KIP-971: Expose replication-record-lag MirrorMaker2 metric

Status

Current state: Vote in progress

Discussion thread: *here*

JIRA: KAFKA-14112

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

Motivation

The current Kafka architecture lacks a built-in mechanism to directly track the replication record lag, i.e the number of to-be-replicated records. This could be essential for monitoring and maintaining the health and performance of data replication processes. The replication record lag, defined as the difference between the last end offset of the source partition (**LEO**) and the last replicated source offset (**LRO**), is beneficial for understanding the progress and potential bottlenecks in data replication scenarios.

Public Interfaces

- **org/apache/kafka/connect/mirror/MirrorMetrics** - new metrics templates for `max` , `min` , `avg` and `value` of `replication-record-lag`

MirrorMetrics.java

```
replicationRecordLag = new MetricNameTemplate(

"replication-record-lag", SOURCE_CONNECTOR_GROUP,

"Count of records to be replicated from source to target cluster.", partitionTags);

replicationRecordLagMax = new MetricNameTemplate(

"replication-record-lag-max", SOURCE_CONNECTOR_GROUP,

"Max count of records to be replicated from source to target cluster.", partitionTags);

replicationRecordLagMin = new MetricNameTemplate(

"replication-record-lag-min", SOURCE_CONNECTOR_GROUP,

"Min count of records to be replicated from source to target cluster.", partitionTags);

replicationRecordLagAvg = new MetricNameTemplate(

"replication-record-lag-avg", SOURCE_CONNECTOR_GROUP,

"Average count of records to be replicated from source to target cluster.", partitionTags)
```

- **org/apache/kafka/connect/mirror/MirrorSourceTaskConfig** - a configuration to control the poll interval for the Consumer.endOffsets() call at LEO acquisition mentioned below.
  - **Proposed name**: *replication.record.lag.metric.refresh.interval*
  - **Proposed default:** *15 seconds*
- **org/apache/kafka/connect/mirror/MirrorSourceTaskConfig** - a configuration to control the time-to-live property of a stale partition LRO in order to avoid the in-memory LRO "cache" from causing memory overhead.
  - **Proposed name:** *replication.record.lag.metric.last.replicated.offset.ttl*
  - **Proposed default:** *20 seconds*

Proposed Changes

This proposal aims to enhance Kafka's monitoring capabilities by introducing a new metric to track the replication record lag for a given topic-partition. The metric will be calculated by taking the difference between the LEO of a partition, which will be periodically updated by calling `Consumer.endOffsets()`, and the LRO of a partition, which will be stored in an in-memory "cache" and updated during the task's producer callback.

The in-memory "cache" entries will be controlled by a TTL-based eviction policy in order to avoid storing stale LRO entries which can cause an eventual OOM error.

The `replication.record.lag.metric.refresh.interval` metric will also accept a value of `-1` to indicate that a user is willing to disable reporting for this metrics - upon reading this value for this config, the period poll job for a partition LEO will not be submitted causing the metric reporting to be skipped.

The proposed changes involve the following steps:

1. **LRO Tracking Mechanism:**
    - Introduce a new in-memory "cache" to store the LRO for each topic-partition.
    - Update the producer callback logic to keep this LRO cache up-to-date.
2. **LEO Acquisition:**
    - Introduce a periodic poll mechanism to keep a fresh set of end offsets for all partitions
        - An alternative is to call `Consumer.endOffsets()` at each poll, which would be inefficient and expensive.
3. Calculate the replication offset lag as `LEO - LRO`
4. **Expose Replication Record Lag Metric:**
    - Introduce a new MirrorMaker metric, named "replication-record-lag", which represents the calculated replication offset lag.

Compatibility, Deprecation, and Migration Plan

- This proposal ensures backward compatibility by introducing new metrics and modifying existing poll loop and callback mechanisms which do not impact existing Kafka features, APIs, or configurations.

Test Plan

Metrics will be tested in **org.apache.kafka.connect.mirror.MirrorSourceTaskTest** in a unit test fashion. Unfortunately there are no system tests that I could find where this change could be tested, but I am happy to do it if there is a suitable place where it is already done and that I have missed, or to implement a new system test.

Rejected Alternatives

It might be argued that the existing `replication-latency-ms` metric already provides satisfactory information about the replication lag.

`replication-latency-ms` is calculated in the producer callback: it is the difference between the system time at the moment of callback invocation and the timestamp of the original source record.

This metric is useful for tracking *time-based* latency of the replication, but it does not provide information about *count-based* latency which the new `replication-record-lag` is aimed for.

Note: `replication-latency-ms` may be inaccurate under specific circumstances: