# 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

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。

### 远程命令

jps rmi://10.37.129.3:12020/li

## JMAP 命令

Jmap 命令