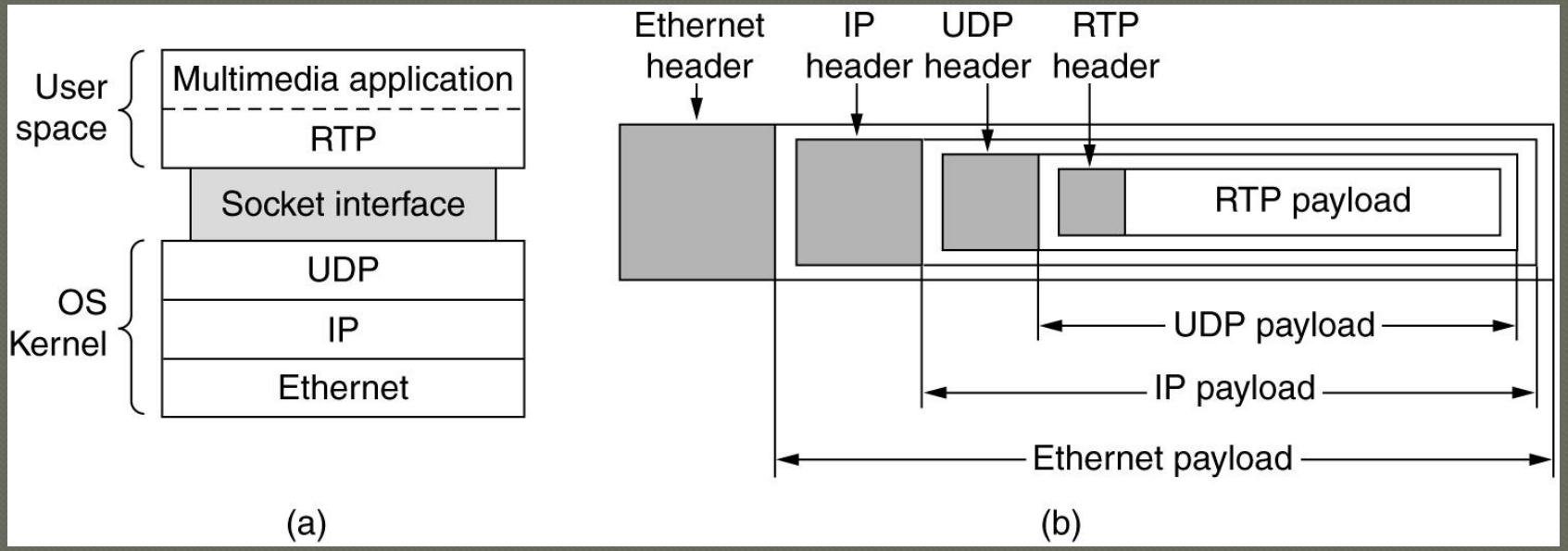
Steps in making a remote procedure call (stubs are shaded)

Remote Procedure Call (2)

- General copy-restore replaces the standard call-by-reference
- Unfortunately:
  1. Pointers to complex data structures have problems
  2. Legal commands in weakly-typed languages (see C) cannot work.
  3. The types of the parameters are not always identifiable
  4. Global variable cannot be used since they are not shared remotely.
- Restrictions can make RPC work well.
- RPC generally uses UDP
  - TCP is used
    - when parameters and results are larger than UDP max size packet
    - when operation requested cannot be repeated safely (i.e. counter increment)

The Real-Time Transport Protocol

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



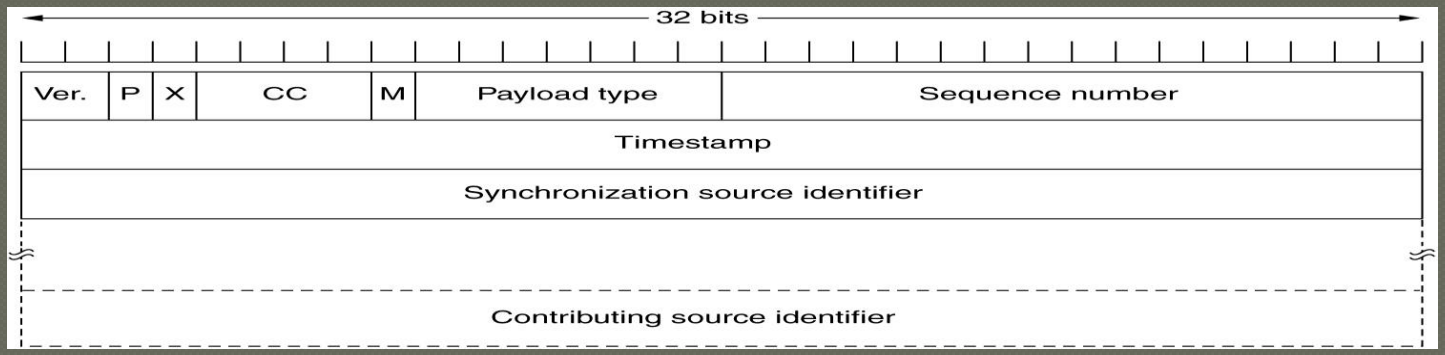
The Real-Time Transport Protocol (2)

- Where does RTP belong to?
  - To the application layer since it runs in the user space?
  - To the transport layer since it provides a generic independent protocol with transport facilities?

  Let's say that is a transport protocol implemented in the application layer.

- Unicast and multicast of UDP packets are permitted.
- RTP streams are numbered
  - If a packet is missing, interpolation is performed (not retransmission!)
  - No flow control, no error control, no ack, no retransmission.

The RTP header



Ver: we are at 2 now.

P: packet padded to a multiple of 4 bytes

Last padding byte indicates how many bytes were added.

X: extension header for future requirements

M: appl. specific marker bit

Payload Type:

ex. Mp3, uncompressed 8-bit audio, ect

- Seq. Number: counter
- Timestamp: time
  - used to reduce jitter and for synch.
- Synch Source Id: which stream packet belongs to
  - Used to multiplex and demultiplex stream onto a single data stream UDP packet.
- Contributing Source Id: when mixer is the synch resource, the sources are listed here.

Next Time

- RTP Protocol