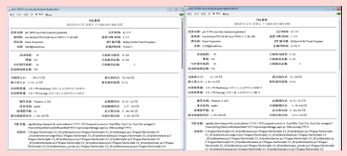
prometheus性能分析

测试环境配置

jvm配置：
两服务jvm配置均为-Xmx2048m -Xms512m -Xmn256m

机器配置：
操作系统:
Windows 11 10.0
体系结构: amd64
处理程序数: 16



两应用详细运行环境配置为上图

无prometheus相关代码

压测：
无逻辑接口：4000线程循环30次：无任何处理逻辑接口

获取指标接口：同时100线程循环5次：拉取prometheus指标信息

无逻辑接口：
平均响应时间：184ms（偏高因部分拉取prometheus指标接口对应用代来的影响）
min：0ms
max：3311ms（有部分是因为线程数过多，程序资源不足报错导致请求响应延时）

因资源线程开的过多，导致没办法同时检测拉取性能指标接口的响应时间（因为线程都在处理前面的压测，前面结束才进行该线程组的执行）在prometheus拉取性能做短量简单分析（因prometheus请求并没有非常频繁而且频率可控）

有prometheus相关代码

压测：
4000线程循环30次：无任何处理逻辑接口

平均响应时间：143ms
min：0ms
max：3310ms（有部分是因为线程数过多，程序资源不足报错导致请求响应延时）

获取prometheus性能指标接口性能

案例1：
压测：
无逻辑接口：1000线程循环10次：无任何处理逻辑接口

获取指标接口：同时20线程循环5次：拉取prometheus指标信息

无逻辑接口：
平均响应时间：0ms（也能看出前面是因为线程过多出现5xx才影响响应时间）
min：0ms
max：42ms

性能指标接口：
平均响应时间：2207ms（也能看出前面是因为线程过多出现5xx才影响响应时间）
min：685ms
max：2573ms

案例2：
压测：
无逻辑接口：500线程循环5次：无任何处理逻辑接口

获取指标接口：同时10线程循环3次：拉取prometheus指标信息

无逻辑接口：
平均响应时间：0ms
min：0ms
max：5ms

性能指标接口：
平均响应时间：932ms
min：173ms
max：1318ms

案例3：
仅测试获取指标接口

获取指标接口：同时10线程循环3次：拉取prometheus指标信息

性能指标接口：
平均响应时间：883ms
min：162ms
max：1318ms

结论

①在同等配置环境下，未达到高压情况，加入prometheus监控对应用程序的响应影响不大；

②prometheus本身获取性能指标的接口，需内部逻辑计算，耗时略长，可根据实际需要设置拉取频率。