National University of Singapore
School of Computing
CS2105: Introduction to Computer Networks
Semester 1, 2015/2016

Assignment 1
HTTP Proxy

Release date: 31 August 2015
Due: 25 September 2015, 23:59

Overview

In this assignment, you will implement a simple HTTP/1.0 Web proxy server that passes data between a web client and a web server. This exercise will give you a chance to learn more about the most popular protocol on the Internet: the Hyper-text Transfer Protocol (HTTP).

HTTP Proxies

Ordinarily, HTTP is a client-server protocol. The client (usually your web browser) communicates directly with the server (the web server software). However, in some circumstances it may be useful to introduce an intermediate entity called a proxy. Conceptually, the proxy sits between the client and the server. In the simplest case, instead of sending requests directly to the server the client sends all its requests to the proxy. The proxy then opens a connection to the server, and passes on the client's request. The proxy receives the reply from the server, and then sends that reply back to the client. Notice that the proxy is essentially acting like both a HTTP client (to the remote server) and a HTTP server (to the initial client).

Why use a proxy? There are a few possible reasons:

- **Performance**: By saving a copy of the pages that it fetches, a proxy can reduce the need to create connections to remote servers. This can reduce the overall delay involved in retrieving a page, particularly if a server is remote or under heavy load. Moreover, the system performance could be significantly enhanced via using caching proxies. Caching proxies keep local copies of frequently requested resources, allowing large organizations to significantly reduce their upstream bandwidth usage and costs, while significantly increasing performance.
- **Content Filtering and Transformation**: While in the simplest case the proxy merely fetches a resource without inspecting it, there is nothing that says that a proxy is limited to blindly fetching and serving files. The proxy can inspect the requested URL and selectively block access to certain domains, reformat web pages (for instances, by stripping out images to make a page easier to display on a hand-held or other limited-resource client), or perform other transformations and filtering.
- **Privacy**: Normally, web servers log all incoming requests for resources. This information typically includes at least the IP address of the client, the browser or other client program that they are using (called the User-Agent), the date and time, and the requested file. If a

client does not wish to have this personally identifiable information recorded, routing HTTP requests through a proxy is one solution. All requests coming from clients using the same proxy appear to come from the IP address and User-Agent of the proxy itself, rather than the individual clients. If a number of clients use the same proxy (say, an entire business or university), it becomes much harder to link a particular HTTP transaction to a single computer or individual.

Submission

You will submit your program on IVLE. Zip your files into a single zip file and name it "`a1-<student number>.zip`" and upload to the Assignment 1 folder. For example, if your student number is A0123456X, then you should name your file "`a1-a0123456x.zip`".

Your zip file should include a java file named `WebProxy.java` which contains the class `WebProxy`. You can include other necessary java files in the zip file.

We will unzip the contents of your zip file and compile your submission using `javac WebProxy.java` and run it with `java WebProxy <port>` , where `port` will be the port number that your proxy should listen for incoming TCP connections.

We will use `curl` to test your proxy and compare the output to our test data. Note that just because the web page appears the same in Firefox does not mean it is exactly identical, e.g., the headers or whitespaces might be different.

The dateline for submission is at 25 September 2015, 23:59 hrs. Late submission will be penalized 10% per day.

Submitting in Python

Like assignment 0, you may write your proxy in Python. Your zip file should contain the file `WebProxy.py` and we will run it with `python Webproxy.py <port>`.

Submitting in C++

Like assignment 0, you may write your proxy in C++. Your zip file should contain the file `WebProxy.cpp` and we will compile it with `g++ WebProxy.cpp -o WebProxy` and then execute it with `./WebProxy <port>`.

Testing

You can test your proxy server either directly using `curl` or using it "for real" with your web browser. We recommend using Firefox as the settings only affect Firefox, while Chrome and Internet Explorer changes rely on a system-wide setting.

Configuring Firefox

There are two steps you need to do.

1. Set the location of the proxy

1. Select Options from the menu button.
2. Go to the Network tab, and select Settings.
3. Select "Manual proxy configurations" and fill in the hostname of your HTTP proxy and the port. If your proxy is on the same machine, you can use "localhost" as the hostname.

2. Set Firefox to use HTTP/1.0

Because Firefox defaults to using HTTP/1.1 and your proxy speaks HTTP/1.0, there are a couple of minor changes that need to be made to Firefox's configuration. Fortunately, Firefox is smart enough to know when it is connecting through a proxy, and has a few special configuration keys that can be used to tweak the browser's behavior.

1. Enter "`about:config`" in the URL address field.
2. In the Search bar, enter "`network.http.proxy`"
3. Set `network.http.proxy.version` to `1.0`, and `network.http.proxy.pipelining` to `false`.

Details and Grading

The score for this assignment is 13 marks. If you obtain above 13 marks, half of the extra marks can be passed-on to other assignments.

Basic criterias:

- Your proxy can handle small web objects like a small file or image. **(2 marks)**
- Your proxy can handle a complex web page with multiple objects, e.g., `www.comp.nus.edu.sg`. **(2 marks)**
- Your proxy can handle very large files of up to 1 GB. **(2 marks)**
- Your proxy should be able to handle erroneous requests such as a "404" response. It should also return a "502" response if cannot reach the web server. **(2 marks)**
- Your proxy can also handle the POST method in addition to GET method, and will correctly include the request body sent in the POST-request. **(2 marks)**

Advanced criterias:

- Simple caching. A typical proxy server will cache the web pages each time the client makes a particular request for the first time. The basic functionality of caching works as follows:

  When the proxy gets a request, it checks if the requested object is cached, and if it is, it simply returns the object from the cache, without contacting the server. If the object is not cached, the proxy retrieves the object from the server, returns it to the client and caches a copy for future requests.

  Your implementation will need to be able to write responses to the disk (i.e., the cache) and fetch them from the disk when you get a cache hit. For this you need to implement some internal data structure in the proxy to keep track of which objects are cached and where they are on the disk. You can keep this data structure in main memory; there is no need to make it persist across shutdowns. **(2 marks)**

- Advanced caching. Your proxy server must verify that the cached objects are still valid and that they are the correct responses to the client's requests. Your proxy can send a request to the origin server with a header `If-Modified-Since` and the server will response with a HTTP 304 - Not Modified if the object has not changed. This is also known as a Conditional Request. **(2 marks)**
- Text censorship. A text file `censor.txt` containing a list of censored words is placed in the same directory as your WebProxy. Each line of the file contains one word. Your proxy should detect text or HTML files that are being transfered (from the `Content-type` header) and replace any censored words with 3 dashes "`--`". The matching word should be case insensitive. **(2 marks)**
- Multi-threading. Your proxy can concurrently handle multiple connections from several clients. **(2 marks)**