KIP-834: Pause / Resume KafkaStreams Topologies

Status

Current state: Adopted (3.3.0): (Vote thread: here)

Discussion thread: *here*

JIRA: *KAFKA-13873*

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

Motivation

In order to reduce resources used or modify data pipelines, users may want to pause processing temporarily. Presently, this would require stopping the entire KafkaStreams instance (or instances).

This KIP proposes the addition of the ability to pause and resume topologies. When the need to pause processing has passed, then users should be able to resume processing.

Public Interfaces

This proposal adds public methods to pause and resume KafkaStreams clients.

```
KafkaStreams.java
        /*
       Paused topologies will only skip over a) processing, b) punctuation, and c) standby tasks.
           Notably, paused topologies will still poll Kafka consumers, and commit offsets.
           This method sets transient state that is not maintained or managed among instances.
           Note that pause() can be called before start() in order to start a KafkaStreams instance in a manner
where the processing
                is paused as described, but the consumers are started up.
    */
    public void pause()
        // Returns the state of the metadata.  The last run through the topology may still be executing.
    public boolean isPaused()
    public void resume()
```

Proposed Changes

The change is to add the above methods. The implementation will manage internal metadata to achieve the desired results.

During task execution, processing tasks (both active and standby tasks) and punctuation will be skipped. Kafka consumers will continue to poll and potentially fill up their buffers. Commits will continue to happen on the schedule that they would have happened.

⚠ Since a pause Streams instance still continues to interact with Kafka consumers, the Streams resetter tool (kafka-streams-application-reset.sh) s hould not be used while an instance is paused.

Changes to NamedTopologies / Modular Topologies.

Since NamedTopologies are internal, we do not have to provide details, but since a KIP for modular topologies is expected, let us discuss a few details.

First, there needs to be API calls for modular/named topologies to pause, resume, and check if a topology is paused. Second, calling the above methods on an instance using modular topologies will pause/resume all of the topologies.

Compatibility, Deprecation, and Migration Plan

Since this KIP is additive, there are no compatibility concerns.

Rejected Alternatives

Alternative: Adding a new PAUSED state.

Given that Kafka consumers are still reading data, it seems unnecessary to designate the change at the level of a state change.

(Further, there is likely to be a future KIP for modular topologies. The intersection of this KIP and that would be that one ought to be able to pause individual modular topologies.)