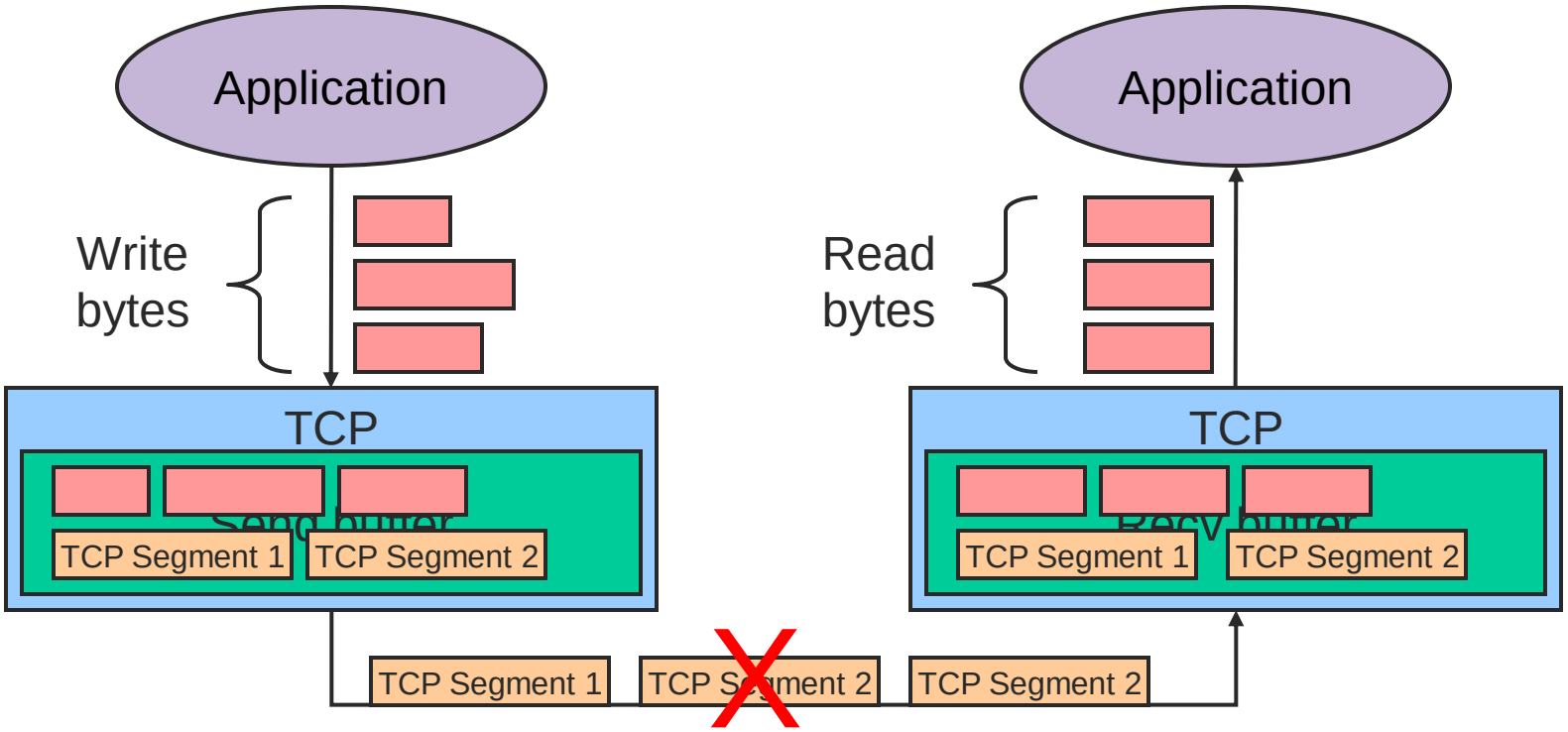
Application Layer

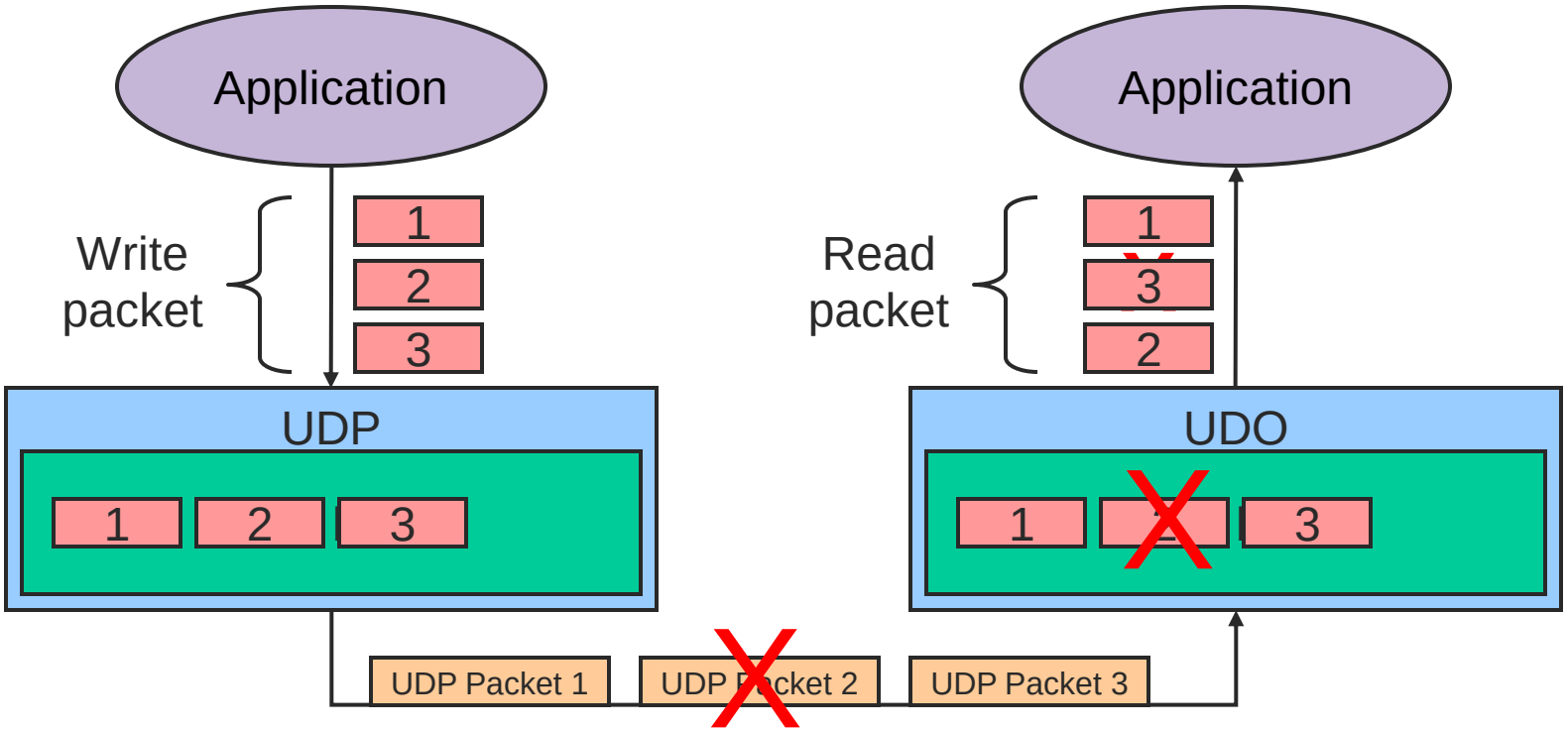
Using TCP and UDP

- TCP: Byte Stream
  - Application passes data to TCP
  - TCP collects data to send larger packets
  - TCP has no knowledge of/pays no attention to application data boundaries
  - TCP receiver gets packet
  - Receiving application reads data
  - Data is guaranteed to be in order and complete
- UDP: Packet Stream
  - Application passes packet to UDP
  - UDP sends packet
  - UDP receiver gets packet
  - Receiving application reads packet
  - Packets have no ordering or delivery guarantees

TCP Byte Stream



UDP Packet Stream

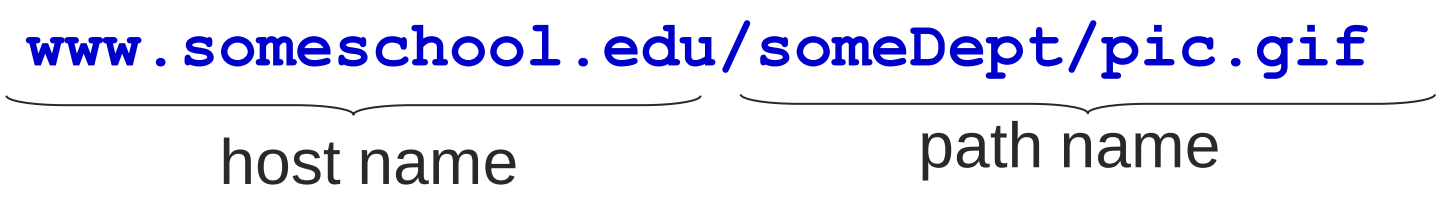


Networked Applications

- All networked applications use “application level” protocols to communicate
- Examples
  - HTTP
  - FTP
  - SMTP
  - …

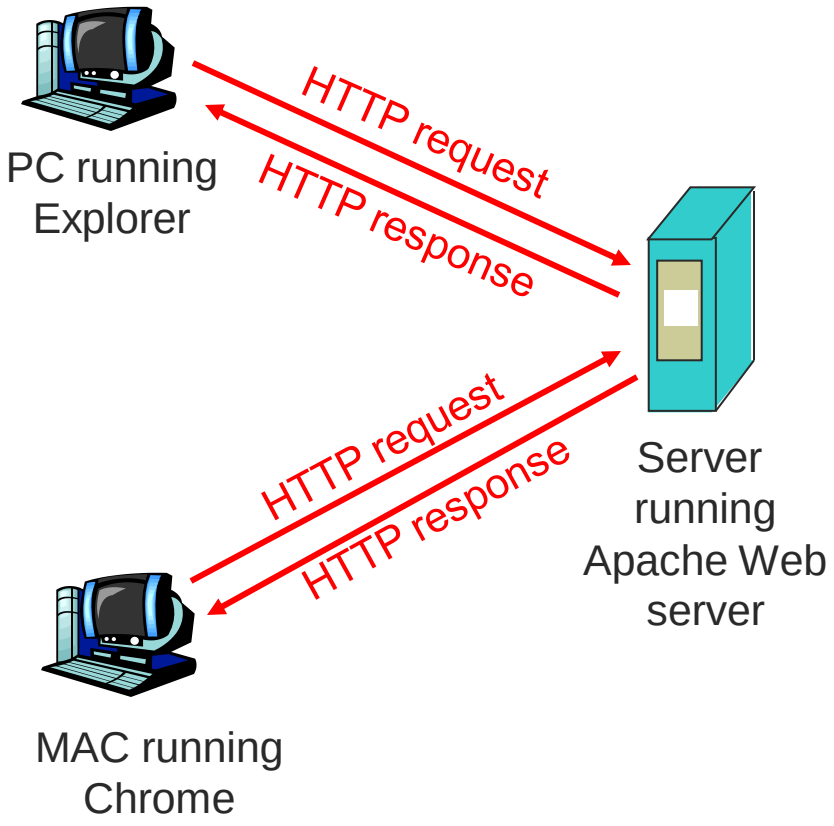
Web and HTTP

- Web pages consist of
  - Objects
    - HTML files, JPEG images, Java applets, audio files,…
  - Base HTML-file
    - Includes several referenced objects
- Each object is addressable by a URL
- Example URL:



HTTP (Hypertext Transfer Protocol)

- Web's application layer protocol
- Client/server model
  - Client
    - Browser that requests, receives, "displays" Web objects
  - Server
    - Web server sends objects in response to requests



HTTP

- Uses TCP
  - Client initiates TCP connection (creates socket) to server, port 80
  - Server accepts TCP connection from client
  - HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
  - TCP connection closed
- Stateless
  - Server maintains no information about past client requests

HTTP Connections

- Nonpersistent HTTP
  - At most one object is sent over a TCP connection
- Persistent HTTP
  - Multiple objects can be sent over single TCP connection between client and server

Nonpersistent HTTP

- User enters URL
  - Text plus references to 10 jpeg images

`www.someschool.edu/someDepartment/home.index`

1a. HTTP client initiates TCP connection to HTTP server at **`www.someschool.edu`** on port 80

2. HTTP client sends HTTP *request message* (containing URL) into TCP socket. Message indicates that client wants object **`someDepartment/home.index`**

1b. HTTP server at host **`www.someschool.edu`** waiting for TCP connection at port 80. "accepts" connection, notifying client

3. HTTP server receives request message, forms *response message* containing requested object, and sends message into its socket

time

Nonpersistent HTTP

3. HTTP server receives request message, forms *response message* containing requested object, and sends message into its socket

4. HTTP server closes TCP connection.

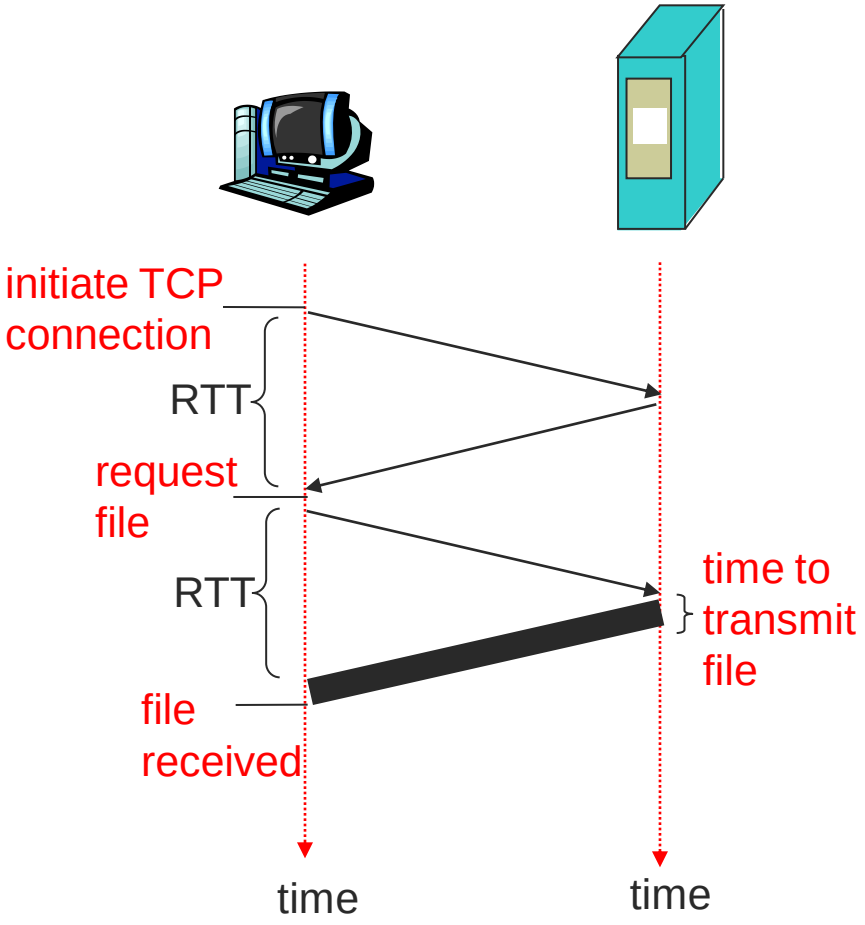
5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects

6. Steps 1-5 repeated for each of 10 jpeg objects

time

Non-Persistent HTTP: Response Time

- RTT
  - Time for a small packet to travel from client to server and back
- 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
  - = 2RTT+transmit time

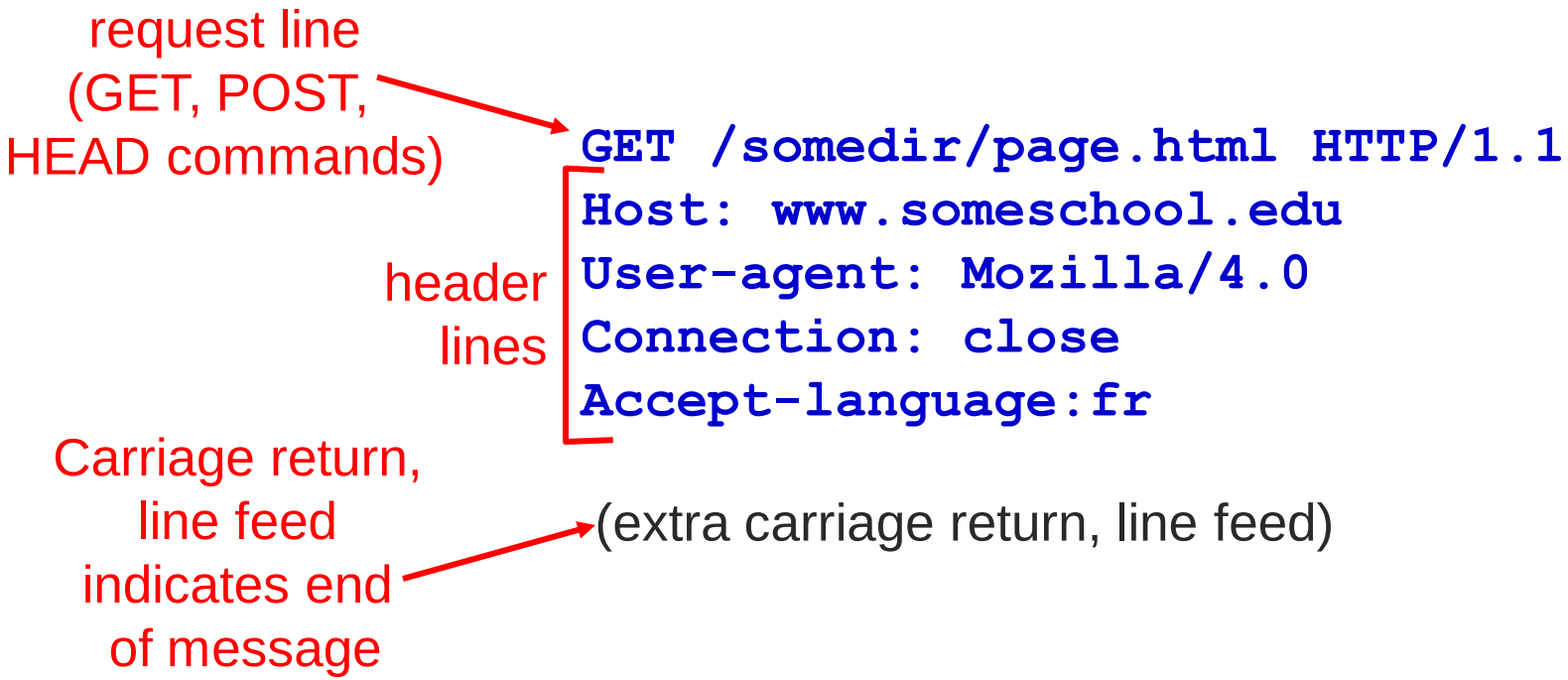


Persistent HTTP

- Nonpersistent HTTP
  - 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

HTTP Request Message

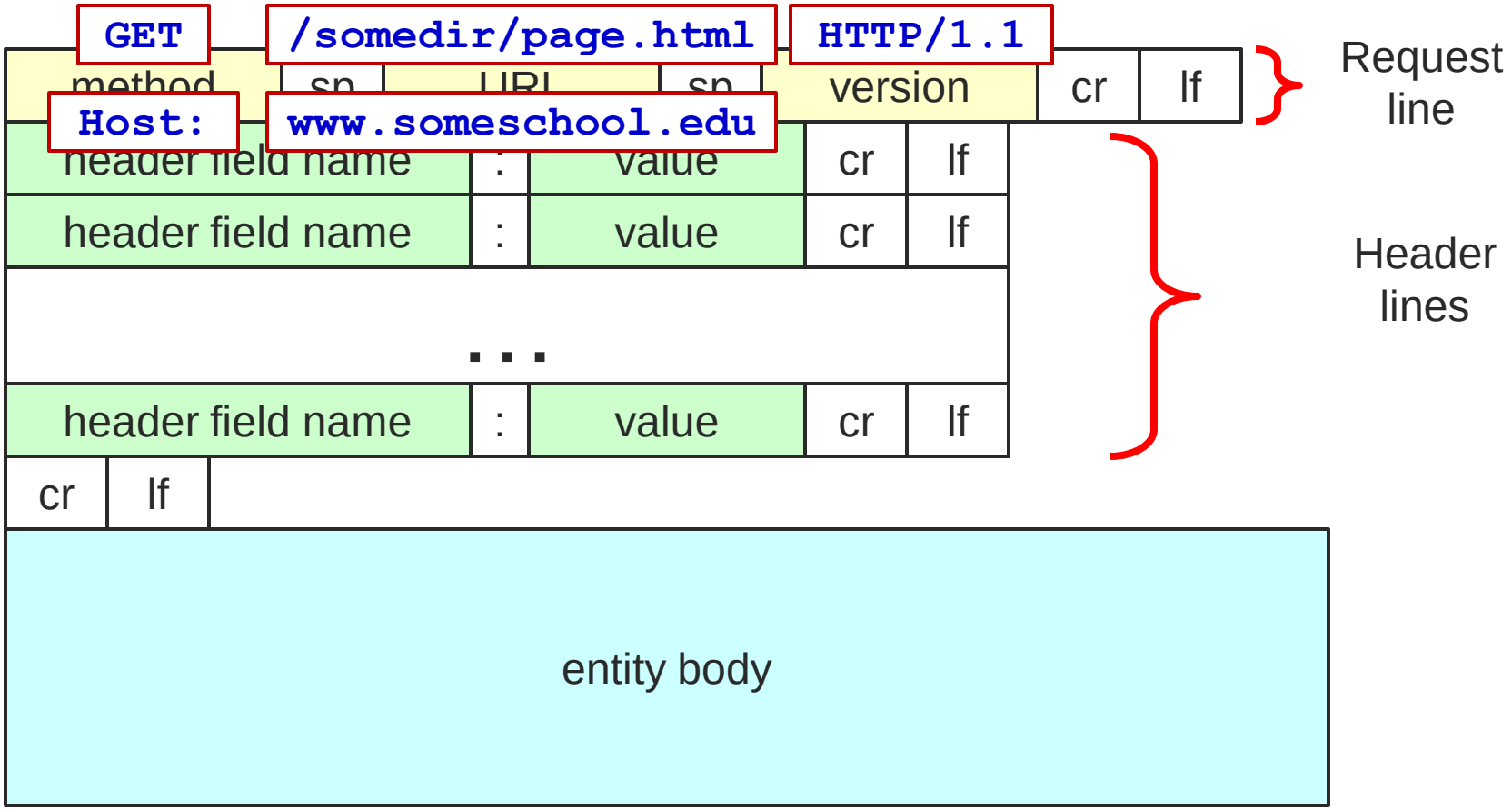
- Two types of HTTP messages: *request*, *response*
  - ASCII (human-readable format)
- HTTP request message:



Method Types

- HTTP/1.0
  - GET
  - POST
  - HEAD
    - Asks server to leave requested object out of response
- HTTP/1.1
  - GET, POST, HEAD
  - PUT
    - Uploads file in entity body to path specified in URL field
  - DELETE
    - Deletes file specified in the URL field

HTTP Request Message: General Format

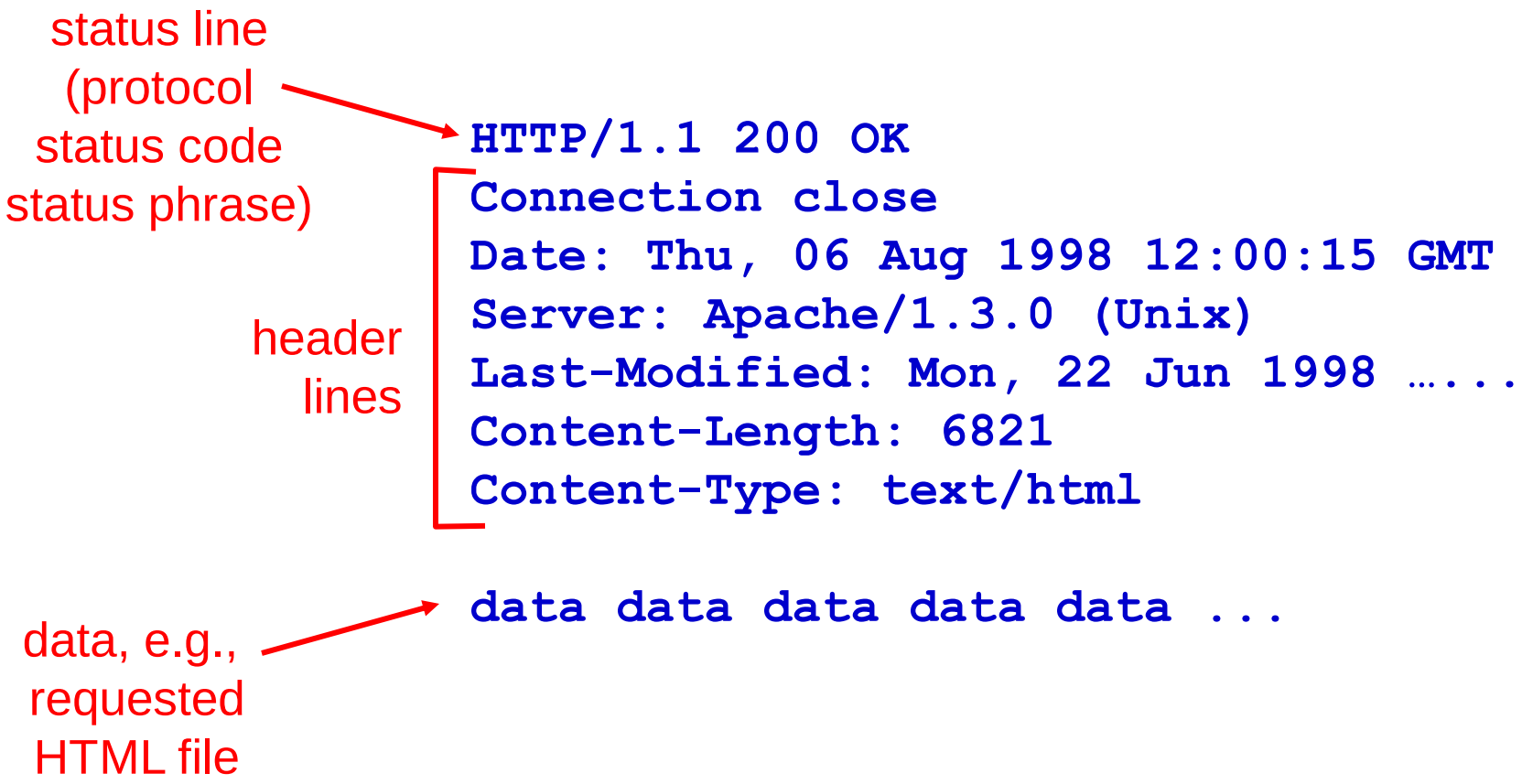


Uploading Form Input

- Post method
  - Web page often includes form of input
  - Input is uploaded to server in entity body
- URL method
  - Uses GET method
  - Input is uploaded in URL field of request line:

    `www.somesite.com/animalsearch?monkeys&banana`

HTTP 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:), client automatically retrieves new URL
400	Bad Request	request message not understood by server
404	Not Found	requested document not found on this server
505	HTTP Version Not Supported	

Trying out HTTP (client side) For Yourself

1. Telnet to your favorite Web server

```
telnet cs.illinois.edu 80
```

Opens TCP connection to port 80 (default HTTP server port) at `cs.illinois.edu`. Anything typed in sent to port 80 at `cs.illinois.edu`

2. Type in a GET HTTP request

```
GET /index.html HTTP/1.0
```

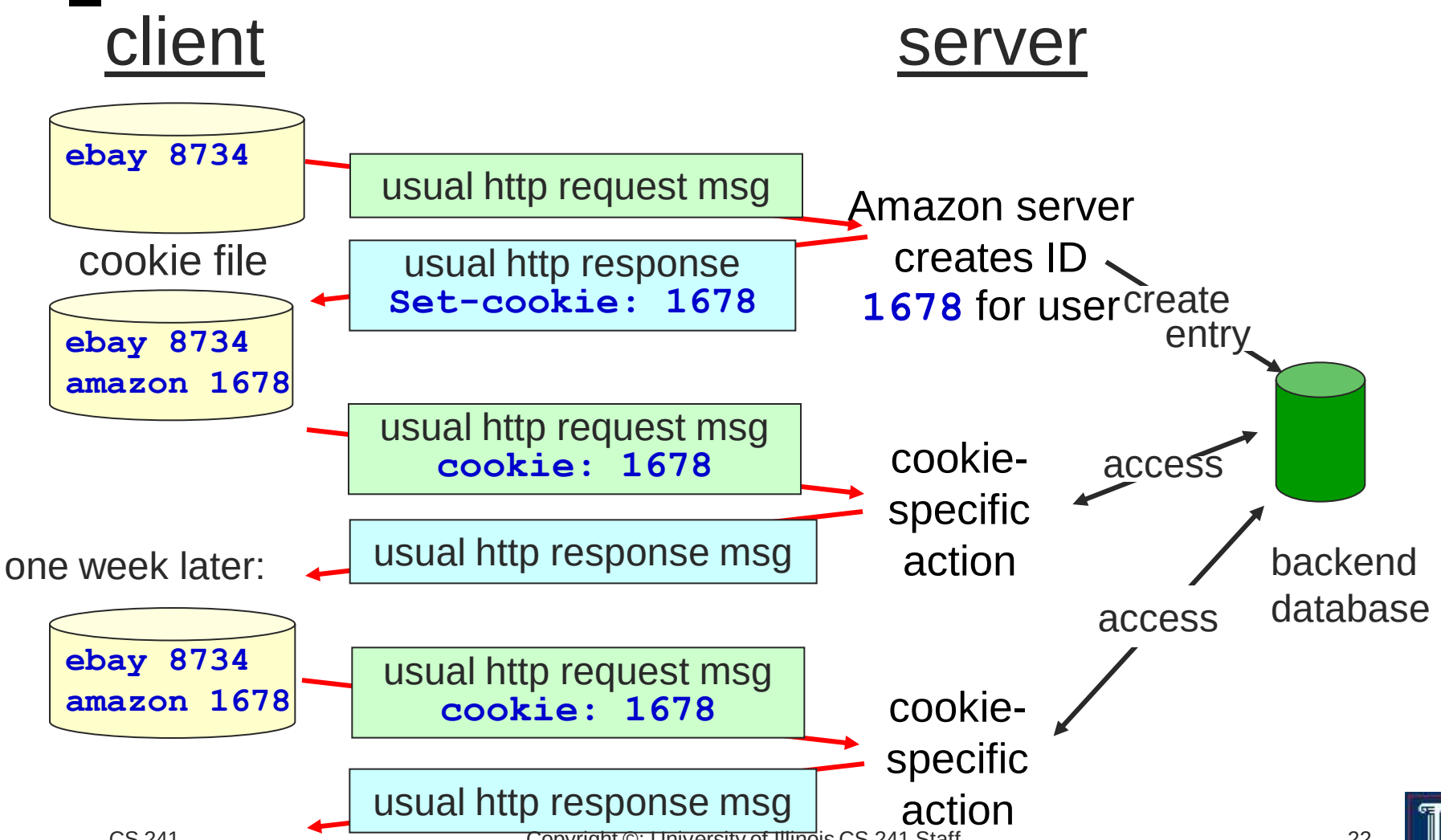
By typing this in (hit carriage return twice), you send this minimal (but complete) GET request to HTTP server

3. Look at response message sent by HTTP server!

User-server State: Cookies

- Many major Web sites use cookies
- Four components
  1. Cookie header line of HTTP response message
  2. Cookie header line in HTTP request message
  3. Cookie file kept on user's host, managed by user's browser
  4. Back-end database at Web site
- Example
  - Alice always accesses Internet from PC
  - Visits specific e-commerce site for first time
  - When initial HTTP requests arrives at site, site creates:
    - unique ID
    - entry in backend database for ID

Cookies: Keeping "State"



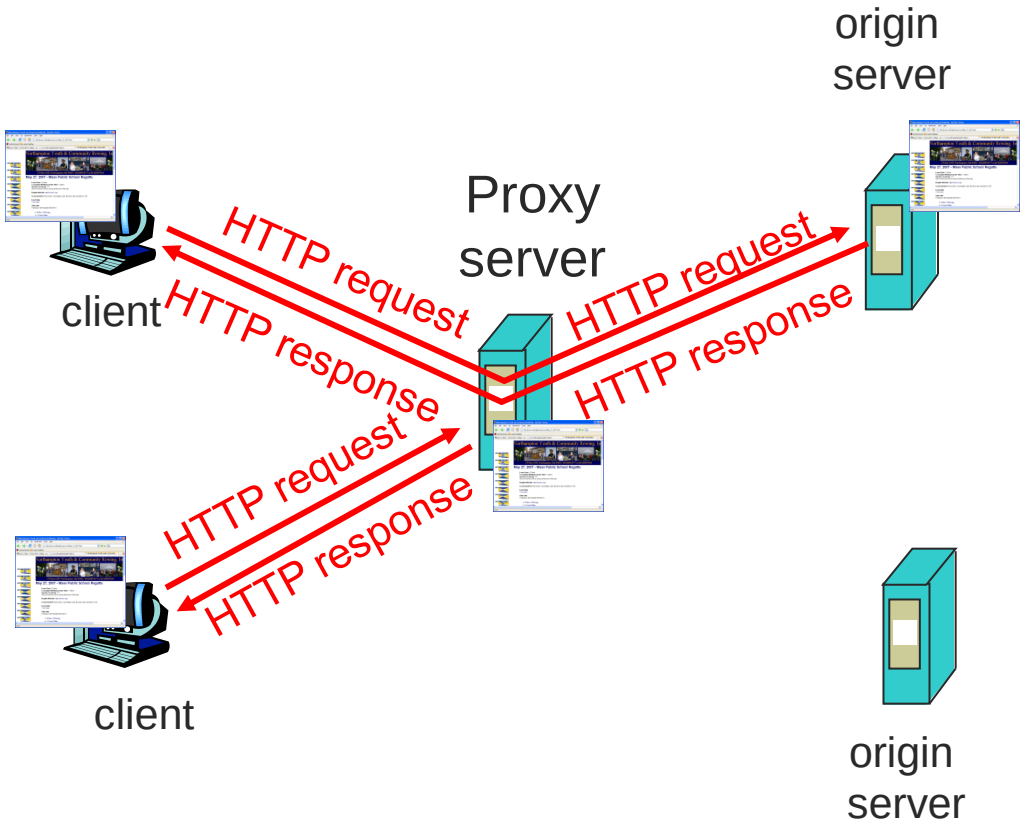
Cookies

- What cookies can bring
  - Authorization
  - Shopping carts
  - Recommendations
  - User session state (Web e-mail)
- How to keep “state”
  - Protocol endpoints: maintain state at sender/receiver over multiple transactions
  - cookies: http messages carry state
- Cookies and privacy:
  - Cookies permit sites to learn a lot about you
  - You may supply name and e-mail to sites

Web Caches (Proxy Server)

- Goal
  - Satisfy client request without involving origin server
- Caching
  - User sets browser: Web accesses via cache
  - Browser sends all HTTP requests to cache
    - Object in cache: cache returns object
    - Else cache requests object from origin server, then returns object to client

Web Caches (Proxy Server)

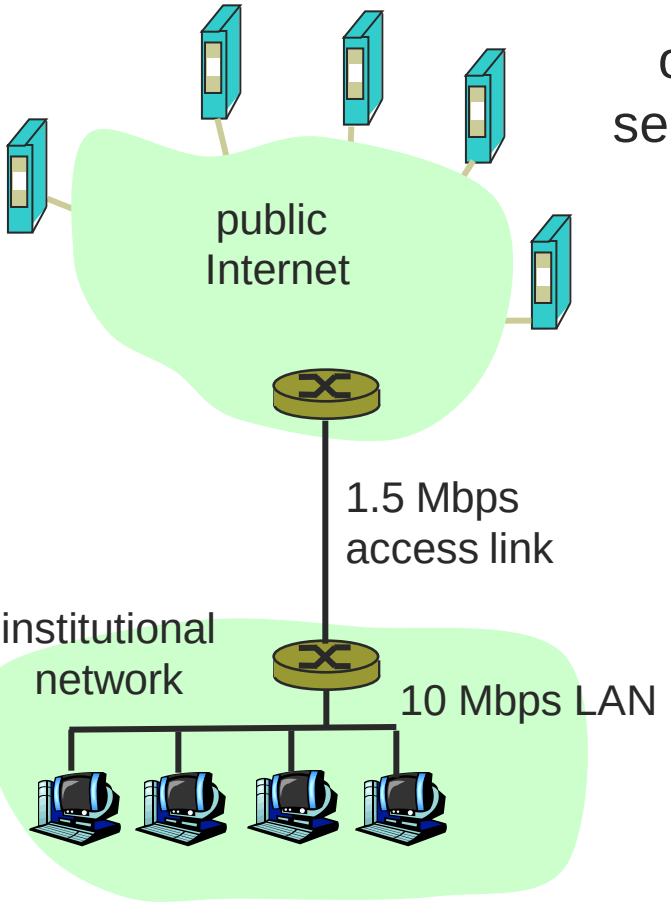


More about Web Caching

- Cache
  - Acts as both client and server
  - Typically installed by ISP (university, company, residential ISP)
- Why Web caching?
  - Reduce response time for client request
  - Reduce traffic on an institution's access link.
- Internet dense with caches
  - Enables "poor" content providers to effectively deliver content

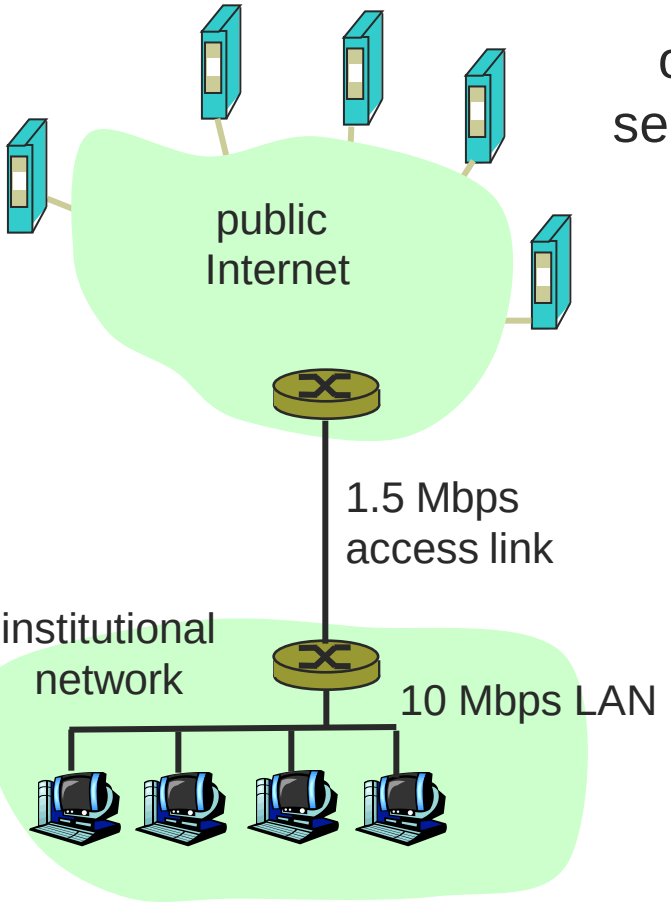
Caching Example

- Assumptions
  - Average object size = 100,000 bits
  - Average request rate from institution's browsers to origin servers = 15/sec
  - Delay from institutional router to any origin server and back to router = 2 sec



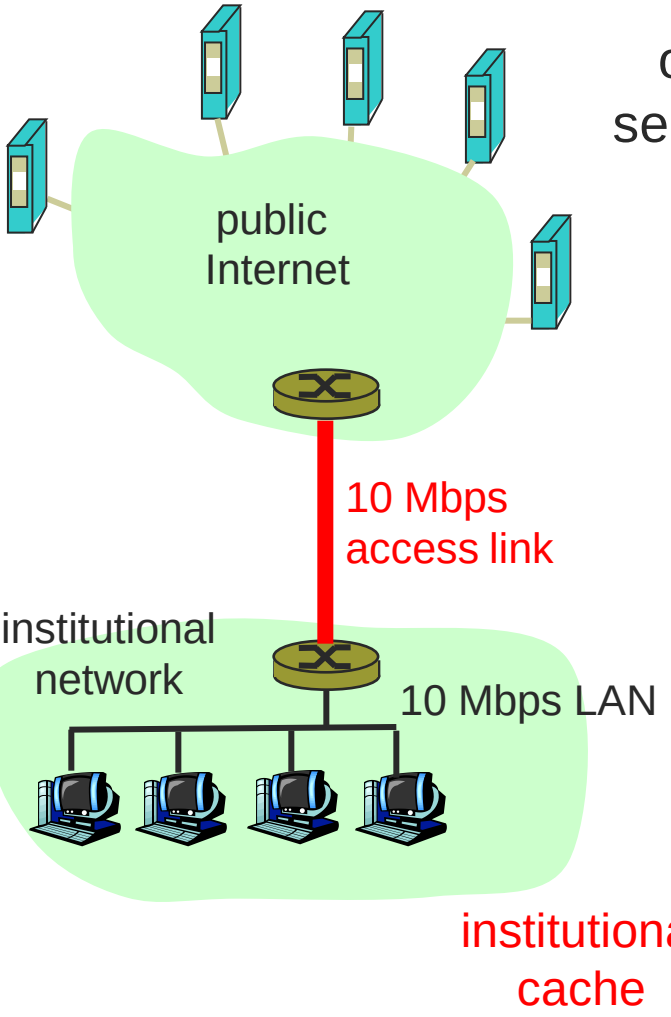
Caching Example

- Consequences
  - Utilization on LAN = 15%
  - Utilization on access link = 100%
  - total delay = Internet delay + access delay + LAN delay
  - = 2 sec + minutes + milliseconds



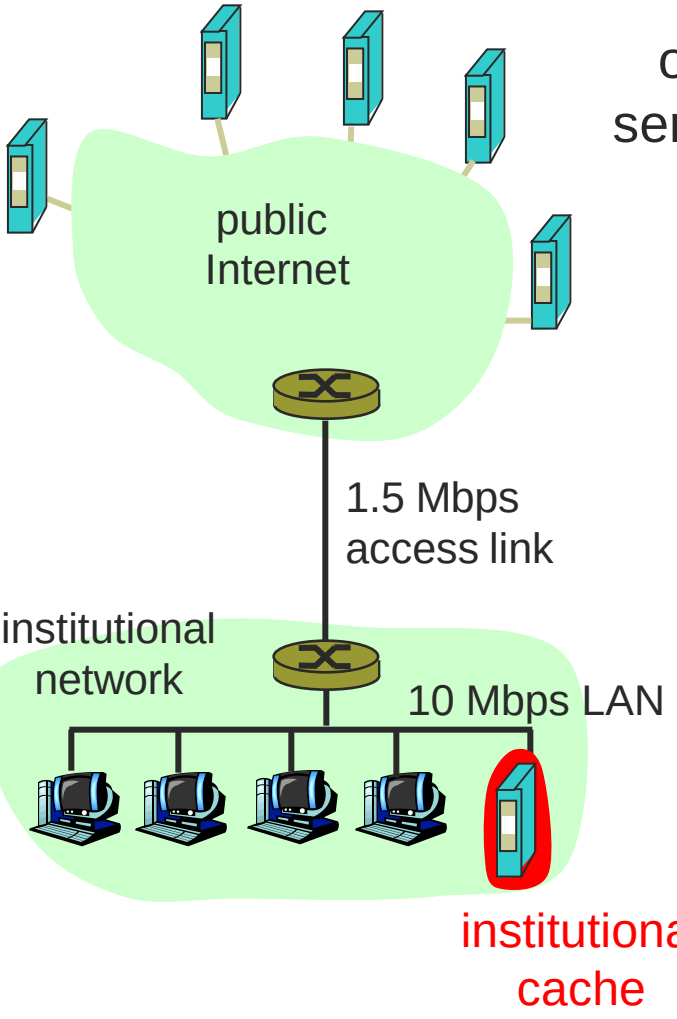
Caching Example

- Possible solution
  - Increase bandwidth of access link to 10 Mbps
- Consequence
  - Utilization on LAN = 15%
  - Utilization on access link = 15%
  - Total delay = Internet delay + access delay + LAN delay
  - = 2 sec + msecs + msecs
  - Often a costly upgrade



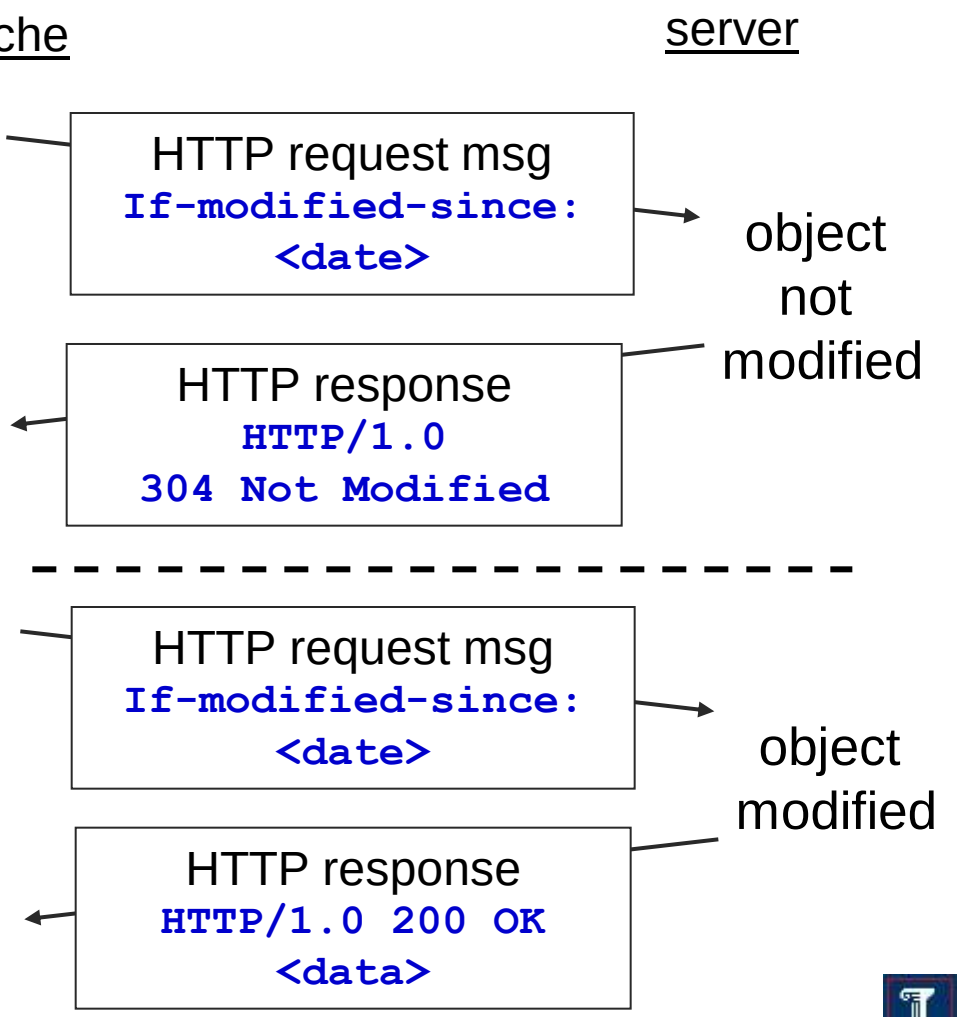
Caching Example

- Possible solution: Cache
  - Assume hit rate is 0.4
    - 40% satisfied immediately
    - 60% satisfied by origin server
- Consequence
  - Utilization on access link = 60%, resulting in negligible delays (say 10 msec)
  - Total avg delay = Internet delay + access delay + LAN delay
    = .6*(2.01) secs + .4*milliseconds < 1.4 secs



Practicalities: Conditional GET

- Goal
  - Don't send if cache has up-to-date version
- Cache
  - Specify date of cached copy in HTTP request
    `If-modified-since: <date>`
- Server
  - Response contains no object if up-to-date:
  - `HTTP/1.0 304 Not Modified`



No Free Lunch: Problems of Web Caching

- The major issue: maintaining consistency
- Two ways
  - Pull
    - Web caches periodically polls the web server to see if a document is modified
  - Push
    - Server gives a copy of a web page to a web cache,
    - Sign a lease with an expiration time
    - If web page is modified before the lease, server notifies cache

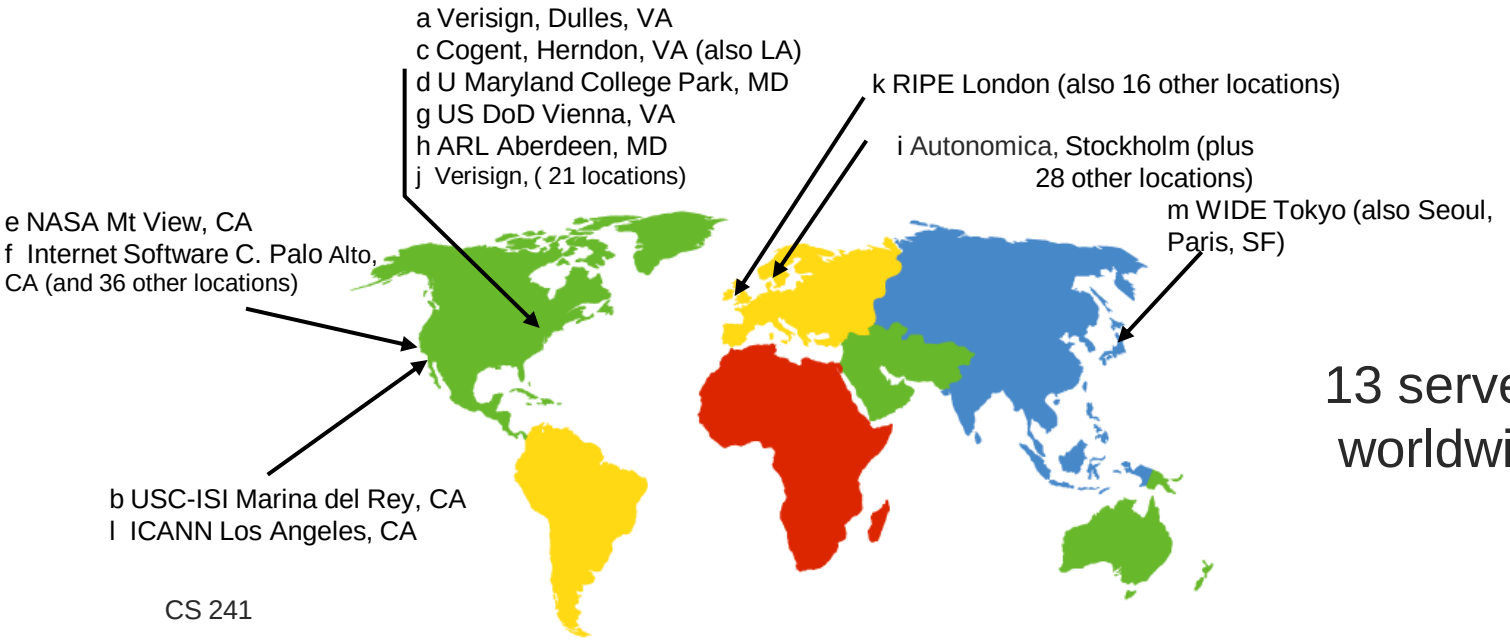
Which solution would you implement?

DNS: Domain Name System

- Internet hosts
  - IP address (32 bit)
    - Used for addressing datagrams
  - Host name (e.g., ww.yahoo.com)
    - Used by humans
- DNS: provides translation between host name and IP address
  - Distributed database implemented in hierarchy of many name servers
  - Distributed for scalability & reliability

DNS

- DNS services
  - Hostname to IP address translation
  - Host aliasing
    - Canonical, alias names
  - Mail server aliasing
  - Load distribution
    - Replicated Web servers: set of IP addresses for one canonical name



13 servers worldwide

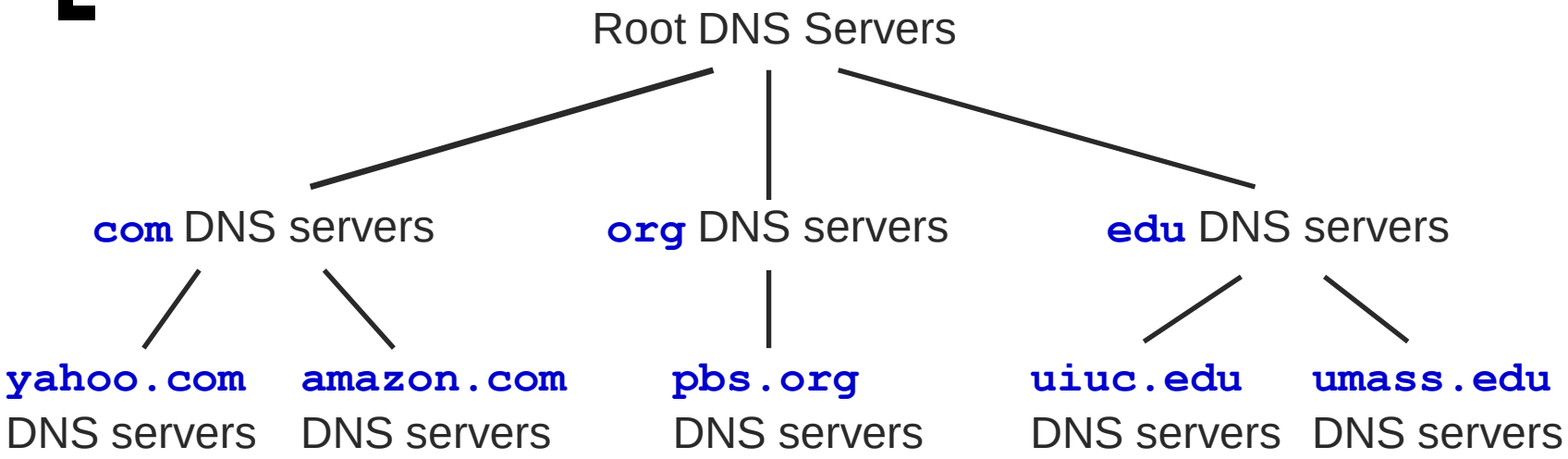
DNS

- Why not centralize DNS?
  - Single point of failure
  - Traffic volume
  - Distant centralized database
  - Maintenance
- Doesn't scale!
- Root name server
  - Contacted by local name server that can not resolve name
  - Contacts authoritative name server if mapping not known
  - Gets mapping and returns it to local name server

TLD and Authoritative Servers

- Top-level domain (TLD) servers
  - Responsible for **`com`**, **`org`**, **`net`**, **`edu`**, etc, and all top-level country domains **`uk`**, **`fr`**, **`ca`**, **`jp`**.
    - Network Solutions maintains servers for **`com`** TLD
    - Educause for **`edu`** TLD
- Authoritative DNS servers
  - Organization's DNS servers
  - Provide authoritative hostname to IP mappings for organization's servers (e.g., Web, mail).
  - Can be maintained by organization or service provider

Distributed, Hierarchical Database



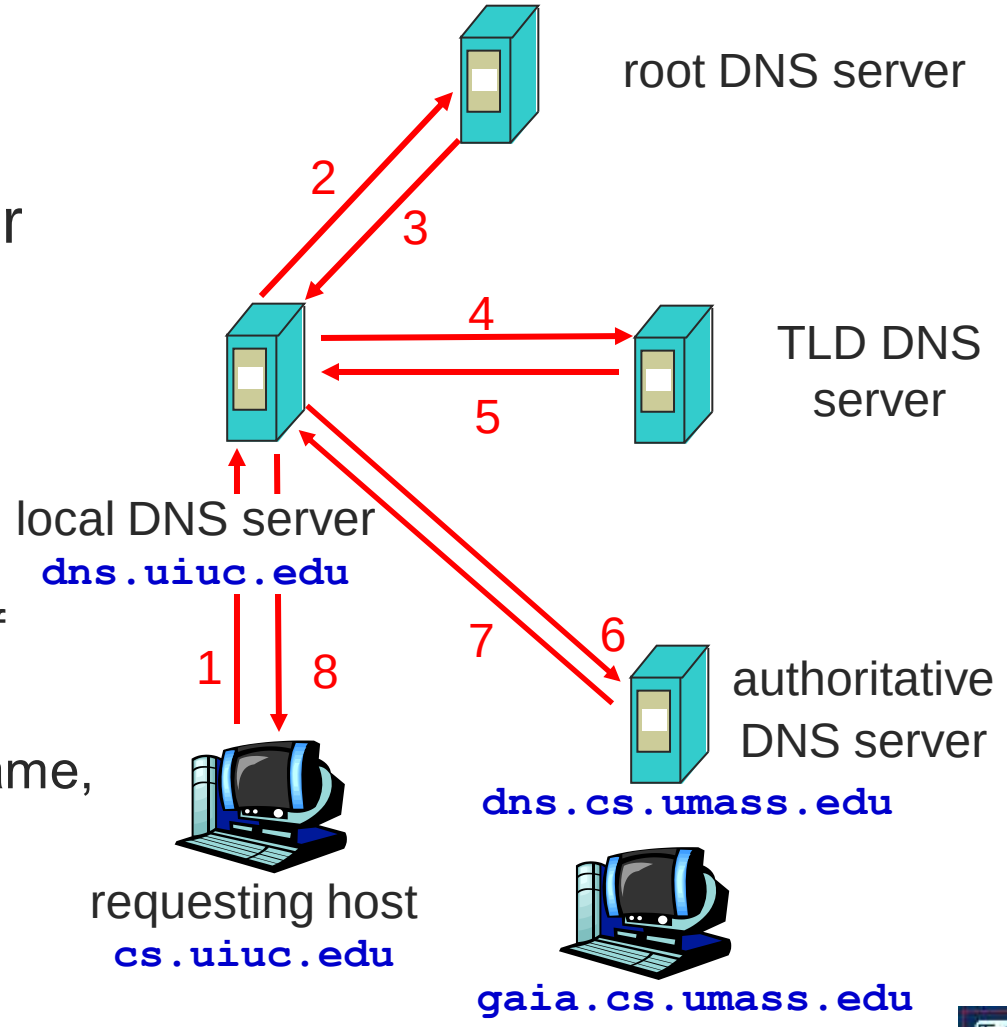
- Client wants IP for www.amazon.com
  - Client queries a root server to find **com** DNS server
  - Client queries **com** DNS server to get **amazon.com** DNS server
  - Client queries **amazon.com** DNS server to get IP address for **www.amazon.com**

Local Name Server

- One per ISP (residential ISP, company, university)
  - Also called “default name server”
- When host makes DNS query, query is sent to its local DNS server
  - Acts as proxy, forwards query into hierarchy
  - Reduces lookup latency for commonly searched hostnames

DNS name resolution example

- Host at cs.uiuc.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"

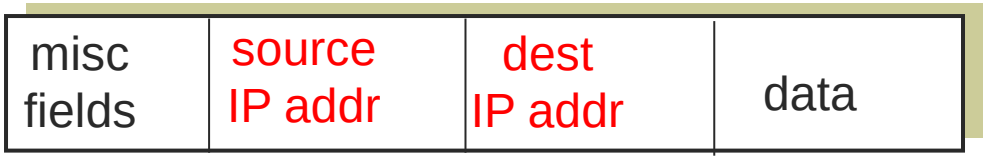


DNS: Caching

- Once (any) name server learns mapping, it caches mapping
  - Cache entries timeout (disappear) after some time
  - TLD servers typically cached in local name servers
    - Thus root name servers not often visited

IP Routing

- Quick background on IP datagrams



- Private IP network
  - IP network that is not directly connected to the Internet
  - Not registered and not guaranteed to be globally unique

IP Routing

- Problem
  - Depletion of IP addresses
- Solutions
  - Long term: IP v6
  - Short term CIDR (Classless InterDomain Routing)
  - Short term: NAT
    - Hide a number of hosts behind a single IP address

NAT: Network Address Translation

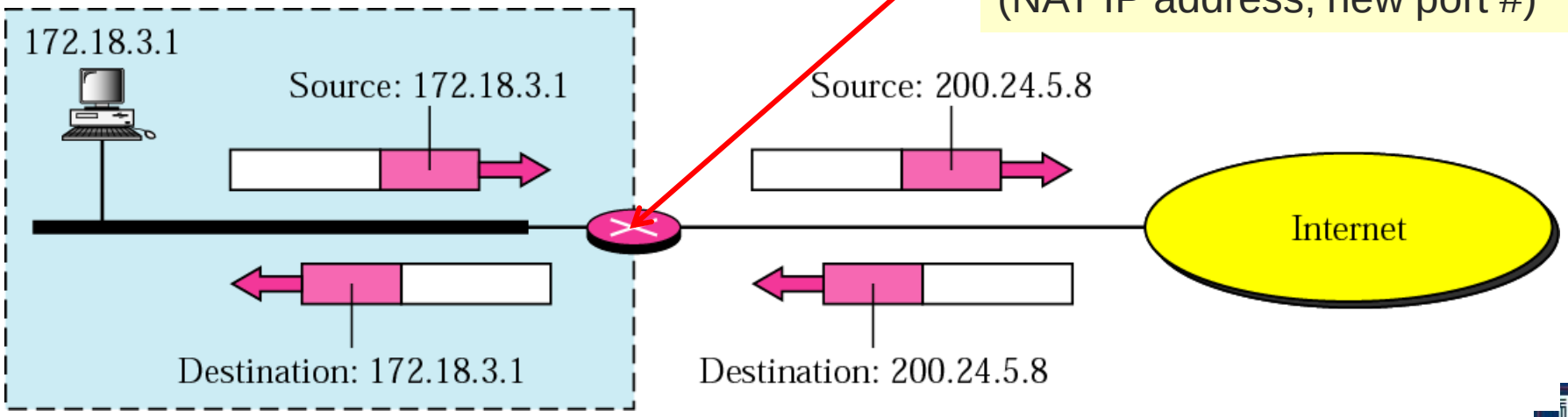
- Approach
  - Assign one router a global IP address
  - Assign internal hosts local IP addresses
- Change IP Headers
  - IP addresses (and possibly port numbers) of IP datagrams are replaced at the boundary of a private network
  - Enables hosts on private networks to communicate with hosts on the Internet
  - Run on routers that connect private networks to the public Internet

NAT: Network Address Translation

What address do the remote hosts respond to?

- Outgoing packet
  - Source IP address (private IP) replaced by global IP address maintained by NAT router
- Incoming packet
  - Destination IP address (global IP of NAT router) replaced by appropriate private IP address

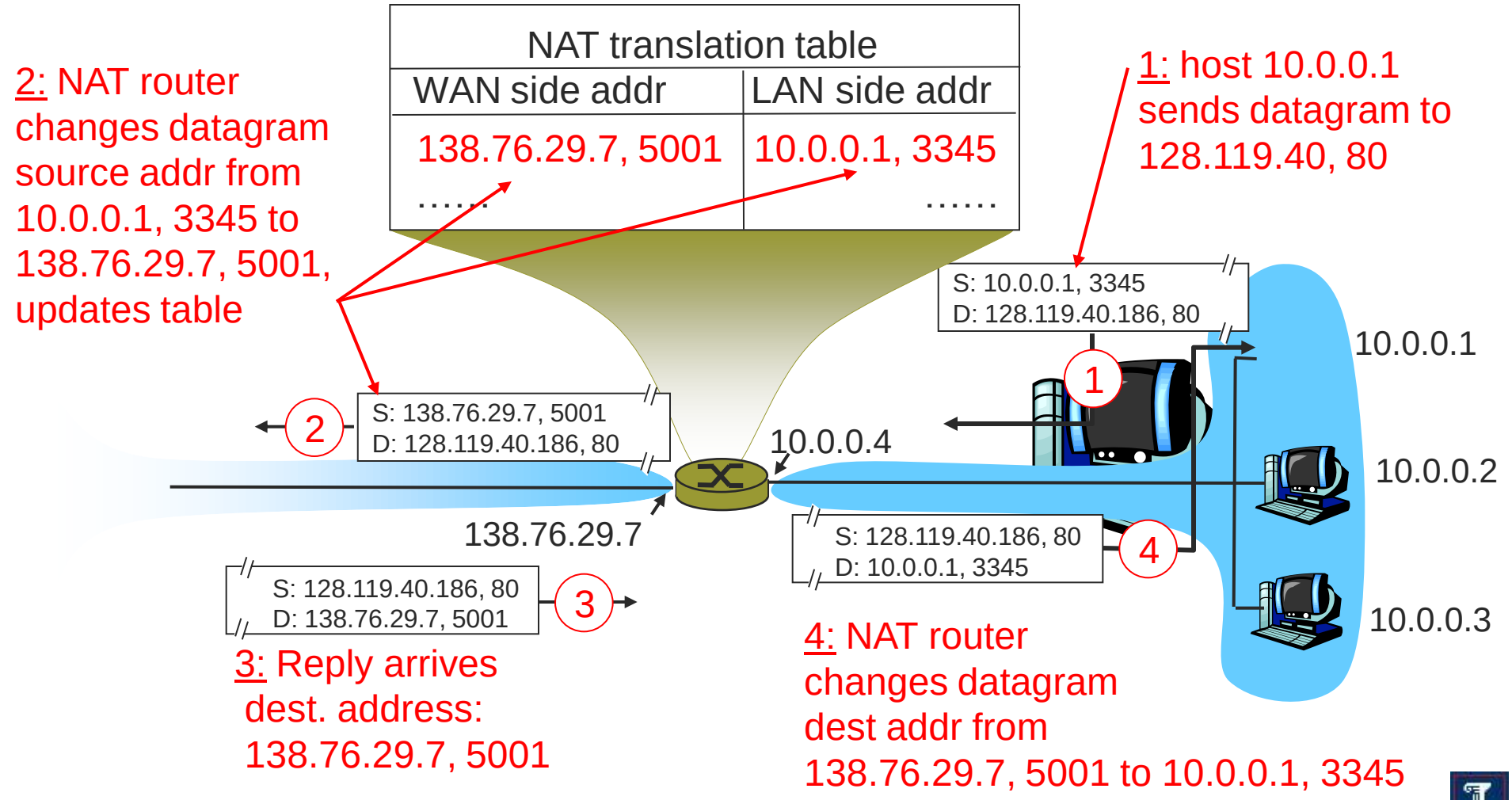
NAT router caches translation table:
(source IP address, port #) ➔ (NAT IP address, new port #)



NAT: Network Address Translation

- Benefits: local network uses just one (or a few) IP address as far as outside word is concerned
  - No need to be allocated range of addresses from ISP
    - Just one IP address is used for all devices
  - Can change addresses of devices in local network without notifying outside world
  - Can change ISP without changing addresses of devices in local network
  - Devices inside local net not explicitly addressable, visible by outside world (a security plus)

NAT: Network Address Translation



NAT: Network Address Translation

- Address Pooling
  - Corporate network has many hosts
  - Only a small number of public IP addresses
- NAT solution
  - Manage corporate network with a private address space
  - NAT, at boundary between corporate network and public Internet, manages a pool of public IP addresses
  - When a host from corporate network sends an IP datagram to a host in public Internet, NAT picks a public IP address from the address pool, and binds this address to the private address of the host

NAT: Network Address Translation

- Load balancing
  - Balance the load on a set of identical servers, which are accessible from a single IP address
- NAT solution
  - Servers are assigned private addresses
  - NAT acts as a proxy for requests to the server from the public network
  - NAT changes the destination IP address of arriving packets to one of the private addresses for a server
  - Balances load on the servers by assigning addresses in a round-robin fashion

NAT: Consequences

- 16-bit port-number field
  - 60,000 simultaneous connections with a single LAN-side address!
- End-to-end connectivity
  - NAT destroys universal end-to-end reachability of hosts on the Internet
  - A host in the public Internet often cannot initiate communication to a host in a private network
  - The problem is worse, when two hosts that are in different private networks need to communicate with each other

NAT: Consequences

- Performance
  - Modifying the IP header by changing the IP address requires that NAT boxes recalculate the IP header checksum
  - Modifying port number requires that NAT boxes recalculate TCP checksum
- Fragmentation
  - Datagrams fragmented before NAT device must not be assigned different IP addresses or different port numbers

NAT: Consequences

- IP address in application data
  - Applications often carry IP addresses in the payload of the application data
  - No longer work across a private-public network boundary
  - Hack: Some NAT devices inspect the payload of widely used application layer protocols and, if an IP address is detected in the application-layer header or the application payload, translate the address according to the address translation table