Simplified HTTP Retriever and Server

Start Assignment

Due Apr 25 by 11:59pm **Points** 100 **Submitting** a file upload **File Types** zip and tgz

You will write two programs that exercise a simplified version of HTTP. The retriever will work in conjunction with any web server and the server will work in conjunction with any web browser. This way, you can test your software independently of each other. You may use HTTP 1.0 or 1.1.

1) Your retriever takes in an input from the command line and parses the server address and file that is being requested. The program then issues a GET request to the server for the requested file. When the file is returned by the server, the retriever outputs the file to the screen and to the file system. If the server returns an error code instead of a OK code, then the retriever should not save the file and should display on the screen whatever error page was sent with the error. Your retriever should exit after receiving all the responses.

2) Your server waits for a connection and an HTTP GET request (you may do this single threaded or multi-threaded). After receiving the GET request, the server opens the file that is requested and sends it (along with the HTTP 200 OK code, of course). If the file requested does not exist, the server should return a 404 Not Found code along with a custom File Not Found page. If the HTTP request is for SecretFile.html then the web server should return a 401 Unauthorized. If the request is for a file that is above the directory structure where your web server is running (for example, "GET ../../../etc/passwd"), you should return a 403 Forbidden. Finally, if your server cannot understand the request, return a 400 Bad Request. After you handle the request, your server should return to waiting for the next request.

**for your demo script, you can create a list of "unauthorized files" and "forbidden files" and check that list rather than actually trying to determine if you don't have the permissions **

You will submit the code for both assignments with a build and demo script. The demo script should run through all of the following test cases. You will also submit screen shots of your programs executing all of the following test cases.

(screen shot only)

1) Real Web browser accessing your server

(screen shot and demo script)

2) Your retriever accessing a real server

3) Your retriever accessing a file from your server

4) Your retriever requesting an unauthorized file from your server

5) Your retriever requesting a forbidden file from your server

6) Your retriever requesting a non-existent file from your server

7) Your retriever sending a malformed request to your server

Make sure your screenshots are labeled.

For reference: **http://tools.ietf.org/html/rfc1945** (http://tools.ietf.org/html/rfc1945)