Concatable Aggregated Logs

YARN-2942

5/22/15 -- Robert Kanter

Problem

Turning on log aggregation allows users to easily store container logs in HDFS and subsequently view them in the YARN and JHS Web UIs from a central place. Currently, there is a separate log file for each Node Manager per YARN application. This can be a problem for HDFS if you have a cluster with many nodes as you'll slowly start accumulating many (possibly small) files and blocks. To work around this, users are forced to delete the log files (through JHS configs) frequently even though disk space itself is not a concern.

Proposal

We propose a two-step solution to address these issues with log aggregation:

1. Concatenate log files to reduce the number of files.
2. Compact (rewrite) the log files to reduce the number of blocks.

We can provide a better solution by concatenating the aggregated log files for each node into a single log file per application.

All the per-node aggregated log files for an application currently reside in the same directory, and are in a binary format (`AggregatedLogFormat`, which is essentially a `TFile`). Each file consists of some header metadata (ACLs, owner, and version), followed by key-value pairs. Each of the keys is a Container ID and the values are the logs (stdout, stderr, and syslog are all here) of that container and their lengths.

Unfortunately, the `AggregatedLogFormat` is not concat-friendly, so we can't concatenate multiple aggregated log files together. We propose a new format, `ConcatableAggregatedLogFormat`, which stores all the logs in one file similar to the `AggregatedLogFormat`, but also has a separate index file. The log files themselves can be concatenated together, but the index file is small enough that we can just append the subsequent index entries (the index file isn't quite concat-friendly because of some header information). The format of the data itself can be similar to that of the original aggregated log files, which gives us the advantage of reusing most of the log parsing code that already exists when reading and writing the file.

The index file allows the JHS to look up the position in the file corresponding to a specific container, without having to read through the entire file, as it currently does. The proposed indexed format is shown below:

```
|-------------------------------------|
|Version                              |
|ACLs                                 |
|Owner                                |
|ContainerID,logtype,offset,length    |
|...                                  |
|-------------------------------------|
```
It has header information at the top, followed by an index entry for each log.

YARN-1376 and related JIRAs recently added the log aggregation status of an application to the NM heartbeat; this allows the RM to know the log aggregation status for every application. Armed with this information, the RM can wait for the aggregation to be complete and do the concatenation itself, as part of a new `RMAggregatedLogsConcatenationService` Service in the RM.

The resulting concatenated log file solves the problem of having too many small files, but can still leave a file using a lot of blocks due to the concatenation. We propose adding an `rmadmin` command to compact these concatenated log files - copy the file and replace the original with it. The copied file will inherently use an appropriate number of blocks. The cluster admin could run this command when appropriate, considering the HDFS load. If/when HDFS supports compaction, this command can be deprecated.

Deleting old logs will be handled automatically by the existing `AggregatedLogDeletionService` because the new log format and concatenation reuses the same directory setup as the original aggregated logs.

After a rolling upgrade to a version that uses the new log format, it is possible for an application to have logs in both formats. In this situation, the concatenation service would ignore the old formated files. The JHS and `yarn` CLI readers know how to read both formats and would handle this appropriately.

The work can be split up into 5 tasks:

1. Create the `ConcatableAggregatedLogFormat` Reader and Writer
2. Modify the NM to write using the `ConcatableAggregatedLogFormat`
3. Create the `RMAggregatedLogsConcatenationService`
4. Modify the JHS and `yarn` CLI to be able to read using either the `CombinedAggregatedLogFormat` (both before and after concatenation) or the `AggregatedLogFormat` (for backwards compatibility with existing logs)
5. Add the `rmadmin` command to compact the concatenated log files

We will include some configuration options, such as:

- Enable/Disable concatenation

Previous Designs

This design incorporates pieces of previous designs. With this v8 design, I believe we have striked a good balance of the advantages and disadvantages of each design. Some highlights:

- ZooKeeper is not needed. In fact, no extra coordination is needed at all!
- We're still essentially writing each file twice, but it's be done at a time when the cluster is not busy (determined by the admin)
- Logs are available as soon as possible in HDFS for the user
- The new process isn't too "invasive"
- Relatively simple; the biggest change is the file format

Follow-up Work

- Logs from long running services are not considered here at this time, though the new format is designed to be accommodating pending future work in YARN-2548. For now, they should be left alone by the concatenation code.