Problem: There is more than one network (heterogeneity & scale)

Internetworking:
- Internet Protocol (IP)
- Routing and scalability
- Group Communication

Every seeming equality conceals a hierarchy.

--- Mason Cooley

Acknowledgement: this lecture is partially based on the slides of Dr. Larry Peterson
Process Groups

- Example uses
  - data dissemination (e.g., news)
  - replicated servers

- Group properties
  - Any set of processes that want to cooperate
  - Processes can join/leave either implicitly or explicitly
  - A process can belong to many groups

- Use multicast rather than point-to-point messages
  - group name (address) provides a useful level of indirection
Outline

- Multicast Routing

- A digression: replication of state machine
  - An application of multicast in improving systems dependability
Outline

- Multicast Routing

- A digression: replication of state machine
  - An application of multicast in improving systems dependability
Multicast Routing: Link State

- Each host on a LAN periodically announces the groups it belongs to using Internet Group Management Protocol (IGMP)

- Augment update message (LSP) to include set of groups that have members on a particular LAN

- Each router uses Dijkstra’s algorithm to compute shortest-path spanning tree for each source/group pair
Example of LS multicast routing

Example internet with members of group G in color
Example of shortest-path multicast trees
Scalability issue of L-S multicast routing

- (in addition to scalability issues of LS routing) Need to maintain the shortest-path routing tree for each source-group pair
  - Will consume too much memory

- Ameliorating approach: each router only caches trees for currently active source/group pairs
  - (-) With added computation cost when a group transits from “inactive” to “active” (this may well be affordable); similar to the caching issue in computer memory system
Multicast Routing: Distance Vector (D-V)

- **Reverse Path Broadcast (RPB)**
  - Each router already knows that its shortest path to source node S goes through a neighboring router, say N; then
  - When receive multicast packet from S, forward on all outgoing links (except one it arrived on), iff. packet arrived from N

- (-) a given packet will be forwarded over a LAN by each of the routers connected to that LAN
  - Solution: eliminate duplicate broadcast packets by letting only “parent” for LAN (relative to S) forward
    - shortest path to S (learn from distance vector): e.g., A
    - smallest address to break ties
D-V multicast (contd.)

- **Reverse Path Multicast (RPM)**
  - **Goal:** prune networks (from RPB tree) that have no hosts in group G
  - **Step 1:** determine if LAN is a *leaf* with no members in G
    - leaf if parent is the only router on the LAN
    - determine if any hosts are members of G using IGMP
  - **Step 2:** “propagate” “no members of G here” information up along the tree
    - augment (destination, cost) update sent to neighbors with set of groups for which this network is interested in receiving multicast packets
    - To avoid high memory overhead, only happens when multicast address becomes active (i.e., first use RPB, then prune unnecessary subtrees)
Protocol independent multicast (PIM)

- Deals with inefficiency of existing multicast routing protocols (especially D-V multicast) when groups only consist of a small percentage of routers
  - E.g., the (initial) broadcast in RPB (RPM)

- Two modes
  - Sparse mode: PIM-SM
  - Dense mode: PIM-DM (similar to RPM)
PIM-SM

- Each group is assigned a rendezvous point (RP)
  - Acts as the central relay between “source” and “group”

RP = Rendezvous point  
--- Shared tree  
------------ Source-specific tree for source R1

R4 sends Join to RP and joins shared tree
R5 sends Join to RP and joins shared tree: R2 does not forward Join to RP since it knows link (RP, R2) has been a part of the shared tree
Source R1 tunnels the multicast packet to RP, which forwards it along the shared tree to R4 and R5
PIM-SM: optimization (e.g., when there is a lot of data traffic to the group)

- Avoid overhead incurred by “tunneling from source to RP”
- Avoid the increased path length (or tree depth) due to transmission relay via RP
Note on PIM

- PIM is “protocol independent” in terms of “unicast routing protocol independent”
  - Unicast used in tree maintenance (e.g., delivery of “Join” message)

- It is pretty much bound with the Internet Protocol --- it is NOT protocol independent in terms of network-layer protocols
Outline

- Multicast Routing

- A digression: replication of state machine
  - An application of multicast in improving systems dependability
High availability via Replicated State Machine

- Service is characterized as a state machine that modifies variables in response to outside operations
- State machine is replicated to improve availability
- Key is ensuring
  - all operations are atomic (applied at all functioning replicas)
  - all replicas remain consistent (ops applied in same order)
- Implementation
  - encapsulate operations in messages
  - send using group communication
Atomic Messages

- Atomicity property: a message is delivered to all members, or to none

- First try...
  - each recipient acknowledges message
  - sender retransmits if ACK not received
  - problem: sender could crash before message is delivered everywhere
Atomic Messages (contd.)

- Fix: if sender crashes, a recipient volunteers to be “backup sender” for the message
  - re-sends message to everybody, waits for ACKs
  - use simple algorithm to choose volunteer
  - apply method again if backup fails

- Must remember all received messages in case we need to become backup sender
  - periodic protocol to “prune” old messages
  - how to know it’s safe to prune?
Message Ordering

- So far: different members may see messages in different orders

- Ordered group communication requires all members to agree about the order of messages

- Within group, assign global ordering to messages

- Hold back messages that arrive out-of-order
Ordering: First Approach

- *Central ordering server* assigns global sequence numbers

- Hosts apply to ordering server for numbers, or ordering server sends all messages itself

- Have to deal with case where ordering server fails
  - leader election we saw earlier

- Hold-back easy since sequence numbers are sequential
Ordering: Second Approach

- Use *time* when message was sent
  - measured on sending host
  - use host address to break ties

- Advantage
  - simple and decentralized

- Disadvantage
  - requires nearly synchronized clocks
  - must hold back messages for a period equal to maximum clock difference
Logical Time

- Insight: often don’t care about when something happened, only about which thing happened first

- Happened before relationship
  - $X < Y$ means “$X$ happened before $Y$”
  - three rules:
    - if $X$ and $Y$ occur in the same process and $X$ occurs before $Y$, then $X < Y$
    - if $M$ is a message, then $\text{send}(M) < \text{receive}(M)$
    - if $X < Y$ and $Y < Z$, then $X < Z$
Logical Time (contd.)

- Given two events $X$ and $Y$, either
  - $X < Y$, or
  - $Y < X$, or
  - neither ($X$ and $Y$ are concurrent)

- $<$ relation defines a partial order

- Example
Message Context to implement logical time

- Key: how to identify *partial order*?
- A process sends a message *in the context of* all the messages it has received.
- Group communication represented with a *context graph*.
- Example: 3 senders, denoted *a*, *b*, and *c*
Protocol

- Each node maintains a copy of the context graph
  - union of all copies equals “global graph”

- Send:
  - message-id (sender, seqno)
  - message-id of all predecessor messages
    - Only need to send *leaves* of sender’s copy of context graph
    - bounded by number of participants (why?)

- Receive:
  - add the partial context graph to local copy
  - deliver message to application
    - hold back if not all predecessors are present
    - ask sender to retransmit missing messages (why?)
    - pass up to application in “context” order
Protocol (contd.)

- Applications can inspect context graph
  - leaves, precedes, root, stable

- Message stability
  - A message is stable if it is followed by a message from all other participants

- System can free all stable messages from its copy
  - will never be asked to retransmit them
Host Failures

- How to guarantee
  - all running processes are able to continue exchanging messages
  - a message contained in any running host’s copy will eventually be incorporated into every running host’s copy

- Application support
  - mask out failed processes
  - adjusts message stability
Message Order

- Context graph preserves partial order among messages

- Each host can produce same total order by running a topological sort on context graph (with “tie-breaking” mechanism to order “concurrent packets”)
  - incremental since messages continually arriving

- Commit next “wave” of messages to application as soon as one message in wave becomes stable
  - know that no future messages will be at same logical time
Summary of Internetworking

- Internet Protocol (IP)
  - Best Effort Service Model
  - Global Addressing Scheme
  - Common IP format; datagram forwarding
  - Address translation (ARP)
  - Host configuration (DHCP)
  - Error reporting (ICMP)
  - Virtual private networks and IP tunnels

- Routing and algorithms
  - Algorithms: D-V, L-S, metrics, Mobile IP
  - Scalability: subnetting, supernetting (CIDR), BGP (P-V), IPv6

- Group communication
  - Multicast routing: L-S, D-V (RPB, RPM), PIM-SM
  - Atomic and ordered messaging
Discussion

- Routing in wireless networks (e.g., mesh networks, sensornets, MANETs, etc)
  - Link quality estimation, Routing metric

- Alec Woo, Terence Tong, and David Culler, Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks, ACM SenSys’03

- R. Draves, J. Padhye, and B. Zill, Routing in Multi-radio, Multi-hop Wireless Mesh Networks, ACM MobiCom’04

Discussion (contd.)

- Routing in mobile ad hoc networks
  - AODV, DSR, OLSR, etc.
  - IETF Manet working group:

- Routing in disruption(delay)-tolerant networks
  - Delay Tolerant Networking Research Group:
    http://www.dtnrg.org/wiki
  - Standards, papers ...: http://www.dtnrg.org/wiki/Docs
Discussion (contd.)

- Multicast routing
  - Fault tolerant distributed algorithms for minimum-spanning tree (instead of shortest-path spanning tree)
  - Harder especially for wireless and mobile networks where we have high degree of network dynamics

Further reading

- TCP/IP architecture (2004 Turing Award!)

- Scalability issue of IPv4, and IPv6

- Internet routing behavior
  - V. Paxson, End-to-end Routing Behavior in the Internet, ACM SIGCOMM’96
Further reading (contd.)

- Multicast routing
  - S. Deering and D. Cheriton, Multicast Routing in Datagram Internetworks and Extended LANs, ACM Transactions on Computer Systems, 8(2), May 1990

- IETF (Internet Engineering Task Force)
  - http://www.ietf.org
  - RFCs, Internet Drafts, and working group charters
Assignment – Chapter 3 & 4

- TinyLab#2 (mandatory)
  - Study the source code of Contiki Rime data collection protocol (see core\net\rime\collect.c, core\net\rime\collect-link-estimate.c, examples\rime\example-collect.c etc), and figure out how link estimation and distance-vector routing is implemented in real-world source code.
  - Change the code of Contiki data collection protocol to have two alternatives:
    - Collect-ETX: use link quality metric ETX as the routing metric (i.e., the default one in Contiki)
    - Collect-Hop: use hop-count as the routing metric
  - Measure and compare the packet delivery reliability of Collect-ETX and Collect-Hop in a multi-hop wireless network of 100 nodes uniform-randomly spreading around a space of 100 meters by 100 meters
  - References:
    - ETX metric: “Taming the underlying challenges of reliable multihop routing in sensor networks” (http://dl.acm.org/citation.cfm?id=958494)
    - CTP: “Collection Tree Protocol” (http://dl.acm.org/citation.cfm?id=1644040)

- Exercise#3
  - Chapter 3: Exercises 36, 46, 48, 54, 55, 68, 71, 72
  - Chapter 4: Exercise 17

- TinyExam#3