Learning Switches & Spanning Trees

CS 640, 2015-02-05

Announcements

- Assignment #1 due next Tuesday

Outline

- Learning Switches
- Spanning Trees
- Quiz

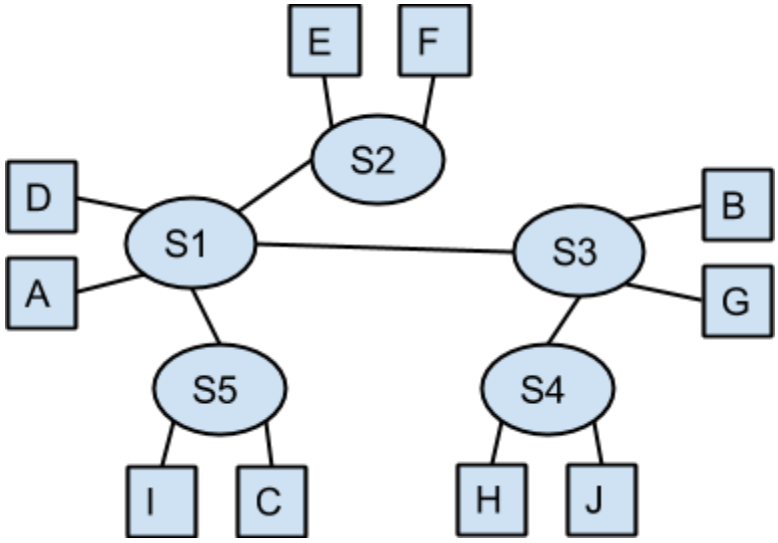
****What is the difference between a hub, a switch, and a router?**

- Hub
  - Naively forwards signal out all interfaces
- Switch (or bridge)
  - Forwards packets based on MAC addresses
  - Take a path to the destination -- not necessarily the shortest
  - Uses incoming packets to learn the interface to use to reach a specific destination
  - Broadcast packets if you do not know the path
- Router
  - Forwards packets based on IP addresses
  - Take the shortest (logical) path to the destination
  - Exchanges control packets with neighboring routers to learn the shortest path to specific destinations
  - Send along default path if you do not know the path

Link Layer Forwarding

- Switches forward Ethernet packets
  - Ethernet packet arrives on interface
  - Reads dst MAC address from Ethernet header
  - Looks up dst MAC address in forwarding table
  - Send out interface specified in forwarding table
  - If no matching entry, broadcast packet on all interfaces except the incoming interface
- Each forwarding table entry contains
  - MAC address
  - Interface
  - Timeout
- ****How do we determine the contents of the forwarding table?**
  - Statically specified
  - Endpoints broadcast advertisements
  - Switches exchange forwarding tables
  - Propagate based on incoming packets
- Challenge: hosts may move or go offline

- Learning the forwarding table
    - Ethernet packet arrives on interface
    - Reads src MAC address from Ethernet header
    - Inserts/updates entry in forwarding table with src MAC, incoming interface, and timeout
- ****Example**

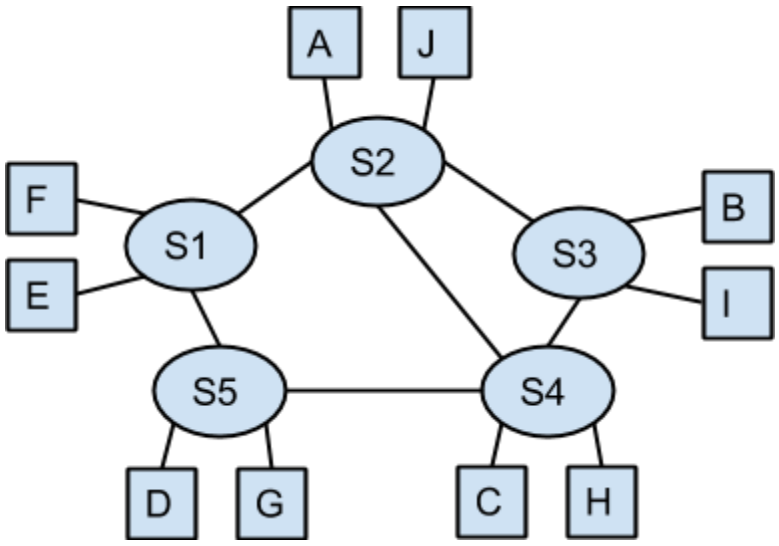


    - F → G
    - G → F
    - C → F
- ****What problem do we encounter with this approach?**
    - Loops can cause infinite broadcast ⇒ network crash
- ****How do we handle loops?**
    - Spanning tree

Spanning Tree

- "Disable" some links to remove loops
- Construct a spanning tree -- graph with exactly one path between any pair of nodes (switches)
- Basic idea
    - Select root -- switch with lowest identifier
    - Other switches find the shortest path to the root
    - Only ports on the paths are unblocked (i.e., packets are broadcast over these ports)
- Spanning tree algorithm
    - Switch broadcasts a message to all neighbors with:
        - Its identifier
        - Identifier for the switch it thinks is the root
        - Distance (number of hops/links) to reach the root
    - Initially, every switch thinks it is the root
    - When a switch receives a message, it checks if the message contains a "better solution"
        - A root with a lower identifier ⇒ receiver is not the root

            - Stop generating config msgs
            - Assume root is root id in msg
            - Forward received msg on all non-blocked ports, adding 1 to the distance
        - Same root, but lower distance ⇒ previous sender with same root is worse
            - Block port to previous sender
        - Same root and distance, but sender has a lower identifier ⇒ previous sender with same root and distance is worse
            - Block port to previous sender
        - Same root, sender has lower identifier, higher distance ⇒ sender is worse
            - Block path to sender
- ****Example -- What are the messages sent/received by S4?**



    - (S4, S4, 0) → S5, S2, S3
    - S5 → (S5, S5, 0)
        - S5 is higher root ⇒ Do nothing
    - S2 → (S2, S2, 0)
        - S2 is lower root ⇒ Stop generating msgs; S2 is root; (S4, S2, 1) → S5, S3
    - S3 → (S3, S3, 0)
        - S3 is higher root ⇒ Do nothing

    - S5 → (S5, S1, 1)
        - S1 is lower root ⇒ S1 is root; (S4, S1, 2) → S2, S3
    - S2 → (S2, S1, 1)
        - S1 is same root, distance is same ⇒ via S5 is worse; block port to S5; (S4, S1, 2) → S3
    - S3 → (S3, S2, 1)
        - S2 is higher root ⇒ Do nothing

    - S3 → (S3, S1, 2)
        - S1 is same root id, but distance is longer and S3 is lower than S4 ⇒ Block port
- Simple algorithm
    - Lowest switch is root

    - Prefer switch closer to root; if there is a tie, prefer switch with lower id
- ****Example -- What ports are blocked?**