KIP-356: Add KafkaConsumer fetch-error-rate and fetch-error-total metrics

Status

Current state: Under Discussion

Discussion thread: *here*

JIRA: *KAFKA-7300*

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

Motivation

The KafkaConsumer is a complex client that requires many different components to function properly. When a consumer is not operating properly, it can be difficult to identify the root cause and which component is causing issues (ConsumerCoordinator, Fetcher, ConsumerNetworkClient, etc).

This KIP aims to improve the monitoring and detection of KafkaConsumer's Fetcher component.

Fetcher will send a fetch request for each node that the consumer has assigned partitions for.

This fetch request may fail under the following cases:

- Intermittent network issues
- Node sent an invalid full/incremental fetch response
    - FetchSessionIdNotFound
    - InvalidFetchSessionEpochException

These cases are logged, but it would be valuable to provide a corresponding metric that allows for monitoring and alerting.

Public Interfaces

This proposal adds the following metrics:

```
MetricName("fetch-error-rate", "consumer-fetch-manager-metrics", "The average per-second number of fetch
requests that resulted in errors")

MetricName("fetch-error-total", "consumer-fetch-manager-metrics", "The total number of fetch requests that
resulted in errors")
```

Proposed Changes

We will add a new sensor in FetchManagerMetrics fetchErrors which will use a Meter to provide the rate and total calculations.

```
private final Sensor fetchErrors;
```

There will also be a helper method in FetchManagerMetrics to increment the fetchErrors sensor.

```
private void recordFetchError() {
    fetchErrors.record(1);
}
```

Fetch errors will be recorded in two places:

```
/**
 * Set-up a fetch request for any node that we have assigned partitions for which doesn't already have
 * an in-flight fetch or pending fetch data.
 * @return number of fetches sent
 */
public int sendFetches() {
...
client.send(fetchTarget, request)
    .addListener(new RequestFutureListener<ClientResponse>() {

                // (1) onSuccess handler of sendFetches when response is invalid
                @Override
                public void onSuccess(ClientResponse resp) {
                    FetchResponse<Records> response = (FetchResponse<Records>) resp.responseBody();
                    FetchSessionHandler handler = sessionHandlers.get(fetchTarget.id());
                    if (handler == null) {
                        log.error("Unable to find FetchSessionHandler for node {}. Ignoring fetch response.",
                            fetchTarget.id());
                        return;
                    }
                    if (!handler.handleResponse(response)) {
                        sensors.recordFetchError();     // ***Record error
                        return;
                    }
                    ...
                }

                // (2) onFailure handler of sendFetches
                @Override
                public void onFailure(RuntimeException e) {
                    sensors.recordFetchError();        // ***Record error
                    FetchSessionHandler handler = sessionHandlers.get(fetchTarget.id());
                    if (handler != null) {
                        handler.handleError(e);
                    }
                }
                ...
}
```

Compatibility, Deprecation, and Migration Plan

This proposal adds new metrics without making any changes to the underlying operations so there should not be any issues.

Rejected Alternatives

None so far.