学生应知信息知识

Web服务器知识手册（四）

狄登峰 主编

目　　录

关于双连接的负载均衡

包级别的 TCP/UDP 负载均衡和 NAT(Network Address Translate)不能并存。

这是什么意思呢？简单来说，如果你有动态 IP 的连接，一个使用私有 IP 地址的局域网，通过 NAT(Network Address Translate)上网， 现在觉得速度不够，想再加一条使到某一 Internet 主机的速度加倍（现在中国大陆的典型情况），对不起，你就死了这条心吧，这是不可能的。

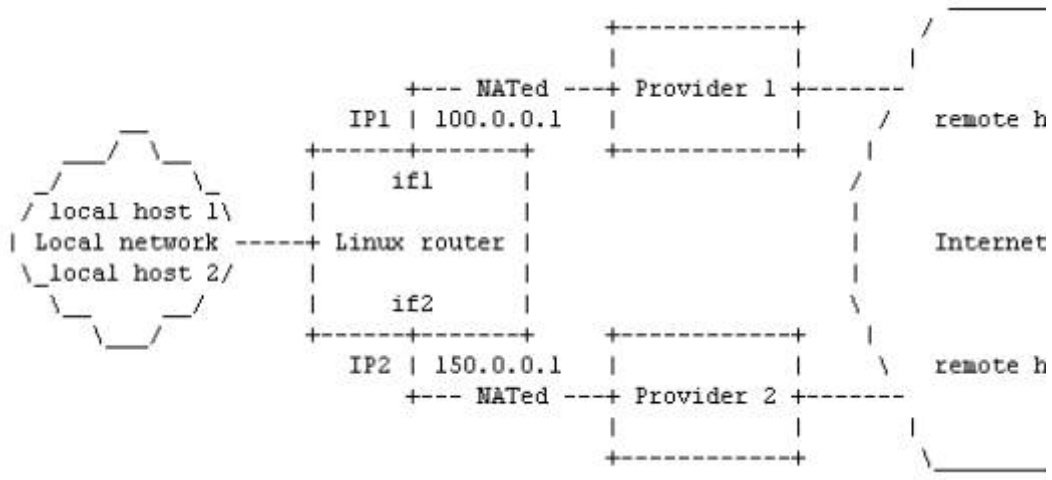
这里所说的连接包括 Ethernet，PPPoE,PPPoA，PPP，以及 SLIP 等等。并且，这是 TCP/UDP 协议的内在机制造成的，和软/硬件无关。任何架构在 TCP/UDP 之上的应用，不论硬件还是软件，皆不能绕过此限制。

注意，如果你的局域网使用公用 IP（public IP）而不需要 NAT，则不在此列。

如果你想知道原因或者质疑我的结论:)，请继续往下读。

本文假设

①、你已经阅读过 Linux Advanced Routing & Traffic Control HOWTO，特别是 4.2 小节的 Routing for multiple uplinks/providers，4.2.1 小节 Split access 和 4.2.2 小节 Load balancing。此文（以下按惯例简称 lartc）非常清楚的说明了连接级别的 TCP/UDP 负载均衡。



上面这幅示意图即摘自 lartc**，官方站点是** http://lartc.org/**。为行文方便，列在这里并做轻微改动，特此说明。**

②、为便于理解，以下以常见的 Linux **下** ADSL PPPoE **拨号上网为例，但如前所述，讨论适用于其它各种情况。用数对**(IP address, port)**来表示一个连结的端点。假定**

Host IP Port

---

local host 1 192.168.0.15000

Linux routeif1 100.0.0.16000

Linux routeif2 150.0.0.17000

remote host1 200.0.0.180

那么连接级别和包级别的负载均衡究竟有何不同？

1、连结级别的负载均衡

Linux 2.4 **内核支持连结级别的负载均衡。为理解这个问题，这里首先简述** Linux **的静态路由机制。当一部** Linux **主机收到一个需要转发的** IP **包时，它首先检查**

Cache 中是否有相关的路由信息：如果有的话，会直接使用；否则再查找路由表。Cache 中一段时间不使用的路由信息会被丢弃。如果设置了 MultiPath 路由，则选择哪一个路由是随机的，但随后的 IP 包都会走这条路由，直到 Cache 中的路由信息被丢弃。

比如，当 LAN 上的主机 local host 1 向 remote host 1 发起一次连结的时候，Linux router 在 Cache 中找不到有关 remote host 1 的路由信息，因此转去查找路由表，当它发现有一个 MultiPath 路由的时候，会随机选择一个路由，比方说 if1，并把此路由加入 Cache；下一个从 local hsot 1 到 remote host 1 的 IP 包到达时，Linux router 直接在 Cache 中找到路由信息然后直接转发，也就是说，所有目的地为 remote host 1 的 IP 包都会经由 if1 转发直到 Cache 中相关路由信息失效并被丢弃为止而不会经过 if2，即使 if2 完全空闲也是如此。

因此，和 remote host 1 之间的通信并不会从附加的第二条连接中获益。其实际效果就是你用 Realplay 看网络电影时效果没有改善，当然心理作用除外:)

如果与此同时 local host 1 有发起一条到 remote host 2 的连接，则有可能这一次会使用 if2。其实际效果就是你用 Realplay 看网络电影时用 flashget 开 8 个并发线程到一个 FTP 站下载一个 600M Redhat ISO 文件，两者不会互相影响。

那么，下载 ISO 文件的同时 localhost 2 发起一条到 remote host 2 的连接，会出现怎样的情况？是简单地走 if1？又或别有蹊径？请读者思考。

2、包级别的负载均衡

现实中这种一边忙一边无所事事的情况在所多有，

毋庸赘述:)。那么如何改善呢？很自然的想到，我们可以把同一条 TCP/IP 连接中的 IP 包同时从两条上行连接中发出，左右开弓，不就可以了吗？实际上，我们在 PPP Multilink 就是这么干的，不过现在没有服务器的支持，要靠自己。这是 Internet，因此已经有人想到这一点并且编写了内核补丁供下载，请搜索 equalize_2.4.18.patch。

需要特别提醒你的是，请确保该补丁的日期是 Fri Mar 22 2002，而不是有问题的 Thu Mar 21 2002 版本。另外，根据我的经验，这个 patch 也可以用于 2.4.19 内核。

重新编译了内核以后，你可以使用 equalize 关键字了。如果你的 LAN 使用公有 IP 地址，那么恭喜你，你现在可以在双倍速度下用 Realplay 看网络电影了。但是正所谓人生不如意事常八九，如果你使用的是私有 IP 地址配合以 NAT 上网，让我们来看看当 local host 1 对 remote host 1 发起一次连结时会发生什么。

local host 1 第一次从（192.168.0.1，5000）向 remote host 1 (200.0.0.1, 80) 发起一条连接，现在第一个 IP 包（SYN 包）已经到达 Linux router。既然是第一次，Cache 中当然没有有关路由信息，Linux router 查找路由表，发现带 equalize 标记的 MultiPath 路由，因此随机决定如何转发此 IP 包 -- 好，这次茫茫中命里注定从 if2 走:) 并且雁过留声，在 Cache 中添加一条到 remote host 1 的路由信息-- 既然 local host 1 的 IP 是私有的，NAT 起作用，转换源地址为 (150.0.0.1,7000) 后发出，没问题，remote host 1 收到并向 (150.0.0.1,7000) 发回 SYN+ACK 包表示接受连结。Linux router if2 收到返回的 SYN+ACK 包，早有准备，NAT 再次转换目的

地址为 (192.168.0.1, 5000)。

第一回合。到目前为止一切 OK。

紧接着 local host 1 (192.168.0.1, 5000) 按规矩再向 (200.0.0.1, 80) 发送 ACK 包表示确认。此 ACK 包又来到 Linux router 这个岔路口，它将走向何方呢？既然设定了 equalize，Linux 不会根据先前生成的路由表 Cache 中的记录（就是由 SYN 包生成的）来决定走法，相反，它从 Cache 中删除该记录，然后重新查找路由表。这导致又一次重新选择路由，可能是 if1，也可能是 if2。

如果又是 if2，很好，remote host 1 收到，握手完成，连接建立。

若是 if1，又当如何呢？很不幸，if1 将该包源地址经 NAT 转换为 (100.0.0.1,6000)，然后丢上 Internet，一路平安到达 remote host 1。remote host 1 正在苦等从 if2 (也就是 150.0.0.1,7000) 来的 ACK 包，没想却收到一个从(100.0.0.1,6000) 来的 ACK 包，它作何打算呢？难道说，反正是 ACK 包，马马虎虎将就着用用吗？很明显，这不是个好主意。其次，即便意欲如此，remote host 1 如何判断这个 ACK 包和哪一个 SYN 包关联呢？因此，唯一合理的措施就是：忽略它，当没看见好了。

丢弃此来路不明的 ACK 包后，remote host 1 继续等待中.....结果可想而知：不能同步，连结建立失败。这里 TCP/IP 的重传机制并不能解决问题，remote host 1 并不认为这是一个正常的 TCP 包，虽然它是一个正常的 IP 包。

刚才提到如果走的是 if2，非常幸运，连接可以建立。同样，以后 remote host 1 会收到所有数据包，不论是从 if1 还是 if2 来的，只是从 if1 来的 TCP 包一概

按例办理，全部丢弃。好在 local host 1 会重传这些被丢弃的包，总有机会走到 if2 而被收到，因此连结还可以勉强维持，只是比较慢而已；当然，在某种极端情况下，也可能中途断掉。

所以，equalize 负载均衡和 NAT 在一起工作，可能会导致建立连结失败或者较缓慢的连结。换句话说，1+1 < 1。

最后有一个好消息，不是大好，是小好：ICMP 协议在此种情况下可以工作,也许你可以试着用它干点什么。当然这也意味着一个使用动态 IP 对你发动 ICMP 攻击的黑客有双倍的可用带宽:)

注 1：2.4 内核中 MultiPath 设定的具体的位置在 Networking options -> TCP/IP networking -> IP: advanced router -> IP: equal cost multipath，.config 文件中的对应行为 CONFIG_IP_ROUTE_MULTIPATH

注 2：iproute2 的手册明确指出，仅当内核打过补丁的情况下 equalize 关键字才能工作，原文"equalize only works if the kernel is patched." iproute2-ss010824, p26.

注 3：本文所说的随机并不是真正意义的随机，但我们假定考察真正随机的情况。

注 4：对于如何处理所谓"来路不明的 ACK 包"，可能是实现相关的，还请了解的朋友说一说。

附录 A

关于 lartc 中 split access 和 load-balancing 设置的一点说明：

在 load-balancing 小节中提到"如果你已经如前所述设置了 split access 的话 load-balancing 真的不是太

难"。而在 split access 小节中就给出了一堆命令，T1 T2 P1 P2 IF1 IF2 什么的。很多人不明白这些命令到底有何作用。其实这些命令是说，到 P1_NET 去的走 if1，到 P2_NET 去的走 if2，大家各行其道。那么，我到 P1_NET 去的走 if2，到 P2_NET 去的走 if1，可不可以呢？理论上也是可以的，但是这里有个问题。如果 P1_NET 和 P2_NET 分属两家不同的 ISP，则从 if2 到 P1_NET 就要走远路经过交换中心，而这可能会造成瓶颈。

于从 P1_NET 和 P2_NET 回来的包，已经超出了我们的控制范围，只能等它到 IF1 或 IF2。如果出于某种原因路由不正常，没有收到回音，那我们能做的也只是承认失败而已。

因此这段命令想要防止舍近求远，如此而已。

外这段命令似乎有点问题。我已经在 lartc 邮件列表中请求帮助了，希望能尽快有一个满意的答复。

在搞清楚这一点后，如果读者的 P1_NET 和 P2_NET 和我一样其实是同一家 ISP 的网络，这段准备功夫是完全可以省去的。

在局域网中实现 Web 共享

从网上邻居一步步打开其他用户计算机的共享文件，总感觉有些麻烦。有没有其他的共享文件的方式呢？其实我们可以在局域网中通过 Web 方式轻松实现资源共享。

一、设置 Web 共享文件夹

设置 Web 共享时，需要 Web 服务器的支持，如果你的局域网中没有架设 Web 服务器，可以先安装 Windows XP/2000 中自带的 IIS，依次点击“开始→设置→控制面板→添加/删除程序→添加/删除 Windows 组件”，选择安装“Internet 信息服务（IIS）”。

IIS 安装后，在查看文件夹属性（右键单击文件夹选择“属性”）时，我们可以发现属性对话框中多出一个“Web 共享”标签项。选中需要共享的文件夹，在其属性对话框内的“Web 共享”标签项中，勾选“共享文件夹”，随后弹出一个“编辑别名”对话框（图 1），在“别名”项中输入该文件夹在局域网中显示的名称，在“访问权限”中系统提供了读取、写入、脚本资源访问、浏览目录四个多选项，建议勾选“读取”项，其他选项可根据需要选择。在“应用程序权限”中选择“无”，最后点击“确定”退出。

图 1

以后我们就可以在该局域网内的其他计算机上，通

过在 IE 地址栏中输入要访问计算机的 IP 地址和共享文件夹名称（如 http：//192.168.0.5/ My Music），并回车后，就能看到该文件夹中的所有内容（图 2）。我们还可以单击该文件夹中标有“ <Dir> ”字样的链接，访问下一级文件夹中的内容，如果要想将某一个文件保存到自己的计算机中，我们可以单击右键选择“ 另存为 ”即可。

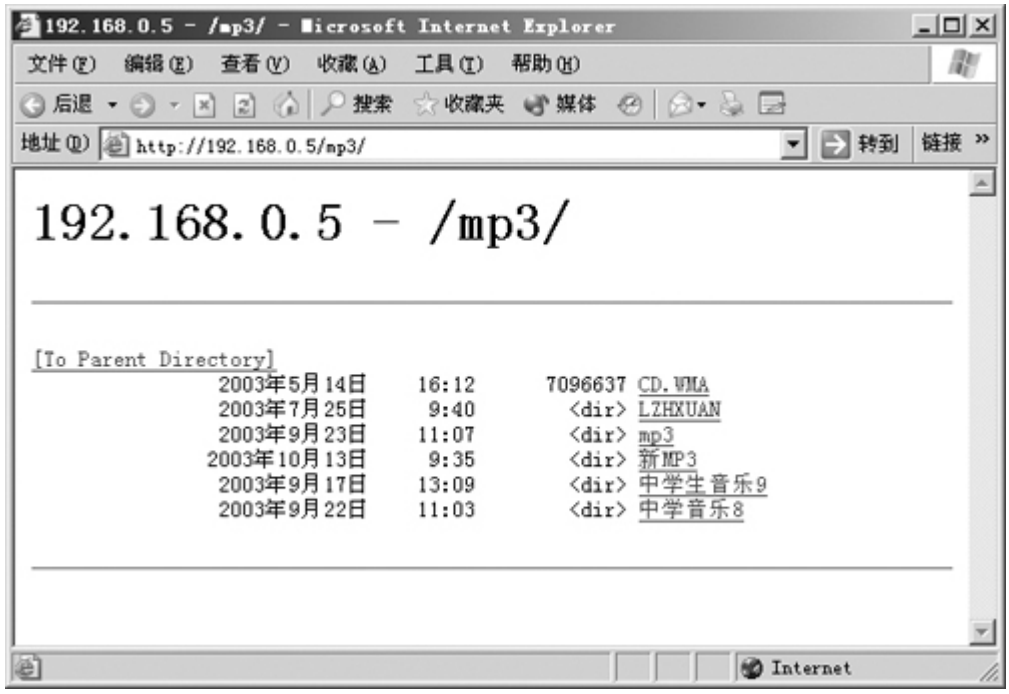


图 2

二、打造个性的 Web 共享文件夹

用上面的方法设置的 Web 共享文件夹界面可能不太美观，如果要想在 IE 中像浏览网页内容那样浏览文件夹中的内容那该多好。其实我们只要对 Web 共享文件夹稍做编辑即可实现这一效果。

首先将需要 Web 共享的文件夹按照上面的方法设置好，随后启动 FrontPage 制作一个网页文件，再将要共享文件夹中的所有文件都链接到该网页中。网页制作

好后，将页面文件保存到共享文件夹中，并将它命名为“index.htm”文件。以后我们再从其他计算机的 IE 地址栏中输入该计算机的 IP 地址和这个共享文件夹名称时，IE 便可以直接读取该文件夹下的 index.htm 文件，进入制作好的网页界面，单击相应的链接即可访问需要的文件（图 3）。

图 3

Apache 服务器配置技巧

本文简要介绍了十几个 Apache 的配置技巧：

1、如何设 置请求等待时间

在 httpd.conf 里面设置：

TimeOut n

其中 n 为整数，单位是秒。

设置这个 TimeOut 适用于三种情况：

2、如何接收一个 get 请求的总时间

接收一个 post 和 put 请求的 TCP 包之间的时间

TCP 包传输中的响应（ack）时间间隔

3、如何使得 apache 监听在特定的端口

修改 httpd.conf 里面关于 Listen 的选项，例如：

Listen 8000

是使 apache 监听在 8000 端口

而如果要同时指定监听端口和监听地址,可以使用：

Listen 192.170.2.1:80

Listen 192.170.2.5:8000

这样就使得 apache 同时监听在 192.170.2.1 的 80 端口和 192.170.2.5 的 8000 端口。

当然也可以在 httpd.conf 里面设置：

Port 80

这样来实现类似的效果。

4、如何设置 apache 的最大空闲进程数

修改 httpd.conf，在里面设置：

MaxSpareServers n

其中 n 是一个整数。这样当空闲进程超过 n 的时候，apache 主进程会杀掉多余的空闲进程而保持空闲进程在 n，节省了系统资源。如果在一个 apache 非常繁忙的站点调节这个参数才是必要的，但是在任何时候把这个参数调到很大都不是一个好主意。

同时也可以设置：

MinSpareServers n

来限制最少空闲进程数目来加快反应速度。

5、apache 如何设置启动时的子服务进程个数

在 httpd.conf 里面设置：

StartServers 5

这样启动 apache 后就有 5 个空闲子进程等待接受请求。

也可以参考 MinSpareServers 和 MaxSpareServers 设置。

6、如何在 apache 中设置每个连接的最大请求数

在 httpd.conf 里面设置：

MaxKeepAliveRequests 100

这样就能保证在一个连接中，如果同时请求数达到 100 就不再响应这个连接的新请求，保证了系统资源不会被某个连接大量占用。但是在实际配置中要求尽量把这个数值调高来获得较高的系统性能。

7、如何在 apache 中设置 session 的持续时间

在 apache1.2 以上的版本中，可以在 httpd.conf 里面设置：

KeepAlive on

KeepAliveTimeout 15

这样就能限制每个 session 的保持时间是 15 秒。session 的使用可以使得很多请求都可以通过同一个 tcp 连接来发送，节约了网络资源和系统资源。

8、如何使得 apache 对客户端进行域名验证

可以在 httpd.conf 里面设置：

HostnameLookups on|off|double

如果是使用 on，那么只有进行一次反查，如果用

double，那么进行反查之后还要进行一次正向解析，只有两次的结果互相符合才行，而 off 就是不进行域名验证。

如果为了安全，建议使用 double；为了加快访问速度，建议使用 off。

9、如何使得 apache 只监听在特定的 ip

修改 httpd.conf，在里面使用

BindAddress 192.168.0.1

这样就能使得 apache 只监听外界对 192.168.0.1 的 http 请求。如果使用：

BindAddress *

就表明 apache 监听所有网络接口上的 http 请求。

当然用防火墙也可以实现。

10、apache 中如何限制 http 请求的消息主体的大小

在 httpd.conf 里面设置：

LimitRequestBody n

n 是整数，单位是 byte。

cgi 脚本一般把表单里面内容作为消息的主体提交给服务器处理，所以现在消息主体的大小在使用 cgi 的时候很有用。比如使用 cgi 来上传文件，如果有设置：

LimitRequestBody 102400

那么上传文件超过 100k 的时候就会报错。

11、如何修改 apache 的文档根目录

修改 httpd.conf 里面的 DocumentRoot 选项到指定的目录，比如：

DocumentRoot /www/htdocs

这样 http://localhost/index.html 就是对应 /www/htdocs/index.html

12、如何修改 apache 的最大连接数

在 httpd.conf 中设置：

MaxClients n

n 是整数，表示最大连接数，取值范围在 1 和 256 之间，如果要让 apache 支持更多的连接数，那么需要修改源码中的 httpd.h 文件，把定义的 HARD_SERVER_LIMIT 值改大然后再编译。

13、如何使每个用户有独立的 cgi-bin 目录

有两种可选择的方法：

(1)在 Apache 配置文件里面关于 public_html 的设置后面加入下面的属性：

ScriptAliasMatch ^/~([^/]*)/cgi-bin/(.*) /home/$1/cgi-bin/$2

(2)在 Apache 配置文件里面关于 public_html 的设置里面加入下面的属性：

<Directory /home/*/public_html/cgi-bin>

Options ExecCGI

SetHandler cgi-script

</Directory>

14、如何调整 Apache 的最大进程数

Apache 允许为请求开的最大进程数是 256,MaxClients 的限制是 256.如果用户多了，用户就只能看到 Waiting for reply....然后等到下一个可用进程的出现。这个最大数，是 Apache 的程序决定的--它的 NT 版可以有 1024，但 Unix 版只有 256，你可以在 src/include/httpd.h 中看到：

#ifndef HARD_SERVER_LIMIT

#ifdef WIN32

#define HARD_SERVER_LIMIT 1024

#else

#define HARD_SERVER_LIMIT 256

#endif

#endif

你可以把它调到 1024**，然后再编译你的系统。**

15**、如何屏蔽来自某个** Internet **地址的用户访问** Apache **服务器**

可以使用 deny **和** allow **来限制访问，比如要禁止** 202.202.202.xx **网络的用户访问：**

<Directory /www/htdocs>

order deny,allow

deny from 202.202.202.0/24

</Directory>

16**、如何在日志里面记录** apache **浏览器和引用信息**

你需要把 mod_log_config **编译到你的** Apache **服务器中，然后使用下面类似的配置：**

CustomLog logs/access_log "%h %l %u %t "%r" %s %b "%{Referer}i" "%{User-Agent}i""

17**、如何修改** Apache **返回的头部信息**

问题分析：当客户端连接到 Apache **服务器的时候，** Apache **一般会返回服务器版本、非缺省模块等信息，例如：**

Server: Apache/1.3.26 (Unix) mod_perl/1.26

解决：

你可以在 Apache **的配置文件里面作如下设置让它返回的关于服务器的信息减少到最少：**

ServerTokens Prod