Kafka replication system tests

- A. Overview
- Design documentation
- B. Kafka Replication Testing Plan
 - B.1 Test Contract:
 - ° B.2 Test dimensions: Varying each parameters to provide different test scenario
 - ° C. Test Cases

A. Overview

According to Kafka Replication Design document, "The purpose of adding replication in Kafka is for stronger durability and higher availability. We want to guarantee that any successfully published message will not be lost and can be consumed, even when there are server failures. Such failures can be caused by machine error, program error, or more commonly, software upgrades."

1. Design documentation

a. https://issues.apache.org/jira/secure/attachment/12487175/kafka_replication_highlevel_design.pdf
 b. https://cwiki.apache.org/confluence/display/KAFKA/kafka+Detailed+Replication+Design+V3

B. Kafka Replication Testing Plan

B.1 Test Contract:

- 1. Produce and consume messages to x topics and y partitions.
- 2. This test sends m messages to n replicas.
- 3. At the end verifies the log size and contents as well as using a consumer to verify that there is no message loss.

B.2 Test dimensions: Varying each parameters to provide different test scenario

Parameter	Value Set
No. of partitions	1, 5, 10
No. of replica factors	1 ~ 6
Log segment sizes	1K, 2K, 10K
No. of topics	1, 5, 10, 100
Producer compression	On / Off
Producer acks	-1, 1
Producer mode	Sync, Async
Failure Type (Applicable in Failure Testcases)	 Controlled Failure (kill -15) Hard Failure (kill -9) Soft Failure (long pause during GC)

C. Test Cases

Functional Test	Description
C.1 Replication Basic	 Setup: Configure 1 Zookeeper, 1 ~ 6 brokers, 1 producer, 1 consumer Test Description: a. Follow the steps in B.1 Test Dimensions: Varying the parameters within the Value Set to observe the behavior of replication against different combinations Validation: a. Verify that all message id are matching in both producer log and consumer log b. Verify that all corresponding topic-partition log segment files checksums are matching across all replicas

C.2 Replication Leader Election	 Setup: Configure 1 Zookeeper, 1 ~ 6 brokers, 1 producer, 1 consumer Test Description: Follow the steps in B.1 During the test session, find leader from brokers' log4j message and introduce failure to Leader Leader re-election will be triggered Test Dimensions: Varying the parameters within the Value Set to observe the behavior of replication against different combinations Validation: Verify that new leader is re-elected by parsing the brokers' log4j messages log files
C.3 Replication with Leade r Failure	 Setup: Configure 1 Zookeeper, 1 ~ 6 brokers, 1 producer, 1 consumer Test Description: Follow the steps B.1 During the test session, find leader from brokers' log4j message and introduce failure to Leader The no. of failures can be specified in the corresponding testcase_<n>_properties.json</n> The type of failures are defined in Test Dimentions in B.2 (Controlled, Hard, Soft) Test Dimensions: Varying the parameters within the Value Set to observe the behavior of replication against different combinations Validation: Verify that all message id are matching in both producer log and consumer log Verify that all corresponding topic-partition log segment files checksums are matching across all replicas
C.4 Replication with Follow er Failure	 Setup: Configure 1 Zookeeper, 1 ~ 6 brokers, 1 producer, 1 consumer Test Description: Follow the steps in B.1 During the test session, find leader from brokers' log4j message and exclude that broker and introduce failure to one of the other brokers which are Followers. The no. of failures can be specified in the corresponding testcase_<n>_properties.json</n> The type of failures are defined in Test Dimentions in B.2 (Controlled, Hard, Soft) Test Dimensions: Varying the parameters within the Value Set to observe the behavior of replication against different combinations Validation: Verify that all message id are matching in both producer log and consumer log Verify that all corresponding topic-partition log segment files checksums are matching across all replicas
C.5 Replication with Contro ller Failure	 Setup: Configure 1 Zookeeper, 1 ~ 6 brokers, 1 producer, 1 consumer Test Description: Follow the steps in B.1 During the test session, find Controller from either brokers' log4j messages or querying the Bean and introduce failure to Controller. The no. of failures can be specified in the corresponding testcase_<n>_properties.json</n> The type of failures are defined in Test Dimentions in B.2 (Controlled, Hard, Soft) Test Dimensions: Varying the parameters within the Value Set to observe the behavior of replication against different combinations Validation: Verify that all message id are matching in both producer log and consumer log Verify that all corresponding topic-partition log segment files checksums are matching across all replicas
C.6 Replication with Mirror Maker Failure	 Setup: Configure 2 Clusters with 1 Mirror Maker: a. Source: 1 Zookeeper, 1 ~ 6 brokers, 1 producer b. Mirror Maker to replicate data from Source to Target c. Target: 1 Zookeeper, 1 ~ 6 brokers, 1 consumer Test Description: a. Follow the steps in B.1 b. During the test session, introduce failure to Mirror Maker. The no. of failures can be specified in the corresponding testcase_<n>_properties.json</n> c. The type of failures are defined in Test Dimentions in B.2 (Controlled, Hard, Soft) Test Dimensions: Varying the parameters within the Value Set to observe the behavior of replication against different combinations Validation: a. Verify that all message id are matching in both producer log and consumer log b. Verify that all corresponding topic-partition log segment files checksums are matching across all replicas

C.7 Replication with Backw ard Compatibility (0.7 & 0.8 Kafka jars) / Migration Tool	 Setup: Configure 2 Clusters with 1 Mirror Maker: a. Source: 1 Zookeeper, 1 ~ 6 brokers, 1 producer (<i>running in 0.7 Kafka jar</i>) b. Mirror Maker to replicate data from Source to Target c. Target: 1 Zookeeper, 1 ~ 6 brokers, 1 consumer (<i>running in 0.8 Kafka jar</i>) Test Description: a. Follow the steps in B.1 Test Dimensions: Varying the parameters within the Value Set to observe the behavior of replication against different combinations Validation: a. Verify that all message id are matching in both producer log and consumer log b. Verify that all corresponding topic-partition log segment files checksums are matching across all replicas
C.8 Replication with Produ ction Setup	 Setup: Configuration as follows: Zookeeper: 5 nodes cluster Brokers: 8 nodes cluster Brokers: 8 nodes cluster log segment size: 1GB Producer compression: On Async Producer: Yes Producer Acks: -1 Replica Factor: 3 No. Topics: 1000 No. Partitions: 10 Test Description: Repeat the steps in B.1 During the test session, randomly introduce failure to Leader, Follower, Mirror Maker or Controller constantly. This would be a reliability / stress test which should last a few hours. The type of failures are defined in Test Dimentions in B.2 (Controlled, Hard, Soft) Test Dimensions: Fixed at the set up stage. Validation: Verify that all message id are matching in both producer log and consumer log Verify that all corresponding topic-partition log segment files checksums are matching across all replicas