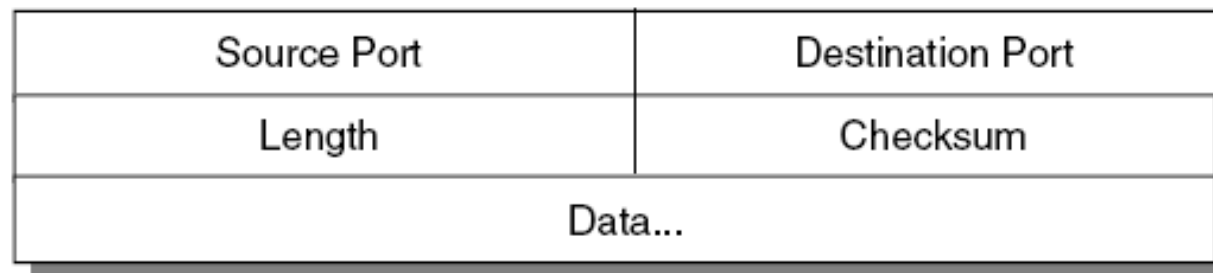


# [ Transport Layer ]

- v For a connection on a host (single IP address), there exist many entry points through which there may be many-to-many connections. These are called **ports**.
- v A port is a 16-bit number used by the host-to-host protocol to identify to which higher level protocol or application process it must deliver incoming messages.
- v Internet ports 1-1023 are considered as *well-known*
  - γ Controlled by the IANA
  - γ 80 for http, 20/21 for ftp, 23 for telnet
- v When using TCP, it is a connection-based protocol thus a client will make a connection to a server
  - γ A connection is established when a connection tuple is established on each side (called *socket tuple*)
    - v *<protocol, local-addr, local-port, foreign-addr, foreign-port>*
  - γ Client uses ephemeral ports 1024-65535

# [ User Datagram Protocol ]

- v Simply provides multiplexing/de-multiplexing capabilities on top of IP
  - γ Introduces ports on top of IP
- v UDP Header (diagram taken from IBM Redbook):



- v Length is length of whole packet including header
- v Checksum is over pseudo-IP header, UDP header and data
- v UDP is used by TFTP, DNS, RPC, SNMP and LDAP

# [ Transmission Control Protocol ]

- v TCP offers
  - γ Stream Data Transfer
    - v Application sees a stream of bytes
    - v TCP groups the data into packets and vice-versa
  - γ Reliability
    - v A sequence number to each byte is transmitted to the receiver, and sender expects ACK packet for it
      - γ Only sequence number of first byte is sent, since a **segment** will be received entirely
    - v Receiver will use sequence number to re-order packets if needed
  - γ Flow control
    - v When sending back ACK, receiver will also indicate remaining buffer size

# [ TCP (cont) ]

---

- γ Multiplexing
- γ Logical Connections
  - v A socket tuple is established before data transfer
- γ Full duplex
  - v Bi-directional concurrent data transfers

# [ Window Principle ]

- v Basic Reliability Protocol

- v Send Data Packet
- v Expect ACK for Packet
- v Send Next Data Packet

- γ Expensive in terms of bandwidth

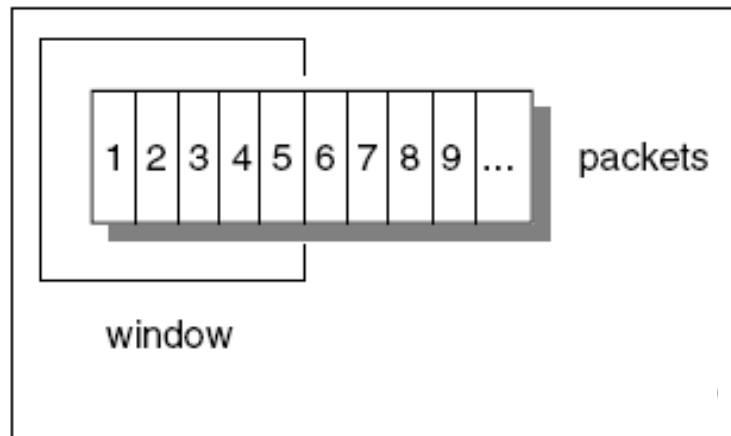
- v Better Reliability Protocol

- γ Send all Packets within Window (setting timeout for each)
- γ Slide window for every ACK'd packet
- γ Receiver will send sequence ACK of last well-received packet for each packet received

- v Scenarios:

- γ Lost Packets
- γ Lost ACK

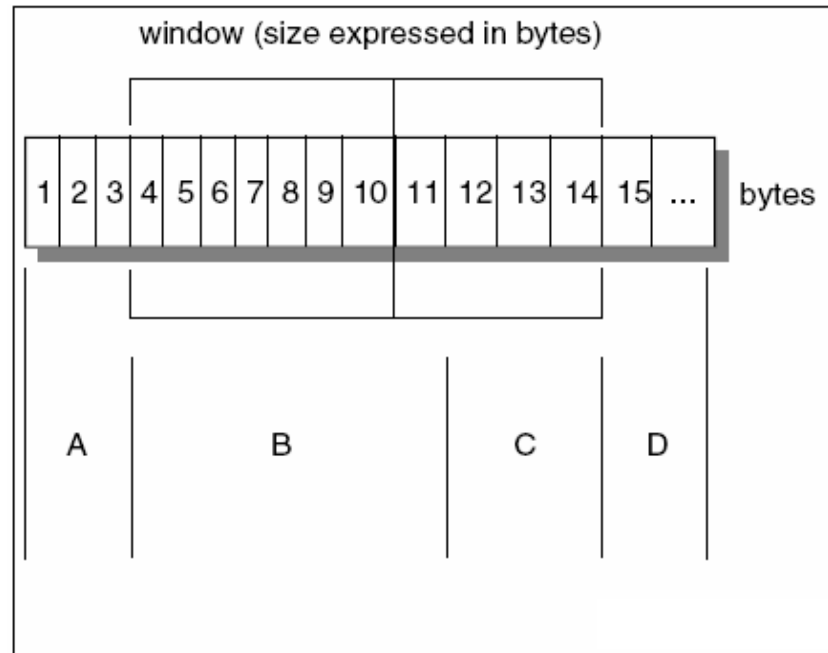
(diagram taken from IBM Redbook)



# [ Window Principle (cont) ]

- v Window Principle provides
  - γ Data Reliability
  - γ Better Throughput
  - γ Flow-control
    - v Receiver can delay ACK
- v In TCP
  - γ Sequence numbers are at the byte level and one sequence sent per TCP Segment
  - γ Window Size is in terms of bytes also
  - γ Window Size is transmitted at connection initiation
  - γ Each ACK will also contain the amount of data the receiver is ready to deal with

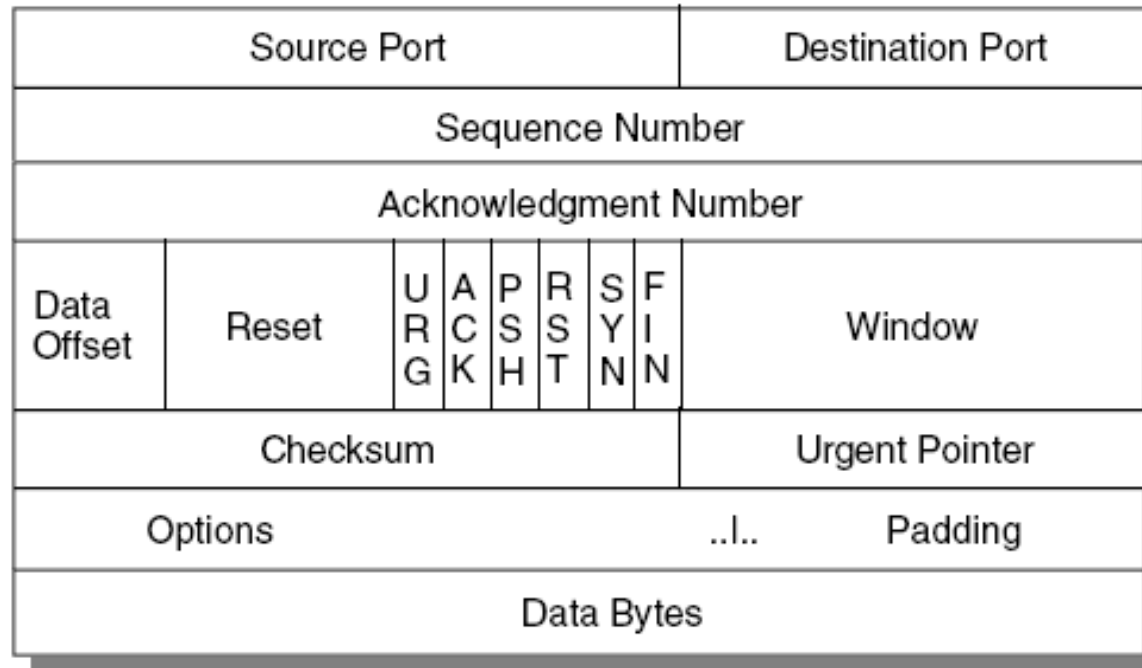
# [ TCP Windowing ]



- A: Bytes that are transmitted and have been acknowledged.
- B: Bytes that are sent but not yet acknowledged.
- C: Bytes that can be sent without waiting for any acknowledgment.
- D: Bytes that cannot be sent yet.

(diagram taken from IBM Redbook)

# [ TCP Segment ]



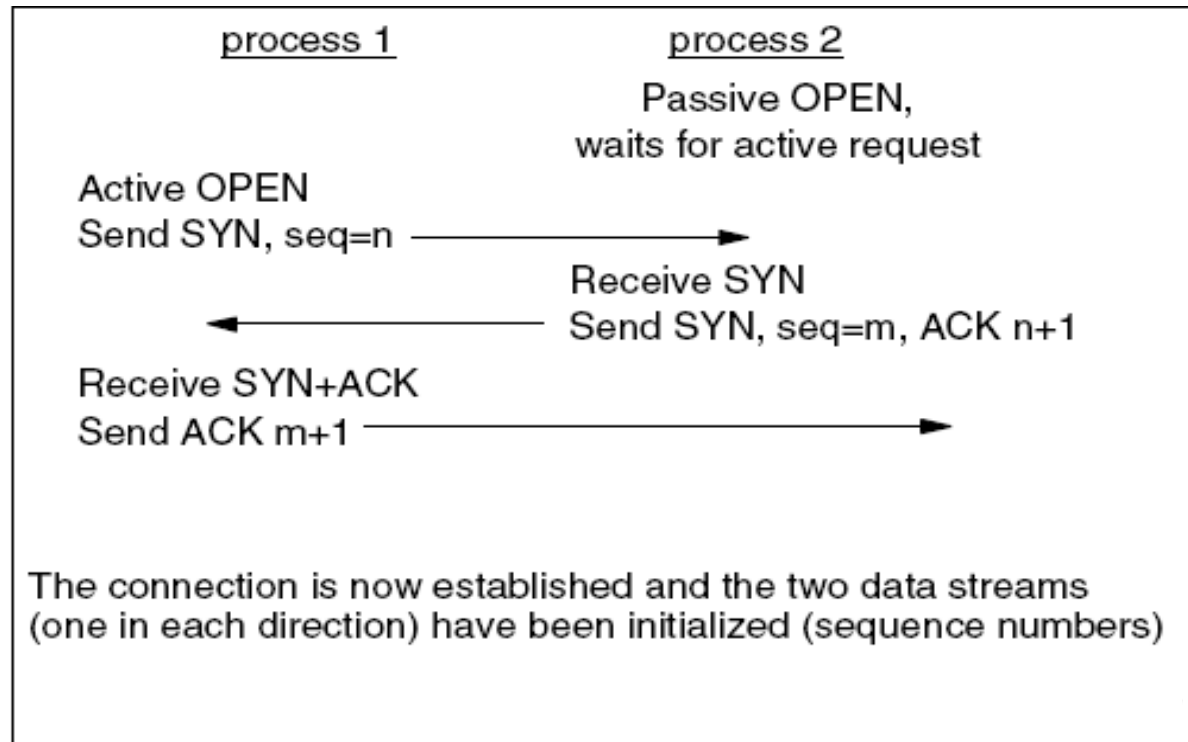
(diagram taken from IBM Redbook)

- v Source Port, Destination Port
- v Sequence Number: seq number of first data byte in this segment, unless SYN is set where first data byte is n+1
- v ACK Number: If ACK is set, number of next sequence number receiver is expecting (note ACK piggy-back)

# [ TCP Segment (Cont) ]

- v Data Offset: Number of 32 bits in header
- v Fields:
  - r URG: urgent Pointer is valid
  - r ACK: Ack number if valid
  - r PSH: Push Function (used to transmit data before a whole segment is filled)
  - r RST: Resets Connection
  - r SYN: Synchronizes the sequence numbers
  - r FIN: No more data from sender
- v Window: In ACK Segments, number of bytes sender of packet is willing to accept beyond ACK number
- v Checksum: Over pseudo IP header, TCP header and TCP data
- v Urgent Pointer: Used for urgent data
- v Options: Several that are rarely used apart from *window scale*

# [ Three-way handshake ]



- v Note above is packet-loss safe (diagram taken from IBM Redbook)
- v To close a connection, each side has to send the FIN packet
- r Note that each side is closed independently since data might still need to be transmitted on one side while the other has closed connection