Project 2 Discussion

Ming Liu
Project Overview

• In this assignment, you will implement a HTTP proxy that passes requests and data between multiple web clients and web servers, concurrently.
  ✓ Capable of both relaying HTTP requests and HTTP CONNECT tunneling
  ✓ For non-CONNECT HTTP requests, you'll slightly edit the HTTP request header and communicate between the web browser and the origin server
  ✓ For CONNECT HTTP requests, you'll establish a TCP connection between the browser and the remote server then any data can be passed through
  ✓ Capable of handling the traffic caused by real user browsing
HTTP

• The Hypertext Transfer Protocol (HTTP) is the protocol used for communication on this web
• It defines how your web browser requests resources from a web server and how the server responds
  ✓ Version 1.0 of the HTTP protocol
  ✓ RFC 1945
• HTTP communications happen in the form of transactions
  ✓ A client sending a request to a server and then reading the response
HTTP request and response message format

• An initial line (a request or response line, as defined below)
• Zero or more header lines
• A blank line (CRLF)
  ✓ The initial line and header lines are each followed by a "carriage-return line-feed" (\r\n) signifying the end-of-line
• An optional message body
HTTP protocol

1. User issues URL from a browser: http://host:port/path/file
2. Browser sends a request message:
   GET URL HTTP/1.1
   Host: host:port
3. Server maps the URL to a file or program under the document directory.
4. Server returns a response message:
   HTTP/1.1 200 OK

Client (Browser)  HTTP (Over TCP/IP)  Server (@ host:port)
Demo

- **telnet** [www.washington.edu](http://www.washington.edu) 80
  - Telnet is an application layer protocol

```
Trying 128.95.155.135...
Escape character is '^]'.
GET http://www.washington.edu/ HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 29 Oct 2015 03:30:39 GMT
Server: Apache/2.2.24 (Unix) mod_ssl/2.2.24 OpenSSL/1.0.1e-fips PHP/5.6.11 mod_pubcookie/3.4a mod_uwa/3.2.1
ETag: "185bf8-a679-523329acc000"
Accept-Ranges: bytes
Content-Length: 42617
Connection: close
Content-Type: text/html
```
HTTP Proxy

- Proxies are an example of the use of "interposition" - placing something between two things that communicate using a well-defined interface.

- Monitoring or debugging (by capturing a log of browser requests and server responses)
- Improve performance by maintaining a cache of web pages
- Enforce some policy about which sites can be accessed.
More proxy example: FB Memcached
Assignments Details 1

- Fetch the request page from the origin web server and return it to the browser
- Print out the first line of each HTTP request
- Demo

```plaintext
GET http://www.cnn.com/ HTTP/1.1
Host: www.cnn.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:40.0) Gecko/20100101 Firefox/40.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
```
Assignments Details 2

• Determine the web server’s address
  ✓ Parse the header and find the Host line

• Turn off keep-alive
  ✓ keep-alive → close

• Change HTTP version
  ✓ Lower the version of the request to HTTP 1.0

• HTTP CONNECT tunneling
  ✓ A two-hop TCP connection between the client and some server via Proxy
  ✓ Forward to the server any bytes it receives after the request header on its connection with the client
  ✓ Forward to the client any bytes it receives on its connection with the server
Restrictions

• TCP socket
• Any method
• Any language
• run script
• Testing
  ✓ Configuring the Firefox
  ✓ Check sample outputs → Don’t need to be exactly the same.
• Demo