Universidade Federal do Espírito Santo
Centro Tecnológico

Redes de Computadores
Prof. Magnos Martinello

Projeto de Programação de Sockets

UDP Pinger

Nesta tarefa de programação, você escreverá um programa *ping* do cliente na sua linguagem de preferência (C, Python, Java). Seu cliente enviará uma mensagem *ping* simples a um servidor, receberá uma mensagem *pong* correspondente de volta do servidor e determinará o atraso entre o momento em que o cliente enviou a mensagem *ping* e recebeu a mensagem *pong*.

Esse atraso é denominado tempo de viagem de ida e volta (*round-trip time* - RTT). A funcionalidade oferecida pelo cliente e servidor é semelhante à fornecida pelo programa *ping* padrão, disponível nos sistemas operacionais modernos. Porém, os programas *ping* padrão usam o Internet Control Message Protocol (ICMP) (que veremos no Capítulo 4). Aqui, criaremos um programa *ping* baseado em UDP, fora do padrão (porém simples!).

Seu programa *ping* deverá enviar 10 mensagens *ping* ao servidor de destino por meio de UDP. Para cada mensagem, seu cliente deverá determinar e imprimir o RTT quando a mensagem *pong* correspondente for retornada. Como o UDP é um protocolo não confiável, um pacote enviado pelo cliente ou servidor poderá ser perdido. Por esse motivo, o cliente não poderá esperar indefinidamente por uma resposta a uma mensagem *ping*. Você deverá fazer que o cliente espere um certo

tempo por uma resposta do servidor; se nenhuma resposta for recebida, o cliente deverá considerar que o pacote foi perdido e imprimir uma mensagem de acordo.

As mensagens enviadas serão respondidas pelo servidor *independente da ordem de envio*. Isso quer dizer que seu código do cliente ping deve tratar as mensagens conforme seu número de sequência. Seu programa deve também tratar os erros de implementação do protocolo, como campos fora do tamanho correto ou fora da faixa de valores suportados. O campo "mensagem do pacote" deve conter caracteres alinhados à esquerda com no máximo 30 caracteres. Os pacotes identificados com erro devem ser ignorados.

O protocolo se baseia na técnica ***stop-and-wait*** no qual há apenas uma mensagem em trânsito a cada rodada de transmissão (RTT). No entanto, se alguma mensagem transmitida em uma rodada anterior chegar, a opção de projeto será aceitá-la caso esteja consistente e contabilizar o tempo medido, apesar do atraso.

Extra: Comparar o desempenho entre camadas de transporte distintas, medindo com UDP, QUIC e TCP.

Ao final, o programa deve calcular e imprimir na saída o número de pacotes perdidos, o RTT min, médio, max e desvio padrão. Exemplo de execução :

5 packets transmitted, 5 received, 0% packet loss, time 4005ms rtt min/avg/max/mdev = 17.879/19.492/24.632/2.587 ***ms***

Especificação do Protocolo:

Nome do Campo	**Tamanho fixo (Bytes)**
Número de sequência	5

Tipo da requisição (ping/pong)	0 = ping 1 = pong
Timestamp (unidades de mili-segundos)	4
Mensagem do pacote	30