在压测过程中使用jmap、jstack、jstat、jconsole和jmc对gateway进行分析。

1.jmap

数据展示

```
C:\Program Files\Java\jdk1.8.0_201\bin>jmap -heap 12164
Attaching to process ID 12164, please wait...
Debugger attached successfully.
Server compiler detected.
JVM version is 25.201-b09

using thread-local object allocation.
Parallel GC with 6 thread(s)

Heap Configuration:
   MinHeapFreeRatio         = 0
   MaxHeapFreeRatio         = 100
   MaxHeapSize              = 4242538496 (4046.0MB)
   NewSize                  = 88604672 (84.5MB)
   MaxNewSize               = 1414004736 (1348.5MB)
   OldSize                  = 177733632 (169.5MB)
   NewRatio                 = 2
   SurvivorRatio            = 8
   MetaspaceSize            = 21807104 (20.796875MB)
   CompressedClassSpaceSize = 1073741824 (1024.0MB)
   MaxMetaspaceSize         = 17592186044415 MB
   G1HeapRegionSize         = 0 (0.0MB)

Heap Usage:
PS Young Generation
Eden Space:
   capacity = 294125568 (280.5MB)
   used     = 288508696 (275.1433334350586MB)
   free     = 5616872 (5.356666564941406MB)
   98.09031495011001% used
From Space:
   capacity = 14680064 (14.0MB)
   used     = 2231288 (2.1279220581054688MB)
   free     = 12448776 (11.872077941894531MB)
   15.19944327218192% used
To Space:
   capacity = 15204352 (14.5MB)
   used     = 0 (0.0MB)
   free     = 15204352 (14.5MB)
   0.0% used
PS Old Generation
   capacity = 149422080 (142.5MB)
   used     = 18329568 (17.480438232421875MB)
   free     = 131092512 (125.01956176757812MB)
   12.26697419819079% used

18458 interned Strings occupying 2370672 bytes.
```

分析

1.堆信息:

堆的最大大小：4046.0MB

新生代默认大小：84.5MB，最大大小：1348.5MB

老年代大小：169.5MB

2.堆内存:

年轻代:

Eden区，总容量280.5MB， 已使用92%

surviorE区，From区（总容量14.0MB）使用 15%，To区（总容量14.0MB）为空

老年代：总容量142.5MB，已使用12.2%

2.jstack

数据展示

```
#内容过多，部分内容已省略。
C:\Program Files\Java\jdk1.8.0_201\bin>jstack -l 12164
2021-01-14 15:21:45
Full thread dump Java HotSpot(TM) 64-Bit Server VM (25.201-b09 mixed mode):

"RMI TCP Connection(5)-192.168.22.75" #41 daemon prio=5 os_prio=0
tid=0x000000001ed44000 nid=0x7fa0 in Object.wait() [0x000000002686c000]
   java.lang.Thread.State: TIMED_WAITING (on object monitor)
        at java.lang.Object.wait(Native Method)
        - waiting on <0x000000076c1c7fc0> (a
com.sun.jmx.remote.internal.ArrayNotificationBuffer)
        at
com.sun.jmx.remote.internal.ArrayNotificationBuffer.fetchNotifications(Unknown
Source)
        - locked <0x000000076c1c7fc0> (a
com.sun.jmx.remote.internal.ArrayNotificationBuffer)
        at
com.sun.jmx.remote.internal.ArrayNotificationBuffer$ShareBuffer.fetchNotificatio
ns(Unknown Source)
        at com.sun.jmx.remote.internal.ServerNotifForwarder.fetchNotifs(Unknown
Source)
        at javax.management.remote.rmi.RMIConnectionImpl$4.run(Unknown Source)
        at javax.management.remote.rmi.RMIConnectionImpl$4.run(Unknown Source)
        at
javax.management.remote.rmi.RMIConnectionImpl.fetchNotifications(Unknown Source)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(Unknown Source)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(Unknown Source)
        at java.lang.reflect.Method.invoke(Unknown Source)
        at sun.rmi.server.UnicastServerRef.dispatch(Unknown Source)
        at sun.rmi.transport.Transport$1.run(Unknown Source)
        at sun.rmi.transport.Transport$1.run(Unknown Source)
        at java.security.AccessController.doPrivileged(Native Method)
        at sun.rmi.transport.Transport.serviceCall(Unknown Source)
        at sun.rmi.transport.tcp.TCPTransport.handleMessages(Unknown Source)
        at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run0(Unknown
Source)
        at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.lambda$run$0(Unknown
Source)
        at
sun.rmi.transport.tcp.TCPTransport$ConnectionHandler$$Lambda$334/124623357.run(U
nknown Source)
        at java.security.AccessController.doPrivileged(Native Method)
        at sun.rmi.transport.tcp.TCPTransport$ConnectionHandler.run(Unknown
Source)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(Unknown Source)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(Unknown Source)
        at java.lang.Thread.run(Unknown Source)

   Locked ownable synchronizers:
        - <0x000000076bd7e4e0> (a
java.util.concurrent.ThreadPoolExecutor$Worker)

...

"http-nio-8088-exec-1" #18 daemon prio=5 os_prio=0 tid=0x000000001ed42000
nid=0x4498 waiting on condition [0x00000000269ef000]
   java.lang.Thread.State: WAITING (parking)
        at sun.misc.Unsafe.park(Native Method)
        - parking to wait for  <0x00000006c39852a8> (a
java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject)
        at java.util.concurrent.locks.LockSupport.park(Unknown Source)
        at
java.util.concurrent.locks.AbstractQueuedSynchronizer$ConditionObject.await(Unkn
own Source)
        at java.util.concurrent.LinkedBlockingQueue.take(Unknown Source)
        at org.apache.tomcat.util.threads.TaskQueue.take(TaskQueue.java:103)
        at org.apache.tomcat.util.threads.TaskQueue.take(TaskQueue.java:31)
        at java.util.concurrent.ThreadPoolExecutor.getTask(Unknown Source)
        at java.util.concurrent.ThreadPoolExecutor.runWorker(Unknown Source)
        at java.util.concurrent.ThreadPoolExecutor$Worker.run(Unknown Source)
        at
org.apache.tomcat.util.threads.TaskThread$WrappingRunnable.run(TaskThread.java:6
1)
        at java.lang.Thread.run(Unknown Source)

   Locked ownable synchronizers:
        - None

...

"VM Thread" os_prio=2 tid=0x000000001c858000 nid=0x708 runnable

"GC task thread#0 (ParallelGC)" os_prio=0 tid=0x0000000003066800 nid=0x7398
runnable

"GC task thread#1 (ParallelGC)" os_prio=0 tid=0x0000000003068000 nid=0x7e04
runnable
```

```
"GC task thread#2 (ParallelGC)" os_prio=0 tid=0x000000000306a000 nid=0x2cdc
runnable

"GC task thread#3 (ParallelGC)" os_prio=0 tid=0x000000000306c800 nid=0x56bc
runnable

"GC task thread#4 (ParallelGC)" os_prio=0 tid=0x000000000306d800 nid=0x9e90
runnable

"GC task thread#5 (ParallelGC)" os_prio=0 tid=0x000000000306f800 nid=0x934
runnable

"VM Periodic Task Thread" os_prio=2 tid=0x000000001e315800 nid=0x700 waiting on
condition

JNI global references: 939
```

分析

线程守护线程http-nio-8088-exec-1正处于等待状态，有6个并行GC线程正在运行，

3.jstat

数据展示

```
#内容过长，部分内容省略
C:\Program Files\Java\jdk1.8.0_201\bin>jstat -gcutil 12164 1s 50
  S0     S1     E      O      M     CCS    YGC     YGCT    FGC    FGCT     GCT
  0.00  10.78  70.71  12.27  94.48  90.83      9    0.050     2    0.061
 0.111
  0.00  10.78  71.48  12.27  94.48  90.83      9    0.050     2    0.061
 0.111
  0.00  10.78  72.25  12.27  94.48  90.83      9    0.050     2    0.061
 0.111
  ...
  0.00  10.78  98.87  12.27  94.48  90.83      9    0.050     2    0.061
 0.111
  0.00  10.78  99.45  12.27  94.48  90.83      9    0.050     2    0.061
 0.111
  0.00  10.78  99.97  12.27  94.48  90.83      9    0.050     2    0.061
 0.111
 44.64   0.00   3.24  12.27  94.64  90.84     10    0.052     2    0.061
 0.113
 44.64   0.00   3.31  12.27  94.64  90.84     10    0.052     2    0.061
 0.113
 ...
 44.64   0.00   8.10  12.27  94.64  90.84     10    0.052     2    0.061
 0.113
 44.64   0.00   8.38  12.27  94.64  90.84     10    0.052     2    0.061
 0.113
```

分析

1.起始时，Eden区使用了70.71%,Survivor区的数据在s1区，使用了10.78%，old区使用了12.27%，Young GC累计执行了9次，总耗时0.05秒，Full GC执行了2次，总耗时0.061秒，垃圾回收总耗时0.111秒。

2.随着程序的运行，新对象不断增加，Eden区的以用容量不断增加，其他区域容量不变。

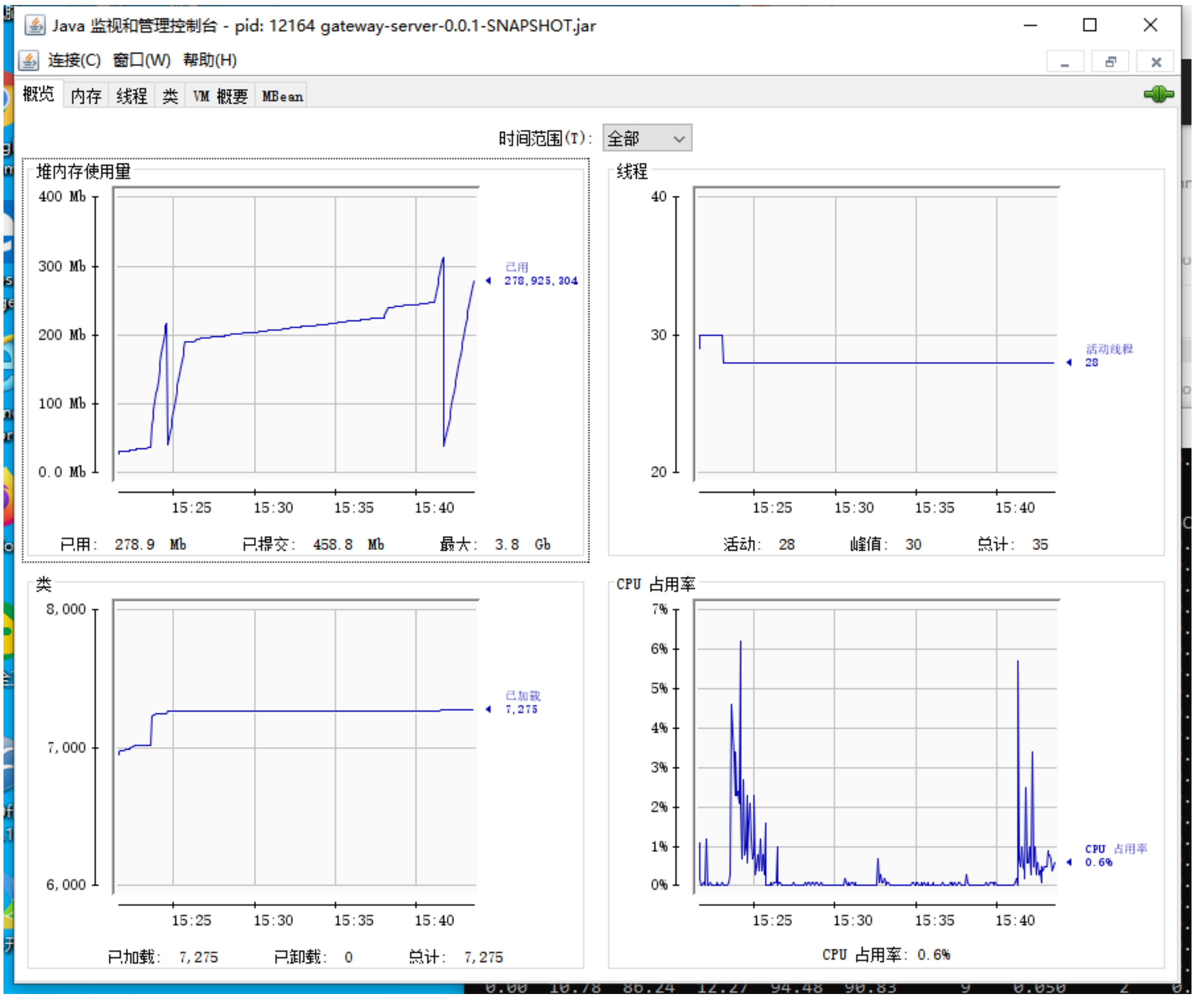
3.当Eden区的容量超过99.97%后，发生了一次Young GC，Eden区的存活对象和S1区的存活对象被复制到了s0区，Eden区和s1区清空。Young GC累次次数和总耗时增加到10次，0.052s，垃圾回收总耗时随之增加。

4.压测还没结束，Eden区的对象还在不断新增。

4.jconsole

数据展示

在15：24和15：42左右进行了压测。

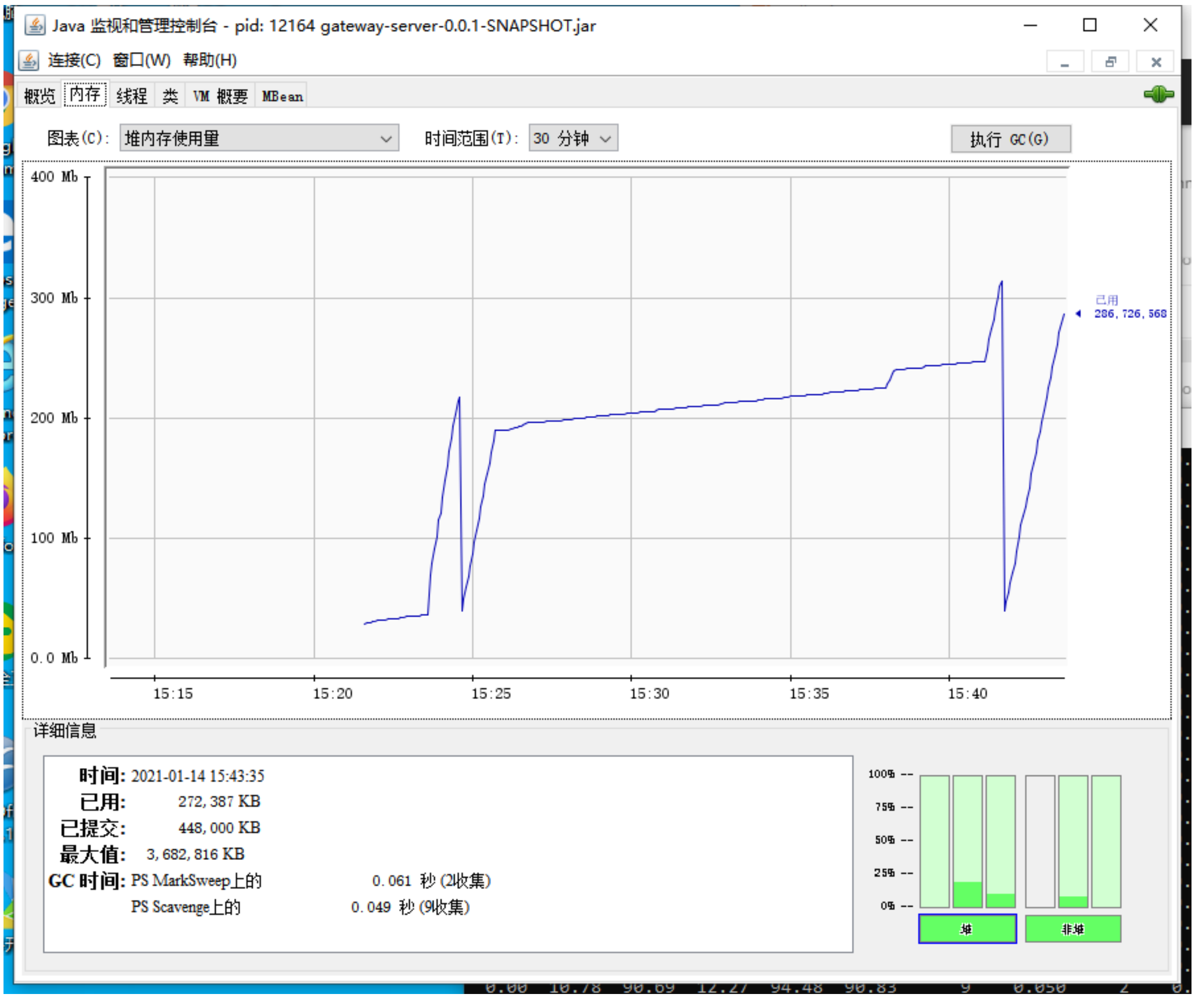


已用: 278.9 Mb 已提交: 458.8 Mb 最大: 3.8 Gb

活动: 28 峰值: 30 总计: 35

已加载: 7,275 已卸载: 0 总计: 7,275

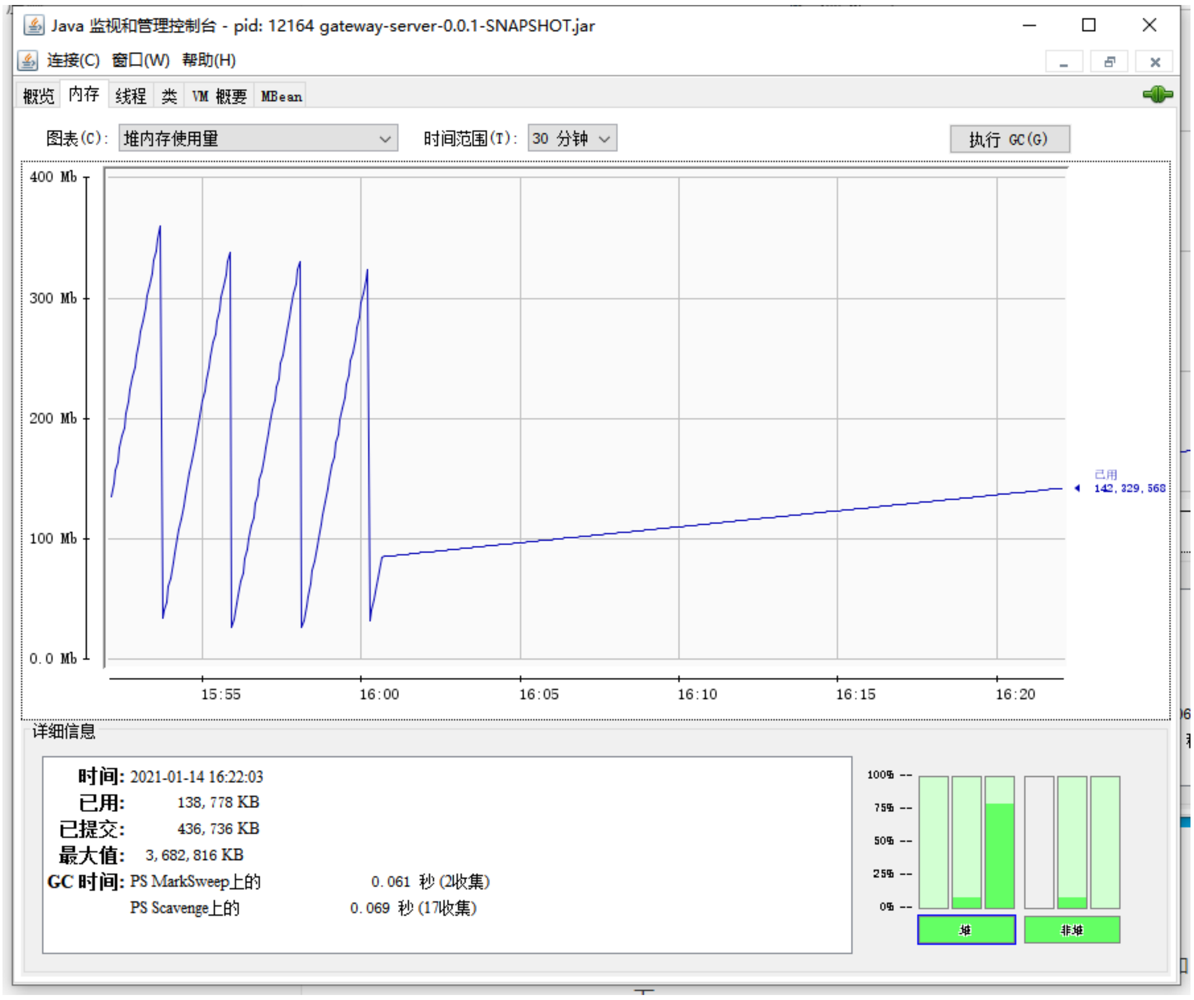
CPU 占用率: 0.6%



图表(C): 堆内存使用量　时间范围(T): 30 分钟

详细信息

```
      时间: 2021-01-14 15:43:35
      已用:      272, 387 KB
    已提交:      448, 000 KB
    最大值:    3, 682, 816 KB
   GC 时间: PS MarkSweep上的             0.061 秒 (2收集)
            PS Scavenge上的            0.049 秒 (9收集)
```



图表(C): 堆内存使用量　时间范围(T): 30 分钟

详细信息

```
      时间: 2021-01-14 16:22:03
      已用:      138, 778 KB
    已提交:      436, 736 KB
    最大值:    3, 682, 816 KB
   GC 时间: PS MarkSweep上的             0.061 秒 (2收集)
            PS Scavenge上的            0.069 秒 (17收集)
```

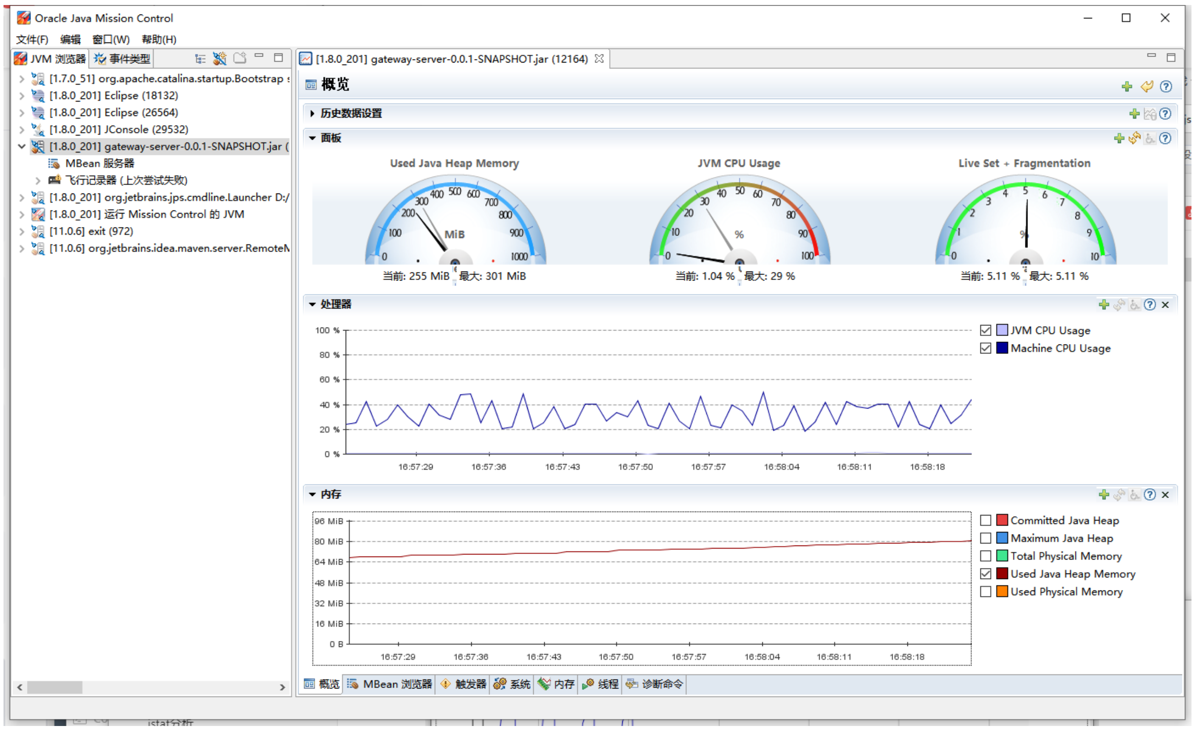
分析

以15：42分的测试为例，在开始时，堆内存使用量不断增加，cpu占用率也在增加，堆内存使用量超过300MB后进行了一次GC，使堆内存使用量降低到50MB以下。16：00分压测结束，其间发生了多次GC，由详细信息可知，本次压测，年轻代共发生了8次GC，老年代没有发生GC。

5.jmc

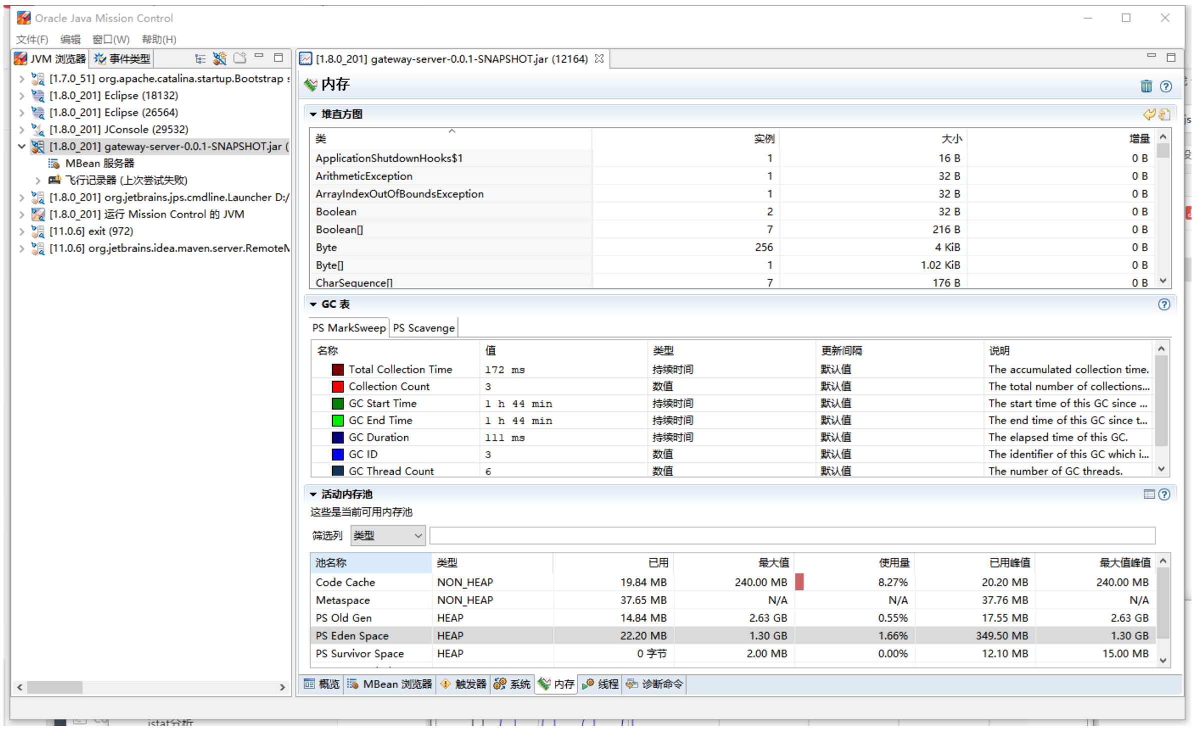
数据展示

概览



Oracle Java Mission Control
文件(F) 编辑 窗口(W) 帮助(H)
JVM 浏览器
事件类型
[1.7.0_51] org.apache.catalina.startup.Bootstrap
[1.8.0_201] Eclipse (18132)
[1.8.0_201] Eclipse (26564)
[1.8.0_201] JConsole (29532)
[1.8.0_201] gateway-server-0.0.1-SNAPSHOT.jar (
MBean 服务器
飞行记录器 (上次尝试失败)
[1.8.0_201] org.jetbrains.jps.cmdline.Launcher D:/
[1.8.0_201] 运行 Mission Control 的 JVM
[11.0.6] exit (972)
[11.0.6] org.jetbrains.idea.maven.server.RemoteM
[1.8.0_201] gateway-server-0.0.1-SNAPSHOT.jar (12164)
概览
历史数据设置
面板
Used Java Heap Memory
MiB
当前: 255 MiB 最大: 301 MiB
JVM CPU Usage
%
当前: 1.04 % 最大: 29 %
Live Set + Fragmentation
当前: 5.11 % 最大: 5.11 %
处理器
JVM CPU Usage
Machine CPU Usage
内存
Committed Java Heap
Maximum Java Heap
Total Physical Memory
Used Java Heap Memory
Used Physical Memory
概览 MBean 浏览器 触发器 系统 内存 线程 诊断命令

压测中的内存信息1



Oracle Java Mission Control

[1.8.0_201] gateway-server-0.0.1-SNAPSHOT.jar (12164)

内存

堆直方图

类	实例	大小	增量
ApplicationShutdownHooks$1	1	16 B	0 B
ArithmeticException	1	32 B	0 B
ArrayIndexOutOfBoundsException	1	32 B	0 B
Boolean	2	32 B	0 B
Boolean[]	7	216 B	0 B
Byte	256	4 KiB	0 B
Byte[]	1	1.02 KiB	0 B
CharSequence[]	7	176 B	0 B

GC 表

PS MarkSweep | PS Scavenge

名称	值	类型	更新间隔	说明
Total Collection Time	172 ms	持续时间	默认值	The accumulated collection time.
Collection Count	3	数值	默认值	The total number of collections...
GC Start Time	1 h 44 min	持续时间	默认值	The start time of this GC since ...
GC End Time	1 h 44 min	持续时间	默认值	The end time of this GC since t...
GC Duration	111 ms	持续时间	默认值	The elapsed time of this GC.
GC ID	3	数值	默认值	The identifier of this GC which i...
GC Thread Count	6	数值	默认值	The number of GC threads.

活动内存池

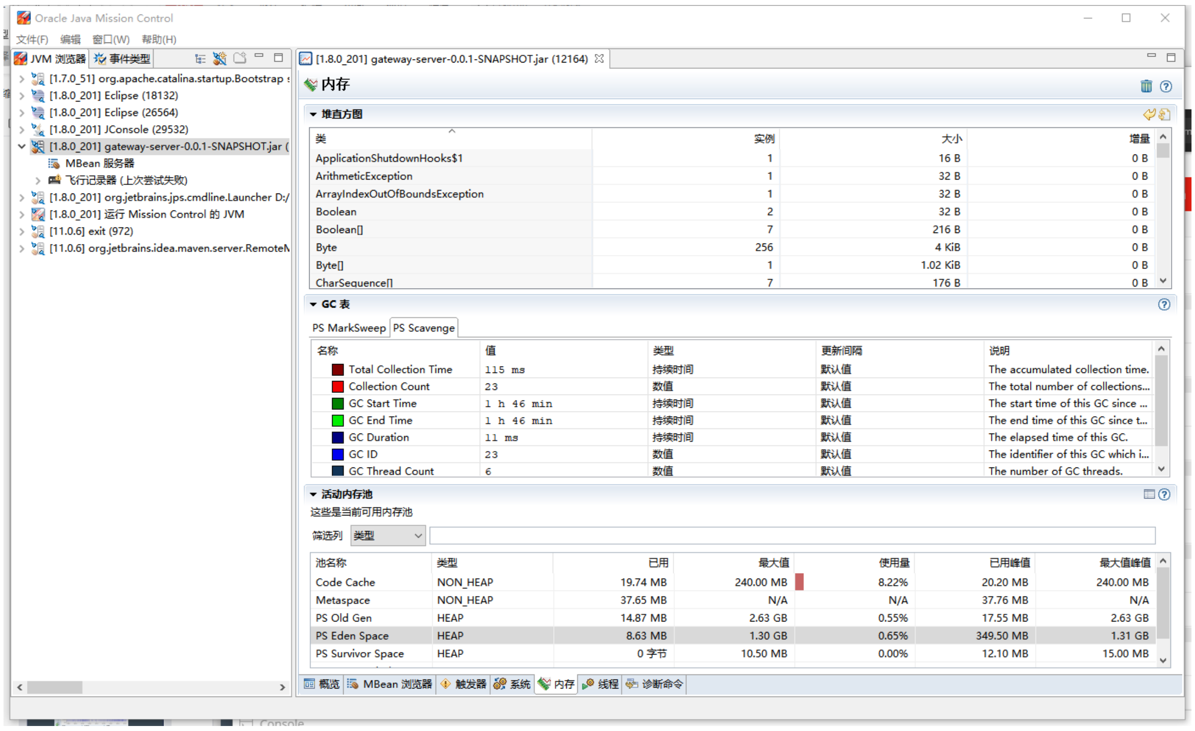
这些是当前可用内存池

筛选列 类型

池名称	类型	已用	最大值	使用量	已用峰值	最大值峰值
Code Cache	NON_HEAP	19.84 MB	240.00 MB	8.27%	20.20 MB	240.00 MB
Metaspace	NON_HEAP	37.65 MB	N/A	N/A	37.76 MB	N/A
PS Old Gen	HEAP	14.84 MB	2.63 GB	0.55%	17.55 MB	2.63 GB
PS Eden Space	HEAP	22.20 MB	1.30 GB	1.66%	349.50 MB	1.30 GB
PS Survivor Space	HEAP	0 字节	2.00 MB	0.00%	12.10 MB	15.00 MB

概览 MBean 浏览器 触发器 系统 内存 线程 诊断命令

压测中的内存信息2



Oracle Java Mission Control

文件(F) 编辑 窗口(W) 帮助(H)

JVM 浏览器 | 事件类型

- [1.7.0_51] org.apache.catalina.startup.Bootstrap
- [1.8.0_201] Eclipse (18132)
- [1.8.0_201] Eclipse (26564)
- [1.8.0_201] JConsole (29532)
- [1.8.0_201] gateway-server-0.0.1-SNAPSHOT.jar (
  - MBean 服务器
  - 飞行记录器 (上次尝试失败)
- [1.8.0_201] org.jetbrains.jps.cmdline.Launcher D:/
- [1.8.0_201] 运行 Mission Control 的 JVM
- [11.0.6] exit (972)
- [11.0.6] org.jetbrains.idea.maven.server.RemoteN

[1.8.0_201] gateway-server-0.0.1-SNAPSHOT.jar (12164)

内存

堆直方图

类	实例	大小	增量
ApplicationShutdownHooks$1	1	16 B	0 B
ArithmeticException	1	32 B	0 B
ArrayIndexOutOfBoundsException	1	32 B	0 B
Boolean	2	32 B	0 B
Boolean[]	7	216 B	0 B
Byte	256	4 KiB	0 B
Byte[]	1	1.02 KiB	0 B
CharSequence[]	7	176 B	0 B

GC 表

PS MarkSweep | PS Scavenge

名称	值	类型	更新间隔	说明
Total Collection Time	115 ms	持续时间	默认值	The accumulated collection time.
Collection Count	23	数值	默认值	The total number of collections...
GC Start Time	1 h 46 min	持续时间	默认值	The start time of this GC since ...
GC End Time	1 h 46 min	持续时间	默认值	The end time of this GC since t...
GC Duration	11 ms	持续时间	默认值	The elapsed time of this GC.
GC ID	23	数值	默认值	The identifier of this GC which i...
GC Thread Count	6	数值	默认值	The number of GC threads.

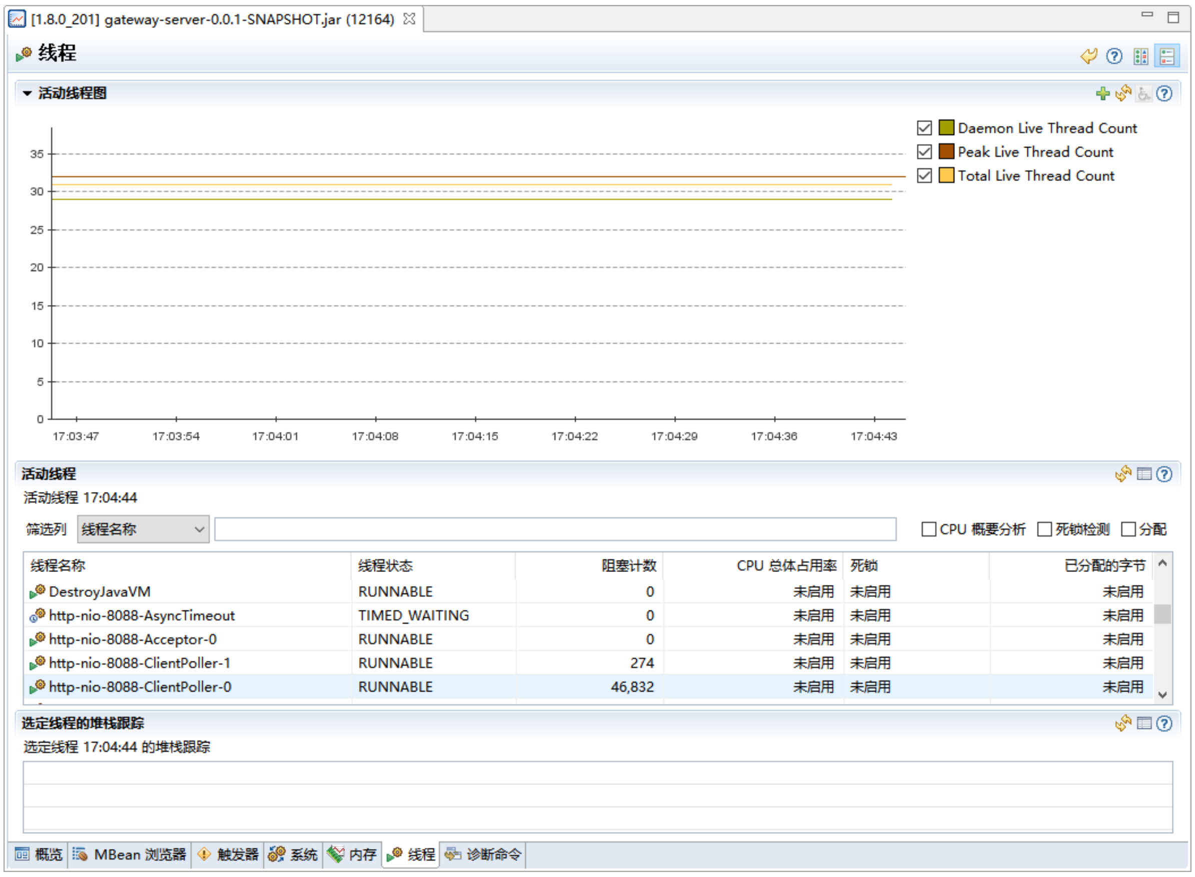
活动内存池

这些是当前可用内存池

筛选列 类型

池名称	类型	已用	最大值	使用量	已用峰值	最大值峰值
Code Cache	NON_HEAP	19.74 MB	240.00 MB	8.22%	20.20 MB	240.00 MB
Metaspace	NON_HEAP	37.65 MB	N/A	N/A	37.76 MB	N/A
PS Old Gen	HEAP	14.87 MB	2.63 GB	0.55%	17.55 MB	2.63 GB
PS Eden Space	HEAP	8.63 MB	1.30 GB	0.65%	349.50 MB	1.31 GB
PS Survivor Space	HEAP	0 字节	10.50 MB	0.00%	12.10 MB	15.00 MB

线程信息



[1.8.0_201] gateway-server-0.0.1-SNAPSHOT.jar (12164)

线程

活动线程图

活动线程

活动线程 17:04:44

筛选列 线程名称 ☐ CPU 概要分析 ☐ 死锁检测 ☐ 分配

线程名称	线程状态	阻塞计数	CPU 总体占用率	死锁	已分配的字节
DestroyJavaVM	RUNNABLE	0	未启用	未启用	未启用
http-nio-8088-AsyncTimeout	TIMED_WAITING	0	未启用	未启用	未启用
http-nio-8088-Acceptor-0	RUNNABLE	0	未启用	未启用	未启用
http-nio-8088-ClientPoller-1	RUNNABLE	274	未启用	未启用	未启用
http-nio-8088-ClientPoller-0	RUNNABLE	46,832	未启用	未启用	未启用

选定线程的堆栈跟踪

选定线程 17:04:44 的堆栈跟踪

分析

在概览页面可知，堆内存的使用量在不断增加，但cpu利用率不高。

内存信息图1中，Old区使用了14.84MB，最大容量是2.63G，老年代空间还有很多，Eden区使用了22.20MB，使用量是1.66%。GC表中，老年代的GC次数是3次，启动了6个GC线程，年轻代发生了22次GC(未截取到图片)。

内存信息图2中，Old区使用了14.87MB，Eden区使用了8.63MB。GC表中，老年代的GC次数是3次，启动了6个GC线程，年轻代发生了23次GC，启动了6个GC线程。

在图1到图2的过程中发生了一次年轻代GC，有0.03MB的对象，从年轻代复制到了老年代中，Eden区的数据清除后对象还在增加。

根据线程信息图，我们可以知道在17:04:44时刻，线程http-nio-8088-Async处于限时等待状态，http-nio-8088-ClientPoller-0和http-nio-8088-ClientPoller-1处于运行状态。