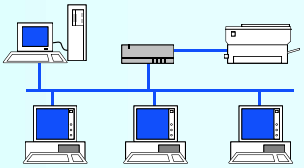


Distributed Systems
601.417
Introduction

Department of Computer Science
The Johns Hopkins University

Course Information

- Lecture/Tutorial: Tuesday 3pm – 4:15pm, Shaffer 301
- Lecture/Tutorial: Thursday 3pm – 4:15pm, Shaffer 301
- Instructor: Yair Amir
  - Office hours: Malone 209 Tuesday 5:00pm – 6:00pm
- TA : Jerry Chen
  - Office hours: Malone 207/209 Thursday 1:30pm – 2:30pm
- Special help : Yair, Jerry, and Sahiti Bommareddy
  - DSN lab – Malone 207
- Course web page: www.dsn.jhu.edu/courses/cs417/
- Slack space: you should have gotten and invite – please join

Slack is best, e-mail is possible. Or, come to office / lab

This Week and Next

- Tuesday
  - Getting to know each other
  - Introduction to the course
- Thursday
  - Introduction to the course (cont)
  - Basic network protocols
- Next Tuesday
  - Tutorial, first practical exercise
- Goal
  - By the next Tuesday, you have the information you need to decide if you want to take the course

Lets go around the room

- Name
- Department
- Degree (BS, BS/MS, MS, PhD)
- Year in degree (1, 2, 3, 4)
- Programming experience
  (C, C++, Java, etc. / school & outside of school)
- Other relevant experience (networking, systems)
- ☺ Why are you here ☺
  (what do you expect from the course ?)

Grading Policy

- Two theoretical written assignments
- Three non-equal programming exercises
- One final project, presentation date: 12/17/2021
- Attendance!
- No exam ☺

  18%+ 42% + 30% + 10% = 100%

- Ethics code: standard CS code www.cs.jhu.edu
- Zero tolerance for ethics problems
  - We invest a lot and expect a lot in return

Programming language: C or C++
Testing environment: the undergrad lab - ugrad1-24
Need to get an account!!

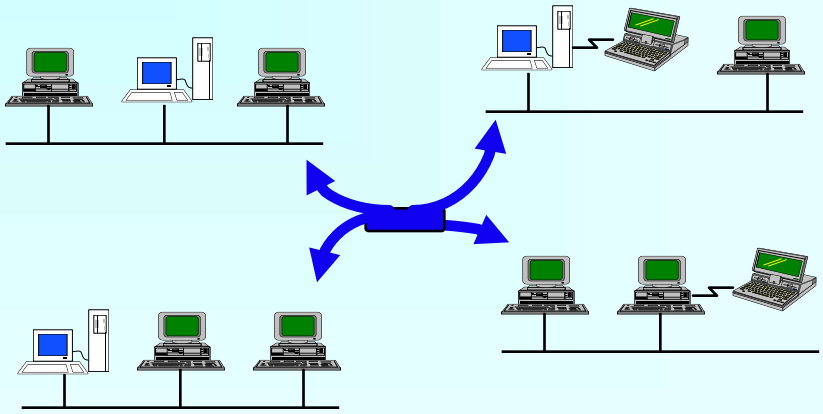
Course Overview

Lecture 1

Recommended Reading:
See course reference page:

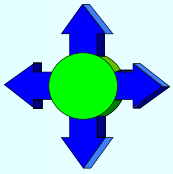
http://www.dsn.jhu.edu/courses/cs417/ref.html

A Distributed System



Why Distribute?

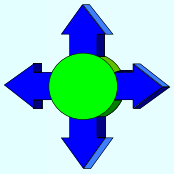
- Bridge geographic distances.
- Improve performance.
- Improve availability.
- Maintain autonomy.
- Lower the cost.
- Specialization (e.g GPUs)
- Allow for interaction.



Why Distribute?

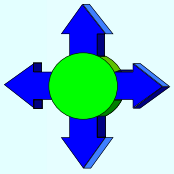
?

- Bridge geographic distances.
- Improve performance.
- Improve availability.
- Maintain autonomy.
- Lower the cost.
- Specialization (e.g GPUs)
- Allow for interaction.



Why Distribute?

- Bridge geographic distances.
- Improve performance.
- Improve availability.
- Maintain autonomy.
- Lower the cost.
- Specialization (e.g GPUs)
- Allow for interaction.



Software solutions are needed!

Shaffer 301

Course Content

- Introduction. Aug 31
- Course intro + Basic Network Protocols Sep 2
- Synchronous models in distributed environments Sep 9-14
- Messaging Systems (benign model) Sep 14–16–21–23
  - Multicast, group communication, overlay networks
- Asynchronous models in distributed environments. Sep 30 – Oct 5
- Consistent State (benign model) Oct 7–12–14–19
  - Distributed transactions, replication, Paxos, Raft, Congruity

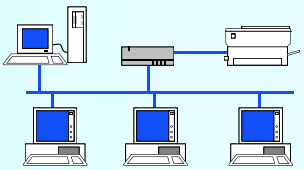
Shaffer 301

Course Content

- Intrusion-Tolerant Replication Oct 21 – 26 – 28, Nov 2
  - BFT, Prime, Proactive Recovery, applications of intrusion tolerant replication
- Intrusion-Tolerant Messaging Nov 4 – 9
- Final project discussions Nov 11, Nov 18
- Large-scale Data Stores & Probabilistic Protocols Nov 16
- Knowledge in Distributed Systems (game) Nov 30
- Course summary Dec 2

Shaffer 301

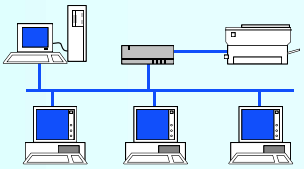
Tutorial



- Presenting and discussing exercises
- Communication using Unix sockets:
  - Reliable point-to-point communication (TCP/IP)
  - Non-reliable p-to-p communication (UDP/IP)
  - Non-reliable Broadcast, Multicast (UDP/IP)
- The Spread toolkit
- The Spines toolkit

Shaffer 301

Tutorial



- Tuesday Sep 7 – point to point tutorial
- First practical exercise – Tuesday Sep 7
- Submission of first exercise: Friday Sep 17, 11pm
- Tuesday Sep 28 – Multicast tutorial
- Second practical exercise –Tuesday Sep 28

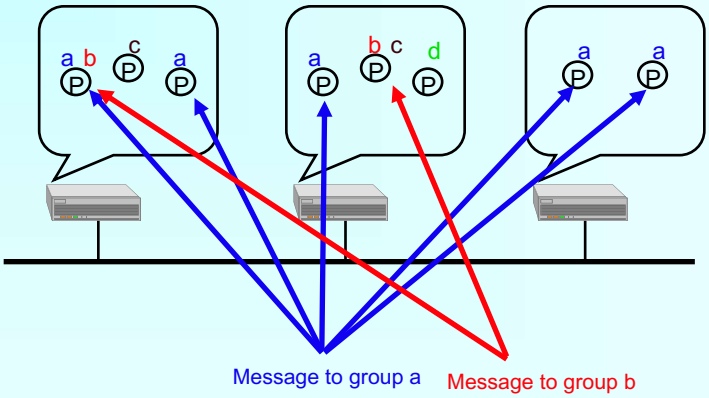
Synchronous and asynchronous models.

Some Theory :)

Messaging Systems: Multicast Protocols

- Reliability
- Ordering
- Membership
- Network span
- Routing
- Quality of Service

Messaging Systems: Group Communication

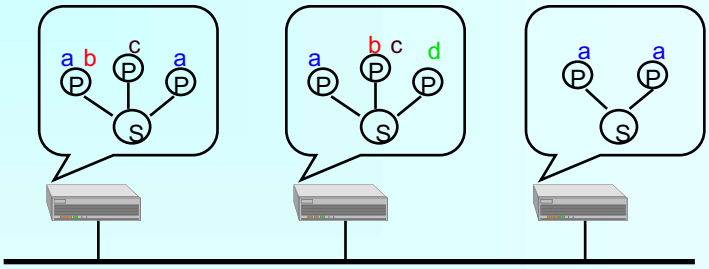


Spread: A Group Communication Toolkit

Spread provides:

- Process groups across local and wide area networks (Internet).
- Consistent service semantics:
  - Various levels of reliability
  - Various levels of message ordering
  - Membership services.
- High performance.

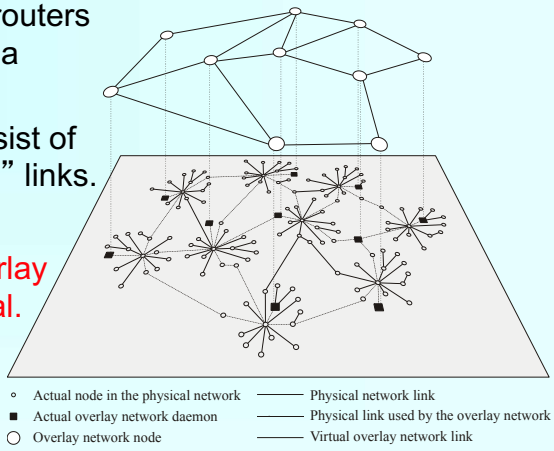
Process groups in Spread



- One Spread daemon in each machine
- Multiple destination groups per message

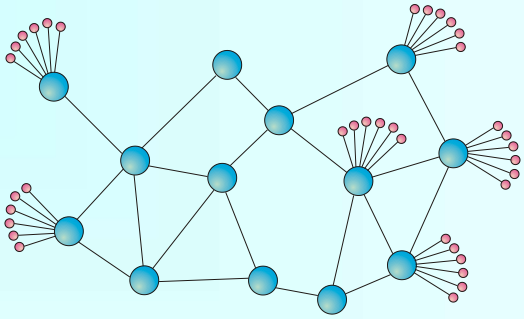
Messaging Systems: Overlay Networks

- Application-level routers working on top of a physical network.
- Overlay links consist of multiple "physical" links.
- Incurs overhead.
- Placement of overlay routers not optimal.
- **Flexible use of peer-protocols.**
- **Provides added value.**



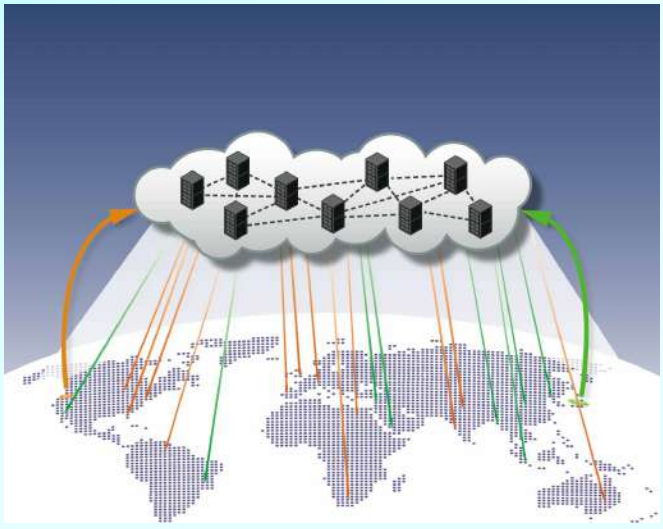
The Spines Overlay Messaging System

www.spines.org

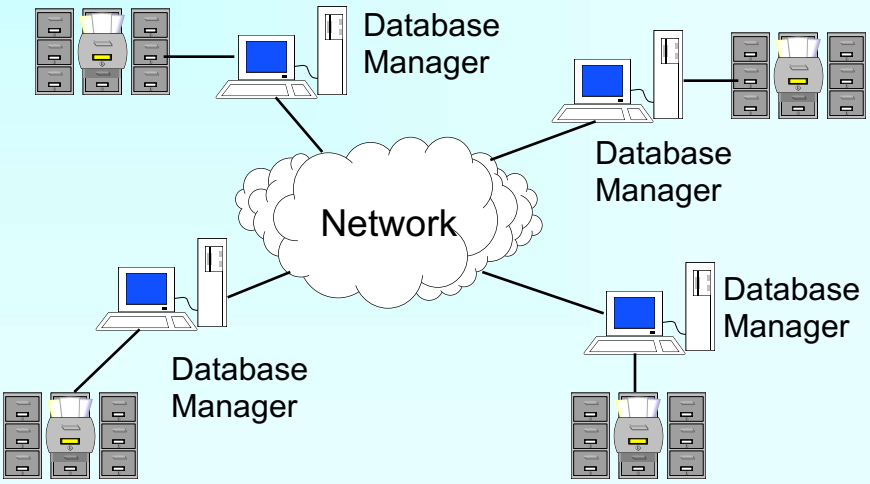


- Daemons create an overlay network on the fly
- Clients are identified by logical addresses identical to Internet addresses (IP address and a port ID)
- Clients feel they work with standard Internet protocols
- Protocols designed to support up to 1000 daemons (locations), each daemon can handle up to about 1000 clients

A Global Overlay in Action



Consistent State: A Distributed Database



Distributed Transactions

When data is spread over several database servers, there should be a way to coordinate transactions so that they will be:

- Atomic - either all effects take place, or none.
- Consistent - correct.
- Isolated - as if there was one serial database.
- Durable - effects are not lost.

Atomic Commit Protocols correctly coordinate distributed transactions.

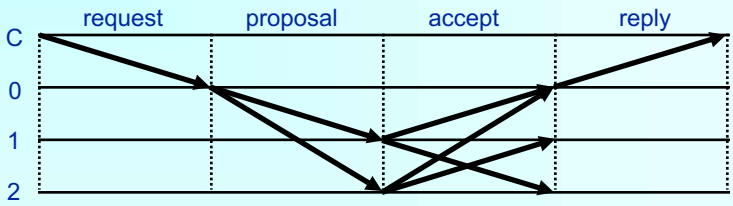
Consistent State: Replication

Considerations:

- Improve availability
- Improve performance for queries (higher throughput, lower latency)
- Cost
- Soft state / Persistent state
- Update rate / State size
- Dynamic instantiation and consolidation

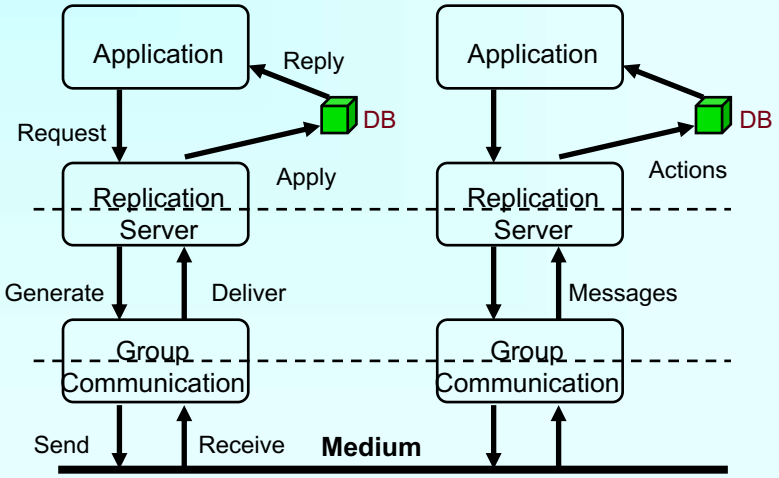
Fault model? Guarantees? Performance?

Paxos Replication

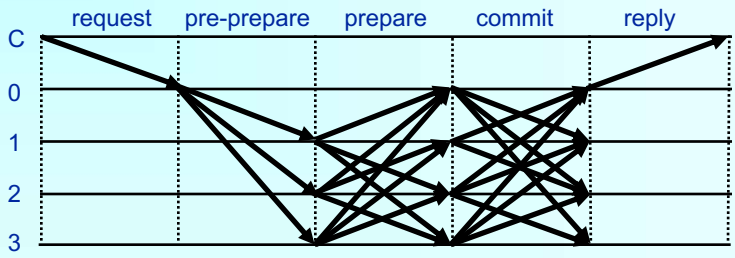


- Lamport, 1998, 2001
- A very resilient protocol. Only a majority of participants are required to make progress
- All participants are trusted
- Works well on unstable networks

Congruity Replication
Replication over Group Communication

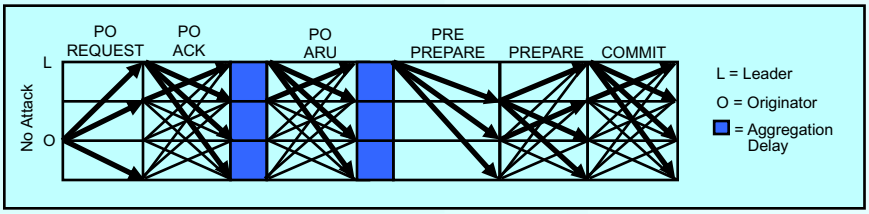


Intrusion Tolerant Replication
BFT



- Byzantine Fault Tolerance [*Castro and Liskov, 99*]
- 2/3 total servers +1 are required to make progress
- Three rounds of message exchanges
- Works even if up to (but less than) 1/3 of servers are compromised!

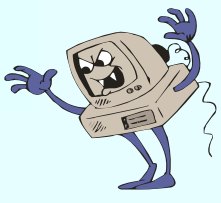
Intrusion Tolerant Replication
Prime



- Performance Guarantees under Attack [*Amir, Coan, Kirsch, Lane, 2008*]
- Works even if up to (but less than) 1/3 of servers are compromised!
- **Bounded-Delay**: There exists a time after which the update latency for any update initiated by a stable server is upper-bounded

Intrusion Tolerant Replication
Application: Critical Infrastructure

- Supervisory Control and Data Acquisition (SCADA) systems form the backbone of critical infrastructure services
- Today's systems tolerate "benign" faults but are not built to survive intrusion attacks
  - Assumption of private network no longer holds - systems move to the Internet
  - SCADA is increasingly a target for attackers
- Intrusion tolerant replication serves as an important building block for intrusion-tolerant SCADA



Intrusion Tolerant Messaging

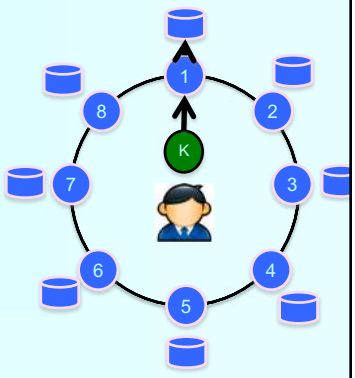
- Distributed systems can span wide area locations and rely on geographically-distributed networks to communicate
- Underlying network must be intrusion-tolerant to protect against network intrusions
- Normal routing algorithms are insufficient
  - Nodes are trusted and routing updates determine path calculations
  - Compromised nodes can disrupt the routing protocol by lying in their updates

Intrusion Tolerant Messaging

- Any node can be a source
- Any node can be compromised
- Compromised nodes may be undetectable
  - Cannot prefer one node's traffic over another's
  - Risk of favoring compromised nodes and starving correct sources traffic
- Ensures fairness and guarantees performance for flows even while under attack
- Requires cryptographic mechanisms for authentication and integrity

Large-scale Data Stores

- The promise:
  - Data stores can be built to scale horizontally (by adding more machines)
- Advantages:
  - Can run on hundreds of machines
  - Can scale up and down elastically as needs change
- Challenges:
  - Stability – the system needs to handle failures
  - Consistency – transactions can require coordination across many machines



Basic Communication Protocols

Lecture 1

Further reading: Data Networks,
D. Bertsekas and R. Gallager,
Prentice-Hall.

Internetworking with TCP/IP
Volume I
D. E. Comer

Automatic Repeat reQuest (ARQ) Protocols.

Causes for message omission:

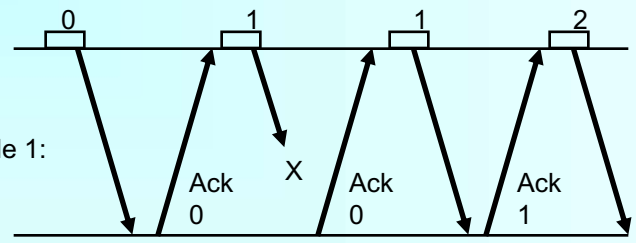
- Buffer spill.
- Error detection in a packet.

ARQ protocols:

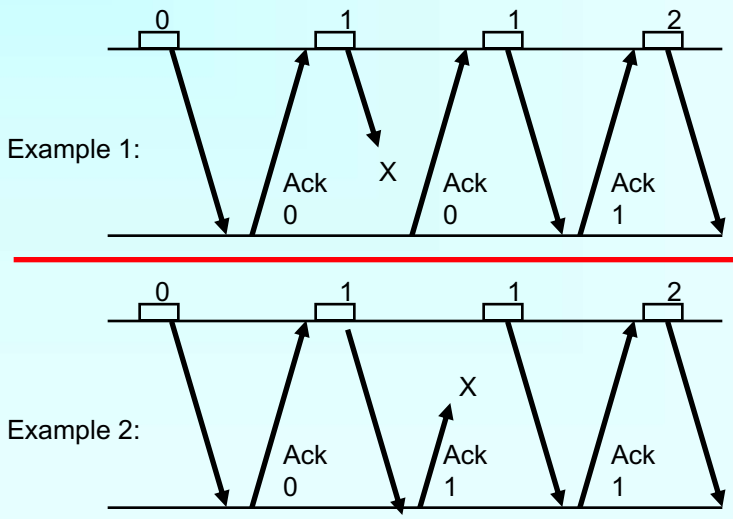
- Send & Wait.
- Arpanet.
- Go back *n*.
- Selective Repeat.

Send & Wait ARQ

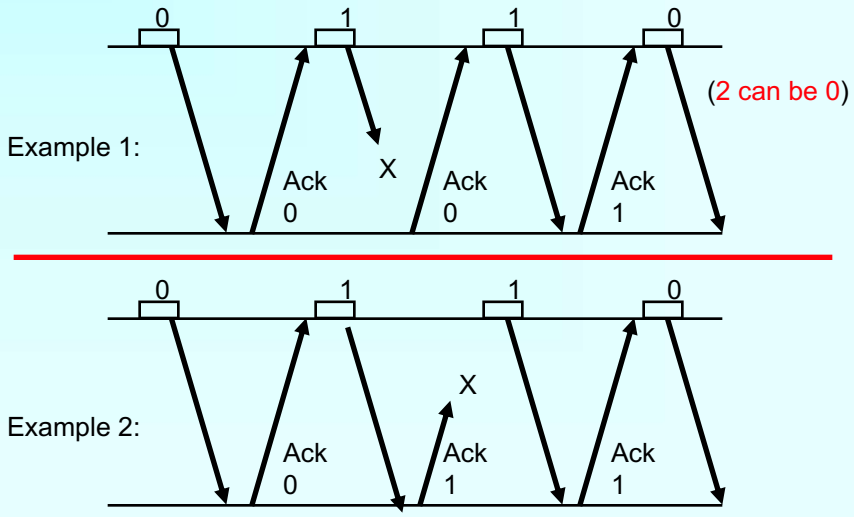
Example 1:



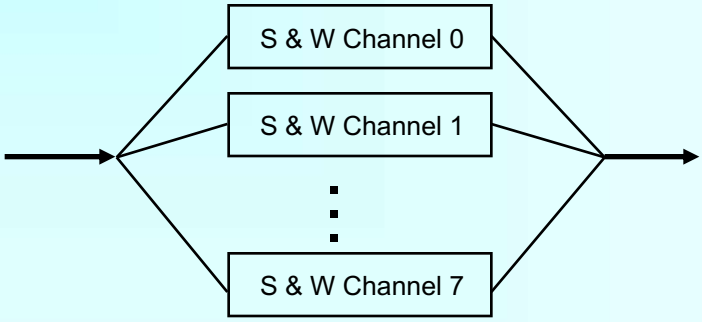
Send & Wait ARQ (cont.)



Send & Wait ARQ (cont.)



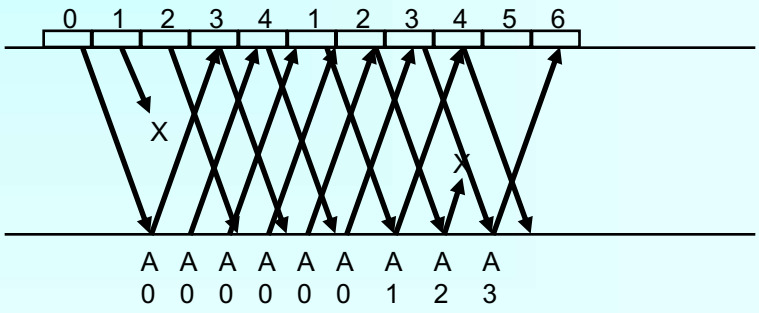
Arpanet ARQ



- Better line utilization than S & W.
- Unlimited memory required in theory.

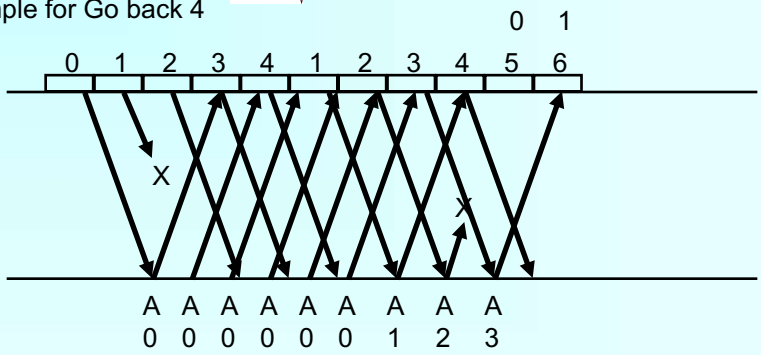
Go back *n* ARQ

Example for Go back 4



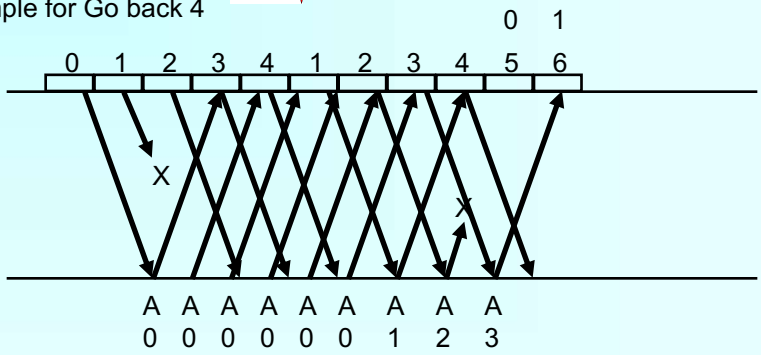
Go back *n* ARQ

Example for Go back 4



Go back *n* ARQ (cont.)

Example for Go back 4



- Good utilization
- limited memory required (one packet only).
- Full window is retransmitted in case of (one) error.

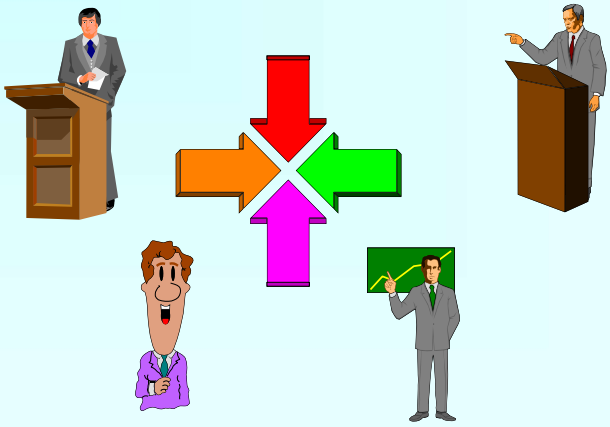
Selective Repeat ARQ

- Sliding window technique (as Go back *n*).
- Specifically indicating which packet is missing.
- Combines nacks and cumulative acks.
  - Acks acknowledge all messages with index of up to and including the ack value.
  - Nacks (negative acknowledgements) specifically request the messages with the indices in the nacks' values.
- Limited memory required (a full window).

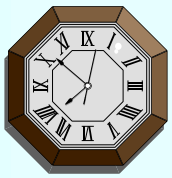
Question: What if there is no feedback?

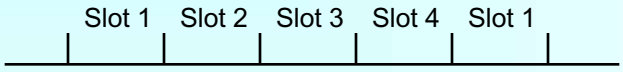
- A word about forward error correction (FEC), Internet loss patterns, etc.

Medium Access Control for Multi-access Communication.



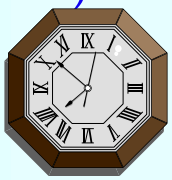
Time Division Multiplexing (TDM)





- The best utilization if everyone always has something to send.
- Wastes time if this is not the case.
- Slots can be unevenly assigned.

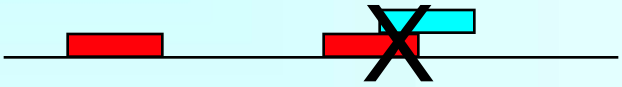
Slotted Aloha (Theoretical)



- Send at the next slot.
- If collision occurs - pick a random waiting time and send again at the next slot.

- Breaks.
- Maximal utilization is 0.36 (but much less for a desired behavior).

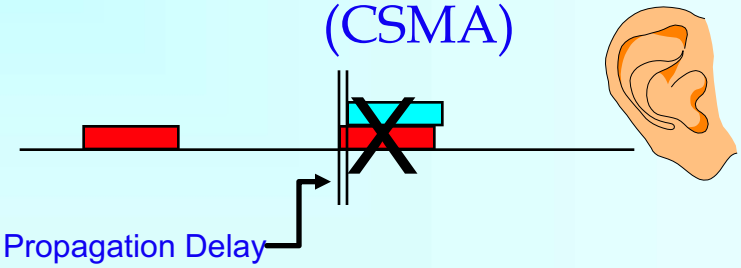
Aloha



- Send immediately.
- If collision occurs - pick a random waiting time and send again at that time.

- Breaks.
- Maximal utilization is 0.18 (but much less for a desired behavior).

Carrier Sense Multiple Access (CSMA)



- Listen to the line. Send if line is free.
- If collision occurs - pick a random waiting time and **try** again at that time.

CSMA/CD

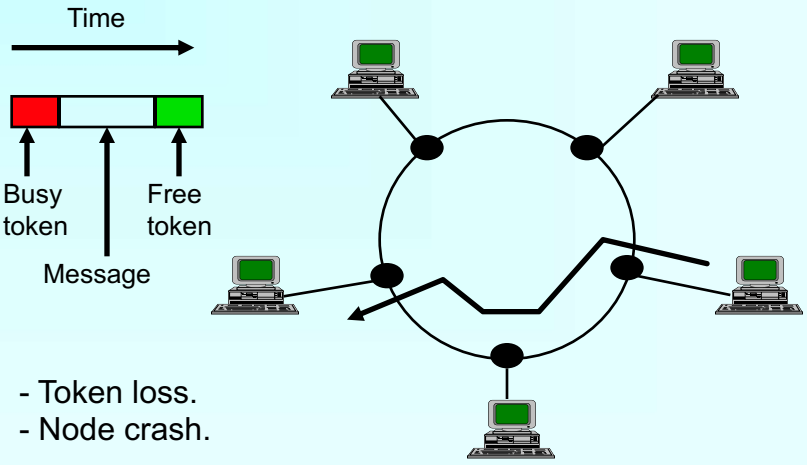
Carrier Sense, Multiple Access with Collision Detection.

Points to clarify:

- Propagation delay.
- X persistent CSMA
- Splitting algorithm for collisions

Ethernet = Persistent CSMA/CD with binary exponential backoff.

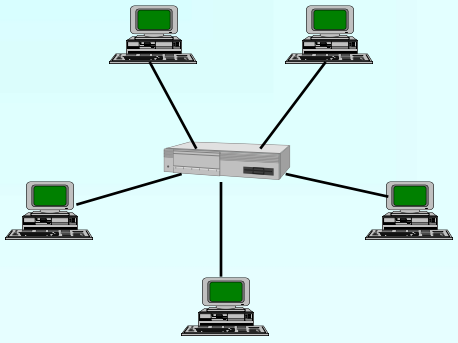
Token Ring



- Token loss.
- Node crash.

A Star Configuration

Can be used to mimic a bus configuration.
e.g. for Ethernet, Fast Ethernet, 1Gig Ethernet, 10Gig Ethernet, or Token Ring.



Routing

- Distance vector routing
- Link state routing
- Inter-network routing

Distance Vector Routing

- Each router knows the id of every other router in the network.
- Each router maintains a vector with an entry for every destination that contains:
  - The cost to reach the destination from this router.
  - The first link that is on that least-cost path.
- Each router periodically sends its vector to its direct neighbors.
- Upon receiving a vector, a router updates the local vector based on the direct link's cost and the received vector.

Link State Routing

- Each router knows the id of every other router in the network.
- Each router maintains a topology map of the whole network.
- Each router periodically floods its direct links state (with its direct connectivity information).
- Upon receiving a vector, a router updates the local topology map and re-calculates shortest paths.

Internet Routing

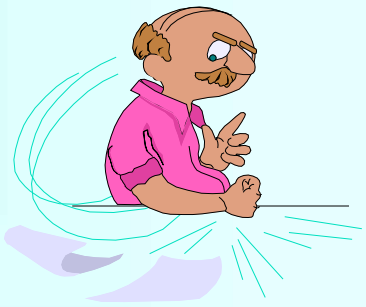
- Routing Information Protocol:
  - **Distance vector protocol.**
  - **Hop count metric**
  - **Exchange is done every 30 seconds, fault detection every 180 seconds.**
  - **Cheap and easy to implement, unstable in the presence of faults.**
- Open Shortest Path First:
  - **Link state protocol.**
  - **Internal hierarchy for better scaling.**
  - **Optimization for broadcast LANs with routers on them. (A designated router represents the whole LAN) - Saves control messages and size.**

Internet Routing (cont).

- A hierarchical routing protocol that connects networks, each of which runs an internal routing protocol.
- OSPF or RIP are common internal protocols.
- BGP - Border Gateway Protocol -
  - **A path vector protocol with additional policy information for each path. Path vector protocols have the complete path in each entry and not only the next direct member.**
  - **Generally used as the hierarchical routing protocol.**

Important Issues

- Flow Control.
- Stability.
- Management.
- Security.



Information Slide

- Code material is available on the CS undergrad lab machines (ugrad1-ugrad24) in the directory ~cs417/tutorials/
- A web page of reference material and programming documentation is available at http://www.dsn.jhu.edu/courses/cs417/ref.html
- Lecture slides can be obtained at the course web page http://www.dsn.jhu.edu/courses/
- Join our Slack space!!
  - you should have received an invite
- Make sure you have a Linux account for the ugrad lab (ugrad1-ugrad24)