# Spark 3.0 终于支持 event logs 滚动了

## 背景

相信经常使用 Spark 的同学肯定知道 Spark 支持将作业的 event log 保存到持久化设备。默认这个功能是关闭的，不过我们可以通过 spark.eventLog.enabled 参数来启用这个功能，并且通过 spark.eventLog.dir 参数来指定 event log 保存的地方，可以是本地目录或者 HDFS 上的目录，不过一般我们都会将它设置成 HDFS 上的一个目录。

但是这个功能有个问题，就是这个 Spark Job 运行的过程中产生的所有 event log 都是写到单个文件中，这就导致了 event log 文件的大小和这个 Spark Job 的并行度、复杂度以及运行的时间有很大关系。如果我们是运行 Spark Streaming 作业，这个问题特别明显，我们经常看到某个 Spark Streaming 作业的 event log 达到几十 GB 大小！我们没办法清理或者删除一些不需要的事件日志，当我们使用 Spark 历史服务器打开这个几十 GB 大小的 event log，打开速度可想而知。

| Permission | Owner | Group | Size | Last Modified | Replication | Block Size | Name |
|---|---|---|---|---|---|---|---|
| -rwxrwx--- | hadoopuser | supergroup | 8.25 GB | 2019/2/26 下午9:10:33 | 3 | 128 MB | application_1535616282827_1089709_2.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 428.55 MB | 2019/2/27 下午5:24:31 | 3 | 128 MB | application_1535616282827_1094178_2.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 66.64 GB | 2019/2/14 下午6:43:11 | 3 | 128 MB | application_1535616282827_1187650_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 42.2 GB | 2019/2/27 下午4:56:42 | 3 | 128 MB | application_1535616282827_1194265_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 118.63 GB | 2019/2/6 上午12:12:13 | 3 | 128 MB | application_1535616282827_1232575_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 37.26 MB | 2019/2/11 上午10:34:52 | 3 | 128 MB | application_1535616282827_1263008_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 59.45 KB | 2019/2/14 下午3:57:11 | 3 | 128 MB | application_1535616282827_1291922_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 577.45 MB | 2019/2/19 下午4:12:35 | 3 | 128 MB | application_1535616282827_1327842_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 43.86 MB | 2019/2/19 下午4:34:31 | 3 | 128 MB | application_1535616282827_1328149_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 336.84 MB | 2019/2/19 下午7:23:54 | 3 | 128 MB | application_1535616282827_1328787_2.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 81.56 MB | 2019/2/21 上午11:12:46 | 3 | 128 MB | application_1535616282827_1339984_2.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 370.45 MB | 2019/2/21 上午11:12:25 | 3 | 128 MB | application_1535616282827_1339985_2.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 237 B | 2019/2/25 上午5:10:47 | 3 | 128 MB | application_1535616282827_1363561_1.inprogress |
| -rwxrwx--- | hadoopuser | supergroup | 152.74 MB | 2019/3/4 下午1:59:02 | 3 | 128 MB | application_1535616282827_1395963_1 |
| -rwxrwx--- | hadoopuser | supergroup | 344.24 MB | 2019/3/4 下午2:06:43 | 3 | 128 MB | application_1535616282827_1395978_1 |
| -rwxrwx--- | hadoopuser | supergroup | 225.09 MB | 2019/3/4 下午2:07:38 | 3 | 128 MB | application_1535616282827_1395991_1 |
| -rwxrwx--- | hadoopuser | supergroup | 145.09 MB | 2019/3/4 下午2:12:30 | 3 | 128 MB | application_1535616282827_1395997_1 |
| -rwxrwx--- | hadoopuser | supergroup | 179.41 MB | 2019/3/4 下午2:16:59 | 3 | 128 MB | application_1535616282827_1396018_1 |
| -rwxrwx--- | hadoopuser | supergroup | 231.71 MB | 2019/3/4 下午2:29:09 | 3 | 128 MB | application_1535616282827_1396038_1 |
| -rwxrwx--- | hadoopuser | supergroup | 133.64 MB | 2019/3/4 下午2:27:39 | 3 | 128 MB | application_1535616282827_1396042_1 |
| -rwxrwx--- | hadoopuser | supergroup | 445.56 MB | 2019/3/4 下午2:29:04 | 3 | 128 MB | application_1535616282827_1396045_1 |



如果大家经常使用 Log4j 的话，Log4j 提供了一个 RollingFileAppender，可以使长时间运行应用

的日志按照时间或者日志文件大小进行切割，从而达到限制单个日志文件的大小。Spark 的 event log 为什么不可以提供类似功能呢？值得高兴的是，即将发布的 Spark 3.0 为我们带来了这个功能（具体参见 SPARK-28594）。当然，对待 Spark 的 event log 不能像其他普通应用程序的日志那样，简单切割，然后删除很早之前的日志，而需要保证 Spark 的历史服务器能够解析已经 Roll 出来的日志，并且在 Spark UI 中展示出来，以便我们进行一些查错、调优等。

## 如何使用

### 事件日志滚动

首先必须使用 Spark 3.0，同时将 spark.eventLog.rolling.enabled 设置为 true（默认是 false）。那么 Spark 在 writeEvent 的时候会判断当前在写的 event log 文件大小加上现在新来的事件日志大小总和是否大于 spark.eventLog.rolling.maxFileSize 参数配置的值，如果满足将启动 event log roll 操作。

### 事件日志压缩

所谓事件日志压缩就是将多个滚动出来的事件日志文件合并到一个压缩的文件中。日志压缩涉及到的参数有 spark.history.fs.eventLog.rolling.maxFilesToRetain 和 spark.history.fs.eventLog.rolling.compaction.score.threshold。第一个参数的意思是进行 Compaction 之后需要保存多少个 event logs 为不压缩的状态，这个参数的默认值是 Int.MaxValue。也就是默认其实不启用事件日志 Compaction，所有 event logs 都将不会被 Compaction 到一个文件里面。

需要注意的：

- event logs 的 Compaction 操作是在 Spark 历史服务器端进行的，而且是在 Spark 历史服务器检查到有新的事件日志写到 spark.eventLog.dir 参数配置的目录中，这时候对应 Spark 作业的 event logs 将可能进行 compact 操作。
- event logs 的 Compaction 操作可能会删除一些没用的事件日志，关于删除的逻辑请看下一小结。这样经过 Compaction 操作之后，新生成的压缩文件大小将会变小。
- 一个 Spark 作业最多只会有一个 Compact 文件，文件的后缀是 .compact。已经有 compact 之后的合并文件在下一次进行 compact 的时候会被读出来和需要被 compact 的文件再一次合并，然后写到新的 compact 文件里。
- 已经被选中进行 compact 的 event logs 在执行完 compact 之后会被删除。

## 核心思想

整个 event logs 滚动项目应该可以大致分为两个阶段：

- 第一个阶段就是支持 event logs 滚动以及 event logs Compaction，这个在 SPARK-28594 里面，已经合并到 Spark 3.0 代码中。
- 第二个阶段是采用 AppStatusListener 使用的方法，即把 event logs 持久化到底层的

KVStore 中，并支持从 KVStore 把 event logs restore 出来，这个可以参见 SPARK-28870 ，这个还在开发中。

### 阶段一

支持 event log
滚动的隐层含义是支持删除旧的事件日志，要不然光支持滚动不支持删除，只是解决了单个 event log 文件的大小，解决不了整个作业 event log 总和大小。为了保证删除旧的事件日志之后 event logs 仍然可以被 Spark 历史服务器重放，我们需要定义出哪些事件日志是可以删除的。

拿 Streaming 作业来说，每个批次都是运行不同的作业。如果我们想删除一些事件日志，在大多数情况下，我们都会保存最近一些批次作业的事件日志，因为这些事件日志有助于我们分析刚刚遇到的问题。换句话说，删除那些比较旧的作业对应的事件日志是比较安全的，而且是比较可行的。这个在 SQL 查询的作业来说一样是适用的。

目前 Spark 在内存中会维护一些 liveExecutors、liveRDDs、liveJobs、liveStages 以及 liveTasks 等信息，当 Spark 历史服务器触发 compact 操作的时候，会读取需要 compact 的事件日志文件， 然后根据前面的 liveExecutors、liveRDDs、liveJobs、liveStages 以及 liveTasks 等信息判断哪些事件需要删除，哪些事件需要保留。满足 EventFilter 定义的 Event 会被保留，不满足的就删除，具体可以参见 EventFilter 的 applyFilterToFile 方法实现。

### 阶段二

AppStatusListener 会利用外部 KVStore 来存储事件日志，所以社区建议利用现有特性在底层 KVStore 中保留最多数量的 Jobs、Stages 以及 SQL 执行。为了存储对象到 KVStore 以及从 KVStore 恢复对象，社区采用的方法是将 KVStore 中存储的对象 dump 到一个文件中，这个称为 snapshot。

从空间使用的角度来看，这个想法非常有效，因为在 POC 中，只需 5MB 内存就可以将 KVStore 中的数据 dump 到文件中，其中回放了8.4GB 的事件日志。结果看起来很令人惊讶，但是很有意义，根据这个机制，在大多数情况下，dump 数据到文件需要的内存大小不太可能发生显著变化。

需要注意的是，快照里面的内容与当前事件日志文件的内容是不同的。因为这个快照文件是从 KVStore dump 出来的，这些对象不会按创建的顺序写入。我们可以压缩这些对象以节省空间和 IO 成本。在分析某个问题时，新产生的事件日志可能没什么用，这就需要读取和操作之前的事件日志文件。为了支持这种情况，需要将基本的 listener events 编写为原始格式，然后滚动事件日志文件，然后再将旧的事件日志保存到快照中，两种格式的文件共存。这样就满足我们之前的需求。由于快照的存在，事件日志的总体大小不会无限增长。

## 新的方案中 event log 是如何存储的呢

之前每个 Spark 作业的 event log 都是保存在单个文件里面，如果事件日志没有完成，会使用 .inprogress 后缀表示。新的 event log 方案会为每个 Spark 作业创建一个目录来保存，因为每个 Spark 作业可能会生成多个事件日志文件。事件日志的文件夹名称格式为：eventlog_v2_appId(_<

appattemptid>)。在事件日志文件夹里面存储的是对应作业的事件日志，日志文件名称格式为：events_<sequence>_<appid>(_<appattemptid>)(.<codec>)。

为了说明，我这里进行了一些测试，/data/iteblog/eventlogs 这个目录就是 spark.eventLog.dir 参数设置的值，下面是这个目录下的内容：

```
iteblog@www.iteblog.com:/data/iteblog/eventlogs|
⇒  ll
total 0
drwxrwx---  15 iteblog  wheel   480B  3  9 14:26 eventlog_v2_local-1583735123583
drwxrwx---   7 iteblog  wheel   224B  3  9 14:50 eventlog_v2_local-1583735259373

iteblog@www.iteblog.com:/data/iteblog/eventlogs/eventlog_v2_local-1583735259373|
⇒  ll
total 416
-rw-r--r--  1 iteblog  wheel    0B  3  9 14:27 appstatus_local-1583735259373.inprogress
-rwxrwx---  1 iteblog  wheel   64K  3  9 14:50 events_2_local-1583735259373.compact
-rwxrwx---  1 iteblog  wheel   102K  3  9 14:50 events_3_local-1583735259373
-rwxrwx---  1 iteblog  wheel   374B  3  9 14:50 events_4_local-1583735259373
```

可以看到，当 Spark 作业还没有完成的时候，会存在一个 appstatus_local-1583735259373.inprogress 的空文件，真正的事件日志是写到 events_x_local-1583735259373 文件里面。

这里再多说一下，Spark 还使用 HDFS 的 EC（erasure coding，参见过往记忆大数据的 Hadoop 3.0 纠删码(Erasure Coding)：节省一半存储空间）功能来进一步节省 event log 的大小。不过从 Spark 3.0 开始，默认写 event log 不启用 EC，原因是 HDFS EC 的 hflush() 或 hsync() 实现是不做任何操作的，也就意味着如果你的应用程序不完成，那么你在磁盘是看不到数据的。不过你如果实在需要 EC 功能，在 Spark 3.0 可以通过 spark.eventLog.allowErasureCoding 参数启用，具体参见 SPARK-25855