Trabalho Prático II

Igor Lacerda

Redes de Computadores

O objetivo do trabalho foi aprender sobre o gerenciamento de um servidor com múltiplos clientes, fazendo uso da linguagem C, por meio da implementação de serviço de *blogs*. Esse objetivo foi atingido com sucesso.[1]

Desafios, Dificuldades e Imprevistos

Tendo o trabalho anterior como inspiração, não foi tão difícil começar dessa vez (eu praticamente só copiei o antigo e fui removendo o que era específico dele). Mas havia uma certa dúvida se isso seria reaproveitável quando eu começasse a tornar o servidor *multithreaded*. Acho que a especificação ficou um pouco ambígua em alguns detalhes: por exemplo, não foi claro se era necessário fazer um *subscribe in* ou um *subscribe to*: ambos aparecem nos exemplos da especificação. No final, esse exemplo foi resolvido no fórum como apenas *subscribe*, mas é um sentimento angustiante: qualquer um desses detalhes pode descartar as horas de esforço investidas na implementação.

Acho que de maneira geral, essa angústia foi o maior desafio do trabalho. Depois que clicamos para enviar e o prazo acaba, não tem como voltar atrás. Sugiro que no futuro seja desenvolvido algum mecanismo para que nós, alunos, possamos validar nosso e não tenhamos que ficar preocupados com detalhes que realmente não deveriam importar. O trabalho já é complicado o suficiente por si só.

De fato, em quesitos técnicos, as principais dificuldades foram: o uso de múltiplas *threads* e a função para publicar os *posts*. Ambas são as dificuldades são parecidas e talvez estejam relacionadas com o fato de eu ainda estar fazendo Sistemas Operacionais. De qualquer maneira, o uso de múltiplas

[1] Eu decidi não usar o bendito Arial 12 dessa vez.

threads no servidor foi resolvido consultando-se um especialista e sua *playlist* do tema. Os bons e velhos *mutexes* vieram ao resgate, mas também foi usada uma *pool* de *threads* para limitar o tamanho do número de *threads* que o programa criaria. Foram feitos mais alguns ajustes, seguindo-se a *playlist*.

Para resolver a publicação de *posts* foi necessária uma certa gambiarra: criar uma *thread* no cliente só pra ouvir (e processar) as respostas do servidor, enquanto a *thread* principal leria a entrada e a enviaria para o servidor. Enquanto isso, do lado do servidor, a função de publicação procura entre todos os clientes e envia para quem for necessário, usando a *thread* associada a quem publicou. Realmente não sei se essa era a solução esperada, mas apesar de eu ter chamado de gambiarra, achei bem elegante.

Dessa vez, não tive nenhuma dificuldade relacionada ao uso da linguagem C. Claro que tive meus *segfaults*, e senti falta de um `std::vector`, mas foi muito menos insuportável do que eu achei que seria. Achei um trabalho legal de fazer e bastante proveitoso, especialmente os aspectos não relacionados a redes em si, como o uso dos mecanismos para evitar condições de corrida.