Transfer blocks of information called packets.

Packets are the connectivity between two users for communication.

Two perspectives:
1) External View: Related to service provided to transport layer.
2) Internal View: Concerned with Topology, links, switches and routers.

Two approaches:
1) Virtual Circuits
2) Datagrams
ROUTING AND FORWARDING

Routing is
• to know about other routers
• where each of them is located
• which networks they are connected to

Forwarding is
• finding out one’s own neighboring routers
• forward a specified packet to its nearest neighbor to make it reach to a given destination
ROUTING AND FORWARDING

Routing table contains information about nearest router for given destination

Routing algorithms decide placement of routers

Virtual circuit is mechanism used for connection oriented forwarding

Datagrams are units of data sent in connectionless forwarding
NETWORK LAYER DUTIES

Handling accounting for usage of network resources
Devise and implement mechanisms of identifying each machine uniquely
Implement connectionless or connection-oriented forwarding
Multiplexing and demultiplexing the transport layer and the data link layer jobs
Routing & Forwarding
Scheduling & Quality of Service
Handle Addressing
TWO DIFFERENT TYPES OF ROUTING

Collection of networks organized by a single party is known as an autonomous system (AS)

Exterior routing is across AS
- BGP (Border Gateway Protocol)

Interior routing is within AS
- Distance Vector
- Link state
- AODV
REQUIREMENTS/GOALS OF A GOOD ROUTING ALGORITHMS

Packets continuously forwards them to their destinations in minimum possible time.
Impartial to all the nodes.
Simple enough to be implemented
Should not oscillate frequently to make packet forwarding erratic.
Good path to receiver is recorded.
Fault tolerance: Must continue functioning irrespective of nodes and links going up and down.
Must not be bogged by increasing or decreasing number of nodes.
Fast enough to reflect changes in network topology in real time.
Speed is maintained.

Other requirements:
- Dynamism and flexibility
- Performance
- Robustness
<table>
<thead>
<tr>
<th>Issue</th>
<th>Datagram subnet</th>
<th>Virtual-circuit subnet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circuit setup</td>
<td>Not needed</td>
<td>Required</td>
</tr>
<tr>
<td>Addressing</td>
<td>Each packet contains the full source and destination address</td>
<td>Each packet contains a short VC number</td>
</tr>
<tr>
<td>State information</td>
<td>Routers do not hold state information about connections</td>
<td>Each VC requires router table space per connection</td>
</tr>
<tr>
<td>Routing</td>
<td>Each packet is routed independently</td>
<td>Route chosen when VC is set up; all packets follow it</td>
</tr>
<tr>
<td>Effect of router failures</td>
<td>None, except for packets lost during the crash</td>
<td>All VCs that passed through the failed router are terminated</td>
</tr>
<tr>
<td>Quality of service</td>
<td>Difficult</td>
<td>Easy if enough resources can be allocated in advance for each VC</td>
</tr>
<tr>
<td>Congestion control</td>
<td>Difficult</td>
<td>Easy if enough resources can be allocated in advance for each VC</td>
</tr>
</tbody>
</table>
STRUCTURE OF PACKET SWITCH

Packet switch components

- Input ports: Port 1, Port 2, ..., Port N
- Routing processor
- Switching fabric
- Output ports: Port 1, Port 2, ..., Port N
STRUCTURE OF PACKET SWITCH

Input & Output Ports: They are generally connected.

Line Card: Made up of various chipsets. Performs task such as lookup and scheduling. Has several input output ports. High speed link assures complete utilization of line. Implements network layer functions. Does jobs like timing, line coding, framing, error checking, addressing, etc. Handles broadcast networks. Has routing tables. Lookup is performed to search through the tables. Also has buffers for queuing.


Interconnection Fabric: It transfers packets between line cards. It may get bottleneck because of presence of multiple high speed line cards. Types: 1) Bus 2) Cross Bar
CLASSIFICATION OF ROUTING ALGORITHM

- Routing Algorithm
  - Responsiveness
    - Static
    - Dynamic / Adaptive
  - Network Control
    - Centralized
    - Distributed
HIERARCHICAL ROUTING
Hierarchical Routing

![Diagram of hierarchical routing with regions and lines]

**Table:**

<table>
<thead>
<tr>
<th>Destination</th>
<th>Line</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>1</td>
</tr>
<tr>
<td>Region 2</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>Region 3</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>Region 4</td>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>Region 5</td>
<td>C</td>
<td>4</td>
</tr>
</tbody>
</table>

*Figure 12.17 Hierarchical routing*
FLOODING

Incoming packet is forwarded to all the ports other than the one it is received from.

Useful when routing path to destination does not exist.

Generates exponential growth rate and unnecessary network traffic and load.

As a solution we can implement TTL-Time to Live, Loop avoidance & Sequence#.
To reach from A to F, shortest path is selected.

A → C
A → C → E
A → C → E → D
A → C → E → D → F
BELLMAN FORD ALGORITHM

Also called Ford Fulkerson Algorithm.

Each neighbor of node knows path to destination.

Take advice from each neighbor and select the shortest path.
DIJKSTRA’S ALGORITHM

(a) Diagonal view of a graph showing costs and paths.

(b) Vertex 0 is selected as the start vertex.

(c) Vertex 5 is selected as the second vertex.

(d) Vertex 7 is selected as the third vertex.

(e) Vertex 9 is selected as the forth vertex.

(f) Vertex 13 is selected as the fifth vertex.
DISTANCE VECTOR ROUTING ALGORITHM

So far we have studied Static Routing Algorithms.

But practically dynamic Routing Algorithms are used.

Following two are Dynamic Routing Algorithms:
- 1. Distance Vector Routing Algorithm.
- 2. Link State Routing Algorithm.

Distance Vector Routing Algorithm:

At each step within a router:
- Get routing tables from neighbours
- Compute distance to neighbours
- Compute new routing table

1. Router transmits its distance vector to each of its neighbors.

2. Each router receives and saves the most recently received distance vector from each of its neighbors.

3. A router recalculates its distance vector when:
   a. It receives a distance vector from a neighbor containing different information than before.
   b. It discovers that a link to a neighbor has gone down (i.e., a topology change).

The DV calculation is based on minimizing the cost to each destination.

The distance vector routing algorithm is sometimes called by other names, the distributed Bellman-Ford routing algorithm and the Ford-Fulkerson algorithm.
THE SUBNET FOR ROUTING AND J SENDER
### PARTIAL ROUTING TABLE FOR J

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>G</td>
<td>1</td>
</tr>
<tr>
<td>A</td>
<td>I</td>
<td>2</td>
</tr>
<tr>
<td>B</td>
<td>I</td>
<td>2</td>
</tr>
<tr>
<td>X</td>
<td>I</td>
<td>2</td>
</tr>
<tr>
<td>V</td>
<td>G</td>
<td>1</td>
</tr>
<tr>
<td>W</td>
<td>G</td>
<td>1</td>
</tr>
<tr>
<td>Q</td>
<td>C</td>
<td>3</td>
</tr>
<tr>
<td>……</td>
<td>……</td>
<td>……</td>
</tr>
</tbody>
</table>
CONSTRUCTING ROUTING TABLES
ANOTHER EXAMPLE
### Constructing Routing Tables
#### Another Example

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>3</td>
<td>A</td>
<td>A</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>6</td>
<td>B</td>
<td>B</td>
<td>6</td>
</tr>
<tr>
<td>K</td>
<td>A</td>
<td>10</td>
<td>K</td>
<td>A</td>
<td>12</td>
</tr>
<tr>
<td>H</td>
<td>D</td>
<td>8</td>
<td>H</td>
<td>H</td>
<td>2</td>
</tr>
<tr>
<td>J</td>
<td>D</td>
<td>12</td>
<td>J</td>
<td>H</td>
<td>6</td>
</tr>
<tr>
<td>G</td>
<td>D</td>
<td>11</td>
<td>G</td>
<td>G</td>
<td>5</td>
</tr>
</tbody>
</table>
COUNT TO INFINITY PROBLEM:

Drawback of Distance Vector Routing:

- Count to Infinity Problem:
- It reacts rapidly to good news,
- But, leisurely to bad news.
- Updates value fast when neighbor is down, but not when neighbor is again up. How?
- Lie to neighbour about distance if routing via neighbour
- The core of the problem is that when A tells B that it has a path to D, B has no way of knowing whether it itself(B) is on the path? This is how problem is created.
- It does not take bandwidth into account.
- Take too long to converge changes in one node to all other nodes.

Solution?

Split Horizon Hack. Lets see what it is.
COUNT TO INFINITY PROBLEM
ANOTHER EXAMPLE
COUNT TO INFINITY: C’S AND B’S ROUTING TABLES BEFORE D IS DOWN

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>C</td>
<td>4</td>
</tr>
</tbody>
</table>
### A’s Routing Table and C’s Modified Routing Table When D is Down

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
<td>6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
<td>6</td>
</tr>
</tbody>
</table>
### SOLUTION BY LYING: THE SPLIT HORIZON HACK

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>C</td>
<td>infinity</td>
</tr>
</tbody>
</table>

- Rule: If B ask A, distance from A to D, and if B lies in the path, A should reply to B that A has path to D equal to infinite.
- This would avoid trapping of B into the loop of infinity.
- The best remedy is not to enter into Count to Infinity problem.
- This is called Split Horizon Hack.
LINK STATE ROUTING ALGORITHM

Each router must do the following:

1. Discover its neighbors, learn their network address.
2. Measure the delay or cost to each of its neighbors.
3. Construct a packet telling all it has just learned.
4. Send this packet to all other routers.
5. Compute the shortest path to every other router.

A complete topology is developed. Then Dijkstra’s Algorithm can be used to compute the shortest path.

Following 5 steps are followed to implement it.

1. Learning about the Neighbors
4. Distributing the Link State Packets.
STEP1: LEARNING ABOUT THE NEIGHBORS

(a) Nine routers and a LAN. (b) A graph model of (a).
Step 1: Learning about the Neighbours:
- Upon boot of router,
  - Send HELLO packet on each point-to-point line
  - Routers are supposed to send reply with a globally unique name

Step 2: Measuring the Line Cost:
- Measure round-trip delay using ECHO Packet and wait for its reply
- Take load into account? Yes. Arguments both ways: when choice is given to router having same number of hops from S to D.
  - Yes! preference for unloaded line as shortest path.
  - No! where oscillations are possible.
- Better Solution? Distribute Load over multiple lines.
2. MEASURING LINE COST

A subnet in which the East and West parts are connected by two lines.
Step 3: Building Link State Packets:

- Packet containing:
  - Identity of sender
  - Sequence number + age
  - For each neighbour:
    - name + distance
- When to build the link state packets?
  - Periodically
    - when significant events occur

See next figure.
3. BUILDING LINK STATE PACKETS

(a) A subnet. (b) The link state packets for this subnet.
Step 4: Distributing Link State Packets:

Distributing link state packets

- Trickiest part of algorithm
  - Arrival time for packets different
  - How to keep consistent routing tables?

- Basic algorithm
  - Flooding +
  - Sequence number (in each packet) to limit duplicates.

- Manageable problems
  - Wrap around of sequence numbers results to wrong data. Solution? Use 32 bit sequence number.
  - Wrong sequence number used in case of:
    - lost in case of crash
    - Corrupted data transmitted.

- Solution? include the age of each packet after the sequence number and decrement it once per second. When the age hits zero, the information from that router is discarded.
  - duplicates are discarded
  - Old packets are thrown out
Step 5: Computing new routes:

- With a full set of link state packets, a router can:
  - Construct the entire subnet graph
  - Run Dijkstra’s algorithm to compute the shortest path to each destination

Problems for large subnets

- Memory to store data
- Compute time for developing these tables.

Usage:

- IS-IS protocol (Intermediate System, Intermediate System)
  - Designed for DECnet (Digital Equipment Corporation network protocol suite), adopted by ISO (International Standardization Organization), used still in Internet.
  - Supports multiple network layer protocols

- OSPF (Open Shortest Path First) protocol used in Internet

Common features:

- Self-stabilizing method of flooding link state updates
- Concept of a designated router on a LAN
- Method of computing and supporting path splitting and multiple metrics.
- Useful in Multi Protocol Environment.
LINK STATE ALGORITHM
THE GRAPH CONSTRUCTED FROM ALL THE PACKETS
E’S ROUTING TABLE

<table>
<thead>
<tr>
<th>Network</th>
<th>Next router</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>8</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>D</td>
<td>4</td>
</tr>
<tr>
<td>K</td>
<td>B</td>
<td>15</td>
</tr>
<tr>
<td>H</td>
<td>D</td>
<td>6</td>
</tr>
<tr>
<td>G</td>
<td>D</td>
<td>9</td>
</tr>
<tr>
<td>J</td>
<td>D</td>
<td>10</td>
</tr>
</tbody>
</table>
THE ISSUE WHEN LAN IS A PART
SOLUTION: DEPICT LAN AS A NODE
<table>
<thead>
<tr>
<th><strong>Link State</strong></th>
<th><strong>Distance Vector</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Link states algorithm is an algorithm <strong>using global information</strong></td>
<td>the distance vector algorithm is <strong>iterative, asynchronous, and distributed</strong></td>
</tr>
<tr>
<td>each node <strong>talks with all other nodes</strong>, but tell them only the cost of its directly comparison of some of their attribute</td>
<td>each node <strong>talks to only its directly connected neighbors</strong>, but provides its neighbor with least cost estimates from itself to all the nodes.</td>
</tr>
<tr>
<td><strong>Message complexity</strong>: With link state, every node has to keep the information about the cost of each link within the network.</td>
<td><strong>Message complexity</strong>: with distance vector algorithm, message is exchanged between two hosts which are directly connected to each other.</td>
</tr>
<tr>
<td>very times, if any of the link cost is changed, all the nodes are <strong>updated</strong>.</td>
<td>change of cost in the link which is belong to the least cost path for one of the nodes, the DV algorithm will update the new value. But if the change doesn’t belong to the least cost part between 2 hosts, there will <strong>no updating</strong>.</td>
</tr>
<tr>
<td><strong>Speed of convergence</strong>: can converge faster in comparison of later.</td>
<td><strong>Speed of convergence</strong>: can converge slowly and have routing loops while the algorithm is converging.</td>
</tr>
<tr>
<td>Such probability is less.</td>
<td>DV algorithm also suffers from the <strong>count to infinity</strong> problem.</td>
</tr>
<tr>
<td><strong>Robustness</strong>: For LS, when a router is down, it can broadcast a wrong cost for the closest one. LS node is computing for its own forwarding table and other node do the calculation for themselves. <strong>Better than DV.</strong></td>
<td><strong>Robustness</strong>: DV, the wrong least cost path can be passed to more than one or all of the node so the wrong calculation will be process in the entire net work. This problem of DV is much <strong>worse than LS algorithm</strong>.</td>
</tr>
</tbody>
</table>
CONGESTION CONTROL ALGORITHMS

1) Open Loop Control
2) Admission Control
3) Policing
4) Leaky Bucket Algorithm
5) Traffic Shaping
6) Closed Loop Control
7) End to End v/s Hop by Hop
8) Implicit v/s Explicit Feedback
CONGESTED SWITCH

Figure 12.18 (a) A Congested switch
(b) Throughput with and without congestion switch
### Congestion Control Policies Implemented At Various Levels

<table>
<thead>
<tr>
<th>Layer</th>
<th>Policies</th>
<th>Policies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport</td>
<td>• Retransmission policy</td>
<td>• Out-of-order caching policy</td>
</tr>
<tr>
<td></td>
<td>• Acknowledgement policy</td>
<td>• Flow control policy</td>
</tr>
<tr>
<td></td>
<td>• Timeout determination</td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td>• Virtual circuits versus datagram inside the subnet</td>
<td>• Packet discard policy</td>
</tr>
<tr>
<td></td>
<td>• Packet queuing and service policy</td>
<td>• Packet lifetime management</td>
</tr>
<tr>
<td></td>
<td>• Routing algorithm</td>
<td></td>
</tr>
<tr>
<td>Data link</td>
<td>• Retransmission policy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Out-of-order caching policy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Acknowledgement policy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Flow control policy</td>
<td></td>
</tr>
</tbody>
</table>
ADMISSION CONTROL

Makes decision based on congestion whether to accept or reject new traffic flow.

Used in virtual circuit packet switching such as ATM over connections n so called CAC – Connection Admission Control.

If QoS parameters like delay, loss probability, variance, bandwidth, etc can be satisfied by available resources, packets/flows are accepted, else rejected.

Effective bandwidth is obtained when the flow lies between average rate and peak rate.
POLICING

When the flow is accepted by admission control, QoS is satisfied negotiated traffic parameters during lifetime of flow.

To prevent source from violating its negotiated parameters, monitoring of traffic flow is required continuously.

This process of monitoring and enforcing the traffic flow is called policing.

Network may discard the flows that violates the negotiated contracts.

To do so, tagging is done to set lowest priority.

When network resources are exhausted, tagged traffic is first to be discarded.
LEAKY BUCKET
ALGORITHM / TRAFFIC SHAPING

Water poured in the bucket with a hole at bottom.
Bucket leaks at constant rate.
Hole ensures that bucket will never overflow.
Process of altering traffic flow to ensure conformation.
Incoming packets are stored in bucket and stream of packets are served to server so that output is smooth.
Bucket has buffer that store momentary bursts of water/packets.
Incoming packets are discarded when buffer is full.
TOKEN BUCKET ALGORITHM / TRAFFIC SHAPING

Regulates packet that are not conforming.

Packets that are conforming are passed through channel without further delay.

Tokens are generated periodically at constant rate & stored in Token Bucket.

If token bucket is full, additional tokens are discarded.
CLOSED LOOP CONTROL

Performs no reservation.
Performs based on feedback.
Parameters are buffer content and link utilization.
Regulates packet flow rate.
Two types:
1) End to End v/s Hop by Hop
2) Implicit v/s Feedback
FEEDBACK: EXPLICIT OR IMPLICIT

Explicit Feedback
- ECN: Explicit Congestion Notification field in TCP and IP header indicating congestion.
- Turned on indicating Possibility of Congestion
- ICMP: Internet Control Message Protocol

Implicit Feedback
- Older TCP implementation had no field to indicate congestion.
- Retransmission, RTT and Dropping events indicate congestion.
REPEATER
GATEWAY
TUNNELING
FRAGMENTATION

Figure 12.24 Fragmentation

Packet

$G_1$

$G_1$ fragments a large packet

Network 1

$G_2$

$G_2$ reassembles the fragments

$G_3$

$G_3$ fragments again

$G_4$

$G_4$ reassembles again
### How Networks Differ?

Networks vary in physical and data link layer by modulation techniques and frame format. Some of the differences in the network implemented in the network layer are as follows:

<table>
<thead>
<tr>
<th>S. No</th>
<th>Parameter</th>
<th>Some Possibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Service offered</td>
<td>Connection oriented versus connection less</td>
</tr>
<tr>
<td>2</td>
<td>Protocols</td>
<td>IP, IPX, SNA, ATM, MPLS etc.</td>
</tr>
<tr>
<td>3</td>
<td>Addressing</td>
<td>Flat versus hierarchical</td>
</tr>
<tr>
<td>4</td>
<td>Multicasting</td>
<td>Present or Absent</td>
</tr>
<tr>
<td>5</td>
<td>Packet Size</td>
<td>Every network has its own maximum</td>
</tr>
<tr>
<td>6</td>
<td>Quality Service</td>
<td>Present or Absent</td>
</tr>
<tr>
<td>7</td>
<td>Error Handling</td>
<td>Reliable, ordered and ordered delivery</td>
</tr>
<tr>
<td>8</td>
<td>Flow Control</td>
<td>Sliding window, rate control</td>
</tr>
<tr>
<td>9</td>
<td>Congestion control</td>
<td>Leaky bucket, RED, Token bucket choke etc.</td>
</tr>
<tr>
<td>10</td>
<td>Security</td>
<td>Encryption</td>
</tr>
<tr>
<td>11</td>
<td>Parameters</td>
<td>Different timeouts, flow specification etc</td>
</tr>
<tr>
<td>12</td>
<td>Accounting</td>
<td>By connect time, by packet, by byte</td>
</tr>
</tbody>
</table>
## COMPARISON, ANSWER OF ASSIGNMENT

### 10.4.4 Comparisons Of Scheduling Approaches

<table>
<thead>
<tr>
<th>Scheduling Approaches</th>
<th>Description</th>
</tr>
</thead>
</table>
| Reservation            | - Stations submit a request for the next round of data transmission.  
                        | - Requires a lot of overhead information. |
| Polling                | - Centralized controller repeatedly polls stations and allows each to transmit one protocol.  
                        | - Uses dynamic form of time-division multiplexing  
                        | - Provides fairness through regular access opportunities  
                        | - Can provide bounds on access delay |
| Token Passing          | - Stations circulates a token, each time it receives a token it transmit one packet.  
                        | - Dynamic form of time-division multiplexing when users transmit in round-robin scheme  
                        | - Requires token management system for efficient performance. |
### Comparisons of Random Access and Scheduling MAC Control

<table>
<thead>
<tr>
<th>Random access approach</th>
<th>Scheduling approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. In random or contention methods, no station is superior to another station and none is assigned the control over another.</td>
<td>In scheduled or controlled access, the stations consult one another to find which station has the right to send. A station cannot send unless it has been authorized by another station.</td>
</tr>
<tr>
<td>2. Random access provides chaotic, uncoordinated, and unordered access.</td>
<td>Scheduling approach provides orderly access to the medium.</td>
</tr>
<tr>
<td>3. When bandwidth is plentiful, random access systems can provide very small delays as long as the systems are operated with light load.</td>
<td>The scheduling approach has less variability in the delays encountered by packets. Therefore, they are used for supporting applications with stringent delay requirements.</td>
</tr>
<tr>
<td>4. Popular random access methods are ALOHA, CSMA, CSMA/CD.</td>
<td>Popular scheduling access methods are Reservation, Polling, Token Passing.</td>
</tr>
<tr>
<td>5. Relatively simple.</td>
<td>Sophisticated.</td>
</tr>
<tr>
<td>6. Channel bandwidth is used to alert stations during collisions.</td>
<td>Channel bandwidth carries explicit information that allows station to schedule their transmissions.</td>
</tr>
<tr>
<td>7. More collision</td>
<td>Avoids collision</td>
</tr>
</tbody>
</table>
SYLLABUS OVER.

BEST WISHES FOR YOUR EXAMS.

IT WAS GREAT TIME TEACHING YOU ALL. THANK YOU FOR EVERYTHING. BLESSINGS.