

HTTP; The World Wide Web Protocol

HTTP
Web Content
Caching



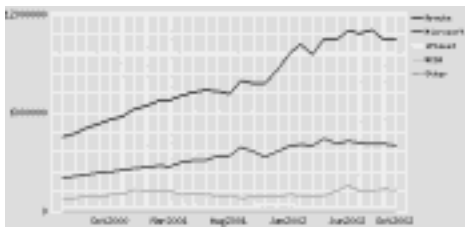
Marc Andreessen

Brief History of HTTP

- 1990-1993: The idea of a web "browser" is contemplated – poor interfaces hinder browser use
- 1993: Marc Andreessen (then a grad student at NSCA) posts Mosaic on an ftp cite. New features include:
 - Hyperlinks
 - Embedded images
- December 1993: Mosaic growth makes the front page of New York Times
- 1994: Marc Andreessen and colleagues leave NSCA to form Mosaic Corp. (now Netscape)

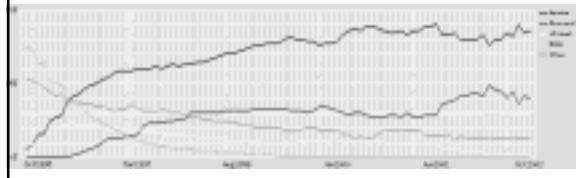
HTTP Growth

- Number of server sites increases rapidly



Web Server Statistics

- Apache is the most popular web server today (freely available)
- Microsoft IIS is second



Versions of HTTP

- 80% of Internet flows are HTTP connections
- Early protocol is HTTP 0.9
 - read only
- Today we use
 - HTTP 1.0
 - read, input, delete, ...
 - HTTP 1.1
 - performance optimizations

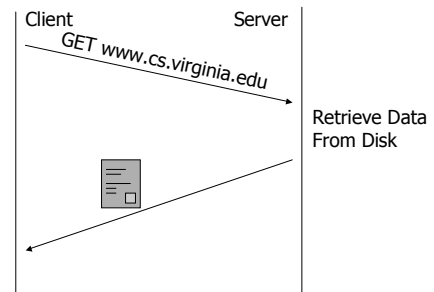
HTTP Overview

- Client (browser) sends HTTP request to server
- Request specifies affected *URL*
- Request specifies desired *operation*
- Server performs *operation* on *URL*
- Server sends response
- Request and reply headers are in pure text

Static Content and HTML

- Most static web content is written in HTML
- HTML allows
 - Text formatting commands
 - Embedded objects
 - Links to other objects
- Server need not understand or interpret HTML

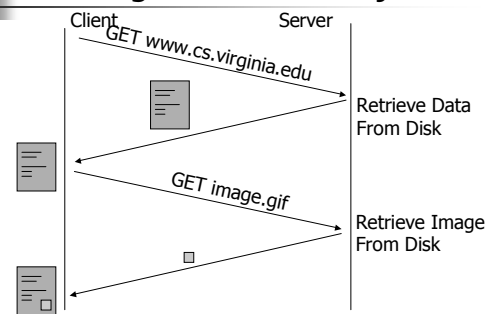
Example of an HTTP Exchange



Fetching Multiple Objects

- Most web-pages contain embedded objects (e.g., images, backgrounds, etc)
- Browser requests HTML page
- Server sends HTML file
- Browser parses file and requests embedded objects
- Server sends requested objects

Fetching Embedded Objects



HTTP Operations

- GET: retrieves URL (most widely used)
- HEAD: retrieves only response header
- POST: posts data to server
- PUT: puts page on server
- DELETE: deletes page from server

Simple HTTP Request and Reply

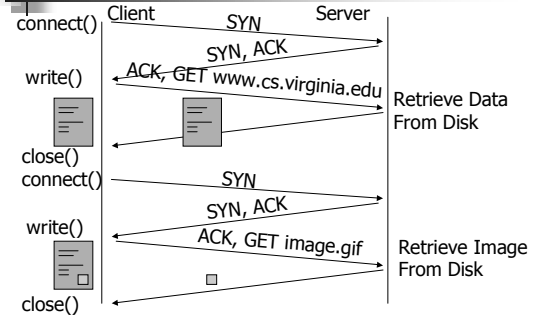
Request:
GET http://www.server.com/page.html HTTP/1.0

Response:
HTTP-Version: HTTP/1.0 200 OK
Content-Length: 3012
Content-Type: text/html
<body>

HTTP 1.0

- Client opens a separate TCP connection for each requested object
- Object is served and connection is closed
- Advantages
 - maximum concurrency
- Limitations
 - TCP connection setup/tear-down overhead
 - TCP slow start overhead

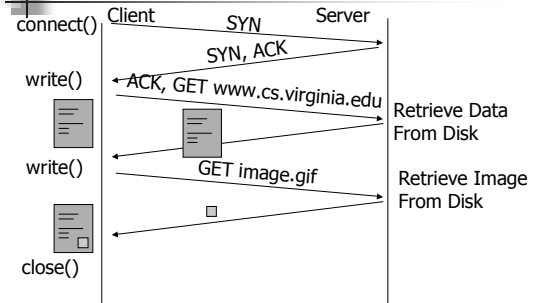
HTTP 1.0



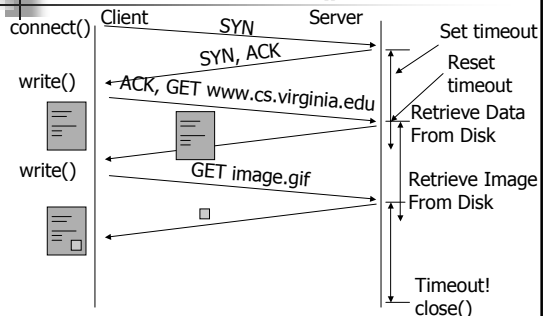
HTTP 1.1

- To avoid a connection per object model, HTTP 1.1 supports *persistent connections*
- Client opens TCP connection to server
- All requests use same connection
- Problems
 - Less concurrency
 - Server does not know when to close idle connections

HTTP 1.1



Server Side Close()



Example 1 (HTTP 1.0)

- Try the following exercise:
 - Login to a UNIX account
 - From your UNIX prompt type:


```
telnet www.cs.virginia.edu 80
```
 - Hit enter, then on the next blank line type:


```
GET http://www.cs.virginia.edu/index.html HTTP/1.0
```
 - Hit enter twice.

Example 2 (HTTP 1.1)

- Try the following exercise:
 - Login to a UNIX account
 - From your UNIX prompt type:
telnet www.cs.virginia.edu 80
 - Hit enter, then on the next blank line type:
GET <http://www.cs.virginia.edu/index.html> HTTP/1.1
Host: <http://www.cs.virginia.edu>
 - Hit enter twice.
- What was the HTTP timeout on the server?

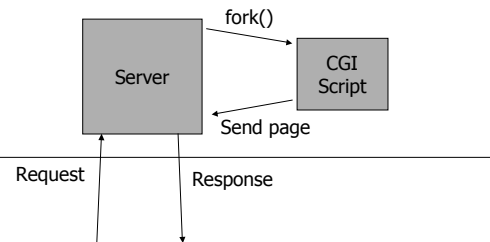
Dynamic Content

- Web pages can be created as requests arrive
- Advantages
 - Personalization (e.g., my.yahoo.com),
 - interaction with client input
 - interaction with back-end applications
- Disadvantages
 - Performance penalty
 - Generating dynamic content (CGIs, ASPs)

CGI Scripts

- CGI scripts are URLs with a .cgi extension
- The script is a program (e.g., C, JAVA, ...)
- When the URL is requested, server invokes the named script, passing to it client info
- Script outputs HTML page to standard output (redirected to server)
- Server sends page to client

CGI Execution



Fast CGI

- CGI scripts are invoked on each request
- Invocation overhead is large (e.g., UNIX `fork()`)
- Fast CGIs are invoked *a priori* and blocked on server I/O
- Client requests unblock script
- Script outputs page and blocks again

Active Server Pages (ASPs)

- Active server pages are HTML documents with extensions for embedded program execution
- When request arrives, server fetches and parses the HTML document
- Server executes embedded executable code and plugs output into page
- Expanded page is sent to client