Zookeeper

A critical dependency of Apache Kafka is Apache Zookeeper, which is a distributed configuration and synchronisation service. Its serves as a co-ordinator interface between Kafka Broker, Producer and Consumer. Kafka stores the basic metadata in zookeeper, such as information about brokers, topics, partitions, partition leaders/followers, consumers, offset, etc.

Zookeeper as the central Coordinator

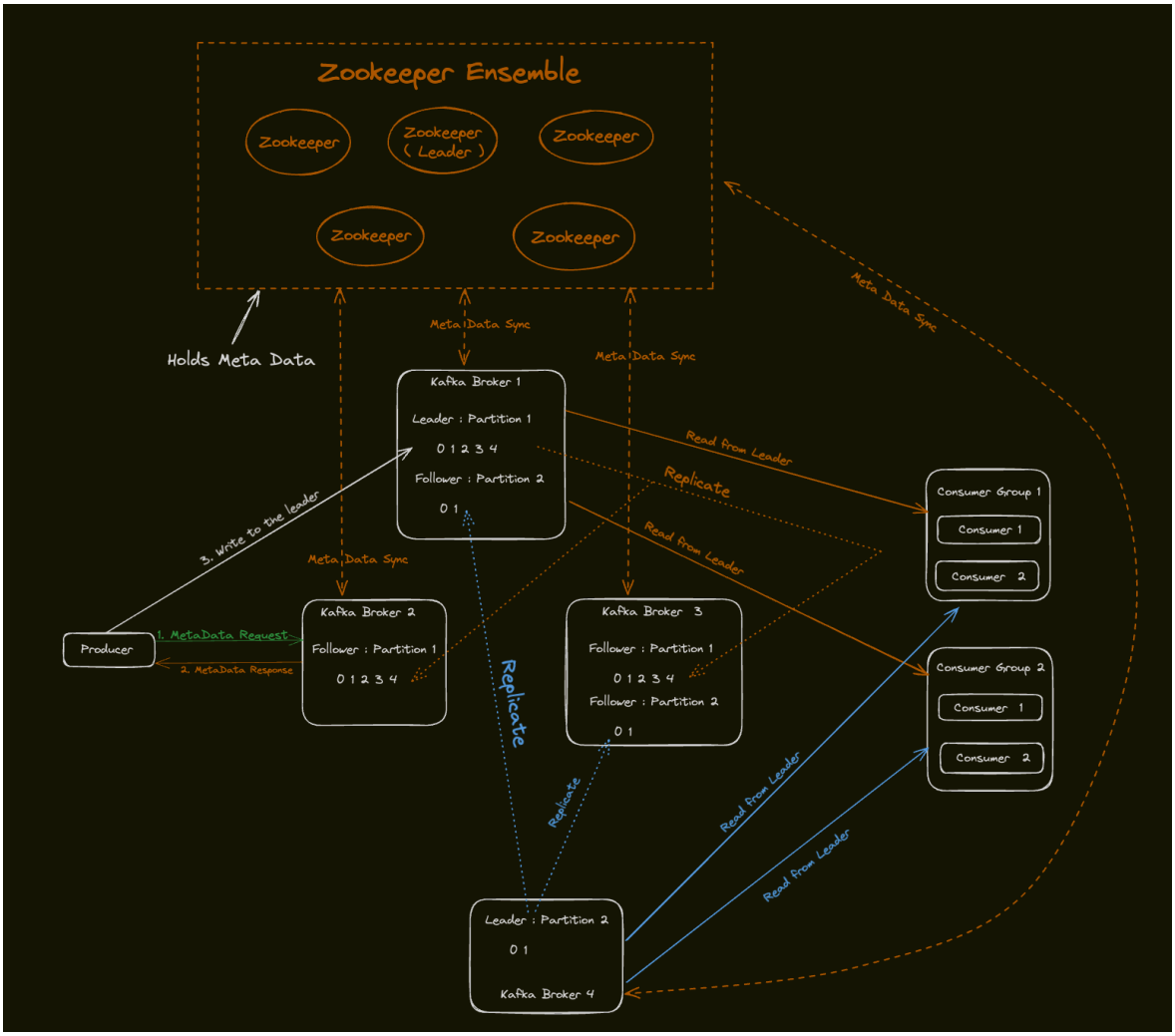
As we know, Kafka Broker are stateless, they rely on Zookeeper to maintain and coordinate brokers, such as notifying Consumer and Producer on the arrival of new Broker or failure of an existing Brokers, as well as routing request to Partition Leader.

Zookeeper is used for storing all sort of metadata about Kafka Cluster:

- It maintains the last offset position of each Consumer group per partition, so the consumer can quickly recover from the last position in case of failure [Although modern client save offset in a separate Kafka Topic]
- It tracks the Topics, number of partitions assigned to those topics, and leader/followers location in each position.
- It also manages the access control list [ACL] to different topics in a cluster. Access Control Lists are used to ensure access or authorisation.

How do Producers or Consumers find out who the leader of the partition is ?

In the older version of Kafka, all clients(i.e Producers and Consumers) used to directly talk to Zookeeper to find the partition leader. Kafka has moved away from this coupling , and in Kafka



Role of Zookeeper in Kafka

latest release, clients fetch metadata from Kafka Broker directly, Broker talk to Zookeeper to get the latest metadata.

In the diagram, the producer goes through following steps:
- Producer Connect to Kafka broker and ask for leader Partition.
- Kafka Broker responds the Leader Broker as Partition 1.
- The Producer connects to the leader broker to publish the message.

All the Critical Information is stored in the Zookeeper and Zookeeper replicates this data to its cluster, therefore failure of Kafka Broker(or Zookeeper itself) doesn't affect the state of Kafka Cluster. Upon Zookeeper failure , Kafka will always be able to store the state once the Zookeeper restarts after failure. Zookeeper is also responsible for coordinating the partition leader election between the Kafka Broker in case of leader failure.