集群负载均衡

计算机集群

> 计算机集群是一种计算机系统。通过将松散集成的计算机通过软件或硬件连接起来高度紧密协作的是完成计算工作。在某种意义上，他们可以被看作是一台计算机。集群系统中的单个计算机通常称为节点，通常通过局域网连接，但也有其它的可能连接方式。集群计算机通常用来改进单个计算机的计算速度和/或可靠性。一般情况下集群计算机比单个计算机，比如工作站或超级计算机性能价格比要高得多。

集群分类

> 1. 负载均衡集群(Load balancing clusters)
> 2. 高可用性集群(High-availability (HA) clusters)
> 3. 高性能计算集群(High-performance（HPC）clusters)

负载均衡集群(Load balancing clusters)

> LB负载均衡模型运行时，一般通过一个或者多个前端负载均衡器，将工作负载分发到后端的一组服务器上，从而达到整个系统的高性能和高可用性。这样的计算机集群有时也被称为服务器群（Server Farm）。一般高可用性集群和负载均衡集群会使用类似的技术，或同时具有高可用性与负载均衡的特点。

高可用性集群(High-availability (HA) clusters)

> 一般是指当集群中有某个节点失效的情况下，其上的任务会自动转移到其他正常的节点上。还指可以将集群中的某节点进行离线维护再上线，该过程并不影响整个集群的运行。

负载均衡集群的实现

> 负载均衡集群的实现有通过硬件和软件方式实现。通过硬件的方式有F5、A10等计算机硬件来实现负载均衡集群。通过软件的方式有LVS、haproxy、nginx、ats等，其中LVS基于4层实现，且直接在内核中工作，有较好的性能及负载吞吐率，haproxy、nginx、ats基于7层实现，工作与应用层。

计算机集群体系结构

> 通过　计算机集群伴随着高伸缩性、高可用、高性能的实现。后端存储（backend storage），它为服务器池提供一个共享的存储区，这样很容易使得服务器池拥有相同的内容，提供相同的服务。**调度器**采用IP负载均衡技术、基于内容请求分发技术或者两者相结合。在IP负载均衡技术中，需要服务器池拥有相同的内容提供相同的服务。当客户请求到达时，调度器只根据负载情况从服务器池中选出一个服务器，将该请求转发到选出的服务器，并记录这个调度；当这个请求的其他报文到达，也会被转发到前面选出的服务器。在基于内容请求分发技术中，服务器可以提供不同的服务，当客户请求到达时，调度器可根据请求的内容和服务器的情况选择服务器执行请求。因为所有的操作都是在操作系统核心空间中将完成的，它的调度开销很小，所以它具有很高的吞吐率。
>
> **服务器池**的结点数目是可变的。当整个系统收到的负载超过目前所有结点的处理能力时，可以在服务器池中增加服务器来满足不断增长的请求负载。对大多数网络服务来说，结点与结点间不存在很强的相关性，所以整个系统的性能可以随着服务器池的结点数目增加而线性增长。
>
> **后端存储**通常用容错的分布式文件系统，如AFS、GFS、Coda和Intermezzo等。分布式文件系统为各服务器提供共享的存储区，它们访问分布式文件系统就像访问本地文件系统一样。同时，分布式文件系统提供良好的伸缩性和可用性。然而，当不同服务器上的应用程序同时访问分布式文件系统上同一资源时，应用程序的访问冲突需要消解才能使得资源处于一致状态。这需要一个分布式锁管理器（Distributed Lock Manager），它可能是分布式文件系统内部提供的，也可能是外部的。开发者在写应用程序时，可以使用分布

> 式锁管理器来保证应用程序在不同结点上并发访问的一致性。
> **高速网络。**负载调度器、服务器池和分布式文件系统通过高速网络相连，如100Mbps交换机、Myrinet、CompactNET和Gigabit交换机等。使用高速的网络，主要为避免当系统规模扩大时互联网络成为瓶颈。

LVS集群

LVS介绍

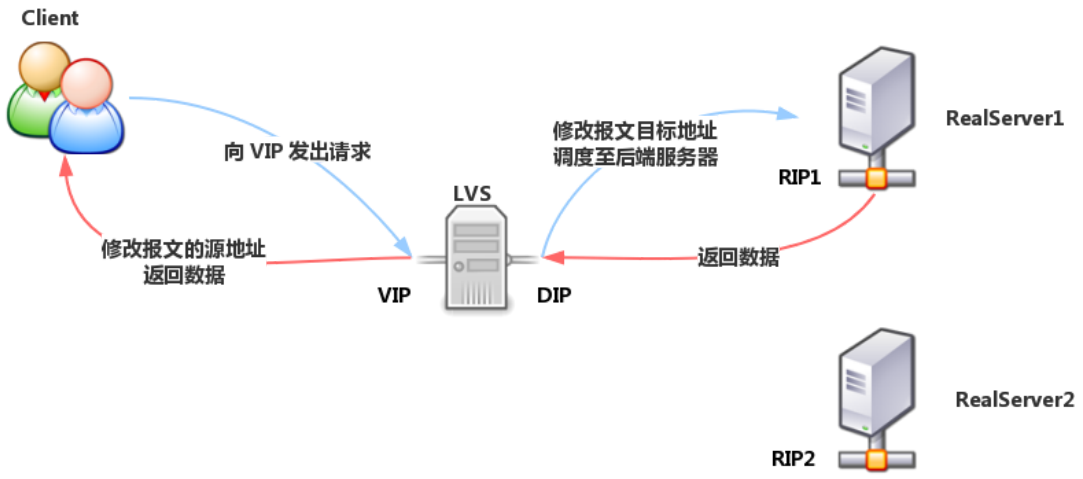
> LVS(Linux Virtual Server)。一组服务器通过高速的局域网或者地理分布的广域网相互连接，在它们的前端有一个负载调度器（Load Balancer）。负载调度器能无缝地将网络请求调度到真实服务器上，从而使得服务器集群的结构对客户是透明的，客户访问集群系统提供的网络服务就像访 问一台高性能、高可用的服务器一样。系统的伸缩性通过在服务机群中透明地加入和删除一个节点来达到，通过检 测节点或服务进程故障和正确地重置系统达到高可用性。由于我们的负载调度技术是在Linux内核中实现的，我们称之为Linux虚拟服务器（Linux Virtual Server）。LVS实现了使用集群技术和Linux操作系统实现一个高性能、高可用的服务器，它具有很好的可伸缩性（Scalability）、可靠性（Reliability）和可管理性(Manageability)。
>
> LVS是在内核中实现的一个功能，工作于传输层。LVS根据请求报文的目标IP和PORT根据挑选算法将其转发至后端主机集群中的某一个主机。在LVS框架中，提供了含有三种IP负载均衡技术的IP虚拟服务器软件分别是IPVS、Layer-7、KTCPVS。

IP负载均衡技术

VS/NAT

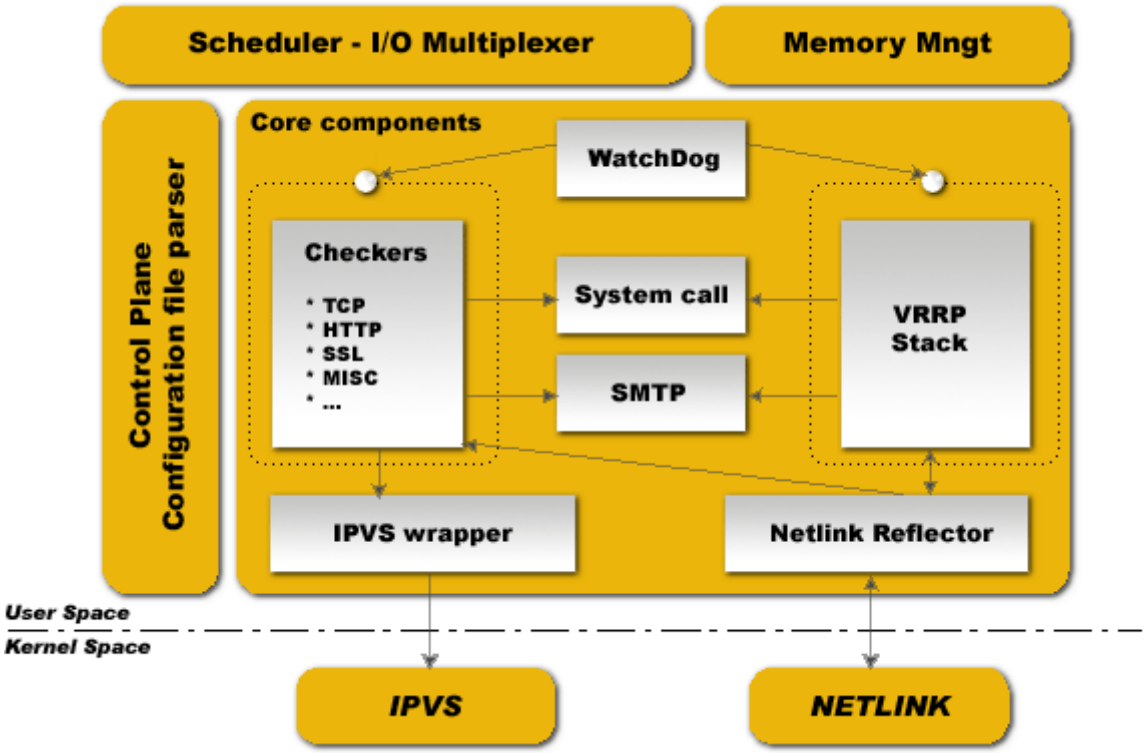
> Virtual Server via Network Address Translation(VS/NAT)。通过网络地址转换，调度器重写请求报文的目标地址，根据预设的调度算法，将请求分派给后端的真实服务器；真实服务器的响应报文通过调度器时，报文的源地址被重写，再返回给客户，完成整个负载

> 调度过程。地址转换模型，前端负载均衡器接受到客户端请求报文，将报文中的目标IP修改为真实服务器IP并转交给真实服务器。真实服务器将通过指向前端均衡器的网关将报文响应给前端负载均衡器，前端负载均衡器在将源IP修改后发送给客户端。真实服务器应使用私有网络地址，且网关要指向负载均衡服务器;真实服务器IP和负载均衡服务器IP必须在同一个网段，请求和响应报文都要经由负载均衡服务器转发;支持端口映射。



VS/DR

> Virtual Server via Direct Routing(VS/DR)。VS/DR通过改写请求报文的MAC地址，将请求发送到真实服务器，而真实服务器将响应直接返回给客户。同VS/TUN技术一样，VS/DR技术可极大地提高集群系统的伸缩性。这种方法没有IP隧道的开销，对集群中的真实服务器也没有必须支持IP隧道协议的要求，但是要求调度器与真实服务器都有一块网卡连在同一物理网段上。负载均衡调度器和真实服务器需要在同一个物理网络中，即需要可以通过广播找到MAC地址。前端路由器将目标IP为负载均衡器的报文发送给Director。RS的RIP可以使用私有网络地址，也可以使用公网IP地址。不支持端口映射。负载均衡器接受到外部请求后，通过ARP协议解析MAC地址进行封装然后转交给真实服务器。真实服务器收到报文后，拆除MAC帧，直接响应客户端。



VS/TNU

> 采用NAT技术时，由于请求和响应报文都必须经过调度器地址重写，当客户请求越来越多时，调度器的处理能力将成为瓶颈。为了解决这个问题，调度器把请求报文通过IP隧道转发至真实服务器，而真实服务器将响应直接返回给客户，所以调度器只处理请求报文。由于一般网络服务应答比请求报文大许多，采用 VS/TUN技术后，集群系统的最大吞吐量可以提高10倍。不修改请求报文的的IP首部，而是通过原有IP首部之外，再次封装一个IP，然后转交给真实服务器。真实服务器处理后直接响应客户端。可以跨互联网，通过路由。全部主机必须有公网地址。

VS/Full-NAT

> 负载均衡服务器接收到客户端的请求报文后，修改报文中的源IP和目标IP;真实服务器将报文返回负载均衡服务器再将报文中的源IP和目的IP修改，最后发给发出请求的客户端。

负载调度算法

> LVS调度算法(IPVS scheduler)，调度算法分为静态方法和动态方法。静态方法，仅仅根据算法本身进行调度，不受其他因素干扰；动态方法，根据算法及各个真实服务器的真实负载状况进行调度。

RR 轮叫调度(Round-Robin Scheduling)

> 以轮叫的方式依次调度到不同的服务器，每一台服务器接受的调度力度是相同的。这种调度算法假设每台真实服务器的性能都相同，都能够承受相同的连接数。但性能相差较大的服务器，不适应此种调度算法。另外，当请求时间跨度较大时，会造成负载不均衡的问题。

WR 加权轮叫调度(Weighted Round-Robin Scheduling)

> 在轮叫调度的基础上，引入另外一种机制，当某台真实服务器的权值为0时，表示不给此真实服务器调度请求，以便于对服务器进行维护和更换。

SH 源地址散列调度(Source Hashing Scheduling)

> 正好与目标地址散列调度算法相反，它根据请求的源IP地址，作为散列键（Hash Key）从静态分配的散列表找出对应的服务器，若该服务器是可用的且未超载，将请求发送到该服务器，否则返回空。它采用的散列函数与目标地址散列调度算法的相同。它的算法流程与目标地址散列调度算法的基本相似，除了将请求的目标IP地址换成请求的源IP地址。

DH 目标地址散列调度(Destination Hashing Scheduling)

> 根据目标IP地址的负载均衡，通过一个hash函数将请求的目标IP调度到服务器上。

LC 最小连接调度(Least-Connection Scheduling)

> 算法会把请求调度到当前连接数最少的真实服务器上去。

WLC 加权最小连接调度(Weighted Least-Connection Scheduling)

> 每台真实服务器都设置有权值，默认为1。在调度新连接时，调度器会尽可能的将已建立的连接数与权值成比例。

LBLC 基于局部性的最少链接(Locality-Based Least Connections Scheduling)

> 算法是针对请求报文的目标IP地址的负载均衡调度，目前主要用于Cache集群系统，因为在Cache集群中客户请求报文的目标IP地址是变化的。这里假设任何后端服务器都可以处理任一请求，算法的设计目标是在服务器的负载基本平衡情况下，将相同目标IP地址的请求调度到同一台服务器，来提高各台服务器的访问局部性和主存Cache命中率，从而整个集群系统的处理能力。

LBLCR 带复制的基于局部性最少链接(Locality-Based Least Connections with Replication Scheduling)

SED 最短预期延时调度(Shortest Expected Delay Scheduling)

NQ 不排队调度(Never Queue Scheduling)

ipvsadm用法

> ipvsadm是用户空间用来管理集群的一个工具。一个ipvs主机可以定义多个集群服务，一个集群上至少应有一个RS。

查看内核是否支持ipvs

```
modprobe -l | grep ipvs
```

管理集群服务

```
#添加或编辑集群服务
ipvsadm -A|E -t|u|f service-address [-s scheduler]
```

> `-A`添加一个集群服务
>
> `-E` 编辑一个集群服务
>
> `-t|u|f` 指定集群使用协议
>
> `service-adderss` 指定集群服务IP及端口，例如-t 192.168.1.150:80
>
> `-s`指定调度算法。例如-s rr

```
#删除集群服务
ipvsadm -D -t|u|f service-address
```

管理集群中真实服务器

```
#添或编辑集群中的真实服务器
ipvsadm -a|e -t|u|f service-address -r server-address [-g|i|m] [-w weight]
```

> `-r server-address` 指定真实服务器IP和端口，例如-r 192.168.2.2:80
>
> `-g|i|m` -g gatway DR模型，默认类型；-i TNU模型；-m NAT模型
>
> `[-w weight]` 权重

```
#删除真实服务器
ipvsadm -d -t|u|f service-address -r server-address
```

查看及清理集群规则

```
#清理所有集群规则
ipvsadm -C
```

```
#查看规则
ipvsadm -L|l [options]
```

> option：
>
> -n：基于数字显示地址和端口
> -c：显示当前ipvs建立的连接
> --stats：显示统计数据
> --rate：速率
> --exact：显示精确值，不作单位换算

```
#置零计数器
ipvsadm -Z [-t|u|f service-address]
```

保存及重载规则

```
#保存规则
/sbin/ipvsadm-save -n > /etc/sysconfig/ipvsadm
#重载
/sbin/ipvsadm-restore < /etc/sysconfig/ipvsadm
```

> 尽管ipvsadm的工作没有在用户空间启动任何进程，但ipvs依然以服务的形式存在，可以使用服务进行保存及重载。
>
> `systemctl restart ipvsadm`