Proyecto 1 P2P - Sistemas Distribuidos Documentation

version 1.0.0

2024, Samuel Pérez Hurtado

September 22, 2024

Contenido

Documentación Proyecto 1 P2P - Sistemas Distribuidos

Introducción

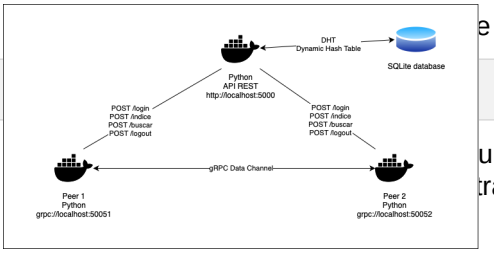
Este proyecto consiste en la implementación de un sistema P2P (Peer-to-Peer) para la compartición de archivos entre nodos (peers) utilizando microservicios, API REST, y gRPC. La arquitectura está diseñada para permitir la comunicación entre múltiples nodos, donde cada nodo actúa como cliente y servidor simultáneamente, permitiendo la consulta y descarga de archivos.

El sistema se despliega utilizando Docker y se basa en microservicios definidos para soportar concurrencia, permitiendo que múltiples procesos remotos se comuniquen con los peers simultáneamente. La comunicación entre nodos se realiza tanto mediante API REST como gRPC, con servicios definidos para la localización de archivos y transferencia simulada de estos (ECO/DUMMY).

Arquitectura del Sistema

El sistema está diseñado con una red P2P no estructurada basada en un servidor de Directorio y Localización. Los nodos (peers) pueden conectarse entre sí de manera descentralizada. La arquitectura del sistema incluye los siguientes componentes clave:

1. **Peer Cliente (PCliente):** Encargado de enviar solicitudes para localizar y descargar archivos.
2. **Peer Servidor (PServidor):** Responde las solicitudes de otros peers proporcionando información sobre los archivos almacenados y simula la transferencia de archivos mediante servicios DUMMY.



e facilita la localización de archivos y recursos en los distintos nodos.

uida donde cada peer se comunica con otros peers a través de API REST y traliza la información de localización de recursos.

Componentes del Proyecto

Los componentes principales del proyecto están estructurados en varias capas, cada una con responsabilidades específicas.

1. **peer_client.py:** Implementa la lógica del cliente, permitiendo a los peers enviar solicitudes a otros peers para buscar y descargar archivos.
2. **peer_server.py:** Define el servidor gRPC que escucha las solicitudes entrantes de otros peers y responde con información sobre los archivos almacenados en el nodo.
3. **file_transfer.proto:** Define el protocolo gRPC utilizado para la comunicación entre peers, especificando los mensajes y servicios.
4. **app.py:** Implementa el microservicio de API REST que facilita la localización de archivos en el sistema.
5. **docker-compose.yml:** Archivo de configuración para el despliegue de contenedores Docker que define la estructura de red y servicios del sistema P2P■17†source■.
6. **Dockerfile:** Define el entorno y las dependencias necesarias para construir y ejecutar los contenedores Docker que albergan los peers.

Descripción de los Microservicios

Peer Cliente (PCliente)

El peer cliente es responsable de enviar solicitudes de localización de archivos a otros peers utilizando API REST o gRPC. Implementa la funcionalidad para interactuar con otros peers, permitiendo la descarga simulada de archivos.

Peer Servidor (PServidor)

El peer servidor responde las solicitudes entrantes de otros peers, proporcionando una lista de archivos disponibles en su directorio y simulando la transferencia de archivos mediante servicios DUMMY. La transferencia real no se implementa, pero la lógica para realizar un «eco» de carga y descarga de archivos está presente.

API REST

El servidor API se comunica con los peers a través de una API REST, actuando como un directorio centralizado que ayuda a los peers a localizar recursos en la red.

Configuración del Entorno Virtual

Para el desarrollo del proyecto, se requiere configurar un entorno virtual en Python. A continuación, se detallan los pasos necesarios:

1. **Crear el entorno virtual:**

   ```
   python3 -m venv venv
   ```

2. **Activar el entorno virtual:**

   - En Linux/Mac:

     ```
     source venv/bin/activate
     ```

   - En Windows:

     ```
     .\venv\Scripts\activate
     ```

Instalación de Dependencias

Las dependencias del proyecto están listadas en el archivo `requirements.txt`. Para instalarlas, ejecute el siguiente comando dentro del entorno virtual activo:

```
pip install -r requirements.txt
```

Este archivo incluye las librerías necesarias para el funcionamiento del proyecto, como `grpcio` y `Flask` para la implementación de gRPC y API REST, respectivamente.

gRPC y REST API

El proyecto utiliza gRPC para la comunicación entre peers y REST API para la localización de recursos.

gRPC

El archivo `file_transfer.proto` define los servicios y mensajes utilizados en gRPC. El servidor gRPC implementa los servicios de localización y transferencia simulada de archivos, mientras que los clientes (otros peers) envían solicitudes de descarga de archivos.

REST API

La API REST actúa como un directorio donde los peers pueden consultar información sobre los archivos almacenados en otros nodos. Utiliza Flask para manejar las solicitudes HTTP y devolver respuestas en formato JSON.

Ejecución del Proyecto

Para ejecutar el proyecto, asegúrese de tener Docker instalado y siga los siguientes pasos:

1. **Construir los contenedores Docker:**

   ```
   docker-compose build
   ```

2. **Levantar los servicios:**

   ```
   docker-compose up
   ```

Esto iniciará los contenedores para los peers y el servidor API. Los peers estarán escuchando en los puertos 50051 y 50052, mientras que el servidor API estará disponible en el puerto 5000■17†source■.

Pruebas y Despliegue en Docker

El proyecto se prueba localmente utilizando Docker. La configuración de Docker permite levantar múltiples peers en contenedores separados. El archivo `docker-compose.yml` define la estructura de red y los puertos expuestos para cada servicio.

Configuración de la Red P2P

El archivo `docker-compose.yml` define una red `p2p_network` que conecta los contenedores de los peers y el servidor API. Cada peer tiene un volumen específico donde se almacenan los archivos locales que serán compartidos entre los nodos.