# JVM 分析工具

去年的某一个发布日，我们发布了新版本，上线后发现程序变慢了很多，频繁 Full GC,然后OOM，问找了好久，后来导出OOM 现场日志发现PIO 一个错原来是报表导出功能有问题。后来就顺利解决了。

我们在生产环境经常遇到,程序发布后执行方法性能下降,JVM 崩溃,挂起,死锁,频繁在Full Gc.CPU使用过高 OOM.遇到这些问题我们怎么处理呢。遇到内存异常高负载，频繁Full GC 或者OOM 其实我们首先知道什么样的对象存在于内存之中，那个现存或对象导致了整个问题。JVM 内置了工具可以查看线程对象。

## Jstatd 命令

Jstatd是一个RMI的Server，它可以监视Hotspot的JVM 启动和结束。同时可以提供接口连接到远程机器的JVM.比如jpos jstat。

### 命令格式

**jstatd** [ *options* ]

### 实战

Jstatd-J-Djava.rmi.server.hostname=192.168.1.102 -J-Djava.security.policy=/export/home/server/java7/jdk1.7.0\_79/jre/lib/security/my.policy -p 2020 -n li -J-Djava.rmi.server.logCalls=true &

-J-Djava.rmi.server.hostname=192.168.1.102　授权给某个ip

-J-Djava.security.policy=/export/home/server/java7/jdk1.7.0\_79/jre/lib/security/my.policy 授权文件

-p 2020 端口

-n li 服务名

-J-Djava.rmi.server.logCalls=true　是否打印日志

my.policy 配置

grant codebase "file:${java.home}/../lib/tools.jar" {

permission java.security.AllPermission;

};

## Jps 命令

Jps 是jvm内置产看进程信息的小工具。

### ****命令格式****

jps [ options ] [ hostid ]

常用

jps 列出本地JVM进程

jps –l输出应用程序主类的完整包名，或者是应用程序JAR文件的完整路径。

Jps –v 输出启动参数

[protocol:][[//]hostname][:port][/servername]

protocol - 如果protocol及hostname都没有指定，那表示的是与当前环境相关的本地协议，如果指定了hostname却没有指定protocol，那么protocol的默认就是rmi。

hostname - 服务器的IP或者名称，没有指定则表示本机。

port - 远程rmi的端口，如果没有指定则默认为1099。

Servername - 注册到RMI注册中心中的jstatd的名称。

### 实战

jps rmi://10.37.129.3:12020/li

## JMAP 命令

Jmap(Java Memory Map) 打印java某个进程内存的所有“对象”的情况

**命令格式**

jmap [ option ] pid

jmap [ option ] executable core

jmap [ option ] [server-id@]remote-hostname-or-IP

基本参数

-dump:[live,]format=b,file=<filename> 使用hprof二进制形式,输出jvm的heap内容到文件=. live子选项是可选的，假如指定live选项,那么只输出活的对象到文件.

-finalizerinfo 打印正等候回收的对象的信息.

-heap 打印heap的概要信息，GC使用的算法，heap的配置及wise heap的使用情况.

-histo[:live] 打印每个class的实例数目,内存占用,类全名信息. VM的内部类名字开头会加上前缀”\*”. 如果live子参数加上后,只统计活的对象数量.

-permstat 打印classload和jvm heap长久层的信息. 包含每个classloader的名字,活泼性,地址,父classloader和加载的class数量. 另外,内部String的数量和占用内存数也会打印出来.

-F 强迫.在pid没有相应的时候使用-dump或者-histo参数. 在这个模式下,live子参数无效.

-h | -help 打印辅助信息

-J 传递参数给jmap启动的jvm.

pid 需要被打印配相信息的java进程id

实战

jmap -heap 4318

jmap –histo:live 4318

jmap -F -dump:format=b,file=tomcat.bin 4318

# jhat

用途：是用来分析java堆的命令，可以将堆中的对象以html的形式显示出来，包括对象的数量，大小等等，并支持对象查询语言

**命令格式**

参考文献

http://www.cnblogs.com/ggjucheng/archive/2013/04/16/3024986.html