CANTHO UNIVERSITY

Chapter 7

Transport Layer

Tran Thanh Dien, PhD

College of Information and communication Technology

Can Tho University

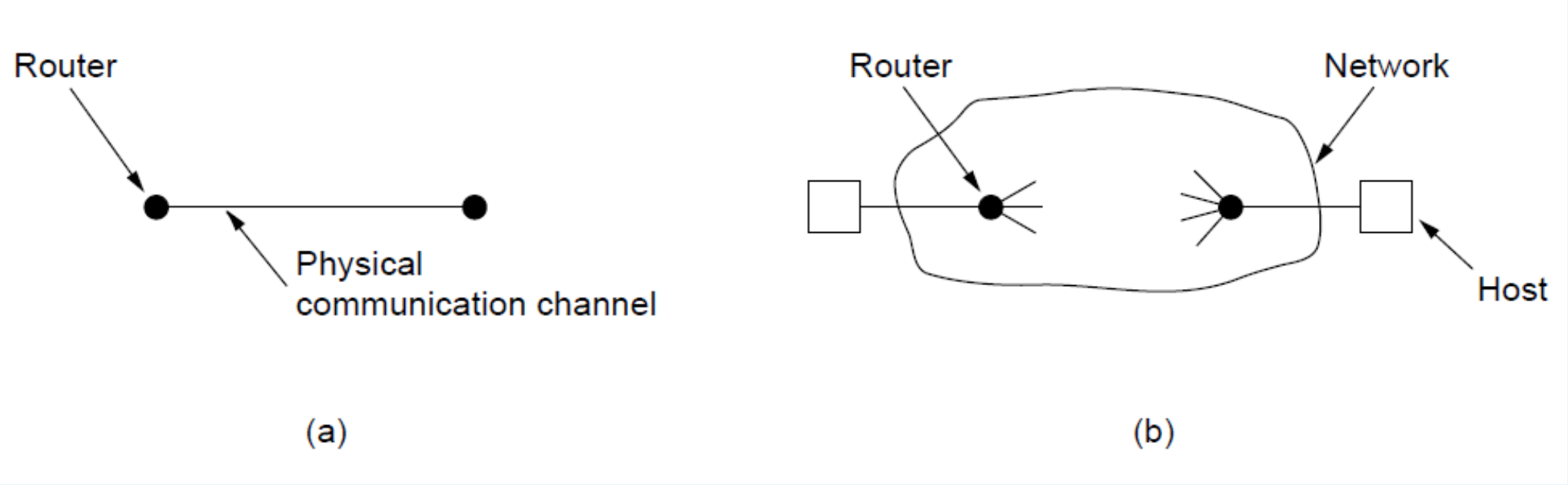
CANTHO UNIVERSITY

Contents

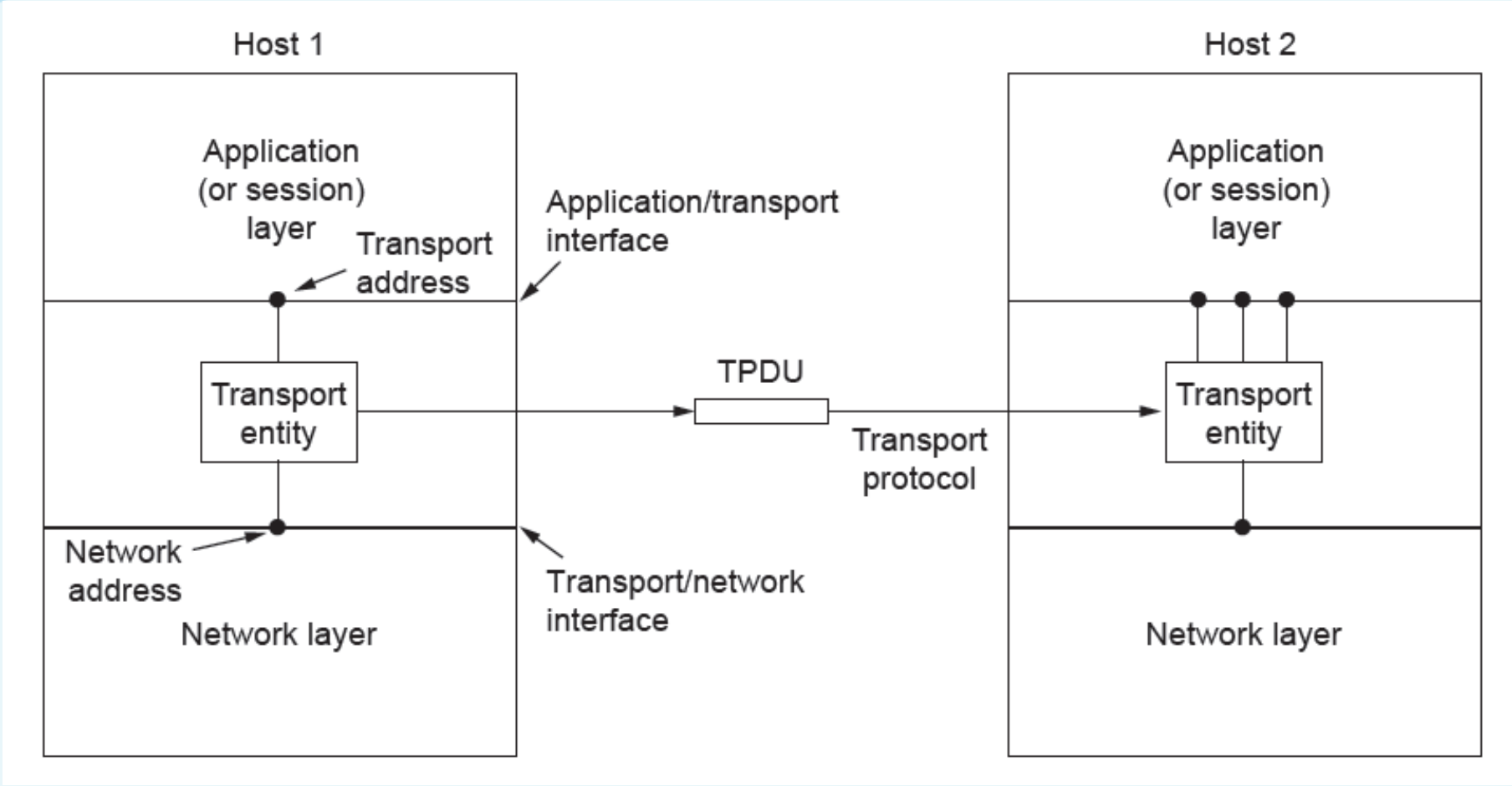
- Roles of the transport layer
- Services provided by the transport layer
- Connection establishment
- Connection release
- TCP and UPD protocols

Roles of Transport layer

- While Network layer provides a host-to-host communication, Transport layer provides a End point-to-End point communication
- End points are running applications
- Provides effective, reliable and cost savings packet transmission service for users



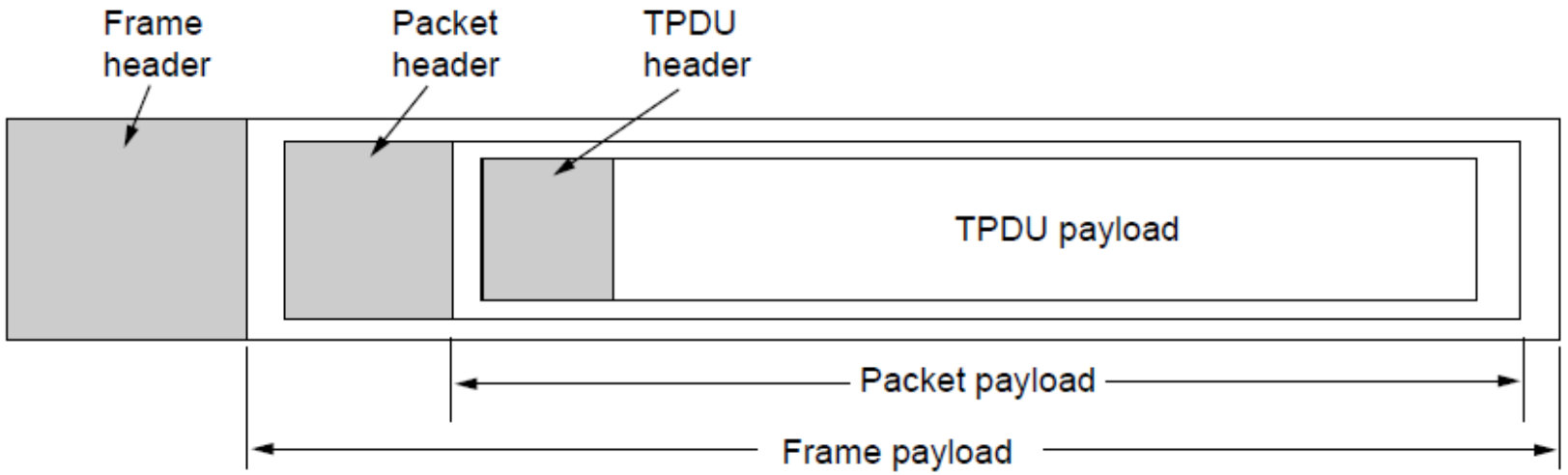
Services Provided to the Upper Layers



Services Provided to the Upper Layers

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



Services Provided to the Upper Layers

Transport Service Primitives: Berkeley Sockets

Primitive	Meaning
SOCKET	Create a new communication end point
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

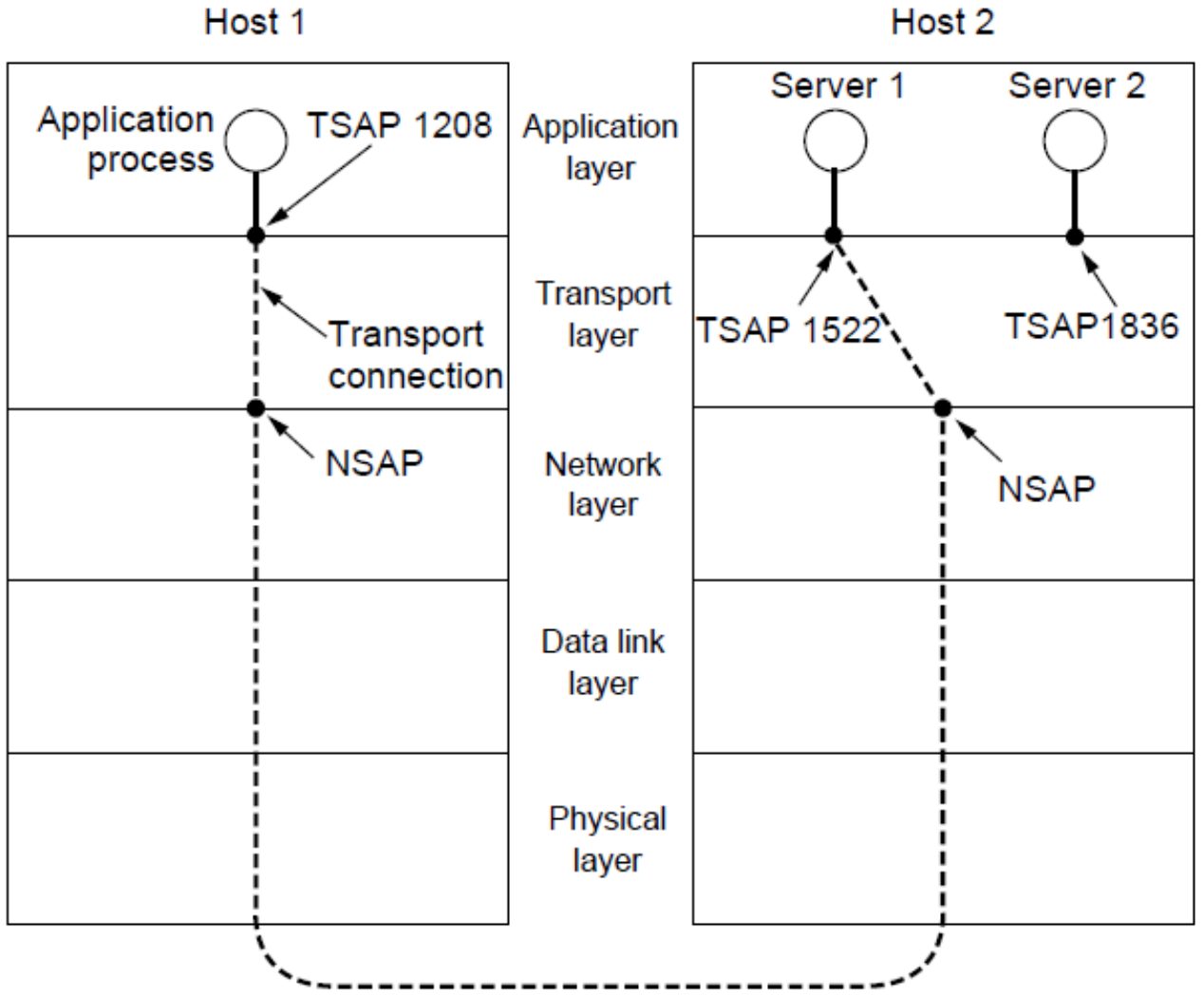
The socket primitives for TCP

Elements of Transport Protocols

- Addressing
- Connection establishment
- Connection release
- Error control and flow control
- Multiplexing
- Crash recovery

Elements of Transport Protocols

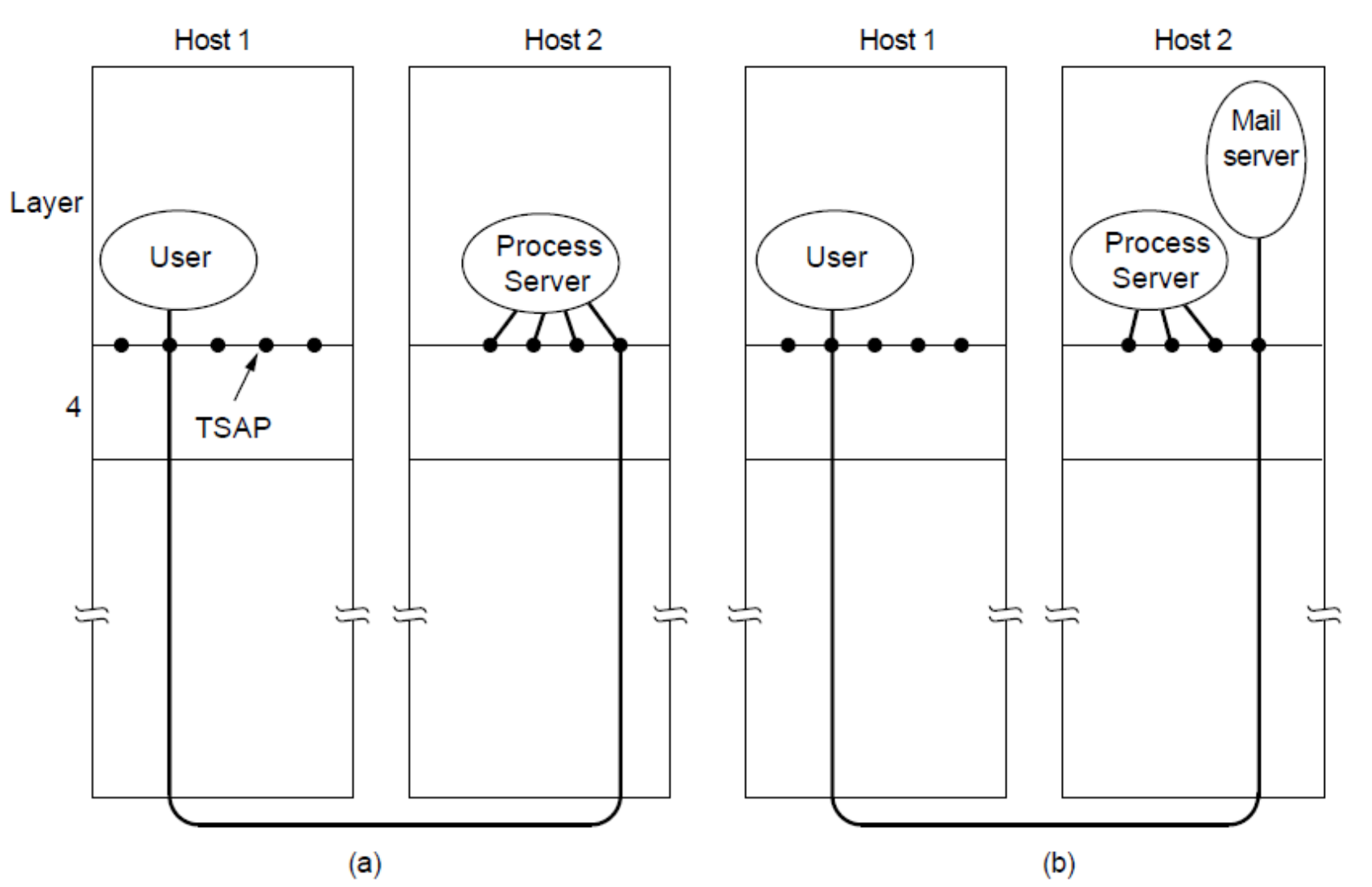
Addressing



Elements of Transport Protocols

Addressing

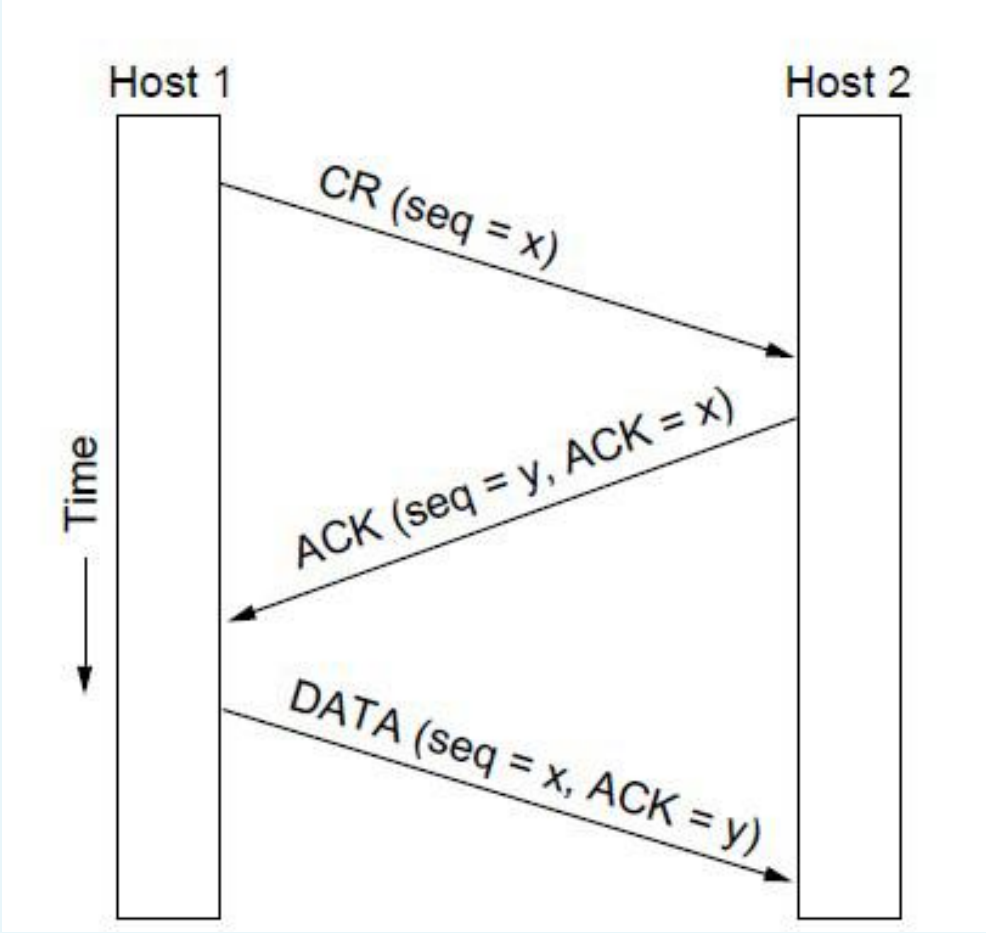
A user process in host 1 establishes a connection with a mail server in host 2 via a process server.



Elements of Transport Protocols

Connection Establishment

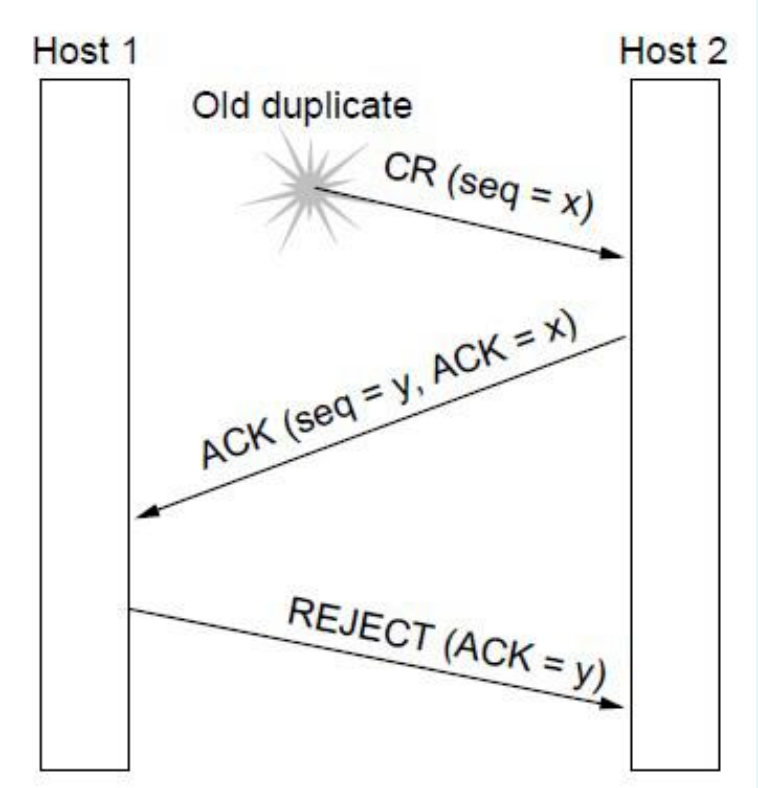
- Establishing a connection using a three-way handshake.
- CR denotes CONNECTION REQUEST.



Normal operation

Elements of Transport Protocols

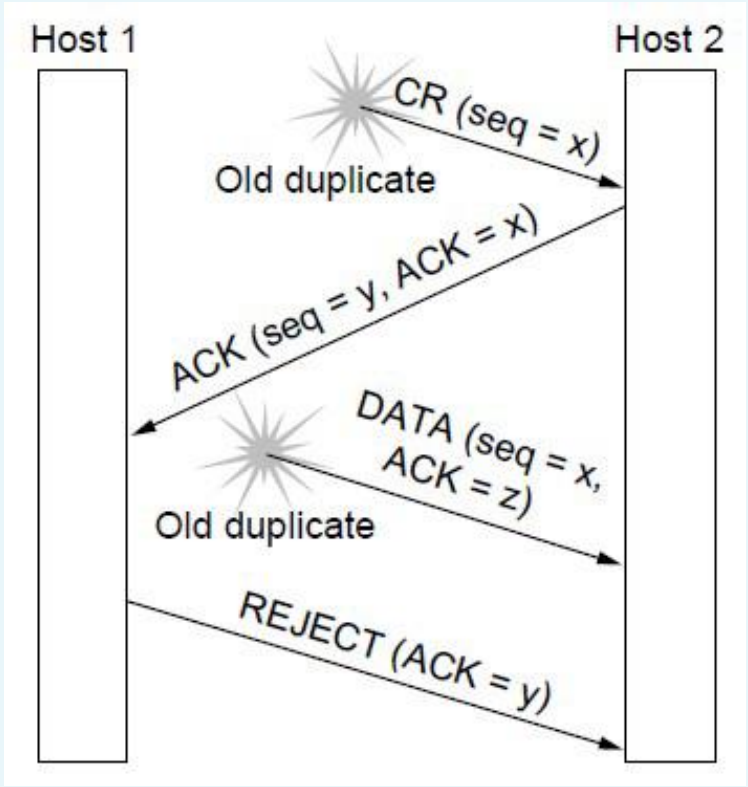
Connection Establishment



Old duplicate CONNECTION REQUEST appearing out of nowhere

Elements of Transport Protocols

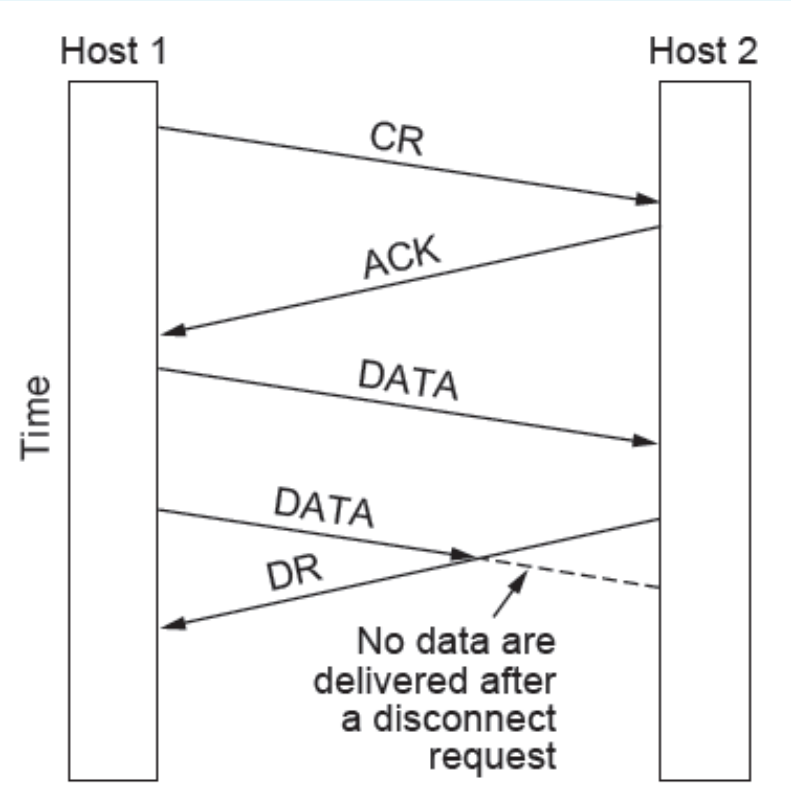
Connection Establishment



Duplicate CONNECTION REQUEST and duplicate ACK

Elements of Transport Protocols

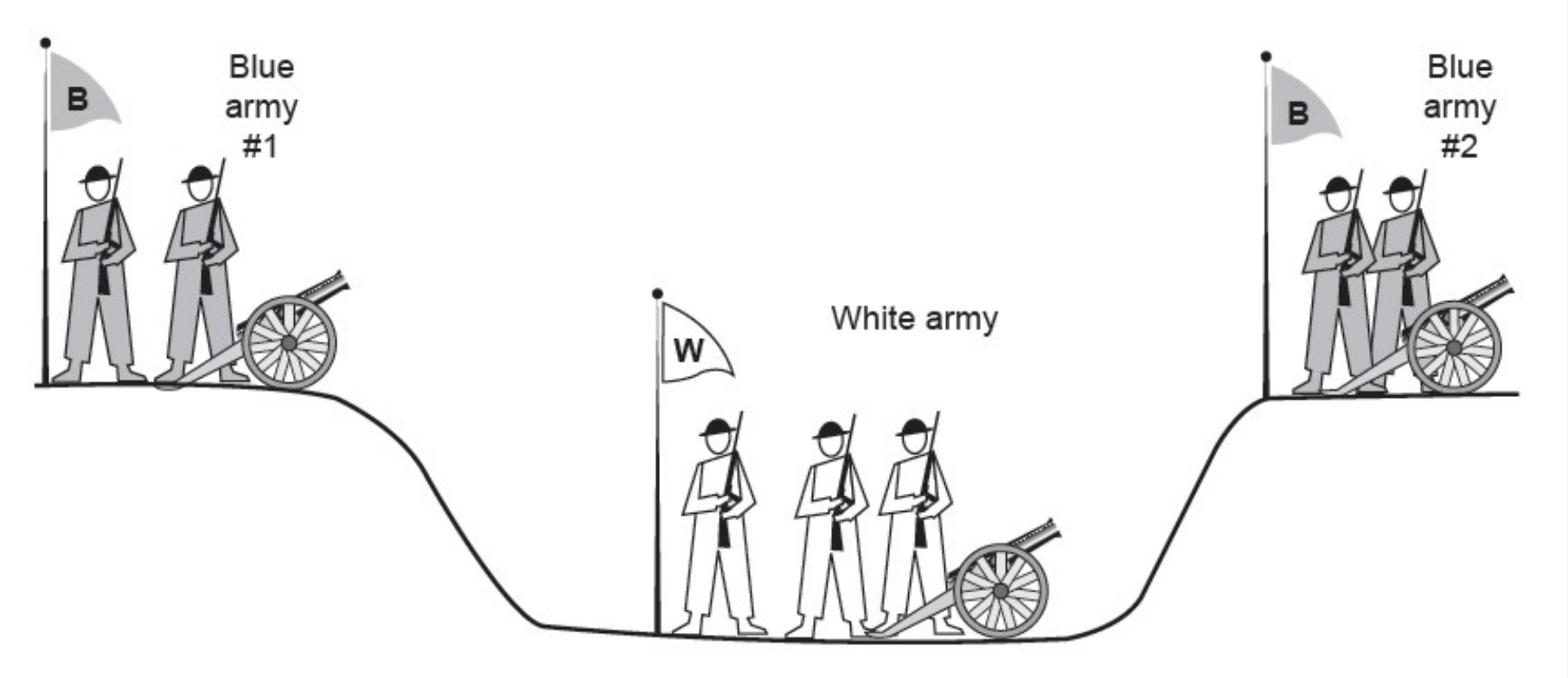
Connection Release



Abrupt disconnection with loss of data

Elements of Transport Protocols

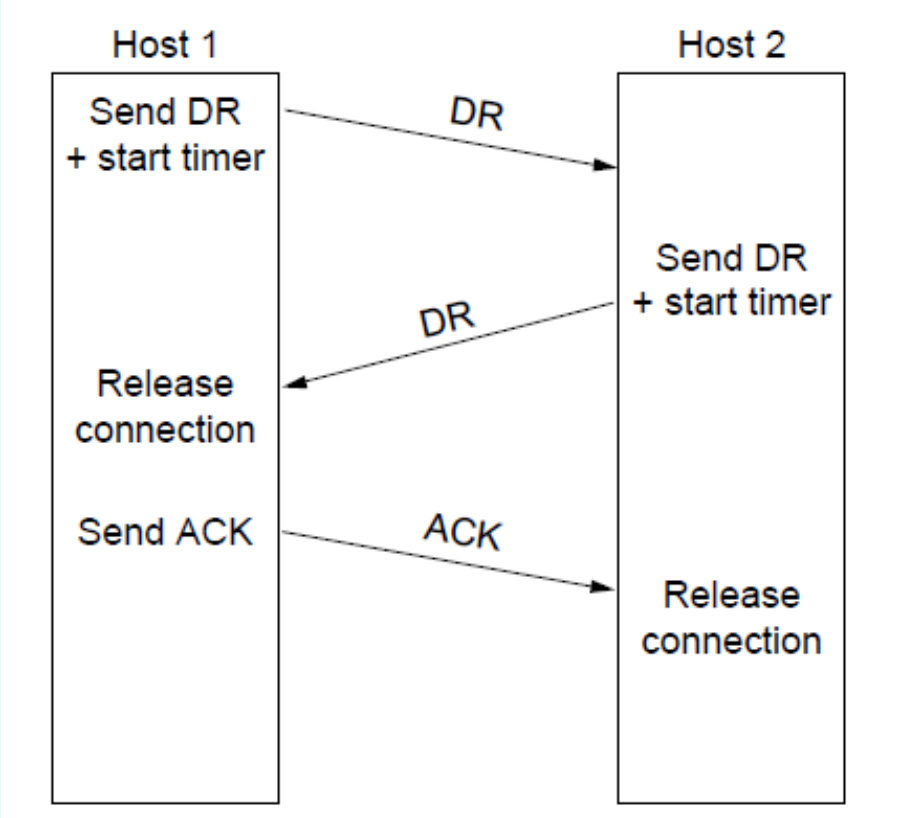
Connection Release



The two-army problem

Elements of Transport Protocols

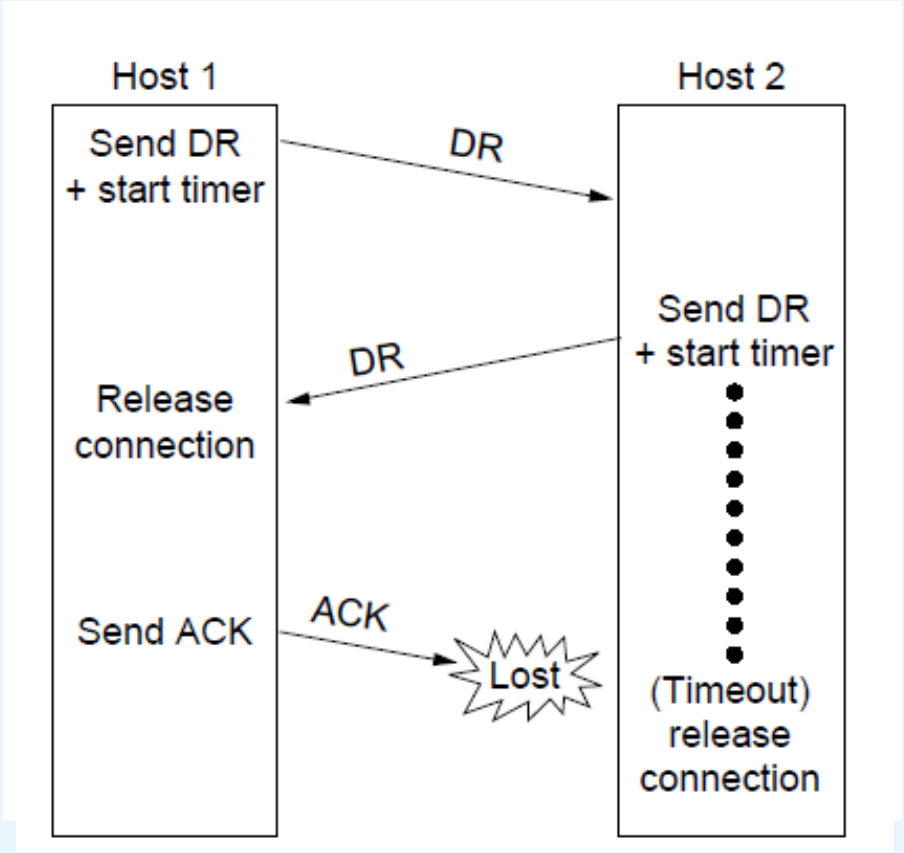
Connection Release



a) Normal case of three-way handshake for releasing a connection

Elements of Transport Protocols

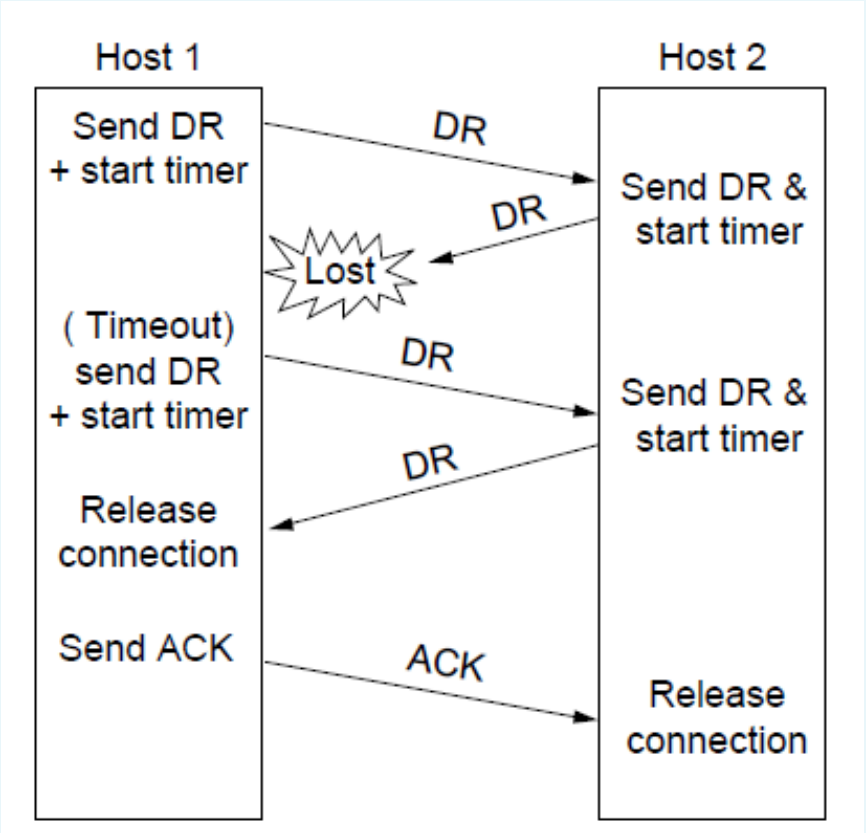
Connection Release



b) Final ACK lost

Elements of Transport Protocols

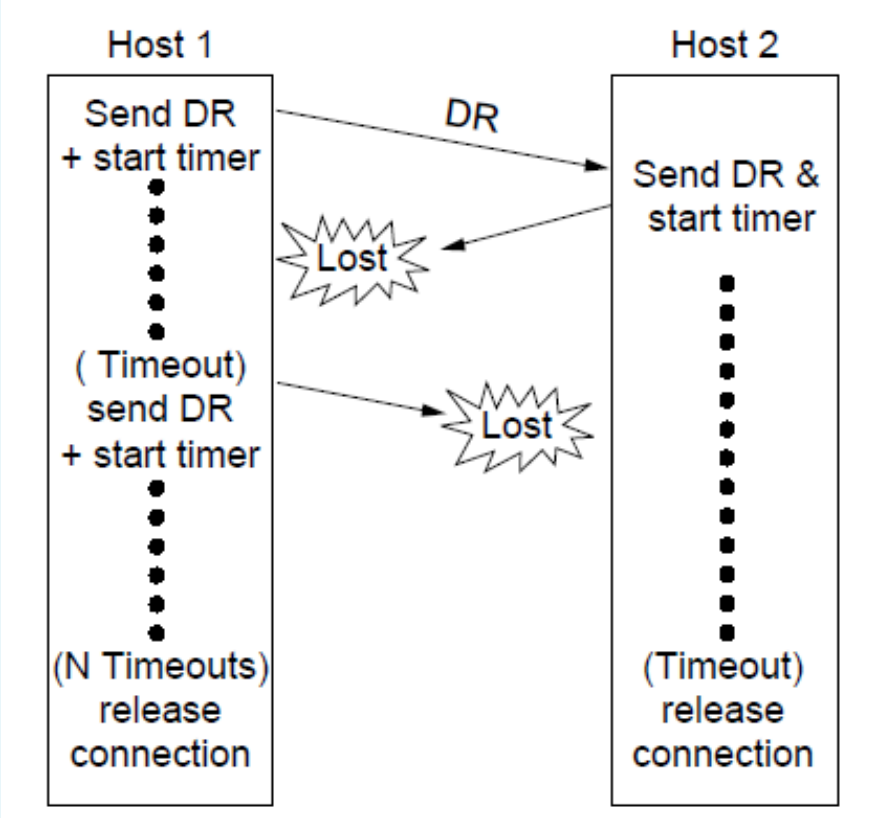
Connection Release



c) Response lost

Elements of Transport Protocols

Connection Release



d) Response lost and subsequent DRs lost.

Elements of Transport Protocols

Flow Control

- Using Sliding window protocol where the size of the sending window and the receiving window are different
- Need a scheme for allocating buffer dynamically

CANTHO UNIVERSITY

Elements of Transport Protocols

Flow Control

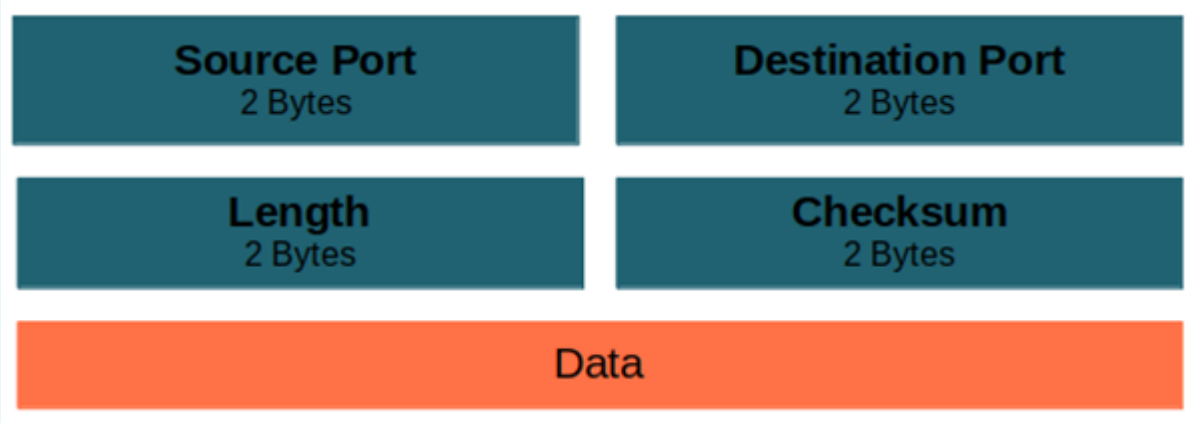
Dynamic buffer allocation. The arrows show the direction of transmission. An ellipsis (...) indicates a lost TPDU buffer dynamically

	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

Internet Protocol Suite

UDP - User Datagram Protocol

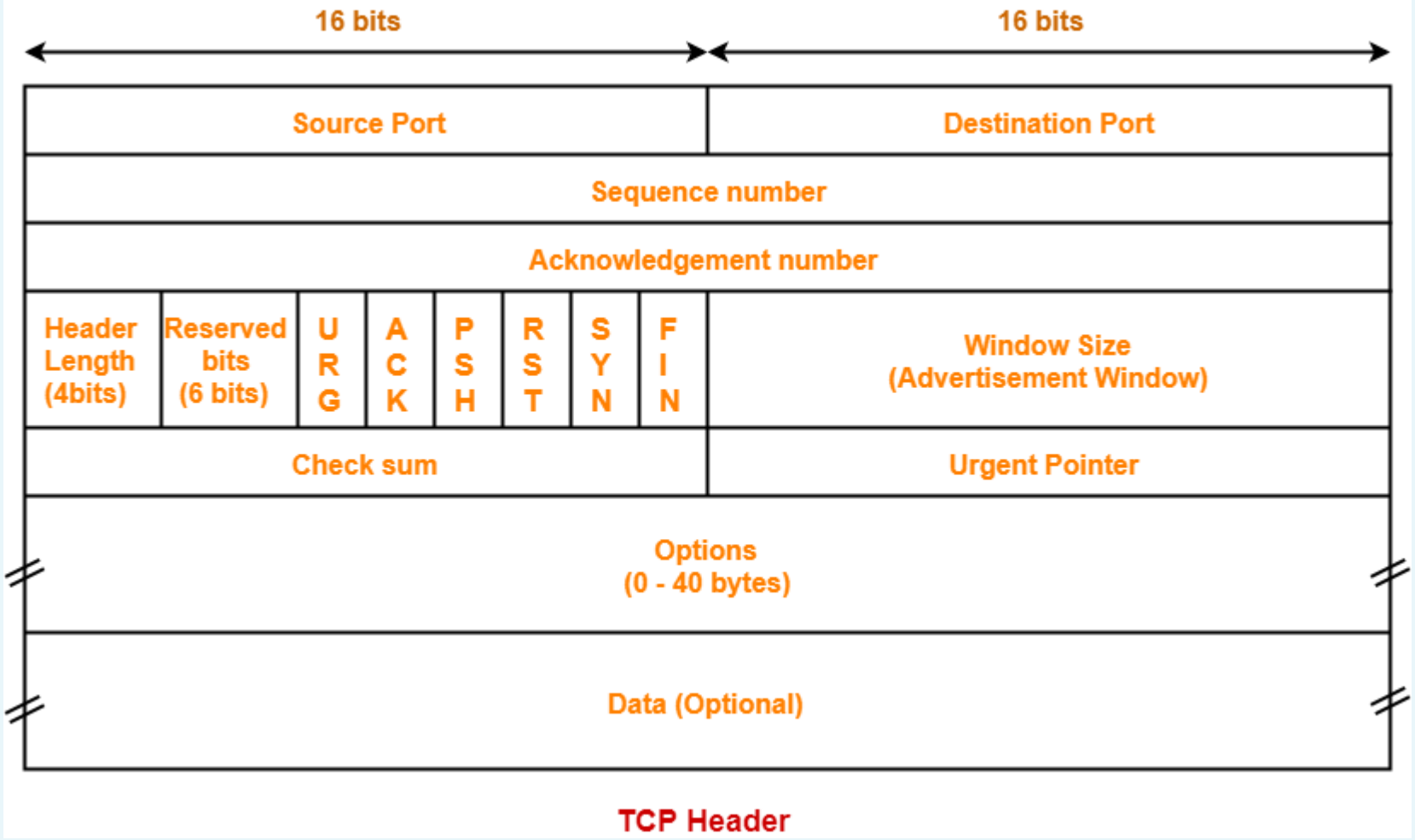
- A connectionless communication protocol
- Not required to establish a connection between two end points
- A UDP segment could appear at a destination at anytime
- A UDP segment contains enough information that can be used to forward the segment to the destination



Internet Protocol Suite

TCP - Transmission Control Protocol

- A connection-oriented communication protocol
- Byte-oriented protocol



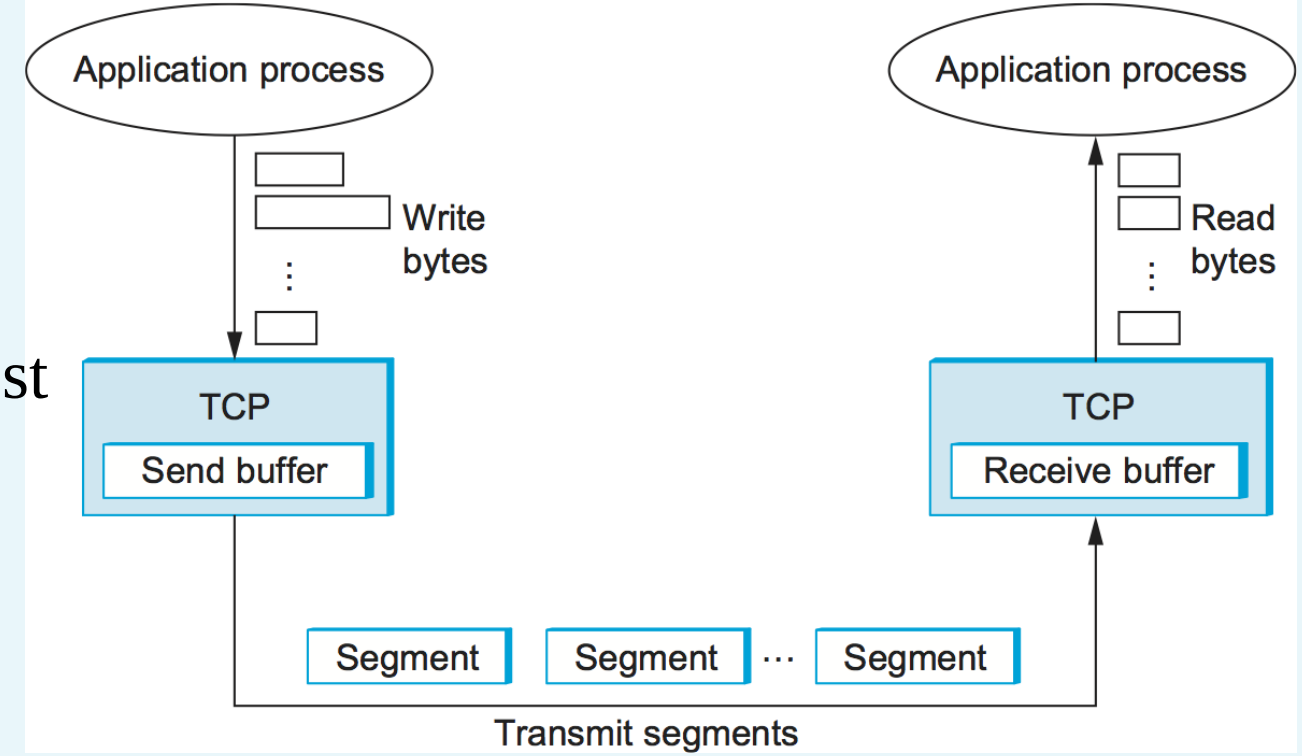
TCP Header

Internet Protocol Suite

TCP - Transmission Control Protocol

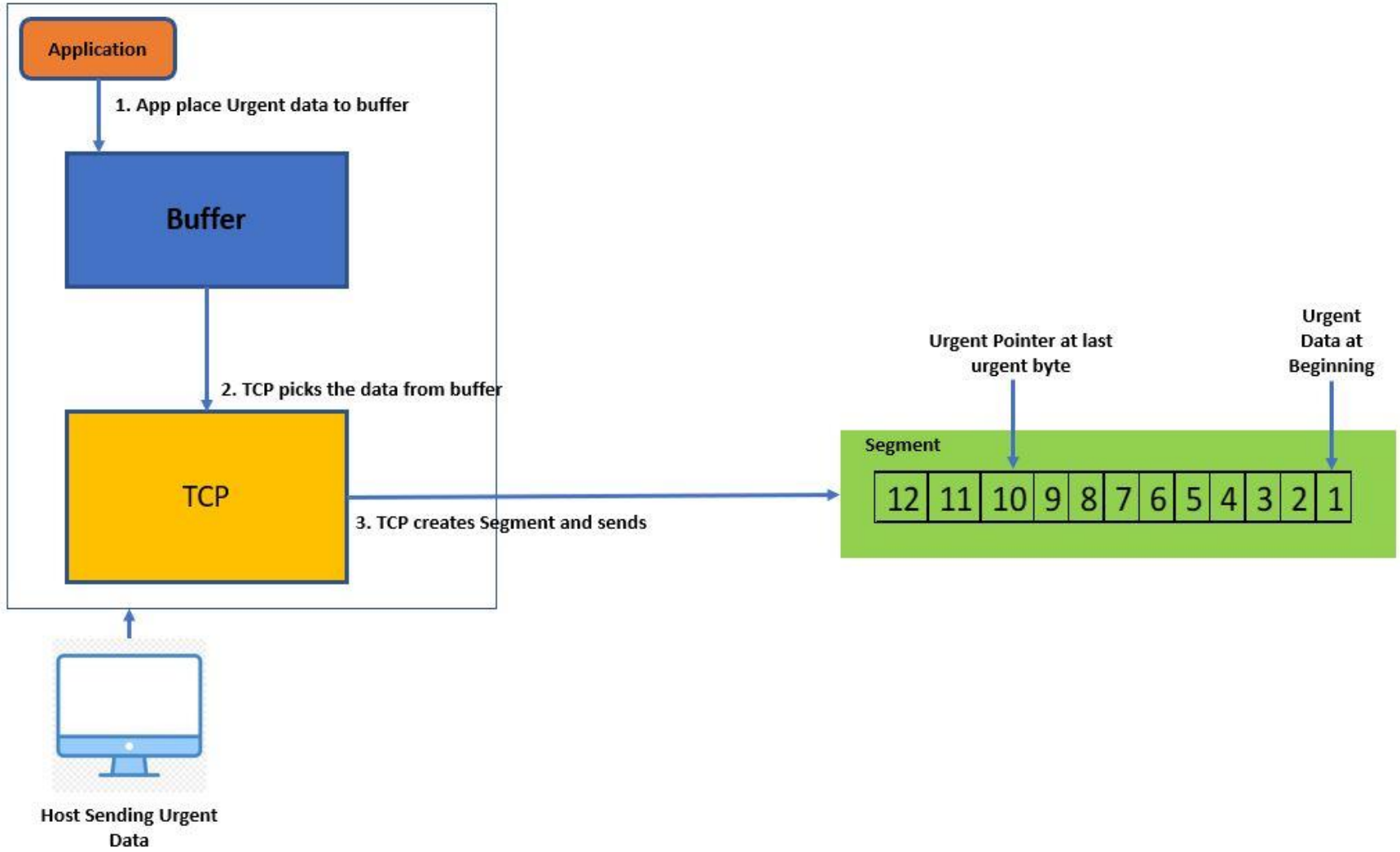
- Sender writes bytes into a TCP connection and the receiver reads bytes out of the TCP connection

- Not transmit individual bytes over the Internet
- Source host buffers enough bytes from the sending process to fill a reasonably sized packet and sends to destination host
- Destination host then empties the contents of the packet into a receive buffer, and the receiving process reads from this buffer at its leisure



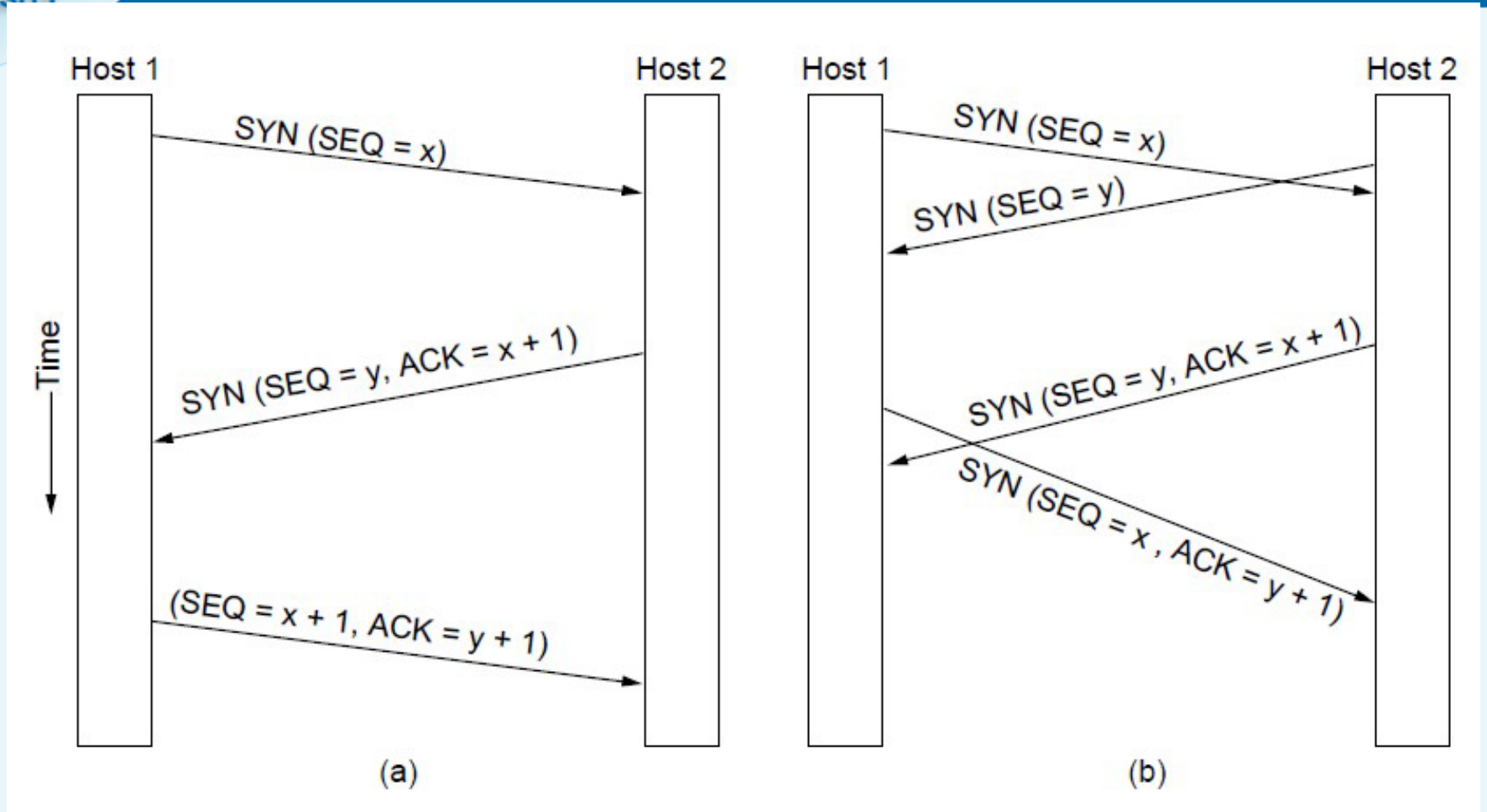
Internet Protocol Suite

TCP - Transmission Control Protocol



Internet Protocol Suite

TCP - Connection establishment



a) TCP connection establishment in the normal case

b) Simultaneous connection establishment on both sides

Internet Protocol Suite

TCP - Flow control with sliding window protocol