KIP-831: Add metric for log recovery progress

This page is meant as a template for writing a KIP. To create a KIP choose Tools->Copy on this page and modify with your content and replace the heading with the next KIP number and a description of your issue. Replace anything in italics with your own description.

Status

Current state: *"Adopted"*

Discussion thread: *here*

Vote thread: *here*

JIRA: *here*

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

Motivation

Log recovery is a process when a broker start up, if it has previous unclean shutdown, it'll be triggered to make sure the log is in a good state and not get corrupted. The process of log recovery is as below (producer snapshot and other steps are skipped here):

1. iterate all dirs in "log.dirs" config one by one
    a. find out the topic partition log folder under the dir
    b. Iterate all the topic partition log folders and add them as jobs to thread pool with the "num.recovery.threads.per.data.dir" config number of threads
        i. load all the segments under log folder, suppose there are 10 segments
        ii. filter out the segments after "recovery checkpoint", suppose there are 5 segments needed to be recovered
        iii. recover the 5 segments, one by one
            1. iterate all the record batches inside the segment
            2. validate all the batches
            3. rebuilt the indexes.

As we can imagine, if the broker stores a lot of logs, the log recovery process might take hours or days for the log recovery.

So far, we have no way to know the log recovery progress. All we can do is check the broker log and know it is busy on doing recovery. In this KIP, we're going to expose a `RemainingLogsToRecover` metric for each log.dir and `RemainingSegmentsToRecover` metric for each recovery thread, to allow the admin have a way to monitor the progress of log recovery.

Public Interfaces

Full Name	Type	Description
kafka.log:type=LogManager,name=remainingLogsToRecover ,dir=([-._\/\w\d\s]+) note: The dir format is the valid directory path string for OS(and valid for JAVA). Since the rule is different from each OS, here is just a simple example format.	32-bit gauge	The remaining logs number for each log.dir to be recovered
kafka.log:type=LogManager,name=remainingSegmentsToRecover,dir=([-._\/\w\d\s]+), threadNum=([0-9]+) note: The dir format is the valid directory path string for OS(and valid for JAVA). Since the rule is different from each OS, here is just a simple example format.	32-bit gauge	The remaining segments for the current log assigned to the recovery thread.

Proposed Changes

The proposal is to propose 2 metrics:

1. `RemainingLogsToRecover`: It's to show the remaining logs number for each log.dir to be recovered. The total number of logs to be recovered will be summed in step (1.b) described in "motivation" section. When each log completes the recovery for all the segments under the log, the `RemainingLogsToRecover` will be decremented, and in the end, it'll be 0. When broker is not under log recovery state, the metric will be removed.

2. `RemainingSegmentsToRecover`: It's to show the remaining segments to be recovered in each recovery thread (i.e. in each replica log). The total number of segments to be recovered will be calculated in step (1.b.ii) described in "motivation" section. When each segment completes the recovery, the `RemainingSegmentsToRecover` will be decremented, and in the end, it'll be 0. When broker is not under log recovery state, the metric will be removed.

For example:

configs:

- log.dirs=/tmp/log1,tmp/log2
- num.recovery.threads.per.data.dir=2

In the jmx, we'll see something like this during log recovery

- kafka.log
  - LogManager
    - RemainingLogsToRecover
      - /tmp/log1 => 5 — there are 5 logs under /tmp/log1 needed to be recovered
      - /tmp/log2 => 0
    - RemainingSegmentsToRecover
      - /tmp/log1 — 2 threads are doing log recovery for /tmp/log1
        - 0 => 1000 — there are 1000 segments needed to be recovered for thread 0
        - 1 => 3
      - /tmp/log2
        - 0 => 0
        - 1 => 0

It showed, currently, there are still 5 logs (partitions) needed to recover under /tmp/log1 dir. And there are 2 threads doing the jobs, where one thread has 1000 segments needed to recover, and the other one has 3 segments needed to recover.

After a while, the metrics might look like this:
It said, now, there are only 3 logs needed to recover in /tmp/log1, and the thread 0 has 9000 segments left, and thread 1 has 5 segments left (which should imply the thread already completed 2 logs recovery in the period)

- kafka.log
  - LogManager
    - RemainingLogsToRecover
      - /tmp/log1 => 3 — there are 3 logs under /tmp/log1 needed to be recovered
      - /tmp/log2 => 0
    - RemainingSegmentsToRecover
      - /tmp/log1 — 2 threads are doing log recovery for /tmp/log1
        - 0 => 300 — there are 300 segments needed to be recovered for thread 0
        - 1 => 5
      - /tmp/log2
        - 0 => 0
        - 1 => 0

After log recovery completes, the `RemainingLogsToRecover` and `RemainingSegmentsToRecover` metrics will be removed.

Compatibility, Deprecation, and Migration Plan

No compatibility issue and no migration plan needed because this KIP only adds a metric for log recovery.

Rejected Alternatives

1. output the log recovery progress in logs

This is not conflicted with the KIP, but finding the log recovery progress inside the broker logs is not easy for admins. Actually, during the implementation, we'll also improve the log output to have much clear info for log recovery progress. On the other hands, having the metrics is still a better way to monitor the log recovery progress for admins.

2. Provide a RemainingBytesToRecover metric:

Currently, when log manager start up, we'll try to load all logs (segments), and during the log loading, we'll try to recover logs if necessary.
And the logs loading is using "thread pool" as you thought.

So, here's the problem:
All segments in each log folder (partition) will be loaded in each log recovery thread, and until it's loaded, we can know how many segments (or how many Bytes) needed to recover.

That means, if we have 10 partition logs in one broker, and we have 2 log recovery threads (num.recovery.threads.per.data.dir=2), before the threads load the segments in each log, we only know how many logs (partitions) we have in the broker (i.e. RemainingLogsToRecover metric). We cannot know how many segments/Bytes needed to recover until each thread starts to load the segments under one log (partition).

That said, the `RemainingBytesToRecover` metric is difficult to achieve as you expected. I think the current proposal with `RemainingLogsToRecover` and `RemainingSegmentsToRecover` should already provide enough info for the log recovery progress.