Laboratório de Redes de Computadores - Trabalho 1

Objetivos

O objetivo geral do trabalho é desenvolver um programa que implementa um protocolo de comunicação customizado, encapsulado em pacotes UDP. Tal protocolo deve ser utilizado para busca automática de dispositivos em uma topologia de rede, permitindo que tais dispositivos se comuniquem. Os objetivos específicos incluem:

- Trabalhar conceitos relacionados à pilha de protocolos de rede;
- Compreender de maneira prática o mecanismo de comunicação por *sockets*;
- Desenvolver uma aplicação distribuída utilizando o conceito de comunicação entre pares e requisitos de confiabilidade.

Descrição

A aplicação desenvolvida deve ser capaz de permitir a comunicação entre diversos dispositivos, executando de forma distribuída. No contexto do trabalho, os dispositivos deverão comunicar-se por um protocolo a ser desenvolvido, encapsulado em pacotes UDP. O trabalho deverá ser realizado com *sockets*, sendo necessário compreender o uso da biblioteca e o modelo de comunicação baseado em datagramas.

Os dispositivos irão descobrir automaticamente quem são seus vizinhos, ou seja, todos os outros dispositivos ativos na rede. Essa funcionalidade será implementada em uma aplicação que irá prover duas funções principais: 1) responder às requisições de outros dispositivos, mantendo atualizada uma lista dos dispositivos ativos; 2) permitir que um dispositivo ingresse no ambiente e envie mensagens para dispositivos específicos.

Descrição do protocolo de descoberta e comunicação

Cada dispositivo deve enviar uma mensagem HEARTBEAT para todos os endereços da rede (isto é, usando o endereço de broadcast) logo após sua inicialização, para informar sua presença na rede. Adicionalmente, cada dispositivo deve enviar uma mensagem do tipo HEARTBEAT para todos os endereços da rede a cada 5 segundos, informando que encontra-se ativo. Os outros dispositivos irão receber essa mensagem e atualizar sua lista de dispositivos conhecidos. Quando um dispositivo ficar inativo por mais de 10 segundos, o mesmo deve ser removido da lista. Para comunicar-se com outro dispositivo deve-se enviar mensagens do tipo TALK ou SEND, identificando o dispositivo específico.

A seguir, é apresentada a especificação de cada mensagem:

- `HEARTBEAT <nome>`. Mensagem enviada para o endereço de broadcast durante a inicialização de cada dispositivo e a cada 5 segundos para informar que o dispositivo ainda está ativo. O campo `<nome>` identifica o remetente.

- `TALK <id> <dados>`. Mensagem enviada para se comunicar com outro dispositivo. O campo `<id>` é um identificador único da mensagem (gerado pelo remetente) e `<dados>` representa o conteúdo da mensagem como string. O destinatário deve responder com `ACK <id>` para confirmar o recebimento. Mensagens do tipo TALK devem ser geradas sob comando do usuário.

- `FILE <id> <nome-arquivo> <tamanho>`. Inicia uma transferência de arquivo. O campo `<id>` é o identificador único da mensagem, `<nome-arquivo>` é o nome do arquivo a ser enviado e `<tamanho>` é seu tamanho total em bytes. O destinatário deve responder com `ACK <id>` para confirmar o recebimento da solicitação. Após essa mensagem, o remetente deve iniciar o envio dos dados do arquivo por meio de blocos menores, numerados sequencialmente, utilizando mensagens do tipo CHUNK.

- `CHUNK <id> <seq> <dados>`. Essa mensagem transporta uma parte dos dados do arquivo sendo transferido. O campo `<id>` é o identificador único da mensagem, `<seq>` é o número de sequência do bloco, e `<dados>` contém o conteúdo do bloco do arquivo codificado em base64. O destinatário deve responder com `ACK <id>` para cada bloco recebido. Caso não receba a confirmação em tempo razoável, o remetente

deve retransmitir o bloco correspondente. Blocos duplicados devem ser descartados pelo destinatário.

- `END <id> <hash>`. Essa mensagem indica o final da transferência de um arquivo. O campo `<id>` é o identificador único da mensagem, e `<hash>` é um código de verificação de erros (hash SHA-256 ou MD5, por exemplo) do conteúdo completo do arquivo enviado O destinatário deve comparar o hash recebido com o hash calculado localmente e acusar erro em caso de divergência. Arquivos corrompidos devem ser descartados. O destinatário deve responder com `ACK <id>` caso o conteúdo do arquivo seja validado com sucesso, caso contrário deve responder com `NACK <id>`.

- `ACK <id>`. Confirma o recebimento de qualquer mensagem identificada por `<id>`. Pode ser usada para confirmar mensagens do tipo `TALK`, `CHUNK`, `FILE` ou `END`.

- `NACK <id> <motivo>`. Indica que a mensagem identificada por `<id>` não foi aceita ou processada corretamente. Pode ser usada, por exemplo, para informar que um arquivo transferido está corrompido (hash inválido) ou que um bloco não pôde ser processado. O campo `<motivo>` pode ser usado de forma livre para identificar o erro.

Requisitos de confiabilidade

Como o protocolo será implementado sobre UDP, é responsabilidade do aluno garantir os mecanismos de confiabilidade necessários. A implementação deve conter, no mínimo:

- Confirmação de recebimento (ACK) e retransmissão em caso de perda de pacotes;
- Detecção e eliminação de mensagens duplicadas;
- Detecção de pacotes fora de ordem para mensagens do tipo CHUNK;
- Validação de integridade ao final da transferência de arquivos, comparando o hash recebido com o hash calculado;
- Envio de arquivos grandes deve ser feito em blocos, sem carregar o conteúdo completo na memória ao mesmo tempo (deve funcionar qual qualquer tamanho de arquivo).

Além de implementar os mecanismos de confiabilidade descritos, é fundamental que o grupo realize experimentos para validar o funcionamento correto do protocolo em condições adversas. O protocolo deve ser capaz de lidar adequadamente com perda de pacotes, duplicação, entrega fora de ordem, atrasos e pacotes corrompidos. Recomenda-se o uso de ferramentas como o `netem`, disponível no Linux, para simular esses cenários. Os testes devem demonstrar que a transferência de arquivos, em particular, ocorre de forma íntegra e confiável mesmo sob essas condições, e que o protocolo responde corretamente com retransmissões, confirmações e verificação de integridade.

Utilize capturas de pacotes com um sniffer de rede como `Wireshark` ou `tcpdump` para demonstrar as condições validadas durante os experimentos. As capturas devem ser usadas como evidência de que o protocolo foi testado sob as situações adversas simuladas, comprovando o funcionamento correto dos mecanismos de confiabilidade.

Interface de usuário

A aplicação desenvolvida deve fornecer uma interface de linha de comando interativa para que o usuário possa testar o funcionamento do protocolo de forma manual. Essa interface deve permitir o envio de mensagens e arquivos para outros dispositivos ativos na rede, bem como listar os dispositivos ativos. Ao menos os seguintes comandos devem ser implementados:

- `devices`. Exibe a lista de dispositivos atualmente ativos na rede. Para cada dispositivo, devem ser mostradas as seguintes informações:
  - Nome do dispositivo;
  - Endereço IP e porta associados;
  - Tempo (em segundos) desde o último `HEARTBEAT` recebido.
- `talk <nome> <mensagem>`. Envia uma mensagem de texto para o dispositivo com o nome especificado. Após o envio, a interface deve exibir uma mensagem indicando se a operação foi concluída com sucesso ou se houve falha na transmissão.
- `sendfile <nome> <nome-arquivo>`. Inicia a transferência de um arquivo para o dispositivo com o nome indicado. A interface deve indicar o progresso da transferência (por exemplo, porcentagem ou número de blocos enviados) e informar se a transferência foi concluída com sucesso ou falhou devido à perda ou corrupção de dados. O nome do arquivo especificado deve ser lido a partir do sistema de arquivos local.

Entrega

O trabalho deverá ser realizado em duplas ou individualmente. Envie um relatório descrevendo e documentando a implementação, apresentando cenários de teste e validação do seu funcionamento. Esse relatório deverá ter entre 5 e 8 páginas. Juntamente com o relatório, deve ser enviado o código-fonte do programa desenvolvido. O trabalho será apresentado no dia 05/05, sendo responsabilidade do grupo organizar-se para uma apresentação de aproximadamente 5 a 7 minutos, demonstrando o funcionamento de sua implementação.