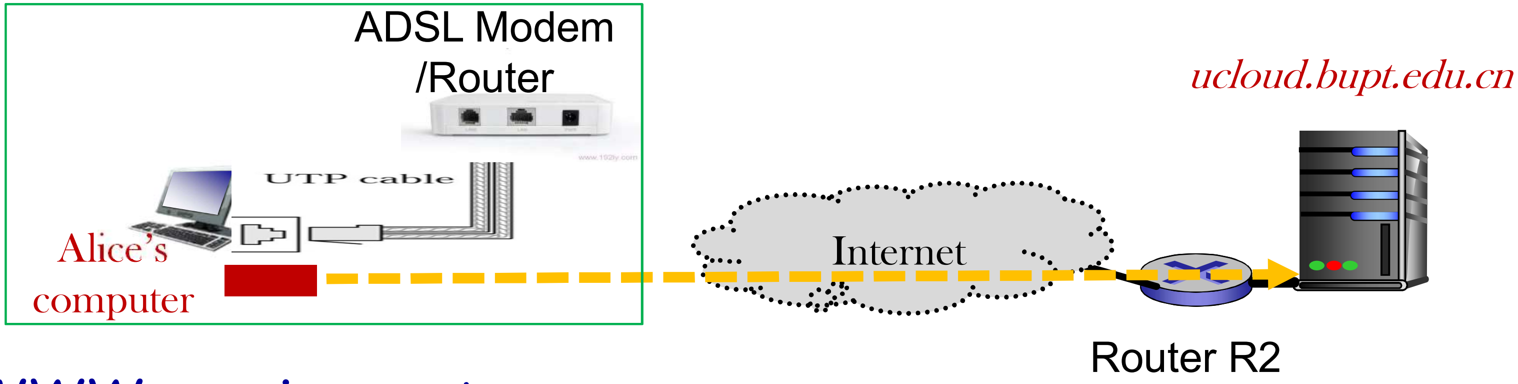
Chapter 6
The Transport Layer

(Around 6-7 hours)

The task: Uploading homework image to 教学云平台



- WWW requirements
  - End-to-end delivery
  - QoS: Reliable
- What IP provides?
  - Host-to-host delivery
  - Best effort: NOT reliable
- How to provide reliable transmission over IP?

→ Transport layer

Motivation of Transport Layer

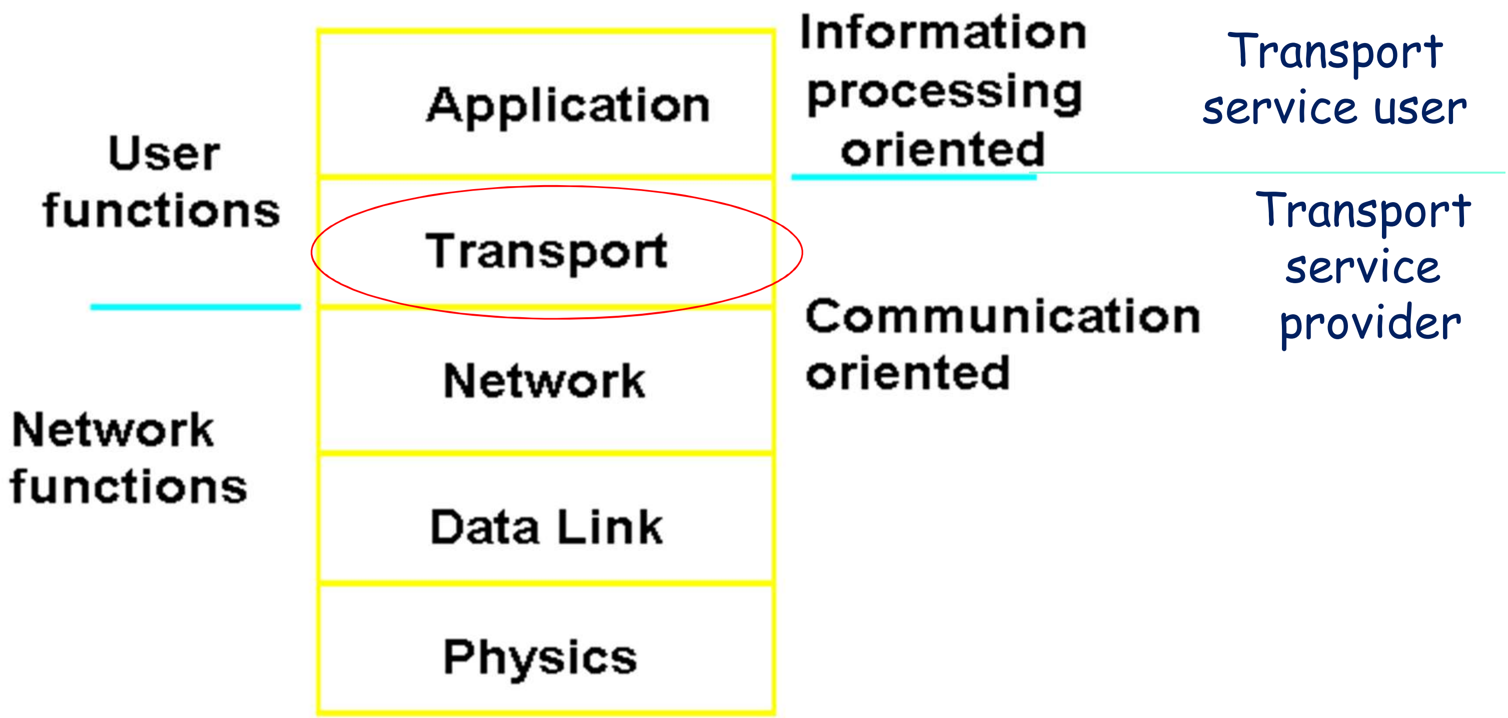
- IP provides a weak, but efficient service model (*best-effort*)
  - Packets may be delayed, dropped, reordered, duplicated
  - Packets have limited size（MTU）
- IP packets are addressed to a host
  - How to decide which application gets which packets?
- How should hosts send packets into the network?
  - Too fast may overwhelm the network
  - Too slow is not efficient

Outline

- *Function and Services to Transport Layer*
- Elements of Transport Protocols
  - Addressing
  - Connection Establishment and Release
  - Flow Control and Buffering
  - Congestion Control
- The Transport Layer in Internet
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)

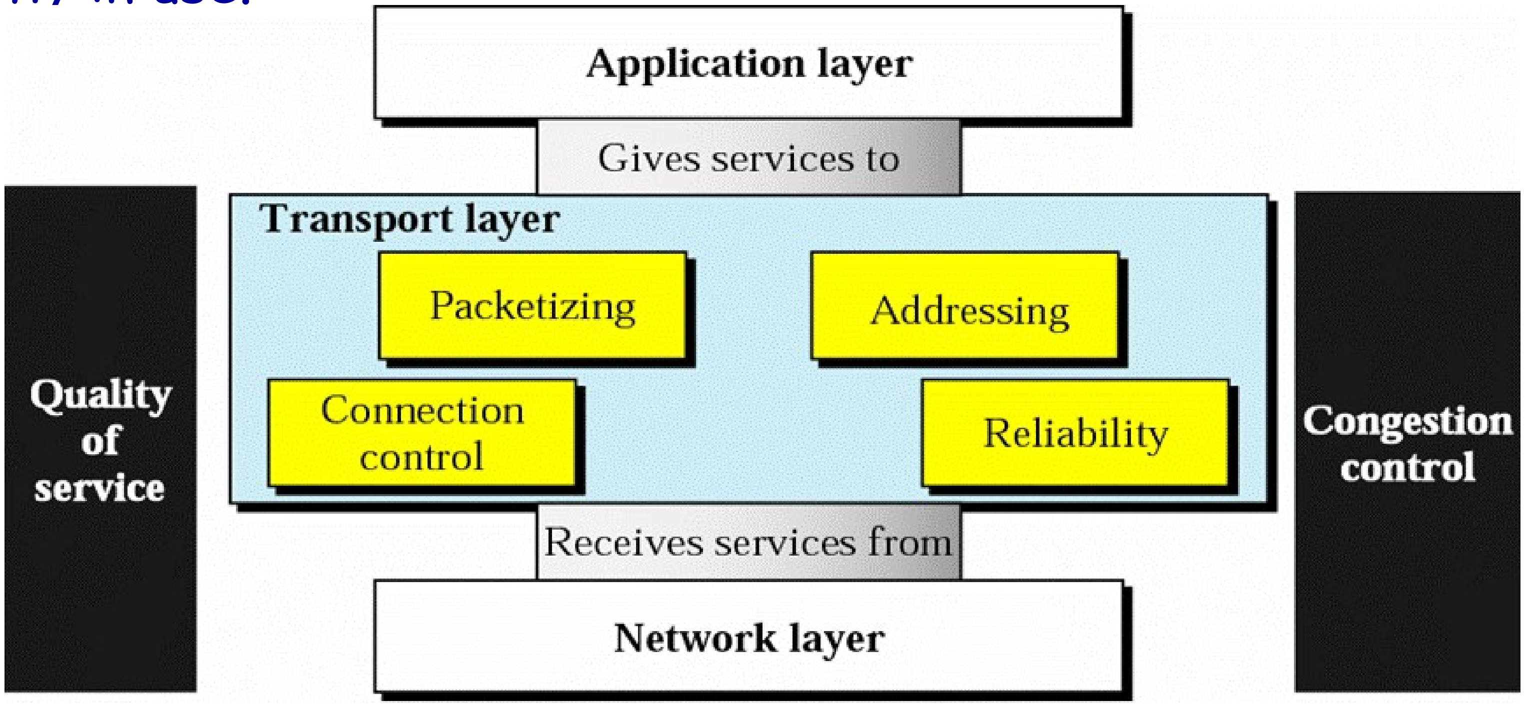
Position of Transport Layer

- Heart of the whole protocol hierarchy.



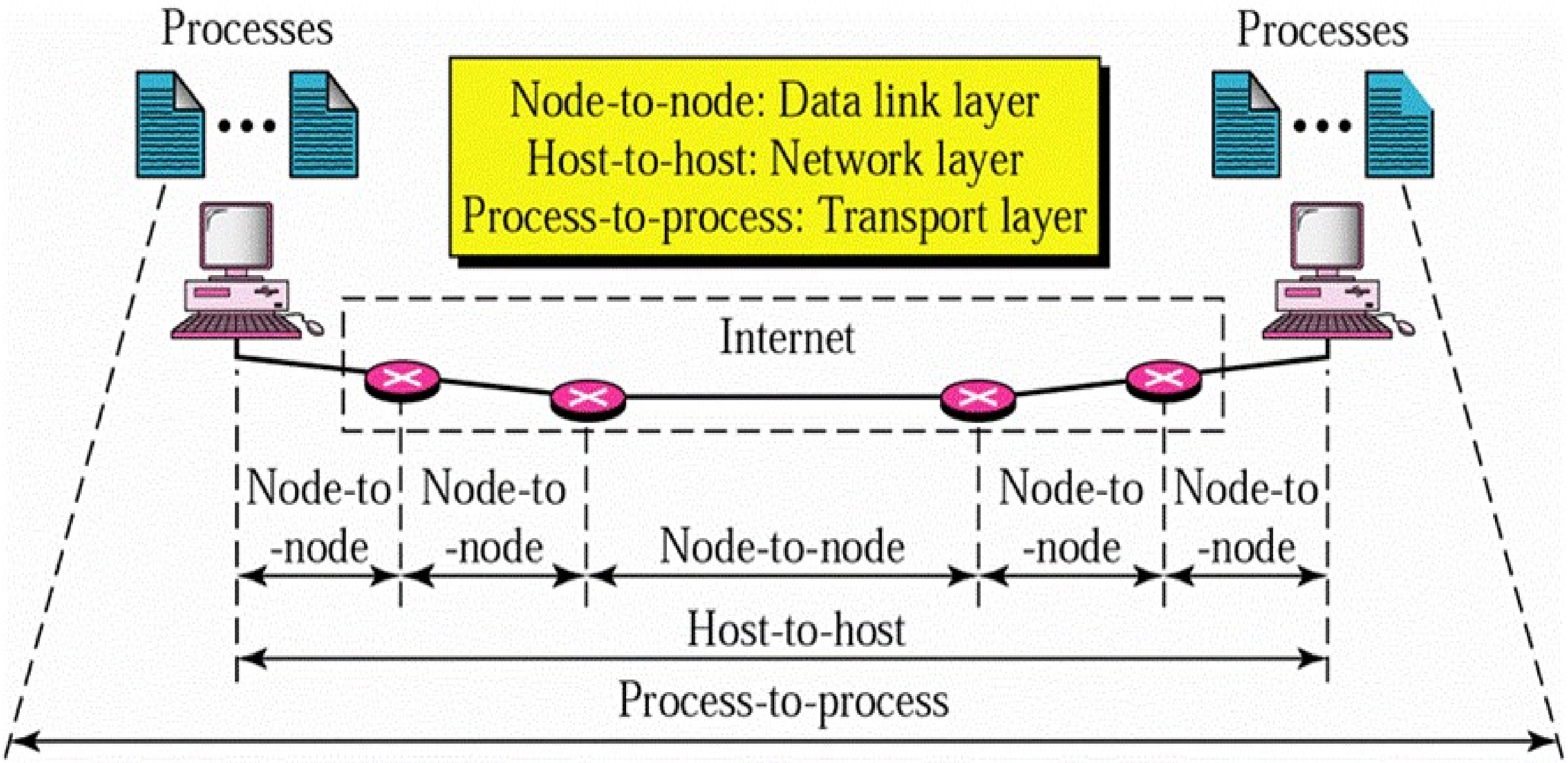
Functions of Transport Layer

- Providing efficient, reliable, cost-effective data transport from the source to the destination, independently of the physical network or networks currently in use.



[Forouzan]

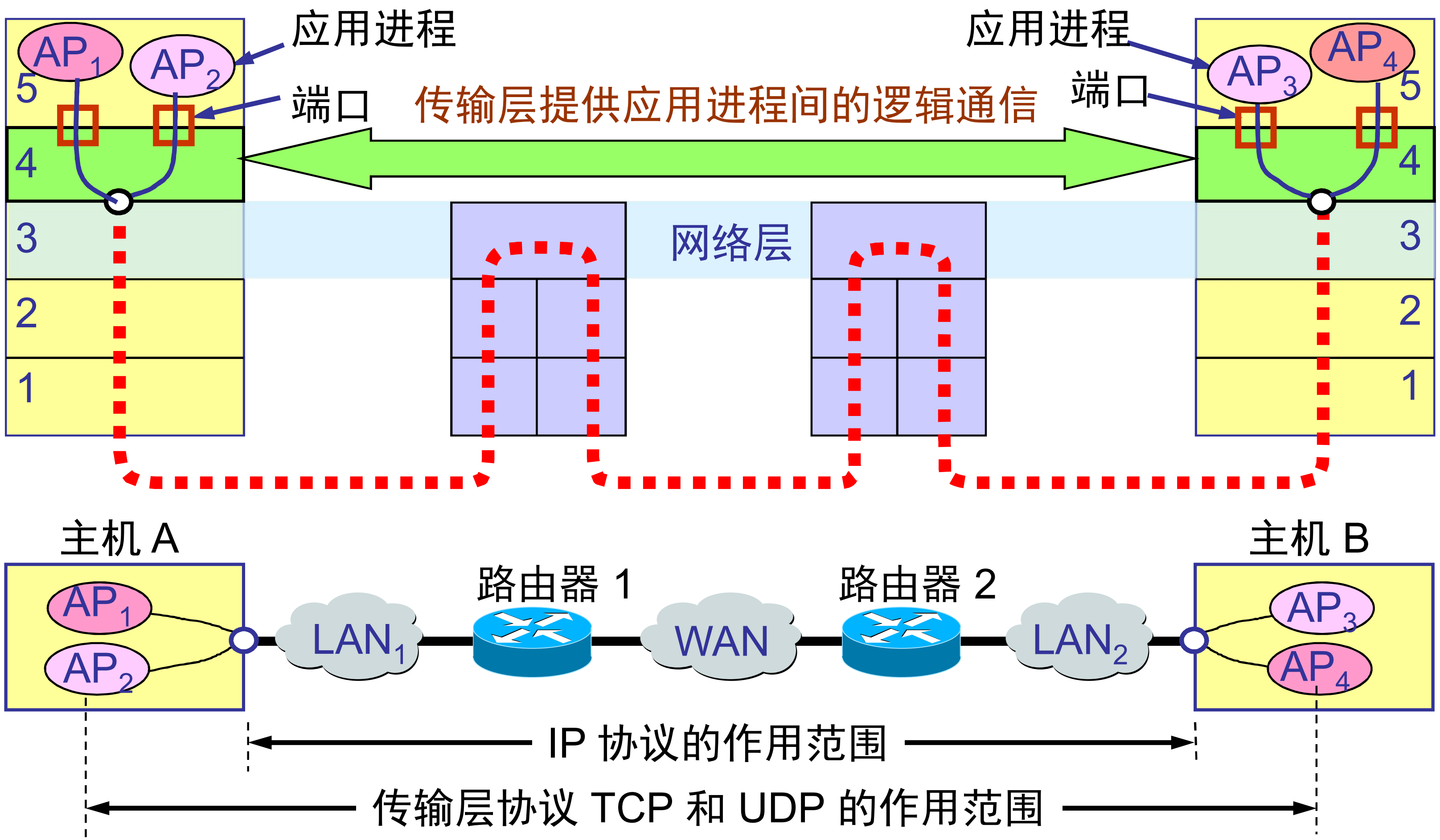
Process-to-Process Delivery



进程-进程

[Forouzan]

Scope of Transport Layer



[谢]

Services of Transport Layer

- Masking the diversity of communication subnets and providing a common interface to the upper layer
- Providing multiple SAPs(Service Access Points) sharing one network link
- Connection-oriented service vs. connectionless service

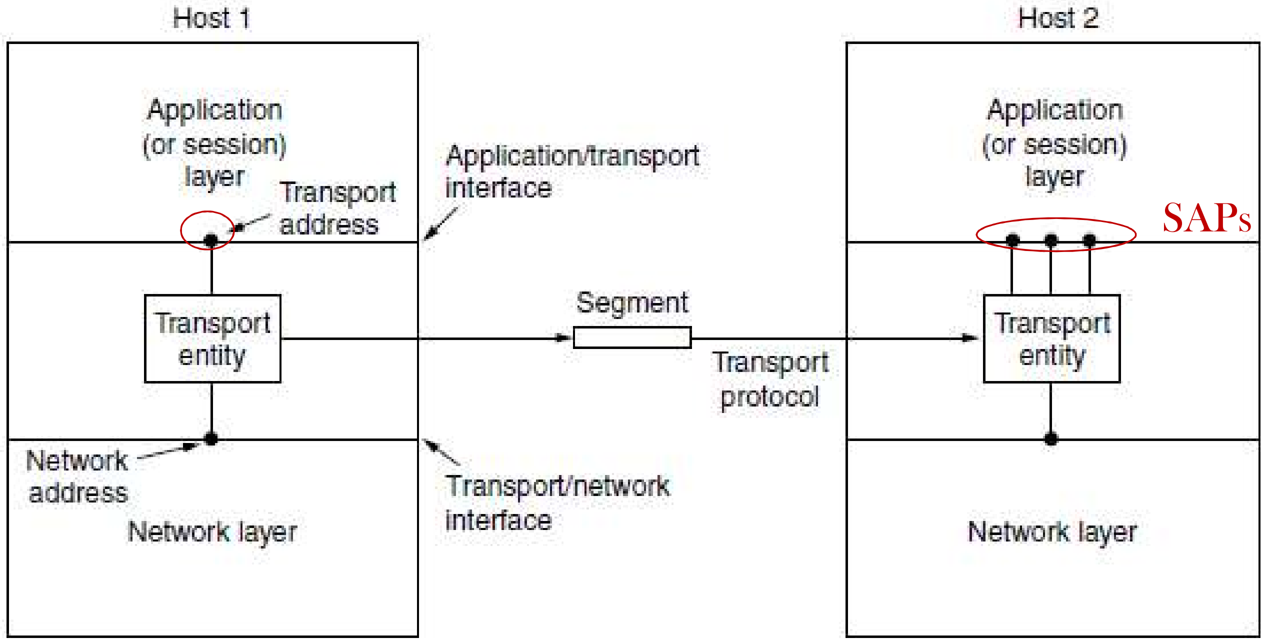
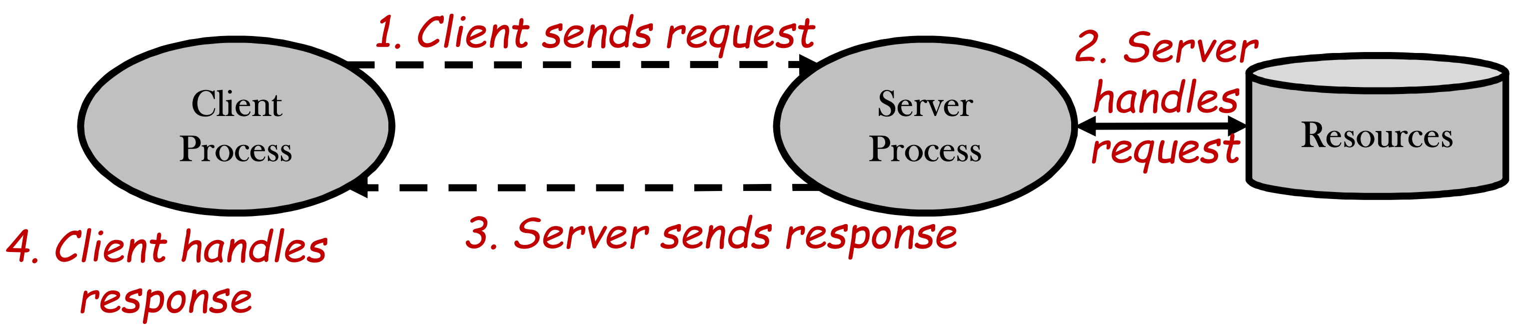


Fig. 6-1

Client-Server Model

- Used by almost all network applications
- Server
  - Manages some resources and provides service by manipulating resources for clients
  - Begins execution first
  - Waits passively at prearranged location
- Client
  - Begins execution later
  - Actively initiates contact with a server



How does Application layer talk with Transport layer?

- API：提供给应用进程的接口
- Example of a simple connection-oriented service

Fig. 6-2

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.	Request a release of the connection

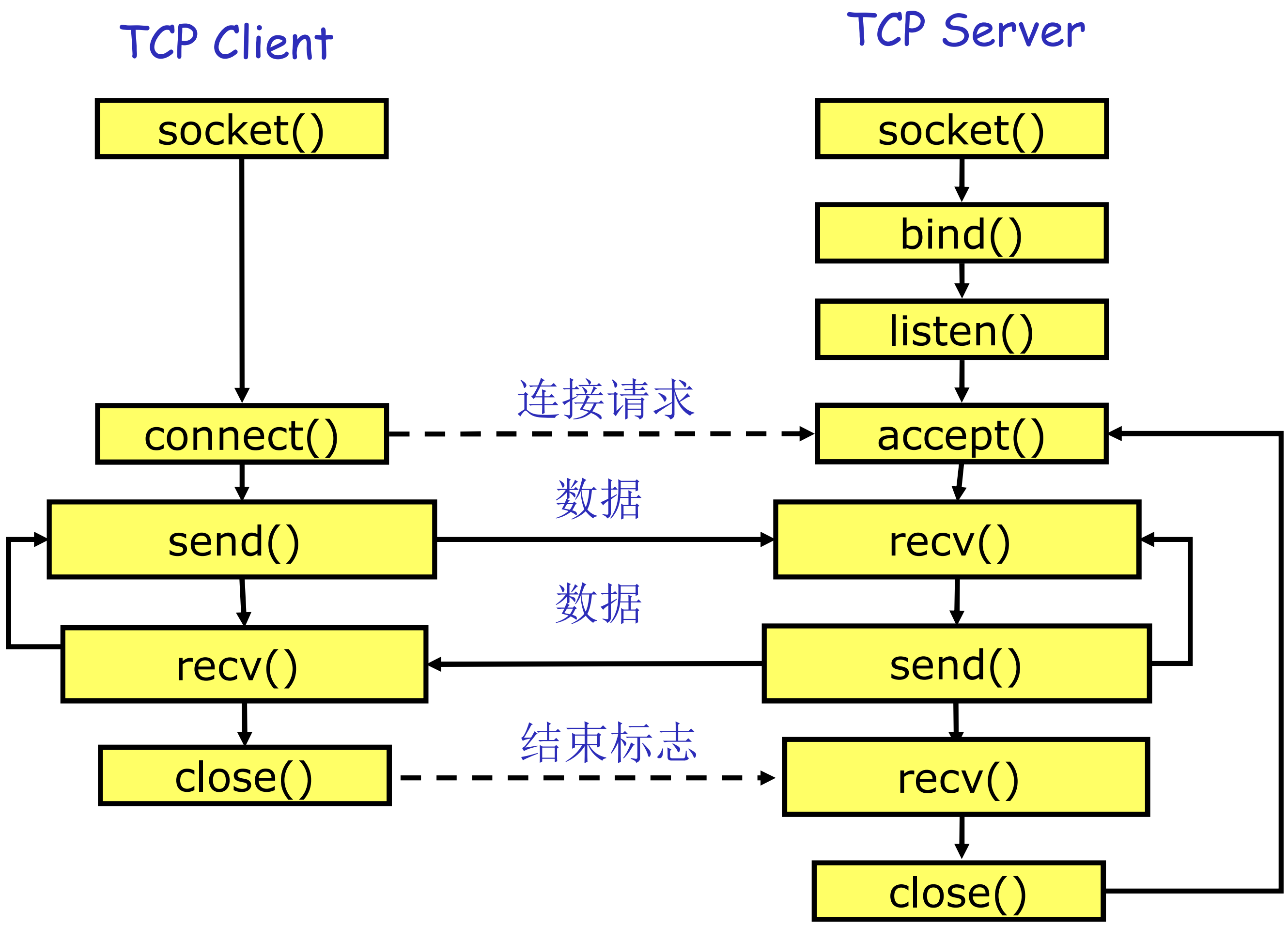
Berkeley Sockets（套接字）

- Widely used API in Internet

Primitive	Meaning
SOCKET	Create a new communication endpoint
BIND	Associate a local address with a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Passively establish an incoming connection
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

Figure 6-5. The socket primitives for TCP.

基于TCP的Socket程序流程



Outline

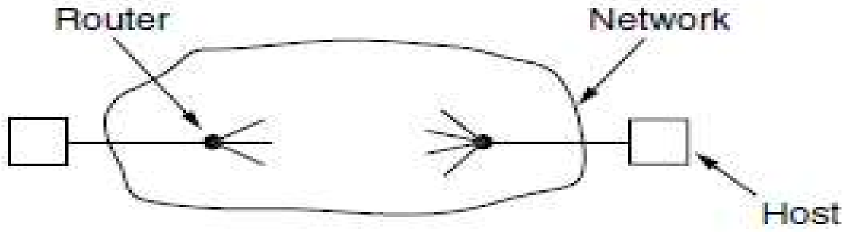
- Function and Services to Transport Layer
- *Elements of Transport Protocols*
  - Addressing
  - Connection Establishment and Release
  - Flow Control and Buffering
  - Congestion Control
- The Transport Layer in Internet
  - User Datagram Protocol (UDP)
  - Transmission Control Protocol (TCP)

Transport Protocol vs. Data Link Protocol

- Both have to deal with error control, sequencing, flow control (using ARQ)
- Node-to-node delivery vs. Process-to-process delivery
- Different Environments
  - Addressing
  - Connection establishment
  - Storage capacity in the subnet
  - Allocation of buffers

Fig. 6-7

Data Link Layer: link in between



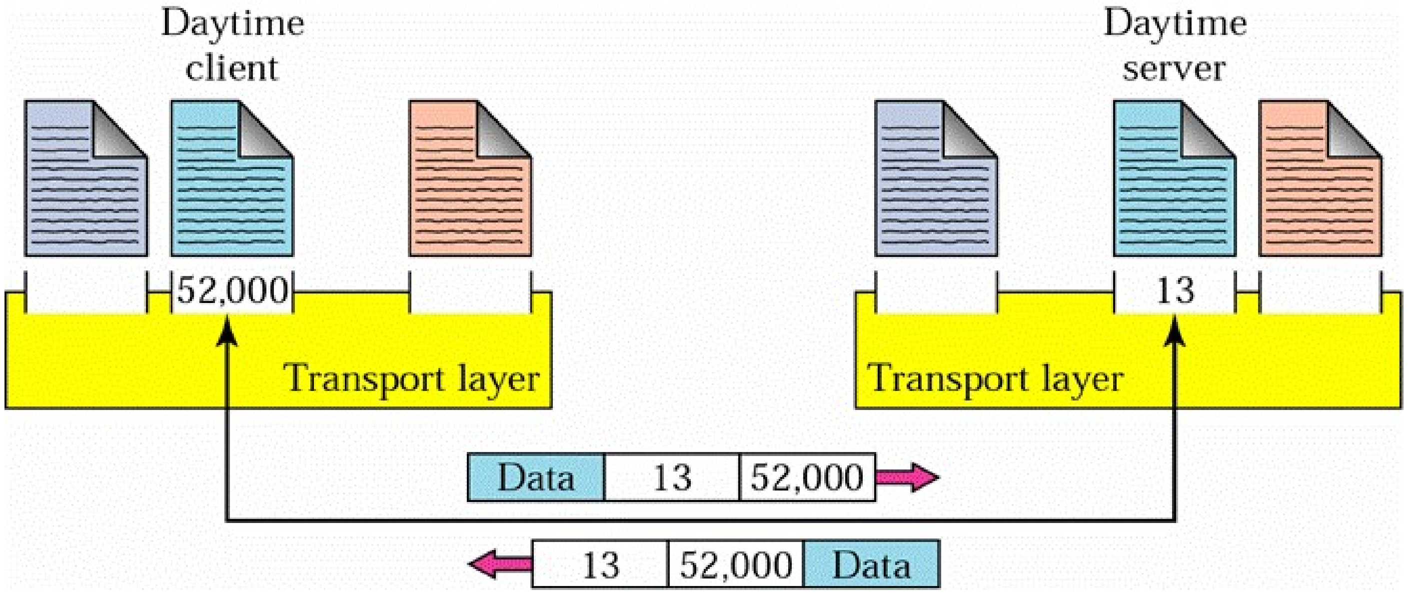
Transport Layer: network(s) in between

Process-to-Process Delivery involving

- Addressing
- Multiplexing and Demultiplexing
- Connectionless/Connection-Oriented
- Reliable/Unreliable
- Flow control and Buffering
- Congestion control

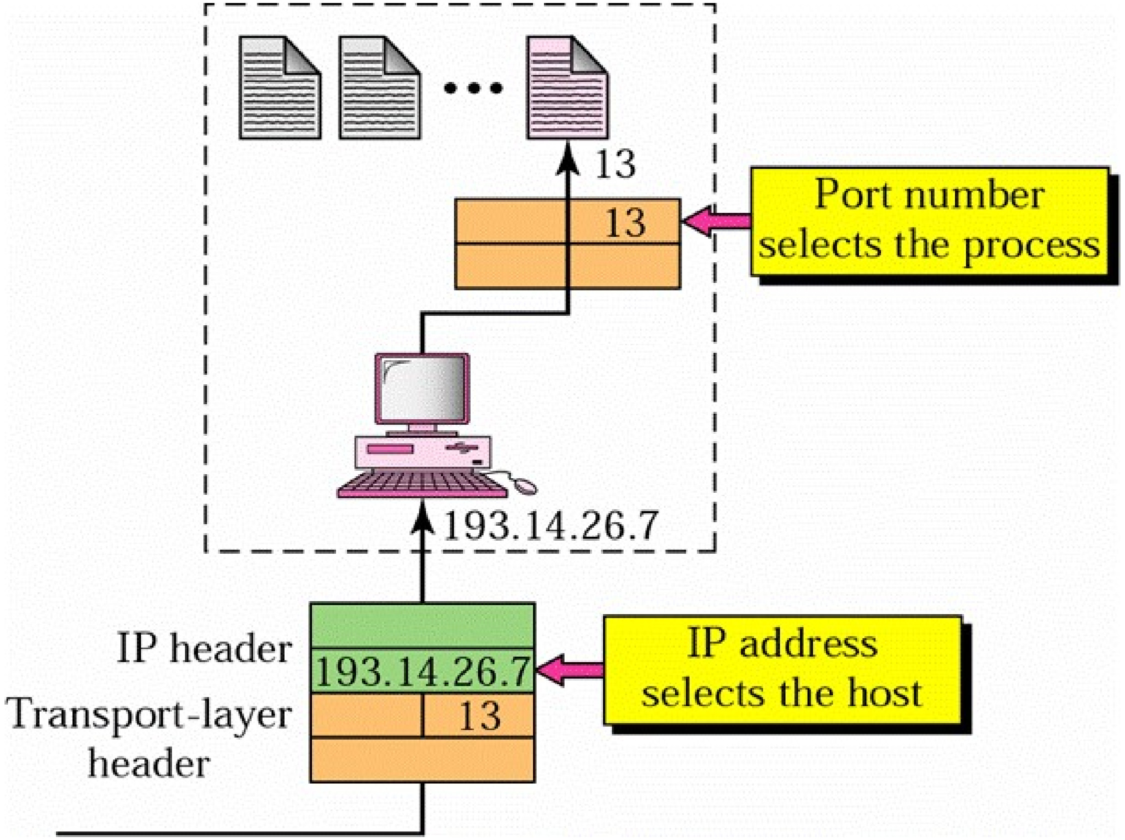
Addressing: Port Number

- SAP of the application process
- Deciding which application gets which messages



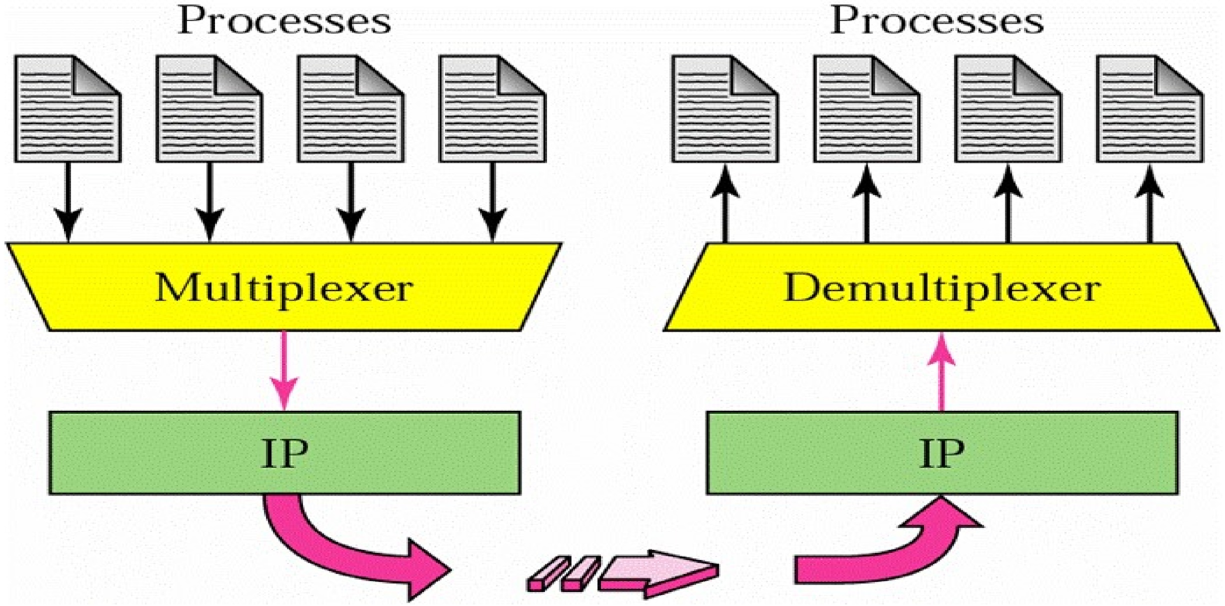
[Forouzan]

IP Address vs. Port Number



[Forouzan]

Multiplexing(复用)



- At sender site, multiplexing at transport layer accepts packets from different processes, differentiated them by port numbers.
- At the receiver site, demultiplexing process works. Transport layer delivers each message to appropriate process based on port number.

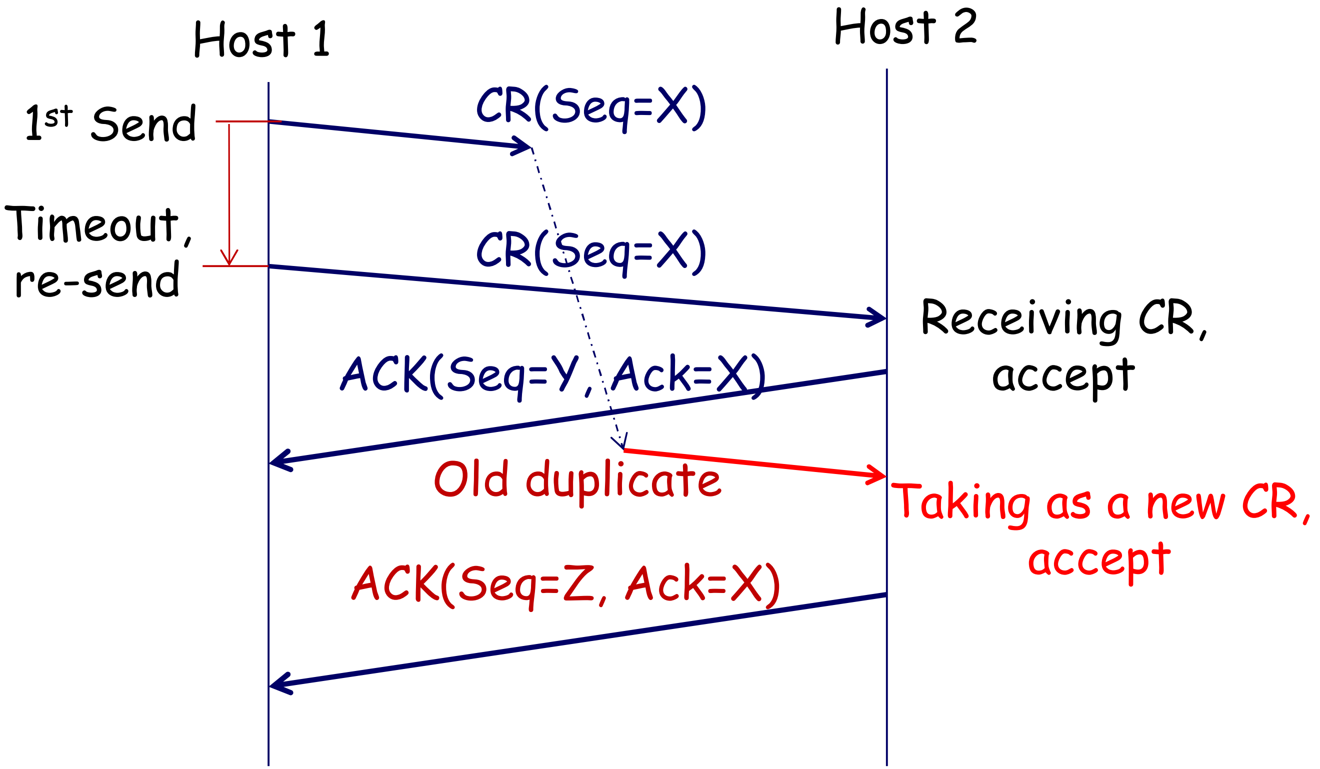
[Forouzan]

Connection Establishment: Problems

- Establishing a connection sounds easy
  - 2-way handshake （两次握手）
  - One sends a ConnectionRequest to the destination and wait for a ConnectionAccepted reply
- Problems
  - Network may delay packets, causing duplicate TPDU (Transport layer PDU)
  - Receiver can not distinguish the duplicate and regard as a new one

Problem of Delayed Duplicate CR

- Connection Request(CR) TPDU was delayed and retransmitted , causing Host2 to regard the duplicate CR as a new one



Solutions: 3 Way Handshake(三次握手/三步握手)

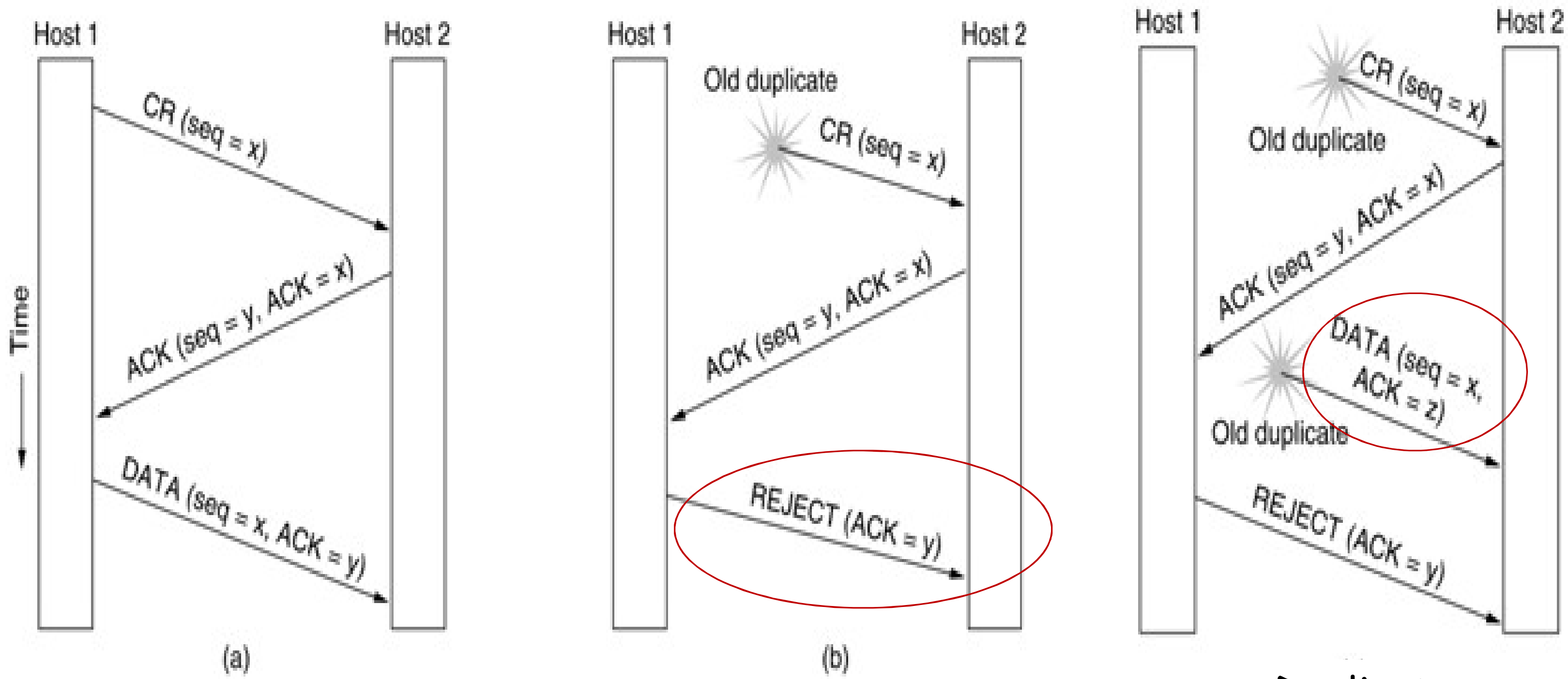


Fig. 6-11

Normal operation

Duplicate
Connection Request

Duplicate
Connection Request
and
duplicate ACK

Connection Release: Problem

- Asymmetric release：abrupt disconnection(突然释放), may cause data loss

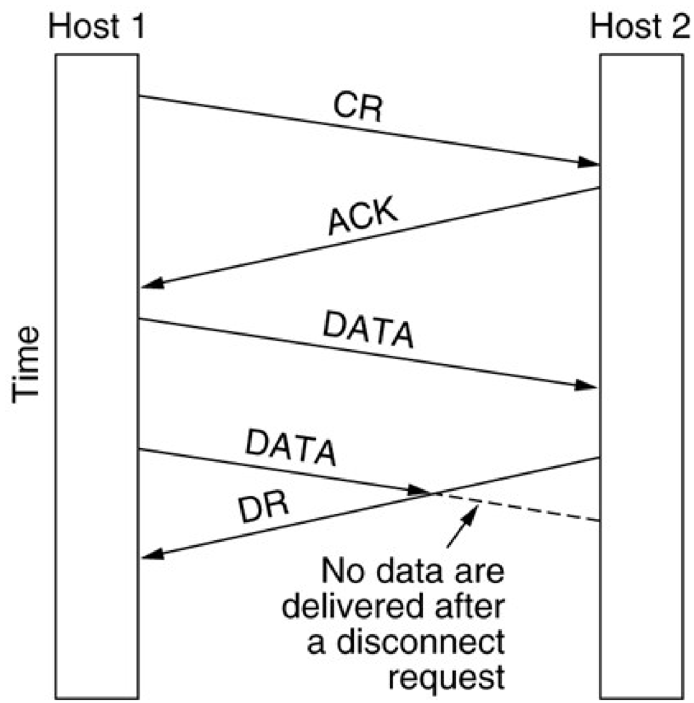


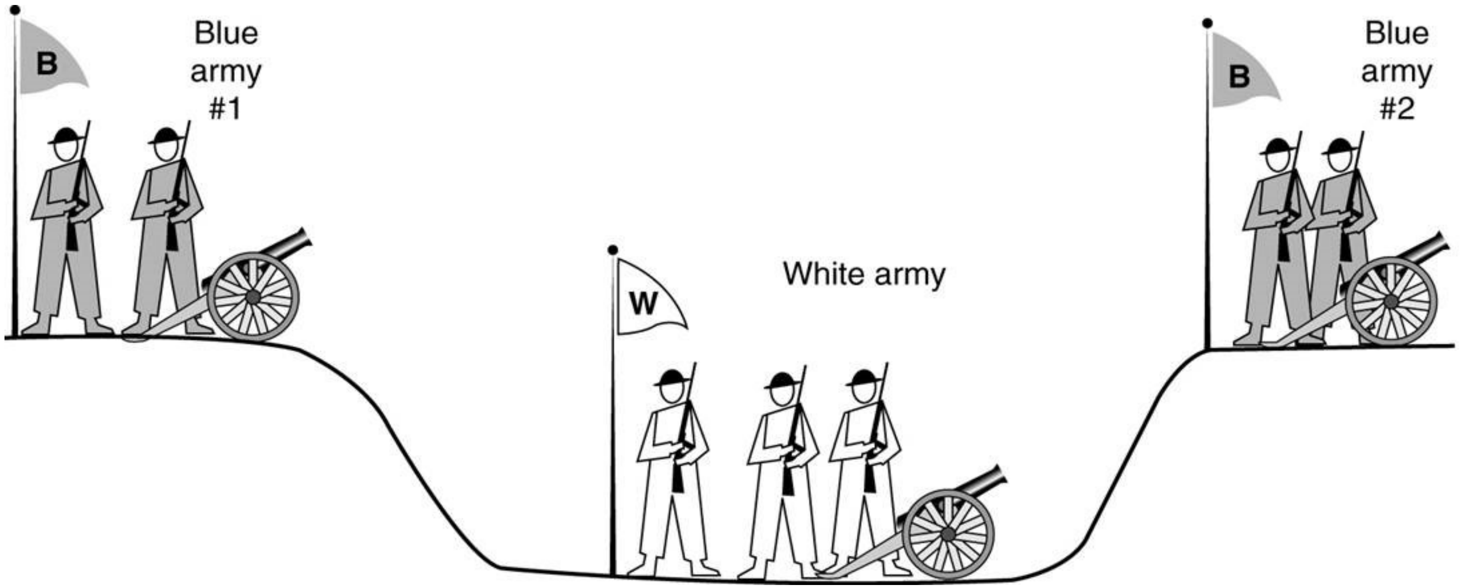
Fig. 6-12

- Symmetric release, in which each direction is released independently of the other one.

Connection Release: 2 Army Problem

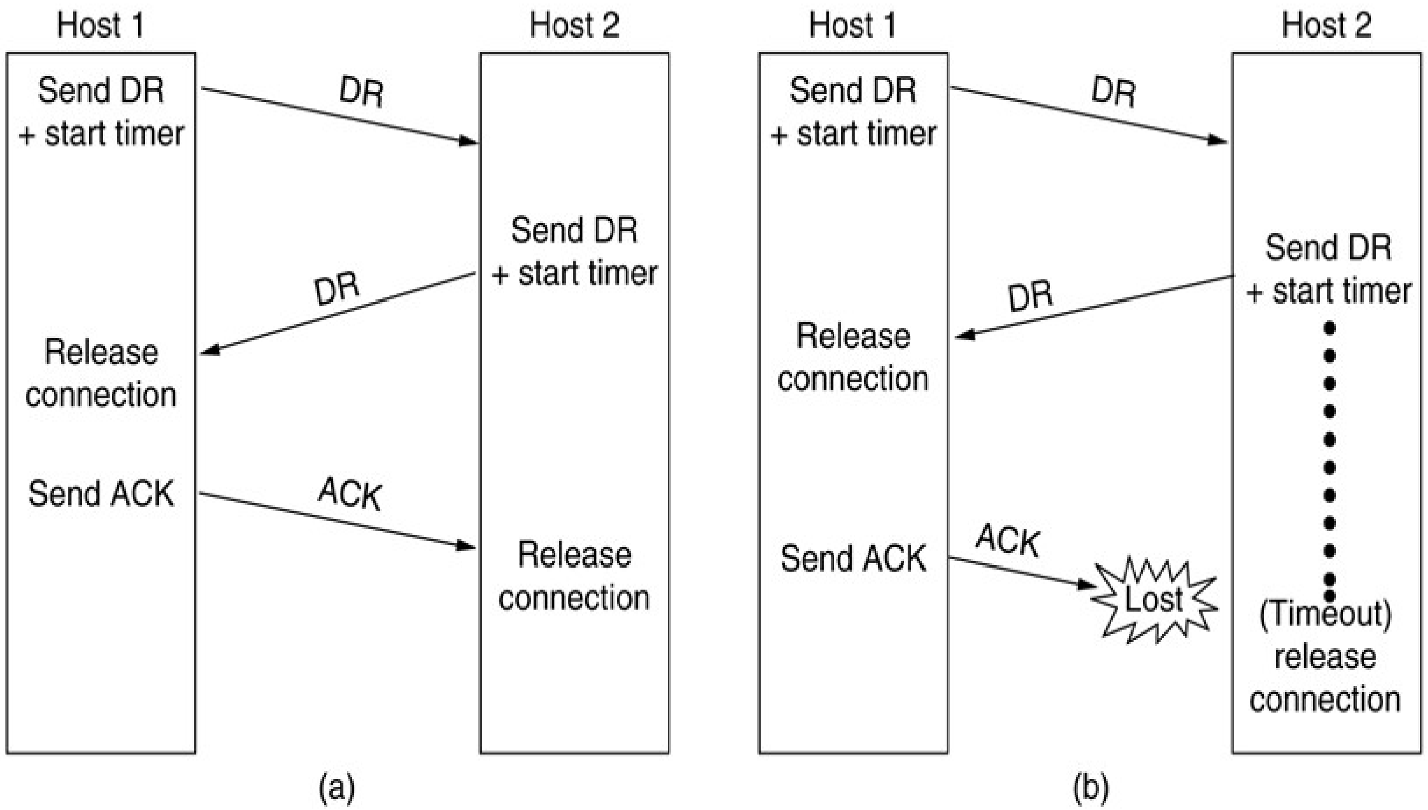
- "Two blue armies need to simultaneously attack the white army to win; otherwise they will be defeated.
- The blue army can communicate only across the area controlled by the white army which can intercept the messengers."

Fig. 6-13



Solution: 3 Way Handshake(1)

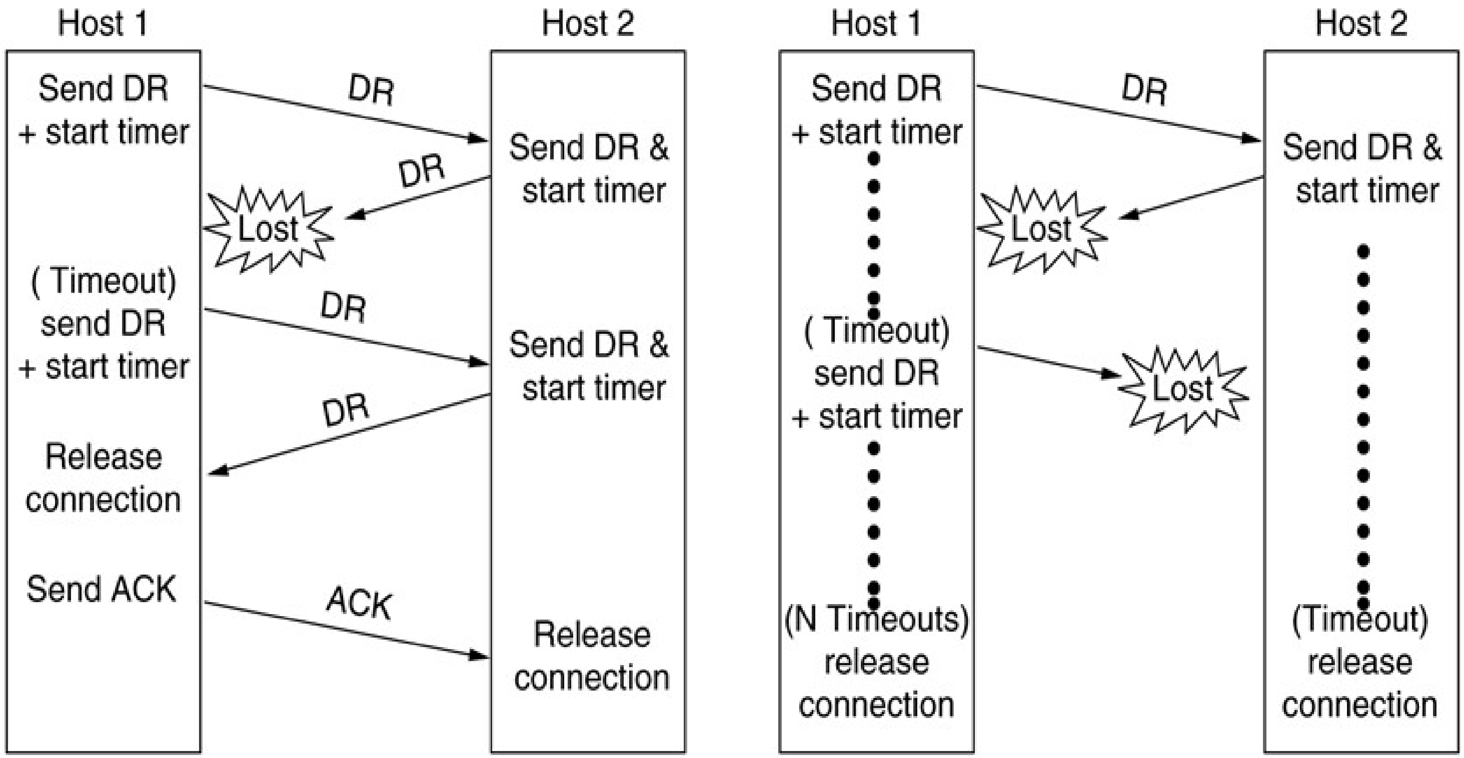
Fig. 6-14



Normal operation

Final ACK lost

3 Way Handshake(2)



(c)
Response lost

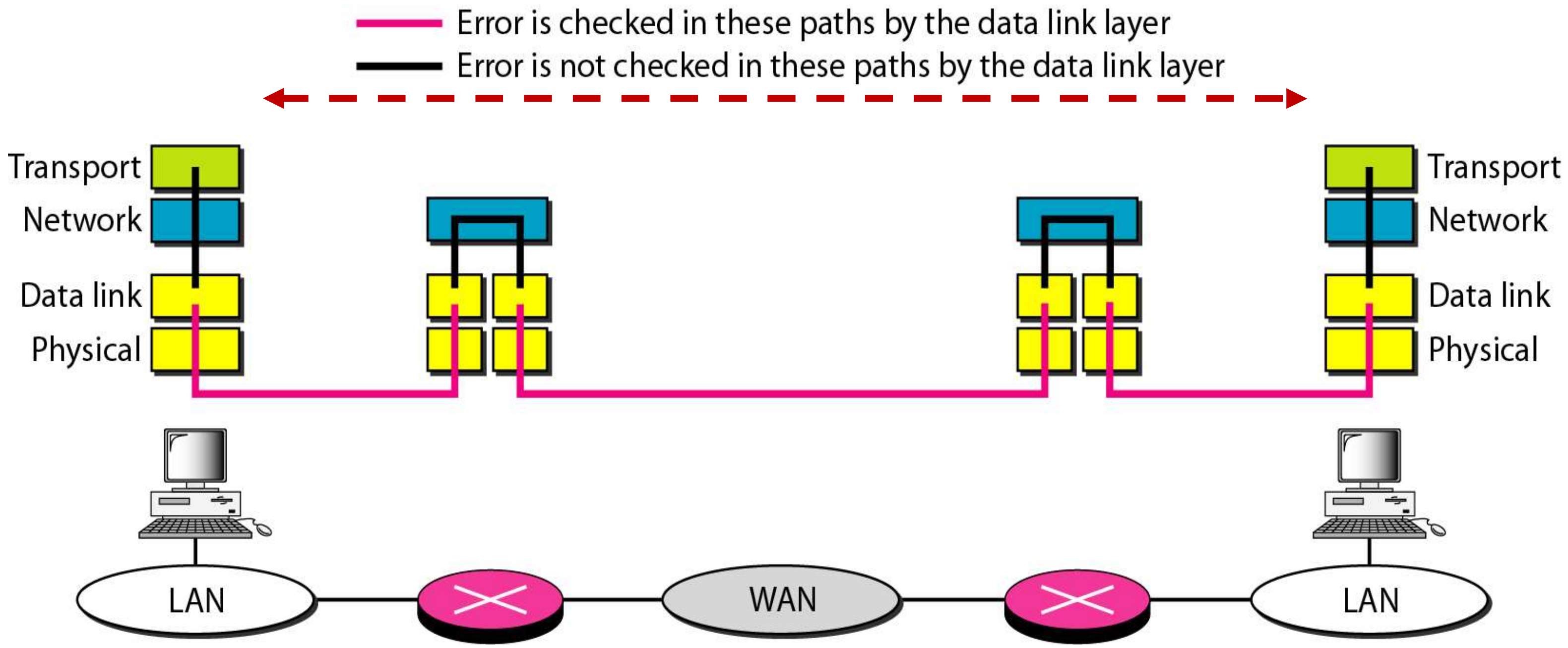
(d)
Response lost and subsequent DRs lost

Fig. 6-14

Error Control

- ARQ(Automatic Repeat reQuest)
  - A segment carries an error-detecting code, a checksum
  - A segment carries a sequence number to identify itself and is retransmitted by the sender until it receives an acknowledgement of successful receipt from the receiver
  - Sliding window protocol combines the features and is also used to support bidirectional data transfer.
- End-to-end argument(端到端/进程-进程)

Error Control: Transport Layer vs. Data Link Layer

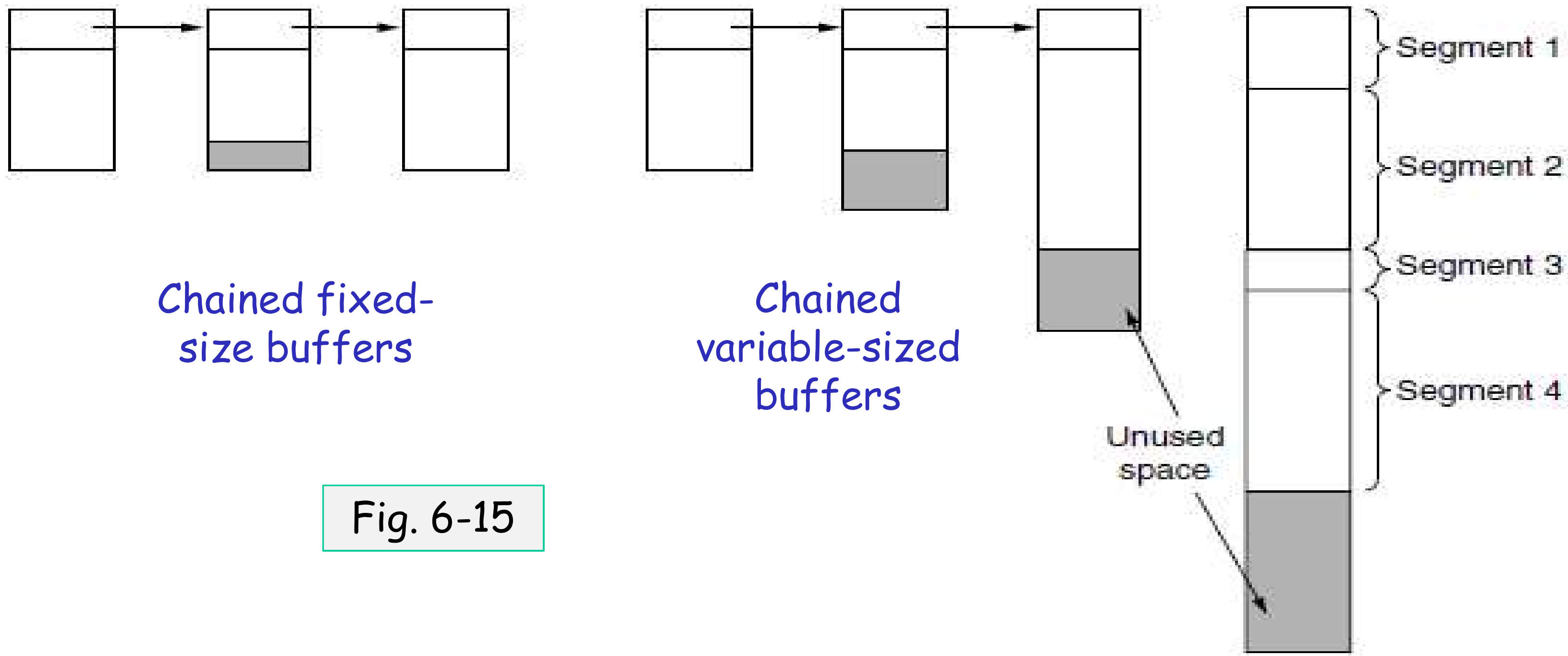


- Difference in function
  - Reliability crossing a single link vs. Reliability crossing an entire network path
- Difference in degree
  - Small window size vs. much larger window size(complex)

Buffering Trade-off

- Source buffering or destination buffering?
  - Depending on the type of traffic carried by the connection
- For low-bandwidth bursty traffic, such as BBS, it is reasonable not to dedicate any buffers, but rather to acquire them dynamically at both ends, relying on buffering at the sender.
- For file transfer and other high-bandwidth traffic, the receiver shall does dedicate a full window of buffers, to allow the data to flow at maximum speed.
  - TCP uses this strategy.

How to organize the buffer pool



Chained fixed-size buffers

Chained variable-sized buffers

One large circular buffer per connection

Fig. 6-15

Flow Control: Dynamic Buffer Allocation

- *Arrows: direction of transmission; ellipsis (…) indicates a lost segment*

	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