Building a Simple Web Proxy

COS 461 Assignment 1
A Brief History of HTTP

- Mar 1989 - "Information Management: A Proposal"
- Oct 1990 - "WorldWideWeb" coined
- Oct 1994 - W3C founded
- May 1996 - RFC 1945 (HTTP 1.0)
- June 1999 - RFC 2616 (HTTP 1.1)
Anatomy of HTTP 1.0

Web Client: Firefox

Connect: Request
GET / HTTP/1.0
Host: www.yahoo.com
CRLF

Web Server:

Response: Close
HTTP/1.0 200 OK
Date: Tue, 16 Feb 2010 19:21:24 GMT
Content-Type: text/html;
CRLF
<html><head><title>Yahoo!</title>
Anatomy of HTTP 1.0

Web Client: Request Line, Request Header, Request Delimiter

Connect: Request:
- GET / HTTP/1.0
- Host: www.yahoo.com
- CRLF

Web Server: Response Status, Response Header, Response Delimiter, Response Body

Response: Close:
- HTTP/1.0 200 OK
- Date: Tue, 16 Feb 2010 19:21:24 GMT
- Content-Type: text/html;
- CRLF
- <html><head><title>Yahoo!</title></html>
HTTP 1.1 vs 1.0

- Additional Methods (PUT, DELETE, TRACE, CONNECT + GET, HEAD, POST)
- Additional Headers
- Transfer Coding (chunk encoding)
- Persistent Connections (content-length matters)
- Request Pipelining
Why Use a Proxy?

- Caching
- Content Filtering
- Privacy
Building a Simple Web Proxy

- Forward client requests to the remote server and return response to the client
- Handle HTTP 1.0 (GET)
- Multi-process, non-caching web proxy
- ./proxy <port>
Handling Requests

• What you need from a client request: host, port, and URI path
  GET http://www.princeton.edu:80/ HTTP/1.0

• What you send to a remote server:
  GET / HTTP/1.0
  Host: www.princeton.edu:80
  Connection: close

Check request line and header format
Handling Responses

Web Client

Parse Request: Host, Port, Path

Simple Proxy

Forward Response to Client Including Errors

Web Server
Handling Errors

- Method != GET: Not Implemented (501)
- Unparseable request: Bad Request (400)
- Parse using the Parsing library
- Postel’s law: Be liberal in what you accept, and conservative in what you send
  convert HTTP 1.1 request to HTTP 1.0
  convert \r to \r
  etc...
Testing Your Proxy

- Telnet to your proxy and issue a request
  
  ```
  > ./proxy 5000
  > telnet localhost 5000
  Trying 127.0.0.1...
  Connected to localhost.localdomain (127.0.0.1).
  Escape character is '^]'.
  GET http://www.google.com/ HTTP/1.0
  ```
  
  (HTTP response...)

- Direct your browser to use your proxy

- Use the supplied proxy_tester.py and proxy_tester_conc.py
Proxy Guidance

• Assignment page
• Assignment FAQ
• RFC 1945 (HTTP 1.0)
• Google, wikipedia, man pages
• Must build on Friend 010 machines