KIP-672: Introduce Kafka Streams Specific Uncaught Exception Handler

This page is meant as a template for writing a KIP. To create a KIP choose Tools->Copy on this page and modify with your content and replace the heading with the next KIP number and a description of your issue. Replace anything in italics with your own description.

Status

Current state: *"Under Discussion"*

Discussion thread: *here [Change the link from the KIP proposal email archive to your own email thread]*

JIRA: *here [Change the link from KAFKA-1 to your own ticket]*

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

Motivation

In order to give users the chance to handle exceptions we plan on introducing and KafkaStreams specific uncaught exception handler. Currently we use the java default. However by the time this is triggered the stream thread that threw the exception will already be dead. This prevents any actions that would make use of the information in the stream thread. Also if this was the last thread alive there is no way to close all clients without first launching another thread. If the thread was killed by a cascading thread death due to a serdes error that new thread will die as well.

Public Interfaces

Public Interfaces

```
package org.apache.kafka.streams;

public enum UncaughtExceptionHandlerResponse {
        DEFAULT,
        SHUTDOWN_STREAM_THREAD,
        REPLACE_STREAM_THREAD,
        SHUTDOWN_KAFKA_STREAMS_CLIENT,
        SHUTDOWN_KAFKA_STREAMS_APPLICATION;
}

public interface UncaughtExceptionHandler {
        UncaughtExceptionHandlerResponse handleUncaughtException(Thread thread, Throwable exception);
}
```

Streams configuration error.shutdown.timeout.ms: How long to wait for the Kafka Stream client to shutdown when the shutdown is due to the result of the uncaught exception handler.

Proposed Changes

The user will return one of the following values in the error handler to trigger the corresponding action.

DEFAULT:

- Throws error and kills thread to be handled in the generic uncaughtExceptionHandler

SHUTDOWN_STREAM_THREAD:

- The current stream thread is shutdown and transits to state DEAD.
- The Kafka Streams client transits to ERROR if no other stream thread is alive.

REPLACE_STREAM_THREAD

- The current stream thread is shutdown and transits to state DEAD.
- A new stream thread is started if the Kafka Streams client is in state RUNNING or REBALANCING.

SHUTDOWN_KAFKA_STREAMS_CLIENT

- All Stream Threads in the client are shutdown and they transit to state DEAD
- The Kafka Streams client is shutdown. (shutdown or error?)
- The Kafka Streams client transits to state ERROR.
- The State directory cleaner thread stop
- The RocksDB metrics recording thread is not shutdown.

SHUTDOWN_KAFKA_STREAMS_APPLICATION

- The shutdown is communicated to the other Kafka Streams clients through the rebalance protocol.
- All Stream Threads across the entire application are shutdown and they transit to state DEAD
- All Kafka Streams clients, i.e., the entire Kafka Streams application, is shutdown.
- All Kafka Streams clients transit to state ERROR. (shutdown or error?)
- The State directory cleaner thread stop
- The RocksDB metrics recording thread is not shutdown.

Other Changes:

When all Stream Threads are dead the State directory cleaner thread will automatically shutdown to prevent the loss of any state. When Kafka Streams transitions from no alive threads to one in the creation of a new thread the State directory cleaner thread will relaunch. (not sure if we kill the thread or simply pause it)

Compatibility, Deprecation, and Migration Plan

- *We are not removing any behavior (should we over replace the current uncaught exception handler or use the default I added above?)*
- We should deprecate the setUncaughtException handler in KafkaStreams in favor of setStreamsUncaughtException

Rejected Alternatives

- make users do this manually in the standard uncaught exception handler