

## COMP 431

### Internet Services & Protocols

# Application-Layer Protocols: The Web & HTTP

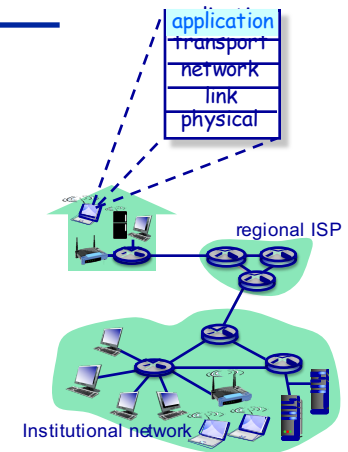
*Jasleen Kaur*

January 28, 2020

1

## Application-Layer Protocols Outline

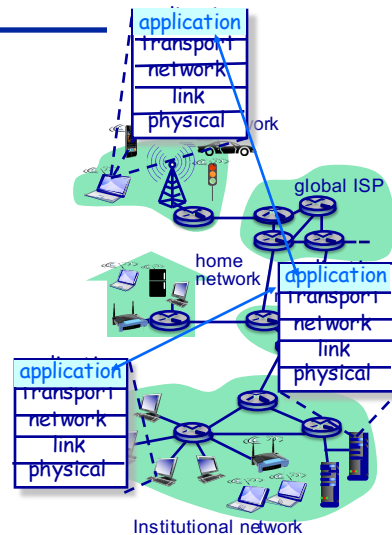
- ◆ Example client/server systems and their application-level protocols:
  - » The World-Wide Web (HTTP)
  - » Reliable file transfer (FTP)
  - » E-mail (SMTP & POP)
  - » Internet Domain Name System (DNS)
- ◆ Example p2p applications systems:
  - » BitTorrent
- ◆ Other protocols and systems:
  - » Streaming media — DASH
  - » Content delivery networks (CDNs)



2

## Applications and Application-Layer Protocols Overview

- ◆ Applications:
  - » Communicating, distributed processes
  - » Running in network hosts in “user space”
  - » Exchange messages
- ◆ Application-layer protocols
  - » One “piece” of an application
  - » Defines messages exchanged and actions taken
  - » Uses services provided by lower layer protocols



3

## Application-Layer Protocols The Web

- ◆ User agent (client) for the Web is called a **browser**:
  - » MS Internet Explorer
  - » Google Chrome
  - » Apple Safari
  - » Mozilla Firefox
- ◆ Server for the Web is called a **Web server**:
  - » Apache (public domain)
  - » MS Internet Information Server (IIS)



4

# Application-Layer Protocols

## Web Terminology

- ◆ Web page:
  - » Addressed by a **URL**
  - » Consists of “objects”

- ◆ Most Web pages consist of:
  - » Base HTML page
  - » Embedded objects



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html lang="en">
<head>
  <meta http-equiv="content-type" content="text/html; charset=iso-8859-1">
  <title>CNN.com</title>
  <meta http-equiv="refresh" content="1800; URL=http://www.cnn.com/?">
  <link rel="stylesheet" href="http://i.cnn.net/cnn/virtual/2001/style/main.css" type="text/css">
  <script language="JavaScript" src="http://i.cnn.net/cnn/virtual/2000/code/main.js"
    type="text/javascript"> </script>
  <script language="JavaScript" type="text/javascript"> </script>
  <script language="JavaScript" src="http://ar.atwola.com/file/adsWrapper.js"> </script>
  <style type="text/css"> </style>
  <script language="JavaScript"> </script>
</head>

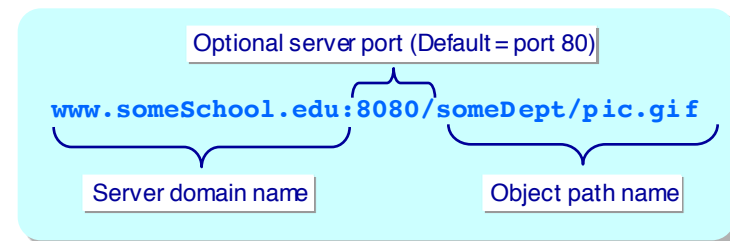
<body class="cnnMainBody" bgcolor="#FFFFFF">

<!--
  -->
```

5

# Web Terminology

## URLs (Universal Resource Locators)



- ◆ Each object on the web is addressable by a URL
  - » RFC 1738 & 3986
- ◆ URL components
  - » Server address
  - » (Optional port number)
  - » Path name

6

## Web Terminology

### The Hypertext Transfer Protocol (HTTP)

- ◆ Web's application layer protocol

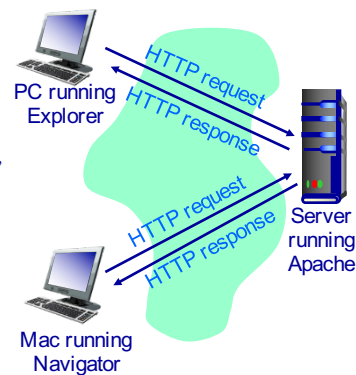
- ◆ Client/server model

- » *client*:

- ❖ browser that requests, receives, "displays" Web objects

- » *server*:

- ❖ Web server sends objects in response to requests



7

## The Hypertext Transfer Protocol

### HTTP Overview

- ◆ HTTP uses TCP sockets
  - » Browser initiates TCP connection to server (on port 80)
- ◆ HTTP messages (application-layer protocol messages) exchanged between browser and Web server
- ◆ HTTP/1.0: RFC 1945
  - » One request/response interaction per connection
- ◆ HTTP/1.1: RFC 2616
  - » Persistent connections
  - » Pipelined connections
- ◆ HTTP/2.0: RFC 7540
  - » Pipelined, encrypted by default

- ◆ HTTP is "stateless"
  - » Server maintains no information about past browser requests

— aside —

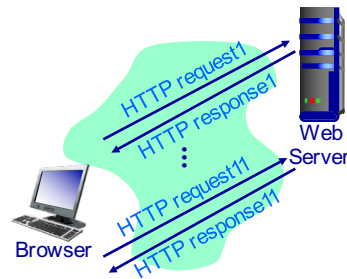
- ◆ Protocols that maintain "state" are complex!
  - » Past history (state) must be maintained
  - » If server or client crashes, their views of "state" may be inconsistent and must be reconciled

8

## The Hypertext Transfer Protocol

### HTTP example

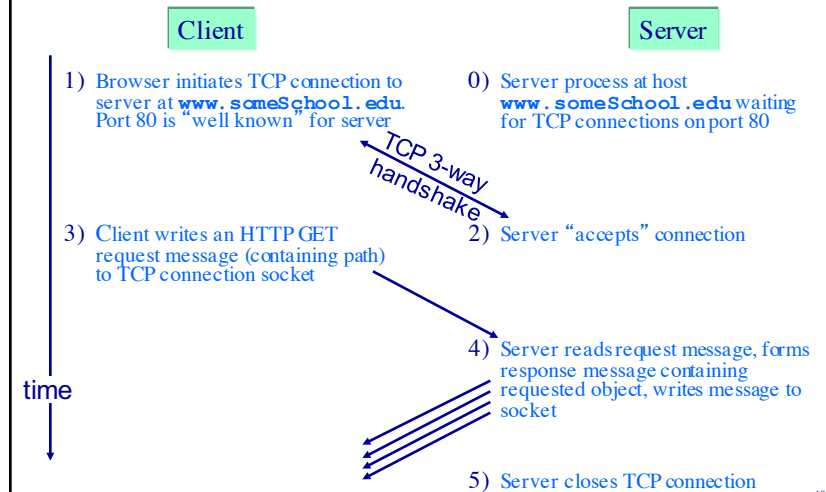
- ◆ User enters URL `www.someSchool.edu/someDept/home.index`
  - » Referenced object contains HTML text and references 10 JPEG images
- ◆ Browser sends an HTTP “GET” request to the server `www.someSchool.edu`
- ◆ Server will retrieve and send the HTML file
- ◆ Browser will read the file and sequentially make 10 separate requests for the embedded JPEG images



9

## HTTP 1.0 Example

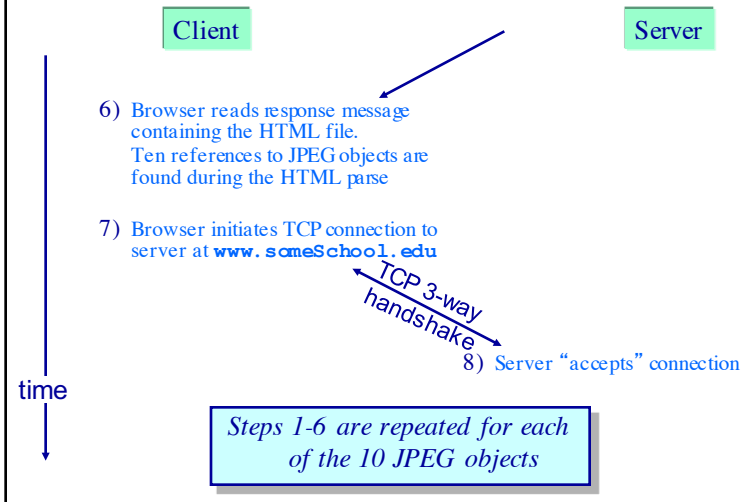
URL `www.someSchool.edu/someDept/home.index`



10

## HTTP 1.0 Example

URL `www.someschool.edu/someDept/home.index`



11

## The Hypertext Transfer Protocol

### HTTP message format

- ◆ Two types of HTTP message formats: *request* and *response* messages
  - » ASCII (human-readable format)

- ◆ HTTP request message:

- » Request line

method <SP> path <SP> version <CR><LF>

- » Optional header lines

headerfield name ":" value <CR><LF>

⋮

headerfield name ":" value <CR><LF>

<CR><LF>

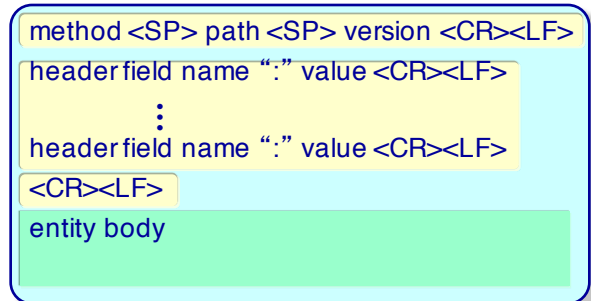
- » Present only for some methods (e.g., POST)

entity body

12

## The Hypertext Transfer Protocol

### HTTP request message format



- ◆ HTTP 1.0 Method types:
  - » GET
  - » POST — Form input sent in entity body
  - » HEAD — Just headers, no object
- ◆ HTTP/1.1 adds
  - » PUT — Uploads in entity body to path specified in URL field
  - » DELETE — Delete file specified in URL field

13

## HTTP Message Format

### Mozilla & MS Explorer request examples

```

GET /~jasleen HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.74 [en] (WinNT; U)
Host: dove.cs.unc.edu:80
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
        image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Cookie: SITESEVER=ID=8a064b7855a043146e45991174a3d970
  
```

- ◆ How does Mozilla process:
  - http://dove.cs.unc.edu:80/~jasleen ??*
- ◆ List of standard header fields:
  - [https://en.wikipedia.org/wiki/List\\_of\\_HTTP\\_header\\_fields](https://en.wikipedia.org/wiki/List_of_HTTP_header_fields)
- ◆ Chrome -> Inspect -> Network -> Headers

14

Different browsers display information in different ways

port is optional in 1.0 and required in 1.1

## HTTP Message Format

### Mozilla & MS Explorer request examples

```
GET /~jasleen HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.74 [en] (WinNT; U)
Host: dove.cs.unc.edu:80
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
        image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Cookie: SITESEVER=ID=8a064b7855a043146e45991174a3d970
```

```
GET /~jasleen HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
        application/msword, application/vnd.ms-excel,
        application/vnd.ms-powerpoint, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 4.0)
Host: dove.cs.unc.edu:80
Connection: Keep-Alive
```

## HTTP Message Format

### General response message format

#### ◆ Response messages

- » ASCII (human-readable format)

#### ◆ Message structure:

- » Response line
- » Optional header lines
- » Requested object, error message, etc.

```
version <SP> code <SP> phrase <CR><LF>
header field name ":" value <CR><LF>
:
header field name ":" value <CR><LF>
<CR><LF>
entity body
```



## HTTP Message Format

### HTTP response status codes

#### ◆ Sample response 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:)

**400 Bad Request**

» Request message not understood by server

**404 Not Found**

» Requested document not found on this server

**505 HTTP Version Not Supported**

17

## HTTP Message Format

### Typical Response Headers

Request  
headers

```
Connection: Keep-Alive
User-Agent: Mozilla/4.74 [en] (WinNT; U)
Host: dove.cs.unc.edu:80
Accept: image/gif, image/x-xbitmap, image/jpeg,
        image/pjpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Cookie: SITESEVER=ID=8a064b785a043146e4599174a3d970
```

Response  
headers

```
Date: Fri, 02 Feb 2009 19:10:11 GMT
Server: Apache/1.3.9 (Unix) (Red Hat/Linux)
Last-Modified: Tue, 30 Jan 2009 21:48:14 GMT
ETag: "1807135e-67-3a77369e"
Accept-Ranges: bytes
Content-Length: 103
Connection: close
Content-Type: text/plain
```

18

## HTTP Message Format

### Telnet example

Connect to HTTP server port → > telnet www.cs.unc.edu 80  
Telnet output { Trying 152.2.131.240...  
Connected to rock.cs.unc.edu.  
Escape character is '^['.  
Type GET command plus blank line { GET /~jasleen/foo.txt HTTP/1.0  
HTTP response status line → HTTP/1.1 200 OK  
HTTP response headers plus blank line { Date: Tue, 12 Feb 2013 01:43:25 GMT  
Server: Apache/2.2.15 (Red Hat)  
Last-Modified: Tue, 12 Feb 2013 01:43:02 GMT  
ETag: "44d2e7a0-6e-4d57d25675580"  
Accept-Ranges: bytes  
Content-Length: 110  
Connection: close  
Content-Type: text/plain; charset=UTF-8  
Object content { \*\* This test file is stored in the UNIX  
\*\* file system at  
\*\* /afs/cs.unc.edu/home/jasleen/public\_html/foo.txt  
Telnet output → Connection closed by foreign host.

19

## HTTP Message Format

### Telnet example (2)

Connect to HTTP server port → > telnet www.msn.com 80  
Telnet output { Trying 207.46.179.134...  
Connected to www.msn.com.  
Escape character is '^['.  
Type GET command plus blank line { GET /~index.html HTTP/1.0  
HTTP response status line → HTTP/1.1 404 Object Not Found  
HTTP response headers plus blank line { Server: Microsoft-IIS/5.0  
Date: Mon, 11 Feb 2002 18:33:15 GMT  
Content-Length: 1638  
Content-Type: text/html  
Object content { <HTML> <HEAD> . . .  
. . . .  
Error type 404 - Object Not Found  
</body> </html>  
Telnet output → Connection closed by foreign host.

20