The Network Layer

- Concerned with getting packets from the source all the way to the destination: Routing through the subnet, load balancing, congestion control.

- Protocol Data Unit (PDU) for network layer protocols = packet

- Types of network services to the transport layer:
  - **Connectionless**: Each packet carries full destination address.
  - **Connection-oriented**:
    - Connection is set up between network layer processes on the sending and receiving sides.
    - The connection is given a special identifier until all data has been sent.

- Internal organization of the network layer (in the subnet):
  - **Datagram**: Packets are sent and routed independently with each carrying the full destination address (TCP/IP)
  - **Virtual circuit**: A virtual circuit is set up to the destination using a circuit number stored in tables in routers along the way. Packets only carry the virtual circuit number. All packets follow the same route (ATM).
Routing Algorithms

To decide which output line an incoming packet should be transmitted on.

• Static Routing (Nonadaptive algorithms):
  – Shortest path routing:
    • Build a graph of the subnet with each node representing a router and each arc representing a communication link.
    • The weight on the arcs represents: a function of distance, bandwidth, communication costs mean queue length and other performance factors.
    • Several algorithms exist including Dijkstra’s shortest path algorithm.
  – Selective flooding: Send the packet on all output lines going in the right direction to the destination.
  – Flow-based routing: Based on known capacity and link loads.
Static Routing: Shortest Path Routing

- First five steps of an example using Dijkstra’s algorithm

![Diagrams](image-url)
Static Routing: Flow-Based Routing

- A routing matrix is constructed; used when the mean data flow in network links is known and stable.
- Given: Capacity matrix $C_{ij}$, Traffic matrix $F_{ij}$,
- Mean delay at each line $T = 1/(\mu C - \lambda)$

$$\mu C \text{ line capacity packet/sec}$$

$$\lambda \text{ traffic packet/sec}$$

A subnet with link capacities given in kbps

The traffic in packets/sec and routing table
Flow-based routing analysis example for network on previous page, assuming:
Mean packet size = 800 bits
Reverse traffic is the same as forward traffic
weight_i = flow in link i / total flow in the subnet
Mean delay per packet = \( \sum \text{weight}_i \times T_i \) = 86 msec in above example
Goal find a flow with minimum mean delay per packet.
Dynamic Routing: Distance Vector

- Each router maintains a table with one entry for each router in the subnet including the preferred outgoing link, and an estimate of time or distance to the destination router.

- Neighbor router table entries are gathered by sending ECHO packets which are sent back with a time stamp.

- Each router exchanges its table of estimates with its neighbors; the best estimate is chosen.

Used in ARPANET until 1979

<table>
<thead>
<tr>
<th>To</th>
<th>A</th>
<th>I</th>
<th>H</th>
<th>K</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>24</td>
<td>20</td>
<td>21</td>
<td>8</td>
</tr>
<tr>
<td>B</td>
<td>12</td>
<td>36</td>
<td>31</td>
<td>28</td>
<td>20</td>
</tr>
<tr>
<td>C</td>
<td>25</td>
<td>18</td>
<td>19</td>
<td>36</td>
<td>28</td>
</tr>
<tr>
<td>D</td>
<td>40</td>
<td>27</td>
<td>8</td>
<td>24</td>
<td>20</td>
</tr>
<tr>
<td>E</td>
<td>14</td>
<td>7</td>
<td>30</td>
<td>22</td>
<td>17</td>
</tr>
<tr>
<td>F</td>
<td>23</td>
<td>20</td>
<td>19</td>
<td>40</td>
<td>30</td>
</tr>
<tr>
<td>G</td>
<td>18</td>
<td>31</td>
<td>6</td>
<td>31</td>
<td>18</td>
</tr>
<tr>
<td>H</td>
<td>17</td>
<td>20</td>
<td>0</td>
<td>19</td>
<td>12</td>
</tr>
<tr>
<td>I</td>
<td>21</td>
<td>0</td>
<td>14</td>
<td>22</td>
<td>10</td>
</tr>
<tr>
<td>J</td>
<td>9</td>
<td>11</td>
<td>7</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>K</td>
<td>24</td>
<td>22</td>
<td>22</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>L</td>
<td>29</td>
<td>33</td>
<td>9</td>
<td>9</td>
<td>15</td>
</tr>
</tbody>
</table>

New estimated delay from J

Vectors received from J's four neighbors

New routing table for J
**Distance Vector Routing: Count-to-Infinity Problem**

The main reason Distance Vector Routing has been mostly abandoned.

---

**Link AB Initially down**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>3</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>5</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>5</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>7</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>7</td>
<td></td>
<td>8</td>
</tr>
</tbody>
</table>

**Link AB Initially Up**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>∞</td>
<td></td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

---

*initially*
Dynamic Routing: Link State Routing
• Resolves Count-to-Infinity Problem present in Distance Vector Routing.
• Variants of link state routing are widely used.
• Each router using Link State Routing must:
  – Discover its neighbors and know their network address
  – Measure the delay or cost to each of the neighbors using ECHO packets.
  – Construct a link state packet to include what it learned about its
    neighbors including the age of the information.
  – Send the link state packet to all other routers in the subnet
  – Compute the shortest path to every other router based on
    information gathered from all link state packets (Dijkstra’s algorithm may be used).
Hierarchical Routing

- The subnet is divided into several regions of routers.
- Each router maintains a table of routing information to routers in its region only.
- Packets destined to another region are routed to a designated router in that region.

A two-level Hierarchical Routing Example:

Optimum number of hierarchy levels:
For a subnet of \(N\) routers:
- Optimum \# of levels = \(\ln N\)
- Total \# of table entries = \(e \ln N\)
Congestion Control Methods

- **Traffic Shaping:**
  - Heavily used in VC subnets including ATM networks.
  - Avoid bursty traffic by producing more uniform output at the hosts.
  - Representative examples: Leaky Bucket, Token Bucket.

- **Admission Control:**
  - Used in VC subnets.
  - Once congestion has been detected in part of the subnet, no additional VCs are created until the congestion level is reduced.

- **Choke Packets:**
  - Used in both datagram and VC subnets
  - When a high level of line traffic is detected, a choke packet is sent to source host to reduce traffic.
  - Variation Hop-by-Hop choke packets.

- **Load Shedding:**
  - Used only when other congestion control methods in place fail.
  - When capacity is reached, routers or switches may discard a number of incoming packets to reduce their load.
Congestion Control Algorithms: The Leaky Bucket

- A traffic shaping method that aims at creating a uniform transmission rate at the hosts.
- Used in ATM networks.
- An output queue of finite length is connected between the sending host and the network.
- Either built into the network hardware interface or implemented by the operating system.
- One packet (for fixed-size packets) or a number of bytes (for variable-size packets) are allowed into the queue per clock cycle.
- Congestion control is accomplished by discarding packets arriving from the host when the queue is full.
Congestion Control Algorithms: The Token Bucket

• An output queue is connected to the host where tokens are generated and a finite number is stored at the rate of $\Delta T$.
• Packets from the host can be transmitted only if enough tokens exist.
• When the queue is full tokens are discarded not packets.
• Implemented using a variable that counts tokens.
Congestion Control Algorithms: Choke Packets

- Used in both VC and datagram subnets.
- A variable “u” is associated by the router to reflect the recent utilization of an output line:
  \[ u = au_{\text{old}} + (1 - a)f \]
- When “u” goes above a given threshold, the corresponding line enters a warning state.
- Each new packet is checked if its output line is in warning state if so:
  - The router sends a choke packet to the source host with the packet destination.
  - The original packet is tagged (no new choke packets are generated).
- A host receiving a choke packet should reduce the traffic to the specified destination.
- A variation (Hop-by-Hop Choke Packets) operate similarly but take effect at each hop while choke packets travel back to the source.
INTERNETWORKING

• When several network types with different media, topology and protocols, are connected to form a larger network:
  – UNIX: TCP/IP
  – Mainframe networks: IBM’s SNA, DEC’s DECnet
  – PC LANs: Novell: NCP/IPX, AppleTalk
  – ATM, wireless networks etc.

• The “black box” converter unit used to connect two different networks depend on the layer of connection:
  – Layer 1 (physical): Repeaters, bit level
  – Layer 2 (data link): Bridges, data link frames
  – Layer 3 (network): Multiprotocol routers, packets
  – Layer 4 (transport): Transport gateways
  – Above 4 (application): Application gateways.
Internetworking Issues: Fragmentation

- When packets from a subnet travel to another subnet with a smaller maximum packet size, packets have to be broken down into **fragments** and send them as **internet packets**.

---

**Transparent fragmentation**

**Non-transparent fragmentation**
The Internet

- Evolved from the ARPANET (the Advanced Research Projects Agency Network), a project funded by The U.S. Department of Defense (DOD) in 1969.

- ARPANET's purpose was to provide the U.S. Defense Network (DDN) with redundant links between its sites and the Pentagon, relying on intelligent data packets that could automatically route themselves around failed network routers and links.

- During the 1970s, the ARPANET gradually transformed and expanded into the current Internet as new protocols and technologies became available, and as additional defense, research, scientific, commercial and development organizations were added to the network.

- At the network layer level:

  The Internet is a global collection of subnets held together by a common main network layer protocol: IP (Internet Protocol).

- Example Transport Layer Protocols:
  - Connection-oriented: TCP (Transport Control Protocol),
  - Connectionless: UDP (User Data Protocol)
Classic IP Addressing Architecture

- The classical IP network prefix is the Class A, B, C, D, or E network prefix.
- These address ranges are discriminated by observing the values of the most significant bits of the address, and break the address into simple network prefix (or number) and host number fields:

  \[
  \text{IP-address ::=} \{ \text{<Network-prefix>}, \text{<Host-number>} \}
  \]

- The network classes are identified as follows:
  - 0xxx Class A general purpose unicast addresses with standard 8 bit prefix.
  - 10xx Class B general purpose unicast addresses with standard 16 bit prefix.
  - 110x Class C general purpose unicast addresses with standard 24 bit prefix.
  - 1110 Class D IP Multicast Addresses - 28 bit prefix, non-aggregatable
  - 1111 Class E reserved for experimental use.

- To allow hierarchical routing, an IP address can be further divided:
  - \[\text{IP-address ::=} \{ \text{<Network-number>}, \text{<Subnet-number>}, \text{<Host-number>} \}\]

- The interconnected physical networks within an organization use the same network prefix but different subnet numbers.
- Routers outside the network treat <Network-prefix> and <Host-number> together as an uninterpreted part of the 32-bit IP address.
Primary IP Primary Address Classes

**Class A**
- Max # of class A networks = \(2^7 - 2 = 126\) networks
- Each containing \(2^{24} - 2 = 16,777,214\) host addresses
- 50% of the total IPv4 unicast address space

**Class B**
- Max # of class B networks = \(2^{14} = 16,384\) networks
- Each containing \(2^{16} - 2 = 65,534\) host addresses
- 25% of the total IPv4 unicast address space

**Class C**
- Max # of class C networks = \(2^{21} = 2,097,152\) networks
- Each containing \(2^{8} - 2 = 254\) host addresses
- 12.5% of the total IPv4 unicast address space

Allocated Network Numbers By Class

Growth of Internet Routing Tables
Network Mask

• A 32-bit number indicating the range of IP addresses residing on a single IP network/subnet/supernet and the length of the network-prefix.

• For example, the network mask for a class C IP network is given as 255.255.255.0

• To identify the network/subnet of a destination IP address, routers logically AND the mask and the full destination IP address then compare the result with network addresses in routing table to determine the next hop.

• One of the fundamental features of IP addressing is that each address contains a self-encoding key that identifies the dividing point between the network-prefix and the host-number.
  – For example, if the first two bits of an IP address are 1-0, the dividing point falls between the 15th and 16th bits.
  – This simplified the routing system during the early years of the Internet because the original routing protocols did not supply a "mask" with each route to identify the length of the network-prefix.
### IP (Internet Protocol): Header and Options

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Specifies how secret the datagram is</td>
</tr>
<tr>
<td>IHL</td>
<td>Gives the complete path to be followed</td>
</tr>
<tr>
<td>Type of service</td>
<td>Gives a list of routers not to be missed</td>
</tr>
<tr>
<td>Total length</td>
<td>Makes each router append its IP address</td>
</tr>
<tr>
<td>Identification</td>
<td>Makes each router append its address and timestamp</td>
</tr>
<tr>
<td>Time to live</td>
<td></td>
</tr>
<tr>
<td>Protocol</td>
<td></td>
</tr>
<tr>
<td>Header checksum</td>
<td></td>
</tr>
<tr>
<td>Source address</td>
<td></td>
</tr>
<tr>
<td>Destination address</td>
<td></td>
</tr>
<tr>
<td>Options (0 or more words)</td>
<td></td>
</tr>
</tbody>
</table>
Internet Control Message Protocol (ICMP)

- ICMP is an Internet network protocol that provides an error-reporting mechanism.
- Usually used by routers to report unexpected events and errors and to measure delays (ping), explore new routers and routes (traceroute).
- ICMP messages are encapsulated in IP packets:
  - When reporting an error, router sends message back to source in an ICMP datagram message contains information about problem.
  - Ping program uses ICMP echo request and echo reply messages sent by host to test if the target host is reachable.
Routing In The Internet

• TCP/IP Networks and LANs:
  – The Address Resolution Protocol (ARP).
  – Table Lookup Address Resolution.
  – Reverse Address Resolution Protocol (RARP)

• Internal Routing in Autonomous systems:
  – Link state based Open Shortest Path First (OSPF).

• External routing between Autonomous systems:

• Classless Inter-Domain Routing (CIDR).
The Address Resolution Protocol (ARP)

- **Address resolution**: Finding hardware address that corresponds to a network layer protocol address.

- In Ethernet-based LANs, each machine connected to the LAN has a unique flat 48 bit **Ethernet address** encoded in its NIC by the manufacturer.

- **ARP**: When the transport layer on a LAN-connected machine passes a message to be transmitted to the IP layer and destined to another machine on the LAN:
  - Translate the host name of the receiver to its IP address using the **Domain Name System (DNS)**.
  - Broadcast a packet to the LAN requesting the Ethernet address of the machine with the given IP (step 1 of ARP).
  - The target machine with this IP replies with its Ethernet address E (step 2 of ARP).
  - IP software on the source machine builds an Ethernet frame with Ethernet address E and puts the IP packet its payload field.
  - The destination machine picks up the Ethernet frame and extracts and passes the IP packet to its IP software.
Table Lookup Address Resolution

- A table containing the IP address and hardware address of each host on the LAN and its corresponding hardware (Ethernet) address is used.
- When sending frames to another host on the LAN, the table is searched on the IP address and the corresponding hardware address in the table is found.

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Hardware Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>197.15.3.2</td>
<td>0A:07:4B:12:82:36</td>
</tr>
<tr>
<td>197.15.3.3</td>
<td>0A:9C:28:71:32:8D</td>
</tr>
<tr>
<td>197.15.3.4</td>
<td>0A:11:C3:68:01:99</td>
</tr>
<tr>
<td>197.15.3.5</td>
<td>0A:74:59:32:CC:1F</td>
</tr>
<tr>
<td>197.15.3.6</td>
<td>0A:04:BC:00:03:28</td>
</tr>
<tr>
<td>197.15.3.7</td>
<td>0A:77:81:0E:52:FA</td>
</tr>
</tbody>
</table>

A portion of an IP/Ethernet address resolution table

- Often used in conjunction with ARP to reduce address resolution overhead:
  - The table initially cleared at system startup.
  - For every host with no entry in the table ARP is used to find its hardware address.
  - The corresponding hardware address obtained from ARP is added or cached in the sending host’s table.
  - Table entries are periodically discarded to prevent stale addresses.
Reverse Address Resolution Protocol (RARP)

- Used by machines joining the network, with no IP address stored in the machine, to find out the assigned IP addresses corresponding to the machine’s Ethernet NIC addresses.

- Such a machine broadcasts a request with its Ethernet address using RARP.

- The RARP server on the LAN replies to the request with the IP address from its configuration files.

- RARP broadcasts are limited to a LAN and not forwarded to routers → Each LAN must have an RARP server.

- Bootstrap protocol BOOTP: Uses UDP packets which can be forwarded to routers → No need for a BOOTP server on each LAN.
Routing In The Internet

- The Internet as a whole is formed from a number of Autonomous Systems.
- Each AS is further divided into areas with a special area 0 (the backbone) connected to all its other areas.
- **Internal routing** in an AS is handled by an interior gateway protocol: Open Shortest Path First (OSPF), a hierarchical, dynamic link-state routing algorithm which supports:
  - Point-to-point connection between two routers.
  - Multi-access networks, with broadcasting (LANS), and without (WANS).
- **External routing** between ASes is handled by an exterior gateway protocol: Border Gateway Protocol (BGP):
  - The network is reduced to BGP routers and their links.
  - Based on a distance vector protocol with actual path used being exchanged between routers.
Open Shortest Path First (OSPF)

- OSPF is a TCP/IP link-state based Internet routing protocol designed to run internal to a single Autonomous System.
- IP packets are routed based solely on the destination IP address found in the IP packet header without adding further protocol headers.
- Each OSPF router maintains an identical link-state database describing the router's usable interfaces, reachable neighbors and the Autonomous System's topology.
- From this database, a routing table is initially calculated by constructing a shortest-path tree.
- When several equal-cost routes to a destination exist, traffic is distributed equally among them.
- Topological changes in the AS (such as router interface failures) are quickly detected by calculating new loop-free routes.
- Each router distributes its local state throughout the Autonomous System by flooding.
- Sets of networks may be grouped together in an area where the topology of an area is hidden from the rest of the Autonomous System.
The Relation Between: 
ASes, 
Backbones 
and 
Areas in OSPF
Border Gateway Protocol (BGP)

- BGP is intended for use between networks owned by different organizations (Backbone Providers).
- BGP is often referred to as a tool for "policy" routing, because
  - It may not take into account network constraints such as available bandwidth or network load.
  - The primary routing protocol that Internet backbone providers use to exchange routing information.
- Each provider will configure its border routers to announce certain routes to its neighbors.
- The neighboring provider will filter those announcements based on its own policies and will discard some of those announced routes.
- Of the routes that are accepted, some may only be used locally, in the provider's own routing tables, and some may be announced to other neighboring backbones.
Classless Inter-Domain Routing (CIDR)

- Eliminates the traditional concept of Class A, Class B, and Class C network addresses, replacing them a generalized concept of a "network-prefix."
- Supports route aggregation where a single routing table entry can represent the address space of perhaps thousands of traditional class network routes.
- Without the rapid deployment of CIDR in 1994 and 1995, the Internet routing tables would have been in excess of 70,000 routes (instead of the current 30,000+).
- A prefix-length is included with each piece of routing information. The prefix-length is a way of specifying the number of leftmost contiguous bits in the network-portion of each routing table entry.
  - For example, a network with 20 bits of network-number and 12-bits of host-number would have with a 20-bit prefix length, which could be a former Class A, Class B, or Class C.
- Routers that support CIDR do not make assumptions based on the first 3-bits of the address, they rely on the prefix-length information provided with the route.
Asynchronous Transfer Mode (ATM)

- ATM is a specific asynchronous packet-oriented information, multiplexing and switching transfer model standard, originally devised for digital voice and video transmission, which is
  - Based on 53-byte fixed-length cells.
  - Each cell consists of a 48 byte information field and a 5 byte header, which is mainly used to determine the virtual channel and to perform the appropriate routing.
  - Cell sequence integrity is preserved per virtual channel. Thus all cells belonging to a virtual channel must be delivered in their original order.
  - Original primary rate: 155.52 Mbps. Additional rate: 622.08 Mbps

- ATM is connection-oriented.
  - Header values including virtual path/circuit numbers are assigned to each section of a connection for the complete duration of the connection.

- The information field of ATM cells is carried transparently through the network. No processing like error control is performed on it inside the network.

- All services (voice, video, data, ) can be transported via ATM, including connectionless services.
  - To accommodate various services an appropriate adaptation layer is provided to fit information of all services into ATM cells and to provide service specific functions (e.g. clock recovery, cell loss recovery, ...).
Virtual Circuits

• When a virtual circuit is established:
  – The route is chosen from beginning to end (circuit setup needed).
  – Routers or switches along the circuit create table entries used to route data transmitted on the virtual circuit.
  – Permanent virtual circuits - Switched virtual circuits
ATM Cells & Switches

ATM Cell Format

Fixed cell size = 53 bytes

Cell Duration: ~ 2.7 μsec for 155.52 Mbps ATMs
~ 700 nsec for 622.08 Mbps ATMs

An ATM switch

Input side

Output side

Cells come in

Cells are switched

Switching fabric

Cells go out
ATM Layer Headers

ATM layer header at User-Network Interface UNI

ATM layer header at Network-Network Interface NNI

GFC: General Flow Control
VPI: Virtual Path Identifier
VCI: Virtual Channel Identification

PTI: Payload Type
CLP: Cell Loss Priority
HEC: Header Error Check
The Network Layer In ATM Networks

• The ATM layer handles the functions of the network layer:
  – Moving cells from source to destination in order.
  – Routing algorithms within ATM switches, global addressing.

• Connection-oriented without acknowledgments.

• The basic element is the unidirectional virtual circuit or channel with fixed-size cells.

• Two possible interfaces:
  – UNI (User-Network Interface): Boundary between an ATM network and host.
  – NNI (Network-Network Interface): Between two ATM switches (or routers).
ATM Network Connection Setup/Release

Connection Setup

Connection Release
ATM Virtual Path Re-routing Example

Rerouting a virtual path re-routes all of its virtual circuits
## ATM Routing Example

<table>
<thead>
<tr>
<th>Source</th>
<th>Incoming line</th>
<th>Incoming VPI</th>
<th>Destination</th>
<th>Outgoing line</th>
<th>Outgoing VPI</th>
<th>Path:</th>
</tr>
</thead>
<tbody>
<tr>
<td>NY</td>
<td>1</td>
<td>1</td>
<td>SF</td>
<td>4</td>
<td>1</td>
<td>New</td>
</tr>
<tr>
<td>NY</td>
<td>1</td>
<td>2</td>
<td>Denver</td>
<td>4</td>
<td>2</td>
<td>New</td>
</tr>
<tr>
<td>LA</td>
<td>3</td>
<td>1</td>
<td>Minneapolis</td>
<td>0</td>
<td>1</td>
<td>New</td>
</tr>
<tr>
<td>DC</td>
<td>1</td>
<td>3</td>
<td>LA</td>
<td>3</td>
<td>2</td>
<td>New</td>
</tr>
<tr>
<td>NY</td>
<td>1</td>
<td>1</td>
<td>SF</td>
<td>4</td>
<td>1</td>
<td>Old</td>
</tr>
<tr>
<td>SF</td>
<td>4</td>
<td>3</td>
<td>DC</td>
<td>1</td>
<td>4</td>
<td>New</td>
</tr>
<tr>
<td>DC</td>
<td>1</td>
<td>5</td>
<td>SF</td>
<td>4</td>
<td>4</td>
<td>New</td>
</tr>
<tr>
<td>NY</td>
<td>1</td>
<td>2</td>
<td>Denver</td>
<td>4</td>
<td>2</td>
<td>Old</td>
</tr>
<tr>
<td>SF</td>
<td>4</td>
<td>5</td>
<td>Minneapolis</td>
<td>0</td>
<td>2</td>
<td>New</td>
</tr>
<tr>
<td>NY</td>
<td>1</td>
<td>1</td>
<td>SF</td>
<td>4</td>
<td>1</td>
<td>Old</td>
</tr>
</tbody>
</table>

Possible routes through the Omaha ATM switch
ATM Routing Example: Table Entries

Table entries corresponding to routes through the Omaha ATM switch

<table>
<thead>
<tr>
<th>Incoming VPI</th>
<th>Outgoing VPI</th>
<th>Line 0</th>
<th>Line 1</th>
<th>Line 2</th>
<th>Line 3</th>
<th>Line 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4095</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ATM Switch Functions

• The main function of an ATM switch is to relay user data cells from input ports to the appropriate output ports. The switch processes only user data cell headers and the payload is carried transparently.
  – As soon as the cell comes in through the input port, Virtual Path/Channel Identifiers (VPI/VCI) information is extracted from the cell and used to route the cells to the appropriate output port.
  – This function can be divided into three functional blocks: the input module at the input port, the cell switch fabric (or switch matrix) that performs the actual routing, and the output modules at the output ports.

• Establishment and control of the VP/VC connections.
  – Unlike user data cells, information in signaling or control cells payload is not transparent to the network.
  – The switch identifies signaling cells, and even generates some itself.
  – Connection Admission Control (CAC) carries out the major signaling functions required.
  – Signaling/control information may not pass through the cell switch fabric, and instead is exchanged through a separate signaling network.

• Network management functions, concerned with monitoring the controlling the network to ensure its correct and efficient operation.
  – Fault management functions,
  – Performance management functions,
  – Configuration management functions.

• Connection admission control, usage/network parameter control and congestion control, usually handled by input modules.
A Generic ATM Switching Architecture

IM = Input Module
OM = Output Module
CAC = Connection Admission Control
SM = Switch Management

ATM/Sonet Lines

Cell Switch Fabric

Input Side

Output Side

ATM/Sonet Lines

Switch Interface
### Running TCP/IP Over An ATM Subnet

<table>
<thead>
<tr>
<th>Email</th>
<th>FTP</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>IP</td>
<td></td>
</tr>
<tr>
<td>ATM</td>
<td>Data link</td>
<td></td>
</tr>
<tr>
<td>Physical</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The Transport Layer

• Provides reliable end-to-end service to processes in the application layer:
  – Connection-oriented or connection-less services.
• TPDUs (Transport Protocol Data Units): Refer to messages sent between two transport entities.
• Transport service primitives: Allow application programs to access the transport layer services.
• Data received from application layer is broken into TPDUs that should fit into the data or payload field of a packet.
• Packets received possibly out-of-order from the network layer are reordered and assembled for delivery to application layer.
• Transport Entity: Hardware/software in the transport layer:
  – In operating system kernel or,
  – In a separate user process or,
  – In the network interface card.
• Option Negotiation: The process of negotiating quality of service (QoS) parameters between the user and remote transport entities as specified by applications.
Data Link Layer Vs. Transport Layer

Data Link Layer Environment:
Adjacent routers.

Transport Layer Environment:
End-to-End from source to destination.
Simple Transport Layer
Primitives

Primitives used to provide transport services to applications

<table>
<thead>
<tr>
<th>Primitive</th>
<th>TPDU sent</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>LISTEN</td>
<td>(none)</td>
<td>Block until some process tries to connect</td>
</tr>
<tr>
<td>CONNECT</td>
<td>CONNECTION REQ.</td>
<td>Actively attempt to establish a connection</td>
</tr>
<tr>
<td>SEND</td>
<td>DATA</td>
<td>Send information</td>
</tr>
<tr>
<td>RECEIVE</td>
<td>(none)</td>
<td>Block until a DATA TPDU arrives</td>
</tr>
<tr>
<td>DISCONNECT</td>
<td>DISCONNECTION REQ.</td>
<td>This side wants to release the connection</td>
</tr>
</tbody>
</table>

Transport Flow Control

- To accomplish transport flow control a Sliding Window protocol is used end-to-end using TDPUs as protocol transfer units
  - Available receiver capacity and buffering used as a receive window RWIN.
  - Receiver buffer over-runs are usually not allowed.

- Each TPDU must carry an identifier or sequence number to distinguish between original TPDUs and delayed duplicates.

- To curtail the effect of delayed duplicates:
  - Packets are not allowed to live forever.
  - Each packet has a restricted maximum lifetime = T.

- The low-order k-bits of a time-of-day clock, of the form of a binary counter, are usually used to generate initial TPDU sequence numbers for new connections.
  - This clock is assumed to keep running even if the host crashes.
  - The clock frequency and k are selected such that a generated initial sequence number should not repeat (i.e. be assigned to another TPDU) for a period longer than the maximum packet lifetime T [forbidden region].
Transport Flow Control

- Once an initial TDPU sequence number is assigned, it’s incremented as required by the connection.
- TDPU sequence numbers of a connection may run into forbidden region if:
  - A host sends too much data too fast on a newly opened connection:
    - Here, actual used sequence number vs. time is more steep than initial sequence number generation vs. time.
    - This restricts the maximum data rate of a connection to one TDPU per cycle.
  - At any connection data rate less than the initial sequence number generation clock rate:
    - The actual sequence numbers used will eventually run into the forbidden region from the left.
    - This condition must be checked by transport entity requiring a TDPU delay of $T$, or sequence number re- synchronization.
TPDU Sequencing

TPDUs may not be issued in the forbidden region

T = Maximum Packet Lifetime

The re-synchronization problem.
Connection data rate less than initial sequence number generation clock rate
Transport Connection Protocol:

Three-Way Handshake

Normal operation

Old duplicate CONNECTION REQUEST

Duplicate CONNECTION REQUEST and duplicate ACK
Transport Connection Release Scenarios

Normal case of three-way handshake

Response lost

Final ACK lost

Response lost and subsequent DRS lost
The Internet Transport Protocols (TCP, UDP)

- **TCP (Transmission Control Protocol), RFC 1323:**
  - Connection-oriented protocol designed to provide reliable end-to-end byte streams over unreliable internetworks.
  - TCP transport entity (TCP) is either implemented as a user process or as part of the operating system kernel.
  - TCP accepts user data streams from application processes (the application layer interface) as segments and breaks them down into a sequence of separate IP datagrams (of size Max Transfer Unit : (MTU)= 64k, usually 1500 bytes) for transmission.
  - Arriving IP datagrams containing TCP data are passed to the TCP transport entity to reorder, reassemble and reconstruct the original data stream.
  - TCP service and connection is provided to sender and receiver application processes by creating end points (sockets) with a socket address consisting of the IP address and a local 16-bit port number.
### TCP (continued)

<table>
<thead>
<tr>
<th>socket address</th>
<th>(IP address, Port number)</th>
</tr>
</thead>
<tbody>
<tr>
<td>32 bits</td>
<td>16 bits</td>
</tr>
</tbody>
</table>

- To utilize TCP services, a connection must be established between a socket on the sending machine and a socket on the receiving machine using a number of socket calls.
- A socket may be used by a number of open connections.
- A TCP connection is always full-duplex, point-to-point and is identified by the socket identifiers at both end: \((socket1, socket2)\)
- Data passed to TCP by an application may be transmitted immediately, or buffered to collect more data.
- The lowest 256 port numbers are reserved for standard services, Examples: FTP: port 21, Telnet: port 23, SMTP: port 25, HTTP: port 80, NNTP: port 119, etc.
- Client/Server Model: A server application is one always listening to serve incoming data service requests on a specific port number issued by client processes requesting the service.
Of TCP Segments and IP Datagrams

- TCP connections are byte streams not message streams.
- The original segment boundaries at the sender are not preserved at the receiver.
- Example:
  - The sending application sends data to the sending TCP entity as four 512-byte TCP segments in four writes transformed into four IP datagrams.
  - The receiving application can get the data from the receiver TCP entity as four 512-byte segments, two 1024-byte segments or, as given below, as a single 2048-byte segment in a single read.

![Diagram showing TCP segments and IP datagrams]

Four 512-byte TCP segment writes by sending application

A single TCP 2048-byte segment read by receiving application
The TCP Segment Header

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source port</td>
<td></td>
</tr>
<tr>
<td>Destination port</td>
<td></td>
</tr>
<tr>
<td>Sequence number</td>
<td></td>
</tr>
<tr>
<td>Acknowledgement number</td>
<td></td>
</tr>
<tr>
<td>TCP header length</td>
<td></td>
</tr>
<tr>
<td>Urgent pointer</td>
<td></td>
</tr>
<tr>
<td>Window size</td>
<td></td>
</tr>
<tr>
<td>Checksum</td>
<td></td>
</tr>
<tr>
<td>Options (0 or more 32-bit words)</td>
<td></td>
</tr>
<tr>
<td>Data (optional)</td>
<td></td>
</tr>
</tbody>
</table>
Establishing TCP Connections

**Normal Case**

1. Host 1 sends a SYN (SEQ = x) to Host 2.
2. Host 2 responds with a SYN (SEQ = y, ACK = x + 1).
3. Host 1 acknowledges with a SYN (SEQ = x + 1, ACK = y + 1).

**Call Collision**

1. Host 1 sends a SYN (SEQ = x) to Host 2.
2. Host 2 responds with a SYN (SEQ = y).
3. Host 1 sends a SYN (SEQ = x, ACK = y + 1) before receiving Host 2’s SYN.
4. Host 2 acknowledges with a SYN (SEQ = y, ACK = x + 1).
Basic TCP Sliding Window Flow Control

- When a sender transmits a segment it starts a timer.
- When the segment arrives and is accepted at the destination, the receiving TCP entity sends back an acknowledgment:
  - With data if any exist.
  - Has an acknowledgment sequence number equal to the next byte number of this connection it expects to receive.
  - Includes the Receive window, RWIN size it can handle depending on available buffer space.
- If the sender’s timer goes off before the acknowledgment is received the segment is re-transmitted.
TCP Segment Sequence Numbers, Timeout Selection

- TCP segment sequence numbers are needed to make sure stale and delayed duplicate TCP segments do not create confusion and to insure correct sliding window protocol operation.
- Both the transmitter and receiver must identify their segments and these identifiers are usually different.
- The lower $k = 32$ bits from the local time-of-day timer or clock are used to generate initial TCP segment sequence numbers.
- It’s assumed that no segment remains alive longer than the intervening time of $2^k = 2^{32}$ cycles.
  - For the Internet, Maximum Segment Life, MSL = 120 seconds.
- To generate timeout periods, round trip times, RTTs, are maintained for each distinct destination and a timeout is calculated from the most recent RTTs.
  - An estimated RTT may be computed that is the exponential average of the RTTs and then the timeout is chosen as 2 times that estimate.
  - Exponential averaging assumes a number $a$, $0 \leq a \leq 1$, and computes a sequence of estimated RTTs according to the formula:
    \[ ERTT(i+1) = a \times ERTT(i) + (1-a) \times RTT(i) \]
Sliding Window Flow Control In TCP

Sender

Application does a 2K write

2K | SEQ = 0

Receiver

Receiver's buffer

0 4K

Empty

2K

Full

Application reads 2K

Sender is blocked

ACK = 2048 WIN = 2048

Sender may send up to 2K

ACK = 4096 WIN = 0

ACK = 4096 WIN = 2048

Sender does a 3K write

1K | SEQ = 4096

1K 2K
The Silly Window Syndrome

To Avoid It:
Senders and receivers may refrain from sending data or acknowledgments until:
- A minimum amount of data has been received/removed, or
- A timer expires (usually 500 msec).

Diagram:
- Receiver's buffer is full
- Application reads 1 byte
- Room for one more byte
- Window update segment sent
- New byte arrives
- Receiver's buffer is full

Diagram shows the flow of data and buffer status.
An Internet Congestion Control Algorithm: Slow Start

• In addition to the receiver's window size from the Sliding Window Protocol, a transmitter using Slow Start maintains a Congestion Window, and a Threshold, initially set at 64KB.

• The amount of data that can be transmitted at once in a burst of TCP segments is the minimum of the sliding window size and the congestion window size.

• The congestion window starts at the maximum size of a segment.

• If the message is acknowledged, the congestion window is doubled, and so on until the threshold is reached or a message is lost or times out.

• When the threshold is reached, the congestion window can still grow, but now it is incremented by a single maximal segment per successful transmission.

• If no more timeouts occur, the congestion window will continue to grow up to the size of of the receiver's window.

• When a message is lost or timed-out, the threshold is set to 1/2 of the congestion window and the congestion window is restarted at the size of the maximum segment.
Internet Congestion Control:

Threshold Initially = 64K

After an initial timeout before transmission #0: Threshold set to = 64K / 2 = 32K

Congestion Window = TCP segment size = 1K

Maximum Segment size = 1K

Minimum time between consecutive transmissions = Round Trip Time (RTT)

Assuming a timeout has occurred just before transmission number 0 shown.

Threshold Initially = 64K

New Threshold

40K / 2 = 20K

Slow Start Example

64K / 2 = 32K

40K / 2 = 20K
User Datagram Protocol (UDP)

- A connectionless Internet transport protocol that delivers independent messages, called datagrams between applications or processes on host computers.
- Unreliable: Datagrams may be lost, delivered out of order.
- Each datagram must fit into the payload of an IP packet.
- Used by a number of server-client applications with only one request and one response.
- Checksum is optional; may be turned off for digital speech and video transmissions where data quality is less important.

The UDP header:
ATM Adaptation Layer (AAL) Types

In order for ATM to support a variety of services with different traffic characteristics and system requirements:

• It is necessary to adapt the different classes of applications to the ATM layer.
• This function is performed by the AAL, which is service-dependent.
• Four types of AAL were proposed, but two of these (3 and 4) have now been merged into one, AAL 3/4:
  – AAL 1: Supports connection-oriented, constant bit rate, time-dependent services.
  – AAL 2: Supports connection-oriented services that do not require constant bit rates.
  – AAL 3/4: Intended for both connectionless and connection oriented variable bit rate services.
  – AAL 5: Supports connection-oriented variable bit rate data services.
    • More efficient compared with AAL 3/4 at the expense of error recovery and built in retransmission.
**Original Obsolete Service Classes Supported By**

ATM Adaptation Layer (AAL)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timing</td>
<td>Real time</td>
<td>None</td>
<td>Real time</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Real time</td>
<td>None</td>
<td>Real time</td>
<td>None</td>
</tr>
<tr>
<td>Bit rate</td>
<td>Constant</td>
<td>Variable</td>
<td>Constant</td>
<td>Variable</td>
</tr>
<tr>
<td>Mode</td>
<td>Connection orientated</td>
<td>Connectionless</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Characteristics of ATM Service Categories

<table>
<thead>
<tr>
<th>Service Characteristic</th>
<th>CBR</th>
<th>RT-VBR</th>
<th>NRT-VBR</th>
<th>ABR</th>
<th>UBR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth guarantee</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Optional</td>
<td>No</td>
</tr>
<tr>
<td>Suitable for real-time traffic</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Suitable for bursty traffic</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Feedback about congestion</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

- **Constant Bit Rate (CBR)**
- **Real-Time Variable Bit Rate (rt-VBR)**
- **Non-Real-Time Variable Bit Rate (nrt-VBR)**
- **Available Bit Rate (ABR)**
- **Unspecified Bit Rate (UBR)**
Headers, Trailers Added To A Message In ATM Networks

Output by application

Output by convergence sublayer

Output by SAR sublayer

Output by ATM layer

ATM header

SAR header

Convergence sublayer header

SAR trailer

Header

Message

Trailer

CS

SAR

CS

SAR

ATM

SAR

CS

SAR

44-48 Bytes

48

53

Unused

Convergence sublayer trailer
AAL Types

**AAL 1**
- Class A traffic: real-time constant bit rate, connection-oriented, such as uncompressed audio and video.
  - No time-outs, retransmissions, or error-detection provided.
- Convergence sublayer (CS):
  - Detects lost and mis-directed cells.
  - Breaks messages into 46 or 47 byte units given to SAR
- Segmentation Reassembly Sublayer (SAR)
  - 3-bit cell Sequence Number (SN)
  - 3-bit Sequence Number Protection (SNP) or checksum.
  - P cells used to preserve message boundary, pointer field used to provide new message offset (pointer = 0 to 92).

**AAL 2**
- Used for Class B: variable bit rate compressed audio/video traffic with error-detection.
- Similar to AAL 1, no special CS protocol.
- SAR
  - SN (sequence number), IT (information type): start, middle or end of message. LI (Length Indicator) if payload is less than 45 bytes. CRC
  - No field sizes included in the standard AAL 2, thus not often used.
The AAL 1 SAR-PDU Format
(5-byte ATM cell header added to SAR-PDU to form ATM cell)

The AAL 2 SAR-PDU Format
(5-byte ATM cell header added to SAR-PDU to form ATM cell)
AAL 3/4

- Supports Class C/D traffic: variable bit rate, delay-tolerant data traffic requiring some sequencing and/or error detection.
  - Reliable or unreliable stream or message modes.
- Originally two AAL types, connection-oriented and connectionless, which have been combined.
- Only AAL protocol to offer multiple sessions on a single virtual circuit.
- Suffers from high overhead:
  - 8 bytes to each message (CS) and 4 bytes in each cell (SAR).
- Convergence Sublayer (CS):
  - Messages up to 65535 bytes from application are padded into multiples of 4 bytes then a header and trailer is added.
  - CS Header: CPI (common part indicator), Btag (beginning Tag, one byte incremented by one for each new message), BA Size (for buffer allocation).
  - CS Trailer: Etag (same value as Btag, for message framing), Length
  - Message with headers cut into 44 byte chunks to SAR.
- SAR
  - ST (segment type), first, middle, end, only cell of a message.
  - 4-bit SN Sequence Number, 10-bit MID (Multiplexing ID).
  - 6-bit LI (Length Indicator) size of payload in bytes, 10 bit CRC.
AAL 3/4 CPCS-PDU: Convergence Sublayer Message Format

(5-byte ATM cell header added to SAR-PDU to form ATM cell)

AAL 3/4 SAR-PDU Format

Multiplexing ID

00 Middle
01 End
10 Beginning
11 Single cell message

48 Bytes
### SAR-PDU Format for AAL 3/4

<table>
<thead>
<tr>
<th>Segment Type (ST)</th>
<th>Sequence Number (SN)</th>
<th>Multiplexing Identification (MID)</th>
<th>SAR-PDU Payload</th>
<th>Length Indication (LI)</th>
<th>Cyclic Redundancy Check (CRC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 = BOM</td>
<td></td>
<td></td>
<td>44 bytes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00 = COM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01 = EOM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 = SSM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **SAR-PDU Header**: 2 bytes
- **SAR-PDU**: 44 bytes
- **SAR-PDU Trailer**: 2 bytes
AAL Types

- Other AAL (1 - 3/4) protocols were designed by the telecommunications industry without specifically addressing the requirements of the computer industry and suffered from:
  - High overhead, complexity, short message checksum (10 bits).
- Original AAL5 name: SEAL (Simple Efficient Adaptation Layer).
- Supports connection-oriented variable bit rate data services.
- Offers reliable and unreliable services to applications.
- Both message and stream modes supported.
- Convergence Sublayer (CS):
  - No CS header just a trailer.
  - 1-byte UU (user to user), used by higher layers.
  - 2-byte length of actual payload without padding.
  - 4-byte CRC
  - Message with headers cut into 48 byte chunks to SAR.
- SAR
  - No additional headers or trailers are added here.
# Common Part Convergence Sublayer (CPCS)

## AAL 5 CPCS-PDU Header

<table>
<thead>
<tr>
<th>CPCS-PDU Payload (CPCS-SDU)</th>
<th>Pad</th>
<th>CPCS-UU</th>
<th>CPI</th>
<th>Length</th>
<th>CRC</th>
</tr>
</thead>
</table>

- **CPCS-PDU Trailer**: 8 bytes
- **Pad**: From 0 to 47 bytes
- **CPCS User-to-User Indication**
- **Common Part Indicator (1 byte)**
- **Length of Payload (2 bytes)**
- **Cyclic Redundancy Check (CRC) 4 bytes**
The Application Layer

- Client/Server Computing, Basic Approaches:
  - Passing Messages.
    - Example: Communication through sockets (socket programming).
  - Remote Procedure Call (RPC).
    - Examples: Sun RPC, Distributed Computing Environment (DCE) RPC.

- Application/Transport Layer Interface:
  - Example: Berkeley Sockets.

- Application Layer Protocols:
  - Examples: Telnet, FTP, HTTP, etc.

- Network Security:
  - Encryption basics,
  - DES,
  - Public-Key Encryption.

- Domain Name System (DNS).

- Data Compression Basic Techniques.
  - Digital Image Compression Example: JPEG.
Client/Server Computing

- **Client**: A program that initiates a connection request for data from another program using a specific high-level protocol.

- **Server or daemon**: A program running on a machine that responds to communication requests from clients by listening to a specific local address or port and responds with the required data or services using the same specific high-level protocol as the clients.
Realizing Client/Server Communication: Berkeley Sockets

- Sockets, introduced in Berkeley Unix, are a basic mechanism for Inter-Process Communication (IPC) on the same computer system, or on different computer systems connected over a network.
- A socket is an endpoint used by a process for bidirectional communication with a socket associated with another process.
- The communication channel created with sockets can be connection-oriented or connectionless datagram oriented, with the sockets serving as mailboxes.
- IPC using sockets is similar to file I/O operation:
  - A socket appears to the user program to be like a file descriptor on which one can read, write, etc.
  - In the connection-oriented mode, the socket acts like a regular file where a sequence of characters can be read using many read operations.
  - In the connectionless mode, one must get the whole message in a single read operation. Otherwise the rest of the message is lost.
## Berkeley Sockets Primary Functions

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOCKET</td>
<td>Create a new communication end point</td>
</tr>
<tr>
<td>BIND</td>
<td>Attach a local address to a socket</td>
</tr>
<tr>
<td>LISTEN</td>
<td>Announce willingness to accept connections; give queue size</td>
</tr>
<tr>
<td>ACCEPT</td>
<td>Block the caller until a connection attempt arrives</td>
</tr>
<tr>
<td>CONNECT</td>
<td>Actively attempt to establish a connection</td>
</tr>
<tr>
<td>SEND</td>
<td>Send some data over the connection</td>
</tr>
<tr>
<td>RECEIVE</td>
<td>Receive some data from the connection</td>
</tr>
<tr>
<td>CLOSE</td>
<td>Release the connection</td>
</tr>
</tbody>
</table>
A Server Application Using TCP Socket Primitives

socket => bind => listen => {accept => {read | recvfrom => write | sendto}* }* => close | shutdown

- Create a socket,
- Bind it to a local port,
- Set up service with indication of maximum number of concurrent services,
- Accept requests from connection oriented clients,
- receive messages and reply to them,
- Terminate.
A Client Application Using TCP Socket Primitives

socket => [bind =>] connect => {write | sendto => read | recvfrom }* => close | shutdown

- Create a socket,
- Bind it to a local port,
- Establish the address of the server,
- Communicate with it,
- Terminate.
- If bind is not used, the kernel will select a free local port.
Application Layer Protocols

- Network Applications Requirements
- Application Layer Protocol Functions.
- Sample Internet Applications & Protocols:
  - File Transfer Protocol (FTP).
  - Sending E-Mail: SMTP.
  - HyperText Transfer Protocol (HTTP).
- Domain Name System (DNS)
# Common Network Applications Requirements

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Data Loss</th>
<th>Bandwidth Requirements</th>
<th>Latency sensitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>File transfer</td>
<td>No loss</td>
<td>Variable</td>
<td>none</td>
</tr>
<tr>
<td>Web documents</td>
<td>No loss</td>
<td>Variable</td>
<td>none</td>
</tr>
<tr>
<td>Real-time audio/video</td>
<td>Loss-tolerant</td>
<td>Audio: few Kbps to 1Mbpsyes</td>
<td>100's of msec</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Video: 10's Kbps to 5 Mbps</td>
<td></td>
</tr>
<tr>
<td>Stored audio/video</td>
<td>Loss-tolerant</td>
<td>Same as interactive audio/video</td>
<td>few seconds</td>
</tr>
<tr>
<td>Interactive games</td>
<td>Loss-tolerant</td>
<td>Few Kbps to 10's Kbps</td>
<td>100's msecs</td>
</tr>
<tr>
<td>Financial applications</td>
<td>No loss</td>
<td>Variable</td>
<td>Application-dependent</td>
</tr>
</tbody>
</table>
Domain Name System (DNS)

- DNS is a hierarchical system, based on a distributed database, that uses a hierarchy of Name Servers to resolve Internet host names into the corresponding IP addresses required for packet routing by issuing a DNS query to a name server.
- Name servers are usually Unix machines running the Berkeley Internet Name Domain (BIND) software.
- On many Unix-based machines using the sockets-API, gethostbyname() is the library routine that an application calls in order to issue a DNS query.
- Resource record: Associated with each host on the Internet, includes IP address, domain name, domain name server, etc.
- When resolving a host name, DNS returns the associated resource record of the host.
- Internet domain names are divided into generic top-level domains (edu, com, gov, mil) which include all US domains and country domains.
- The DNS space is divided into non-overlapping zones.
- Resource records of all hosts in a sub-domain are kept as a DNS database stored at the domain name server responsible for that sub-domain or zone.
Zone Division of DNS Name Space
Recursive DNS Queries Example

A two-level name server hierarchy is shown here as an example. In reality, several levels of name servers may be queried recursively.

A network application running on beast.isc.rit.edu issues a DNS query using gethostbyname() to resolve hostname halcyon.usc.edu.

Returns DNS Resource record for halcyon.usc.edu including IP address(s)
Properties of Secure Network Communication

- **Secrecy:** Only the sender and intended receiver should be able to understand the contents of the transmitted message.
  - Because eavesdroppers may intercept the message, this necessarily requires that the message be somehow encrypted.
  - This aspect of secrecy is probably the most commonly perceived meaning of the term "secure communication."

- **Authentication:** Both the sender and receiver need to confirm the identity of the other party involved in the communication - to confirm that the other party is indeed who or what they claim to be.
  - Most common authentication method used: password protection.
  - Other: using Public key encryption, SSL.

- **Message Integrity:** Even if the sender and receiver are able to authenticate each other, they also want to insure that the content of their communication is not altered, either maliciously or by accident, in transmission (e.g. using CRCs).
Secret-Key Encryption Algorithms

- Complex encryption algorithms that rely on series of transpositions and substations.
- **P-box**: Performs a specific permutation on input characters/bits.
- **S-box**: Performs a specific substitution on input character/bits.
- **Product cipher**: Encryption using a series of P and S boxes.
Public Key Encryption

- Encryption and decryption keys are different:
  - Public key is known and made public.
  - Private key secret and is held by owner.
- To encrypt a message: The recipient's public key along with the sender's private key are used.
- To decrypt a message the receiver's private key along with the sender's public key are used.
- Digital Signature: Encrypt using private key of user. Decrypt using public key. Only owner of private key could have generated original message
- Example Algorithm: The RSA (Rivest, Shamir, Adleman) Algorithm

Example:

Alice's computer

Transmission line

Bob's computer

\[ P \rightarrow D_A(P) \]

\[ E_B(D_A(P)) \rightarrow P \]
Data Compression Basics

• Main motivation: The reduction of data storage and transmission bandwidth requirements.
  – Example: The transmission of high-definition uncompressed digital video at 1024x 768, 24 bit/pixel, 25 frames requires 472 Mbps (~ bandwidth of an OC9 channel), and 1.7 GB of storage for one hour of video.

• A digital compression system requires two algorithms: Compression of data at the source (encoding), and decompression at the destination (decoding).

• For stored multimedia data compression is usually done once at storage time at the server and decoded upon viewing in real time.

• Types of compression:
  – Lossless: Decompressed (decoded) data is identical to the source,
    • Required for physical storage and transmission.
    • Usually algorithms rely on replacing repeated patterns with special symbols without regard to bit stream meaning (Entropy Encoding).
  – Lossy: Decompressed data is not 100% identical to the source.
    • Useful for audio, video and still image storage and transmission over limited bandwidth networks.
Entropy

• For a statistically independent source, the extension of the previous definition of average information, is called Entropy $H(s)$:

$$H(s) = -\sum_{i=1}^{n} p_i \log(p_i)$$

• It can be seen as the probability-weighted average of the information associated with a message.

• It can be interpreted much like it is in thermodynamics, the higher the entropy, the greater its lack of predictability.
  ⇒ Hence, more information is conveyed in it.

• This leads to the relationship:

  Average Length $\geq H(S)$
Entropy (continued)

Example:
Assume a three symbol alphabet \{A, B, C\} with symbol probabilities given as:

\[ P(A) = 0.5 \quad P(B) = 0.33 \quad P(C) = 0.167 \]

- In this case, Entropy: \( H(S) = 1.45 \) bits

- If \( P(A) = P(B) = P(C) = 1/3 \)
  - Then \( H(S) = 1.58 \) bits

- This demonstrates that sources which have symbols with high probability should be represented by fewer bits.

- In reality, information is not always coded using the theoretic entropy (optimal encoding).

- The excess is defined as redundancy and is seen in character repetition, character distribution, usage patterns, and from positional relativeness.
Lossless Compression

- All lossless compression methods work by identifying some aspect of non-randomness (redundancy) in the input data, and by representing that non-random data in a more efficient way.

- There are at least two ways that data can be non-random.
  - Some symbols may be used more often than others, for example in English text where spaces and the letter E are far more common than dashes and the letter Z.
  - There may also be patterns within the data, combinations of certain symbols that appear more often than other combinations. If a compression method was able to identify that repeating pattern, it could represent it in some more efficient way.

- Examples:
  - Run-Length Encoding (RLE).
  - Huffman Coding.
  - Arithmetic coding.
  - Shannon-Fano Coding.
  - LZ78, LZH, LZW, … etc.
Lossless Compression Example:

**Run-Length Encoding (RLE)**

- Works by representing runs of a repeated symbol by a count-symbol combination.
- Run-length encoding may be used as one of the steps of more elaborate compression schemes.
- Example:
  - The input string: aaaaaaabbaaabbbbbcccccc
  - Becomes: <esc>7abb<esc>3a<esc>4b<esc>5c
  - 7 bytes are saved.
- Effective only for runs greater than 3 bytes.
- Applications are limited.
Lossless Compression Example: Shannon-Fano Coding

- Number of bits required to represent a code is proportional to the probability of the code.
- Resulting Codes are uniquely decodable. They obey a prefix property.
- Compressor requires two passes or a priori knowledge of the source.
- Decompressor requires a priori knowledge or communication from compressor.
- Develop a probability for the entire alphabet, such that each symbol’s relative frequency of appearance is known.
- Sort the symbols in descending frequency order.
- Divide the table in two, such that the sum of the probabilities in both tables is relatively equal.
- Assign 0 as the first bit for all the symbols in the top half of the table and assign a 1 as the first bit for all the symbols in the lower half. (Each bit assignment forms a bifurcation in the binary tree.)
- Repeat the last two steps until each symbol is uniquely specified.
Shannon-Fano Coding Example

Symbol Encoding Binary Tree forming code table.
Huffman Coding

- Similar to Shannon-Fano in that it represents a code proportional to its probability.
- Major difference: Bottom Up approach.
- Codes are uniquely decodable. They obey a prefix property.
- Develop a probability for the entire alphabet, such that each symbol’s relative frequency of appearance is known.

- Sort the symbols, representing nodes of a binary tree, in prob. order
- Combine the probabilities of the two smallest nodes and create/add a new node, with its probability equal to this sum, to the list of available nodes.
- Label the two leaf nodes of those just combined as 0 and 1 respectively, and remove them from the list of available nodes.
- Repeat this process until a binary tree is formed by utilizing all of the nodes.
Huffman Coding Example

Symbol Encoding Binary Tree
forming Huffman code table.

<table>
<thead>
<tr>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
</tr>
<tr>
<td>T</td>
</tr>
<tr>
<td>R</td>
</tr>
<tr>
<td>H</td>
</tr>
<tr>
<td>O</td>
</tr>
<tr>
<td>S</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>Y</td>
</tr>
</tbody>
</table>
Lossy Compression

• Takes advantage of additional properties of the data to produce more compression than that possible from using redundancy information alone.

• Usually involves a series of compression algorithm-specific transformations to the data, possibly from one domain to another (e.g. to frequency domain in Fourier Transform), without storing all the resulting transformation terms and thus losing some of the information contained.

• Examples:
  – Differential Encoding: Store the difference between consecutive data samples using a limited number of bits.
  – Discrete Cosine Transform (DCT): Applied to image data.
  – Vector Quantization.
  – JPEG (Joint Photographic Experts Group).
  – MPEG (Motion Picture Experts Group).
Lossy Compression Example:

Vector Quantization

Image is divided into fixed-size blocks

A code book is constructed which has indexed image blocks of the same size representing types of image blocks.

Instead of transmitting the actual image:

The code book is transmitted + The corresponding block index from the code book or the closest match block index is transmitted.

```
0 0 1 0
2 2 2 0
3 2 2 0
0 4 0 0
```

Encoded Image Transmitted
Lossy Compression Example:

Joint Photographic Experts Group (JPEG)

JPEG Lossy Sequential Mode

JPEG Compression Ratios:
- 30:1 to 50:1 compression is possible with small to moderate defects.
- 100:1 compression is quite feasible for very-low-quality purposes.
JPEG Overview

Transform: Discrete cosine transform
Quantize: Quantization
Encode: Differential quantization, run-length encoding, statistical output encoding

Decompression:
Reverse the order

RGB to YIQ (optional)

Block Preparation

for each plane (scan)

for each 8x8 block

01101... Huffman or arithmetic

DPCM

Zig-zag

EECC694 - Shaaban