KIP-60 - Make Java client classloading more flexible

- Status
- Motivation
- Public Interfaces
- Proposed Changes
  - Classloading for default classes
  - Classloading of custom classes
- Compatibility, Deprecation, and Migration Plan
- Rejected Alternatives
  - Add classloader as a separate property
  - Add an option to disable TCCL

Status

Current state: *Under Discussion*

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

JIRA: *KAFKA-3680*

Released: 0.10.1.0

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

Motivation

Kafka producers and consumers have several configuration options which are classes or list of classes. At the moment, these classes are dynamically loaded using the thread context classloader (TCCL) if one is set, or using the classloader that loaded Kafka classes if a context classloader is not set on the current thread. This works well in most environments including JEE where thread context classloaders are typically used. But some environments like OSGi don’t use thread context classloader and hence the current implementation of dynamic classloading in Kafka doesn’t work well in these environments. This KIP proposes some simple changes to enable Kafka clients to be run in any classloading environment including modular multi-classloader environments like OSGi.

Public Interfaces

Custom classes may be specified for configuration properties of type `Type.CLASS` and `Type.LIST`. Values of `Type.CLASS` may be an actual class object or a class name. Elements of `Type.LIST` may currently only be Strings and hence are a list of class names.

This KIP proposes to modify `Type.LIST` to optionally specify class objects when the list represents classes. This enables all classloading to be performed by client applications outside of Kafka, making the configuration of Kafka fully flexible. The existing classloading implementation will be retained for dynamic classloading when class names are specified, avoiding any impact on existing applications.

Proposed Changes

Classloading for default classes

The current Kafka implementation uses static classloading for some classes used as default configuration values (eg. `DefaultPrincipalBuilder`) and TCCL based classloading for others (eg. `DefaultPartitioner`). Classloading of all default classes will be made consistent and modified to use static classloading to enable this to work regardless of TCCL. Since default classes are bundled with Kafka client classes, this will work for any classloading environment including OSGi, unlike the current thread context classloader.

Classloading of custom classes

Handling of `Type.LIST` will be modified to handle Strings as well as Class objects. Classloading for classes where class names are specified will not be modified.

Compatibility, Deprecation, and Migration Plan

Since handling of configuration properties specified as class names will not be modified, existing clients will continue to work.

Rejected Alternatives

Add classloader as a separate property

A single classloader may not be sufficient for loading any configurable class in Kafka. For example, one OSGi bundle may contain custom serializers and another bundle may contain custom partitioners and these may not necessarily have visibility of each other. The proposed solution works in these scenarios.

Add an option to disable TCCL

Kafka currently uses TCCL if set on the current thread and the classloader of the current class if TCCL is not set. An option to choose between the two may be useful in some environments. But since dynamic classloading of application classes in Kafka can be avoided altogether by specifying class objects rather than class names, an extra configuration option may be unnecessary.