Documentação Trabalho Prático 2 – Redes de Computadores

Gabriel Castelo Branco Rocha Alencar Pinto
2020006523

Introdução

No trabalho prático 2 de redes de computadores, foi implementado um Client/Server que funciona através de um modelo publish/subscribe, onde os comandos digitados por um cliente são enviados pela rede, através de um soquete para serem processados por um servidor, que retorna uma resposta adequada.

O trabalho foi desenvolvido na linguagem C padrão, com uso da biblioteca <socket.h>.

Referências

J. Donahoo, Michael, L. Kenneth, Calvert; TCP/IP Sockets in C, Second Edition Practical Guide for Programmers

Beej's Guide to Network Programming Using Internet SocketsURL

Desenvolvimento e Desafios

O desenvolvimento do trabalho se deu em duas etapas distintas, porém complementares. Cada etapa teve desafios diferentes, sendo um deles um problema comum a ambas: a conexão do cliente com o servidor de maneira funcional. O principal problema que se apresentou ao desenvolver essa etapa foi fazer com que o cliente recebesse corretamente, durante o tempo de execução, as informações que deveria em paralelo à captação de novas ordens, ou seja, uma struct action que contivesse o post realizado, indicando quem postou e em qual tópico.

Doravante, analisar-se-ão as características individuais de cada parte do processo de desenvolvimento, em conjunto com seus respectivos desafios.

Servidor

A criação do servidor perpassou por 3 pontos principais: a criação de um soquete e subsequente conexão com um cliente, que rodaria em uma thread paralela, o recebimento adequado de informações vindas do cliente e, por fim, processamento dessas informações e envio da resposta correta.

A geração do soquete se deu através da biblioteca <socket.h>. Estabelecida a conexão entre cliente e servidor, todo o processo de conversa entre as partes se dá por meio da struct BlogOperation, que tem a forma seguinte:

```
struct BlogOperation{
        int client_id;
        int operation_type;
        int server_response;
        char topic[50];
        char content[2048];
};
```

Uma dificuldade enfrentada foi o reaproveitamento de uma action definida no início do processo de conexão, que era enviada de um lado ao outro. O processo de manipulação para a resposta se deu através de ponteiros, portanto foram necessários diversos ajustes para garantir funcionamento adequado do código. Ao receber uma action através do soquete, o servidor passava um ponteiro para a mesma para uma função processaEntrada, que, então, identificaria qual ação o cliente quis realizar e, na sequência, gerava uma resposta adequada, que era então salva na própria struct. Por fim, ela era reenviada através do mesmo canal de comunicação

Cliente

A implementação do cliente, tal qual a do servidor, se deu em 3 etapas distintas: a criação de um soquete e subsequente conexão com um servidor, a leitura de um comando com o seu processamento adequado e, por fim, envio da struct para o servidor, com espera pela resposta do mesmo.

A base para o funcionamento do cliente, assim como a do servidor, é uma struct BlogOperation, responsável levar as informações desejadas e a resposta do servidor ao comando anterior. Nesse quesito, um desafio enfrentado foi: como lidar com postagens de outros clientes? A solução encontrada para tal foi a utilização de uma thread paralela, que verificava constantemente se outro cliente havia feito postagem.

Conclusão

A criação de uma aplicação que funcione em um modelo cliente/servidor tal qual o do exercício se prova bastante desafiadora, principalmente quando exige diversas adaptações e código em paralelo para garantir o funcionamento adequado.