原问题分析: java.lang.OutOfMemoryError:unable to create new native thread

标签: JAVA jvm 线程

2015-09-08 09:52 🔍 444人阅读 🖵 评论(0) 收藏 举报

版权声明:本文为博主原创文章,未经博主允许不得转载。

目录(?)	[-]
1. 背景 2. 分析 3. 结论	

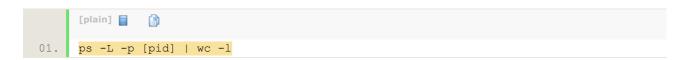
背景

后台系统抛出OOM错误,操作无法正常进行。调用后台日志,发现错误信息

为: java.lang.OutOfMemoryError:unable to create new native thread

分析

- 1、一般<mark>OOM比较常见的原因</mark>是<u>堆内存不够</u>或者<u>持久代内存不够</u>导致,但<u>这个问题是因为<mark>无法创建本地线程</u>。</u></mark>
- 2、最初怀疑JVM的堆内存或持久代内存耗尽了,<u>使用<mark>jmap</mark>查看JVM内存</u>信息,发现一切正常。(有可能jmap 等命令根本无法执行。)
- 3、<u>通过free、top</u>命令<u>查看系统内存,发现系统物理内存<mark>耗尽了</mark>。使用jmap</u>命令将JVM的信息dump到文件,使用JVisualVM分析,发现有<mark>大量的线程处于等待状态</mark>,并且这些线程都是ThreadPool。通过执行下面命令,可以获知该进程下的线程数:



发现线程数量有几千个,线程数量没有达到操作系统限制的数量,但是内存不够用了。

4、<u>根据获得的线索,怀疑代码里对线程池</u>没有正确使用。分析代码后发现,<mark>线程池对象</mark>没有使用static标识, 导致每次在new一个新的对象时就会创建一个线程池。由于使用的是<mark>50个线程的固定线程池,每次任务运行完</mark> 后,线程并不会关闭,而是处于等待状态。因此,<u>这样</u>一样,调用次数过多,就会导致创建很多个线程,直至 OOM。 线程池对象持有的线程泄漏引发OOM

结论

线程池对象持有的线程泄漏引发OOM 线程池对象必须声明为类变量

- 1、其实分析中的第二步没有必要进行。分析<mark>JVM的内存区域</mark>:堆内存、持久代、虚拟机栈、<u>本地方法栈</u>、 Direct内存。我们用jmap只能查看堆内存和持久代内存。<mark>线程</mark>直接使用的是<u>系统物理内存</u>,用jmap无法看出来。
- 2、我们在配置<mark>-Xms、-Xmx</mark>参数时,虽然<u>指定了<mark>堆内存</mark>的大小,但是JVM在启动时,并不会一下申请这么多</u> 内存。而是在创建新的对象时,找操作系统申请内存。同时,在GC后,JVM并不会将申请下来的内存返回给

第1页 共2页 2017/5/14 下午4:54

操作系统,而是由它自身来进行管理。

3、<mark>堆内存配置过大,JVM可用的剩余物理内存就会少</mark>。因此,需要<u>综合考虑系统的情况,合理分配JVM的堆内存,不然就容易出现各类OOM</u>了。本次遇到的这个问题就是因为<mark>配置有问题</mark>:<u>操作系统共有4G的内存,结果堆内存就设置了3G,如果JVM把3G的内存吃完了,也就剩下1G的内存供<mark>线程、缓冲区</mark>等其他用户使用,所以明显容易出问题。</u>

第2页 共2页 2017/5/14 下午4:54