End-to-End Communication

Goal: Interconnect multiple LANs.

Why?

- Physical limitations on the number of hosts and distance of link.
- Performance limitations (even in the absence of physical ones). When valid?

Problems:

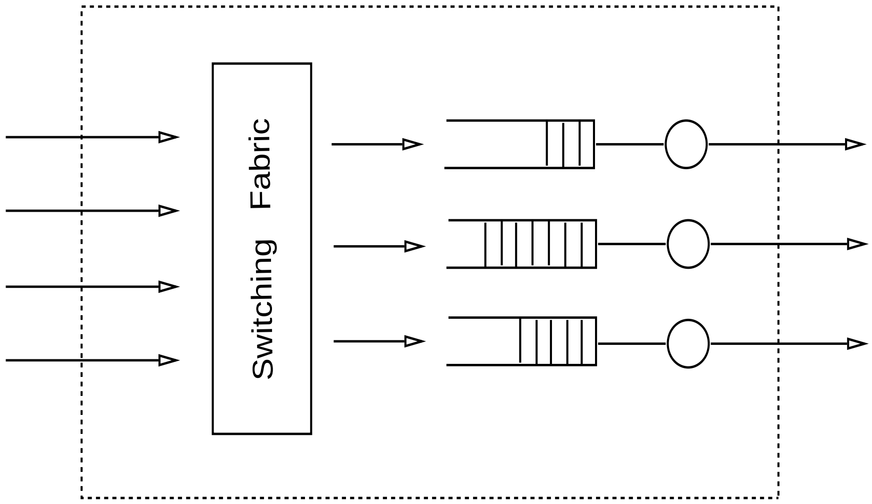
- Diverse LANs; how to make them talk to each other?
- How to choose paths? (Graph and network flow problem—routing.)
- How to dynamically regulate flow? (Congestion control problem.)

- How to provision *differentiated* quality-of-service (QoS)?
- How to render *transparent* and *efficient* network services (network computing)?
- How to achieve fault-tolerance?

Packet switching vs. circuit switching

Switch design:

- Hardware (e.g., shuffle-exchange network).
- Software (workstation as router or gateway).
- Hybrid (e.g., DSP).

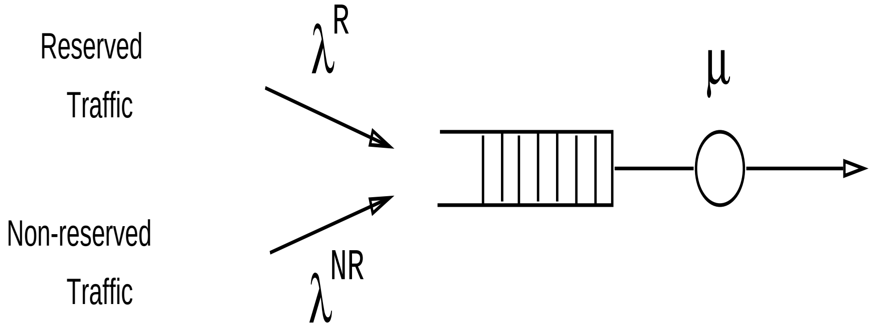


Problem with input-buffered switch design:

$\longrightarrow$ head-of-line blocking

In general, less efficient than output-buffered switches.

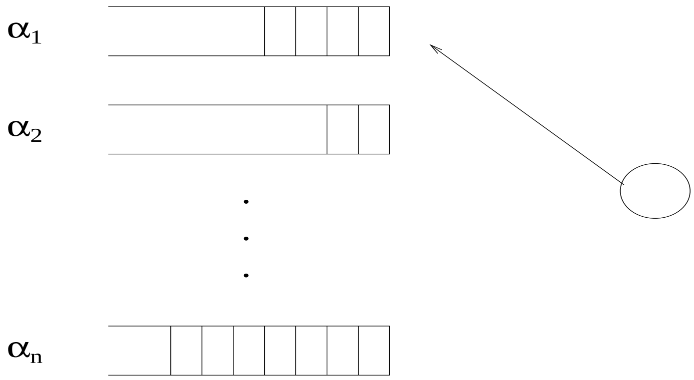
Logical switch:



- Enqueueing (who-to-drop, overwrite).
- Dequeueing (FIFO, fair queueing).

Fair queueing

Given n sources and priority weights $\alpha_1, \alpha_2, \ldots, \alpha_n$, perform *weighted* round-robin on n queues $q_1, q_2, \ldots, q_n$.



Given Δt service time, dequeue $\alpha_i \Delta t$ packets (bits) from queue q_i.

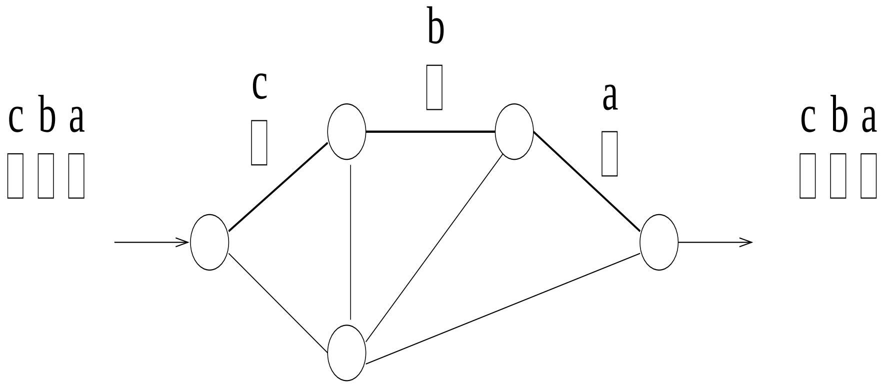
As $\Delta t \to 0$, more finegrained and fair. In practice, need approximation for $\Delta t \gg 0$.

Circuit switching

Establish fixed path (route) from source to destination—*channel*.

$\longrightarrow$ *connection-oriented*

All packets belonging to the same connection traverse same path.



- Permanent virtual circuits (PVC). E.g., line leasing.
- Switched virtual circuits (SVC). E.g., regular telephone calls.

Benefit: Simplicity.

- One-time call set-up cost (admission control).
- Allows simplified switch design.
- Under low packet loss rate, in-order delivery.
- Easier accounting for reservation-based resource allocation.

Drawback: Performance.

- For lengthy connection, "goodness" of initial path may change.
- High initial call set-up cost (real-time applications).
- Less fault-tolerant.

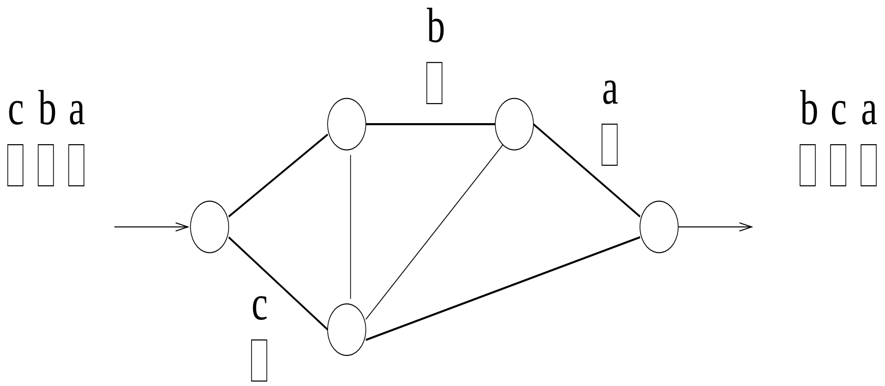
Packet switching

Treat each packet as *independent* unit, with full source/ destination addressing.

Even during one conversation, packets may take different routes.

⟶ *store-and-forward* networks

⟶ *connectionless* service (need to be careful)



Benefits: Performance.

- Can adaptively find "good" path for each packet of a conversation.
- More fault-tolerant.
- More responsive to interactive real-time applications.

Drawback: Increased complexity.

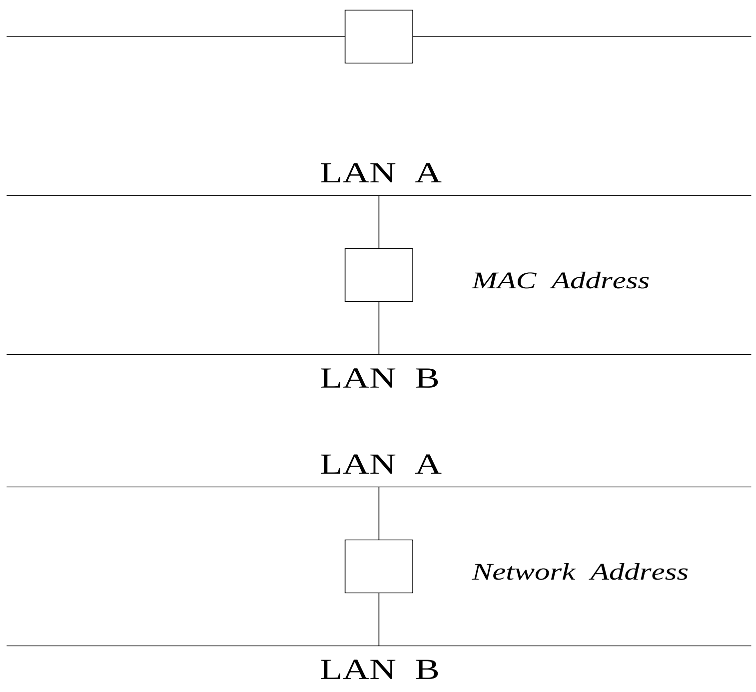
- Switch design is more complex.
- Increased processing overhead incurred at switches.
- Out-of-order delivery—resequencing cost.

"Message switching."

Interconnecting LANs

Methods:

- Repeaters (physical layer).
- Bridges (data link layer).
- Routers or gateways (network layer).



Bridges

Properties:

- Promiscuous mode.
- Backward learning (track source addresses).
- Transparency (e.g., IEEE LAN standards).
- Spanning Tree (loop problem).
  - Goal: Build spanning tree rooted at lowest ID (serial number) bridge.
  - Send out/forward configuration messages containing smallest *locally observed* ID with distance information.
  - Stop generating and only forward if own priority is overridden.
  - Update shortest distance by 1.
  - Eventually stabilizes to shortest path solution.
  - Perlman's method is a form of *self-stabilization*.
- Source routing.

Routers

Maintain shortest-path table to relevant nodes. Forward network layer packets (e.g., IP datagrams) based upon this table.

$\longrightarrow$ routing problem

Actually:

- If network address matches local address, then use ARP to look up MAC address of the destination and pass to data link layer.
- If it does not, then use ARP to look up MAC address of next hop and pass to data link layer.

Benefit of bridge:

- Simple form of modularization.
- Achieves load splitting (performance), fault-tolerance, security.

Drawback of bridge design vis-a-vis routers?

Internet Protocol (IP)

Goals:

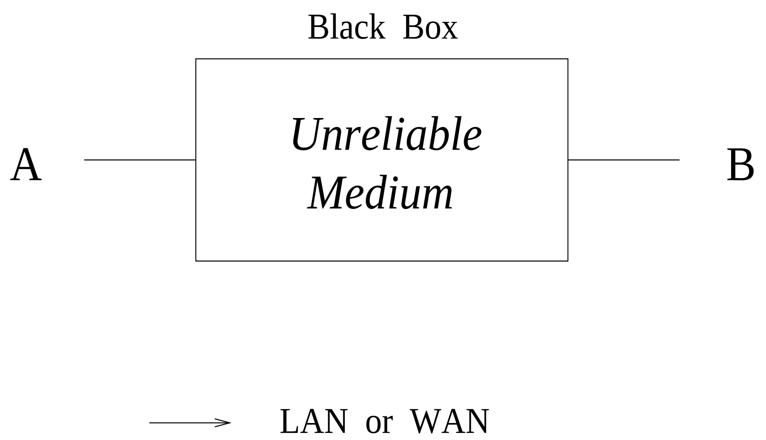
- Interconnect diverse LANs into one logical entity.
- Implement *best effort* (unreliable, connectionless) service model.

Thus, specifies

- Common language for carrying out non-LAN-specific conversations (protocol standards).
- Functionality and design philosophy.

Best effort vs. guaranteed service:

- Much easier to implement best effort service; simplifies router design.
- Increases complexity of end stations; trade-off.
- Necessitates higher-up functional layer—*transport layer*—to achieve reliable transmission over unreliable medium.
- Duplicates work.
- Routers/switches already becoming more complex due to QoS; why not dispense with transport layer . . .



IP packet (datagram) format:

4	4	8	16	
version	header length	TOS	total length	
fragmentation identifier			flags	fragment offset
TTL		protocol	header checksum	
source address				
destination address				
options (if any)				

- Header length: in 4 byte (word) units).
- TOS (type-of-service): Most routers do not support.
- 4 bytes used for fragmentation.
- TTL (time-to-live): Prevent cycling (default 64).
- Protocol: demultiplexing key (TCP 6, UDP 17).

Fragmentation and reassembly:

LAN has *maximum transmission unit* (MTU)—maximum frame size; e.g., Ethernet 1500B, FDDI 4500B.

$\longrightarrow$ potential size mismatch problem (64kB)

Solution: Fragment IP packet when needed, maintain sequencing information, then reassemble at destination.

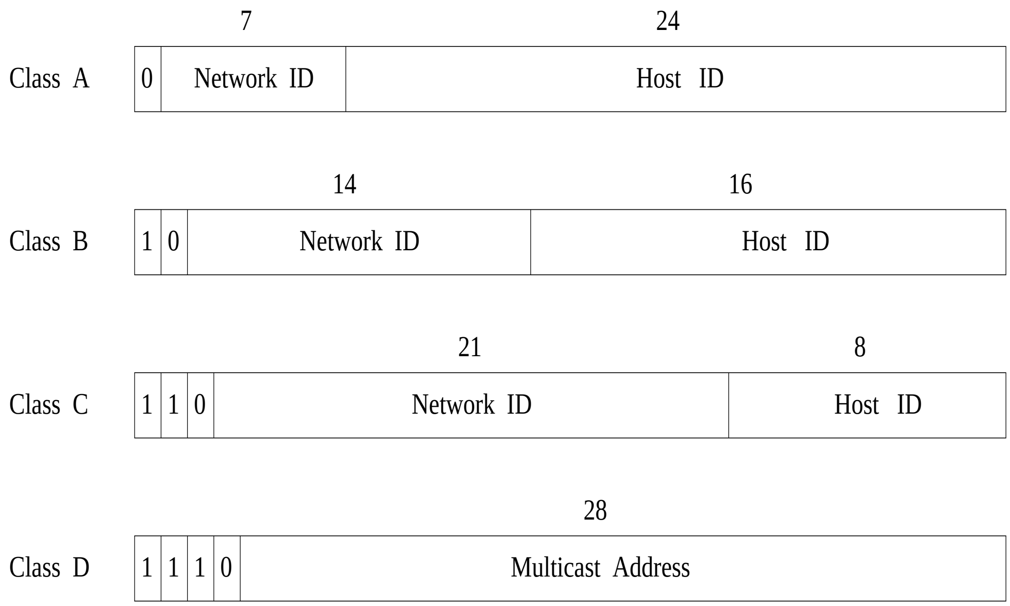
- Assign unique fragmentation ID.
- Set 3rd flag bit if fragmentation in progress.
- Sequence fragments using offset in units of 8 bytes.

Note: Each fragment is an *independent* IP packet.

Destination discards all fragments of an IP packet if one is lost.

TCP: Negotiate at start-up TCP segment (packet) size based on MTU; 1kB or 512B are common. Seek compatibility with IP.

IP address format:



Dotted decimal notation: 10000000 00001011 00000011 00011111 $\leftrightarrow$ 128.11.3.31

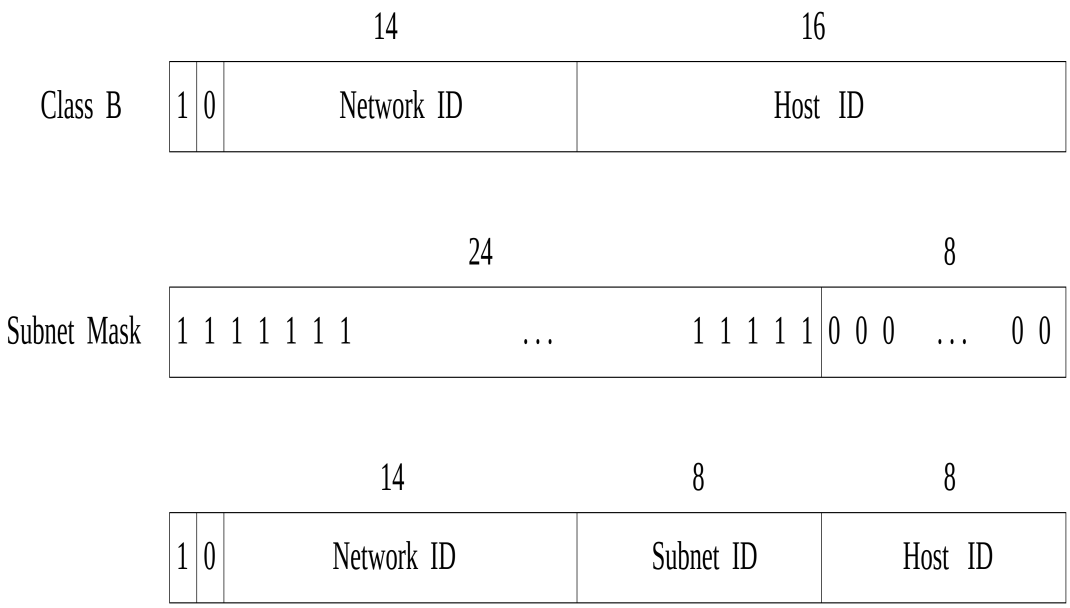
Symbolic name to IP address translation—domain name server (DNS).

Notice hierarchical organization (2-level).

Each interface (NIU) has an IP address; single host can have multiple IP addresses.

Potential problem: Waste of address space. Giving each network own network ID is inefficient.

Solution: *Subnetting*; group several physical networks into one.



To determine subnet ID:

- Perform ANDing of IP address and subnet mask.
- Needed for routing.

Forwarding and address resolution:

... mechanics of routing when *routing table* is given.

Subnet ID	Subnet Mask	Next Hop
128.10.2.0	255.255.255.0	Interface 0
128.10.3.0	255.255.255.0	Interface 1
128.10.4.0	255.255.255.0	128.10.4.250

Either destination host is directly connected on the same LAN or not.

Table look-up I:

- For each entry, compute *DestSubnetID* = *DestAddr* AND *SubnetMask*.
- Compare *DestSubnetID* with *SubnetID* and take action.

One more task left: Translate destination host (or next hop node) IP address into LAN address.

$\longrightarrow$ address resolution protocol (ARP)

Table look-up II:

- If ARP table contains entry. using LAN address send to destination.
- If ARP table does not contain entry, broadcast ARP Request packet with destination IP address.
- Update ARP table upon receipt of feedback.

Dynamically maintain ARP table:

- Use timer for each entry (15min) to invalidate entries.
- Upon receipt of ARP Request (if applicable), update own ARP if entry is absent; ARP Request frame contains source IP address and LAN address.