CSE 30264

Computer Networks

Prof. Aaron Striegel
Department of Computer Science & Engineering
University of Notre Dame

Lecture 7 – February 2, 2010

Today's Lecture

Read Chapter 3.2 f

- Project 1
  - Q&A
- Packet Forwarding
  - Shared Access
  - Chapter 3.1

Application
Transport
Network
Data
Physical

Read Chapter 3.2 for Thursday

Project 1

- Example
  - Client
  - Telnet to debug port (7000)

Shared Access Networks

Outline
Bus (Ethernet)
Token ring (IBM, FDDI, RPR)
Wireless (802.11, WiMAX)

Wi-Fi

- IEEE 802.11: 2.4 GHz band, 1 Mb/s
- IEEE 802.11b: 2.4GHz band, 11Mb/s
- IEEE 802.11a: 5GHz band, 54Mb/s
- IEEE 802.11g: 2.4GHz band, 54Mb/s
- IEEE 802.11n: Dual band (2.4, 5), 270 Mb/s

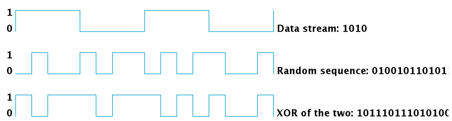
MIMO Block Ack

Spread Spectrum

- Idea
  - Spread signal over wider frequency band than required
  - Originally designed to thwart jamming
- Frequency Hopping
  - Transmit over random sequence of frequencies
  - Sender and receiver share…
    - Pseudorandom number generator
    - Seed
  - 802.11 uses 79 x 1MHz-wide frequency bands

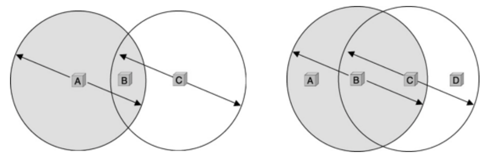
Spread Spectrum (cont)

- Direct Sequence
  - For each bit, send XOR of that bit and *n* random bits
  - Random sequence known to both sender and receiver
  - Called *n*-bit *chipping code*
  - 802.11 defines an 11-bit chipping code



Collision Avoidance

- Similar to Ethernet
- Problem: *hidden* and *exposed* nodes



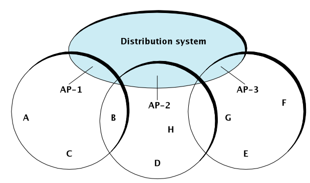
MACA

- Multiple Access with Collision Avoidance
- Sender transmits **`RequestToSend`** (RTS) frame
- Receiver replies with **`ClearToSend`** (CTS) frame
- Neighbors…
  - see CTS: keep quiet
  - see RTS but not CTS: ok to transmit
- Receiver sends ACK when has frame
  - neighbors silent until see ACK
- Collisions
  - no collision detection
  - known when CTS not received
  - exponential backoff

In general,
bad, bad, bad

Supporting Mobility

- Case 1: *ad hoc* networking
- Case 2: *access points* (AP)
  - Tethered
  - Each mobile node associates with an AP



Mobility (cont)

- Scanning (selecting an AP)
  - Node sends **`Probe`** frame
  - All AP's w/in reach reply with **`ProbeResponse`** frame
  - Node selects one AP; sends it **`AssociateRequest`** frame
  - AP replies with **`AssociationResponse`** frame
  - New AP informs old AP via tethered network
- When
  - Active: when join or move
  - Passive: AP periodically sends **`Beacon`** frame

802.11

16	16	48	48	48	16	48	0–18,496	32
Control	Duration	Addr1	Addr2	Addr3	SeqCtrl	Addr4	Payload	CRC

- Up to 2312 bytes of data
- 32-bit CRC
- 4 addresses, usage depends on mode:
  - Addr1 is target, Addr2 is source
  - Addr1 is ultimate target, Addr2: immediate sender, Addr3 is intermediate target, Addr4: original source

WiMAX

- Worldwide Interoperability for Microwave Access
- Standardized by WiMAX Forum, IEEE 802.16
- Typical distance: 1-6miles, up to 30miles
- “subscriber stations” (e.g., antenna on roof)
- Up to 70Mbps
- Time Division Duplexing (TDD)
- Frequency Division Duplexing (FDD)

Cell Phone Technologies

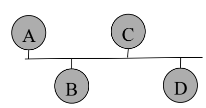
- Uses base stations, area served called “cell”
- 1G: analog
- 2G, 2.5G (e.g., GSM): digital
- GPRS: General Packet Radio Service (typically 30-70Kbps)
- 3G:
  - UMTS (Universal Mobile Telecommunications System)

Switching and Forwarding

Outline
  Store-and-Forward Switches
  Bridges and Extended LANs

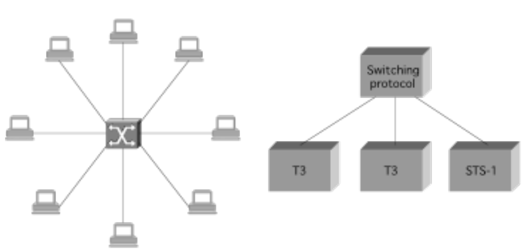
Scalability

- Shared Access
  - More users -> worse performance
  - Why -> more competition
  - Fairness vs. quality of access?

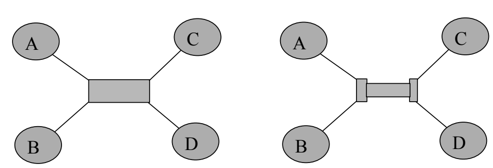


A → B, C → D
AB
CD
AB
CD

Switch



Scalability

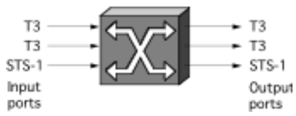


Switch allows us to logically separate the network
  Allows us to “connect on demand”

Demand =
  Destination Address

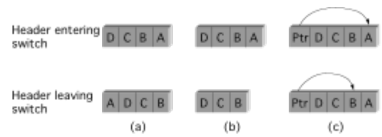
Scalable Networks

- Switch
  - Forwards packets from input port to output port
  - Port selected based on address in packet header



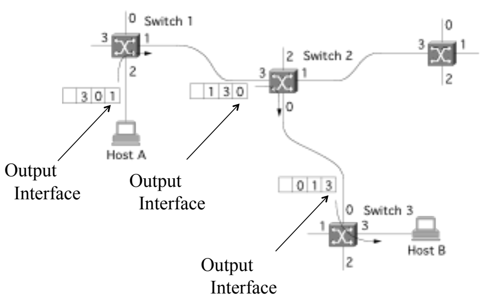
- Advantages
  - Cover large geographic area (tolerate latency)
  - Support large numbers of hosts (scalable bandwidth)

Source Routing



Use, rotate Pop, fwd Point, fwd

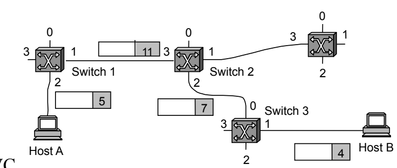
Source Routing



Virtual Circuit Switching

- Set up and tear down
- Circuit on demand
- *Connection-oriented* model

- Analogy: phone call
- Each switch maintains a VC table



Example Tables

- Circuit Table (switch 1, port 2)

VC In	VC Out	Port Out
5	11	1
6	8	1
…	…	…

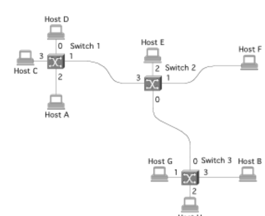
- Forwarding Table (switch 1)

Address	Port
A	2
C	3
F	1
G	1
…	…

Datagram Switching

- No connection setup phase
- Each packet forwarded independently
- Sometimes called *connectionless* model

- Analogy: postal system
- Each switch maintains a forwarding (routing) table



Virtual Circuit Model

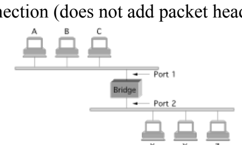
- Typically wait full RTT for connection setup before sending first data packet.
- While the connection request contains the full address for destination, each data packet contains only a small identifier, making the per-packet header overhead small.
- If a switch or a link in a connection fails, the connection is broken and a new one needs to be established.
- Connection setup provides an opportunity to reserve resources.

Datagram Model

- There is no round trip delay waiting for connection setup; a host can send data as soon as it is ready.
- Source host has no way of knowing if the network is capable of delivering a packet or if the destination host is even up.
- Since packets are treated independently, it is possible to route around link and node failures.
- Since every packet must carry the full address of the destination, the overhead per packet is higher than for the connection-oriented model.

Bridges and Extended LANs

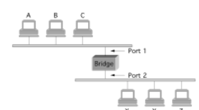
- LANs have physical limitations (e.g., 2500m)
- Connect two or more LANs with a *bridge*
  - accept and forward strategy
  - level 2 connection (does not add packet header)



- Ethernet Switch = Bridge on Steroids

Learning Bridges

- Do not forward when unnecessary
- Maintain forwarding table

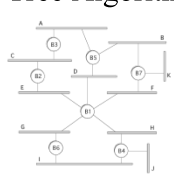


Host	Port
A	1
B	1
C	1
X	2
Y	2
Z	2

- Learn table entries based on source address
- Table is an optimization; need not be complete
- Always forward broadcast frames

Spanning Tree Algorithm

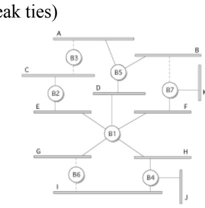
- Problem: loops



- Bridges run a distributed spanning tree algorithm
  - Select which bridges actively forward
  - Developed by Radia Perlman
  - Now IEEE 802.1 specification

Algorithm Overview

- Each bridge has unique id (e.g., B1, B2, B3)
- Select bridge with smallest id as root
- Select bridge on each LAN closest to root as designated bridge (use id to break ties)
- Each bridge forwards frames over each LAN for which it is the designated bridge



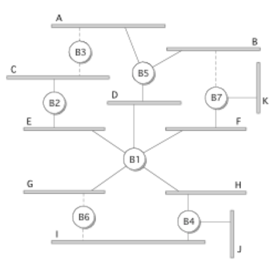
Algorithm Details

- Bridges exchange configuration messages
  - *id* for bridge sending the message
  - *id* for what the sending bridge believes to be root bridge
  - distance (hops) from sending bridge to root bridge
- Each bridge records current best configuration message for each port
- Initially, each bridge believes it is the root

Algorithm Detail (cont)

- When learn not root, stop generating config messages
  - in steady state, only root generates configuration messages
- When learn not designated bridge, stop forwarding config messages
  - in steady state, only designated bridges forward config messages
- Root continues to periodically send config messages
- If any bridge does not receive config message after a period of time, it starts generating config messages claiming to be the root

Configuration



Broadcast and Multicast

- Forward all broadcast/multicast frames
  - Current practice
- Learn when no group members downstream
- Accomplished by having each member of group *G* send a frame to bridge multicast address with *G* in source field

Limitations of Bridges

- Do not scale
  - Spanning tree algorithm does not scale
  - Broadcast does not scale
- Do not accommodate heterogeneity
- Caution: beware of transparency