Report

Apache Kafka is an open-source distributed event streaming platform used by thousands of companies for high-performance data pipelines, streaming analytics, data integration, and mission-critical applications. Kafka has come a long way since the 0.8 version we were running back then, and our tooling (some of it open source) has also significantly improved, increasing reliability and reducing the amount of operational work required to run our clusters. We can now scale clusters with a single configuration push, load balance, and decommission brokers, automatically trigger rolling-restarts to pick up new cluster configuration, and more.

Changes: One of the core values of Yelp infrastructure engineers is developer velocity: developers should be able to use the technology we provide with as little effort as possible. Providing Kafka as a service by hosting and maintaining Kafka clusters that other teams could directly access was our first approach and allowed us to quickly power many critical use cases.

Limitations: Clusters were getting big and difficult to maintain. A wide variety of critical and non-critical topics were sharing the same resources.

Advantages: 1) **Low Latency:** Apache Kafka offers low latency value, i.e., up to 10 milliseconds. It is because it decouples the message which lets the consumer consume that message anytime. 2) **High Throughput:** Due to low latency, Kafka is able to handle more messages of high volume and high velocity. Kafka can support thousands of messages in a second. Many companies such as Uber use Kafka to load a high volume of data. 3) **Fault tolerance:** Kafka has an essential feature to provide resistance to node/machine failure within the cluster.

Disadvantages: 1) **Do not have a complete set of monitoring tools:** Apache Kafka does not contain a complete set of monitoring as well as managing tools. Thus, new startups or enterprises fear to work with Kafka. 2) **Reduces Performance:** Brokers and consumers reduce the performance of Kafka by compressing and decompressing the data flow. This not only affects its performance but also affects its throughput. 3) **Clumsy Behavior:** Apache Kafka most often behaves a bit clumsy when the number of queues increases in the Kafka Cluster.

Applications: The demand for Apache Kafka is increasing at a tremendous speed. Many best enterprises today make use of Kafka to ease and grow their data pipelining requirements. Applications are LinkedIn, Twitter, Uber, etc.

Challenges: 1) In Sync Replica Alerts. 2) Kafka Liveness Check Problems and Automation. 3) New Brokers Can Impact the Performance. 4) Finding Perfect Data Retention Settings. 5) Overly Complex Data Transformations on-Fly. 6) Upscaling and Topic Rebalancing.