

月活 12.8 亿的微信，海量请求下是如何防止崩溃的？

dbaplus社群 2022-06-29 07:15 发表于广东

以下文章来源于腾讯技术工程，作者腾讯程序员



腾讯技术工程

腾讯技术官方号。腾讯技术创新、前沿领域发布解读平台。



dbaplus社群

围绕Database、BigData、AIOps的企业级专业社群。资深大咖、技术干货，每天...
976篇原创内容

公众号

一、背景

最近在研究过载保护，微信是一个国民级的应用，月活用户过 10 亿，而且经常过年过节消息量暴增，服务很容易出现过载，但微信的服务一直比较稳定，他们是怎么做的呢？

本文以微信 2018 年发表于 Socc 会议上的文章，《Overload Control for Scaling Wechat Microservices》为基础，介绍了微信大规模微服务的过载保护策略，其中很多方法很有借鉴意义。

下面是对这篇文章做的一些解读。

二、过载保护基本概念

1、什么是服务过载？

服务过载就是服务的请求量超过服务所能承受的最大值，从而导致服务器负载过高，响应延迟加大，用户侧表现就是无法加载或者加载缓慢，这会引起用户进一步的重试，服务一直在处理过去的无效请求，导致有效请求跌 0，甚至导致整个系统产生雪崩。

2、为什么会发生服务过载？

互联网天生就会有突发流量，秒杀，抢购，突发大事件，节日，甚至恶意攻击等，都会造成服务承受平时数倍的压力，微博经常出现某明星官宣结婚或者离婚导致服务器崩溃的场景，这就是服务过载。


3、过载保护的好处

主要是为了提升用户体验，保障服务质量，在发生突发流量时仍然能够提供一部分服务能力，而不是整个系统瘫痪，系统瘫痪就意味着用户流失，口碑变差，夫妻吵架，甚至威胁生命安全（假如腾讯文档崩溃，这个文档正好用于救灾）。

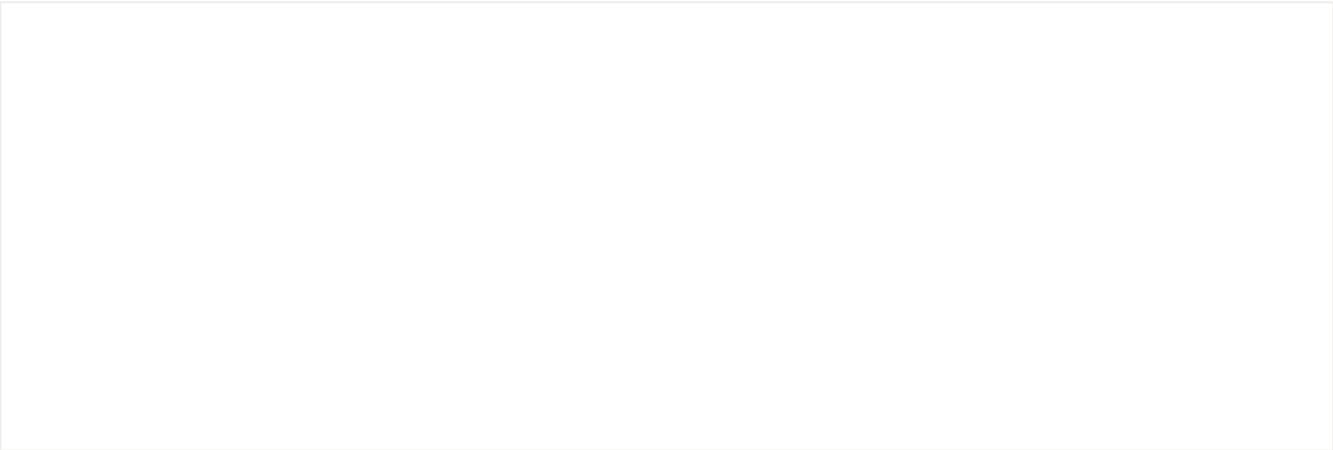
三、微信中的过载场景

微信采用的是微服务，说是微服务，其实我理解就是采用统一的 RPC 框架搭建的一个个独立的服务，服务之间互相调用，实现各种各样的功能，这也是现代服务的基本架构。毕竟谁也不想看到我朋友圈崩了，导致我聊天也不行了。

微信这边的服务是分三层，接入服务，逻辑服务，基础服务，大多数服务属于逻辑服务，接入服务比如登录，发消息，支付服务，每日请求量在 10 亿-100 亿之间，入口协议触发对逻辑服务和基础服务更多的请求，核心服务每秒要处理上亿次的请求。



在大规模微服务场景下，过载会变得比较复杂，如果是单体服务，一个事件只用一个请求，但微服务下，一个事件可能要请求很多的服务，任何一个服务过载失败，就会造成其他的请求都是无效的。如下图所示。



比如在一个转账服务下，需要查询分别两者的卡号，再查询 A 时成功了，但查询 B 失败，对于查卡号这个事件就算失败了，比如查询成功率只有 50%，那对于查询两者卡号这个成功率只有 $50\% * 50\% = 25\%$ 了，一个事件调用的服务次数越多，那成功率就会越低。

四、如何判断过载

通常判断过载可以使用吞吐量，延迟，CPU 使用率，丢包率，待处理请求数，请求处理事件等等。微信使用在请求在队列中的平均等待时间作为判断标准，就是从请求到达，到开始处理的时间。

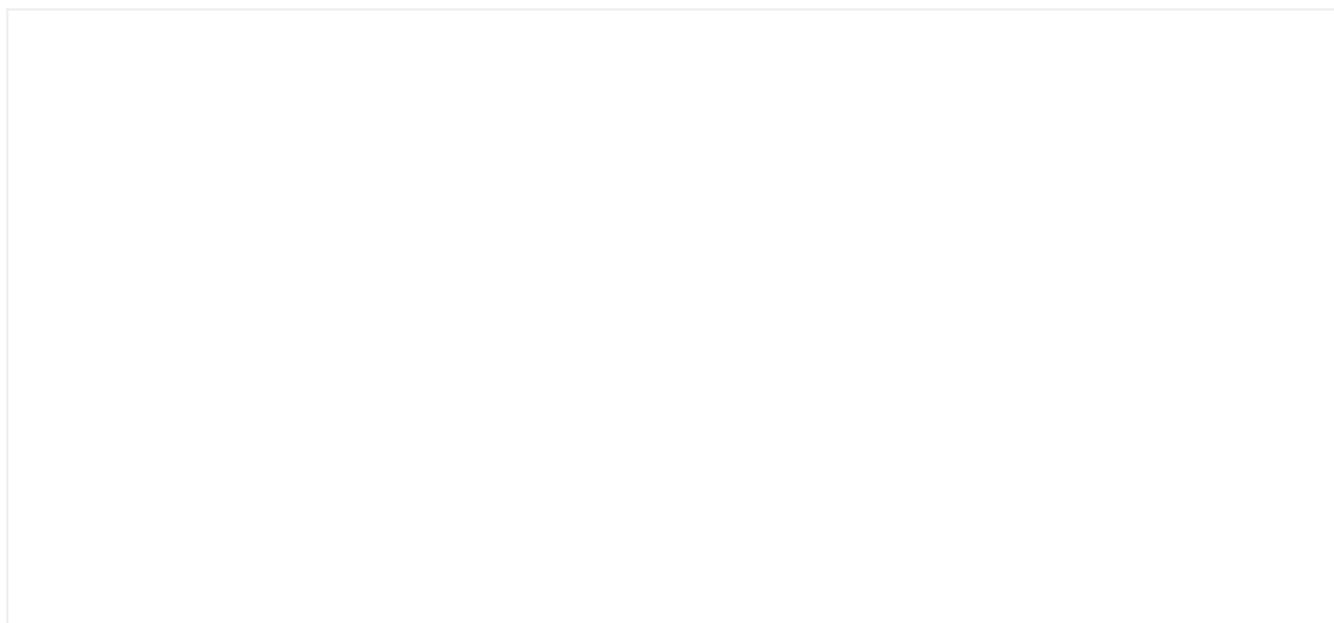
为啥不使用响应时间？因为响应时间是跟服务相关的，很多微服务是链式调用，响应时间是不可控的，也是无法标准化的，很难作为一个统一的判断依据。

那为什么不使用 CPU 负载作为判断标准呢，因为 CPU 负载高不代表服务过载，因为一个服务请求处理及时，CPU 处于高位反而是比较良好的表现。实际上 CPU 负载高，监控服务是会告警出来，但是并不会直接进入过载处理流程。

腾讯微服务默认的超时时间是 500ms，通过计算每秒或每 2000 个请求的平均等待时间是否超过 20ms，判断是否过载，这个 20ms 是根据微信后台 5 年摸索出来的阈值。

采用平均等待时间还有一个好处，是这个是独立于服务的，可以应用于任何场景，而不用关联于业务，可以直接在框架上进行改造。

当平均等待时间大于 20ms 时，以一定的降速因子过滤调部分请求，如果判断平均等待时间小于 20ms，则以一定的速率提升通过率，一般采用快降慢升的策略，防止大的服务波动，整个策略相当于一个负反馈电路。



五、过载保护策略

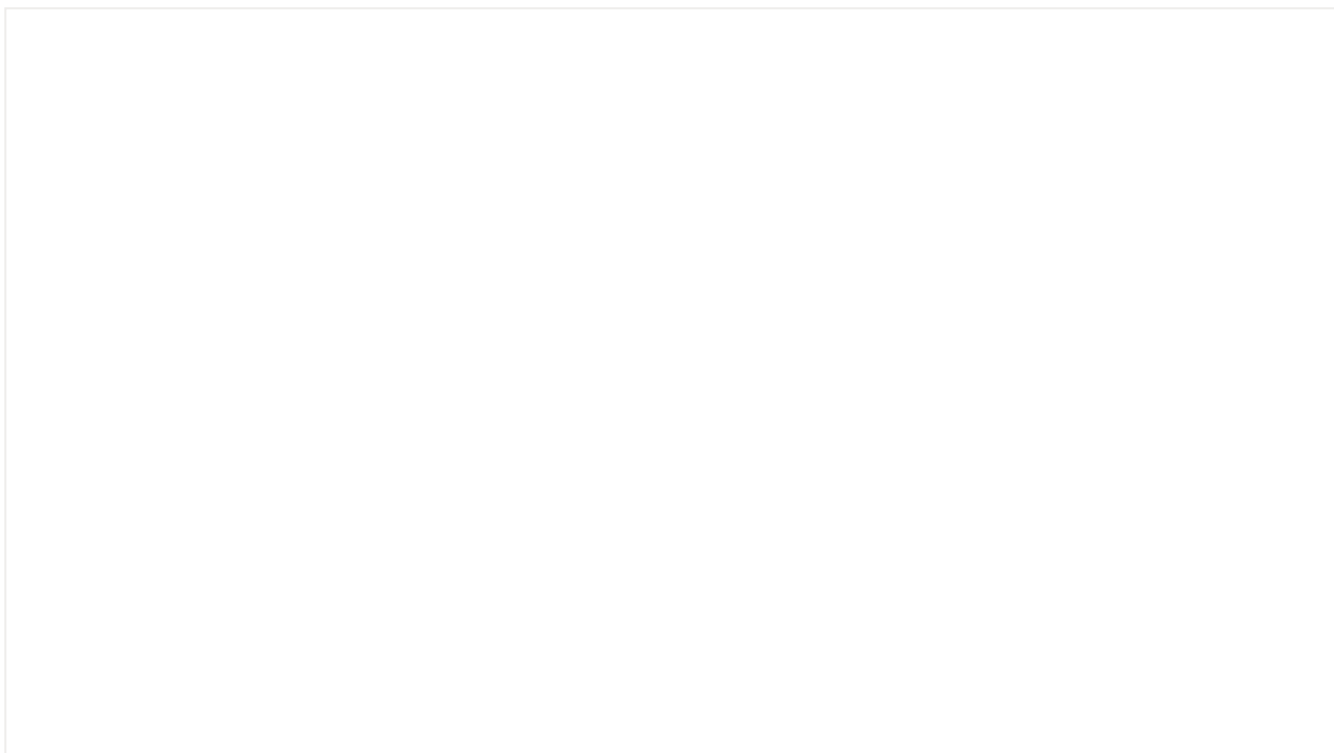
一旦检测到服务过载，需要按照一定的策略对请求进行过滤，前面分析过，对于链式调用的微服务场景，随机丢弃请求会导致整体服务的成功率很低。所以请求是按照优先级进行控制的，优先级低的请求会优先丢弃。

1、业务优先级

对于不同的业务场景优先级是不同的，比如登录场景是最重要的业务，不能登录一切都白瞎，另外支付消息比普通消息优先级高，因为用户对金钱是更敏感的，但普通消息又比朋友圈消息优先级高，所以在微信内是天然存在业务优先级的。

用户的每个请求都会分配一个优先级，并且在微服务的链式调用下，下游请求的优先级也是继承的，比如我请求登录，那么检查账号密码等一系列的后续请求都是继承登录优先级的，这就保证了优先级的一致性。

每个后台服务维护了业务优先级的 hash 表，微信的业务太多了，不是每个业务都记录在表里，不在表里的业务就是最低优先级。



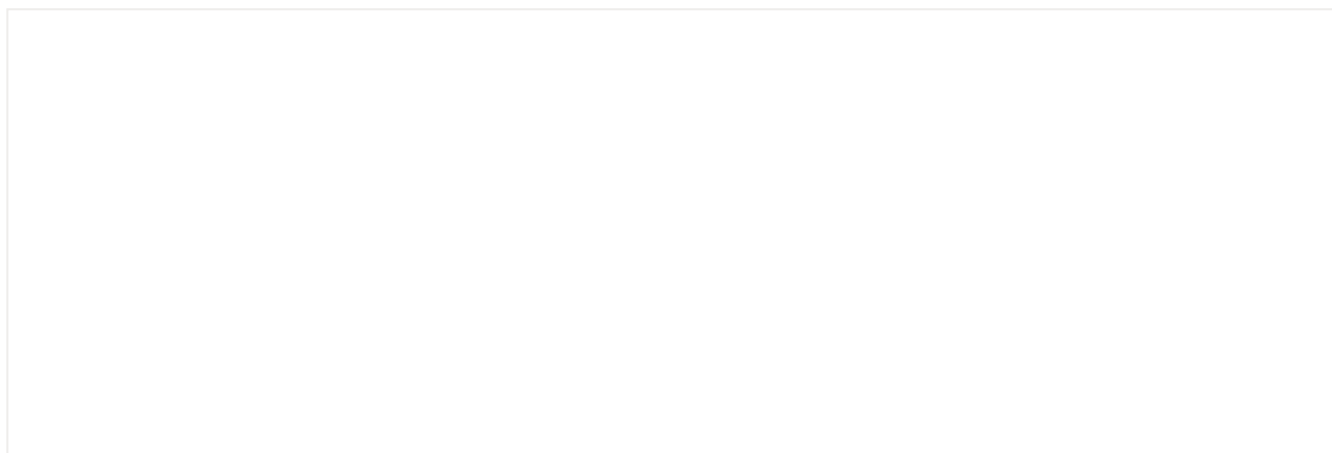
2、用户优先级

很明显，只基于业务优先级的控制是不够的，首先不可能因为负载高，丢弃或允许通过一整个业务的请求，因为每个业务的请求量很大，那一定会造成负载的大幅波动，另外如果在业务中随机丢弃请求，在过载情况下还是会导致整体成功率很低。

为了解决这个问题，可以引入用户优先级，首先用户优先级也不应该相同，对于普通人来说通过 hash 用户唯一 ID，计算用户优先级，为了防止出现总是打豆豆的现象，hash 函数每小时更换，跟业务优先级一样，单个用户的访问链条上的优先级总是一致的。

这里有个疑问，为啥不采用会话 ID 计算优先级呢，从理论上来说采用会话 ID 和用户 ID 效果是一样的，但是采用会话 ID 在用户重新登录时刷新，这个时候可能用户的优先级可能变了，在过载的情况下，他可能因为提高了优先级就恢复了，这样用户会养成坏习惯，在服务有问题时就会重新登录，这样无疑进一步加剧了服务的过载情况。

因为引入了用户优先级，那就和业务优先级组成了一个二维控制平面，根据负载情况，决定这台服务器的准入优先级(B,U)，当过来的请求业务优先级大于 B，或者业务优先级等于 B，但用户优先级高于 U 时，则通过，否则决绝。



3、自适应优先级调整

在大规模微服务场景下，服务器的负载是变化非常频繁的，所以服务器的准入优先级是需要动态变化的，微信分了几十个业务优先级，每个业务优先级下有 128 个用户优先级，所以总的优先级是几千个。

如何根据负载情况调整优先级呢？最简单的方式是从右到左遍历，每调整一次判断下负载情况，这个时间复杂度是 $O(n)$ ，就算使用二分法，时间复杂度也为 $O(\log n)$ ，在数千个优先级下，可能需要数十次调整才能确定一个合适的优先级，每次调整好再统计优先级，可能几十秒都过去了，这个方法无疑是非常低效的。

微信提出了一种基于直方图统计的方法快速调整准入优先级，服务器上维护者目前准入优先级下，过去一个周期的（1s 或 2000 次请求）每个优先级的请求量，当过载时，通过消减下一个周期的请求量来减轻负载，假设上一个周期所有优先级的通过的请求总和是 N ，下一个周期的请求量要减少 $N \cdot a$ ，怎么去减少呢，每提升一个优先级就减少一定的请求量，一直提升到减少的数目大于目标量，恢复负载使用相反的方法，只是系数为 b ，比 a 小，也是为了快降慢升。根据经验值 a 为 5%， b 为 1%。

为了进一步减轻过载机器的压力，能不能在下游过载的情况下不把请求发到下游呢？否则下游还是要接受请求，解包，丢弃请求，白白的浪费带宽，也加重了下游的负载。

为了实现这个能力，在每次请求下游服务时，下游把当前服务的准入优先级返回给上游，上游维护下游服务的准入优先级，如果发现请求优先级达不到下游服务的准入门槛，直接丢弃，而不再请求下游，进一步减轻下游的压力。

六、总结

微信整个负载控制的流程如图所示：



- 当用户从微信发起请求，请求被路由到接入层服务，分配统一的业务和用户优先级，所有到下游的字请求都继承相同的优先级。
- 根据业务逻辑调用 1 个或多个下游服务，当服务收到请求，首先根据自身服务准入优先级判断请求是接受还是丢弃。服务本身根据负载情况周期性的调整准入优先级。

- 当服务需要再向下游发起请求时，判断本地记录的下游服务准入优先级，如果小于则丢弃，如果没有记录或优先级大于记录则向下游发起请求。
- 下游服务返回上游服务需要的信息，并且在信息中携带自身准入优先级。
- 上游接受到返回后解析信息，并更新本地记录的下游服务准入优先级。

整个过载保护的策略有以下三个特点：

- **业务无关的**，使用请求等待时间而不是响应时间，制定用户和业务优先级，这些都与业务本身无关。
- **独立控制和联合控制结合**，准入优先级取决于独立的服务，但又可以联合下游服务的情况，优化服务过载时的表现。
- **高效且公平**，请求链条的优先级是一致的，并且会定时改变 hash 函数调整用户优先级，过载情况下，不会总是影响固定的用户。

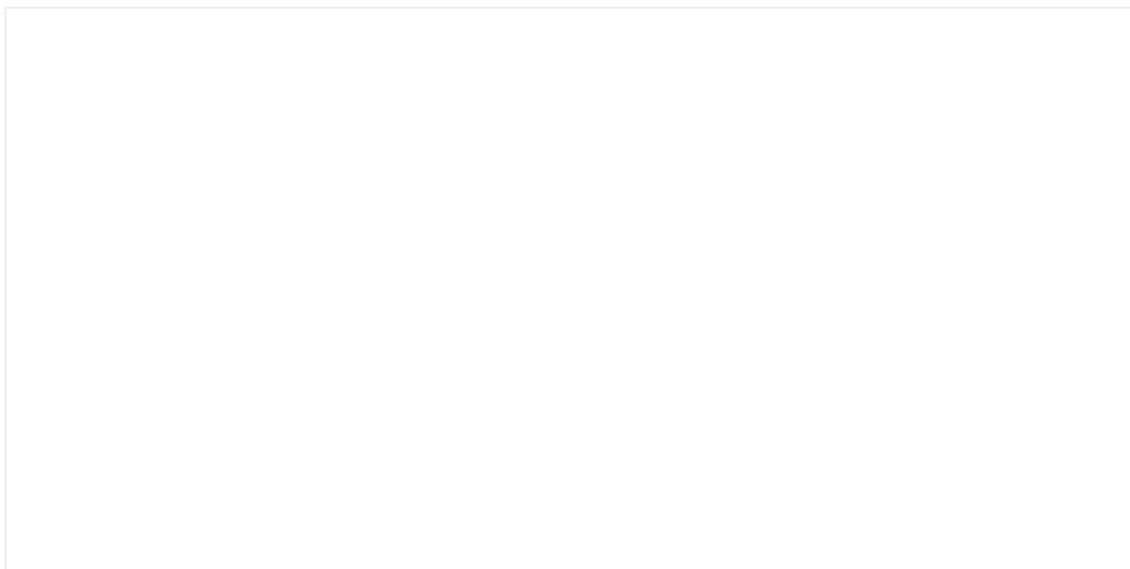
>>>> 参考资料

- <https://www.cs.columbia.edu/~ruigu/papers/socc18-final100.pdf>

作者 | alexccdong

来源 | 公众号：腾讯技术工程 (ID: Tencent_TEG)

dbaplus社群欢迎广大技术人员投稿，投稿邮箱：editor@dbaplus.cn



喜欢此内容的人还喜欢

重新洗牌 | 社群即将全面升级！

七级宇宙数字科技

微信可以开“小号”了，工作生活轻松分开

电脑报

说说你微信ID的由来

文案局