KIP-815: Support max-timestamp in GetOffsetShell

Status

Current state: Adopted

Discussion thread: https://lists.apache.org/thread/z93sw7cx5tcq33v28vv19d34lhobfy3z   https://lists.apache.org/thread/pqm4881dsp5xw8245206052wm525ntjt

JIRA: *KAFKA-13509*

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

Motivation

We support a new type of `OffsetSpec` in KIP-734 which is max-timestamp, and it's preferable to extend it to GetOffsetShell. In the future, maybe more `OffsetSpec` types will be added to it.

Currently, we use KafkaConsumer to get offsets in GetOffsetShell, whereas the new OffsetSpec is only supported in AdminClient, so we need to change the client from KafkaConsumer to AdminClient.

Public Interfaces

This KIP change 2 parameters for command line tool *kafka-get-offsets.sh*. These 2 arguments are:

- *--time ,* we could pass -1(latest), -2(earliest) or a specified timestamp currently, in this KIP, we support -3(max-timestamp) which is introduced in KIP-734, and we'll also support "earliest" "latest" "max-timestamp" directly.
- --command-config, currently the property file will be passed to KafkaConsumer Client, In this KIP, we change it to the property file of AdminClient.

here are some examples,

```
# get the latest offset of topic1 :
bin/kafka-get-offsets.sh --bootstrap-server localhost:9092 --topic topic1 --time -1

# get the latest offset of topic1 :
bin/kafka-get-offsets.sh --bootstrap-server localhost:9092 --topic topic1 --time latest

# get the earliest offset of topic1 :
bin/kafka-get-offsets.sh --bootstrap-server localhost:9092 --topic topic1 --time -2

# get the earliest offset of topic1 :
bin/kafka-get-offsets.sh --bootstrap-server localhost:9092 --topic topic1 --time earliest

# get the offset of max timestamp of topic1:
bin/kafka-get-offsets.sh --bootstrap-server localhost:9092 --topic topic1 --time max-timestamp

# get the offset of max timestamp of topic1:
bin/kafka-get-offsets.sh --bootstrap-server localhost:9092 --topic topic1 --time -3


# contents of kafka_admin_client.properties
bootstrap.servers=localhost:9092
security.protocol=SASL_PLAINTEXT
sasl.mechanism=SCRAM-SHA-256
sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule required username="root" password="
123456";

# get offset from sasl kafka broker
bin/kafka-get-offsets.sh --command-config kafka_admin_client.properties --topic topic1 --time -1
```

Proposed Changes

1. Support max timestamp in GetOffsetShell
2. Support All AdminClient config in the file specified by --command-config, the only new config is `retries`, which means we will resend any request that fails when getting offsets
3. Some old KafkaConsumer config will be ignored, for example, key.deserializer and value.deserializer

Compatibility, Deprecation, and Migration Plan

The only incompatible change is the --command-config param, currently the property file will be passed to KafkaConsumer Client, In this KIP, we change it to the property file of AdminClient, we list the difference in two sections.

AdminClientConfig

1. Only one AdminClientConfig is not presented in ConsumerConfig which is `retries`, and it is not mandatory for AdminClient with default value=Integer.MaxValue, so this has very little effect on the client.
2. The only mandatory config in AdminClient is bootstrap.servers, which is also mandatory in KafkaConsumer.

ConsumerConfig

1. Most ConsumerConfig would not reasonably be used to configure the tool, for example, group.id and key.deserializer, they will be ignored by AdminClient and which has no influence on the tool.
2. Some config could possibly reasonably be used to configure the tool, they are listed as follows:

config	Consumer behavior	AdminClient behavior	Description
client.dns.lookup, bootstrap.servers	use ClientUtils. parseAndValidateAddresses to get InetSocketAddress of broker, and to sendMedatataRequest to the broker.	use ClientUtils. parseAndValidateAddresses to get InetSocketAddress of broker, and to send MedatataRequest to the broker.	Both clients will take the same action

default.api.timeout.ms	The consumer will retry until the timeout is reached	The AdminClient will retry until the timeout is reached or the number of retries exceeds the limit	AdminClient does the same with Consumer since the default of `retries` is In Integer.MaxValue and have the same default timeout value
request.timeout.ms	Used by NetworkClient for individual rpcs to await acknowledgment from servers	Used by KafkaAdminClient to decide whether each `Call` is timeout, and the timeout for NetworkClient is 3600000	There is a small difference wheras the result are the same.
connections.max.idle.ms, reconnect.backoff.ms, reconnect.backoff.max.ms, send.buffer.bytes, receive.buffer.bytes, socket.connection.setup.timeout.ms, socket.connection.setup.timeout.max.ms	Used to construct NetworkClient	Used to construct NetworkClient	Both clients will take the same action
retry.backoff.ms	The amount of time to wait before attempting to retry a failed ListOffsetsRequest rpc and MetadataRequest rpc	The amount of time to wait before attempting to retry a failed ListOffsetsRequest rpc and MetadataRequest rpc	Both clients will take the same action
client.id	just an identifier and are hardcoded to GetOffsetShell	just an identifier and are hardcoded to GetOffsetShell	Both clients will take the same action
metadata.max.age.ms	The period of time to evict metadata cache for `ConsumerMetadata`	The period of time to evict metadata cache for `AdminMetadataManager`	There may be difference in implementation details, the metadata cache will have the same expire time.
metric.reporters, and all other metric releated configs	Used to get client metrics	Used to get client metrics	Both clients will take the same action
security.protocol, and all other security related configs	Used to establish security connection	Used to establish security connection	Both clients will take the same action
retries	Consumer will retries until default.api.timeout.ms is reached	default value is Integer.MaxValue	The AdminClient will act the same with Consumer by default, and we can set a config to control how many times we can retry

So we can conclude this is a compatible change and the transition won't be noticed.

Rejected Alternatives

Extend KafkaConsumer to support max-timestamp

Currently, we can get the earliest and latest offset using KafkaConsumer, we can also simply support max-timestamp in GetOffsetShell if we support it in KafkaConsumer.

Ultimately, we determine that admin client is a better way to implement this, or otherwise we need to extend KafkaConsumer every time we add a new OffsetSpec, in addition, AdminClient is more lightweight since we need to construct many unused components in KafkaConsumer, e.g. ConsumerCoordinator.