



Linux 实用技术阅读手册

BENET 网络工程师认证课程 第二学期

北京阿博泰克北大青鸟信息技术有限公司 编著

2009 年 08 月 V1.0 版

【扉页】

目 录

目 录.....	3
第一章 构建本地 YUM 源服务器.....	6
1.1 YUM 概述.....	7
1.2 构建本地 YUM 源服务器.....	7
1.2.1 为什么要构建本地源.....	7
1.2.2 如何构建本地源.....	7
1.3 使用 yum 工具安装软件.....	9
1.3.1 设置 YUM 源位置.....	9
1.3.2 使用 yum 命令.....	10
1.3.3 使用图形工具“添加删除程序”.....	11
第二章 sed 和 awk 工具的应用.....	12
2.1 正则表达式.....	13
1. 元字符.....	13
2. 正则表达式应用示例.....	13
2.2 sed 工具.....	14
2.2.1 sed 工作原理.....	14
2.2.2 sed 的调用方式.....	14
2.2.3 sed 选项.....	15
2.2.4 sed 编辑命令.....	15
2.2.5 sed 应用举例.....	15
2.3 awk 工具.....	18
2.3.1 awk 工作原理.....	18
2.3.2 awk 调用方式.....	19
2.3.3 awk 应用举例.....	20
第三章 Snort 网络入侵检测系统.....	22
3.1 IDS 介绍.....	23
3.2 Snort 介绍.....	23
3.3 构建 Snort+Base NIDS 系统.....	25
3.3.1 系统结构.....	25
3.3.2 安装前的准备工作.....	25
3.3.3 安装 Snort.....	26
3.3.4 设置报警信息记录到 MySQL 数据库.....	28
3.3.5 构建安全图形分析引擎.....	29
3.4 简单的测试.....	31
第四章 构建 LVS 负载均衡系统.....	33
4.1 Linux 集群系统概述.....	34
4.1.1 常见集群系统分类.....	34

目 录

4.1.2 Linux Virtual Server 项目	34
4.1.3 LVS 体系结构介绍	35
4.2 LVS 的负载均衡模型及算法	36
4.2.1 LVS 负载均衡模型	36
4.2.2 三种模型比较	39
4.2.3 LVS 负载调度算法	40
4.2.4 IPVS-DR +heartbeat+ldirectord构建高可用负载均衡集群方案	40
4.3 LVS 应用案例	41
4.3.1 基于直接路由模式 (DR) 的 LVS	41
4.3.2 IPVS-DR +heartbeat+ldirectord构建高可用负载均衡集群	45
参考文献:	53
第五章 Linux 中的 XEN 虚拟化技术	54
5.1 Xen 虚拟化概述	55
5.2 Xen 的安装和配置	55
5.2.1 安装 Xen	55
5.2.2 配置 Xen 启动	56
5.2.3 Xen 服务控制命令	56
5.3 创建 Xen 虚拟系统	57
5.3.1 创建 Xen 虚拟系统安装树	57
5.3.2 使用字符工具 virt-install 创建 Xen 虚拟系统	57
5.3.3 使用图形工具 virt-manager 创建 Xen 虚拟系统	59
5.4 管理 Xen 虚拟系统	62
5.4.1 使用 xm 命令管理 Xen 虚拟机	62
5.4.2 使用 virt-manager 图形工具管理 Xen 虚拟机	63
5.4.3 配置虚拟系统随服务器启动	64
第六章 使用 rsync 远程文件同步	66
6.1 Rsync 概述	67
6.2 配置用于 rsync 同步的远程主机	67
6.2.1 基于 OpenSSH 的模式	67
6.2.2 基于 rsync 的--daemon 模式	67
6.3 使用 rsync 文件同步工具	69
6.3.1 rsync 命令的常见用法示例	70
6.3.2 结合 crond 实现定期同步 (下行)	71
6.3.3 结合 inotify-tools 实现实时同步 (上行)	72
第七章 Nginx 网站及负载均衡系统	75
7.1 Nginx 概述	76
7.2 Nginx 的安装及运行控制	76
7.2.1 Nginx 的安装	76
7.2.2 Nginx 运行控制	77
7.3 Nginx 网站服务配置示例	79
7.3.1 常规静态网站配置	79
7.3.2 添加 nginx 状态统计	80

7.3.3	通过 FastCGI 方式支持 PHP 页面.....	81
7.3.4	基于域名的虚拟主机配置.....	82
7.3.5	基于反向代理的负载均衡配置.....	83
第八章	基于 Linux 的 OpenVPN 网络.....	85
8.1	OpenVPN 概述.....	86
8.2	OpenVPN 的安装及运行控制.....	86
8.2.1	在 RHEL5 系统中的安装.....	86
8.2.2	在 Windows XP 系统中的安装.....	86
8.2.3	OpenVPN 服务的运行控制.....	87
8.3	OpenVPN 网络架构应用实例.....	87
8.3.1	案例需求分析.....	87
8.3.2	配置 GW1 <----> GW2 隧道连接.....	88
8.3.3	配置 “GW1 <----> PC1” 隧道连接.....	98
第九章	构建软 RAID 磁盘阵列.....	106
9.1	RAID 概述.....	107
9.2	构建使用软 RAID 磁盘阵列.....	107
9.2.1	准备用于 RAID 阵列的分区.....	107
9.2.2	创建 RAID 设备.....	109
9.2.3	在 RAID 设备中建立文件系统.....	110
9.2.4	挂载并使用文件系统.....	110
9.3	RAID 阵列的管理及设备恢复.....	111
9.3.1	基本管理操作.....	111
9.3.2	设备恢复操作.....	112



四
四
之
四
十

1

构建本地 YUM 源服务器

3
.
0

1.1 YUM 概述

YUM 是 YUP (Yellow dog Updater, 用于 Yellowdog Linux 的软件更新器) 工具的改进版, 最初由 TSS 公司 (Terra Soft Solutions, INC.) 使用 python 语言开发而成, 后来由杜克大学的 Linux 开发队伍进行改进, 命名为 YUM (Yellow dog Updater, Modified)。YUM 主要用于自动升级、安装/移除 rpm 软件包, 它能够自动查找并解决 rpm 包之间的依赖关系, 而无需管理员逐个、手工的去安装每一个 rpm 包, 使管理员在维护大量的 Linux 主机时更加轻松自如。

相关网站可参考如下:

<http://www.terrasoftsolutions.com/>

<http://wiki.linux.duke.edu/YumFaq>

要成功使用 YUM 工具更新系统和软件, 需要有一个包含各种 rpm 软件包的 repository (软件仓库), 提供软件仓库的服务器习惯上称为“源”服务器。软件仓库可以基于 HTTP、FTP 协议或者本地文件目录提供服务, 并收集目录中所有 rpm 包的 header (头部) 信息组成 repodata (仓库数据), 以供 YUM 客户端工具查询分析。

1.2 构建本地 YUM 源服务器

1.2.1 为什么要构建本地源

在 Linux 主机中使用 YUM 工具在线升级、安装软件时, 往往受到网络连接速度、带宽的限制, 导致软件安装耗时过长甚至失败。特别是当有大量服务器、大量软件包需要升级时, 更新的缓慢程度可能令人难以忍受。

相比较而言, 本地 YUM 源服务器最大的优点在于局域网的快速网络连接和稳定性。有了局域网中的 YUM 源服务器, 即便在 Internet 连接中断的情况下, 也不会影响其他 YUM 客户端的软件升级和安装。

1.2.2 如何构建本地源

通过从 DVD 光盘、YUM 客户端缓存目录 (/var/cache/yum/) 以及 Internet 下载等途径获取需要的 rpm 包, 可以组建基于本地文件系统的 rpm 软件包仓库。进一步结合使用 HTTP 或 FTP 协议的服务软件, 即可构建一个位于本地局域网的高速 YUM“源”服务器。

本例中使用 RHEL5 安装光盘 (DVD) 中的软件包组成软件仓库, 并结合 VSFTPD 服务, 提供基于 FTP 方式的简易 YUM 源。实现的主要步骤如下。

1. 安装 vsftpd、createrepo 软件包

createrepo 工具主要用于收集目录中 rpm 包文件的头信息, 以创建 repodata 软件仓库数据 (经 gzip 压缩的 xml 文件)。

1

构建本地 YUM 源服务器

```
[root@yumserver ~]# mkdir -p /media/cdrom
[root@yumserver ~]# mount /dev/cdrom /media/cdrom
mount: block device /dev/cdrom is write-protected, mounting read-only
[root@yumserver ~]# cd /media/cdrom/Server/
[root@yumserver Server]# rpm -ivh createrepo-0.4.4-2.fc6.noarch.rpm
vsftpd-2.0.5-10.el5.i386.rpm
warning: createrepo-0.4.4-2.fc6.noarch.rpm: Header V3 DSA signature: NOKEY, key ID
37017186
Preparing... ##### [100%]
  1:vsftpd ##### [ 50%]
  2:createrepo #####
[100%]
```

2. 准备软件库目录

注意要保证/var/ftp/rhel5 目录有足够的可用空间（在本例中最好大于 3GB），以便存放复制的软件包。必要时可以使用单独的硬盘分区（挂载到/var/ftp/rhel5 目录中）。

```
[root@yumserver Server]# mkdir -p /var/ftp/rhel5/
[root@yumserver Server]# cp -prf /media/cdrom/* /var/ftp/rhel5/
```

查看复制好的目录结构：Cluster、ClusterStorage、Server、VT——主要是光盘中包含 rpm 软件包的几个文件夹）。

```
[root@yumserver Server]# ls -lh /var/ftp/rhel5/ | grep ^d
drwxr-xr-x 3 root root 6.0K 2007-03-18 Cluster
drwxr-xr-x 3 root root 6.0K 2007-03-18 ClusterStorage
drwxr-xr-x 4 root root 2.0K 2007-03-18 images
drwxr-xr-x 2 root root 2.0K 2007-03-18 isolinux
drwxr-xr-x 3 root root 330K 2007-03-18 Server
drwxr-xr-x 3 root root 6.0K 2007-03-18 VT
```

3. 创建 repository 仓库信息文件

在各个软件包目录中分别执行 createrepo 命令，生成当前目录下的 repodata 数据。使用 -g 选项可以指定用于创建组信息的 xml 文件模板。

```
[root@yumserver ~]# cd /var/ftp/rhel5/Cluster/
[root@yumserver Cluster]# createrepo -g repodata/comps-rhel5-cluster.xml ./

[root@yumserver Cluster]# cd /var/ftp/rhel5/ClusterStorage/
[root@yumserver ClusterStorage]# createrepo -g repodata/comps-rhel5-cluster-st.xml ./
```



```
[root@yumserver ClusterStorage]# cd /var/ftp/rhel5/Server/
[root@yumserver Server]# createrepo -g repodata/comps-rhel5-server-core.xml ./

[root@yumserver Server]# cd /var/ftp/rhel5/VT/
[root@yumserver VT]# createrepo -g repodata/comps-rhel5-vt.xml ./

[root@yumserver VT]# rm -rf /var/ftp/rhel5/*.olddata/           //清除旧的数据文件
```

4. 配置启动 vsftpd 服务（开启默认的匿名 FTP 服务即可）

```
[root@yumserver ~]# chkconfig --level 2345 vsftpd on
[root@yumserver ~]# service vsftpd start
```

若 FTP 服务无法访问，请参考以下配置重新启动 vsftpd 服务：

```
[root@yumserver ~]# vi /etc/vsftpd/vsftpd.conf
anonymous_enable=yes
local_enable=NO
write_enable=no
dirmessage_enable=YES
xferlog_enable=YES
connect_from_port_20=YES
pasv_enable=YES
pasv_max_port=3200
pasv_min_port=3100
xferlog_std_format=YES
listen=YES
listen_address=192.168.0.11           //服务器的 IP 地址
pam_service_name=vsftpd
tcp_wrappers=yes
[root@yumserver ~]# service vsftpd restart
```

1.3 使用 yum 工具安装软件

最常用的 YUM 客户端工具是字符模式下的 yum 命令，在 RHEL5 系统中由默认安装的 yum-3.0.1-5.el5 软件包提供。RHEL5 系统还提供了一个图形 YUM 工具 pirut，该工具只是 yum 工具的一个 X 图形前端。

1.3.1 设置 YUM 源位置

使用 YUM 源服务器之前，必须为客户端建立指定的配置文件，设置好源服务器的位置和可用目录等选项。



1

构建本地 YUM 源服务器

```
[root@client ~]# cd /etc/yum.repos.d/
[root@client yum.repos.d]# vi rhel5-pkgs-yumsvr.repo //新建配置文件，名称自定
[Cluster]
name=Cluster Directory
baseurl=ftp://192.168.0.11/rhel5/Cluster
enabled=1 //启用该目录
gpgcheck=0 //不检查 gpg key

[ClusterStorage]
name=ClusterStorage Directory
baseurl=ftp://192.168.0.11/rhel5/ClusterStorage
enabled=1
gpgcheck=0

[Server]
name=Server Directory
baseurl=ftp://192.168.0.11/rhel5/Server
enabled=1
gpgcheck=0

[VT]
name=VT Directory
baseurl=ftp://192.168.0.11/rhel5/VT
enabled=1
gpgcheck=0

[root@client yum.repos.d]# yum clean all //更新缓存
```

1.3.2 使用 yum 命令

1. 列表查看软件包信息

可以使用 **list** 选项查看软件包相关的各项信息。

```
[root@client ~]# yum list updates //查看有哪些可用于升级的软件包
[root@client ~]# yum list installed //查看本机已安装的软件包
[root@client ~]# yum list available //查看 yum 源中所有可用的软件包
[root@client ~]# yum list available lynx* //查看 yum 源中以 lynx 开头的软件包
[root@client ~]# yum info installed bind //查看已安装的 bind 软件包的信息
```

2. 软件包升级、卸载、安装

升级（**update**）、卸载（**remove**）、安装（**install**）软件包时，系统会自动检查并解决软件包之间的依赖关系。开始各项操作前，系统会提示用户按 **y** 键进行确认（如果希望系统自动回



答为 **y**，可以在 **yum** 命令后增加 **-y** 选项）。

```
[root@client ~]# yum -y update           //升级所有可用的软件包， 由系统自动确认
[root@client ~]# yum update bind         //升级 bind 软件包
[root@client ~]# yum remove net-snmp-utils //卸载 net-snmp-utils 软件包
[root@client ~]# yum install lynx        //安装 lynx 软件包
```

1.3.3 使用图形工具“添加删除程序”

在 RHEL5 系统中，进入 X-Windows 图形模式以后，可以使用系统自带的“添加删除程序”工具（**pirut**）进行软件包的升级、安装、卸载等管理。

点击GNOME面板菜单组“应用程序”中的“添加删除程序”，或者运行“**pirut**”就可以打开该工具（如图 1.1所示）。运行RHEL5 提供的系统配置链接“**system-config-packages**”也可以打开该工具。

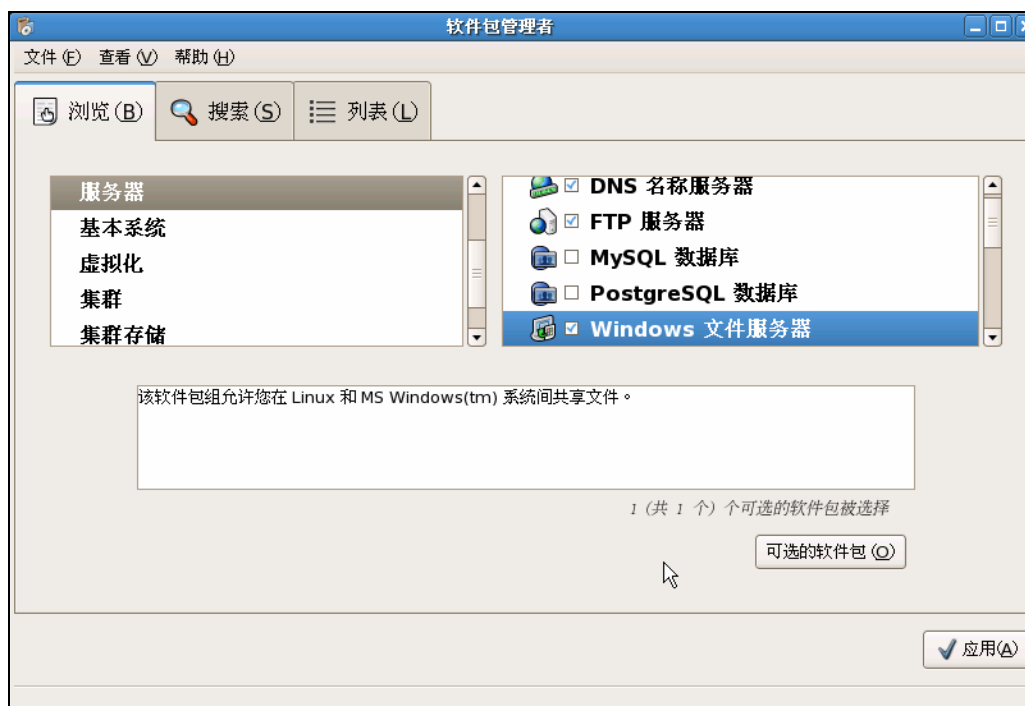


图1.1 使用 **pirut** 工具“添加删除程序”

使用图形工具来管理软件包，要更加直观、形象。具体使用在这里不再赘述。

四
四
之
四
十

2

sed 和 awk 工具的应用

3
.
0

sed 和 awk 是 shell 编程中经常用到的工具。sed 是一个非交互式的流编辑器 (stream editor)，用于对文件执行一系列的编辑操作。awk 是它的三个开发者 Aho、Weinberger 和 Kernighan 名字首字母的缩写，它是一种编程语言，用来处理结构化的数据和生成格式化的报告。

2.1 正则表达式

在介绍 sed 和 awk 之前，首先需要介绍正则表达式的使用。

正则表达式 (regular expression) 是描述一种字符串匹配的模式，常用来检查一个字符串或文件中是否含有特定的字符串，以便对其进行替换、删除、插入等操作。

很多编程语言都支持正则表达式，如：php、perl、python、java 等。在大多数编程语言中，正则表达式都被括在两个正斜杠 (/) 之间，典型的形式是：

/pattern/
pattern 是由普通字符 (数字、字母等) 以及特殊字符 (元字符) 组成的字符串模式。

1. 元字符

元字符是用于正则表达式中的一类特殊的字符，最常见的元字符是“*” (如表 2.1 所示)。

表 2.1 常用元字符

元字符	功能
^	只匹配行首
\$	只匹配行尾
*	一个单字符后紧跟*，匹配 0 个或多个此单字符
[]	匹配一组字符串中的一个
\	转义字符
.	配任意单字符
.*	配任意字符
pattern\{n\}	用来匹配前面 pattern 出现的次数，n 为次数
pattern\{n , \}	含义同上，但次数最少为 n
pattern\{n , m\}	含义同上，但 pattern 出现次数在 n 与 m 之间

2. 正则表达式应用示例

提取以“#”为标记的注释行：

```
/^#/
```

2

sed 和 awk 工具的应用

提取空白行:

```
/^$/
```

提取文本中的 IP 地址:

```
/[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}/
```

提取文本中的电话号码:

```
/[0-9]{3,4}-[0-9]{7,8}/
```

2.2 sed 工具

sed 是一个非交互式的编辑器，在使用中必须指定行号或正则表达式。它的使用比较复杂，sed 常用来查找替换文件或字符串中特定信息的程序。本节只对其进行简单的应用介绍。

2.2.1 sed 工作原理

sed 每次只从文件或标准输入中读取一行数据，将之拷贝到一个编辑缓冲区，然后对该行数据按指定命令进行处理，并将结果输出到屏幕（标准输出），接着读入下一行。整个文件像流水一样被逐行处理然后逐行输出，所以说 sed 是一个流编辑器。

sed 默认并不直接修改文件，它只是操作编辑缓冲区内的信息，编辑完的信息默认在屏幕显示（如图 2.1 所示），如有需要可采用重定向的方式导出到指定文件（或者需要结合“-i”选项直接在文件中进行修改）。

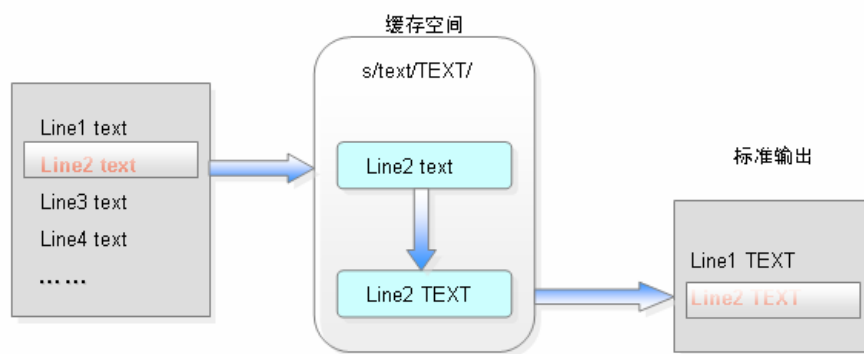


图 2.1 sed 命令工具的一般处理流程

2.2.2 sed 的调用方式

sed 的调用有三种方式:

1. 在命令行键入命令

```
sed [选项] sed 编辑命令 输入文件  
shell 命令 | sed [选项] sed 编辑命令
```

2. 将 sed 命令写入脚本文件，然后调用该脚本文件

```
sed [选项] -f sed 脚本文件 输入文件
```

3. 将 sed 命令插入脚本文件，并使 sed 脚本可执行

```
sed 脚本文件 [选项] 输入文件
```

第一种调用方式相对较简单，也最常用。

2.2.3 sed 选项

sed 常用的选项有“-n”、“-e”、“-f”

- -n 取消缺省输出
- -e 当需要执行多个编辑命令时需要使用-e 选项
- -f 从脚本文件中读取内容并执行
- -i 直接在指定的文件中进行替换操作，而不是输出到屏幕

2.2.4 sed 编辑命令

编辑命令是 sed 的关键功能（如表 2.2 所示），通过使用编辑命令 sed 可以完成插入、删除、替换等编辑功能。常用的编辑命令：

表 2.2 sed 的常用编辑命令

命令	说明
p	打印匹配行
=	显示文件行号
d	删除指定行
s	字符串替换

2.2.5 sed 应用举例

1. 根据行号打印指定行信息

打印指定行信息时需要使用“-n”选项，否则 sed 会将缓冲区中所有（编辑的和未编辑的）信息都在标准输出中显示，同时还需要指定打印的行号（一行）或者指定打印行范围（多行）。命令格式：

```
sed -n [行号 1[,行号 2]]p 输入文件
```

打印/etc/passwd 文件中第 3 行信息



2

sed 和 awk 工具的应用

```
[root@localhost ~]# sed -n '3p' /etc/passwd
daemon:x:2:2:daemon:/sbin:/sbin/nologin
```

打印/etc/passwd 最后一行信息

```
[root@localhost ~]# sed -n '$p' /etc/passwd      //$表示最后一行
hacluster:x:24:24:heartbeat processes:/var/lib/heartbeat/cores/hacluster:/bin/bash
```

打印/etc/passwd 文件中第 3 到第 5 行信息，

```
[root@localhost ~]# sed -n '3,5p' /etc/passwd
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
```

2. 根据模式匹配查找打印指定行

sed 支持模式“/pattern/”格式进行查找，这里 pattern 可以是精确的字符串，也可以是一个正则表达式。

查找/etc/passwd 中包含“root”的行，并打印

```
[root@localhost ~]# sed -n '/root/p' /etc/passwd
root:x:0:0:root:/root:/bin/bash
operator:x:11:0:operator:/root:/sbin/nologin
```

查找 apache 日志文件/var/log/httpd/access_log，将访问来源地址是 10/20/30.*.*的行输出到 http_statistic 文件中，为进一步分析做准备。

```
[root@localhost ~]# sed -n '/^[1-3]0./p' /var/log/httpd/access_log > http_statistic
```

- ^ : 指定位于行首
- [1-3] : 说明可以是 1, 2, 3 中任意一个

3. 删除

sed 编辑命令“d”实现删除的功能，删除操作可根据行号也可根据模式匹配进行
根据行号删除的命令格式：

```
sed [行号 1[,行号 2]]d 输入文件
```

删除/etc/passwd 文件第 1 行：

```
[root@localhost ~]# sed '1d' /etc/passwd
```

删除/etc/passwd 第 1 至 5 行：


```
[root@localhost ~]# sed '1,5d' /etc/passwd
```

根据模式匹配删除的命令格式：

```
sed /pattern-to-find/d 输入文件
```

将 Apache 配置文件中的所有注释行和空行删除

```
[root@localhost ~]# sed -e /^#/d -e /^$/d /etc/httpd/conf/httpd.conf
```

- -e 执行多个编辑命令需要在编辑命令前加上-e
- ^# 行首为#
- ^\$ 空行

4. 替换

替换操作是 sed 命令使用频率最高的一个功能，实现替换操作需要 sed 编辑命令“s”和替换选项“g”、“p”、“w”组合完成，一般的命令格式是：

```
sed [-n] [行号 1[,行号 2]]s/pattern-to-find/replacement-pattern/[g,p,w] 输入文件
```

s 编辑命令通知 sed 这是一个替换操作，sed 将查询到的 pattern-to-find，用 replacement-pattern 进行替换。

替换选项的作用：

- g : sed 缺省只替换第一次出现的模式，使用 g 选项替换所有行全局出现的模式。如果行内有多个符合的字符串，需加上 g 选项，否则将只有第一个字符串被替换。
- p : 替换选项“p”和 sed 选项“-n”组合使用只显示被执行替换操作的行
- w : 将执行替换操作的行输出到指定文件

将/etc/passwd 中的/bin/bash 全部替换成/bin/tcsh

```
[root@localhost ~]# sed 's/VbinVbash/VbinVtcsh/g' /etc/passwd
```

只显示被替换的行

```
[root@localhost ~]# sed -n 's/VbinVbash/VbinVtcsh/gp' /etc/passwd
```

只替换 1 到 10 行中的数据

```
[root@localhost ~]# sed '1,10s/VbinVbash/VbinVtcsh/g' /etc/passwd
```

将被执行替换操作的行输出到 sed.out 文件

```
[root@localhost ~]# sed 's/VbinVbash/VbinVtcsh/gw sed.out' /etc/passwd
```



2

sed 和 awk 工具的应用

2.3 awk 工具

awk 提供了极其强大的功能：它可以完成 **grep** 和 **sed** 所能完成的工作，同时，它还可以进行样式装入、流程控制、数学运算符、进程控制语句甚至于使用内置的变量和函数。**awk** 的功能强大，使用比较复杂，本小结对 **awk** 的功能不做深入介绍，只介绍一些基本常用功能。

2.3.1 awk 工作原理

在 Linux 系统中，**/etc/passwd** 是一个非常典型的格式化文件，各字段间使用“:”作为分隔符隔开，Linux 系统中的大部分日志文件也是格式化文件，处理这些文件从中提取相关信息是管理员的日常工作内容之一，有 **awk** 的帮助这些工作变得轻而易举。

```
[root@localhost ~]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
... ..
```

若需要查找并打印出 **/etc/passwd** 的用户名、用户 ID、组 ID，使用下面的 **awk** 命令就可完成：

```
[root@localhost ~]# awk -F ':' '{print $1,$3,$4}' /etc/passwd
```

awk 从文件或者标准输入读入信息，与 **sed** 一样信息的读入也是逐行读取的，处理过程与 **sed** 类似。不同的是 **awk** 将文本文件中的一行视为一个记录，而将一行中的某一部分作为记录中的一个字段。为了操作这些不同的字段，**awk** 借用 **shell** 的方法，用 **\$1**，**\$2**，**\$3**... 这样的方式来顺序地表示行（记录）中的不同字段。特殊地，**awk** 用 **\$0** 表示整个行（记录）。不同的字段之间是用称作分隔符的字符分隔开的。系统默认的分隔符是空格。**awk** 允许在命令行中用 **[-F 分隔符]** 的形式来改变这个分隔符。

在上述示例中，**awk** 命令对 **/etc/passwd** 的处理过程如图 2.2 所示。

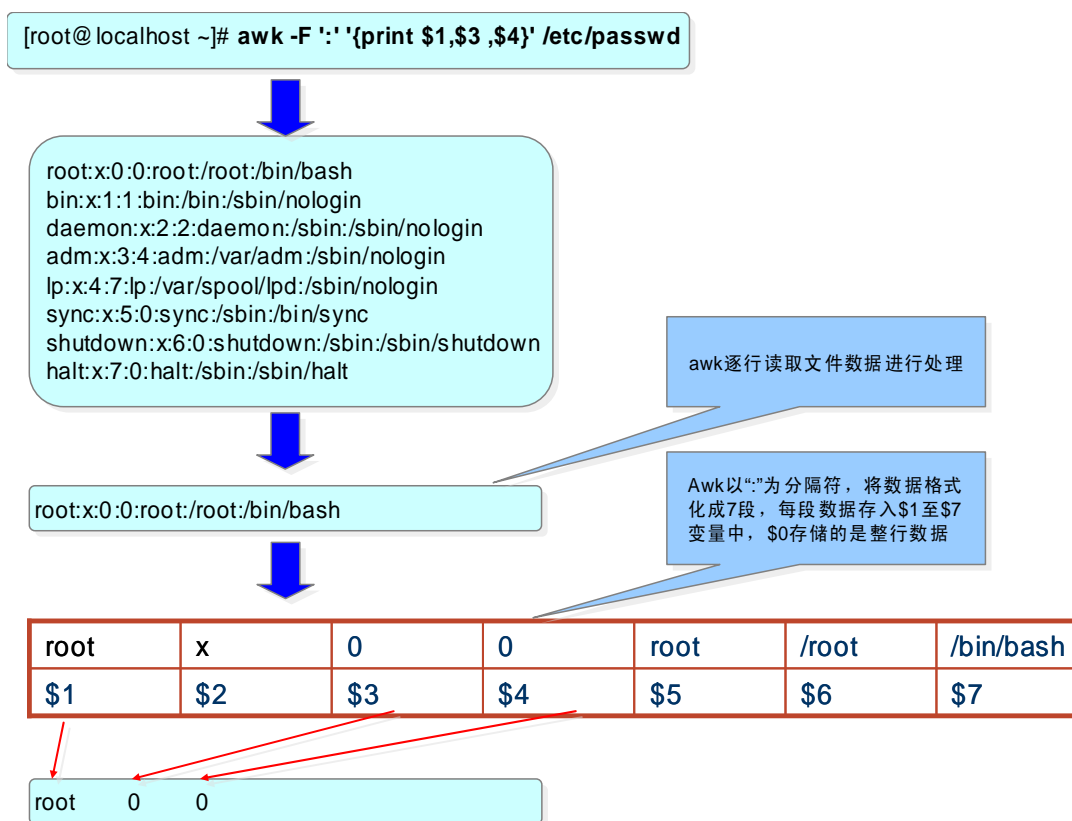


图 2.2 awk 命令的简要处理流程

2.3.2 awk 调用方式

与 sed 一样 awk 也有三种方式调用方式：

1. 命令行键入方式，这也是最常用的方式。命令格式为：

```
awk [-F 域分隔符] awk 指令 输入文件
shell 命令 | awk [-F 域分隔符] awk 指令
```

其中的 **[-F 域分隔符]** 为可选，awk 使用空格作为缺省的域分隔符，因此如果要浏览域间有空格的文本，不必指定这个选项，但如果要浏览诸如 passwd 文件，此文件各域以冒号为分隔符，则必须指明 -F 选项。

awk 指令 由 pattern（模式）和 action（动作）或是两者的组合组成，常见的形式是：'/pattern/{action}'，awk 指令必须放到单引号中。pattern 是正则表达式、判断条件真伪的表达式或两者的组合构成，多个 pattern 间使用“,”分隔。action 是 awk 所要采取的动作，由 awk 语句组成，多个 awk 语句间使用“;”进行分隔

2. 编写 awk 运行脚本

是将所有 awk 命令插入一个文件，并使用 awk 程序执行，然后用 awk 命令解释器作为脚本的首行，通过键入脚本名称来调用。

2

sed 和 awk 工具的应用

3. 将所有的 **awk** 命令插入一个单独文件，然后调用：

awk -f awk-script-file 输入文件

- “-f” 选项指明写有 **awk** 命令的文件名，如 **awk_script_file**
- “输入文件” 需要使用 **awk** 进行处理的文件名。

2.3.3 awk 应用举例

1. 模式查找打印

查找/etc/passwd 文件中包含“nologin”的行

```
[root@localhost ~]# awk '/nologin/' /etc/passwd
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
.....
```

2. 模式查找打印特定域

查找/etc/passwd 文件中包含“nologin”的行，以“:”为分隔符，只打印第一个字段

```
[root@localhost ~]# awk -F':' '/nologin/{print $1}' /etc/passwd
bin
daemon
adm
.....
```

查找/etc/passwd 文件中包含“nologin”的行，以“:”为分隔符，只打印第一、二字段

```
[root@localhost log]# awk -F':' '/nologin/{print $1,$2}' /etc/passwd
bin x
daemon x
adm x
.....
```

查找/etc/passwd 文件中包含“nologin”的行，以“:”为分隔符，只打印第一和第二个字段，并打印所在行的行号。

```
[root@localhost log]# awk -F':' '/nologin/{print NR,$1,$2}' /etc/passwd
2 bin x
3 daemon x
4 adm x
.....
```

NR 是 **awk** 的内部变量，记录的是已读入的记录数



3. 使用比较运算符

通过检查/etc/passwd 文件的第三个字段，检查有无特权用户：

```
[root@localhost ~]# awk -F: '$3==0{print $1}' /etc/passwd
```

当第 3 个字段等于 0 时打印改行的第一个字段

检查有无空口令用户

```
[root@localhost ~]# awk -F: '$2==" "{print $1}' /etc/passwd
```

4. 使用内部变量

awk 自身提供了很多内部变量，例如变量 NR：每条记录的记录号保存在内置变量 NR 中，每处理完一条记录，NR 的值加一。

显示/etc/passwd 文件中第 7 行到第 15 行中以字符 “:” 分隔的第 1 字段，第 3 字段和第 7 字段：

```
[root@localhost ~]# awk -F: 'NR==7,NR==15 {print $1,$3,$7}' /etc/passwd
```

5. 使用内部函数

awk 内置定义了很多函数，每个函数实现不同的功能，前面用到的 print 即为一个函数，而 length 函数用来计算字符串包含的字符数。可以利用其实现一个简单的功能——检查有无空口令用户。

```
[root@localhost ~]# awk -F: 'length($2)==0 {print $1}' /etc/passwd
```



网络安全

3.0

3

Snort 网络入侵检测系统

3.1 IDS 介绍

IDS(Intrusion Detection System, 入侵检测系统)是网络安全体系一个重要组成部分。它的作用是监控网络或主机是否被入侵。按监测对象的不同,IDS 分为两类:监控主机的 HIDS 和监控网络的 NIDS。

NIDS 扫描当前网络的活动,监视和记录网络的数据,根据定义好的规则来过滤网络上的数据,检查网络或系统中是否存在违反安全策略的行为和被攻击的迹象,当入侵发生时提供实时报警。

NIDS 一般由两部分组成:网络流量传感器和入侵检测系统控制台。网络流量传感器一般被部署在网络中心的核心交换机或部门交换机的镜像端口上(如图 3.1 所示)。在网络安全管理员的工作站上,通过入侵检测系统控制台来接收、分析网络传感器传来的日志来做报警处理。也可以将网络流量传感器和入侵检测系统控制台集成在同一台主机上同时进行检测和分析的工作。

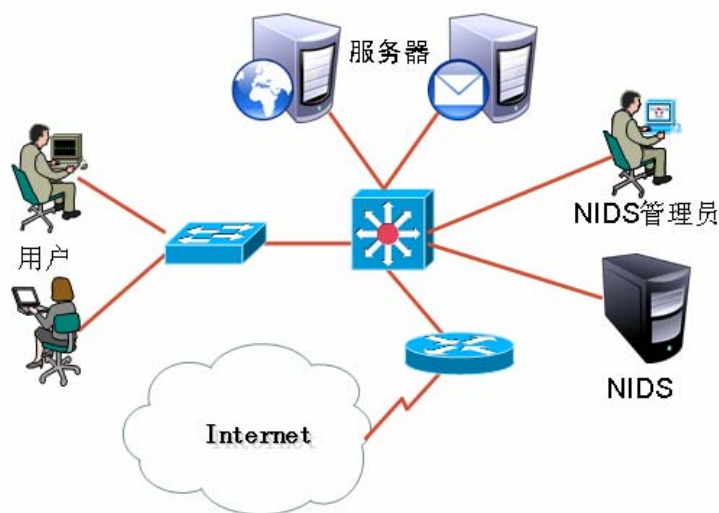


图 3.1 snort 入侵检测体系结构

安全市场成熟的商业 NIDS 产品很多,例如安氏中国的 LinkTrust NIDS Giga、思科的 IDSM-2 入侵检测模块、McAfee 的 Intrushield 以及上海金诺网安的 KIDS 等。但商业化的 NIDS 产品一般都是硬件级产品,大多配置使用比较复杂,比较难以掌握,而且比较昂贵。本文将介绍一个出色的免费 NIDS 系统---snort,它基于 GPL 发布,作者是 Martin Roesch,目前最新版本是 2.8.3.1 版。本文将介绍 snort 的技术特点及如何在 Linux 下利用 snort 构建 NIDS 系统。

3.2 Snort 介绍

Snort 是一个高性能的网络入侵检测系统,主要适用于中小型企业网络,尤其适合一些无力承

3

Snort 网络入侵检测系统

受大型商业入侵检测系统高昂费用的中小型公司。Snort 具有如下特点：

- 采用误用检测模型，即首先建立入侵行为特征库，然后在检测过程中，将收集到的数据包和特征代码进行比较，以得出是否被入侵的结论
- 主要检测模式：信息包嗅探器、信息包记录器、成熟的入侵探测系统
- Snort 是一个轻量级的网络入侵检测系统，所谓轻量级是指该软件在运行时只占用极少的网络资源，对原有网络性能影响很小
- 支持多种系统平台，包括 Linux、Solaris、BSD、IRIX、HP-UX、Windows 等
- 实时流量分析，丰富的报警机制，能够快速检测网络攻击，及时地发出报警
- 能够进行协议分析，支持的协议有 TCP、UDP 和 ICMP
- 能够检测多种方式的攻击和探测，例如：缓冲区溢出、秘密端口扫描、CGI 攻击、SMB 探测、探测操作系统指纹特征的企图等等
- 日志格式支持二进制和 ASCII，支持把日志记入数据库，当前支持的数据库包括：Postgresql、MySQL、Oracle 等
- 扩展性能较好，对于新的攻击威胁反应迅速
- 支持多种插件扩展，包括数据库日志输出、碎数据包检测、端口扫描检测等各种插件
- snort 的规则语言非常简单，发现新的攻击后，可以很快根据 Bugtraq 邮件列表，找出特征码，写出新的检测规则。
- 遵循公共通用许可证 GPL 协议，采用 C 语言编写，其源代码可以被自由的读取、修改

Snort 的体系结构包括数据收集、数据分析、日志和告警三个主要部分（如图 3.2 所示）。

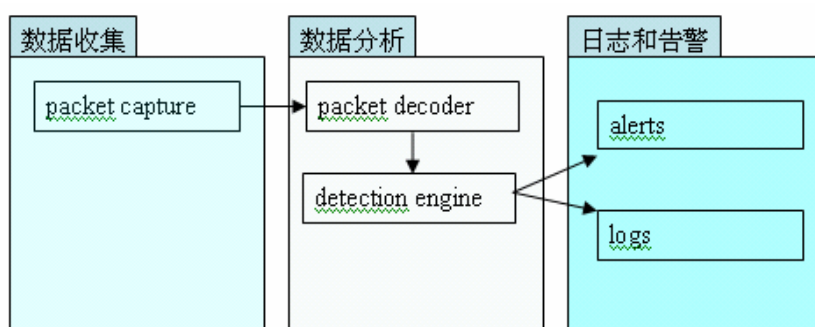


图 3.2 Snort 的体系结构

1. 数据收集：Snort 使用 libpcap 捕获数据包。
2. 数据分析：包含包解码和探测引擎两部分。包解码为探测引擎准备数据，探测引擎按照启动时加载的规则，对每个数据包进行分析，如数据与规则匹配则触发规则中指定的动作。
3. 日志记录/告警系统：日志和告警是两个分离的子系统。日志将包解码收集到的信息记录下

来，缺省情况下，所有的日志将会写到 `/var/log/Snort` 文件夹中，而告警信息将会写到 `/var/log/Snort/alert` 文件中。

3.3 构建 Snort+Base NIDS 系统

3.3.1 系统结构

基于 Snort 和 BASE 的入侵检测系统是“传感器—数据库—分析平台”的三层架构体系。

libpcap 和 snort 构成系统的传感器部分。libpcap 作为系统底层网络接口驱动，Snort 作为数据包解码、筛选和转储程序。为了完整监控整个网络，通常在多个网络关键节点上部署 IDS 传感器。

Snort 获得记录信息后可以存储到本地日志也可以发送到 Syslog 服务器或是直接存储到数据库中，数据库可以是本地也可以是远程的，Snort2.8.0 支持 MySQL、MSSQL、PostgreSQL、ODBC、Oracle 等数据库。

Snort 的日志和报警记录包含大量的信息，人工对这些信息进行整理分析是一件无法完成的任务，所以需要有一个能够操作查询数据库的分析平台。ACID 是 Snort 早期最流行的 WEB 分析平台，使用 PHP 开发，但其开发组织已不再提供更新和支持，现在已经被 BASE 所取代。

这三种角色既可以部署于同一个主机平台也可以部署在不同的物理平台上，架构非常灵活。

3.3.2 安装前的准备工作

1. LAMP 平台准备

安装 RHEL5 操作系统，定制安装的软件包，勾选“开发工具包”、“万维服务器”、“MySQL 数据库”选项。安装完成后关闭 SELinux 和防火墙功能。

2. 确保以下软件包已安装

Snort 与 tcpdump 命令一样，都是使用 Libpcap 包捕获网络数据信息，所以需要确保 libpcap 包已被系统安装。

```
libpcap-1.10-26.i386.rpm
libpcap-devel-1.10-26.i386.rpm
zlib-1.2.3-3.i386.rpm
zlib-devel-1.2.3-3.i386.rpm
pcre-6.6-1.1.i386.rpm
pcre-devel-6.6-1.1.i386.rpm
mysql-devel-5.0.22-2.1.i386.rpm
```



3

Snort 网络入侵检测系统

除 `pcre-devel-*.rpm`、`mysql-devel-*.rpm` 和 `libpcap-devel-*.rpm` 外，其余软件包一般情况下系统均已自动安装，这几个软件包均在 RHEL5 安装光盘 `Server` 目录下，如没有安装请自行安装。

3. 需要额外下载的软件包（如表 3-1 所示）

表 3-1 软件包列表

软件包名称	参考下载地址
<code>snort-2.8.3.1.tar.gz</code>	http://www.snort.org
<code>snortrules-pr-2.4.tar.gz</code>	http://www.snort.org
<code>php-pear-1.7.2-2.fc10.noarch.rpm</code>	http://rpm.pbone.net
<code>adodb4991.gz</code>	http://sourceforge.net/projects/adodb/
<code>base-1.4.1.tar.gz</code>	http://sourceforge.net/projects/secureideas/
<code>Image_Color-1.0.2.tgz</code>	http://pear.php.net/packages.php?catpid=12&catname=Images
<code>Image_Canvas-0.3.1.tgz</code>	http://pear.php.net/packages.php?catpid=12&catname=Images
<code>Image_Graph-0.7.2.tgz</code>	http://pear.php.net/packages.php?catpid=12&catname=Images
<code>Numbers_Roman-1.0.2.tgz</code>	http://pear.php.net/packages.php?catpid=17&catname=Numbers
<code>Numbers_Words-0.15.0.tgz</code>	http://pear.php.net/packages.php?catpid=17&catname=Numbers
<code>Mail-1.1.14.tgz</code>	http://pear.php.net/packages.php?catpid=14&catname=Mail
<code>Mail_mimeDecode-1.5.0.tgz</code>	http://pear.php.net/packages.php?catpid=14&catname=Mail
<code>Mail_Mime-1.5.2.tgz</code>	http://pear.php.net/packages.php?catpid=14&catname=Mail

3.3.3 安装 Snort

1. 编译安装

1) 建立 snort 用户及目录

```
[root@localhost ~]# useradd -d /etc/snort -s /sbin/nologin snort
[root@localhost ~]# mkdir /var/log/snort/ //创建日志存放目录
```

2) 编译安装 snort

```
[root@localhost ~]# tar zxvf snort-2.8.3.1.tar.gz
[root@localhost ~]# cd snort-2.8.3.1
[root@localhost snort-2.8.3.1]# ./configure --with-mysql
[root@localhost snort-2.8.3.1]# make && make install
```

3) 复制配置文件



```
[root@localhost snort-2.8.3.1]# cp -rf etc/* /etc/snort/
```

4) 安装规则包

```
[root@localhost snort-2.8.3.1]# cd /etc/snort/
[root@localhost snort]# tar -zxvf ~/src/snortrules-pr-2.4.tar.gz rules/
```

5) 修改 snort 配置文件

```
[root@localhost ~]# cp /etc/snort/snort.conf snort.conf.bak
[root@localhost ~]# vi /etc/snort/snort.conf
```

只需要改动一个地方：

```
var RULE_PATH /etc/snort/rules           //指定规则文件位置
```

2. 启动 snort

首先需要注释掉/etc/snort/rules/web-misc.rules 文件的 97, 98, 452 行, 否则运行 snort 时将会提示错误信息:

```
"FATAL ERROR: (/etc/snort/rules/web-misc.rules)97"
"FATAL ERROR: (/etc/snort/rules/web-misc.rules)98"
"FATAL ERROR: (/etc/snort/rules/web-misc.rules)452"的错误
```

以 NIDS 模式启动 snort 系统:

```
[root@localhost ~]# /usr/local/bin/snort -dev -c /etc/snort/snort.conf
```

3. Snort 的运行模式

snort 有三种工作模式: 嗅探器、数据包记录器、网络入侵检测系统。嗅探器模式仅仅是从网络上读取数据包并作为连续不断的流显示在终端上。数据包记录器模式把数据包记录到硬盘上。网路入侵检测模式是最复杂的, 而且是可配置的。我们可以让 snort 分析网络数据流以匹配用户定义的一些规则, 并根据检测结果采取一定的动作。

1) 嗅探器

嗅探器模式就是 snort 从网络上读出数据包然后显示在控制台上。

只输出 IP 和 TCP/UDP/ICMP 的包头信息

```
[root@localhost ~]# snort -v
```

在输出包头信息的同时显示包的数据信息

```
[root@localhost ~]# snort -vd
```

附加显示数据链路层的信息



3

Snort 网络入侵检测系统

```
[root@localhost ~]# snort -vde
```

2) 数据包记录器

在嗅探器模式的基础上, 增加“-l”参数项, 指定日志文件存放目录, **snort** 将变成数据包记录器模式。

```
[root@localhost ~]# snort -dev -l /var/log/snort
```

这个命令告诉 **snort** 把进入网络的所有包的数据链路、TCP/IP 以及应用层的数据记录到目录/var/log/snort 中。

如果 **snort** 监控的网络流量很大, 那么应该使用二进制的日志文件格式存储日志。使用下面的命令把所有的包记录到一个单一的二进制文件中:

```
[root@localhost ~]# snort -v -l /var/log/snort -b
```

3) 网络入侵检测系统(NIDS)

snort 最重要的用途是作为网络入侵检测系统, 使用下面命令行启动 **NIDS** 模式:

```
[root@localhost ~]# snort -dev -l /var/log/snort -c /etc/snort/snort.conf
```

snort.conf 是规则集文件, **snort** 会对每个包与规则集进行匹配, 发现符合规则的包就采取相应的行动。如果不指定输出目录, **snort** 默认输出到/var/log/snort 目录。

如果长期使用 **snort** 作为入侵检测系统, 建议不要使用-v 选项。因为使用这个选项, 使 **snort** 向屏幕上输出一些信息, 会大大降低 **snort** 的处理速度。此外, 在绝大多数情况下, 也没有必要记录数据链路层的包头, 所以-e 选项也可以不用, 所以使用 **snort** 作为网络入侵检测系统最基本的命令是:

```
[root@localhost ~]# snort -d -l /var/log/snort -c /etc/snort/snort.conf
```

3.3.4 设置报警信息记录到 MySQL 数据库

1. 初始化数据库

```
[root@localhost ~]# mysql -u root -p
mysql> create database snort;
Query OK, 1 row affected (0.05 sec)
mysql> grant all on snort.* to snort@localhost identified by '123456';
mysql> quit
Bye
```

将预先定义好的默认的 **snort** 所需要的表批量导入 **mysql** 的 **snort** 数据库中:



```
[root@localhost ~]# cd /root/src/snort-2.8.3.1/
[root@localhost snort-2.8.3.1]# mysql -u snort -p snort < ./schemas/create_mysql
```

2. 检查数据库

```
[root@localhost snort-2.8.3.1]# mysql -u snort -p
mysql> use snort;
mysql> show tables;
mysql> quit;
```

3. 修改 snort.conf

```
[root@localhost ~]# vi /etc/snort/snort.conf
```

将 output database 行修改为:

```
output database: log, mysql, user=snort password=123456 dbname=snort
host=localhost
```

3.3.5 构建安全图形分析引擎

1. 安装 php-GD 库

```
[root@localhost ~]# rpm -ivh /mnt/Server/php-gd-5.1.6-5.el5.i386.rpm
```

2. 安装 php-mysql

```
[root@localhost ~]# rpm -ivh /mnt/Server/php-pdo-5.1.6-5.el5.i386.rpm
[root@localhost ~]# rpm -ivh /mnt/Server/php-mysql-5.1.6-5.el5.i386.rpm
```

3. 通过 Pear 安装 PHP 图表插件

```
[root@localhost ~]# cd /root/src/
[root@localhost src]# rpm -ivh php-pear-1.7.2-2.fc10.noarch.rpm
[root@localhost src]# pear install Image_Color-1.0.2.tgz
[root@localhost src]# pear install Image_Canvas-0.3.1.tgz
[root@localhost src]# pear install Numbers_Roman-1.0.2.tgz
[root@localhost src]# pear install Numbers_Words-0.15.0.tgz
[root@localhost src]# pear install Image_Graph-0.7.2.tgz
[root@localhost src]# pear install Mail-1.1.14.tgz
```



3

Snort 网络入侵检测系统

```
[root@localhost src]# pear install Mail_mimeDecode-1.5.0.tgz Mail_Mime-1.5.2.tgz
```

4. 安装 ADODB

ADODB 基于 php 开发，是 PHP 用于连接数据库的插件，它的安装很简单只需要将其解压到 WEB 发布目录下即可。

```
[root@localhost ~]# cd /root/src
[root@localhost src]# tar zxvf adodb4991.gz -C /usr/local/apache/htdocs/
```

5. 安装配置 BASE

BASE 是一个以 PHP 为基础的网络图形用户界面(GUI)，用于运行日志分析。它包括一个搜索引擎，信息包阅读器(packet viewer)，警报管理及图表和统计表的产生工具以及网页的图形界面，使网络管理员处理警报及日志的工作变得更简单容易。

```
[root@localhost src]# cd /root/src
[root@localhost src]# tar zxvf base-1.4.1.tar.gz -C /usr/local/apache/htdocs/
[root@localhost src]# cd /usr/local/apache/htdocs/
[root@localhost htdocs]# mv base-php4 base
[root@localhost htdocs]# cd base
[root@localhost base]# cp base_conf.php.dist base_conf.php
```

编辑 “base_conf.php”

```
[root@localhost base]# vi base_conf.php
```

所需要修改的内容包括：

```
$BASE_urlpath = "/base";
$DBlib_path = "/usr/local/apache/htdocs/adodb/";
$DBtype = "mysql";
$alert_dbname = "snort";
$alert_host = "localhost";
$alert_port = "";
$alert_user = "snort";
$alert_password = "123456";
```

BASE 支持中文界面，只需更改“\$BASE_Language”参数即可。但是若设置成中文语言，可能会导致“Graph Alert Data”无法显示图表，因此非必要时不建议使用。

```
$BASE_Language = 'simplified_chinese'
```

通过浏览器进行最后设置

访问<http://10.0.0.108/base>

首次登录会出现如下信息：

```
Basic Analysis and Security Engine (BASE)
The underlying database snort@localhost appears to be incomplete/invalid.
The database version is valid, but the BASE DB structure (table: acid_ag)is not
present. Use the Setup page to configure and optimize the DB.
```

点击 setup page，按向导提示进行安装即可。

3.4 简单的测试

使用 putty 打开第一个会话窗口：

```
[root@localhost ~]# tail -f /var/log/snort/alert
```

使用 putty 打开第二个会话窗口：

```
[root@localhost ~]# snort -d -c /etc/snort/snort.conf
```

使用 putty 打开第三个会话窗口：

安装 RHEL5 安装光盘上的 nmap-4.11-1.1.i386.rpm 软件包。

```
[root@localhost ~]# mount /dev/cdrom /mnt
[root@localhost ~]# rpm -ivh /mnt/Server/nmap-4.11-1.1.i386.rpm
```

执行端口扫描

```
[root@localhost ~]# nmap -sS 10.0.0.200
```

将会观察到第一窗口中 alert 文件内容的变化。登陆 base 页面后可以看到报警界面（如图 3.3 所示）。



3

Snort 网络入侵检测系统

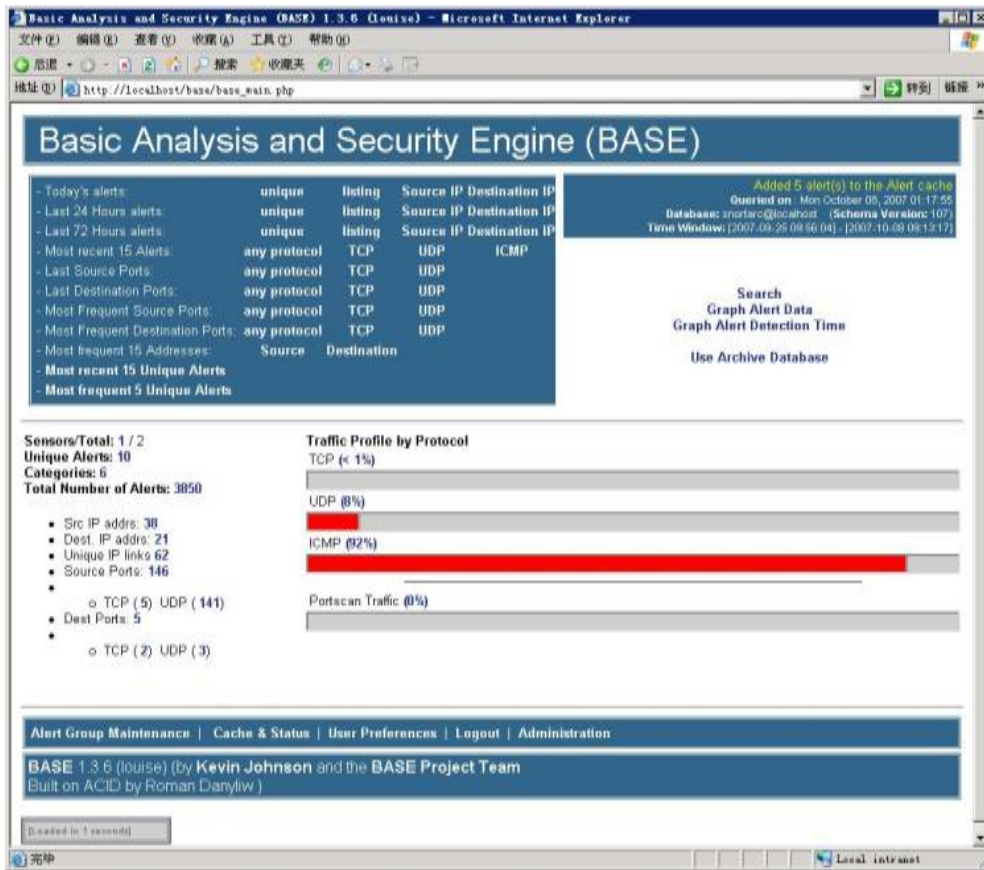


图 3.3 snort 报警提示界面



四
四
之
四
十

4

构建 LVS 负载均衡系统

3
.
0

4

构建 LVS 负载均衡系统

4.1 Linux 集群系统概述

4.1.1 常见集群系统分类

1. 高可用性集群 (High Availability Cluster)

运行于两个或多个节点上，目的是在系统出现某些故障的情况下，最大限度地减少服务中断时间，保障应用程序持续提供服务的能力。这类集群中比较著名的有 Turbolinux TurboHA、Heartbeat、Kimberlite 等。对于此类集群还有很多通俗的名称，如“双机热备”，“双机互备”等。

2. 负载均衡集群 (Load Balance Cluster)

提供和节点个数成正比的负载能力，这种集群适合需要提供大负载访问量的服务，如 Web。这类集群中比较著名的有 Turbolinux Cluster Server、Linux Virtual Server。此类集群把负载压力根据某种算法合理分配到集群中的每一台计算机上，以减轻主服务器的压力，降低对主服务器的硬件和软件要求。

3. 科学计算集群 (High Performance Computing Cluster)

利用超级计算集群软件将多个节点的计算机联结在一起，完成通常只有超级计算机才能完成的计算任务。这类软件有 Turbolinux EnFusion、SCore 等。

本章将介绍 Linux Virtual Server 负载均衡集群系统，并以此为基础构建高可用负载均衡集群。

4.1.2 Linux Virtual Server 项目

Linux Virtual Server 项目由章文嵩博士在 1998 年 5 月发起创立，是国内最早的自由软件项目之一。Linux Virtual Server 针对高可伸缩、高可用网络服务的需求，提出了基于 IP 层和基于内容请求分发的负载均衡调度解决方案。

由于该项目的负载调度技术是在 Linux 内核中实现的，所以称之为“Linux 虚拟服务器” (Linux Virtual Server)，简称为 LVS。

LVS 集群的特点可以归结如下：

- **功能：**基于内容的请求分发技术和 3 种 IP 负载均衡技术，十种连接调度算法
- **适用性：**后端服务器可运行 Linux，Unix，Mac/OS 和 Windows NT/2000 等多种操作系统。支持绝大多数的 TCP 和 UDP 协议
- **灵活性：**无需对客户机和服务器作任何修改
- **性能：**支持几百万并发连接，系统的最大吞吐量可接近 10Gbits/s
- **可靠性：**已经在很多大型的、关键性的站点的应用
- **软件许可证：**基于 GPL 许可证发行

LVS 集群系统已在美、英、德、澳等国的几十个站点上正式使用，例如：

- 英国国家 JANET Cache Service (www.cache.ja.net)



- Linux 的门户网站 (www.linux.com)
- SourceForge (sourceforge.net)
- Real 公司 (www.real.com)
- NetWalk (www.netwalk.com)

很多应用系统厂商都利用 LVS 的代码提供了基于 LVS 的集群方案:

- RedHat 从其 6.1 发行版起已包含 LVS 代码, 并开发了一个 LVS 图形化的集群管理工具 Piranha, 用于控制 LVS 集群。
- VA Linux 向客户提供基于 LVS 的服务器集群系统, 并且提供相关的服务和支持。
- TurboLinux 的 Linux 集群产品“TurboCluster”实际上是基于 LVS 的构想和代码。
- 红旗 Linux 和中软都提供基于 LVS 的集群解决方案, 并在 2000 年 9 月召开的 Linux World China 2000 上展示

4.1.3 LVS 体系结构介绍

LVS 通过前端一个负载调度器 (Load Balancer, 也叫负载均衡器) 无缝地将网络请求调度到真实服务器上, 从而使得服务器集群的结构对客户是透明的, 客户访问集群系统提供的网络服务就像访问一台高性能、高可用的服务器一样。客户程序不受服务器集群的影响不需作任何修改。系统的伸缩性通过在服务器机群中透明地加入或删除一个节点来达到, 通过检测节点或服务进程故障和正确地重置使系统达到高可用性。

LVS 的抽象体系结构分为三个层次: 负载调度器 (load balancer)、服务器池 (server pool)、共享存储 (shared storage) (如图 4.1 所示)。

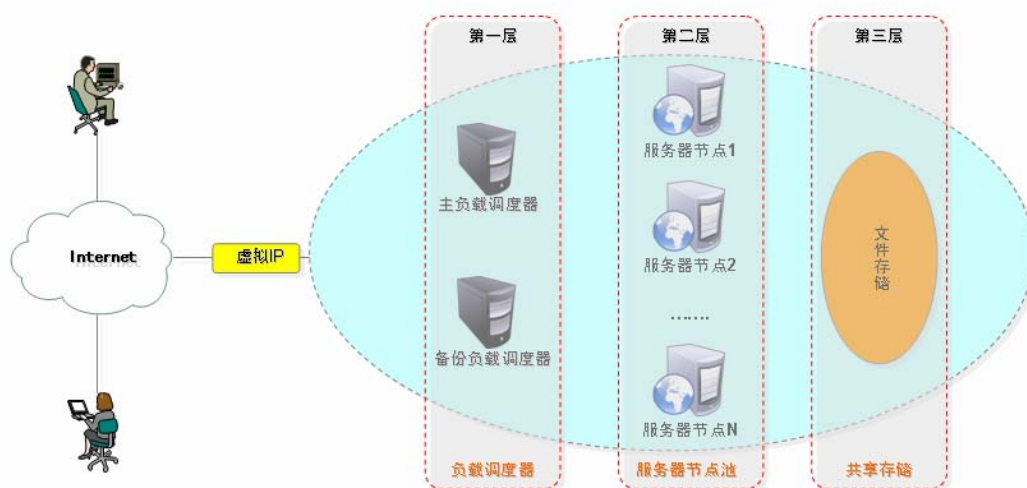


图 4.1 LVS 的三层体系结构示意图

4 构建 LVS 负载均衡系统

1. 第一层是负载调度器

这是集群的唯一入口点。负载调度器上设定一个 VIP (Virtual IP, 虚拟 IP), 整个集群共用这个 VIP, 通过它客户端可以把整个集群看作一个独立的具有合法 IP 地址的主机系统, 客户端的所有访问都发往这个 VIP。

LVS 的负载调度器使用 IP 负载均衡技术、基于内容请求分发技术或者两者相结合, 将客户的请求发送至其后的服务器节点池。

在实际应用中, 一般都对负载调度器使用 HA 技术, 实现热备份, 在主负载调度器失效时平滑替换至备份负载调度器, 常用 Heartbeat 软件实现 HA 的功能。

2. 第二层是服务器池

客户端发出的服务请求经过均衡器处理以后, 转交到服务器池由具体的服务器响应请求并返回数据。通常在服务器节点池上提供 Web 服务、FTP 服务等。

服务器节点有可能出现暂时失效的情况, 因此, 需要容错机制识别这种错误, 及时进行处理。同样, 当错误排除后, 集群应能够自动识别恢复, 把结点重新纳入集群继续运行。LVS 常用 ldirectord 软件实现容错机制。

3. 第三层是共享存储

为集群内部提供稳定、一致的文件存取服务。

4.2 LVS 的负载均衡模型及算法

LVS 的负载调度器使用 IP 负载均衡技术、基于内容请求分发技术或者两者相结合。

在调度器的实现技术中, IP 负载均衡技术是效率最高的。而 IP 负载均衡技术中常见的是 VS/NAT 技术 (Virtual Server via Network Address Translation)。它通过网络地址转换 (Network Address Translation) 将一组服务器构成一个高性能的、高可用的虚拟服务器, 大多数商品化的 IP 负载均衡调度器产品都是使用这种技术, 如 Cisco 的 LocalDirector、F5 的 Big/IP 和 Alteon 的 ACEDirector。

而 LVS 在 VS/NAT 的基础上, 还拓展实现了基于 IP 隧道的 VS/TUN (Virtual Server via IP Tunneling), 和通过直接路由的 VS/DR (Virtual Server via Direct Routing), 极大地提高系统的伸缩性。

4.2.1 LVS 负载均衡模型

LVS 的三种负载均衡模型 (地址转换 (NAT)、IP 隧道 (IP Tunneling) 和直接路由 (DR)) 原理如下:

1. Virtual Server via Network Address Translation (VS/NAT)

NAT 模式下的服务器节点使用的是私有 IP，均衡器是集群的唯一出入口，网络结构呈现为一种类似防火墙的私有网结构，服务器结点无法和客户端直接通信，所有数据都需要经过均衡器进行处理（如图 4.2 所示）。

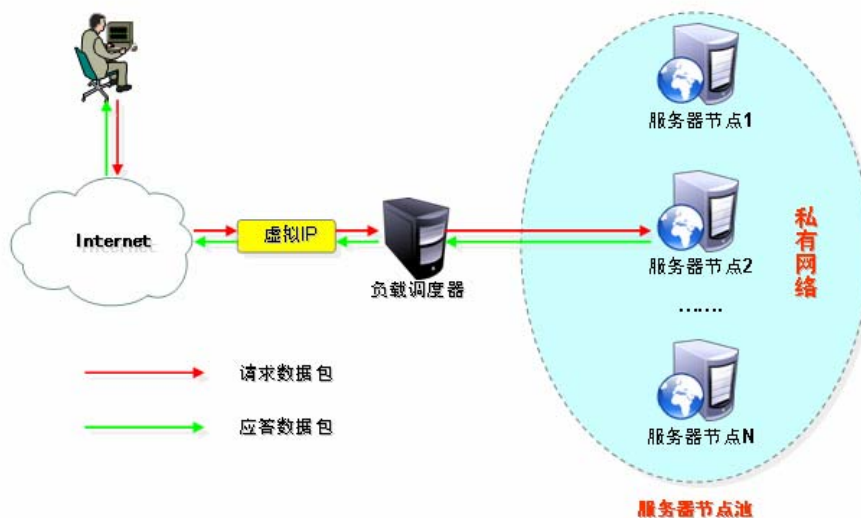


图 4.2 VS/NAT 模型

2. Virtual Server via IP Tunneling (VS/TUN)

VS/TUN 模式采用开放的网络结构，负载均衡器仅仅处理进入集群的请求数据包，而返回数据包不经过负载均衡器。服务器结点拥有合法的公网 IP 地址，可将应答包直接返回给客户端。负载均衡器和服务器结点的连接可以是同一 LAN 上，也可以跨越 WAN 在不同的网段上。

负载均衡器通过 IP/IP 协议将客户端的请求包封装为新的 IP 包，发给服务器节点。服务器节点收到均衡器发来的 IP/IP 数据包后，将包解开，根据包内的客户端源地址将处理结果，直接返回给客户端（如图 4.3 所示）。

4

构建 LVS 负载均衡系统

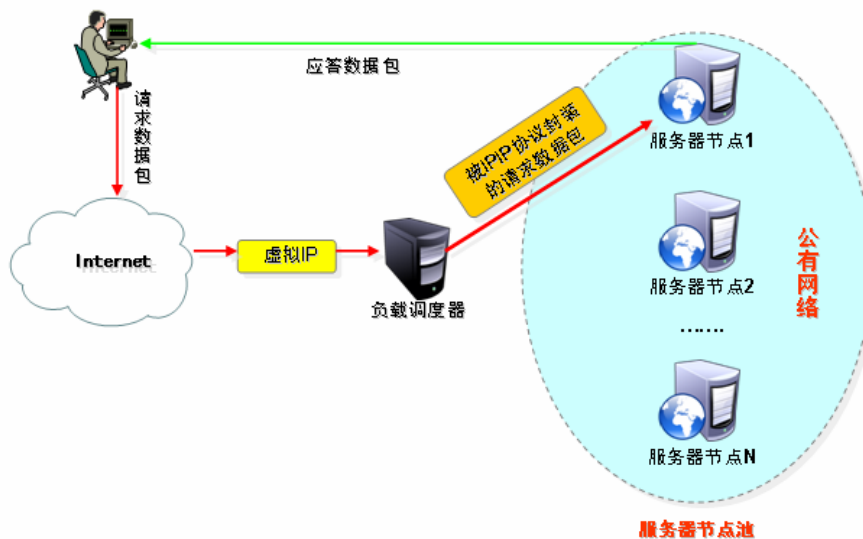


图 4.3 VS/TUN 模型

3. Virtual Server via Direct Routing (VS/DR)

DR 模式下服务器节点的应答数据也经过均衡器，而是直接返回给客户端。服务器结点也必须拥有合法 IP 地址。而且，负载均衡器和服务器结点必须位于同一个网段。

负载均衡器接收到客户端请求数据包后，选择合适的服务器结点，将请求包的 MAC 地址改写为目的服务器结点的 MAC 地址，再将此包广播到服务器器节点所在网段。每个服务器结点都设定一个虚拟的网络设备 (lo:0)，这个设备绑定了和均衡器一样的 VIP，只是该设备并不响应对 VIP 的 ARP 解析，不会和均衡器的 VIP 产生地址冲突。负载均衡器收到符合自身 MAC 的 IP 包后，经过处理后直接将应答数据返回给客户（如图 4.4 所示）。



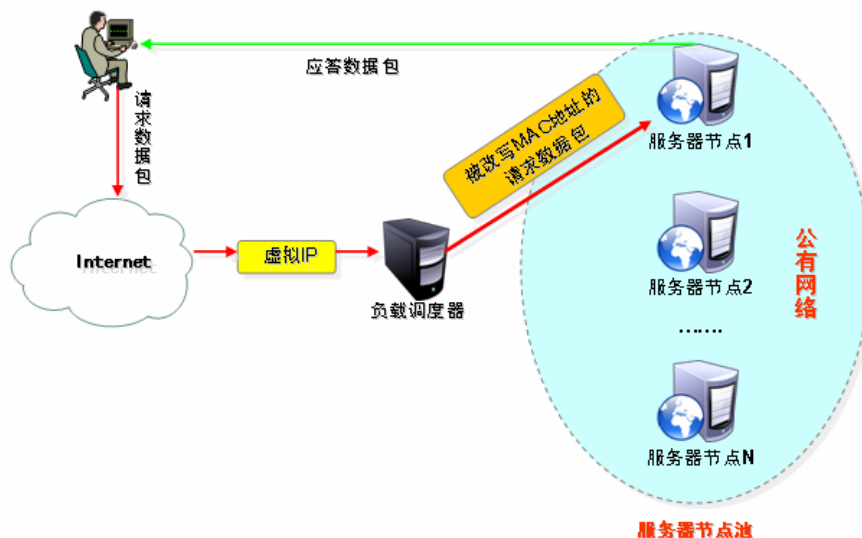


图 4.4 VS/DR 模型

4.2.2 三种模型的比较

选择 LVS 作为集群负载均衡解决方案的时候，首先根据当前应用环境的情况确定使用合适 IP 负载均衡结构。如果只拥有一个合法的 IP 地址，或者需要构造一个安全性更高的集群，而又不担心性能问题，建议使用 NAT 模式；如果对性能有比较高的要求，而应用又是基于 Linux 系统，建议使用 TUN 或者 DR 模式（如表 4.1 所示）。

表 4.1 三种负载均衡模式的比较

	NAT 模式	TUN 模式	DR 模式
对结点的要求	任何操作系统	必须支持 IP 隧道协议	支持虚拟网卡设备，能禁用 ARP 响应功能
网络要求	拥有私有 IP 地址的局域网	拥有合法 IP 地址的局域网或者广域网	拥有合法 IP 地址的局域网，服务器结点与均衡器必须在同一个网段
支持结点数量	10~20 个，视均衡器的处理能力而定	较高，可以支持到 100 个服务器结点	较高，可以支持到 100 个服务器结点
网 关	均衡器即为服务器结点的网关	服务器结点同自己的网关连接，不经过均衡器	服务器结点同自己的网关连接，不经过均衡器
安全性	较好，采用内部 IP，服务器结点隐蔽	较差，采用公用 IP 地址，结点完全暴露	较差，采用公用 IP 地址，结点完全暴露

4

构建 LVS 负载均衡系统

IP 要求	仅需要一个合法 IP 地址 作为 VIP	除 VIP 外, 每个服务器结 点需拥有合法的 IP 地址, 可以直接路由至客户端	除 VIP 外, 每个服务器结 点需拥有合法的 IP 地址, 可以直接路由至客户端
效率	一般	高	最高

4.2.3 LVS 负载调度算法

针对不同的网络服务需求和服务器配置, IPVS 调度器实现了如下 10 种负载调度算法:

- **轮叫 (Round Robin):** 以顺序循环将服务请求分配到集群中的内容服务器上, 所有服务器地位平等, 不考虑服务器上实际的连接数和系统负载。
- **加权轮叫 (Weighted Round Robin):** 循环方式调度, 但在循环中给每个内容服务器分配指定权重的连接, 从而充分考虑各内容服务器处理能力之间的差异。
- **最少链接 (Least Connections):** 将新到的连接请求动态地分配给现有活跃连接数最少的内容服务器。如果集群系统的真实服务器具有相近的系统性能, 可考虑采用“最小连接”。
- **加权最少链接 (Weighted Least Connections):** 在最少链接的算法基础上, 增加权重参考值, 较高权值的服务器承受较大比例的负载。在集群系统中的服务器性能差异较大的情况下, 建议采用“加权最少链接”。
- **基于局部性的最少链接 (Locality-Based Least Connections):** 专门为缓存集群(cache cluster)设计的算法, 将对同一目的地址的连接请求分配给固定的一台缓存集群节点。
- **带复制的基于局部性最少链接 (Locality-Based Least Connections with Replication):** 在集群节点中维护若干针对特定目的地址的缓存集群集合, 当有匹配地址的连接请求时, 分配给该集合中活跃连接数最少的缓存集群。
- **目标地址散列 (Destination Hashing):** 根据目标地址查找事先设定的静态哈希表来分配连接。
- **源地址散列 (Source Hashing):** 按客户地址查找设定的静态哈希表来分配连接, 可实现服务就近提供, 保证服务质量。
- **最短期望延迟 (Shortest Expected Delay):** 按内容服务器的最短期望延迟分配连接, $SED=(C_i+1)/U_i$, 即(任务数+1)/权重。
- **无须队列等待 (Never Queue):** 将请求优先转发给空闲的服务器, 否则按最短预计延迟策略分配连接。

4.2.4 IPVS-DR +heartbeat+ldirectord 构建高可用负载均衡集群方案

IPVS-DR +ldirectord+heartbeat 高可用负载均衡集群方案前端负载调度器采用双机热备份方式, 双机均安装双网卡, 一个网卡用于连接集群系统, 另一个作为冗余心跳线路连接双机。

主负载调度器及备份负载调度器同时安装 ldirectord 及 heartbeat, 并同时运行 Heartbeat, 相



互监视“健康”状况。一旦备份负载调度器监测到主负载调度器发生故障，备份负载调度器上的 **Heartbeat** 通过运行脚本来启动备份调度器上的 **LVS** 服务和 **ldirectord** 服务，完成虚拟 IP 故障转移。一旦主负载调度器恢复正常工作，备份负载调度器上的 **Heartbeat** 将通过脚本停掉备份负载调度器上的 **LVS** 及 **ldirectord** 服务，主负载调度器重新恢复对集群的资源管理。

ldirectord 工作原理：**ldirectord** 需要在真实服务器内启用 **web** 服务，然后 **ldirectord** 通过循环检查 **web** 服务是否存活。如果真实服务器不存活，则使用 **ipvsadm** 命令将其权重设为 **0**，以确保客户的连接不会导向失效的真实服务器；如果真实服务器修复上线，则又将其的权重设为 **1** 以使其能够继续为客户端连接提供服务。简单的说 **ldirectord** 功能就是动态维护 **ipvs** 虚拟服务器表，确保资源节点的可用性。

heartbeat 的工作原理：**heartbeat** 核心的功能包括两个部分：心跳监测和资源接管。心跳监测通过网络链路或串口进行，主/备服务器之间相互发送报文来告诉对方自己当前的状态，如果在指定的时间内未收到对方发送的报文，那么就认为对方失效，同时启动资源接管模块接管运行在对方上的资源。

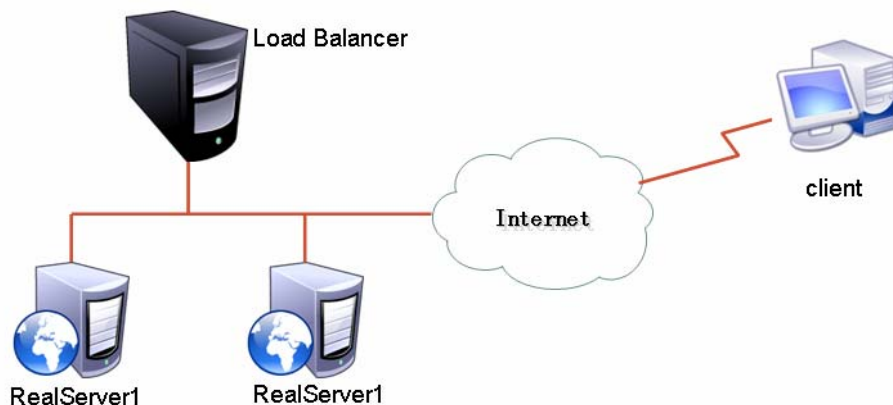
4.3 LVS 应用案例

4.3.1 基于直接路由模式（DR）的 LVS

1. 实现目标

客户通过 VIP 211.100.1.198 访问服务器。Load Balancer 将来自客户的访问按一定的负载均衡机制分发到 **RealServer1** 和 **RealServer1** 这两台实际提供服务的服务器。

2. 网络结构



4 构建 LVS 负载均衡系统

图 4.5 基于直接路由的 LVS 负载均衡

3. IP 地址规划

三台计算机，系统均使用 RHEL5 系统，位于同一网段，其中一台作为均衡器（Load Balancer）进行负载分发，另两台 Real Server 提供实际的 Web 服务。

表 4.2 IP 地址参数设置

名称	角色	IP 地址
Load Balancer	均衡器	eth0:0 (Vip): 211.100.1.198
		Eth0:211.100.1.196
RealServer1	服务器节点 1	lo:0 (VIP) 211.100.1.198 Eth0:211.100.1.191
RealServer2	服务器节点 2	lo:0 (VIP) 211.100.1.198 Eth0:211.100.1.192

4. 所需软件列表

ipvsadm-1.24-8.1.i386.rpm 软件包，RHEL5 安装光盘已带有此软件包。

5. 安装配置过程

■ 负载调度器（Master）上的设置

1) 设定网卡 ip

```
[root@localhost ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=211.100.1.196
NETMASK=255.255.255.0
.....
```

2) 安装 ipvsadm 管理程序

```
[root@localhost ~]# mount /dev/cdrom /mnt
[root@localhost ~]# rpm -ivh /mnt/Cluster/ipvsadm-1.24-8.1.i386.rpm
```

3) 配置 VIP 设定脚本

```
[root@localhost ~]# vi /opt/vip.sh
```



```
#!/bin/bash
# description: Set the VIP of Directorserver
VIP=211.100.1.198
/sbin/ifconfig eth0:0 $VIP broadcast $VIP netmask 255.255.255.255 up
/sbin/route add -host $VIP dev eth0:0
```

4) 设置脚本可执行权限

```
[root@localhost ~]# chmod u+x /opt/vip.sh
```

5) 脚本运行

```
[root@localhost ~]# echo "/opt/vip.sh" >> /etc/rc.local //开机自启动
[root@localhost ~]# /opt/vip.sh //运行
[root@LVS-Master ~]# ifconfig eth0:0
eth0:0    Link encap:Ethernet  HWaddr 00:0C:29:38:E3:69
          inet addr:192.168.1.198  Bcast:192.168.1.198  Mask:255.255.255.255
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          Interrupt:169 Base address:0x2000
```

6) ipvsadm 命令工具的使用

```
[root@localhost ~]# ipvsadm -A -t 192.168.1.198:80 -s rr -p 600
```

其中，用到的几个选项含义如下：

- A 增加一个 virtual service
- t tcp 协议，-u 是 udp 协议
- s 指定使用的算法
- s rr 指定使用轮叫算法 (rr)，可以自行选择相应的算法，使用 ipvsadm -h 查看帮助
- p 设置连接保持时间 (将同一客户端的请求转发至同一个 realserver，默认为 300 秒)

```
[root@localhost ~]# ipvsadm -a -t 192.168.1.198:80 -r 192.168.1.192:80 -g
```

其中，用到的几个选项含义如下：

- a 增加一个 virtual server
- g 是指定 lvs 使用 DR 直接路由模式，可自行修改 (-i 为 TUN，-m 为 NAT)

```
[root@localhost ~]# ipvsadm -a -t 192.168.1.198:80 -r 192.168.1.191:80 -g
[root@localhost ~]# ipvsadm-save > /etc/sysconfig/ipvsadm //保存设置命令
```

将上述 ipvsadm 设置保存到 /etc/sysconfig/ipvsadm 文件以后，下次启动可直接执行 “service ipvsadm start” 命令，不再需要逐条输入命令。



4

构建 LVS 负载均衡系统

```
[root@localhost ~]# ipvsadm -l //查看 ipvsadm 列表
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port      Forward Weight ActiveConn InActConn
TCP  211.100.1.198:http rr persistent 600
  -> 211.100.1.192:http      Route    1      0      0
  -> 211.100.1.191:http      Route    1      0      0
```

■ Realserver 服务器上的设置

提供服务的 RealServer1 和 RealServer2 不需要安装任何 IPVS 软件，只需要设置在开机后自动执行以下脚本：

```
[root@localhost ~]# vi /opt/lvs-dr
#!/bin/sh
VIP="211.100.1.198"
/sbin/ifconfig eth0 211.100.1.191/24
/sbin/ifconfig lo:0 $VIP broadcast $VIP netmask 255.255.255.255 up
/sbin/route add -host $VIP dev lo:0
echo "1" >/proc/sys/net/ipv4/conf/lo/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/lo/arp_announce
echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce
#end
```

此脚本使 realserver 不响应 arp 请求，将此脚本分别在 realserver 设置开机自执行

```
[root@localhost ~]# chmod u+x /opt/lvs-dr
[root@localhost ~]# echo "/opt/lvs-dr" >> /etc/rc.local
```

提示：关闭 ARP 响应的另外一个办法是修改文件/etc/sysctl.conf，把下面四句添加在文件最后：

```
net.ipv4.conf.lo.arp_ignore = 1
net.ipv4.conf.lo.arp_announce = 2
net.ipv4.conf.all.arp_ignore = 1
net.ipv4.conf.all.arp_announce = 2
```

创建测试页面



在 realserver1 上:

```
[root@localhost ~]# echo "This realserver1 !" > /var/www/html/index.html
```

在 realserver2 上:

```
[root@localhost ~]# echo "This realserver2 !" > /var/www/html/index.html
```

6. 测试

在两台客户机上分别打开浏览器，输入“<http://211.100.1.198>”，将分别看到不同的页面。

4.3.2 IPVS-DR +heartbeat+ldirectord 构建高可用负载均衡集群

1. 实现目标

前面的案例利用 LVS 构建了 LVS Load Balancer 实现了对 Web 服务的负载均衡调度，但是一旦 Load Balancer 发生故障将导致整个集群失效，为此需要构建备用 Load Balancer，实现高可用性。这里我们将使用 Heartbeat 实现对主 LVS Load Balancer 的冗余，同时利用 ldirectord 实现对资源节点的动态监控，实现高可用负载均衡集群。

2. 网络结构



4 构建 LVS 负载均衡系统

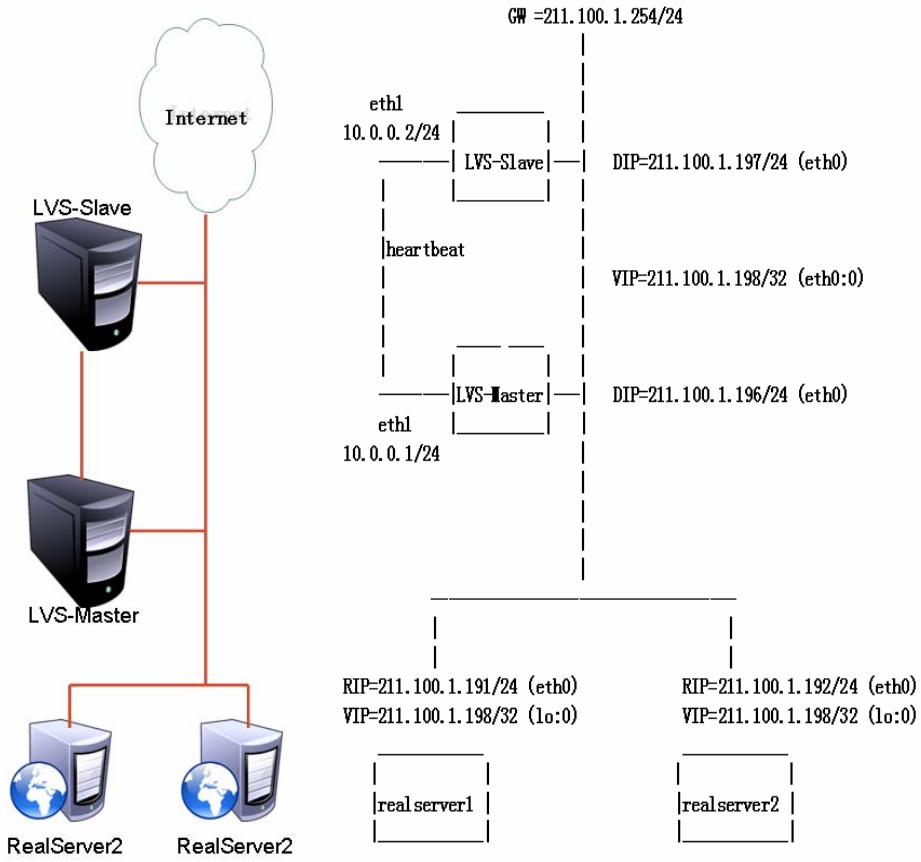


图 4.5 高可用负载均衡

3. IP 地址规划

表 4.3 IP 地址参数设置

名称	角色	IP 地址
LVS-Master	主 Load Balancer	Vip: 211.100.1.198/24
		Eth0:211.100.1.196/24
		Eth1:10.0.0.1/24
LVS-Slave	备份 Load Balancer	Eth0:211.100.1.197/24
		Eth1:10.0.0.2/24
Realserver1	内容服务器 1	Eth0:211.100.1.191/24
Realserver2	内容服务器 2	Eth0:211.100.1.192/24

4. 所需软件列表

表 4.4 所需软件列表



软件包名称	参考下载地址
Heartbeat-STABLE-2-1-STABLE-2.1.4.tar.bz2	http://www.linux-ha.org
libnet-0.10.11.tar.gz	http://search.cpan.org/~gbarr/libnet/
MailTools-2.04.tar.gz	http://search.cpan.org/~markov/MailTools/
perl-libwww-perl-5.805-1.1.1.noarch.rpm	RHEL5 安装光盘
ipvsadm-1.24-8.1.i386.rpm	RHEL5 安装光盘

5. 安装配置过程

■ 在 LVS-master 均衡器上的配置

1) 网络设置

```
[root@localhost ~]# vi /etc/hosts
211.100.1.196  LVS-Master
211.100.1.197  LVS-Slave
[root@localhost ~]# vi /etc/sysconfig/network
HOSTNAME=LVS-Master
[root@localhost ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth0    //配置 eth0 网卡
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=211.100.1.196
NETMASK=255.255.255.0
[root@localhost ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth1    //配置 eth1 网卡
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=static
IPADDR=10.0.0.1
NETMASK=255.255.255.0
```

2) 安装软件

检查 perl-libwww-perl-*软件包是否已安装

```
[root@LVS-Master ~]# rpm -qa|grep perl-libwww-perl
perl-libwww-perl-5.805-1.1.1
```

安装 ipvsadm



4 构建 LVS 负载均衡系统

```
[root@localhost ~]# mount /dev/cdrom /mnt
[root@localhost ~]# rpm -ivh /mnt/Cluster/ipvsadm-1.24-8.1.i386.rpm
```

编译安装 libnet

```
[root@LVS-Master ~]# cd src
[root@LVS-Master src]# tar zxvf libnet-0.10.11.tar.gz
[root@LVS-Master libnet]# ./configure
[root@LVS-Master libnet]# make
[root@LVS-Master libnet]# make install
```

编译安装 MailTools

```
[root@LVS-Master libnet]# cd -
[root@LVS-Master src]# tar zxvf MailTools-2.04.tar.gz
[root@LVS-Master src]# cd MailTools-2.04
[root@LVS-Master MailTools-2.04]# perl Makefile.PL
[root@LVS-Master MailTools-2.04]# make
[root@LVS-Master MailTools-2.04]# make install
```

编译安装 Heartbeat

```
[root@LVS-Master MailTools-2.04]# cd -
[root@LVS-Master src]# tar jxvf Heartbeat-STABLE-2-1-STABLE-2.1.4.tar.bz2
[root@LVS-Master src]# cd Heartbeat-STABLE-2-1-STABLE-2.1.4
//添加 heartbeat 运行需要的 haclient 组, hacluster 用户
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]# groupadd haclient
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]# \
> useradd hacluster -g haclient -s /sbin/nologin
```

Heartbeat 的编译前的配置比较特别, 需要使用其自带的 ConfigureMe 脚本进行配置

```
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]# ./ConfigureMe
.....
./ConfigureMe {configure|make|install|dist|distcheck|package|flags|bootstrap}
.....
```

根据提示使用“./ConfigureMe configure”进行编译前的配置

```
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]# ./ConfigureMe configure
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]# ./ConfigureMe make
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]# ./ConfigureMe install
```



将 heartbeat 设置成开机自启动

```
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]#  
chkconfig --add heartbeat  
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]#  
chkconfig heartbeat on
```

3) 配置 Ldirectord 故障检测及管理

将 ldirectord 的配置文件 ldirectord.cf 的配置文件复制至/etc/ha.d/目录

```
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]#  
cp ./ldirectord/ldirectord.cf /etc/ha.d/  
[root@localhost src]# vi /etc/ha.d/ldirectord.cf           //参照以下内容修改  
checktimeout=3  
checkinterval=1  
fallback=127.0.0.1:80  
autoreload=yes  
quiescent=yes  
virtual=211.100.1.198:80  
    real=211.100.1.191:80 gate  
    real=211.100.1.192:80 gate  
    fallback=127.0.0.1:80 gate  
    service=http  
    scheduler=rr  
    persistent=600  
    #netmask=255.255.255.255  
    protocol=tcp  
    checktype=negotiate  
    checkport=80
```

4) 设定 heartbeat

Heartbeat 的配置文件有三个：ha.cf、haresources、authkeys。这三个配置文件需要放置在/etc/ha.d 目录下面，但是此目录下默认是没有这三个文件的，所以需要复制样本文件。

```
[root@LVS-Master Heartbeat-STABLE-2-1-STABLE-2.1.4]# \  
cp ./doc/ha.cf ./doc/haresources ./doc/authkeys /etc/ha.d/
```

编辑 ha.cf 文件

```
[root@localhost src]# vi /etc/ha.d/ha.cf
```



4 构建 LVS 负载均衡系统

```
#调试日志文件文件
debugfile /var/log/ha-debug
#系统运行日志文件
logfile /var/log/ha-log
#日志等级
logfacility local0
#心跳频率
keepalive 2
#节点死亡时间阈值
deadtime 30
#发出警告时间阈值
warntime 10
#指定 heartbeat 守护进程启动后等待 120 秒再启动资源
initdead 120
#指定心跳信息传递使用的 udp 端口
udpport 694
#在 eth1 使用广播方式发送心跳
bcast eth1
#允许主节点重启成功后拿回资源
auto_failback on
#节点名称，与 uname -n 保持一致
node lvs-master
node lvs-slave
respawn root /usr/lib/heartbeat/ipfail
apiauth ipfail gid=haclient uid=hacluster
```

编辑 haresources 文件，定义 heartbeat 启动设定 VIP，启动 ldirectord 服务

```
[root@localhost src]# vi /etc/ha.d/haresources
LVS-Master IPaddr::211.100.1.198/32 ldirectord:: ldirectord.cf
```

其中，“LVS-Master”是主节点名称，“211.100.1.198/32”是 VIP，“ldirectord:: ldirectord.cf”指定 ldirectord 使用 ldirectord.cf 配置文件。

编辑 authkeys 文件，指定使用认证的方式

```
[root@localhost src]# vi /etc/ha.d/authkeys
auth 1
```



```
1 crc
```

authkeys 文件的权限必须为 600

```
[root@localhost src]# chmod 600 authkeys
```

■ 在 LVS-Slave 均衡器上的配置

1) 网络设置

```
[root@localhost ~]# vi /etc/hosts
211.100.1.196  LVS-Master
211.100.1.197  LVS-Slave
[root@localhost ~]# vi /etc/sysconfig/network,
HOSTNAME=LVS-Master
[root@localhost ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth0    //配置 eth0 网卡
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=211.100.1.197
NETMASK=255.255.255.0
[root@localhost ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth1    //配置 eth1 网卡
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=static
IPADDR=10.0.0.2
NETMASK=255.255.255.0
```

2) 安装软件

LVS-Slave 上的安装的软件及安装步骤过程与 LVS-Master 一致。

3) IPVS、heartbeat、ldirectord

备份节点上的配置文件内容要求与主节点服务器中的保持一致，所以可直接从主节点服务器 LVS-Master 上复制 heartbeat 的三个配置文件（ha.cf、haresources、authkeys），ldirectord 的配置文件（ldirectord.cf）

■ 在 Real server 服务器上的配置

以 real1 上的设置为例：



4 构建 LVS 负载均衡系统

- 1) 安装 Apache 服务
- 2) 网络设置

```
[root@localhost ~]# vi /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
IPADDR=211.100.1.191
NETMASK=255.255.255.0
```

- 3) 编写 IPVS 脚本

```
[root@localhost ~]# vi /opt/ipvs-dr
#!/bin/bash
#description : start realserver
VIP=211.100.1.198
/sbin/ifconfig lo:0 $VIP broadcast $VIP netmask 255.255.255.255 up
/sbin/route add -host $VIP dev lo:0
echo "1" >/proc/sys/net/ipv4/conf/lo/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/lo/arp_announce
echo "1" >/proc/sys/net/ipv4/conf/all/arp_ignore
echo "2" >/proc/sys/net/ipv4/conf/all/arp_announce
sysctl -p
#end

[root@localhost ~]# chmod u+x /opt/ipvs-dr
[root@localhost ~]# /opt/ipvs-dr
[root@localhost ~]# echo "This is realserver1" > /var/www/html/index.html
[root@localhost ~]# service httpd restart
```

6. 测试

- 1) 启动 heartbeat 服务

分别在主、副均衡服务器上启动 heartbeat 服务，启动命令为：

```
[root@localhost ~]# service heartbeat start
```

- 2) 确定 realserver1 和 realserver2 服务器的 httpd 服务启动，页面能正常访问

- 3) 在客户机的web浏览器中输入http://211.100.1.198

从两台具有不同ip的cleint分别登陆浏览<http://211.100.1.198>，应能分别看到“This is realserver1”或者“This is realserver2”



4) 测试 HA 功能

断掉 LVS-Master 的 eth0 和 eth1 的网络连接或者关闭系统模拟故障发生。

在 LVS-Slave 上输入“tail -f /var/log/ldirectord.log”观察日志，大约应在 30 秒内 LVS-Slave 接管 LVS-Master 的资源启动 LVS 服务。主均衡服务器恢复正好正常后，必须能够从备份均衡服务器接管资源。

5) 测试 ldirectord 的功能

在正在提供服务的均衡服务器上输入“tail -f /var/log/ldirectord.log”动态观察日志变动。

断掉 realserver1 的网络连接，将会看到 ldirectord 使用 ipvsadm 命令删除 realserver1 的服务器记录；恢复 realserver1 的网络连接后，将会看到 ldirectord 使用 ipvsadm 命令恢复了 realserver1 服务器的记录。此外还可使用“watch ipvsadm -L -n”命令动态观察。

参考文献:

章文嵩：《Linux 服务器集群系统》

<http://www.linuxvirtualserver.org/zh/lvs1.html~lvs4.html>

林凡：《集群的可扩展性及其分布式体系结构》

http://www-128.ibm.com/developerworks/cn/linux/cluster/cluster_system/lvs/part1/index.htm

|

秦刘，兰巨龙，杨帅，智英建 《动态反馈负载均衡在 LVS 集群中的设计与实现》

<http://www.chinaaet.com/jishu/jslw/2008-07-14/13331.shtml>



网
网
之
网
4

3
.
0

5

Linux 中的 XEN 虚拟化技术

5.1 Xen 虚拟化概述

Xen 是剑桥大学计算机实验室开发的一个开放源代码的虚拟化软件，功能与 VMware 类似。

此外 Xen 是基于 Linux 内核的虚拟程序，使用一个叫 hypervisor 的软件层来调节操作系统对真实硬件的访问和控制，实现在一套物理硬件上安装运行多个虚拟操作系统。

Xen 同时支持完全虚拟化（Full-Virtualized）和半虚拟化（Para-Virtualized）两种运行模式。完全虚拟化提供底层物理系统的全部抽象化，虚拟系统不需要修改操作系统内核就可以直接运行在 Xen 上面，VMware 是完全虚拟化技术的代表。

完全虚拟化功能需要依赖于 CPU 指令集的支持，例如：Intel CPU 的 VT 指令集，AMD CPU 的 SVM 指令集。如果服务器 CPU 不支持虚拟化指令集，Xen 只能以半虚拟化模式运行，在 Xen 上“Full-Virtualized”这个选项是灰色的，即功能不能启用。

半虚拟化需要对运行在 Xen 上的操作系统内核进行修改，这些修改提高了操作系统与 hypervisor 间通讯的有效性和性能，所以半虚拟化的性能比全虚拟化更佳。

只支持半虚拟化的 Xen 不支持安装运行无法修改系统内核的 Windows 系列操作系统，但支持安装运行经过修改的 Linux 系统。

Xen 启动运行后，第一个虚拟的操作系统 Domain 0，就是 Xen 本身。Domain 0 是其它虚拟主机的管理者和控制者，Domain 0 可以构建其它的更多的 Domain，并管理虚拟设备，它还能执行管理任务，比如虚拟机的休眠、唤醒和迁移其它虚拟机。

5.2 Xen 的安装和配置

5.2.1 安装 Xen

Red Hat 公司在 RHEL5 版本的 Linux 操作系统中正式加入了对 Xen 虚拟技术的支持。而在 RHEL5 之前的 Linux 系统中安装 Xen 需要进行编译内核的工作，安装过程较为繁琐和复杂。

在 RHEL5 中安装 Xen 有多种方式：

方法一：在 RHEL5 安装时输入包含支持 virtualization 的功能的安装号，系统默认会自动安装 Xen 相关软件包。

方法二：在 RHEL5 的安装光盘中已包含全部 Xen 的软件包，所以完全可以手动安装 RHEL5 安装光盘中的 Xen 相关软件包，由于需要安装软件包较多，此方法比较繁琐，不建议初学者使用。

方法三：在能连入互联网的前提下，使用 RHEL5 订阅号，利用 yum 工具安装 Xen 内核补丁、Xen 虚拟机、virt-manager 虚拟化管理工具。

方法四：利用 RHEL5 的 DVD 安装光盘构建本地 yum 安装源，利用 yum 工具安装 Xen 内核补丁、Xen 虚拟机、virt-manager 虚拟化管理工具。

这里推荐使用第四种方法进行安装。

5

Linux 中的 XEN 虚拟化技术

现在以一个已安装 GNOME 图形桌面，已安装 Apache，但没有安装任何 Xen 软件的 RHEL5 系统为例讲解如何安装 Xen 虚拟化软件。

首先参照本阅读手册第一章的内容构建本地 yum 安装源，过程这里不再赘述。构建完成后执行以下命令安装 Xen 内核补丁、Xen 虚拟机、virt-manager 虚拟化管理工具

```
[root@localhost ~]# yum install kernel-xen xen virt-manager
```

5.2.2 配置 Xen 启动

修改 grub.conf 文件，调整 kernel 启动顺序，默认以支持 Xen 的内核启动 Linux 系统。

```
[root@localhost ~]# vi /etc/grub.conf
```

将“default=1”改为“default=0”，grub.conf 文件内容具体内容为：

```
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Server (2.6.18-8.el5xen)
    root (hd0,0)
    kernel /xen.gz-2.6.18-8.el5
    module /vmlinuz-2.6.18-8.el5xen ro root=/dev/VolGroup00/LogVol00
    module /initrd-2.6.18-8.el5xen.img
title Red Hat Enterprise Linux Server (2.6.18-8.el5)
    root (hd0,0)
    kernel /vmlinuz-2.6.18-8.el5 ro root=/dev/VolGroup00/LogVol00
    initrd /initrd-2.6.18-8.el5.img
```

重新启动系统

```
[root@localhost ~]# init 6
```

5.2.3 Xen 服务控制命令

Xen 是由一个叫做 xend 后台守护进程维护，要运行虚拟系统，必须先将其启动。Xen 的配置文件是/etc/xen/xend-config.sxp，内容包括宿主系统的类型，网络的连接结构、宿主操作系统的资源使用设定，以及 vnc 连接的一些内容，一般不需要对其进行设置，如果要添加新的网络设备（比如新的虚拟网卡），则需要在 xend-config.sxp 文件中添加新的设备内容。

启动 xend 的命令

```
[root@localhost ~]# /etc/init.d/xend start
```



停止 xend 的命令

```
[root@localhost ~]# /etc/init.d/xend stop
```

重新启动 xend 的命令

```
[root@localhost ~]# /etc/init.d/xend restart
```

将 xend 服务设置成开机自启动

```
[root@localhost ~]# chkconfig --add xend
[root@localhost ~]# chkconfig --add xenddomains
[root@localhost ~]# chkconfig --level 345 xend on
[root@localhost ~]# chkconfig --level 345 xenddomains on
```

5.3 创建 Xen 虚拟系统

5.3.1 创建 Xen 虚拟系统安装树

与 EMC VMware 和微软 Virtual Server 一样, Xen 在完全虚拟化的环境中可以支持从 ISO 光盘镜像文件或物理光驱作为安装源来引导安装系统,但在半虚拟化环境中必须使用安装树进行安装。安装树是一个包含所有系统安装文件及相关程序的目录,这个目录通过 HTTP、FTP、NFS 进行发布。

这里我们将要在 Xen 半虚拟化环境下安装一个 RHEL5 系统,以 RHEL5 为例介绍制作 HTTP 安装树的方法:

1. 安装并启动 Apache 服务

Apache 的安装过程省略,这里假定 Apache 网页存放目录为/var/www/html

2. 将 RHEL5 安装映像文件复制到 Linux 系统中

将 RHEL5.iso 文件复制到/tmp/iso/下

3. 将 RHEL5 安装映像文件挂载到 Apache 根目下

```
[root@localhost ~]# mount -o loop /tmp/iso/RHEL5.iso /var/www/html
```

5.3.2 使用字符工具 virt-install 创建 Xen 虚拟系统

1. 创建块设备文件存储目录

```
[root@localhost ~]# mkdir /vmdisk
```



5

Linux 中的 XEN 虚拟化技术

2. 执行 virt-install 安装程序

```
[root@localhost ~]# virt-install
```

3. 配置虚拟系统环境

运行 virt-install 程序后将进入 Xen 的字符引导安装界面，按提示信息输入虚拟系统配置。

1) 输入 Xen 虚拟系统的名称

What is the name of your virtual machine? **vmrhel5**

2) 输入需要分配给虚拟系统的内存

How much RAM should be allocated (in megabytes)? **300**

3) 输入虚拟块设备的路径

What would you like to use as the disk (path)? **/vmdisk/vmrhel5**

4) 输入块设备的大小

How large would you like the disk (/vmdisk/vmrhel5) to be (in gigabytes)? **6**

5) 是否使用图形安装界面

Would you like to enable graphics support? (yes or no) **no**

6) 输入安装树路径

What is the install location? **http://10.0.0.150**

4. 安装 RHEL5 系统

1) 选择安装向导语言

字符界面下只支持英文，所以保持默认选项“English”，直接回车

```

Welcome to Red Hat Enterprise Linux Server

      +-----+ Choose a Language +-----+
      |
      | What language would you like to use
      | during the installation process?
      |
      | Catalan          ^
      | Chinese(Simplified) :
      | Chinese(Traditional) #
      | Croatian         :
      | Czech            :
      | Danish           :
      | Dutch            :
      | English          v
      |
      |          +----+
      |          | OK |
      |          +----+
      |
      +-----+

<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen

```

图5.1 选择安装向导语言

2) 设定 TCP/IP 选项

手动设定 IPv4 网络设定，这个步骤非常关键，如果 IP 设置不正确，将无法连接到安装树，导致无法安装系统

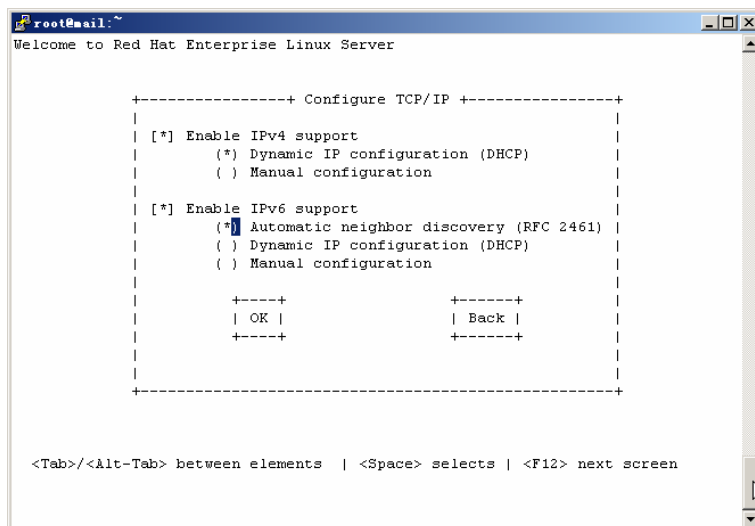


图5.2 设定 TCP/IP

3) 进入 Linux 字符安装向导

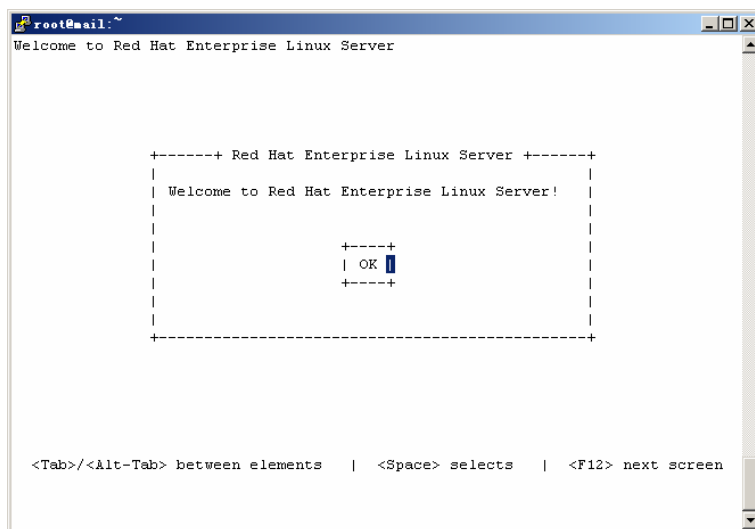


图5.3 进入 RHEL5 字符安装引导欢迎界面

根据安装向导提示，设置 RHEL5 安装选项，安装过程与在真实硬件环境的安装过程一致

5.3.3 使用图形工具 virt-manager 创建 Xen 虚拟系统

1 打开 virt-manager 管理工具

2 连接 Xen 服务器

5

Linux 中的 XEN 虚拟化技术

在 Linux 图形桌面中打开“应用程序”->“系统工具”->“Virtual Machine Manager”



图5.4 打开 virt-manager 管理工具

3 virt-manage 图形管理界面

在 virt-manage 图形管理界面中点击“文件”->“新系统”，打开创建虚拟系统的向导

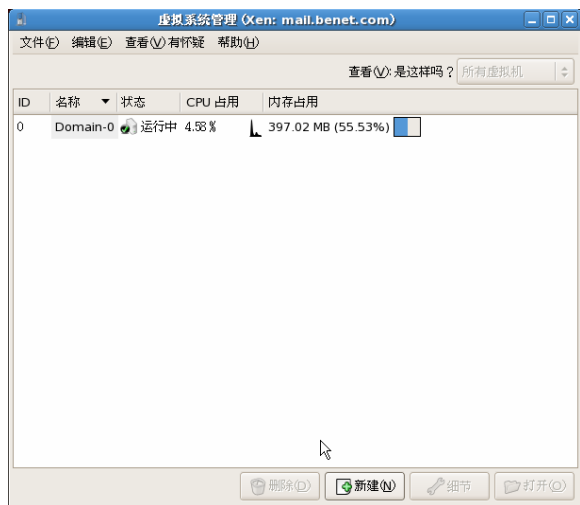


图5.6 virt-manager 图形管理界面

5 命名虚拟系统

在“为虚拟系统命名”界面输入要创建的虚拟系统名称，如“vmrhel5”，点击“前进”按钮

保持默认，点击“连接”按钮，连接至本地 Xen 服务器



图5.5 连接本地 Xen 服务器

4 启动创建新虚拟向导

无选择项，点击“前进”按钮，进入下一步操作

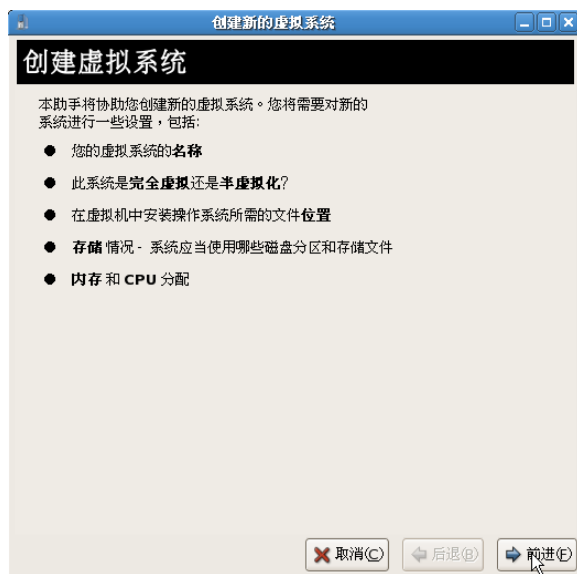


图5.7 创建虚拟系统引导界面

6 选择虚拟化方式

Xen 会自动检测 CPU 是否支持虚拟化，如果 CPU 支持完全虚拟化，则“完全虚拟化”的选项不为虚。选择“半虚拟化”，点击“前进”按钮





图5.8 命令虚拟系统



图5.9 选择虚拟化方式

7 指定安装树路径

输入安装树路径“http://10.0.0.150”。点击“前进”按钮



图5.10 定位安装介质

8 分配存储空间

选择“简单文件”，在文件位置输入“/vmdisk/vmrhel5”，文件大小设定为“5000”



图5.11 分配存储空间

9 分配内存和 CPU

根据当前物理内存大小设定分配给虚拟系统的内存

10 Xen 虚拟系统设置完成

此界面显示最终的虚拟系统配置内容，点击“前进”按钮将启动 RHEL5 安装引导程序，进入系统安装阶段。

5

Linux 中的 XEN 虚拟化技术

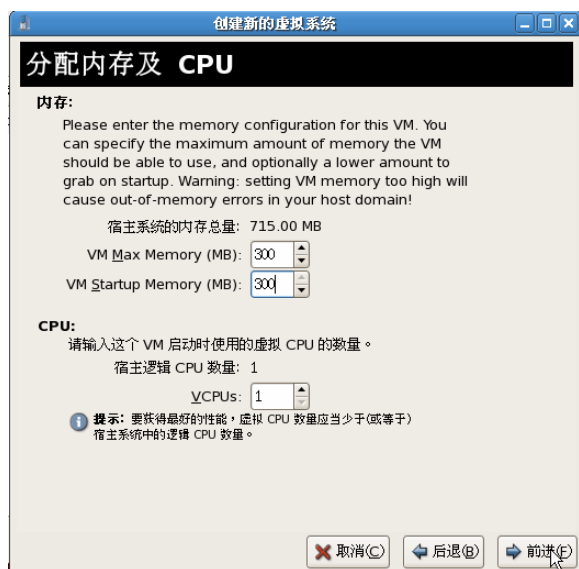


图5.12 分配内存和 CPU



图5.13 显示最终设定信息

11 启动 RHEI5 安装引导程序

按照引导界面的指导安装操作系统

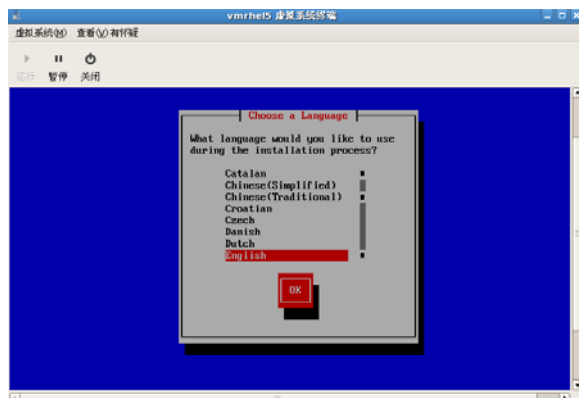


图5.14 开始安装 RHEL5 操作系统

5.4 管理 Xen 虚拟系统

5.4.1 使用 xm 命令管理 Xen 虚拟机

1. 显示运行状态

在命令行下使用“xm list”命令显示 Xen 虚拟系统当前运行的状况

```
[root@localhost ~]# xm list
```

Name	ID	Mem(MiB)	VCPUs	State	Time(s)
Domain-0	0	683	1	r-----	815.2
vmrhe15	2	299	1	-b----	212.8

2. 连接虚拟系统

使用“xm console”命令通过字符界面连接到运行中的 vmrhe15 虚拟系统。

```
[root@localhost ~]# xm console vmrhe15
```

3. 启动虚拟系统

使用“xm create”命令启动 vmrhe15 虚拟系统

```
[root@localhost ~]# xm create /etc/xen/vmrhe15
```

4. 关闭虚拟系统

使用“xm shutdown”命令启动 vmrhe15 虚拟系统

```
[root@localhost ~]# xm shutdown vmrhe15
```

5. 重启虚拟系统

使用“xm reboot”命令重启 vmrhe15 虚拟系统

```
[root@localhost ~]# xm reboot vmrhe15
```

5.4.2 使用 virt-manager 图形工具管理 Xen 虚拟机

在 Linux 图形桌面 X-Windows 中打开“应用程序”->“系统工具”->“Virtual Machine Manager”，选择“本地 Xen 宿主”，点击“连接”，启动 virt-manager 图形管理工具，如图 5.15 所示

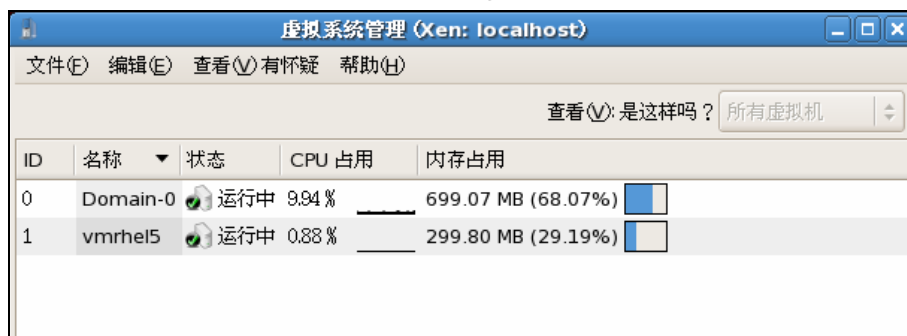


图 5.15 virt-manager 图形管理工具

在 virt-manager 中右键单击要管理的虚拟系统，在弹出菜单中选择“Details”将打开“虚拟系统状态窗口”，如图 5.16 所示，在此窗口中可以查看虚拟系统的名称、CPU 和内存占用情况，还可对虚拟系统进行“暂停”和“关闭”的操作。

5

Linux 中的 XEN 虚拟化技术

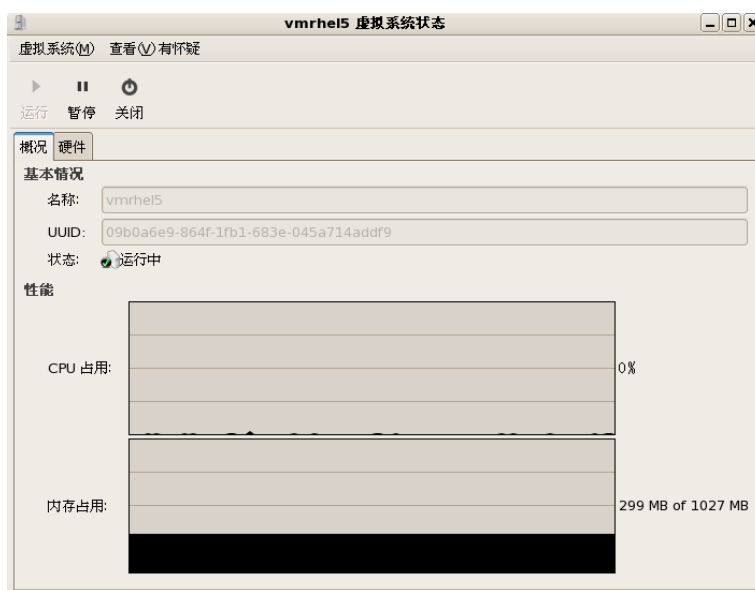


图 5.16 虚拟系统状态

在“虚拟系统状态窗口”，点击“硬件”选项卡可查看并修改虚拟系统的硬件配置参数。



图 5.17 硬件参数设置

在 `virt-manager` 中右键单击要管理的虚拟系统，在弹出菜单中选择“打开”，将打开虚拟系统的终端，并且还可对虚拟系统进行“运行”、“关闭”、“暂停”和“保存”的操作。

5.4.3 配置虚拟系统随服务器启动

当物理服务器重启时，所有的虚拟系统会自动停止。但当物理服务器重启完成时，这些虚拟系统并不会自动启动，而需要手动的方式进行启动。

要设置 Xen 的虚拟系统随服务器启动，首先要了解 Xen 上已安装的虚拟系统的配置文件存放位置。通过 `virt-install` 命令或者使用 `virt-manager` 图形管理工具在 Xen 服务器上创建的虚拟系统，其配置文件默认保存在 `/etc/xen` 目录下。


```
[root@localhost ~]# cat /etc/xen/vmrhel5
# Automatically generated xen config file
name = "vmrhel5"
memory = "300"
disk = [ 'tap:aio:/vmdisk/vmrhel5,xvda,w', ]
vif = [ 'mac=00:16:3e:25:6f:e3, bridge=xenbr0', ]

uuid = "09b0a6e9-864f-1fb1-683e-045a714addf9"
bootloader="/usr/bin/pygrub"
vcpus=1
on_reboot    = 'restart'
on_crash     = 'restart'
```

要使虚拟系统能够随服务器启动，其操作很简单，只需要将需要随服务器启动的虚拟系统的配置文件放到/etc/xen/auto 目录中，Xen 会在启动时将 auto 目录下的虚拟系统启动。

```
[root@localhost ~]# mv /etc/xen/vmrhel5 /etc/xen/auto/
```



四
四
之
四
十

6

使用 `rsync` 远程文件同步

3
.
0

6.1 Rsync 概述

Rsync (Remote Sync, 远程同步) 是一个快速的文件同步和传输工具, 主要用于快速、安全、高效的数据备份 (如服务器间的镜像同步)。

Rsync 程序类似于 rcp (Remote Copy, 远程文件复制) 命令的工作方式, 但是拥有更多的增强特性。Rsync 使用 rsync 远程更新 (remote-update) 协议, 根据校验和搜索算法 (checksum-search algorithm) 仅传输新增或更改过的文件, 且支持数据压缩, 从而大大提高了文件同步的速度。

Rsync 的官方网站是 <http://rsync.samba.org>, 目前的最新版本为 rsync-3.0.5pre1。

6.2 配置用于 rsync 同步的远程主机

用于 rsync 同步的远程主机, 最常见的包括两种服务实现方式: 其一, 基于远程 shell 的程序, 如 OpenSSH; 其二, rsync 程序自身的“daemon”服务器模式。

6.2.1 基于 OpenSSH 的模式

使用基于 SSH 的服务方式时, 服务器端无需配置 rsync, 只要安装好 openssh-server, 并配置启动 sshd 服务即可。这种方式可以直接以 Linux 系统用户进行 rsync 验证, 非常方便和灵活。

在 RHEL5 系统中, 使用自带的 openssh-server 软件包, 按默认配置即可。基于安全性考虑, 可以根据需要对配置文件做适当的调整。

```
[root@www1 ~]# useradd sshuser           //建立一个用于远程备份的系统用户
[root@www1 ~]# passwd sshuser
[root@www1 ~]# vi /etc/ssh/sshd_config
PermitRootLogin no
UseDNS no
AllowUsers sshuser           //添加该行配置, 仅允许用户 sshuser 远程登录
[root@www1 ~]# service sshd start
[root@www1 ~]# netstat -anp | grep :22
tcp        0      0 0 :::22          :::*            LISTEN      1689/sshd
[root@www1 ~]# chkconfig --level 2345 sshd on
```

6.2.2 基于 rsync 的--daemon 模式

使用 rsync 程序的服务器模式时, 需要为 rsync 程序配置 rsyncd.conf 文件, 并以“--daemon”的选项将 rsync 运行为服务进程。这种方式可以使用匿名用户, 或者基于文本文件的用户数据库进行验证。通常用于建立只读模式的下载备份源。

在 RHEL5 系统中, 默认已安装有 rsync-2.6.8-3.1 软件包。主要配置过程如下:

6

使用 rsync 远程文件同步

1. 建立 rsyncd.conf 配置文件

rsyncd.conf 文件位于/etc/目录下（缺省没有该文件，需手工建立），配置项的格式类似于 Samba 服务的配置，可以使用“man rsyncd.conf”命令查看帮助信息。

```
[root@www1 ~]# vi /etc/rsyncd.conf
uid = nobody
gid = nobody
use chroot = yes
address = 192.168.0.11          //监听服务的 IP 地址
port 873          //监听服务的端口，缺省为 873
log file = /var/log/rsyncd.log
pid file = /var/run/rsyncd.pid
hosts allow = 192.168.0.0/24    //允许访问的客户端地址
[webroot]          //发布的共享模块名
    path = /var/www/html        //发布用于远程备份的目录位置
    comment = Web Root Directory of www1.benet.com
    read only = yes
    dont compress = *.gz *.bz2 *.tgz *.zip *.rar *.z          //不作压缩的文件
    auth users = backupper      //授权用户名，若使用匿名方式，此行和下行可以省略
    secrets file = /etc/rsyncd_users.db          //授权用户的数据库文件位置
```

2. 建立 rsync 用户数据库

创建一个用于备份操作的用户，并设置好可靠的密码。该用户必须对备份源目录（/var/www/html）有读取权限（other 组），才能进行远程同步。

```
[root@www1 ~]# vi /etc/rsyncd_users.db
backupper:pwd@123
[root@www1 ~]# chmod 600 /etc/rsyncd_users.db
```

3. 启动 rsync 服务

若 rsync 服务需要被频繁使用，建议采用独立守护进程的方式运行。若 rsync 服务的使用频率并不高，则可以交由 xinetd 超级服务器管理（需要安装有 xinetd-2.3.14-10.el5.i386.rpm 软件包），以节省系统资源。以下两种方式任选一种即可。

1) 独立启动的守护进程

```
[root@www1 ~]# rsync --daemon
[root@www1 ~]# netstat -anp | grep rsync
tcp        0      0 192.168.0.11:873      0.0.0.0:*        LISTEN      32668/rsync
[root@www1 ~]# echo "/usr/bin/rsync --daemon &" >> /etc/rc.local
```

2) Xinetd 管理的客户进程

```
[root@www1 ~]# vi /etc/xinetd.d/rsync
service rsync
{
    disable = no
    socket_type      = stream
    wait            = no
    user            = root
    server          = /usr/bin/rsync
    server_args     = --daemon
    log_on_failure += USERID
}
[root@www1 ~]# chkconfig --list rsync
rsync          on
[root@www1 ~]# service xinetd restart
```

6.3 使用 rsync 文件同步工具

将远程主机的备份源目录同步到本地，称为下行（下载）同步。将本地的备份源目录同步到远程主机中，称为上行（上传）同步。**rsync** 也可以用于本地目录之间的同步，但是不能直接用于远程主机和远程主机之间的目录同步。

在使用 **rsync** 文件同步时，命令的基本格式如下：

```
rsync [选项] 备份源 备份目标
```

以下列出几个较常用的命令选项及其含义：

- **-a** 使用归档（archive）模式，保留文件原有的权限、属性、属主等信息，等同于使用“**-rlptgoD**”等多个选项的组合
- **-l** 符号(软)连接文件仍然复制为符号连接



6

使用 rsync 远程文件同步

- -H 保留硬连接文件
- -r 递归模式，包含目录及子目录中所有文件
- -v 显示同步过程的详细（verbose）信息
- -z 在传输文件时进行压缩（compress）
- -o 保留文件的属主标记（仅超级用户使用）
- -g 保留文件的属组标记（仅超级用户使用）
- -t 保留文件的时间标记
- -p 保留文件的权限标记
- -D 保留设备文件及其他特殊文件
- --delete 删除目标文件夹有而源文件夹中没有的文件
- --checksum 根据校验和来决定是否跳过文件（而不是根据文件大小、修改时间）

在使用 rsync 命令时，根据远程主机使用的服务方式不同，主机地址与备份目录之间的采用了两种不同的分隔符形式。

- 当连接 openssh-server 服务时，使用一个冒号“:”分隔。目录名对应远程主机中的绝对路径，以远程主机的系统用户进行验证。例如：

```
[root@www2 ~]# rsync -avz sshuser@www1.benet.com:/var/www/html/ /tmp/
```

- 当连接 rsync --daemon 服务时，使用两个冒号“::”分隔（或者使用斜杠“/”分隔，同时在主机地址前加“rsync://”前缀）。目录名对应远程主机发布的备份模块名，以远程主机指定的 rsyncd_users.db 文件中的用户进行验证（允许匿名）。例如：

```
[root@www2 ~]# rsync -avz backupper@192.168.0.11::webroot /tmp/
```

或者

```
[root@www2 ~]# rsync -avz rsync://backupper@192.168.0.11/webroot /tmp/
```

6.3.1 rsync 命令的常见用法示例

1. 查看 rsync 服务器可用的备份模块列表

```
[root@www2 ~]# rsync rsync://192.168.0.11
webroot          Web Root Directory of www1.benet.com
```

2. 查看 rsync 服务器中 webroot 备份模块下的目录及文件

如果是匿名 rsync 服务，则无需用户名及密码。否则需要指明授权客户端使用的用户名（如



backuper), 根据提示输入正确的密码即可。

```
[root@www2 ~]# rsync rsync://backuper@192.168.0.11/webroot
Password:      //根据提示输入 backuper 用户的密码
drwxr-xr-x      4096 2008/11/03 22:06:27 .
-rwxrwxr-x      42 2008/10/24 19:03:17 index.php
drwxrwxr-x      4096 2008/10/24 19:10:40 bbs
drwxr-xr-x      4096 2008/10/24 19:45:40 cacti
drwxrwxr-x      4096 2008/10/24 19:10:45 ucenter
drwxr-xr-x      4096 2008/10/26 19:04:05 webmail
```

3. 将 rsync 服务器中 webroot 备份模块下的内容同步备份到本地（下行）

```
[root@www2 ~]# mkdir -p /webbak/www1/
[root@www2 ~]# rsync -aHvz --delete backuper@192.168.0.11::webroot/
/webbak/www1/
```

4. 将本地目录中的内容同步备份到远程的 rsync 服务器（上行）

上传同步文件时需要用户对该目录有可写权限。

```
[root@www2 ~]# rsync -arvz /webbak/www1/ webmaster@192.168.0.11:/var/www/html/
```

6.3.2 结合 crond 实现定期同步（下行）

定期同步通常适用于对实时性要求不高的冗余备份。可以结合 Linux 系统的 crond 服务，安排有计划的、可周期性自动执行的 rsync 同步任务。

1. 建立远程同步文件的 Shell 脚本

```
[root@www2 ~]# vi /opt/rsync_www1.sh
#!/bin/bash
HOST="192.168.0.11"
SRC="webroot/"
USER="backuper"
export RSYNC_PASSWORD="pwd@123"
DST="/webbak/www1/"
ARGS="-aHvz --delete --checksum"
mkdir -p $DST
```

6

使用 rsync 远程文件同步

```
/usr/bin/rsync $ARGS $USER@$HOST::$SRC $DST
[root@www2 ~]# chmod a+x /opt/rsync_www1.sh
[root@www2 ~]# /opt/rsync_www1.sh
```

2. 配置 crontab 计划任务

设置每隔 10 分钟自动执行一次同步。

```
[root@www2 ~]# crontab -e
*/10 * * * * /opt/rsync_www1.sh
[root@www2 ~]# chkconfig --level 2345 crond on
[root@www2 ~]# service crond start
```



请注意

在使用 rsync 下载或上传同步时，若再次重复执行同一条命令，则会自动进行**增量更新**，完全重复的内容不会再拷贝，以加快同步速度。

6.3.3 结合 inotify-tools 实现实时同步（上行）

实时同步通常适用于有一定实时性要求的场合，如网站镜像、开发数据备份等。可以结合 Linux 内核的 inotify 机制，根据备份源目录的变化情况，安排触发更新式的 rsync 同步任务。

Linux 内核自 2.6.13 版本以后提供了 inotify API 编程接口，用于监控文件系统事件，如文件存取、删除、移动、修改等。在 RHEL5 系统中，默认已经编入了 inotify 机制（可以检查是否存在 /proc/sys/fs/inotify/ 目录）。

inotify-tools 是在 shell 环境中使用 inotify 功能的一套辅助工具（需要另行下载）。使用 inotify-tools 提供的 inotifywait、inotifywatch 工具，编写合适的 shell 脚本程序，可以完成许多触发式的系统管理任务。

通过监控本地源目录中的文件修改、删除、新建……等变化，可以实时触发 rsync 命令，与远程的目标主机保持同步（基于安全性考虑，上传同步建议只在内部网络使用）。

1. 配置远程的备份目标主机

基于 ssh 的上传同步通常需要交互式登录验证，在 shell 脚本中不便于使用；而基于 rsync 用户的上传同步需要调整目录及用户权限，不便于保留同步后的目录/文件的属性等信息。本例中将使用 RHEL5 系统自带的 nfs 服务（如图 6.1 所示），实现备份目标（www2 主机）目录的发



布，以允许源主机同步写入。

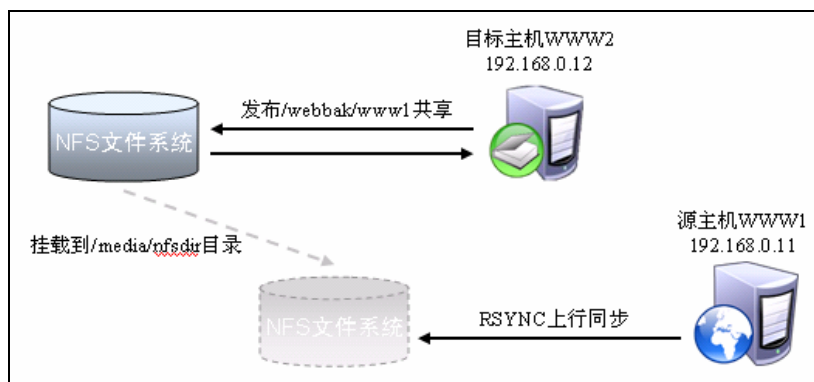


图6.1 Rsync + NFS 上行同步示意图

配置 NFS 发布共享目录的过程参考如下。

```
[root@www2 ~]# mkdir -p /webbak/www1
[root@www2 ~]# vi /etc/exports
/webbak/www1    192.168.0.11(rw,no_root_squash)
[root@www2 ~]# chkconfig --level 2345 portmap on
[root@www2 ~]# chkconfig --level 2345 nfs on
[root@www2 ~]# service portmap start
[root@www2 ~]# service nfs start
```

2. 配置本地的备份源主机

1) 安装 inotify-tools

在备份源主机上下载、安装 inotify-tools。按默认的配置完成安装后，inotifywait、inotifywatch 命令均在/usr/local/bin/inotifywait 目录下。可以使用 man 命令查看两个命令的详细帮助手册。

```
[root@www1 ~]# wget
http://download.sourceforge.net/inotify-tools/inotify-tools-3.13.tar.gz
[root@www1 ~]# tar zxvf inotify-tools-3.13.tar.gz
[root@www1 ~]# cd inotify-tools-3.13
[root@www1 ~]# ./configure
[root@www1 ~]# make && make install
```

2) 挂载 nfs 共享目录

6

使用 rsync 远程文件同步

在备份源主机 **www1** 上挂载目标主机 **www2** 发布的 **nfs** 共享目录。要使用 **nfs** 共享，需要先启动 **portmap** 服务。

```
[root@www1 ~]# chkconfig --level 2345 portmap on
[root@www1 ~]# service portmap start
[root@www1 ~]# vi /etc/fstab
192.168.0.12:/webbak/www1          /media/nfsdir          nfs          defaults0 0
[root@www1 ~]# mkdir -p /media/nfsdir
[root@www1 ~]# mount /media/nfsdir
```

3) 编写 inotify 监控及触发同步脚本

在脚本中使用 **inotifywait** 命令，监控 **/var/www/html/** 目录下的文件修改、创建、移动、删除事件，当出现监控的事件时立即触发 **rsync** 命令执行增量更新。

```
[root@www1 ~]# vi /opt/in_rsync.sh
#!/bin/bash
inotifywait -mrq -e modify,create,move,delete /var/www/html/ | while read DIR EVENT FILE
do
    rsync -aHvz --delete /var/www/html/ /media/nfsdir
done
```

4) 执行触发同步脚本

```
[root@www1 ~]# chmod 700 /opt/in_rsync.sh
[root@www1 ~]# echo "/opt/in_rsync.sh &" >> /etc/rc.local
[root@www1 ~]# /opt/in_rsync.sh &
```

3. 测试实时同步效果

在源主机 **www1** 中，对 **/var/www/html/** 目录进行增、删文件/目录、修改文件内容等操作，查看目标主机 **www2** 中 **/webbak/** 目录下的相应变化。

网
网
之
网
4

3
.
0

7

Nginx 网站及负载均衡系统

7.1 Nginx 概述

Nginx（发音为[engine x]）是一个高性能的轻量级 HTTP 和反向代理服务器软件，由俄罗斯的 Igor Sysoev 开发。Nginx 专为性能优化而开发，最知名的优点是它的稳定性和低系统资源消耗，对 HTTP 并发连接的处理能力强（单台物理服务器可支持 3~5 万个并发请求）。在 Internet 中，大量的虚拟主机站点和访问负载较高的网站（如俄罗斯的 Rambler.ru）都正在使用 Nginx 提供服务。

根据 2008 年 11 月份 Netcraft.com 的统计报告(<http://survey.netcraft.com/Reports/200811/>)，全球已有 300 多万的 Web 站点在使用 Nginx，而且这一数字正在快速上升。国内如新浪博客、网易新闻、新华网 RSS、六间房视频、迅雷安全中心、腾讯 3G 下载.....等站点，都已经开始在使用 Nginx 服务。

Nginx 的官方站点：<http://sysoev.ru/en/>、<http://nginx.net>

Nginx 中文 Wiki：<http://wiki.codemongers.com/NginxChs>

Nginx 的最新稳定版本为 0.6.32，开发版本为 0.7.20，下载地址参考如下：

<http://sysoev.ru/nginx/nginx-0.6.32.tar.gz>

<http://sysoev.ru/nginx/nginx-0.7.20.tar.gz>

7.2 Nginx 的安装及运行控制

7.2.1 Nginx 的安装

1. 安装 pcre 支持软件

pcre 软件包的作用主要是为 nginx 提供兼容 perl 的正则表达式库，此处以使用 RHEL5 光盘自带的 rpm 包安装为例，需要安装 pcre、pcre-devel 两个软件包。

```
[root@localhost ~]# mkdir -p /media/cdrom
[root@localhost ~]# mount /dev/cdrom /media/cdrom
mount: block device /dev/cdrom is write-protected, mounting read-only
[root@localhost ~]# cd /media/cdrom/Server/
[root@localhost Server]# rpm -ivh pcre-6.6-1.1.i386.rpm pcre-devel-6.6-1.1.i386.rpm
warning: pcre-6.6-1.1.i386.rpm: Header V3 DSA signature: NOKEY, key ID 37017186
Preparing...                               ##### [100%]
 1:pcre                                     #####
[ 50%]
 2:pcre-devel                             #####
[100%]
```

2. 安装 nginx 软件包

Nginx 软件包的安装过程相对比较简单，此处以最新开发版 nginx-0.7.20 为例。

```
[root@localhost ~]# wget http://sysoev.ru/nginx/nginx-0.7.20.tar.gz
[root@localhost ~]# tar zxvf nginx-0.7.20.tar.gz
[root@localhost ~]# cd nginx-0.7.20
[root@localhost nginx-0.7.20]# ./configure --with-http_stub_status_module
//以上"--with-http_stub_status_module"选项用于启用站点状态统计模块
//其他更多配置选项可以使用"./configure --help"命令查看
[root@localhost nginx-0.7.20]# make
[root@localhost nginx-0.7.20]# make install
```

Nginx 默认安装在/usr/local/nginx 目录下，为了方便管理员使用，可以添加一个到 nginx 主程序的符号链接。

```
[root@localhost nginx-0.7.20]# ln -sf /usr/local/nginx/sbin/nginx /usr/sbin/
```

7.2.2 Nginx 运行控制

1. 检查配置文件有无语法错误

nginx 命令的“-t”选项可用于检查 nginx.conf 配置文件有无语法错误。

```
[root@localhost ~]# nginx -t
2008/11/03 13:03:54 [info] 24195#0: the configuration file /usr/local/nginx/conf/nginx.conf
syntax is ok
2008/11/03 13:03:54 [info] 24195#0: the configuration file /usr/local/nginx/conf/nginx.conf
was tested successfully
```

如果配置文件不在默认位置，可以使用如“**nginx -t -c /etc/my_nginx.conf**”的形式，用“-c”选项指定配置文件的路径即可。

2. 启动服务

```
[root@localhost ~]# service httpd stop //为了避免冲突，先关闭系统自带的 httpd 服务
[root@localhost ~]# nginx
[root@localhost ~]# netstat -anpt | grep :80
tcp        0      0 0.0.0.0:80          0.0.0.0:*          LISTEN     24135/nginx: master
```

不带任何选项的 nginx 命令将使用默认的配置文件加载服务，若需要使用其他配置文件，则需要增加“-c”选项，如“**nginx -c /etc/my_nginx.conf**”。

成功启动 nginx 服务后，可以在浏览器中查看初始 Web 页面。



7

Nginx 网站及负载均衡系统

——在命令行下可以使用 `elinks`、`lynx` 等文本浏览器工具，如：

```
[root@localhost ~]# elinks http://localhost/
Welcome to nginx!
```

3. 使用系统信号控制 nginx 进程

Nginx 通过识别标准的系统信号来控制进程状态。如 `HUP`、`KILL`、`QUIT`、`TERM` 等（可以使用“`kill -l`”命令查看可用的进程控制信号列表）。

```
[root@localhost ~]# killall -s HUP nginx      //重新加载配置文件，等同于“killall -1 nginx”
[root@localhost ~]# killall -s QUIT nginx     //安全退出，等同于“kill -3 nginx”
[root@localhost ~]# killall -s TERM nginx     //快速退出，不等待处理完当前连接
```

4. 添加 nginx 服务控制脚本

考虑到 Red Hat Linux 的使用习惯，可以添加一个 nginx 服务脚本，以便使用 `chkconfig` 和 `service` 命令管理 nginx 服务。

```
[root@localhost ~]# vi /etc/init.d/nginx
#!/bin/sh
# chkconfig: - 99 20
# description: Nginx Service Control Script
#           //如需使用 chkconfig 管理，注意以上两行内容不可少
case "$1" in
    start)
        /usr/local/nginx/sbin/nginx
        ;;
    stop)
        /usr/bin/killall -s QUIT nginx
        ;;
    restart)
        $0 stop
        $0 start
        ;;
    reload)
        /usr/bin/killall -s HUP nginx
        ;;
    *)
        echo "Usage: $0 {start|stop|restart|reload}"
        exit 1
esac
exit 0
[root@localhost ~]# chmod a+x /etc/init.d/nginx
```

```
[root@localhost ~]# chkconfig --add nginx
[root@localhost ~]# chkconfig --level 2345 nginx on
```

以上 nginx 脚本可以通过 service 命令进行管理控制。

```
[root@localhost ~]# service nginx stop
[root@localhost ~]# service nginx start
[root@localhost ~]# service nginx reload
```

7.3 Nginx 网站服务配置示例

7.3.1 常规静态网站配置

本例中的 Nginx 静态网站配置，列出了实现基本 HTTP 服务的常用选项，包括监听地址/端口、主机名、网站根目录、默认字符编码、日志文件、索引文件以及部分优化设置等。

```
[root@localhost ~]# vi /usr/local/nginx/conf/nginx.conf
user nobody;
worker_processes 10;          //工作进程数
error_log logs/error.log;     //“错误日志”文件的位置
events {
    use epoll;                //处理连接（I/O）的一种方法，用于 2.6 以上的内核
    worker_connections 2048;  //每个工作进程可接受的连接数
}
http {
    include mime.types;
    default_type application/octet-stream;
    log_format main '$remote_addr - $remote_user [$time_local] $request '
        '$status' $body_bytes_sent "$http_referer" '
        "$http_user_agent" "$http_x_forwarded_for";
    access_log logs/access.log main; //“访问日志”文件的位置和格式
    sendfile on;
    tcp_nopush on;
    tcp_nodelay on;
    keepalive_timeout 65;
    server {
        listen 80; //设置监听端口，注意不要与别的服务（如 httpd）冲突
        server_name www.benet.com; //站点的 FQDN 名称
        charset utf-8; //站点页面文件使用的默认字符编码
        access_log logs/host.access.log main;
        location / {
            root html; //网站根目录，可以使用绝对路径
            index index.html index.php; //目录索引文件名
```

7

Nginx 网站及负载均衡系统

```

    }
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root html;
    }
}
[root@localhost ~]# service nginx restart

```

设置监听的端口时，可以使用“listen 192.168.0.11:80;”的形式同时指定监听的 IP 地址。

设置网站根目录时，可以使用“root /var/www/html;”的形式指定非默认的其他网页目录。

“use epoll;”项用于指定使用 epoll 事件模型，在 kernel 2.6 版本及以后的 Linux 系统中，可以更好的优化 nginx 服务性能。

7.3.2 添加 nginx 状态统计

利用Nginx自带的http_stub_status模块，可以实现简单的站点连接、请求数统计。要使用该功能，需要在编译nginx的时候，增加“--with-http_stub_status_module”配置项（参考第7.2.1小节）。

1. 配置 nginx 支持状态统计

修改 nginx.conf 文件，在 server { }配置部分中添加一项 location 设置。

```

[root@localhost ~]# vi /usr/local/nginx/conf/nginx.conf
    location ~ ^/Status {
        stub_status on;           //启用状态统计模块
        access_log off;          //关闭日志记录
    }
[root@localhost ~]# service nginx restart

```

2. 查看状态统计页面

重新应用配置以后，在浏览器中访问地址<http://www.benet.com/Status>，即可看到Nginx的状态统计信息。

```

[root@localhost ~]# elinks http://www.benet.com/Status
Active connections: 1
server accepts handled requests
 7 7 7
Reading: 0 Writing: 1 Waiting: 0

```

在上述状态信息中，

- “Active Connections”表示当前活动的连接数；
- “Server accepts handled requests”表示服务器处理的连接数，三个数字依次表示：处理的连接数、成功的 TCP 握手次数、处理的请求数。

7.3.3 通过 FastCGI 方式支持 PHP 页面

使用 RHEL5 系统自带的 php 环境时，php-cgi 工具由 rpm 包 php-cli-5.1.6-5.el5 提供，位于“/usr/bin/php-cgi”。若使用源码包编译安装 php 环境，需要在“./configure”时添加“--enable-cgi”选项，同时去掉“--with-apxs2”选项，否则可能无法编译出 php-cgi 程序。本例中使用前者。

如果站点需要处理的 php 页面请求数较多，可以结合 spawn-fcgi 工具同时开启多个 php-cgi 进程。

1. 获取 spawn-fcgi 工具

Lighttpd 是另一种轻量级 Web 服务软件，本例中只需要用到该软件包提供的 spawn-fcgi 工具，因此编译后无需安装，只要将执行程序 spawn-fcgi 复制出来即可。

```
[root@localhost ~]# wget http://www.lighttpd.net/download/lighttpd-1.4.20.tar.gz
[root@localhost ~]# tar zxvf lighttpd-1.4.20.tar.gz
[root@localhost ~]# cd lighttpd-1.4.20
[root@localhost lighttpd-1.4.20]# ./configure && make
[root@localhost lighttpd-1.4.20]# cp src/spawn-fcgi /usr/sbin/
```

2. 启动 php-cgi

以下命令将同时启动 16 个 php-cgi 子进程，在 127.0.0.1 的 9000 端口监听服务。

```
[root@localhost ~]# spawn-fcgi -a 127.0.0.1 -p 9000 -f /usr/bin/php-cgi -C 16
spawn-fcgi.c.206: child spawned successfully: PID: 31883
```

验证 spawn-fcgi 运行状态：

```
[root@localhost ~]# netstat -anp | grep :9000
tcp        0      0 127.0.0.1:9000      0.0.0.0:*           LISTEN
31883/php-cgi
[root@localhost ~]# ps aux | grep php-cgi
root      31883  0.0  1.1 16132 4312 ?        Ss   16:33   0:00 /usr/bin/php-cgi
root      31884  0.0  0.3 16132 1404 ?        S    16:33   0:00 /usr/bin/php-cgi
root      31885  0.0  0.3 16132 1404 ?        S    16:33   0:00 /usr/bin/php-cgi
.....
```

若需要在每次开机后都运行该 spawn-fcgi 命令，可以将其添加到/etc/init.d/nginx 或



7

Nginx 网站及负载均衡系统

/etc/rc.local 脚本中（建议采用前者）。

```
[root@localhost ~]# vi /etc/rc.local
/usr/sbin/spawn-fcgi -a 127.0.0.1 -p 9000 -f /usr/bin/php-cgi -C 16
```

3. 配置 nginx 支持 PHP 页面

修改 nginx.conf 文件，在 server { } 配置部分中添加一项 location 设置，将访问.php 页面的 web 请求转交给 php-cgi 程序处理。

```
[root@localhost ~]# vi /usr/local/nginx/conf/nginx.conf
    location ~ \.php$ {
        root html;
        fastcgi_pass 127.0.0.1:9000;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
[root@localhost ~]# service nginx restart
```

4. 测试 php 页面解析

创建一个php测试网页，并访问<http://www.benet.com/test.php>查看PHP解析结果。

```
[root@localhost ~]# vi /usr/local/nginx/html/test.php
<?php
echo "PHP is OK!";
phpinfo();
?>
[root@localhost ~]# elinks http://www.benet.com/test.php
PHP is OK!
```

7.3.4 基于域名的虚拟主机配置

1. 虚拟主机的域名解析

各虚拟主机需要有 DNS 服务器提供域名解析，在实验过程中也可以使用修改客户端主机 hosts 文件的方式解决。

```
[root@client ~]# vi /etc/hosts
192.168.0.11    bbs.benet.com mail.benet.com
```

2. 配置 nginx 支持虚拟主机

只需要修改 nginx.conf 文件，为每个虚拟主机设置一个对应的 server { } 配置段即可，各虚拟主机的 server 段参数可以独立配置。



```
[root@localhost ~]# vi /usr/local/nginx/conf/nginx.conf
server {          //第一个虚拟主机的配置段
    listen        80;
    server_name   bbs.benet.com;
    location / {
        root      /var/www/bbs;
        index     index.html;
    }
}
server {          //第二个虚拟主机的配置段
    listen        80;
    server_name   mail.benet.com;
    location / {
        root      /var/www/mail;
        index     index.html;
    }
}
[root@localhost ~]# service nginx restart
```

3. 验证虚拟主机页面

在 Nginx 服务器上创建虚拟主机对应的网页目录及测试页面。

```
[root@localhost ~]# mkdir -p /var/www/bbs/ /var/www/mail/
[root@localhost ~]# echo "This is bbs.benet.com" > /var/www/bbs/index.html
[root@localhost ~]# echo "This is mail.benet.com" > /var/www/mail/index.html
```

使用浏览器访问不同的虚拟主机，确认所对应的网页正确。

```
[root@localhost ~]# elinks http://bbs.benet.com
This is bbs.benet.com
[root@localhost ~]# elinks http://mail.benet.com
This is mail.benet.com
```

7.3.5 基于反向代理的负载均衡配置

Nginx 采用基于反向代理的简单负载均衡，后端可以是代理服务器，也可以是网站服务器。

配置负载均衡时，需要先设置好后端服务器使用的地址和端口，还可以指定各服务器的权重等参数（可选），然后在需要转发 HTTP 请求的位置（Location）启用反向代理。

1. 配置 nginx 的负载均衡

修改 nginx.conf 文件，在 http { }配置部分中添加 upstream 设置，并在 server{ }

7

Nginx 网站及负载均衡系统

部分设置代理转发。

```
[root@localhost ~]# vi /usr/local/nginx/conf/nginx.conf
    upstream wwwcluster {
        ip_hash;           //启用会话保持（不要和权重 weight 参数同时使用）
        server 192.168.0.21:80;    //后端 web 服务器的地址和端口
        server 192.168.0.22:80;
        server 192.168.0.23:80;
    }
    server {
        listen      80;
        server_name  bbs.benet.com;
        location / {
            proxy_pass http://wwwcluster;
            proxy_set_header x-real-IP $remote_addr;
        }
    }
[root@localhost ~]# killall -1 nginx
```

2. 验证负载均衡

开启各后端服务器的Web服务（为了测试效果，实验时可以创建不同内容的测试网页），在不同客户端的浏览器中访问<http://bbs.benet.com>，比较显示结果。

关闭掉其中一个或两个后端的 Web 服务，在客户端浏览器中再次刷新页面，查看页面变化情况。

网
网
之
网
+

8

基于 Linux 的 OpenVPN 网络

3
.
0

8.1 OpenVPN 概述

OpenVPN 是一个开源的加密隧道构建工具, 基于 OpenSSL 的 SSL/TLS 协议, 可以在 Internet 中实现点对点的 SSL VPN 安全连接。使用 OpenVPN 的好处是安全、易用和稳定, 且认证方式灵活, 具备实现 SSL VPN 解决方案的完整特性。OpenVPN 可以应用于 Linux、Unix、Mac OS 以及 Windows 等各种操作系统平台。

OpenVPN 提供两种类型的虚拟网络接口: TUN 和 TAP, 分别用于建立 IP 隧道、以太网桥接。在 Linux 中使用这两种虚拟设备, 需要对应的内核模块支持。RHEL5 系统默认已编译好 tun 模块, 直接使用即可。

OpenVPN 的官方网站是 <http://openvpn.net>, 目前发布的最新测试版本为 OpenVPN-2.1_rc13, 稳定版为 OpenVPN-2.0.9。

本章案例中, 需要下载使用的相关软件包参考如下:

最新稳定版源码包: <http://openvpn.net/release/openvpn-2.0.9.tar.gz>

Windows 安装包:

http://openvpn.se/files/install_packages/openvpn-2.0.9-gui-1.0.3-install.exe

LZO 工具包: <http://www.oberhumer.com/opensource/lzo/download/lzo-2.03.tar.gz>

8.2 OpenVPN 的安装及运行控制

8.2.1 在 RHEL5 系统中的安装

需要先安装 lzo 软件包, 用于压缩隧道通讯数据以加快传输速度。lzo 和 openvpn 源码包都可以按照默认的配置进行编译安装。OpenVPN 需要使用的 openssl 等软件均使用 RHEL5 系统自带的 rpm 包, 安装过程不再赘述。

```
[root@gw1 ~]# tar zxvf lzo-2.03.tar.gz
[root@gw1 ~]# cd lzo-2.03
[root@gw1 lzo-2.03]# ./configure && make && make install
[root@gw1 lzo-2.03]# cd ../
[root@gw1 ~]# tar zxvf openvpn-2.0.9.tar.gz
[root@gw1 ~]# cd openvpn-2.0.9
[root@gw1 openvpn-2.0.9]# ./configure && make && make install
```

8.2.2 在 Windows XP 系统中的安装

双击下载的 openvpn-2.0.9-gui-1.0.3-install.exe 文件, 按照向导提示安装即可。若只作为 OpenVPN 客户端使用, 安装时可以不勾选定制组件列表中的“OpenVPN Service” (如图 8.1 所示)。

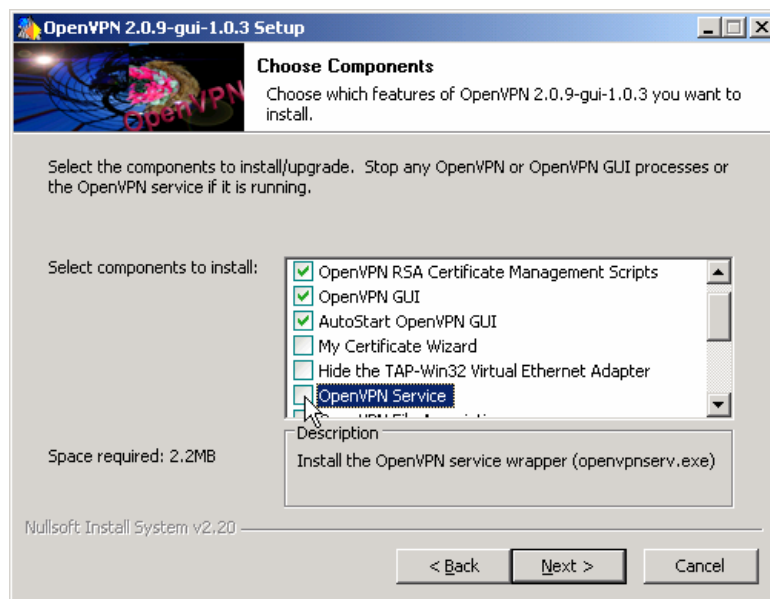


图8.1 安装 OpenVPN-GUI 客户端工具

8.2.3 OpenVPN 服务的运行控制

在 RHEL5 系统中，可以直接复制 `openvpn` 源码目录中的脚本样例文件，用于控制 `openvpn` 服务的开启、重启或关闭。

```
[root@gw1 ~]# cp -p openvpn-2.0.9/sample-scripts/openvpn.init /etc/init.d/openvpn
[root@gw1 ~]# chkconfig --add openvpn
[root@gw1 ~]# chkconfig --level 2345 openvpn on
[root@gw1 etc]# service openvpn status
openvpn: service not started
```

启动 `openvpn` 进程需要建立对应的配置文件，具体配置过程请参考第 8.3 节的应用案例。

8.3 OpenVPN 网络架构应用实例

8.3.1 案例需求分析

本案例主要基于 RHEL5 和 Windows XP 系统环境，跨越不安全的 Internet 网络，为异地的两个局域网及远程网管工作站建立安全的 SSL VPN 连接（如图 8.2 所示）。

其中，北京总部和上海分公司的网关服务器均使用 RHEL5 系统，需要分别配置 OpenVPN，用于连接两个异地的局域网 LAN1、LAN2。此外，位于 Internet 中的网管工作站使用 Windows XP 系统，需要随时通过 VPN 安全隧道访问总部的局域网 LAN1 和上海分公司的局域网 LAN2。

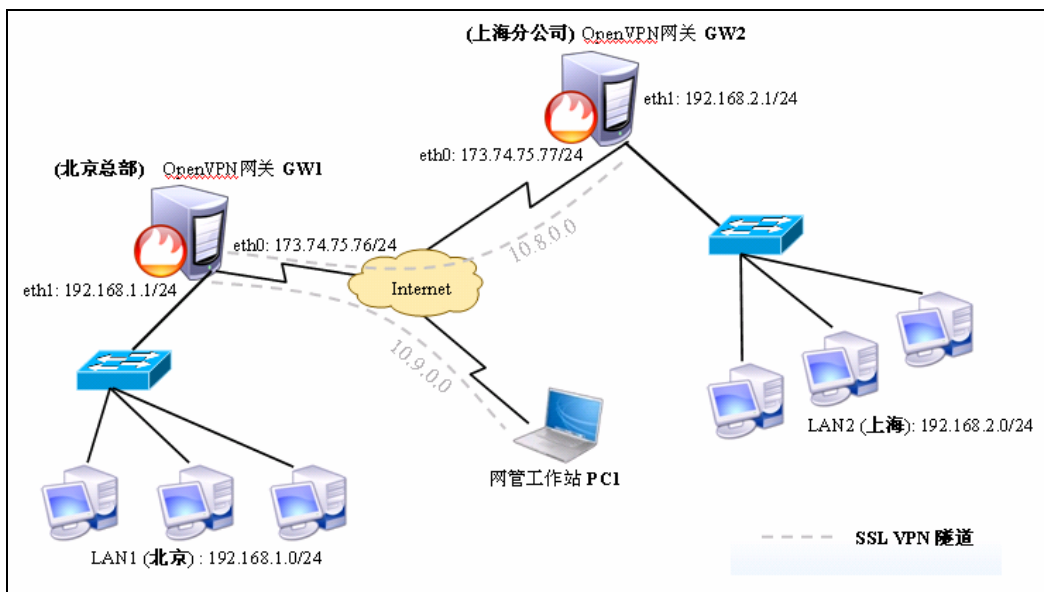


图8.2 OpenVPN 远程虚拟专用网络架构

基于上述需求，可以将北京总部的网关服务器 GW1 配置为 VPN Server 模式，上海的网关服务器 GW2 和 Internet 网管工作站 PC1 均使用 VPN Client 模式。分别建立两条点对点（Point-to-Point）的 SSL VPN 安全隧道——“GW1 <----> GW2”、“GW1 <----> PC1”即可。

由于 Internet 网络的细节不是本案例的重点，因此 GW1、GW2 的公网 IP 地址分别使用 173.74.75.76 和 173.74.75.77 来模拟。其他网络接口地址设置如下：

- GW1、GW2 的内网接口 IP 地址分别为 192.168.1.1、192.168.2.1。
- GW1 <----> GW2 隧道：分别使用虚拟 IP 地址 10.8.0.1/30、10.8.0.2/30。
- GW1 <----> PC1 隧道：分别使用虚拟 IP 地址 10.9.0.1/30、10.9.0.2/30。

另外，两地局域网的客户机需要正确设置好 IP 地址、默认网关等参数：

- LAN1 的主机使用 192.168.1.0/24 网段，默认网关设为 192.168.1.1。
- LAN2 的主机使用 192.168.2.0/24 网段，默认网关设为 192.168.2.1。

8.3.2 配置 GW1 <----> GW2 隧道连接

本小节主要阐述如何创建第 1 条 SSL VPN 隧道，用于连接 GW1、GW2 两台服务器，以便实现北京、上海两地局域网（LAN1、LAN2）的安全互联。

主要实现过程如下。

第一步、配置主服务器（GW1）——北京

1. 配置 Internet 连接及 SNAT、路由转发



1) 配置 IP 地址

eth0 接口(173.74.75.76/24)用于连接 Internet, eth1 接口 (192.168.1.1/24) 用于连接局域网 (配置过程略)。

2) 开启路由及 SNAT 转换

```
[root@gw1 ~]# vi /opt/gw1_nat.sh
sysctl -w net.ipv4.ip_forward=1
/sbin/iptables -t nat -I POSTROUTING -o eth0 -j SNAT --to-source 173.74.75.76
[root@gw1 ~]# chmod a+x /opt/gw1_nat.sh
[root@gw1 ~]# echo "/opt/gw1_nat.sh" >> /etc/rc.local
[root@gw1 ~]# /opt/gw1_nat.sh
```

2. 安装 OpenVPN 服务

参考第8.2.1、8.2.3小节。

3. 创建证书和密钥文件

证书和密钥文件主要用于点对点客户端的认证, 以便增强安全性。为了降低密钥创建过程的复杂性, 可以充分利用 OpenVPN 源码包提供的 **easy-rsa** 目录, 该目录中包含一系列简单易用的脚本工具 (参考“[openvpn-2.0.9/easy-rsa/README](#)”文件)。

3) 配置变量环境

修改 **easy-rsa/vars** 文件, 根据实际情况适当修改预定义变量, 或保持默认。在后续创建相关文件的过程中, 将会直接读取这些变量的内容。其中“**KEY_DIR**”变量的值决定了新创建的密钥等文件的存放位置。

```
[root@gw1 openvpn-2.0.9]# cd easy-rsa/
[root@gw1 easy-rsa]# vi vars
export D=`pwd`
export KEY_CONFIG=$D/openssl.cnf
export KEY_DIR=$D/keys
echo NOTE: when you run ./clean-all, I will be doing a rm -rf on $KEY_DIR
export KEY_SIZE=1024
export KEY_COUNTRY=CN           //粗体部分根据具体应用情况进行修改
export KEY_PROVINCE=BeiJing
export KEY_CITY=BeiJing
export KEY_ORG="BENET.Inc"
export KEY_EMAIL="vpnadm@benet.com"
[root@gw1 easy-rsa]# source vars           //执行 vars 文件中的代码
NOTE: when you run ./clean-all, I will be doing a rm -rf on /root/openvpn-2.0.9/easy-rsa/keys
[root@gw1 easy-rsa]# ./clean-all         //预先清除$KEY_DIR 目录
```

BEZEL

后续创建的密钥文件需要依据该 CA 文件。

执行“./build-dh”脚本即可建立 dh 文件。

执行“./build-key-server”脚本可以建立 VPN 服务端密钥文件，根据提示设置好 Common Name (gw1.benet.com)，最后依次按“y”键签署 (Sign) 及提交 (Commit)。

```
[root@gw1 easy-rsa]# ./build-key-server gw1
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to 'gw1.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [CN]:
State or Province Name (full name) [BeiJing]:
Locality Name (eg, city) [BeiJing]:
Organization Name (eg, company) [BENET.Inc]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:gw1.benet.com
Email Address [vpnadm@benet.com]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /root/openvpn-2.0.9/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName             :PRINTABLE:'CN'
stateOrProvinceName     :PRINTABLE:'BeiJing'
localityName            :PRINTABLE:'BeiJing'
organizationName        :PRINTABLE:'BENET.Inc'
commonName              :PRINTABLE:'gw1.benet.com'
emailAddress            :IA5STRING:'vpnadm@benet.com'
Certificate is to be certified until Nov  2 23:10:13 2018 GMT (3650 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```



8

基于 Linux 的 OpenVPN 网络

7) 创建 GW2 对端服务器密钥

执行“./build-key”脚本可以建立 VPN 客户端密钥文件,同样根据提示设置好 Common Name (gw2.benet.com), 最后依次按“y”签署 (Sign) 及提交 (Commit)。

```
[root@gw1 easy-rsa]# ./build-key gw2
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to 'gw2.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [CN]:
State or Province Name (full name) [BeiJing]:
Locality Name (eg, city) [BeiJing]:
Organization Name (eg, company) [BENET.Inc]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:gw2.benet.com
Email Address [vpnadm@benet.com]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /root/openvpn-2.0.9/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName             :PRINTABLE:'CN'
stateOrProvinceName     :PRINTABLE:'BeiJing'
localityName            :PRINTABLE:'BeiJing'
organizationName        :PRINTABLE:'BENET.Inc'
commonName              :PRINTABLE:'gw2.benet.com'
emailAddress            :IA5STRING:'vpnadm@benet.com'
Certificate is to be certified until Nov  2 23:23:40 2018 GMT (3650 days)
Sign the certificate? [y/n]:y
```



```
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```



使用“./build-key”脚本创建密钥时，不同的客户端对应的“Common Name”不能相同。

8) 生成 tls-auth 密钥

tls-auth 密钥可以为点对点的 VPN 连接提供了进一步的安全验证，如果选择使用这一方式，服务器端和客户端都必须拥有该密钥文件。

openvpn 命令跟上“--genkey --secret”选项可以用于建立 ta 密钥文件。

```
[root@gw1 easy-rsa]# openvpn --genkey --secret keys/ta.key
```

9) 最后将上述文件所在的 keys/文件夹转移至/etc/openvpn/目录

```
[root@gw1 easy-rsa]# mkdir -p /etc/openvpn/
[root@gw1 easy-rsa]# mv keys/ /etc/openvpn/
```

4. 创建主服务器配置文件

在服务器配置文件中指定使用 Server 模式，监听默认的 UDP 1194 端口。虚拟接口采用 tun0 设备。可以参考 openvpn 源码目录中的配置范例（openvpn-2.0.9/sample-config-files/server.conf）。

```
[root@gw1 ~]# vi /etc/openvpn/gw1_tun0.conf
local 173.74.75.76           //指定监听服务的 IP 地址
port 1194                   //为第 1 条隧道开启默认的 1194 端口
proto udp
dev tun                     //使用 SSL Tune 的 VPN 隧道模式
ca keys/ca.crt
cert keys/gw1.crt
key keys/gw1.key
dh keys/dh1024.pem
server 10.8.0.0 255.255.255.0 //使用服务器模式，并指定 VPN 虚拟网络地址
ifconfig-pool-persist ipp.txt
push "route 192.168.1.0 255.255.255.0" //为 GW2 添加到 LAN1 网段的路由
push "route 10.9.0.0 255.255.255.0"   //为 GW2 添加到 PC1 的路由
```

8

基于 Linux 的 OpenVPN 网络

```

push "dhcp-options DNS 210.22.84.3"      //为客户端设置 DNS 服务器地址
route 192.168.2.0 255.255.255.0          //为 GW1 添加到 LAN2 网段的路由
client-config-dir ccd                    //允许读取 ccd/目录下的客户端配置文件
keepalive 10 120
tls-auth keys/ta.key 0                  //指定 tls-auth 密钥
cipher BF-CBC                          //加密算法与客户端要保持一致
comp-lzo
max-clients 100                        //允许的最大并发 VPN 连接数
user nobody
group nobody
persist-key
persist-tun
status openvpn-status.log
log-append openvpn.log
verb 3
mute 20

```

5. 建立用于 GW2 的 ccd 配置文件

```

[root@gw1 ~]# mkdir -p /etc/openvpn/ccd
[root@gw1 ~]# cd /etc/openvpn/ccd/
[root@gw1 ccd]# vi gw2.benet.com        //为对端服务器 GW2 创建独立的配置文件
iroute 192.168.2.0 255.255.255.0        //声明 GW2 后端的 LAN2 子网络
ifconfig-push 10.8.0.2 10.8.0.1         //指定 GW2 的本地地址 (tun0)、对端地址 (P-t-P)

```

6. 启动 OpenVPN 服务

使用安装时建立的服务脚本启动 openvpn 服务。

```

[root@gw1 ~]# service openvpn start
[root@gw1 ~]# netstat -anp | grep openvpn
udp        0      0 173.74.75.76:1194    0.0.0.0:*           3528/openvpn

```

第二步、配置对端服务器 (GW2) —— 上海

1. 配置 Internet 连接及 SNAT、路由转发

1) 配置 IP 地址

eth0 接口(173.74.75.77/24)用于连接 Internet, eth1 接口 (192.168.2.1/24) 用于连接局域网 (配置过程略)。

2) 开启路由及 SNAT 转换



```
[root@gw2 ~]# vi /opt/gw2_nat.sh
sysctl -w net.ipv4.ip_forward=1
/sbin/iptables -t nat -I POSTROUTING -o eth0 -j SNAT --to-source 173.74.75.77
[root@gw2 ~]# chmod a+x /opt/gw2_nat.sh
[root@gw2 ~]# echo "/opt/gw2_nat.sh" >> /etc/rc.local
[root@gw2 ~]# /opt/gw2_nat.sh
```

2. 安装 OpenVPN 服务

参考第8.2.1、8.2.3小节。

3. 下载证书和密钥文件

下载在 GW1 主服务器端创建的 ca.crt、ta.key、gw2.key、gw2.crt 文件，使用 FTP、SCP、Email 等任何方式均可。

—— 在服务器 GW1 上：

```
[root@gw1 ~]# cd /etc/openvpn/keys/
[root@gw1 keys]# tar jcvf /var/ftp/send_to_gw2.tar.bz2 ca.crt ta.key gw2.key gw2.crt
[root@gw1 keys]# service vsftpd start
```

—— 在服务器 GW2 上：

```
[root@gw2 ~]# wget ftp://173.74.75.76/send_to_gw2.tar.bz2
```

将下载后的文件释放保存到 openvpn 配置目录。

```
[root@gw2 ~]# mkdir -p /etc/openvpn/keys/
[root@gw2 ~]# tar jxvf send_to_gw2.tar.bz2 -C /etc/openvpn/keys/
```

4. 创建客户端配置文件

在客户端配置文件中指定使用 Client 模式，以及远程 VPN 服务器的地址和端口。可以参考 openvpn 源码目录中的配置范例（openvpn-2.0.9/sample-config-files/client.conf）。

```
[root@gw2 ~]# vi /etc/openvpn/gw2.conf
client          //使用客户端模式
dev tun
proto udp
remote 173.74.75.76 1194      //点对点主服务器的地址、端口
resolv-retry infinite
nobind
user nobody
group nobody
```

8

基于 Linux 的 OpenVPN 网络

```

persist-key
persist-tun
ca keys/ca.crt
cert keys/gw2.crt
key keys/gw2.key
ns-cert-type server
tls-auth keys/ta.key 1
cipher BF-CBC           //需和 GW1 服务器端保持一致
comp-lzo
verb 3
mute 20

```

5. 启动 OpenVPN 服务

使用安装时建立的服务脚本启动 `openvpn` 服务。

```

[root@gw2 ~]# service openvpn start
[root@gw2 ~]# netstat -anp | grep openvpn
udp        0      0 0.0.0.0:32790      0.0.0.0:*           2886/openvpn
unix  2      []                DGRAM               27729  2886/openvpn

```

第三步、验证 SSL VPN 连接

1. GW1 服务器的 VPN 接口状态

5) GW1 主服务器 `tun0` 接口的 IP 地址为 `10.8.0.1`，点对点对端地址为 `10.8.0.2`。

```

[root@gw1 ~]# ifconfig tun0
tun0                Link        encap:UNSPEC        HWaddr
00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
    inet addr:10.8.0.1  P-t-P:10.8.0.2  Mask:255.255.255.255
    UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
    RX packets:0 errors:0 dropped:0 overruns:0 frame:0
    TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:100
    RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

```

6) 路由表中应有通过 GW2 访问 LAN2 网段的路由记录。

```

[root@gw1 ~]# route -n
Kernel IP routing table

```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
10.8.0.2	0.0.0.0	255.255.255.255	UH	0	0	0	tun0
192.168.2.0	10.8.0.2	255.255.255.0	UG	0	0	0	tun0

10.8.0.0	10.8.0.2	255.255.255.0	UG	0	0	0 tun0
192.168.1.0	0.0.0.0	255.255.255.0	U	0	0	0 eth1
173.74.75.0	0.0.0.0	255.255.255.0	U	0	0	0 eth0

2. GW2 服务器的 VPN 接口状态

1) GW2 服务器 tun0 接口的 IP 地址为 10.8.0.2，点对点对端地址为 10.8.0.1

```
[root@gw2 ~]# ifconfig tun0
tun0                                Link        encap:UNSPEC          HWaddr
00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
    inet addr:10.8.8.2  P-t-P:10.8.8.1  Mask:255.255.255.255
    UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
    RX packets:0 errors:0 dropped:0 overruns:0 frame:0
    TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:100
    RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

2) 路由表中应有通过 GW1 访问 LAN1 网段的路由记录

```
[root@gw2 ~]# route -n
Kernel IP routing table
Destination    Gateway        Genmask         Flags Metric Ref    Use Iface
10.8.0.1       0.0.0.0       255.255.255.255 UH          0      0      0 tun0
192.168.2.0    0.0.0.0       255.255.255.0  U           0      0      0 eth1
10.8.0.0       10.8.0.1     255.255.255.0  UG          0      0      0 tun0
192.168.1.0    10.8.0.1     255.255.255.0  UG          0      0      0 tun0
173.74.75.0    0.0.0.0       255.255.255.0  U           0      0      0 eth0
10.9.0.0       10.8.0.1     255.255.255.0  UG          0      0      0 tun0
```

3. 测试 SSL VPN 隧道连接

1) 测试 GW1 与 GW2 的连接

```
[root@gw1 ~]# ping 10.8.0.2 -c 2
PING 10.8.0.2 (10.8.0.2) 56(84) bytes of data.
64 bytes from 10.8.0.2: icmp_seq=1 ttl=64 time=6.90 ms
64 bytes from 10.8.0.2: icmp_seq=2 ttl=64 time=4.06 ms
--- 10.8.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 4.063/5.484/6.906/1.423 ms
```

2) 测试 LAN1、LAN2 中主机之间的互联（如图 8.3 所示）。

8

基于 Linux 的 OpenVPN 网络

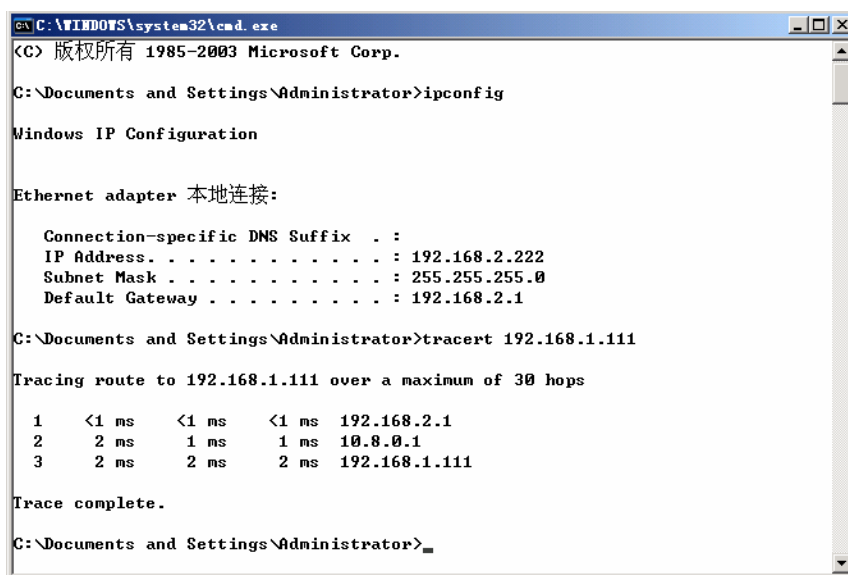


图8.3 测试 LAN1、LAN2 中主机之间的互联

8.3.3 配置“GW1 <----> PC1”隧道连接

本小节主要描述如何创建第 2 条 SSL VPN 隧道，用于在 GW1、PC1 之间建立点到点的安全连接，以便 Windows XP 客户端能够从 Internet 中安全的接入总部的 VPN 网络，进一步访问两地的局域网络。

推荐主要实现步骤如下。

第一步、配置主服务器（GW1）——北京

1. 创建 PC1 用户端密钥

在 GW1 服务器上为 PC1 网管机建立单独的密钥文件，注意 Common Name 不要和其他密钥重复。生成的密钥文件必需放置到/etc/openvpn/keys/目录中（可以直接调整 KEY_DIR 变量）。

```

[root@gw1 ~]# cd openvpn-2.0.9/easy-rsa/
[root@gw1 easy-rsa]# source vars
[root@gw1 easy-rsa]# export KEY_DIR=/etc/openvpn/keys/
[root@gw1 easy-rsa]# ./build-key client-pc1
Generating a 1024 bit RSA private key
.....++++++
.....++++++
writing new private key to 'client-pc1.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.

```

```

What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [CN]:
State or Province Name (full name) [BeiJing]:
Locality Name (eg, city) [BeiJing]:
Organization Name (eg, company) [BENET.Inc]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:client-pc1.benet.com
Email Address [vpnadm@benet.com]:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /root/openvpn-2.0.9/easy-rsa/openssl.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName             :PRINTABLE:'CN'
stateOrProvinceName     :PRINTABLE:'BeiJing'
localityName            :PRINTABLE:'BeiJing'
organizationName        :PRINTABLE:'BENET.Inc'
commonName              :PRINTABLE:'client-pc1.benet.com'
emailAddress            :IA5STRING:'vpnadm@benet.com'
Certificate is to be certified until Nov  2 23:32:05 2018 GMT (3650 days)
Sign the certificate? [y/n]:y

1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated

```

2. 创建第 2 个服务配置文件

该配置文件用于建立“GW1 <----> PC1”隧道连接，将使用 tun1 设备、监听 UDP 1195 端口。

```

[root@gw1 ~]# vi /etc/openvpn/gw1_tun1.conf
local 173.74.75.76
port 1195          //为第 2 条隧道开启新的 1195 端口监听服务
proto udp

```

8

基于 Linux 的 OpenVPN 网络

```

dev tun
ca keys/ca.crt
cert keys/gw1.crt
key keys/gw1.key
dh keys/dh1024.pem
server 10.9.0.0 255.255.255.0      //设置第 2 条隧道的虚拟网络地址
ifconfig-pool-persist ipp.txt
push "route 192.168.1.0 255.255.255.0"    //为 PC1 添加到 LAN1 的路由
push "route 192.168.2.0 255.255.255.0"    //为 PC1 添加到 LAN2 的路由
push "route 10.8.0.0 255.255.255.0"      //为 PC1 添加到 GW2 的路由
push "dhcp-options DNS 202.106.0.20"
client-to-client
client-config-dir ccd
keepalive 10 120
tls-auth keys/ta.key 0
cipher BF-CBC
comp-lzo
max-clients 10
user nobody
group nobody
persist-key
persist-tun
status openvpn-status-2.log      //启用新的日志文件
log-append openvpn-2.log
verb 3
mute 20

```

3. 建立用于 PC1 的 ccd 配置文件

```

[root@gw1 ~]# vi /etc/openvpn/ccd/client-pc1.benet.com
ifconfig-push 10.9.0.2 10.9.0.1

```

4. 重新启动 OpenVPN 服务

重启 openvpn 服务后应在 1194、1195 端口分别监听服务。

```

[root@gw1 ~]# service openvpn restart
[root@gw1 ~]# netstat -anp | grep openvpn
udp        0      0 173.74.75.76:1194    0.0.0.0:*        4133/openvpn
udp        0      0 173.74.75.76:1195    0.0.0.0:*        4147/openvpn

```

第二步、配置 Windows XP 客户机 (PC1) ——上海



1. 配置 IP 地址及 Internet 连接（略）

2. 安装 OpenVPN-GUI 客户端工具

参考第8.2.2小节。

3. 下载证书和密钥文件

下载在 GW1 主服务器端创建的 ca.crt、ta.key、client-pc1.key、client-pc1.crt 文件。

将下载的密钥等相关文件复制到 OpenVPN 配置文件目录下（默认为 C:\Program Files\OpenVPN\config\）。注意保留好备份。

4. 创建 PC1 用户端配置文件

使用 Windows XP 系统自带的记事本工具创建客户端配置文件 client-pc1.ovpn，文件内容可以参考范例（C:\Program Files\OpenVPN\sample-config\client.ovpn）。

```
Microsoft Windows [版本 5.2.3790]
(C) 版权所有 1985-2003 Microsoft Corp.
C:\Documents and Settings\Administrator> notepad client-pc1.ovpn
client
dev tun
proto udp
remote 173.74.75.76 1195
resolv-retry infinite
nobind
persist-key
persist-tun

ca ca.crt
cert client-pc1.crt
key client-pc1.key
ns-cert-type server

tls-auth ta.key 1
cipher BF-CBC

comp-lzo
verb 3
mute 20
```



8

基于 Linux 的 OpenVPN 网络

将建好的配置文件保存到 **C:\Program Files\OpenVPN\config** 文件夹下 (如图 8.4 所示)。

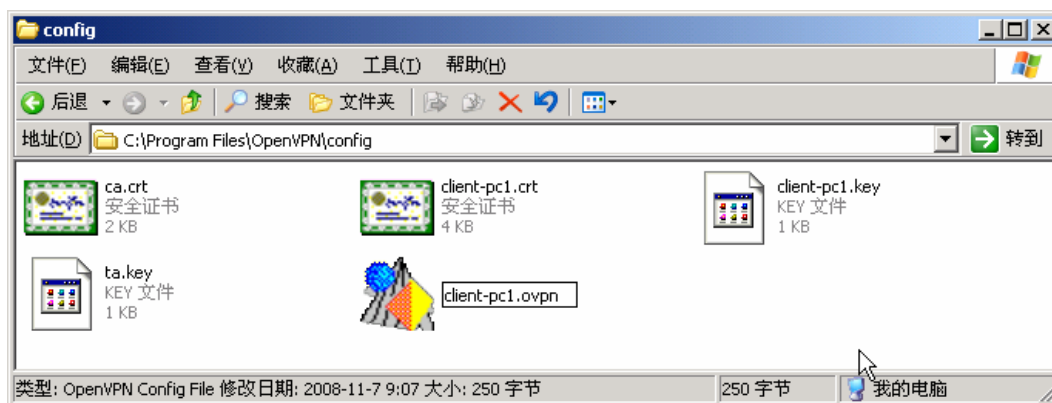


图8.4 client-pc1.ovpn 文件存放位置

5. 建立 OpenVPN 连接

方法一：在 Windows XP 任务栏右侧（系统托盘区）的 OpenVPN-GUI 图标上点击右键，选择“Connect”，即可使用创建好的配置文件建立 VPN 连接。

方法二：在创建好的 client-pc1.ovpn 文件上点击右键，选择“Start OpenVPN on this config file”即可。——用此方法可以查看建立连接的过程信息，适合用于调试/排错。

第三步、验证 SSL VPN 连接

1. GW1 服务器的 VPN 接口状态

GW1 主服务器将会新增接口 tun1，IP 地址为 10.9.0.1，点对点对端地址为 10.9.0.2。

```
[root@gw1 ~]# ifconfig tun1
tun1                                Link        encap:UNSPEC          HWaddr
00-00-00-00-00-00-00-00-00-00-00-00
    inet addr:10.9.0.1  P-t-P:10.9.0.2  Mask:255.255.255.255
    UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
    RX packets:34 errors:0 dropped:0 overruns:0 frame:0
    TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:100
    RX bytes:3034 (2.9 KiB)  TX bytes:2998 (2.9 KiB)
```

2. Windows XP 客户机 PC1 的 VPN 接口状态

使用 OpenVPN-GUI 连接成功以后，系统将会增加一个“TAP-Win32 Adapter V8”的本地连接，用户可以通过“开始 ---> 设置 ---> 网络连接”进行查看。

使用“**ipconfig /all**”命令查看PC1 新增的VPN接口，IP地址应为 10.9.0.2（如图 8.5所示），由对端服务器 10.9.0.1 自动配置。

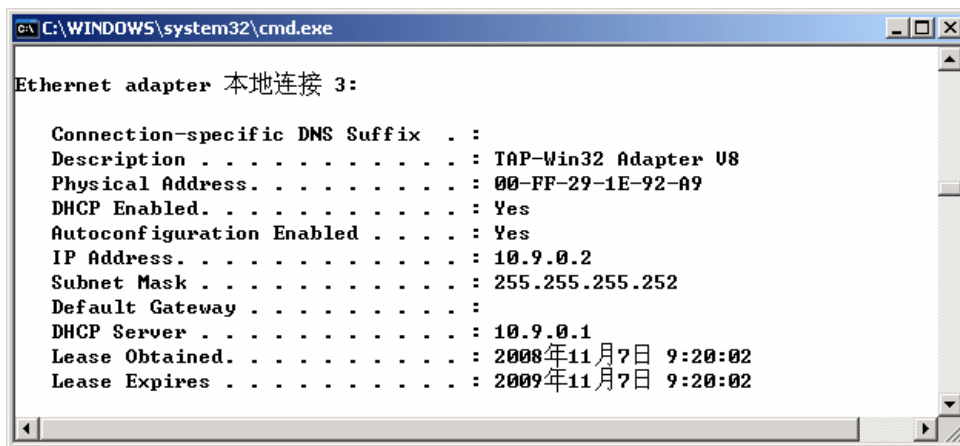


图8.5 客户机 PC1 的 VPN 接口地址

3. 测试 PC1 与 GW1、GW2 之间的连接

1) 连接GW1（如图 8.6所示）

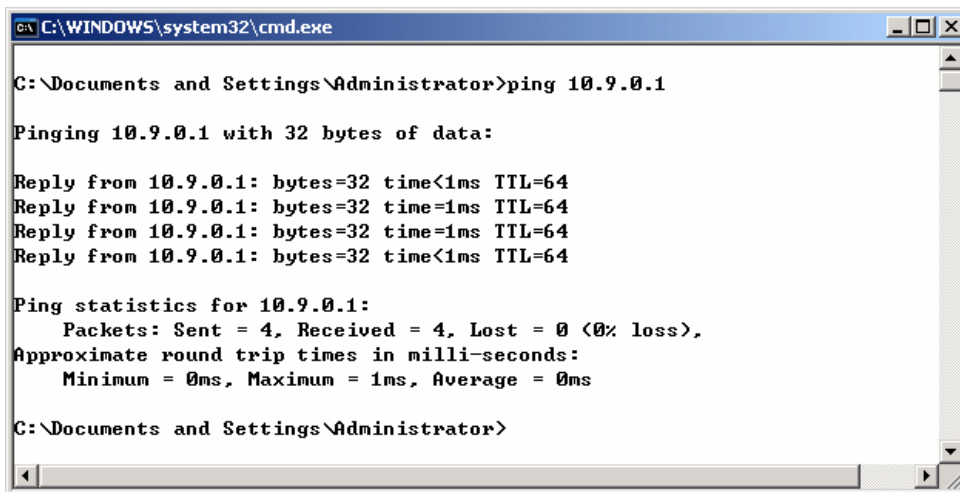
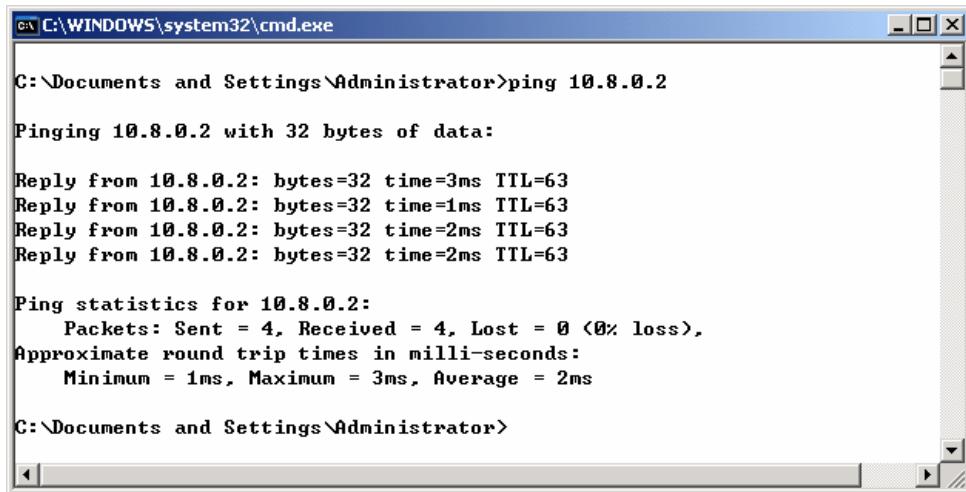


图8.6 PC1 到 GW1 的 ping 测试

2) 连接GW2，使用ping测试（如图 8.7所示）

8

基于 Linux 的 OpenVPN 网络



```
C:\WINDOWS\system32\cmd.exe

C:\Documents and Settings\Administrator>ping 10.8.0.2

Pinging 10.8.0.2 with 32 bytes of data:

Reply from 10.8.0.2: bytes=32 time=3ms TTL=63
Reply from 10.8.0.2: bytes=32 time=1ms TTL=63
Reply from 10.8.0.2: bytes=32 time=2ms TTL=63
Reply from 10.8.0.2: bytes=32 time=2ms TTL=63

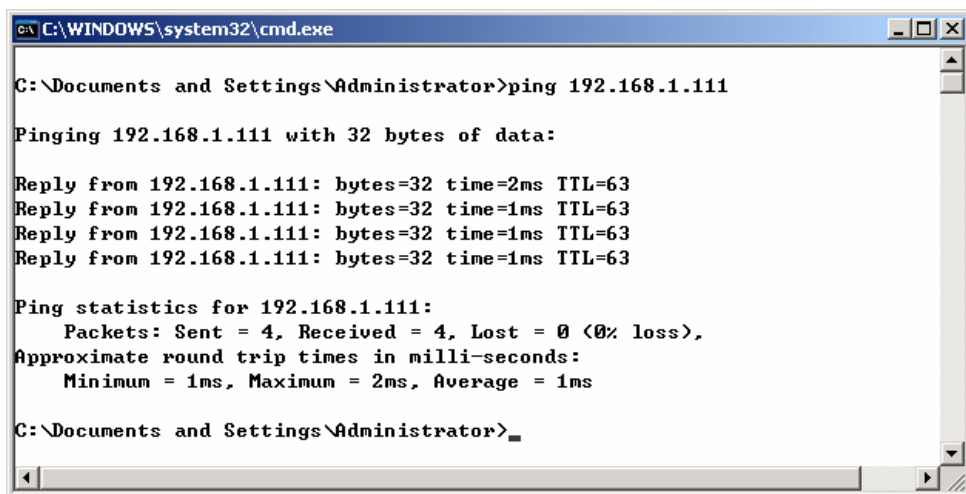
Ping statistics for 10.8.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 3ms, Average = 2ms

C:\Documents and Settings\Administrator>
```

图8.7 PC1 到 GW2 的 ping 测试

4. 测试 PC1 与 LAN1 中主机之间的连接

连接 LAN1 中的主机，使用 ping 测试（如图 8.8 所示）、tracert 路由追踪（如图 8.9 所示）。



```
C:\WINDOWS\system32\cmd.exe

C:\Documents and Settings\Administrator>ping 192.168.1.111

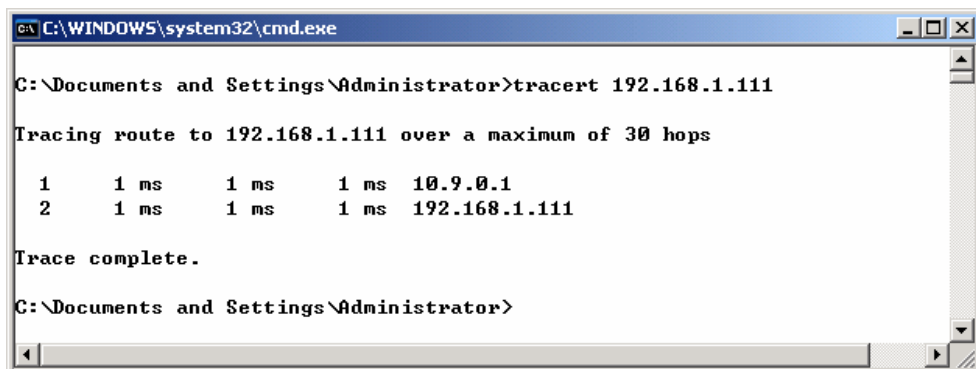
Pinging 192.168.1.111 with 32 bytes of data:

Reply from 192.168.1.111: bytes=32 time=2ms TTL=63
Reply from 192.168.1.111: bytes=32 time=1ms TTL=63
Reply from 192.168.1.111: bytes=32 time=1ms TTL=63
Reply from 192.168.1.111: bytes=32 time=1ms TTL=63

Ping statistics for 192.168.1.111:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 2ms, Average = 1ms

C:\Documents and Settings\Administrator>
```

图8.8 PC1 到 LAN1 中主机的 ping 测试



```
C:\WINDOWS\system32\cmd.exe

C:\Documents and Settings\Administrator>tracert 192.168.1.111

Tracing route to 192.168.1.111 over a maximum of 30 hops

    1    1 ms    1 ms    1 ms  10.9.0.1
    2    1 ms    1 ms    1 ms  192.168.1.111

Trace complete.

C:\Documents and Settings\Administrator>
```



图8.9 PC1 到 LAN1 中主机的 tracert 路由追踪

5. 测试 PC1 与 LAN2 中主机之间的连接

连接LAN2 中的主机，使用ping测试（如图 8.10所示）、tracert路由追踪（如图 8.11所示）。

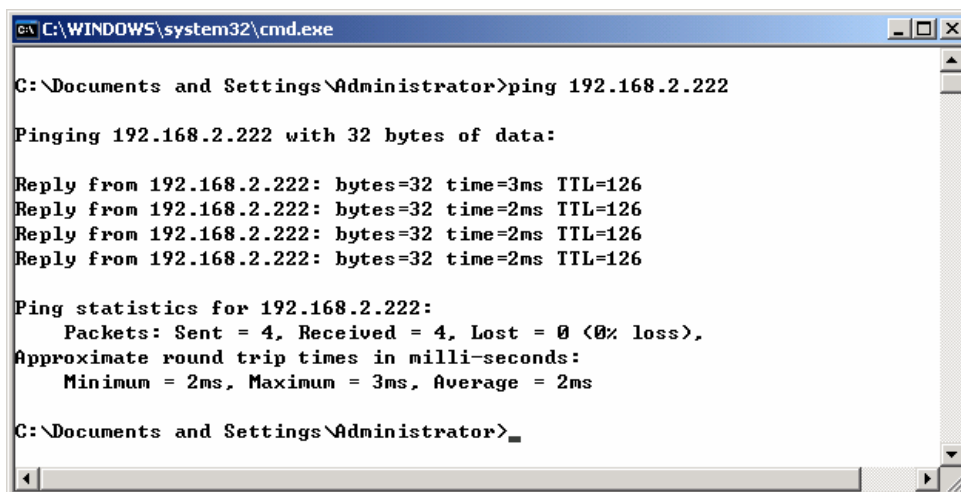


图8.10 PC1 到 LAN2 中主机的 ping 测试

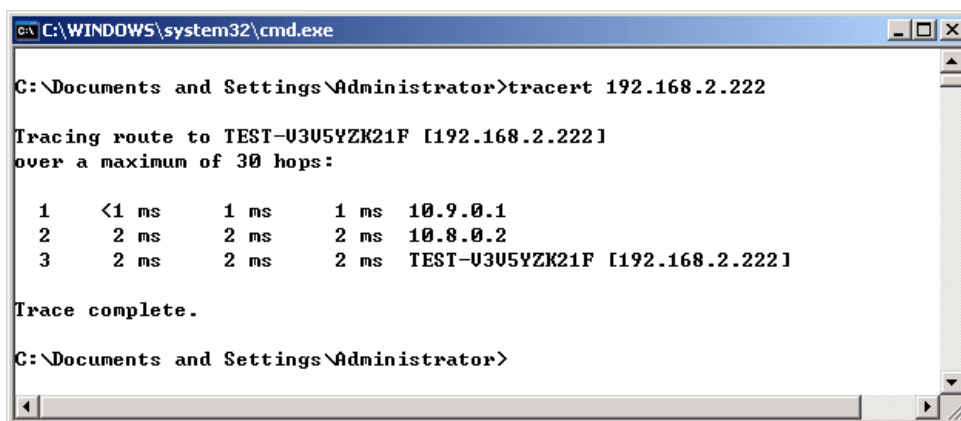


图8.11 PC1 到 LAN2 中主机的 tracert 路由追踪

网
网
之
网
斗

3
.
0

9

构建软 RAID 磁盘阵列

9.1 RAID 概述

RAID 即廉价冗余磁盘阵列 (Redundant Array of Inexpensive Disk) 的简称, 通过该技术可以将多个磁盘组成一个阵列整体, 而应用时可以作为单个磁盘使用。RAID 磁盘阵列根据其使用的技术不同, 可用于提高数据读写效率、提高数据冗余 (备份), 当阵列中的一个磁盘发生故障时, 可以通过校验数据从其他磁盘中进行恢复, 大大增强了应用系统数据的读写性能及可靠性。

较常见的 RAID 技术包括如下几个级别:

- **RAID 0:** 最基本的一种阵列方式, 通过简单的将多个磁盘 (最少 2 块) 组成到一起, 作为一个大磁盘使用。存取数据时, 通过将数据分段同时写入到不同的磁盘中, 大大提高了效率。但是这种方式没有数据冗余, 其中任何一个磁盘坏了以后, 都可能导致数据丢失。
- **RAID 1:** 即磁盘镜像技术, 需要最少 2 块磁盘 (磁盘利用率: $1/n$)。这种方式将数据同时写入到阵列中的多块磁盘中, 不同磁盘中的数据互为镜像。因此, 其中任何一个磁盘坏了以后, 数据不会丢失。
- **RAID 5:** 通过引入数据校验技术来保证数据的安全, 需要最少 3 块磁盘 (磁盘利用率: $n-1$)。这种方式并不使用固定的某块磁盘存放校验数据, 而是分段存储在各个磁盘中。因此, 其中任何一个磁盘坏了以后, 也可以根据其他磁盘中的校验数据进行恢复。

由于 RAID5 阵列技术既通过数据冗余增强了可靠性, 又通过多块磁盘同时写入数据提高了效率, 一直以来受到广泛的应用。

未使用硬件磁盘卡方式实现的 RAID 技术, 通常称为软 RAID 技术。本文将在 RHEL5 系统中, 使用不同磁盘中的多个分区, 配置实现 RAID 5 磁盘阵列。

9.2 构建使用软 RAID 磁盘阵列

在 RHEL5 系统中, 配置软 RAID 阵列可以通过安装 mdadm 软件包实现。该软件包一般为系统默认安装, 若检查没有安装的话, 从 RHEL5 系统光盘中查找安装即可。

```
[root@localhost ~]# mount /dev/cdrom /media/cdrom/
mount: block device /dev/cdrom is write-protected, mounting read-only
[root@localhost ~]# rpm -ivh /media/cdrom/Server/mdadm-2.5.4-3.el5.i386.rpm
Preparing...      ##### [100%]
   1:mdadm        ##### [100%]
[root@localhost ~]# rpm -qi mdadm | grep "Summary"
Summary          : mdadm 控制 Linux md 设备 (软件 RAID 阵列)
```

下面将以 RAID5 磁盘阵列为例, 讲解软磁盘阵列的配置使用方法。

9.2.1 准备用于 RAID 阵列的分区

9

构建软 RAID 磁盘阵列

用于组成 RAID 阵列的各个分区应该位于不同的磁盘设备中，否则实用价值不大。各分区的容量最好也相同，必要时可以将整个硬盘划分为一个分区。

为 Linux 服务器添加 4 块 SCSI 硬盘，并使用 `fdisk` 工具各划分出一块 2GB 的分区，依次为：`/dev/sdb1`、`/dev/sdc1`、`/dev/sdd1`、`/dev/sde1`。分区前请注意确保没有别的程序正在使用对应的磁盘。下一小节中将以这 4 个分区为例（RAID 5 需要至少 3 块磁盘或分区）讲解 RAID5 磁盘阵列的创建方法。

对于上述分区还应该将其类型 ID 更改为“fd”，对应为“Linux raid autodetect”，表示支持用于 RAID 磁盘阵列。

```
[root@localhost ~]# fdisk /dev/sdb
.....
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-522, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-522, default 522): +2048M
Command (m for help): t
Selected partition 1
Hex code (type L to list codes): fd
Changed system type of partition 1 to fd (Linux raid autodetect)
Command (m for help): p
.....
```

Device	Boot	Start	End	Blocks	Id	System
/dev/sdb1		1	250	2008093+	fd	Linux raid autodetect

```

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@localhost ~]#
```

创建好三个分区以后，执行“`partprobe`”重新探测分区表（或重启系统），验证分区类型和容量等信息。

```
[root@localhost ~]# partprobe
[root@localhost ~]# fdisk -l /dev/sd[b-e] | grep "^/dev/sd"
```



/dev/sdb1	1	250	2008093+	fd	Linux raid autodetect
/dev/sdc1	1	250	2008093+	fd	Linux raid autodetect
/dev/sdd1	1	250	2008093+	fd	Linux raid autodetect
/dev/sde1	1	250	2008093+	fd	Linux raid autodetect

9.2.2 创建 RAID 设备

使用 `mdadm` 工具可以组合多个 RAID 分区作为一个磁盘阵列，阵列设备文件名习惯上使用“/dev/md0”、“/dev/md1”等。

```
[root@localhost ~]# mdadm -Cv /dev/md0 -a yes -n4 -l5 /dev/sd[b-e]1
mdadm: layout defaults to left-symmetric
mdadm: chunk size defaults to 64K
mdadm: /dev/sdb1 appears to be part of a raid array:
        level=raid5 devices=4 ctime=Sat Jul 25 08:44:50 2009
mdadm: /dev/sdc1 appears to be part of a raid array:
        level=raid5 devices=4 ctime=Sat Jul 25 08:44:50 2009
mdadm: /dev/sdd1 appears to be part of a raid array:
        level=raid5 devices=4 ctime=Sat Jul 25 08:44:50 2009
mdadm: /dev/sde1 appears to be part of a raid array:
        level=raid5 devices=4 ctime=Sat Jul 25 08:44:50 2009
mdadm: size set to 2008000K
Continue creating array? y
mdadm: array /dev/md0 started.
[root@localhost ~]#
```

在上述命令操作中，“/dev/md0”为新建的 RAID 阵列设备文件名，“/dev/sd[bcd]1”表示此阵列将使用/dev/sdb1、/dev/sdc1、/dev/sdd1 这三个分区。其他各部分选项、参数的含义如下：

- **-C**，等同于 `--create`：创建一个新的阵列设备
- **-v**，等同于 `--verbose`：执行过程中输出细节信息
- **-a**，等同于 `--auto=`：指定参数为 `yes` 时，表示若对应的设备文件不存在则自动创建
- **-n**，等同于 `--raid-devices=`：用于组成阵列的分区设备个数，“-n3”表示为 3 个
- **-l**，等同于 `--level=`：使用的 RAID 级别，“-l5”表示为 RAID 5

关于 `mdadm` 命令更多选项的使用，请参考“`man mdadm`”帮助信息。

创建好 md0 阵列设备后，将自动被激活，执行“`cat /proc/mdstat`”可以观察阵列设备的运行状态。

```
[root@localhost ~]# ls -l /dev/md0
brw----- 1 root root 9, 0 07-25 09:03 /dev/md0
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sde1[3] sdd1[2] sdc1[1] sdb1[0]
```

9

构建软 RAID 磁盘阵列

```
6024000 blocks level 5, 64k chunk, algorithm 2 [4/4] [UUUU]
```

其中，“[4/4]”部分中的第一个“4”表示成员设备个数，后边的“4”表示当前的活动设备个数，“UUUU”对应为成员设备的状态。例如，若出现“[4/3][UUU_]”的信息时，则表示第4个成员设备（/dev/sde1）出现故障了。

9.2.3 在 RAID 设备中建立文件系统

创建好磁盘阵列设备文件“/dev/md0”以后，就可以在该设备中建立文件系统了。在 RHEL5 系统中，可以使用 **mkfs** 命令格式化该设备，将其作为 **ext3** 文件系统来使用。

```
[root@localhost ~]# mkfs -t ext3 /dev/md0
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
753664 inodes, 1506000 blocks
75300 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1543503872
46 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
This filesystem will be automatically checked every 33 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

9.2.4 挂载并使用文件系统

创建挂载点目录“/mdata”，并将上一小节中建立好的文件系统挂载到该目录，即可正常使用。

```
[root@localhost ~]# mkdir /mdata
[root@localhost ~]# mount /dev/md0 /mdata
[root@localhost ~]# df -T | grep "md0"           //验证挂载的“/mdata”文件系统
/dev/md0      ext3      5929360    142976    5485184    3% /mdata
```

除去一个成员设备作为校验盘以后，磁盘阵列的有效存储空间为 3 个成员设备的容量之和（大约为 $2\text{GB} \times 3 = 6\text{GB}$ ）。

如果希望在每次开机后自动挂载该阵列设备，可以在“/etc/fstab”文件中添加相应设置。



```
[root@localhost ~]# vi /etc/fstab
/dev/md0      /mdata      ext3    defaults    0    0
```

9.3 RAID 阵列的管理及设备恢复

9.3.1 基本管理操作

1. 扫描或查看磁盘阵列信息

使用 `mdadm` 命令时，“-D”选项相当于“--detail”，表示显示扫描结果的详细内容；“-s”选项相当于“--scan”，用于扫描阵列设备。

未指定阵列设备文件时，可以显示出当前系统中的阵列配置信息、RAID 设备列表。

```
[root@localhost ~]# mdadm -vDs
ARRAY                  /dev/md0                  level=raid5              num-devices=4
UUID=35bcffa1:cdc5ba41:0c5b5702:e32a3259
devices=/dev/sdb1,/dev/sdc1,/dev/sdd1,/dev/sde1
```

当指定阵列设备作为参数时，可以输出指定阵列设备的详细参数，包括活动设备个数、失效设备个数、更新时间、列表成员设备位置等。

```
[root@localhost ~]# mdadm -vDs /dev/md0
/dev/md0:
    Version : 00.90.03
  Creation Time : Sat Jul 25 11:23:07 2009
    Raid Level : raid5
    Array Size : 6024000 (5.74 GiB 6.17 GB)
    Device Size : 2008000 (1961.27 MiB 2056.19 MB)
    Raid Devices : 4
    Total Devices : 4
Preferred Minor : 0
    Persistence : Superblock is persistent
    Update Time : Sat Jul 25 11:26:01 2009
    State : clean
    Active Devices : 4
    Working Devices : 4
    Failed Devices : 0
    Spare Devices : 0
    Layout : left-symmetric
    Chunk Size : 64K
    UUID : 35bcffa1:cdc5ba41:0c5b5702:e32a3259
    Events : 0.6
    Number  Major  Minor  RaidDevice State
```

9

构建软 RAID 磁盘阵列

0	8	17	0	active sync	/dev/sdb1
1	8	33	1	active sync	/dev/sdc1
2	8	49	2	active sync	/dev/sdd1
3	8	18	3	active sync	/dev/sde1

2. 创建配置文件 mdadm.conf

mdadm 的配置文件为 “/etc/mdadm.conf”，该文件只是用来方便用户管理和使用，缺少此文件并不会影响磁盘阵列的功能。在配置文件中可以保存多个磁盘阵列的配置信息。配置文件中的基本信息可以通过前面讲过的 “mdadm -vDs” 命令获得。

```
[root@localhost ~]# vi /etc/mdadm.conf
DEVICE /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
ARRAY                                /dev/md0                level=raid5                num-devices=4
UUID=35bcffa1:cdc5ba41:0c5b5702:e32a3259
devices=/dev/sdb1,/dev/sdc1,/dev/sdd1,/dev/sde1
CREATE owner=root group=root mode=0640
```

在上述文件中，“ARRAY”、“UUID” “devices” 部分是位于同一行中的内容，最后一行中的“CREATE”用于设置自动创建阵列设备文件的属主、属组及默认权限。关于 mdadm.conf 配置文件中更多配置项的使用，可以参考 “man mdadm.conf” 帮助信息。

3. 启动/停止 RAID 阵列

在确保没有相关程序读写磁盘阵列设备的情况下，可以停止阵列设备。只需使用 mdadm 命令结合 “-S” 选项（等同于 “--stop” 选项）即可。执行该操作将会禁用对应的阵列设备，释放相关资源。

```
[root@localhost ~]# mdadm -S /dev/md0
mdadm: stopped /dev/md0
```

结合 “-A” 选项（等同于 “--assemble” 选项）可以重新组合对应的磁盘阵列设备。

```
[root@localhost ~]# mdadm -A /dev/md0
mdadm: /dev/md0 has been started with 4 drives.
[root@localhost ~]# mount /dev/md0 /mdata/
```

9.3.2 设备恢复操作

1. 模拟阵列设备故障

对于运行中的磁盘阵列，可以结合 mdadm 命令的 “-f” 选项（等同于 “” 选项）用于模拟成员设备故障，例如可将阵列中的 “/dev/sdb1” 标记为故障设备。

```
[root@localhost ~]# mdadm /dev/md0 -f /dev/sde1
mdadm: set /dev/sde1 faulty in /dev/md0
```



当阵列中的某个成员设备出现故障时，阵列会将其标记为失活状态。此时通过“cat /proc/mdstat”可以观察到丢失出现故障的设备（/dev/sde1）。

```
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sde1[3](F) sdd1[2] sdc1[1] sdb1[0]
      6024000 blocks level 5, 64k chunk, algorithm 2 [4/3] [UUU_]
```

2. 更换故障设备，并恢复数据

对于出现故障的设备，可以结合“-r”选项将其移除，然后换上正常的设备，结合“-a”选项重新添加到阵列中即可。

```
[root@localhost ~]# mdadm /dev/md0 -r /dev/sde1
mdadm: hot removed /dev/sde1                //移除故障设备
[root@localhost ~]# mdadm /dev/md0 -a /dev/sde1
mdadm: re-added /dev/sde1                    //重新加入正常设备
```

RAID5 磁盘阵列能够在较短时间内进行重构和数据恢复，当需要恢复的数据较多时，可以在此期间观察到阵列状态的恢复进度。

```
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sde1[3] sdd1[2] sdc1[1] sdb1[0]
      6024000 blocks level 5, 64k chunk, algorithm 2 [4/3] [UUU_]
      [===>.....]      recovery = 16.3% (328192/2008000)  finish=2.6min
      speed=10586K/sec
      unused devices: <none>
```

待数据恢复完成以后，再次查看阵列状态即显示为正常了。

```
[root@localhost ~]# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sde1[3] sdd1[2] sdc1[1] sdb1[0]
      6024000 blocks level 5, 64k chunk, algorithm 2 [4/4] [UUUUU]
      unused devices: <none>
```

