Universidade Federal Do Paraná

Sistema de Replicação de Arquivos

Pedro Henrique Gurski de Oliveira | pedrogurski@ufpr.br
Ulisses Curvello Ferreira | ulissesferreira@ufpr.br

Curitiba, 29 de novembro de 2025

Disciplina: CI1061 BCC2 - Redes de Computadores II

Professor: Giovanni Venâncio de Souza

1 Introdução

Este trabalho é sobre um sistema de replicação de arquivos baseado em sistemas cliente-servidor. O objetivo é que clientes possam enviar arquivos (e.g., .txt, .pdf, etc.) para um servidor primário, e este servidor será responsável por replicar este arquivo entre múltiplos servidores denominados de réplicas. A figura abaixo ilustra o funcionamento do sistema.

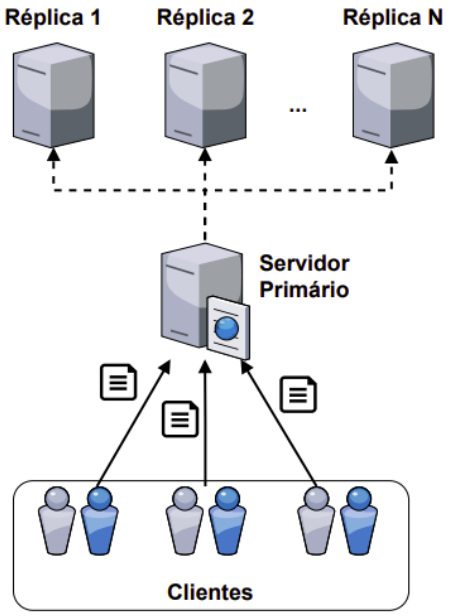

Figura 1: Funcionamento do sistema

O sistema é composto por três tipos de processos:

1. Cliente. Conecta-se com o servidor primário e envia diferentes tipos de comandos.
2. Servidor primário. Responsável pela comunicação com o cliente, armazenamento local de arquivos, e replicação dos arquivos para os servidores réplica.
3. Servidores réplica. Mantém cópias dos arquivos do servidor primário.

2 Decisões de Projeto e Protocolo

Para atender aos requisitos de funcionamento com múltiplos clientes e integridade de dados binários, foram tomadas as seguintes decisões de projeto:

2.1 Protocolo de Aplicação

Foi definido um protocolo de camada de aplicação simples para gerenciar a comunicação e o envio de metadados antes do conteúdo binário dos arquivos.

Cada transação inicia com o envio de um **cabeçalho de tamanho fixo** (1024 bytes). O cabeçalho é uma string com os campos separados por ('|'). A estrutura do cabeçalho é:

`ID_CLIENTE | TIPO_OPERACAO | NOME_ARQUIVO | TAMANHO_BYTES`

Isso permite que o servidor saiba quantos bytes ler do socket e onde salvar o arquivo antes de começar a receber a mensagem.

2.2 Gerenciamento de Múltiplos Clientes

Para suportar M clientes diferentes, implementamos:

1. **Concorrência:** O Servidor Primário utiliza *multithreading* (biblioteca threading). Para cada nova conexão aceita (accept), uma nova thread é disparada, permitindo que múltiplos uploads ocorram simultaneamente sem bloqueio.

2. **Isolamento de Dados:** No sistema de arquivos do servidor e das réplicas, os arquivos são salvos em subdiretórios nomeados com o ID do cliente (ex: `./data/1/arquivo.txt`). Pois, quando fazer um 'list', conseguimos obter apenas os arquivos do cliente específico.

2.3 Consistência e Replicação

A estratégia de replicação escolhida foi a **Replicação Síncrona Sequencial**. Ao receber um arquivo, o Servidor Primário:

1. Salva o arquivo em seu disco local.

2. Itera sobre a lista de réplicas configuradas.

3. Abre uma conexão TCP com a Réplica N, envia o arquivo, aguarda a confirmação ('OK') e fecha a conexão.

4. Repete para a próxima réplica.

O cliente só recebe a confirmação de sucesso após todas as réplicas terem confirmado a gravação.

3 Detalhes de Implementação

3.1 Transferência de Arquivos

Para arquivos de qualquer formato (txt, png, PDF), a leitura e escrita foram implementadas em modo binário (‘rb‘/‘wb‘). A transferência é realizada em blocos (chunks) de 4096 bytes. A lista de réplicas (IP e Porta), por padrão temos 3, mas caso tenha mais, adicionar manualmente no código ’servidor.py’ antes da execução.

4 Execução

Para executar o sistema, abra vários terminais no mesmo diretório onde estão os arquivos fonte. A execução mostrada no terminal 3 abaixo vale da mesma forma para os demais clientes.

Iniciar as Réplicas

```
# Terminal 1
python3 replica.py 9000
```

Iniciar o Servidor Primário

```
# Terminal 2
python3 servidor.py 8500
```

Executar o Cliente

```
# Terminal 3
python3 cliente.py 1 127.0.0.1 8500
```

- **Enviar arquivo**

```
# Terminal 3
> upload teste.txt
```

- **Listar arquivos no servidor**

```
# Terminal 3
> list
```

- **Sair do cliente**

```
# Terminal 3
> exit
```