FTP Distribuido basado en Kademlia

Alejandro Lamelas Delgado
Dario Rodriguez Llosa

January 15, 2025

Introduccin

En este documento se expone el diseo de un File Transfer Protocol (FTP) distribuido que emplea arquitectura Kademlia para almacenar y recuperar archivos.

Arquitectura

- Organizacion del Sistema Distribuido:
  El sistema se vasa en la arquitectura Kademlia donde cada nodo tiene identificador unico (hash de su direccin IP). Los archivos se identifican mediante el hash de su path, al igual que las carpetas. Esta estrategia asegura unicidad, independientemente del nombre de los archivos.
- Roles del Sistema:
  Tendremos en nuestro sistema dos tipos de nodos, los nodos de archivos y los nodos de carpetas, la diferencia entre estos sera la forma que tendran de guardar la informacin que contrendran, ya que los nodos de archivos guardaran los archivos, mientras los nodos de carpetas guardaran las key de lo que contrendra dicha carpeta que pudiera ser otras carpetas y/o archivos, al igual que sera distinto la forma de descarga, lo cual hace necesario esta diferencia.
- Distribucion de Servicios en Redes Docker:
  Empleamos dos redes Docker separadas una para clientes y otra para servidores, interconectadas por un router. La red de servidores aloja los nodos de Kademlia, gestionando la comunicacin directa entre ellos para el mantenimiento de la estructura, enrutamiento de claves y almacenamiento de archivos. La red de clientes provee acceso independiente a los nodos, usando el router como intermediario.

Procesos

Tanto el cliente como los nodos servidor manipularan mltiples procesos concurrentes utilizando hilos.

- Cliente: Las operaciones de subida y descarga se ejecutan en hilos independientes, gestionando eventos de manera asncrona, lo que permite a los usuarios realizar otras acciones simultneamente.

- Servidor: Un hilo principal se encarga de iniciar el servidor, aceptar conexiones entrantes y redirigir solicitudes a hilos secundarios. Hilos dedicados manejan de forma independiente las solicitudes de clientes, como subidas y descargas, mientras que otros hilos gestionan la comunicacin entre nodos. Este modelo de hilos permite un procesamiento concurrente ms eficiente.

Comunicacin

Utilizaremos RPC para la comunicacin entre los nodos en Kademlia ya que seria una forma sencilla de realizar todas las operaciones.La comunicacin entre cliente y servidor se hara mediante socket usando tcp.

Coordinacin

- Acceso Exclusivo a Recursos: Para garantizar la integridad y consistencia al manipular archivos dentro de un nodo, se implementa el uso de threading.Lock en Python. Este mecanismo asegura que los recursos compartidos solo sean accedidos por un hilo a la vez, previniendo conflictos como condiciones de carrera, corrupcin de datos y errores de concurrencia, esenciales en entornos de sistemas distribuidos

- Toma de Decisiones Distribuidas: La responsabilidad de la toma de decisiones se delega de forma autnoma a cada nodo. Este enfoque descentralizado no solo simplifica el proceso global al eliminar la dependencia de un controlador central, sino que tambin mejora la eficiencia y escalabilidad del sistema, permitiendo una mayor adaptabilidad ante cambios en la red o en la carga de trabajo.

Nombrado y Localizacin

- Identificacin de Datos y Servicios:
  Los archivos se identifican de manera nica mediante el clculo de su hash SHA-256, lo que asegura su integridad y facilita su distincin en el sistema. Los nodos, por su parte, se identifican a travs del hash de su direccin IP, lo que permite una asignacin precisa de responsabilidades. Los nombres de archivo y sus extensiones se almacenan en metadatos, los cuales son indexados utilizando el hash del nombre del archivo, optimizando el proceso de bsqueda y recuperacin de informacin.

- Ubicacin de Datos y Servicios:
  La localizacin de archivos se realiza mediante el hash del contenido de los

mismos, lo que permite dirigir la bsqueda hacia el nodo responsable de la clave correspondiente. Adems, los metadatos desempean un papel crucial al facilitar la localizacin de la clave asociada al contenido, ya sea a partir del nombre o de la extensin del archivo, mejorando as la eficiencia en el acceso a los recursos.

Replicacion, consistencia y tolerancia a fallas

Una de las ventajas que trae Kademlia es que tanto la tolerancia a fallas, la forma de localizar los datos, la caida de nodos y la distribucion es que ya trae soluciones a estos problemas y en cuanto a las replicas, tenemos pensado realizar copias de un archivo a los 3 nodos mas cercanos de su hash key, asi podremos tener nivel 2 de tolerancia a fallos.