KIP-884: Add config to configure KafkaClientSupplier in Kafka Streams

- Status
- Motivation
- Public Interfaces
- Proposed Changes
- Compatibility, Deprecation, and Migration Plan
- Test Plan
- Rejected Alternatives

Status

Current state: Accepted

Discussion thread: here

JIRA: here

Motivation

Currently there is a way to pass a `KafkaClientSupplier` in `KafkaStreams` constructor. This KIP proposes to add a public config in `StreamsConfig` to pass the class through config and use reflection to create

a new `KafkaClientSupplier` object. The major motivation is to allow existing *KafkaStreams* applications to upgrade without application code changes. Note that if existing *KafkaStreams* applications want to

use a customized `KafkaClientSupplier` , they may still need to add required dependencies.

Public Interfaces

We will add a public field in `StreamsConfig` called `default.client.supplier` to configure the `KafkaClientSupplier` we use in KafkaStreams. The default value of the config will the classname

of `DefaultKafkaClientSupplier` .

We will also add a new public method in `StreamsConfig` called `getKafkaClientSupplier` . It will return `KafkaClientSupplier` based on the config.

Proposed Changes

In `KafkaStreams` constructor, we will read the value of `default.client.supplier` from `StreamsConfig` . We will use reflection to create an object of given class and throw

an exception if the class type isn't `KafkaClientSupplier` . If user provides both the config and supply `KafkaClientSupplier` in `KafkaStreams` co nstructor, the config will be ignored and supplied

`KafkaClientSupplier` will be used.

Compatibility, Deprecation, and Migration Plan

There will be no compatibility, deprecation and migration issues.

Test Plan

Unit test can capture the changes in this KIP. We can test happy path where object can be created successfully and exception case where class couldn't be found or type is wrong.

Rejected Alternatives

An alternative is that we don't need this change and user can use `KafkaStreams` constructor to supply any `KafkaClientSupplier` since it's a public constructor.

The benefits of this alternative are:

1. Don't need this KIP and code changes in this KIP 🙂
2. Users don't need to upgrade AK version to use new `KafkaClientSupplier`
3. Users using same config for different `KafkaStreams` can have a different `KafkaClientSupplier`

The downsides of this alternative are:

1. Users need to change every `KafkaStreams` constructor call in order to use a new `KafkaClientSupplier` .

So to cater for users who don't want to update their source code, we are proposing to add this config option as well.