Packet Forwarding
Reading: Chapter 4

Hop-by-Hop Packet Forwarding

- Each router has a forwarding table
  - Maps destination addresses…
  - … to outgoing interfaces
- Upon receiving a packet
  - Inspect the destination IP address in the header
  - Index into the table
  - Determine the outgoing interface
  - Forward the packet out that interface
- Then, the next router in the path repeats
  - And the packet travels along the path to the destination

Datagram Forwarding - Algorithm

- Extract destination IP address, Id from datagram
- Compute IP address of destination network In
- If In matches any directly connected network, send datagram to that network
- Else If Id appears as a host-specific route, route datagram as specified in table;
- Else If In appears in routing table, route datagram as specified in the table;
- Else if a default route available, route datagram to default gateway;
- Else, routing error.

Datagram Forwarding

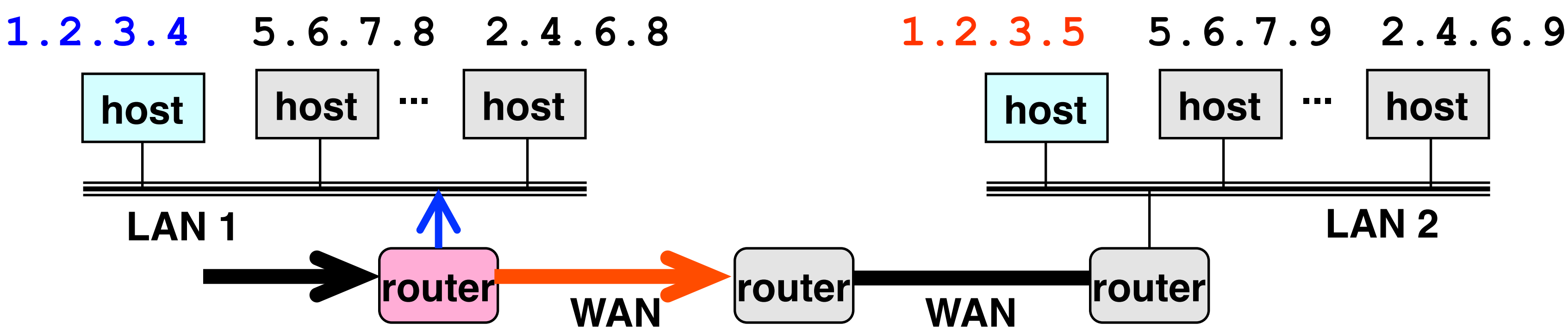
- Strategy
  - every datagram contains destination's address
  - if connected to destination network, then forward to host
  - if not directly connected, then forward to some router
  - forwarding table maps network number into next hop
  - each host has a default router
  - each router maintains a forwarding table

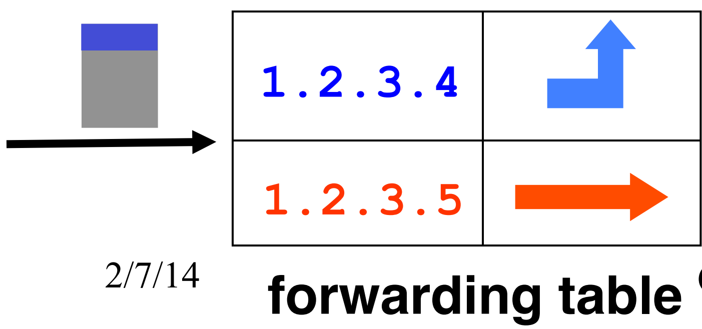
- Example (R2)

Network Number	Next Hop
1	R3
2	R1
3	interface 1
4	interface 0

Separate Table Entries Per Address

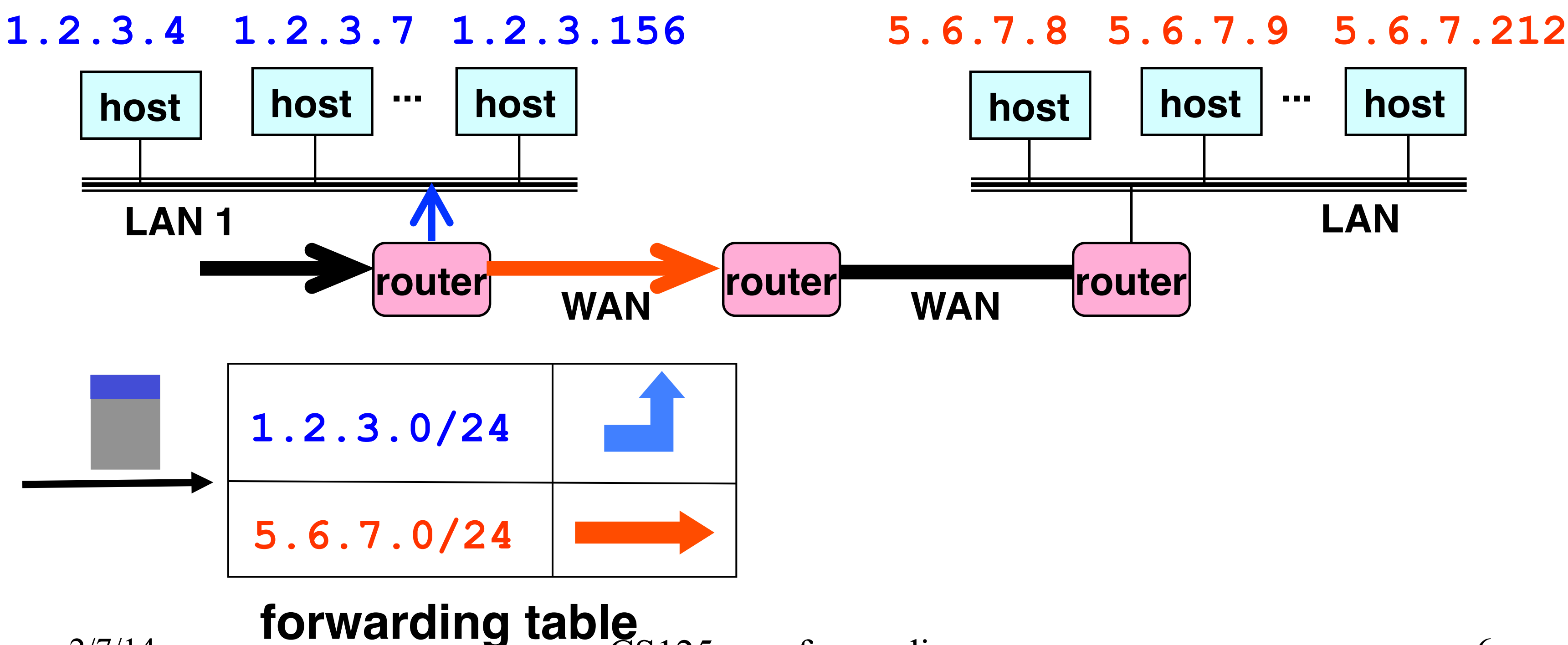
- If a router had a forwarding entry per IP address
  - Match *destination address* of incoming packet
  - … to the *forwarding-table entry*
  - … to determine the *outgoing interface*





Separate Entry Per 24-bit Prefix

- If the router had an entry per 24-bit prefix
  - Look only at the top 24 bits of the destination address
  - Index into the table to determine the next-hop interface

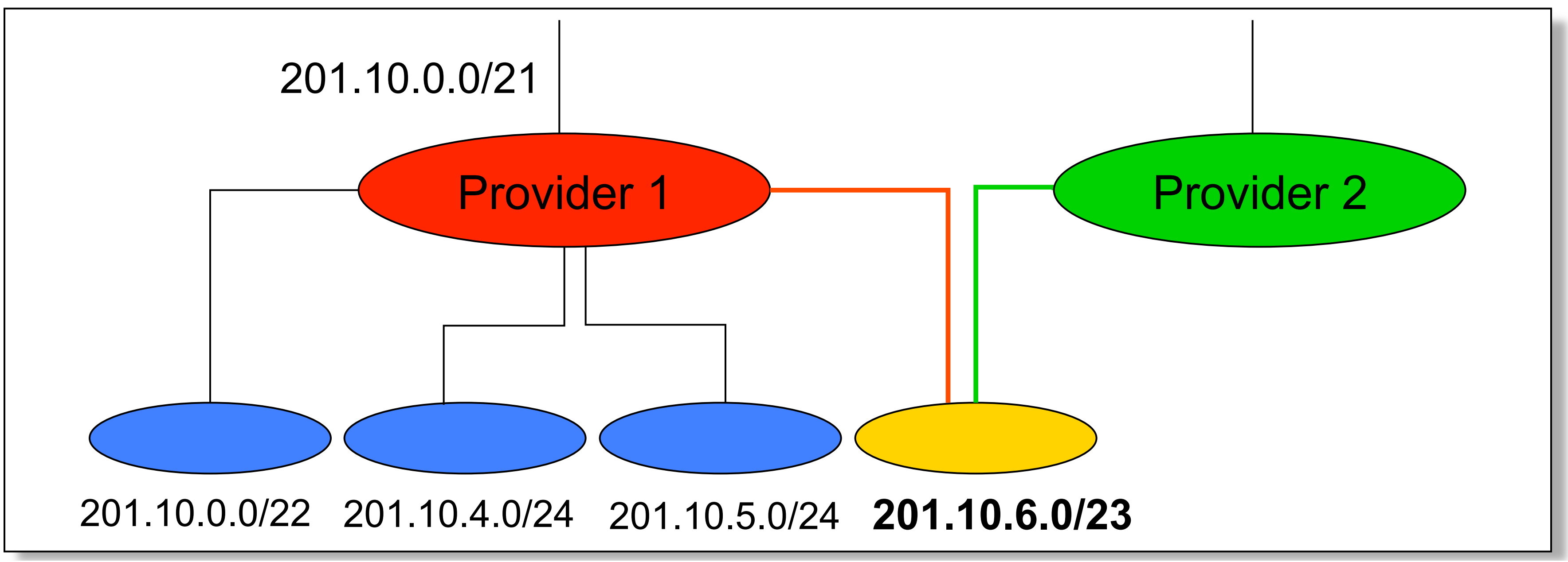


Separate Entry Classful Address

- If the router had an entry per classful prefix
  - Mixture of Class A, B, and C addresses
  - Depends on the first couple of bits of the destination
- Identify the mask automatically from the address
  - First bit of 0: class A address (/8)
  - First two bits of 10: class B address (/16)
  - First three bits of 110: class C address (/24)
- Then, look in the forwarding table for the match
  - E.g., 1.2.3.4 maps to 1.2.3.0/24
  - Then, look up the entry for 1.2.3.0/24
  - … to identify the outgoing interface

CIDR Makes Packet Forwarding Harder

- There's no such thing as a free lunch
  - CIDR allows efficient use of the limited address space
  - But, CIDR makes packet forwarding much harder
- Forwarding table may have many matches
  - E.g., table entries for 201.10.0.0/21 and 201.10.6.0/23



Longest Prefix Match Forwarding

- Forwarding tables in IP routers
  - Maps each IP prefix to next-hop link(s)
- Destination-based forwarding
  - Packet has a destination address
  - Router identifies longest-matching prefix
  - Cute algorithmic problem: very fast lookups

forwarding table

destination

201.10.6.17 →

4.0.0.0/8
4.83.128.0/17
201.10.0.0/21
201.10.6.0/23 →
126.255.103.0/24

outgoing link

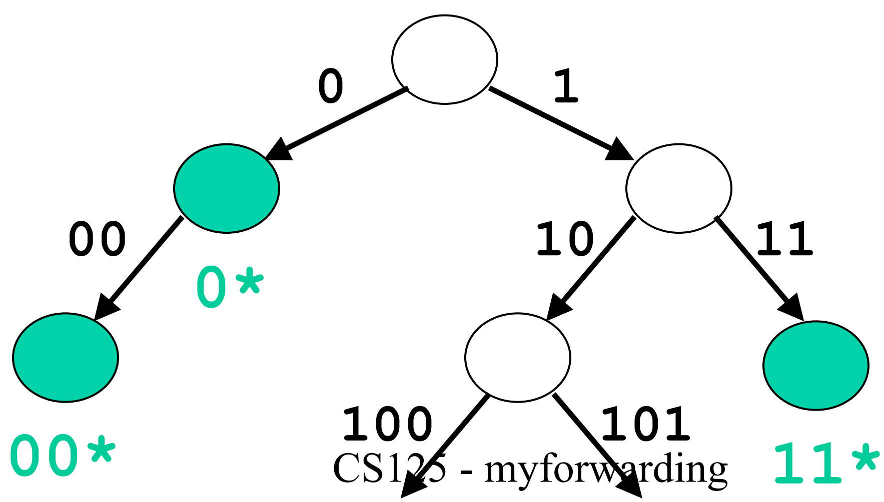
Serial0/0.1

Simplest Algorithm is Too Slow

- Scan the forwarding table one entry at a time
  - See if the destination matches the entry
  - If so, check the size of the mask for the prefix
  - Keep track of the entry with longest-matching prefix
- Overhead is linear in size of the forwarding table
  - Today, that means 150,000-200,000 entries!
  - And, the router may have just a few nanoseconds
  - … before the next packet is arriving
- Need greater efficiency to keep up with *line rate*
  - Better algorithms
  - Hardware implementations

Patricia Tree

- Store the prefixes as a tree
  - One bit for each level of the tree
  - Some nodes correspond to valid prefixes
  - ... which have next-hop interfaces in a table
- When a packet arrives
  - Traverse the tree based on the destination address
  - Stop upon reaching the longest matching prefix



Even Faster Lookups

- Patricia tree is faster than linear scan
  – Proportional to number of bits in the address
- Patricia tree can be made faster
  – Can make a k-ary tree
    - E.g., 4-ary tree with four children (00, 01, 10, and 11)
  – Faster lookup, though requires more space
- Can use special hardware
  – Content Addressable Memories (CAMs)
  – Allows look-ups on a key rather than flat address
- Huge innovations in the mid-to-late 1990s
  – After CIDR was introduced (in 1994)
  – … and longest-prefix match was a major bottleneck

Where do Forwarding Tables Come From?

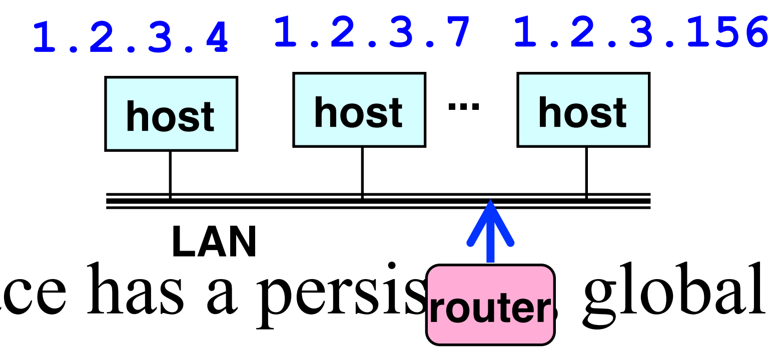
- Routers have forwarding tables
  - Map prefix to outgoing link(s)
- Entries can be statically configured
  - E.g., "map 12.34.158.0/24 to Serial0/0.1"
- But, this doesn't adapt
  - To failures
  - To new equipment
  - To the need to balance load
  - …
- That is where other technologies come in…
  - Routing protocols, DHCP, and ARP

What End Hosts Sending to Others?

- End host with single network interface
  - PC with an Ethernet link
  - Laptop with a wireless link
- Don't need to run a routing protocol
  - Packets to the host itself (e.g., 1.2.3.4/32)
    - Delivered locally
  - Packets to other hosts on the LAN (e.g., 1.2.3.0/24)
    - Sent out the interface
  - Packets to external hosts (e.g., 0.0.0.0/0)
    - Sent out interface to local gateway
- How this information is learned
  - Static setting of address, subnet mask, and gateway
  - Dynamic Host Configuration Protocol (DHCP)

What About Reaching the End Hosts?

- How does the last router reach the destination?



- Each interface has a persis[illegible], global identifier
  - MAC (Media Access Control) address
  - Burned in to the adaptors Read-Only Memory (ROM)
  - Flat address structure (i.e., no hierarchy)
- Constructing an address resolution table
  - Mapping MAC address to/from IP address
  - Address Resolution Protocol (ARP)

Conclusions

- IP address
  - A 32-bit number
  - Allocated in prefixes
  - Non-uniform hierarchy for scalability and flexibility
- Packet forwarding
  - Based on IP prefixes
  - Longest-prefix-match forwarding