KIP-154 Add Kafka Connect configuration properties for creating internal topics

Status

Current state: *Adopted*

Discussion thread: here

Vote thread: here

JIRA: KAFKA-4667

Released*: 0.11.0.0*

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

Motivation

Kafka Connect is a framework for running source connectors that load data in external systems into Kafka topics and sink connectors that consume data in Kafka and write to external systems. Users set up Kafka Connect and install and configuration connectors, while Kafka Connect manages the connector configurations, tracks the status of the connectors, and records the progress the connectors make via offsets, and when needed distributes and restarts the connectors using that persisted information.

Kafka Connect stores connector configurations, offsets, and status using several Kafka topics, and these topics must be compacted and properly replicated and partitioned to ensure this persisted information remains available. Kafka Connect currently relies upon users to manually create these topics, but doing so is error prone and complicates the initial setup and management of Kafka Connect. If users do not manually create the required topics, then Kafka Connect falls back to having the broker auto-create them, although this is brittle as it requires that auto-creation of topics is enabled on the broker and that the default topic-level configurations are satisfactory for Kafka Connect. The latter assumption is quite problematic, and has led to connector configurations and offsets being reclaimed and preventing Kafka Connect from being able to properly restart these connectors.

Now that Kafka clients include administrative functionality (KIP-117, KAFKA-3265), it is possible for Kafka Connect to explicitly create its internal topics when they don't already exist. Existing configuration properties specify the names of the topics, but additional information is necessary to control the replication factor and number of partitions for these internal topics. Sensible default value can be provided, but it is still likely that users will want to choose the number of partitions and replication factor that best satisfy their needs.

Public Interfaces

We will add several new Kafka Connect distributed worker configuration properties to specify the replication factor and number of partitions when Kafka Connect creates these internal storage topics.

Proposed Changes

The following configuration properties will be added:

- `config.storage.replication.factor` will specify the replication factor for the topic used to store connector configurations. The default value will be '3'. The configuration importance is 'low'.
- `offset.storage.replication.factor` will specify the replication factor for the topic used to store connector offset information. The default value will be '3'. The configuration importance is 'low'.
- `offset.storage.partitions` will specify the number of partitions for the topic used to store connector offset information. The default value will be '25'. The configuration importance is 'low'.
- `status.storage.replication.factor` will specify the replication factor for the topic used to store connector and task status information. The default value will '3'. The configuration importance is 'low'.
- `status.storage.partitions` will specify the number of partitions for the topic used to store connector and task status information. The default value will be '5'. The configuration importance is 'low'.

The names of these configuration properties intentionally mirror the existing `config.storage.topic`, `offset.storage.topic`, and `status.storage.topic` configurations.

Note that the replication factor may not be larger than the number of available Kafka brokers in the cluster. In such cases, Kafka Connect will be unable to create the topics and will fail with an error stating this limitation, and the user will need to explicitly set the aforementioned configurations.

It is not a mistake that a `config.storage.partitions` configuration property is not mentioned above, since such a configuration property is not useful as the configuration storage mechanism requires a single partition.

Compatibility, Deprecation, and Migration Plan

These new configurations are used only when Kafka Connect needs to create its internal topics for storing configurations, offsets, and status. Users that are already running Kafka Connect will already have created such topics, and therefore they would have no need to explicitly set the configuration to non-default values. Users that are running new Kafka Connect distributed worker clusters may want to override the defaults in their new configuration files to reflect their own environments.

Rejected Alternatives

1. Providing options to configure all of the available topic-specific configuration settings. Users can still manually create the topics using the `kafka-topics.sh` tool and specifying any of the topic-specific configuration settings.
2. Having Kafka Connect compute the minimum replication factor based upon the desired replication factor and available number of brokers. This is unintuitive behavior that can lead to topics with insufficient replication factors that make it more likely to lose persisted information about connectors.
3. Using a single configuration for replication factor of all internal topics. The name of such a property did not mirror the pattern already in place for the existing `config.storage.topic`, `offset.storage.topic`, and `status.storage.topic` configurations. Using explicit properties for each of the topics is also more straightforward.