EE 122: HyperText Transfer Protocol (HTTP)

Ion Stoica
Nov 25, 2002

Background

- World Wide Web (WWW): a set of cooperating clients and servers that communicate through HTTP
- HTTP history
  - First HTTP implementation - 1990
    • Tim Berners-Lee at CERN
  - HTTP/0.9 – 1991
    • Simple GET command for the Web
  - HTTP/1.0 – 1992
    • Client/Server information, simple caching
  - HTTP/1.1 - 1996

Basics

- Client-server architecture
- Synchronous request/reply protocol
- Stateless
- Uses unicast
- Implemented on top of TCP/IP

Terminology

- Resource – file or service (e.g., dynamic results from the execution of a script)
- Entity – information transferred in a request or response
- Entity Tag – unique identifier for a resource
Universal Resource Locator

- An address or location of a resource, e.g., http://www.eecs.berkeley.edu/index.html
- Prefix up to ":" represents the protocol to be used to obtain the resource

Client Request

- Steps to get the resource: http://www.eecs.berkeley.edu/index.html
  1. Use DNS to obtain the IP address of www.eecs.berkeley.edu, A
  2. Send to A an HTTP request:
     GET /index.html HTTP/1.0
  3. Server response: (see next slide)

Server Response

HTTP/1.0 200 OK
Content-Type: text/html
Content-Length: 1234
Last-Modified: Mon, 19 Nov 2001 15:31:20 GMT
<TITLE>EECS Home Page</TITLE>
</HEAD>
...</BODY>
</HTML>

Big Picture

Client
Establish connection
TCP Syn
TCP syn + ack
Client request
TCP Ack + HTTP GET
Request response
Close connection

Server
### Request Methods

- **GET** – transfer resource from given URL
- **HEAD** – GET resource metadata (headers) only
- **PUT** – store/modify resource under the given URL
- **DELETE** – remove resource
- **POST** – provide input for a process identified by the given URL (usually used to post CGI parameters)

### Response Codes

- 1x informational
- 2x success
- 3x redirection
- 4x client error in request
- 5x server error, can’t satisfy the request

### HTTP/1.0 Example

```
Client          Server

Request image 1
  Transfer image 1
Request image 2
  Transfer image 2
Request text
  Transfer HTML

Finish display page
```

### HHTP/1.0 Performance

- Create a new TCP connection for each resource
  - Large number of embedded objects in a web page
  - Many short lived connections
- TCP transfer
  - Too slow for small object
  - May never exit slow-start phase
Web Proxies

- Intermediaries between client and server

Client 1

Client 2

... Proxy

... Proxy

Server

Web Proxies (cont’d)

- Location: close to the server, client, or in the network
- Functions:
  - Filter requests/responses
  - Modify requests/responses
    - Change http requests to ftp requests
    - Change response content, e.g., transcoding to display data efficiently on a Palm Pilot
  - Provide better privacy
  - Caching

HTTP/1.0 Caching

- A request directive:
  - Pragma: no-cache – ignore all caches and get resource directly from server
- A modifier to the GET request:
  - If-modified-since – return a “not modified” response if resource was not modified since specified time
- A response header:
  - Expires – specify to the client for how long it is safe to cache the resource

HTTP/1.1

- Performance:
  - Persistent connections
  - Pipelined requests/responses
  - ...
- Support for virtual hosting
- Efficient caching support
### Persistent Connections

- Allow multiple transfers over one connection
- Avoid multiple TCP connection setups
- Avoid multiple TCP slow starts

### Pipelined Requests/Responses

- Buffer requests and responses to reduce the number of packets
- Multiple requests can be contained in one TCP segment
- Note: order of responses has to be maintained

### Support for Virtual Hosting

- Problem: recall that a request to get `http://www.foo.com/index.html` has in its header only:
  - GET /index.html HTTP/1.0
- It is not possible to run two web servers at the same IP address, because GET is ambiguous
  - This is useful when outsourcing web content, i.e., company "foo" asks company "outsourcer" to manage its content
- HTTP/1.1 addresses this problem by mandating "Host" header line, e.g.,
  ```
  GET /index.html HTTP/1.1
  Host: www.foo.com
  ```

### HTTP/1.1 - Caching

- HTTP/1.1 provides better support for caching
  - Separate "what to cache" and "whether a cache response can be used safely"
  - Allow server to provide more info on resource cacheability
  - A cache does not return unknowingly a stale resources
  - Not depending on absolute timestamps
HTTP/1.1 - Caching (cont’d)

- Four new headers associated to caching: age header, entity tags, cache-control, and vary
- Age Header – the amount of time that is known to have passed since the response message was retrieved
- Entity tags – unique tags to differentiate between different cached versions of the same resource

Cache-Control
- no-cache: get resource only from server
- only-if-cached: obtain resource only from cache
- no-store: don’t allow caches to store request/response
- max-age: response’s should be no grater than this value
- max-stale: expired response OK but not older than staled value
- min-fresh: response should remain fresh for at least staled value
- no-transform: proxy should not change media type

Vary
- Accommodate multiple representations of the same resource
- Used to list a set of request headers to be used to select the appropriate representation

Example:
- Server sends the following response
  HTTP/1.1 200 OK
  Vary: Accept-Language

- Request will contain:
  Accept-Language: en-us
- Cache return the response that has:
  Accept-Language: en-us

HTTP/1.1 – Caching (cont’d)

Summary

- HTTP the backbone of WWW
- Evolution of HTTP has concentrated on increasing the performance
- Next generations (HTTP/NG) concentrate on increasing extensibility