ft_irc

Bate-papo de retransmissão da Internet

Resumo:

Este projeto é sobre criar seu próprio servidor IRC.

Você usará um cliente IRC real para se conectar ao seu servidor e testá-lo.

A Internet é regida por protocolos de padrões sólidos que permitem que computadores conectados interajam entre si.

É sempre bom saber.

Versão: 8

Conteúdo

Capítulo I

Introdução

Internet Relay Chat ou IRC é um protocolo de comunicação baseado em texto na Internet.
Ele oferece mensagens em tempo real que podem ser públicas ou privadas. Os usuários podem trocar mensagens diretas e entrar em canais de grupo.

Os clientes IRC conectam-se aos servidores IRC para ingressar nos canais. Os servidores IRC são conectados juntos para formar uma rede.

Capítulo II

Regras gerais

- Seu programa não deve travar em nenhuma circunstância (mesmo quando ficar sem memória) e não deve parar inesperadamente.
  Se isso acontecer, seu projeto será considerado não funcional e sua nota será 0.

- Você tem que entregar um Makefile que irá compilar seus arquivos de origem. Ele não deve revincular.

- Seu Makefile deve conter pelo menos as regras:
  $(NAME), todos, limpo, fclean e re.

- Compile seu código com c++ e os sinalizadores -Wall -Wextra -Werror

- Seu código deve estar em conformidade com o **padrão C++ 98.** Então, ele ainda deve compilar se você adicionar o sinalizador -std=c++98.

- Tente sempre desenvolver usando o máximo de recursos C++ que puder (por exemplo, escolha <cstring> em vez de <string.h>). Você tem permissão para usar funções C, mas sempre prefira suas versões C++, se possível.

- Qualquer biblioteca externa e bibliotecas Boost são proibidas.

Capítulo III

Parte obrigatória

Nome do programa	ircserv
Entregar arquivos	Makefile, *.{h, hpp}, *.cpp, *.tpp, *.ipp, um arquivo de configuração opcional NOME, all, clean, fclean, re port: A porta
Faça o arquivo	de escuta senha: A senha de conexão Tudo em
Argumentos	C++ 98. socket, close, setsockopt, getsockname, getprotobyname, gethostbyname, getaddrinfo,
Funções externas.	freeaddrinfo, bind, connect, listen, accept, htons, htonl, ntohs, ntohl, inet_addr, inet_ntoa, send, recv, signal, sigaction, lseek, fstat, fcntl, poll (ou equivalente) n/a Um servidor IRC em C++ 98
Autorizado pela Libft	
Descrição	

Você precisa desenvolver um servidor IRC em C++ 98.

Você **não deve** desenvolver um cliente.
Você **não deve** manipular a comunicação entre servidores.

Seu executável será executado da seguinte maneira:

./ircserv <porta> <senha>

• porta: O número da porta na qual seu servidor IRC estará escutando para mensagens de entrada Conexões IRC.

• senha: A senha de conexão. Será necessária para qualquer cliente IRC que tente se conectar ao seu servidor.

Mesmo que poll() seja mencionado no assunto e na escala de avaliação, você pode usar qualquer equivalente, como select(), kqueue() ou epoll().

III.1 Requisitos

- O servidor deve ser capaz de lidar com vários clientes ao mesmo tempo e nunca pendurar.
- Bifurcação não é permitida. Todas as operações de E/S devem ser **não bloqueantes.**
- Somente **1** poll() (ou equivalente) pode ser usado para manipular todas essas operações (leitura, escrita, mas também escuta, e assim por diante).

Como você precisa usar descritores de arquivo não bloqueantes, é possível usar funções de leitura/recepção ou gravação/envio sem poll() (ou equivalente), e seu servidor não estaria bloqueando.
Mas isso consumiria mais recursos do sistema.
Portanto, se você tentar ler/receber ou escrever/enviar qualquer descritor de arquivo sem usar poll() (ou equivalente), sua nota será 0.

- Existem vários clientes IRC. Você tem que escolher um deles como **referência.** Seu O cliente de referência será usado durante o processo de avaliação.
- Seu cliente de referência deve ser capaz de se conectar ao seu servidor sem encontrar qualquer erro.
- A comunicação entre cliente e servidor deve ser feita via TCP/IP (v4 ou v6).
- Usar seu cliente de referência com seu servidor deve ser similar a usá-lo com qualquer servidor IRC oficial. No entanto, você só precisa implementar os seguintes recursos:
  - ÿ Você deve ser capaz de autenticar, definir um apelido, um nome de usuário, entrar em um canal, envie e receba mensagens privadas usando seu cliente de referência.
  - ÿ Todas as mensagens enviadas de um cliente para um canal devem ser encaminhadas para todos os outros clientes que aderiram ao canal.
  - ÿ Você deve ter operadores e usuários regulares.
  - ÿ Então, você tem que implementar os comandos que são específicos para o canal operadores:
    - ÿ KICK - Ejetar um cliente do canal
    - ÿ CONVIDAR - Convidar um cliente para um canal
    - ÿ TÓPICO - Alterar ou visualizar o tópico do canal
    - ÿ MODE - Altera o modo do canal: i: Define/ · Remove canal somente para convidados · t: Define/Remove as restrições do comando TOPIC para o canal operadores · k: Definir/remover a chave do canal (senha) · o: Dar/ receber privilégio de operador de canal

- l: Definir/remover o limite de usuários para o canal

• É claro que se espera que você escreva um código limpo.

III.2 Somente para MacOS

Como o MacOS não implementa write() da mesma forma que outros sistemas operacionais Unix, você tem permissão para usar fcntl().

Você deve usar descritores de arquivo no modo não bloqueante para obter um comportamento semelhante ao de outros sistemas operacionais Unix.

Entretanto, você tem permissão para usar fcntl() somente da seguinte maneira: fcntl(fd, F_SETFL,

O_NONBLOCK); Qualquer outro sinalizador é proibido.

III.3 Exemplo de teste

Verifique absolutamente todos os erros e problemas possíveis (recebimento de dados parciais, baixa largura de banda e assim por diante).

Para garantir que seu servidor processe corretamente tudo o que você envia a ele, o o seguinte teste simples usando nc pode ser feito:

```
\$> nc 127.0.0.1 6667 com^Dman^Dd
\$>
```

Use ctrl+D para enviar o comando em várias partes: 'com', depois 'man' e depois 'd\n'.

Para processar um comando, você deve primeiro agregar os pacotes recebidos em para reconstruí-lo.

Capítulo IV

Parte bônus

Aqui estão os recursos extras que você pode adicionar ao seu servidor IRC para que ele se pareça ainda mais com um servidor IRC real:

- Lidar com transferência de arquivos.
- Um bot.

A parte bônus só será avaliada se a parte obrigatória for PERFEITA. Perfeito significa que a parte obrigatória foi feita integralmente e funciona sem mau funcionamento. Se você não passou em TODOS os requisitos obrigatórios, sua parte bônus não será avaliada.

Capítulo V

Submissão e avaliação por pares

Entregue sua tarefa no seu repositório Git como de costume. Apenas o trabalho dentro do seu repositório será avaliado durante a defesa. Não hesite em verificar novamente os nomes dos seus arquivos para garantir que estejam corretos.

Você é encorajado a criar programas de teste para seu projeto, mesmo que eles **não sejam enviados e não sejam classificados.** Esses testes podem ser especialmente úteis para testar seu servidor durante a defesa, mas também o de seus colegas se você tiver que avaliar outro ft_irc um dia. De fato, você é livre para usar quaisquer testes que precisar durante o processo de avaliação.

Seu cliente de referência será usado durante o processo de avaliação.

16D85ACC441674FBA2DF65190663F432222F81AA0248081A7C1C1823F7A96F0B74495
15056E97427E5B22F07132659EC8D88B574BD62C94BB654D5835AAD889B014E078705
709F6E02