Lista de Exercícios Avaliativa sobre Sockets UDP e TCP.

Introdução

Esta documentação descreve como foi implementado o envio e o recebimento do tamanho do arquivo em um sistema cliente-servidor utilizando sockets UDP. Este processo é essencial para garantir que o cliente saiba o tamanho total do arquivo antes de começar a recebê-lo, permitindo controlar adequadamente o fluxo de dados e validar a integridade da transferência.

Objetivo

O objetivo desta funcionalidade é permitir que o cliente:

1. Receba o tamanho total do arquivo do servidor antes de iniciar o recebimento dos dados.
2. Utilize essa informação para saber quando parar de receber os dados e garantir que o arquivo seja transferido corretamente.

Implementação no Servidor

No servidor, foram realizadas as seguintes etapas para implementar o envio do tamanho do arquivo:

1. Obtenção do Tamanho do Arquivo

Utilizou-se a função os.path.getsize() para obter o tamanho do arquivo em bytes:

```
# Obtendo o tamanho do arquivo
tamanho_arquivo = os.path.getsize(DIRBASE + fileName)
print(f"Tamanho do arquivo {fileName}: {tamanho_arquivo} bytes")
```

2. Envio do Tamanho ao Cliente

O tamanho do arquivo foi convertido para string e enviado ao cliente antes do envio dos dados do arquivo:

```
# Enviando o tamanho do arquivo ao cliente
sock.sendto(str(tamanho_arquivo).encode(), source)
print(f"Tamanho do arquivo {tamanho_arquivo} enviado para o cliente {source}")
#-----------------------------------------------------
```

3. Tratamento de Erros

Caso o arquivo não seja encontrado, o servidor envia `0` ao cliente para indicar que o arquivo não existe:

```
except FileNotFoundError:
    #-------------------------------------------------------
    # Enviando mensagem de erro
    print(f"Arquivo {fileName} não encontrado.")
    sock.sendto(b'0', source)
    #-------------------------------------------------------
```

Implementação no Cliente

No cliente, as seguintes etapas foram realizadas para receber e utilizar o tamanho do arquivo:

1. Recebimento do Tamanho do Arquivo

O cliente aguarda a resposta do servidor com o tamanho do arquivo antes de iniciar a gravação dos dados:

```
#-------------------------------------------------------
# Recebendo o tamanho do arquivo
tamanho_arquivo_bytes, source = sock.recvfrom(4096)
tamanho_arquivo = int(tamanho_arquivo_bytes.decode())
```

2. Verificação do Tamanho

Caso o tamanho recebido seja 0, o cliente informa que o arquivo não foi encontrado:

```
if tamanho_arquivo == 0:
    print(f"Arquivo {fileName} não encontrado no servidor.")
    continue

print(f"Tamanho do arquivo {fileName} recebido: {tamanho_arquivo} bytes.")
#-------------------------------------------------------
```

3. Controle do Recebimento

Com base no tamanho recebido, o cliente controla o fluxo de dados e sabe quando parar de receber:

```
bytes_recebidos = 0
while bytes_recebidos < tamanho_arquivo:
    data, source = sock.recvfrom(4096)
    fd.write(data)
    bytes_recebidos += len(data)
    print(f"Recebidos {bytes_recebidos}/{tamanho_arquivo} bytes...")

fd.close()
print(f"Arquivo {fileName} recebido e salvo com sucesso!\n")
```

Benefícios da Implementação

1. Controle do Fluxo: O cliente pode calcular quanto já recebeu e quanto falta receber.
2. Indicação de Erros: Caso o arquivo não seja encontrado, o cliente é informado imediatamente e não fica esperando.
3. Integridade: Permite que o cliente valide o tamanho total recebido, garantindo a integridade dos dados