## 优质IT资源微信x923713



### 《手写OS操作系统》小班二期招生,全程直播授课,大牛带你掌握硬核技术!



慕课网首页

免费课

实战课

体系课

慕课教程

专栏 手记

企业服务

Q 🗏

从所有教程的词条中查询...

8.常见的负载均... 下一节 ▶

.

我拍

莩

◆ 上一节 6. 动态获取可用...

### 全部开发者教程 :Ξ

- 11. 分词的重要性
- 11. 万四里文压
- 12. ik分词器安装和配置
- 13. es集成到mxshop\_srvs
- 14. 倒排索引算法 扩展文档

#### 第17周 分布式理论基础、分 布式事务解决方案

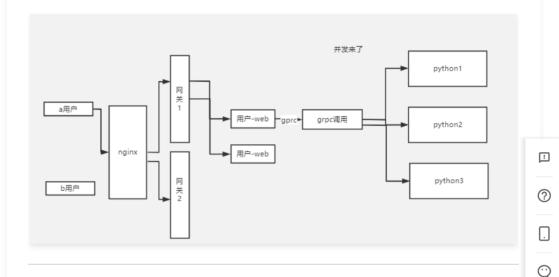
- 1. 事务和分布式事务
- 2. 程序出哪些问题会导致数据 不一致?
- 3. CAP和BASE理论
- 4. 两\_三阶段提交
- 5. tcc分布式事务
- 6. 基于本地消息表的最终一致性
- 7. 基于可靠消息的最终一致 性- 最常用

#### 首页 > 慕课教程 > Go工程师体系课全新版 > 7. 负载均衡策略

bobby・更新于 2022-11-08

# 2. 负载均衡策略

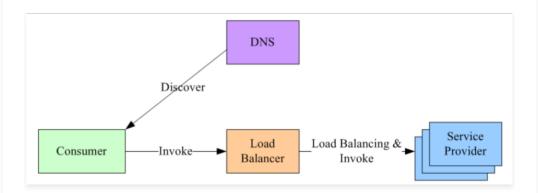
### 1. 什么是负载均衡



### 2. 负载均衡策略

#### 1. 集中式load balance

集中式LB方案,如下图。首先,服务的消费方和提供方不直接耦合,而是在服务消费者和服务提供者之间有一个独立的LB(LB通常是专门的硬件设备如F5,或者基于软件如LVS,HAproxy等实现)。



LB上有所有服务的地址映射表,通常由运维配置注册,当服务消费方调用某个目标服务时,它向LB发起请求,由LB以某种策略(比如Round-Robin)做负载均衡后将请求转发到目标服务。





□ 标记书签

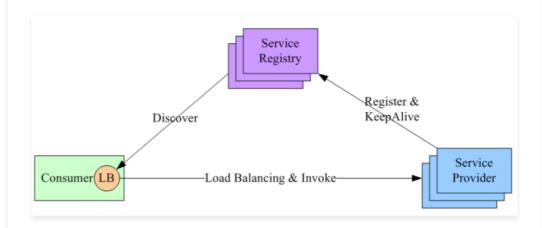
# 优质IT资源微信x923713

服务消费方如何发现LB呢?通常的做法是通过DNS,运维人员为服务配置一个DNS域名,这个域名指向LB。

这种方案基本可以否决,因为它有致命的缺点: 所有服务调用流量都经过load balance服务器,所以load balance服务器成了系统的单点,一旦LB发生故障对整个系统的影响是灾难性的。为了解决这个问题,必然需要对这个load balance部件做分布式处理(部署多个实例,冗余,然后解决一致性问题等全家桶解决方案),但这样做会徒增非常多的复杂度。

#### 2. 进程内load balance

进程内load balance。将load balance的功能和算法以sdk的方式实现在客户端进程内。先看架构图:



可看到引入了第三方: 服务注册中心。它做两件事:

- 维护服务提供方的节点列表,并检测这些节点的健康度。检测的方式是:每个节点部署成功,都通知服务注册中心;然后一直和注册中心保持心跳。
- 2. 允许服务调用方注册感兴趣的事件,把服务提供方的变化情况推送到服务调用方。

这种方案下,整个load balance的过程是这样的:

- 1. 服务注册中心维护所有节点的情况。
- 2. 任何一个节点想要订阅其他服务提供方的节点列表,向服务注册中心注册。
- 3. 服务注册中心将服务提供方的列表(以长连接的方式)推送到消费方。
- 4. 消费方接收到消息后,在本地维护一份这个列表,并自己做load balance。

可见,服务注册中心充当什么角色?它是唯一一个知道整个集群内部所有的节点情况的中心。所以对它的可用性要求会非常高,这个组件可以用Zookeeper实现。

这种方案的缺点是:每个语言都要研究一套sdk,如果公司内的服务使用的语言五花八门的话,这方案的成本会很高。第二点是:后续如果要对客户库进行升级,势必要求服务调用方修改代码并重新发布,所以该方案的升级推广有不小的阻力。

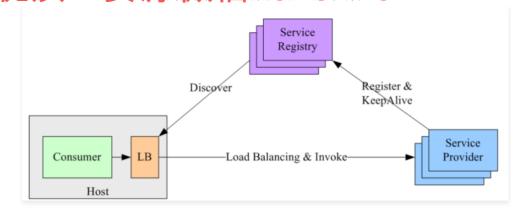
#### 3. 独立进程load balance

该方案是针对第二种方案的不足而提出的一种折中方案,原理和第二种方案基本类似,不同之处是,他将 LB和服务发现功能从进程内移出来,变成主机上的一个独立进程,主机上的一个或者多个服务要访问目 标服务时,他们都通过同一主机上的独立LB进程做服务发现和负载均衡。如图 ②

 $\Box$ 

0

## 优质IT资源微信x923713



这个方案解决了上一种方案的问题,不需要为不同语言开发客户库,LB的升级不需要服务调用方改代码。

但新引入的问题是:这个组件本身的可用性谁来维护?还要再写一个watchdog去监控这个组件?另外,多了一个环节,就多了一个出错的可能,线上出问题了,也多了一个需要排查的环节。

▶ 我要提出意见反馈

企业服务 网站地图 网站首页 关于我们 联系我们 讲师招募 帮助中心 意见反馈 代码托管

Copyright © 2022 imooc.com All Rights Reserved | 京ICP备 12003892号-11 京公网安备11010802030151号

0