**LVS的负载调度算法**

在内核中的连接调度算法上，IPVS已实现了以下八种调度算法：  
一、轮叫调度（Round-Robin Scheduling） ----rr  
轮叫调度（Round Robin Scheduling）算法就是以轮叫的方式依次将请求调度不同的服务器，即每次调度执行i = (i + 1) mod n，并选出第i台服务器。算法的优点是其简洁性，它无需记录当前所有连接的状态，所以它是一种无状态调度。

二、加权轮叫调度（Weighted Round-Robin Scheduling） ----wrr  
加权轮叫调度（Weighted Round-Robin Scheduling）算法可以解决服务器间性能不一的情况，它用相应的权值表示服务器的处理性能，服务器的缺省权值为1。假设服务器A的权值为1，B的权值为2，则表示服务器B的处理性能是A的两倍。加权轮叫调度算法是按权值的高低和轮叫方式分配请求到各服务器。权值高的服务器先收到的连接，权值高的服务器比权值低的服务器处理更多的连接，相同权值的服务器处理相同数目的连接数。

三、最小连接调度（Least-Connection Scheduling） ---lc  
最小连接调度（Least-Connection Scheduling）算法是把新的连接请求分配到当前连接数最小的服务器。最小连接调度是一种动态调度算法，它通过服务器当前所活跃的连接数来估计服务器的负载情况。调度器需要记录各个服务器已建立连接的数目，当一个请求被调度到某台服务器，其连接数加1；当连接中止或超时，其连接数减一。

四、加权最小连接调度（Weighted Least-Connection Scheduling）---wlc  
加权最小连接调度（Weighted Least-Connection Scheduling）算法是最小连接调度的超集，各个服务器用相应的权值表示其处理性能。服务器的缺省权值为1，系统管理员可以动态地设置服务器的权值。加权最小连接调度在调度新连接时尽可能使服务器的已建立连接数和其权值成比例。

五、基于局部性的最少链接（Locality-Based Least Connections Scheduling） --lblc  
基于局部性的最少链接调度（Locality-Based Least Connections Scheduling，以下简称为LBLC）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。LBLC调度算法先根据请求的目标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其一半的工作负载，则用“最少链接”的原则选出一个可用的服务器，将请求发送到该服务器。

六、带复制的基于局部性最少链接（Locality-Based Least Connections with Replication Scheduling）--lblcr  
带复制的基于局部性最少链接调度（Locality-Based Least Connections with Replication Scheduling，以下简称为LBLCR）算法也是针对目标IP地址的负载均衡，目前主要用于Cache集群系统。它与LBLC算法的不同之处是它要维护从一个目标IP地址到一组服务器的映射，而LBLC算法维护从一个目标IP地址到一台服务器的映射。对于一个“热门”站点的服务请求，一台Cache 服务器可能会忙不过来处理这些请求。这时，LBLC调度算法会从所有的Cache服务器中按“最小连接”原则选出一台Cache服务器，映射该“热门”站点到这台Cache服务器，很快这台Cache服务器也会超载，就会重复上述过程选出新的Cache服务器。这样，可能会导致该“热门”站点的映像会出现在所有的Cache服务器上，降低了Cache服务器的使用效率。LBLCR调度算法将“热门”站点映射到一组Cache服务器（服务器集合），当该“热门”站点的请求负载增加时，会增加集合里的Cache服务器，来处理不断增长的负载；当该“热门”站点的请求负载降低时，会减少集合里的Cache服务器数目。这样，该“热门”站点的映像不太可能出现在所有的Cache服务器上，从而提供Cache集群系统的使用效率。LBLCR算法先根据请求的目标IP地址找出该目标IP地址对应的服务器组；按“最小连接”原则从该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“最小连接”原则从整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。

七、目标地址散列调度（Destination Hashing Scheduling） ---dh  
目标地址散列调度（Destination Hashing Scheduling）算法也是针对目标IP地址的负载均衡，但它是一种静态映射算法，通过一个散列（Hash）函数将一个目标IP地址映射到一台服务器。目标地址散列调度算法先根据请求的目标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。

八、源地址散列调度（Source Hashing Scheduling）---sh  
源地址散列调度（Source Hashing Scheduling）算法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，所以这里不一一叙述。在实际应用中，源地址散列调度和目标地址散列调度可以结合使用在防火墙集群中，它们可以保证整个系统的唯一出入口