UNIVERSIDADE DE ÉVORA

Relatório do Trabalho Prático
Redes de Computadores

42647 - Ricardo Oliveira
43315 - Filipe Alfaiate

15 de fevereiro de 2021

1 Introdução

O trabalho prático realizado no âmbito da Unidade Curricular de Redes de Computadores tem como objetivo, desenvolver um servidor que estabelece a comunicação, de acordo com o protocolo definido *"plain text single line"*, entre diversos clientes. Foi também desenvolvida uma variante, utilizando o código de Hamming, de modo a corrigir os erros gerados no envio. Para além dos clientes comunicarem entre si, através do servidor, podem ter acesso a diversos comandos, definidos por quatro caracteres seguidos, ou não, por um ou mais parâmetros, separados por espaços.

2 Decisões

Inicialmente, o grupo decidiu utilizar como base, o cliente desenvolvido no decorrer das aulas práticas, tanto como o servidor que foi disponibilizado. Deste modo, foi então possível efetuar o envio de uma mensagem por parte do cliente para o servidor, retornando-a este, novamente à fonte, isto é, ao cliente (designado por *echo*). Após esta etapa concluída, foi necessário adaptar o servidor, de modo a conseguir, quando existe mais do que um cliente conectado, selecionar a qual dos clientes deve, ou não, ser dirigida a mensagem.

Visto que a implementação da interface deve ser a mais simples possível, determinámos que utilizaríamos o terminal para tal efeito.

Assim sendo, os clientes já conseguem enviar mensagens entre si. Portanto, a partir desta fase, previamente a iniciar com a implementação dos comandos, era necessário criar uma estrutura de dados de modo a armazenar a informação relativa a cada utilizador, para tal, decidimos criar uma struct.

Tendo a base da estrutura pronta, iniciámos a implementação da resposta por parte do servidor aos comandos indicados pelos utilizadores. Para isto, efetuámos a comparação dos primeiros 4 caracteres enviados em cada mensagem, de modo a perceber se é ou não, um comando. De modo ao servidor conseguir responder a cada um dos comandos, no ficheiro onde foi desenvolvida a struct User, implementámos as diversas funções que necessitámos de modo a obter a resposta necessária para o cliente. Para tal, o grupo decidiu que deveria guardar, escrevendo nos ficheiros "online.txt"e "regs.txt", os nomes já atribuídos a utilizadores online e os utilizadores registados, respetivamente. No caso, dos utilziadores registados, de modo a conseguir efetuar uma melhor e mais eficaz gestão de todos os utilizadores, decidimos guardar as suas informações por linha, contendo o nome, password e um valor boleano, determinando se é ou não operador, dividas por dois pontos ("nome:password:operador").

Nas funções implementadas que percorriam os ficheiros anteriormente referidos, tomamos duas posições distintas, sendo que para o ficheiro que guarda somente os nomes que estão a ser atualmente utilizados ("online.txt"), comparamos a mensagem enviada pelo utilizador com cada linha, por outro lado, para o ficheiro que contem todas as informações dos utilizadores registados ("regs.txt"), percorremos todo o ficheiro linha a linha, porém, quando comparada com a mensagem enviada pelo utilizador, fizemo-lo caracter a caracter, de modo a saber quando estávamos no nome (ou seja, antes da primeira ocorrência com dois pontos), na password (ou seja, antes da segunda ocorrência com dois pontos) ou no valor de operador (ou seja, após a segunda ocorrência com dois pontos).

Outra decisão que tomámos, após pesquisa, não sendo possível remover uma determinada linha completa ou alterar somente um caracter de uma outra linha, de um ficheiro, tomamos a decisão de criar um ficheiro temporário auxiliar ("temp.txt") que irá copiar todas as linhas existentes no ficheiro, menos a que desejamos apagar. Após todo o ficheiro percorrido, apagamos

o não temporário, renomeando o temporário para o nome do ficheiro que estávamos a utilizar. Quanto a alterar um caracter, foi utilizado o mesmo modo, desta vez, alterando a parte da linha em questão antes de a reescrever.

Numa fase final, decidimos tentar implementar o uso do código desenvolvido no decorrer da segunda aula prática, de modo a codificar e descodificar a mensagem, e para a correção de erros gerados, através do código de *Hamming*.

3 Compilação

De modo a colocar o trabalho prático totalmente funcional, é necessário compilar os ficheiros de código através dos seguintes comandos:

```
gcc -o server server.c          (ou equivalente)
gcc -o client client.c          (ou equivalente)
```

Após a compilação dos ficheiros, deve executá-los pela seguinte ordem:

```
./server
./client                        (num outro terminal)
```

Nota: De modo a simular a conexão de um novo cliente, deve abrir outro terminal e correr novamente o último comando.

Nota 2: Caso a compilação seja para o sistema que utiliza *Hamming*, adicionar '-lm' no final do comando de compilação.

4 Conclusão

Apesar das diversidades enfrentadas ao longo do trabalho, foi agradável pois conseguimos sempre pesquisar mais sobre a matéria lecionada nas aulas, adquirindo ainda mais conhecimento e, ao mesmo tempo, a ultrapassar as diversidades, interagindo assim muito com o código que teve vindo a ser desenvolvido.

Ao desenvolver este trabalho, o grupo realmente interessou-se no projeto e conseguimos vê-lo como algo que pode vir a ser utilizado, após algum trabalho adicional de melhoria funcional e desenvolvimento de *GUI (graphical user interface)*, a nível profissional.