HTTP request message

- two types of HTTP messages: *request, response*
- HTTP request message:
  - ASCII (human-readable format)

carriage return character

line-feed character

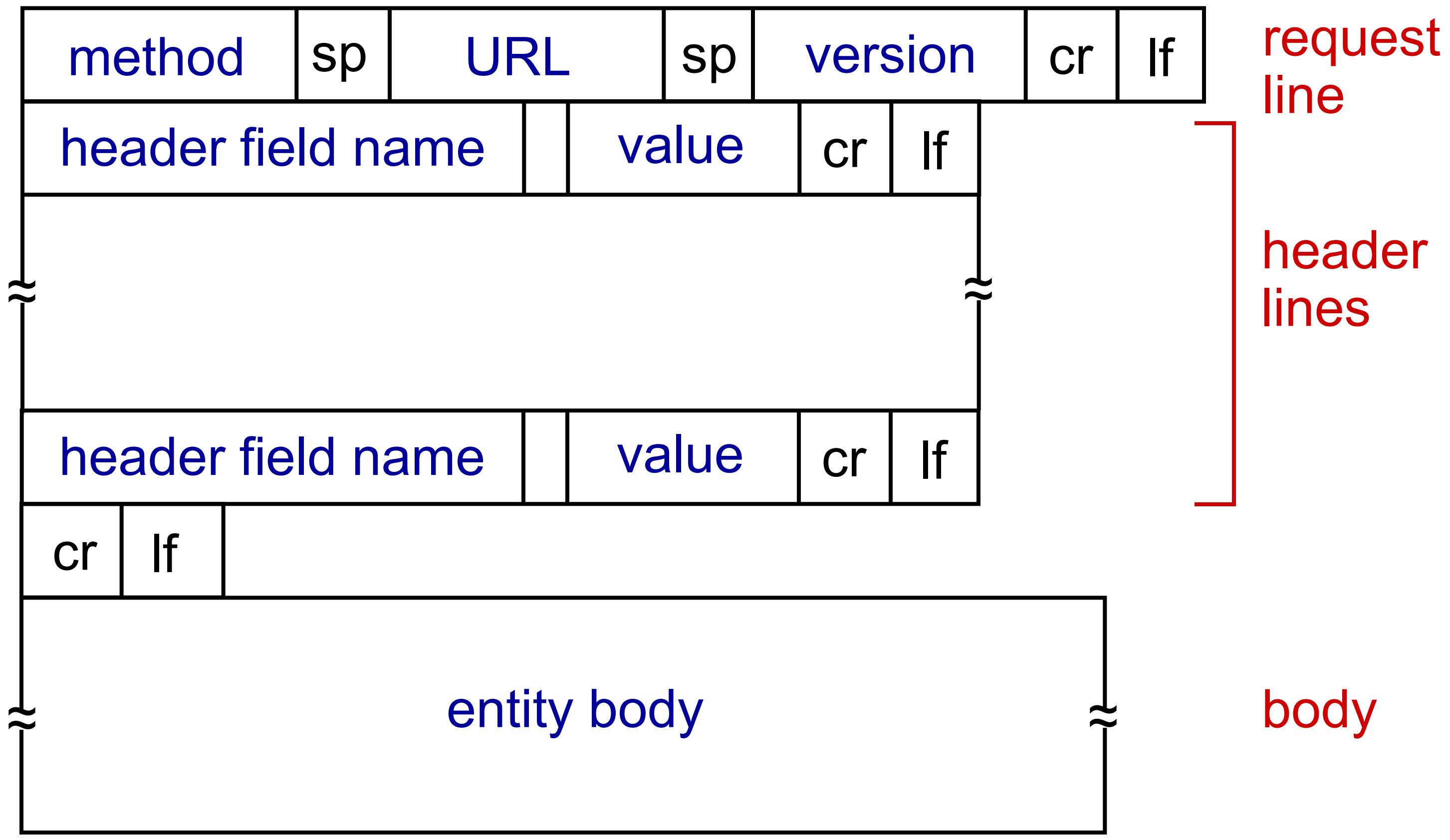
request line (GET, POST, HEAD commands)

header lines

carriage return, line feed at start of line indicates end of header lines

```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

HTTP request message: general format



Q1

- a) The document request was http://gaia.cs.umass.edu/cs453/index.html.

  The Host : field indicates the server's name and /cs453/index.html indicates the file name.

- b) The browser is running HTTP version 1.1, as indicated just before the first <cr><lf> pair.

- c) The browser is requesting a persistent connection, as indicated by the Connection: keep-alive.

Q1

- d) This is a trick question. This information is not contained in an HTTP message anywhere. So there is no way to tell this from looking at the exchange of HTTP messages alone. One would need information from the IP datagrams (that carried the TCP segment that carried the HTTP GET request) to answer this question.

- e) Mozilla/5.0. The browser type information is needed by the server to send different versions of the same object to different types of browsers.

DNS: services, structure

DNS services

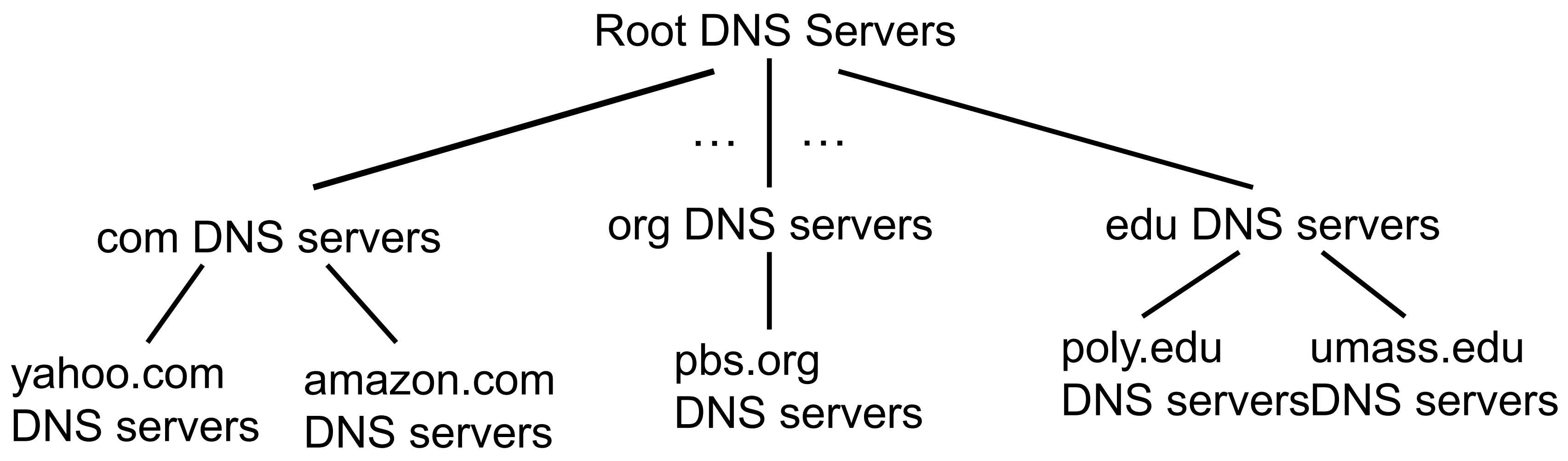
- hostname to IP address translation
- host aliasing
  - canonical, alias names
- mail server aliasing
- load distribution
  - replicated Web servers: many IP addresses correspond to one name

why not centralize DNS?

- single point of failure
- traffic volume
- distant centralized database
- maintenance

A: *doesn't scale!*

DNS: a distributed, hierarchical database



client wants IP for www.amazon.com; 1st approx:

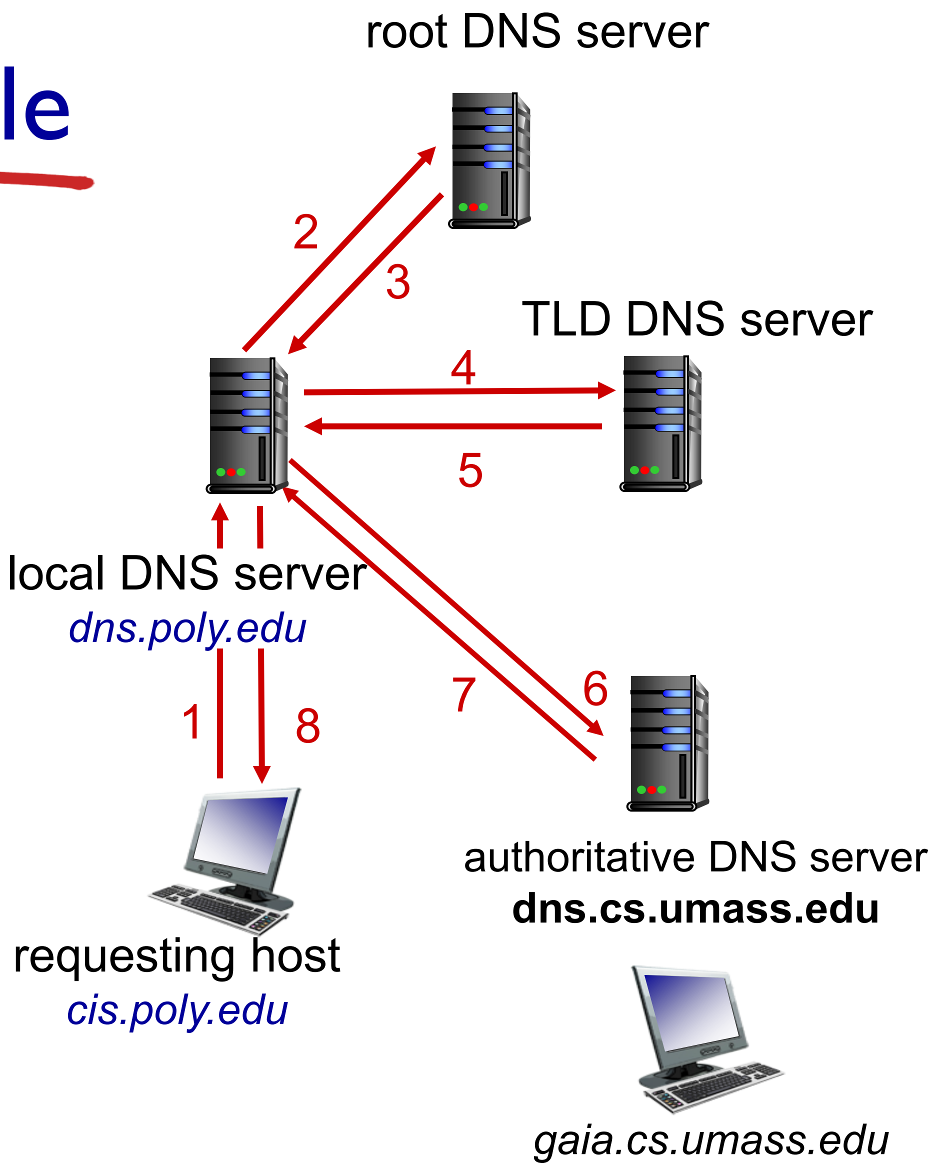
- client queries root server to find com DNS server
- client queries .com DNS server to get amazon.com DNS server
- client queries amazon.com DNS server to get IP address for www.amazon.com

DNS name resolution example

- host at cis.poly.edu wants IP address for gaia.cs.umass.edu

iterated query:

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



Q2

- As no DNS records are cached at any of the servers, all interactions are needed.

- 1) The total time to get the IP address is $RTT_L + 3\mathrm{RTT}_r$.

- 2) Just RTT_L.

Note: A TLD DNS server is a Top-Level Domain DNS server, which handles queries for a specific top-level domain (TLD) such as .edu, .com, or .org.

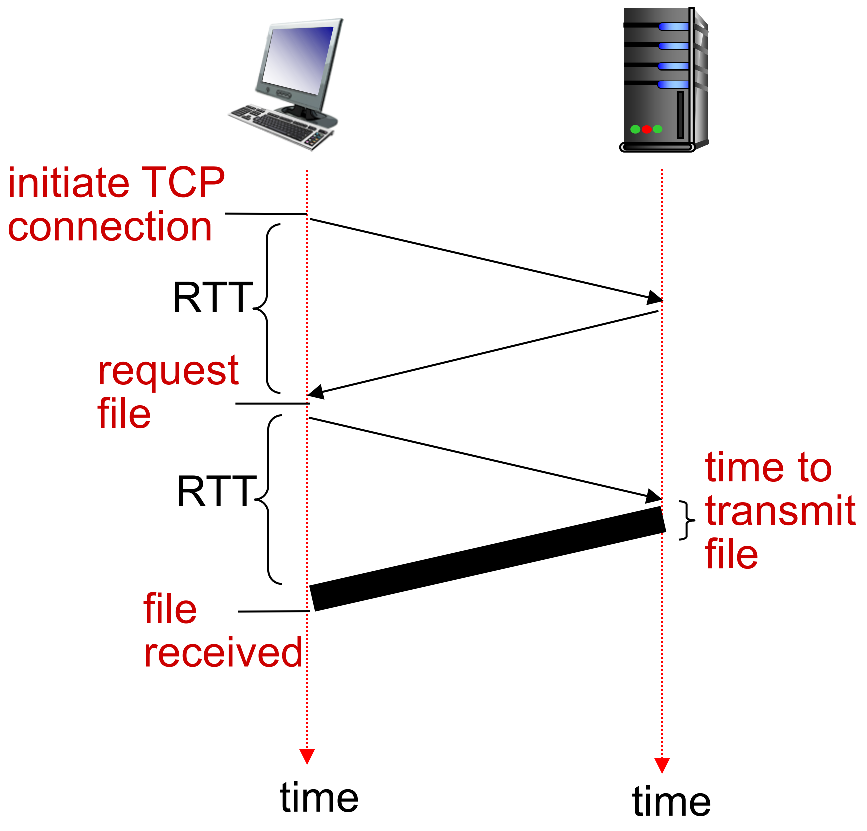
Non-persistent HTTP: response time

RTT (definition): time for a small packet to travel from client to server and back

HTTP response time:

- one RTT to initiate TCP connection
- one RTT for HTTP request and first few bytes of HTTP response to return
- file transmission time
- non-persistent HTTP response time =

  2RTT+ file transmission time



Persistent HTTP

non-persistent HTTP issues:

- requires 2 RTTs per object
- OS overhead for *each* TCP connection
- browsers often open parallel TCP connections to fetch referenced objects

persistent HTTP:

- server leaves connection open after sending response
- subsequent HTTP messages between same client/server sent over open connection
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

Q3a

- Time needed

= Time for DNS + (TCP connection setup time + request-response RTT for base HTML file) + (TCP connection setup time + request-response RTT for an object) * 8

= $(RTT_1 + \ldots + RTT_n) + 2RTT_0 + (2RTT_0) * 8$

= $(RTT_1 + \ldots + RTT_n) + 18RTT_0$

Q3b

- Time needed

= Time for DNS + (TCP connection setup time + request-response RTT for base HTML file) + (TCP connection setup time + request-response RTT for an object) * 2

$= (RTT_1 + \ldots + RTT_n) + 2RTT_0 + (2RTT_0) * 2$

$= (RTT_1 + \ldots + RTT_n) + 6RTT_0$

Q3c

- In persistent HTTP, after the TCP connection has been setup, the nine objects (i.e., one base HTML file plus eight very small objects) have to be requested one after another. Thus,

- Time needed

= Time for DNS + TCP connection setup time + request-response RTT * 9

= $(RTT_1 + \ldots + RTT_n) + RTT_0 + RTT_0 * 9$

= $(RTT_1 + \ldots + RTT_n) + 10RTT_0$