Distributed Systems Pub/Sub
Feature Set

Asket Agarwal, Apurv Kumar, Shubham Sharma

Features :

- The system comprises of servers and clients (Publisher or Subscriber).
- Each client will be mapped to a unique front end server.
- There will be three types of server processes :
  - **Front-end servers** : These are directly connected to clients which can be publishers or subscribers. These servers acts as access points for clients to the system.
  - **Topic servers** : These servers store events pertaining to different topics and forward the new events to respective front-end servers after querying central server. These servers can act as replicated node for topics to load balance. This server stores backup for previous events which it flushes periodically (every week). This process and front-end server process can run on same node.
  - **Central server** : The central server handles the metadata about :
    - For each topic, the servers it has been replicated to.
    - Mapping of each topic server to the front-end server it serves to.

    The central server will have a primary backup. It acts as storage unit.
- It is a push model based system where the topic servers forward the new events it receives to the respective front-end servers which in turn pushes it to the subscriber.
- This system supports dynamic load balancing. It detects heavy load on a front-end server and starts a replicated topic server process to rebalance the load accordingly.

Workflow :

1. **Publisher :** When a client publishes an event for some topic x, the front-end server it is connected to queries from the central server all the replicated topic servers for topic x. After the central server replies to the front-end server, the front-end server forwards the publish event to all the replicated topic servers.

2. **New Subscriber :** When a client subscribes to some topic x, the front-end server it is connected to sends subscribe request to the central server.
   If the front-end server hasn't already sent subscribe request for topic x, then central server updates the metadata by assigning a topic server for topic x (such as to balance load for previous topic servers) to this front-end server. The central server requests the topic server assigned to send the backup events to the front-end server.

   If the front-end server has already sent a subscribe request for topic x, then the central server only requests the assigned topic server for topic x to send all events of topic x to the requesting front end server.

3. **Old Subscriber :** If a topic server gets a new event it pings the central server for the front-end servers it has to forward that event to. After getting a reply the topic server forwards the event to all the front-end servers which was in the reply. Those will in turn forward those events to the subscribers.

4. **Dynamic Load Balancing:** Whenever the number of subscribers for some topic x on a particular front-end server exceeds the threshold value, the front-end server then sends a replication request to the central server. The central server updates metadata by rebalancing the load for the new replicated topic server. The central server requests the topic server assigned to send the backup events to the front-end server.
   Whenever the number of subscribers for some topic x on a particular front-end server falls below the threshold value, the front-end server then sends a de-replication request to the central server. The central server updates metadata by rebalancing the load.