KIP-933: Publish metrics when source connector fails to poll data

Status

Current state: *Under Discussion*

Discussion thread: *here*

JIRA: *here*

Abstract

This KIP proposes adding new metrics to Kafka source connectors to track and report failures encountered during the record polling process. By providing these granular metrics, it becomes easier to monitor and diagnose issues related to record polling failures, enabling users to take appropriate actions for fault tolerance and troubleshooting.

Motivation

Currently, there is no metric in Kafka Connect to track when a source connector fails to poll data from the source. This information would be useful to operators and developers to visualize, monitor and alert when the connector fails to poll records from the source.

Existing metrics like **kafka_producer_producer_metrics_record_error_total** and **kafka_connect_task_error_metrics_total_record_failures** only cover failures when producing data to the Kafka cluster but not when the source task fails with a retryable exception or ConnectException.

Polling from source can fail due to unavailability of the source system or errors with the connect configuration. Currently, this cannot be monitored directly using metrics and instead operators have to rely on log diving which is not consistent with how other metrics are monitored.

Public Interfaces

The new metrics will be added at the granularity of a task and will be exposed via JMX interface similar to existing metrics,

Metrics group: Ref

```
kafka.connect:type=task-error-metrics,connector="{connector}",task="{task}"
```

Metric Names:

source-record-poll-error-total - The total number of times a source connector task failed to poll data from the source system. This will include both retryable and non-retryable exceptions.

source-record-poll-error-rate - The rate of total errors encountered per second while polling the source system. This is useful for calculating the errors as a percentage.

Proposed Changes

This proposal suggests the following modifications to the Kafka source connector framework:

1. **Record new metrics:** The poll function in *AbstractWorkerSourceTask* will have an exception handling block that will record these errors. The error metric will be incremented whenever the exception is encountered.
2. **Register new metrics:** The recorded metrics will be added to *ConnectMetricsRegistry* along with other Connect metrics.
3. **Reporting metrics**: The connector framework will expose these new metrics via JMX (Java Management Extensions) for monitoring and integration with existing monitoring systems. Operators can configure their monitoring tools to collect and analyze these metrics to gain insights into the health and performance of the source connectors.
4. **Documentation**: The Kafka documentation will be updated to include details on the new metrics and their usage. It will provide guidance on how to leverage these metrics for monitoring and troubleshooting purposes.

Compatibility, Deprecation, and Migration Plan

- *What impact (if any) will there be on existing users?*
  - This change only adds new metrics that users can utilize to improve their monitoring of source connectors. There is no impact to existing metrics.
- *If we are changing behavior how will we phase out the older behavior?*
  - N/A
- *If we need special migration tools, describe them here.*
  - N/A
- *When will we remove the existing behavior?*
  - N/A

Test Plan

Describe in few sentences how the KIP will be tested. We are mostly interested in system tests (since unit-tests are specific to implementation details). How will we know that the implementation works as expected? How will we know nothing broke?

1. Unit testing: Unit tests will be added to make sure the logic of adding the metrics is correct. It will also validate the correctness of the metric.
2. The change will also be manually tested to ensure that it works with real source connectors.

Rejected Alternatives

1. The alternative is to monitor the source system for failures but it is not always possible when the source system does not provide these metrics. It is also good to know the errors encountered, from Kafka Connect perspective rather than an external system.
2. The other alternative is to push the metric publishing logic to connectors but that's not a good idea as every connector implementation has to implement this.