CMSC417 Spring 2016 Lecture #5 2/10/2016

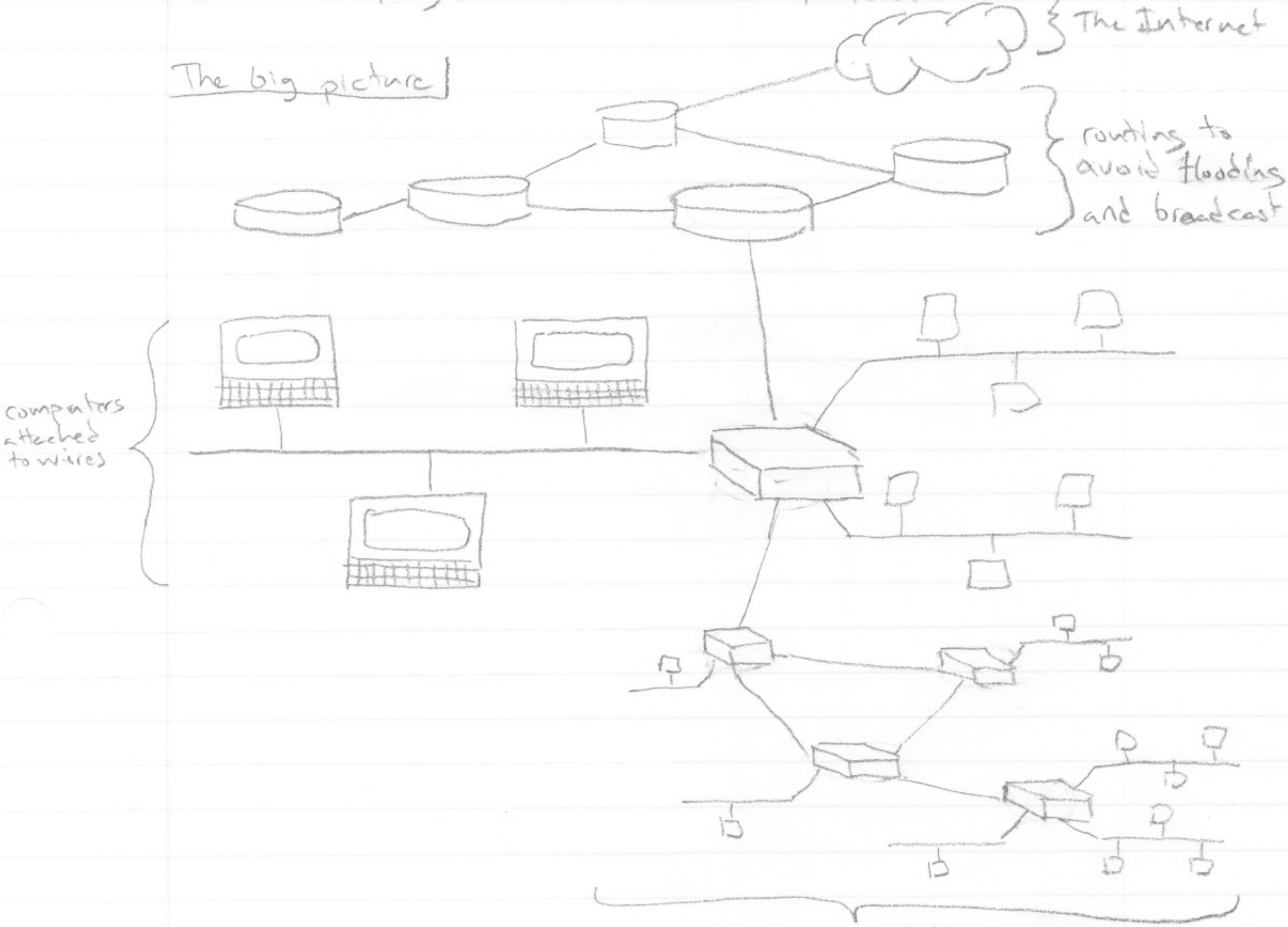
Agenda

⇒ quick review of spaning tree
- scope: small, local area networks
- limited bandwidth, still falls back to broadcast
- broadcast lets you not know where people are
- some exact details of STP?

⇒ routing (vs. forwarding)
- distance vector routing
- link state routing
- convergence

CMSC417 Spring 2016 Lecture #5 2/10/2016

The big picture



repeaters, switches, bridges, learning bridges and spanning tree protocol scale the wire abstraction to 100s-1000s of computers and 10s-100s of switches

⇒ today we'll start with routing, after that we'll move on to the global Internet, IP, BGP, etc.

CMSC417 Spring 2016 Lecture #5 2/10/2016

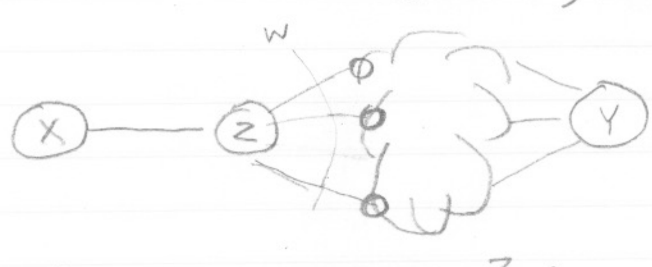
Routing

- ⇒ Until now, we've talked about forwarding — taking a frame/packet and using local state to send it out a given port/interface
- ⇒ the assumption has been no help is needed for a given device to build it's state
  - can learn all locations of addresses by snooping
  - if you're missing state, just flood
- ⇒ routing is the process of populating that state in the absence of cheap flooding / broadcast

Distance Vector

- ⇒ all routers track all known other routers and the distance and next hop-router to reach them
- ⇒ start with distance 0 to themselves

$D^x(y, z)$ = distance to reach y through z from x, where z is a direct neighbor of x



$$D^x(y, z) = \min_w D^z(y, w)$$

CMSC417 Spring 2016 Lecture #5 2/10/2016

Distance Vector cont'd

⇒ Start with $D^x(x,x) = 0$
$D^x(y,z) = \infty \quad \forall\, y, z$

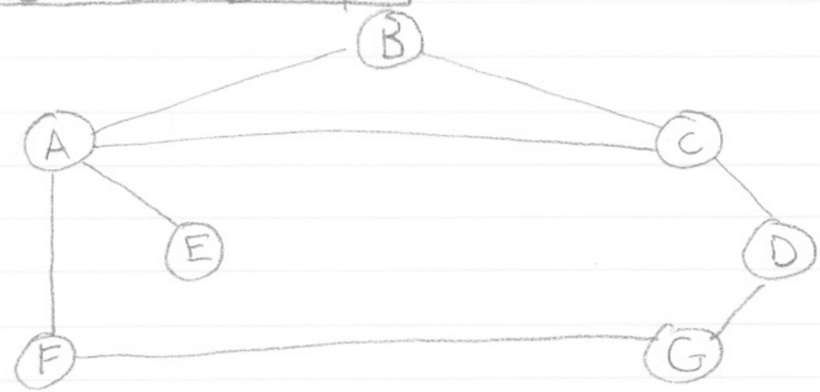
⇒ Send your information to all neighbors
$\forall y$ send "I can reach y with dist $\min_z D^x(y,z)$"

- send periodically on a timer (seconds - minutes)
- send immediately on a change in your minimum cost to reach a dest
- send immediately on topology changes
  - link goes down
  - neighbor becomes unreachable (no updates from them for a while)
  - link coming up and new neighbors are handled on reciept of a routing message resulting in an update to min costs to reach destinations
  - link changing cost is also handled by minimum distance updates

⇒ on receiving an update from z at x
for each $D^z(y,*)$
$D^x(y,z) = D^z(y,*) + \text{link\_cost}(x,z)$
if the minimum cost to reach any dest changed, send a new routing message to all neighbors

CMSC417 Spring 2016 Lecture #5 2/10/2016

Distance Vector Example)



assume all links have unit cost

At A (initially)

dest	next hop	distance
A	A	0
B	B	1
C	C	1
D	—	∞
E	E	1
F	F	1
G	—	∞

after updates from F, C

dest	next hop	distance
A	A	0
B	B	1
C	C	1
D	C	2
E	E	1
F	F	1
G	F	2

F ⇒ G link fails

⇒ at F, $D^F(G,G) \leftarrow \infty \Rightarrow D^F(G,*) \leftarrow \infty$

⇒ send update to A, $D^A(G,F) \leftarrow \infty \Rightarrow D^A(G,*) \leftarrow 3$ via C→D→G

depends if A is keeping the full routing table, e.g., $D^A(y,z)\ \forall\ y,z$ or just $\min_z D^A(y,z)$ and z minimizing it as shown above.

full routing table takes more space, but allows for faster convergence

CMSC 417 Spring 2016 Lecture #5 2/10/2016

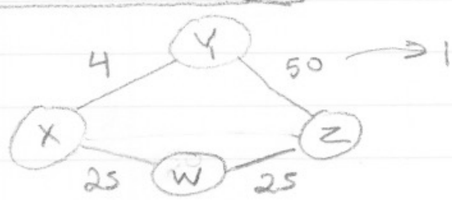
Count to infinity

⇒ Link A → E fails (see prev page for network)
⇒ before recieiving A's update, C sends that
$D^C(E, *) = 2$
⇒ A decides $D^A(E, C) = 3$, sends update
⇒ C decides $D^C(E, A) = 4$, sends update

keeps going adding 1 to
cost until cost hits ∞

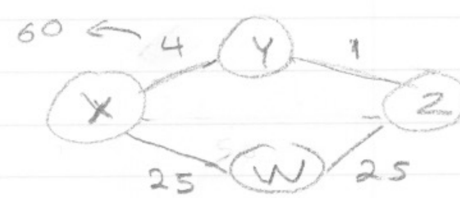
"count to infinity" problem

good news travels fast



⇒ link_cost(Y, Z) drops to 1
⇒ Y tells X
⇒ X updates
done in 2 messages

bad news travels slowly



⇒ link_cost(X, Y) rises to 60
⇒ X tells W, but W thinks it
has a shorter route via Z (cost 30)
⇒ Y tells Z, but Z thinks it has a better route
through W
⇒ counts up until somebody sets to 61

CMSC 417 Spring 2016 Lecture #5 2/10/2016

Split horizon

⇒ don't tell people about your paths that use them as them as the next hop

⇒ in the example with the A-E link failing this solves the problem where C tells A it can get to E

⇒ still fails if C tells B who tells A causing a count to ∞

Poison Reverse

⇒ advertise ∞ back to people when you use them as the next hop for that destination

Neither fixes all problems

⇒ loops >2 nodes have ways to count to ∞ even with split-horizon w/poison rev

Other Solutions

⇒ use a small version of ∞, e.g., 16

⇒ wait for a timer to expire so that

Emergent properties of Distance Vector

- ⇒ good news travels fast
- ⇒ bad news travels slowly
- ⇒ count to ∞
- ⇒ split horizon (w/ poison reverse)
- ⇒ timers to converge before re-announcing