Lvs是我国章文嵩博士开发的一个开源项目，被默认编译成ip\_vs内核模块，lvs模块可以说是内核上的一个参数（也可说是一个程序），那么lvs的编译工具叫做ipvsadm，公司里通过lvs负载均衡技术能够实现一个高性能、高可用的服务器集群，它具有良好的可靠性、扩展性和可操作性，从而实现了低廉价的成本实现最优的服务性能，他的承受负载能力高、稳定、占用服务器资源小，传输效率非常高、并且高效，配合着keepalived软件可以支持节点健康检测机制，从负载上来说lvs可以支持10万+的并发连接，从稳定方面来说的话他是基于内核上的一个程序，所以运行起来是非常稳定，那么lvs从传输效率来说，他是基于IP+端口进行的数据转发，主要说明lvs工作在第四层

那么下面就具体的介绍一下四层负载和七层负载：

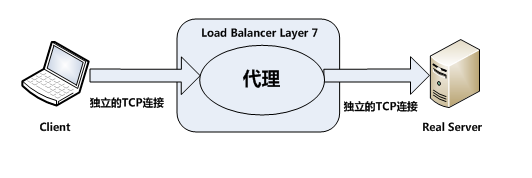
**四成负载：**

**四层负载是指ISO参考模型中的第四层，可以说是四层交换机，主要是通过IP层以及TCP/UDP层的流量来实现ip+端口的的负载均衡，那我们常见的四层负载均衡器有LVS、F5等，以常见的TCP应用为例，负载均衡器在收到客户发来的第一个SYN请求报文时，那么负载均衡器会根据自己设定的调度算法，通过修改后的目标IP+端口，直接将请求转发给后端的RS节点，这样的一个负载均衡请求算是完成了。从这个过程来看，一个TCP连接是客户端和服务器直接建立的，而负载均衡器只不过完成了一个类似路由器的转发动作。**



**七层负载：**

**在ISO参考模型中的最高层是第七层 ，应用层，七层负载均衡器也可以称之为七层交换机，此时负载均衡器可以支持多种应用协议，常见的有HTTP、FTP、SMTP等。七层负载均衡器可以根据报文中的内容，再配合负载均衡算法来选择后端服务器，这样的做法被称为“内容交换器”。相对于web服务器的负载均衡，七层负载均衡器不但可以根据“IP+端口”的方式进行负载分流，还可以根据网站的URL、访问域名、浏览器类别、语言等决定负载均衡的策略。常见的七层负载均衡器有HAproxy、Nginx等两款开源软件。以常见的TCP应用为例，由于负载均衡器需要获取客户发送的请求主体内容，需要先代替后端RS节点受理请求并建立连接，然后根据客户发到服务器的求取主体内容，通过负载均衡器中设置的调度算法来决定最终选择的内部服务器，七层负载均衡器在这种情况下类似于一个代理服务器，代理里客户的请求再次与后端节点建立tcp连接。**



那么lvs负载均衡器主要有三种模式，我们的（LVS/NAT）（LVS/DR）（LVS/TUN）模式那么在我工作的期间内并没有发现lvs做负载均衡器的时候使用的是NAT模式，我们公司用到的是DR模式，因为NAT模式作为网关来使用，客户的请求和响应都会经过LVS/NAT调度器，这样很容易实现高负载场景下的性能瓶颈，如果说非要使用这种模式的话我建议多端的节点最大不能超过20台，那么他的好处还是有的，他能够做到网络隔离的效果，使之内网节点更安全，节约了一定的ip地址，那我们公司为什么会选用LVS/DR模式呢，因为DR模式能够将请求和响应分开处理，负载均衡器只接受请求，而后端的节点直接响应与客户端，将极大地提高整个集群系统的吞吐量，（在这里说明一点如果客户访问的是一些字节量较大的图片或视频之类的，这些数据不会经过调度器了，而是直接由RS节点响应客户，如果说这里用到的是NAT模式，那么要求我们内网带宽一定要保证到位，企业里最好用千兆带宽，能够更好地传递数据）这就是DR模式的一点好处，

那么就具体的说明LVS/NAT模式及原理，首先说明一下为什么需要进行ip地址的转换，由于IPv4地址空间紧缺，我们网络中常使用的保留IP为：

（10.0.0.0/255.0.0.0、 172.16.0.0/255.128.0.0和192.168.0.0/255.255.0.0）

这些地址不在Internet上使用，而是专门为内部网络预留的，当内部网络中的主机要访问Internet或被Internet访问时，就需要LVS/NET采用网络地址转换，将内部地址转化为Internets上可用的外部地址, LVS/NAT模式的工作原理是报文头的（目标地址、源地址和端口等）被正确改写后，客户相信 它们连接一个IP地址

那么从原理上来说的话就是，客户的请求会通过互联网DNS服务器解析到公司负载均衡设备上的外网地址，通过上层路由器将数据转发到我的调度器上，来实现对请求负载均衡，我们这个调度器作为网关来使用是有两个物理网卡的，一个是外网网卡一个是内网网卡，为了安全会在外网网卡上做一次DNET策略（将公网DIP转换内网的VIP），通过网关调度器外网网卡的DNET策略将外网地址转换为内网的虚拟VIP， 通过虚拟VIP地址可以访问到网络服务，请求报文达到调度器，调度器根据连接的调度算法从一组真实服务器中选出一台服务器，将报文中的**目标VIP**改写成选定的**服务器的地址**，报文的**目标端口**改写成选定**服务器的**相应**端口**，最后将修改后的报文发送给选出的服务器。当真实服务器的响应报文经过调度器时，调度器将报文的源地址和源端口改为VIP和调度器自身相应的端口，然后响应的数据通过外网网卡做SNET策略将内网的VIP转换为外网ip，再把响应的报文数据发给用户

**DR模式的原理：**

客户端会通过互联网DNS服务器对自身发送的域名进行解析并找到网站的公网IP地址，最后找到我站服务的上层路由器，ARP请求包通过路由器形成数据转发，数据到达交换机之后，所有的服务器都会接受客户发来的ARP请求包，看是否是找自己的，这些请求到达自己的网关时会通过自身的端口映射转换成VIP，所有的服务的VIP都是一致的，那么为什么lvs调度器会处理客户发送的ARP请求包呢，因为lvs调度器开启的是DR模式，并且VIP地址是网卡的子IP是绑定在网卡上的并能够形成通信，会直接处理ARP请求包，而其他节点的VIP是绑定在内核上的并无法进行通信，是无法处理ARP请求的，最后调度器将自身的VIP以及Mac封装到ARP请求包内，以单播的形式原路返回给客户端，最后与调度器建立了连接。 建立连接完成之后客户端发送的请求数据包通过路由器ARP缓存表会自动找到lvs调度器，调度器会根据自己的调度算法去指定一台服务处理客户的请求，这时候数据包的头部的目标Mac地址改为指定服务器的Mac地址，通过交换机找到相应的节点服务，服务器处理请求之后会直接响应于客户端，这时候原IP是我的VIP地址，目标IP是客户端IP， 原Mac是此节点Mac，目标Mac是客户的Mac，通过这些数据包头部可直接响应客户端。

如果说这时候用到的是LVS/DR模式我们需要做arp混淆问题，这时候我们做完lvs策略规则之后，需要在RS节点上的内核配置单上添加一些arp请求不受理，不响应参数的优化，在/etc/sysctl下添加.....（具体参数）

接下来了解一下lvs的静态调度算法以及动态的调度算法，

静态的RR 轮询算法，调度器收到的请求会按照顺序将请求分发至每台节点。

WRR 加权轮询算法，按照处理能力不同进行加权处理。

动态的调度算法;

LC 最少连接数算法，调度器收到请求之后会根据这种动态算法，将请求分发给建立连接最少的主机，

WLC 加权最少连接数，比如说最少连接着 客户的请求并且有一台机子做了加权处理，那么将请求分发给此节点。

LBLC基于本地的最少连接调度算法，