SCC0641 - Redes de Computadores

Plataforma de troca de mensagens

Lucas Rovere - 8139750

Luiz Votto - 8504006

Prof. Julio Cézar Estrella

Prof. Carlos Gomes Ferreira

São Carlos - 16 de junho de 2015

Engenharia da Computação

INTRODUÇÃO

O objetivo do projeto é criar uma plataforma que permita a troca de mensagens entre vários computadores diferentes. Implementamos o programa em linguagem C e usamos as plataformas de sockets que foram ensinadas em aula. Assim sendo, foi criado o programa que traz opções de envio de mensagens tais como adicionar contato a partir do IP, excluir contato, enviar mensagem e ver mensagem. O projeto de código se embasa em quatro threads - principal, de interface, de recebimento e de envio - as quais interagem entre sí para dar forma ao resultado final descrito acima. Essas threads geram algumas regiões críticas e problemas de programação concorrente os quais foram resolvidos e não cabe discorrer sobre essas soluções no espectro de Redes de Computadores. Nessa monografia, discorreremos com mais detalhes o funcionamento da plataforma, como se dá a utilização dela, aprofundar em algumas estruturas de dados usadas além de explicar as threads implementadas e qual papel desempenham.

UTILIZAÇÃO

Do ponto de vista do usuário, o programa pode ser tratado como um serviço de e-mail, com aspectos de mensagens instantâneas, pois não é possível que o usuário "acesse conversas", como num serviço de mensagens instantâneas. No entanto, assim como nos serviços de mensagens instantâneas, a conexão é feita a partir de um padrão P2P. O usuário pode adicionar ou remover contatos, além de enviar mensagens a um ou mais contatos de uma vez, como sugerido pela especificação do trabalho. Não é possível a personalização dos nomes de contato. Ao conectar no serviço, o usuário deve se "cadastrar" com apenas um nome de usuário. Quando desejar adicionar outro usuário, ele deve ter acesso ao IP em si da pessoa e apenas isso. Caso sua solicitação seja confirmada, ele irá receber uma mensagem de confirmação e o usuário será, neste momento, adicionado a sua lista de contatos. Ao ser excluído por um de seus contatos, ele receberá, da mesma forma, uma mensagem de notificação, e terá aquele contato removido de sua lista. Voltando ao funcionamento do envio de mensagens, o usuário poderá escolher a quantos contatos a mensagem será enviada. E deve fornecer os nomes dos contatos desejados. Basicamente, o IP deve ser fornecido pelo usuário, apenas no momento de adicionar outros contatos, depois disso, toda interação com o software é feita a partir do nome de usuário.

FUNCIONAMENTO

O programa é organizado para funcionar em 4 Threads: a thread principal, a de interface de usuário, a de envio e a de recebimento. A comunicação entre essas threads se dá por meio de duas listas encadeadas, declaradas como variáveis globais, e algumas variáveis de controle associadas, principalmente a essas listas, como o tamanho delas, por exemplo.

REGIÕES CRÍTICAS

Como explicado acima, o programa utiliza duas listas encadeadas para comunicar entre as threads. Estas listas, por sua vez, serão as regiões críticas do programa. Para facilitar essa comunicação, as regiões críticas do programa, na verdade, estão associadas a funções de manipulação. Para que isso funcione, qualquer alteração nas variáveis globais é feita apenas dentro dessas sub-rotinas. Não cabe ser tratado aqui o funcionamento dos semáforos binários usados para garantir a semântica das threads. Em nossa proposta, as listas encadeadas são manipuladas para funcionar como filas. Temos então a *fila de envio* e a *fila de recebimento*. A fila de envio é acessada pela thread de interface do usuário e pela de envio. O que acontece, em resumo, é: cada vez que o usuário envia uma mensagem, ela é incluída no final da fila de envio. E cada vez que o usuário vê uma mensagem, ela é retirada do final da fila de recebimento. Além disso, a thread de envio checa de tempos em tempos o tamanho da fila de envio e, caso haja alguma mensagem a ser enviada, ela a retira da fila. Ao mesmo tempo, a thread de recebimento fica bloqueada esperando por uma conexão. Conforme chegam mensagens, ela as adiciona ao final da fila. Isso tudo segue a estrutura do problema clássico *produtor-consumidor* em programação concorrente.

.

ESTRUTURA DA MENSAGEM

A mensagem é implementada a partir de uma *struct* chamada pacote de envio. Esse nome se da, pois nessa *struct*, temos todos os campos necessários para a comunicação entre os computadores que se comunicam. O tipo do pacote indica qual a sua função. No caso do programa enviado, implementamos solicitações de contato, confirmações de contato e exclusão de contato, além de mensagem em si. Temos também os campos de emissor e destinatário, que não são enviados e existem apenas para facilitar a manipulação da mensagem e a abstração do funcionamento no âmbito de redes, em si, na thread de interface. Isto facilita a programação modularizada do código. Por último, temos o campo de conteúdo, utilizado para a mensagem em si e para enviar o nome do usuário que está adicionando ou sendo adicionado.

THREAD PRINCIPAL

A thread principal, ou função "main" do programa, tem apenas a tarefa de inicializar as outras threads e os semáforos utilizados. Ela então será impedida de finalizar o programa a partir da função pthread_join() que aguarda a finalização da thread de interface do usuário.

INTERFACE DE USUÁRIO

A thread de interface de usuário funciona centrada em um *loop*. Esse *loop* é o menu que mostra as opções ao usuário. Além disso, mostra no início da tela o número de mensagens não lidas. Por se tratar de um programa implementado apenas no terminal do *Linux*, ele não atualizará sozinho a quantidade de mensagens não lidas. Para essa finalidade, temos a opção "atualizar", que apenas fará o *loop* rodar mais uma vez atualizando sozinho a quantidade.

Além dessa função temos as opções:

- **Adicionar contato**: Nessa opção o usuário deve fornecer o número de IP do contato a ser adicionado. O contato a ser adicionado receberá em seu *inbox* uma solicitação de contato e terá a opção de confirmar ou de excluir. Caso exclua, o outro não será notificado e a solicitação apenas não terá resposta. Tal sistema é parecido com o implementado na plataforma *Facebook*. Ao aceitar a solicitação, o usuário terá, automaticamente, aquele contato adicionado à sua lista de contatos e enviará, ao mesmo tempo, uma mensagem de confirmação. O usuário que enviou, originalmente, a solicitação apenas terá aquele contato adicionado à sua lista ao visualizar a mensagem de confirmação. Estas mensagens, além de informar o usuário do que está se passando, também informam o programa, que as usa para adicionar os contatos.

- **Listar contatos**: Essa opção apenas lista todos os contatos adicionados até então pelo usuário. Perceba que, caso o usuário não tenha visualizado a confirmação ou exclusão, ele não terá sua lista atualizada.

- **Excluir contato**: A opção de excluir contato funciona de forma análoga à de adicionar. No entanto, nesse caso, uma mensagem de exclusão é enviada ao contato desejado. Ele será automaticamente excluído de sua lista, no entanto, o usuário excluído, só terá o primeiro removido de sua ao visualizar a mensagem. No entanto, pacotes de mensagem de usuários não adicionados ou excluídos serão sempre ignorados.

- **Enviar mensagem**: Esta opção dá ao usuário a opção de escolher a quantos contatos a mensagem será entregue. O usuário, após escolher o número de contatos, deve dar ao programa os nomes dos usuários aos quais deseja enviar aquela mensagem. Feito isso, basta digitar a mensagem com até 1000 caracteres.

- **Ver mensagem**: A opção ver mensagem mostra na tela, de uma em uma, todas as mensagens do inbox. Sejam elas mensagens, solicitações, etc.

- **Sair**: Essa opção apenas finaliza o programa da forma correta, apagando os ponteiros alocados nas listas.

THREAD DE ENVIO

Essa thread manipula as mensagens no início da estrutura de região crítica que denominamos *fila de envio* e faz uma tentativa de enviar tais mensagens, abrindo o socket e tentando criar uma conexão, funcionando como na semântica de cliente no socket. Se ocorrer uma falha de envio, a mensagem retorna ao final da fila. Se a mensagem for, de fato, enviada com sucesso, teremos este elemento da fila descartado já que não há mais utilidade para ele.

THREAD DE RECEBIMENTO

Essa thread mantém um socket aberto esperando por conexões. Ela funciona na semântica de servidor no socket. Assim, a thread de recebimento insere na estrutura a qual chamamos de *fila de recebimento* a mensagem recebida. Após isso, a thread de interface faz a manipulação para exibir a mensagem ao usuário em sua tela.

Em resumo, a thread de envio e a de recebimento fazem o papel de cliente e servidor da conexão respectivamente.