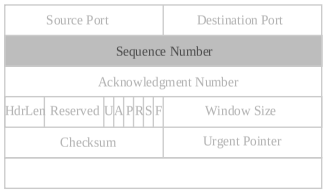
Introduction to TCP

- RFC 793 (+ RFC1122) (+ others)
- Transmission Control Protocol

The Transmission Control Protocol (TCP) is intended for use as a highly reliable host-to-host protocol between hosts in packet-switched computer communication networks, and in interconnected systems of such networks.

TCP is a connection-oriented, end-to-end reliable protocol designed to fit into a layered hierarchy of protocols which support multi-network applications.

TCP Header - Sequence Number



- 32 bit number
- Every byte transmitted is allocated a number
- Sequence numbers cycle around

TCP Specification

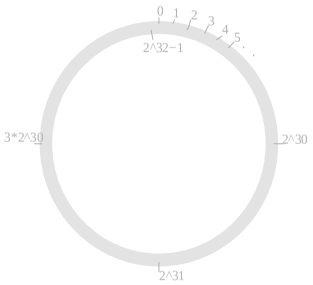
The TCP must recover from data that is damaged, lost, duplicated, or delivered out of order by the internet communication system.

This is achieved by assigning a sequence number to each octet transmitted, and requiring a positive acknowledgment (ACK) from the receiving TCP.

If the ACK is not received within a timeout interval, the data is retransmitted.

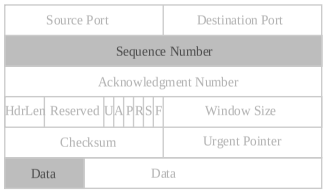
At the receiver, the sequence numbers are used to correctly order segments that may be received out of order and to eliminate duplicates.

TCP Sequence Number Space



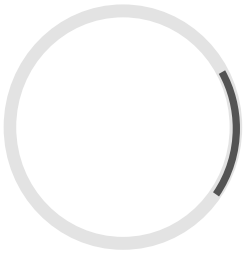
- Sequence numbers run from 0 to 2^32 - 1
- 0, 1, 2, 3, ..., 2^32-2, 2^32-1, 0, 1, 2, ...

TCP Sequenced Data



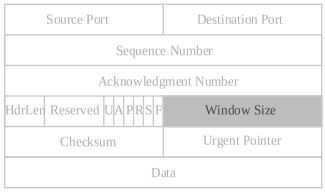
- Sequence number in header
  - identifies the first data byte in the packet
- Sequence numbers of later data bytes
  - obtained by arithmetic

TCP Window



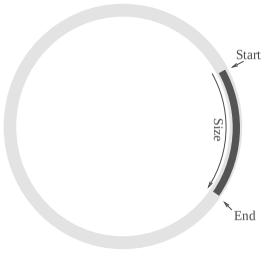
- A segment of the sequence number space
- Data can only be transmitted in the window
- Allows receiver to control buffer space consumed

TCP Window Size



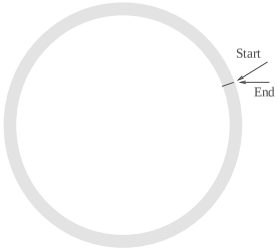
- Window Size sent from receiver to sender of data
- Indicates the current window size available

TCP Window Parameters



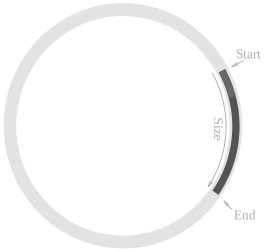
- Window Size: TCP header of received packet
- Start (later)
- End calculated from Start + Size (-1)

A better representation (scaling)



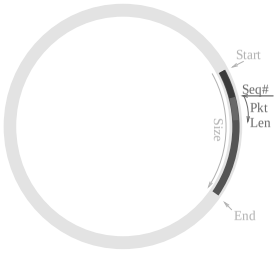
- 2^16 bytes max window size
- 2^32 bytes sequence space
  - Max window approx 1/2 of 1/100 of 1 degree

Filling the Window



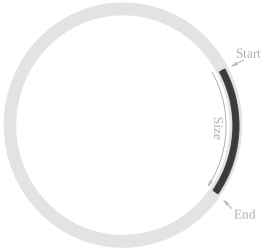
- Data transmitted gradually fills the window

Sending data



- TCP sequence number, and length of data
  - IP length - IP header len - TCP header len

Filling the Window



- Must not send past the end of the window
  - Might mean sending a small packet
- Window fills, no more data can be sent

TCP Specification

Transmission is made reliable via the use of sequence numbers
and acknowledgments.

Conceptually, each octet of data is assigned a sequence number.

The sequence number of the first octet of data in a segment is transmitted
with that segment and is called the segment sequence number.

Segments also carry an acknowledgment number which is
the sequence number of the next expected data octet of transmissions
in the reverse direction.

When the TCP transmits a segment containing data, it puts a copy
on a retransmission queue and starts a timer; when the acknowledgment
for that data is received, the segment is deleted from the queue.

If the acknowledgment is not received before the timer runs out,
the segment is retransmitted.

TCP Specification

A fundamental notion in the design is that every octet of data sent over a TCP connection has a sequence number.

Since every octet is sequenced, each of them can be acknowledged.

The acknowledgment mechanism employed is cumulative so that an acknowledgment of sequence number X indicates that all octets up to but not including X have been received.

This mechanism allows for straight-forward duplicate detection in the presence of retransmission.

Numbering of octets within a segment is that the first data octet immediately following the header is the lowest numbered, and the following octets are numbered consecutively.

TCP Specification

The sender of data keeps track of the next sequence number to use
in the variable SND.NXT.

The receiver of data keeps track of the next sequence number to expect
in the variable RCV.NXT.

The sender of data keeps track of the oldest un-acknowledged
sequence number in the variable SND.UNA.

If the data flow is momentarily idle and all data sent has been acknowledged
then the three variables will be equal.

When the sender creates a segment and transmits it
the sender advances SND.NXT.

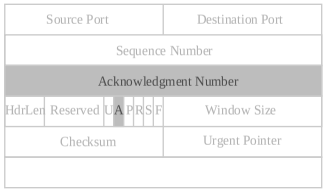
When the receiver accepts a segment
it advances RCV.NXT and sends an acknowledgment.

When the data sender receives an acknowledgment it advances SND.UNA.

The extent to which the values of these variables differ
is a measure of the delay in the communication.

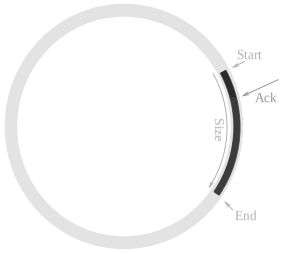
The amount by which the variables are advanced
is the length of the data in the segment.

Acknowledgements



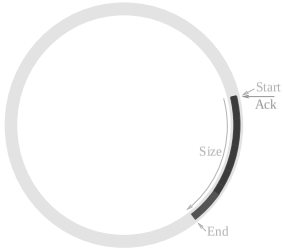
- Sent from Receiver to Sender of data
- Indicates the sequence number of first data byte
  - not yet received
- Valid only when A (ACK) flag is set

Ack Arrives



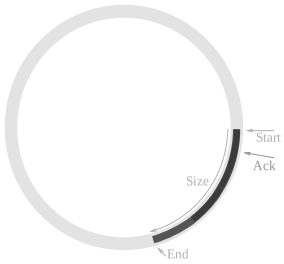
- Acknowledgement arrives for first transmitted packet

Window Movement



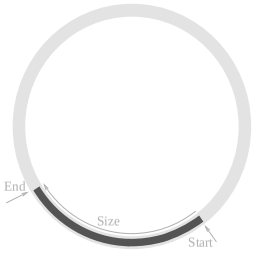
- Sender can discard buffered data
  - Recipient now has it
- Window advances
  - Window start is acknowledgement number

The Next Steps



- Sender can now transmit more data
- Another ACK should arrive
- Window advances again

Quiescent State



- All transmitted data acknowledged
- No more data to send (for now)
  - Idle state

A Perfect Connection

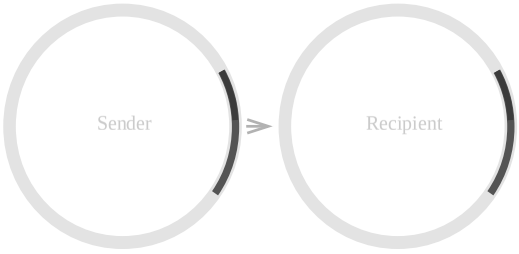
- Sender picks sequence number
- Receiver sets window size
- Sender sends up to window size bytes of data
- Receiver acknowledges data and advances window
- Sender sends more data
    - until no more to send
- Receiver acknowledges final data

- Connection establishment & termination
  - Later...

TCP Notes

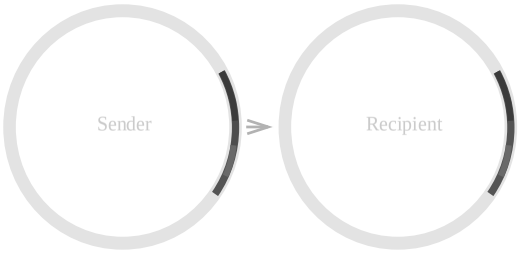
- TCP connections are bi-directional
  - Every packet has sequence number & acknowledgement number
    - Sequence number indicates which data are in this packet (if any)
    - Or where in sequence next data will go
    - Acknowledgement indicates which data are expected to be received next
  - Can acknowledge data received,
    - and send answer (any reply)
    - in the same packet
- Window size sent in every packet
  - Can be varied as buffer space at receiver varies

Lost Data



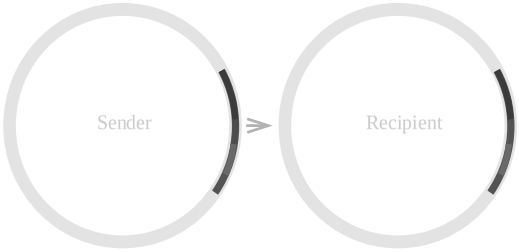
- Sender has transmitted some data to receiver
  - data arrived
  - acknowledgements not yet received

Lost Data (2)



- Sender transmits another packet
  - This one is lost somewhere
- Sender transmits its next packet
  - This one arrives

Retransmission



- Sender never receives acknowledgement for the lost data, or anything after it
- Sender retransmits the lost packet
  - the one with initial sequence number equal to last ack received
- Receiver receives that, acknowledges it, and the other packet that was received.

Alternate representation

[Seq=0,Ack=100,Len=40,WS=200] ->

<-

[S

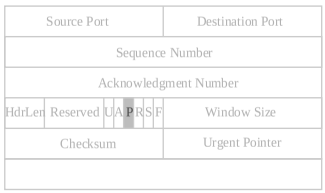
[Seq=40,Ack=100,Len=40,WS=200] ->

[Seq=80,Ack=100,Len=40,WS=200] ->

<-

[S

TCP PUSH



- Instruction from sending TCP to receiving TCP
  - that data should be delivered rather than buffered
- NOT a record marker
  - no particular place in the packet is indicated
- Almost Obsolete
  - but must still be set