KIP-894: Use incrementalAlterConfigs API for syncing topic configurations

- Status
- Motivation
- Public Interfaces
- Proposed Changes
- Compatibility, Deprecation, and Migration Plan
- Test Plan
- Rejected Alternatives

Status

Current state: Accepted

Discussion thread: here

JIRA: here

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

Currently Mirrormaker uses deprecated AlterConfigs API when syncing topic configurations for broker compatibility to 0.11.0. In addition, Mirrormaker has a configurable ConfigPropertyFilter class to select the topic configuration properties to replicate and uses org.apache.kafka.connect.mirror.DefaultConfigPropertyFilter by default. The default class excludes certain topic configurations from replication such as follower.replication.throttled.replicas and leader.replication.throttled.replicas.

However, the AlterConfigs API replaces any existing configuration with the new configuration. Due to this, additional configurations set on a remote topic get cleared up when Mirrormaker syncs topic configurations whether they are excluded from replication or not. For example this prevents running Cruise Control on the target cluster as it may set follower.replication.throttled.replicas and leader.replication.throttled.replicas.

MirrorMaker should not clear topic configurations that are excluded from replication.

Public Interfaces

A new method will be added to ConfigPropertyFilter.

```java
public interface ConfigPropertyFilter extends Configurable, AutoCloseable {
    boolean shouldReplicateSourceDefault(String prop);
}
```

Proposed Changes

This KIP proposes to use the IncrementalAlterConfigs API for syncing topic configurations in MirrorMaker. The IncrementalAlterConfigs API has been around since Kafka 2.3.0 and addresses the shortcoming of the AlterConfigs API. In order not to break the compatibility, a new setting will be introduced to MirrorMaker that gives the option to enable or disable IncrementalAlterConfigs API for syncing topic configurations. This new setting is expected to serve as a temporary measure until the next major release when the API is always used.

- A new configuration setting to MirrorMaker:
  - Name: use.incremental.alter.configs
  - Description: Deprecated. Whether to automatically use the IncrementalAlterConfigs API for syncing topic configurations. This configuration will be removed in Kafka 4.0 and the IncrementalAlterConfigs API will be used by default. Users should make sure that the target cluster is running Kafka 2.3.0 or later.
  - Type: String
  - Default: requested

  When set to "requested", MirrorMaker will use the IncrementalAlterConfigs for syncing topic configurations. If any request receives an error from an incompatible broker, it will fallback to the deprecated AlterConfigs API for the subsequent calls and log a WARN message e.g. "The target cluster <ALIAS> is not compatible with IncrementalAlterConfigs API. Therefore using deprecated AlterConfigs API for syncing topic configurations."

  When explicitly set to "never", MirrorMaker will use the deprecated AlterConfigs API for syncing topic configurations.
When explicitly set to "required", MirrorMaker will use the IncrementalAlterConfigs API for syncing topic configurations. If it receives an error from an incompatible broker, the MirrorMaker will report this to the user and fail the connector.

Currently if users delete a topic configuration in the source cluster, it gets also deleted from the target cluster due to the replace all behaviour of AlterConfigs API. However with IncrementalAlterConfigs, the topic configuration in the target cluster would potentially be left unchanged. Therefore the default topic configuration on the source cluster will explicitly be cleared from the target cluster via Delete operation when using IncrementalAlterConfigs API.

The KIP also proposes adding the ability for users to choose between target cluster’s default or source cluster’s default when using IncrementalAlterConfigs for syncing topic configurations. For instance, users would be able to configure ConfigPropertyFilter to choose whether or not to replicate the default configurations of the source topic to the target cluster. This gives users a bit more control in how they want to replicate their topic configurations.

- Extend ConfigPropertyFilter class to have a new method called shouldReplicateSourceDefault which will return false by default. If set to false while using IncrementalAlterConfigs API, the default configurations in the source cluster will explicitly be cleared from the target cluster using the Delete operation. User will be able to configure this method to return true in order to include source cluster’s default configurations in the replication.

- Extend DefaultConfigPropertyFilter with a new property "use.defaults.from" that accepts "source" or "target" values. This will allow users to control which cluster’s defaults get applied on the target. The user can also use the existing "config.properties.exclude" property to exclude configuration from the replication regardless of which cluster’s default is chosen.

Compatibility, Deprecation, and Migration Plan

By default, the new setting will be set to "requested" which means we’ll use IncrementalAlterConfigs if possible, and fallback to AlterConfigs API if incompatible.

This means there will be a behaviour change while syncing topic configurations in MM2 after this KIP. For example, if changing retentions.ms of a topic in the source cluster, filtered configurations such as follower.replication.throttled.replicas of the topic currently get cleared in the target cluster. After this KIP, the current value of follower.replication.throttled.replicas would be kept in the target cluster unless the target broker’s version is older than Kafka 2.3.0.

This setting will be marked as deprecated immediately.

In the next major version:

- use.incremental.alter.configs will be removed.
- MirrorMaker will use the IncrementalAlterConfigs API to sync topic configurations.

Users running the latest MirrorMaker connector against a target cluster with brokers older than Kafka 2.3.0, will have to upgrade.

Test Plan

Currently, the integration tests for MirrorMaker do not check if the target topic has the expected configurations after applying configuration filters, hence this bug was not caught. New integration tests that checks target topic configurations once they have been synced and tests different settings of the new configuration will be added.

Rejected Alternatives

- We could have changed the current updateConfigs() method to call DescribeConfigs API before AlterConfigs so that it had an incremental mode. This would have avoided the migration, however, the topics configurations could change between DescribeConfigs and AlterConfigs calls so we could still end up syncing incorrect topic configurations. We would need to stop using the deprecated API at some point anyway.