Protocol Document

Trevor Mee

02/16/2025

Project 2

COP4635 - Systems and Networks 2

Table of Contents

1 Overview

This document describes the protocol and message exchange between a client and a HTTP server implemented in `httpServer.cpp` and `httpClient.cpp`. The server follows the HTTP/1.1 standard and uses TCP sockets to connect with a client via a web browser or command line tool. Project 2 implements a standalone client program that connects to the server using a user-specified IP address. The server processes GET requests while listening on port 60001.

2 Communication

Communication between the server and client follows a request-response model using TCP.

2.1 Client Request

1. The client prompts the user to enter the server's IP address.
2. The client establishes a TCP connection to the user-specified IP address on the ephemeral port 60001.
3. The client requests a specific file by sending a HTTP GET request.
4. The request is sent over the established TCP connection.

2.2 Server Processing

1. The server listens for incoming client connections on port 60001.
2. Once a connection has been established, the server parses the HTTP request and returns an HTTP response to the client.

2.3 Server Response

After processing, the server responds with either a successful response or an error message. Successful responses look similar to “HTTP/1.1 200 OK” while error messages look similar to “404 Not Found”.

2.4 Connection Handling

Once the server has been created, binded, and begins listening on port 60001, the server remains open while waiting for incoming client connections. After server processing, the connection is closed with the client. Additionally, the server has the capability of a graceful shutdown (Ctrl + C) via the `SIGINT` signal.

3 OSI Breakdown

The following sections describe how each layer from the OSI Reference Model is used in this project.

3.1 Application Layer

The application layer supports the HTTP/1.1 protocol and processes GET requests. The client formats and sends GET requests while the server processes these requests.

3.2 Transport Layer

The transport layer encapsulates the TCP segments to prepare the segments for transportation. The transport layer utilizes the socket(), connect(), bind(), listen(), and accept() system calls to manage connections.

3.3 Network Layer

The network layer routes the client request to the server's IP address and port number. The server can bind to any IP address through INADDR_ANY and uses the IPv4 protocol defined by AF_INET address family. The client specifies an IP address when connecting.