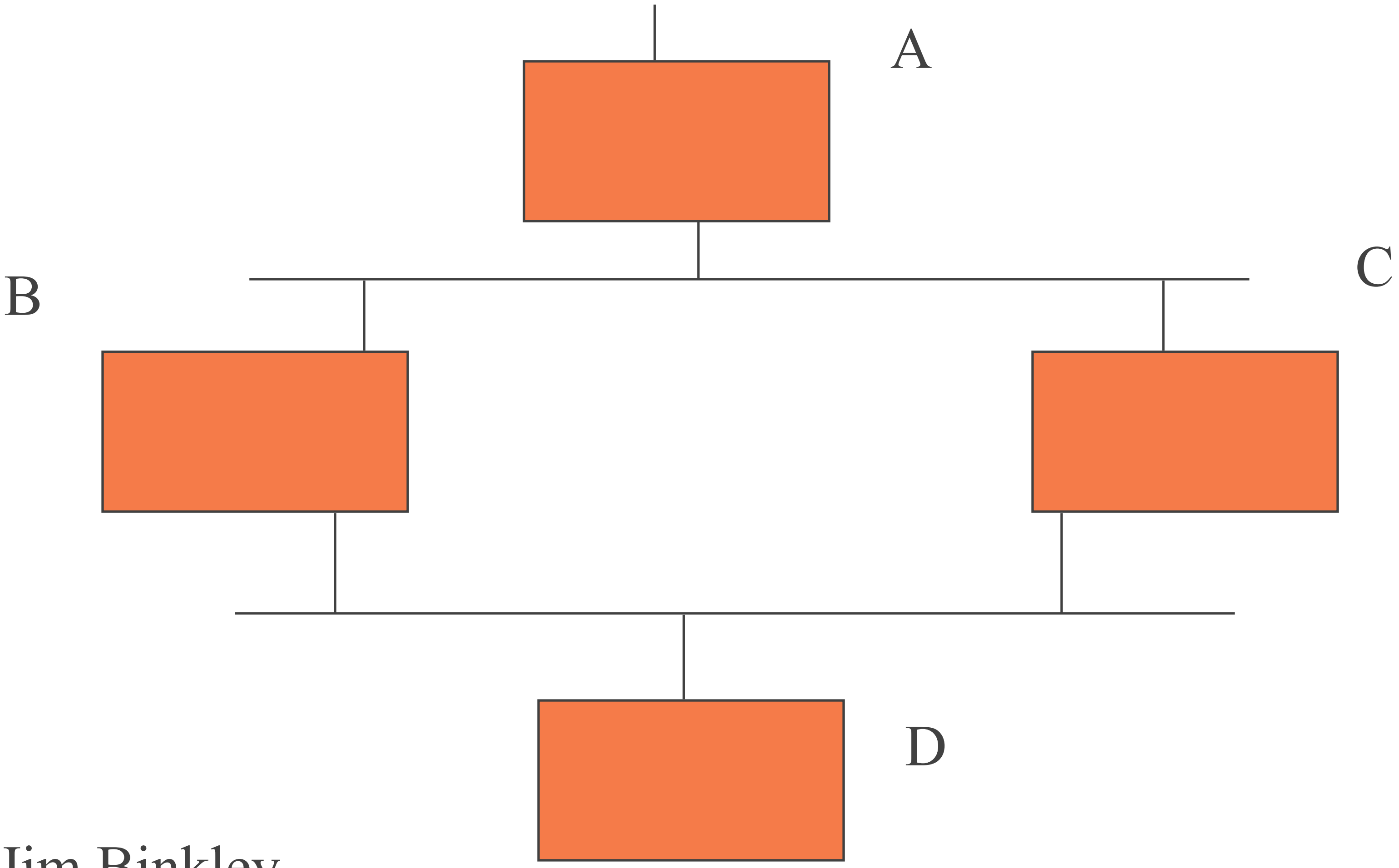
802.1d – spanning tree

- see Stallings, Local and Metropolitan Area Networks, for more info
- IEEE 802 standard (802.1D)
- bridge protocol at link layer
- bridges form rooted spanning tree, **no cycles**
  - **aka no loops**
- ports ultimately in {forwarding, blocked} state
  - on or off
- done with simple L2 flooding protocol

4 bridges, what happens?



Jim Binkley

operation

- each bridge has ID based on 1 mac address
- each port has MAC address (port ID)
- root bridge is top of tree
- root chosen by Spanning Tree Algorithm
- (low) path cost may be associated with bridges by manager in order to influence choice
- may also set **PRIORITY** to influence root

more operation

- **designated bridge/port**, bridge on LAN that is chosen to forward packets to/from another lan
- **root port** - each bridge discovers 1st hop on minimum-cost path to root bridge. if two ports on a LAN, then use lower port number.

basic idea:

- 1. determine root bridge
- 2. determine root port on other bridges
- 3. determine designated port on each LAN
- consequence: if two bridges connect same two LANs, one is left out
- timers used so that if designated port fails, another may be chosen; i.e.,
- **at boot, or at change, STA recalculated**

BPDU/s

- BPDU - bridge protocol data unit
- sent out on all ports to ALL BRIDGES multicast group address
- in general, BPDU from one bridge flooded out the other ports, and used in both
- send whilst maintaining (periodic resend) or
- rooted tree STA recalculation

BPDU cont.

- 2 packet types, config, topological (start over)
- configuration BPDU is 35 bytes, root resends at hello time interval, hello time default is 2 seconds (root sends)
- root id field in BPDU (5 bytes in), 8 bytes
  - 2 bytes of root priority, 6 bytes of MAC
- config sent during STA, stable state, election time
- topo packet only 4 bytes
- topological change sent when bridge believes configuration change occurred, therefore redo STA
- stable state: root issues configuration/everybody else forwards it

BPDU encapsulation

dst	src	DestSAP	SendSAP |	BPDU part

dst - group multicast address
src - unique per port
SAP/s - 01000010 (palindrome)

# tcpdump -e -n -i <ifname>

```
11:32:41.457906 0:a0:c9:47:cb:21 > 1:80:c2:0:0:0 802.1d
          ui/C len=43
          0000 0000 0080 0000 a0c9 47cb 2000 0000
           0080 0000 a0c9 47cb 2080 0200 0014 0002
           000f 0000 0000 0000 0000 00
```

note: mac dst is 1:80:c2:0.0.0 - ALL-BRIDGES mcast
note 0:a0:c9:47:cb:20 in data portion, part of root ID

better:
tcpdump –vvv –e –i xl1

- src mac 1:80:c2:0:0:0 0026 64: 802.1d config 8000.00:d0:58:3a:9b:42.8019 root 8000.00:d0:58:3a:9b:42 pathcost 0 age 0 max 20 hello 2 fdelay 15

port state machine

- **listening** - STA algorithm used, but bridge does not learn, on timer elapse can become
- **learning** - in addition, bridge can learn, timer elapse can become
- **forwarding** - bridge port root/designated
- **blocking** - bridge learns that this port is not part of ST, therefore blocks port
  - any change puts in listening state
- listening/learning/forwarding on timer elapse done to prevent loops - downside is can be slow

STA operation

- everybody assumes root to start with
- flooding clues them in to who actually has the lowest root ID
- root announces I AM ROOT
- directly connected bridges, send BPDU to say one hop away out other port
- closest bridge becomes path
  - if more than one, smaller bridge MAC wins

election algorithm (cost, priority, MAC):

- if 2 paths to root, we choose the one with the “lowest cost”
- path cost first (choose faster link) then
- choose between priority+MAC “id”
- smallest value wins for that 3 tuple
- is this a good idea or a bad idea in terms
  - of root selection?
- remember Murphy is watching ...

spanning tree algorithm - summary

- 50 seconds or so to resettle network possible upon failure (default is 30, reality can take longer)
- you can “feel it” (net is down)
- pro: **redundancy**, and somewhat idiot-proof
  - function is anti L2 loop after all
- con: ports not in use, downtime is con too
- may wish to use root **priority** to decide who is ROOT, but usually not tweaked
  - set priority LOWER to win
  - make sure implementation supports redundancy

Spanning tree design thoughts

- KISS design - keep from you making a loop and taking a net down
  - do not turn it off unless you know what you are doing (typically on by default)
  - consider wiring morass, especially inter-building on campus
- LARGE scale (e.g., campus-wide) tree probably a BAD idea
  - flaky switch on DMZ could cause 50-sec. outages?
- SPT 1-1 with VLAN, IP subnet (bcast domain)

it’s not dead yet, Jim: but

- IEEE 802.1w – Rapid Spanning Tree protocol has been introduced
- goal is to NOT WAIT 60 seconds to reform spanning tree
- not a good thing in an exchange for example
- ironically: OSPF may converge faster than 802.1d

basic idea:

- decouple port state (blocked, forwarding) from role (root, designated port)
- 3 states in RSTP:
  - learning, forwarding, discarding
- 4 port roles in RSTP:
  - root port, designated port, alternate, backup
  - root port – port closest to root bridge
  - designated port – port not root port, that is best port for forwarding pkts (downstream port)

more

- port roles
  - alternate – blocked better better BPDUs come from some other bridge
  - backup – port blocked by better BPDUs from same bridge it is on
  - alternate + blocking more or less == old blocked

BPDU format overview

- type 2, version 2 – therefore older switches will ignore it
- every BPDU issued has port role and state marked in flags
  - therefore recv. can figure out what to do
- BPDUs are sent per port
  - not “flooded” from root anymore
  - must reflect sender’s state

BPDU protocol changes

- BPDU is now hello
  - must hear from neighbor with 6 seconds
  - 3 retries at 2 times per sec.
  - else begin election
  - can be sure problem between you and neighbor
    - not somewhere between you and root
  - fault is now local, not global
  - this allows faster aging to occur

BPDU protocol changes

- accepting inferior (less good path) information
  - if we hear less good news from the root
  - we believe it immediately
  - e.g., B talks to root and C
  - B loses root, tells C B is root
  - C tells B, nope … I have path to root
  - B believes C

BPDU protocol changes

- fast transition to forwarding state
  - don’t need to wait for slow timers due to port info and bridge feedback about convergence
  - 2 new variables: 1. edge ports, 2. link type
  - edge port: if port is connected to workstation, it cannot create a bridging loop
    - » if link toggles does not generate topo change
  - link type: if edge port or full-duplex can make rapid transition, otherwise cannot

feedback mechanism

- an inferior bridge can tell superior to start forwarding
  - and it blocks downstream ports to prevent a loop
- this recursively works to create a loop-free tree
- and make convergence much faster

new topo change mechanism

- in 802.1d when topo change is detected
  - any non-root bridge notifies in direction of root bridge
  - root advertises TC for max-age+forward delay
- in RSTP
  - TC sent by forwarding state change, not edge port
  - very different from 802.1d

topo change in RSTP

- if bridge detects TC
  - 1. starts TC while timer for 2*hello time on non-edge designated and root ports
    - » BPDUs have TC bit set
  - 2. flushes mac addresses associated with those ports
- so any bridge can do this, not just root
- takes a few seconds
- clears MAC forwarding tables (VLAN CAM tables in Cisco speak)