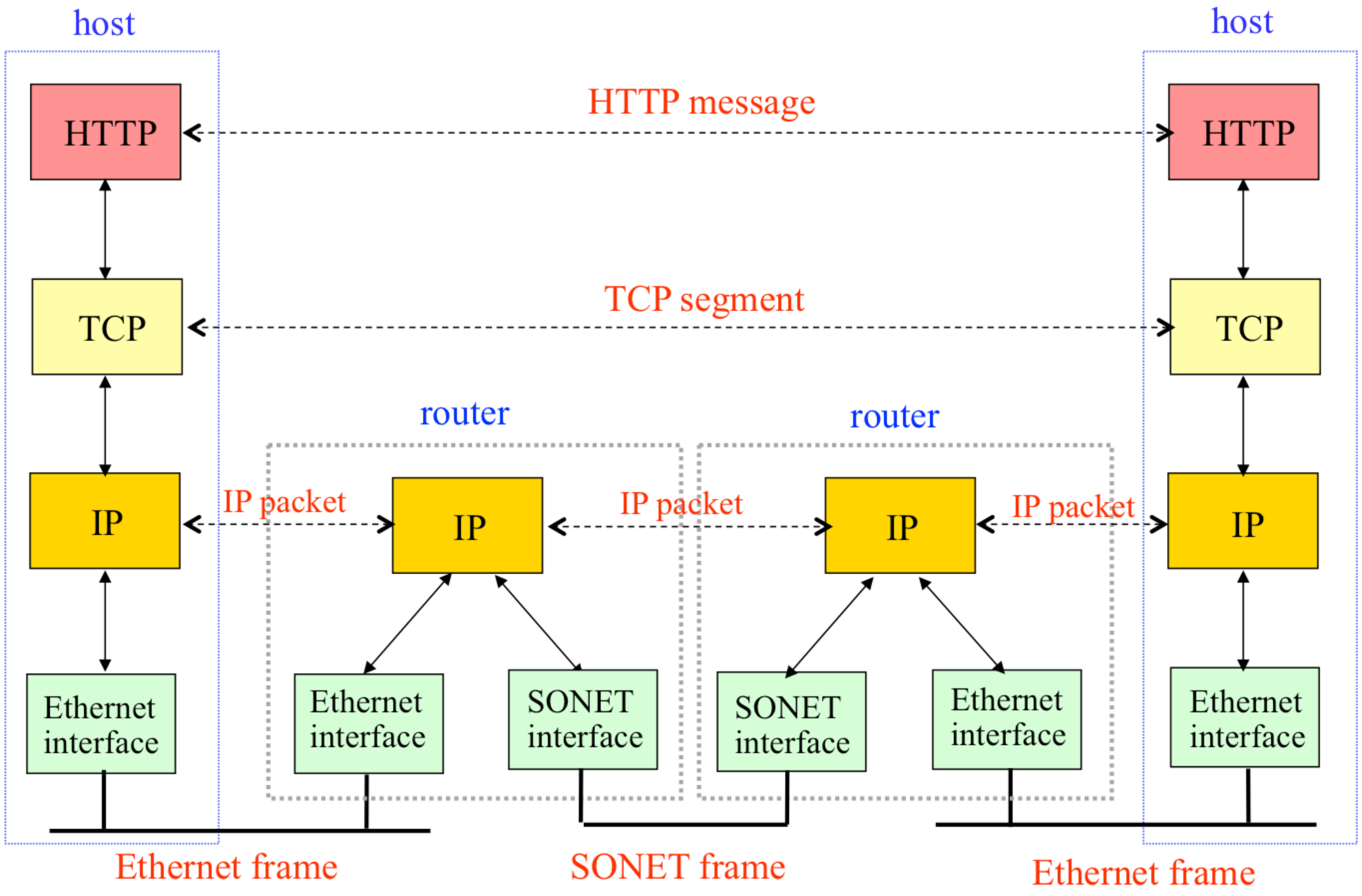
CMSC417

Computer Networks

Prof. Ashok K. Agrawala

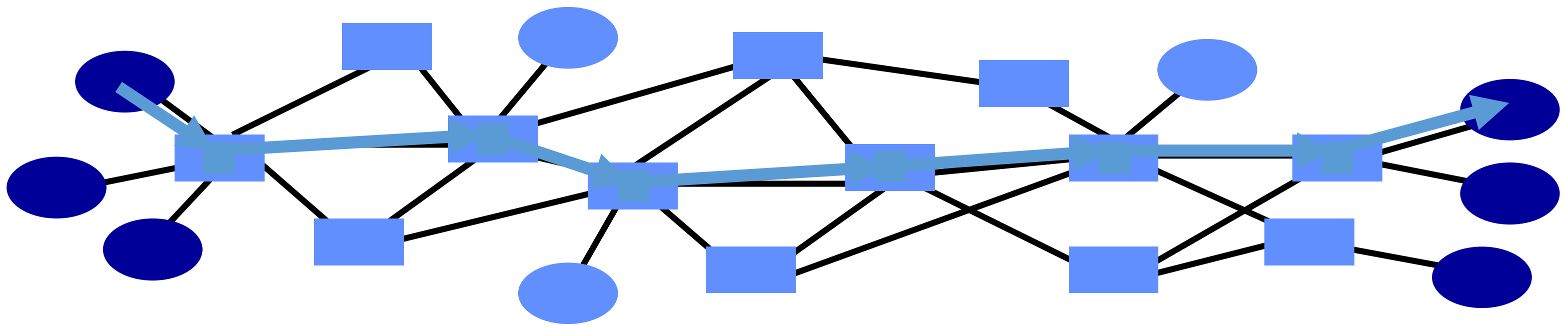
September 13, 2018

Message, Segment, Packet, and Frame

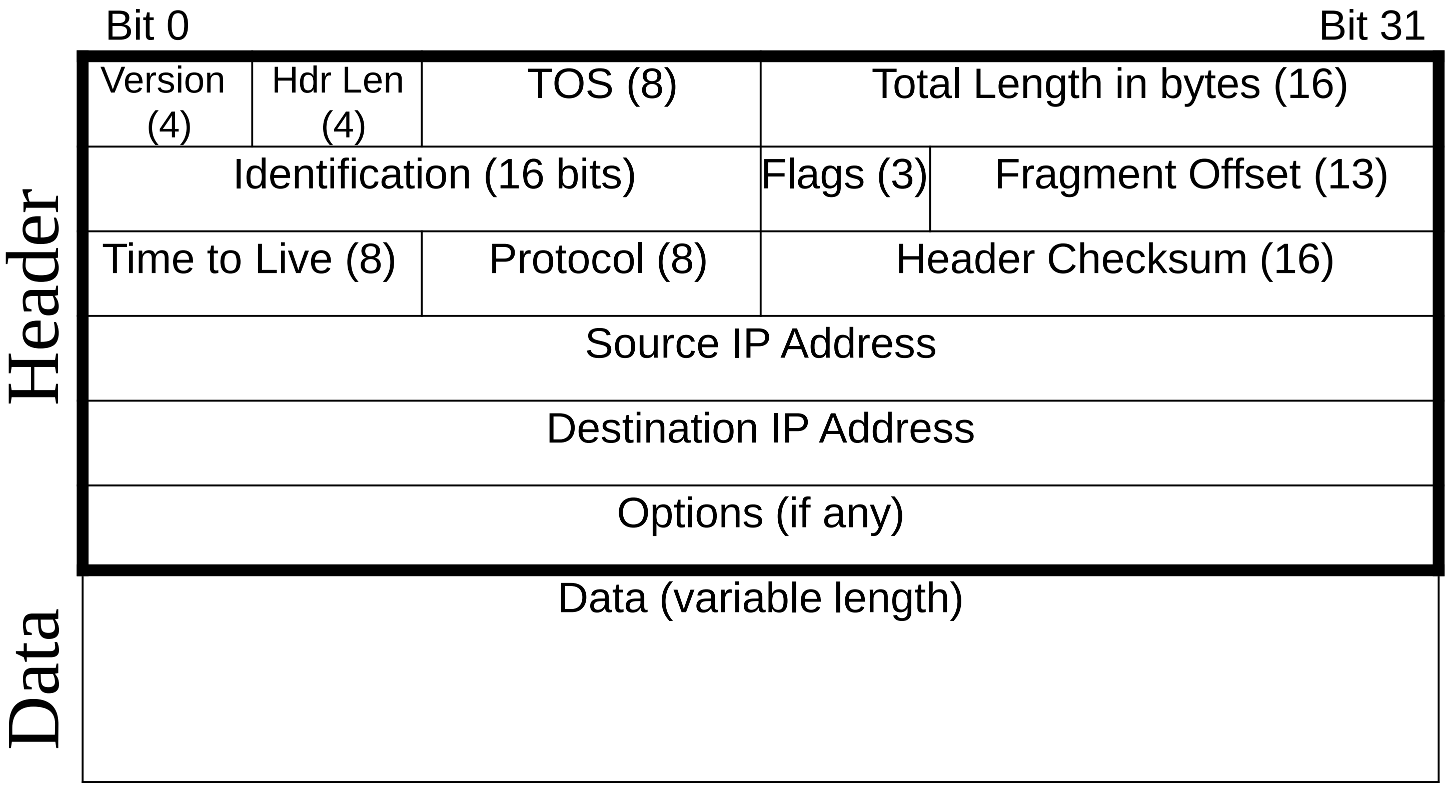


Packet Switching (e.g., Internet)

- Data traffic divided into packets; each packet contains a header (with address)
- Packets travel separately through network
  - Packet forwarding based on the header
  - Network nodes may store packets temporarily
- Destination reconstructs the message

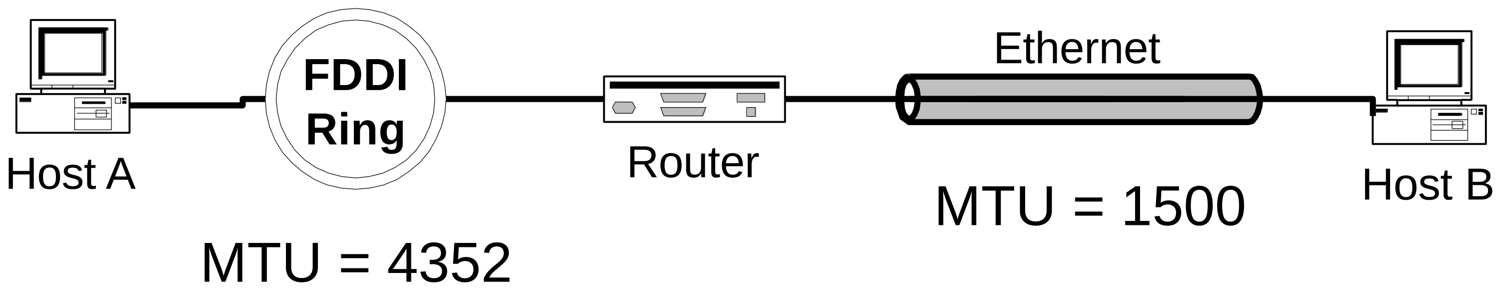


The IP Datagram

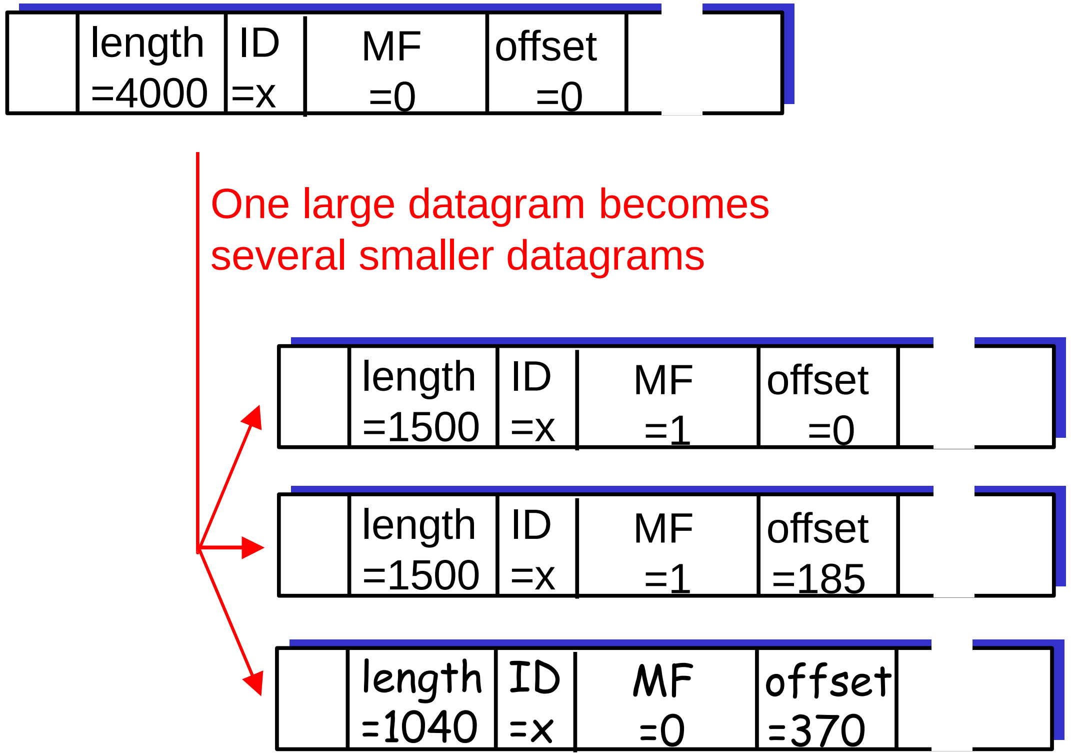


IP Fragmentation Example

Host A wants to send to Host B an IP datagram of size = 4000 Bytes



IP Fragmentation Example



Multiple Fragmenting Points

Let MTUs along internet path be

- 1500
- 1500
- 1500
- 576
- 1500

Result: fragmentation can occur twice

Fragmentation Example

- Transport layer Segment size 4500 bytes
- IP Packet Size 4520 bytes – 20 byte header – no options
- MTU – 2500 bytes

Fragment	Total bytes	Header bytes	Data bytes	"More fragments" flag	Fragment offset (8-byte blocks)
1	2500	20	2480	1	0
2	2040	20	2020	0	310

- Total data Size = 2480+ 2020 = 4500
- Offset – 0
  - 0+2480/8 =310

Fragmentation Example- Cont'd

Fragment	Total bytes	Header bytes	Data bytes	"More fragments" flag	Fragment offset (8-byte blocks)
1	2500	20	2480	1	0
2	2040	20	2020	0	310

- Assume a link with MTU of 1500 bytes follows

Fragment	Total bytes	Header bytes	Data bytes	"More fragments" flag	Fragment offset (8-byte blocks)
1	1500	20	1480	1	0
2	1020	20	1000	1	185
3	1500	20	1480	1	310
4	560	20	540	0	495

Fragmenting a Fragment

- Needed when fragment too large for network MTU
- Arbitrary subfragmentation possible
- Router divides fragments into smaller pieces All fragments at same "level"
- Offset given with respect to original datagram
- Destination cannot distinguish subfragments

Fragment Lost

Receiver

- Collects incoming fragments
- Reassembles when all fragments arrive
- Does not know identity of router that did fragmentation
- Cannot request missing pieces

Consequence: Loss of one fragment means entire datagram lost

IP Packet Header (Continued)

- Two IP addresses
  - Source IP address (32 bits)
  - Destination IP address (32 bits)
- Destination address
  - Unique identifier for the receiving host
  - Allows each node to make forwarding decisions
- Source address
  - Unique identifier for the sending host
  - Recipient can decide whether to accept packet
  - Enables recipient to send a reply back to source

The IP Protocol

Some of the IP options

Option	Description
Security	Specifies how secret the datagram is
Strict source routing	Gives the complete path to be followed
Loose source routing	Gives a list of routers not to be missed
Record route	Makes each router append its IP address
Timestamp	Makes each router append its address and timestamp

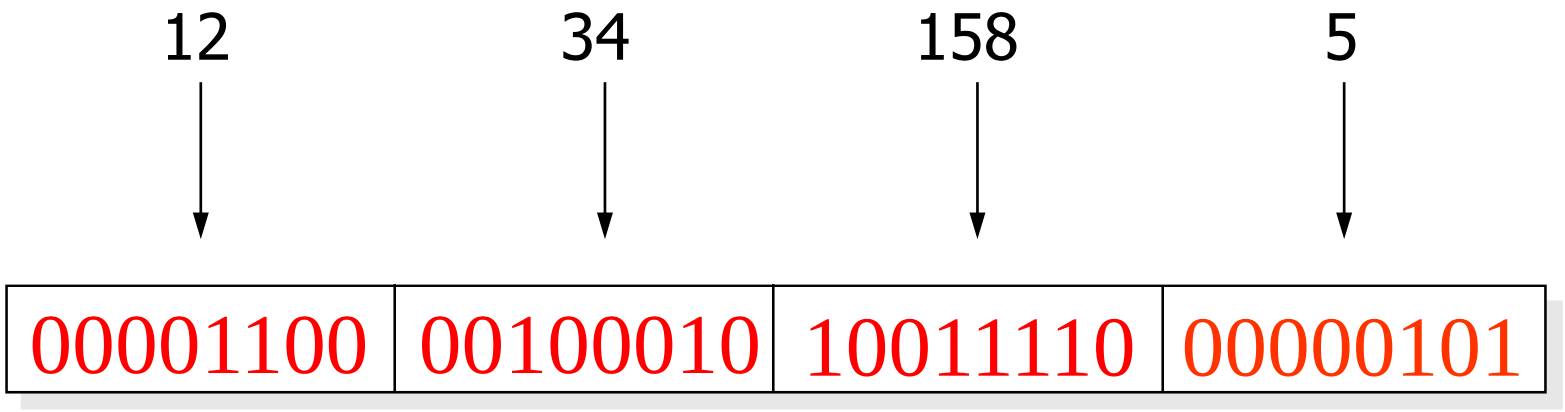
What if the Source Lies?

- Source address should be the sending host
  - But, who's checking, anyway?
  - You could send packets with any source you want
- Why would someone want to do this?
  - Launch a denial-of-service attack
    - Send excessive packets to the destination
    - This overloads the node, or the links leading to the node
  - Evade detection by "spoofing"
    - But, the victim could identify you by the source address
    - So, you can put someone else's source address in the packets
  - Also, an attack against the spoofed host
    - Spoofed host is wrongly blamed
    - Spoofed host may receive return traffic from the receiver

IP Addressing and Forwarding

IP Address (IPv4)

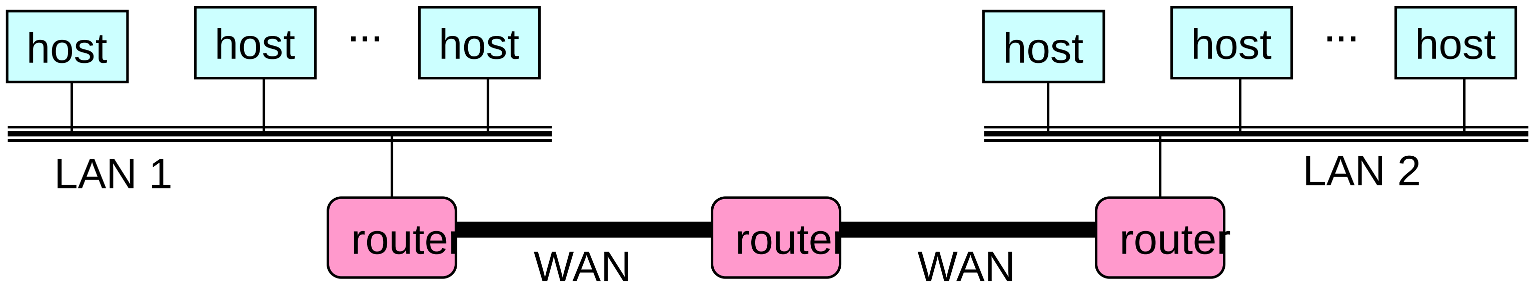
- A unique 32-bit number
- Identifies an interface (on a host, on a router, …)
- Represented in dotted-quad notation



Grouping Related Hosts

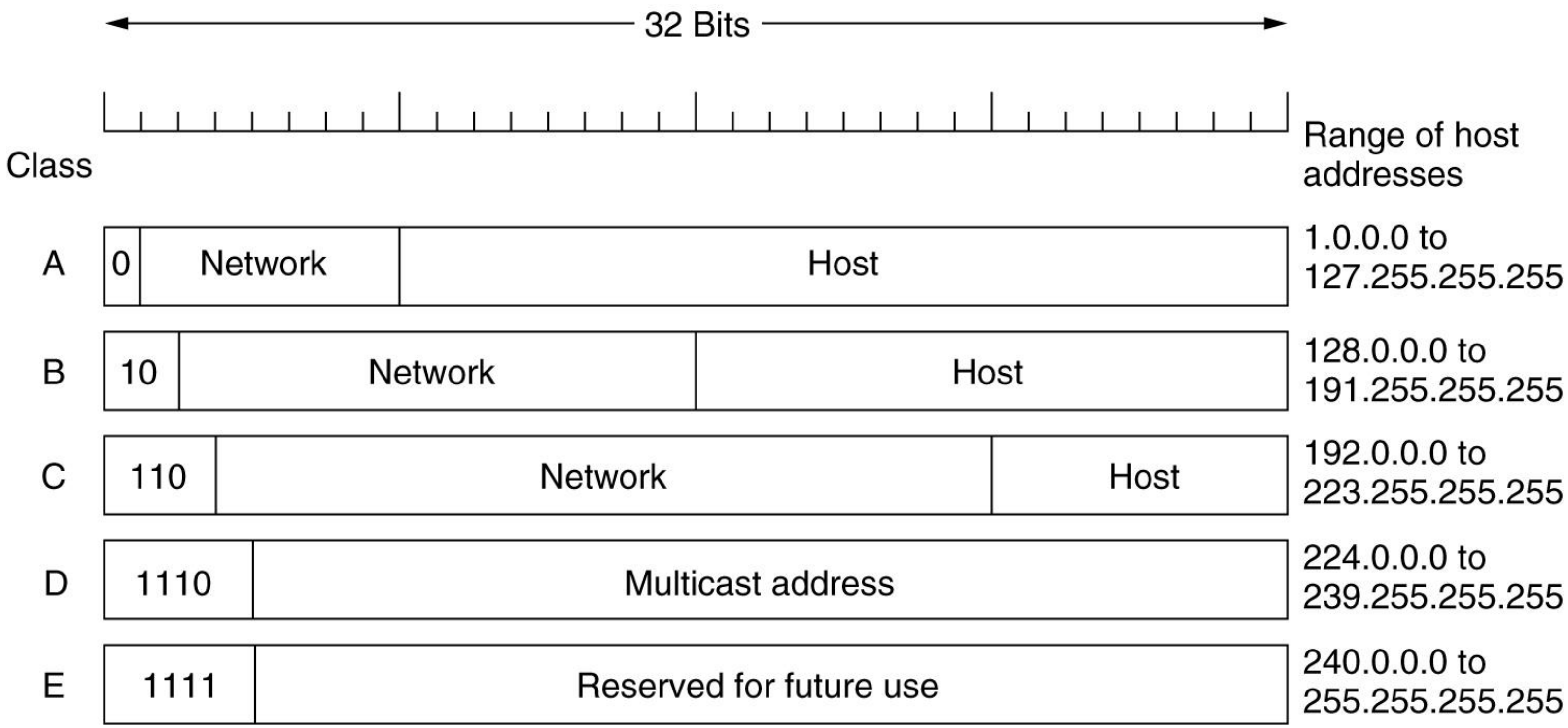
The Internet is an "inter-network".

- Used to connect *networks* together, not *hosts*
- Needs a way to address a network (i.e., group of hosts)



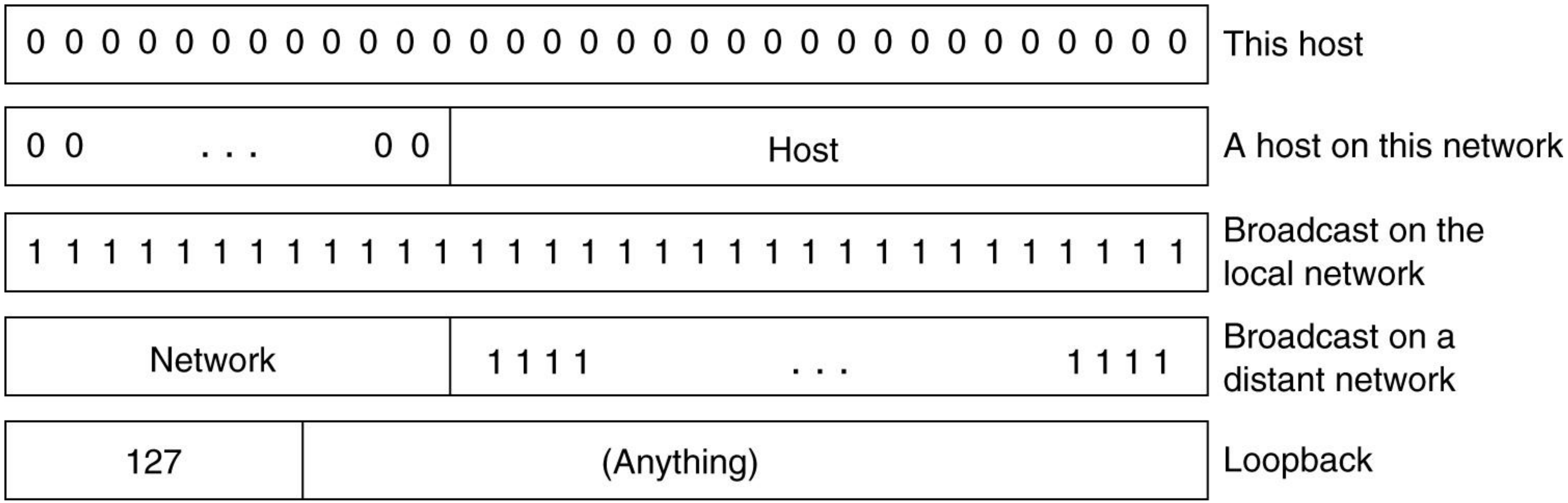
LAN: Local Area Network
WAN: Wide Area Network

IP Addresses



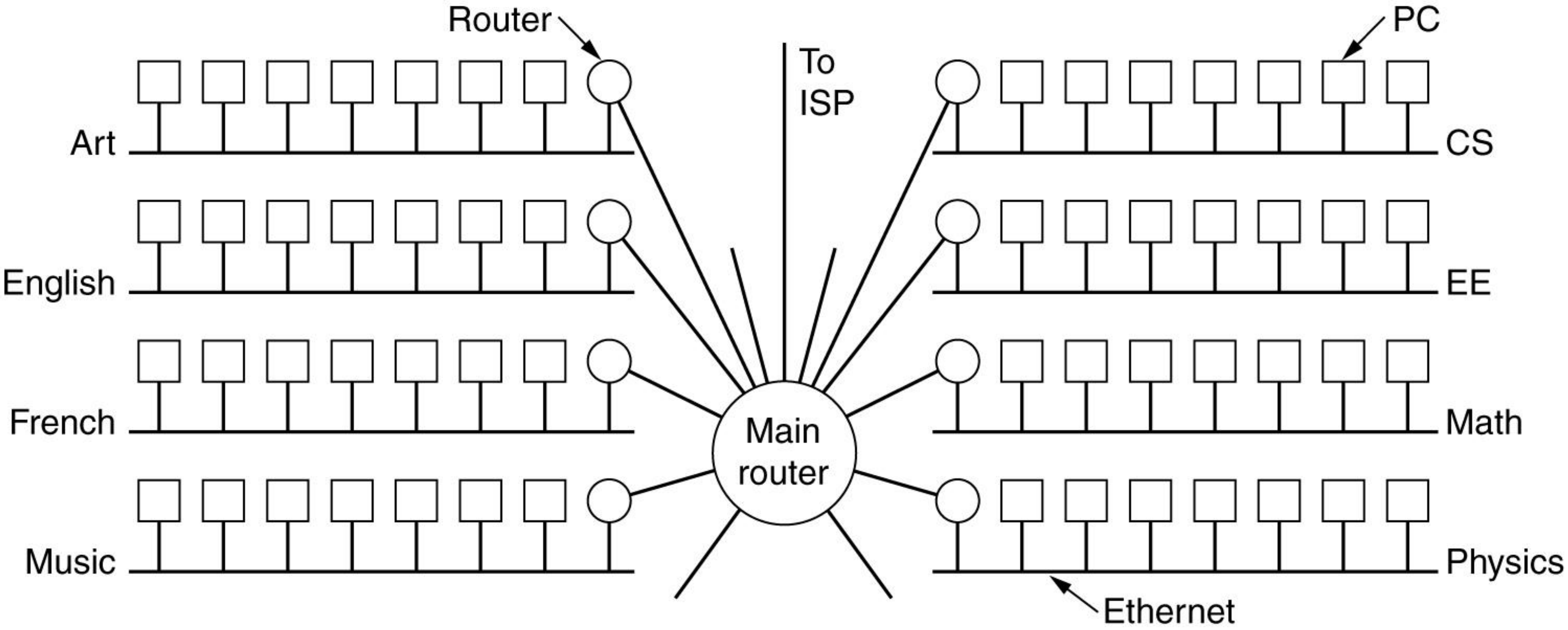
IP Addresses

Special IP addresses



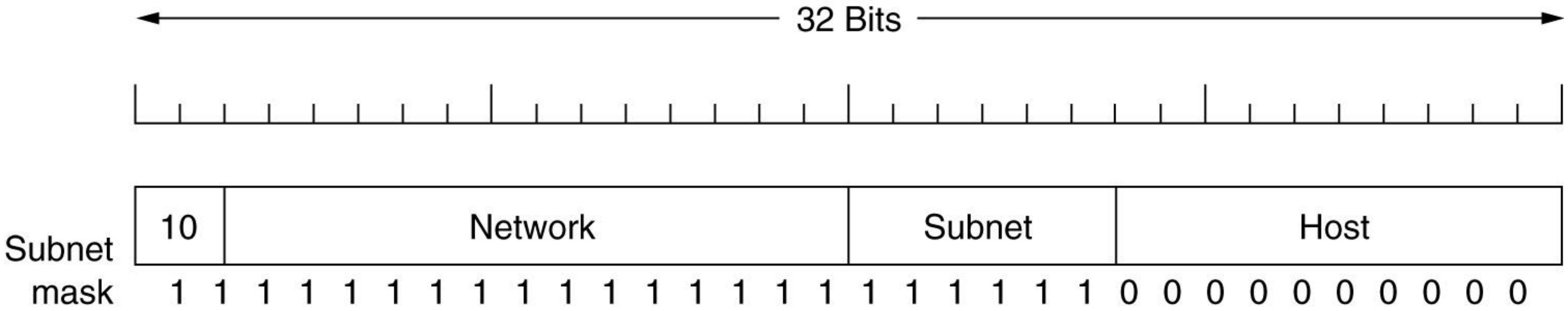
Subnets

A campus network consisting of LANs for various departments

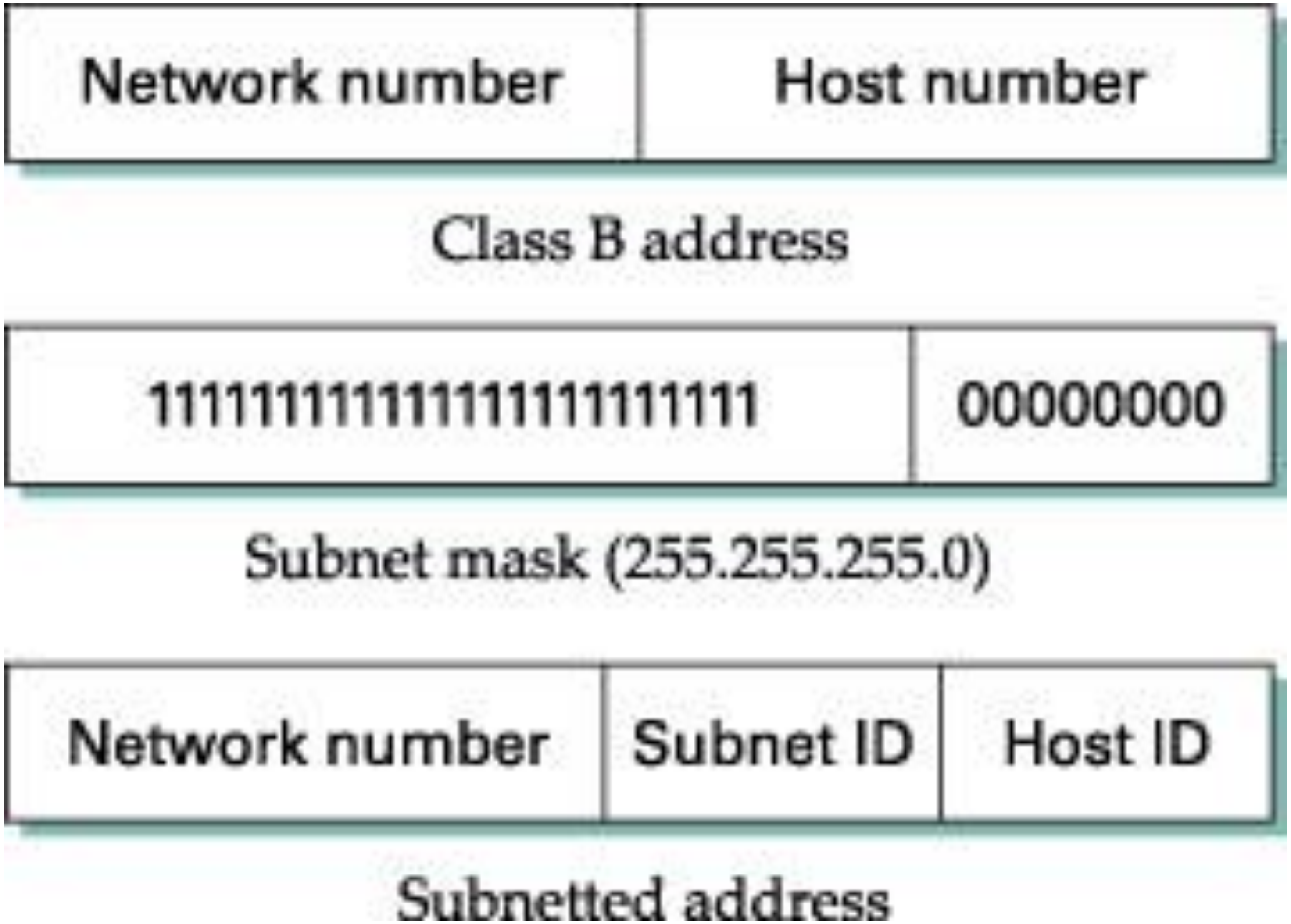


Subnets

A class B network subnetted into 64 subnets



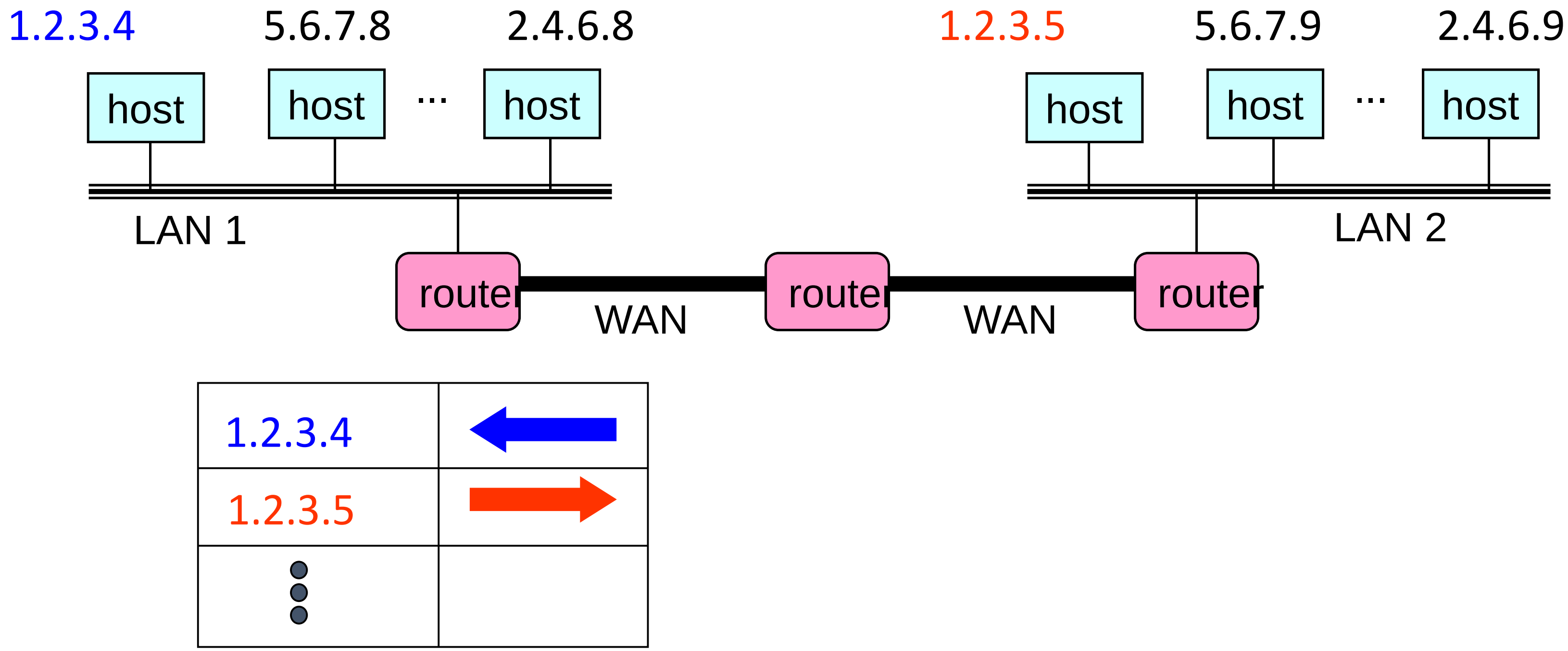
Subnetted Address



Scalability Challenge

Suppose hosts had arbitrary addresses

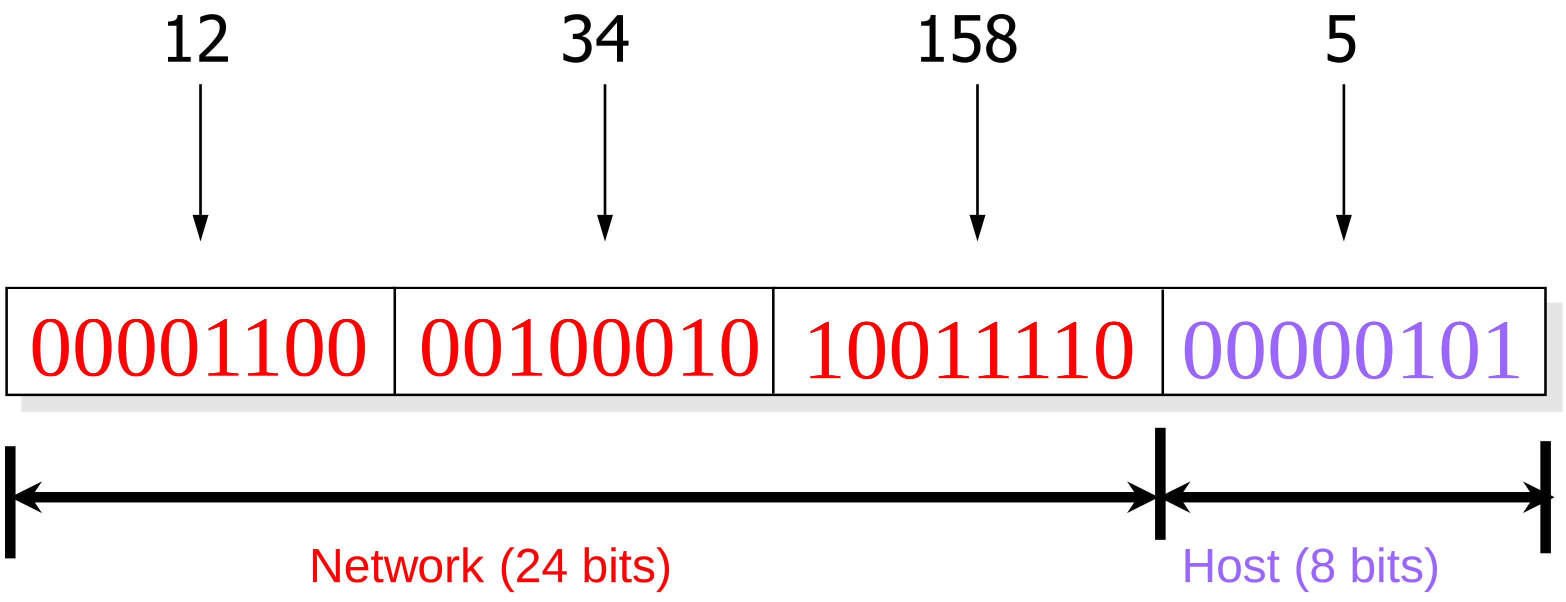
- Then every router would need a lot of information
- …to know how to direct packets toward the host



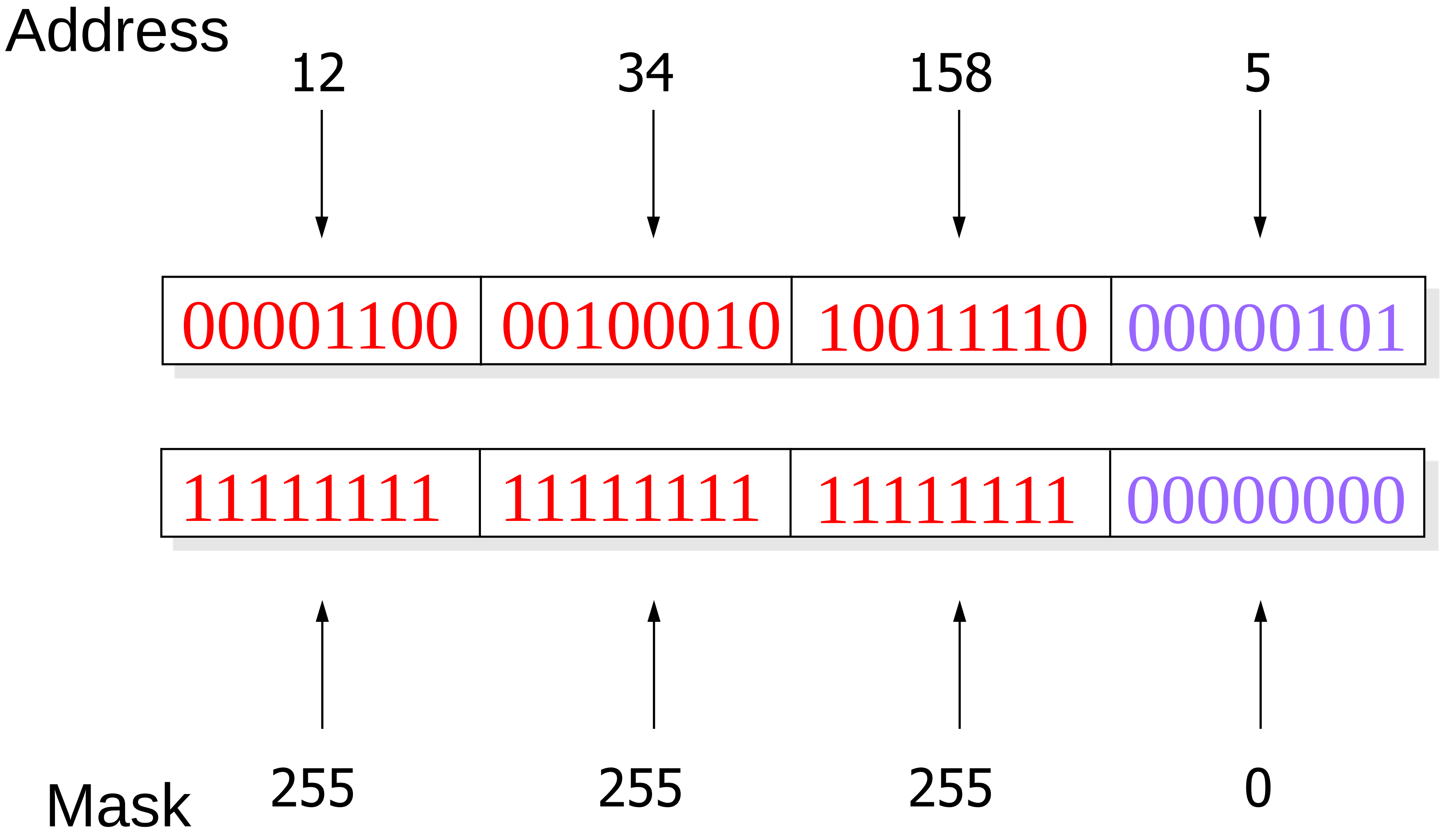
forwarding table

Hierarchical Addressing: IP Prefixes

- Divided into network & host portions (left and right)
- 12.34.158.0/24 is a 24-bit prefix with 2^8 addresses



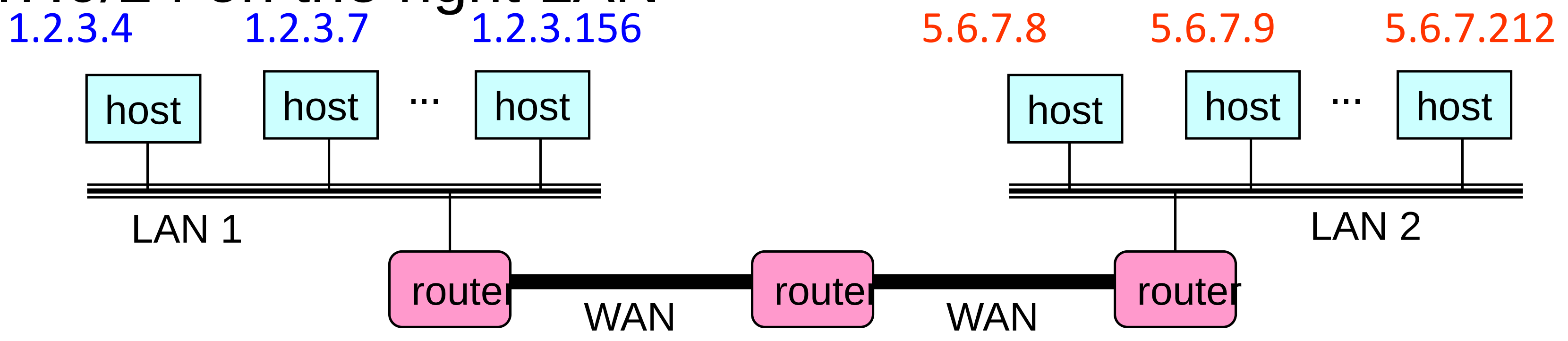
IP Address and a 24-bit Subnet Mask



Scalability Improved

Number related hosts from a common subnet

- 1.2.3.0/24 on the left LAN
- 5.6.7.0/24 on the right LAN



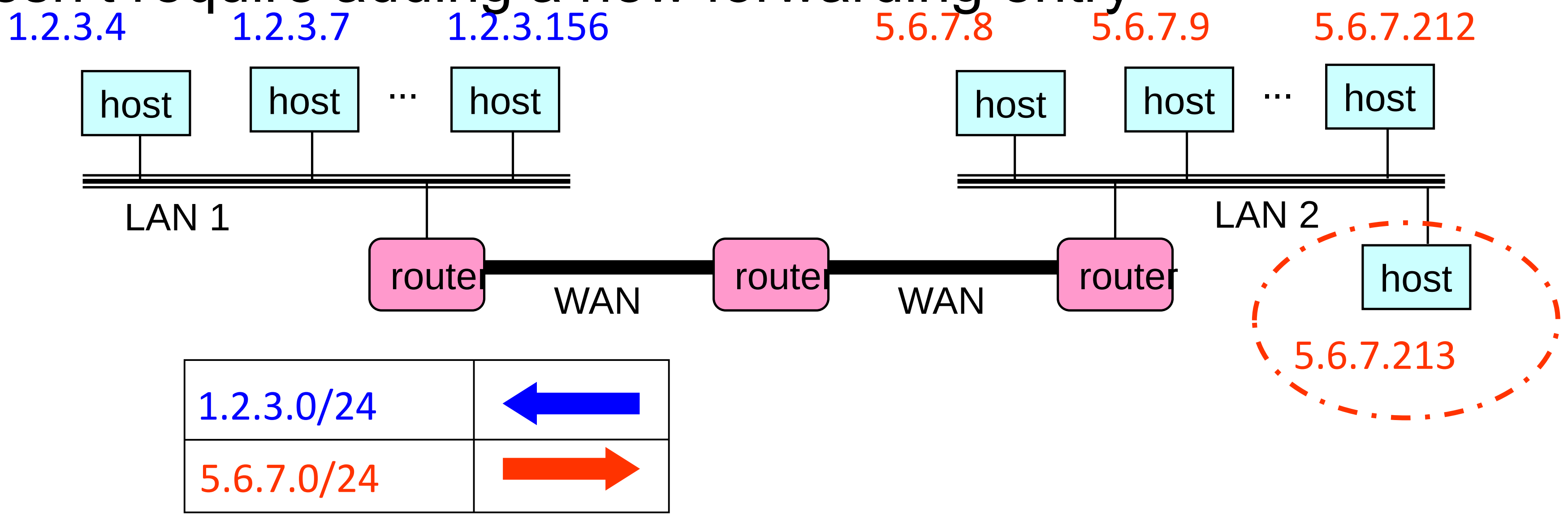
1.2.3.0/24	←
5.6.7.0/24	→

forwarding table

Easy to Add New Hosts

No need to update the routers

- E.g., adding a new host 5.6.7.213 on the right
- Doesn't require adding a new forwarding entry



forwarding table

Address Allocation

Classful Addressing

In the olden days, there were only fixed allocation sizes.

Class	Leading Bits	Purpose
A	0	Very large /8 blocks (e.g., MIT has 18.0.0.0/8)
B	10	Large /16 blocks (e.g,. Princeton has 128.112.0.0/16)
C	110	Small /24 blocks (e.g., AT&T Labs has 192.20.225.0/24)
D	1110	Multicast groups
E	11110	Reserved for future use

Classless Inter-Domain Routing (CIDR)

Use two 32-bit numbers to represent a network.
Network number = IP address + Mask

IP Address : 12.4.0.0   IP Mask: 255.254.0.0

Address	00001100	00000100	00000000	00000000
Mask	11111111	11111110	00000000	00000000

|← Network Prefix →|← for hosts →|

Written as 12.4.0.0/15

CDR – Classless InterDomain Routing

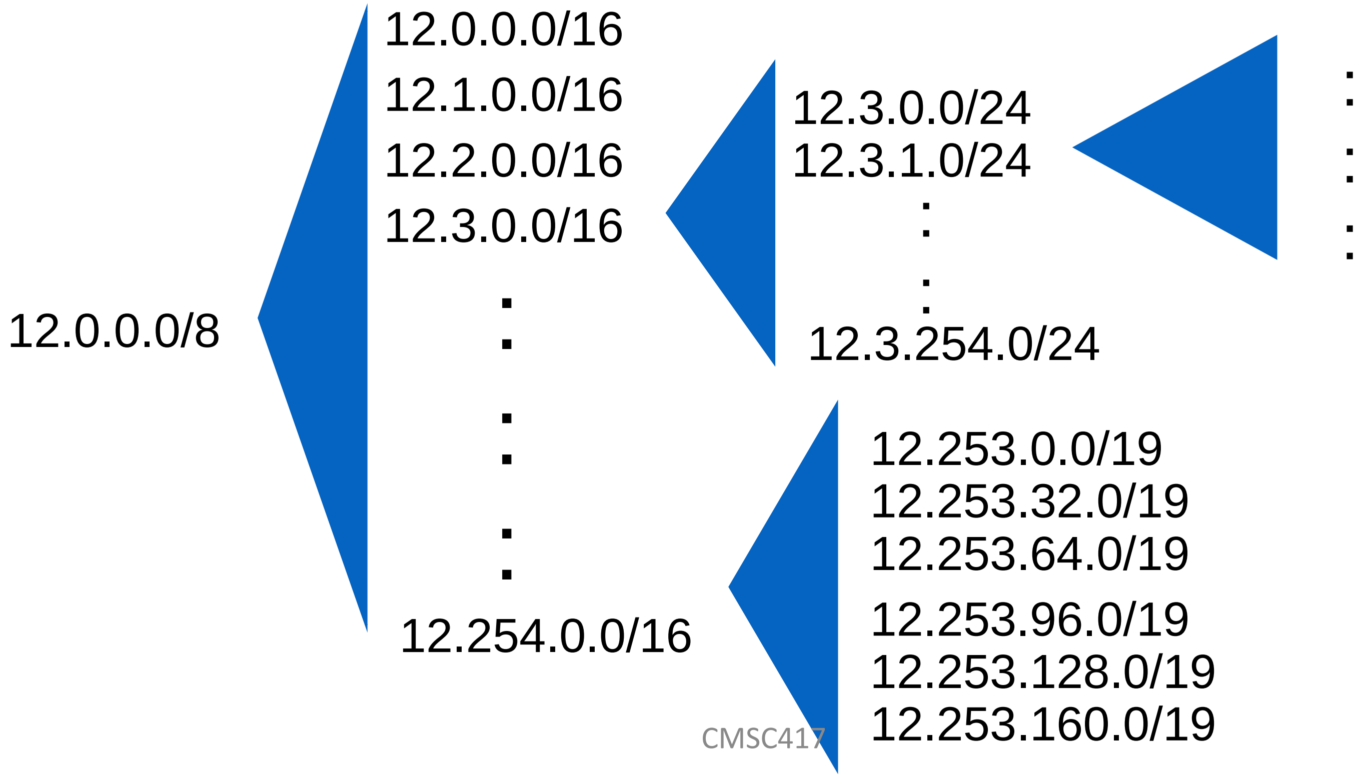
A set of IP address assignments

University	First address	Last address	How many	Written as
Cambridge	194.24.0.0	194.24.7.255	2048	194.24.0.0/21
Edinburgh	194.24.8.0	194.24.11.255	1024	194.24.8.0/22
(Available)	194.24.12.0	194.24.15.255	1024	194.24.12/22
Oxford	194.24.16.0	194.24.31.255	4096	194.24.16.0/20

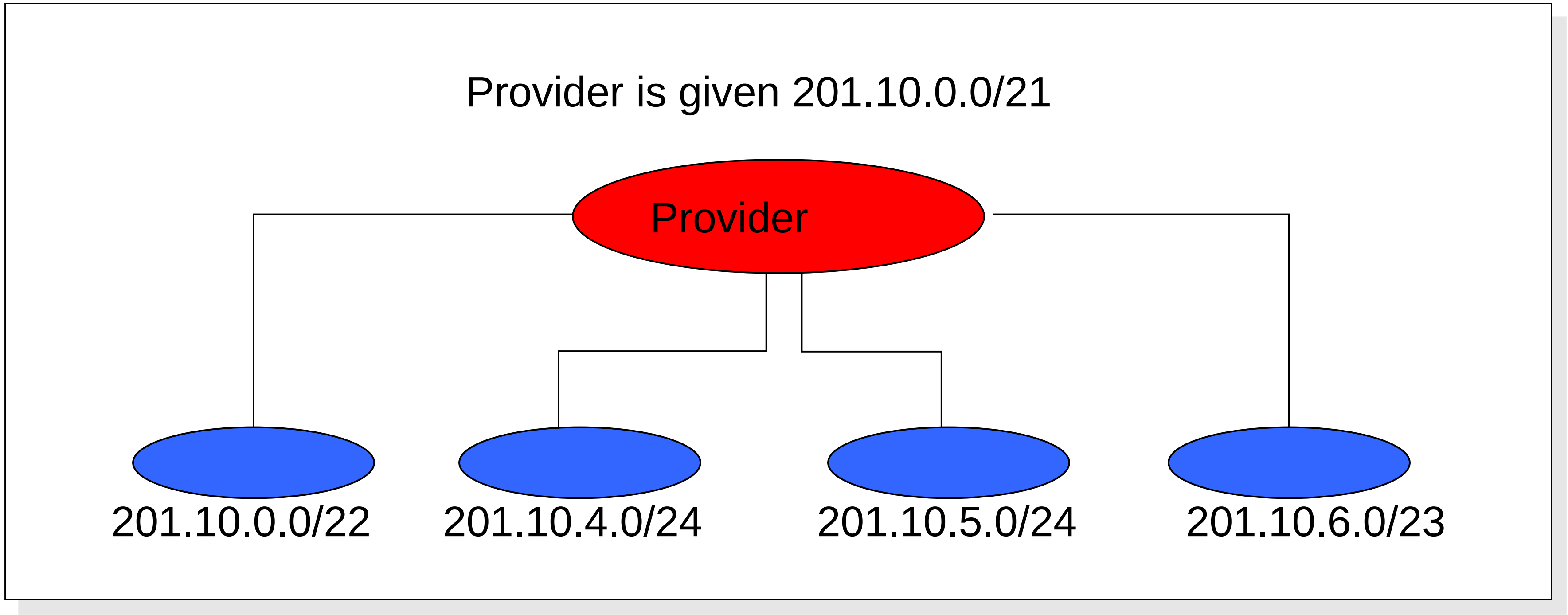
CIDR: Hierarchal Address Allocation

Prefixes are key to Internet scalability

- Address allocated in contiguous chunks (prefixes)
- Routing protocols and packet forwarding based on prefixes
- Today, routing tables contain ~150,000-200,000 prefixes

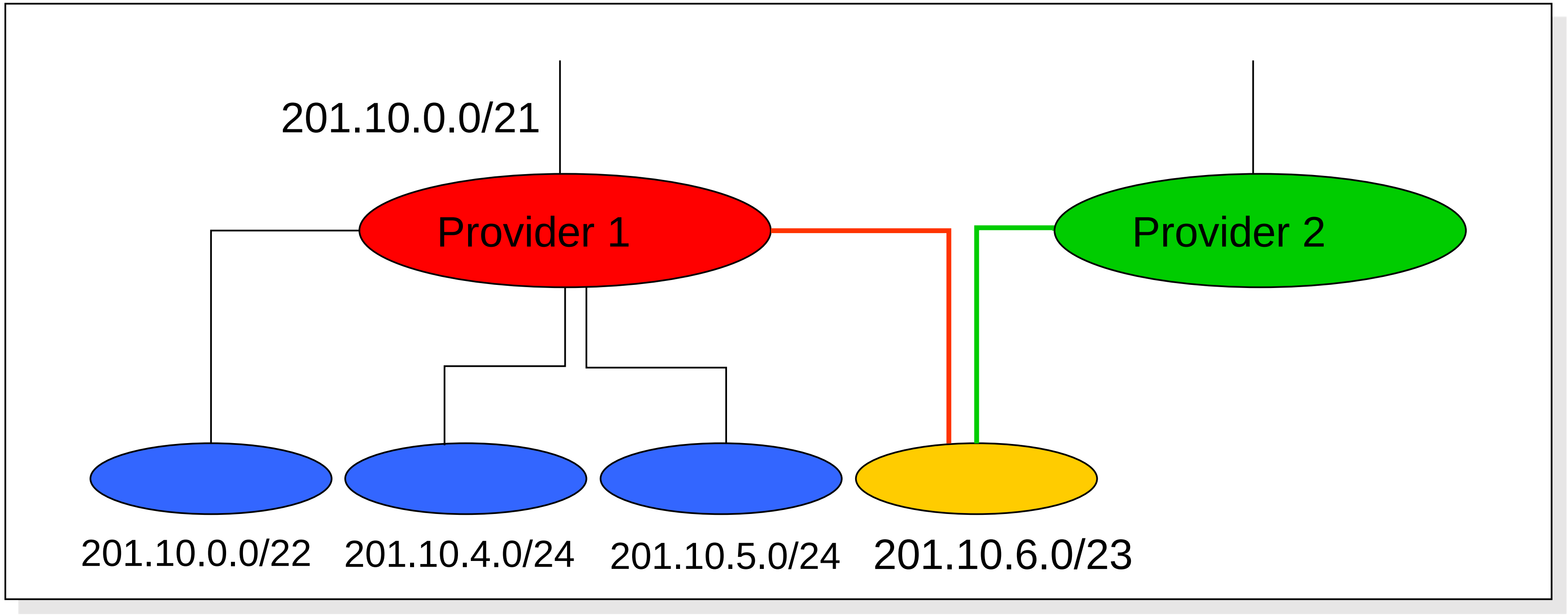


Scalability: Address Aggregation



Routers in the rest of the Internet just need to know how to reach **201.10.0.0/21**. The provider can direct the IP packets to the appropriate **customer**.

But, Aggregation Not Always Possible



Multi-homed customer with **201.10.6.0/23** has two providers. Other parts of the Internet need to know how to reach these destinations through **both** providers.

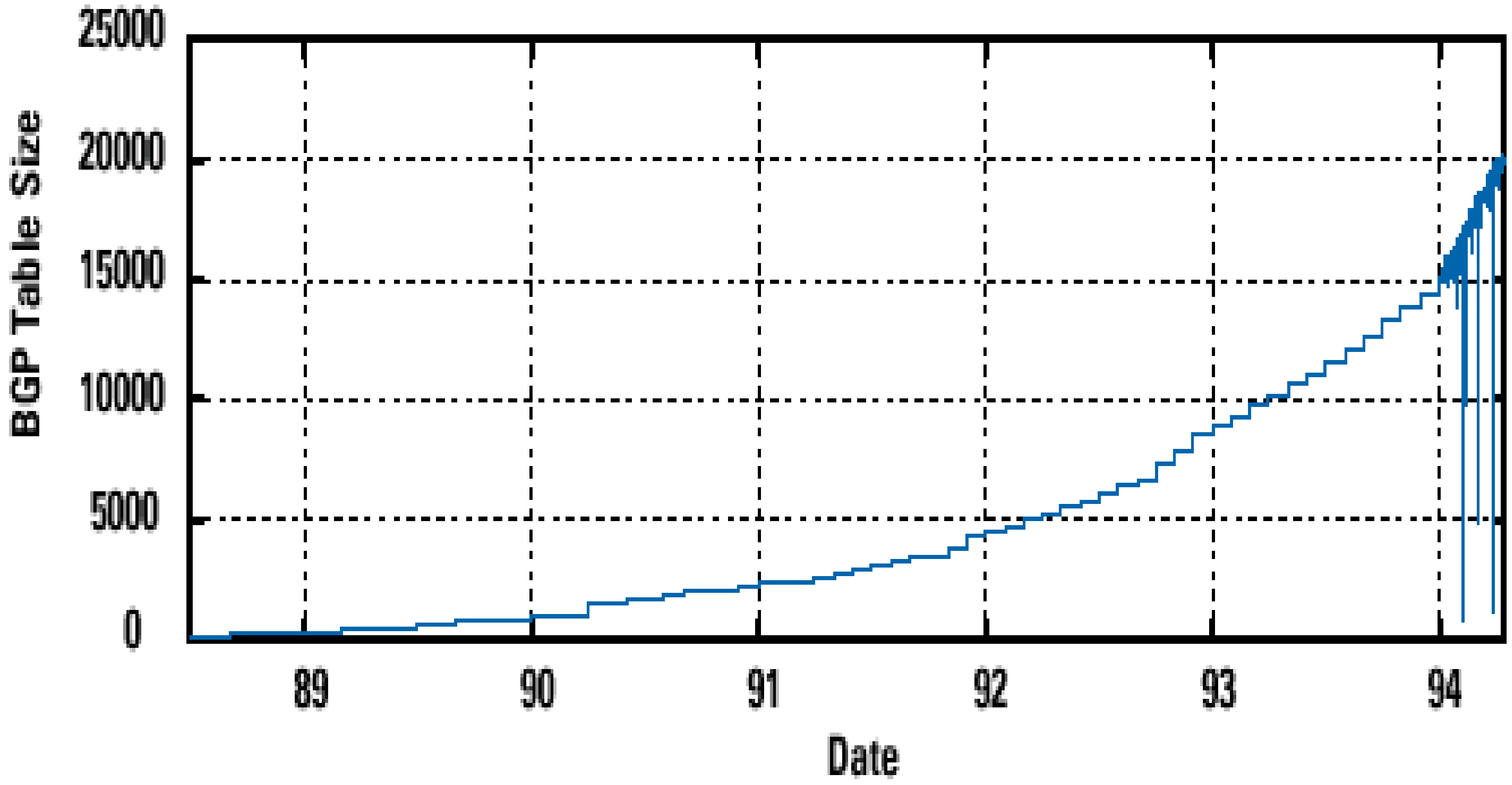
Scalability Through Hierarchy

- Hierarchical addressing
  - Critical for scalable system
  - Don’t require everyone to know everyone else
  - Reduces amount of updating when something changes
- Non-uniform hierarchy
  - Useful for heterogeneous networks of different sizes
  - Initial class-based addressing was far too coarse
  - Classless InterDomain Routing (CIDR) helps

Next Few Slides

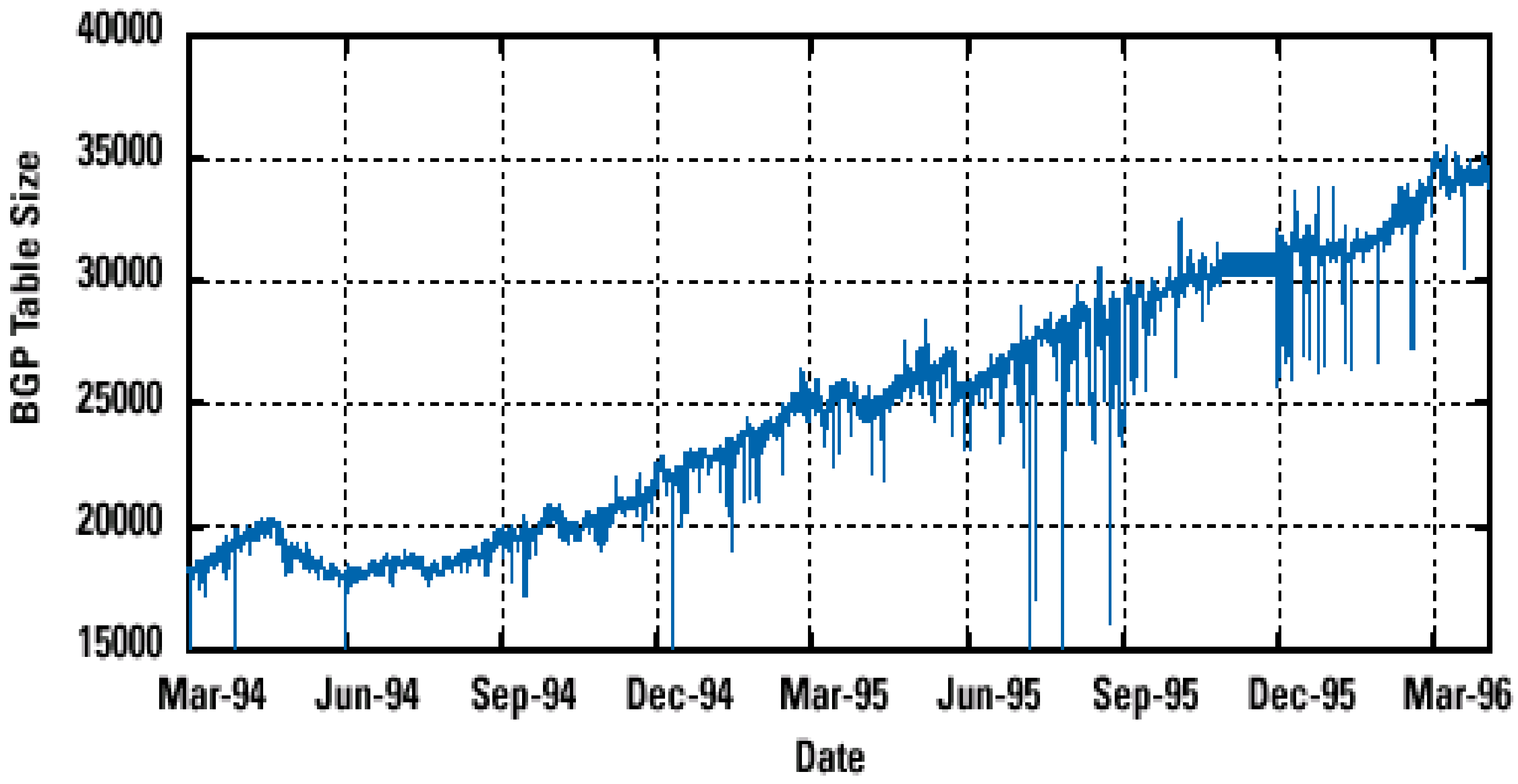
- History of the number of globally-visible prefixes
- Plots are # of prefixes vs. time

Pre-CIDR (1988-1994): Steep Growth



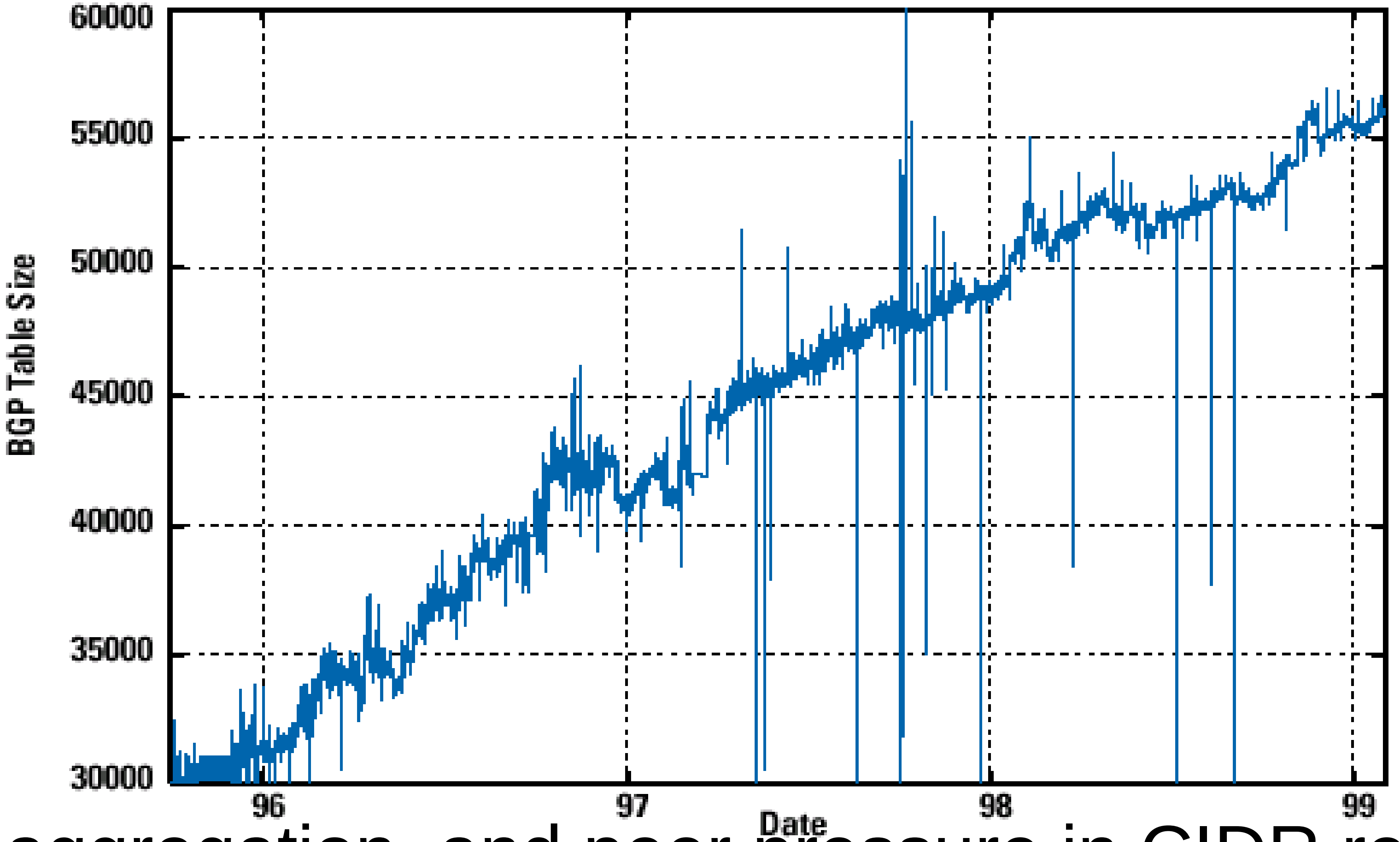
Growth faster than improvements in equipment capability

CIDR Deployed (1994-1996): Much Flatter



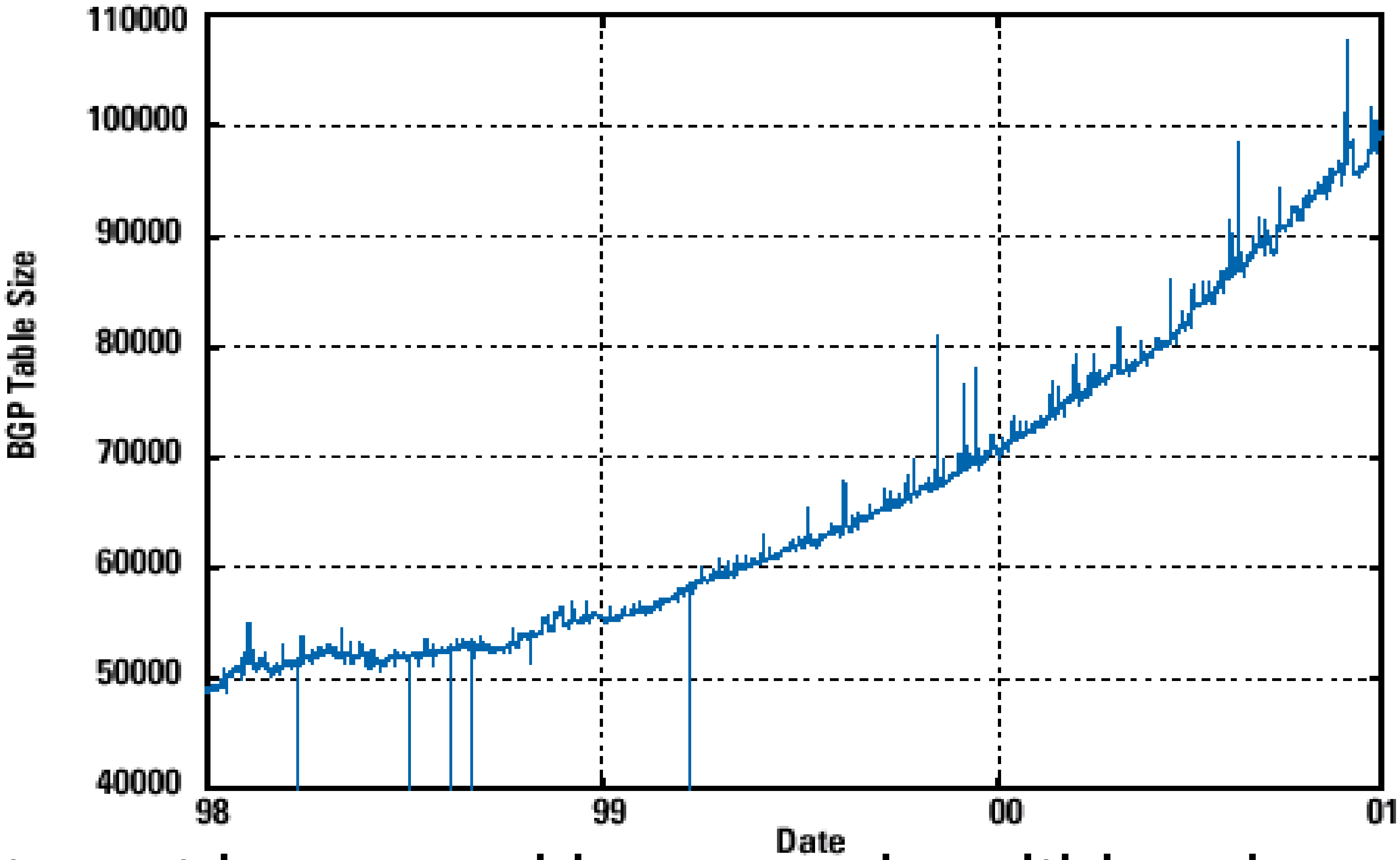
Efforts to aggregate (even decreases after IETF meetings!)

CIDR Growth (1996-1998): Roughly Linear



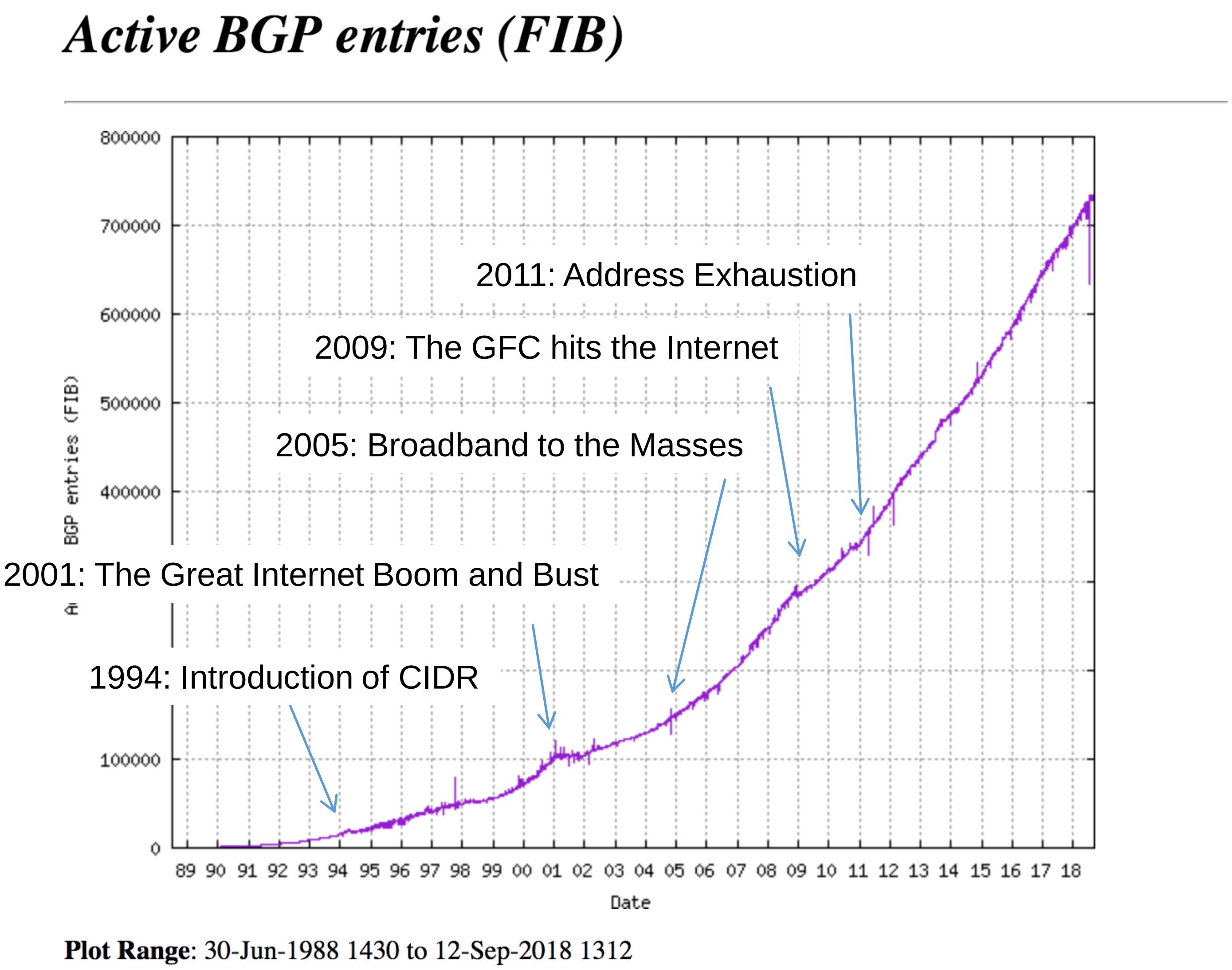
Good use of aggregation, and peer pressure in CIDR report

Boom Period (1998-2001): Steep Growth



Internet boom and increased multi-homing

The Big Picture of the v4 Routing Table



Obtaining a Block of Addresses

- Separation of control
  - Prefix: assigned *to* an institution
  - Addresses: assigned *by* the institution to their nodes
- Who assigns prefixes?
  - **Internet Corporation for Assigned Names and Numbers:** Allocates large address blocks to Regional Internet Registries
  - **Regional Internet Registries** (RIRs)
    - e. g. **ARIN** (American Registry for Internet Numbers)
    - Allocates address blocks within their regions
    - Allocated to Internet Service Providers and large institutions
  - **Internet Service Providers** (ISPs)
    - Allocate address blocks to their customers
    - Who may, in turn, allocate to their customers…

Figuring Out Who Owns an Address

- Address registries
  - –Public record of address allocations
  - –Internet Service Providers (ISPs) should update when giving addresses to customers
  - –However, records are notoriously out-of-date
- Ways to query
  - –UNIX: “whois –h whois.arin.net 128.8.130.75”
  - –http://www.arin.net/whois/
  - –http://www.geektools.com/whois.php

Example Output for 128.8.130.75

OrgName: University of Maryland
OrgID: UNIVER-262
Address: Office of Information Technology
Address: Patuxent Building
City: College Park
StateProv: MD PostalCode: 20742
Country: US

NetRange: 128.8.0.0 - 128.8.255.255
CIDR: 128.8.0.0/16
NetName: UMDNET
NetHandle: NET-128-8-0-0-1
Parent: NET-128-0-0-0-0
NetType: Direct Assignment
NameServer: NOC.UMD.EDU
NameServer: NS1.UMD.EDU
NameServer: NS2.UMD.EDU
NameServer: NASANS4.NASA.GOV
Comment:
RegDate:
Updated: 2004-04-12

RTechHandle: UM-ORG-ARIN
RTechName: UMD DNS Admin Role Account
RTechPhone: +1-301-405-3003
RTechEmail: dnsadmin@noc.net.umd.edu

OrgAbuseHandle: UARA-ARIN
OrgAbuseName: UMD Abuse Role Account
OrgAbusePhone: +1-301-405-8787
OrgAbuseEmail: abuse@umd.edu

OrgTechHandle: UM-ORG-ARIN
OrgTechName: UMD DNS Admin Role Account
OrgTechPhone: +1-301-405-3003
OrgTechEmail: dnsadmin@noc.net.umd.edu

Are 32-bit Addresses Enough?

- Not all that many unique addresses
  - 2^{32} = 4,294,967,296 (just over four billion)
  - Plus, some are reserved for special purposes
  - And, addresses are allocated in larger blocks
- And, many devices need IP addresses (computers, PDAs, routers, tanks, toasters…)

Are 32-bit Addresses Enough?

- Long-term solution: a larger address space (IPv6 has 128-bit addresses ($2^{128} = 3.403 \times 10^{38}$)
- Short-term solutions: limping along with IPv4
  - Private addresses
  - Network address translation (NAT)
  - Dynamically-assigned addresses (DHCP)

Hard Policy Questions

- How much address space per geographic region?
  - Equal amount per country?
  - Proportional to the population?
  - What about addresses already allocated?
- Address space portability?
  - Keep your address block when you change providers?
  - Pro: avoid having to renumber your equipment
  - Con: reduces the effectiveness of address aggregation
- Keeping the address registries up to date?
  - What about mergers and acquisitions?
  - Delegation of address blocks to customers?
  - As a result, the registries are horribly out of date

Packet Forwarding

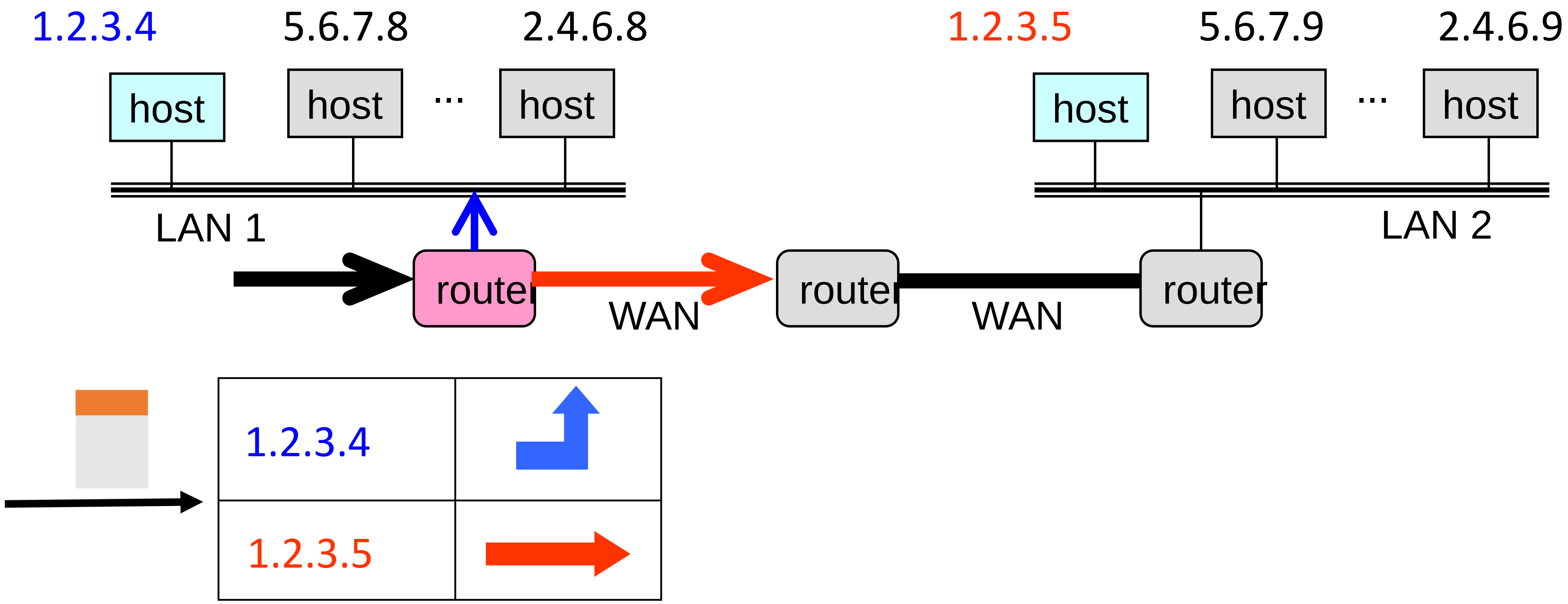
Hop-by-Hop Packet Forwarding

- Each router has a forwarding table which 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

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**

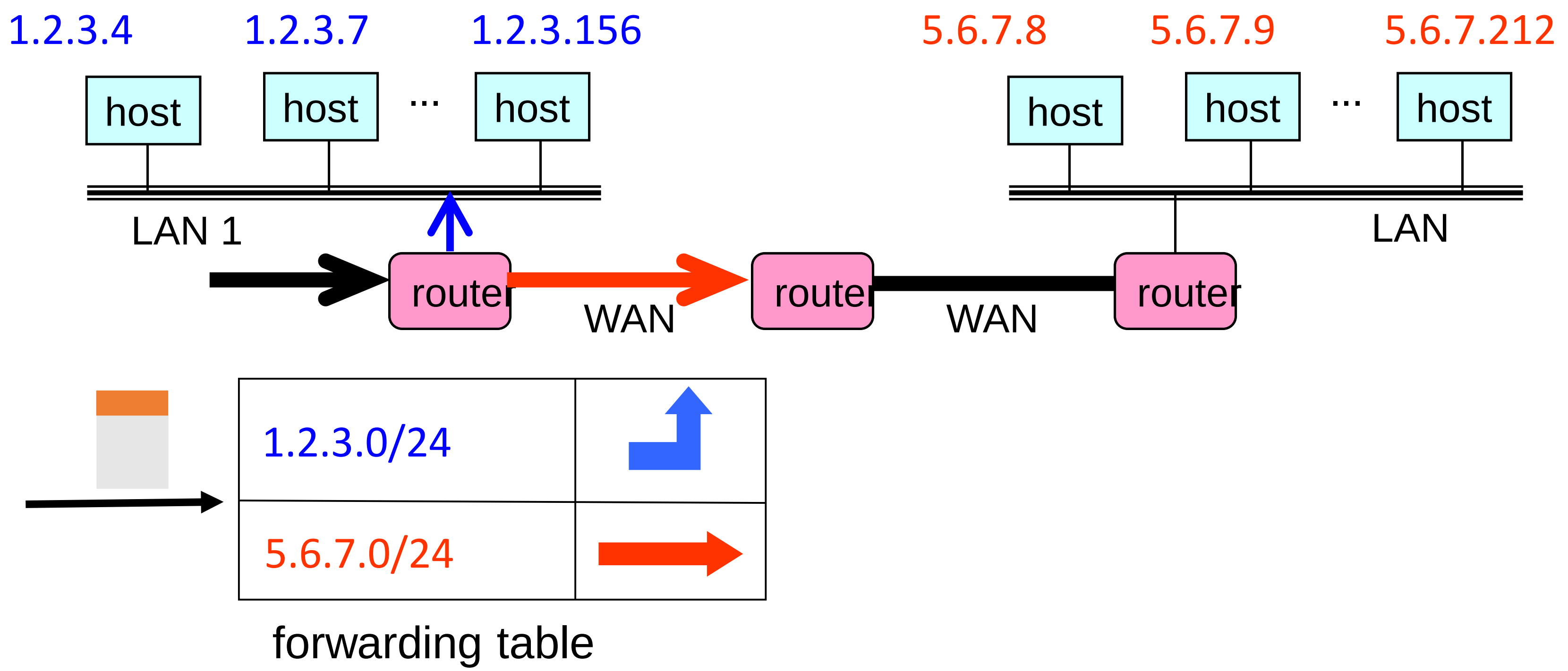


forwarding table

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

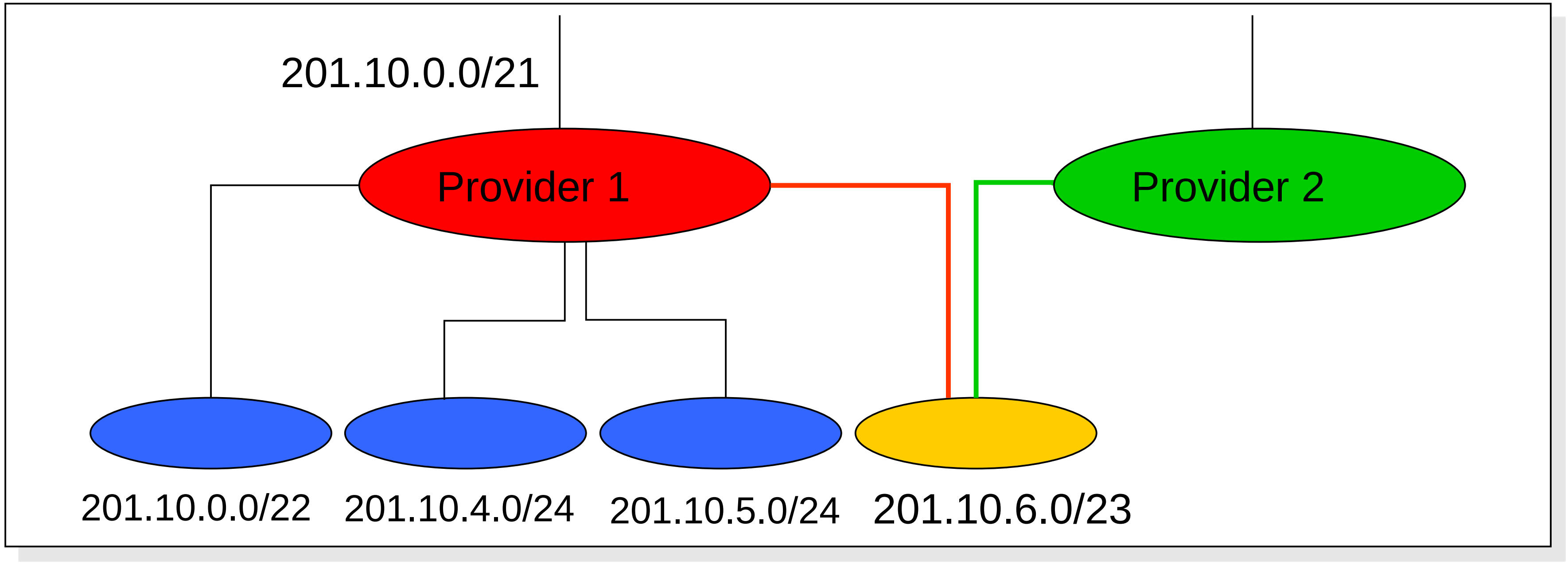


Separate Entry Classful Address

- 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

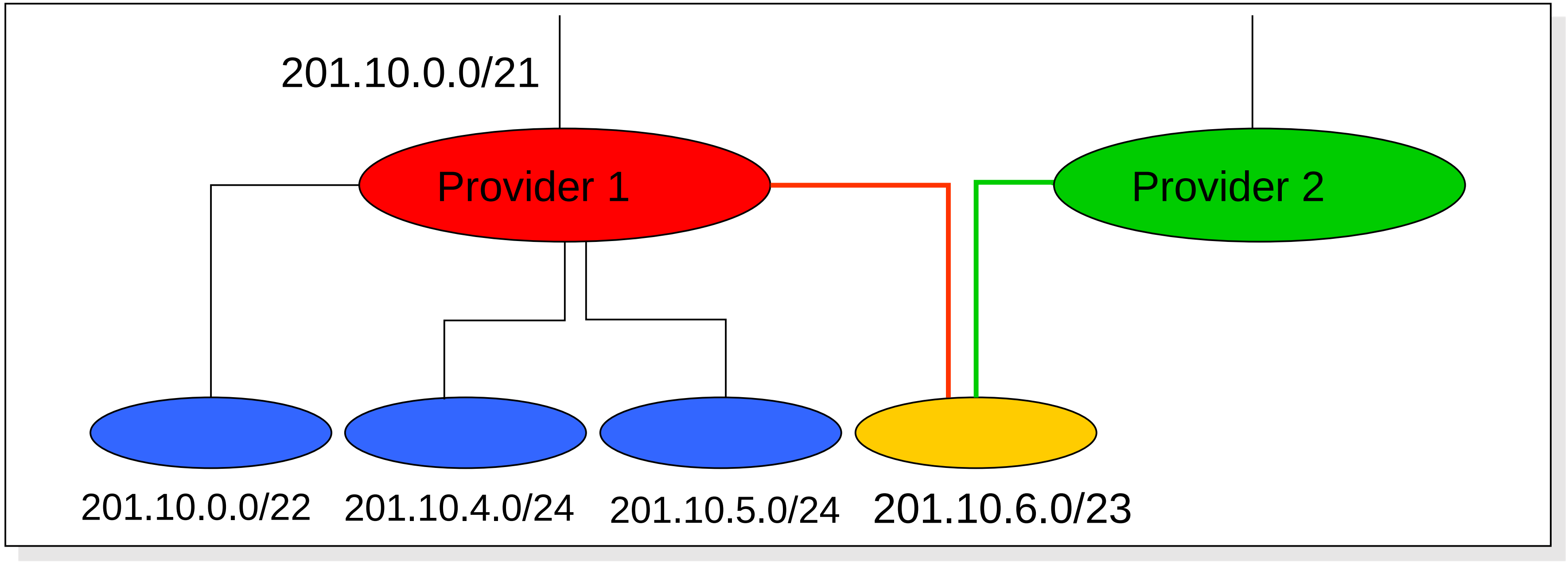
CIDR Makes Packet Forwarding Harder

- 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



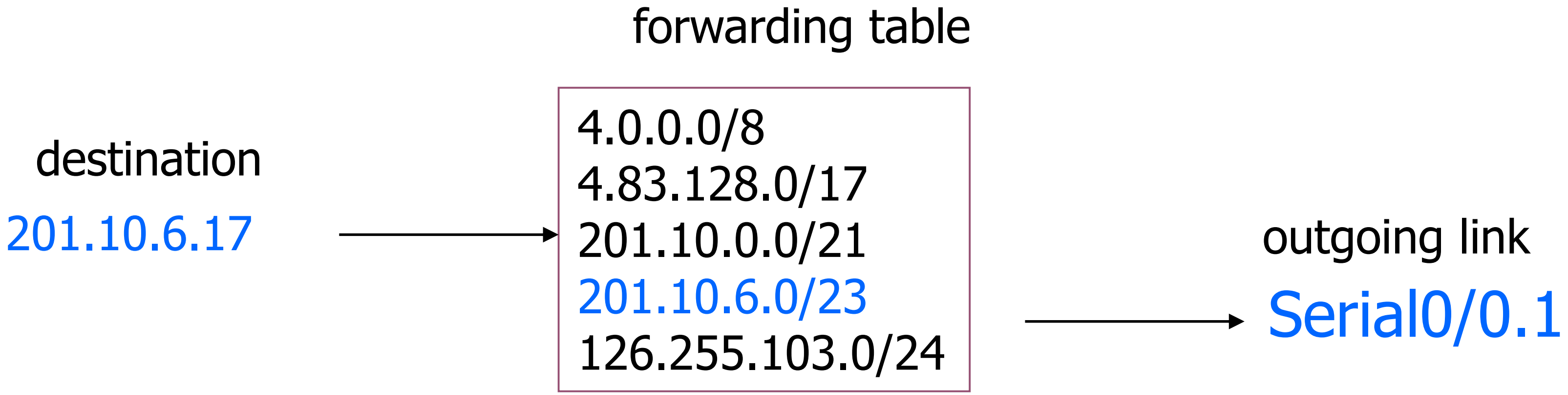
CIDR Makes Packet Forwarding Harder

- Forwarding table may have many matches
  - e. g. table entries for 201.10.0.0/21 and 201.10.6.0/23
  - The IP address 201.10.6.17 would match *both*!



Longest Prefix Match Forwarding

- Forwarding tables in IP routers map 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

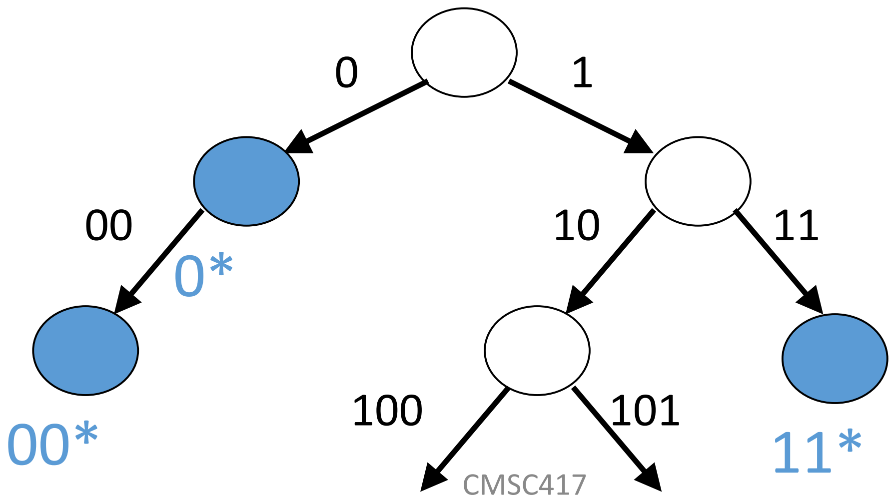


Simplest Algorithm is Too Slow

- 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 arrives
- 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 (time is proportional to number of bits in the address); 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—longest-prefix match was a major bottleneck

Where do Forwarding Tables Come From?

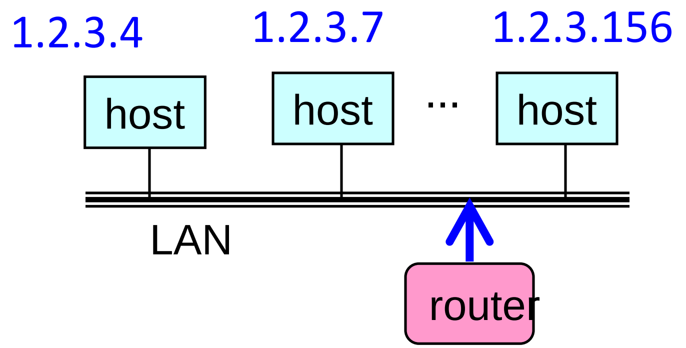
- Routers have forwarding tables which map prefix to outgoing link(s)
- Entries can be statically configured (e. g. “map 12.34.158.0/24 to Serial0/0.1”)
- But, this doesn’t adapt …
  - To failures
  - To new equipment
  - To the need to balance load
- That is where other technologies come in: routing protocols, DHCP, and ARP (later in course)

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 persistent, 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)



In Closing

- 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
- Future topics: protocols, DHCP, and ARP