Chapter 1

Application Layer

Outline

Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookies
- Proxy
- FTP
- email

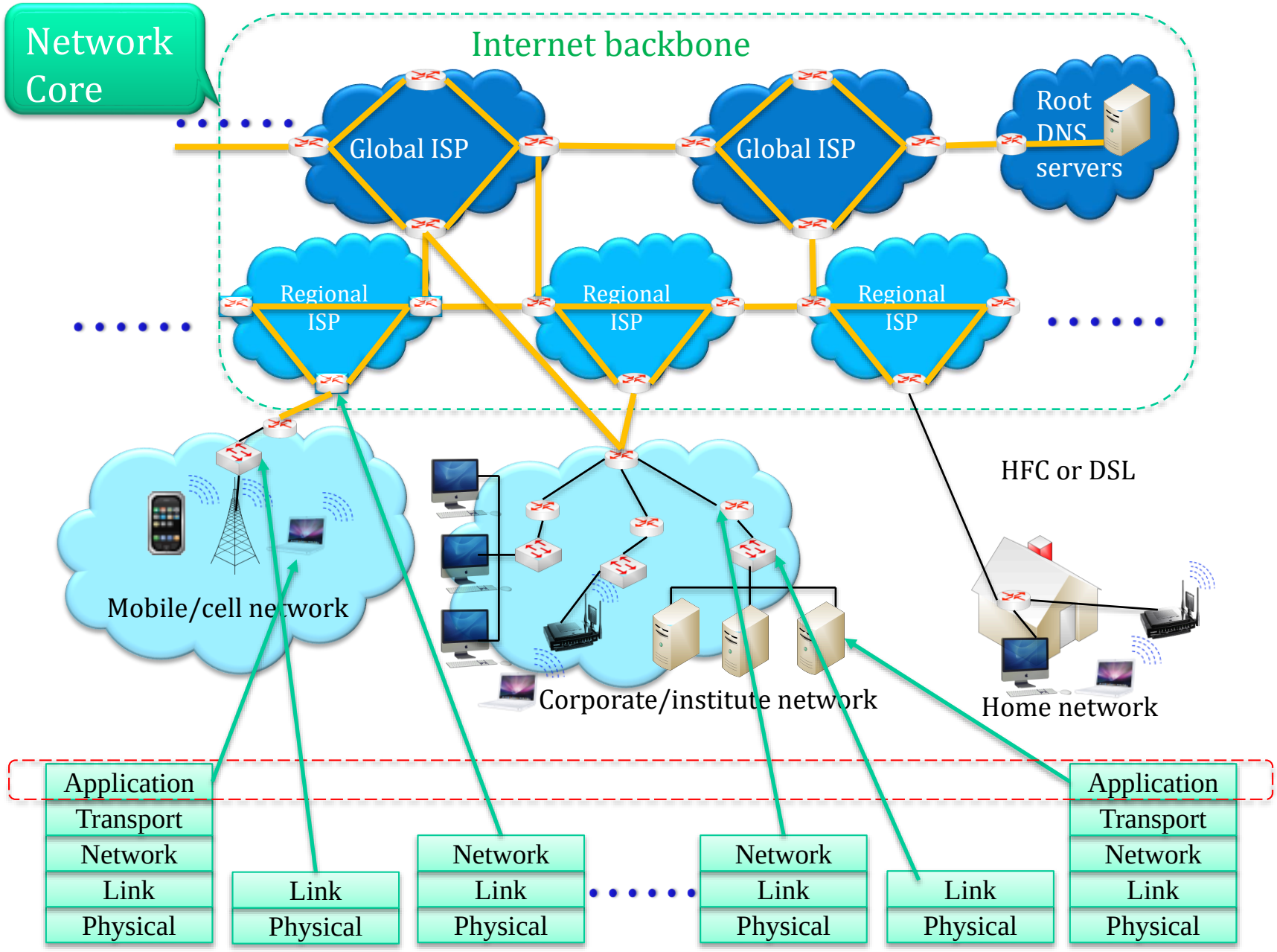
Popular network applications

- Web: http/https
- e-mail
- Cloud computing
- Social networking: Facebook, Twitter
- File transfer: FTP, SCP, SFTP
- Remote login: SSH
- Voice over IP: SIP, Skype
- Instant messaging
- P2P file sharing
- Streaming video: Youtube
- Real-time video conferencing
- Multi-user network games

Type	Application
Web and e-commerce	HTTP/HTTPS
Messaging	e-mail, voice over IP: Skype, video conferencing
Social networking	Facebook, Twitter
Multimedia	Streaming video: Youtube, video surveillance
Gaming	Multi-user network games
Utilities	Google Search, FTP, SSH, SFTP, P2P file sharing
Virtualization	Cloud computing for applications, security and storage

Network applications and architectures

- Application layer programs that
  - Run on end hosts
  - Communicate over network
  - E.g., Firefox browser communicates with Apache web server
- No application software is written for the network core
  - Network core does not run user applications
  - Core is implemented in firmware or hardware
- Applications on end hosts foster rapid application development
- Application architectures
  - Client/server
  - Peer-to-peer (P2P)
  - Hybrid of client/server and P2P


Network Core
Internet backbone
Global ISP
Global ISP
Root DNS servers
Regional ISP
Regional ISP
Regional ISP
Mobile/cell network
Corporate/institute network
HFC or DSL
Home network
Application
Transport
Network
Link
Physical
Link
Physical
Network
Link
Physical
Network
Link
Physical
Link
Physical
Application
Transport
Network
Link
Physical

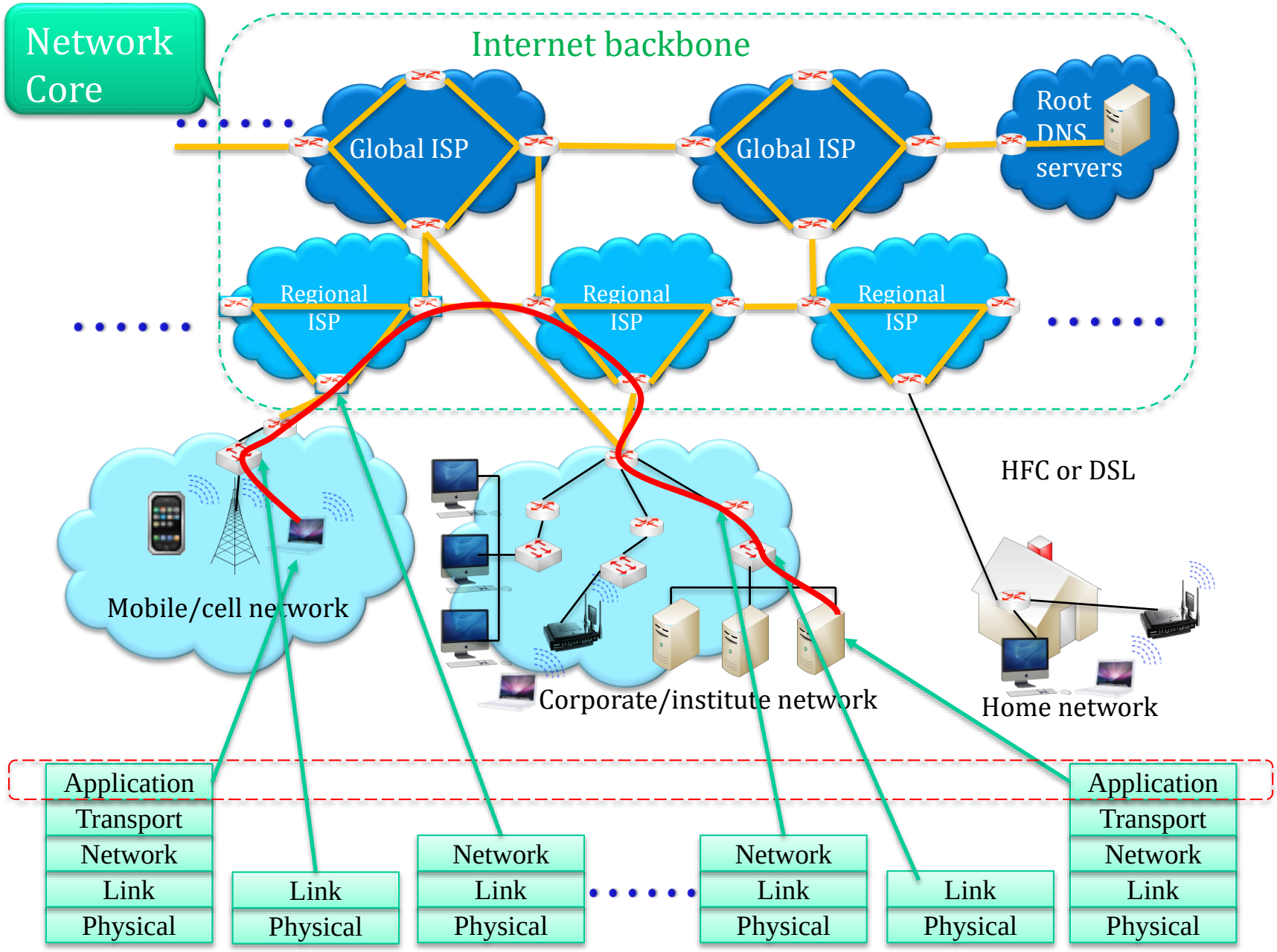
Outline

Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookie
- Proxy
- FTP
- email

Client/server architecture

- Server:
  - Always-on host
  - Fixed IP address so DNS entry is fixed
  - Server farms for performance
- Clients:
  - Communicate with server
  - Intermittently connected
  - Dynamic IP addresses
  - Do not communicate directly with other clients


Network Core
Internet backbone
Global ISP
Global ISP
Root DNS servers
Regional ISP
Regional ISP
Regional ISP
Mobile/cell network
Corporate/institute network
HFC or DSL
Home network
Application
Transport
Network
Link
Physical
Link
Physical
Network
Link
Physical
Network
Link
Physical
Link
Physical
Application
Transport
Network
Link
Physical

P2P architecture

- A computer serves as both server and client
  - No server bottleneck in P2P
  - Highly scalable but difficult to manage
  - End hosts directly communicate with each other
  - Peers are intermittently connected and have dynamic IP addresses
- Example: in the order of popularity
  - bitTorrent
    - 35% of Internet traffic by CacheLogic, ref: http://en.wikipedia.org/
    - Client software: Limewire, µtorrent, etc.
  - eDonkey
    - eDonkey architecture is rather common in Europe
  - Fast Track
    - Client software: Kazaa, iMesh, and Grokster
  - Gnutella
    - Client software: Limewire, Morpheus
- Widely used for illegal file sharing

Processes communication

- Process: a program running within a host
  - Inter-process communication (IPC): two processes communicate using IPC supported by OS within one host

Network applications: Processes in two hosts communicate by exchanging messages supported by both OS through the network

Client process: a process that initiates communication

Server process: a process that waits to be contacted

Application feels the information is virtually in local memory (with latency, jitter, and loss/error)

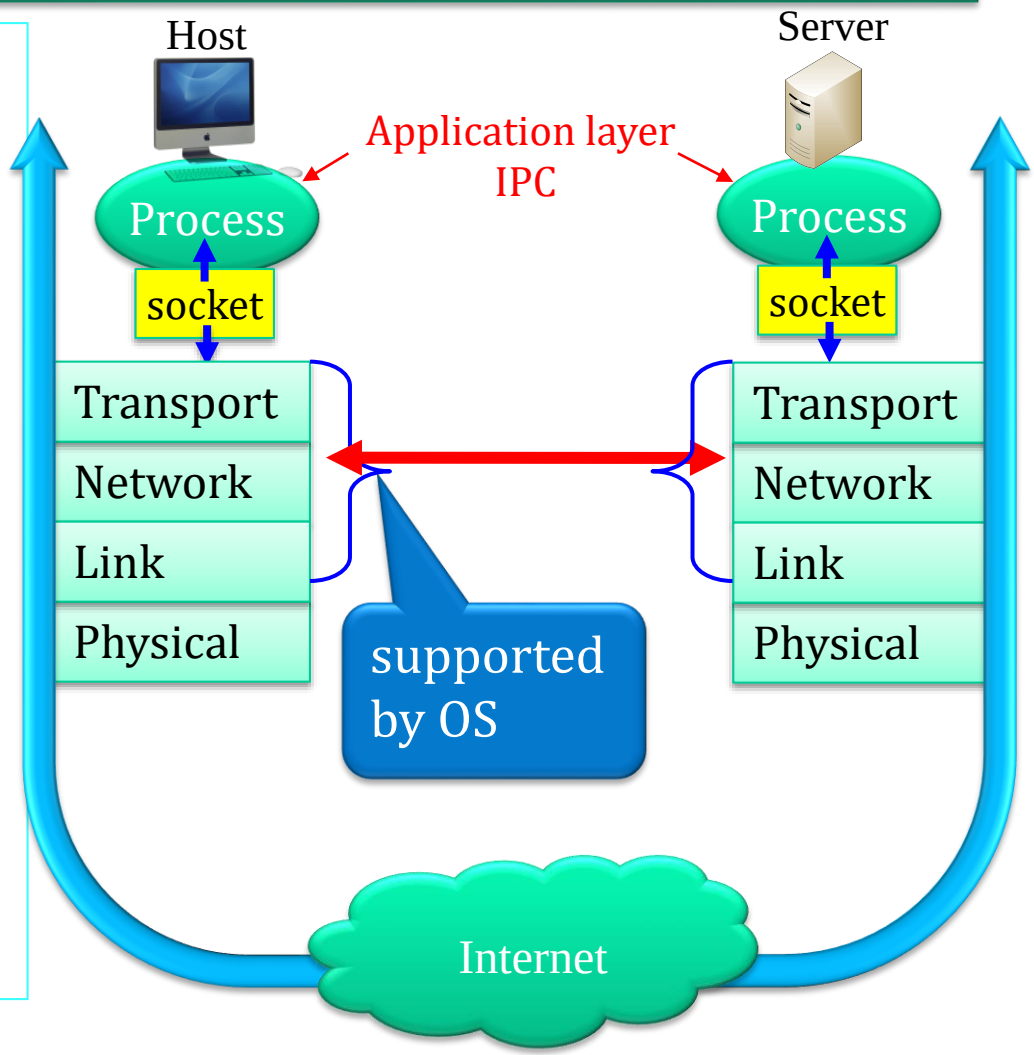
Outline

Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookie
- Proxy
- FTP
- email

Sockets

- A process sends/receives messages to/from its socket
- A socket
  - Sending process delivers a message as if writing to local memory or a device
  - Sending process relies on transport layer and layers below to deliver a message to the socket at receiving process
- API in a socket: programming
  - Select a transport protocol, such as TCP or UDP
  - Ability to specify parameters, such as IP address and port number



Addressing a process

- To receive a message, an IPC must have identifier
- A host device has a unique 32-bit IPv4 address
- A number of services on same host
  - A server can serve HTTP and email simultaneously
- Identifier includes both IP address and port numbers associated with a process on a host
- Example port numbers:
  - HTTP server: 80
  - DNS server: 53
  - Mail server: 25
- To send HTTP message to cnn.com server:
  - IP address: 64.236.29.120
  - Port number: 80

Socket Definition

- A TCP (Transmission Control Protocol) socket contain a 4 tuple:
  - Source IP address
  - Source port number
  - Destination IP address
  - Destination port number
- A UDP (User Datagram Protocol) socket contains a two tuple:
  - Server IP address
  - Server port number

Application layer protocol

- Types of messages exchanged
- E.g., request, and response
- Message syntax:
  - Fields in messages
  - How fields are interpreted
- Message semantics
  - Action for information in fields
- Rules
  - How a process sends and responds to messages

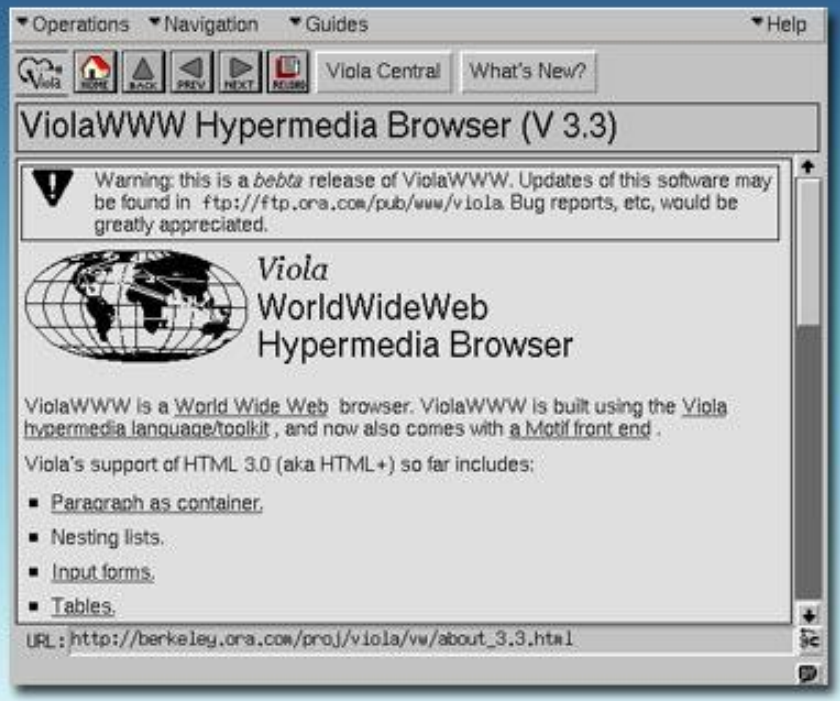
- Public-domain Internet protocols:
  - Defined in RFCs by IETF
    - Request for comments (rfc): rfc-editor.org for download files
  - Allows for interoperability
  - E.g., HTTP, SMTP
- Proprietary protocols:
  - E.g., Skype

The first web browser

Jim Rapoza Picks the Top Web Technologies of All Time

ViolaWWW

Though it was quickly eclipsed by the Mosaic Web browser that followed, ViolaWWW was the first breed of what would become the modern Web browser.



The first search engine

Jim Rapoza Picks the Top Web Technologies of All Time

WAIS

Basically, the first iteration of search on the Web, WAIS was adapted to add the ability to search for text on Web sites and paved the way for the popular search engines that followed.

Outline

Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookie
- Proxy
- FTP
- email

Transport service selections for applications

- Data loss/error
  - Some applications (e.g., audio, video) can tolerate some loss/error
  - Some applications (e.g., file transfer, telnet) require 100% reliable data transfer
- Minimum bandwidth
  - Some applications require a minimum amount of bandwidth
    - Video streaming
    - Internet telephony/VoIP
  - Some applications are flexible
    - Email
    - Web surfing

- Timing
  - Some applications require low delay jitter
    - Video streaming
    - Internet telephony / VoIP
    - Interactive games

Transport service requirements for Internet apps

Applications	Error/loss tolerance	Min. Bandwidth required	Delay Jitter tolerance
Web/file transfer	No	No	Yes
Email	No	No	Yes
Video/audio streaming	Yes	Yes	No

Internet transport protocols services

- TCP service:
  - Connection-oriented: setup required between client and server processes
  - Reliable transport between sending and receiving process
  - Flow control: sender does not overwhelm receiver's buffer/link
  - Congestion control: sender throttles back when network is overloaded
  - TCP cannot assure:
    - Timing guarantee
    - Delay jitter guarantee
    - Minimum bandwidth

- UDP service:
  - Connectionless
    - Directly sends a datagram without establishing a connection
    - Low latency and low overhead
    - Unreliable/best effort data transfer between sending and receiving process
  - UDP cannot provide:
    - Flow control
    - Congestion control
    - Timing guarantee
    - Delay jitter guarantee
    - Minimum bandwidth

Internet apps, application layer and transport layer protocols

Application	Application layer protocol	Transport layer protocol
Web	http/https	TCP
Email	SMTP/IMAP/POP3/https	TCP
File transfer	FTP/SFTP	TCP
Multimedia	RTSP/RTP/RTCP	TCP and UDP
VoIP	SIP/H.323/RTP/RTCP	TCP/UDP

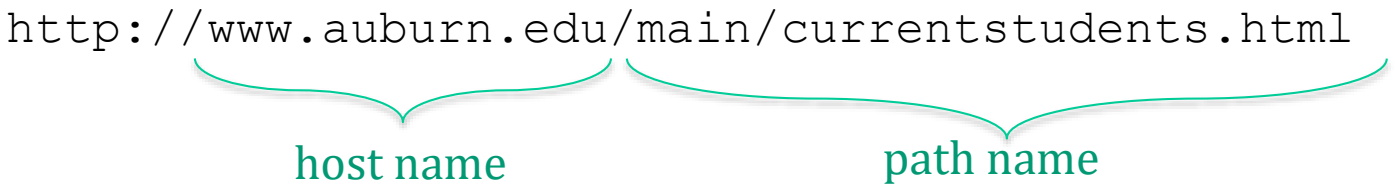
Outline

Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookie
- Proxy
- FTP
- email

Web and HTTP

- Web page consists of objects
- Object can be base HTML file, JPEG image, JavaScript, etc.
- Base HTML-file includes several referenced objects, such as images
- Each object is addressable by a URL
- Example URL:



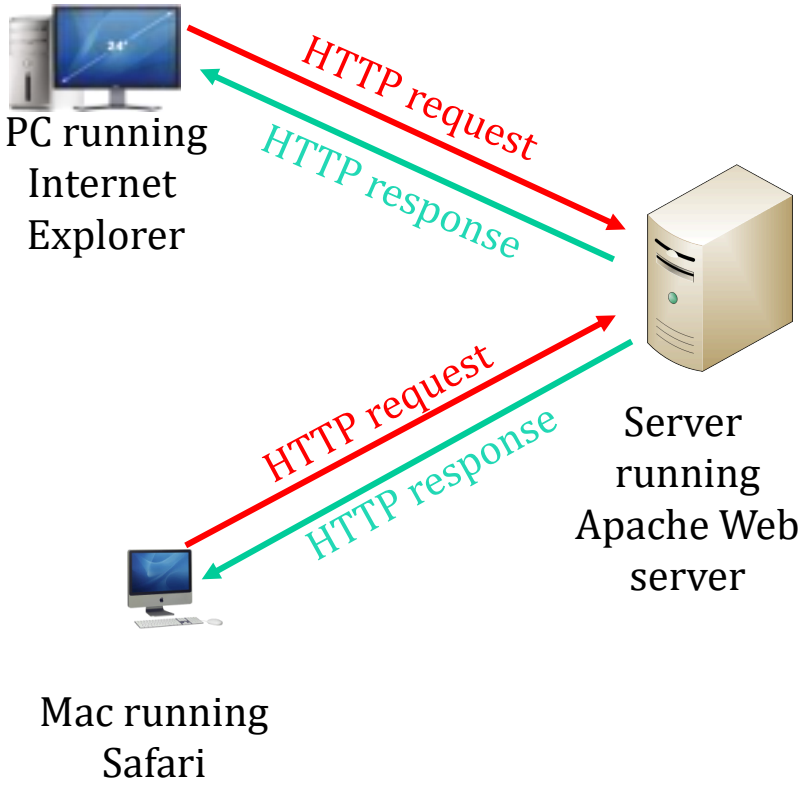
HTTP overview (1)

`http://www.auburn.edu/main/currentstudents.html`

host name

path name

- HTTP: hypertext transfer protocol
  - Web's application layer protocol
  - Client/server model
    - Client: a browser that requests, receives, and illustrates Web objects
    - Server: a Web server that sends objects in response to requests
  - HTTP 1.0: RFC 1945
  - HTTP 1.1: RFC 2068



HTTP overview (2)

- Based on TCP
  - Client initiates a TCP connection for creating a socket that connects to a Web server using server port 80
  - Server accepts TCP connection from client
  - HTTP messages are exchanged between browser (HTTP client) and Web server (HTTP server)
  - TCP connection closed

- HTTP is a stateless protocol
  - Server maintains no information about past client requests
  - Protocols that maintain state information are complex
    - Past history (state) must be maintained
    - E.g., FTP, SFTP are not stateless protocol
    - Must maintain the residing directory information

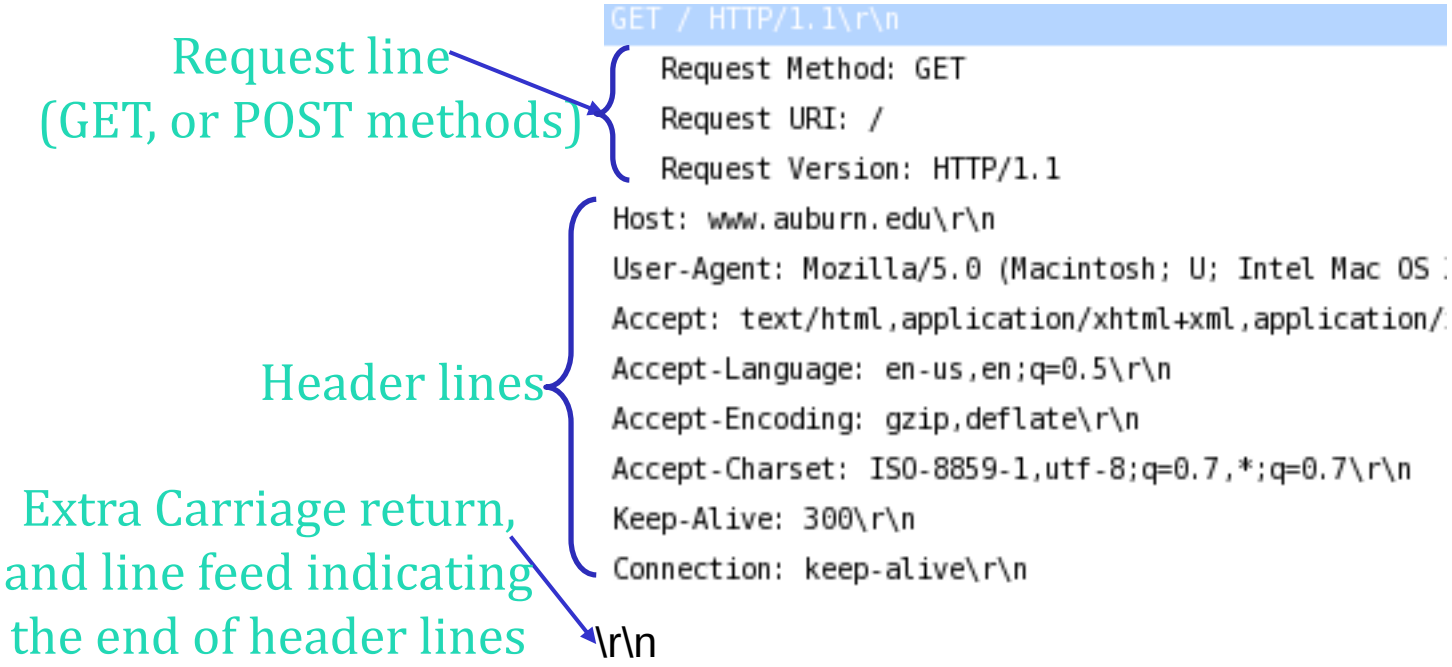
Methods available in HTTP

- HTTP/1.0
  - GET
  - POST
  - HEAD
    - For debugging
    - Permits server to leave requested objects out of response in order to save debugging time

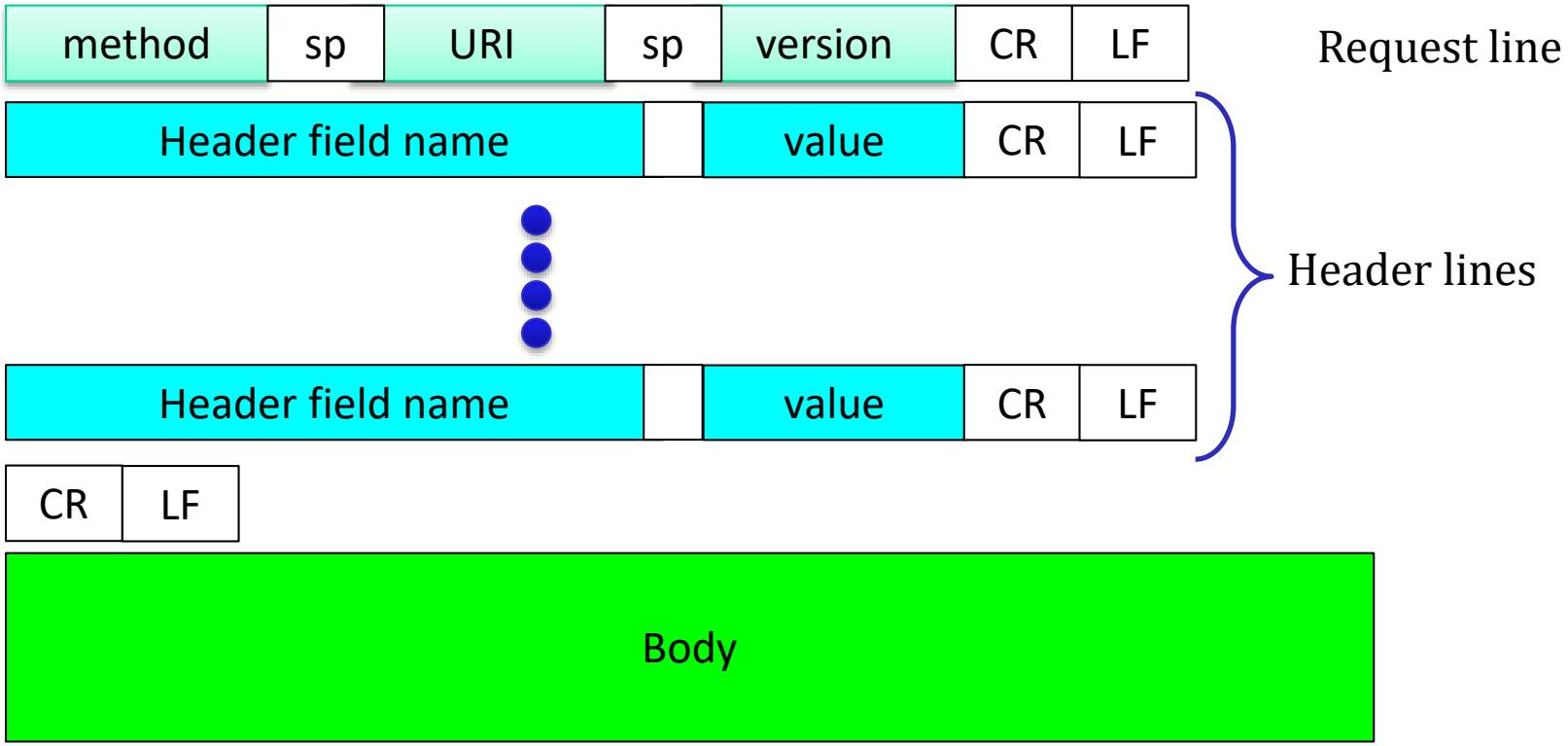
- HTTP/1.1
  - GET, POST, HEAD
  - PUT
    - Uploads file in entity body to path specified in the URL field
  - DELETE
    - Deletes file specified in the URL field

HTTP request message

- RFC 2616
  - Two types of HTTP messages: request, and response
  - HTTP request message:
    - ASCII format



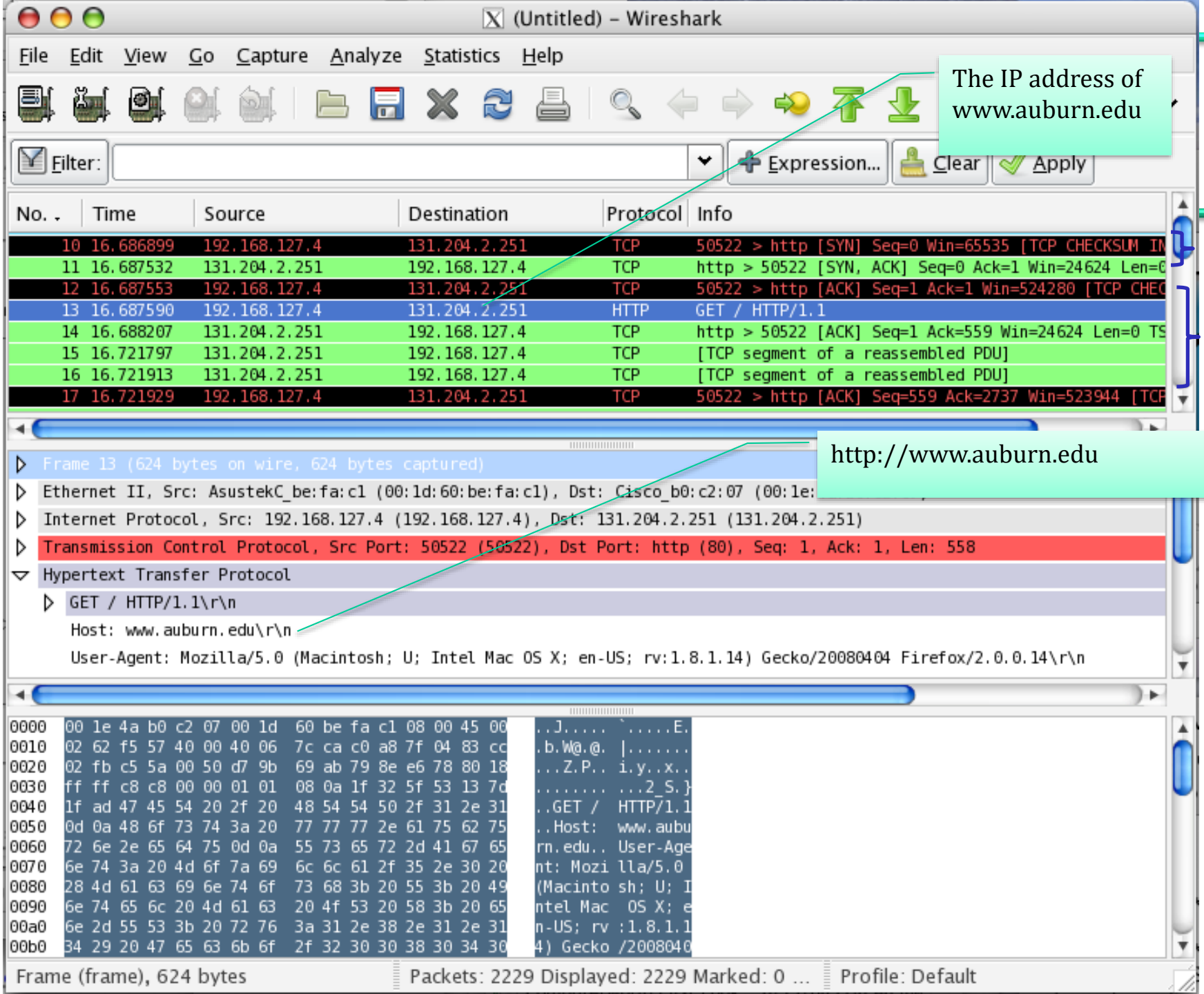
HTTP request message: general format



sp: space
CR: carriage return
LF: line feed

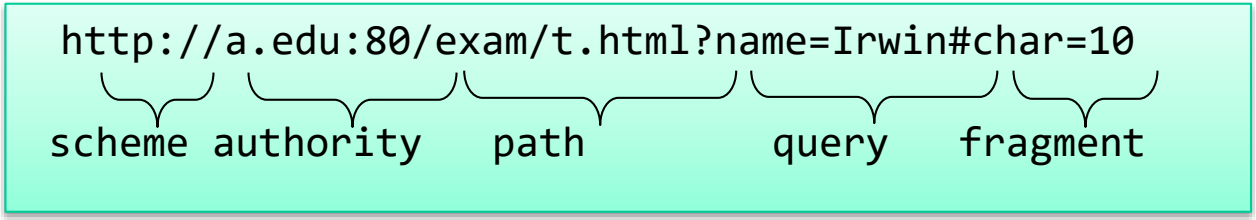
Wireshark

- Network protocol analyzer or sniffer
- www.wireshark.org
- Free download
- Available for any OS
- Puts Ethernet in promiscuous mode

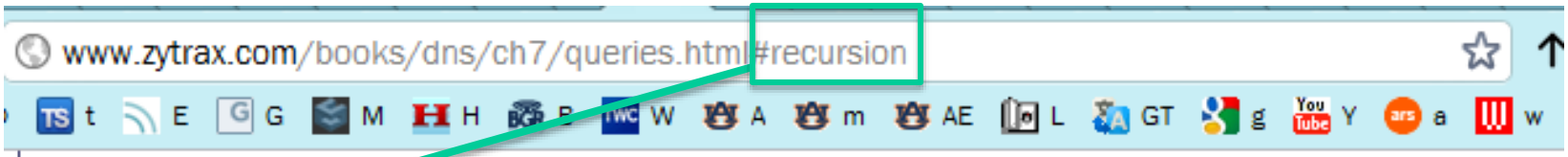

(Untitled) – Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply
The IP address of www.auburn.edu
No. Time Source Destination Protocol Info
10 16.686899 192.168.127.4 131.204.2.251 TCP 50522 > http [SYN] Seq=0 Win=65535 [TCP CHECKSUM IN
11 16.687532 131.204.2.251 192.168.127.4 TCP http > 50522 [SYN, ACK] Seq=0 Ack=1 Win=24624 Len=0
12 16.687553 192.168.127.4 131.204.2.251 TCP 50522 > http [ACK] Seq=1 Ack=1 Win=524280 [TCP CHEC
13 16.687590 192.168.127.4 131.204.2.251 HTTP GET / HTTP/1.1
14 16.688207 131.204.2.251 192.168.127.4 TCP http > 50522 [ACK] Seq=1 Ack=559 Win=24624 Len=0 TS
15 16.721797 131.204.2.251 192.168.127.4 TCP [TCP segment of a reassembled PDU]
16 16.721913 131.204.2.251 192.168.127.4 TCP [TCP segment of a reassembled PDU]
17 16.721929 192.168.127.4 131.204.2.251 TCP 50522 > http [ACK] Seq=559 Ack=2737 Win=523944 [TCP
1st RTT
2nd RTT
Frame 13 (624 bytes on wire, 624 bytes captured)
Ethernet II, Src: AsustekC_be:fa:c1 (00:1d:60:be:fa:c1), Dst: Cisco_b0:c2:07 (00:1e:
Internet Protocol, Src: 192.168.127.4 (192.168.127.4), Dst: 131.204.2.251 (131.204.2.251)
Transmission Control Protocol, Src Port: 50522 (50522), Dst Port: http (80), Seq: 1, Ack: 1, Len: 558
Hypertext Transfer Protocol
GET / HTTP/1.1\r\n
Host: www.auburn.edu\r\n
User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X; en-US; rv:1.8.1.14) Gecko/20080404 Firefox/2.0.0.14\r\n
http://www.auburn.edu
0000 00 1e 4a b0 c2 07 00 1d 60 be fa c1 08 00 45 00 ..J..... `.....E.
0010 02 62 f5 57 40 00 40 06 7c ca c0 a8 7f 04 83 cc .b.W@.@. |.....
0020 02 fb c5 5a 00 50 d7 9b 69 ab 79 8e e6 78 80 18 ...Z.P.. i.y..x..
0030 ff ff c8 c8 00 00 01 01 08 0a 1f 32 5f 53 13 7d ....2_S.}
0040 1f ad 47 45 54 20 2f 20 48 54 54 50 2f 31 2e 31 ..GET / HTTP/1.1
0050 0d 0a 48 6f 73 74 3a 20 77 77 77 2e 61 75 62 75 ..Host: www.aubu
0060 72 6e 2e 65 64 75 0d 0a 55 73 65 72 2d 41 67 65 rn.edu.. User-Age
0070 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 nt: Mozi lla/5.0
0080 28 4d 61 63 69 6e 74 6f 73 68 3b 20 55 3b 20 49 (Macinto sh; U; I
0090 6e 74 65 6c 20 4d 61 63 20 4f 53 20 58 3b 20 65 ntel Mac OS X; e
00a0 6e 2d 55 53 3b 20 72 76 3a 31 2e 38 2e 31 2e 31 n-US; rv :1.8.1.1
00b0 34 29 20 47 65 63 6b 6f 2f 32 30 30 38 30 34 30 4) Gecko /2008040
Frame (frame), 624 bytes
Packets: 2229 Displayed: 2229 Marked: 0 ...
Profile: Default

URI

- Uniform Resource Identifier (URI) definition
  - <scheme>://<authority><path>?<query>#fragment



    - RFC 2396: Uniform resource identifiers (URI): Generic syntax
    - Each component, except <scheme>, may be absent from a particular URI
    - E.g. mailto:xyzt@auburn.edu uses mailto scheme for electronic mail addresses
  - Uniform Resource Locator (URL) is a type of Uniform Resource Identifier (URI)
    - URL: specifies where an identified resource is available and the mechanism for retrieving it
- HTTP URL takes the form:
  - http://<host>:<port>/<path>?<search part>#fragment
    - A percent-encoded octet is encoded as a character triplet, consisting of the percent character "%" followed by the two hexadecimal digits representing that octet's numeric value
      - For example, "%20" is in US-ASCII corresponds to the space character (SP)



recursion

```
recursion yes | no;
```

If **recursion** is set to 'yes' (the default) the server will always provide recursive query behaviour if requested by the client (resolver). If set to 'no' the server will only provide iterative query behaviour - normally resulting in a referral. If the answer to the query already exists in the cache it will be returned irrespective of the value of this statement. This statement essentially controls caching behaviour in the server. The allow-recursion statement and the view clauses can provide fine-grained control. This statement may be used in a view or a global options clause.

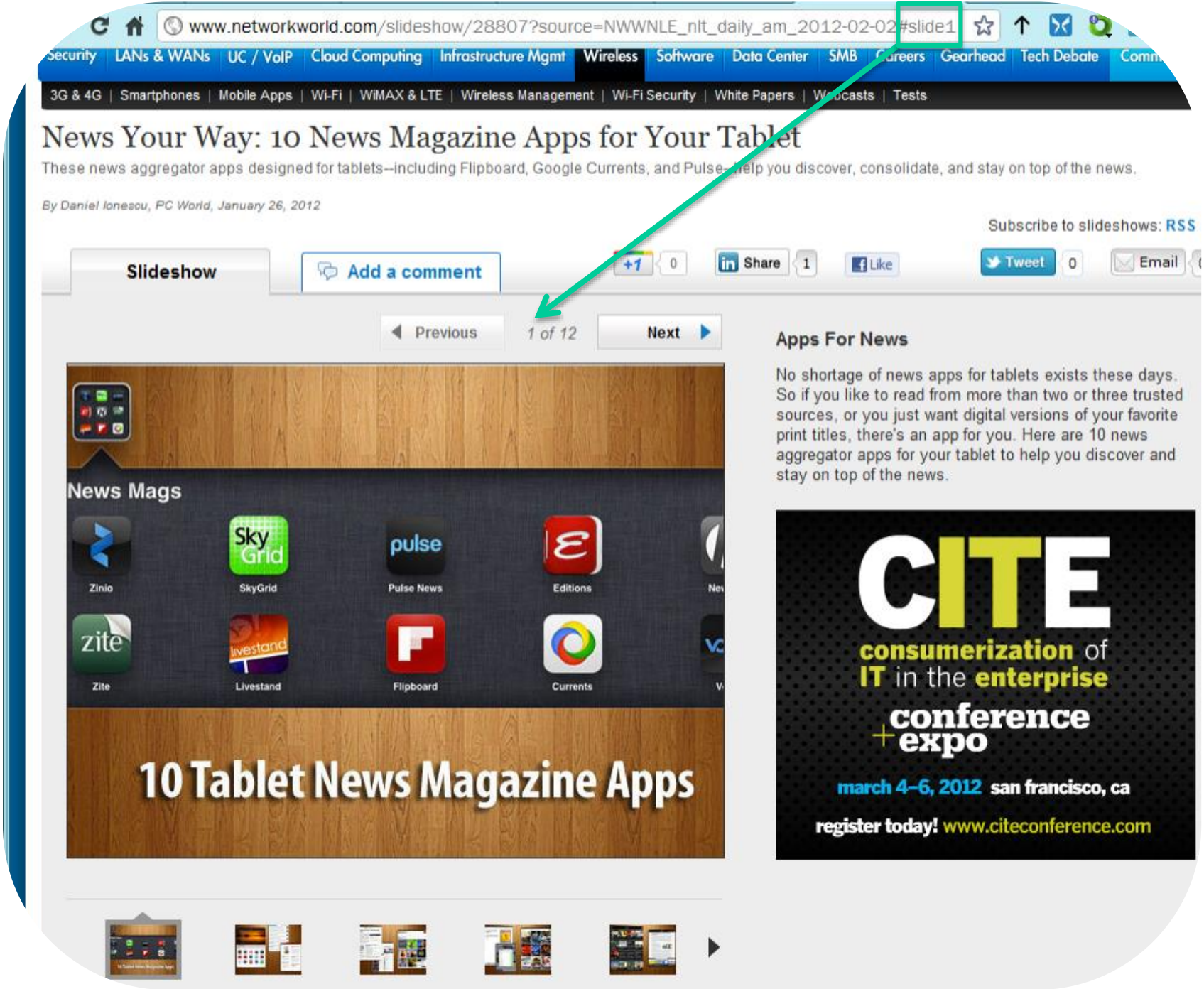
recursive-clients

```
recursive-clients number;
recursive-clients 25;
```

Defines the number of simultaneous recursive lookups the server will perform on behalf of its clients. BIND 9 default is 1000, that is, it will support 1000 simultaneous recursive lookup requests - which should be enough! This statement may only be used in a global options clause.

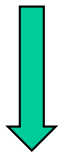
The #fragment indicates the element with id in a web page

- The browser in the Figure sends in the URL and receives the complete page of queries.html file
- Then the browser processes the whole page, follows the URL with a fragment “#recursion” which indicates the element with id="recursion" and displays the element id recursion as the first line


www.networkworld.com/slideshow/28807?source=NWWNLE_nlt_daily_am_2012-02-02#slide1
Security | LANs & WANs | UC / VoIP | Cloud Computing | Infrastructure Mgmt | Wireless | Software | Data Center | SMB | Careers | Gearhead | Tech Debate
3G & 4G | Smartphones | Mobile Apps | Wi-Fi | WiMAX & LTE | Wireless Management | Wi-Fi Security | White Papers | Webcasts | Tests
News Your Way: 10 News Magazine Apps for Your Tablet
These news aggregator apps designed for tablets--including Flipboard, Google Currents, and Pulse--help you discover, consolidate, and stay on top of the news.
By Daniel Ionescu, PC World, January 26, 2012
Subscribe to slideshows: RSS
Slideshow
Add a comment
+1 0
Share 1
Like
Tweet 0
Email
Previous
1 of 12
Next
News Mags
Zinio
SkyGrid
Pulse News
Editions
Zite
Livestand
Flipboard
Currents
10 Tablet News Magazine Apps
Apps For News
No shortage of news apps for tablets exists these days. So if you like to read from more than two or three trusted sources, or you just want digital versions of your favorite print titles, there's an app for you. Here are 10 news aggregator apps for your tablet to help you discover and stay on top of the news.
CITE
consumerization of IT in the enterprise
conference + expo
march 4–6, 2012 san francisco, ca
register today! www.citeconference.com

Uploading form input

- GET method:
  - Input is uploaded in URL field of request line:



http://search.auburn.edu/query.html?col=au&qt=war+eagle+camp

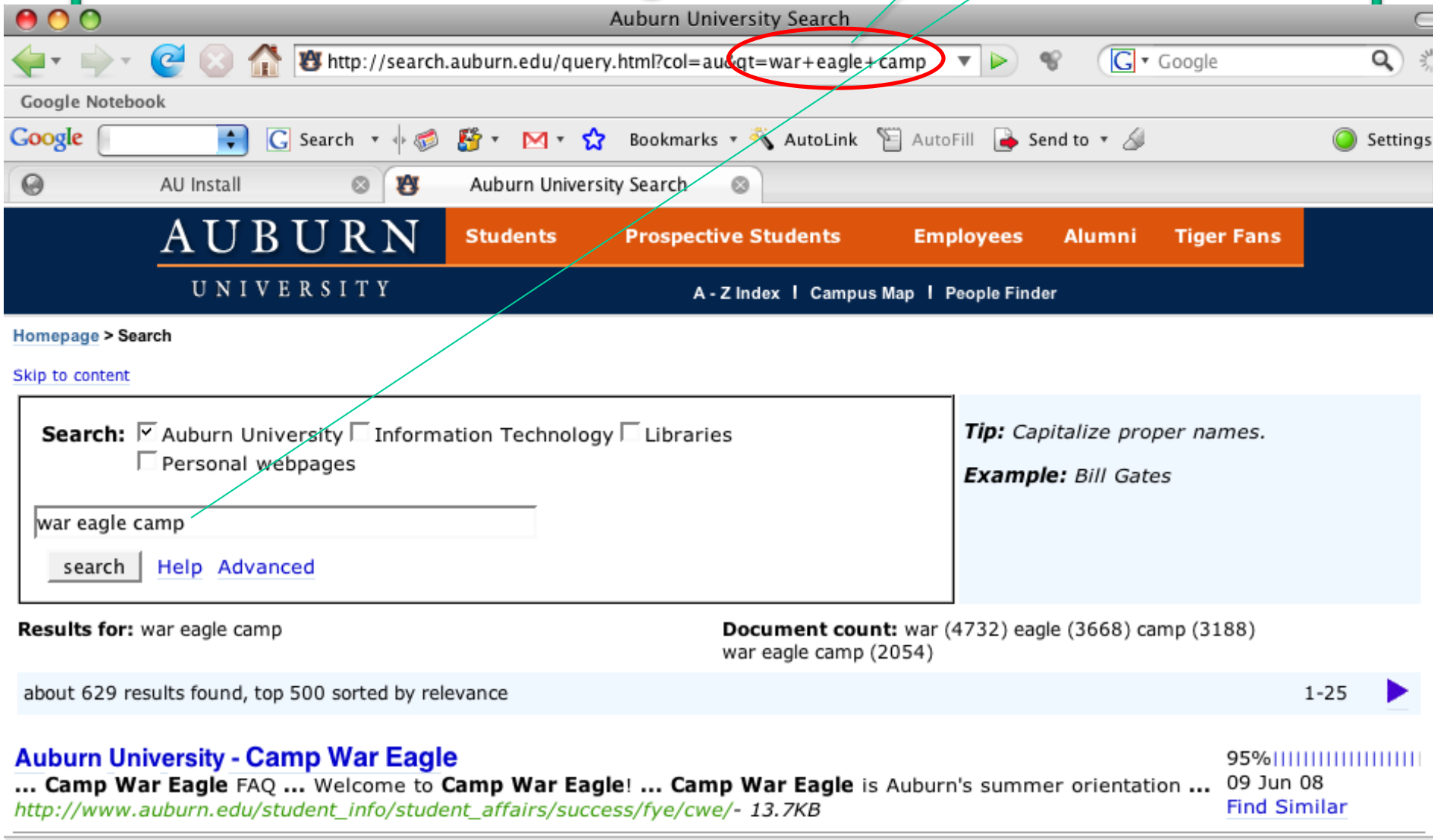
The browser sends out:

GET /query.html?col=au&qt=war+eagle+camp HTTP/1.1\r\n

- Post method:
  - Web page often includes a form input
    - Input is uploaded to server in body
    - The values in a form are collected with method="post"
    - Information sent from a form with the POST method is invisible in the URL
  - There are no limits on the amount of information sent from a browser
    - IE has a limit of 2083 characters in URL (GET method)

Get method: using header lines

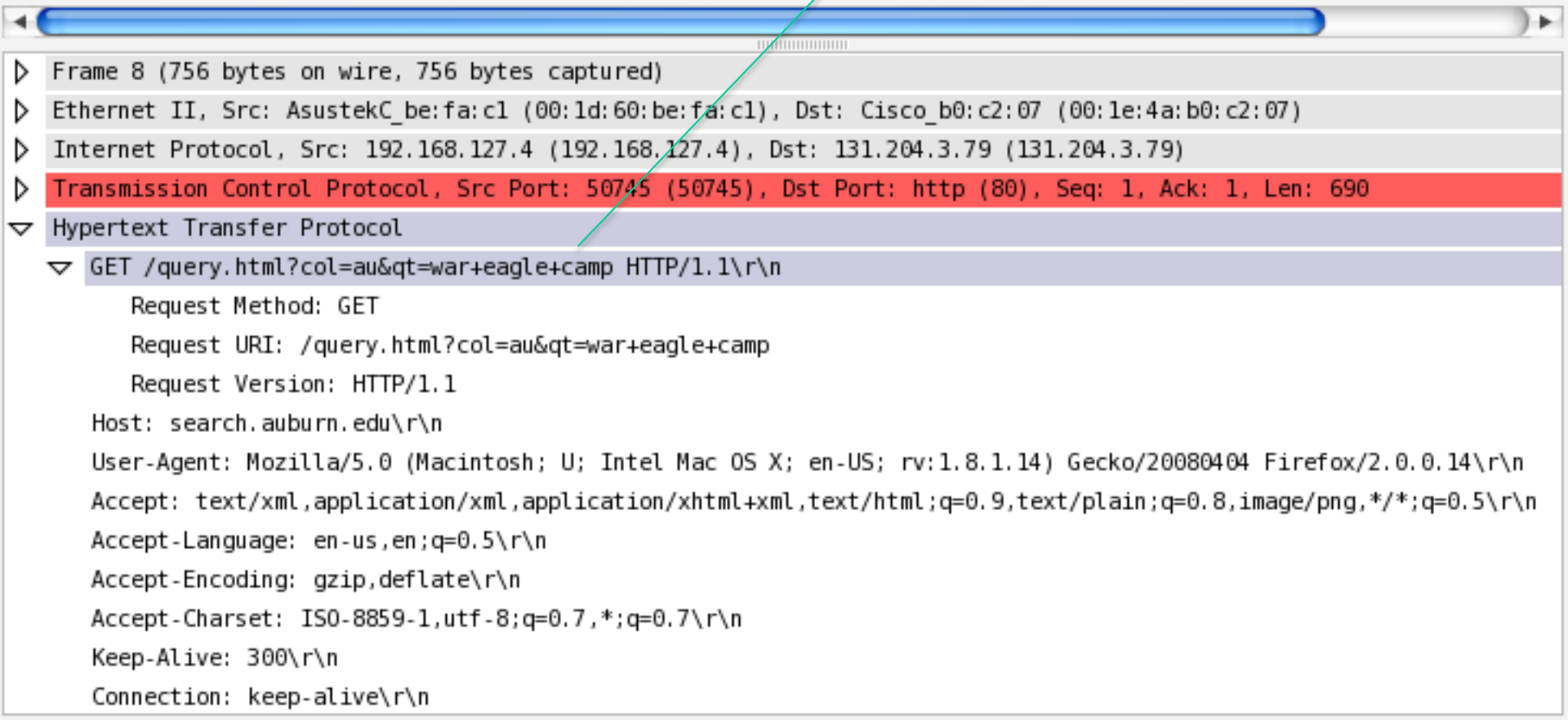
Get with query in clear text



Get method

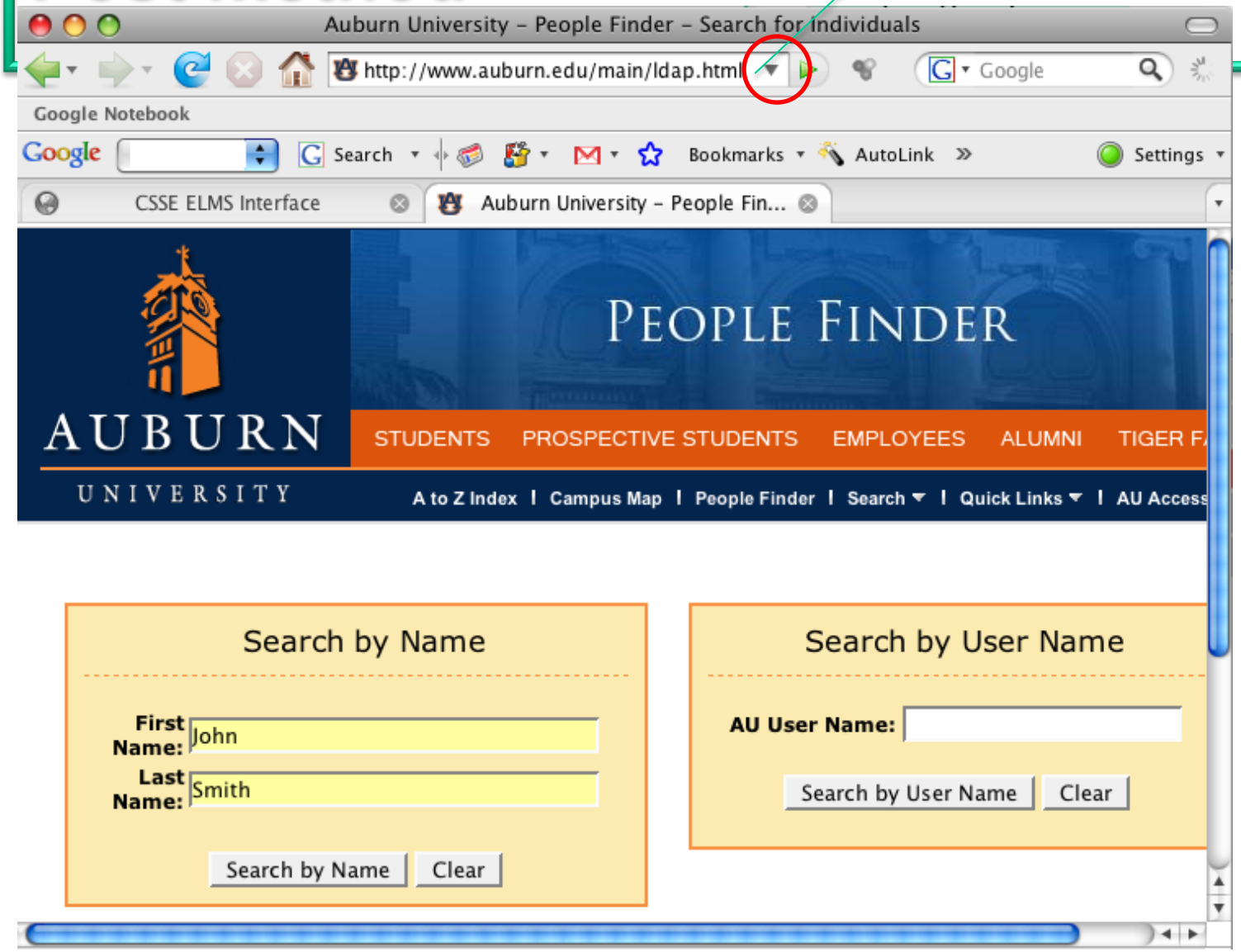
Get with query in clear text

No.	Time	Source	Destination	Protocol	Info
7	8.570677	192.168.127.4	131.204.3.79	TCP	50745 > http [ACK] Seq=1 Ack=1 Win=524280 [TC
8	8.570709	192.168.127.4	131.204.3.79	HTTP	GET /query.html?col=au&qt=war+eagle+camp HTTP
9	8.571545	131.204.3.79	192.168.127.4	TCP	http > 50745 [ACK] Seq=1 Ack=691 Win=24624 Le

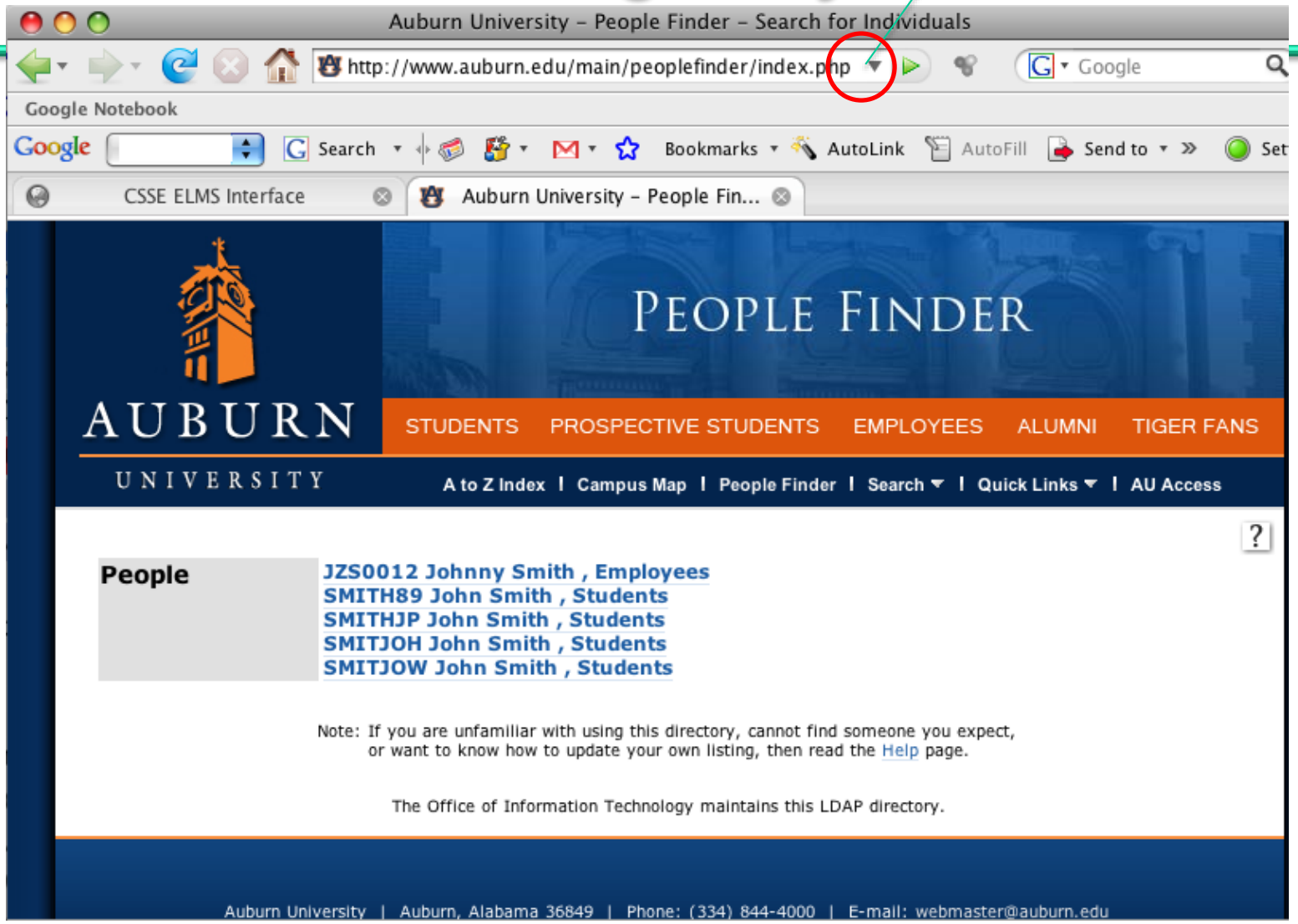


Post method

post without query in clear text

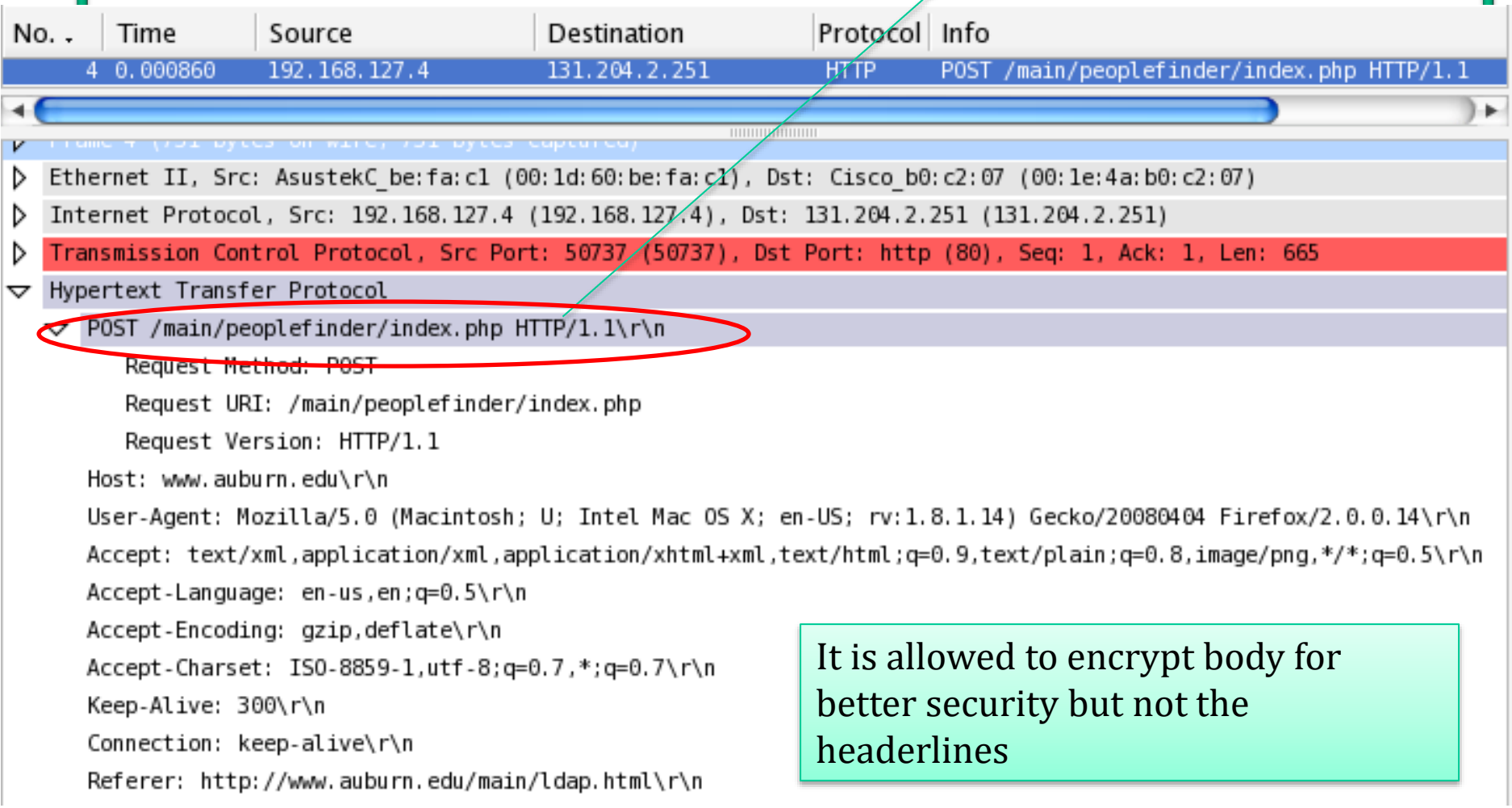


Post method: using body



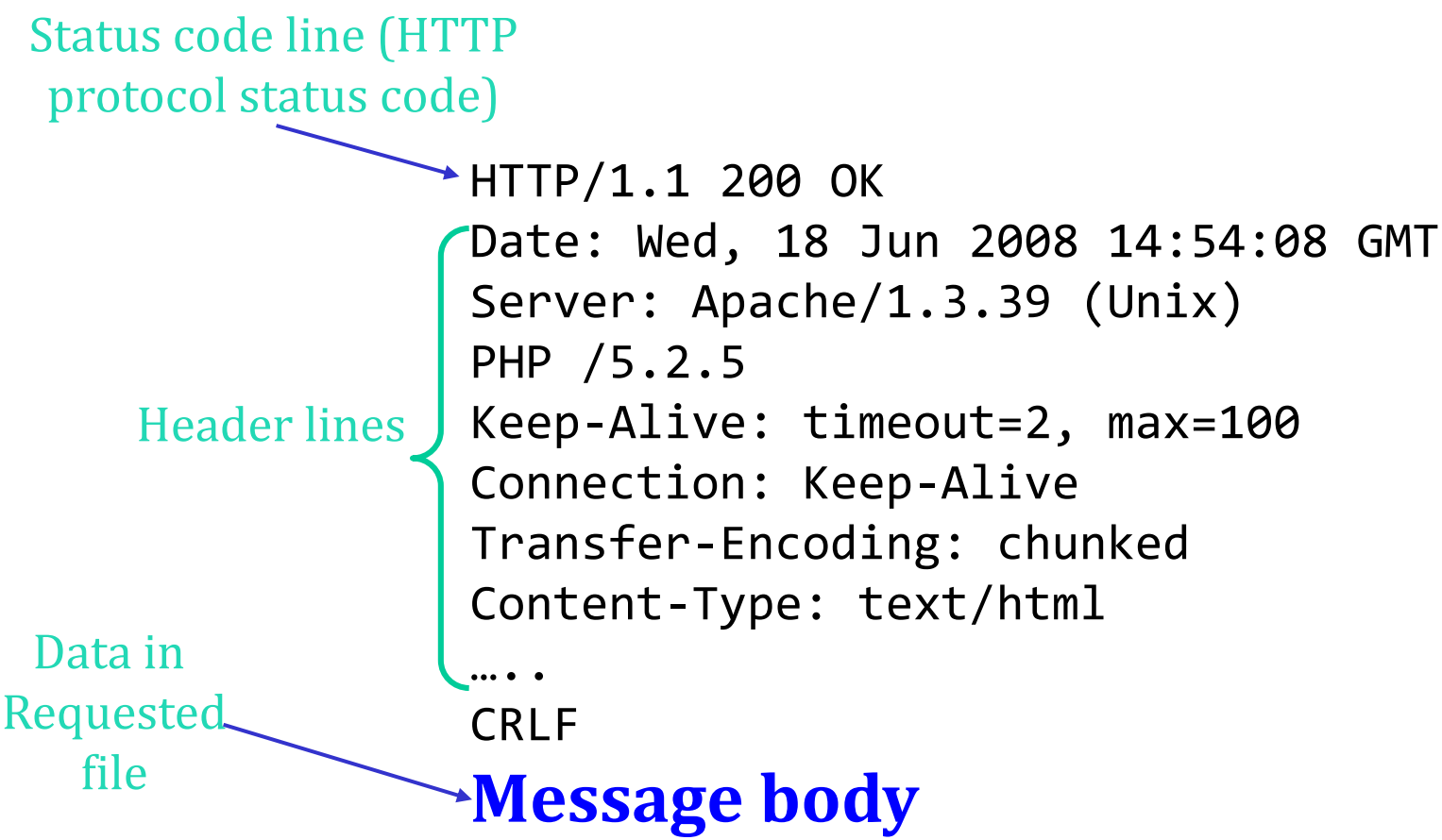
Post method: better security

post without query in clear text



It is allowed to encrypt body for better security but not the headerlines

HTTP response message



HTTP response status codes

- A status code is shown in the first line of a Web server to client response message
- Some frequently used status codes:

STATUS code	Description
200 OK	The request has succeeded and the requested object appears later in this message
301 Moved Permanently	The requested object has moved and its new location is specified later in this message
400 Bad Request	The requested message was not understood by the server
404 Not Found	The requested document was not found on this server
505 HTTP Version not supported	The web server does not support the version of the request

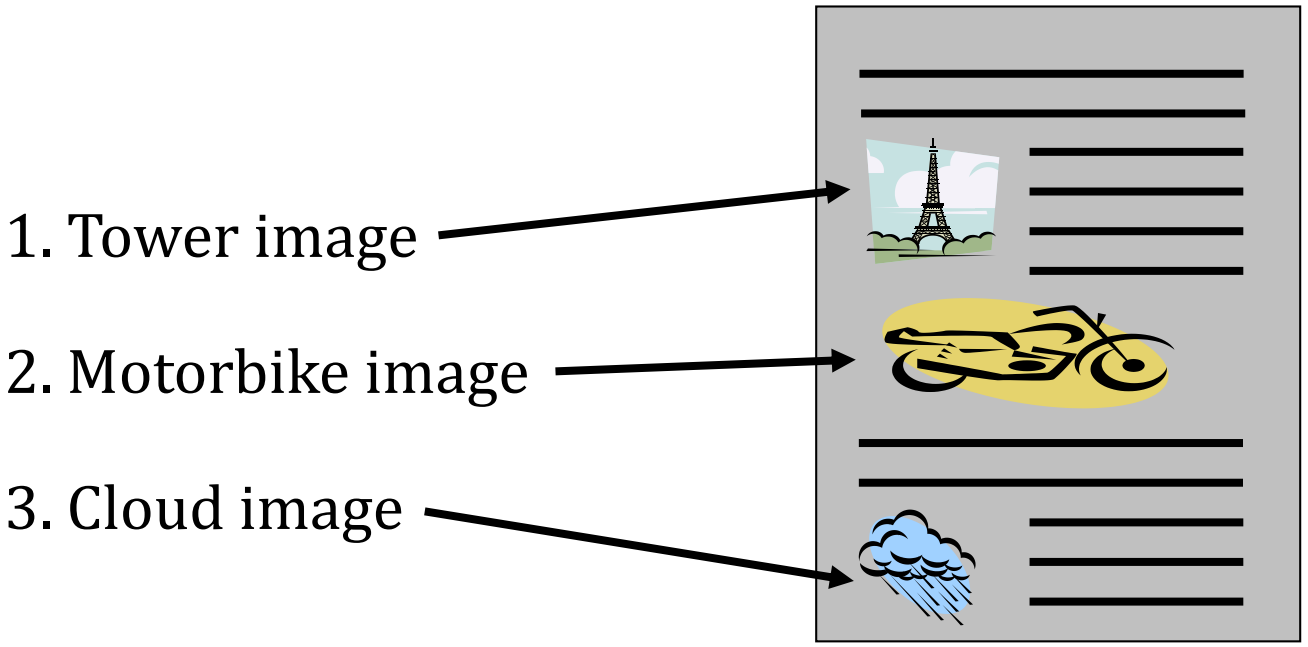
HTTP connections

- Non-persistent HTTP
  - Establishes one TCP connection
  - One object is sent over a TCP connection
  - Closes that TCP connection
  - Repeat this process for each object
  - HTTP/1.0 uses non-persistent HTTP

- Persistent HTTP
  - Multiple objects can be sent over a single TCP connection between client and server
  - No need to establish/close multiple connections
  - HTTP/1.1 uses persistent connections in default mode

Example: the assumptions

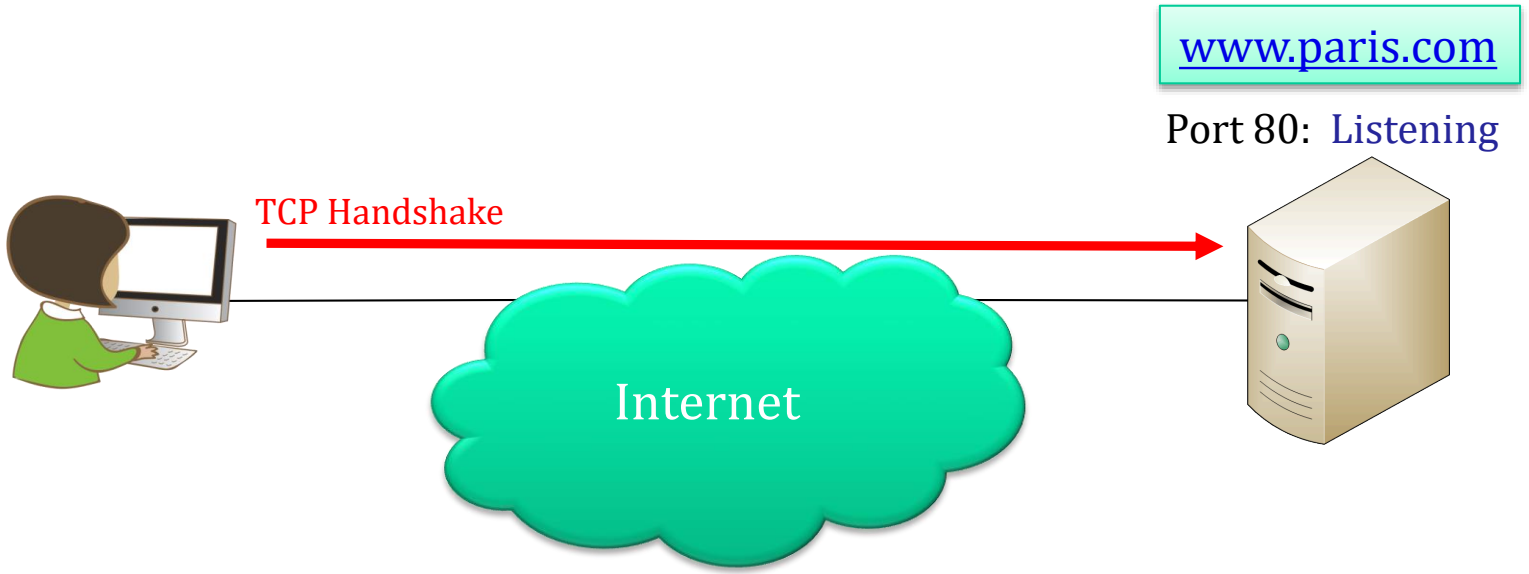
- The main page of a server has three image objects which reside on the same server
- A client browser wants to download the page



http://www.paris.com/index.html

Non-persistent HTTP

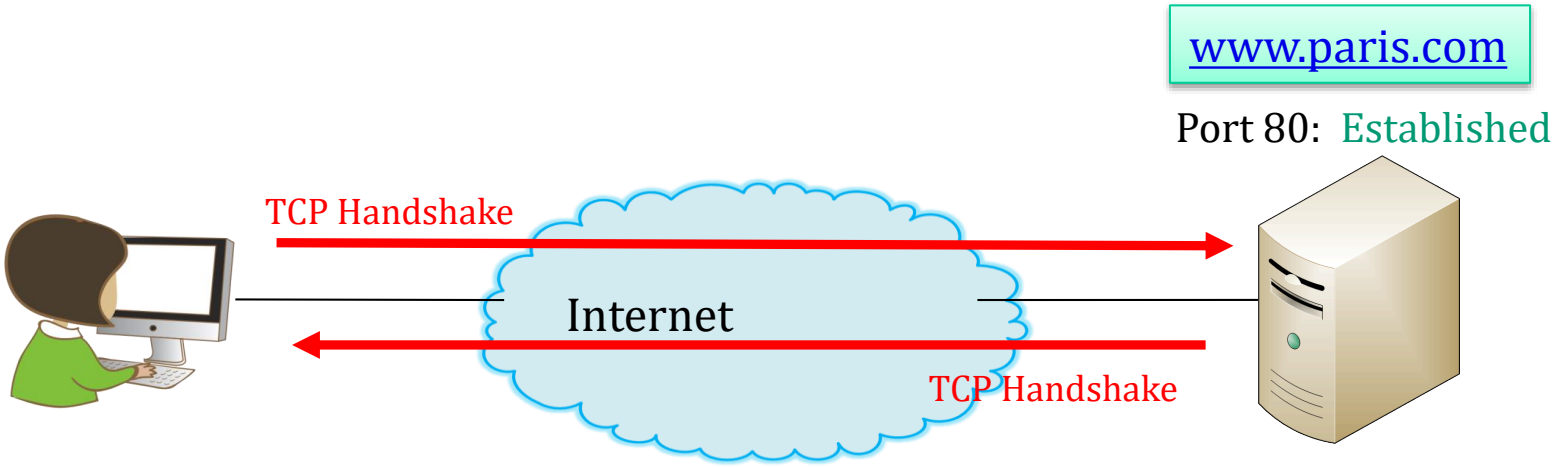
- http://www.paris.com/index.html contains text, and references to 3 jpeg images)



1. HTTP client initiates TCP connection to HTTP server (process)

Non-persistent HTTP

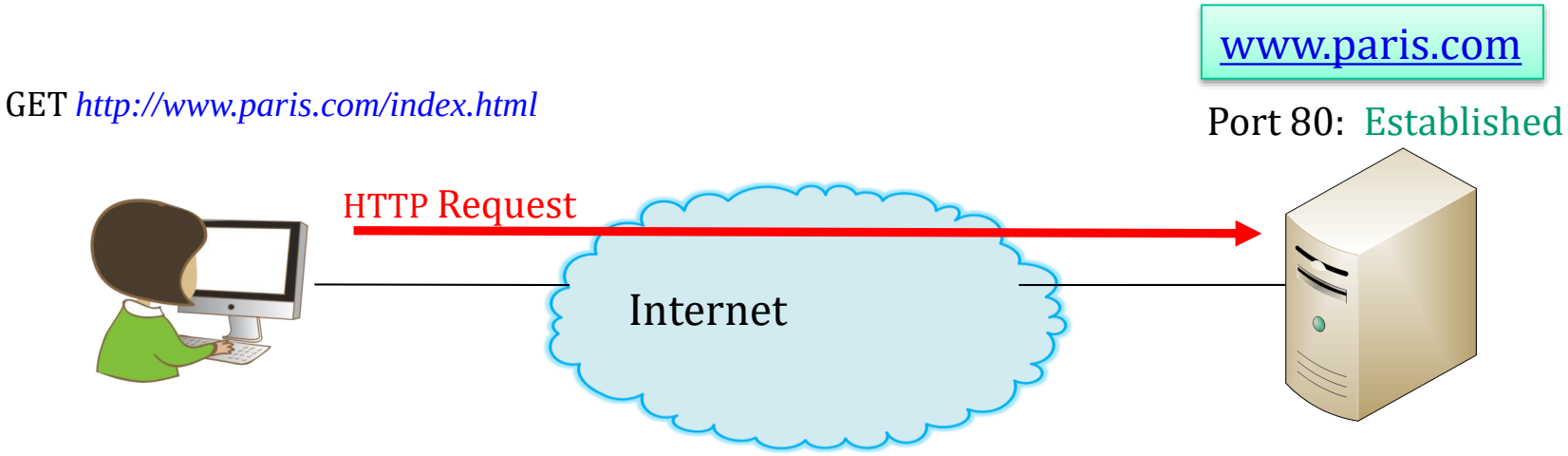
- *http://www.paris.com/index.html* contains text, and references to 3 jpeg images)



2. HTTP server waits for TCP connection at port 80.
3. Then it "accepts" connection from client
4. TCP handshake is sent to client for confirmation of the established connection

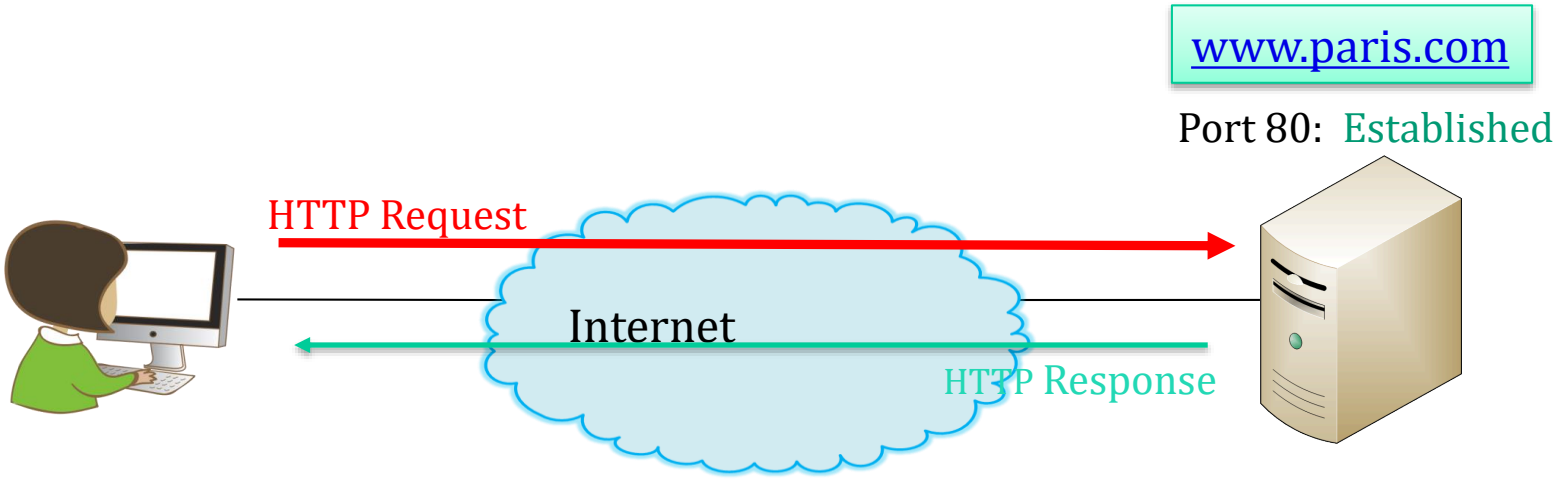
Non-persistent HTTP

- *http://www.paris.com/index.html* contains text, and references to 3 jpeg images)



5. HTTPclient sends HTTP *request message* (containing URL) into TCP connection socket. Message indicates that client wants a base html file

Non-persistent HTTP



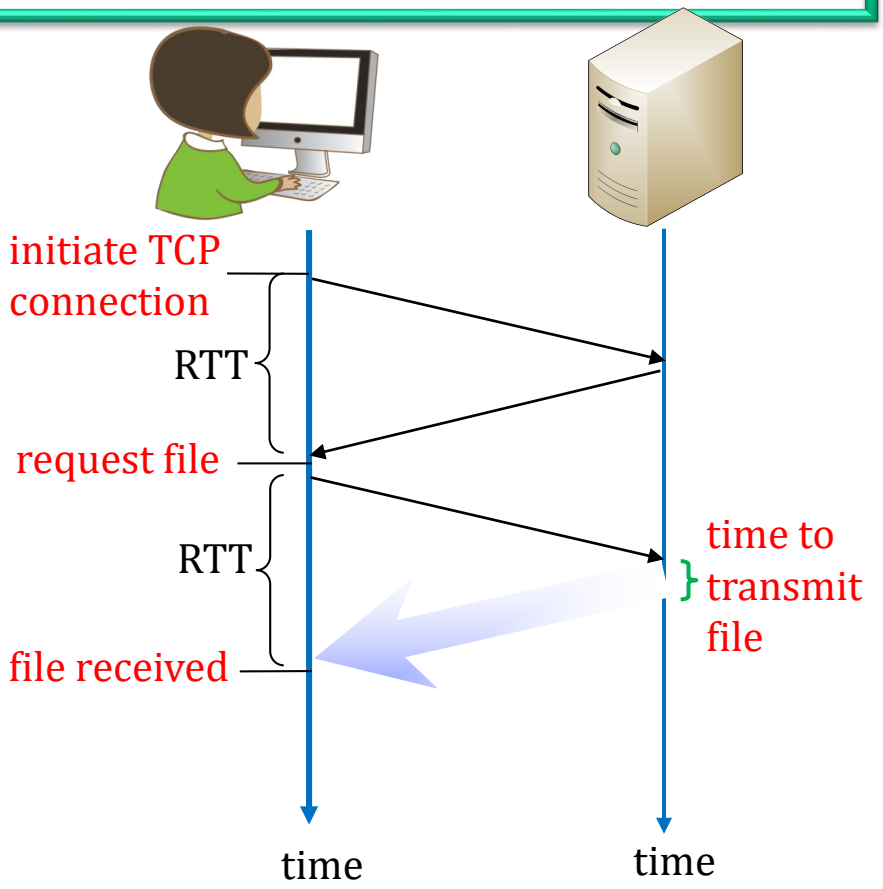
6. HTTP server receives request message, forms *response message* containing requested object, and sends message into its socket

HTTP/1.0 200 OK
Date: Fri, 16 Aug 2007 11:48:52 GMT
Server: Apache/1.1.1 UKWeb/1.0
Content-type: text/html
Content-length: 3406
Last-modified: Fri, 09 Aug 2007 14:21:40 GMT

<< index.html >>

HTTP Request and Response Latency

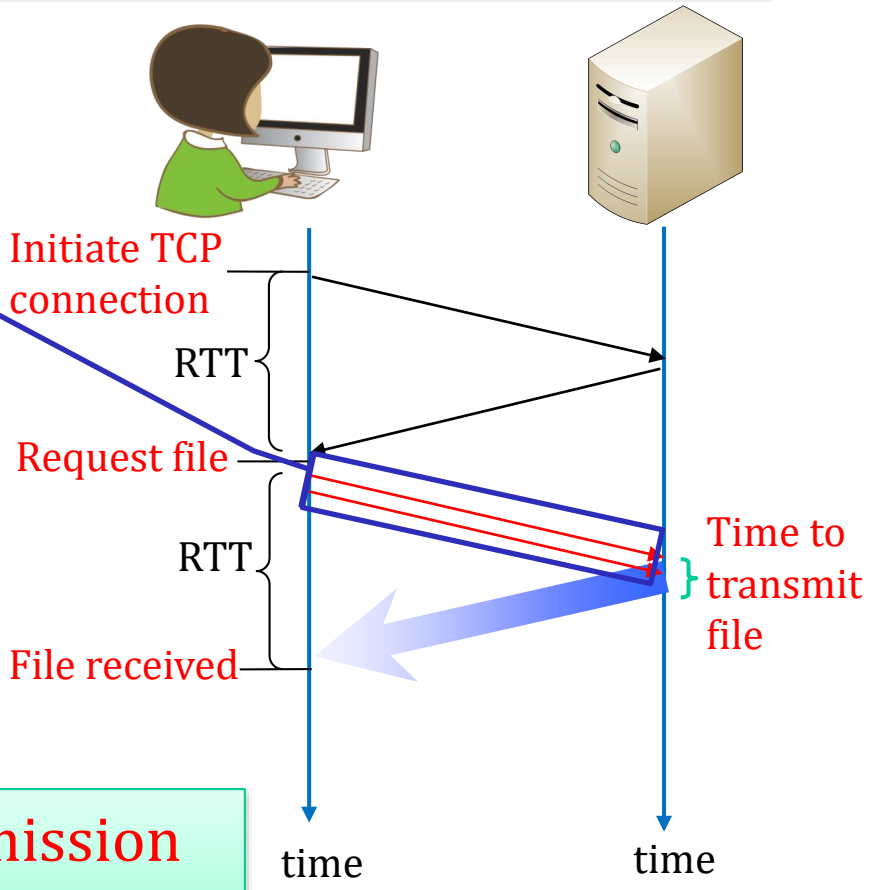
- Definition of RTT:
  - Time required to send a small packet from a client to server and back
- Total time for downloading the base object:
  - One RTT to initiate TCP connection
  - One RTT for HTTP request and first few bytes of HTTP response to return
  - Plus file transmission time (neglecting other delay factors)



Total time = 2RTT + file transmit time

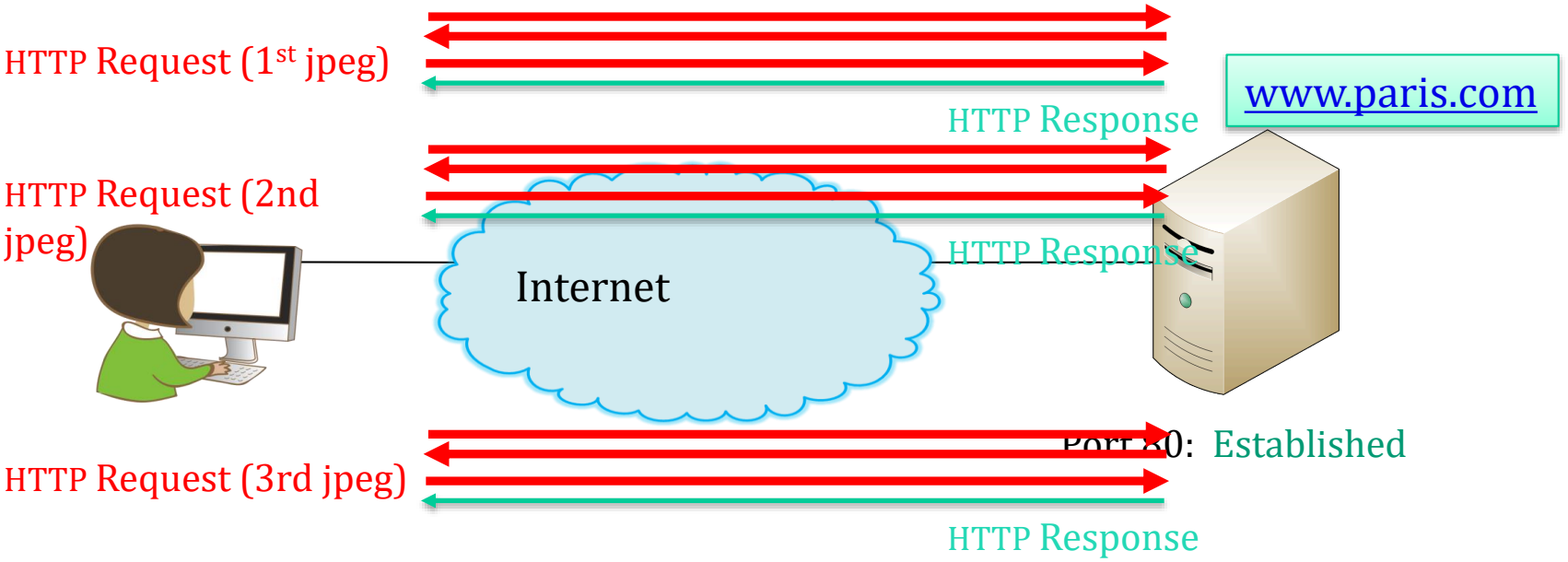
HTTP ACK and GET: Reality

- Two packets are sent by browser:
  - One for ACK
  - One for http request
- Multiple packets may be sent for a single file transmission
- Total time for obtaining the first file can be approximated as



Total time = 2RTT + file transmission time

Non-persistent HTTP



7. HTTP client receives response message containing html file, displays html. Parsing html file, yields 3 referenced jpeg objects

8. Steps 1-6 repeated for each of the 3 jpeg objects

Non-persistent HTTP summary

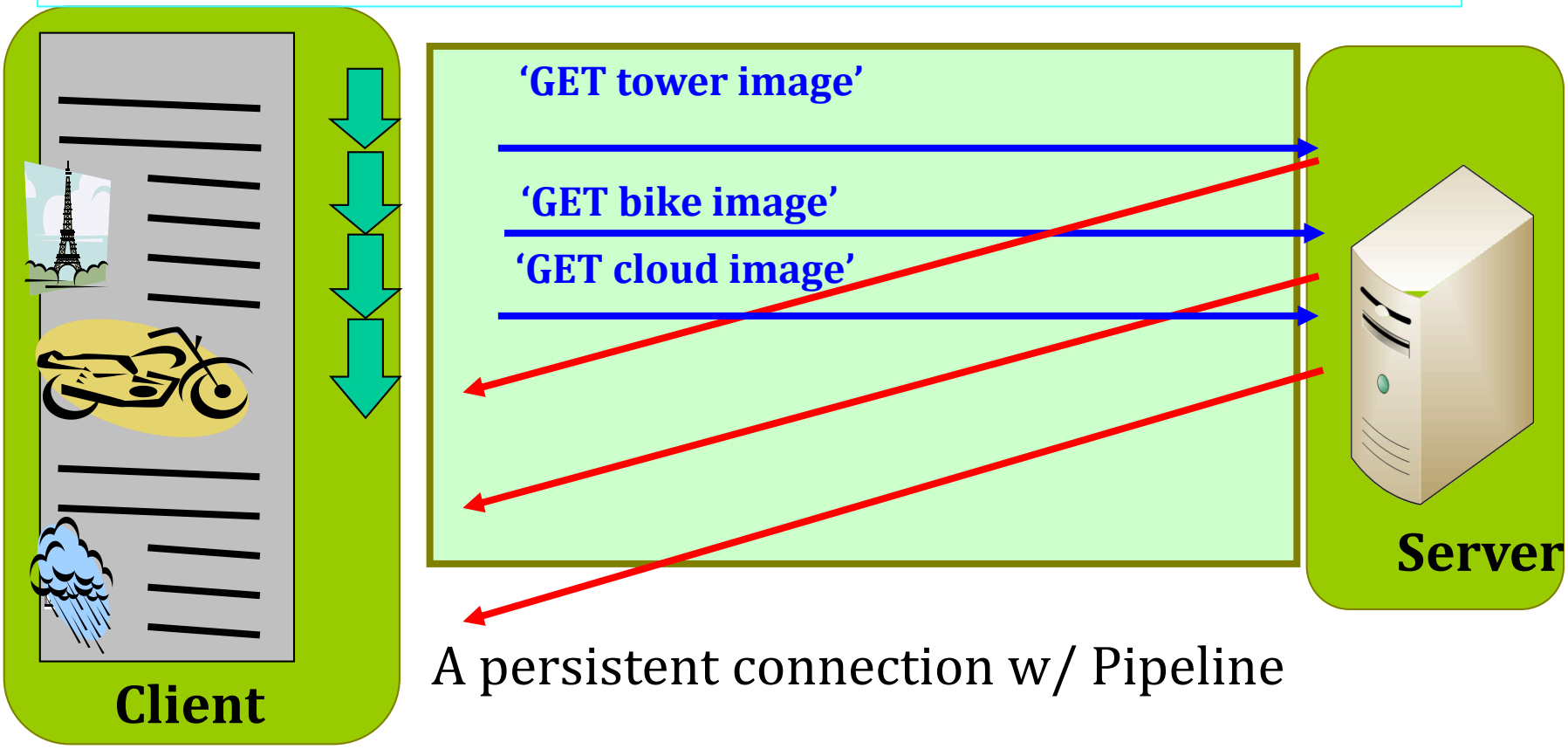
- One connection is established for each object
- Server closes connection after sending an object
- Requires 2 RTTs per object
- OS overhead for each TCP connection
- After the base html file is processed by the client browser, browser opens parallel TCP connections to fetch referenced objects

Persistent HTTP

- A single connection for all objects
  - Server keeps connection open after sending response
  - Subsequent HTTP messages between same client and server sent over the opened connection
- Persistent without pipelining:
  - Client issues new request only when previous response has been received
  - One RTT for each referenced object
- Persistent with pipelining:
  - Default in HTTP/1.1
  - Client sends multiple requests without waiting for response objects
  - One RTT for all the referenced objects

Pipelining

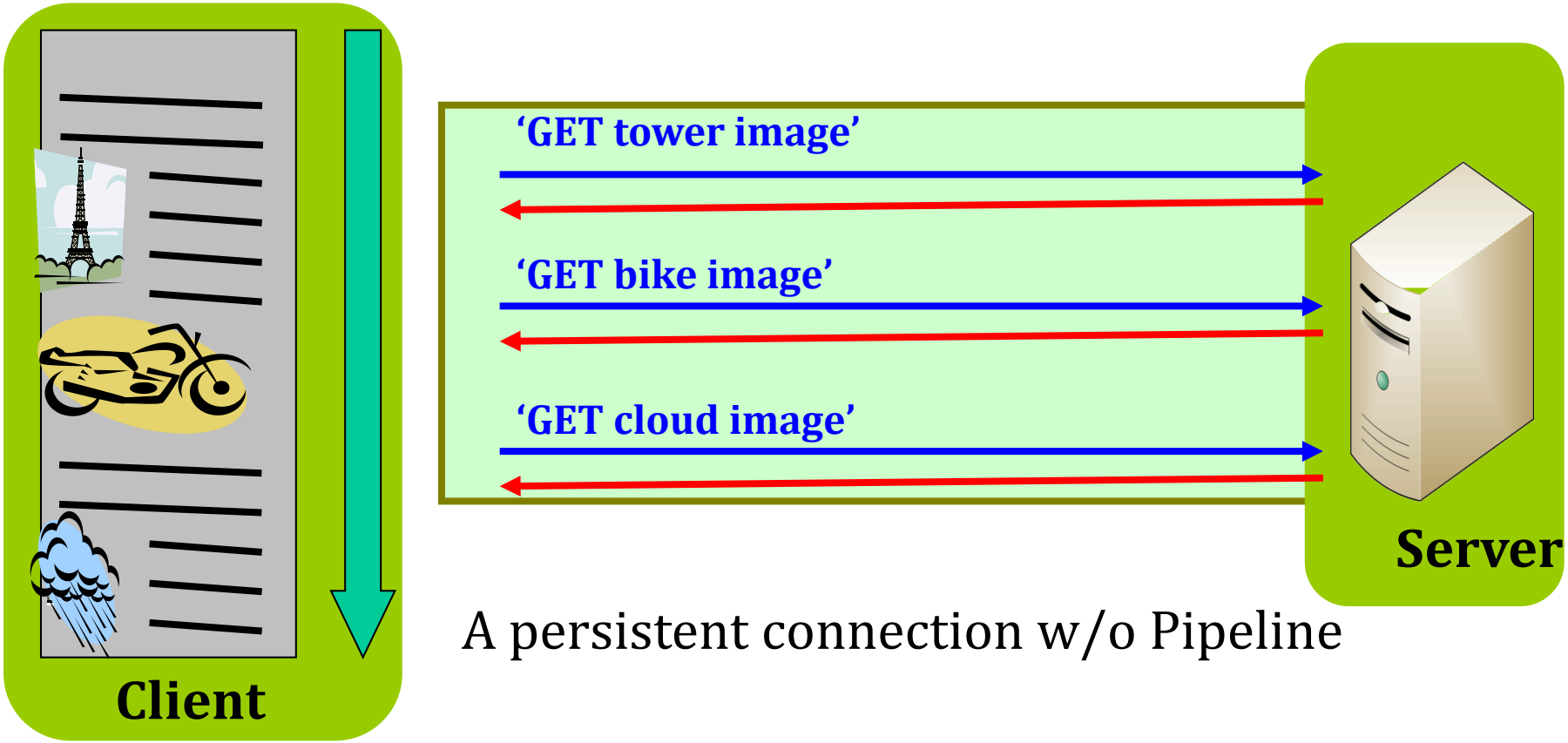
- A client browser can request image files after interpreting HTML tags



A persistent connection w/ Pipeline

Non-Pipelining

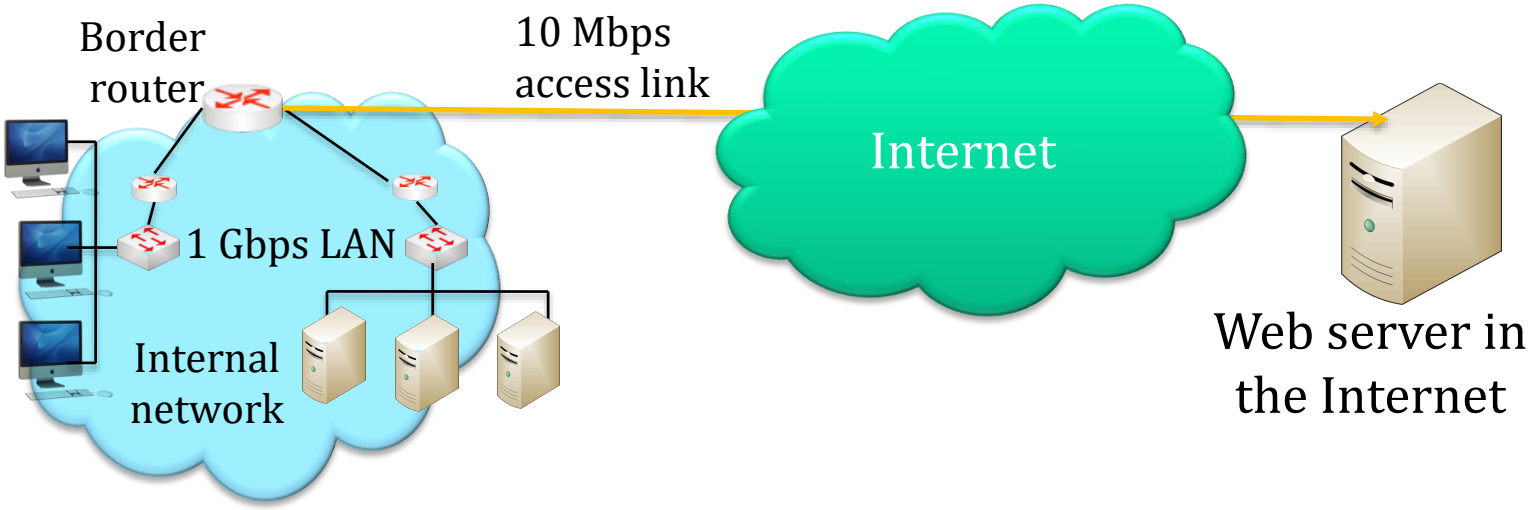
- After completely interpreting a HTML document, a client browser requests image files one by one



A persistent connection w/o Pipeline

Example: delay (1)

- For most organizations, the access link to the Internet is almost always full, hence there is a long queuing delay when sending a packet
  - Assume the average queuing delay at the border router is 500 ms when a packet travels to the Internet
  - When a response packet travels from Internet to a Gbps LAN, the queuing delay is negligible when compared with that of the request packet
- Assume the distance to web server is 100 Km, then the propagation delay RTT ≈ 2 * 100 Km/ 2* 10^8 m/s = 1 ms



Example: delay (2)

- Assume downloading a homepage that has only one base file 1000 Kbits long
- Neglecting all other delays, a HTTP request and response can be approximated as
  - One round trip to establish a connection = queuing delay + propagation delay for RTT = 500 ms + 1 ms = 501 ms
  - One round trip to obtain and download the file = queuing delay + propagation delay for the request + file transmission delay + file propagation delay = 500 ms + 0.5 ms + 1000 Kbits / 10 Mbps + 0.5 ms = 500 + 1 + 100 = 601 ms
  - Total delay for downloading the home page = 501 + 601 = 1102 ms

Outline

Part 1

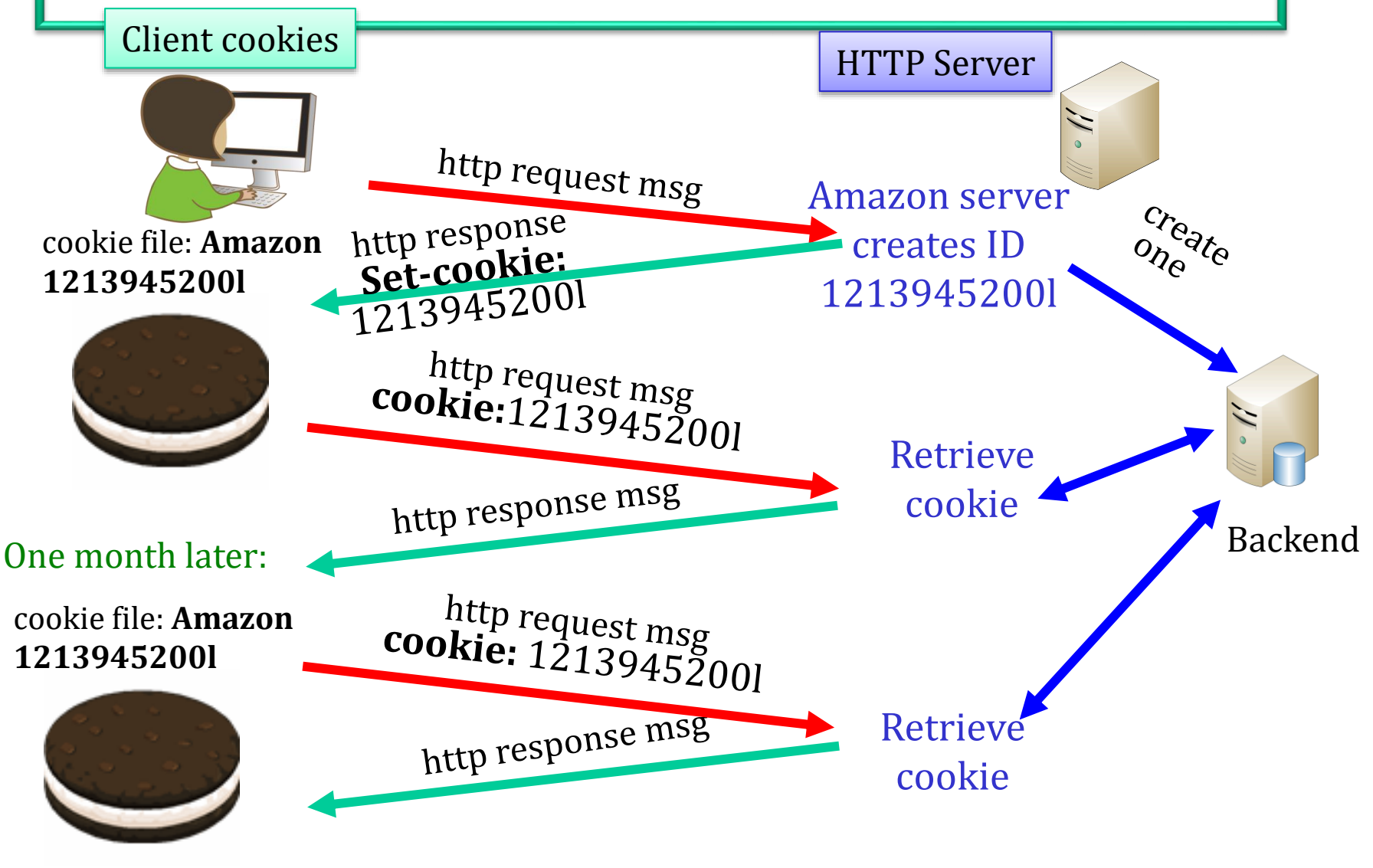
- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookie
- Proxy
- FTP
- email

Enable state in HTTP using cookies

- Many major Web sites use cookies for maintaining state information
- Four components
  - Cookie header line in HTTP response message
  - Cookie header line in HTTP request message
  - Cookie files kept on user's host, managed by user's browser
  - Back-end database at Web site

- Cookie creation:
  - Alice always accesses Internet from the same PC
  - Alice visits an e-commerce site, e.g. amazon.com, for first time
  - When initial HTTP request arrives at site, site creates:
    - Unique ID
    - Entry in backend database for ID

Cookies: keeping state information



Set-Cookies by amazon.com

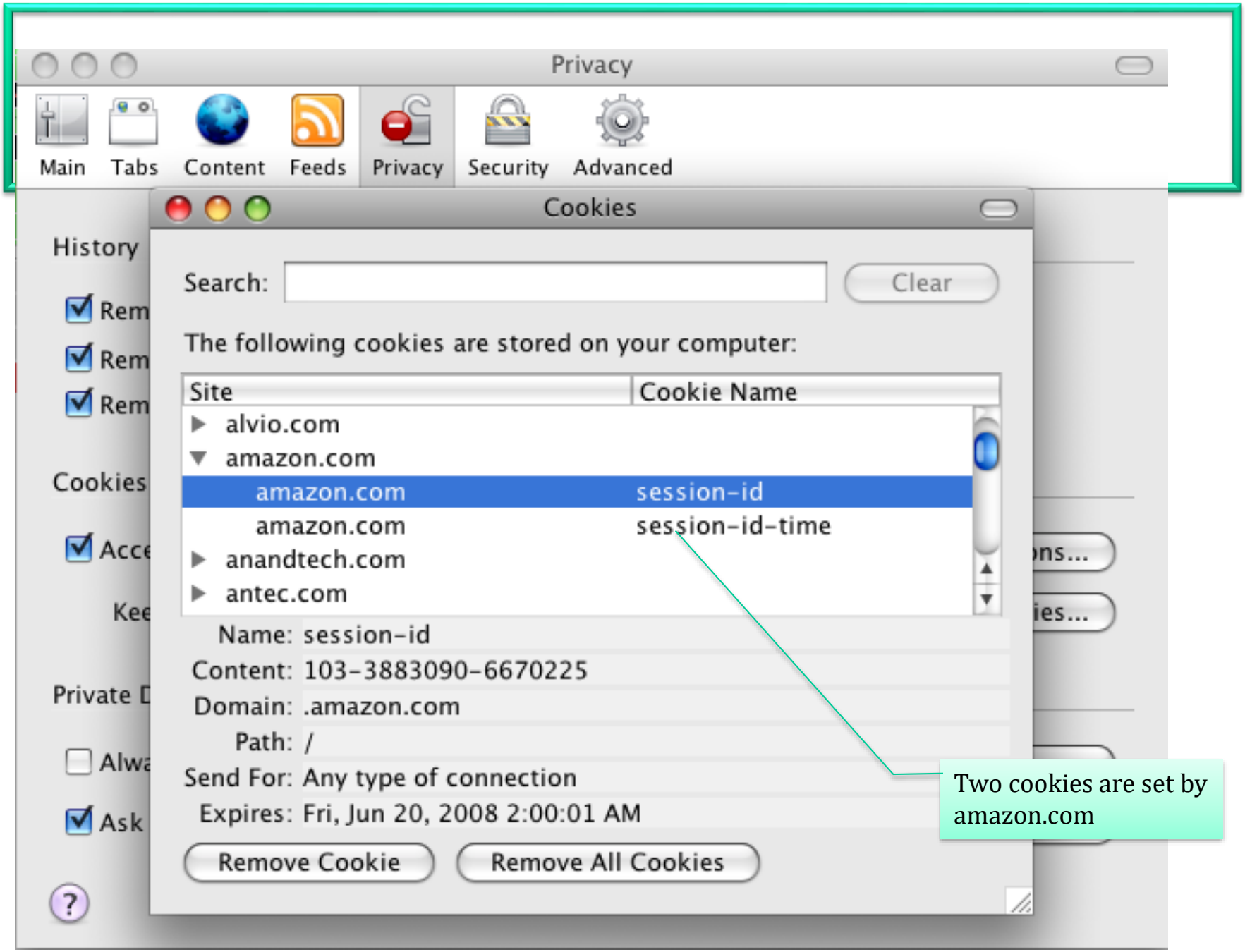
No.	Time	Source	Destination	Protocol	Info
73	1.424269	72.21.210.11	192.168.127.4	HTTP	HTTP/1.1 200 OK (text/html)

```
Frame 73 (1202 bytes on wire, 1202 bytes captured)
Ethernet II, Src: Cisco_b0:c2:07 (00:1e:4a:b0:c2:07), Dst: AsustekC_be:fa:c1 (00:1d:60:be:fa:c1)
Internet Protocol, Src: 72.21.210.11 (72.21.210.11), Dst: 192.168.127.4 (192.168.127.4)
Transmission Control Protocol, Src Port: http (80), Dst Port: 50640 (50640), Seq: 35881, Ack: 411, Len: 1148
[Reassembled TCP Segments (37028 bytes): #16(1380), #17(1380), #19(1380), #20(1380), #22(1380), #23(1380), #25(1380), #26(1380), #27(13
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
    Request Version: HTTP/1.1
    Response Code: 200
  Date: Fri, 13 Jun 2008 19:41:48 GMT\r\n
  Server: Server\r\n
  Set-Cookie: skin=noskin; path=/; domain=.amazon.com; expires=Fri, 13-Jun-2008 19:41:48 GMT\r\n
  x-amz-id-1: 0GEFX9Y5884K83NJF5D7\r\n
  x-amz-id-2: welDMn6OkLmK5aOSxJ6u/s+8zPgwwzcr\r\n
  Set-cookie: session-id-time=1213945200l; path=/; domain=.amazon.com; expires=Fri Jun 20 07:00:00 2008 GMT\r\n
  Set-cookie: session-id=103-3883090-6670225; path=/; domain=.amazon.com; expires=Fri Jun 20 07:00:00 2008 GMT\r\n
  Vary: Accept-Encoding,User-Agent\r\n
  Content-Encoding: gzip\r\n
  Content-Type: text/html; charset=ISO-8859-1\r\n
  Transfer-Encoding: chunked\r\n
  \r\n
  HTTP chunked response
  Content-encoded entity body (gzip): 36258 bytes -> 259651 bytes
Line-based text data: text/html
```

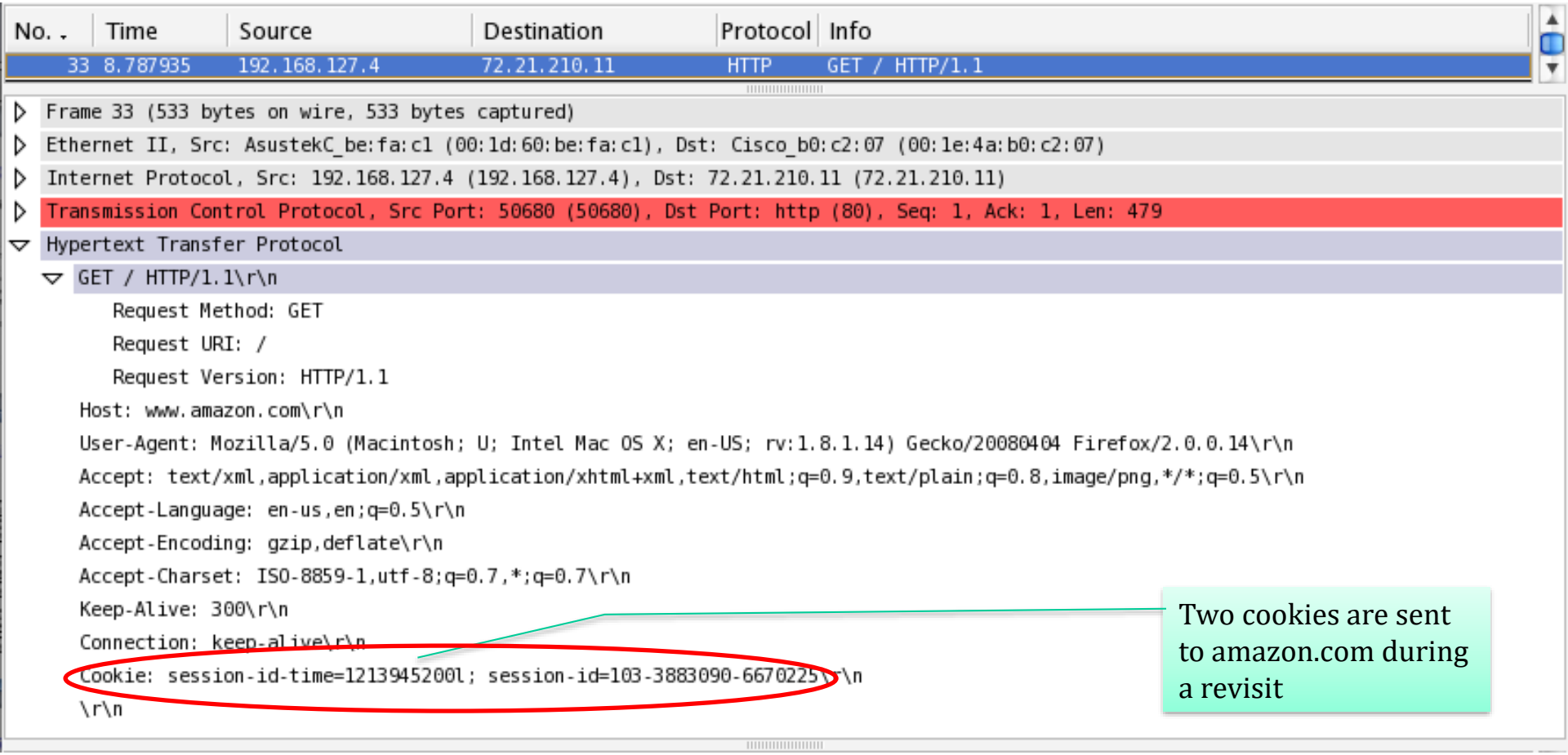
Two cookies are set by amazon.com during the first visit

Two cookies

- Cookie:
  - session-id-time=1213945200l
  - session-id=103-3883090-6670225


Privacy
Main
Tabs
Content
Feeds
Privacy
Security
Advanced
History
Cookies
Private
Cookies
Search:
Clear
The following cookies are stored on your computer:
Site
Cookie Name
alvio.com
amazon.com
amazon.com
session-id
amazon.com
session-id-time
anandtech.com
antec.com
Name: session-id
Content: 103-3883090-6670225
Domain: .amazon.com
Path: /
Send For: Any type of connection
Expires: Fri, Jun 20, 2008 2:00:01 AM
Remove Cookie
Remove All Cookies
Two cookies are set by amazon.com

Revisit amazon.com



Set a cookie

- A browser requests a page from web servers by sending them a short text called a HTTP request
- For example, to access the page http://www.amazon.com/index.html, a browser connects to the server www.amazon.com by sending it the following request:

browser → server

```
GET /index.html HTTP/1.1
Host: www.amazon.com
```

- The server replies by sending the requested page preceded by a similar packet of text, called the HTTP header
- This packet may contain lines requesting the browser to store cookies:

browser ← server

```
HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: name=value
 ……
(content of page)
```

Cookies

- The line Set-cookie is only sent if the server wishes the browser to store a cookie
- It requests the browser to store the string name=value and send it back in all future requests to the server
- If the browser supports cookies and they are enabled, every subsequent page request to the same server will contain the cookie
- For example, the browser requests the page http://www.amazon.com/index.html by sending the server www.amazon.com the following request:

browser ⟶ server

```
GET /index.html HTTP/1.1
Host: www.amazon.com
Cookie: name=value
Accept: */*
```

Cookie format

- The following is a cookie sent by a Web server :

Name:	apn-user-id
Content:	P7V03VZX5KQCC
Domain:	.amazon.com
Path:	/
Send For:	Any type of connection
Expires:	Thu, Jan 1, 2037 2:00:01 AM

- The name of this particular cookie is apn-user-id, while its value is the string P7V03VZX5KQCC
- The path, domain strings / and .amazon.com tell the browser to send the cookie when requesting an arbitrary page of the domain amazon.com, with an arbitrary path

Cookie Expiration

- Cookies expire, and therefore not sent by the browser to the server, under one of the following conditions:
  - At the end of the user session, i.e. when the browser is shut down, if the cookie is not persistent
  - A specified expiration date has passed
  - The expiration date of the cookie is changed (by the server or the script) to a date in the past
  - The browser deletes the cookie in response to user
- The third condition allows a server or script to explicitly delete a cookie

Cookie Authentication

- Cookies can be used by a server to
  - Recognize authenticated users
  - Personalize the web pages of a site
- This authentication procedure can be accomplished as follows:
  - The user inserts username and password in the text fields of a login page and sends them to the server
  - The server receives username and password and checks them
    - if correct, it sends back a page confirming that login has been successful together with a cookie, storing the pair user/cookie
  - Every time the user requests a page from the server, the browser automatically sends the cookie back to the server
    - The server compares the cookie with those stored
    - If a match is found, the server knows which user has requested that page
  - Single sign-on relies on cookies too

Cookies uses and risks

- Cookies are used for:
  - User session state
  - Authentication
  - Shopping carts
  - Collecting statistics and generating recommendations
  - Collect marketing or advertising information
- Risks
  - An intrusion to privacy
  - Loss of authentication information stored in cookies

Outline

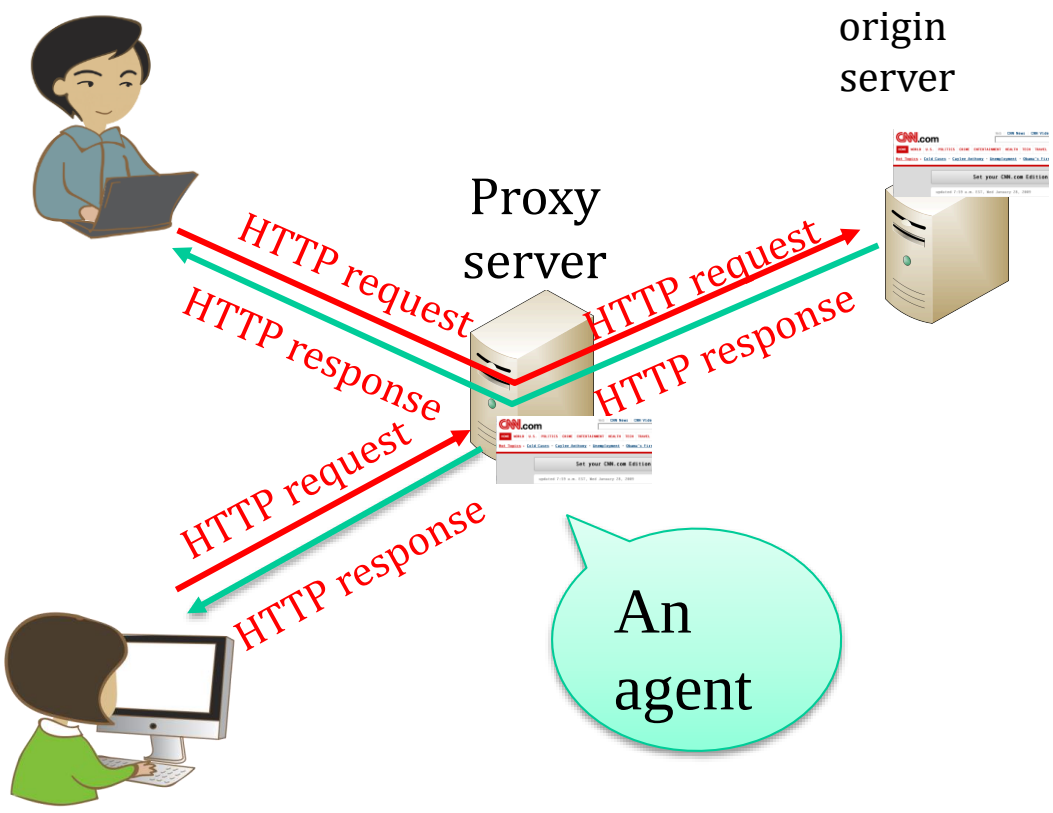
Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookie
- Proxy
- FTP
- email

Web cache/proxy server

Goal: Reduce the request and file download time

- Web accesses via a proxy
  - Bob requests cnn.com
  - If requested objects in cache are up-to-date, proxy returns objects to Alice
  - Or proxy requests and downloads objects from origin server, then returns object to Alice

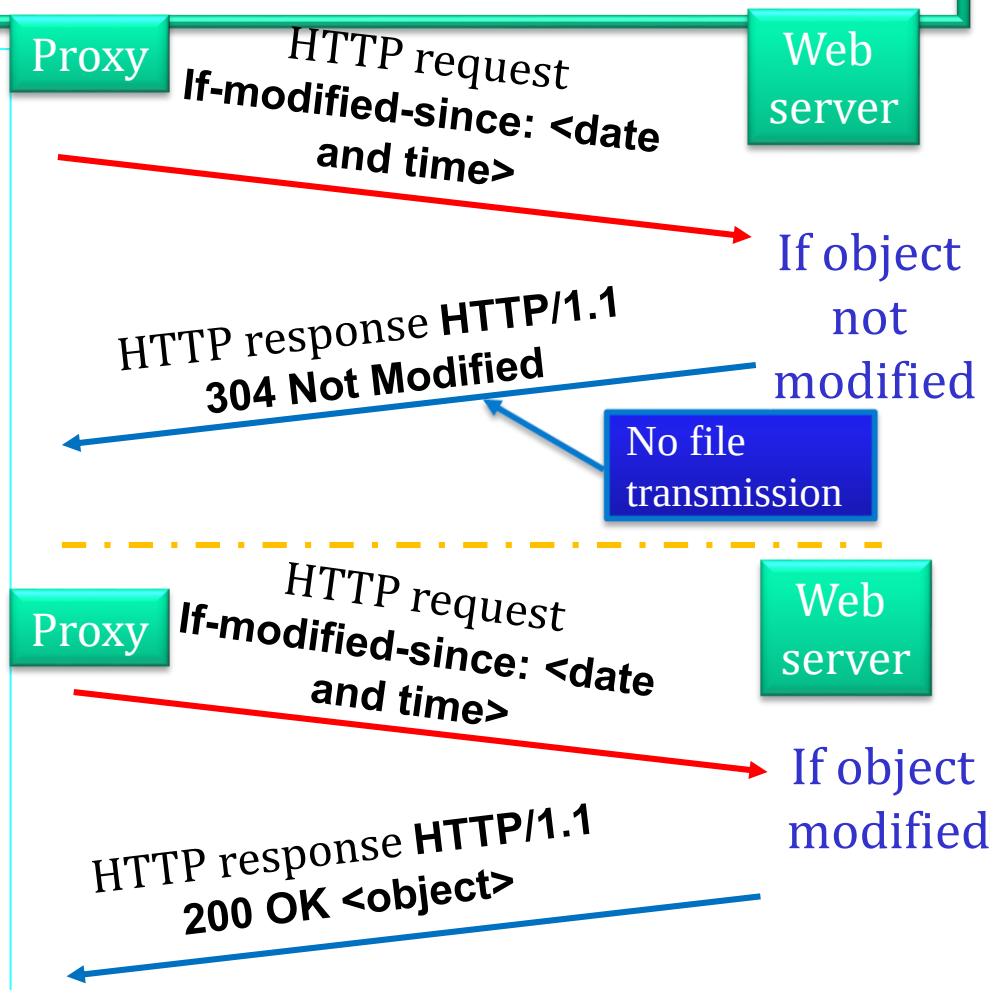


Proxy Caching

- Bob requests http://www.cnn.com/index.html
  - Browser sends “GET www.cnn.com” to the proxy
  - Proxy sends “GET www.cnn.com” to the web server
  - Web server sends response to the proxy
  - Proxy stores the response in cache, and forwards to Bob
- Alice requests http://www.cnn.com/index.html
  - Browser sends “GET www.cnn.com” to the proxy
  - Proxy makes sure that the content is up-to-date and sends response to the Alice from the cache
- Benefits
  - Faster response time to clients
  - Reduced bandwidth consumption

Sync cache/replica: conditional GET

- Reduce the file request and transmission time
  - No object update if cache has up-to-date version
  - Specified in RFC 2616 http/1.1
- Proxy:
  - Specify date of cached copy in HTTP request
  - If-modified-since: <date and time>
    - RFC 2822 formatted <date and time> as the value: Tue, 27 Jan 2009 05:25:11 GMT
- Web server:
  - Response contains no object if cached copy is up-to-date:
  - HTTP/1.1 304 not modified



Subsequent requesting and delivering the objects contained in the web page

- The file transmission delay for the page's object is eliminated
- This also removes the associated processing delays, propagation delays, and queueing delays for subsequent requesting and delivering the objects contained in the web page
- As a consequence, the congested border router of an organization may have fewer outgoing http request packets and reduces the queueing delays for achieving a faster Internet link to an ISP
- Conditional POST can also be used in a similar manner

Proxy roles

- Proxies play two roles
  - A server to the client
  - A client to the destination server
- Performance acceleration for information sharing
  - Provide LAN-like services in a WAN network that connects branch offices
    - Remove long RTT by caching and prefetching
    - File compression
- Security role
  - No important information is stored locally
  - Serving as a sacrificial lamb in case of penetration
  - Problem: If the cache is poisoned, then attacks can be propagated to computer to access the cache

Proxy types

- Explicit configuration
  - Browser configured to use a proxy
  - Directs all requests through the proxy
  - Problem: requires user action
- Transparent proxy (or interception proxy)
  - Proxy lies in path from the client to the servers
    - Proxy intercepts packets en route to the server
    - Benefit: does not require user action
    - Overhead of reconstructing the requests/responses
  - Location: at the border router of a campus or company
  - Could be a violation of user privacy
    - User does not know the proxy lies in the path
    - Proxy may keep logs of the user's activities

Functions of Proxies

- Transcoding
  - Converting data from one form to another
    - E.g., reducing the size of images for cell-phone browsers
  - ISPs improve the link performance for dial-up lines
- Prefetching
  - Requesting content before the user asks for it
- Traffic shaping
  - Limit the bandwidth for certain types of traffic such as P2P
- Filtering/security
  - Blocking access to sites, based on URL or content
  - IDS/IPS: Blocking a malicious packet based on content
  - Data leakage prevention
- Anonymity
  - Server sees requests coming from the proxy IP address
  - Server cannot track the individual user IP address

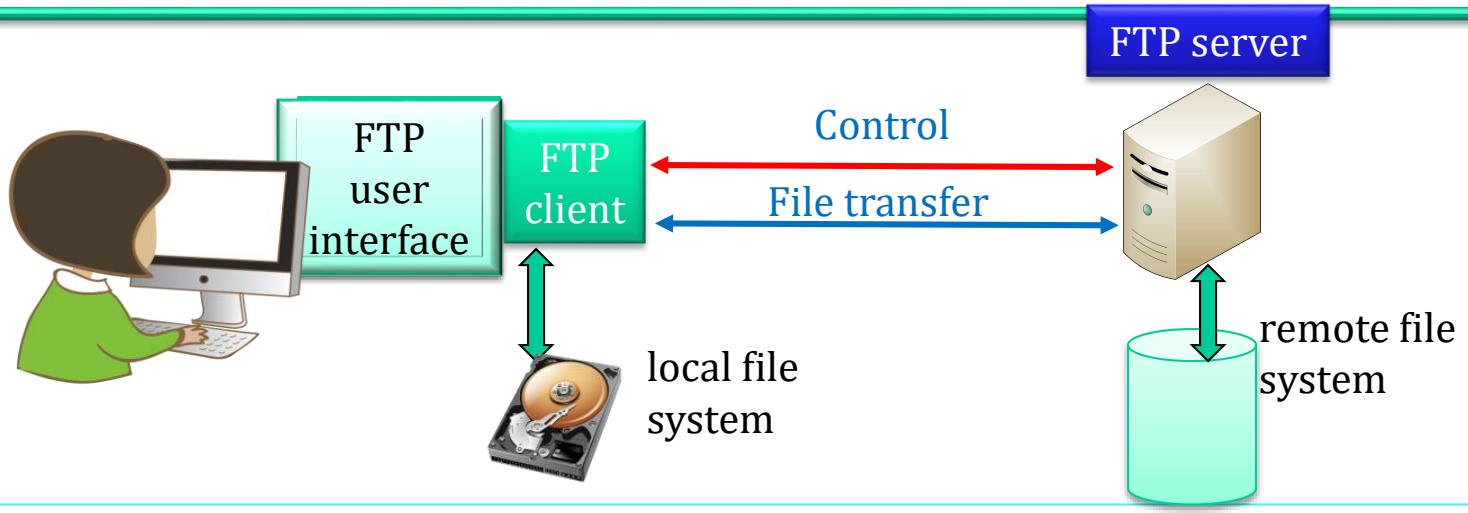
The first proxy server

Outline

Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services
- HTTP
- Cookie
- Proxy
- FTP
- email

FTP: the file transfer protocol



- Transfer file to/from remote FTP server
- Client/server model
  - FTP client: initiates transfer (either to/from remote)
- ftp standard: RFC 959
- ftp server:
  - TCP based
  - Port 21: for control functions such as login, change directory, etc.

FTP uses two separate connections

- FTP client contacts FTP server at port 21 and establishes a control connection
  - Control connection: port 21
  - Client is authenticated over control connection
  - Client browses remote directory by sending commands over control connection
- When server receives file transfer command, server opens a 2nd TCP connection (for file transfer) to client
  - After transferring one file, FTP server closes data connection
  - FTP server opens another TCP data connection to transfer another file
- FTP server maintains state information: current directory, and user authentication

FTP vs. SFTP

- FTP is not a secure protocol: Password is sent in clear text
- SSH File Transfer Protocol (aka Secure File Transfer Protocol or SFTP) is a secure protocol
  - IETF Internet Draft Based on SSH (Secure Shell) protocol (port 22)
    - Use with the SSH Connection Protocol
    - SFTP protocol follows a simple request-response model
      - Each request and response contains a sequence number and multiple requests may be pending simultaneously
      - Each request has one or more response messages that may be returned in the result (e.g., a read either returns data or reports error status)
  - SFTP is not FTP run over SSH, but rather a new protocol
  - Better than SCP (secure copy):
    - Compared to the earlier SCP protocol, which allows only file transfers, the SFTP protocol allows for a range of operations on remote files
    - SFTP is a remote file system protocol
  - The additional capabilities of a SFTP client when compared to an SCP client include resuming interrupted transfers, directory listings, and remote file removal
- Move to SFTP for better security if a SFTP server is available

Secure Shell (SSH) protocol: RFC 4251

- Secure Shell (SSH) is a protocol for secure remote login and other secure network services over an insecure network
- It consists of three major components:
  - The Transport Layer Protocol provides server authentication, confidentiality, and integrity with perfect forward secrecy
    - It may optionally also provide compression
  - The User Authentication Protocol authenticates the client-side user to the server
    - It runs over the transport layer protocol
  - The Connection Protocol multiplexes the encrypted tunnel into several logical channels
    - It runs over the user authentication protocol
    - channels can be used for a wide range of purposes: e.g. setting up secure interactive shell sessions
- SSH allows new protocols to be defined and coexist
  - The client sends a service request once a secure transport layer connection has been established
  - A second service request is sent after user authentication is complete

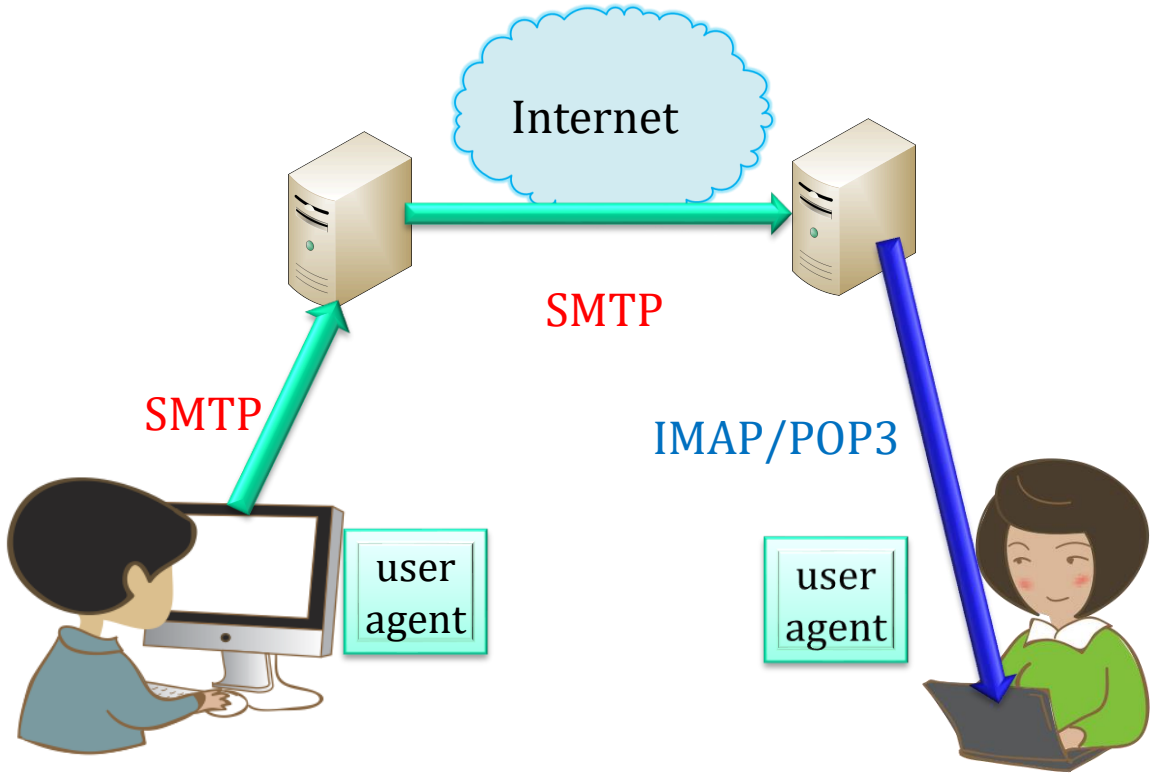
Outline

Part 1

- Overview of application Layer
- Client server and P2P architecture
- Socket
- Transport layer services HTTP
- Cookie
- Proxy
- FTP
- email

Electronic Mail

- Four components:
  - User agents
  - Mail servers
  - Simple mail transfer protocol (SMTP) for sending email
  - IMAP/POP3 for email retrieval
- User Agent (UA)
  - Composing, editing, and reading received mail messages
  - E.g., Eudora, Outlook, Mozilla Thunderbird
  - Outgoing, and incoming messages can be stored on mail server



Email server

- Mail Server
  - Mailbox (inbox) contains incoming/received messages for user
  - Message queue (outbox) contains outgoing (to be sent) mail messages
  - SMTP protocol between mail servers to send email messages
    - Sending server: sending mail server
    - Receiving server: receiving mail server
    - SMTP: RFC 2821

- SMTP is based on TCP to reliably transfer email message from client to server using port 25
- Direct transfer: sending server to receiving server using port 25
- SMTP has three phases:
  - handshaking
  - transfer of messages
  - Closing connection
- Command/response interaction
  - command: ASCII text
  - response: status code and phrase
- Messages must be in 7-bit ASCII

Bob sends a message to Alice

① Bob uses his user agent to compose message "to" for alice@auburn.edu

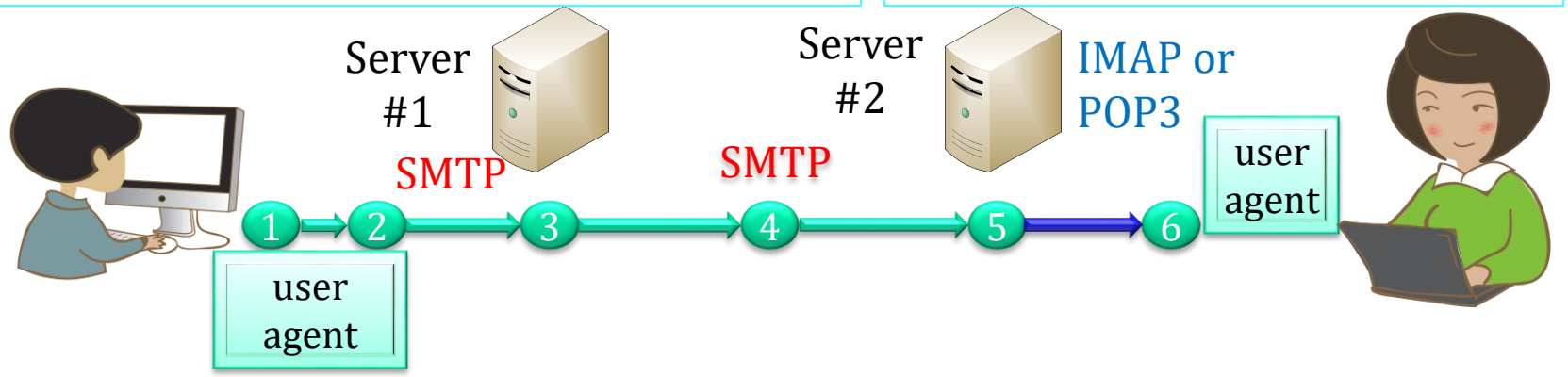
② Bob's UA sends message to his mail server (Server #1) and message is placed in message queue

③ Client side of SMTP server opens TCP connection with Alice's mail server (Server #2)

④ Server #1 sends Bob's message over the TCP connection to Alice's mail server (Server #2)

⑤ Alice's mail server places the message in Alice's mailbox

⑥ Alice invokes her user agent to read message



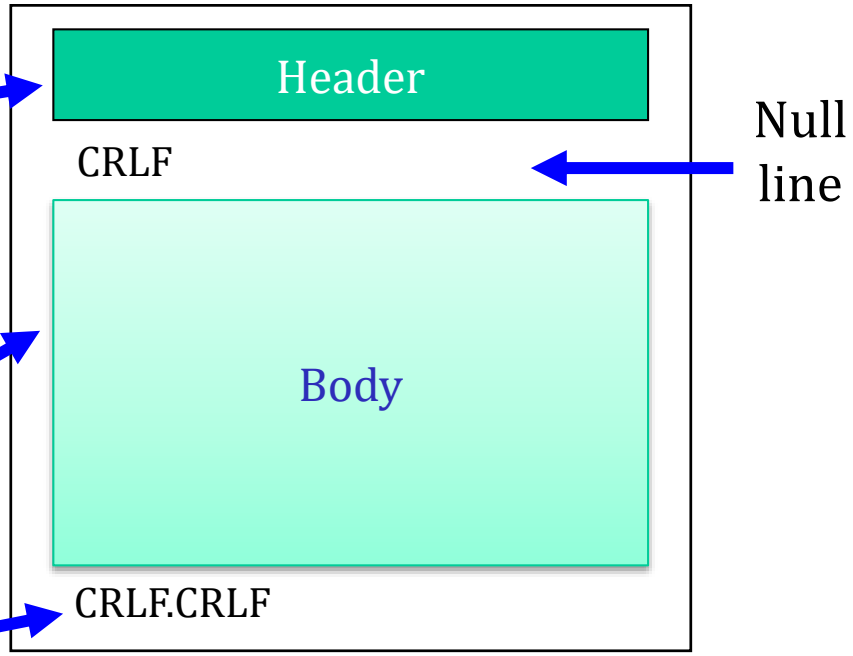
SMTP vs. HTTP

- HTTP: pull
- SMTP: push
- SMTP uses persistent connections
- SMTP requires message (header & body) to be in 7-bit ASCII
  - In contrast, HTTP allows Unicode for many languages
  - The 8-bit MIME extension was standardized in 1994
    - RFC1652
    - It facilitates the exchange of e-mail messages containing octets outside the seven-bit ASCII character set

- Both have ASCII command/response interaction, and status codes
- HTTP: each object is encapsulated in its own response message
- SMTP: multiple objects sent in a single multipart message
- Neither SMTP nor HTTP provide encryption

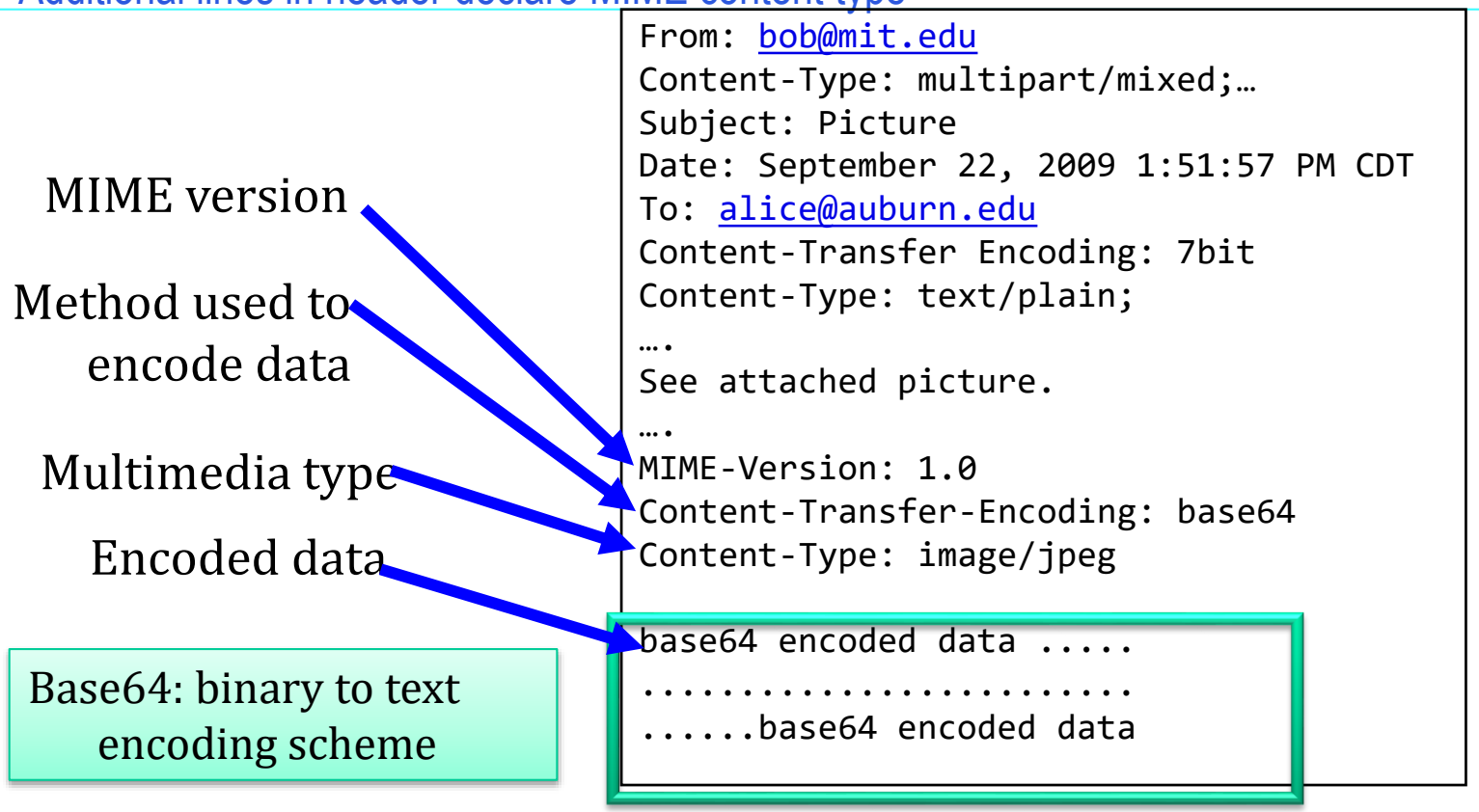
Mail message format

- RFC 822: standard for text message format:
- Header lines:
  - To:
  - From:
  - Subject:
- Body
  - the “message”, ASCII characters only
- End of message
  - CRLF.CRLF (0D0A2E0D0A)
  - The ASCII code of period is 2E



Message format: multimedia extensions

- MIME (Multipurpose Internet Mail Extensions): multimedia mail extension, RFC 2045, 2056
- Additional lines in header declare MIME content type

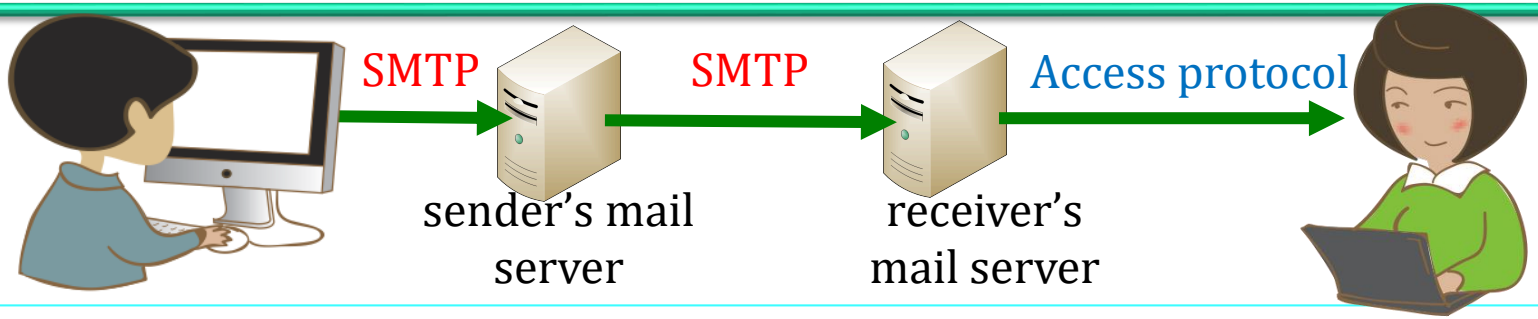


Wireshark

No. .	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.127.20	131.204.3.77	TCP	49589 > smtp [SYN] Seq=0 Win=65535 [TCP CHECKSUM INCORRECT] Len=0 MSS
2	0.002333	131.204.3.77	192.168.127.20	TCP	smtp > 49589 [SYN, ACK] Seq=0 Ack=1 Win=6144 Len=0 MSS=1460 WS=0
3	0.002393	192.168.127.20	131.204.3.77	TCP	49589 > smtp [ACK] Seq=1 Ack=1 Win=524280 [TCP CHECKSUM INCORRECT] Le
4	0.004568	131.204.3.77	192.168.127.20	SMTP	S: 220 **********
5	0.004588	192.168.127.20	131.204.3.77	TCP	49589 > smtp [ACK] Seq=1 Ack=114 Win=524280 [TCP CHECKSUM INCORRECT]
6	0.009823	192.168.127.20	131.204.3.77	SMTP	C: EHLO [192.168.127.20]
7	0.010319	131.204.3.77	192.168.127.20	TCP	smtp > 49589 [ACK] Seq=114 Ack=24 Win=7604 Len=0
8	0.010562	131.204.3.77	192.168.127.20	SMTP	S: 250-auburn.edu | 250-AUTH LOGIN | 250-8BITMIME | 250-SIZE | 250 XX
9	0.010578	192.168.127.20	131.204.3.77	TCP	49589 > smtp [ACK] Seq=24 Ack=184 Win=524280 [TCP CHECKSUM INCORRECT]
10	0.010893	192.168.127.20	131.204.3.77	SMTP	C: MAIL FROM:< auburn.edu>
11	0.011314	131.204.3.77	192.168.127.20	TCP	smtp > 49589 [ACK] Seq=184 Ack=55 Win=9064 Len=0
12	0.011811	131.204.3.77	192.168.127.20	SMTP	S: 250 Ok
13	0.011826	192.168.127.20	131.204.3.77	TCP	49589 > smtp [ACK] Seq=55 Ack=192 Win=524280 [TCP CHECKSUM INCORRECT]
14	0.011984	192.168.127.20	131.204.3.77	SMTP	C: RCPT TO:< @auburn.edu>
15	0.012565	131.204.3.77	192.168.127.20	TCP	smtp > 49589 [ACK] Seq=192 Ack=85 Win=10524 Len=0
16	0.012566	131.204.3.77	192.168.127.20	SMTP	S: 250 Ok
17	0.012586	192.168.127.20	131.204.3.77	TCP	49589 > smtp [ACK] Seq=85 Ack=200 Win=524280 [TCP CHECKSUM INCORRECT]
18	0.012732	192.168.127.20	131.204.3.77	SMTP	C: DATA
19	0.013312	131.204.3.77	192.168.127.20	TCP	smtp > 49589 [ACK] Seq=200 Ack=91 Win=11984 Len=0
20	0.013313	131.204.3.77	192.168.127.20	SMTP	S: 354 Enter mail, end with "." on a line by itself
21	0.013337	192.168.127.20	131.204.3.77	TCP	49589 > smtp [ACK] Seq=91 Ack=250 Win=524280 [TCP CHECKSUM INCORRECT]
22	0.013606	192.168.127.20	131.204.3.77	SMTP	C: DATA fragment, 1460 bytes
23	0.015066	131.204.3.77	192.168.127.20	TCP	smtp > 49589 [ACK] Seq=250 Ack=1551 Win=13444 Len=0
24	0.015096	192.168.127.20	131.204.3.77	SMTP	C: DATA fragment, 1460 bytes

- No.6: The command EHLO (Extended HELLO) defined in Extended SMTP (ESMTP) that is a definition of protocol extensions to the SMTP
  - The extension format was defined in IETF publication RFC 1869
- No. 7-8: A server will respond with success (code 250), failure (code 550) or error
- No. 22 and beyond: MIME message

Mail protocols



- SMTP: delivery from Bob to Alice's mail server
- Mail access protocol: retrieval of mails from mail server
  - POP3: Post Office Protocol (RFC 1939)
    - Authentication between user agent and server
    - Download mails from mail box
    - Not secure
  - IMAP: Internet Mail Access Protocol (RFC 1730)
    - More features (more complex)
    - Manipulation of stored mails on server
    - Secure
  - HTTPS: Gmail, Hotmail, Yahoo! Mail, etc.

POP3 vs. IMAP

- POP3
  - POP3 by default uses "download and delete" mode
  - Alice cannot read e-mail if she changes to another computer
  - Download-and-keep mode: keep a copy of mails on the mail server and different computers
  - POP3 is stateless across sessions
- IMAP
  - Keep all messages in one place: the mail server
  - Allows user to organize messages in folders
  - IMAP keeps user state across sessions:
    - Names of folders and mappings between mail IDs and folder name

open source email server products

- Six free or open source email server products:
  - Apache James
    - Both Linux and Windows platforms.
  - Citadel
    - Linux-only
  - hMailServer
    - Windows
  - SmarterMail
    - Windows; marketed as an 'Exchange-level' mail server
    - Limited to 10 users
  - Zarafa
    - Linux-only
  - Axigen
    - Both Linux and Windows platforms
    - A free version for up to 100 users