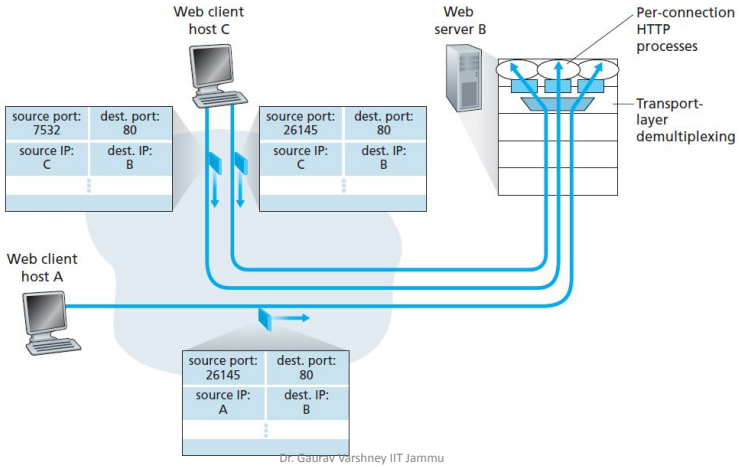
Transport Layer: Connection Oriented TCP



Dr. Gaurav Varshney IIT Jammu

Underlying Transport Layer Protocol [UDP/TCP]

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	Typically UDP
Streaming multimedia	typically proprietary	UDP or TCP
Internet telephony	typically proprietary	UDP or TCP
Network management	SNMP	Typically UDP
Routing protocol	RIP	Typically UDP
Name translation	DNS	Typically UDP

User Datagram Protocol [RFC 768]

UDP is said to be *connectionless as no handshaking is performed between the communicating processes.*

the host-side UDP adds header fields to the message and passes the resulting segment to the network layer. The network layer encapsulates the UDP segment into a datagram and sends the datagram

UDP provides

1. Finer application layer control
2. No connection establishment
3. No connection state
4. Small packet overhead

Psuedo header consists of the following things from the IP header.

- Source Address(IP)
- Destination Address(IP)
- Protocol Header(Like weather its TCP, or UDP or ICMP etc)
- length(TCP/UDP)
- Reserved 8 Bit

0110011001100000
0101010101010101
1000111100001100

The sum of first two of these 16-bit words is

0110011001100000
0101010101010101
1011101110110101

Adding the third word to the above sum gives

1011101110110101
1000111100001100
0100101011000010

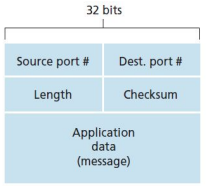


Figure 3.7 ♦ UDP segment structure

Dr. Gaurav Varshney IIT Jammu

At sender side,

The given data unit is divided into segments of 8 bits as-

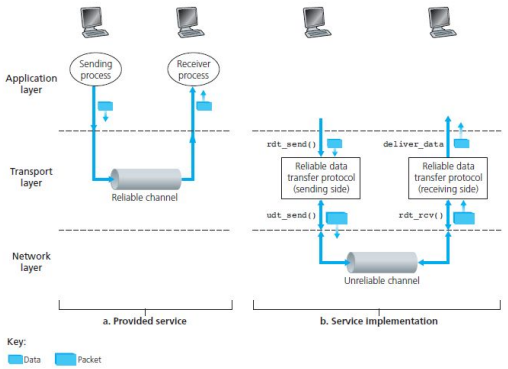
10011001	11100010	00100100	10000100

Now, all the segments are added and the result is obtained as-

- 10011001 + 11100010 + 00100100 + 10000100 = 100100011
- Since the result consists of 10 bits, so extra 2 bits are wrapped around.
- 00100011 + 10 = 00100101 (8 bits)
- Now, 1's complement is taken which is 11011010.
- Thus, checksum value = 11011010

Principles of Reliable Data Transfer [TCP]

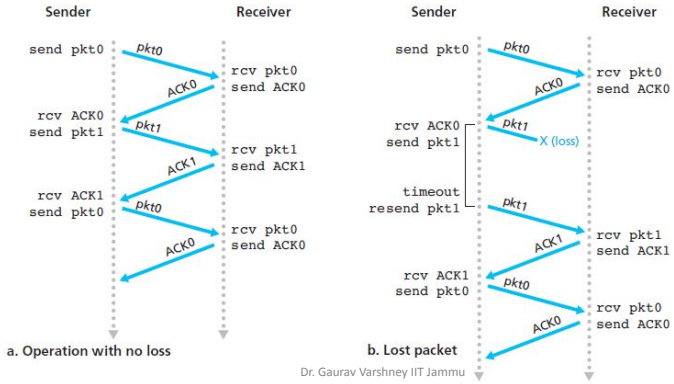
Mechanism	Use, Comments
Checksum	Used to detect bit errors in a transmitted packet.
Timer	Used to timeout/retransmit a packet, possibly because the packet (or its ACK) was lost within the channel. Because timeouts can occur when a packet is delayed but not lost (premature timeout), or when a packet has been received by the receiver but the receiver-to-sender ACK has been lost, duplicate copies of a packet may be received by a receiver.
Sequence number	Used for sequential numbering of packets of data flowing from sender to receiver. Gaps in the sequence numbers of received packets allow the receiver to detect a lost packet. Packets with duplicate sequence numbers allow the receiver to detect duplicate copies of a packet.
Acknowledgment	Used by the receiver to tell the sender that a packet or set of packets has been received correctly. Acknowledgments will typically carry the sequence number of the packet or packets being acknowledged. Acknowledgments may be individual or cumulative, depending on the protocol.
Negative acknowledgment	Used by the receiver to tell the sender that a packet has not been received correctly. Negative acknowledgments will typically carry the sequence number of the packet that was not received correctly.
Window, pipelining	The sender may be restricted to sending only packets with sequence numbers that fall within a given range. By allowing multiple packets to be transmitted but not yet acknowledged, sender utilization can be increased over a stop-and-wait mode of operation. We'll see shortly that the window size may be set on the basis of the receiver's ability to receive and buffer messages, or the level of congestion in the network, or both.



Dr. Gaurav Varshney IIT Jammu

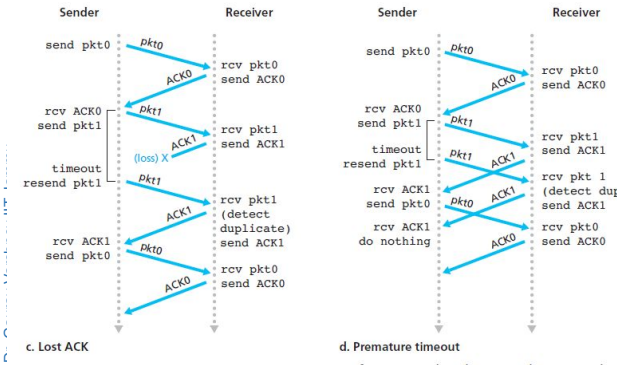
Principles of Reliable Data Transfer: Stop and Wait ARQ

Automatic Repeat ReQuest (**ARQ**) is a group of error – control protocols for transmission of data over noisy or unreliable communication network.



Dr. Gaurav Varshney IIT Jammu

Principles of Reliable Data Transfer: Stop and Wait ARQ



$$d_{trans} = \frac{L}{R} = \frac{8000 \text{ bits/packet}}{10^9 \text{ bits/sec}} = 8 \text{ microseconds}$$

$$U_{sender} = \frac{L/R}{RTT + L/R} = \frac{.008}{30.008} = 0.00027$$

Figure 3.16 ♦ Operation of `rdt3.0`, the alternating-bit protocol

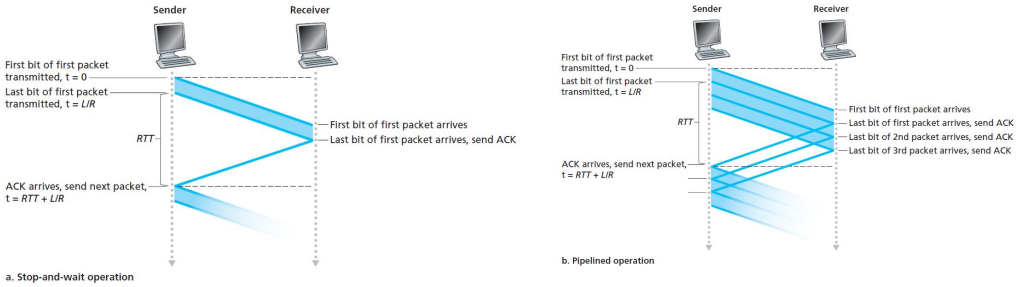
the sender was able to send only 1,000 bytes in 30.008 milliseconds, an effective throughput of only 267 kbps—even though a 1 Gbps link was available!

Dr. Gaurav Varshney IIT Jammu

Reliable Data Transfer: Go Back N ARQ

To improve the performance if the sender is allowed to transmit three packets before having to wait for acknowledgments, the utilization of the sender is essentially tripled. Since the many in-transit sender-to receiver packets can be visualized as filling a pipeline, this technique is known as **pipelining**

In a **Go-Back-N (GBN) protocol**, the sender is allowed to transmit multiple packets (when available) without waiting for an acknowledgment, but is constrained to have no more than some maximum allowable number, *N,* of unacknowledged packets in the pipeline



Dr. Gaurav Varshney IIT Jammu

Principles of Reliable Data Transfer: Go Back N ARQ

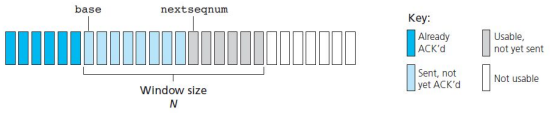


Figure 3.19 ♦ Sender's view of sequence numbers in Go-Back-N

The GBN sender must respond to three types of events:

• *Invocation from above, Receipt of an ACK, Λ timeout event*

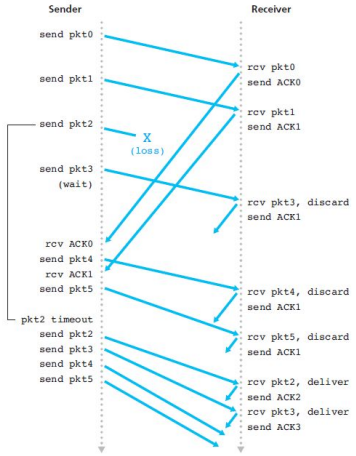


Figure 3.22 ♦ Go-Back-N in operation

Principles of Reliable Data Transfer: Go Back N ARQ

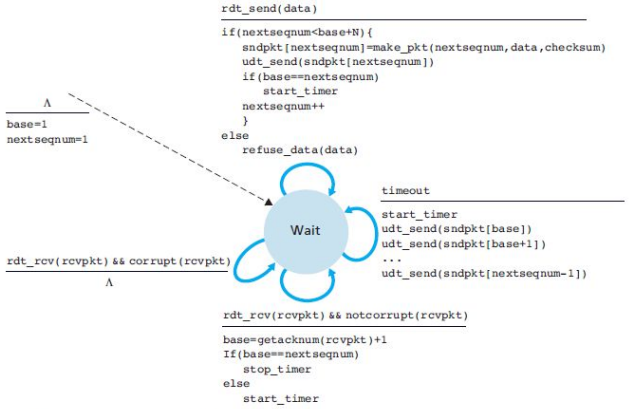


Figure 3.20 ♦ Extended FSM description of GBN sender

Principles of Reliable Data Transfer: Go Back N ARQ

The advantage of this approach is the simplicity of receiver buffering—the receiver need not buffer *any* out-of-order packets. Thus, while the sender must maintain the upper and lower bounds of its window and the position of `nextseqnum` within this window, the only piece of information the receiver need maintain is the sequence number of the next in-order packet. This value is held in the variable `expectedseqnum`

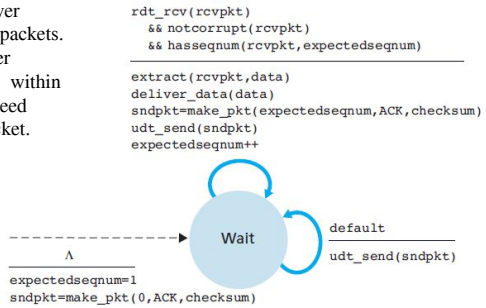


Figure 3.21 ♦ Extended FSM description of GBN receiver

Principles of Reliable Data Transfer: Selective Repeat ARQ

A single packet error can cause GBN to retransmit a large number of packets, many unnecessarily

As the name suggests, selective-repeat protocols avoid unnecessary retransmissions by having the sender retransmit only those packets that it suspects were received in error (that is, were lost or corrupted) at the receiver. This individual, as needed, retransmission will require that the receiver *individually* acknowledge correctly received packets

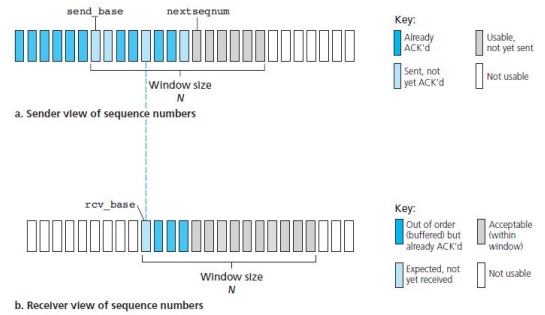


Figure 3.23 ♦ Selective-repeat (SR) sender and receiver views of sequence-number space

Principles of Reliable Data Transfer: Selective Repeat ARQ

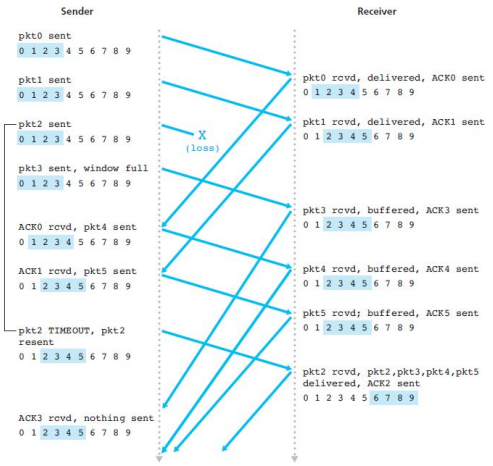


Figure 3.26 ♦ SR operation

Selective Repeat ARQ: Window size and Sequence Number Space

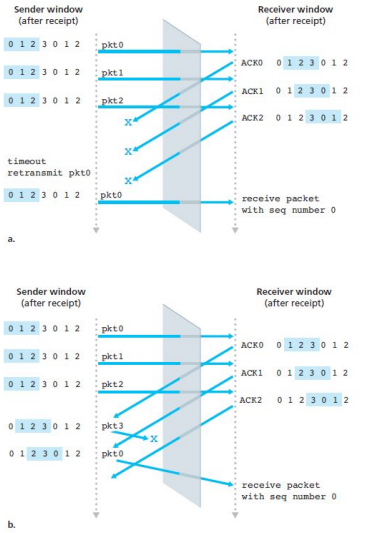


Figure 3.27 ♦ SR receiver dilemma with too-large windows: A new packet or a retransmission?

TCP: Transmission Control Protocol

- TCP is said to be **connection-oriented** because before one application process can begin to send data to another, the two processes must first "handshake" with each other—that is, they must send some preliminary segments to each other to establish the parameters of the ensuing data transfer
- In fact, the intermediate routers are completely oblivious to TCP connections; they see datagrams, not connections.

TCP: Packet Format, Segmentation, MSS

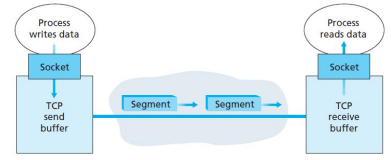
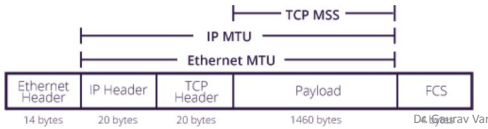
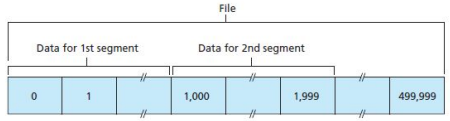


Figure 3.28 ♦ TCP send and receive buffers

a TCP connection consists of buffers, variables, and a socket connection to a process in one host, and another set of buffers, variables, and a socket connection to a process in another host. As mentioned earlier, no buffers or variables are allocated to the connection in the network elements (routers, switches, and repeaters) between the hosts.





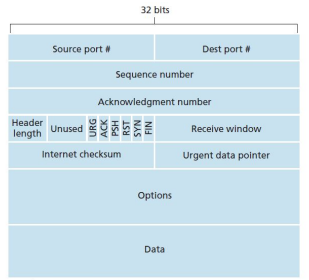


Figure 3.29 ♦ TCP segment structure

TCP: Packet Format, Segmentation, MSS

The 32-bit **sequence number field** and the 32-bit **acknowledgment number field** are used by the TCP sender and receiver in implementing a reliable data transfer service,

The 16-bit **receive window** field is used for flow control. We will see shortly that it is used to indicate the number of bytes that a receiver is willing to accept.

The optional and variable-length **options field** is used when a sender and receiver negotiate the maximum segment size (MSS) or as a window scaling factor for use in high-speed networks. A time-stamping option is also defined.

The **flag field** contains 6 bits. The **ACK bit** is used to indicate that the value carried in the acknowledgment field is valid; that is, the segment contains an acknowledgment for a segment that has been successfully received. The **RST**, **SYN**, and **FIN** bits are used for connection setup and teardown, as we will discuss at the end of this section. Setting the **PSH** bit indicates that the receiver should pass the data to the upper layer immediately. Finally, the **URG** bit is used to indicate that there is data in this segment that the sending-side upper-layer entity has marked as "urgent." The location of the first byte of this urgent data (relative to sequence number) is indicated by the 16-bit **urgent data pointer field**. TCP must inform the receiving-side upper-layer entity when urgent data exists and pass it a pointer to the end of the urgent data.

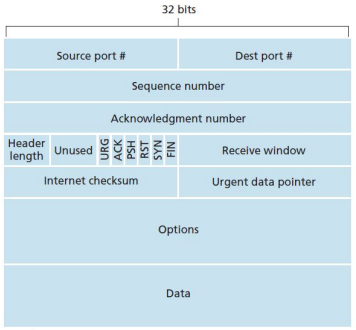
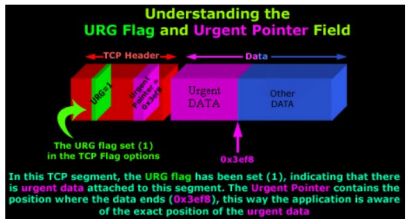
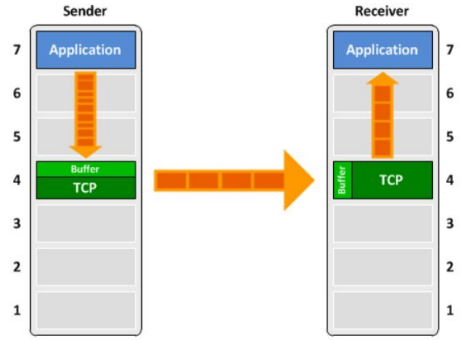


Figure 3.29 ♦ TCP segment structure

https://packetlife.net/blog/2011/mar/2/tcp-flags-psh-and-urg/

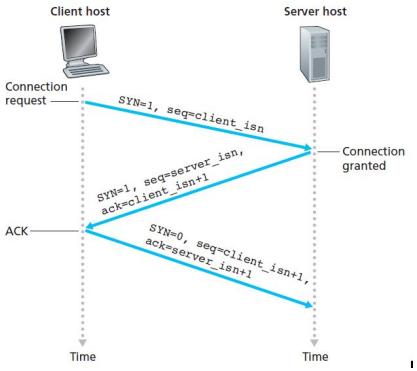


http://www.firewall.cx/networking-topics/protocols/tcp/137-tcp-window-size-checksum.html



https://packetlife.net/blog/2011/mar/2/tcp-flags-psh-and-urg/

TCP Connection Establishment and Termination



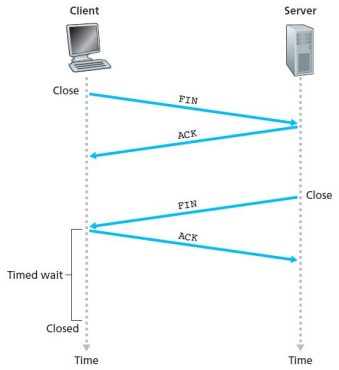
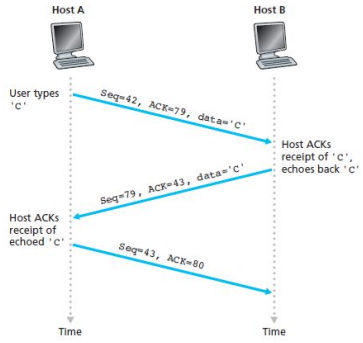


Figure 3.40 ♦ Closing a TCP connection

Sequence and acknowledgment numbers for a simple Telnet application over TCP



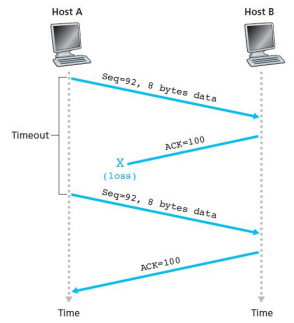


Figure 3.34 ♦ Retransmission due to a lost acknowledgment