CS350 – Lab 20

Text Based Web Search

Have you ever loaded a search engine and wondered why the page was so heavy, slow, and filled with ads and images? Have you ever wanted to be able to just quickly search something and see the results from the command line? Then this is the lab for you!

Alhea (alhea.com) is an alternate search engine started since at least 2012, designed to give combined results from other search engies. Alhea has been running mostly unchanged for many years - their search engine still works great. We will use this as the basis for our text-based web search program.

Be respectful of Alhea's server resources and bandwidth. Do not do something foolish such as spamming their server with requests in a while loop, or you will ruin it for everybody. A properly written script should send one request per run. Think carefully before you execute your code.

1. Manually enter some searches in the Alhea search engine, and, as demoed in class, use your browser to view the HTTP traffic. Take note of what the URL looks like and what the raw headers are. Try to see if any parts of the URL can be removed while still getting the search results. *Be sure to remove the s from https before recording this information.*
2. Write a C program that opens an internet socket to Alhea's search server. Make sure the server name, port number, etc. are correct.
3. Based on the information you gathered in part 2, send over the appropriate HTTP request headers to the server to request the desired search page. Hardcode a search term for now (we will change this in step 6, but best to start small). You might want to change the "Accept Encoding" to "identity" so as not to have to worry about zip compression, etc. Note that webservers are picky about whitespace – for instance, they need windows-style newlines, and they need two newlines at the end.
4. Read the webpage into a large preallocated character buffer. You will need to perform several read() calls in a loop as the server sends over the information in chunks. It can be hard to tell when the server is done, so I recommend using a short (but not too short) socket timeout for this.
5. Copy-and-paste the function I wrote in the provided `printresults.c` to parse the HTML and print the titles of the search results. There should almost always be 7 or 10, and they should match the ones on the webpage (aside from the missing ads). *If nothing prints from this function, something is amiss. Check the error codes on the system calls*

as well as inspect the HTML that was returned by the server for clues as to what went wrong.

6. Modify your code to ask the user for a search term, relay that term in your Alhea search request, and print the search results for that term. Restricting it to just a single word is fine for this lab. Be careful to avoid buffer overflows, for instance by using fgets().
7. Why do you think I had you do this on Alhea instead of another search engine? In other words, what about other search engines would have prevented this simple socket approach?

Do NOT use any includes other than:

```
#include <sys/socket.h>
#include <netdb.h>
#include <string.h>
#include <stdio.h>
#include <unistd.h>
```

Submit your C code and answers to question 7 to Laulima.