1 Objectives

The objectives of this lab are:

1. Getting more practice with C programming in general.
2. Getting experience with network programming (with respect to a client).

2 Requirements

2.1 Big Picture

Create a simple command-line utility for downloading text-based files from a webserver. Much of the code is already provided for you, so you only need to focus on the networking code. Look for "TBD" in the code to see those places where you need to write your code.

2.2 Interface Requirements

Create a command-line file downloader that complies with the following requirements:

1. The name of the source file is "mywget.c". A file with this name is provided for you as your starting point.
2. The syntax for using mywget is:
   ```
   ./mywget servername filename
   ```
   where "servername" is the name of the webserver, and "filename" is the name of the file to download from the server.
3. Assuming the requested file exists on the webserver, it shall be downloaded via HTTP into a local file with the same name. In other words, if you issued the following command:
   ```
   ./mywget tor.ern.nps.edu index.html
   ```
   it should result in a local file called "index.html" containing the contents of the file on the server named "tor.ern.nps.edu".
4. If a local file with the same name already exists in the current working directory, then a meaningful error message shall be reported to the user, and the program shall terminate.
5. Assume that the maximum file name is 64 characters, and that the maximum webserver name is 64 characters.
6. You shall detect errors and display appropriate messages on `stderr`.

2.3 Other Requirements

1. Complete the program in the file provided.
2. You **must** use the `getaddrinfo` function to resolve DNS. In other words, you cannot use `gethostbyname` or other functions that are being obsoleted.

3 Notes

1. When testing on-campus, use the following command:
   ```
   ./mywget tor.ern.nps.edu index.html
   ```
2. To compare your resulting file with the one on the webserver, you can download the file with the real `wget` command:
   ```
   wget tor.ern.nps.edu/index.html
   ```

3. If working correctly, your program should work on any webserver if you know the name of a text-based file on that webserver. If the path to the filename contains a directory, such as:

   ```
   ./mywget serverpath.net data/README.txt
   ```

   then the code provided for you will create the file in the current working directory.
4. You are given a starting file with some helper functions already provided, and which also contains an "outline" of what needs to be done to finish the program.
   a. The supplied code will compile with warnings because a good chunk of the code is commented out using a big "#if 0" block. A good approach would be to move the "#if 0" line down a little and implement the required code above it. Compile your code and verify it is working before doing more.
   b. Some debugging code is also provided, surround by a "#ifdef DEBUG" statement. To enable these blocks of code, add the "-DDEBUG" flag when compiling.
5. Per the HTTP protocol, a client makes a request for a file using the following format:

   ```
   GET /filename HTTP/1.1\r\n
   User-Agent: mywget/0.1 (linux-gnu)\r\n
   Accept: text/html\r\n
   Host: servername\r\n
   Connection: Close\r\n
   \r\n
   ```

   Where "filename" is the name of the file you want to download, and "servername" is the DNS name of the webserver. Part of your job is to build this request and then send it to the webserver. This is commented as "Build my request" in mywget.c. This request is nothing more than a string (i.e., an array of `char`) that is filled in as shown above. For example, using the example request given in step 1 above, the request would look like the following:

   ```
   GET /index.html HTTP/1.1\r\n
   User-Agent: mywget/0.1 (linux-gnu)\r\n
   Accept: text/html\r\n
   Host: tor.ern.nps.edu\r\n
   Connection: Close\r\n
   \r\n
   ```
6. Library functions that may be useful to you:
   a. `strcpy`
   b. `strcat`
   c. `gai_strerror`
   d. `bzero`
7. For your information, the IP address of tor.ern.nps.edu is 172.20.108.14.
8. Your code will not work correctly on non-text files, such as pictures and movies.

4 Submission

Post mywget.c on Sakai by the due date

5 Grading

Your grade will be based on the following **guidelines**:

1. Compiles without errors (20).
2. Compiles without warnings (5).
3. Proper memory handling, such as `valgrind` reporting no errors, proper use of pointers, and no segmentation faults. (10).
4. Handles the following error conditions properly: (12)
   a. `./mywget`
   b. `./mywget foo`
   c. `./mywget foo bar baz`
   d. `./mywget nonexistentserver.net  index.html`
   e. `./mywget tor.ern.nps.edu nonexistentfile.txt`
   f. `touch index.html`
      `./mywget tor.ern.nps.edu index.html`
5. The program acts as specified in Section 2:
   a. The downloaded file matches the original on the webserver. (20)
   b. The program works properly on files bigger than the input buffer. (10)
   c. Error messages go to stderr. (5)
   d. No leftover debugging statements displaying on the screen. (3)
6. Code review:
   a. `getaddrinfo` was used to resolve DNS. (10)
   b. No style-guide violations. (5)
7. I reserve the right to deduct points as I see fit.