Nome: Matheus Lucca Alves RGA:2020.1907.041-0
Nome: Danilo dos Santos de Oliveira RGA:2020.1907.045-3
Relatório - Sistema de Arquivo de Rede com Soquetes

1. Introdução

Este relatório tem como objetivo explicar o código do servidor e do cliente desenvolvido para implementar um sistema de arquivo de rede mínimo utilizando soquetes em Python. O sistema permite que um cliente se conecte a um servidor e execute diversas funcionalidades, como criar diretórios, remover diretórios, listar conteúdo de diretórios, enviar arquivos e remover arquivos.

2. Funcionalidades do Sistema

O sistema de arquivo de rede possui as seguintes funcionalidades:

2.1. Criar diretório

Permite que o cliente crie um novo diretório no servidor remoto.

2.2. Remover diretório

Permite que o cliente remova um diretório do servidor remoto, incluindo todos os arquivos e diretórios contidos dentro dele.

2.3. Listar conteúdo de diretório

Permite que o cliente obtenha a lista de arquivos e diretórios presentes em um diretório específico no servidor remoto.

2.4. Enviar arquivo

Permite que o cliente envie um arquivo para o servidor remoto, armazenando-o em um diretório específico.

2.5. Remover arquivo

Permite que o cliente remova um arquivo específico do servidor remoto.

3. Código do Servidor

O código do servidor é responsável por aguardar a conexão com o cliente e processar as requisições recebidas. Ele utiliza a biblioteca socket para criar um soquete TCP/IP e esperar por conexões. Abaixo está uma visão geral do código do servidor:

O servidor começa criando um soquete TCP/IP usando socket.socket(socket.AF_INET, socket.SOCK_STREAM).

Em seguida, define o endereço IP e a porta em que o servidor ficará escutando as conexões.
O servidor chama o método bind() para associar o endereço IP e a porta ao soquete.
Após a associação, o servidor chama o método listen() para aguardar por conexões de clientes.
O servidor entra em um loop infinito, aceitando conexões entrantes usando o método accept().
Para cada conexão aceita, o servidor recebe a requisição do cliente usando o método recv() e processa a requisição de acordo com a funcionalidade solicitada.
O servidor envia a resposta de volta para o cliente usando o método send().
Por fim, o servidor fecha a conexão com o cliente usando o método close().

4. Código do Cliente

O código do cliente é responsável por se conectar ao servidor e enviar as requisições. Ele também utiliza a biblioteca socket para criar um soquete TCP/IP e se comunicar com o servidor. Aqui está uma visão geral do código do cliente:

O cliente começa criando um soquete TCP/IP usando socket.socket(socket.AF_INET, socket.SOCK_STREAM).
O cliente solicita ao usuário o endereço IP e a porta do servidor.
Em seguida, o cliente chama o método connect() para se conectar ao servidor usando o endereço IP e a porta fornecidos.
O cliente exibe um menu de opções para o usuário selecionar a funcionalidade desejada.
Com base na opção selecionada, o cliente solicita as informações adicionais necessárias, como nome do diretório, caminho do arquivo, etc.
O cliente envia a requisição ao servidor usando o método send() e aguarda a resposta usando o método recv().
O cliente exibe a resposta recebida na tela.
O processo se repete até que o usuário escolha a opção para sair do programa.

5. Executando o Sistema

Para executar o sistema, siga as seguintes etapas:

5.1. Executando o Servidor

Certifique-se de ter o código do servidor disponível em um arquivo Python.
Abra um terminal ou prompt de comando e navegue até o diretório onde o arquivo do servidor está localizado.
Execute o comando python servidor.py para iniciar o servidor.
O servidor agora está aguardando por conexões de clientes.

5.2. Executando o Cliente

Certifique-se de ter o código do cliente disponível em um arquivo Python.
Abra um terminal ou prompt de comando e navegue até o diretório onde o arquivo do cliente está localizado.
Execute o comando python cliente.py para iniciar o cliente.
O cliente solicitará o endereço IP e a porta do servidor. Insira as informações solicitadas.
O cliente exibirá um menu de opções. Selecione a opção desejada digitando o número correspondente e siga as instruções adicionais conforme necessário.
O cliente enviará a requisição ao servidor e exibirá a resposta recebida na tela.
O processo se repetirá até que você escolha a opção para sair do programa.

6. Conclusão

O sistema de arquivo de rede mínimo implementado utilizando soquetes em Python permite que um cliente execute diversas operações em um servidor remoto. O código do servidor aguarda por conexões de clientes, enquanto o código do cliente se conecta ao servidor e envia as requisições. Ambos os códigos são complementares e devem ser executados em conjunto para que o sistema funcione corretamente.