PUCRS – Laboratório de Redes de Computadores

Trabalho 1 – UDP– 2025/1

Samara Tavares

---

1. Introdução

O código em questão implementa um sistema de comunicação baseado em Datagramas (UDP), permitindo que dispositivos se comuniquem de maneira eficiente em uma rede local. O objetivo principal do programa é enviar e receber mensagens, arquivos e chunks de dados entre dispositivos, além de manter a visibilidade na rede através do envio de pacotes de "heartbeat". O código utiliza a linguagem Java e conceitos como DatagramSocket, Threads e manipulação de arquivos.

2. Estrutura do Código

O código é estruturado em torno de três componentes principais: escuta de pacotes recebidos, envio de mensagens e arquivos, e a manutenção da lista de vizinhos na rede. Além disso, a lógica de comunicação é baseada em pacotes de mensagens identificados por prefixos, como "TALK", "FILE", "HEARTBEAT", entre outros.
A classe Dispositivo é responsável por executar os métodos principais de comunicação e gerenciamento, e utiliza uma ConcurrentHashMap<String, Vizinho> para armazenar informações sobre os dispositivos vizinhos (nome e IP).

3. Descrição dos Métodos e Funcionalidades

3.1. Método devices

Este método é utilizado para exibir a lista de dispositivos ativos na rede. Ele percorre o mapa vizinhos, exibindo o nome e o endereço IP de cada dispositivo armazenado. Este comando permite ao usuário verificar quais dispositivos estão disponíveis na rede local.

- **Entrada**: Não há parâmetros.

- **Saída**: Exibe a lista de dispositivos ativos, com seus respectivos nomes e endereços IP.

```
nome do dispositivo: A
> devices
dispositivos ativos:
 - B (172.18.0.3)
```

3.2. Método talk

O método talk permite que o dispositivo envie uma mensagem de texto para outro dispositivo na rede. O comando talk <nome> <mensagem> envia uma mensagem formatada como "TALK:<id>:<nome do remetente>:<mensagem>" para o dispositivo especificado.

- **Entrada**: O nome do dispositivo de destino e a mensagem a ser enviada.

- **Saída**: A mensagem é enviada via UDP para o dispositivo especificado. O método também imprime um log informando o envio da mensagem.

Trabalho 1 – UDP– 2025/1

Samara Tavares

```
nome do dispositivo: A
> talk B hello
enviado para B
```

```
nome do dispositivo: B
> mensagem de A: hello
```

3.3. Método file

O método file permite o envio de arquivos para dispositivos vizinhos. O comando file <nome> <arquivo> envia o nome do arquivo e seu tamanho para o dispositivo de destino. Após a recepção de um ACK (confirmação de recebimento), o arquivo é enviado em pedaços (chunks) utilizando a função chunk. Não foi possível verificar o recebimento do ACK nos testes.

- **Entrada**: O nome do dispositivo de destino e o caminho do arquivo a ser enviado.

- **Saída**: O arquivo é transmitido em partes de 512 bytes, utilizando pacotes UDP. Após o envio, o código imprime logs de progresso e confirmações.

```
nome do dispositivo: A
> recebendo arquivo: teste.txt (1958 bytes)
ACK enviado para 172.18.0.3 com id 1
```

```
nome do dispositivo: B
> file A teste.txt
FILE enviado, aguardando ACK...
> ACK recebido para id 1
```

3.4. Método escutar

Este método fica em execução contínua, aguardando pacotes de dados recebidos na rede. Ele processa diferentes tipos de mensagens, como "HEARTBEAT", "TALK", "FILE", "CHUNK", "END", "ACK" e "NACK". O método verifica o tipo de cada mensagem e executa a ação apropriada, como armazenar um novo vizinho, exibir uma mensagem recebida ou enviar um ACK em resposta a um arquivo ou chunk recebido.

- **Entrada**: Não há parâmetros; o método escuta pacotes recebidos na rede.

- **Saída**: Processamento das mensagens recebidas, incluindo logs e respostas (ACKs ou NACKs).

3.5. Método heartbeat

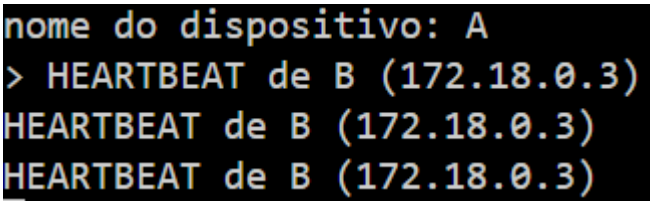
O método heartbeat envia periodicamente pacotes de "heartbeat" para a rede, a fim de manter o dispositivo visível para os outros dispositivos. A cada 5 segundos, um pacote "HEARTBEAT" é enviado para o endereço de broadcast.

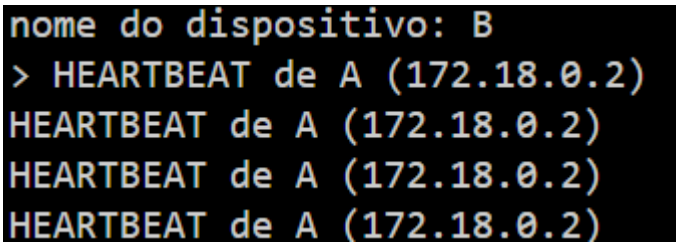
- **Entrada**: Não há parâmetros.

- **Saída**: Envio periódico de pacotes UDP.

Trabalho 1 – UDP– 2025/1

Samara Tavares





3.6. Método chunk

Este método é utilizado para enviar partes de um arquivo em pacotes UDP. Cada pedaço de arquivo é codificado em Base64 e enviado como uma mensagem "CHUNK" com um identificador único, juntamente com um número de sequência. Não foi possível verificar o recebimento do ACK nos testes.

- **Entrada**: O IP do dispositivo de destino, o ID do arquivo e o nome do arquivo.
- **Saída**: O arquivo é transmitido em partes (chunks), cada uma sendo enviada como um pacote UDP. O método aguarda um intervalo entre os pacotes para garantir uma transmissão eficiente.

3.7. Método end

O método end é responsável por enviar uma mensagem de término do arquivo para o dispositivo de destino. Após a transmissão do arquivo, um hash do arquivo é gerado utilizando o algoritmo SHA-256 para garantir a integridade dos dados. O hash é enviado junto com a mensagem "END" para o dispositivo de destino.

- **Entrada**: O IP do dispositivo de destino, o ID do arquivo e o nome do arquivo.
- **Saída**: Uma mensagem "END" com o hash do arquivo é enviada para o dispositivo de destino.

3.8. Métodos de Confirmação: ack e nack

Os métodos ack e nack são utilizados para enviar confirmações de recebimento (ACK) ou mensagens de erro (NACK). O ack é enviado quando um pacote ou arquivo é recebido corretamente, enquanto o nack é enviado caso haja algum erro, como um arquivo corrompido.

- **Entrada**: O IP do dispositivo e o ID do pacote ou arquivo.
- **Saída**: Um pacote de ACK ou NACK é enviado de volta para o dispositivo de origem.

3.9. Método limparVizinhos

Este método monitora a lista de vizinhos e remove aqueles que não enviaram um "heartbeat" nos últimos 15 segundos. A limpeza dos vizinhos é realizada periodicamente a cada 5 segundos.

- **Entrada**: Não há parâmetros.
- **Saída**: Dispositivos que não enviaram um "heartbeat" dentro do prazo são removidos da lista de vizinhos.

4. Simulação de Duas Máquinas Usando Docker Desktop

A simulação de dois dispositivos de rede foi realizada utilizando o Docker Desktop, com o objetivo de criar um ambiente de rede isolado entre dois contêineres. Os dispositivos se comunicam entre si como se estivessem em uma rede local.

Passos para a Simulação:

PUCRS – Laboratório de Redes de Computadores

Trabalho 1 – UDP– 2025/1

Samara Tavares

Construção da Imagem Docker:
A imagem do dispositivo foi criada com o seguinte comando:

```
docker build -t labredes .
```

Criação de Rede Virtual:
Foi criada uma rede Docker isolada para que os contêineres possam se comunicar:

```
docker network create labredes-net
```

Inicialização dos Contêineres:
Os contêineres para simular os dispositivos foram iniciados com os seguintes comandos:

```
docker run -it --network labredes-net --name dispositivo1 labredes
docker run -it --network labredes-net --name dispositivo2 labredes
```

Comunicação entre os Dispositivos:

Após a inicialização, os dispositivos podem se comunicar entre si usando os comandos talk e file. A rede Docker permite que os dispositivos se reconheçam e se comuniquem diretamente como se estivessem em uma rede local.