链滴

【网络】复习网络七层模型

作者：yuanhenglizhen
原文链接：https://ld246.com/article/1539499913439
来源网站：链滴

百度定义：
<blockquote>
七层模型，亦称 OSI（Open System Interconnection）参考模型，是参考模型是国际标准化组（ISO）制定的一个用于计算机或通信系统间互联的标准体系。
它是一个七层的、抽象的模型体，不仅包括一系列抽象的术语或概念，也包括具体的协议。
</blockquote>
<h3 id="OSI七层模型">OSI 七层模型</h3>

1、OSI 七层网络模型称为开发式系统互联网参考模型，是一个逻辑上的定义和规范；
2、把网络从逻辑上分为七层，每一层都有相应的物理设备
3、OSI 七层网络模型是一种框架式的设计方法，最主要的功能就是帮助不同类型的主机实现数据传；
4、最大的优点就是将服务、接口和协议三个概念明确的区分起来
5、复杂且不实用；经常使用的是 TCP/IP 四层模型。
各部分及功能：
1、应用层：针对你特定应用的协议
2、表示层：设备固定的数据格式和网络标准数据格式之间的转化
3、会话层：通信管理，负责建立和单开通信连接，管理传输层 以下分层
4、传输层：管理两个节点之间的数据传递。负责可靠传输
5、网络层：地址管理和路由选择
6、数据链路层：互联设备之间传送和识别数据帧
7、物理层：界定连接器和网线之间的规格
<h3 id="TCP-IP四-五-层模型">TCP/IP 四（五）层模型</h3>
每一层都呼叫它的下一层提供的网络来完成自己的需求。（如果是四层模型数据链路层和物理层一层）
1、物理层：负责光电信号传递方式。集线器工作在物理层。以太网协议。
2、数据链路层：负责设备之间的数据帧的传输和识别。交换机工作在数据链路层。例如网卡设备的动，帧同步，冲突检测，数据差错校验等工作。
3、网络层：负责地址管理和路由选择。路由器工作在网络层。
4、传输层：负责两台主机之间的数据传输。
5、应用层：负责应用程序之间的沟通。网络编程主要针对的就是应用层。
传输层和网络层的封装在操作系统完成。应用层的封装在应用程序中完成。
数据链路层和物理层的封装在设备驱动程序与网络接口中完成。
关系：

一般而言：

对于一台主机，它的操作系统内核实现了传输层到物理层的内容
对于一台路由器，它实现了从网络层到物理层
对于一台交换机，它实现了由数据链路层到物理层
对于集线器，他只实现了物理层。

补充：
<h4 id="四层-七层负载均衡的区别">四层、七层负载均衡的区别</h4>
==========

所谓四层就是基于 IP+ 端口的负载均衡；七层就是基于 URL 等应用层信息的负载均衡；同理，有基于 MAC 地址的二层负载均衡和基于 IP 地址的三层负载均衡。 换句换说，二层负载均衡会通过个虚拟 MAC 地址接收请求，然后再分配到真实的 MAC 地址；三层负载均衡会通过一个虚拟 IP 地

接收请求，然后再分配到真实的 IP 地址；四层通过虚拟 IP+ 端口接收请求，然后再分配到真实的服器；七层通过虚拟的 URL 或主机名接收请求，然后再分配到真实的服务器。

所谓的四到七层负载均衡，就是在对后台的服务器进行负载均衡时，依据四层的信息或七层的信来决定怎么样转发流量。比如四层的负载均衡，就是通过发布三层的 IP 地址（VIP），然后加四层的口号，来决定哪些流量需要做负载均衡，对需要处理的流量进行 NAT 处理，转发至后台服务器，并录下这个 TCP 或者 UDP 的流量是由哪台服务器处理的，后续这个连接的所有流量都同样转发到同一服务器处理。七层的负载均衡，就是在四层的基础上（没有四层是绝对不可能有七层的），再考虑应用层的特征，比如同一个 Web 服务器的负载均衡，除了根据 VIP 加 80 端口辨是否需要处理的流量，还可根据七层的 URL、浏览器类别、语言来决定是否要进行负载均衡。举个例，如果你的 Web 服务器分成两组，一组是中文语言的，一组是英文语言的，那么七层负载均衡就可当用户来访问你的域名时，自动辨别用户语言，然后选择对应的语言服务器组进行负载均衡处理。</>

负载均衡器通常称为四层交换机或七层交换机。四层交换机主要分析 IP 层及 TCP/UDP 层，实四层流量负载均衡。七层交换机除了支持四层负载均衡以外，还有分析应用层的信息，如 HTTP 协议RI 或 Cookie 信息。

负载均衡分为 L4 switch（四层交换），即在 OSI 第 4 层工作，就是 TCP 层啦。此种 Load Balace 不理解应用协议（如 HTTP/FTP/MySQL 等等）。例子：LVS，F5。

另一种叫做 L7 switch（七层交换），OSI 的最高层，应用层。此时，该 Load Balancer 能理解用协议。例子： haproxy，MySQL Proxy。

注意：上面的很多 Load Balancer 既可以做四层交换，也可以做七层交换。
<h5 id="二-区别">二、区别</h5>

技术原理上
** 所谓四层负载均衡**，也就是主要通过报文中的目标地址和端口，再加上负载均衡设备设置的服器选择方式，决定最终选择的内部服务器。
以常见的 TCP 为例，负载均衡设备在接收到第一个来自客户端的 SYN 请求时，即通过上述方式选择个最佳的服务器，并对报文中目标 IP 地址进行修改(改为后端服务器 IP），直接转发给该服务器。TCP的连接建立，即三次握手是客户端和服务器直接建立的，负载均衡设备只是起到一个类似路由器的转动作。在某些部署情况下，为保证服务器回包可以正确返回给负载均衡设备，在转发报文的同时可能会对报文原来的源地址进行修改。

<img src="https://ld246.com/images/img-loading.svg" alt="" data-src="https://b3logfilecom/e/0828bc8396624893833eb69f95d0461a.webp?imageView2/2/interlace/1/format/jpg">/p>

所谓七层负载均衡**，也称为“内容交换”，也就是主要通过报文中的真正有意义的应用层内容再加上负载均衡设备设置的服务器选择方式，决定最终选择的内部服务器。
以常见的 TCP 为例，负载均衡设备如果要根据真正的应用层内容再选择服务器，只能先代理最的服务器和客户端建立连接(三次握手)后，才可能接受到客户端发送的真正应用层内容的报文，然后根据该报文中的特定字段，再加上负载均衡设备设置的服务器选择方式，决定最终选择的内部服务器

负载均衡设备在这种情况下，更类似于一个代理服务器。负载均衡和前端的客户端以及后端的服务器分别建立 TCP 连接。所以从这个技术原理上来看，七层负载均衡明显的对负载均衡设备的要求更高处理七层的能力也必然会低于四层模式的部署方式。
<ol start="2">
应用场景
七层应用负载的好处，是使得整个网络更智能化。例如访问一个网站的用户流量，可以通过七层的方，将对图片类的请求转发到特定的图片服务器并可以使用缓存技术；将对文字类的请求可以转发到特的文字服务器并可以使用压缩技术。当然这只是七层应用的一个小案例，从技术原理上，这种方式可对客户端的请求和服务器的响应进行任意意义上的修改，极大的提升了应用系统在网络层的灵活性。多在后台，例如 Nginx 或者 Apache 上部署的功能可以前移到负载均衡设备上，例如客户请求中的 Hader 重写，服务器响应中的关键字过滤或者内容插入等功能。
另外一个常常被提到功能就是安全性。网络中最常见的 SYN Flood 攻击，即黑客控制众多源客户端使用虚假 IP 地址对同一目标发送 SYN 攻击，通常这种攻击会大量发送 SYN 报文，耗尽服务器上的关资源，以达到 Denial of Service(DoS)的目的。从技术原理上也可以看出，四层模式这些 SYN 攻击都会被转发到后端的服务器上；而七层模式下这些 SYN 攻击自然在负载均衡设备上就止，不会影响后台服务器的正常运营。另外负载均衡设备可以在七层层面设定多种策略，过滤特定报，例如 SQL Injection 等应用层面的特定攻击手段，从应用层面进一步提高系统整体安全。
现在的 7 层负载均衡，主要还是着重于应用 HTTP 协议，所以其应用范围主要是众多的网站或者内部息平台等基于 B/S 开发的系统。 4 层负载均衡则对应其他 TCP 应用，例如基于 C/S 开发的 ERP 等统。

<h5 id="三-Nginx-LVS及HAProxy负载均衡软件的优缺点">三、Nginx、LVS 及 HAProxy 负载均软件的优缺点</h5>
负载均衡 （Load Balancing） 建立在现有网络结构之上，它提供了一种廉价有效透明的方法扩网络设备和服务器的带宽、增加吞吐量、加强网络数据处理能力，同时能够提高网络的灵活性和可用。
Nginx/LVS/HAProxy 是目前使用最广泛的三种负载均衡软件。
一般对负载均衡的使用是随着网站规模的提升根据不同的阶段来使用不同的技术。具体的应用需还得具体分析，如果是中小型的 Web 应用，比如日 PV 小于 1000 万，用 Nginx 就完全可以了；如机器不少，可以用 DNS 轮询，LVS 所耗费的机器还是比较多的；大型网站或重要的服务，且服务器较多时，可以考虑用 LVS。
一种是通过硬件来进行，常见的硬件有比较昂贵的 F5 和 Array 等商用的负载均衡器，它的优点是有专业的维护团队来对这些服务进行维护、缺点就是花销太大，所以对于规模较小的网络服务来说时还没有需要使用；另外一种就是类似于 Nginx/LVS/HAProxy 的基于 Linux 的开源免费的负载均衡件，这些都是通过软件级别来实现，所以费用非常低廉。
目前关于网站架构一般比较合理流行的架构方案：Web 前端采用 Nginx/HAProxy+ Keepalived 作负载均衡器；后端采用 MySQL 数据库一主多从和读写分离，采用 LVS+Keepalived 的架构。当然根据项目具体需求制定方案。
下面说说各自的特点和适用场合。
Nginx 的优点是：

工作在网络的 7 层之上，可以针对 http 应用做一些分流的策略，比如针对域名、目录结构，它正则规则比 HAProxy 更为强大和灵活，这也是它目前广泛流行的主要原因之一，Nginx 单凭这点可用的场合就远多于 LVS 了。

Nginx 对网络稳定性的依赖非常小，理论上能 ping 通就就能进行负载功能，这个也是它的优势一；相反 LVS 对网络稳定性依赖比较大。

Nginx 安装和配置比较简单，测试起来比较方便，它基本能把错误用日志打印出来。LVS 的配置测试就要花比较长的时间了，LVS 对网络依赖比较大。

可以承担高负载压力且稳定，在硬件不差的情况下一般能支撑几万次的并发量，负载度比 LVS 对小些。

Nginx 可以通过端口检测到服务器内部的故障，比如根据服务器处理网页返回的状态码、超时等，并且会把返回错误的请求重新提交到另一个节点，不过其中缺点就是不支持 url 来检测。比如用户在上传一个文件，而处理该上传的节点刚好在上传过程中出现故障，Nginx 会把上传切到另一台服务重新处理，而 LVS 就直接断掉了，如果是上传一个很大的文件或者很重要的文件的话，用户可能会因而不满。

Nginx 不仅仅是一款优秀的负载均衡器/反向代理软件，它同时也是功能强大的 Web 应用服务。LNMP 也是近几年非常流行的 web 架构，在高流量的环境中稳定性也很好。

Nginx 现在作为 Web 反向加速缓存越来越成熟了，速度比传统的 Squid 服务器更快，可以考用其作为反向代理加速器。

Nginx 可作为中层反向代理使用，这一层面 Nginx 基本上无对手，唯一可以对比 Nginx 的就只 lighttpd 了，不过 lighttpd 目前还没有做到 Nginx 完全的功能，配置也不那么清晰易读，社区资料远远没 Nginx 活跃。

Nginx 也可作为静态网页和图片服务器，这方面的性能也无对手。还有 Nginx 社区非常活跃，三方模块也很多。

Nginx 的缺点是：

Nginx 仅能支持 http、https 和 Email 协议，这样就在适用范围上面小些，这个是它的缺点。</>

对后端服务器的健康检查，只支持通过端口来检测，不支持通过 url 来检测。不支持 Session 的接保持，但能通过 ip_hash 来解决。

LVS：使用 Linux 内核集群实现一个高性能、高可用的负载均衡服务器，它具有很好的可伸缩性 Scalability)、可靠性（Reliability)和可管理性（Manageability)。
LVS 的优点是：

抗负载能力强、是工作在网络 4 层之上仅作分发之用，没有流量的产生，这个特点也决定了它在载均衡软件里的性能最强的，对内存和 cpu 资源消耗比较低。

配置性比较低，这是一个缺点也是一个优点，因为没有可太多配置的东西，所以并不需要太多接，大大减少了人为出错的几率。

工作稳定，因为其本身抗负载能力很强，自身有完整的双机热备方案，如 LVS+Keepalived。</

>

无流量，LVS 只分发请求，而流量并不从它本身出去，这点保证了均衡器 IO 的性能不会受到大量的影响。

应用范围比较广，因为 LVS 工作在 4 层，所以它几乎可以对所有应用做负载均衡，包括 http、据库、在线聊天室等等。

LVS 的缺点是：

软件本身不支持正则表达式处理，不能做动静分离；而现在许多网站在这方面都有较强的需求，个是 Nginx/HAProxy+Keepalived 的优势所在。

如果是网站应用比较庞大的话，LVS/DR+Keepalived 实施起来就比较复杂了，特别后面有 Windows Server 的机器的话，如果实施及配置还有维护过程就比较复杂了，相对而言，Nginx/HAProxy+eepalived 就简单多了。

HAProxy 的特点是：

HAProxy 也是支持虚拟主机的。

HAProxy 的优点能够补充 Nginx 的一些缺点，比如支持 Session 的保持，Cookie 的导；同时支持通过获取指定的 url 来检测后端服务器的状态。

HAProxy 跟 LVS 类似，本身就只是一款负载均衡软件；单纯从效率上来讲 HAProxy 会比 Nginx 有更出色的负载均衡速度，在并发处理上也是优于 Nginx 的。

HAProxy 支持 TCP 协议的负载均衡转发，可以对 MySQL 读进行负载均，对后端的 MySQL 节点进行检测和负载均衡，大家可以用 LVS+Keepalived 对 MySQL 主从做负载衡。

HAProxy 负载均衡策略非常多，HAProxy 的负载均衡算法现在具体有如下 8 种：
① roundrobin，表示简单的轮询，这个不多说，这个是负载均衡基本都具备的；
② static-rr，表示根据权重，建议关注；
③ leastconn，表示最少连接者先处理，建议关注；
④ source，表示根据请求源 IP，这个跟 Nginx 的 IP_hash 机制类似，我们用其作为解决 session 问的一种方法，建议关注；
⑤ ri，表示根据请求的 URI；
⑥ rl_param，表示根据请求的 URl 参数’balance url_param’ requires an URL parameter nam；
⑦ hdr(name)，表示根据 HTTP 请求头来锁定每一次 HTTP 请求；
⑧ rdp-cookie(name)，表示根据据 cookie(name)来锁定并哈希每一次 TCP 请求。

Nginx 和 LVS 对比的总结：

Nginx 工作在网络的 7 层，所以它可以针对 http 应用本身来做分流策略，比如针对域名、目录构等，相比之下 LVS 并不具备这样的功能，所以 Nginx 单凭这点可利用的场合就远多于 LVS 了；但ginx 有用的这些功能使其可调整度要高于 LVS，所以经常要去触碰触碰，触碰多了，人为出问题的率也就会大。

Nginx 对网络稳定性的依赖较小，理论上只要 ping 得通，网页访问正常，Nginx 就能连得通，是 Nginx 的一大优势！Nginx 同时还能区分内外网，如果是同时拥有内外网的节点，就相当于单机有了备份线路；LVS 就比较依赖于网络环境，目前来看服务器在同一网段内并且 LVS 使用 direct 方分流，效果较能得到保证。另外注意，LVS 需要向托管商至少申请多一个 ip 来做 Visual IP，貌似是能用本身的 IP 来做 VIP 的。要做好 LVS 管理员，确实得跟进学习很多有关网络通信方面的知识，就再是一个 HTTP 那么简单了。

Nginx 安装和配置比较简单，测试起来也很方便，因为它基本能把错误用日志打印出来。LVS 的装和配置、测试就要花比较长的时间了；LVS 对网络依赖比较大，很多时候不能配置成功都是因为网问题而不是配置问题，出了问题要解决也相应的会麻烦得多。

Nginx 也同样能承受很高负载且稳定，但负载度和稳定度差 LVS 还有几个等级：Nginx 处理所流量所以受限于机器 IO 和配置；本身的 bug 也还是难以避免的。

Nginx 可以检测到服务器内部的故障，比如根据服务器处理网页返回的状态码、超时等等，并且把返回错误的请求重新提交到另一个节点。目前 LVS 中 ldirectd 也能支持针对服务器内部的情况来控，但 LVS 的原理使其不能重发请求。比如用户正在上传一个文件，而处理该上传的节点刚好在上传程中出现故障，Nginx 会把上传切到另一台服务器重新处理，而 LVS 就直接断掉了，如果是上传一很大的文件或者很重要的文件的话，用户可能会因此而恼火。

Nginx 对请求的异步处理可以帮助节点服务器减轻负载，假如使用 apache 直接对外服务，那么现很多的窄带链接时 apache 服务器将会占用大 量内存而不能释放，使用多一个 Nginx 做 apache 理的话，这些窄带链接会被 Nginx 挡住，apache 上就不会堆积过多的请求，这样就减少了相当多的源占用。这点使用 squid 也有相同的作用，即使 squid 本身配置为不缓存，对 apache 还是有很大助的。

Nginx 能支持 http、https 和 email（email 的功能比较少用），LVS 所支持的应用在这点上会 Nginx 更多。在使用上，一般最前端所采取的策略应是 LVS，也就是 DNS 的指向应为 LVS 均衡器，VS 的优点令它非常适合做这个任务。重要的 ip 地址，最好交由 LVS 托管，比如数据库的 ip、webservice 服务器的 ip 等等，这些 ip 地址随着时间推移，使用面会越来越大，如果更换 ip 则故障会接踵至。所以将这些重要 ip 交给 LVS 托管是最为稳妥的，这样做的唯一缺点是需要的 VIP 数量会比较多Nginx 可作为 LVS 节点机器使用，一是可以利用 Nginx 的功能，二是可以利用 Nginx 的性能。当然一层面也可以直接使用 squid，squid 的功能方面就比 Nginx 弱不少了，性能上也有所逊色于 Ngin。Nginx 也可作为中层代理使用，这一层面 Nginx 基本上无对手，唯一可以撼动 Nginx 的就只有 lighttpd 了，不过 lighttpd 目前还没有能做到 Nginx 完全的功能，配置也不那么清晰易读。另外，中层理的 IP 也是重要的，所以中层代理也拥有一个 VIP 和 LVS 是最完美的方案了。具体的应用还得具体析，如果是比较小的网站（日 PV 小于 1000 万），用 Nginx 就完全可以了，如果机器也不少，可以 DNS 轮询，LVS 所耗费的机器还是比较多的；大型网站或者重要的服务，机器不发愁的时候，要多考虑利用 LVS。

现在对网络负载均衡的使用是随着网站规模的提升根据不同的阶段来使用不同的技术
第一阶段：利用 Nginx 或 HAProxy 进行单点的负载均衡，这一阶段服务器规模刚脱离开单服务、单数据库的模式，需要一定的负载均衡，但是仍然规模较小没有专业的维护团队来进行维护，也没需要进行大规模的网站部署。这样利用 Nginx 或 HAproxy 就是第一选择，此时这些东西上手快，置容易，在七层之上利用 HTTP 协议就可以。这时是第一选择。
第二阶段：随着网络服务进一步扩大，这时单点的 Nginx 已经不能满足，这时使用 LVS 或者商用 Array 就是首要选择，Nginx 此时就作为 LVS 或者 Array 的节点来使用，具体 LVS 或 Array 的是选是根据公司规模和预算来选择，Array 的应用交付功能非常强大，本人在某项目中使用过，性价比也高于 F5，商用首选，但是一般来说这阶段相关人才跟不上业务的提升，所以购买商业负载均衡已经为了必经之路。
第三阶段：这时网络服务已经成为主流产品，此时随着公司知名度也进一步扩展，相关人才的能以及数量也随之提升，这时无论从开发适合自身产品的定制，以及降低成本来讲开源的 LVS，已经成首选，这时 LVS 会成为主流。
最终形成比较理想的基本架构为：Array/LVS — Nginx/Haproxy — Squid/Varnish — AppSer er。