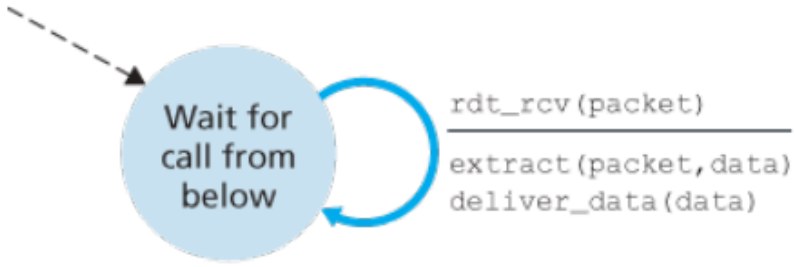
reliable data transfer

RDT 1.0

a. rdt1.0: sending side



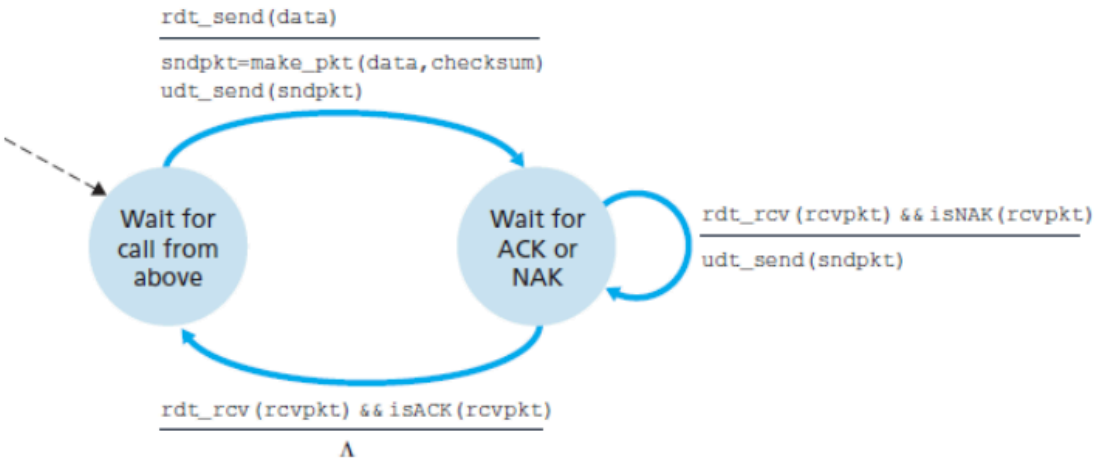
b. rdt1.0: receiving side

Figure 3.9 *rdt1.0* – A protocol for a completely reliable channel

- sending side
  - `rdt_send(data)` event : accepts data from the upper layer via the rdt_send(data) event,
  - `make_pkt(data)` action : creates a packet containing data
  - `udt_send(packet)` action : sends packet into the channel
- receiving side
  - `rdt_rcv(packet)` event : On the receiving side, rdt receives a packet from the underlying channel
  - `extract(packet,data)` action : removes/extracts data from the packet
  - `deliver data` action : passes the data to upper layer
- the only consideration taken here is that the channel is completely unreliable

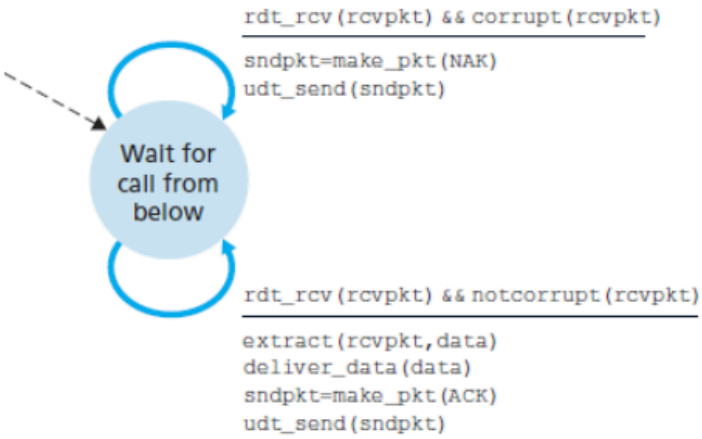
RDT 2.0

- consideration : channel with bit errors (all packets are received in the order they are sent but packets may be corrupted), uses checksum to mitigate errors along with `ACK` and `NACK`
- use of control messages like `ACK` and `NACK`
  - RDT protocols based on retransmission with use of `ACK` and `NACK` are `ARQ(Automatic Repeat reQuest)` protocols
  - 3 protocol capabilities are required in ARQ protocols to handle the presence of bit errors:
    - `ERROR DETECTION` : using checksum ; extra bits are sent(data+checksum)
    - `RECIEVER FEEDBACK` : 1 bit `ACKS` (1) and `NACKS` (0)
    - `RETRANSMISSION` : packet that is received in error at the receiver will be re transmitted by the sender



a. rdt2.0: sending side

Figure 3.10 *rdt2.0* – A protocol for a channel with bit errors



b. rdt2.0: receiving side

-
- flaws
  - not accounted for the possiblity that `ACK` and `NACK` may be corrupted(can be solved by checksum fr `ACK` in rdt 2.1)
  - possibility of duplicates when retransmitted so receiver doesn't know whether the ACK or NAK it last sent was received correctly at the sender.(can be solved by adding sequence number ; implemented in rdt 2.1)

RDT 2.1

- considerations :
  - corrupted `ACK` and `NACK` : mitigated by use of checksum for `ACK` and `NACK`
  - duplicates due to retransmission : mitigated by use of sequence numbers
  - here 1 bit sequence nuber is used, so twice as many states as rdt 2.0

- note sender sends duplicate pckts(pckst with same seq no) if it didnt recieve `ACK` or if `ACK` has errors, so reciever knows `ACK` isnt recieved correctly and accordinly retransmits. seq number is used by sender to let the reciever know which `ACK` it has recieved correctly or not, this way reciever can know if packet received is retransmission or not

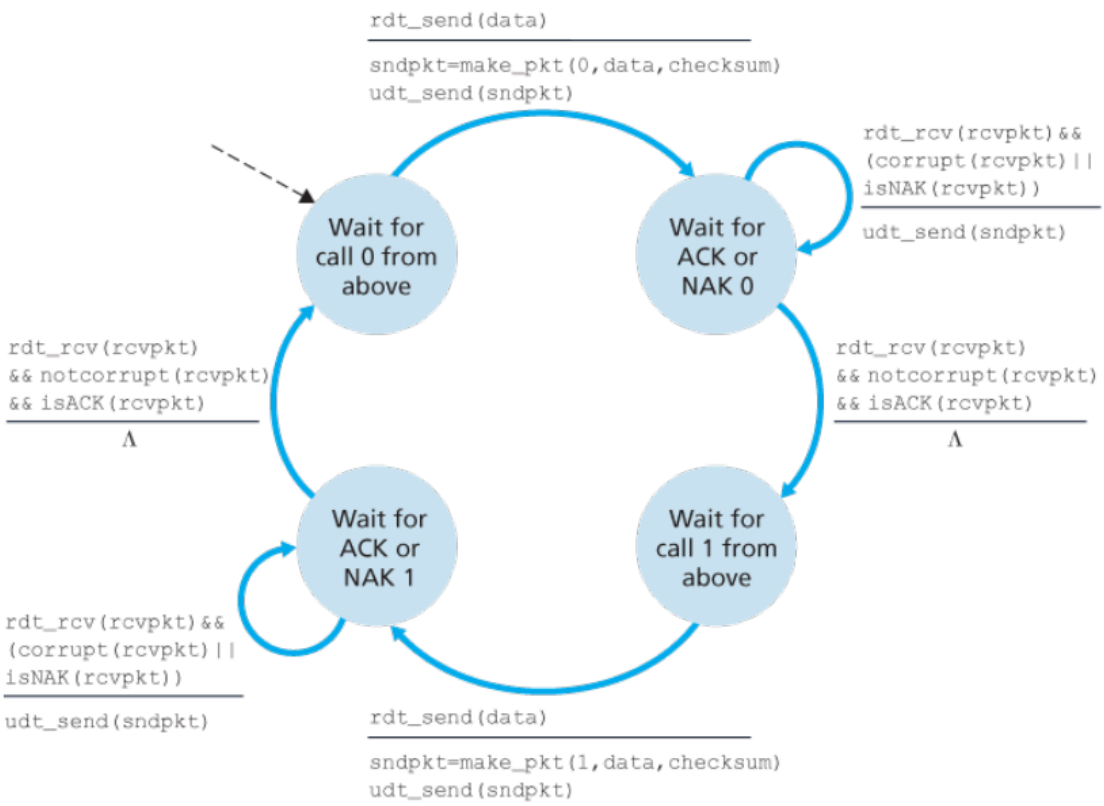


Figure 3.11 ***rdt2.1*** **sender**

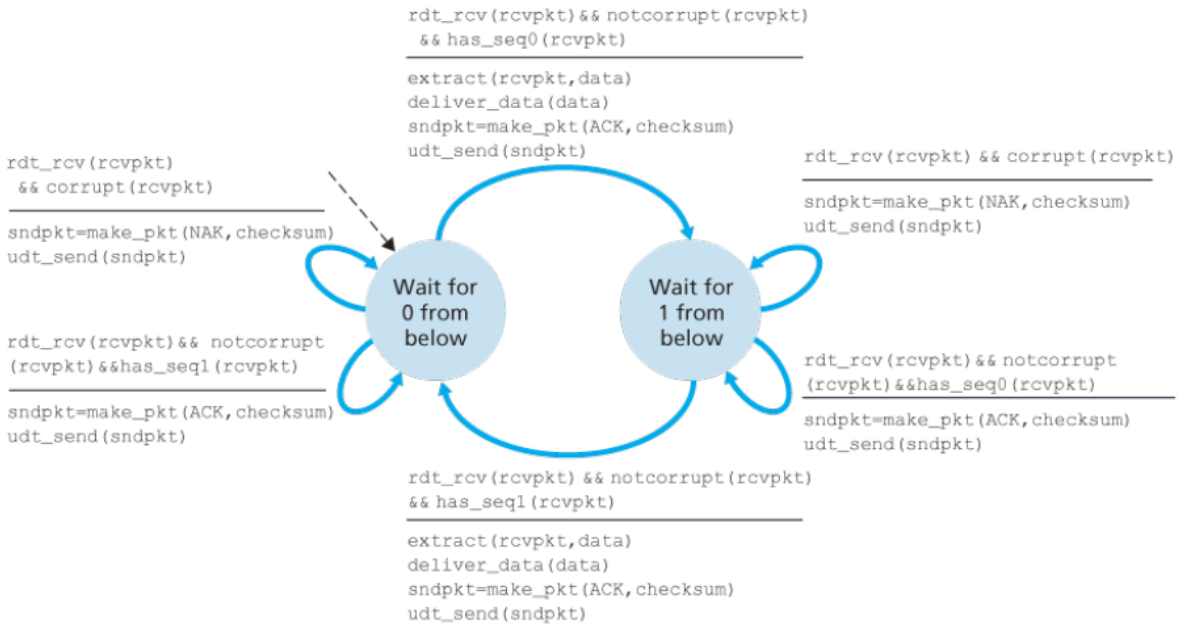


Figure 3.12 *rdt2.1* **receiver**

RDT 2.2

- considerations
  - sequence number for `ACK` sent by reciever as well(rdt 2.1 had seq numm only for packets it sends)
  - NAK free protocol : instead of sending a NAK, we send an ACK for the last correctly received packet.A sender that receives two/duplicate ACKs for the same packet knows that the receiver did not correctly receive the packet following the packet that is being ACKed twice.
  - reciever includes sequence number of packet being ACKed in the ACK message it sends

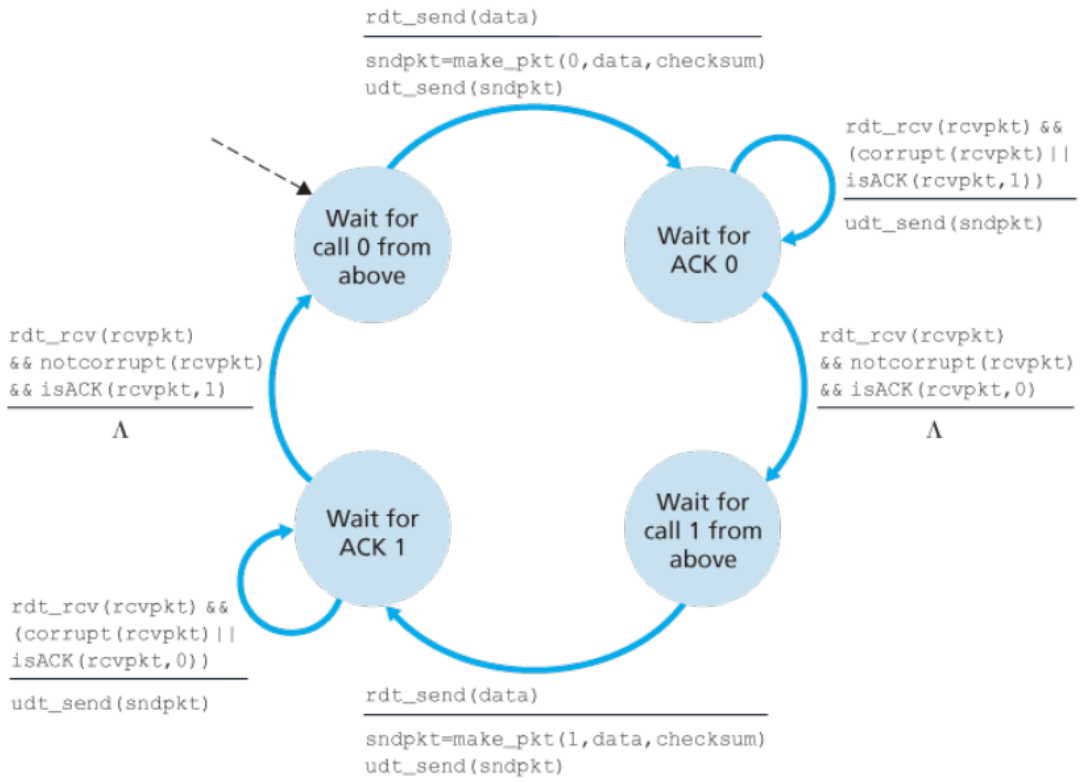


Figure 3.13 *rdt2.2* **sender**

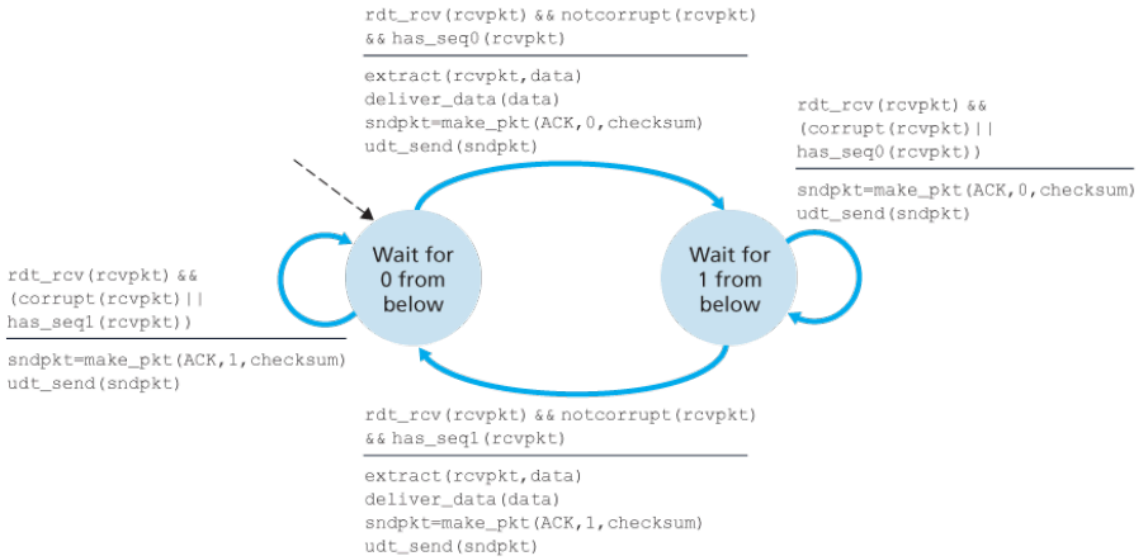
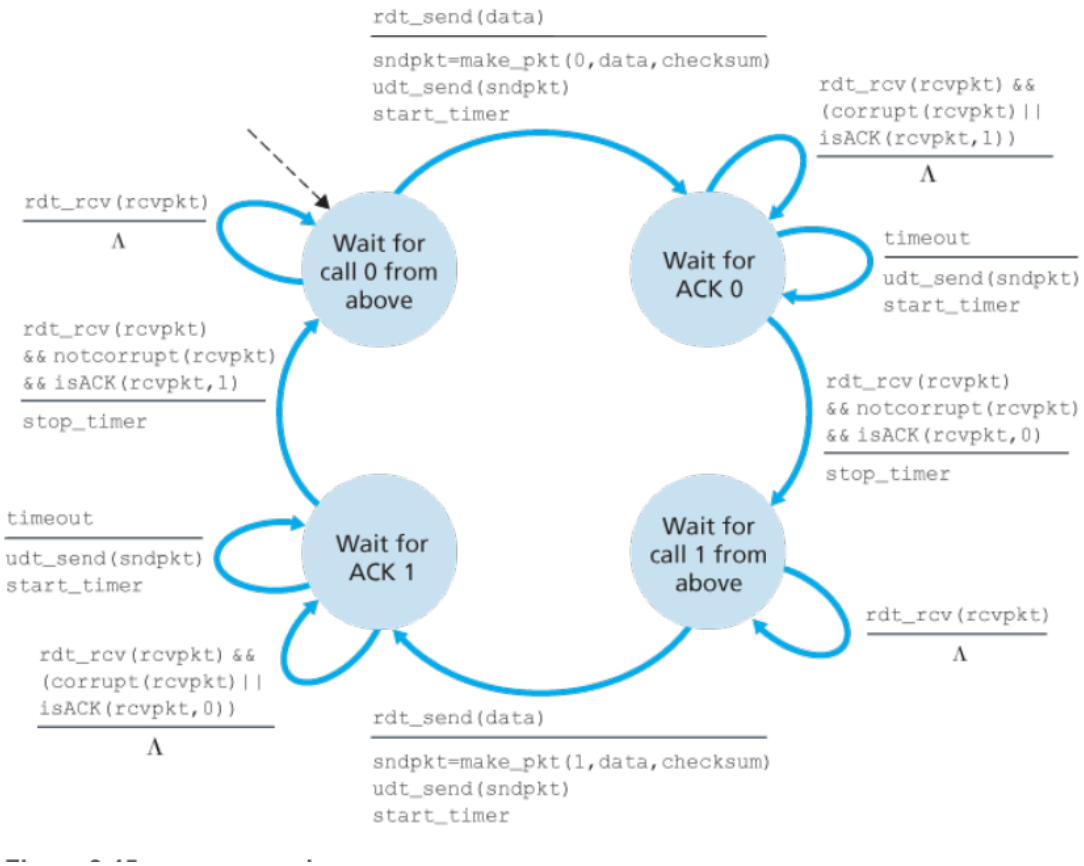


Figure 3.14 *rdt2.2* **receiver**

RDT 3.0

- consideration :
  - underlying channel can lose packets as well, mitigated by checksumming, sequence numbers, ACK packets, and retransmissions

    - use of timers , if packet has not been recieved during this time period , retransmit the packet
    - note that duplicate packets means the packet is likely lost so retransmit(ex reciever recieves packet 0 twice instead of getting packet 0 packet 1 so packet 1 likely to be lost)
- as packet sequence numbers alternate between 0 and 1, protocol rdt3.0 is sometimes known as the alternating-bit protocol .
- it is stop and wait protocol- sender utilisation is less as senders sends the other packets only after it receives ACK for previously sent packet

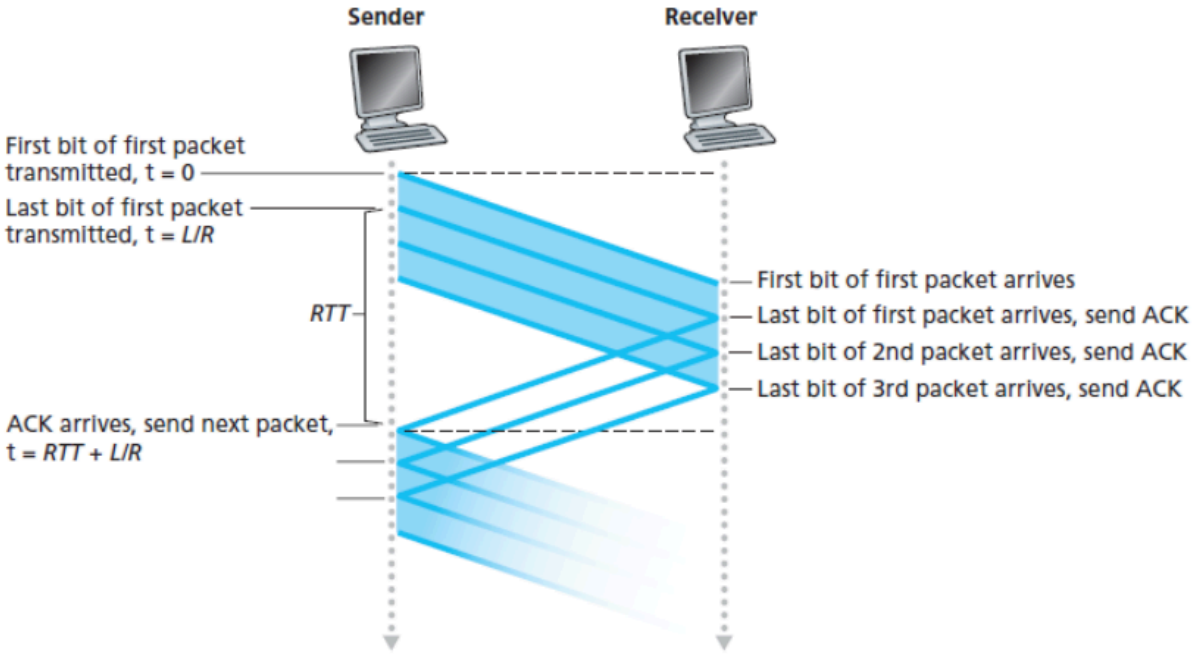


PIPELINED PROTOCOLS

- sender is allowed to send multiple packets without waiting for acknowledgment
- sender utilisation =

$$U_{\text{sender}} = \frac{\frac{nL}{R}}{RTT + \frac{L}{R}}$$

- where n is number of packets sent at once (pipelined)



- Two basic approaches toward pipelined error recovery can be identified: Go-Back-N and selective repeat.

Go-Back-N (GBN)

- sender is allowed to transmit multiple packets (when available) without waiting for an acknowledgment, but is constrained to have no more than some aximum allowable number, N, of unacknowledged packets in the pipeline

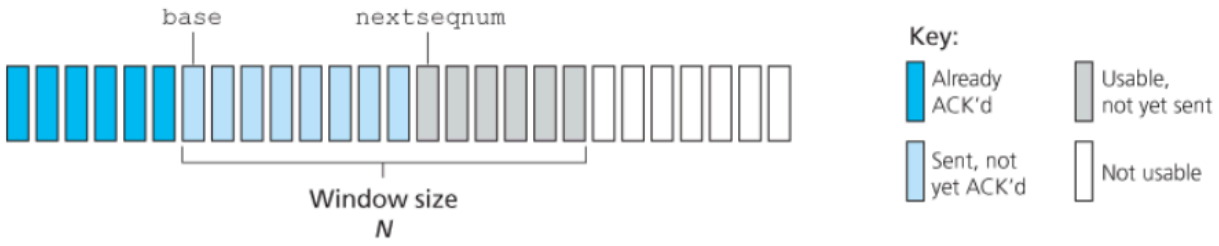


Figure 3.19 Sender's view of sequence numbers in Go-Back-N

- Usable Sequence numbers are those that can be used for packets that can be sent immediately, should data arrive from the upper layer.
- sequence numbers greater than or equal to base+N cannot be used until an unacknowledged packet currently in the pipeline has been acknowledged.
- the range of permissible sequence numbers for transmitted but not yet acknowledged packets can be viewed as a window of size N over the range of sequence numbers. As the protocol operates, this window slides forward over the sequence number

space. For this reason, N is often referred to as the window size and the GBN protocol itself as a sliding-window protocol.

- the benefit of limiting number of outstanding, unacknowledged packets to N is to enable to efficient flow control
- if k is number of bits in sequence number , the range of sequence numbers is 0,2^k-1 and follows modulo 2^k arithmetic

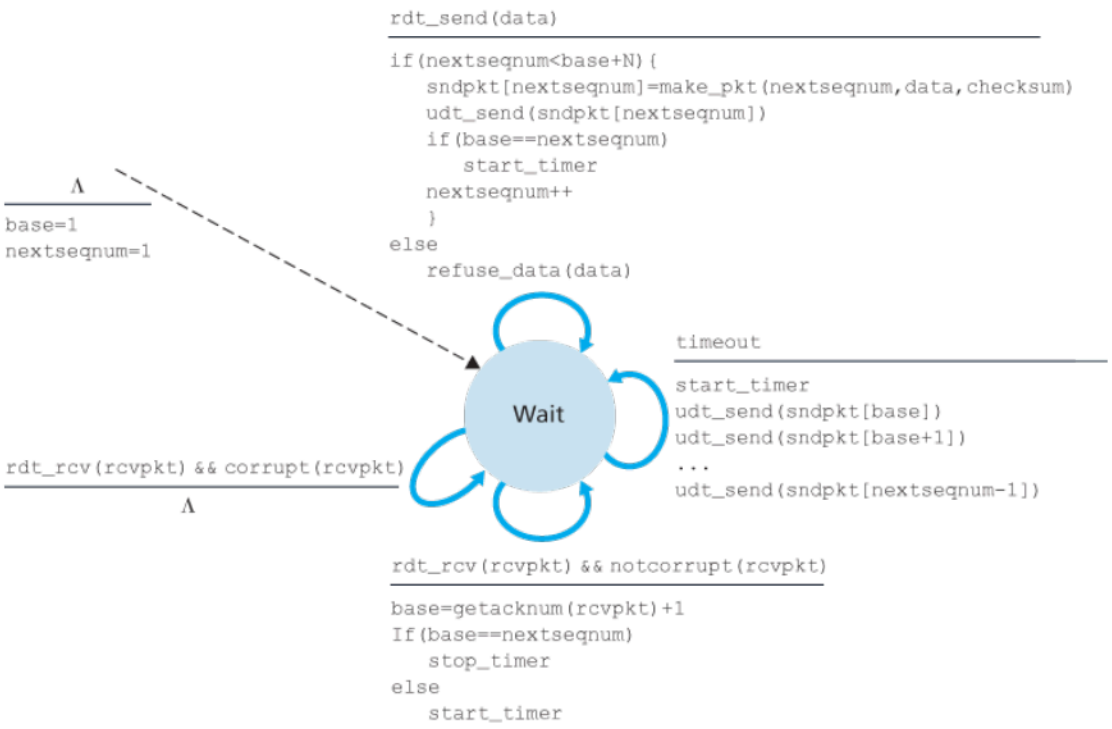


Figure 3.20 Extended FSM description of the GBN sender

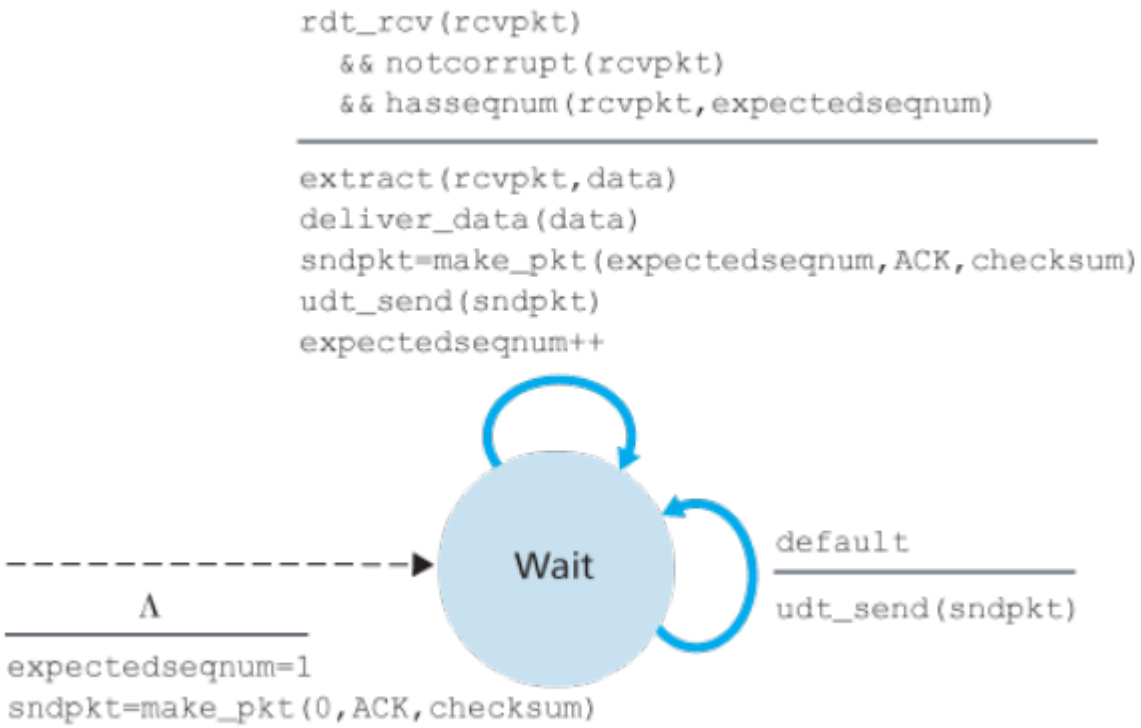


Figure 3.21 Extended FSM description of the GBN receiver

- 3 types of events at sender side
  - invocation from above : sender checks if window is full, if it is full it returns packet back to upper layer , or it will be buffered
  - receipt of ACK : acknowledgment for a packet with sequence number n will be taken to be a cumulative acknowledgment, indicating that all packets with a sequence number up to and including n have been correctly received
  - timeout :f a timeout occurs, the sender resends all packets that have been previously sent but that have not yet been acknowledged.If an ACK is received but there are still additional transmitted but not yet acknowledged packets, the timer is restarted. If there are no outstanding, unacknowledged packets, the timer is stopped.
- reciever side :
- if packet with seq. no. n is received correctly and is in order receiver sends an ACK for packet n and delivers the data portion of the packet to the upper layer. In all other cases, the receiver discards the packet and resends an ACK for the most recently received in-order packet.if packet k has been received and delivered, then all packets with a sequence number lower than k have also been delivered.
- In GBN protocol, the receiver discards out-of-order packets. for ex if packet n is lost, both it and packet n+1 will eventually be retransmitted as a result of the GBN retransmission rule at the sender. Thus, the receiver can simply discard packet n+1.The only piece of

information the receiver need maintain is the sequence number of the next in-order packet(`expectedseqnum`)

Selective repeat

- in `GBN` single packet error can cause GBN to unnecessarily retransmit a lot of packets , resulting in reduced performance
- 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
- SR receiver will acknowledge a correctly received packet whether or not it is in order. Out-of-order packets are buffered until any missing packets (that is, packets with lower sequence numbers) are received, at which point a batch of packets can be delivered in order to the upper layer.
- note the use of selective acks iunlike the cumulative acks seen in GBN
- window size must be less than or equal to half the size of the sequence number space for SR protocols.

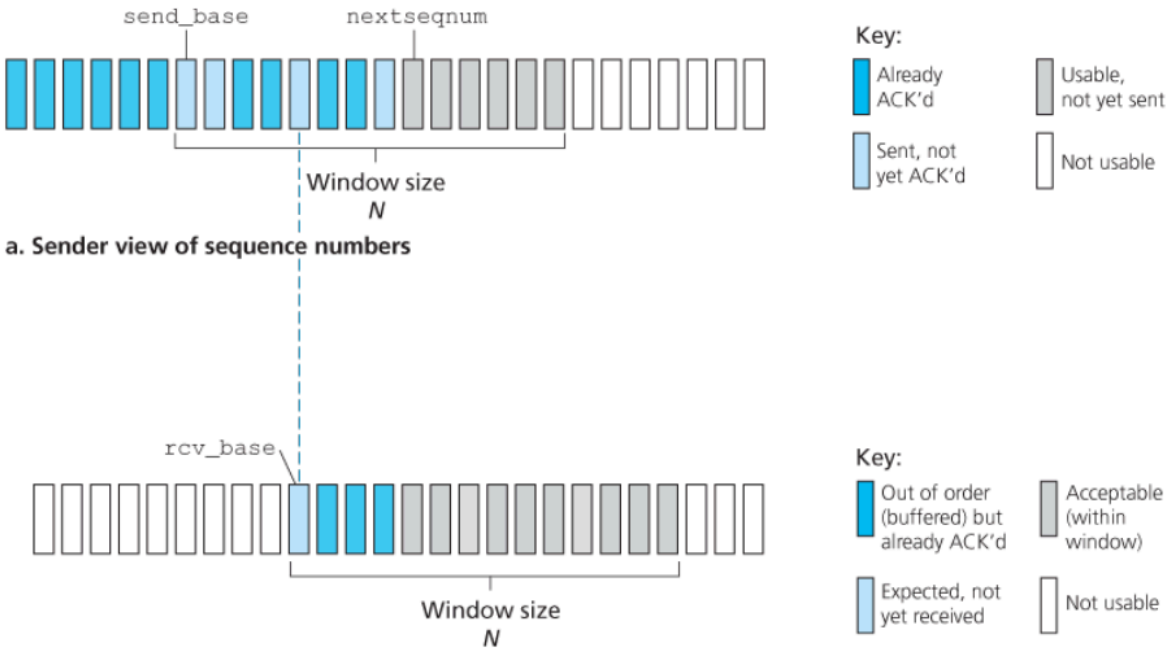


Figure 3.23 Selective-repeat (SR) sender and receiver views of sequence-number space

1. *Data received from above.* When data is received from above, the SR sender checks the next available sequence number for the packet. If the sequence number is within the sender's window, the data is packetized and sent; otherwise it is either buffered or returned to the upper layer for later transmission, as in GBN.
2. *Timeout.* Timers are again used to protect against lost packets. However, each packet must now have its own logical timer, since only a single packet will be transmitted on timeout. A single hardware timer can be used to mimic the operation of multiple logical timers [Varghese 1997].
3. *ACK received.* If an ACK is received, the SR sender marks that packet as having been received, provided it is in the window. If the packet's sequence number is equal to `send_base`, the window base is moved forward to the unacknowledged packet with the smallest sequence number. If the window moves and there are untransmitted packets with sequence numbers that now fall within the window, these packets are transmitted.

Figure 3.24 SR sender events and actions

1. *Packet with sequence number in* `[rcv_base, rcv_base+N-1]` *is correctly received.* In this case, the received packet falls within the receiver's window and a selective ACK packet is returned to the sender. If the packet was not previously received, it is buffered. If this packet has a sequence number equal to the base of the receive window (`rcv_base` in Figure 3.22), then this packet, and any previously buffered and consecutively numbered (beginning with `rcv_base`) packets are delivered to the upper layer. The receive window is then moved forward by the number of packets delivered to the upper layer. As an example, consider Figure 3.26. When a packet with a sequence number of `rcv_base=2` is received, it and packets 3, 4, and 5 can be delivered to the upper layer.
2. *Packet with sequence number in* `[rcv_base-N, rcv_base-1]` *is correctly received.* In this case, an ACK must be generated, even though this is a packet that the receiver has previously acknowledged.
3. *Otherwise.* Ignore the packet.

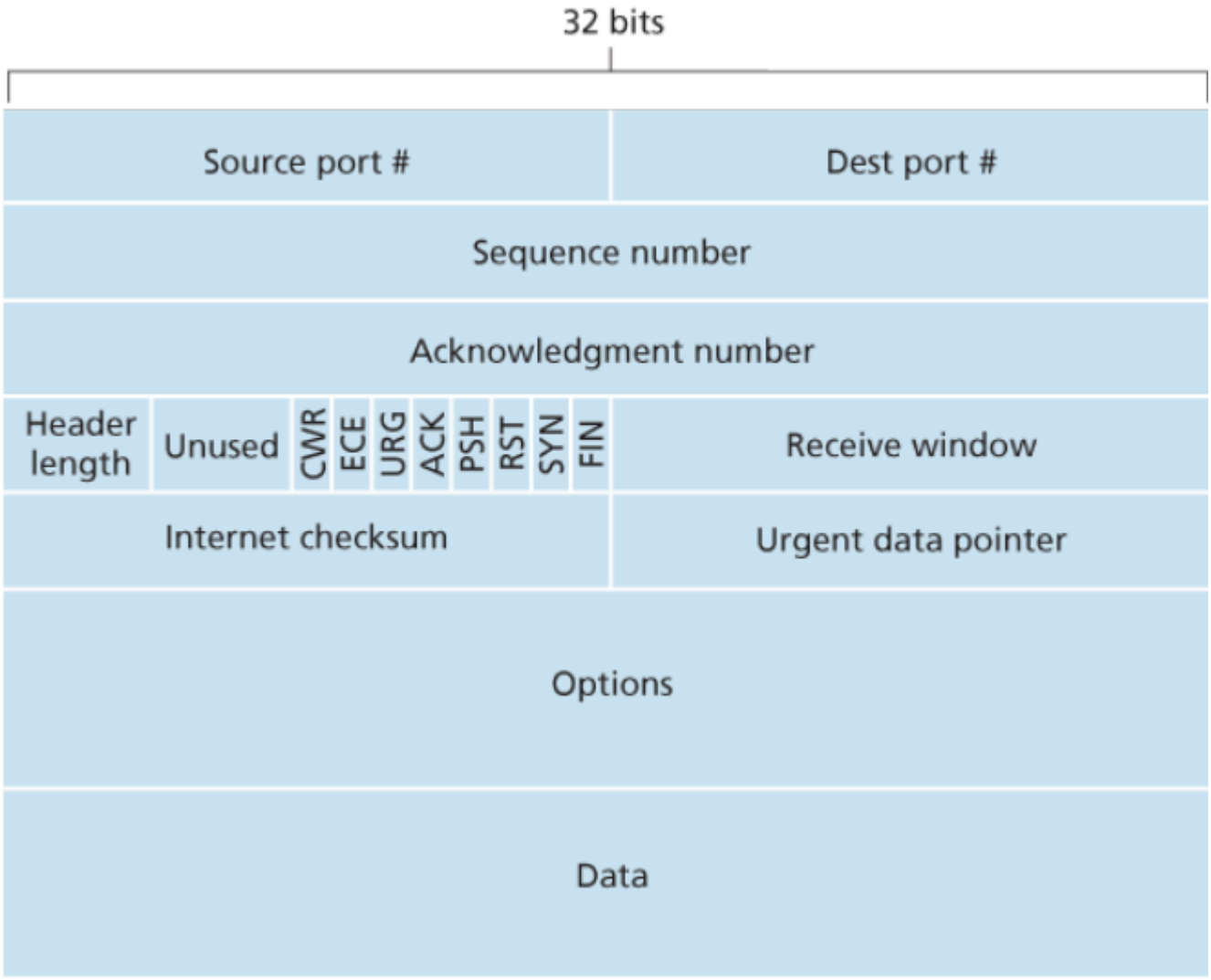
Figure 3.25 SR receiver events and actions

TCP

- connection oriented,3-way handshake mechanism
- `MSS` - Maximum segment size : max amount of data in segment (without TCP Header)

- `MTU` - largest link-layer frame that can be sent by the local sending host(typically 1500 bytes for ethernet)
- `MSS` = `MTU` - TCP header size(typically 40 bytes)

TCP SEGMENT STRUCTURE



- options field : set when a sender and receiver negotiate MSS or as a window scaling factor for use in high-speed networks. A time-stamping option is also defined.if options field is empty then header size is 20 bytes
- 4 bit header length
- 6 bit flag field :
  - ACK indicates ACK number(he acknowledgment number that Host A puts in its segment is the sequence number of the next byte Host A is expecting from Host B).TCP provides cumulative ACKs
  - RST,SYN AND FIN are used for tcp connection management

- CWR and ECE are used in explicit congestion notification
- 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 last byte of this urgent data is indicated by the 16-bit urgent data pointer field

ESTIMATING ROUND TRIP TIME

- The sample RTT, denoted SampleRTT , for a segment is the amount of time between when the segment is sent (that is, passed to IP) and when an acknowledgment for the segment is received. Instead of measuring a SampleRTT for every transmitted segment, most TCP implementations take only one SampleRTT measurement at a time. That is, at any point in time, the SampleRTT is being estimated for only one of the transmitted but currently unacknowledged segments, leading to a new value of SampleRTT approximately once every RTT. Also, TCP never computes a SampleRTT for a segment that has been retransmitted;
- due to fluctuation of SampleRTT values , TCP maintains an weighted average, called EstimatedRTT , of theSampleRTT values

$$EstimatedRTT = (1 - \alpha) \cdot EstimatedRTT + \alpha \cdot SampleRTT$$

- recommended value of α is 0.125
- as this weighted average puts more weight on recent samples than on old samples it is called exponential weighted moving average (EWMA).
- in addition to having an estimate of the RTT, it is also valuable to have a measure of the variability of the RTT. [RFC 6298] defines the RTT variation, DevRTT , as an estimate of how much SampleRTT typically deviates from EstimatedRTT :

$$DevRTT = (1 - \beta) \cdot DevRTT + \beta \cdot |SampleRTT - EstimatedRTT|$$

- The recommended value of β is 0.25.

$$TimeoutInterval = EstimatedRTT + 4 \cdot DevRTT$$

- intial timeout of 1s is used.when timeout occurs , timeout interval is doubled , however when as soon as a segment is received and EstimatedRTT is updated, the TimeoutInterval is again computed using the formula above.

rdt in tcp

- individual timers with each pakcet result in huge timer overhead, so TCP uses only single transmission timer
- tcp uses duplicates acks in addition to single timer
- whenever timeout occurs tcp sets timeout to twice previous value
- tcp fast retransmit : as timeout increases exponentially , end to end delay might increase.Fortunately, the sender can often detect packet loss well before the timeout event occurs by noting so-called duplicate ACKs. when an ACK arrives that reacknowledges a segment for which the sender has already received an earlier acknowledgment, as cumulativ acks are used, it indicates that segments with seq number after ACK hasnt been recieved correctly ,so tcp immediately retransmits without waiting for timeout
- thus f the TCP sender receives three duplicate ACKs for the same data, it takes this as an indication that the segment following the segment that has been ACKed three times has been lost , so TCP sender performs fast retransmit , so segment is retransmitted before timeout occurs
- TCP's error recovery is considered a hybrid of Selective repead and go back N as , like go back n tcp uses cumulativ acks, but unlike go back n , if packet n is lost tcp only retransmits that packet and not the other packets in the window.it also buffers the out of order packets

```
/* Assume sender is not constrained by TCP flow or congestion control, that data from above is less
than MSS in size, and that data transfer is in one direction only. */

NextSeqNum=InitialSeqNumber
SendBase=InitialSeqNumber

loop (forever) {
    switch(event)

        event: data received from application above
            create TCP segment with sequence number NextSeqNum
            if (timer currently not running)
                start timer
            pass segment to IP
            NextSeqNum=NextSeqNum+length(data)
            break;

        event: timer timeout
            retransmit not-yet-acknowledged segment with
                smallest sequence number
            start timer
            break;

        event: ACK received, with ACK field value of y
            if (y > SendBase) {
                SendBase=y
                if (there are currently any not-yet-acknowledged segments)
                    start timer
                }
            break;

    } /* end of loop forever */
```

Figure 3.33 Simplified TCP sender

FLOW CONTROL

- TCP provides a flow-control service to its applications to eliminate the possibility of the sender overflowing the receiver's buffer. Flow control is thus a speed-matching service—matching the rate at which the sender is sending against the rate at which the receiving application is reading
- TCP provides flow control by having the sender maintain a variable called the receive window .Informally, the receive window is used to give the sender an idea of how much free buffer space is available at the receiver. Because TCP is full-duplex, the sender at each side of the connection maintains a distinct receive window.

$$LastByteRcvd - LastByteRead \leq RcvBuffer$$

- The receive window, denoted rwnd is set to the amount of spare room in the buffer:

$$rwnd = RcvBuffer - [LastByteRcvd - LastByteRead]$$

- Host B tells Host A how much spare room it has in the connection buffer by placing its current value of rwnd in the receive window field of every segment it sends to A. Initially, Host B sets rwnd = RcvBuffer .
- host A keeps track of LastByteSent and LastByteAcked.LastByteSent – LastByteAcked , is the amount of unacknowledged data that A has sent into the connection,By keeping the amount of unacknowledged data less than the value of rwnd , Host A is assured that it is not overflowing the receive buffer at Host B

$$LastByteSent - LastByteAcked \leq rwnd$$

TCP CONNECTION MANAGEMENT

- 3 way tcp handshake
  - SYN : client sends a special TCP segment with no application layer data but SYN bit of flag field set to 1.client randomly chooses an initial sequence number (client_isn) and puts this number in the sequence number field of the initial TCP SYN segment
  - SYN-ACK : server extracts the TCP SYN segment from the datagram, allocates the TCP buffers and variables to the connection, and sends a connection-granted segment to the client TCP.This segment has SYN and ACK bit set to 1, ACK field set to client_isn +1 and server chooses its own initial sequence number (server_isn)and puts in seq no. field
  - ACK : Upon receiving the SYNACK segment, the client also allocates buffers and variables to the connection.client sends server another segment which ACKs SYNACK.in this segment SYN=0 , and this segment may carry application layer data in payload

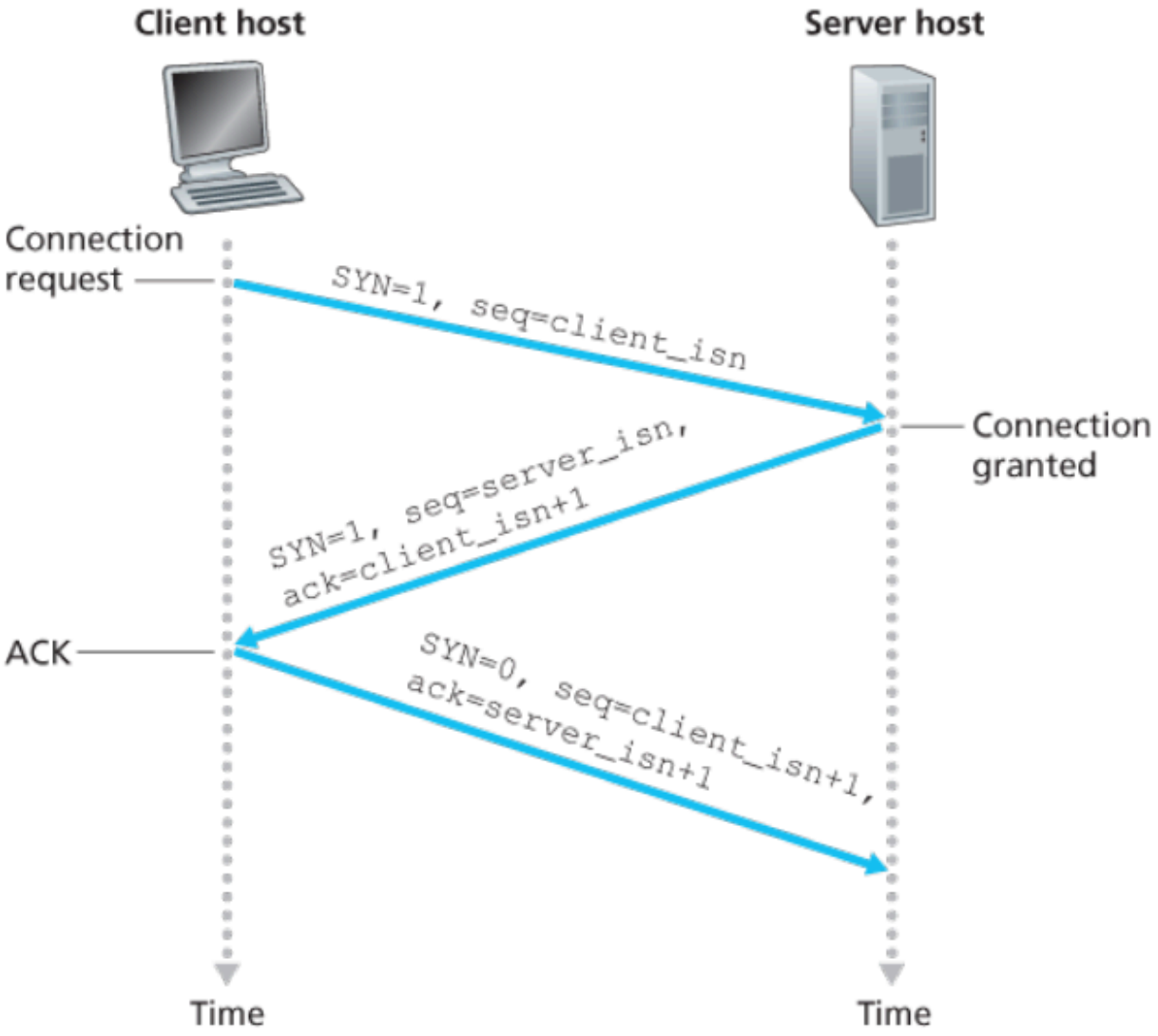
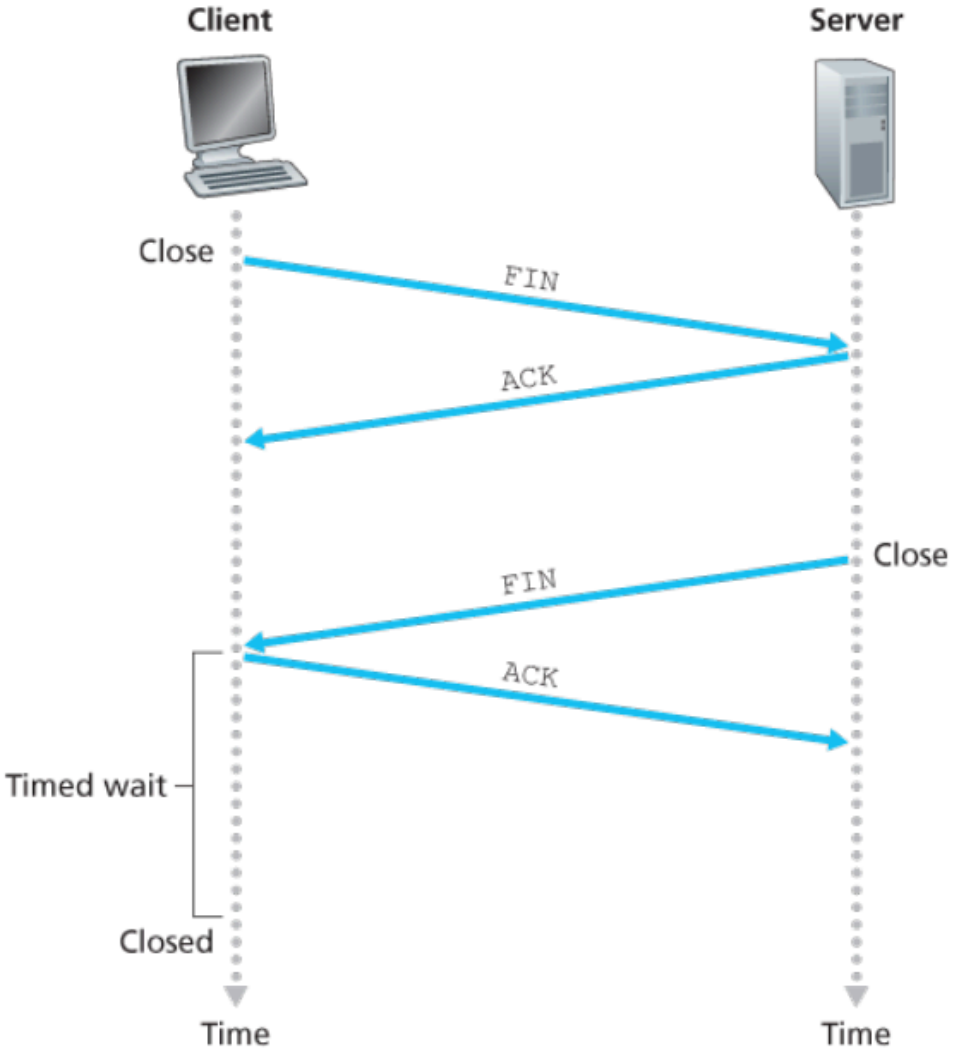


Figure 3.39 TCP three-way handshake: segment exchange



- if client wants to close the connection : the client TCP to send a special TCP segment to the server process with FIN =1.. When the server receives this segment, it sends the client an acknowledgment segment in return. The server then sends its own shutdown segment, which has the FIN bit set to 1. Finally, the client acknowledges the server's shutdown segment. At this point, all the resources in the two hosts are now deallocated.
- nmap : To explore a specific TCP port, say port 6789, on a target host, nmap will send a TCP SYN segment with destination port 6789 to that host.three possible outcomes :
  - source host recieves TCP SYNACK segment from the target host.,so nmap returns open
  - The source host receives a TCP RST segment from the target host.This means that the SYN segment reached the target host, but the target host is not running an application with TCP port 6789. But the attacker at least knows that the segments destined to the host at port 6789 are not blocked by any firewall on the path between source and target hosts.
  - The source receives nothing. This likely means that the SYN segment was blocked by an intervening firewall and never reached the target host.

- TCP SYN FLOOD ATTACK
  - In this attack, the attacker(s) send a large number of TCP SYN segments, without completing the third handshake step. With this deluge of SYN segments, the server's connection resources become exhausted as they are Allocated (but never used!) for half-open connections; legitimate clients are then denied service
  - an effective defense known as SYN cookies [RFC 4987] are used .they work as follows :
  - instead of creating a half-open TCP connection for this SYN, the server creates an initial TCP sequence number that is hash function of source and destination IP addresses and port numbers of the SYN segment, as well as a secret number only known to the server. This is called "cookie.
  - if server recieves an ACK , it checks if the value of ACK field in clients SYN ACK is = orginally used hash_fn(source ip,destination ip,port nos in SYN ACK field,cookie) + 1.If this is relation is true server creates fully open connection , otherwise no harm done as server hasnt allocated any resources

TCP CONGESTION CONTROL

tcp congestion control

- slow start , congestion avoidance and fast recovery
- MSS is without TCP header
- slow start vs congestion avoidance : cwnd is increased by 1 MSS every time transmitted segment acknowledged but in CA it is increased by 1 MSS every rtt (1 rtt has `cwnd` no of packets)

slow start :

- Thus, in the slow-start state, the value of cwnd begins at 1
  MSS and increases by 1 MSS every time a transmitted segment is first acknowledged
- This process results in a doubling of the sending rate every RTT as in 1 rtt cwnd packets r sent . Thus, the TCP send rate starts slow but grows exponentially during the slow start phase.
- First, if there is a loss event (i.e., congestion) indicated by a timeout, the TCP sender sets the value of cwnd to 1 and begins the slow start process anew
- ssthresh = cwnd/2 —half of the value of the congestion window value when congestion was detected

- when the value of cwnd equals ssthresh , slow start ends and TCP transitions into congestion avoidance mode
- The final way in which slow start can end is if three duplicate ACKs are detected, in which case TCP performs a fast retransmit

congestion avoidance :

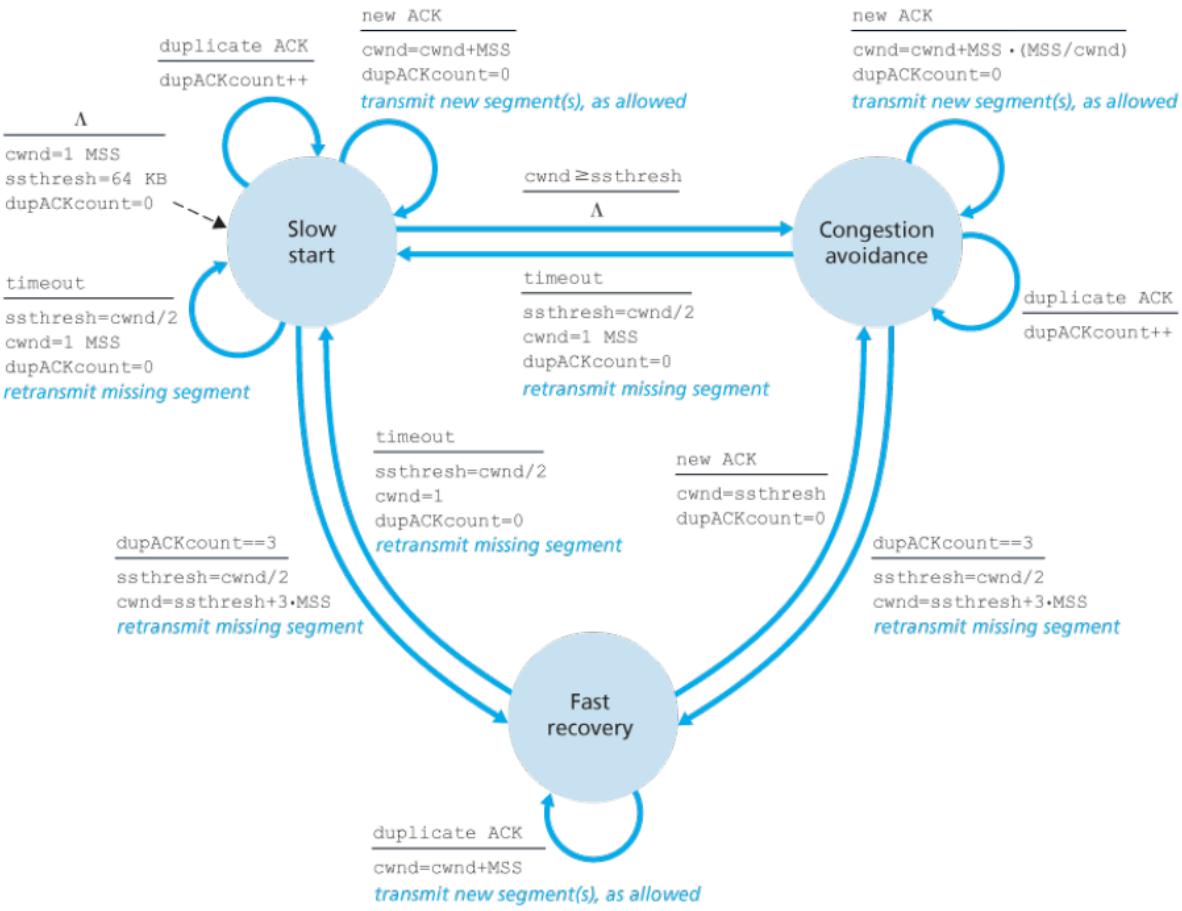
- rather than doubling the value of cwnd every RTT, TCP adopts a more conservative approach and increases the value of cwnd by just a single MSS every RTT
- ex : A common approach is for the TCP sender to increase cwnd by MSS bytes (MSS/ cwnd) whenever anew acknowledgment arrives.
    - For example, if MSS is 1,460 bytes and cwnd is 14,600 bytes, then 10 segments are being sent within an RTT.
    - Each arriving ACK (assuming one ACK per segment) increases the congestion window size by 1/10 MSS, and thus, the value of the congestion window will have increased by one MSS after ACKs when all 10 segments have been received.
- TCP’s congestion - avoidance algorithm behaves the same when a timeout occurs. As in the case of slow start: The value of cwnd is set to 1 MSS, and the value of ssthresh is updated to half the value of cwnd when the loss event occurred. Recall, however, that a loss event also can be triggered by a triple duplicate ACK event.
- when it encounters triple duplicate , TCP halves the value of cwnd (adding in 3 MSS for good measure to account for the triple duplicate ACKs received) and records the value of ssthresh to be half the value of cwnd when the triple duplicate ACKs were received. The fast-recovery state is then entered.so
- if timeout occurs it tcp reverts to slow start (by setting `cwnd` to 1 and setting new value for ssthresh), nd resumes congestion avoidance when cwnd > ssthresh
- ssthresh = cwnd / 2 , and cwnd = ssthresh + 3MSS (to retrasnmit missing segment)

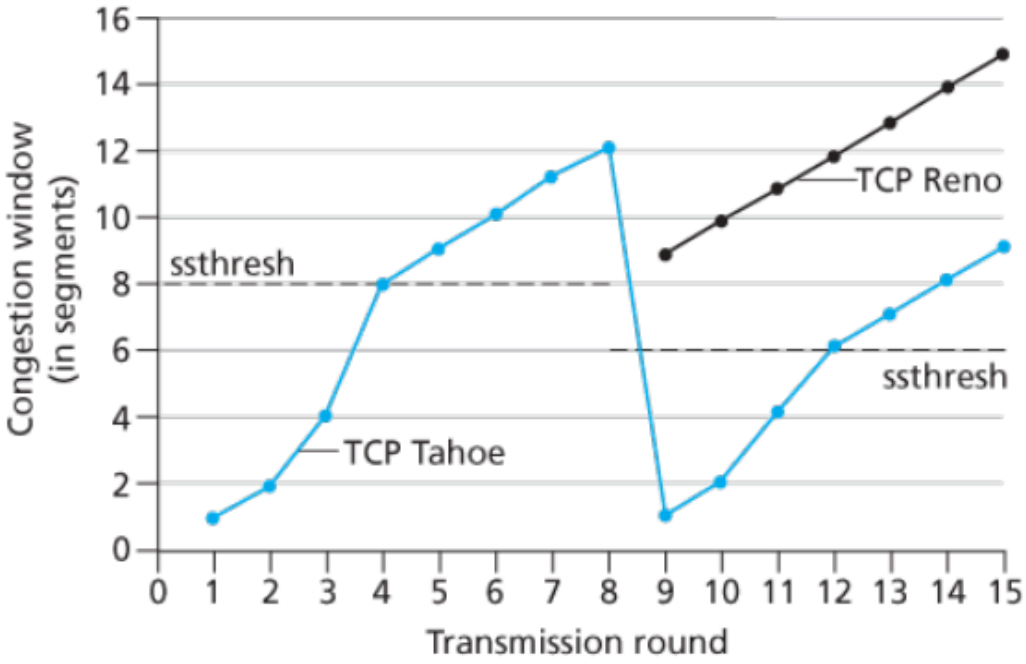
fast recovery

- In fast recovery, the value of cwnd is increased by 1 MSS for every duplicate ACK received for the missing segment that caused TCP to enter the fast-recovery state
- when ack arrives for missing packet TCP enters congesttion avoidance after deflating `cwnd`
- if time out occurs , fast recovery transitions to slow start after performing same actions as in slow start and congestion avoidance , i.e cwnd set to 1 MSS and ssthresh = cwnd /2

note :

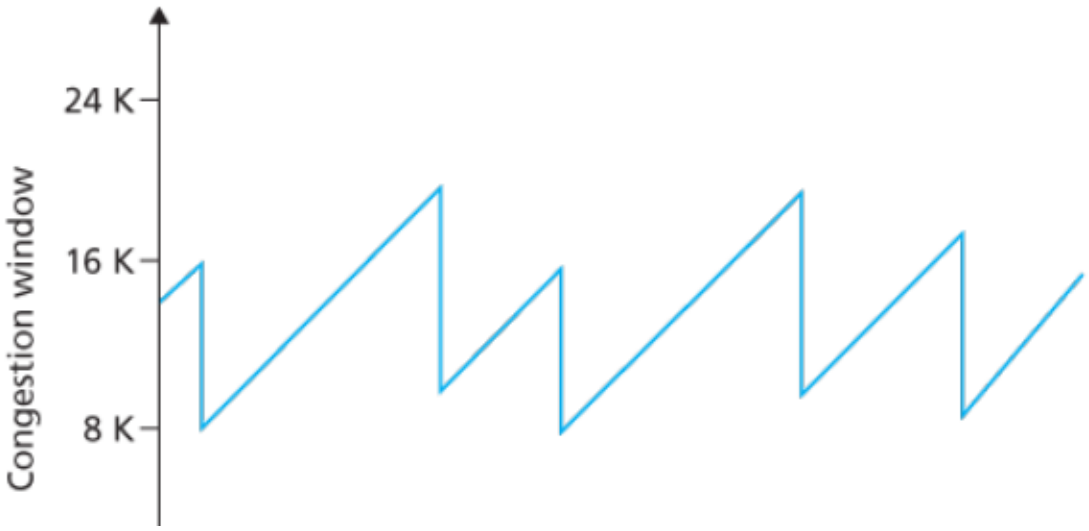
- earlier version of TCP, known as TCP Tahoe, unconditionally cut its congestion window to 1 MSS and entered the slow-start phase after either a timeout-indicated or triple-duplicate-ACK-indicated loss event.
- The newer version of TCP, TCP Reno, incorporated fast recovery.





other points :

- ignoring slow start , we see that TCP congestion control increases `cwnd` by 1 for every RTT (congn avoidance) and havles `cwnd` for triple duplicat ACK, so its called AIMD (additive increase multiplicative decrease)
- this is a sort of probing behaviour by tcp (increases window til loss occurs , when loss ocurs , half it nd start increasing again)
- saw tooth behaviour graph exhibited by AIMD



- TCP Vegas :

    - (1) detect congestion in the routers between source and destination before packet loss occurs, and
    - (2) lower the rate linearly when this imminent packet loss is detected.
    - Imminent packet loss is predicted by observing the RTT. The longer the RTT of the packets, the greater the congestion in the routers. (note that RTT was calculated dynamically refer to 3.5.3 in tb pg no280)
- average throughput of a connection=0.75 x W x RTT (W is value of cwnd when loss occurs ,rtt is round trip time)
- another formula average throughput of a connection=1.22 x MSS x RTT x L(loss rate)
- a congestion control mechanism is said to be fair if he average transmission rate of each connection is approximately R/K;that is, each connection gets an equal share of the link bandwidth. (R is bottleneck throughput and k is number of tcp connections)
- tcp congestion control is fair while udp isnt

tcp splitting :

- consider the delay in receiving a response for a search query. Typically, the server requires three TCP windows during slow start to deliver the response
- Thus the time from when an end system initiates a TCP connection until the time when it receives the last packet of the response is roughly 4·RTT (one RTT to set up the TCP connection plus three RTTs for the three windows of data) plus the processing time in the data center
- this leads to huge delay so what we can do instead is :
- (1)deploy front-end servers closer to the users, and (2) utilize TCP splitting by breaking the TCP connection at the front-end server.
- With TCP splitting, the client establishes a TCP connection to the nearby front-end, and the front-end maintains a persistent TCP connection to the data center with a very large TCP congestion window
- here response time becomes : 4·RTTFE + RTTBE+ processing time, where RTT FE is the round- trip time between client and front-end server, and RTT BE is the round-trip time between the front- end server and the data center.if front end server close to client rtt fe becomes negligle , so response time becomes rtt be + processing time

EXPLICIT CONGESTION NOTIFICATION

- a TCP sender receives no explicit congestion indications from the network layer, and instead infers congestion through observed packet loss
- At the network layer, two bits in the Type of Service field of the IP datagram header are used for ECN

- one setting of ECN bits is used by router to indicate that it is experiencing congestion .this indication is carried to destination which then informs source of congestion
- A second setting of the ECN bits is used by the sending host to inform routers that the sender and receiver are ECN-capable, and thus capable of taking action in response to ECN-indicated network congestion.
- when the TCP in the receiving host receives an ECN congestion indication via received datagram, the TCP in the receiving host informs the TCP in the sending host of the congestion indication by setting the ECE (Explicit Congestion Notification Echo) bit in TCP ACK segment
- TCP sender reacts to ACK with with ECE by halfing `cwnd` as it would react to a lost segment using fast retransmit, and sets the CWR (Congestion Window Reduced) bit in the header of the next transmitted TCP sender-to-receiver segment.

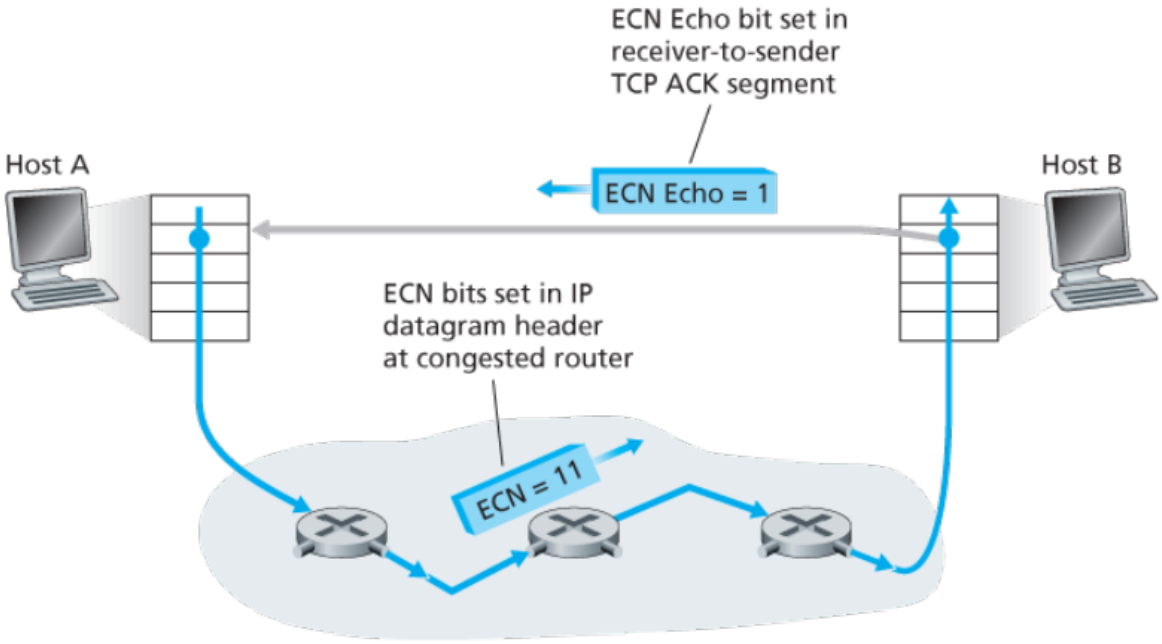


Figure 3.56 Explicit Congestion Notification: network-assisted congestion control

Network layer

- the primary role of the network control plane is to coordinate these local, per-router forwarding actions so that datagrams are ultimately transferred end-to-end, along paths of routers between source and destination host
- two functions :
- forwarding : When a packet arrives at a router’s input link, the router must move the packet to the appropriate output link

- Routing. The network layer must determine the route or path taken by packets as they flow from a sender to a receiver
- The control-plane approach is at the heart of software-defined networking (SDN) , where the network is “software-defined” because the controller that computes forwarding tables and interacts with routers is implemented in software. here a physically separate (from the routers), remote controller computes and distributes the forwarding tables to be used by each and every router.The remote controller is on control plane and the routers which just forward packets using given fowarding table are on data plane.
- services provided
  - Guaranteed delivery
  - Guaranteed delivery with bounded delay. This service not only guarantees delivery of the packet, but delivery within a specified host-to-host delay bound
  - In-order packet delivery.
  - Guaranteed minimal bandwidth.
  - security. The network layer could encrypt all datagrams at the source and decrypt them at the destination, thereby providing confidentiality to all transport-layer segments.
- internets network layer service is a best effort service
- Intserv is roposed service model extensions to the Internet architecture which aims to provide end-end delay guarantees and congestion-free communication

ROUTER

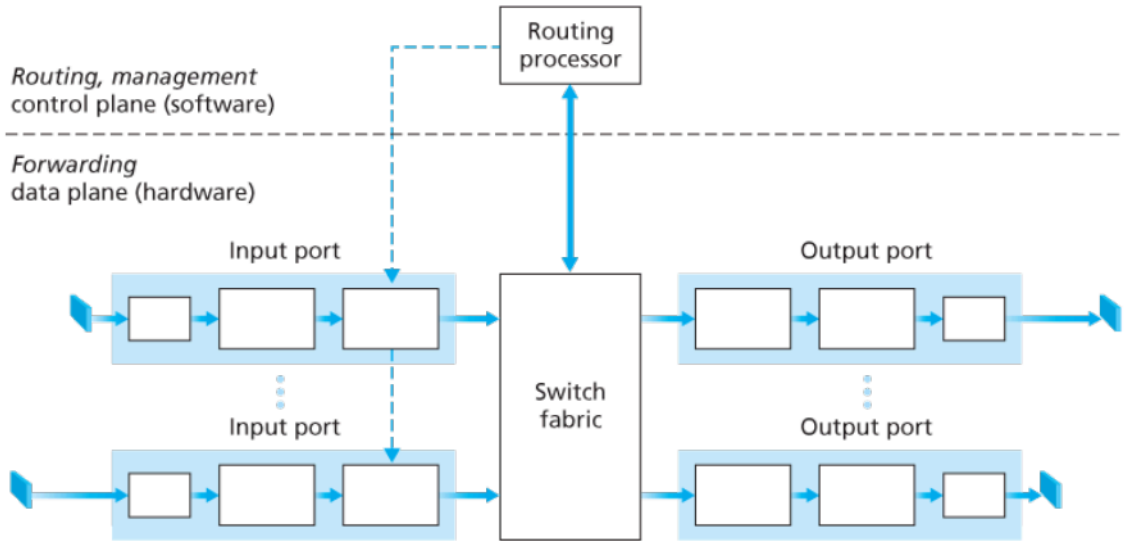


Figure 4.4 Router architecture

- input ports : It is here that the forwarding table is consulted to determine the router output port to which an arriving packet will be forwarded via the switching fabric.Control packets are forwarded from input port to routing processor. Juniper MX2020, edge router, for example, supports up to 960 10 Gbps Ethernet ports, with an overall router system capacity of 80 Tbps
- switching fabric : connects the router’s input ports to its output ports
- output ports : stores packets received from the switching fabric and transmits these packets on the outgoing link by performing the necessary link-layer and physical-layer functions.
- routing processor : performs control-plane functions. In traditional routers, it executes the routing protocols, maintains routing tables and attached link state information, and computes the forwarding table for the router.In SDN routers, the routing processor is responsible for communicating with the remote controller in order to receive forwarding table entries computed by the remote controller, and install these entries in the router’s input ports
- input ports,switching fabric and output ports are implemented in hardware always
- control plane functions are implemented on software and executed on routing processor
- two types of forwardinng destination based forwarding (as name indicates) and generalised forwarding (uses ip header fields in addn to destn to forward packets)
- Line termination converts the incoming electrical or optical signal from the connected network device (like a switch or another router) into a format compatible with the router's internal processing.

destination based forwarding

- a line card is act as the router's interface to the physical network (the first box before the port in the diagram)
- at the input port yhe router uses the forwarding table to look up the output port to which an arriving packet will be forwarded via the switching fabric.This forwarding table either computer by processor or recieved by SDN controller s copied from the routing processor to the line cards over a separate bus (e.g., a PCI bus) indicated by the dashed line from the routing processor to the input line cards in the image.With such a shadow copy at each line card, forwarding decisions can be made locally, at each input port, without invoking the centralized routing processor on a per-packet basis and thus avoiding a centralized processing bottleneck
- the forwarding table usually links a range of ip adress with a particular outgoin link
- router uses the longest prefix matching rule; that is, it finds the longest matching entry in the table and forwards the packet to the link interface associated with the longest prefix match.but this linear search may take up more time than required so instead we can use :

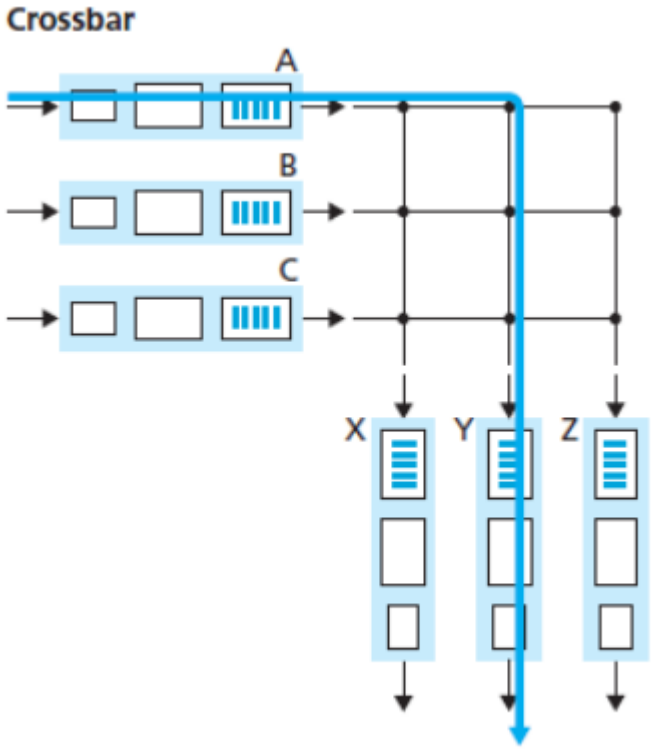
- Ternary Content Addressable Memories (TCAMs) , where a 32-bit IP address is presented to the memory, which returns the content of the forwarding table entry for that address in essentially constant time.
- the process of looking up a destination IP address (“match”) at input port and then sending the packet into the switching fabric to the specified output port (“action”) is also known as "match plus action" abstraction which is performed not only in routers but also switches,firewalls and NAT

switching

- switching via memory :
  - An input port with an arriving packet first signaled the routing processor via an interrupt. The packet was then copied from the input port into processor memory. The routing processor then extracted the destination address from the header, looked up the appropriate output port in the forwarding table, and copied the packet to the output port’s buffers.
  - if memory bandwitdth is B packets/s that can be written/read to memory , overall forwarding throughput is always =B/2
  - here two packets cant be forwarded at same time
  - modern routers use switching via memory but packet lookup,processing and forwarding are instead done on input line cards
  - Cisco’s Catalyst 8500 series nternally switches packets via a shared memory
- switching via bus:
  - input port transfers packet directrly to output port without intervention of router.
  - input port inspects packet and find the destination adress. then it prepends a switch-internal label (header) to the packet which indicates the output port the packet must go to
  - the packet is transmitted on the bus and only the output port matching the label will capture the packet
  - switching speed of the router is limited to the bus speed
  - CISCO 6500 router switches packet over 32gbps bus
- switching via an interconnection network :
  - it uses a a crossbar switch which is an interconnection network consisting of 2N buses that connect N input ports to N output ports
  - When a packet arrives from port A and needs to be forwarded to port Y, the switch controller closes the crosspoint at the intersection of busses A and Y, and port A then sends the packet onto its bus, which is picked up (only) by bus Y. Note that a packet

from port B can be forwarded to port X at the same time, since the A-to-Y and B-to-X packets use different input and output busses.

- it basically employs multi switching fabric (switching logic is done by switching fabric itself)
- it is therefore capable of sending multiple packets in parallel
- it is non blocking unless 2 packets from different ports forwarded to same output port , then one will have to wait
- CISCO 1200 uses interconnection network while CISCO 7600 can be configured as bus or interconnection
- Cisco CRS employs a three-stage non-blocking switching strategy.



output port processing

- HOL (head of line blocking) : queued packet in an input queue must wait for transfer through the fabric (even though its output port is free) because it is blocked by another packet at the head of the line. this is input queueing
- packet queues can form at the output ports even when the switching fabric is N times faster than the port line speeds this results in output queueing.when many packets are queued at output port a packet scheduler is used to determine which packet is to be transmitted next and so on

- packet dropping policies called AQM - active queue management.a widely implemented AQM algorithm is RED Random Early Detection.
- rule of thumb for buffering size : the amount of buffering (B) should be equal to an average round-trip time (RTT) times the link capacity (C)
- when there are a large number of TCP flows (N) passing through a link, the amount of buffering needed is B=RTT x C /N.
- FIFO packet scheduling :selects packets for link transmission in the same order in which they arrived at the output link queue.
- Under priority queuing, packets arriving at the output link are classified into priority classes upon arrival at the queue. ex : In practice, a network operator may configure a queue so that packets carrying network management information (e.g., as indicated by the source or destination TCP/UDP port number) receive priority over user traffic; additionally, real-time voice-over-IP packets might receive priority over non-real traffic such as SMTP or IMAP e-mail packets
- Round Robin and Weighted Fair Queuing (WFQ) ; Under the round robin queuing discipline, packets are sorted into classes as with priority queuing. However, rather than there being a strict service priority among classes, a round robin scheduler alternates service among the classes.
- WFQ differs from round robin in that each class may receive a differential amount of service in any interval of time. Specifically, each class, i, is assigned a weight, w

IP PROTOCOL

- network layer protocol

IPV4

IPV4 DATAGRAM FORMAT

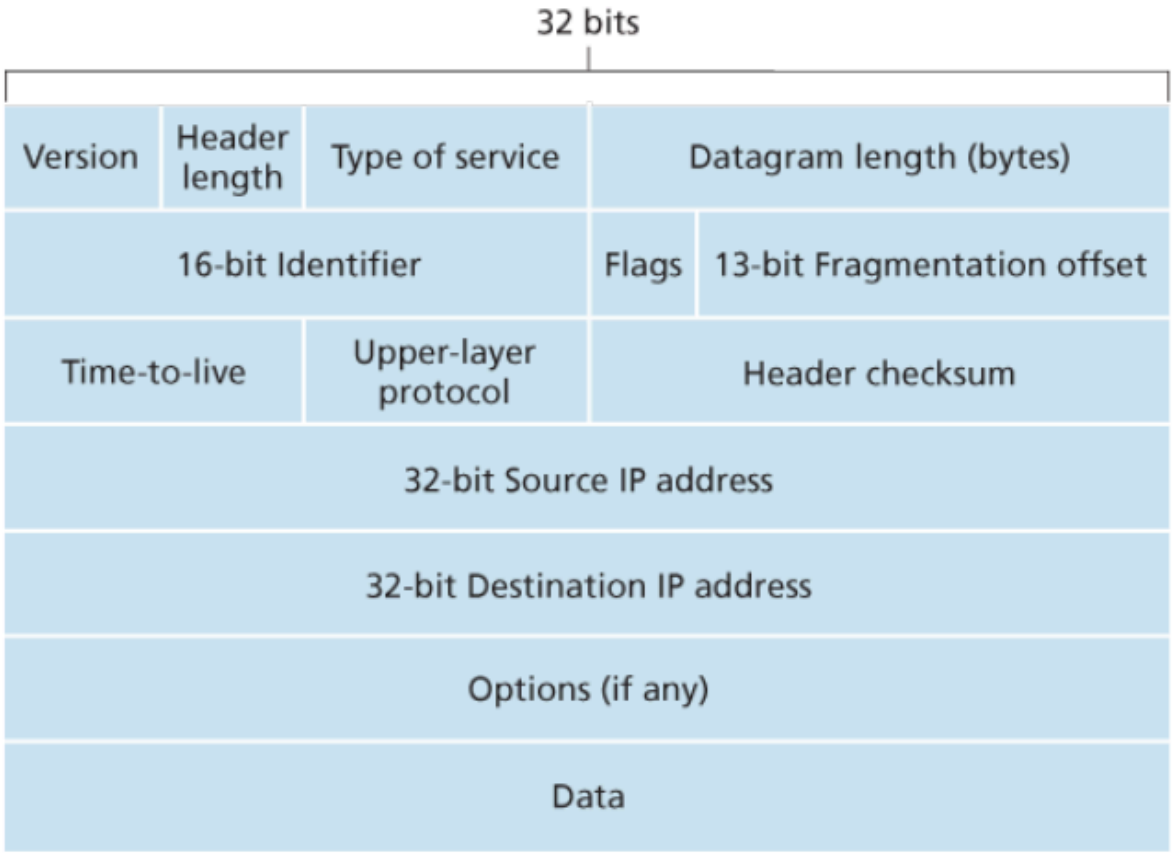


Figure 4.16 IPv4 datagram format

- Version (4 bits)
- Header legngth(4 bits): determines where in IP datagram payload is
- Type of Service(8 bits): to distinguish different types od datagrams like real time from non real time.also used for ECN
- Upper layer protocol : indicates transport layer protocol , data portion of IP is passed to.6 indicates TCP while 17 indicates UDP
- header checksum : header checksum is computed by treating each 2 bytes in the header as a number and summing these numbers using 1s complement arithmetic
- IP datagram has 20 bytes of header.If IP carries TCP segment then it has 40 bytes of header

fragmentation

- he maximum amount of data that a link-layer frame can carry is called the maximum transmission unit (MTU).
- Routers incoming and outgoing links can have different MTU therefore IP datagram must be fragmented

- reassembly of fragment is done at end system.this is done by using identification number,flag and fragmentation offset.
- . When the destination receives a series of datagrams from the same sending host, it can examine the identification numbers of the datagrams to determine which of the datagrams are actually fragments of the same larger datagram.
- the offset field is used to specify where the fragment fits within the original IP datagram.
- the last fragment has a flag bit set to 0, whereas all the other fragments have this flag bit set to 1
- first fragment has offset 0 , while other fragments have offset governed by formula offset = total length of fragment - header length / 8
- offset must be a multiple of 8(lfragment len- header len must be divisble by 8) , so chose a fragment size such that offset is integer (non decimal)

ipv4 adressing

- 223.1.2.0/24 indicates that first 24 bits are subnet adress
- in CIDR classless interdomain routing : 32-bit IP address is divided into two parts and again has the dotted-decimal form a.b.c.d/x, where x indicates the number of bits in the first part of the address.
- routing protocl among ISPs : BGP
- class A adressing is /8(8 bits so 2^8 subnet tnetwork , 2^24-2 hosts), B is /16 , C is /24 class D is unicast white class E is multicast.Note that for all these classes 2 adresses are reserved :-network adres x.x.x.0 and broadcast adress x.x.x.255
- ICANN gives ip adresses to ISPs which inturn give IPs to organisations
  **

Class	HOB	NET ID Bits	Host ID Bits	No of Networks	Host Per Network	Start Address	End Address
Class A	0	8	24	2^7=128	2^{24}=16,777,216	0.0.0.0	127.255.255.255
Class B	10	16	16	2^{14}=16,384	2^{16}=65,536	128.0.0.0	191.255.255.255
Class C	110	24	8	2^{21}=2,097,152	2^8=256	192.0.0.0	223.255.255.255
Class D	1110	-	-	-	-	224.0.0.0	239.255.255.255
Class E	1111	-	-	-	-	240.0.0.0	255.255.255.255

- hob is host operating block (this bit is reseved)

DHCP

- A network administrator can configure DHCP so that a given host receives the same IP address each time it connects to the network, or a host may be assigned a temporary IP address that will be different each time the host connects to the network.
- note that DHCP is an application layer protocol acting on behalf of network layer
- it is a client server protocol
- each subnet has DHCP server , if it doesnt then it needs a DHCP relay agent which knows adress of DHCP server for the network
- 4 steps in DHCP :
    - DHCP server discovery : first task is to find DHCP server.Client sends DHCP discover message (UDP packet encapsulated within IP packet) with the broadcast destination IP address of 255.255.255.255,port 67 and a “this host” source IP address of 0.0.0.0.The DHCP client passes the IP datagram to the link layer, which then broadcasts this frame to all nodes attached to the subnet
    - DHCP server offer : A DHCP server receiving a DHCP discover message responds to the client with a DHCP offer message that is broadcast to all nodes on the subnet, using the IP broadcast address of 255.255.255.255.Each server offer message contains transaction ID of the received discover message, the proposed IP address for the client, the network mask, and an IP address lease time—the amount of time for which the IP address will be valid
    - DHCP request : e newly arriving client will choose from among one or more server offers and respond to its selected offer with a DHCP request message, echoing back the configuration parameters.
    - DHCP ACK. The server responds to the DHCP request message with a DHCP ACK message , confirming the requested parameters.
- yiaddr refers to your internet adress

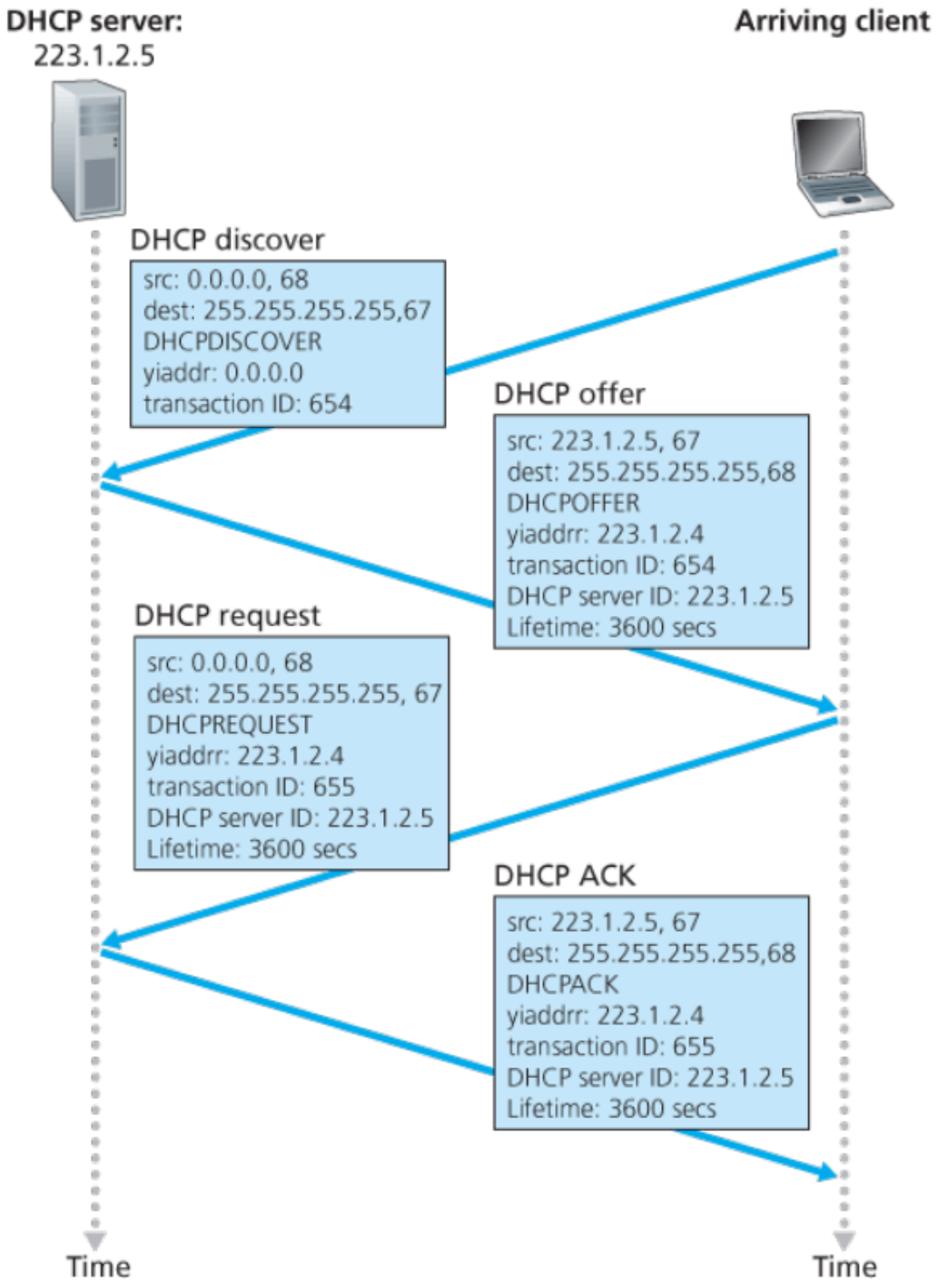
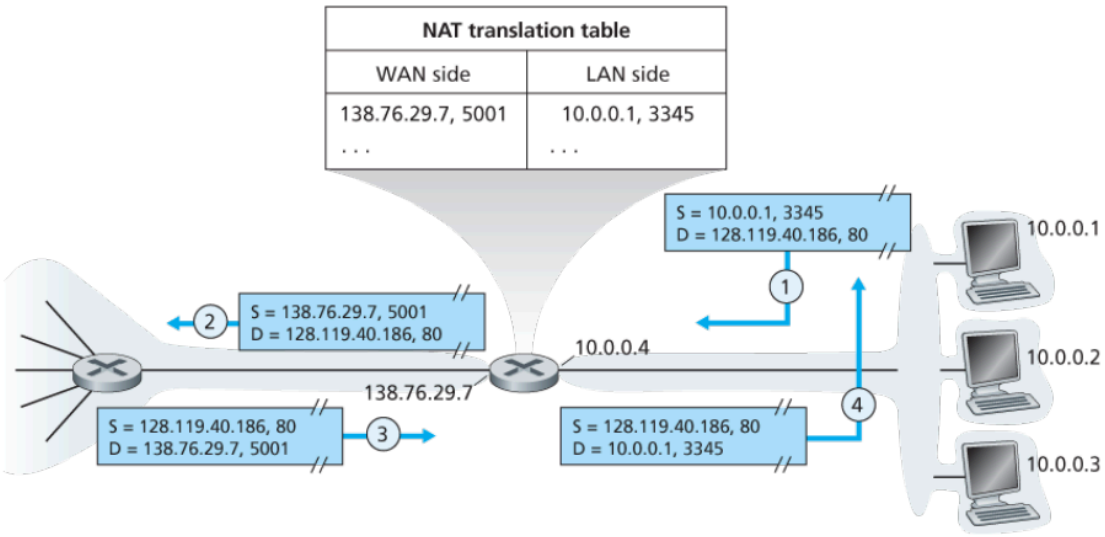


Figure 4.24 DHCP client-server interaction

NAT

- network adress translation



- the NAT router behaves to the outside world as a single device with a single IP address.In essence, the NAT-enabled router is hiding the details of the home network from the outside world
- f all datagrams arriving at the NAT router from the WAN have the same destination IP address (specifically, that of the WAN-side interface of the NAT router), then how does the router know the internal host to which it should forward a given datagram? The trick is to use a NAT translation table at the NAT router, and to include port numbers as well as IP addresses in the table entries.
- basically a NAT makes a particular port nnumber on WAN side for a particular combination of ip and port number on LAN side. ex 10.0.0.1 , 3345 is 138.76.29.7 , 5001 while 10.0.0.2 , 4421 is 138.76.29.7 5004
- disadvantages of NAT :
  - port numbers are meant to be used for addressing processes, not for addressing hosts. This violation can cause problems for servers running on the home network, since, server processes wait for incoming requests at well-known port numbers and peers in a P2P protocol need to accept incoming connections when acting as servers. Technical solutions to these problems include NAT traversal tools [RFC 5389] and Universal Plug and Play (UPnP), a protocol that allows a host to discover and configure a nearby NAT
  - Here, the concern is that routers are meant to be layer 3 (i.e., network-layer) devices, and should process packets only up to the network layer. NAT violates this principle that hosts should be talking directly with each other, without interfering nodes modifying IP addresses, much less port numbers.Basically theres a loss of transparency
- attacks can do port scans using ICMP echo messages