UNIVERSIDADE DE ÉVORA

Sistema de Comunicação – Trabalho 2

Ricardo Matono (44448), João Rouxinol (44451) e André Rato (45517)

Ano Letivo 2020/2021

Redes de Computadores
Docente: Pedro Patinho

Índice

Problema

A empresa UltraSecure, Lda. pretende utilizar um sistema de comunicação para os membros da sua equipa, mas não pode confiar em software de terceiros. Assim sendo, precisa de implementar "in-house" um software deste tipo, garantindo que não há possibilidade de falhas de segurança nesta aplicação.

Servidor e cliente

O servidor e o cliente foram implementados em C, utilizando o protocolo TCP.

O servidor possui várias funcionalidades:

- sistema de *Login* e *Logout* através de *Usernames*, com um máximo de 20 clientes conectados simultaneamente;
- sistema de envio de mensagens privadas (comunicação em tempo real um para um);
- sistema de envio de mensagens globais (comunicação em tempo real um para todos);
- sistema de envio de mensagens privadas offline (comunicação em diferido um para um);
- sistema de envio de ficheiros privados (comunicação em tempo real um para um);
- sistema de envio de ficheiros globais (comunicação em tempo real um para todos);
- sistema de envio de ficheiros privados offline (comunicação em diferido um para um);

O servidor utiliza a função *Select* com o objetivo de lidar, não só com múltiplas conexões, mas também com o envio e receção de mensagens em simultâneo.

Login e Logout

O sistema de *Login* e *Logout* foi implementado com auxílio de um *Array* de apontadores (*Array* de *Strings*). Deste modo foi possível associar um número (o id do *Socket* correspondente ao cliente) a um *Username* (escolhido inicialmente pelo cliente).

Quando é estabelecida a conexão entre o servidor e um cliente, este tem a possibilidade de escolher um *Username*, com mais de 0 caracteres e menos de 21, que ainda não esteja a ser utilizado. Nesta verificação, sempre que um *Username* válido é inserido, isto é, sempre que uma *String* de tamanho maior ou igual a 1 e menor ou igual a 20, este é enviado para o servidor, no formato "/username user", onde é feito uma busca pelo mesmo no *Array* de *Usernames*: caso encontre, retorna o id do *Socket* onde este *Username* se encontra conectado; caso contrário, retorna -1. O cliente é então informado que o *Username* está em uso e que é necessário escolher outro *Username*.

Após o *Username* ser aceite, uma mensagem informativa de *Login* é enviada ao cliente.

Assim que um cliente se desconecta do servidor, o seu *Username* é retirado do *Array* de *Usernames*, deixando assim uma vaga adicional para outro cliente se conectar ao servidor.

Envio de mensagens privadas

O comando responsável pelo envio de mensagens privadas é "/msg dest msg", onde "dest" é o *Username* do destinatário e "msg" é o corpo da mensagem.

Neste caso, o servidor recebe o comando e descodifica-o, identificando qual o remetente, qual o destinatário e qual a mensagem. Depois da descodificação, o servidor procura o *Username* no *Array* de *Usernames* conectados; se encontrar, envia a mensagem para o *Socket* correspondente, no formato "sender to receiver: mensagem"; caso não encontre, estamos perante um envio diferido de mensagem.

Envio de mensagens globais

Neste caso, nenhum comando é necessário, para que possa ser enviado uma mensagem para todos os clientes conectados. Por isso, o servidor não necessita de descodificar a mensagem (obter destinatário e outros elementos), sendo apenas necessário guardar o id do *Socket* a partir do qual a mensagem foi enviada.

O servidor envia a mensagem para todos os clientes, exceto para quem enviou, utilizando o formato "sender to Everyone: mensagem".

Envio de mensagens privadas offline

O comando responsável pelo envio de mensagens privadas é "/msg dest msg", tal como foi visto num dos pontos anteriores.

Após a descodificação, o servidor procura o *Username* do destinatário no *Array* de *Usernames* de clientes conectados ao servidor: caso encontre, a mensagem é enviada; caso não encontre, a mensagem é guardada num *Array* de *Structs*, onde são guardadas todas as mensagens enviadas, mas não entregues.

Assim que um cliente se conecta ao servidor e escolhe um *Username* que está associado a uma das mensagens da lista de mensagens pendentes, o servidor envia a mesma, utilizando o mesmo formato ("sender to receiver: mensagem").

Envio de ficheiros privados

O comando responsável pelo envio de ficheiros privados é "/file dest file_path", onde "dest" é o *Username* do destinatário e "file_path" é o nome do ficheiro (ou caminho para o mesmo).

Neste caso, o ficheiro desejado é aberto, caso exista, e é lido de forma binária (bit a bit). Os bits correspondentes ao ficheiro são enviados ao servidor e é colocado o caractere 'e' no final, para marcar o fim do ficheiro.

Após isto, os bits do ficheiro são enviados para o cliente alvo e, aqui, é criado um outro ficheiro (.txt) onde são guardados os bits recebidos. De seguida, o ficheiro .txt é convertido noutro com o formato correto e, por fim, é apagado, concluindo-se assim o envio do ficheiro.

Envio de ficheiros globais

O comando responsável pelo envio de ficheiros privados é "/file all file_path", tal como no ponto anterior. O envio do ficheiro é feito exatamente da mesma forma que é feito no ponto anterior: o cliente envia para o servidor, o servidor envia para todos os clientes e, em cada cliente o ficheiro .txt é convertido para o formato correto.

Envio de ficheiros privados offline

O comando responsável pelo envio privado de ficheiros é "/file dest file_path", tal como num dos pontos anteriores.

Após a descodificação, o servidor procura o *Username* do destinatário no *Array* de *Usernames* de clientes conectados ao servidor: caso encontre, o ficheiro é enviado; caso não encontre, o ficheiro é guardado num *Array* de *Structs*, onde são guardados todos os ficheiros enviados, mas não entregues.

Assim que um cliente se conecta ao servidor e escolhe um *Username* que está associado a um dos ficheiros da lista de ficheiros pendentes, o servidor envia o mesmo, utilizando o mesmo formato ("sender to receiver: file").

Outros comandos úteis

O servidor dispõe de outros comandos para melhorar a experiência do utilizador:

- **/help:** apresenta a lista de comandos disponíveis e as suas sintaxes;
- **/list:** apresenta a lista de utilizadores online;
- **/exit:** disconeta o utilizador do servidor.