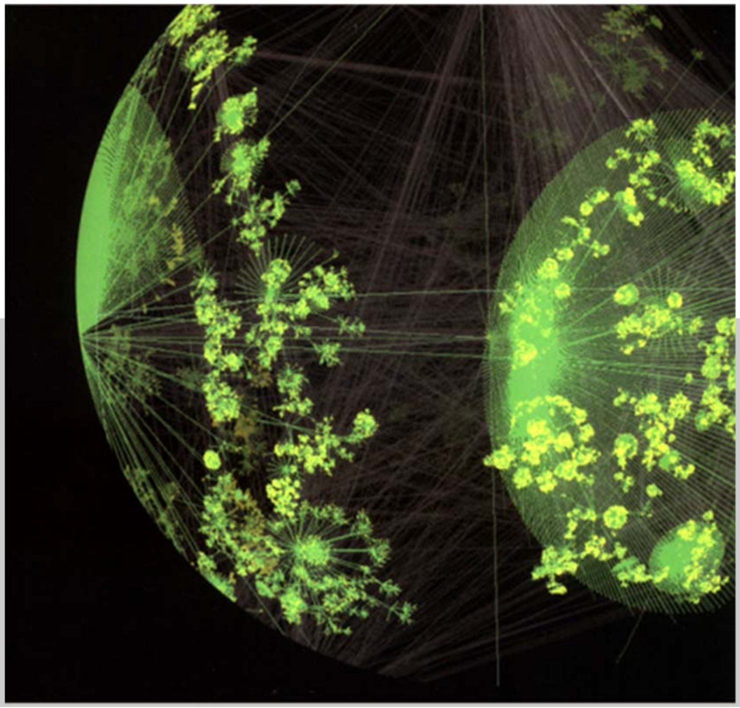


EL5373 Review 2

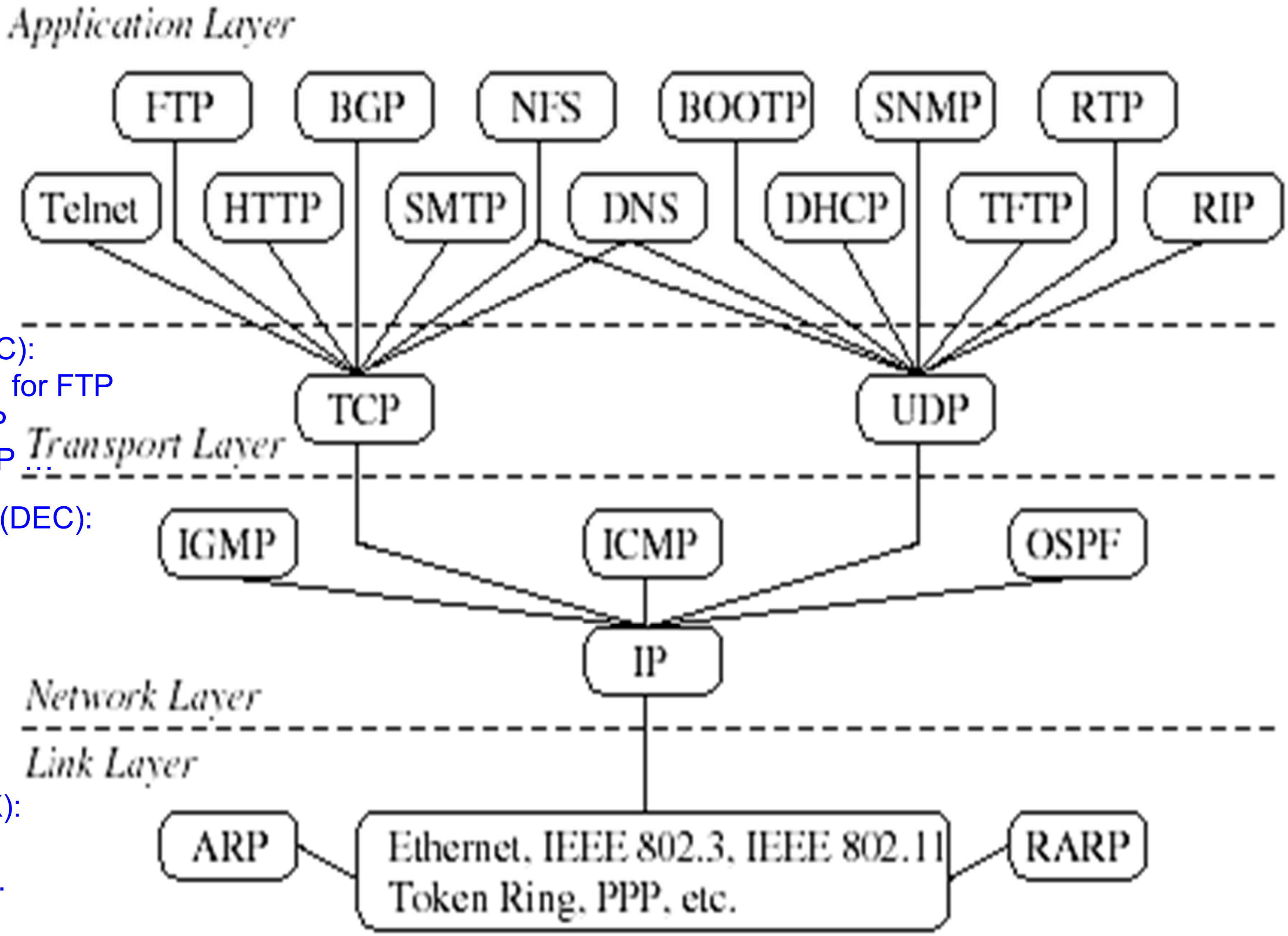
TCP/IP Essentials
A Lab-Based Approach

Spring 2017

Caveat

- This slide deck doesn’t cover the materials learnt before the midterm exam. Please use the review slides previously distributed for your study.
- IPv6 is recently covered as a supplement topic. Although this deck of slides does not cover it, students are required to study the lecture notes as discussed in the class for the final exam.
- Labs are an important part of this course. Lab related questions are made based on what you’ve observed from the carried exercises in nine labs.

Protocols in Different Layers



Port Number (DEC):
- TCP/20, TCP/21 for FTP
- UDP/520 for RIP
- TCP/80 for HTTP …

Protocol Number (DEC):
- 1 for ICMP
- 2 for IGMP
- 6 for TCP
- 17 for UDP
- 47 for GRE
- 89 for OSFP …

Frame Type (HEX):
- 08:00 for IP
- 08:06 for ARP …

IP Packet Forwarding from Source to Destination

- Find out the IP address by DNS query for a given domain name of the destination
- If the destination is
  - in the same network (or subnet), send the packet directly to the destination
  - in a different network, a router is needed to forward the datagram
- If no router available to reach the destination, drop the packet
- IP packets have to be encapsulated in a link layer frame (e.g., Ethernet frame)
  - A link layer frame can only be sent within the same network (or subnet)
  - The link layer frame has to be sent with the MAC address of the other end
    - > ARP

Communications in the Same Network/Subnet
What is "the Same Network/Subnet"?

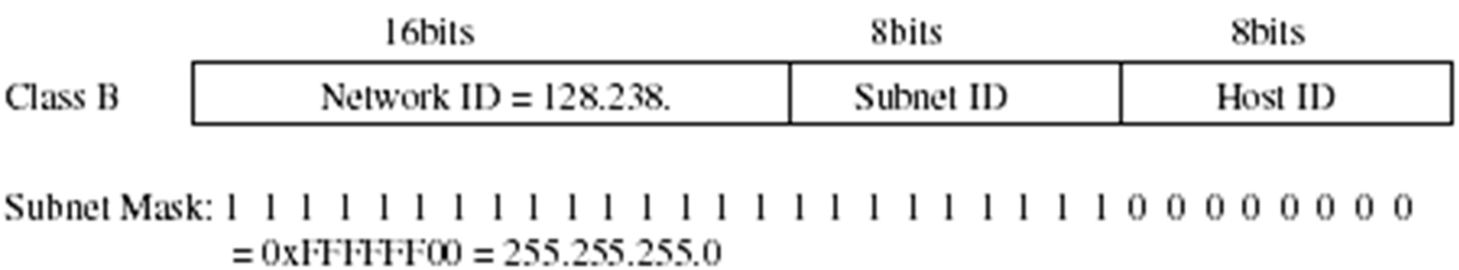
- Host X wants to send IP packets to host Y
- What does the X know
  - X's IP address
  - X's subnet mask
  - Y's IP address
- Computation by X
  - X's network/subnet ID: (X's IP add) & (X's subnet mask)
  - Y's network/subnet ID: (Y's IP add) & (X's subnet mask)
  - If the above two results are the same, X believes that Y is in the same network/subnet
- If X and Y have different subnet masks, they may have different calculation results
  - Each calculates network/subnet ID by using its own subnet mask

Subnetting

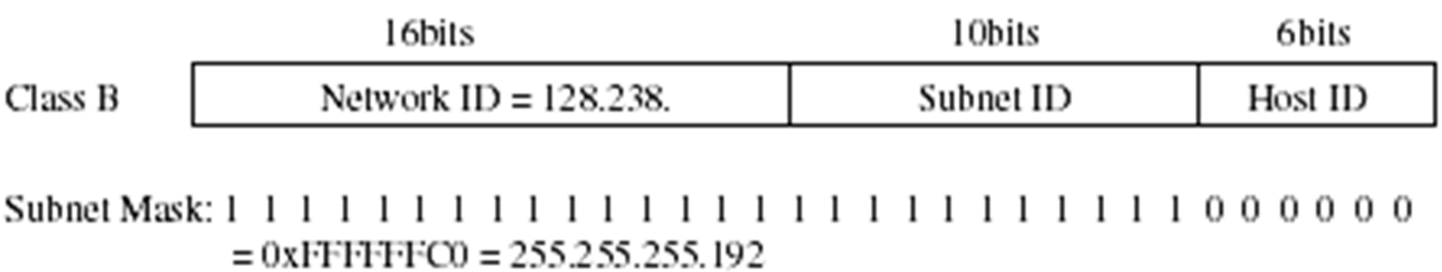
- Use three levels of an IP address:
  - Network ID
  - Subnet ID
  - Host ID
- Subnet mask: separates subnet ID and host ID

Here is another subnetting example with two masks in a design of 640 subnets.

- Use a /24 mask to define 128 subnets, say with subnet address from 128.238.0.0 to 128.238.127.0
- Use a /26 mask to define another 512 subnets, say in this same example with subnet address, from 128.238.128.0 to 128.238.255.192



128.238.0.0/24 net then contains: 2^8 = 256 subnets with 2^8-2 = 254 hosts in each subnet



128.238.0.0/26 net then contains: 2^{10} = 1024 subnets with 2^6-2 = 62 hosts in each subnet

CIDR – Type Address

- IP address in CIDR (Classless Inter-Domain Routing)
  - Not classified into classes
  - Two components of an IP address
    - Network prefix ranging from 13 to 27 bits – a Variable Length Subnet Mask
    - Host ID using the remaining bits
  - Slashed-notation

  A dotted-decimal IP address + */* + *Number of bits used for the network prefix*
- Network address are assigned in a hierarchical manner.
- In the core network, routing entries for networks with the same higher level prefix, a CIDR block, can be summarized into one entry – i.e. supernetting for route aggregation
- The longest-prefix-matching rule is still used in table lookups

Communications between Two Network Segments (in the Same Network)

- Two segments connected by a bridge, host X in segment 1 and host Y in segment 2
- Assume that at the beginning,
  - the ARP tables of X and Y are empty
  - the bridge already has correct entries for X and Y in its filtering database
- X tries to send an IP packet to Y
  - X broadcasts an ARP request to resolve Y's MAC address
  - The bridge forwards the ARP request to segment 2, and any other segments
  - Y sends an ARP reply destined to X
  - The bridge forwards the ARP reply to segment 1
  - X sends out an Ethernet frame containing the IP packet
  - The bridge forward the frame to segment 2 for Y
- In each packet, what are the values in the following fields?
  - IP: source IP address, destination IP address
  - ARP: sender IP address, target IP address
  - Ethernet for ARP: source Ethernet address, destination Ethernet address
  - Ethernet for IP: source Ethernet address, destination Ethernet address

Communications between Two Networks

- Two networks connected by a router, host X in network 1 and host Y in network 2
- Assume that at the beginning,
  - the ARP tables of X, Y and the router are empty
  - the router already has entries for X and Y in its routing table
- X tries to send an IP packet to Y
  - X broadcast an ARP request (in network 1) to resolve the router's MAC
  - The router sends an ARP reply to X
  - X sends the IP packet to the router
  - The router broadcasts an ARP request (in network 2) to resolve Y's MAC
  - Y sends an ARP reply to the router
  - The router forwards the IP packet to Y
- A router NEVER forwards an ARP message. (Why?)
- In each packet, (how many?) what is the value in the following fields?
  - Source/sender IP address, Destination/target IP address
  - Source/sender Ethernet address, destination/target Ethernet address

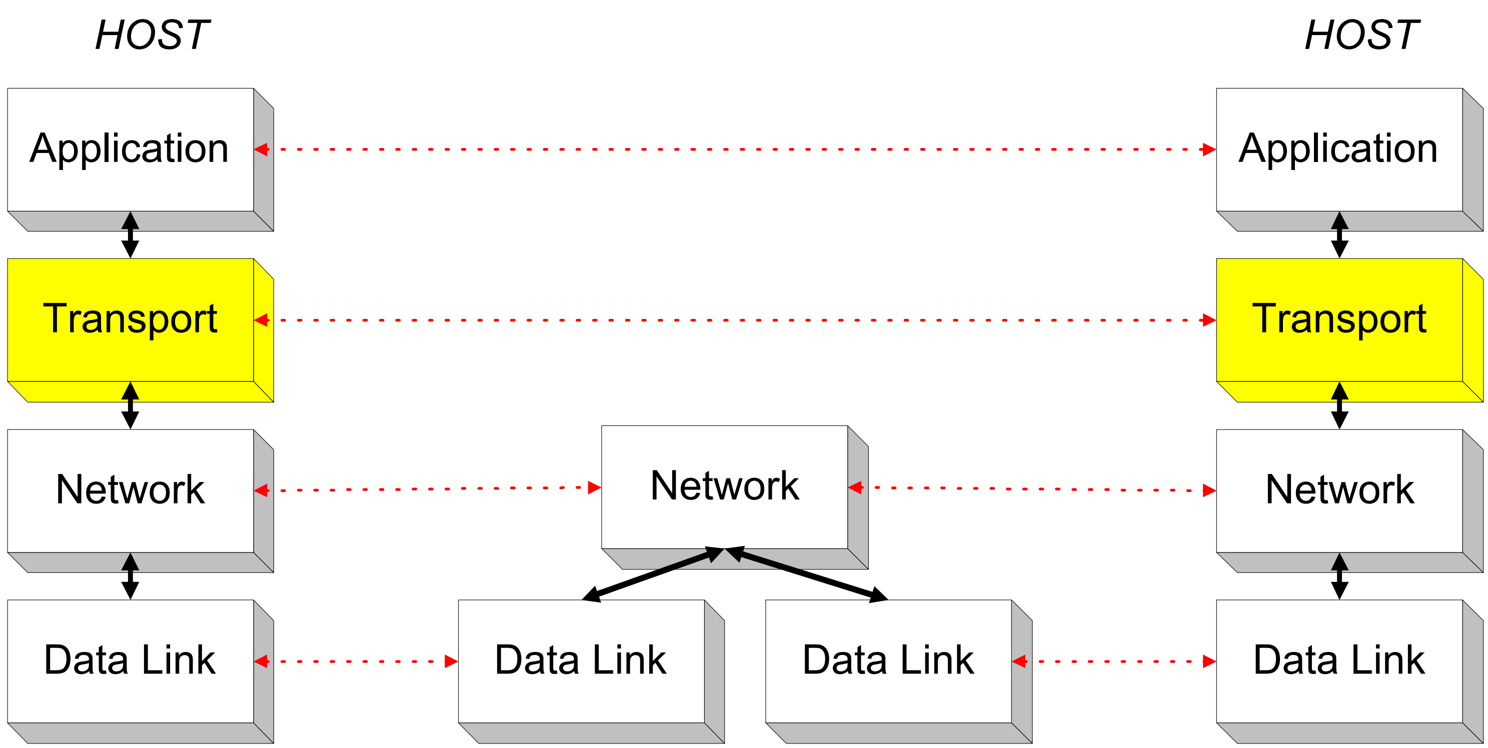
Transport Protocols

Two Transport Protocols discussed

- User Datagram Protocol (UDP)
- Transmission Control Protocol (TCP)

Transport Layer Protocols

- Transport layer protocols are end-to-end protocols
- Their headers are not examined by intermediate routers



UDP and TCP

The Internet supports two transport protocols:

UDP – User Datagram Protocol

- Datagram oriented
- Unreliable, connectionless
- Simple
- Unicast and multicast
- Commonly used for network control signaling services
  - Network management (SNMP), routing (RIP), naming (DNS), etc.
- Useful for increasing number of applications, e.g., multimedia applications

TCP – Transmission Control Protocol

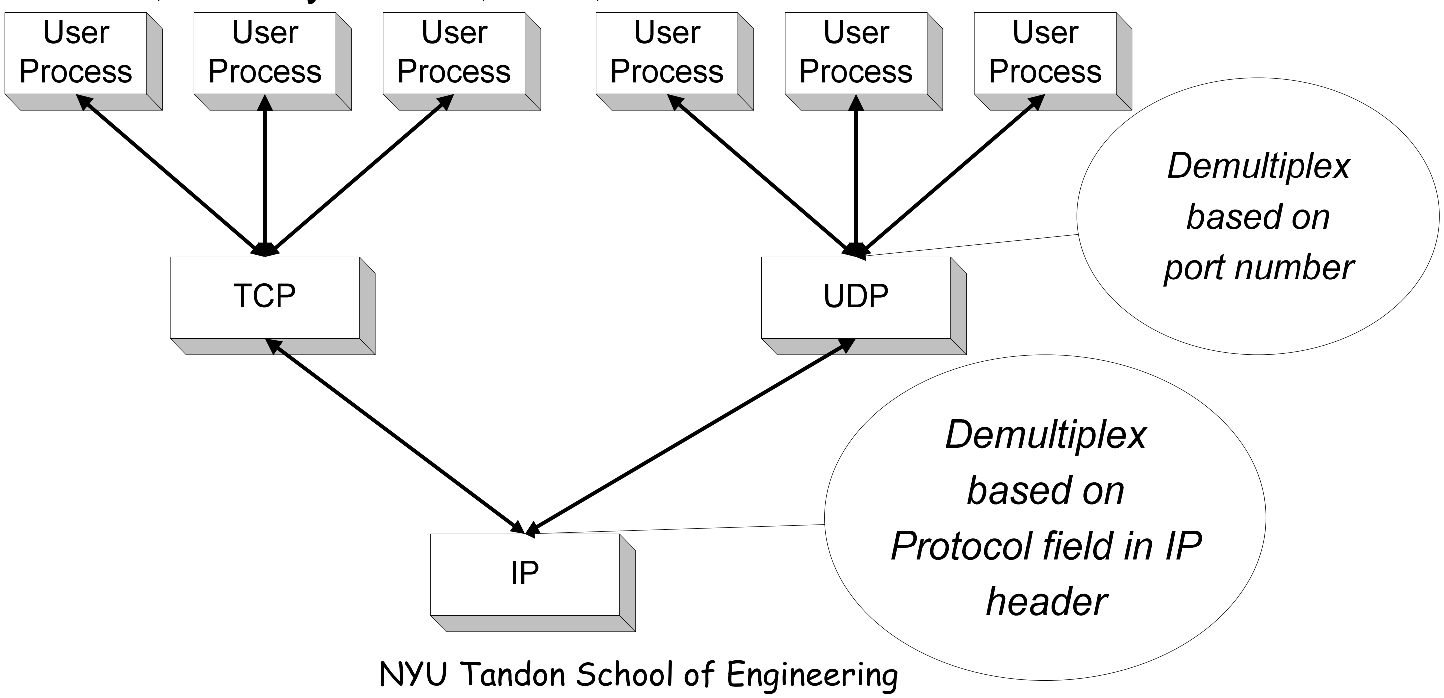
- Stream oriented
- Reliable, connection-oriented
- Complex
- Unicast only
- Currently used by most Internet applications:
  - Web (HTTP), email (SMTP), file transfer (FTP), terminal (telnet), etc.

Port Numbers

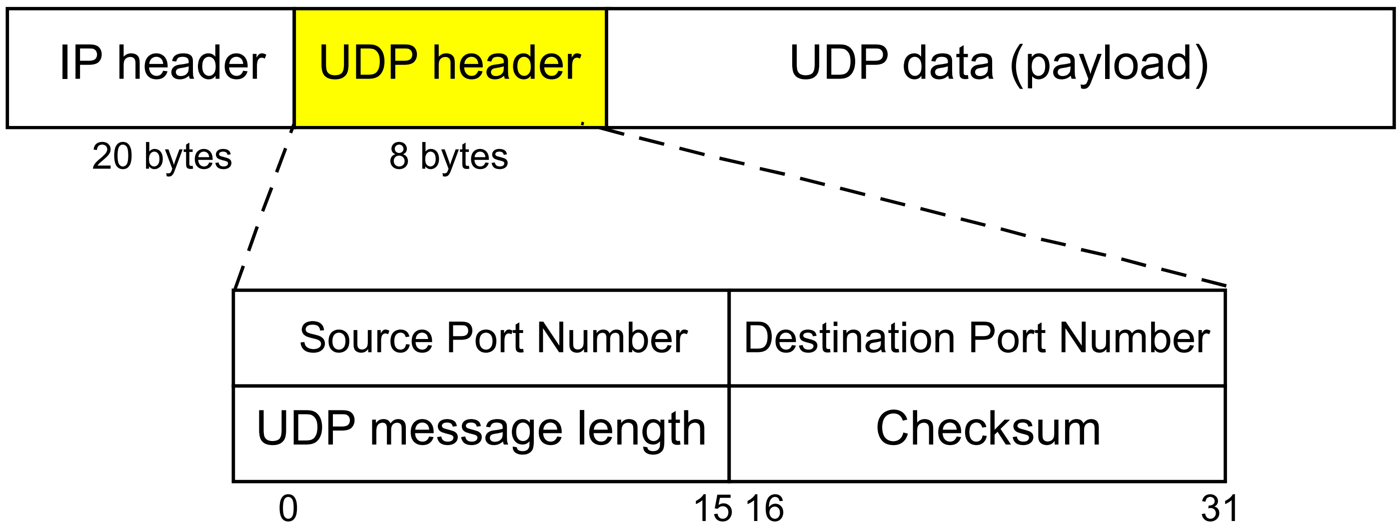
- UDP (and TCP) use port number to identify the supported application
- A globally unique flow of host application can be identified by a 5-tuple

<Src. IP, Dst IP, Src. Port, Dst. Port, Protocol No.>

- There are 65,535 UDP ports available per host
  - dynamic/private , used by clients, randomly picked, >49,151 (per IANA)
  - Registered, used by ordinary user processes, 1024 – 49,151
  - well-known, used by servers, fixed, 1~1023



UDP Format



- Port Numbers identify sending and receiving applications (processes). The maximum value for a port number is 2^{16}-1= 65,535
- Message Length is between 8 bytes (i.e., data field can be empty) and 65,535 bytes (length of UDP header and data in bytes)
- Checksum is for UDP header and UDP data

UDP Checksum

- Optional
  - set all 0's if not calculated
  - A calculated checksum can never be all 0's.
- Computed using the UDP header, UDP data and a pseudo-header as below.
- All fields of pseudo-header are available in UDP layer

<table>
<tr><td colspan="3">32-bit Source IP Address</td></tr>
<tr><td colspan="3">32-bit Destination IP Address</td></tr>
<tr><td>0x00</td><td>8-bit Protocol (0x17)</td><td>16-bit UDP Length</td></tr>
</table>

* check RFC 2460 for the pseudo-header definition with IPv6

Maximum Transmission Unit (MTU)

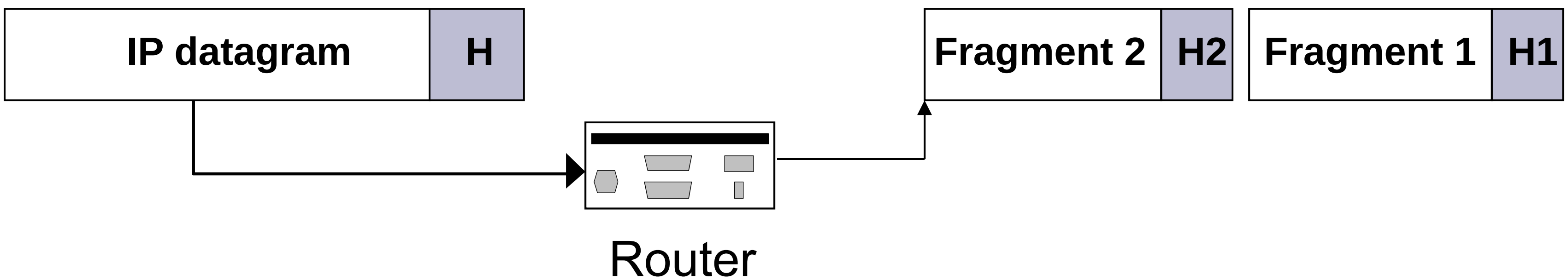
- The frame size limit of data link protocol specifies a limit on the size of the IP datagram that can be encapsulated by the protocol.
- This limit is called Maximum Transmission Unit (MTU)
- MTUs for various data link layers:

Ethernet: 1500	FDDI: 4352
802.2/802.3: 1492	ATM AAL5: 9180
802.5: 4464	PPP: 296 (low delay)

- What if the size of an IP datagram exceeds the MTU?
  - IP datagram is fragmented into smaller units.
- What if the route contains networks with different MTUs?
  - The smallest MTU of any data link is used as the Path MTU.

Where is Fragmentation done in IPv4?

- Fragmentation can be done at the sender or at intermediate routers.
- The same datagram can be fragmented several times.
- Reassembly of original datagram is only done at destination hosts.



- How does the IP fragmentation get carried with IPv6?

IP Header Fields for Fragmentation

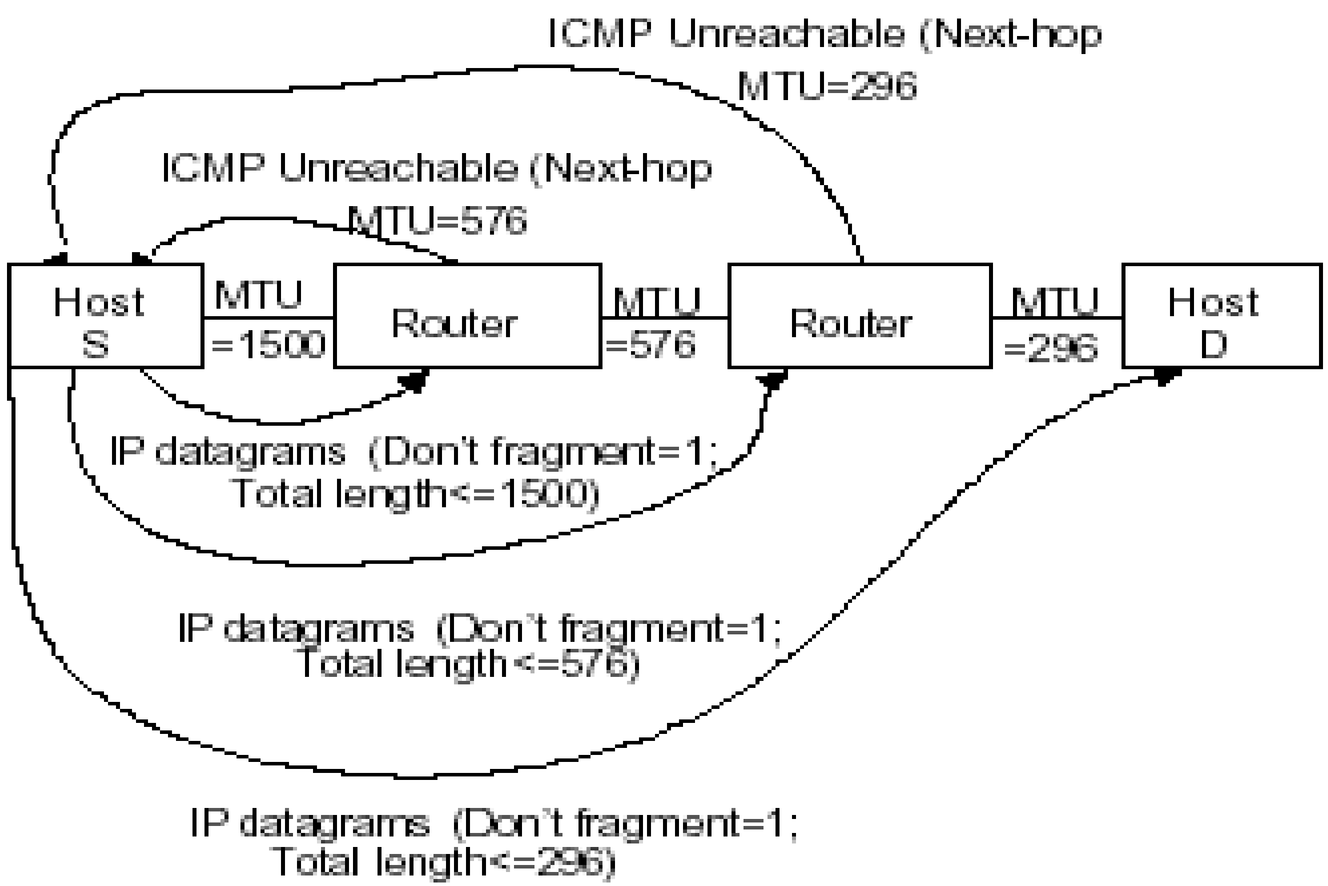
The following fields in the IP header are involved:

Version	**Header Length**	**Type of Service (TOS)**	Total Length (bytes)			
Identification			0	DF	MF	Fragment Offset (8-bytes units)
Time-To-Live (TTL)		**Protocol Type**	**Header Checksum (16 bits)**			
… …						

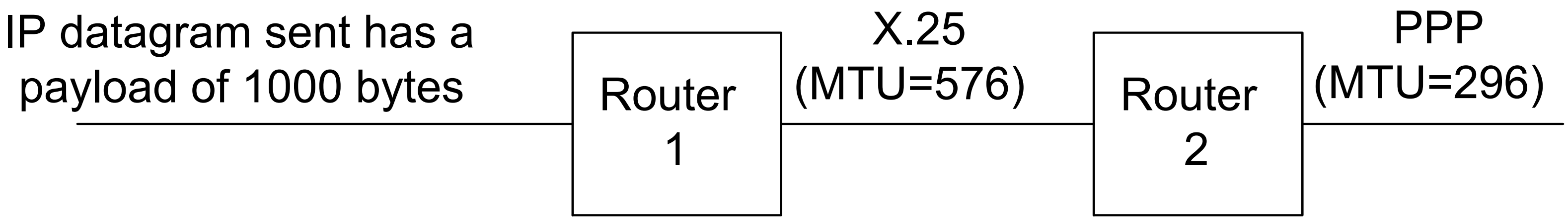
- Identification is the same in all fragments.
- Flags field contains
  - a reserved bit, must be zero,
  - a Don't Fragment (DF) bit that can be set, and
  - a More Fragments (MF) bit.
- Fragment Offset contains the offset (<u>in 8-byte units</u>) of current fragment in the original datagram.
- Total Length is changed to be the size of the fragment.

Path MTU Discovery

A host sends a set of IP datagrams with various lengths and the “don’t fragment” bit set



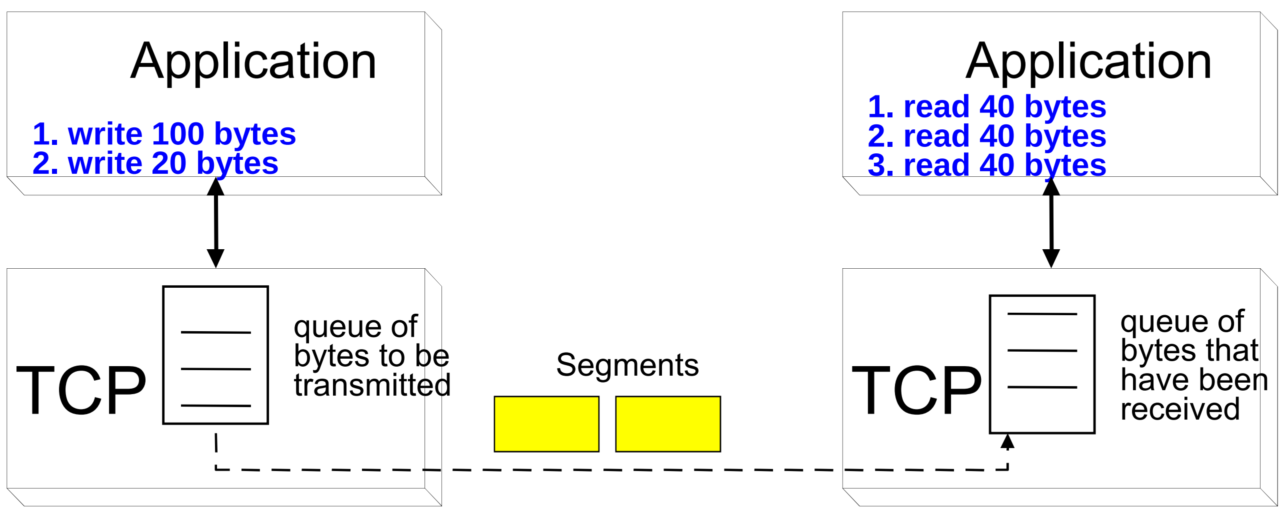
Fragmentation through Multiple Links



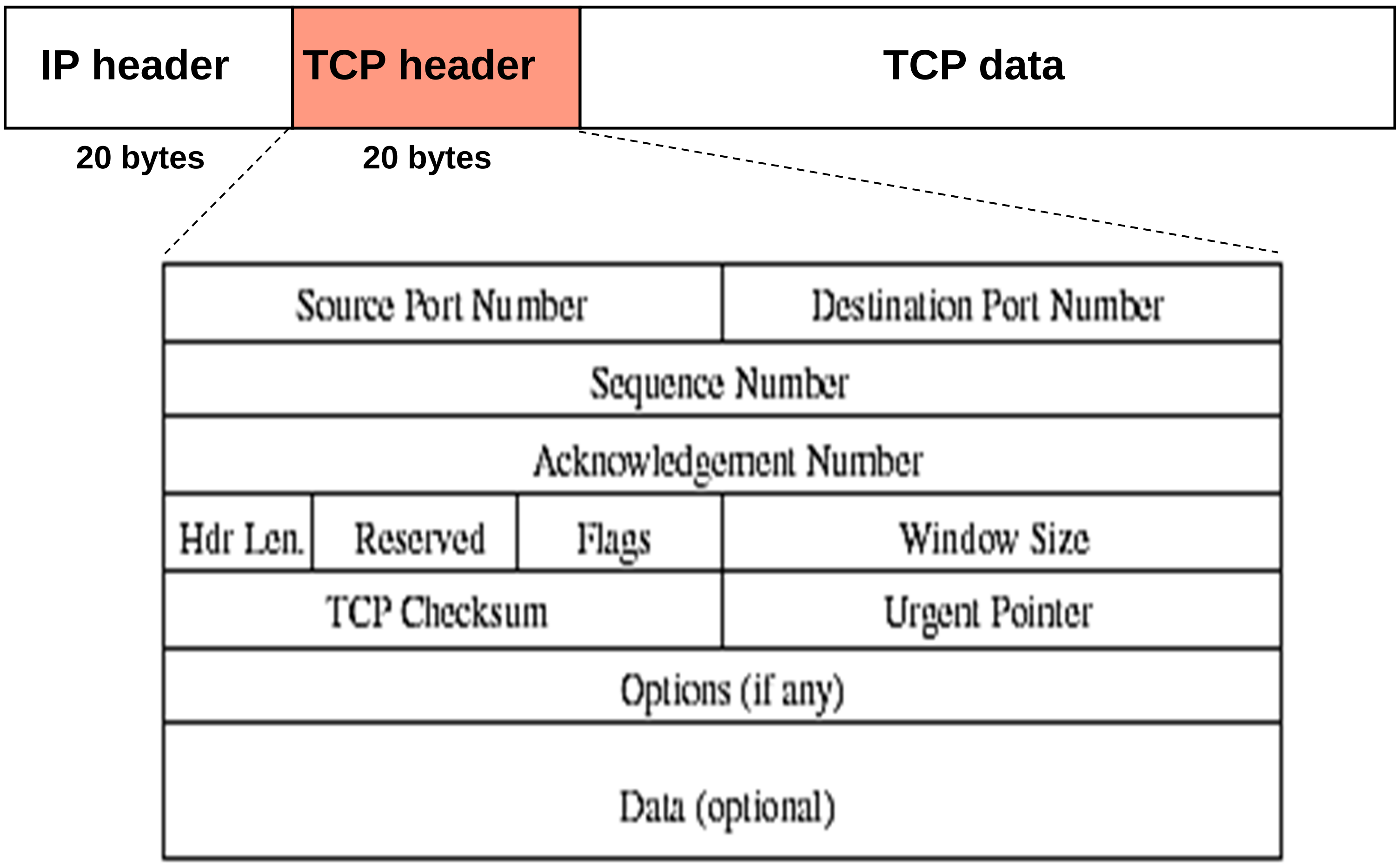
- **The ID field stays the same for all fragments of a datagram sent by a sender to allow for reassemble**
- **The fragment offset is relative to the datagram sent by the sender.**
- **Two fragments created on X.25 link (offsets 0, 69)**
  - **576 – 20 (IP header) = 556; 552 divides by 8 as 69.**
  - **First fragment: Offset 0, bytes 1~552; second fragment: Offset 69, bytes 553~1000**
- **Each fragment is fragmented further on the PPP link**
  - **ID stays the same on all fragments**
  - **Fragment offset on the second set of fragments is relative to the original (0, 34, 68, 69, 103)**
    - **296-20=276; 272/8 = 34**

TCP – A Byte Stream Service

- To the lower layers, TCP handles data in blocks – the segments.
- To the higher layers TCP handles data as a sequence of bytes and does not identify boundaries between bytes
- Higher layers do not know about the beginning and end of segments!



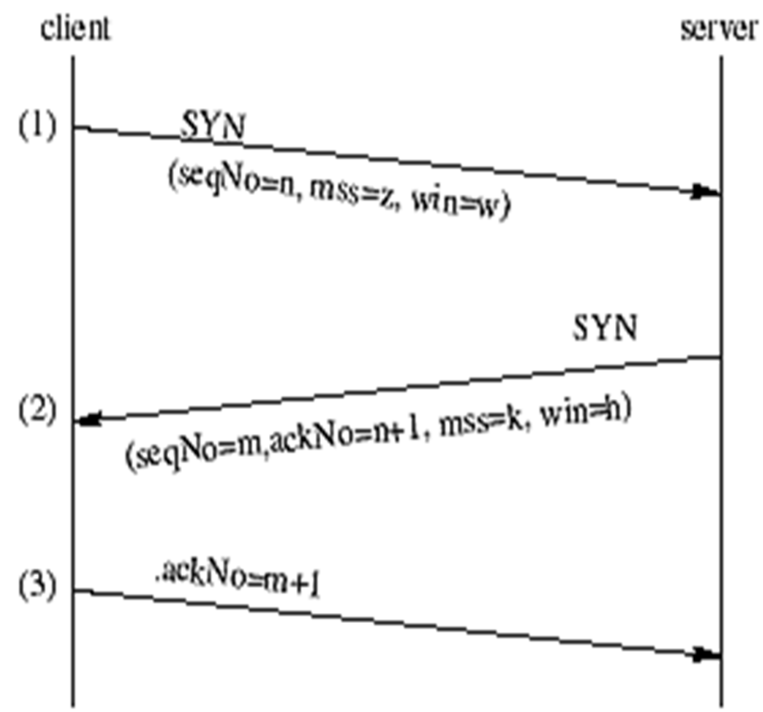
TCP Header Format



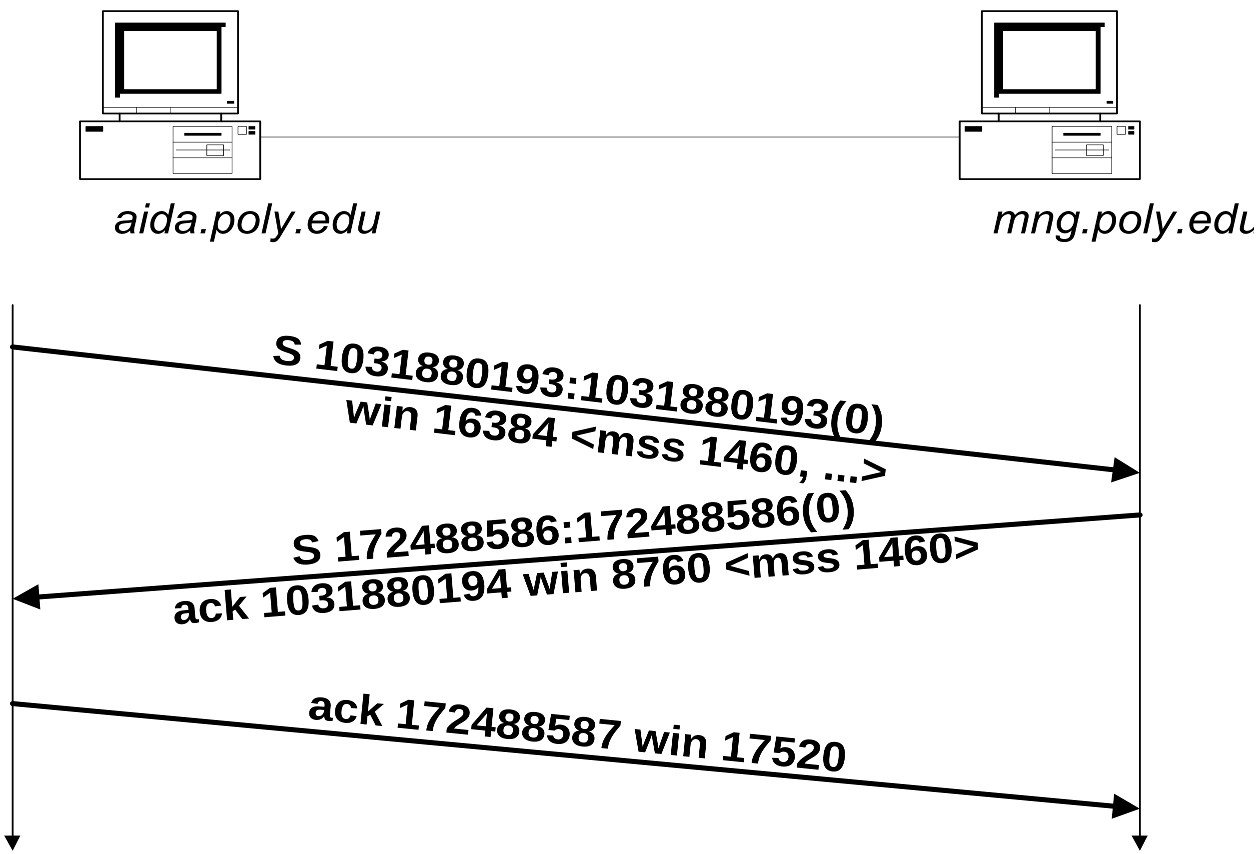
TCP Connection Establishment

Three-way Handshake

- An end host initiates a TCP connection (Active Open) by sending a SYN packet with
  - ISN, *n*, in the sequence number field
  - An empty payload field
  - MSS, and
  - TCP receiving window size
  - SYN flag bit is set.
- The other end replies (Passive Open) a SYNACK packet with
  - ACK=*n+1*
  - Its own ISN, *m*
  - Its own MSS, and
  - Its own TCP receiving window size
- The initiating host sends an acknowledgement: ACK=*m+1*



Three-Way Handshake Example

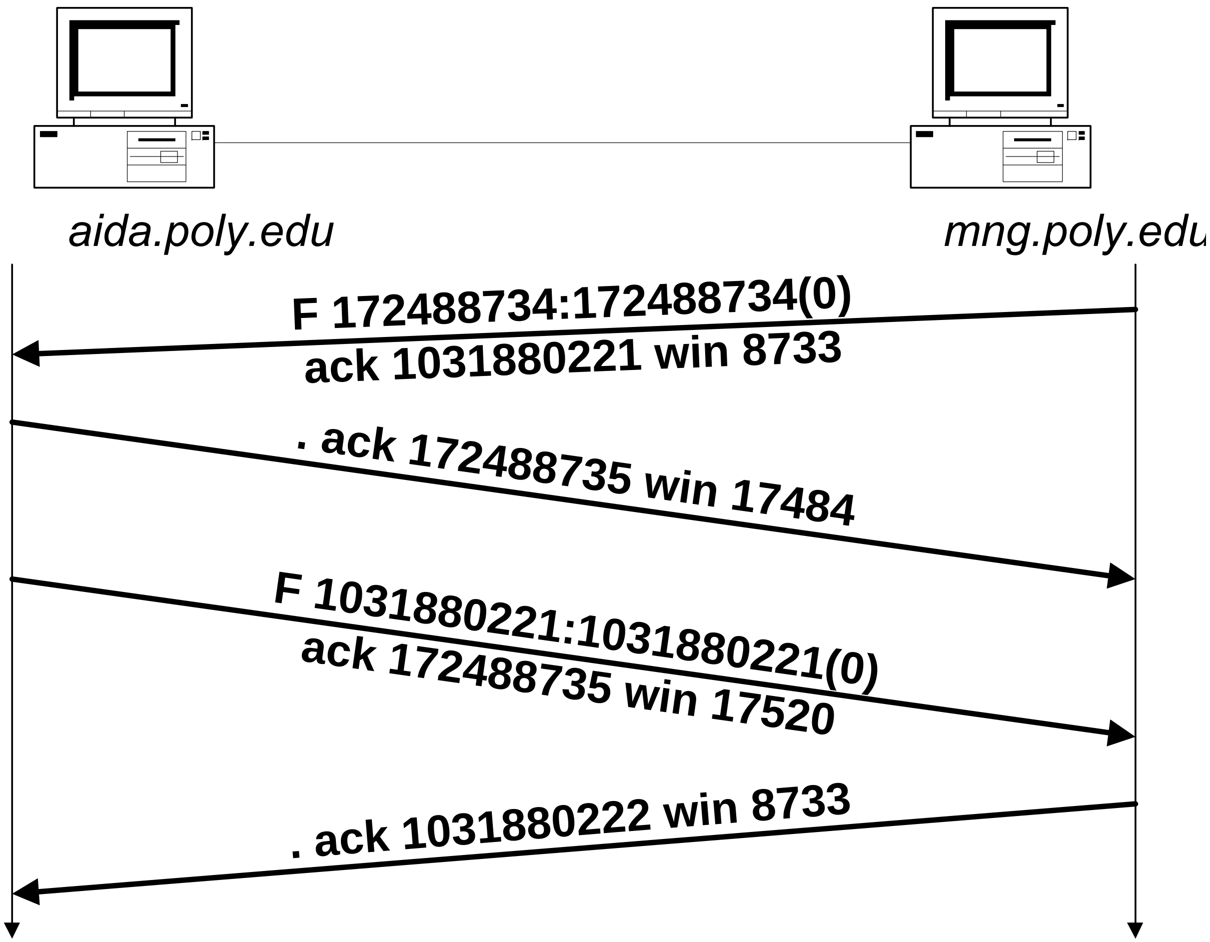


* Note that the data segment following the three-way handshake will start with the sequence number following that of the SYN segment (the first “S” is SYN, the second “S” is SYNACK)

TCP Connection Termination

- Each end of the data flow must be shut down independently (Half-Close)
- If one end is done with data transfer, it sends a FIN segment. This means that no more data will be sent
- Four steps involved:
  - (1) X sends a FIN to Y (Active Close)
  - (2) Y ACKs the FIN,
    - *(at this time: Y can still send data to X, X still has to ACK the data)*
  - (3) and Y sends a FIN to X (Passive Close)
  - (4) X ACKs the FIN … waits 2MSL before closing (Time_Wait)

TCP Connection Termination Example



Interactive and Bulk Data Flow

- TCP applications can be put into the following categories
  - Bulk Data Flow - ftp, mail, http
  - Interactive Data Flow - telnet, rlogin

- TCP has algorithms to deal which each type of applications efficiently.

Interactive Data Transfer Implementation

- Use Delayed Acknowledgement
  - Set delayed ACK timer
  - ACK transmission may be delayed up to 200 ms
- Enable Nagle's Algorithm
  - "Each TCP connection can have only one small segment (less than MSS) outstanding that has not been acknowledged" → Stop & Wait for the small segment
  - Nagle's rule reduces the amount of small segments

Bulk Data Transfer Implementation

Flow Control - How to prevent that the sender overruns the receiver with information?

- Maximum Segment Size (MSS)
- Sliding Window
    - Advertised Window Size (awnd)
- Acknowledgement
    - Cumulative in general implementation
    - Selective acknowledgement is an option if two ends negotiate SACK while a TCP connection is being established)
    - NACK (negative ACK) not allowed

Window Management in TCP

- The receiver is returning two parameters to the sender

AckNo	window size (win) a.k.a. awnd
32 bits	16 bits

- The interpretation is:

  I am ready to receive new data with

  SeqNo= AckNo, AckNo+1, …., AckNo+Win-1

- Receiver can acknowledge data without opening the window
- Receiver can change the window size without acknowledging data
- The sender rate is impacted by
  - The advertised window, and
  - How quickly a segment is acknowledged (to slide the window)
- Congestion can still occur.

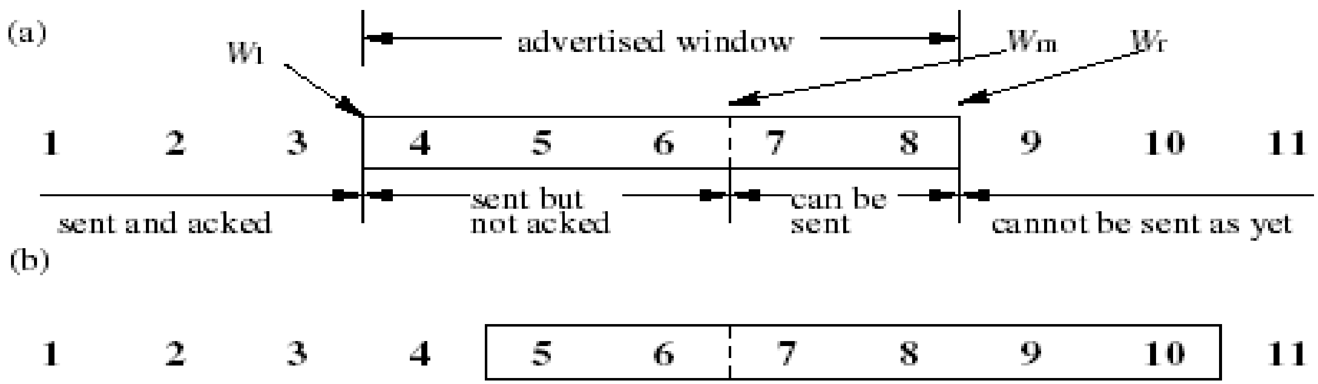
TCP Sliding Window Flow Control

The receiver notifies the sender

- The next segment it expects to receive (AckNo)
- The amount of data it can receive (win)

The sliding window

- W_l moves forward (to the right) when a new segment is acknowledged.
- W_m moves forward when new segments are sent.
- W_r moves
  - Forward (to the right when a larger window is advertised by the receiver or when new segments are acknowledged,
  - Backward (to the left when a smaller window is advertised.



TCP Congestion Control (1/5)

- TCP uses a congestion control to adapt to network congestion and achieve a high throughput.
- Usually the buffer in a router is shared by many TCP connections and other non-TCP data flows.
- TCP needs to adjust its sending rate in reaction to the rate fluctuations of other flows sharing the same buffer.
  - A new TCP connection should increase its rate as quickly as possible to take all the available bandwidth.
  - TCP should slow down its rate increase when the sending rate is higher than some threshold.
- The sender can infer congestion when a retransmission timer goes off.
- The receiver reports “congestion” implicitly by sending duplicate acknowledgements.

TCP Congestion Control (2/5)
– Parameters

- The receiver provides two variables to influence senders transmission rate:
  - advertised Window size (*awnd*)
  - Maximum Segment Size (*MSS*)
- The sender maintains two variables for congestion control:
  - congestion window size (*cwnd*): as the upper bound of the transmission rate.
  - slow start threshold (*ssthresh*)
- The sender uses Allowed Window = min (cwnd, awnd) as the size of the sliding window.

TCP Congestion Control (3/5)
– Slow Start & Congestion Avoidance

- Slow Start and Congestion Avoidance
  1) if *cwnd* ≤ *ssthresh* then /* Slow Start Phase */

     each time an ACK is received:

     cwnd = *cwnd* + *segsize* (= *MSS*)

     else (i.e. *cwnd* > *ssthresh*) /* Congestion Avoidance Phase */

     each time an ACK is received:

     cwnd = *cwnd* + *segsize* x *segsize* / *cwnd* + *segsize / 8*

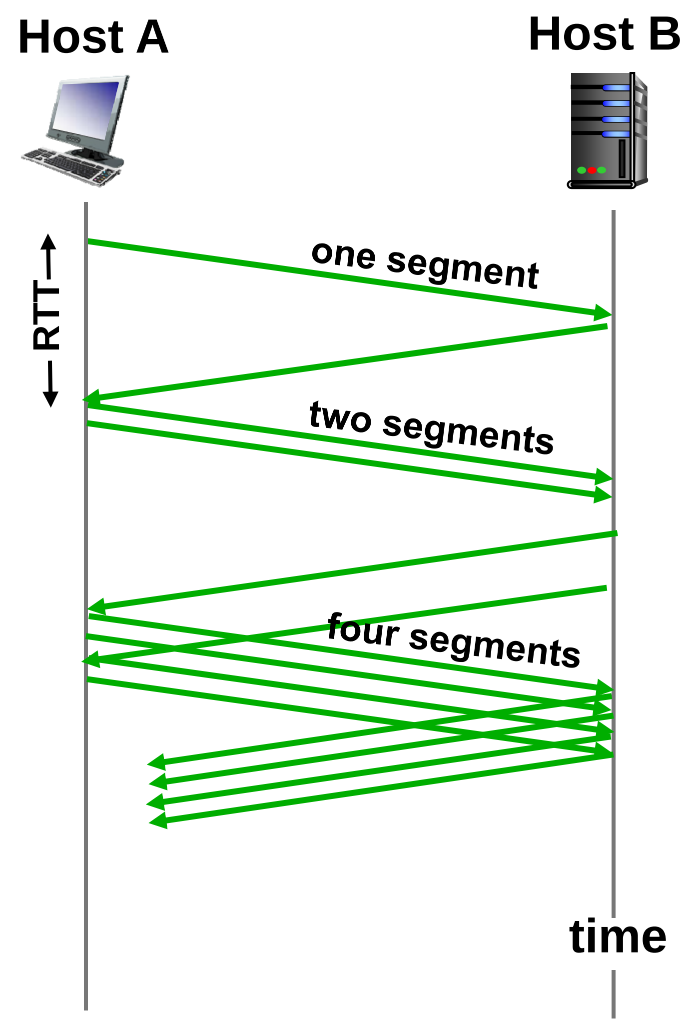
     end
  2) when a congestion occurs (indicated by retransmission timeout), reset

     <u>*ssthresh* = max [2 *segsize*, min (*cwnd*, *awnd*)/2]</u>

     <u>*cwnd* = 1 *segsize*</u> /* back to Slow Start Phase */
- Note:
  - Set *cwnd* = 1 *segsize* (= *1 MSS* bytes) whenever starting traffic on a new connection, or whenever increasing traffic after congestion was experienced.
  - *ssthresh* changes only when a congestion occurs

TCP Slow Start

- when connection begins, increase rate exponentially until first loss event:
  - initially cwnd = 1 MSS
  - double cwnd every RTT
  - done by incrementing cwnd for every ACK received
- Summary: initial rate is slow but ramps up exponentially fast



TCP Congestion Control (4/5)
– Fast Retransmit & Fast Recovery

Fast Retransmit

- After receiving three duplicate acknowledgments, the sender retransmits the segments without waiting for the retransmission timer to expire.
- After the retransmission, congestion avoidance is performed,

Fast Recovery – used when three or more duplicated ACKs are received

1) after the third duplicate ACK is received:

   $\underline{ssthresh = max\ [\ 2\ segsize,\ min\ (cwnd,\ awnd)/2\ ]}$

   retransmit the missing segment, and then

   $\underline{cwnd = ssthresh + 3\ segment}$

2) for each additional duplicate acknowledgement received:

   cwnd = cwnd + segsize

   transmit one segment if allowed by the window size

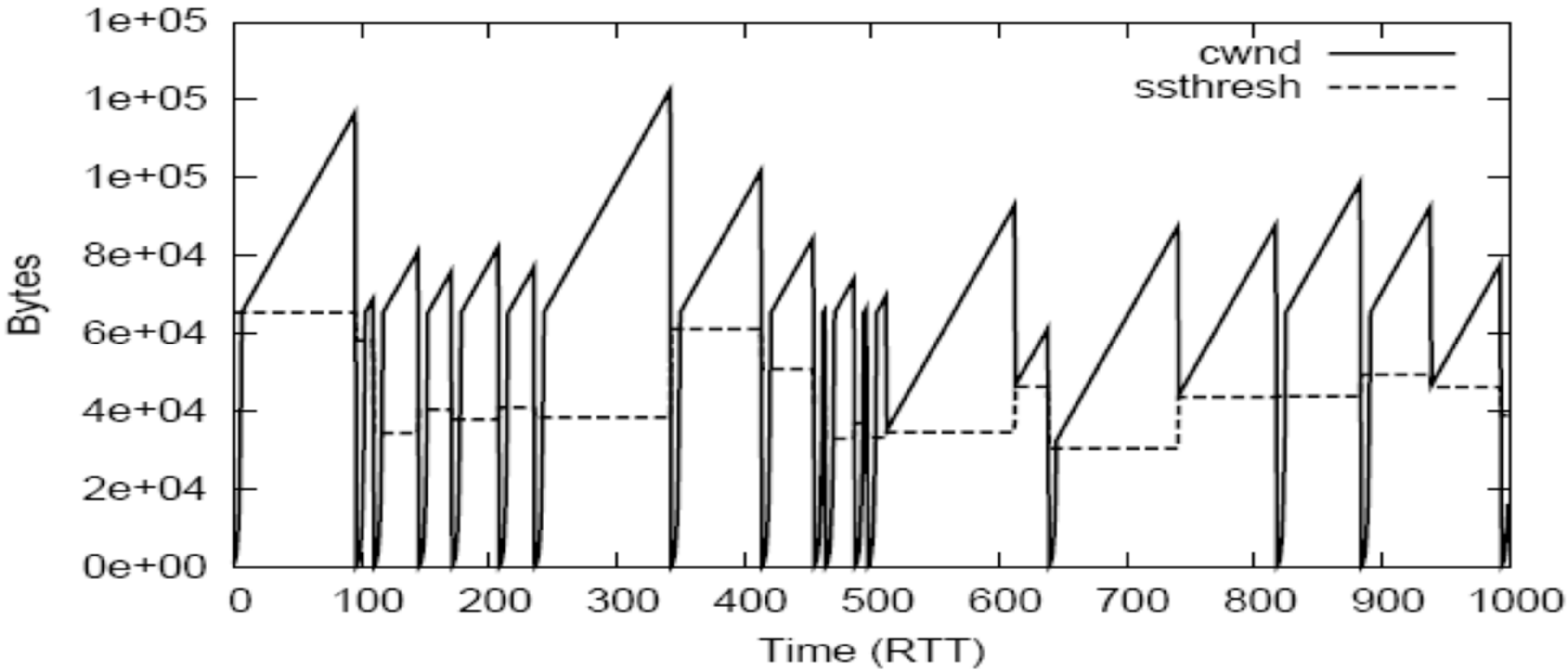
3) when the acknowledgement for the retransmitted segment arrives (new ACK):

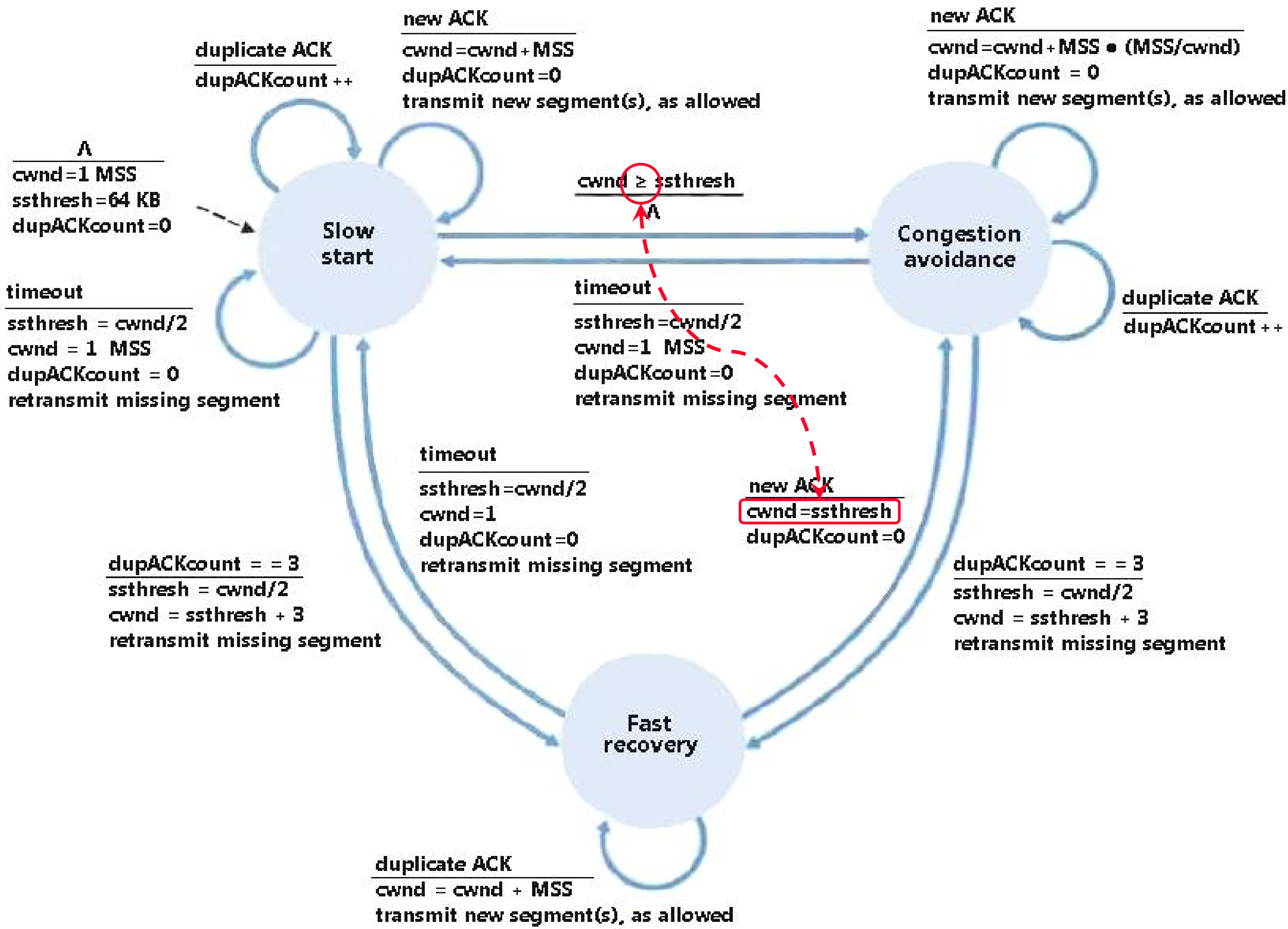
   cwnd = ssthresh + segsize /* Congestion Avoidance Phase */

TCP Congestion Control (5/5)

The evolution of cwnd and ssthresh for a TCP connection, including

- Slow start and Congestion avoidance
  - *cwnd* has two phases: an exponential increase phase and a linear increase phase.
  - *cwnd* drops drastically when there is a packet loss.
- Fast retransmit and fast recovery, occur at time around 610, 740, 950.





duplicate ACK
dupACKcount ++
new ACK
cwnd =cwnd + MSS
dupACKcount =0
transmit new segment(s), as allowed
new ACK
cwnd =cwnd + MSS • (MSS/cwnd)
dupACKcount = 0
transmit new segment(s), as allowed
Λ
cwnd =1 MSS
ssthresh =64 KB
dupACKcount =0
Slow start
cwnd ≥ ssthresh
Λ
Congestion avoidance
timeout
ssthresh = cwnd/2
cwnd = 1 MSS
dupACKcount = 0
retransmit missing segment
timeout
ssthresh =cwnd/2
cwnd =1 MSS
dupACKcount =0
retransmit missing segment
duplicate ACK
dupACKcount ++
timeout
ssthresh =cwnd/2
cwnd =1
dupACKcount =0
retransmit missing segment
new ACK
cwnd =ssthresh
dupACKcount =0
dupACKcount = = 3
ssthresh = cwnd/2
cwnd = ssthresh + 3
retransmit missing segment
dupACKcount = = 3
ssthresh = cwnd/2
cwnd = ssthresh + 3
retransmit missing segment
Fast recovery
duplicate ACK
cwnd = cwnd + MSS
transmit new segment(s), as allowed

Bulk Data Transfer Implementation

Error Control - involving error detection and retransmission of lost or corrupted segments

Retransmission Timer for Automatic Repeat reQuest (ARQ) error control

- Set to a Retransmission Timeout (RTO) value.
- Make RTO adaptively based on RTT – the Round-Trip Time measurement that TCP performs
- Exponential Backoff Algorithm applied in lack of RTT
- Karn’s Algorithm: don’t update RTO on any segments that have been retransmitted

RTT Measurement

- The time difference between sending a target segment and receiving the ACK for the segment is measured.
  - TCP sender of each connection only sets one segment at a time in delay measurement
- Each measured delay is one RTT Measurement, denoted by *M*.
- Compute the RTO per RFC 2988:
  - RTT^s: smoothed RTT, set to the first measured RTT as $RTT^s_0 = M_0$.
  - RTT^d: smoothed RTT mean deviation, set initially as $RTT_0^d = M_0/2$
  - The initial value, $RTO_0 = RTT^s_0 + max\{G, 4 \times RTT^d_0\}$, where G is the timeout interval of the base timer.
  - For the i^{th} measured RTT value M_i:
    - $RTT^s_i = (1 - \alpha) \times RTT^s_{i-1} + \alpha \times M_i$,
    - $RTT^d_i = (1 - \beta) \times RTT^d_{i-1} + \beta \times |M_i - RTT^s_{i-1}|$,
    - $RTO_i = RTT^s_i + max\{G, 4 \times RTT^d_i\}$, where $\alpha=1/8$, $\beta=1/4$.
  - If RTO is less than 1 sec, round up to 1 sec. RTO may be capped (at least 60 sec.)

File Transfer: FTP vs. TFTP

FTP

- Complex but reliable file transfer use TCP
- Specified in RFC 959, well-known port 21 (control) and 20 (data)
- Data retransmission carried in lower layer by TCP
- Used for general purpose, high throughput applications
- Security feature provided
  - Username and password checking
  - Data transfer may fail when address translation/firewall implemented with random port passing

TFTP

- Simple and quick file transfer over UDP
- Specified in RFC 1350, well-known UDP port 69 (for originating request to server)
- Both ends use a timeout retransmission to resend a block of data
- Often used to
  - Load into a batch file for multiple hosts
  - Bootstrap diskless systems
- No username and password checking; constitutes a security hole (… serving the Bootstrap case well).

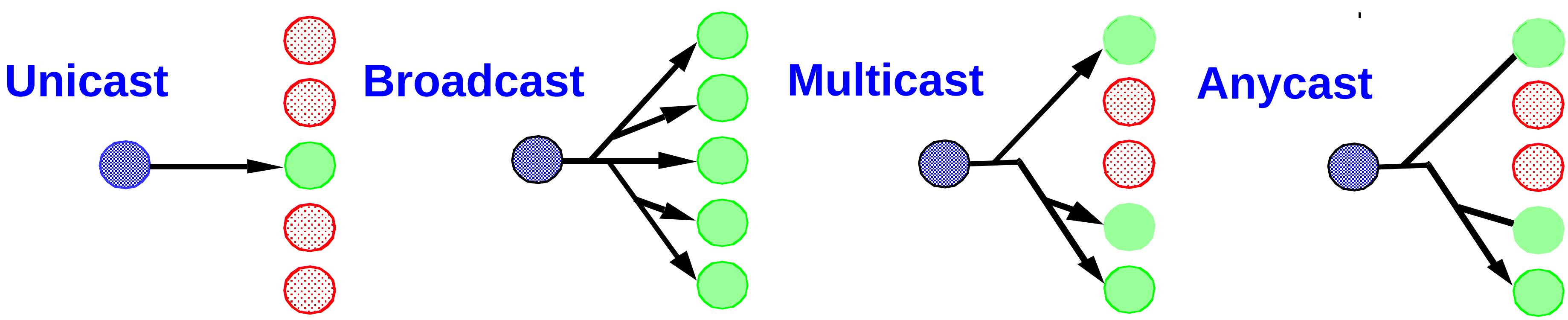
Multicast

& Realtime Service Support

- Multicast Addressing
- Internet Group Management Protocol (IGMP)
- Multicast Routing Protocols
- Realtime Streaming and Its Supporting Protocols

Multicast

- Multicasting is one-to-many or many-to-many communications.
- A simple implementation of multicasting can be built on top of the unicast (point to point) service …
  - Each multicast source send N-1 copies for total N members in the multicast group that leads to an inefficient N^2 problem
  - The desired case: a packet should be transmitted on one link exactly once (least packet replication in network)
- IP Multicasting uses less network resources.
- IP supports multicasting via the help of IGMP and additional routing protocols.



IP Multicasting Key Components

- Multicast addressing
  - Define a common group address for all nodes in a group.
  - Map a multicast group address to a MAC address.
- Multicast group management
  - The multicast group is dynamic, meaning that users may join and leave the group during the multicast session.
  - A multicast router needs to keep track of the memberships of the multicast groups.
  - A participant may want to know who else is in the group.
- Multicast routing
  - Find and maintain a multicast tree from a participating node to all other nodes in the group.
  - The tree should be updated when
    - The network topology changes, or
    - The group membership changes.

IPv4 Multicast Addressing

- Desired properties of multicasting group addressing
  - Decouple group from group members
  - Dynamic group members for a well-known group
- All Class D addresses are designated as multicast IP addresses

Class D	1	1	1	0	multicast group id
					28 bits

Class	From	To
D	224 .0.0.0	239 .255.255.255

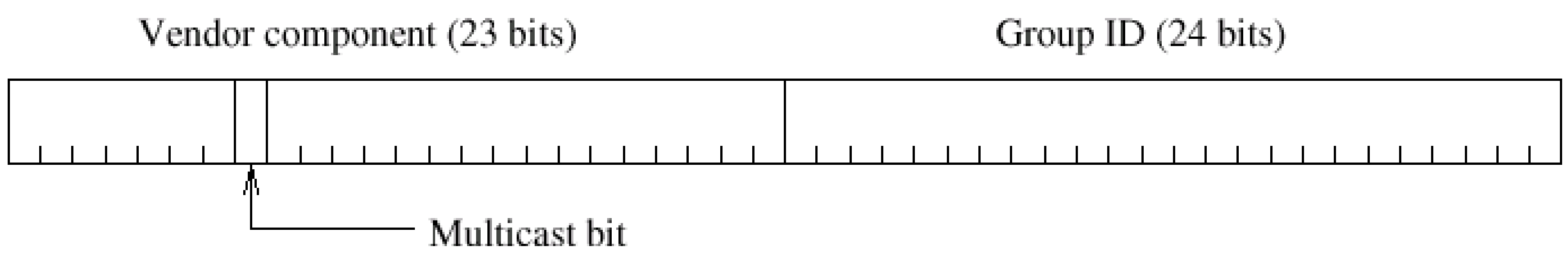
Ethernet Multicast Address

A 48-bit long Ethernet address consists of

- A 23-bit vendor component
- A 24-bit group identifier: assigned by vendor
- A multicast bit: set if the address is an Ethernet multicast address.

An example

- The vendor component of Cisco is 0x00-00-0C.
- A multicast Ethernet address used by Cisco made hardware starts with 0x01-00-0C.



Ethernet Multicast Address (cont'd)

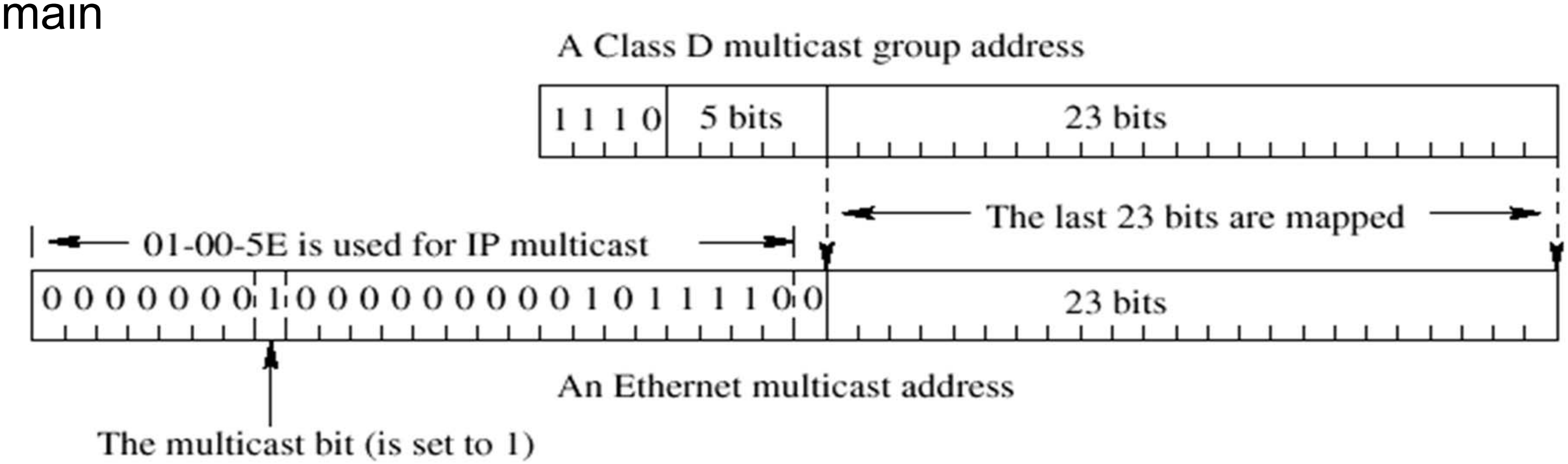
- Ethernet frames with a value of 1 in the least-significant bit of the first octet are flooded
- Ethernet switch generally does not distinguish between multicast and broadcast frames
- Some multicast Ethernet frames may be treated differently, e.g.
  - Dropped by a filter to reduce CPU load
  - Processed when encapsulated with layer-2 control protocol messages

Some well known Ethernet multicast addresses:

Address	Type Field	Usage
FF-FF-FF-FF-FF-FF	**Various**	**Broadcast**
01-80-C2-00-00-00	**0x0802**	**IEEE 802.1D Spanning Tree Protocol**
01-80-C2-00-00-08	**0x0802**	**IEEE 802.1AD Q-in-Q Spanning Tree Protocol**
01-00-0C-CC-CC-CC	**0x0802**	**Cisco Discovery Protocol (CDP)**
01-00-5E-xx-xx-xx	**0x0800**	**IPv4 Multicast**
33-33-xx-xx-xx-xx	**0x86DD**	**IPv6 Multicast**

Multicast Address Mapping: IP ←→ Ethernet

- Ethernet addresses corresponding to IP multicasting are in the range of 01:00:5e:00:00:00 to 01:00:5e:7f:ff:ff.
- At the sender, a multicast destination IP address is directly mapped to an Ethernet multicast address.
  - No ARP request and reply are needed.
  - Only the last 23 bits of the IP address is mapped into the multicast MAC address.
- Ethernet frames with multicast MAC address are often broadcasted in a layer2 domain



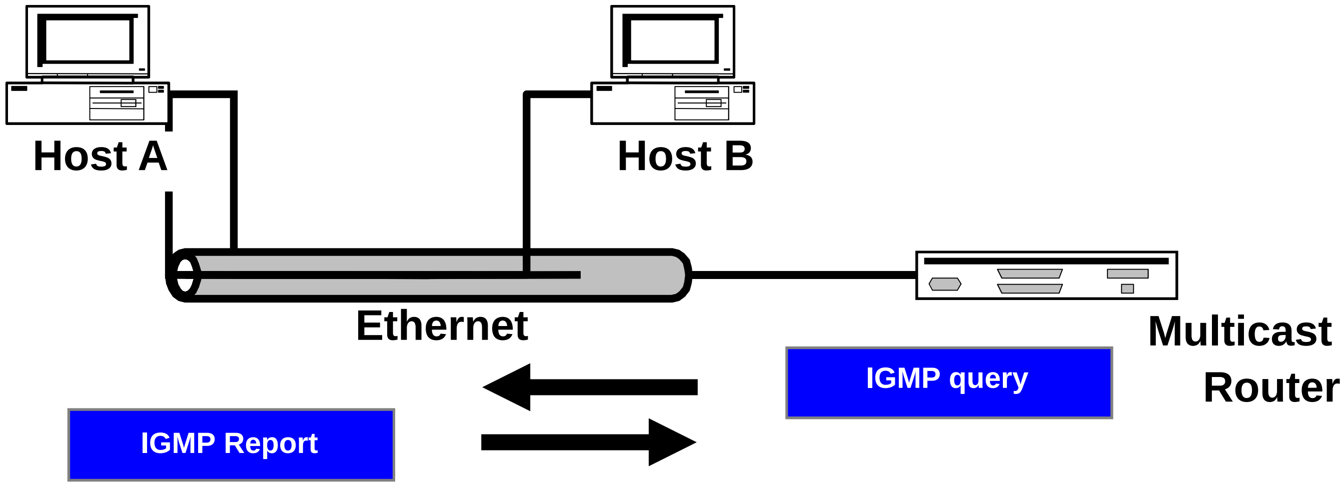
Multicast Address Mapping at the Receiver

A router interface should then be able to receive all the multicast IP datagrams.

At the receiver

- The upper layer protocol should be able to ask the IP module to join or leave a multicast group.
- The IP module maintains a list of group memberships, which is updated when an upper layer process joins or leaves a group.
- The network interface should be able to join or leave a multicast group.
  - When a network interface joins a new group, its reception filters are modified to enable reception of multicast Ethernet frames belonging to the group.

IGMP Multicast Group Management



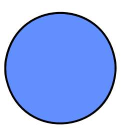
- A host sends an IGMP report when it joins a multicast group
- A host may not send out a report when it leaves a group.
- A multicast router regularly multicasts an IGMP query to all hosts
- A host responds to an IGMP query with an IGMP report for each multicast group to which it is a member.
- Multicast router keeps a table of which of its interfaces have one or more hosts in a multicast group.
- When the router receives a multicast datagram, it forwards the datagram only out the interfaces that still have hosts with processes belonging to that group.
- The router uses a time-out mechanism to discover the empty groups

IP Multicast Routing

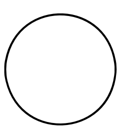
Goal: find a tree of links that connects all routers that have attached hosts belonging to a multicast group

- The participants in a group could be in different geographical locations.
- A host can join and leave the multicast session at will → impact its router's status
- The size of a group could be 1 or larger.

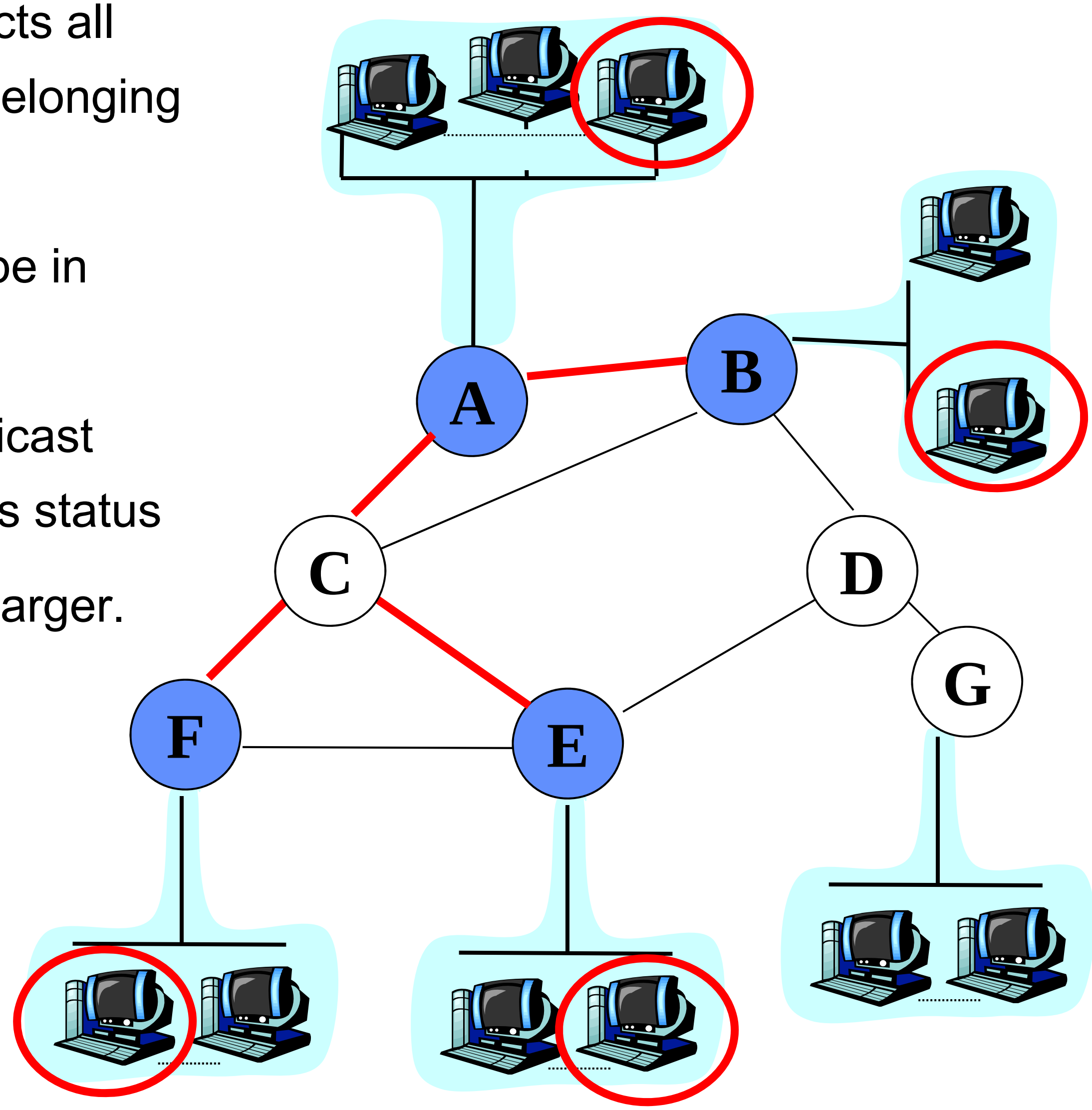
LEGEND



Multicast router with attached group member



Multicast router with no attached group member



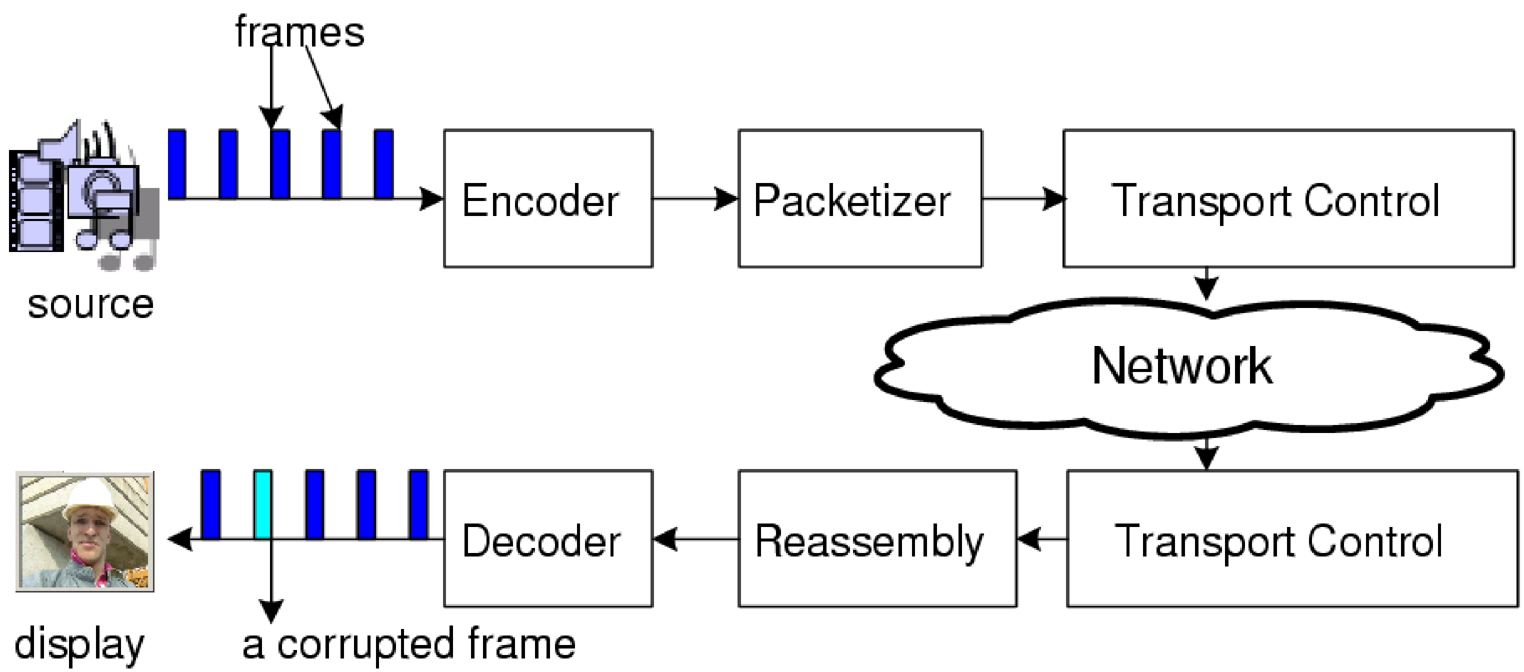
Two types of multicast routing protocols

- Source-tree based protocols
  - Facilitate a more even distribution of the multicast traffic
  - Multicast datagrams from a source are distributed in the shortest path tree, resulting in a better delay performance
  - Each multicast router has to maintain state for all sources in all the multicast groups. Too costly for a large number of multicast sessions.
- Shared-tree based protocols
  - Use a shared tree for all the sources in a multicast group. Greatly reduce the number of states in the routers
  - Has the traffic concentration problem
  - The shared tree may not be optimal for all the sources, resulting in larger delay and jitter.
  - The performance depends on how the Rendezvous Point (RP) is chosen.

Realtime Multimedia Streaming

Realtime multimedia applications

- Video teleconferencing
- Internet Telephony (VoIP)
- Internet audio, video streaming



The Architecture of video streaming

Multimedia Networking Applications

Application Classes:

1) Streaming stored audio and video

2) Streaming live audio and video

3) Real-time interactive audio and video

Fundamental characteristics:

- Typically delay sensitive
  - end-to-end delay
  - delay jitter
- But loss tolerant: infrequent losses cause minor glitches
- Antithesis of data, which are loss intolerant but delay tolerant.

QoS Concerns

TCP/IP protocol suite is not designed to accommodate realtime traffic

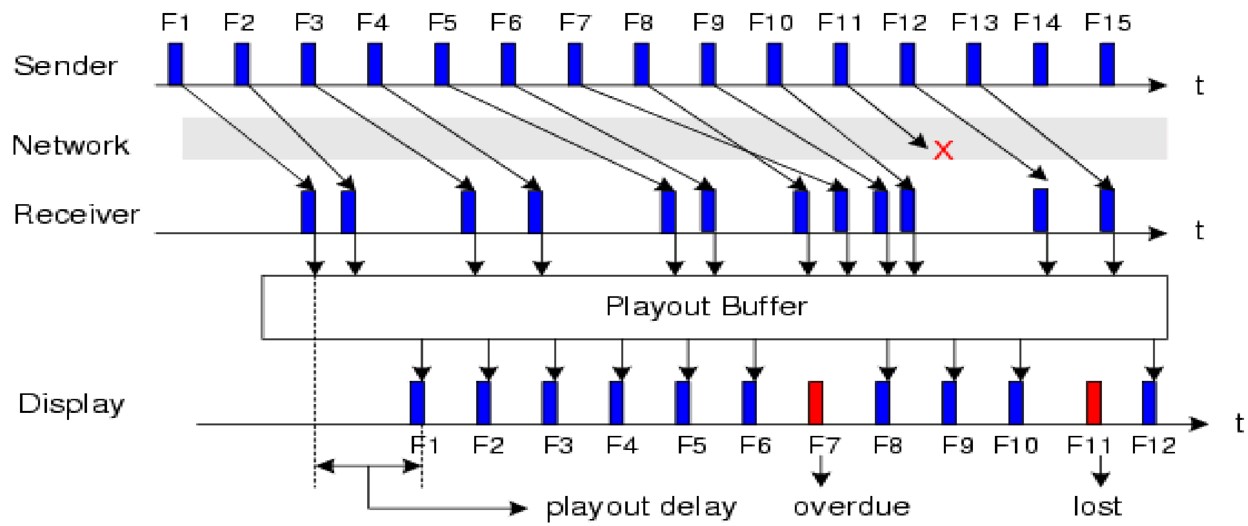
- Lack of support to synchronous, real-time demands
- Traffic loss and variable delays (due to bandwidth limit, non-cooperative network behavior from other data traffic)
- Long call setup time
- Connection-less nature
- Reliability

Jitter Control

Jitter: the variation in the inter-arrival times of received packets

Jitter Control

- Larger playout delay, each frame is due to play at a later time, makes the real time streaming application more tolerable to jitter
- Interactive realtime applications, like VoIP, require tight jitter control due to the strict requirement on end-to-end round trip delay



An example: the playout buffer is used to absorb jitter

Streaming Multimedia: UDP or TCP?

UDP

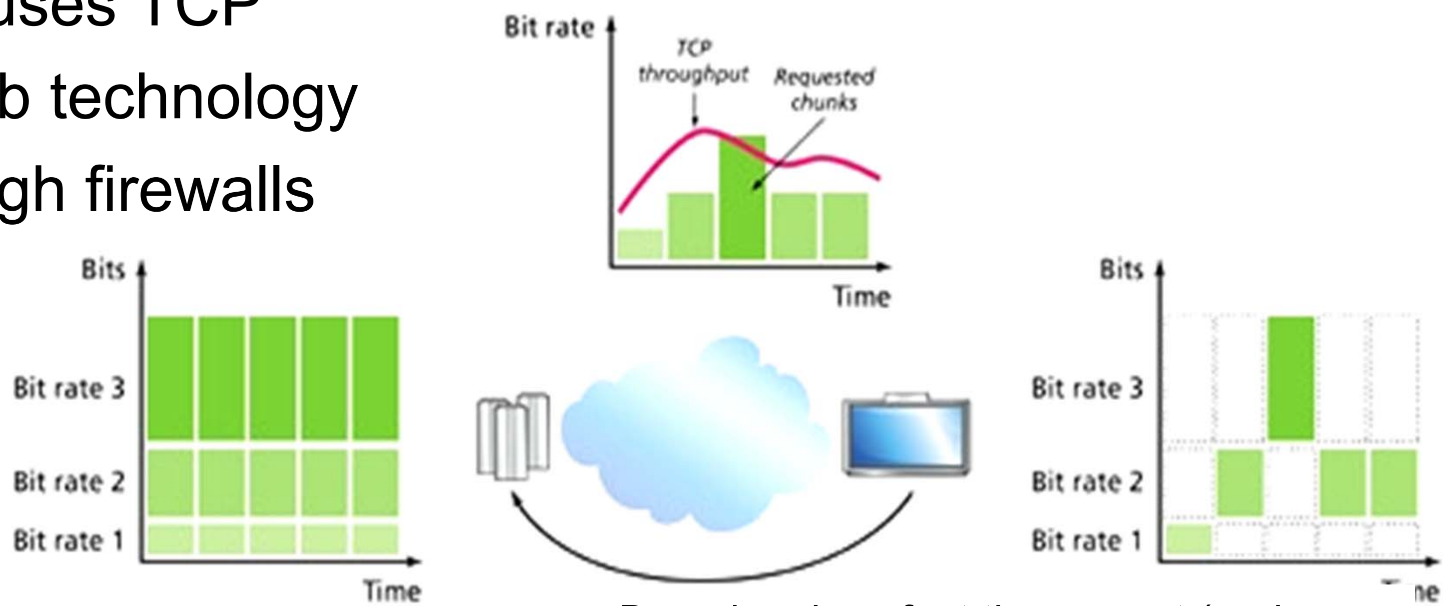
- Server sends at rate appropriate for client (oblivious to network congestion!)
  - often send rate = encoding rate = constant rate
  - then, fill rate = constant rate - packet loss
- Short playout delay (2-5 sec) to compensate for network delay jitter
- Error recover: time permitting
- Usually used for multimedia services

TCP

- Not applicable in multicast!
- Send at maximum possible rate under TCP
- Fill rate fluctuates due to TCP congestion control
- Larger playout delay is intolerable to meet real-time requirements
- HTTP/TCP passes more easily through firewalls
- There are also some advantages to use TCP! HAS example in next slide

HTTP Adaptive Streaming (HAS)

- HAS adapts to available bandwidth
- ISO Standardised: MPEG–DASH
- HAS variants: MS–Silverlight, Apple HLS
- Reliable – uses TCP
- Reuses web technology
- Goes through firewalls



Based on how fast the current (and previous segments) are downloaded, the bit rate of the next segment is selected

More Streaming Performance Requirements

- End-to-end transport control
  - Sequencing – need it in upper layer since UDP does not support sequence numbering
  - Timestamping – for playout, jitter and delay calculation
  - Payload type identification – for media interpretation
  - Error control – need it on upper layer since UDP/IP does not support Forward Error Control (FEC), ARQ, …
  - Error concealment – method to cover up errors from lost packets by using the redundancy in most adjacent-frame image information
  - QoS – from the receiver to the sender for operation adjustment
  - Rate control – from the sender to reduce sending rate adaptively to network congestion
- Network support
  - Bandwidth reservation
  - Call admission and scheduling policy
  - QoS specific routing
  - Traffic shaping and policing

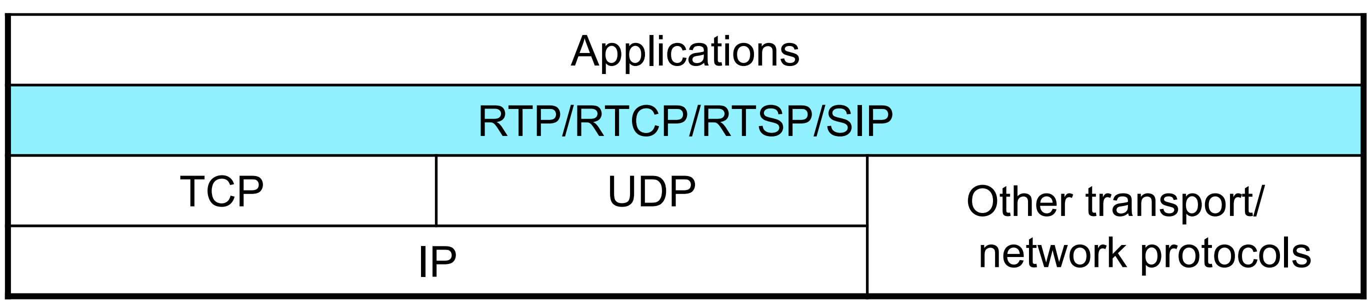
Protocol Stack for Multimedia Services

Application protocols supporting multimedia services:

- Realtime Transport Protocol (RTP)
- Realtime Transport Control Protocol (RTCP)
- Real Time Streaming Protocol (RTSP)
- Session Initiation Protocol (SIP)
  - Basic components: SIP user agent and SIP network server
  - Widely used in IP telephony.

Transport layer protocols

- UDP is usually used for multimedia services
- TCP is not used for a number of reasons
  - The delay and jitter caused by TCP retransmission may be intolerable
  - TCP does not support multicast
  - TCP slow-start may not be suitable for realtime transport.



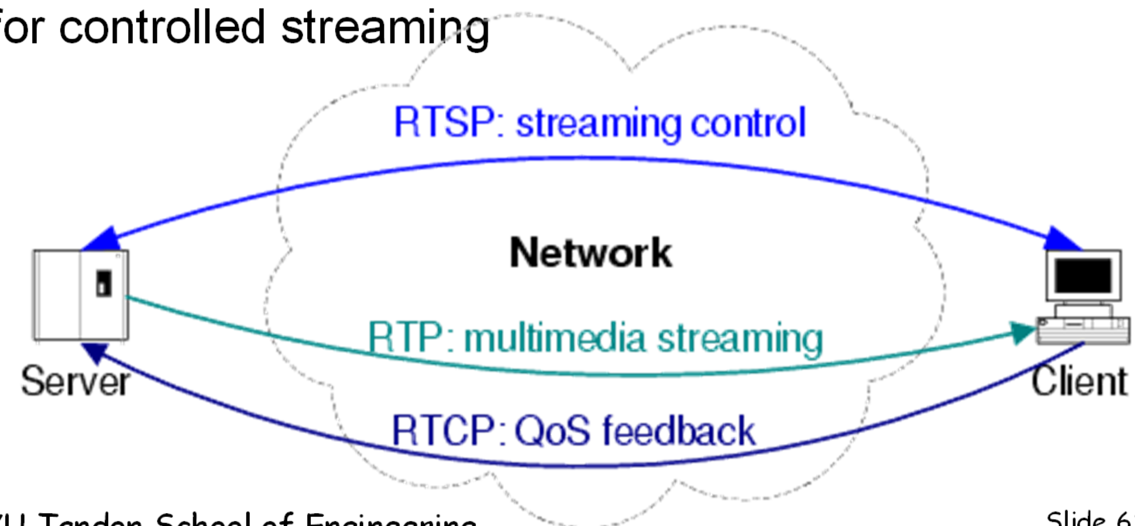
Multimedia Streaming Example

RTCP

- QoS feedback reports containing number of packets lost at receiver, interarrival jitter that allows senders to adjust data rate
- Binding across multiple medias sent by a user (SDES)
- Rate control of RTCP packets by noting how many participants are on session
- Minimal session control

Real Time Streaming Protocol (RTSP)

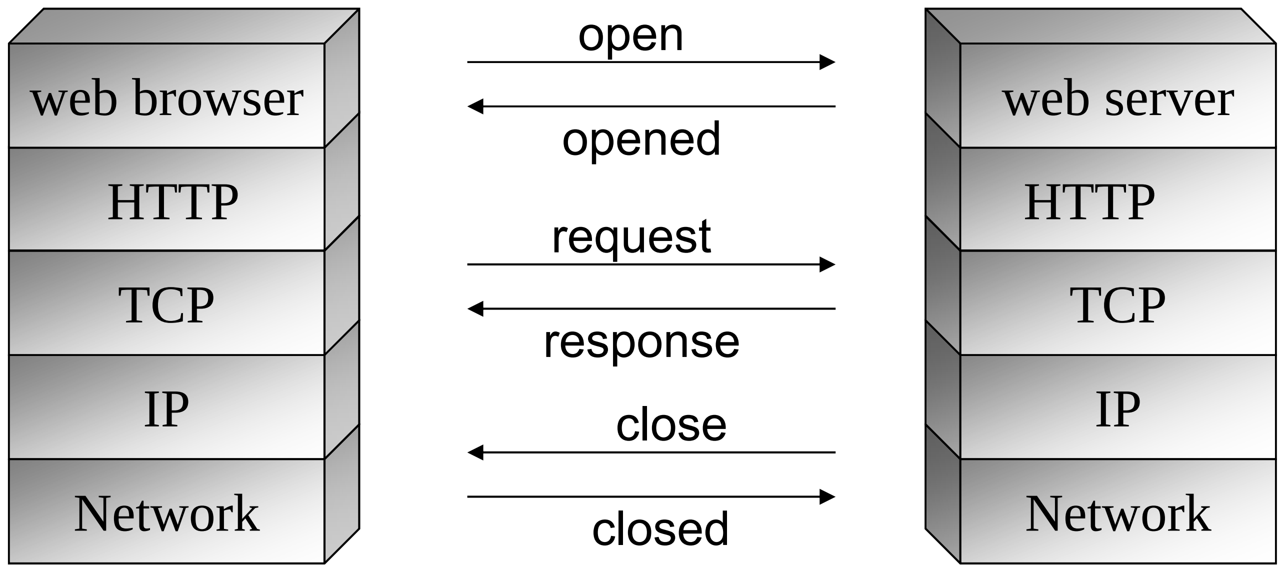
- Internet VCR remote control, initiating and directing realtime streaming
- Transported using UDP or TCP
- Works with RTP/RTCP for controlled streaming



Web, DHCP, NTP, & NAT

- HTML, CGI, HTTP request and response messages
- DHCP, DHCP Transition States
- NTP and network timing service
- Private IP address, NAT, and PAT

HTTP Requests & Responses

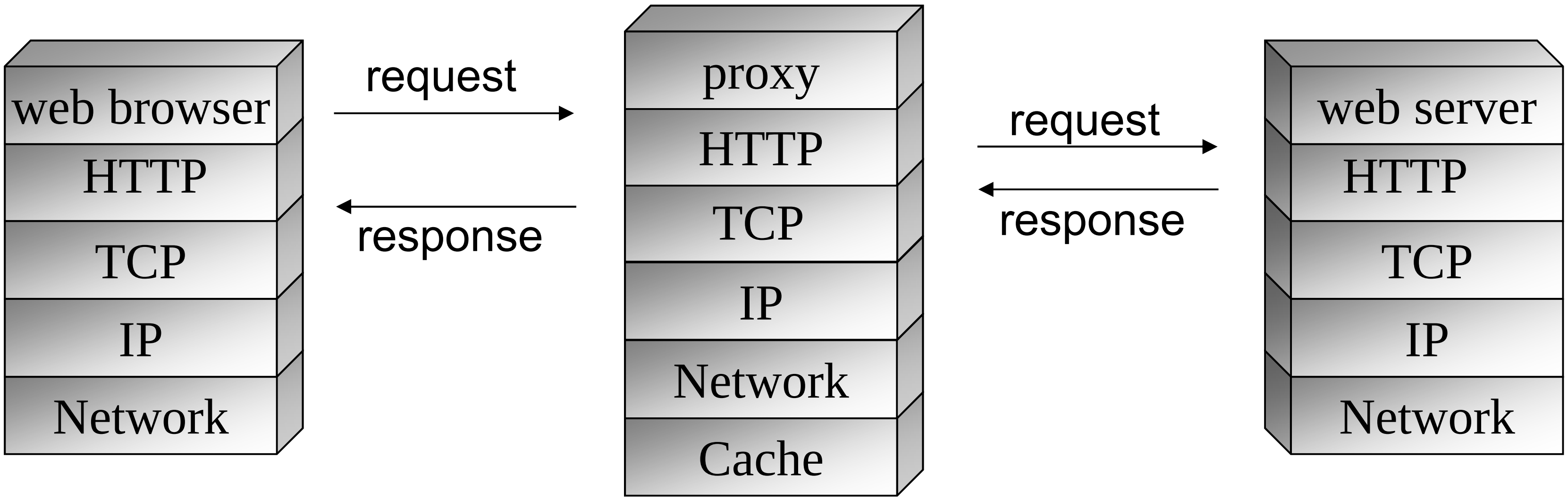


- HTTP has four stages: Open, Request, Response, Close
- A TCP session for HTTP/1.0 does not stay open and wait for multiple requests/responses – not efficient when HTML file has many embedded objects like pictures
- HTTP/1.1 supports persistent connections that allow all the embedded objects sent through the same TCP connection

HTTP TCP Connections

- The client first establishes a TCP connection to the server before an HTTP request
- The server may terminate the TCP connection after the HTTP response is sent
- For embedded objects in a HTML file
  - The client sends a request for each embedded object
  - In HTTP/1.0, the client establishes a TCP connection for each request, not efficient for a file with many embedded objects
  - In HTTP/1.1, persistent connections are supported
    - > All embedded objects are sent through the same TCP connection established for the first request
    - > Both the client and server have to enable the persistent connection feature

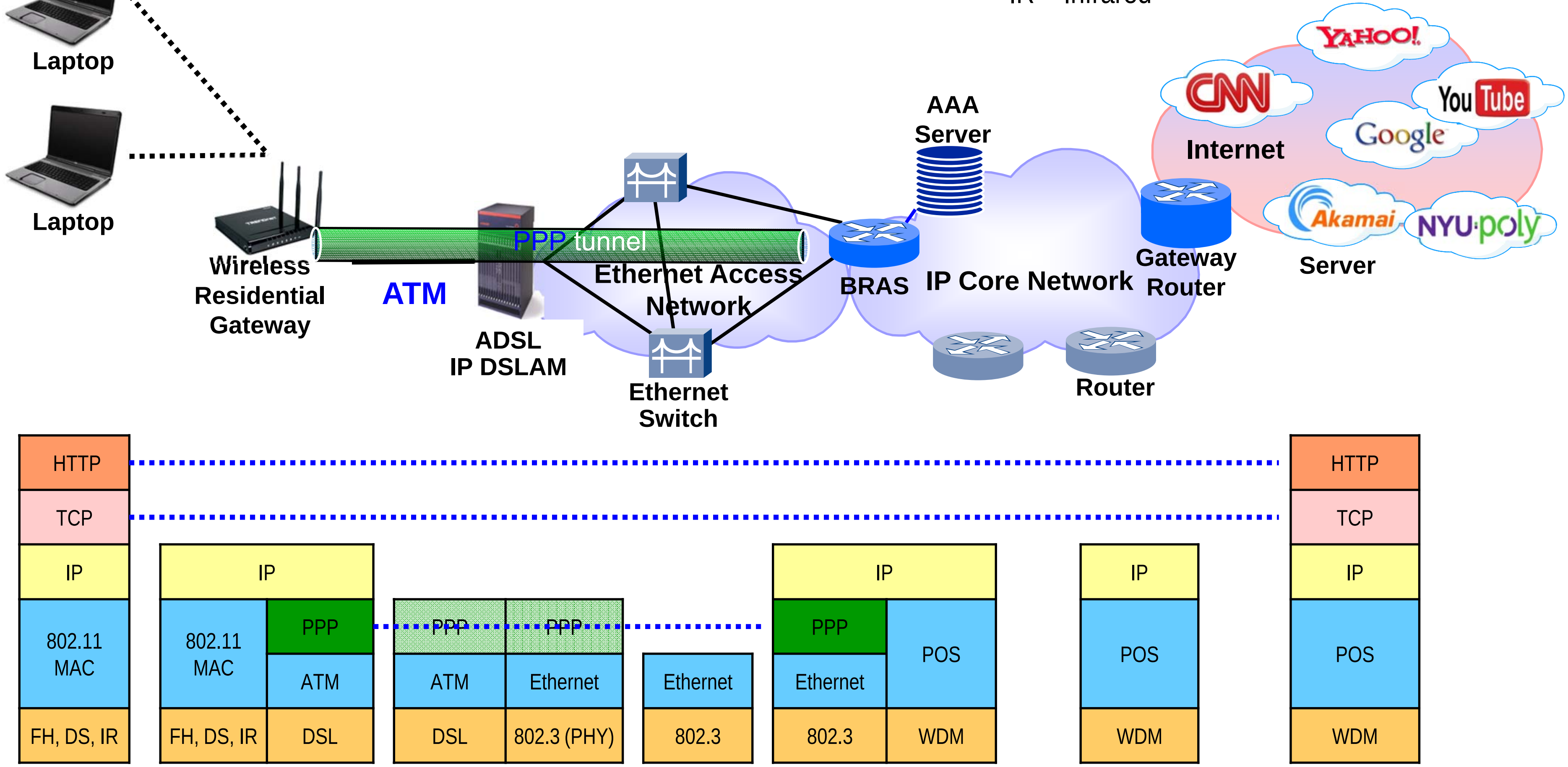
HTTP Proxies



- Proxy server acts as both a client and server
  - receiving client's initial requests, translating requests, passing requests to other servers
- Proxies can be used with firewalls to block undesired traffic
- Cache feature of a Web proxy server reduces network traffic by saving recently viewed pages on the disk driver

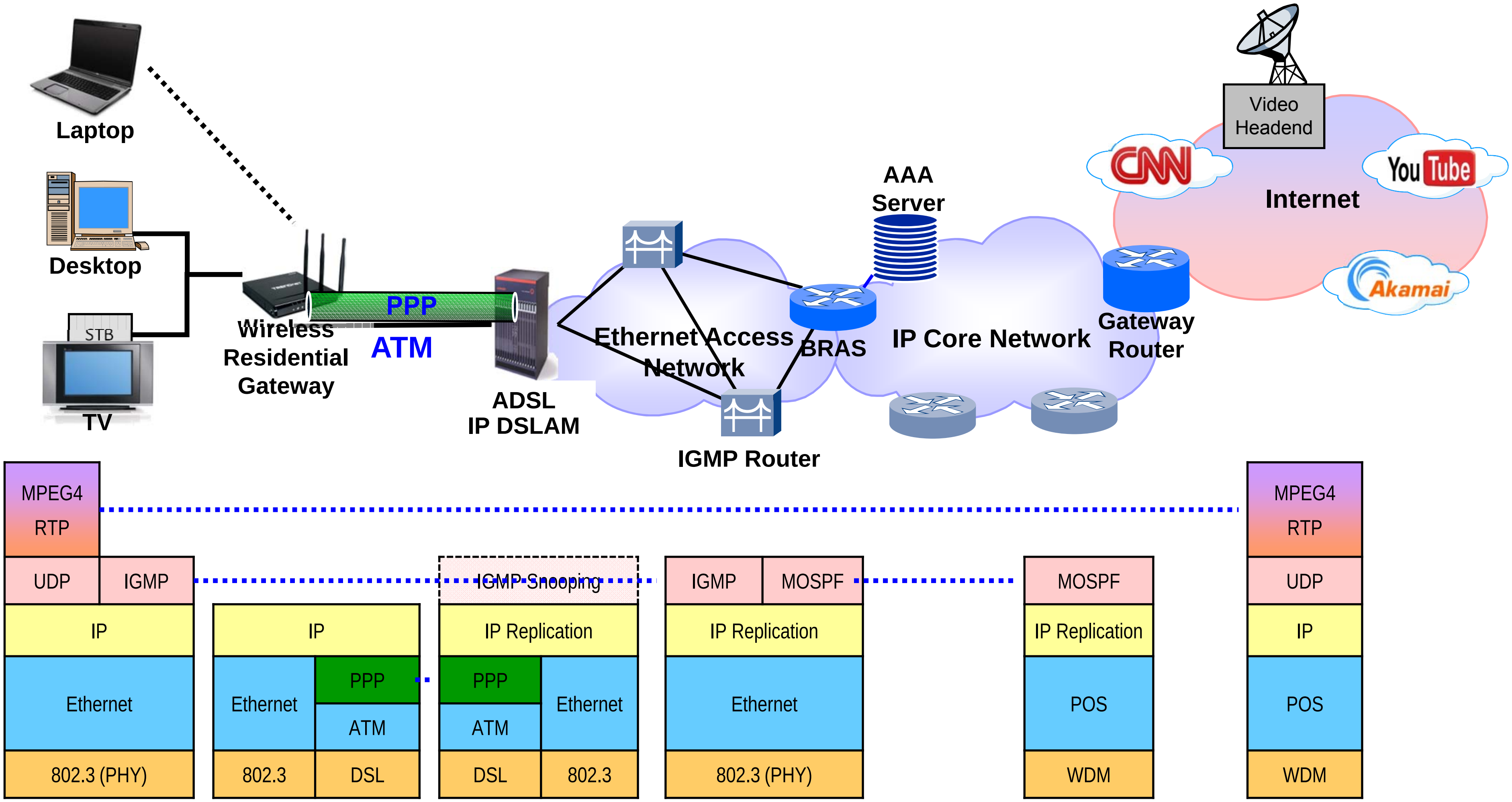
IP Networking Example
- High Speed Internet Browsing

DSL – Digital Subscriber Line
DSLAM – DSL Access Multiplexer
BRAS – Broadband Remote Access Server
WDM – Wavelength Division Multiplexing
POS – Packet Over SDH/SONET
FHSS – Frequency Hopping Spread Spectrum
DSSS – Direct Sequencing Spread Spectrum
IR – Infrared



IP Networking Example
- IPTV Multicasting

DSL – Digital Subscriber Line
DSLAM – DSL Access Multiplexer
WDM – Wavelength Division Multiplexing
POS – Packet Over SDH/SONET
STB – Set Top Box



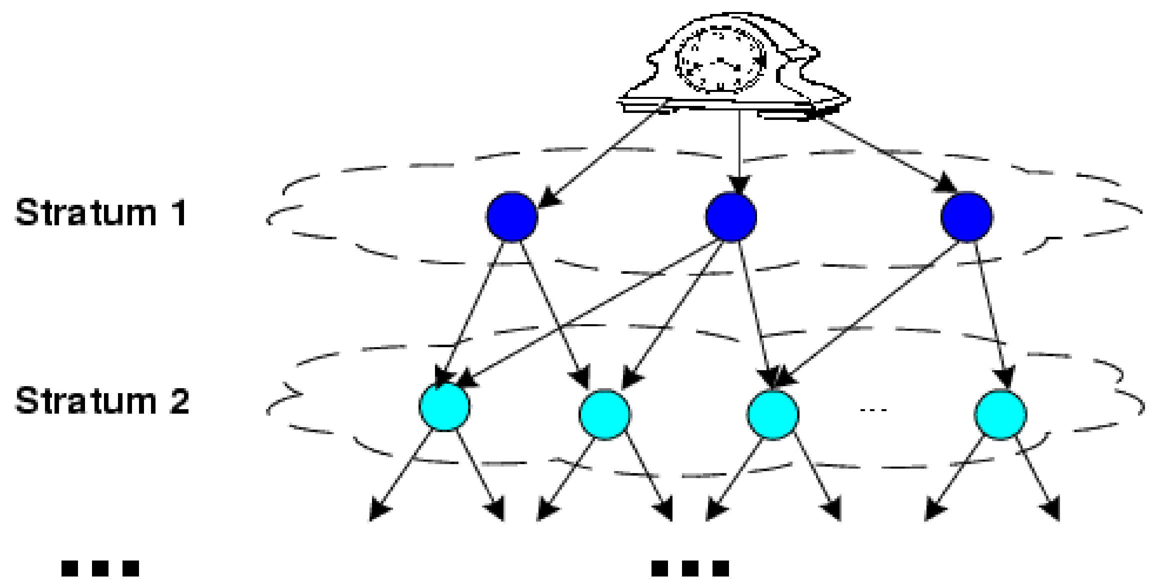
Network Time Protocol (NTP)

- Accurate timing is important in network design, management, security, and diagnosis.
- NTP is an application layer protocol, with UDP or TCP port 123, used to
  - Provide accurate timing in the network
  - Synchronize routers, hosts, and other network devices

NTP Timing Service

NTP timing service uses a hierarchical architecture organized into 16 stratums

- An NTP primary server, or stratum-1, is synchronized with a high precision clock
  - Over 300 valid stratum-1 servers
- About 175,000 hosts running NTP in the Internet, Each server chooses one or more higher stratum servers and synchronizes with them



NTP Operation Modes

Clients and servers can operate in the multicast or broadcast mode.

- Timing information is broadcast or multicast by the servers.
- A client can proactively poll the servers for timing information.

NTP client synchronize with a server in two ways

- Query time information from and synchronize to a remote NTP server, use ***rdate*** or ***ntpdate***
- Synchronize with a remote server continuously and automatically, use ***ntpd***

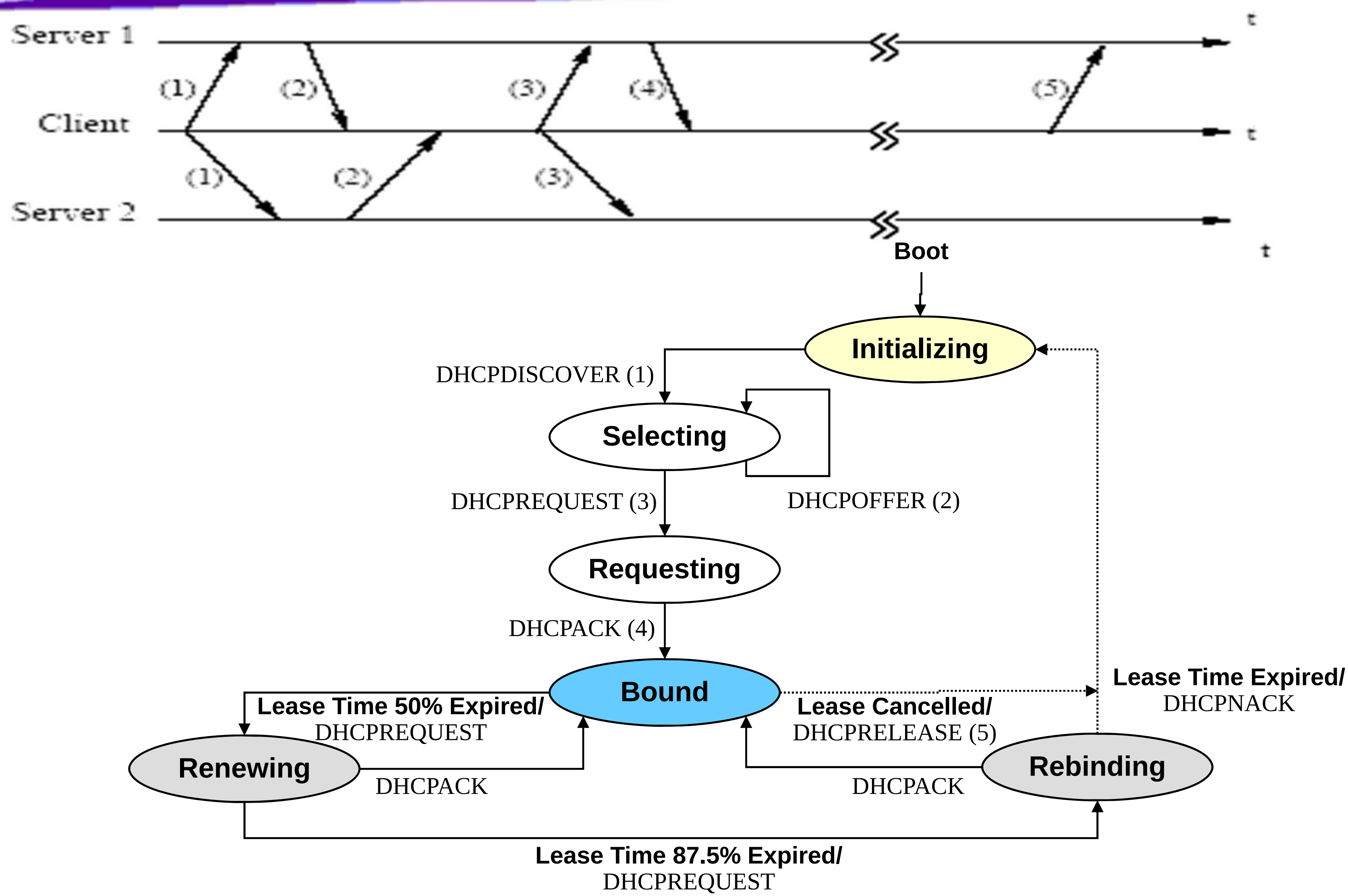
Dynamic Host Configuration Protocol (DHCP)

- DHCP is designed to dynamically configure TCP/IP hosts in a centralized manner from DHCP server.
- DHCP server maintains a collection of configuration parameters, such as IP addresses, subnet mask, default gateway IP address, to make a configured host work in the network.
- A DHCP client queries the server for the configuration parameters.
- The DHCP server returns configuration parameters to the client.
- Often use assigned UDP port numbers for BOOTP (BootStrap Protocol): 67 for DHCP server and 68 for DHCP client

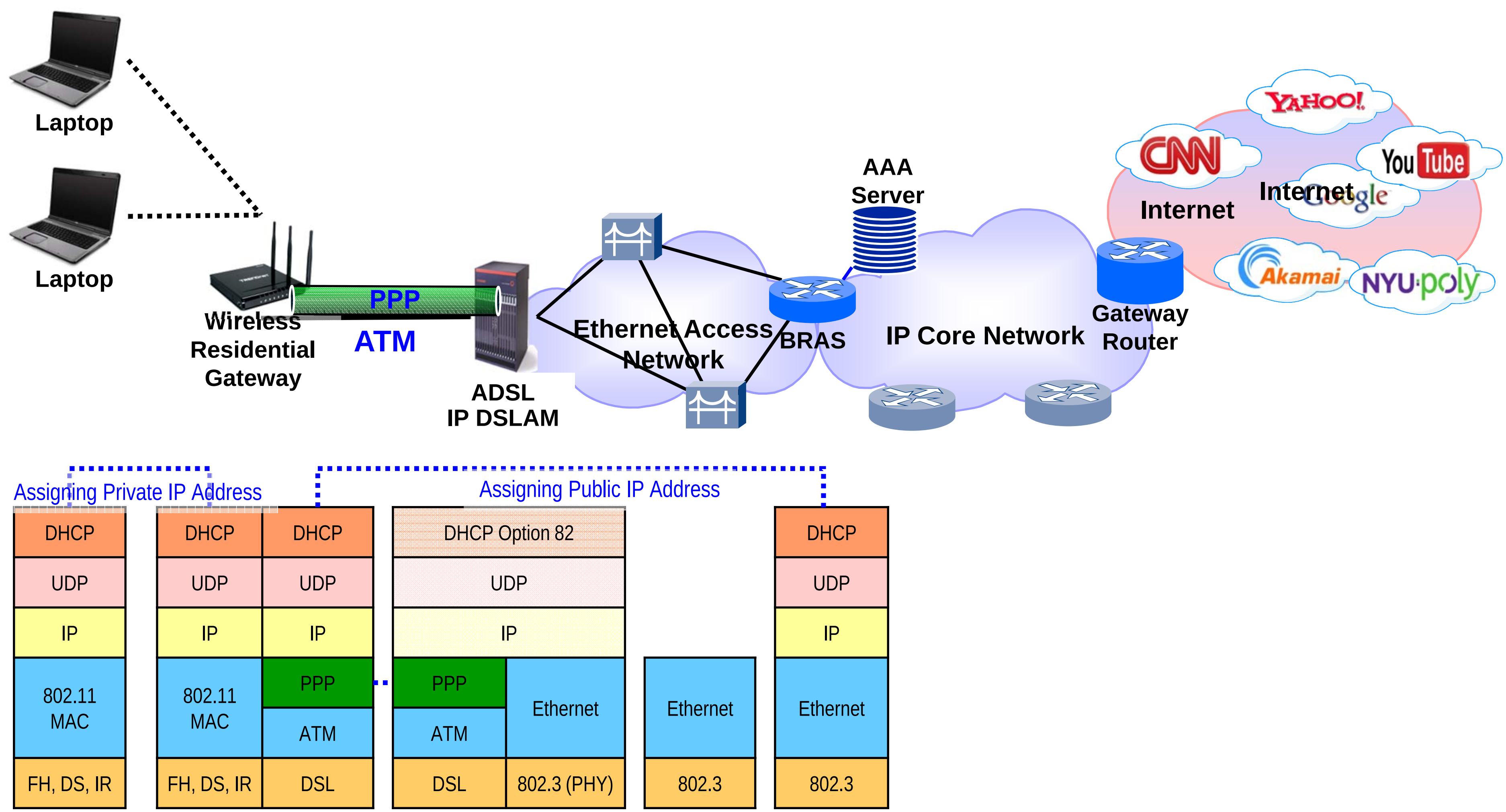
DHCP Network Parameters Assignment

- DHCP can provide persistent storage of network parameters for the clients
  - A client can be assigned with same set of parameters whenever it bootstraps, or is moved to another subnet
  - The DHCP server keeps a key-value entry for each client and uses the entries to match queries from the clients
  - The entry could be a combination of a subnet address and the MAC address (or domain name) of a client
- DHCP can also assign configuration parameters dynamically
  - The DHCP server maintains a pool of parameters and assigns an unused set of parameters to a querying client
  - A DHCP client leases an IP address for a period of time. When the lease expires, the client may renew the lease, or the IP address is put back to the pool for future assignments

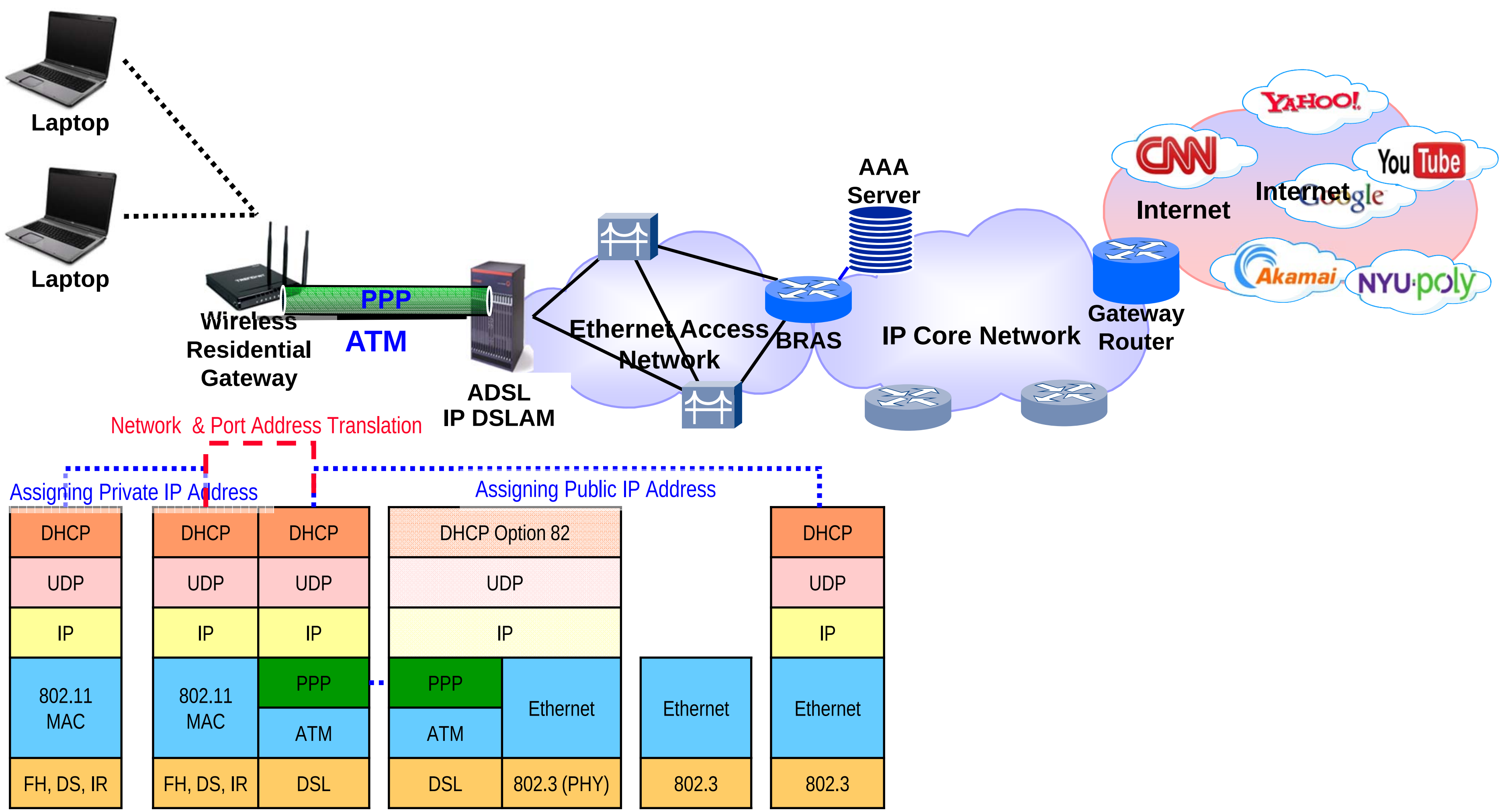
DHCP Client Transition States



IP Networking Example
- Dynamic Host Configuration Protocol



IP Networking Example
- Network Address Translation & Port Address Translation



Private IP Address

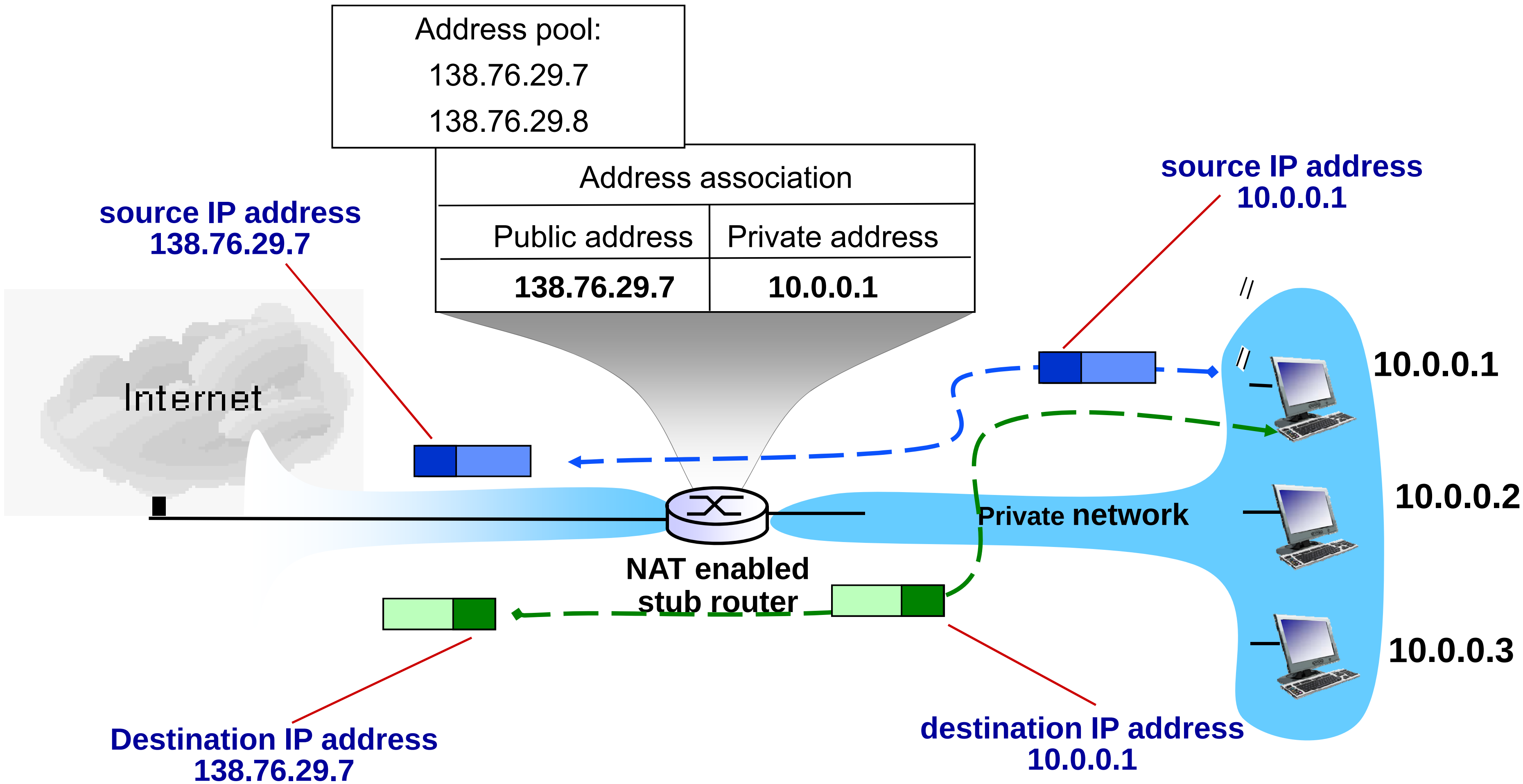
- A *Private Network* is designed to use mainly inside an organization
  - *Intranet* is a private network (LAN) that its access is limited to the users inside the organization
  - *Extranet* is also a private network (LAN) like the intranet but it allows some users outside the organization to access the network
- A number of blocks in each class are assigned for private use
- Private IP addressed are not recognized globally
- Private IP addresses are used either in isolation or in connection with Network Address Translation (NAT) technique

Class	NetID	Block
A	**10.0.0**	**1**
B	**172.16 to 172.31**	**16**
C	**192.168.0 to 192.168.255**	**256**

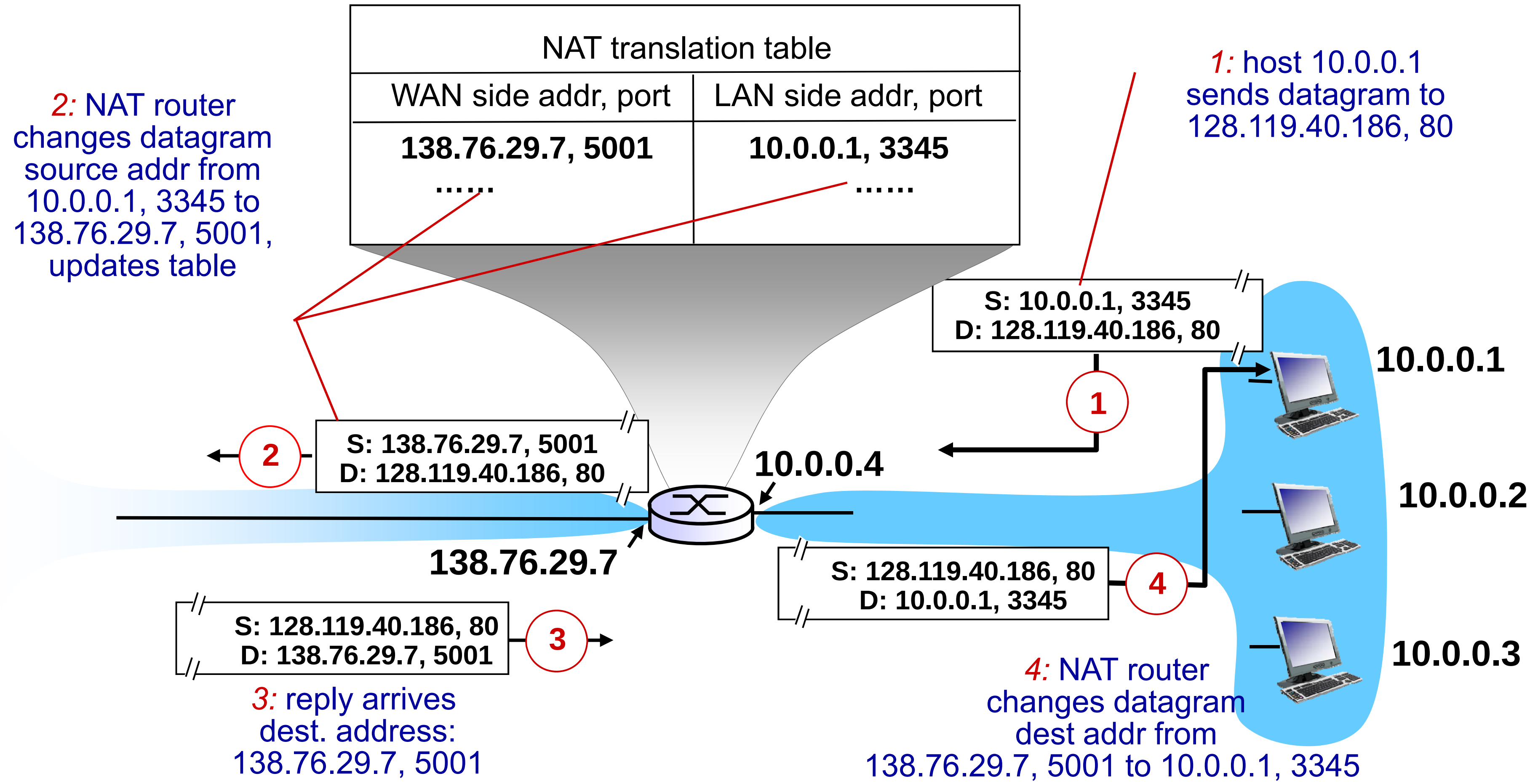
Network Address Translation (NAT)

- NAT is an Internet standard (RFC 1631) that enables a LAN to map between private IP addresses and public IP address
  - Static NAT: one-to-one based mapping between a private address and a public address, e.g. for web server, email server, …
  - Dynamic NAT: mapping a private address to a public one from a pool of public addresses
  - Overloading: a form of dynamic NAT that maps multiple private addresses to a single or a few public addresses by using different ports, a.k.a Port Address Translation (PAT)
- Advantages for NATing:
  - Enables an organization to conserve limited external IP addresses to share by more users
  - Provides a type of firewall by hiding internal IP addresses
  - Supports easy configuration change to access Internet without requiring changes to hosts in the private network

NAT: A Simple Example



NAT: An Example with Single External Address



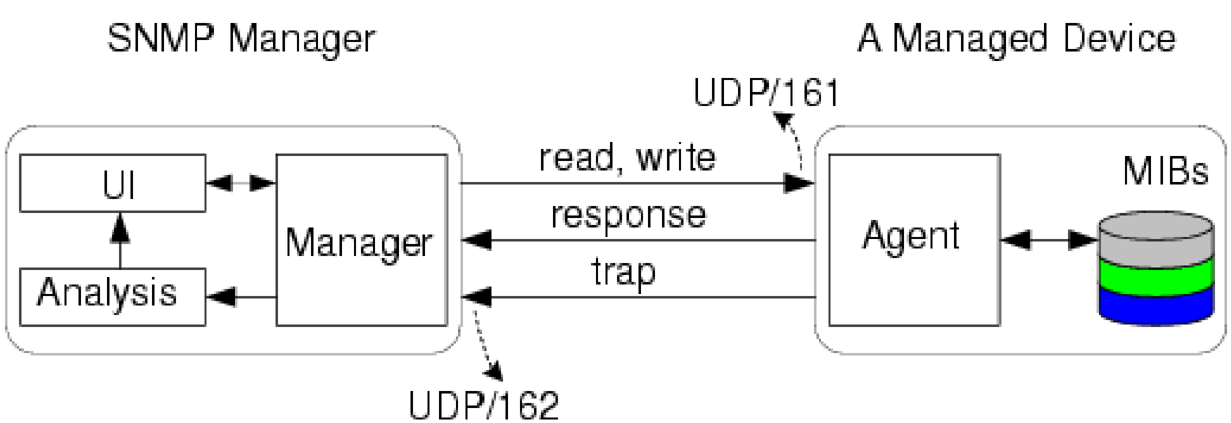
Network Management & Security

- Simple Network Management Protocol
- Network Security Models
- Encryption and Decryption Applications
- IPSec in VPN Network

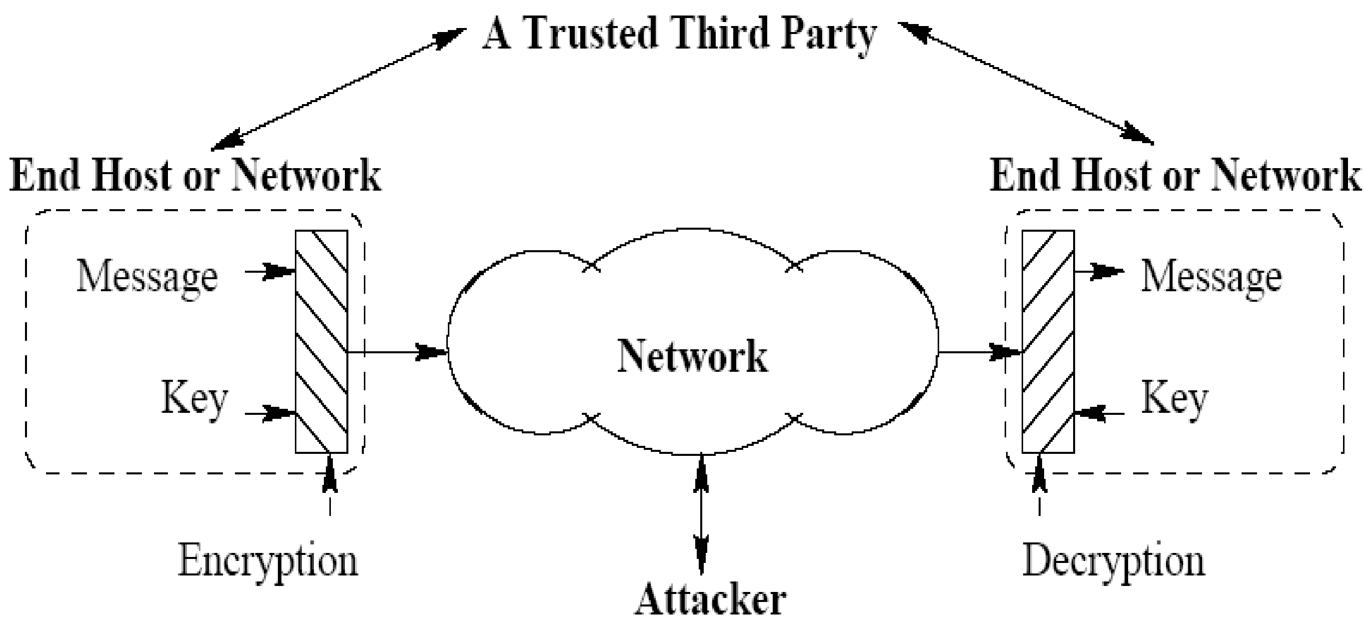
SNMP

Simple Network Management Protocol (SNMP) is an application layer protocol for exchange management information between network devices

- Each Managed Device, a host or a router, maintains a number of Management Information Bases (MIBs)
- Each managed device has an SNMP Agent to provide interface between MIBs and an SNMP Manager
- An SNMP manager, usually implemented in Network Management System, can work with multiple SNMP agents
- Well-known UDP port number 161/162 at SNMP agent/manager



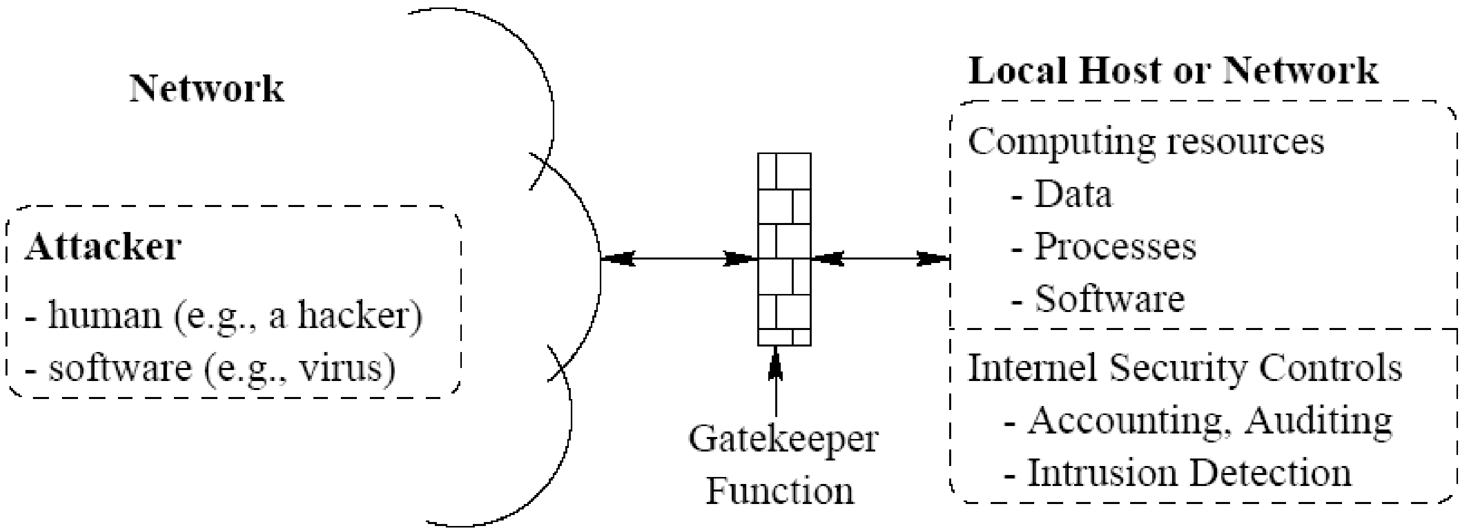
Network Security Model



Security aspects between end host users:

- *Privacy* – a.k.a the expected *Confidentiality* between a data sender and a data receiver
- *Nonrepudiation* – a receiver must be able to prove that received data came from a specific sender; the sender must not be able to deny sending the data
- *Integrity* – the data must be received exactly as it was sent

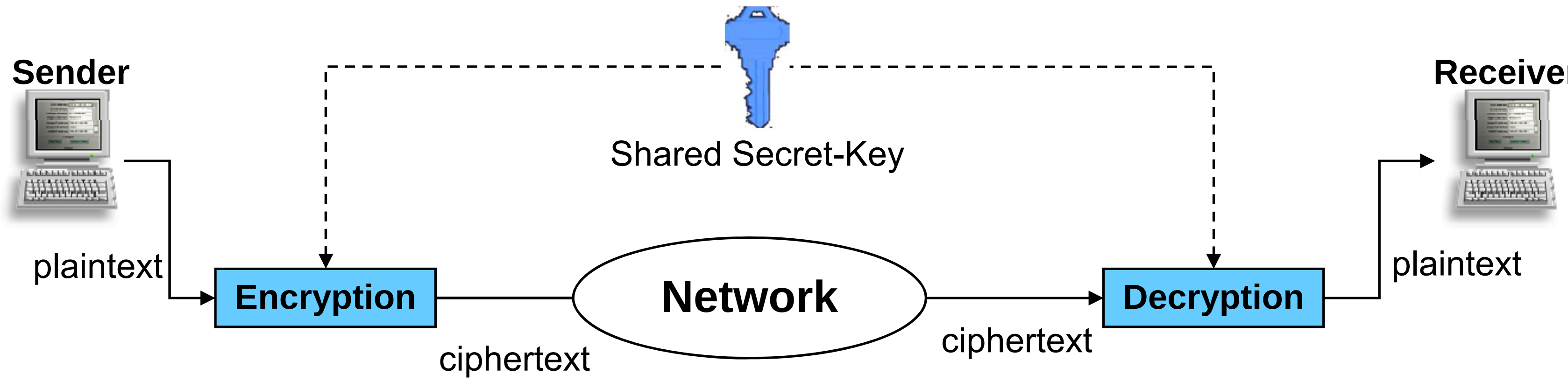
Network Access Security Model



Security aspects for network operators:

- AAA elements in information security
  - *Authentication* to ensure users' identity
  - *Authorization* to assign legitimate privilege to users
  - *Accounting* to logs user/network behavior for security analysis
- Service *Availability* to ensure the accessibility to users
- Network security dimensions: Access Control and Communication Security

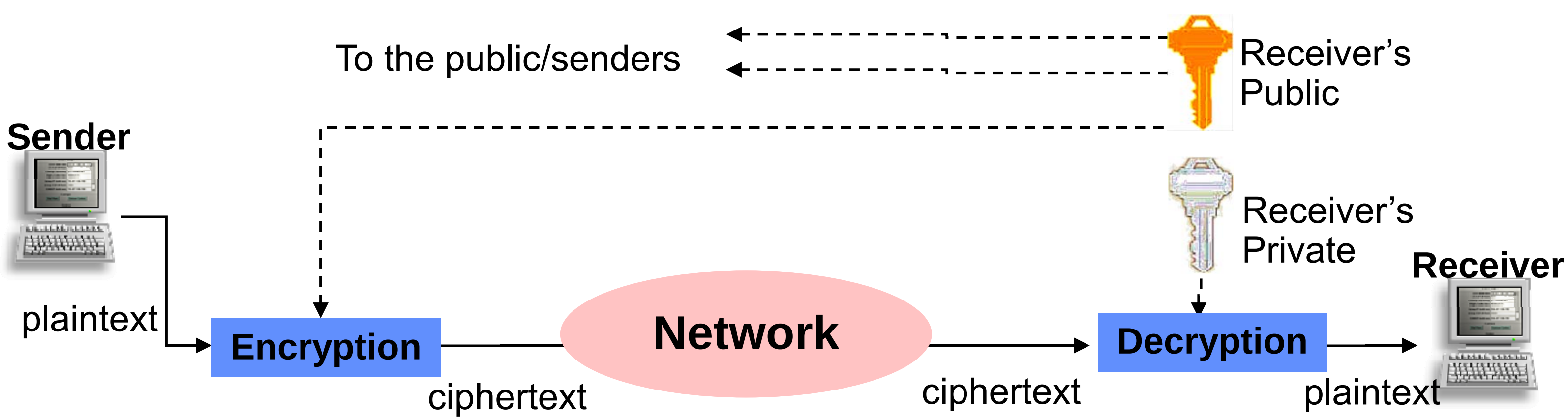
Secret-Key Encryption/Decryption



- Symmetric encryption as the same key shared by both sender and receiver
- The decryption algorithm is the inverse of the algorithm used for encryption
- Advantage
  - Efficient with relative smaller key for long messages
- Disadvantage
  - Too many keys, N(N-1)/2 keys for N users
  - Difficult to distribute shared keys (through trusted third party)

Public-Key Encryption/Decryption

- Two keys for each receiver
  - The public key for message encryption/decryption by a sender
  - The private key for message decryption/encryption by the receiver
- Advantage
  - Easy to distribute public key
  - More scalable with less keys, 2N keys for N users
- Disadvantage
  - Complexity of the algorithm (okay for short messages)
  - Need receiver authentication for the public key



Using Public-Key

To provide authentication

- Bob encrypts a message using his own private key and sends to Alice
- Alice decrypts the received message using Bob's public key
- All other users can decrypt the message since Bob's public key is known
- Alice knows that the message can only be sent by Bob since only Bob knows his own private key

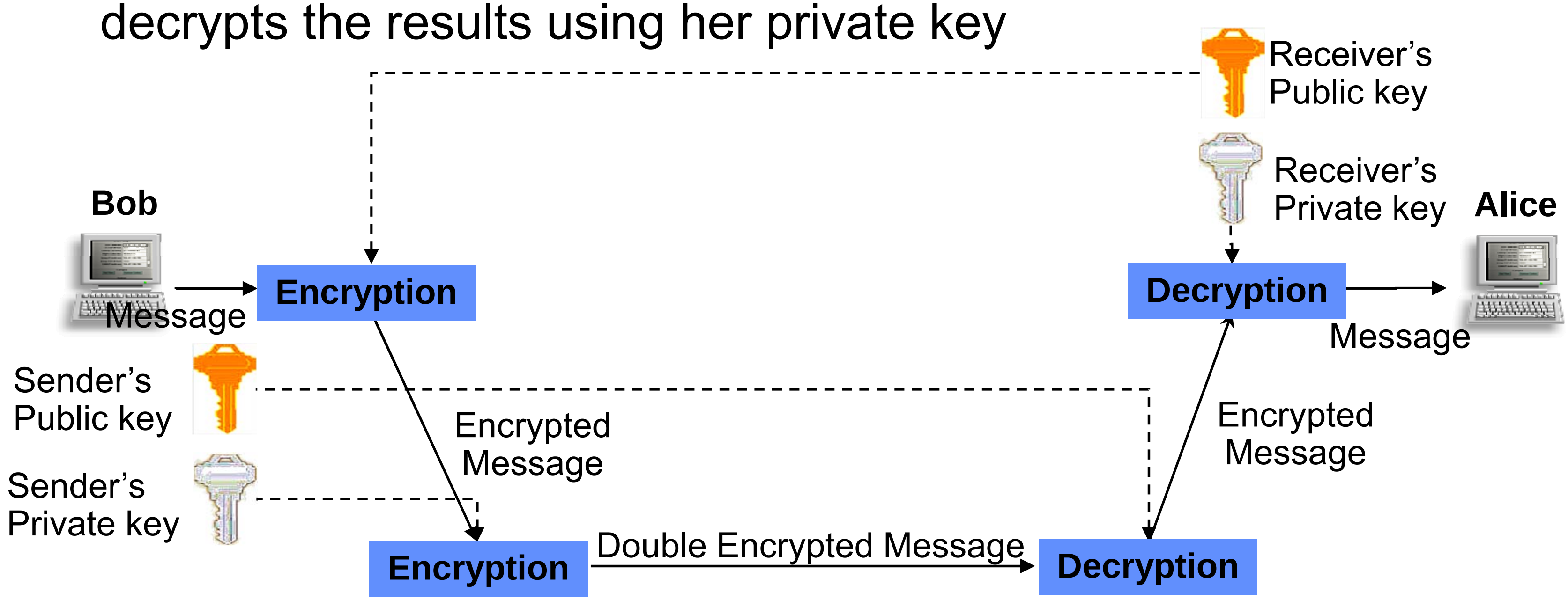
To provide confidentiality

- Bob can encrypt the message using Alice's public key so that other users cannot read the message
- Alice decrypts the received message using her private key

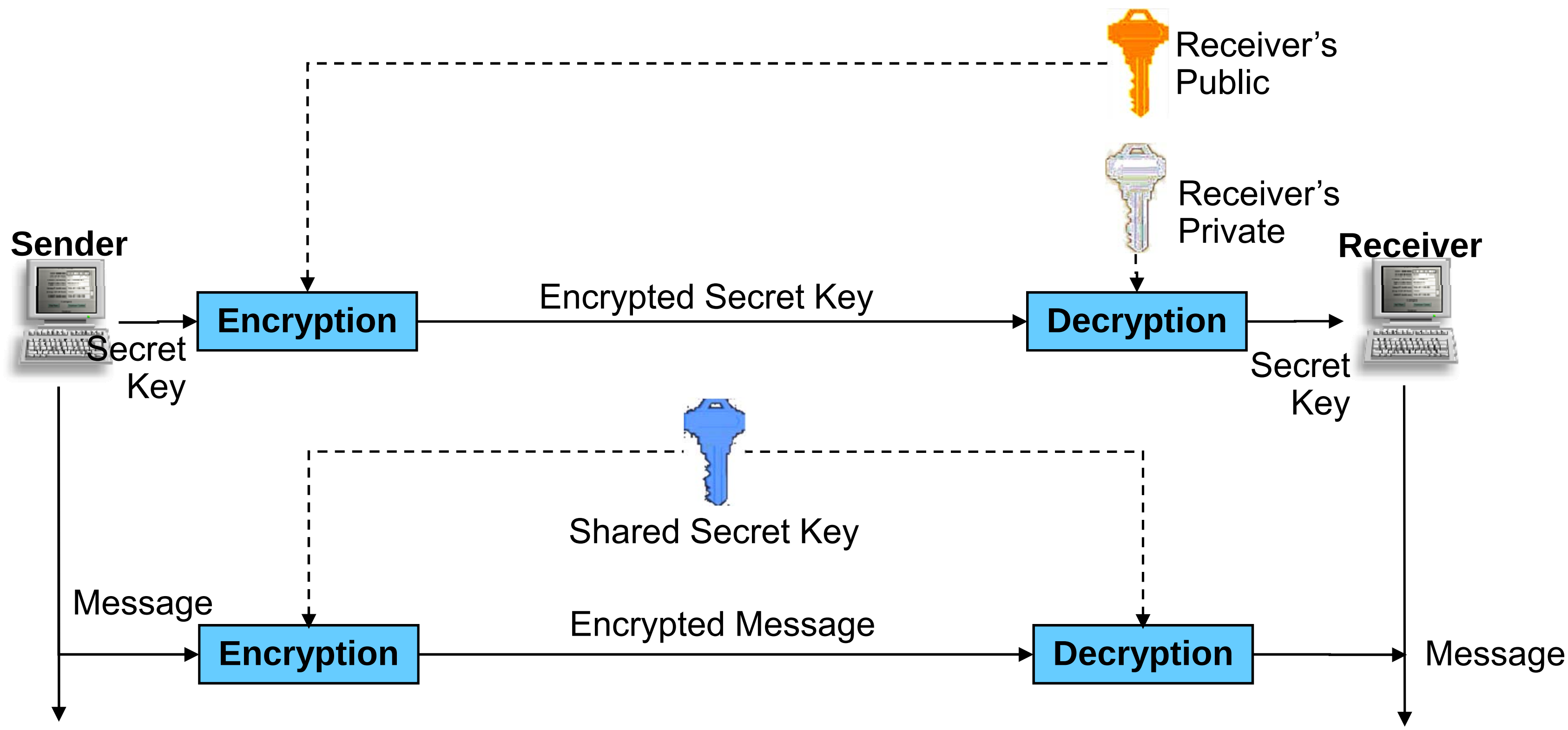
Using Public-Key (cont'd)

To provide both authentication and confidentiality

- Bob first encrypts the message using Alice's public key, then further encrypts the ciphertext with his private key
  - The 1st encryption ensures communication confidentiality
  - The 2nd encryption provides sender authentication
- Alice first decrypts the message using Bob's public key, then decrypts the results using her private key

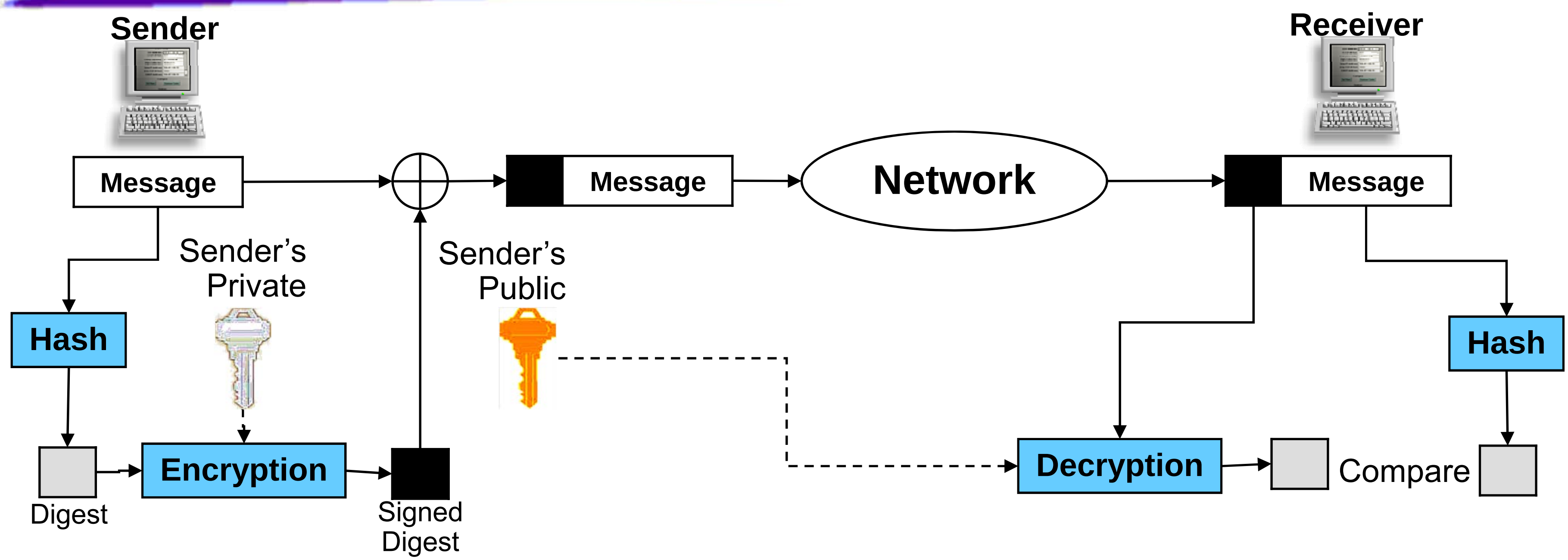


Example: Using Combination of Keys



- Take the efficiency advantage from the secret-key and the advantage of easy key distribution from the public-key

Example: Digital Signature

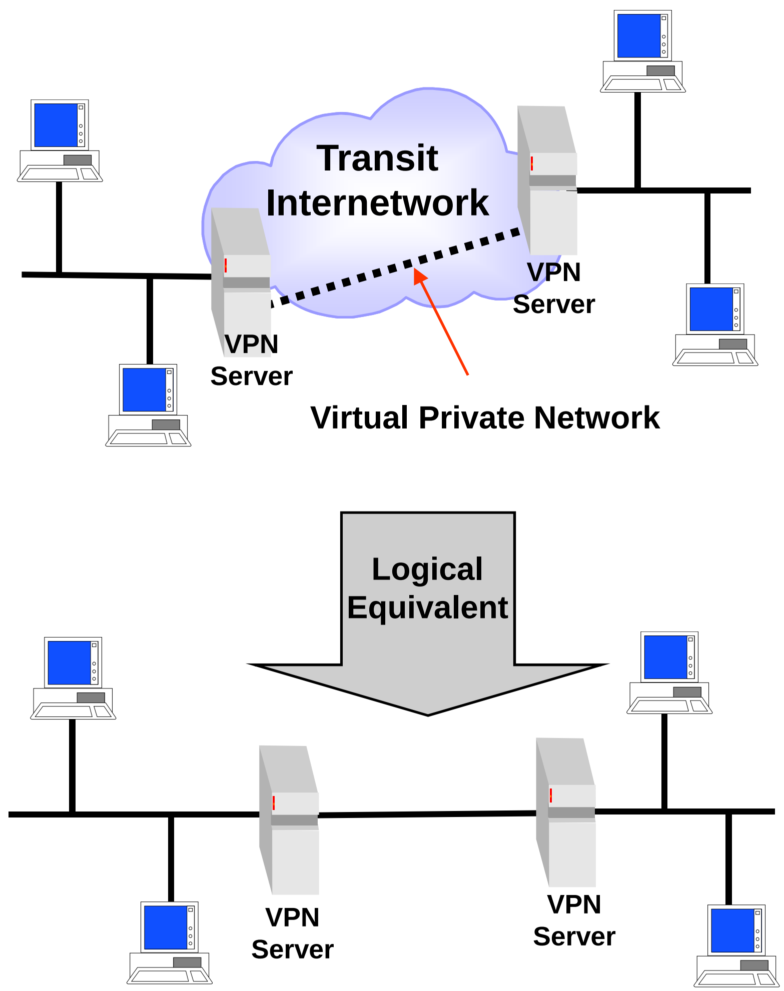


- Digital signature cannot be achieved using only secret keys
- How to overcome the inefficiency of public-key encryption for lengthy document with digital signature?
  - Using *Hash Function* to create a fixed-size digest from a variable-length document
  - Signing the document digest and attaching it with the document
- Digital signature provides integrity, authentication, and nonrepudiation

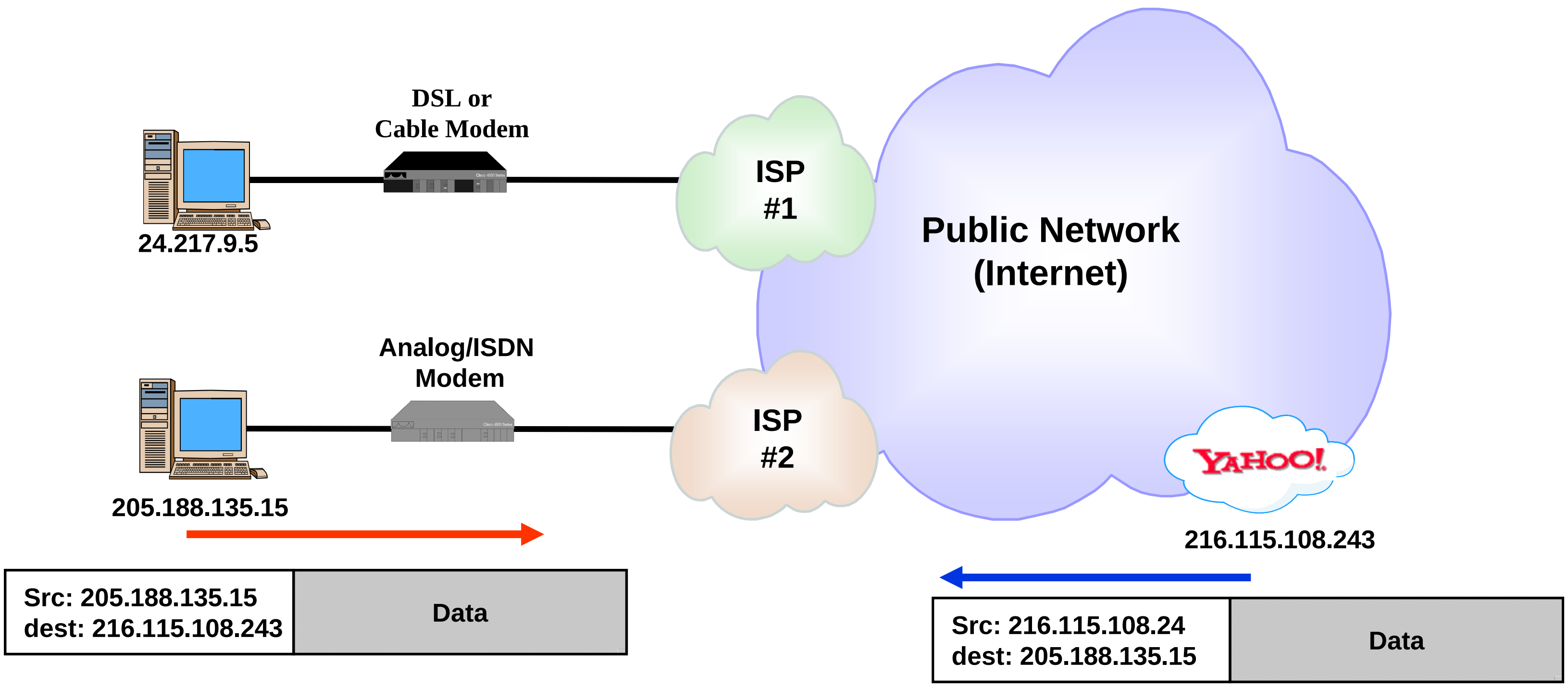
Network Layer Security Example
Virtual Private Network (VPN)

Basic Requirements

- User Authentication
- Address Management
- Data Encryption
- Key Management
- Multiprotocol Support



Internet Access without Tunnel

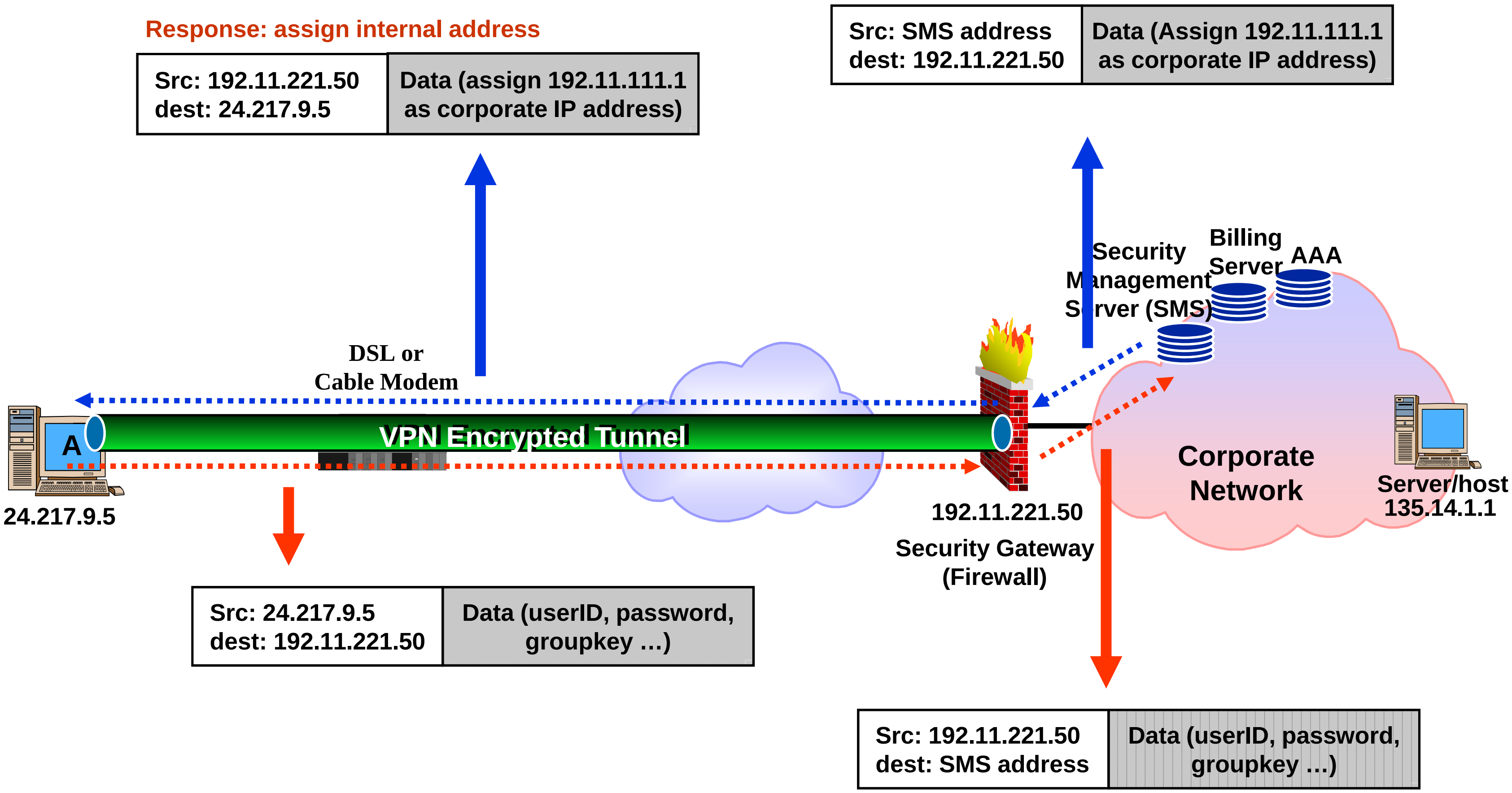


IP security (IPsec)

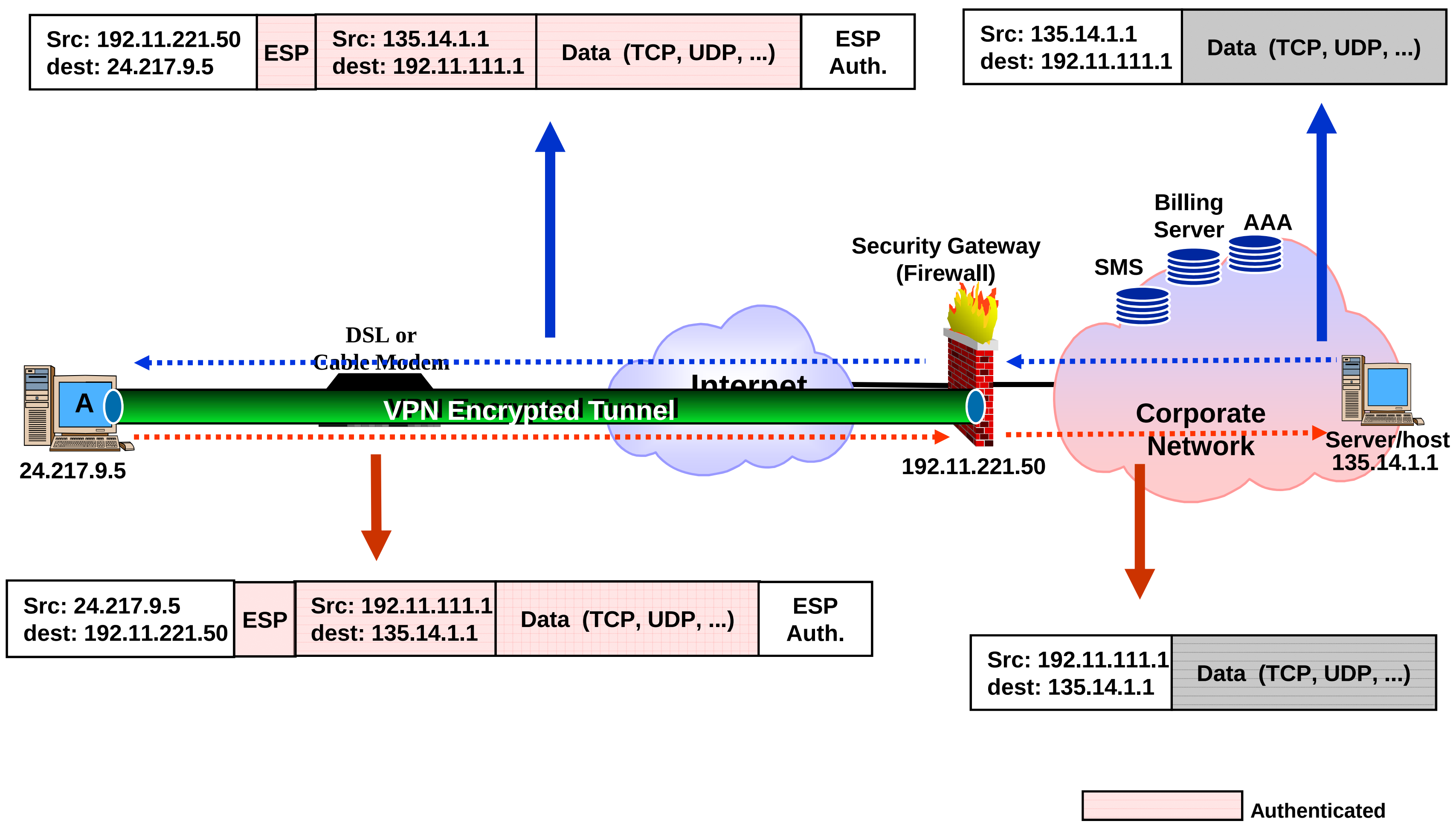
- A set of protocols providing authentication and confidentiality services in the network layer
- Protects all distributed applications
- Higher layer protocols can enjoy the protection provided by IPsec transparently
- Two protocols
  - Authentication protocol, using an Authentication Header (AH)
  - Encryption/authentication protocol, called the Encapsulating Security Payload (ESP)
- Two modes of operation
  - Transport mode: provides protection for upper-layer protocols
  - Tunnel mode: protects the entire IP datagram

Internet Access with IPSec Tunnel

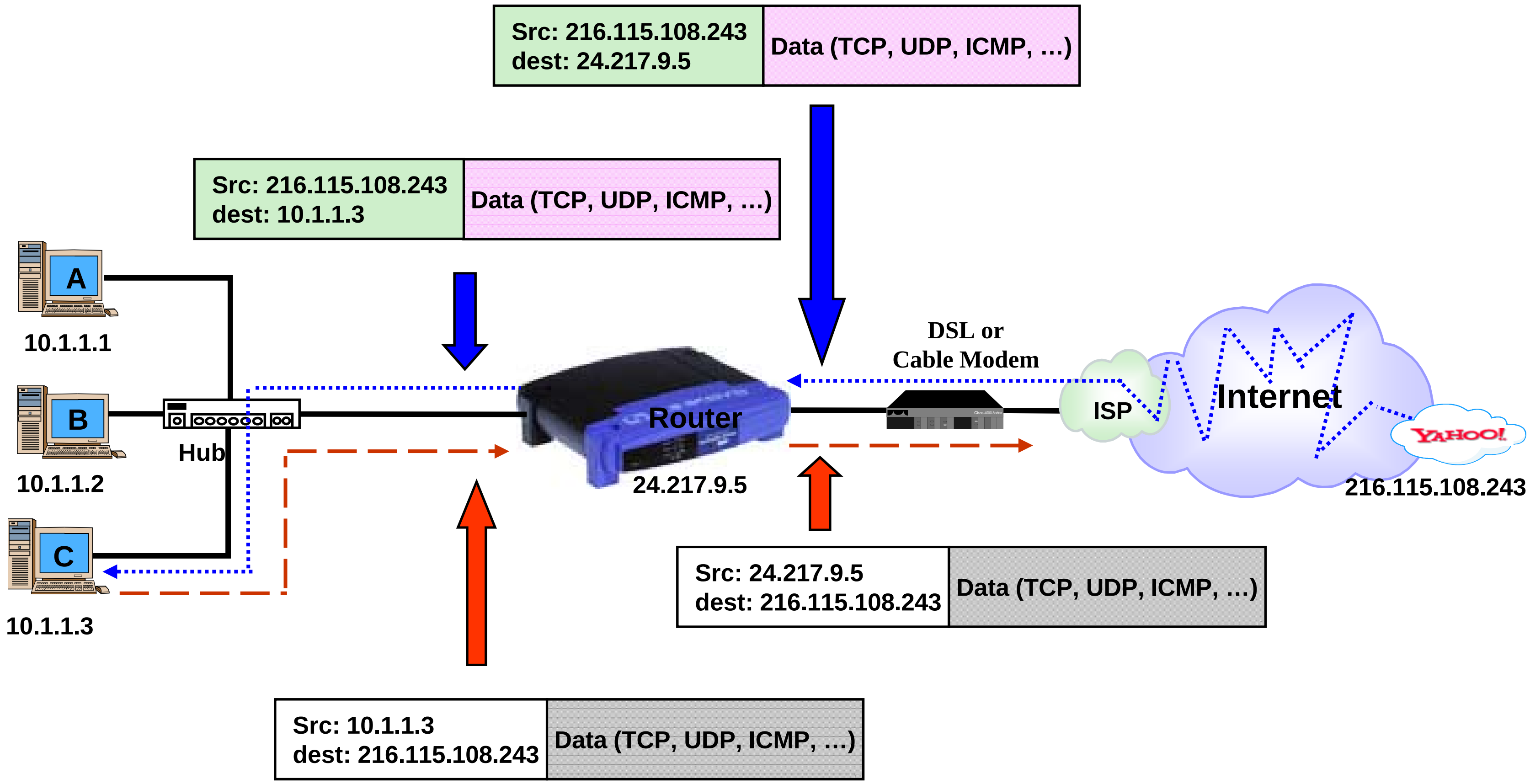
Establish VPN Tunnel



Internet Access with IPSec Tunnel
Data Transfer

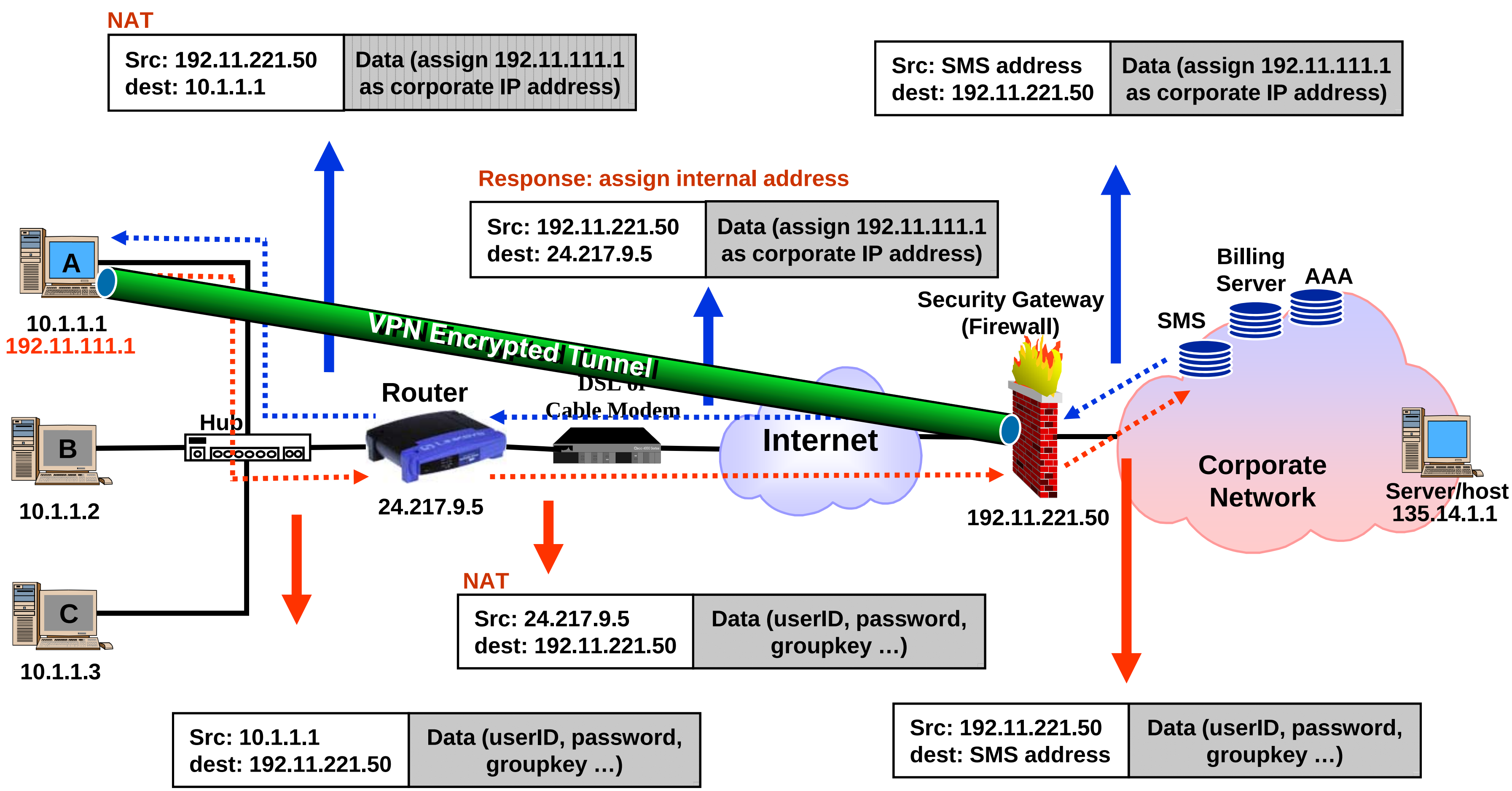


Home LAN Internet Access
without IPSec Tunnel



Home LAN Internet Access with IPSec Tunnel

Establish VPN Tunnel



Home LAN Internet Access with IPSec Tunnel

Data Transfer