

# Lecture 13:

# Discovering Addresses

---

CSE 123: Computer Networks

Aaron Schulman





# Lecture 13 Overview

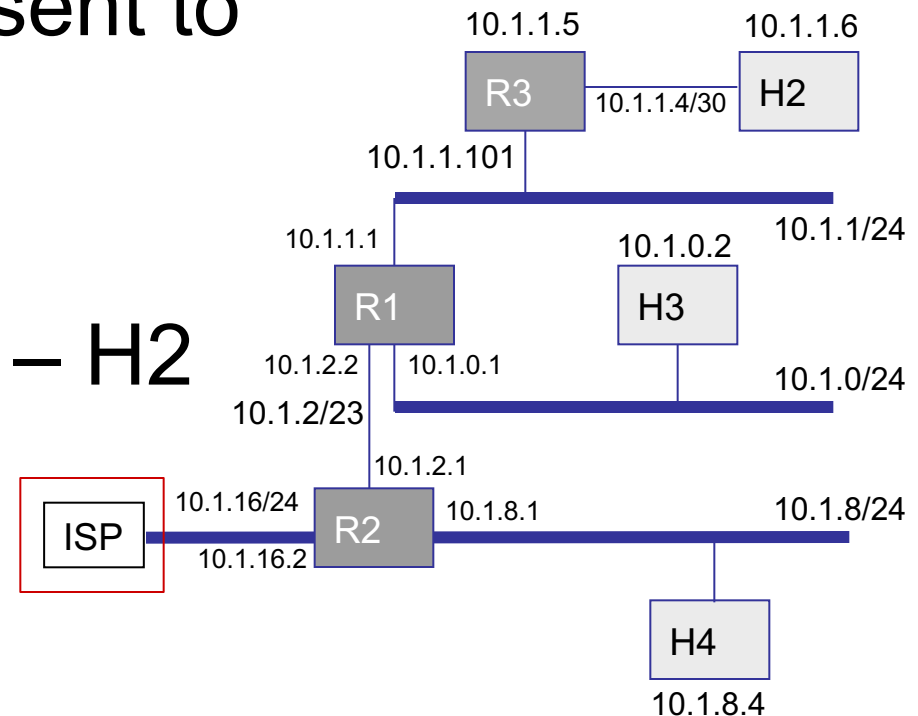
---

- Finish discussing routing
- Discovering addresses of hosts (link and network layer)
  - ◆ Link Layer (ARP)
  - ◆ Network Layer (DHCP)
  - ◆ Naming (DNS)



# IP Forwarding Example

- Packet to 10.1.1.6 is sent to to an ISP's router
- Path is R2 – R1 – R3 – H2



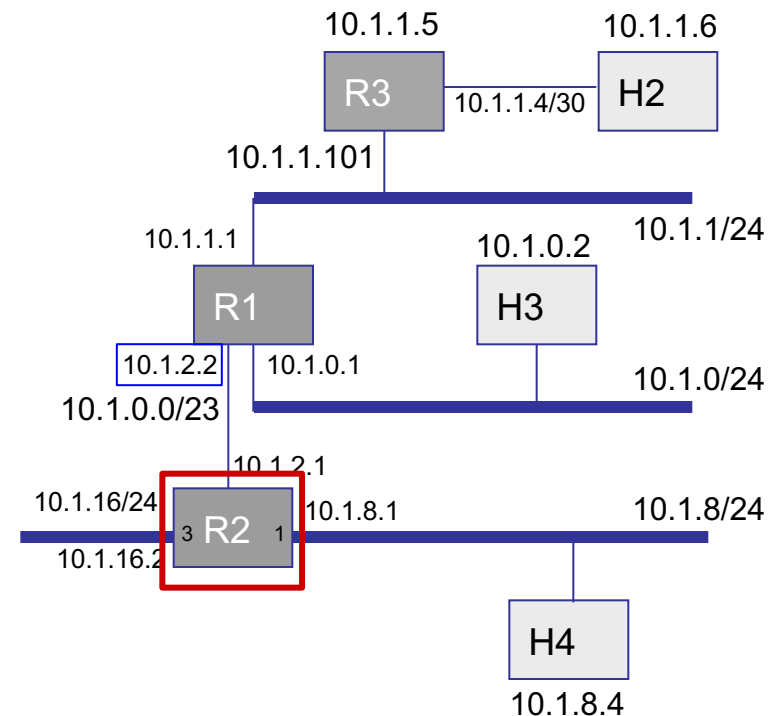


# Forwarding example (2)

- Packet to 10.1.1.6
- Matches 10.1.0.0/23

Forwarding table at R2

| Destination        | Next Hop        |
|--------------------|-----------------|
| 127.0.0.1          | loopback        |
| Default or 0/0     | 10.1.16.1       |
| 10.1.8.0/24        | interface1      |
| 10.1.2.0/23        | interface2      |
| <b>10.1.0.0/23</b> | <b>10.1.2.2</b> |
| 10.1.16.0/24       | interface3      |



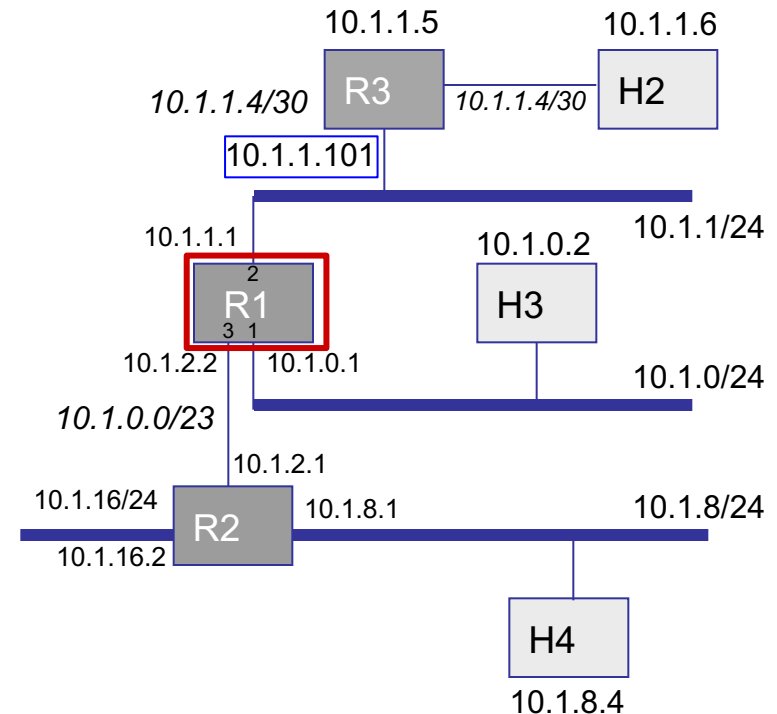


# Forwarding example (3)

- Packet to 10.1.1.6
- Matches 10.1.1.4/30
  - Longest prefix match

Routing table at R1

| Destination        | Next Hop          |
|--------------------|-------------------|
| 127.0.0.1          | loopback          |
| Default or 0/0     | 10.1.2.1          |
| 10.1.0.0/24        | interface1        |
| <b>10.1.1.0/24</b> | interface2        |
| 10.1.2.0/23        | interface3        |
| <b>10.1.1.4/30</b> | <b>10.1.1.101</b> |



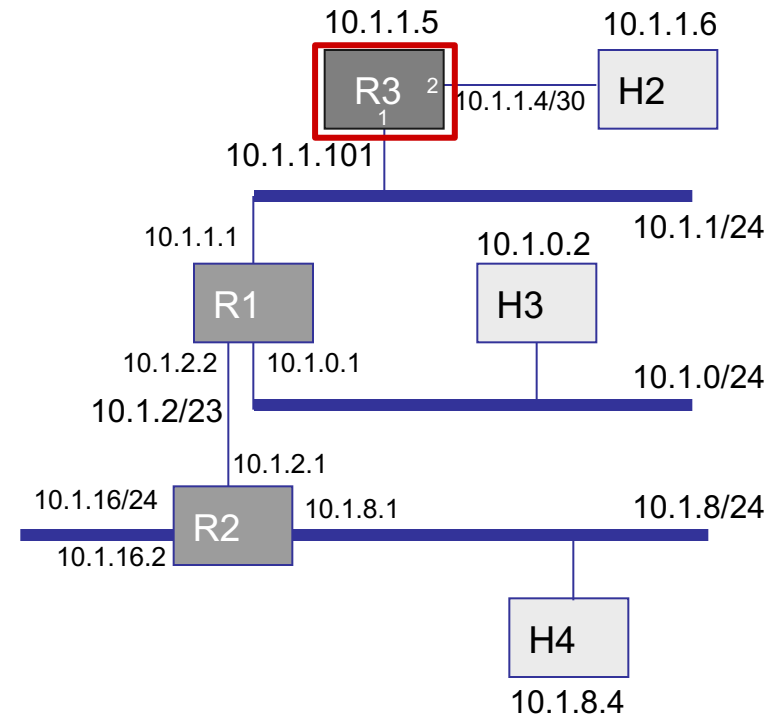


# Forwarding example (4)

- Packet to 10.1.1.6
- Direct route
  - Longest prefix match

Routing table at R3

| Destination        | Next Hop   |
|--------------------|------------|
| 127.0.0.1          | loopback   |
| Default or 0/0     | 10.1.1.1   |
| <b>10.1.1.0/24</b> | interface1 |
| <b>10.1.1.4/30</b> | interface2 |





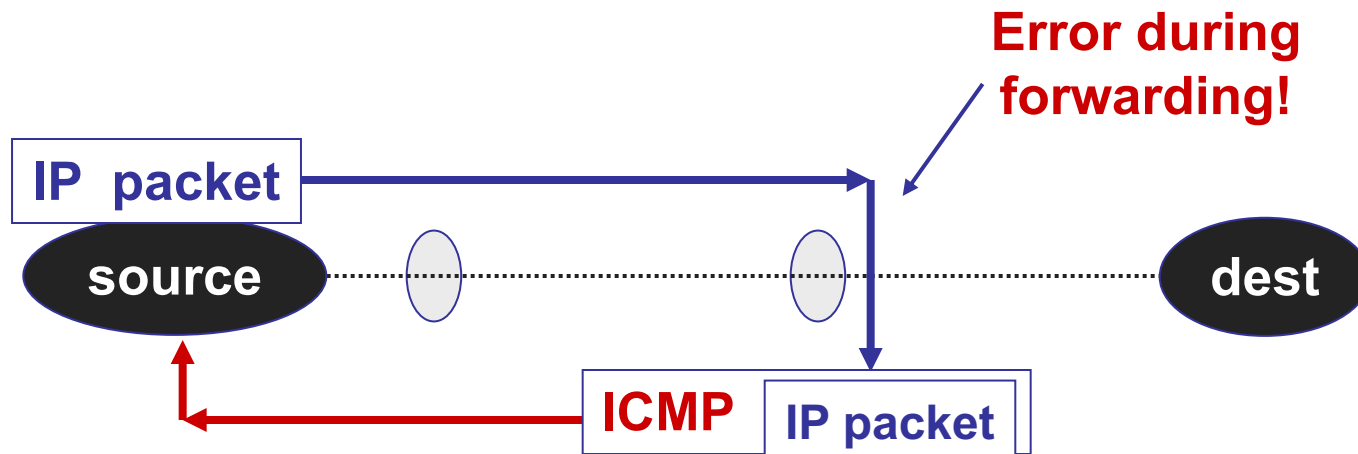
# Dealing with errors: ICMP

---

- What happens when things go wrong in IP?
  - ◆ Need a way to test/debug a large, widely distributed system
- ICMP = Internet Control Message Protocol (RFC792)
  - ◆ Companion to IP – required functionality
- Used for error and information reporting:
  - ◆ Errors that occur during IP forwarding
  - ◆ Queries about the status of the network



# ICMP Error Message Generation





# Common ICMP Messages

---

- Destination unreachable
  - ◆ “Destination” can be host, network, port, or protocol
- Redirect
  - ◆ To shortcut circuitous routing
- TTL Expired
  - ◆ Used by the “traceroute” program
    - » traceroute traces packet routes through Internet
- Echo request/reply
  - ◆ Used by the “ping” program
    - » ping just tests for host liveness
- ICMP messages include portion of IP packet that triggered the error (if applicable) in their payload



# Example of a traceroute

```
schulman@yem ~ % traceroute www.google.com
traceroute to www.google.com (142.250.68.36), 64 hops max, 52 byte packets
 1  ebu-3b-720-vl441-cse-sysnet-gw-222-1.ucsd.edu (137.110.222.1)  0.655 ms  0.340 ms  0.343 ms
 2  m-core-agg-b.ucsd.edu (132.239.254.45)  0.374 ms  0.382 ms  0.342 ms
 3  mx0--nodem-core-30ge.ucsd.edu (132.239.254.162)  2.423 ms  4.941 ms  1.427 ms
 4  dc-sdg-agg4--ucsd-100ge.cenic.net (137.164.23.176)  0.830 ms  0.687 ms  0.985 ms
 5  dc-tus-agg8--sdg-agg4-300g.cenic.net (137.164.11.84)  3.541 ms  3.414 ms  3.601 ms
 6  dc-lax-agg8--tus-agg8-300g.cenic.net (137.164.11.82)  3.751 ms  3.509 ms  4.144 ms
 7  dc-lax-agg10--lax-agg8-300g.cenic.net (137.164.11.70)  3.721 ms  4.785 ms  3.972 ms
 8  72.14.222.56 (72.14.222.56)  3.082 ms  2.919 ms
    74.125.49.165 (74.125.49.165)  2.953 ms
 9  108.170.247.225 (108.170.247.225)  3.094 ms  3.119 ms *
10  142.251.60.98 (142.251.60.98)  3.080 ms
    142.250.226.109 (142.250.226.109)  3.700 ms
    lax17s46-in-f4.1e100.net (142.250.68.36)  3.017 ms
```



# ICMP Restrictions

---

- The generation of error messages is limited to avoid cascades ... error causes error that causes error...
- Don't generate ICMP error in response to:
  - ◆ An ICMP error
  - ◆ Broadcast/multicast messages (link or IP level)
  - ◆ IP header that is corrupt or has bogus source address
  - ◆ Fragments, except the first
- ICMP messages are often rate-limited too
  - ◆ Don't waste valuable bandwidth sending tons of ICMP messages

# Whence come IP Addresses?



- You already have a bunch from the days when you called Jon Postel and asked for them (e.g. BBN)
- You get them from another provider
  - ◆ E.g. buy service from Level 3 and get a /24 from one of their address blocks
  - ◆ UCSD or Spectrum/AT&T are giving you addresses now...
    - » One address per customer in IPv4, /64 per customer in IPv6
- You get one directly from a Internet number registry
  - ◆ ARIN: North America, APNIC (Asia Pacific), RIPE (Europe), LACNIC (Latin America), AFRINIC (Africa)
  - ◆ Registries get address from IANA (Internet Assigned Numbers Authority)



# How Do Hosts Get One?

---

- Manual configuration
  - ◆ They tell you over the phone or email, you type that number into your computer (along with the default gateway, DNS server, etc.)
- Automated configuration
  - ◆ Dynamic Host Resolution Protocol (DHCP)
  - ◆ Domain Name System (DNS)

# Mapping Between Identifiers

---



- Address Resolution Protocol (**ARP**)
  - ◆ Given an **IP address (L3)**, provide the **MAC address (L2)**
  - ◆ To enable communication within the Local Area Network
- Dynamic Host Configuration Protocol (**DHCP**)
  - ◆ Automates host boot-up process
  - ◆ Given a **MAC address**, assign a unique **IP address**
  - ◆ ... and tell host other stuff about the Local Area Network
- Domain Name System (**DNS**)
  - ◆ Given a **host name**, provide the **IP address**
  - ◆ Given an **IP address**, provide the **host name**



# Layers of Identifiers

---

- **MAC address** (e.g., 58:B0:35:F2:3C:D9)
  - ◆ Used by *network adaptors* to identify interesting frames
  - ◆ Unique, hard-coded identifier burned into network adaptor
  - ◆ Flat name space (of 48 bits in Ethernet)
- **IP address** (e.g., 128.54.70.238)
  - ◆ Used by *routers* to forward packets
  - ◆ Unique, topologically meaningful locator
  - ◆ Hierarchical namespace of 32 bits (or 128 bits for IPv6)
- **Host name** (e.g., www.ucsd.edu)
  - ◆ Used by *humans* to specify host of interest
  - ◆ Unique, selected by host administrator
  - ◆ Hierarchical, variable-length string of alphanumeric characters

# Naming – it has hierarchy too!

---



- MAC addresses: **58:B0:35:F2:3C:D9**
  - ◆ **OIDs**: assigned to vendors by the IEEE
  - ◆ **Adapters**: assigned by the vendor from its block
- IP addresses: **128.54.70.238**
  - ◆ **Prefixes**: ICANN, regional Internet registries, and ISPs
  - ◆ **Hosts**: static configuration, or dynamic using DHCP
- Host **name**: **www.ucsd.edu**
  - ◆ **Domain**: registrar for each top-level domain (e.g., .edu)
  - ◆ **Host name**: local org. administrator assigns to each host

# Address Resolution Protocol

---



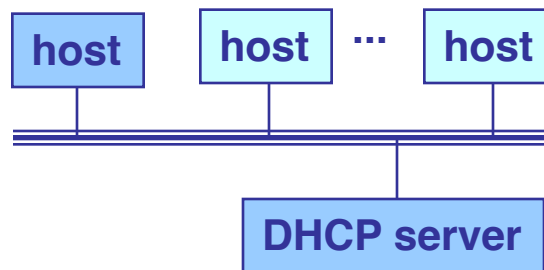
- Every node maintains an ARP table
  - ◆ (IP address, MAC address) pair
- Consult the table when sending a packet
  - ◆ Map destination **IP address to MAC address**
  - ◆ Encapsulate and transmit the data packet
- What if the IP address is not in the table?
  - ◆ Broadcast: “Who has IP address x.x.x.x?”
  - ◆ Response: “MAC address yy:yy:yy:yy:yy:yy”
  - ◆ Sender caches the result in its ARP table



# Bootstrapping Problem

---

- Host doesn't have an IP address yet at boot
  - ◆ So, host doesn't know what source address to use
- Host doesn't know who to ask for an IP address
  - ◆ So, host doesn't know what destination address to use
- Solution: broadcast to discover a server who can help
  - ◆ Install a special server on the LAN to answer IP distress calls



# DHCP



- Broadcast-based LAN protocol algorithm
  - ◆ Host broadcasts “**DHCP discover**” on LAN (e.g. Ethernet broadcast)
  - ◆ DHCP server responds with “**DHCP offer**” message
  - ◆ Host requests IP address: “**DHCP request**” message
  - ◆ DHCP server sends address: “**DHCP ack**” message w/IP address
- Easy to have fewer addresses than hosts (e.g. UCSD wireless) and to *renumber* network (use new addresses)
- What if host goes away (how to get address back?)
  - ◆ Address is a “lease” not a “grant”, has a timeout
  - ◆ Can a host have different IP addresses at different times?

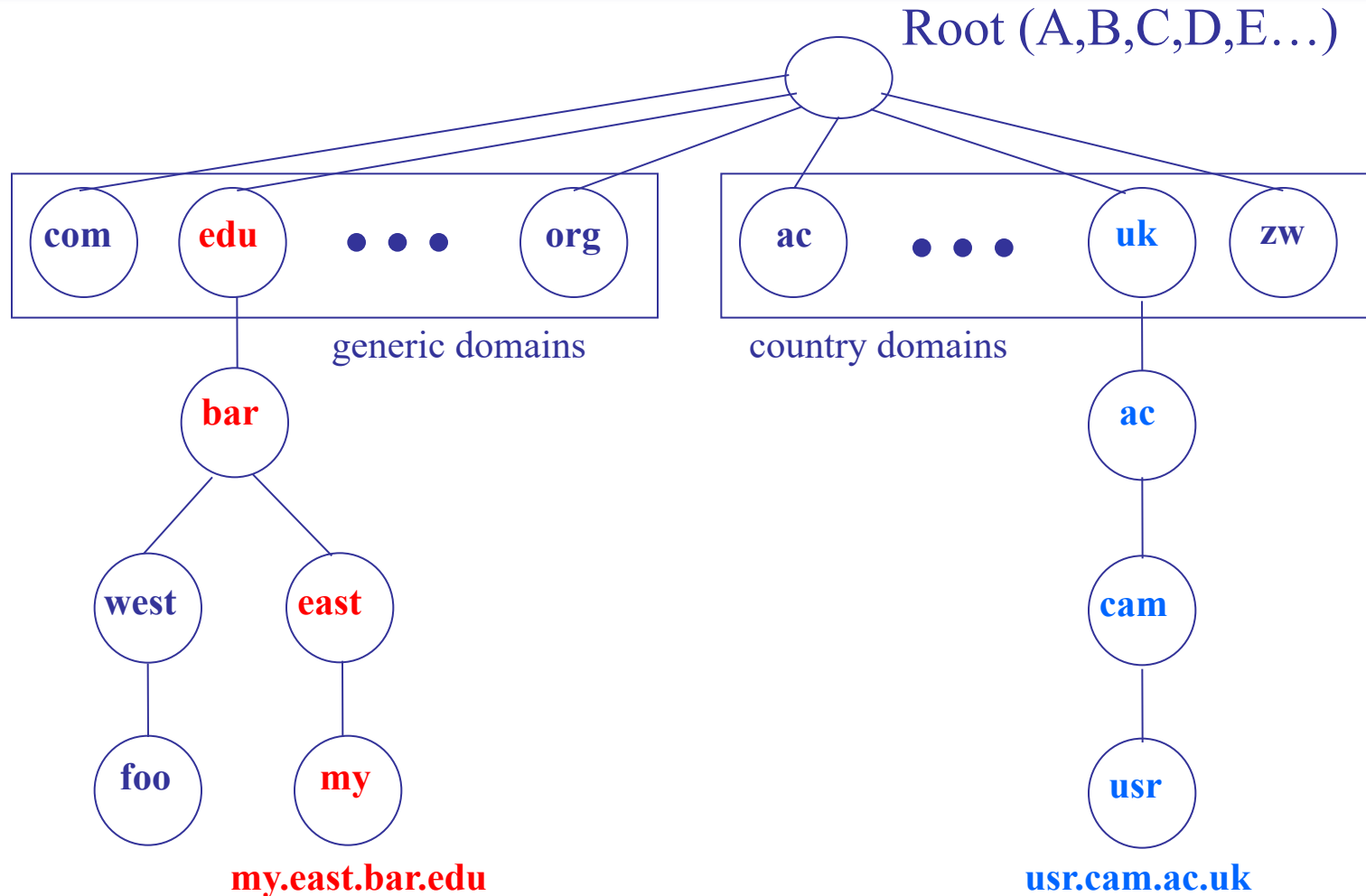
# Domain Name System (DNS)

---



- Distributed administrative control
  - ◆ **Hierarchical** name space divided into zones
  - ◆ **Distributed** over a collection of DNS servers
- Hierarchy of DNS servers
  - ◆ Root servers
  - ◆ Top-level domain (TLD) servers
  - ◆ Authoritative DNS servers
- Performing the translations
  - ◆ Local DNS servers
  - ◆ Resolver software

# DNS: Distributed Database



# DNS Root Servers



- 13 root servers (see <http://www.root-servers.org/>)
  - ◆ Labeled A through M



# Using DNS

---

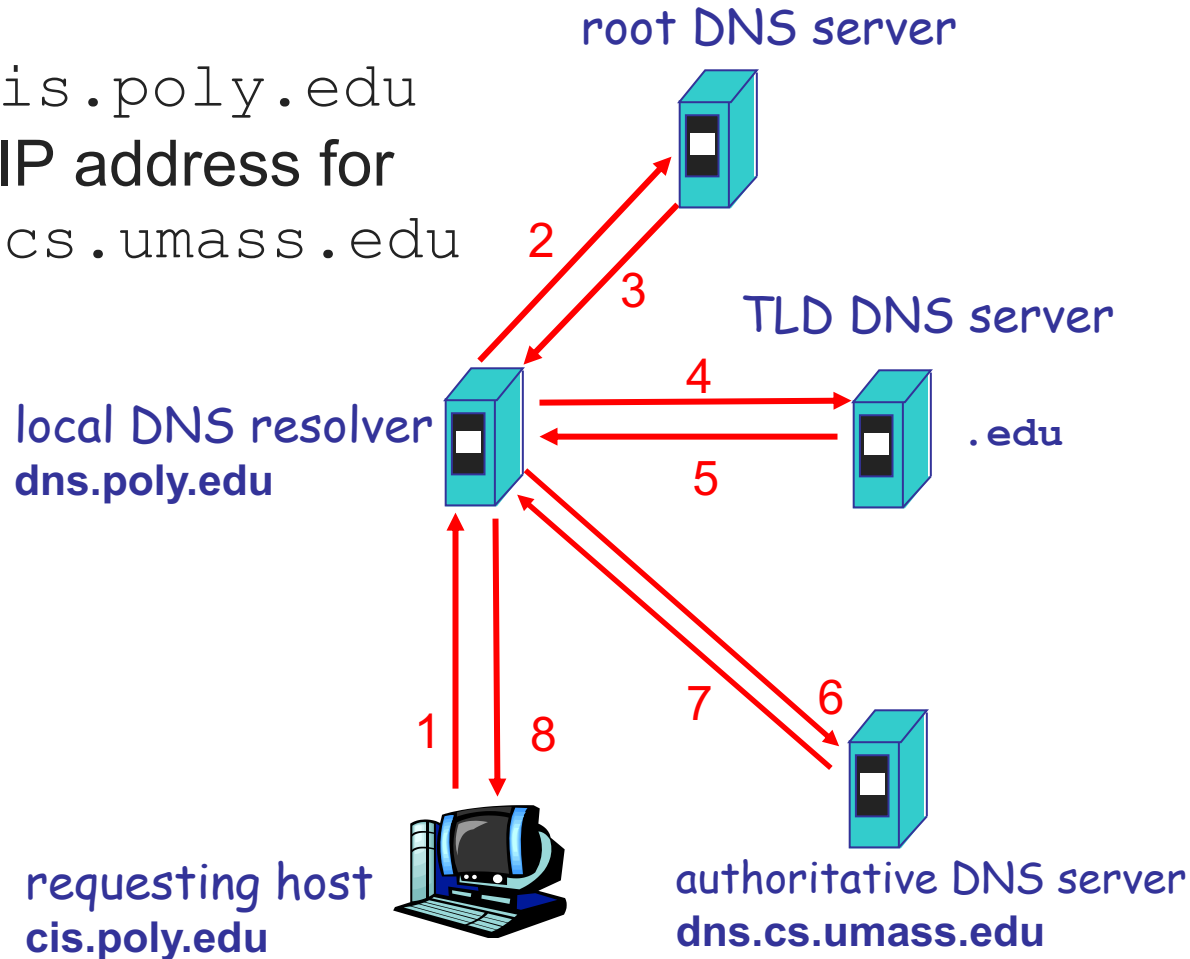


- Local DNS resolver (“default name server”)
  - ◆ Usually near the end hosts who use it
  - ◆ Local hosts configured with local server (e.g., `/etc/resolv.conf`) or learn the server via DHCP
- Client application
  - ◆ Extract server name (e.g., from the URL)
  - ◆ Do *gethostbyname()* to trigger resolver code
- Server application
  - ◆ Extract client IP address from socket
  - ◆ Optional *gethostbyaddr()* to translate into name

# Example



Host at `cis.poly.edu`  
wants IP address for  
`gaia.cs.umass.edu`



# Reliability

---



- DNS servers are replicated
  - ◆ Name service available if at least one replica is up
  - ◆ Queries can be load balanced between replicas
- UDP used for queries (Why not TCP?)
  - ◆ Need reliability: so must implement this on top of UDP
  - ◆ Try alternate servers on timeout
  - ◆ Exponential backoff when retrying same server
- Cache responses to decrease load
  - ◆ Both at end hosts and local servers



# Private IP Address Space

---

- Sometimes you can't get/don't want IP addresses
  - ◆ An organization wants to change service providers without having to renumber its entire network
  - ◆ A network may be unable obtain (or cannot afford) enough IP addresses for all of its hosts
- IP provides **private address space** anyone can use
  - ◆ 10/8, 192.168/16, 172.16.0/20
  - ◆ These addresses are not routable—Internet routers should drop packets destined to these so-called **bogons**
- What good are they if can't use them on the Internet?



# For Next Time

---

- Read P&D section 3.4
- Midterm on Friday!