使用JProfiler跟踪服务器性能

在实际项目中会遇到服务器响应非常慢的情况,有时候为了跟踪服务器的性能瓶颈就需要使用JProfiler进行跟踪。

要跟踪服务器性能则需要在服务器与客户端机器上面都安装有JProfiler,同是必须为同一个版本。并且在服务器的启动参数中添加相应的参数:

- Tomcat: startup.sh/cmd
- WebLogic: startWebLogic.sh/cmd
- WebSphere: 进入控制台,应用程序服务器 > server1 Java 和进程管理 进程定义 Java 虚拟机 , 启动 "调试方式"
- JBoss: run.sh/cmd

在这些文件中(WebSphere除外)的找到Java参数设置的参数名称,通常为JAVA_OPTS、JAVA_OPTIONS等,添加以下设置:

- Windows: SET JAVA_OPTS=%JAVA_OPTS% -agentlib:jprofilerti=port=8849
- Unix: JAVA_OPTS=\${JAVA_OPTS} -agentlib:jprofilerti=port=8849

同时需要再设置加载动态链接库的路径,在Windows中上PATH, Unix上为LD_LIBRARY_PATH, 添加以下设置:

- Windows: SET PATH=%PATH%;C:\jprofiler7.2.3\bin\windows-x64\
- Unix: export LD_LIBRARY_PATH=\${LD_LIBRARY_PATH}:/jprofiler7/bin/linux-x64

注意上述的路径要根据实际的安装目录进行修改,同时根据运行运行JDK的版本不同,相应的bin下的子目录也要修改。如果在启动服务器过程中显示异常:

Error occurred during initialization of VM Could not find agent library jprofilerti on the library path, with error: Can't find dependent libraries

则表示PATH或LD_LIBRARY_PATH不正确。

在设置参数后,启动服务器时则显示等待JProfiler连接的信息:

JProfiler>	Protocol version 37
JProfiler>	Using JVMTI
JProfiler>	JVMTI version 1.1 detected.
JProfiler>	64-bit library
JProfiler>	Listening on port: 8849.
JProfiler>	Instrumenting native methods.
JProfiler≻	Can retransform classes.
JProfiler>	Can retransform any class.
JProfiler>	Native library initialized
JProfiler>	VM initialized
JProfiler>	Waiting for a connection from the JProfiler GUI

这时必须在客户端使用JProfiler连接到服务器使服务器可以继续运行,在客户端中启动JProfiler并选择New Session,输入正确的连接地址:

Session Settings	
Application Settings Session Type	Code Editor & Compilation : New session Id:
Filter Settings	此性比較 Attach to an already running JVM and profile it Attach type: ③ Select from all local JVMs ④ Attach to profiled JVM (local or remote Launch a new JVM and profile it Launch type: ③ Application ③ Applet ◎ Web Start
Profiling Settings Iriggers Settings JEE & Probes Probes Probes Profiled JW Ihe profiled JW Ihe profiled JW Ihe profiled JW Ihe profiled JW Integration line utility Imeout:	Settings ag agent must already be running in the profiled JVM. This is usually done by running an wizard and restarting the JVM_To_avoid restarting. please invoke the jpenable command on the remote machine. 10.0.1 Profiling port: 8849 Default 20 seconds mand: wser with URL:
Java File Pa	th Note: the classpath is used for the bytecode viewer only.
Help <u>G</u> eneral Setti	ngs OK Cancel

再点击OK可以Attach 到服务器的JVM上并可以选择两种跟踪方式: Instrumentation和Sampling,前者功能更多包括支持计算方法调用次数等但是对性能有一 定影响,后者功能较少但是对性能影响较少,根据实际情况使用不同的方式。在连接成功后服务器的JVM会继续正常的启动过程。

在开始跟踪性能时,点击左边的CPU Views再点击工具栏的Record CPU跟踪,再次点击时会停止跟踪(注意,每次开始跟踪时会清空上一次的记录)

<u>S</u> ession <u>V</u> iew <u>P</u> rofiling <u>G</u> o To <u>W</u> indow <u>H</u> elp								
🤿 🚳 🔒	4	Ş	7	-	.	-		
Start Detach Save Center Snapshot	Emport	Fan GC	Add Boolmark	Fecore. Hemor:	Fecord CFU	Start Tacking		
۵.	Press		to record	CPV da	ta			
Memory Views								
-								
Heap Walker								
CPU Views								
June 2015 Thread Views								

跟踪后则在下方的列表中跟踪查看性能的瓶颈

Thread selection:	All thread groups	· Thread statu	: 💷 All states 👻			
Aggregation level:	Methods	View mode:	🔄 Iree 👻			
89.1%	- 700 s direct calls to methods of filtered classes					
🕀 🔟 🛯 6.6% - 51,76	9 ms java. lang. Thread. run					
🗄 🔟 1.8% - 14,28	5 ms smartbi.usc\$1.run					
i - 0 1.8% - 14,28	🔬 📶 1, 8% - 14, 285 ms smartbi, session, SessionThread, run					
🛓 🅘 0.3% - 2,354	ms URL: /smartbi/visior/css/bof_merge.css.jsp					
i						
🛛 🌑 0.1% - 769 ms URL: /smartbi/vision/						
🗄 🎱 0.1% - 434 m	s URL: /smartbi/pagenotfound.jsp					
i 🕀 🎱 0.0% – 4,972	μs URL: /smartbi/vision/css/login.css					
🗄 🕘 0.0% – 4,890	μs URL: /smartbi/vision/img/login/remember_message_on.gif					