Facultad de Ingeniería de Sistemas e Informática
Arquitecturas de nube y Sistemas distribuidos
Proyecto 2 – mayo 8 de 2025
Fecha de entrega: mayo 25 de 2025 GMT-5

Estudiante: ID:

SISTEMA DE ARCHIVOS DISTRIBUIDOS POR BLOQUES

Objetivo: Diseñar e implementar un sistema de archivos distribuidos por bloques minimalista.

Descripción: Un sistema de archivos distribuidos, permite compartir y acceder de forma concurrente un conjunto de archivos que se encuentran almacenados en diferentes nodos. Uno de los sistemas más maduros, vigente y antiguo de estos sistemas es el NFS (Network File System) desarrollado en su momento por Sun Microsystems y que hoy en día es ampliamente usado en sistemas Linux. Hay otros sistemas de archivos distribuidos como AFS (Andrew File System) y SMB (Server Message Block) conocido como CIFS.

En general hay dos acercamientos para el diseño e implementación de un DFS: 1) basado en bloques y basado en objetos.

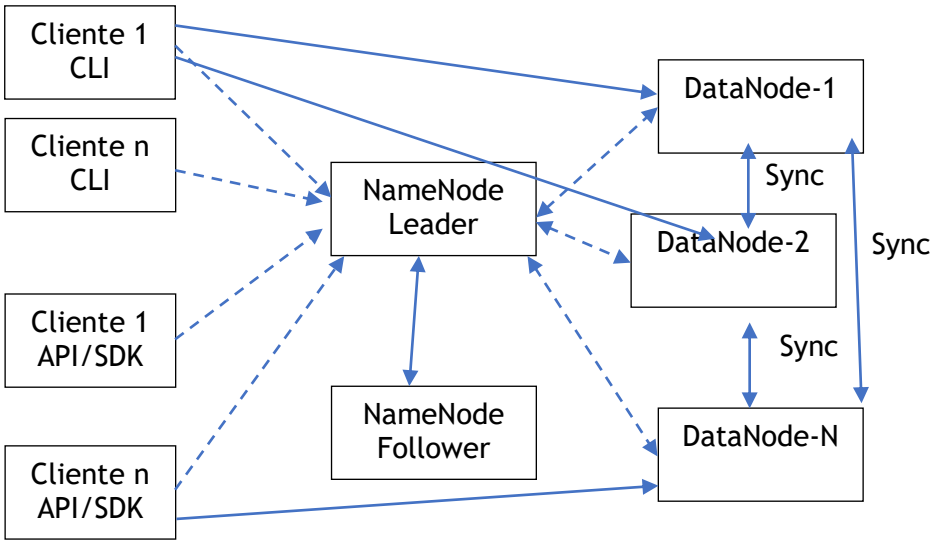
Los DFS basados en bloques generalmente garantizan 2 aspectos: 1) la unidad de escritura y lectura es a nivel de bloque, y los bloques pueden ser distribuidos en diferentes nodos, la idea es que los bloques de un archivo estén distribuidos en un conjunto de nodos. 2) el sistema operativo cliente de un DFS garantiza transparencia en el sentido de que la API ofrecida desde el SO es igual para acceder archivos locales que remotos, porque el DFS se integra con el sistema de gestión de archivos del sistema operativo (ej: NFS, AFS, SMB, etc).

Los DFS basados en objetos (object storage, ej: AWS S3), los dos aspectos anteriores se manejan así: 1) la unidad de distribución es a nivel de archivo y no de bloque, es decir, y se garantiza que se lee o escribe un archivo como un todo y no a nivel de bloque. No está diseñado como un sistema de acceso aleatorio al archivo sino secuencial. No soporta la operación de actualización parcial del archivo, sino que se debe reemplazar todo el archivo. Son sistemas distribuidos de archivos principalmente diseñado para un enfoque WORM (Read-Once-Read-Many). Típicamente estos DFS soporta altos niveles de escalabilidad, redundancia y rendimiento. Si bien desde el cliente tiene una visión de archivo completo, en el sistema de backend podría tener (y normalmente lo hay) un mecanismo de particionamiento del archivo por bloques u otro criterio para mejorar la escalabilidad, tolerancia a fallos y rendimiento. 2) el sistema operativo local del cliente NO integra directamente la gestión de este DFS y en vez de ello se cuenta con un SDK o API para las diferentes primitivas de la gestión de archivos y normalmente tienen su propio CLI.

A nivel de recomendación para este proyecto2, realizaremos el diseño e implementación del DFS intermedio principalmente orientado a bloques, pero con la característica de WORM del almacenamiento por Objetos. Este tipo de DFS es el enfoque de sistemas de archivos como GFS y HDSF. Lo primero que deberá hacer el equipo de trabajo es leer y comprender los papers fundacionales de estos DFS:

- GFS:
  - https://es.wikipedia.org/wiki/Google_File_System
  - The Google File System - https://g.co/kgs/XzwmU76
- HDFS:
  - https://es.wikipedia.org/wiki/Hadoop_Distributed_File_System
  - The Hadoop Distributed File System – link

A nivel minimalista de este DFS, se propone la siguiente arquitectura:



Especificaciones:

- Se tendrán dos tipos de protocolos o comunicaciones entre procesos donde debe emplear (REST API y gRPC),:
  - Canal de Control:
  - Canal de Datos: 
- La escritura y lectura de los archivos, debe ser directamente realizado entre el Cliente y el DataNode. Debe definir un algoritmo para distribución de los bloques y su replicación.
- Cada archivo debe ser particionado en *n* bloques que se distribuyen por los datanodes (obviamente referenciados por namenode). Se deja opcional si se pueden cambiar los tamaños de los bloques en la configuración inicial.
- La unidad mínima de replicación – por facilidad – se tomará como un bloque.
- Un bloque al menos debe estar en dos DataNode, se debe garantizar en todo momento esta replicación de bloques.
- La transferencia de un archivo se hace desde cada uno de los Datanodes que contengan bloques principales o replicas. Por facilidad y producto mínimo viable, el namenode entrega al cliente la lista y el orden donde se encuentran los bloques de un archivo (lista de bloques y URI).
- A nivel de escritura de un archivo en el sistema, se realizará la transferencia directa entre el cliente y un grupo de DataNode seleccionado con un criterio de optimización del NameNode para elegir los DataNodes más adecuado de acuerdo a alguna métrica.
- Un DataNode que recibe un bloque de un Cliente se convierte en un Leader del bloque y este será encargado de replicar a otro DataNode el bloque de este archivo para Tolerancia a fallos, este segundo DataNode lo conoceremos como Follower para este archivo.
- Se debe implementar a nivel de cliente una interfaz de comandos (CLI) que implemente mínimo los siguientes comandos:
  - ls, cd, put, get, mkdir, rmdir, rm, etc
- (Opcional): Cada cliente puede solo ver y manipular sus propios archivos, para ello implementará la autenticación básica user/pass, muy básica.

Entregables:

- Documento con el Diseño detallado y especificación de los servicios.
- Códigos y validación: Implementación (códigos) y ejecución (pantallazos de funcionamiento)
- Plantilla de autoevaluación y requisitos del proyecto (**Sin autoevaluación se considera un 80% de alcance del proyecto)**
- Video Sustentación: donde participen todos los integrantes del grupo donde explique el proceso de

diseño, desarrollo y ejecución (no más de 30 mins).

(Este enunciado se seguirá actualizando de acuerdo con aclaraciones y retroalimentación recibida, los cambios o adiciones serán resaltados)

Éxitos !!!