链滴

一次线上 java 应用响应时间过长问题的排查

作者：xinzhongtianxia
原文链接：https://ld246.com/article/1569574881351
来源网站：链滴

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfile
com/bing/20190203.jpg?imageView2/1/w/960/h/540/interlace/1/q/100">
最近接手一个老 java 应用，没多久接到响应时间太长的报警，整个排查过程还是挺有意思的，
录一下。
整个过程中，设计到 cpu，内存，垃圾回收，引用，spring, 单例 等等知识，整个下来，心情愉悦。<
p>
<h2 id="接到报警">接到报警</h2>
吃完晚饭回来，接到报警短信，服务响应时间太长，达到 2s 以上。
第一反应，怎么可能，这个应用很简单，就提供了几个查询接口，QPS 单机也就 10 左右，居然响应
间 2s 以上，这不科学。
<h2 id="看监控">看监控</h2>
看监控平台，有问题的机器，cpu 占用很高，这。。。更不科学了。理论上，这应用 cpu 能到 1
就不错了，现在居然 80%，吓得我赶紧跑到机器上，top 了一把，确实是 java 应用占用 cpu 很高。
/p>
<h2 id="找出占用cpu的线程">找出占用 cpu 的线程</h2>
<pre><code class="highlight-chroma"><span class="highlight
cl">ps -mp pid -o THREAD,tid,time
</code></pre>
发现 cpu 几乎全被一个线程占用了，将线程 id 转换成 16 进制
<pre><code class="highlight-chroma"><span class="highlight
cl">printf "%x\n" tid
</code></pre>
打印线程堆栈信息
<pre><code class="highlight-chroma"><span class="highlight
cl">jstack pid |grep tid -A 30
</code></pre>
看到的结果居然是，CMS 垃圾收集器的线程。。。
这，这，这不科学。
<h2 id="垃圾回收">垃圾回收</h2>
看垃圾回收日志，发现一直在进行 Full GC，但是几乎没啥效果，GC 完了，老年代依然是几乎
满的状态。
即使是 CMS，也不能避免传说中的 stop the world，所以响应时间变长了。
但是，现在问题来了，Full GC 回收不掉的对象都是啥啥啥？
<h2 id="分析java堆内存">分析 java 堆内存</h2>
先把线上流量从这台机器切走，然后 dump，分析后发现，一共 160 多万个 PoolHttpConnecti
nManager 对象占领了绝大多数内存。
会议了一下代码，和 http 请求有关的只有 oss 存储的 client 了。
猜测是 new oss client 时的问题
<h2 id="oss-client-源码">oss client 源码</h2>
看了一下应用中的代码，发现每次使用 oss 时都会 new OssClient，用完之后没有 shutdown。
br>
但是，new 是在方法内部的，方法内的局部变量按道理说，方法结束后直接就可以回收的啊，这不科
，肯定是 ossClient 的源码中搞了鬼。于是直接去源码中看了一下。
发现在 new OssClient 时，源码中会新建上面提到的 PoolHttpConnectionManager，然后调用 oss
lient 存取数据时，PoolHttpConnectionManager 会从内部维护的 http 连接池中找到可用连接，进
操作。
这没问题，大多是框架都是这么设计的。
既然 JVM 堆内存中有 160 多万个 PoolHttpConnectionManager 对象，说明肯定是有地方把内次 n
w OssClient 时新建的 PoolHttpConnectionManager 给缓存起来了，导致垃圾回收无能为力。<br

继续看源码，发现确实和我想的一样。给存起来了,只有显式调用 ossClient 的 shutdown，才会清理
缓存。
想想这样设计也没问题，这和线程池的思想是一样的，我们使用线程池时，肯定不会每次都 new 一
新的，只不过这个 ossClient 隐藏的比较深。。。

<h2 id="解决">解决</h2>
解决办法很简单，把 OssClient 改成单例就行了。
但是，集团封网期间不允许上线，这。。。
没办法，把应用重启一遍，JVM 内存满血复活，清醒 QPS 很低，重启后可以撑过封网的时间断了</
>
<h2 id="Spring的Controller是单例的">Spring 的 Controller 是单例的</h2>
突然想到，其他应用也使用了这个 OssClient，一直没问题，想必是单例吧，忍不住上去瞅了瞅
码，发现，是在一个 Controller 中使用了成员变量，像这样：
<pre><code class="highlight-chroma"><span class="highlight
cl">@RestController
@RequestMappin
("/sdk/map/styles")
public class MyCo
troller {
    private OSSClie
t ossClient = new OSSClient();

    public put(Obje
t o) {
    ossClient.put(0);
    }
</code></pre>
我记得上学那会儿，捣腾 Struts 的时候，Action 并不是单例的啊，单里会有问题，难道 Spring
是单例了？
查了一下，确实，Spring 的 Controller，默认是单例。
想了想，单例也没啥问题，只要不在方法中去操作成员变量就可以了。
<h2 id="后记">后记</h2>
你所学的知识，都是有用的，认为没用，也许只是时候未到而已。