全链路压测经验

monkey01 关注
0.6 2018.01.16 09:58 字数 4841 阅读 4904 评论 2 喜欢 20

前言

随着业务的快速发展我们日常遇到的系统性能压力问题也逐渐出现，甚至在部分场合会遇到一些突发的营销活动，会导致系统性能突然暴涨，可能导致我们系统的瘫痪。最近几年随着电商的各种促销活动，有一个词也渐渐进入我们眼帘－－“全链路压测”。全链路压测被众多互联网公司的程序员定义为核武器，传统性能测试更多的是以事务为核心，更多的是由单个或者多个事务构成业务场景进行压测。那全链路压测到底是什么？一般指完全引入相关联的系统 真实模拟线上硬件环境，更多的是以请求为核心，完全模拟真实请求流量，通过引流等方式进行场景的模拟进行压测，更多的适用于业务链路较长的交易。

笔者以前只是一直听说全链路压测，但是并没有真正经历过，对全链路压测的理解也不是很全面，前年在互联网电商公司双11的时候参加过一次全链路的压测，当时全公司第一次做大范围的全链路压测，整个架构部也是第一次牵头来完成了整个全链路压测，经过大家1个月的努力，最后在活动期间完全扛住了压力，并且还有性能过剩。当时做完后因为忙的太累，没有进行过总结，最近新的公司正好在压测，借此也总结下全链路压测。

1. 为什么需要全链路压测

我们在整个业务流程中，最大的困难在于评估从用户登录到完成全部交易的整个链条中，核心页面和交易关键交易的实际承载能力。如果得到了各个系统的实际承载能力，就可以在路由网关进行相关交易限流控制，来防止在大并发来了以后系统出现宕机，我们都知道一旦系统宕机就会导致灾难性的后果，而且就算运维短时间重启了起来恢复了运行，但是可能过了一会儿过程系统承载量又出现宕机，早期阿里在双十一的时候就发生过这样的问题，系统在0点，出现大面积瘫痪，重启后又瘫痪，为什么会出现这个问题，就是因为大家对整个全交易链条上的各个环节的系统承压能力不清楚，所以在出现了全链路压测，一方面能够让各个产品知道自己的承压极限在哪？有的同学会问了通过单系统压测不是也可以知道各个系统的承压能力吗？但是实际情况不可能那么简单，那么顺利，在活动开始的瞬间，从CDN、网关接入、前端、缓存、中间件、后端服务、数据库整个交易链路都会面临巨大的访问压力，这个时候系统服务除了受自生的影响，还依赖于其他关联系统的情况，并且影响会一直蔓延，只要有一个节点出现故障，那么故障在上下游系统经过层层累加后会造成的影响谁都说不清楚，所以最好的办法就是模拟完全的真实情况来做到提前心里有数。验证的最好办法就是让事件提前发生，通过全链路压测就可以提早发现问题。

另一方面也可以让各个系统能够有个明确的优化目标并找出性能瓶颈，同时对于一些特殊环节可以通过临时增加公有云的方式来提高整体的性能，一旦通过全链路压测，了解了瓶颈所在就可以坦然的去按照压测指标去申请公有云资源，活动结束后再归还资源，这样做到成本最低化。

2. 全链路压测常常遇到的问题

如何开展全链路压测？在说这个问题前，我们先考虑下，全链路压测有哪些问题比较难解决。

1）涉及的系统太多，牵扯的开发人员太多；

在压测过程中，做一个全链路的压测一般会涉及到大量的系统，在整个压测过程中，光各个产品的人员协调就是一个比较大的工程，牵扯到太多的产品经理和开发人员，如果公司对全链路压测早期没有足够的重视，那么这个压测工作是非常难开展的。

2）模拟的测试数据和访问流量不真实；

在压测过程中经常会遇到压测后得到的数据不准确的问题，这就使得压测出的数据参考性不强，为什么会产生这样的问题？主要就是因为压测的环境可能和生成环境存在误差、参数存在不一样的地方、测试数据存在不一样的地方这些因素综合起来导致测试结果的不可信。

3）压测生产数据未隔离，影响生产环境；

在全链路压测过程中，压测数据可能会影响到生产环境的真实数据，举个例子，电商系统在生产环境进行全链路压测的时候可能会有很多压测模拟用户去下单，如果不做处理，直接下单的话会导致系统一下子会产生很多废订单，从而影响到库存和生产订单数据，影响到日常的正常运营。

3. 如何开展全链路压测

其实进行全链路压测对于整个公司技术要求还是很高的，如果没有一定能力的公司最好不要贸然尝试全链路压测，因为如果没做好可能会把生产环境搞宕，所以对于没有一定科技能力的公司还是尽量不要贸然追潮流实施全链路压测。对于科技能力不强的公司如果也想达到全链路压测那该怎么办？其实办法也很简单，可以在压测环境进行模拟，做一个比例模拟，模拟1%－2%的访问量在测试环境进行压测，得到数据后乘以对应的倍数来得到总的压测指标，这里的比例当然不是简单的倍数相乘，需要各个系统得到系统线性扩展和单机压测指标的一个线性值，因为一般来说的线性扩展都不可能是100%的，一定会有一定扩展后的损失。

1）分析需压测业务场景涉及系统

在压测前我们一定要首先分析清楚需要压测的业务场景，只有分析清楚了业务场景才能梳理出来涉及的相关系统，分析清楚后也可以更快的找到性能瓶颈进行系统优化。这个工作一般是由架构师来梳理并确认涉及的相关系统，梳理清楚后就可以反馈给总压测负责人进行人员和资源的协调了。

2）协调各个压测系统资源

在全链路压测过程中，最难的工作其实不是系统优化、压测环境搭建等技术工作，最难的是压测资源的协调工作。这里的资源不单单指压测硬件、公有云等资源，还包括了人力资源，在整个压测过程中，需要各个相关产品的产品经理和技术开发人人员参与进去，这个工作可能不是架构师或者一个牵头产品的负责人能够协调的动的，必须要有一个自上而下的推力去推动，需要一个有一定级别的领导做背书，我们当时是分管科技的老总召

集了所有的产品经理和各个产品技术开发负责人开了一个全链路压测的方案讨论会，这个会一方面说明了这个事情的重要性，让各个产品都当场立下军令状，并且安排出支持的人员，同时也授权给压测负责人。这个搞定以后压测负责人后续就可以光明正大的协调各个产品的配合人员了。

3）压测环境

压测环境也是个比较头疼的问题，很多系统可能压根就没有压测环境，所以全链路压测有个和传统压测比较大的区别就是，全链路压测是在生产环境，这种做法其实是存在一定风险的，一方面是系统风险，一方面是业务数据风险。对于全链路压测系统风险一定是首要考虑的问题，不能因为压测把系统搞宕机影响到日常生产环境的正常运营，我们当时做的电商系统还好，不涉及相关的监管。但是在银行业这样做还是有很大的风险的，一旦生产系统出现关键交易系统的宕机可能导致一些金融事故，会对金融市场造成恐慌，而且会被银监会通报，所以银行的压测还是不要进行全链路压测，不过可以在测试环境尽量仿真的模拟全链路压测。

前年在电商环境上做全链路压测直接在生产上进行压测，用生产环境最大的好处就是环境的真实性，通过生产环境能够最高程度的还原生产环境不用额外准备压测环境。但是使用生产环境进行压测需要考虑将请求和访问、业务数据处理都进行隔离，防止影响到生产环境，具体如何实施，涉及到比较多的细节这里就不详细描述了。总的思路就是在发起请求的时候通过请求报文头中的压测标示来进行区分处理，将压测的流量都分流到指定的应用服务器和指定的存储进行数据保存和处理。

进行全链路压测的时候，为了防止系统崩溃，可以选择在凌晨用户量最小的时候进行，这样就算发生故障也可以将影响降到最低。

4）压测数据

环境准备好了，可能就需要考虑造压测数据了，压测数据准备有两方面数据需要准备，一方面是压测请求数据的准备，需要模拟请求数据，请求数据最好的办法就是采用生产的真实数据，我们当时的做法是直接录制3-5天的生产访问请求流量，只需要对录制的请求进行数据清洗就可以了，将某个用户的请求替换成一个压测虚拟用户的请求，请求的商品也替换成压测虚拟商品，这个数据漂白替换的工作是比较复杂的，对于做数据漂白的的同学对系统的接口非常了解，否则很可能造成业务数据的混乱，造成大量的业务报表和系统数据的脏数据；另一方面是测试数据的准备，我们当时准备了压测虚拟商品的数据、虚拟商品库存数据、虚拟供货商、虚拟用户。

5）压测数据隔离

因为是在生产环境做的压测，压测数据需要与正常的业务数据隔离开，我们当时的方案是对于压测的这些脏数据都做了特定标示，对于虚拟用户、虚拟商品、虚拟订单、虚拟库存都是有特殊标示的，这样这类数据在统计的时候都不会进行统计，在压测后也会对这些数据进行清理，防止污染正常业务数据。

6）压测数据实时监控

在压测过程中，为了能发现性能问题，我们需要对压测过程中各个系统的cpu、内存、磁盘io都进行系统层面的监控，同时也需要对各个业务节点的耗时进行监控，一方面从业务层面去监控压测事务性能，另一方面从系统层面监控，这样我们可以先从业务层面找到性能瓶颈，再单独分析各个系统的系统层面的瓶颈，最终找到优化方案。

我们当时公司内部有个实时监控平台，这个平台是基于大众点评开源的cat实现的多平台监控系统，能够实时监控各个系统的实时交易运行情况，这样能够在第一时间发现遇到大流量的情况后，性能瓶颈在哪？然后进行针对性的优化。

4. 全链路压测优化思路

性能优化的核心在我看来其实就是一个充分利用系统资源并平衡IO的过程。这句话怎么理解，首先在保证代码没有问题的情况下，充分利用系统的cpu、内存、磁盘资源，一般来说在保证cpu、内存都消耗到80%以上基本上就达到了性能峰值了，但是我们在压测过程中常常遇到的问题是cpu、内存消耗都不高，而是卡在了IO上，IO包括了磁盘IO、数据库IO、网络IO，我们需要根据监控的数据从这3方面去找到瓶颈，并去解决IO的问题。一般来说造成这种情况一般都是因为IO聚集导致了阻塞，可以考虑采用缓存、异步的方式去解决，对于一些关键交易的事务的完整性可以考虑采用先缓存最后通过缓存同步数据库的方式来保证最终一致性。

全链路压测一般可以从3个层面去进行优化：

1）优化单个系统性能

就算不进行全链路压测，单个系统的性能优化也是要考虑的问题，对单个系统的优化，其实方法有很多，但是万变不离其宗，就是在压测过程中监控系统各项指标，从中挑出慢交易，针对慢交易进行优化，对于联机系统大部分都是因为各种IO问题导致性能上不去。可以根据各种介质IO访问的性能来优化（内存缓存>文件>数据库>网络），基本上通过缓存和异步处理这两颗银弹就可以解决80%的性能问题。

当链路上的单个系统性能提升了，整体的全链路性能自然就提升了。

2）优化关联路径

但是在优化的过程中，我们常常发现绝大部分系统性能都很高，但是总的TPS还是很低，这就需要去根据监控了解下目前整个链路上的性能瓶颈到底在哪？通过全链路监控可以发现整个业务流程在哪个节点耗时最长，那么这个耗时最长的节点就是我们需要优化的地方，只要这些关键路径的性能提升上来以后整体的性能就上来了。关键节点的优化方式和单系统优化思路一致。

3）优化业务流程

很多开发人员都会将优化思路集中在技术层面，但是很多时候从业务流程上进行优化效果可能更好，而且提升的效果会非常明显。业务层面的优化主要是从分散IO的角度去考虑，将实际业务场景中的用户请求进行分散，例如常见的大秒系统、验证码系统、游戏工具等都是为了进行业务层面的IO分散来保证。这类业务流程的优化

首先要梳理清楚整个业务流程，包括所有的细节。然后针对每个环节在保证用户体验的情况下分散用户请求，这样可以最大限度的保证体验。

总结

整个压测优化过程就是一个不断优化不断改进的过程，通过长期的循序渐进的改进不断发现问题，优化系统，才能让系统的稳定性和性能都得到质的提升。整个压测优化的思路其实和高并发架构设计的思路是一致的，接下来也会写一些关于高并发架构的文章，本篇的全链路压测只是给大家做个入门介绍，其中涉及到的问题远远不止文中提到的这些，而且问题的解决办法也远远不是说的那么简单，造虚拟用户、虚拟商品并不是随便造的，数据隔离也不是简单加个前缀什么的就可以的，也是有很多讲究的，因为全链路压测涉及到的内容太多而且还涉及到各家公司的组织架构，所以这里就不展开了，只是给大家一个思路，按照这个思路结合自己公司的情况去实施，慢慢摸索总结出一套适合自己产品的全链路压测。