KIP-1014: Managing Unstable Metadata Versions in Apache Kafka

Status

Current state: *"Under Discussion"*

Discussion thread: *here*

JIRA: KAFKA-15922

PR: 14860, 14984

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

Motivation

The MetadataVersion (MV) is used to define the on disk and over the wire record format for messages stored in the metadata log. A new metadata version is introduced each time a record format is changed or a new record is introduced to support a new feature. Examples of this are the support of SCRAM in MV IBP_3_5_IV2, support of Delegation Tokens in MV IBP_3_6_IV2 and the support of JBOD in MV IBP_3_7_IV2. All brokers and controllers must support a given MV before the cluster can start using that MV. Once the brokers and controllers agree to use a MV then the record format for that MV and all earlier MV is supported thus also enabling all earlier features, thus if you want to use Delegation Tokens you get the record format changes necessary to support SCRAM also.

The development of a new feature that requires a new MV often requires multiple commits before the feature is complete, sufficiently tested, and ready for production. Sometimes there is a need for multiple features to be developed at the same time and in some cases not all of these features with be complete and ready for production for a given release. In these scenarios we need to reorder MetadataVersions such that production ready features have their MV before those that are not ready so that customers can specify to use only those features which are stable.

Proposed Changes

The idea is to define a production-ready MetadataVersion defined in MetadataVersion.java as `LATEST_PRODUCTION` such that all MetadataVersions less than or equal to this value are stable and ready for production. By default only those features with a MV less than or equal to the `LATEST_PRODUCTION` MV will be available to use in a cluster. A cluster can set `"unstable.metadata.versions.enable"` to true in the broker and controller properties file to enable the use of a MV greater than `LATEST_PRODUCTION` in a cluster. Unit tests will default to use all MetadataVersions defined not just those less than or equal to `LATEST_PRODUCTION` so that developers can add unit tests for new features easily. Any unstable feature which requires a new MV can have that MV change in a later release of Apache Kafka to a higher MV. This will occur when other features with a higher MV become stable and ready for production. A MV defined as unstable will never have its MV moved to a lower MV.

Public Interfaces

A new property `"unstable.metadata.versions.enable"` will be created to allow for developers to create clusters and use features with an unstable MV. Using this feature has risks and may result in the cluster with this feature being unable to upgrade to a future release or even a more recent build from trunk.

Compatibility, Deprecation, and Migration Plan

- Features with a stable MV will never have that MV change!
- Users will only be able to use features with a stable MV unless they explicitly configure their clusters with the new property.
- Features with an unstable MV may have that MV increased if a feature with a higher MV becomes stable and ready for production.
- Using an unstable MV may result in the cluster being unable to upgrade to a future AK release or even a more recent build from trunk.

Test Plan

We will validate that the StorageTool will only format the metadata log to a stable MV unless "unstable.metadata.versions.enable" is set to true in the properties file.

We will validate that only stable features are enabled in the defaultFeatureMap unless the enableUnstable is set to true.

We will validate that stable features are enabled in a their specific MV. (TODO)