Module 5 – Part 1

Vipin Das
SAINTGITS College of Engineering

Transport layer

The basic functionality of transport layer is the delivery of data between processes.

The transport layer collects data from the layer above it and creates a transport layer segment.

This is passed to the network layer.

Every application that is transmitting data across any network will be given a unique identifier.

This identifier is known as **port number**.

Port number is a logical concept to identify the application.

A combination of the node IP address and port number along with the transport layer protocol defines a **socket**.

When an application is developed the sending side and the receiving side needs to have a port number.

16 bits are used to represent a port number.

Numbers 0 to 1024 are reserved for common applications.

If port numbers are required for communication, how do I know the port numbers of Google,Facebook,.....??

There are port numbers reserved.

Common applications like webservers always use the same port number.

Port Number	Usage
20	File Transfer Protocol (FTP) Data Transfer
21	File Transfer Protocol (FTP) Command Control
22	Secure Shell (SSH)
23	Telnet - Remote login service, unencrypted text messages
25	Simple Mail Transfer Protocol (SMTP) E-mail Routing
53	Domain Name System (DNS) service
80	Hypertext Transfer Protocol (HTTP) used in World Wide Web
110	Post Office Protocol (POP3) used by e-mail clients to retrieve e-mail from a server
119	Network News Transfer Protocol (NNTP)
123	Network Time Protocol (NTP)
143	Internet Message Access Protocol (IMAP) Management of Digital Mail
161	Simple Network Management Protocol (SNMP)
194	Internet Relay Chat (IRC)
443	HTTP Secure (HTTPS) HTTP over TLS/SSL

Transport layer services

Transport layer protocol defines connection oriented and connection less services.

In connection less service the data is transmitted without ensuring any reliability.

Ex:- UDP protocol

Commonly used in applications which require realtime transmission of data.

In connection oriented service reliability of data is ensured.

Ex:- TCP Protocol.

Reliability in Transport layer

Reliablity is ensured by making use of acknowlegments.

There are mechanims ensure a connection setup before the actual data delivery.

Reliable protocols also ensures flow control by adjusting the number of data packets that are being sent.

Congestion control mechanisms are also employed by the reliable protocols.

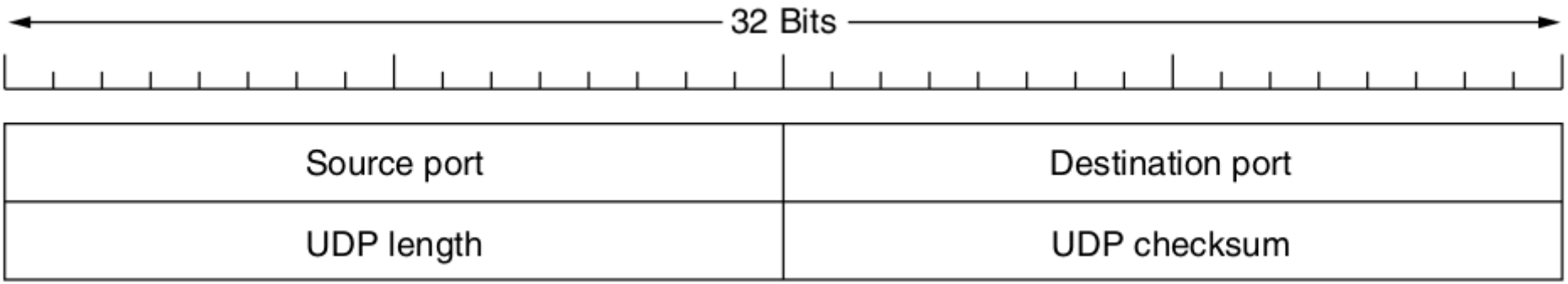
UDP -User Datagram Protocol

A protocol with the simplest header structure.

Contains a 8 byte header .

Data from application layer is added along with the header.

Although there is no size limitation for the data,there are restrictions imposed based on the upper layer services.



UDP is a send and forget protocol.

Used by applications which require quick and realtime delivery of data.

There is no retransmission of the data.

Even if there are errors in the data ,the only option that the receiver is having is to drop the data.

The maximum size of UDP data is controlled by the applications.

Things to Check

A protocol named Real Time Transport Protocol (RTP) is used along with UDP to deliver real time data .

Has got provision to arrange data segments in order.

Transmission Control Protocol [TCP]

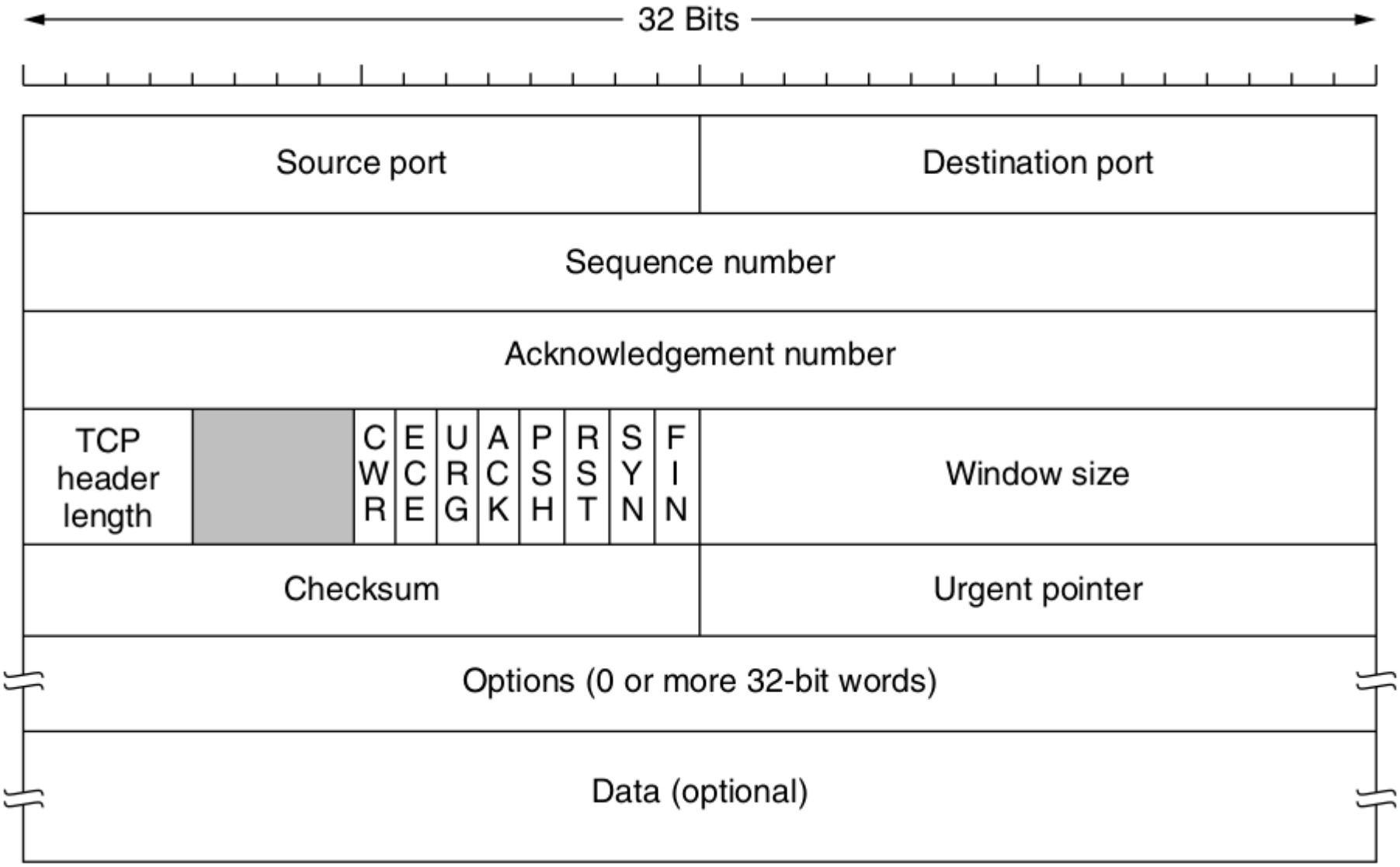
TCP is a connection oriented protocol used when the reliability of data is important.

TCP protocol require that the sender and receiver establish a connection before the actual data transfer.

This connection will be closed after the final byte is transmitted.

TCP protocol includes mechanisms for retransmissions and congestion control

TCP Header structure



Source port and destination port – 16 bit port number to determine the process at the sender and the receiver.

Sequence number and acknowledgment number – 32 bit sequence number and ACK number. ACK is cumulative.

TCP Header length – The length of the TCP header.

A 4 bit field is not used and is reserved for future purpose.

A set of flags are defined

CWR and ECE flags are used to define congestion

The sender will set Explicit Congestion notification Echo bit as 1 to indicate congestion.

The receiver will set Congestion Window reduce bit as 1 to let the sender know that the sending pace has been reduced.

If URG*ent* bit is set,the receiver is informed that from the offset mentioned in urgent pointer field is important data.

If PSH bit is set the reciver is told not to buffer the data and it should be delivered immediately to the process.

If ACK flag is set the acknowledgment number is valid.

RST,SYN and FIN are used in connection setup and closing the connection.[Upcoming slides]

The window size field indicates the maximum number of bytes the receiver is expecting.

The value is adjusted according to the load at the receiver.

A checksum field is provided for extra reliablity of the header ,data and options provided.

Optional informations can be provied.

Ex:- Timestamp --> So as to compute the round trip time

Selective ACK --> So as to inform the sender that the data segments with the specified sequence numbers are received.The Acknowldegment number in the main header is cumulative.

Why is the data part mentioned as optional ??

There could be situations in which TCP headers are sent across with out data.

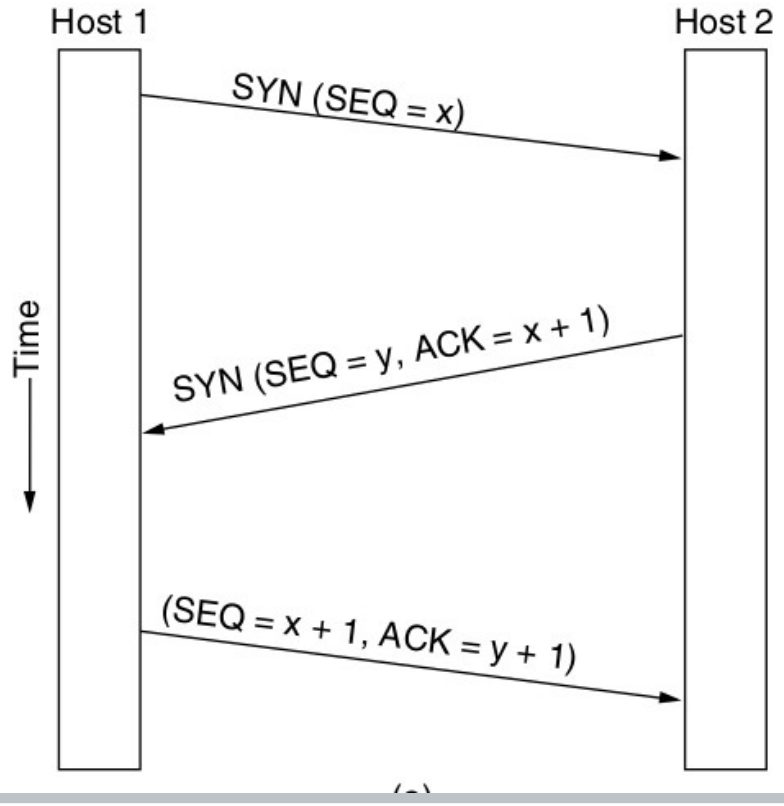
Ex:- During connection set up and closing.

The reciver may send a TCP header with a modified window size.

TCP Connection setup

The sequence numbers are mostly chosen by a random number generator.

It certainly cannot start with a zero.



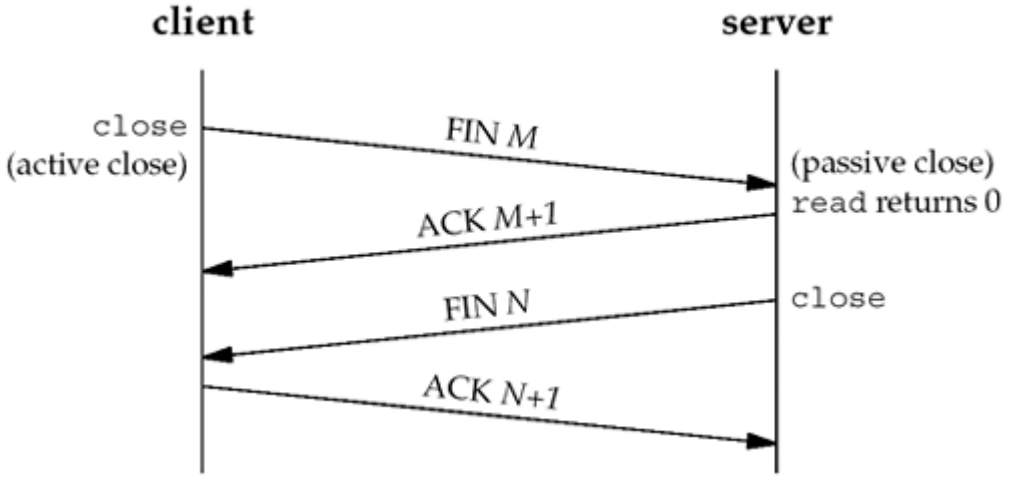
TCP Connection release

TCP connections are full duplex.

Any one of the party can send the FIN segment.

The data from other side still continues.

Full connection is closed when both the parties send FIN.



TCP RST flag

RST indicate termination of connection.

It may be as the node is busy or due to any mishap.

Compared to TCP FIN, RST is rather an ungrafecul way to exit.

No confirmation is also required.

Things to check

- TCP Syn flood attack.

- Man in the middle attacks on TCP connection establishment.

-RST attacks.

TCP Connection management policy

TCP servers are created to LISTEN until they hear the conncection request from the client.

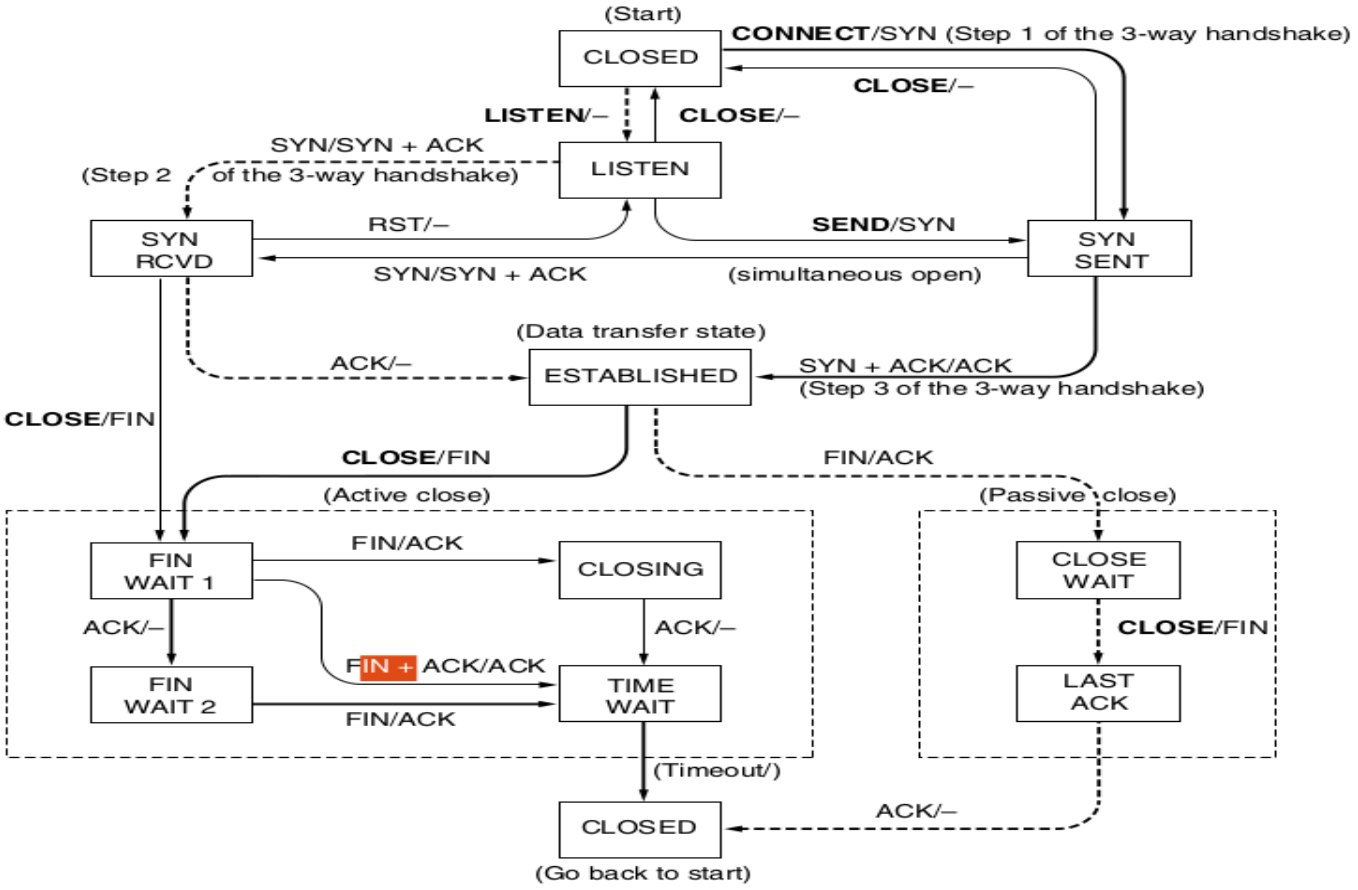
Also known as Blocking receiver.

The state will only change once the client successfully finishes the connection setup.

After this only the next logic of the server side will be implemented.

The states are in Boxes.

The arrows have event/action



TCP retransmission

TCP Employs retransmission techniques to compensate for loss of data. [Sliding window]

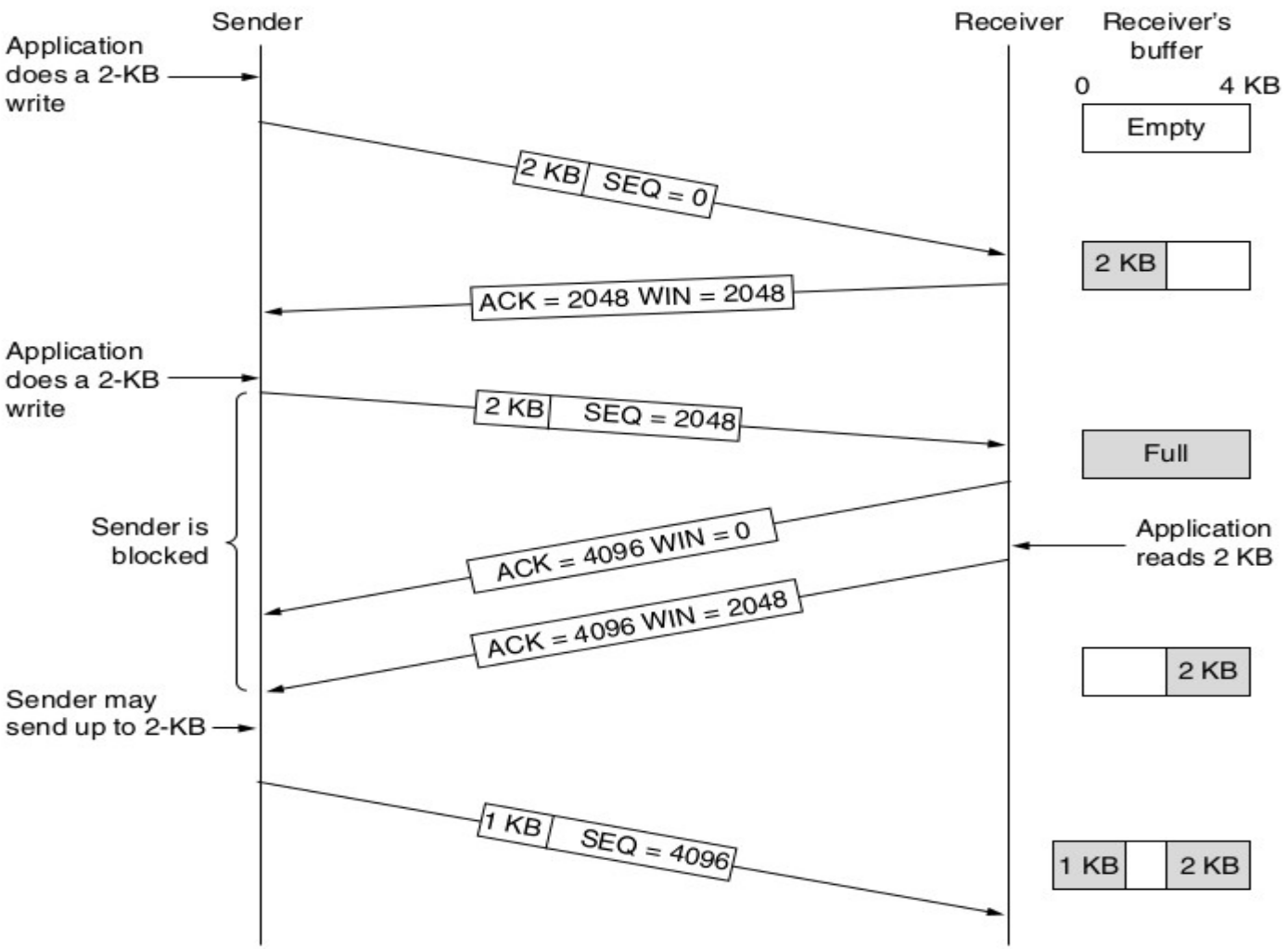
The window size field in the protocol header helps to *slide* the window based on the ACK.

There are timers which are started when the segment(s) are transmitted.

The ACK are expected to reach before the expiry.

TCP also employs clever mechansims to make sure that segments are not too small.

A sample scenario



Clever strategies

The sender will send a window probe segment just to make sure that the sender will inform the new window size.

The receiver may delay ACK so as to ensure that the sender has enough data to send.

- Done in case where sender transmitts small segments.
- Known as Nagles algorithm

Another issue addresses is silly window problem

The receiver always reply with a vey small window size.

Clarke suggested a solution that the receiver should wait until a certain space is available at the receiver

TCP Congestion Management

We have seen that TCP has an inherent window size field.

The sender will transmitt data only based on the window size reported by the receiver.

This window may be represented as rwind.

The loss of any data segment prompts the sender to guess that data is lost because of congestion.

The sender will have to adjust the speed of transmission .

Sender will maintain another window size known as Congestion window -rwind.

The data transmitted in min(rwind,cwind)

Congestion control mechansims work by adjusting the growth of cwind.

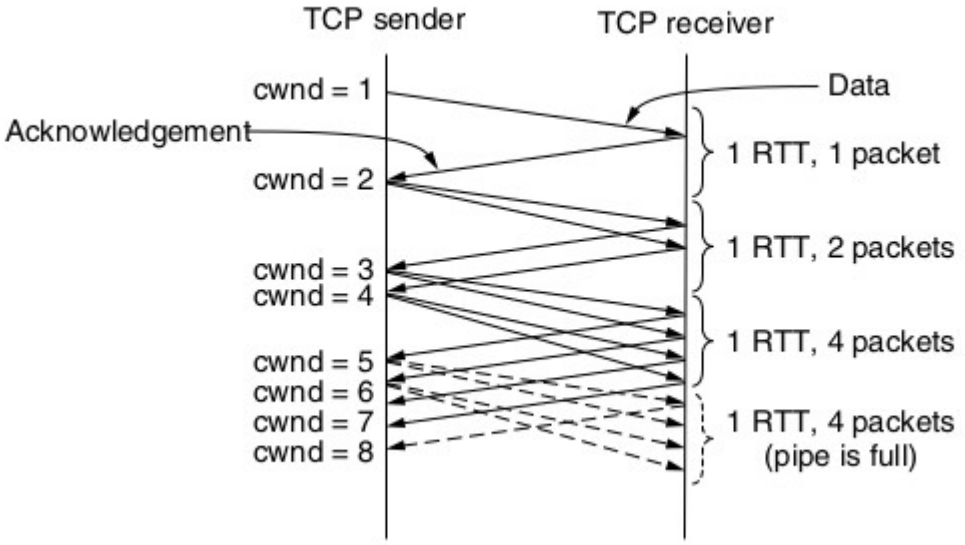
There are mechanisms specified on how the cwind is updated.

The cwind value is increased if there are no congestion.

When congestion is identified,cwind value is decremented.

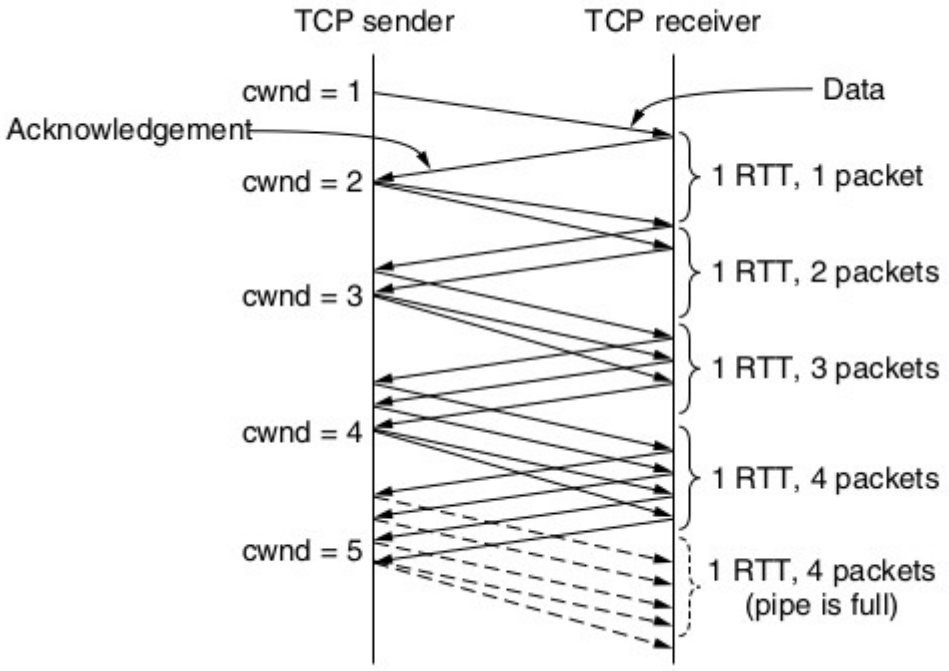
Slow start mechanism

A rather deceiving name for a mechanism which exponentially increases the size of cwind.



Additive Increase

Justifying the name the cwind is updated by increments of one segment size.



TCP Tahoe

A threshold is maintained for the growth of cwind.

Upto threshold cwind grows by using slow start.

After treshold growth is in additive increase

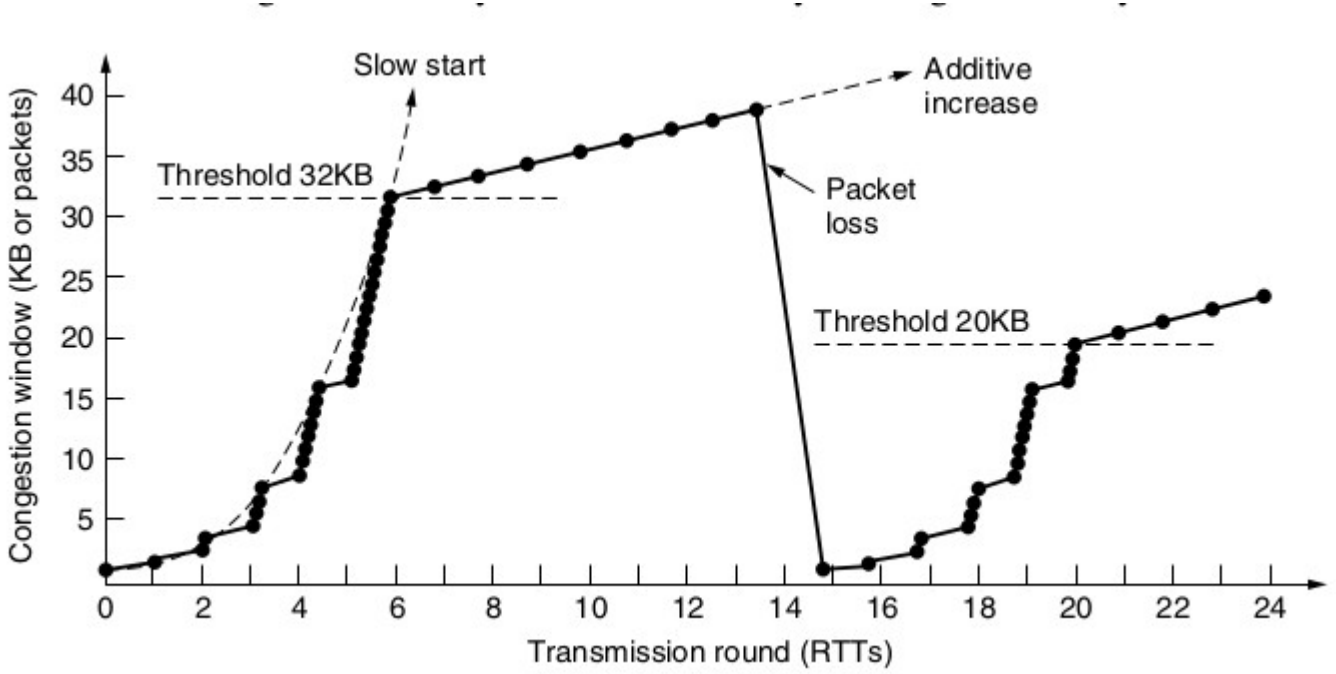
If congestion happens in between

Treshold is reduced to half.

Cwind reset to the beginning

TCP Tahoe uses a method known as fast retransmit.

- Avoids waiting till timer expiry to retransmit.
- When transmitting more segments if one of the segment is lost ,the segments reach the destination not in order.
- The receiver will send ACK for the last segment which was received in order.
- This is infact a duplicate ACK.
- So before timer expiry the sender will retransmit the data.


Slow start
Additive increase
Threshold 32KB
Packet loss
Threshold 20KB
Congestion window (KB or packets)
40
35
30
25
20
15
10
5
0
2
4
6
8
10
12
14
16
18
20
22
24
Transmission round (RTTs)

TCP Reno

A threshold is maintained for the growth of cwind.

Upto threshold cwind grows by using slow start.

After treshold growth is in additive increase

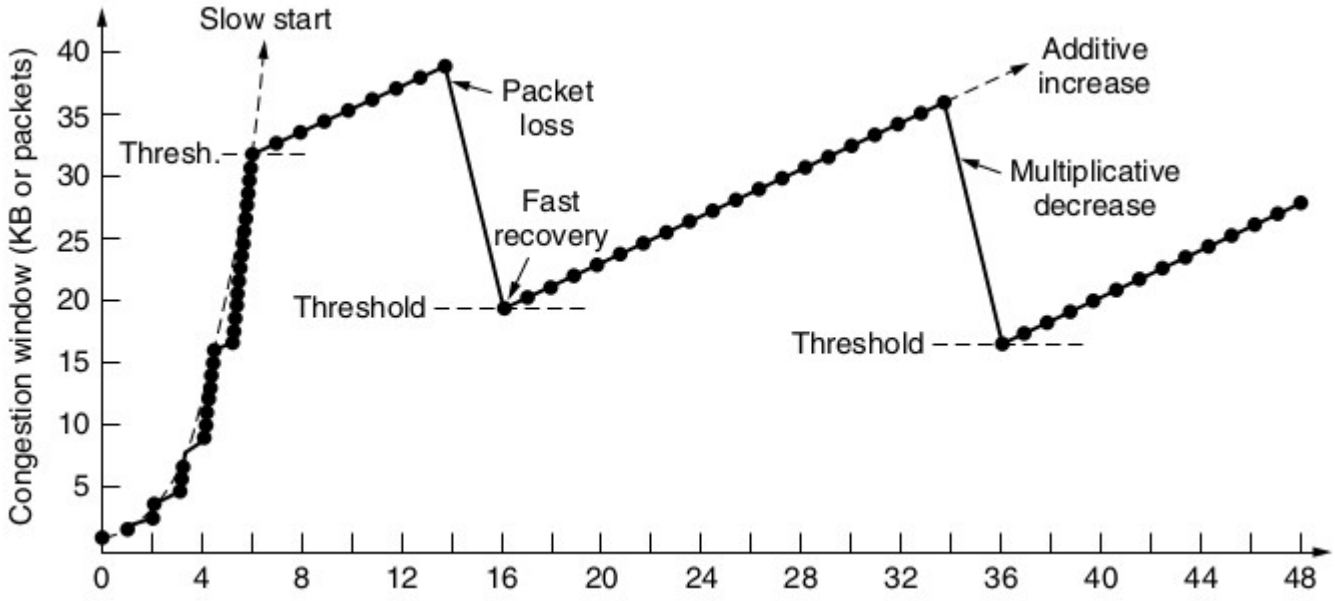
If congestion happens in between

Treshold is reduced to half.

Cwind reset to threshold.

TCP Reno uses a method known as fast recovery along with fast retransmission

- Avoids waiting till timer expiry to retransmit.
- When transmitting more segments if one of the segment is lost ,the segments reach the destination not in order.
- The receiver will send ACK for the last segment which was received in order.
- Based on these duplicate ACK it is understood that some other segment has already left the sender.



TCP Reno showing the saw tooth behaviour.