KIP-690: Add additional configuration to control MirrorMaker 2 internal topics naming convention

Status

Current state: Approved

Discussion thread: here

Voting thread: here

JIRA: *here*

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

Motivation

MM2 internal topic names (heartbeats, checkpoints and offset-syncs) are hardcoded in the source code which makes it hard to run MM2 with any Kafka Cluster that has rules around topic's naming convention and doesn't allow auto-creation for topics. In this case you will need to create these internal topics up-front manually and make sure they do follow the cluster rules and guidance for topic creation, so MM2 should have flexibility to let you override the name of internal topics to follow the one you create.

Public Interfaces and Proposed Changes

This KIP proposes adding new methods to `ReplicationPolicy` that defines how MM2 internal topics are named based on some new configuration.

```
@InterfaceStability.Evolving
public interface ReplicationPolicy extends Configurable {
    ....
    /** Returns heartbeats topic name.*/
    default String heartbeatsTopic() { return "heartbeats";}

    /** Returns the offset-syncs topic for given cluster alias. */
    default String offsetSyncsTopic(String targetAlias) { return " mm2-offset-syncs." + targetCluster + ".
internal";}

    /** Returns the name checkpoint topic for given cluster alias. */
    default String checkpointsTopic(String clusterAlias) { return clusterAlias + ".checkpoints.internal"; }

    /** check if topic is a heartbeat topic, e.g heartbeats, us-west.heartbeats. */
    default boolean isHeartbeatsTopic(String topic) {
        return heartbeatsTopic().equals(originalTopic(topic));
    }

    /** check if topic is a checkpoint topic. */
    default boolean isCheckpointsTopic(String topic) {return  topic.endsWith(".checkpoints.internal")}

    /** Check topic is one of MM2 internal topic, this is used to make sure the topic doesn't need to be
replicated.*/
    default boolean isMM2InternalTopic(String topic) {return  topic.endsWith(".internal");}

    /** Internal topics are never replicated. */
    default boolean isInternalTopic(String topic) {
                boolean isKafkaInternalTopic = topic.startsWith("__") || topic.startsWith(".") ||  topic.
endsWith("-internal");
        return isMM2InternalTopic(topic) || isKafkaInternalTopic;
    }
}
```

The default implementation of these methods will keep the current behaviour and We will use the `DefaultReplicationPolicy` class to add the ability to override the separator for these topics using `replication.policy.separator which is '.' by default.`

By using the `DefaulReplicationPolicy` the values will be

- `heartbeatsTopic() heartbeats`
- `checkpointsTopic(clusterAlias) clusterAlias.checkpoint.internal`
- `offsetSyncsTopic(clusterAlias) mm2-offset-syncs.clusterAlias.internal`
- `isMM2InternalTopic(topic.internal) -> return true if topic name has internal suffix`

```
For MM2 users who use a custom ReplicationPolicy they will not be able to use replication.policy.separator to
control internal topics suffix, these users will need to handle these methods if they wish to customise it or use
the `replication.policy.separator`
```

Compatibility, Deprecation, and Migration Plan

- When users upgrade an existing MM2 cluster they don't need to change any of their current configuration If they are using the default `replication.policy.separator` as this proposal maintains the default behaviour for MM2 with the default configs.
- If users upgrade an existing MM2 to 3.1.x, 3.2.x, 3.3.x, 3.4.x, or 3.5.x and they are using a customised `replication.policy.separator` they need to provide a new version of `ReplicationPolicy` (which can optionally subclass the `DefaultReplicationPolicy` class) that overrides the `ReplicationPolicy.offsetSyncsTopic` and `ReplicationPolicy.checkpointsTopic` methods to use old topics if they still want to use the old internal topics. Related JIRA to this is discussed in KAFKA-15102.

Rejected Alternatives

1- Add new interface for internal policies, the reason to reject is to minimise the number of MM2 customised classes

2- Add a configuration for the full name of each topic, this looks simple from the first look however this solution have many downsides

1. Increasing MirrorMaker2 config as the more advanced and complicated is replication design between clusters the more config will be needed if the clusters don't share the same naming convention for topics. For example use cases where user migrating data from multiple clusters used by different teams who don't share the same naming conventions.
2. MM2 needs to distinguish between internal topics from different clusters, for example right now the name for offset-syncs topic to mm2-offset-syncs.<cluster-alias>.internal and for checkpoints is <cluster-alias>.checkpoints.internal so the name has a pattern to link it back to the herder of source -> target mirror. So for keeping backward compatibility we will need the MM2 config to load cluster aliases first in order to determine the default value of these 2 topics.
3. Allowing users to set full name for each topic may create more problem if the user by mistake used the same topic for multiple clusters. For example, considering use case where MM2 is used to mirror between multiple clusters

   MM2 config

   ```
   source1 -> target.enabled = true
   source2 -> target.enabled = true
   ```

   Now if user set the same topic for checkpoints for both source1 and source2 this will led MM2 to push checkpoints data for both source clusters into one topic which goes against one of MM2 designs decisions regards distinguishing internal topics from each cluster.