Carnegie Coding Check: Multi-GET

Target: Build an application that downloads part of a file from a web server, in chunks.

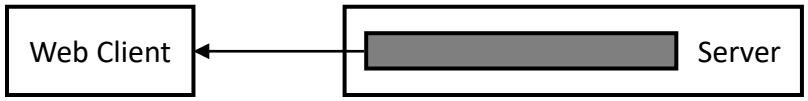
Context: This is a simplified version of a "download booster", which speeds up downloads by requesting files in multiple pieces simultaneously (saturating the network), then reassembling the pieces.

Why: This is a simple application that has some coding and some networking, but both parts are straightforward. Even if you are not familiar with networking, the APIs and concepts are pretty simple so you should be able to complete this in a few hours.

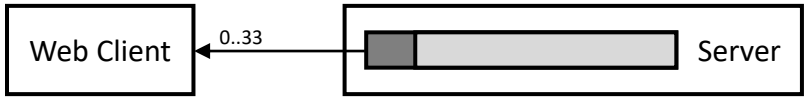
What we'll be looking at: Whether you met all the requirements, how you solved the problem, how clear your code is, how you chose to structure your program, how easy your program is to use, and how you communicate the design decisions you made (in code/comments).

Details

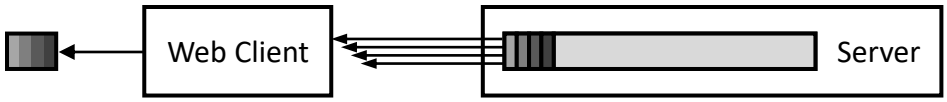
When downloading a file, web clients (e.g. browser, curl) create a connection to the server, request the file, and then receive the whole thing:



If the web client wants only part of the file, it can use the HTTP 'Range' header to do this:



Your app should request the first 4 MiB of the file in 1 MiB chunks, and write the result to disk:



Sample Client

We've built a sample client to show you how we might solve this problem. You don't have to solve it the same way as us, as long as you meet the requirements.

Note: This version supports some of the optional features that you don't need to support

Download link: http://dist.pravala.com/coding/multiGet-example.zip

Requirements

Your app should meet all of the following requirements:

- Source URL should be specified with a required command-line option
- File is downloaded in 4 chunks (4 requests made to the server)
- Only the first 4 MiB of the file should be downloaded from the server
- Output file may be specified with a command-line option (but may have a default if not set)
- No corruption in file - correct order, correct size, correct bytes
  (you don't need to verify correctness, but it should not introduce errors)
- File is retrieved with GET requests

Optional features

If you finish quickly, you *may* improve your app by adding features like:

- File is downloaded in parallel rather than serial
- Support files smaller than 4 MiB (less chunks/adjust chunk size)
- Configurable number of chunks/chunk size/total download size

Out of scope

Don't go out of your way to support these things:

- HTTPS
- Server doesn't support range request, or serves it incorrectly
- Other HTTP methods (PUT/POST/DELETE)
- Re-use existing connections with HTTP keep-alive

Language

We will have specified the language to use when writing the application in our email to you, based on the languages we use in the target position. Generally:

- **Network services:** C++
- **Web services:** TypeScript or JavaScript
- **Mobile Apps:** C++, Java, or Objective-C

Additional notes

- You may use any system or open-source libraries you need to perform this task
- You can send us your code by email, or through a link to a GitHub/BitBucket/etc. repo
- We should be able to compile your code – we prefer Ubuntu (gcc/clang), but also have Windows (with Visual Studio), and macOS (with Xcode)
- The file we'll be using to test is in our email to you; you should make sure your program works on this file before sending your solution
- You can assume the test file will be bigger than the test size (> 4 MiB)
- Document your assumptions if this document is unclear and they affect your design