25 不懂监控与调优，就像白天不懂夜的黑

更新时间：2019-10-12 11:33:36

> 富贵必从勤苦得。——杜甫

大家好，我是风落，是一名测试，也是半个开发，同时还偶尔客串下面试官。很突兀的重复介绍自己一半是怕大家忘记了我是谁，另外一半是想从 “客串面试官” 聊到今天的话题。粗算一下，大概我面试过的早就超过了三位数，但是在面试过程中，尤其是性能测试，给我一种很奇怪的感觉：大多数会用 LoadRunner 或者 Jmeter 的同学，认为自己是熟悉性能测试；而把工具用的比较不错，能截图给出工具的结果，分析下 TPS 的就认为自己是掌握了性能测试。这不是夸张，而是这些面试者真的这么认为，并且也认为性能测试自己能够给出通过或者不通过的结果，剩下的就不关自己事情了。

就像前边说过的，所谓工具使用真的只是性能测试的一部分，甚至是很小的一部分。上一回，我们谈了，我认为最重要的性能场景设计，今天我们来谈另一个话题：监控与调优。既然有了 “最

性能测试的本质

再说监控和调优之前，我们先来想一下我们进行性能测试的最终目的是什么？是给出性能结果么？

这当然是我们的目的之一，其实在我看来，本质上的目标主要分两个方面：

1. **有没有足够的能力。**这就是说我们来验证系统是不是符合我们想要的性能要求，比如我们预计的性能指标是 100w 存量用户、1w 同时在线、800 交易并发下能够控制响应时间在 3s 内，性能测试的目的，首先就是要看看现有情况下是否能够满足。
2. **能力的规划。**一般情况下很难一下子就满足，那么这就是怎么样才能让系统达到我们要求的性能能力。是不是可以通过增加服务器、修改配置、优化 SQL、修改程序等方式来提升系统能力，使性能达标。

在这两方面之中，更重要的就是后者，也就是通过监控调优的手段，让系统运行的更好，打破软件硬件方面的瓶颈。

系统可能存在的瓶颈

先来说说瓶颈，知己知彼方能百战不殆，再去监控它调试它之前我们先要做的就是了解它，知道什么地方可能出现问题。

从硬件的角度上说，一般可能存在瓶颈的点，包括磁盘空间不足，导致的运行速度下降、CPU、内存、磁盘 I/O 读写速率等方面。

软件的角度就包括多个方面，一方面是服务器方面的配置，比如由于 TOMCAT 或者 Weblogic 等中间件，连接池参数配置不合理会造成瓶颈；另一方面是应用自身，例如程序架构的不合理，代码不当引起的内存泄露、GC 不彻底等问题；最后一方面，应该算得上是性能测试里遇到瓶颈比较多的了，那就是数据库方面。包括数据库的索引、锁、不合理的表空间设计、慢 SQL 等等。

其实说起来应该还有一个地方，就是网络层面，但是由于我们的性能测试一般更多的是在局域网下进行，忽略网络影响，所以在测试过程中可以忽略。

性能测试的监控

由于上边提到的各方面的瓶颈，所以我们在进行性能测试的时候，一定要注意对软硬件方面的监

1. **操作系统监控**：Windows 服务器 - Perfmon；Linux 服务器 - TOP / Nmon /netstat 等
2. **数据库性能监控**：Oracle - Spotlight on Oracle； Mysql - MySQLMTOP
3. **TOMCAT 监控**：Lambda Probe
4. **JVM 监控**： Jmap / Jstack / Jconsole / JProfiler 等

综合这些，可以达到对整个系统的有效监控，包含所有的软硬件数据，结合着性能测试工具里 TPS、响应时间等数据，可以更好的横向对比分析出可能存在的性能问题，并逐步优化解决。

性能优化

优化其实是很难说清楚的一个点，也是在性能测试里很需要经验的地方，我们聊几种常见的场景：

Round 1： 响应时间慢

这恐怕是我们遇到最多的场景，直观上看过去就是响应的非常慢。这个时候我们的调优方式就是把整个的 Response Time （RT）通过日志进行不断的分解，例如，我们现在是把一个 RPC （Dubbo）的服务部署在 Tomcat 中，上游用 Nginx 做反向代理。在这样的架构下，一个完整的 RT 就会包括前端 RT、网络传输时间、Nginx RT、Tomcat RT、Dubbo 服务 RT 和数据库 RT。如果存在问题，我们可以请开发配合在各个局部增加日志，观察具体响应时间慢的点，从而进行后续的优化动作。

Round 2： TPS 波动大

一般来说，性能测试的 TPS 应该随着并发量的上升而呈跟随上升趋势直至稳定，但是有时候我们会发现，被测系统的 TPS 非常不稳定，上下波动非常大。排除网络可能造成的影响，在实际测试过程中可能遇到最大的可能就是：**被压测服务器上存在其他运行的服务争抢资源或者垃圾回收的问题。**

其中后者的可能性居多，一般都是出现有频繁的 FGC。这就是需要我们结合着对 JVM 监控的数据来进行分析，修改 JVM 内存参数或代码逻辑，达到优化的目的。

Round 3： 开始加压正常，到某一个点突然开始出现错误，并越来越多

这种情况多数可能是由于服务器、中间件等配置的线程数、超时时间造成的影响，线程数过小，服务端同时可以处理的请求就少，自然到达一定的并发就会等待，最终超时错误。

这类问题我遇到过的更多时候是由于 **数据库的慢 SQL 或者不恰当的代码或数据库锁机制造成的。** 一般优先排查数据库 SQL 问题，通过数据库日志判断 SQL 执行时间，查看是否需要优化；如果 SQL 没有问题，那么就需要与开发同学一起沟通确认代码中是否有不恰当的同步锁等等。

当然，这都是经过不断总结到的经验来分析出来的常见的性能优化场景。那么如果你碰到一个性能问题，却不知道问题出在哪儿，一般情况下，我们排查问题的顺序是从上到下：先查服务器硬件瓶颈（CPU、内存、I/O），再看是否由于配置问题引起，接下来查看数据库、SQL 是否存在瓶颈，最后是应用代码逻辑、JVM。

同时，一定不要忘记另外一个点：前端性能。当你发现后端请求响应时间一切正常，但是通过前端访问仍然存在性能瓶颈的时候，一定不要忘记前端加载过程带来的影响，而针对前端性能的测试使用 YSlow 是比较方便的。

顺便也介绍一下，前端性能问题最常见的一些优化方式：

- 合并 HTTP 请求，减少请求数量
- 资源压缩
- 使用浏览器缓存
- 图片的优化：雪碧图等方式

当然，这里边只是介绍了一些性能监控与调优要关注的东西和一部分要点，并没有详细去跟大家说到底怎么监控、用什么命令、怎么去看，也没有去细节的讲每个部件到底怎么调优，所以，如果想要真正掌握性能测试，还需要真正上手去实验、去探索、“遇事不决找百度 ”，这样才能搞定” 性能 “这座大山。

← 24 比 “码脚本” 更重要的事：性能场景分析
26 站在LoadRunner与Jmeter的十字路口 →

优秀测试工程师的必备思维39讲 / 25 不懂监控与调优，就像白天不懂夜的黑

精选留言 1

欢迎在这里发表留言，作者筛选后可公开显示

SCXR

老师，我近期才学性能测试，看了您的LoadRunner的视频课程，也算入门了。性能测试是在做完功能测试之后才做的，那如果性能测试测出bug之后并且修改了代码，那功能测试是否还需要再做一遍?

1 回复 2019-12-03

风落几番 回复 **SCXR**

看修改什么了，这其实需要你对代码有一定掌握，了解这次修改具体的修改点是什么。其实如果有自动化的话，每次修改都应该回归一下为好

回复 2019-12-10 08:36:46

千学不如一看，千看不如一练