2da Entrega Sistemas Distribuidos

- Ariel González Gómez
- Alex Samuel Bas Beovides

1. Arquitectura

El sistema distribuido a desarrollar replica una versión básica de Twitter, permitiendo a los usuarios publicar posts, seguir a otros usuarios y autenticarse. Para la arquitectura nos basamos en una red de servidores y clientes que se comunican utilizando gRPC. Los servidores están organizados en un anillo de Chord para la distribución y almacenamiento de datos.

Organización del Sistema Distribuido

El sistema se organiza en dos redes de Docker: una para los clientes y otra para los servidores. Los clientes se encargan de interactuar con los usuarios finales, mientras que los servidores manejan la lógica de negocio y el almacenamiento de datos.

Roles del Sistema

El sistema se organiza en los siguientes roles clave:

1. **Cliente:**
    - Solicita la publicación de posts, seguimiento de usuarios o autenticación.
    - Localiza un servidor activo en la red y envía las solicitudes correspondientes.
2. **Servidor:**
    - Recibe y procesa solicitudes de los clientes.
    - Ofrece servicios de negocio como publicación de posts, manejo de relaciones entre usuarios y autenticación.

3. **Repositorio de Datos:**
    - Intermediario encargado de gestionar la lógica de almacenamiento y recuperación de datos.
    - Se comunica con los nodos del Chord Ring para realizar el almacenamiento distribuido.
4. **Nodo de Chord:**
    - Implementa la funcionalidad de la red Chord para distribuir y localizar los datos.
    - Cada nodo almacena datos según el hash calculado por la clave del post o usuario.

Distribución de Servicios en Redes Docker

1. **Red de Clientes:**
   Alojamiento de múltiples instancias de cliente que interactúan directamente con los usuarios finales, estas instancias no comparten información entre sí y dependen completamente de la red de servidores para las operaciones.
2. **Red de Servidores:**
   Contiene los contenedores que ejecutan los servicios del servidor, incluyendo el repositorio de posts y los nodos de Chord. Los servidores se comunican entre sí para mantener la consistencia y disponibilidad de los datos.

2. Procesos

Los principales procesos incluyen los clientes, que manejan las solicitudes de los usuarios; los servidores, que ejecutan servicios específicos como la publicación de posts, autenticación y gestión de relaciones; un repositorio de datos que actúa como intermediario; y los nodos del anillo Chord, encargados de distribuir y almacenar los datos hashados. Cada servicio opera de manera asíncrona, aprovechando hilos para manejar múltiples solicitudes concurrentes, y está diseñado bajo un patrón de diseño orientado a la escalabilidad y tolerancia a fallos, empleando redes de Docker separadas para aislar la comunicación entre clientes y servidores. Esta estructura permite un desempeño eficiente en entornos distribuidos al garantizar una interacción fluida y segura entre los componentes.

3. Comunicación

Para la comunicación cliente-servidor, se utiliza gRPC como protocolo principal, aprovechando sus capacidades de serialización eficiente y su modelo de comunicación. Esta elección facilita las operaciones síncronas como la autenticación de usuarios y la publicación de posts, permitiendo una interfaz clara y tipada para las interacciones.
La comunicación entre servidores dentro del anillo Chord se implementa mediante sockets, lo que proporciona un control más granular sobre la transferencia de datos y permite una gestión eficiente de la distribución de información entre los nodos. Los sockets permiten mantener conexiones persistentes entre los nodos del anillo, facilitando la propagación de actualizaciones y la sincronización del estado distribuido.
La comunicación entre procesos se gestiona a través del sistema de red de Docker, que proporciona el aislamiento necesario mientras permite una conectividad fluida entre los contenedores. Las redes Docker separadas para clientes y servidores garantizan una segregación adecuada del tráfico y mejoran la seguridad general del sistema.
Este diseño de comunicación híbrido permite aprovechar las fortalezas de cada protocolo: gRPC para interfaces estructuradas cliente-servidor y sockets para la comunicación eficiente entre nodos del anillo Chord, resultando en un sistema distribuido robusto y eficiente.

4. Coordinación

La coordinación entre los servicios en el sistema distribuido se manejará mediante mecanismos de sincronización y consenso para garantizar acceso exclusivo a recursos compartidos y evitar condiciones de carrera. El uso de algoritmos distribuidos, como el de exclusión mutua distribuida, asegurará que solo un nodo pueda modificar un recurso en un momento dado. Además, se implementará un protocolo de toma de decisiones distribuida para coordinar acciones críticas, como la actualización de datos o el manejo de fallos, permitiendo que los nodos alcancen acuerdos consistentes en el sistema de manera eficiente y confiable.

5. Nombrado y Localización

El sistema aborda el problema de localizar recursos mediante la integración del anillo Chord, que asigna cada recurso (usuarios, posts, servicios) a un nodo específico según un hash único. Los datos y servicios

se identifican por claves hash derivadas de identificadores únicos, garantizando una distribución uniforme. Para ubicar y acceder a los datos o servicios, el cliente realiza una solicitud inicial que se redirige a través del anillo Chord hasta el nodo correspondiente. Este enfoque asegura que la localización sea eficiente y escalable, incluso con cambios dinámicos en la red.

6. Consistencia y Replicación

La consistencia y replicación de datos en el sistema distribuido están diseñadas para abordar los desafíos inherentes a la existencia de múltiples copias de datos. El sistema utiliza un modelo de consistencia eventual, garantizando que todas las réplicas converjan hacia el mismo estado después de un tiempo, incluso si las actualizaciones se realizan de forma concurrente.

La distribución de los datos se basa en el anillo de Chord, donde los datos son asignados a nodos según el hash de su clave. Este enfoque asegura un balance adecuado de la carga y minimiza el riesgo de sobrecarga en nodos individuales.

Para garantizar alta disponibilidad, cada dato es replicado en k nodos consecutivos del anillo. El valor de k es configurable y se determina en función de los requisitos de confiabilidad del sistema, ya que para le entrega final nos piden un nivel de tolerancia 2, el valor de k debe ser 3. Esta replicación protege contra fallos de nodos individuales y asegura que las solicitudes puedan ser atendidas incluso si algunos nodos están fuera de servicio.

Al realizar una actualización, el nodo responsable de la clave (nodo primario) propaga los cambios a sus réplicas. Un protocolo de confirmación asegura que las actualizaciones sean reconocidas por una mayoría de réplicas antes de considerarlas completadas. Este enfoque fortalece la confiabilidad de los datos y garantiza que las actualizaciones sean consistentes a pesar de posibles fallos en los nodos o la red.

7. Tolerancia a fallas

En caso de que un nodo del anillo Chord falle temporalmente, su ausencia no detiene el funcionamiento del sistema. Los datos almacenados en dicho nodo se replican en nodos vecinos para garantizar su recuperación y

disponibilidad. Esto asegura que las solicitudes de clientes puedan ser atendidas incluso ante fallos parciales.
El sistema admite la incorporación de nuevos nodos en tiempo de ejecución. Cada nuevo nodo calcula su posición en el anillo Chord utilizando el hash correspondiente y redistribuye las claves relevantes de forma automática, minimizando la pérdida de datos y asegurando una transición suave.
Cuando un cliente intenta interactuar con un nodo inactivo, el sistema redirige automáticamente la solicitud a otro nodo disponible mediante la estructura del anillo Chord. Además, se implementan mecanismos de reintento y reconexión para mantener la confiabilidad del servicio.

8. Seguridad

gRPC proporciona una comunicación segura entre clientes y servidores mediante el uso de canales seguros (TLS/SSL), protegiendo la información de posibles ataques como la intercepción de datos (man-in-the-middle). El uso del anillo Chord garantiza una distribución descentralizada de datos, reduciendo puntos únicos de falla, aunque la seguridad del diseño depende de la validación de nodos participantes para evitar que nodos maliciosos comprometan el sistema. La segregación de redes Docker proporciona aislamiento del tráfico entre redes de clientes y servidores, minimizando la exposición de componentes críticos. El sistema utiliza autenticación de usuarios para garantizar que solo personas autorizadas puedan realizar acciones como publicar posts o seguir usuarios, fortalecida mediante el uso de tokens seguros como JWT. Para la autorización, los servidores verifican los permisos del usuario antes de ejecutar operaciones sensibles, reduciendo el riesgo de accesos indebidos.