Informe. Implementación Distribuida. Bittorrent

1. Arquitectura o el Problema de Diseñar el Sistema

Organización de su Sistema Distribuido

El sistema BitTorrent se organiza como una red descentralizada donde los nodos (peers) interactúan para compartir datos. Cada nodo puede actuar como cliente y servidor simultáneamente, promoviendo la colaboración entre los participantes.

Roles de su Sistema

- **Seeder**: Nodo que posee una copia completa del archivo y lo comparte con otros nodos.
- **Leecher**: Nodo que está descargando partes del archivo y también comparte las partes ya descargadas.
- **Tracker**: Servicio que facilita la comunicación entre los peers al mantener información sobre los nodos activos.
  -> Nota: A su vez tanto los seeders como los leechers se conocen como peers.

Distribución de Servicios en Ambas Redes de Docker

- **Red 1: Tracker**
  - Contenedor dedicado para el tracker que almacena y facilita la información de los peers.
- **Red 2: Peers (Seeders y Leechers)**
  - Múltiples contenedores ejecutando instancias de clientes BitTorrent, actuando como seeders o leechers.

2. Procesos o el Problema de Cuántos Programas o Servicios Posee el Sistema

Peers (Seeders y Leechers)

- Los peers funcionan mediante hilos para gestionar la descarga y subida de archivos. Cuando un peer nuevo entra a la red hay dos casos posibles:

- Si es un seeder, es el caso más sencillo. Solo se encarga de compartir el archivo a la red. Este seeder abre un hilo nuevo por cada conexión entrante, es decir por cada peer que solicita el archivo.
- Si es un leecher, tendría dos hilos (O procesos, se probará para ver mayor eficiencia), uno para descarga y otro para subida. A su vez, cada uno de esos hilos maneja cada conexión de manera independiente en otro hilo.

Tracker

El Tracker funciona como un servicio independiente y maneja cada conexión entrante con un hilo independiente.

3. Comunicación o el Problema de Cómo Enviar Información Mediante la Red

Toda la comunicación se realiza mediante **Sockets**: Para la comunicación directa entre peers o con el tracker.

Comunicación

Existen dos tipos de comunicación:

- Cliente-Tracker: Para crear el torrent y registrarse como peer. También para obtener todos los peers de un archivo torrent.
- Peer-Peer: Luego de tener la información de un archivo torrent ya no es necesario mantener conexión con el tracker. Con los datos anteriores los peers pueden mantener la conexión entre ellos.

4. Coordinación o el Problema de Poner Todos los Servicios de Acuerdo

Sincronización de Acciones

- Las descargas utilizan un "Bitfield" que funciona como una máscara booleana, para tener registro de las partes descargadas de cada archivo. En el momento de que un peer comienza a descargar un archivo se busca en su Bitfield para ver cuáles partes faltan, en este proceso se usan locks para evitar descargar la misma parte más de una vez.

Toma de Decisiones

- Selección de piezas basada en algoritmos de rareza para optimizar la distribución de las piezas. Osea, a la hora de descargar una pieza se elige cual es el "mejor" peer para llevar a cabo el proceso.

5. Nombrado y Localización

Identificación de los Datos y Servicios

- Los archivos .torrent tienen un hash que funciona como identificador único.
- Los peers también tienen un identificador único, además de su IP.

6. Consistencia y Replicación

- División del archivo, o los diferentes archivos mediante los peers que lo contengan. En caso de que no exista peer activo no hay forma de descargar el archivo.
- Verificación de integridad de cada archivo y de cada parte mediante hashes SHA-1.

Tracker Distribuido

Se distribuirá la información del tracker para que sea tolerante a fallas. La idea es una implementación con **Chord** teniendo la base de datos distribuida, de esta manera los peers siempre podrían obtener la información de los torrents.

Otro enfoque (Tracker vs DHT)

- En lugar de depender de un tracker, los peers se unen a la DHT, anunciando su disponibilidad y las piezas que poseen.
- Cada vez que un peer necesita descargar un archivo, busca en la DHT el hash correspondiente para obtener la lista de nodos (peers) que poseen partes del archivo.
- Esto distribuye completamente el rol del tracker entre todos los nodos, eliminando puntos únicos de fallo en el tracker.

7. Tolerancia a Fallas

Respuesta a Errores

- La comunicación entre los peers está definida de manera consistente y tolerante a fallos. Los peers se envían mensajes cada cierta cantidad de tiempo para verificar que

se mantiene activa la conexión, además, cada vez que se descarga una parte se verifica la integridad de la misma.
- En caso de que un peer se desconecte, los demás peers pueden continuar la descarga con otros peers que posean las partes faltantes.

Nivel de Tolerancia a Fallos Esperado

- Cada peer funciona de manera independiente, por lo que la caída de un peer no afecta a los demás.
- En el peor de los casos si caen todos los peers se pierde la posibilidad de descargar el archivo y se tendría que esperar a que al menos un peer se conecte para continuar la descarga del archivo.

8. Seguridad

- A modo general Bittorrent no es un protocolo seguro, sin embargo mediante la verificación de integridad de las partes descargadas se puede garantizar que no se descarguen archivos corruptos.