

测试专栏特别放送 | 浅谈全链路压测

2018-11-14 茹炳晟

软件测试52讲

[进入课程 >](#)



讲述：茹炳晟

时长 13:45 大小 6.30M



你好，我是茹炳晟。今天我和你分享的主题是：浅谈全链路压测。

时光飞逝，从专栏上线至今，我已经和你分享了 52 篇文章和 7 篇答疑文章，和你分享了软件测试中的各个主题，希望已经帮你构建了一幅软件测试的知识全景图。

在前面的“性能测试”系列文章中，我以 LoadRunner 为例，和你分享了传统企业在实际开展企业级性能测试的实践。并且在第 32 篇文章[《无实例无真相：基于 LoadRunner 实现企业级服务器端性能测试的实践（上）》](#)中，我和你分享了这么安排的原因，并承诺在专栏结束前，通过一篇“加餐”文章，和你分享开展全链路压测的难点，以及应对方案。

现在，就是我践行承诺的时间了。

我也不太清楚，你现在具备多少全链路压测的知识。所以，我会先和你分享一些全链路压测的理论知识，然后再分享具体的难点以及解决思路，帮你加深理解，希望可以让你听得明白、学得会、用得着。

什么是全链路压测？

全链路压测，是基于真实的生产环境来模拟海量的并发用户请求和数据，对整个业务链路进行压力测试，试图找到所有潜在性能瓶颈点并持续优化的实践。

目前，一线互联网公司的大型电商平台都会不定期地开展全链路压测，比如淘宝、京东、饿了么和美团这些企业，基本都已经有了自己的全链路压测方案和平台。

其中，最为典型的要数淘宝的双 11 活动了。每年到了 11 月 11 日的零点，淘宝的整个系统都会面临极大的流量冲击，如果事先没有经过充分的测试和容量预估，很可能在流量爆发时瘫痪。

记得在早些年的淘宝双 11 大促中，就出现了不同程度的网站故障，严重影响了用户体验，所以从 2013 年开始，淘宝开始实施全面的全链路压测。由于在真正的双 11 到来前，淘宝内部已经模拟了比双 11 流量还要高的负载，并且逐个解决了已经发现的问题，因此真正双 11 到来的时候，就不会出现严重的问题了。

因此，为了防止此类事故，淘宝会在每在双 11 之前，就对系统的稳定性以及负载承受能力进行必要的测试和评估。

当然，**全链路压测的应用场景，不仅仅包括验证系统在峰值期间的稳定性，还会包含新系统上线后的性能瓶颈定位以及站点容量的精准规划。**

比如，由于某些业务模块的操作负载会集中到几个最核心的组件上，那么通过全链路压测的模拟，我们就能快速识别出哪些模块的负载过大，哪些模块的负载偏小。这样我们在对系统进行扩容时，就可以把资源更多地给到那些承受大负载的模块，而那些承受负载偏小的模块就可以进行适当的收缩来让出更多的可用资源。这，就是精准的容量规划。

单系统的独立压测

早先的时候，压测并不是针对业务的全链路来开展的，而是采用了“各个击破”的原则，即对生产环境中的单机或者单系统进行独立的压测。这时，压测主要是通过模拟单一系统的海

量并发请求来实现的。而模拟海量请求主要有两种实现方式：

一种是，根据设计的压力来直接模拟大量的并发调用；

另一种是，先获取线上真实的流量请求，然后经过数据清洗后，再回放模拟大量的并发调用。

不管采用的是哪种方式，都会涉及流量模拟、数据准备、数据隔离等操作。除此之外，单系统的独立压测局限性也非常明显。

这里，我把单系统独立压测的局限性，归纳为了以下几点：

单系统压测的时候，会假设其依赖的所有系统能力都是无限的，而实际情况一定不是这样，这就造成了单系统压测的数据普遍比较乐观的情况；

在大压力环境下，各系统间的相互调用会成为系统瓶颈，但这在单系统压测的时候根本无法体现；

大压力环境下，各系统还会出现抢占系统资源（比如网络带宽、文件句柄）的情况，这种资源抢占必然会引入性能问题，但是这类问题在单系统压测过程中也无法体现出来；

由于是单系统测试，所以通常都只会先选择最核心的系统来测试，这就意味着其他的非核心系统会被忽略，而在实际项目中，这些非核心系统也很有可能会造成性能瓶颈。

因此，为了解决单系统独立压测的一系列问题，业界就衍生出了全链路压测。全链路压测会把整个系统看作一个整体，然后在真实的生产环境上尽可能真实地去模拟业务的海量并发操作，以此来衡量系统的实际承载能力，或者找出系统可能的瓶颈点并给出相应的解决方案。

目前来看，全链路压测需要解决的技术难点有很多，这里我会和你讨论其中最重要的四个点：

1. 海量并发请求的发起；
2. 全链路压测流量的隔离；
3. 实际业务负载的模拟；
4. 测试完成后的数据清理。

海量并发请求的发起

由于全链路压测需要发起的海量并发，通常会超过每秒 1000 万次以上请求的压力量级，所以传统的性能测试工具 LoadRunner 已经很难满足要求了，原因有二：

一来，LoadRunner 按并发用户数收费，这就使得采用 LoadRunner 进行互联网的全链路压测的费用会异常高；

二来，LoadRunner 本身也很难支持千万级乃至亿级的海量并发。

所以，业界基本都是采用免费的 JMeter 来完成全链路压测，这也是 JMeter 近几年被互联网企业广泛使用的原因。

但是，即便有了 JMeter，我们在开展全链路压测时，也会有很多问题需要解决。其中，最主要的问题包括以下三个：

1. **虽然采用了分布式的 JMeter 方案，并发数量也会存在上限**，比如面对亿级的海量并发时，主要原因是分布式的 JMeter 方案中，Master 节点会成为整个压测发起的瓶颈。为了解决这个难题，很多公司并不会直接采用分布式 JMeter 架构来完成海量并发，而是会使用 Jenkins Job 单独调用 JMeter 节点来控制 and 发起测试压力。这样就避免了 Master 节点引发的瓶颈问题。而且，由于各个 JMeter 是完全独立的，所以只要 Jenkins Job 足够多，并且网络带宽不会成为瓶颈的情况下，就能发起足够大的并发。
2. **测试脚本、测试数据和测试结果在分布式 JMeter 环境中的分发难题**。如果直接采用分布式的 JMeter 方案，测试脚本需要通过 JMeter 的 Master 节点来分发，测试数据文件则要用户自行上传至每套虚拟机，同时测试结果还要通过 JMeter 的 Slave 节点回传至 Master 节点。
所以，更好的做法是基于 JMeter 来搭建一个压测框架，诸如脚本分发、数据分发以及结果回传等工作，都由压测框架完成。这也是目前绝大多数大型互联网企业的做法。比如，饿了么就采用这种方式搭建了压测平台，并且取得了很好的效果。
3. **流量发起的地域要求**。全链路压测流量的发起很多时候是有地理位置要求的，比如 30% 的压力负载来自上海、30% 的压力负载来自北京等，这就要求我们在多个城市的数据中心都搭建 JMeter Slave，以便可以发起来自多个地域的组合流量。

全链路压测流量和数据的隔离

因为全链路压测是在实际的生产环境中执行的，所以测试产生的数据与真实的用户数据必须进行有效隔离，以防止压测的流量和数据污染、干扰生产环境的情况。比如，不能将压测数据记录到统计分析报表里；再比如，压测完成后可以方便地清洗掉压测产生的数据。

为了达到这个目的，**我们就需要对压测流量进行特殊的数据标记，以区别于真实的流量和数据**。这就要求各个链路上的系统，都能传递和处理这种特殊的数据标记，同时写入数据库中的数据也必须带有这种类型的标记以便区分数据，或者直接采用专门的影子数据库来存储压测的数据。

可以看出，为了实现压测产生的和真实的流量和数据隔离，我们就需要对各个业务模块和中间件进行特殊的改造和扩展。而这个工作量相当大，而且牵涉的范围也非常广，也就进一步增加了实施全链路压测的难度。

而且通常来讲，首次全链路压测的准备周期会需要半年以上的时间，这其中最大的工作量在于对现有业务系统和中间件的改造，来实现压测流量和数据的隔离。所以，在实际的工程项目中，如果全链路压测不是由高层领导直接牵头推动的话，很难推进。

另外，在对各个业务模块和中间件添加特殊标记的改造过程中，我们会尽可能少地改动业务模块，而是更倾向于通过中间件来尽可能多地完成特殊数据标记的处理和传递。

实际业务负载的模拟

一直以来，如何尽可能准确地模拟业务系统的负载，都是设计全链路压测时的难题。这里的难点主要体现在两个方面：首先，要估算负载的总体量级；其次，需要详细了解总负载中各个操作的占比情况以及执行频次。

业界通常采用的策略是，采用已有的历史负载作为基准数据，然后在此基础上进行适当调整。具体到执行层面，通常的做法是，录制已有的实际用户负载，然后在此基础上做以下两部分修改：

录制数据的清洗，将录制得到的真实数据统一替换成为压测准备的数据，比如，需要将录制得到的真实用户替换成专门为压测准备的测试用户等等；

基于用户模型的估算，在全链路压测过程中，按比例放大录制脚本的负载。

最后，再用这个负载来模拟全链路压测的负载。

真实交易和支付的撤销以及数据清理

由于全链路压测是在真实的生产环境中进行的，那么完成的所有交易以及相关的支付都是真实有效的，所以我们就需要在测试结束后，将这些交易撤销。

因为，我们已经对这些交易的流量和数据进行了特定标记，所以我们可以比较方便地筛选出需要撤销的交易，然后通过自动化脚本的方式来完成批量的数据清理工作。

除了上面的四大问题以外，全链路压测还需要考虑测试执行过程中的性能监控、高强度压测负载下的测试熔断机制、全链路压测执行期间对原有系统正常负载的影响、全链路压测数据对外的不可见等等。

所以说，全链路压测的技术含量很高，而且需要多方共同配合才有可能顺利完成。所以，今天这篇文章的目的，意在抛砖引玉。希望你可以借由这篇文章，先对全链路压测的难点以及对应的解决思路有个全局的认识。而如果你想要更好地了解并掌握全链路压测，最好的方式还是要在实际项目中多加历练。

另外，你还可以参考一些网上的优秀资源，我在这里列出了两条供你参考：

https://mp.weixin.qq.com/s?biz=MzIzMzk2NDQyMw==&mid=2247486703&idx=1&sn=0c448c40469e6a8f32476a7c7fbab6cd&source=41#wechat_redirect
<https://www.cnblogs.com/imyalost/p/8439910.html>

总结

今天这篇文章，我和你分享了全链路压测的基本知识，以及在开展全链路压测的难点、对应的解决思路。现在，我再和你一起回顾下。

全链路压测，是基于真实的生产环境来模拟海量并发用户请求和数据，对整个业务链路进行压力测试，试图找到所有潜在性能瓶颈点并持续优化的实践。它的应用领域不仅仅包含验证系统在峰值期间的稳定性，还会包含新系统上线后的性能瓶颈定位以及站点容量的精准规划。

了解了全链路的基本概念，以及适用场景后，我和你分享了全链路压测中最关键的四个技术难点，即：海量并发请求的发起、全链路压测流量和数据的隔离、实际业务负载的模拟，以及测试完成后的数据清理。

海量并发请求的发起主要借助于 JMeter，并且通过 Jenkins Job 来实现海量并发的调度控制；

全链路压测流量和数据的隔离主要借助含有特定标记的流量和数据来实现，同时需要对业务模块以及中间件进行必要的改造，数据库这边还会使用影子数据库；

实际业务负载的模拟，主要是采用基于历史流量修改后的回放来实现；

全链路压测完成后的数据清洗，则是借助自动化的手段来批量完成。

思考题

你能想到全链路压测中还有哪些技术上的难点吗？

感谢你的收听，欢迎你给我留言一起讨论。

 极客时间

软件测试52讲

从小工到专家的实战心法

茹炳晟

eBay中国研发中心
测试基础架构技术主管



新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 测试专栏特别放送 | 答疑解惑第七期

下一篇 结束语 | 不是结束，而是开始

精选留言 (9)

 写留言



mark

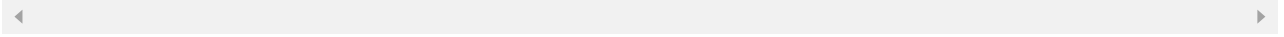
2019-01-08



你们那边的方案是哪种？可以分享一下么？

展开 ▾

作者回复: 我们是使用jmeter的



mark

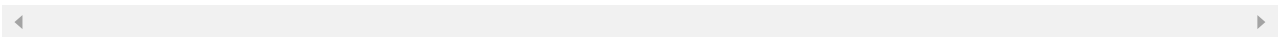
2018-12-19



影子数据库是指同样结构的数据库，应用把库切过来？

展开 ▾

作者回复: 可以这样的理解，具体实现上还有很多不同的技术方案



派森

2018-12-14



希望余生能有幸经历一次从0到1的全链路压测

展开 ▾



小老鼠

2018-12-01



全链条压测的测试场景如何设计？

展开 ▾



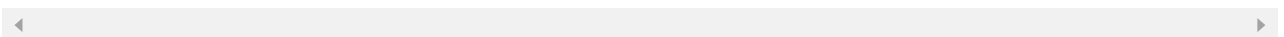
mindy

2018-11-28



在真实业务高流量访问下，全链路压测会对生产环境下的用户的访问造成冲击不？还是选择压测的时间段会被隔离？

作者回复: 压测一般是在实际生产环境上直接进行，期间会严密监控系统指标，以免真的压垮系统，一般选择低峰时段来进行全链路压测





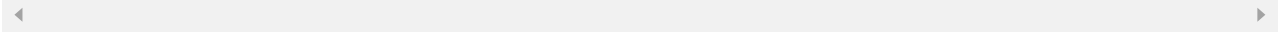
Maxwell

2018-11-23



请问中间件改造是怎么改造的？能否有案例实践下呢？还有数据隔离具体咋实现的？

作者回复: 这个不是一两句可以说清楚的，中间件改造主要是去支持特定的流量隔离标志位，数据是采用影子数据库实现的



郭小菜

2018-11-16



老师说的中间件改造和特殊标记这个是可以说的具体些吗？可以举个例子说明吗？

作者回复: 主要目的是能够区分压测流量和真实流量



kaiserin

2018-11-14



使用脚本来启动多个jmeter来压测和使用jenkins job是否是一个原理？

展开 ∨



£ Pitt•Xu...

2018-11-14



坚持不懈学习

展开 ∨