ICS 351: Today's plan

- ARP protocol
- `arp` command
- proxy ARP
- internet packet forwarding
- IP routing table and routing cache
IPv6 addresses

- IPv6 addresses (128 bits) are also split into network and host parts
- this split is not determined by a netmask, but by the first few bits of the address
- an address beginning with 80 zero bits followed by 16 1 bits is an "IPv4-mapped IPv6 address", is valid as long as the interface's IPv4 address is used as the last 32 bits (80+16+32 = 128)
- 0:0:0:0:0:ffff:abcd:ef01 is 171.205.239.1
Class based addressing

• using the first few bits to determine the class of the address was first developed for IPv4
  • but abandoned as inefficient when IPv4 addresses started to become scarce
• there are many IPv6 addresses!
IPv6 addresses: netmasks not req'd

- IPv6 addresses (128 bits) are split into network and host parts
- "normal" IPv6 addresses have a global routing prefix (essentially a network number) and a subnet ID in the first 64 bits, followed by 64 bits of interface ID
- such an interface ID can, for example, be taken from the globally unique 48-bit Ethernet (MAC) address associated with the Ethernet/WiFi hardware
- RFC 4291 (http://tools.ietf.org/html/rfc4291) has many more details

- the routing table for IPv6 still has netmasks
ARP protocol

- when communicating over the local network, the routing table only records the IP address of the next (or final) interface
- one advantage of this is that the next hop (e.g. a router) can be replaced relatively easily
- an ARP request (ARP who-has) is broadcast whenever the MAC address is needed for a local IP address
- the ARP reply is unicast back to the sender, and carries both IP and both Ethernet addresses
ARP details

- the ARP reply could be sent by an ARP proxy if the intended destination does not support ARP.
- or by an attacker!
- ARP packets are not IP packets (ping packets are IP packets)

arp command

• always use "-n" in the lab for all these commands, to request numerical output rather than domain name resolution

• **arp -a -n**: print all the entries in the ARP table

• **arp -d address**: remove the table entry corresponding to the given IP address

• **arp -s address MAC**: add a table entry mapping the given IP address to the given MAC address (use temp at the end of the command to install a normal temporary translation)
Proxy ARP

- given a router connected to two networks
- e.g. 192.168.10.0/24 and 192.168.11.0/24
- the router can forward everything between the two networks
  - to get the packets, the router must reply to ARP requests for the "other side"
  - hosts in the two networks use a 23-bit netmask, so they believe they are in the same network and try to send directly
- this generalizes to more than two networks
- but only works for networks directly connected by a single router
- in-class exercise: which MAC address does the router give in its ARP replies?
Internet packet forwarding

- the IP protocol tells us how to forward packets on the Internet
- forwarding needs a gateway/next hop reachable through a given interface
- different steps:
  1. check routing cache for an exact match. If found, use corresponding gateway and interface
  2. otherwise, find all the routing table entries for which the network part of the address matches the IP destination address
  3. if found, use the match *with the most bits in its network part* (the longest match)
  4. if there is no matching route, the packet is discarded
Internet packet forwarding, details

- since the netmask is stored in the routing table, the same IP destination address will (typically) match routing table entries with different masks
- the default route has a mask of 0.0.0.0, so is the shortest match – not used if any other route matches
- a router receiving a packet for destination D may issue an ICMP redirect that says "for destination D use router R instead of me"
  - for example, if a packet is sent to the default router, but the router R directly attached to the destination D is connected on the same network as the sender of the packet
- ICMP redirects only affect the routing cache
IP routing table metric

- it is fine for a routing table to have more than one (equal-length) route for a given destination
- each routing table entry has a metric
  - can be thought of as “cost”, or “distance”
  - less metric is better!
- if they have different metrics, the one with the smallest metric is used
- if they have the same metric, any one can be used, possibly even in round-robin fashion
IP routing table issues

- the routing tables of different routers/hosts should be such as to deliver packets to their destination in the least number of hops.
- when routing tables of different routers/hosts can deliver every packet to its destination, they are consistent
- inconsistent routing tables will lead a packet back to a router which has already seen it: this is a routing loop
- an IP packet will go around the loop a few times
- no IP packet can be forwarded more times than its maximum number of hops, or time-to-live/TTL, or hop limit in IPv6
- the time to live/hop limit in the IP header is decremented each time the packet is forwarded
Static routing

- routing tables can be built by hand
- this works well when:
  - the routes are not changed very often, and
  - the network is small
- whenever equipment is configured manually, it is possible that there will be an error, e.g. a routing loop
- tools such as traceroute (tracert on Windows) can be used to debug this
Lab 3

- errata: p. 113, "clear arp-cache" instead of "clear arp"
- class uses minicom instead of kermit
- using a Linux PC as a router (a Windows PC or MAC works just as well)
- using a Cisco router
- verifying network routing setup
- ICMP route redirect
- routing loops
- network prefixes