CS6551- COMPUTER NETWORKS SHORT NOTES

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

IV SEM CSE

Prepared by

V.R. VIMAL
ASSISTANT PROFESSOR-CSE

BE-CSE Regulations-2013
CS6551 – COMPUTER NETWORKS
Unit-1
ISO/OSI MODEL ARCHITECTURE

The ISO defined a common way to connect computers, called the Open Systems Interconnection (OSI) architecture. (eg. Public X.25 network)

It defines partitioning of network functionality into seven layers as shown.

The bottom three layers, i.e., physical, data link and network are implemented on all nodes on the network including switches.

Physical Layer

The physical layer coordinates the functions required to carry a bit stream over a physical medium.

Representation of bits—To be transmitted, bits must be encoded into signals, electrical or optical. The physical layer defines the type of encoding.

Data rate—It defines the transmission rate (number of bits sent per second).

Line configuration—The physical layer is concerned with the connection of devices to the media (point-to-point or multipoint configuration).

Physical topology—It defines how devices are connected (mesh, star, ring, bus or hybrid) to make a network.

Transmission mode—The physical layer also defines the direction of transmission between two devices: simplex, half-duplex, or full-duplex

Data Link Layer

The data link layer transforms a raw transmission facility to a reliable link.

Framing—The data link layer divides the stream of bits received into manageable data units called *frames*.

Physical addressing—The data link layer adds a header to the frame to define the sender and/or receiver of the frame.

Flow control—If the receiving rate is less than the transmission rate, the data link layer imposes a flow control mechanism to avoid overwhelming the receiver.

Error control—The data link layer adds reliability to the physical layer by adding a trailer to detect and retransmit damaged/lost frames and to recognize duplicate frames. *Access control*—When two or more devices are connected to the same link, data link

layer protocols determines which device has control over the link at any given time.

Network Layer

The network layer is responsible for the source-to-destination delivery of a data unit called packet.

Logical addressing—The packet is identified across the network using the logical addressing system provided by network layer and is used to identify the end systems. *Routing*—The connecting devices (routers or *switches)* prepare routing table to send packets to their destination.

Transport Layer

The transport layer is responsible for *process-to-process* delivery of the entire message.

Service-point addressing—It includes a service-point address or *port* address so that a process from one computer communicates to a specific process on the other computer.

Segmentation and reassembly—A message is divided into transmittable segments, each containing a sequence number. These numbers enable the transport layer to reassemble the message correctly at the destination and to identify/replace packets that were lost.

Connection control—The transport layer can be either connectionless or connectionoriented. *Flow control*—The flow control at this layer is performed end to end.

Error control—The error control at this layer is performed process-to-process. Error correction is usually achieved through retransmission.

Session Layer

The session layer is the network dialog controller. It establishes, maintains, and synchronizes the interaction among communicating systems.

Dialog control—It allows two systems to enter into a dialog and communication between two processes to take place in either half-duplex / full-duplex mode.

Synchronization—The session layer allows a process to add checkpoints, or synchronization points, to a stream of data. For example, when checkpoints are inserted for every 100 pages and if a crash happens during transmission of page 523, then only pages 501 to 523 need to be resent.

Binding—binds together the different streams that are part of a single application. For example, audio and video stream are combined in a teleconferencing application.

Presentation Layer

The presentation layer is concerned with the syntax and semantics of the information exchanged between peers.

Translation—Because different computers use different encoding systems, the presentation layer is responsible for interoperability between these encoding methods. *Encryption*—To carry sensitive information, a system ensures privacy by encrypting the message before sending and decrypting at the receiver end.

Compression—Data compression reduces the number of bits contained in the information. It is particularly important in multimedia transmission.

Application Layer

The application layer enables the user, whether human or software, to access the network. It provides user interface and support for services such as electronic mail, remote file access and transfer, shared database management and several types of distributed information services.

Network virtual terminal—A network virtual terminal is a software version of a physical terminal, and it allows a user to log on to a remote host.

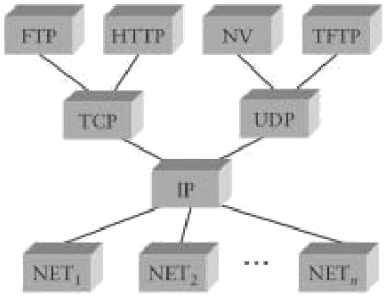
File transfer, access, and management—This application allows a user to access/retrieve files in a remote host, and to manage or control files in a remote computer locally.

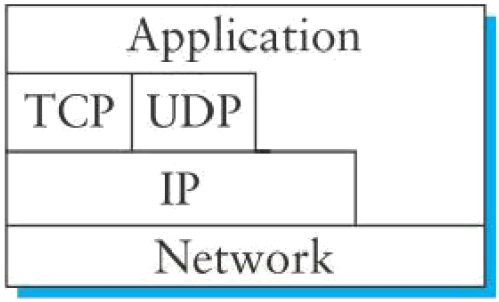
Mail services—This application provides the basis for e-mail forwarding and storage.

Directory services—This application provides distributed database sources and access for global information about various objects and services.

TCP/IP ARCHITECTURE

The *Internet* architecture is also known as TCP/IP architecture and is widely used. This architecture evolved out of a packet-switched network ARPANET. It is a four layered model. It does not enforce *strict layering*, i.e., applications are free to bypass transport layer and can directly use IP or any of the underlying networks.





IP layer serves as focal point in the architecture i.e., it defines a common method for exchanging packets to any type of network and segregates host-to-host delivery from process-to-process delivery.

For any protocol to be added to the architecture, it must also be accompanied by at least one working implementation of the specification. Thus efficiency is ensured.

TCP/IP does not define any specific protocol for the lowest level (physical/data link layers of OSI).

All standard and proprietary protocols such as Ethernet, FDDI, etc are supported.

The protocols are generally implemented by a combination of hardware/software.

Network layer consists of a major protocol, the *Internetworking Protocol* (IP).

It supports the interconnection of multiple networking technologies into a logical internetwork.

It is an unreliable and connectionless protocol.

IP sends data in packets called *datagrams,* each of which is transported separately and independently.

The other protocols supported in this layer are ARP, RARP, ICMP and IGMP.

Transport layer is responsible for delivery of a message from one process to another process. The two protocols supported in this layer are:

Transmission Control Protocol (TCP) for connection-oriented reliable bytestream channel.

User Datagram Protocol (UDP) for connectionless unreliable datagram delivery channel.

Application layer supports a wide range of protocols such as FTP, TFTP, Telnet (remote login), SMTP, etc., that enable the interoperation of popular applications. This layer is equivalent to combined session, presentation, and application layers in the OSI model.

ERROR DETECTION

Error detection is only to see if any error has occurred

A single-bit error or a burst error is immaterial

The basic idea behind any error detection scheme is to add redundant information to a frame that can be used to determine if errors have been introduced.

An efficient system should have *k* redundant bits for *n* data bits such that *k* << *n*

Simple Replication

Transmit two copies of the data

If the two copies are identical at the receiver, then it is correct

If they differ, then an error was introduced and henceforth discarded This is a poor error detection scheme for two reasons:

It sends *n* redundant bits for an *n*-bit message

Any error that corrupts the same bit positions in both copies is undetected

Vertical Redundancy Check (VRC)

It is based on simple parity, which adds one extra bit to a 7-bit code.

The 8th bit is set to make number of 1s in the byte as even, otherwise 0. It is used to detect all odd-number errors in the block.

0110011 -01100110

0110001 – 01100011

Longitudinal Redundancy Check (LRC)

The data bits are divided into equal segments and organized as a table. Parity bit is computed for each column.

The parity byte is appended and transmitted.

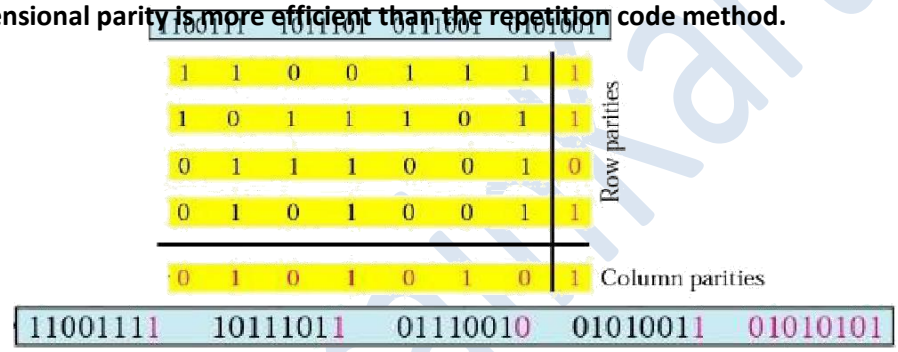
Two-Dimensional Parity

Data is divided into seven byte segments. Both VRC and LRC methods are applied. Even parity is computed for all bytes (VRC).

Even parity is also calculated for each bit position across each of the bytes (LRC). Thus a parity byte for the entire frame, in addition to a parity bit for each byte is sent.

The receiver recomputes the row and column parities. If parity bits are correct, the frame is accepted else discarded.

Two-dimensional parity is more efficient than the repetition code method.



Internet Checksum

The 16-bit checksum is not used at the link layer but by the upper layer protocols (UDP).

Sender

The data is divided into 16-bit words.

The initial *checksum* value is 0.

All words (incl. checksum) are summed using one's complement arithmetic. Carries (if any) are wrapped and added to the sum.

The complement of sum is known as *checksum* and is sent with data

Another way is subtracting *number* from $2n - 1$ (where n is no. of bits)

Receiver

The message (including checksum) is divided into 16-bit words.

All words are added using one's complement addition.

The sum is complemented and becomes the *new checksum*.

If the value of checksum is 0, the message is accepted, otherwise it is rejected.

7	0111		0111
11	1011		1011
12	1100		1100
6	0110		0110
Initial Checksum	0000	Received checksum	1001
Sum	100100	Sum	101101
Carry	10	Carry	10
Sum	0110	Sum	1111
Checksum	1001	New Checksum	0000

Analysis

The nature of checksum is well-suited for software implementation.

It is not as strong as the CRC in error-checking capability

If value of one word is incremented and another word is decremented by the same amount, the errors are not detected because sum and checksum remain the same. If the values of several words are incremented and the total change is a multiple of 65535, then also errors are not detected

Cyclic Redundancy Check (CRC)

Cyclic redundancy check uses the concept of finite fields. CRC was developed by IBM. A *n* bit message is represented as a polynomial of degree *n* – 1.

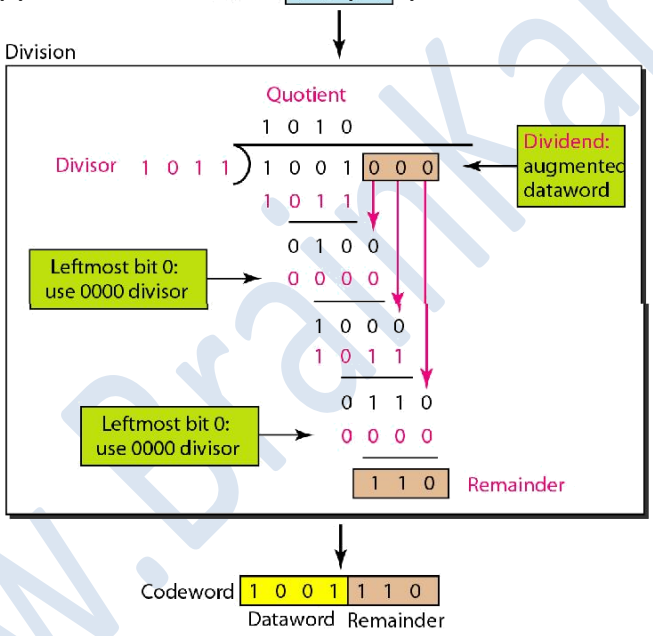
The message M(*x*) is represented as a polynomial by using the value of each bit in the message as coefficient for each term. For eg., 10011010 represents x7 + x4 + x3+ x

For calculating a CRC, a sender and receiver have to agree on a divisor polynomial, *C*(*x*) of degree *k* such that *k n* – 1

Sender

Multiply M(*x*) by *xk* i.e., append *k* zeroes. Let the modified poly be T(*x*)

Divide T(*x*) by C(*x*) using XOR operation. The remainder has *k* bits XOR the remainder with T(*x*) and transmit the resultant (*n* + *k*) bits.

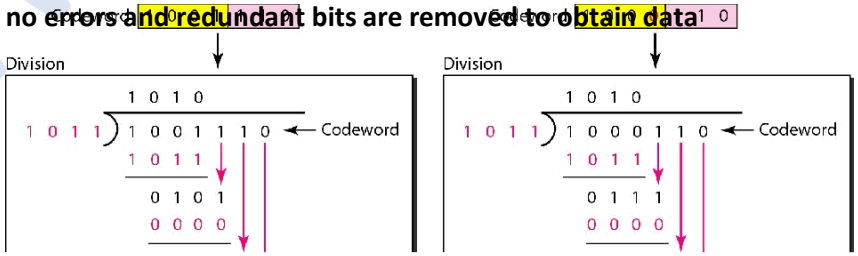


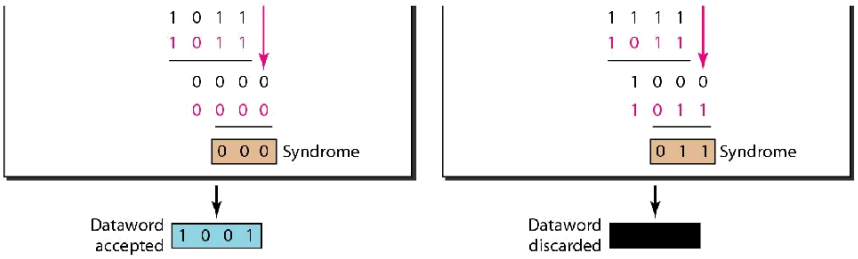
Receiver

Divide the received polynomial by C(*x*) as done in sender

If the remainder is non-zero then discard the frame

If zero, then no errors and redundant bits are removed to obtain data





Divisor Polynomial

The divisor polynomial C(*x*) should has the following error-detecting properties:

All single-bit errors, as long as the xk and x0 terms have nonzero coefficients. Any "burst" error for which the length of the burst is less than *k* bits.

Any odd number of errors, as long as C(*x*) contains the factor (*x* + 1)

The versions of C(*x*) widely used in link-level protocols are CRC-8, CRC-10, CRC-12, CRC-16, CRC-CCITT and CRC-32.

CRC algorithm is implemented in hardware using a *k*-bit shift register and XOR gates. CRC is widely used in networks such as LANs and WANs

ERROR-CORRECTION

Error correcting codes determine the corrupted bits and correct it

Allows the recipient to reconstruct the correct message even after it has been corrupted Hamming code is used to correct single bit error

Reed Solomon code is used to correct burst errors

The use of error-correcting codes is often referred to as forward error correction

Hamming code

Hamming codes are code words formed by adding redundant check bits, or parity bits, to a data word

A frame consists of *m* data (i.e., message) bits and *r* redundant or parity bits.

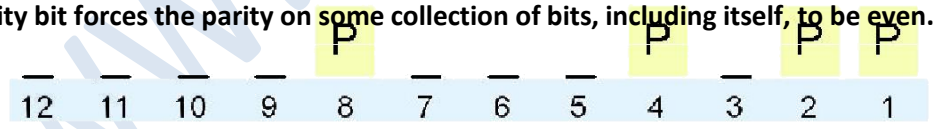
An n-bit unit containing data and parity bits is known as an *n*-bit codeword.

The error-detecting & error-correcting properties of a code depend on Hamming distance. To design a code with *m* message bits and *r* parity bits, that will allow all single errors to be corrected, then 2r >= m + r + 1.

The bits of the codeword are numbered consecutively, starting with bit 1 from the left. The bits that are powers of 2 (*1, 2, 4, 8, 16*, etc.) are parity bits.

The remaining bits (*3, 5, 6, 7, 9, 10, 11, 12*, … etc.) are *m* data bits.

Each parity bit forces the parity on some collection of bits, including itself, to be even.



Each parity bit calculates the parity for some of the bits in the code word. The position of parity bit determines the sequence of bits that it alternately checks and skips.

P1-check parity for bits *1, 3, 5, 7, 9, 11, 13*, … (skip every bit alternately) P2-check parity for bits *2,3, 6,7, 10,11, 14,15*, … (skip every 2 bits alternately)

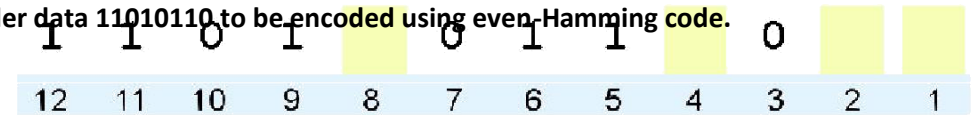
P4-check parity for bits *4,5,6,7, 12,13,14,15*, … (skip every 4 bits alternately)

P8-check parity for bits *8-15, 24-31*, … (skip every 8 bits alternately) P16-check parity for bits *16-31, 48-63*, … (skip every 16 bits alternately)

Set a parity bit to 1 if the total number of ones in the positions it checks is odd.

Example

Consider data 11010110 to be encoded using even-Hamming code.



For the given data, *m* = 8 (no. of bits) and *r* takes value 4 such that 2*r* *m* + *r* + 1
The *unknown* parity bits are embedded into data at bit positions shown below:
For P1, check parity for bits 1, 3, 5, 7, 9 and 11.
Its value is 1 (? 0 1 0 1 1).
For P2, check parity for bits 2, 3, 6, 7, 10 and
11. Its value is 0 (? 0 1 0 0 1).
For P4, check parity for bits 4, 5, 6, 7 and
12. Its value is 1 (? 1 1 0 1).
For P8, check parity for bits 8, 9, 10, 11 and 12. So its value is
1. Its value is 1 (? 1 1 0 1).
The transmitted code word is 110110111001
Error correction
If due to error, the received word is

The receiver *recomputes* parity bits and checks whether it is same as the received one. For *P1'*, check parity for bits 1, 3, 5, 7, 9 and 11.
Its value is 0 (? 0 0 0 1 1) and *differs* with the received
value. For *P2'*, check parity for bits 2, 3, 6, 7, 10 and 11.
Its value is 0 (? 0 1 0 0 1) and is same as the received
value. For *P4'*, check parity for bits 4, 5, 6, 7 and 12.
Its value is 0 (? 1 0 0 1) and *differs* with the received
value. For *P8'*, check parity for bits 8, 9, 10, 11 and 12.
Its value is 1 (? 1 1 0 1) and is same as the received value.
The bit positions of the non-matching parity bits are *added* to determine the corrupt bit. The corrupt bit is *complemented* and hence the original data is restored.
In the given example, parity bits at position 1 and 4 *do not match*. Therefore, the 5th bit is corrupt and changed to 1. The corrected data is 110110111001.

Unit 2

Address Resolution Protocol (ARP)

The main issue is that IP datagrams contain IP addresses.
But the physical address to send, the datagram only understands the addressing scheme of that particular network.
Thus we need to translate the IP address to a link-level address or physical address.
ARP protocol operates between the network layer and the data link layer in the Open System Interconnection (osi) model.
Used to find host's hardware address when only its network layer address is known. Solution would be for each host to maintain a table of address pairs.
The table would map IP addresses into physical addresses.
This table is managed by a system administrator and then copied to each host on network. This is accomplished by using the Address Resolution Protocol (ARP).
Goal of ARP is to enable each host on a network to build up a table of mappings between IP addresses and link-level addresses.
The IP-to-MAC address mappings are derived from an ARP cache maintained on each device.
It first checks for a mapping in the cache. If no mapping is found, it needs to invoke the Address Resolution Protocol over the network.

Working of ARP:

When a device wishes to send data to another target device over Ethernet, it must first determine the MAC address of that target given its IP address.

These IP-to-MAC address mappings are derived from an ARP cache maintained on each device.
If the given IP address does not appear in a device's cache, that device cannot direct messages to that target until it obtains a new mapping.
To do this, the initiating device first sends an ARP request broadcast message on the local subnet.
The host with the given IP address sends an ARP reply in response to the broadcast, allowing the initiating device to update its cache and proceed to deliver data.

ARP packet format :

ARP packet

A HardwareType field, which specifies the type of physical network (e.g., Ethernet).
A ProtocolType field, which specifies the higher-layer protocol (e.g., IP).
HLen ("hardware" address length) and PLen ("protocol" address length) fields, specify the length of the link-layer address and higher-layer protocol address, resp.
An Operation field, specifies whether this is a request or a response. The source and target hardware (Ethernet) and protocol (IP) addresses.

RARP

The Reverse Address Resolution Protocol (RARP) is an obsolete computer networking protocol used by a host computer to request its Internet Protocol address from a gateway server's Address Resolution Protocol table or cache.
It has available its Link Layer address, such as a MAC address.
To obtain IP address the host first broadcasts an RARP request packet containing its MAC address on the network.
All hosts on the network receive the packet, but only the server replies to the host by sending an RARP response packet.
It has the host's MAC and IP addresses.
One limitation with RARP is that the server must be located on the same physical network as the host.

RARP mechanism

RARP message is sent from one machine to the another encapsulated in the data portion of a network frame.
Sender braodcasts a RARP request that specifies itself as both the sender and receiver. supplies its physical network address in the target hardware address field.
All machines on the network receive the request, but only those authorised to supply the RARP services process the request and send a reply.
Such machines are known as RARP servers. For RARP to succeed, the network must contain at least one RARP server.
Servers answers the request by filling in the target protocol address field, changing the message type from request to reply
Sending the reply back directly to the machine that makes the request.

DHCP

It is not possible for the IP address to be configured once into a host when it is manufactured. (IP address=Network part + host part)
IP addresses need to be reconfigurable. Since they move to another n/ w.
some other pieces of information a host needs to have before it can start sending packets. most notable of these is the address of a default router.
DHCP lets a network administrator supervise and distribute IP addresses from a central point and automatically sends a new IP address when a computer is plugged into a different place in the network.
Most host operating systems provide a way for a system administrator, or even a user, to manually configure the IP information needed by a host. There are some drawbacks to such manual configuration.
One is that it is simply a lot of work to configure all the hosts in a large network directly. The process is very error-prone.
For these reasons, automated configuration methods are required.
Primary method is to use a protocol known as the Dynamic Host Configuration Protocol (DHCP). DHCP relies on the existence of a DHCP server for providing configuration information to hosts. DHCP server function just as a centralized repository for host configuration information. Configuration information for each host could be stored in the DHCP server. Automatically retrieved by each host when it is connected to the network.

DHCP process works

A network device attempts to connect to the
Internet. The network requests an IP address.
The DHCP server maintains a pool of IP addresses and leases an address to any DHCP-enabled client when it starts up on the network.
The DHCP server allocates (leases) the network device an IP address, which is forwarded to the network by a router.
DHCP updates the appropriate network servers with the IP address and other configuration information.
The network device accepts the IP address. The IP address lease
expires. DHCP either reallocates the IP address or leases one that is
available. The network device is no longer connected to the Internet.
The IP address becomes an available address in the network pool of IP addresses
To contact a DHCP server, a newly booted host sends a DHCPDISCOVER message to a special IP address (255.255.255.255) that is an IP broadcast address.
It will be received by all hosts and routers on that network.

One of these nodes is the DHCP server for the network.
The server then reply to the host that generated the discovery message (all the other nodes would ignore it.)
It is not really desirable to require one DHCP server on every network. DHCP uses the concept of a *relay agent*.
At least one relay agent on each network, and it is configured with the information: the IP address of the DHCP server.
When a relay agent receives a DHCPDISCOVER message, it unicasts it to the DHCP server and send back the response to the requesting client.

Message from host to DHCP

DHCP packet format

Internet Control Message Protocol (ICMP)

IP is always configured with a companion protocol, known as the Internet Control Message Protocol (ICMP).
It is a collection of error messages that are sent back to the source host whenever a router or host is unable to process an IP datagram successfully.
For example, ICMP defines error messages

- that the destination host is unreachable (perhaps due to a link failure),

- that the reassembly process failed
- that the TTL had reached 0
- that the IP header checksum failed etc.,
- ICMP also defines a handful of control messages that a router can send back to a source host.

Most useful control messages, called an ICMP-Redirect, tells the source host that there is a better route to the destination.
Suppose a host is connected to a network that has two routers , R1 and R2, where the host uses R1 as its default router.
R1 receive a datagram from the host, based on its forwarding table it knows that R2 would have been a better choice for a particular destination address.
The router sends an ICMP-Redirect back to the host, instructing it to use R2 for future datagrams.
The host then adds this new route to its forwarding table

Dynamic Host Configuration Protocol (DHCP)

A network protocol used to configure devices that are connected to a network. so they can communicate on that network using the Internet Protocol (IP).
The protocol is implemented in a client-server model, in which DHCP clients request configuration data, such as an IP address, a default route, and one or more DNS server addresses from a DHCP server.

Routers

We have assumed that the switches and routers have enough knowledge of the network topology so they can send packet to the right destination.
Fundamental problem of routing is, how do switches and routers acquire the information in their forwarding tables?
Forwarding: Taking a packet, looking at its destination address, consulting a table, and sending the packet in a direction determined by that table.
The forwarding table is used when a packet is being forwarded and so must contain enough information to accomplish the forwarding function.
Routing is the process by which forwarding tables are built. Cont..
Routing table is built up by the routing algorithms as a precursor to building the forwarding table.
It generally contains mappings from network numbers to next hops.

Bridges and Switches

To inter connect pair of Ethernets a repeater may be used.
But it may exceed the physical limitations of the Ethernet. (no more than four repeaters between any pair of hosts.)
Alternative: a node b/w the two Ethernets & the node forward frames from one Ethernet to the other.
This node is called a *bridge.* Collection of LANs connected by one or more bridges is usually said to form an *extended LAN.*
Bridge: multi-input, multi output device, transfers packets from an input to one or more outputs.

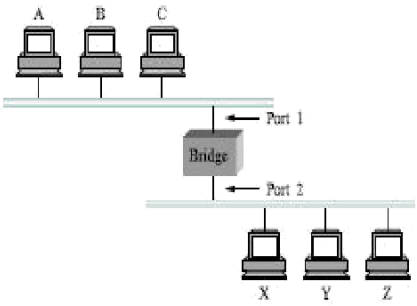


Figure 3.11 Illustration of a learning bridge.

Bridges

Whenever a frame from host A addressed to host B arrives on port 1, no need for the bridge to forward the frame to port 2.

How does a bridge knows which port hosts reside. Download a table into the bridge. Bridge inspect the *source address in all the frames it receives.*

Bridge records that a frame from host A received on port 1.

Forwarding Table

Host	Port
A	1
B	1
C	1
X	2
Y	2
Z	2

Table 3.4 Forwarding table maintained by a bridge.

Working of a bridge Filtering:

Check destination address of a frame and decide frame should be forwarded or dropped. frame to be forwarded, specify the port.

A bridge has a table that maps addresses to ports.

Implementation

```
void
updateTable (MacAddr src, int inif)
{
BridgeEntry *b;
if (mapResolve(bridgeMap, &src, (void
**)&b) == FALSE )
{
/* this address is not in the
table, so try to add it */
if (numEntries < BRIDGE_TAB_SIZE)
{
b = NEW(BridgeEntry);
b->binding = mapBind( bridgeMap, &src,
b); /* use source address of packet as
dest. Cont..
```

address in table */
b->destination =
src; numEntries++;
}
else
{
/* can't fit this address in the table
now, so give up */
return;
}
}
/* reset TTL and use most recent input interface
*/ b->TTL = MAX_TTL;
b->ifnumber = inif;
}
Loop problem:
work well if no redundant bridges.
if redundant bridges[more than one bridge b/w pair of LANs] – make system more reliable. create loops.
if bridge fails, another one
works. Problem :
maintains two copies of frame on LAN1.
in each iteration newly generated fresh copies of the frames will be flooded the n/w.

SPANNIG TREE ALGORITHM

Main idea of the spanning tree is to select the ports over which they will forward frames Process to find the spanning Tree:

- Every bridge has unique ID (serial number)
  Each bridge broadcasts ID.
  Bridge with smallest ID selected as root bridge. Bridge B1.
- algorithm find the shortest path (path with shortest cost) from bridge to other bridge or LAN.
  Shortest Path : examining total cost from root bridge to the destination.
- combination of shortest paths creates the shortest tree.
- ports are part of the spanning tree.

forwarding ports:
forward a frame that the bridge receives.
blocking ports:
block the frames received by the bridge.

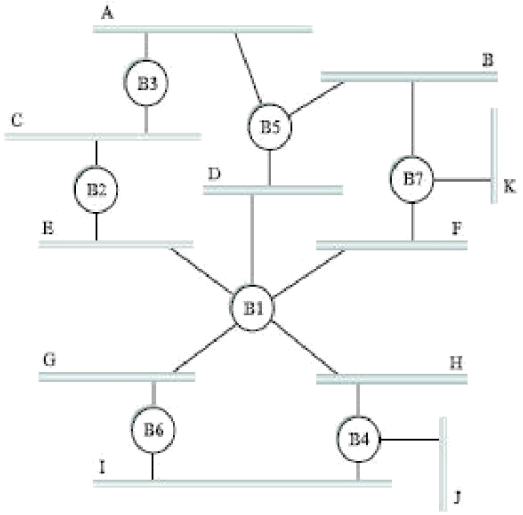


Figure 3.12 Extended LAN with loops.

Spanning Tree is a graph in which there is no loop.
LAN reached another LAN through one path only (no loop).
connecting arcs show the connection of a LAN to a bridge and vice versa. To find spanning tree, need to assign a cost(metric) to each arc.
Cost: - path with minimum hops (nodes)
path with minimum delay.
path with maximum bandwidth.
Minimum hop is chosen.
Hop count – 1 from bridge to LAN & 0 in reverse.
Bridges B1, B4, and B6 form a loop

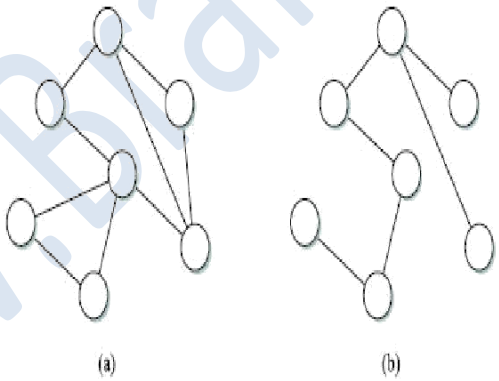


Figure 3.13 Example of (a) a cyclic graph; (b) a corresponding spanning tree.

Extended LAN as being represented by a graph has loops (cycles)
Spanning tree is a sub graph of this graph that covers (spans) all the vertices, but contains no cycles.
Each LAN's designated bridge is the one that is closest to the root, if two or more bridges are equally close to the root, then the bridges' identifiers are used to break ties
In above example, B1 is the root bridge, since it has the smallest ID.
Both B3 and B5 are connected to LAN A, but B5 is the designated bridge since it is closer to the root.

Both B5 and B7 are connected to LAN B, but B5 is the designated bridge (smaller ID) both are an equal distance from B1.

Configuration message

Bridges have to exchange configuration messages with each other and then decide whether or not they are the root or a designated bridge.
configuration messages contain three pieces of information:
1 The ID for the bridge that is sending the message;
2 The ID for what the sending bridge believes to be the root bridge;
3 The distance, measured in hops, from the sending bridge to the root bridge. Each bridge thinks it is the root, and sends a configuration message.
Receiving a configuration message over a particular port, the bridge checks to see if the new message is better than the current best configuration message
If the new message is better than the current information, the bridge discards the old information and saves the new information.

Unit 3

Routers

Switches and routers have enough knowledge of the network topology so they can send packet to the right destination.
Fundamental problem of routing is, how do switches and routers acquire the information in their forwarding tables?
Forwarding: Taking a packet, looking at its destination address, consulting a table, and sending the packet in a direction determined by that table.
The forwarding table is used when a packet is being forwarded and so must contain enough information to accomplish the forwarding function.
Routing is the process by which forwarding tables are built.
Routing table is built up by the routing algorithms as a pre work to build the forwarding table. It generally contains mappings from network numbers to next hops.
The difference between
Routing and Forwarding
Routing means finding a suitable path for a packet from sender to destination and Forwarding is the process of sending the packet toward the destination based on routing information

Network as a Graph

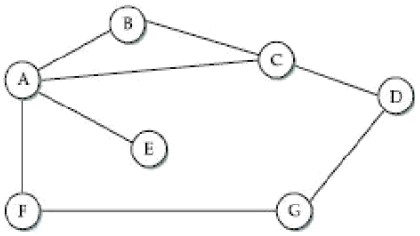
It is a graph representing a network. The nodes of the graph, labeled A through F, may be either hosts, switches, routers, or networks.
Edges of the graph correspond to the network links. Each edge has an associated *cost.* Two main classes of routing protocols:

1. *distance vector and*
2. *link state*

Distance Vector (RIP)

Distance vector algorithms use the Bellman–Ford algorithm Bellman–Ford algorithm. Routing Information Protocol (RIP)
Each node constructs a one-dimensional array containing the “distances” (costs) to all other nodes and distributes to its immediate neighbors.
The starting assumption of RIP routing is that each node knows the cost of the link to each of its directly connected neighbors.

Working of RIP



Distance-vector routing: an example network.

The cost of each link is set to 1, so that a least-cost path is simply the one with the fewest hops. Distances to all other nodes is given in Table

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

Table 4.5 Initial distances stored at each node (global view).

The table is a list of distances from one node to all other nodes. Step 1:
Initially, each node sets a cost of 1 to its directly connected neighbors and ∞ to all other nodes. Thus, A initially can reach B in one hop and that D is unreachable.
The routing table at A includes the name of the next hop that A use to reach any reachable node. (one that bears its name in the left column)
Each node only knows the information in one row of the table. Cont..
Step 2:
Next step in distance-vector routing is every node sends a message to its neighbors containing its personal list of distances.

Destination	Cost	Next Hop
B	1	B
C	1	C
D	∞	—
E	1	E
F	1	F
G	∞	—

Table 4.6 Initial routing table at node A.

For example, node F tells node A that it can reach node G at a cost of 1; A also knows it can reach F at a cost of 1, so it adds these costs to get the cost of reaching G by means of F. Destination Cost Next Hop

B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Final routing table at node A.

The process of getting consistent routing information to all the nodes is called *convergence.* Table shows the final set of costs from each node to all other nodes when routing has converged.

Two different circumstances for a node to decide to send a routing update to its neighbors. They are

1. periodic update and

2. triggered update

1. *periodic update:* each node automatically sends an update message every other node so often, even if nothing has changed

2. *triggered update:* when a node receives an update from one of its neighbors that causes it to change one of the routes in its routing table.

Final Distance

Sharing & Updation

Each node *shares its cost list (distance) to all of its directly connected neighbors.*

Node *A receives distance vectors from B, C, E and F.*

The tables received by *A from C and F*

are: Tables received by *A from C & F*

Now node *A can use information from its neighbors to reach other unreachable nodes.*

For example, node *F tells node A that it can reach node G at a cost of 1.*

Each node updates its routing table by comparing with its neighbor tables as follows o For each destination Total Cost is computed as:

Total Cost = Cost(*Node, Neighbor) + Cost(Neighbor, Destination).*

If Total Cost < NodeCost(*Destination) then*

NodeCost(*Destination) = Total Cost and Next Hop(Destination) = Neighbor*

Node failure

when a link or node fails:
nodes that notice the failure send new lists of to neighbors.
The question how a node detects a failure. couple of answers:
1. a node continually tests by sending a control packet to another node and seeing if it receives an acknowledgment.
2. a node determines that the link is down if it does not receive the expected periodic routing update for the last few update cycles.
For example:
when F detects that its link to G has failed. First, F sets its new distance to G to infinity and passes that information along to A. Since A knows that its 2 hop path to G is through F, A would also set its distance to G to infinity.
Next update from C, A knows that C has a 2-hop path to G. Thus, A would know that it can reach G in 3 hops through C, which is less than infinity, and so A update its table.
Slightly different circumstances can prevent the network from
stabilizing. For example: The link from A to E goes down.
In the next updates, A advertises a distance of infinity to E, but B and C advertise a distance of 2 to E.
Node B, hearing that E can be reached in 2 hops from C, can reach E in 3 hops and advertises this to A.
Node A can now reach E in 4 hops and sends this to C. A concludes that it can reach E in 5 hops. This cycle stops only when the distances reach some large enough no. to be considered infinite. In the meantime, none of the nodes knows that E is unreachable, and the routing tables for the network do not stabilize. This situation is known as the *count to infinity problem*
Several partial solutions to this problem.
1. To use some relatively small number as an approximation of infinity.
For example, we might decide that the maximum number of hops across a certain network is not more than 16, and so we pick 16 as the value that represents infinity.

Stabilize routing

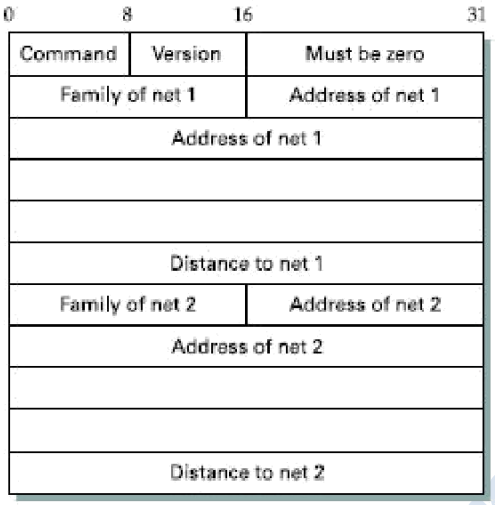
A technique to improve the time to stabilize routing is called *split horizon.*
When a node updates its neighbors, it does not send those routes it learned from each neighbor back to that neighbor. This is known as *split horizon.*
Cont..
Infinity is redefined to a small number. Most implementations define 16 as infinity.
o Distance between any two nodes should not exceed 15 hops.
o Thus distance vector routing *cannot be used in large networks.*

Routing Information Protocol (RIP)

RIP is an intra-domain routing protocol based
on distance-vector algorithm.
It is extremely simple and widely used.
The routers advertise the cost of reaching networks, instead of reaching other routers. RIP takes the simplest approach, with all link costs being equal to 1.
The distance is defined as the number of links to reach the
destination. The metric in RIP is called a *hop count.*
One of the most widely used routing protocols in IP networks is the Routing Information Protocol (RIP)
Rather than advertising the cost of reaching other routers, the routers advertise the cost of reaching networks.
For example, in Figure router C would advertise to router A that it can reach

networks 2 and 3 at a cost of
0; networks 5 and 6 at cost 1;
network 4 at cost 2.

RIP packet format



RIP packet format.

Link State (OSPF)

Each node knows the *state of link to its neighbors and the cost involved.*
Link-state routing protocols rely on two mechanisms:
1. Reliable dissemination of link state information
2. Route calculation from the accumulated link state knowledge
Link-state routing is the second major class of intra domain routing protocol. Link-state routing are similar to distance-vector routing.
Each node is assumed to be capable of finding out the state of the link to its neighbors and the cost of each link.
Need to provide each node with enough information to enable it to find the least-cost path to destination.

Reliable Flooding

It is the process of making all the nodes to get a copy of the link state information from all the other nodes.
The basic idea is for a node to send its link-state information to all its directly connected links. Receiving node forwards this information to all of *its* links. This process continues until the information has reached all the nodes in the network.
Each node creates an update packet, called a link-state packet (LSP).
Link State Packet
LSP contains
The ID of the node that created the LSP.
A list of directly connected neighbors of that node, with the cost of the link to each one. A sequence number.
A time to live for this packet.
The first two are needed to enable route calculation.
The last two are used to make the process reliable.
Reliability includes making sure that the information, is the most recent copy, since there may be multiple, LSPs from one node.

Working of LSP

First, the transmission of LSPs between adjacent routers is made reliable using acknowledgments and retransmissions.
Node X receives a copy of an LSP originated at node Y. X checks to see if it has already stored a copy of an LSP from Y. If not, it stores.
If already has, it compares the sequence numbers. if the new LSP is larger, it is assumed to be the more recent, and LSP is stored, replacing the old one.
A smaller (or equal) sequence number would imply an LSP older than the one stored, so it discards.
If the received LSP was the newer one, X then sends a copy of that LSP to all of its neighbors except from which the LSP was just received.

Flooding

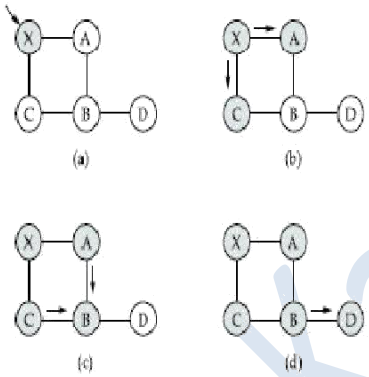


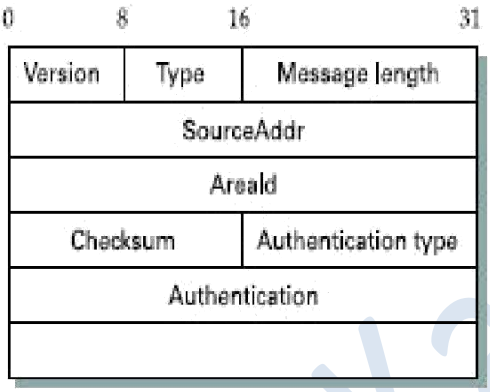
Figure 4.17 Flooding of link-state packets. (a) LSP arrives at node X; (b) X floods LSP to A and C; (c) A and C flood LSP to B (but not X); (d) flooding is complete.

Link failure

The failure of a link can be detected by the link-layer protocol.
The loss of connectivity to that neighbor can be detected using periodic "hello" packets. Each node sends to its immediate neighbors at defined intervals.
If a long time passes without receipt of a "hello" from a neighbor, the link to that neighbor will be declared down.
A new LSP will be generated to reflect this fact.
Important design goals of a link-state protocol's flooding mechanism:
The newest information must be flooded to all nodes as quickly as possible, and old information is removed and not allowed to circulate.
To make sure that old information is replaced by newer information, LSPs carry sequence numbers.
Each time a node generates a new LSP, it increments the sequence number by 1.
LSPs also carry a time to live. A node always decrements the TTL of a newly received LSP before flooding it to its neighbors.
Difference b/w Distance-vector and Link-state algorithms
In distance vector, each node talks only to its directly connected neighbors.
It tells them everything it has learned (i.e., distance to all nodes).
In link state, each node talks to all other nodes, but it tells them only what it knows for sure (only the state of its directly connected links)

Open Shortest Path First Protocol (OSPF)

One of the most widely used link-state routing protocols is OSPF.
It adds a number of features to the basic link-state algorithm. They are



OSPF header format.

Authentication of routing messages: Early OSPF used a simple 8-byte password for authentication. This is not a strong enough form of authentication users, but it alleviates many problems.

Strong cryptographic authentication was later added.

Additional hierarchy: Hierarchy is one of the fundamental tools used to make systems more scalable.

OSPF introduces another layer of hierarchy into routing by allowing a domain to be partitioned into *areas.*

Load balancing: OSPF allows multiple routes to the same place to be assigned the same cost and will cause traffic to be distributed evenly over those routes.

OSPF header format

Unit 4

User Datagram Protocol (UDP)

The User Datagram Protocol (UDP)is a datagram transport which uses the underlying IP protocol for its network layer. It is used when there is a need to transmit short packets through a network where there is no stream of data to be sent as in TCP. It is consequently a much simpler protocol and therefore much easier to handle. It is also less reliable in that there are no sequence numbers and other error recovery techniques available. If errors and lost packets are important then the user of UDP must cater for these.

The format of a UDP header is shown in

figure *source port*

This field performs the same function as in

TCP. *destination port*

This field is also the same as in TCP. Note that the same port number can be used by TCP and UDP for different purposes.

length

This field contains the length of the data field. *checksum*

As in TCP, this field is the checksum of the header plus parts of the IP header. *Data* This field is passed to the relevant

program

No connection needs to be set up

Throughput may be higher because UDP packets are easier to process,

especially at the source

The user doesn't care if the data is transmitted reliably

The user wants to implement his or her own transport protocol

TCP

The Transmission Control Protocol (TCP) is one of the main transport layer protocols used with IP. It is a connection oriented protocol based on the connectionless IP protocol. Because it is the lowest layer which has end-to-end communication, it needs to handle things such as lost packets. In this respect it is similar to the data-link layer which must handle errors on an individual link.
Consequently many of its facilities are familiar from our discussion of the data-link layer.
The format of a TCP header is shown in

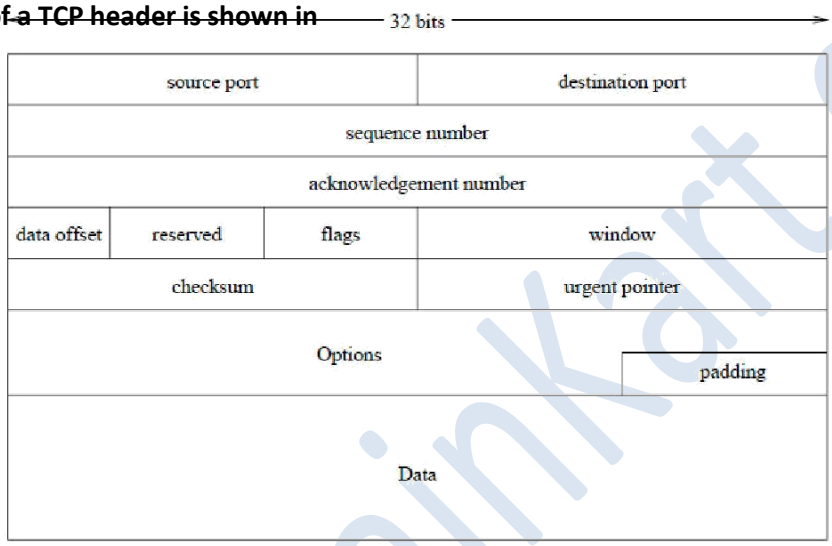


Figure 15.1. TCP header format

Source port

All of the address fields in the lower layer protocols are only concerned with getting the packet to the correct host. Often, however, we want to have multiple connections between two hosts. The source port is simply the number of the outgoing connection from the source host.

Destination port

Similarly, this is the number of the incoming connection on the destination host. There must be a program on the destination host which has somehowtold the networking system on that host that it will accept packets destined for this port number. Standard system services such as SMTP, NNTP and NTP (all described in later chapters) have well known standard port numbers. Thus to connect to the SMTP port on a particular host (in order to transmit some email) a TCP connectionwould be
set up with the correct destination port number (25). The source port number does not matter except that it should be unique on the sending
machine so that replies cvan be received correctly.

Sequence number

This is the sequence number of this packet. It differs from the usual data-link layer sequence number in that it is in fact the sequence number of the first byte of information and is incremented by the number of bytes in this packet for the next message. In other words, it counts the number of bytes transmitted rather than the number of packets.

Acknowledgement number

This is the sequence number of the last byte being acknowledged. This is a piggybacked

acknowledgement.

data offset

This field is the offset in the packet of the beginning of the data field. (In other words it is the length of the header.)

flags

This field contains several flags relating to the transfer of the packet. We will not consider them further here.

window

This field is used in conjunction with the acknowledgement number field. TCPuses a sliding window protocol with a variable window size (often depending on the amount of buffer space available. This field contains the number of bytes which the host is willing to accept from the remote host.

checksum

This field contains a checksum of the header. It actually uses a modified form of the header which includes some of the information from the IP header to detect some unusual types of errors.

urgent pointer

There is provision in TCP for some urgent data messages to be sent bypassing the normal
sequence number system. This field is used to indicate where such data is stored in the packet.

UDP

The User Datagram Protocol (UDP) is a transport layer protocol defined for use with IP network layer protocol .

User Datagram Protocol (UDP) is a connectionless, unreliable transport protocol.

Simplest transport protocol should extend the host-to-host delivery service.

There are many processes running on a host, so the protocol needs a level of demultiplexing, to allow multiple processes on each host to share the network.

User Datagram Protocol (UDP) is an example of such a transport protocol.

Only issue in such a protocol is the form of the address used to identify the target process.

The User Datagram Protocol (UDP) is one of the core members of the Internet protocol suite (the set of network protocols used for the Internet).

UDP does not implement flow control or reliable/ordered delivery

With UDP, computer applications can send messages, as *datagram's*, to other hosts on an Internet Protocol (IP) network without prior communications to set up special transmission channels or data paths.

It has no handshaking dialogues.

It is used in videoconferencing applications , computer games specially tuned for real-time performance.

If a process wants to send a small message and does not require reliability, UDP is used.

The first eight (8) bytes of a datagram contain header information and the remaining bytes contain message data.

There is no flow-control mechanism.

Port Number:

Each process is assigned a unique 16-bit port number on that host. Processes are identified by (host, port) pair.

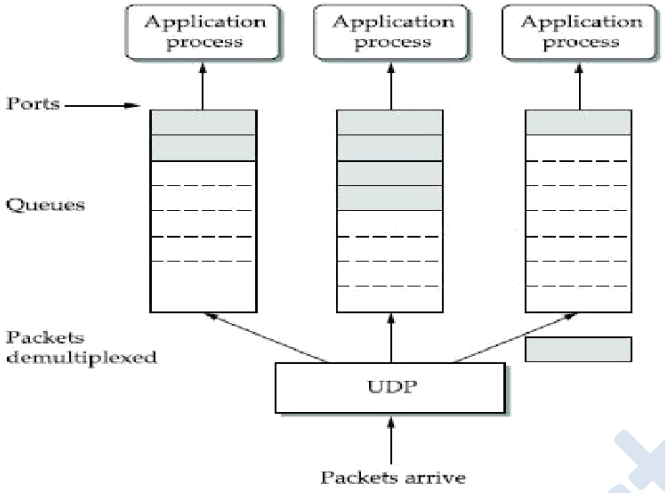
Processes can be classified as either as *client / server.*

Ports are usually implemented as a message queue.

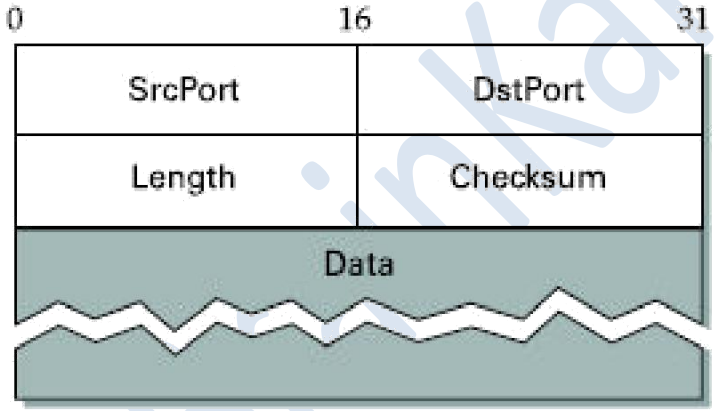
When a message arrives, UDP appends the message to the end of the queue. When queue is full, the message is discarded.

When a message is read, it is removed from the queue. When queue is empty the process is blocked

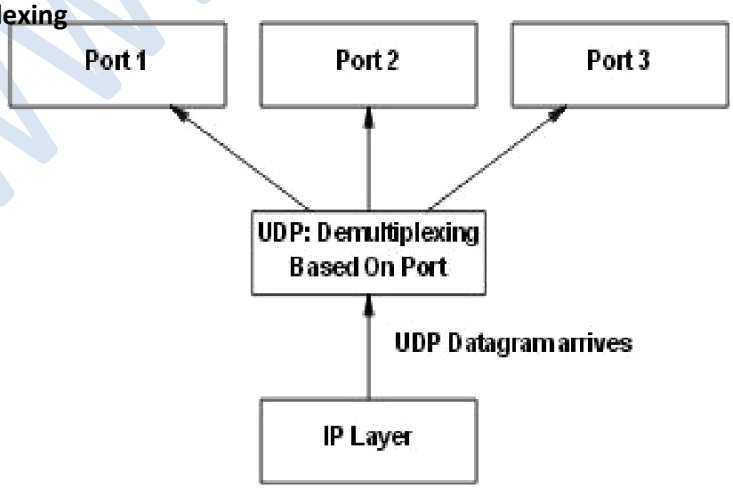
UDP message queue



UDP header



Example of demultiplexing



User Datagram Protocol:

UDPPorts and Port Numbers

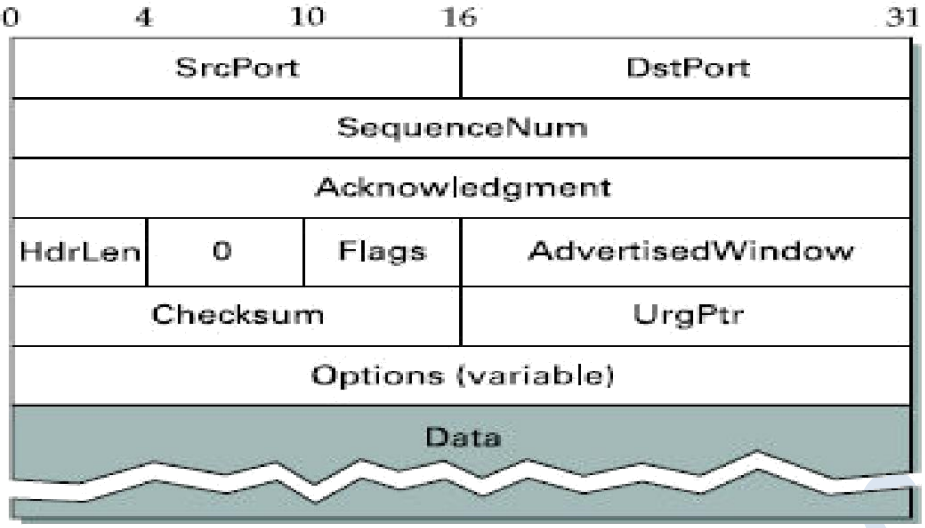
Multiplexing and demultiplexing is only based on port numbers.
For using UDP, an application must receive a port from the OS.
{
Port can be used for sending a UDP datagram to other hosts.
{
Port can be used for receiving a UDP datagram from other
hosts. A port is basically a queue (bu
er)
UDP packets, called user datagrams, have a fixed-size header of 8 bytes.
SrcPort and DstPort—Contains port number for both the sender (source) and receiver (destination) of the message.
Length—This 16-bit field defines total length of the user datagram, header plus data. The total length is less than 65,535 bytes as it is encapsulated in an IP datagram.
UDP length = IP length - IP header's length
Checksum—It is computed over pseudo header, UDP header and message content to ensure that message is correctly delivered to the exact recipient.
The *pseudoheader consists of three fields from the IP header (protocol number* i.e., *17, source and destination IP address), plus the UDP lengthfield.*
Well-known ports range from 0 to 1023 are assigned and controlled by Internet Assigned Number Authority (IANA)
Registered ports range from 1024 to 49,151 are not assigned or controlled by IANA. They can only be registered with IANA to prevent duplication.
Ephemeral (dynamic) ports range from 49,152 to 65,535 is neither controlled nor registered. It is usually assigned to a client process by the operating system.
Difference between Network and Transport layer

Network layer	*Transport layer*
The network layer is responsible for *host*-to-*host* delivery	The transport layer is responsible for *process*-to-*process* delivery of a packet
Host address is required for delivery	Host address and port number is required for delivery
Error detection is not offered	Error detection is done using checksum
Flow control is not done	Flow control is not done
Multicasting capability is not inbuilt	Multicasting is embedded into UDP

Transmission Control Protocol

Transmission Control Protocol (TCP) offers a reliable, connection-oriented, bytestream service TCP guarantees the reliable, in-order delivery of a stream of bytes.It is a full-duplex protocol TCP supports demultiplexing mechanism for process-to-process communication.
TCP has built-in congestion-control mechanism, i.e., sender is prevented from overloading the network.
It is a full-duplex protocol, meaning that each TCP connection supports a pair of byte streams, one flowing in each direction.
It also includes a flow-control mechanism.
TCP supports a demultiplexing mechanism that allows multiple application programs on any given host to simultaneously carry on a conversation with their peers.
TCP's demux key is given by the 4-tuple SrcPort, SrcIPAddr, DstPort, DstIPAddr

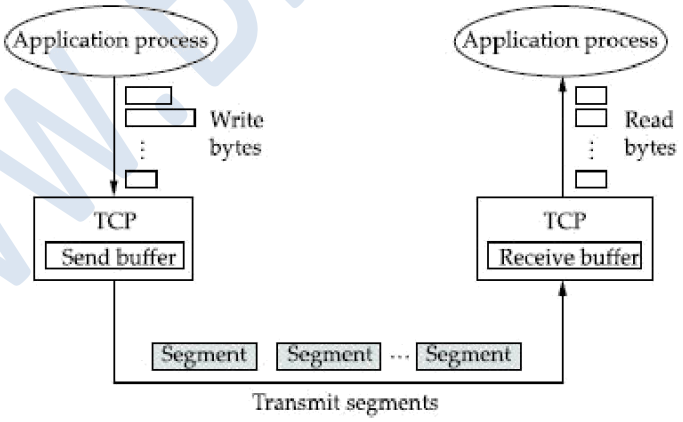
TCP HEADER FORMAT:



SrcPort and DstPort fields identify the source and destination ports. SequenceNum field contains sequence number, i.e. first byte of data in that segment. Acknowledgment defines byte number of the segment, the receiver expects next. HdrLen field specifies the number of 4-byte words in the TCP header. Flags field contains six control bits or flags. They are set to indicate:

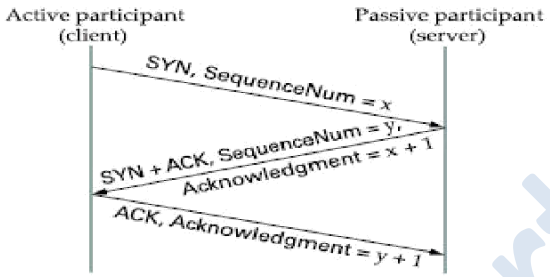
- **o *URG—indicates that the segment contains urgent data.* o *ACK—the value of acknowledgment field is valid.***
- **o *PSH—indicates sender has invoked the push operation.***
- **o *RESET—signifies that receiver wants to abort the connection.* o*SYN—synchronize sequence numbers during connection establishment.* o *FIN—terminates the connection***

AdvertisedWindow field defines the receiver window and acts as flow control. Checksum field is computed over the TCP header, the TCP data, and pseudoheade. UrgPtr field indicates where the non-urgent data contained in the segment begins. Optional information (max. 40 bytes) can be contained in the header.



A TCP connection begins with a client (caller) doing an active open (willing to accept) to a server (passive open- accept a connection (callee).
After the connection establishment two sides begin sending data. After sending data, it closes one direction of the connection.

Connection Establishment and Termination



Timeline for three-way handshake algorithm.

First, the client sends a segment to the server stating the initial sequence number it plans to use (Flags = SYN, SequenceNum = *x).*
It responds with a single segment that both acknowledges the client's sequence number (Flags = ACK, Ack = *x + 1) and states its own beginning sequence number* (Flags = SYN, SequenceNum = *y)*
Both the SYN and ACK bits are set in the Flags field of this second message.
Finally, the client responds with a third segment that acknowledges the server's sequence number (Flags = ACK, Ack = *y + 1).*
Why the client and server have to exchange starting sequence numbers at connection setup time.
It is simpler if each side simply started at some "well-known" sequence number, such as 0. TCP is complex enough that its specification includes a state-transition diagram.
The diagram shows the states involved in opening a connection and in closing a connection. Each circle denotes a state that one end of a TCP connection can find itself in.
All connections start in the CLOSED state.
As the connection progresses, the connection moves from state to state according to the arcs. Each arc is labeled with a tag of the form *event/action.*
If a connection is in the LISTEN state and a SYN segment arrives the connection makes a transition to the SYN_RCVD state.
It takes the action of replying with an ACK+ SYN segment.
Two kinds of events trigger a state transition: (1) a segment arrives from the peer (2) the local application process invokes an operation on TCP.

Adaptive Retransmission

TCP guarantees the reliable delivery of data.
It retransmits each segment if an ACK is not received in a certain period of time.
TCP sets this timeout as a function of the RTT it expects between the two ends of the connection.
Given the range of possible RTTs between any pair of hosts.
The problem is variation in RTT between the same two hosts over time.
To overcome this problem TCP uses an adaptive retransmission mechanism.

Original algorithm

Simple algorithm for computing a timeout value between a pair of hosts.

The idea is to keep a average of the RTT and then to compute the timeout as a function of this RTT.

When TCP sends a data segment, it records the time.

When an ACK arrives, TCP reads the time again.

Takes the difference between these two times as a SampleRTT.

TCP then computes an EstimatedRTT as a average between the previous estimate and this new sample. That is,

EstimatedRTT =

α × EstimatedRTT + (1−α) × SampleRTT

The parameter *α is selected to smooth the EstimatedRTT.* A small *α tracks changes in* the RTT.

A large *α is more stable but perhaps not quick enough to adapt to real changes.*

The original TCP specification recommended a setting of *α between 0.8 and 0.9.* TCP then uses EstimatedRTT to compute the timeout in a conservative way TimeOut = 2×EstimatedRTT

Karn/Partridge Algorithm

The problem was that an ACK does not really acknowledge a transmission. It actually acknowledges the receipt of data.

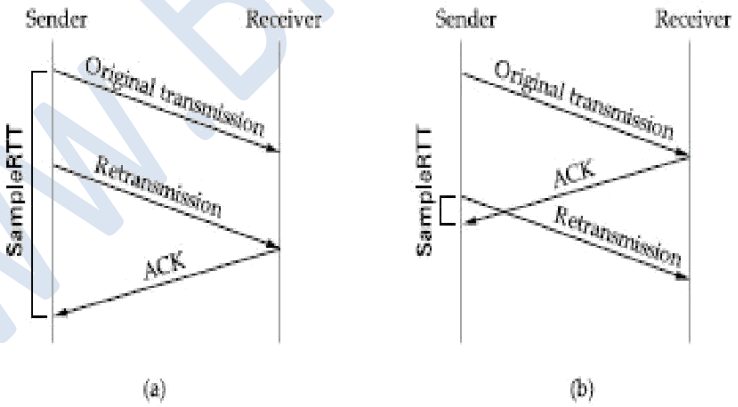
when a segment is retransmitted an ACK arrives at the sender, it is impossible to determine if this ACK is associated with the first or the second transmission.

As in Figure if we assume that the ACK is for the original transmission but it was really for the second, then the SampleRTT is too large (a),

If you assume that the ACK is for the second transmission but it was actually for the first, then the SampleRTT is too small (b). Solution is when the TCP retransmits a segment, it stops taking samples of the RTT.

It only measures SampleRTT for segments that have been sent only once. This solution is known as the Karn/Partridge algorithm.

Each time TCP retransmits, it sets the next timeout to be twice the last timeout.



Congestion control

TCP congestion control is for source to determine the capacity available in the network. AIMD

(Additive Increase / Multiplicative Decrease)

CongestionWindow (cwnd) is a variable held by the TCP source for each connection. cwnd is based on the perceived level of congestion.

TCP maintains a new state variable for each connection, called CongestionWindow.

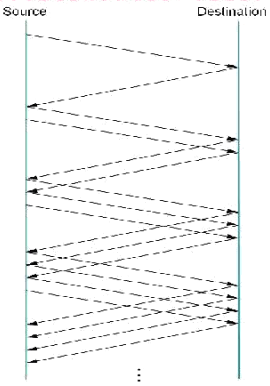
Cwnd used by the source to limit how much data it is allowed to have in transit at a given time.

Additive Increase

Additive Increase is a reaction to perceived available capacity.

Linear Increase basic idea:: For each "cwnd's worth" of packets sent, increase cwnd by 1 packet. In practice, cwnd is incremented fractionally for each arriving ACK.

```
increment = MSS x (MSS /cwnd)
cwnd = cwnd + increment
```



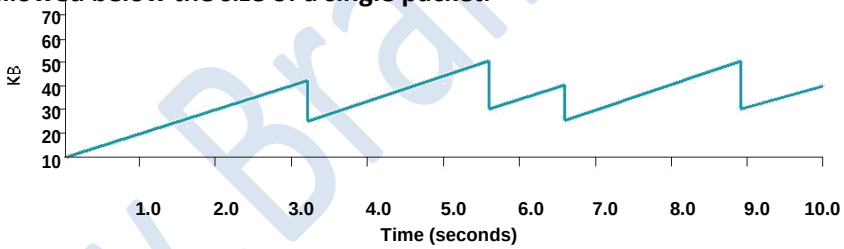
Multiplicative Decrease

The key assumption is that a dropped packet and the resultant timeout are due to congestion at a router or a switch.

Multiplicate Decrease:: TCP reacts to a timeout by halving cwnd.

Although cwnd is defined in bytes, the literature often discusses congestion control in terms of packets (or more formally in MSS == Maximum Segment Size).

cwnd is not allowed below the size of a single packet.



Typical TCP

Slow Start

The source starts with cwnd = 1.

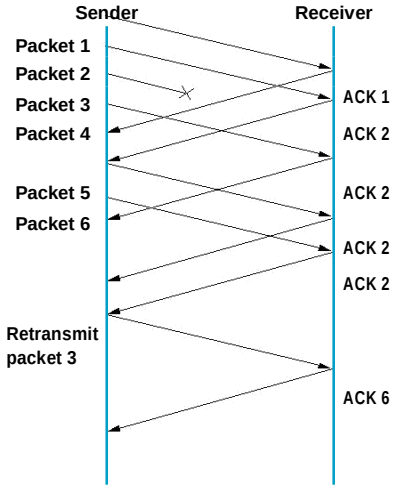
Every time an ACK arrives, cwnd is incremented. cwnd is effectively doubled per RTT "epoch". Two slow start situations:

- At the very beginning of a connection {cold start}.
- When the connection goes dead waiting for a timeout to occur (i.e, the advertized window goes to zero!)

Fast Retransmit

Coarse timeouts remained a problem, and Fast retransmit was added with TCP Tahoe.

Since the receiver responds every time a packet arrives, this implies the sender will see duplicate ACKs.



Basic Idea:: *use duplicate ACKs to signal lost packet.*

Congestion-Avoidance Mechanisms

Three different congestion-avoidance mechanisms. The first two take a similar approach. The third mechanism is very different from the first two.

1. DECbit

2. Random Early Detection (RED)

3.Source-Based Congestion Avoidance

DECbit

The mechanism developed for use on the Digital Network Architecture (DNA), a connectionless network with a connection-oriented transport protocol.

Split the responsibility for congestion control between the routers and the end nodes. Each router monitors the load it experiences and notifies the end nodes when congestion is about to occur.

A single congestion bit is added to the packet header. A router sets this bit if its average queue length is greater than or equal to 1.

Random Early Detection

A second mechanism, called *random early detection (RED), is similar to the DECbit.*

RED *implicitly notifies the* source of congestion by dropping one of its packets. Sending a congestion notification message to the source.

The difference between RED and DECbit is how RED decides when to drop a packet and what packet it decides to drop.

Consider a simple FIFO queue. Rather than wait the queue to become full and then drop each arriving packet, decide to drop each arriving packet with some *drop probability whenever the queue* length exceeds some *drop level.*

Second, RED has two queue length thresholds that trigger certain activity: MinThreshold and MaxThreshold. When a packet arrives at the gateway, RED compares the current AvgLen with these two thresholds.

if AvgLen ≤ MinThreshold –→ queue the packet

if MinThreshold < *AvgLen* < *MaxThreshold* –→ calculate probability

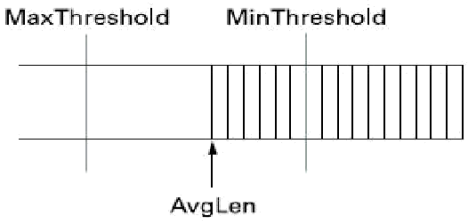
P –→ drop the arriving packet with probability P

if MaxThreshold ≤ AvgLen –→ drop the arriving packet

If the average queue length is smaller than the lower threshold, no action is taken.

If the average queue length is larger than the upper threshold, then the packet is always dropped.

If the average queue length is between the two thresholds, then the newly arriving packet is dropped with so.me probability P



RED thresholds on a FIFO queue.

Source-Based Congestion Avoidance

The general idea of these techniques is to watch for some sign from the network that congestion will happen soon.

For example, the source might notice that as packet queues build up there is a measurable increase in the RTT for each successive packet it sends.

Congestion window is normally minimum and maximum RTTs increases in TCP.

Every two round-trip delays the algorithm checks current RTT is greater than the average of the minimum and maximum RTTs.

If it is, then the algorithm decreases the congestion window by one-eighth.

A second algorithm does something similar. The decision as to whether or not to change the current window size is based on changes to both the RTT and the window size.

The window is adjusted once every two round-trip delays based on the product

(CurrentWindow−OldWindow) ×(CurrentRTT− OldRTT)

result positive, the source decreases the window size by one-eighth;

Result negative or 0, the source increases the window by one maximum packet

size Diff = ExpectedRate – ActualRate

If Diff > X, decrease congestion window

If Diff < Y, increase congestion window

How would you choose the thresholds X and Y?

UNIT-5

1. Write short notes on the following:

i. Telnet
ii. E-mail
iii. HTTP (World Wide Web)

i. Telnet

Telnet is a user command and an underlying TCP/IP protocol for accessing remote computers. Through Telnet, an administrator or another user can access someone else's computer remotely. On the Web, HTTP and FTP protocols allow you to request specific files from remote computers, but not to actually be logged on as a user of that computer. With Telnet, you log on as a regular user with whatever privileges you may have been granted to the specific application and data on that computer.

Telnet is a network protocol used on the Internet or local area networks to provide a bidirectional interactive text-oriented communication facility using a

virtual terminal connection. User data is interspersed in-band with Telnet control information in an 8-bitbyte oriented data connection over the Transmission Control Protocol (TCP).

Telnet provided access to a command-line interface (usually, of an operating system) on a remote host. Most network equipment and operating systems with a TCP/IP stack support a Telnet service for remote configuration (including systems based on Windows NT).

Telnet is a client-server protocol, based on a reliable connection-oriented transport. Typically this protocol is used to establish a connection to Transmission Control Protocol(TCP) port number 23, where a Telnet server application (telnetd) is listening. Telnet, however, predates TCP/IP and was originally run over Network Control Program (NCP) protocols.

ii. E-mail

The fulform of e-mail is **Electronic Mail.**

It is a system for sending and receiving messages electronically over a computer network, as between personal computers. A message or messages sent or received by such a system.

Email is used as a means of communication between 1 or more parties. It is often more efficient than traditional mail which has a longer transit between the sender and the receiver(s).

There is many advantages and some disadvantages of e-mail.

Advantages of email are

1. email is effective in providing quick answers to yes and no, type questions.
2. Email is effective in finding the right person in an organisation or company to answer your question.

3. Email is good to make appointments for busy people. 4. Email can distribute information quickly to many people for the time it takes to email one person. And so on…

The disadvantages are:

1. Email can become time consuming for answering complicated questions and misunderstandings can arise because cultural differences in the interpretation of certain words. The telephone, is much better for providing detailed answers or if you feel that the question is not absolutely clear.

2. Email can compromise the security of an organisation because sensitive information can be easily distributed accidently or deliberately. Email should be entrusted to well trained and trusted staff members. 3. Email can become impersonal or misunderstood.

iii. HTTP (WWW)

The **Hypertext Transfer Protocol** (**HTTP**) is a networking protocol for distributed, collaborative, hypermedia information systems, HTTP is the foundation of data communication for the World Wide Web.

The World Wide Web, abbreviated as WWW and commonly known as the Web, is a system of interlinked hypertext documents accessed via the Internet. With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks. Using concepts from earlier hypertext systems, English engineer and computer scientist Sir Tim Berners-Lee, now the Director of the World Wide Web Consortium, wrote a proposal in March 1989 for what would eventually become the World Wide Web. At CERN in Geneva, Switzerland, Berners-Lee and Belgian computer scientist Robert Cailliau proposed in 1990 to use "HyperText ... to link and access information of various kinds as a web of nodes in which the user can browse at will",and publicly introduced the project in December. "The World-Wide Web (W3) was developed to be a pool of human knowledge, and human culture, which would allow collaborators in remote sites to share their ideas and all aspects of a common project.

There are several applications called Web browsers that make it easy to access the World Wide Web; Two of the most popular being Firefox and Microsoft's Internet Explorer.

2. Discuss in detail about DNS and its frame format.

Short for ***D**omain **N**ame **S**ystem* (or ***S**ervice* or ***S**erver*), an Internet service that translates *domain names* into IP addresses. Because domain names are alphabetic, they're easier to remember. The Internet however, is really based on IP addresses. Every time you use a domain name, therefore, a DNS service must translate the name into the corresponding IP address. For example, the domain name *www.example.com* might translate to *198.105.232.4*.

Some basic terminology, First, a *name space* defines the set of possible names.A name space can be either *flat* (names are not divisible into components) or *hierarchical* (Unix file names are the obvious example). Second, the naming system maintains a collection of *bindings* of names to values. The value can be anything we want the naming system to return when presented with a name; in many cases it is an address. Finally, a *resolution mechanism* is a procedure that, when invoked with a name, returns the corresponding value. A *name server* is a specific implementation of a resolution mechanism that is available on a network and that can be queried by sending it a message.

Domain Hierarchy:

DNS names are processed from right to left and use periods as the separator. (Although they are "processed" from right to left, humans still "read" domain names from left to right.) An example domain name for a host is cicada.cs.princeton.edu. Notice that we said domain names are used to name Internet "objects." What we mean by this is that DNS is not strictly used to map host names into host addresses. It is more accurate to say that DNS maps domain names into values.

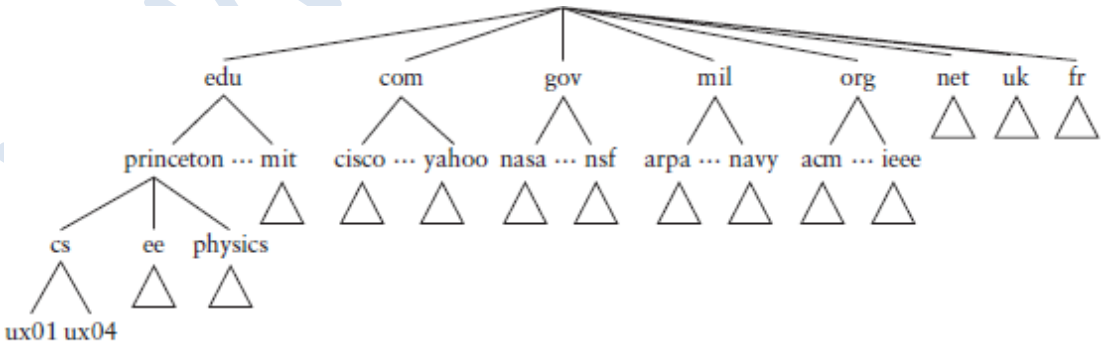


Figure 9.2 Example of a domain hierarchy.

Name Servers:

The first step is to partition the hierarchy into subtrees called *zones*. Each zone can be thought of as corresponding to some administrative authority that is responsible for that portion of the hierarchy. For example, the top level of the hierarchy forms a zone that is managed by the NIC.

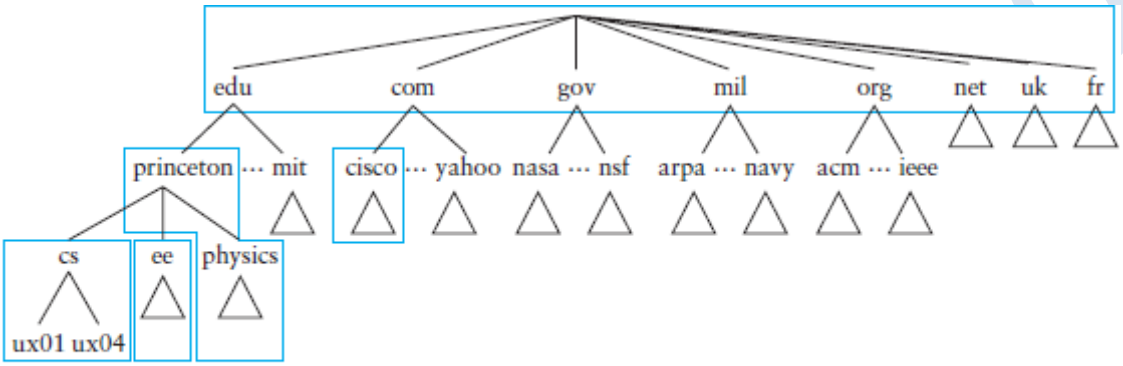


Figure 9.3 Domain hierarchy partitioned into zones.

3.

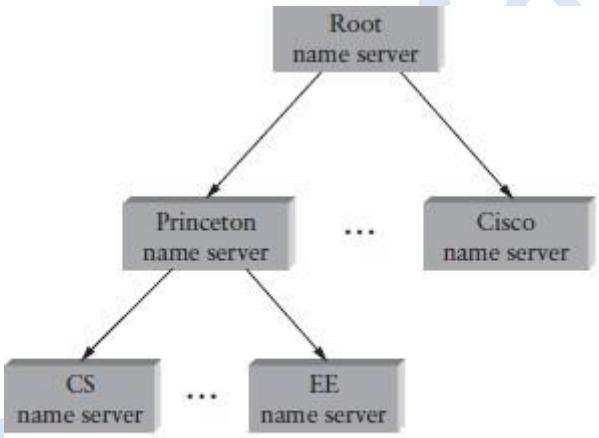


Figure 9.4 Hierarchy of name servers.

Each name server implements the zone information as a collection of *resource records*. In essence, a resource record is a name-to-value binding, or more specifically, a 5-tuple that contains the following fields: _ Name, Value, Type, Class, TTL _

The Name and Value fields are exactly what you would expect, while the Type field

specifies how the Value should be interpreted. For example, Type = A indicates that the Value is an IP address. Thus, A records implement the name-to-address mapping we have been assuming. Other record types include

- NS: The Value field gives the domain name for a host that is running a name server that knows how to resolve names within the specified domain.
- CNAME: The Value field gives the canonical name for a particular host; it is used to define aliases.
- MX: The Value field gives the domain name for a host that is running a mail server that accepts messages for the specified domain.

Name Resolution:

The client first sends a query containing this name to the root server. The root server, unable to match the entire name, returns the best match it has—the NS record for princeton.edu. Theserver also returns all records that are related to this record, in this case, the A record for cit.rinceton.edu. The client, having not received the answer it was after, next sends the same query to the name server at IP host 128.196.128.233. This server also cannot match the whole name, and so returns the NS and corresponding A records for the cs.princeton.edu domain. Finally, the client sends the same query as before to theserver at IP host 192.12.69.5, and this time gets back the A record for cicada.cs.princeton.edu.

4. Explain SMTP in detail.

As a third example, MIME defines an application type, where the subtypes correspond to the output of different application programs (e.g., application/postscript and application/ msword).

MIME also defines a multipart type that says how a message carrying more than one data typeis structured. This is like a programming language that defines both base types (e.g., integers andfloats) and compound types (e.g., structures and arrays).

It is important that email messages contain only ASCII, because they might pass through a number of intermediate systems (gateways, as described below) that assume all email is ASCII and would corrupt the message if it contained non-ASCII characters. To address this issue, MIME uses a straightforward encoding of binary data into the ASCII character set. The

encoding is called base64. The idea is to map every three bytes of the originalbinary data into four ASCII characters.

Putting this all together, a message that contains some plain text, a JPEG image, and a PostScript file would look something like this:

MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="------ 417CA6E2DE4ABCAFBC5"
From: Alice Smith <Alice@cisco.com>
To: Bob@cs.Princeton.edu
Subject: promised material
Date: Mon, 07 Sep 1998 19:45:19 -0400 ---
------417CA6E2DE4ABCAFBC5
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
Bob,
Here's the jpeg image and draft report I promised. --
Alice
---------417CA6E2DE4ABCAFBC5
Content-Type: image/jpeg
Content-Transfer-Encoding: base64
. . . *unreadable encoding of a jpeg figure* ---
------417CA6E2DE4ABCAFBC5
Content-Type: application/postscript; name="draft.ps"
Content-Transfer-Encoding: 7bit
. . . *readable encoding of a PostScript document*

Message Transfer:

SMTP—the protocol used to transfer messages from one host to another. To place SMTP inthe right context, we need to identify the key players. First, users interact with a *mail reader* when they compose, file, search, and read their email.

There are countless mail readers available, just like there are many Web browsers to choose from. In fact, mostWeb browsers now include a mail reader. Second, there is a *mail daemon* (or process) running on each host. You can think of this process as playing

the role of a post office: Mail readers give the daemon messages they want to send to other users, the daemon uses SMTP running over TCP to transmit the message to a daemon running on another machine, and the daemon puts incoming messages into the user's *mailbox*.

The sendmail program on a sender's machine establishes an SMTP/TCP connection to the send mail program on the recipient's machine, in many cases the mail traverses one or more *mail gateways* on its route from the sender's host to the receiver's host. Like the end hosts, these gateways also run a send mail process. It's not an accident that these intermediate nodes are called "gateways" since their job is to store and forward email messages, much like an "IP gateway" (which we have referred to as a router) stores and forwards IP datagrams.

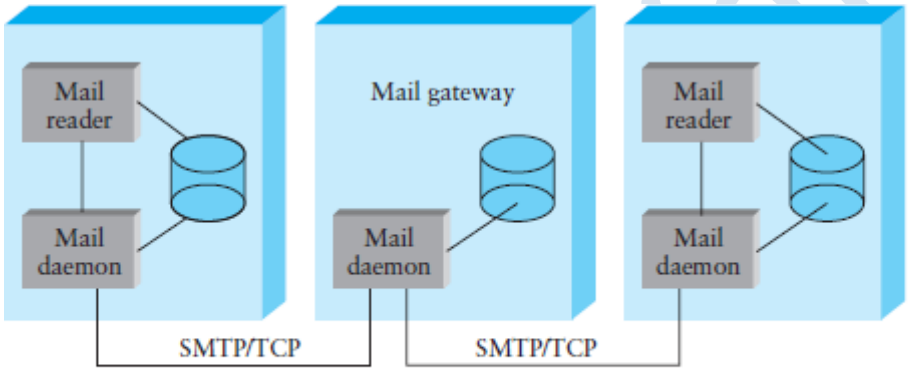


Figure 9.6 Sequence of mail gateways store and forward email messages.

SMTP is best understood by a simple example. The following is an exchange between sending host cs.princeton.edu and receiving host cisco.com. In this case, user Bob at Princeton is trying to send mail to users Alice and Tom at Cisco. The lines sent by cs.princeton.edu are shown in black and the lines sent by cisco.com are shown in green. Extra blank lines have been added to make the dialog more readable.

HELO cs.princeton.edu

250 Hello daemon@mail.cs.princeton.edu [128.12.169.24]

MAIL FROM:<Bob@cs.princeton.edu>

250 OK

RCPT TO:<Alice@cisco.com>

250 OK

RCPT TO:<Tom@cisco.com>

550 No such user here

DATA

354 Start mail input; end with <CRLF>.<CRLF>

Blah blah blah...

...etc. etc. etc.

9.2 Traditional Applications 649

<CRLF>.<CRLF>

250 OK

QUIT 221

<CRLF>.<CRLF>

250 OK

QUIT

221 Closing connection

What actually happens is that the mail daemon parses the message to extract the information

it needs to run SMTP. The information it extracts is said to form an *envelope* for the message. The SMTP client uses this envelope to parameterize its exchange with the SMTP server.

5. Explain the various processes involved after typing the URL in the task bar.

Your browser, if it doesn't already know, will ask your OS's DNS system what the address (IP address) of the host ("www.google.com," for example) is. If your OS doesn't know, it will query third-party DNS servers (those of your ISP, for example).

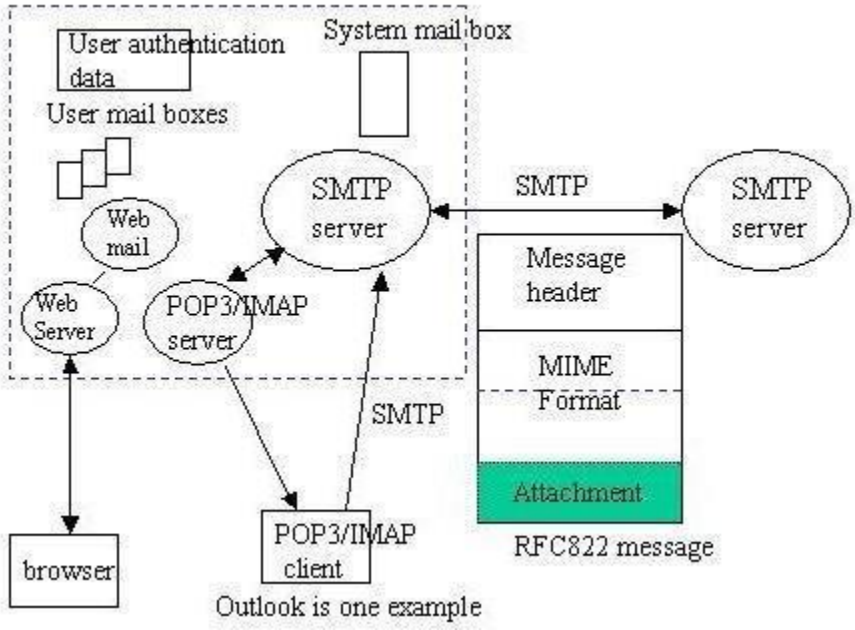
Once an address is obtained, your web browser establishes a TCP/IP socket connection, typically on TCP port 80, with the web server at the IP address it resolved the host name to.

Once your browser has established this connection, it sends an HTTP GET request to the web server for whatever resource was requested in your URL. For example, http://www.google.com/would mean you'd send a '/' request to whatever web server is at www.google.com.

The web server will then, typically, respond to the request with an HTTP response, typically containing HTML. Your web browser downloads this response.

Your web browser renders the HTML. It may need to send additional requests for any scripts, stylesheets, images, or other resources linked to in the HTML.

6. Explain the architecture of E-mail system.



E-mail system normally consists of two sub systems

1. the user agents

2. the message transfer agents

The user agents allow people to read and send e-mails. The message transfer agents move the messages from source to destination. The user agents are local programs that provide a command based, menu-based, or graphical method for interacting with e-mail system. The message transfer agents are daemons, which are processes that run in background. Their job is to move datagram e-mail through system.

A key idea in e-mail system is the distinction between the envelope and its contents. The

envelope encapsulates the message. It contains all the information needed for transporting the message like destinations address, priority, and security level, all of which are distinct from the message itself.

The message transport agents use the envelope for routing. The message inside the envelope consists of two major sections:

The Header:

The header contains control information for the user agents. It is structured into fields such as summary, sender, receiver, and other information about the e-mail.

· Body:

The body is entirely for human recipient. The message itself as unstructured text; sometimes containing a signature block at the end

2 Header format

The header is separated from the body by a blank line.

consists of following fields

· From: The e-mail address, and optionally name, of the sender of the message.

· To: one or more e-mail addresses, and optionally name, of the receiver's of the message.

· Subject: A brief summary of the contents ofthe message.

· Date: The local time and date when the message was originally sent.

E-mail system based on RFC 822 contains the message header as shown in figure 8.2. The figure gives the fields along with their meaning.

The fields in the message header of E-mail system based on RFC 822 related to message transport are given in figure 8.3. The figure gives the fields along with their meaning.

3 User agents:

It is normally a program and sometimes called a mail reader. It accepts a variety of commands for composing, receiving, replying messages as well as manipulating the mail boxes. Some user agents have a fancy menu or icon driven interfaced that require a mouse where as others are one character commands from keyboard. Functionally these are same. Some systems are menu or icon driven but also have keyboard shortcuts.

To send an e-mail, user provides the message, the destination address and possibly some other parameters. Most e-mail system supports mailing lists.

Example: Reading e-mail

When a user is started up, it looks at the user's mailbox for incoming e-mail before displaying anything on the screen. Then it announces the number of messages in the mailbox or displays a one-line summary of each e-mail and wait for a command.

The display may look something like that is shown in figure 8.4. Each line of the display contains several fields extracted from the envelope or header of the corresponding message. In a simple e-mail system, the choice of fields is built into the program. In more sophisticated system, user can specify which fields are to be displayed by providing a user profile.

Referring to the display it contains following fields

1. Message number: it is serial number of the message. It can be displayed from the most currently received messages or vice versa.

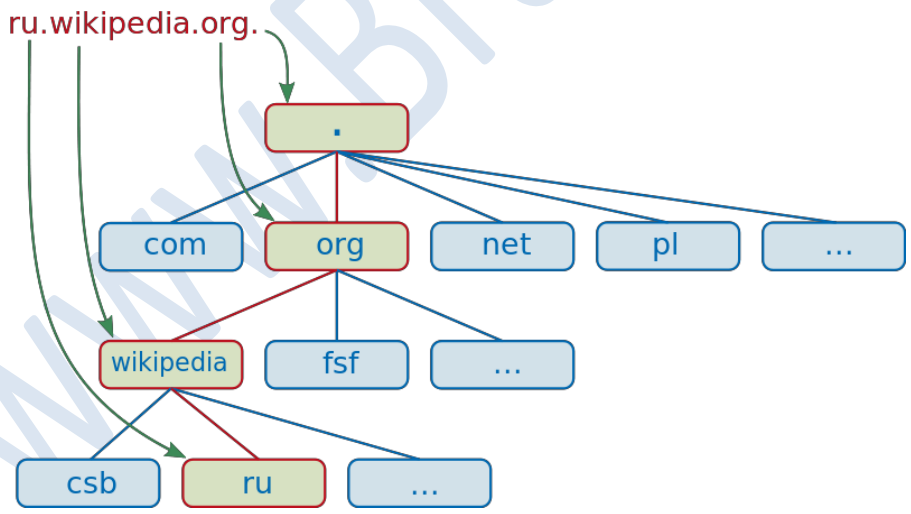
2. Flags: contains K means the message is not new, A means the message is already read and F means the message has been forwarded to someone else.

3. size of the message: indicates the length of the message

4. source of the message: originator's address

5. subject: gives a brief summary of what the message is about.

7. With a relevant example explain how the domain space is divided.



A **domain name** is an identification string that defines a realm of administrative autonomy, authority, or control on the Internet. Domain names are formed by the rules and procedures of the Domain Name System (DNS). Any name registered in the DNS is a domain name.

Domain names are used in various networking contexts and application-specific naming and

addressing purposes. In general, a domain name represents an Internet Protocol (IP) resource, such as a personal computer used to access the Internet, a server computer hosting a web site, or the web site itself or any other service communicated via the Internet.

Domain names are organized in subordinate levels (subdomains) of the DNS root domain, which is nameless. The first-level set of domain names are the top-level domains (TLDs), including the generic top-level domains (gTLDs), such as the prominent domains com, info, net and org, and the country code top-level domains (ccTLDs). Below these top-level domains in the DNS hierarchy are the second-level and third-level domain names that are typically open for reservation by end-users who wish to connect local area networks to the Internet, create other publicly accessible Internet resources or run web sites. The registration of these domain names is usually administered by domain name registrars who sell their services to the public.

A fully qualified domain name (FQDN) is a domain name that is completely specified in the hierarchy of the DNS, having no omitted parts.

Domain names are usually written in lowercase, although labels in the Domain Name System are case-insensitive.

A domain name consists of one or more parts, technically called *labels*, that are conventionally concatenated, and delimited by dots, such as *example.com*.

The right-most label conveys the top-level domain; for example, the domain name*www.example.com* belongs to the top-level domain *com*.
The hierarchy of domains descends from the right to the left label in the name; each label to the left specifies a subdivision, or subdomain of the domain to the right.