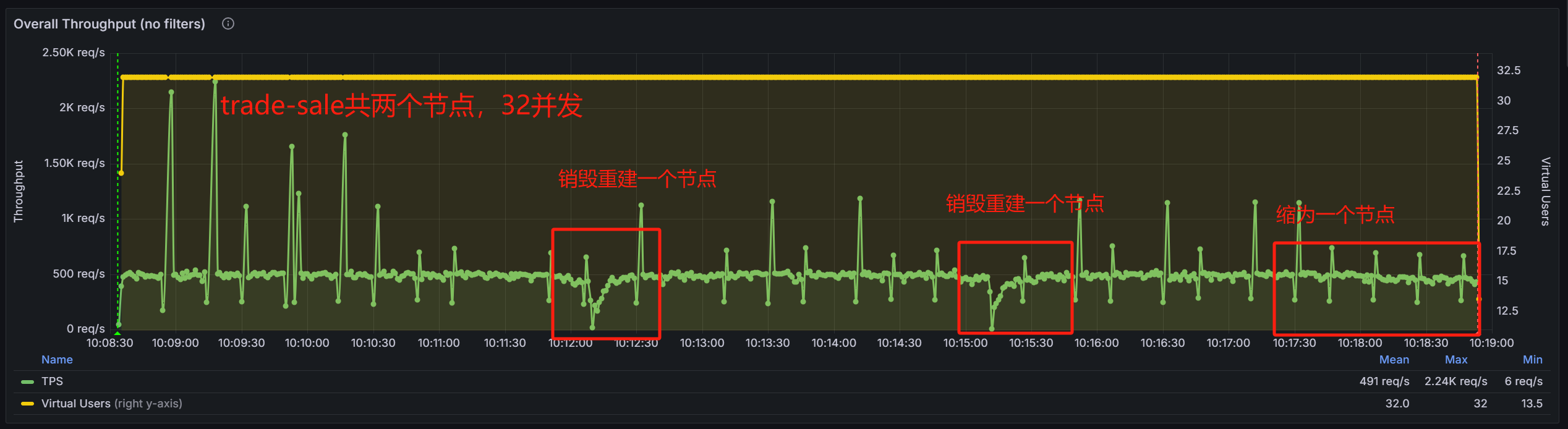
实例销毁重建时 tps 抖动的初步分析

# 初始现象

客户使用 sdk 版本 1.40.2-SpringCloud2021-RELEASE + dubbo3.2.10

客户在压测过程中，对核心交易服务（共2个节点，dubbo 接口）的其中一个节点进行销毁重建 或 缩容，并观察 tps 情况。现象为，销毁重建时，tps 会有个明显的波动（大概下降 90%），缩容时 tps 没有明显抖动



# 分析

从实例下线角度，销毁重建和节点缩容是一样的。所以 tps 下降的原因可能是在节点上线阶段。即服务注册后，刚开始的请求还涉及一些初始化操作（例如 db、redis 连接的初始化）。响应时间比正常要慢些。但默认流量分配策略是轮训，造成总体 tps 的下降。

客户侧进一步验证，也证明了抖动是出现在新实例上线时。14点02分这个抖动，是从2节点扩容到3 节点造成的。此时平均响应时长，从约 100ms 大幅增加到约 1.4s（这个是平均后的结果，新节点的响应耗时可能超 1.4s）



客户侧更多节点时的验证情况：

两个节点时，销毁一个重建，tps由500降至48，降幅达到约90%；

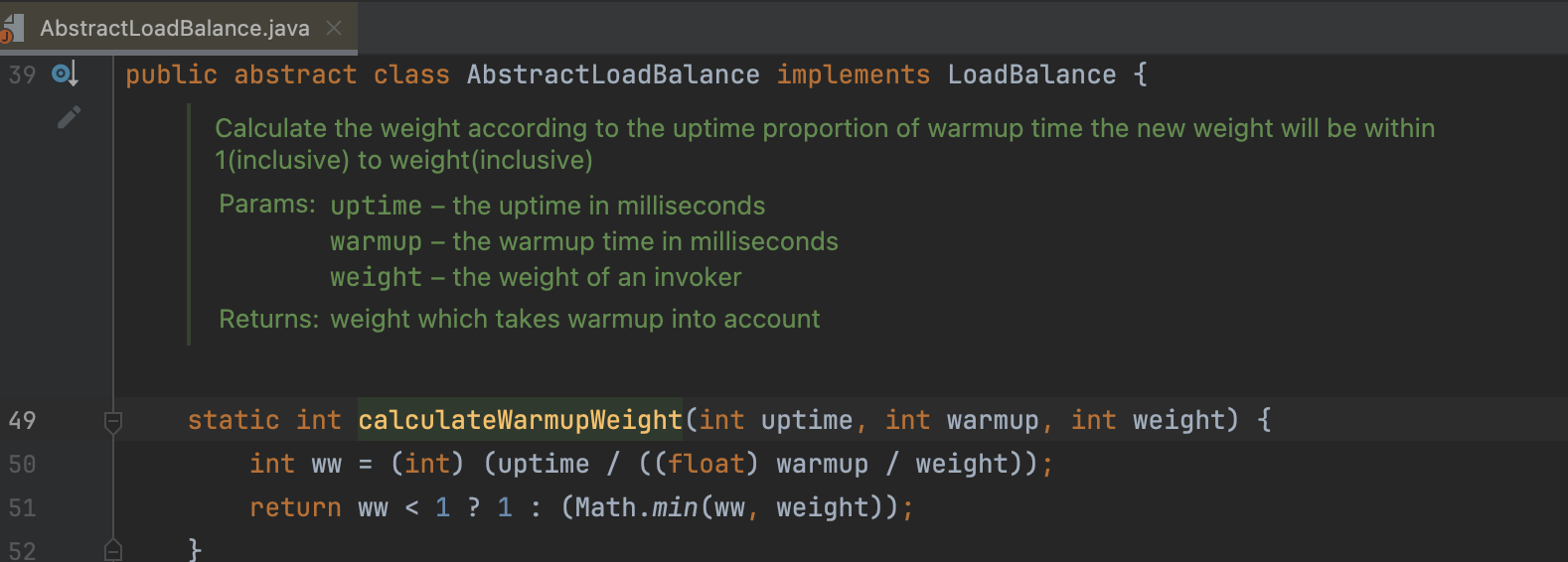
四个节点时，销毁一个重建，tps由500降至102，降幅达到约80%；

八个节点时，销毁一个重建，tps由500降至147，降幅达到约70%；

# 产品适配

使用 dubbo 的小流量预热功能。即新上线的节点，权重较低，后面逐步提高。这样可以让新上线节点刚上线时有少量流量用于初始化，减少影响。预热权重计算参考 org.apache.dubbo.rpc.cluster.loadbalance.AbstractLoadBalance#calculateWarmupWeight

预热节点的权重计算规则如下，uptime 为当前时间戳减去节点启动时间时间戳（单位ms），warmup 为预热时长，默认为 600000（单位ms，即10min），weight 为初始权重（默认为100）



dubbo 框架有提供小流量预热功能，但当前 tsf sdk 未适配，需额外适配（预计4月15日提供临时版本用于验证）。

# tps 波动过大的可能原因

该问题和业务请求关系密切，这里仅做通用分析，以实际情况为准。并且上面场景是在压测场景下的，通常场景可能影响会相对小些

默认策略是轮训。但如果下游其中一个节点响应耗时大幅增加，影响请求的比例不仅仅是 异常节点数/总节点数

极端场景：如果一个节点的请求一直卡住（非常慢）。会快速消耗完上游的资源。

假如上游线程池资源是 200 个，下游两个节点，一个新节点（会卡住请求）和一个旧节点（正常）

第一轮 100请求， 50个新节点，50 旧节点 。造成上游50个线程卡住

第二轮 100请求， 50个新节点，50 旧节点 。造成上游100个线程卡住

第三轮 100请求， 50个新节点，50 旧节点 。造成上游150个线程卡住

第四轮 100请求， 50个新节点，50 旧节点 。造成上游200个线程卡住

这个场景下，即使是一个节点无响应，经过 400 个请求后，就把上游 200 个线程池全部占用。此时再有新的请求过滤，也会因为没有线程资源，一直处于等待状态。

新实例上线时，该实例的响应时长只是相对长一些，远远没有到无响应的情况，所以只是稍等卡一下，造成短暂的 tps 下降。

# 建议解决方案

1. **dubbo场景：**使用 dubbo 的小流量预热功能。即新上线的节点，权重较低，后面逐步提高。这样可以让新上线节点刚上线时有少量流量用于初始化，减少影响。当前 tsf sdk 未适配，需额外适配，腾讯**预计4月15日提供临时版本进行验证适配。**
2. **spring cloud场景：**标准的spring cloud熔断器暂时是不支持预热模式和预热排队模式的功能，TSF 产品也暂时不支持预热模式，因此需要业务系统侧结合实际场景来解决此问题，比如负载模式不要采用轮询，随机的模式，建议采用最小响应时间加权等负载模式，或者业务系统进行二次开发改造，或者采用 Sentinel 高可用流量管理框架的预热模式和预热排队模式的功能实现。

# 附件

[服务销毁及新增过程tps影响.docx](https://drive.weixin.qq.com/s?k=AJEAIQdfAAoNJmT32zAXQAAQapAMk)（故障说明）