Chapter 20: TCP

Raj Jain

Professor of CIS

Raj Jain is now at
Washington University in Saint Louis
Jain@cse.wustl.edu
http://www.cse.wustl.edu/~jain/

Overview

❑ Key features, Header format

❑ Mechanisms, Implementation choices

❑ Slow start congestion avoidance, Fast Retransmit/Recovery

❑ Selective Ack and Window scaling options

❑ UDP

Raj Jain

Key Features of TCP

- ❑ Connection oriented

- ❑ Point-to-point communication: Two end-points

- ❑ Reliable transfer: Data is delivered in order

- ❑ Full duplex communication

- ❑ Stream interface: Continuous sequence of octets

- ❑ Reliable connection startup: Data on old connection does not confuse new connections

- ❑ Graceful connection shutdown: Data sent before closing a connection is not lost.
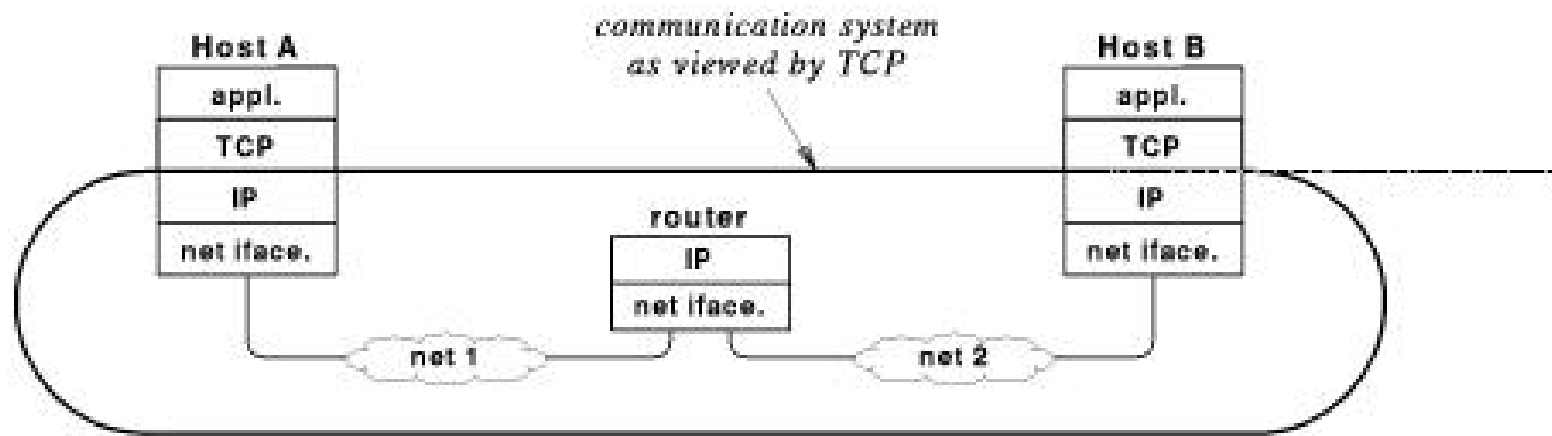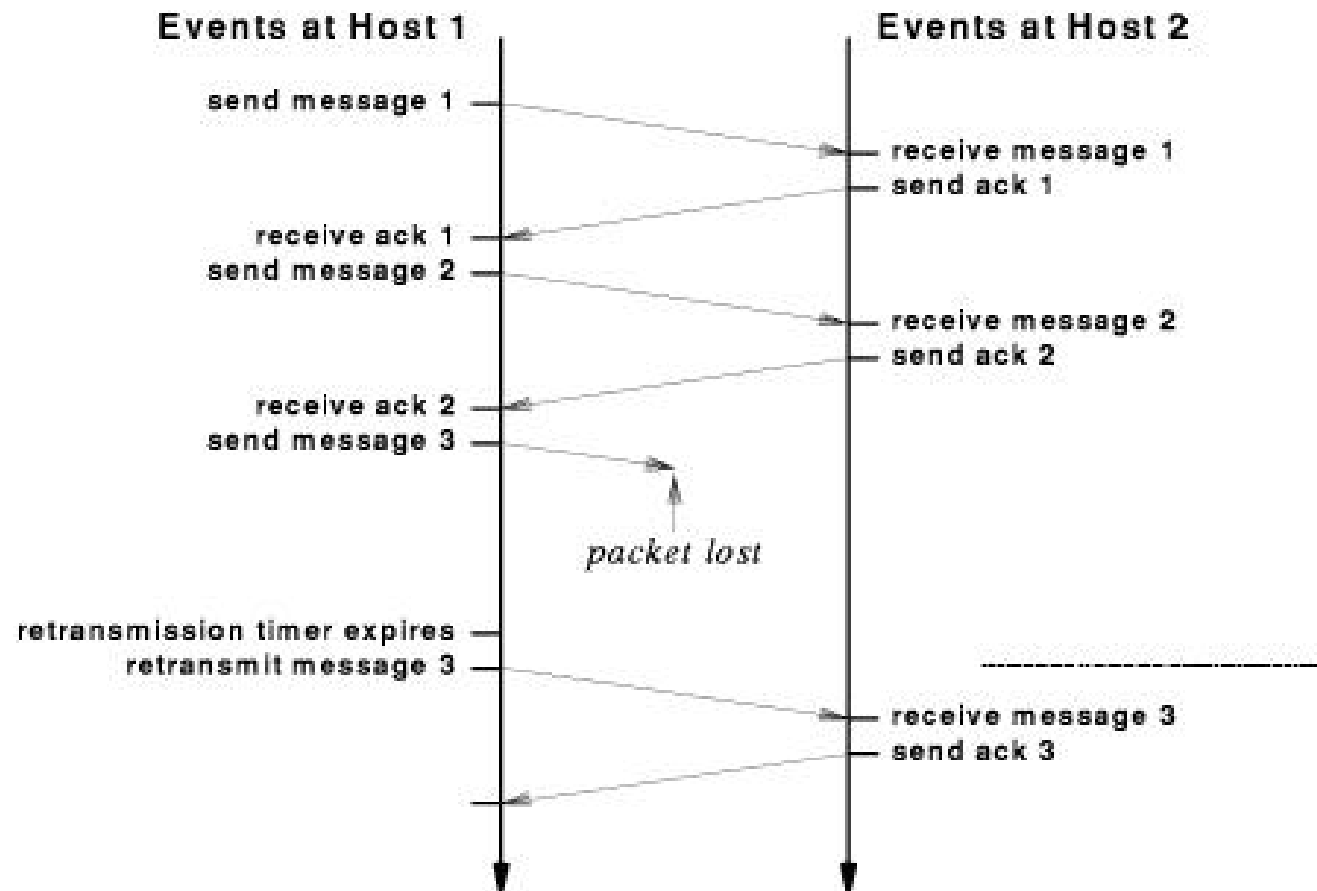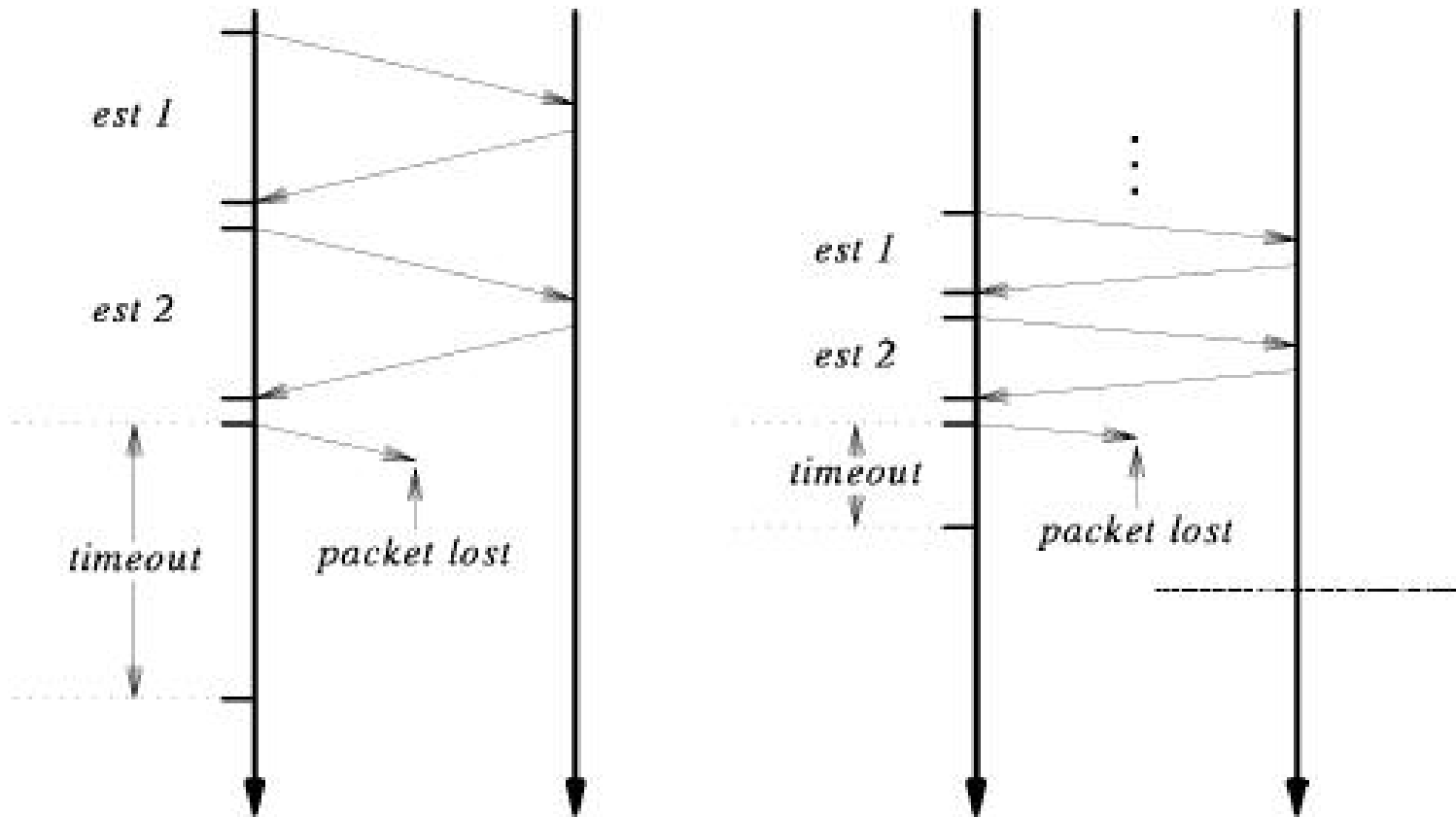
End-to-end Service

Fig 20.1

Reliable Transmission

Events at Host 1   Events at Host 2

send message 1
                                                    receive message 1
                                                    send ack 1
receive ack 1
send message 2
                                                    receive message 2
                                                    send ack 2
receive ack 2
send message 3

packet lost

retransmission timer expires
retransmit message 3
                                                    receive message 3
                                                    send ack 3

❑ Data not acked is retransmitted.

Fig 20.2

Adaptive Retransmission

est 1

est 2

timeout

packet lost
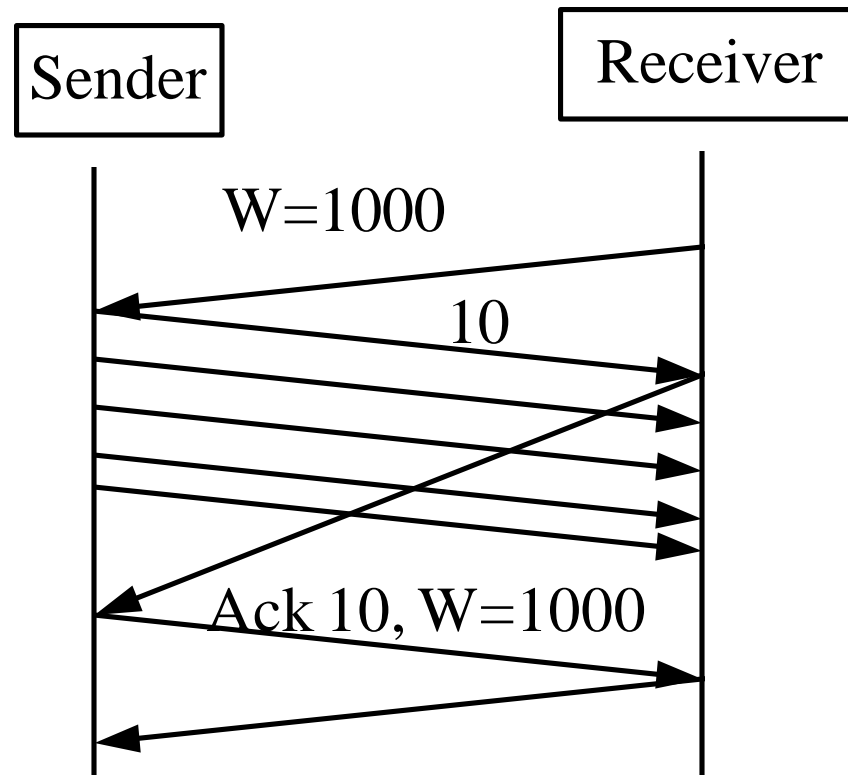
est 1

est 2

timeout

packet lost

❑ Retransmission timeout is based on measured round-trip time

Fig 20.3

Raj Jain

Window Flow Control

Fig 20.4
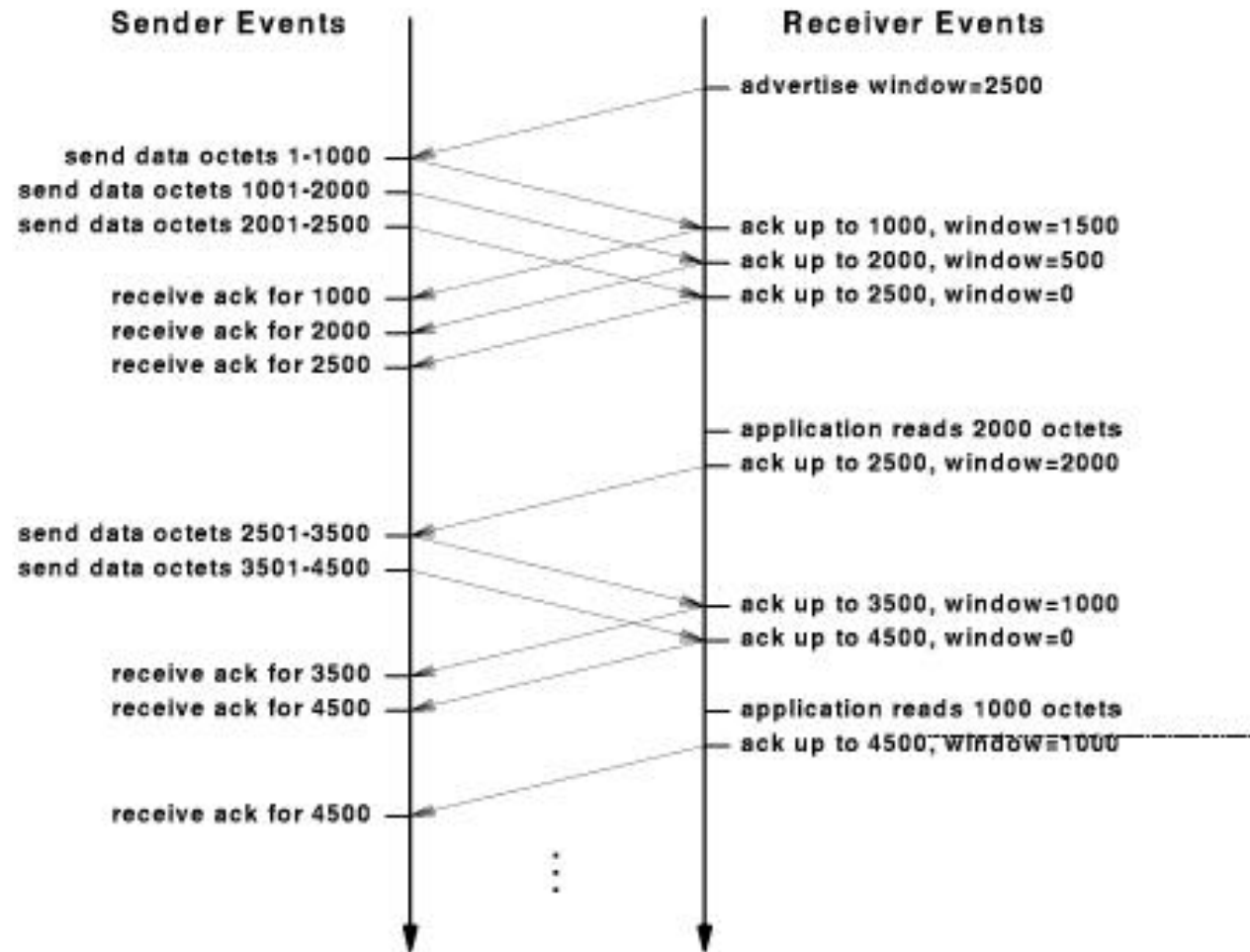
Silly Window Syndrome

❑ The system operates at a small window even if the receiver grants a large window.

❑ Ref: RFC0813

Transport Control Protocol (TCP)

❑ Key Services:

    ❑ Send: Please send when convenient

    ❑ Data stream push: Please send it all now

    ❑ Urgent data signaling: Destination TCP! please give this urgent data to the user

TCP Header Format

Source Port	Dest Port	Seq No	Ack No	Data Offset	Resvd	Flags	Window
16	16	32	32	4	6	6	16

Check-sum	Urgent	Options	Pad	Data
16	16	x	y	← Size in bits

TCP Header

❑ Source Port (16 bits): Identifies source user process

❑ Destination Port (16 bits)

❑ Sequence Number (32 bits): Sequence number of the first byte in the segment. If SYN is present, this is the initial sequence number (ISN) and the first data byte is ISN+1.

❑ Ack number (32 bits): Next byte expected

❑ Data offset (4 bits): Number of 32-bit words in the header

❑ Reserved (6 bits)

TCP Header (Cont)

❑ Flags (6 bits):   Urgent pointer field significant,
                     ack field significant,
                     push function, reset the connection,
                     synchronize the sequence numbers,
                     no more data from sender

❑ Window (16 bits): Will accept [Ack] to [Ack]+[window]

TCP Header (Cont)

❑ Checksum (16 bits): covers the segment plus a pseudo header
Includes the following fields from IP header: source and dest adr, protocol, segment length. Protects from IP misdelivery.

❑ Urgent pointer (16 bits): Points to the byte following urgent data. Lets receiver know how much urgent data is coming.

❑ Options (variable):
Max TPDU size (Default 536 bytes)
Window scale, SACK permitted

TCP Checksum

❑ Checksum is the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the TCP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

❑ Checksum field is filled with zeros initially

❑ TCP length (in octet) is not transmitted but used in calculations

Source Adr	Dest. Adr	Zeros	Protocol	TCP Length
32	32	8	8	16

| TCP Header	TCP data
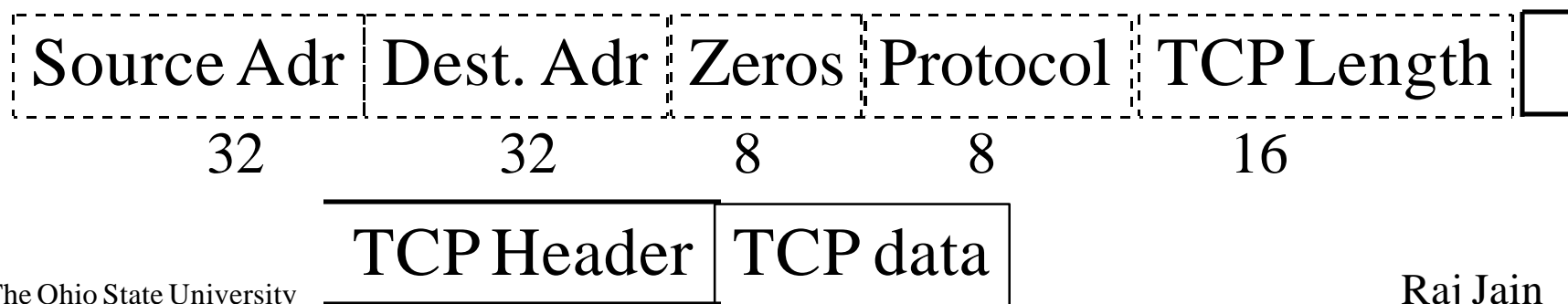
TCP Service Requests

❑ Unspecified passive open:
  Listen for connection requests from any user
❑ Full passive open:
  Listen for connection requests from specified user
❑ Active open: Request connection
❑ Active open with data: Request connection and transmit data
❑ Send: Send data
❑ Allocate: Issue incremental allocation for receive data
❑ Close: Close the connection gracefully
❑ Abort: Close the connection abruptly
❑ Status: Report connection status

TCP Service Responses

❑ Open ID: Informs the name assigned to the pending request

❑ Open Failure: Your open request failed

❑ Open Success: Your open request succeeded

❑ Deliver: Reports arrival of data

❑ Closing: Remote TCP has issued a close request

❑ Terminate: Connection has been terminated

❑ Status Response: Here is the connection status

❑ Error: Reports service request or internal error

TCP Mechanisms

❑ Connection Establishment

  ❑ Three way handshake

  ❑ SYN flag set $\Rightarrow$ Request for connection

SYN, ISN = 100

SYN, ISN = 350, Ack 101

Ack 351
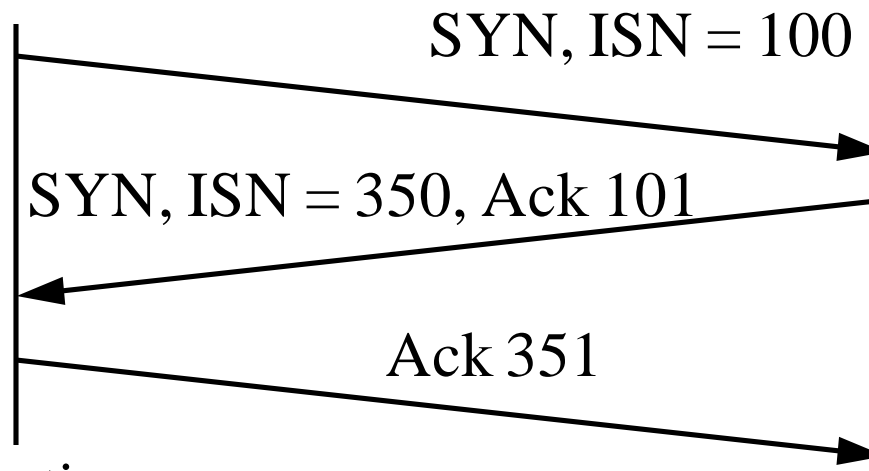
❑ Connection Termination

  ❑ Close with FIN flag set

  ❑ Abort

Three-Way Handshake

❏ 3-way handshake for opening and closing connections. Necessary and sufficient for unambiguity despite loss, duplication, and delay
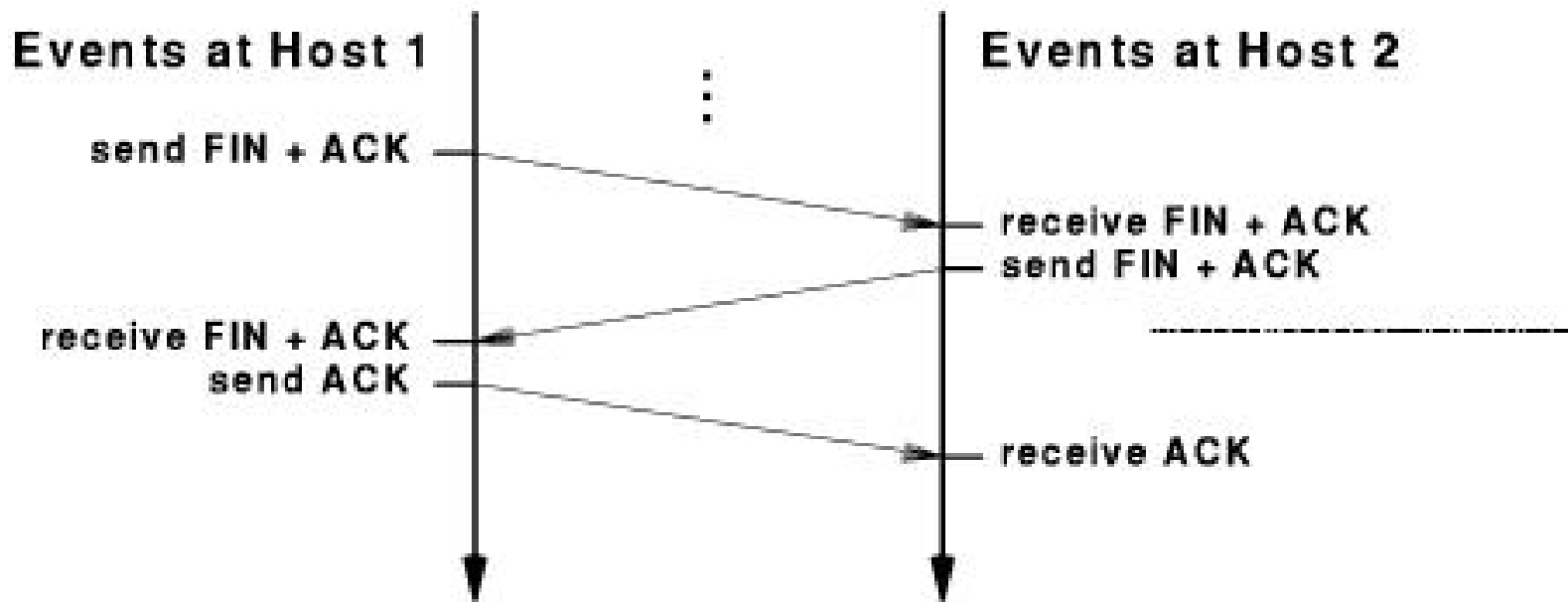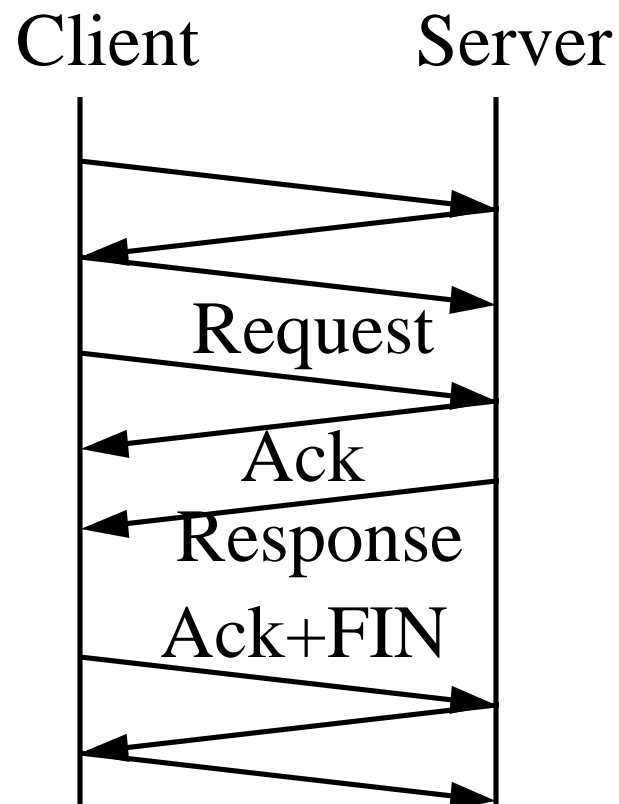

Fig 20.5

T/TCP: Transaction Oriented TCP

❑ Three-way handshake ⇒ Long delays for transaction-oriented (client-server) applications.
T/TCP avoids 3-way handshakes [RFC 1644].

Client   Server

Request

Ack

Response

Ack+FIN

Data Transfer

❑ Stream: Every byte is numbered modulo 2^{32}.

❑ Header contains the sequence number of the first byte

❑ Flow control: Credit = number of bytes

❑ Data transmitted at intervals determined by TCP
  Push $\Rightarrow$ Send now

❑ Urgent: Send this data in ordinary data stream with
  urgent pointer

❑ If TPDU not intended for this connection is received,
  the "reset" flag is set in the outgoing segment

Implementation Policies (Choices)

❑ Send Policy:

   Too little $\Rightarrow$ More overhead. Too large $\Rightarrow$ Delay
   Push $\Rightarrow$ Send now.

❑ Delivery Policy:

   May store or deliver each in-order segment.
   Push $\Rightarrow$ Send now.

❑ Accept Policy:

   May or May not discard out-of-order segments

Implementation Policies (Cont)

❑ Retransmit Policy:

First only

Retransmit all

Retransmit individual

(maintain separate timer for each segment)

❑ Ack Policy:

Immediate (no piggybacking)

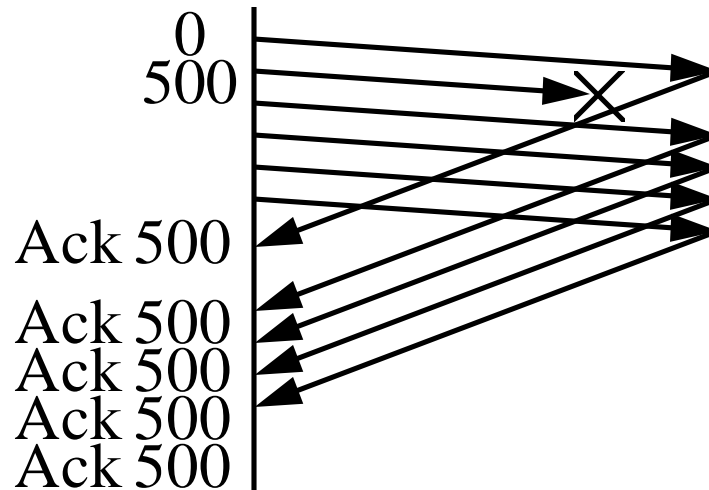Cumulative (wait for outgoing data or timeout)

Slow Start Flow Control

❑ Window = Flow Control Avoids receiver overrun

❑ Need congestion control to avoid network overrun

❑ The sender maintains two windows:
Credits from the receiver
Congestion window from the network
Congestion window is always less than the receiver window

❑ Starts with a congestion window of 1 segment
(one max segment size)
$\Rightarrow$ Do not disturb existing connections too much.

❑ Increase CWND by 1 every time an ack is received

Slow Start (Cont)

❏ If packets lost, remember slow start threshold to CWND/2
Set CWND to 1
Increment by 1 per ack until SS threshold
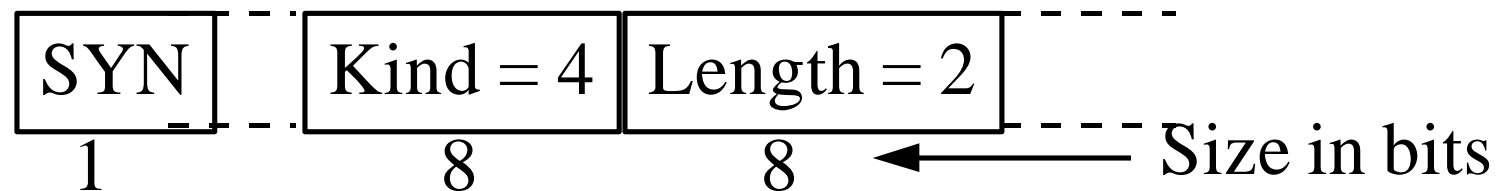Increment by 1/CWND per ack afterwards
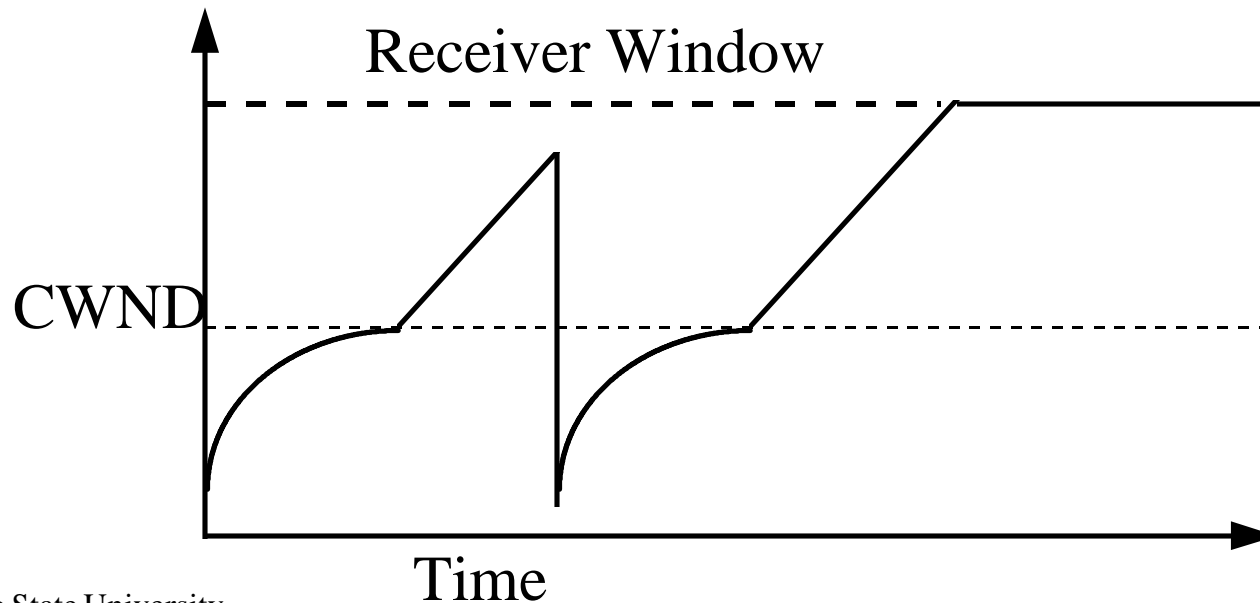
Fast Retransmit and Recovery

```
    0
  500  ────────────────────────→
        ──────────→ X
        ────────────────────────→
        ────────────────────────→
        ────────────────────────→
Ack 500 ←───────────────────
Ack 500 ←───────────────────
Ack 500 ←───────────────────
Ack 500 ←───────────────────
Ack 500 ←───────────────────
```
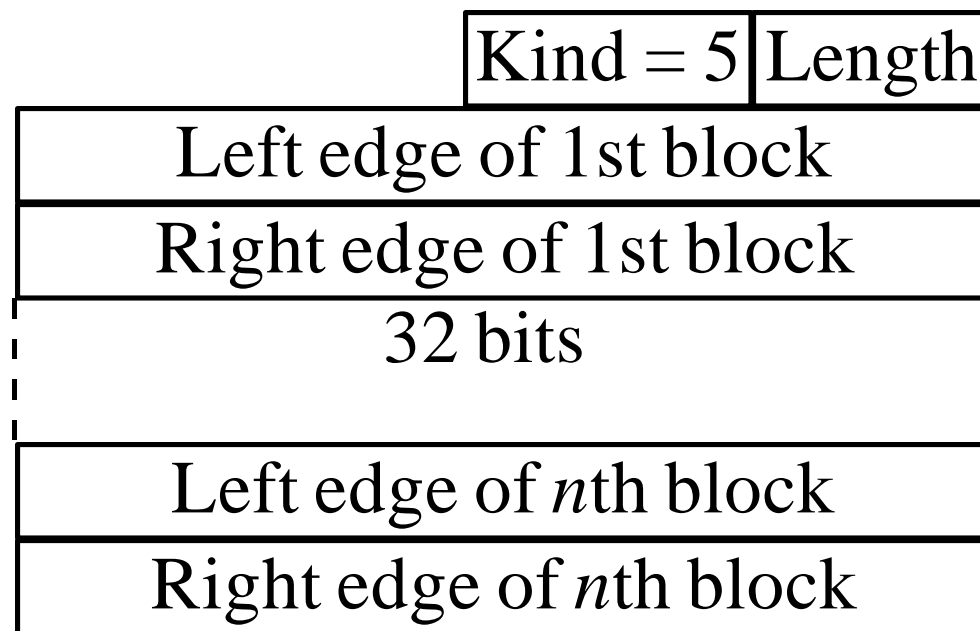
❑ If the same packet is acked 3 times, assume that the next packet has been last. Retransmit it right away. Retransmit only one packet.

❑ Helps if a single packet is lost.

❑ Does not help if multiple packets lost.

❑ Ref: Stevens, Internet draft

Selective Ack (SACK)

❑ Initial Negotiation: Sender to receiver: "sack permitted"

SYN		Kind = 4	Length = 2	
1		8	8	← Size in bits

❑ Selective Ack: Variable length. Receiver to sender

Kind = 5	Length

| Left edge of 1st block |
|:---:|
| Right edge of 1st block |
| 32 bits |
| Left edge of nth block |
| Right edge of nth block |

SACK (Cont)

❑ Left edge = 1st sequence number in this block

❑ Right edge = sequence number immediately after the last sequence number in this block

❑ Ack field meaning is same as before.
It is the next byte the receiver is expecting.

❑ When missing segments are received, ack field is advanced.

❑ Receiver can send SACK only if sender has "sack permitted" option in the SYN segment of the connection.

❑ Option Length = 8*n+2 byte for n blocks.
40 Bytes max options $\Rightarrow$ Max n = 4

SACK (Cont)

❑ Data receiver can discard SACKed (queued) data
   Sender must not discard data until acked.

❑ Example: 500 byte segments

Ack 5000

5000

5500

6000

Ack 5500
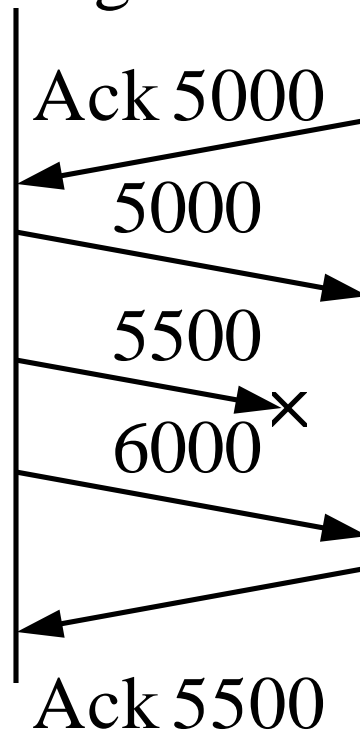
SACK = 6000-6500

Window Scaling Option

- Long Fat Pipe Networks (LFN): Satellite links Pronounced elephan(t)

- Need very large window sizes.

- Normally, Max window $= 2^{16} = 64$ KBytes

- Window scale option: W = W*2^Scale

Kind = 3	Length = 3	Scale

- Max window $= 2^{16} \times 2^{255}$

- Option sent only in SYN an SYN + Ack segment

- RFC 1323
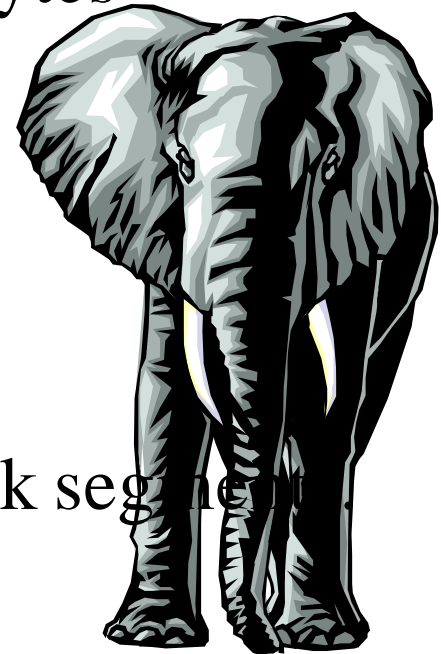
TCP/IP Tools

- nslookup
- ping
- finger
- traceroute
- People: whois, knowbot, netfind
- Files: archie, gopher, WWW
- Ref: RFC 1739, RFC 1470

User Datagram Protocol (UDP)

❑ Connectionless end-to-end service

❑ No flow control. No error recovery (no acks)

❑ Provides port addressing

❑ Error detection (Checksum) optional. Applies to pseudo-header (same as TCP) and UDP segment. If not used, it is set to zero.

❑ Used by network management

Source Port	Dest Port	Length	Check-sum
16	16	16	16

← Size in bits
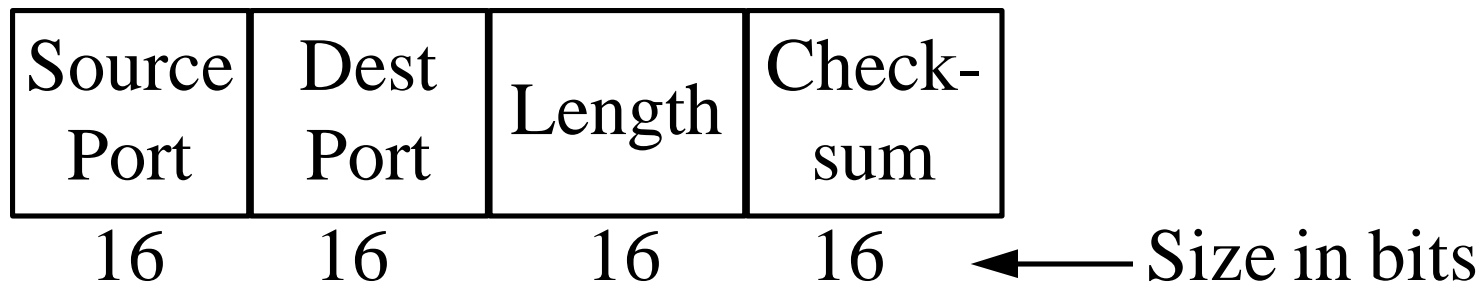
Summary

❑ TCP provides reliable full-duplex connections.

❑ TCP Streams, credit flow control, 3-way handshake

❑ Slow-start, Fast retransmit/recovery, SACK, Scaling

❑ UDP is connectionless and simple. No flow/error control.

References

- [RFC2018] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow, "TCP Selective Acknowledgment Options", 10/17/1996, 12 pages.

- [RFC1739] G. Kessler, S. Shepard, "A Primer On Internet and TCP/IP Tools", 12/22/1994, 32 pages.

- [RFC1693] T. Connolly, P. Amer, P. Conrad, "An Extension to TCP : Partial Order Service", 11/01/1994, 36 pages.

- [RFC1644] R. Braden, "T/TCP -- TCP Extensions for Transactions Functional Specification", 07/13/1994, 38 pages.

❑ [RFC1470] R. Enger, J. Reynolds, "FYI on a Network Management Tool Catalog: Tools for Monitoring and Debugging TCP/IP Internets and Interconnected Devices", 06/25/1993, 216 pages. (FYI 2)

❑ [RFC1379] R. Braden, "Extending TCP for Transactions -- Concepts", 11/05/1992, 38 pages.

❑ [RFC1347] R. Callon, "TCP and UDP with Bigger Addresses (TUBA), A Simple Proposal for Internet Addressing and Routing", 06/19/1992, 9 pages.

❑ [RFC1337] R. Braden, "TIME-WAIT Assassination Hazards in TCP", 05/27/1992, 11 pages.

- [RFC1323] D. Borman, R. Braden, V. Jacobson, "TCP Extensions for High Performance", 05/13/1992, 37 pages.

- [RFC1263] L. Peterson, S. O'Malley, "TCP Extensions Considered Harmful", 10/22/1991, 19 pages.

- [RFC1146] J. Zweig, C. Partridge, "TCP Alternate Checksum Options", 03/01/1991, 5 pages.

- [RFC1144] V. Jacobson, "Compressing TCP/IP headers for low-speed serial links", 02/01/1990, 43 pages.

❑ [RFC1110] A. McKenzie, "Problem with the TCP big window option", 08/01/1989, 3 pages.

❑ [RFC1106] R. Fox, "TCP big window and NAK options", 06/01/1989, 13 pages.

❑ [RFC1072] R. Braden, V. Jacobson, "TCP extensions for long-delay paths", 10/01/1988, 16 pages.

❑ [RFC0896] J. Nagle, "Congestion control in IP/TCP internetworks", 01/06/1984, 9 pages.

❑ [RFC0879] J. Postel, "TCP maximum segment size and related topics", 11/01/1983, 11 pages.

- [RFC0813] D. Clark, "Window and acknowledgment strategy in TCP", 07/01/1982, 22 pages.

- [RFC0793] J. Postel, "Transmission Control Protocol", 09/01/1981, 85 pages.

- [RFC0768] J. Postel, "User Datagram Protocol", 08/28/1980, 3 pages. (STD 6)

- V. Cerf and R. Kahn, "A Protocol for Packet Network Intercommunication", IEEE Transactions on Communications, Vol. COM-22, No. 5, pp 637-648, May 1974.

❑ V. Jacobson, "Congestion Avoidance and Control", Proceedings of SIGCOMM '88, Stanford, CA., August 1988.

Homework

❑ Read RFCs 0768 (UDP) 0793 (TCP), 1323 (Large Windows), 1470+1739 (TCP/IP Tools), 2018 (SACK)

❑ All RFCs up to 1949 are on the CD-ROM in the book Others can be found on http://ds.internic.net/

❑ Read internet draft:
Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", 03/15/1996, ftp://cnri.reston.va.us/internet-drafts/ draft-stevens-tcpca-spec-01.txt