LVS 手册

修改历史

时间	说明	修改人
2006-11-29	创建文档	*****
2006-12-6	修改文档	**********
2007-1-11	增加受攻击的应对措施	*********

系统有限公司

欢迎点击这里的链接进入精彩的Linux公社 网站

Linux公社（www.Linuxidc.com）于2006年9月25日注册并开通网站，Linux现在已经成为一种广受关注和支持的一种操作系统，IDC是互联网数据中心，LinuxIDC就是关于Linux的数据中心。

Linux公社是专业的Linux系统门户网站，实时发布最新Linux资讯，包括Linux、Ubuntu、Fedora、RedHat、红旗Linux、Linux教程、Linux认证、SUSE Linux、Android、Oracle、Hadoop、CentOS、MySQL、Apache、Nginx、Tomcat、Python、Java、C语言、OpenStack、集群等技术。

Linux公社（LinuxIDC.com）设置了有一定影响力的Linux专题栏目。

Linux公社 主站网址：www.linuxidc.com 旗下网站：www.linuxidc.net

包括：**Ubuntu 专题** **Fedora 专题** **Android 专题** **Oracle 专题** **Hadoop 专题** **RedHat 专题** **SUSE 专题** **红旗 Linux 专题** **CentOS 专题**

Linux 公社微信公众号：linuxidc_com

目 录

第一章 IPVS 负载均衡技术

1.1 LVS 集群的通用结构

LVS 集群采用 IP 负载均衡技术，属于 IP 层的交换（L4），具有很好的吞吐率。调度器分析客户端到服务器的 IP 报头信息，将请求均衡地转移到不同的服务器上执行，且调度器自动屏蔽掉服务器的故障，从而将一组服务器构成一个高性能的、高可用的虚拟服务器，LVS 集群系统的通用结构如图 1.1 所示，主要包含四大部分：

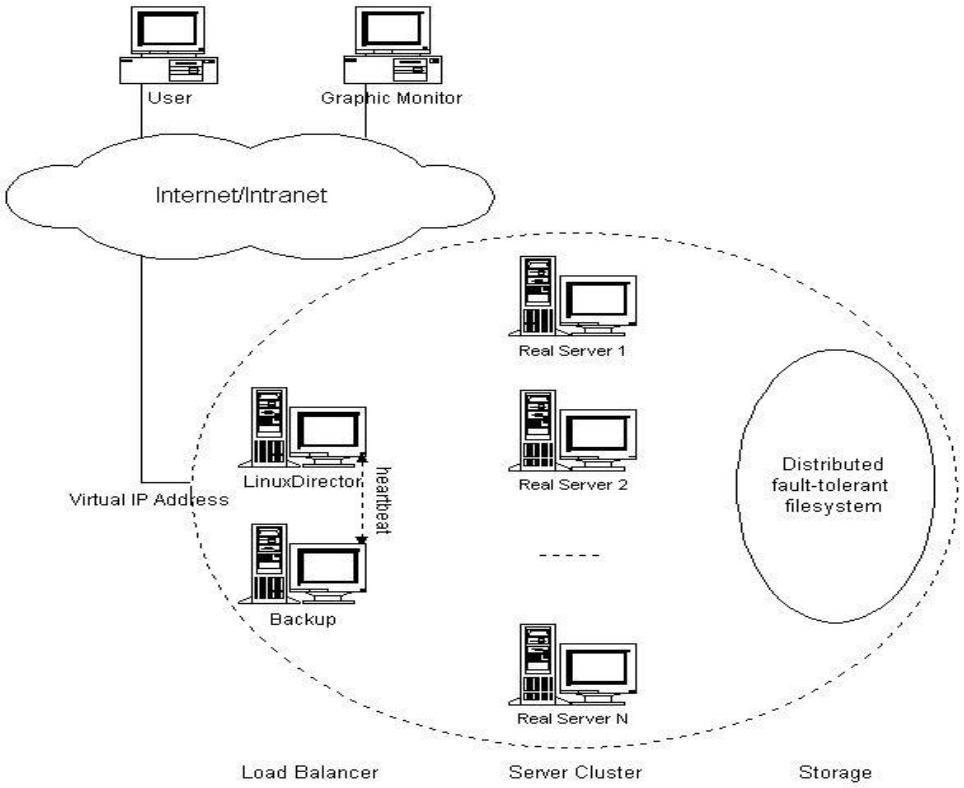


图 1.1　LVS 集群的通用结构

负载调度器（load balancer），它是整个集群对外面的前端机，负责将客户的请求发送到一组服务器上执行，而客户认为服务是来自一个 IP 地址上的。当客户请求到达时，调度器只根据负载情况从服务器池中选出一个服务器，将该请求转发到选出的服务器，并记录这个调度；当这个请求的其他报文到达，也会被转发到前面选出的服务器。因为所有的操作都是在操作系统核心空间中将完成的，它的调度开销很小，所以具有很高的吞吐率。

服务器池（server pool），是一组真正执行客户请求的服务器，执行的任务有 WEB、MAIL、FTP 和 DNS 等。服务器池的结点数目是可变的，当整个系统收到的负载超过目前所有结点的处理能力时，可以在服务器池中增加服务器来满足不断增长的请求负载。对大多数网络服务来说，结点与结点间不存在很强的相关性，所以整个系统的性能可以随着服务器池的结点数目增加而线性增长。

后端存储（backend storage），它为服务器池提供一个共享的存储区，这样很容易使得服务器池拥有相同的内容，提供相同的服务。

Graphic Monitor 是为系统管理员提供整个集群系统的监视器，它可以监视系统中每个结点的状况。

1.2 IP 负载均衡技术

在已有的 IP 负载均衡技术中有三种，一是通过网络地址转换实现虚拟服务器的 VS/NAT 技术（Virtual Server via Network Address Translation），二是通过直接路由的 VS/DR 技术（Virtual Server via Direct Routing），三是通过 IP 隧道实现虚拟服务器的 VS/TUN 技术（Virtual Server via IP Tunneling）。

1.2.1 通过 NAT 实现虚拟服务器（VS/NAT）

VS/NAT 的体系结构如图 1.2 所示，在一组服务器前有一个调度器，它们是通过 Switch/HUB 相连接的。这些服务器提供相同的网络服务、相同的内容，即不管请求被发送到哪一台服务器，执行结果是一样的。

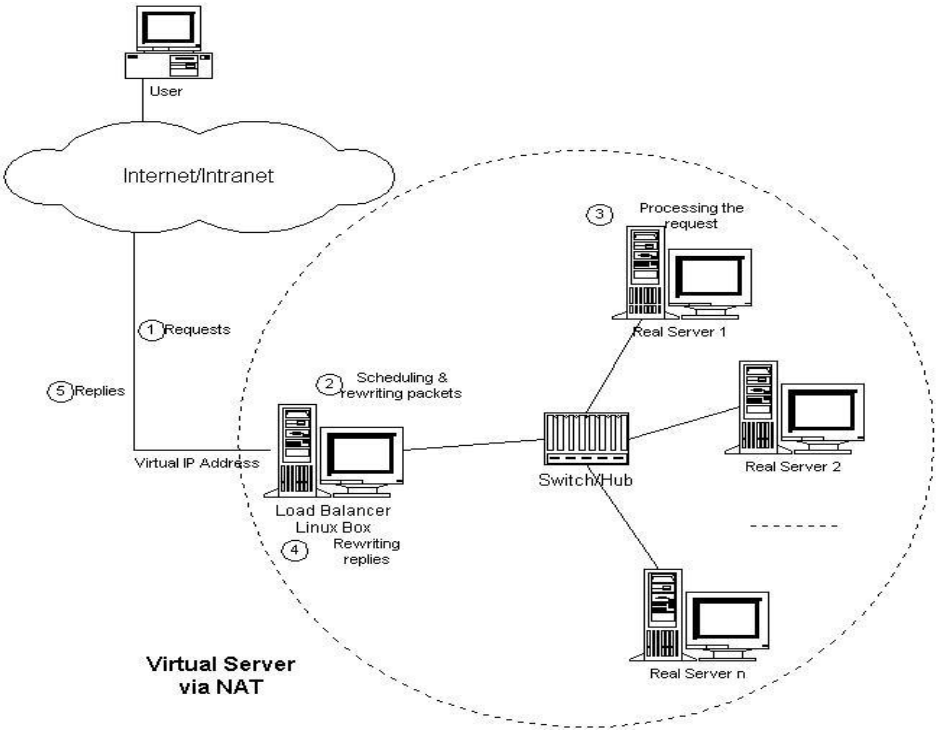


图 1.2　VS/NAT 的体系结构

以如下的 VS/NAT 配置为例，来了解报文的流程：

Protocol	Virtual IP Address	Port	Real IP Address	Port	Weight
TCP	58.251.62.141	80	172.16.81.143	80	1
			172.16.81.144	80	2
TCP	58.251.62.141	21	172.16.81.145	21	1

数据流程的时序图为：

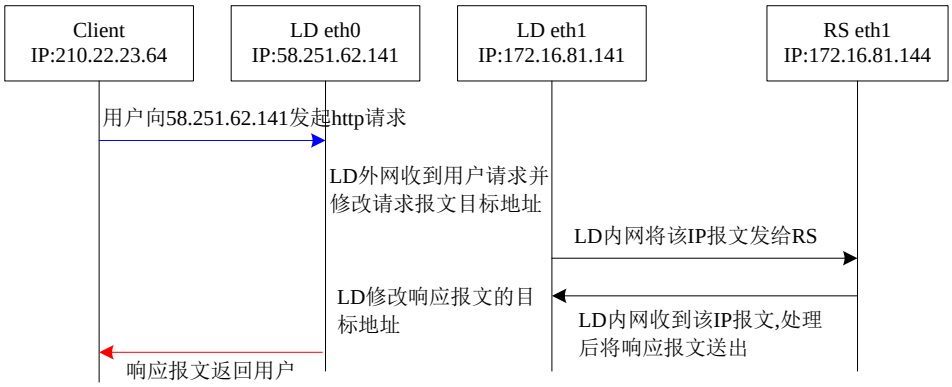


图 2.3　VS/NAT 数据流程时序图

1. 客户端浏览器输入 58.251.62.141 向 58.251.62.141 发出 http 请求.

2. Load Balancer 的外网(eth0)收到该次请求.

SOURCE	210.22.23.64:4143	DEST	58.251.62.141:http

3. IPVS 调度器根据各个 Real Server 的负载情况，动态地选择一台 Real Server(例如 172.16.81.144)，将请求报文的目标地址改写发送给 172.16.81.144

SOURCE	210.22.23.64:4143	DEST	172.16.81.144:http

4. Real Server 收到请求报文并处理形成响应报文，由于 Real Server 上的网关地址为 Load Balancer，响应报文从 Real Server 发往 Load Balancer。

SOURCE	58.251.62.141:http	DEST	210.22.23.64:4143

5. Load Balancer 收到 172.16.81.144 的响应报文后，将响应报文的原地址修改为虚拟 IP 地址，并发送给客户端。

SOURCE	172.16.81.144:http	DEST	210.22.23.64:4143

6. 客户认为得到正常的服务，而不知道是哪一台服务器处理的.

1.2.2 通过直接路由实现虚拟服务器（VS/DR）

在 VS/NAT 的集群系统中，请求和响应的数据报文都需要通过负载调度器，当真实服务器的数目在 10 台和 20 台之间时，负载调度器将成为整个集群系统的新瓶颈。大多数 Internet 服务都有这样的特点：请求报文较短而响应报文往往包含大量的数据。如果能将请求和响应分开处理，即在负载调度器中只负责调度请求而响应直接返回给客户，将极大地提高整个集群系统的吞吐量。VS/DR 的体系结构如图 1.4 所示：调度器和服务器组都必须在物理上有一个网卡通过不分断的局域网相连，如通过交换机或者高速的 HUB 相连。VIP 地址为调度器和

服务器组共享，调度器配置的 VIP 地址是对外可见的，用于接收虚拟服务的请求报文；所有的服务器把 VIP 地址配置在各自的 Non-ARP 网络设备上，它对外面是不可见的，只是用于处理目标地址为 VIP 的网络请求。

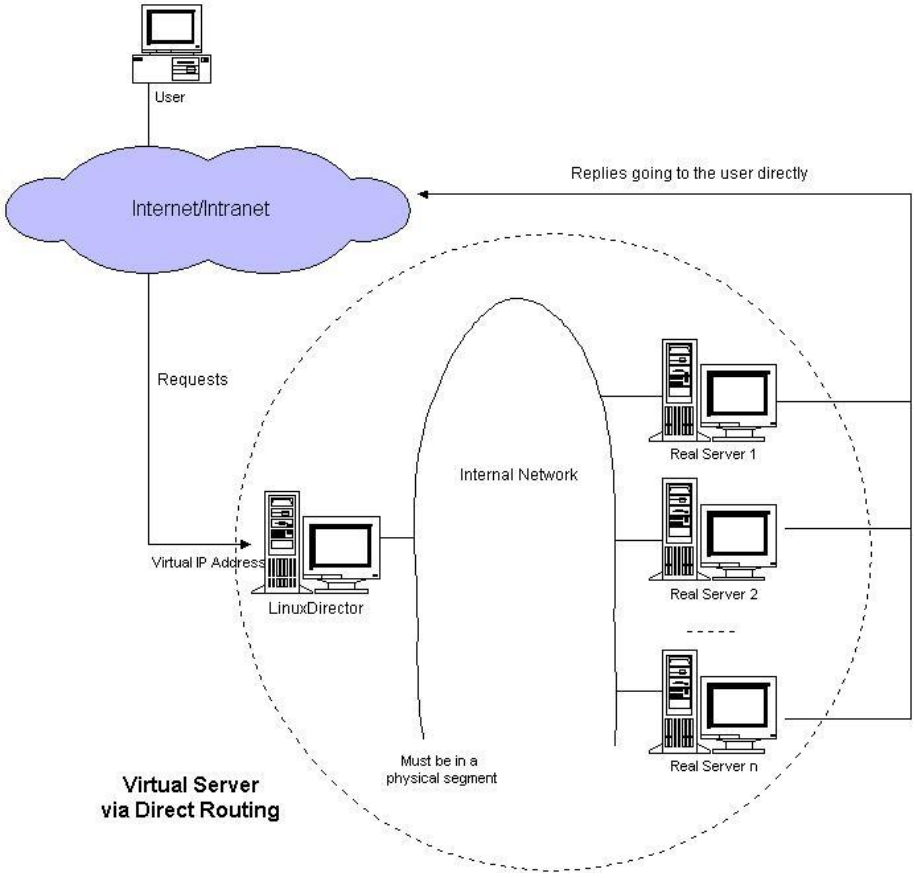


图 1.4　VS/DR 的体系结构

以如下的 VS/NAT 配置为例，来了解报文的流程：

Protocol	Virtual IP Address	Port	Real IP Address	Port	Weight
TCP	58.251.62.141 172.16.81.49	80	58.251.62.138(172.16.81.138)	80	1
			58.251. 63.49(172.16.80.49)		2
			58.251. 13.52(172.16.13.52)		1

数据流程的时序图为：

欢迎点击这里的链接进入精彩的Linux公社 网站

Linux公社（www.Linuxidc.com）于2006年9月25日注册并开通网站，Linux现在已经成为一种广受关注和支持的一种操作系统，IDC是互联网数据中心，LinuxIDC就是关于Linux的数据中心。

Linux公社是专业的Linux系统门户网站，实时发布最新Linux资讯，包括Linux、Ubuntu、Fedora、RedHat、红旗Linux、Linux教程、Linux认证、SUSE Linux、Android、Oracle、Hadoop、CentOS、MySQL、Apache、Nginx、Tomcat、Python、Java、C语言、OpenStack、集群等技术。

Linux公社（LinuxIDC.com）设置了有一定影响力的Linux专题栏目。

Linux公社 主站网址：www.linuxidc.com 旗下网站：www.linuxidc.net

包括：**Ubuntu 专题** **Fedora 专题** **Android 专题** **Oracle 专题** **Hadoop 专题** **RedHat 专题** **SUSE 专题** **红旗 Linux 专题** **CentOS 专题**

Linux 公社微信公众号：linuxidc_com

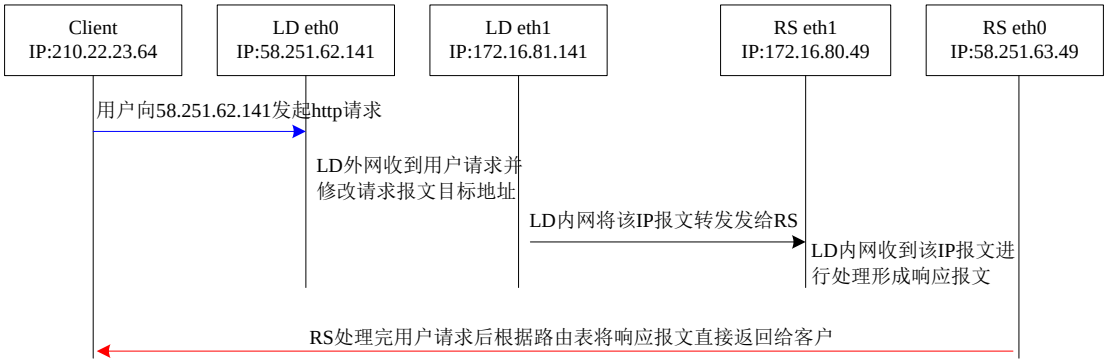


图 1.5　VS/DR 数据流程时序图

1. 客户端浏览器输入 58.251.62.141 向 58.251.62.141 发出 http 请求.

2. Load Balancer 的外网(eth0)收到该次请求.

SOURCE	210.22.23.64.4157	DEST	172.16.80.49:http

3. IPVS 调度器根据各个 Real Server 的负载情况，动态地选择一台 Real Server，将请求报文转发给 Real Server（例如 172.16.80.49）

SOURCE	210.22.23.64.4157	DEST	172.16.80.49:http

4. Real Server 的内网(eth1)收到 Load Balancer 发过来的 IP 报文并对 IP 报文解包，得到客户的请求包，发现包的目标地址被配置在本地的 lo 设备上，所以就处理这个请求。

5. Real Server 根据路由表将响应报文通过外网(eth0)直接返回给客户，请求报文的目标地址为 VIP，响应报文的源地址也为 VIP，所以响应报文不需要作任何修改，可以直接返回给客户.

SOURCE	58.251.62.141:http	DEST	210.22.23.64.4157

6. 客户认为得到正常的服务，而不知道是哪一台服务器处理的。

在 VS/DR 响应报文根据服务器的路由表直接返回给客户，而不经过负载调度器，所以负载调度器只处于从客户到服务器的半连接中，我们给出半连接的 TCP 有限状态机。如图 1.6 为 VS/DR 的 TCP 状态迁移，圈表示状态，箭头表示状态间的转换，箭头上的标识表示在当前状态上收到该标识的输入，迁移到下一个状态。VS/DR 的 TCP 状态迁移是按照半连接的 TCP 有限状态机进行的。

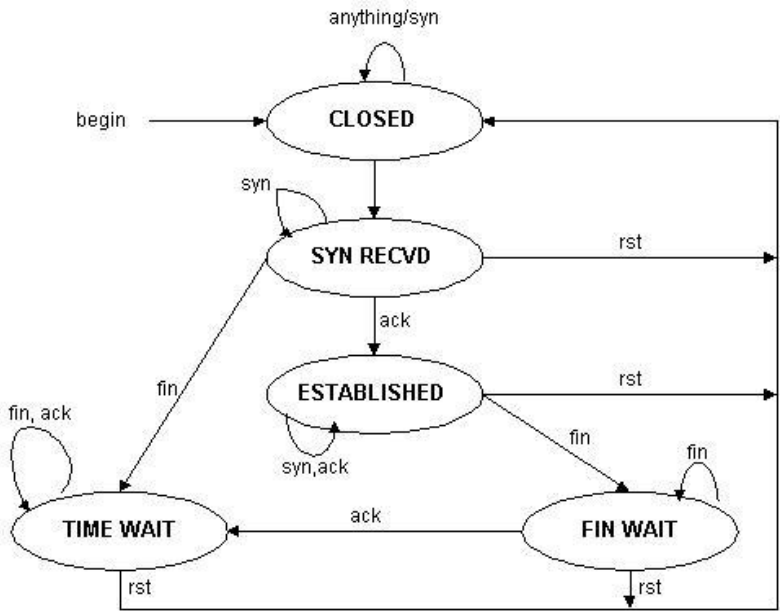


图 1.6　VS/DR 的 TCP 状态迁移

1.2.3 通过 IP 隧道实现虚拟服务器（VS/TUN）

跟 VS/DR 方法相同，VS/TUN 多数 Internet 服务的非对称特点，负载调度器中只负责调度请求，而服务器直接将响应返回给客户，可以极大地提高整个集群系统的吞吐量。IP 隧道（IP tunneling）是将一个 IP 报文封装在另一个 IP 报文的技术，这可以使得目标为一个 IP 地址的数据报文能被封装和转发到另一个 IP 地址。

我们利用 IP 隧道技术将请求报文封装转发给后端服务器，响应报文能从后端服务器直接返回给客户。但在这里，后端服务器有一组而非一个，所以我们不可能静态地建立一一对应的隧道，而是动态地选择一台服务器，将请求报文封装和转发给选出的服务器。这样，我们可以利用 IP 隧道的原理将一组服务器上的网络服务组成在一个 IP 地址上的虚拟网络服务。VS/TUN 的体系结构如图 3.3 所示，各个服务器将 VIP 地址配置在自己的 IP 隧道设备上。VS/TUN 的体系结构如图 1.7 所示

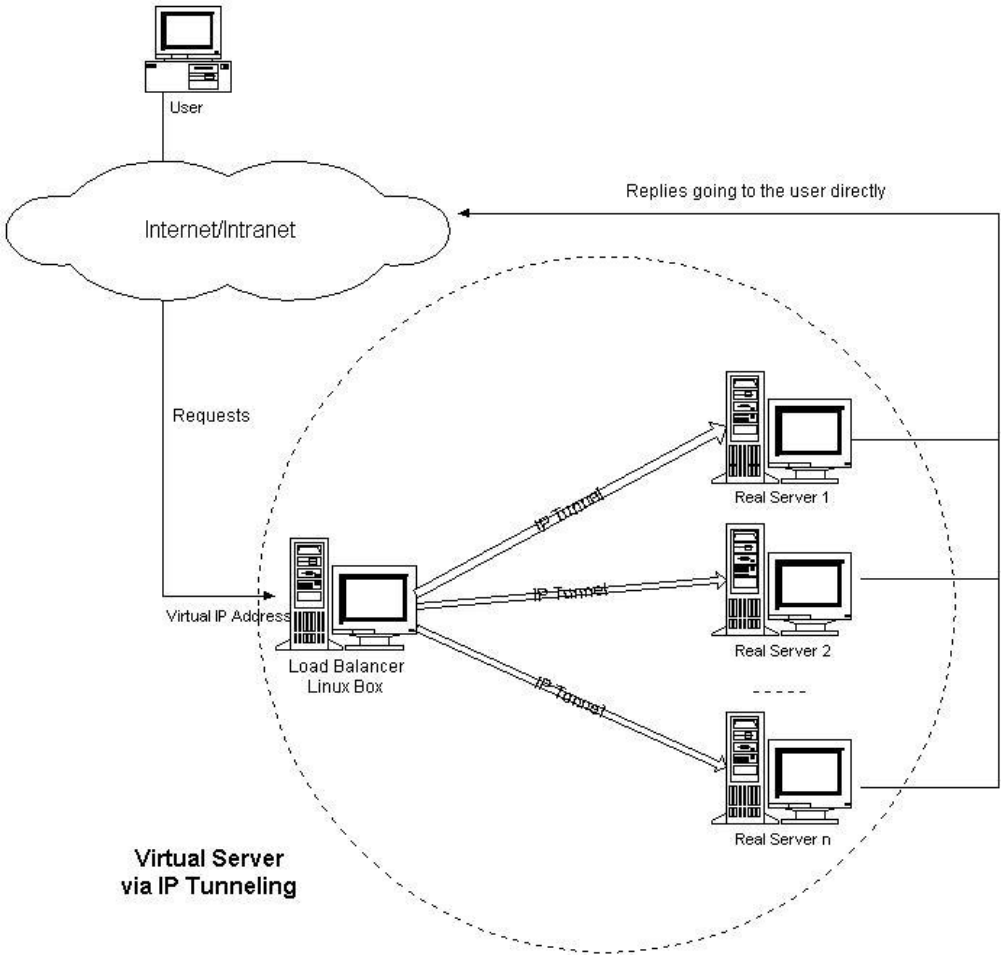
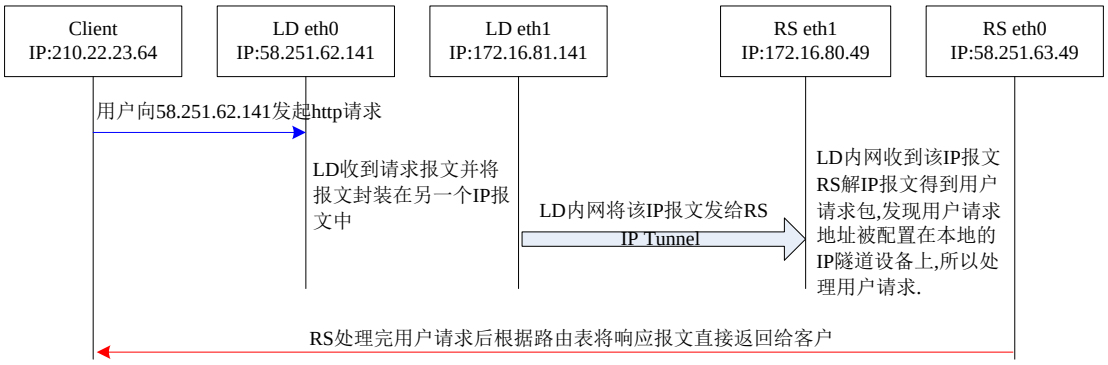


图 1.7　VS/TUN 的体系结构

以如下的 VS/NAT 配置为例，来了解报文的流程：

Protocol	Virtual IP Address	Port	Real IP Address	Port	Weight
TCP	58.251.62.141 172.16.81.49	80	58.251.62.138(172.16.81.138)	80	1
			58.251. 63.49(172.16.80.49)		2
			58.251. 13.52(172.16.13.52)		1

数据流程的时序图为：



1. 客户端浏览器输入 58.251.62.141 向 58.251.62.141 发出 http 请求.

2. Load Balancer 的外网(eth0)收到该次请求.

SOURCE	210.22.23.64.4157	DEST	58.251.62.141:http

3. IPVS 调度器根据各个 Real Server 的负载情况，动态地选择一台 Real Server，将请求报文封装在另一个 IP 报文中.

4. Load Balancer 的内网(eth1)将封装后的 IP 报文发给选出的 Real Server

SOURCE	172.16.81.141	DEST	172.16.80.49	
SOURCE	210.22.23.64.4157	DEST	58.251.62.141:http	(ipip-proto-4)

5. Real Server 的内网(eth1)收到 Load Balancer 发过来的 IP 报文并对 IP 报文解包，得到客户的请求包，发现包的目标地址被配置在本地的 IP 隧道设备上，所以就处理这个请求。

6. Real Server 根据路由表将响应报文通过外网(eth0)直接返回给客户，请求报文的目标地址为 VIP，响应报文的源地址也为 VIP，所以响应报文不需要作任何修改，可以直接返回给客户.

SOURCE	58.251.62.141:http	DEST	210.22.23.64.4157

7. 客户认为得到正常的服务，而不知道是哪一台服务器处理的。

VS/DR 负载调度器也只处于从客户到服务器的半连接中，按照半连接的 TCP 有限状态机进行状态迁移。

1.2.4 三种 IP 负载均衡技术比较

三种 IP 负载均衡技术的优缺点归纳在下表中：

	VS/NAT	VS/DR	VS/TUN
Server	any	Non-arp device	Tunneling
server network	private	LAN	LAN/WAN
server number	low (10~20)	High (100)	High (100)
server gateway	load balancer	Own router	own router

VS/NAT 的优点是服务器可以运行任何支持 TCP/IP 的操作系统，它只需要一个 IP 地址配置在调度器上，服务器组可以用私有的 IP 地址。缺点是它的伸缩能力有限，当服务器结点数目升到 20 时，调度器本身有可能成为系统的新瓶颈，因为在 VS/NAT 中请求和响应报文都需要通过负载调度器。

VS/DR 优点是负载调度器可以处理大量的请求，因为调度器只处理客户到服

务器端的连接，响应数据可以直接从独立的网络路由返回给客户，这可以极大地提高 LVS 集群系统的伸缩性。缺点是要求负载调度器与实际服务器都有一块网卡连在同一物理网段上，服务器网络设备（或者设备别名）不作 ARP 响应，或者能将报文重定向（Redirect）到本地的 Socket 端口上。

VS/TUN 的优点是负载调度器可以处理大量的请求，它甚至可以调度百台以上的服务器（同等规模的服务器），而它不会成为系统的瓶颈，因为负载调度器只将请求调度到不同的后端服务器，后端服务器将应答的数据直接返回给用户。缺点是 VS/TUN 技术有 IP 隧道的开销并且对服务器有要求，即所有的服务器必须支持“IP Tunneling”或者“IP Encapsulation”协议。

第二章 IPVS+HeartBeat+Mon 建立 LVS 系统

2.1 系统搭建流程

由 IPVS 负载均衡技术可知，VS/TUN 体系结构分为 Load Balancer 和 Real Server 两大部分:

2.1.1 Load Balancer 的搭建流程

1 内核升级以支持 IPVS 和 IP Tunneling

2 安装 IPVS 的管理工具 ipvsadm 并进行 IPVS 配置

3 安装 Real Server 的监控软件 mon 并进行 mon 设置

4 安装实现 HA 系统的软件 HeartBeat 并进行 HeartBeat 设置

5 系统启动脚本的配置.

6 假如您对安全步骤及安装过程不感兴趣,请直接跳到 2.7 节,按照说明文字,下载 LVSPackage.tar.gz 可以直接安装.

2.1.2 Real Server 的搭建流程

1 内核升级以支持 IP Tunneling

2 系统配置

a 对于使用了状态机及对应的 iptables 规则的主机，需要更新 iptables 的规则，取消其中所有依赖于状态机的配置，这有可能意味着现有的 iptables 规则的重写。

b 对于未使用状态机机器对应的 iptables 规则的主机，则无需做 iptables 规则的修改。

c 当整体切换完成后，需要关闭 Real Server 上对外服务的端口，以确保安全。

2.2 内核升级

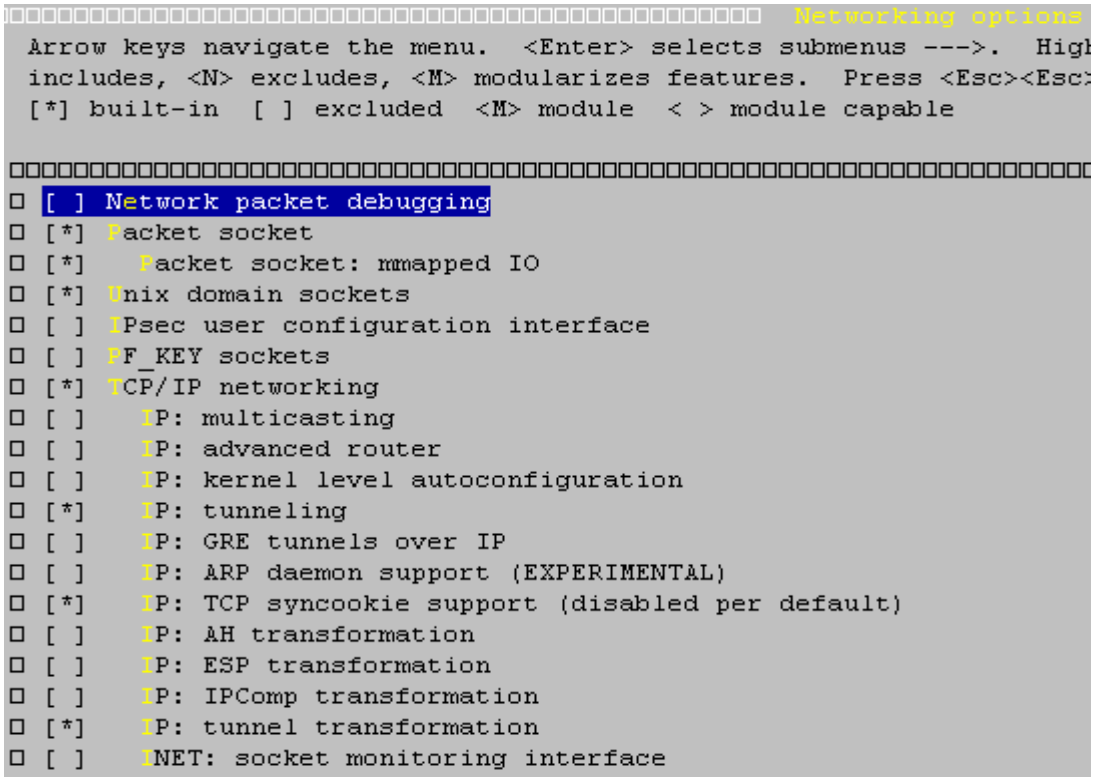
Ipvs 内核由其用途来说，主要分为 2 种： LD 的和 RS 的。下面以内核版本 2.6.16.21 为例，分别说明 2 种内核的内核配置参数。

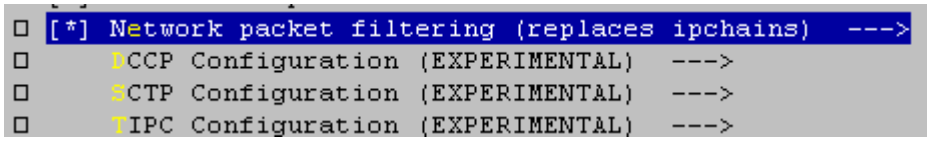
2.2.1 LD Server 的内核编译参数

LD Server 所需内核中需要进行的改动较多，主要为 network option; lvs;及

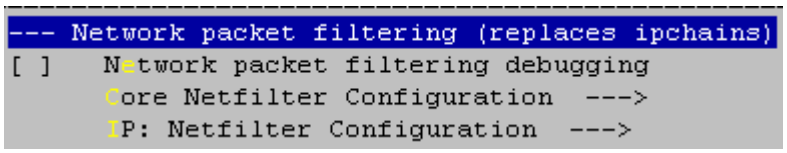
iptables，状态机。

a networking options:

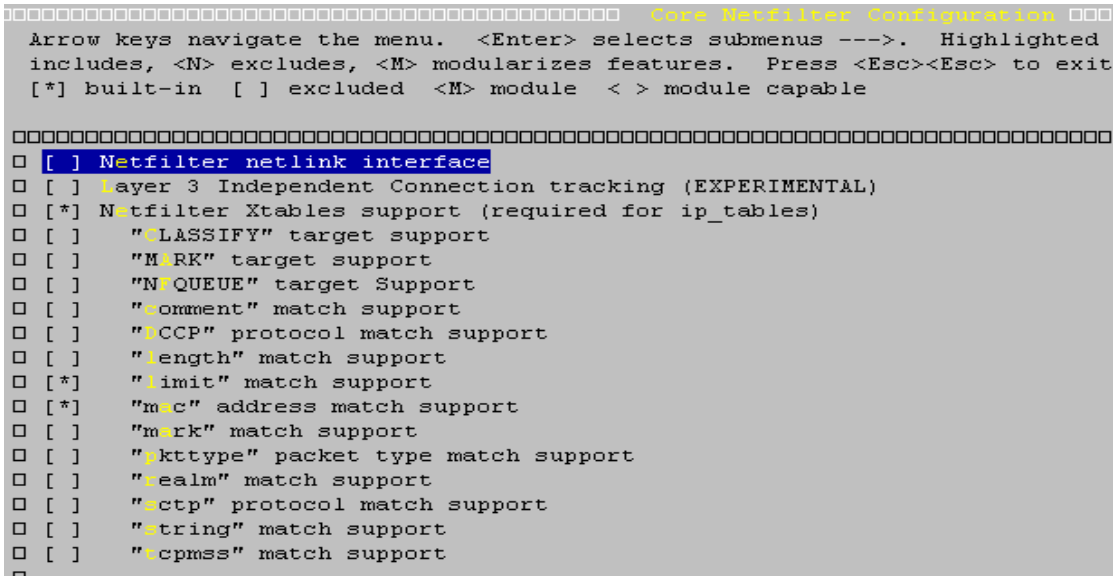




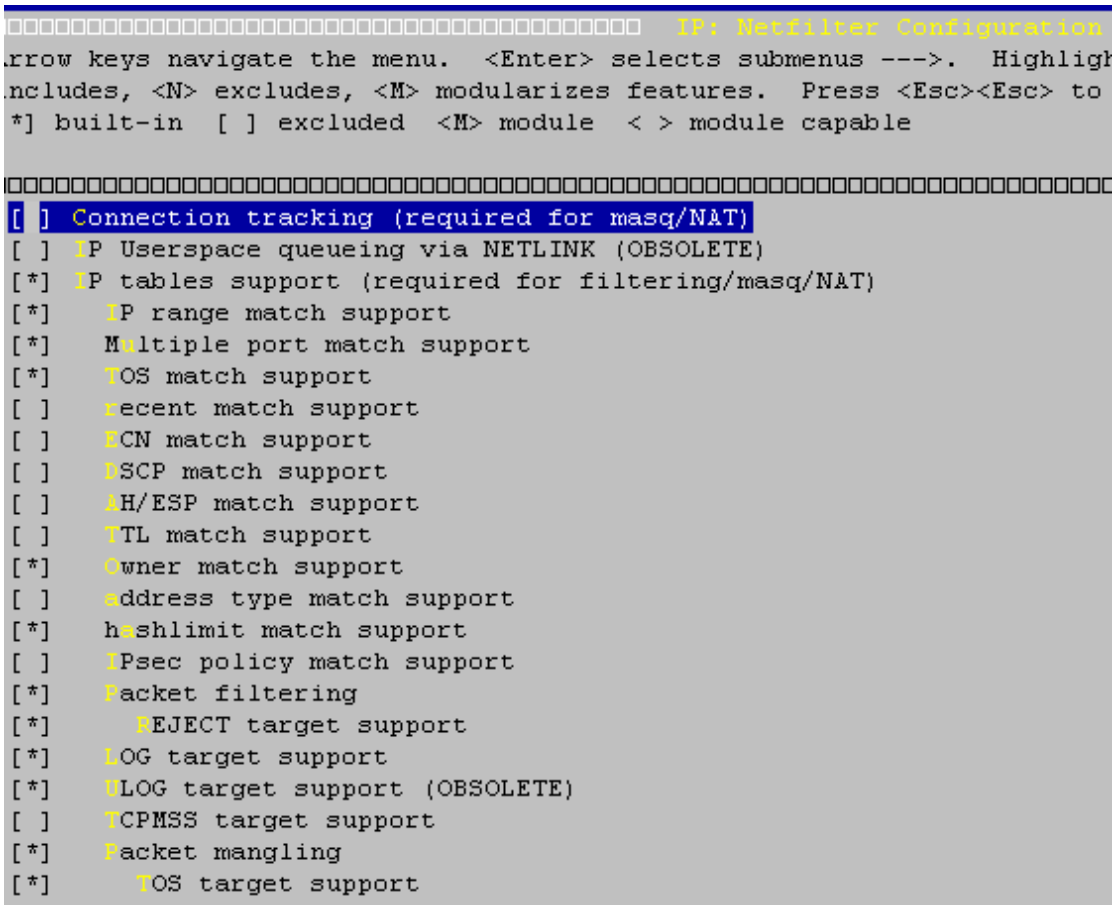
进入 network packet filtering



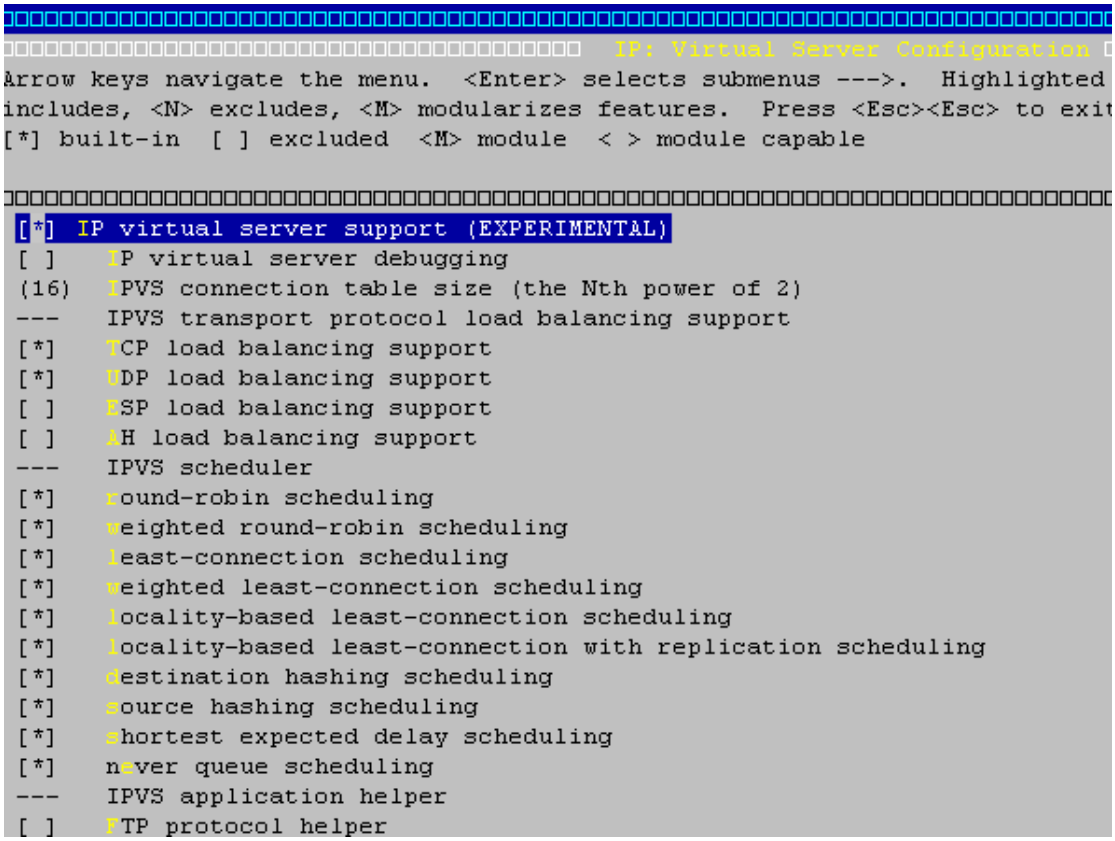
配置 Core


Core Netfilter Configuration
Arrow keys navigate the menu. <Enter> selects submenus --->. Highlighted
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit
[*] built-in [ ] excluded <M> module < > module capable
[ ] Netfilter netlink interface
[ ] Layer 3 Independent Connection tracking (EXPERIMENTAL)
[*] Netfilter Xtables support (required for ip_tables)
"CLASSIFY" target support
"MARK" target support
"NFQUEUE" target Support
"comment" match support
"DCCP" protocol match support
"length" match support
"limit" match support
"mac" address match support
"mark" match support
"pkttype" packet type match support
"realm" match support
"sctp" protocol match support
"string" match support
"tcpmss" match support

配置 IP：

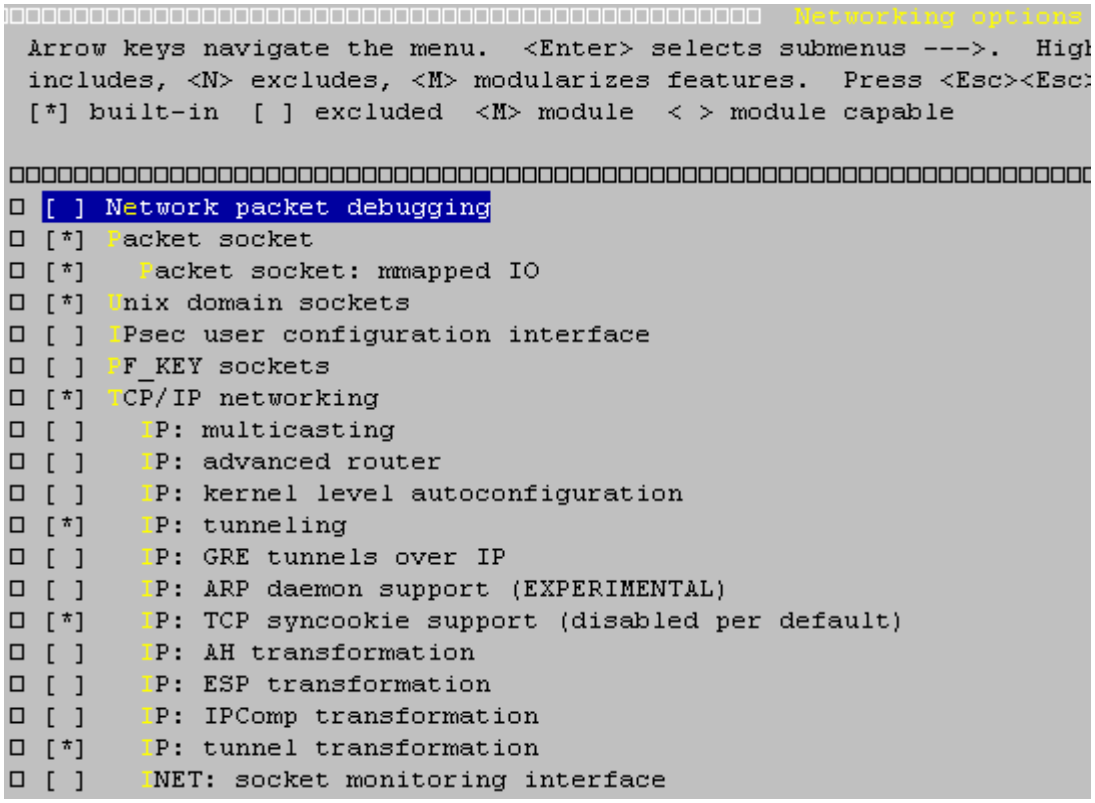

IP: Netfilter Configuration
Arrow keys navigate the menu. <Enter> selects submenus --->. Highligh
Includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to
*] built-in [ ] excluded <M> module < > module capable
[ ] Connection tracking (required for masq/NAT)
[ ] IP Userspace queueing via NETLINK (OBSOLETE)
[*] IP tables support (required for filtering/masq/NAT)
[*] IP range match support
[*] Multiple port match support
[*] TOS match support
[ ] recent match support
[ ] ECN match support
[ ] DSCP match support
[ ] AH/ESP match support
[ ] TTL match support
[*] Owner match support
[ ] address type match support
[*] hashlimit match support
[ ] IPsec policy match support
[*] Packet filtering
[*] REJECT target support
[*] LOG target support
[*] ULOG target support (OBSOLETE)
[ ] TCPMSS target support
[*] Packet mangling
[*] TOS target support

b 配置 LVS

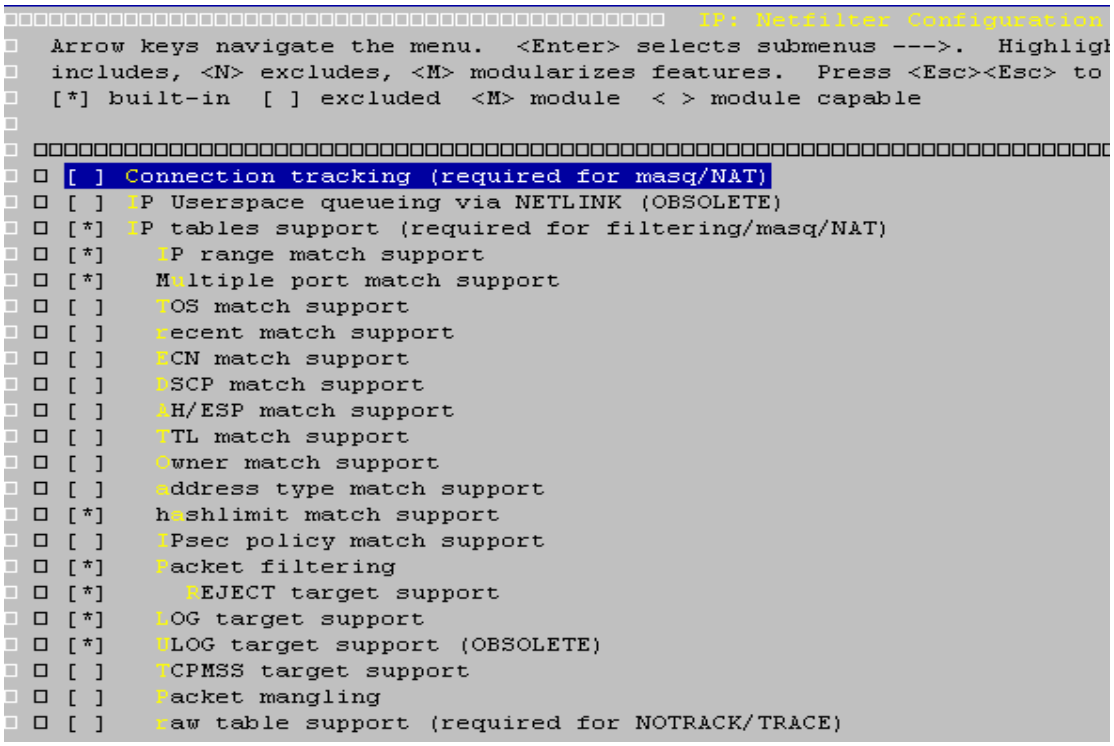


2.2.2 Real Server 的内核编译参数

RS 的内核只需在现有内核的基础上开启 net options 中的 ip tunnlig 选项即可，如下图：



同时需要关闭状态机，公司内核中默认没有状态机的，因此，也可以不进行该配制。



2.2.3 内核升级步骤

a 根据 LD/RS，将对应的内核放置在系统的/boot 目录下。

b 修改/etc/lilo.conf 文件

c 执行 lilo 命令，进行变更

d reboot

需要注意的是，如果是跨内核版本升级，如 2.4 升到 2.6，需要考虑主机的网线顺序的更换问题。

2.3 安装 ipvsadm 及配置 IPVS

2.3.1 安装 ipvsadm(install_ipvs.sh)

a http://www.linuxvirtualserver.org/software/ipvs.html 下载对应内核版本的 ipvsadm

b 进入 ipvsadm 目录，并 make 以及 make install

c 输入命令 ipvsadm，若有如下提示则安装正确。

IP Virtual Server version 1.2.1 (size=65536)

Prot LocalAddress:Port Scheduler Flags

-> RemoteAddress:Port Forward Weight ActiveConn InActConn

2.3.2 配置 IPVS(config_ipvs.sh)

a 配置系统参数（eth0 为绑定 VIP 的网卡设备）

```
echo "0" >/proc/sys/net/ipv4/ip_forward
echo "1" >/proc/sys/net/ipv4/conf/all/send_redirects
echo "1" >/proc/sys/net/ipv4/conf/default/send_redirects
echo "1" >/proc/sys/net/ipv4/conf/eth0/send_redirects
```

b 配置 IPVS 的服务类型、VIP 地址以及对应的 RS 信息，例如：

```
/sbin/ipvsadm -A –t 172.16.81.141:80 -s wlc
/sbin/ipvsadm -a -t 172.16.81.141:80 -r 172.19.81.139 -i -w 1
/sbin/ipvsadm -a -t 172.16.81.141:80 -r 172.19.80.50 -i -w 1
…（ipvsadm的使用请参见附录1）
```

2.4 安装 mon 及配置 mon

2.4.1 安装 mon(install_mon.sh)

a 软件的准备

1 mon-0.99.1.tar.gz

2 Mon-0.11.tar.gz

3 Time-HiRes-01.20.tar.gz

4 Period-1.20.tar.gz

5 Convert-BER-1.31.tar.gz

b 安装 perl 模块，就是 2.3.4.5，三个 perl 模块和一个 Mon 编译进系统

```
cd <模块>
perl MakeFile.pl
make
make install
```

c 直接 tar xvzf mon-0.99.1.tar.gz ，mon 就安装完毕。

2.4.2 配置 mon(config_mon.sh)

a 将 lvs 的 alert 脚本 lvs.alert 复制到 mon-0.99.1/alert.d，(lvs.alert 请见 LVSPackage.tar.gz)

b 将 mon 的配置文件 mon.cf 复制到 mon-0.99.1（mon.cf 请见 LVSPackage.tar.gz）

2.5 安装 HeartBeat 及配置 HeartBeat

2.5.1 安装 HeartBeat(install_HB.sh)

a 软件准备

1 libnet-1.1.2.1.tar.gz

2 heartbeat-2.0.7.tar.gz

b 安装 libnet

```
tar -xzvf libnet-1.1.2.1.tar.gz
cd /root/libnet
./configure
make
make install
```

c 安装HeartBeat

```
tar -xzvf heartbeat-2.0.7.tar.gz
rm heartbeat-2.0.7.tar.gz
cd heartbeat-2.0.7
groupadd -g 65 haclient
useradd -g 65 -u 17 hacluster
./ConfigureMe configure
make
make install
```

2.5.2 配置 HeartBeat(config_HB.sh)

a 复制配置文件（下述配置文件请见LVSPackage.tar.gz）

```
cp doc/ha.cf doc/haresources doc/authkeys /etc/ha.d/
cp doc/ha.cf /etc/ha.d/ha.cf
cp doc/haresources /etc/ha.d/haresources
cp doc/TencentLvs /etc/ha.d/resource.d/
chmod 600 authkeys
```

b 修改配置文件

在/etc/ha.d/ha.cf中加入Active和Standby LD的节点信息以及heartbeat心跳线的制

作，例如

echo ″node $A_hostname″ >> /etc/ha.d/ha.cf

echo ″node $S_hostname″ >> /etc/ha.d/ha.cf

echo “ucast eth1 $S_hostip” >> /etc/ha.d/ha.cf

在/etc/ha.d/haresources中加入需要拉动的资源信息，例如

echo ″S_hostname IPaddr::172.16.81.141/24/eth1 TencentLvs″

>> etc/ha.d/haresources

2.6 系统配置信息

将如下三句加入到/etc/rc.d/rc.local中

/usr/lib/heartbeat/heartbeat

Sleep(40)

/usr/lib/heartbeat/hb_takeover

2.7 自动化安装包使用说明（LVSPakeage.tar.gz）

1. cp vmlinuz-2.6.16.21-p4-LVS-LD /boot/vmlinuz-2.6.16.21

   vi /etc/lilo.conf，增加如下语句

   image = /boot/vmlinuz-2.6.16.21

   root = /dev/sda1

   label = Linux2616

   read-only

   将 default 修改为 default = Linux2616

   reboot
2. ./install_ipvs.sh
3. ./install_mon.sh
4. ./install_HB.sh
5. 修改/usr/local/lvs/lvs.conf 配置文件
6. ./config_ipvs.sh，利用 ipvsadm 查看结果
7. ./config_mon.sh
8. ./configHB.sh uname ucastip phost phostip (ucastip 为本机心跳 IP，phostip 为对方的心跳 IP)

欢迎点击这里的链接进入精彩的Linux公社 网站

Linux公社（www.Linuxidc.com）于2006年9月25日注册并开通网站，Linux现在已经成为一种广受关注和支持的一种操作系统，IDC是互联网数据中心，LinuxIDC就是关于Linux的数据中心。

Linux公社是专业的Linux系统门户网站，实时发布最新Linux资讯，包括Linux、Ubuntu、Fedora、RedHat、红旗Linux、Linux教程、Linux认证、SUSE Linux、Android、Oracle、Hadoop、CentOS、MySQL、Apache、Nginx、Tomcat、Python、Java、C语言、OpenStack、集群等技术。

Linux公社（LinuxIDC.com）设置了有一定影响力的Linux专题栏目。

Linux公社 主站网址：www.linuxidc.com 旗下网站：www.linuxidc.net

包括：**Ubuntu 专题** **Fedora 专题** **Android 专题** **Oracle 专题** **Hadoop 专题** **RedHat 专题** **SUSE 专题** **红旗 Linux 专题** **CentOS 专题**

Linux 公社微信公众号：linuxidc_com

9. ./startHB.sh ucastip (ucastip 为自己的心跳 IP)

10. 相应的 iptables 设置

2.8 Real Server 的配置

a 配置系统参数

```
echo "0" >/proc/sys/net/ipv4/ip_forward
```

b 配置tunl0

```
/sbin/ifconfig tunl0 ${VIP} broadcast ${VIP} netmask 0xffffffff up

/sbin/route add -host ${VIP} dev tunl0
```

c 解决arp问题

```
echo 1 > /proc/sys/net/ipv4/conf/tunl0/arp_ignore

echo 2 > /proc/sys/net/ipv4/conf/tunl0/arp_announce

echo 1 > /proc/sys/net/ipv4/conf/all/arp_ignore

echo 2 > /proc/sys/net/ipv4/conf/all/arp_announce
```

d 解决源地址验证问题

```
echo 0 > /proc/sys/net/ipv4/conf/tunl0/rp_filter

echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter
```

第三章 VS/TUN 模式压力测试报告

3.1 VS/TUN 模式压力测试结论

1 对 Load Banlancer 来说，在 VS/TUN 模式，千兆网卡的条件下，请求包的长度<1K 的条件下，有如下结论：

a LD SERVER 可以支持到每秒 200K 个包

b 在同等的情况下 VS/DR 模式比 VS/TUN 模式每秒处理的包量要多 30%

c 千兆网带宽不是瓶径

d 由于对网卡的处理需要一个单独的 CPU 进行处理，系统的瓶径是处理网卡所消耗的 CPU 资源。

e 对 Load Banlancer，由于 IP 封装，出流量稍微大于入流量

2 当 Banlancer 使用百兆网卡时，网络资源也会是 Banlancer 的瓶径。

3 在请求包比较大的时，比如大量上传请求的情况下，带宽也将成为瓶径。

4 对 REAL SERVER 来说，压力的情况等同于普通的 SERVER 的情况。

3.2 千 M 网卡，模式 VS/TUN（外网 VIP,HTTP 服务）

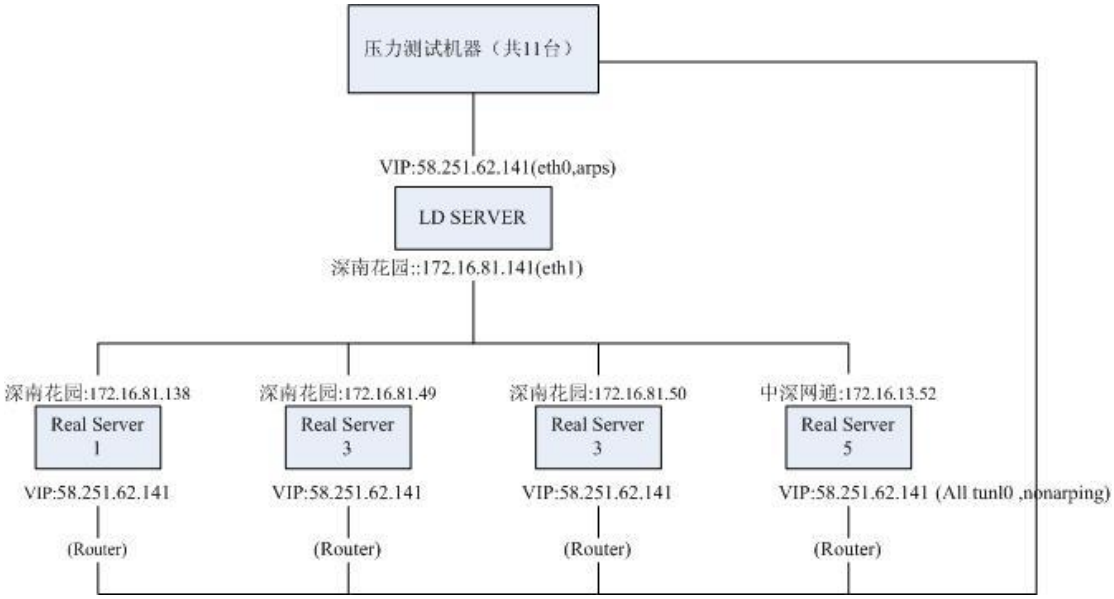


图 3.1 千 M 网卡的压力测试结构

3.2.1 压力测试条件

请求包长度：360 byte 响应包长度：1K byte

压力测试机器配置：PE1850,4G,1*146G-4G,1*146G

LDServer 配置：PESC1425,8G,1*80G SATA-8G,1*80G

RS 配置：PESC1425,8G,1*80G SATA-8G,1*80G

3.2.2 LD SERVER 的情况

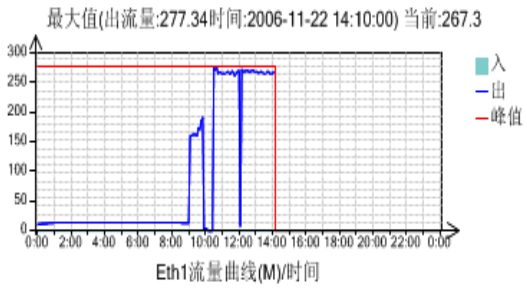


图 3.2 LD SERVER 流量情况

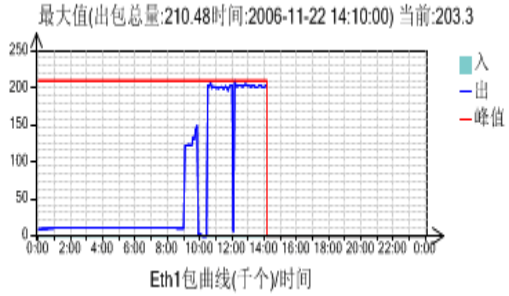


图 3.3 LD SERVER 包量情况

```
top - 14:33:30 up 5 days, 20:41,  2 users,  load average: 0.58, 0.57, 0.55
Tasks:  72 total,   1 running,  71 sleeping,   0 stopped,   0 zombie
Cpu0  :  0.3% us,  0.0% sy,  0.0% ni, 99.7% id,  0.0% wa,  0.0% hi,  0.0% si
Cpu1  :  0.0% us,  0.0% sy,  0.0% ni, 83.7% id,  0.0% wa,  3.0% hi, 13.3% si
Cpu2  :  0.0% us,  0.0% sy,  0.0% ni, 100.0% id,  0.0% wa,  0.0% hi,  0.0% si
Cpu3  :  0.0% us,  0.0% sy,  0.0% ni,  8.0% id,  0.0% wa,  0.3% hi, 91.7% si
Mem:   8312876k total,  1015504k used,  7297372k free,   122740k buffers
Swap:  2048276k total,        0k used,  2048276k free,   775124k cached
```

图 5.4 LD SERVER 每个 CPU 的情况。

结论：

从 CPU 的情况可以看出，CPU1 和 CPU3 有工作在上面，如图 5.4 所示，CPU 使用基本上已经到 90%以上，系统基本上已经达到极限，此时：

1. 流量达到 270M 以上
2. 包量达到 200K 以上
3. 网卡对应的 CPU 使用率 90%以上

3.2.3 REAL SERVER 172.16.80.49 的情况

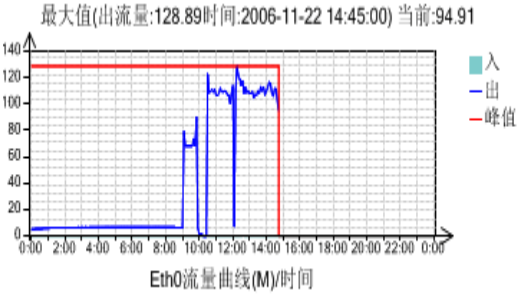


图 3.5 172.16.80.49 流量曲线

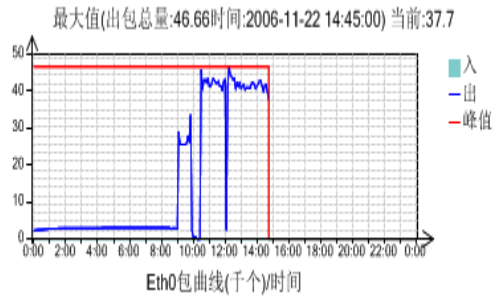


图 3.6 172.16.80.49 包量曲线

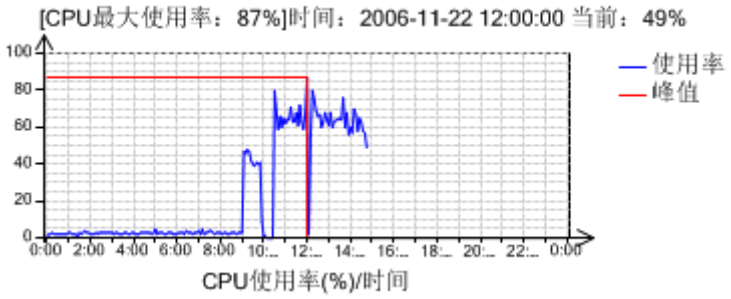


图 3.7 172.16.80.49 CPU 使用情况

3.2.4 REAL SERVER 172.16.81.138 的情况

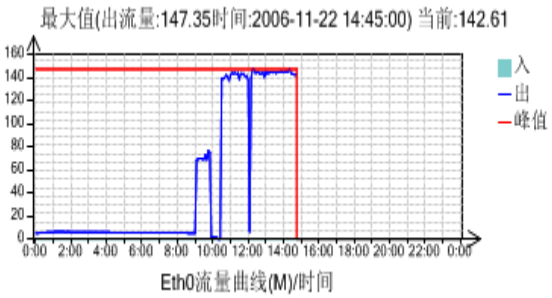


图 3.8 172.16.81.138 流量曲线

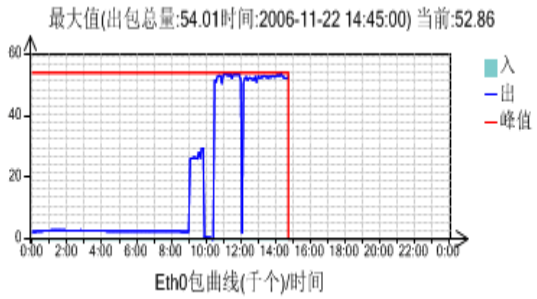


图 3.9 172.16.81.138 包量曲线

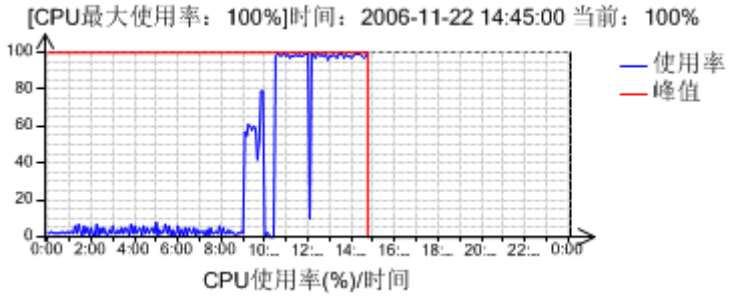


图 3.10　172.16.81.138 CPU 使用情况

3.2.5 REAL SERVER 172.16.13.52 的情况

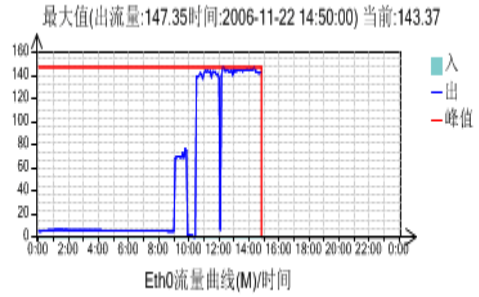


图 3.11 172.16.13.52 流量曲线

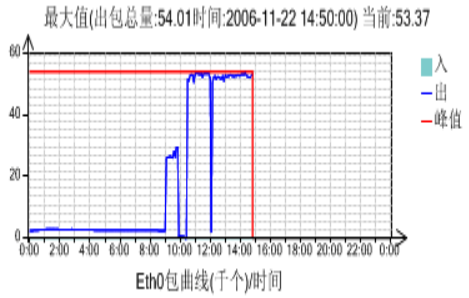


图 3.12　172.16.13.52 包量曲线

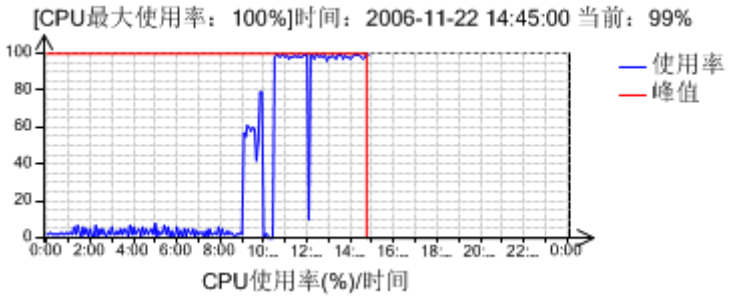


图 3.13 172.16.13.52 CPU 使用情况

3.2.6．REAL SERVER 172.16.80.50 的情况

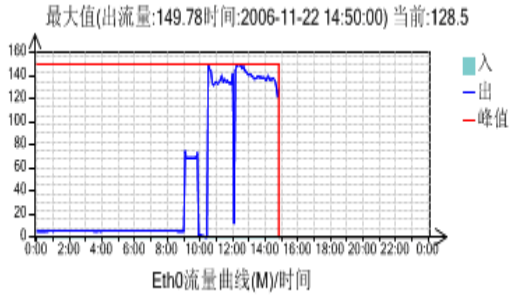


图 3.14 172.16.80.50 流量曲线

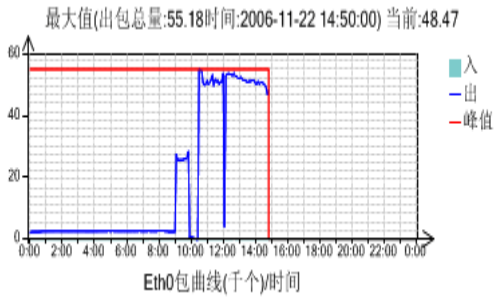


图 3.15　172.16.80.50 包量曲线

图 3.16　172.16.80.50 CPU 使用情况

3.2.6．REAL SERVER 结论

从 Real Server 的 CPU 使用情况可以看出, CPU 使用基本上已经到 90%以上，系统基本上已经达到极限，此时：

4．流量在 140M 左右

5．包量 50K 以上，四个 RS 的出包量之和等于 LD 的出包量

6．网卡对应的 CPU 使用率 90%以上

3.3 百 M 网卡，模式 VS/TUN（内网 VIP，UDP 服务）

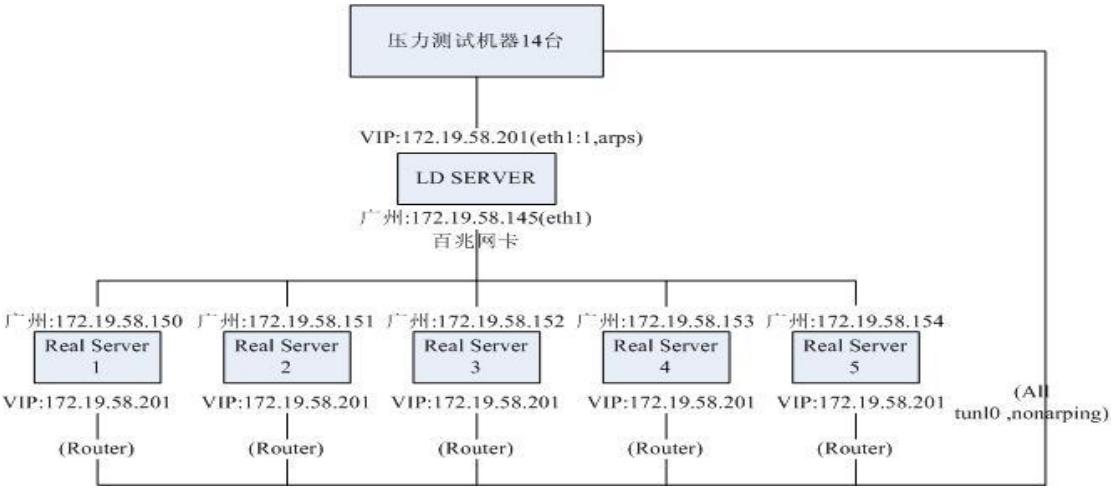


图 3.17　UDP 压力测试结构

3.3.1　压力测试条件

请求包长度：12 byte　响应包长度：12 byte

压力测试机器配置：PE1850,4G,1*146G-4G,1*146G

LDServer 配置：PE1850,4G,1*146G-4G,1*146G

RS 配置：PE1850,4G,1*146G-4G,1*146G

3.3.2　LD SERVER 的情况

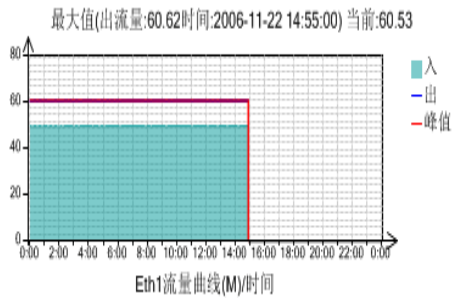


图 3.18　LD SERVER 流量情况

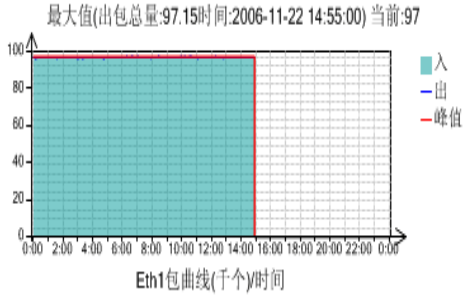


图 3.19　LD SERVER 包量情况

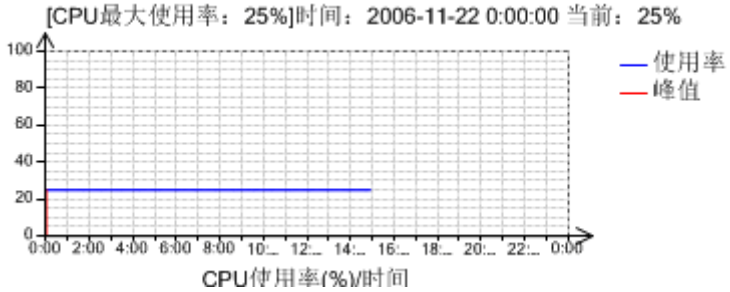


图 3.20　LD SERVERCPU 情况

```
top - 15:00:04 up 2 days,  4:06,  1 user,  load average: 1.00, 1.00, 1.00
Tasks:  30 total,   2 running,  28 sleeping,   0 stopped,   0 zombie
Cpu0  :   0.0% user,   0.0% system,   0.0% nice, 100.0% idle
Cpu1  :   0.0% user,   0.0% system,   0.0% nice, 100.0% idle
Cpu2  :   0.0% user,   0.0% system,   0.0% nice, 100.0% idle
Cpu3  :   0.0% user, 100.0% system,   0.0% nice,   0.0% idle
Mem:   4128648k total,   255272k used,  3873376k free,    70612k buffers
Swap:  2048276k total,        0k used,  2048276k free,    22712k cached
```

图 3.21　LD SERVER 各个 CPU 情况

3.3.3　REAL SERVER 172.19.58.150 的情况

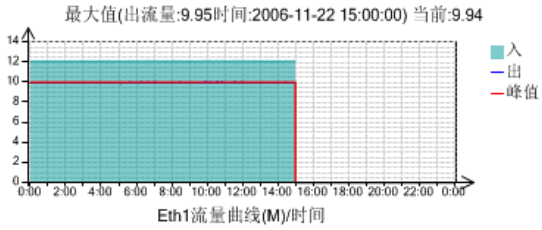


图 3.22　172.19.58.150 的流量图

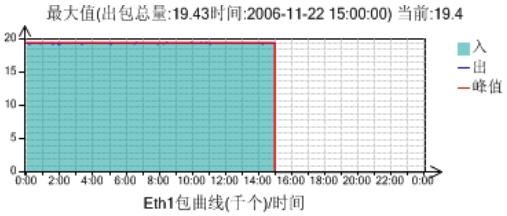


图 3.23　172.19.58.150 的包量图

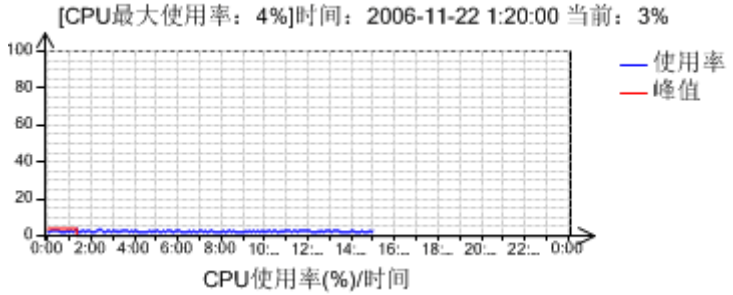


图 3.24　172.19.58.150 的 CPU 使用情况图

3.3.4　REAL SERVER 172.19.58.151 的情况

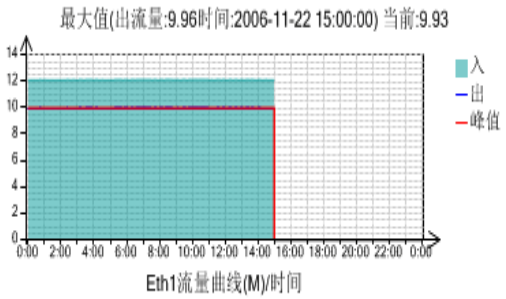


图 3.25 172.19.58.151 的流量图

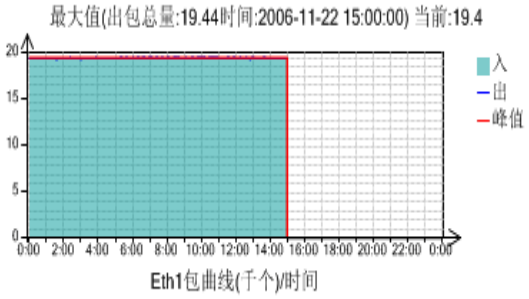


图 3.26　172.19.58.151 的包量图

图 3.27　172.19.58.151 的 CPU 使用情况图

3.3.5　REAL SERVER 172.19.58.152 的情况

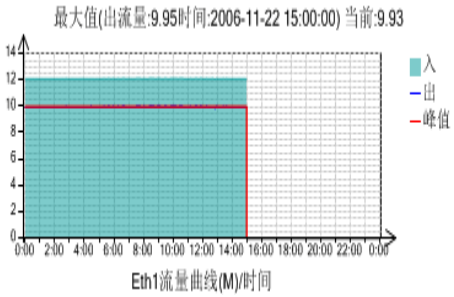


图 3.28 172.19.58.152 的流量图

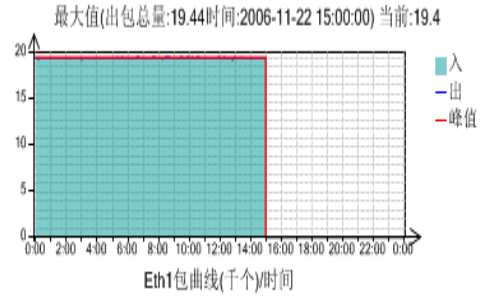


图 3.29 172.19.58.152 的包量图

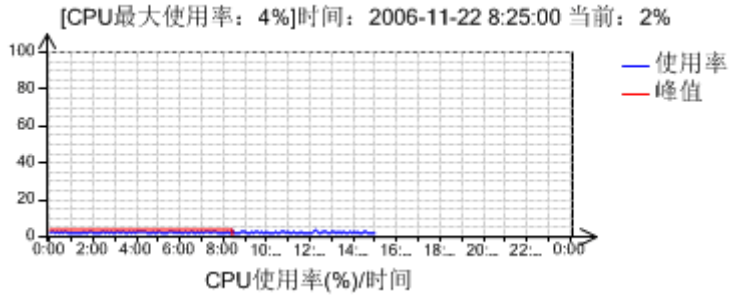


图 3.30 172.19.58.152 的 CPU 使用情况图

3.3.6　REAL SERVER 172.19.58.153 的情况

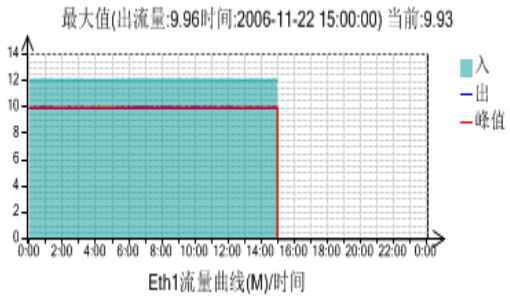


图 3.31　172.19.58.153 的流量图

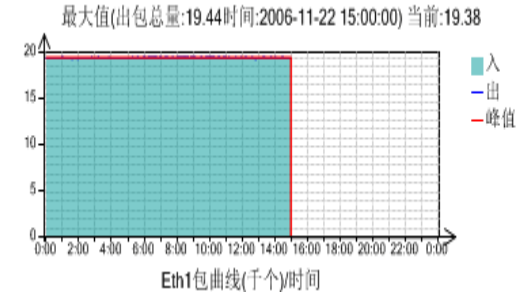


图 3.32　172.19.58.153 的包量图

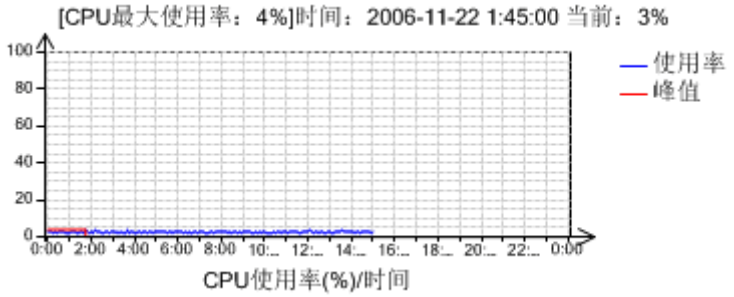


图 3.33 172.19.58.153 的 CPU 使用情况图

3.3.7 REAL SERVER 172.19.58.154 的情况

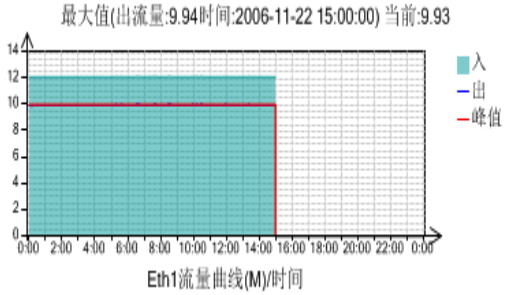


图 3.34 REALSERVER 的流量图

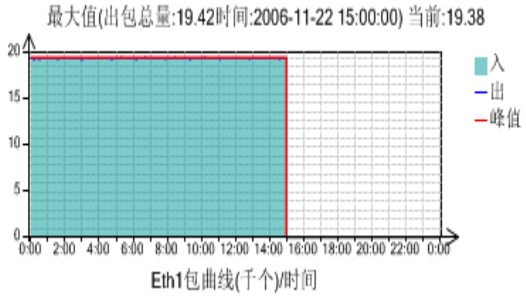


图 3.35 REALSERVER 的包量图

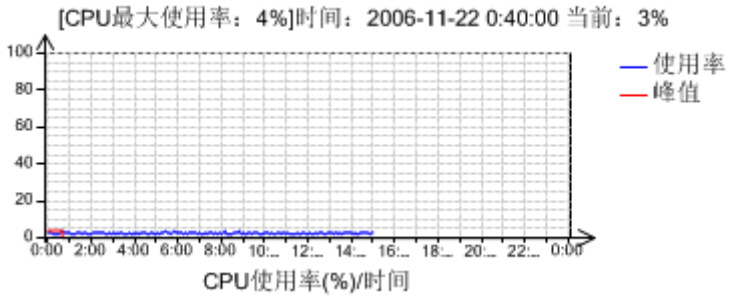


图 3.36 REALSERVER 的 CPU 使用情况图

3.3.8 结论

在百 M 网卡，模式 VS/TUN（内网，UDP）下，可以再次印证到，CPU 是 LD SERVER 的瓶径，172.19.58.145 作为 LD SERVER 所能支持的最大包量是 97K，TCP 方式和 UDP 方式的最大包量差不多是相等的。

第四章 高级话题

4.1 充分利用服务器资源发挥 LVS 性能

由测试报告可以看出，Load Balancer 的 CPU 资源的占用为整个 LVS 系统的瓶颈所在，因此，如何充分使用服务器的 CPU 资源使得 LVS 性能得到最大化，就是一个重要的问题，现就双 CPU 超线程的至强服务器和双 CPU 双核心的至强服务器做一个讨论。

4.1.1 双 CPU 超线程的至强服务器

在 linux 下，对于双 CPU 超线程的服务器来说，只要内核中开启了 CPU 的 HT（超线程）功能，则可通过 cat /proc/cpuinfo 识别到 4 块 CPU，分别为 CPU0；CPU1；CPU2；CPU3。其中，CPU0 和 CPU1 为同一个真实 CPU 的本身和它的超线程出的 CPU。CPU2 和 CPU3 同理。

在 linux 下。我们可以手动的静态配置网卡的 irq 路由，来达到使专门的一块 CPU 完全处理某对应网卡中断的目的。这样可以使得 eth0 网卡和 eth1 网卡的计算量完全分配到 2 块真实 CPU 上，充分发挥出双 CPU 的性能。

具体处理方法如下：

1， cat /proc/cpuinfo | grep processor | wc –l

查看 CPU 处理器个数

2， ETH0_NU=`cat /proc/interrupts | grep eth0 | awk -F ':' '{ print $1 }'``

查看处理 eth0 网卡的 irq 中断号。

3， echo 8 > /proc/irq/$ETH0_NU/smp_affinity

指定 CPU3 来处理 eth0 网卡的中断请求。

4， ETH1_NU=`cat /proc/interrupts | grep eth1 | awk -F ':' '{ print $1 }'``

查看处理 eth1 网卡的 irq 中断号。

5， echo 2 > /proc/irq/$ETH1_NU/smp_affinity

指定 CPU1 来处理 eth1 网卡的中断请求。

4.1.2 双 CPU 双核心的至强服务器

在 linux 下，对于双 CPU 双核心的服务器来说，若内核中没有开启 HT 选项，则默认可以识别到 4 块 CPU。分别为 CPU0；CPU1；CPU2；CPU3。其中 CPU0

和 CPU1 为同一 CPU 的 2 个核。

基于双核至强 CPU 的架构，对于同一物理 CPU 的 2 个核心是共享 4M 二级 Cache 的，因此，将一块网卡的中断对应到同一个物理 CPU 的 2 个核心上，可以在保证二级 Cache 命中率的前提下，充分发挥双核心 CPU 的性能，进一步提高 LD Server 的处理能力。

具体处理方法如下：

1. cat /proc/cpuinfo | grep processor | wc –l

   查看 CPU 处理器个数

2. ETH0_NU=`cat /proc/interrupts | grep eth0 | awk -F ':' '{ print $1 }'``

   查看处理 eth0 网卡的 irq 中断号。

3. echo c > /proc/irq/$ETH0_NU/smp_affinity

   指定 CPU2 和 CPU3 来处理 eth0 网卡的中断请求。

4. ETH1_NU=`cat /proc/interrupts | grep eth1 | awk -F ':' '{ print $1 }'``

   查看处理 eth1 网卡的 irq 中断号。

5. echo 3 > /proc/irq/$ETH1_NU/smp_affinity

   指定 CPU0 和 CPU1 来处理 eth1 网卡的中断请求。

4.2 连接的相关性

在之前的 LVS 配置和使用中，均假设每个连接都相互独立的，所以每个连接被分配到一个服务器，跟过去和现在的分配没有任何关系。但是，有时由于功能或者性能方面的原因，一些来自同一用户的不同连接必须被分配到同一台服务器上。

FTP 是一个因为功能设计导致连接相关性的例子，在 FTP 使用中，客户需要建立一个控制连接与服务器交互命令，建立其他数据连接来传输大量的数据。在主动的 FTP 模式下，客户通知 FTP 服务器它所监听的端口，服务器主动地建立到客户的数据连接，服务器的端口一般为 20。IPVS 调度器可以检查报文的内容，可以获得客户通知 FTP 服务器它所监听的端口，然后在调度器的连接 Hash 表中建立一个相应的连接，这样服务器主动建立的连接可以经过调度器。但是，在被动的 FTP 模式下，服务器告诉客户它所监听的数据端口，服务器被动地等待客户的连接。在 VS/TUN 或 VS/DR 下，IPVS 调度器是在从客户到服务器的

半连接上，服务器将响应报文直接发给客户，IPVS 调度器不可能获得服务器告诉客户它所监听的数据端口。

SSL（Secure Socket Layer）是一个因为性能方面原因导致连接相关性的例子。当一个 SSL 连接请求建立时，一个 SSL 的键值（SSL Key）必须要在服务器和客户进行选择和交换，然后数据的传送都要经过这个键值进行加密，来保证数据的安全性。因为客户和服务器协商和生成 SSL Key 是非常耗时的，所以 SSL 协议在 SSL Key 的生命周期内，以后的连接可以用这个 SSL Key 和服务器交换数据。如果 IPVS 调度器将以后的连接调度到其他服务器，这会导致连接的失败。

在 IPVS 中解决连接相关性的方法是持久服务（Persistent Service）的处理。在 IPVS 中使用两个模板来表示客户和服务器之间的持久服务，模板〈protocol, client_ip, 0, virtual_ip, virtual_port, dest_ip, dest_port〉表示来自同一客户 client_ip 到虚拟服务〈virtual_ip, virtual_port〉的任何连接都会被转发到目标服务器〈dest_ip, dest_port〉，模板〈protocol, client_ip, 0, virtual_ip, 0 dest_ip, 0〉表示来自同一客户 client_ip 到虚拟服务器 virtual_ip 的任何连接都会被转发到目标服务器 dest_ip，前者用于单一的持久服务，后者用于所有端口的持久服务。

当一个客户访问一个持久服务时，IPVS 调度器会在连接 Hash 表中建立一个模板，所以在采用持久服务时，调度器需要记录每个连接的状态，会占用一定的内存空间（每个连接占用 128byte）。这个模板会在一个可设置的时间内过期，如果模板有所控制的连接没有过期，则这个模板不会过期。在这个模板没有过期前，所有来自这个客户到相应服务的任何连接会被发送到同一台服务器。在 ipvsadm 设置时可以指定-p 选项加上超时时间代表调度器实现持久服务（超时时间单位为 s），例如 ipvsadm -A -t 211.1.1.1:80 -p 30。

持久服务还可设置持久的粒度，即可设置将来自一个 C 类地址范围的所有客户请求发送到同一台服务器。这个特征可以保证当使用多个代理服务器的客户访问集群时，所有的连接会被发送到同一服务器。

虽然持久服务可能会导致服务器间轻微的负载不平衡，因为持久服务的一般调度粒度是基于每个客户机的，但是这有效地解决连接相关性问题，如 FTP、

SSL 和 HTTP Cookie 等。

4.3 本地节点

本地结点（Local Node）功能是让调度器本身也能处理请求，在调度时就相当一个本地结点一样，在实现时就是根据配置将部分连接转交给在用户空间的服务进程，由服务进程处理完请求将结果返回给客户。该功能的用处如下：

当集群中服务器结点较少时，如只有三、四个结点，调度器在调度它们时，大部分的 CPU 资源是闲置着，可以利用本地结点功能让调度器也能处理一部分请求，来提高系统资源的利用率。

在分布式服务器中，我们可以利用 IPVS 调度的本地结点功能，在每台服务器上加载 IPVS 调度模块，在一般情况下，利用本地结点功能服务器处理到达的请求，当管理程序发现服务器超载时，管理程序将其他服务器加入调度序列中，将部分请求调度到其他负载较轻的服务器上执行。

在地理上分布的服务器镜像上，镜像服务器利用本地结点功能处理请求，当服务器超载时，服务器通过 VS/TUN 将请求调度到邻近且负载较轻的服务器上。

4.4 Mon 监测程序

在 mon 的 mon.d 目录下有大量的服务检测脚本，但由于 RS 上搭建的服务种类繁多，mon.d 中的脚本不一定能完全满足要求，所以必须编写监测 RS 上服务的脚本和程序。实现针对具体业务的检测脚本非常简单，只需做到如下两点：

1．可以用任何语言编写检测脚本或程序，用监测 RS 上业务时 exit 0 代表 RS 上的业务服务正常，exit 1 代表 RS 上的业务服务不正常。

2．将检测脚本或程序复制到 mon.d 目录下即可，在 mon.cf 中就可以指定

Mon 总是在运行检测脚本完毕后才会运行下一检测脚本，例如 mon.cf 中设定时间间隔为 10s，但是检测脚本运行时间超过 10 秒，那么 mon 不会运行新的检测脚本，所以检测脚本的运行时间需要比 mon 设定的检测间隔时间短。

4.5 系统可用性分析

在 LVS/TUN 模式下，客户端首先向 Load Balancer 发出请求，之后 Load Balancer 将请求报文转给 Real Server，最后 Real Server 将响应报文回复给客户端。在这个过程中，为保证系统的可用性，必须做到：

1，当 Load Balancer 不能正常工作时（譬如遭受恶意或非恶意的故障），必须立刻由另外一台服务器接管 Load Balancer 的 VIP，并且能提供和 Load Balancer 相同的服务。

2，当某一台 Real Server 不能正常提供服务时，Load Balancer 必须立刻收到该消息，在 Real Server 不能正常提供服务的时段内不转发任何客户端的请求，当 Real Server 能恢复正常时，Load Balancer 也必须能收到该消息，将客户端请求负载均衡到该机器上。

在 IPVS+HeartBeat+Mon 系统中，很好的完成了如上两点，利用 HeartBeat 的心跳机制进行主备 Load Balancer 之间的切换，利用 Mon 对 Real Server 进行实时的监测。在 my.qq.com 中进行如下测试很好的论证了这两点：

1，Load Balancer 重启，此时服务遭受中断，但在三秒后服务恢复正常，备用 Load Balancer 将主 Load Balancer 完全接管，此后备用 Load Balancer 的流量和主 Load Balancer 的流量提供服务时的流量相同。

2，三台 Real Server 中某台的 http 服务关闭，八秒后 Load Balancer 探测到该 Real Server 不能正常提供服务（探测间隔可设置，最小为一秒），实时刷新 ipvs 规则，以后的请求将不再导向到该 Real Server。

3，启动第 2 步中那台 Real Server 的 http 服务，八秒后 Load Balancer 探测到该 Real Server 能正常提供服务，实时刷新 ipvs 规则，以后的请求将继续负载均衡到该 Real Server。

4.5 RS 上运行 Squid

对于很多 cache 类的业务采用 squid 能够大大提供系统性能

4.6 系统绑定端口分析

按照第二章的步骤建立 LVS 系统后，利用 netstat 会发现在 0.0.0.0 上绑定了四个端口：

Tcp	0.0.0.0:2583	0.0.0.0:*	LISTEN	23614/perl
Udp	0.0.0.0:32774	0.0.0.0:*		11363/heartbeat: wr
Udp	0.0.0.0:2583	0.0.0.0:*		23614/perl
Udp	0.0.0.0:694	0.0.0.0:*		11363/heartbeat: wr

由上可以得知，由 perl 绑定了一个 TCP 端口处于 listen 状态，还绑定了一个 udp 端口，heartbeat 绑定了两个 UDP 端口。Perl 绑定的两个外网端口可以通过修

改配置文件来解决，在 mon.cf 中加入如下两句即可（可通过 config_mon.sh 自动加入，更新后的 config_mon.sh 已经解决该问题,ipL 为内网 IP）：

```
echo "serverbind = $ipL" > mon.cf
echo "trapbind = $ipL" >> mon.cf
```

heartbeat 配置文件中并没有选项来配置需要绑定的 IP，故需要修改源码解决。但是修改源码有一定风险，所以推荐尽量用 iptables 对端口进行控制。附件中 heartbeat-2.7.0-bindeth1.tar.gz 为只绑定内网 IP 的源码，安装和第二章所述安装一致，只是在配置需要注意，在文件 ha.cf 中的 ucast 选项必须为 ucast eth1 phostip，其中 phostip 一定是另外一台 LD 的内网 IP。

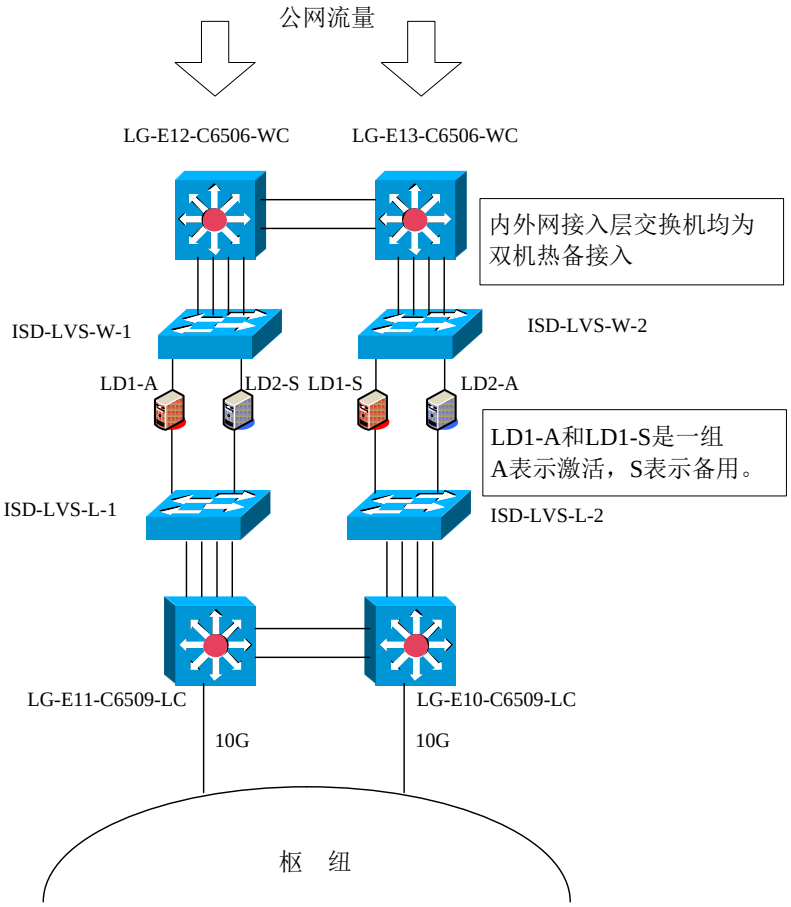
4.7 LD 的网络拓扑及受攻击时的应对措施

4.7.1 LD 的网络拓扑

作为对外服务的 LVS 的前端 LD 服务器，原则上需要部署在专区中，并配置上相应的网络、机架资源环境作为保障。目前对于主—备的 LD 部署方案来讲，比较推荐采用下面的网络拓扑方案：

同组的主 LD Server 和备份 LD Server 分别接入不同的接入层交换机，并上连到不同的核心交换机中，一旦出现内网接入层或单核心故障，可以快速进行切换。

示意图如下：



4.7.2 LD 受攻击时的应对措施

由于 LD 承担着后面所有接入层服务器外网流量的接入任务，且自身不具有 AntiDDOS 功能，一旦受到恶意的攻击，其带来的影响将会是非常严重的。因此，我们必须考虑这样的情况并预先做好应对的防范措施。

现就具体的情况进行一个分析：

（1） 单个或多个 LD 的 VIP 受到攻击

对于整个 LD 专区来说，在其外网接入端，会需要部署一台 AntiDDOS 的黑洞设备，并预先设置好对应的路由及应急切换策略，一旦出现针对某 VIP 的攻击时，则将该 VIP 的所有外网入流量切换到黑洞设备上，过滤后再转回 LD 的 VIP。最大程度上避免 LD 受到影响。

（2） LD 专区所在 IDC 的外网核心受到攻击或 LD 专区所在 IDC 掉电

由于 LD 专区是建立在某一物理的 IDC 中的，一旦出现该物理 IDC 外网核心受攻击或该物理 IDC 掉电的情况，则所有 LD 对应的服务都无可避免的将受到影响。此时，我们只能通过采取 DNS 回退的办法来应急处理。该办法需要满足以下 2 个条件：

A、 预先对所有切换到 LVS 上的 DNS 进行 TTL=1800 或更低的设置。

B、 有维护一个 LD--DNS—IP 对应关系表，一旦出现攻击，可以通过该表作为进行 DNS 回退的依据。

如果以上两条件满足，则受攻击时，服务恢复的时间大概可估算为：

服务受影响时间 = DNS 变更的实施时间 + DNS 变更的完全生效时间

如果操作得当，服务理论上可以在 1 小时左右完全恢复正常。

附录 1 IPVSADM 使用指南

1 名词解释

virtual-service-address（VIP）：虚拟服务器的 ip 地址

real-service-address(RIP)：是指真实服务器的 ip 地址

scheduler：调度方法

2 ipvsadm 的用法和格式如下

ipvsadm -A|E -t|u|f virutal-service-address:port [-s scheduler] [-p[timeout]] [-M netmask]

ipvsadm -D -t|u|f virtual-service-address

ipvsadm -C

ipvsadm -R

ipvsadm -S [-n]

ipvsadm -a|e -t|u|f service-address:port -r real-server-address:port

[-g|i|m] [-w weight]

ipvsadm -d -t|u|f service-address -r server-address

ipvsadm -L|l [options]

ipvsadm -Z [-t|u|f service-address]

ipvsadm --set tcp tcpfin udp

ipvsadm --start-daemon state [--mcast-interface interface]

ipvsadm --stop-daemon

ipvsadm -h

3 命令选项

-A --add-service

在内核的虚拟服务器表中添加一条新的虚拟服务器记录。也就是增加一台新的虚拟服务器。

-E --edit-service

编辑内核虚拟服务器表中的一条虚拟服务器记录。

-D --delete-service

删除内核虚拟服务器表中的一条虚拟服务器记录。

-C --clear

清除内核虚拟服务器表中的所有记录。

-R --restore

恢复虚拟服务器规则

-S --save

保存虚拟服务器规则，输出为-R 选项可读的格式

-a --add-server

在内核虚拟服务器表的一条记录里添加一条新的真实服务器记录。也就是在一个虚拟服务器中增加一台新的真实服务器

-e --edit-server

编辑一条虚拟服务器记录中的某条真实服务器记录

-d --delete-server

删除一条虚拟服务器记录中的某条真实服务器记录

-L|-l --list

显示内核虚拟服务器表，输出对应文件**/proc/net/ip_vs**

-Z --zero

虚拟服务表计数器清零（清空当前的连接数量等）

--set tcp tcpfin udp

设置连接超时值

--start-daemon

启动同步守护进程。他后面可以是 master 或 backup，用来说明 LVS Router 是 master 或是 backup。在这个功能上也可以采用 keepalived 的 VRRP 功能。

--stop-daemon

停止同步守护进程

-h --help

显示帮助信息

其他的选项:：

-t --tcp-service service-address

说明虚拟服务器提供的是 tcp 的服务[vip:port] or [real-server-ip:port]

-u --udp-service service-address

说明虚拟服务器提供的是 udp 的服务[vip:port] or [real-server-ip:port]

-f --fwmark-service fwmark

说明是经过 iptables 标记过的服务类型。

-s --scheduler scheduler

使用的调度算法，有这样几个选项 rr|wrr|lc|wlc|lblc|lblcr|dh|sh|sed|nq。默认的调度算法是：wlc.

-p --persistent [timeout]

持久稳固的服务。这个选项的意思是来自同一个客户的多次请求，将被同一台真实的服务器处理，timeout 的默认值为 300 秒。

-r --real-server server-address

真实的服务器[Real-Server:port]

-g --gatewaying

指定 LVS 的工作模式为直接路由模式（也是 LVS 默认的模式）

-i --ipip

指定 LVS 的工作模式为隧道模式

-m --masquerading

指定 LVS 的工作模式为 NAT 模式

-w --weight weight

真实服务器的权值

--mcast-interface interface

指定组播的同步接口

-c --connection

显示 LVS 目前的连接 如：ipvsadm -L –，输出对应文件**/proc/net/ip_conn**

--timeout

显示 tcp tcpfin udp 的 timeout 值 如：ipvsadm -L --timeout

--daemon

显示同步守护进程状态

--stats

显示统计信息, 输出对应文件**/proc/net/ip_vs_stats**

--rate

显示速率信息，输出对应文件**/proc/net/ip_vs_stats**

--sort

对虚拟服务器和真实服务器排序输出

--numeric -n

输出 IP 地址和端口的数字形式

4 Q&A

a．编译问题

Ipvsadm 必须与内核版本对应。

在 ipvsadm 编译时没有 configure 过程，所以需要把建立/usr/src/linux 与内核 source 的连接。

b．编辑/etc/hosts，加入 RS 的主机名

c．ActiveConn/InActConn (Active/Inactive) connnection 的意义

ActiveConn　ESTABLISHED 状态的连接

InActConn　非 ESTABLISHED 状态的连接（例如 TIME_WAIT、SYN_RECV 等）

在 LVS-NAT 模式下，所有客户端和 RS 的数据交互均通过 LD，LD 能够准确的得出所有客户端的连接数。但在 LVS-DR 和 LVS-TUN 的模式下，RS 到客户端的数据并不通过 LD，当 RS 主动断开连接时，LD 只能从客户端发来的 ACK 信号进行判断，所以 LD 上的 ActiveConn/InActConn 连接数量并不是完全准确的。并且对于 UDP 来说，这两个值是没有多大意义的。

欢迎点击这里的链接进入精彩的Linux公社 网站

Linux公社（www.Linuxidc.com）于2006年9月25日注册并开通网站，Linux现在已经成为一种广受关注和支持的一种操作系统，IDC是互联网数据中心，LinuxIDC就是关于Linux的数据中心。

Linux公社是专业的Linux系统门户网站，实时发布最新Linux资讯，包括Linux、Ubuntu、Fedora、RedHat、红旗Linux、Linux教程、Linux认证、SUSE Linux、Android、Oracle、Hadoop、CentOS、MySQL、Apache、Nginx、Tomcat、Python、Java、C语言、OpenStack、集群等技术。

Linux公社（LinuxIDC.com）设置了有一定影响力的Linux专题栏目。

Linux公社 主站网址：www.linuxidc.com 旗下网站：www.linuxidc.net

包括：**Ubuntu 专题** **Fedora 专题** **Android 专题** **Oracle 专题** **Hadoop 专题** **RedHat 专题** **SUSE 专题** **红旗 Linux 专题** **CentOS 专题**

Linux 公社微信公众号：linuxidc_com