# IP – The Internet Protocol

**Reading: Ch. 3.1, 4.1.1 – 4.1.4**

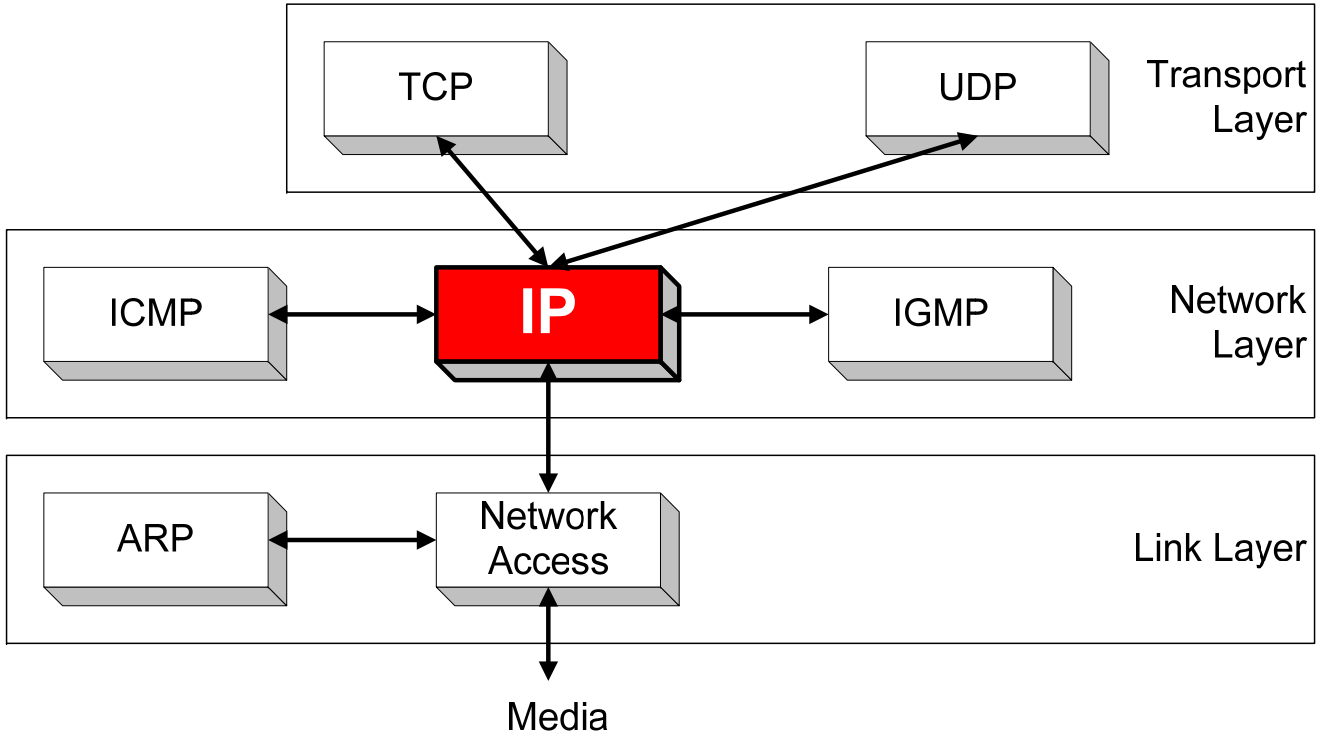
Slides by Rexford @ Princeton & Slides accompanying the Internet Lab Manual, slightly altered by M.D.

# Goals of Today's Lecture

- IP Datagram Format
- IP Addresses
  - IP prefixes for aggregation
- Address allocation
  - Classful addresses
  - Classless InterDomain Routing (CIDR)
- Packet forwarding
  - Forwarding tables
  - Longest-prefix match forwarding
  - Where forwarding tables come from

# Context

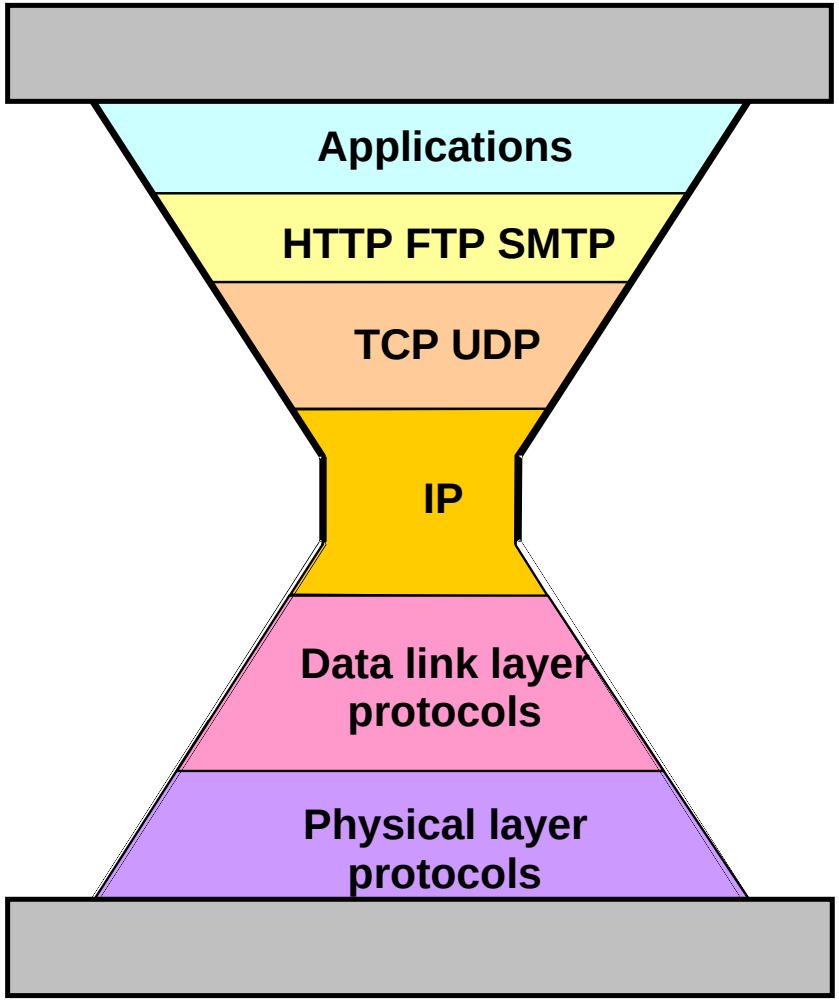
- IP (Internet Protocol) is a Network Layer Protocol.




- Current version is IPv4. It is specified in RFC 891.

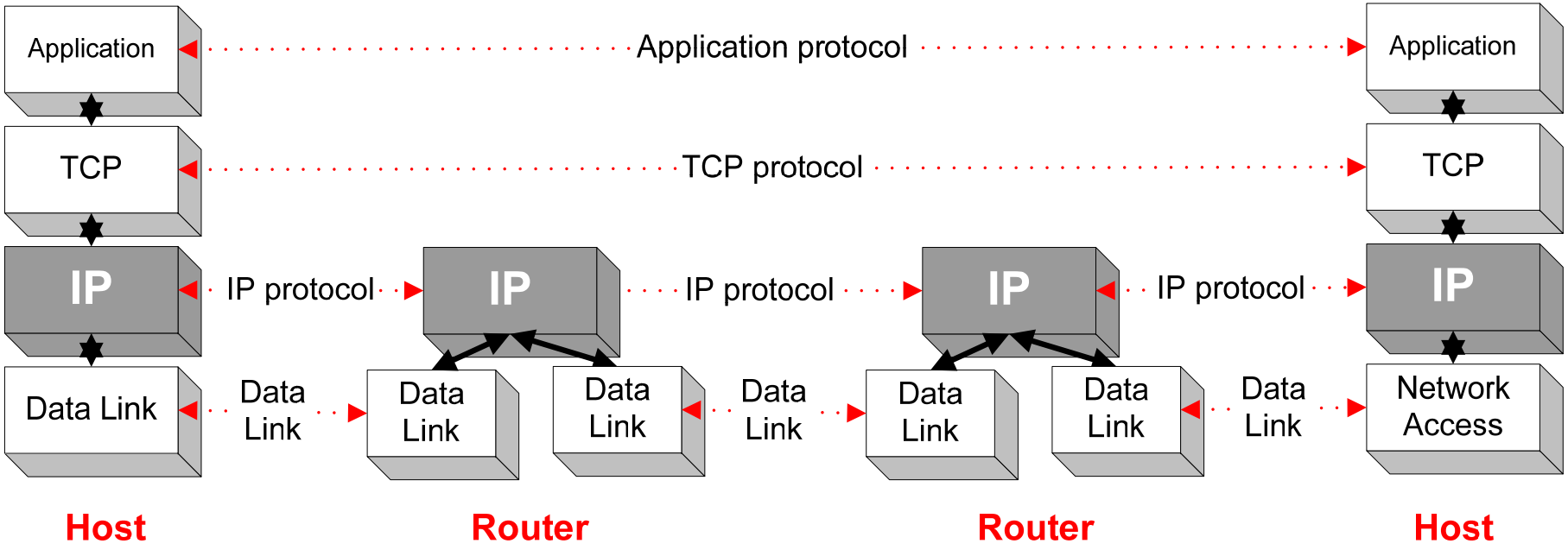
# IP: The Waist of the Hourglass

- IP is the waist of the hourglass of the Internet protocol architecture
- Multiple higher-layer protocols
- Multiple lower-layer protocols
- Only one protocol at the network layer.



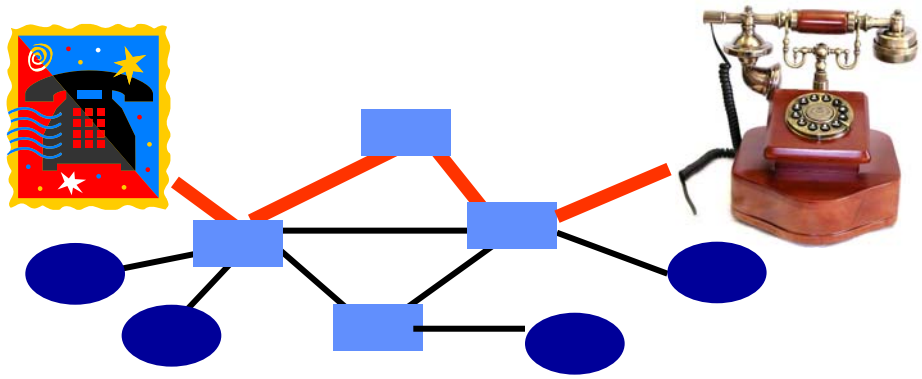
# Routers Know IP

- IP is the highest layer protocol which is implemented at both routers and hosts



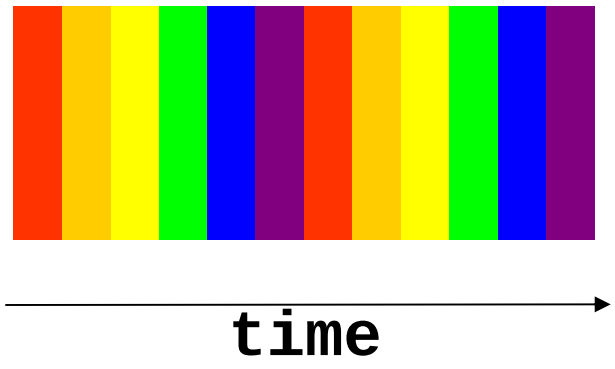
# Circuit Switching (e.g., Phone Network)

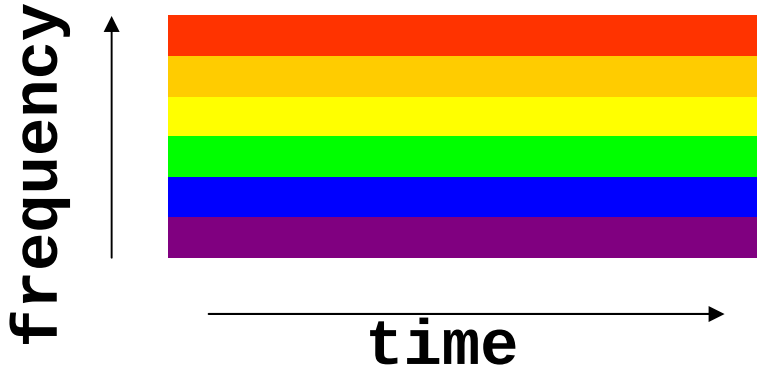
- Source establishes connection to destination
  - Node along the path store connection info
  - Nodes may reserve resources for the connection
- Source sends data over the connection
  - No destination address, since nodes know path
- Source tears down connection when done



# Circuit Switching: Multiplexing a Link

- Time-division
  - Each circuit allocated certain time slots
- Frequency-division
  - Each circuit allocated certain frequencies



# Advantages of Circuit Switching

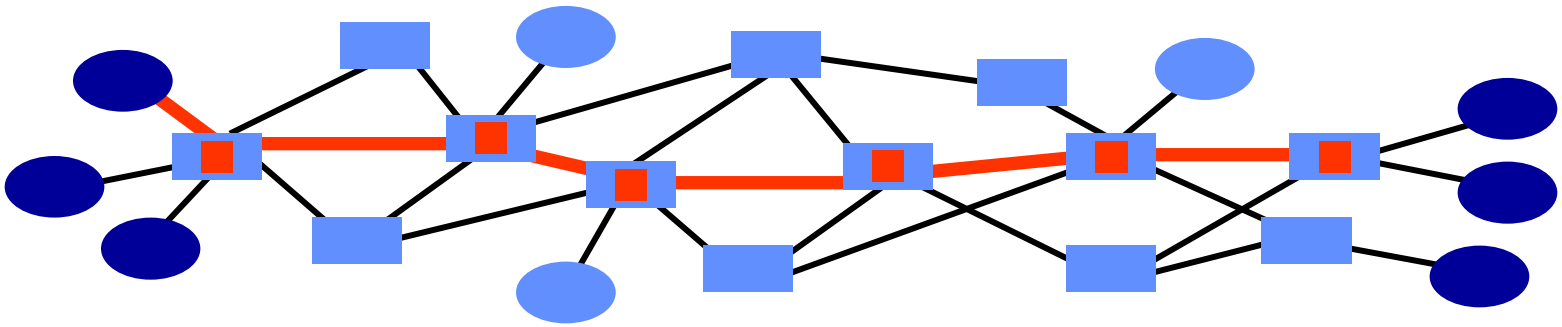
- Guaranteed bandwidth
  - Predictable communication performance
- Simple abstraction
  - Reliable communication channel between hosts
  - No worries about lost or out-of-order packets
- Simple forwarding
  - Forwarding based on time slot or frequency
  - No need to inspect a packet header
- Low per-packet overhead
  - Forwarding based on time slot or frequency
  - No IP (and TCP/UDP) header on each packet

# Disadvantages of Circuit Switching

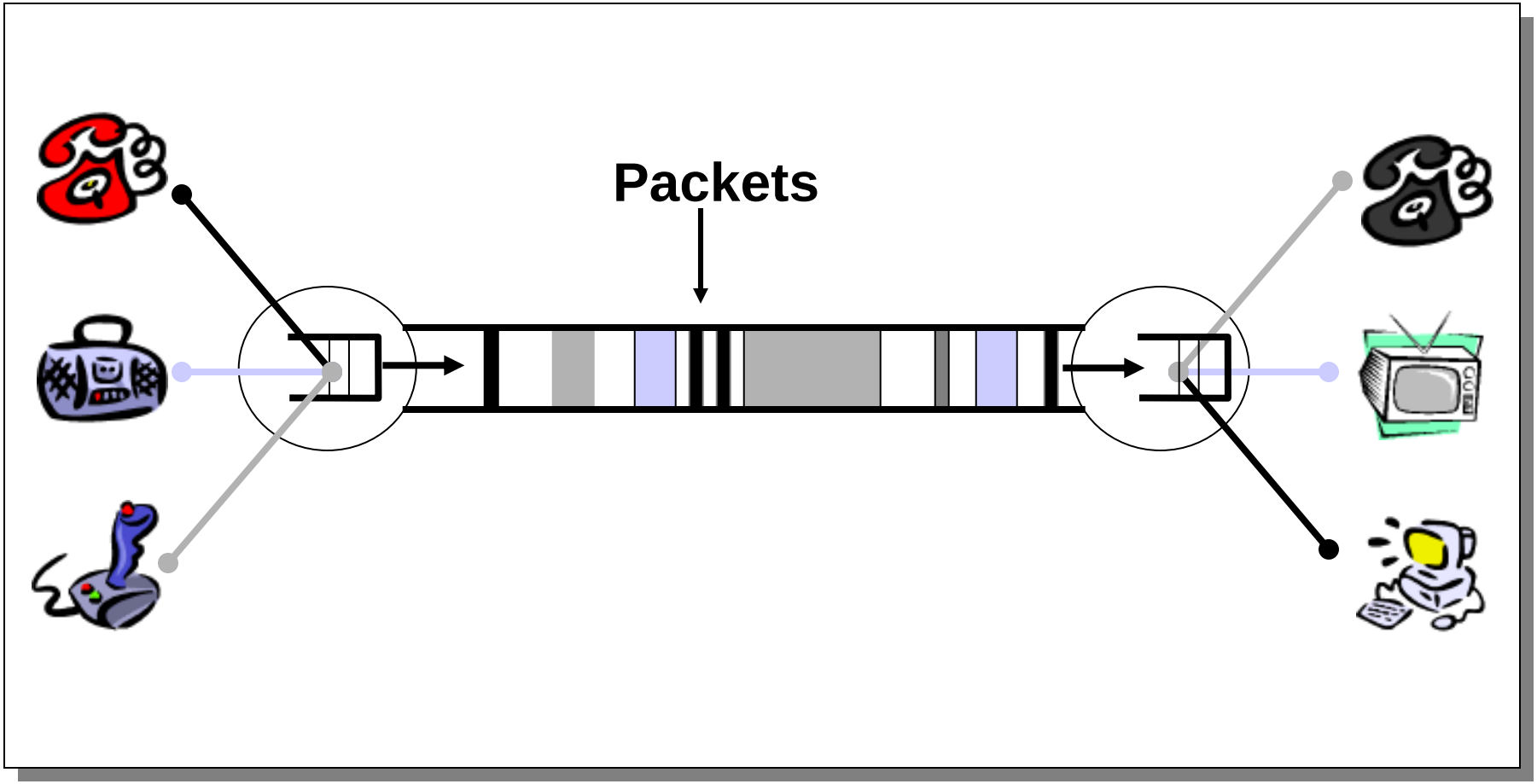
- Wasted bandwidth
  - Bursty traffic leads to idle connection during silent period
  - Unable to achieve gains from statistical multiplexing
- Blocked connections
  - Connection refused when resources are not sufficient
  - Unable to offer “okay” service to everybody
- Connection set-up delay
  - No communication until the connection is set up
  - Unable to avoid extra latency for small data transfers
- Network state
  - Network nodes must store per-connection information
  - Unable to avoid per-connection storage and state

# 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

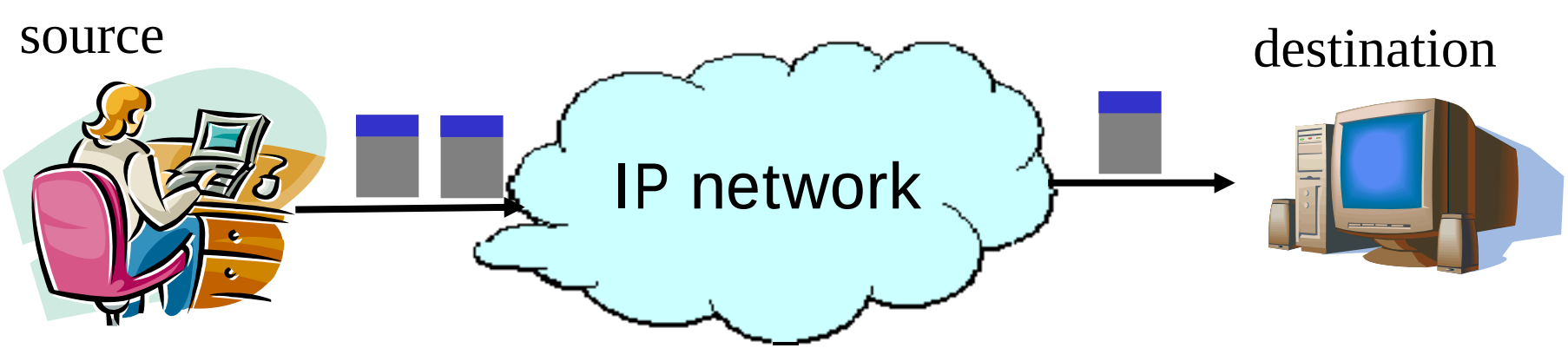


# Packet Switching: Statistical Multiplexing



# IP Service: Best-Effort Packet Delivery

- Packet switching
  - Divide messages into a sequence of packets
  - Headers with source and destination address
- Best-effort delivery
  - Packets may be lost
  - Packets may be corrupted
  - Packets may be delivered out of order



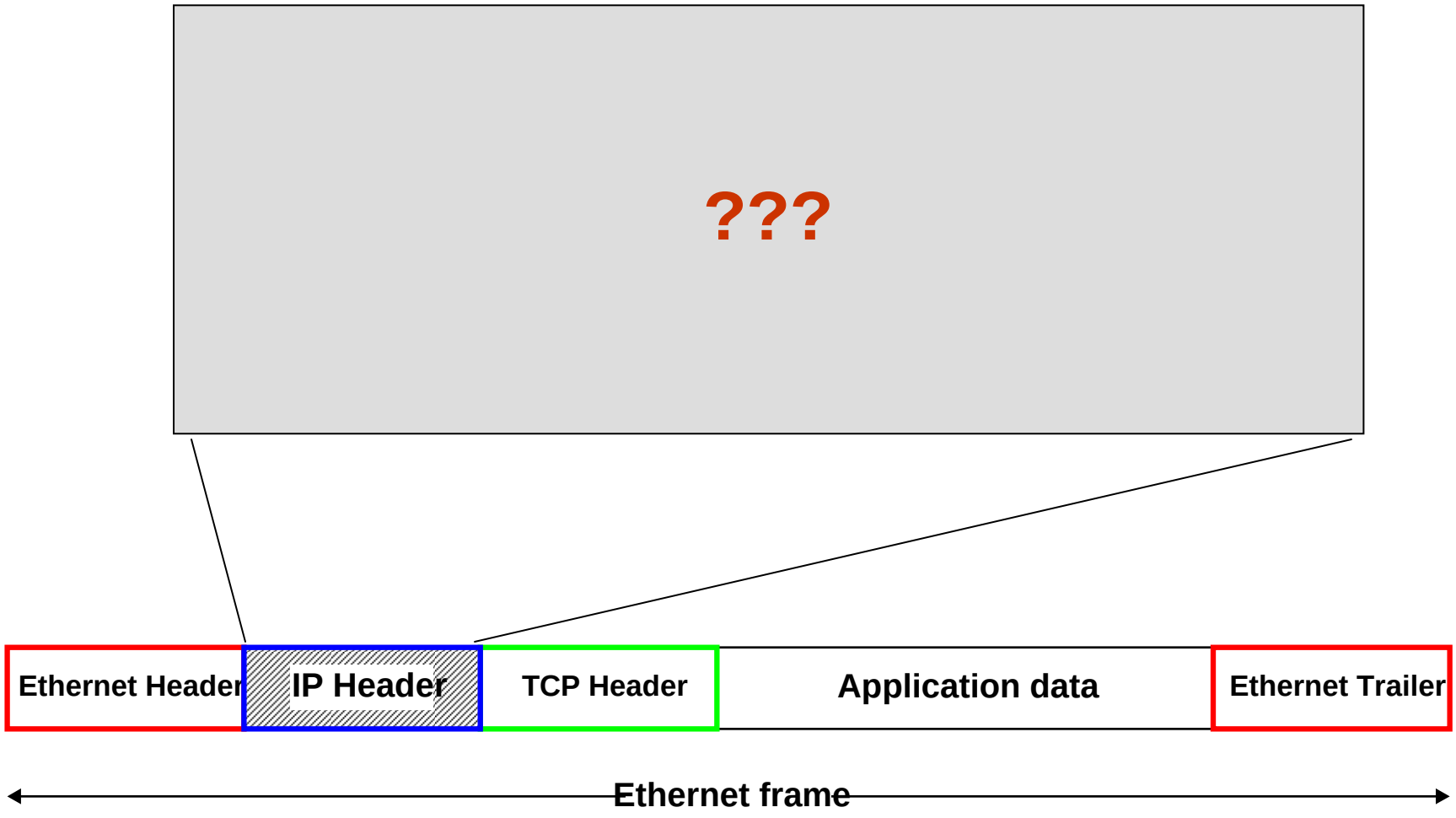
# IP Service Model: Why Best-Effort?

- IP means never having to say you're sorry…
  - Don't need to reserve bandwidth and memory
  - Don't need to do error detection & correction
  - Don't need to remember from one packet to the next
- Easier to survive failures
  - Transient disruptions are okay during failover
- … but, applications *do* want efficient, accurate transfer of data in order, in a timely fashion

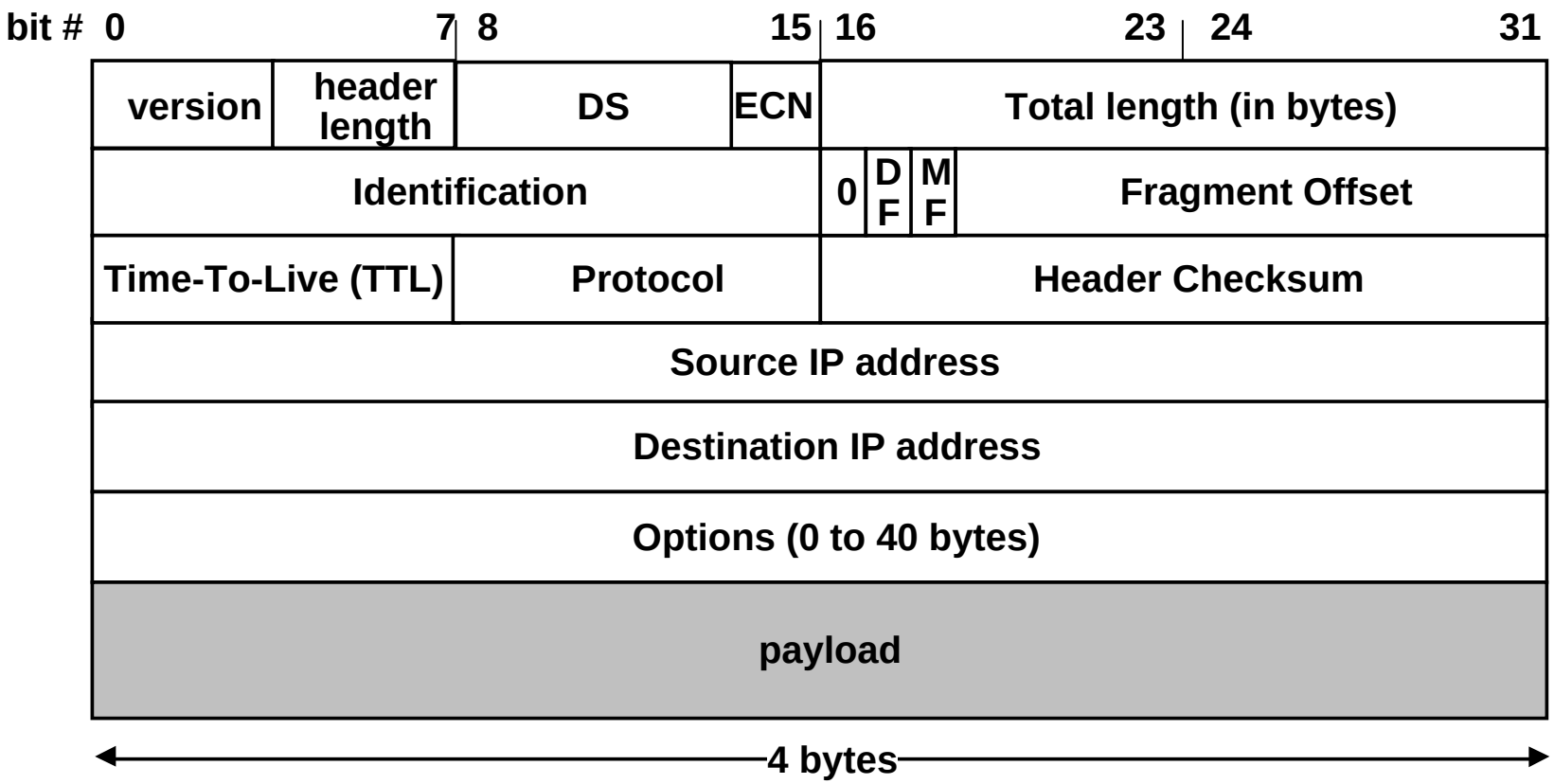
# IP Service: Best-Effort is Enough

- No error detection or correction
  - Higher-level protocol can provide error checking
- Successive packets may not follow the same path
  - Not a problem as long as packets reach the destination
- Packets can be delivered out-of-order
  - Receiver can put packets back in order (if necessary)
- Packets may be lost or arbitrarily delayed
  - Sender can send the packets again (if desired)
- No network congestion control (beyond “drop”)
  - Sender can slow down in response to loss or delay

# IP Datagram - Context



# IP Datagram Format

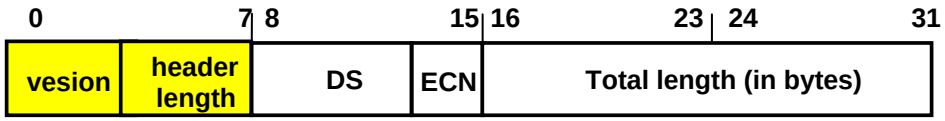



- 20 bytes ≤ Header Size < $2^4$ x 4 bytes = 60 bytes
- 20 bytes ≤ Total Length < $2^{16}$ bytes = 65536 bytes

# IP Datagram Format

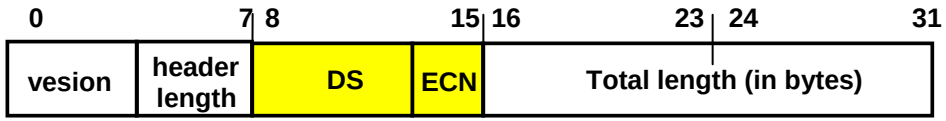
- Question: In which order are the bytes of an IP datagram transmitted?
- Answer:
  - Transmission is row by row
  - For each row:
    - » 1. First transmit bits 0-7
    - » 2. Then transmit bits 8-15
    - » 3. Then transmit bits 16-23
    - » 4. Then transmit bits 24-31
- This is called network byte order or big endian byte ordering.

# IP Header: Version, Length



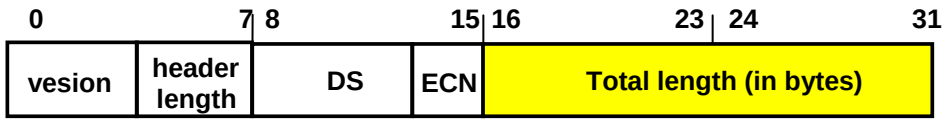

- Version number (4 bits)
  - Indicates the version of the IP protocol
  - Necessary to know what other fields to expect
  - Typically “4” (for IPv4), and sometimes “6” (for IPv6)
- Header length (4 bits)
  - Number of 32-bit words in the header
  - Typically “5” (for a 20-byte IPv4 header)
  - Can be more when “IP options” are used

# IP Header: DS/ECN



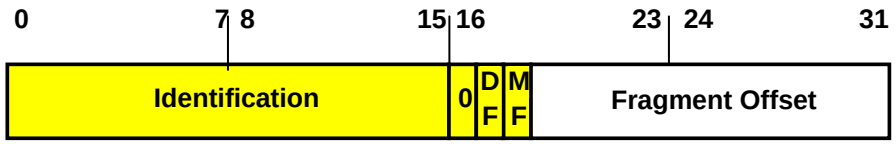

- DS/ECN field (1 byte)
  - This field was previously called as Type-of-Service (TOS) field. The role of this field has been re-defined, but is "backwards compatible" to TOS interpretation
- Differentiated Service (DS) (6 bits):
  - Used to specify service level (currently not supported in the Internet)
- Explicit Congestion Notification (ECN) (2 bits):
  - New feedback mechanism used by TCP

# IP Header: Total Length




- Total length (16 bits)
  - Number of bytes in the datagram
  - Maximum size is 63,535 bytes ($2^{16}$ -1)
  - … though underlying links may impose harder limits

# Fields of the IP Header

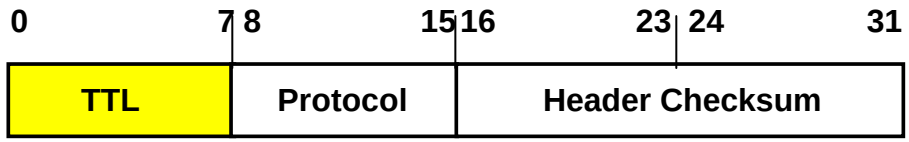



- Identification (16 bits):
  - Unique identification of a datagram from a host. Incremented whenever a datagram is transmitted.

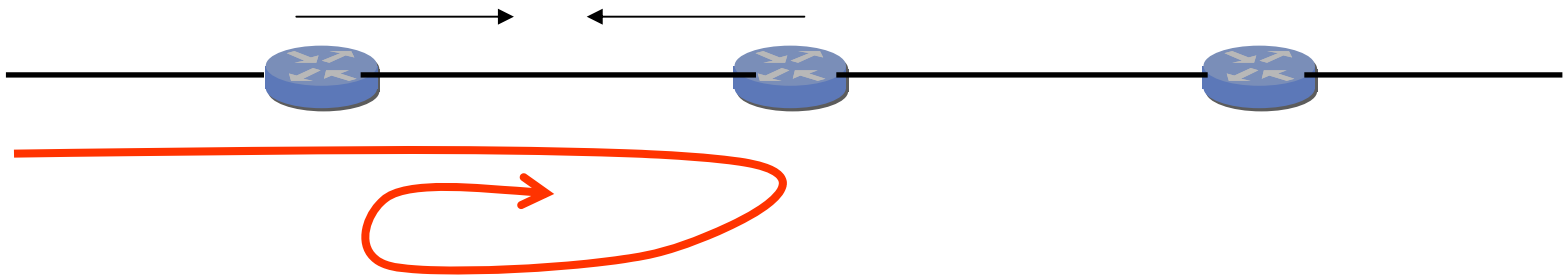
- Flags (3 bits):
  - First bit always set to 0
  - DF bit (Do not fragment)
  - MF bit (More fragments)

  Will be explained later→ Fragmentation

# Fields of the IP Header




- Time To Live (TTL) (1 byte):
  - Specifies longest paths before datagram is dropped
  - Role of TTL field: Ensure that packet is eventually dropped when a routing loop occurs

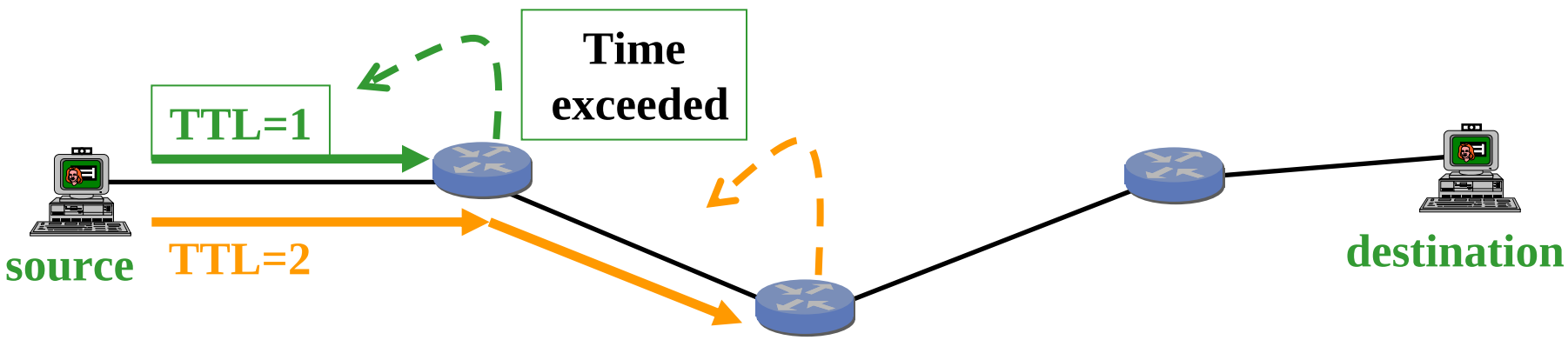


Used as follows:

- Sender sets the value (e.g., 64)
- Each router decrements the value by 1
- When the value reaches 0, the datagram is dropped

# IP Header: Use of TTL in Traceroute

- Time-To-Live field in IP packet header
  - Source sends a packet with a TTL of *n*
  - Each router along the path decrements the TTL
  - "TTL exceeded" sent when TTL reaches *0*
- Traceroute tool exploits this TTL behavior




**Send packets with TTL=1, 2, … and record source of "time exceeded" message**

# Example Traceroute: Villanova to google

Hop number, IP address, DNS name

| | | |
|---|---|---|
| 1 | 153.104.200.1 | ? |
| 2 | 153.104.0.1 | ? |
| 3 | 153.104.0.254 | ? |
| 4 | 4.78.146.145 | ge-5-1.hsa2.Philadelphia1.Level3.net |
| 5 | 64.159.0.149 | ge-6-0-1.mp1.Philadelphia1.Level3.net |
| 6 | 4.68.128.206 | as-3-0.bbr2.Washington1.Level3.net |
| 7 | 4.68.17.144 | ae-3-89.edge1.Washington1.Level3.net |
| 8 | 4.79.231.6 | GOOGLE-INC.edge1.Washington1.Level3.net |
| 9 | 64.233.175.169 | ? |
| 10 | 216.239.49.149 | ? |
| 11 | 64.233.169.99 | yo-in-f99.google.com |

No name resolution

# Try Running Traceroute Yourself

- On UNIX machine
  - traceroute
  - E.g., “traceroute www.google.com” or “traceroute 64.233.169.99”
- On Windows machine
  - tracert
  - E.g., “tracert www.cnn.com” or “tracert 64.233.169.99”

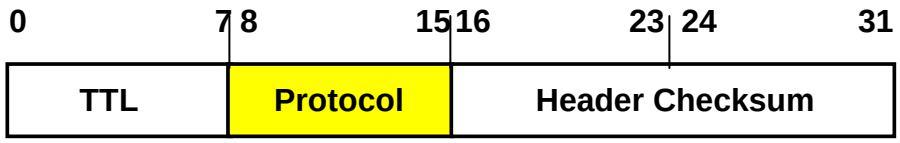
- Common uses of traceroute
  - Discover the topology of the Internet
  - Debug performance and reachability problems

# Try Running Traceroute Yourself

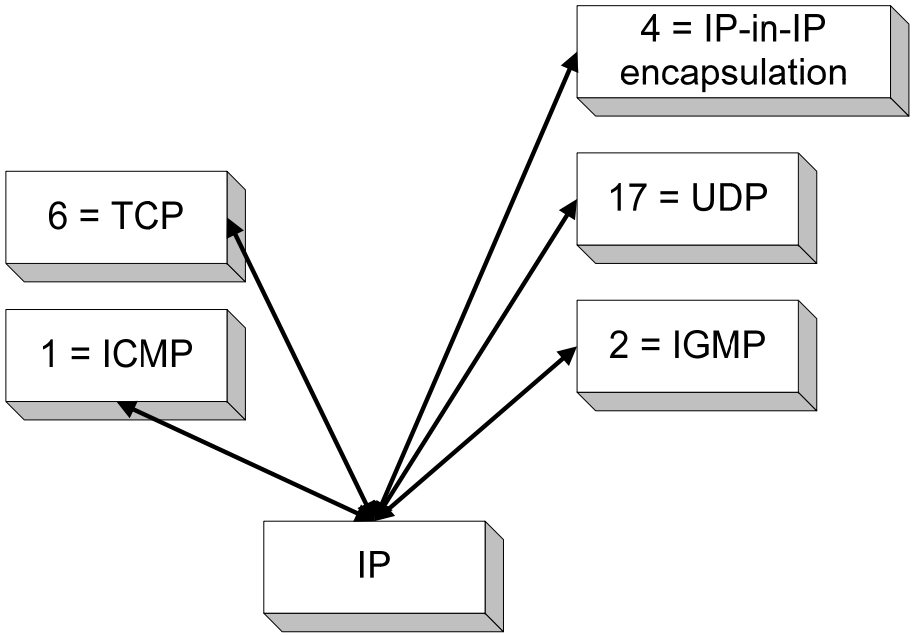
- Check out Visual Route -- it is a great tool that puts together pings & traceroutes and displays the data in a graphical and very intuitive way.
  - Go to http://visualroute.visualware.com/
  - Type in any foreign domain name, e.g., any one of:

    www.gcal.ac.uk
    www.sfi.ie
    mousse.ens.fr
    iprimus.com.au

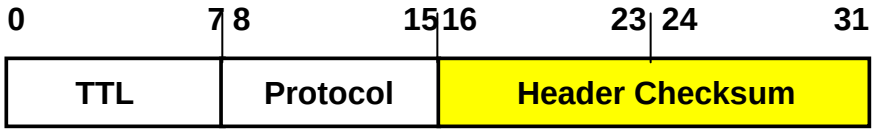
# Fields of the IP Header




- Protocol (1 byte):
  - Indicates what kind of header to expect next.
  - Used for demultiplexing to higher layers.



# Fields of the IP Header




- Header checksum (2 bytes):
  - 16-bit long checksum computed for the IP header
  - If any bits of the header are corrupted in transit
  - … the checksum won't match at receiving host
  - Receiving host discards corrupted packets
    - Sending host will retransmit the packet, if needed

```
   134
+  212
------
=  346
```

```
   134
+  216
------
=  350
```

**Mismatch!**

# Fields of the IP Header

**Options (0 to 40 bytes)**

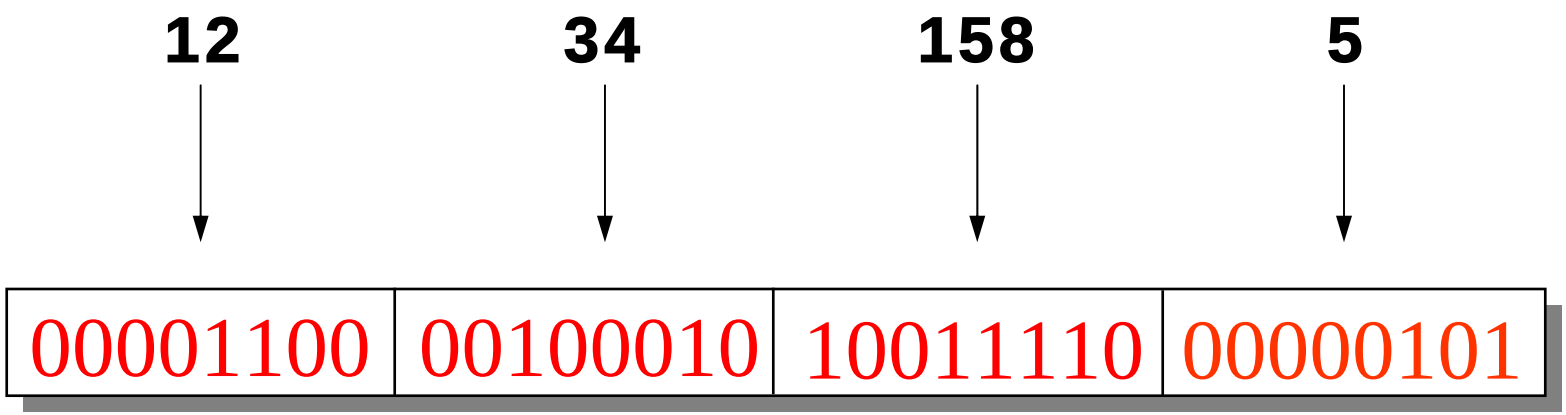
- Options:
  - Record Route: each router that processes the packet adds its IP address to the header.
  - Timestamp: each router that processes the packet adds its IP address and time to the header.
  - (loose) Source Routing: specifies a list of routers that must be traversed.
  - (strict) Source Routing: specifies a list of the only routers that can be traversed.

# IP Header: To and From Addresses

- 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

# IP Address (IPv4)

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



# Source Address: What if 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**
    - **… to overload 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 use someone else's source address**
  - **Also, an attack against the spoofed host**
    - **Spoofed host is wrongly blamed**
    - **Spoofed host may receive return traffic from the receiver**

# Maximum Transmission Unit

Slides by Rexford @ Princeton & Slides accompanying the Internet Lab Manual, slightly altered by M.D.

# Maximum Transmission Unit

- Maximum size of IP datagram is 65535, but the data link layer protocol generally imposes a limit that is much smaller

- For example:
  - Ethernet frames have a maximum payload of 1500 bytes→ IP datagrams encapsulated in Ethernet frame cannot be longer than 1500 bytes

- The limit on the maximum IP datagram size, imposed by the data link protocol is called **maximum transmission unit (MTU)**

# Maximum Transmission Unit

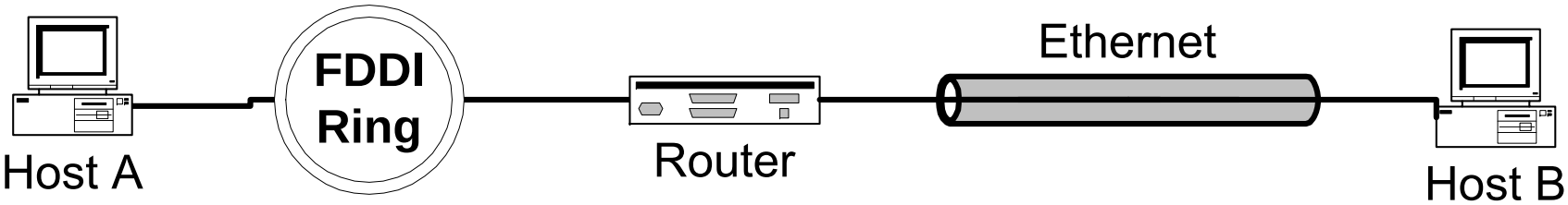
- MTUs for various data link layers:

| | | | |
|---|---|---|---|
| Ethernet: | 1500 | FDDI: | 4352 |
| 802.3: | 1492 | ATM AAL5: | 9180 |
| 802.5: | 4464 | PPP: | 296 |

# IP Fragmentation

- What if the IP datagram exceeds the MTU?
  - IP datagram is fragmented into smaller units.

- What if route passes networks with different MTUs?

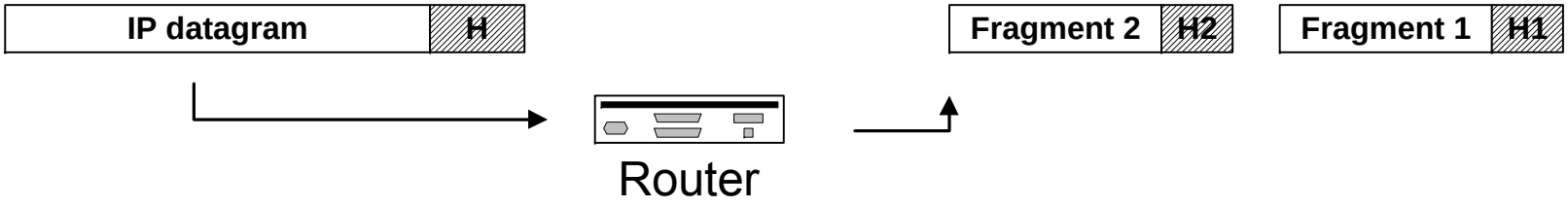



MTUs: FDDI: 4352 Ethernet: 1500

- Fragmentation:
  - IP router splits the datagram into several datagram
  - Fragments are reassembled at receiver

# Where is Fragmentation Done?

- Fragmentation can be done at the sender or at intermediate routers
- The same datagram can be fragmented several times.
- Reassembly of original datagram is only done at destination hosts !!



# What is Involved in Fragmentation?

The following fields in the IP header:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| version | header length | DS | ECN | total length (in bytes) | | | |
| Identification | | | | 0 | DF | MF | Fragment offset |
| time-to-live (TTL) | | protocol | | header checksum | | | |

Identification — When a datagram is fragmented, the identification is the same in all fragments

Flags

DF bit is set: Datagram cannot be fragmented and must be discarded if MTU is too small

MF bit set: This datagram is part of a fragment and an additional fragment follows this one

# What is Involved in Fragmentation?

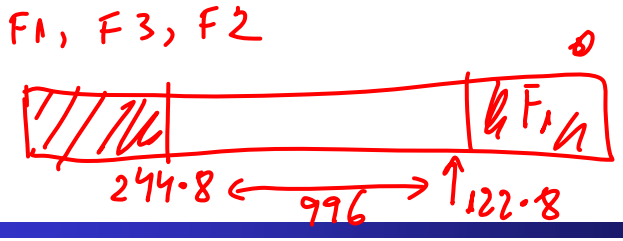
The following fields in the IP header:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| version | header length | DS | ECN | total length (in bytes) | | | |
| Identification | | | | 0 | DF | MF | Fragment offset |
| time-to-live (TTL) | | protocol | | header checksum | | | |

Fragment offset — Offset of the payload of the current fragment in the original datagram (measured in units of 8-bytes)

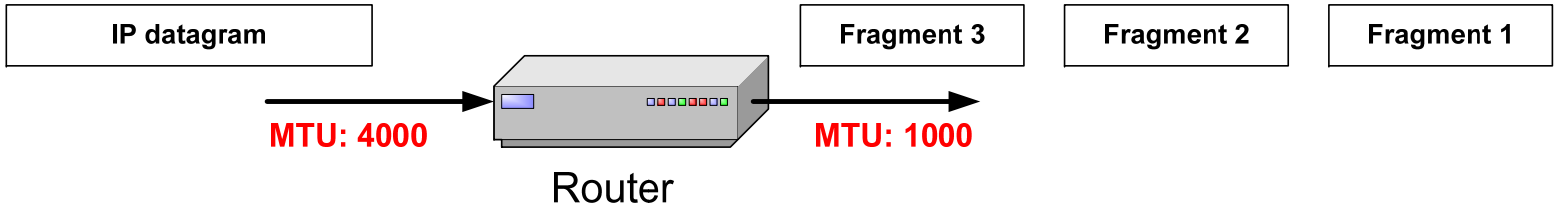
Total length — Total length of the current fragment

# Fragmentation Example




- A datagram with size 2400 bytes must be fragmented according to an MTU limit of 1000 bytes.

| | | | |
|---|---|---|---|
| Header length: 20 | Header length: 20 | Header length: 20 | Header length: 20 |
| Total length: 2400 | Total length: 448 | Total length: 996 | Total length: 996 |
| Identification: 0xa428 | Identification: 0xa428 | Identification: 0xa428 | Identification: 0xa428 |
| DF flag: 0 | DF flag: 0 | DF flag: 0 | DF flag: 0 |
| MF flag: 0 | MF flag: 0 | MF flag: 1 | MF flag: 1 |
| Fragment offset: 0 | Fragment offset: 244 | Fragment offset: 122 | fragment offset: 0 |

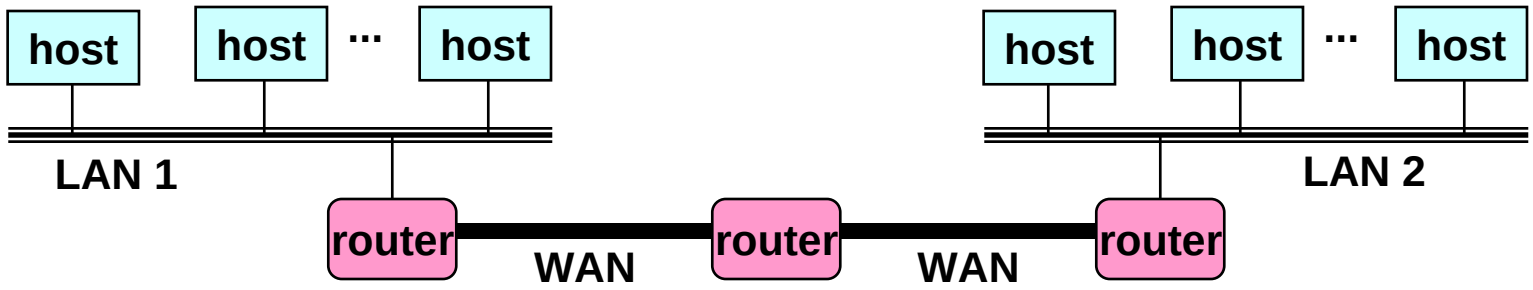


# Network Addresses

Slides by Rexford @ Princeton & Slides accompanying the Internet Lab Manual, slightly altered by M.D.

# Grouping Related Hosts

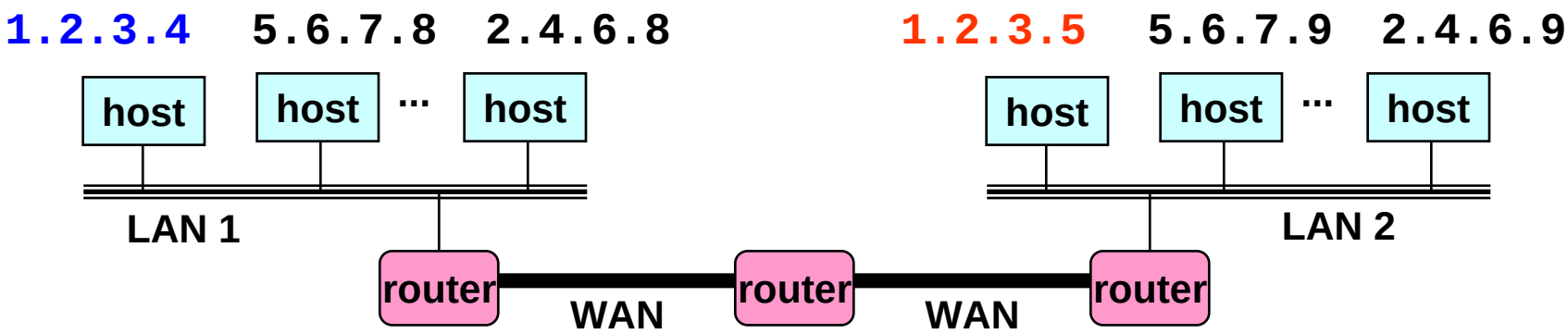
- The Internet is an "inter-network"
  - Used to connect *networks* together, not *hosts*
  - Needs a way to address a network (i.e., group of hosts)




LAN = Local Area Network
WAN = Wide Area Network

# Scalability Challenge

- Suppose hosts had arbitrary addresses
  - Then every router would need a lot of information
  - …to know how to direct packets toward *every* host




| | |
|---|---|
| 1.2.3.4 | ← |
| 1.2.3.5 | → |
| ⋮ | |

**Forwarding table**

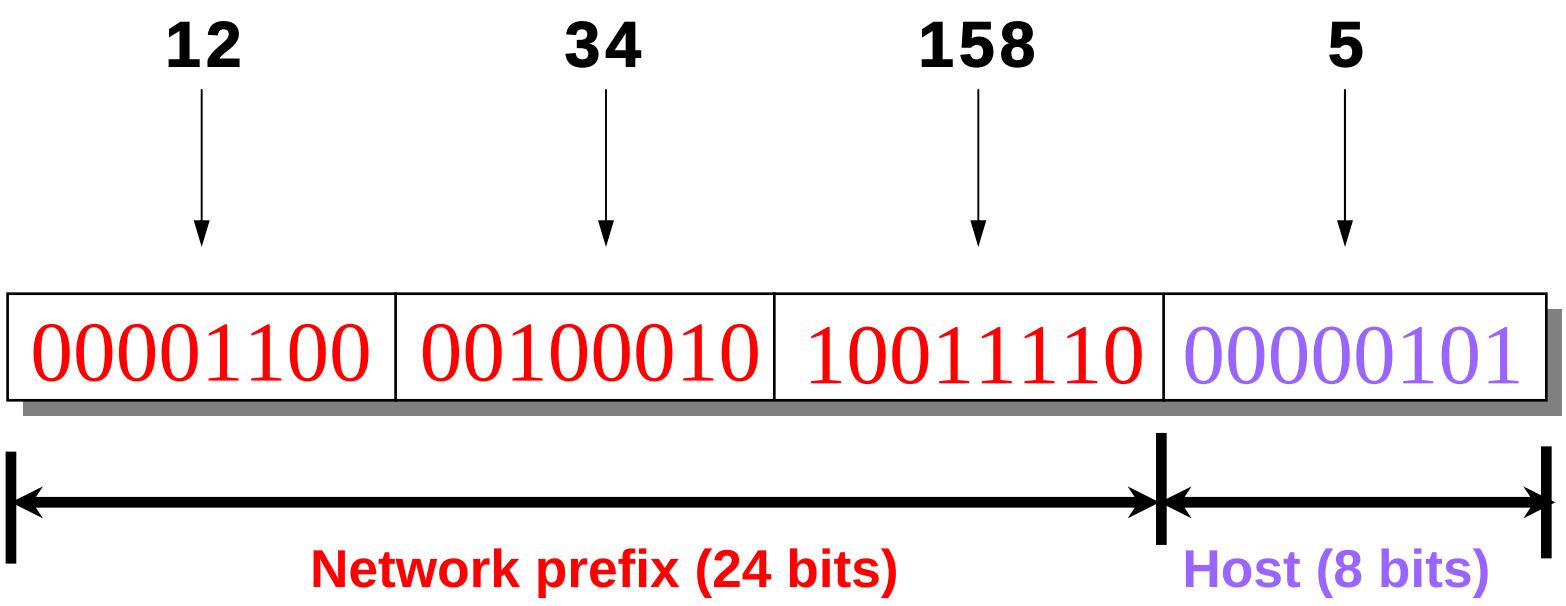
# Standard CS Trick

Have a scalability problem?

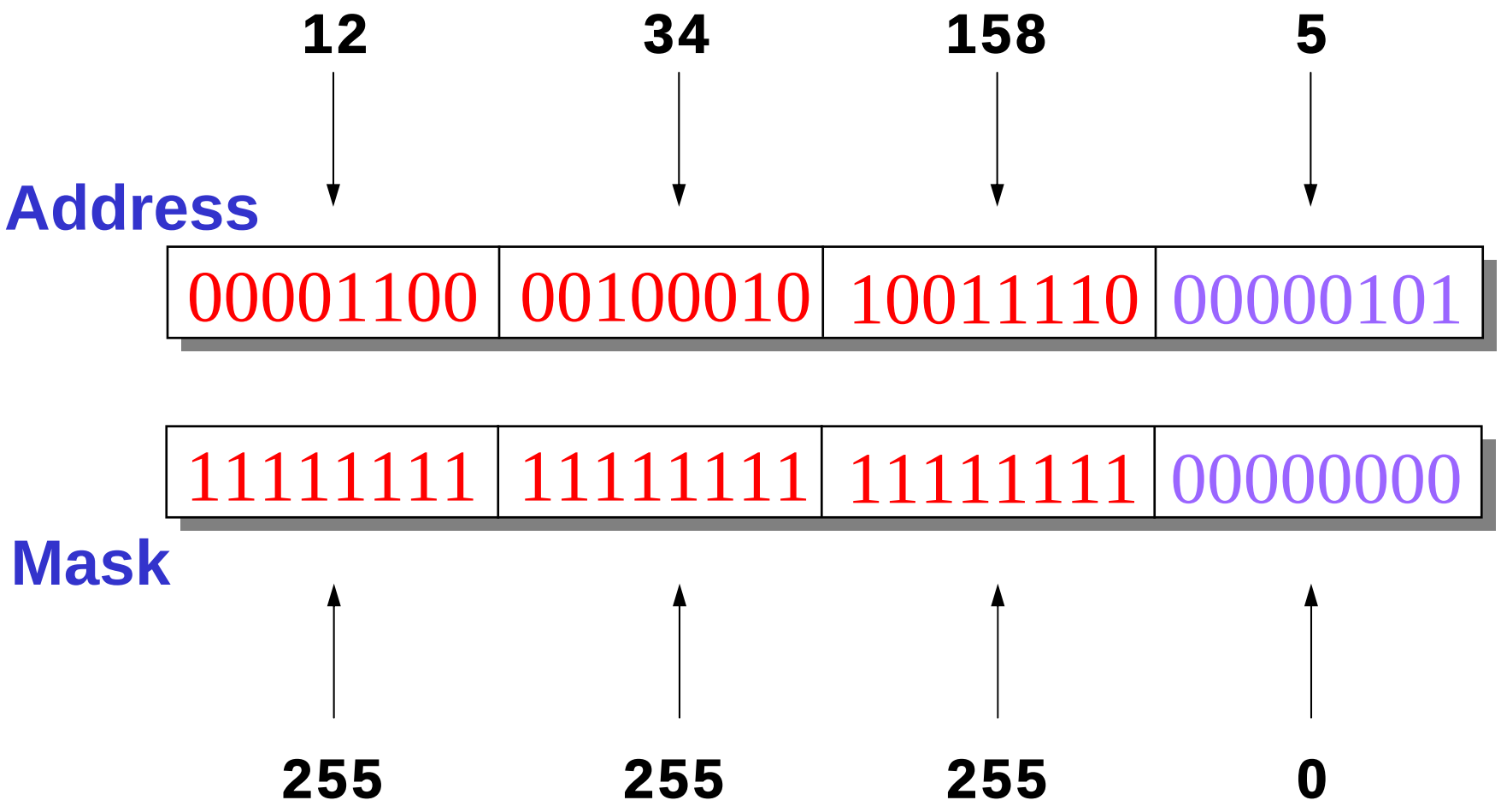
Introduce hierarchy…

# 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

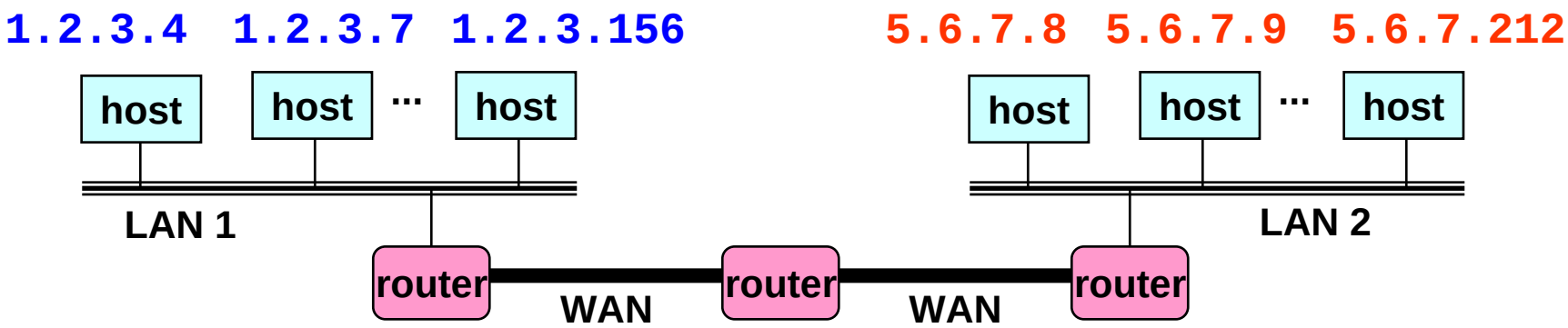


# IP Address and a 24-bit Subnet Mask



# 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

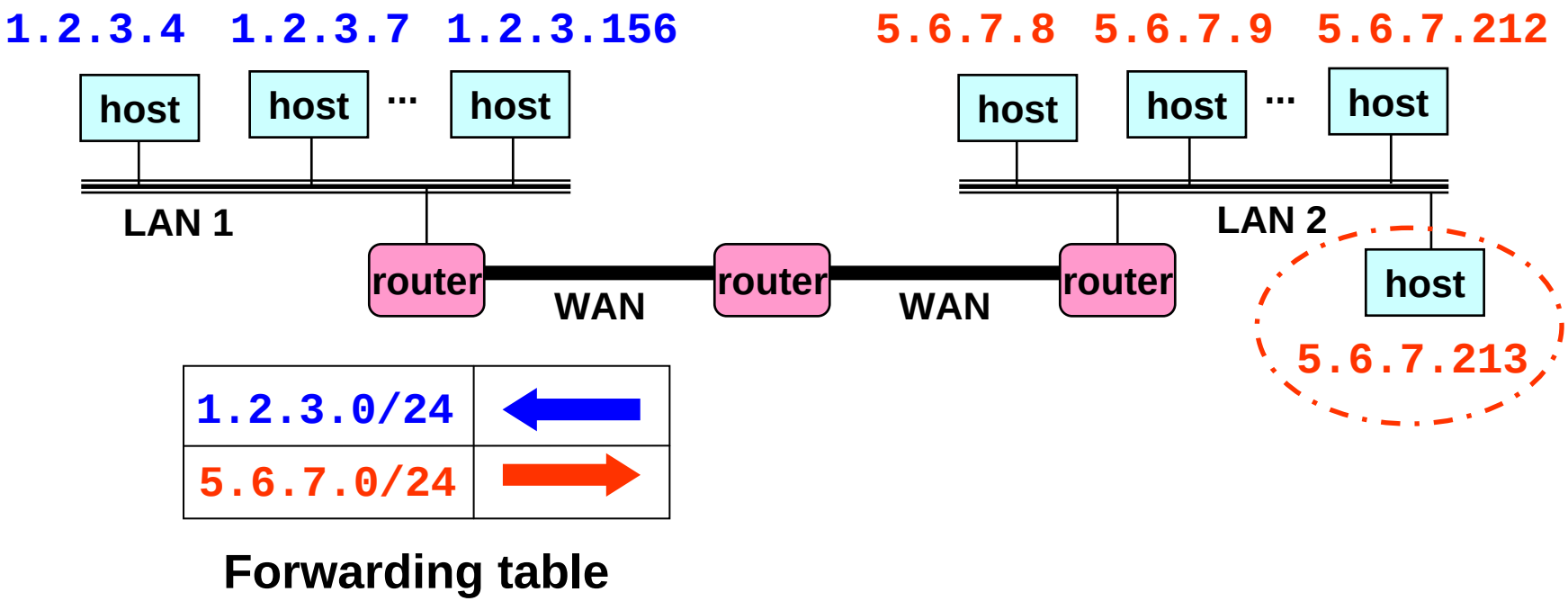



| | |
|---|---|
| 1.2.3.0/24 | ← |
| 5.6.7.0/24 | → |

**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-table entry



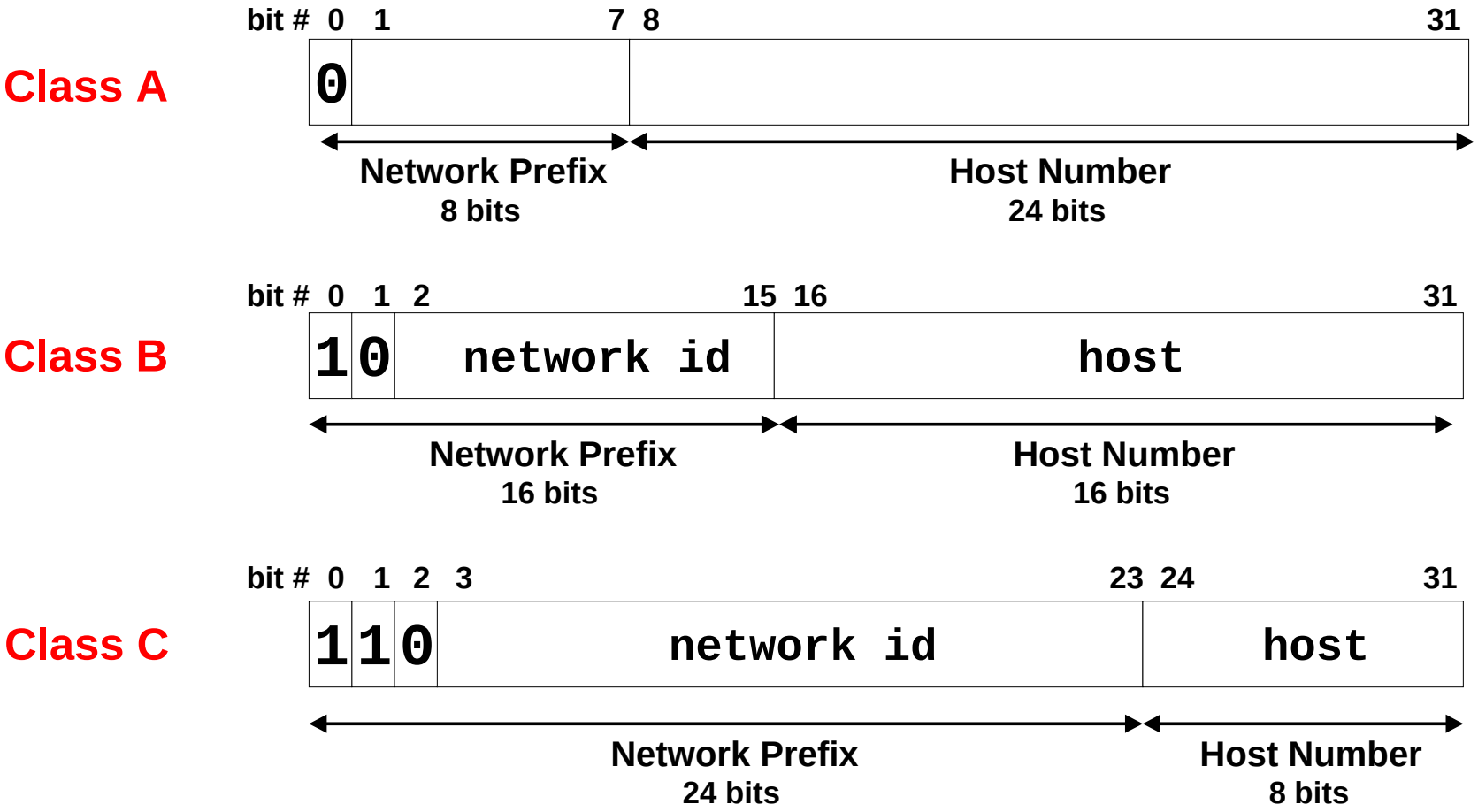
# Subnetting

Slides by Rexford @ Princeton & Slides accompanying the Internet Lab Manual, slightly altered by M.D.

# Classful Addressing

- In the olden days, only fixed allocation sizes
  - Class A: 0*
    - Very large /8 blocks (e.g., MIT has 18.0.0.0/8)
  - Class B: 10*
    - Large /16 blocks (e.g,. Princeton has 128.112.0.0/16)
  - Class C: 110*
    - Small /24 blocks (e.g., AT&T Labs has 192.20.225.0/24)
  - Class D: 1110*
    - Multicast groups
  - Class E: 11110*
    - Reserved for future use
- This is why folks use dotted-quad notation!

# Old Way: Internet Address Classes



# Old Way: Internet Address Classes

**Class D**

| bit # | 0 | 1 | 2 | 3 | 4 … 31 |
|---|---|---|---|---|---|
| | 1 | 1 | 1 | 0 | multicast group id |

**Class E**

| bit # | 0 | 1 | 2 | 3 | 4 | 5 … 31 |
|---|---|---|---|---|---|---|
| | 1 | 1 | 1 | 1 | 0 | (reserved for future use) |

# Problems with Classful IP Addresses

- The original classful address scheme had a number of problems

  Problem 1. Too few network addresses for large networks

  – Class A and Class B addresses are gone

  Problem 2. Two-layer hierarchy is not appropriate for large networks with Class A and Class B addresses.

  – **Fix #1: Subnetting**

# Problems with Classful IP Addresses

Problem 3. Inflexible.

- Assume a company requires 2,000 addresses
- Class A and B addresses are overkill
- Class C address is insufficient (requires 8 Class C addresses)

- **Fix #2: Classless Interdomain Routing (CIDR)**

# Problems with Classful IP Addresses

Problem 4: Exploding Routing Tables:

- Routing on the backbone Internet needs to have an entry for each network address. In 1993, the size of the routing tables started to outgrow the capacity of routers.

- **Fix #2: Classless Interdomain Routing (CIDR)**

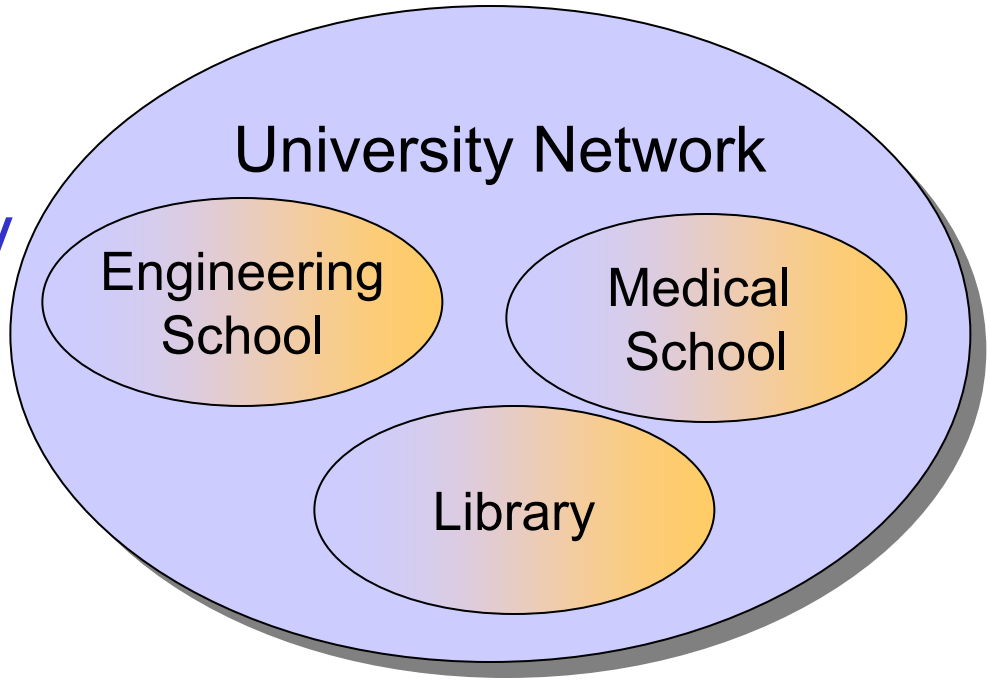
# Problems with Classful IP Addresses

Problem 5. The Internet is going to outgrow the 32-bit addresses

– **Fix #3: IP Version 6**

# Subnetting

- Problem: Organizations have multiple networks which are independently managed

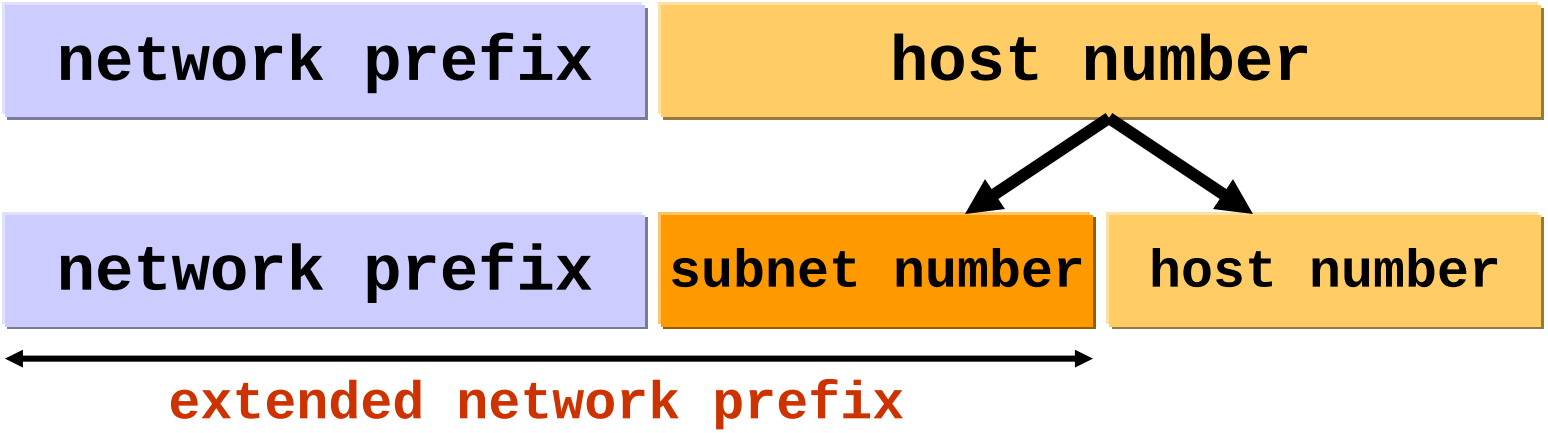



Solution: Add another level of hierarchy to the IP addressing structure

**Subnetting**

# Basic Idea of Subnetting

- Split the host number portion of an IP address into a **subnet number** and a (smaller) **host number**.
- Result is a 3-layer hierarchy




- Subnets can be freely assigned within the organization
- Internally, subnets are treated as separate networks
- Subnet structure is not visible outside the organization

# Subnet Masks

- Use two 32-bit numbers to represent a network:

  **Network number = IP address + Mask**

- Example: IP Address = 12.4.0.0, Mask = 255.254.0.0

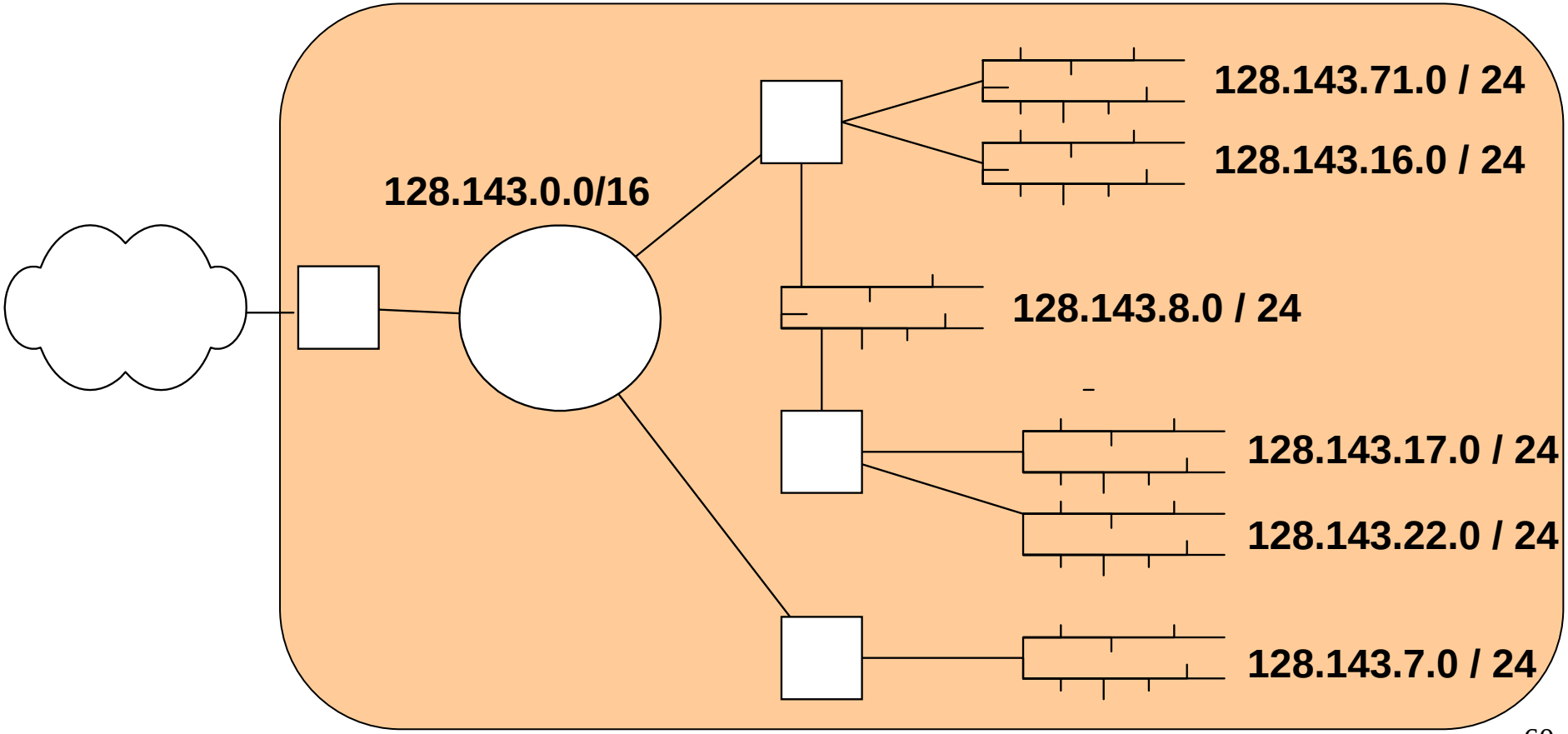
| | | | | |
|---|---|---|---|---|
| **Address** | 00001100 | 00000100 | 00000000 | 00000000 |
| **Mask** | 11111111 | 11111110 | 00000000 | 00000000 |

**Network Prefix** | **for hosts**

**Written as 12.4.0.0/15**

# Typical Organization Network

- Each layer-2 network (Ethernet segment, FDDI segment) is allocated a subnet address.



# CIDR: Classless Inter-Domain Routing

- Goals:
  - Restructure IP address assignments to increase efficiency
  - Hierarchical routing aggregation to minimize route table entries
- CIDR (Classless Interdomain routing) abandons the notion of classes.
- Key Concept: The length of the network id (prefix) in the IP addresses is kept arbitrary
- Consequence: Routers advertise the IP address and the length of the prefix

# CIDR Example

- CIDR notation of a network address:

  **192.0.2.0/18**

  – "18" says that the first 18 bits are the network part of the address (and 14 bits are available for specific host addresses)

- The network part is called the prefix

- Assume that a site requires a network address with 1000 addresses

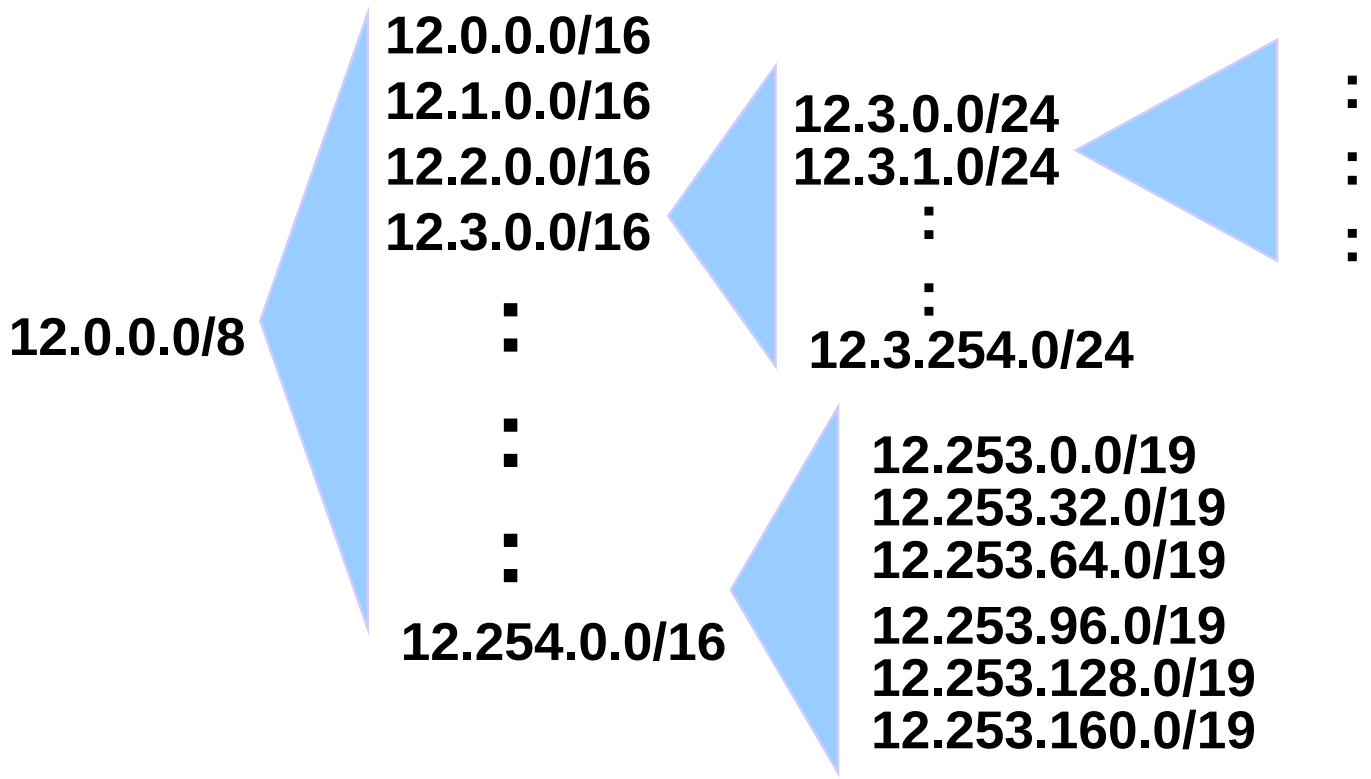
  – With CIDR, the network is assigned a continuous block of 1024 addresses with a 22-bit long prefix

# CIDR: Prefix Size vs. Network Size

| CIDR Block Prefix | # of Host Addresses |
|---|---|
| /27 | 32 hosts |
| /26 | 64 hosts |
| /25 | 128 hosts |
| /24 | 256 hosts |
| /23 | 512 hosts |
| /22 | 1,024 hosts |
| /21 | 2,048 hosts |
| /20 | 4,096 hosts |
| /19 | 8,192 hosts |
| /18 | 16,384 hosts |
| /17 | 32,768 hosts |
| /16 | 65,536 hosts |
| /15 | 131,072 hosts |

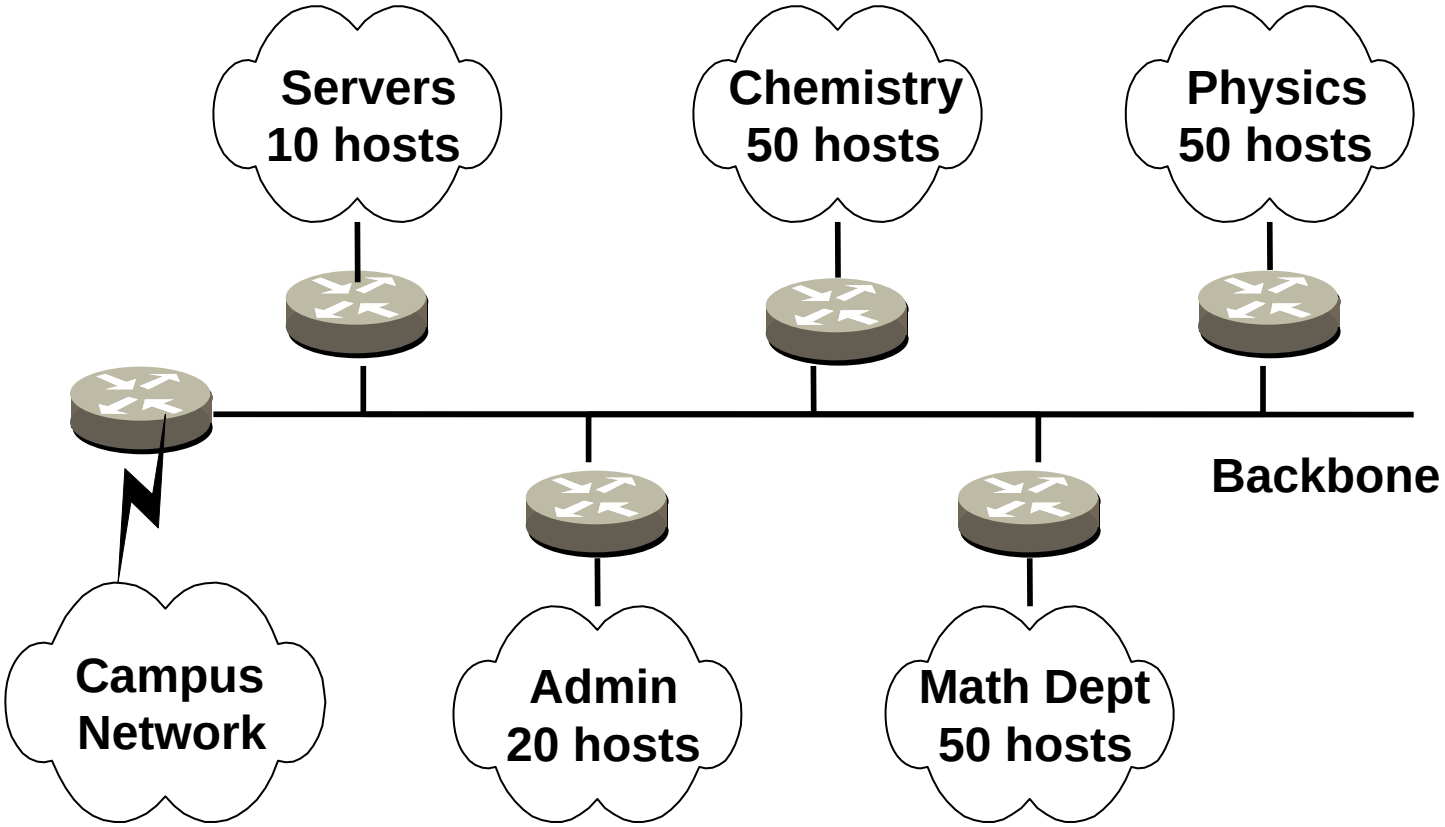
# CIDR: Hierarchal Address Allocation

- Prefixes are key to Internet scalability
  - Address allocated in contiguous chunks (prefixes)
  - Routing protocols and packet forwarding based on prefixes
  - Today, routing tables contain ~200,000 prefixes

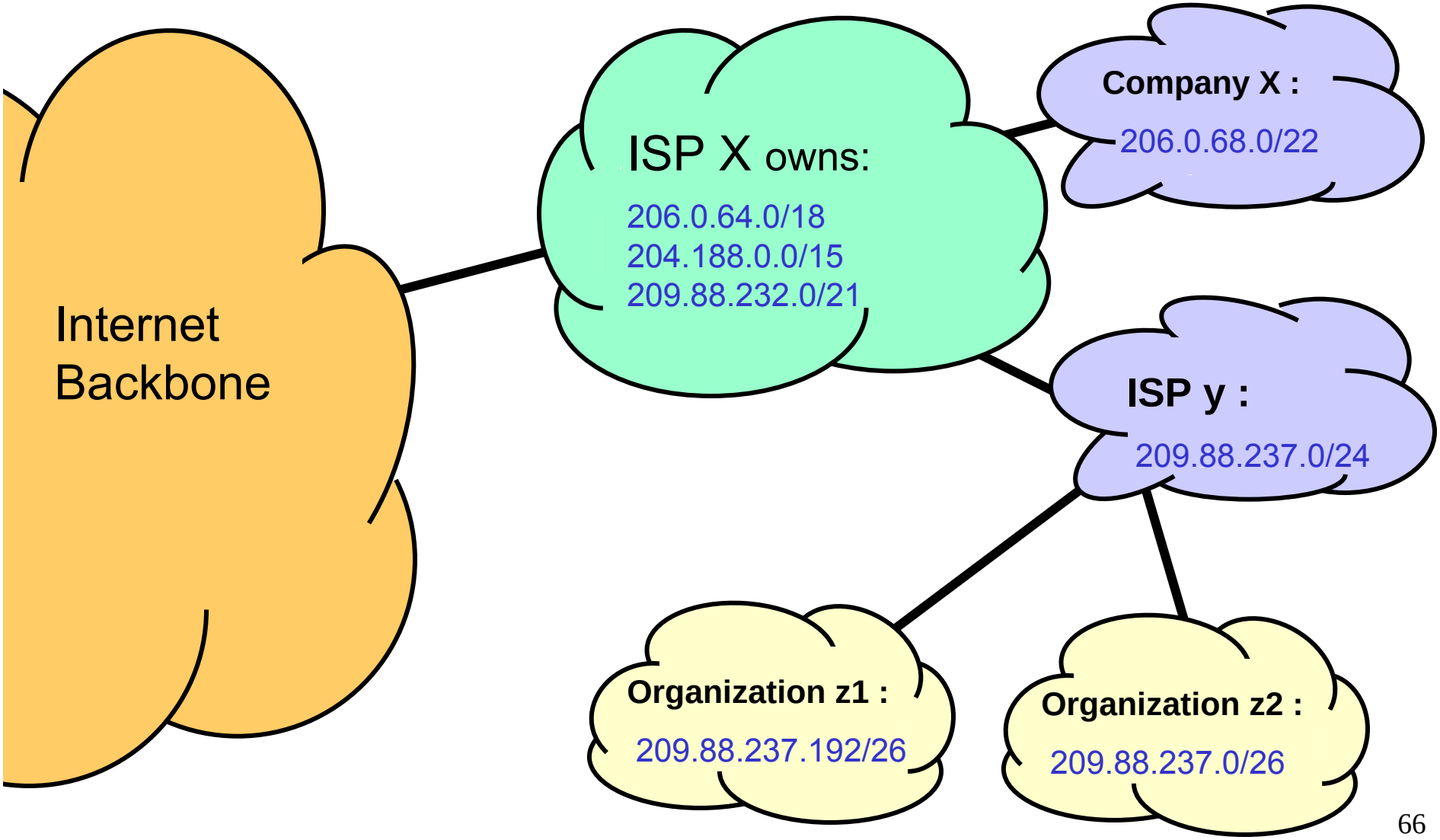


# Exercise

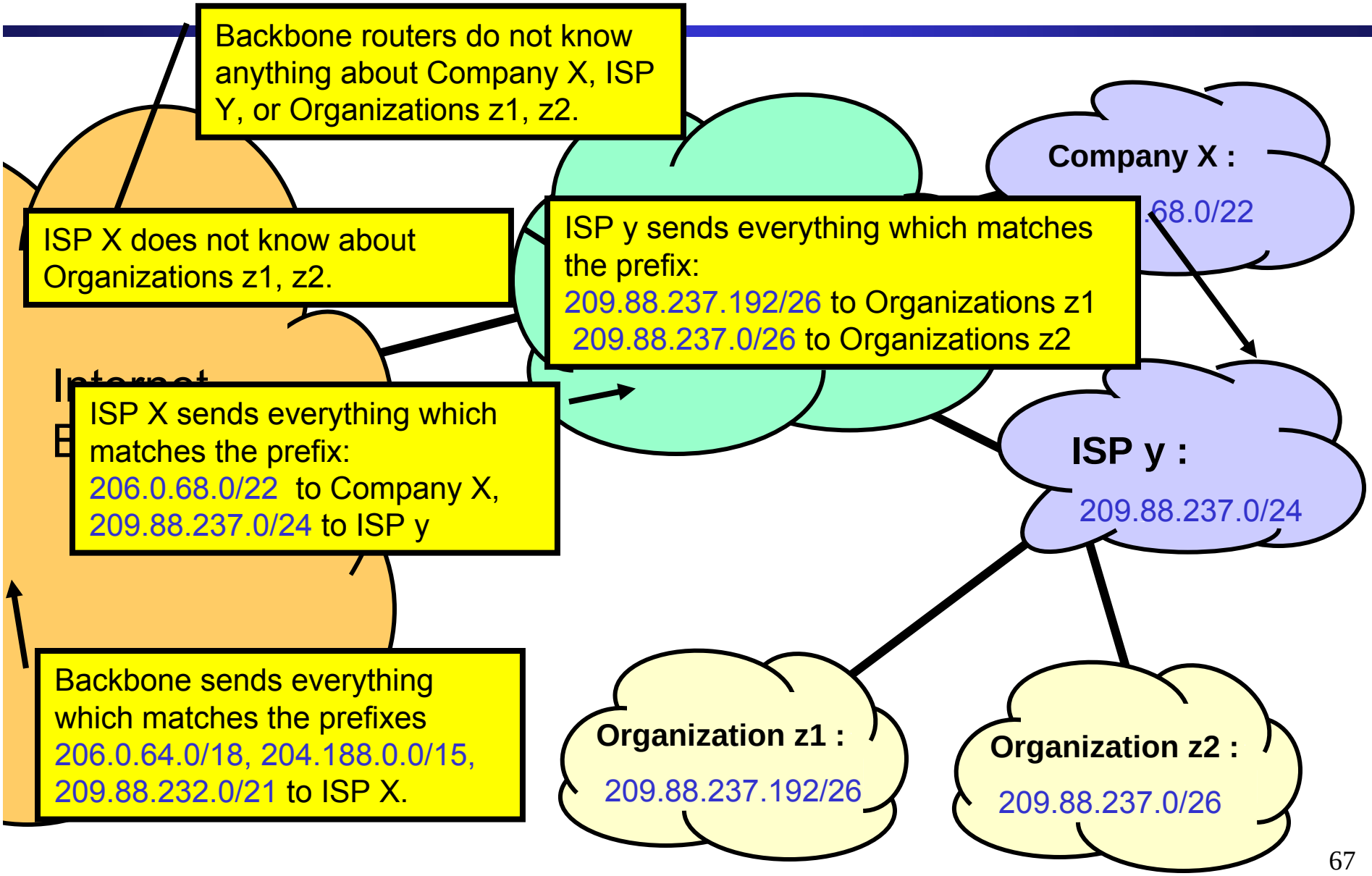
- Suppose that you are allocated the block of addresses 153.104.11.0/24 for assigning IP addresses. Choose a network address and a network mask for each network below.



# CIDR and Routing Information



# CIDR and Routing Information



# IP Forwarding

Slides by Rexford @ Princeton & Slides accompanying the Internet Lab Manual, slightly altered by M.D.

# Routing Tables

- Each router and host keeps a routing table which tells the router how to process an outgoing packet
- Main columns:
  1. Destination address: where is the IP datagram going to?
  2. Next hop or interface: how to send the IP datagram?
- Routing tables are set so that a datagram gets closer to the its destination every hop

Routing table of a host or router

IP datagrams can be directly delivered ("direct") or are sent to a router ("R4")

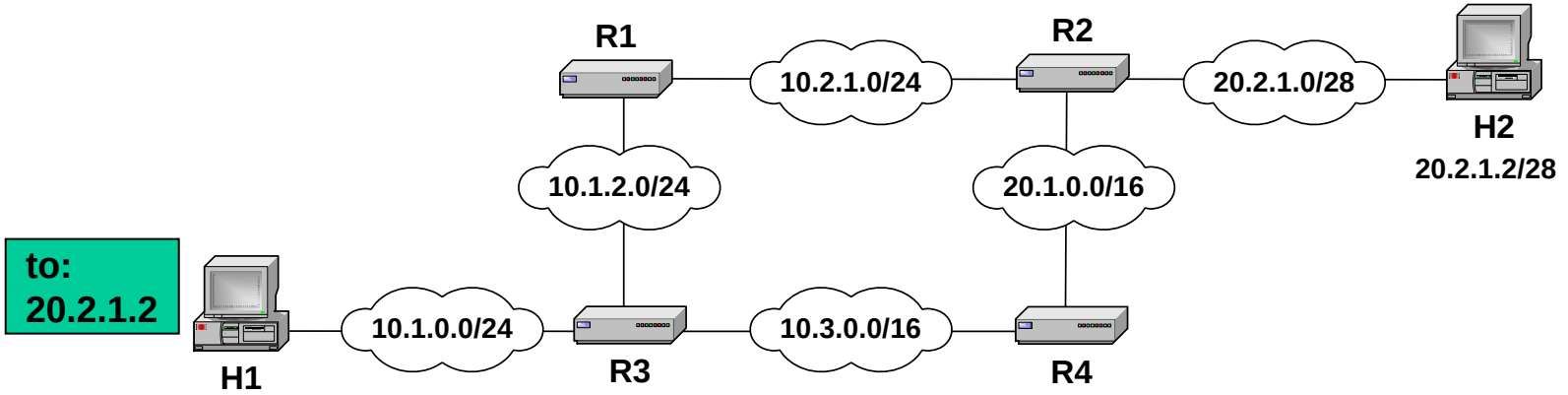
| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | direct |
| 10.1.2.0/24 | direct |
| 10.2.1.0/24 | R4 |
| 10.3.1.0/24 | direct |
| 20.1.0.0/16 | R4 |
| 20.2.1.0/28 | R4 |

# Delivery with Routing Tables

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | R3 |
| 10.1.2.0/24 | direct |
| 10.2.1.0/24 | direct |
| 10.3.1.0/24 | R3 |
| 20.2.0.0/16 | R2 |
| 30.1.1.0/28 | R2 |

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | R1 |
| 10.1.2.0/24 | R1 |
| 10.2.1.0/24 | direct |
| 10.3.1.0/24 | R4 |
| 20.1.0.0/16 | direct |
| 20.2.1.0/28 | direct |

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | R2 |
| 10.1.2.0/24 | R2 |
| 10.2.1.0/24 | R2 |
| 10.3.1.0/24 | R2 |
| 20.1.0.0/16 | R2 |
| 20.2.1.0/28 | direct |



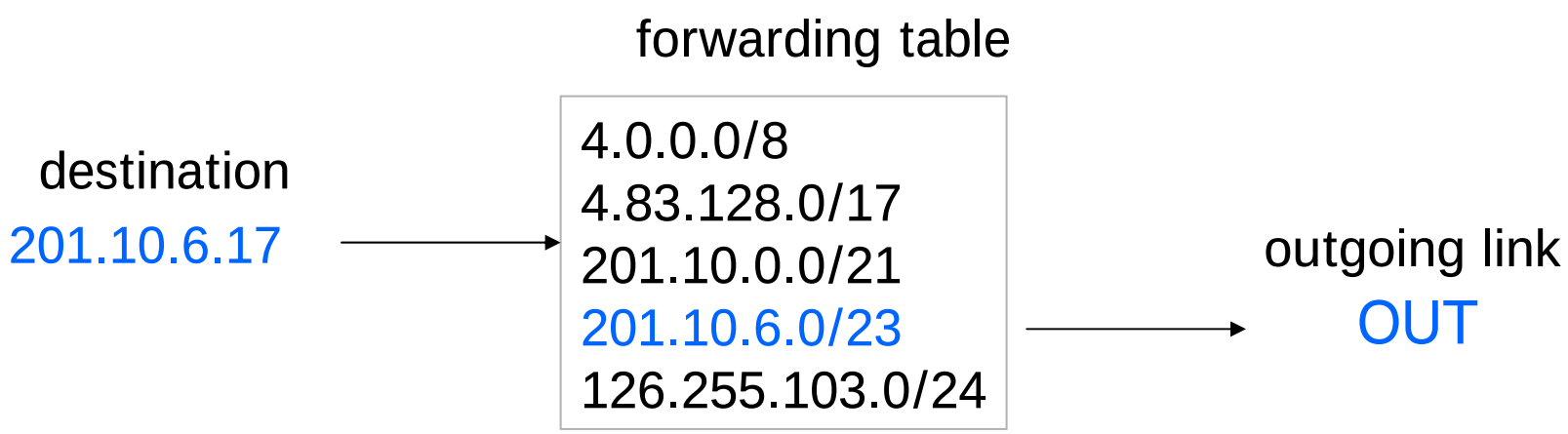

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | direct |
| 10.1.2.0/24 | R3 |
| 10.2.1.0/24 | R3 |
| 10.3.1.0/24 | R3 |
| 20.1.0.0/16 | R3 |
| 20.2.1.0/28 | R3 |

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | direct |
| 10.1.2.0/24 | direct |
| 10.2.1.0/24 | R4 |
| 10.3.1.0/24 | direct |
| 20.1.0.0/16 | R4 |
| 20.2.1.0/28 | R4 |

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | R3 |
| 10.1.2.0/24 | R3 |
| 10.2.1.0/24 | R2 |
| 10.3.1.0/24 | direct |
| 20.1.0.0/16 | direct |
| 20.2.1.0/28 | R2 |

# 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



# Exercise 1

What is the outgoing interface for 128.143.137.0?

| Prefix | Interface |
|---|---|
| 128.0.0.0/4 | interface #5 |
| 128.128.0.0/9 | interface #2 |
| 128.143.128.0/17 | interface #1 |

Routing table

# Forwarding vs. Routing

- Two distinct processes:

  1. Forwarding: How to pass a packet from an input interface to the output interface?

  2. Routing: How to find and setup the routing tables?

- Forwarding must be done as fast as possible:
  - on routers, is often done with support of hardware
  - on PCs, is done in kernel of the operating system
- Routing is less time-critical
  - On a PC, routing is done as a background process

# Processing of an IP Datagram at a Router

**Receive an IP datagram**

1. IP header validation
2. Process options in IP header
3. Parse the destination IP address
4. Routing table lookup
5. Decrement TTL
6. Perform fragmentation (if necessary)
7. Calculate checksum
8. Transmit to next hop
9. Send ICMP packet (if necessary)

# Routing Table Lookup

- When a router or host needs to transmit an IP datagram, it performs a routing table lookup
- Routing table lookup: Use the IP destination address as a key to search the routing table.
- Result of the lookup is the IP address of a next hop router, or the name of a network interface

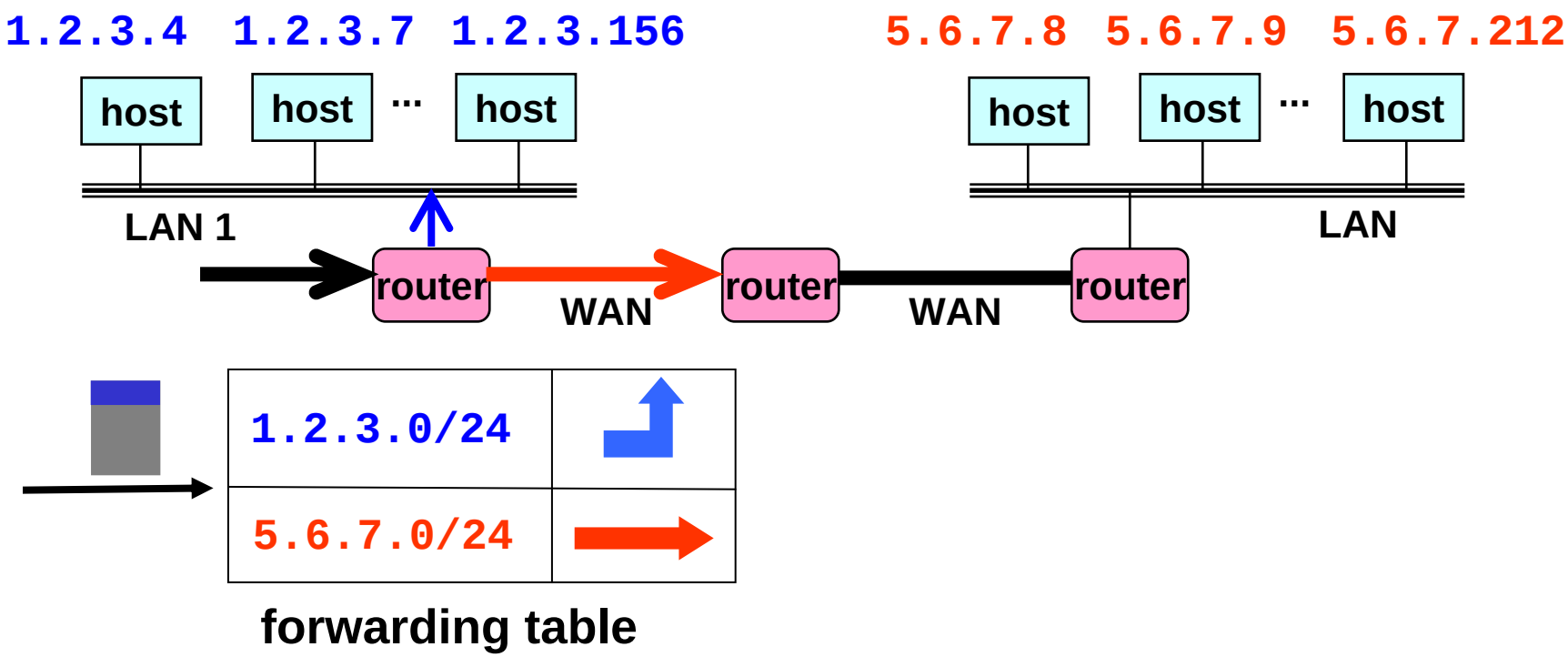
| Destination address | Next hop |
|---|---|
| network prefix<br>*or*<br>host IP address<br>*or*<br>loopback address<br>*or*<br>default route | IP address of next hop router<br>*or*<br>Name of a network interface |

# Types of Routing Table Entries

- Network route
  - Destination is a network address (e.g., 10.0.2.0/24)
  - Most entries are network routes
- Host route
  - Destination is an interface address (e.g., 10.0.1.2/32)
  - Used to specify a separate route for certain hosts
- Default route
  - Used when no network or host route matches
  - The router that is listed as the next hop of the default route is the default gateway (for Cisco: “gateway of last resort)
- Loopback address
  - Loopback address: 127.0.0.1
  - Next hop is loopback (lo0) interface as outgoing interface

# Separate Entry Per Network 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

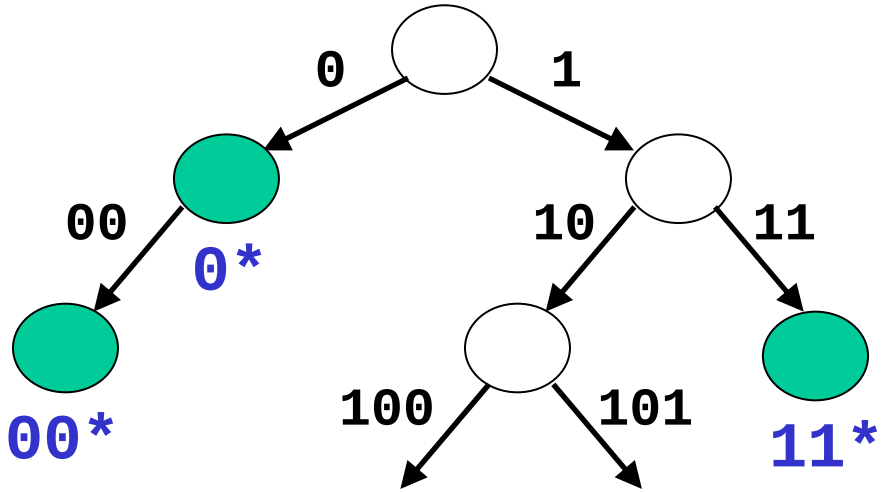


# Simplest Algorithm is Too Slow

- 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 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

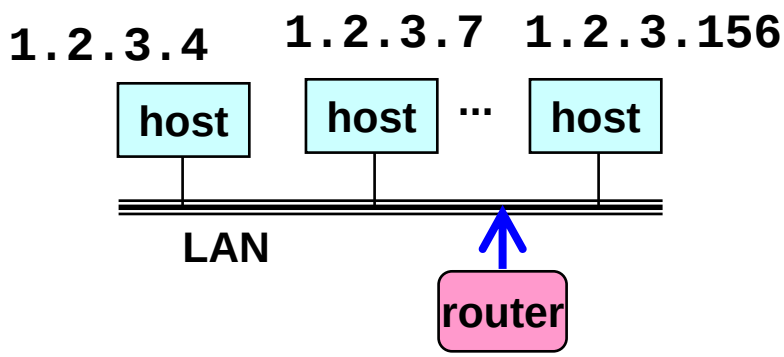
- Patricia tree is faster than linear scan
  - Proportional to number of bits in the address
- Patricia tree can be made faster
  - Can make a k-ary tree
    - E.g., 4-ary tree with four children (00, 01, 10, and 11)
  - Faster lookup, though requires more space
- Can use special hardware
  - Content 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

# How Do End Hosts Forward Packets?

- 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?

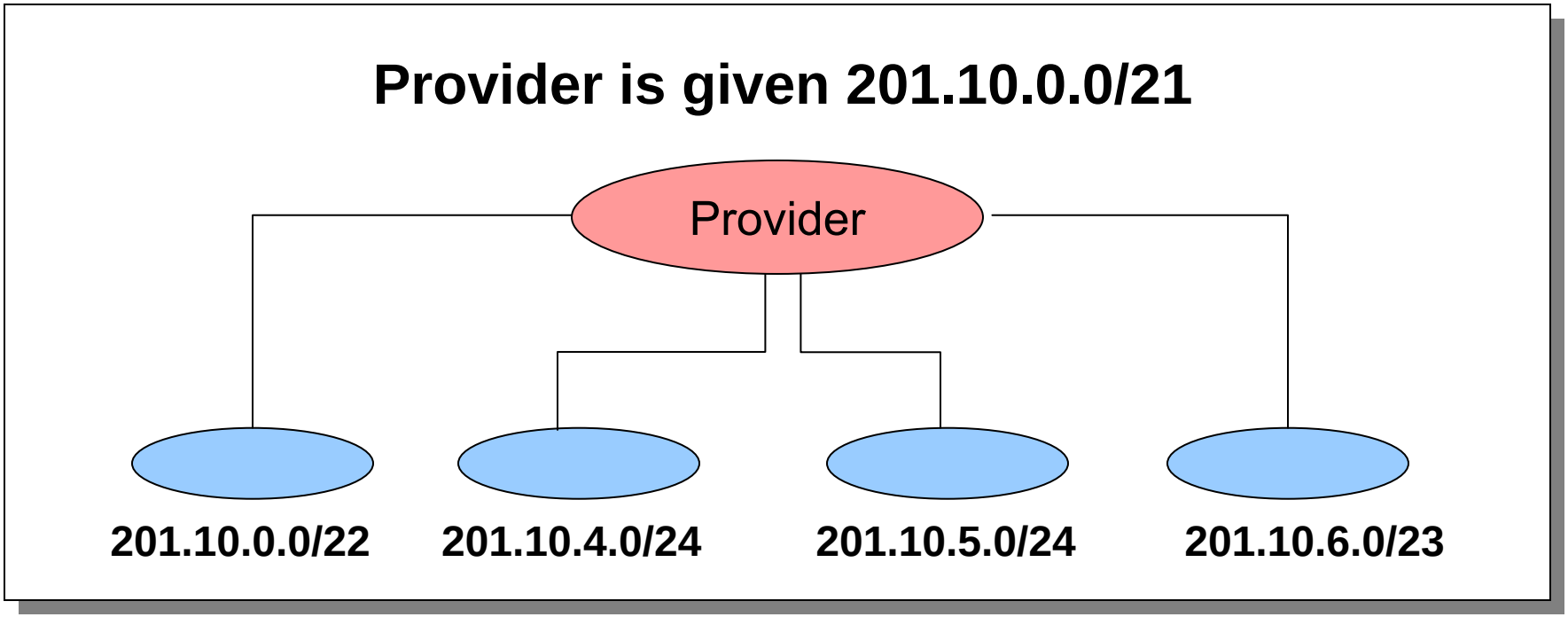



- Each interface has a persistent, global identifier
  - MAC (Media Access Control) address
  - Burned in to the adaptors Read-Only Memory (ROM)
  - Flat address structure (i.e., no hierarchy)
- Constructing an address resolution table
  - Mapping MAC address to/from IP address
  - Address Resolution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

# IP Forwarding

Slides by Rexford @ Princeton & Slides accompanying the Internet Lab Manual, slightly altered by M.D.

# 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.**

# Route Aggregation

- Longest prefix matching permits to aggregate prefixes with identical next hop address to a single entry
- This contributes significantly to reducing the size of routing tables of Internet routers

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | R3 |
| 10.1.2.0/24 | direct |
| 10.2.1.0/24 | direct |
| 10.3.1.0/24 | R3 |
| 20.2.0.0/16 | R2 |
| 20.1.1.0/28 | R2 |

| Destination | Next Hop |
|---|---|
| 10.1.0.0/24 | R3 |
| 10.1.2.0/24 | direct |
| 10.2.1.0/24 | direct |
| 10.3.1.0/24 | R3 |
| 20.0.0.0/8 | R2 |

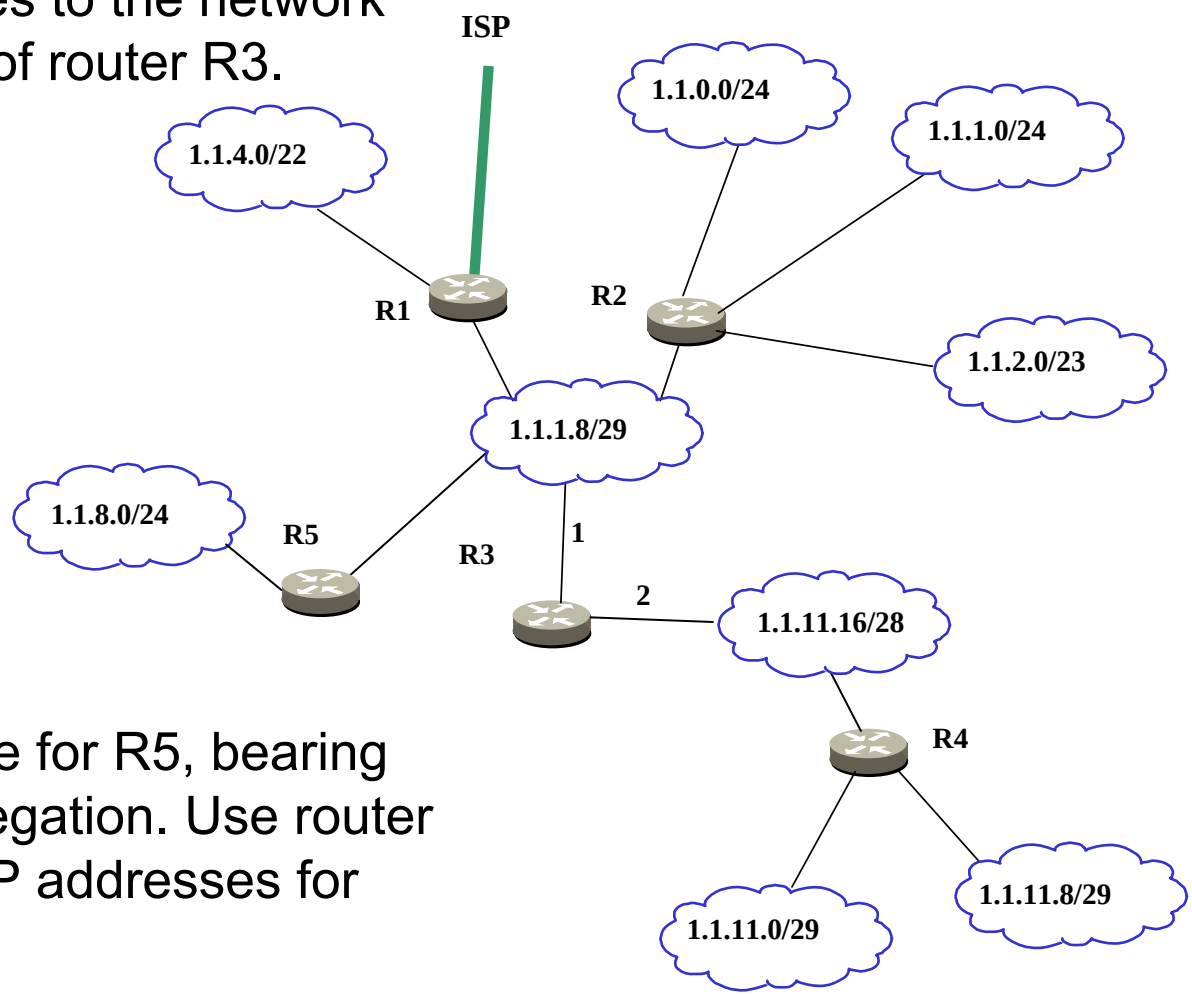
# Exercise

- Aggregate the addresses

  128.143.0.0/16 and 128.144.0.0/16

# Exercise

1. Assign IP addresses to the network interfaces 1 and 2 of router R3.

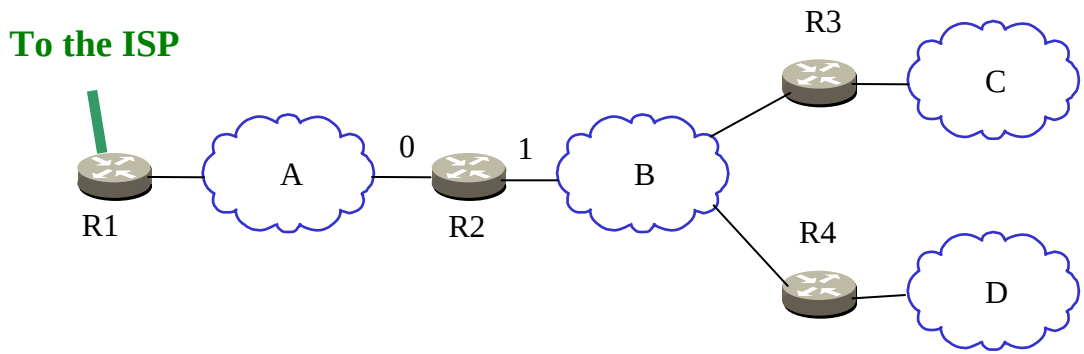



2. Build a routing table for R5, bearing in mind route aggregation. Use router names instead of IP addresses for next hops.

# Exercise

An organization is granted the block of addresses 153.104.11.0/24. The administrator wants to create 4 networks as follows:

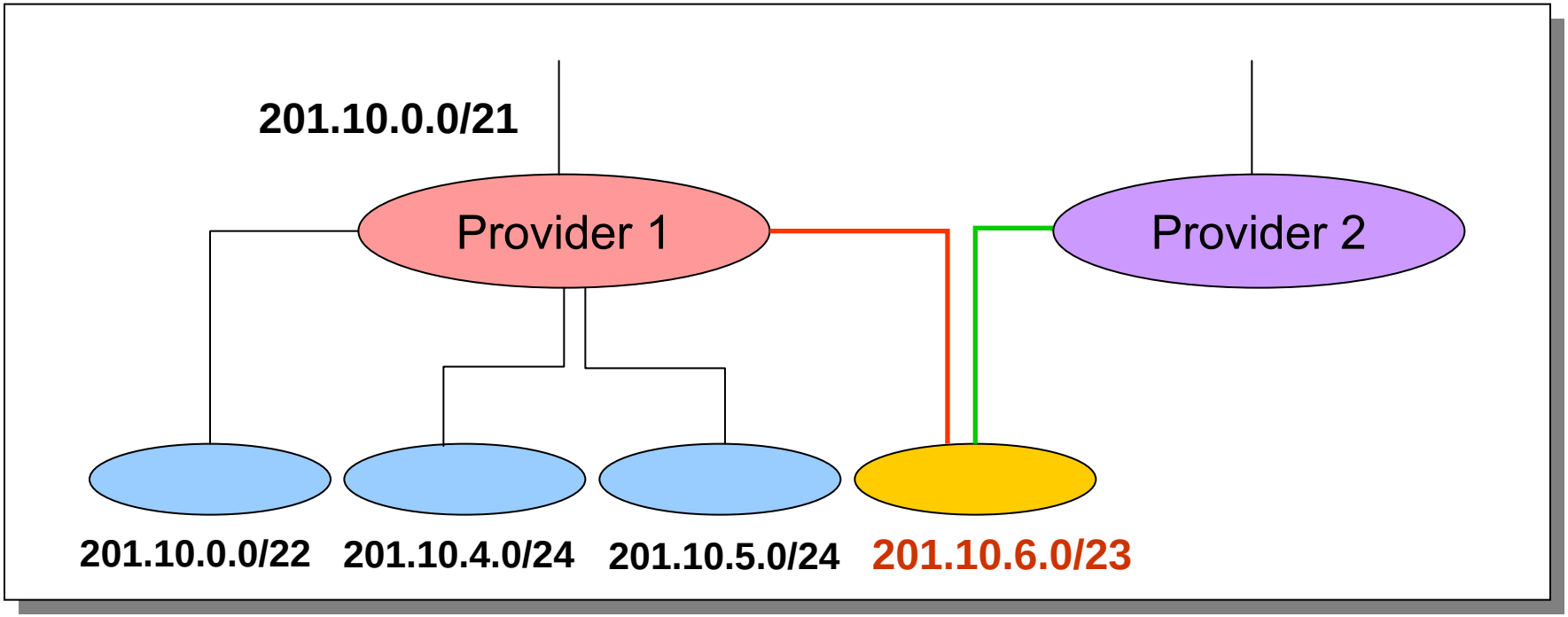
- Network A needs 100 addresses
- Network B needs 50 addresses
- Network C needs 25 addresses
- Network D needs 20 addresses.




a) Choose a network address and a network mask for each network.

b) Show the routing table for R2

# 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.**

# Obtaining a Block of Addresses

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 RIRs
- Regional Internet Registries (RIRs)
  - E.g., ARIN (American Registry for Internet Numbers)
  - Allocates address blocks within their regions
  - Allocates to ISPs 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 153.104.1.2”
  - http://www.arin.net/whois/
  - http://www.geektools.com/whois.php
  - …

# **Example Output for** 153.104.1.2

| | |
|---|---|
| OrgName: | Villanova University |
| OrgID: | VILLAN |
| Address: | 800 Lancaster Avenue |
| City: | Villanova |
| StateProv: | PA |
| PostalCode: | 19085 |
| Country: | US |
| | |
| NetRange: | 153.104.0.0 - 153.104.255.255 |
| CIDR: | 153.104.0.0/16 |
| NetName: | VILLANOVA |
| NetHandle: | NET-153-104-0-0-1 |
| Parent: | NET-153-0-0-0-0 |
| NetType: | Direct Assignment |
| NameServer: | NS1.VILLANOVA.EDU |
| RegDate: | 1989-03-13 |
| Updated: | 2007-02-09 |

# IPv6

Slides by Rexford @ Princeton & Slides accompanying the Internet Lab Manual, slightly altered by M.D.

# Are 32-bit Addresses Enough?

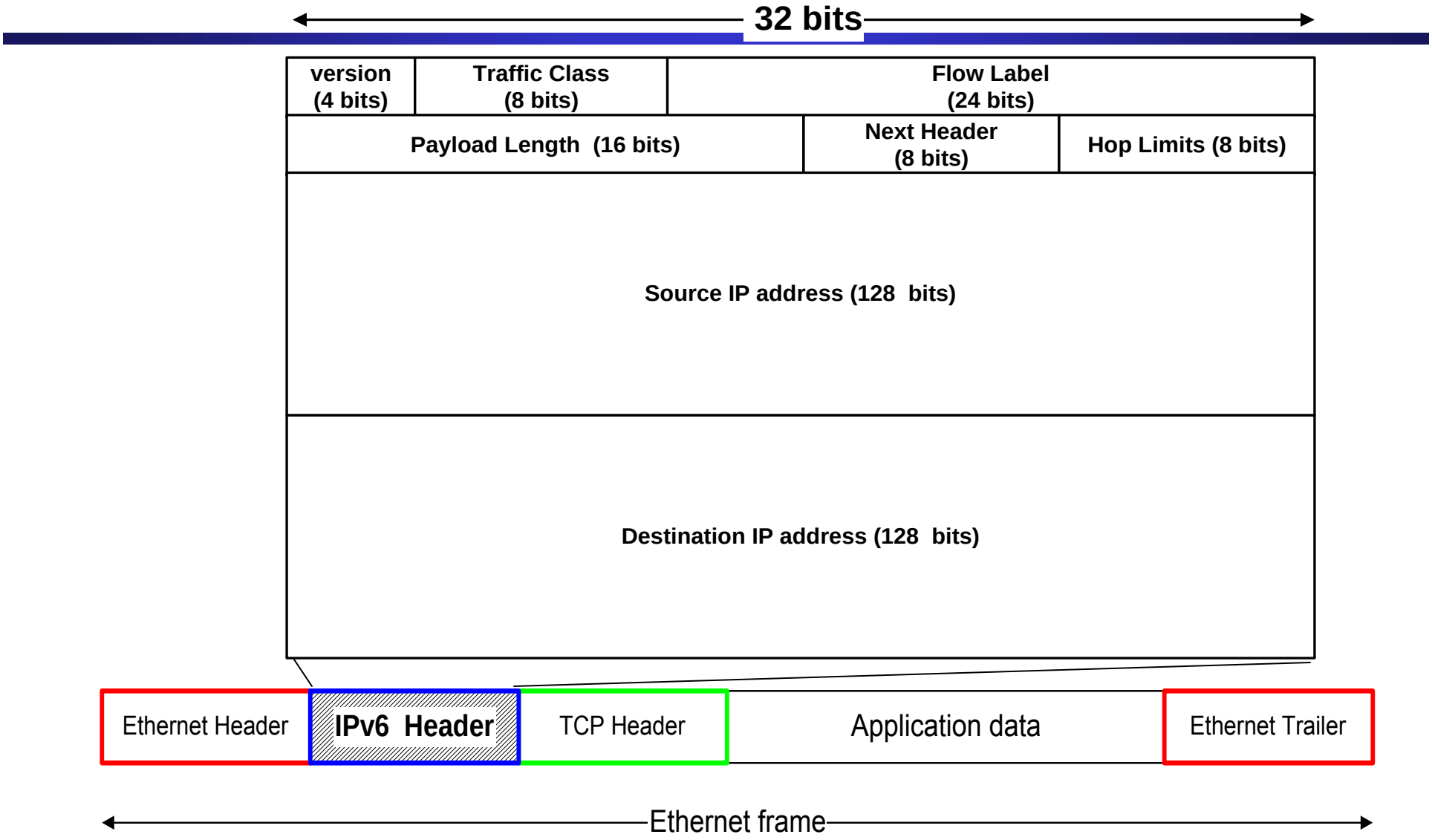
- Not all that many unique addresses
  - $2^{32}$ = 4,294,967,296 (just over four billion)
  - Plus, some are reserved for special purposes
  - And, addresses are allocated in larger blocks
- And, many devices need IP addresses
  - Computers, PDAs, routers, toasters, …
- Short-term solutions: limping along with IPv4
  - Private addresses
  - Network address translation (NAT)
  - Dynamically-assigned addresses (DHCP)
- Long-term solution: a larger address space
  - IPv6 has 128-bit addresses ($2^{128} = 3.403 \times 10^{38}$)

# IPv6 - IP Version 6

- **IP Version 6**
  - Is the successor to the currently used IPv4
  - Specification completed in 1994
  - Makes improvements to IPv4 (no revolutionary changes)

- One (not the only !) feature of IPv6 is a significant increase in of the IP address to **128 bits (16 bytes)**
  - IPv6 will solve – for the foreseeable future – the problems with IP addressing

# IPv6 Header


32 bits
version (4 bits)
Traffic Class (8 bits)
Flow Label (24 bits)
Payload Length (16 bits)
Next Header (8 bits)
Hop Limits (8 bits)
Source IP address (128 bits)
Destination IP address (128 bits)
Ethernet Header
IPv6 Header
TCP Header
Application data
Ethernet Trailer
Ethernet frame

# IPv6 vs. IPv4: Address Comparison

- **IPv4** has a maximum of

  $2^{32} \approx 4$ billion addresses

- **IPv6** has a maximum of

  **$2^{128} = (2^{32})^4 \approx 4$ billion x 4 billion x 4 billion x 4 billion**

  **addresses**

# Notation of IPv6 addresses

- **Convention**: The 128-bit IPv6 address is written as **eight 16-bit integers** (using hexadecimal digits for each integer)

  CEDF:BP76:3245:4464:FACE:2E50:3025:DF12

- Abbreviations of leading zeroes:

  CEDF:BP76:0000:0000:009E:0000:3025:DF12
  → CEDF:BP76:0:0:9E:0:3025:DF12

- ":0000:0000:0000" can be written as "::"

  CEDF:BP76:0:0:FACE:0:3025:DF12
  → CEDF:BP76::FACE:0:3025:DF12

- IPv6 addresses derived from IPv4 have 96 leading zero bits. Convention allows to use IPv4 notation for the last 32 bits.

  ::80:8F:89:90 → ::128.143.137.144

# IPv6 Provider-Based Addresses

- The first IPv6 addresses will be allocated to a provider-based plan

| 010 | Registry ID | Provider ID | Subscriber ID | Subnetwork ID | Interface ID |
|---|---|---|---|---|---|

- Type: Set to “010” for provider-based addresses
- Registry: identifies the agency that registered the address

*The following fields have a variable length*

- Provider: Id of Internet access provider *(16 bits)*
- Subscriber: Id of the organization at provider *(24 bits)*
- Subnetwork: Id of subnet within organization *(32 bits)*
- Interface: identifies an interface at a node *(48 bits)*

# More on IPv6 Addresses

- The provider-based addresses have a similar flavor as CIDR addresses

- IPv6 provides address formats for:
  - Unicast – identifies a single interface
  - Multicast – identifies a group. Datagrams sent to a multicast address are sent to all members of the group
  - Anycast – identifies a group. Datagrams sent to an anycast address are sent to one of the members in the group.
- Reading:

  http://en.wikipedia.org/wiki/IPv6

# 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

# Conclusions

- IP address
  - A 32-bit number
  - Allocated in prefixes
- Packet forwarding
  - Based on IP prefixes
  - Longest-prefix-match forwarding
- Next lecture
  - Transmission Control Protocol (TCP)
- We'll cover some topics later
  - Routing protocols, DHCP, and ARP