tTransmissão de arquivos usando UDP com controle sobre TCP

Versionamento: V 0.0.1 - 24/01/2021

Introdução

Neste trabalho iremos implementar um sistema de armazenamento de arquivos na nuvem, similar ao Dropbox ou Google Drive. Clientes enviam seus arquivos para serem armazenados e o servidor fica responsável por armazená-los.

Iremos estabelecer um protocolo com dois canais: o canal de controle utilizará o protocolo TCP (Transmission Control Protocol). O canal de transmissão dos arquivos irá utilizar o protocolo UDP (*User Datagram Protocol*). Dessa forma, utilizaremos o UDP para a transmissão.

O trabalho deve expor os alunos a pelo menos dois aspectos de projeto em redes de computadores: os aspectos de implementação de um protocolo de controle e a utilização do protocolo UDP para transmissão de dados.

O Problema

Você tem uma empresa de armazenamento de arquivos na Nuvem. Diante disso, você precisa desenvolver uma aplicação para que os seus clientes possam enviar (*upload*) os arquivos que serão salvos e armazenados em seus servidores. Como mencionado, o envio dos arquivos será feito através do UDP. No entanto, precisaremos garantir a integridade dos arquivos, realizando a entrega em ordem e a retransmissão de pacotes perdidos. Para isto, utilizaremos uma janela deslizante. O controle da janela será feito sobre o TCP (canal de controle), enquanto os dados serão transmitidos usando UDP via canal de dados .

Arquitetura do Sistema

A arquitetura do sistema é composta por 1 servidor que atende uma quantidade arbitrária de clientes.

Servidor:
O protocolo de controle precisa ser capaz de lidar com vários clientes simultaneamente. Para isso, você pode utilizar a função [select] ou múltiplas *threads* para ler dados de vários soquetes de rede ao mesmo tempo. Para cada cliente conectado no servidor, o servidor deverá abrir um “porto” distinto utilizado pelo UDP para cada cliente. Essa informação deve ser repassada para os clientes. Além disso, o servidor deve implementar as estruturas de dados necessárias para criar a janela deslizante do receptor.

Cliente:
O cliente terá comunicação TCP e UDP com o servidor. Será usado UDP para transmissão do arquivo e com isto, pacotes podem se perder e chegar fora de ordem. Por isso, a ordenação dos pacotes é necessária. A sua implementação deve lidar com perdas fazendo

a retransmissão dos pacotes. O controle de retransmissão deverá ser feito através de uma janela deslizante controlada via canal de controle.

Mensagens do Protocolo

A operação do protocolo é dividida em duas partes. A primeira consiste em estabelecer uma conexão entre o servidor e o cliente. A segunda, por sua vez, lida com a transmissão do arquivo usando uma janela deslizante. Servidor e clientes trocam mensagens para decidir o início do *upload* do arquivo, envio de mensagens de controle da janela deslizante e o fim da transmissão.

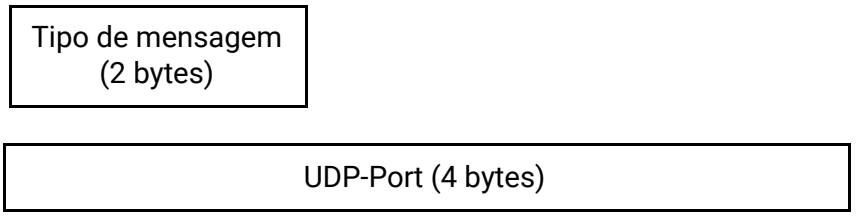
Abaixo descrevemos os tipos e cabeçalhos das mensagens de controle TCP e o cabeçalho das mensagens. Elas estão especificadas na ordem em que o sistema cliente/servidor deve seguir inicialmente. Todas mensagens possuem em seu cabeçalho um **tipificador de mensagem** que será definido por um número inteiro de 2 bytes. As mensagens estão descritas da seguinte forma: **NOME (Tipificador de mensagem)** - **Canal** - Descrição. O **NOME** serve para simplificar o entendimento do protocolo. Ele não é transmitido na rede. O **tipificador de mensagem** que é efetivamente transmitido na rede. É ele que será usado na implementação para identificar o tipo de mensagem. **Canal** indica se a mensagem será enviada via canal de controle (usando TCP) ou via canal de dados (usando UDP).

Estabelecendo uma conexão.

- **HELLO (1) - Controle:** Primeira mensagem de um programa cliente para se identificar para o servidor. O tamanho total da mensagem é de 2 bytes. Portanto, não há *payload*.

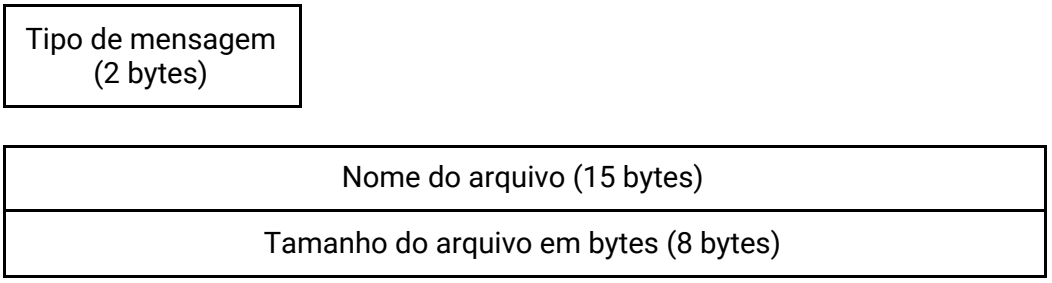
- **CONNECTION (2) - Controle:** Após o servidor receber uma mensagem **HELLO**, o mesmo deve enviar ao cliente uma mensagem do tipo **CONNECTION**, cuja função é informar ao cliente qual é o porto UDP em que ele estará escutando pelo respectivo cliente.

Cabeçalho da mensagem **CONNECTION**



- **INFO FILE (3) - Controle:** O cliente envia ao servidor informações sobre o tipo de arquivo que ele irá enviar. O nome do arquivo deverá ser codificado segundo a tabela ASCII.O tamanho máximo da *string* que compõe o nome do arquivo é de 15 bytes. Caso você receba um nome de arquivo acima de 15 bytes como parâmetro, sem o caractere ponto (.), com dois ou mais caracteres ponto (.), com menos de 3 caracteres após o ponto, ou com caracteres que não pertencem à tabela ASCII, o cliente deve informar ao usuário (imprimir na tela) **Nome não permitido.** O caractere nulo `'\0'` para terminação de strings em C *não* deve ser enviado na rede. O arquivo termina no terceiro caractere depois do (.) que separa o nome do arquivo com a extensão do mesmo. Por exemplo: **arquivo.doc**
  O final da string termina quando o servidor lê o terceiro caractere depois do (.), ou seja, neste caso depois do **c**.

Cabeçalho da mensagem **INFO FILE**



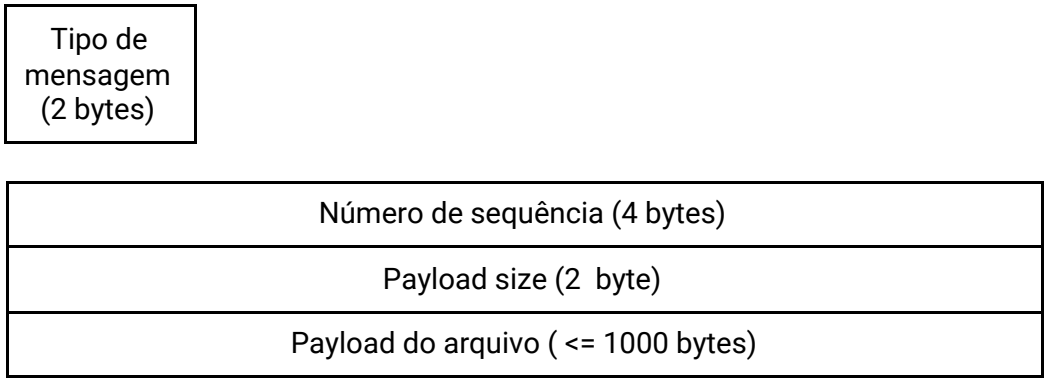
- **OK (4) - Controle**: Após o recebimento da mensagem com informações do arquivo, o servidor deve alocar as estruturas de dados necessárias para o início da transferência, o que inclui a estrutura de dados para compor a janela deslizante. Depois disso, o servidor deve enviar uma mensagem **OK** para o cliente. O tamanho total da mensagem é de 2 bytes. Portanto, não há cabeçalhos extras.

- **FIM (5) - Controle**: A mensagem FIM é enviada do servidor para o cliente. Essa mensagem informa para o cliente que o servidor recebeu todos os pacotes e já pode finalizar a conexão TCP entre o servidor e o cliente. O tamanho total da mensagem é de 2 bytes. Portanto, não há cabeçalhos extras.

Especificação do tipo de mensagem de envio do arquivo: Transferência do arquivo e controle da janela deslizante.

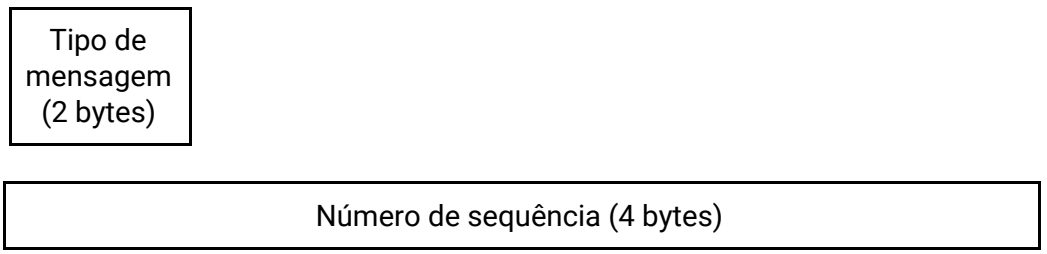
- **FILE (6) - Dados:** Após o cliente receber a mensagem **OK** do servidor, ele começará a transmitir o arquivo. A mensagem do tipo **FILE** será utilizada para realizar a transmissão do arquivo, portanto, esse cabeçalho refere-se à comunicação UDP. O campo *Número de sequência* é um contador que será incrementado a cada pacote enviado. A primeira mensagem deve ter número de sequência zero e as mensagens seguintes devem ter o número de sequência da mensagem anterior mais 1. No caso de reenvio de parte dos dados (o que pode ocorrer quando houverem erros de transmissão), o número de sequência deve ser o mesmo empregado na primeira transmissão daquela parte do arquivo. O tamanho de cada pacote será limitado a 1 kbyte.

Cabeçalho da mensagem **FILE**



- **ACK (7) - Controle:** O servidor deve enviar para o cliente mensagens de confirmação (ACK) para pacotes corretamente recebidos. O campo Número de sequência refere-se a qual sequência de pacote o servidor está confirmando o recebimento.

Cabeçalho da mensagem **ACK**



Os tipos de mensagens citados acima são o mínimo que deverá ser implementado para garantir o funcionamento. Caso adicione novos tipos de mensagens, você deverá especificar e justificar as alterações no relatório, de tal forma que fique clara a sua decisão. Fica a critério de cada aluno decidir qual janela deslizante implementar (*go back n,* Janela deslizante com retransmissão seletiva ou *"selective repeat"*). Além disso, o tamanho da janela deslizante e o tempo de recebimento de um ACK (*timeout*), bem como decidir por quanto tempo ficará tentando uma retransmissão. Fica, portanto, vetado o uso de janela deslizante com tamanho 1 (tanto para o transmissor quanto receptor). Isso implica que não será permitido o *stop-and-wait* e a retransmissão sucessiva de um mesmo pacote.

Implementação

Você deve implementar uma versão do servidor e uma versão do cliente que sejam compatíveis com os protocolos IPv4 e IPv6. Seu cliente deve ler qualquer arquivo que você deseja enviar.

Seu servidor deve receber um número de porta na linha de comando especificando em qual porta ele vai estabelecer o canal de controle.

Seu cliente deve receber na linha de comando o endereço IP e a porta do servidor para estabelecimento do canal de controle. Além disso, deve receber uma string contendo o nome do arquivo que será enviado. Exemplo de execução dos programas em dois terminais distintos:

```
no terminal 1: ./servidor 51511
no terminal 2: ./cliente 127.0.0.1 51511 arquivo.doc  # IPv4
no terminal 3: ./cliente ::1 51511 arquivo.doc  # IPv6
```

Emulador do estado da rede:

Para emular vários estados diferentes da rede e o comportamento do protocolo, você deverá utilizar a ferramenta TC[1] (*traffic control*). Essa ferramenta permite configurar alguns parâmetros de rede, como por exemplo, *delay*, vazão, perda e corrupção de pacotes. A retransmissão via UDP será testada adicionando perdas aleatórias na interface, como no exemplo abaixo.

Exemplo:

tc qdisc add dev ***dummy*** root netem rate 1mbit limit 20 delay 10ms loss 10%

[1] https://man7.org/linux/man-pages/man8/tc-netem.8.html

No exemplo acima, o TC implementou uma taxa de erro de 10% na interface de rede ***dummy***. Deve ser criada uma interface (chamada aqui de dummy) para que as alterações não afetem o funcionamento do computador.

tc qdisc del dev ***dummy*** root ip link del dev ***dummy***

Você deve deletar as configurações do TC para incluir novas. O exemplo acima mostra como deletar a configuração na interface dummy através de linhas de comando.

Abaixo um exemplo de como configurar uma interface ***dummy*** **(estes comandos requerem senha de administrador ou acesso via sudo)**.

```
modprobe dummy
ip link add dummy type dummy
ip link set dummy up
ip addr add 192.168.0.1 dev dummy
```

Avaliação

Este trabalho deve ser realizado individualmente e deve ser implementado em uma das seguintes linguagens: C, C++, Java, Python, Rust e Go, utilizando apenas as bibliotecas *sockets* das respectivas linguagens. Seu programa deve rodar no sistema operacional Linux e, em particular, não deve utilizar bibliotecas do Windows, como o winsock. Procure escrever seu código de maneira clara, com comentários pontuais e bem indicados; isto facilita a correção dos monitores e tem impacto positivo na avaliação.

Correção:

Seu servidor e cliente serão corrigidos de forma semi-automática através de uma bateria de testes. Os testes irão avaliar a funcionalidade do protocolo, principalmente a verificação do recebimento correto dos arquivos (arquivo não corrompido). Seu cliente deve imprimir se o arquivo foi enviado corretamente ou não (caso tenha ocorrido erro, como por exemplo o erro de *timeout*). Além disso, será avaliada manualmente a implementação da janela deslizante. Os testes serão conduzidos através de simulações de situações adversas no enlace entre o servidor e o cliente, como por exemplo, uma taxa de perda de pacote alta.

Entrega

Cada aluno deve entregar documentação em PDF de até 4 páginas (duas folhas), sem capa, utilizando fonte tamanho 10, e figuras de tamanho adequado ao tamanho da fonte. A documentação deve discutir desafios, dificuldades e imprevistos do projeto, bem como as soluções adotadas para os problemas. Além disso, deve-se apresentar qual algoritmo de janela deslizante você implementou e os detalhes de sua implementação.

Cada aluno deverá entregar o código fonte. A implementação deve utilizar apenas as bibliotecas padrão de cada linguagem (i.e., não devem requerer a instalação de bibliotecas adicionais) e devem compilar no Linux (i.e., não devem utilizar bibliotecas específicas de outros sistemas operacionais). Entregas em C e C++ devem incluir um Makefile para compilação; entregas em Java devem incluir arquivos JAR; entregas em Rust devem utilizar cargo para compilação. Em qualquer um dos casos, os programas devem ser chamados “cliente” e “servidor” (com extensões das respectivas linguagens, quando adequado).

Ponto extra

- Modificações ao protocolo de controle para permitir a transferência de múltiplos arquivos simultaneamente do mesmo cliente. Suas modificações devem ser compatíveis com o protocolo original (*backwards compatibility*).
- Comparar o desempenho da janela deslizante GO-Back-N e janela deslizante seletiva. Você deve considerar diferentes taxas de perda de pacotes e analisar o impacto disso em cada tipo de janela deslizante. Apresente e discuta os resultados em forma de gráficos.

Dicas e cuidados a serem observados

- O guia de programacão em rede do Beej (http://beej.us/guide/bgnet/) e o Python Module of the Week (https://pymotw.com/2/select/) têm bons exemplos de como organizar um servidor com *select*.
- Poste suas dúvidas no fórum específico para este TP 2 na disciplina, para que todos vejam.
- Escreva seu código de maneira clara, com comentários pontuais e indentado. O código será considerado na nota.
- Consulte-nos antes de usar qualquer biblioteca diferente que não seja padrão.
- Implemente o trabalho por partes. Por exemplo, crie o formato das mensagens e tenha certeza que o envio e recebimento estão corretos antes de se envolver com o envio do arquivo. Posteriormente, pense na implementação da janela deslizante.

Desconto de nota por atraso

Os trabalhos poderão ser entregues até a meia-noite do dia especificado para a entrega. O horário de entrega deve respeitar o relógio do sistema Moodle, ou seja, a partir de 00:01 do dia seguinte à entrega no relógio do Moodle, os trabalhos já estarão sujeitos a penalidades. A fórmula para desconto por atraso na entrega do trabalho prático ou resenha é:

Desconto = 2^{d-1}/0.32 %

onde d é o atraso em dias. Note que após 5 dias, o trabalho não pode ser mais entregue.