Domain Name System

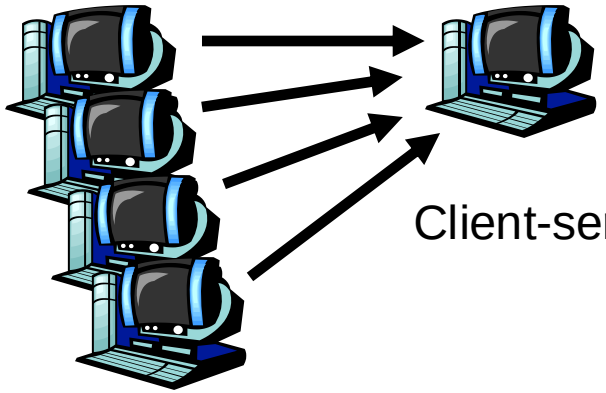
Lecture 4

http://www.cs.rutgers.edu/~sn624/352-F24

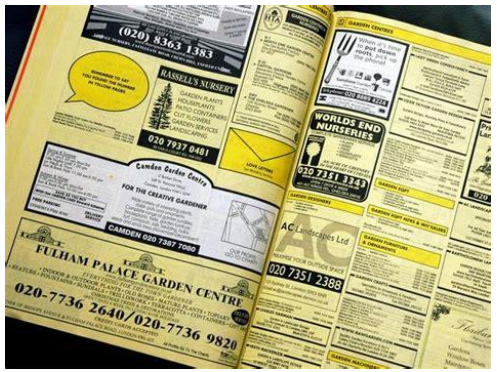
Srinivas Narayana

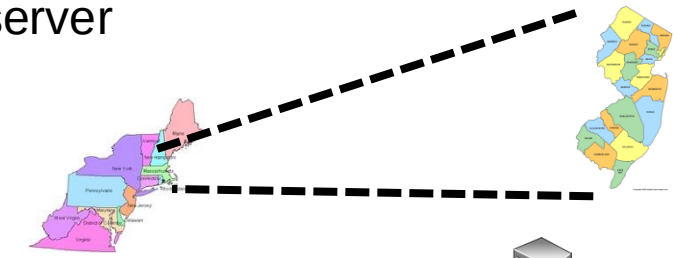
RUTGERS
UNIVERSITY | NEW BRUNSWICK

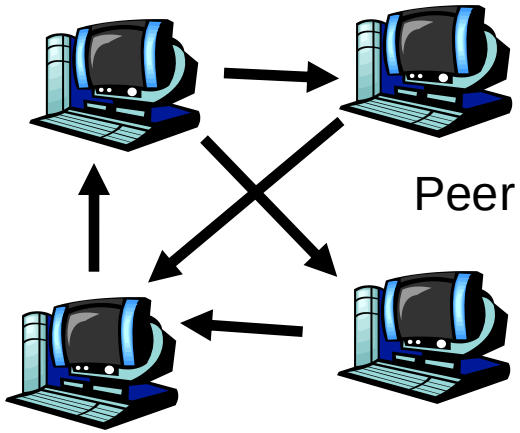
Review



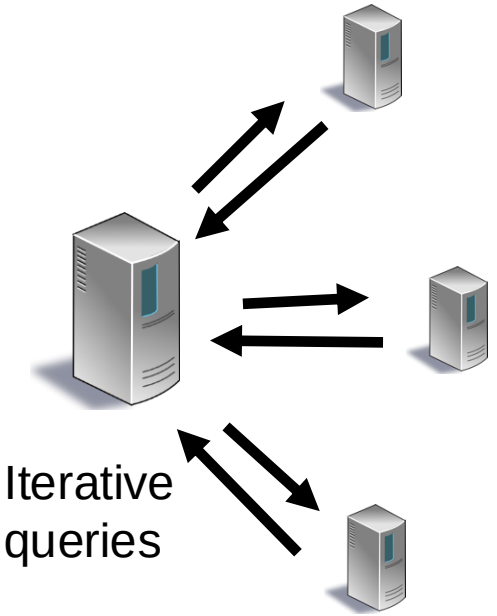
Client-server







Peer to peer



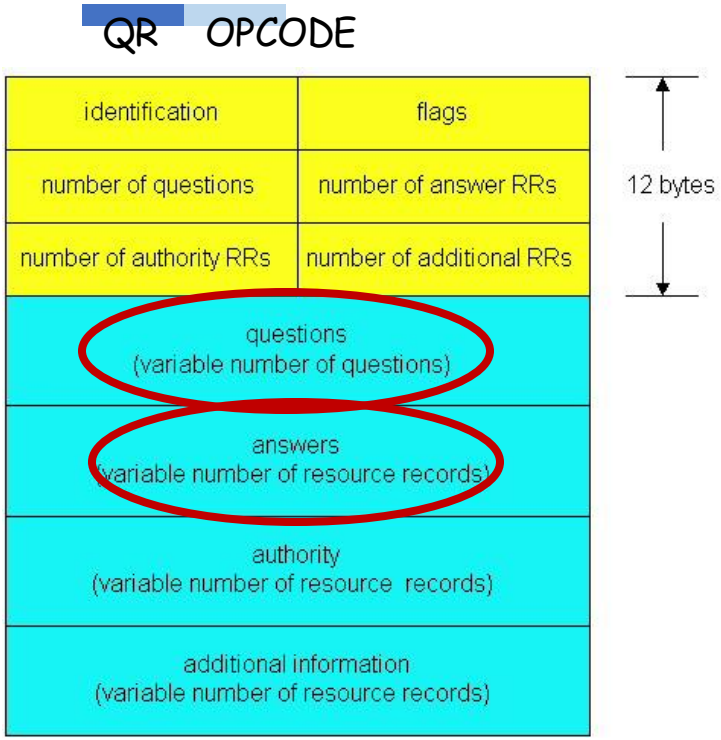
Iterative queries

Domain Name System

Human readable → IP addresses

Hierarchical, distributed database

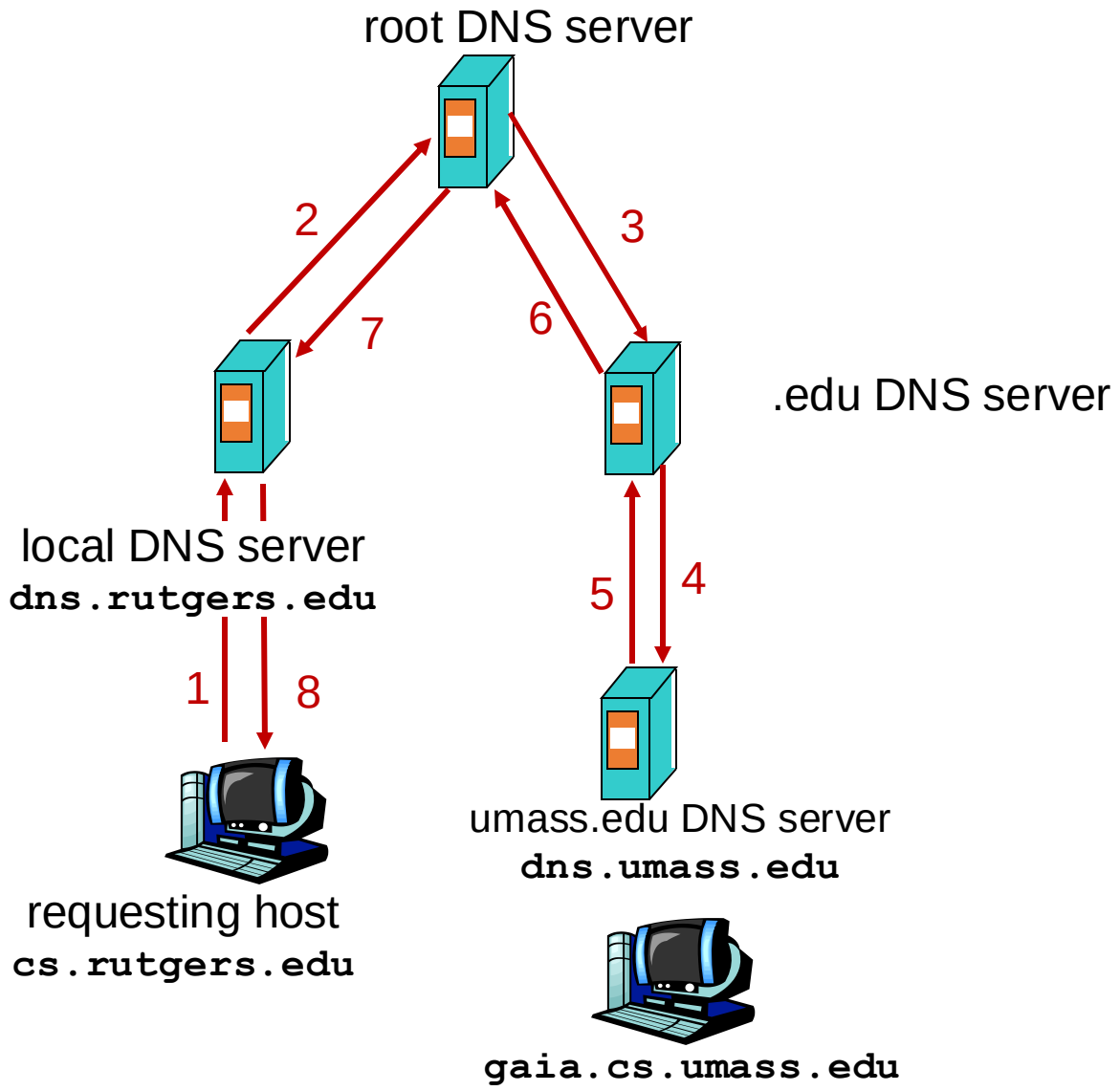
Root server, TLD server, Authoritative name server



Query type

Recursive query:

- Puts burden of name resolution on the contacted (e.g., root) name server
- Query 2 (to root DNS server) is recursive from the local server
- In general, recursive is not preferred for higher levels of the DNS hierarchy



Problem: Load on Higher Levels of DNS

Think about the query load on the root DNS server (regardless of recursive/iterative)

- Must root server answer every DNS query?

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

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 name server) must be known a priori
- The local name server may be bootstrapped “statically”, e.g.,
  - File `/etc/resolv.conf` in unix
  - `Start -> settings-> control panel-> network ->TCP/IP -> properties` in windows
- The local DNS server or with another protocol!
  - DHCP: Dynamic Host Configuration Protocol
- The local DNS server must know the root servers

DNS may seem “basic”, low level, but …

Gone in Minutes, Out for Hours:
Outage Shakes Facebook

Akamai DNS outage knocks many major websites and services offline: PSN, Steam, Fidelity, more [U]

Overloaded Azure DNS Servers to Blame For Microsoft Outage

April 5, 2021

POSTED ON OCTOBER 5, 2021 TO NETWORKING & TRAFFIC

More details about the October 4 outage

DNS Resource Records

DNS is a distributed database

- DNS stores resource records (RRs)
- (Incomplete) message format for each resource record (RR):
  - Class, type, name, value, TTL
- You can read all the gory details of the message format at https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml

DNS records

Type=A

- **name** is hostname
- **value** is IPv4 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**

More complete info at https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml

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

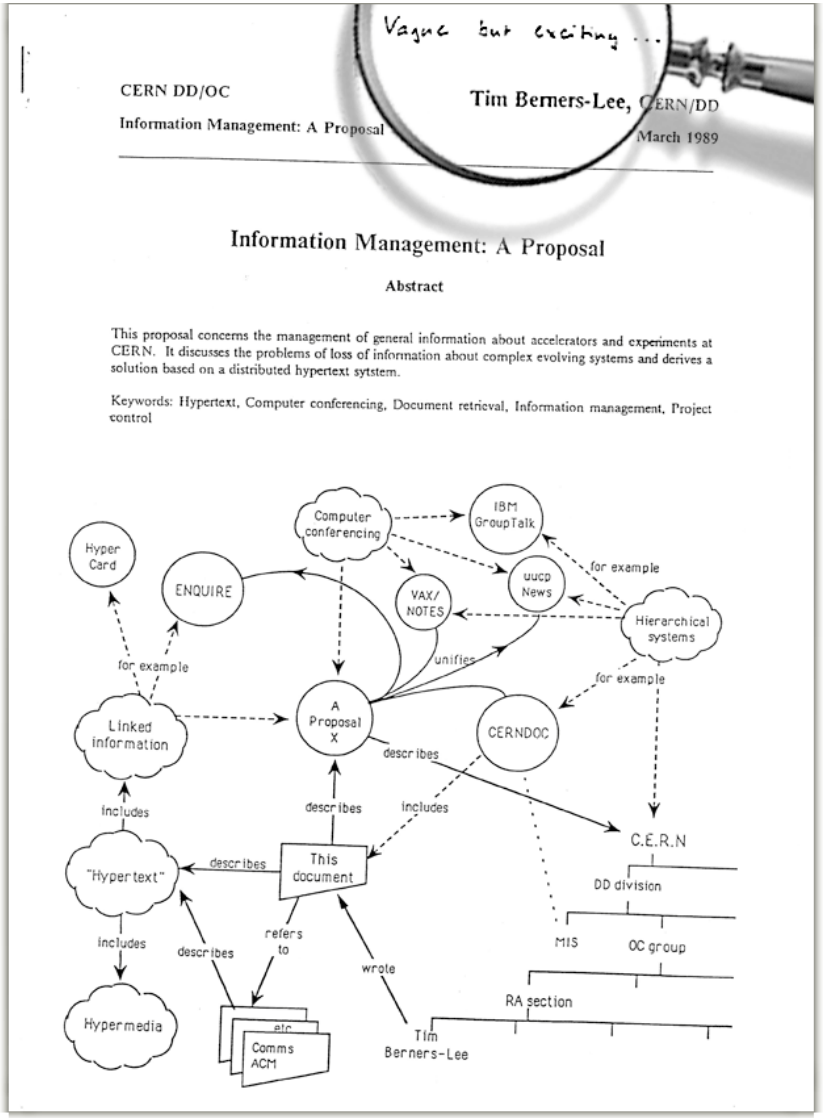
- `dig -t <type> <domain-name>`

Summary of DNS

- Hostname to IP address translation via a global network of servers
- Embodies several scaling principles
  - Partition through a hierarchy to silo query load
  - Replication to scale out at each level of hierarchy
  - Caching to reduce query load
- Once you have a reliable DB, can implement many useful things on top!
- Example 1: Scaling large web services, e.g., google search, by redirecting different clients to different servers (IP addresses)
  - Reliability, load balancing, performance optimization
- Example 2: Associating certificates, keys (security info) with domain names
  - https://www.rfc-editor.org/rfc/rfc8162.html
  - https://datatracker.ietf.org/doc/draft-ietf-dnsop-svcb-https/00/

The Web (HTTP)

The Web: Humble origins

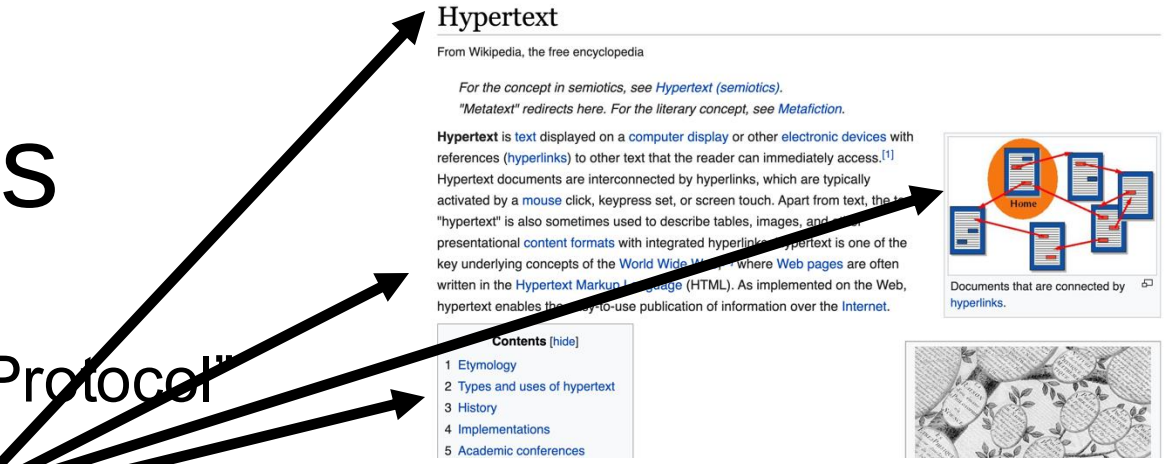


Tim Berners-Lee: a way to manage and access documents at CERN research lab

Info containing links to other info, accessible remotely through a standardized mechanism independent of the heterogeneity of the underlying machines

“Hypertext”

Web and HTTP: Terms


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 ... "hypertext" is also sometimes used to describe tables, images, and ... presentational content formats with integrated hyperlinks. ... ypertext is one of the key underlying concepts of the World Wide ... where Web pages are often written in the Hypertext Markup ... guage (HTML). As implemented on the Web, hypertext enables the ... y-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.

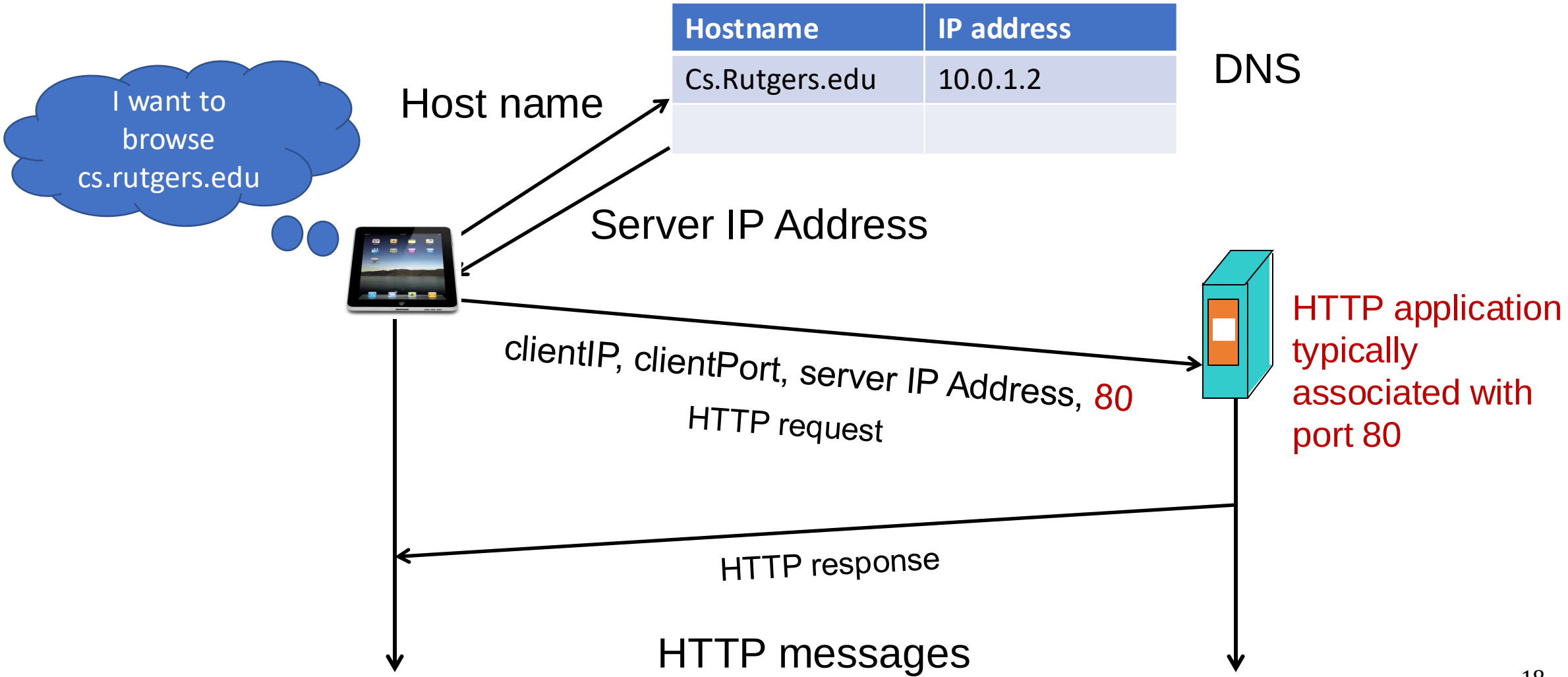
- HTTP stands for “HyperText Transfer Protocol”
- A web page consists of many objects
- Object can be HTML file, JPEG image, video stream chunk, audio file,…
- Web page consists of base HTML-file which embeds several objects
- Each object is addressable by a uniform resource locator (URL)
  - sometimes also referred to as uniform resource identifier (URI)
- Example URL:

`www.cs.rutgers.edu/~sn624/index.html`

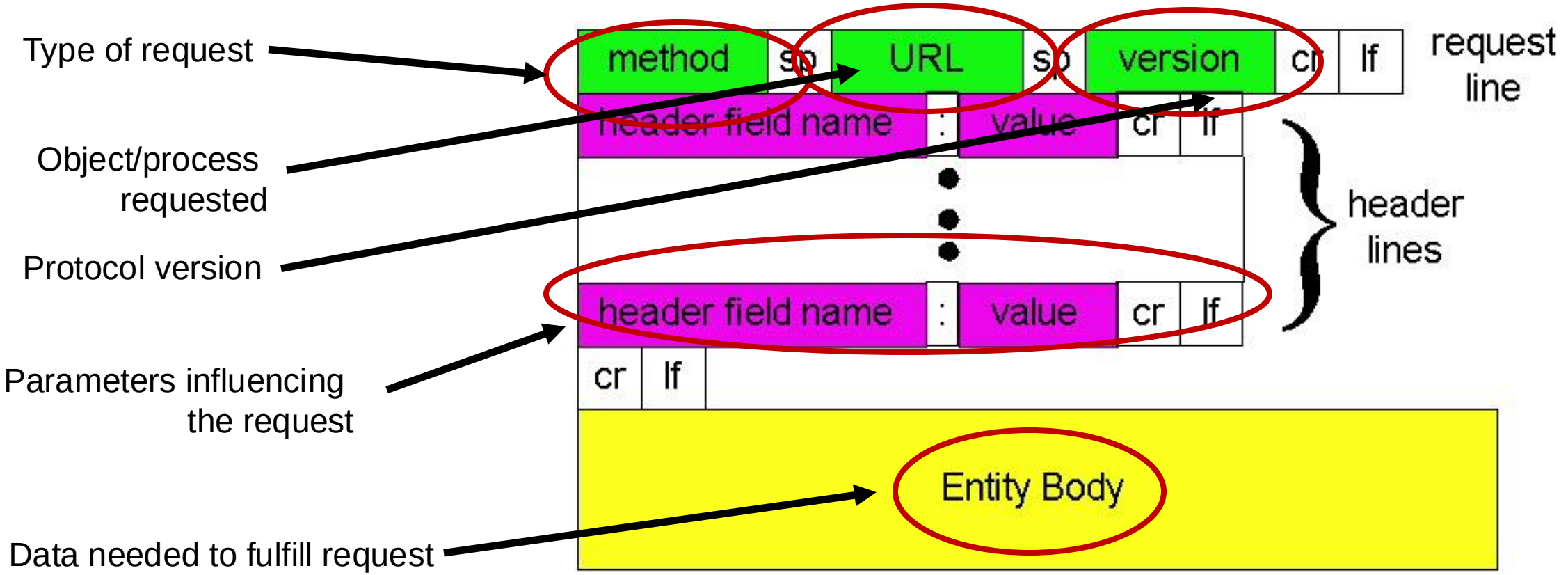
Domain/host name | path

HTTP Protocol

Client server protocol



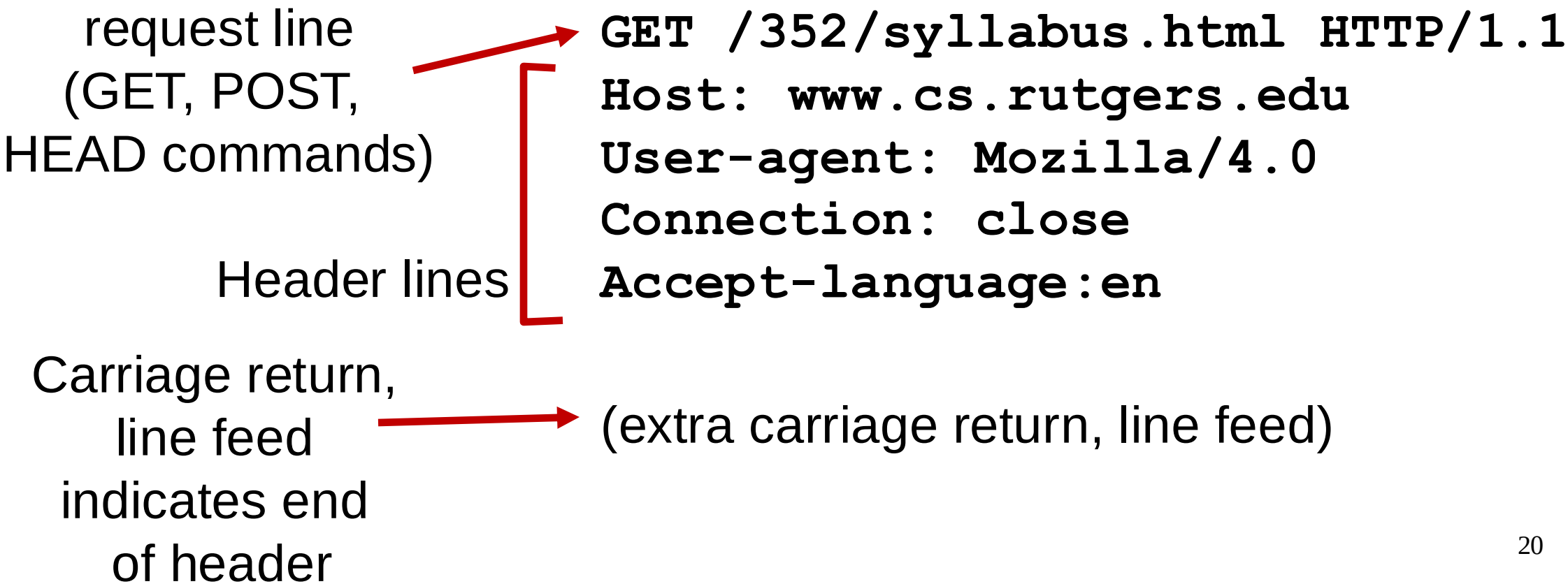
HTTP Request: Message Format



http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14

HTTP messages: request message

- ASCII (human-readable format)



The URL

- Universal Resource Locator: a way to name objects on server
- But can also name an application process on the server!
- Examples:
  - Data storage from data entered in web forms
  - Login pages
  - Web carts
- Providing almost any service requires data handling by running code at the server
  - Not just rendering “static” resources

HTTP method types

- GET
  - Get the resource specified in the requested URL (could be a process)
- POST
  - Send entities (specified in the entity body) to a data-handling process at the requested URL
- HEAD
  - Asks server to leave requested object out of response, but send the rest of the response
  - Useful for debugging
- PUT
  - Update a resource at the requested URL with the new entity specified in the entity body
- DELETE
  - Deletes file specified in the URL
- and other methods

Uploading form input: GET and POST

POST method:

- Web page often includes form input
- Input is uploaded to server in entity body

- Posted content not visible in the URL
  - Free form content (ex: images) can be posted since entity body interpreted as data bytes

GET method:

- Entity body is empty
- Input is uploaded in URL field of request line

- URL must contain a restricted set of characters

- Example:
  - http://site.com/form?first=jane&last=austen

Difference between POST and PUT

- POST: the URL of the request identifies the resource that processes the entity body
- PUT: the URL of the request identifies the resource that is contained in the entity body

https://tools.ietf.org/html/rfc2616

Difference between HEAD and GET

- GET: return the requested resource in the entity body of the response along with response headers (we'll see these shortly)
- HEAD: return all the response headers in the GET response, but without the resource in the entity body

https://tools.ietf.org/html/rfc2616

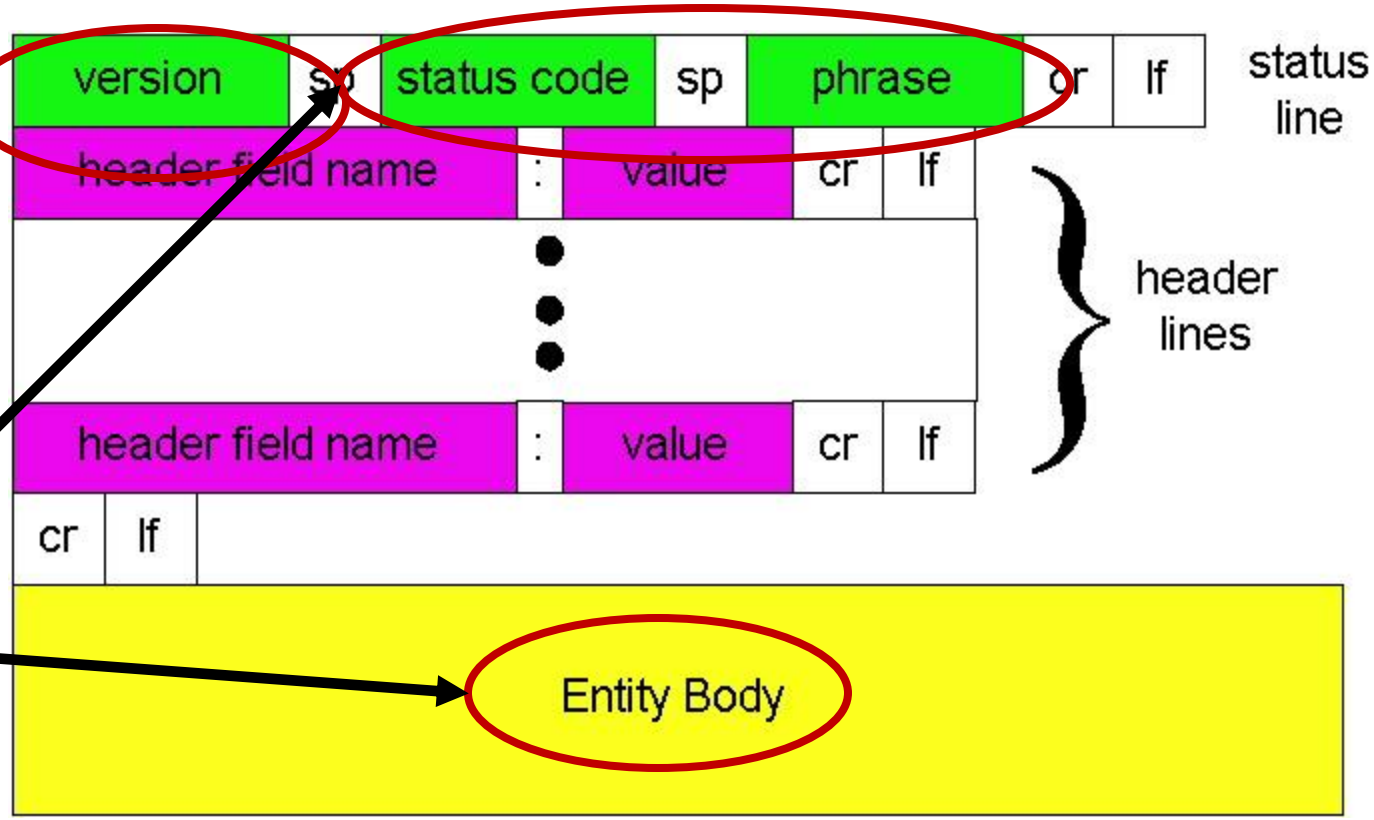
HTTP Response: General format

Unlike HTTP request,
No method name

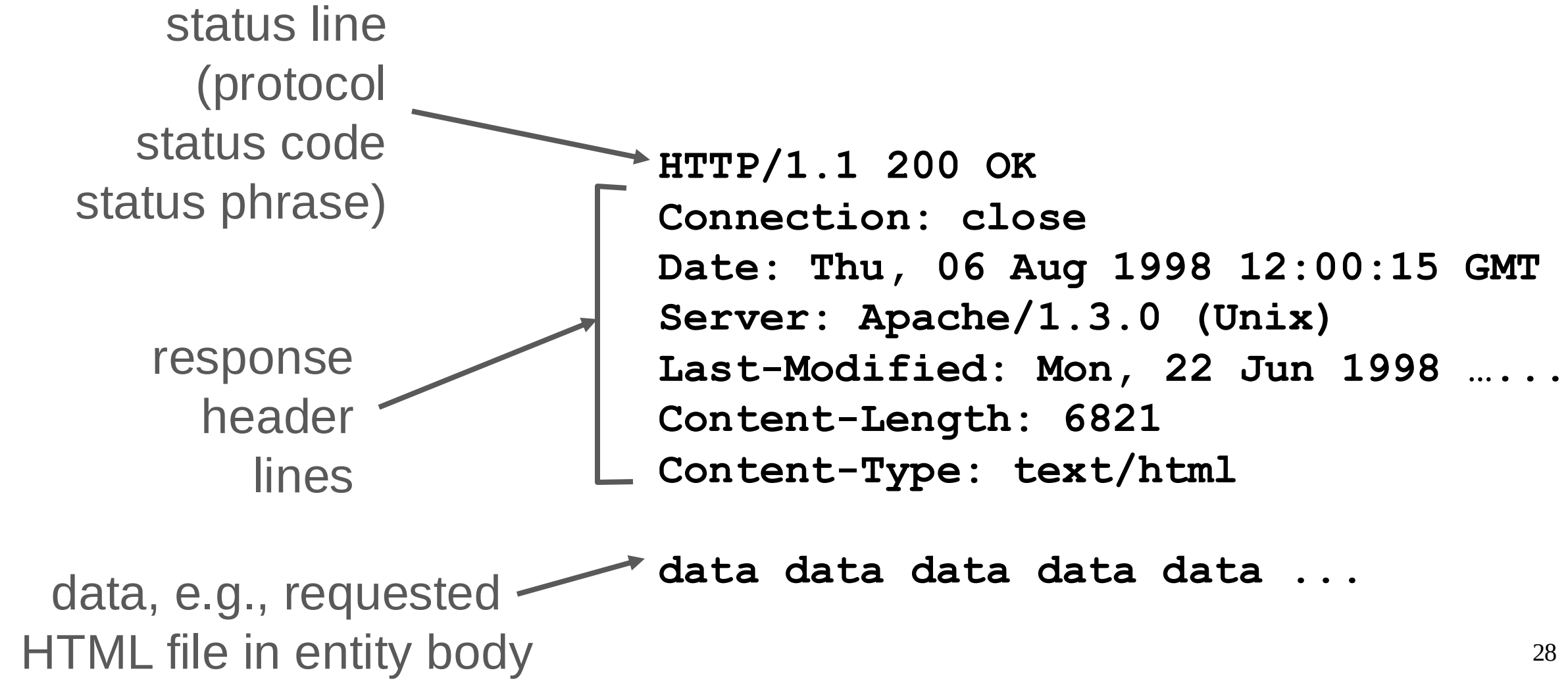
HTTP protocol version used by server

Was request successful? (or error condition)

Returned object data



HTTP message: response message



HTTP response status codes

In first line in server->client response message.

A few sample codes:

200 OK

- request succeeded, requested object later in this message

301 Moved Permanently

- requested object moved, new location specified later in this message (Location:)

403 Forbidden

- Insufficient permissions to access the resource

404 Not Found

- requested document not found on this server

505 HTTP Version Not Supported

Observing HTTP behaviors

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

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

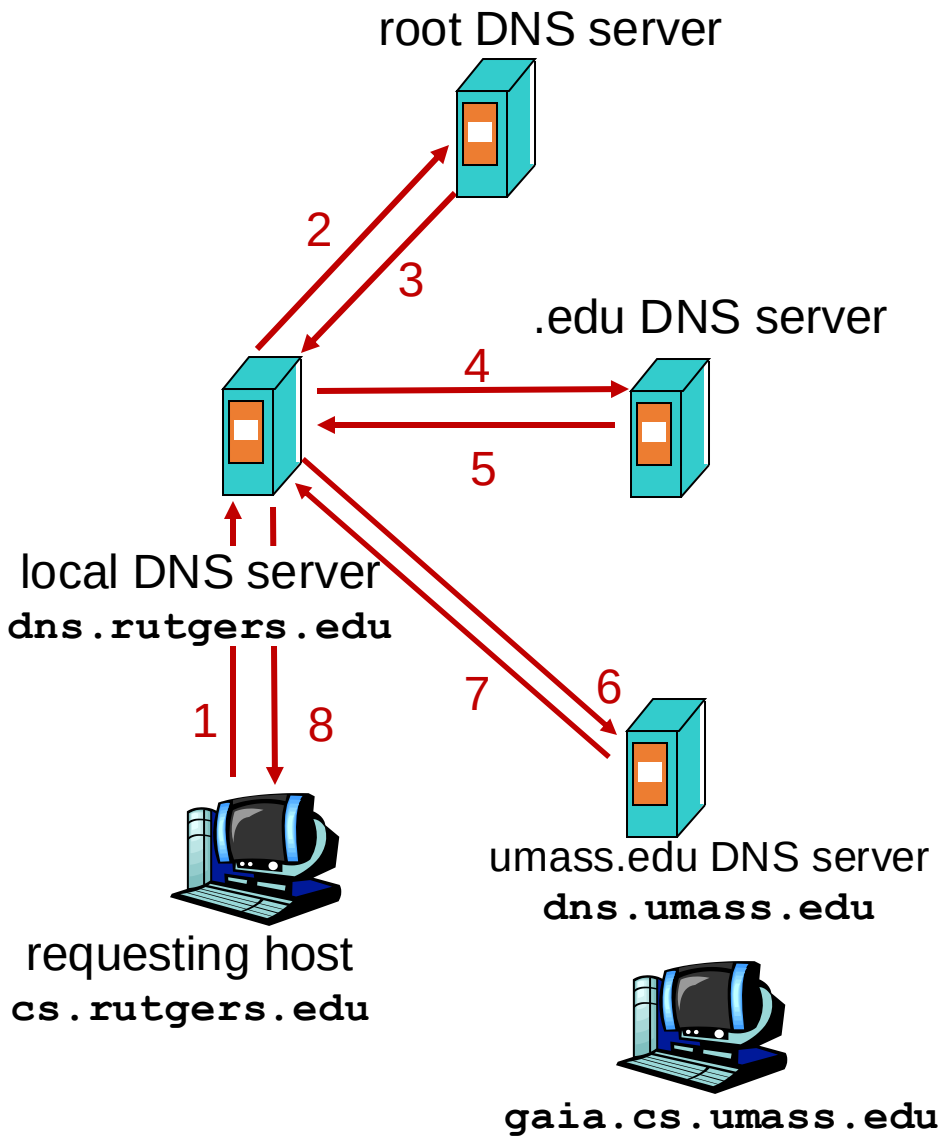
(followed by two enter's)

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

Example

- 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



Query type

- Iterative query

- Contacted server replies with name of server to contact

- “I don’t know this name, but ask this other server”

- Queries 2,4,6 are iterative from point of view of the local DNS server