

Java常见问题排查

毕玄

2014.3



Java常见问题

- **NoSuchMethodException**
- **应用没响应**
- **调用另一应用超时**
- **java.lang.OutOfMemoryError**
- **CPU us高**
- **CPU sy高**
- **CPU iowait高**
- **Java进程退出**



Java常见问题排查

■ NoSuchMethodException

● 出现这种现象的原因

- Java ClassLoader机制
- Java里让人极度头疼的Jar版本冲突问题

● 同类型的问题

- ClassNotFoundException/NoClassDefFoundError/ClassCastException



Java常见问题排查

■ NoSuchMethodException

● 排查方法

■ `-XX:+TraceClassLoading`

■ `jar -tvf *.jar`



Java常见问题排查

■ NoSuchMethodException

● 解决方法

- mvn pom里去除不需要的版本<scope>provided</scope>
- 在打包阶段就尽可能避免掉版本冲突的问题



Java常见问题排查

■ 应用没响应

- 出现这种现象的典型原因
 - 资源被耗光（CPU、内存），这种后面再说
 - 死锁
 - 处理线程池耗光
- 表现出来可能是
 - HTTP响应返回499、502、504



Java常见问题排查

■ 应用没响应

● 排查方法

■ 死锁

- `jstack -l`
- 仔细看线程堆栈信息

■ 处理线程池耗光

- `jstack`
- 查看从请求进来的路径上经过的处理线程池中的线程状况
 - 例如netty io threadpool--->business threadpool



Java常见问题排查

■ 应用没响应

● 解决方法

■ 死锁

● 想办法解开锁

➤ 例如spring 3.1.14前的死锁bug

■ 处理线程池耗光

● 加大线程池 or 减小超时时间等



Java常见问题排查

■ 调用另一应用超时

● 出现这个现象的典型原因

■ 真心比较多

- 服务端响应慢
- 调用端或服务端GC频繁
- 调用端或服务端CPU消耗严重
- 反序列化失败
- 网络问题



Java常见问题排查

■ 调用另一应用超时

● 排查方法

- 查看服务端对应的日志和响应时间监控信息
- 查看调用端和服务端的gc log
- 查看调用端和服务端的CPU利用率
- 查看有没有反序列化失败的log
- 查看网络的重传率
- 利器
 - EagleEye



Java常见问题排查

■ 调用另一应用超时

● 解决方法

■ 按排查出来的原因针对性解决



Java常见问题排查

■ **java.lang.OutOfMemoryError**

- GC overhead limit exceeded
- Java Heap Space
- Unable to create new native thread
- PermGen Space
- Direct buffer memory
- Map failed
- request {} bytes for {}. Out of swap space?



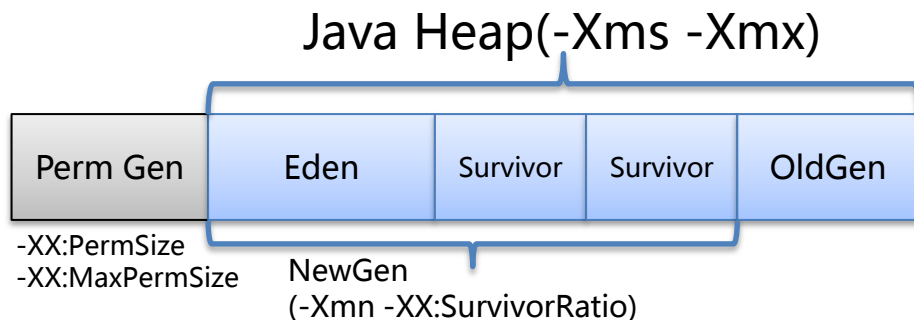
Java常见问题排查

■ java.lang.OutOfMemoryError

- GC overhead limit exceeded/Java Heap Space

■ 出现这个现象的原因

- Java Heap分配不出需要的内存了





Java常见问题排查

■ java.lang.OutOfMemoryError

- GC overhead limit exceeded/Java Heap Space

■ 排查方法（确定不是因为Heap Size大小的情况下）

- 拿到HeapDump文件

- -XX:+HeapDumpOnOutOfMemoryError

- jmap -dump:file=<文件名>,format=b [pid]

- gcore [pid]

- 分析HeapDump文件

- MAT – Dominator Tree

- 根据MAT分析的结果来定位到代码

- btrace



Java常见问题排查

■ java.lang.OutOfMemoryError

- GC overhead limit exceeded/Java Heap Space

■ 排查方法

- 分析HeapDump文件时可能会碰到

- 占用的内存并不多

- 分配了一个巨大的对象

- grep -i 'allocating large' 日志文件 (必须是ali jdk)

- 死循环

- jstack



Java常见问题排查

■ java.lang.OutOfMemoryError

- GC overhead limit exceeded/Java Heap Space

■ 解决方法

- 根据定位到的消耗了较多内存的代码，针对性的处理

➤ 例如

- 自增长的数据结构对象没限制大小
- 引用未释放，造成内存泄露



Java常见问题排查

■ java.lang.OutOfMemoryError

- GC overhead limit exceeded/Java Heap Space

■ 同类型的问题

- CMS GC频繁

- CMS GC频繁要注意还有可能是由于缺少一个参数

- -XX:+UseCMSInitiatingOccupancyOnly

- Full GC频繁

- 还有可能是因为悲观策略



Java常见问题排查

■ java.lang.OutOfMemoryError

● Unable to create new native thread

■ 出现这个现象的原因

● 线程数超过了ulimit限制

- 会导致执行ps等出现resource temporarily unavailable
- 出现时可以先临时调整下ulimit，以便能操作

● 线程数超过了kernel.pid_max

- 会导致执行ps等出现resource temporarily unavailable
- 只能重启

● 机器完全没有内存了

- 这种可能性通常很小，原因是...



Java常见问题排查

■ java.lang.OutOfMemoryError

● Unable to create new native thread

■ 排查方法

- `ps -eLf | grep java -c`
- `cat /proc/[pid]/limits`
 - 如果太小，可以调大点max open processes值
- `sysctl -a | grep kernel.pid_max`
- 如果真的是线程创建太多了
 - jstack 有线程名的话通常会比较好排查
 - btrace
 - `new Thread || new ThreadPoolExecutor`



Java常见问题排查

■ `java.lang.OutOfMemoryError`

- Unable to create new native thread

■ 解决方法

- 线程池并限制大小
- 对使用到的API一定要非常清楚
 - 例如很容易误用的netty client



Java常见问题排查

■ java.lang.OutOfMemoryError

● PermGen Space

■ 出现这个问题的原因

● PermGen被用满

➤ PermGen的使用和回收



Java常见问题排查

■ java.lang.OutOfMemoryError

● PermGen Space

■ 排查方法

● btrace

➤ `ClassLoader.defineClass`



Java常见问题排查

■ java.lang.OutOfMemoryError

● PermGen Space

■ 解决方法

- 确认是不是真的需要装载那么多，如果是

- 调大PermSize

- 如果不是

- 控制ClassLoader

- 常见于Groovy的误用



Java常见问题排查

■ java.lang.OutOfMemoryError

● Direct buffer memory

■ 出现这个现象的原因

● Direct ByteBuffer使用超出了限制的大小

➤ 默认的大小为-Xmx

➤ jinfo -flags

➤ Java中只能通过ByteBuffer.allocateDirect来使用Direct ByteBuffer



Java常见问题排查

■ java.lang.OutOfMemoryError

● Direct buffer memory

■ 排查方法

● btrace

➤ `ByteBuffer.allocateDirect`



Java常见问题排查

■ java.lang.OutOfMemoryError

● Direct buffer memory

■ 解决方法

- 如果真的是不够用，在内存够用的情况下可以调大
 - -XX:MaxDirectMemorySize
- 常见的是类似网络通信未做限流这种



Java常见问题排查

■ java.lang.OutOfMemoryError

● Map failed

■ 出现这个现象的原因

● FileChannel mapped的文件超出了限制

➤ vm.max_map_count



Java常见问题排查

■ java.lang.OutOfMemoryError

● Map failed

■ 排查方法

● btrace

➤ FileChannel.map

● 看看是不是加了-XX:+DisableExplicitGC参数



Java常见问题排查

■ java.lang.OutOfMemoryError

● Map failed

■ 解决方法

- 有必要的话调大vm.max_map_count
- 如map file存活个数其实不多则去掉-XX:+DisableExplicitGC
 - 在CMS GC的情况下，增加-XX:+ExplicitGCInvokesConcurrent



Java常见问题排查

■ java.lang.OutOfMemoryError

- request {} bytes for {}. Out of swap space?

- 出现这个现象的原因

- 地址空间不够用
- 物理内存耗光



Java常见问题排查

■ java.lang.OutOfMemoryError

- request {} bytes for {}. Out of swap space?

■ 排查方法

- 物理内存耗光

- 按经验

- btrace

- Deflater.init | Deflater.end | Inflater.init | Inflater.end

- 强制执行full gc

- jmap -histo:live [pid]

- 如果执行几次后内存明显下降，则基本是Direct ByteBuffer造成的

- google Perftools



Java常见问题排查

■ java.lang.OutOfMemoryError

- request {} bytes for {}. Out of swap space?

■ 解决方法

- 地址空间不够
 - 升级到64 bit
- 物理内存耗光
 - Inflater/Deflater问题的话则显式调用end
 - Direct ByteBuffer问题可以调小-XX:MaxDirectMemorySize
 - 其他case根据google perftools显示的来跟进



Java常见问题排查

■ CPU us高

- 出现这个现象的原因
 - CMS GC/Full GC频繁
 - 代码中出现非常耗CPU的操作
 - 整体代码的消耗



Java常见问题排查

■ CPU us高

● 排查方法

■ CMS GC/Full GC频繁

- 查看gc log , 或jstat -gcutil [pid] 1000 10

■ 代码中出现非常耗CPU的操作

- top -H + jstack , 做pid到nid的16进制转化

➤ printf '0x%x'

■ 整体代码的消耗

- top -H看到每个线程消耗都差不多, 而且不断变化

- perf -top

➤ 必须是ali版的perf和jdk



Java常见问题排查

■ CPU us高

● 解决方法

■ CMS GC/Full GC频繁

- 详见前面的内容

■ 代码中出现非常耗CPU的操作

- 通常需要进一步btrace
 - 正则匹配某些字符串造成cpu us高的case
- 也不一定很好处理
 - 覆盖Exception的getCause造成cpu us高的case

■ 整体代码的消耗

- 通常很难搞



Java常见问题排查

■ CPU sy高

● 出现这个现象的原因

- 锁竞争激烈

- 线程主动切换频繁

- 还有一个经验是

- linux 2.6.32后的高精度的问题



Java常见问题排查

■ CPU sy高

● 排查方法

■ jstack

- 看看锁状况
- 看看是不是有主动线程切换等

■ btrace

- `AbstractQueuedSynchronizer.ConditionObject.awaitNanos`



Java常见问题排查

■ CPU sy高

● 解决方法

■ 锁竞争激烈

- 根据业务实现要求合理做锁粒度控制，或引入无锁数据结构

■ 线程主动切换

- 改为通知机制

■ 高精度问题

- 至少调大到1ms+的await



Java常见问题排查

■ CPU iowait高

● 出现这个现象的原因

■ io读写操作频繁



Java常见问题排查

■ CPU iowait高

● 排查方法

■ 确认硬件状况

- 例如raid卡的cache策略

■ 借助系统工具

- blktrace+debugfs
- iotop
- btrace



Java常见问题排查

■ CPU iowait高

● 解决方法

- 提升dirty page cache
- cache
- 同步写转异步写
- 随机写转顺序写
- 实在搞不定
 - 昂贵的硬件



Java常见问题排查

■ Java进程退出

- 出现这个现象的原因
 - 原因非常的多



Java常见问题排查

■ Java进程退出

● 排查方法

- 查看生成的hs_err_pid[pid].log
- 确保core dump已打开, cat /proc/[pid]/limits
- dmesg | grep -i kill
- 根据core dump文件做相应的分析
 - gdb [java路径] core文件
 - c调试技巧



Java常见问题排查

■ Java进程退出

● crash demo

■ `jinfo -flag FLSLargestBlockCoalesceProximity <pid>`



Java常见问题排查

■ Java进程退出

● 常见的cases

- native stack溢出导致java进程退出的case
- 编译不了某些代码导致的Java进程退出的case
 - -
XX:CompileCommand=exclude,the/package/and/Class,methodName
- 内存问题导致的进程退出的case
- JVM自身bug导致退出的case



Java常见问题排查—总结

- 知其因
- 惟手熟尔

各种Java问题的cases排查过程大家可访问<http://hellojava.info>