

# CS 352

# The Application Layer

Lecture 3.1, Spring 2020

<http://www.cs.rutgers.edu/~sn624/352>

Srinivas Narayana

# Application-layer Protocol

- Types of messages exchanged,
  - e.g., request, response
- **Message format:**
  - Syntax: what fields in messages & how fields are delineated
  - Semantics: meaning of information in fields
- **Actions:** when and how processes send & respond to messages

## Public-domain protocols:

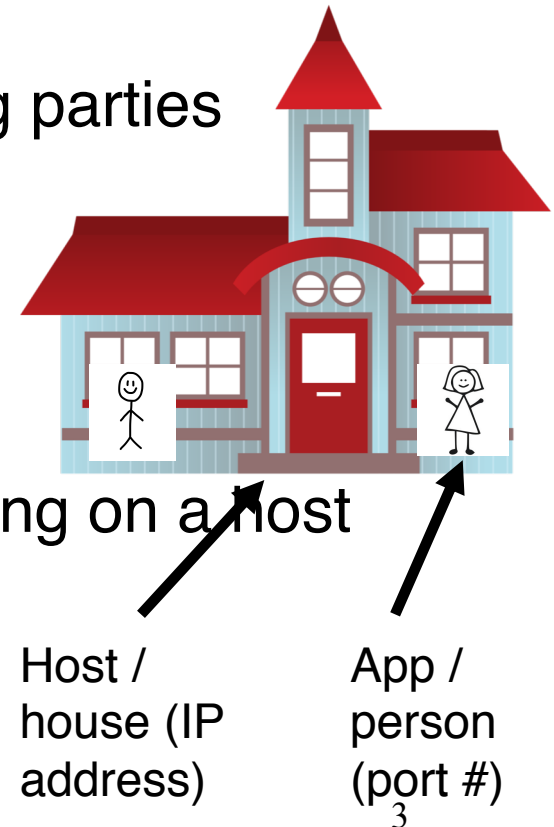
- defined in RFCs
- allows for interoperability
- e.g., HTTP, SMTP

## Proprietary protocols:

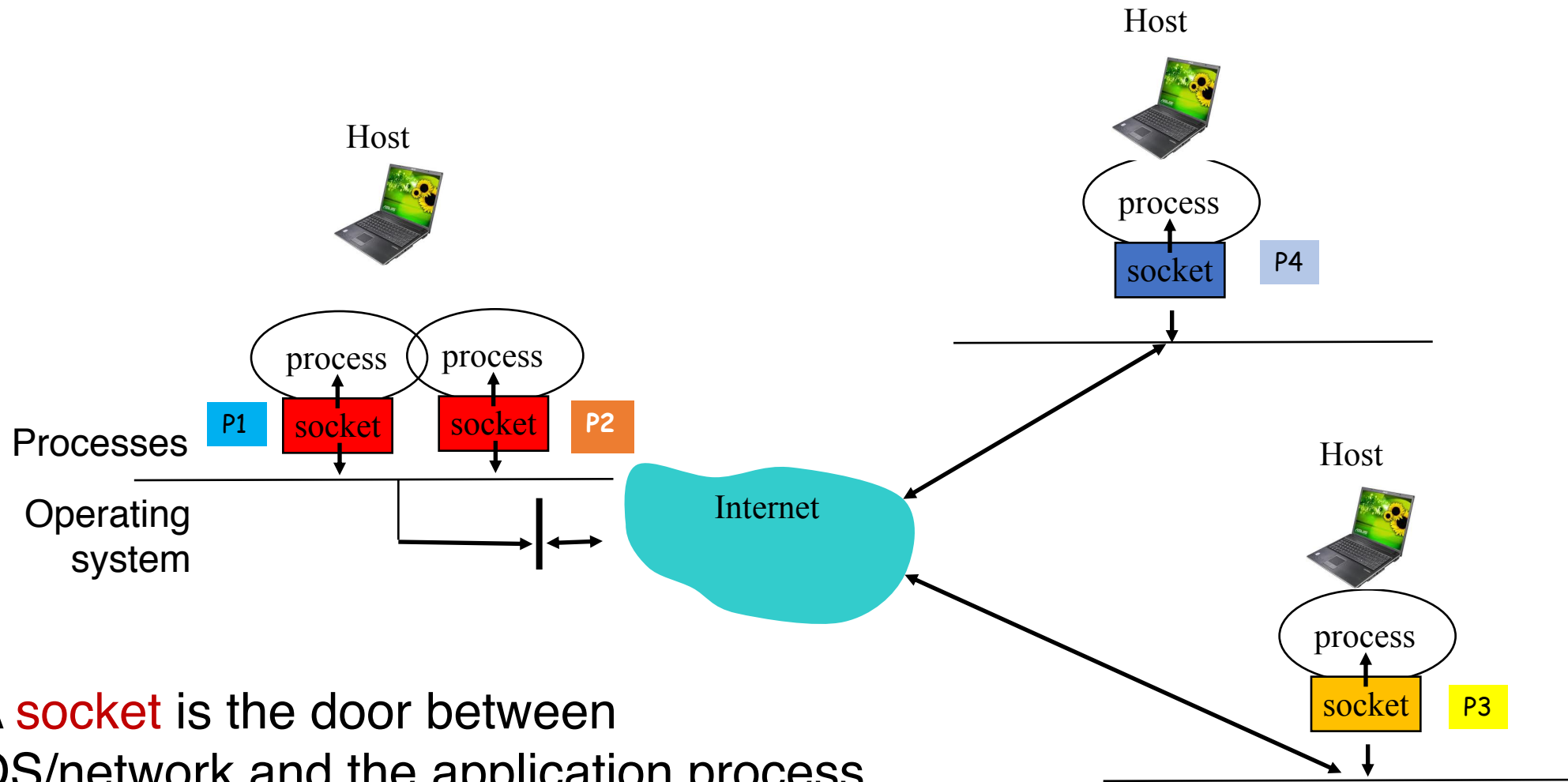
- e.g., Skype

# Application Addresses

- We usually think of an application executing on a single endpoint
- However, applications can reside on, say, 2 different endpoints connected by a network
- In order to communicate, need to identify the communicating parties
  - Telephone network: phone number (10 digits)
- Computer network: **IP address**
  - IPv4 (32 bits) 128.6.24.78
  - IPv6 (128 bits) 2001:4000:A000:C000:6000:B001:412A:8000
- Suppose there is more than one networked program executing on a host
  - In addition to host address, we need one more address
    - “Which Program to talk to?”
- The identity for an application: **port number (+ IP addr)**



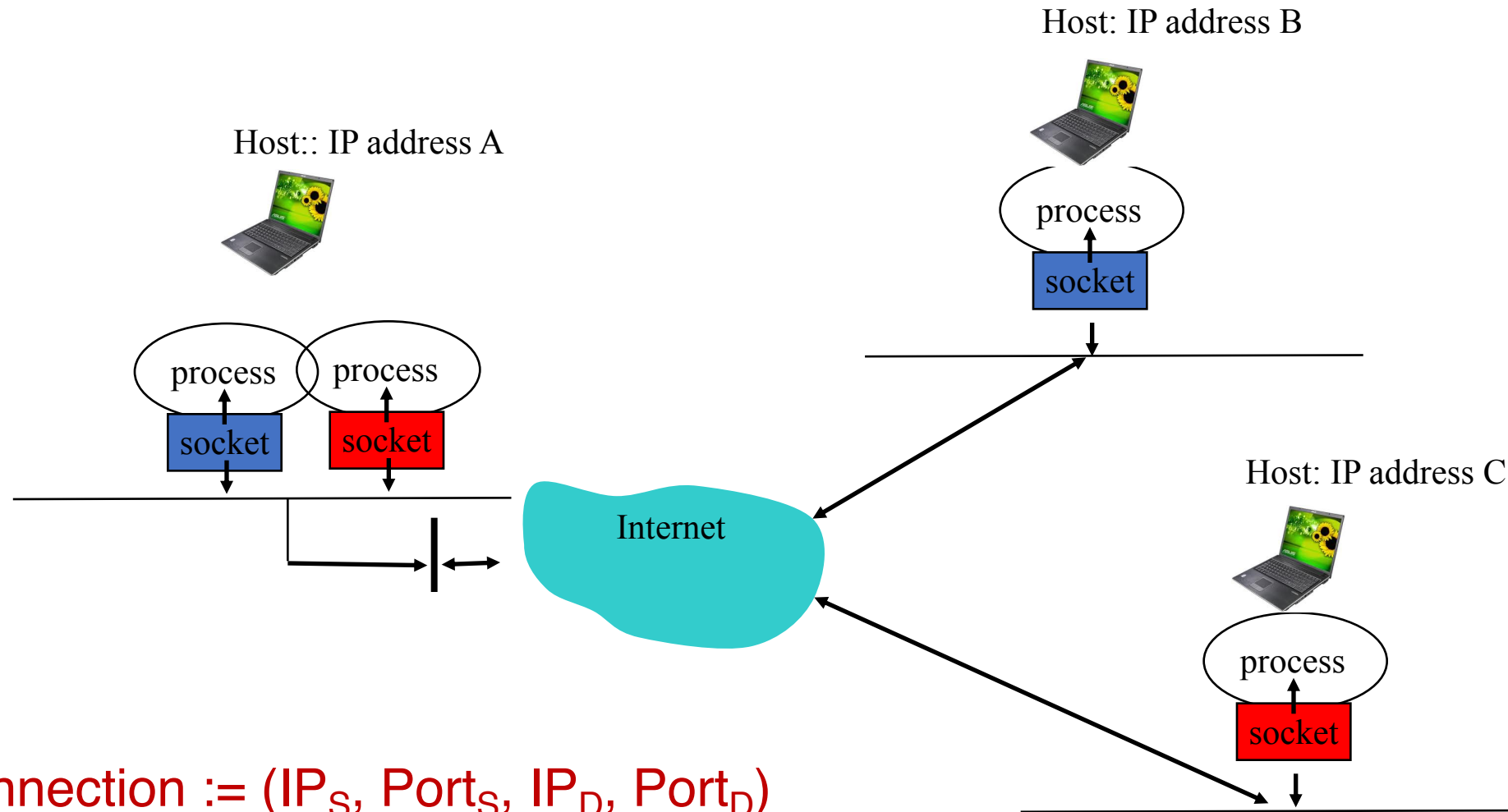
# IP address & port number



A **socket** is the door between OS/network and the application process

The **application's programming interface** to the network

# An app-layer connection is a 4-tuple

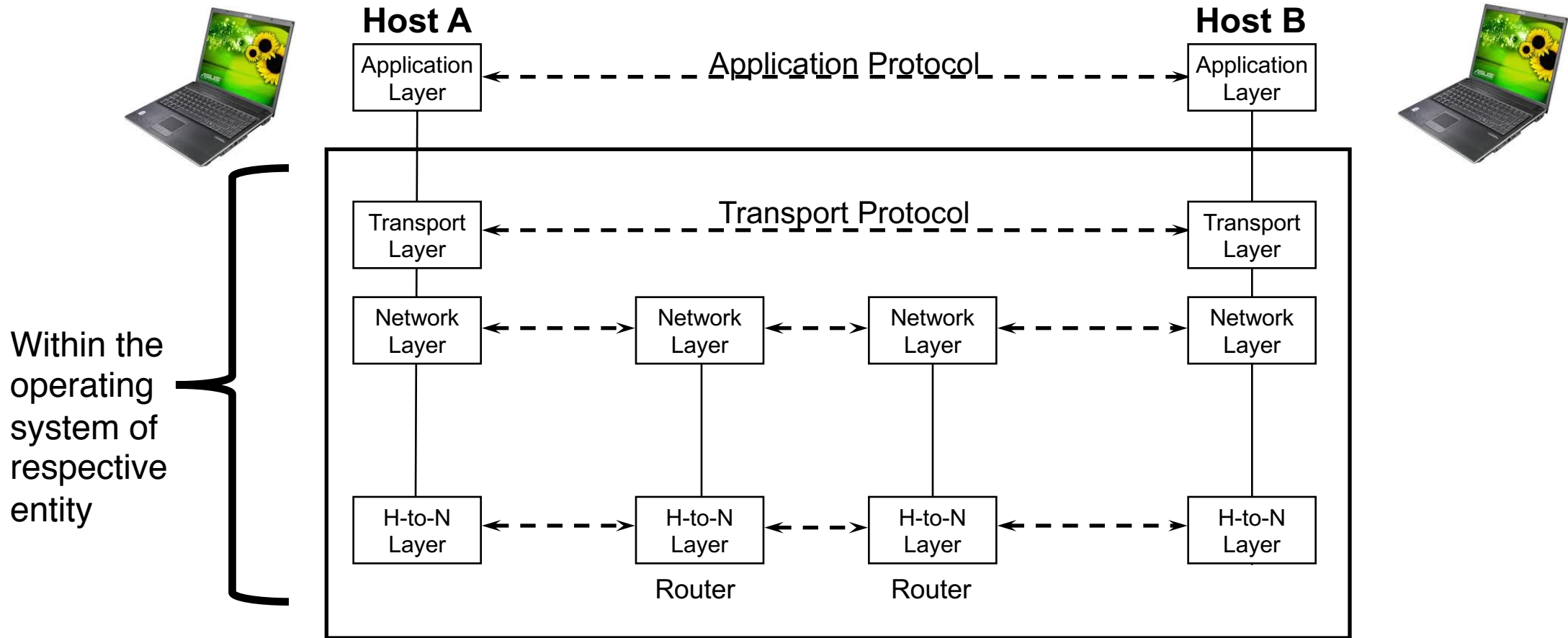


Connection := (IP<sub>S</sub>, Port<sub>S</sub>, IP<sub>D</sub>, Port<sub>D</sub>)  
(S = source, D = destination)

# App-layer connections

- A small demo

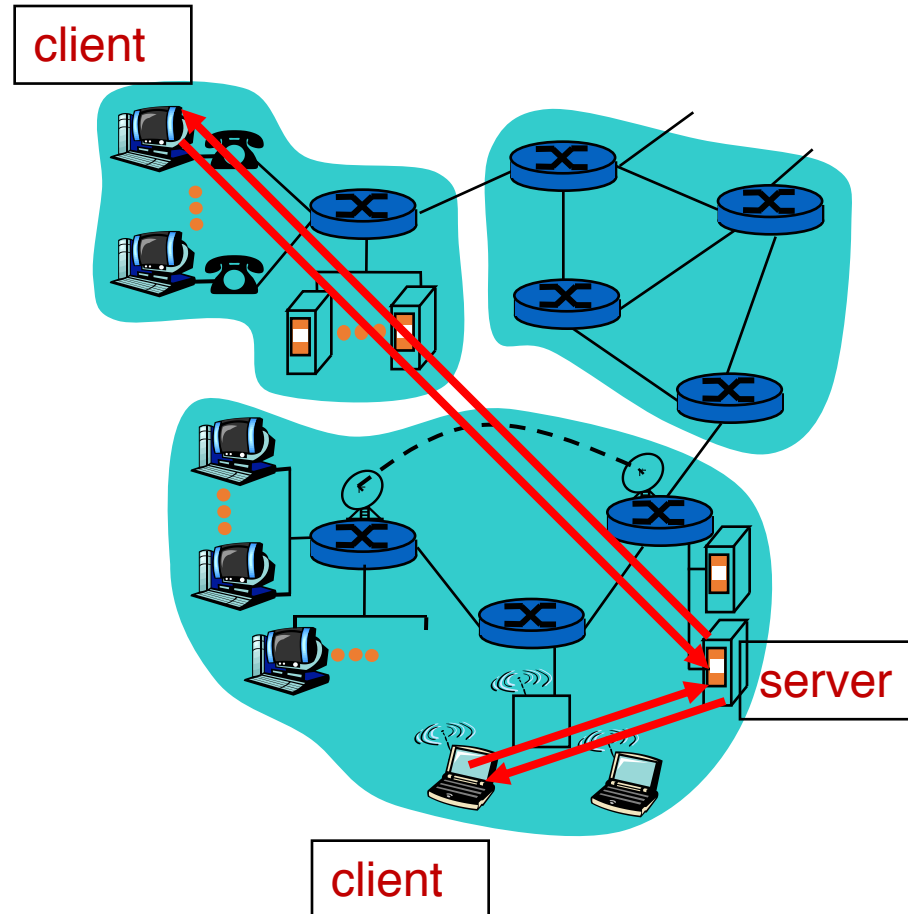
# Recall: Apps rely on services by lower layers



# Common Architectures of Applications



# Client-server architecture



## Server:

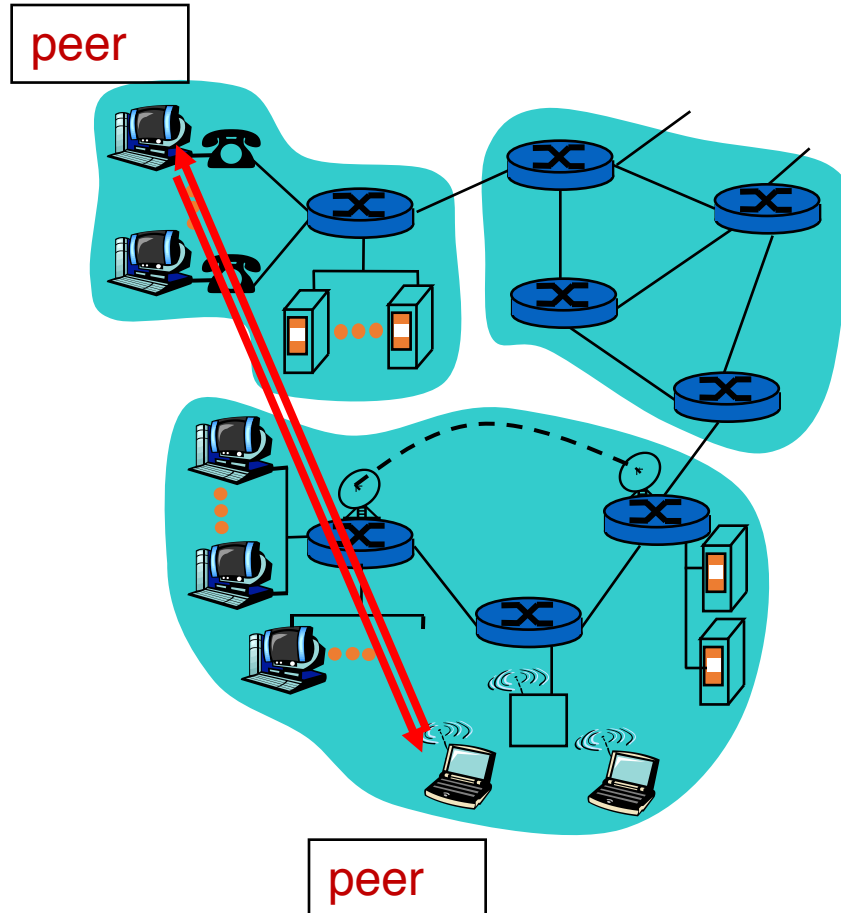
- always-on endpoint
- “permanent” IP address
- server farms (“data centers”) for scaling

## Clients:

- communicate with server
- may be intermittently connected
- may have dynamic IP addresses
- do not communicate directly with each other

- The web (HTTP) works this way.
- Many mobile apps work this way (e.g., Instagram)

# Peer-to-peer (P2P) architecture



- **Peers:**
  - Intermittently connected hosts
  - Directly talking to each other
- Little to no reliance on always-up servers
  - Examples: BitTorrent, Skype
- Today, many applications use a **hybrid** model
  - Example: Skype “supernodes”

# Going forward: A few applications

- Domain Name System
- The web: HTTP
- Mail
- File transfer



# CS 352

# Domain Name System

Lecture 3.2, Spring 2020

<http://www.cs.rutgers.edu/~sn624/352>

Srinivas Narayana

“You have my name. Can you  
lookup my address?”

# Domain Name System (DNS)

- Problem statement:

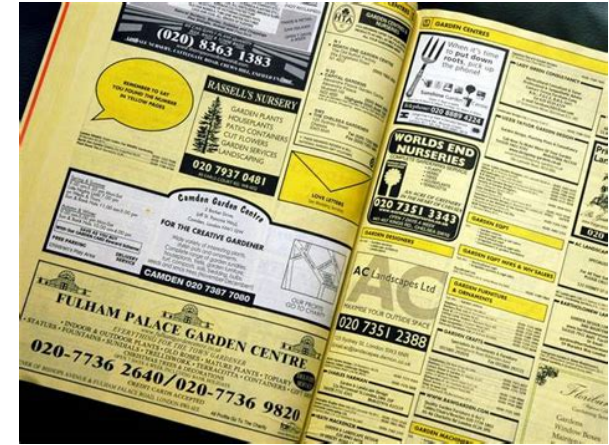
- Average brain can easily remember 7 digits for a few names
- On average, IP addresses have 12 digits
- We need an easier way to remember IP addresses

- Solution:

- Use alphanumeric names to refer to hosts. Called **host names** or **domain names**
  - Example: cs.rutgers.edu
- We need a **directory**: add a service to map between alphanumeric host names and binary IP addresses
- We call this process **Address Resolution**

# Types of Directories

- Directories map a *name* to an *address*
- Simplistic designs
  - Central directory
  - Ask everyone (e.g., flooding)
  - Tell everyone (e.g., push to a file like /etc/hosts)
- Scalable distributed designs
  - Hierarchical namespace (e.g., Domain Name System (DNS))
  - Flat name space (e.g., Distributed Hash Table)





# Simple DNS

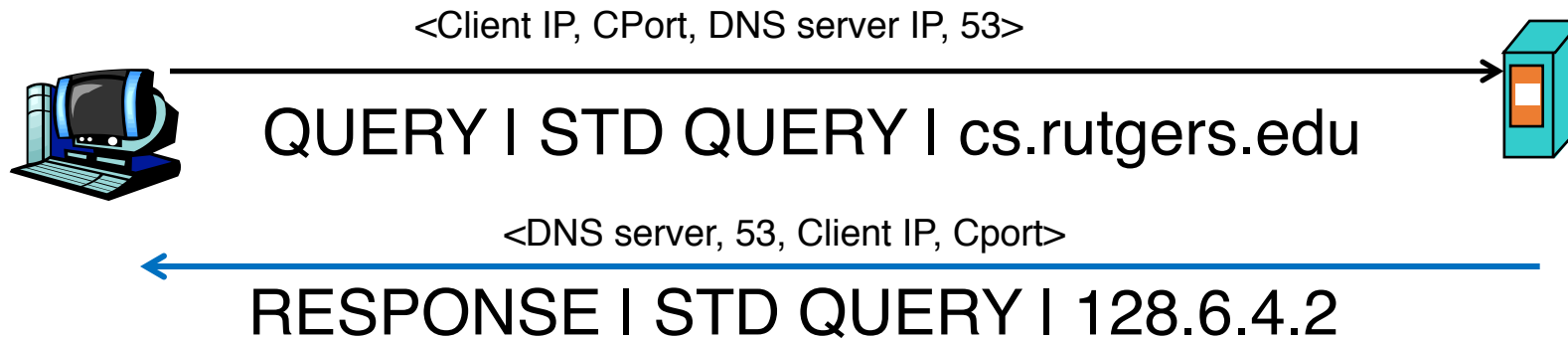
- What if every host has a local directory?
- /etc/hosts.txt
  - How things worked in the early days of the Internet!
- What if hosts moved around? How do you keep this up to date?

The image shows a scan of a directory page, likely from a Polish telephone or business directory. The text is arranged in columns and contains various entries with names, addresses, and phone numbers. A red circle highlights the entry for 'Spiro Gertrud'.

nowski Maciej Czerw. Kryst. tr 11 610 41	Zakład Ubezpie. Społecznych u. Biuro Zarządu. Socialversiche- rung	Spallinski Mieczyslaw Sniadek kuchnia 1 740 59	Spaltenstein Franciszek Lud. nastr 9 927 27	Sparkasse s. unter Kassel Sparkasse Holzindustrie GmbH Blumenstr 4 323 02	Spartaria Holzindustrie GmbH Madalinskistr 87 422 02	Spasinska Jadwiga Rakowiec- kastr 3 425 35	Spasowicz Eugeniusz 6 Sier- pienski 24 944 47	Spasowiczowa Aniela + Be- niamin Bedarskastr 26 238 95	Spaw Stahlkonstruktionswerke Kwiecinski Wl. Pradajnskistr 17 321 49	Specht Elzbieta Kurstr 108 Wohnung 10 23 49	Specht Willi Ingenieurbau- anstalt 900 89	Speck Paula Welo. u. Spiritus- schenklg Neue Welt 3 805 72	Ordenstr 19 633 14	Spedition Transportbüro Postpl 9 338 00	Speditionhaus Adolf u. Ede- ard Holler Zweigniederlassung Dlugastr 29 11 19 70	Spedo Sped.-Büro Marschallstr 70 692 59	Speich Walter + Ing. Kfm. Marsstr 8 738 24	Speldel Max Beauftragte d. Kom- missar. Verwaltung sichergestellt. Grundstücke 1. Warschau Grot- tengasse 2 426 35	Spel + elektr. Anl. u. Materialien- lager Bartoszowski M. Giesowski B. Wspolnast 9 734 57	Sperling J. & Co. Wapen u. Mo- tillwarenbr. GmbH Miynarska- str 30 253 59	Sperling Juliusz Kfm. Krasnast 10 946 28	Spiermaszyn. Eren. zast. Zywniast 20 636 39	Spisakowski Stanislaw Sniadek kuchnia 1 740 59	Spisakowski Stanislaw Sniadek kuchnia 1 740 59	Spiro Gertrud Verk. u. Spirit. u. Zigaretten Nowiniarskastr 2 11 00 21	Spiro Gertrud Geschäftsleh- rerin Dlugastr 18 224 04	Spiszczyk Wacław Ing. Arch. Potockastr 9 12 50 15	Spitsbarth-Benda Karol + Schauspieler Neue Welt 30 u. 248 76	Spiz Arbeitsgenossenschaft, Un- tern. 1. Tief- u. Hochbauarb. Kro- kazast 14 960 62	Spizowski Jan Zahnarzt Jawo- winskiast 7 723 12	Splawa-Neyman Helena Neue Burgstr 10 998 49	Splawa-Neyman Jan Ing.-Arch. Radomez Str 43 946 28	Spychalski Wit solim. Skazast 11	Spys Jan Nap Inh. techn. Hand- skast 1	Srebrny Kazimi- larsz 16 418 39	Srednicka Wlad Kosmetikerin F	Srednicki Br. M we Kolesstr 10	Srednicki Broni Lubi Wolkowstr 1	Srednicki Stani- sław Kindarzt Targow	Srednicki Stanis- lawstr 62	Srednicki Leon str 31	Sroeki Stefan Pi	Sroczyńska Apol str 20	Sroczyńska Iren	Sroczyńska Kar- boly. Dobrast 26	Sroczyński u. He- lmi. Nowolager u. schallstr 91	Sroczyński E. S Metallw. Abt. elek- trischer Str 4/4	Sroczyński J. & med. Laborat. E	Sroczyński Jan H ra-Konimiera-Str	Sroczyński Kar- Lecyast 4	Sroczyński Karo Lecyast 4	+ Grybowadzka	Sroczyński Kazim Kindarzt Sporta	Sroczyński Wlto str 2A
---	--	---	--	---	---	---	--	---	---	--	--	---	--------------------	--	--	--	---	---	---	---	---	--	---	---	--	--	--	--	---	--	--	---	-------------------------------------	--	------------------------------------	----------------------------------	-----------------------------------	-------------------------------------	--	--------------------------------	--------------------------	------------------	---------------------------	-----------------	-------------------------------------	--	--	------------------------------------	--------------------------------------	------------------------------	------------------------------	---------------	-------------------------------------	---------------------------

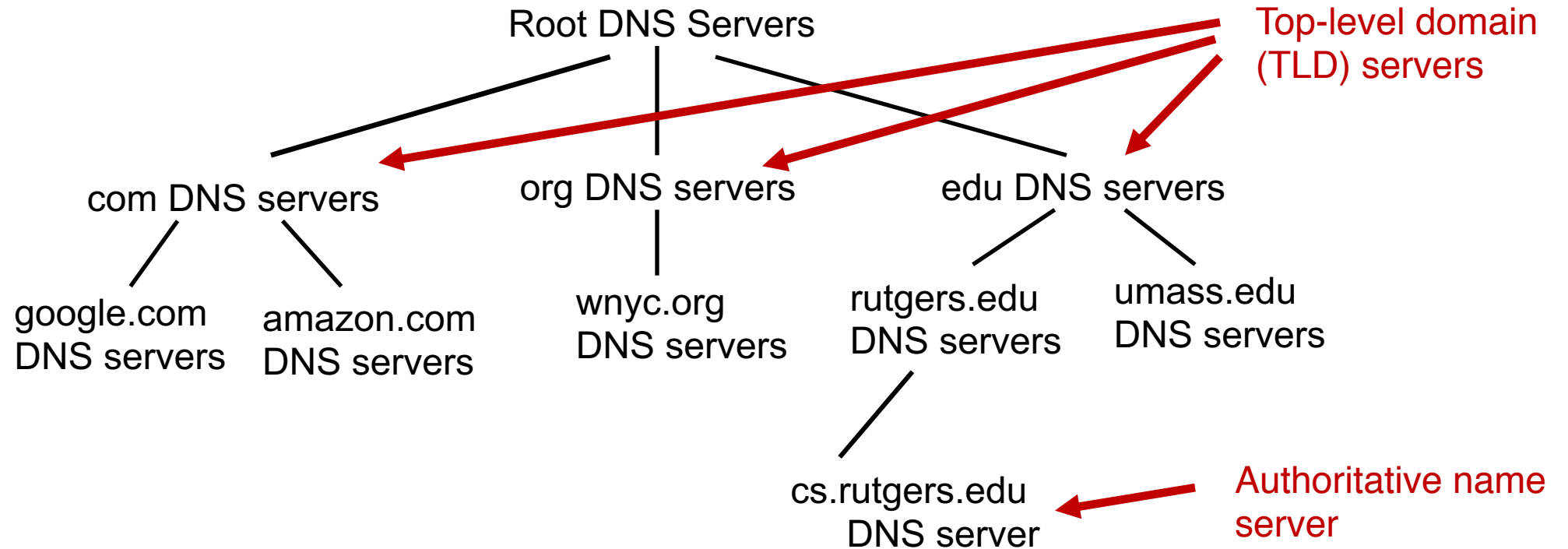
# Simple DNS

DOMAIN NAME	IP ADDRESS
www.yahoo.com	98.138.253.109
cs.rutgers.edu	128.6.4.2
www.google.com	74.125.225.243
www.princeton.edu	128.112.132.86



- Key idea: Implement a server that looks up a table.
- Will this scale?
  - Every new host needs to be entered in this table
  - Performance: can the server serve billions of Internet users
  - Failure: what if the server or the database crashes?
  - How to secure this server?

# Distributed and hierarchical database



RFC 1034: **Distribution through hierarchy enables scaling**

# DNS Protocol

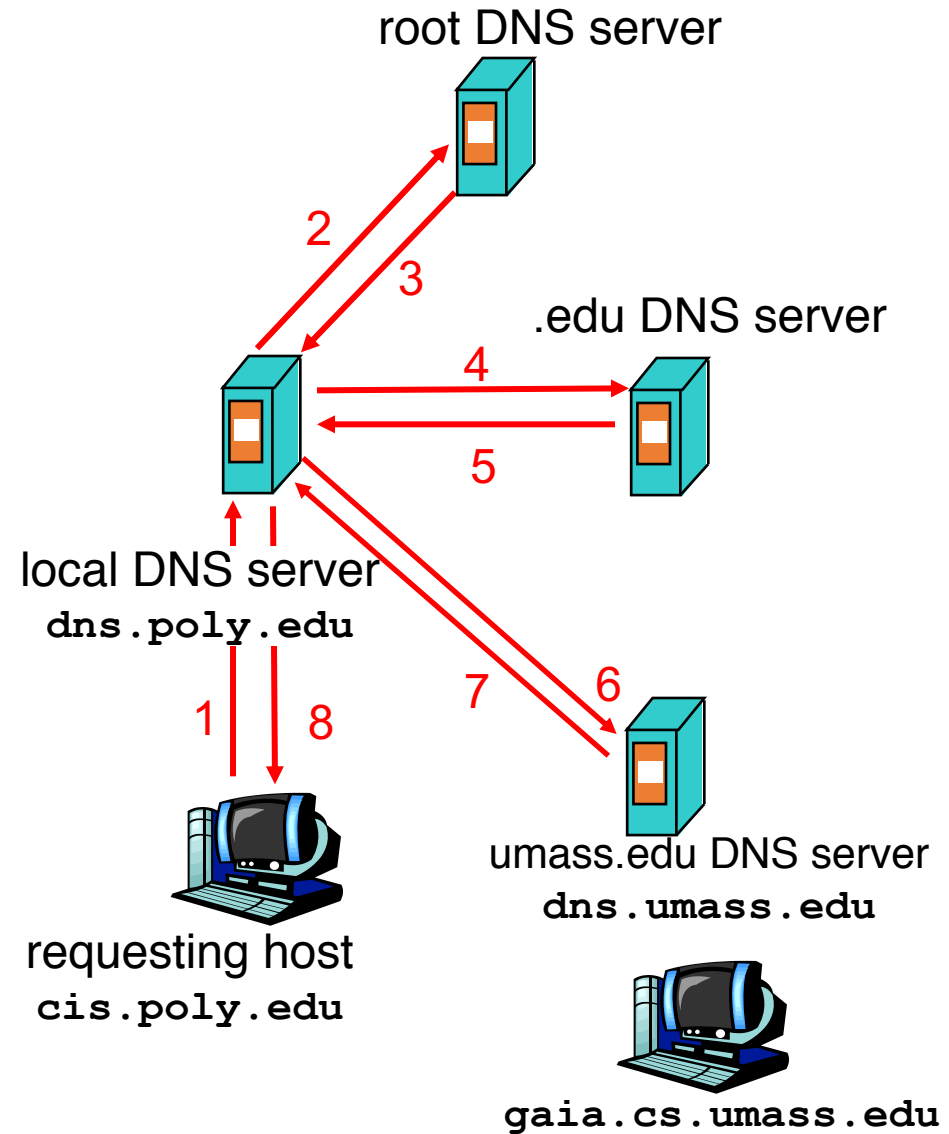
- Client and Server
- Client connects to Port 53 on server
- Assume DNS server IP known
- Two types of messages
  - Queries
  - Responses
- Type of Query (OPCODE) methods
  - Standard query (0x0)
    - e.g., Request IP address for a given domain name
  - Updates (0x5)
    - Provide a binding of IP address to domain name
- Each type has a common message format that follows the header

# DNS Protocol

- When client wants to know an IP address for a host name
  - Client sends a DNS query to the “local” name server in its network
  - If name server contains the mapping, it returns the IP address to the client
  - Otherwise, the name server forwards the request to the root name server
  - The request works its way down the tree toward the host until it reaches a name server with the correct mapping

# Example

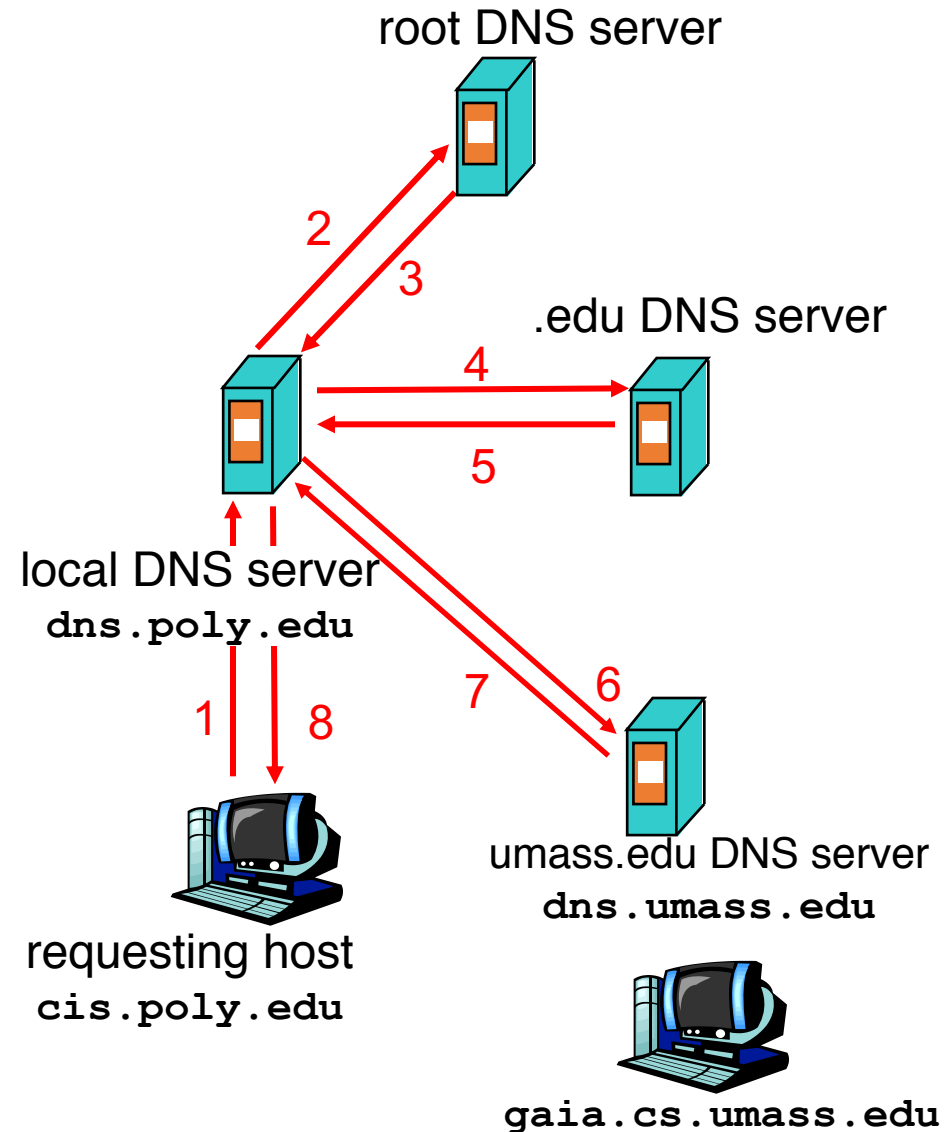
- Host at cis.poly.edu wants IP address for gaia.cs.umass.edu
- Local DNS server
- Root DNS server
- TLD DNS server
- **Authoritative** DNS server



# Query type

## Iterative query:

- Contacted server replies with name of server to contact
- “I don’t know this name, but ask this server”
- Queries are iterative for the local DNS server



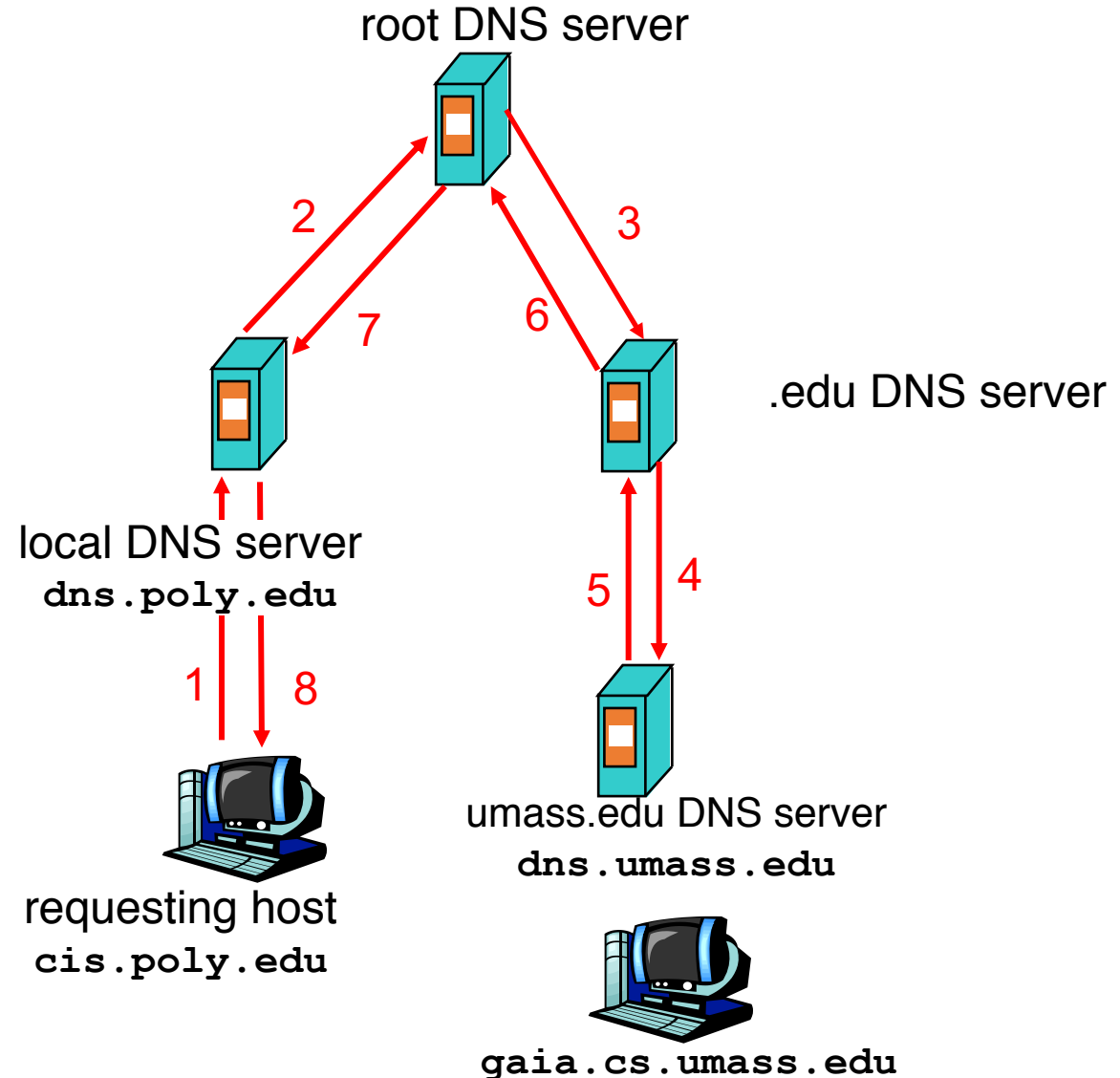
# Query type

## Recursive query:

- Puts burden of name resolution on the contacted name server

Problem: think about the root DNS server.

- Must it answer every DNS query?





# DNS in action

- A small demo



# CS 352

## DNS Records

Lecture 3.3, Spring 2020

<http://www.cs.rutgers.edu/~sn624/352>

Srinivas Narayana

# DNS records

DNS: distributed database storing resource records (RR)

RR format: (name, type, class, ttl, addr)

Type=A

- ❖ **name** is hostname
- ❖ **value** is IP address

Type=AAAA

- ❖ **name** is hostname
- ❖ **value** is IPv6 address

• Type=NS

- **name** is domain (e.g. foo.com)
- **value** is hostname of authoritative name server for this domain

Type=CNAME

- ❖ **name** is alias name for some “canonical” (the real) name  
e.g., www.ibm.com is really servereast.backup2.ibm.com
- ❖ **value** is canonical name

Type=MX

- ❖ **value** is name of mailserver associated with **name**

# DNS Record example

RRs in response  
to query



NAME	Design.cs.rutgers.edu
TYPE	A
CLASS	IN
TTL	1 day(86400)
ADDRESS	192.26.92.30

records for  
authoritative  
servers  
Information about  
nameserver



NAME	Cs.rutgers.edu
TYPE	NS
CLASS	IN
TTL	1 day(86400)
NSDNAME	Ns-lcsr.rutgers.edu

DNS serves as a general repository of information for the Internet!

# DNS record types

- A small demo

# DNS caching and updating records

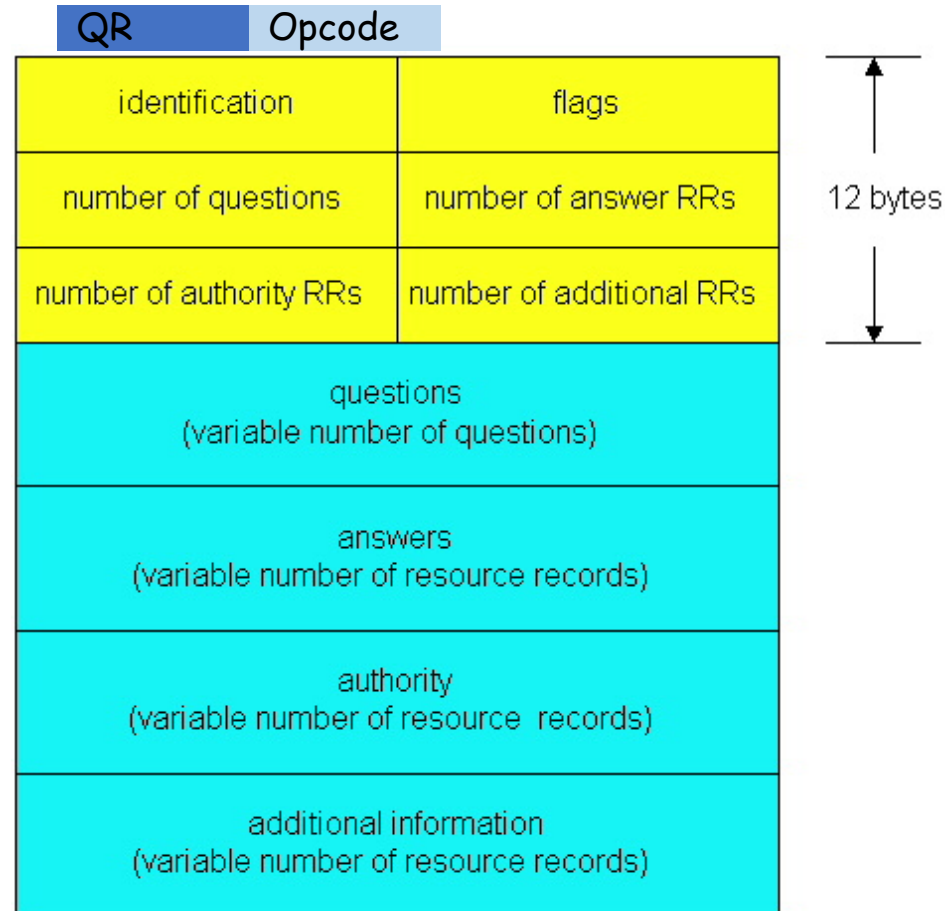
- Once (any) name server learns a name to IP address mapping, it *caches* the mapping
  - Cache entries timeout (disappear) after some time
  - TLD servers typically cached in local name servers
  - In practice, root name servers aren't visited often

# DNS protocol messages

DNS protocol: *query* and *reply* messages, both with same *message format*

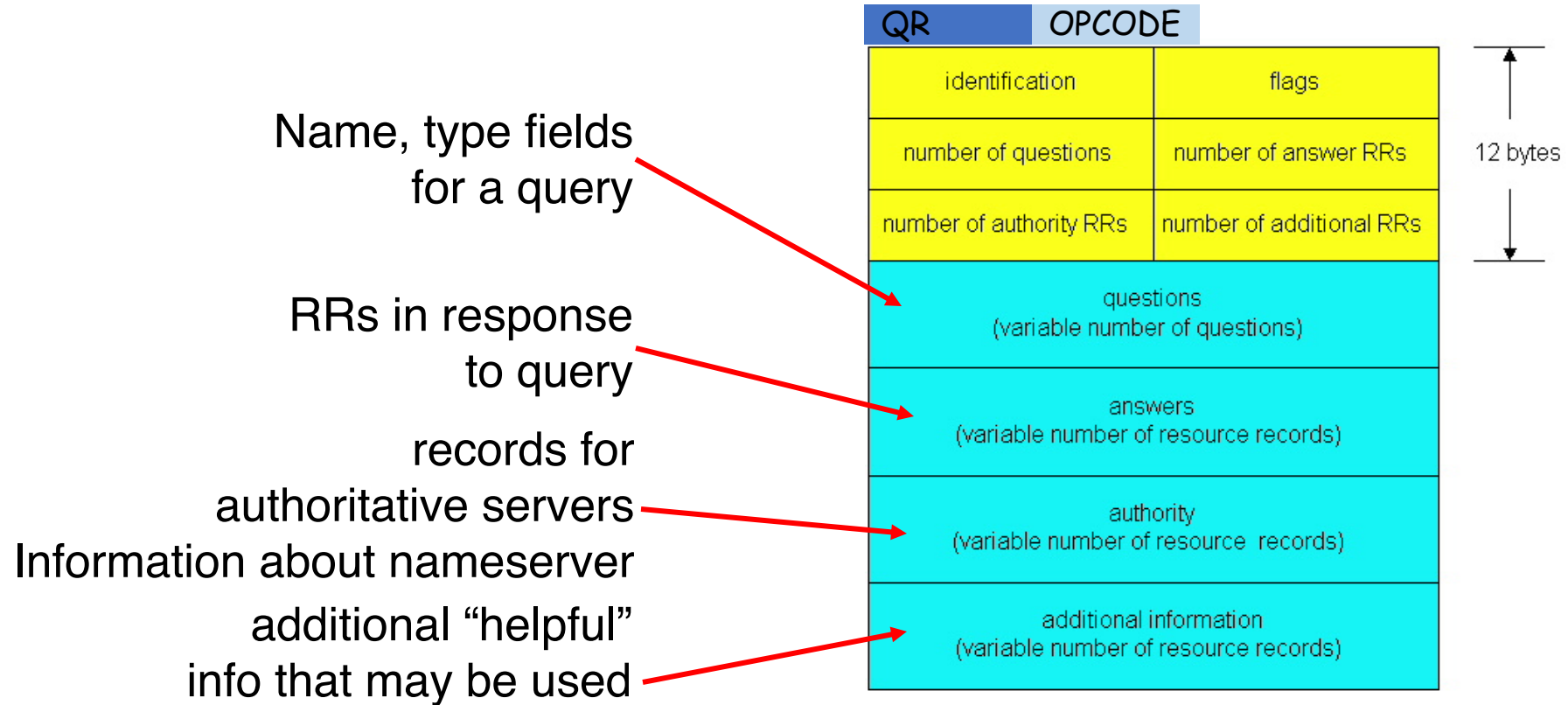
## Message header

- QR = 0 for Query, 1 for response
- Opcode= 0 standard
- **identification**: 16 bit # for query, reply to query uses same #
- **flags**:
  - Authoritative answer
  - recursion desired
  - recursion available
  - reply is authoritative





# DNS protocol, messages



# Bootstrapping DNS

- How does a host contact the name server if all it has is the domain name and no (name server) IP address?
- IP address of at least 1 nameserver (usually, a local resolver) must be known a priori
- The name server may be bootstrapped “statically”, e.g.,
  - File `/etc/resolv.conf` in unix
  - Start -> settings-> control panel-> network ->TCP/IP -> properties in windows
- ... or with another protocol!
  - **DHCP**: Dynamic Host Configuration Protocol (more on this later)

# Summary of DNS

- Hostname to IP address translation via a global network of servers
- Use Multiple layers of indirection
  - Hierarchically scale
  - Good performance (load distribution)
  - Resilient to local transient failure
- Additional load distribution can happen at each level (e.g., TLD server)
- Uses **caching** all over for better performance
  
- DNS can be used to implement useful primitives atop domain names:
- Example: Scaling large web services, e.g., google search
- Domain-authoritative server will return an address from a pool of IP addresses, for example from Google's server "farm"

# Some themes and observations on DNS

- Request/response nature of the protocol
- How messages are structured: simple, text-based protocol
  - Similarly in HTTP, SMTP, FTP
- Caching is an effective method to improve performance