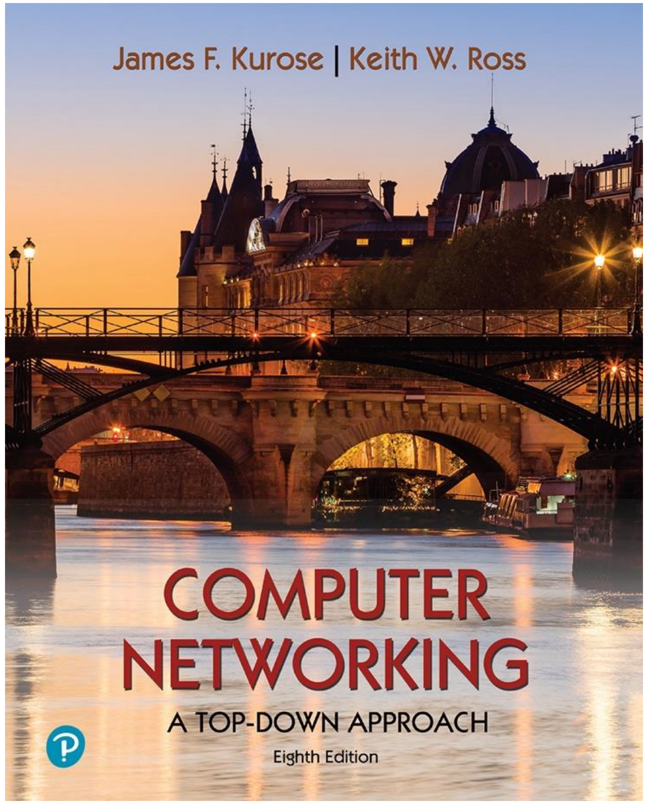
Chapter 2
Application Layer

Yaxiong Xie

Department of Computer Science and Engineering
University at Buffalo, SUNY

Adapted from the slides of the book's authors



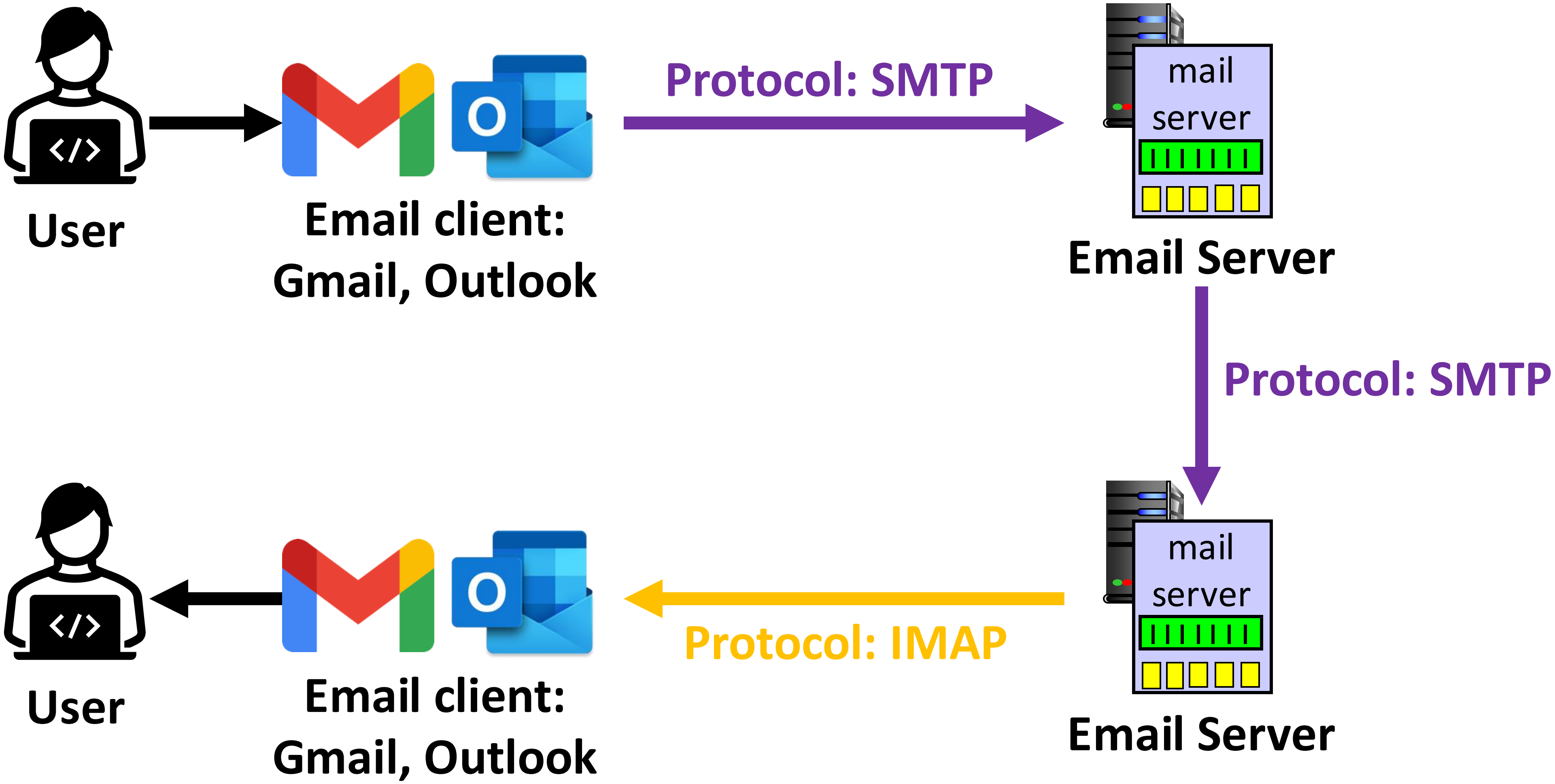
Computer Networking: A Top-Down Approach
8th edition n
Jim Kurose, Keith Ross
Pearson, 2020

Application layer: overview

- Principles of network applications
- socket programming with UDP and TCP
- Web and HTTP
- E-mail, SMTP, IMAP
- The Domain Name System DNS
- P2P applications
- video streaming and content distribution networks

E-mail: work flow

SMTP: Simple Mail Transfer Protocol

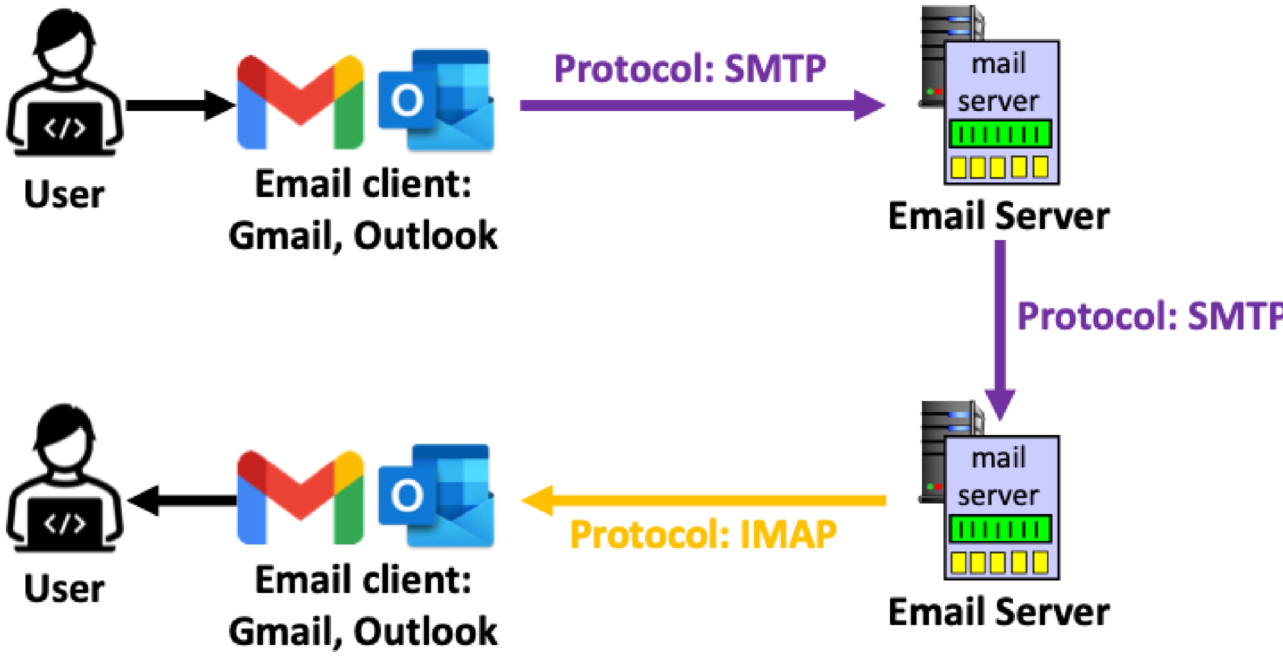


IMAP: Internet Message Access Protocol

E-mail: work flow

Three major components:

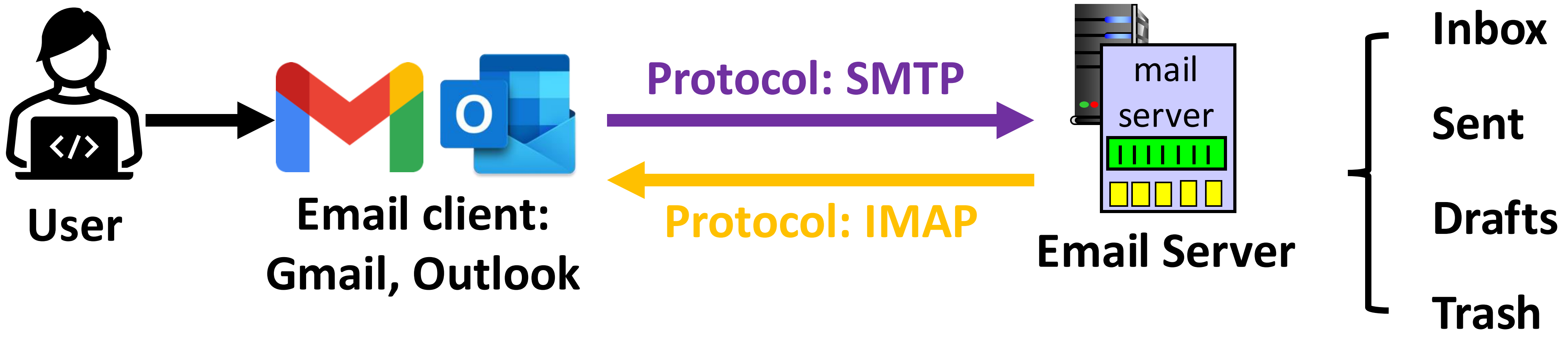
- Mail client
- Mail server
- Protocols



SMTP: Simple Mail Transfer Protocol

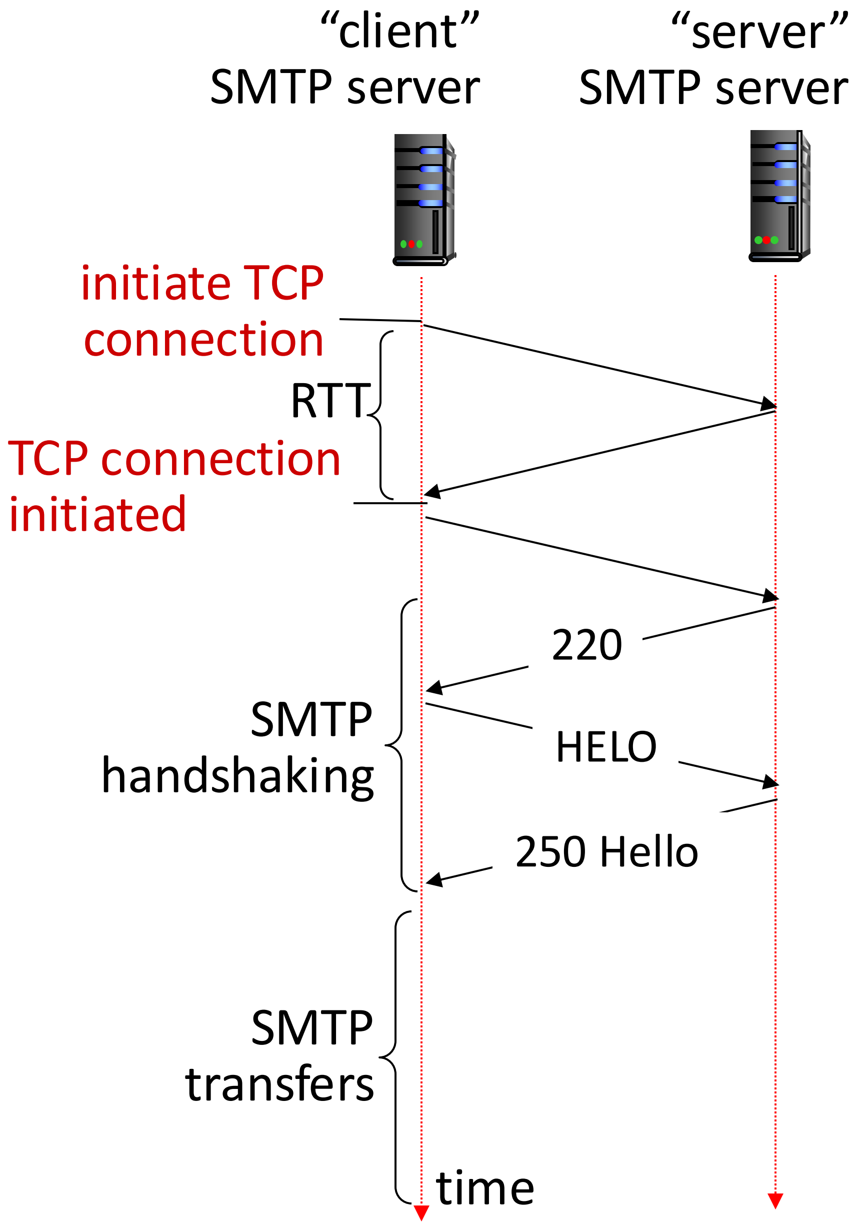
IMAP: Internet Message Access Protocol

E-mail: mailbox

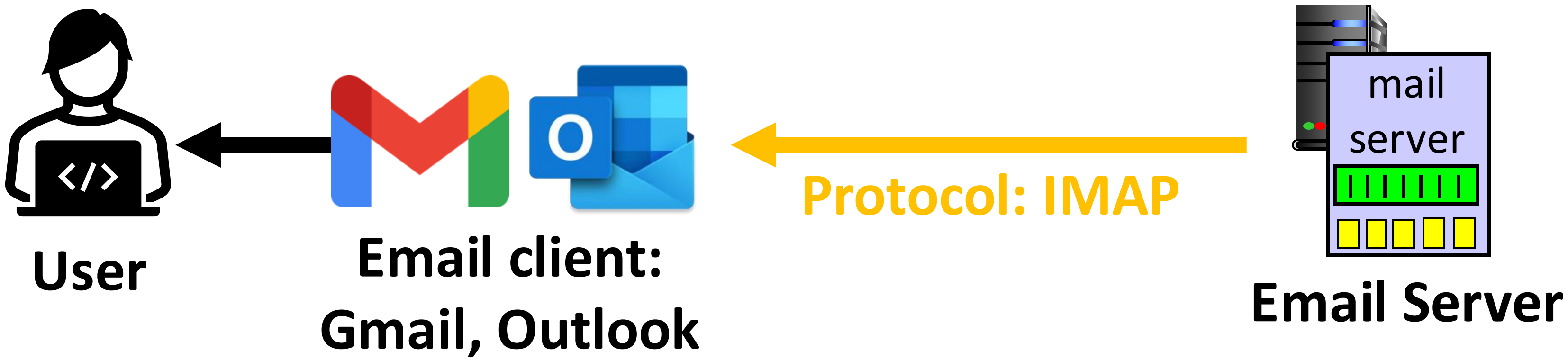


SMTP RFC (5321)

- uses TCP to reliably transfer email messages from client (mail server initiating connection) to server, port 25
  - direct transfer: sending server (acting like client) to receiving server
- three phases of transfer
  - SMTP handshaking (greeting)
  - SMTP transfer of messages
  - SMTP closure
- command/response interaction (like HTTP)
  - commands: ASCII text
  - response: status code and phrase



Retrieving email: mail access protocols

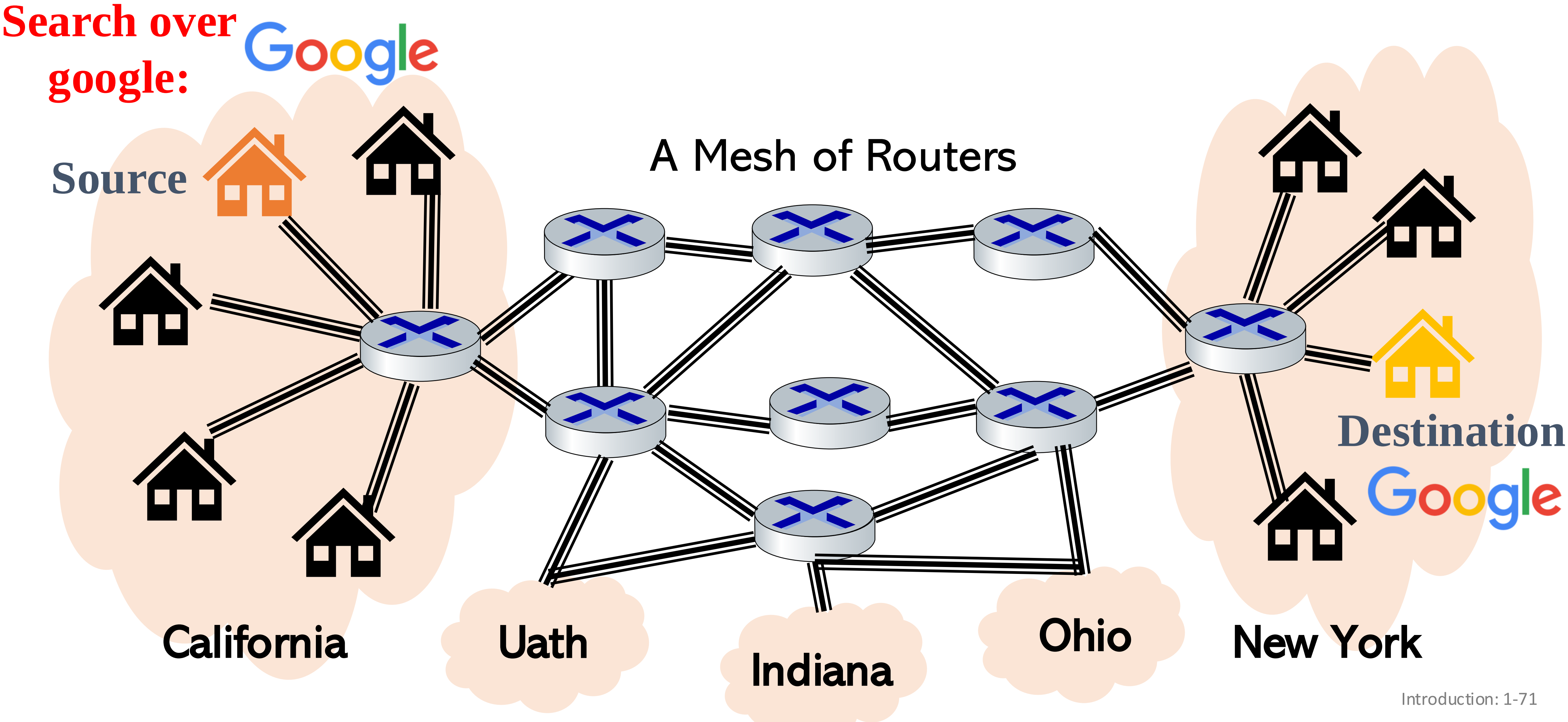


- mail access protocol: retrieval from server
  - IMAP: Internet Mail Access Protocol [RFC 3501]: messages stored on server, IMAP provides retrieval, deletion, folders of stored messages on server

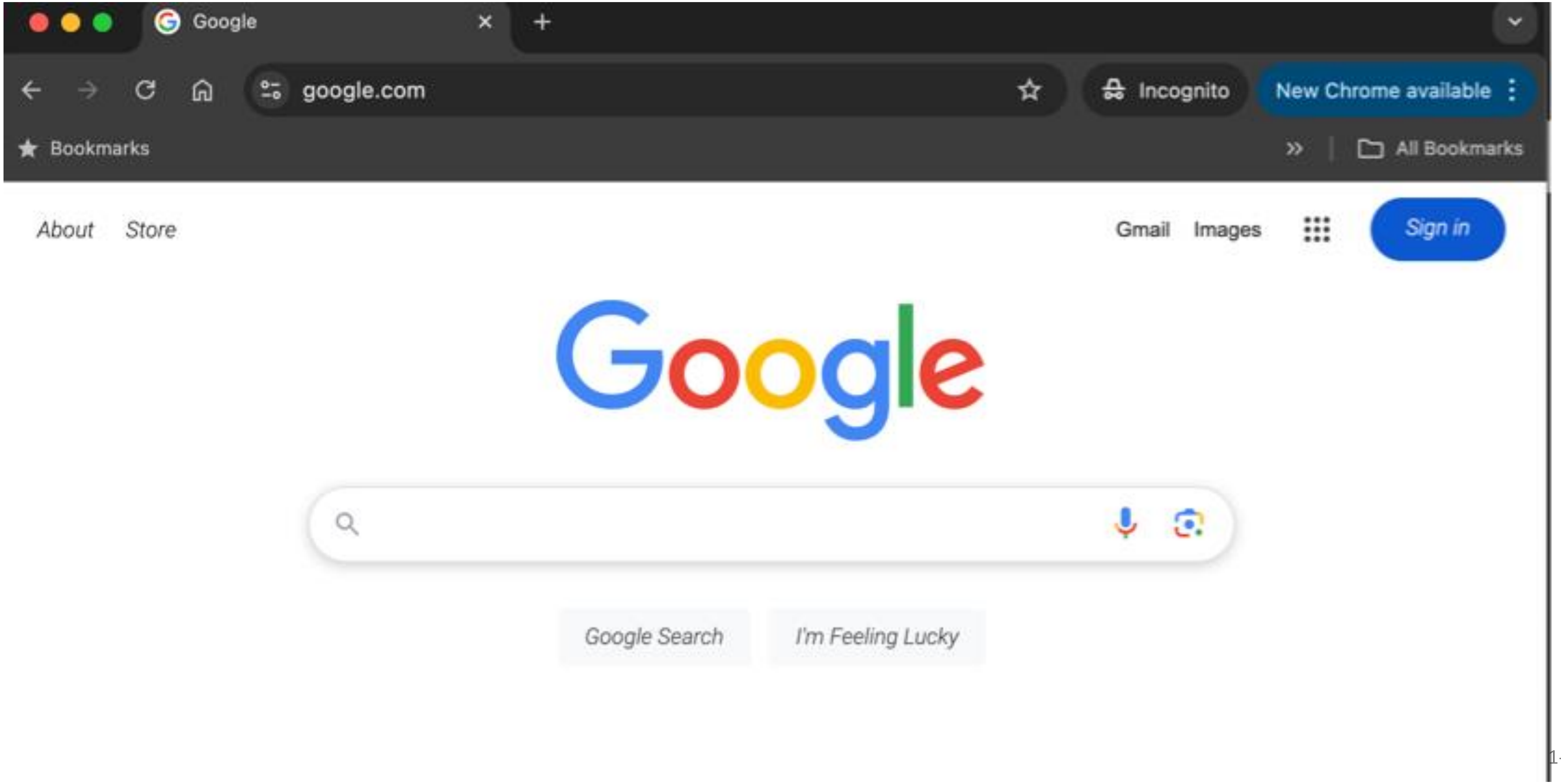
Application Layer: Overview

- Principles of network applications
- Web and HTTP
- E-mail, SMTP, IMAP
- The Domain Name System DNS
- P2P applications
- video streaming and content distribution networks
- socket programming with UDP and TCP

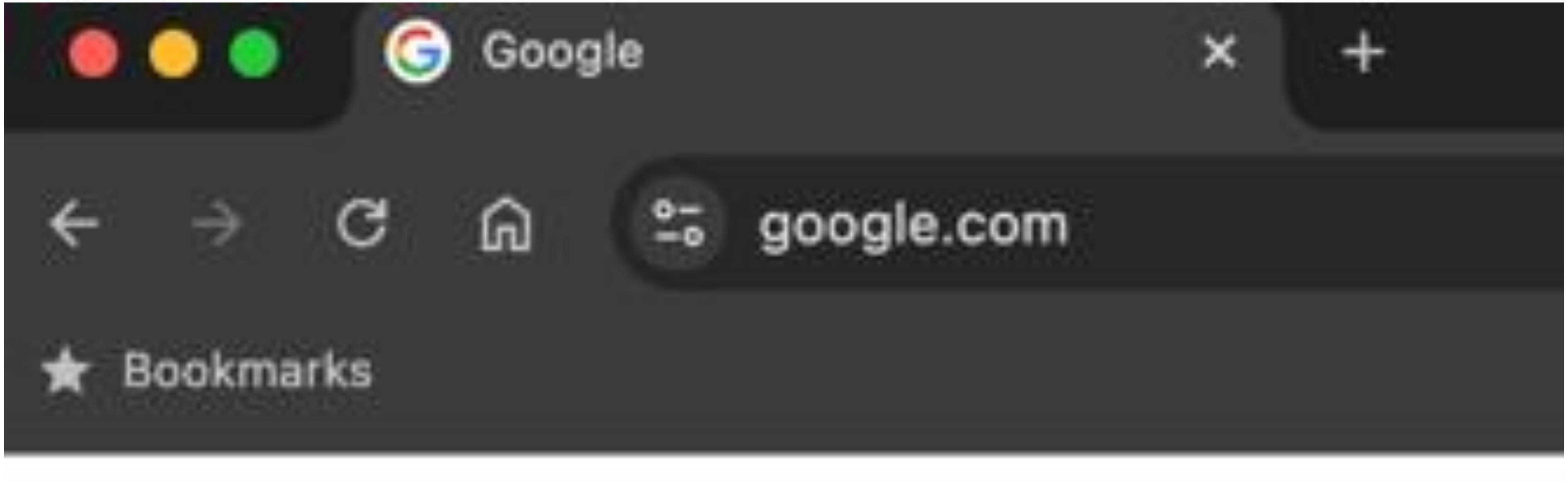
DNS: Domain Name System



DNS: Domain Name System



DNS: Domain Name System



We input the domain name instead of the IP address of the server!

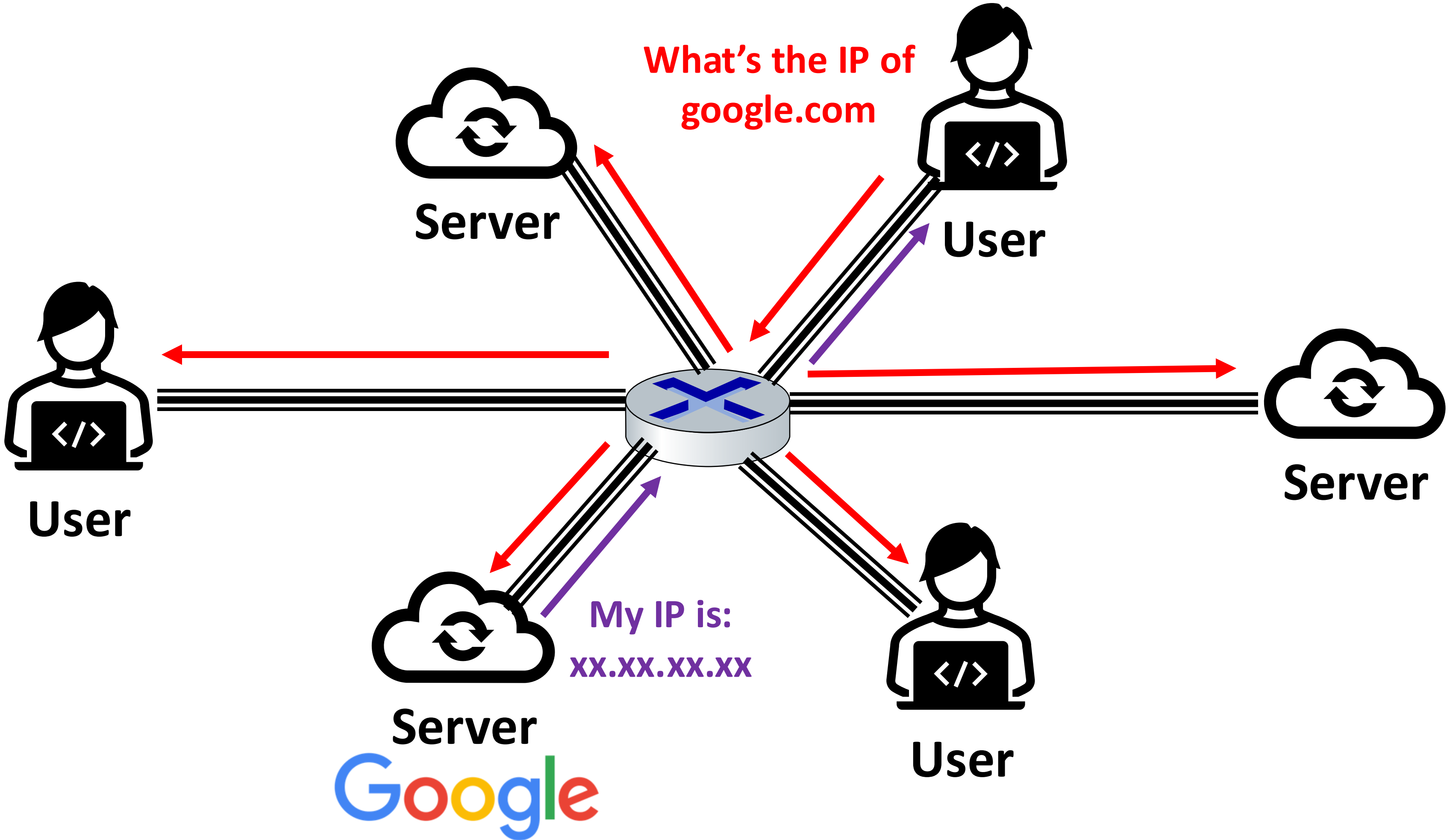
DNS: Domain Name System

domain name —**Mapping**→ IP address

Task of DNS system: Mapping the domain to it’s corresponding IP address

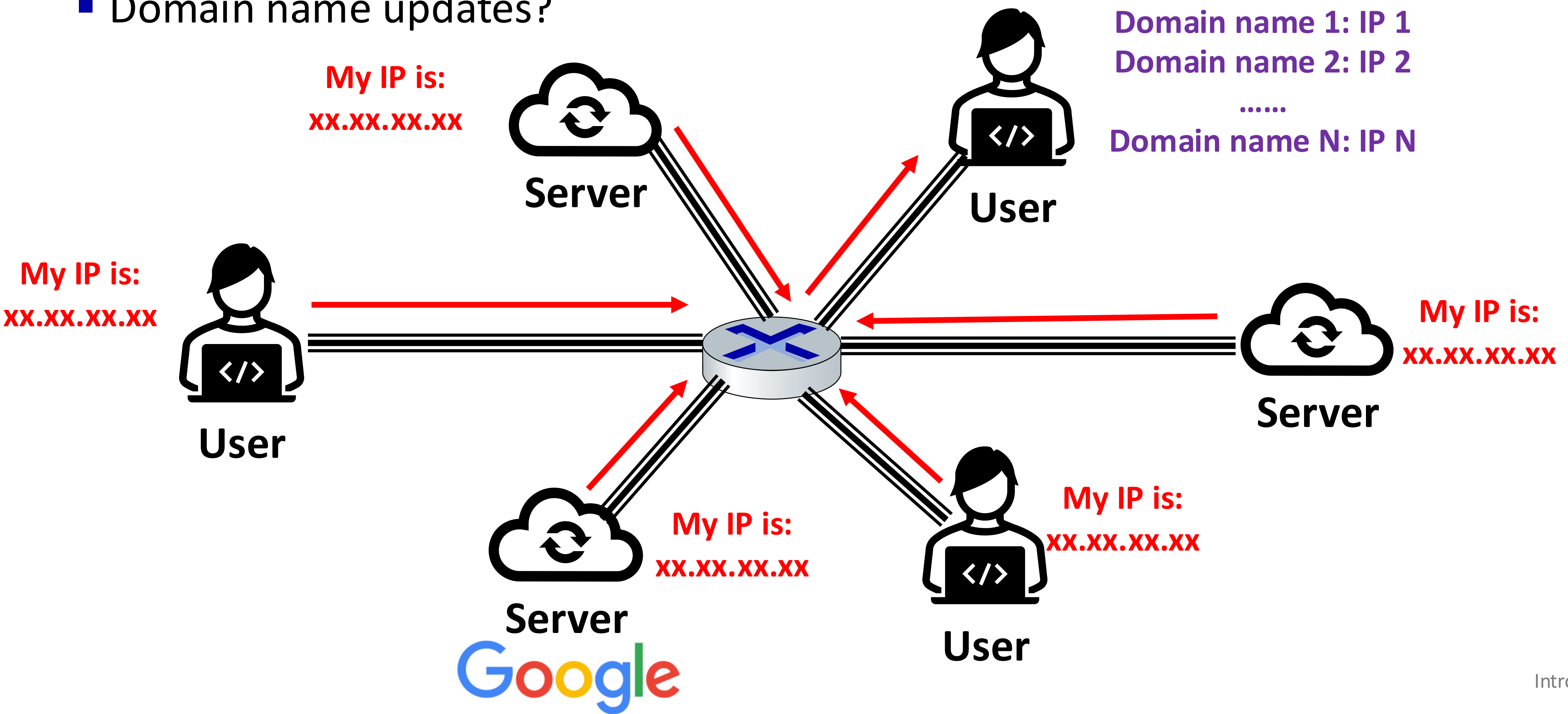
DNS: three extremes

- Flooding the query: the server responds with its IP address



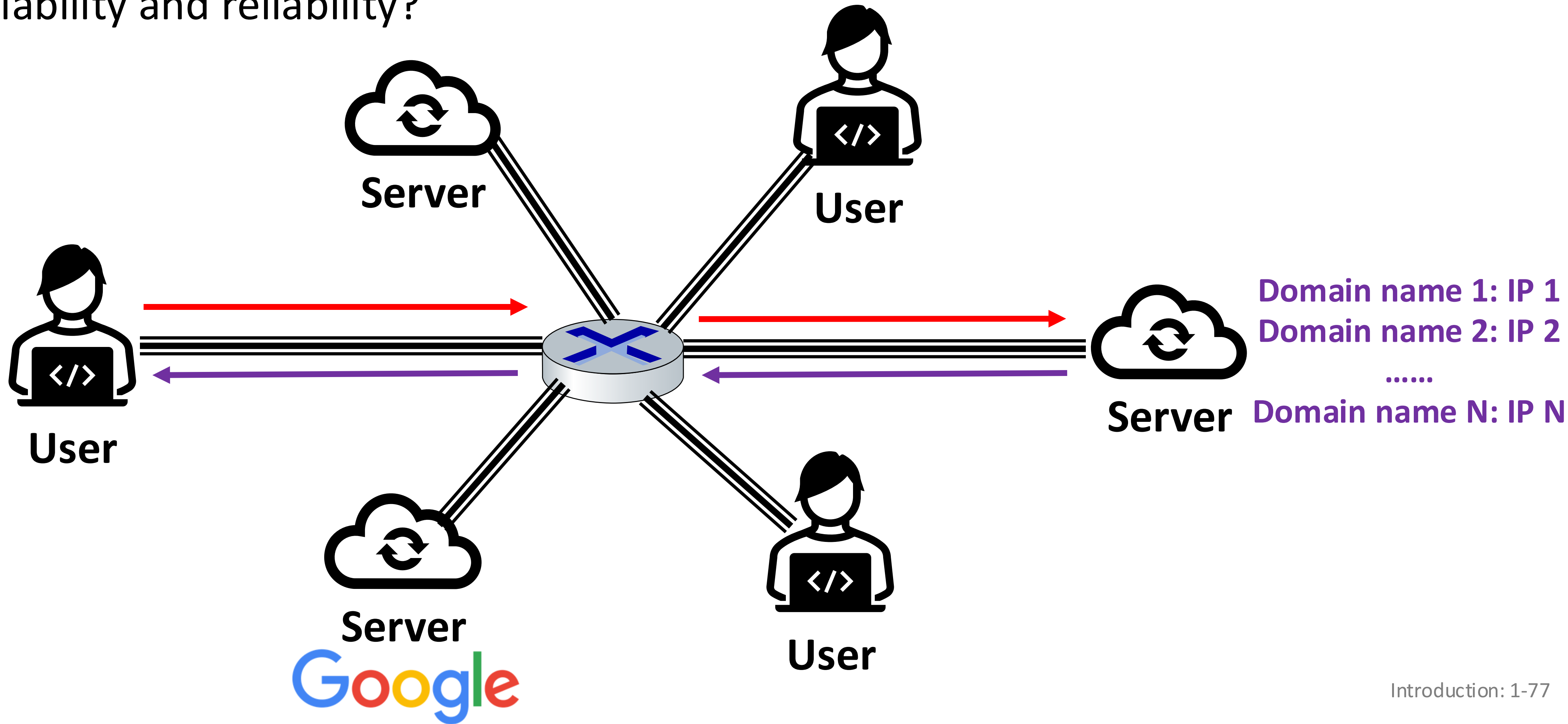
DNS: three extremes

- Push data to all devices: all devices stores a full copy of all the mappings
  - Domain name updates?



DNS: three extremes

- Central DNS server: All data and queries handled by one machine
  - Scalability and reliability?



Thinking about the DNS

Size: humongous distributed database:

- ~ billion records, each simple

Performance: handles many *trillions* of queries/day:

- *many* more reads than writes
- *performance matters:* almost every Internet transaction interacts with DNS - msecs count!

Geographical distribution: organizationally, physically decentralized:

- millions of different organizations responsible for their records

"bulletproof": reliability, security

Thinking about the DNS

Size: humongous distributed database:

- ~ billion records, each simple

Performance: handles many *trillions* of queries/day:

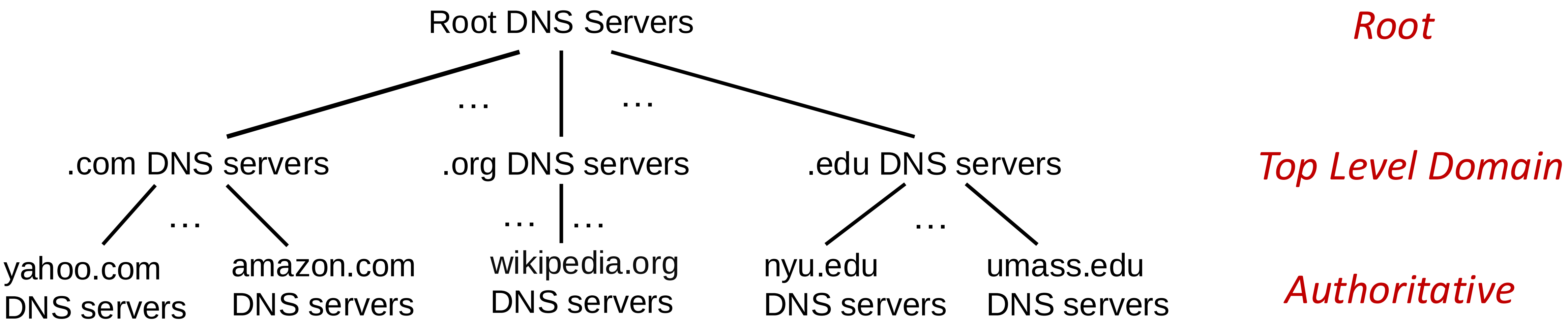
- *many* more reads than writes
- *performance matters:* almost every Internet transaction interacts with DNS - msecs count!

Geographical distribution: organizationally, physically decentralized:

- millions of different organizations responsible for their records

"bulletproof": reliability, security

DNS: a distributed, hierarchical database

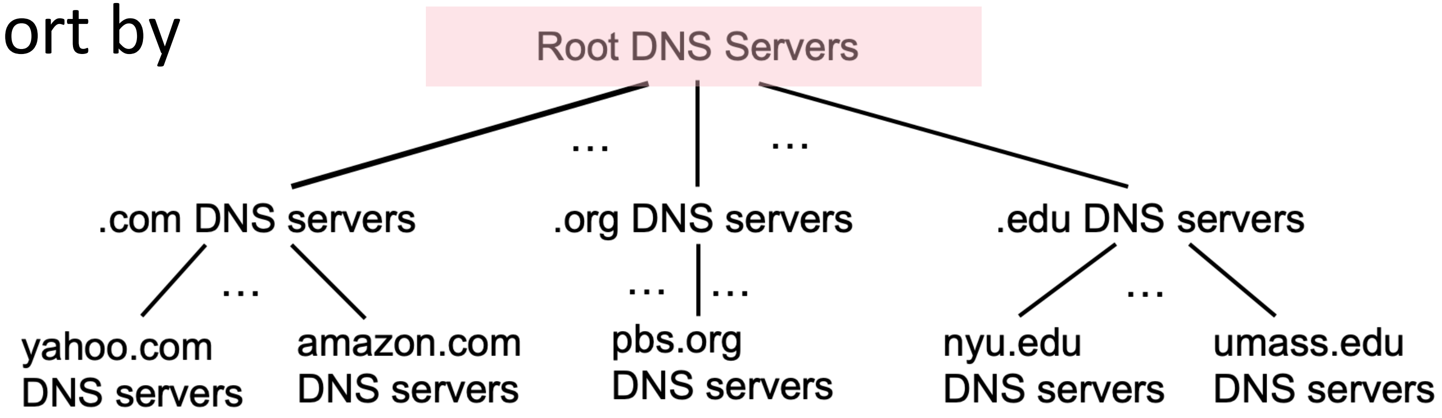


Client wants IP address for www.amazon.com; 1st approximation:

- client queries root server to find .com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for www.amazon.com

DNS: root name servers

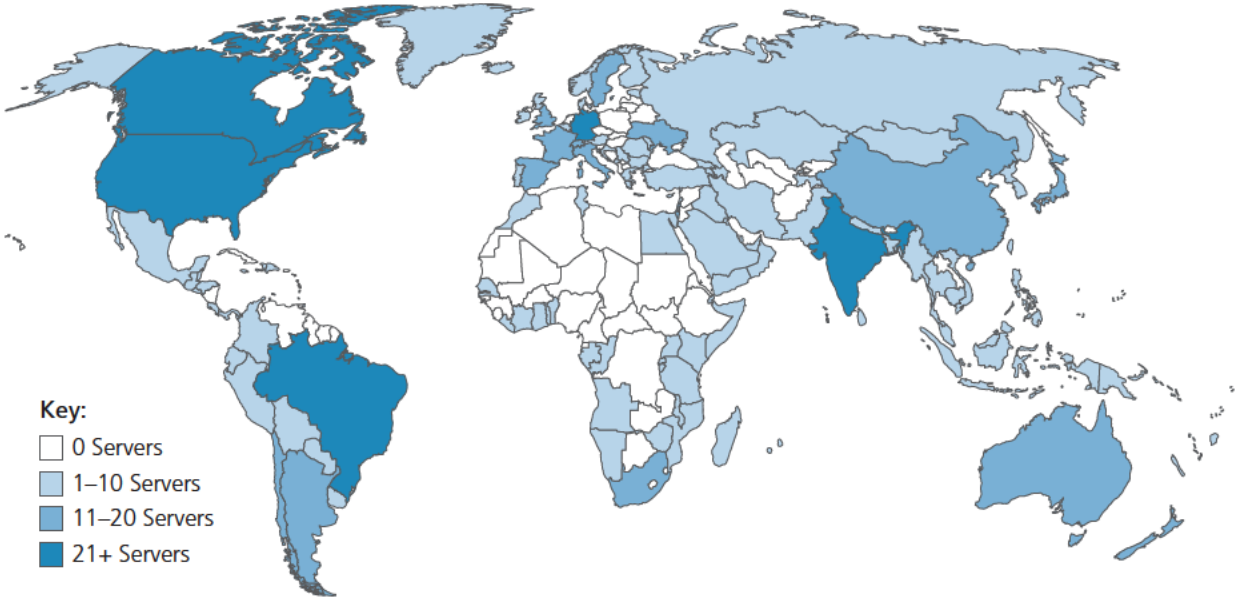
- official, contact-of-last-resort by name servers that can not resolve name



DNS: root name servers

- official, contact-of-last-resort by name servers that can not resolve name
- *incredibly important* Internet function
  - Internet couldn't function without it!

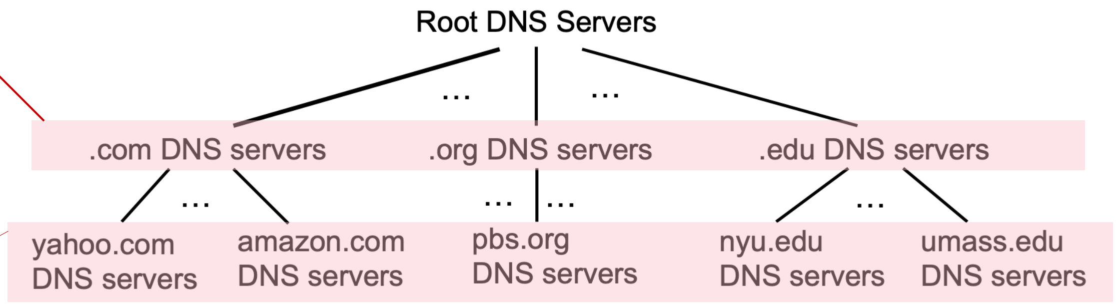
13 logical root name "servers" worldwide each "server" replicated many times (~200 servers in US)



Top-Level Domain, and authoritative servers

Top-Level Domain (TLD) servers:

- responsible for .com, .org, .net, .edu, .aero, .jobs, .museums, and all top-level country domains, e.g.: .cn, .uk, .fr, .ca, .jp
- Network Solutions: authoritative registry for .com, .net TLD
- Educause: .edu TLD

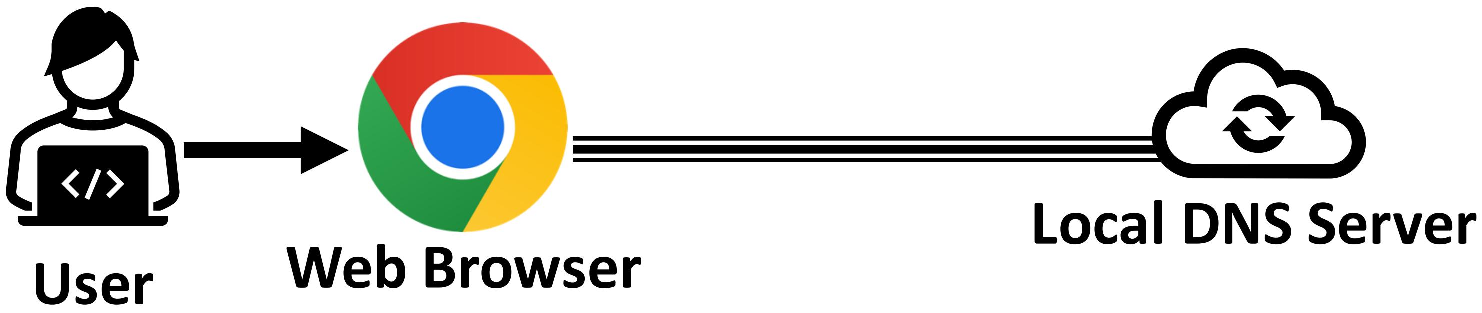


authoritative DNS servers:

- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider

Local DNS name servers

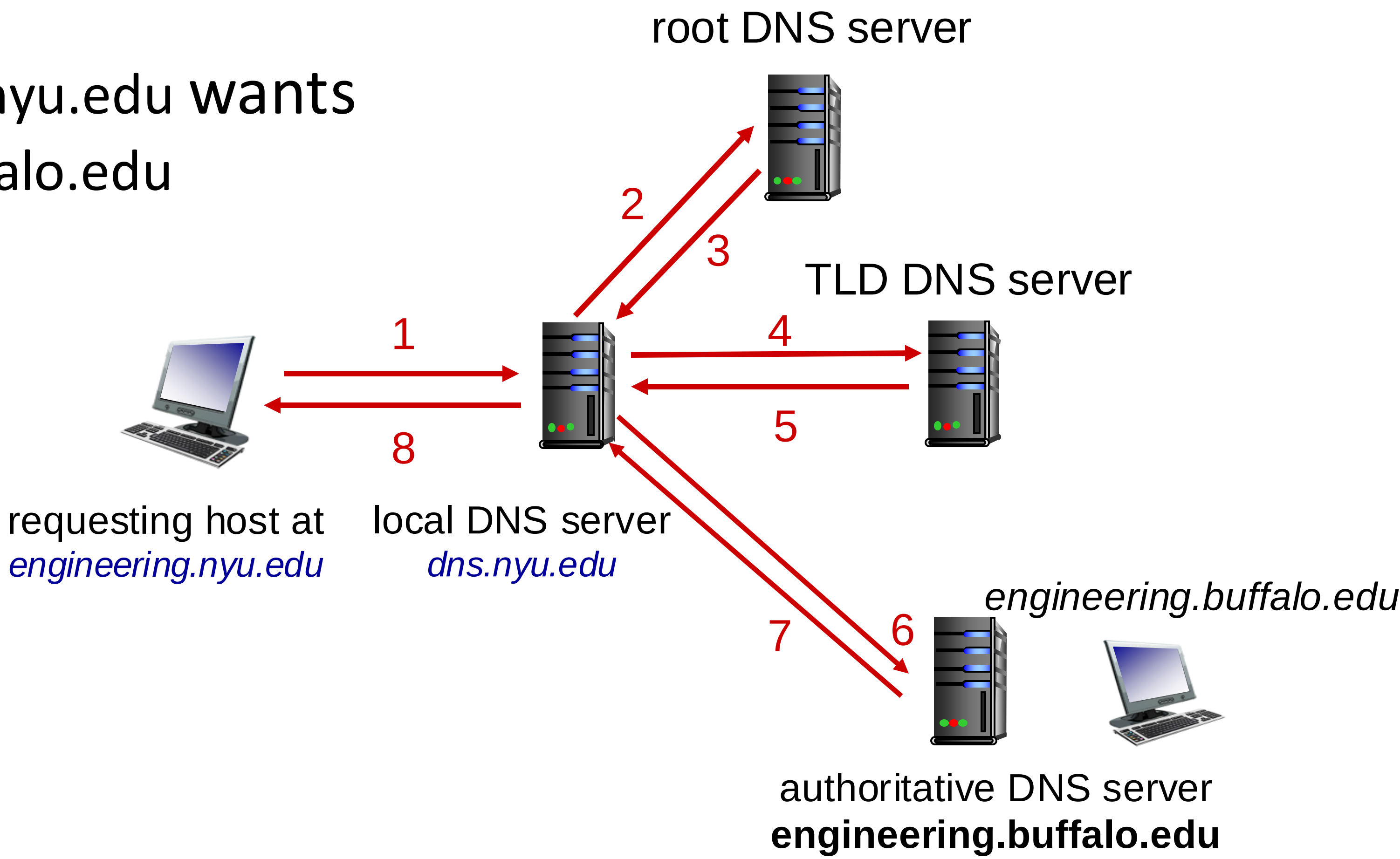
- when host makes DNS query, it is sent to its *local* DNS server
  - Local DNS server returns reply, answering:
    - from its local cache of recent name-to-address translation pairs (possibly out of date!)
    - forwarding request into DNS hierarchy for resolution
  - each ISP has local DNS name server; to find yours:
    - MacOS: `% scutil --dns`
    - Windows: `>ipconfig /all`



DNS name resolution: iterated query

Example: host at engineering.nyu.edu wants IP address for engineering.buffalo.edu

Iterated query:
- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"

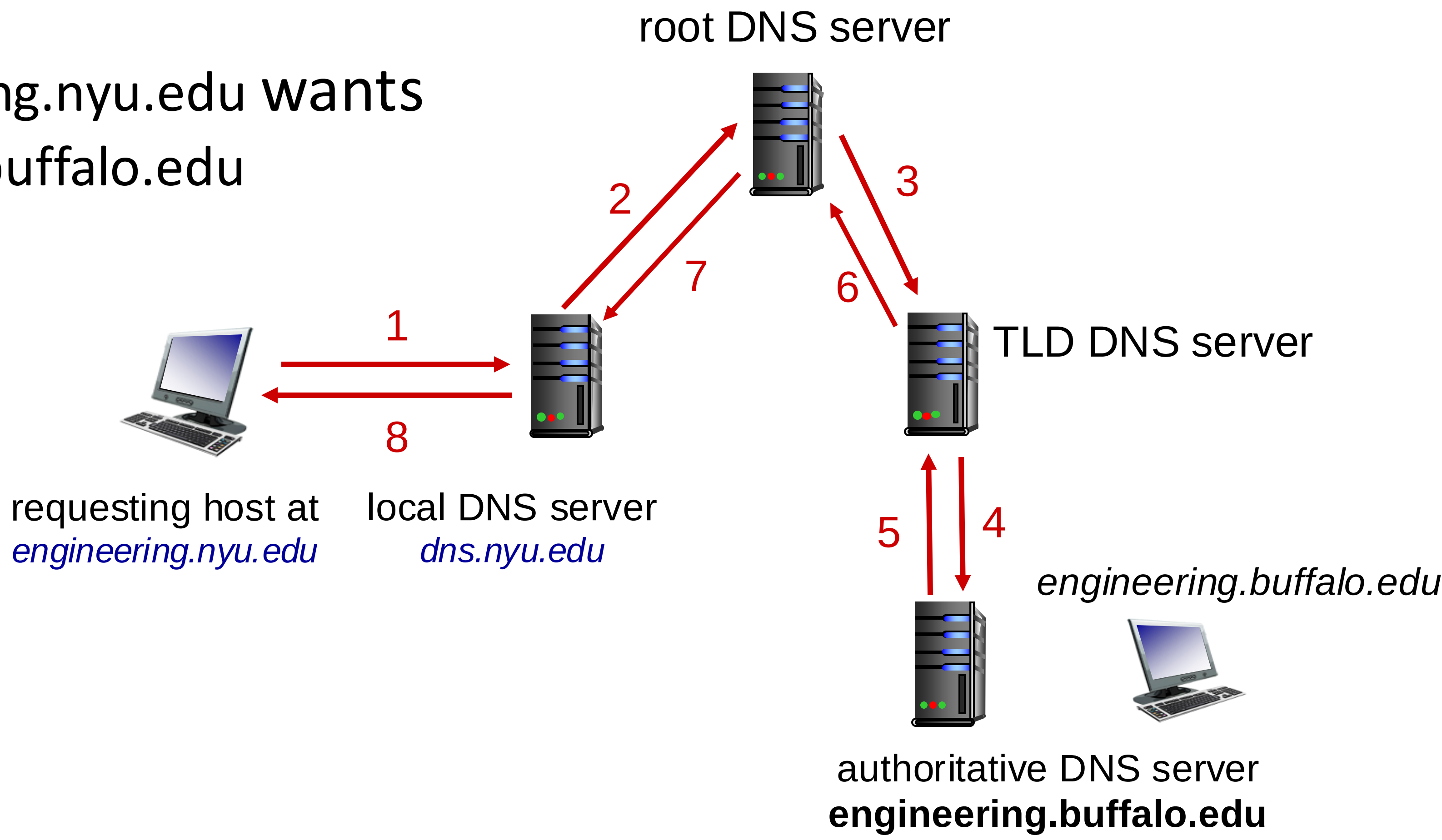


DNS name resolution: recursive query

Example: host at engineering.nyu.edu wants IP address for engineering.buffalo.edu

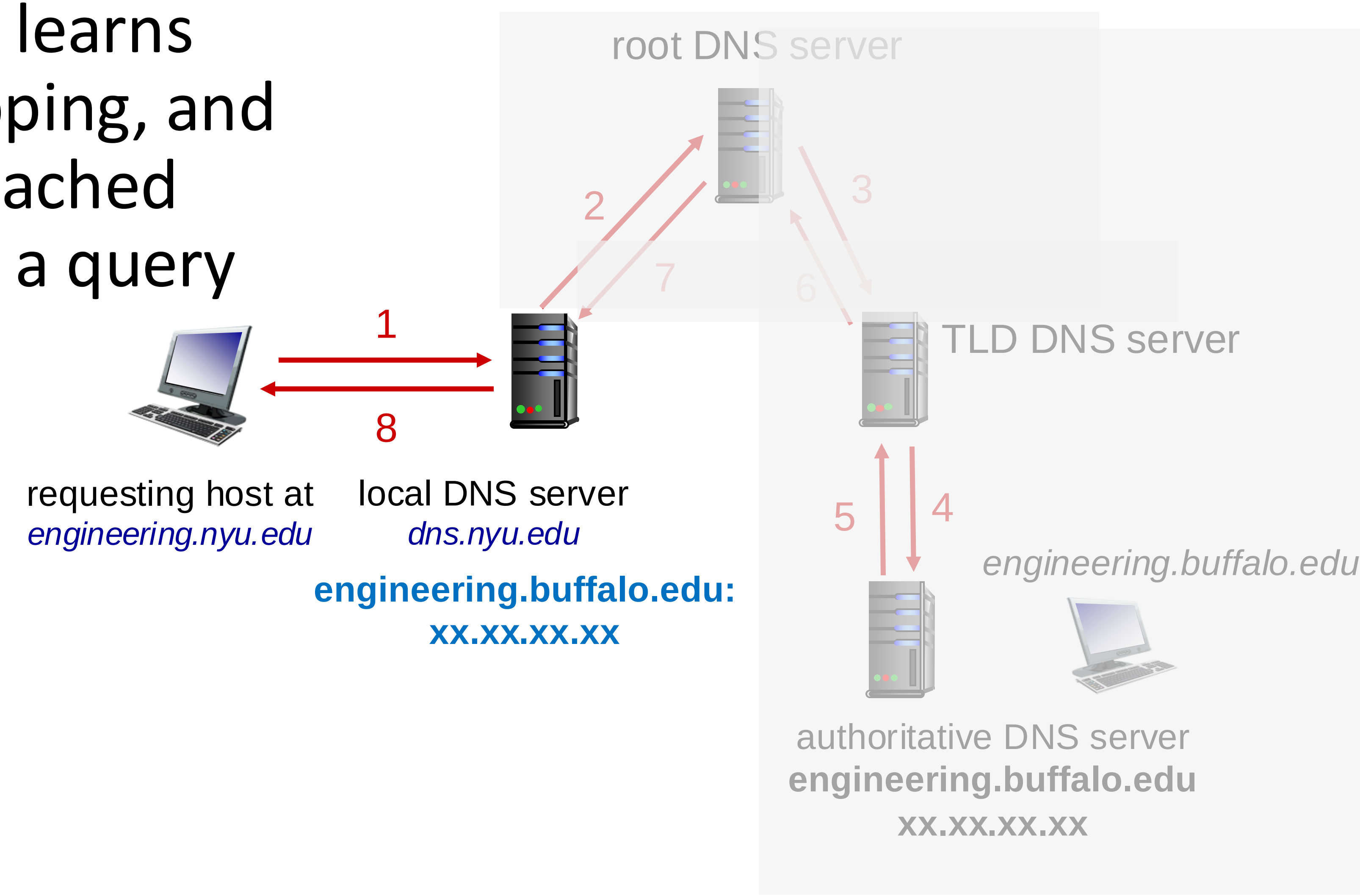
Recursive query:

- puts burden of name resolution on contacted name server
- heavy load at upper levels of hierarchy?



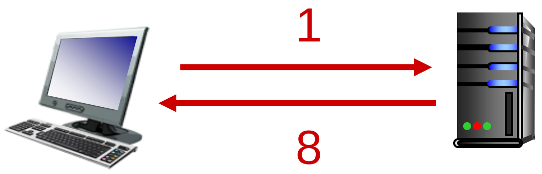
DNS Caching

- once (any) name server learns mapping, it *caches* mapping, and *immediately* returns a cached mapping in response to a query
- Caching reduces delay and overhead
- Where to cache?
  - Local DNS server
  - Web browser
  - All other DNS servers



DNS Cache Consistency

- Goal: Ensuring cached data is up to date
- Avoiding stale information
  - Responses include a “time to live” (TTL) field
  - Delete the cached entry after TTL expires



- Setting the TTL is hard
- TTL trade-offs
  - Small TTL: fast response to change
  - Large TTL: higher cache hit rate

- Follow the hierarchy
  - Top of the hierarchy: days or weeks
  - – Bottom of the hierarchy: seconds to hours
- Tension in practice
  - CDNs set low TTLs for load balancing
  - Browsers cache for 15-60 seconds

DNS records

DNS: distributed database storing resource records **(RR)**

RR format: (`name, TTL, type, Data`)

type=A
- `name` is hostname
- `value` is IP 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
- www.ibm.com is really servereast.backup2.ibm.com
- `value` is canonical name

type=MX
- `value` is name of SMTP mail server associated with `name`

DNS records

DNS: distributed database storing resource records (RR)

RR format: (`name, type, ttl, data`)

type=A

- `name` is hostname
- `data` is IP address

name	value	type	data
example.com.	3600	A	192.168.1.1

DNS records

DNS: distributed database storing resource records (RR)

RR format: (`name, type, ttl, data`)

type=NS

- `name` is domain (e.g., foo.com)
- `data` is hostname of authoritative name server for this domain

name	ttl	type	data
example.com.	**86400**	**NS**	**ns1.example.com.**

ns1.example.com are responsible for handling DNS queries for **example.com**

DNS records

DNS: distributed database storing resource records (RR)

RR format: (`name, type, ttl, data`)

type=CNAME

- `name` is alias name for some "canonical" (the real) name
- `data` is canonical name

name	ttl	type	data
www.example.com.	3600	CNAME	example.com

www.example.com
blog.example.com
shop.example.com
} **example.com**

www.example.com → example.com → 192.168.1.1

DNS records

DNS: distributed database storing resource records (RR)

RR format: (`name, type, ttl, data`)

type=MX

- `data` is name of SMTP mail server associated with `name`

`name`	`ttl`	`type`	`data`
example.com.	**3600**	**MX**	**10 mail.example.com**

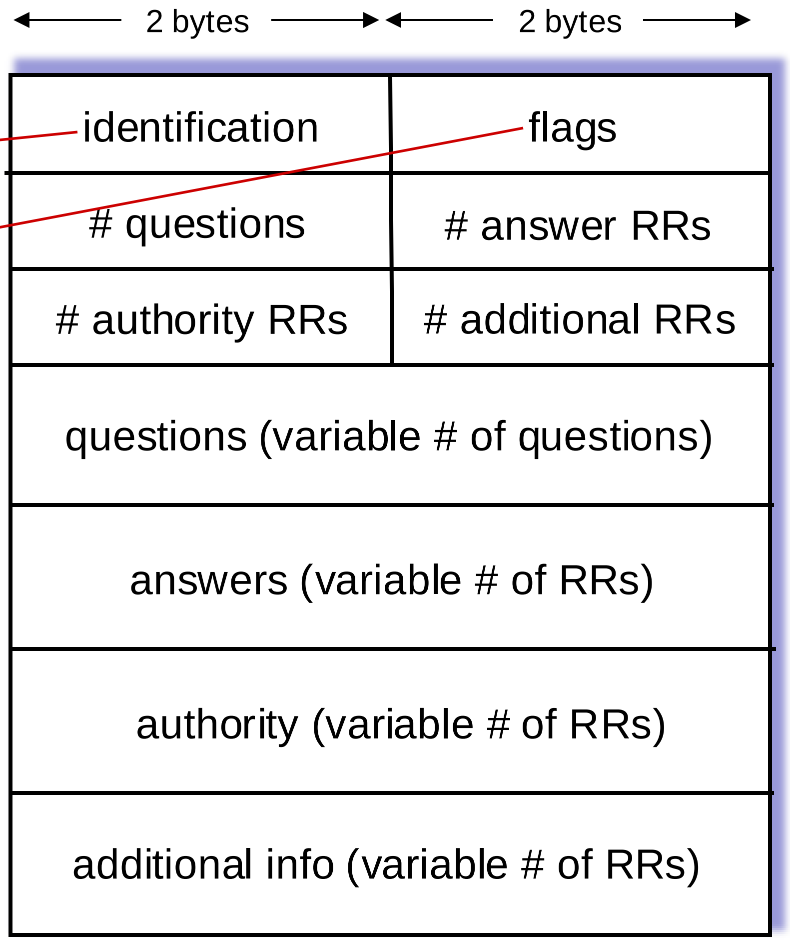
The priority 10 determines which mail server to try first.

DNS protocol messages

DNS *query* and *reply* messages, both have same *format:*

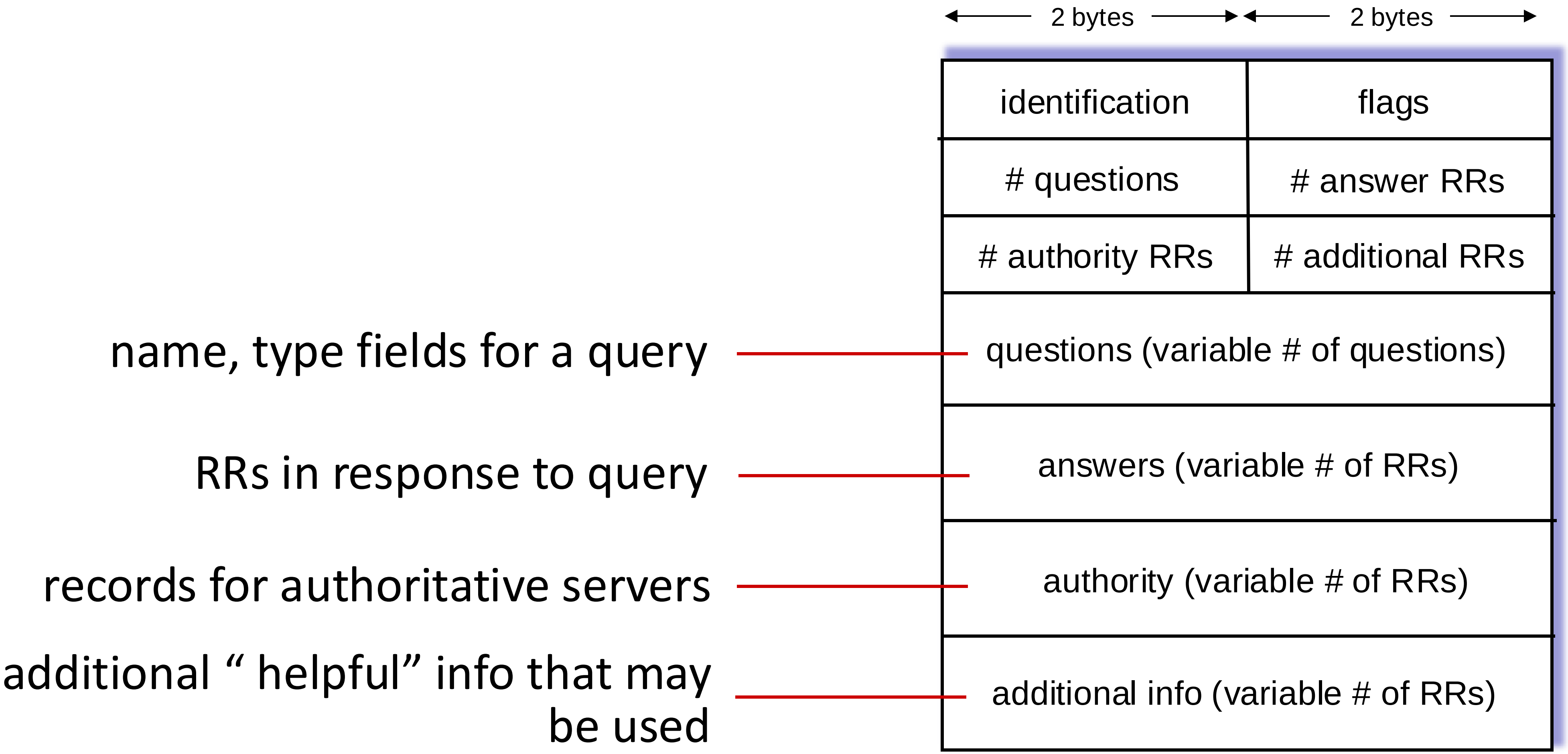
message header:

- identification: 16 bit # for query, reply to query uses same #
- flags:
  - query or reply
  - recursion desired
  - recursion available
  - reply is authoritative



DNS protocol messages

DNS *query* and *reply* messages, both have same *format:*



Getting your info into the DNS

example: new startup "Network Utopia"

- register name networkuptopia.com at *DNS registrar* (e.g., Network Solutions)
  - provide names, IP addresses of authoritative name server (primary and secondary)
  - registrar inserts info into .com TLD server:

    `(networkutopia.com, NS,` **`dns1.networkutopia.com`**`)`

    `(`**`dns1.networkutopia.com`**`, A, 212.212.212.1, A)`
- create authoritative server locally **`dns1.networkutopia.com`** with IP address `212.212.212.1`

DNS security

DDoS attacks

- bombard root servers with traffic
  - not successful to date
  - traffic filtering
  - local DNS servers cache IPs of TLD servers, allowing root server bypass
- bombard TLD servers
  - potentially more dangerous

Spoofing attacks

- intercept DNS queries, returning bogus replies
  - DNS cache poisoning
  - RFC 4033: DNSSEC authentication services

DNS hijacking

- Attacker sends forged DNS reply to client