(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 111800458 B
(45)授权公告日 2021.04.23

(21)申请号 202010441526.9

(22)申请日 2020.05.22

(65)同一申请的已公布的文献号
申请公布号 CN 111800458 A

(43)申请公布日 2020.10.20

(73)专利权人 浙商银行股份有限公司
地址 311200 浙江省杭州市萧山区鸿宁路1788号

(72)发明人 李富 赵鸿博 富浩

(74)专利代理机构 杭州求是专利事务所有限公司 33200
代理人 刘静

(51)Int.Cl.
H04L 29/08(2006.01)
H04L 29/12(2006.01)

(56)对比文件
CN 105245617 A,2016.01.13
CN 107465765 A,2017.12.12

审查员 李亢亢

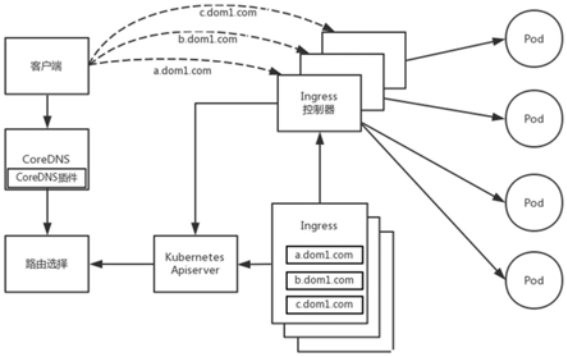
权利要求书3页 说明书7页 附图1页

(54)发明名称

一种Kubernetes容器云平台的动态负载均衡方法及系统

(57)摘要

本发明公开了一种Kubernetes容器云平台的动态负载均衡方法及系统，该系统包括客户端、Core DNS、CoreDNS插件、路由选择、kubernetes apiserver、若干Ingress、若干Ingress控制器、若干Pod；本发明通过自定义CoreDNS插件提供的扩展能力，基于各Ingress控制器健康状态及负载情况，将客户端请求域名解析到当前处于健康状态且负载最低的Ingress控制器IP，该过程无需配置修改CoreDNS域名解析，实现动态负载均衡，克服目前Ingress方案单点问题以及难以适应容器云平台动态负载均衡需求的问题，并可通过横向扩展Ingress控制器数量提供高性能负载转发能力，避免转发能力瓶颈问题。



CN 111800458 B

1.一种Kubernetes容器云平台的动态负载均衡系统,其特征在于,包括:客户端、Core DNS、CoreDNS插件、路由选择、kubernetes apiserver、若干Ingress、若干Ingress控制器、若干Pod;

所述CoreDNS插件,用于拦截CoreDNS收到的DNS查询请求,提取待查询的域名,查找该域名对应的Ingress控制器IP,若本地缓存无记录则向路由选择发起域名解析请求;

所述路由选择,通过定时从kubernetes apiserver读取容器云平台Ingress和Ingress控制器信息,维护域名和Ingress控制器的对应关系;通过定时读取各Ingress控制器的负载状态,计算并更新其权重;当接收到域名解析请求后,返回选择出的Ingress控制器IP;

所述Ingress控制器,通过不断的与Kubernetes apiserver交互,实时感知Ingress的变化,动态更新负载均衡配置并提供访问入口。

2.根据权利要求1所述的一种Kubernetes容器云平台的动态负载均衡系统,其特征在于,所述客户端可以是移动设备或固定设备,也可以是容器云平台外部服务;

所述Core DNS位于独立服务器中,或多台服务器组成的集群中;

所述Core DNS插件与Core DNS绑定在一起;

所述路由选择位于容器云平台内部、独立服务器或多台服务器组成的集群中;

所述Ingress位于容器云平台内部;

所述Ingress控制器位于容器云平台内部单个或多个主机节点,每个主机节点最多有1个Ingress控制器。

3.根据权利要求1所述的一种Kubernetes容器云平台的动态负载均衡系统,其特征在于,所述CoreDNS插件包括请求解析模块,所述请求解析模块用于处理DNS查询请求,根据DNS查询请求携带的待解析域名和客户端IP,首先从本地缓存中查找域名,若存在该域名则直接返回域名对应的Ingress控制器IP;若无记录则向路由选择发起域名解析请求,如果请求成功,则将查询到的域名对应的Ingress控制器IP存入缓存。

4.根据权利要求1所述的一种Kubernetes容器云平台的动态负载均衡系统,其特征在于,所述CoreDNS插件包括缓存管理模块,所述缓存管理模块用于维护缓存列表,单个缓存条目包括域名、Ingress控制器IP和最近一次被访问时间;定时检查各缓存条目是否过期,若过期则重新发起DNS查询请求更新缓存;若缓存条目长时间未被查询,则从缓存列表中删除该条目;若接收到路由选择监控的异常Ingress控制器的下线请求时,删除异常Ingress控制器对应的缓存条目。

5.根据权利要求1所述的一种Kubernetes容器云平台的动态负载均衡系统,其特征在于,所述路由选择包括服务维护模块,所述服务维护模块定时请求Kubernetes apiserver,获取容器云平台中Ingress中的域名信息和Ingress控制器IP及端口,并根据Ingress控制器的检测接口读取的负载状态数据,计算各Ingress控制器的权重值,创建服务列表,服务列表中单个条目包括服务名和一个或多个服务实例,服务名为域名,服务实例为Ingress控制器,服务实例信息包括Ingress控制器的名称、IP、端口和权重;当接收到域名解析请求后,返回选择出的Ingress控制器IP。

6.根据权利要求5所述的一种Kubernetes容器云平台的动态负载均衡系统,其特征在于,所述路由选择包括Ingress控制器选择模块,所述Ingress控制器选择模块根据Ingress控制器的权重值,从服务名对应的服务实例的Ingress控制器列表中选择一个。

7.根据权利要求5所述的一种Kubernetes容器云平台的动态负载均衡系统,其特征在于,所述路由选择包括健康监控模块,所述健康监控模块周期性轮询请求Ingress控制器检测接口,若返回状态数据则认为Ingress控制器可用,否则认为Ingress控制器异常,并将该异常Ingress控制器从服务列表中每条域名后对应的服务实例的Ingress控制器列表中删除。

8.一种Kubernetes容器云平台的动态负载均衡方法,其特征在于,该方法包括:

路由选择定时从kubernetes apiserver获取容器云平台所有Ingress和Ingress控制器,通过Ingress控制器检测接口读取负载状态数据,计算各Ingress控制器权重值,创建服务列表,服务条目包括服务名和服务实例,服务名为域名,服务实例为Ingress控制器;

客户端访问服务域名时,向Core DNS发起DNS查询请求;CoreDNS收到的DNS查询请求被CoreDNS插件拦截;CoreDNS插件提取该DNS查询请求中携带的待解析域名,查找本地缓存是否有解析记录,若无记录则向路由选择发起域名解析请求;

路由选择收到域名解析请求后,返回选择出的Ingress控制器IP;

客户端收到域名解析结果后,通过对应的Ingress控制器将请求连接转发到应用Pod。

9.根据权利要求8所述的一种Kubernetes容器云平台的动态负载均衡方法,其特征在于,所述路由选择根据Ingress控制器检测接口读取负载状态数据,计算各Ingress控制器的权重值,计算策略如下:

策略1,通过访问Ingress控制器检测接口,获取Ingress控制器活动连接数,选取最小活动连接数为除数,各Ingress控制器活动连接数作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值;

策略2,通过访问Ingress控制器检测接口,获取Ingress控制器内存使用量,选取最小内存使用量为除数,各Ingress控制器内存使用量作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值;

策略3,通过访问Ingress控制器提供接口,获取Ingress控制器CPU使用率,选取最小CPU使用率为除数,各Ingress控制器CPU使用率作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值。

10.根据权利要求8所述的一种Kubernetes容器云平台的动态负载均衡方法,其特征在于,所述路由选择根据各Ingress控制器的权重值选择Ingress控制器,包括:

假设有n个Ingress控制器,其权重值分别为:

$a_0,a_1,a_2\cdots a_{n-1}(0\leqslant a_i\leqslant 1)$

归一化处理后,得到新的数组,

$$b_i=\frac{a_i}{\sum_{j=0}^{n-1}a_j}\ (0\leq i\leq n-1)$$

经过运算,得到有序数组,数组从小到大排列,

$$c_i=\sum_{j=0}^{i}b_j\ (0\leq i\leq n-1)$$

在[0,1]区间生成均匀分布随机数r,

在有序数组$c_i(0\leqslant i\leqslant n-1)$中,从左向右查找第一个大于r的数,得到其在数组的位置i(0≤i≤n-1;

则选择第i个Ingress控制器作为结果。

一种Kubernetes容器云平台的动态负载均衡方法及系统

技术领域

[0001] 本发明属于通信技术领域，尤其涉及一种Kubernetes容器云平台的动态负载均衡方法及系统。

背景技术

[0002] Kubernetes是一个基于容器技术的分布式架构领先方案，已成为云环境中大规模部署容器化应用程序的事实标准。通过提供Service资源，实现对容器的服务发现与负载均衡。

[0003] 目前Service共有LoadBalancer、NodePort和Ingress三种服务暴露的方式。其中LoadBalancer只能在某些公有云使用，NodePort由于每个端口只能供一个服务使用而容易造成端口冲突。而Ingress通过与Ingress控制器结合使用，可动态监控pod及service的变化，提供完善的四层及七层负载均衡能力，在同一IP下可对外提供多个服务。因此Ingress广泛应用在容器云平台用于提供统一的服务入口。

[0004] Ingress通过指定host字段和tcp、udp端口，以Kubernetes集群中单个Ingress控制器IP对外提供服务，如果提供服务的Ingress控制器故障，需要客户端重新修改DNS配置，将服务域名配置到可用Ingress控制器IP。若故障Ingress控制器IP承载较多域名，切换过程会造成长时间业务中断。

[0005] 在高流量高并发业务场景中，单个Ingress控制器容易遇到转发性能瓶颈，通过增加Ingress控制器数量，可将达到性能瓶颈的Ingress控制器的部分服务域名转移到低负载Ingress控制器，但需要手动频繁修改DNS配置，不仅造成业务中断同时难以适应容器云平台快速动态变化的负载均衡需求。

发明内容

[0006] 本发明的目的是提供一种Kubernetes容器云平台的动态负载均衡方法及系统，克服目前Ingress方案单点故障和难以适应容器云平台动态变化的负载均衡需求的问题。

[0007] 为实现上述目的，本发明提供了如下技术方案：

[0008] 一方面，本申请提出了一种Kubernetes容器云平台的动态负载均衡系统，包括：客户端、Core DNS、CoreDNS插件、路由选择、kubernetes apiserver、若干Ingress、若干Ingress控制器、若干Pod；

[0009] 所述CoreDNS插件，用于拦截CoreDNS收到的DNS查询请求，提取待查询的域名，查找该域名对应的Ingress控制器IP，若本地缓存无记录则向路由选择发起域名解析请求；

[0010] 所述路由选择，通过定时从kubernetes apiserver读取容器云平台Ingress和Ingress控制器信息，维护域名和Ingress控制器的对应关系；通过定时读取各Ingress控制器的负载状态，计算并更新其权重；当接收到域名解析请求后，返回选择出的Ingress控制器IP；

[0011] 所述Ingress控制器，通过不断的与Kubernetes apiserver交互，实时感知

Ingress的变化,动态更新负载均衡配置并提供访问入口。

[0012] 进一步地,所述客户端可以是移动设备或固定设备,也可以是容器云平台外部服务;

[0013] 所述Core DNS位于独立服务器中,或多台服务器组成的集群中;

[0014] 所述Core DNS插件与Core DNS绑定在一起;

[0015] 所述路由选择位于容器云平台内部、独立服务器或多台服务器组成的集群中;

[0016] 所述Ingress位于容器云平台内部;

[0017] 所述Ingress控制器位于容器云平台内部单个或多个主机节点,每个主机节点最多有1个Ingress控制器。

[0018] 进一步地,所述CoreDNS插件包括请求解析模块,所述请求解析模块用于处理DNS查询请求,根据DNS查询请求携带的待解析域名和客户端IP,首先从本地缓存中查找域名,若存在该域名则直接返回域名对应的Ingress控制器IP;若无记录则向路由选择发起域名解析请求,如果请求成功,则将查询到的域名对应的Ingress控制器IP存入缓存。

[0019] 进一步地,所述CoreDNS插件包括缓存管理模块,所述缓存管理模块用于维护缓存列表,单个缓存条目包括域名、Ingress控制器IP和最近一次被访问时间;定时检查各缓存条目是否过期,若过期则重新发起DNS查询请求更新缓存;若缓存条目长时间未被查询,则从缓存列表中删除该条目;若接收到路由选择监控的异常Ingress控制器的下线请求时,删除异常Ingress控制器对应的缓存条目。

[0020] 进一步地,所述路由选择包括服务维护模块,所述服务维护模块定时请求Kubernetes apiserver,获取容器云平台中Ingress中的域名信息和Ingress控制器IP及端口,并根据Ingress控制器的检测接口读取的负载状态数据,计算各Ingress控制器的权重值,创建服务列表,服务列表中单个条目包括服务名和一个或多个服务实例,服务名为域名,服务实例为Ingress控制器,服务实例信息包括Ingress控制器的名称、IP、端口和权重;当接收到域名解析请求后,返回选择出的Ingress控制器IP。

[0021] 进一步地,所述路由选择包括Ingress控制器选择模块,所述Ingress控制器选择模块根据Ingress控制器的权重值,从服务名对应的服务实例的Ingress控制器列表中选择一个。

[0022] 进一步地,所述路由选择包括健康监控模块,所述健康监控模块周期性轮询请求Ingress控制器检测接口,若返回状态数据则认为Ingress控制器可用,否则认为Ingress控制器异常,并将该异常Ingress控制器从服务列表中每条域名后对应的服务实例的Ingress控制器列表中删除。

[0023] 另一方面,本申请提出一种Kubernetes容器云平台的动态负载均衡方法,该方法包括:

[0024] 路由选择定时从kubernetes apiserver获取容器云平台所有Ingress和Ingress控制器,通过Ingress控制器检测接口读取负载状态数据,计算各Ingress控制器权重值,创建服务列表,服务条目包括服务名和服务实例,服务名为域名,服务实例为Ingress控制器;

[0025] 客户端访问服务域名时,向Core DNS发起DNS查询请求;CoreDNS收到的DNS查询请求被CoreDNS插件拦截;CoreDNS插件提取该DNS查询请求中携带的待解析域名,查找本地缓存是否有该条解析记录,若无记录则向路由选择发起域名解析请求;

[0026] 路由选择收到域名解析请求后,返回选择出的Ingress控制器IP;

[0027] 客户端收到域名解析结果后,通过对应的Ingress控制器将请求连接转发到应用Pod。

[0028] 进一步地,所述路由选择根据Ingress控制器检测接口读取负载状态数据,计算各Ingress控制器的权重值,计算策略如下:

[0029] 策略1,通过访问Ingress控制器检测接口,获取Ingress控制器活动连接数,选取最小活动连接数为除数,各Ingress控制器活动连接数作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值;

[0030] 策略2,通过访问Ingress控制器检测接口,获取Ingress控制器内存使用量,选取最小内存使用量为除数,各Ingress控制器内存使用量作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值;

[0031] 策略3,通过访问Ingress控制器提供接口,获取Ingress控制器CPU使用率,选取最小CPU使用率为除数,各Ingress控制器CPU使用率作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值。

[0032] 进一步地,所述路由选择根据各Ingress控制器的权重值选择Ingress控制器,包括:

[0033] 假设有n个Ingress控制器,其权重值分别为:

[0034] $a_0, a_1, a_2 \cdots a_{n-1} (0 \leqslant a_i \leqslant 1)$

[0035] 归一化处理后,得到新的数组,

[0036] $$b_i = \frac{a_i}{\sum_{j=0}^{n-1} a_j} \quad (0 \leq i \leq n-1)$$

[0037] 经过运算,得到有序数组,数组从小到大排列,

[0038] $$c_i = \sum_{j=0}^{i} b_j \quad (0 \leq i \leq n-1)$$

[0039] 在[0,1]区间生成均匀分布随机数r,

[0040] 在有序数组$c_i (0 \leqslant i \leqslant n-1)$中,从左向右查找第一个大于r的数,得到其在数组的位置i0≤i≤n-1)。

[0041] 则选择第i个Ingress控制器作为结果。

[0042] 本发明的有益效果是:本发明提供的Kubernetes容器云平台的动态负载均衡方法及系统,通过自定义CoreDNS插件提供的扩展能力,基于各Ingress控制器健康状态及负载情况,将客户端请求域名解析到当前处于健康状态且负载最低的Ingress控制器IP,该过程无需配置修改CoreDNS域名解析,实现动态负载均衡,克服目前Ingress方案单点问题以及难以适应容器云平台动态负载均衡需求的问题,并可通过横向扩展Ingress控制器数量提供高性能负载转发能力,避免转发能力瓶颈问题。

附图说明

[0043] 为了更清楚地说明本发明实施例或现有技术中的技术方案,下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍,显而易见地,下面描述中的附图仅仅是本发明的一些实施例,对于本领域普通技术人员来讲,在不付出创造性劳动的前提下,还可以

根据这些附图获得其他的附图。

[0044] 图1为本申请实施例提供的Kubernetes容器云平台的动态负载均衡系统的一种结构示意图。

[0045] 图2为本申请实施例提供的Kubernetes容器云平台的动态负载均衡方法的一种实现流程图。

具体实施方式

[0046] 下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例仅仅是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有付出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0047] 请参阅图1，图1为本申请实施例提供的Kubernetes容器云平台的动态负载均衡系统的一种结构示意图，包括：至少一客户端、Core DNS、CoreDNS插件、路由选择、kubernetes apiserver、若干Ingress、若干Ingress控制器、若干Pod。

[0048] 客户端可以是移动设备或固定设备，也可以是容器云平台外部服务。Core DNS可以位于独立服务器中，或多台服务器组成的集群中。Core DNS插件与Core DNS绑定在一起。路由选择可以位于容器云平台内部，也可位于独立服务器或多台服务器组成的集群中。Ingress位于容器云平台内部。Ingress控制器位于容器云平台内部单个或多个主机节点，每个主机节点最多有1个Ingress控制器，通过扩展Ingress控制器个数可提高系统转发能力。Pod中承载的服务可以包括但不限于Web应用。

[0049] 进一步地，应用以Pod形式运行在Kubernets容器云平台，可根据业务负载进行扩容缩容，使业务具备横向扩展能力。

[0050] 所述Ingress是Kubernets提供的一种对集群外部提供服务的解决方案，是外部连接到达容器服务的规则集合，指定了域名和应用Pod的对应关系。

[0051] 所述Ingress控制器，通过不断的跟Kubernetes apiserver交互，实时感知Ingress的变化，动态更新负载均衡配置并提供访问入口，Ingress控制器的常用开源实现方式有nginx、haproxy、traefik等，均可对外提供检测接口例如prometheus监控接口/metrics，供prometheus或其它软件读取Ingress控制器状态数据。

[0052] 所述CoreDNS是Golang编写的一个插件式DNS服务器，通过扩展插件可完成自定义功能，是Kubernetes 1.13后内置的默认DNS服务器，提供容器云平台内部服务发现能力，也可部署在kubernetes集群外部作为独立DNS。

[0053] 所述CoreDNS插件，用于拦截CoreDNS收到的DNS查询请求，提取要查询的域名，查找该域名对应的Ingress控制器IP。

[0054] 所述路由选择，通过定时从kubernetes apiserver读取容器云平台Ingress和Ingress控制器信息，维护域名和Ingress控制器的对应关系。通过定时读取各Ingress控制器的负载状态，计算并更新其权重。当接收到域名解析请求后，根据Ingress控制器选择算法，返回选择出的Ingress控制器IP。

[0055] 所述kubernetes apiserver是kubernetes重要组件之一，提供了kubernetes容器云平台各类资源对象的增删改查及watch等HTTP Rest接口，是整个kubernetes集群的数据

总线和数据中心,提供了集群管理的REST API接口(包括认证授权、数据校验以及集群状态变更)。

[0056] 进一步地,所述CoreDNS插件,包括请求解析模块以及缓存管理模块:

[0057] 所述请求解析模块,用于处理拦截到的DNS查询请求,根据DNS查询请求携带的需要解析的域名和客户端IP,首先从本地缓存中查找域名,若存在该域名,则直接返回域名对应的Ingress控制器IP;若无记录,则向路由选择发起域名解析请求,如果查询成功,则将查询得到的域名对应的Ingress控制器IP存入缓存,否则报错。

[0058] 所述缓存管理模块,用于维护缓存列表,单个缓存条目以key:value的形式存储,域名为key,value值包括Ingress控制器IP和最近一次被访问时间。定时检查各缓存条目是否过期,若过期则向所述请求解析模块重新发起DNS查询请求更新缓存。若缓存条目长时间未被查询,则从缓存列表中删除该条目。当收到Ingress控制器下线请求时,则遍历缓存列表条目,若条目value值中的Ingress控制器IP与Ingress控制器下线请求中携带的Ingress控制器IP相同,则删除该缓存条目。

[0059] 进一步地,所述路由选择包括服务维护模块、Ingress控制器选择模块和健康监控模块:

[0060] 所述服务维护模块定时请求Kubernetes apiserver,获取容器云平台中Ingress中的域名信息和Ingress控制器IP及端口,并根据Ingress控制器的/metrics接口读取的负载状态数据,计算各Ingress控制器的权重值,创建服务列表,服务列表中单个条目包括服务名和一个或多个服务实例,服务名为域名,服务实例为Ingress控制器,服务实例信息包括Ingress控制器的名称、IP、端口和权重,当接收到域名解析请求后,调用Ingress控制器选择模块,返回选择出的Ingress控制器IP。

[0061] 所述Ingress控制器选择模块,根据Ingress控制器的权重值,按照权重值算法从服务名对应的服务实例的Ingress控制器列表中选择一个。

[0062] 所述健康监控模块周期性轮询请求Ingress控制器提供的/metrics接口,若返回状态数据则认为Ingress控制器可用,否则认为Ingress控制器异常,并将该异常Ingress控制器从服务列表中每条域名后对应的服务实例的Ingress控制器列表中删除,并向CoreDNS插件的缓存管理模块发送Ingress控制器下线请求,请求中携带异常Ingress控制器IP。

[0063] 本发明实施例提供的Kubernetes容器云平台的动态负载均衡方法可以基于图1所示的Kubernetes容器云平台的动态负载均衡系统实现,当然,也可以基于其它Kubernetes容器云平台的动态负载均衡系统实现,只要该Kubernetes容器云平台的动态负载均衡系统包括Core DNS插件、路由选择和Ingress控制器。

[0064] 如图2所示,为本发明实施例提供的Kubernetes容器云平台的动态负载均衡方法的一种实现流程图,可以包括:

[0065] 步骤1:服务维护模块定时(每60秒)从kubernetes apiserver获取容器云平台所有Ingress和Ingress控制器,通过Ingress控制器提供的/metrics接口读取负载状态数据,依次计算各Ingress控制器的负载情况,可有多种计算负载情况的策略,计算策略可在运行过程中动态配置修改。

[0066] 策略1,通过访问Ingress控制器提供的/metrics接口,获取Ingress控制器活动连接数,选取最小活动连接数为除数,各Ingress控制器活动连接数作为被除数,依次做除法

运算,运算结果作为各Ingress控制器的权重值。如被除数中出现0值,此Ingress控制器权重值为1。

[0067] 策略2,通过访问Ingress控制器提供的/metrics接口,获取Ingress控制器内存使用量,选取最小内存使用量为除数,各Ingress控制器内存使用量作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值。如被除数中出现0值,此Ingress控制器权重值为1。

[0068] 策略3,通过访问Ingress控制器提供的/metrics接口,获取Ingress控制器CPU使用率,选取最小CPU使用率为除数,各Ingress控制器CPU使用率作为被除数,依次做除法运算,运算结果作为各Ingress控制器的权重值。如被除数中出现0值,此Ingress控制器权重值为1。

[0069] 若计算完成各Ingress控制器权重后,依次提取各Ingress中host字段的域名作为服务名,全部Ingress控制器作为服务实例,实例信息包括Ingress控制器名称、IP、端口和权重,形成服务列表。

[0070] Ingress控制器在容器云平台单台主机节点上最多创建一个,以daemonset形式部署在容器云平台,因此各Ingress控制器的cpu和内存配置均一致。当Ingress控制器需要横向扩展时,需要通过在目标主机节点上打标签,将新增Ingress控制器调度到目标主机节点上。

[0071] 步骤2:健康监控模块定时(每3秒)telnet各Ingress控制器IP+端口,若连接成功,认为该Ingress控制器正常,否则认为该Ingress控制器异常,遍历服务列表中所有服务,从服务实例中的Ingress控制器列表删除该Ingress控制器,并向CoreDNS插件的缓存管理模块发送Ingress控制器下线请求,请求中携带异常Ingress控制器IP。

[0072] 步骤3:客户端访问服务域名时,首先向Core DNS发起DNS查询请求。客户端要将DNS地址配置为Core DNS的IP。

[0073] 步骤4:CoreDNS收到的DNS查询请求被CoreDNS插件拦截。CoreDNS插件提取该DNS查询请求中携带的待解析域名,以待解析域名为key,查找缓存管理模块维护的缓存列表。

[0074] 步骤5:若在缓存列表查到记录,则将该条记录中value值中的最近一次被访问时间更新为当前时间,并返回value中的Ingress控制器IP。

[0075] 缓存管理模块定时(每30秒)遍历缓存列表,用当前时间减去缓存条目value值中的最近一次被访问时间,若差值大于2分钟,则将该条记录判断为过期,将该记录从缓存列表中删除。

[0076] 缓存管理模块收到Ingress控制器下线请求时,遍历缓存列表条目,若条目value值中的Ingress控制器IP与Ingress控制器下线请求中携带的Ingress控制器IP相同,则删除该缓存条目。

[0077] 若缓存列表无记录,则向请求解析模块发起域名解析请求,请求中携带待解析的域名。

[0078] 步骤6:请求解析模块收到上述的域名解析请求后,通过调用服务维护模块解析域名对应Ingress控制器IP。若成功返回IP值,则通知缓存管理模块在缓存列表中增加缓存条目,域名作为key值,value值为上述返回IP值和当前时间。

[0079] 步骤7:服务维护模块收到待解析域名后,从服务列表中查找该域名,若存在该域

名,通过Ingress控制器选择模块选择其中一个Ingress控制器,并返回其IP值。

[0080] Ingress控制器选择模块是根据各Ingress控制器的权重值来选择,其选择算法如下:

[0081] 假设有n个Ingress控制器,其权重值分别为:

[0082] $a_0,a_1,a_2\cdots a_{n-1}(0\leqslant a_i\leqslant 1)$

[0083] 归一化处理后,得到新的数组,

[0084] $$b_i=\frac{a_i}{\sum_{j=0}^{n-1}a_j}\ (0\leq i\leq n-1)$$

[0085] 经过运算,得到有序数组,数组从小到大排列,

[0086] $$c_i=\sum_{j=0}^{i}b_j\ (0\leq i\leq n-1)$$

[0087] 在[0,1]区间生成均匀分布随机数r,

[0088] 在有序数组$c_i(0\leqslant i\leqslant n-1)$中,从左向右查找第一个大于r的数,得到其在数组的位置i0≤i≤n-1。

[0089] 则选择第i个Ingress控制器作为结果。

[0090] 请求解析模块收到服务维护模块返回的Ingress控制器IP后,组装DNS回复报文,由CoreDNS回复客户端。

[0091] 步骤8:客户端收到域名解析结果后,则通过对应的Ingress控制器将请求连接转发到应用Pod。Ingress控制器的解决方案有nginx、haproxy和traefik等,具有完善的转发策略,可以将请求连接转发至目标Pod。

[0092] 以上所述仅是本发明的优选实施方式,虽然本发明已以较佳实施例披露如上,然而并非用以限定本发明。任何熟悉本领域的技术人员,在不脱离本发明技术方案范围情况下,都可利用上述揭示的方法和技术内容对本发明技术方案做出许多可能的变动和修饰,或修改为等同变化的等效实施例。因此,凡是未脱离本发明技术方案的内容,依据本发明的技术实质对以上实施例所做的任何的简单修改、等同变化及修饰,均仍属于本发明技术方案保护的范围内。

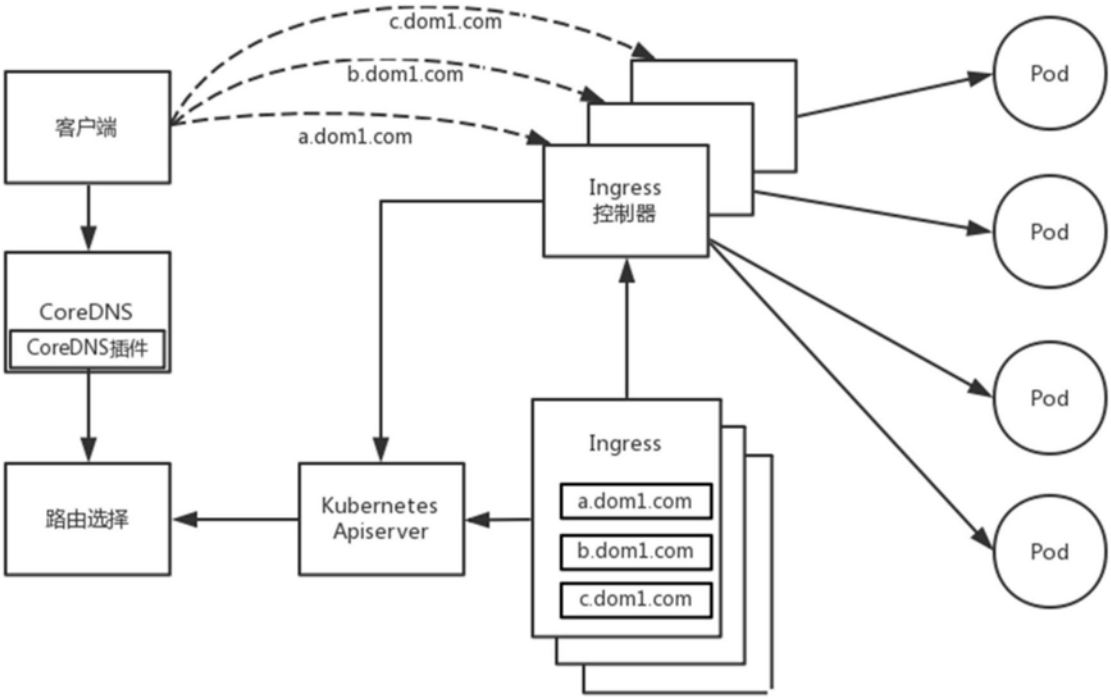

c.dom1.com
b.dom1.com
a.dom1.com
客户端
CoreDNS
CoreDNS插件
路由选择
Kubernetes
Apiserver
Ingress
控制器
Ingress
a.dom1.com
b.dom1.com
c.dom1.com
Pod
Pod
Pod
Pod

图1

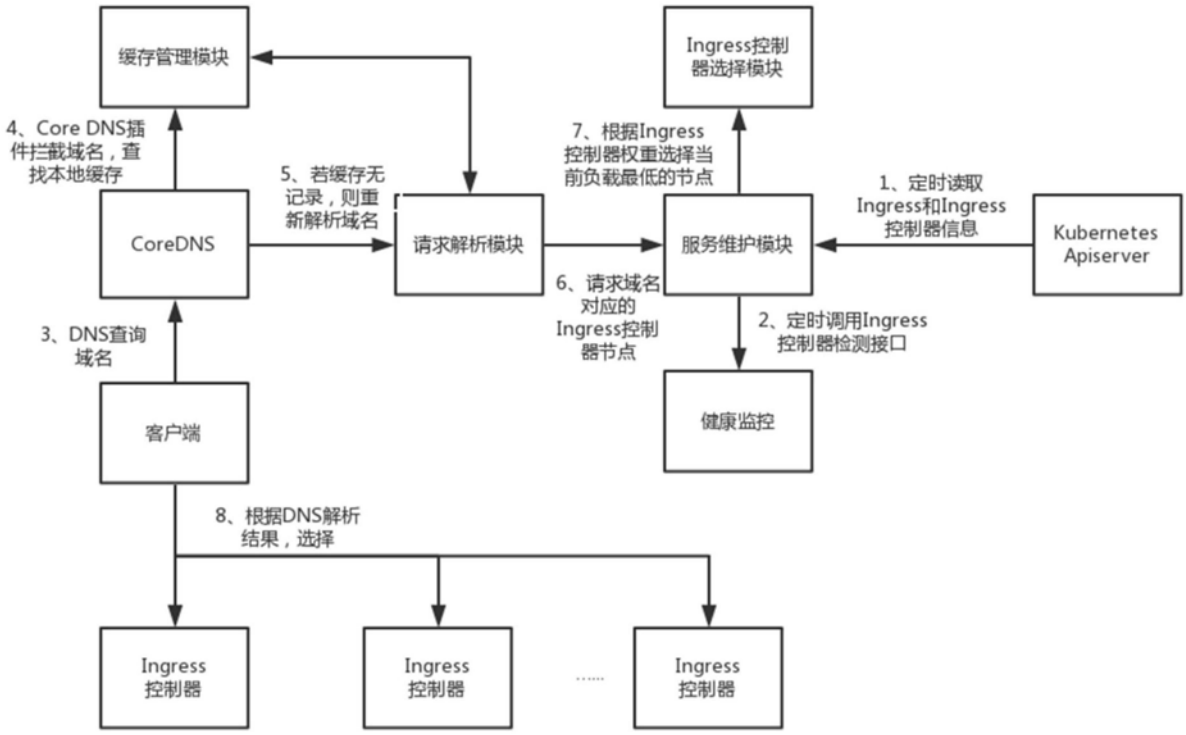

缓存管理模块
4、Core DNS插件拦截域名，查找本地缓存
CoreDNS
5、若缓存无记录，则重新解析域名
请求解析模块
Ingress控制器选择模块
7、根据Ingress控制器权重选择当前负载最低的节点
服务维护模块
1、定时读取Ingress和Ingress控制器信息
Kubernetes
Apiserver
6、请求域名对应的Ingress控制器节点
2、定时调用Ingress控制器检测接口
健康监控
3、DNS查询域名
客户端
8、根据DNS解析结果，选择
Ingress
控制器
Ingress
控制器
.....
Ingress
控制器

图2