Laboratório de Redes de Computadores

Felipe Teles dos Santos[1], Vinicius Biondi Warlet[2]

[1]Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Avenida Ipiranga 6681 – Partenon – CEP: 90619-900 – Porto Alegre – RS – Brazil
{felipe.teles, vinicius.warlet}@acad.pucrs.br

Resumo. *Este artigo descreve sobre questões a respeito de uma aplicação que utiliza sockets. Esta aplicação é do tipo Cliente/Servidor, possui duas implementações, uma usando TCP e outra UDP, e é capaz de transferir um arquivo texto.*

1. Aplicação

- TCP: No arquivo tcp/tcp_client.c está a implementação do cliente TCP responsável por enviar o arquivo texto para o servidor TCP. A implentação do servidor TCP que vai receber o arquivo está localizada no arquivo tcp/tcp_server.c.
- UDP: No arquivo udp/udp_client.c está a implementação do cliente UDP responsável por enviar o arquivo texto para o servidor UDP. A implentação do servidor UDP que vai receber o arquivo está localizada no arquivo udp/udp_server.c.

Como executar:
Na hora de executar os clientes é preciso informar por parâmetro o "ip do servidor" e o "caminho do arquivo texto a ser enviado".

Exemplo de execução: `sudo ./tcp_client 192.168.1.1 ../800bytes.txt`

Na hora de executar os servidores é necessário passar por parâmetro o "tamanho em bytes do arquivo que o servidor vai receber".

Exemplo de execução: `sudo ./tcp_server 800`

2. Resolução dos Exercícios

Para os testes foram utilzados dois arquivos texto, um de 800 bytes (800bytes.txt) e outro de 2000 bytes (2000bytes.txt) que se encontram na raiz do projeto.

Exercício 1:

Arquivo de 800 bytes:

- Aplicação UDP: Enviado apenas um pacote UDP com os dados do texto.
- Aplicação TCP: Foram trocadas 8 mensagens TCP no total. Cliente enviou um pacote TCP SYN de 74 bytes, recebeu do servidor um TCP SYN+ACK de 74 bytes, o cliente enviou três novos pacotes TCP ACK de 74 bytes, TCP PSH+ACK com os dados todo(800 bytes de dados mais 67 bytes de overload) e

um TCP FIN+ACK. O servidor enviou três pacotes, TCP ACK de 66 bytes, RST+ACK de 66 bytes e um RST de 60 bytes.

Arquivo de 2000 bytes:

- Aplicação UDP: Foram gerados dois pacotes, um IPV4 com a flag "mais fragmentos ligada", e 1480 bytes de dados no arquivo.
- Aplicação TCP: Foram trocadas 7 mensagens TCP no total. Cliente enviou um pacote TCP SYN de 74 bytes, recebeu do servidor um TCP SYN+ACK de 74 bytes. o cliente enviou três novos pacotes TCP ACK de 66 bytes, TCP ACK um parte dos dados(1448 bytes de dados mais 66 bytes de overload) e um TCP FIN+PSH+ACK o restante(553 bytes de dados mais 66 bytes de overload).

Exercício 2:

2.1

Atráves do netem foi realizada a seguinte configuração:

```
tc qdisc add dev enp3s0 root netem loss 50%
```

Isso significa que vai simular a perda de 50% dos pacotes enviados.

No UDP, foi necessário realizar diversas tentativas para conseguir transferir o arquivo. Já no TCP, o próprio protocolo fez as retransmissões dos pacotes que foram pedidos.

- Aplicação TCP arquivo de 800 bytes: Foram necessários a troca de 14 pacotes para completar a transmissão do arquivo. Nesse caso houve a retransmissão de diversos pacotes.

- Aplicação TCP arquivo de 2000 bytes: Foram necessários a troca de 11 pacotes para completar a transmissão do arquivo. Também houve retransmissão de pacotes.

2.2

Atráves do netem foi realizada a configuração abaixo:

```
sudo tc qdisc add dev enp3s0 root netem delay 2000ms 1500ms
```

Essa configuração faz com que a latência de envio varie entre 2000 milisegundos e 1500 milisegundos.

Na transmissão via socket TCP houve retransmissão de pacotes, já com socket UDP continuou como nas execuções do exercício 1.