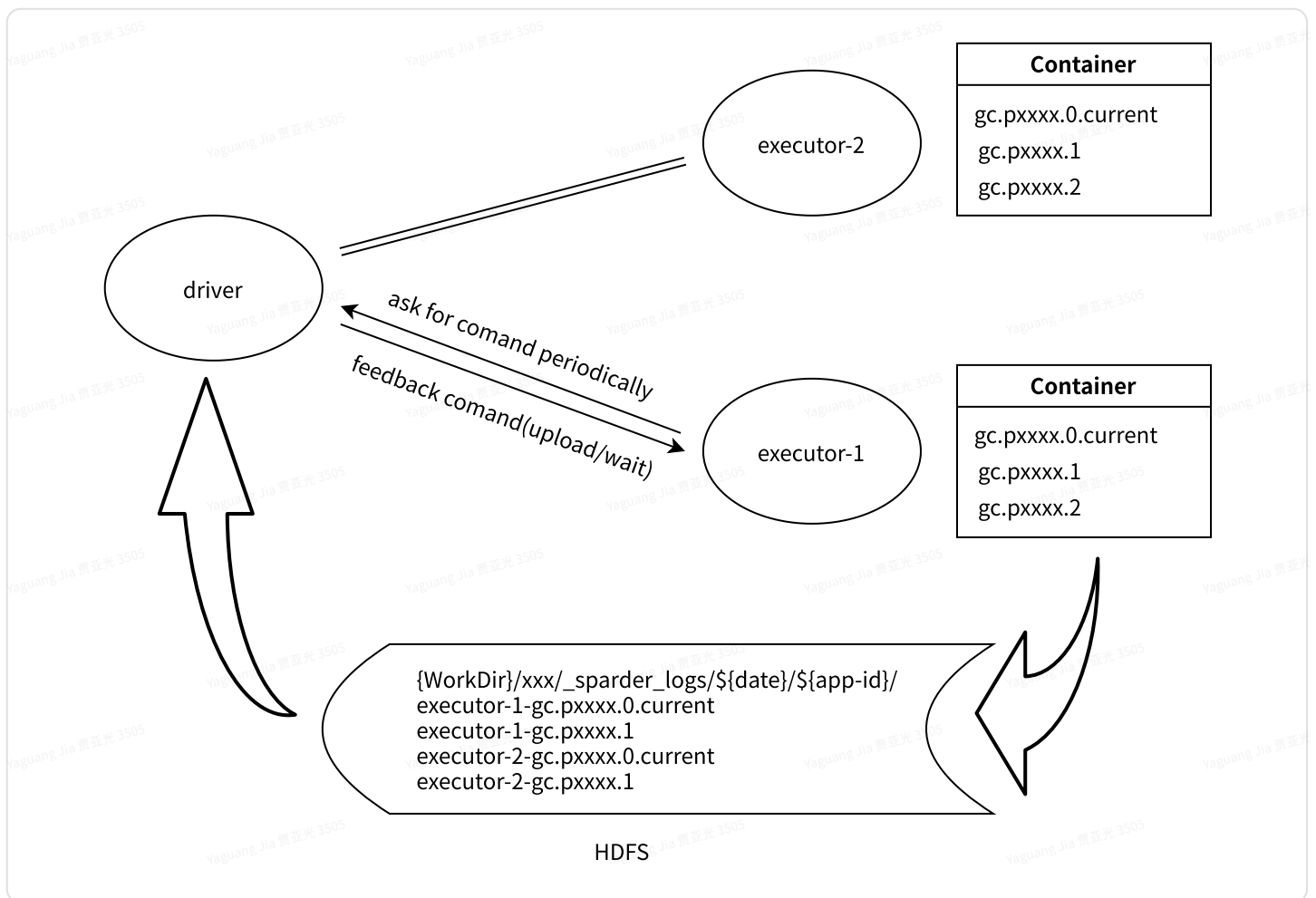


KYLIN-5528 Add Sparder (Query Spark) Executor's GC information to the diagnostic package

Dev Design



整体流程分为3个步骤

1. 在Executor启动时指定JVM参数，将 GC log 文件保存到 Executor 的 Container 中

参数指定了 gc 文件命名为 gc.pidxxxx.0.current，开启了日志文件滚动，单个gc 文件大小为64M，最多保留10个gc log，考虑到诊断包大小，保留1个gc log文件

```
1 // 添加到默认的 kylin.storage.columnar.spark-  
conf.spark.executor.extraJavaOptions 中
```



```
2 -XX:+PrintFlagsFinal -XX:+PrintReferenceGC -verbose:gc -XX:+PrintGCDetails -
  XX:+PrintGCTimeStamps -XX:+PrintGCDateStamps -XX:+PrintAdaptiveSizePolicy -
  XX:+UnlockDiagnosticVMOptions -XX:+UseGCLogFileRotation -
  XX:NumberOfGCLogFiles=1 -XX:GCLogFileSize=64M -Xloggc:gc.%p
```

2. KE在打诊断包时, Executor 将保存在Container本地的GC log 文件上传到HDFS的指定目录
在Executor上传GC log 时, 会扫描Container本地的文件, 过滤出"gc"开头的文件, 如
gc.pid60338.0.current, 然后上传到HDFS中 (如果有多个gc文件会上传多个gc文件到HDFS)

```
1 total 32
2 lrwxrwxrwx 1 yarn yarn      64 Feb 16 11:14 __spark_conf__ -> /data05/yarn/nm/user
3 lrwxrwxrwx 1 yarn yarn      83 Feb 16 11:14 __spark_libs__ -> /data05/yarn/nm/user
4 -rw-r--r-- 1 yarn yarn      91 Feb 16 11:14 container_tokens
5 -rw----- 1 yarn yarn    667 Feb 16 11:14 default_container_executor.sh
6 -rw----- 1 yarn yarn    612 Feb 16 11:14 default_container_executor_session.sh
7 -rw-r--r-- 1 yarn yarn 11785 Feb 16 11:19 gc.pid60338.0.current
8 -rw----- 1 yarn yarn   4384 Feb 16 11:14 launch_container.sh
9 lrwxrwxrwx 1 yarn yarn      77 Feb 16 11:14 libasynCProfiler-linux-arm64.so -> /da
10 lrwxrwxrwx 1 yarn yarn      75 Feb 16 11:14 libasynCProfiler-linux-x64.so -> /data
11 lrwxrwxrwx 1 yarn yarn      60 Feb 16 11:14 newten-job.jar -> /data05/yarn/nm/user
12 lrwxrwxrwx 1 yarn yarn      71 Feb 16 11:14 spark-appmaster-log4j.xml -> /data05/y
13 lrwxrwxrwx 1 yarn yarn      70 Feb 16 11:14 spark-executor-log4j.xml -> /data05/ya
```

3. 等待所有Executor的GC log 都上传好后, KE将GC log从HDFS中拷贝到诊断包中

```
1 |— spark_logs
2 |   |— application_1670689631385_1023
3 |       |— executor-1-gc.pid60335.0.current
4 |       |— executor-1.log
5 |       |— executor-2-gc.pid60338.0.current
6 |       |— executor-2.log
```

Driver 和 Executor 通信详细过程

```
1 // state
2 val STATE_WAIT = "STATE_WAIT"
3 val STATE_COLLECT = "STATE_COLLECT"
4
```



```
5 // executor message
6 val NEXTCMD = "NEXTCMD"
7 val SENDRESULT = "SENDRESULT"
8 val HDFSDIR = "HDFSDIR"
9
10 // driver message
11 val NOP = "NOP"
12 val COLLECT = "COLLECT"
13
14 // empty
15 val EMPTY = ""
```

初始状态

Driver => STATE_WAIT

Executor => STATE_WAIT

此时 Executor 发送NEXTCMD，会接收到Driver返回的 NOP 命令

开始打诊断包

Driver 端收到命令，将状态调整为 STATE_COLLECT

Executor 发送 NEXTCMD 给 Driver，会接收到Driver返回的 COLLECT 命令，调整自己状态为 STATE_COLLECT，并开始收集gc

收集完成

对于Executor 收集完成后，将状态调整为 STATE_WAIT

对于Driver，所有Executor 都收集完成后，将状态调整为STATE_WAIT，等待下次命令

Test Evidence

启动KE，设置Sparder executor个数为2（最终可以收集到两个gc log 文件）

```
kylin.storage.columnar.spark-conf.spark.driver.memory=1024m
kylin.storage.columnar.spark-conf.spark.executor.memory=1024m
kylin.storage.columnar.spark-conf.spark.yarn.am.memory=512m
kylin.storage.columnar.spark-conf.spark.executor.cores=2
kylin.storage.columnar.spark-conf.spark.executor.instances=2
```

到两个executor 的container 上找gc 文件


```
total 32
lrwxrwxrwx 1 yarn yarn 64 Feb 16 11:14 __spark_conf__ -> /data05/yarn/nm/usercache/root/filecache/3233/__spark_conf__.zip
lrwxrwxrwx 1 yarn yarn 83 Feb 16 11:14 __spark_libs__ -> /data05/yarn/nm/usercache/root/filecache/3232/__spark_libs__2968055886522187463.zip
-rw-r--r-- 1 yarn yarn 91 Feb 16 11:14 container_tokens
-rwx----- 1 yarn yarn 667 Feb 16 11:14 default_container_executor.sh
-rwx----- 1 yarn yarn 612 Feb 16 11:14 default_container_executor_session.sh
-rw-r--r-- 1 yarn yarn 11785 Feb 16 11:14 gc.pid60338.0.current
-rwx----- 1 yarn yarn 4384 Feb 16 11:14 launch_container.sh
lrwxrwxrwx 1 yarn yarn 77 Feb 16 11:14 libasynProfiler-linux-arm64.so -> /data05/yarn/nm/usercache/root/filecache/3236/libasynProfiler-linux-arm64.so
lrwxrwxrwx 1 yarn yarn 75 Feb 16 11:14 libasynProfiler-linux-x64.so -> /data05/yarn/nm/usercache/root/filecache/3231/libasynProfiler-linux-x64.so
lrwxrwxrwx 1 yarn yarn 60 Feb 16 11:14 newten-job.jar -> /data05/yarn/nm/usercache/root/filecache/3235/newten-job.jar
lrwxrwxrwx 1 yarn yarn 71 Feb 16 11:14 spark-appmaster-log4j.xml -> /data05/yarn/nm/usercache/root/filecache/3234/spark-appmaster-log4j.xml
lrwxrwxrwx 1 yarn yarn 70 Feb 16 11:14 spark-executor-log4j.xml -> /data05/yarn/nm/usercache/root/filecache/3230/spark-executor-log4j.xml
drwx--x--- 2 yarn yarn 106 Feb 16 11:15 tmp

-----
lrwxrwxrwx 1 yarn yarn 64 Feb 16 11:14 __spark_conf__ -> /data05/yarn/nm/usercache/root/filecache/3233/__spark_conf__.zip
lrwxrwxrwx 1 yarn yarn 83 Feb 16 11:14 __spark_libs__ -> /data05/yarn/nm/usercache/root/filecache/3232/__spark_libs__2968055886522187463.zip
-rw-r--r-- 1 yarn yarn 91 Feb 16 11:14 container_tokens
-rwx----- 1 yarn yarn 667 Feb 16 11:14 default_container_executor.sh
-rwx----- 1 yarn yarn 612 Feb 16 11:14 default_container_executor_session.sh
-rw-r--r-- 1 yarn yarn 12856 Feb 16 13:10 gc.pid60335.0.current
-rwx----- 1 yarn yarn 4384 Feb 16 11:14 launch_container.sh
lrwxrwxrwx 1 yarn yarn 77 Feb 16 11:14 libasynProfiler-linux-arm64.so -> /data05/yarn/nm/usercache/root/filecache/3236/libasynProfiler-linux-arm64.so
lrwxrwxrwx 1 yarn yarn 75 Feb 16 11:14 libasynProfiler-linux-x64.so -> /data05/yarn/nm/usercache/root/filecache/3231/libasynProfiler-linux-x64.so
lrwxrwxrwx 1 yarn yarn 60 Feb 16 11:14 newten-job.jar -> /data05/yarn/nm/usercache/root/filecache/3235/newten-job.jar
lrwxrwxrwx 1 yarn yarn 71 Feb 16 11:14 spark-appmaster-log4j.xml -> /data05/yarn/nm/usercache/root/filecache/3234/spark-appmaster-log4j.xml
lrwxrwxrwx 1 yarn yarn 70 Feb 16 11:14 spark-executor-log4j.xml -> /data05/yarn/nm/usercache/root/filecache/3230/spark-executor-log4j.xml
drwx--x--- 2 yarn yarn 106 Feb 16 11:15 tmp
```

然后打全量诊断包和查询诊断包，到HDFS上确认gc文件上传成功

application_1670689631385_1023 Go! 📁 ⬆️ 📄

Show 25 entries Search:

<input type="checkbox"/>	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	<input type="checkbox"/>
<input type="checkbox"/>	-rw-f--f--	root	supergroup	11.32 KB	Feb 16 11:17	3	128 MB	executor-1-gc.pid60335.0.current	<input type="checkbox"/>
<input type="checkbox"/>	-rw-f--f--	root	supergroup	29.48 KB	Feb 16 11:13	3	128 MB	executor-1.log	<input type="checkbox"/>
<input type="checkbox"/>	-rw-f--f--	root	supergroup	11.51 KB	Feb 16 11:17	3	128 MB	executor-2-gc.pid60338.0.current	<input type="checkbox"/>
<input type="checkbox"/>	-rw-f--f--	root	supergroup	29.72 KB	Feb 16 11:13	3	128 MB	executor-2.log	<input type="checkbox"/>

Showing 1 to 4 of 4 entries Previous 1 Next

下载诊断包，到 spark_logs 文件夹下确认gc 文件是否存在

全量诊断包

EXPLORER

FULL_2023_02_16_11_15_42

client

conf

hadoop_conf

job_tmp

logs

metadata

monitor_metrics

rec_candidate

snapshot_auto_refresh

sparder_history

spark_logs/2023-02-16

application_1670689631385_1001

application_1670689631385_1022

application_1670689631385_1023

executor-1-gc.pid60335.0.current

executor-1.log

executor-2-gc.pid60338.0.current

executor-2.log

system_metrics

system_usage

catalog_info

commit_SHA1

hadoop_env

info

kylin_env

time_used_info

executor-2-gc.pid60338.0.current

spark_logs > 2023-02-16 > application_1670689631385_1023 > executor-2-gc.pid60338.0.current

1

OpenJDK 64-Bit Server VM (25.352-b08) for linux-amd64 JRE (1.8.0_352-b08), built

2

Memory: 4k page, physical 131614760k(69446092k free), swap 4194300k(4194300k free)

3

CommandLine flags: -XX:GCLogFileSize=67108864 -XX:InitialHeapSize=1073741824 -XX

4

2023-02-16T11:15:00.309+0800: 1.437: [GC (Metadata GC Threshold) 2023-02-16T11:1

5

AdaptiveSizeStart: 1.462 collection: 1

6

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 5368709

7

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.016854 major_

8

AdaptiveSizeStop: collection: 1

9

[PSYoungGen: 131081K->14841K(305664K)] 131081K->14929K(1005056K), 0.0246613 secs

10

2023-02-16T11:15:00.334+0800: 1.462: [Full GC (Metadata GC Threshold) 2023-02-16

11

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 3748250

12

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.016854 major_

13

PSAdaptiveSizePolicy::compute_old_gen_free_space: costs minor_time: 0.016854 maj

14

AdaptiveSizePolicy::old generation size: collection: 2 (716177408) -> (445644800

15

AdaptiveSizeStop: collection: 2

16

[PSYoungGen: 14841K->0K(305664K)] [ParOldGen: 88K->14000K(435200K)] 14929K->1400

17

2023-02-16T11:15:01.893+0800: 3.021: [GC (Metadata GC Threshold) 2023-02-16T11:1

18

AdaptiveSizeStart: 3.035 collection: 3

19

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 3591667

20

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.013107 major_

21

AdaptiveSizeStop: collection: 3

22

[PSYoungGen: 229200K->13572K(305664K)] 243201K->27580K(740864K), 0.0148618 secs]

23

2023-02-16T11:15:01.908+0800: 3.036: [Full GC (Metadata GC Threshold) 2023-02-16

24

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 3523407

25

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.013107 major_

26

PSAdaptiveSizePolicy::compute_old_gen_free_space limits: desired_promo_size: 727

27

PSAdaptiveSizePolicy::compute_old_gen_free_space: costs minor_time: 0.013107 maj

28

AdaptiveSizePolicy::old generation size: collection: 4 (445644800) -> (716177408

29

AdaptiveSizeStop: collection: 4

30

[PSYoungGen: 13572K->0K(305664K)] [ParOldGen: 14008K->17463K(699392K)] 27580K->1

31

2023-02-16T11:15:09.411+0800: 10.539: [GC (Allocation Failure) 2023-02-16T11:15:

32

AdaptiveSizeStart: 10.564 collection: 5

查询诊断包

EXPLORER

QUERY_2023_02_16_11_2...

client

conf

hadoop_conf

logs

metadata

spark_logs/2023-02-16

application_1670689631385_1022

application_1670689631385_1023

executor-1-gc.pid60335.0.current

executor-1.log

executor-2-gc.pid60338.0.current

executor-2.log

catalog_info

commit_SHA1

hadoop_env

info

kylin_env

time_used_info

executor-1-gc.pid60335.0.current

spark_logs > 2023-02-16 > application_1670689631385_1023 > executor-1-gc.pid60335.0.current

1

OpenJDK 64-Bit Server VM (25.352-b08) for linux-amd64 JRE (1.8.0_352-b08), built on Oct 21 202

2

Memory: 4k page, physical 131614760k(69449564k free), swap 4194300k(4194300k free)

3

CommandLine flags: -XX:GCLogFileSize=67108864 -XX:InitialHeapSize=1073741824 -XX:MaxDirectMemo

4

2023-02-16T11:15:00.280+0800: 1.413: [GC (Metadata GC Threshold) 2023-02-16T11:15:00.295+0800:

5

AdaptiveSizeStart: 1.435 collection: 1

6

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 536870912 old_eden_si:

7

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.015275 major_cost: 0.000000

8

AdaptiveSizeStop: collection: 1

9

[PSYoungGen: 131084K->14803K(305664K)] 131084K->14891K(1005056K), 0.0218913 secs] [Times: user:

10

2023-02-16T11:15:00.302+0800: 1.435: [Full GC (Metadata GC Threshold) 2023-02-16T11:15:00.309+

11

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 393622342 old_eden_si:

12

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.015275 major_cost: 0.017479

13

PSAdaptiveSizePolicy::compute_old_gen_free_space: costs minor_time: 0.015275 major_cost: 0.017

14

AdaptiveSizePolicy::old generation size: collection: 2 (716177408) -> (426770432)

15

AdaptiveSizeStop: collection: 2

16

[PSYoungGen: 14803K->0K(305664K)] [ParOldGen: 88K->13918K(416768K)] 14891K->13918K(722432K), [I

17

2023-02-16T11:15:01.906+0800: 3.038: [GC (Metadata GC Threshold) 2023-02-16T11:15:01.914+0800:

18

AdaptiveSizeStart: 3.054 collection: 3

19

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 380737905 old_eden_si:

20

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.012572 major_cost: 0.017479

21

AdaptiveSizeStop: collection: 3

22

[PSYoungGen: 229224K->13502K(305664K)] 243143K->27428K(722432K), 0.0160742 secs] [Times: user=

23

2023-02-16T11:15:01.922+0800: 3.054: [Full GC (Metadata GC Threshold) 2023-02-16T11:15:01.928+

24

PSAdaptiveSizePolicy::compute_eden_space_size limits: desired_eden_size: 374858894 old_eden_si:

25

PSAdaptiveSizePolicy::compute_eden_space_size: costs minor_time: 0.012572 major_cost: 0.019139

26

PSAdaptiveSizePolicy::compute_old_gen_free_space: costs minor_time: 0.012572 major_cost: 0.019

27

AdaptiveSizePolicy::old generation size: collection: 4 (426770432) -> (679477248)

28

AdaptiveSizeStop: collection: 4

Test suggestion

1. 启动KE，打诊断包，分别检查 Executor 的 Container、HDFS 和诊断包中是否有GC 文件
2. 更改Executor 个数，查看GC文件个数是否正确
3. ~~调小-XX:GCLogFileSize=64M 参数，查看Container中的GC文件达到GCLogFileSize后是否会滚动，然后打诊断包，查看是否都能收集到诊断包中，废弃，现在调整为收集一个gc 文件。~~

Limitation

由于收集Executor gc 的这个操作需要手动触发，所以打诊断包时，目前只能收集到当前正在运行的 sparder 的gc 日志