讲师介绍

Hash　　QQ：805921455

从事Java软件研发近十年。

前新浪支付核心成员、

咪咕视讯(中国移动)项目经理、

对分布式架构、高性能编程有深入的研究。

明天，你一定会感谢今天奋力拼搏的自己

基于LVS高可用架构实现Nginx集群分流

分布式高并发—负载均衡

目录

课程安排

01

LVS简介

虚拟服务带来的世界

02

LVS负载策略分析

细数LVS负载策略

03

基于Keepalived实现LVS高可用

LVS可用性谁来保证？

04

LVS和Nginx异同分析

同样是负载均衡器LVS跟Nginx差别在哪里？

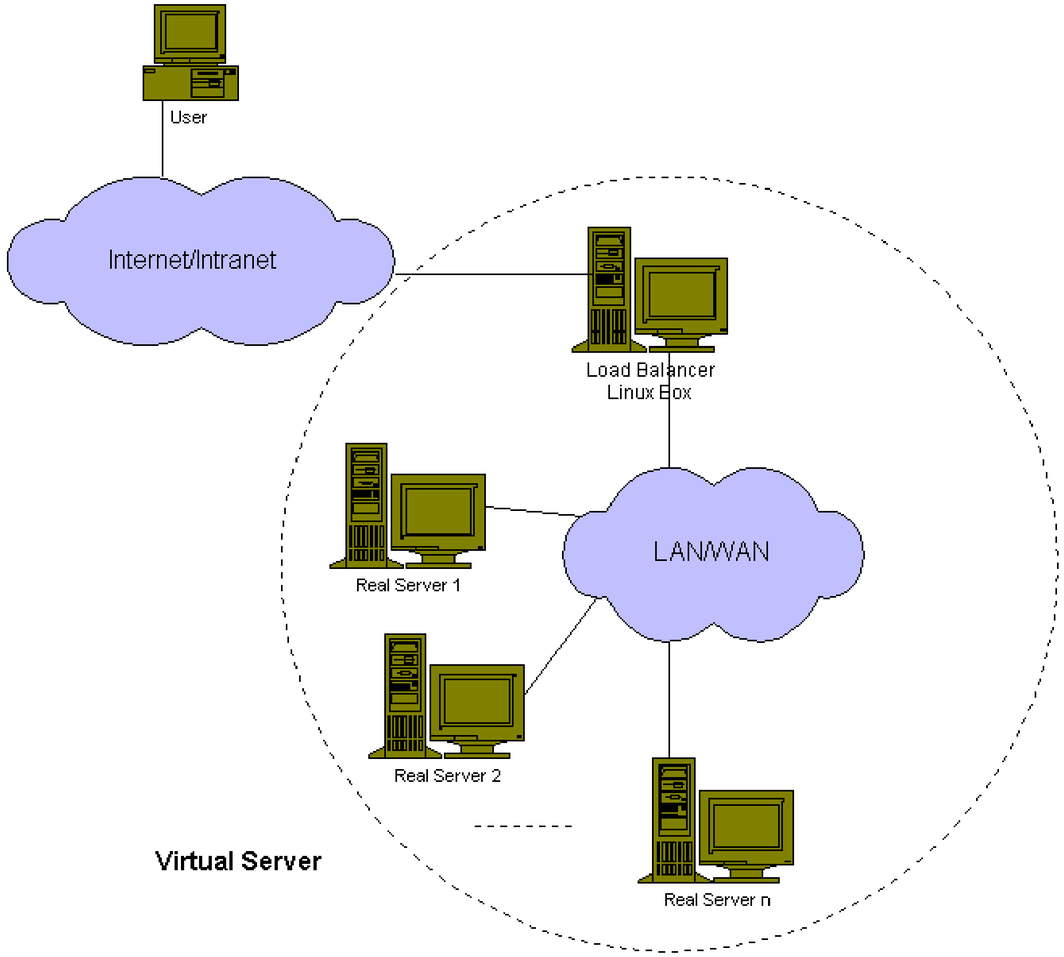
目录

LVS简介

什么是LVS

LVS（Linux Virtual Server）linux 虚拟服务，由中国人章文嵩创造和开发的linux内核项目。

一句话理解：将一堆的linux服务节点虚拟成一台服务器，对外提供相同的ip访问，对内实现各服务器的负载调度。



特点

- 有一个负载调度器—负载均衡
- 内部结构对客户端透明—封装
- 无感知的增删服务节点—可伸缩
- 检测服务健康状态—高可用

官网地址：http://www.linuxvirtualserver.org

为什么要用LVS？

互联网发展需求

要求web服务能具有可伸缩性、高可用性、可管理、性价比高这几个特性

服务器面临提升瓶颈

升级单台硬件设备显然代价非常大，服务集群化更具可扩展性和成本效益。

构建集群的方法

基于DNS，DNS将域名解析为服务器的不同IP地址来将请求分发到不同的服务器，实现集群负载

基于客户端，客户端需要值得服务集群信息。

基于调度程序，调度程序可以以精细的粒度调度请求（例如每个连接），以便在服务器之间实现更好的负载平衡。

基于IP

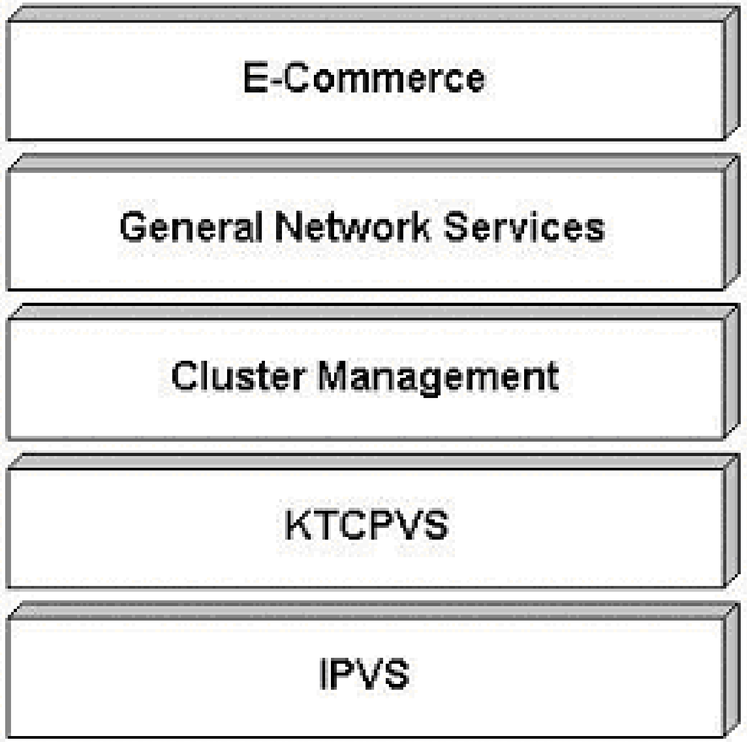
LVS属于基于IP级负载均衡，IP级的开销很小，服务器节点的最大数量可以达到25或高达100

LVS应用框架结构

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IPVS，基于内容请求分发的内核Layer-7交换机KTCPVS和集群管理软件。再上层则是集群管理、网络服务、电子商务应用。

IP级负载均衡—IPVS

应用级负载均衡—KTCPVS



Linux Virtual Server Framework

LVS如何工作？

第四层负载均衡IPVS

IPVS（IP Virtual Server）在Linux内核中实现传输层负载平衡，即所谓的第4层交换。在主机上运行的IPVS充当真实服务器集群前端的负载均衡器，它可以将对基于TCP / UDP的服务的请求定向到真实服务器，并使真实服务器的服务在虚拟服务上显示为虚拟服务。单个IP地址。LinuxDirector中共有三种IP负载均衡技术（数据包转发方法）。它们是通过**NAT**的虚拟服务器、通过**IP隧道**的虚拟服务器、通过**直接路由**的虚拟服务器。

第七层负载均衡KTCPVS

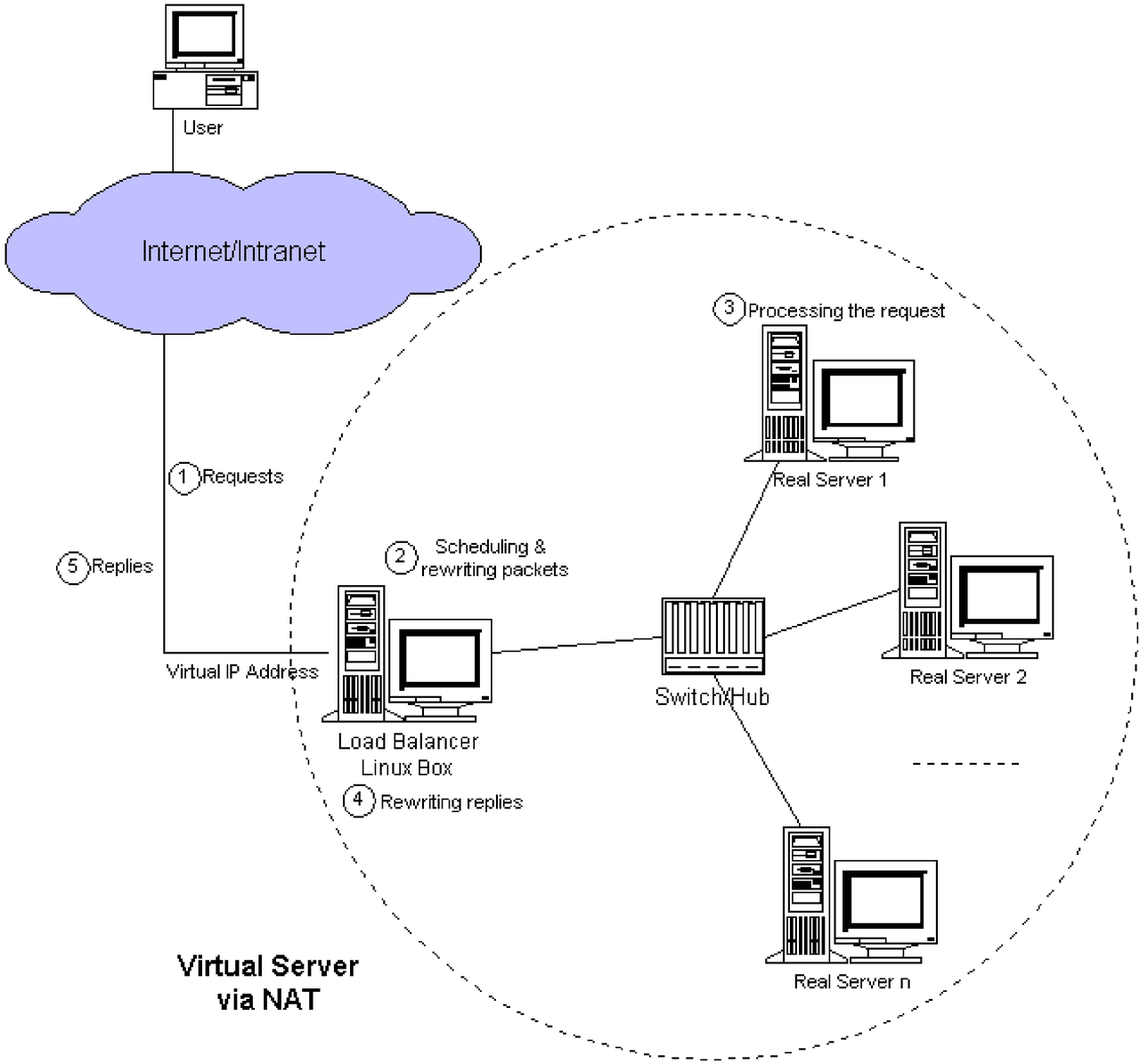
由于用户空间TCP Gateway的开销太大，我们提出在操作系统的内核中实现Layer-7交换方法，来避免用户空间与核心空间的切换和内存复制的开销。在Linux操作系统的内核中，我们实现了Layer-7交换，称之为KTCPVS（Kernel TCP Virtual Server）。目前，KTCPVS已经能对HTTP请求进行基于内容的调度，但它**还不很成熟**，在其调度算法和各种协议的功能支持等方面，有大量的工作需要做。

LVS如何工作？IPVS - NAT

NAT —（Network Address Translation网络地址转换）。将IP地址从一个组映射到另一个组的功能，**易于设置**。负载均衡器可能是服务器数量超过20的整个系统的瓶颈，因为请求数据包和响应数据包都需要由负载均衡器重写。

通过NAT的虚拟服务器的优点是真实服务器可以运行任何支持TCP/IP协议的操作系统，真实服务器可以使用私有Internet地址，并且负载均衡器只需要IP地址。

缺点是通过NAT的虚拟服务器的可扩展性是有限的。当服务器节点（通用PC服务器）的数量增加到大约20或更多时，负载平衡器可能是整个系统的瓶颈，因为请求包和响应包都需要由负载平衡器重写。



LVS如何工作？IPVS - IP TUN

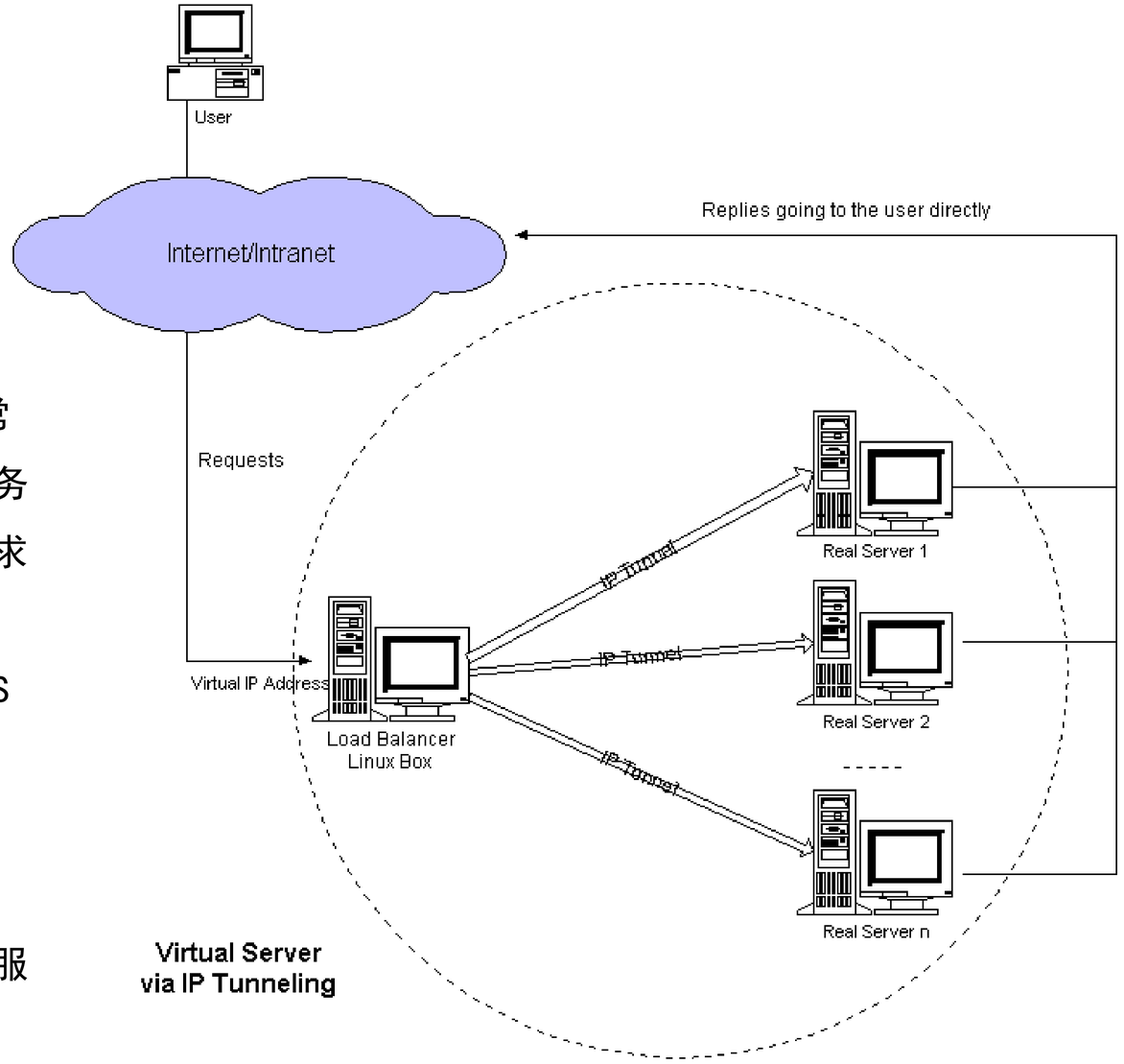
IP TUN — IP（tunnel 隧道）。将发往一个IP地址的数据报包装并重定向到另一个IP地址。是**最具扩展性**的。

TUN模式是通过ip隧道技术减轻lvs调度服务器的压力，许多Internet服务（例如WEB服务器）的请求包很短小，而应答包通常很大，负载均衡器只负责将请求包分发给物理服务器，而物理服务器将应答包直接发给用户。所以，负载均衡器能处理很巨大的请求量。

相比NAT性能要高的多，比DR模式的优点是不限制负载均衡器与RS在一个物理段上。

缺点，需要所有的服务器（lvs、RS）支持"IP Tunneling"(IP Encapsulation)协议

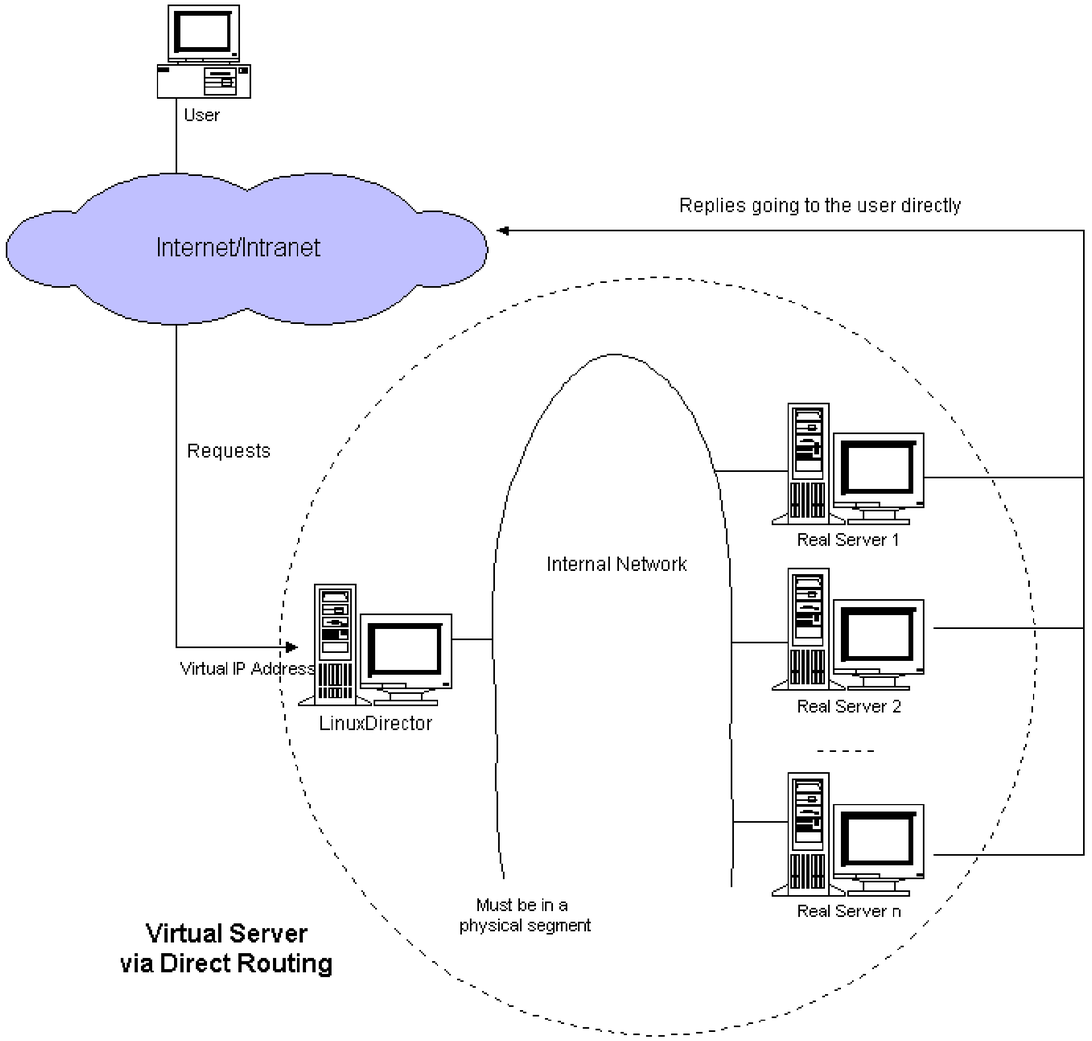
优点，由于服务器通过使用IP隧道相互连接，负载均衡器和真实服务器可以驻留在不同的LAN甚至WAN上。



LVS如何工作？IPVS - DR

DR —（direct route 直接路由）虚拟IP地址由真实服务器和负载均衡器共享。负载均衡器的接口也配置了虚拟IP地址，用于接受请求数据包，并直接将数据包路由到选定的服务器。所有真实服务器的**非arp别名**接口都配置了虚拟IP地址，或者将发往虚拟IP地址的数据包重定向到本地套接字，以便真实服务器可以在本地处理数据包。

具有**最佳性能**。VS / DR使用MAC欺骗技术，因此它要求负载均衡器的NIC和真实服务器的NIC之一必须位于同一IP网段和物理段中。



目录

LVS负载策略分析

LVS负载策略

轮询（Round Robin）

调度器通过"轮询"调度算法将外部请求按顺序轮流分配到集群中的真实服务器上，它均等地对待每一台服务器，而不管服务器上实际的连接数和系统负载。

加权轮询（Weighted Round Robin）

调度器通过"加权轮询"调度算法根据真实服务器的不同处理能力来调度访问请求。这样可以保证处理能力强的服务器处理更多的访问流量。调度器可以自动问询真实服务器的负载情况，并动态地调整其权值。

最少连接（Least Connections）

调度器通过"最少连接"调度算法动态地将网络请求调度到已建立的链接数最少的服务器上。如果集群系统的真实服务器具有相近的系统性能，采用"最小连接"调度算法可以较好地均衡负载。

加权最少链接（Weighted Least Connections）

在集群系统中的服务器性能差异较大的情况下，调度器采用"加权最少链接"调度算法优化负载均衡性能，具有较高权值的服务器将承受较大比例的活动连接负载。调度器可以自动问询真实服务器的负载情况，并动态地调整其权值。

LVS负载策略

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

针对目标IP地址的负载均衡，主要用于Cache集群系统。根据请求的目标IP地址找出该目标IP地址最近使用的服务器，若该服务器 是可用的且没有超载，将请求发送到该服务器；若服务器不存在，或者该服务器超载且有服务器处于一半的工作负载，则用“最少连接"的原则选出一个可用的服务器，将请求发送到该服务器。

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

与LBLC算法类似，不同的是它维护一个从目标IP地址到一组服务器的映射，LBLC算法维护一个从目标IP地址到一台服务器的映射。根据请求的目标IP地址找出该目标IP地址对应的服务器组，按“最少连接"原则从服务器组中选出一台服务器，若服务器没有超载，将请求发送到该服务器，若服务器超载；则按"最少连接"原则从这个集群中选出一台服务器，将该服务器加入到服务器组中，将请求发送到该服务器。同时，当该服务器组有一段时间没有被修改，将最忙的服务器从服务器组中删除，以降低复制的程度。

LVS负载策略

目标地址散列（Destination Hashing）

"目标地址散列"调度算法根据请求的目标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。

源地址散列（Source Hashing）

"源地址散列"调度算法根据请求的源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。

LVS实现Nginx负载均衡

参考《高可用Nginx集群安装搭建手册》

目录

基于Keepalived实现LVS高可用

什么是Keepalived?

Keepalived是一款为Linux系统、基于Linux的基础架构，提供**负载平衡**和**高可用性**的网络路由软件。由C语言编写，简单而强大。一句话理解，类似于heartbeat，用来防止单点故障的软件。

负载平衡，通过LVS的IPVS内核模块，实现第四层负载均衡。

高可用性，通过虚拟冗余路由协议（VRRP），实现了动态自适应地管理负载平衡的服务器池。

Keepalived可以独立使用，也可以一起使用提供弹性基础架构

Keepalived提供两个主要功能：

- LVS系统的健康检查
- 实现VRRPv2堆栈以处理负载均衡器故障转移

官网地址：https://www.keepalived.org

Keepalived的设计

```
# Keepalived有三个不同的进程，一个父进程，两个子进程
PID         111     Keepalived  <--父进程负责fork、监控子进程
            112     \_ Keepalived   <-- VRRP 子进程
            113     \_ Keepalived   <-- Healthchecking 子进程
```

子进程都有自己的I/O多路复用调度器，这样可以优化VRRP调度抖动，VRRP调度比健康检查器更敏感关键。这样拆分最小化了健康检查外部库的使用，并将其自身操作最小化到空闲主循环，以避免由自身引起的故障。

父进程监视框架称为看门狗。每个子进程打开一个接受unix域套接字，然后在守护进程引导时，父进程连接到那些unix域套接字并向子进程发送周期性（5s）hello数据包。如果父进程无法向远程连接的unix域套接字发送hello数据包，则只需重启子进程。

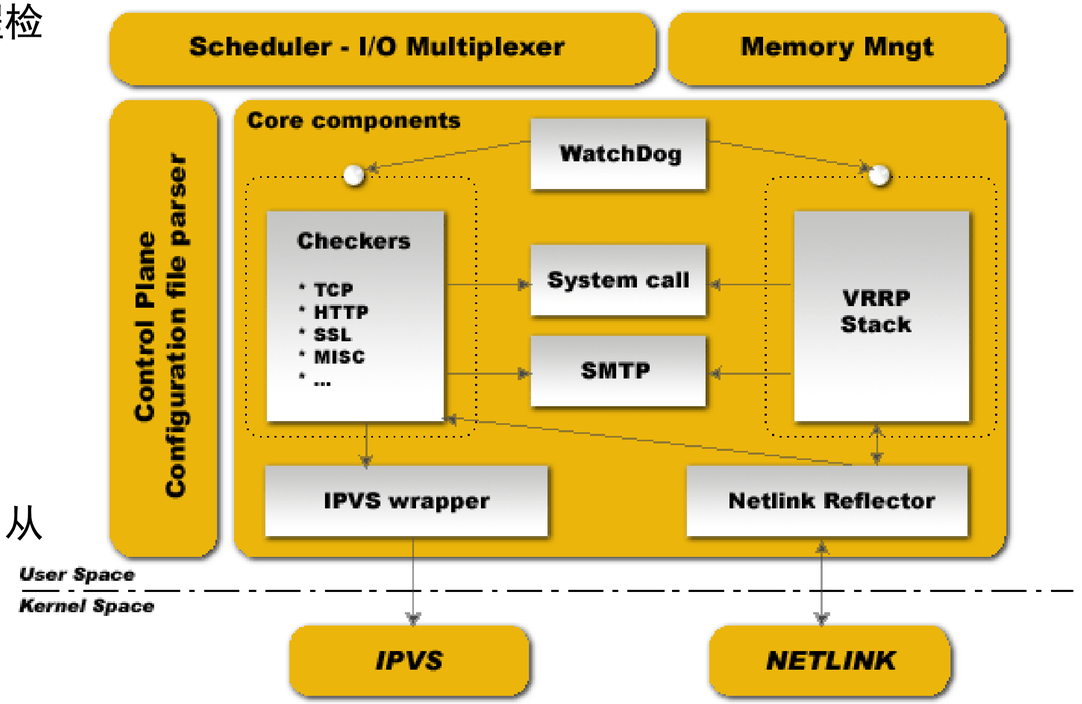
这样的设计确保Keepalived的健壮性和稳定性，另外两个好处：

1. 父进程发给远程连接的hello数据包是通过子进程I/O多路复用器调度程序完成的，可以检测子进程调度框架中的deadloop
2. 通过使用sysV信号来检测死亡的子进程

Keepalived的结构

Keepalived由控制平面、IO多路复用调度器、内存管理、核心组件构成。

核心组件中包含了前面提到的三个进程，父进程WatchDog程序、子进程检查程序、子进程VRRP协议栈程序。



WatchDog

派生、监控(VRRP & Healthchecking)子进程。

检查程序

负责realserver的运行状况检查。检查器测试realserver是否还活着，从而决定，自动从LVS拓扑中删除或添加realserver。

VRRP协议栈

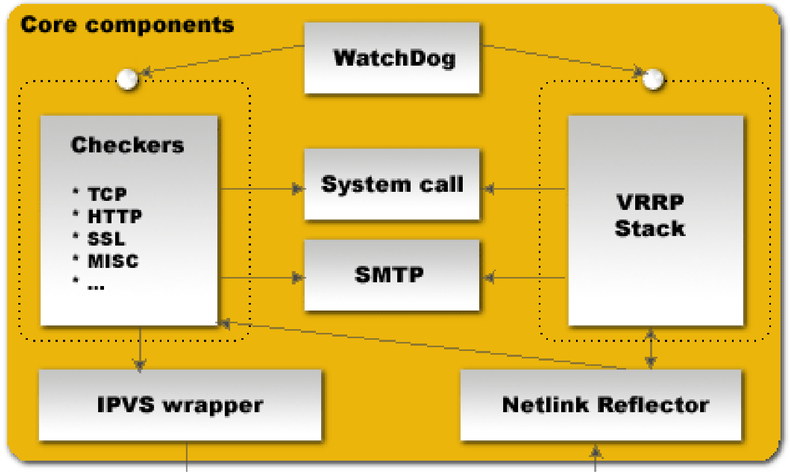
VRRP（Virtual Router Redundancy Protocol 虚拟路由器冗余协议）专门用于处理一组路由器组中路由器故障接管工作。

健康检查—Checkers

健康检查注册在全局调度框架中，有以下类型的运行状况检查：

TCP_CHECK

在第4层工作，使用无阻塞/超时的TCP连接，如果远程服务器未回复此请求（超时），则测试错误，并从服务器池中删除该服务器。



HTTP_GET

在第5层工作，对指定的URL执行HTTP GET。使用MD5算法对HTTP GET结果进行摘要，与期望值不匹配，则测试不通过，并将服务器从服务器池中删除。

SSL_GET

与HTTP_GET相同，但使用到远程Web服务器的SSL连接。

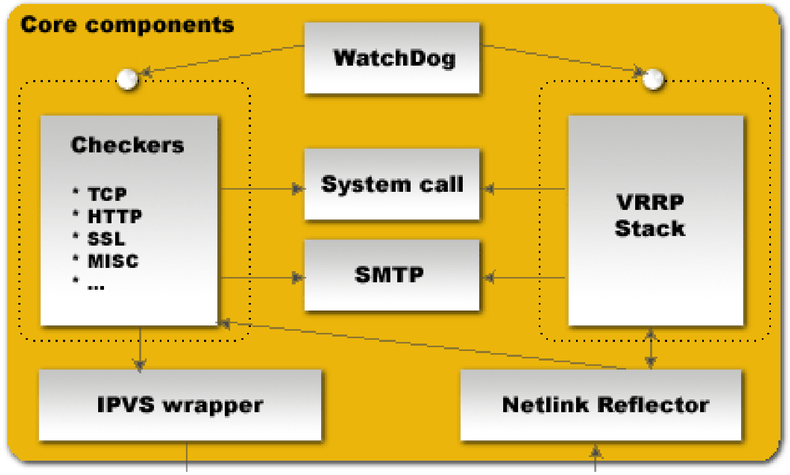
MISC_CHECK

通过System call调用自定义的脚本作为运行状况检查器运行，结果必须为0或1。脚用来测试内部应用程序的理想方法。

高可用特性—VRRP Stack

在设计网络的时候必须考虑冗余容灾，包括线路冗余，设备冗余等，防止网络存在单点故障，在路由器或三层交换机处实现冗余就显得尤为重要。

在网络里面VRRP协议就是来做这事的，Keepalived运用VRRP协议来实现高可用性。



VRRP是用来实现路由器冗余的协议，VRRP协议将多台路由器设备虚拟成一个设备，对外提供虚拟路由器IP。路由器组内部，路由器分为Master、Backup两种状态，Master是一台对外提供服务的IP的路由器，或者是通过算法选举产生的，其他的状态为BACKUP 。

MASTER路由器，实现针对虚拟路由器IP的各种网络功能，如ARP请求，ICMP，以及数据的转发等。

BACKUP状态路由器，只接收MASTER的VRRP状态信息，不执行对外的网络功能。

当Master失效时，BACKUP将接管原先MASTER的网络功能。

由于设计和健壮性的原因，这个模块已完全集成在Keepalived守护程序中

安装Keepalived

二进制安装

```
yum install keepalived
```

源码编译安装

```
# 安装依赖
sudo yum install -y curl gcc openssl-devel libnl3-devel net-snmp-devel
# 下载解压
sudo wget https://github.com/acassen/keepalived/archive/v2.0.18.tar.gz
tar -xvf keepalived.tar.gz
cd keepalived
# 编译安装，安装到/usr/local/keepalived目录
./configure --prefix=/usr/local/keepalived --sysconf=/etc
make && make install
```

通过Keepalived保证LVS高可用

参考《高可用Nginx集群安装搭建手册》

目录

LVS和Nginx异同分析

LVS和Nginx的差异

LVS后面的服务一定要接Nginx服务吗？

能直接接Tomcat吗？能直接接RPC服务吗？

为什么不这样做呢？

LVS基于网络模型的第四层负载均衡，即传输层。

Nginx基于网络模型的第七层负载均衡，及应用用层。

两个都可以用来做Web服务器的负载均衡

传输层的LVS比Nginx能够处理的并发更高，性能更强大

应用层的Nginx则比LVS的功能特性更加丰富、更加成熟稳定

谢谢观看