

Batalha Naval Cliente Servidor TCP

Lucca Augusto

May 18, 2020

1 Descrição

Trabalho desenvolvido para a disciplina de redes do primeiro semestre de 2020. Este trabalho consiste em um jogo de batalha naval entre cliente e servidor utilizando a biblioteca socket e o padrão TCP. O jogo ocorre em turnos, primeiro o cliente ataca, depois o servidor, até que algum dos dois perca. O programa foi implementado em C e usa a biblioteca sys/socket.

2 Compilação e execução

O programa é compilado pelo makefile usando o comando

```
make
```

Para executar o programa é preciso executar primeiro o servidor e depois o cliente, já que o cliente não conseguirá se conectar se o servidor não estiver rodando. Utilizando o comando seguinte é possível executar o programa em apenas uma instância do shell. Executa o servidor em background e o cliente em seguida.

```
./servidor [-p porta] &  
./cliente [-i ip] [-p porta] [-a arquivo]
```

Uma outra forma de executar é usar um shell para o servidor e um shell para o cliente.
(shell 1)

```
./servidor [-p porta]
```

(shell 2)

```
./cliente [-i ip] [-p porta] [-a arquivo]
```

Caso nenhum argumento seja passado aos programas, o servidor iniciará na porta padrão 8000, o cliente acessará a porta padrão 8000 com o ip padrão 127.0.0.1 e carregará o tabuleiro do arquivo tabuleiro.txt

O arquivo de configuração do tabuleiro deve ser um arquivo texto no seguinte formato: Cada linha representa um navio, a linha deve começar com 2 letras que indicam qual navio a linha representa. pa para Porta-Avião, nt para Navio-Tanque, ct para Contratorpedeiro e sb para submarino. Depois dessa informação vem as coordenadas mais a esquerda e a cima

do navio, o ponto inicial do navio, separadas por espaços. Em seguida vem a letra 'v' ou 'h', indicando se o navio está na vertical ou horizontal respectivamente.

3 Código

O código é composto por 4 arquivos:

- servidor.c
- cliente.c
- batnaval.h
- utils.h

3.1 servidor.c

O arquivo servidor.c é responsável por inicializar o socket e ouvir a porta. Após inicializar o socket, o servidor fica em espera até algum cliente se conectar na porta. Feita a conexão, o servidor espera pela mensagem do cliente com as coordenadas de seu ataque. O servidor executa um loop até que receba do cliente a mensagem "perdi", informando que o cliente perdeu, ou até que o servidor perca o jogo, enviando para o cliente a mensagem "perdi" e finalizando sua execução. Recebida a mensagem de ataque, o ataque é computado e o servidor devolve uma mensagem para o cliente. Essa mensagem é da forma "x y 1" onde x e y são as coordenadas do ataque gerado pelo servidor e 1 é uma flag dizendo se o ataque anterior do cliente acertou algum navio.

3.1.1 Ataque do servidor

O ataque do servidor acontece da seguinte forma: Chuta uma coordenada aleatória na primeira vez, se acertar algum navio, os próximos ataques acontecem nas coordenadas adjacentes ao ponto acertado.

3.1.2 Recebendo um ataque do cliente

Caso a mensagem do cliente não seja "perdi", o servidor computa o ataque marcando um 0 em seu tabuleiro nas coordenadas informadas pelo jogador.

3.2 cliente.c

O arquivo cliente.c é responsável por se conectar a porta e realizar os ataques indicados pelo usuário. Após inicializar o socket, o cliente conecta na porta e espera pela entrada do usuário com as coordenadas de seu ataque. Após o usuário digitar as coordenadas, o cliente envia para o servidor uma mensagem com as coordenadas do ataque na forma "x y 1" onde x e y são a linha e a coluna do ataque, respectivamente, e 1 é uma flag que indica se

o último ataque do servidor acertou. Depois disso o cliente executa um loop até receber do servidor a mensagem "perdi" informando que o servidor perdeu, ou até que o cliente perca o jogo, enviando para o servidor a mensagem "perdi" e finalizando sua execução.

3.2.1 Ataque do cliente

O usuário digita as coordenadas de seu ataque, linha (em número) e coluna (em letra) e o cliente monta a mensagem e envia ao servidor. Caso o usuário digite 'p' ao informar a linha, o cliente mostrará na tela a disposição de seu tabuleiro e um mapa de seus acertos no tabuleiro inimigo.

3.2.2 Recebendo um ataque do servidor

Caso a mensagem enviada pelo servidor não seja "perdi", o cliente computa o ataque marcando um 0 em seu tabuleiro nas coordenadas enviadas pelo servidor.

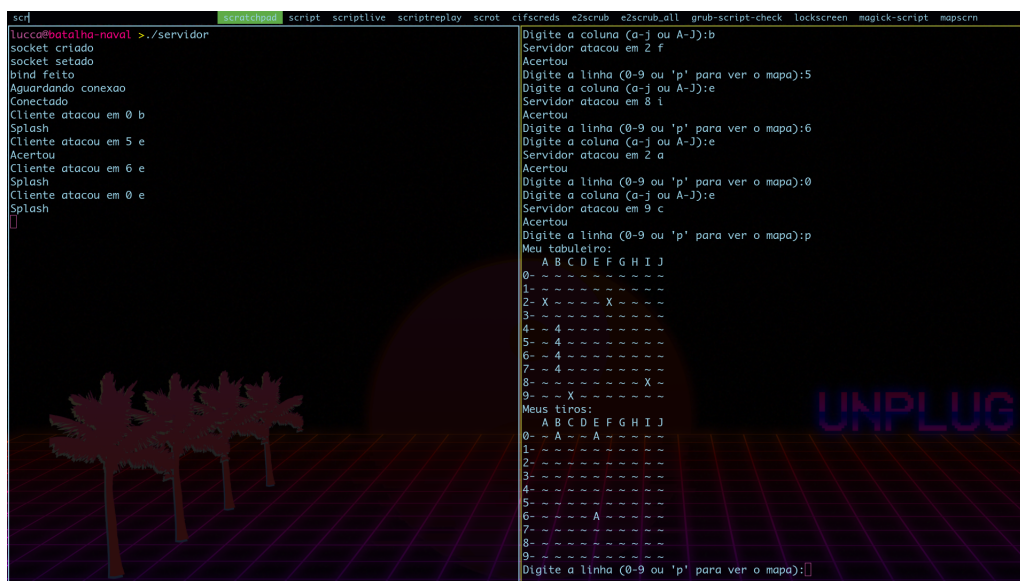
3.3 utils.h

Esse arquivo contém funções de IO e úteis para todos os arquivos em geral.

3.4 batnaval.h

Esse arquivo implementa o jogo de batalha naval em si. Ele é responsável por computar as jogadas, verificar se o dono do tabuleiro atual perdeu e inicializar os tabuleiros, lendo de um arquivo ou gerando aleatoriamente.

4 Screenshots



```
scr | lucca@batalha-naval > ./servidor |
socket criado
socket setado
bind feito
Aguardando conexao
Conectado
Cliente atacou em 0 b
Splash
Cliente atacou em 5 e
Acertou
Cliente atacou em 6 e
Splash
Cliente atacou em 0 e
Splash

Digite a coluna (a-j ou A-J):b
Servidor atacou em 2 f
Acertou
Digite a linha (0-9 ou 'p' para ver o mapa):5
Digite a coluna (a-j ou A-J):e
Servidor atacou em 8 i
Acertou
Digite a linha (0-9 ou 'p' para ver o mapa):6
Digite a coluna (a-j ou A-J):e
Servidor atacou em 2 a
Acertou
Digite a linha (0-9 ou 'p' para ver o mapa):0
Digite a coluna (a-j ou A-J):e
Servidor atacou em 9 c
Acertou
Digite a linha (0-9 ou 'p' para ver o mapa):p
Meu tabuleiro:
A B C D E F G H I J
0- ~ ~ ~ ~ ~ ~ ~ ~
1- ~ ~ ~ ~ ~ ~ ~ ~
2- X ~ ~ ~ X ~ ~ ~
3- ~ ~ ~ ~ ~ ~ ~ ~
4- ~ 4 ~ ~ ~ ~ ~ ~
5- ~ 4 ~ ~ ~ ~ ~ ~
6- ~ 4 ~ ~ ~ ~ ~ ~
7- ~ 4 ~ ~ ~ ~ ~ ~
8- ~ ~ ~ ~ ~ X ~ ~
9- ~ ~ ~ ~ ~ ~ ~ ~
Meus tiros:
A B C D E F G H I J
0- ~ ~ ~ ~ A ~ ~ ~
1- ~ ~ ~ ~ ~ ~ ~ ~
2- ~ ~ ~ ~ ~ ~ ~ ~
3- ~ ~ ~ ~ ~ ~ ~ ~
4- ~ ~ ~ ~ ~ ~ ~ ~
5- ~ ~ ~ ~ ~ ~ ~ ~
6- ~ ~ ~ ~ A ~ ~ ~
7- ~ ~ ~ ~ ~ ~ ~ ~
8- ~ ~ ~ ~ ~ ~ ~ ~
9- ~ ~ ~ ~ ~ ~ ~ ~
Digite a linha (0-9 ou 'p' para ver o mapa):
```

Figure 1: Screenshot mostrando a execução do programa