Internet Architecture

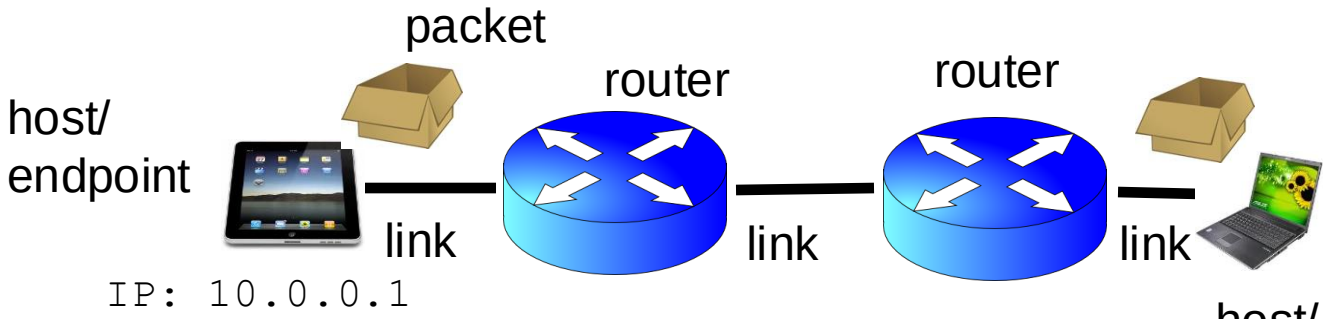
Lecture 2

Srinivas Narayana

http://www.cs.rutgers.edu/~sn624/553-S25

RUTGERS
UNIVERSITY | NEW BRUNSWICK

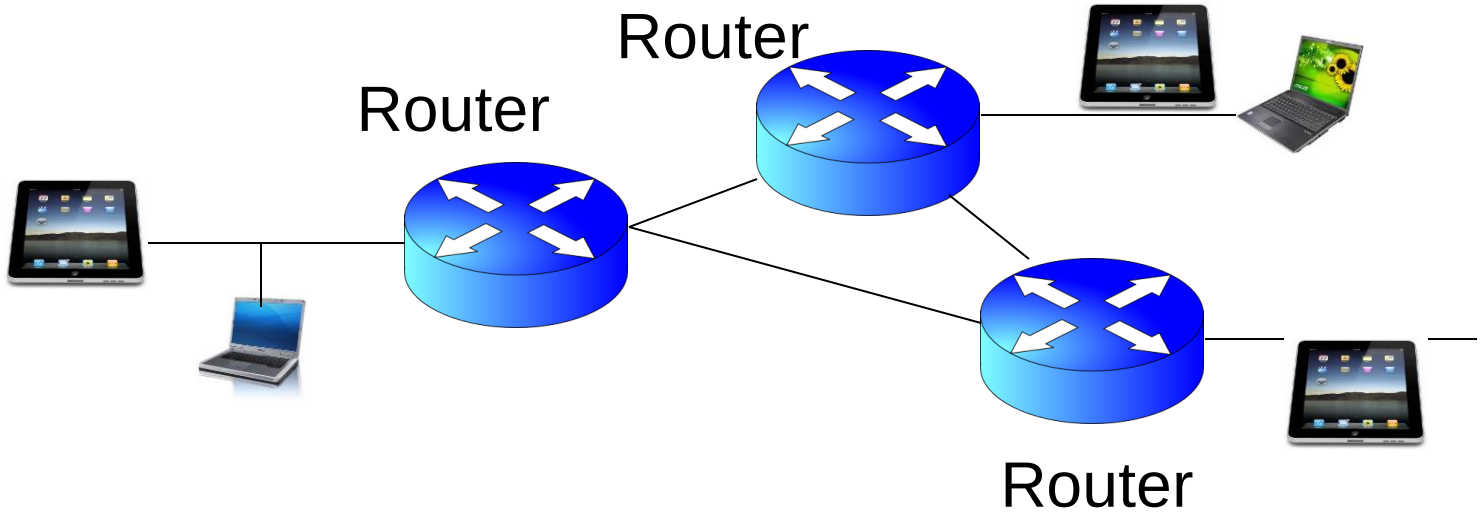
Some definitions



- The Internet is an example of a computer network
- Endpoint or Host: Machine running user application
- Packet: a unit of data transmission (ex: 1500 bytes)
- Link: physical communication channel between two or more machines
- Router: A machine that processes packets moving them from one link to another towards a destination
- Network: Collection of interconnected machines
- Address: a unique name given to a machine

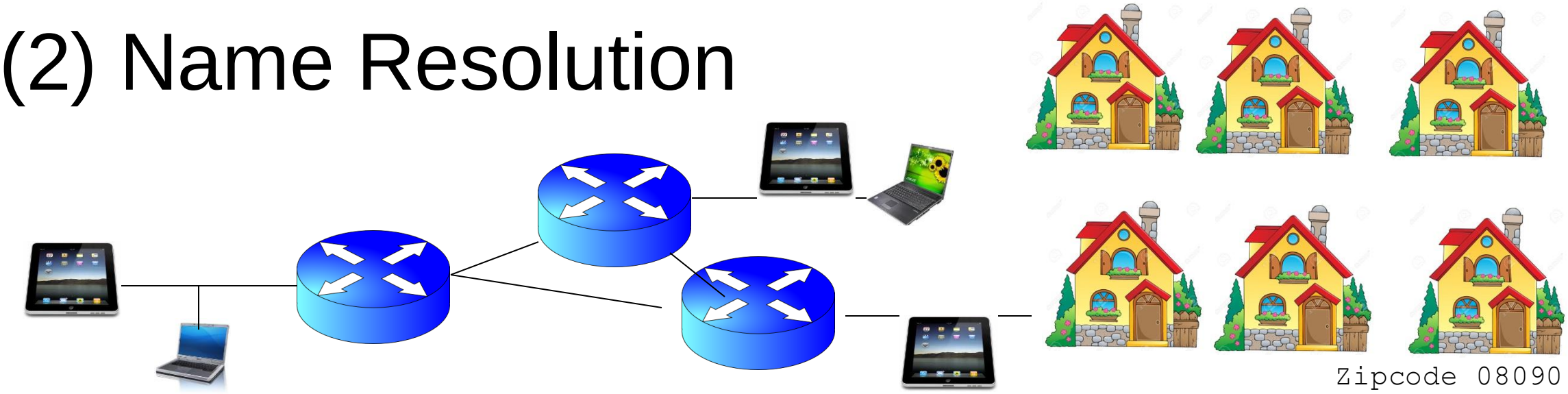
Some fundamental problems

(1) Routing



- Networks must move data between different hosts
- Need to figure out how to move packets from one host to another host, e.g., how to reach google.com from your laptop
- Known as the routing problem

(2) Name Resolution



- Routing effectively requires locating the endpoints appropriately
  - Memory, speed, reactivity
- Internet addresses allocated hierarchically
  - Machine readable, not easy for humans to remember
- Link addresses are tied to the hardware on the endpoint
- Name resolution: how to turn human-readable names (google.com) into routable addresses?

In general, networks give no guarantees

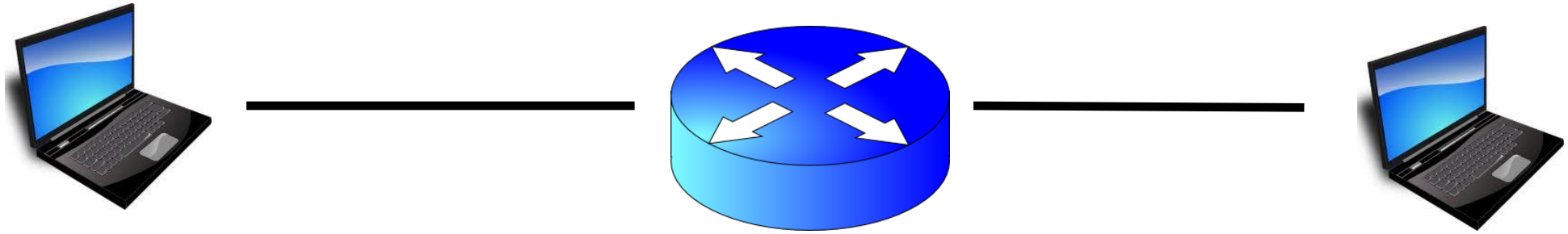
- Packets may be lost, corrupted, reordered, on the way to the destination
  - Best effort delivery

- Advantage: The network becomes very simple to build
  - Don’t have to make it reliable
  - Don’t need to implement any performance guarantees
  - Don’t need to maintain packet ordering
  - Almost any medium can deliver individual packets
    - Example: RFC 1149: “IP Datagrams over Avian Carriers”

- Early Internet thrived: easy to engineer, no guarantees to worry about

Providing guarantees for applications

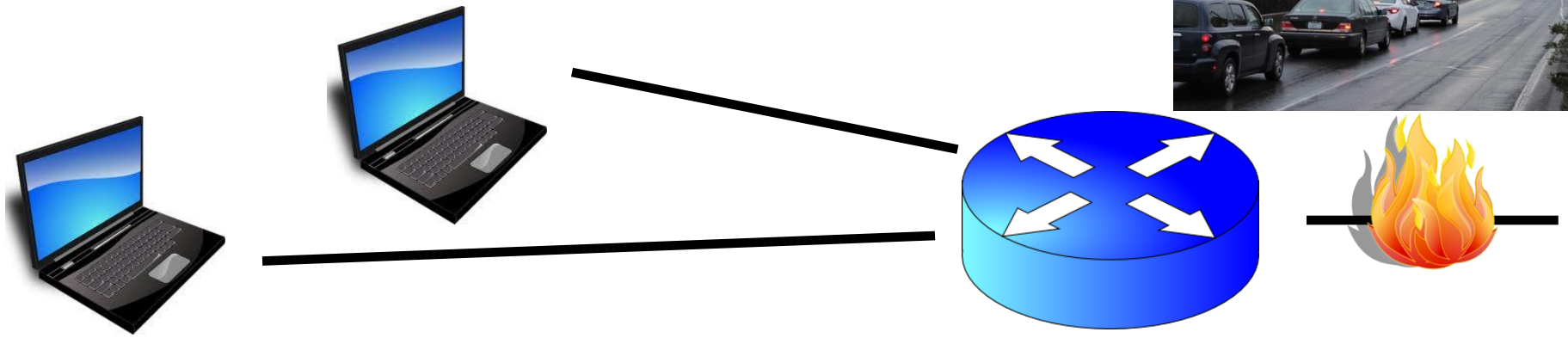
- How should endpoints provide guarantees to applications?



- Transport software on the endpoint oversees implementing guarantees on top of an unreliable network
- Reliable delivery, ordered delivery, fair sharing of resources

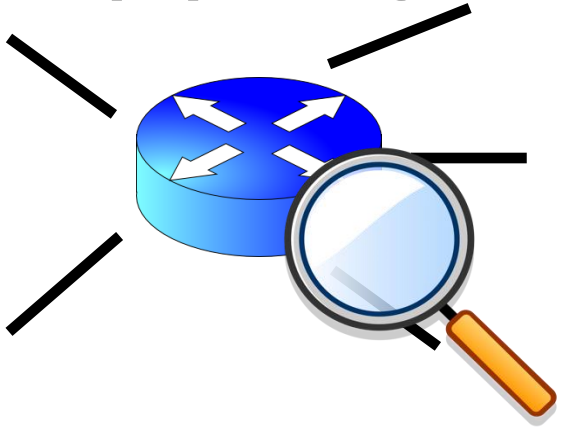
(3) Congestion control

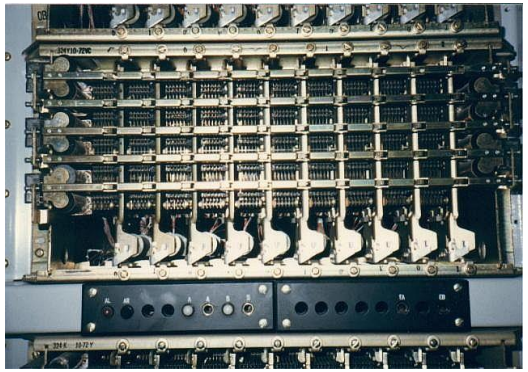
- How quickly should endpoints send data?

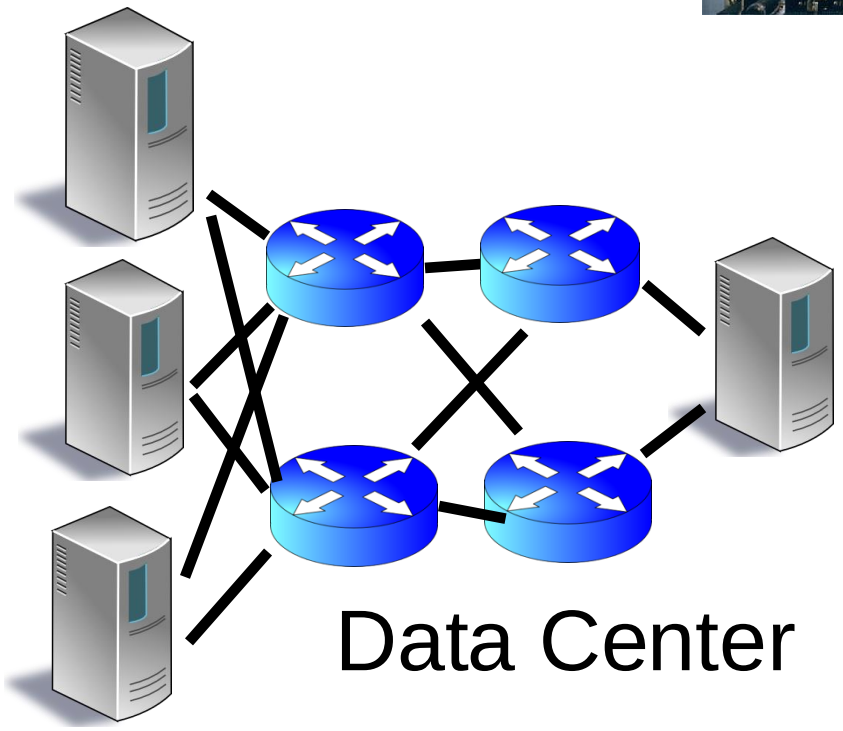


- Known as the congestion control problem
- Congestion control algorithms at source endpoints react to remote network congestion. Part of the transport sw/hw stack.
- Key question: How to vary the sending rate based on network signals?

(4) High-Speed Interconnect





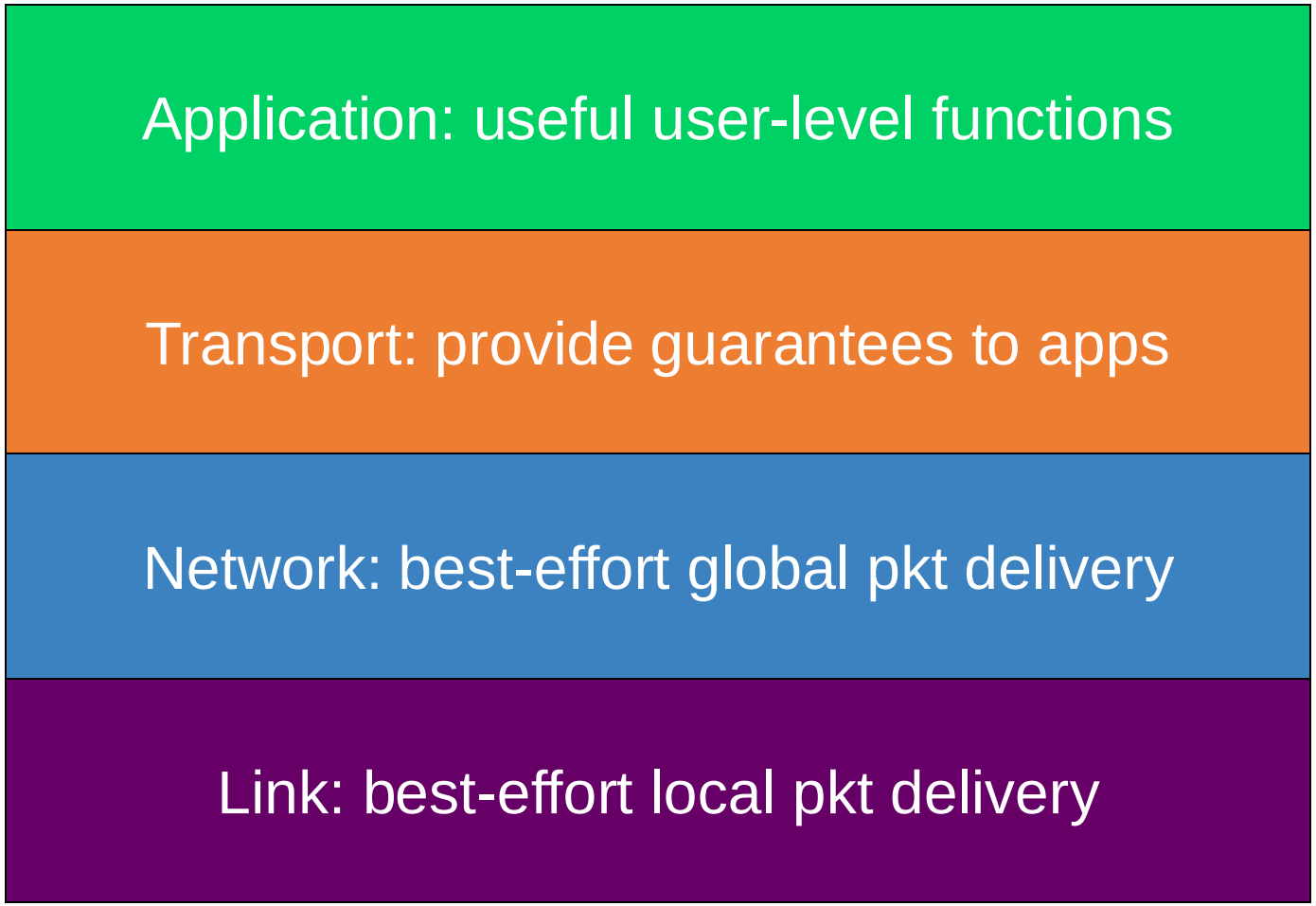


Data Center

- Transport won't help if the network has choke points: e.g., routers
- The interconnection problem: how do you design routers to achieve high end-to-end performance between endpoints?
  - Also designing large data center networks

Layering and Protocols

Software/hardware organization at hosts



Communication functions broken up and “stacked”

Each layer depends on the one below it.

Each layer supports the one above it.

The interfaces between layers are well-defined and standardized.

Internet software and hardware are arranged in layers.

Layering provides modularity

Each layer: well-defined function & interfaces to layers above & below it.

Functionality is implemented in protocols.

Protocols: The “rules” of networking

- Protocols consist of two things

- Message format
  - structure of messages exchanged with an endpoint

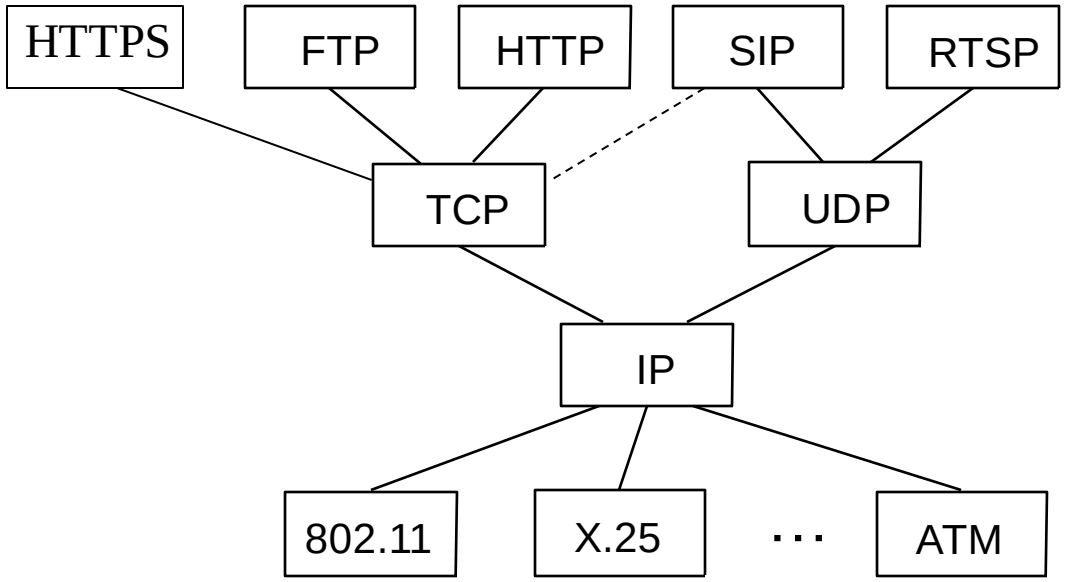
- Actions
  - operations upon receiving, or not receiving, messages

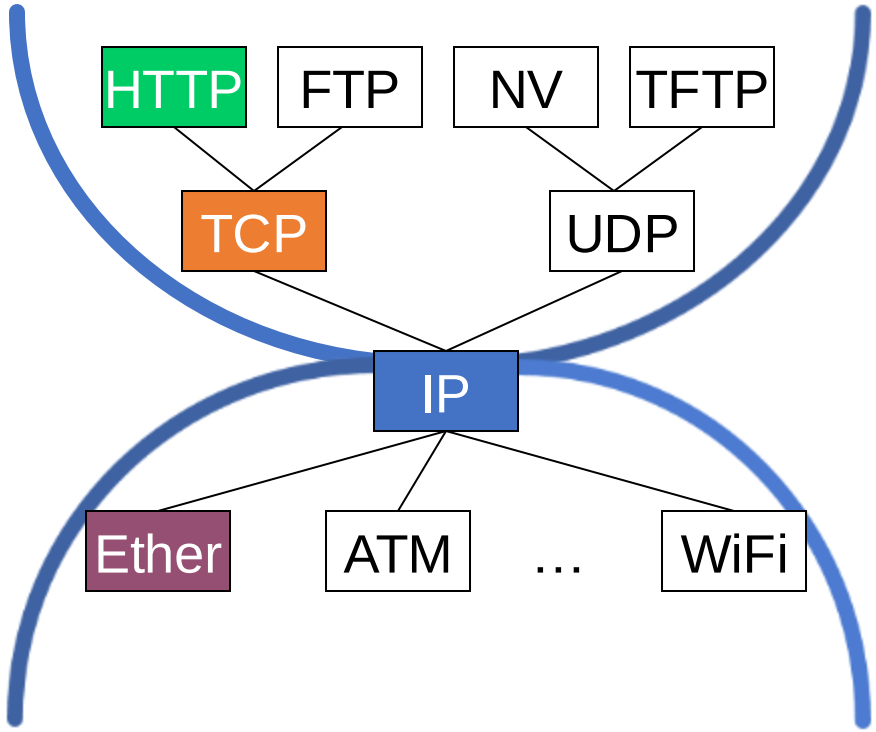
- Example of a Zoom conversation:
  - Message format: English words and sentences
  - Actions: when a word is heard, say “yes”; when nothing is heard for more than 3 seconds, say “can you hear me?”

The protocols of the Internet

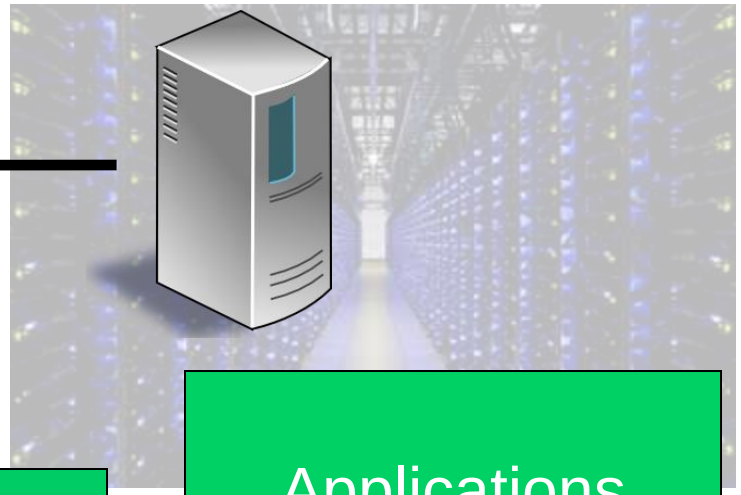
- Standardized by the Internet Engineering Task Force (IETF)
  - through documents called RFCs (“Request For Comments”)

- Layering of protocols





The Internet



Applications

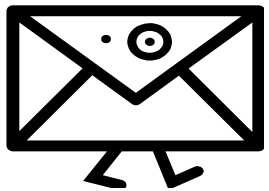
Transport

Network

Link layer

Packet starts as an app “payload”

Packet takes on headers (metadata) at each layer



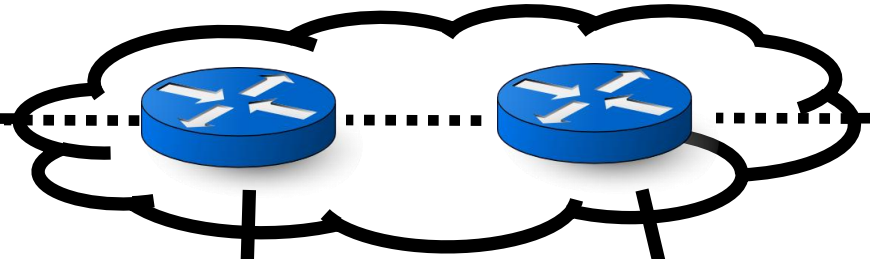
The Internet

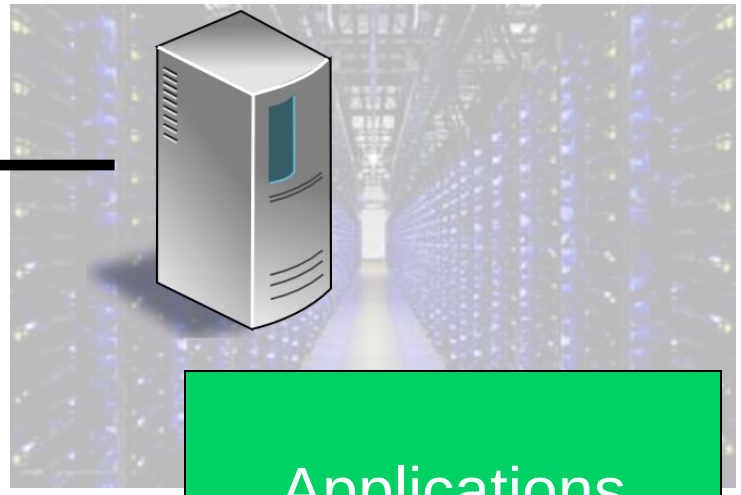
Applications

Transport

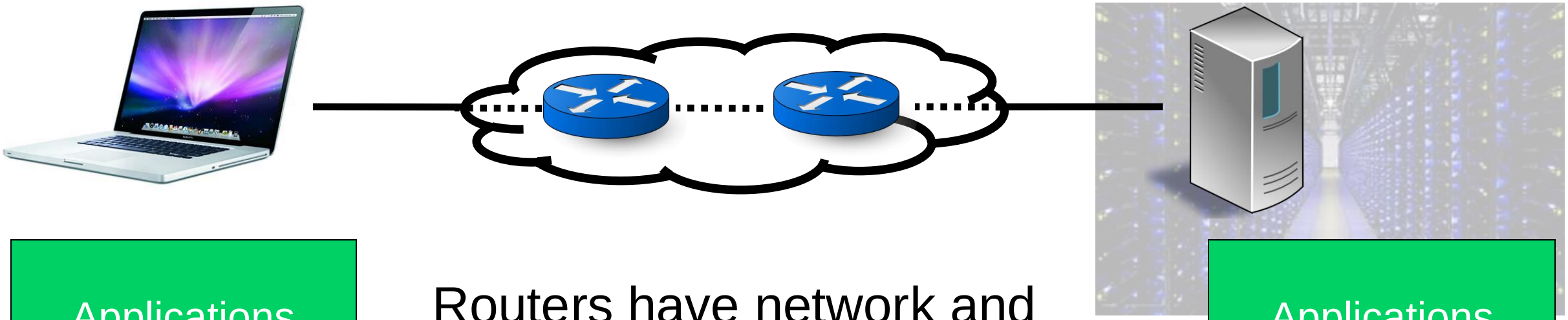
Network

Link layer

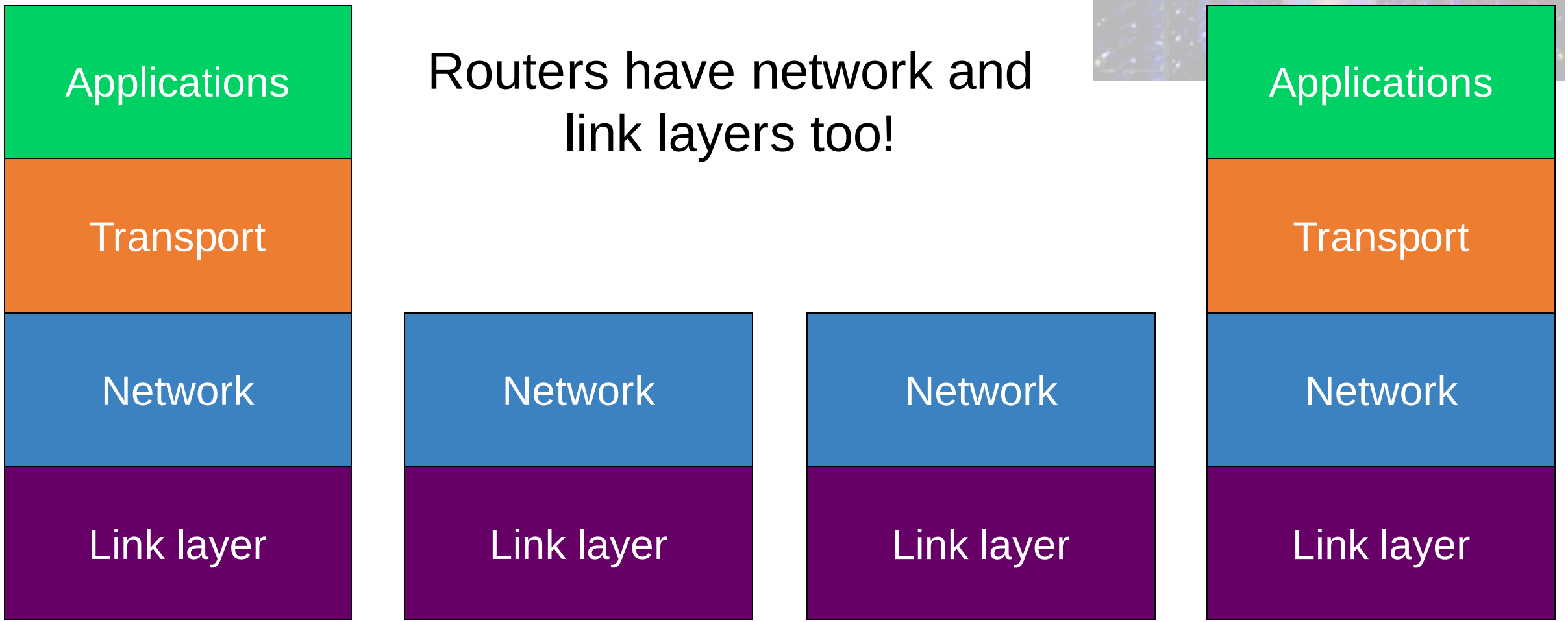




Applications			Applications
Transport			Transport
Network	Network	Network	Network
Link layer	Link layer	Link layer	Link layer



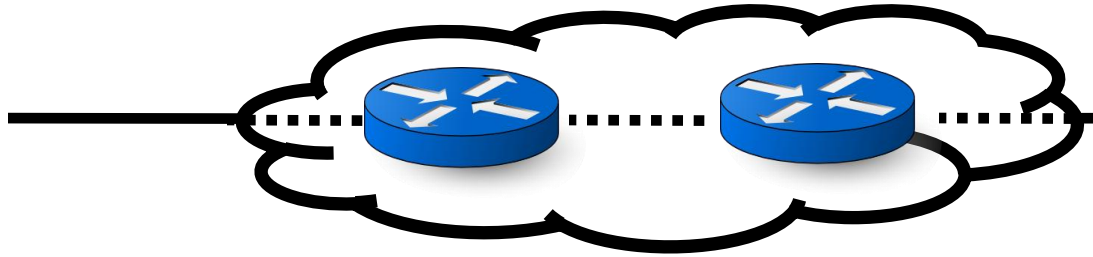
Routers have network and link layers too!

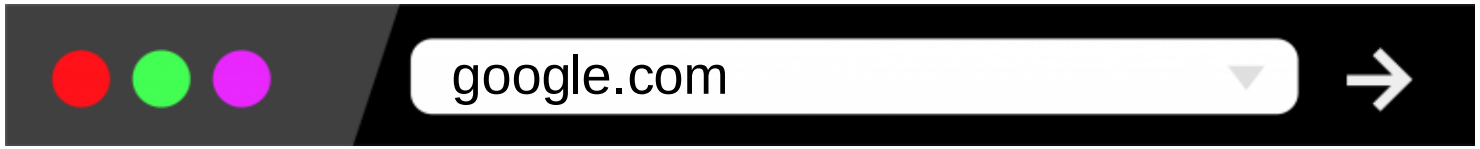


Layering

- Communication over the Internet is a complex problem.
- Layering simplifies understanding, testing, maintaining
- Easy to improve or replace protocol at one layer without affecting others

Name Resolution





Machines communicate using IP addresses and ports
IP addresses: ~12 digits (IPv4) or more
Ports: fixed based on application (e.g., 80: web)

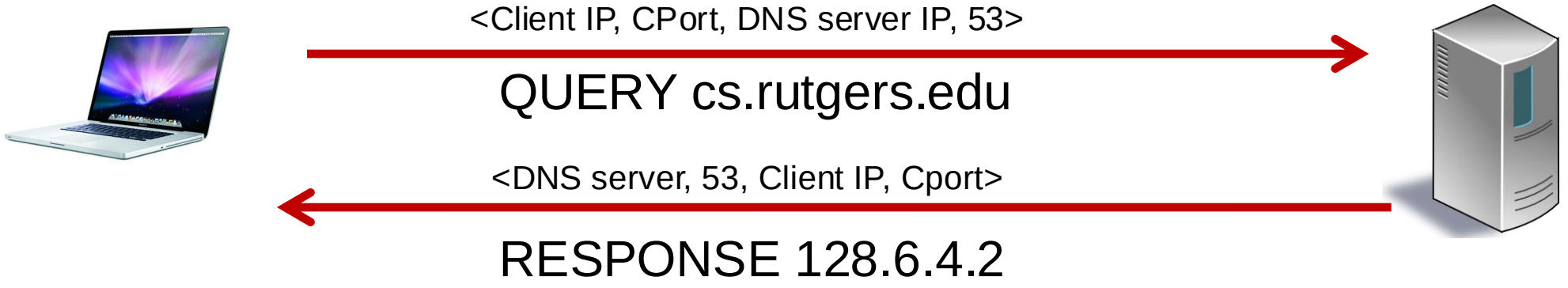
Need a way to turn human-readable addresses into Internet addresses.

Ask someone	Ask everyone	Tell everyone
Directory service	Query broadcast	Information flooding

Asking “someone” could involve asking many machines…

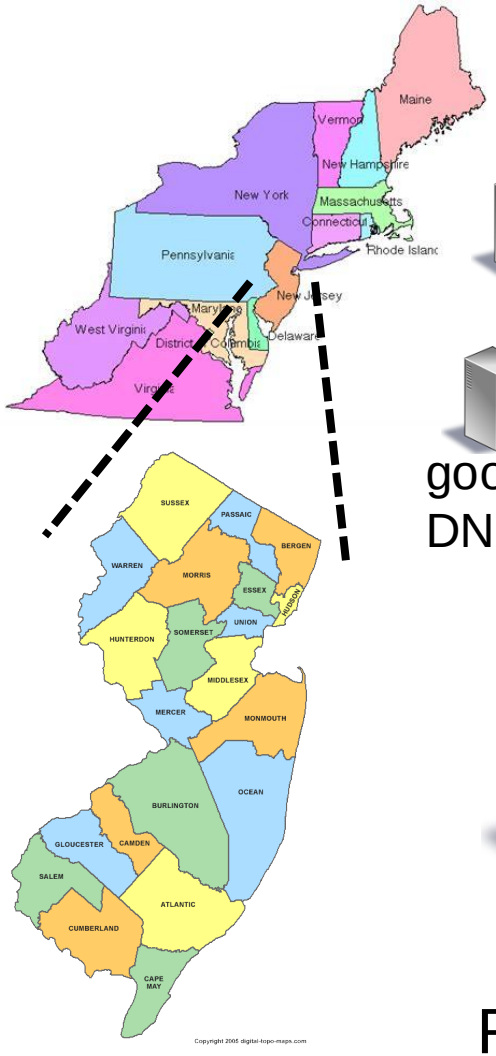
Domain Name Service

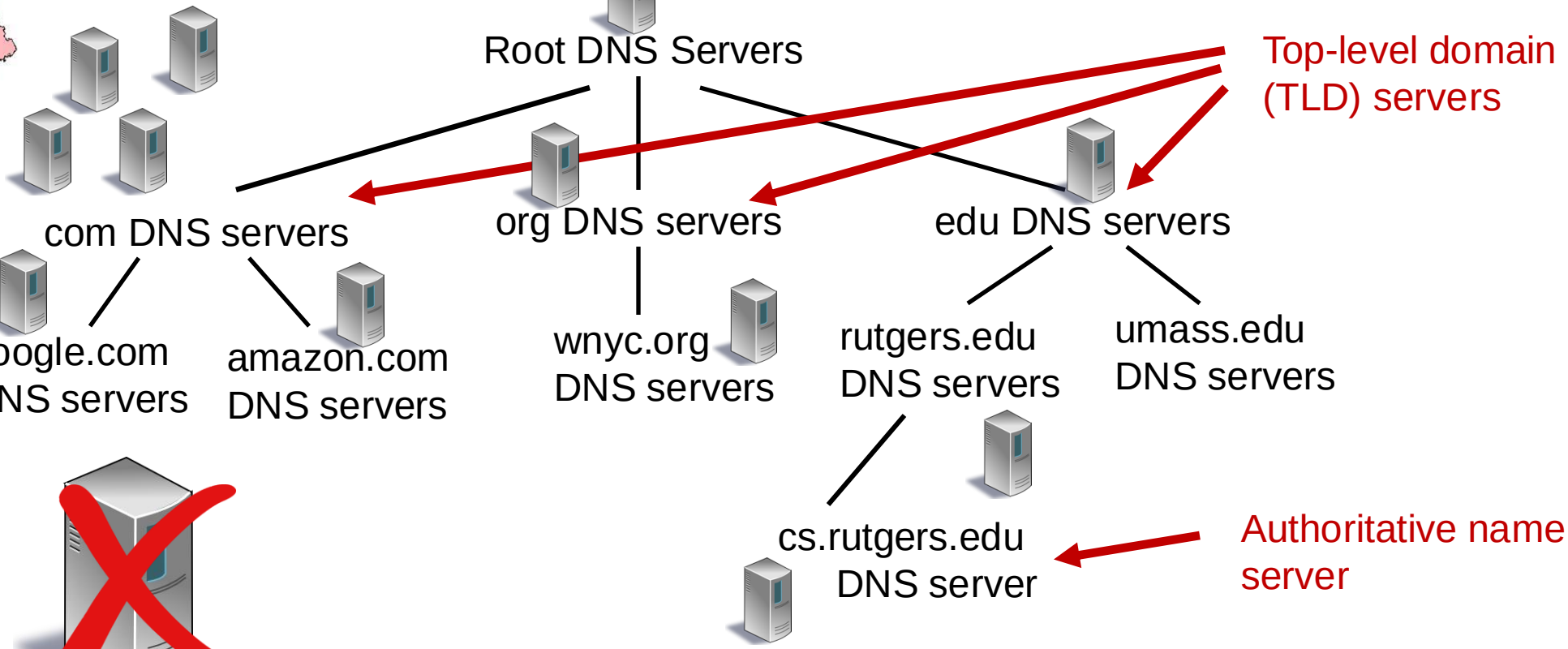
DOMAIN NAME	IP ADDRESS
spotify.com	98.138.253.109
cs.rutgers.edu	128.6.4.2
www.google.com	74.125.225.243
www.princeton.edu	128.112.132.86



- Key idea: Implement a server that looks up a table.
- Will this scale?
  - Every new (changed) host needs to be (re)entered in this table
  - Performance: can the server serve billions of Internet users?
  - Failure: what if the server or the database crashes?
  - Security: What if someone “takes over” this server?

Distributed and hierarchical database





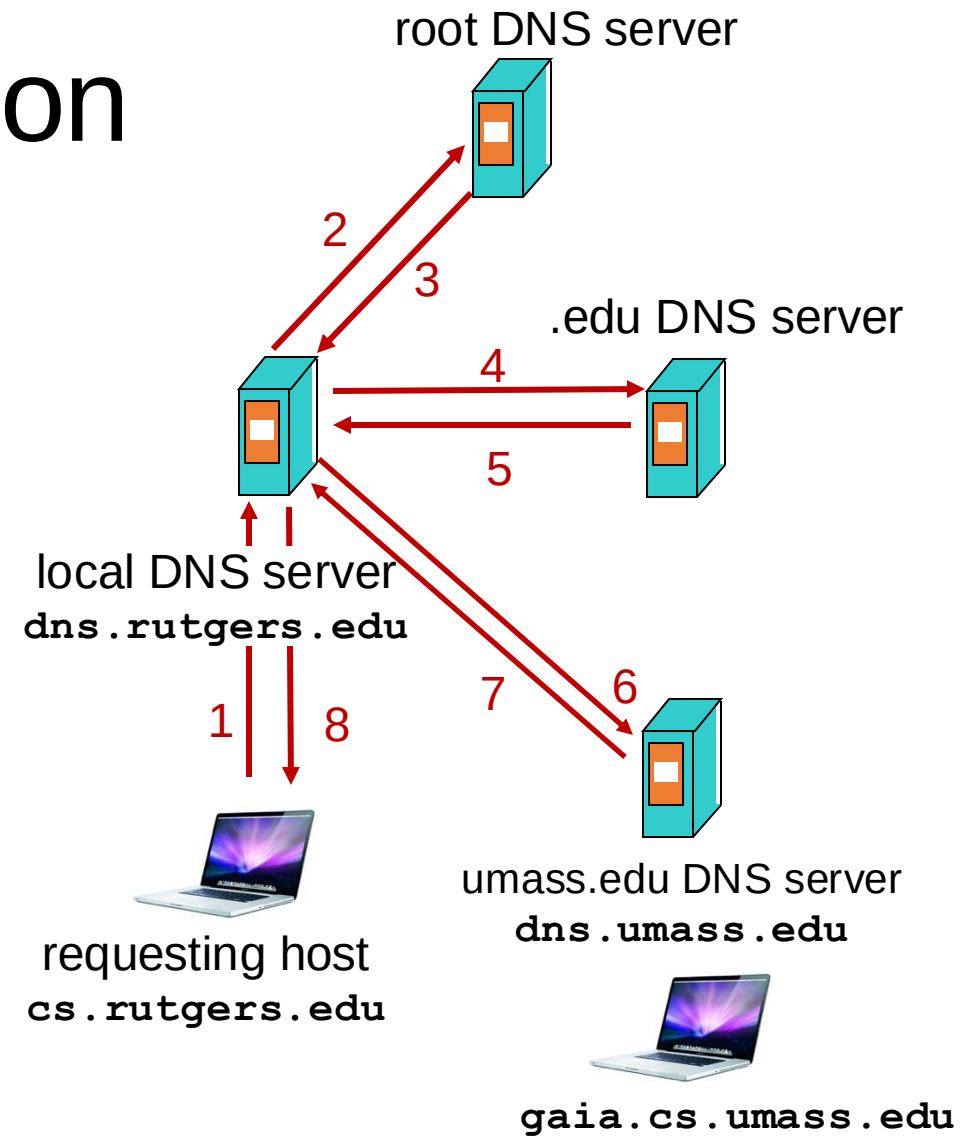
RFC 1034

Hierarchy Replication Indirection

DNS name resolution

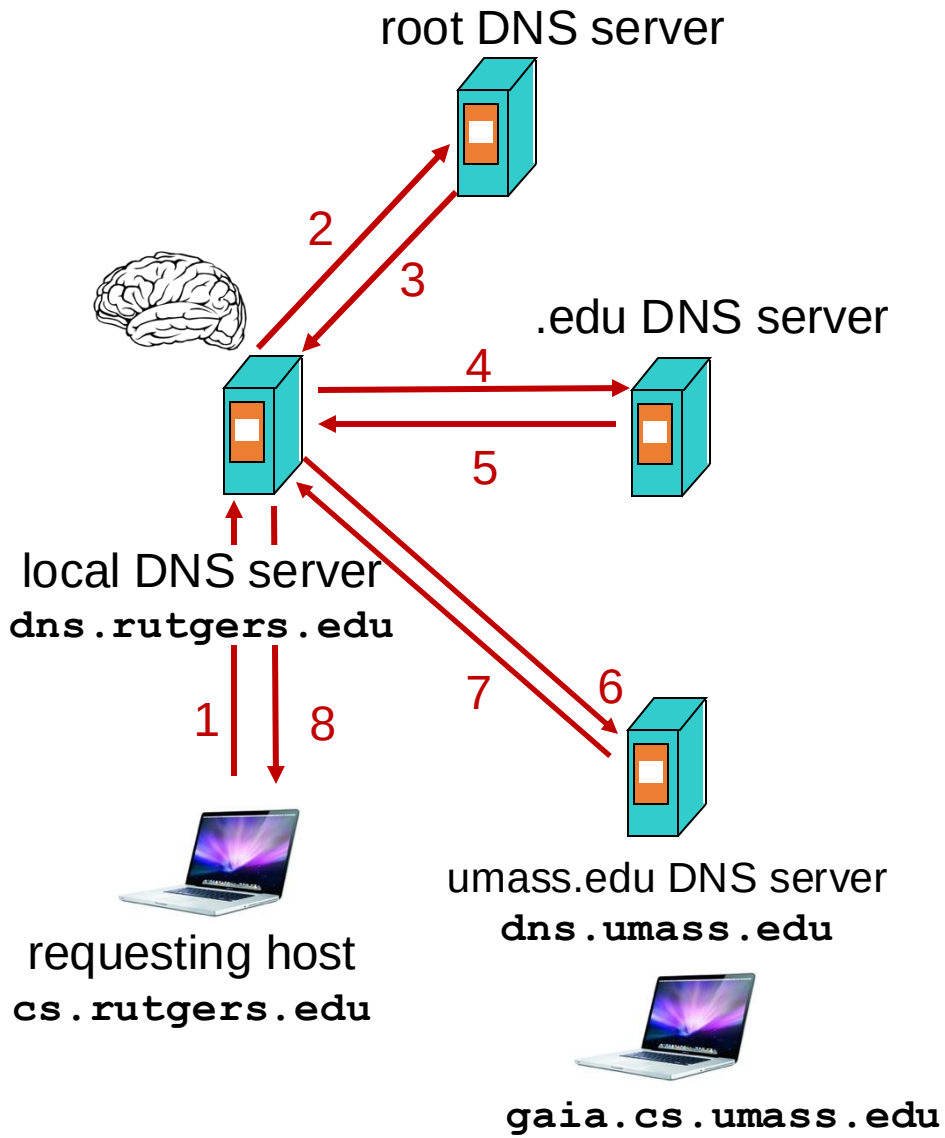
- Host at cs.rutgers.edu wants IP address for gaia.cs.umass.edu

- Local DNS server
- Root DNS server
- TLD DNS server
- Authoritative DNS server



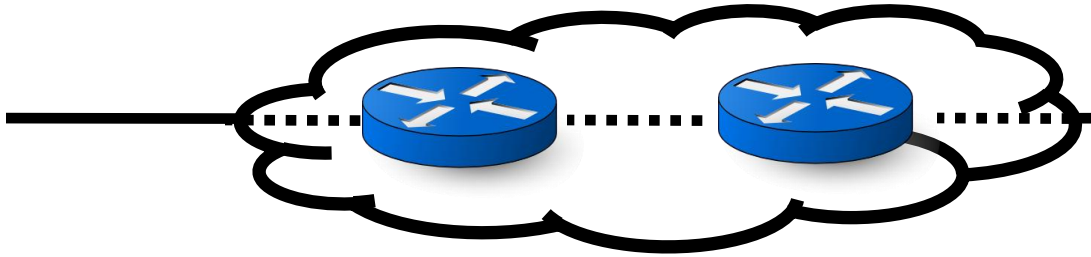
DNS caching

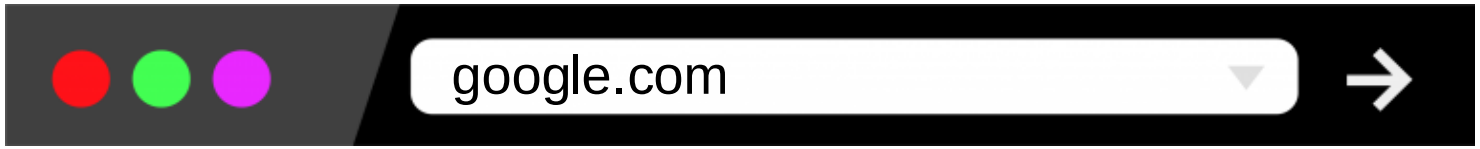
- Once (any) name server learns a name to IP address mapping, it *caches* the mapping
- Cache entries timeout (disappear) after some time
- TLD servers typically cached in local name servers
- In practice, root name servers aren't visited often!
- Caching is pervasive in DNS



Example DNS interactions

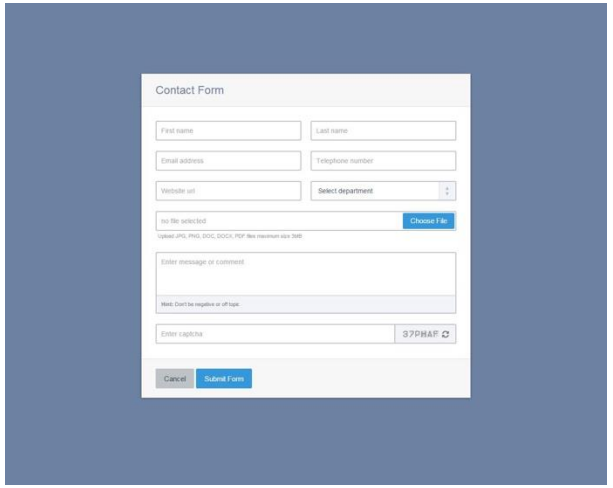
- `dig <domain-name>`
- `dig +trace <domain-name>`
- `dig @<dns-server> <domain-name>`





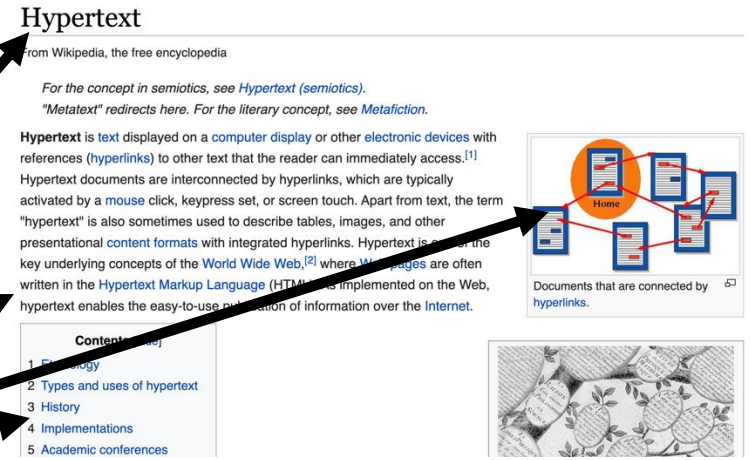
The web is a *specific* application protocol running over a network: HyperText Transfer Protocol (HTTP)

Each object addressable by a name (URL)



Named objects can be static (image, video)

… or the result of a dynamic app process


Hypertext

From Wikipedia, the free encyclopedia

For the concept in semiotics, see Hypertext (semiotics).
"Metatext" redirects here. For the literary concept, see Metafiction.

Hypertext is text displayed on a computer display or other electronic devices with references (hyperlinks) to other text that the reader can immediately access.[1] Hypertext documents are interconnected by hyperlinks, which are typically activated by a mouse click, keypress set, or screen touch. Apart from text, the term "hypertext" is also sometimes used to describe tables, images, and other presentational content formats with integrated hyperlinks. Hypertext is one of the key underlying concepts of the World Wide Web,[2] where Web pages are often written in the Hypertext Markup Language (HTML) or implemented on the Web, hypertext enables the easy-to-use publication of information over the Internet.

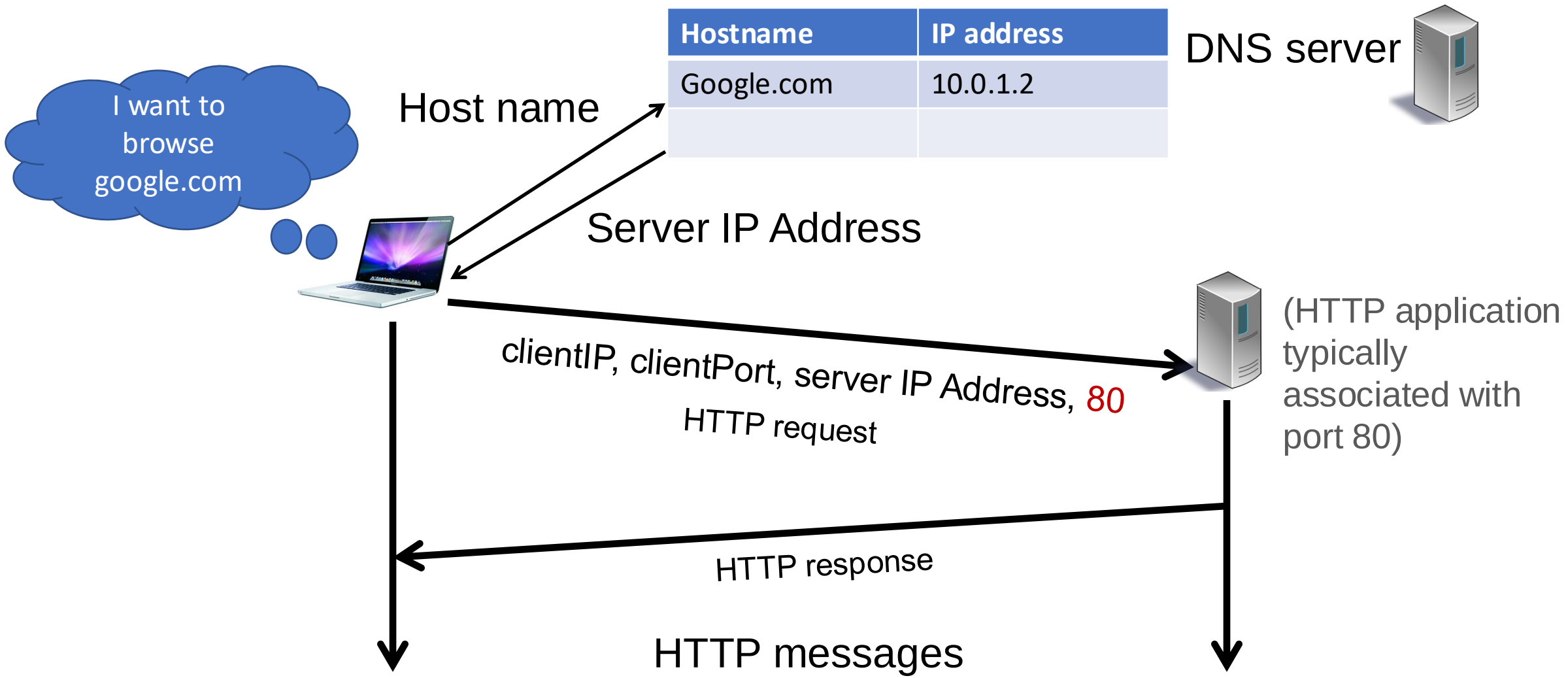
Contents [hide]
1 Etymology
2 Types and uses of hypertext
3 History
4 Implementations
5 Academic conferences

Home

Documents that are connected by hyperlinks.

Objects

Web interactions



Example HTTP interactions

- `wget google.com (or) curl google.com`

- `telnet example.com 80`
  - `GET / HTTP/1.1`
  - `Host: example.com`

(followed by two enter's)

- Exercise: try
  - `telnet google.com 80`
  - `telnet web.mit.edu 80`

Remembering users: cookies

Cookie is typically opaque to client.

client

server

Cookie file
Netflix: 436
Amazon: 1678

http request msg + auth

server creates ID 1678 for user

entry in backend database

http response + **Set-cookie: 1678**

Cookie file
Netflix: 436
Amazon: 1678

http request (no auth) **cookie: 1678**

cookie-specific action

access

Personalized http response

one week later:

Cookie file
Netflix: 436
Amazon: 1678

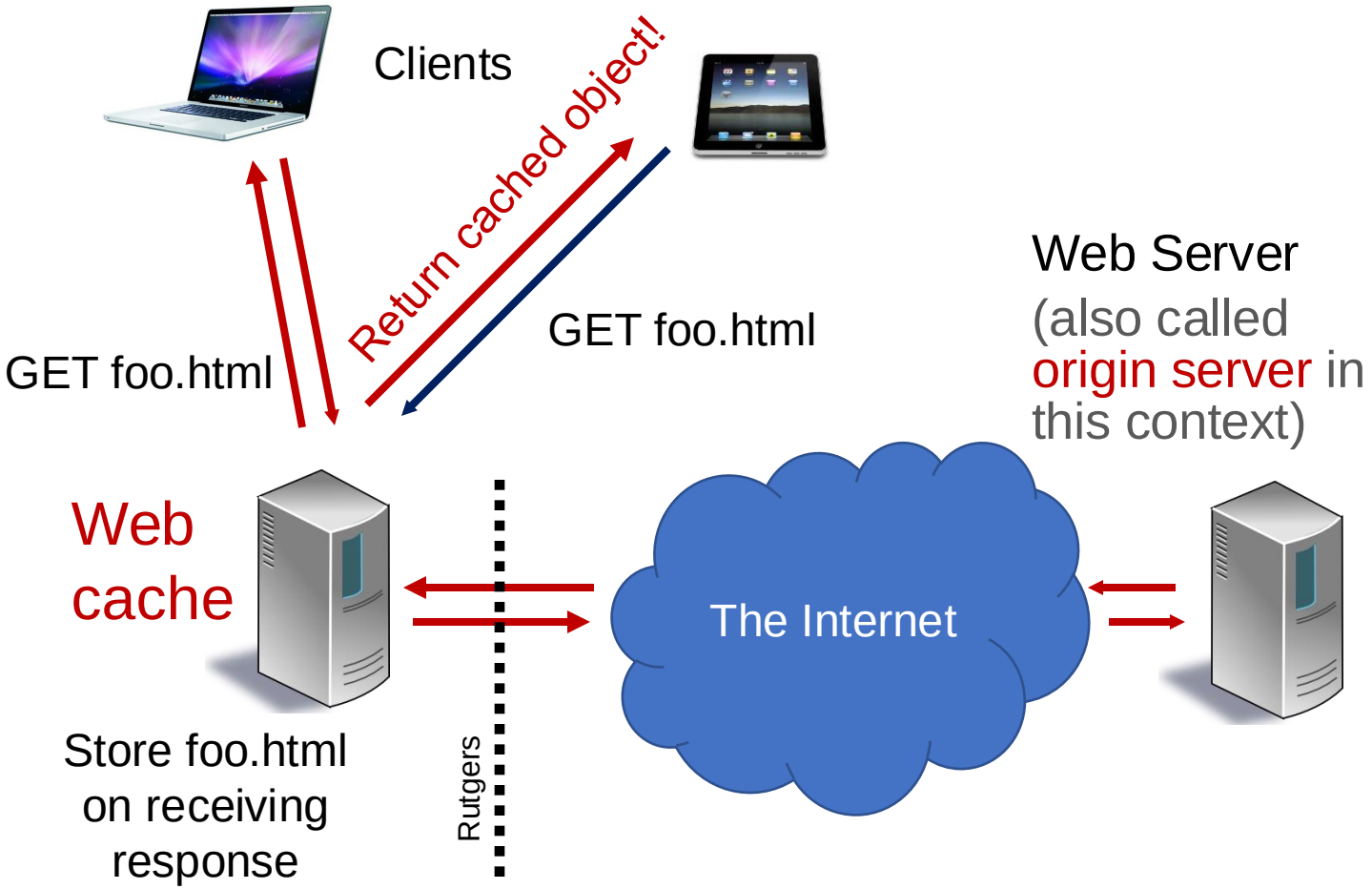
http request (no auth) **cookie: 1678**

cookie-specific action

access

Personalized http response

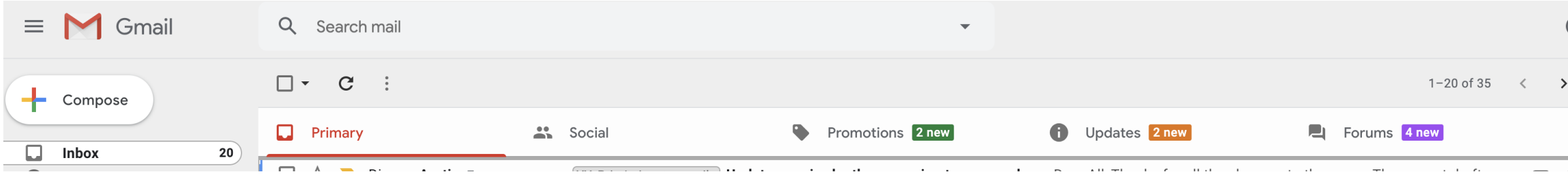
Improving performance: Web caching



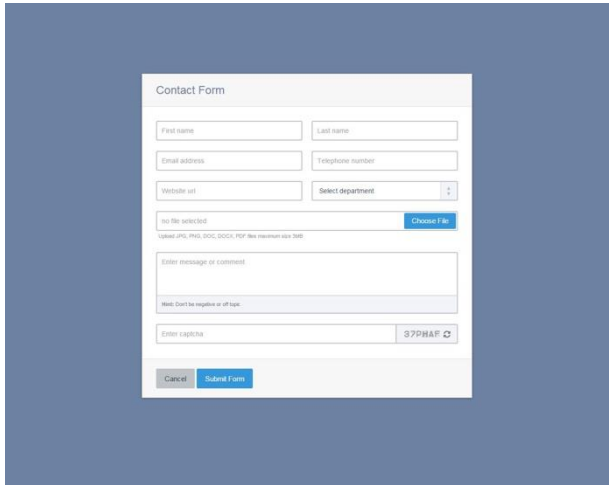
- Network administrators (e.g., Rutgers) may run web caches to remember popular web objects
- Hit: cache returns object
- Miss: obtain object from originating web server (origin server) and return to client
  - Also cache the object locally
- Reduce response time
- Reduce traffic requirements (and $$) on an organization's network connections

Not all content is effectively cacheable

- Personalized content



- Interactive processing
  - e.g., forms, shopping carts, ajax, etc.



- Long tail of (obscure) content