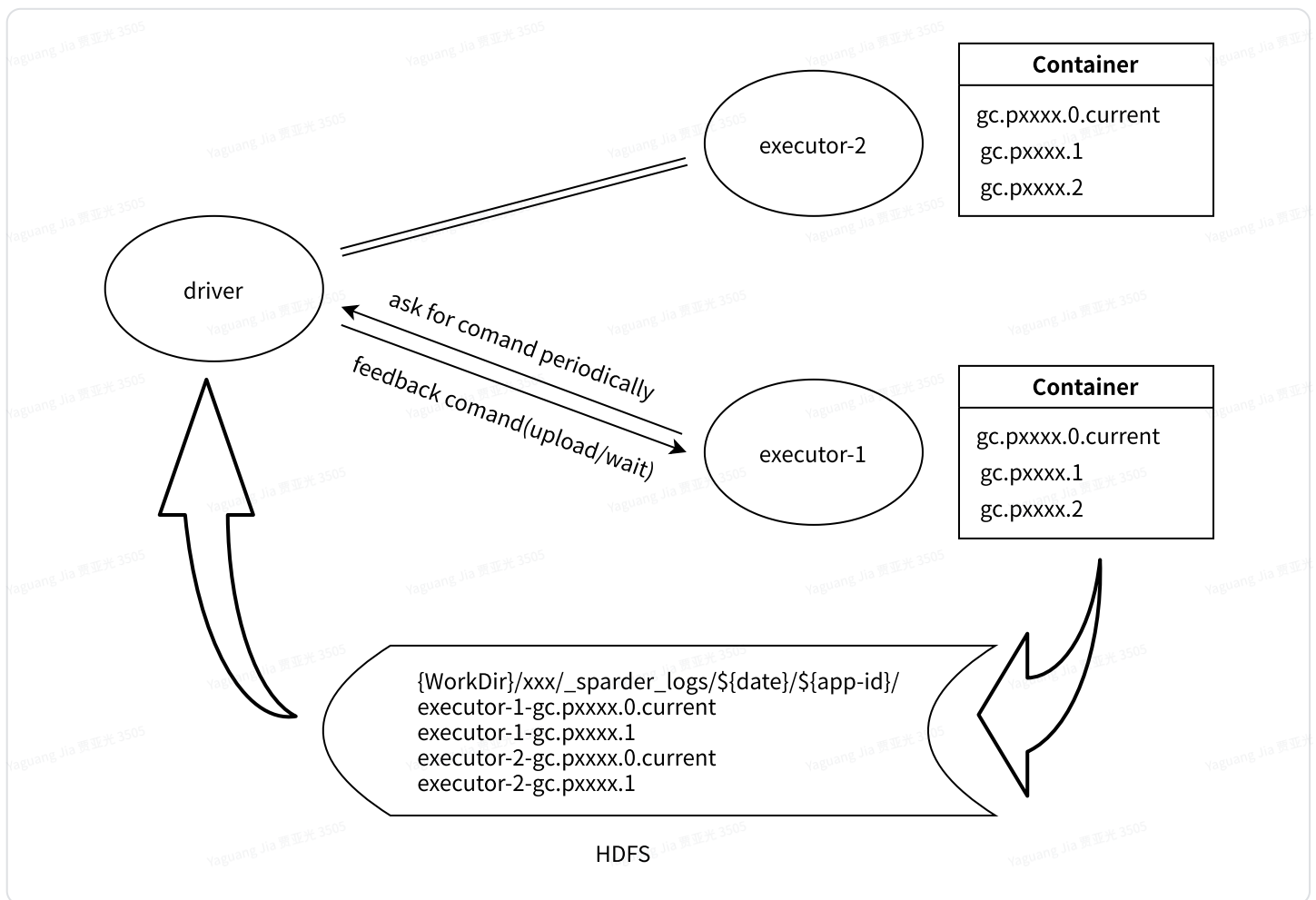


# KYLIN-5528 Add Sparder (Query Spark) Executor's GC information to the diagnostic package

## Dev Design



## The overall process is divided into 3 steps

1. Specify JVM parameters when the Executor starts, save the GC log file to the Executor's Container

The parameter specifies that the gc file is named gc.pidxxxx.0.current, the log file scrolling is enabled, the size of a single gc file is 64M, up to 10 gc logs are reserved, 1 gc log file is reserved considering the size of the diagnostic package

```
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 When typing the diagnostic package, the Executor uploads the GC log file saved locally in the Container to the specified directory in HDFS

When the Executor uploads the GC log, it will scan the local files of the Container, filter out the files starting with "gc", such as gc.pid60338.0.current, and then upload them to HDFS (if there are multiple gc files, multiple gc files will be uploaded to 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 libasynProfiler-linux-arm64.so -> /da  
10 lrwxrwxrwx 1 yarn yarn 75 Feb 16 11:14 libasynProfiler-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. After all the Executor GC logs are uploaded, KE copies the GC logs from HDFS to the diagnostic package

```
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 and Executor Communication Detailed Procedure

```

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 = ""

```

## Initial state

Driver => STATE\_WAIT

Executor => STATE\_WAIT

At this time, the Executor sends NEXTCMD and will receive the NOP command returned by the Driver

## Start the diagnostic kit

The driver side receives the command and adjusts the state to STATE\_COLLECT

The executor sends NEXTCMD to the Driver, receives the COLLECT command returned by the Driver, adjusts its status to STATE\_COLLECT, and starts collecting gc

## Collection is complete

For Executor collection is complete, adjust the state to STATE\_WAIT

For Driver, after all Executors are collected, adjust the state to STATE\_WAIT and wait for the next command

## Test Evidence

Start KE and set the number of Sparder executors to 2 (eventually two gc log files can be collected)

```
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
```

Find the gc file in the containers of the two executors




```
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 libasyncProfiler-linux-arm64.so -> /data05/yarn/nm/usercache/root/filecache/3236/libasyncProfiler-linux-arm64.so
lrwxrwxrwx 1 yarn yarn 75 Feb 16 11:14 libasyncProfiler-linux-x64.so -> /data05/yarn/nm/usercache/root/filecache/3231/libasyncProfiler-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 libasyncProfiler-linux-arm64.so -> /data05/yarn/nm/usercache/root/filecache/3236/libasyncProfiler-linux-arm64.so
lrwxrwxrwx 1 yarn yarn 75 Feb 16 11:14 libasyncProfiler-linux-x64.so -> /data05/yarn/nm/usercache/root/filecache/3231/libasyncProfiler-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
```

Then hit the full diagnostic package and query the diagnostic package, and confirm that the gc file is uploaded successfully on HDFS

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-r--r--	root	<a href="#">supergroup</a>	11.32 KB	Feb 16 11:17	<a href="#">3</a>	128 MB	<a href="#">executor-1-gc.pid60335.0.current</a>	<input type="checkbox"/>
<input type="checkbox"/>	-rw-r--r--	root	<a href="#">supergroup</a>	29.48 KB	Feb 16 11:13	<a href="#">3</a>	128 MB	<a href="#">executor-1.log</a>	<input type="checkbox"/>
<input type="checkbox"/>	-rw-r--r--	root	<a href="#">supergroup</a>	11.51 KB	Feb 16 11:17	<a href="#">3</a>	128 MB	<a href="#">executor-2-gc.pid60338.0.current</a>	<input type="checkbox"/>
<input type="checkbox"/>	-rw-r--r--	root	<a href="#">supergroup</a>	29.72 KB	Feb 16 11:13	<a href="#">3</a>	128 MB	<a href="#">executor-2.log</a>	<input type="checkbox"/>

Showing 1 to 4 of 4 entries

Previous

1

Next

Download the diagnostic package and go to the `spark_logs` folder to confirm whether the gc file exists

Full diagnostic package



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

## Query diagnostic package

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. Start KE, type the diagnostic package, and check whether there are GC files in the Executor's Container, HDFS and diagnostic package respectively
2. Change the number of executors to check whether the number of GC files is correct
3. ~~Adjust the small-XX: GCLogFileSize = 64M parameter to see if the GC file in the Container will scroll after reaching GCLogFileSize, and then hit the diagnostic package to see if it can be collected into the diagnostic package, discarded, now adjusted to collect a gc file.~~

## Limitation

Since this operation of collecting Executor gc needs to be triggered manually, when typing the diagnostic package, only the gc log of the currently running sparder can be collected at present