ICS 351: Today's plan

- dynamic routing
- RIP
- distance-vector routing

dynamic routing

- in lab 3, routes are set by hand, which is static routing
- when using proxy arp, the router automatically decided to ARP for another network to which it had a route (as long as proxy arp was enabled)
- it is also possible to build routing tables automatically: **dynamic routing**
- this is possible because, once routers are configured, each router knows to which networks it is connected
- if each router distributes this information to all other routers, every router can build a working routing table

dynamic routing example

- router 1 connected to 10.0.1/24 and 10.0.2/24
- router 2 connected to 10.0.2/24 and 10.0.3/24
- router 1 could tell router 2 how to reach 10.0.2/24, and router 2 could tell router 1 how to reach 10.0.3/24
- this is all automatic
- for any correct routing protocol, the only questions are:
  - how long does the network take to reach a consistent state again (to converge) after a change? (e.g. 30s, 10min)
  - how much network traffic does the routing protocol add? (e.g. 10Mb/s/router)

RIPv2

- Routing Information Protocol, version 2 (v2 supports network masks)
- RIP generally used within a single Autonomous System (AS), i.e. within an organization
- this makes it an IGP, Interior Gateway Protocol
- defined in RFC 2453, http://tools.ietf.org/html/rfc2453

RIPv2 details

- reliably finds shortest paths in networks that don't change very often
- links may have a "metric" (cost, distance) associated with them, or "1" may be used for each link, but in any case metrics should be static
- limited to networks with a maximum metric of 15 between any two nodes
- router must be configured to know which interfaces to run RIP on

distance-vector routing

- Bellman-Ford, applying Bellman's equation in the Ford and Fulkerson algorithm
- the basic idea is that each router sends its routing table to all directly-connected routers, that is, to its neighbors
- when receiving such a message (routing update), a router must update its own routing table:

DV routing updates

- when receiving a routing update, a router must update its own routing table:
  - any new route is simply added, with the next hop (gateway) being the router that sent the routing update, and the metric being the metric received plus the metric of the link over which it was received
  - any existing route is replaced if the new route has better metric (after adding the link metric)
  - any existing route is replaced if the routing update comes from the router listed as next hop

DV route deletion

- routes through a given next hop G are *timed out* if no routing updates are received from G within a certain period (about 6 times longer than the 30s routing update time)
- if a route is deleted, it is actually marked as having an infinite cost (for RIP, infinity is 16)
- also, when sending a routing update to neighbor G, routes through G are "poisoned” by giving them infinite metric (16)
- This is **split horizon with poisoned reverse**

RIPv2 details

- routing tables sent to multicast address
- split horizon (do not send G routes for which G is the next hop) is required
  - poisoned reverse (send but with infinite cost) is optional
- whenever the routing table changes
  - e.g. due to an interface change, or a routing update

  a triggered update is sent out, perhaps with only the new route(s)
- triggered updates should be limited to about 1 every 5 seconds
- regular updates are sent every 30 seconds

counting to infinity

- suppose that a router R loses one of its connections (to network N with metric m), but does not send a triggered update, and does not do split horizon
- then, its neighbor G will tell it about a route to N with metric m+2
- R will use a route to N with metric m+3, next hop G
- at the next update, G will know R has a route to N with metric m+3, so will have a route with metric m+4, next hop R
- then R updates its route to N to have metric m+5
- then G updates its route to N to have metric m+6
- and so on
- this only stops when one of the routers reaches 16 (infinity)
- it could potentially take a long time to delete bad routes