End-to-End Communication

Goal: Interconnect multiple LANs.

Why?

- Physical limitations on the number of hosts and distance of link.
- Intrinsic performance limitations.

Problems:

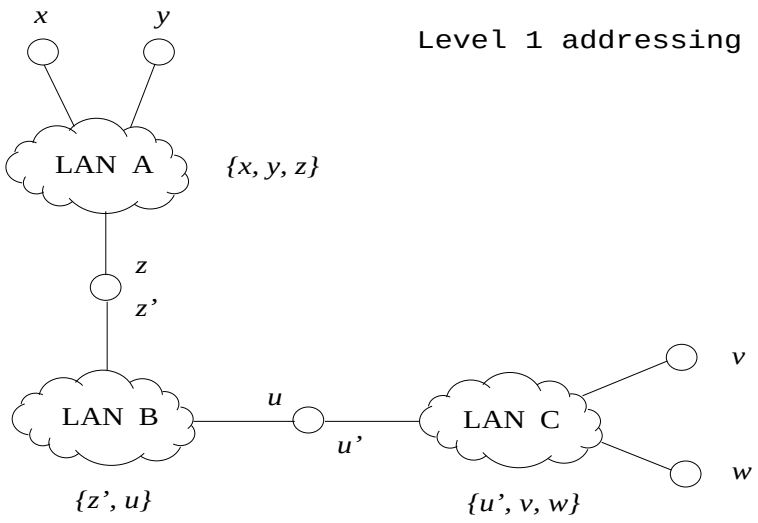
- Diverse LANs; how to make them talk to each other (internetworking)?
- How to choose paths (routing)?
- How to dynamically regulate flow (congestion control)?

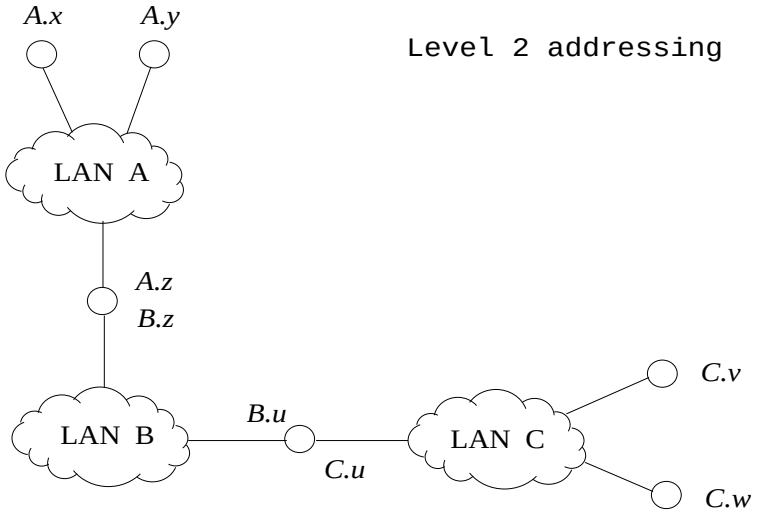
- How to provide QoS (network support/end system support)?
- How to render transparent and efficient network services (e.g., network computing)?
- How to achieve robustness and fault-tolerance?

Translation problem of internetworking:

- address translation (LAN addr. $\leftrightarrow$ WAN addr.)
- protocol translation
  - frame format
  - MAC

  $\longrightarrow$ minimum necessary mechanism

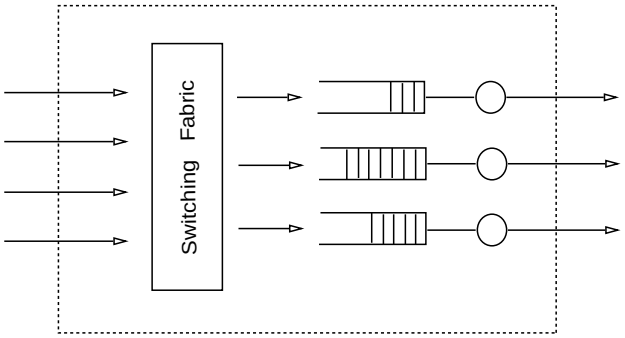

x
y
Level 1 addressing
LAN A
{x, y, z}
z
z'
v
u
LAN B
LAN C
u'
w
{z', u}
{u', v, w}


A.x
A.y
Level 2 addressing
LAN A
A.z
B.z
C.v
B.u
LAN B
LAN C
C.u
C.w

Packet switching vs. circuit switching

Switch design:

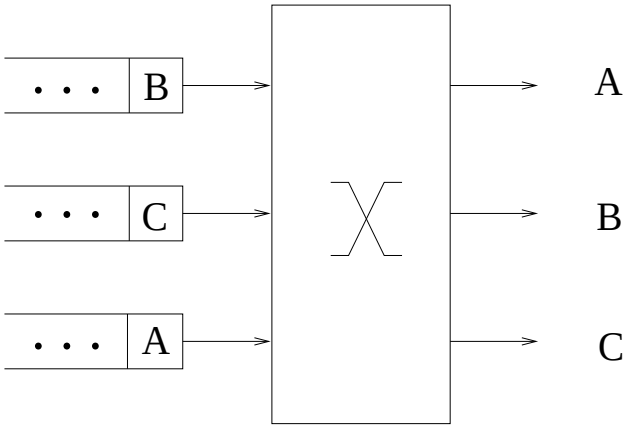
- Hardware (e.g., shuffle-exchange network).
- Software (workstation as router or gateway).
- Hybrid (e.g., DSP).



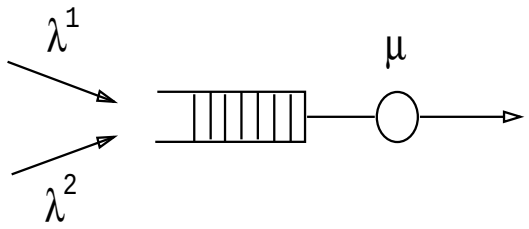
Problem with input-buffered switch design:

$\longrightarrow$ head-of-line blocking

In general, less efficient than output-buffered switches.



Logical switch:



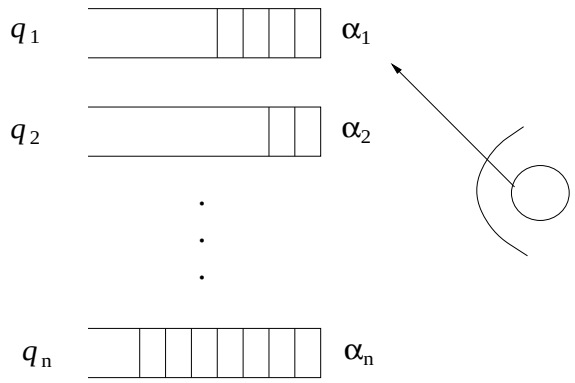
- Enqueueing (who-to-drop, overwrite).
- Dequeueing (FIFO, weighted fair queueing).

$\longrightarrow$ scheduling with real-time constraints (O.S.)

$\longrightarrow$ real-time systems community

Weighted 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.

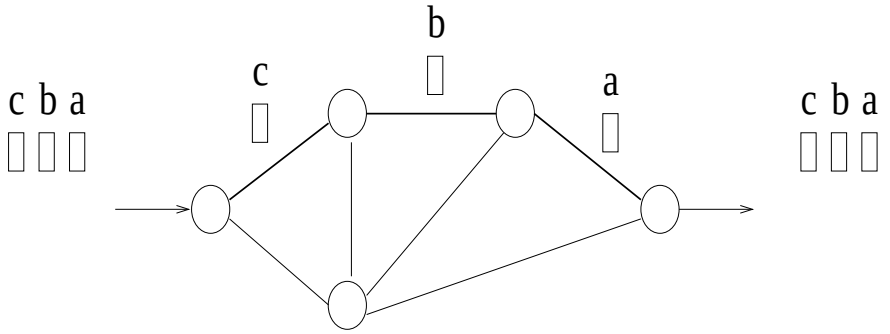
As $\Delta t \to 0$, more finegrained and fair. In practice, need approximation for $\Delta t \gg 0$.

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).
- Smaller routing table.
- 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.

$\longrightarrow$ solution adopted for ATM networks

Following a fixed path:

- source routing
- call set-up

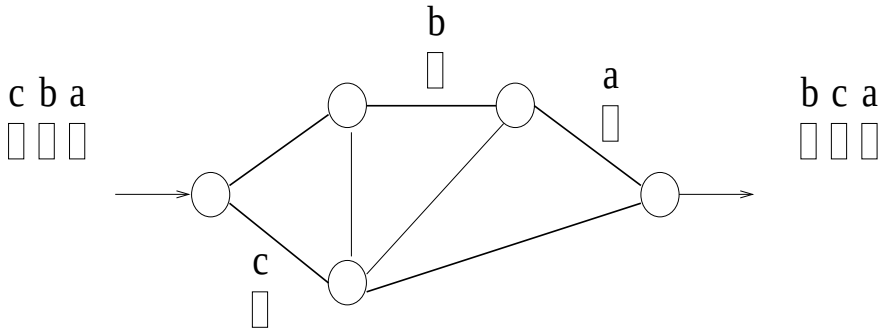
Packet switching

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

$\longrightarrow$ packet as fully autonomous entity

During single conversation, packets may take different routes.

$\longrightarrow$ store-and-forward networks



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.
- bigger routing table.
- Increased processing overhead incurred at switches.
- Out-of-order delivery—re-sequencing cost.

"Message switching."

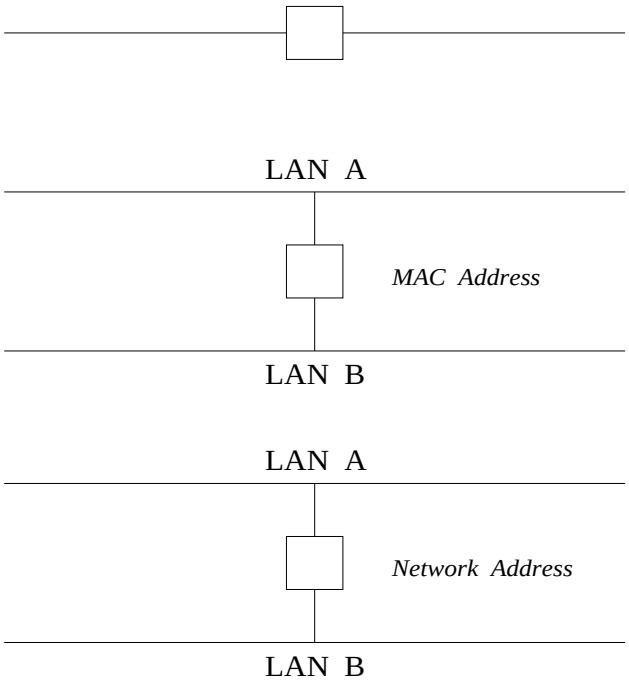
Active network proposal.

$\longrightarrow$ packet contains *program* and *data*

Interconnecting LANs

Methods:

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



Bridges

- 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*.

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.

$\longrightarrow$ two-level addressing

Benefit of bridge:

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

Drawback of bridge-based design vis-à-vis routers?

Fragmentation and reassembly:

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

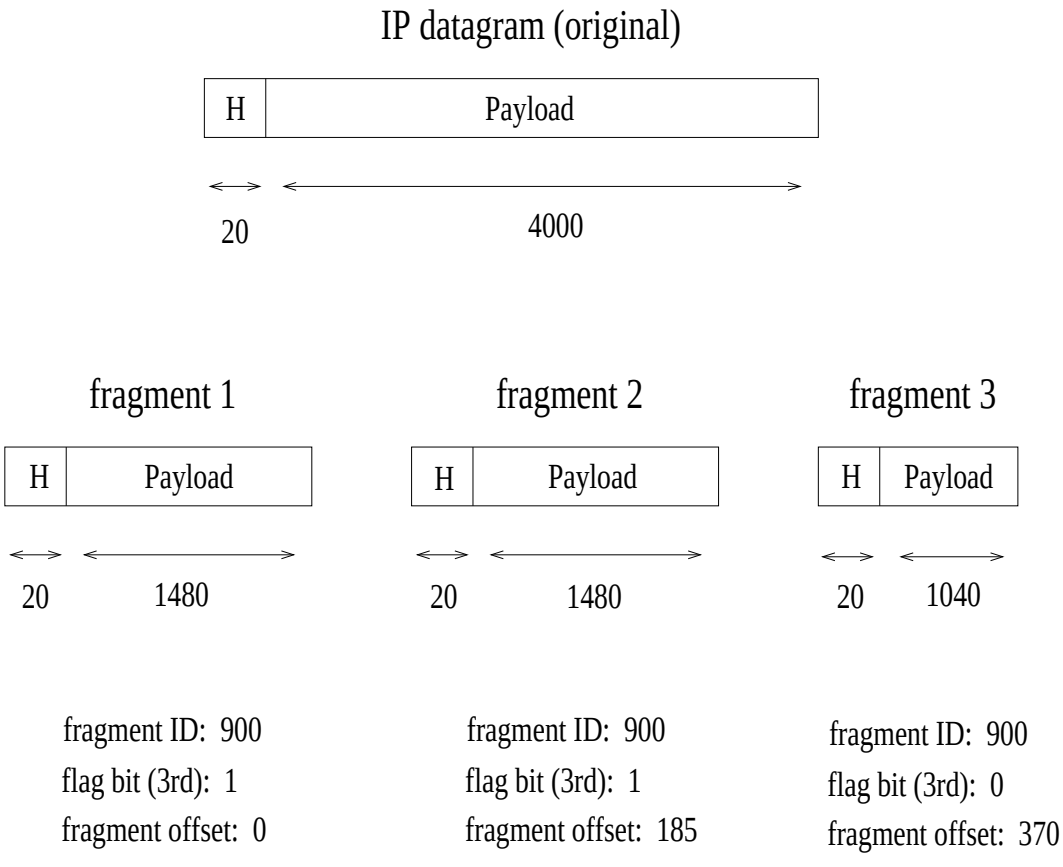
$\longrightarrow$ potential size mismatch problem (64 kB)

$\longrightarrow$ . . . when hopping from LAN to LAN

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.

Example: IP fragmentation (Ethernet MTU)



fragment ID: 900
flag bit (3rd): 1
fragment offset: 0

fragment ID: 900
flag bit (3rd): 1
fragment offset: 185

fragment ID: 900
flag bit (3rd): 0
fragment offset: 370

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

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

$\longrightarrow$ fragmentation problem

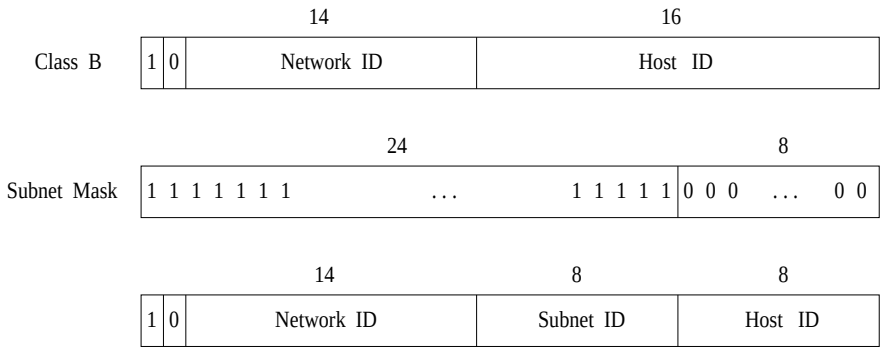
$\longrightarrow$ exists at several boundaries in protocol stack

$\longrightarrow$ set 2nd flag bit to disable fragmentation

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

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.
- 3-level hierarchy (IP).

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.
- Encapsulate ARP packet into LAN frame.
- Update ARP table upon receipt of feedback.

Dynamically maintain ARP table:

- Use timer for each entry (15 min) 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.

Standards documents: RFC (Request for Comments)

- RFC 791 (IP)
- RFC 826 (ARP)
- RFC 903 (RARP)
- RFC 894 (Ethernet)
- RFC 793 (TCP)
- RFC 768 (UDP)
- etc.