中文版本 1.1

LinuxAid.com.cn

Get Acquainted with Linux Security and Optimization System

熟悉 Linux 系统的安全和优化

原著： **Gerhard Mourani**

关于这本书

这本书的英文版“Get Acquainted with Linux Security and Optimization System”（简称 LinuxSOS）是 Linux 文档计划（Linux Document Project）中比较新的一本指南，介绍的是 Linux 系统的安全和优化。它既不象 HOWTO 那么难懂，又很实用，有了这本书大概就能自己做一个像样的服务器了。英文原文作者 Gerhard Mourani 是一个很有经验的工程师，这本书涉及了 Linux 系统的很多方面，有软件也有硬件，从内核优化编译到防火墙、Proxy、LDAP、SSL、数据库还有 FTP、WEB、Sendmail。英文原文可以到这下载：http://www.globec.com.au/LDP/LDP/gawlso/linuxsos-1.1.pdf。

请注意，这本书不是 Linux 的入门书籍，读这本书需要有 Linux 或者 Unix 的背景知识。如果你已经安装过 Linux 而且能够使用一些简单的 Unix 命令，那么这本书会对你有很大的帮助。

为什么要翻译这本书

这些年 Linux 在中国流行得很快，而且国内已经有很多公司在对 Linux 进行汉化，发行中国人自己的 Linux。汉化 Linux 操作系统对 Linux 在中国的普及起到了巨大的推动作用。但是，冷静下来想一想，在这些风风火火的汉化运动的后面，我们对 Linux 的知识还是那么的平乏。缺乏使用 Linux 的知识，这才是 Linux 在中国普及的真正障碍。Linux 在中国需要“启蒙”运动。

现在在书店里有不少关于 Linux 的书籍，但是真正的精品并不多，简单重复的太多了。而且，多数都是介绍如何安装以及如何使用象 KDE 和 GNOME 这些图形界面。大多数人都把 Linux 当作象玩具一样的操作系统，玩过之后就把它卸载掉了。这并不是我所说的“启蒙”运动。Linux 真正发展的方向是作为服务器操作系统。我们要把 Linux 真正的用起来。在 Internet 上，可以把 Linux 做成 WEB 服务器、FTP 服务器……；在企业内部网中，可以把 Linux 做成文件服务器、邮件服务器……。Linux 有各种各样的应用，可以安装成各式各样的服务器。在不远的将来 Linux 完全有潜力代替 NT Server 成为最流行的服务器操作系统。现在最需要的是安装和配置 Linux 服务器的知识。

LinuxSOS 这本书很全面地介绍了安装、配置和优化的 Linux 服务器以及保证 Linux 系统安全的知识，是不可多得的好书，所以我们把它翻译成中文介绍给大家。

关于译者

这本书是由 LinuxAid.com.cn 站点的注册工程师翻译的。我们是在网络上协同工作完成这本书的翻译。参加翻译的先后有 8 位工程师，在“附录 C”中列出了这些工程师翻译的相应章节。

版权和许可协议

本书的维护和更新

这本书由 brimmer 负责维护，email 地址：brimmer@linuxaid.com.cn。如果有什么意见和建议，请发 email 给他。

如果你喜欢这本书而且想知道这本书更新的消息，请到 forum.linuxaid.com.cn 注册一下。一旦这本书有所更新，或者我们有别的好书介绍给你，就会用 email 通知你。

同时也欢迎你加入我们，因为我们现在做的是“栽树”的事，熟话说“前人栽树，后人乘凉”，希望以后千千万万人都能从中受益。

Linux 最大的优点并不在于它是免费的，而且在于自由、开放和协作精神。“众人拾柴火焰高”，通过和大家一起努力，我们会尽量做得更好。

目录

介绍

第一部分：与安装相关的参考资料

第一章：介绍 Linux

第二章：安装 Linux 服务器

第二部分：与安全和优化相关的参考资料

第三章：系统安全概要

第四章：系统优化概要

第三部分：与内核相关的参考资料

第五章：配置和编译内核

第四部分：与网络相关的参考资料

第六章：TCP/IP 网络管理

第七章：网络防火墙

第五部分：与软件相关的参考资料

第八章：编译器的功用

第九章：系统安全软件

第六部分：与备份相关的参考资料

第十一章：备份和恢复的过程

第七部分：附录

介绍

当开始写这本书的时候，我问自己的第一个问题是：怎样安装一个 Linux 服务器，使得没有经过安全验证，任何人都无法访问它。接着，我又想有没有什么方法可以优化系统的性能。于是我开始在 Internet 上检索并且阅读了不少书籍以获得系统安全和性能优化的知识。经过多年的学习和研究，我最终找到了这些问题的答案。这些答案是从不同的文献、书籍、文章和 Internet 站点零零碎碎地收集而得到的。于是，我就把我的这些研究和积累写成一篇文章，希望它对我有所帮助。这些年来，这篇文章越来越长，越来越像一本书而不是零碎的笔记了。我决定把这本书在 Internet 上公开，让其他人也能从中受益。通过与别人分享这本书，我可以回报 Linux 社团，是他们创造了 Linux，一个可靠的、健壮的、强大的、快速的和自由的操作系统，使包括我在内的千千万万人都从中受益。我也收到了大量关于这本书的反馈和评价，帮助我改进和完善了这本书。同时，我也发现许多人都希望这本书能够出版，使得更多的人从中受益，以促进 Linux 的发展。

概述

这本书从实用的角度出发，主要教你如何一步一步地完成一项任务，用例子来说明，而不是关于 Linux 的详细介绍。Linux 的详细介绍在 Linux 文档计划的 HOWTO 中已经覆盖得很全面了。这本书主要是面向技术人员的。它介绍了如何安装一个 RedHat Linux 服务器，使之成为尽可能安全和优化的高性能服务器。因为我们强调优化和更多的配置选择，所以对于那些关键性的服务器程序，如 Apache、Bind、Samba、Squid、Openssl 等等，我们使用以源代码形式（tar.gz）发布的这些软件。使用以源代码形式发布的软件的好处是：升级得更快；可以为特殊的机器进行优化和配置（如果使用 RPM 包是得不到这些好处的）。我在写这本书的时候用了很多免费的资源，只有把它重新回馈给 Linux 社团才是公平的。这本书主要是面向 Intel x86 硬件平台的，你可能找不到一些 PPC、ARM、SPARPC、APX 等其它硬件平台上的特性。安装这本书介绍的服务器要求重新编译内核，其它的程序可以根据需要自行安装。

学习这本书的一些要求

要有一个光驱和 RedHat Linux 的光盘。这本书介绍的服务器软件的安装在 RedHat Linux 6.1 上测试过。还要熟悉自己的硬件环境。在下面的章节中，这本书将一步一步地指导你完成整个安装过程。

本书作者 Gerhard Mourani 的 PGP 密匙

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.0.0 (GNU/Linux)
Comment: For info see http://www.gnupg.org
mQGiBDgU8UcRBADiuIKn95nz0qsvjU1GzBxv0AOxJHVTNhFBl6lt+3DzDA0G7UTu
```

介绍

```
hOhT0aGwVGts3bzjXVbhS44CTfAvvuVYQq7Ic/BHkwIhFvSu/Xv/fGbD3IQy+Gn5
UYzhZegCGwB0KQhGkIwQPus2ONOS5oT3ChZ8L7JlCPBnlOcVBT+hZ3BXUwCg4y4L
Mz5aEe0MPCZ3xkcNE7AE71EEAL4Jf2uVhIRgOfwIpdB1rKVKrDDFxZLx+yZeOZmq
gdwa4m7wV+Rk+c4I1+qBxxkmcUBhTHigx+9kpBDE2J0aEGQezDN+RoqlmdyVFO98
T/znf4ZLIf0upu5aP4kAItJJuFB1AaJyDLesB5xGjfyWz+RhbKOmeqr2zHniOsa8
HcZ/BACKZFBjNElqFUf0niWf822W6IbNf7ASh8pwTgR9PmXcq2qtBBq8uCIpEYcD
wzk+ccl2jt8qt5RB7DXz/r/uG+3YHU+ID4iz6Qm6zl84gYQLDXST2YXZ5BPURo7H
O4nEIJfeHEuUCstE5ROKnblG2U+t5QmxSGbETnK9I/OZrzFwILRDR2VyaGFyZCBN
b3VyYW5pIChPcGVuIE5ldHdvcmsgQXJjaGl0ZWN0dXJlKSA8Z21vdXJhbmlAdmlk
ZW90cm9uLmNhPohVBBMRAgAVBQI4FPFHAwsKAwMVAwIDFgIBAheAAAoJEDPaC2+7
tLqbGcYAnjHIPAsZrRC5qU5OrqdPvvEmICUWAKCdeyWwJ785A58U8Vh1bpxzCVVb
PbkCDQQ4FPI0EAgAy7qa88bVYWIEyAWxJPZRxl8G2GcxgshSu4+5udeP+4PlVAm8
3DUynzlcax4/ikx8Q8MoVR7s6lCLJXCycLENE8xFCJJQ26IxzBjdftGdmvKteVkZ
Kld9PZMzjUsxKzmhZbGEWug6xaav68EIewTw/S0TFtPhXyUKFrYPV6aID7YGatzB
P4hQJfh4Wt3NdP9QznASBze6bPZxR07iEZaUO0AMHeeBKwL6rptEcGuxHPMYc00R
s+SdGTOAa9E/REIiiEike9mXTKKWJYG2e7leDP3SBruM/c7n+DC9ptFAapg1GD9f
Re7LLFqj6EQzZqybPB61B9rB/8ShIrApcNYF4wADBQgAvROi9N0/J5kYvBVb60no
xBUBYtZp4cJO9X1uVdVahCb9XZpbvxhKujaUoWpPCIb0pm8K+J8x0o9HFl9f/JTs
25N/eJwksr63+j8OdCHqxv4z+qQYgc/qvU42ekHlSfMc7vsiAIE1e1liuTBdN9KR
7oSBoaht+dKi16ffxXmMDvQs1YSBR114XXDSzI+xXRuaIISpi75NE6suLLlrksnL
+i/NcLRbCTEv4p1UJGYT4OVnX6quC3CC+U4Drpjf2ohawsXqS7jKUYduZRr9Hbar
/sE0pQ/P0uf+VAspQJgpvBqiDxbIRCDSx8VgDoRL7iayxPDXtFmbPOrUEPdS7qYX
pIhGBBgRAgAGBQI4FPI0AAoJEDPaC2+7tLqbdzQAniStW48nFU6CWkvQTy8fr0lu
ZXmXAKC5bgSLgg1gZAvx61Z20yzM+hwNFQ==
=95nO
-----END PGP PUBLIC KEY BLOCK-----
```

第一部分：与安装相关的参考资料

第一章

介绍Linux

什么是Linux

Linux 是一个操作系统。最初它是由在芬兰赫尔辛基大学念书的 Linus Torvalds 作为一种业余爱好开发出来的。Linus 对用于教学的一种小型的 Unix 操作系统 —— Minix 很感兴趣，决定开发出超过 Minix 的操作系统。在 1991 年，他开始 Linux 的开发，那时候的 Linux 是 0.02 版。到了 1994 年，Linux 内核的 1.0 版发布了。现在稳定的 Linux 内核版本是 2.2 版，开发工作还在继续进行。

Linux 的开发是遵循 GPL 的，它的源代码任何人都可以免费获得。但是这并不是说 Linux 以及 Linux 的发行版本是免费的。只要源代码保持公开，开发人员是可以要求获得适当的报酬。Linux 可以用在很多方面，包括：网络、软件开发和桌面平台。Linux 常被认为是其它昂贵的操作系统的替代品。

因为 Linux 操作系统的稳定和健壮，以及可以很容易地得到，它在世界上越来越流行。成千上万的程序员在根据自己的需要利用 Linux 的源代码。现在，有很多正在进行中的项目尝试着把 Linux 移植到不同的硬件平台上，或把 Linux 用于其它的用途。

为什么要用Linux

首先，它是免费的。尽管 Linus Torvalds 拥有 Linux 的商标，但是 Linux 的内核和相关的软件是遵循 GPL 的。这意味着你可以更改源代码，并靠卖程序获利，但是最初的作者拥有版权而且你必须公开修改过的源代码。

尽管 Linux 在基于 Intel 平台的计算机上最流行，但是相对其它操作系统，它可以运行在更多的 CPU 和硬件平台上。其原因是：除了拥有很多天才的开发人员之外，Linux 是带着源代码一起发行的，而且它的内核是可移植的。

现在软件和硬件行业的趋势是让用户去买更快的计算机，不断地扩充内存和增加硬盘空间。Linux 并不受这些“增肥”趋势的影响，它甚至可以在内存不是很多的 486 计算机上运行得很好。

Linux 很少死机，你可以停止那些有问题的进程，而让操作系统正常地运行下去。而且，Linux 用的是最先进的内存管理技术，不会让操作系统失去控制，也根本不要经常重启操作系统。

如果需要安装的是一个服务器操作系统，那么 Linux 是有很多优势的，特别是比起其它操作系统，如 Windows 2000，要便宜得多了。还有一个优点是：Linux 几乎不受病毒的攻击。因为遵循 GPL 和开放源代码，基本上你可以获得操作系统以及操作系统上所有软件的源代码。

让我们排除使用 Linux 的一些害怕和疑虑

Linux 不过是一个象玩具一样的操作系统

可能微软公司希望这是真的，但是事实并不是象他们希望的那样。Linux 正被越来越多的 Fortune 500 的企业、政府部门和消费者所采用。不信的话，去问一问 IBM、Compaq、Dell、Apple Computer、Burlington Coast Factory、Amtrak、Virginia Power、NASA 或者成百上千万的 Linux 用户吧。

Linux 没有技术支持

尽管有不少人认为 Linux 是没有技术支持的操作系统，但是每个 Linux 的发行版都提供 12,000 多页的文档。Linux 的商业发行版，例如：RedHat Linux、Caldera、SuSE 和 OpenLinux 为注册用户提供安装支持，小企业和公司可以得到一些商业技术支持公司的服务支持。Linux 作为一个开放源代码的操作系统，可以得到它的全部源代码。如果你遇到一些问题而且自己有解决问题的能力，就自己解决吧！不用为了得到技术支持等上很长的时间，许多严重的问题（如系统安全问题）可以在 Internet 上的 Linux 社团的帮助下几个小时内解决。

第二章

安装 Linux 服务器

了解硬件环境

了解硬件环境是成功安装 RedHat Linux 的关键。因此，必须花一些时间熟悉一下自己的硬件环境。准备好回答如下的问题：

1. 计算机上有几个硬盘？
2. 每个硬盘的大小？
3. 如果有两个以上的硬盘，哪一个是主盘？
4. 内存有多大？
5. 计算机上有 SCSI 卡吗？如果有，型号是什么？由哪家公司生产的？
6. 鼠标是什么类型的（串口，还是 PS/2）？
7. 鼠标有几个键？
8. 如果是串口鼠标，那么连在哪个串口上？
9. 显卡的产商和型号，有多少显存？
10. 显示器的产商和型号？
11. 如果想把计算机连接到网络上，请再注意下面几个问题：
    a) 计算机的 IP 地址？
    b) 网络掩码？
    c) 网关地址？
    d) 域名服务器的地址？
    e) 域名？
    f) 主机名？

g) 网卡型号和厂商？

创建启动盘 引导 Linux

如果你安装过很多次 Linux，结果都失败了，那么可能需要一个修正过的软盘镜像。在这种情况下，可以去 RedHat Linux 的勘误网站去下载特殊的软盘镜像。

因为这是很少见的情况，应该尽可能地先试一下标准的安装软盘，只有在实在不能完成安装的情况下，才有必要去勘误网站看看。RedHat 6.1 的光盘可以从光驱直接引导。如果你的计算机不支持从光盘直接启动，那么也可以在 MS-DOS 下创建启动盘，请用下面的命令（假定 CD-ROM 的盘符是 d:，里面放的是 RedHat Linux 6.1 的光盘）：

在 windows 下打开 MS-DOS 方式（开始 → 程序→ MS-DOS 方式）：

```
C:\> d:
D:\> cd \dosutils
D:\dosutils> rawrite
Enter disk image source file name: ..\images\boot.img
Enter target diskette drive: a:
Please insert a formatted diskette into drive A: and press --ENTER-- :
D:\dosutils>
```

当 rawrite.exe 要求输入软盘镜像的时候，输入完整的“boot.img”的路径名，把软盘插入 A 驱。当程序问你把镜像写入哪个软盘驱动器的时候，输入“a:”。

RedHat Linux 的勘误网站是：http://www.redhat.com/errata。

因为我们不从 CD-ROM 启动，而从软盘启动安装，所以把软盘插入 A:驱，然后重新启动计算机。当出现 boot:的时候，按下回车键，继续引导。

- 选择语言
- 选择键盘类型
- 选择鼠标类型

安装类型和方法

RedHat Linux 6.1 有以下几种安装类型：

- GNOME 工作站
- KDE 工作站
- 服务器

- 自定义

有些安装类型（GNOME 工作站、KDE 工作站和服务器），简化了安装过程，但是失去了很多灵活性。

因此，我们建议你选择“自定义”安装，这样你可以选择安装什么服务器软件以及如何划分你的硬盘。

我们的目标是安装尽可能少的软件，以保证系统的高效率。而且，安装的软件越少，安全漏洞也就越少。

选择“Custom”，继续安装。

创建分区

警告

我们强烈建议在分区之前，先备份你当前的系统。为了保证系统高效、稳定和安全，我们还建议象下面介绍的那样分区。我们这样分区是假定你要配置一个 Web 和 Proxy 服务器。

之所以在分区中有两个特殊的“/chroot”和“/cache”分区是因为：“/chroot”分区是为 DNS 服务器、Apache 服务器和其它需要改变根目录的（chrooted）服务器准备的;“/cache”分区是为 Squid Proxy 服务器准备的。如果你不打算安装 Squid Proxy 服务器，那么不必创建“/cache”分区，但是请注意：Squid+Apache 的结合会把服务器的性能和安全提高很多。

如果想让数量较多的用户访问你的服务器，必须把“/tmp”和“/home”放在不同的分区上，这几乎是强制的。这样做可以避免分区被一些用户文件填满。把“/var”和“/usr”放在不同的分区也是一个好主意。把“/var”和其它分区分开也可以避免分区被填满。

在我们的分区安排中，特别保留了 400MB 的磁盘空间给改变根目录的（chrooted）程序，比如：Apache、DNS 以及其它软件。这是必须的，因为 Apache 的 DocumentRoot 文件和其它可执行的程序都会被安装在这个分区上。注意：Apache chrooted 目录在“/chroot”分区上的大小是和“DocumentRoot”所有文件的大小成正比的。如果不准备安装和使用 Apache，可以把分区大小减小到 10M，这 10M 是 DNS 服务器所必需的。

最小的分区要求

下面是 Linux 能够正常运行的最小分区要求。这样的分区安排甚至可以适用于只有 512MB 硬盘的老式 486 计算机机。当然，这种分区安排只是一个建议。

```
/ 35MB
/boot 5MB
/chroot 10MB
/home 100MB
/tmp 30MB
/usr 232MB
/var 25MB
```

Disk Druid 分区工具

Disk Druid 是 RedHat Linux 带的一个分区工具。选择“Add”添加新的分区，“Edit”改变分区，“Delete”删除分区，“Reset”恢复原来的分区状态。在添加一个新分区的时候，会出现一个窗口，要求你添上有关这个分区的一些必要的信息。要求添的信息是：

- Mount Point：分区装载（mount）的目录
- Size：分区的大小，以兆为单位
- Partition Type：分区类型。Linux 的文件系统要使用 Linux native 类型的分区，Linux 交换分区请用 Linux Swap 类型。

如果你有一个 SCSI 硬盘，那么这个设备的名称将是“/dev/sda”；如果你有一个 IDE 硬盘，那么这个设备名称将是“/dev/hda”。如果你很在意系统的高性能和稳定性，我们建议你用 SCSI 硬盘。

Linux 分区的命名方式是字母和数字相结合的，这种命名方式很灵活也很直观。下面是一个小小的总结：

- **前两个字母**：前两个字母表示设备类型，“hd”代表 IDE 硬盘，“sd”表示 SCSI 硬盘
- **第三个字母**：这个字母是说明具体的设备。例如：“/dev/hda”表示第一个 IDE 硬盘；“/dev/hdb”表示第二个 IDE 硬盘。

请记住这些，这样给 Linux 分区的时候，就会觉得容易一些同时也不易混淆。

交换分区

交换分区是用于虚拟内存的。如果计算机的内存少于 16M，你必须创建交换分区。即使有更多的内存，我们还是建议你建立交换分区。交换分区最小必须等于计算机的内存，但是如果内存多于 16M，最小只要 16M 就行了。交换分区最大大约为 1GB，（Linux 2.2 内核现在可以支持 1GB 的交换文件，如果多于 1GB 就是浪费）。注意：可以创建多于一个的交换分区，尽管只有在安装大型服务器的时候才有这个必要。尽量把交换分区放在硬盘驱动器的起始位置，因为一个磁盘的起始位置在物理上是在最外的柱面上，所以磁头每转一圈可以覆盖更大的面积。

下面是一个例子，教你怎样建立分区（这样的分区安排是安装我们在书中介绍的服务器所需要的），下面的例子是用 Disk Druid 的命令：

```
Add
Mount Point: /boot □←our /boot directory
Size (Megs): 5
Partition Type: Linux Native
Ok

Add
Mount Point: /usr □←our /usr directory.
Size (Megs): 1000
Partition Type: Linux Native
Ok

Add
Mount Point: /home □←our /home directory.
Size (Megs): 500
Partition Type: Linux Native
Ok

Add
Mount Point: /chroot □←our /chroot directory.
Size (Megs): 400
Partition Type: Linux Native
Ok

Add
Mount Point: /cache □←our /cache directory.
Size (Megs): 400
Partition Type: Linux Native
Ok

Add
```

```
Mount Point: /var □←our /var directory.
Size (Megs): 200
Partition Type: Linux Native
Ok

Add
Mount Point: □←our /Swap partition (leave the Mount Point Blank).
Size (Megs): 150
Partition Type: Linux Swap
Ok

Add
Mount Point: /tmp □←our /tmp directory.
Size (Megs): 100
Partition Type: Linux Native
Ok

Add
Mount Point: / □←our / directory.
Size (Megs): 316
Partition Type: Linux Native
Ok
```

分区完毕之后，你可以在屏幕上看到类似的信息：

Mount Point	Device	Requested	Actual	Type
/boot	sda1	5M	5M	Linux Native
/usr	sda5	1000M	1000M	Linux Native
/home	sda6	500M	500M	Linux Native
/chroot	sda7	400M	400M	Linux Native
/cache	sda8	400M	400M	Linux Native
/var	sda9	200M	200M	Linux Native
<Swap>	sda10	150M	150M	Linux Swap
/tmp	sda11	100M	100M	Linux Native
/	sda12	316M	315M	Linux Native

Drive	Geom [C/H/S]	Total (M)	Free (M)	Used (M)	Used (%)
Sda	[3079/64/32]	3079M	1M	3078M	99%

现在，选择“Next”继续安装。分区创建完之后，安装程序会让你格式化分区。选择想要格式化的分区，选中“Check for bad blocks during format”选择框，按“Next”继续。这样就开始格式化分区，并且激活分区。Linux 现在就可以使用这个分区了。

下一步如果选择安装 LILO，你会看到 LILO 的配置。可以选择把 LILO 装在主引导扇区（MBR）或引导分区的第一个扇区。

在通常情况下，应该选择把 LILO 安装在主引导扇区。（如果你的计算机上装有 NT，或一些多重引导程序，如 System Command，你最好仔细看一遍 LILO-HOWTO，以免不必要的损失）。然后，开始配置网络和时钟。接着，要输入 root 口令和进行安全验证的配置。不要忘了选上：

- Enable MD5 passwords
- Enable MD5 passwords

没有必要选择 Enable NIS，因为我们不在这台服务器上安装 NIS 服务。

部件安装　选择软件包组

上面都完成了之后，该选择安装哪些软件包了。在默认情况下，Linux 是一个强大的操作系统，可以提供很多服务。但是，这些服务大多数都是没有必要的，而且会造成安全隐患。

安装 Linux 的正确方法是：安装一个稳定和安全的系统。首先，你要选择必须安装的部件，也就是软件包组。通过选上“Select individual package”这个单选框，在后面的安装过程中，你可以选中或不选单独的软件包。

因为我们安装的是 Linux 服务器，所以没有必要安装图形界面（XFree86）。在服务器上安装图形界面意味着：更低的处理能力，更少的 CPU 时间，更少的内存，更多的安全问题，以及等等。图形界面一般只在工作站上使用。选择安装下面的软件包组：

- Networked Workstation
- Network Management Workstation
- Utilities

选择好软件包组之后，就应该选择单个的软件包了。

注意：选上“Select individual package”单选框（非常重要），因为只有这样才会让你选择安装单个软件包。

单个软件包的选择

安装程序列出可以选择的软件包组，每个软件包组下面是单独的软件。

下面列出的软件，因为安全、优化或者其它原因，请不要安装，待一会儿会解释的。

```
Applications/Archiving: dump
Applications/File: git
Applications/Internet: finger, ftp, fwhois, ncftp, rsh, rsync, talk, telnet
Applications/Publishing: ghostscript, ghostscript-fonts, mpage,
rhs-printfilters
Applications/System: arpwatch, bind-utils, knfsd-clients, procinfo, rdate, rdist,
screen, ucd-snmp-utils
Documentation: indexhtml
System Environment/Base: chkfontpath, yp-tools
System Environment/Daemons: XFree86-xfs, lpr, pidentd, portmap, routed, rusers,
rwho, tftp, ucd-snmp, ypbind
System Environment/Libraries: XFree86-libs, libpng
User Interface/X: XFree86-75dpi-fonts, urw-fonts
```

在解释为什么不装这些软件包之前，可能有人会问：为什么不在服务器上装 finger、ftp、fwhois 和 telnet 呢？因为这些软件从本质上来说是不安全的。假定有一个黑客已经入侵了你的服务器，他可以用 finger、ftp、fwhois 和 telnet 查询或入侵网络中的其它计算机。如果没有装这些软件，他就不能使用这些软件或者就得试图在你的服务器上安装这些软件，如果真的这样做，就会被你很容易地跟踪到。可以用 Tripware 这样的软件进行跟踪。

Applications/Archiving:

dump 这个软件包包括 dump 和 restore 这两个程序。dump 用来检查文件系统中的文件以确定哪些需要备份，然后把这些文件拷贝到磁盘、磁带或其它介质上。**【没必要，我们用别的方法】**

Applications/File:

GIT（GNU 交互式工具）可以浏览文件系统，查看文本或二进制文件，查看或停止进程，还包括一些其它相关的工具和 shell 脚本。（很象 DOS 下的 NC）。**【没必要】**

Applications/Internet:

finger 可以让用户查看系统中其他用户的登录名（login name）、家目录（home directory）、真实姓名（name）以及登录系统的时间，等等。**【安全隐患】**

ftp 是标准的 UNIX 下的 FTP 客户软件（命令行）。FTP 是在 Internet 上使用很广的文件传输协议。**【安全隐患】**

fwhois 可以让系统中的用户查询 whois 数据库。**【安全隐患】**

ncftp 是增强型的 FTP 客户软件。ncftp 的增强包括：支持命令行编辑，命令历史记录，递归下载（包含子目录的下载），自动匿名登录，还有其它很多功能。**【安全隐患，没必要】**

rsh 包括一些程序，这些程序可以在远程计算机上运行命令，登录到其它计算机或在计算机之间拷贝文件（rsh、rlogin 和 rcp）。**【安全隐患】**

ntalk 包括 Internet talk 协议的客户端和服务器程序，你可以用它与在不同计算机上的人进行交谈（chat）。**【安全隐患】**

telnet 是用得很多的登录远程计算机的协议。**【安全隐患】**

Applications/Publishing:

ghostscript 是一套软件包括：PostScript™解释器、C 语言的函数库（ghostscript 函数库实现了对 PostScript 语言的图形操作）和 PDF 文件的解释器。**【没必要】**

ghostscript font 是一些 PostScript™字体，ghostscript 解释器要用到这些字体。同时，这些字体也是 ghostscript 和 X11 共享的。**【没必要】**

mpage 把纯文本的文件和 PostScript™文件输出到 PostScript 打印机上，可以在一张纸上打印多于一页的内容。**【没必要，我们的服务器上没装打印机】**

rhs-printfilter 包括一组打印驱动，这主要是和 RedHat 的 printtool 结合使用的。。**【没必要，我们的服务器上没装打印机】**

Applications/System:

arpwatch 包括 arpwatch 和 arpsnmp 两个程序。arpwatch 和 arpsnmp 都是网络监控程序。都是用来监控以太网和 FDDI 网络流量并且建立以太网网址（物理地址）和 IP 地址之间对应关系的数据库，如果两者的对应关系发生变化了，会自动用 email 报告。**【没必要】**

bind-utils 包括一套工具用来查询 DNS（域名服务器）以获得 Internet 上主机的信息。**【在后面的章节我们会自己编译】**

knfsd-clients 包括 showmount 程序。showmount 查询远程主机的 mount daemon 以获取远程主机上的 NFS 信息。**【安全隐患】**

procinfo 命令可以从“/proc”（Linux 内核虚拟出来的目录）目录获取系统信息，并用适当的格式显示在标准输出上。**【没必要，我们用别的方法】**

rdate 根据 RFC 868 协议可以从网络中的其它计算机上获取日期和时间信息。**【安全隐患】**

rdist 程序维护多台主机上相同文件的多个拷贝。如果可能，rdist 会保留文件的 owner，group，mode 和 mtime 这些属性，而且它还可以动态地更新正在运行的程序。**【安全隐患】**

ucd-snmp 包括各式各样的用于 UCD-SNMP 网络管理的实用工具。**【安全隐患，没必要】**

screen 工具允许你在一个终端上，同时登录多次。screen 对于使用 telnet 登录远程服务器或使用哑终端的用户比较有用。**【没必要】**

Documentation:

indexhtml 包括一些 HTML 文件以及一些图片，在你成功安装 RedHat Linux 之后，作为浏览器的欢迎界面。**【没必要】**

System Environment/Base:

chkfontpath 是简单的命令行程序，用来添加、删除和列出 X Window 的字体路径。**【没必要】**

NIS 为网络上的所有计算机提供网络信息，如：登录名、口令、家目录和组信息。**【安全隐患】**

System Environment/Daemons:

XFree86-xfs 是 XFree86 的字体服务程序。能为远程的 X server 提供字体。（xfs 支持 TrueType™字体）**【没必要】**

lpr 提供管理打印服务的基本工具。**【没必要，我们的服务器上不装打印机】**

portmapper 是一个安全工具，可以防止别人盗用 NIS、NFA 和其它敏感的信息。portmapper 管理 RPC（远程调用）连接。象 NFS 和 NIS 这些协议都要用到 RPC。**【没必要，安全隐患】**

pidentd 包含 identd。identd 是用来实现 RFC1413 身份验证服务的。identd 查询 TCP/IP 连接，返回用户名以及其它一些关于连接进程的信息。**【没必要，网络上几乎不会要求发送端运行 identd，因为很多计算机都没有安装 identd 而且很多人把它关掉】**

routed 是路由 daemon，接受 RIP 并且对外广播网络路由情况的 RIP，这样才能维护当前的路由表。路由表对网络上的计算机是很重要的。有了路由表，计算机才能知道往哪儿发 IP 包。**【没必要，安全隐患】**

rusers 允许用户查询连接在本地网络的计算机上的已登录用户的信息。rusers 命令的输出格式很像 who 命令，但是列出的是一组或所有局域网上计算机的登录用户信息。**【安全隐患】**

rwho 命令的输出格式也很像 who 命令，不过它可以显示那些运行 rwho daemon 的本地网上的计算机的登录用户信息。**【安全隐患】**

tftp 提供 TFTP 协议的用户界面，允许用户上传和下传远程计算机上的文件。TFTP（Trivial File Transfer）协议通常用在启动无盘工作站。**【安全隐患，没有必要】**

ucd-snmp 提供对 SNMP 协议的支持。SNMP（Simple Network Management Protocol）是一个网络管理协议。**【安全隐患，没有必要】**

System Environment/Libraries:

XFree86-libs 包含 X 程序运行所需要的共享库，这些共享库统一放在一个软件包里是为了减少磁盘空间。（X Window 的 Client/Server 结构，允许本机不装 X Server 而运行 X 程序——也就是 X 客户。如果想要运行 X 程序，在本机上就要装这个库）。**【没有必要，我们不打算使用 X 程序】**

libpng 是用来处理 PNG（Portable Network Graphics）图形文件的函数库。PNG 是类似 GIF 的位图文件格式。**【没有必要】**

User Interface/X:

XFree86-75dpi-fonts 是用于 X Window 的 75dpi 字体。

uwr-fonts 是免费的 35 种标准 PostScript™字体。主要用于 ghostscript。**【没有必要】**

下面安装程序开始格式化分区了，格式化完成之后，开始安装软件包。

怎样使用 rpm 命令

这一节主要介绍如何使用 rpm 命令在 Linux 系统上安装、卸载、升级、查询、列出清单、检查和编译 RPM 软件包。

- 安装 RPM 包:

```
[root@deep]# rpm -ivh foo-1.0-2.i386.rpm
```

RPM 包的文件名的含义是这样的：比如,foo-1.0-2.i386.rpm，软件包的名字是 foo，版本 1.0，发布号 2，适用于 i386 体系结构。

- 卸载 RPM 包:

```
[root@deep]# rpm -e foo
```

注意：用软件包的名字（foo）而不是文件名（foo-1.0-2）。

- 升级 RPM 包:

```
[root@deep]# rpm -Uvh foo-1.0-2.i386.rpm
```

这个命令自动卸载旧的 foo 软件包，再安装新的。通常都是用“rpm –Uvh”来安装软件包，因为即使没有安装过旧版本的软件，它也能正常运行，不受丝毫影响。

- 查询 RPM 包:

```
[root@deep]# rpm -q foo
```

这个命令显示已安装的软件包的名字、版本、发布号。同时，这个命令还可以用来查看软件包是否安装。

- 显示软件包信息:

```
[root@deep]# rpm -qi foo
```

这个命令显示软件包的信息，包括：名字，版本以及软件的描述。

- 显示软件包中的文件:

```
[root@deep]# rpm -ql foo
```

列出 RPM 包中文件的清单。

- 检查软件包的签名:

```
[root@deep]# rpm --checksig foo
```

这个命令用来检查软件包的 PGP 签名，以确保软件包的完整性以及没有被别人改动过。PGP 的配置信息必须从配置文件中读入。在把软件包安装到系统中之前，要用这个命令先检查一下软件包。使用这个命令要求你先安装 GnuPG 或 pgp。

- 用以源代码发布的 RPM 包来安装软件:

```
[root@deep]# rpm -ivh --rebuild foo.src.rpm
```

这个命令会编译“foo”这个软件包，在“/usr/src/redhat/RPMS/i386/”目录下生成二进制的 RPM 包。然后，你就可以用上面介绍的命令，正常地安装软件包了。

启动和停止 daemon 服务

init 是内核在引导的时候运行的程序。它负责管理那些在引导的时候要启动的进程。这些进程包括：Apache daemon、网络 daemon 和其它你想在引导时运行的进程。

那么 init 是怎么启动和停止服务呢？每一个启动脚本都可以接受一个参数，参数的值为：“start”或“stop”。这些脚本在“/etc/rc.d/init.d/”目录下。你也可以手工运行这些脚本，用类似下面的命令：

例如：

- 在 Linux 上手工启动 httpd Web 服务器

```
[root@deep]# /etc/rc.d/init.d/httpd start
```

- 停止 httpd Web 服务器

```
[root@deep]# /etc/rc.d/init.d/httpd stop
```

可以在“/etc/rc.d/init.d/”目录下查看一下有什么服务，然后，用参数“start”或“stop”，启动或停止服务。

安装完服务器之后必须卸载的软件

RedHat Linux 在默认情况下会安装一些预置的软件，而且在安装的过程中，不能够选择不安装。因此，必须在安装完成之后卸载下面这些软件：

pump	apmd	lsapnptools	redhat-logos
mt-st	kernel-pcmcia-cs	Setserial	redhat-release
eject	linuxconf	kudzu	gd
bc	getty_ps	raidtools	pciutils
mailcap	setconsole	gnupg	

用下面的 rpm 命令卸载这些软件：

```
[root@deep]# rpm -e softwarenames
```

softwarenames 在这里指的是软件包的名字，如：foo。

apmd、kudzu 和 sendmail 是 daemon 进程，在卸载它们之前最好先停止这些进程。

停止这些进程用如下的命令：

```
[root@deep]# /etc/rc.d/init.d/apmd stop
[root@deep]# /etc/rc.d/init.d/sendmail stop
[root@deep]# /etc/rc.d/init.d/kudzu stop
```

现在你可以象卸载其它程序那样卸载它们，步骤如下：

第一步

卸载这些软件包。

```
[root@deep]# rpm -e pump mt-st eject bc mailcap apmd kernel-pcmcia-cs linuxconf
getty_ps setconsole isapnptools setserial kudzu raidtools gnupg redhat-logos
redhat-release gd pciutils
```

第二步

手工删除 linux.conf-installed 文件。

```
[root@deep]# rm -f /etc/conf.linuxconf-installed
```

注意：这是和 linuxconf 程序相关的的配置文件，必须手工删除。

如果你有 IDE 硬盘，hdparm 程序是必须要的，所以一定要保留下来。否则，你可以把它从硬盘中卸载掉。

kdbconfig、mouseconfig 和 timeconfig 是用来设置键盘类型、鼠标类型和时区的。在这些都设置好之后，就很少有机会再用到它们了。所以，可以把它们卸载掉。等需要改变键盘、鼠标和时区的时候，可以再从 CD-ROM 用 rpm 命令安装。

sendmail、procmail 和 mailx 在通常情况下是必须要的。因为服务器上运行的各种各样的服务会用这些程序发消息给 root 用户，以建立系统日志（syslog）。

sendmail 是一个强大的邮件传送代理（Mail Transport Agent，简称 MTA），可以把邮件从一台计算机发送到另一台计算机。它实际上所起的作用就是通过网络或 Internet 把 email 传送到 email 的目的地。sendmail 有很多不同的配置方式：可以做为一个把邮件转给邮件集中服务器（Mail Hub Server）的内部邮件服务器，也可以做为单独的 mail 服务器，或者做为网络上所有 sendmail 服务器的中央邮件集中服务器（Central Mail Hub Server）。因此，可以根据需要配置不同的 sendmail 服务器。所以，你要先把 sendmail 卸载掉，参考本书 sendmail 的配置和安装的相关章节，根据需要建立自己的 sendmail 服务器。

sendmail 并不是自己处理邮件的分发，它是通过运行其它程序来完成这项工作。procmail 是 RedHat Linux 用于本地邮件分发的分发代理(delivery agent)。因此，procmail 只要装在中央邮件集中服务器（Central Mail Hub Server）上就行了。所以，没有必要在内部所有运行 sendmail 服务的计算机上都安装 procmail 程序。因为，这些内部的计算机会通过“/bin/mail”或 sendmail 把邮件转发到中央邮件集中服务器上。

Pump DHCP（Dynamic Host Configuration Protocol）和 BOOTP（Boot Protocol）协议允许在 IP 网络上的设备，从服务器上获取它们的网络配置信息（IP 地址、子网掩码、广播地址，等等）。**【没有必要】**

mt-st 软件包包括 mt 和 st 磁带机驱动管理程序。mt（用于磁性磁带驱动器（magnetic tape drivers））和 st（用于 SCSI 磁带驱动器）可以控制磁带的回卷、弹出、跳过文件或块以及其它。**【没有必要】**

eject 程序允许用户用软件弹出可移动的介质，如：CD-ROM、软盘以及 Jaz 或 Zip 盘。**【没有必要】**

bc 软件包包括：bc 和 dc 两个应用程序，bc 是一种非常精确的用于处理算术运算的语言。dc 是基于堆栈的交互式计算器，是用在文本模式下的。**【没有必要】**

mailcap 文件是用于 metamail 程序的。metamail 通过读 mailcap 文件来确定怎么显示非文本的和多媒体的邮件信息。**【没有必要】**

apmd 是高级能源管理（Advanced Power Management）daemon 和实用工具。它可以监控笔记本电脑的电池使用情况。当电池电压过低的时候，它会给出警告。**【没有必要】**

所有支持 PCMCIA 的笔记本电脑都要用到 kernel-pcmcia-cs 软件包。PCMCIA 卡是很小的卡片，可以是 SCSI 卡或是 modem。**【没有必要】**

linuxconf 是非常好的系统配置工具。linuxconf 提供四种界面供你选择：命令行、文本菜单、X Window 的图形界面和基于 Web 的界面。**【没有必要，有 bug 的程序】**

getty_ps 软件包包括 getty 和 uugetty 两个程序，这两个程序是 RedHat Linux 系统中实现进程登录的基本程序。getty 和 uugetty 是用于接受控制台或终端的登录请求。**【没有必要】**

setconsole 是基本的系统工具，用在为一个新的控制台创建/etc/inittab、/dev/systty 和/dev/console 文件。控制台可以是本地的终端（直接通过显示卡连接到系统上）或是通过串口连接的控制台。**【没有必要】**

ispnptools 软件包包括用于配置 ISA 接口的即插即用（Plug-and-Play）卡，符合 PnP ISA Specification 1.0a 标准。**【没有必要】**

kudzu 是系统引导的时候运行的硬件探测程序，可以检测到哪些硬件从系统中删除或添加到系统中。**【没有必要】**

raidtools 软件包包括在 Linux 系统中建立和维护软件 RAID（磁盘阵列，用于容错和提高性能，需要两个以上的磁盘）。**【取决于你用不用 RAID】**

GnuPG 是 GNU 的数据通讯和存储的安全工具。它可以用来加密数据和创建数字签名。包括高级的密匙管理工具，而且遵循 RFC2440 中描述的 OpenPGP 的国际标准。**【我们会在后面的章节介绍如何自己编译】**

redhat-logos 包括 RedHat “Shawdow Man”和 RPM 的标志图片。**【没有必要】**

redhat-release 是 RedHat Linux 的发行文件。**【没有必要】**

gd 是处理 gif 文件的图形函数库。gd 可以用于显示图像（线条、弧、文本、多种颜色、从其它图像中剪切和粘贴、填充），并把结果保存为 gif 文件。**【没有必要】**

pciutils 包括各种各样的工具，用于检测和设置连接到 PCI 总线上的设备。**【我们用其它的方法】**

kbdconfig 是文本模式的程序，提供了设置键盘映像的一个简单的界面。**【没有必要】**

mousconfig 是基于文本模式的鼠标设置程序。mouseconfig 在 RedHat Linux 系统中，创建配置和使用鼠标所需要的文件和链接。**【没有必要】**

timeconfig 包括：timeconfig 和 setclock。timeconfig 提供了一个简单的文本模式的工具，用来配置/etc/sysconfig/clock 和/etc/localtime 中的时间参数。**【没有必要】**

procmail 是 RedHat Linux 用于处理本地邮件发送的程序。除了发送邮件，procmail 还可以用于自动过滤、排序和其它处理 mail 的工作。而且，procmail 是 SmartList 邮件列表程序的基础。**【只用在邮件集中服务器上】**

安装完服务器之后必须安装的软件

为了可以在你的服务器上编译软件，你还必须安装下面的 RPM 软件包。这部分安装是非常重要的，要求你安装所有如下所述的软件包。所有的这些软件都在 RedHat 6.1 第一张光盘的“/RedHat/RPMS”目录下，而且都是使你的 Linux 系统能够编译程序所必须要的。

- 第一步

首先，先 mount 上 CD-ROM 驱动器，转到“RPMS”子目录下。

mount 上 CD-ROM 驱动器和转到 RPMS 目录用下面的命令：

```
[root@deep]# mount /dev/cdrom /mnt/cdrom/
[root@deep]# cd /mnt/cdrom/RedHat/RPMS/
```

下面是我们需要安装的软件包，只有安装了这些软件包才在 Linux 系统上编译程序。请记住，这不过是能够编译“.tar.gz”软件最少必须的软件包。有一些软件的编译可能还需要用到别的特殊的软件包。这些特殊的软件包在 RedHat 的光盘中都能找到。所以，编译程序的时候，如果出错了，可以先看看需要编译的软件包的 README 文件。

```
autoconf-2.13-5.noarch.rpm
m4-1.4-12.i386.rpm
automake-1.4-5.noarch.rpm
dev86-0.14.9-1.i386.rpm
bison-1.28-1.i386.rpm
byacc-1.9-11.i386.rpm
cdecl-2.5-9.i386.rpm
cpp-1.1.2-24.i386.rpm
cproto-4.6-2.i386.rpm
ctags-3.2-1.i386.rpm
egcs-1.1.2-24.i386.rpm
```

```
ElectricFence-2.1-1.i386.rpm
flex-2.5.4a-7.i386.rpm
gdb-4.18-4.i386.rpm
kernel-headers-2.2.12-20.i386.rpm
glibc-devel-2.1.2-11.i386.rpm
make-3.77-6.i386.rpm
patch-2.5-9.i386.rpm
```

注意：最好把这些软件一起全装了，省得在安装这些 RPM 包的时候总出现一些相关出错的信息。

- 第二步

安装所有这些软件只要用一个命令就够了。这个命令是：

```
[root@deep]# rpm -Uvh autoconf-2.13-5.noarch.rpm m4-1.4-12.i386.rpm
automake-1.4-5.noarch.rpm dev86-0.14.9-1.i386.rpm bison-1.28-1.i386.rpm
byacc-1.9-11.i386.rpm cdecl-2.5-9.i386.rpm cpp-1.1.2-24.i386.rpm
cproto-4.6-2.i386.rpm ctags-3.2-1.i386.rpm egcs-1.1.2-24.i386.rpm
ElectricFence-2.1-1.i386.rpm flex-2.5.4a-7.i386.rpm gdb-4.18-4.i386.rpm
kernel-headers-2.2.12-20.i386.rpm glibc-devel-2.1.2-11.i386.rpm
make-3.77-6.i386.rpm patch-2.5-9.i386.rpm
```

- 第三步

为了让所有的改变都生效，必须退出再重新登录。退出的命令是：

```
[root@deep]# exit
```

安装和编译完在服务器上需要的所有软件之后，最好把上面几步安装的软件包都卸掉，除非有什么特殊的需要。这样做，其中一个原因是：如果黑客成功地入侵了你的服务器，他也不可能用上面这些软件来编译软件或改变二进制程序。同时，把它们卸载掉也释放了很多磁盘空间，这样当进行系统的安全性和一致性检查的时候，可以加快扫描所有文件的时间（文件少了）。

当然，有很多建立服务器的方法和策略，我在这本书中介绍的不过是我自己的观点，我的观点如下：

第一：每台服务器最好只安排一个特定的任务。你不应该把所有的服务都放在一台服务器上，否则，这台服务器的速度会受到影响（系统的资源要同时分给很多个进程），而且会降低系统的安全（在同一台服务器上运行太多的服务，黑客就有更多的机会找到系统的安全漏洞）。

第二：不同的服务器完成不同的任务。这样有利于简化管理（你可以清楚地知道每台服务器具体的用途、运行哪些服务、客户计算机（client）可以访问哪些端口，

你也可以知道在日志文件中会看到什么，等等），而且还可以更好更灵活地控制每台服务器（专门用于邮件、Web、数据库、开发、备份，等等）。

举一个例子，一台专门用作开发和测试的服务器，就没有必要象上面介绍的那样，每一次装软件之前都要先装编译器，装完软件之后又把编译器卸掉，完全可以把编译器保留着。如果想知道更多的编译器的信息，请参考第八章《编译器的功用》。

服务器上已经安装的程序

因为我们选择定制安装我们的 Linux 系统，下面是服务器上已安装的程序列表。这个列表必须和“/tmp”目录下的“install.log”文件一致，否则你就是没装全。不要忘了装全上一节《安装完服务器之后必须安装的软件》中的那些软件包，不然你编译一些程序的时候就会遇到问题。

```
Installing setup.
Installing filesystem.
Installing basesystem.
Installing ldconfig.
Installing glibc.
Installing shadow-utils.
Installing mktemp.
Installing termcap.
Installing libtermcap.
Installing bash.
Installing MAKEDEV.
Installing SysVinit.
Installing XFree86-SVGA.
Installing chkconfig.
Installing apmd.
Installing arpwatch.
Installing ncurses.
Installing info.
Installing fileutils.
Installing grep.
Installing ash.
Installing at.
Installing authconfig.
Installing bc.
Installing bdflush.
Installing binutils.
Installing bzip2.
Installing sed.
Installing console-tools.
Installing e2fsprogs.
Installing rmt.
```

```
Installing cpio.
Installing cracklib.
Installing cracklib-dicts.
Installing crontabs.
Installing textutils.
Installing dev.
Installing diffutils.
Installing ed.
Installing eject.
Installing etcskel.
Installing file.
Installing findutils.
Installing gawk.
Installing gd.
Installing gdbm.
Installing getty_ps.
Installing glib.
Installing gmp.
Installing gnupg.
Installing gpm.
Installing groff.
Installing gzip.
Installing hdparm.
Installing initscripts.
Installing ipchains.
Installing isapnptools.
Installing kbdconfig.
Installing kernel.
Installing kernel-pcmcia-cs.
Installing kudzu.
Installing ld.so.
Installing less.
Installing libc.
Installing libstdc++.
Installing lilo.
Installing pwdb.
Installing pam.
Installing sh-utils.
Installing redhat-release.
Installing linuxconf.
Installing logrotate.
Installing losetup.
Installing lsof.
Installing mailcap.
```

```
Installing mailx.
Installing man.
Installing mingetty.
Installing mkbootdisk.
Installing mkinitrd.
Installing modutils.
Installing mount.
Installing mouseconfig.
Installing mt-st.
Installing ncompress.
Installing net-tools.
Installing netkit-base.
Installing newt.
Installing ntsysv.
Installing passwd.
Installing pciutils.
Installing perl.
Installing procmail.
Installing procps.
Installing psmisc.
Installing pump.
Installing python.
Installing quota.
Installing raidtools.
Installing readline.
Installing redhat-logos.
Installing rootfiles.
Installing rpm.
Installing sash.
Installing sendmail.
Installing setconsole.
Installing setserial.
Installing setuptool.
Installing shapecfg.
Installing slang.
Installing slocate.
Installing stat.
Installing sysklogd.
Installing tar.
Installing tcp_wrappers.
Installing tcpdump.
Installing tcsh.
Installing time.
Installing timeconfig.
```

```
Installing timed.
Installing tmpwatch.
Installing traceroute.
Installing utempter.
Installing util-linux.
Installing vim-common.
Installing vim-minimal.
Installing vixie-cron.
Installing which.
Installing zlib.
```

给终端加点颜色

给终端加一点颜色，可以帮助区分目录、文件、设备、符号连接和可执行文件。我认为加上颜色可以让人少犯错误，并且可以让人感觉 Linux 更好用。

编辑“profile”文件（vi /etc/profile）加入下面这些行：

```
# Enable Colour ls
eval 'dircolors /etc/DIR_COLORS -b'
export LS_OPTIONS='-s -F -T 0 --color=yes'
```

编辑“bashrc”文件（vi /etc/bashrc），加入下面这一行：

```
alias ls='ls --color=auto'
```

退出，再重新登录，与颜色相关的环境变量就设定好了。

使软件保持最新的版本

为了使你的软件保持最新的版本，请定期查看 RedHat Linux 的勘误网页：http://www.redhat.com/corp/support/errata/index.html。勘误网页通常可以解决 90% RedHat Linux 的系统问题。而且，RedHat 在得到安全漏洞的通知之后，如果已经找到解决方案了，就会在 24 小时之内，在勘误网站上发布出来。必须经常查看这个地方。RedHat Linux 服务器现在必须更新的软件是：

```
groff-1_15-1_i386.rpm
sysklogd-1_3_31-14_i386.rpm
initscripts-4_70-1_i386.rpm
e2fsprogs-1.17-1.i386.rpm
pam-0_68-10_i386.rpm
Linux kernel 2.2.14 (linux-2_2_14_tar.gz)
```

注意：Linux 的内核是最重要的，必须经常更新。下面的一些章节将会介绍如何为你的服务器编译一个定制的内核。

在更新之前，你可以用下面的命令查看在系统中已经安装的软件。

```
[root@deep]# rpm -q <softwarename>
```

这里<softwarename>表示想查看的软件名，比如：XFree86、telnet,等等。

第二部分：与安全和优化相关的参考资料

第三章

系统安全概要

Linux 的安全

概述

UNIX 的系统安全和系统管理员有很大的关系。安装越多的服务，越容易导致系统的安全漏洞。一些其它的操作系统，如：SCO，实际上更容易有安全漏洞，因为，为了更加“用户友好”，这些操作系统集成了更多的服务。

Linux 本身是稳定和安全的，但是它可以以不同的形式发行。在安装 Linux 时候，最好先最小化安装，然后再加上必要的软件。这样可以减小某个程序出现安全隐患的可能。如果管理得好，Linux 可以是最安全的系统。

如果在系统中有隐患存在，在网络上成千上万的自愿者就会指出隐患，并给出修正。大的公司，比如：商业软件公司，只能把有限的人力用于解决这方面的问题。把问题都公开出来，对它们来说可能没有什么好处，而且通过正规的发行和升级渠道，一个小小修正到用户手中都要很长的时间。也许这些修正会以补丁的方式出现，但是商业软件的系统管理员都倾向于使用现成的软件，而且自以为这些软件都是很专业的。

在编程的时候随时可能发生错误，但是，当有上万个人在看程序的源代码，这些错误就能很快地被发现。现在全球共有一千二百万的 Linux 拷贝，想想看，这是一种什么情况。如果系统有安全漏洞，很容易就被某个人发现了。

这一章将讨论一些保证服务器安全的一般方法。我们假定这台服务器是连在 Internet 上的。在通常情况下，有些服务，如：NFS、Samba、Imap 和 Pop 只供内部用户使用的。当然，也可以把这台服务器和外界隔离，这样就可以简单地避免外部的入侵。但是这没有什么实际意义，我们下面要介绍的是如何避免来自外部的和内部的攻击。

1. 基础知识

尽量少让外人知道有关系统的信息。有时候简单地用 finger 程序就能知道不少系统信息，比如：有多少用户、管理员什么时候登录、什么时候工作、他们是谁、谁现在正在使用这个系统以及其它有利于黑客猜出用户口令的信息。你可以用一个强大的 finger daemon 和 tcpd 限制可以连接到服务器的用户以及他们可以知道的有关系统的信息。但是，最好还是把 finger 软件包卸载掉。

日志是了解 Linux 系统运行情况的唯一方法。当然，这是以黑客没有破坏日志文件为前提的，但是这种情况很少见。把所有的连接都记录在日志中，可以发现攻击者和他们试图进行的攻击。如果你看不懂日志，可以向别人请教，一定要学会看懂它们。向别人请教你不懂的东西这是很正常的，不要不好意思。我自己就是从不断地犯错误和改正错误中学到不少知识的。因为骄傲自大而不能学到东西和本身的资质差是两码事。看下面的《31. 创建所有重要的日志文件的硬拷贝》，你可以学到怎样管理日志文件的一些有用的知识。

限制系统中 SUID 的程序。SUID 的程序是以 root（UNIX 世界中的上帝）权限运行的程序。有时候这是必须的，但是在多数情况下这没有必要。因为 SUID 程序可以做任何 root 可以做的事，这样有更多的机会出现安全隐患。也许，它们有时候会带来安全隐患，有时候不会，但是黑客可以利用 SUID 的程序来破坏系统的安全。这就是一种叫 exploit（译者注：也就是利用缓冲溢出的程序）的程序的由来。exploit 程序是一种利用 SUID 程序的程序或脚本，具有很大的破坏力，可以用来得到 root 的 shell、获取 password 文件、读其他人的邮件、删除文件，等等。这方面的知识可以参考《34. 带“s”位的程序》。

2. BIOS 安全，设定引导口令

禁止从软盘启动，并且给 BIOS 加上密码。每次启动的时候都手工检查一下 BIOS，这样可以提高系统的安全性。禁止从软盘启动，可以阻止别人用特殊的软盘启动你的计算机；给 BIOS 加上密码，可以防止有人改变 BIOS 的参数，比如：允许从软盘启动或不用输入口令就可以引导计算机。

3. 安全策略

有一点很重要而且必须指出的是：如果你不知道要保护什么，那么更本没有办法保证系统的安全。所以必须要有一个安全策略，基于这样的一个策略才可以决定哪些东西允许别人访问，哪些不允许。如何制定一个安全策略完全依赖于你对于安全的定义。下面的这些问题提供一些一般性的指导方针：

- 你怎么定义保密的和敏感的信息？

- 你想重点防范哪些人？
- 远程用户有必要访问你的系统吗？
- 系统中有保密的或敏感的信息吗？
- 如果这些信息被泄露给你的竞争者和外面的人有什么后果？
- 口令和加密能够提供足够的保护吗？
- 你想访问 Internet 吗？
- 你允许系统在 Internet 上有多大的访问量？
- 如果发现系统被黑客入侵了，下一步该怎么做？

这个列表很短，真正的安全策略可能包含比这多得多的内容。可能你要做的第一件事是：评估一下自己的“偏执”程度。任何一个安全策略多少都有一定程度的“偏执”，也就是确定到底在多大程度上相信别人，包括内部的人和外部的人。安全策略必须在允许用户合理地使用可以完成工作所必须的信息和完全禁止用户使用信息之间找到平衡点。这个平衡点就是由系统策略决定的。

4. 口令

这章的 Linux 安全概要就从口令的安全开始讲起。许多人都把所有的东西保存在计算机上，防止别人查看这些信息的方法就是用口令把计算机保护起来。没有什么东西是绝对安全的。与常识相反的是：无法破解的口令是不存在的。只要给足时间和资源，所有的口令都能用社会工程（译者注：原文是 social engineering，找不出更好的翻译，大致的意思是用社会和心理学的知识，而不是用纯粹的技术手段）或强行计算的方法猜出来。

通过社会工程或其它方法获得服务器的口令是最简单和最流行的入侵服务器的方法。决大多数的技术支持人员很容易获得其他用户的口令，因为用户的安全意识很差而且很轻易就相信自己的同事，特别是帮助自己解决问题的人。有很多登记在案的成功入侵就是因为一些别有用心的人利用安全管理上的松懈而获得成功的。有时候，在特定的时间在特定的地点，上级或老板对员工喊话就有可能泄露机密，导致可怕的后果。

因为破解口令是一项很耗时间和资源的工作，所以应该使得口令文件难于破解，这样即使黑客获取了口令文件也不能轻易破解。作为一个系统管理员，自己在每个周末运行一下口令破解程序，是保证系统安全的好方法。这有利于尽早地发现和替换那些很容易被猜出来的口令。而且，还要有一个好的口令检查机制，在用户选择新口令或改变旧口令的时候，来排除那些有安全隐患的口令。那些字典里的单词、或者全是大写或全是小写的以及没有包含数字或特殊字符的字符串是不能用来做口令的。我建议用下面的规则选择有效的口令：

- 口令至少要有 6 个字符，最好包含一个以上的数字或特殊字符。
- 口令不能太简单，所谓的简单就是很容易猜出来，也就是用自己的名字，电话号码、生日、职业或者其它个人信息作为口令。
- 口令必须是有有效期的，在一段时间之后就要更换口令。
- 口令在这种情况下必须作废或者重新设定：如果发现有人试图猜测你的口令，而且已经试过很多次了。

5. 口令长度

安装完 Linux 系统之后默认的最小口令长度为 5。这就是说一个新的用户可以访问服务器，那么他的口令必须多于 5 字符。但是这样是不够安全的，最好口令的长度能够大于 8。可以强制用户使用 8 个字符以上的口令。编辑“/etc/login.defs”文件，把最小口令长度由 5 改成 8。找到 PASS_MIN_LEN 5 这一行，改为：PASS_MIN_LEN 8 。“login.defs”是很重要的配置文件。可以在这个文件中设定一些其它的安全策略，比如：口令的有效期。

6. root 帐号

“root”帐号是 Unix 系统中享有特权的帐号。“root”帐号是不受任何限制和制约的。因为系统认为 root 知道自己在做些什么，而且会按 root 说的做，不问任何问题。因此，可能会因为敲错了一个命令，导致重要的系统文件被删除。用 root 帐号的时候，要非常非常小心。因为安全原因，在不是绝对必要的情况下，不要用 root 帐号登录。特别要注意的是：不在自己的服务器上的时候，千万不要在别的计算机上用“root”登录自己的服务器。这是非常非常非常糟糕的一件事。

7. 加密

加密时要用到密匙，密匙是一个特殊的数字，把密匙和需要加密的信息经过加密算法加密之后，只有知道密匙的人才能把信息读出来。如果所有的计算机主机都在你的控制下，加密当然是一个好方法。但是，如果其中一台“被信任的”主机被黑客控制了，你马上就有危险了。而且不仅仅是用户的帐号和口令有危险了。在通常情况下，加密是用来保证机密信息在系统中传送的安全。如果一台计算机被控制了，那么这些加密信息就会让人知道或是泄密了。有一个好的安全策略，这种危险的可能性会降到最低，但是如果某台主机的密匙被泄露出去，那么危险始终存在。使用如：OpenSSL、SSH 和 MD5 这样的加密技术是非常有用的，可以参考后面的章节。

8. “/etc/exports”文件

如果通过 NFS 把文件共享出来，那么一定要配置“/etc/exports”文件，使得访问限制尽可能的严。这就是说，不要用通配符，不允许对根目录有写权限，而且尽可能只给只读权限。编辑 exports 文件（vi /etc/exports）加入：

例如：

```
/dir/to/export host1.mydomain.com(ro,root_squash)
/dir/to/export host2.mydomain.com(ro,root_squash)
```

“/dir/to/export”是你想共享出来的目录，host.mydomain.com 是允许访问这个目录的计算机。<ro>代表只读，<root_squash>代表不允许对根目录进行写操作。使这些改变生效，你还要运行“/usr/sbin/exportfs –a”命令。

注意：在服务器上装 NFS 服务是会有安全隐患的，就我个人而言，不建议你使用 NFS。

9. 禁止使用控制台程序

一个最简单而且最常用的保证系统安全的方法就是禁止使用所有的控制台程序，如：shutdown 和 halt。可以运行下面的命令来实现：

```
[root@deep]# rm -f /etc/security/console.apps/servicename
```

这里 servicename 是你要禁止的控制台程序名。除非你使用 xdm，否则不要把 xserver 文件删掉，如果这样除了 root 之外，没有人可以启动 X 服务器了。（如果使用 xdm 启动 X 服务器，这时 root 是唯一需要启动 X 服务器的用户，这才有必要把 xserver 文件删掉）。例如：

```
[root@deep]# rm -f /etc/security/console.apps/halt
[root@deep]# rm -f /etc/security/console.apps/poweroff
[root@deep]# rm -f /etc/security/console.apps/reboot
[root@deep]# rm -f /etc/security/console.apps/shutdown
[root@deep]# rm -f /etc/security/console.apps/xserver (如果删除，只有 root 可以启动 X).
```

这些命令就可以禁止所有的控制台程序：halt、poweroff、reboot 和 shutdown。记住，只有装了 Xwindow，删除 xerver 文件才会有效果。

注意：根据我们前一章的介绍安装服务器，Xwindow 是没有安装上的，上面说的那些文件可能不会出现在“/etc/security”目录下的，如果这样就可以不管这一节介绍的方法。

10. 禁止控制台的访问

为了禁止所有的控制台访问，包括程序和文件，请在“/etc/pam.d/”目录下的所有文件中，给那些包含 pam_console.so 的行加上注释。这一步是上一节《禁止使用控制台程序》的延续。下面的脚本可以自动完成这项工作。转成 root 身份，创建 disabling.sh 脚本文件（touch disabling.sh），接着加入下面这些行：

```
# !/bin/sh
cd /etc/pam.d
for i in * ; do
sed '/[^#].*pam_console.so/s/^/#/' < $i > foo && mv foo $I
done
```

用下面的命令使脚本有可执行的权限，并执行它：

```
[root@deep]# chmod 700 disabling.sh
[root@deep]# ./disabling.sh
```

这样“/etc/pam.d”目录下所有文件中包含“pam_console.so”的行都被加上注释。这个脚本运行完之后，可以把它从系统中删掉。

11. “/etc/inetd.conf”文件

inetd，也叫作“超级服务器”，根据网络请求装入网络程序。“inetd.conf”文件告诉 inetd 监听哪些网络端口，为每个端口启动哪个服务。把 Linux 系统放在任何的网络环境中，第一件要做的事就是了解一下服务器到底要提供哪些服务。不需要的那些服务应该被禁止掉，最好卸载掉，这样黑客就少了一些攻击系统的机会。查看“/etc/inetd.conf”文件，了解一下 inetd 提供哪些服务。用加上注释的方法（在一行的开头加上#号），禁止任何不需要的服务，再给 inetd 进程发一个 SIGHUP 信号。

第一步：把文件的许可权限改成 600。

```
[root@deep]# chmod 600 /etc/inetd.conf
```

第二步：确信文件的所有者是 root。

```
[root@deep]# stat /etc/inetd.conf
```

这个命令显示出来的信息应该是：

```
File: "/etc/inetd.conf"
Size: 2869 Filetype: Regular File
Mode: (0600/-rw-------) Uid: ( 0/ root) Gid: ( 0/ root)
Device: 8,6 Inode: 18219 Links: 1
Access: Wed Sep 22 16:24:16 1999(00000.00:10:44)
Modify: Mon Sep 20 10:22:44 1999(00002.06:12:16)
Change: Mon Sep 20 10:22:44 1999(00002.06:12:16)
```

第三步：编辑“inetd.conf”文件（vi /etc/inetd.conf），禁止所有不需要的服务，如：ftp、 telnet、 shell、 login、 exec、talk、ntalk、 imap、 pop-2、pop-3、finger、auth，等等。如果你觉得某些服务有用，可以不禁止这些服务。但是，把这些服务禁止掉，系统受攻击的可能性就会小很多。改变后的“inetd.conf”文件的内容如下面所示：

```
# To re-read this file after changes, just do a 'killall -HUP inetd'
#
#echo stream tcp nowait root internal
#echo dgram udp wait root internal
#discard stream tcp nowait root internal
#discard dgram udp wait root internal
#daytime stream tcp nowait root internal
#daytime dgram udp wait root internal
#chargen stream tcp nowait root internal
#chargen dgram udp wait root internal
#time stream tcp nowait root internal
#time dgram udp wait root internal
#
# These are standard services.
#
#ftp stream tcp nowait root /usr/sbin/tcpd in.ftpd -l -a
#telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
#
# Shell, login, exec, comsat and talk are BSD protocols.
#
#shell stream tcp nowait root /usr/sbin/tcpd in.rshd
#login stream tcp nowait root /usr/sbin/tcpd in.rlogind
#exec stream tcp nowait root /usr/sbin/tcpd in.rexecd
#comsat dgram udp wait root /usr/sbin/tcpd in.comsat
#talk dgram udp wait root /usr/sbin/tcpd in.talkd
#ntalk dgram udp wait root /usr/sbin/tcpd in.ntalkd
#dtalk stream tcp wait nobody /usr/sbin/tcpd in.dtalkd
#
```

```
# Pop and imap mail services et al
#
#pop-2 stream tcp nowait root /usr/sbin/tcpd ipop2d
#pop-3 stream tcp nowait root /usr/sbin/tcpd ipop3d
#imap stream tcp nowait root /usr/sbin/tcpd imapd
#
# The Internet UUCP service.
#
#uucp stream tcp nowait uucp /usr/sbin/tcpd /usr/lib/uucp/uucico -l
#
# Tftp service is provided primarily for booting. Most sites
# run this only on machines acting as "boot servers." Do not uncomment
# this unless you *need* it.
#
#tftp dgram udp wait root /usr/sbin/tcpd in.tftpd
#bootps dgram udp wait root /usr/sbin/tcpd bootpd
#
# Finger, systat and netstat give out user information which may be
# valuable to potential "system crackers." Many sites choose to disable
# some or all of these services to improve security.
#
#finger stream tcp nowait root /usr/sbin/tcpd in.fingerd
#cfinger stream tcp nowait root /usr/sbin/tcpd in.cfingerd
#systat stream tcp nowait guest /usr/sbin/tcpd /bin/ps -auwwx
#netstat stream tcp nowait guest /usr/sbin/tcpd /bin/netstat -f inet
#
# Authentication
#
#auth stream tcp nowait nobody /usr/sbin/in.identd in.identd -l -e -o
#
# End of inetd.conf
```

注意：改变了“inetd.conf”文件之后，别忘了给 inetd 进程发一个 SIGHUP 信号（killall –HUP inetd）。

```
[root@deep /root]# killall -HUP inetd
```

第四步：

为了保证“inetd.conf”文件的安全，可以用 chattr 命令把它设成不可改变。把文件设成不可改变的只要用下面的命令：

```
[root@deep]# chattr +i /etc/inetd.conf
```

这样可以避免“inetd.conf”文件的任何改变（意外或是别的原因）。一个有“i”属性的文件是不能被改动的：不能删除或重命名，不能创建这个文件的链接，不能往这个文件里写数据。只有系统管理员才能设置和清除这个属性。如果要改变 inetd.conf 文件，你必须先清除这个不允许改变的标志：

```
[root@deep]# chattr -i /etc/inetd.conf
```

12. TCP_WRAPPERS

在默认情况下，RedHat Linux 允许所有的服务请求。用 TCP_WRAPPERS 来保护服务器的安全，使其免受外部的攻击，比想象的要简单和轻松得多。在“/etc/hosts.deny”文件中加入“ALL: ALL@ALL, PARANOID”以禁止所有计算机访问你的服务器，然后在“/etc/hosts.allow”文件中一个一个加入允许访问你的服务器的计算机。这种作法是最安全的。

TCP_WRAPPERS 是由两个文件控制的，依次是：“/etc/hosts.allow”和“/etc/hosts.deny”。判断是依次进行的，具体的规则如下：

- 如果在“/etc/hosts.allow”文件中有匹配的项（daemon，client），那么允许访问；
- 否则，查看“/etc/hosts.deny”，如果找到匹配的项，那么访问被禁止；
- 否则，访问被允许。

第一步：编辑 hosts.deny 文件（vi /etc/hosts.deny）加入下面这些行：

```
Access is denied by default.
# Deny access to everyone.
ALL: ALL@ALL, PARANOID #Matches any host whose name does not match its address, see
bellow.
```

这样做的意思是：所有的服务、访问位置，如果没有被明确地允许，也就是在“/etc/hosts.allow”中找不到匹配的项，就是被禁止的。

注意：加上“PARANOID”参数之后，如果要在服务器上使用 telnet 或 ftp 服务，就要在服务器的“/etc/hosts”文件中加入允许使用 telnet 和 ftp 服务的客户端计算机的名字和 IP 地址。否则，在显示登录提示之前，因为 DNS 的域名解析，可能要等上几分钟时间。

第二步：编辑“hosts.allow”文件（vi /etc/hosts.allow）。例如，可以加入下面这些行（被授权访问的计算机要被明确地列出来）：

```
sshd: 208.164.186.1 gate.openarch.com
```

被授权访问的计算机的 IP 地址是：208.164.186.1，主机名是：gate.openarch.com，允许使用的服务是：sshd。

第三步：tcpdchk 是检查 TCP_WAPPERS 配置的程序。它检查 TCP_WAPPERS 的配置，并报告它可以发现的问题或潜在的问题。在所有的配置都完成了之后，请运行 tcpdchk 程序：

```
[root@deep]# tcpdchk
```

13. “/etc/aliases”文件

aliases 文件如果管理错误或管理得太粗心了就会造成安全隐患。例如：很多的软件产商都把“decode”这个别名放在 aliases 文件里。这样做的目的是为了方便通过 email 传送二进制文件。在发送邮件的时候，用户把二进制文件用“uuencode”转成 ASCII 文件，然后把结果发给接收端的“decode”。由这个别名让邮件信息通过“/usr/bin/uuencode”程序把二进制文件重新转换成 ASCII 文件。如果允许“decode”出现在 aliases 文件中，可以想象将会有什么样的安全隐患。

把定义“decode”这个别名的行从 aliases 文件中删除。同样地，每一个会运行程序的别名都要好好查看一下，很有可能要把它们删除掉。要使改动生效，还必须运行：

```
[root@deep]# /usr/bin/newaliases
```

编辑 aliases 文件（vi /etc/aliases），删除或注释掉下面这些行：

```
# Basic system aliases -- these MUST be present.
MAILER-DAEMON: postmaster
postmaster: root
# General redirections for pseudo accounts.
bin: root
daemon: root
#games: root ← remove or comment out.
#ingres: root ← remove or comment out.
nobody: root
#system: root ← remove or comment out.
#toor: root ← remove or comment out.

#uucp: root ← remove or comment out.
# Well-known aliases.
#manager: root ← remove or comment out.
#dumper: root ← remove or comment out.
#operator: root ← remove or comment out.
# trap decode to catch security attacks
```

```
#decode: root
# Person who should get root's mail
#root: marc
```

别忘了运行“/usr/bin/newaliases”使改变生效。

14. 防止 sendmail 被没有授权的用户滥用

最新版的 sendmail（8.9.3）集成了很强大的防止垃圾邮件（anti-spam）的功能，可以防止邮件服务器被没有授权的用户滥用。要实现这个功能可以通过编辑“/etc/sendmail.cf”文件，改变配置文件以阻止那些发垃圾邮件的人。

编辑“sendmail.cf”文件（vi /etc/sendmail.cf），把这一行：

```
O PrivacyOptions=authwarnings
```

改为：

```
O PrivacyOptions=authwarnings,noexpn,novrfy
```

这些改变可以防止发垃圾邮件的人使用 sendmail 中的“EXPN”和“VRFY”命令。这些命令经常被没有授权的人使用。参考本书 sendmail 配置这一节以获得更多这方面的信息。

编辑“sendmail.cf”文件（vi /etc/sendmail.cf），把这一行：

```
O SmtpGreetingMessage=$j Sendmail $v/$Z; $b
```

改为:

```
O SmtpGreetingMessage=$j Sendmail $v/$Z; $b NO UCE C=xx L=xx
```

这将改变 sendmail 接受连接时所显示的提示信息。你要把“C=xx L=xx”中的“xx”改成你所在的国家和地区编码。例如：我是这样写的“C=CA L=QC”，代表加拿大，魁北克。这个改变不会对 sendmail 有什么影响，但是 news.admin.net-abuse.email 新闻组的人建议这么做，主要是为了预防法律上的问题。

15. 使系统对 ping 没有反应

防止你的系统对 ping 请求做出反应，对于网络安全很有好处，因为没人能够 ping 你的服务器并得到任何反应。TCP/IP 协议本身有很多的弱点，黑客可以利用一些技术，把传输正常数据包的通道用来偷偷地传送数据。使你的系统对 ping 请求没有反应可以把这个危险减到最小。用下面的命令：

```
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

运行完这个命令后，系统对 ping 就没有反应了。可以把这一行加到“/etc/rc.d/rc.local”文件中去，这样当系统重新启动的时候，该命令就会自动运行。对 ping 命令没有反应，至少可以把绝大多数的黑客排除到系统之外，因为黑客不可能知道你的服务器在哪里。重新恢复对 ping 的响应，可以用下面的命令：

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

16. 不要显示系统提示信息

如果你不想让远程登录的用户看到系统的提示信息，你可以改变“/etc/inetd.conf”文件中的 telnet 设置：

```
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd -h
```

在末尾加上“-h”参数可以让 daemon 不显示任何系统信息，只显示登录提示。当然，只有在服务器上装了 telnet 服务器才有这样做的必要。

17. “/etc/host.conf”文件

Linux 用解析器（resolver）库把主机名翻译成 IP 地址。“/etc/host.conf”文件定义主机名是怎样解析的。“/etc/host.conf”文件中的项告诉解析器库用什么服务，以什么顺序解析主机名。

编辑“host.conf”文件（vi /etc/host.conf）加入下面这些行：

```
# Lookup names via DNS first then fall back to /etc/hosts.
order bind,hosts
# We have machines with multiple IP addresses.
multi on
# Check for IP address spoofing.
nospoof on
```

order 选项指明的是选择服务的顺序。上面“order bind, hosts”说的是解析器库解析文件名的时候先查询域名服务器，然后再查看“/etc/hosts”文件。因为性能和安全上的原因，最好将解析器库的查找顺序设成先查域名服务器（bind）。当然也要先安装了 DNS/BIND 软件，否则这样配置根本没有任何作用。

multi 选项决定在“/etc/hosts”文件中出现的主机能不能有多个 IP 地址（多个网络界面）。具有多个 IP 网络界面的主机被称为具有多个网络界面（multiomed），因为同时有多个 IP 地址也就意味着这台主机有多个网络界面。例如：网关服务器就有多个 IP 地址，必须把这个选项设成 ON。

nospoof 选项指明不允许 IP 伪装。IP 伪装是把自己伪装成别的计算机去欺骗其它的计算机，获得它的信任。这种攻击方法把自己伪装成别的服务器，并且与其它客户机、服务器和大型数据存储系统建立网络连接或其它类型的网络活动。不管对任何类型的服务器，这个选项都要设成 ON。

18. 路由协议

路由和路由协议会导致一些问题。IP 原路径路由（IP source routing），也就是 IP 包包含到达底目的地址的详细路径信息，是非常危险的，因为根据 RFC 1122 规定目的主机必须按原路径返回这样的 IP 包。如果黑客能够伪造原路径路由的信息包，那么它就能截取返回的信息包，并且欺骗你的计算机，让它觉得正在和它交换信息的是可以信任的主机。我强烈建议你禁止 IP 原路径路由以避免这个安全漏洞。

用下面的命令在你的服务器上禁止 IP 原路径路由：

```
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 0 > $f
done
```

把上面的命令加到“/etc/rc.d/rc.local”文件中去，你就不用在系统重新启动之后再把这些命令敲一遍。注意，上面的命令将禁止所有的网络界面（lo、ethN、pppN，等等）的原路径路由包。如果你打算安装书中介绍的 IPCHAINS 防火墙，就不必用这些命令了，因为在防火墙的脚本文件中已经包含这些命令了。

19. 使 TCP SYN Cookie 保护生效

“SYN Attack”是一种拒绝服务（DoS）的攻击方式，会消耗掉系统中的所有资源，迫使服务器重新启动。拒绝服务（这种攻击方式用巨大的信息流来消耗系统的资源，以至于服务器不能够响应正常的连接请求）是很容易被黑客利用的。在 2.1 系列的内核中，“syn cookie”只是一个可选项，并没有使其生效。想要使其生效必须用下面的命令：

```
[root@deep]# echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

把这个命令加入“/etc/rc.d/rc.local”文件中，等下次系统重新启动的时候就不必重新敲一遍了。如果打算安装 IPCHAINS 防火墙，你就没有必要用这个命令，因为它已经包含在防火墙的脚本文件里了。

20. 防火墙

安全问题的另一个解决方案是把计算机主机和内部计算机间的信息传送同外部的网络隔离开，只让内部网络与外部网络之间的信息交流，通过一个安全的网关进行。这样一个网关叫做防火墙，在下面的一些章节我们会用很大的篇幅介绍防火墙。

21. “/etc/services”文件

端口号和标准服务之间的对应关系在 RFC 1700“Assigned Numbers”中有详细的定义。“/etc/services”文件使得服务器和客户端的程序能够把服务的名字转成端口号，这张表在每一台主机上都存在，其文件名是“/etc/services”。只有“root”用户才有权限修改这个文件，而且在通常情况下这个文件是没有必要修改的，因为这个文件中已经包含了常用的服务所对应的端口号。为了提高安全性，我们可以给这个文件加上保护以避免没有经过授权的删除和改变。为了保护这个文件可以用下面的命令：

```
[root@deep]# chattr +i /etc/services
```

22. “/etc/securetty”文件

“/etc/securetty”文件允许你规定“root”用户可以从那个 TTY 设备登录。登录程序（通常是“/bin/login”）需要读取“/etc/securetty”文件。它的格式是：列出来的 tty 设备都是允许登录的，注释掉或是在这个文件中不存在的都是不允许 root 登录的。

注释掉（在这一行的开头加上＃号）所有你想不让 root 登录的 tty 设备。

编辑 securetty 文件（vi /etc/securetty）象下面一样，注释掉一些行：

```
tty1
#tty2
#tty3
#tty4
#tty5
```

```
#tty6
#tty7
#tty8
```

上面这样做的意思是只允许 root 在 tty1 上登录。我建议只允许 root 在一个 tty 设备上登录，如果从其它 tty 上登录，用“su”命令把身份转成“root”。

23. 特殊的帐号

禁止操作系统中不必要的预置帐号（每次升级或安装完都要检查一下）。Linux 系统中就提供这样一些你可能不需要的预置帐号。如果确实不需要这些帐号，就把它们删掉。系统中有越多的帐号，就越容易受到攻击。

我们假定你已经在系统中使用 shadow 口令。如果不是这样，最好在系统中加上 shadow 口令的支持，因为这样系统会更安全。如果你是按照上一章介绍的方法安装服务器，那么在“安全验证配置”这一步就已经选上“Enable Shaow Passwords”这个选项了。

- 在系统中删除一个用户可以用这个命令：

```
[root@deep]# userdel username
```

- 在系统中删除一个组可以用这个命令：

```
[root@deep]# groupdel username
```

第一步

用下面的命令删除一些不必要的用户：

```
[root@deep]# userdel adm
[root@deep]# userdel lp
[root@deep]# userdel sync
[root@deep]# userdel shutdown
[root@deep]# userdel halt
[root@deep]# userdel news
[root@deep]# userdel uucp
[root@deep]# userdel operator
[root@deep]# userdel games （如果不用 X Window 服务器,可以删除这个用户）
[root@deep]# userdel gopher
[root@deep]# userdel ftp （如果没安装匿名 ftp 服务器，可以删除这个用户）
```

第二步

输入下面的命令删除一些不必要的组：

```
[root@deep]# groupdel adm
[root@deep]# groupdel lp
[root@deep]# groupdel news
[root@deep]# groupdel uucp
[root@deep]# groupdel games (delete this group if you don't use X Window Server).
[root@deep]# groupdel dip
[root@deep]# groupdel pppusers
[root@deep]# groupdel popusers (delete this group if you don't use pop server for email).
[root@deep]# groupdel slipusers
```

第三步

在系统中加入必要的用户：

- 在系统中添加用户，用这个命令：

```
[root@deep]# useradd username
```

- 给系统中的用户添加或改变口令，用这个命令：

```
[root@deep]# passwd username
```

例如：

```
[root@deep]# useradd admin
[root@deep]# passwd admin
```

这些命令的输出是这样的：

```
Changing password for user admin
New UNIX password: somepasswd
passwd: all authentication tokens updated successfully
```

第四步

“不允许改变”位可以用来保护文件使其不被意外地删除或重写，也可以防止有些人创建这个文件的符号连接。删除“/etc/passwd”、“/etc/shadow”、“/etc/group”或“/etc/gshadow”都是黑客的攻击方法。

给口令文件和组文件设置不可改变位，可以用下面的命令：

```
[root@deep]# chattr +i /etc/passwd
[root@deep]# chattr +i /etc/shadow
[root@deep]# chattr +i /etc/group
[root@deep]# chattr +i /etc/gshadow
```

注意：如果将来要在口令或组文件中增加或删除用户，就必须先清除这些文件的不可改变位，否则就不能做任何改变。如果没有清除这些文件的不可改变位，安装那些会自动在口令文件和组文件中加入新用户的 rpm 软件包的时候，在安装过程中就会出现出错的提示。

24. 防止任何人都可以用 su 命令成为 root

如果不想任何人都可以用“su”命令成为 root 或只让某些用户有权使用“su”命令，那么在“/etc/pam.d/su”文件中加入下面两行。我建议尽量限制用户通过“su”命令成为 root。

第一步

编辑 su 文件（vi /etc/pam.d/su）在文件的头部加入下面两行：

```
auth sufficient /lib/security/pam_rootok.so debug
auth required /lib/security/pam_wheel.so group=wheel
```

加入这两行之后，“/etc/pam.d/su”文件变为：

```
#%PAM-1.0
auth sufficient /lib/security/pam_rootok.so debug
auth required /lib/security/pam_wheel.so group=wheel
auth required /lib/security/pam_pwdb.so shadow nullok
account required /lib/security/pam_pwdb.so
password required /lib/security/pam_cracklib.so
password required /lib/security/pam_pwdb.so shadow use_authtok nullok
session required /lib/security/pam_pwdb.so
session optional /lib/security/pam_xauth.so
```

这两行的意思是只有“wheel”组的成员才能用 su 命令成为 root。注意，“wheel”组是系统中用于这个目的的特殊帐号。不能用别的组名。把这节介绍的方法和《22.“/etc/securetty”文件》中介绍的方法结合起来，可以更好地加强系统的安全性。

第二步

我们在“/etc/pam.d/su”配置文件中定义了“wheel”组，现在介绍一下怎样让一些用户可以用“su”命令成为“root”。下面是一个例子，让 admin 用户成为“wheel”组的成员，这样就可以用“su”命令成为“root”：

```
[root@deep]# usermod -G10 admin
```

“G”是表示用户所在的其它组。“10”是“wheel”组的 ID 值，“admin”是我们加到“wheel”组的用户。用同样的命令可以让其他的用户可以用 su 命令成为 root。

25. 资源限制

限制用户对系统资源的使用，可以避免拒绝服务（如：创建很多进程、消耗系统的内存，等等）这种攻击方式。这些限制必须在用户登录之前设定。例如，可以用下面的方法对系统中用户加以。

第一步

编辑“limits.conf”文件（vi /etc/security/limits.conf），加入或改变下面这些行：

```
* hard core 0
* hard rss 5000
* hard nproc 20
```

这些行的的意思是：“core 0”表示禁止创建 core 文件；“nproc 20”把最多进程数限制到 20；“rss 5000”表示除了 root 之外，其他用户都最多只能用 5M 内存。上面这些都只对登录到系统中的用户有效。通过上面这些限制，就能更好地控制系统中的用户对进程、core 文件和内存的使用情况。星号“*”表示的是所有登录到系统中的用户。

第二步

必须编辑“/etc/pam.d/login”文件，在文件末尾加入下面这一行：

```
session required /lib/security/pam_limits.so
```

加入这一行后“/etc/pam.d/login”文件是这样的：

```
#%PAM-1.0
auth required /lib/security/pam_securetty.so
auth required /lib/security/pam_pwdb.so shadow nullok
auth required /lib/security/pam_nologin.so
account required /lib/security/pam_pwdb.so
```

```
password required /lib/security/pam_cracklib.so
password required /lib/security/pam_pwdb.so nullok use_authtok md5 shadow
session required /lib/security/pam_pwdb.so
session required /lib/security/pam_limits.so
#session optional /lib/security/pam_console.so
```

26. 更好地控制 mount 上的文件系统

可以用一些选项，如：noexec、nodev 和 nosuid，更好地控制 mount 上的文件系统，如：“/home”和“/tmp”。这些都在“/etc/fstab”文件中设定。fstab 文件包含了各个文件系统的描述信息。如果想知道在这个文件中可以设定哪些选项，请用 man 命令查看关于 mount 的帮助。

编辑 fstab 文件（vi /etc/fstab），并根据需要把这两行：

```
/dev/sda11 /tmp ext2 defaults 1 2
/dev/sda6 /home ext2 defaults 1 2
```

改变成:

```
/dev/sda11 /tmp ext2 nosuid,nodev,noexec 1 2
/dev/sda6 /home ext2 nosuid,nodev 1 2
```

“nodev”表示不允许在这个文件系统上有字符或特殊的块设备。“nosuid”表示不允许设定文件的 suid（set-user-identifier）和 sgid（set-group-identifier）许可位。“noexec”表示不允许文件系统上有任何可执行的二进制文件。

注意：上面的例子中，“/dev/sda11”mount 到“/tmp”目录上，而“/dev/sd6”mount 到“/home”目录上。当然这和你的实际情况会有所不同，这些取决于你是怎么分区的以及用什么样的硬盘，例如：IDE 硬盘是 hda、hdb，等等，而 SCSI 硬盘是 sda、sdb，等等。

27. 把 rpm 程序转移到一个安全的地方，并改变默认的访问许可

一旦在 Linux 服务器上用 rpm 命令安装完所有需要的软件，最好把 rpm 程序转移到一个安全的地方，如：软盘或其它你认为安全的地方。因为如果有人入侵了你的服务器，他就不能用 rpm 命令安装那些有害的软件。当然，如果将来要用 rpm 安装新的软件，你就要把 rpm 程序拷回原来的目录。

- 把 rpm 程序移到软盘上，用下面的命令：

```
[root@deep]# mount /dev/fd0 /mnt/floppy/
[root@deep]# mv /bin/rpm /mnt/floppy/
[root@deep]# umount /mnt/floppy
```

注意：千万不要把 rpm 程序从系统中卸载掉，否则以后就不能重新安装它，因为安装 rpm 程序或其它软件包本身就要用 rpm 命令。

还有一点要注意的是，把 rpm 命令的访问许可从默认的 755 改成 700。这样非 root 用户就不能使用 rpm 命令了。特别是考虑到万一在安装完新软件之后忘了把 rpm 程序移到一个安全的地方，这样做就更有必要了。

- 改变“/bin/rpm”默认的访问权限，用下面这个命令：

```
[root@deep]# chmod 700 /bin/rpm
```

28. 登录 shell

为了方便重复输入很长的命令，bash shell 可以在“~/.bash_history”文件（“~/”是家目录，每个用户都是不一样的）中存 500 个曾经输入过的命令。每一个有自己帐号的用户，在自己的家目录中，都会有“.bash_history”文件。可能会有这种情况，用户在不该输入口令的地方输入了口令，而输入的口令会在“.bash_history”文件中保存下来。而且“.bash_history”文件越大这种可能性也越大。

在“/etc/profile”文件中 HISTFILESIZE 和 HISTSIZE 这两行决定了系统中所有用户的“.bash_history”文件可以保存多少命令。我建议把“/etc/profile”文件中的 HISTFILESIZE 和 HISTSIZE 都设成一个比较小的值，如：20。

编辑 profile 文件（vi /etc/profile），把这些行改成：

```
HISTFILESIZE=20
HISTSIZE=20
```

这样每个用户家目录下的“.bash_history”就最多只能存 20 个命令。如果黑客试图在用户的“~/.bash_history”文件中发现一些口令，他就没有什么机会了。

29. “/etc/lilo.conf”文件

LILO 是 Linux 上一个多功能的引导程序。它可以用于多种文件系统，也可以从软盘或硬盘上引导 Linux 并装入内核，还可以做为其它操作系统的“引导管理器”。

根（/）文件系统对 LILO 来说很重要，有下面这两个原因：第一：LILO 要告诉内核到哪里去找根文件系统；第二：LILO 要用到的一些东西，如：引导扇区、“/boot”目录和内核就存放在根文件系统中。引导扇区包括 LILO 引导程序的第一部分，这个部分在引导阶段的后半部分还要装入更大的引导程序。这两个引导程序通常存在“/boot/boot.b”文件中。内核是由引导程序装入并启动的。在 RedHat Linux 系统中，内核通常在根目录或“/boot”目录下。

因为 LILO 对 Linux 系统非常重要，所以我们要尽可能地保护好它。LILO 最重要的配置文件是“/etc”目录下的“lilo.conf”文件。用这个文件我们可以配置或提高 LILO 程序以及 Linux 系统的安全性。下面是 LILO 程序的三个重要的选项设置。

- 加入：timeout=00

这项设置设定 LILO 在引导默认的系统之前，等候用户输入的时间。C2 安全等级规定这个时间间隔必须设成 0，因为多重引导会使系统的安全措施形同虚设。除非想用多重引导，否则最好把这项设成 0。

- 加入：restricted

当 LILO 引导的时候，输入参数 linux single，进入单用户（single）模式。因为单用户模式没有口令验证，所以可以在 LILO 引导时，加上口令保护。“restricted”选项只能和“password”合起来用。注意要给每个内核都要加上口令保护。

- 加入：password=<password>

用单用户模式启动 Linux 系统的时候，系统要求用户输入这个口令。口令是大小写敏感的，而且要注意，要让“/etc/lilo.conf”文件，除了 root 之外，其他用户没有读的权限，这样也就看不到口令了。下面是用“lilo.conf”文件保护 LILO 的一个具体例子。

第一步

编辑 lilo.conf 文件（vi /etc/lilo.conf），加上或改变下面介绍的三个设置：

```
boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=00 ← change this line to 00.
Default=linux
restricted ← add this line.
password=<password> ← add this line and put your password.
image=/boot/vmlinuz-2.2.12-20
label=linux
initrd=/boot/initrd-2.2.12-10.img
root=/dev/sda6
read-only
```

第二步

因为“/etc/lilo.conf”配置文件里，存在没有经过加密的口令，所以只有 root 才能有读的权限。用下面的命令改变文件的权限：

```
[root@deep]# chmod 600 /etc/lilo.conf (will be no longer world readable).
```

第三步

使改变后的“/etc/lilo.conf”配置文件生效：

```
[root@deep]# /sbin/lilo -v (to update the lilo.conf file).
```

第四步

为了更安全一点，可以用 chattr 命令给“lilo.conf”文件加上不可改变的权限。

- 让文件不可改变用下面的命令：

```
[root@deep]# chattr +i /etc/lilo.conf
```

这样可以避免“lilo.conf”文件因为意外或其它原因而被改变。如果想要改变“lilo.conf”文件，必须先清除它的不可改变标志。

- 清除不可改变的标记用下面的命令：

```
[root@deep]# chattr -i /etc/lilo.conf
```

30. 使 Control-Alt-Delete 关机键无效

把“/etc/inittab”文件中的一行注释掉可以禁止用 Control-Alt-Delete 关闭计算机。如果服务器不是放在一个安全的地方，这非常重要。

编辑 inittab 文件（vi /etc/inittab）把这一行：

```
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

改为：

```
#ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

用下面的命令使改变生效：

```
[root@deep]# /sbin/init q
```

31. 创建所有重要的日志文件的硬拷贝

保证在“/var/log”目录下的不同日志文件的完整性是保证系统安全所要考虑的非常重要的一个方面。如果我们在服务器上已经加上了很多安全措施，黑客还是能够成功入侵，那么日志文件就是我们最后的防范措施。因此，很有必要考虑一下用什么方法才能保证日志文件的完整性。如果服务器上或网络中的其它服务器上已经安装了打印机，就可以把重要的日志文件打印出来。这要求有一个可以连续打印的打印机，并用 syslog 把所有重要的日志文件传到“/dev/lp0”（打印设备）。黑客可以改变服务器上的文件、程序，等等，但是，把重要的日志文件打印出来之后，他就无能为力了。

例如：

记录下服务器上所有的 telnet、mail、引导信息和 ssh 连接，并打印到连接在这台服务器上的打印机。需要在“/etc/syslog.conf”文件中加入一行。

编辑 syslog.conf 文件（vi /etc/syslog.conf），在文件末尾加入下面这一行：

```
authpriv.*;mail.*;local7.*;auth.*;daemon.info /dev/lp0
```

重新启动 syslog daemon 使改动生效：

```
[root@deep]# /etc/rc.d/init.d/syslog restart
```

又例如：

记录下服务器上所有的 telnet、mail、引导信息和 ssh 连接，并打印到本地网络中其它服务器上连接的打印机，要在这台接收日志文件的服务器的“/etc/syslog.conf”文件中加入一行。如果本地网中没有打印机，可以把所有的日志文件拷贝到别的服务器上，只要忽略下面第一步，把“/dev/lp0”加到其它服务器的“syslog.conf”文件中，直接跳到在其它服务器上设置“-r”参数那一步。把所有日志文件拷贝到其它计算机上，使你可以在一台计算机上管理多台计算机的日志文件，从而简化管理工作。

编辑接收日志文件的服务器（例如：mail.openarch.com）上的“syslog.conf”文件（vi /etc/syslog.conf），在文件的末尾加入下面这一行：

```
authpriv.*;mail.*;local7.*;auth.*;daemon.info /dev/lp0
```

因为 syslog daemon 的默认配置是拒绝接收来自网络上的信息，我们必须使它能够接收来自网络上的信息，在 syslog daemon 的脚本文件（指的是接收日志文件的服务器上的脚本文件）中加入下面的“-r”参数。

编辑 syslog 脚本文件（vi +24 /etc/rc.d/init.d/syslog），把这一行：

```
daemon syslogd -m 0
```

改为：

```
daemon syslogd -r -m 0
```

重新启动 syslog daemon 使改动生效：

```
[root@mail]# /etc/rc.d/init.d/syslog restart
```

如果接收日志文件的服务器上有防火墙，你可以检查一下防火墙的脚本文件中有没有下面几行（没有就加上）：

```
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $SYSLOG_CLIENT \
-d $IPADDR 514 -j ACCEPT
```

在这个例子中防火墙的脚本文件中定义了 EXTERNAL_INTERFACE="eth0"。

```
IPADDR="208.164.186.2";
SYSLOG_CLIENT=”208.164.168.0/24"
```

重新启动接收日志文件的服务器上的防火墙，使改动生效：

```
[root@mail]# /etc/rc.d/init.d/firewall restart
```

这个防火墙规则允许接收日志文件的服务器接收来自端口 514（syslog 的端口）的 UDP 包。关于防火墙的资料可以查看《第七章 网络防火墙》。

最后，编辑一下发送日志文件的服务器上的“syslog.conf”文件（vi /etc/syslog.conf），在末尾加上这一行：

```
authpriv.*;mail.*;local7.*;auth.*;daemon.info @mail
```

“mail”是接收日志文件的计算机主机名。如果有人试图黑你的计算机并且威胁把所有重要的系统日志文件都删掉，你就不用怕了，因为你已经打印出来或者在别

的地方还有一个拷贝。这样就可以根据这些日志文件分析出黑客在什么地方，然后出理这次入侵事件。

重新启动 syslog daemon，使改变生效：

```
[root@deep]# /etc/rc.d/init.d/syslog restart
```

同样还要看看发送日志文件的服务器的防火墙的脚本文件中有没有这几行（没有加上）。

```
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR 514 \
-d $SYSLOG_SERVER 514 -j ACCEPT
```

这里防火墙的脚本文件中定义了：

```
EXTERNAL_INTERFACE="eth0"
IPADDR="208.164.186.1"
SYSLOG_SERVER="mail.openarch.com"
```

重新启动防火墙，使改变生效：

```
[root@deep]# /etc/rc.d/init.d/firewall restart
```

这个防火墙的规则允许发送日志文件的服务器通过端口 514（syslogd 端口）发送 UDP 包。关于防火墙的资料可以查看《第七章 网络防火墙》。

注意：千万不要用网关服务器来收集和管理所有的系统日志信息。有关 syslogd 程序的其它一些参数和策略，可以用 man 命令查看帮助：syslogd(8)、syslog(2)和 syslog.conf(5)。

32. 改变“/etc/rc.d/init.d/”目录下的脚本文件的访问许可

改变启动和停止 daemon 的脚本文件的权限。

```
[root@deep]# chmod -R 700 /etc/rc.d/init.d/*
```

这样只有 root 可以读、写和执行这个目录下的脚本。我想一般用户没有什么必要知道脚本文件的内容。

注意：如果你安装或升级了一个程序，要用到“/etc/rc.d/init.d/”中 system V 脚本，不要忘记再检查一下改变和检查这个脚本文件的许可。

33. “/etc/rc.d/rc.local”文件

在默认情况下，当登录装有 Linux 系统的计算机时，系统会告诉你 Linux 发行版的名字、版本号、内核版本和服务器名称。这泄露了太多的系统信息。最好只显示一个“Login:”的提示信息。

第一步

编辑“/ect/rc.d/rc.local”文件，在下面这些行的前面加上“#”：

```
--
# This will overwrite /etc/issue at every boot. So, make any changes you
# want to make to /etc/issue here or you will lose them when you reboot.
#echo "" > /etc/issue
#echo "$R" >> /etc/issue
#echo "Kernel $(uname -r) on $a $(uname -m)" >> /etc/issue
#
#cp -f /etc/issue /etc/issue.net
#echo >> /etc/issue
--
```

第二步

删除“/etc”目录下的“issue.net”和“issue”文件：

```
[root@deep]# rm -f /etc/issue
[root@deep]# rm -f /etc/issue.net
```

注意：“/etc/issue.net”文件是用户从网络登录计算机时（例如：telnet、SSH），看到的登录提示。同样在“/etc”目录下还有一个“issue”文件，是用户从本地登录时看到的提示。这两个文件都是文本文件，可以根据需要改变。但是，如果想删掉这两个文件，必须向上面介绍的那样把“/etc/rc.d/rc.local”脚本中的那些行注释掉，否则每次重新启动的时候，系统又会重新创建这两个文件。

34. 带“s”位的程序

用 ls -l 命令列出来的文件，如果文件的权限位中出现“s”，则这些文件的 SUID（-rwsr-xr-x）或 SGID（-r-xr-sr-x）位被设定了。因为这些程序给执行它的用户一些特权，所以如果不需要用到这些特权，最好把这些程序的“s”位移去。可以用下面这个命令“chmod a –s <文件名>”移去相应文件的“s”位。

可以清除“s”位的程序包括但不限于：

- 从来不用的程序
- 不希望非 root 用户运行的程序
- 偶尔用用，但是不介意先用 su 命令变为 root 后再运行。

下面加了星号（*）的程序，我个人认为有必要移去“s”位。注意，系统可能需要一些 SUID 的程序才能正常运行，所以要千万小心。

用下面的命令查找所有带“s”位的程序：

```
[root@deep]# find / -type f \( -perm -04000 -o -perm -02000 \) \-exec ls –lg {} \;
*-rwsr-xr-x 1 root root 35168 Sep 22 23:35 /usr/bin/chage
*-rwsr-xr-x 1 root root 36756 Sep 22 23:35 /usr/bin/gpasswd
*-r-xr-sr-x 1 root tty 6788 Sep 6 18:17 /usr/bin/wall
-rwsr-xr-x 1 root root 33152 Aug 16 16:35 /usr/bin/at
-rwxr-sr-x 1 root man 34656 Sep 13 20:26 /usr/bin/man
-r-s--x--x 1 root root 22312 Sep 25 11:52 /usr/bin/passwd
-rws--x--x 2 root root 518140 Aug 30 23:12 /usr/bin/suidperl
-rws--x--x 2 root root 518140 Aug 30 23:12 /usr/bin/sperl5.00503
-rwxr-sr-x 1 root slocate 24744 Sep 20 10:29 /usr/bin/slocate
*-rws--x--x 1 root root 14024 Sep 9 01:01 /usr/bin/chfn
*-rws--x--x 1 root root 13768 Sep 9 01:01 /usr/bin/chsh
*-rws--x--x 1 root root 5576 Sep 9 01:01 /usr/bin/newgrp
*-rwxr-sr-x 1 root tty 8328 Sep 9 01:01 /usr/bin/write
-rwsr-xr-x 1 root root 21816 Sep 10 16:03 /usr/bin/crontab
*-rwsr-xr-x 1 root root 5896 Nov 23 21:59 /usr/sbin/usernetctl
*-rwsr-xr-x 1 root bin 16488 Jul 2 10:21 /usr/sbin/traceroute
-rwxr-sr-x 1 root utmp 6096 Sep 13 20:11 /usr/sbin/utempter
-rwsr-xr-x 1 root root 14124 Aug 17 22:31 /bin/su
*-rwsr-xr-x 1 root root 53620 Sep 13 20:26 /bin/mount
*-rwsr-xr-x 1 root root 26700 Sep 13 20:26 /bin/umount
*-rwsr-xr-x 1 root root 18228 Sep 10 16:04 /bin/ping
*-rwxr-sr-x 1 root root 3860 Nov 23 21:59 /sbin/netreport
-r-sr-xr-x 1 root root 26309 Oct 11 20:48 /sbin/pwdb_chkpwd
```

用下面的命令禁止上面选出来的 SUID 的程序：

```
[root@deep]# chmod a-s /usr/bin/chage
[root@deep]# chmod a-s /usr/bin/gpasswd
[root@deep]# chmod a-s /usr/bin/wall
[root@deep]# chmod a-s /usr/bin/chfn
[root@deep]# chmod a-s /usr/bin/chsh
[root@deep]# chmod a-s /usr/bin/newgrp
```

```
[root@deep]# chmod a-s /usr/bin/write
[root@deep]# chmod a-s /usr/sbin/usernetctl
[root@deep]# chmod a-s /usr/sbin/traceroute
[root@deep]# chmod a-s /bin/mount
[root@deep]# chmod a-s /bin/umount
[root@deep]# chmod a-s /bin/ping
[root@deep]# chmod a-s /sbin/netreport
```

如果你想知道这些程序到底有什么用，可以用 man 命令查看帮助。

例如：

```
[root@deep]# man netreport
```

35. 异常和隐含文件

在系统的每个地方都要查看一下有没有异常和隐含文件（点号是起始字符的，用“ls”命令看不到的文件），因为这些文件可能是隐藏的黑客工具或者其它一些信息（口令破解程序、其它系统的口令文件，等等）。在 UNIX 下，一个常用的技术就是用一些特殊的名，如：“…”、“.. ”（点点空格）或“..^G”（点点 control-G），来隐含文件或目录。用“find”程序可以查找到这些隐含文件。

例如：

```
[root@deep]# find / -name ".. " -print -xdev
[root@deep]# find / -name ".*" -print -xdev | cat -v
```

同时也要注意象“.xx”和“.mail”这样的文件名的。（这些文件名看起来都很象正常的文件名）

36. 查找所有 SUID/SGID 位有效的文件

系统中 SUID 和 SGID 文件很有可能成为安全隐患，必须被严密监控。因为这些程序都给执行它的用户一些特权，所以要确保危险的 SUID 程序没有被安装。

黑客常常利用 SUID 程序，故意留下一个 SUID 的程序作为下次进入系统的后门。注意系统中所有的 SUID 和 SGID 的程序，并跟踪它们，这样你就可以尽早发现入侵者。

用下面的命令查找系统中所有的 SUID 和 SGID 程序：

```
[root@deep]# find / -type f \( -perm -04000 -o -perm -02000 \) \-exec ls -lg {}
\;
```

注意：参考《第九章 安全软件》中关于 sXid 的介绍，sXid 可以为你每天自动的完成这项任务，并用 email 报告结果。

37. 查找任何人都有写权限的文件和目录

如果黑客获得并改变了一些系统文件，这些系统文件就会成为安全漏洞。任何人都有写权限的目录也同样有危险，因为黑客可以根据他们的需要自由地添加或删除文件。在正常情况下有些文件是可写的，包括一些“/dev”目录下的文件和符号连接。

在系统中定位任何人都有写权限的文件和目录用下面的命令：

```
[root@deep]# find / -type f \( -perm -2 -o -perm -20 \) -exec ls -lg {} \;
[root@deep]# find / -type d \( -perm -2 -o -perm -20 \) -exec ls -ldg {} \;
```

注意：象 Tripware 软件这样的文件和目录完整性的检查器能够扫描、管理和方便地发现被改变过的任何人都有写权限的文件。参考《第九章 安全软件》以获得关于 Tripware 的更多信息。

38. 查找没有主人的文件

发现没有主人的文件就意味着有黑客入侵你的系统了。不能允许没有主人的文件存在。如果在系统中发现了没有主人的文件或目录，先查看它的完整性，如果一切正常，给它一个主人。有时候卸载程序可能会出现一些没有主人的文件或目录，在这种情况下可以把这些文件和目录删除掉。

定位系统中没有主人的文件用下面的命令：

```
[root@deep]# find / -nouser -o -nogroup
```

注意：不用管“/dev”目录下的那些文件。

39. 查找“.rhosts”文件

查找“.rhosts”文件是日常管理工作的一部分，因为这些文件不允许在系统中存在。记住，黑客有可能只要有系统中的一个帐号就可能入侵整个网络。

可以用下面的命令定位系统中的“.rhosts”文件：

```
[root@deep]# find /home -name .rhosts
```

也可以用一个 cron 任务定期地查看、报告和删除$HOME/.rhosts 文件。同时，也必须让用户知道你会经常地进行这种审核。

用 root 身份在“/etc/cron.daily”目录下，创建“find_rhosts_files”脚本文件（touch /etc/cron.daily/find_rhosts_files），并在脚本文件中加入下面几行：

```
#!/bin/sh
/usr/bin/find /home -name .rhosts | (cat <<EOF
This is an automated report of possible existent “.rhosts” files on the server
deep.openarch.com, generated by the find utility command.
New detected “.rhosts” files under the “/home” directory include:
EOF
cat
) | /bin/mail -s "Content of .rhosts file audit report" root
```

然后，让这个脚本可执行，把所有者和组设置成“root”：

```
[root@deep]# chmod 755 /etc/cron.daily/find_rhosts_files
[root@deep]# chown 0.0 /etc/cron.daily/find_rhosts_files
```

每一天都会有一份主题为“Content of .rhosts file audit report”的邮件发给“root”，报告新发现的“.rhosts”文件。

40. 系统已经被黑客控制

如果你确信系统已经被黑客控制，赶快联系 CERT® Coordination Center 或 FIRST（Forum of Incident Response and Security Teams）。

电子邮件：cert@cert.org

CERT 热线电话：(+1) 412-268-7090

传真：(+1) 412-268-6989

CERT/CC 的人员在工作日回答问题的时间是：8:00 a.m. – 8:00 p.m. EST (GMT –5)/EDT (GMT –4))。在其它时间或者节假日他们也接收紧急求救电话。

第四章

系统优化概要

Linux 的优化

概述

优化网络性能在很大程度上与网络上使用的软硬件相关。如何优化网络是很难用一本书说得清楚的。在网络真正运行起来之前是很难知道网络的瓶颈所在。性能优化并不是很简单和直观的，必须当作一个很复杂的任务。而且，性能优化不仅受到很多约束还需要很高的精确度。除非进行专门的测试以诊断系统中的瓶颈，否则对一些现象很难做出解释。有时，性能优化会变成一项让人十分失望并且乏味的工作，尤其是在经过大量的分析之后所得到的结果仍然不可确定的时候。但是，对系统性能的优化是一项很有回报的工作，并且会给整个系统带来长期的益处。

1. “/etc/profile”文件

“etc/profile”文件含有系统大量的环境和启动程序的配置信息。你在该文件中进行的配置，可以通过申请全局环境变量来实现。因此，在该文件中设置优化标志，是一种明智的选择。要想使 x86 程序获得最佳性能，可以在编译时，使用最佳的优化选项-O9 。许多程序的“Makefile”文件中已经含有-O2 选项，而-O9 使编译器采用最高级别的优化。尽管它将增加最终程序的大小，但这样可以获得更高的运行速度。编译时，使用“-fomit-frame-pointer”选项，程序运行时，访问变量时将使用堆栈。但是，使用这一选项，生产的代码将无法调试。使用“-mcpu=cpu_type”和“-march=cpu_type”选项，Gcc 将针对这种型号 CPU 进行专门的优化，但生成的代码只能在所指定的 CPU 或更高系列的 CPU 上运行。

对于 CPU i686 或 PentiumPro、Pentium II、Pentium III

在“/etc/profile”文件中 , 加入一行：

```
CFLAGS='-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-
frame-pointer -fno-exceptions'
```

对于 CPU i586 或 Pentium

在“etc/profile”文件中，加入一行：

```
CFLAGS='-O3 -march=pentium -mcpu=pentium -ffast-math -funroll-loops
-fomit-frame-pointer -fforce-
mem -fforce-addr -malign-double -fno-exceptions'
```

对于 CPU　i486

在“etc/profile”文件中，加入一行：

```
CFLAGS='-O3 -funroll-all-loops -malign-double -mcpu=i486 -march=i486
-fomit-frame-pointer -fno-exceptions'
```

在进行完以上设置之后，紧接者着把“CFLAGS LANG LESSCHARSET”加入到“etc/profile”文件中的“export”行中：

```
export PATH PS1 HOSTNAME HISTSIZE HISTFILESIZE USER LOGNAME MAIL INPUTRC CFLAGS
LANG LESSCHARSET
```

然后，重新登录，这时，环境变量 CFLAGS 已经被赋值，编译器和其它配置工具可以使用它。对 Pentium（Pro/II/III）的优化必须使用 egcs 或 pgcc 编译器。Linux 的缺省安装中，已经装上了 egcs，所以无需担心。

基准测试结果　按体系结构分类　：

由于 CPU 的体系结构和使用的 gcc/egcs 的版本不同，优化选项也会不同。下面的图表可以帮助你根据自己的 CPU 和编译器，选择最佳的编译选项。

Redhat 6.1 中安装的编译器的版本是 egcs 2.91.66，但是，即使你安装的就是 Redhat 6.1，在选择编译选项之前也务必检查一下编译器的版本.

为了确认编译器的版本，使用如下命令：

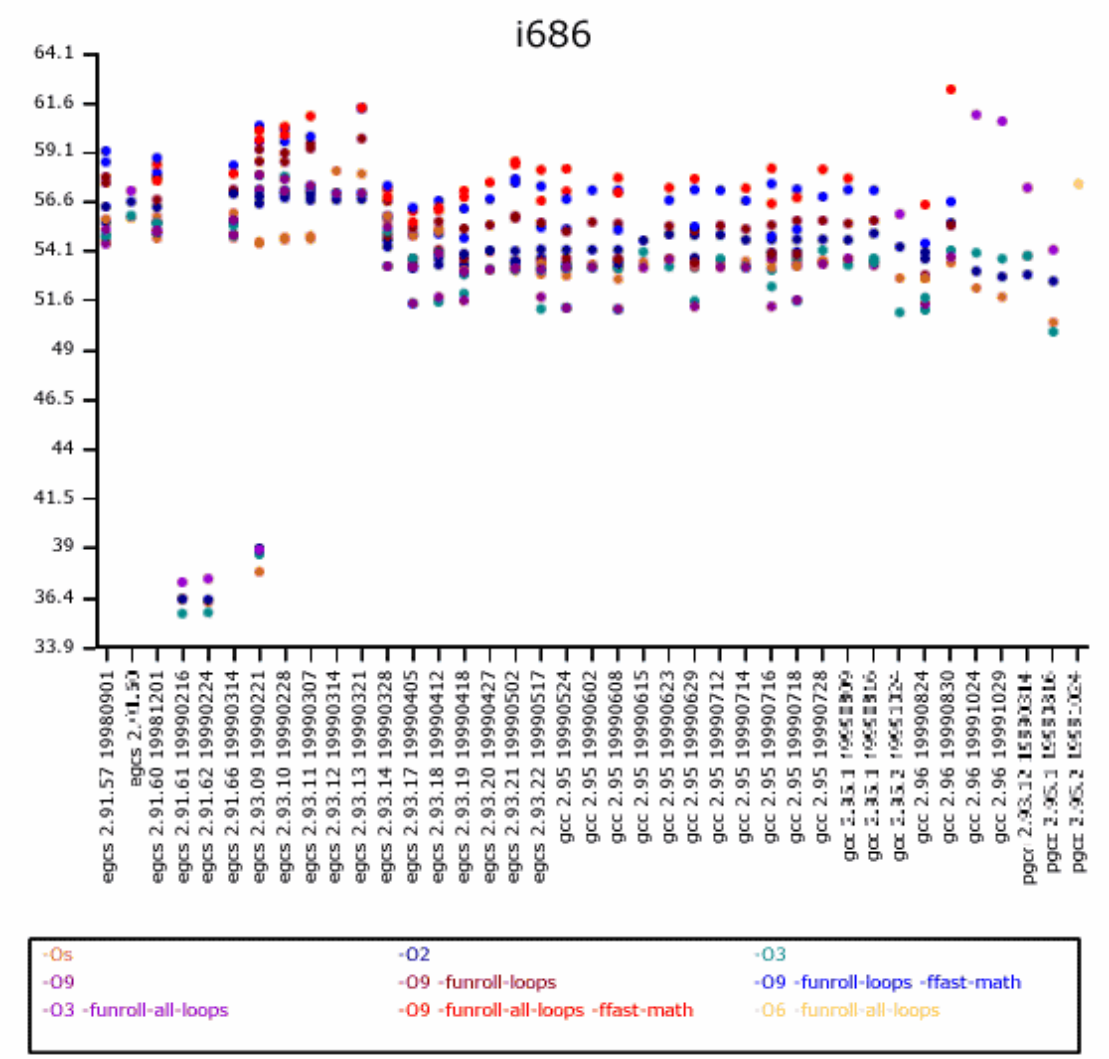
[root@deep]# egcs --version。

注意：所有的测试结果可以从 GCC 的主页：http://egcs.cygnus.com/上检索到。

现举例说明：

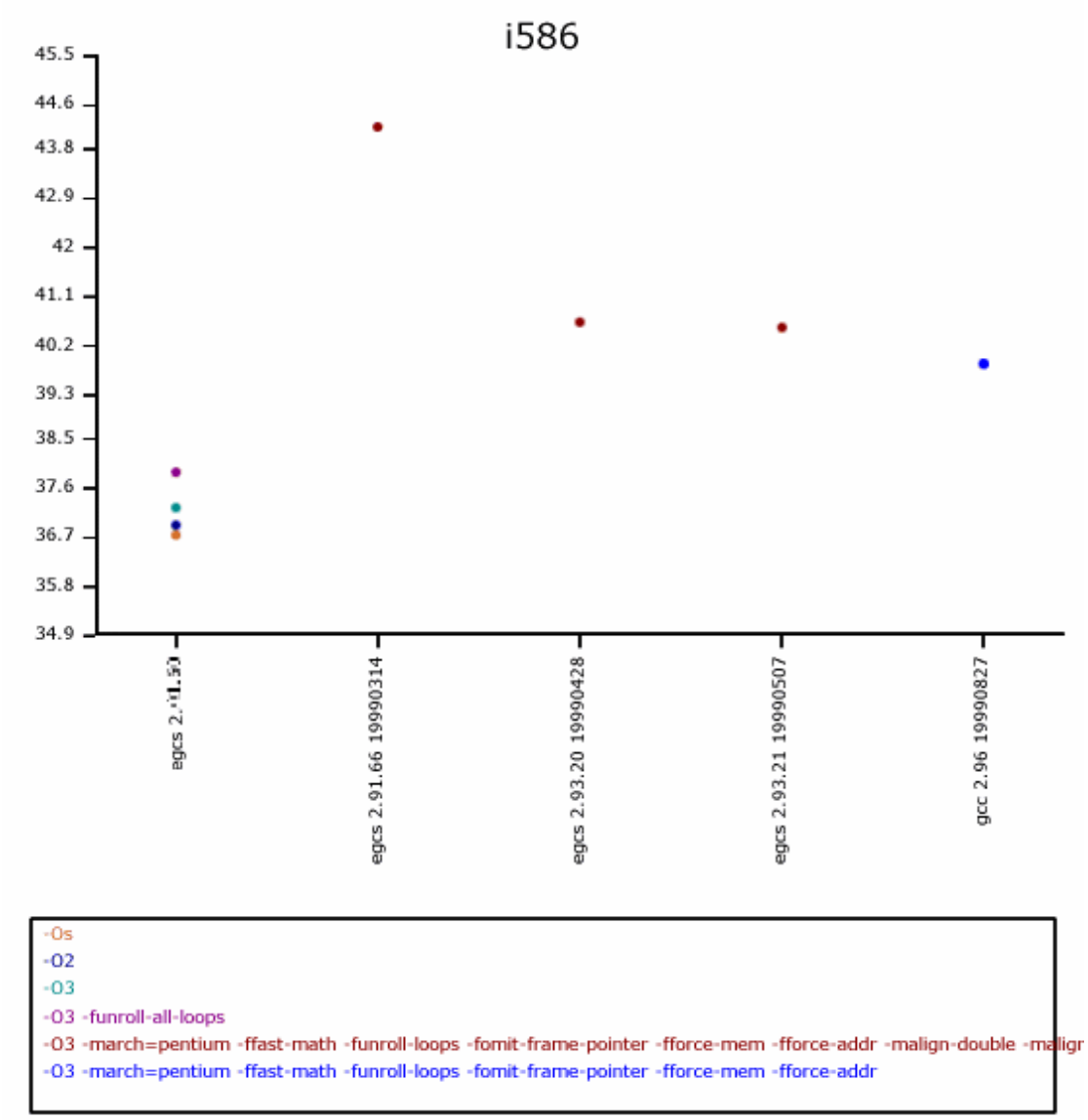
对于 CPU pentium II/III（i686），安装了 egcs-2.91.66，最佳的编译选项是：

```
CFLAGS='-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-pointer -fno-exceptions'
```



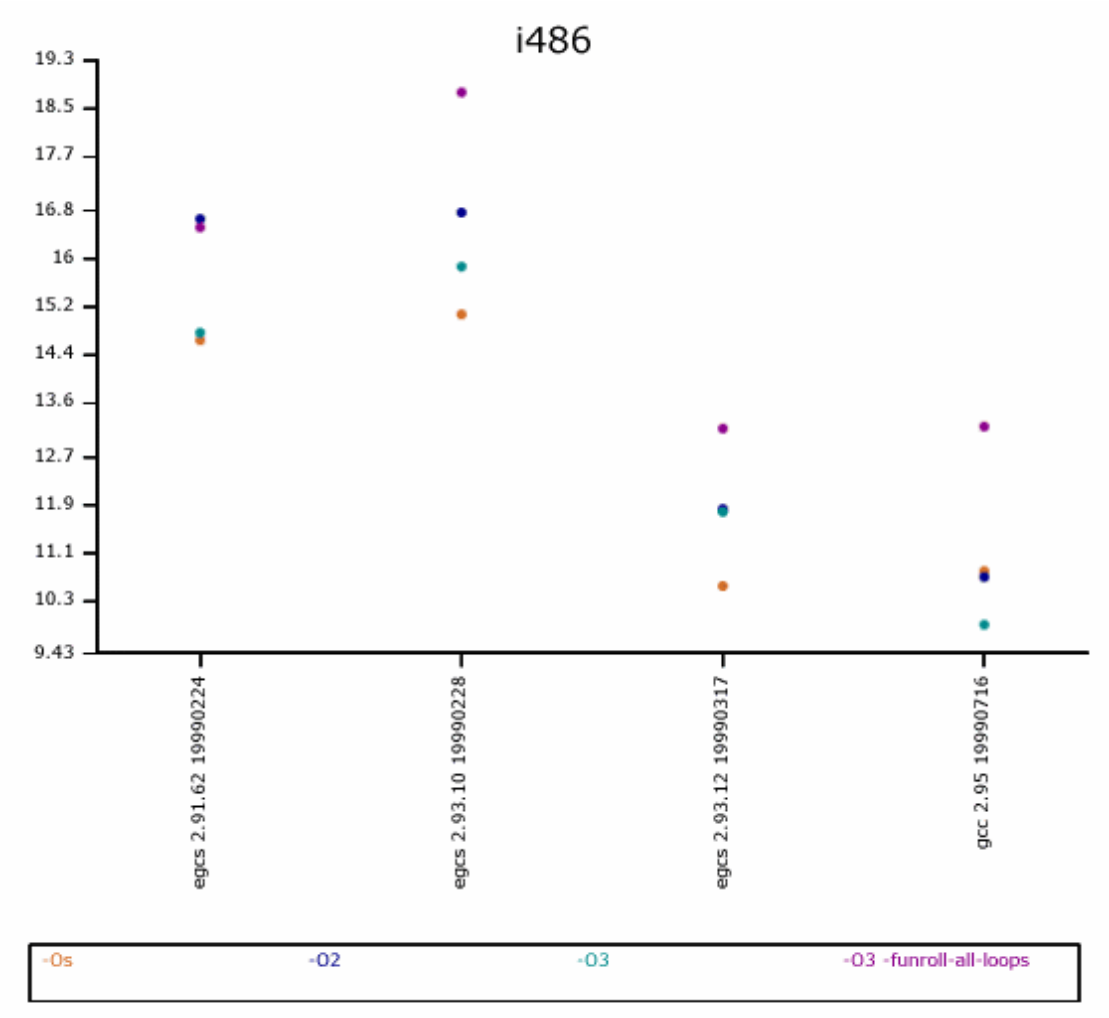
对于 CPU pentium (i586)，安装了 egcs-2.91.66， 最佳的编译选项是：

```
CFLAGS='-O3 -march=pentium -mcpu=pentium -ffast-math -funroll-loops
-fomit-frame-pointer -fforce-mem -fforce-addr -malign-double -fno-exceptions'
```



对于 CPU i486，安装了 egcs-2.91.66，最佳的编译选项是：

```
CFLAGS='-O3 -funroll-all-loops -malign-double -mcpu=i486 -march=i486
-fomit-frame-pointer -fno-exceptions'
```



-funroll-loops 选项

对“loop unrolling”进行优化，只对在编译或运行时循环次数能确定的循环语句有效。

-funroll-all-loops

对“loop unrolling”进行优化，对所有的循环语句有效，但通常使程序运行变慢。

-ffast-math

该选项使 GCC 可以不遵从 ANSI 或 IEEE 的规则，以获得运行更快的优化代码。例如：它允许编译器假设 sqrt()函数的

输入参数非负以及所有的浮点数的值都是 NaNs。

-malign-double

GCC 把 double、long double、and long long 类型变量定界在双字还是单字边界上，由该选项控制。

double 类型变量定界于双字边界时，产生的代码在 Pentium 机器上可以运行得更快一些，但是会占用更多的内存。

-mcpu=cpu_type

设定在生成指令时缺省的机器和 CPU 类型，设定好某一特定的 CPU 类型后，编译器将针对这种芯片产生相应的指令，如果不使用“-march=cpu_type”选项，编译器不会产生任何不能在 i386 上运行的代码。“i586”等价于“pentium”，“i686”等价于“pentiumpro”，“k6”指明是使用 AMD 的芯片而非 Intel 系列。

-march=cpu_type

为指定类型的机器和 CPU 产生指令。这里的 CPU 类型与“-mcpu”中列出的相同。而且，使用本选项已经隐含了“-mcpu=cpu_type”选项。

-fforce-mem

对于涉及内存操作的运算，强制把操作数拷贝到寄存器中。这是通过把所有的内存引用转换成潜在的普通子表达式，以获得优化代码。如果，这些内存引用不是普通子表达式，可以通过指令的组合，消除单独的寄存器装载。

-fforce-addr

运算前把内存地址常数拷贝到寄存器中。所产生的优化代码与选项“-fforce-mem”类似。

-fomit-frame-pointer

对于不必要的框架指针（frame pointer），不在寄存器中保存。这就避免了相应的用于保存、设置和恢复框架指针所需的指令；这样，许多函数中可以使用额外的寄存器。但是，这一选项使得在大多数机器上无法进行调试。

注意：本书将要讨论的所有优化，缺省都是针对 Pentium II/III 系列 CPU。因此，如有必要，对于某些专门的 CPU 需要调整编译参数。

2. “bdflush”参数

下文讨论目录“/proc/sys/vm”下的系统控制文件，且只在 Linux 内核版本 2.2 下有效。控制该目录下的文件，可以调整 Linux 内核

子系统——虚拟内存（VM）的行为，其中 bdflush 文件对于硬盘使用有一定影响。

该文件控制了 bdflush 内核守护进程的行为。我们通常使用以下命令来提高文件系统的性能：

```
echo "100 1200 128 512 15 5000 500 1884 2">/proc/sys/vm/bdflush
```

修改某些值，可以使系统响应更快，例如：可以使系统在写入硬盘之前等待更长时间，从而避免了一些硬盘访问的冲突。

把该命令加入文件“etc/rc.d/rc.local”之中，每次重新启动机器时，就不必再次手工敲入这条命令了。

如果需要进一步理解如何改进有关虚拟内存、硬盘缓冲和交换空间（swap）的内核参数，可以参照“/usr/src/linux/Documentation/sysctl/vm.txt”。

3. “ip_local_port_range”参数

下文讨论目录“/proc/sys/net/ipv4/ip_local_port_range”下的系统控制文件，且只在 Linux 内核版本 2.2 下有效。

“ip_local_port_range”文件中有两个参数分别定义了用作 TCP 和 UDP 本地端口的端口范围。第一个参数是第一个端口号。第二个参数是最后一个本地端口号。对于使用率很高的系统，可以修改为：32768 到 61000。

```
echo *32768 61000* > /proc/sys/net/ipv4/ip_local_port_range
```

把该命令加入文件“/etc/rc.d/rc.local”之中，每次重新启动机器时，就不必再次手工敲入这条命令了。

4. “/etc/nsswitch.conf”文件

“/etc/nsswitch.conf”文件定义了系统使用哪些服务来解析主机名、获得口令文件和组文件（group file）。我们的系统中由于没有使用 NIS 服务，因此口令文件和组文件我们没有使用。这里，我们只讨论该文件中的 hosts 这一行。

编辑“nsswitch.conf”文件（vi /etc/nsswitch.conf），把 host 一行改为：

```
"hosts: dns files"
```

含义：当请求解析地址时，首先访问 DNS 服务器，如果 DNS 服务器没有响应，则使用“/etc/hosts”文件。

我建议把该文件中每一行中的 NIS 都删掉。当然，如果你一定要使用 NIS，就不能删掉 NIS。最后，这个文件会是这样：

```
passwd: files
shadow: files
group: files
hosts: dns files
bootparams: files
ethers: files
netmasks: files
networks: files
protocols: files
rpc: files
services: files
automount: files
aliases: files
```

5. “/proc”文件系统

下文讨论目录“proc/sys/fs”下的系统控制文件，且只在 Linux 内核版本 2.2 下有效。该目录下的文件可以用来调整和监测 Linux 内核的一些行为。对这些文件的误操作可能搅乱系统，因此在实际调整系统之前，最好把文档和源代码都读一下。

适当的增加“/proc/sys/fs/file-max”的值：每 4M 内存对应 256，例如：内存为 128M 的机器，该值可以设为 8192（128/4=32 32*256=8192）。同理，可以增加“/proc/sys/fs/inode-max”的值，使其值为打开文件数目的 3 到 4 倍(8192*4=32768)。这是因为：i 节点的数目至少等于打开的文件数，一般而言，对于大文件，i 节点数远大于打开的文件数目。

用于改变/proc 目录及其子目录下的任意参数的常用命令是(必须以 root 登录)：echo “新的参数值”> “/proc/所需更改的文件”，对于上面所涉及的修改，其命令为：

```
echo "8192" >/proc/sys/fs/file-max
echo "32768" >/proc/sys/fs/inode-max
```

上文所讨论的方法修改了内核源代码的常数。但是，在新的内核源代码树中并不能起作用，因此还不能算是最好的方法。最好的一种方法是把上述命令加入文件“etc/rc.d/rc.local”之中。在该文件的最后加入以下两行（假设系统有 128M 内存）：

```
echo "8192" >/proc/sys/fs/file-max
echo "32768" >/proc/sys/fs/inode-max
```

其中的数值因系统不同，差异很大，应该根据各自系统，按照上述的公式计算。一台文件服务器或 WEB 服务器需要打开的文件数目就很大，而用于数值的服务器该数值就较小。

对于内存非常多的系统，特别是 512M 或更多内存的系统，打开的文件数和 i 节点数最好不要超过 50,000 和 150,000。

“file-max”参数是指 Linux 内核可以分配的文件句柄的最大数目。当系统经常报错：文件句柄不够时，就需要适当增大该参数的值。系统缺省值为：4096。

“inode-max”参数是指系统 i 节点句柄的最大数目。其值应该是 file-max 值的 3 到 4 倍。因为标准输入输出文件和网络套接字

都要使用 i 节点来进行处理。如果系统经常性的出现 i 节点被耗尽的情况，就需要增大其值。

6. “ulimit”参数

Linux 本身对每个用户拥有的最大进程数有限制。可以在用户根目录下的“.bashrc”文件或者实际使用与“.bashrc”功能相当的 shell 的脚本中加入这种限制。编辑“.bashrc”文件（例如：vi /root/.bashrc）并加入下面一行：

```
ulimit -u unlimited
```

然后退出，重新登录。为了验证，可以以 root 身份登录，然后键入：“ulimit –a”，在最大用户进程数一项中应该

显示“unlimited”，例如：

```
[root@deep]# ulimit -a
core file size (blocks) 1000000
data seg size (kbytes) unlimited
file size (blocks) unlimited
max memory size (kbytes) unlimited
stack size (kbytes) 8192
cpu time (seconds) unlimited
max user processes unlimited * this line.
pipe size (512 bytes) 8
open files 1024
virtual memory (kbytes) 2105343
```

注意：你可能更倾向于在命令行键入“unlimit –u”而不是把它加入到文件“/root/.bashrc”中。但为保险起见，建议

还是把它加入文件“/root/.bashrc”中。

7. 增加系统打开的文件数目

增加当前进程打开文件的数目。RedHat 6.0（内核 2.2.5）中，用这种方法进程可以至少打开 31000 个文件描述符；内核版本为 2.2.12 中，可以至少打开 90000 个文件描述符（在适当的限制下）。它的上限仅受限于可用内存。

编辑“.bashrc”文件（例如：vi /root/.bashrc）并加入下面一行：

```
ulimit -n 90000
```

然后退出，重新登录。为了验证，可以以 root 身份登录，然后键入“ulimit –a”，在打开文件数一项中应该显示“90000”，例如：

```
[root@deep]# ulimit -a
core file size (blocks) 1000000
data seg size (kbytes) unlimited
file size (blocks) unlimited
max memory size (kbytes) unlimited
stack size (kbytes) 8192
cpu time (seconds) unlimited
max user processes unlimited
pipe size (512 bytes) 8
open files 90000 * this line.
virtual memory (kbytes) 2105343
```

注意：在早于 2.2 版内核的系统中，即使进行了上述修改，每个进程所能打开的文件数目仍然限制为 1024。

8. 文件“atime”属性

Linux 除了记录文件的创建和修改时间外，还记录文件最近的访问时间。该信息作用不大，但为保留它，系统需要消耗相应的资源。ext2 文件系统允许超级用户对单个文件进行标记，以忽略对这条信息的记录。

这种优化调整，对于文件查找操作，系统性能提高显著，另外，对于需要经常访问的文件（如：/var/spool/news）也是很有用的。设置该属性的命令为：

```
[root@deep]# chattr +A filename
```

若需要对某个目录下的所有文件进行这种设置，可以使用：

```
[root@deep /root]# chattr -R +A /var/spool/
[root@deep /root]# chattr -R +A /cache/
[root@deep /root]# chattr -R +A /home/httpd/ona/
```

9. 文件的“noatime”属性

Linux 在 mount 文件系统时，可以使用“noatime”选项。并可以在文件“/etc/fstab”的 mount 选项区域中加入。当含有该选项的

文件系统被挂入（mount）系统时，对该文件系统中的文件的读访问，不再更新文件的 atime 信息。一般情况下，atime 信息没有用，所以不更新该域并无大碍。这一选项的重要性在于：当只对文件进行读操作时，不再需要向文件系统中该文件的相应区域写入信息。因为写入操作的开销某种意义下是昂贵的，因此该选项可以获得明显的性能的改善。该选项对于文件的 wtime 属性没有影响，每次文件写操作时，都会更新文件的 wtime 信息。

编辑“fstab”文件（vi /etc/fstab）并且加入如下一行（举例说明）：

```
E.I: /dev/sda7 /chroot ext2 defaults,noatime 1 2
```

重新启动系统, 然后用以下命令测试结果：

```
[ root@deep ]# reboot
[ root@deep ]# cat /proc/mounts
```

10. 特定的 TCP/IP 栈

RedHat Linux , 一般情况下，并不优化 TCP/IP 窗口大小。这能使系统性能的差别很大。如需更多的信息，参阅：RFC 1106 - High Latency WAN links - Section 4.1 and RFC 793 - Transmission Control Protocol。

编辑文件“/etc/sysconfig/network-scripts/ifup”，在 110、112、117、125 和 134 行，有：

```
110: "route add -net ${NETWORK} netmask ${NETMASK} ${DEVICE}"
112: "route add -host ${IPADDR} ${DEVICE}"
117: "route add default gw ${GATEWAY} metric 1 ${DEVICE}"
125: "route add default gw ${GATEWAY} ${DEVICE}"
134: "route add default gw $gw ${DEVICE}"
```

修改为：

```
110: "route add -net ${NETWORK} netmask ${NETMASK} window 8192 ${DEVICE}"
112: "route add -host ${IPADDR} window 8192 ${DEVICE}"
117: "route add default gw ${GATEWAY} window 8192 metric 1 ${DEVICE}"
125: "route add default gw ${GATEWAY} window 8192 ${DEVICE}"
134: "route add default gw $gw window 8192 ${DEVICE}"
```

11. 交换分区

尽量把交换分区放在硬盘的开始区域。硬盘的开始区域物理上位于硬盘柱面的外环部分，因此硬盘的每转能在这一部分读写更多的信息。我曾经见过，在运行命令“hdparm –t”时，把交换分区放在硬盘的结尾部分市，系统的读写速度比放在硬盘的开始部分低 3MB/s。

12. 调整 IDE 硬盘性能

在有大量的磁盘 I/O 操作时，设置 IDE 硬盘使用 DMA 、32 位的传送和多重的扇区模式（Multiple sector mode），可以大幅

提高系统性能。除非显式的告诉内核使用这些模式，内核缺省是使用保守设置的。

使用如下命令，以使 PCI 总线允许 32 位 I/O 操作：

```
[root@deep]# /sbin/hdparm -c 1 /dev/hda (或 hdb, hdc 等等)
```

hdparm(8) 的帮助信息（manpage）中说明了：对于某些芯片需要使用选项“-c 3”。所有的(E)IDE 硬盘与接口卡相连的扁平电缆中只用 16-bit 的连接。

使用如下命令，以允许 DMA 方式：

```
[root@deep]# /sbin/hdparm -d 1 /dev/hda (或 hdb, hdc 等等)
```

这取决于编译内核时选择的主板芯片组。

使用如下命令，以允许多字的 DMA 模式 2 传送：

```
[root@deep]# /sbin/hdparm -d 1 -X34 /dev/hda (或 hdb, hdc 等等)
```

这为较新的(E)IDE/ATA2 硬盘设置 IDE 的传送模式。（参阅硬盘手册，以核实你的硬盘是否支持这种模式）。

使用如下命令，以允许 UltraDMA 模式 2：

```
[root@deep]# /sbin/hdparm -d 1 -X66 /dev/hda (或 hdb, hdc 等等)
```

使用该命令之前，事先应该有支持 UltraDMA 的芯片组，并参看 hdparm 的帮助信息。使用该命令时，务必小心。

使用如下命令，以允许多扇区 I/O 模式：

```
[root@deep]# /sbin/hdparm -m XX /dev/hda （或 hdb, hdc 等等）
```

其中，参数 XX 是指使用硬盘支持的最大设置。可以使用“-i”选项来自动查找所装硬盘支持的最大设置。可以在输出中查看 MaxMultSect 的值。例如：

```
[root@deep]# /sbin/hdparm -i /dev/hda （或 hdb, hdc 等等）
```

```
/dev/hda:
Model=Maxtor 7540 AV, FwRev=GA7X4647, SerialNo=L1007YZS
Config={ HardSect NotMFM HdSw>15uSec Fixed DTR>5Mbs FmtGapReq }
RawCHS=1046/16/63, TrkSize=0, SectSize=0, ECCbytes=11
BuffType=3(DualPortCache), BuffSize=32kB, MaxMultSect=8, MultSect=8
DblWordIO=yes, maxPIO=2(fast), DMA=yes, maxDMA=1(medium)
CurCHS=523/32/63, CurSects=379584528, LBA=yes, LBA=yes, LBAsects=1054368
tDMA={min:150,rec:150}, DMA modes: sword0 sword1 *sword2 *mword0
IORDY=on/off, tPIO={min:240,w/IORDY:180}, PIO modes: mode3
```

目前的大多数的硬盘驱动器都支持多扇区模式（Multiple sector mode 或 aka IDE Block Mode），它允许一次中断中传送多个扇区而不是一次中断传送一个扇区。具有这种特性的硬盘驱动器，能使操作系统在硬盘 I/O 时的负载下降 30-50%。在许多系统中，它能提高数据的吞吐率 5-50%。

可以用 hdparm 的测试模式，来测试所作修改的结果：

```
[root@deep]# /sbin/hdparm -t /dev/hda (或 hdb, hdc 等等)
```

进行了以上修改后，不要忘记把它们加入到文件“/etc/rc.d/rc.local”中，以便每次重启系统时，能够自动运行。

第三部分：与内核相关的参考资料

第五章

配置和编译内核

Linux 内核

概述

下面讨论如何根据自己系统的需求重新编译生成新的内核。编译内核是非常简单的工作，一般只要根据“/usr/src/linux/”目录下的 README 文件中的指示就都可以完成。为了正确地合理地设置内核编译配置选项，从而只编译系统需要的功能的代码，一般主要有下面四个考虑：①自己定制编译的内核运行更快（具有更少的代码）；②系统将拥有更多的内存（内核部分将不会被交换到虚拟内存中）；③不需要的功能编译进入内核可能会增加被系统攻击者利用的漏洞；④将某种功能编译为模块方式会比编译到内核内的方式速度要慢一些。

在下面的内核配置和编译示例中，我们将编译生成一单块内核（monolithic kernel）。单块内核指在回答编译配置选项时只回答“yes”或“no”，而不回答“M”，也就是将所有要支持的功能都直接编译到内核中，而不编译为模块方式，故在编译内核过程中可以忽略 make_modules 和 make_modules_install 两个步骤。同样我们这里对编译的内核添加“缓冲区溢出防范”补丁，如： Solar 设计者开发的不可执行性栈补丁（non-executable stack patch）。

若需要使用防火墙的 IP 伪装及 PPP 拨号连接功能，则不能采用下面的介绍的单块内核方式，因为这些功能必须缺省地被编译为模块方式。

一个新的编译生成的内核是与特定计算机硬件相关的，这些都是在内核编译配置中确定的。下面的例子里我们假设计算机有如下的硬件配置：

- 1 Pentium II 400 MHz (i686) 处理器
- 1 SCSI 主板
- 1 SCSI 硬盘
- 1 Adaptec AIC 7xxx SCSI 控制器
- 1 ATAPI IDE CD-ROM

- 1 软驱
- 2 Intel EtherExpressPro 10/100 网卡
- 1 PS/2 鼠标

安装说明如下

命令为 Unix 兼容。

源代码位于“/usr/src”目录下。

安装测试是在 RedHat Linux 6.1 进行的。

所有的步骤都是以 root 用户的身份进行的。

最后的内核版本是 2.2.14。

最后的内核补丁版本号是 2_2_14-ow1。

软件包

内核源代码下载路径 http://www.kernelnotes.org/。

下载 linux-2_2_14_tar.gz。

安全 Linux 核补丁主页为 http://www.openwall.com/linux/。

下载 linux-2_2_14-ow1_tar.gz。

做一张紧急启动盘

安装前的准备工作的第一步是创建一个紧急启动软盘：使用 mkbootdisk 命令来实现创建启动盘。首先查看文件“/etc/lilo.conf”来确定系统当前是使用哪个核启动的。例如：

```
[root@deep]# cat /etc/lilo.conf
boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
```

```
image=/boot/vmlinuz-2.2.12-20
label=linux
root=/dev/sda6
initrd=/boot/initrd-2.2.12-20.img
read-only
```

现在需要查看用来启动的内核，在标准安装中，将是映像标签为 linux 的内核映像。在上面的例子显示该系统是使用“/boot/vmlinuz-2.2.12-20”内核来作为启动映像的。在软驱中插入一张经过格式化的 1.44 英寸软盘，以根用户的身份登录，执行如下命令：

```
[root@deep]# mkbootdisk --device /dev/fd0 2.2.12-20
Insert a disk in /dev/fd0. Any information on the disk will be lost.
Press <Enter> to continue or ^C to abort:
```

按照上面的步骤，将成功创建一个带有可以正常工作的内核映像的启动软盘，以在升级内核出现错误时使用。推荐在进行下一步以前首先使用该软盘引导进入系统成功以后在继续进行下面的步骤。

优化

解压内核源代码：

```
[root@deep]# cp linux-version_tar.gz /usr/src/
[root@deep]# cd /usr/src/
[root@deep]# rm -rf linux (这个一般是一个符号链接)
[root@deep]# rm -rf linux-old.version.number (这是存放系统内核头文件目录)
```

注意：上面删除 Linux 符号链接（rm -rf linux）和 Linux 头文件子目录的做法仅仅适用于以前曾经安装了过 tar 格式的 Linux 内核的情况。若这是第一次安装 Linux kernel 内核，则应该使用删除在系统中的 kernel-headers-version.i386.rpm, kernel-version.i386.rpm 两个 RPM 包的方法，删除这两个 RPM 包的同时将使目录“/usr/src/linux”及目录“/lib/modules/2.2.XX”下的相关的模块文件将自动被去除。

若系统安装的是一个 RPM 格式的内核包，而不是 tar 格式的包，则说明你以前没有更新安装过 Linux 系统或者是使用 RPM 包来更新 Linux 系统。则使用下面的方法来删除 Linux 内核。

可以使用下面的命令来验证你的系统安装了 RPM 内核包：

```
[root@deep]# rpm -qa | grep kernel
kernel-headers-2.2.12-20.i386.rpm
kernel-2.2.12-20.i386.rpm
```

使用下面的命令来删除安装的 RPM 内核包：

```
[root@deep]# rpm -e --nodeps kernel-headers kernel
```

使用“rpm –e”命令以后，再手工删除空的“/usr/src/linux-2.2.12”和“/lib/modules/2.2.12”目录（RPM 反安装程序并不能完全删除这些目录）。然后解开内核源代码，改变新的 Linux 源代码的所有者为根用户，然后删除压缩的源代码。

```
[root@deep]# rm -rf /usr/src/linux-2.2.12/
[root@deep]# rm -rf /lib/modules/2.2.12-20/
[root@deep]# tar xzpf linux-version_tar.gz
[root@deep]# chown -R 0.0 /usr/src/linux/
[root@deep]# rm -f linux-version_tar.gz
```

增加任务数

为了增加允许的任务数（每个用户的最大进程数），需要修改文件“/usr/src/linux/include/linux/tasks.h”中如下所示的内容：

```
NR_TASKS from 512 to 3072
MIN_TASKS_LEFT_FOR_ROOT from 4 to 24
```

注意：

1. NR_TASKS 指定内核将分配给每个用户的最大数量的任务。增加这个数字将允许系统服务器进程处理客户端更多连接请求（例如 WEB 服务器将能处理多个客户端的服务请求）

2. Linux 被设计为避免所有的进程槽被普通用户所占用。必须保留至少 MIN_TASKS_LEFT_FOR_ROOT 个时间槽给根用户。也就实现了避免所有的内存被普通用户所占用。

优化内核

为了优化定制适合系统需要的内核来适应系统的 CPU 类型及优化参数需求，也许需要编辑“/usr/src/linux/Makefile”文件并改变如下的参数部分：

- 修改：

```
CFLAGS = -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer
```

为

```
CFLAGS = -Wall -Wstrict-prototypes -O9 -funroll-loops -ffast-math -malign-double
-mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions
```

- 修改

```
HOSTCFLAGS = -Wall -Wstrict-prototypes -O2 -fomit-frame-pointer
```

为

```
HOSTCFLAGS = -Wall -Wstrict-prototypes -O9 -funroll-loops -ffast-math
-malign-double -mcpu=pentiumpro -march=pentiumpro -fomit-frame-pointer
-fno-exceptions
```

上面的修正是一些比较冒进的优化，这些优化并不一定适用于所有的情况，如果这些优化不适合你的情况，请不要勉强使用这些优化。

增强内核的安全性

来自 Openwall 工程的安全内核补丁是一个非常好安全增强补丁，可以防止如栈溢出等攻击。这个补丁集合了一系列针对内核的安全相关特性，所有的新增加的安全选项配置将增加内核的安全性。

除了新增的特性，某些版本的补丁还包含了各种其他安全弥补措施。这种弥补是随着版本的不同而不同的，某些修补已经消失了（因为新的内核的发布已经弥补了错误），而有些却是针对出现的新的安全漏洞。

linux-2_2_14-ow1_tar.gz 的新的特性包括

- 不可执行的用户堆栈区
- 限制“/tmp”目录下的链接
- 限制“/tmp”下的 FIFO
- 受限制的“/proc”目录
- 特殊文件句柄 0、 1 和 2
- 在 execve(2)中对 RLIMIT_NPROC 进行了加强
- 将不使用的共享内存段释放归还给系统

注意：当施用内核补丁 linux-2_2_14-ow1 补丁时，将在内核配置的最后添加新的安全选项部分。想知道到关于这些新配置选项的意思，可以参见该补丁带的自述文件。

对内核施用补丁如下

```
[root@deep]# cp linux-2_2_14-ow1_tar.gz /usr/src/
[root@deep]# cd /usr/src/
[root@deep]# tar xzpf linux.2_2_14-ow1_tar.gz
[root@deep]# cd linux-2.2.14-ow1/
[root@deep]# mv linux-2.2.14-ow1.diff /usr/src/
[root@deep]# cd ..
[root@deep]# patch -p0 < linux-2.2.14-ow1.diff
[root@deep]# rm -rf linux-2.2.14-ow1
[root@deep]# rm -f linux-2.2.14-ow1.diff
[root@deep]# rm -f linux-2_2_14-ow1_tar.gz
```

注意：和 linux-2.2.14-ow1 相关的所有的安全信息：如非可执行堆栈等将被记录到“/var/log/messages”中。

编译

确保“/usr/include/asm”、“/usr/include/linux”和“/usr/include/scsi”等子目录是指向内核源代码的链接。子目录 asm、linux 和 scsi 都是链向源代码目录下的真正的、该计算机体系结构所需要的真正的 include 子目录。如：asm 指向“/usr/src/linux/include/asm-i386”等。若没有这些链接，就需要手工创建，按照下面的步骤进行：

```
[root@deep]# cd /usr/include/
[root@deep]# rm -rf asm linux scsi
[root@deep]# ln -s /usr/src/linux/include/asm-i386 asm
[root@deep]# ln -s /usr/src/linux/include/linux linux
[root@deep]# ln -s /usr/src/linux/include/scsi scsi
```

这是配置非常重要的一部分。删除掉“/usr/include”下的 asm、linux 和 scsi 目录后，再创建新的链接指向新内核源代码目录下的同名的目录。这些头文件目录包含着保证内核在系统上正确编译所需要的重要的头文件。

使用下面的命令来确保系统没有陈旧的“.o”文件及依赖关系：

```
[root@deep]# cd /usr/src/linux/
[root@deep]# make mrproper
```

注意：上面的两个命令把以前无意留下来的垃圾文件清除掉。

现在已经正确地安装了源代码。可以通过三种方式来配置你的内核：第一种是使用命令“make config”。这个命令提供了一个基于文本的交互式的内核选项配置方式；第二种是使用命令“make menuconfig”，它提供了一个易于使用的菜单式的配置界面；第三种方式是使用命令“make xconfig”，它提供了一个完全的图形化的内核配置界面。

这里我们以“make config”为例来说明，其他两种方式类似：.

```
[root@deep]# cd /usr/src/linux/
[root@deep]# make config
```

内核配置

Code maturity level options

```
Prompt for development and/or incomplete code/drivers (CONFIG_EXPERIMENTAL) [N]
```

Processor type and features

```
Processor family (CONFIG_M386) [Ppro/6x86MX]
Maximum physical Memory (CONFIG_1GB) [1GB]
Math emulation (CONFIG_MATH_EMULATION) [N]
MTRR Memory Type Range Register support (CONFIG_MTRR) [N]
Symmetric multi-processing support (CONFIG_SMP) [Y] N
```

Loadable module support

```
Enable loadable module support (CONFIG_MODULES) [Y] N
```

General setup

```
Networking support (CONFIG_NET) [Y]
PCI support (CONFIG_PCI) [Y]
PCI access mode (BIOS, Direct, Any) [Any]
PCI quirks (CONFIG_PCI_QUIRKS) [Y] N
Backward-compatible /proc/pci (CONFIG_PCI_OLD_PROC) [Y] N
MCA support (CONFIG_MCA) [N]
SGI Visual Workstation support (CONFIG_VISWS) [N]
System V IPC (CONFIG_SYSVIPC) [Y]
```

```
BSD Process Accounting (CONFIG_BSD_PROCESS_ACCT) [N]
Sysctl support (CONFIG_SYSCTL) [Y]
Kernel support for a.out binaries (CONFIG_BINFMT_AOUT) [Y]
Kernel support for ELF binaries (CONFIG_BINFMT_ELF) [Y]
Kernel support for MISC binaries (CONFIG_BINFMT_MISC) [Y]
Parallel port support (CONFIG_PARPORT) [N]
Advanced Power Management BIOS supports (CONFIG_APM) [N]
```

Plug and Play support

```
Plug and Play support (CONFIG_PNP) [N]
```

Block devices

```
Normal PC floppy disk support (CONFIG_BLK_DEV_FD) [Y]
Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support (CONFIG_BLK_DEV_IDE) [Y]
Use old disk-only driver on primary interface (CONFIG_BLK_DEV_HD_IDE) [N]
Include IDE/ATA-2 disk support (CONFIG_BLK_DEV_IDEDISK) [Y]
Include IDE/ATAPI CDROM support (CONFIG_BLK_DEV_IDECD) [Y]
Include IDE/TAPE support (CONFIG_BLK_DEV_IDETAPE) [N]
Include IDE/FLOPPY support (CONFIG_BLK_DEV_IDEFLOPPY) [N]
SCSI emulation support (CONFIG_BLK_DEV_IDESCSI) [N]
CMD640 chipset bugfix/support (CONFIG_BLK_DEV_CMD640) [Y] N
RZ1000 chipset bugfix/support (CONFIG_BLK_DEV_RZ10000 [Y] N
Generic PCI IDE chipset support (CONFIG_BLK_DEV_IDEPCI) [Y]
Generic PCI bus-master DMA support (CONFIG_BLK_DEV_IDEDMA) [Y]
Boot off-board chipsets first support (CONFIG_BLK_DEV_OFFBOARD) [N]
Use DMA by default when available (CONFIG_IDEDMA_AUTO) [Y]
Other IDE chipset support (CONFIG_IDE_CHIPSETS) [N]
Loopback device support (CONFIG_BLK_DEV_LOOP) [N]
Network block device driver support (CONFIG_BLK_DEV_NBD) [N]
Multiple device driver support (CONFIG_BLK_DEV_MD) [N]
RAM disk support (CONFIG_BLK_DEV_RAM) [N]
XT hard disk support (CONFIG_BLK_DEV_XD) [N]
Mylex DAC960/DAC1100 PCI RAID Controller support (CONFIG_BLK_DEV_DAC960) [N]
Parallel port IDE device support (CONFIG_PARIDE) [N]
Compaq SMART2 support (CONFIG_BLK_CPQ_DA) [N]
```

Networking options

```
Packet socket (CONFIG_PACKET) [Y]
Kernel/user netlink socket (CONFIG_NETLINK) [N]
Network firewalls (CONFIG_FIREFALL) [N] Y
Socket filtering (CONFIG_FILTER) [N]
```

```
Unix domain sockets (CONFIG_UNIX) [Y]
TCP/IP networking (CONFIG_INET) [Y]
IP:Multicasting (CONFIG_IP_MULTICAST) [N]
IP:Advanced router (CONFIG_IP_ADVENCED_ROUTER) [N]
IP:Kernel level autoconfiguration (CONFIG_IP_PNP) [N]
IP:Firewalling (CONFIG_IP_FIREWALL) [N] Y
IP:Transparent proxy support (CONFIG_IP_TRANSPARENT_PROXY) [N]
IP:Masquerading (CONFIG_IP_MASQUERADE0 [N]
IP:ICMP masquerading (CONFIG_IP_MASQUERADE_ICMP) [N]
IP:Optimize as router not host (CONFIG_IP_ROUTER) [N]
IP:Tunneling (CONFIG_NET_IPIP) [N]
IP:GRE tunnels over IP (CONFIG_NET_IPGRE) [N]
IP:Aliasing support (CONFIG_IP_ALIAS) [N]
IP:TCP syncookie support (CONFIG_SYN_COOKIES) [N] Y
IP:Reverse ARP (CONFIG_INET_RARP) [N]
IP:Allow large windows (CONFIG_SKB_LARGE) [Y]
The IPX protocol (CONFIG_IPX) [N]
AppleTalk DDP (CONFIG_ATALK) [N]
```

Telephony support

```
Linux telephony support (CONFIG_PHONE) [N/y/m/?] (NEW)
```

SCSI support

```
SCSI support (GONFIG_SCSI) [Y]
SCSI disk support (CONFIG_BLK_DEV_SD) [Y]
SCSI tape support (CONFIG_CHR_DEV_ST) [N]
SCSI CD-ROM support (CONFIG_BLK_DEV_SR) [N]
SCSI generic support (CONFIG_CHR_DEV_SG) [N]
Probe all LUMs on each SCSI device (CONFIG_SCSI_MULTI-LUM) [Y] N
Verbose SCSI error reporting (kernel size +=12K) (CONFIG_SCSI_CONSTANTS) [Y] N
SCSI logging facility (CONFIG_SCSI_LOGGING) [N]
```

SCSI low-level drivers

```
7000FASST SCSI support (CONFIG_SCSI_7000FASST) [N]
ACARD SCSI support (CONFIG_SCSI_ACARD) [N]
Adaptec AHA152X/2825 support (CONFIG_SCSI_AHA152X) [N]
Adaptec AHA1542 support (CONFIG_SCSI_AHA1542) [N]
Adaptec AHA1740 support (CONFIG_SCSI_AHA1740) [N]
Adaptec AIC7xxx support (CONFIG_SCSI_AIC7XXX) [N] Y
Enable Tagged Command Queuering (CONFIG_AIC7XXX_TCQ_ON_BY_DEFAULT) [N] Y
Maximum number of TCQ commands per device (CONFIG_AIC7XXX_CMDS_PER_DEVICE) [8]
```

```
Collect statistics to report in /proc (CONFIG_AIC7XXX_PROC_STATS) [N]
Delay in seconds after SCSI bus reset (CONFIG_AIC7XXX_RESET_DELAY) [5]
IBM ServeRAID support (CONFIG_SCSI_IPS) [N]
AdvanSys SCSI support (CONFIG_SCSI_ADVANSYS) [N]
Always IN2000 SCSI support (CONFIG_SCSI_IN2000) [N]
AM53/79C974 PCI SCSI support (CONFIG_SCSI_AM53C974) [N]
AMI MegaRAID support (CONFIG_SCSI_MEGARAID) [N]
BusLogic SCSI support (CONFIG_SCSI_BUSLOGIC) [N]
DTC3180/3280 SCSI support (CONFIG_SCSI_DTC3280) [N]
EATA ISA/EISA/PCI (DPT and generic EATA/DMA-compliant boards) support
(CONFIG_SCSI_EATA) [N]
EATA-DMA (DPT, NEC, AT&T, SNI, AST, Olivetti, Alphatronix) support
(CONFIG_SCSI_EATA_DMA) [N]
EATA-PIO (old DPT PM2001, PM2012A) support (CONFIG_SCSI_EATA_PIO) [N]
Future Domain 16xx SCSI/AHA-2920A support (CONFIG_SCSI_FUTURE_DOMAIN) [N]
GDT SCSI Disk Array Controller support (CONFIG_SCSI_GDTH) [N]
Generic NCR5380/53c400 SCSI support (CONFIG_SCSI_GENERIC_NCR5380) [N]
Initio 9100U(W) support (CONFIG_SCSI_INITIO) [N]
Initio INI-A100U2W support (CONFIG_SCSI_INIA100) [N]
NCR53c406a SCSI support (CONFIG_SCSI_NCR53C406A) [N]
symbios 53c416 SCSI support (CONFIG_SCSI_SYM53C416) [N]
Simple 53c710 SCSI support (Compaq, NCR machines) (CONFIG_SCSI_SIM710) [N]
NCR53c7,8xx SCSI support (CONFIG_SCSI_NCR53C7xx) [N]
NCR53C8XX SCSI support (CONFIG_SCSI_NCR53C8XX) [N]
SYM53C8XX SCSI support (CONFIG-SCSI_SYM53C8XX) [Y] N
PAS16 SCSI support (CONFIG_SCSI_PAS16) [N]
PCI2000 support (CONFIG_SCSI_PCI2000) [N]
PCI2220i support (CONFIG_SCSI_PCI2220I) [N]
PSI240i support (CONFIG_SCSI_PSI240I) [N]
Qlogic FAS SCSI support (CONFIG_SCSI_QLOGIC_FAS) [N]
Qlogic ISP SCSI support (CONFIG_SCSI_QLOGIC_ISP) [N]
Qlogic ISP FC SCSI support (CONFIG_SCSI_QLOGIC_FC) [N]
Seagate ST-02 and Future Domain TMC-8xx SCSI support (CONFIG_SCSI_SEAGATE) [N]
Tekram DC390(T) and Am53/79C974 SCSI support (CONFIG_SCSI_DC390T) [N]
Trantor T128/T128F/T228 SCSI support (CONFIG_SCSI_T128) [N]
UltraStor 14F/34F support (CONFIG_SCSI_U14_34F) [N]
UltraStor SCSI support (CONFIG_SCSI_ULTRASTOR) [N]
```

Network device support

```
Network device support (CONFIG_NETDEVICES) [Y]
```

ARQnet devices

```
ARCnet support (CONFIG_ARCNET) [N]
Dummy net driver support (CONFIG_DUMMY) [M] Y
EQL (serial line load balancing) support (CONFIG_EQUALIZER) [N]
General Instruments Surfboard 1000 (CONFIG_NET_SB1000) [N]
```

Ethernet (10 or 100Mbit)

```
Ethernet (10 or 100Mbit) (CONFIG_NET_ETHERNET) [Y]
3COM cards (CONFIG_NET_VENDOR_3COM) [N]
AMD LANCE and PCnet (AT1500 and NE2100) support (CONFIG_LANCE) [N]
Western Digital/SMC cards (CONFIG_NET_VENDOR_SMC) [N]
Racal-Interlan (Micom) NI cards (CONFIG_NET_VENDOR_RACAL) [N]
Other ISA cards (CONFIG_NET_ISA) [N]
EISA, VLB, PCI and on board controllers (CONDIF_NET_EISA) [Y]
AMD PCnet32 (VLB and PCI) support (CONFIG_PCNET32) [N]
Apricot Xen-II on board Ethernet (CONFIG_APRICOT) [N]
CS89x0 support (CONFIG_CS89x0) [N]
DM9102 PCI Fast Ethernet Adapter support (EXPERIMENTAL) (CONFIG_DM9102) [N]
Generic DECchip & DIGITAL EtherWORKS PCI/EISA (CONFIG_DE4X5) [N]
DECchip Tulip (dc21x4x) PCI support (CONFIG_DEC_ELCP) [N]
Digi Intl. RightSwitch SE-X support (CONFIG_DGRS) [N]
EtherExpressPro/100 support (CONFIG_EEXPRESS_PRO100) [Y]
PCI NE2000 support (CONFIG_NE2K_PCI) [N]
TI ThunderLAN support (CONFIG_TLAN) [N]
VIA Rhine support (CONFIG_VIA_RHINE) [N]
SiS 900/7016 PCI Fast Ethernet Adapter support (CONFIG_SIS900) [N/y/m/?] (NEW)
Pocket and portable adaptors (CONFIG_NET_POCKET) [N]
SysKonnect SK-98xx support (CONFIG_SK98LIN) [N/y/m/?] (NEW)
FDDI driver support (CONFIG_FDDI) [N]
PPP (point-to-point) support (CONFIG_PPP) [N]
SLIP (serial line) support (CONFIG_SLIP) [N]
Wireless LAN (non-hamradio) (CONFIG_NET_RADIO) [N]
```

Token ring devices

```
Token Ring driver support (CONFIG_TR) [N]
Fibre Channel driver support (CONFIG_NET_FC) [N]
```

Wan interfaces

```
Comtrol Hostess SV-11 support (CONFIG_HOSTESS_SV11) [N]
COSA/SRP sync serial boards support (CONFIG_COSA) [N]
Sealevel Systems 4021 support (CONFIG_SEALEVEL_4021) [N]
MultiGate (COMX) synchronous serial boards support (CONFIG_COMX) [N/y/m/?] (NEW)
```

```
Frame relay DLCI support (CONFIG_DLCI) [N]
WAN drivers (CONFIG_WAN_DRIVERS) [N]
SBNI12-xx support (CONFIG_SBNI) [N]
```

Amateur Radio support

```
Amateur Radio support (CONFIG_HAMRADIO) [N]
```

IrDA subsystem support

```
IrDA subsystem support (CONFIG_IRDA) [N]
```

ISDN subsystem

```
ISDN support (CONFIG_ISDN) [N]
```

Old CD-ROM drivers (not SCSI, not IDE)

```
Support non-SCSI/IDE/ATAPI CDROM drives (CONFIG_CD_NO_IDESCSI) [N]
```

Character devices

```
Virtual terminal (CONFIG_VT) [Y]
Support for console on virtual terminal (CONFIG_VT_CONSOLE) [Y]
Standard/generic (dumb) serial support (CONFIG_SERIAL) [Y]
Support for console on special port (CONFIG_SERIAL_CONSOLE) [N]
Extended dumb serial driver options (CONFIG_SERIAL_EXTENDED) [N]
Non-standard serial port support (CONFIG_SERIAL_NONSTANDAR) [N]
Unix98 PTY support (CONFIG_UNIX98_PTYS) [Y]
Maximum numbers of Unix98 PTYs in use (0-2048) (CONFIG_UNIX98_PTY_COUNT) [256]
128
Mouse support (Not serial mice) (CONFIG_MOUSE) [Y]
```

Mice

```
ATIXL busmouse support (CONFIG_ATIXL_BUSMOUSE) [N]
Logitech busmouse support (CONFIG_BUSMOUSE) [N]
Microsoft busmouse support (CONFIG_MS_BUSMOUSE) [N]
PS/2 mouse (aka 揌uxiliary device? support (CONFIG_PSMOUSE) [Y]
C&T 82C710 mouse port support (CONFIG_82c710_MOUSE) [Y] N
PC110 digitizer pad support (CONFIG_PC110_PAD) [N]
```

Joystick support

```
Joystick support (CONFIG_JOYSTICK) [N]
QIC-02 tape support (CONFIG_QIC02_TAPE) [N]
Watchdog Timer support (CONFIG_WATCHDOG) [N]
/dev/nvram support (CONFIG_NVRAM) [N]
Enhanced Real Time Clock support (CONFIG_RTC) [N]
```

Video for Linux

```
Video for Linux (CONFIG_VIDEO_DEV) [N]
Double Talk PC internal speech controller support (CONFIG_DTLK) [N]
```

Ftape, the floppy tape device driver

```
Ftape (QIC-80/Travan) support (CONDFIG_FTAPE) [N]
```

Filesystems

```
Quota support (CONFIG_QUOTA) [N] Y
Kernel automounter support (CONFIG_AUTOFS_FS) [Y] N
Amiga FFS filesystem support (CONFIG_AFFS_FS) [N]
Apple Macintosh filesystem support (CONFIG_HFS_FS) [N]
DOS FAT fs support (CONFIG_FAT_FS) [N]
ISO 9660 CDROM filesystem support (CONFIG_ISO9660FS) [Y]
Microsoft Joliet CDROM extensions (CONFIG_JOLIET) [N]
Minix fs support (CONFIG_MINIX_FS) [N]
NTFS filesystem support (CONFIG_NTFS_FS) [N]
OS/2 MPFS filesystem support (CONFIG_HPFS_FS) [N]
/proc filesystem support (CONFIG_PROC_FS) [Y]
/dev/pts filesystem support (CONFIG_DEVPTS_FS) [Y]
ROM filesystem support (CONFIG_ROMFS_FS) [N]
Second extended filesystem (CONFIG_EXT2_FS) [Y]
System V and coherent filesystem support (CONFIG_SYSV_FS) [N]
UFS filesystem support (CONFIG_UFS_FS) [N]
```

Network File Systems

```
Coda filesystem support (Advanced Network fs) (CONFIG_CODA_FS) [N]
NFS filesystem support (CONFIG_NFS_FS) [Y] N
SMB filesystem support (CONFIG_SMB_FS) [N]
NCP filesystem support (CONFIG_NCP_FS) [N]
```

Partition Types

```
BSD disklabel (BSD partition tables) support (CONFIG_BSD_DISKLABEL) [N]
Macintosh partition map support (CONFIG_MAC_PARTION) [N]
SMD disklabel (Sun partition tables) support (CONFIG_SMD_DISKLABEL) [N]
Solaris (x86) partition table support (CONFIG_SOLARIS_X86_PARTITION) [N]
```

Console drivers

```
VGA text console (CONFIG_VGA_CONSOLE) [Y]
Video mode selection support (CONFIG_VIDEO_SELECT) [N]
Sound
Sound card support (CONFIG_SOUND) [N]
(Security options will appear only if your are patched your kernel with the Openwall
Project patch).
```

Security options

```
Non-executable user stack area (CONFIG_SECURE_STACK) [Y]
Autodetect and emulate GCC trampolines (CONFIG_SECURE_STACK_SMART) [Y]
Restricted links in /tmp (CONFIG_SECURE_LINK) [Y]
Restricted FIFOs in /tmp (CONFIG_SECURE_FIFO) [Y]
Restricted /proc (CONFIG_SECURE_PROC) [N] Y
Special handling of fd 0, 1, and 2 (CONFIG_SECURE_FD_0_1_2) [Y]
Enforce RLIMIT_NPROC on execve(2) (CONFIG_SECURE_RLIMIT_NPROC) [Y]
Destroy shared memory segments not in use (CONFIG_SECURE_SHM) [N] Y
```

Kernel hacking

```
Magic SysRq key (CONFIG_MAGIC_SYSRQ) [N]
```

注意：各个配置选项的意义可以在查看配置时的帮助。

现在返回到“/usr/src/linux/”目录下，下面开始进行内核编译工作，按照下面的命令进行：

```
[root@deep]# make dep; make clean; make bzImage
```

这行包含三个命令，第一个命令“make dep”实际上读取上一步配置过程生成的配置文件，来创建对应于配置的依赖关系树，从而决定哪些需要编译而哪些不需要；第二命令“make clean”完成删除前面步骤留下的文件，以避免出现一些错误；第三步“make bzImage”实现完全编译内核。

处理结束以后，生成的被压缩的新内核就可以被安装了。如果你在回答 Enable loadable module support （CONFIG_MODULES）选“Yes”，在安装新内核以前，就还需要编译一些模块并且正确地安装。通过下面的命令来实现对模块的编译和安装：

```
[root@deep]# make modules;make modules_install
```

安装新内核

拷贝新内核文件“/usr/src/linux/arch/i386/boot/bzImage”到启动目录，并改为合适的名字：

```
[root@deep]# cp /usr/src/linux/arch/i386/boot/bzImage
/boot/vmlinuz-kernel.version.number
```

注意：推荐新内核的命名格式为 vmlinuz-2.*.*。给新内核起这样的新名字，对于希望用 makebootdisk 命令来创建一个新的紧急启动盘是有意义的，因为 makebootdisk 命令对参数有特殊的要求。

拷贝“/usr/src/linux/System.map”到启动目录下，并设定合适的名字：

```
[root@deep]# cp /usr/src/linux/System.map
/boot/System.map-kernel.version.number
```

进入启动目录下，将目录下的链接文件“vmlinuz”及“System.map”指向新的内核：

```
[root@deep]# cd /boot
[root@deep]# ln -fs vmlinuz-kernel.version.number vmlinuz
[root@deep]# ln -fs System.map-kernel.version.number System.map
```

必须将链接文件 vmlinuz 及 System.map 指向新内核的相关文件，因为如果没有新的链接，LILO 程序将缺省地自动使用老内核。

删除启动目录下无用过期的文件：

```
[root@deep]# rm -f module-info
[root@deep]# rm -f initrd-2.2.12-20.img
```

“module-info”链接指向系统的旧内核的模块目录。由于现在安装了新内核，我们就无需保持这个文件。文件“initrd-2.2.12-2”包含在可以使用磁盘之前初始化系统所需的 RAM 盘映像。这个文件只有在系统有 scsi 设备的情况下才会在安装系统时

生成。由于这里生成编译的是一个单块（非模块化）的新内核。所以即使系统有 scsi 设备，这里仍然可以安全的删除该文件。

创建一个新 Linux 内核目录，使以后编译别的程序可以找到和该内核相关的系统头文件：

为了编译内核，前面在“/usr/include/”目录下创建了三个符号链接，由于顺利地进行了新内核的编译，所以编译别的程序也应该是成功的。“/usr/indlude”目录是包含所有的系统头文件的目录，编译程序时，系统会到该目录下寻找头文件信息。当被编译的程序需要系统当前内核的某些函数信息时，就会使用 asm、linux 及 scsi 链接来获取信息。

```
[root@deep]# mkdir -p /usr/src/linux-2.2.14/include
[root@deep]# cp -r /usr/src/linux/include/asm-generic
/usr/src/linux-2.2.14/include
[root@deep]# cp -r /usr/src/linux/include/asm-i386
/usr/src/linux-2.2.14/include
[root@deep]# cp -r /usr/src/linux/include/linux /usr/src/linux-2.2.14/include
[root@deep]# cp -r /usr/src/linux/include/net /usr/src/linux-2.2.14/include
[root@deep]# cp -r /usr/src/linux/include/video /usr/src/linux-2.2.14/include
[root@deep]# cp -r /usr/src/linux/include/scsi /usr/src/linux-2.2.14/include
[root@deep]# rm -rf /usr/src/linux
[root@deep]# cd /usr/src
[root@deep]# ln -s /usr/src/linux-2.2.14 linux
```

首先，创建一个新的根据新内核的版本命名的子目录：Linux-2.2.14。然后拷贝 /usr/linux/include 的子目录 asm-generic、asm-i386、linux,net、video 及 scsi 到 /usr/src/linux-2.2.14/include。然后删除整个编译新内核的子目录，创建一个新的符号链接 linux 指向/usr/src/linux-2.2.14/。从而随后的编译新的程序可以正确地找到和新内核相对应的头文件信息。

最后，编辑“/etc/lilo.conf”加入新内核作为引导选项：

第一步：

编辑文件“lilo.conf ”文件（vi /etc/lilo.conf）对“image=/boot/”一行做适当的修改：

```
[root@deep]# vi /etc/lilo.conf
```

例：

```
boot=/dev/sda
map=/boot/map
install=/boot/boot.b
```

```
prompt
timeout=00
restricted
password=somepasswd
image=/boot/vmlinuz-kernel.version.number #(add your new kernel name file here).
label=linux
root=/dev/sda6
read-only
```

注意：记住删除包含“initrd=/boot/initrd-2.2.12-20.img”一行的内容，因为单块（非模块化）的内核不需要 initrd 文件。

第二步：更新对 lilo.conf 作的修改。

```
[root@deep]# /sbin/lilo -v
LILO version 21, [Copyright 1992-1998 Werner Almesberger
Reading boot sector from /dev/sda
Merging with /boot/boot.b
Boot image: /boot/vmlinuz-2.2.14
Added linux *
/boot/boot.0800 exits ?no backup copy made.
Writing boot sector.
```

注意：如果在配置新内核回答“Unix98 PTY support(CONFIG_UNIX98_PTYS)”为“No”时，则需要编辑“文件/etc/fstab”，删除下面一行：

```
none /dev/pts devpts gid=5,mode=620 0 0
```

删除和模块相关的程序、文件和内容

缺省地，当你第一次安装 RedHat 版本的 Linux 时，内核是被编译为模块化的方式。也就是说需要的每个设备或功能都作为一个模块存在，并且由名为 kerneld 的内核守护进程（在 2.2.*版本名为 kmod）在需要时自动将功能对应的模块加载到内核里，在不需要时自动将其从内核中去除。

kerneld 守护进程使用“/etc/conf.modules”文件来获取模块信息：例如，网卡需要特殊的参数时，kerneld 从文件“conf.modules”中取得网卡模块的相关参数，但是由于我们这里编译得到的内核是单块（非模块化）内核，所以我们不需要“/etc/conf.modules”文件，并且可以删除“modutils”程序。

Modutils 软件包包括程序 kerneld，用于实现自动从内核中加载和删除模块。需要加载的模块一般完成设备驱动及文件系统功能。

为了去除“conf.modules”文件，使用命令：

```
[root@deep]# rm -f /etc/conf.modules
```

为了卸载 modutils 包，使用命令：

```
[root@deep]# rpm -e --nodeps modutils
```

最后，编辑文件“rc.sysinit”，将所有的有“depmod –a”内容的行注释掉（通过在相关行前面加#）。注释掉这些行是因为在启动时，系统会缺省的读取“rc.sysinit”脚本文件来查看模块依赖性。

在“rc.sysinit”文件中做如下修改：

```
if [ -x /sbin/depmod -a -n "$USEMODULES" ]; then
```

改为：

```
#if [ -x /sbin/depmod -a -n "$USEMODULES" ]; then



if [ -L /lib/modules/default ]; then
INITLOG_ARGS= action "Finding module dependencies" depmod -a default
else
INITLOG_ARGS= action "Finding module dependencies" depmod -a
fi
fi
```

改为：

```
# if [ -L /lib/modules/default ]; then
# INITLOG_ARGS= action "Finding module dependencies" depmod -a default
# else
# INITLOG_ARGS= action "Finding module dependencies" depmod -a
# fi
# fi
```

注意：再次强调，只有在配置内核时对问题“Enable loadable module support (CONFIG_MODULES)?”回答“NO”时，才可以实施删除模块相关内容的步骤。

现在重新启动机器：

```
[root@deep]# reboot
```

系统重新启动以后，使用下面的命令来证实使用的是新的内核：

```
[root@deep]# uname -a
Linux deep.openarch.com 2.2.14 #1 Mon Jan 10 10:40:35 EDT 2000 i686 unknown
```

创建新的急救盘

如果前面的步骤都很顺利的话，应该为新内核创建一个紧急恢复盘。以根用户登录，使用命令如下：

```
[root@deep]# mkbootdisk --device /dev/fd0 2.2.14
Insert a disk in /dev/fd0. Any information on the disk will be lost.
Press <Enter> to continue or ^C to abort:
```

注意：mkbootdisk 程序只能在模块化内核的系统中运行。在单块内核系统中，是不能使用该命令的。

什么是拯救（Rescue）模式

拯救模式（Rescue Mode）指从软盘启动一个小型的 linux 系统。通过拯救模式，使管理员可以在不能从硬盘启动的情况下访问硬盘的数据。

制作紧急启动软盘

由于只能在模块化内核系统中创建紧急恢复盘，所以应该寻求另外一种在硬盘上的内核被破坏的情况下启动 linux 系统的方法。成功地以根用户身份登录到系统以后，按照下面的方法创建一个紧急启动盘：

将一个软盘格式化：

```
[root@deep]# fdformat /dev/fd0H1440
Double-sided, 80 tracks, 18 sec/track. Total capacity 1440 kB.
Formatting ... done
Verifying ... done
```

拷贝启动目录中的文件 vmlinuz 到软盘中：

```
[root@deep]# cd /boot
[root@deep]# cp vmlinuz /dev/fd0
cp: overwrite `/dev/fd0'? y
```

这里的的 vmlinuz 是系统启动使用的内核映相。

使用下面的命令确定内核启动设备：

```
[root@deep]# rdev
/dev/sda12 /
```

内核的根设备（root device）是指根文件系统所在的分区。这里的例子根设备为“/dev/sda12”，你的系统也许会不同。

设置内核的根设备：

```
[root@deep]# rdev /dev/fd0 /dev/sda12
```

第二个参数就是第三步得到的值

设置根设备为只读标记：

```
[root@deep]# rdev -R /dev/fd0 1
```

这将使 linux 初始将根文件系统以只读模式加载，通过设置为只读模式，可以避免出现若干警告错误信息。

然后将启动软盘放进软盘驱动器中，重新启动机器：

```
[root@deep]# reboot
```

更新“/dev”目录下的项

若系统内核做了升级或在系统中添加了新的设备，就需要确保更新了“/dev”目录的内容：

```
[root@deep]# cd /dev
[root@deep]# ./MAKEDEV update
```

注意：MAKEDEV 是调用 mknod 的一个脚本文件。

第四部分：与网络相关的参考资料

第六章

TCP/IP 网络管理

概述

网络管理涵盖了很多方面的内容：一般来说包括收集网络的统计数据和状态信息，并且在需要时采取一定的措施来解决出现的故障。最原始的网络监控技术是定期“ping”重要的主机。更复杂的网络监控则需要监视网络上的各个设备的状态和统计信息，如：各种类型的数据包（datagram）统计及各种类型的错误的统计。

这一章主要是关于联网设备，网络相关配置文件、重要的网络命令和 TCP/IP 安全等方面的内容。

在服务器上安装多块网卡

在使用 Linux 作为两个以太网之间的网关的情况下，服务器至少需要配置两块网卡。为了减少启动时可能出现的问题，Linux 内核不会自动检测多个网卡。若需要在服务器上安装多块网卡，对于已经将网卡的驱动编译进内核中的系统，则需要在“/etc/lilo.conf”文件中指定各个网卡的参数信息；而对于没有将网卡的驱动编译到内核而是作为模块动态载入的系统，应该在“conf.modules”文件中进行相应的配置。

若设备驱动被编译为模块（内核的模块）：对于 PCI 设备，模块将自动检测到所有已经安装到系统上的设备；对于 ISA 卡，则需要向模块提供 IO 地址，以使模块知道在何处寻找该卡，这些信息在“/etc/conf.modules”中提供。

例如，我们有两块 ISA 总线的 3c509 卡，一个 IO 地址是 0x300，另一个是 0x320。编辑“conf.modules”文件如下：

```
alias eth0 3c509
alias eth1 3c509
options 3c509 io=0x300,0x320
```

这是说明 3c509 的驱动程序应当被 eth0 或 eth1 加载（alias eth0,eth1），并且它们应该以参数 io=0x300,0x320 被装载，这样驱动程序知道到哪里去寻找网卡，其中 0x 是不可缺少的。

对于 PCI 卡，仅仅需要 alias 命令来使 ethN 和适当的驱动模块名关联，PCI 卡的 IO 地址将会被自动的检测到。对于 PCI 卡，编辑“conf.modules”文件如下：

```
alias eth0 3c509
alias eth1 3c509
```

若驱动已经被编译进了内核：系统启动时的 PCI 检测程序将会自动找到所有相关的网卡。ISA 卡一般也能够被自动检测到，但是在某些情况下，ISA 卡仍然需要做下面的配置工作：

在“/etc/lilo.conf”中增加配置信息，其方法是通过 LILO 程序将启动参数信息传递给内核。对于 ISA 卡，编辑“lilo.conf”文件，增加如下内容：

```
append="ether="0,0,eht1"
```

注意：先不要在“lilo.conf”中加入启动参数，测试一下你的 ISA 卡，若失败再使用启动参数。

如果用传递启动参数的方法，eth0 和 eth1 将按照启动时被发现的顺序来设置。因为我们已经重新编译了内核，所以必须使用第二种方法（在 lilo.conf 中加入启动参数）在系统中安装我们的第二块网卡。这种方法只对 ISA 卡有必要，PCI 卡会被自动查找到，所以没有什么必要。

和网络相关的一些配置文件

在 Linux 系统中，TCP/IP 网络是通过若干个文本文件进行配置的，也许你需要编辑这些文件来完成联网工作。下面的部分介绍基本的 TCP/IP 配置文件：

“/etc/HOSTNAME”文件：

该文件包含了系统的主机名称，包括完全的域名，如：

```
deep.openarch.com
```

“/etc/sysconfig/network-scripts/ifcfg-ethN”文件：

在 RedHat6.1 中，系统网络设备的配置文件保存在“/etc/sysconfig/network-scripts”目录下，ifcfg-eth0 包含第一块网卡的配置信息，ifcfg-eht1 包含第二块网卡的配置信息。

下面是“/etc/sysconfig/network-scripts/ifcfg-eth0”文件的示例：

```
DEVICE=eth0
IPADDR=208.164.186.1
NETMASK=255.255.255.0
NETWORK=208.164.186.0
BROADCAST=208.164.186.255
ONBOOT=yes
BOOTPROTO=none
USERCTL=no
```

若希望手工修改网络地址或在新的接口上增加新的网络界面，可以通过修改对应的文件（ifcfg-ethN）或创建新的文件来实现。

DEVICE=name　　name 表示物理设备的名字

IPADDR=addr　　addr 表示赋给该卡的 IP 地址

NETMASK=mask　　mask 表示网络掩码

NETWORK=addr　　addr 表示网络地址

BROADCAST=addr　　addr 表示广播地址

ONBOOT=yes/no　　启动时是否激活该卡

BOOTPROTO=proto　　proto 取值可以是：

- none：无须启动协议
- bootp：使用 bootp 协议
- dhcp：使用 dhcp 协议

USERCTL=yes/no　　是否允许非 root 用户控制该设备

“/etc/resolv.conf”文件：

该文件是由解析器（resolver，一个根据主机名解析 IP 地址的库）使用的配置文件，示例如下：

```
search openarch.com
nameserver  208.164.186.1
nameserver  208.164.186.2
```

“search domainname.com”表示当提供了一个不包括完全域名的主机名时，在该主机名后添加 domainname.com 的后缀；“nameserver”表示解析域名时使用该地址指定的主机为域名服务器。其中域名服务器是按照文件中出现的顺序来查询的。

“/etc/host.conf”文件：

该文件指定如何解析主机名。Linux 通过解析器库来获得主机名对应的 IP 地址。下面是一个“/etc/host.conf”的示例：

```
order bind,hosts
multi on
ospoof on
```

“order bind,hosts”指定主机名查询顺序，这里规定先使用 DNS 来解析域名，然后再查询“/etc/hosts”文件。

“multi on”指定是否“/etc/hosts”文件中指定的主机可以有多个地址，拥有多个 IP 地址的主机一般称为具有多个网络界面。

“nospoof on”指不允许对该服务器进行 IP 地址欺骗。IP 欺骗是一种攻击系统安全的手段，通过把 IP 地址伪装成别的计算机，来取得其它计算机的信任。

“/etc/sysconfig/network”文件

该文件用来指定服务器上的网络配置信息，下面是一个示例：

```
NETWORK=yes
RORWARD_IPV4=yes
HOSTNAME=deep.openarch.com
GAREWAY=0.0.0.0
GATEWAYDEV=
```

NETWORK=yes/no	网络是否被配置；
FORWARD_IPV4=yes/no	是否开启 IP 转发功能
HOSTNAME=hostname	hostname 表示服务器的主机名
GAREWAY=gw-ip	gw-ip 表示网络网关的 IP 地址
GAREWAYDEV=gw-dev	gw-dw 表示网关的设备名，如：etho 等

注意：为了和老的软件相兼容，“/etc/HOSTNAME”文件应该用和 HOSTNAME=hostname 相同的主机名。

“/etc/hosts”文件

当机器启动时，在可以查询 DNS 以前，机器需要查询一些主机名到 IP 地址的匹配。这些匹配信息存放在/etc/hosts 文件中。在没有域名服务器情况下，系统上的所有网络程序都通过查询该文件来解析对应于某个主机名的 IP 地址。

下面是一个“/etc/hosts”文件的示例：

```
IP Address              Hostname                Alias

127.0.0.1               Localhost               Gate.openarch.com

208.164.186.1           gate.openarch.com       Gate

208.164.186.2           forest.openarch.com     Forest

208.164.186.3           deep.openarch.com       Deep
```

最左边一列是主机 IP 信息，中间一列是主机名。任何后面的列都是该主机的别名。一旦配置完机器的网络配置文件，应该重新启动网络以使修改生效。使用下面的命令来重新启动网络：

```
/etc/rc.d/init.d/network restart
```

注意：tcpd 程序是负责检测如 telnet、ftp 等的服务请求。

一旦有服务请求到来，inetd 进程将启动 tcpd 进程，tcpd 在日志文件中记录该请求，并且完成一些其他的检测工作。如果一切正常通过，tcpd 将启动相应的服务器进程，然后自己结束。 tcpd 要通过查询 DNS 服务器，先对该客户机的 IP 地址进行反向解析得到主机名，然后再解析得到该主机名对应的 IP 地址，接着，把得到 IP 地址和发出请求的客户的机器的 IP 地址进行比较，通过这些步骤来实现对发送请求的客户机的验证。若两者不匹配，则 tcpd 认为发出请求的机器在伪装成其它的机器，则拒绝请求，这就是为什么有时候 telnet 到 Linux 机器要等待那么长时间的原因，可以通过在“/etc/hosts”加入客户的机器的 IP 地址和主机名的匹配项，就可以减少登录等待时间。

tcpd 可以设置成禁止源路径路由（source-routing）socket 的连接请求。这个设置可以避免黑客用把自己的 IP 地址伪装成别的计算机的 IP 地址的方法，攻击服务器。请注意这个设置对 UDP 无效。当服务器试图把请求服务的客户机的 IP 地址反向解

析成主机名的时候经常会出现超时（timeout）错误。其原因可能是 DNS 服务器没有配置好，或者 DNS 根本没有请求服务的客户端计算机的任何信息。

用命令行手工配置 TCP/IP 网络

ifconfig 是用来设置和配置网卡的命令工行具，为了手工配置网络你需要熟悉这个命令。用该命令的好处是无须重新启动机器。

赋给 eth0 接口 IP 地址 208.164.186.2，使用命令：

```
[root@deep]#ifconfig eth0 208.164.186.2 netmask 255.255.255.0
```

列出所有的网络接口，你可以使用命令：

```
[root@deep]# ifconfig
```

这个命令的输出是这样的：

```
eth0      Link encap:Ethernet HWaddr 00:E0:18:90:1B:56
          inet addr:208.164.186.2 Bcast:208.164.186.255 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:1295 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1163 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:11 Base address:0xa800


lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:3924 Metric:1
          RX packets:139 errors:0 dropped:0 overruns:0 frame:0
          TX packets:139 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

若运行不带任何参数的 ifconfig 命令，这个命令将显示机器所有激活的接口的信息。带有-a 参数的该命令则显示所有的接口的信息，包括没有激活的接口，例如：

```
[root@deep]# ifconfig -a
```

这个命令的输出是这样的：

```
eth0      Link encap:Ethernet HWaddr 00:E0:18:90:1B:56
          inet addr:208.164.186.2 Bcast:208.164.186.255 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
```

```
          RX packets:1295 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1163 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:11 Base address:0xa800


eth1      Link encap:Ethernet HWaddr 00:E0:18:90:1B:56
          inet addr:192.168.1.1 Bcast:192.168.1.255 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:1295 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1163 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:5 Base address:0xa320


lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          UP LOOPBACK RUNNING MTU:3924 Metric:1
          RX packets:139 errors:0 dropped:0 overruns:0 frame:0
          TX packets:139 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

注意：用 ifconfig 命令配置的网络设备参数，在重新启动以后，这些参数设置将会丢失。

给 208.164.186.1 配置缺省网关，使用命令：

```
[root@deep]# route add default gw 208.164.186.1
```

在这个例子中，缺省网关设置为 208.164.186.1。

然后，测试一下是否可以连通本网段的机器，从网络上随便选一台主机测试一下，比如选择 208.164.186.1。

用下面的命令测试一下能否连通这台计算机：

```
[root@deep]# ping 208.164.186.1
```

输出会是这样的：

```
PING 208.164.186.1 (208.164.186.1) from 208.164.186.2 : 56 data bytes
64 bytes from 208.164.186.2: icmp_seq=0 ttl=128 time=1.0 ms
64 bytes from 208.164.186.2: icmp_seq=1 ttl=128 time=1.0 ms
64 bytes from 208.164.186.2: icmp_seq=2 ttl=128 time=1.0 ms
```

```
64 bytes from 208.164.186.2: icmp_seq=3 ttl=128 time=1.0 ms
--- 208.164.186.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.0/1.0/1.0 ms
```

现在可以使用 route 命令输出路由表信息来查看。

用下面的命令显示路由信息：

```
[root@deep]# route -n
```

输出是这样的：

```
Kernel IP routing table
Destination  Gateway      Genmask          Flags Metric Ref Use Iface
208.164.186.20.0.0.0      255.255.255.255  UH       0    0    0 eth0
208.164.186.0 208.164.186.2 255.255.255.0       UG       0    0    0 eth0
208.164.186.0 0.0.0.0     255.255.255.0         U        0    0    0 eth0
127.0.0.0         0.0.0.0      255.0.0.0         U        0    0    0 lo
```

为了快速检查接口状态信息，使用 netstat -i 命令：

```
[root@deep]# netstat -i
```

输出是这样的：

```
Kernel Interface table
Iface MTU Met RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0  1500 0    4236  0       0      0      3700    0      0      0     BRU
lo    3924 0   13300  0       0      0     13300    0      0      0     LRU
ppp0  1500 0     14   1       0      0        16    0      0      0     PRU
```

netstat 命令的另外一个有用处的选项是-t，其将显示所有的激活的 TCP 连接：

```
[root@deep]# netstat -t
```

输出是这样的：

```
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address      Foreign Address State
Tcp      0    0 deep.openar:netbios-ssn gate.openarch.com:1045 ESTABLISHED
Tcp      0    0 localhost:1032          localhost:1033 ESTABLISHED
```

```
Tcp      0    0 localhost:1033          localhost:1032 ESTABLISHED
Tcp      0    0 localhost:1030          localhost:1034 ESTABLISHED
Tcp      0    0 localhost:1031          localhost:1030 ESTABLISHED
Tcp      0    0 localhost:1028          localhost:1029 ESTABLISHED
Tcp      0    0 localhost:1029          localhost:1028 ESTABLISHED
Tcp      0    0 localhost:1026          localhost:1027 ESTABLISHED
Tcp      0    0 localhost:1027          localhost:1026 ESTABLISHED
Tcp      0    0 localhost:1024          localhost:1025 ESTABLISHED
Tcp      0    0 localhost:1025          localhost:1024 ESTABLISHED
```

显示所有的活动的和被监听的 TCP 连接，使用命令：

```
[root@deep]# netstat -vat
```

输出是这样的：

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q   Local AddressForeign Address   State
tcp  0       0 deep.openarch.co:domain *:*          LISTEN
tcp  0       0 localhost:domain        *:*          LISTEN
tcp  0       0 deep.openarch.com:ssh gate.openarch.com:1682 ESTABLISHED
tcp  0       0 *:webcache              *:*          LISTEN
tcp  0       0 deep.openar:netbios-ssn *:*          LISTEN
tcp  0       0 localhost:netbios-ssn   *:*          LISTEN
tcp  0       0 localhost:1032          localhost:1033 ESTABLISHED
tcp  0       0 localhost:1033          localhost:1032 ESTABLISHED
tcp  0       0 localhost:1030          localhost:1031 ESTABLISHED
tcp  0       0 localhost:1031          localhost:1030 ESTABLISHED
tcp  0       0 localhost:1028          localhost:1029 ESTABLISHED
tcp  0       0 localhost:1029          localhost:1028 ESTABLISHED
tcp  0       0 localhost:1026          localhost:1027 ESTABLISHED
tcp  0       0 localhost:1027          localhost:1026 ESTABLISHED
tcp  0       0 localhost:1024          localhost:1025 ESTABLISHED
tcp  0       0 localhost:1025          localhost:1024 ESTABLISHED
tcp  0       0 deep.openarch.com:www   *:*           LISTEN
tcp  0       0 deep.openarch.com:https *:*          LISTEN
tcp  0       0 *:389                   *:*          LISTEN
tcp  0       0 *:ssh                   *:*          LISTEN
```

使系统中所有的网络接口停止工作，用命令：

```
[root@deep]# /etc/rc.d/init.d/network stop
```

启动系统中所有的网络接口，用命令：

```
[root@deep]# /etc/rc.d/init.d/network start
```

TCP/IP 安全问题概述

下面的内容要求读者知道 TCP/IP 协议的基本原理，例如：IP 和 TCP 头的各个字段的功能以及连接的建立过程。这里先简单地介绍一下 TCP/IP 协议。

IP 数据包（packet）

数据包这个术语指的是 IP 协议消息。消息分为消息头和消息体，消息头是 IP 协议使用的各种参数信息，如：源、目的地址等等。消息体是上层协议——TCP 协议的数据内容。

IP 机制

Linux 支持三种 IP 消息类型：ICMP、UDP 和 TCP。一个 ICMP 数据报是网络级的 IP 控制和状态消息。ICMP 消息控制两台端主机之间的与通信相关信息。UDP 类型的 IP 包在网络应用程序之间传送数据，但是不保证传送质量和数据包的传送顺序。UDP 数据的传送类似邮局的明信片发送方式。TCP 类型的 IP 包同样在网络应用程序之间传送数据，但是 TCP 数据包头包含了更多的信息，从而保证了可靠有序的数据传输。传送 TCP 数据类似通过电话系统交谈的过程：数据可靠而且保证传送顺序。大多数 Internet 应用程序都使用 TCP 协议，用 UDP 的比较少。也就是说绝大多数的 Internet 服务都倾向于在客户程序和服务器程序之间建立双向的数据传送通道（用 TCP），而不是单向的数据传送通道（用 UDP）。

IP 数据包头

所有的 IP 数据包（ICMP、UDP 或 TCP）都要包含源、目的的 IP 地址和 IP 包的上层协议的类型，如：ICMP、UDP 或 TCP。根据不同的协议类型，IP 数据包头还包含其它不同的字段。ICMP 类型的 IP 数据包包含一个标识是控制或状态消息的类型字段，以及一个进一步指示是何种类型的控制或状态消息类型的字段。对于 UDP 和 TCP 类型还包含两个分别标识源和目的端口号的字段。TCP 数据包头还包含区分每个数据包的标识符及保证连接状态的信息的字段。

TCP/IP 安全问题

TCP/IP 协议本身有不少的缺陷，允许攻击者利用隐藏通道的方式在看上去正常的数据包中秘密地传输数据。下面我们就通过理论结合例子来对这些缺陷进行进一步说明：

一个隐藏通道指任何被进程用来进行可能对系统安全策略造成威胁的数据传输的通道。TCP/IP 的设计本身并不想有什么隐藏通道，但是设计上的缺陷导致了用来非法传输信息的隐藏通道的存在。

在 TCP/IP 环境下，有若干可以用来建立隐藏通道，在主机之间实现非法通信的方法，例如：

- 旁路包过滤器，网络探测器（sniffer）和“不洁词”（dirty word）搜索引擎。
- 以在正常的信息包中封装经过加密的或没有经过加密的信息的方式，通过网络秘密传输信息。
- 通过将封装有非法信息的伪装包在 Internet 上的某一个站点上“中转”一次来隐藏被传输的数据的源发送位置。

众所周知，TCP 是面向连接的可靠的协议。简单的说，TCP 协议采取一定的措施保证到达远端机器的数据没有被篡改。实现这个特性依赖于 TCP 的三次握手建链机制：

第一步：发送一个带有 ISN（Initial Sequence Number，序列号）的同步数据包-SYN。

主机 A 希望同主机 B 建立连接。主机 A 发送一个数据包，该数据包的 SYN 位被置位，表示希望建立一个新的连接，并且该数据包带有一个 ISN 号，来识别主机之间发送的不同的数据包。

```
Host A ------ SYN(ISN) ------> Host B
```

第二步：远程机器响应一个确认包 ACK。

主机 B 通过发送一个 SYN 位和 ACK 位被置位的数据包来响应该主机 A 的连接请求。这个响应包中不但包含了主机 B 的 ISN，而且包含了主机 A 的 ISN+1 表示刚才的连接请求数据包被正确的接收，等待接收 SN 为 ISN+1 的数据包。

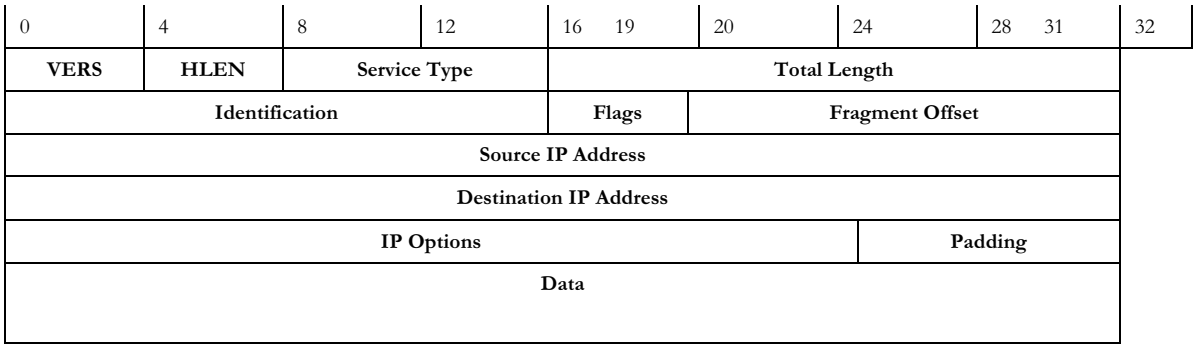
```
Host A <------ SYN(ISN+1)/ACK ------ Host B
```

第三步：主机 A 通过再发送一个确认包 ACK 给主机 B 来完成连接的建立。

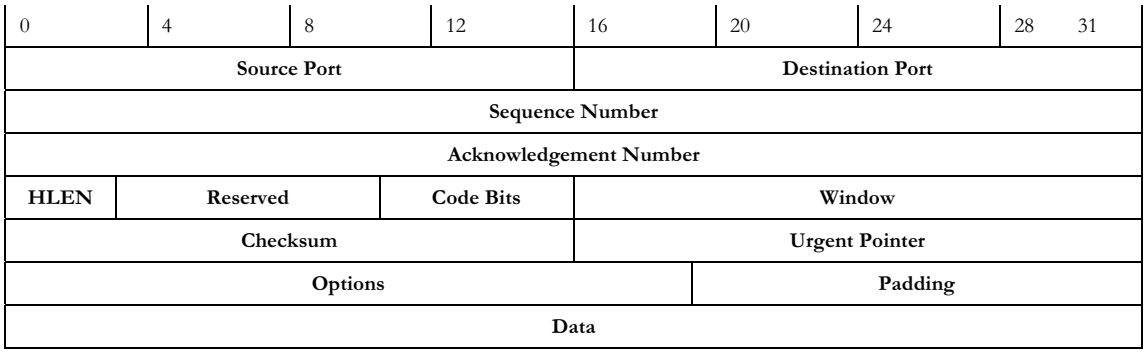
```
Host A ------ ACK ------> Host B
```

整个连接过程在几个毫秒内完成。握手协议确保在主机之间建立一个可靠的链接。这就是为什么 TCP 被看作是面向连接的协议。而 UDP 协议并没有这样的建立连接的协商过程，所以 UDP 被看作是不可靠的协议。

IP 包头：



TCP 包头：



在两种数据包头内，都有多个字段不是进行正常数据传输所必须的，这些选项字段是由发送者根据需要来决定是否使用的。通过对这些选项字段的分析可以发现有很多种可能用来存储和传输数据。利用这些字段的基础是对取值 0-255 的 ASCII 字符进行编码。使用这个方法，可以通过在主机之间传送看似正常的数据包如：建立连接请求、正常的数据传输等，来秘密的传输数据。这些数据报包可以在上层协议数据字段中不包含任何数据或者包含一些无用的数据信息。这些数据包甚至可以包含伪装的目的、源地址和端口号,来实现穿越包过滤防火墙。另外，伪装的数据报还可以估计经过网络上一个无关站点的中转（bounce）来逃避追踪。

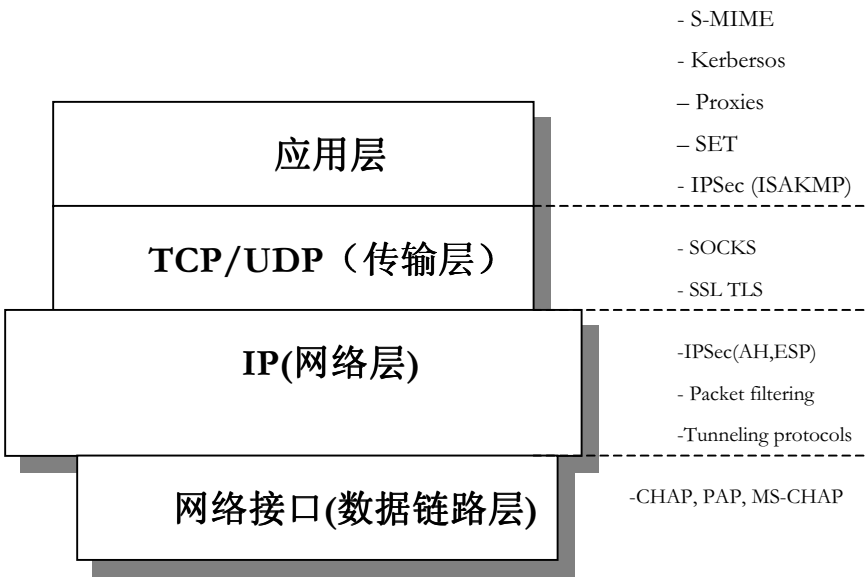
安全问题的解决方案

下面的协议和系统通常被用来在计算机网络中解决和提供不同程度的安全服务。

- 包过滤
- 网络地址转换（NAT）

- IP 安全体系结构（IPSec）
- SOCKS
- 安全套接字层（SSL）
- 应用级代理
- 防火墙
- Kerberos 和其它认证系统
- 安全电子传输（SET）

下面这个图解释了这些安全解决方案的在 TCP/IP 层中的实现：



总结

通过学习和阅读本章，应该掌握如下内容：

应该学会编辑如下的文件来配置 Linux 系统的网络：“/etc/hosts”、“/etc/host.conf”、“/etc/resolv.conf”、“/etc/HOSTNAME”、“/etc/sysconfig/network”和在“/etc/sysconfig/network-scripts”目录下面的脚本文件。

第七章

网络防火墙

概述

有人能够告诉我为什么需要一个商业性的防火墙产品，而不仅仅是使用 IPchains 来限制某些数据包和信息资料？那么，在使用时 IPchains 时我会失去什么呢？是安全性，还是日志登录功能？

现在，请恕我直言，在这个问题上毫无疑问的是 IPchains 已经相当好了，而且从功能和支持范围（support）上来讲，绝大部分时间里甚至比那些商业性防火墙更好用。而且相对于使用商业解决方案，使用 IPchains 会让你更加清楚防火墙对网络产生的影响。现在，许多公司会告诉他们的股东，诸如 CEO/CTO 等等，他们有著名的安全软件公司做支持。防火墙实际上什么都不做，只是让网络中所有信息都顺利通过而已。而且如果使用商业性产品，公司将会感到比使用那些稍有不慎就出麻烦的软件更加容易方便。

最后一点，如果网络被攻破，许多公司都想能够恢复到原状，并从防火墙厂商那里获得一定的赔偿，他们也不管是否确实能从厂商那里得到什么。对于开源解决方案（Open Source Solution），他们所做的比较典型的方法就是只是解雇实施这个方案的人。至少对一些基于 Linux 的商业性防火墙是这样的。因此就有这种可能，IPchains 对于你是足够安全的，而对那些需要与大量股本和债券打交道的人就不是。（在准备为一个价格昂贵的防火墙花费大笔开销之前，做一个性价比分析，并问一些关心的问题是我们推荐的做法，否则，你可能反而会被一些不如 IPchains 的产品搞得焦头烂额）。现在，只有很少的 NT 防火墙性能跟 IPchains 差不多，而且总的来说，对 NT 防火墙一致的意见是在系统 bug 上，而 NT 的系统 bug 是“NT 远非一个真正安全的系统”。

网络防火墙安全策略

一个组织的全局性安全策略必须根据安全分析和业务需求分析来决定。因为防火墙只与网络安全有关，所以只有在正确定义了全局安全策略的情况下，防火墙才具有一定的价值。网络防火墙安全策略是指要明确定义那些允许使用或禁止使用的网络服务，以及这些服务的使用规定和规定中的一些特殊情况。而且，网络防火墙安全策略中的每一条规定都应该在实际应用时得到实现。总的来说，一个防火墙应该使用以下方法之一。

- 每一个没有明确允许的都被拒绝

这种方法堵塞了两个网络之间的所有流量，除了那些被明确允许的服务和应用程序（application）。因此，每一个想保留的服务和应用程序都应该挨个实现，而任何一个可能成为防火墙漏洞的服务和应用程序都不能允许使用。刚才所说的是一个最安全的方法，那就是除非是系统管理员明确允许使用的服务和应用程序，否则都必须拒绝。另一方面，从用户的角度来看，这样可能会限制更多，不是非常方便。在本书中，我们在防火墙配置中会使用这种方法。

- 每一个没有明确拒绝的都允许

这种方法允许两个网络之间所有流量，除非那些被明确禁止的服务和应用程序。因此，每一个不信任或有潜在危害的服务和应用程序都应该逐个拒绝。但是，虽然这对用户是一个灵活和方便的方法，它却可能引起一些严重的安全问题。

包过滤

包过滤是一种内置于 Linux 核心的防火墙类型。过滤型防火墙工作在网络层。数据只有在防火墙规定允许的情况下才能发出去，而到达的包要则根据它们的类型，源地址，目的地址和每个包中包含的端口信息进行过滤。

在绝大部分时间里，包过滤的工作是由一个能根据过滤规定转发数据包的路由器完成的。当一个数据包到达一个能进行包过滤的路由器时，这个路由器从该数据包的包头中解读某些信息，然后根据过滤规定决定数据包是通过还是被丢弃。

下面是能从包头中解读的信息：

- 源 IP 地址
- 目的 IP 地址
- TCP/UDP 源端口
- TCP/UDP 目的端口
- ICMP 消息类型
- 协议信息（TCP，UDP，ICMP 或 IP 隧道）

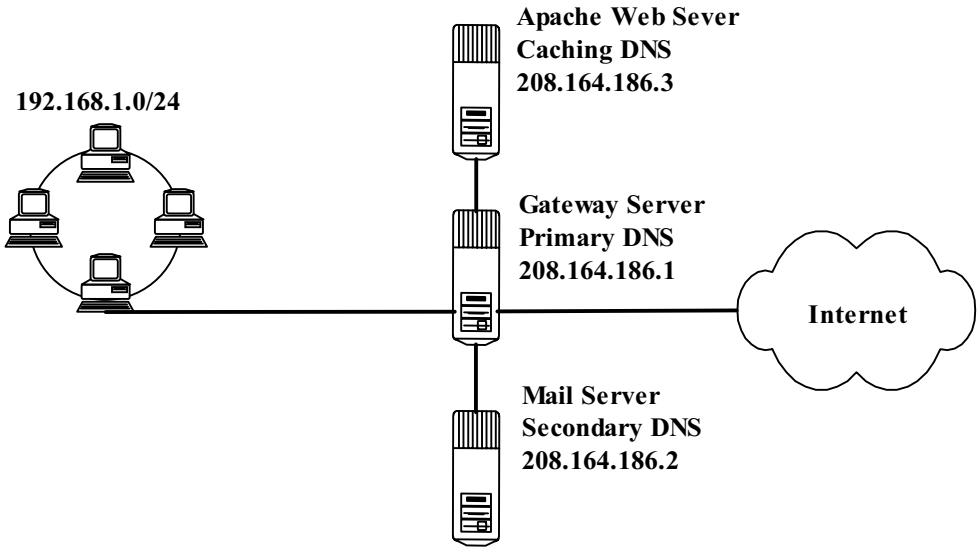
因为只需要分析很少的数据，而且登录到防火墙只占用很少 CPU 时间，网络延迟也非常小，所以如果想使用防火墙保护网络系统，可以通过很多种方法来建设网络。

拓扑结构

在网络中，所有的服务器都至少必须关闭所有没有用的端口，即使它不是一个防火墙服务器。这样做是为了更安全。想象一下，有人获得了对防火墙服务器的访问权，而这只是因为你邻近的服务器没有配置成关闭所有端口，才造成这种情况。

对于本地连接，这也是一样的，没有安全认证的员工能从内部的其他服务器获得对另一个服务器的访问权。

在我们下面的配置中，我们将会给出三个例子，它们有助于你根据要保护的服务器类型和它们在网络结构中的地位决定防火墙规定。第一个防火墙规定适用于 Web 服务器，第二个适用于邮件服务器，最后一个适用于作为内部代理服务器使用的网关服务器。详见图。



www.openarch.com **Caching Only DNS** **208.164.186.3**	**Deep.openarch.com** **Master DNS Server** **208.164.186.1**	**mail.openarch.com** **Slave DNS Server** **208.164.186.2**
1. Unlimited traffic on the loopback interface allowed 2. ICMP traffic allowed 3. DNS Caching and Client Server on port 53 allowed 4. SSH Server on port 22 allowed 5. HTTP Server on port 80 allowed 6. HTTPS Server on port 443 allowed 7. SMTP Client on port 25 allowed 8. FTP Server on ports 20, 21 allowed 9. Outgoing traceroute request allowed	1. Unlimited traffic on the loopback interface allowed 2. ICMP traffic allowed 3. DNS Server and Client on port 53 allowed 4. SSH Server and Client on port 22 allowed 5. HTTP Server and Client on port 80 allowed 6. HTTPS Server and Client on port 443 allowed 7. WWW-CACHE Client on port 8080 allowed 8. External POP Client on port 110 allowed 9. External NNTP NEWS Client on port 119 allowed 10. SMTP Server and Client on port 25 allowed 11. IMAP Server on port 143 allowed 12. IRC Client on port 6667 allowed 13. ICQ Client on port 4000 allowed 14. FTP Client on port 20, 21 allowed 15. RealAudio / QuickTime Client allowed 16. Outgoing traceroute request allowed	1. Unlimited traffic on the loopback interface allowed 2. ICMP traffic allowed 3. DNS Server and Client on port 53 allowed 4. SSH Server on port 22 allowed 5. SMTP Server and Client on port 25 allowed 6. IMAP Server on port 143 allowed 7. Outgoing traceroute request allowed

上表显示了根据防火墙脚本文件在不同服务器上缺省打开的端口。根据服务器必须要对外提供的服务，你必须配置相应的防火墙脚本文件，以允许在指定端口上的通讯。表中，www.openarch.com 是我们的 Web 服务器，mail.openarch.com 是唯一对外的邮件服务器，deep.openarch.com 是网关服务器。它们会用在本章所有的例子中。

编译一个支持 IPCHAINS 防火墙的内核

首先，必须确信 LINUX 内核已经编译成“Network Firewall support”和“Firewalling”（支持网络防火墙——译者注）。记住，所有服务器都至少必须关闭所有不使用的端口，即使它不是防火墙服务器。在内核 2.2.14 中，必须对下面两个问题回答“Y”。

Networking 选项：

```
Network firewalls (CONFIG_FIREFALL) [N] Y
IP: Firewalling (CONFIG_FIREWALL) [N] Y
IP: TCP syncookie support(CONFIG_SYN_COOKIES) [N] Y
```

注释：如果你在阅读《Linux 内核》一章时就重新编译了内核，那么上面这些选项应该已经设置好了。

只对网关服务器有用的 IP Masquerading 和 IP ICMP Masquerading：

```
IP: Masquerading (CONFIG_IP_MASQUERADE) [N] Y
IP: ICMP Masquerading (CONFIG_IP_MASQUERADE_ICMP) [N] Y
```

注释：只有网关服务器才需要支持“IP: Masquerading”和“IP: ICMP Masquerading”内核选项，它需要把内部网对外界伪装起来。

在这里，伪装的意思是，如果在本地网中的一台计算机想要发送一些东西到网络外部，而这个本地网络由一个 Linux 盒 （Linux box：可以是任何简易的 Linux 设备——译者注）充当网络防火墙，那么这个 Linux 盒就可以伪装成那台要发送内容的计算机。例如，Linux 盒转发了到网络外部的所有流量，但是对外部来说，这些都象是来自防火墙本身。

它可以通过两种方式工作：如果外部主机应答，Linux 防火墙就将把这些流量转发到相应的本地计算机，在这种情况下，在本地网络中的计算机对于外部是完全不可见的，即使它们可以访问外界并收到应答。这样，即使本地网络中的计算机没有合法的 IP 地址也能够访问 Internet。

IP 伪装的代码只能工作在下面这种条件下，即在系统启动并安装（mount）了 /proc 文件系统之后，“IP 转发”能够通过下面这行代码执行：

```
echo “1” > /proc/sys/net/ipv4/ip_forward
```

你可以在“/etc/rc.d/rc.local”文件中加上这一行，这样在下次计算机重新启动时就会自动支持 IP 转发。

编辑“rc.local”文件 （通过 vi /etc/rc.d/rc.local ）并加上下面这行：

```
echo “1” > /proc/sys/net/ipv4/ip_forward
```

注释：上面有关 IP 转发的命令行只有在对内核选项“IP: Masquerading (CONFIG_IP_MASQUERDE)”回答“Y”，并且配置了网关服务器来伪装内部网络的情况下才是必须的。

如果选择了支持 IP Masquerading，模块 ip_masq_ftp.o（用于 ftp 文件传输），ip_masq_irc.o（用于 irc chats），ip_masq_quake.o （用途你可以猜得到），ip_masq_vdolive.o（用于 VDOLive 的视频连接），ip_masq_cuseeme.o（用于 CU-SeeMe 广播）和 ip_masq_raudio.o（用于 RealAudio 下载）将会自动编译，它们是这些协议工作时所需要的。

同时，你需要在回答“Enable loadale module support (CONFIG_MODULES)”时选择“Y”以编译一个模块化的内核而不是整体型的内核，这样就可以在网关服务器上使用伪装功能和象 ip_masq_ftp.o 之类的模块。

上面所讲的对“IP: masquerading”而言的基本伪装代码只能处理 TCP 或 UDP 包（以及当前连接的 ICMP 错误）。IP: ICMP Masquerading 增加了对伪装 ICMP 包的附加支持， 比如 Windows 95 跟踪程序使用的 ping 或 probe。

注意：记住，其它类型的服务器象 Web 服务器和邮件服务器并不需要支持这些选项，因为它们要不是拥有一个真实的 IP 地址，就是不用担任内部网络的网关。

注意事项

如果你的系统与 Internet 相连，那你确实可以假设你处在潜在的危险中。因为你的网关是对 Internet 的暴露点，所以我们建议以下几点：

- 网关服务器除非确实有必要，一定不要在上面新增任何应用程序。
- 网关服务器上应该严格限制能够通过的协议种类和数量（许多协议都是潜在的安全漏洞，比如 FTP 和 telnet ）。
- 任何装有机密和敏感信息的系统都不应该能从 Internet 上直接访问。

解释一下防火墙脚本文件的一些规则

下面列出了对将用于防火墙例子的一些规则的解释。这些只是一个参考，防火墙脚本文件都有很清晰的注释说明，也非常好修改。

脚本文件中使用的常量

在脚本文件中，常量定义了大部分将会使用的数值。其中最基本的常量是：

EXTERNAL_INTERFACE

这是与 Internet 相连的对外网卡名字。在以后的例子中定义成“eth0”。

LOCAL_INTERFACE_1

这是与内部局域网相连的对内网卡名字。在以后的例子中定义成“eth1”。

LOOPBACK_INTERFACE

这是回馈网卡名字。在以后的例子中定义成“lo”。

IPADDR

这是对外网卡的 IP 地址。这或者是一个与 InterNIC（网卡——译者注）绑定的静态地址，或者是由 ISP 动态分配的地址（通常是通过 DHCP）。

LOCALNET_1

这是局域网的网络地址。这应该是局域网中所有机器使用的 IP 地址范围。它应该是静态指定的，可以用一个 DHCP 服务器来分配。在后面的例子中，IP 地址范围是 192.168.1.0/24，是 C 类地址的一部分。

ANYWHERE

这是 ipchains 用来匹配所有地址（非广播地址）的地址的一个标志。所有程序都为这个地址提供一个“any/0”的标志，这个地址是 0.0.0.0/0。

NAMESERVER_1

这是主 DNS 服务器或 ISP 的 IP 地址。

NAMESERVER_1

这是第二 DNS 服务器或 ISP 的 IP 地址。

LOOPBACK

回馈地址的范围是 127.0.0.0/8。网卡自己的地址是 127.0.0.1（在/etc/hosts 文件中指定）。

PRIVPORTS

指定优先端口，通常从 0 到 1023。

UNPRIVPORTS

指定非优先端口，通常从 1024 到 65535。它们是动态分配给连接客户端的。

Default Policy

一个防火墙通常有一个缺省的安全策略，以及一系列对应于特殊消息类型的反应动作。这意味着如果有一个数据包不适用于任何已定义的策略，这个缺省策略就会发挥作用。

注释：一个 IP 转发性质（IPFW）的防火墙有两个基本策略，一个是缺省拒绝所有信息，只允许明确规定允许的信息；另一个是缺省接受一切信息，只拒绝明确规定不允许的信息。其中，缺省拒绝一切的策略是我们推荐的，因为通过它更容易建立一个安全得多的防火墙。

允许本地流量

因为缺省策略是拒绝一切信息，所以其中一些需要放开。本地网络服务不通过对外的网卡进行，它们只通过一个特殊的、私有的网卡，叫做回馈网卡。只有回馈网卡允许工作了，本地网络才能正常工作。

```
#Unlimited traffic on the loopback interface.
ichains –A input –i $LOOPBACK_INTERFACE –j ACCEPT
ichains –A output –i $LOOPBACK_INTERFACE –j ACCEPT
```

源地址过滤

在 IP 数据包头中，在 IP 协议中唯一有标识含义的是包的源地址。这种情况就为利用源地址进行欺骗开了后门，因为只要把源地址替换成一个不存在的地址，或是另外一个地址就可以了。这就允许有恶意的人侵入你的系统或伪装成你去攻击别人。

```
# Refuse spoofed packets pretending to be from the external address.
ipchains –A input –i $EXTERNAL_INTERFACE –s $IPADDR –l –j DENY
```

在任何情况下，起码有 7 种源地址需要在对外网卡上设置成拒绝。

下面这些地址是从外面进来的数据包所需要的：

- 自己对外的 IP 地址

- A 类私有地址
- B 类私有地址
- C 类私有地址
- D 类多址地址
- E 类保留地址
- 回馈（loopback）网卡地址

除了你自己的 IP 地址以外，应该阻塞所有包含这些源地址的外出数据包，这样才能保护自己避免因为配置上的错误而受到攻击。

其余的规定

在防火墙脚本文件中使用的其它规定是：

- 从外部访问一个服务（Access a Service from the Outside World）
- 向外部提供一个服务
- 伪装内部网络中的计算机

防火墙脚本文件

使用 ipchains 可以建立防火墙，使用 IP 伪装等等。Ipchains 与系统核心交互，并告诉内核过滤哪些数据包。因此所有的防火墙设置都保存在内核中，在系统重新启动时就丢掉了。

为了避免出现这种情况，我们推荐使用 System V（系统 V）的 init 脚本来使安全策略永远有效。要达到这个目的，就应该象下面的例子一样，为每一个服务器在“/etc/rc.d/init.d”下创建一个防火墙脚本文件。为了保险起见，每一个服务器提供不同的服务，并使用不同的防火墙配置。由于这个原因，我们提供了一系列不同的防火墙配置，你可以对它们进行测试并修改成自己所需要的样子。同时，我们也假设你具有关于过滤型防火墙和防火墙规定工作过程的最基本知识。

为 Web 服务器配置 /etc/rc.d/init.d/firewall 脚本文件

下面是用于我们 Web 服务器的配置脚本文件。这个配置允许在回馈网卡上的所有流量，缺省情况下是 ICMP ，DNS 缓存（Caching）和客户服务器（53），SSH 服务器（22），HTTP 服务器（80），HTTPS 服务器（443），SMTP 客户机（25），FTP 服务器（20，21）和 OUTGOING TRACEROUTE 请求（用于了解在访问某个地址过程中出现的错误----译者注）。

如果不需要我在下面文件中缺省列出的某些服务，你可以用行开头加“#”来注释掉该行。如果需要某些被注释掉的服务，去掉该行开头的“#”就可以了。

请在 Web 服务器上创建如下的防火墙脚本文件（用 touch /etc/rc.d/init.d/firewall ）：

```
#!/bin/sh
#
#
------------------------------------------------------------------------
# Last modified by Gerhard Mourani: 02-01-2000
#
------------------------------------------------------------------------
# Copyright (C) 1997, 1998, 1999 Robert L. Ziegler
#
# Permission to use, copy, modify, and distribute this software and its
# documentation for educational, research, private and non-profit purposes,
# without fee, and without a written agreement is hereby granted.
# This software is provided as an example and basis for individual firewall
# development. This software is provided without warranty.
#
# Any material furnished by Robert L. Ziegler is furnished on an
# "as is" basis. He makes no warranties of any kind, either expressed
# or implied as to any matter including, but not limited to, warranty
# of fitness for a particular purpose, exclusivity or results obtained
# from use of the material.
#
------------------------------------------------------------------------
#
# Invoked from /etc/rc.d/init.d/firewall.
# chkconfig: - 60 95
# description: Starts and stops the IPCHAINS Firewall \
# used to provide Firewall network services.
# Source function library.
. /etc/rc.d/init.d/functions
# Source networking configuration.
. /etc/sysconfig/network
# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0
# See how we were called.
case "$1" in
start)
echo -n "Starting Firewalling Services: "
# Some definitions for easy maintenance.
#
```

```
------------------------------------------------------------------------
# EDIT THESE TO SUIT YOUR SYSTEM AND ISP.
EXTERNAL_INTERFACE="eth0" # whichever you use
LOOPBACK_INTERFACE="lo"
IPADDR="208.164.186.3"
ANYWHERE="any/0"
NAMESERVER_1="208.164.186.1" # Your primary name server
NAMESERVER_2="208.164.186.2" # Your secondary name server
SMTP_SERVER="mail.openarch.com" # Your Mail Hub Server.
SYSLOG_SERVER="mail.openarch.com" # Your syslog internal server
SYSLOG_CLIENT="208.164.168.0/24" # Your syslog internal client
LOOPBACK="127.0.0.0/8"
CLASS_A="10.0.0.0/8"
CLASS_B="172.16.0.0/12"
CLASS_C="192.168.0.0/16"
CLASS_D_MULTICAST="224.0.0.0/4"
CLASS_E_RESERVED_NET="240.0.0.0/5"
BROADCAST_SRC="0.0.0.0"
BROADCAST_DEST="255.255.255.255"
PRIVPORTS="0:1023"
UNPRIVPORTS="1024:65535"
#
------------------------------------------------------------------------
# SSH starts at 1023 and works down to 513 for
# each additional simultaneous incoming connection.
SSH_PORTS="1022:1023" # range for SSH privileged ports
# traceroute usually uses -S 32769:65535 -D 33434:33523
TRACEROUTE_SRC_PORTS="32769:65535"
TRACEROUTE_DEST_PORTS="33434:33523"
#
------------------------------------------------------------------------
# Default policy is DENY
# Explicitly accept desired INCOMING & OUTGOING connections
# Remove all existing rules belonging to this filter
ipchains -F
# Set the default policy of the filter to deny.
ipchains -P input DENY
ipchains -P output REJECT
ipchains -P forward REJECT
#
------------------------------------------------------------------------
# Enable TCP SYN Cookie Protection
echo 1 >/proc/sys/net/ipv4/tcp_syncookies
# Enable IP spoofing protection
```

```
# turn on Source Address Verification
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
echo 1 > $f
done
# Disable ICMP Redirect Acceptance
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 0 > $f
done
# Disable Source Routed Packets
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 0 > $f
done
#
------------------------------------------------------------------------------
# LOOPBACK
# Unlimited traffic on the loopback interface.
ipchains -A input -i $LOOPBACK_INTERFACE -j ACCEPT
ipchains -A output -i $LOOPBACK_INTERFACE -j ACCEPT
#
------------------------------------------------------------------------------
# Network Ghouls
# Deny access to jerks
# /etc/rc.d/rc.firewall.blocked contains a list of
# ipchains -A input -i $EXTERNAL_INTERFACE -s address -j DENY
# rules to block from any access.
# Refuse any connection from problem sites
#if [ -f /etc/rc.d/rc.firewall.blocked ]; then
# . /etc/rc.d/rc.firewall.blocked
#fi
#
------------------------------------------------------------------------------
# SPOOFING & BAD ADDRESSES
# Refuse spoofed packets.
# Ignore blatantly illegal source addresses.
# Protect yourself from sending to bad addresses.
# Refuse spoofed packets pretending to be from the external address.
ipchains -A input -i $EXTERNAL_INTERFACE -s $IPADDR -j DENY -l
# Refuse packets claiming to be to or from a Class A private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_A -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_A -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_A -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_A -j REJECT -l
# Refuse packets claiming to be to or from a Class B private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_B -j DENY -l
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_B -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_B -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_B -j REJECT -l
# Refuse packets claiming to be to or from a Class C private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_C -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_C -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_C -j REJECT –l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_C -j REJECT -l
# Refuse packets claiming to be from the loopback interface
ipchains -A input -i $EXTERNAL_INTERFACE -s $LOOPBACK -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $LOOPBACK -j REJECT -l
# Refuse broadcast address SOURCE packets
ipchains -A input -i $EXTERNAL_INTERFACE -s $BROADCAST_DEST -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $BROADCAST_SRC -j DENY -l
# Refuse Class D multicast addresses (in.h) (NET-3-HOWTO)
# Multicast is illegal as a source address.
# Multicast uses UDP.
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_D_MULTICAST -j DENY -l
# Refuse Class E reserved IP addresses
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_E_RESERVED_NET -j DENY -l
# refuse addresses defined as reserved by the IANA
# 0.*.*.*, 1.*.*.*, 2.*.*.*, 5.*.*.*, 7.*.*.*, 23.*.*.*, 27.*.*.*
# 31.*.*.*, 37.*.*.*, 39.*.*.*, 41.*.*.*, 42.*.*.*, 58-60.*.*.*
# 65-95.*.*.*, 96-126.*.*.*, 197.*.*.*, 201.*.*.* (?), 217-223.*.*.*
ipchains -A input -i $EXTERNAL_INTERFACE -s 1.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 2.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 5.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 7.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 23.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 27.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 31.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 37.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 39.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 41.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 42.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 58.0.0.0/7 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 60.0.0.0/8 -j DENY -l
#65: 01000001 - /3 includes 64 - need 65-79 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 65.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 66.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 67.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 68.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 69.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 70.0.0.0/8 -j DENY -l
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -s 71.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 72.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 73.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 74.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 75.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 76.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 77.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 78.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 79.0.0.0/8 -j DENY -l
#80: 01010000 - /4 masks 80-95
ipchains -A input -i $EXTERNAL_INTERFACE -s 80.0.0.0/4 -j DENY -l
# 96: 01100000 - /4 makses 96-111
ipchains -A input -i $EXTERNAL_INTERFACE -s 96.0.0.0/4 -j DENY -l
#126: 01111110 - /3 includes 127 - need 112-126 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 112.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 113.0.0.0/8 -j DENY –l
ipchains -A input -i $EXTERNAL_INTERFACE -s 114.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 115.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 116.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 117.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 118.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 119.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 120.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 121.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 122.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 123.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 124.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 125.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 126.0.0.0/8 -j DENY -l
#217: 11011001 - /5 includes 216 - need 217-219 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 217.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 218.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 219.0.0.0/8 -j DENY -l
#223: 11011111 - /6 masks 220-223
ipchains -A input -i $EXTERNAL_INTERFACE -s 220.0.0.0/6 -j DENY -l
#
--------------------------------------------------------------------------
# ICMP
# To prevent denial of service attacks based on ICMP bombs, filter
# incoming Redirect (5) and outgoing Destination Unreachable (3).
# Note, however, disabling Destination Unreachable (3) is not
# advisable, as it is used to negotiate packet fragment size.
# For bi-directional ping.
# Message Types: Echo_Reply (0), Echo_Request (8)
```

```
# To prevent attacks, limit the src addresses to your ISP range.
#
# For outgoing traceroute.
# Message Types: INCOMING Dest_Unreachable (3), Time_Exceeded (11)
# default UDP base: 33434 to base+nhops-1
#
# For incoming traceroute.
# Message Types: OUTGOING Dest_Unreachable (3), Time_Exceeded (11)
# To block this, deny OUTGOING 3 and 11
# 0: echo-reply (pong)
# 3: destination-unreachable, port-unreachable, fragmentation-needed, etc.
# 4: source-quench
# 5: redirect
# 8: echo-request (ping)
# 11: time-exceeded
# 12: parameter-problem
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 0 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 3 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 4 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 11 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 12 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s 208.164.186.0/24 8 -d $IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 0 -d 208.164.186.0/24 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 3 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 4 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 8 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 12 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 11 -d 208.164.186.0/24 -j ACCEPT
#
--------------------------------------------------------------------------
# UDP INCOMING TRACEROUTE
# traceroute usually uses -S 32769:65535 -D 33434:33523
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s 208.164.186.0/24 $TRACEROUTE_SRC_PORTS \
-d $IPADDR $TRACEROUTE_DEST_PORTS -j ACCEPT -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $ANYWHERE $TRACEROUTE_SRC_PORTS \
-d $IPADDR $TRACEROUTE_DEST_PORTS -j DENY -l
#
--------------------------------------------------------------------------
# DNS server
# ----------
# DNS forwarding, caching only nameserver (53)
# ------------------------------------------
# server to server query or response
# Caching only name server only requires UDP, not TCP
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $NAMESERVER_1 53 \
-d $IPADDR 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR 53 \
-d $NAMESERVER_1 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $NAMESERVER_2 53 \
-d $IPADDR 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR 53 \
-d $NAMESERVER_2 53 -j ACCEPT
# DNS client (53)
# ---------------
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $NAMESERVER_1 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_1 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $NAMESERVER_1 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_1 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $NAMESERVER_2 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
```

```
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_2 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $NAMESERVER_2 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_2 53 -j ACCEPT
#
--------------------------------------------------------------------------
# TCP accept only on selected ports
# --------------------------------
# ----------------------------------------------------------------
# SSH server (22)
# ---------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 22 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $SSH_PORTS \
-d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 22 \
-d $ANYWHERE $SSH_PORTS -j ACCEPT
# SSH client (22)
# ---------------
# ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
# -s $ANYWHERE 22 \
# -d $IPADDR $UNPRIVPORTS -j ACCEPT
# ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
# -s $IPADDR $UNPRIVPORTS \
# -d $ANYWHERE 22 -j ACCEPT
# ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
# -s $ANYWHERE 22 \
# -d $IPADDR $SSH_PORTS -j ACCEPT
# ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
# -s $IPADDR $SSH_PORTS \
# -d $ANYWHERE 22 -j ACCEPT
# ----------------------------------------------------------------
# HTTP server (80)
# ----------------
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 80 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 80 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# ----------------------------------------------------------------
# HTTPS server (443)
# ------------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 443 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 443 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# ----------------------------------------------------------------
# SYSLOG server (514)
# ------------------
# Provides full remote logging. Using this feature you're able to
# control all syslog messages on one host.
# ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
# -s $SYSLOG_CLIENT \
# -d $IPADDR 514 -j ACCEPT
# SYSLOG client (514)
# ------------------
# ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
# -s $IPADDR 514 \
# -d $SYSLOG_SERVER 514 -j ACCEPT
# ----------------------------------------------------------------
# AUTH server (113)
# -----------------
# Reject, rather than deny, the incoming auth port. (NET-3-HOWTO)
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE \
-d $IPADDR 113 -j REJECT
# ----------------------------------------------------------------
# SMTP client (25)
# ----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $SMTP_SERVER 25 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $SMTP_SERVER 25 -j ACCEPT
```

```
# ------------------------------------------------------------------
# FTP server (20, 21)
# -------------------
# incoming request
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 21 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 21 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# PORT MODE data channel responses
#
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 20 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR 20 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# PASSIVE MODE data channel responses
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# ------------------------------------------------------------------
# OUTGOING TRACEROUTE
# -------------------
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $TRACEROUTE_SRC_PORTS \
-d $ANYWHERE $TRACEROUTE_DEST_PORTS -j ACCEPT
#
-------------------------------------------------------------------------
# Enable logging for selected denied packets
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-d $IPADDR -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-d $IPADDR $PRIVPORTS -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-d $IPADDR $UNPRIVPORTS -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 5 -d $IPADDR -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 13:255 -d $IPADDR -j DENY -l
```

```
#
------------------------------------------------------------------------
;;
stop)
echo -n "Shutting Firewalling Services: "
# Remove all existing rules belonging to this filter
ipchains -F
# Reset the default policy of the filter to accept.
ipchains -P input ACCEPT
ipchains -P output ACCEPT
ipchains -P forward ACCEPT
# Reset TCP SYN Cookie Protection to off.
echo 0 >/proc/sys/net/ipv4/tcp_syncookies
# Reset IP spoofing protection to off.
# turn on Source Address Verification
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
echo 0 > $f
done
# Reset ICMP Redirect Acceptance to on.
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 1 > $f
done
# Reset Source Routed Packets to on.
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 1 > $f
done
;;
status)
echo -n "Now do you show firewalling stats?"
;;
restart|reload)
$0 stop
$0 start
;;
*)
echo "Usage: firewall {start|stop|status|restart|reload}"
exit 1
esac
```

现在，让这个脚本文件成为可执行的，并改变它的缺省权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/firewall
```

```
[root@deep]# chown 0.0 /etc/rc.d/init.d/firewall
```

创建防火墙文件与 rc.d 的符号链接：

```
[root@deep]# chkconfig --add firewall
[root@deep]# chkconfig --level 345 firewall on
```

现在，防火墙规则就通过使用系统 V 的 init 配置好了（系统 V 的 init 负责启动所有在系统引导阶段需要运行的普通程序），并且它会在服务器重起时自动执行。

要手工停止防火墙，用命令：

```
[root@deep]# /etc/rc.d/init.d/firewall stop
```

要手工运行防火墙，用命令：

```
[root@deep]# /etc/rc.d/init.d/firewall start
```

为邮件服务器配置 /etc/rc.d/init.d/firewall 脚本文件

下面是用于我们邮件服务器的配置脚本文件。这个配置允许在回馈网卡上的所有流量，缺省情况下是 ICMP ， DNS 服务器和客户机（53），SSH 服务器（22），SMTP 服务器和客户机（25），IMAP 服务器（143）和 OUTGOING TRACEROUTE 请求。

如果你不需要我在下面文件中缺省列出的某些服务，可以用行开头加“#”来注释掉该行。如果需要那些被注释掉的服务，去掉该行开头的“#”就可以了。

请在邮件服务器上创建如下的防火墙脚本文件（用 touch /etc/rc.d/init.d/firewall）：

```
#!/bin/sh
#
#
------------------------------------------------------------------------
# Last modified by Gerhard Mourani: 02-01-2000
#
------------------------------------------------------------------------
# Copyright (C) 1997, 1998, 1999 Robert L. Ziegler
#
# Permission to use, copy, modify, and distribute this software and its
# documentation for educational, research, private and non-profit purposes,
# without fee, and without a written agreement is hereby granted.
# This software is provided as an example and basis for individual firewall
```

```
# development. This software is provided without warranty.
#
# Any material furnished by Robert L. Ziegler is furnished on an
# "as is" basis. He makes no warranties of any kind, either expressed
# or implied as to any matter including, but not limited to, warranty
# of fitness for a particular purpose, exclusivity or results obtained
# from use of the material.
#
------------------------------------------------------------------------
#
# Invoked from /etc/rc.d/init.d/firewall.
# chkconfig: - 60 95
# description: Starts and stops the IPCHAINS Firewall \
# used to provide Firewall network services.
# Source function library.
. /etc/rc.d/init.d/functions
# Source networking configuration.
. /etc/sysconfig/network
# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0
# See how we were called.
case "$1" in
start)
echo -n "Starting Firewalling Services: "
# Some definitions for easy maintenance.
#
------------------------------------------------------------------------
# EDIT THESE TO SUIT YOUR SYSTEM AND ISP.
EXTERNAL_INTERFACE="eth0" # whichever you use
LOOPBACK_INTERFACE="lo"
IPADDR="208.164.186.2"
ANYWHERE="any/0"
NAMESERVER_1="208.164.186.1" # Your primary name server
NAMESERVER_2="208.164.186.2" # Your secondary name server
SYSLOG_SERVER="mail.openarch.com" # Your syslog internal server
SYSLOG_CLIENT="208.164.168.0/24" # Your syslog internal client
LOOPBACK="127.0.0.0/8"
CLASS_A="10.0.0.0/8"
CLASS_B="172.16.0.0/12"
CLASS_C="192.168.0.0/16"
CLASS_D_MULTICAST="224.0.0.0/4"
CLASS_E_RESERVED_NET="240.0.0.0/5"
BROADCAST_SRC="0.0.0.0"
BROADCAST_DEST="255.255.255.255"
```

```
PRIVPORTS="0:1023"
UNPRIVPORTS="1024:65535"
#
------------------------------------------------------------------------
# SSH starts at 1023 and works down to 513 for
# each additional simultaneous incoming connection.
SSH_PORTS="1022:1023" # range for SSH privileged ports
# traceroute usually uses -S 32769:65535 -D 33434:33523
TRACEROUTE_SRC_PORTS="32769:65535"
TRACEROUTE_DEST_PORTS="33434:33523"
#
------------------------------------------------------------------------
# Default policy is DENY
# Explicitly accept desired INCOMING & OUTGOING connections
# Remove all existing rules belonging to this filter
ipchains -F
# Set the default policy of the filter to deny.
ipchains -P input DENY
ipchains -P output REJECT
ipchains -P forward REJECT
#
------------------------------------------------------------------------
# Enable TCP SYN Cookie Protection
echo 1 >/proc/sys/net/ipv4/tcp_syncookies
# Enable IP spoofing protection
# turn on Source Address Verification
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
echo 1 > $f
done
# Disable ICMP Redirect Acceptance
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 0 > $f
done
# Disable Source Routed Packets
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 0 > $f
done
#
------------------------------------------------------------------------
# LOOPBACK
# Unlimited traffic on the loopback interface.
ipchains -A input -i $LOOPBACK_INTERFACE -j ACCEPT
ipchains -A output -i $LOOPBACK_INTERFACE -j ACCEPT
#
```

```
------------------------------------------------------------------------
# Network Ghouls
# Deny access to jerks
# /etc/rc.d/rc.firewall.blocked contains a list of
# ipchains -A input -i $EXTERNAL_INTERFACE -s address -j DENY
# rules to block from any access.
# Refuse any connection from problem sites
#if [ -f /etc/rc.d/rc.firewall.blocked ]; then
# . /etc/rc.d/rc.firewall.blocked
#fi
#
------------------------------------------------------------------------
# SPOOFING & BAD ADDRESSES
# Refuse spoofed packets.
# Ignore blatantly illegal source addresses.
# Protect yourself from sending to bad addresses.
# Refuse spoofed packets pretending to be from the external address.
ipchains -A input -i $EXTERNAL_INTERFACE -s $IPADDR -j DENY -l
# Refuse packets claiming to be to or from a Class A private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_A -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_A -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_A -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_A -j REJECT -l
# Refuse packets claiming to be to or from a Class B private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_B -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_B -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_B -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_B -j REJECT -l
# Refuse packets claiming to be to or from a Class C private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_C -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_C -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_C -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_C -j REJECT -l
# Refuse packets claiming to be from the loopback interface
ipchains -A input -i $EXTERNAL_INTERFACE -s $LOOPBACK -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $LOOPBACK -j REJECT -l
# Refuse broadcast address SOURCE packets
ipchains -A input -i $EXTERNAL_INTERFACE -s $BROADCAST_DEST -j DENY –l
ipchains -A input -i $EXTERNAL_INTERFACE -d $BROADCAST_SRC -j DENY -l
# Refuse Class D multicast addresses (in.h) (NET-3-HOWTO)
# Multicast is illegal as a source address.
# Multicast uses UDP.
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_D_MULTICAST -j DENY -l
# Refuse Class E reserved IP addresses
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_E_RESERVED_NET -j DENY -l
# refuse addresses defined as reserved by the IANA
# 0.*.*.*, 1.*.*.*, 2.*.*.*, 5.*.*.*, 7.*.*.*, 23.*.*.*, 27.*.*.*
# 31.*.*.*, 37.*.*.*, 39.*.*.*, 41.*.*.*, 42.*.*.*, 58-60.*.*.*
# 65-95.*.*.*, 96-126.*.*.*, 197.*.*.*, 201.*.*.* (?), 217-223.*.*.*
ipchains -A input -i $EXTERNAL_INTERFACE -s 1.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 2.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 5.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 7.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 23.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 27.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 31.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 37.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 39.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 41.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 42.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 58.0.0.0/7 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 60.0.0.0/8 -j DENY -l
#65: 01000001 - /3 includes 64 - need 65-79 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 65.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 66.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 67.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 68.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 69.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 70.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 71.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 72.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 73.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 74.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 75.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 76.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 77.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 78.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 79.0.0.0/8 -j DENY -l
#80: 01010000 - /4 masks 80-95
ipchains -A input -i $EXTERNAL_INTERFACE -s 80.0.0.0/4 -j DENY -l
# 96: 01100000 - /4 makses 96-111
ipchains -A input -i $EXTERNAL_INTERFACE -s 96.0.0.0/4 -j DENY -l
#126: 01111110 - /3 includes 127 - need 112-126 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 112.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 113.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 114.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 115.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 116.0.0.0/8 -j DENY -l
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -s 117.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 118.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 119.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 120.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 121.0.0.0/8 -j DENY –l
ipchains -A input -i $EXTERNAL_INTERFACE -s 122.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 123.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 124.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 125.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 126.0.0.0/8 -j DENY -l
#217: 11011001 - /5 includes 216 - need 217-219 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 217.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 218.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 219.0.0.0/8 -j DENY -l
#223: 11011111 - /6 masks 220-223
ipchains -A input -i $EXTERNAL_INTERFACE -s 220.0.0.0/6 -j DENY -l
#
--------------------------------------------------------------------------
# ICMP
# To prevent denial of service attacks based on ICMP bombs, filter
# incoming Redirect (5) and outgoing Destination Unreachable (3).
# Note, however, disabling Destination Unreachable (3) is not
# advisable, as it is used to negotiate packet fragment size.
# For bi-directional ping.
# Message Types: Echo_Reply (0), Echo_Request (8)
# To prevent attacks, limit the src addresses to your ISP range.
#
# For outgoing traceroute.
# Message Types: INCOMING Dest_Unreachable (3), Time_Exceeded (11)
# default UDP base: 33434 to base+nhops-1
#
# For incoming traceroute.
# Message Types: OUTGOING Dest_Unreachable (3), Time_Exceeded (11)
# To block this, deny OUTGOING 3 and 11
# 0: echo-reply (pong)
# 3: destination-unreachable, port-unreachable, fragmentation-needed, etc.
# 4: source-quench
# 5: redirect
# 8: echo-request (ping)
# 11: time-exceeded
# 12: parameter-problem
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 0 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
```

```
-s $ANYWHERE 3 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 4 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 11 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 12 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s 208.164.186.0/24 8 -d $IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 0 -d 208.164.186.0/24 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 3 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 4 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 8 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 12 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 11 -d 208.164.186.0/24 -j ACCEPT
#
--------------------------------------------------------------------------
# UDP INCOMING TRACEROUTE
# traceroute usually uses -S 32769:65535 -D 33434:33523
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s 208.164.186.0/24 $TRACEROUTE_SRC_PORTS \
-d $IPADDR $TRACEROUTE_DEST_PORTS -j ACCEPT -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $ANYWHERE $TRACEROUTE_SRC_PORTS \
-d $IPADDR $TRACEROUTE_DEST_PORTS -j DENY -l
#
--------------------------------------------------------------------------
# DNS server
# ----------
# DNS: full server
# server/client to server query or response
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR 53 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# DNS client (53)
```

```
# ---------------
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $NAMESERVER_1 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_1 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $NAMESERVER_1 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_1 53 -j ACCEPT
#
-------------------------------------------------------------------------------
# TCP accept only on selected ports
# ---------------------------------
# ---------------------------------------------------------------------
# SSH server (22)
# ---------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 22 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $SSH_PORTS \
-d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 22 \
-d $ANYWHERE $SSH_PORTS -j ACCEPT
# SSH client (22)
# ---------------
# ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
# -s $ANYWHERE 22 \
# -d $IPADDR $UNPRIVPORTS -j ACCEPT
# ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
# -s $IPADDR $UNPRIVPORTS \
# -d $ANYWHERE 22 -j ACCEPT
# ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
# -s $ANYWHERE 22 \
# -d $IPADDR $SSH_PORTS -j ACCEPT
# ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
```

```
# -s $IPADDR $SSH_PORTS \
# -d $ANYWHERE 22 -j ACCEPT
# ------------------------------------------------------------------
# AUTH server (113)
# -----------------
# Reject, rather than deny, the incoming auth port. (NET-3-HOWTO)
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE \
-d $IPADDR 113 -j REJECT
# ------------------------------------------------------------------
# SYSLOG server (514)
# -----------------
# Provides full remote logging. Using this feature you're able to
# control all syslog messages on one host.
# ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
# -s $SYSLOG_CLIENT \
# -d $IPADDR 514 -j ACCEPT
# SYSLOG client (514)
# -----------------
# ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
# -s $IPADDR 514 \
# -d $SYSLOG_SERVER 514 -j ACCEPT
# ------------------------------------------------------------------
# SMTP server (25)
# ----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 25 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 25 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# SMTP client (25)
# ----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 25 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 25 -j ACCEPT
# ------------------------------------------------------------------
# IMAP server (143)
# -----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
```

```
-d $IPADDR 143 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 143 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# ------------------------------------------------------------------
# OUTGOING TRACEROUTE
# -------------------
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $TRACEROUTE_SRC_PORTS \
-d $ANYWHERE $TRACEROUTE_DEST_PORTS -j ACCEPT
#
--------------------------------------------------------------------------
# Enable logging for selected denied packets
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-d $IPADDR -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-d $IPADDR $PRIVPORTS -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-d $IPADDR $UNPRIVPORTS -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 5 -d $IPADDR -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 13:255 -d $IPADDR -j DENY -l
#
--------------------------------------------------------------------------
;;
stop)
echo -n "Shutting Firewalling Services: "
# Remove all existing rules belonging to this filter
ipchains -F
# Reset the default policy of the filter to accept.
ipchains -P input ACCEPT
ipchains -P output ACCEPT
ipchains -P forward ACCEPT
# Reset TCP SYN Cookie Protection to off.
echo 0 >/proc/sys/net/ipv4/tcp_syncookies
# Reset IP spoofing protection to off.
# turn on Source Address Verification
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
echo 0 > $f
done
# Reset ICMP Redirect Acceptance to on.
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 1 > $f
```

```
done
# Reset Source Routed Packets to on.
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 1 > $f
done
;;
status)
echo -n "Now do you show firewalling stats?"
;;
restart|reload)
$0 stop
$0 start
;;
*)
echo "Usage: firewall {start|stop|status|restart|reload}"
exit 1
esac
```

现在，让这个脚本文件成为可执行的，并改变它的缺省权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/firewall
[root@deep]# chown 0.0 /etc/rc.d/init.d/firewall
```

创建防火墙文件与 rc.d 的符号链接：

```
[root@deep]# chkconfig --add firewall
[root@deep]# chkconfig --level 345 firewall on
```

现在，防火墙规则就通过使用系统 V 的 init 配置好了（系统 V 的 init 负责启动所有在系统引导阶段需要运行的普通程序），并且它会在服务器重启时自动执行。

要手工停止防火墙，用命令：

```
[root@deep]# /etc/rc.d/init.d/firewall stop
```

要手工运行防火墙，用命令：

```
[root@deep]# /etc/rc.d/init.d/firewall start
```

为网关服务器配置 /etc/rc.d/init.d/firewall 脚本文件

下面是用于我们网关服务器的配置脚本文件。这个配置允许在回馈地址上的所有流量，缺省情况下是 ICMP ，DNS 服务器和客户机（53），SSH 服务器和客户机（22），HTTP 服务器和客户机（80），HTTPS 服务器和客户机（443），POP 客户机（110），NNTP NEWS 客户机（119），SMTP 服务器和客户机（25），IMAP 服务器（143），IRC 客户机（6667），ICQ 客户机（4000），FTP 客户机（20，21），RealAudio/QuickTime 客户机和 OUTGOING TRACEROUTE 请求。

如果你不需要在下面文件中缺省列出的某些服务，可以用行开头加“#”来注释掉该行。如果你需要某些被注释掉的服务，去掉该行开头的“#”就可以了。如果你在服务器上配置了 IP 伪装，可以去掉伪装相应服务所需模块前的注释符号，比如 ip_masq_irc.o，ip_masq_raudio.o 等等。

请在邮件服务器上创建如下的防火墙脚本文件（用 touch /etc/rc.d/init.d/firewall ）：

```
#!/bin/sh
#
#
------------------------------------------------------------------------
# Last modified by Gerhard Mourani: 02-01-2000
#
------------------------------------------------------------------------
# Copyright (C) 1997, 1998, 1999 Robert L. Ziegler
#
# Permission to use, copy, modify, and distribute this software and its
# documentation for educational, research, private and non-profit purposes,
# without fee, and without a written agreement is hereby granted.
# This software is provided as an example and basis for individual firewall
# development. This software is provided without warranty.
#
# Any material furnished by Robert L. Ziegler is furnished on an
# "as is" basis. He makes no warranties of any kind, either expressed
# or implied as to any matter including, but not limited to, warranty
# of fitness for a particular purpose, exclusivity or results obtained
# from use of the material.
#
------------------------------------------------------------------------
#
# Invoked from /etc/rc.d/init.d/firewall.
# chkconfig: - 60 95
# description: Starts and stops the IPCHAINS Firewall \
# used to provide Firewall network services.
# Source function library.
```

```
. /etc/rc.d/init.d/functions
# Source networking configuration.
. /etc/sysconfig/network
# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0
# See how we were called.
case "$1" in
start)
echo -n "Starting Firewalling Services: "
# Some definitions for easy maintenance.
#
------------------------------------------------------------------------
# EDIT THESE TO SUIT YOUR SYSTEM AND ISP.
EXTERNAL_INTERFACE="eth0" # whichever you use
LOCAL_INTERFACE_1="eth1" # whichever you use
LOOPBACK_INTERFACE="lo"
IPADDR="208.164.186.1"
LOCALNET_1="192.168.1.0/24" # whatever private range you use
ANYWHERE="any/0"
NAMESERVER_1="208.164.186.1"
NAMESERVER_2="208.164.186.2"
POP_SERVER="pop.videotron.ca" # Your pop external server
NEWS_SERVER="news.videotron.ca" # Your news external server
SYSLOG_SERVER="mail.openarch.com" # Your syslog internal server
LOOPBACK="127.0.0.0/8"
CLASS_A="10.0.0.0/8"
CLASS_B="172.16.0.0/12"
CLASS_C="192.168.0.0/16"
CLASS_D_MULTICAST="224.0.0.0/4"
CLASS_E_RESERVED_NET="240.0.0.0/5"
BROADCAST_SRC="0.0.0.0"
BROADCAST_DEST="255.255.255.255"
PRIVPORTS="0:1023"
UNPRIVPORTS="1024:65535"
#
------------------------------------------------------------------------
# SSH starts at 1023 and works down to 513 for
# each additional simultaneous incoming connection.
SSH_PORTS="1022:1023" # range for SSH privileged ports
# traceroute usually uses -S 32769:65535 -D 33434:33523
TRACEROUTE_SRC_PORTS="32769:65535"
TRACEROUTE_DEST_PORTS="33434:33523"
#
------------------------------------------------------------------------
```

```
# Default policy is DENY
# Explicitly accept desired INCOMING & OUTGOING connections
# Remove all existing rules belonging to this filter
ipchains -F
# Set the default policy of the filter to deny.
ipchains -P input DENY
ipchains -P output REJECT
ipchains -P forward REJECT
# set masquerade timeout to 10 hours for tcp connections
ipchains -M -S 36000 0 0
# Don't forward fragments. Assemble before forwarding.
ipchains -A output -f -i $LOCAL_INTERFACE_1 -j DENY
#
------------------------------------------------------------------------
# Enable TCP SYN Cookie Protection
echo 1 >/proc/sys/net/ipv4/tcp_syncookies
# Enable IP spoofing protection
# turn on Source Address Verification
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
echo 1 > $f
done
# Disable ICMP Redirect Acceptance
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 0 > $f
done
# Disable Source Routed Packets
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 0 > $f
done
# These modules are necessary to masquerade their respective services.
/sbin/modprobe ip_masq_ftp.o
/sbin/modprobe ip_masq_raudio.o ports=554,7070,7071,6970,6971
/sbin/modprobe ip_masq_irc.o
#/sbin/modprobe/ip_masq_vdolive.o
#/sbin/modprobe/ip_masq_cuseeme.o
#/sbin/modprobe/ip_masq_quake.o
#
------------------------------------------------------------------------
# LOOPBACK
# Unlimited traffic on the loopback interface.
ipchains -A input -i $LOOPBACK_INTERFACE -j ACCEPT
ipchains -A output -i $LOOPBACK_INTERFACE -j ACCEPT
#
------------------------------------------------------------------------
```

```
# Network Ghouls
# Deny access to jerks
# /etc/rc.d/rc.firewall.blocked contains a list of
# ipchains -A input -i $EXTERNAL_INTERFACE -s address -j DENY
# rules to block from any access.
# Refuse any connection from problem sites
#if [ -f /etc/rc.d/rc.firewall.blocked ]; then
# . /etc/rc.d/rc.firewall.blocked
#fi
#
--------------------------------------------------------------------------
# SPOOFING & BAD ADDRESSES
# Refuse spoofed packets.
# Ignore blatantly illegal source addresses.
# Protect yourself from sending to bad addresses.
# Refuse spoofed packets pretending to be from the external address.
ipchains -A input -i $EXTERNAL_INTERFACE -s $IPADDR -j DENY -l
# Refuse packets claiming to be to or from a Class A private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_A -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_A -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_A -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_A -j REJECT -l
# Refuse packets claiming to be to or from a Class B private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_B -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_B -j DENY –l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_B -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_B -j REJECT -l
# Refuse packets claiming to be to or from a Class C private network
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_C -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $CLASS_C -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $CLASS_C -j REJECT -l
ipchains -A output -i $EXTERNAL_INTERFACE -d $CLASS_C -j REJECT -l
# Refuse packets claiming to be from the loopback interface
ipchains -A input -i $EXTERNAL_INTERFACE -s $LOOPBACK -j DENY -l
ipchains -A output -i $EXTERNAL_INTERFACE -s $LOOPBACK -j REJECT -l
# Refuse broadcast address SOURCE packets
ipchains -A input -i $EXTERNAL_INTERFACE -s $BROADCAST_DEST -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -d $BROADCAST_SRC -j DENY -l
# Refuse Class D multicast addresses (in.h) (NET-3-HOWTO)
# Multicast is illegal as a source address.
# Multicast uses UDP.
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_D_MULTICAST -j DENY -l
# Refuse Class E reserved IP addresses
ipchains -A input -i $EXTERNAL_INTERFACE -s $CLASS_E_RESERVED_NET -j DENY -l
```

```
# refuse addresses defined as reserved by the IANA
# 0.*.*.*, 1.*.*.*, 2.*.*.*, 5.*.*.*, 7.*.*.*, 23.*.*.*, 27.*.*.*
# 31.*.*.*, 37.*.*.*, 39.*.*.*, 41.*.*.*, 42.*.*.*, 58-60.*.*.*
# 65-95.*.*.*, 96-126.*.*.*, 197.*.*.*, 201.*.*.* (?), 217-223.*.*.*
ipchains -A input -i $EXTERNAL_INTERFACE -s 1.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 2.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 5.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 7.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 23.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 27.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 31.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 37.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 39.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 41.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 42.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 58.0.0.0/7 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 60.0.0.0/8 -j DENY -l
#65: 01000001 - /3 includes 64 - need 65-79 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 65.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 66.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 67.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 68.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 69.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 70.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 71.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 72.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 73.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 74.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 75.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 76.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 77.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 78.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 79.0.0.0/8 -j DENY -l
#80: 01010000 - /4 masks 80-95
ipchains -A input -i $EXTERNAL_INTERFACE -s 80.0.0.0/4 -j DENY –l
# 96: 01100000 - /4 makses 96-111
ipchains -A input -i $EXTERNAL_INTERFACE -s 96.0.0.0/4 -j DENY -l
#126: 01111110 - /3 includes 127 - need 112-126 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 112.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 113.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 114.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 115.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 116.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 117.0.0.0/8 -j DENY -l
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -s 118.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 119.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 120.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 121.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 122.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 123.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 124.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 125.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 126.0.0.0/8 -j DENY -l
#217: 11011001 - /5 includes 216 - need 217-219 spelled out
ipchains -A input -i $EXTERNAL_INTERFACE -s 217.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 218.0.0.0/8 -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -s 219.0.0.0/8 -j DENY -l
#223: 11011111 - /6 masks 220-223
ipchains -A input -i $EXTERNAL_INTERFACE -s 220.0.0.0/6 -j DENY -l
#
--------------------------------------------------------------------------
# ICMP
# To prevent denial of service attacks based on ICMP bombs, filter
# incoming Redirect (5) and outgoing Destination Unreachable (3).
# Note, however, disabling Destination Unreachable (3) is not
# advisable, as it is used to negotiate packet fragment size.
# For bi-directional ping.
# Message Types: Echo_Reply (0), Echo_Request (8)
# To prevent attacks, limit the src addresses to your ISP range.
#
# For outgoing traceroute.
# Message Types: INCOMING Dest_Unreachable (3), Time_Exceeded (11)
# default UDP base: 33434 to base+nhops-1
#
# For incoming traceroute.
# Message Types: OUTGOING Dest_Unreachable (3), Time_Exceeded (11)
# To block this, deny OUTGOING 3 and 11
# 0: echo-reply (pong)
# 3: destination-unreachable, port-unreachable, fragmentation-needed, etc.
# 4: source-quench
# 5: redirect
# 8: echo-request (ping)
# 11: time-exceeded
# 12: parameter-problem
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 0 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 3 -d $IPADDR -j ACCEPT
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 4 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 11 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 12 -d $IPADDR -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s 208.164.186.0/24 8 -d $IPADDR -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 0 -d 208.164.186.0/24 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 3 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 4 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 8 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 12 -d $ANYWHERE -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p icmp \
-s $IPADDR 11 -d 208.164.186.0/24 -j ACCEPT
#
--------------------------------------------------------------------------
# UDP INCOMING TRACEROUTE
# traceroute usually uses -S 32769:65535 -D 33434:33523
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s 208.164.186.0/24 $TRACEROUTE_SRC_PORTS \
-d $IPADDR $TRACEROUTE_DEST_PORTS -j ACCEPT -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $ANYWHERE $TRACEROUTE_SRC_PORTS \
-d $IPADDR $TRACEROUTE_DEST_PORTS -j DENY -l
#
--------------------------------------------------------------------------
# DNS server
# ----------
# DNS: full server
# server/client to server query or response
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 53 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR 53 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# DNS client (53)
# ---------------
```

```
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $NAMESERVER_1 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_1 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $NAMESERVER_1 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_1 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $NAMESERVER_2 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_2 53 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $NAMESERVER_2 53 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $NAMESERVER_2 53 -j ACCEPT
#
------------------------------------------------------------------------------
# TCP accept only on selected ports
# --------------------------------
# ------------------------------------------------------------------
# SSH server (22)
# ---------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 22 \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE $SSH_PORTS \
-d $IPADDR 22 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR 22 \
-d $ANYWHERE $SSH_PORTS -j ACCEPT
# SSH client (22)
```

```
# ---------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 22 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 22 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 22 \
-d $IPADDR $SSH_PORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $SSH_PORTS \
-d $ANYWHERE 22 -j ACCEPT
# ------------------------------------------------------------------
# HTTP client (80)
# ----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 80 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 80 -j ACCEPT
# ------------------------------------------------------------------
# HTTPS client (443)
# ------------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 443 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 443 -j ACCEPT
# ------------------------------------------------------------------
# POP client (110)
# ----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $POP_SERVER 110 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $POP_SERVER 110 -j ACCEPT
# ------------------------------------------------------------------
# NNTP NEWS client (119)
# ----------------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
```

```
-s $NEWS_SERVER 119 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $NEWS_SERVER 119 -j ACCEPT
# ------------------------------------------------------------------
# FINGER client (79)
# ------------------
# ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
# -s $ANYWHERE 79 \
# -d $IPADDR $UNPRIVPORTS -j ACCEPT
# ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
# -s $IPADDR $UNPRIVPORTS \
# -d $ANYWHERE 79 -j ACCEPT
# ------------------------------------------------------------------
# SYSLOG client (514)
# -----------------
# ipchains -A output -i $LOCAL_INTERFACE_1 -p udp \
# -s $IPADDR 514 \
# -d $SYSLOG_SERVER 514 -j ACCEPT
# ------------------------------------------------------------------
# AUTH server (113)
# -----------------
# Reject, rather than deny, the incoming auth port. (NET-3-HOWTO)
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE \
-d $IPADDR 113 -j REJECT
# AUTH client (113)
# -----------------
# ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
# -s $ANYWHERE 113 \
# -d $IPADDR $UNPRIVPORTS -j ACCEPT
# ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
# -s $IPADDR $UNPRIVPORTS \
# -d $ANYWHERE 113 -j ACCEPT
# ------------------------------------------------------------------
# SMTP client (25)
# ----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 25 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 25 -j ACCEPT
```

```
# ------------------------------------------------------------------
# IRC client (6667)
# -----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 6667 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 6667 -j ACCEPT
# ------------------------------------------------------------------
# ICQ client (4000)
# -----------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 2000:4000 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 2000:4000 -j ACCEPT
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $ANYWHERE 4000 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 4000 -j ACCEPT
# ------------------------------------------------------------------
# FTP client (20, 21)
# -------------------
# outgoing request
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 21 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 21 -j ACCEPT
# NORMAL mode data channel
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-s $ANYWHERE 20 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
# NORMAL mode data channel responses
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 20 -j ACCEPT
# PASSIVE mode data channel creation
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
```

```
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# PASSIVE mode data channel responses
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
# ------------------------------------------------------------------
# RealAudio / QuickTime client
# ---------------------------
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 554 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 554 -j ACCEPT
# TCP is a more secure method: 7070:7071
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
-s $ANYWHERE 7070:7071 \
-d $IPADDR $UNPRIVPORTS -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE 7070:7071 -j ACCEPT
# UDP is the preferred method: 6970:6999
# For LAN machines, UDP requires the RealAudio masquerading module and
# the ipmasqadm third-party software.
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-s $ANYWHERE $UNPRIVPORTS \
-d $IPADDR 6970:6999 -j ACCEPT
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
-s $IPADDR $UNPRIVPORTS \
-d $ANYWHERE $UNPRIVPORTS -j ACCEPT
# ------------------------------------------------------------------
# WHOIS client (43)
# -----------------
# ipchains -A input -i $EXTERNAL_INTERFACE -p tcp ! -y \
# -s $ANYWHERE 43 \
# -d $IPADDR $UNPRIVPORTS -j ACCEPT
# ipchains -A output -i $EXTERNAL_INTERFACE -p tcp \
# -s $IPADDR $UNPRIVPORTS \
# -d $ANYWHERE 43 -j ACCEPT
# ------------------------------------------------------------------
# OUTGOING TRACEROUTE
# -------------------
ipchains -A output -i $EXTERNAL_INTERFACE -p udp \
```

```
-s $IPADDR $TRACEROUTE_SRC_PORTS \
-d $ANYWHERE $TRACEROUTE_DEST_PORTS -j ACCEPT
#
# ------------------------------------------------------------------------
# Unlimited traffic within the local network.
# All internal machines have access to the firewall machine.
ipchains -A input -i $LOCAL_INTERFACE_1 -s $LOCALNET_1 -j ACCEPT
ipchains -A output -i $LOCAL_INTERFACE_1 -d $LOCALNET_1 -j ACCEPT
#
# ------------------------------------------------------------------------
# Masquerade internal traffic.
# All internal traffic is masqueraded externally.
ipchains -A forward -i $EXTERNAL_INTERFACE -s $LOCALNET_1 -j MASQ
#
# ------------------------------------------------------------------------
# Enable logging for selected denied packets
ipchains -A input -i $EXTERNAL_INTERFACE -p tcp \
-d $IPADDR -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-d $IPADDR $PRIVPORTS -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p udp \
-d $IPADDR $UNPRIVPORTS -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 5 -d $IPADDR -j DENY -l
ipchains -A input -i $EXTERNAL_INTERFACE -p icmp \
-s $ANYWHERE 13:255 -d $IPADDR -j DENY -l
#
# ------------------------------------------------------------------------
;;
stop)
echo -n "Shutting Firewalling Services: "
# Remove all existing rules belonging to this filter
ipchains -F
# Reset the default policy of the filter to accept.
ipchains -P input ACCEPT
ipchains -P output ACCEPT
ipchains -P forward ACCEPT
# Reset TCP SYN Cookie Protection to off.
echo 0 >/proc/sys/net/ipv4/tcp_syncookies
# Reset IP spoofing protection to off.
# turn on Source Address Verification
for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
echo 0 > $f
done
```

```
# Reset ICMP Redirect Acceptance to on.
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 1 > $f
done
# Reset Source Routed Packets to on.
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 1 > $f
done
;;
status)
echo -n "Now do you show firewalling stats?"
;;
restart|reload)
$0 stop
$0 start
;;
*)
echo "Usage: firewall {start|stop|status|restart|reload}"
exit 1
esac
```

现在，让这个脚本文件成为可执行的，并改变它的缺省权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/firewall
[root@deep]# chown 0.0 /etc/rc.d/init.d/firewall
```

创建防火墙文件与 rc.d 的符号链接：

```
[root@deep]# chkconfig --add firewall
[root@deep]# chkconfig --level 345 firewall on
```

现在，你的防火墙规则就通过使用系统 V 的 init 配置好了（系统 V 的 init 负责启动所有在系统引导阶段需要运行的普通程序），并且它会在服务器重起是自动执行。

要手工停止防火墙，用命令：

```
[root@deep]# /etc/rc.d/init.d/firewall stop
```

要手工运行防火墙，用命令：

```
[root@deep]# /etc/rc.d/init.d/firewall start
```

拒绝一些地址的访问

在某些时候，你知道这样一个地址，那就是要拒绝从该地址到你服务器的任何访问。这可以通过在“/etc/rc.d/”目录下创建一个叫“rc.firewall.blocked”文件，并且把防火墙文件中下面几行前的注释去掉来实现。

编辑 firewall 脚本文件（vi /etc/rc.d/init.d/firewall），并去掉下面几行前的注释符号：

```
if [ -f /etc/rc.d/rc.firewall.blocked ]; then
        ./etc/rc.d/rc.firewall.blocked
fi
```

创建“rc.firewall.blocked”文件（touch /etc/rc.d/rc.firewall.blocked），并且在文件中加进你所想阻塞的 IP 地址：

例如：

```
204.254.45.9
187.231.11.5
```

详细文档

如果需要更详细信息，下面有几篇 man 手册可以阅读：

```
$ ipchains(8)              - IP firewall administration
$ ipchains –restore (8)    - restore IP firewall chains from stdin
$ ipchains –save (8)       - save IP firewall chains to stdout
```

IPCHAINS 管理工具

下面列出的命令是我们需要经常使用的，你可以通过 man 手册和其它文档去了解它们的详细信息。

ipchains

ipchains 是用来建立，维护并随时检查在内核中的防火墙规定。这些规定分成 4 种类型： IP 输入链，IP 输出链，IP 转发链和用户定义链。

要列出所选择的链的所有规定，可以使用下面的命令：

```
[root@deep]# ipchains –L
```

这个命令会列出所选择链的所有规定，如果没有选择链，所有的链就都会列出。

要列出所选择的链的所有输入规定，可以使用下面的命令：

```
[root@deep]# ipchains -L input
```

这个命令会列出所选择链的所有输入规定。

要列出所选择的链的所有输出规定，可以使用下面的命令：

```
[root@deep]# ipchains -L output
```

这个命令会列出所选择链的所有输出规定。

要列出所选择的链的所有转发规定，可以使用下面的命令：

```
[root@deep]# ipchains -L forward
```

这个命令会列出所选择链的所有转发规定。这只在配置了 IP 伪装的时候有用。

要列出所选择的链的所有伪装规定，可以使用下面的命令：

```
[root@deep]# ipchains -ML
```

这个命令允许查看所有当前伪装后的连接，这只在配置了 IP 伪装的时候有用。

要列出所选择的链的所有规定以数字输出，可以使用下面的命令：

```
[root@deep]# ipchains -nL
```

这个命令会以数字的形式列出所有规定。IP 地址和端口号会以数字的形式打印出来。但在缺省状态下，它们是以主机名，网络名或服务名的形式显示的。

第五部分：与软件相关的参考资料

第八章

编译器的功用

概述

在我们解释如何编译和安装安全和优化的服务器软件之前，有必要先知道一下用什么命令和程序来完成这项任务。首先，必须保证在系统中已经安装了必要的软件包，能够进行编译工作。这些软件包必须安装在服务器上，否则将无法编译程序。

必要的一些软件包

为了能在服务器上编译软件，在重新编译完内核之后，必须安装下面的软件包。这些软件包在 RedHat 6.1 第一张光盘的 RedHat/RPMS 目录下。

用下面的命令进入软件包所在的目录：

```
[root@deep]# mount /dev/cdrom /mnt/cdrom/
[root@deep]# cd /mnt/cdrom/RedHat/RPMS/
```

需要安装的软件是：

```
autoconf-2.13-5.noarch.rpm
m4-1.4-12.i386.rpm
automake-1.4-5.noarch.rpm
dev86-0.14.9-1.i386.rpm
bison-1.28-1.i386.rpm
byacc-1.9-11.i386.rpm
cdecl-2.5-9.i386.rpm
cpp-1.1.2-24.i386.rpm
cproto-4.6-2.i386.rpm
ctags-3.2-1.i386.rpm
egcs-1.1.2-24.i386.rpm
ElectricFence-2.1-1.i386.rpm
flex-2.5.4a-7.i386.rpm
gdb-4.18-4.i386.rpm
glibc-devel-2.1.2-11.i386.rpm
```

```
make-3.77-6.i386.rpm
patch-2.5-9.i386.rpm
```

把 RPM 软件包安装到系统中的命令是：

```
[root@deep]# rpm -Uvh foo-1.0-2.i386.rpm
```

检验 RPM 软件包是否已经安装到系统中的命令是：

```
[root@deep]# rpm -q foo
```

一旦安装和编译完所有应该在服务器上安装的软件之后，应该把上面的软件包都卸掉（编译器、函数库，等）。这可以保证没有经过授权的用户不能在服务器上编译程序。

还要把“rpm”程序移到一个安全的地方，如：软盘，其原因是让没有经过授权的用户不能随便安装软件。假定一个怀有恶意的人想在服务器上编译程序，而且已经知道服务器上没装编译器。他就会试图用 rpm 命令把所有上面列出来的软件包安装到服务器中去。当他发现 rpm 命令也不能用的时候，一定会感到很吃惊而又无可奈何。当然，今后如果你想在服务器上安装新的软件也要用到 rpm 程序，但是所要做的不过是把 rpm 程序从软盘上拷回原来的地方。

用下面的命令把 rpm 程序移动到软盘上：

```
[root@deep]# mount /dev/fd0 /mnt/floppy/
[root@deep]# mv /bin/rpm /mnt/floppy
[root@deep]# umount /mnt/floppy/
```

用下面的命令把 rpm 程序移回原来的地方：

```
[root@deep]# mount /dev/fd0 /mnt/floppy/
[root@deep]# cp /mnt/floppy/rpm /bin/
[root@deep]# umount /mnt/floppy/
```

注意：不要用 rpm 命令把 rpm 软件包完全卸载掉，否则以后就再也没有办法安装它了（安装 rpm 软件包也要用 rpm 程序）。

为什么选择 tarballs

RedHat Linux 是以 RPM 文件的形式发行的。RPM 文件，也就是 RedHat Linux 系统中所谓“软件包”。用 RPM 软件包的形式来发行软件的优势是可以很方便的安装、升级、查询和卸载。然而，在 Unix 世界，用“tarballs”（.tar.gz 文件）发行软

件包是事实上的标准。tarballs 是用“tar”就可以处理的简单文件格式。但是安装“tarballs”比安装 RPM 软件包麻烦得多了。那么为什么我们还选择“tarballs”呢？

1. 因为许多开发者都是先用“tarballs”来发行软件的，所以一般要等几个星期他们才会把最新的软件转成 RPM 包。

2. 当开发者发行新的 RPM 软件包的时候，他们并不知道你要什么或不要什么，所以编译的时候用通用的选项来满足多数人的需要。

3. RPM 包通常没有为特殊的处理器优化。象 RedHat Linux 这样的公司发行的 RPM 软件包是基于标准 PC，也就是以 i386 的标准来编译软件，这样程序就可以在各种计算机上运行。

4. 有时你会下载并安装别人已经做好的现成的 RPM 软件包。但是这有可能导致冲突，因为每个人的软硬件环境都是不一样的，而且还有可能产生安全隐患或其它问题。

在系统中编译软件

简单地说，程序就是计算机可以执行的东西。一个人用自己可以理解的语言写出程序的“源代码”，这种语言有可能是 C 或其它语言。编译器把“源代码”转换成处理器（386、486，等）可以执行的二进制文件。现在 Linux 系统中可执行的二进制文件的格式通常 elf。程序员用编译器编译源代码以得到二进制程序。编译通不过，或是通过了，但是程序却并不能正常地运行，这是很常见的事，所以有一大半的编程时间是用来跟踪并查找错误（debugging）。

本书中用到的有关如何编译源代码的名词和术语包括但不限于：

单独编译

只用单个文件编译的程序很少见。通常情况下，有多个文件（例如：*.c）要编译成目标文件（*.o），然后再链接成可执行文件。编译器通过调用链接器（“ld”程序）来完成链接工作。

Makefile

Makefile 是用来保证编译的一致性（每次都用同样的方法编译程序），而且还可以提高编译的速度。大型的程序都要用很长的时间——几十分钟才能完成编译。“make”根据 Makefile 里定义的“相关性”来决定程序的哪些部分需要重新编译。如果五十个源程序文件中有一个发生变化，那么只要重新编译一个文件然后再重新链接就行了，而不要全部重新编译。请注意 Makefile 的格式要求有些行必须用 TAB 字符作为起始字符，而不是空格。

函数库

不仅可以用目标文件（*.o）链接成程序，还可以用函数库（目标文件的集合）链接。有时可能要链接系统的函数库（如：-lm 数学函数库，用于数学运算的 C 程序必须用到这个函数库）。有两种链接库函数的方式：静态（函数的目标代码被编译到可执行文件中）和动态（程序运行的时候才把函数动态载入）。编译器的手册中有很大一部分讨论这两种方式各自的优缺点。

补丁

以前补丁是用来修正可执行文件的，而不必重新编译程序。现在已经不用这种方法了，通常是给源代码打“补丁”，也就是改变一小部分的源代码。Larry Warry 的“patch”程序就是用于这个目的。程序版本的更新现在可以用打补丁这种方式，而用不着每次都发行不同的软件包。

编译和链接中出现的错误

有很多原因会导致这些错误，如：输入错误、不小心漏掉以及语法错误等等。请注意检查源程序中有没有包含需要调用的函数的头文件。无法引用的符号错误（unreferenced symbols）通常是链接时出现的问题，所以要检查一下系统中有没有安装必要的函数库和一些工具。

调试

调试涉及很多方面的知识。有时候可以在源代码中加入调试的语句，这样可以知道程序运行的状态。为了避免一下子有太多的输出，也可以在循环语句中每循环三次输出一下。查看变量在模块之间能否正确地传递也有助于问题的解决。还有就是要熟悉调试工具。

编译和安装软件

在下面的章节中我们可能会用其它一些不同的命令来编译和安装服务器软件。这一节介绍的命令是一些通用的命令，是 Unix 兼容的而且可以在各种各样的 Unix 变种下编译和安装软件。

在服务器上编译和安装“tarballs”软件包的过程如下：

首先必须到可靠的站点下载“tarball”。

下载完之后，转到“/var/tmp”目录（其它的目录也可以），以 root 身份用带参数的“tar”命令解压软件包。例如：

```
[root@deep]# tar xzpf foo.tar.gz
```

上面这个命令解压“foo.tar.gz”这个软件包。

- “x”选项告诉 tar 解压压缩文件
- “z”选项告诉 tar 压缩文件是用 gzip 压缩的
- “p”选项告诉 tar 保留文件的权限
- “f”选项告诉 tar 下一个参数是文件名

“tarball”解压之后，就会在相应的目录中找到“README”和“INSTALL”文件。读完这些文件之后就知道如何编译和安装软件了。很有可能会运行下面的命令：

```
./configure
make
make install
```

“./configure”命令对软件进行配置以保证系统中有进行编译所必要的函数库，“make”把所有的源代码文件编译成二进制文件，“make install”把二进制文件和其它必要的文件安装到相应的目录。其它有可能见到的命令还有：

```
make depend
strip
chown
```

“make depend”命令在不同的文件之间建立相关性。“strip”命令从目标文件中清除符号信息。这样最后生成的二进制文件就会比较小，也可以提高程序的性能。“chown”命令给二进制文件设定合适的所有者和组的权限。

注意：在后面服务器软件安装的相关章节会介绍和解释更多的命令。

所有第九章和第十章列出来的软件，可以根据自己的需要、这台服务器要完成什么样的任务以及要求它在 Intranet/Internet 上扮演什么样的角色，来选择安装。为了保证远程管理的安全有可能会用 ssh 替代 telnet。另外还有可能安装 Tripware 来帮助系统管理员监控系统中文件的变化。

第九章

系统安全软件

Linux sXid

概述

sXid 是一个 SUID/SGID 程序的集成监控程序，被设计成用 cron 定期运行。它可以跟踪 SUID/SGID 文件或目录的变化。如果发现有什么变化，如：出现了新的 SUID/SGID 的文件或目录，原来不是 SUID 的程序现在变成是 SUID 了，或者 SUID/SGID 的程序的权限或其它出现了变化，sXid 程序就会用命令行或者 email 报告这些变化。sXid 会自动地完成查看 SUID/SGID 程序的任务并把结果报告给你。只要安装上之后，就可以不用管它了，它会自动为你完成一切工作。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

sXid 的版本是 4.0.1。

软件包的来源

sXid 的 FTP 站点：ftp://marcus.seva.net/pub/sxid/。

下载 sxid_4_0_1_tar.gz 这个软件包。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >sxid1”，在编译和安装完软件之后运行命令“find /* > sxid2”，最后用命令“diff sxid1 sxid2 > sxid”找出变化。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp sxid_version_tar.gz /var/tmp/
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf sxid_version_tar.gz
```

编译和优化

转到 sXid 的新目录下，运行下面的命令：

```
make install
```

这个命令会自动地配置软件，保证使用正确的函数库，并把所有的源程序都编译成可执行的二进制文件，接着，安装这些二进制文件并把其它运行这个软件所需要的文件安装到相应的目录下。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf sxid-version/ sxid_version_tar.gz
```

“rm”命令删除所有的编译和安装 sXid 软件所需的源文件，并把 sXid 软件的压缩包删除。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必

要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 sXid，必须在“/etc”目录下创建“sxid.conf”或者把现成的“sxid.conf”文件拷贝到“/etc”目录下。可以把“floppy.tgz”解压之后，找到“sxid.conf”文件，并拷贝到“/etc”目录下，或者用拷贝粘贴的方法从本书中直接粘贴出“sxid.conf”文件。

配置 /etc/sxid.conf 文件

在 sXid 的配置文件“/etc/sxid.conf”中，可以设置 sXid 程序的一些选项和运行方式。这些都很简单而且在相应的地方都注释得很清楚。

第一步：根据需要编辑“sxid.conf”文件（vi /etc/sxid.conf）。

```
# Configuration file for sXid
# Note that all directories must be absolute with no trailing /'s
# Where to begin our file search
SEARCH = "/"
# Which subdirectories to exclude from searching
EXCLUDE = "/proc /mnt /cdrom /floppy"
# Who to send reports to
EMAIL = "root"
# Always send reports, even when there are no changes?
ALWAYS_NOTIFY = "no"
# Where to keep interim logs. This will rotate 'x' number of
# times based on KEEP_LOGS below
LOG_FILE = "/var/log/sxid.log"
# How many logs to keep
KEEP_LOGS = "5"
# Rotate the logs even when there are no changes?
ALWAYS_ROTATE = "no"
# Directories where +s is forbidden (these are searched
# even if not explicitly in SEARCH), EXCLUDE rules apply
FORBIDDEN = "/home /tmp"
# Remove (-s) files found in forbidden directories?
ENFORCE = "yes"
# This implies ALWAYS_NOTIFY. It will send a full list of
# entries along with the changes
LISTALL = "no"
# Ignore entries for directories in these paths
# (this means that only files will be recorded, you
# can effectively ignore all directory entries by
# setting this to "/"). The default is /home since
```

```
# some systems have /home g+s.
IGNORE_DIRS = "/home"
# File that contains a list of (each on it's own line)
# of other files that sxid should monitor. This is useful
# for files that aren't +s, but relate to system
# integrity (tcpd, inetd, apache...).
# EXTRA_LIST = "/etc/sxid.list"
# Mail program. This changes the default compiled in
# mailer for reports. You only need this if you have changed
# it's location and don't want to recompile sxid.
# MAIL_PROG = "/usr/bin/mail"
```

第二步：在“root”的 crontab（用来定期地完成一些任务）中加入一项，把 sXid 作为一项定期工作。

sXid 将被 crond daemon 运行。它将跟踪 SUID/SGID 文件或目录的任何变化，比如：是否新增了 SUID/SGID 的文件、SUID/SGID 文件的访问权限或内容是否发生了变化。而且，sXid 会把这些变化报告出来。把 sXid 加入定期执行的任务中，用下面的命令编辑 crontab：

```
[root@deep]# crontab -e
```

加入这几行：

```
# Sample crontab entry to run every day at 4am
0 4 * * * /usr/bin/sxid
```

更多的资料

如果想查找详细的资料可以用 man 命令查帮助页，读取相关信息：

$ man sxid.conf (5) – sXid 的配置设定

$ man sxid (1) – 如何查找 SUID/SGID 文件和目录。

sXid 的管理工具

sXid 是用来做为定期执行的任务（cronjob）运行的。每天必须运行一次，如果访问量大的服务器每天要运行两次。也可以手工运行它。

用下面的命令，手工运行 sXid：

```
[root@deep]# sxid -k
```

其输出可能是这样的：

```
sXid Vers : 4.0.1
Check run : Wed Dec 29 12:40:32 1999
This host : mail.openarch.com
Spotcheck : /home/admin
Excluding : /proc /mnt /cdrom /floppy
Ignore Dirs: /home
Forbidden : /home /tmp
No changes found
```

这个命令检查当前目录及其所有子目录。不生成日志文件，也不用 email 报告结果，只是在标准输出（stdout）显示结果。

安装到系统中的文件

```
> /etc/sxid.conf
> /usr/bin/sxid
> /usr/man/man1/sxid.1
> /usr/man/man5/sxid.conf.5
```

Linux SSH1 Client/Server

概述

SSH（Secure Shell）是象 rlogin 和 rsh 这样的不安全的远程登录程序的替代程序，是安全的远程登录程序。根据 SSH 的官方站点的介绍，它不仅是安全的远程登录程序，而且是远程管理网络主机的一次重大的变革。它是一个强大而易用的程序，用很强的加密算法加密需要保密的数据，包括口令、二进制文件和管理命令，以保证数据的安全传输。SSH1 最大的好处是对最终用户和商业用户都是完全免费的。

为了有更好的安全性，我们把 sshd1 daemon 配置成支持 tcp-wrappers（inetd 超级服务器）。SSH2 原来是免费的，但是现在用的是商用的许可协议。尽管我们都会介绍两种版本的 SSH 的配置，但是还是建议使用 SSH1（免费）而不用 SSH2（商用）。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

ssh1 的版本是 1.2.27。

软件包的来源

SSH1 的主页：http://www.ssh.fi/。

下载：ssh-1.2.27.tar.gz。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >ssh1”，在编译和安装完软件之后运行命令“find /* > ssh2”，最后用命令“diff ssh1 ssh2 > ssh”找出变化。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp ssh-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf ssh-version.tar.gz
```

编译和优化

转到 ssh1 的新目录下，先设置编译器的编译参数：

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--with-etcdir=/etc/ssh \
--without-idea \
--enable-warnings \
--without-rsh \
--with-libwrap \
--disable-server-port-forwardings \
--disable-client-port-forwardings \
--disable-server-x11-forwarding \
--disable-client-x11-forwarding \
--disable-suid-ssh
```

这些告诉编译器如何编译 SSH1：

- 避免商用的版权问题
- 在使用 gcc/egcs 时，使-Wall 选项生效
- 在任何情况下不要用 rsh 命令
- 编译的时候支持 libwrap（tcp_wrappers）
- 禁止服务器上所有的端口（X11 除外）转发（port forwarding）
- 禁止客户机上所有的端口（X11 除外）转发（port forwarding）
- 禁止服务器上 X11 转发

- 禁止客户机上 X11 转发
- 安装 ssh 时不加上 SUID 位

编译用下面的命令：

```
[root@deep]# make clean
[root@deep]# make
[root@deep]# make install
```

“make clean”，删除所有以前编译的时候留下来的中间文件，以避免任何错误，接着“make”命令把源程序编译成可执行的二进制程序，最后“make install”安装软件。

清除不必要的文件

用下面的命令删除不必要的文件：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf ssh1-version/ ssh-version.tar.gz
```

“rm”命令删除所有编译和安装 SSH1 所需要的源程序，并且把 SSH1 软件的压缩包删除掉。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 SSH1 Client/Server，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“sshd_config”文件拷贝到“/etc/ssh”目录下
- 把“ssh_config”文件拷贝到“/etc/ssh”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到“/etc/ssh”目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

配置 /etc/ssh/ssh_config 文件

“/etc/ssh/ssh_config”是 ssh1 的配置文件，允许设置一些选项来改变客户端程序的运行方式。这个文件的每一行包含“关键词－值”的匹配，其中“关键词”是忽略大小写的。下面列出来的是最重要的关键词，用 man 命令查看帮助页（ssh (1)）可以得到详细的列表。

编辑“ssh_config”文件（vi /etc/ssh/ssh_config），加入：

```
# Site-wide defaults for various options
Host *
ForwardAgent no
ForwardX11 no
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
TISAuthentication no
PasswordAuthentication yes
FallBackToRsh no
UseRsh no
BatchMode no
Compression yes
StrictHostKeyChecking no
IdentityFile ~/.ssh/identity
Port 22
KeepAlive yes
Cipher blowfish
EscapeChar ~
```

下面逐行说明上面的选项设置：

Host *

选项“Host”只对能够匹配后面字串的计算机有效。“*”表示所有的计算机。

ForwardAgent no

“ForwardAgent”设置连接是否经过验证代理（如果存在）转发给远程计算机。

ForwardX11 no

“ForwardX11”设置 X11 连接是否被自动重定向到安全的通道和显示集（DISPLAY set）。

RhostsAuthentication no

“RhostsAuthentication”设置是否使用基于 rhosts 的安全验证。

RhostsRSAAuthentication no

“RhostsRSAAuthentication”设置是否使用用 RSA 算法的基于 rhosts 的安全验证。

RSAAuthentication yes

“RSAAuthentication”设置是否使用 RSA 算法进行安全验证。

TISAuthentication no

“TISAuthentication”设置是否使用 TIS 安全验证。

PasswordAuthentication yes

“PasswordAuthentication”设置是否使用口令验证。

FallBackToRsh no

“FallBackToRsh”设置如果用 ssh 连接出现错误是否自动使用 rsh。

UseRsh no

“UseRsh”设置是否在这台计算机上使用“rlogin/rsh”。

BatchMode no

“BatchMode”如果设为“yes”，passphrase/password（交互式输入口令）的提示将被禁止。当不能交互式输入口令的时候，这个选项对脚本文件和批处理任务十分有用。

Compression yes

“Compression”设置是否使用压缩。压缩能够提高通信速度。

StrictHostKeyChecking no

“StrictHostKeyChecking”如果设置成“yes”，ssh 就不会自动把计算机的密匙加入“$HOME/.ssh/known_hosts”文件，并且一旦计算机的密匙发生了变化，就拒绝连接。

IdentityFile ~/.ssh/identity

“IdentityFile”设置从哪个文件读取用户的 RSA 安全验证标识。

Port 22

“Port”设置连接到远程主机的端口。

KeepAlive yes

“KeepAlive”设置系统是否发送“keep alive”的消息，如果发送“keep alive”的消息，如果连接失败或者有一台计算机死机，就很容易被注意到。

Cipher blowfish

“Cipher”设置加密用的密码。

EscapeChar ~

“EscapeChar”设置 escape 字符。

配置 /etc/ssh/sshd_config 文件

“/etc/ssh/sshd_config”是 sshd1 的配置文件，允许设置选项改变这个 daemon 的运行。这个文件的每一行包含“关键词－值”的匹配，其中“关键词”是忽略大小写的。下面列出来的是最重要的关键词，用 man 命令查看帮助页（sshd (8)）可以得到详细的列表。

编辑“sshd_config”文件（vi /etc/ssh/sshd_config），加入：

```
# This is ssh server systemwide configuration file.
Port 22
ListenAddress 192.168.1.1
HostKey /etc/ssh/ssh_host_key
RandomSeed /etc/ssh/ssh_random_seed
ServerKeyBits 1024
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin no
IgnoreRhosts yes
StrictModes yes
QuietMode no
X11Forwarding no
FascistLogging no
PrintMotd yes
KeepAlive yes
SyslogFacility AUTH
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication yes
PermitEmptyPasswords no
AllowUsers admin
AllowHosts 192.168.1.4
```

下面逐行说明上面的选项设置：

Port 22

“Port”设置 sshd 监听的端口号。

ListenAddress 192.168.1.1

“ListenAddress”设置 sshd 服务器绑定的 IP 地址。

HostKey /etc/ssh/ssh_host_key

“HostKey”设置包含计算机私人密匙的文件。

RandomSeed /etc/ssh/ssh_random_seed

“RandomSeed”为服务器设置随机数的种子，这个文件定期地被创建和更新。

ServerKeyBits 1024

“ServerKeyBits”定义服务器密匙的位数。

LoginGraceTime 600

“LoginGraceTime”设置如果用户不能成功登录，在切断连接之前服务器需要等待的时间（以秒为单位）。

KeyRegenerationInterval 3600

“KeyRegenerationInterval”设置在多少秒之后自动重新生成服务器的密匙（如果使用密匙）。重新生成密匙是为了防止用盗用的密匙解密被截获的信息。

PermitRootLogin no

“PermitRootLogin”设置 root 能不能用 ssh 登录。这个选项一定不要设成“yes”。

IgnoreRhosts yes

“IgnoreRhosts”设置验证的时候是否使用“rhosts”和“shosts”文件。

StrictModes yes

“StrictModes”设置 ssh 在接收登录请求之前是否检查用户家目录和 rhosts 文件的权限和所有权。这通常是必要的，因为新手经常会把自己的目录和文件设成任何人都有写权限。

QuietMode no

“QuietMode”设置系统是否“安静”地运行，在“安静”模式下，系统日志的文件除了严重错误之外，不记录其它东西。

X11Forwarding no

“X11Forwarding”设置是否允许 X11 转发。

FascistLogging no

“FascistLogging”设置是否使用详细登录。详细登录会侵犯用户的隐私，所以不推荐。

PrintMotd yes

“PrintMotd”设置 sshd 是否在用户登录的时候显示“/etc/motd”中的信息。

KeepAlive yes

“KeepAlive”设置系统是否发送“keep alive”的消息，如果发送“keep alive”的消息，如果连接失败或者有一台计算机死机，就很容易被注意到。

SyslogFacility AUTH

“SyslogFacility”设置在记录来自 sshd 的消息的时候，是否给出“facility code”。

RhostsAuthentication no

“RhostsAuthentication”设置只用 rhosts 或“/etc/hosts.equiv”进行安全验证是否已经足够了。

RhostsRSAAuthentication no

“RhostsRSA”设置是否允许用 rhosts 或“/etc/hosts.equiv”加上 RSA 进行安全验证。

RSAAuthentication yes

“RSAAuthentication”设置是否允许只有 RSA 安全验证。

PasswordAuthentication yes

“PasswordAuthentication”设置是否允许口令验证。

PermitEmptyPasswords no

“PermitEmptyPasswords”设置是否允许用口令为空的帐号登录。

AllowUsers admin

“AllowUsers”的后面可以跟着任意的数量的用户名的匹配串（patterns）或 user@host 这样的匹配串，这些字符串用空格隔开。主机名可以是 DNS 名或 IP 地址。

AllowHosts 192.168.1.4

“AllowHosts” 的后面可以跟着任意的数量的主机名的匹配串（patterns），中间用空格隔开。如果设定了，只能用“.shosts”、“.rhosts”和“/etc/hosts.quiv”中出现的主机名和匹配串（patterns）进行匹配。客户机的主机名要和对应成上面这些文件中出现的主机名。如果主机名不能对应，要用 IP 地址。

配置 sshd 使其使用 TCP-WRAPPERS　inetd 超级服务器

TCP-WRAPPERS 用来启动和停止 sshd1 服务。当 inetd 运行的时候，它会从配置文件（默认为“/etc/inetd.conf”）中读入配置信息。在配置文件中每一行的不同项是用 TAB 或空格分开。

第一步

编辑“inetd.conf”文件（vi /etc/inetd.conf）并加入这一行：

```
ssh stream tcp nowait root /usr/sbin/tcpd sshd -i
```

注意：“-i”参数很重要，它说明 sshd 是被 inetd 运行的。在加入这一行后，通过发送一个 SIGHUP 信号（killall –HUP inetd）来更新“inetd.conf”文件。

```
[root@deep /root]# killall -HUP inetd
```

第二步

编辑“hosts.allow”文件（vi /etc/hosts.allow）并加入这一行：

```
sshd: 192.168.1.4 win.openarch.com
```

这一行表示 IP 地址为“192.168.1.4”，主机名为“win.openarch.com”的计算机允许用 ssh 访问服务器。

下面这些“daemon”字符串（用于 TCP-WRAPPERS）被 sshd1 使用：

```
sshdfwd-X11 (允许/禁止 X11 转发).
sshdfwd-<port-number> (TCP 转发).
sshdfwd-<port-name> (port-name 在/etc/services 中定义。用于 TCP 转发).
```

注意：如果准备使用 ssh，一定要用在所有的服务器上。如果十台安全的服务器和一台不安全的服务器配在一起，也谈不上什么安全性。

更多的资料

如果想查找详细的资料可以用 man 命令查帮助页，读取相关信息：

$ man ssh-add1 (1) － 给安全验证的代理加上标识

$ man ssh-agent1 (1) － 安全验证的代理

$ man ssh-keygen1 (1) － 生成安全验证的密匙

$ man ssh1 (1) － SSH 的客户端程序（远程登录）

$ man sshd1 (8) － SSH 的 daemon

SSH1 每用户配置

第一步

为本地服务器创建私有和公用密匙，执行下面的命令：

```
[root@deep]# su username
[username@deep]$ ssh-keygen1
```

举个例子，显示出来的结果可能是：

```
Initializing random number generator...
Generating p: .............................++ (distance 430)
Generating q: ......................++ (distance 456)
Computing the keys...
Testing the keys...
Key generation complete.
Enter file in which to save the key (/home/username/.ssh/identity): 【按下回车键】
Enter passphrase:
Enter the same passphrase again:
Your identification has been saved in /home/username/.ssh/identity.
Your public key is:
1024 37
149377575112519555336911203184772938622900493947151365111458061088700017643784946768 31
297577843158532272361206100623146044053648718436774842332409194184809889078609971752 44
469775896471277570307287799737085699930170431415635363330688889440381784616085924838 44
```

```
590202154102756903055846534063365635584899765402181
username@deep.openarch.com
Your public key has been saved in /home/username/.ssh/identity.pub
```

注意：如果有多个帐号需要为每个帐号创建一个密匙。

你可能要为下面的服务器创建密匙：

- Mail 服务器
- Web 服务器
- 网关服务器

这允许对这些服务器进行有限的访问，例如，不允许用 Mail 服务器的帐号访问 Web 服务器或网关服务器。这样可以增加整体的安全性，即使因为某种原因有一个密匙被泄密了，也不会影响到其它的服务器。

第二步

把本机的公用密匙（identity.pub）拷贝到远程主机的“/home/username/.ssh”目录下，例如，使用“authorized_keys”这个名字。

注意：拷贝文件的一个方法使用 ftp 命令，另一个办法是把公用密匙用 email（包含“~/.ssh/identity.pub”文件的内容）发给系统管理员。

如果还是不能访问远程计算机，那么就需要查看下面这些文件的许可权限了：

- 家目录
- “~/.ssh”目录
- “~/.ssh/authorized_keys”文件

只有文件的主人才有写权限。下面是一个例子，列出这些文件的权限，可以参考一下：

```
[admin@deep]$ cd
[admin@deep admin]$ ls -ld . .ssh .ssh/authorized_keys
drwx------ 5 admin admin 1024 Nov 28 07:05 .
drwxr-xr-x 2 admin admin 1024 Nov 29 00:02 .ssh
-rw-r--r-- 1 admin admin 342 Nov 29 00:02 .ssh/authorized_keys
```

改变 pass-phrase　保护密匙的口令

用加上“-p”参数的“ssh-keygen”命令，在任何时候都可以改变 pass-phrase。用下面的命令，改变 pass-phrase：

```
[root@deep]# su username
[username@deep]$ ssh-keygen1 –p


Enter file key is in (/home/username/.ssh/identity): [按下回车键]
Enter old passphrase:
Key has comment 'username@deep.openarch.com'
Enter new passphrase:
Enter the same passphrase again:
Your identification has been saved with the new passphrase.
```

SSH1 用户工具

下面列出的是一些我们经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

ssh1

ssh1（Secure Shell）是用来登录远程计算机和在远程计算机上执行命令的程序。它是用来替代 rlogin 和 rsh，以及在不安全的网络环境下在两台计算机之间提供安全和加密的信息交流。X11 连接和 TCP/IP 端口可以被转发到一个安全的通道里。

用下面的命令，登录远程计算机：

```
[root@deep]# ssh1 <login_name> <hostname>
```

例如：

```
[root@deep]# ssh1 username www.openarch.com
username@deep.openarch.com’s password:
Last login: Tue Oct 19 1999 18:13:00 -0400 from gate.openarch.com
Welcome to www.openarch.com on Deepforest.
```

<login_name>是用来登录 ssh 服务器的用户名，<hostname>是 ssh 服务器主机的地址。

scp1

可以用这个命令把文件从本地计算机拷贝到远程计算机，或者反之，甚至可以在两台远程计算机之间用“scp1”命令拷贝文件。把远程主机上的文件拷贝到当前目录的一个简单的方法如下。

用下面的命令把文件从远程主机拷贝到本地主机上：

```
[root@deep]# su username
[username@deep]$ scp1 -p <login_name@hostname>:/dir/for/file
localdir/to/filelocation
```

例如：

```
[username@deep]$ scp1 -p username@mail:/etc/test1 /tmp
Enter passphrase for RSA key 'username@mail.openarch.com':
test1 | 2 KB | 2.0 kB/s | ETA: 00:00:00 | 100%
```

用下面的命令把文件从本地主机拷贝到远程主机上：

```
[root@deep]# su username
[username@deep]$ scp1 -p localdir/to/filelocation
<username@hostname>:/dir/for/file
```

例如：

```
[username@deep]$ scp1 -p /usr/bin/test2 username@mail:/var/tmp
username@mail's password:
test2 | 7 KB | 7.9 kB/s | ETA: 00:00:00 | 100%
```

注意：“-p”选项表示文件的改变和访问时间属性以及权限，在拷贝过程中被保留。通常是需要这样的。

安装到系统中的文件

```
> /etc/ssh
> /etc/ssh/ssh_host_key
> /etc/ssh/ssh_host_key.pub
> /etc/ssh/ssh_config
> /etc/ssh/sshd_config
> /root/.ssh
> /root/.ssh/random_seed
```

```
> /usr/bin/ssh1
> /usr/bin/ssh
> /usr/bin/slogin
> /usr/bin/ssh-keygen1
> /usr/bin/ssh-keygen
> /usr/bin/ssh-agent1
> /usr/bin/ssh-agent
> /usr/bin/ssh-add1
> /usr/bin/ssh-add
> /usr/bin/scp1
> /usr/bin/scp
> /usr/bin/make-ssh-known-hosts1
> /usr/bin/make-ssh-known-hosts
> /usr/man/man1/scp1.1
> /usr/man/man1/ssh-keygen1.1
> /usr/man/man1/ssh-keygen.1
> /usr/man/man1/ssh-agent1.1
> /usr/man/man1/ssh-agent.1
> /usr/man/man1/ssh-add1.1
> /usr/man/man1/ssh-add.1
> /usr/man/man1/scp.1
> /usr/man/man1/slogin1.1
> /usr/man/man1/slogin.1
> /usr/man/man1/ssh1.1
> /usr/man/man1/ssh.1
> /usr/man/man1/make-ssh-known-hosts1.1
> /usr/man/man1/make-ssh-known-hosts.1
> /usr/man/man8/sshd1.8
> /usr/man/man8/sshd.8
> /usr/sbin/sshd1
> /usr/sbin/sshd
```

Windows 平台上的免费 SSH 客户程序

putty

PuTTY 是 Win32 平台上（在 Win95 和 WinNT 上经测试，据说在 Win98 和 Win2000 上也运行得很好）的免费的 telnet 和 SSH 程序。

软件包

Putty 主页：http://www.chiark.greenend.org.uk/~sgtatham/putty.html。

Tera Term Pro 和 TTSSH

TTSSH 是 Windows 平台上的免费 SSH 客户程序。它是作为 Teraterm Pro 的 DLL 插件。Teraterm Pro 是 Windows 平台上非常优秀的免费终端模拟器和 telnet 客户程序。TTSSH 使得 Teraterm Pro 具有 SSH 的功能而不影响 Teraterm Pro 的功能。TTSSH 也是可以免费下载的，它的源代码也是可以得到的。

软件包

Teraterm Pro 的主页：http://hp.vector.co.jp/authors/VA002416/teraterm.html

TTSSH 的主页：http://www.zip.com.au/~roca/download.html

Linux SSH2 Cient/Server

概述

SSH2 是商用版本。因为有一些人还在使用它，所以我们也介绍一下 SSH2 的配置。为了保证安全性，我们把 sshd2 配置成支持 tcp-wrappers（inetd 超级服务器）。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

ssh2 的版本是 2.0.13。

软件包的来源

SSH2 的主页是：http://www.ssh.fi/

下载：ssh-2.0.13.tar.gz

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >ssh1”，在编译和安装完软件之后运行命令“find /* > ssh2”，最后用命令“diff ssh1 ssh2 > ssh”找出变化。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp ssh-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf ssh-version.tar.gz
```

编译和优化

转到 ssh1 的新目录下，先设置编译器的编译参数：

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-
pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--without-ssh-agent1-compat \
--disable-suid-ssh-signer \
--disable-tcp-port-forwarding \
--disable-X11-forwarding \
--enable-tcp-nodelay \
--with-libwrap
```

这些编译参数告诉编译器如何编译 SSH2：

- 保持和 ssh-agent1 的兼容
- 安装的 ssh-signer 不设置 SUID 位
- 禁止端口转发（port forwarding）
- 禁止 X11 转发（X11 forwarding）
- 使 TCP_NODELAY 的 socket 选项生效
- 编译的时候加上 libwrap（tcp_wrappers）的支持

编译用下面的命令：

```
[root@deep]# make clean
[root@deep]# make
[root@deep]# make install
```

“make clean”，删除所有以前编译的时候留下来的中间文件，以避免任何错误，接着用“make”命令把源程序编译成可执行的二进制程序，最后“make install”安装软件。

清除不必要的文件

用下面的命令删除不必要的文件：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf ssh2-version/ ssh-version.tar.gz
```

“rm”命令删除所有编译和安装 SSH2 所需要的源程序，并且把 SSH2 软件的压缩包删除掉。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 SSH2 Client/Server，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“sshd2_config”文件拷贝到“/etc/ssh2”目录下
- 把“ssh2_config”文件拷贝到“/etc/ssh2”目录下
- 把“ssh”文件拷贝到“/etc/pam.d”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到相应的目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

配置 /etc/ssh2/ssh2_config 文件

“/etc/ssh2/ssh2_config”是 ssh2 的配置文件，允许设置一些选项来改变客户端程序的运行方式。这个文件的每一行包含“关键词－值”的匹配，其中“关键词”是忽略大小写的。下面列出来的是最重要的关键词，用 man 命令查看帮助页（ssh (2)）可以得到详细的列表。

编辑“ssh2_config”文件（vi /etc/ssh2/ssh2_config），加入：

```
# ssh2_config
# SSH 2.0 Client Configuration File
*:
Port 22
```

```
Ciphers AnyStdCipher
Compression yes
IdentityFile identification
AuthorizationFile authorization
RandomSeedFile random_seed
VerboseMode no
ForwardAgent no
ForwardX11 no
PasswordPrompt "%U's password: "
Ssh1Compatibility no
Ssh1AgentCompatibility none
NoDelay yes
KeepAlive yes
QuietMode no
```

下面逐行说明上面的选项设置：

Port 22

“Port”设置 sshd2 监听的端口号。

Ciphers AnyStdCipher

“Cipher”设置加密所用的密码。“AnyStdCipher”表示只允许使用标准的密码。

Compression yes

“Compression”设置是否使用压缩。

IdentityFile identification

“IdentityFile”设置用户标识文件的名字。

AuthorizationFile authorization

“Authorization”设置用户验证文件的名字。

RandomSeedFile random_seed

“RandomSeedFile”设置用户随机数种子（randamseed）文件的名字。

VerboseMode no

“VerboseMode”选项可以让 ssh2 显示调试信息。这在调试连接、验证和配置问题的时候十分有用。

ForwardAgent no

“ForwardAgent”设置是否把与安全验证代理的连接转发到远程计算机。

ForwardX11 no

“ForwardX11”设置 X11 的连接是否被自动转发到安全的通道（channel）和显示集（DISPLAY set）。

PasswordPrompt "%U's password: "

“PasswordPrompt”设置口令提示，也就是当用户连接主机时所看到的。“%U”和“%H”分别表示用户的登录名和主机名。

Ssh1Compatibility no

“Ssh1Compatibility”设置是否和 SSH1 兼容。

Ssh1AgentCompatibility none

“Ssh1AgentCompatibility”是否转发 SSH1 代理连接。

NoDelay yes

“NoDelay”如果设置成“yes”将使 TCP_NODELAY 这个 socket 选项生效。能够提高网络的性能。

KeepAlive yes

“KeepAlive”设置系统是否发送“keep alive”的消息，如果发送“keep alive”的消息，如果连接失败或者有一台计算机死机，就很容易被注意到。

QuietMode no

“QuietMode”忽略所有的警告和诊断消息，只有严重的错误才会被显示出来。

配置 /etc/ssh2/sshd2_config 文件

“/etc/ssh2/sshd2_config”是 sshd2 的配置文件，允许设置选项改变这个 daemon 的运行。这个文件的每一行包含“关键词－值”的匹配，其中“关键词”是忽略大小写的。下面列出来的是最重要的关键词，用 man 命令查看帮助页（sshd2(8)）可以得到详细的列表。

编辑“ssh2_config”（vi /etc/ssh2/ssh2_config）文件并加入：

```
# sshd2_config
# SSH 2.0 Server Configuration File
*:
Port 22
ListenAddress 192.168.1.1
```

```
Ciphers AnyStdCipher
IdentityFile identification
AuthorizationFile authorization
HostKeyFile hostkey
PublicHostKeyFile hostkey.pub
RandomSeedFile random_seed
ForwardAgent no
ForwardX11 no
PasswordGuesses 3
MaxConnections 5
PermitRootLogin no
AllowedAuthentications publickey,password
RequiredAuthentications publickey,password
VerboseMode no
PrintMotd yes
CheckMail yes
UserConfigDirectory "%D/.ssh2"
SyslogFacility AUTH
Ssh1Compatibility no
NoDelay yes
KeepAlive yes
UserKnownHosts yes
AllowHosts 192.168.1.4
DenyHosts *
QuietMode no
# subsystem definitions
subsystem-sftp sftp-server
```

下面逐行说明上面的选项设置：

Port 22

“Port”设置 sshd2 监听的端口号。

ListenAddress 192.168.1.1

“ListenAddress”设置 sshd2 服务器绑定的 IP 地址。

Ciphers AnyStdCipher

“Cipher”设置加密所用的密码。“AnyStdCipher”表示只允许使用标准的密码。

IdentityFile identification

“IdentityFile”设置用户标识文件的名字。

AuthorizationFile authorization

“Authorization”设置用户验证文件的名字。

HostKeyFile hostkey

“HostKeyFile”设置包含主机私有密匙的文件（默认是“/etc/ssh2/hostkey”）。

PublicHostKeyFile hostkey.pub

“PublicHostKeyFile”设置包含主机公用密匙的文件（默认是“/etc/ssh2/hostkey.pub”）。

RandomSeedFile random_seed

“RandomSeedFile”设置用户随机数种子（randamseed）文件的名字。

ForwardAgent no

“ForwardAgent”设置连接是否经过验证代理（如果存在）转发给远程计算机。

ForwardX11 no

“ForwardX11”设置 X11 连接是否被自动重定向到安全的通道和显示集（DISPLAY set）。

PasswordGuesses 3

“PasswordGuesses”设置口令验证的时候允许用户输入几次口令。

MaxConnections 5

“MaxConnections”设置 sshd2 允许最多同时处理多少个连接。因为太多的连接有可能导致系统的不稳定或死机。

PermitRootLogin no

“PermitRootLogin”设置是否允许 root 用 ssh 登录。千万不要把这项设成“yes”。

AllowedAuthentications publickey,password

“AllowedAuthentications”设置安全验证的方法。可以设定的值是“password”、“publickey”和“hostbased”，中间用逗号隔开。默认为“password, publickey”。与“RequiredAuthentications”搭配使用，系统管理员可以强制用户经过多次的安全验证。

RequiredAuthentications publickey,password

“RequiredAuthentications”是与“AllowedAuthentications”相关的，设置用户需要经过怎样的安全验证。这个选项没有默认值，而且后面的参数必须是“AllowedAuthentications”参数的子集。否则，服务器不允许建立连接。

VerboseMode no

“VerboseMode”选项可以让 ssh2 显示调试信息。这在调试连接、验证和配置问题的时候十分有用。

PrintMotd yes

“PrintMotd”设置 sshd2 是否在用户登录的时候显示“/etc/motd”中的信息。

CheckMail yes

“CheckMail”设置当用户登录的时候，ssd 是否提示有新邮件。

UserConfigDirectory "%D/.ssh2"

“UserConfigDirectory”设置从什么地方读取每个用户的配置信息。用这个选项可以设置用户配置信息。这个选项用“匹配串”的方式设定。用户登录的时候 sshd2 会把这个匹配串展开。“%D”为用户的家目录，“%U”为用户的登录名，“%IU”为用户的 UID，“%IG”为用户的组 ID（GID）。

SyslogFacility AUTH

“SyslogFacility”设置在记录来自 sshd2 的消息的时候，是否给出“facility code”。

Ssh1Compatibility no

“Ssh1Compactibility”设置是否和 SSH1 兼容。

NoDelay yes

“NoDelay”如果设置成“yes”将使 TCP_NODELAY 这个 socket 选项生效。能够提高网络的性能。

KeepAlive yes

“KeepAlive”设置系统是否发送“keep alive”的消息，如果发送“keep alive”的消息，如果连接失败或者有一台计算机死机，就很容易被注意到。

UserKnownHosts yes

“UserKnownHosts”设置当使用“hostbased”验证方式的时候，是否能够从用户的“$HOME/.ssh2/knownhosts”目录获取主机的公用密匙。

AllowHosts 192.168.1.4

“AllowHosts”后面可以跟着任意数量的主机名的匹配串（用空格隔开）。设定之后，用户只允许从与匹配串相匹配的主机上登录。匹配串可以包含“*”和“?”这样的通配符。通常用域名服务器完成客户机到正式主机名的对应，如果找不到对应，就只好用 IP 地址了。

DenyHosts *

“DenyHosts”后面也可以跟着任意数量的主机名的匹配串（用空格隔开）。设定之后，用户只禁止从与匹配串相匹配的主机上登录。

QuietMode no

“QuietMode”忽略所有的警告和诊断消息，只有严重的错误才会被显示出来。

配置 sshd2 使其使用 TCP-WRAPPERS　inetd 超级服务器

TCP-WRAPPERS 用来启动和停止 sshd2 服务。当 inetd 运行的时候，它会从配置文件（默认为“/etc/inetd.conf”）中读入配置信息。在配置文件中每一行的不同项是用 TAB 或空格分开。

第一步

编辑“inetd.conf”文件（vi /etc/inetd.conf）并加入这一行：

```
ssh stream tcp nowait root /usr/sbin/tcpd sshd -i
```

注意：“-i”参数很重要，它说明 sshd 是被 inetd 运行的。在加入这一行后，通过发送一个 SIGHUP 信号（killall –HUP inetd）来更新“inetd.conf”文件。

```
[root@deep /root]# killall -HUP inetd
```

第二步

编辑“hosts.allow”文件（vi /etc/hosts.allow）并加入这一行：

```
sshd: 192.168.1.4 win.openarch.com
```

这一行表示 IP 地址为“192.168.1.4”，主机名为“win.openarch.com”的计算机允许用 ssh 访问服务器。

下面这些“daemon”字符串（用于 TCP-WRAPPERS）被 sshd2 使用：

```
sshd, sshd2 (sshd2 被调用时用的名字(通常是“sshd”)).
sshdfwd-X11 (允许/禁止 X11 转发).
```

```
sshdfwd-<port-number> (TCP 转发).
sshdfwd-<port-name> (port-name 在/etc/services 中定义。用于 TCP 转发).
```

注意：如果准备使用 ssh，一定要用在所有的服务器上。如果十台安全的服务器和一台不安全的服务器配在一起，也谈不上什么安全性。

配置 /etc/pam.d/ssh 文件

为了使用 PAM 验证必须配置“/etc/pam.d/ssh”文件。

创建“ssh”文件（touch /etc/pam.d/ssh）并加入：

```
#%PAM-1.0
auth required /lib/security/pam_pwdb.so shadow
auth required /lib/security/pam_nologin.so
account required /lib/security/pam_pwdb.so
password required /lib/security/pam_cracklib.so
password required /lib/security/pam_pwdb.so use_authtok nullok md5 shadow
session required /lib/security/pam_pwdb.so
```

更多的资料

如果想查找详细的资料可以用 man 命令查帮助页，读取相关信息：

```
$ man ssh-add2 (1) - adds identities for the authentication agent
$ man ssh-agent2 (1) - authentication agent
$ man ssh-keygen2 (1) - authentication key pair generation
$ man ssh2 (1) - secure shell client (remote login program)
$ man sshd2 (8) - secure shell daemon
```

SSH2 每用户配置

第一步

为本地服务器创建私有和公用密匙，执行下面的命令：

```
[root@deep]# su username
[username@deep]$ ssh-keygen2
```

第二步

在用户本地计算机的家目录的“.ssh2”目录下创建“标识”文件：

```
[username@deep]$ cd ~/.ssh2
[username@deep]$ echo “IdKey id_dsa_1024_a” > identification
```

注意：在远程计算机的上创建表示文件是可选的。

第三步

把本地的公用密匙（id_dsa_1024_a.pub）用“Local.pub”的名字拷贝到远程计算机的“.ssh2”目录下。

第四步

在远程计算机的“.ssh2”目录下创建“authorization”文件：

```
[username@deep]$ touch authorization
```

第五步

把下面这一行加入“authorization”文件（vi authorization）中：

```
key Local.pub
```

SSH2 用户工具

下面列出的是一些我们经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

ssh2

ssh2（Secure Shell）是用来登录远程计算机和在远程计算机上执行命令的程序。它是用来替代 rlogin 和 rsh，以及为在不安全的网络环境下在两台计算机之间提供安全和加密的信息交流。X11 连接和 TCP/IP 端口可以被转发到一个安全的通道里。

用下面的命令，登录远程计算机：

```
[root@deep]# ssh2 -l <login_name> <hostname>
```

例如：

```
[root@deep]# ssh2 -l username www.openarch.com
```

```
Passphrase for key "/home/username/.ssh2/id_dsa_1024_a" with comment "1024-bit
dsa,
```

```
username@deep.openarch.com, Tue Oct 19 1999 14:31:40 -0400":
username's password:
Last login: Tue Oct 19 1999 18:13:00 -0400 from gate.openarch.com
Welcome to www.openarch.com on Deepforest.
```

<login_name>是用来登录 ssh 服务器的用户名，<hostname>是 ssh 服务器主机的地址。

sftp2

sftp（Secure File Transfer）是用于在网络中传送文件的类似 ftp 的客户端软件。sftp 用 ssh2 进行数据传送，所以文件传送是安全的。必须先用 ssh2 连接到主机上才能用 sftp2。

用下面的命令使用 sftp2：

```
[username@deep]$ sftp2 <hostname>
```

例如：

```
[username@deep]$ sftp2 www.openarch.com
```

```
local path : /home/username
Passphrase for key "/home/username/.ssh2/id_dsa_1024_a" with comment "1024-bit
dsa,
username@deep.openarch.com, Tue Oct 19 1999 14:31:40 -0400":
username's password:
username's password:
remote path : /home/username
sftp>
```

这里<hostname>是用 sftp2 命令传送文件的远程主机名。

安装到系统中的文件

```
> /etc/pam.d/ssh
> /etc/ssh2
> /etc/ssh2/hostkey
> /etc/ssh2/hostkey.pub
> /etc/ssh2/sshd2_config
> /etc/ssh2/ssh2_config
```

```
> /root/.ssh2
> /root/.ssh2/random_seed
> /root/ssh2
> /usr/man/man1/ssh2.1
> /usr/man/man1/ssh-keygen2.1
> /usr/man/man1/ssh-add2.1
> /usr/man/man1/ssh-agent2.1
> /usr/man/man1/scp2.1
> /usr/man/man1/sftp2.1
> /usr/man/man8/sshd2.8
> /usr/man/man8/sshd.8
> /usr/bin/ssh2
> /usr/bin/scp2
> /usr/bin/sftp2
> /usr/bin/sftp-server2
> /usr/bin/ssh-agent2
> /usr/bin/ssh-keygen2
> /usr/bin/ssh-signer2
> /usr/bin/ssh-add2
> /usr/bin/ssh
> /usr/bin/ssh-agent
> /usr/bin/ssh-add
> /usr/bin/ssh-askpass
> /usr/bin/ssh-keygen
> /usr/man/man1/ssh.1
> /usr/man/man1/ssh-add.1
> /usr/man/man1/ssh-agent.1
> /usr/man/man1/ssh-keygen.1
> /usr/man/man1/scp.1
> /usr/man/man1/sftp.1
> /usr/bin/scp
> /usr/bin/sftp
> /usr/bin/sftp-server
> /usr/bin/ssh-signer
> /usr/sbin/sshd2
> /usr/sbin/sshd
```

Linux Tripwire 2.2.1

概述

Tripwire 最早的 1.0 版本是在 1992 年设计的。Tripwire 2.0 版是完全重新设计的，而且不公布源代码。策略（policy）的设定不再放在配置文件中，而且策略语言也发生了变化。基于 64 位的编码和原来也不一样。Tripwire2.0 以及更高版本使用四种加密签名。ASR 提供了 8 个签名、“no signature”可选项以及增加定制签名的能力。但是，以现在的标准来看多个 ASR 签名还是有其弱点的。

根据 Tripwire 官方站点的介绍，Tripwire 运行在操作系统的底层，可以保护组成企业网的每个服务器和工作站。Tripwire 首先要扫描计算机并创建有关系统文件的数据库，也就是系统在已知的安全状态的一个“快照”（snapshot）。用户可以很精确地配置 Tripwire，既可以让它监控每一台计算机上不同的文件和目录，又可以创建一个标准的模板，并把它用于企业中所有的计算机。

一旦基准数据库（baseline database）建立了，系统管理员可以在任何时候用 Tripwire 检测系统的一致性。通过扫描当前的系统并且和数据库中的信息比较，Tripwire 可以发现并报告在限定范围之外，系统中任何的添加、删除和改变。如果这些变化是正常的，管理员可以更新基准数据库。如果发现这些变化是异常的，系统管理员马上就可以知道网络中的哪些部分受到影响。

这个版本的 Tripwire 比低版本的 Tripwire 在某些方面有了很大的增强。这些增强包括：

- 不同级别的报表允许根据需要选择报表中出现的信息。
- syslog 这个选项可以设置把有关数据库初始化、数据库更新、策略更新和一致性检验的信息发送给 syslog。
- 优化了数据库的性能，可以很大地提升一致性检测的效率。
- 一个报表的不同部分可以发送给不同的接收者。
- 支持用 SMTP 发送 email 报表。
- email 测试模式可以测试 email 的设置是否正确。
- 能够在一个策略文件中创建独立运行的不同部分。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

Tripwire 的版本是 2.2.1。

软件包的来源

Tripwire 的主页：http://www.tripwiresecurity.com/。

下载：Tripwire_221_for_Linux_x86_tar.gz。

安装 Tripwire

把软件包（tar.gz）解压缩：

```
[root@deep]# cp Tripwire_version_for_Linux_x86_tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf Tripwire_version_for_Linux_x86_tar.gz
```

注意：解压完 Tripwire 之后，可在“/var/tmp”目录下看到下面这些有关 Tripwire 的文件：License.txt、README、Release_Notes、 install.cfg、install.sh、pkg 目录和 Tripwire 软件包 Tripwire_version_for_Linux_x86_tar.gz。

安装过程

安装过程有这么几步：

1. 设定安装选项
2. 计划好为站点和本地密匙设定两个密码（passphrase）
3. 运行安装脚本
4. 编辑配置文件和策略文件
5. 根据需要解决配置和策略文件中的问题
6. 初始化 Tripwire 数据库

配置 /var/tmp/install.cfg 文件

前面已经说过 Tripwire 2.2.1 版并不是开放源代码的软件，因此不能像其它软件那样编译和安装，必须修改 Tripwire 的“install.cfg”文件（自动安装 Tripwire 软件的脚本文件），设定软件的安装路径。而且，还要根据 RedHat 文件系统的结构，修改这个文件，把 Tripwire 的二进制文件安装在“PATH”环境变量指定的路径下。

第一步

编辑“install.cfg”文件（vi /var/tmp/install.cfg），把文件改成：

```
#
# install.cfg
#
# default install.cfg for:
# Tripwire(R) 2.2.1 for Unix
#
# NOTE: This is a Bourne shell script that stores installation
# parameters for your installation. The installer will
# execute this file to generate your config file and also to
# locate any special configuration needs for your install.
# Protect this file, because it is possible for
# malicious code to be inserted here
#
# To set your Root directory for install, set TWROOT= to something
# other than /usr/TSS as necessary.
#
#=========================================================
# If CLOBBER is true, then existing files are overwritten.
# If CLOBBER is false, existing files are not overwritten.
CLOBBER=false
# The root of the TSS directory tree.
TWROOT="/usr"
# Tripwire binaries are stored in TWBIN.
TWBIN="${TWROOT}/bin"
# Tripwire policy files are stored in TWPOLICY.
TWPOLICY="${TWROOT}/TSS/policy"
# Tripwire manual pages are stored in TWMAN.
TWMAN="${TWROOT}/man"
# Tripwire database files are stored in TWDB.
TWDB="${TWROOT}/TSS/db"
# The Tripwire site key files are stored in TWSITEKEYDIR.
TWSITEKEYDIR="${TWROOT}/TSS/key"
# The Tripwire local key files are stored in TWLOCALKEYDIR.
TWLOCALKEYDIR="${TWROOT}/TSS/key"
```

```
# Tripwire report files are stored in TWREPORT.
TWREPORT="${TWROOT}/TSS/report"
# This sets the default text editor for Tripwire.
TWEDITOR="/bin/vi"
# TWLATEPROMTING controls the point when tripwire asks for a password.
TWLATEPROMPTING=false
# TWLOOSEDIRCHK selects whether the directory should be monitored for
# properties that change when files in the directory are monitored.
TWLOOSEDIRCHK=false
# TWMAILNOVIOLATIONS determines whether Tripwire sends a no violation
# report when integrity check is run with --email-report but no rule
# violations are found. This lets the admin know that the integrity
# was run, as opposed to having failed for some reason.
TWMAILNOVIOLATIONS=true
# TWEMAILREPORTLEVEL determines the verbosity of e-mail reports.
TWEMAILREPORTLEVEL=3
# TWREPORTLEVEL determines the verbosity of report printouts.
TWREPORTLEVEL=3
# TWSYSLOG determines whether Tripwire will log events to the system log
TWSYSLOG=false
#####################################
# Mail Options - Choose the appropriate
# method and comment the other section
#####################################
#####################################
# SENDMAIL options - DEFAULT
#
# Either SENDMAIL or SMTP can be used to send reports via TWMAILMETHOD.
# Specifies which sendmail program to use.
#####################################
TWMAILMETHOD=SENDMAIL
TWMAILPROGRAM="/usr/lib/sendmail -oi -t"
#####################################
# SMTP options
#
# TWSMTPHOST selects the SMTP host to be used to send reports.
# SMTPPORT selects the SMTP port for the SMTP mail program to use.
#####################################
# TWMAILMETHOD=SMTP
# TWSMTPHOST="mail.domain.com"
# TWSMTPPORT=25
###############################################################################
####
# Copyright (C) 1998-2000 Tripwire (R) Security Systems, Inc. Tripwire (R) is
```

```
a
# registered trademark of the Purdue Research Foundation and is licensed
# exclusively to Tripwire (R) Security SystemsInc.
```

第二步

根据默认和自定义的设置，现在可以运行安全脚本安装 Tripwire 软件了。

用下面的命令安装 Tripwire：

```
[root@deep tmp]# ./install.sh
```

注意：“install.sh”是安装脚本，用来安装 Tripwire。

第三步

安装完 Tripwire 之后，安装程序会自动把“License.txt”、“README”和“Release_Notes”文件拷贝到“/usr”目录下。当然，读完这些文件之后可以用下面的命令把它们删除掉：

```
[root@deep tmp]# rm -f /usr/Lincense.txt README Release_Notes
```

清除不必要的文件

用下面的命令删除不必要的文件：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf License.txt README Release-Notes install.cfg install.sh
pkg/Tripwire_version_for_Linux_x86_tar.gz
```

“rm”命令删除所有安装 Tripwire 所需要的源程序，并且把 Tripwire 软件的压缩包删除掉。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不

管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 Tripwire for Linux，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“twpol.txt”文件拷贝到“/usr/TSS/policy”目录下。

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到相应的目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

配置 /usr/TSS/policy/twpol.txt 文件

“/usr/TSS/policy/twpol.txt”文件是纯文本的策略文件，设置 Tripwire 需要检测哪些文件和目录（也叫系统对象）。其中有一个规则设定如何检测需要监控的对象，还有一个特性（property）设置如何检测。特性掩码（property mask）设定进行一致性检验的时候单独文件的特性（property）。属性（Attributes）帮助规定成组的策略如何运作。需要注意的是编辑策略文件是一个不断反复的过程，在这个过程中要根据实际经验总结出 Tripwire 报表中必须报告的信息。

第一步

必须改变默认的策略文件或建立新的文件。可以看看“/usr/TSS/policy”目录下的“policyguide.txt”文件中的介绍。用文本编辑器打开“twpol.txt”策略文件（vi /usr/TSS/policy/twpol.txt）把它改变成：

```
@@section GLOBAL
TWROOT="/usr";
TWBIN="/usr/bin";
TWPOL="/usr/TSS/policy";
TWDB="/usr/TSS/db";
TWSKEY="/usr/TSS/key";
TWLKEY="/usr/TSS/key";
TWREPORT="/usr/TSS/report";
HOSTNAME=deep.openarch.com;
@@section FS
SEC_CRIT = $(IgnoreNone)-SHa; # Critical files - we can't afford to miss any changes.
SEC_SUID = $(IgnoreNone)-SHa; # Binaries with the SUID or SGID flags set.
SEC_TCB = $(ReadOnly); # Members of the Trusted Computing Base.
SEC_BIN = $(ReadOnly); # Binaries that shouldn't change
SEC_CONFIG = $(Dynamic); # Config files that are changed infrequently but accessed
often.
SEC_LOG = $(Growing); # Files that grow, but that should never change ownership.
SEC_INVARIANT = +pug; # Directories that should never change permission or
ownership.
SIG_LOW = 33; # Non-critical files that are of minimal security impact
```

```
SIG_MED = 66; # Non-critical files that are of significant security impact
SIG_HI = 100; # Critical files that are significant points of vulnerability
# Tripwire Binaries
(emailto = admin@openarch.com, rulename = "Tripwire Binaries", severity =
$(SIG_HI))
{
$(TWBIN)/siggen -> $(ReadOnly);
$(TWBIN)/tripwire -> $(ReadOnly);
$(TWBIN)/twadmin -> $(ReadOnly);
$(TWBIN)/twprint -> $(ReadOnly);
}
# Tripwire Data Files - Configuration Files, Policy Files, Keys, Reports, Databases
(emailto = admin@openarch.com, rulename = "Tripwire Data Files", severity =
$(SIG_HI))
{
# NOTE: Removing the inode attribute because when Tripwire creates a backup
# it does so by renaming the old file and creating a new one (which will
# have a new inode number). Leaving inode turned on for keys, which shouldn't
# ever change.
# NOTE: this rule will trigger on the first integrity check after database
# initialization, and each integrity check afterward until a database update
# is run, since the database file will not exist before that point.
$(TWDB) -> $(Dynamic) -i;
$(TWPOL)/tw.pol -> $(SEC_BIN) -i;
$(TWBIN)/tw.cfg -> $(SEC_BIN) -i;
$(TWLKEY)/$(HOSTNAME)-local.key -> $(SEC_BIN) ;
$(TWSKEY)/site.key -> $(SEC_BIN) ;
#don't scan the individual reports
$(TWREPORT) -> $(Dynamic) (recurse=0);
}
# These files are critical to a correct system boot.
(emailto = admin@openarch.com, rulename = "Critical system boot files", severity
= 100)
{
/boot -> $(SEC_CRIT) ;
!/boot/System.map ;
!/boot/module-info ;
}
# These files change the behavior of the root account
(emailto = admin@openarch.com, rulename = "Root config files", severity = 100)
{
/root -> $(SEC_CRIT) ;
/root/.bash_history -> $(SEC_LOG) ;
}
```

```
# Commonly accessed directories that should remain static with regards to owner
and group
(emailto = admin@openarch.com, rulename = "Invariant Directories", severity =
$(SIG_MED))
{
/ -> $(SEC_INVARIANT) (recurse = 0);
/home -> $(SEC_INVARIANT) (recurse = 0);
/etc -> $(SEC_INVARIANT) (recurse = 0);
/chroot -> $(SEC_INVARIANT) (recurse = 0);
/cache -> $(SEC_INVARIANT) (recurse = 0);
}
(emailto = admin@openarch.com, rulename = "Shell Binaries")
{
/bin/bsh -> $(SEC_BIN);
/bin/csh -> $(SEC_BIN);
/bin/sh -> $(SEC_BIN);
}
# Rest of critical system binaries
(emailto = admin@openarch.com, rulename = "OS executables and libraries", severity
= $(SIG_HI))
{
/bin -> $(ReadOnly) ;
/lib -> $(ReadOnly) ;
}
# Local files
(emailto = admin@openarch.com, rulename = "User binaries", severity = $(SIG_MED))
{
/sbin -> $(SEC_BIN) (recurse = 1);
/usr/sbin -> $(SEC_BIN) (recurse = 1);
/usr/bin -> $(SEC_BIN) (recurse = 1);
}
# Temporary directories
(emailto = admin@openarch.com, rulename = "Temporary directories", recurse = false,
severity =
$(SIG_LOW))
{
/usr/tmp -> $(SEC_INVARIANT);
/var/tmp -> $(SEC_INVARIANT);
/tmp -> $(SEC_INVARIANT);
}
# Libraries
(emailto = admin@openarch.com, rulename = "Libraries", severity = $(SIG_MED))
{
/usr/lib -> $(SEC_BIN);
```

```
}
# Include
(emailto = admin@openarch.com, rulename = "OS Development Files", severity =
$(SIG_MED))
{
/usr/include -> $(SEC_BIN);
}
# Shared
(emailto = admin@openarch.com, rulename = "OS Shared Files", severity = $(SIG_MED))
{
/usr/share -> $(SEC_BIN);
}
# Kernel headers files
(emailto = admin@openarch.com, rulename = "Kernel Headers Files", severity =
$( SIG_HI))
{
/usr/src/linux-2.2.14 -> $(SEC_BIN);
}
# setuid/setgid root programs
(emailto = admin@openarch.com, rulename = "setuid/setgid", severity = $(SIG_HI))
{
/bin/su -> $(SEC_SUID);
/sbin/pwdb_chkpwd -> $(SEC_SUID);
/sbin/dump -> $(SEC_SUID);
/sbin/restore -> $(SEC_SUID);
/usr/bin/at -> $(SEC_SUID);
/usr/bin/passwd -> $(SEC_SUID);
/usr/bin/suidperl -> $(SEC_SUID);
/usr/bin/crontab -> $(SEC_SUID);
/usr/sbin/sendmail -> $(SEC_SUID);
/usr/bin/man -> $(SEC_SUID);
/usr/bin/sperl5.00503 -> $(SEC_SUID);
/usr/bin/slocate -> $(SEC_SUID);
/usr/sbin/utempter -> $(SEC_SUID);
/sbin/netreport -> $(SEC_SUID);
}
(emailto = admin@openarch.com, rulename = "Configuration Files")
{
/etc/hosts -> $(SEC_CONFIG);
/etc/inetd.conf -> $(SEC_CONFIG);
/etc/initlog.conf -> $(SEC_CONFIG);
/etc/inittab -> $(SEC_CONFIG);
/etc/resolv.conf -> $(SEC_CONFIG);
/etc/syslog.conf -> $(SEC_CONFIG);
```

```
}
(emailto = admin@openarch.com, rulename = "Security Control")
{
/etc/group -> $(SEC_CRIT);
/etc/security/ -> $(SEC_CRIT);
/lib/security/ -> $(SEC_CRIT);
/var/spool/cron -> $(SEC_CRIT);
}
(emailto = admin@openarch.com, rulename = "Login Scripts")
{
/etc/csh.login -> $(SEC_CONFIG);
/etc/profile -> $(SEC_CONFIG);
}
# These files change every time the system boots
(emailto = admin@openarch.com, rulename = "System boot changes", severity =
$(SIG_HI))
{
/dev/log -> $(Dynamic) ;
/dev/cua0 -> $(Dynamic) ;
/dev/console -> $(Dynamic) ;
/dev/tty2 -> $(Dynamic) ; # tty devices
/dev/tty3 -> $(Dynamic) ; # are extremely
/dev/tty4 -> $(Dynamic) ; # variable
/dev/tty5 -> $(Dynamic) ;
/dev/tty6 -> $(Dynamic) ;
/dev/urandom -> $(Dynamic) ;
/dev/initctl -> $(Dynamic) ;
/var/lock/subsys -> $(Dynamic) ;
/var/run -> $(Dynamic) ; # daemon PIDs
/var/log -> $(Dynamic) ;
/etc/ioctl.save -> $(Dynamic) ;
/etc/.pwd.lock -> $(Dynamic) ;
/etc/mtab -> $(Dynamic) ;
/lib/modules -> $(Dynamic) ;
}
# Critical configuration files
(emailto = admin@openarch.com, rulename = "Critical configuration files",
severity = $(SIG_HI))
{
/etc/conf.modules -> $(ReadOnly) ;
/etc/crontab -> $(ReadOnly) ;
/etc/cron.hourly -> $(ReadOnly) ;
/etc/cron.daily -> $(ReadOnly) ;
/etc/cron.weekly -> $(ReadOnly) ;
```

```
/etc/cron.monthly -> $(ReadOnly) ;
/etc/default -> $(ReadOnly) ;
/etc/fstab -> $(ReadOnly) ;
/etc/group- -> $(ReadOnly) ; # changes should be infrequent
/etc/host.conf -> $(ReadOnly) ;
/etc/hosts.allow -> $(ReadOnly) ;
/etc/hosts.deny -> $(ReadOnly) ;
/etc/lilo.conf -> $(ReadOnly) ;
/etc/logrotate.conf -> $(ReadOnly) ;
/etc/pwdb.conf -> $(ReadOnly) ;
/etc/securetty -> $(ReadOnly) ;
/etc/sendmail.cf -> $(ReadOnly) ;
/etc/protocols -> $(ReadOnly) ;
/etc/services -> $(ReadOnly) ;
/etc/rc.d/init.d -> $(ReadOnly) ;
/etc/rc.d -> $(ReadOnly) ;
/etc/motd -> $(ReadOnly) ;
/etc/passwd -> $(ReadOnly) ;
/etc/passwd- -> $(ReadOnly) ;
/etc/profile.d -> $(ReadOnly) ;
/etc/rpc -> $(ReadOnly) ;
/etc/sysconfig -> $(ReadOnly) ;
/etc/shells -> $(ReadOnly) ;
/etc/nsswitch.conf -> $(ReadOnly) ;
}
# Critical devices
(emailto = admin@openarch.com, rulename = "Critical devices", severity = $(SIG_HI),
recurse = false)
{
/dev/kmem -> $(Device) ;
/dev/mem -> $(Device) ;
/dev/null -> $(Device) ;
/dev/zero -> $(Device) ;
/proc/devices -> $(Device) ;
/proc/net -> $(Device) ;
/proc/tty -> $(Device) ;
/proc/sys -> $(Device) ;
/proc/cpuinfo -> $(Device) ;
/proc/modules -> $(Device) ;
/proc/mounts -> $(Device) ;
/proc/dma -> $(Device) ;
/proc/filesystems -> $(Device) ;
/proc/ide -> $(Device) ;
/proc/interrupts -> $(Device) ;
```

```
/proc/ioports -> $(Device) ;
/proc/scsi -> $(Device) ;
/proc/kcore -> $(Device) ;
/proc/self -> $(Device) ;
/proc/kmsg -> $(Device) ;
/proc/stat -> $(Device) ;
/proc/ksyms -> $(Device) ;
/proc/loadavg -> $(Device) ;
/proc/uptime -> $(Device) ;
/proc/locks -> $(Device) ;
/proc/version -> $(Device) ;
/proc/meminfo -> $(Device) ;
/proc/cmdline -> $(Device) ;
/proc/misc -> $(Device) ;
}
```

注意：这是标准的策略文件，必须根据实际情况加以改变。

第二步

当第一次准备使用策略文件的时候，用下面的安装它：

```
[root@deep]# twadmin --create-polfile /usr/TSS/policy/twpol.txt


Please enter your site passphrase:
Wrote policy file: /usr/TSS/policy/tw.pol
```

保证 Tripwire for Linux 的安全

一定要保证系统的一致性不会遭到破坏。为了最大限度地保证安全性，必须用 Linux 发行商提供的安装光盘安装操作系统，这样就可以得到一个干净的基准数据库。最好把在“/usr/bin”目录下的名为“twcfg.txt”的纯文本的 Tripwire 配置文件删掉，这样就可以隐藏 Tripwire 文件安装的地方，也可以防止有人创建另外一个配置文件。

用下面的命令把纯文本的 Tripwire 配置文件删掉：

```
[root@deep]# rm -f /usr/bin/twcfg.txt
```

更多的资料

如果想查找详细的资料可以用 man 命令查帮助页，读取相关信息：

```
siggen (8) - signature gathering routine for Tripwire
tripwire (8) - a file integrity checker for UNIX systems
twadmin (8) - Tripwire administrative and utility tool
twconfig (4) - Tripwire configuration file reference
twfiles (5) - overview of files used by Tripwire and file backup process
twintro (8) - introduction to Tripwire software
twpolicy (4) - Tripwire policy file reference
twprint (8) - Tripwire database and report printer
```

命令

下面列出的是一些我们经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

第一次创建基准数据库

在数据库初始化的时候，Tripwire 根据策略文件中的规则建立关于文件系统中对象的数据库。

这个数据库作为今后一致性检查的基准。初始化数据库的语法是：

```
[root@deep]# tripwire { --init }
```

用下面的命令初始化数据库：

```
[root@deep]# tripwire --init
Please enter your local passphrase:
Parsing policy file: /usr/TSS/policy/tw.pol
Generating the database...
*** Processing Unix File System ***
Wrote database file: /usr/TSS/db/deep.openarch.com.twd
The database was successfully generated.
```

注意：当这命令执行完之后，数据库就准备好了，就可以检查系统的一致性并查看报表了。

进行一致性检查

一致性检查把当前文件系统中的对象及其属性和 Tripwire 数据库中的进行比较。一旦发现异常情况，就会在标准输出上显示出来，报表文件也会被保存下来，以后可以用“twprint”命令查看：

一致性检查的语法是：

```
[root@deep]# tripwire { --check }
```

用下面的命令进行一致性检查：

```
[root@deep]# tripwire –check
```

也可以在交互模式下使用 Tripwire。在交互模式下，可以通过终端自动更新系统的变化。加上“--interative”参数，报表会在一个文本编辑器中打开。

用下面的命令进行交互式的检查：

```
[root@deep]# tripwire --check --interactive
```

加上“--email-report”参数可以让 Tripwire 给你用 email 发送报表。接受者是在策略文件中定义的，还有一些选项在默认的配置文件中设置。

用下面的命令进行一致性检查并用 email 发送报表：

```
[root@deep]# tripwire --check --email-report
```

一致性检查之后更新数据库

如果系统的变化是有效的，数据库更新模式允许你在完成一致性检查之后更新 Tripwire 的数据库。数据库的更新可以节省很多的时候，不用再重新生成数据库了。更重要的是这种更新是有选择性的，而重新生成数据库是无法办到这一点的。因为在 Tripwire 的配置文件中我们使用基于时间的变量作为报表的文件名，这样报表在正常的更新模式下就找不到了。这是因为$(DATE)变量为了反应当前的系统时间会发生变化。为了解决这个问题，必须给下面的命令加上“-r”或“--twrfile”参数。

更新数据库的语法是：

```
[root@deep]# tripwire { --update -r}
```

用下面的命令更新数据库：

```
[root@deep]# tripwire --update -r
/usr/TSS/report/deep.openarch.com-200001-021854.twr
```

“-r”参数读取指定的报表文件（deep.openarch.com-200001-021854.twr）。因为当前的配置文件的 REPORTFILE 变量使用$(DATE)，所以“-r”这个参数是必须的。

注意：在数据库更新模式或交互式检查模式下，Tripwire 的报表给每一个违反策略文件中所定义的规则的地方加上一个选择框。可保留选择框中的“x”，表示接受这个变化。如果把选择框中的“x”移掉，表示数据库不会更新这个变化。等退出编辑器并输入本地的 passphrase（密码）之后，Tripwire 就会更新并保存数据库。

更新策略文件

可以改变策略文件中定义的规则，这将改变 Tripwire 软件扫描系统的方式，而且不需要重新生成数据库就可以更新它。这样可以省掉很多时间，更重要的是因为策略文件和数据库同步可以保证系统的安全性。

策略更新模式的语法是：

```
[root@deep]# tripwire { --update-policy /path/to/new/policy/file}
```

用下面的命令更新策略文件：

```
[root@deep]# tripwire --update-policy /usr/TSS/policy/newtwpol.txt
```

在默认情况下，策略更新模式使用“--secure-mode high”。如果文件系统在最近的一次数据库更新之后发生了变化，而且这个变化会违反策略文件中定义的规则，那么在高安全级别的模式下运行你可能会遇到一些问题。例如这种情况：其他的管理员在策略更新的过程中，改变了一些文件。为了解决这个问题，确信在高安全级别的模式下所有的变化都是正常的之后，可以采用低级别的安全模式更新策略文件：

用下面的命令在低级别的安全模式下更新策略文件：

```
[root@deep]# tripwire --update-policy --secure-mode low
/usr/TSS/policy/newtwpol.txt
```

安装到系统中的文件

```
> /usr/TSS
> /usr/bin
> /usr/bin/siggen
> /usr/bin/twprint
> /usr/bin/twadmin
> /usr/bin/tripwire
> /usr/bin/twcfg.txt
> /usr/bin/tw.cfg
> /usr/TSS/policy
> /usr/TSS/policy/policyguide.txt
> /usr/TSS/policy/twpol.txt
```

```
> /usr/TSS/policy/tw.pol
> /usr/TSS/policy/twpol.txt.bak
> /usr/TSS/report
> /usr/TSS/db
> /usr/TSS/key
> /usr/TSS/key/site.key
> /usr/TSS/key/deep.openarch.com-local.key
> /usr/man
> /usr/man/man4
> /usr/man/man4/twconfig.4
> /usr/man/man4/twpolicy.4
> /usr/man/man5
> /usr/man/man5/twfiles.5
> /usr/man/man8
> /usr/man/man8/siggen.8
> /usr/man/man8/tripwire.8
> /usr/man/man8/twadmin.8
> /usr/man/man8/twintro.8
> /usr/man/man8/twprint.8
> /usr/README
> /usr/Release_Notes
> /usr/License.txt
```

Linux Tripwire ASR 1.3.1

概述

随着黑客入侵 Unix 系统的技术越来越熟练，越来越狡猾，也就越来越需要有一种工具能够发现没有经过授权的文件改动。Tripwire 就是这样的工具，可以帮助系统管理员监控一个指定的文件集的任何改变。如果定期地运行 Tripwire，它就会在文件遭到破坏和窜改之后通知系统管理员，这样就能防范于未然。

Tipwire 是一个文件和目录一致性检查器（checker），也就是把指定的一组文件和目录与以前生成的基准数据库中的信息进行比较的软件。任何变化都会被标记并记录下来，包括删除或添加文件或目录。如果定期运行 Tripwire，一旦关键的系统文件被破坏，它就会立即通知你，这样就能及时地采取补救措施。只要 Tipwire 没有报告有变化出现，系统管理员就可以在很大程度上确定指定的一组文件没有发生非法的变化。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

Tripwire 的版本是 1.3.1-1。

软件包的来源

Tripwire 的主页：http://www.tripwiresecurity.com/。

下载：Tripwire-1.3.1-1.tgz。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前

运行一下命令“find /* >trip1”，在编译和安装完软件之后运行命令“find /* > trip2”，最后用命令“diff trip1 trip2 > trip”找出变化。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp Tripwire-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf Tripwire-version.tar.gz
```

编译和优化

转到 Tripwire 的新目录下：

编辑“utils.c”文件（vi +462 src/utils.c），改变下面这一行：

```
else if (iscntrl(*pcin)) {
```

改为：

```
else if (!(*pcin & 0x80) && iscntrl(*pcin)) {
```

编辑“config.parse.c”文件（vi +356 src/config.parse.c），改变下面这一行：

```
rewind(fpout);
```

改为：

```
else {
rewind(fpin);
}
```

编辑“config.h”文件（vi +106 include/config.h），改变这几行：

```
#define CONFIG_PATH "/usr/local/bin/tw"
#define DATABASE_PATH "/var/tripwire"
```

改为：

```
#define CONFIG_PATH "/etc"
#define DATABASE_PATH "/var/spool/tripwire"
```

编辑“config.h”文件（vi +165 include/config.h），改变这一行：

```
#define TEMPFILE_TEMPLATE "/tmp/twzXXXXXX"
```

改为：

```
#define TEMPFILE_TEMPLATE "/var/tmp/.twzXXXXXX"
```

编辑“config.pre.y”文件（vi +66 src/config.pre.y），改变这一行：

```
#ifdef TW_LINUX
```

改为：

```
#ifdef TW_LINUX_UNDEF
```

编辑“Makefile”文件（vi +13 Makefile），并改变这些行：

```
DESTDIR = /usr/local/bin/tw
```

改为：

```
DESTDIR = /usr/sbin
```

```
DATADIR = /var/tripwire
```

改为：

```
DATADIR = /var/spool/tripwire
```

```
LEX = lex
```

改为：

```
LEX = flex
```

```
CC=gcc
```

改为：

```
CC=egcs
```

```
CFLAGS = -O
```

改为：

```
CFLAGS = -O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-pointer -fno-exceptions
```

运行下面的命令：

```
[root@deep]# make
[root@deep]# make install
[root@deep]# chmod 700 /var/spool/tripwire/
[root@deep]# chmod 500 /usr/sbin/tripwire
[root@deep]# chmod 500 /usr/sbin/siggen
[root@deep]# rm -f /usr/sbin/tw.config
```

上面的“make”和“make install”可以配置软件以保证系统中有编译所需要的函数库，然后把所有的源文件都编译成可执行的二进制文件，最后把二进制文件和配置文件安装到相应的目录里。

“chmod”命令把“tripwire”目录的默认权限改成 700（drwx------），也就是可读、可写和能够被超级用户“root”执行。“chmod 500 /usr/sbin/tripwire”命令把二进制文件“/usr/sbin/tripwire”的权限改成可读和可被超级用户“root”执行（-r-x------），最后的“chmod”命令，把“/usr/sbin”目录下的“siggen”程序的权限改成可以被“root”执行和读取。

“rm”命令把“/usr/sbin”目录下的“tw.config”文件删除。我们不需要这个文件，因为我们会在“/etc”目录下创建一个新的。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf tw_ASR_version/ Tripwire-version.tar.gz
```

“rm”命令删除所有编译和安装 Tripwire 所需要的源程序，并且把“/var/tmp”目录下的 Tripwire 软件的压缩包删除掉。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 Tripwire，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“tw.config”文件拷贝到“/etc”目录下
- 把“tripwire.verify”脚本拷贝到“/etc/cron.daily”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到相应的目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

配置 /etc/tw.config 文件

Tripwire 运行在四种模式下：数据库生成、一致性检测、数据库更新和交互式更新。为了进行一致性检测，Tripwire 必须要有一个基准数据库以进行比较。所以，首先必须确定需要 Tripwire 监控哪些文件。这些文件的列表保存在“/etc/tw.config”文件里。“/etc/tw.config”文件里定义了所有需要监控的文件和目录。

第一步

创建“tw.config”文件（touch /etc/tw.config），在文件中加入需要监控的文件和目录。配置文件的格式在它的头部和 man 页“tw.config(5)”中有详细的介绍。

```
# Gerhard Mourani: gmourani@videotron.ca
# last updated: 1999/11/12
# First, root's "home"
/root R
!/root/.bash_history
/ R
# OS itself
/boot/vmlinuz R
# critical boot resources
```

```
/boot R
# Critical directories and files
/chroot R
/etc R
/etc/inetd.conf R
/etc/nsswitch.conf R
/etc/rc.d R
/etc/mtab L
/etc/motd L
/etc/group R
/etc/passwd L
# other popular filesystems
/usr R
/usr/local R
/dev L-am
/usr/etc R
# truncate home
=/home R
# var tree
=/var/spool L
/var/log L
/var/lib L
/var/spool/cron L
!/var/lock
# unusual directories
=/proc E
=/tmp
=/mnt/cdrom
=/mnt/floppy
```

第二步

因为安全方面的原因，用下面的命令把配置文件的权限改为 0600：

```
[root@deep]# chmod 600 /etc/tw.config
```

配置 /etc/tripwire.verify 脚本

通常把 Tripwire 配置成用 email 把报表发送给系统管理员。然而，在正常情况下系统中有一些文件也会发生变化，这样就要更新 Tripwire 的数据库。“tripwire.verify”文件是一个很小的脚本文件，被设计成用 crond 定期执行，用来检查硬盘中文件和目录的变化并用 email 把结果报告给系统管理员。

第一步

创建“tripwire.verify”脚本（vi /etc/cron.daily/tripwire.verify），加入下面这几行：

```
#!/bin/sh
/usr/sbin/tripwire -loosedir -q | (cat <<EOF
This is an automated report of possible file integrity changes, generated by
the Tripwire integrity checker. To tell Tripwire that a file or entire
directory tree is valid, as root run:
/usr/sbin/tripwire -update [pathname|entry]
If you wish to enter an interactive integrity checking and verification
session, as root run:
/usr/sbin/tripwire -interactive
Changed files/directories include:
EOF
cat
) | /bin/mail -s "File integrity report" root
```

第二步

用下面的命令使脚本可执行并且把权限改成 0700：

```
[root@deep]# chmod 700 /etc/cron.daily/tripwire.verify
```

保证 Tripwire 的安全

为了增加系统的安全性，最好把 Tripwire 的数据库（tw.db_[hostname]）文件拷贝到其它地方（如：软盘），这样就不会有人改变它。因为数据库对 Tripwire 很重要，必须小心地处理它。

最好把数据库的内容打印出来。这样的话，如果你对数据库的一致性有怀疑，可以手工把数据库和打印出来的文档进行比较。

更多的资料

如果想查找详细的资料可以用 man 命令查帮助页，读取相关信息：

```
siggen (8) - signature generation routine for Tripwire
tripwire (8) - a file integrity checker for UNIX systems
tw.config (5) - configuration file for Tripwire
```

命令

下面列出的是一些我们经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

在交互式模式下运行 Tripwire

在交互式模式下运行 Tripwire，类似在一致性检测模式下。一旦文件或目录被添加、删除或改变，Tripwire 会问用户是否更新数据库。尽管在这个模式下可以很方便地更新数据库，但是要求用户始终“在键盘边上”。

第一步

首先要运行“tripwire --initialize”命令。这个命令会在你设定的目录下创建名为“tw.db_[hostname]”的数据库文件（这里[hostname]是计算机的主机名）。为了进行一致性检测，Tripwire 需要一个基准数据库。

用下面的命令创建基准数据库：

```
[root@deep]# cd /var/spool/tripwire/
[root@deep]# /usr/sbin/tripwire --initialize
```

我们转到包含数据库文件的目录下，然后创建用于一致性检测的基准数据库。

第二步

有两种更新 Tripwire 数据库的方法。第一种方法是交互式的，Tripwire 提示用户是否更新数据库以反应系统当前的状态。第二种方法是命令行方式的，在运行的时候指定哪些项需要更新。

用下面的命令运行在交互式模式下：

```
[root@deep]# cd /var/spool/tripwire/database/
[root@deep]# cp tw.db_myserverhostname /var/spool/tripwire/
[root@deep]# cd ..
[root@deep]# /usr/sbin/tripwire --interactive
Tripwire(tm) ASR (Academic Source Release) 1.3.1
File Integrity Assessment Software
(c) 1992, Purdue Research Foundation, (c) 1997, 1999 Tripwire
Security Systems, Inc. All Rights Reserved. Use Restricted to
Authorized Licensees.
### Phase 1: Reading configuration file
```

```
### Phase 2: Generating file list
### Phase 3: Creating file information database
### Phase 4: Searching for inconsistencies
###
### Total files scanned: 15722
### Files added: 34
### Files deleted: 42
### Files changed: 321
###
### Total file violations: 397
###
added: -rwx------ root 22706 Dec 31 06:25:02 1999 /root/tmp/firewall
---> File: '/root/tmp/firewall'
---> Update entry? [YN(y)nh?]
```

在交互式模式下，Tripwire 会报告被添加、删除和改变得文件，而且允许用户交互式更新数据库中的每一项。

在数据库更新模式下运行 Tripwire

Tripwire 支持基于每文件/目录和“tw.config”项的增量数据库更新。Tripwire 在数据库中存储和“tw.config”中的项相关的信息。把在交互式模式下运行 Tripwire 和用“tripwire.verify”脚本文件把结果用 email 报告给系统管理员结合起来，可以节省扫描整个系统的时间。为了避免在交互式模式下运行 Tripwire 而且在扫描结束之前等上很长的时间，“tripwire.verify”脚本文件可以扫描整个系统并且用 email 报告结果，然后系统管理员可以在数据库更新模式下进行数据库的操作。

例如：

有一个文件发生变化，并且这个变化是正常的，可以用下面这个命令更新数据库：

```
[root@deep]# tripwire -update /etc/newly.installed.file
```

或者，如果在“tw.config”文件中的整组文件都发生了变化，可以用下面的命令来更新：

```
[root@deep]# tripwire -update /usr/lib/Package_Dir
```

在这两种情况下，Tripwire 为指定的文件重新生成数据库中的记录项。旧的数据库会被备份到“./databases”的目录下。

安装到系统中的文件

```
> /etc/cron.daily/tripwire.verify
> /etc/tw.config
> /usr/man/man5/tw.config.5
> /usr/man/man8/siggen.8
> /usr/man/man8/tripwire.8
> /usr/sbin/tripwire
> /usr/sbin/siggen
> /var/spool/tripwire
> /var/spool/tripwire/tw.db_TEST
```

Tripwire 的替代软件

ViperDB

ViperDB 比 Tripwire 小而且速度快。ViperDB 也不使用“all-in-one”的数据库，而是在每一个被监控的目录下创建纯文本的数据库。这样入侵者就没有单独的攻击点。如果每 5 分钟运行一次 ViperDB（用 cron），就能减小入侵者改变被监控的文件或目录的可能性。

软件包

ViperDB 的主页：http://www.resentment.org/projects/viperdb/。

FCHECK

FCHECK 是一个非常稳定的 PERL 脚本，用来监控任何文件的变化并用 syslog、console 或其它报告系统管理员。如果系统的磁盘空间足够小，就可以每隔一分钟监控一次，这样就能很好地保证系统的安全性。FCHECK 是“Tripwire”的替代品，而且是开放源代码、经过长时间测试并且更容易配置和使用。

软件包

FCHECK 的主页：http://sites.netscape.net/fcheck/fcheck.html。

Sentinel

Sentinel 是快速的文件/磁盘扫描器，类似于 Tripwire 和 Viper.pl。使用与 Tripwire 相似的数据库，但是使用 RIPEMD-160bit MAC 校验算法（不受版权限制），而且比受版权限制的 MD5 128bit 校验码更安全。它可以在大部分的 Unix 下运行。

软件包

sentinel 的主页：http://zurk.netpedia.net/zfile.html。

Linux GnuPG

概述

GnuPG（GNU Privacy Guard）是保证数据传输和存储安全的 GNU 工具。它可以用于加密数据和创建数字签名。GnuPG 包括了高级的密匙管理工具而且遵循 RFC2440 中描述的 OpenPGP 的国际标准。

因为 GnuPG 不使用任何有专利限制的算法，所以它和 PGP2 的各个版本并不兼容。PGP 2.x 只使用 IDEA（在全世界范围享有专利）和 RSA（在美国的专利将在 2000 年 9 月 20 日终止）。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

GnuPG 的版本是 1.0.1。

软件包的来源

GnuPG 的主页：http://www.gnupg.org/。

下载：gnupg-1_0_1_tar.gz。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >pg1”，在编译和安装完软件之后运行命令“find /* > pg2”，最后用命令“diff pg1 pg2 > pg”找出变化。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp gnupg-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf gnupg-version.tar.gz
```

编译和优化

转到 GnuPG 的新目录下，先设置编译器的编译参数：

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-
pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--enable-shared
```

运行下面的命令：

```
[root@deep]# make
[root@deep]# make check
[root@deep]# make install
[root@deep]# strip /usr/bin/gpg
```

“make”命令把所有的源文件编译成可执行的二进制文件，“make check”在安装之前进行测试，最后“make install”命令把所有的二进制文件安装在相应的目录里。“strip”命令可以把“gpg”程序的文件大小缩小，这样可以提高程序的性能。

清除不必要的文件

用下面的命令删除不必要的文件：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf gnupg-version/ gnupg-version.tar.gz
```

“rm”命令删除所有编译和安装 GnuPG 所需要的源程序，并且把“/var/tmp”目录下的 GnuPG 软件的压缩包删除掉。

命令

下面列出的是一些经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

创建密匙

第一次使用 GnuPG 软件的时候，我们必须先给自己创建一对新的密匙，

第一步：

用下面的命令创建一对新的密匙：

```
[root@deep]# gpg --gen-key


gpg (GnuPG) 1.0.1; Copyright (C) 1999 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.
gpg: /root/.gnupg: directory created
gpg: /root/.gnupg/options: new options file created
gpg: you have to start GnuPG again, so it can read the new options file
This asks some questions and then starts key generation.
```

第二步：

用下面的命令再次运行 GnuPG（以 root 身份）：

```
[root@deep]# gpg --gen-key


gpg (GnuPG) 1.0.1; Copyright (C) 1999 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.
gpg: /root/.gnupg/secring.gpg: keyring created
gpg: /root/.gnupg/pubring.gpg: keyring created
Please select what kind of key you want:
(1) DSA and ElGamal (default)
(2) DSA (sign only)
(4) ElGamal (sign and encrypt)
Your selection? 1
DSA keypair will have 1024 bits.
About to generate a new ELG-E keypair.
```

```
minimum keysize is 768 bits
default keysize is 1024 bits
highest suggested keysize is 2048 bits
What keysize do you want? (1024) 2048
Do you really need such a large keysize? y
Requested keysize is 2048 bits
Please specify how long the key should be valid.
0 = key does not expire
<n> = key expires in n days
<n>w = key expires in n weeks
<n>m = key expires in n months
<n>y = key expires in n years
Key is valid for? (0) 0
correct (y/n)? y
You need a User-ID to identify your key; the software constructs the user id
from Real Name, Comment and Email Address in this form:
"Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"
Real name: Gerhard Mourani
Email address: gmourani@videotron.ca
Comment: [Press Enter]
You selected this USER-ID:
"Gerhard Mourani <gmourani@videotron.ca>"
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
You need a Passphrase to protect your secret key.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
+++++..+++++++++++++++..+++++.++++++++++++++++++++++++++++++++++++++..++++
+++
+++.++++++++++++++++++++++++.+++++++++++++++...++++++++++++++++++++++++.++
++
+
..+++++>+++++...+++++++++++++++>+++++.......>+++++.......<+++++.............
...
.........+++++^^^^
public and secret key created and signed.
```

在“root”的家目录下（~/root），一对新的密匙（私有密匙和公用密匙）被创建出来。

导入密匙

如果你收到了别人的公用密匙，可以把它放在你的公用密匙数据库里，这样就能方便地使用这些密匙。这就叫做“导入”（importing）。

用下面的命令把公用密匙导入公用密匙库：

```
[root@deep]# gpg --import <file>
```

例如：

```
[root@deep]# gpg --import redhat2.asc
```

```
gpg: key DB42A60E: public key imported
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: Total number processed: 1
gpg: imported: 1
```

新的密匙被加入到密匙库里而且密匙库被重新更新。请注意 GnuPG 不会导入不能自我签定（self-signed）的密匙。在上面的例子中我们把从 RedHat 网站下载下来的公用密匙文件“redhat2.asc”导入我们的密匙库。

签定密匙 key signing

当导入公用密匙并且确定这个人就是其本人（不是别人冒冲的），我们就能签订他的密匙：

用下面的命令签订上面例子中我们加到密匙库的 RedHat 公司的密匙：

```
[root@deep]# gpg --sign-key <UID>
```

例如：

```
[root@deep]# gpg --sign-key RedHat
```

```
pub 1024D/DB42A60E created: 1999-09-23 expires: never trust: -/q
sub 2048g/961630A2 created: 1999-09-23 expires: never
(1) Red Hat, Inc <security@redhat.com>
pub 1024D/DB42A60E created: 1999-09-23 expires: never trust: -/q
Fingerprint: CA20 8686 2BD6 9DFC 65F6 ECC4 2191 80CD DB42 A60E
Red Hat, Inc <security@redhat.com>
Are you really sure that you want to sign this key
```

```
with your key: "Gerhard Mourani <gmourani@videotron.ca>"
Really sign? y
You need a passphrase to unlock the secret key for
user: "Gerhard Mourani <gmourani@videotron.ca>"
1024-bit DSA key, ID E92D6C97, created 1999-12-30
Enter passphrase:
```

只有在决对确信这个密匙是可信的，才能签订这个密匙。千万别贸然行事。

加密和解密

在根据我们的需要安装和配置完 GnuPG 之后，可以开始加密和解密了。

用下面的命令，可以从公用密匙库中取出 RedHat 的密匙进行加密并签订数据：

```
[root@deep]# gpg -sear RedHat <file>
```

例如：

```
[root@deep]# gpg -sear RedHat message-to-RedHat.txt


You need a passphrase to unlock the secret key for
user: "Gerhard Mourani (Open Network Architecture) <gmourani@videotron.ca>"
1024-bit DSA key, ID BBB4BA9B, created 1999-10-26
Enter passphrase:
```

“s”表示签订（signing）（为了避免有人冒冲你，最好签订所有加密的数据），“e”表示加密（encrypting），“a”表示创建 ASCII 字符的输出（“.asc”是为了方便用 email 传送），“r”加密用户的标识名，<file>表示想要加密的文件名。

用下面的命令解密数据：

```
[root@deep]# gpg -d <file>
```

例如：

```
[root@deep]# gpg -d message-to-Gerhard.asc


You need a passphrase to unlock the secret key for
user: "Gerhard Mourani (Open Network Architecture) <gmourani@videotron.ca>"
2048-bit ELG-E key, ID 71D4CC44, created 1999-10-26 (main key ID BBB4BA9B)
Enter passphrase:
```

“-d”表示加密，<file>表示想要加密的文件名。

导入密匙

GnuPG 有一些选项有助于发布公用密匙。这叫做“导出”密匙。通过导出公用密匙，可以让更多的人知道你的公用密匙。可以把密匙在主页上，或密匙服务器，或用其它方式公开出来。

用下面的命令可以导出 ASCII 形式的公用密匙：

```
[root@deep]# gpg --export --armor > Public-key.asc
```

“--export”表示从公用密匙库取出你的公用密匙，“--armor”表示创建 ASCII 形式的密匙，这样就能把它放在用邮件、出版物或主页上了。“>Public-key”表示把标准输出重定向到“Public-key.asc”文件里。

签定和检查

每个知道你的公用密匙（可以把公用密匙放在密匙服务器或主页上）的人都可以检查是否真的是你签订这个文件。

用下面的命令签定文本或二进制文件：

```
[root@deep]# gpg --detach-sign --armor <Data>
```

“--detach-sign”表示签名放在别的文件里。最好是这样做，特别是签订二进制文件的时候。“--armor”表示的意思和上面介绍的一样，在这里非常有用。<data>表示文件或二进制文件，如：tar 文件。

用下面的命令检查加密数据的签名：

```
[root@deep]# gpg --verify <Data>
```

GnuPG 检查签名是否是有效的并显示出相应的信息。如果签名是有效的，还必须知道公用密匙所对应的私有密匙。如果加密的数据被签订了，当数据被解密的时候就要检查签名。当然还必须拥有需要发送加密信息的人的公用密匙。

安装到系统中的文件

```
> /usr/bin/gpg
> /usr/lib/gnupg
> /usr/lib/gnupg/rndunix
```

```
> /usr/lib/gnupg/rndegd
> /usr/lib/gnupg/tiger
> /usr/man/man1/gpg.1
> /usr/share/gnupg
> /usr/share/gnupg/options.skel
```

第十章

服务器软件

Linux DNS and BIND 服务器

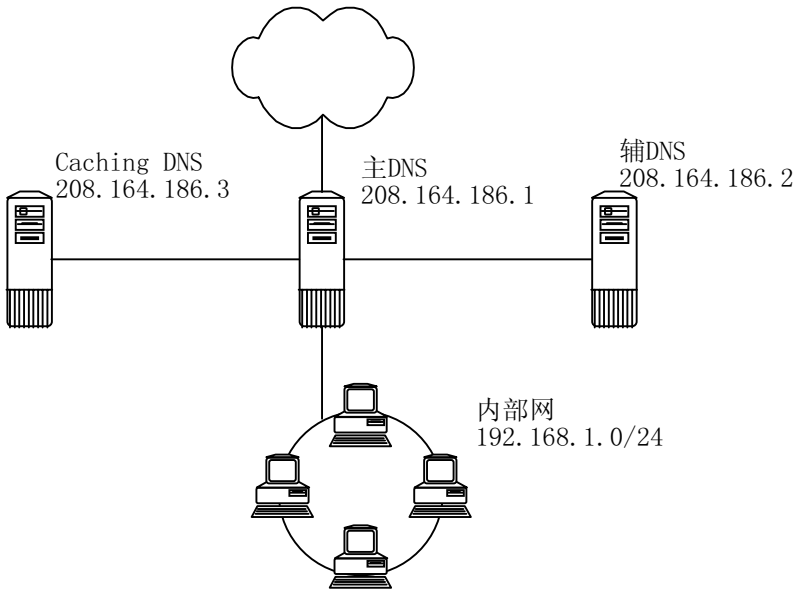
概述

安装完所有必要的系统安全软件之后，下一步要考虑如何提高和调整服务器的网络性能。DNS 对 IP 网络通讯来说是最重要的网络服务，因为所有的 Linux 客户机（client）至少都要配置成具有缓冲（caching）能力。为本地的客户端计算机建立缓冲服务可以降低主域名服务器（primary server）的负荷。一台“caching only”的域名服务器可以进行域名解析并把结果记录下来下次还要用到这个域名的时候就不要花很长时间解析了。这可以在很大程度上缩短下次域名解析的时间。

因为安全方面的问题，保证在企业内部网的主机和企业外部的主机之间不存在 DNS 这一点十分重要。如果只用 IP 地址连接企业外部的主机，当然就更安全了。

在我们的配置和安装中我们以非“root”用户在“chrooted”环境中运行 BIND/DNS。我们提供了三种不同的配置：1）简单的“caching only”域名服务器（客户端）；2）辅域名服务器（secondary server）；3）主域名服务器（primary server）。

配置简单的“caching only”域名服务的服务器是不会用来做主/辅域名服务器的，只有主域名服务器或辅域名服务器才需要配置主域名或辅域名服务。通常情况下只有一台服务器作为主域名服务器，另一台服务器作为辅域名服务器，其它的服务器都作为简单的“caching only”客户端域名服务器。



上面是我们在这本书中介绍的 DNS 的配置的图示。我们在不同的服务器上使用了不同的设置（caching only DNS、主 DNS 和辅 DNS）。在实际情况中有很多种可能存在，所以要根据需要和网络结构来配置。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

bind 的版本号是 8.2.2-patchlevel5。

软件包的来源

bind 主页：http://www.isc.org/。

下载：bind-contrib.tar.gz, bind-doc.tar.gz, bind-src.tar.gz。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >dns1”，在编译和安装完软件之后运行命令“find /* > dns2”，最后用命令“diff dns1 dns2 > dns”找出变化。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep /]# mkdir /var/tmp/bind
[root@deep /]# cp bind-contrib.tar.gz /var/tmp/bind/
[root@deep /]# cp bind-doc.tar.gz /var/tmp/bind/
[root@deep /]# cp bind-src.tar.gz /var/tmp/bind/
```

我们创建了一个名为“bind”的目录，用来处理 tar 文档。

转到新的“bind”目录（cd /var/tmp/bind），解压 tar 文件：

```
[root@deep bind]# tar xzpf bind-contrib.tar.gz
[root@deep bind]# tar xzpf bind-doc.tar.gz
[root@deep bind]# tar xzpf bind-src.tar.gz
```

配置和优化

编辑“Makefile.set”文件（vi /src/port/linux/Makefile.set），并加入：

```
'CC=egcs -D_GNU_SOURCE'
'CDEBUG=-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-
frame-pointer -fno-exceptions'
'DESTBIN=/usr/bin'
'DESTSBIN=/usr/sbin'
'DESTEXEC=/usr/sbin'
'DESTMAN=/usr/man'
'DESTHELP=/usr/lib'
'DESTETC=/etc'
'DESTRUN=/var/run'
'DESTLIB=/usr/lib/bind/lib'
'DESTINC=/usr/lib/bind/include'
```

```
'LEX=flex -8 -I'
'YACC=yacc -d'
'SYSLIBS=-lfl'
'INSTALL=install'
'MANDIR=man'
'MANROFF=cat'
'CATEXT=$$N'
'PS=ps -p'
'AR=ar crus'
'RANLIB=:'
```

第一行说明我们用的gcc编译器的名字是egcs，第二行是优化参数。“DESTLIB=”这一行说明bind所需的库函数目录，“DESTING=”说明bind的“include”目录在哪里。

编译和优化

输入下面的命令：

```
[root@deep bind]# make -C src
[root@deep bind]# make clean all -C src SUBDIRS=../doc/man
[root@deep bind]# make install -C src
[root@deep bind]# make install -C src SUBDIRS=../doc/man
```

“make”命令把所有的源文件都编译成二进制文件，接着“make install”把二进制文件和相关的配置文件安装到相应的目录中。

```
[root@deep bind]# strip /usr/bin/addr
[root@deep bind]# strip /usr/bin/dig
[root@deep bind]# strip /usr/bin/dnsquery
[root@deep bind]# strip /usr/bin/host
[root@deep bind]# strip /usr/bin/nslookup
[root@deep bind]# strip /usr/bin/nsupdate
[root@deep bind]# strip /usr/bin/mkservdb
[root@deep bind]# strip /usr/sbin/named
[root@deep bind]# strip /usr/sbin/named-xfer
[root@deep bind]# strip /usr/sbin/ndc
[root@deep bind]# strip /usr/sbin/dnskeygen
[root@deep bind]# strip /usr/sbin/irpd
[root@deep bind]# mkdir /var/named
```

“strip”命令去掉目标文件中的所有符号信息。这样二进制程序就会小一些，可以提高一点程序的性能。“mkdir”命令创建一个“/var/named”目录。

清除不必要的文件

```
[root@deep /]# cd /var/tmp
[root@deep tmp]# rm -rf bind/
```

这些命令删除用来编译和安装 BIND/DNS 的源文件。

配置

不同服务器的配置文件是不相同的，要根据需求和网络结构的实际情况来决定。可能在家里只要安装一个“caching only”的 DNS 服务器，在公司里就可能要安装主 DNS、辅 DNS 和“caching only”DNS。

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行“caching only”域名服务器，必须创建或者拷贝下面的文件到相应的目录下：

- 把“named.conf”文件拷贝到“/etc”目录下
- 把“db.127.0.0”文件拷贝到“/var/named”目录下
- 把“db.cache”文件拷贝到“/var/named”目录下
- 把“named”脚本文件拷贝到“/etc/rc.d/init.d”目录下

为了运行主域名服务器，必须创建或者拷贝下面的文件到相应的目录下：

- 把“named.conf”文件拷贝到“/etc”目录下
- 把“db.127.0.0”文件拷贝到“/var/named”目录下
- 把“db.cache”文件拷贝到“/var/named”目录下
- 把“db.208.164.186”文件拷贝到“/var/named”目录下
- 把“db.openarch”文件拷贝到“/var/named”目录下
- 把“named”脚本文件拷贝到“/etc/rc.d/init.d”目录下

为了运行辅域名服务器，必须创建或者拷贝下面的文件到相应的目录下：

- 把“named.conf”文件拷贝到“/etc”目录下
- 把“db.127.0.0”文件拷贝到“/var/named”目录下
- 把“db.cache”文件拷贝到“/var/named”目录下
- 把“named”脚本文件拷贝到“/etc/rc.d/init.d”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到相应的目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

Caching－only 域名服务器

“caching-only”域名服务器处除了对“0.0.127.in-addr.arpa”有效，对其它的域都是无效的。“caching-only”域名服务器能够在区域（zone）内或在区域外查找域名，就像主域名服务器和辅域名服务器那样。所不同的是当“caching-only”域名服务器开始在区域内查找域名的时候，只查询本区域的一台主域名服务器或辅域名服务器就结束了。

用来建立简单的“caching-only”域名服务器所需的文件为：

- named.conf
- db.127.0.0
- db.cache
- named 脚本

为简单的 caching 服务器配置 /etc/named.conf 文件

只要不作为主域名服务器和辅域名服务器，可以在网络上的其它服务器上使用这一个配置。为本地的客户端计算机建立简单的“caching”服务器可以减轻主域名服务器的负荷。许多拨号上网的用户也是因为这个原因使用这种配置。

创建“named.conf”文件（touch /etc/named.conf），在文件中加入这些行：

```
options {
directory "/var/named";
forwarders { 208.164.186.1; 208.164.186.2; };
forward only;
};
//
```

```
// a caching only nameserver config
zone "." in {
type hint;
file "db.cache";
};
zone "0.0.127.in-addr.arpa" in {
type master;
file "db.127.0.0";
};
```

在“forwarders”这一行，“208.164.186.1”和“208.164.186.2”是主域名服务器和辅域名服务器的 IP 地址。也可以用 ISP 提供的 DNS 服务器或者其它的 DNS 服务器。

为了提高 BIND/DNS 服务器的安全性，需要防止当主/辅域名服务器当机或者没有反应的时候，“caching”域名服务器会试图查询其它服务器。在“named.conf”文件中设置了“forward only”这个参数，可以防止这种情况的发生，也就是当“forwarders”中定义的域名服务器没有反应的时候，域名服务器不会去查询其它服务器。

为简单的 caching 服务器配置 /var/named/db.127.0.0 文件

只要不作为主域名服务器和辅域名服务器，可以在网络上的其它服务器上使用这一个配置。“db.127.0.0”文件包括自转（loopback）网络。在“/var/named”目录下创建下面的文件。

创建“db.127.0.0”文件（touch /var/named/db.127.0.0），在文件中加入下面这些行：

```
$TTL 345600
@ IN SOA localhost. root.localhost. (
00 ; Serial
86400 ; Refresh
7200 ; Retry
2592000 ; Expire
345600 ) ; Minimum
IN NS localhost.
1 IN PTR localhost.
```

为简单的 caching 服务器配置 /var/named/db.cache 文件

在启动 DNS 服务器之前必须要有一个“db.cache”文件的拷贝，把这个文件拷贝到“/var/named”目录下。“db.cache”告诉服务器根区域（root zone）在哪里。

用下面的命令在其它的 Unix 计算机上创建“db.cache”文件，或者从 RedHat Linux 源代码的光盘中拷贝一个。

用下面的命令生成一个新的“db.cache”文件：

```
[root@deep]# dig @.aroot-servers.net . ns > db.cache
```

不要忘了把“db.cache”文件拷贝到需要这个文件的服务器的“/var/named”目录下。

注意：内部网的地址，如“192.168.1/24”因为安全问题不能出现在 DNS 的配置文件中。还有一点很重要，就是在企业网内的主机和外部主机之间不能存在 DNS。

主域名服务器

主域名服务器从文件中读取区域数据，这些数据对整个区域来说都是具有权威性的。

为了建立主域名服务器所必须的文件是：

- named.conf
- db.127.0.0
- db.208.164.186
- db.openarch
- db.cache
- named script

为主域名服务器配置 /etc/named.conf 文件

在作为主域名的服务器中使用这样的配置。编译完 DNS 之后，需要为服务器建立主域名。我们用“openarch.com”作为力争，假定其 IP 地址为：208.164.186.0。为了实现这个目的在“/etc/named.conf”文件中加入这几行：

```
options {
directory "/var/named";
fetch-glue no;
recursion no;
allow-query { 208.164.186/24; 127.0.0/8; };
allow-transfer { 208.164.186.2; };
transfer-format many-answers;
};
// These files are not specific to any zone
zone "." in {
type hint;
file "db.cache";
};
zone "0.0.127.in-addr.arpa" in {
type master;
file "db.127.0.0";
};
// These are our primary zone files
zone "openarch.com" in {
type master;
file "db.openarch";
};
zone "186.164.208.in-addr.arpa" in {
type master;
file "db.208.164.186";
};
```

“fetch-glue no”与“recursion no”选项一起使用以防止服务器的缓冲区增长的过大以至崩溃。禁止递归选项将你的服务器设成被动模式，它不会代理其它名称服务器和解释器而发送请求。禁止递归的名称服务器因为不会发送请求因此不会缓存任何数据，也就很难被欺骗。这是一个安全特性。

在“allow-query”行中“208.164.186/24”和“127.0.0/8”表示本服务器允许向其请求的 IP 地址。

在“allow-query”行中“208.164.186.2”表示本服务器允许接收区带转移的 IP 地址。你必须确保只有辅域名服务器才可以接收你的区带转移。这些信息往往被一些 IP 欺骗程序所利用。

注意：“named.conf”文件中选项“recursion no”，“allow-query”和“allow-transfer”是一些安全方面的特性。

为主域名服务器和辅域名服务器配置 /var/named/db.127.0.0 文件

本配置文件适用于主域名服务器和辅域名服务器。“db.127.0.0”文件包括回环（loopback）网络的设置，是主机用于将信息流直接导向本地的特殊地址。在“/var/named/”目录中创建下列文件。

创建“db.127.0.0”文件（touch /var/named/db.127.0.0）并加上如下几行：

```
; Revision History: April 22, 1999 - admin@mail.openarch.com
; Start of Authority (SOA) records.
$TTL 345600
@   IN  SOA     deep.openarch.com.      admin.mail.openarch.com. (
00      ; Serial
86400   ; Refresh
7200    ; Retry
2592000         ; Expire
345600 )        ; Minimum
; Name Server (NS) records.
NS  deep.openarch.com.
NS  mail.openarch.com.
; only One PTR record.
1   PTR     localhost.
```

为主域名服务器和辅域名服务器配置 /var/named/db.208.164.186 文件

本配置文件适用于主域名服务器。文件“db.208.164.186”用于映射主机名和 IP 地址。在“/var/named/”中创建下列文件：

创建“db.208.164.186”文件（touch /var/named/db.208.164.186）并加入：

```
; Revision History: April 22, 1999 - admin@mail.openarch.com
; Start of Authority (SOA) records.
$TTL 345600
@   IN  SOA     deep.openarch.com.      admin.mail.openarch.com. (
00      ; Serial
86400   ; Refresh
7200    ; Retry
2592000     ; Expire
345600 )    ; Minimum
; Name Server (NS) records.
```

```
NS deep.openarch.com.
NS mail.openarch.com.
; Addresses Point to Canonical Names (PTR) for Reverse lookups
1   PTR     deep.openarch.com.
2    PTR    mail.openarch.com.
3   PTR     www.openarch.com.
```

为主域名服务器配置 /var/named/db.openarch 文件

本配置文件适用于主域名服务器。文件“db.openarch”用于映射主机名和 IP 地址。在“/var/named/”中创建下列文件：

创建“db.openarch”文件（touch /var/named/db.openarch）并加入：

```
; Revision History: April 22, 1999 - admin@mail.openarch.com
; Start of Authority (SOA) records.
$TTL 345600
@   IN  SOA     deep.openarch.com.      admin.mail.openarch.com. (
00          ; Serial
86400       ; Refresh
7200        ; Retry
2592000         ; Expire
345600 )        ; Minimum
; Name Server (NS) records.
NS deep.openarch.com.
NS mail.openarch.com.
; Mail Exchange (MX) records.
MX  0   mail.openarch.com.
; Address (A) records.
localhost       A   127.0.0.1
deep        A   208.164.186.1
mail            A   208.164.186.2
www     A   208.164.186.3
; Aliases in Canonical Name (CNAME) records.
;www        CNAME   deep.openarch.com.
```

为主域名服务器和辅域名服务器配置 /var/named/db.cache 文件

在你开始启动 DNS 服务器之前你必须在“/var/named/”目录中拥有文件“db.cache”。“db.cache”文件告诉你的服务器你的服务器的“根”区带的位置。

用以下的命令从其它的 UNIX 主机上（你的主域名服务器）或者从你的 Red Hat Linux CD-ROM 发行源盘获取一份拷贝。

```
[root@deep]# dig @.aroot-servers.net . ns > db.cache
```

获取文件之后不要忘记把 db.cache 拷贝到你的 DNS 服务器“/var/named/”目录下。

二级域名服务器（辅域名服务器）

二级域名服务器平时可以分担主域名服务器的负载。当主服务器当机时接管主服务器的工作。二级域名服务器可以从网络上接收别的域名服务器传来的数据（通常是主域名服务器）。这个过程叫区带转移。

建立一个二级域名服务器所需的文件包括：

- named.conf
- db.127.0.0
- db.cache
- named 脚本

为二级域名服务器配置 /etc/named.conf 文件

本配置文件适用于二级域名服务器。你必须修改二级域名服务器主机上的“named.conf”文件。把除了“0.0.127.in-addr.arpa”以外每次出现的“primary”改为“secondary”并添上主域名服务器和其 IP 地址的记录。

创建“named.conf”文件（touch /etc/named.conf）并加上如下几行：

```
options {
directory "/var/named";
fetch-glue no;
recursion no;
allow-query { 208.164.186/24; 127.0.0/8; };
allow-transfer { 208.164.186.1; };
transfer-format many-answers;
};
// These files are not specific to any zone
zone "." in {
type hint;
file "db.cache";
```

```
};
zone "0.0.127.in-addr.arpa" in {
type master;
file "db.127.0.0";
};
// These are our slave zone files
zone "openarch.com" in {
type slave;
file "db.openarch";
masters { 208.164.186.1; };
};
zone "186.164.208.in-addr.arpa" in {
type slave;
file "db.208.164.186";
masters { 208.164.186.1; };
};
```

以上设置此域名服务器为区带“openarch.com”二级域名服务器，而且它应该与“208.164.186.1”主机所保存的区带信息保持一致。

二级域名服务器不必从网络上获取所有的 db 文件；总括文件，“db.127.0.0”和“db.cache”,与主域名服务器相同，所以只须在二级域名服务器中保留一份本地拷贝。

从主域名服务器中拷贝“db.127.0.0”文件到二级域名服务器。

从主域名服务器中拷贝“db.cache”文件到二级域名服务器。

为所有名称服务器配置 /etc/rc.d/init.d/named 脚本文件

本配置文件适用于所有类型的名称服务器（缓存、主、辅名称服务器）。配置“/etc/rc.d/init.d/named”脚本文件用以启动和停止 DNS/BIND 服务器守护进程。

创建“named”脚本文件 (touch /etc/rc.d/init.d/named) 并添加：

```
#!/bin/sh
#
# named This shell script takes care of starting and stopping
# named (BIND DNS server).
#
# chkconfig: - 55 45
# description: named (BIND) is a Domain Name Server (DNS) \
# that is used to resolve host names to IP addresses.
# probe: true
# Source function library.
. /etc/rc.d/init.d/functions
```

```
# Source networking configuration.
. /etc/sysconfig/network
# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0
[ -f /usr/sbin/named ] || exit 0
[ -f /etc/named.conf ] || exit 0
RETVAL=0
# See how we were called.
case "$1" in
start)
# Start daemons.
echo -n "Starting named: "
daemon named
RETVAL=$?
[ $RETVAL -eq 0 ] && touch /var/lock/subsys/named
echo
;;
stop)
# Stop daemons.
echo -n "Shutting down named: "
killproc named
RETVAL=$?
[ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/named
echo
;;
status)
/usr/sbin/ndc status
exit $?
;;
restart)
$0 stop
$0 start
;;
reload)
/usr/sbin/ndc reload
exit $?
;;
probe)
# named knows how to reload intelligently; we don't want linuxconf
# to offer to restart every time
/usr/sbin/ndc reload >/dev/null 2>&1 || echo start
exit 0
;;
*)
```

```
echo "Usage: named {start|stop|status|restart}"
exit 1
esac
exit $RETVAL
```

好了，现在更改脚本的缺省权限使其可以执行：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/named
```

通过执行以下命令创建 BIND/DNS 到 rc.d 的符号链接：

```
[root@deep]# chkconfig --add named
```

BIND/DNS 脚本不会在下次重起机器的时候自动启动 named 守护进程。你可以通过执行以下命令使它成为缺省设置：

```
[root@deep]# chkconfig --level 345 named on
```

执行以下命令手工启动 DNS 服务器：

```
[root@deep]# /etc/rc.d/init.d/named start
```

加强 BIND/DNS 安全性

限制 BIND 运行于 虚 根 chroot 环境下

本部分侧重于防止 BIND 被黑客当作攻击运行它的主机系统的切入点。BIND 要完成一项相当复杂和大型的功能，因此其安全方面的潜在漏洞出现的概率就比较高。事实上，人们已经发现 BIND 中的缺陷可以让一个远程的用户在一台运行 BIND 的主机上获取根的访问权限。

BIND 可以作为非 root 用户运行，这样可以把风险降到最低，可以把损害限制在一个普通用户在本地 Shell 所能造成的恶劣后果之内。当然，允许匿名访问用户也会极大影响大多数 DNS 系统的安全性，因此有必要采用进一步措施来对付它——在“虚”根环境中运行 BIND。

采用“虚”根环境最主要的优势在于它可以限制守护进程所能见到的文件系统在“虚”根之内。而且，因为这个“虚”根仅须支持 BIND，“虚”根内所须的程序就非常有限。最为重要的是没有必要让转换为根用户的程序运行，可以避免通过获取根用户权限的办法来打破“虚”根的限制。

注意：“named”程序必须位于你的 PATH 环境变量所列出的目录之内。如果你是根用户，而且已经安装了 BIND，这肯定不成问题。在其他情况下，我们假定你的 named 程序位于“/usr/sbin/named”。

找出支持运行“named”所须的共享库文件，这些文件需要拷进“虚”根之内。

```
[root@deep]# ldd /usr/sbin/named
libc.so.6 => /lib/libc.so.6 (0x40017000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

记下上面所列之文件，过一会儿会用到它们。

第一步：

为运行“named”添加一个新的用户名和用户组。这是非常重要的，因为以根用户运行 BIND 会使“虚”根毫无意义。而用一个现存的用户帐号容易造成服务之间互相访问对方资源，不利于多层次的安全性。

下面是用户和组 id 的例子，检查一下“/etc/passwd”和“/etc/group”两个文件找出尚未使用的用户和组 id，我们将采用 53。

```
[root@deep]# groupadd -g 53 named
[root@deep]# useradd -g 53 -u 53 named
```

第二步：

设置“虚”根环境。首先创建一个“虚”根目录。我们将选择“/chroot/named”，因为我们将它设立在隔离的文件系统内以防止文件系统遭到攻击。在早些时候我们的安装过程中为这个目的创建了一个特别的分区“/chroot”。

```
[root@deep]# /etc/rc.d/init.d/named stop (仅当 named 守护进程在运行时)
[root@deep]# mkdir -p /chroot/named
```

接着，如下创建其余的目录：

```
[root@deep]# mkdir /chroot/named/dev
[root@deep]# mkdir /chroot/named/lib
[root@deep]# mkdir /chroot/named/etc
[root@deep]# mkdir -p /chroot/named/usr/sbin
[root@deep]# mkdir -p /chroot/named/var/run
```

拷贝主要配置文件，带区文件，named，named-xfer 程序：

```
[root@deep]# cp /etc/named.conf /chroot/named/etc/
[root@deep]# mkdir /chroot/named/var/named
[root@deep]# cd /var/named ; cp -a . /chroot/named/var/named/
[root@deep]# mknod /chroot/named/dev/null c 1 3
[root@deep]# chmod 666 /chroot/named/dev/null
[root@deep]# cp /usr/sbin/named /chroot/named/usr/sbin/
[root@deep]# cp /usr/sbin/named-xfer /chroot/named/usr/sbin/
```

重要提示：目录“/chroot/named/var/named”及其目录内的所有文件的属主必须设成二级域名服务器下名字为“named”的进程，并且只有二级域名服务器可以进行区带转移，而你自己不能直接操作。

欲使目录“/chroot/named/var/named”及其目录下的所有文件的属主设成二级域名服务器下名字为“named”的进程，执行以下命令：

```
[root@deep]# chown -R named.named /chroot/named/var/named/
```

为“named.conf”设置不可更改位：

```
[root@deep]# cd /chroot/named/etc/
[root@deep]# chattr +i named.conf
```

拷贝前面查找出的共享库文件到“虚”根下的 lib 目录：

```
[root@deep]# cp /lib/libc.so.6 /chroot/named/lib/
[root@deep]# cp /lib/ld-linux.so.2 /chroot/named/lib/
```

拷贝“localtime” and “nsswitch.conf”文件到“虚”根下以使日志文件的条目符合当地时间：

```
[root@deep]# cp /etc/localtime /chroot/named/etc/
[root@deep]# cp /etc/nsswitch.conf /chroot/named/etc/
```

为“nsswitch.conf”设置不可更改位：

```
[root@deep]# cd /chroot/named/etc/
[root@deep]# chattr +i nsswitch.conf
```

具有“+i”属性的文件不能被修改：它不能被删除和改名，不能创建到这个文件的链接，不能向这个文件写入数据。只有超级用户才能设置和清除这个属性。

第三步：

告诉 syslogd 守护进程新的“虚” 根服务。

通常情况下，进程通过“/dev/log”向 syslogd 发送消息。由于虚” 根的限制，这种通信被禁止。因此 syslogd 需要改为监听“/chroot/named/dev/log”。我们可以通过编辑 syslog 的启动脚本来设定新的监听地点。

编辑 **syslog** 脚本（vi +24 /etc/rc.d/init.d/syslog）作如下改动:

```
daemon syslogd -m 0
```

改为：

```
daemon syslogd -m 0 -a /chroot/named/dev/log
```

第四步：

编辑“named”脚本（vi /etc/rc.d/init.d/named） 作如下改动：

```
[ -f /usr/sbin/named ] || exit 0
```

改为：

```
[ -f /chroot/named/usr/sbin/named ] || exit 0
```

```
[ -f /etc/named.conf ] || exit 0
```

改为：

```
[ -f /chroot/named/etc/named.conf ] || exit 0
```

```
daemon named
```

改为：

```
daemon /chroot/named/usr/sbin/named -t /chroot/named/ -unamed -gnamed
```

- “-t”设定 named 在“虚” 根环境下启动。
- “-u”规定运行进程的用户。
- “-g”规定运行进程的组。

Red Hat 的启动脚本的 daemon()函数不允许规定替代的 PID 文件。但这不会影响 named 初始化脚本的启动和停止，因为它们是从“虚” 根以外调用的。

在 8.2 版本的 BIND 中“ndc”命令已经编译成二进制代码，以致在本设置中，发行版的 ndc 没有实际用途。可以重新编译 BIND 源文件进行弥补。

必须明白这些设置仅对 ndc 有效。如果你需要一个新的 named 及 named xfer，应该采用原来的设置。

为了实现这一点，在源文件的顶层目录中。

对于 ndc：

```
[root@deep]# cp bind-src.tar.gz /vat/tmp
[root@deep]# cd /var/tmp/
[root@deep]# tar xzpf bind-src.tar.gz
[root@deep]# cd src
[root@deep]# cp port/linux/Makefile.set port/linux/Makefile.set-orig
```

编辑“Makefile.set”文件（vi port/linux/Makefile.set）并做如下改动：

```
'CC=egcs -D_GNU_SOURCE'
'CDEBUG=-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-pointer -fno-exceptions'
'DESTBIN=/usr/bin'
'DESTSBIN=/chroot/named/usr/sbin'
'DESTEXEC=/chroot/named/usr/sbin'
'DESTMAN=/usr/man'
'DESTHELP=/usr/lib'
'DESTETC=/etc'
'DESTRUN=/chroot/named/var/run'
'DESTLIB=/usr/lib/bind/lib'
'DESTINC=/usr/lib/bind/include'
'LEX=flex -8 -I'
'YACC=yacc -d'
'SYSLIBS=-lfl'
```

```
'INSTALL=install'
'MANDIR=man'
'MANROOFF=cat'
'CATEXT=$$N'
'PS=ps -p'
'AR=ar crus'
'RANLIB=:'
```

更改 Makefile 前后所带来的变化是我们改变了“DESTSBIN=”、“DESTEXEC=”和 “DESTRUN=”三行，使其指向正确的“虚” 根路径。经过修改后，ndc 程序知道到哪里去寻找“named”。

```
[root@deep]# make clean
[root@deep]# make
[root@deep]# cp bin/ndc/ndc /usr/sbin/
[root@deep]# cp: overwrite `/usr/sbin/ndc`? y
[root@deep]# strip /usr/sbin/ndc
```

我们重新编译二进制文件，并把编译“ndc”程序的结果拷贝到“/usr/sbin”目录下覆盖老版本。我们没有忘记为提高性能而优化代码。

重建一下“named”二进制文件是一个不错的主意，这样可以确保“named”和“ndc”具有相同的版本号。

对于 named：

```
[root@deep]# cd /var/tmp/src
[root@deep]# cp port/linux/Makefile.set-orig port/linux/Makefile.set
[root@deep]# cp: overwrite `port/linux/Makefile.set`? y
```

编辑“Makefile.set”文件（vi port/linux/Makefile.set）并做如下改动：

```
'CC=egcs -D_GNU_SOURCE'
'CDEBUG=-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-
frame-pointer -fno-exceptions
'DESTBIN=/usr/bin'
'DESTSBIN=/usr/sbin'
'DESTEXEC=/usr/sbin'
'DESTMAN=/usr/man'
'DESTHELP=/usr/lib'
'DESTETC=/etc'
'DESTRUN=/var/run'
'DESTLIB=/usr/lib/bind/lib'
```

```
'DESTINC=/usr/lib/bind/include'
'LEX=flex -8 -I'
'YACC=yacc -d'
'SYSLIBS=-lfl'
'INSTALL=install'
'MANDIR=man'
'MANROFF=cat'
'CATEXT=$$N'
'PS=ps -p'
'AR=ar crus'
'RANLIB=:'


[root@deep]# rm -f .settings
[root@deep]# make clean
[root@deep]# make
[root@deep]# cp bin/named/named /chroot/named/usr/sbin
[root@deep]# cp: overwrite `/chroot/named/usr/sbin/named? y
[root@deep]# cp bin/named-xfer/named-xfer /chroot/named/usr/sbin
[root@deep]# cp: overwrite `/chroot/named/usr/sbin/named-xfer? y
[root@deep]# strip /chroot/named/usr/sbin/named
[root@deep]# strip /chroot/named/usr/sbin/named-xfer
```

因为编译系统可以缓存变量，我们可以删除“.settings”文件。运行“make clean”命令可以确保没有过时的垃圾残留。编译完“named”，“named-xfer”之后拷贝这两个文件到“虚” 根目录之下，并且不要忘记优化代码以提高程序性能。

删除不必要的目录和文件

```
[root@deep]# rm -f /usr/sbin/named
[root@deep]# rm -f /usr/sbin/named-xfer
[root@deep]# rm -f /etc/named.conf
[root@deep]# rm -rf /var/named/
```

我们从“/usr/sbin”中删除了“named”，“named-xfer”文件，因为我们已经可以让“虚”根目录之下程序拷贝来完成同样的这些日常工作。对于“named.conf”文件和“/var/named”目录，如法炮制。

第五步：

测试以下“虚”根环境下的配置情况！重新启动“syslogd”守护进程：

```
[root@deep]# /etc/rc.d/init.d/syslog restart
```

启动“虚”根环境下的新的 BIND：

```
[root@deep]# /etc/rc.d/init.d/named start
```

核查以下以确信 named 正在采用新的参数运行。

```
[root@deep]# ps auxw | grep named
```

```
named 11446 0.0 1.2 2444 1580 ? S 23:09 0:00 /chroot/named/usr/sbin/named -t
/chroot/named/ -unamed -gnamed
```

第一列应该是“named”，表示运行这个程序的属主。最后一列应该为“named -t /chroot/named/ -unamed -gnamed”。

清理工作

```
[root@deep]# rm -rf /var/tmp/bind/
```

注意：出于安全考虑，在企业内部网和外部主机之间不要架设 DNS。如果仅仅通过 IP 地址将企业内部网同外界联系起来将极大的提高系统的安全性。

区带（Zone）转移

限制区带转移。

限制区带转移可以预防：

- 他人“借用”域名服务器。
- 黑客列出区带的内容。

以下是一个“named.conf”文件之中包括“allow-transfer”选项的例子：

```
options {
allow-transfer { 208.164.186.2; };
};
```

以上限制了哪些二级域名服务器可以从这台域名服务器上获取区带信息。限制区带转移需注意：不仅要在主域名服务器上限制区带转移，在二级域名服务器也要限制区带转移。

允许查询

限制域名服务器所接受的查询可以规定：

- 允许使用本域名服务器的 IP 地址范围。
- 它们可以查询的区带。

以下是一个“named.conf”文件之中包括“allow-query”选项的例子：

```
options {
allow-query { 208.164.186/24; 127.0.0/8; };
};
```

以上规定了能够向本台域名服务器发出请求的 IP 地址范围。特别的，对于运行在 Internet 防火墙之内的用户而言，他们有一种对于外部世界隐藏其名字空间的要求，然而同时他们又要给有限的授权用户提供域名服务。

转发限制

你可能需要在你的转发服务器当机或没有应答的时候不要尝试站点之外的转发器。“forward only”选项可以实现这一功能。

以下是一个 named.conf 文件之中包括“forward only”选项的例子：

```
options {
forwarders { 205.151.222.250; 205.151.222.251; };
forward-only;
};
```

在“forwarder”一行中，“205.151.222.250”和“205.151.222.251” 分别代表你的 ISP 及另外的 DNS 服务器的 IP 地址。

参考文献

为了获取更详细的信息，可以阅读以下手册页：

```
$ man dnsdomainname (1) - show the system's DNS domain name
$ man dnskeygen (1) - generate public, private, and shared secret keys for DNS
Security
$ man dnsquery (1) - query domain name servers using resolver
$ man named (8) - Internet domain name server (DNS)
```

DNS 管理工具

以下这些命令，是我们日常维护工作时经常要用到的。但是还有许许多多的命令我们没有列出，请阅读以下手册页和其他的参考文献，以便能够得到更加详细的信息。

dig

“**db.cache**”文件规定了“根”区带服务器的位置。它必须定期更新。虽然“根”区带服务器不会经常变动，但有时确实它会变。每隔一到两个月检查一下“**db.cache**”文件是一个不错的实践经验。

用以下命令为你的 DNS 服务器查询一下新的“**db.cache**”文件：

```
[root@deep]# dig @.aroot-servers.net . ns > db.cache
```

获取“db.cache”文件后拷贝它到 “**/var/named/**” 目录下。

```
[root@deep]# cp db.cache /var/named/
```

其中**@.aroot-servers.net** 是你用来请求“根”区带信息文件（db.cache）服务器，而“db.cache”文件是你的新“db.cache”文件的名称。

ndc

这个命令允许系统管理员控制域名服务器的运行情况。如果没有命令输入，ndc 将给出提示符，直到遇到文件结束符 EOF 为止。

在你的终端上键入 ndc 及 help 可以看到各种命令。

```
[root@deep]# ndc
[root@deep]# ndc help
```

DNS 用户工具

以下这些命令，是我们日常维护工作时经常要用到的。但是还有许许多多的命令我们没有列出，请阅读以下手册页和其他的参考文献，以便能够得到更加详细的信息。

nslookup

nslookup 是一个用来查询 Inernet 域名服务器的实用程序。nslookup 有两种运行模式：交互式和非交互式运行模式。交互式运行模式允许用户通过域名服务器检索各种主机，并可以打印出主机名称列表。非交互式运行模式用来打印所请求的某个主机或域的特定的信息。

进入“nslookup”交互式运行模式，运行下面的命令：

```
[root@deep]# nslookup (键入 help 获取有关 nslookup 命令的详细信息).
Default Server: deep.openarch.com
Address: 208.164.186.3
```

进入 nslookup 非交互式运行模式，运行下面的命令：

```
[root@deep]# nslookup www.redhat.com
```

非交互式运行模式用于我们已知所要查询的主机 IP 地址或主机名称的时候，它们通常作为“nslookup”命令的第一个参数，第二个参数通常为域名服务器的 IP 地址或主机名称。

dnsquery

dnsquery 程序提供了一个通过调用 BIND 解释库来访问域名服务器的通用界面。应用这个程序的目的是作为“nslookup”程序的补充和替代。

通过解释器访问域名服务器，使用命令：

```
[root@deep]# dnsquery <-n nameserver> <host>
```

例如：

```
[root@deep]# dnsquery -n localhost 192.168.1.2
```

其中<-n nameserver>代表查询中所用的域名服务器，它可以是 IP 地址的形式 w.x.y.z 或域名两种形式（缺省在“/etc/resolv.conf”文件中指定）<host>代表 Internet 上的主机或域的名字。

host

“host”程序用于检索 Internet 上的主机信息。它从遍布在世界各地的互相连接在一起的服务器来获取这些信息。缺省情况下，它仅完成域名和 IP 地址之间的转换。然而，“-t”及“-a”两个选项用于查询存储在域服务器内的主机的所有信息。

通过域服务器查找主机名称，用以下命令：

```
[root@deep]# host <FQDN, domain names, host names, or host numbers>
```

例如：

```
[root@deep]# host deep.openarch.com
```

其中<FQDN, domain names, host names, or host numbers>表示即可以是 FDQN，例如（deep.openarch.com），域名，例如 （openarch.com），主机名称，例如（deep），也可以是 IP 地址，例如（192.168.1.1）。

为查询主机的所有信息，使用命令：

```
[root@deep]# host <-a domain names >
```

例如：

```
[root@deep]# host -a openarch.com
```

其中<domain names>表示域名，例如（openarch.com）。这个选项用于找出域服务器内关于这台主机的所有信息。

为获取全域的列表，使用下列命令：

```
[root@deep]# host <-l domain names >
```

例如：

```
[root@deep]# host -l openarch.com
```

其中<domain names>表示域名，例如（openarch.com）。这个选项用于获取整个域（openarch.com）以主记录格式保存的区带数据，服务器内关于这台主机的所有信息。

注意：“-l”是通过先进行一次完全的区带转移而后过滤出你所需要的信息来实现的。此项命令只有在必要时才可使用。

安装到系统中的文件

```
> /etc/rc.d/init.d/named
> /etc/rc.d/rc0.d/K45named
> /etc/rc.d/rc1.d/K45named
> /etc/rc.d/rc2.d/K45named
> /etc/rc.d/rc3.d/K45named
> /etc/rc.d/rc4.d/K45named
> /etc/rc.d/rc5.d/K45named
> /etc/rc.d/rc6.d/K45named
> /etc/named.conf
> /usr/bin/addr
> /usr/bin/nslookup
> /usr/bin/dig
> /usr/bin/dnsquery
> /usr/bin/host
> /usr/bin/nsupdate
> /usr/bin/mkservdb
> /usr/lib/bind
> /usr/lib/bind/include
> /usr/lib/bind/include/arpa
> /usr/lib/bind/include/arpa/inet.h
> /usr/lib/bind/include/arpa/nameser.h
> /usr/lib/bind/include/arpa/nameser_compat.h
> /usr/lib/bind/include/isc
> /usr/lib/bind/include/isc/eventlib.h
> /usr/lib/bind/include/isc/misc.h
> /usr/lib/bind/include/isc/tree.h
> /usr/lib/bind/include/isc/logging.h
> /usr/lib/bind/include/isc/heap.h
> /usr/lib/bind/include/isc/memcluster.h
> /usr/lib/bind/include/isc/assertions.h
> /usr/lib/bind/include/isc/list.h
> /usr/lib/bind/include/isc/dst.h
```

```
> /usr/lib/bind/include/isc/irpmarshall.h
> /usr/lib/bind/include/netdb.h
> /usr/lib/bind/include/resolv.h
> /usr/lib/bind/include/res_update.h
> /usr/lib/bind/include/irs.h
> /usr/lib/bind/include/irp.h
> /usr/lib/bind/include/hesiod.h
> /usr/lib/bind/include/sys
> /usr/lib/bind/include/net
> /usr/lib/bind/lib
> /usr/lib/bind/lib/libbind.a
> /usr/lib/bind/lib/libbind_r.a
> /usr/lib/nslookup.help
> /usr/man/man1/dig.1
> /usr/man/man1/host.1
> /usr/man/man1/dnsquery.1
> /usr/man/man1/dnskeygen.1
> /usr/man/man3/hesiod.3
> /usr/man/man3/gethostbyname.3
> /usr/man/man3/inet_cidr.3
> /usr/man/man3/resolver.3
> /usr/man/man3/getnetent.3
> /usr/man/man3/tsig.3
> /usr/man/man3/getaddrinfo.3
> /usr/man/man3/getipnodebyname.3
> /usr/man/man5/resolver.5
> /usr/man/man5/irs.conf.5
> /usr/man/man5/named.conf.5
> /usr/man/man7/hostname.7
> /usr/man/man7/mailaddr.7
> /usr/man/man8/named.8
> /usr/man/man8/ndc.8
> /usr/man/man8/named-xfer.8
> /usr/man/man8/named-bootconf.8
> /usr/man/man8/nslookup.8
> /usr/man/man8/nsupdate.8
> /usr/sbin/ndc
> /usr/sbin/named
> /usr/sbin/named-xfer
> /usr/sbin/irpd
> /usr/sbin/dnskeygen
> /usr/sbin/named-bootconf
> /var/named
```

Linux Sendmail 服务器

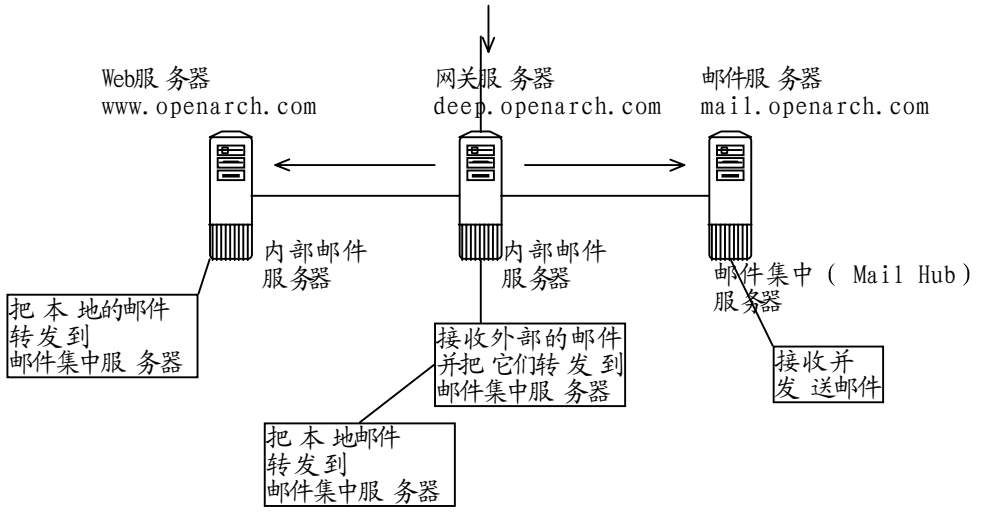
概述

Sendmail 是一种被广泛采用的邮件传输代理程序（Mail Transport Agent，MTA），邮件传输代理程序负责把邮件从一台机器发送到另外一台机器。Sendmail 不是一种用于阅读邮件的客户程序。Sendmail 是一种运行在后台的用于把你的邮件通过网络发送到 Internet 上，最终转发到目的地的程序。Sendmail 的安全漏洞很多，是一个知名的黑客入侵的切入点。尽管，sendmail 的第八版本的先进特性已使攻击它的难度大大增加了。

在我们介绍的安装和配置过程里，我们将提供给你两种不同的 Sendmail 安装配置环境。一种是中央邮件转储中心，另一种是适用于本地或邻居的客户机及服务器。

中央邮件转储中心的配置用于设置一台专用邮件服务器。它被用来为网络内的所有主机，包括本地机、邻居的客户机及服务器发送、接收、转储邮件。本地机、邻居的客户机及服务器是指所有运行 sendmail 并且把发往外部的邮件统统发送到中央邮件转储中心，并由它在将来进行转发。另外，内部的客户机及服务器从来不直接从 Internet 上直接接收邮件，取而代之的是由中央邮件转储中心负责接收和保存邮件。在你的网络之中设置一个中央邮件转储中心是一个很好的主意；这种体系结构可以简化内部客户机及服务器的管理和维护工作。并且可以提高你的站点的安全性。

你可以配置你的邻居上的 sendmail，使它只能接收来自本地的邮件，因此可以将邻居隔离开来，从而获得更好的安全性。网关服务器（位于防火墙之外或之上）可以起到代理服务器的功能。它负责接收来自外部的并发向内部的邮件（通过防火墙文件）并转发给中央邮件转储中心。请注意，网关服务器被配置成象邻居上的 sendmail 一样，它从来不接收来自外部的邮件。（Internet）



上面图示的是本书假定的 Sendmail 配置情况。我们在不同的服务器上作不同的设置（中央邮件转储中心，本地或邻居的客户机及服务器）。根据你自己的网络体系结构的不同情况将有各种各样的可能性存在。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

sendmail 的版本是 8.9.3。

软件包的来源

sendmail 的主页：http://www.sendmail.org/。

必须确保下载：sendmail.8.9.3.tar.gz。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >send1”，在编译和安装完软件之后运行命令“find /* > send2”，最后用命令“diff send1 send2 > send”找出变化。

编译

把软件包（tar.Z）解压缩：

```
[root@deep]# cp sendmail.version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf sendmail.version.tar.gz
```

编译配置

cd 进入新的 Sendmail 目录，而后在终端上键入如下命令：

编辑“**linux.m4**”文件（vi +16 cf/ostype/linux.m4）并做如下改动：

```
define(`LOCAL_MAILER_PATH', /bin/mail.local)dnl
```

改为：

```
define(`PROCMAIL_MAILER_PATH', `/usr/bin/procmail')dnl
dnl define(`LOCAL_MAILER_FLAGS', `ShPfn')dnl
dnl define(`LOCAL_MAILER_ARGS', `procmail -a $h -d $u')dnl
define(`STATUS_FILE', `/var/log/sendmail.st')dnl
```

注意：这些步骤仅对中央邮件转储中心是必要的，它使 sendmail 采用 procmail 作为邮件发送程序，并告诉 sendmail 它的状态文件“sendmail.st”位于“/var/log”目录。

编辑“**linux.m4**”文件（vi +16 cf/ostype/linux.m4）并添加下行：

```
define(`STATUS_FILE', `/var/log/sendmail.st')dnl
```

注意：这些步骤仅对本地或邻居的客户机及服务器是必要的，它告诉 sendmail 它的状态文件“sendmail.st”位于“/var/log”目录。

编辑“**header.m4**”文件（vi +26 BuildTools/M4/header.m4）并做如下改动：

```
define(`confLIBSEARCH', `db bind resolv 44bsd')
```

改为：

```
define(`confLIBSEARCH', `db1 bind resolv 44bsd')
```

编辑“**makemap.c**”文件（vi +30 makemap/makemap.c）并做如下改动：

```
# include <db.h>
```

改为：

```
# include <db_185.h>
```

编辑“**map.c**”文件（vi +29 src/map.c）并做如下改动：

```
# include <db.h>
```

改为：

```
# include <db_185.h>
```

编辑“**udb.c**”文件（vi +28 src/udb.c）并做如下改动：

```
# include <db.h>
```

改为：

```
# include <db_185.h>
```

编辑 praliases.c 文件（vi +37 praliases/praliases.c）并做如下改动：

```
# include <db.h>
```

改为：

```
# include <db_185.h>
```

以上的四处变动，指定了安装在 Linux 上的 Berkeley DB 包的版本号。

编辑 **Linux** 文件（vi BuildTools/OS/Linux）并删去下列行：

删去下列行：

```
define(`confSTDIR', `/etc')
define(`confHFDIR', `/usr/lib')
define(`confDEPEND_TYPE', `CC-M')
define(`confMANROOT', `/usr/man/man')
```

编辑 **Linux** 文件（vi BuildTools/OS/Linux）并增加下列行：

增加下列行：

```
define(`confSTDIR', `/var/log')
define(`confHFDIR', `/usr/lib')
define(`confDEPEND_TYPE', `CC-M')
define(`confMANROOT', `/usr/man/man')
define(`confSBINGRP', `root')
define(`confSBINMODE', `6755')
define(`confEBINDIR', `/usr/sbin')
```

这些宏定义了一些变量，如：log 文件，lib,，man 目录的位置及位于“sbin”目录下的 sendmail 二进制程序的组名称和权限。

编辑“daemon.c”文件（vi +1452 src/daemon.c）并做如下改动：

```
nleft = sizeof ibuf - 1;
```

改为：

```
nleft = sizeof(ibuf) - 1;
```

编辑“smrsh.c”文件（vi +61 smrsh/smrsh.c）并做如下改动：

```
# define CMDDIR "/usr/adm/sm.bin"
```

改为：

```
# define CMDDIR "/etc/smrsh"
```

以上改动指定了所有命令的所在的目录。

编辑“smrsh.c”文件（vi +69 smrsh/smrsh.c）并做如下改动：

```
# define PATH "/bin:/usr/bin:/usr/ucb"
```

改为：

```
# define PATH "/bin:/usr/bin"
```

以上改动指定了运行“smrsh”程序是的缺省搜索命令的路径。

编译和优化

Sendmail 的建立脚本允许你通过 “–f” 标志指定一个定点配置文件（Build -f ../BuildTools/Site/siteconfig.m4）。定点配置文件包括系统安装的定义。我们要建立一个适合我们的安装系统的定点配置文件，并由于缺省情况下建立脚本会从定点配置文件的路径下搜索它，所以我们把它放在缺省目录 Sendmail 发行的源文件的子目录“BuildTools/Site”之下。

cd 进入新的 Sendmail 目录而后创建“siteconfig.m4”文件（touch BuildTools/Site/siteconfig.m4）并在文件中加入以下几行：

```
define(`confMAPDEF', `-DNEWDB') (Require only for Mail Hub configuration)
define(`confENVDEF', `-DPICKY_QF_NAME_CHECK -DXDEBUG=0')
define(`confCC', `egcs')
define(`confOPTIMIZE', `-O9 -funroll-loops -ffast-math -malign-double
-mcpu=pent
iumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions')
define(`confLIBS', `-lnsl')
define(`confLDOPTS', `-s')
define(`confMANOWN', `root')
define(`confMANGRP', `root')
define(`confMANMODE', `644')
define(`confMAN1SRC', `1')
define(`confMAN5SRC', `5')
define(`confMAN8SRC', `8')
```

以上告诉“siteconfig.m4”对于本安装把自己设置为：

define(`confMAPDEF', `-DNEWDB')

本宏选项指定 Sendmail 别名文件和一般映射所用的数据库类型。在我们的配置里我们采用 Berkeley db(3)既有 hash 结构又有 btree 结构。“define(`confMAPDEF', `-DNEWDB')”仅对中央邮件转储中心是必须的。对于本地或邻居的客户机及服务器这没有必要。因为本地或邻居的客户机及服务器使用本地邮件并不需要使用别名数据库，因此也不需要“-DNEWDB”功能。

define(`confENVDEF', `-DPICKY_QF_NAME_CHECK -DXDEBUG=0')

本宏选项主要用来指定那些代码可以被包含进来，那些要被排除出去。定义“-DPICKY_QF_NAME_CHECK”sendmail 将能够把格式错误的“qf”文件记录到日志文件里去，并把“qf”文件改为“Qf”文件。“-DXDEBUG=0”参数禁止编译时的附加检查。

define(`confCC', `egcs')

本宏选项主要用来指定编译 Sendmail 所用的 C 编译器。本次我采用“egcs”编译器来优化配置。

define(`confOPTIMIZE', `-O9 -funroll-loops -ffast-math -malign-double -mcpu=pent

iumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions')

本宏选项指定传递给 CC 编译器使其能够根据我们的 CPU 结构进行优化。

define(`confLIBS', `-lnsl')

本宏选项指定了传递给 ld 的参数-l。

define(`confLDOPTS', `-s')

本宏选项指定了传递给 ld 的连接器选项参数。

define(`confMANOWN', `root')

本宏选项指定了安装的手册页的所有者。

define(`confMANGRP', `root')

本宏选项指定了安装的手册页的所有组。

define(`confMANMODE', `644')

本宏选项指定了安装的手册页的权限。

define(`confMAN1SRC', `1')

本宏选项指定了安装的手册页 confMAN1 的源文件。

define(`confMAN5SRC', `5')

本宏选项指定了安装的手册页 confMAN5 的源文件。

define(`confMAN8SRC', `8')

本宏选项指定了安装的手册页 confMAN8 的源文件。

```
[root@deep]# cd /var/tmp/sendmail-version
[root@deep]# cd src
[root@deep]# sh Build -f ../BuildTools/Site/siteconfig.m4
[root@deep]# cd ..
[root@deep]# cd mailstats
[root@deep]# sh Build -f ../BuildTools/Site/siteconfig.m4
[root@deep]# cd ..
[root@deep]# cd makemap (仅中央邮件转储中心需要)
[root@deep]# sh Build -f ../BuildTools/Site/siteconfig.m4 (仅中央邮件转储中心需要)
[root@deep]# cd ..
[root@deep]# cd praliases (仅中央邮件转储中心需要)
[root@deep]# sh Build -f ../BuildTools/Site/siteconfig.m4 (仅中央邮件转储中心需要)
[root@deep]# cd ..
[root@deep]# cd smrsh
[root@deep]# sh Build -f ../BuildTools/Site/siteconfig.m4
[root@deep]# cd ..
```

注意：“sh Build”这个 sendmail 脚本会在每一个你需要安装程序的子目录下创建一个新目录，它的名字类似于“obj.yourOS.youOSkernelversion.yourCPUarchitecture”，例如“obj.Linux.2.2.13.i686”。而后在这些目录里创建到必要的源文件的链接和 Makefile 文件。

```
[root@deep]# make install -C src/obj.Linux.version.architecture
[root@deep]# make install -C mailstats/obj.Linux.version.architecture
[root@deep]# make install -C makemap/obj.Linux.version.architecture (仅为中央邮件转储中心)
[root@deep]# make install -C praliases/obj.Linux.version.architecture (仅为中央邮件转储中心)
[root@deep]# make install -C smrsh/obj.Linux.version.architecture
[root@deep]# ln -fs /usr/sbin/sendmail /usr/lib/sendmail
[root@deep]# strip /usr/sbin/mailstats
[root@deep]# strip /usr/sbin/makemap (仅为中央邮件转储中心)
[root@deep]# strip /usr/sbin/praliases (仅为中央邮件转储中心)
[root@deep]# strip /usr/sbin/smrsh
[root@deep]# strip /usr/sbin/sendmail
[root@deep]# chown 0.0 /usr/sbin/mailstats
[root@deep]# chown 0.0 /usr/sbin/makemap (仅为中央邮件转储中心)
[root@deep]# chown 0.0 /usr/sbin/praliases (仅为中央邮件转储中心)
[root@deep]# chown 0.0 /usr/sbin/smrsh
[root@deep]# chmod 511 /usr/sbin/smrsh
[root@deep]# install -d -m755 /var/spool/mqueue
[root@deep]# chown root.mail /var/spool/mqueue
[root@deep]# mkdir /etc/smrsh
[root@deep]# mkdir /etc/mail (仅为中央邮件转储中心)
```

“**sh Build –f**”命令在安装 sendmail 系统之前建立“obj.Linux.version.architecture”中用于安装 sendmail 的必须文件的依存关系。

“**make install –C**”命令在安装 sendmail，mailstats，makemap，praliases，smrsh 二进制文件和连接及其相关的手册页。

“**ls -fs**”命令用于在“/usr/lib”目录中创建 sendmail 二进制文件的连接。这个步骤是必要的，因为有些程序期望在这个目录（/usr/lib)中找到 sendmail 二进制文件。

“**strip**”命令可以减小 mailstats，praliases，sendmail，smrsh 和 makemap 等二进制文件 的长度，从而提高系统性能。

“**install**”命令将在“/var/spool”目录下创建一个权限为 755 的新目录“mqueue”。邮件可能因为各种原因暂时不能发送，为确保那些消息最终能够被正确地投递 sendmail 把它们保存一个目录队列里直到它们被成功地发送出去。

“**chown**”命令将文件 mailstats，makemap，praliases，smrsh 的 UID 和 GID 设置成“root”，将 mqueue 目录的 UID 设成“root”GID 设成“mail”。

“**mkdir**”命令在你的系统中创建“/etc/mail”和“/etc/smrsh”两个目录。

注意：对于中央邮件转储中心来说，程序“makemap”和“ praliases”是必须的。“makemap”允许为 sendmail 创建数据库映射如“/etc/aliases”和“/etc/mail/access”这些文件。“praliases”用于显示系统的邮件别名文件（“/etc/aliases”文件的内容）。既然我们最好只有一台机器——中央邮件转储中心用于管理网络内的邮件数据库文件，在其它的主机上使用“makemap”和“praliases”及 db 文件就没有必要了。

配置

在本书“Linuxsos.pdf”中所描述的所有软件都将有自己特定的目录及子目录，一个用 tar 压缩的文件“floppy.tgz”包括它们在特定目录中的特定的配置文件。如果你有这个文件，你就不必重新手工创建下面这些配置文件或者从本文中剪帖它们到你的配置文件里了。如果你打算使用那个压缩文件中的配置文件的拷贝，你必须根据你的需要修正和调整一下，把有关 Sendmail 软件放到你服务器上的一个合适的位置，如下所示。服务器配置文件的地址是：
http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

为了运行中央邮件转储中心服务，以下文件必须存在，而且它们被创建在或拷贝到合适的目录之中。

- 拷贝“access”文件到“/etc/mail/”目录
- 拷贝“aliases”脚本文件到 “/etc/”目录
- 拷贝“sendmail.cw”文件到 “/etc/”目录
- 拷贝“sendmail.mc”文件到 “/etc/”目录
- 拷贝“sendmail”文件到 “/etc/sysconfig”目录
- 拷贝“sendmail”脚本文件到 “/etc/rc.d/init.d/”目录

为了运行本地或邻居的客户机及服务器，以下文件必须存在，而且它们被创建在或拷贝到合适的目录之中：

- 拷贝“null.mc” 文件到“/etc/”目录
- 拷贝“sendmail”文件到 “/etc/sysconfig”目录
- 拷贝“sendmail”脚本文件到 “/etc/rc.d/init.d/”目录

可以从“floppy.tgz”文档中获取以上这些文件，从解压的“floppy.tgz”文档中拷贝这些文件到适当的位置，或者从本书中拷贝并粘贴到相应的文件中。

为中央邮件转储中心配置 /etc/mail/access 和 access.db 文件

创建的“access”数据库可以用来根据邮件的来源决定是否接收或拒绝它们。例如，你可以选择拒绝已知的所有欺骗邮件。如果你决定采用中央邮件转储中心来管理邮件系统的话，对于一般机器而言，“access”和“access.db”两个文件不是必须的。应该清楚的认识到采用中央邮件转储中心可以提高系统的安全性，并可以简化对网络上的其它运行 sendmail 的主机的管理。

第一步

创建“**access**”文件（touch /etc/mail/access）并加入以下几行：

```
# Description showing bellow for the format of this file comes from
# the Sendmail source distribution under "cf/README" file.
#
# The table itself uses e-mail addresses, domain names, and network
# numbers as keys. For example,
#
#   spammer@aol.com         REJECT
#   cyberspammer.com        REJECT
#   192.168.212             REJECT
#
# would refuse mail from spammer@aol.com, any user from cyberspammer.com
# (or any host within the cyberspammer.com domain), and any host on the
# 192.168.212.* network.
#
# The value part of the map can contain:
#
#   OK      Accept mail even if other rules in the
#           running ruleset would reject it, for example,
#           if the domain name is unresolvable.
#   RELAY   Accept mail addressed to the indicated domain or
#           received from the indicated domain for relaying
#           through your SMTP server. RELAY also serves as
#           an implicit OK for the other checks.
#   REJECT  Reject the sender or recipient with a general
```

```
#           purpose message.
#   DISCARD Discard the message completely using the
#           $#discard mailer. This only works for sender
#           addresses (i.e., it indicates that you should
#           discard anything received from the indicated
#           domain).
#   ### any text where ### is an RFC 821 compliant error code
#           and "any text" is a message to return for
#           the command.
#
# For example:
#
#   cyberspammer.com       550     We don't accept mail from spammers
#   okay.cyberspammer.com      OK
#   sendmail.org               OK
#   128.32                 RELAY
#
# would accept mail from okay.cyberspammer.com, but would reject mail
# from all other hosts at cyberspammer.com with the indicated message.
# It would allow accept mail from any hosts in the sendmail.org domain,
# and allow relaying for the 128.32.*.* network.
#
# You can also use the access database to block sender addresses based on
# the username portion of the address. For example:
#
#   FREE.STEALTH.MAILER@ 550 Spam not accepted
#
# Note that you must include the @ after the username to signify that
# this database entry is for checking only the username portion of the
# sender address.
#
# If you use like we do in our "sendmail.mc macro configuration:
#
#   FEATURE(`blacklist_recipients')
#
# then you can add entries to the map for local users, hosts in your
# domains, or addresses in your domain which should not receive mail:
#
#   badlocaluser 550 Mailbox disabled for this username
#   host.mydomain.com 550 That host does not accept mail
#   user@otherhost.mydomain.com 550 Mailbox disabled for this recipient
#
# This would prevent a recipient of badlocaluser@mydomain.com, any
# user at host.mydomain.com, and the single address
```

```
# user@otherhost.mydomain.com from receiving mail. Enabling this
# feature will keep you from sending mails to all addresses that
# have an error message or REJECT as value part in the access map.
# Taking the example from above:
#
#   spammer@aol.com         REJECT
#   cyberspammer.com        REJECT
#
# Mail can't be sent to spammer@aol.com or anyone at cyberspammer.com.
#
# Now our configuration of access file,
# by default we allow relaying from localhost...
localhost.localdomain RELAY
localhost RELAY
127.0.0.1 RELAY
```

第二步

创建“access.db”文件：

由于“/etc/mail/access”是一个数据库，所以在创建完以上的文本文件以后需要运行“makemap”程序来创建数据库映射。

用以下命令创建“access database map”数据库映射。

```
[root@deep]# makemap hash /etc/mail/access.db < /etc/mail/access
```

为中央邮件转储中心配置 /etc/aliases 和 aliases.db 文件

处理别名是一种把邮件接收者的名称转换成另外一种名称的方法。一种用途是把派生的名称（例如 root）转成一个实际的用户名，另外一种用途是把一个名称翻译成一堆名字的列表（邮件列表）。对于每一个列在信封上的J本地邮件接收者，sendmail 都要从“aliases”文件中查找接收者名字。由于 sendmail 要从“aliases”文件中成千上万的名字中检索，将别名存为一个单独的数据库格式的文件“db”可以大大提高查询速度。如果你决定采用中央邮件转储中心来管理邮件系统的话，对于本地或邻居的客户机及服务器而言，“aliases”和“aliases.db”两个文件不是必须的。

第一步

创建“aliases”文件（touch /etc/aliases）并加入以下几行：

```
#
#   @(#)aliases     8.2 (Berkeley) 3/5/94
```

```
#
# Aliases in this file will NOT be expanded in the header from
# Mail, but WILL be visible over networks or from /bin/mail.
#
#   >>>>>>>>>> The program "newaliases" must be run after
#   >> NOTE >> this file is updated for any changes to
#   >>>>>>>>>> show through to sendmail.
#
# Basic system aliases -- these MUST be present.
MAILER-DAEMON: postmaster
postmaster: root
# General redirections for pseudo accounts.
bin: root
daemon: root
nobody: root
# Person who should get root's mail
#root: admin
```

注意：你的别名文件可能要复杂得多，尽管如此，上面的例子显示了别名文件的一种最小模式。

第二步，创建“aliases.db”文件：

由于“/etc/aliases”是一个数据库，所以在创建完以上的文本文件以后需要运行“makemap”程序来创建数据库映射。

用以下命令创建“aliases database map”数据库映射。

```
[root@deep]# makemap hash /etc/aliases.db < /etc/aliases
```

为中央邮件转储中心配置 /etc/mail/virtusertable

domaintable mailertable 和 virtusertable.db

domaintable.db mailertable.db 文件

有些站点在由老域名转换成新域名的过渡时期需要多个域名共存。域表的属性通过改写旧域到新域的办法来实现平滑过渡。

虚拟用户表用来创建虚拟域名到新地址的映射。它是一种特定于域的别名文件，允许在一台机器上保留多个虚拟域名。

邮件代理表是一种数据库，它完成从“host.domain”名称到投递代理和新域名对的映射。它可以覆盖某些域的路由。

用以下命令在“/etc/mail”目录下创建“virtusertable”、“domaintable”、“mailertable”及相应的“.db”文件：

```
[root@deep]# for map in virtusertable domaintable mailertable


> do
> touch /etc/mail/${map}
> chmod 0644 /etc/mail/${map}
> makemap hash /etc/mail/${map}.db < /etc/mail/${map}
> chmod 0644 /etc/mail/${map}.db
> done
```

为中央邮件转储中心配置 /etc/sendmail.cw 文件

“/etc/sendmail.cw”文件用来读取本机的别名。它的一种用途是声明一个主机列表使本机作为它们的 MX 接收者。也应清楚，“sendmail.cw”文件仅对负责接收、发送和转发邮件的服务器例如中央邮件转储中心是必须的。对一般机器而言我们仅需在邮件转储中心的“/etc/sendmail.cw”文件中添上机器名称即可。以下是个例子。

创建“sendmail.cw”文件（touch /etc/sendmail.cw）并加入以下几行：

```
# sendmail.cw - include all aliases for your machine here.
openarch.com
deep.openarch.com
www.openarch.com
win.openarch.com
mail.openarch.com
```

这种类型配置使所有发出的邮件都好象是从“openarch.com”发送出来的。而所有发送到“www.openarch.com”或其它主机上的邮件都被投递到“mail.openarch.com”——我们的邮件转储中心上。

有一点必须清楚，如果你使用了域名伪装，所有从你的系统中发送到你的系统的邮件会被发送到伪装你的域名的机器上去。例如，对于上面的情况由 cron 守护进程周期的发送到“root@www.openarch.com”的邮件会被发送到“root@mail.openarch.com”。

为中央邮件转储中心配置 /etc/sendmail.mc 文件

中央邮件转储中心的工作非常繁忙，负责许许多多的客户机的邮件收发，而且由于它的工作角色很广，它的配置文件就很复杂。第 8 版本的 Sendmail 采用 m4 宏预编译器程序来创建 Sendmail 的配置文件。m4 宏预编译器程序通过处理结尾为“.mc”文件来创建“/etc/sendmail.cf”配置文件。我们来创建这个文件（sendmail.mc）并将必要的宏的值写入这个文件。以便让 m4 程序能够处理这些输入，收集宏的定义，替换宏为它们的变量值，并用输出的结果来创建“sendmail.cf”文件。为获取详细信息，请参考第 8 版本的 Sendmail 下的源文件发行中的子目录中的文档和 README 文件。

“sendmail.cf”配置文件是运行 sendmail 时由 sendmail 读取的第一个文件。这个文件包括所有其它文件的位置，这些文件的缺省权限和 sendmail 运行所需要的目录。它包含有修改 sendmail 工作方式的选项。

第一步

创建“sendmail.mc”文件（touch /etc/sendmail.mc）并加入以下几行：

```
divert(-1)
dnl This is the macro config file used to generate the /etc/sendmail.cf
dnl file. If you modify this file you will have to regenerate the
dnl /etc/sendmail.cf by running this macro config through the m4
dnl preprocessor:
dnl
dnl     cp sendmail.8.9.3.tar.gz /var/tmp
dnl     cd /var/tmp
dnl     tar xzpf sendmail.8.9.3.tar.gz
dnl     cd /var/tmp/sendmail-8.9.3/cf/cf
dnl     m4 ../m4/cf.m4 /etc/sendmail.mc > /etc/sendmail.cf
dnl
dnl You will need to have the sendmail source distribution for this to
dnl work.
divert(0)
define(`confDEF_USER_ID',``8:12'')
OSTYPE(`linux')
define(`confAUTO_REBUILD')
define(`confTO_CONNECT', `1m')
define(`confTRY_NULL_MX_LIST',true)
define(`confDONT_PROBE_INTERFACES',true)
define(`PROCMAIL_MAILER_PATH',`/usr/bin/procmail')
FEATURE(`smrsh',`/usr/sbin/smrsh')
FEATURE(mailertable)
```

```
FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable')
FEATURE(redirect)
FEATURE(always_add_domain)
FEATURE(use_cw_file)
FEATURE(local_procmail)
FEATURE(nouucp)
MAILER(procmail)
MAILER(smtp)
FEATURE(`access_db')
FEATURE(`blacklist_recipients')
FEATURE(`rbl')
```

这些告诉“sendmail.mc”文件，对于本机的特定配置将其设置成：

divert(-1) and divert(0)

divert(-1)删除结果文件中的垃圾，divert(0)保存通常的配置文件。

define(`confDEF_USER_ID',``8:12'')

此配置选项指定缺省的用户 id，当前为用户“mail”。（参见“/etc/passwd”文件）。

OSTYPE(`linux’)

m4 命令通过提供 OSTYPE 参数来支持不同的操作系统。所有的“mc”文件必须通过这个命令来声明操作系统。这是其中一个“mc”文件所必须的最小信息。

define(`confAUTO_REBUILD')

此配置选项指定如果需要就自动重建别名文件。

define(`confTO_CONNECT', `1m')

此配置选项指定初始化一个连接的超时时间限制。这仅用来缩短超时时间限制；操作系统内核会默认的强制指定一个最大时间限制（根据操作系统的变化而变化）。

define(`confTRY_NULL_MX_LIST',true)

此配置选项表示如果这是一台主机的最好的 MX，并且没有其它的安排，可以尝试直接连接那台主机。

define(`confDONT_PROBE_INTERFACES',true)

如果此配置选项被设定，sendmail 将不把任何本地界面的名称和地址写入$=w class（已知的“equivalent”地址）。

define(`PROCMAIL_MAILER_PATH',`/usr/bin/procmail')

此配置选项设置 procmail 程序的路径。它也被用于 FEATURE(`local_procmail')。

FEATURE(`smrsh',`/usr/sbin/smrsh')

m4 宏可以使用“smrsh”（ sendmail 受限 shell (sendmail restricted shell)）。尽管 Sendmail 通过使用“/etc/aliases”和“/etc/aliases”两个文件已经尽可能的安全了，它对于来自内部的攻击还是很脆弱。为了限制 Sendmail 可以运行的程序，第 8 版本的 Sendmail 包含了“smrsh”（ sendmail 受限 shell）程序。这提高了本地管理员来自 e-mail 的程序的控制。“smrsh”程序的缺省路径是“/usr/local/etc/smrsh”，由于我们更改了“smrsh”的缺省安装路径，我们需要给 smrsh 添加一个参数“/usr/sbin/smrsh”。CERT 推荐安装使用“smrsh”，因此我们鼓励你尽可能的使用这个选项。

FEATURE(mailertable)

m4 宏可以使用“数据库选择新的邮件投递代理” 邮件代理表是一种数据库，它完成从“host.domain”名称到投递代理和新域名对的映射。新的域名用于路由，但这并不影响消息的头部。

FEATURE(`virtusertable',`hash -o /etc/mail/virtusertable')

m4 宏可以使用“虚拟域名支持”， 虚拟用户表用来创建虚拟域名到新地址的映射。

FEATURE(redirect)

m4 宏可以使用“添加地址重定向支持（add support for address.REDIRECT）”重定向特性可以允许设置为“退役”的帐户设置别名。这些别名跳转到指定的转发的地址去。记住，消息是被“弹”过去的，而不是转发的。不会向收件人的新地址发送通知的。

FEATURE(always_add_domain)

m4 宏可以使用“在本地邮件中添加本地域（“add the local domain even on local mail”）的特性。总是添加域名的特性是安全的，我们推荐采用之。它确保本地投递的邮件完全合法。

FEATURE(use_cw_file)

m4 宏可以使用“/etc/sendmail.cw”文件查找本地机器。use_cw_file 的特性使机器通过读取“/etc/sendmail.cw”文件来获得本地主机的“别名”。这个特性的一个用途就是在本地机上声明一个主机列表，本地机就作为一个 MX 接收者。

FEATURE(local_procmail)

m4 宏可以使用“procmail”作为本地邮件投递代理。procmail 程序可以自动的处理用户的邮件（例如，根据主题保存来信到不同的目录中去），而且它可以行使 Sendmail 投递代理的职能。

FEATURE(nouucp)

m4 宏可以消除所有“UUCP”支持。如果你的站点不处理任何 UUCP 的地址，你可以是 nouucp 特性生效。所有有关 UUCP 的宏都被忽略。

MAILER(procmail) and MAILER(smtp)

投递邮件代理不是自动声明的。事实上，你必须通过 MAILER m4 宏来指定那些要支持，那些可以被忽略。MAILER（procmail）和 MAILER（smtp）使 sendmail 支持 procmail，smtp，esmtp，smtp8，并且包含了转储投递功能。

FEATURE(`access_db')

打开访问（access）数据库功能。访问（access）数据库使管理者可以具有控制那些邮件可以接收，那些邮件要被拒绝。例如，你可以选择拒绝所有来自已知的欺骗者的邮件。

FEATURE(`blacklist_recipients')

打开这项功能可以挡住某些用户，主机或地址的收信人的邮件。例如，你可以挡住所有发到用户“nobody”，主机“foo.mydomain.com”和地址“guest@bar.mydomain.com”的邮件。

FEATURE(`rbl')

这将使 sendmail 拒绝来自实时黑洞列表数据库中的所有站点的所有邮件。它的参数是连接的域名服务器。缺省情况下采用“rbl.maps.vix.com”。这里保存有欺骗者的 DNS 数据库。详细信息请参见“http://maps.vix.com/rbl/”。

注意：有时，有的域可能因为在 RPL 列表中而被你所屏蔽而你又希望继续与他们联系。可能对你而言与位于 blacklist 的某个用户联系是非常重要的。在这种情况下 sendmail 允许覆盖这些域并允许接收他们的邮件。只要编辑一下文件将适当的域信息写入即可。例如：

```
blacklisted.domain     OK
```

第二步

现在我们已经创建好了宏配置文件“sendmail.mc”，我们可以用下面的命令创建 sendmail 的配置文件：

```
[root@deep]# cd /var/tmp/sendmail-version/cf/cf/
[root@deep]# m4 ../m4/cf.m4 /etc/sendmail.mc > /etc/sendmail.cf
```

注意：这里“../m4/cf.m4”告诉 m4 程序到哪里去寻找他们的缺省配置文件信息。

为本地或邻居的客户机及服务器配置 /etc/null.mc 文件

与其网络内每一台服务器和工作站各自处理自己的邮件不如设立一个强大的中央服务器处理所有的邮件。这样的服务器叫邮件中心。设置邮件中心的好处有：

- 所有的来信都发送到邮件中心而不是直接发送到客户机。
- 所有从客户机发出的邮件都发送到邮件中心，而后邮件中心把邮件转发到最终的目的地。
- 所有从客户机发出的邮件都好象是从一台服务器上发送出去的，外界没有必要知道客户机的名称信息。
- 客户机不需要运行 sendmail 守护进程来监听邮件。

第一步

客户机从来不需要直接接收邮件，而且需要通过邮件中心服务器负责发送、转储邮件，因此我们只需要创建一个叫做“null.mc”的文件，并经 m4 处理之后为本地或邻居的客户机及服务器生成一个适合于当前安装的定制配置文件。

创建“**null.mc**”文件（touch /etc/null.mc），并加入以下几行：

```
divert(-1)
dnl This is the macro config file used to generate the /etc/sendmail.cf
dnl file. If you modify this file you will have to regenerate the
dnl /etc/sendmail.cf by running this macro config through the m4
dnl preprocessor:
dnl
dnl     cp sendmail.8.9.3.tar.gz /var/tmp
dnl     cd /var/tmp
dnl     tar xzpf sendmail.8.9.3.tar.gz
dnl     cd /var/tmp/sendmail-8.9.3/cf/cf
dnl     m4 ../m4/cf.m4 /etc/null.mc > /etc/sendmail.cf
dnl
dnl You will need to have the sendmail source distribution for this to
dnl work.
divert(0)
OSTYPE(`linux')
FEATURE(`nullclient',`mail.openarch.com')
undefine(`ALIAS_FILE')
```

以上告诉文件，对于本特定配置设置成：

divert(-1) and divert(0)

divert(-1)删除结果文件中的垃圾，divert(0)保存通常的配置文件。

OSTYPE(`linux')

m4 命令通过提供 OSTYPE 参数来支持不同的操作系统。所有的“mc”文件必须通过这个命令来声明操作系统。这是其中一个“mc”文件所必须的最小信息。

FEATURE(`nullclient',`mail.openarch.com')

用来设置成客户机或服务器从来不需要直接接收邮件，而且需要通过邮件中心服务器负责发送、转储邮件。这种情况比较特殊——只需创建一个在基于 SMTP 局域网中能够把邮件转发到邮件转储中心的精简的配置文件即可。它的参数为邮件中心的名字。参数“**mail.openarch.com**”表示一个邮件中心名称规范。当然，对于你而言，应当把这个名称规范改成你自己邮件中心服务器的名称。例如：FEATURE(`nullclient',` **my.mailhub.com**')。

undefine(`ALIAS_FILE')

本设置选项用来防止空客户版本的 sendmail 试图访问“/etc/aliases”和“/etc/aliases.db”

两个文件。我们通过加入这一行，就不必在你的所有内部客户机上拥有“aliases”文件。“aliases”文件仅在负责管理你的整个网络邮件的邮件中心服务器才是必须的。

现在我们已经为所有的本地或邻居的客户机及服务器创建好了宏配置文件“sendmail.mc”，我们可以用下面的命令创建 sendmail 的配置文件：

```
[root@client]# cd /var/tmp/sendmail-version/cf/cf/
[root@client]# m4 ../m4/cf.m4 /etc/null.mc > /etc/sendmail.cf
```

第二步

现在所有的邮件不会直接发送到你的客户机上了。既然没有了接收邮件的连接，你也就没有必要在你的本地或邻居的客户机及服务器上运行 sendmail 的守护进程了。

为了在本地或邻居的客户机及服务器停止运行 sendmail 的守护进程，编辑“/etc/sysconfig/sendmail”文件，并把下一行作以下改动：

```
DAEMON=yes
```

改为：

```
DAEMON=no
```

注意：“/etc/sysconfig/sendmail”文件中“QUEUE=1h”使 sendmail 每隔 1 小时处理一次邮件队列。我们之所以保留这一行是因为当邮件中心宕机的时候 sendmail 仍需周期的处理邮件队列。

第三步

由于本地机从来不会用到别名、访问限制和其它映射数据库。既然局域网内所有的机器的都由邮件中心服务器来管理和运行，我们可以很安全的把下列各个命令和手册页从本地机器上删除掉。

```
/usr/bin/newaliases
/usr/man/man1/newaliases.1
/usr/man/man5/aliases.5
```

使用命令删除以下这些文件：

```
[root@client]# rm -f /usr/bin/newaliases
[root@client]# rm -f /usr/man/man1/newaliases.1
[root@client]# rm -f /usr/man/man5/aliases.5
```

第四步

从你的所有本地 sendmail 服务器和客户机中把不必要的 Procmail 程序删除掉。既然本地机会将所有发往内部和外部的邮件发送到邮件中心服务器，我们不需要在客户机上使用象 Procmail 这样如此复杂的邮件投递代理。取而代之的是我们采用的“/bin/mail”程序。

使用命令删除 Procmail 文件：

```
[root@client]# rpm -e procmail
```

为所有的机器配置 /etc/sysconfig/sendmail 文件

“/etc/sysconfig/sendmail”文件用来指定 SENDMAIL 的一些配置信息。例如 sendmail 是否必须以守护进程的方式运行来监听网络上的邮件，如果队列中消息在等待了多少时间之后还没有被投递就要发送一条警告等等。

创建“**sendmail**”文件（touch /etc/sysconfig/sendmail）并加入：

```
DAEMON=yes
QUEUE=1h
```

“DAEMON=yes”告诉 sendmail 以守护进程的方式运行。这一行很有用，当你的运行 sendmail 的客户机被配置成不从外面直接接收邮件并把所有本地邮件发往邮件中心的时候，为了获得更高的安全性，最好不要让 sendmail 以守护进程的方式运行。在这种情况下，所有你需要做的事情就是把“DAEMON=yes”改为“DAEMON=no”。

为所有的机器配置 /etc/rc.d/init.d/sendmail 脚本文件

配置“/etc/rc.d/init.d/sendmail”文件用来启动和停止 Sendmail 守护进程。

创建“**sendmail**”脚本文件（touch /etc/rc.d/init.d/sendmail）并加入：

```
#!/bin/sh
#
# sendmail This shell script takes care of starting and stopping
#   sendmail.
#
# chkconfig: 2345 80 30
# description: Sendmail is a Mail Transport Agent, which is the program \
#   that moves mail from one machine to another.
# processname: sendmail
# config: /etc/sendmail.cf
# pidfile: /var/run/sendmail.pid
# Source function library.
. /etc/rc.d/init.d/functions
# Source networking configuration.
. /etc/sysconfig/network
# Source sendmail configuration.
if [ -f /etc/sysconfig/sendmail ] ; then
. /etc/sysconfig/sendmail
else
DAEMON=yes
QUEUE=1h
fi

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

[ -f /usr/sbin/sendmail ] || exit 0

RETVAL=0
```

```
# See how we were called.
case "$1" in
start)
# Start daemons.
echo -n "Starting sendmail: "
/usr/bin/newaliases > /dev/null 2>&1
for i in virtusertable access domaintable mailertable ; do
if [ -f /etc/mail/$i ] ; then
makemap hash /etc/mail/$i < /etc/mail/$i
fi
done
daemon /usr/sbin/sendmail $([ "$DAEMON" = yes ] && echo -bd) \
$([ -n "$QUEUE" ] && echo -q$QUEUE)
RETVAL=$?
echo
[ $RETVAL -eq 0 ] && touch /var/lock/subsys/sendmail
;;
stop)
# Stop daemons.
echo -n "Shutting down sendmail: "
killproc sendmail
RETVAL=$?
echo
[ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/sendmail
;;
restart|reload)
$0 stop
$0 start
RETVAL=$?
;;
status)
status sendmail
RETVAL=$?
;;
*)
echo "Usage: sendmail {start|stop|restart|status}"
exit 1
esac
exit $RETVAL
```

现在改变脚本的缺省权限，使脚本可执行：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/sendmail
```

用以下命令创建 rc.d links 到 Sendmail 的符号连接：

```
[root@deep]# chkconfig --add sendmail
```

用以下命令手工启动 Sendmail 服务器。

```
[root@deep]# /etc/rc.d/init.d/sendmail start
```

清理工作

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf sendmail-version/ sendmail.version.tar.gz
```

“rm”命令将把所有用于编译和安装 sendmail 的所有源文件删除。它也会删除“/var/tmp”目录下的压缩文件。

保证 Sendmail 的安全

Sendmail 受限 shell　smrsh

smrsh 程序的目的是作为在 mailer 中为 sendmail 定义的“/bin/sh”的替代 shell。smrsh 是一种受限 shell 工具，它通过“/etc/smrsh”目录来明确指定可执行文件的列表。简而言之，即便一个“坏蛋”可以使 sendmail 不通过别名文件和转发文件来运行其它程序，smrsh 限制了他或她可以执行的程序集。当它与 sendmail 程序一起使用的时候，smrsh 有效的将 sendmail 可以执行的程序的范围限制在 smrsh 目录之下。如果你是按照我们所讲的那样操作的话，smrsh 程序已经被编译和安装到你的计算机的“/usr/sbin/smrsh”目录之下。

第一步

我们所需要做的第一件事就是决定 smrsh 可以允许 sendmail 运行的命令列表。缺省情况下应当包含以下命令，但不局限于这些命令：

“/bin/mail” (如果在你的系统中安装了的话)
“/usr/bin/procmail” (如果在你的系统中安装了的话)

注意：不要在你的可接受命令列表里包括其它命令解释程序，例如 sh(1)，csh(1)，perl(1)，uudecode(1)及流编辑器 sed(1)。

第二步

你接着要在“/etc/smrsh”目录下装配允许 sendmail 运行的程序。为了避免程序重复和使工作更完美，我们最好不将程序拷贝到“/etc/smrsh”目录，而是在这个目录中创建这些程序的符号连接。

使用以下命令允许 **mail** 程序“/bin/mail”运行：

```
[root@deep]# cd /etc/smrsh
[root@deep]# ln -s /bin/mail mail
```

用以下命令允许 **procmail** 程序“/usr/bin/procmail”运行：

```
[root@deep]# cd /etc/smrsh
[root@deep]# ln -s /usr/bin/procmail procmail
```

这将允许位于“.forward”和“aliases”中的用户采用“|program”语法来运行 **mail** 及 **procmail** 程序。

第三步

现在我们可以配置 sendmail 使之使用受限 shell。mailer 程序在 sendmail 的配置文件“/etc/sendmail.cf”中仅占据了一行。你必须修改“sendmail.cf”文件中“Mprog”定义的那一行。将“/bin/sh”替换为“/usr/sbin/smrsh”。

编辑“sendmail.cf”文件（vi /etc/sendmail.cf）并改动下面这一行：

例如：

```
Mprog, P=/bin/sh, F=lsDFMoqeu9, S=10/30, R=20/40, D=$z:/, T=X-Unix, A=sh -c $u
```

应该被改为：

```
Mprog, P=/usr/sbin/smrsh, F=lsDFMoqeu9, S=10/30, R=20/40, D=$z:/, T=X-Unix, A=sh
-c $u
```

现在用以下命令手工重起 sendmail 进程：

```
[root@deep]# /etc/rc.d/init.d/sendmail restart
```

注意：在我们为邮件转储中心配置的“sendmail.mc”文件中我们已经用 m4 宏“FEATURE(`smrsh',`/usr/sbin/smrsh')”把包含“Mprog”的行配置成了使用受限 shell“/usr/sbin/smrsh”，因此不要为在见到邮件转储中心的配置文件“/etc/sendmail.cf”

中的“/usr/sbin/smrsh”属性已经被设置而感到惊讶。实际上，我们在上面所示的技术是为了网络上其它的空客户——“本地或邻居的客户机及服务器”准备的。他们的 sendmail 的配置文件“/etc/sendmail.cf”是由“/etc/sendmail.cf”宏配置文件生成的。

/etc/aliases 文件

如果没有加以正确和严格的管理的话，别名文件被用来获取特权。例如，很多发行版本在别名文件中带有“decode”别名。现在这种情况越来越少了。

这样做的目的是为用户提供一个通过 mail 传输二进制文件的方便的方式。在邮件的发送地，用户把二进制文件用“uuencode”转换成 ASCII 格式，并把结果邮递给接收地“decode”别名。那个别名通过管道把邮件消息发送到“/usr/bin/uuencode”程序，由这个程序来完成从 ASCII 转回到原始的二进制文件的工作。

删除“decode”别名。类似的，对于所有用于执行没有被放在 smrsh 目录下的程序的别名，你都要仔细的检查，可能它们都值得怀疑并应当删除它们。要想使你的改变生效，需要运行：

```
[root@deep]# /usr/bin/newaliases
```

编辑别名文件（vi /etc/aliases）并删除以下各行：

```
# Basic system aliases -- these MUST be present.
MAILER-DAEMON: postmaster
postmaster:    root

# General redirections for pseudo accounts.
bin:           root
daemon:        root
games:     root        删除这一行
ingres:    root            删除这一行
nobody:        root
system:        root            删除这一行
toor:      root        删除这一行
uucp:      root            删除这一行

# Well-known aliases.
manager:   root            删除这一行
dumper:        root            删除这一行
operator:      root            删除这一行

# trap decode to catch security attacks
decode:        root            删除这一行
```

```
# Person who should get root's mail
#root:      marc
```

别忘了运行“/usr/bin/newaliases”程序使改动生效

避免你的 Sendmail 被未授权的用户滥用

最新版本的 Sendmail (8.9.3)加入了很强的防止欺骗的特性。它们可以防止你的邮件服务器被未授权的用户滥用。编辑你的“/etc/sendmail.cf”文件，修改一下这个配置文件，使你的邮件服务器能够挡住欺骗邮件。

编辑“sendmail.cf”文件（vi /etc/sendmail.cf）并更改下面一行：

```
O PrivacyOptions=authwarnings
```

改为：

```
O PrivacyOptions=authwarnings,noexpn,novrfy
```

设置“noexpn”使 sendmail 禁止所有 SMTP 的“EXPN”命令，它也使 sendmail 拒绝所有 SMTP 的“VERB”命令。设置“novrfy”使 sendmail 禁止所有 SMTP 的“VRFY ”命令。这种更改可以防止欺骗者使用“EXPN”和“VRFY”命令，而这些命令恰恰被那些不守规矩的人所滥用。

SMTP 的问候信息

当 sendmail 接受一个 SMTP 连接的时候，它会向那台机器发送一个问候信息，这些信息作为本台主机的标识，而且它所做的第一件事就是告诉对方它已经准备好了。

编辑“sendmail.cf”文件（vi /etc/sendmail.cf）并更改下面一行：

```
O SmtpGreetingMessage=$j Sendmail $v/$Z; $b
```

改为：

```
O SmtpGreetingMessage=$j Sendmail $v/$Z; $b NO UCE C=xx L=xx
```

现在手工重起一下 sendmail 进程，使刚才所做的更改生效：

```
[root@deep]# /etc/rc.d/init.d/sendmail restart
```

以上的更改将影响到 Sendmail 在接收一个连接时所显示的标志信息。你应该把“`C=xx L=xx”条目中的“xx”换成你所在的国家和地区代码。例如，对于我而言，我采用“C=CA L=QC”表示加拿大，魁北克省。后面的更改其实不会影响任何东西。但这是“news.admin.net-abuse.email”新闻组的伙伴们推荐的合法做法。

限制可以审核邮件队列内容的人员

通常情况下，任何人都可以使用“mailq”命令来查看邮件队列的内容。为了限制可以审核邮件队列内容的人员在“/etc/sendmail.cf”文件中指定“restrictmailq”选项即可。在这种情况下，sendmail 只允许与这个队列所在目录的组属主相同的用户可以查看它的内容。这将允许权限为 0700 的邮件队列目录被完全保护起来，而我们限定的合法用户仍然可以看到它的内容。

编辑“sendmail.cf”文件（vi /etc/sendmail.cf）并更改下面一行：

```
O PrivacyOptions=authwarnings,noexpn,novrfy
```

改为：

```
O PrivacyOptions=authwarnings,noexpn,novrfy,restrictmailq
```

现在我们更改邮件队列目录的权限使它被完全保护起来：

```
[root@deep]# chmod 0700 /var/spool/mqueue
```

注意：我们已经在 sendmail.cf 中的“PrivacyOptions=”行中添加了“noexpn”和“novrfy”选项，现在在这一行中我们接着添加“restrictmailq”选项。

任何一个没有特权的用户如果试图查看邮件队列的内容会收到下面的信息：

```
[user@deep]$ /usr/bin/mailq
You are not permitted to see the queue
```

限制处理邮件队列的权限为 root

通常，任何人都可以使用“-q”开关来处理邮件队列，为限制处理邮件队列的权限“root”和邮件队列目录的属主在“/etc/sendmail.cf”文件中指定“restrictqrun”即可。

编辑“sendmail.cf”文件（vi /etc/sendmail.cf）并更改下面一行：

```
O PrivacyOptions=authwarnings,noexpn,novrfy,restrictmailq
```

改为：

```
O PrivacyOptions=authwarnings,noexpn,novrfy,restrictmailq,restrictqrun
```

任何一个没有特权的用户如果试图处理邮件队列的内容会收到下面的信息：

```
[user@deep]$ /usr/sbin/sendmail -q

You do not have permission to process the queue
```

在重要的 sendmail 文件上设置不可更改位

可以通过使用“chattr”命令而使重要的 Sendmail 文件不会被擅自更改，可以提高系统的安全性。具有“+i”属性的文件不能被修改：它不能被删除和改名，不能创建到这个文件的链接，不能向这个文件写入数据。只有超级用户才能设置和清除这个属性。

为“sendmail.cf”文件设置不可更改位：

```
[root@deep]# chattr +i /etc/sendmail.cf
```

为“sendmail.cw”文件设置不可更改位：

```
[root@deep]# chattr +i /etc/sendmail.cw
```

为“sendmail.mc”文件设置不可更改位：

```
[root@deep]# chattr +i /etc/sendmail.mc
```

为“null.mc”文件设置不可更改位：

```
[root@deep]# chattr +i /etc/null.mc
```

为“aliases”文件设置不可更改位：

```
[root@deep]# chattr +i /etc/aliases
```

为“access”文件设置不可更改位：

```
[root@deep]# chattr +i /etc/mail/access
```

参考文献

为获取更多的信息，你可以阅读以下手册页：

$ man aliases (5) - sendmail 别名文件

$ man makemap (8) - 为 sendmail 创建映射数据库

$ man sendmail (8) - 一种邮件传输代理程序

$ man mailq (1) - 打印邮件队列

$ man newaliases (1) - 重建邮件别名数据库

$ man mailstats (8) - 显示邮件统计信息

$ man praliases (8) - 显示系统邮件别名

Sendmail 管理工具

以下这些命令，是我们日常维护工作时经常要用到的。但是还有许许多多的命令我们没有列出，请阅读以下手册页和其他的参考文献，以便能够得到更加详细的信息。

newaliases

Newaliases 负责为“/etc/aliases”这个邮件别名文件重建随机访问数据库。每次更改“/etc/aliases”文件时都必须运行一次本命令，以使所做的更改能够生效。“newaliases”命令与“sendmail -bi”命令的功能完全相同。

使用以下命令运行 newaliases：

```
[root@deep]# /usr/bin/newaliases
```

makemap

Makemap 用来创建 sendmail 进行关键词映射检索时使用的数据库。它从标准的输入中读入数据然后把结果输出到指定的映射名字里去。当你打算为象 aliases、access

或 domaintable、mailertable 和 virtusertable 创建一个新的数据库的时候，你应该运行 makemap 命令。

使用下面的命令来利用 makemap 来为 access 创建一个新数据库：

```
[root@deep]# makemap hash /etc/mail/access.db < /etc/mail/access
```

其中<hash>是一种数据库格式，makemap 可以处理三种不同的数据库格式，它们是“hash”“ btree”和“dbm”。 </etc/mail/access.db>表示新数据库的位置和名称，</etc/mail/access>表示 makemap 所要读取的输入文件。

mailq

mailq 工具用来打印准备投递的邮件信息队列的汇总信息。

用下面的命令来打印准备投递的邮件信息队列的汇总信息

```
[root@deep]# mailq
```

Sendmail 用户工具

以下这些命令，是我们日常维护工作时经常要用到的。但是还有许许多多的命令我们没有列出，请阅读以下手册页和其他的参考文献，以便能够得到更加详细的信息。

mailstats

mailstats 工具用来显示邮件的统计信息。

用下面的命令来显示邮件的统计信息

```
[root@deep]# mailstats

Statistics from Tue Dec 14 20:31:48 1999

M msgsfr bytes_from msgsto bytes_to msgsrej msgsdis Mailer

8   7   7K  7   7K  0   0 local

=============================================================
```

```
T   7   7K  7   7K  0   0
```

praliases

praliases 工具用来显示当前系统中的别名，每个一行，并且没有固定顺序。

用下面的命令来显示当前系统中的别名：

```
[root@deep]# praliases

postmaster:root
daemon:root
root:admin
@:@
mailer-daemon:postmaster
bin:root
nobody:root
www:root
```

安装到系统中的文件

Sendmail 为邮件中心的安装文件

```
> /etc/rc.d/init.d/sendmail
> /etc/rc.d/rc0.d/K30sendmail
> /etc/rc.d/rc1.d/K30sendmail
> /etc/rc.d/rc2.d/S80sendmail
> /etc/rc.d/rc3.d/S80sendmail
> /etc/rc.d/rc4.d/S80sendmail
> /etc/rc.d/rc5.d/S80sendmail
> /etc/rc.d/rc6.d/K30sendmail
> /etc/sysconfig/sendmail
> /etc/mail
> /etc/mail/access
> /etc/mail/virtusertable
> /etc/mail/virtusertable.db
> /etc/mail/domaintable
> /etc/mail/domaintable.db
> /etc/mail/mailertable
> /etc/mail/mailertable.db
```

```
> /etc/mail/access.db
> /etc/smrsh
> /etc/aliases
> /etc/sendmail.cw
> /etc/sendmail.mc
> /etc/aliases.db
> /etc/sendmail.cf
> /usr/bin/newaliases
> /usr/bin/mailq
> /usr/bin/hoststat
> /usr/bin/purgestat
> /usr/lib/sendmail.hf
> /usr/lib/sendmail
> /usr/man/man1/mailq.1
> /usr/man/man1/newaliases.1
> /usr/man/man5/aliases.5
> /usr/man/man8/sendmail.8
> /usr/man/man8/mailstats.8
> /usr/man/man8/makemap.8
> /usr/man/man8/praliases.8
> /usr/man/man8/smrsh.8
> /usr/sbin/sendmail
> /usr/sbin/mailstats
> /usr/sbin/makemap
> /usr/sbin/praliases
> /usr/sbin/smrsh
> /var/log/sendmail.st
>/var/spool/mqueue
```

Sendmail 为本地服务器和客户机安装文件

```
> /etc/rc.d/init.d/sendmail
> /etc/rc.d/rc0.d/K30sendmail
> /etc/rc.d/rc1.d/K30sendmail
> /etc/rc.d/rc2.d/S80sendmail
> /etc/rc.d/rc3.d/S80sendmail
> /etc/rc.d/rc4.d/S80sendmail
> /etc/rc.d/rc5.d/S80sendmail
> /etc/rc.d/rc6.d/K30sendmail
> /etc/sysconfig/sendmail
> /etc/null.mc
> /etc/sendmail.cf
> /usr/bin/newaliases
```

```
> /usr/bin/mailq
> /usr/bin/hoststat
> /usr/bin/purgestat
> /usr/lib/sendmail.hf
> /usr/lib/sendmail
> /usr/man/man1/mailq.1
> /usr/man/man1/newaliases.1
> /usr/man/man5/aliases.5
> /usr/man/man8/sendmail.8
> /usr/man/man8/mailstats.8
> /usr/sbin/sendmail
> /usr/sbin/mailstats
> /var/log/sendmail.st
> /var/spool/mqueue
```

Linux OPENSSL 服务器

概述

OpenSSL 项目是一个合作的项目，其目标是开发一个健壮的、商业等级的、完整的开发源代码的工具包，用强大的加密算法来实现安全的 Socket 层（Secure Sockets Layer，SSL v2/v3）和传输层的安全性（Transport Layer Security，TLS v1）。

这个项目是由全世界的志愿者管理的，他们通过 Internet 相互交流、制定计划和开发 OpenSSL 工具包和相关文档。

加密的优势

数据的保密性

信息加密就是把纯文本的输入文件用加密算法转换成加密的文件以实现数据的保密。加密的过程需要用到密匙来加密数据然后再解密。没有了密匙，就无法解开加密的数据。数据加密之后，只有密匙要用一个安全的方法传送。加密过的数据可以公开地传送。

数据的一致性

加密也能保证数据的一致性。例如：加密的校验码，也叫做消息验证码（Message Authentication Code，MAC），能够校验用户提供的加密信息。加密的数据和 MAC 一起发送给接收者，接收者就可以用 MAC 来校验加密数据，保证数据没有被窜改过。

安全验证

加密的另外一个用途是用来作为个人的标识，用户的密匙可以作为他的安全验证的标识。

专利

各种各样的公司在世界各地拥有各种各样算法的专利。在使用加密算法之前必须检查一下这个算法有没有受到本国专利的限制。下面列出一些受到专利保护的算法（可能不确切）：

RSA Data Security 在美国和日本拥有 RSA 和 RC5 算法的专利。必须和 RSA Data Security 联系以得到许可条例。其主页是：http://www.rsa.com/□

RC4□ RSA Data Security□□□□□□□□□□□□□□□ RSA Data Security□□□□

IDEA 算法在澳大利亚、法国、德国、意大利、日本、荷兰、西班牙、瑞典、瑞士、英国和美国受专利保护。如果要使用这个算法必须得到许可，其主页是：http://www.ascom.ch/□

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

OpenSSL 的版本是 0.9.4。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >ssl1”，在编译和安装完软件之后运行命令“find /* > ssl2”，最后用命令“diff ssl1 ssl2 > ssl”找出变化。

软件包的来源

OpenSSL 的主页是：http://www.openssl.org/。

下载：openssl-0.9.4.tar.gz

编译

把软件包（tar.Z）解压缩：

```
[root@deep]# cp openssl_version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf openssl_version.tar.gz
```

编译与优化

转到 OpenSSL 目录下。

第一步

编辑“c_rehash”文件（vi +11 tools/c_rehash），把：

```
DIR=/usr/local/ssl
```

改为：

```
DIR=/usr
```

这个改变使编译和安装 OpenSSL 时用“/usr”这个默认目录。

第二步

在默认情况下 OpenSSL 把 Perl 程序的目录设置为“/usr/local/bin/perl”目录。必须改变所有脚本中的“#!/usr/local/bin/perl”这一行，因为在 RedHat Linux 中 Perl 的路径是“/usr/bin”。用下面的命令：

```
[root@deep]# perl util/perlpath.pl /usr/bin (where your perl program reside).
```

第三步

为了成功编译 OpenSSL，必须知道函数库所在的路径。用下面的命令设置 PATH 环境变量：

```
[root@deep]# export LD_LIBRARY_PATH=`pwd`
```

设置编译器的编译参数：

```
CC="egcs" \
./Configure linux-elf -DSSL_FORBID_ENULL \
--prefix=/usr \
--openssldir=/etc/ssl
```

注意：因为安全方面的原因要禁止“不加密”，所以“-DSSL_FORBID_ENULL”参数是必须的。

编辑“Makefile.ssl”文件（vi +52 Makefile.ssl），加入：

```
CFLAG= -DTHREADS -D_REENTRANT -DSSL_FORBID_ENULL -DL_ENDIAN -DTERMIO -O9
-funroll-loops -ffast-math -malign-double -mcpu=pentiumpro -march=pentiumpro
-fomit-frame-pointer -fno-exceptions -Wall -DSHA1_ASM -DMD5_ASM -DRMD160_ASM
```

这是编译 OpenSSL 的优化参数。

编辑“Makefile.ssl”文件（vi +77 Makefile.ssl），加入：

```
PROCESSOR= 686
```

注意：如果 CPU 是 Pentium，用 586 表示，PentiumPro/II/III 用 686，486 用 486。

```
[root@deep]# make -f Makefile
[root@deep]# make test
[root@deep]# make install
[root@deep]# mv /etc/ssl/misc/* /usr/bin/
[root@deep]# rm -rf /etc/ssl/misc/
[root@deep]# rm -rf /etc/ssl/lib/
[root@deep]# rm -f /usr/bin/CA.pl
[root@deep]# rm -f /usr/bin/CA.sh
[root@deep]# install -m 644 libRSAglue.a /usr/lib/
[root@deep]# install -m 644 rsaref/rsaref.h /usr/include/openssl/
[root@deep]# strip /usr/bin/openssl
[root@deep]# mkdir -p /etc/ssl/crl
```

“make –f”命令编译 OpenSSL 函数库（libcrypto.a 和 libssl.a）以及 OpenSSL 的二进制文件“openssl”。编译完之后函数库在顶层目录，二进制程序在“apps”子目录。完成编译之后，“make test”测试函数库是否正常。最后，“make install”安装 OpenSSL。

“mv”命令把“/etc/ssl/misc”目录下的所有文件移到“/usr/bin”目录下。因为在我们的系统中所有的二进制文件都在“/usr/bin”目录下，所以要把二进制文件都移到这个目录下。

“rm”命令删除“/etc/ssl/misc”和“/etc/ssl/lib”目录，因为这个目录中的文件都在别的地方了。“CA.pl”和“CA.sh”文件是小的脚本文件用来创建 CA 认证。这个脚本和“openssl ca”命令相关，而且有一些奇怪的要求。在默认情况下，OpenSSL 的配置不能很容易地使用“openssl ca”。所以我们后面会用“sign.sh”脚本来替换它们。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf openssl-version/ openssl_version.tar.gz
```

“rm”命令删除所有的编译和安装 OpenSSL 软件所需的源文件，并把 OpenSSL 软件的压缩包删除。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 OpenSSL 服务器，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“openssl.cnf”文件拷贝到“/etc/ssl”目录下
- 把“sign.sh”文件拷贝到“/usr/bin”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到相应的目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

配置 /etc/ssl/openssl.cnf 文件

这是 openssl 程序总的配置文件，可以配置密匙的过期时间、公司的名称、地址，等等。需要改变得配置在**[CA_default]**和**[req_distinguished_name]**这两个 section 里。

编辑“openssl.cnf”文件（vi /etc/ssl/openssl.cnf），加入并改变：

```
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#
RANDFILE = $ENV::HOME/.rnd
oid_file = $ENV::HOME/.oid
oid_section = new_oids
# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions =
# (Alternatively, use a configuration file that has only
```

```
# X.509v3 extensions in its main [= default] section.)
[ new_oids ]
# We can add new OIDs in here for use by 'ca' and 'req'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
# testoid2=${testoid1}.5.6
####################################################################
[ ca ]
default_ca = CA_default # The default ca section
####################################################################
[ CA_default ]
dir = /etc/ssl # Where everything is kept
certs = $dir/certs # Where the issued certs are kept
crl_dir = $dir/crl # Where the issued crl are kept
database = $dir/ca.db.index # database index file.
new_certs_dir = $dir/ca.db.certs # default place for new certs.
certificate = $dir/certs/ca.crt # The CA certificate
serial = $dir/ca.db.serial # The current serial number
crl = $dir/crl.pem # The current CRL
private_key = $dir/private/ca.key # The private key
RANDFILE = $dir/ca.db.rand # private random number file
x509_extensions = usr_cert # The extentions to add to the cert
# Extensions to add to a CRL. Note: Netscape communicator chokes on V2 CRLs
# so this is commented out by default to leave a V1 CRL.
# crl_extensions = crl_ext
default_days = 365 # how long to certify for
default_crl_days = 30 # how long before next CRL
default_md = md5 # which md to use.
Preserve = no # keep passed DN ordering
# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy = policy_match
# For the CA policy
[ policy_match ]
countryName = match
stateOrProvinceName = match
organizationName = match
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
```

```
# types.
[ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
####################################################################
[ req ]
default_bits = 1024
default_keyfile = privkey.pem
distinguished_name = req_distinguished_name
attributes = req_attributes
x509_extensions = v3_ca # The extentions to add to the self signed cert
[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = CA
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = Quebec
localityName = Locality Name (eg, city)
localityName_default = Montreal
0.organizationName = Organization Name (eg, company)
0.organizationName_default = Open Network Architecture
# we can do this but it is not needed normally :-)
#1.organizationName = Second Organization Name (eg, company)
#1.organizationName_default = World Wide Web Pty Ltd
organizationalUnitName = Organizational Unit Name (eg, section)
organizationalUnitName_default = Internet Department
commonName = Common Name (eg, YOUR name)
commonName_default = www.openarch.com
commonName_max = 64
emailAddress = Email Address
emailAddress_default = admin@openarch.com
emailAddress_max = 40
# SET-ex3 = SET extension number 3
[ req_attributes ]
challengePassword = A challenge password
challengePassword_min = 4
challengePassword_max = 20
unstructuredName = An optional company name
```

```
[ usr_cert ]
# These extensions are added when 'ca' signs a request.
# This goes against PKIX guidelines but some CAs do it and some software
# requires this to avoid interpreting an end user certificate as a CA.
basicConstraints=CA:FALSE
# Here are some examples of the usage of nsCertType. If it is omitted
# the certificate can be used for anything *except* object signing.
# This is OK for an SSL server.
# nsCertType = server
# For an object signing certificate this would be used.
# nsCertType = objsign
# For normal client use this is typical
# nsCertType = client, email
# and for everything including object signing:
# nsCertType = client, email, objsign
# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment
# This will be displayed in Netscape's comment listbox.
nsComment = "OpenSSL Generated Certificate"
# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy
# Copy subject details
# issuerAltName=issuer:copy
#nsCaRevocationUrl = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName
[ v3_ca]
# Extensions for a typical CA
# PKIX recommendation.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always
# This is what PKIX recommends but some broken software chokes on critical
# extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true
# Key usage: this is typical for a CA certificate. However since it will
```

```
# prevent it being used as an test self-signed certificate it is best
# left out by default.
# keyUsage = cRLSign, keyCertSign
# Some might want this also
# nsCertType = sslCA, emailCA
# Include email address in subject alt name: another PKIX recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy
# RAW DER hex encoding of an extension: beware experts only!
# 1.2.3.5=RAW:02:03
# You can even override a supported extension:
# basicConstraints= critical, RAW:30:03:01:01:FF
[ crl_ext ]
# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a CRL.
# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
```

注意：编译和安装完 OpenSSL 程序之后，“openssl.cnf”文件在服务器上已经存在了，可以在“/et/ssl”目录下找到。没有必要改变这个文件中所有的默认配置，经常需要修改的只是**[CA_default]**和**[req_distinguished_name]**这两个 section。

创建 /usr/bin/sign.sh 脚本文件

“openssl ca”命令有一些奇怪的要求，OpenSSL 默认的配置并不是很容易直接使用“openssl ca”，因此我们用“sign.sh”脚本文件替代它。

创建“sign.sh”脚本（touch /usr/bin/sign.sh），加入：

```
#!/bin/sh
##
## sign.sh -- Sign a SSL Certificate Request (CSR)
## Copyright (c) 1998-1999 Ralf S. Engelschall, All Rights Reserved.
##
# argument line handling
CSR=$1
if [ $# -ne 1 ]; then
echo "Usage: sign.sign <whatever>.csr"; exit 1
fi
if [ ! -f $CSR ]; then
echo "CSR not found: $CSR"; exit 1
fi
case $CSR in
```

```
*.csr ) CERT="`echo $CSR | sed -e 's/\.csr/.crt/'`" ;;
* ) CERT="$CSR.crt" ;;
esac
# make sure environment exists
if [ ! -d ca.db.certs ]; then
mkdir ca.db.certs
fi
if [ ! -f ca.db.serial ]; then
echo '01' >ca.db.serial
fi
if [ ! -f ca.db.index ]; then
cp /dev/null ca.db.index
fi
# create an own SSLeay config
cat >ca.config <<EOT
[ ca ]
default_ca = CA_own
[ CA_own ]
dir = /etc/ssl
certs = /etc/ssl/certs
new_certs_dir = /etc/ssl/ca.db.certs
database = /etc/ssl/ca.db.index
serial = /etc/ssl/ca.db.serial
RANDFILE = /etc/ssl/ca.db.rand
certificate = /etc/ssl/certs/ca.crt
private_key = /etc/ssl/private/ca.key
default_days = 365
default_crl_days = 30
default_md = md5
preserve = no
policy = policy_anything
[ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
EOT
# sign the certificate
echo "CA signing: $CSR -> $CERT:"
openssl ca -config ca.config -out $CERT -infiles $CSR
echo "CA verifying: $CERT <-> CA cert"
```

```
openssl verify -CAfile /etc/ssl/certs/ca.crt $CERT
# cleanup after SSLeay
rm -f ca.config
rm -f ca.db.serial.old
rm -f ca.db.index.old
# die gracefully
exit 0
```

现在，让这个脚本可执行并改变它的默认权限：

```
[root@deep]# chmod 755 /usr/bin/sign.sh
```

注意：解开“floppy.tgz”文件之后，可以在“mod_ssl-version/pkg.contrib”目录下找到“sign.sh”文件。要根据实际情况改变**[CA_own]**这一节，而且不要忘了改变“**openssl verify -CAfile /etc/ssl/certs/ca.crt $CERT**”这一行。

保证 OPENSSL 的安全

把密匙设置成只能被超级用户“root”可执行和可写。必须保证其他人不能访问这个文件。

用下面的命令使得密匙只能被“root”可执行和可写：

```
[root@deep]# chmod 600 /etc/ssl/certs/ca.crt
[root@deep]# chmod 600 /etc/ssl/certs/server.crt
[root@deep]# chmod 600 /chroot/httpd/etc/ssl/private/ca.key
[root@deep]# chmod 600 /chroot/httpd/etc/ssl/private/server.key
```

命令

下面列出的是一些我们经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

在下面这个例子中，我们指导你如何为 Apache Web 服务器创建认证：

注意：下面所有的命令都在“/etc/ssl”目录下运行的。

为 Apache 服务器创建用口令保护的 RSA 私人密匙

```
[root@deep]# openssl genrsa -des3 -out server.key 1024
Generating RSA private key, 1024 bit long modulus
```

```
.....................+++++
.....+++++
e is 65537 (0x10001)
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
```

请把“server.key”文件备份起来，记住只有在安全的地方才能输入口令。

用服务器的 RSA 私人密匙创建 Certificate Signing Request (CSR)

```
[root@deep]# openssl req -new -key server.key -out server.csr
Enter PEM pass phrase:
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) [CA]:
State or Province Name (full name) [Quebec]:
Locality Name (eg, city) [Montreal]:
Organization Name (eg, company) [Open Network Architecture]:
Organizational Unit Name (eg, section) [Internet Department]:
Common Name (eg, YOUR name) [www.openarch.com]:
Email Address [admin@openarch.com]:
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:.
An optional company name []:.
```

现在可以把这个 CSR（Certificate Signing Request）发送给认证机构（Certifying Authority, CA），让它签订这个 CSR。CSR 被签订之后，就成为真正的证书（Certificate），可以被 Apache 使用。有下面两种选择。第一：可以让商业的 CA，如：Verisign 或 Thawte 签订 CSR。通常需要在 Web 上登记 CSR，然后支付签订所需的费用，接着等待签订后的证书，最后收到证书把它存成 server.crt 文件。第二：可以用自己的 CA 来签订证书。下面介绍如何用自己的 CA 签订 CSR。

首先确信当 OpenSSL 提示输入“CommonName”的时候，输入服务器的 FQDN（Fully Qualified Domain Name，完全合格的域名）。例如：如果要为今后用

http://www.mydomain.com 访问的站点创建 CSR，在这里就需要输入 www.mydomain.com。

为自己的 CA 创建 RSA 私人密匙

```
[root@deep]# openssl genrsa -des3 -out ca.key 1024


Generating RSA private key, 1024 bit long modulus
.........................+++++
.........................................+++++
e is 65537 (0x10001)
Enter PEM pass phrase:
```

备份好 ca.key 文件。注意只有在安全的地方才能输入口令。

用 CA 的 RSA 密匙创建自我签订的证书 x509 结构

```
[root@deep]# openssl req -new -x509 -days 365 -key ca.key -out ca.crt


Enter PEM pass phrase:
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) [CA]:
State or Province Name (full name) [Quebec]:
Locality Name (eg, city) [Montreal]:
Organization Name (eg, company) [Open Network Architecture]:
Organizational Unit Name (eg, section) [Internet Department]:CA Marketing
Common Name (eg, YOUR name) [www.openarch.com]:
Email Address [admin@openarch.com]:
[root@deep]# mv server.key private/
[root@deep]# mv ca.key private/
[root@deep]# mv ca.crt certs/
```

注意：当使用“-x509”参数的时候，“req”命令创建了自我签订的证书。

签订一个证书请求　用自己的 CA

准备一个用于签订证书的脚本是必须的，因为“openssl ca”命令有一些很怪的要求而且在默认情况下 OpenSSL 的配置不是很容易就可以直接使用“openssl ca”。这就需要一个名为“sign.sh”的脚本文件，解开“floppy.tgz”之后就可以在相应的目录中找到。用这个脚本完成签订。

现在用这个 CA 签订服务器的 CSR，这样就能为 Apache 服务器创建真正的 SSL 证书（假定你已经有了“server.csr”这个文件）。

```
[root@deep]# /usr/bin/sign.sh server.csr


Using configuration from ca.config
Enter PEM pass phrase:
Check that the request matches the signature
Signature ok
The Subjects Distinguished Name is as follows
countryName :PRINTABLE:'CA'
stateOrProvinceName :PRINTABLE:'Quebec'
localityName :PRINTABLE:'Montreal'
organizationName :PRINTABLE:'Open Network Architecture'
organizationalUnitName :PRINTABLE:'Internet Department'
commonName :PRINTABLE:'www.openarch.com'
emailAddress :IA5STRING:'admin@openarch.com'
Certificate is to be certified until Dec 1 14:59:29 2000 GMT (365 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
CA verifying: server.crt <-> CA cert
server.crt: OK
```

上面的命令签订了 CSR 并把结果存成“server.crt”文件。

```
[root@deep]# mv server.crt certs/
```

现在有两个文件：“server.key”和“server.crt”。可以在 Apache 的配置文件“httpd.conf”文件中加上：

```
SSLCertificateFile /etc/ssl/certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/server.key
```

“server.csr”文件可以不要了。

```
[root@deep]# rm -f server.csr
```

安装到系统中的文件

```
> /etc/ssl
> /etc/ssl/crl
> /etc/ssl/certs
> /etc/ssl/private
> /etc/ssl/openssl.cnf
> /usr/bin/openssl
> /usr/bin/c_rehash
> /usr/bin/sign.sh
> /usr/bin/c_hash
> /usr/bin/c_info
> /usr/bin/c_issuer
> /usr/bin/c_name
> /usr/bin/der_chop
> /usr/include/openssl
> /usr/include/openssl/e_os.h
> /usr/include/openssl/e_os2.h
> /usr/include/openssl/crypto.h
> /usr/include/openssl/tmdiff.h
> /usr/include/openssl/opensslv.h
> /usr/include/openssl/opensslconf.h
> /usr/include/openssl/ebcdic.h
> /usr/include/openssl/md2.h
> /usr/include/openssl/md5.h
> /usr/include/openssl/sha.h
> /usr/include/openssl/mdc2.h
> /usr/include/openssl/hmac.h
> /usr/include/openssl/ripemd.h
> /usr/include/openssl/des.h
> /usr/include/openssl/rc2.h
> /usr/include/openssl/rc4.h
> /usr/include/openssl/rc5.h
> /usr/include/openssl/idea.h
> /usr/include/openssl/blowfish.h
> /usr/include/openssl/cast.h
```

```
> /usr/include/openssl/bn.h
> /usr/include/openssl/rsa.h
> /usr/include/openssl/dsa.h
> /usr/include/openssl/dh.h
> /usr/include/openssl/buffer.h
> /usr/include/openssl/bio.h
> /usr/include/openssl/stack.h
> /usr/include/openssl/safestack.h
> /usr/include/openssl/lhash.h
> /usr/include/openssl/rand.h
> /usr/include/openssl/err.h
> /usr/include/openssl/objects.h
> /usr/include/openssl/evp.h
> /usr/include/openssl/asn1.h
> /usr/include/openssl/asn1_mac.h
> /usr/include/openssl/pem.h
> /usr/include/openssl/pem2.h
> /usr/include/openssl/x509.h
> /usr/include/openssl/x509_vfy.h
> /usr/include/openssl/x509v3.h
> /usr/include/openssl/conf.h
> /usr/include/openssl/txt_db.h
> /usr/include/openssl/pkcs7.h
> /usr/include/openssl/pkcs12.h
> /usr/include/openssl/comp.h
> /usr/include/openssl/ssl.h
> /usr/include/openssl/ssl2.h
> /usr/include/openssl/ssl3.h
> /usr/include/openssl/ssl23.h
> /usr/include/openssl/tls1.h
> /usr/include/openssl/rsaref.h
> /usr/lib/libcrypto.a
> /usr/lib/libssl.a
> /usr/lib/libRSAglue.a
> /var/lock/subsys/named
```

Linux Imap & Pop 服务器

概述

所谓邮件服务器，通常指运行如下一种或几种服务的服务器，即运行 IMAP 服务，POP3 服务，POP2 服务或者 SMTP 服务，下面我所讲述的涉及 IMAP4，POP3，POP2 服务，所有的这些均来自一个软件包，而 Sendmail 是一种被广泛采用 SMTP 服务器。

POP 是 Post Office Protocol 缩写，中文即“邮局协议”，它允许用户列出邮件，接收邮件，删除邮件，在 Linux 中提供了多种 POP 服务器。对大多数用户来说，各种 Linux 发行版中自带软件包已经足够用了，IMAP 服务是 POP 服务的扩展与延伸，可以方便地管理多个邮件帐号，也可以让多个用户共用一个帐号，或者是只把邮件留在服务器上，把邮件头部或邮件体下载到客户机上而不必下载附件，等等增强特性。

对于比较严格的邮件服务需求而言，IMAP 是一种理想的选择。大多数 Linux 发行版本带有的缺省 POP 和 IMAP 服务器（它们被捆绑在一种称为 Imapd 的软件包之中），足以满足绝大部分要求。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

Imap 的版本号是 4.5；

软件包的来源

Imap 主页：http://www.washington.edu/imap

必须确保下载：imap-4.5.tar.Z

安装软件包需要注意的问题

在安装 imap 前后保存一下文件列表对你也许是一个好主意，而后用 diff 比较一下两个文件列表从而找出 imap 的文件被安装到哪里去了，方法是在安装 imap 之前运行一下 “find /*>imap1”，而在安装 imap 服务之后运行“find /* > imap2”，接着执行命令“diff imap1 imap2 >imap”，从而得到安装文件列表。

编译和安装

把软件包（tar.Z）解压缩：

```
[root@deep]# cp imap-version.tar.Z /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf imap-version.tar.Z
```

编译与优化

使用 cd 命令进入 Imap 目录，而后在终端上键入如下命令：

编辑“Makefile”文件（vi+689 src/osdep/unix/Makefile）并做如下改动：

```
sh -c '(test -f /usr/include/sys/statvfs.h -a $(OS) != sc5 -a $(OS) != sco) &&
$(LN) flocksun.c flockbsd.c || $(LN) flocksv4.c flockbsd.c'
```

改为：

```
sh -c '(test -f /usr/include/sys/statvfs.h -a $(OS) != sc5 -a $(OS) != sco -a
$(OS) != lnx) && $(LN) flocksun.c flockbsd.c || $(LN) flocksv4.c flockbsd.c'
```

这会更改“sys/starvfs”文件，在 linux 中的 glibc 2.1 中所带的这个文件与 Sun 中所带的文件不同。

编辑一下“Makefile”文件。（vi+355 src/osdep/unix/Makefile），并做如下改动：

```
BASECFLAGS="-g -fno-omit-frame-pointer -O6 -DNFSKLUDGE" \
```

改为：

```
BASECFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-pointer -fno-exceptions -DNFSKLUDGE" \
```

这是编译 IMAP/POP 服务器的一种优化设置。

编辑“Makefile”文件（vi+112 src/osdep/unix/Makefile），并做如下改动：

```
BUILDOPTIONS= EXTRACFLAGS="$(EXTRACFLAGS)"\
```

改为：

```
BUILDOPTIONS= EXTRACFLAGS= -DDISABLE_POP_PROXY=1 -DIGNORE_
LOCK_EACCES_ERRORS=1"$(EXTRACFLAGS)"\
```

在缺省情况下，ipop2d/ipop3d 服务器提供的 POP→IMAP 访问代理，允许 POP 的客户通过一个 POP 服务器访问 IMAP 的邮件，可以通过设定“DDISABL-POP-PROXY=1”选项禁止这项功能。

“_D IGNORE_LOCK_EACCES_ERRORS=1”选项禁止在因为试图创建一个邮箱加锁文件失败时而产生 EACCES 错误时出现的“Mailbax vulneralbe – directory must have 1777 protection”警告信息。

编辑“Makefile”文件（vi+58 svc/osdep/unix/Makefile），并做如下改动：

```
ACTIVEFILE=/usr/lib/news/active
```

改为：

```
ACTIVEFILE=/var/lib/news/active
```

```
SPOOLDIR=/usr/spool
```

改为：

```
SPOOLDIR=/var/spool
```

```
RSHPATH=/usr/ucb/rsh
```

改为：

```
RSHPATH=/usr/bin/rsh
```

“ACTIVEFILE=”这行表明 Linux 中 IMAP/POP 活动目录的路径，“SPOOLDIR=”是 Linux IMAP/POP 服务器的缓冲池目录路径。“RSHPATH=”表明了系统中的 rsh 目录的路径，必须指出的是，尽管我们不用 rsh 服务，但仍要指明了 rsh 目录的正确路径。

编辑“Makefile”文件（vi+85 src/osdep/unix/Makefile）并做如下改动：

```
CC=cc
```

改为：

```
CC=egcs
```

这一行代表我们编译 IMAP/POP 服务器所用的 GCC 编译器的名称，此时是（egcs）

```
[root@deep]# make lnp
[root@deep]# mv ./c-client/c-client.a ./c-client/libimap.a
[root@deep]# install -m 644 c-client/libimap.a /usr/lib/
[root@deep]# install -m 644 ./src/ipopd/ipopd.8c /usr/man/man8/ipopd.8c
[root@deep]# install -m 644 ./src/imapd/imapd.8c /usr/man/man8/imapd.8c
[root@deep]# install -s -m 755 ./ipopd/ipop2d /usr/sbin/
[root@deep]# install -s -m 755 ./ipopd/ipop3d /usr/sbin/
[root@deep]# install -s -m 755 ./imapd/imapd /usr/sbin/
[root@deep]# mkdir -p /usr/include/imap
[root@deep]# install -m 644 ./c-client/*.h /usr/include/imap/
[root@deep]# install -m 644 ./src/osdep/tops-20/shortsym.h /usr/include/imap/
[root@deep]# chown root.mail /usr/sbin/ipop2d
[root@deep]# chown root.mail /usr/sbin/ipop3d
[root@deep]# chown root.mail /usr/sbin/imapd
```

上面的命令完成：配制软件以确保在系统中能够有成功编译本软件包所需的函数和类库；把源文件编译成二进制执行码；把可执行文件和支持文件安装到正确的位置。

注意一下上面“make lnp”命令，它将使你的 Linux 系统具有插入认证模块（Pluggable Authentication Modules(PAM)）能力，这样就有更好的安全性。

“mkdir”命令在“usr/include”目录下创建一下叫 Imap 的新目录，这个新目录用于存储与 C-Client/*及 shartsym.h，这些是与 imapd 程序有关的头文件。

“chown”命令把 pop2d，pop3d 和 imapd 的所有权改为超级用户 root 及 mail 用户组。

注意：出于安全考虑：如果只打算使用 imapd 服务，请把 pop2d，pop3d 删除，如果想同时使用 ipop 服务请把 imapd 删除，如果想同时使用 imapd 和 ipopd 服务，就把两个都保留下来。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf imap-version/ imap-version.tar.Z
```

rm 命令把用于编译安装 IMAP/POP 服务的所有源文件清除，同时也把 IMAP/POP 的压缩文件从“var/tmp”目录中删除。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，在相应的目录下会发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

对于运行 IMAP/POP 服务器而言，下面这些文件必须存在而且必须把它们创建或拷贝到在适当的目录中。

- 如果打算用 **imapd** 服务，拷贝 **imapd** 文件到“/etc/pam.d”目录。
- 如果打算用 **popd** 服务，拷贝 **pop** 文件到“/etc/pam.d”目录。

可以从 floppy.tgz 压缩文件中找到下列配置文件，把它们从解压的 floppy.tgz 文件中到适当的目录中，或者直接从书中剪帖到相关文件中。

配制 etc/pam.d/imap 文件

配制“/etc/pam.d/imap”文件使其可以使用 pam 认证。

创建“imap”文件(touch /etc/pam.d/imap)并加上如下几行：

```
#%PAM-1.0
auth required  /lib/security/pam_pwdb.so shadow nullok
account required /lib/security/pam_pwdb.so
```

注意：本文件只有在使用 IMAP 服务时才是必须的。

配制 /etc/pam.d/pop 文件

配制“/etc/pam.d/pop”文件使其通过 pam 认证。

创建 **pop** 文件(touch /etc/pam.d/pop)并加上如下几行：

```
#%PAM-1.0
auth required /lib/security/pam_pwdb.so shadow nullok
account required /lib/security/pam_pwdb.so
```

注意：本文件只有在使用 POP 服务时才是必须的。

IMAP/POP 的安全信息

是否真的需要 IMAP/POP 服务？

某些版的 IMAP/POP 程序中存在严重的缓冲区溢出错误，可以被黑客利用来以“root”身份执行一些命令。确保已经把软件的版本升级到 4.5 或更高。有些版本的 POP 服务器不报告登录失败信息，这样有的攻击者就可以在你完全不知情的情况下强行破解密码。如果出现这种情况，请尽快升级软件。

应当清醒地知到 IMAP/POP 程序在缺省的情况下使用明文方式传送密码，任何一个人都可以利用网络嗅探程序（sniffer）监听网络从而获取你的用户名和密码，进而以之登录。在使用 Linux 系统的邮件阅读器的时候，并不需要运行一个本地的 IMAP/POP 服务程序。检查一下你的配置，如果在使用远程或外部 IMAP/POP 服务器，请关闭或删除本地的守护程序（daemon）。

而且如果打算通过 Internet 使用 Web 界面来阅读邮件，采用 SSL 协议来加密你与 IMAP/POP 服务器之间的信息交流将是一种不错的主意。参见第五部分，软件相关的参考手册，第十章，服务器软件，在 Linux Apache 服务器一节有更详细的介绍。

更多的资料

为获取更加详细的信息，可以参见下面几个 man 说明。

$ man imapd (8C) - Internet Message Access Protocol server

$ man ipopd (8C) - Post Office Protocol server

安装到系统中的文件

```
> /usr/include/imap
> /usr/include/imap/dummy.h
> /usr/include/imap/env.h
> /usr/include/imap/env_unix.h
> /usr/include/imap/fdstring.h
> /usr/include/imap/flstring.h
> /usr/include/imap/fs.h
> /usr/include/imap/ftl.h
> /usr/include/imap/imap4r1.h
> /usr/include/imap/linkage.h
> /usr/include/imap/lockfix.h
> /usr/include/imap/mail.h
> /usr/include/imap/mbox.h
> /usr/include/imap/mbx.h
> /usr/include/imap/mh.h
> /usr/include/imap/misc.h
> /usr/include/imap/mmdf.h
> /usr/include/imap/mtx.h
> /usr/include/imap/mx.h
> /usr/include/imap/netmsg.h
> /usr/include/imap/news.h
> /usr/include/imap/newsrc.h
> /usr/include/imap/nl.h
```

```
> /usr/include/imap/nntp.h
> /usr/include/imap/os_a32.h
> /usr/include/imap/os_a41.h
> /usr/include/imap/os_aix.h
> /usr/include/imap/os_aos.h
> /usr/include/imap/os_art.h
> /usr/include/imap/os_asv.h
> /usr/include/imap/os_aux.h
> /usr/include/imap/os_bsd.h
> /usr/include/imap/os_bsi.h
> /usr/include/imap/os_cvx.h
> /usr/include/imap/os_d-g.h
> /usr/include/imap/os_drs.h
> /usr/include/imap/os_dyn.h
> /usr/include/imap/os_hpp.h
> /usr/include/imap/os_isc.h
> /usr/include/imap/os_lnx.h
> /usr/include/imap/os_lyn.h
> /usr/include/imap/os_mct.h
> /usr/include/imap/os_mnt.h
> /usr/include/imap/os_nxt.h
> /usr/include/imap/os_os4.h
> /usr/include/imap/os_osf.h
> /usr/include/imap/os_ptx.h
> /usr/include/imap/os_pyr.h
> /usr/include/imap/os_qnx.h
> /usr/include/imap/os_s40.h
> /usr/include/imap/os_sc5.h
> /usr/include/imap/os_sco.h
> /usr/include/imap/os_sgi.h
> /usr/include/imap/os_shp.h
> /usr/include/imap/os_slx.h
> /usr/include/imap/os_sol.h
> /usr/include/imap/os_sos.h
> /usr/include/imap/os_sun.h
> /usr/include/imap/os_sv2.h
> /usr/include/imap/os_sv4.h
> /usr/include/imap/os_ult.h
> /usr/include/imap/os_vu2.h
> /usr/include/imap/osdep.h
> /usr/include/imap/phile.h
> /usr/include/imap/pop3.h
> /usr/include/imap/pseudo.h
> /usr/include/imap/rfc822.h
```

```
> /usr/include/imap/smtp.h
> /usr/include/imap/tcp.h
> /usr/include/imap/tcp_unix.h
> /usr/include/imap/tenex.h
> /usr/include/imap/unix.h
> /usr/include/imap/utf8.h
> /usr/include/imap/shortsym.h
> /usr/lib/libimap.a
> /usr/man/man8/ipopd.8c
> /usr/man/man8/imapd.8c
> /usr/sbin/ipop2d
> /usr/sbin/ipop3d
> /usr/sbin/imapd
> /etc/pam.d/imap
> /etc/pam.d/pop
```

Linux MM 共享内存库

概述

如果想在 Apache/EAPI 中具有共享内存的支持，那么就要建立 MM 共享内存库。在这种情况下，它将允许 mod_ssl 使用一种高效的基于 RAM 的会话（session）缓存代替基于磁盘的会话缓存。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

Mm 的版本号是 1.0.12。

软件包的来源

MM 的主页：http://www.engelschall.com/sw/mm/

必须确保下载：mm-1.0.12.tar.gz

安装软件包需要注意的问题

在安装 MM 前后保存一下文件列表对你也许是一个好主意，而后用 Diff 比较一下两个文件列表从而找出 MM 的文件被安装到哪里去了，方法是在安装 MM 之前运行一下“find /*>MM1”，而在安装 MM 服务之后运行“find /* > MM2”，接着执行命令“diff MM1 MM2 >MM”，从而得到安装文件列表。

编译程序

把软件包（tar.gz）解压缩：

```
[root@deep]# cp mm_version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf mm_version.tar.gz
```

编译和安装

cd 进入新的 MM 目录然后在终端上键入如下命令：

```
./configure \
--disable-shared \
--prefix=/usr
```

这一步告诉 MM 对于当前的硬件配置：禁用共享库。

```
[root@deep]# make
[root@deep]# make test
[root@deep]# make install
```

注意：“make test”命令将做一些重要的测试，从而在安装本程序之前验证它是否能够正常的工作，并做出正确的反应。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf mm-version/ mm_version.tar.gz
```

rm 命令把用于编译安装 mm 服务的所有源文件清除，同时也把 MM 的压缩文件从“var/tmp”目录中清除出去。

更多的资料

为获取更加详细的信息，可以参见下面几个 man 说明。

MM (3) — Shared Memory Library

mm-config (1) — MM library configuration/build utility

安装到系统中的文件

```
>usr/bin/mm-config
>usr/include/mm.h
>usr/lib/libmm.la
>usr/lib/libmm.a
>usr/man/man1/mm-config.1
>sr/man/man3/mm.3
```

Linux Samba 服务器

概述

Samba 是一种 PC 之间共享文件和打印机及其它信息的协议。例如，它可以提供当前可用文件或打印机列表支持。本身支持这个协议的操作系统有 Windows95/98/NT、OS/2 和 Linux。其它的一些操作系统,如 DOS、Windows、VMS、各种 UNIX、MVS 也可通过增加模块来支持它。

Apple Macs 系列微机和某些 Web 浏览器也可使用这个协议。SMB 的替代产品包括 Netware，NFS，AppleTalk，Banyan Vines, Decnet 等等。它们都有各自的优点，但没有是一种是被公认的标准，而且广泛地安装在台式机上。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

Samba 的版本号是 2.0.6

软件包的来源

MM 的主页：http://us1.samba.org/samba/samba.html

必须确保下载：samba-2.0.6.tar.gz

安装软件包需要注意的问题

在安装 Samba 前后保存一下文件列表对你也许是一个好主意，而后用 diff 比较一下两个文件列表从而找出 Samba 的文件被安装到哪里去了，方法是在安装 Samba 之前运行一下“find /* > smb1”，而在安装 Samba 服务之后运行“find /* > smb2”，接着执行命令“diff smb1 smb2 > smb”从而得到安装文件列表。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp samba.version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf samba.version.tar.gz
```

配置

cd 进入新的 Samba 目录，然后在终端上键入如下命令：

编辑“Makefile”文件(vi +28 Makefile.in)并做如下改动：

```
SBINDIR = @bindir@
```

改为：

```
SBINDIR = @sbindir@
```

```
VARDIR = @localstadir@
```

改为：

```
VARDIR = /var/log/samba
```

上面的命令将二进制文件的 bin 目录设为“/usr/sbin”目录而 Samba 的日志文件位于“/var/log/Samba 子”目录。

编辑“convert_smbpasswd”文件（vi +10 script/convert_smbpasswd）并做如下改动：

```
nawk 'BEGIN {FS=":"}
```

改为：

```
gawk 'BEGIN {FS=":"}
```

它将指定用 GNU 版本的 awk 文件处理工具取代 Bell 研究实验室版本的 awk 程序来处理“smbpasswd”文件，文件“convert_smbpasswd”用于将格式为 Smb1.9.18 的 “smbpasswd”文件转换成一种 Samba2.0 格式的“smbpasswd”文件。

编辑“include.h”文件，（vi +655 include/include.h）并删除几行。

删除下面几行：

```
#ifdef strcat
#undef strcat
#endif /* strcat */
#define strcat(dest,src) __ERROR__XX__NEVER_USE_STRCAT___;
```

编辑“smbmount.c”文件，（vi +99 client/smbmount.c）并做如下改变：

```
static void close_our_files(int client_fd)
{
int i;
for (i = 0; i < 256; i++) {
if (i == client_fd) continue;
close(i);
}
```

改为：

```
static void close_our_files(int client_fd)
{
struct rlimit limits;
int i;
getrlimit(RLIMIT_NOFILE,&limits);
for (i = 0; i < limits.rlim_max; i++) {
if (i == client_fd) continue;
close(i);
}
```

以上这两步改变将使“include.h”和“smbmount.c”文件和 RedHat glibc 2.1 库兼容。

编译和优化

在终端上键入如下命令：

```
CC="egcs" \
./configure \
--prefix=/usr \
--libdir=/etc \
--with-lockdir=/var/lock/samba \
--with-privatedir=/etc \
--with-swatdir=/usr/share/swat \
--with-pam \
--with-mmap
```

注意：选项“--with-mmap”在某些机器上可以极大的提高应用程序的性能，在其它机器上，它不起任何作用，还有机器会降低其性能。

这些设置告诉 Samba 对于当前的硬件配置：

- 包含 PAM 密码数据库验证支持
- 包含试验性的 MMAP 内核支持（提高应用程序性能）

```
[root@deep]# make all
[root@deep]# make install

[root@deep]# install -m 755 script/mksmbpasswd.sh /usr/bin/
[root@deep]# rm -rf /usr/share/swat/
[root@deep]# rm -f /usr/sbin/swat
[root@deep]# rm -f /usr/man/man8/swat.8
[root@deep]# mkdir -p /var/lock/samba
[root@deep]# mkdir -p /var/spool/samba
[root@deep]# chmod 1777 /var/spool/samba/
```

“install”命令把“mksmbpasswd.sh”脚本安装到“/usr/bin/”目录，这个脚本是设置 Samba 允许用户通过“smbpasswd”文件连接到我们的服务器的文件。参见本文后面的部分有关怎样设置和应用 Samba 密码。

“rm”命令删除“/usr/share/swat”目录及其中所有文件，同时也删除位于“/usr/sbin/”的 Swat 的二进制文件。SWAT 程序是一种基于 web 的配置工具，它允许用户通过 Web 配置“smb.conf”文件。这个工具监听服务器的 901 端口，当然

为了使用 SWAT 工具，必须有一个 web 服务器在后台运行，例如：Apache。SWAT 工具在服务器上打开了一个安全漏洞，所以我建议不用它并把它删除。

“mkdir”命令在系统中为所有共享打印工作创建一个“/var/spool/samba/”目录，当然这个目录仅当你打算使用 Samba 的局域网共享打印机时才是必须的。既然我们不打算配置 Samba 服务器共享打印机，就不必在服务器上创建这个目录（var/spool/samba/），而且不必用“chmod”命令来改变“var/spool/samba/”的“粘性（sticky）”标志位以确保只有文件的所有者才可以删除那个目录中的文件。

注意：安装过程假定在系统中运行影子（Shadow） passwords/pam（而且也应该这样做），如果按我们的方法安装 Linux，这一定已经预先设置好了。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf samba-version/samba.version.tar.gz
```

rm 命令把用于编译安装 Samba 服务的所有源文件清除，同时也把 Samba 的压缩文件从“/var/tmp”目录中清除出去。

配置

不同服务的配置文件根据需求和网络结构的不同有各自的特殊性，即可以将 Samba 服务器安装成只允许一个客户连接，也可以将其安装成允许 1000 个客户连接。

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 Samba 服务，以下文件必须存在，而且它们被创建在或拷贝到合适的目录之中。

- 拷贝 smb.conf 和 lmhosts 文件到“/etc/”目录
- 拷贝 smb 脚本文件到“/etc/rc.d/init.d/”目录
- 拷贝 samba 文件到“/etc/logrotate.d/”目录
- 拷贝 samba 文件到“/etc/pam.d/”目录

可以从“floppy.tgz”文档中获取以上这些文件，从解压的“floppy.tgz”文档中拷贝这些文件到适当的位置，或者从本书中拷贝并粘贴到相应的文件中。

配置 /etc/smb.conf 文件

“/etc/smb.conf”文件是 Samba 服务器的主要配置文件，它规定了哪些目录可以被 Windows 机器访问，哪些 IP 地址被授权等等，在[global]一节中的行中包含了全局配置信息，它们是对于所有的共享都公用的信息，（除非它们被每共享属性所重载），紧跟其后是[share]一节。

在我们的例子中，我们创建了一个目录[tmp]，且允许一个 IP 地址连接到 Samba 服务器，而且不打算在 Samba 和 Windows 之间使用共享打印机。

编辑“smb.conf”文件(vi /etc/smb.conf)并添加：

```
[global]

workgroup = OPENARCH
server string = Samba %h
encrypt passwords = True
security = user
smb passwd file = /etc/smbpasswd
log file = /var/log/samba/log.%m
socket options = IPTOS_LOWDELAY TCP_NODELAY
domain master = Yes
local master = Yes
preferred master = Yes
os level = 65
dns proxy = No
name resolve order = lmhosts host bcast
bind interfaces only = True
interfaces = eth0 192.168.1.1
hosts deny = ALL
hosts allow = 192.168.1.4 127.0.0.1
debug level = 1
create mask = 0640
directory mask = 0750
level2 oplocks = True
wide links = no
read raw = no
```

```
[homes]
comment = Home Directories
browseable = no
read only = no
invalid users = root bin daemon nobody named sys tty disk mem kmem users

[tmp]
comment = Temporary File Space
path = /tmp
read only = No
valid users = admin
invalid users = root bin daemon nobody named sys tty disk mem kmem users
```

下面逐行说明这些配置

[global]

workgroup = OPENARCH

当客户访问时你的服务器时服务器所显示的组名

server string = Samba %h

计算机显示主机名，“%h”表示你所连接计算机的主机名。

encrypt passwords = True

这项设置将与客户机协调起来通过加密来取代明文密码，监听程序不能发现被加密的密码。出于安全的考虑，这个选项必须设置为 True。

security = user

在用户级的安全级别上，一个连接服务器的客户必须首先用一个有效的用户名和密码，否则该连接将被拒绝，这意味着在 Samba 服务器的“/etc/passwd”文件中必须存在这个客户的一个有效的用户名和密码，否则这个客户的连接将被断开。

smb passwd file = /etc/smbpasswd

这个选项“smb passwd file”设置加密的“smbpasswd”文件的路径，“smbpasswd”文件是“etc/passwd”文件的一个拷贝，它包含连接 Samba 服务器所需的有效的用户名和密码。在有连接请求时，Samba 软件将读取这个文件以确定用户的合法性。

log file = /var/log/samba/log.%m

这个选项“log file”带扩展名“%m”允许不同的用户或客户机在 Samba 服务器上拥有各自的日志文件。

socket options = IPTOS_LOWDELAY TCP_NODELAY

这个选项“socket options”协调局域内的连接，以提高 Samba 服务器的文件传输能力。

domain master = Yes

这个选项“domain master”，指定(nmbd) Samba 服务守护程序作为该工作组的主域浏览器，这个选项对于同一个组内或同一个网络中所有 Samba 服务器，只有一个服务器被设置成 Yes。

local master = Yes

这个选项“local master”允许 (nmbd) Samba 服务守护程序尝试使其成为子网内的局部主浏览器，就象如上所述，通常在同一子网内的 Samba 服务器只有一个 Samba 服务器成为局部主浏览器。

preferred master = Yes

这个选项“preferred master”，在 Samba 服务守护程序成为一个工作组中优先的主浏览器与上面的一样，你只能在同一网络内设置一个 Samba 服务器中指定一个作为优先的浏览器。

os level = 65

这个选项“os level”决定了 Samba 服务守护程序是否有机会成为一个工作组广播的区域内的局域主浏览器，数字 65 将使它优先于任何 NT 服务器，如果在你的局域网中存在 NT 服务器，而且希望 Linux Samba 服务器优先于 NT 成为工作组中广播区域内的主局域浏览器，那么你就必须设置 os level 选项为 65，而且这个选项必须设置在一个 Linux Samba 服务器上，在网络内的其它 Linux Samba 服务器必须禁用此选项。

dns proxy = No

这个选项“dns proxy”，如果设置为 Yes，则指定(nmbd) Samba 服务守护进程为一个 WINS 服务器，并且负责查找未登记的 Net BIOS 名称，把 Net BIOS 名字作为 DNS 名称，并作为一个名字查询客户程序从 DNS 服务器中查询 DNS 名称，我们并不打算把 Samba 服务器配置成 WINS 服务器，我们不必把这的选项设成 Yes。

name resolve order = lmhosts host bcast

这个选项“name resolve order”决定使用哪些名字服务程序及使用它们的优先顺序来解析它们的主机名和 IP 地址。

bind interfaces only = True

这个选项“bind interfaces only”如果设置为 True，允许限制机器上哪个网络界面可以为 Smb 请求提供服务，这是一个安全方面的特性。下面配置“interfaces = eth0 192.168.1.1”将完成这个选项。

interfaces = eth0 192.168.1.1

这个选项“interfaces”允许用户重载 Samba 服务程序浏览用的缺省浏览网络界面、名称登记和其它 NBT 通讯设备。缺省的情况下，Samba 将访问内核中所有激活的网络界面，将利用除了 127.0.0.1 以外的所有广播网络界面，对于这个选项 Samba 将只监听 192.168.1.1 的“eth0”界面，这是一个安全特性，并完成上面的配置(bind interfaces only =True).

hosts deny = ALL

在“hosts deny”选项中所列出的主机将被禁止访问 Samba，除非指定的服务有它们允许主机列表来重载它，缺省情况下我们禁止所有来访主机，允许的主机在下一个选项中指定(hosts allow =)。

hosts allow = 192.168.1.4 127.0.0.1

选项“hosts allow”是一组通过逗号、空格，跳格分隔开的主机 IP 列表，它们被允许访问 Samba 服务器，我们允许 192.168.1.4 和本地机 127.0.0.1 访问 Samba 服务器。注意，localhost 必须设置，否则你会收到某些错误信息。

debug level = 1

选项“debug level”允许在“smb.conf”文件中指定调试级别（日志级别），如果你设置调试级别高于 2，你的应用的性能将大为降低，这是因为服务器每一步操作中都将产生大量的调试信息写入日志文件，它的代价将是很高的。

create mask = 0640

选项“create mask”设置从 DOS 模式映射到 UNIX 权限时缺省权限，当选项设为 0640 时，所有从 Windows 系统中转移到 Unix 系统中的文件缺省情况下具有 0640 的权限。

directory mask = 0750

选项“directory mask”设置一个从 DOS 模式到 Unix 模式创建或拷贝目录时缺省的 8 进制权限，若这个选项设置为 0750，所有从 Windows 系统中创建和拷贝到 Unix 系统中的目录在缺省情况下都具有 0750 的权限。

level2 oplocks = True

选项“level2 oplocks”可提高访问以特别方式写的文件的性能（例如，应用程序，EXE 文件）

wide links = no

选项“wide links”控制在 Unix 文件系统中服务器的链接是否可以访问。在服务器所输出的目录树内的链接总是可访问的，这个选项只是控制输出目录树以外的链接的访问，建议出于安全考虑，建议禁用此选项。

read raw = no

选项“read raw”控制的是 Samba 服务器在向客户机传送数据时是否支持直接读取(read raw)SMB 请求。注意，在“read raw”操作中没有使用内存映射，因此，当象我们这样禁用“read raw”选项，即设置为选项值为“No”时，你会发现使用内存映射程序会更有效。

[tmp]

comment = Temporary File Space

选项“comment”是位于共享名旁边的一个文本区域中的字符串，即当你使用 network 或“net view”请求 Samba 服务器时，服务器所罗列出来共享资源所显示的名称。

path = /tmp

选项“path”指定为用户提供服务的目录。

read only = No

选项“read only”指定是否用户仅被允许以只读方式访问文件。

valid users = admin

选项“valid users”列出能够登录服务的用户列表。

invalid users = root bin daemon nobody named sys tty disk mem kmem users

选项“invalid users”列出禁止登录服务的用户列表，这种近乎“偏执”的检验用以绝对保证不合适的设置会破坏你的系统安全。

配制 /etc/lmhosts 文件

配制“/etc/lmhosts”文件，“lmhosts”文件是提供 Samba Net BIOS 名称到 IP 地址的映射的文件，它与“/etc/hosts”文件格式相似，只是主机名字必须符合 Net BIOS 名称格式。

创建“/etc/lmhosts”文件并添加：

```
# Sample Samba lmhosts file.
#
127.0.0.1       localhost
192.168.1.1         deep
192.168.1.4         win
```

在我们的例子里文件中包含三个 IP 到 Net BIOS 名字映射，localhost（127.0.0.1）、一个名为 deep 的（192.168.1.1）客户机，一个叫 win 的（192.168.1.4）的客户机。

配制“/etc/rc.d/init.d/smb”脚本文件

配制“/etc/rc.d/init.d/smb”脚本文件来起动和停止 Samba 服务器 smbd 和 nmbd 守护进程

创建 smb 脚本文件（touch /etc/rc.d/init.d/smb），并添加：

```
#!/bin/sh
#
# chkconfig: - 91 35
# description: Starts and stops the Samba smbd and nmbd daemons \
#          used to provide SMB network services.

# Source function library.
. /etc/rc.d/init.d/functions

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

# Check that smb.conf exists.
[ -f /etc/smb.conf ] || exit 0
```

```
RETVAL=0
# See how we were called.
case "$1" in
start)
echo -n "Starting SMB services: "
daemon smbd -D
RETVAL=$?
echo
echo -n "Starting NMB services: "
daemon nmbd -D
RETVAL2=$?
echo
[ $RETVAL -eq 0 -a $RETVAL2 -eq 0 ] && touch /var/lock/subsys/smb || \
RETVAL=1
;;
stop)
echo -n "Shutting down SMB services: "
killproc smbd
RETVAL=$?
echo
echo -n "Shutting down NMB services: "
killproc nmbd
RETVAL2=$?
[ $RETVAL -eq 0 -a $RETVAL2 -eq 0 ] && rm -f /var/lock/subsys/smb
echo ""
;;
restart)
$0 stop
$0 start
RETVAL=$?
;;
reload)
echo -n "Reloading smb.conf file: "
killproc -HUP smbd
RETVAL=$?
echo
;;
status)
status smbd
status nmbd
RETVAL=$?
;;
*)
echo "Usage: $0 {start|stop|restart|status}"
```

```
exit 1
esac

exit $RETVAL
```

现在使脚本文件执行并改变其缺省的权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/smb
```

用以下的命令来创建 Samba 到 rc.d 的符号链接

```
[root@deep]# chkconfig --add smb
```

Samba 脚本在下次重启服务器时并不会自动启动 smbd、nmbd 守护进程（daemon）。

可以通过执行下面的命令来更改缺省设置：

```
[root@deep]# chkconfig --level 345 smb on
```

用下面的命令来手工起动 Samba 服务器

```
[root@deep]# /etc/rc.d/init.d/smb start
```

配置 /etc/pam.d/samba 文件

配置“/etc/pam.d/samba”文件使其能采用 pam 认证

创建“Samba”文件（touch /etc/pam.d/samba），并添加：

```
Auth required /lib/security/pam_pwdb.so nullok shadow
Account required /lib/security/pam_pwdb.so
```

配置 /etc/logrotate.d/samba 文件

配置“/etc/logrotate.d/samba”文件使日志文件每周自动更新一次。

创建“Samba”文件（touch /etc/logrotate.d/samba），并添加：

```
/var/log/samba/log.nmb {
notifempty
missingok
postrotate
/usr/bin/killall -HUP nmbd
endrotate
}

/var/log/samba/log.smb {
notifempty
missingok
postrotate
/usr/bin/killall -HUP smbd
endrotate
}
```

Samba 的安全性

创建一个加密的 password 文件

“/etc/smbpasswd”文件是 Samba 加密的密码文件，它包括用户名，Unix 用户代号、SMB 控制的用户密码、用户标志信息和密码最后更改日期。

重要信息：创建一个 Samba 用户，必须首先拥有一个有效的 Linux 帐号，用如下的命令从“/etc/passwd”文件来创建 smbpasswd 文件：

```
[root@deep]# cat /etc/passwd | mksmbpasswd.sh > /etc/smbpasswd
```

创建用户帐号：

```
[root@deep]# smbpasswd -a username (remember that “username” must be a valid Linux account).


New SMB password:
Retype new SMB password:
Password changed for user username.
```

```
[root@deep]# chmod 600 /etc/smbpasswd
[root@deep]# testparm (this will verify the smb.conf file for error).
```

注意：详细信息参见“samba/doc/texts/”中的 ENCRYPTION.txt。

保证重要配置文件的安全

设置不可更改标志位可以预防意外删除或覆盖受保护的文件，它也可以预防别人创建到它的符号链接，一旦配置好“smb.conf”和“lmhosts”文件，用下面的命令把它们免疫起来是个不错的主意：

```
[root@deep]# chattr +i /etc/smb.conf
[root@deep]# chattr +i /etc/lmhosts
```

更多的资料

获取更详细的信息，可以阅读以下 man 帮助：

```
$ man Samba (7) - A Windows SMB/CIFS fileserver for UNIX
$ man smb.conf (5) - The configuration file for the Samba suite
$ man smbclient (1) - ftp-like client to access SMB/CIFS resources on servers
$ man smbd (8) - server to provide SMB/CIFS services to clients
$ man smbmnt (8) - mount smb file system
$ man smbmount (8) - mount smb file system
$ man smbpasswd (5) - The Samba encrypted password file
$ man smbpasswd (8) - change a users SMB password
$ man smbrun (1) - interface program between smbd and external programs
$ man smbsh (1) - Allows access to Windows NT filesystem using UNIX commands
$ man smbstatus (1) - report on current Samba connections
$ man smbtar (1) - shell script for backing up SMB shares directly to UNIX tape
drives
$ man smbumount (8) - umount for normal users
$ man testparm (1) - check an smb.conf configuration file for internal correctness
$ man testprns (1) - check printer name for validity with smbd
```

Samba 的管理工具

下面列出一些很常用的命令，还有更多的命令我们没有介绍，必须查阅 man 帮助文件以获取详细信息。

smbstatus

smbstatus 是一个能够列出当前 Samba 连接的简单程序。

用以下命令显示当前 Samba 连接：

```
[root@deep]# smbstatus

Samba version 2.0.6
Service     uid gid pid machine
----------------------------------------------
tmp     webmaster webmaster 3995 gate (192.168.1.3) Sat Sep 25 19:40:54 1999

No locked files

Share mode memory usage (bytes):
1048464(99%) free + 56(0%) used + 56(0%) overhead = 1048576(100%) total
```

Samba 的用户工具

下面列出一些很常用的命令，还有更多的命令我们没有介绍，你必须查阅 man 帮助文件以获取详细信息。

smbclient

smbclient 是一种客户端和 SMB/CIFS 服务器“交谈”的工具。它提供了一种类似 FTP 的用户界面。它所提供的操作包括从服务器上下载文件、从客户机上上传文件和从服务器上获得目录信息等等操作。

用以下命令连接 WINDOWS 机器：

```
[root@deep]# smbclient //sbmserver/sharename -U username
[root@deep]# smbclient //gate/tmp -U webmaster

Password:
Domain=[ OPENARCH] OS=[ Windows NT 4.0] Server=[ NT LAN Manager 4.0]
Smb: \>
```

其中“//sbmserver”是你所连接的服务器的名称。“/sharename”是服务器上共享目录的名称，“-U”表示这台机器的一个用户名。

安装到系统中的文件

```
> /etc/rc.d/init.d/smb
> /etc/rc.d/rc0.d/K35smb
> /etc/rc.d/rc1.d/K35smb
> /etc/rc.d/rc2.d/K35smb
> /etc/rc.d/rc3.d/K35smb
> /etc/rc.d/rc4.d/K35smb
> /etc/rc.d/rc5.d/K35smb
> /etc/rc.d/rc6.d/K35smb
> /etc/codepages
> /etc/lmhosts
> /etc/pam.d/samba
> /etc/smb.conf
> /etc/MACHINE.SID
> /etc/logrotate.d/samba
> /etc/smbpasswd
> /usr/bin/smbclient
> /usr/bin/smbspool
> /usr/bin/testparm
> /usr/bin/testprns
> /usr/bin/smbstatus
> /usr/bin/rpcclient
> /usr/bin/smbpasswd
> /usr/bin/make_smbcodepage
> /usr/bin/nmblookup
> /usr/bin/make_printerdef
> /usr/bin/smbtar
> /usr/bin/addtosmbpass
> /usr/bin/convert_smbpasswd
> /usr/bin/mksmbpasswd.sh
> /usr/man/man1/nmblookup.1
> /usr/man/man1/make_smbcodepage.1
> /usr/man/man1/smbclient.1
> /usr/man/man1/smbrun.1
> /usr/man/man1/smbsh.1
> /usr/man/man1/smbstatus.1
> /usr/man/man1/smbtar.1
> /usr/man/man1/testparm.1
> /usr/man/man1/testprns.1
```

```
> /usr/man/man5/lmhosts.5
> /usr/man/man5/smb.conf.5
> /usr/man/man5/smbpasswd.5
> /usr/man/man7/samba.7
> /usr/man/man8/nmbd.8
> /usr/man/man8/smbd.8
> /usr/man/man8/smbmnt.8
> /usr/man/man8/smbmount.8
> /usr/man/man8/smbpasswd.8
> /usr/man/man8/smbspool.8
> /usr/man/man8/smbumount.8
> /usr/sbin/smbd
> /usr/sbin/nmbd
> /var/log/samba
> /var/lock/samba
```

Linux OpenLDAP 服务器

概述

LDAP（Lightweight Directory Access Protocol）是用来访问信息服务的标准协议。这个协议是用在 Internet 传输协议上，如：TCP，能够访问单独的目录服务或 X.500 目录。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

OpenLDAP 的版本号是 1_2_8。

软件包的来源

OpenLDAP 的主页：http://www.openldap.org/。

下载：openldap-1_2_8.tgz。

安装软件包需要注意的问题

最好在编译前和编译后都做一张系统中所有文件的列表，然后用“diff”命令去比较它们，找出其中的差别并知道到底把软件安装在哪里。只要简单地在编译之前运行一下命令“find /* >ldap1”，在编译和安装完软件之后运行命令“find /* > ldap2”，最后用命令“diff ldap1 ldap2 > ldap”找出变化。

编译

把软件包（tar.gz）解压缩：

```
[root@deep]# cp openldap-version.tgz /var/tmp
[root@deep]# cd /var/tmp/
[root@deep]# tar xzpf openldap-version.tgz
```

编译和优化

转到 OpenLDAP 的新目录下。

编辑“string.h”文件（vi +52 include/ac/string.h），删掉这些几行：

```
#else
/* some systems have strdup(), but fail to declare it */
extern char *(strdup)();
```

上面的这几行对 Linux 系统是没有用的，必须删掉。

下面设置编译器的编译参数：

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-
pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--libexecdir=/usr/sbin \
--localstatedir=/var/run \
--sysconfdir=/etc \
--enable-shared \
--with-gnu-ld
```

这些编译参数告诉编译器如何编译 OpenLDAP：

- 编译共享库
- 假定 C 编译器使用 GNU ld

开始编译：

```
[root@deep]# make depend
[root@deep]# make
[root@deep]# cd tests/
[root@deep]# make
[root@deep]# cd ..
[root@deep]# make install
```

“make depend”命令编译必要的相关文件，“make”把源文件编译成可执行的二进制文件。“make install”把二进制文件和相关的配置文件安装到相应的目录中去。在“test”目录下的“make”命令在安装 LDAP 服务器之前进行一些必要的测试保证软件安装完成后能正常运行。如何测试失败在继续安装之前先要解决这些问题。

```
[root@deep]# install -d -m 700 /var/ldap
[root@deep]# echo localhost > /etc/openldap/ldapserver
[root@deep]# strip /usr/lib/liblber.so.1.0.0
[root@deep]# strip /usr/lib/libldap.so.1.0.0
[root@deep]# strip /usr/lib/libldap.a
[root@deep]# strip /usr/lib/liblber.a
[root@deep]# strip /usr/sbin/in.xfingerd
[root@deep]# strip /usr/sbin/go500
[root@deep]# strip /usr/sbin/go500gw
[root@deep]# strip /usr/sbin/mail500
[root@deep]# strip /usr/sbin/rp500
[root@deep]# strip /usr/sbin/fax500
[root@deep]# strip /usr/sbin/rcpt500
[root@deep]# strip /usr/sbin/slapd
[root@deep]# strip /usr/sbin/ldif2ldbm
[root@deep]# strip /usr/sbin/ldif2index
[root@deep]# strip /usr/sbin/ldif2id2entry
[root@deep]# strip /usr/sbin/ldif2id2children
[root@deep]# strip /usr/sbin/ldbmcat
[root@deep]# strip /usr/sbin/ldif
[root@deep]# strip /usr/sbin/centipede
[root@deep]# strip /usr/sbin/ldbmtest
[root@deep]# strip /usr/sbin/slurpd
[root@deep]# strip /usr/bin/ud
[root@deep]# strip /usr/bin/ldapadd
[root@deep]# strip /usr/bin/ldapsearch
[root@deep]# strip /usr/bin/ldapmodify
[root@deep]# strip /usr/bin/ldapmodrdn
[root@deep]# strip /usr/bin/ldappasswd
[root@deep]# strip /usr/bin/ldapdelete
```

上面的“install”命令在“/var”目录下创建一个名为“ldap”的新目录并且把它的属性设成只能被超级用户“root”读、写和执行。

“strip”命令清除目标文件中所有的符号信息，这样可执行文件就会更小，能够提高一点程序的性能。

清除不必要的文件

用下面的命令删除不必要的文件：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf ldap openldap-version.tgz
```

“rm”命令删除所有编译和安装 OpenLDAP 所需要的源程序，并且把“/var/tmp”目录下的 OpenLDAP 软件的压缩包删除掉。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 OpenLDAP，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“slapd.conf”文件拷贝到“/etc/openldap”目录下
- 把“ldap”文件拷贝到“/etc/rc.d/init.d”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到“/etc/ssh”目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

配置 /etc/ldap/slapd.conf 文件

“/etc/openldap/slapd.conf”文件是 LDAP 服务器最重要的配置文件，配置权限、口令、数据库类型和数据库位置等等。

编辑“slap.conf”文件（vi /etc/openldap/slapd.conf）加入：

```
#
# See slapd.conf(5) for details on configuration options.
```

```
# This file should NOT be world readable.
#
include /etc/openldap/slapd.at.conf
include /etc/openldap/slapd.oc.conf
schemacheck off
#referral ldap://ldap.itd.umich.edu
pidfile /var/run/slapd.pid
argsfile /var/run/slapd.args
########################################################################
# ldbm database definitions
########################################################################
database ldbm
suffix "o=openarch, c=com"
directory /var/ldap
rootdn "cn=admin, o=openarch, c=com"
rootpw secret
# cleartext passwords, especially for the rootdn, should
# be avoid. See slapd.conf(5) for details.
# ldbm indexed attribute definitions
index cn,sn,uid
index objectclass pres,eq
index default none
# ldbm access control definitions
defaultaccess read
access to attr=userpassword
by self write
by dn="cn=admin, o=openarch, c=com" write
by * compare
```

下面解释一下上面配置文件中的一些设置：

suffix “o=openarch, c=com”

suffix（后缀）设置想要创建的子树的根的 DN（distinguished name）。换句话说，就是表示数据库中到底放些什么。

directory /var/ldap

设置 LDAP 数据库和索引文件所在的目录。必须设置为“/var/ldap”因为我们在安装阶段已经指定了 LDAP 的后台数据库。

rootdn “cn=admin, o=openarch, c=com”

设置管理 LDAP 目录的超级用户的 DN。这个用户名（admin）并不要出现在“/etc/passwd”文件里。

rootpw secret

设置这个数据库的超级用户的口令验证方式。也就是上面“rootdn”设置的用户的口令。千万不要用明文进行口令验证，一定要用加密的口令。

index cn,sn,uid | index objectclass pres,eq | index default none

这三句设置在数据库中创建和维护怎样的索引。“index cn,sn,uid）为 cn、sn 和 uid 属性创建和维护索引，“index objectclass pres,eq”为 objectclass 属性创建相等索引（equality indexes），“index default none”表示不为别的属性创建索引。查看用户手册可以得到更多的信息。

“slapd.conf”文件中的最后一部分是和 LDAP 目录的访问控制相关的。

```
defaultaccess read
access to attr=userpassword
by self write
by dn="cn=admin, o=openarch, c=com" write
by * compare
```

这些设置对“o=openarch，c=com”子树中的所有项都是有效的。任何人都有读的权限，除了“userpasswd”属性拥有这一项的用户可以改变其它所有的属性。“userpassword”属性只能被 admin 修改，但是任何人都可以比较。查看用户手册可以得到更多的信息。

配置 /etc/rc.d/init.d/ldap 脚本文件

配置“/etc/rc.d/init.d/ldap”脚本文件用来启动和停止 ldap 服务。

创建“ldap”脚本文件（touch /etc/rc.d/init.d/ldap）并加入：

```
#!/bin/sh
#
# ldap This shell script takes care of starting and stopping
# ldap servers (slapd and slurpd).
#
# chkconfig: - 70 40
# description: LDAP stands for Lightweight Directory Access Protocol, used \
# for implementing the industry standard directory services.
# processname: slapd
# config: /etc/openldap/slapd.conf
# pidfile: /var/run/slapd.pid
# Source function library.
. /etc/rc.d/init.d/functions
# Source networking configuration.
```

```
. /etc/sysconfig/network
# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0
[ -f /usr/sbin/slapd ] || exit 0
[ -f /usr/sbin/slurpd ] || exit 0
RETVAL=0
# See how we were called.
case "$1" in
start)
# Start daemons.
echo -n "Starting ldap: "
daemon slapd
RETVAL=$?
if [ $RETVAL -eq 0 ]; then
if grep -q "^replogfile" /etc/openldap/slapd.conf; then
daemon slurpd
RETVAL=$?
[ $RETVAL -eq 0 ] && pidof slurpd | cut -f 1 -d " " > /var/run/slurpd
fi
fi
echo
[ $RETVAL -eq 0 ] && touch /var/lock/subsys/ldap
;;
stop)
# Stop daemons.
echo -n "Shutting down ldap: "
killproc slapd
RETVAL=$?
if [ $RETVAL -eq 0 ]; then
if grep -q "^replogfile" /etc/openldap/slapd.conf; then
killproc slurpd
RETVAL=$?
fi
fi
echo
if [ $RETVAL -eq 0 ]; then
rm -f /var/lock/subsys/ldap
rm -f /var/run/slapd.args
fi
;;
status)
status slapd
RETVAL=$?
if [ $RETVAL -eq 0 ]; then
```

```
if grep -q "^replogfile" /etc/openldap/slapd.conf; then
status slurpd
RETVAL=$?
fi
fi
;;
restart)
$0 stop
$0 start
RETVAL=$?
;;
reload)
killproc -HUP slapd
RETVAL=$?
if [ $RETVAL -eq 0 ]; then
if grep -q "^replogfile" /etc/openldap/slapd.conf; then
killproc -HUP slurpd
RETVAL=$?
fi
fi
;;
*)
echo "Usage: $0 start|stop|restart|status}"
exit 1
esac
exit $RETVAL
```

使得脚本可执行并改变它的权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/ldap
```

在 rc.d 目录中创建 OpenLDAP 脚本的符号链接：

```
[root@deep]# chkconfig --add ldap
```

当重新引导系统的时候，OpenLDAP 脚本不会自动启动 slapd 脚本。可以用下面的命令让它可以自动启动：

```
[root@deep]# chkconfig --level 345 ldap on
```

用下面的命令也可以手工启动 OpenLDAP 服务器：

```
[root@deep]# /etc/rc.d/init.d/ldap start
```

更多的资料

如果想查找详细的资料可以用 man 命令查帮助页，读取相关信息：

```
$ man ldapd (8) - LDAP X.500 Protocol Daemon
$ man ldapdelete (1) - ldap delete entry tool
$ man ldapfilter.conf (5) - configuration file for LDAP get filter routines
$ man ldapfriendly (5) - data file for LDAP friendly routines
$ man ldapmodify, ldapadd (1) - ldap modify entry and ldap add entry tools
$ man ldapmodrdn (1) - ldap modify entry RDN tool
$ man ldappasswd (1) - change the password of an LDAP entry
$ man ldapsearch (1) - ldap search tool
$ man ldapsearchprefs.conf (5) - configuration file for LDAP search preference
routines
$ man ldaptemplates.conf (5) - configuration file for LDAP display template
routines
$ man ldif (5) - LDAP Data Interchange Format
$ man slapd (8) - Stand-alone LDAP Daemon
$ man slapd.conf (5) - configuration file for slapd, the stand-alone LDAP daemon
$ man slurpd (8) - Standalone LDAP Update Replication Daemon
$ man ud (1) - interactive LDAP Directory Server query program
```

保证 OpenLDAP 的安全

使得重要的配置文件不可改变

设置不可改变位可以避免需要保护的文件被意外的删除或者覆盖。也可以防止有人创建这个文件的符号链接。一旦“slapd.conf”文件配置完成了之后，用下面的命令设置不可改变位：

```
[root@deep]# chattr +i /etc/openldap/slapd.conf
```

OpenLDAP 的创建和维护工具

离线创建数据库

如果一下子要创建成千上万的项(entry)，用这个方法最合适了，因为用“ldapadd”命令逐个添加要花费相当长的时间。这个工具读取 slapd 的配置文件和用文本表示的包含所有需要添加的项的输入文件。“ldif2ldbm”的命令语法如下：

```
[root@deep]# ldif2ldbm -i <inputfile> -f <slapdconfigfile>
```

<input>表示纯文本的 LDIF 输入文件的文件名。<slapdconfigfile>表示 slapd 配置文件的文件名，这个配置文件设置在什么地方创建索引、创建什么索引，等等。

例如：

```
[root@deep]# ldif2ldbm -i my-data-file -f /etc/openldap/slapd.conf
```

纯文本的输入文件的文件名为“my-data-file”

注意：离线创建数据库的时候 slapd daemon 不能运行。

IDIF 输入文件　用文本表示的输入文件

第一次安装 OpenLDAP 必须把大量的数据存入 OpenLDAP 的数据库里。最好把这些数据都用纯文本的文件表示出来，再用下面的命令把数据加入 OpenLDAP 数据库中。

创建“my-data-file”文件（touch /tmp/my-data-file），下面加入的这些行可以作为参考：

```
dn: o=openarch, c=com
o: openarch
objectclass: organization
dn: cn=Ronald Smith, o=openarch, c=com
cn: Ronald Smith
sn: Smith
telephonenumber: (480) 757-5856
title: Operator.
objectclass: top

objectclass: person
dn: cn=Anthony Bay, o=openarch, c=com
cn: Anthony Bay
sn: Bay
homephone: (410) 896-3786
mobile: (410) 833-0590
mail: abay@openarch.com
objectclass: top
objectclass: person
dn: cn=George Parker, o=openarch, c=com
cn: George Parker
sn: Parker
telephonenumber: (414) 389-5695
fax: (414) 778-8785
mobile: (414) 470-8669
```

```
description: E-Commerce.
objectclass: top
objectclass: person
```

上面的例子文件让你了解了如何把数据库中的信息装成纯文本的信息，再把这些信息加入 OpenLDAP 数据库中。还有很多的设置选项可以满足你的需要，请查看 OpenLDAP 的文档或书籍，以获得更多的信息。

为 LDAP 创建数据库

就像可以往数据库中加入记录那样，也可以用“ldapadd”命令在 LDAP 中加入项。例如：用“ldapadd”命令加入“Europe Mourani”这一项，先要创建“/tmp/newentry”文件：

创建“newentry”文件（touch /tmp/newentry），在文件中加入下面的内容：

```
cn=Europe Mourani, o=openarch, c=com
cn=Europe Mourani
sn=Mourani
mail=emourani@old.com
description=Marketing relation.
objectClass=top
objectClass=person
```

用下面的命令在 LDAP 中加入这一项：

```
[root@deep]# ldapadd -f /tmp/newentry -D "cn=admin, o=openarch, c=com" -W


Enter LDAP Password :
```

上面的命令假定在配置文件中设置“rootdn”为“cn=admin, o=openarch, c=com”，“rootpw”设置为“secret”。命令还会提示输入口令。

ldapmodify

“ldapmodify”命令建立到 LDAP 服务器的连接，绑定、修改和加入数据项。通过使用“-f”参数可以设置从标准输入还是文件中得到数据项的信息。

用于“ldapmodify”命令的输入文件的格式：

假定“/tmp/entry”文件存在，而且文件的内容是：

```
cn=Europe Mourani, o=openarch, c=com
- mail=emourani@old.com # will delete the old mail address for Europe Mourani in the database.
+mail=emourani@new.com # will add the new mail address for Europe Mourani in the database.
```

用下面的命令改变 LDAP 中的数据项：

```
[root@deep]# ladpmodify -D 'cn=Admin, o=openarch, c=com' -W -f <inputfile>
```

这个命令改变了“Europe Mourani”数据项的 mail 属性，把这个属性值改为：emourani@new.com。

OpenLDAP 的用户工具

下面列出的是一些我们经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

搜索 LDAP 的数据项

“ldapsearch”命令建立到 LDAP 服务器的连接、绑定并用过滤器进行搜索。

用下面的命令搜索 LDAP 数据库中的数据项：

```
[root@deep]# ldapsearch -b <dn> <attrs>
[root@deep]# ldapsearch -b 'o=openarch.com' 'cn=a*'
```

这个命令把所有以字母“a”开头的数据项的信息显示在道标准输出上。

安装到系统中的文件

```
> /etc/openldap
> /etc/openldap/ldap.conf
> /etc/openldap/ldap.conf.default
> /etc/openldap/ldapfilter.conf
> /etc/openldap/ldapfilter.conf.default
> /etc/openldap/ldaptemplates.conf
> /etc/openldap/ldaptemplates.conf.default
> /etc/openldap/ldapsearchprefs.conf
> /etc/openldap/ldapsearchprefs.conf.default
```

```
> /etc/openldap/slapd.conf
> /etc/openldap/slapd.conf.default
> /etc/openldap/slapd.at.conf
> /etc/openldap/slapd.at.conf.default
> /etc/openldap/slapd.oc.conf
> /etc/openldap/slapd.oc.conf.default
> /etc/openldap/ldapserver
> /etc/rc.d/init.d/ldap
> /etc/rc.d/rc0.d/K40ldap
> /etc/rc.d/rc1.d/K40ldap
> /etc/rc.d/rc2.d/K40ldap
> /etc/rc.d/rc3.d/S70ldap
> /etc/rc.d/rc4.d/S70ldap
> /etc/rc.d/rc5.d/S70ldap
> /etc/rc.d/rc6.d/K40ldap
> /usr/bin/ud
> /usr/bin/ldapsearch
> /usr/bin/ldapmodify
> /usr/bin/ldapdelete
> /usr/bin/ldapmodrdn
> /usr/bin/ldappasswd
> /usr/bin/ldapadd
> /usr/include/ldap.h
> /usr/include/lber.h
> /usr/include/ldap_cdefs.h
> /usr/man/man3/ldap_open.3
> /usr/man/man3/ldap_errlist.3
> /usr/man/man3/ldap_err2string.3
> /usr/man/man3/ldap_first_attribute.3
> /usr/man/man3/ldap_next_attribute.3
> /usr/man/man3/ldap_first_entry.3
> /usr/man/man3/ldap_next_entry.3
> /usr/man/man3/ldap_count_entries.3
> /usr/man/man3/ldap_friendly.3
> /usr/man/man3/ldap_friendly_name.3
> /usr/man/man3/ldap_free_friendlymap.3
> /usr/man/man3/ldap_get_dn.3
> /usr/man/man3/ldap_explode_dn.3
> /usr/man/man3/ldap_explode_dns.3
> /usr/man/man3/ldap_dn2ufn.3
> /usr/man/man3/ldap_is_dns_dn.3
> /usr/man/man3/ldap_get_values.3
> /usr/man/man3/ldap_get_values_len.3
> /usr/man/man3/ldap_value_free.3
```

```
> /usr/man/man3/ldap_value_free_len.3
> /usr/man/man3/ldap_count_values.3
> /usr/man/man3/ldap_count_values_len.3
> /usr/man/man3/ldap_getfilter.3
> /usr/man/man3/ldap_init_getfilter.3
> /usr/man/man3/ldap_init_getfilter_buf.3
> /usr/man/man3/ldap_getfilter_free.3
> /usr/man/man3/ldap_getfirstfilter.3
> /usr/man/man3/ldap_getnextfilter.3
> /usr/man/man3/ldap_setfilteraffixes.3
> /usr/man/man3/ldap_build_filter.3
> /usr/man/man3/ldap_modify.3
> /usr/man/man3/ldap_modify_s.3
> /usr/man/man3/ldap_mods_free.3
> /usr/man/man3/ldap_modrdn.3
> /usr/include/disptmpl.h
> /usr/include/srchpref.h
> /usr/lib/liblber.so.1.0.0
> /usr/lib/liblber.so.1
> /usr/lib/liblber.so
> /usr/lib/liblber.la
> /usr/lib/liblber.a
> /usr/lib/libldap.so.1.0.0
> /usr/lib/libldap.so.1
> /usr/lib/libldap.so
> /usr/lib/libldap.la
> /usr/lib/libldap.a
> /usr/man/man1/ud.1
> /usr/man/man1/ldapdelete.1
> /usr/man/man1/ldapmodify.1
> /usr/man/man1/ldapadd.1
> /usr/man/man1/ldapmodrdn.1
> /usr/man/man1/ldappasswd.1
> /usr/man/man1/ldapsearch.1
> /usr/man/man3/cldap_close.3
> /usr/man/man3/cldap_open.3
> /usr/man/man3/cldap_search_s.3
> /usr/man/man3/cldap_setretryinfo.3
> /usr/man/man3/lber-decode.3
> /usr/man/man3/lber-encode.3
> /usr/man/man3/ldap.3
> /usr/man/man3/cldap.3
> /usr/man/man3/ldap_abandon.3
> /usr/man/man3/ldap_add.3
```

```
> /usr/man/man3/ldap_add_s.3
> /usr/man/man3/ldap_bind.3
> /usr/man/man3/ldap_bind_s.3
> /usr/man/man3/ldap_simple_bind.3
> /usr/man/man3/ldap_simple_bind_s.3
> /usr/man/man3/ldap_kerberos_bind_s.3
> /usr/man/man3/ldap_kerberos_bind1.3
> /usr/man/man3/ldap_kerberos_bind1_s.3
> /usr/man/man3/ldap_kerberos_bind2.3
> /usr/man/man3/ldap_kerberos_bind2_s.3
> /usr/man/man3/ldap_unbind.3
> /usr/man/man3/ldap_unbind_s.3
> /usr/man/man3/ldap_set_rebind_proc.3
> /usr/man/man3/ldap_cache.3
> /usr/man/man3/ldap_enable_cache.3
> /usr/man/man3/ldap_disable_cache.3
> /usr/man/man3/ldap_destroy_cache.3
> /usr/man/man3/ldap_flush_cache.3
> /usr/man/man3/ldap_uncache_entry.3
> /usr/man/man3/ldap_uncache_request.3
> /usr/man/man3/ldap_set_cache_options.3
> /usr/man/man3/ldap_charset.3
> /usr/man/man3/ldap_set_string_translators.3
> /usr/man/man3/ldap_enable_translation.3
> /usr/man/man3/ldap_translate_from_t61.3
> /usr/man/man3/ldap_translate_to_t61.3
> /usr/man/man3/ldap_t61_to_8859.3
> /usr/man/man3/ldap_8859_to_t61.3
> /usr/man/man3/ldap_compare.3
> /usr/man/man3/ldap_compare_s.3
> /usr/man/man3/ldap_delete.3
> /usr/man/man3/ldap_delete_s.3
> /usr/man/man3/ldap_disptmpl.3
> /usr/man/man3/ldap_modrdn_s.3
> /usr/man/man3/ldap_modrdn2.3
> /usr/man/man3/ldap_modrdn2_s.3
> /usr/man/man3/ldap_init.3
> /usr/man/man3/ldap_result.3
> /usr/man/man3/ldap_msgfree.3
> /usr/man/man3/ldap_search.3
> /usr/man/man3/ldap_search_s.3
> /usr/man/man3/ldap_search_st.3
> /usr/man/man3/ldap_searchprefs.3
> /usr/man/man3/ldap_init_searchprefs.3
```

```
> /usr/man/man3/ldap_init_searchprefs_buf.3
> /usr/man/man3/ldap_free_searchprefs.3
> /usr/man/man3/ldap_first_searchobj.3
> /usr/man/man3/ldap_next_searchobj.3
> /usr/man/man3/ldap_sort.3
> /usr/man/man3/ldap_sort_entries.3
> /usr/man/man3/ldap_sort_values.3
> /usr/man/man3/ldap_sort_strcasecmp.3
> /usr/man/man3/ldap_ufn.3
> /usr/man/man3/ldap_ufn_search_s.3
> /usr/man/man3/ldap_ufn_search_c.3
> /usr/man/man3/ldap_ufn_search_ct.3
> /usr/man/man3/ldap_ufn_setprefix.3
> /usr/man/man3/ldap_ufn_setfilter.3
> /usr/man/man3/ldap_ufn_timeout.3
> /usr/man/man3/ldap_url.3
> /usr/man/man3/ldap_is_ldap_url.3
> /usr/man/man3/ldap_url_parse.3
> /usr/man/man3/ldap_free_urldesc.3
> /usr/man/man3/ldap_url_search.3
> /usr/man/man3/ldap_url_search_s.3
> /usr/man/man3/ldap_url_search_st.3
> /usr/man/man5/ldap.conf.5
> /usr/man/man5/ldapfilter.conf.5
> /usr/man/man5/ldapfriendly.5
> /usr/man/man5/ldapsearchprefs.conf.5
> /usr/man/man5/ldaptemplates.conf.5
> /usr/man/man5/ldif.5
> /usr/man/man5/slapd.conf.5
> /usr/man/man5/slapd.replog.5
> /usr/man/man5/ud.conf.5
> /usr/man/man8/centipede.8
> /usr/man/man8/chlog2replog.8
> /usr/man/man8/edb2ldif.8
> /usr/man/man8/go500.8
> /usr/man/man8/go500gw.8
> /usr/man/man8/in.xfingerd.8
> /usr/man/man8/ldapd.8
> /usr/man/man8/ldbmcat.8
> /usr/man/man8/ldif.8
> /usr/man/man8/ldif2ldbm.8
> /usr/man/man8/ldif2index.8
> /usr/man/man8/ldif2id2entry.8
> /usr/man/man8/ldif2id2children.8
```

```
> /usr/man/man8/mail500.8
> /usr/man/man8/fax500.8
> /usr/man/man8/rcpt500.8
> /usr/man/man8/slapd.8
> /usr/man/man8/slurpd.8
> /usr/sbin/ldif
> /usr/sbin/in.xfingerd
> /usr/man/man3/ldap_init_templates.3
> /usr/man/man3/ldap_init_templates_buf.3
> /usr/man/man3/ldap_free_templates.3
> /usr/man/man3/ldap_first_disptmpl.3
> /usr/man/man3/ldap_next_disptmpl.3
> /usr/man/man3/ldap_oc2template.3
> /usr/man/man3/ldap_tmplattrs.3
> /usr/man/man3/ldap_first_tmplrow.3
> /usr/man/man3/ldap_next_tmplrow.3
> /usr/man/man3/ldap_first_tmplcol.3
> /usr/man/man3/ldap_next_tmplcol.3
> /usr/man/man3/ldap_entry2text.3
> /usr/man/man3/ldap_entry2text_search.3
> /usr/man/man3/ldap_vals2text.3
> /usr/man/man3/ldap_entry2html.3
> /usr/man/man3/ldap_entry2html_search.3
> /usr/man/man3/ldap_vals2html.3
> /usr/man/man3/ldap_error.3
> /usr/man/man3/ldap_perror.3
> /usr/man/man3/ld_errno.3
> /usr/man/man3/ldap_result2error.3
> /usr/sbin/go500
> /usr/sbin/go500gw
> /usr/sbin/mail500
> /usr/sbin/rp500
> /usr/sbin/fax500
> /usr/sbin/xrpcomp
> /usr/sbin/rcpt500
> /usr/sbin/slapd
> /usr/sbin/ldif2ldbm
> /usr/sbin/ldif2index
> /usr/sbin/ldif2id2entry
> /usr/sbin/ldif2id2children
> /usr/sbin/ldbmcat
> /usr/sbin/centipede
> /usr/sbin/ldbmtest
> /usr/sbin/slurpd
```

```
> /usr/share/openldap
> /usr/share/openldap/ldapfriendly
> /usr/share/openldap/go500gw.help
> /usr/share/openldap/rcpt500.help
> /var/ldap
```

Linux PostgreSQL Database 服务器

概述

PostgreSQL 最早是由 UC Berkley 大学计算机系开发的，它的许多先进的“对象－关系”概念现在已经在一些商业数据库里得到应用。PostgreSQL 支持 SQL92/SQL3，事务完整性和可扩展性。它现在是一个源于 Berkley 代码并公开源代码的数据库。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

PostgreSQL 的版本是 6_5_3。

而且一定要先安装 egcs-c++-1.1.2-24.i386.rpm 软件包。

软件包的来源

PostgreSQL 的主页：http://www.postgresql.org/。

必须确保下载：postgresql-6_5_3_tar.gz。

安装软件包需要注意的问题

在安装 PostgreSQL 前后保存一下文件列表对你也许是一个好主意，而后用 diff 比较一下两个文件列表从而找出 PostgreSQL 的文件被安装到哪里去了，方法是在安装 PostgreSQL 之前运行一下“find /* > sql1”，而在安装 PostgreSQL 服务之后运行“find /* > sql2”，接着执行命令“diff sql1 sql2 > sql”从而得到安装文件列表。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp postgresql-version_tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf postgresql-version_tar.gz
```

编译和优化

第一步：

首先，创建 postgres 这个数据库超级用户(你也可以用别的名字，不过通常就用这个)。命令为：

```
[root@deep]# useradd -M -o -r -d /var/lib/pgsql -s /bin/bash -c "PostgreSQL Server"
-u 40
postgres >/dev/null 2>&1 || :
```

注意：2>&1 是把 stderr 输出到 stdout 上。

第二步：

在编译 PostgreSQL 之前。首先看一下“egcs-c++-1.1.2-24.i386.rpm”是不是已经安装。没有的话，那就赶紧装吧。“egcs-c++-1.1.2-24.i386.rpm”直接可从 Redhat 的光盘里获得，在“RedHat/RPMS”下。

验证 egcs-c++-1.1.2-24.i386.rpm 是否安装，用命令：

```
[root@deep]# rpm -q egcs-c++
```

安装 egcs-c++-1.1.2-24.i386.rpm，用命令：

```
[root@deep]# rpm -Uvh egcs-c++-1.1.2-24.i386.rpm
```

第三步：

进入到 PostgreSQL 的源代码的目录：

```
[root@deep]# cd src
```

设置编译参数：

```
CC="egcs" \
./configure \
--prefix=/usr \
--enable-locale
```

这些编译参数告诉编译器如何编译 PostgreSQL：

- 使“locale”有效

编辑 Makefile.global（vi +210 Makefile.global），并做如下改变：

```
CFLAGS= -I$(SRCDIR)/include -I$(SRCDIR)/backend
```

改为：

```
CFLAGS= -I$(SRCDIR)/include -I$(SRCDIR)/backend -O9 -funroll-loops -ffast-math
-malign-double -mcpu=
pentiumpro -march=pentiumpro -fomit-frame-pointer -fno-exceptions
```

这是对 PostgreSQL 的一些优化设置。当然，你要根据自己的实际情况改变它。

运行下面的命令：

```
[root@deep]# make all
[root@deep]# cd ..
[root@deep]# make -C src install
[root@deep]# make -C src/man install
[root@deep]# mkdir -p /usr/include/pgsql
[root@deep]# mv /usr/include/access /usr/include/pgsql/
[root@deep]# mv /usr/include/commands /usr/include/pgsql/
[root@deep]# mv /usr/include/executor /usr/include/pgsql/
[root@deep]# mv /usr/include/lib /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq++ /usr/include/pgsql/
[root@deep]# mv /usr/include/port /usr/include/pgsql/
[root@deep]# mv /usr/include/utils /usr/include/pgsql/
[root@deep]# mv /usr/include/fmgr.h /usr/include/pgsql/
[root@deep]# mv /usr/include/os.h /usr/include/pgsql/
[root@deep]# mv /usr/include/config.h /usr/include/pgsql/
[root@deep]# mv /usr/include/c.h /usr/include/pgsql/
[root@deep]# mv /usr/include/postgres.h /usr/include/pgsql/
[root@deep]# mv /usr/include/postgres_ext.h /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq-fe.h /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq-int.h /usr/include/pgsql/
```

```
[root@deep]# mv /usr/include/ecpgerrno.h /usr/include/pgsql/
[root@deep]# mv /usr/include/ecpglib.h /usr/include/pgsql/
[root@deep]# mv /usr/include/ecpgtype.h /usr/include/pgsql/
[root@deep]# mv /usr/include/sqlca.h /usr/include/pgsql/
[root@deep]# mv /usr/include/libpq++.H /usr/include/pgsql/
[root@deep]# mkdir -p /usr/lib/pgsql
[root@deep]# mv /usr/lib/*source /usr/lib/pgsql/
[root@deep]# mv /usr/lib/*sample /usr/lib/pgsql/
[root@deep]# mkdir -p /var/lib/pgsql
[root@deep]# chown -R postgres.postgres /var/lib/pgsql/
[root@deep]# chmod 755 /usr/lib/libpq.so.2.0
[root@deep]# chmod 755 /usr/lib/libecpg.so.3.0.0
[root@deep]# chmod 755 /usr/lib/libpq++.so.3.0
[root@deep]# strip /usr/bin/postgres
[root@deep]# strip /usr/bin/postmaster
[root@deep]# strip /usr/bin/ecpg
[root@deep]# strip /usr/bin/pg_id
[root@deep]# strip /usr/bin/pg_version
[root@deep]# strip /usr/bin/pg_dump
[root@deep]# strip /usr/bin/pg_passwd
[root@deep]# strip /usr/bin/psql
[root@deep]# rm -f /usr/lib/global1.description
[root@deep]# rm -f /usr/lib/local1_template1.description
```

对上面命令的解释如下：

“make”命令把所有的原程序编译并产生可执行程序。然后用 “make install”把可执行文件和另外一些需要的文件安装到系统。用“mkdir”在“/usr/include”和“/usr/lib”下各产生产生“pgsql”的目录。然后用 mv 把所有在“/usr/include”和“/usr/lib”下和 PostgreSQL 相关的文件和目录分别全部移到“/usr/include/pgsql”和“/usr/lib/pgsql”下。

“chown”命令是用来设置“/var/lib/pgsql”目录的正确的权限。“strip”命令将去掉指定文件的一些符号标记比如调试信息等，这样文件会变小一点。这可以提高可执行文件一点速度。

“rm”命令去掉一些 PostgreSQL 不需要的文件。

用 postgres 数据库超级用户完成数据库的安装

```
[root@deep]# su postgres
[postgres@deep]# initdb --pglib=/usr/lib/pgsql --pgdata=/var/lib/pgsql
```

```
We are initializing the database system with username postgres (uid=40).
This user will own all the files and must also own the server process.
Creating Postgres database system directory /var/lib/pgsql/base
Creating template database in /var/lib/pgsql/base/template1
Creating global classes in /var/lib/pgsql/base
Adding template1 database to pg_database...
Vacuuming template1
Creating public pg_user view
Creating view pg_rules
Creating view pg_views
Creating view pg_tables
Creating view pg_indexes
Loading pg_description


[postgres@deep]# chmod 640 /var/lib/pgsql/pg_pwd
[postgres@deep]# exit
```

注意：不要用“root”用户安装数据库。这样会有很严重的安全问题。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf postgresql-version/ postgresql-version_tar.gz
```

卸掉“egcs-c++-1.1.2-24.i386.rpm”包以节省空间：

```
[root@deep]# rpm -e egcs-c++
```

“rm”命令将删除我们刚才展开的所有原程序和 PostgreSQL 的 tar.gz 文件。“rpm －e”将卸掉

“egcs-c++”软件包。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 PostgreSQL Database Server，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“postgresql”脚本文件拷贝到“/etc/rc.d/init.d/”目录下。

配置 /etc/rc.d/init.d/postgresql 脚本文件

配置“/etc/rc.d/ini.d/postgresql”脚本文件，用来启动和停止 PostgreSQL 服务器。

创建“postgresql”脚本文件（touch /etc/rc.d/init.d/postgresql）并加入：

```
#! /bin/sh
# postgresql This is the init script for starting up the PostgreSQL
# server
# chkconfig: 345 85 15
# description: Starts and stops the PostgreSQL backend daemon that handles \
# all database requests.
# processname: postmaster
# pidfile: /var/run/postmaster.pid
#
# Source function library.
. /etc/rc.d/init.d/functions
# Get config.
. /etc/sysconfig/network
# Check that networking is up.
# Pretty much need it for postmaster.
[ ${NETWORKING} = "no" ] && exit 0
[ -f /usr/bin/postmaster ] || exit 0
# This script is slightly unusual in that the name of the daemon (postmaster)
# is not the same as the name of the subsystem (postgresql)
# See how we were called.
case "$1" in
start)
echo -n "Checking postgresql installation: "
# Check for the PGDATA structure
if [ -f /var/lib/pgsql/PG_VERSION ] && [ -d /var/lib/pgsql/base/template1 ]
then
# Check version of existing PGDATA
if [ `cat /var/lib/pgsql/PG_VERSION` != '6.5' ]
then
echo "old version. Need to Upgrade."
echo "See /usr/doc/postgresql-6.5.2/README.rpm for more information."
exit 1
else
echo "looks good!"
```

```
fi
# No existing PGDATA! Initdb it.
else
echo "no database files found."
if [ ! -d /var/lib/pgsql ]
then
mkdir -p /var/lib/pgsql
chown postgres.postgres /var/lib/pgsql
fi
su -l postgres -c '/usr/bin/initdb --pglib=/usr/lib/pgsql
--pgdata=/var/lib/pgsql'
fi
# Check for postmaster already running...
pid=`pidof postmaster`
if [ $pid ]
then
echo "Postmaster already running."
else
#all systems go -- remove any stale lock files
rm -f /tmp/.s.PGSQL.* > /dev/null
echo -n "Starting postgresql service: "
su -l postgres -c '/usr/bin/postmaster -i -S -D/var/lib/pgsql'
sleep 1
pid=`pidof postmaster`
if [ $pid ]
then
echo -n "postmaster [$pid]"
touch /var/lock/subsys/postgresql
echo $pid > /var/run/postmaster.pid
echo
else
echo "failed."
fi
fi
;;
stop)
echo -n "Stopping postgresql service: "
killproc postmaster
sleep 2
rm -f /var/run/postmaster.pid
rm -f /var/lock/subsys/postgresql
echo
;;
status)
```

```
status postmaster
;;
restart)
$0 stop
$0 start
;;
*)
echo "Usage: postgresql {start|stop|status|restart}"
exit 1
esac
exit 0
```

现在让脚本可执行并设置它的缺省权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/postgresql
```

用下面命令创建“rc.d”目录下 PostgresSQL 的符号链接：

```
[root@deep]# chkconfig --add postgresql
```

用下面的命令手工启动 PostgreSQL：

```
[root@deep]# /etc/rc.d/init.d/postgresql start
```

命令

下面列出的是一些我们经常要用到的命令，当然还有很多其它的命令，更详细的信息可以查看 man 帮助页或其它文档。

在"pg_hba.conf"文件的 PG_DATA 段可以用 ip 地址和用户名限制对能数据库的连接。

- 用“createuser”命令在数据库中定义一个新用户：

```
[root@deep]# su postgres
[postgres@deep]$ createuser
Enter name of user to add ---> admin
Enter user's postgres ID or RETURN to use unix user ID: 500 ->
Is user "admin" allowed to create databases (y/n) y
Is user "admin" a superuser? (y/n) y
createuser: admin was successfully added
```

- 用“destroyuser”命令在数据库中删除用户：

```
[root@deep]# su postgres
[postgres@deep]$ destroyuser
Enter name of user to delete ---> admin
destroyuser: delete of user admin was successful.
```

- 用“createdb”命令创建新的数据库：

```
[root@deep]# su postgres
[postgres@deep]$ createdb dbname (the name of the database).
```

也可以用 postgres 的终端程序（psql）完成：

```
[root@deep]# su admin
[admin@deep]$ psql template1


Welcome to the POSTGRESQL interactive sql monitor:
Please read the file COPYRIGHT for copyright terms of POSTGRESQL
[PostgreSQL 6.5.3 on i686-pc-linux-gnu, compiled by egcs ]
type \? for help on slash commands
type \q to quit
type \g or terminate with semicolon to execute query
You are currently connected to the database: template1
template1 ➔  create database foo;
CREATEDB
```

其它一些有用的终端程序（psql）命令：

- 连接到另外的数据库：

```
template1  ➔  \c foo
connecting to new database: foo
foo  ➔
```

- 创建一个表：

```
foo  ➔  create table bar (i int4, c char(16));
CREATE
foo  ➔
```

- 检查新表：

```
foo ➔ \d bar
Table = bar
+-------+--------+--------+
| Field | Type   | Length |
+-------+--------+--------+
| I     | int4   |      4 |
| c     | char() |     16 |
+-------+--------+--------+
foo ➔
```

- 删除表、索引、视图：

```
foo ➔ drop table table_name;
foo ➔ drop index index_name;
foo ➔ drop view view_name;
```

- 往一个表里插入记录：

```
foo ➔ insert into table_name (name_of_attr1, name_of_attr2, name_of_attr3)
foo ➔ values (value1, value2, value3);
```

安装到系统中的文件

```
> /etc/rc.d/init.d/postgresql
> /etc/rc.d/rc0.d/K15postgresql
> /etc/rc.d/rc1.d/K15postgresql
> /etc/rc.d/rc2.d/K15postgresql
> /etc/rc.d/rc3.d/S85postgresql
> /etc/rc.d/rc4.d/S85postgresql
> /etc/rc.d/rc5.d/S85postgresql
> /etc/rc.d/rc6.d/K15postgresql
> /usr/bin/postgres
> /usr/bin/postmaster
> /usr/bin/ecpg
> /usr/bin/pg_id
> /usr/bin/pg_version
> /usr/bin/psql
> /usr/bin/pg_dump
> /usr/bin/pg_dumpall
> /usr/bin/pg_upgrade
> /usr/bin/pg_passwd
> /usr/bin/cleardbdir
```

```
> /usr/bin/createdb
> /usr/bin/createlang
> /usr/bin/createuser
> /usr/bin/destroydb
> /usr/bin/destroylang
> /usr/bin/destroyuser
> /usr/bin/initdb
> /usr/bin/vacuumdb
> /usr/bin/initlocation
> /usr/bin/ipcclean
> /usr/include/lib
> /usr/include/lib/dllist.h
> /usr/include/pgsql
> /usr/include/pgsql/access
> /usr/include/pgsql/access/attnum.h
> /usr/include/pgsql/commands
> /usr/include/pgsql/commands/trigger.h
> /usr/include/pgsql/executor
> /usr/include/pgsql/executor/spi.h
> /usr/include/pgsql/libpq
> /usr/include/pgsql/libpq/pqcomm.h
> /usr/include/pgsql/libpq/libpq-fs.h
> /usr/include/pgsql/libpq++
> /usr/include/pgsql/libpq++/pgconnection.h
> /usr/include/pgsql/libpq++/pgdatabase.h
> /usr/man/manl/begin.l
> /usr/man/manl/close.l
> /usr/man/manl/cluster.l
> /usr/man/manl/commit.l
> /usr/man/manl/copy.l
> /usr/man/manl/create_aggregate.l
> /usr/man/manl/create_database.l
> /usr/man/manl/create_function.l
> /usr/man/manl/create_index.l
> /usr/man/manl/create_language.l
> /usr/man/manl/create_operator.l
> /usr/man/manl/create_rule.l
> /usr/man/manl/create_sequence.l
> /usr/man/manl/create_table.l
> /usr/man/manl/create_trigger.l
> /usr/man/manl/create_type.l
> /usr/man/manl/create_user.l
> /usr/man/manl/create_version.l
> /usr/man/manl/create_view.l
```

```
> /usr/man/manl/declare.l
> /usr/man/manl/delete.l
> /usr/man/manl/drop.l
> /usr/man/manl/drop_aggregate.l
> /usr/man/manl/drop_database.l
> /usr/man/manl/drop_function.l
> /usr/man/manl/drop_index.l
> /usr/man/manl/drop_language.l
> /usr/man/manl/drop_operator.l
> /usr/man/manl/drop_rule.l
> /usr/man/manl/drop_sequence.l
> /usr/man/manl/drop_table.l
> /usr/man/manl/drop_trigger.l
> /usr/man/manl/drop_type.l
> /usr/man/manl/drop_user.l
> /usr/man/manl/drop_view.l
> /usr/man/manl/end.l
> /usr/man/manl/explain.l
> /usr/man/manl/fetch.l
> /usr/man/manl/grant.l
> /usr/man/manl/insert.l
> /usr/man/manl/listen.l
> /usr/man/manl/load.l
> /usr/man/manl/lock.l
> /usr/man/manl/move.l
> /usr/include/pgsql/libpq++/pgtransdb.h
> /usr/include/pgsql/libpq++/pgcursordb.h
> /usr/include/pgsql/libpq++/pglobject.h
> /usr/include/pgsql/port
> /usr/include/pgsql/port/linux
> /usr/include/pgsql/utils
> /usr/include/pgsql/utils/geo_decls.h
> /usr/include/pgsql/utils/elog.h
> /usr/include/pgsql/utils/palloc.h
> /usr/include/pgsql/utils/mcxt.h
> /usr/include/pgsql/fmgr.h
> /usr/include/pgsql/os.h
> /usr/include/pgsql/config.h
> /usr/include/pgsql/c.h
> /usr/include/pgsql/postgres.h
> /usr/include/pgsql/postgres_ext.h
> /usr/include/pgsql/libpq-fe.h
> /usr/include/pgsql/libpq-int.h
> /usr/include/pgsql/ecpgerrno.h
```

```
> /usr/include/pgsql/ecpglib.h
> /usr/include/pgsql/ecpgtype.h
> /usr/include/pgsql/sqlca.h
> /usr/include/pgsql/libpq++.H
> /usr/lib/libpq.a
> /usr/lib/libpq.so.2.0
> /usr/lib/libpq.so.2
> /usr/lib/libpq.so
> /usr/lib/libecpg.a
> /usr/lib/libecpg.so.3.0.0
> /usr/lib/libecpg.so.3
> /usr/lib/libecpg.so
> /usr/lib/libpq++.a
> /usr/lib/libpq++.so.3.0
> /usr/lib/libpq++.so.3
> /usr/lib/libpq++.so
> /usr/lib/plpgsql.so
> /usr/lib/pgsql
> /usr/lib/pgsql/global1.bki.source
> /usr/lib/pgsql/local1_template1.bki.source
> /usr/lib/pgsql/pg_geqo.sample
> /usr/lib/pgsql/pg_hba.conf.sample
> /usr/man/man1/cleardbdir.1
> /usr/man/man1/createdb.1
> /usr/man/man1/createuser.1
> /usr/man/man1/destroydb.1
> /usr/man/man1/destroyuser.1
> /usr/man/man1/ecpg.1
> /usr/man/man1/initdb.1
> /usr/man/man1/initlocation.1
> /usr/man/man1/ipcclean.1
> /usr/man/man1/pg_dump.1
> /usr/man/man1/pg_dumpall.1
> /usr/man/man1/pg_passwd.1
> /usr/man/man1/pg_upgrade.1
> /usr/man/man1/postgres.1
> /usr/man/man1/postmaster.1
> /usr/man/man1/psql.1
> /usr/man/man3/catalogs.3
> /usr/man/man3/libpq.3
> /usr/man/man5/pg_hba.conf.5
> /usr/man/manl
> /usr/man/manl/abort.l
> /usr/man/manl/alter_table.l
```

```
> /usr/man/manl/alter_user.l
> /usr/man/manl/notify.l
> /usr/man/manl/reset.l
> /usr/man/manl/revoke.l
> /usr/man/manl/rollback.l
> /usr/man/manl/select.l
> /usr/man/manl/set.l
> /usr/man/manl/show.l
> /usr/man/manl/sql.l
> /usr/man/manl/update.l
> /usr/man/manl/vacuum.l
> /var/lib/pgsql
> /var/lib/pgsql/base
> /var/lib/pgsql/base/template1
> /var/lib/pgsql/base/template1/pg_proc
> /var/lib/pgsql/base/template1/pg_type
> /var/lib/pgsql/base/template1/pg_attribute
> /var/lib/pgsql/base/template1/pg_class
> /var/lib/pgsql/base/template1/pg_inherits
> /var/lib/pgsql/base/template1/pg_index
> /var/lib/pgsql/base/template1/pg_statistic
> /var/lib/pgsql/base/template1/pg_operator
> /var/lib/pgsql/base/template1/pg_opclass
> /var/lib/pgsql/base/template1/pg_am
> /var/lib/pgsql/base/template1/pg_amop
> /var/lib/pgsql/base/template1/pg_amproc
> /var/lib/pgsql/base/template1/pg_language
> /var/lib/pgsql/base/template1/pg_aggregate
> /var/lib/pgsql/base/template1/pg_ipl
> /var/lib/pgsql/base/template1/pg_inheritproc
> /var/lib/pgsql/base/template1/pg_rewrite
> /var/lib/pgsql/base/template1/pg_listener
> /var/lib/pgsql/base/template1/pg_description
> /var/lib/pgsql/base/template1/pg_attribute_relid_attnam_index
> /var/lib/pgsql/base/template1/pg_attribute_relid_attnum_index
> /var/lib/pgsql/base/template1/pg_attribute_attrelid_index
> /var/lib/pgsql/base/template1/pg_proc_oid_index
> /var/lib/pgsql/base/template1/pg_proc_proname_narg_type_index
> /var/lib/pgsql/base/template1/pg_proc_prosrc_index
> /var/lib/pgsql/base/template1/pg_type_oid_index
> /var/lib/pgsql/base/template1/pg_type_typname_index
> /var/lib/pgsql/base/template1/pg_class_oid_index
> /var/lib/pgsql/base/template1/pg_class_relname_index
> /var/lib/pgsql/base/template1/pg_attrdef
```

```
> /var/lib/pgsql/base/template1/pg_attrdef_adrelid_index
> /var/lib/pgsql/base/template1/pg_relcheck
> /var/lib/pgsql/base/template1/pg_relcheck_rcrelid_index
> /var/lib/pgsql/base/template1/pg_trigger
> /var/lib/pgsql/base/template1/pg_trigger_tgrelid_index
> /var/lib/pgsql/base/template1/pg_description_objoid_index
> /var/lib/pgsql/base/template1/PG_VERSION
> /var/lib/pgsql/base/template1/pg_user
> /var/lib/pgsql/base/template1/pg_rules
> /var/lib/pgsql/base/template1/pg_views
> /var/lib/pgsql/base/template1/pg_tables
> /var/lib/pgsql/base/template1/pg_indexes
> /var/lib/pgsql/pg_variable
> /var/lib/pgsql/pg_database
> /var/lib/pgsql/pg_shadow
> /var/lib/pgsql/pg_group
> /var/lib/pgsql/pg_log
> /var/lib/pgsql/PG_VERSION
> /var/lib/pgsql/pg_hba.conf
> /var/lib/pgsql/pg_geqo.sample
> /var/lib/pgsql/pg_pwd
```

Linux Squid Proxy 服务器

概述

目前市场上若干可供选择的代理服务器产品一般有两个主要的缺陷：其一，作为商业软件价格昂贵；其二，往往不支持 ICP。虽然 Apache Web 服务器从 1.2 版开始包括了一个免费的的代理缓冲模块，可以兼容大多数流行的 web 服务器。但是它也不支持 ICP，并且其健壮性难以和免费的、专门的代理缓冲服务器——Squid 相提并论。

Squid 服务器有下面若干程序组成：

- **squid**：主代理服务器
- **dnsserver:**：一个 DNS 查找程序，实现非并发的，阻塞方式的 DNS 操作
- **unlinkd**：一个在后台清除缓冲目录下的文件的程序

Squid 服务器还带有一个可以让管理员通过 web 界面对配置和性能的统计信息进行监控和管理的 CGI 程序。

Squid 是一个由众多在互联网上的开发人员共同努力完成的高性能的代理缓冲服务器。具体开发是由国家网络应用研究室（the National Laboratory for Applied Network Research）的 Duane Wessels 主持，由 NSF 出资支持的。Squid 来源于 ARPA 出资开发的缓冲服务器 Harvest research Project 工程。Squid 不但实现对 Web 信息的高效缓冲，同样支持 FTP、Gopher 和 HTTP 请求。Squid 在内存里保存访问频繁的对象，并且在硬盘内维护一个健壮的访问对象数据库。Squid 同样支持 SSL 协议来缓冲安全连接，并且提供了复杂的访问控制机制。此外，Squid 能以级连的（hierarchically linked）方式连接到另一个基于 Squid 的缓冲代理服务器来实现对访问页面的高效（streamlined）缓冲。

在下面的编译配置实例中，将配置 Squid 为“httpd-accelerator”（HTTP 加速器）的工作模式以提高效率。在加速器的工作模式中，Squid 完成反向代理缓冲的功能。其接受来自浏览器客户端的连接请求，如果有可能就使用自己缓冲内的内容应答客户端的请求，或者从请求的真正目的服务器处取得数据，Squid 相对于真正的目的服务器来说是反向代理。将 Squid 加速器取代 WWW 服务器监听 80 端口（或任何 WWW 服务器监听的端口），只有 Squid 加速器从真正的 WWW 服务器处访问数据（只有加速器需要知道如何访问真正的 WWW）。而对于外部的客户端来说，并没有什么区别，区别仅仅是速度变快。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

Squid 版本号上是 2_3_STABLE1。

安装包

Squid 主页地址为：http://squid.nlanr.net/。

要下载的文件是 squid-2_3_STABLE1-src_tar.gz。

安装软件包需要注意的问题

最好保存系统安装 Squid 前和安装 Squid 后系统的文件信息，然后比较前后有什么不同来寻找哪些文件被安装在哪里。可以通过：在系统安装 Squid 以前运行“find /* >squid1”，安装之后运行：“find /* > squid2”，然后运行：“diff squid1 squid2 >squid”来得到系统改变信息。

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp squid-version_STABLEz-src_tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf squid-version_STABLEz-src_tar.gz
```

配置和优化

Squid 不能以“root”用户的身份运行，所以要创建一个特殊的用户来运行 Squid 代理服务器：

```
[root@deep]# /usr/sbin/useradd -d /cache/ -r -s /dev/null squid >/dev/null 2>&1
[root@deep]# mkdir /cache/
[root@deep]# chown -R squid.squid /cache/
```

首先，在“/etc/passwd”添加用户 Squid，然后创建“/cache”目录。接着把“/cache”的所有者改为 squid 用户。

注意：通常没有必要创建目录“/cache”，因为在硬盘分区的时候一般已经创建了该目录。若没有创建，才有必要创建该目录。

然后，进入新的 Squid 目录，编辑 Makefile（vi +18 icons/Makefile.in）文件：

把

```
DEFAULT_ICON_DIR = $(sysconfdir)/icons
```

改为：

```
DEFAULT_ICON_DIR = $(libexecdir)/icons
```

这里改变变量（sysconfdir）为（libexecdir）。这个修改将使 Squid 的“icon”目录变为“/usr/lib/squid/”。

编辑“Makefile.in” 文件（vi +34 src/Makefile.in）：

把

```
DEFAULT_CACHE_LOG = $(localstatedir)/logs/cache.log
```

改为：

```
DEFAULT_CACHE_LOG = $(localstatedir)/log/squid/cache.log
```

把

```
DEFAULT_ACCESS_LOG = $(localstatedir)/logs/access.log
```

改为

```
DEFAULT_ACCESS_LOG = $(localstatedir)/log/squid/access.log
```

把

```
DEFAULT_STORE_LOG = $(localstatedir)/logs/store.log
```

改为

```
DEFAULT_STORE_LOG = $(localstatedir)/log/squid/store.log
```

把

```
DEFAULT_PID_FILE = $(localstatedir)/logs/squid.pid
```

改为

```
DEFAULT_PID_FILE = $(localstatedir)/run/squid.pid
```

把

```
DEFAULT_SWAP_DIR = $(localstatedir)/cache
```

改为

```
DEFAULT_SWAP_DIR = /cache
```

把

```
DEFAULT_ICON_DIR = $(sysconfdir)/icons
```

改为

```
DEFAULT_ICON_DIR = $(libexecdir)/icons
```

这里把文件“cache.log”、“access.log”和“store.log”的缺省目录改为“/var/log/squid/”，然后将 Squid 的 pid 文件存改为放在目录“/var/run/”下，最后修改“icons”目录为“/usr/lib/squid/icons/”。

malloc

如果把 Squid 和外部的 malloc 库（如 GNU malloc）链接使用，可提高性能。为了让 Squid 使用外部的 malloc 库，根据下面的示例进行：

1. 下载 GNU 的 malloc 库源代码：http://www.gnu.org/order/ftp.html。

2. 编译 GNU malloc：

```
[root@deep]# tar xzpf malloc.tar.gz
[root@deep]# cd malloc
[root@deep]# vi Makefile and uncomment the line: CPPFLAGS = -DUSG
[root@deep]# export CC=egcs
[root@deep]# make
```

3. 把“libmalloc.a”拷贝到系统库目录，并将其改名为“libgnumalloc.a”：

```
[root@deep]# cp libmalloc.a /usr/lib/libgnumalloc.a
```

4. （可选的）把 GNU“malloc.h”拷贝到系统头文件目录并确保改名为“gnumalloc.h”。这一步不是必须的，但是如果拷贝了头文件，Squid 将能使用 mstat()函数在 cachemgr 信息页报告内存使用统计信息。

```
[root@deep]# cp malloc.h /usr/include/gnumalloc.h
```

编译和优化

返回 Squid 所在目录，敲入如下的命令：

```
[root@deep]# export CACHE_HTTP_PORT=80
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-
pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--exec-prefix=/usr \
--bindir=/usr/sbin \
--libexecdir=/usr/lib/squid \
--localstatedir=/var \
--sysconfdir=/etc/squid \
--enable-cache-digests \
--enable-poll \
```

```
--disable-ident-lookups \
--enable-truncate \
--enable-underscores \
--enable-heap-replacement
```

这些编译参数告诉编译器如何编译 SSH1：

- 使用缓冲摘要（Cache Digest）来提高性能。
- 使用 poll()来替代 select()。
- “Disable-ident-lookups”防止系统使用 RFC931 规定的身份识别方法。
- “Enable-truncate”使系统在删除缓冲文件时使用 truncate()替代 unlink()。Truncate 的性能比 unlink 要好一些。而 Truncate 相对于 unlink 来说要使用更多的文件系统 inodes。
- “Enable-underscores.Squid”缺省地拒绝任何主机名带有“_”的主机的服务请求，来保持和 Internet 标准一致，可以通过打开这个开关来防止 Squid 主机拒绝接受主机名带有“_”的主机请求。
- “Enable-heap-replacement”选项允许系统使用各种缓冲置换算法，而不是标准的 LRU 算法。

```
[root@deep]# make -f makefile
[root@deep]# make install
[root@deep]# mkdir -p /var/log/squid
[root@deep]# rm -rf /var/logs/
[root@deep]# chown squid.squid /var/log/squid/
[root@deep]# chmod 750 /var/log/squid/
[root@deep]# chmod 750 /cache/
[root@deep]# rm -f /usr/sbin/RunCache
[root@deep]# rm -f /usr/sbin/RunAccel
[root@deep]# strip /usr/sbin/squid
[root@deep]# strip /usr/sbin/client
[root@deep]# strip /usr/lib/squid/dnsserver
[root@deep]# strip /usr/lib/squid/unlinkd
[root@deep]# strip /usr/lib/squid/cachemgr.cgi
```

“make –f”命令将编译所有的源文件为可执行代码。“make install”完成拷贝二进制代码和所有支持文件（如配置文件等）到特定的目录下。“mkdir”命令在“/var/log/”目录下创建一个新的 squid 的子目录。“rm –rf”命令删除安装程序设定的缺省的日志文件所在目录“/var/logs”，因为前面的命令创建了新的日志文件的目录。“chown”命令把“/var/log/squid/”所有权改变为用户 squid 所有。出于安

全方面的考虑使用“chmod”命令把“squid”和“cache”子目录的访问权限设置为（0750/drwxr-x---）。

这里还删除了实现以缓冲工作模式或加速模式启动 Squid 的两个脚本文件：RunCache、 RunAccel，因为这里我们使用了一个更好的位于“/etc/rc.d/init.d/”目录下的名为 squid 的脚本文件。出于优化的目的，使用 strip 命令来减小二进制代码的大小。

什么是缓冲摘要 Cache Digest

缓冲摘要（Cache Digest）是互联网对象缓冲服务器（Internet Object Caching Server）所缓冲的内容的一种概括的说法。它以压缩的格式保存了“服务器是否缓冲了某个 URL 资源”的缓冲信息。Squid 使用了一个有信息损耗的压缩技术，这意味着在不能获得 100%的准确信息的情况下使用很高的压缩比因子。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf squid-version/ squid-version_STABLEz-src_tar.gz
[root@deep]# rm -rf malloc/ malloc.tar.gz (if your are used the malloc)
```

rm 命令删除安装 Squid 和 malloc 的源代码文件，同时也删除了“/var/tmp”目录下的 Squid 和 Malloc 压缩文件。

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 Squid 服务器，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“squid.conf”文件拷贝到“/etc/squid”目录下
- 把“squid”脚本文件拷贝到“/etc/rc.d/init.d”目录下
- 把“squid”文件拷贝到“/etc/logrotate.d”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到相应的目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

把 /etc/squid/squid.conf 文件配置为 httpd 加速器工作方式

如果 Web 服务器与 Squid 运行在同一个服务器上，就需要把 Web 改为运行在 81 号端口下，可以通过修改 Web 服务器的配置文件 httpd.conf 来实现。如果不运行在同一个服务器上，则无须考虑这个问题。

编辑“squid.conf”文件（vi /etc/squid/squid.conf），增加下面内容：

```
http_port 80
icp_port 0
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
cache_mem 16 MB
cache_dir ufs /cache 200 16 256
log_icp_queries off
buffered_logs on
emulate_httpd_log on
redirect_rewrites_host_header off
replacement_policy GDSF
half_closed_clients off
acl all src 0.0.0.0/0.0.0.0
http_access allow all
cache_mgr admin
cache_effective_user squid
cache_effective_group squid
httpd_accel_host 208.164.186.3
httpd_accel_port 80
log_icp_queries off
buffered_logs on
```

下面逐行说明“squid.conf”文件中参数的设置：

http_port 80

“http_port”参数指定 Squid 监听浏览器客户请求的端口号。

icp_port 0

“icp_port”参数指定 Squid 从邻居（neighbour）服务器缓冲内发送和接收 ICP 请求的端口号。这里设置为 0 是因为这里配置 Squid 为内部 Web 服务器的加速器，所以不需要使用邻居服务器的缓冲。

acl QUERY urlpath_regex cgi-bin \? and no_cache deny QUERY

用来强制某些特定的对象不被缓存，主要是处于安全的目的。

cache_mem 16 MB

cache_mem 参数为：In-Transit objects,Hot Objects, Negative-Cached objects 指定合理的内存数量。这是一个优化选项，增加该内存值有利于缓存。应该注意的是：Squid 使用的值要远远大于该值。一般来说如果系统有 nM 内存，设置该值为(n/)3M。

cache_dir ufs /cache 200 16 256

cache_dir 参数设定使用的存储系统的类型。一般情况下都类型应该是 ufs，目录应该是“/cache”，在该目录下使用的缓冲值为 200MB，允许在“/cache”下创建的第一级子目录数为 16，每个第一级子目录下可以创建的第二级子目录数量为 256。

emulate_httpd_log on

打开“emulate_httpd_log”选项，将使 Squid 仿照 Web 服务器的格式创建访问记录。如果希望使用 Web 访问记录分析程序，就需要设置这个参数。

redirect_rewrites_host_header off

缺省地，Squid 将改写任何重定向请求的主机头部。若系统运行 Squid 为加速器模式，则这时不需要重定向特性。该参数在负载过重的情况下要旁路重定向器时才打开

replacement_policy GDSF

缓冲置换策略参数决定当需要多余的磁盘空间时哪个对象被废弃（置换）。当编译 Squid 时使用了“-enable-heap-replacement”参数时，这里可以选择使用两个新的，增强的置换策略：

GDSF: Greedy-Dual Size Frequency

LFUDA: Least Frequently Used with Dynamic Aging

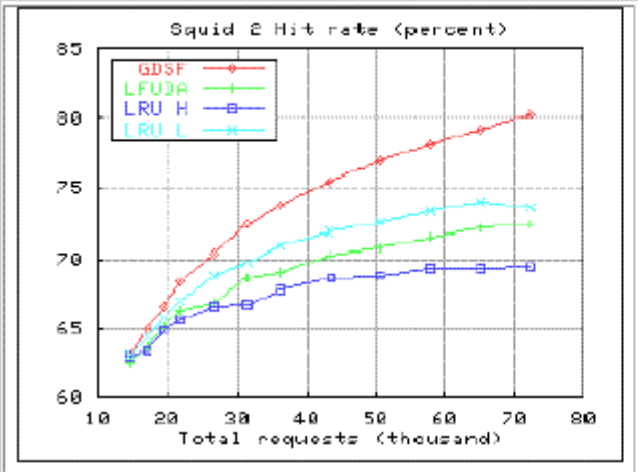


Figure 1: SPECWeb Hit Rate

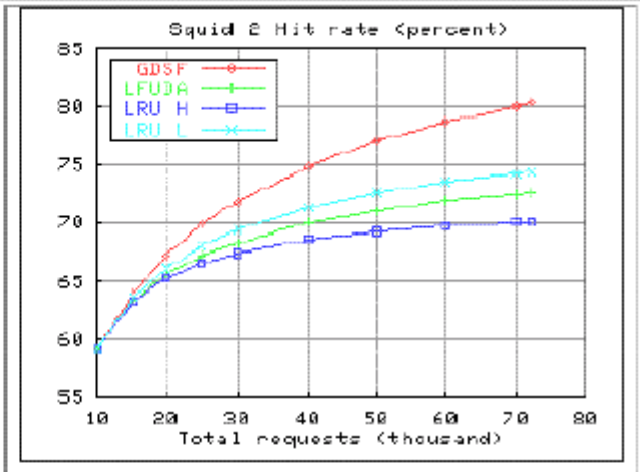


Figure 2: http Hit Rate

Figures 1 and 2 show the hit rate of the Squid 2 implementations under the SPECWeb workload and the http workload. In hit rate there is little difference between the original SPECWeb validation and this revalidation: the relative ordering has not changed and the shape of the curves is broadly similar. There is greater variance in the policies in the SPECWeb version and note that the run length of the http test is slightly greater.

这两个策略从效率上都强于 Squid 缺省的 LRU 策略，当编译时没有使用 -enable-heap-replacement 参数，则 Squid 缺省使用 LRU 策略。

GDSF 策略通过在缓冲中保持更小的热点（hot）对象来优化对象点击率，从而使各个对象有更大的几率被点击。但是 GDSF 相对于 LFUDA 来说其平均字节点击率要小一些，因为 GDSF 往往将大的对象（可能是热点对象）置换出缓冲。

LFUDA 策略将热点对象统统保持在缓冲中，而不考虑其大小，从而获得了更高的平均字节点击率。然而却降低了平均对象点击率，因为大的对象往往使减少小对象保持在缓冲内的几率。

half_closed_clients off

将“half_closed_clients”选项关闭，使 Squid 在当 read(2)不再返回数据时立即关闭客户端的连接。有时 read(2)不再返回数据是由于某些客户关闭 TCP 的发送数据而仍然保持接收数据。而 Squid 分辨不出 TCP 半关闭和完全关闭。

acl all src 0.0.0.0/0.0.0.0 and http_access allow all

“acl and http_access”选项定义了一个访问控制列表。详细情况参见和 Squid 软件携带的文档。这里的访问控制列表允许所有对代理服务的访问，因为这里该代理是加速 web 服务器。

cache_mgr admin

“cache_mgr”指定当缓冲出现问题时，向缓冲管理者发送告警信息的目的地址信息。

cache_effective_user squid and cache_effective_group squid

“cache_effective_user”和“cache_effective_group”指定缓冲服务器运行时的有效 UID/GID。出于安全的原因，一般不能以根用户的身份运行 Squid。这里以用户 squid 的身份 Squid 服务器。

httpd_accel_host 208.164.186.3 and httpd_accel_port 80

选项“httpd_accel_host”和“httpd_accel_port”定义了真正的 Web 服务器的主机名和端口号。在这里的配置中，真正的 HTTP 服务器运行在 IP 地址为 208.164.186.3（www.openarch.com）的主机上，运行端口为 80。主机 www.openarch.com 是不同于运行 Squid 服务器的主机。

log_icp_queries off

选项“log_icp_queries”设置是否把 ICP 请求记录到访问日志文件中。这里的配置不使用 ICP，所以关闭了该选项。

buffered_logs on

若打开选项“buffered_logs”可以稍稍提高加速某些对日志文件的写入，该选项主要是实现优化特性。

使 Squid 为 httpd 加速器工作模式的”/etc/rc.d/init.d/squid”脚本文件的配置

配置系统的“/etc/rc.d/init.d/squid”脚本文件来启动和停止 Squid 对象缓冲。该脚本已经被修改使缓冲目录为“/cache”而不是“/var/spool/squid”。

创建 Squid 脚本文件（touch /etc/rc.d/init.d/squid）并增加内容：

```
#!/bin/bash
# squid This shell script takes care of starting and stopping
# Squid Internet Object Cache
#
# chkconfig: - 90 25
# description: Squid - Internet Object Cache. Internet object caching is \
# a way to store requested Internet objects (i.e., data available \
# via the HTTP, FTP, and gopher protocols) on a system closer to the \
# requesting site than to the source. Web browsers can then use the \
# local Squid cache as a proxy HTTP server, reducing access time as \
# well as bandwidth consumption.
```

```
# pidfile: /var/run/squid.pid
# config: /etc/squid/squid.conf
PATH=/usr/bin:/sbin:/bin:/usr/sbin
export PATH
# Source function library.
. /etc/rc.d/init.d/functions
# Source networking configuration.
. /etc/sysconfig/network
# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0
# check if the squid conf file is present
[ -f /etc/squid/squid.conf ] || exit 0
# determine the name of the squid binary
[ -f /usr/sbin/squid ] && SQUID=squid
[ -z "$SQUID" ] && exit 0
# determine which one is the cache_swap directory
CACHE_SWAP=`sed -e 's/#.*//g' /etc/squid/squid.conf | \
grep cache_dir | sed -e 's/cache_dir//' | \
cut -d ' ' -f 2`
[ -z "$CACHE_SWAP" ] && CACHE_SWAP=/cache
# default squid options
# -D disables initial dns checks. If you most likely will not to have an
# internet connection when you start squid, uncomment this
#SQUID_OPTS="-D"
RETVAL=0
case "$1" in
start)
echo -n "Starting $SQUID: "
for adir in $CACHE_SWAP; do
if [ ! -d $adir/00 ]; then
echo -n "init_cache_dir $adir... "
$SQUID -z -F 2>/dev/null
fi
done
$SQUID $SQUID_OPTS &
RETVAL=$?
echo $SQUID
[ $RETVAL -eq 0 ] && touch /var/lock/subsys/$SQUID
;;
stop)
echo -n "Stopping $SQUID: "
$SQUID -k shutdown &
RETVAL=$?
if [ $RETVAL -eq 0 ] ; then
```

```
rm -f /var/lock/subsys/$SQUID
while : ; do
[ -f /var/run/squid.pid ] || break
sleep 2 && echo -n "."
done
echo "done"
else
echo
fi
;;
reload)
$SQUID $SQUID_OPTS -k reconfigure
exit $?
;;
restart)
$0 stop
$0 start
;;
status)
status $SQUID
$SQUID -k check
exit $?
;;
probe)
exit 0;
;;
*)
echo "Usage: $0 {start|stop|status|reload|restart}"
exit 1
esac
 exit $RETVAL
```

然后，让脚本文件可执行并改变其默认权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/squid
```

使用下面的命令在“/etc/rc.d”下创建 Squid 的符号连接：

```
[root@deep]# chkconfig --add squid
```

在缺省情况下，在 redhat 系统上当重新启动机器后 Squid 脚本将自动启动代理服务器。可以使用下面的命令来改变是否为缺省启动。

```
[root@deep]# chkconfig --level 345 squid on
```

通过下面的命令手工启动系统的 Squid 代理服务器：

```
[root@deep]# /etc/rc.d/init.d/squid start
```

/etc/logrotate.d/squid 文件的配置

配置“/etc/logrotate.d/squid”来实现每周自动轮转记录文件（rotate log files）。

创建 Squid 文件（touch /etc/logrotate.d/squid），并增加：

```
/var/log/squid/access.log {
weekly
rotate 5
copytruncate
compress
notifempty
missingok
}
/var/log/squid/cache.log {
weekly
rotate 5
copytruncate
compress
notifempty
missingok
}
/var/log/squid/store.log {
weekly
rotate 5
copytruncate
compress
notifempty
missingok
# This script asks squid to rotate its logs on its own.
# Restarting squid is a long process and it is not worth
# doing it just to rotate logs
postrotate
/usr/sbin/squid -k rotate
endscript
}
```

增强 Squid 的安全性

对加载的文件系统实施更严格的控制

可以通过在文件系统配置文件“/etc/fstab”中设置如：noexec，nodev 和 nosuid 选项参数来增强 Squid 的缓冲目录所在文件系统的安全性。编辑“fstab”文件（vi /etc/fstab）加入：

```
/dev/sda8 /cache ext2 nosuid,nodev,noexec 1 2
```

注意：这里假定“/dev/sda8”是缓冲目录所在文件系统的分区。

<nodev>参数指不要解释该文件系统上的字符特殊设备或块特殊设备。<nosuid>指该文件系统上不允许 set-uid 位或者 set-groupid 起作用。<noexec>不允许在该文件系统上运行任何二进制程序。具体详细情况可以通过“man fstab”及“man 8 mount”来获得。

设置配置文件的不可修改位以增强安全性

对文件的不可修改位的设置可以保护重要的文件，防止不小心被删除修改，还可以防止任何用户建立对该文件的链接。一旦配置好一些不动的配置文件对这些文件设置不可修改位是一个良好的习惯。

```
[root@deep]# chattr +i /etc/squid/squid.conf
```

优化 Squid

noatime 属性

Linux 可以为文件系统设置一个被称作“noatime”的参数。该参数可以在文件系统配置文件“/etc/fstab”中设置。当一个文件系统带有该参数而被加载，则对该文件系统的文件的读访问将不会导致文件的上次读访问时间（atime）被修改。一般来说文件的该属性用处不是很大，所以更新问文件该属性不是很有必要。

当“noatime”参数被设置，由于无须在每次对文件进行读访问时修改“atime”，所以可以加快对文件系统的访问效率。编辑“fstab”文件（vi /etc/fstab）：

```
/dev/sda8 /cache ext2 nosuid,nodev,noexec,noatime 1 2
```

注意：这里假定分区“/dev/sda8”是 Squid 的缓冲区目录所在分区。

重新启动系统：

```
[root@deep]# reboot
[root@deep]# cat /proc/mounts
```

bdflush 参数

下面的内容是关于“/proc/sys/vm/”目录下“sysctl”文件的，而且是针对 Linux2.2 版本的。

这个文件能被用做调节系统内核的虚拟内存（VM）的操作的。“bdflush”文件同样也对磁盘使用有小的影响。

“bdflush”控制内核守护进程 bdflush 的行为。通常使用该命令来提高文件系统性能。在“/etc/rc.d/rc.local”末尾添加如下内容，使系统启动时自动起作用：

```
echo "100 1200 128 512 15 5000 500 1884 2">/proc/sys/vm/bdflush
```

通过对 bdflush 中的缺省参数的一些调整，可以提高文件系统的响应速度。如：在写入磁盘以前等待片刻，从而有助于提高性能。

参看“/usr/src/linux/Documentation/sysctl/vm.txt”可以获得更多关于虚拟内存、磁盘缓冲及交换分区等更多的信息。

ip_local_port_range 参数

这部分是关于 Linux version 2.2 系统的“/proc/sys/net/ipv4/ip_local_port_range/”目录下的“sysctl”文件的。这个文件设置了 TCP、UDP 使用的本地端口范围。第一个数字表示端口选择范围的下限，第二个数字表示端口选择范围的上限。一般为：32768-61000.

在“/etc/rc/d/rc.local”末尾中添加如下内容：

```
echo “2768 61000”> /proc/sys/net/ipv4/ip_local_port_range
```

物理存储介质

quid 最重要的资源是物理存储介质。系统的处理器不需要是第一流的，但是系统的磁盘性能将会是系统的瓶颈。

安装到系统中的文件

```
> /etc/squid
> /etc/squid/mib.txt
> /etc/squid/squid.conf.default
> /etc/squid/squid.conf
> /etc/squid/mime.conf.default
> /etc/squid/mime.conf
> /etc/squid/errors
> /etc/squid/errors/ERR_ACCESS_DENIED
> /etc/squid/errors/ERR_CACHE_ACCESS_DENIED
> /etc/squid/errors/ERR_CACHE_MGR_ACCESS_DENIED
> /etc/squid/errors/ERR_CANNOT_FORWARD
> /etc/squid/errors/ERR_CONNECT_FAIL
> /etc/squid/errors/ERR_DNS_FAIL
> /etc/squid/errors/ERR_FORWARDING_DENIED
> /etc/squid/errors/ERR_FTP_DISABLED
> /etc/squid/errors/ERR_FTP_FAILURE
> /etc/squid/errors/ERR_FTP_FORBIDDEN
> /etc/squid/errors/ERR_FTP_NOT_FOUND
> /etc/squid/errors/ERR_FTP_PUT_CREATED
> /etc/squid/errors/ERR_FTP_PUT_ERROR
> /etc/squid/errors/ERR_FTP_PUT_MODIFIED
> /etc/squid/errors/ERR_FTP_UNAVAILABLE
> /etc/squid/errors/ERR_INVALID_REQ
> /etc/squid/errors/ERR_INVALID_URL
> /etc/squid/errors/ERR_LIFETIME_EXP
> /etc/squid/errors/ERR_NO_RELAY
> /etc/squid/errors/ERR_ONLY_IF_CACHED_MISS
> /etc/rc.d/rc4.d/S90squid
> /etc/rc.d/rc5.d/S90squid
> /etc/rc.d/rc6.d/K25squid
> /etc/logrotate.d/squid
> /usr/lib/squid
> /usr/lib/squid/dnsserver
> /usr/lib/squid/unlinkd
> /usr/lib/squid/cachemgr.cgi
> /usr/lib/squid/icons
> /usr/lib/squid/icons/anthony-binhex.gif
> /usr/lib/squid/icons/anthony-bomb.gif
> /usr/lib/squid/icons/anthony-box.gif
> /usr/lib/squid/icons/anthony-box2.gif
> /usr/lib/squid/icons/anthony-c.gif
```

```
> /usr/lib/squid/icons/anthony-compressed.gif
> /usr/lib/squid/icons/anthony-dir.gif
> /usr/lib/squid/icons/anthony-dirup.gif
> /usr/lib/squid/icons/anthony-dvi.gif
> /usr/lib/squid/icons/anthony-f.gif
> /usr/lib/squid/icons/anthony-image.gif
> /usr/lib/squid/icons/anthony-image2.gif
> /usr/lib/squid/icons/anthony-layout.gif
> /usr/lib/squid/icons/anthony-link.gif
> /usr/lib/squid/icons/anthony-movie.gif
> /usr/lib/squid/icons/anthony-pdf.gif
> /usr/lib/squid/icons/anthony-portal.gif
> /usr/lib/squid/icons/anthony-ps.gif
> /etc/squid/errors/ERR_READ_ERROR
> /etc/squid/errors/ERR_READ_TIMEOUT
> /etc/squid/errors/ERR_SHUTTING_DOWN
> /etc/squid/errors/ERR_SOCKET_FAILURE
> /etc/squid/errors/ERR_TOO_BIG
> /etc/squid/errors/ERR_UNSUP_REQ
> /etc/squid/errors/ERR_URN_RESOLVE
> /etc/squid/errors/ERR_WRITE_ERROR
> /etc/squid/errors/ERR_ZERO_SIZE_OBJECT
> /etc/rc.d/init.d/squid
> /etc/rc.d/rc0.d/K25squid
> /etc/rc.d/rc1.d/K25squid
> /etc/rc.d/rc2.d/K25squid
> /etc/rc.d/rc3.d/S90squid
> /usr/lib/squid/icons/anthony-quill.gif
> /usr/lib/squid/icons/anthony-script.gif
> /usr/lib/squid/icons/anthony-sound.gif
> /usr/lib/squid/icons/anthony-tar.gif
> /usr/lib/squid/icons/anthony-tex.gif
> /usr/lib/squid/icons/anthony-text.gif
> /usr/lib/squid/icons/anthony-unknown.gif
> /usr/lib/squid/icons/anthony-xbm.gif
> /usr/lib/squid/icons/anthony-xpm.gif
> /usr/sbin/RunCache
> /usr/sbin/RunAccel
> /usr/sbin/squid
> /usr/sbin/client
> /var/log/squ
```

Linux Apache Web 服务器

概述

Apache 是一个功能强大的 web 服务器，支持 HTTP 1.1 标准，web 页面密码验证和其他许多特性。Apache 是当今最流行的 web 服务器之一，其性能可以和任何商业服务器相媲美。

因为 Apache 是一个复杂的软件，有许多安装变量和选项。对于不同的目的可以有许多不同的文档，如果要获得更详细的信息，请阅读 Unix 下的 Apache 安装帮助文档。这里，我解释如何用许多选项编译和优化 Apache 如 mod_ssl、mod_perl、mod_php3、LDAP 等等，我并不想为每一个选项单独写文档，你可以很轻松地阅读你想要的内容，比如：Apache+PHP3，不需要 mod_ssl 或者 PostgreSQL……那么请直接阅读 Apache 和 PHP3 部分，略过其他的部分。

就像你会注意到的，在安装过程中存在许多可能性、变量和选项。所以，在下面我们提供如何按部就班地用第三方的模块配置 Apache 的方法。为简化起见，我们对每一部分内容都假设你已经满足了前提条件，如果不是这样，请自己调整步骤。

本节面向的是新的 Apache 网站管理员，介绍如何安装 Apache 服务器和让其运行。同时介绍了一些途径来调整设置以提高服务器性能。在我们的安装和设置中，我们要以非 root 用户的身份运行 Apache，以 chroot 改变环境进行额外的安全控制。

注意事项

命令均与 UNIX 兼容

源代码位于“/var/tmp”（其他路径也可以）

安装在 Red Hat 6.1 上作过全面测试

安装过程中的每一步骤都是用超级用户完成的

Apache 的版本是 1.3.9

Mod_Perl 的版本是 1.21

Mod_SSL 的版本是 2_4_10-1_3_9

PHP 的版本是 3_0_13

MM 的版本是 1.0.12

软件包的来源

Apache 主页：http://www.apache.org/

Mod_ssl 主页：http://www.modssl.org/

OpenSSL 主页：http://www.openssl.org

MM 主页：http://www.engelschall.com/sw/mm/

Imap 主页：http://www.washington.edu/imap/

PostgreSQL 主页：http://www.pstgresql.org/index.html

OpenLDAP 主页：http://www.openldap.org

Mod_perl 主页：http://perl.apache.org

Mod_php 主页：http://www.php.net

前提条件

OpenSSL 应该已经安装在了你的计算机上（如果你需要 Apache+mod_ssl）；

PosgresSQL 应该已经安装在了你的计算机上（如果你需要 Apache+PHP3+Pgsql）；

Mm 应该已经安装在了你的计算机上（如果你需要 Apache+mod_ssl+mm）；

OpenLDAP 应该已经安装在了你的计算机上（如果你需要 Apache+PHP3+LDAP）；

Perl 应该已经安装在了你的计算机上（如果你需要 Apache+mod_ssl）；

Imap 应该已经安装在了你的计算机上（如果你需要 Apache+PHP3+imap）。

为什么需要使用 MM

如果你需要在 Apache/EAPI 中支持共享内存，那么就需要建立 MM 共享内存库。

安装软件包需要注意的问题

一个比较好的办法是将你在安装 Apache 之前的文件列表和安装之后的文件列表都记录下来，然后用 diff 命令进行比较，以发现安装时都增加了哪些文件。在安装前，运行“find /* 〉apache1”，在安装后，运行“find /* 〉apache2”，然后用“diff apache1 apache2 > apache”来比较文件列表的变化。

编译

解压缩以下 tar 文件（tar.gz）：

```
[root@deep]# cp apache_version.tar.gz /var/tmp
[root@deep]# cp mod_ssl-version-version.tar.gz /var/tmp
[root@deep]# cp mod_perl-version.tar.gz /var/tmp
[root@deep]# cp php-version.tar.gz /var/tmp
[root@deep]# cp /var/tmp/
[root@deep]# tar xzpf apache_version.tar.gz
[root@deep]# tar xzpf mod_ssl-version-version_tar.gz
[root@deep]# tar xzpf mod_perl-version.tar.gz
[root@deep]# tar xzpf php-version_tar.gz
```

编译和优化

将 mod-ssl 加入 Apache 源代码树

进入新的 mod_ssl 的目录（cd mod_ssl-version）并在终端输入以下命令：

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-
pointer -fno-exceptions" \
./configure \
--with-apache=../apache_1.3.9 \
--with-crt=/etc/ssl/certs/server.crt \
--with-key=/etc/ssl/private/server.key
```

进入新的 Apache 目录（cd ../apache-version）并在终端输入以下命令：

```
[root@deep]# vi +331 src/include/httpd.h
```

编辑“httpd.h”文件并改变：

```
#define HARD_SERVER_LIMIT 256
```

为：

```
#define HARD_SERVER_LIMIT 1024
```

设置 PHP3 之前预先配置 Apache

进入新的 Apache 的目录（cd ../apache-version）并在终端输入如下命令：

```
CC="egcs" \
OPTIM="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-pointer
-fno-exceptions" \
CFLAGS="-DDYNAMIC_MODULE_LIMIT=0" \
./configure \
--prefix=/home/httpd \
--bindir=/usr/bin \
--sbindir=/usr/sbin \
--libexecdir=/usr/lib/apache \
--includedir=/usr/include/apache \
--sysconfdir=/etc/httpd/conf \
--localstatedir=/var \
--runtimedir=/var/run \
--logfiledir=/var/log/httpd \
--datadir=/home/httpd \
--proxycachedir=/var/cache/httpd \
--mandir=/usr/man
```

配置 PHP3 并加入 Apache 源代码树

进入新的 php3 目录（cd ../php-version）并在终端输入以下命令：

编辑文件“php3_pgsql.h”（vi +46 functions/php3_pgsql.h）并修改下列行：

```
#include <libpq-fe.h>
#include <libpq/libpq-fs.h>
```

改成：

```
#include </usr/include/pgsql/libpq-fe.h>
#include </usr/include/pgsql/libpq/libpq-fs.h>
```

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-
pointer -fno-exceptions -I/usr/include/openssl" \
./configure \
--prefix=/usr \
--with-config-file-path=/etc/httpd \
--with-imap \
--enable-safe-mode \
--with-exec-dir=/usr/bin \
--disable-debug \
--with-apache=../apache_1.3.9 \
--with-pgsql \ (if you want database PostgreSQL)
--with-ldap \ (if you want LDAP database light directory)
--enable-memory-limit=yes \
--enable-debug=no

[root@deep]# make
[root@deep]# make install
```

将 mod_perl 模块加入 Apache 源代码树，重建并安装基于 Perl 的 mod_perl 模块

进入新的 mod_perl 目录(cd ../mod_perl-version) 并在终端输入以下命令：

```
perl Makefile.PL \
EVERYTHING=1 \
APACHE_SRC=../apache_1.3.9/src \
USE_APACI=1 \
PREP_HTTPD=1 \
DO_HTTPD=1

[root@deep]# make
[root@deep]# make instal
```

建立和安装带有 mod_ssl + mm + mod_perl 和 PHP3 模块的 Apache

进入新的 Apache 目录(cd ../apache-version) 并在终端输入以下命令：

```
SSL_BASE=SYSTEM \  (require for mod_ssl).
EAPI_MM=SYSTEM \   (require for mm).
CC="egcs" \
OPTIM="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
```

```
-march=pentiumpro -fomit-frame-pointer
-fno-exceptions" \
CFLAGS="-DDYNAMIC_MODULE_LIMIT=0" \
./configure \
--prefix=/home/httpd \
--bindir=/usr/bin \
--sbindir=/usr/sbin \
--libexecdir=/usr/lib/apache \
--includedir=/usr/include/apache \
--sysconfdir=/etc/httpd/conf \
--localstatedir=/var \
--runtimedir=/var/run \
--logfiledir=/var/log/httpd \
--datadir=/home/httpd \
--proxycachedir=/var/cache/httpd \
--mandir=/usr/man \
--add-module=src/modules/experimental/mod_mmap_static.c \ (if you are intended
to use mod_mmap).
--add-module=src/modules/standard/mod_auth_db.c \ (if you are intended to use
mod_auth).
--enable-module=ssl \ (require for mod_ssl).
--enable-rule=SSL_SDBM \ (require for mod_ssl).
--disable-rule=SSL_COMPAT \ (require for mod_ssl).
--activate-module=src/modules/php3/libphp3.a \ (require for php).
--enable-module=php3 \ (require for php).
--activate-module=src/modules/perl/libperl.a \ (require for mod_perl).
--enable-module=perl \ (require for mod_perl).
--disable-module=include \
--disable-module=status \
--disable-module=userdir \
--disable-module=negotiation \
--disable-module=autoindex \
--disable-module=asis \
--disable-module=imap \
--disable-module=env \
--disable-module=actions
```

这些设置告诉 Apache 让它为当前特定的硬件配置自己进行设置：

- 加入“mod_mmap”模块，以提高性能
- 加入“mod_auth”模块，以加强口令验证的安全性
- 加入“mod_ssl”模块，以进行数据加密和保证数据的安全

- 加入“mod_php3”模块，使得 Apache 支持 PHP
- 加入“mod_perl”模块，使得 Apache 支持 Perl，Perl 比 CGI 脚本有更高的安全性和更好的性能
- 禁止“include”模块
- 禁止“status”模块
- 禁止“userdir”模块
- 禁止“negotiation”模块
- 禁止“autoindex”模块
- 禁止“asis”模块
- 禁止“imap”模块
- 禁止“env”模块
- 禁止“actions”模块

注意：将所有不需要的模块清除会提高 Apache Web 服务器的性能。

```
[root@deep]# make
[root@deep]# make install
[root@deep]# rm -f /usr/sbin/apachectl
[root@deep]# rm -f /usr/man/man8/apachectl.8
[root@deep]# rm -rf /home/httpd/icons/
[root@deep]# rm -rf /home/httpd/htdocs/
[root@deep]# cd /var/tmp/php-version
[root@deep]# install -m 644 php3.ini.dist /etc/httpd/php.ini
[root@deep]# rm -rf /etc/httpd/conf/ssl.crl/
[root@deep]# rm -rf /etc/httpd/conf/ssl.crt/
[root@deep]# rm -rf /etc/httpd/conf/ssl.csr/
[root@deep]# rm -rf /etc/httpd/conf/ssl.key/
[root@deep]# rm -rf /etc/httpd/conf/ssl.prm/
[root@deep]# rm -f /etc/httpd/conf/srm.conf srm.conf.default access.conf
access.conf.default
```

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf apache-version/ apache-version.tar.gz
mod_ssl-version-version/ mod_ssl-version-version.
```

```
tar.gz php-version/ php-version.tar.gz mod_perl-version/
mod_perl-version.tar.gz
```

配置

基于你的目的和网络结构，不同的服务的配置文件可能有很大差别，有些人可能安装了 Apache 用来显示 Web 页面，有些人则连接数据库或者通过 ssl 实现电子商务等等。在本书中，我们提供一个“httpd.conf”文件，设置了 php，perl，ssl，ldap 和密码确认以显示不同的可能性。

我们集中精力在文件的优化和安全上，其他的随你自己的兴趣修改。所以你需要阅读随软件而来的文档以理解那些参数。

在本文“Linuxsos.pdf”中描述的所有软件都有一个独立的目录和子目录，文件（floppy.tgz）以 tar 的格式提供，包含了特定系统的配置文件。如果你得到了这一文件，就不需要被迫手工修改和剪切、粘贴这些配置文件。不论你是要手工粘贴或者直接把文件从压缩档案中拷贝出来，你需要自己进行必要的修改，根据你的需要，存放如 Apache 软件的适当位置，该档案位于：
http://pages.infinit.net/lotus1/doc/opti/floppy.tgz

为了运行Apache 服务器，以下文件必须创建或拷贝到服务器的指定目录：

- 将“httpd.conf”文件拷贝到“/etc/httpd/conf/”目录。
- 将“apache”文件拷贝到“/etc/logrotate.d/”目录。
- 将“httpd”文件拷贝到“/etc/rc.d/init.d/”目录。

你可以在 floppy.tgz 档案中找到下面的设置文件，把文件拷贝到适合的位置上，或者直接把本文中下面的部分拷贝、粘贴到相关文件中。

/etc/httpd/conf/httpd.conf 文件的设置

配置“/etc/httpd/conf/httpd.conf”文件，以下的例子是带 SSL 协议的 Apache 服务器的最低配置。

编辑“httpd.conf”（vi /etc/httpd/conf/httpd.conf）并加入：

```
ServerType standalone
ResourceConfig /dev/null
AccessConfig /dev/null
ServerRoot "/etc/httpd"
PidFile /var/run/httpd.pid
Timeout 300
```

```
KeepAlive On
MaxKeepAliveRequests 0
KeepAliveTimeout 15
MinSpareServers 16
MaxSpareServers 64
StartServers 16
MaxClients 512
MaxRequestsPerChild 100000
Port 80

<IfDefine SSL>
Listen 192.168.1.1:80
Listen 192.168.1.1:443
</IfDefine>

User www
Group www
ServerAdmin admin@openarch.com
ServerName www.openarch.com
DocumentRoot "/home/httpd/ona"
Include conf/mmap.conf
<Directory />
Options None
AllowOverride None
Order deny,allow
Deny from all
</Directory>

<Directory /home/httpd>
Options None
AllowOverride None
Order allow,deny
Allow from all
</Directory>

<Files .pl>
Options None
AllowOverride None
Order deny,allow
Deny from all
</Files>

DirectoryIndex index.htm index.php index.html default.html index.cgi
UseCanonicalName On
```

```
TypesConfig /etc/httpd/conf/mime.types
DefaultType text/plain
HostnameLookups Off

ErrorLog /var/log/httpd/error_log
LogLevel warn
LogFormat "%h %v %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
/etc/httpd/conf/httpd.conf
SetEnvIf Request_URI \.gif$ gif-image
CustomLog /var/log/httpd/access_log combined env=!gif-image
ServerSignature Off

ScriptAlias /cgi-bin/ "/home/httpd/cgi-bin/"
<Directory "/home/httpd/cgi-bin">
AllowOverride None
Options None
Order allow,deny
Allow from all
</Directory>

AddType application/x-httpd-php3 .php3
AddType application/x-httpd-php3-source .phps

ErrorDocument 500 "The server made a boo boo.
ErrorDocument 404 http://www.openarch.com/error.htm
ErrorDocument 403 "Access Forbidden -- Go away.
BrowserMatch "Mozilla/2" nokeepalive
BrowserMatch "MSIE 4\.0b2;" nokeepalive downgrade-1.0 force-response-1.0
BrowserMatch "RealPlayer 4\.0" force-response-1.0
BrowserMatch "Java/1\.0" force-response-1.0
BrowserMatch "JDK/1\.0" force-response-1.0

<IfDefine SSL>
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl .crl
</IfDefine>

<IfModule mod_ssl.c>
SSLPassPhraseDialog builtin
SSLSessionCache dbm:/var/run/ssl_scache
SSLSessionCacheTimeout 300
SSLMutex file:/var/run/ssl_mutex
SSLRandomSeed startup builtin
SSLRandomSeed connect builtin
```

```
SSLLog  /var/log/httpd/ssl_engine_log
SSLLogLevel warn
</IfModule>

<IfDefine SSL>
<VirtualHost _default_:443>
DocumentRoot "/home/httpd/ona"
ServerName www.openarch.com
ServerAdmin admin@openarch.com
ErrorLog /var/log/httpd/error_log
SSLEngine on
SSLCipherSuite ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
SSLCertificateFile /etc/ssl/certs/server.crt
SSLCertificateKeyFile /etc/ssl/private/server.key
SSLCACertificatePath /etc/ssl/certs
SSLCACertificateFile /etc/ssl/certs/ca.crt
SSLCARevocationPath /etc/ssl/crl
SSLVerifyClient none
SSLVerifyDepth 10

SSLOptions +ExportCertData +StrictRequire
SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-shutdown
SetEnvIf Request_URI \.gif$ gif-image
CustomLog /var/log/httpd/ssl_request_log \
"%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b" env=!gif-image
</VirtualHost>
</IfDefine>
```

注意：如果你对 Apache 服务器加入了 mod_php3 模块，那么千万不要把下面几行加入“/etc/httpd/conf/httpd.conf”文件：

```
AddType application/x-httpd-php3 .php3
AddType application/x-httpd-php3-source .phps
```

注意：如果你对 Apache 服务器加入了 mod_perl 模块，不要忘记把下面的几行加入“/etc/httpd/conf/httpd.conf”文件，以便从服务器查看不同 perls 模块的状态。

```
<Location /perl-status>
SetHandler perl-script
PerlHandler Apache::Status
Order deny,allow
Deny from all
Allow from 192.168.1.3
</Location>
```

/etc/logrotate.d/apache 文件的配置

配置“/etc/logrotate.d/apache”文件，让系统自动循环更新你的日志文件。

创建“apache”文件（touch /etc/logrotate.d/apache）并加入下面几行：

```
/var/log/httpd/access_log {
missingok
postrotate
/usr/bin/killall -HUP httpd
endscript
}
/var/log/httpd/error_log {
missingok
postrotate
/usr/bin/killall -HUP httpd
endscript
}
/var/log/httpd/ssl_request_log {
missingok
postrotate
/usr/bin/killall -HUP httpd
endscript
}
```

/etc/rc.d/init.d/httpd 脚本文件的配置

配置“/etc/rc.d/init.d/httpd”脚本文件以启动和停止 Apache 服务器。

创建“httpd”脚本文件（touch /etc/rc.d/init.d/httpd）并加入以下以下几行：

```
#!/bin/sh
#
# Startup script for the Apache Web Server
#
# chkconfig: 345 85 15
# description: Apache is a World Wide Web server. It is used to serve \
# HTML files and CGI.
# processname: httpd
# pidfile: /var/run/httpd.pid
```

```
# config: /etc/httpd/conf/httpd.conf

# Source function library.
. /etc/rc.d/init.d/functions

# See how we were called.
case "$1" in
start)
echo -n "Starting httpd: "
daemon httpd -DSSL
echo
touch /var/lock/subsys/httpd
;;
stop)
echo -n "Shutting down http: "
killproc httpd
echo
rm -f /var/lock/subsys/httpd
rm -f /var/run/httpd.pid
;;
status)
status httpd
;;
restart)
$0 stop
$0 start
;;
reload)
echo -n "Reloading httpd: "
killproc httpd -HUP
echo
;;
*)
echo "Usage: $0 {start|stop|restart|reload|status}"
exit 1
esac
exit 0
```

设置脚本文件可执行并规定缺省权限：

```
[root@deep]# chmod 700 /etc/rc.d/init.d/httpd
```

用下面的命令给 Apache 建立 rc.d 的链接：

```
[root@deep]# chkconfig --add httpd
```

用下面的命令重新启动 Apache 服务器：

```
[root@deep]# /etc/rc.d/init.d/httpd start
```

注意：“-DSSL”选项会以 SSL 模式启动 Apache。如果你希望以普通模式启动 Apache，在“daemon httpd”附近移去“-DSSL”。

保证 Apache 的安全

有几个重要的配置选项同安全相关。最重要的是 web 服务器以什么用户的身份运行（httpd.conf 文件中的 User 和 Group）。最好让该用户只有很少的特权——一个只能够运行 Web 服务器守护进程的用户。创建这个用户（通常就叫做 www）并确保他对于系统和其他功能只有最小的访问权限。

```
[root@deep]# groupadd -g 80 www
[root@deep]# useradd -g 80 -u 80 www
```

改变你的 Web 服务器上一些重要文件和目录的权限

```
[root@deep]# chmod 511 /usr/sbin/httpd
[root@deep]# chmod 750 /etc/httpd/conf/
[root@deep]# chmod 750 /var/log/httpd/
```

你可以创建一个让其他用户修改的 http 子目录，因为 root 从不在该目录之外执行，并且不应该在那里创建文件。（**例如“/home/httpd/ona”**）。

自动索引

缺省情况下，Apache 一般激活了自动索引目录的选项（httpd.conf 文件中的 IndexOptions）。这意味着如果客户请求访问一个没有索引的目录，服务器会对该目

录下的内容建立索引。通常情况下，这会带来安全问题，因为一般你只希望访问者按照你提供的链接去浏览。

要关闭这个选项，你需要把目录的读权限移走（不是目录里面的文件），也就是说，“**chmod 311**”。取决于目录的所有者，它看起来大概是下面的样子：

```
[root@deep]# cd /home/httpd/
[root@deep]# chmod 311 ona
[root@deep]# ls –la

d-wx--x--x 13 webmaster webmaster 1024 Jul 28 08:12 ona
```

然后，任何对该把保护目录的访问都会看到类似下面的信息：

```
Forbidden

You don't have permission to access “/ona/” on this server.
```

对 mount 上的文件系统作更多的控制

对 mount 上的文件系统，如：“/chroot”，可以加上一些设置选项，如：nosuid（可以参考下面《在 chroot 环境中运行 Apache》这一小节）。可以在“/etc/fstab”文件中进行设置：

编译“**fstab**”文件（vi /etc/fstab）并按需要加入内容：

```
/dev/sda7 /chroot ext2 nosuid 1 2
```

“-<nosuid>”意味着不能在指定的放置 Apache 的分区上设置“set-user-identifier”或者“set-group-identifier”，这会消除设置 SUID/SGID 的可能性。

为验证用户创建 .dbmpasswd 密码文件

只在你需要对用户进行身份验证时这样做。

```
[root@deep]# chmod 750 /usr/bin/dbmmanage
[root@deep]# /usr/bin/dbmmanage /etc/httpd/.dbmpasswd adduser username
```

“-<.dbmpasswd>”是密码文件的名字。<username>是你要在“.dbmpasswd”文件中增加的用户的名字。

保护重要的配置文件

“不可改变位”可以防止配置文件被删除或覆盖。同时，它也放置某些人对文件建立链接。一旦你配置好了“httpd.conf”最好象下面一样利用“不可改变位”对文件加以保护：

```
[root@deep]# chattr +i /etc/httpd/conf/httpd.conf
```

在 chroot 监狱中运行 Apache

这一部分讨论如何防止其他人通过 Apache 来突破系统。Apache 缺省情况下以**非 root 用户**的身份运行，不象有 shell 的其他用户，这将降低对系统可能造成的损害。当然，大多数 Apache 服务器允许用户以匿名的身份访问，这很缺乏安全保障。所以这里需要采取一些特殊的步骤– 也就是，**在 chroot 监狱中运行 Apache**。

chroot 监狱最大的好处是限制（httpd）守护进程只能访问到监狱的根而不是文件系统的根。另外，由于监狱只需要支持 Apache，可以限制在监狱中放置的程序。最重要的是，不需要 setuid 程序，这样的程序常常象 bug 一样被用来获取 root 权限并突破监狱的控制。

给 Apache 加入“chroot”的功能其配置工作比较复杂并可能破坏软件。在我们着手进行前，需要先决定是否值得，所以下面给出一些（但不是全部）优点和缺点。

优点：

- 如果 Apache 被破坏，攻击者应该不能访问到系统的其他部分。确保 Apache 以自己独立的用户/组的身份运行，而不是被过分使用的用户如“nobody”。考虑一下，web 服务器以 nobody 或其他被频繁使用的 UID/GID 的身份运行并被破坏，黑客就可能从 chroot 访问其他以“nobody”身份运行的进程。

- 编写错误的 CGI 脚本可能损害安全，例如允许某些人用 email 发送你的“/etc/passwd”文件，确保这不会发生。

缺点：

- 在 chroot 中你需要额外的库文件。你可以建立硬链接或者直接编译二进制文件。静态二进制文件很受欢迎，因为不管危险多么小，它确实可以降低损害的可能性，防止某些人用危险的库文件替换你的 libc 和其他的共享库。

- 一般来说，你的 web 服务器越时髦，维护其软件的组织结构不变就越困难。例如，如果你使用 Perl/CGI，你就需要在 chroot 内部的适当位置拷贝所需的二进制文件和 perl 库文件，使用 SSL，PHP 和其他功能是也是这样。

下面列出的 chroot 的配置假设你已经为了密码验证而在 Apache 服务器中编译了 mod_ssl、mod_php 和 mod_auth_db 模块。你在 Apache 服务器中已经编译了什么，会决定我们还需要往 chroot 的库目录中拷贝那些库文件。

记住，如果你用 mod_perl 编译了 Apache，还需要把二进制文件和 perl 的库文件拷贝到被 chroot 了的目录。Perl 位于“/usr/lib/perl5”目录，为防止你用到 perl，把这个目录拷贝到“/chroot/httpd/usr/lib/perl5/”。别忘了在拷贝文件之前先在你的 chroot 的结构中创建这个目录（“/chroot/httpd/usr/lib/perl5”）。

检查 httpd 的共享库的一致性，后面我们需要把它们拷贝到 chroot 环境。

```
[root@deep]# ldd /usr/sbin/httpd


libpam.so.0 => /lib/libpam.so.0 (0x40016000)
libm.so.6 => /lib/libm.so.6 (0x4001f000)
libdl.so.2 => /lib/libdl.so.2 (0x4003b000)
libcrypt.so.1 => /lib/libcrypt.so.1 (0x4003e000)
libnsl.so.1 => /lib/libnsl.so.1 (0x4006b000)
libresolv.so.2 => /lib/libresolv.so.2 (0x40081000)
libdb.so.3 => /lib/libdb.so.3 (0x40090000)
libc.so.6 => /lib/libc.so.6 (0x400cb000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

注意一下上面的文件，后面会用得着。

步骤一：

如果还没有做的话，为运行 httpd 创建一个用户 id 和组 id。如果你以 root 的身份运行 Apache 的话，监狱的作用就没有了；如果你用一个已经在系统中存在的用户运行 httpd，那么 httpd 的权限可能超过了它实际需要的程度，会带来一些安全隐患。

下面是用户和组的 id 号的例子，检查“/etc/passwd”和“/etc/group”文件以寻找一个未被使用的 uid/gid 号。我们这里使用 80：

```
[root@deep]# groupadd -g 80 www
[root@deep]# useradd -g 80 -u 80 www
```

步骤二：

建立 chroot 环境。首先，我们需要创建 Apache 的 chroot 结构。我们为 chroot 的 Apache“/chroot/httpd”。“/chroot/httpd”只是我们为了更多的安全控制而在另一个分区建立的目录。

```
[root@deep]# /etc/rc.d/init.d/httpd stop ← if Apache is already installed and
run on your system.
[root@deep]# mkdir /chroot/httpd
```

接下来，象下面一样创建剩下的库文件：

```
[root@deep]# mkdir /chroot/httpd/dev
[root@deep]# mkdir /chroot/httpd/lib
[root@deep]# mkdir /chroot/httpd/etc
[root@deep]# mkdir -p /chroot/httpd/usr/sbin
[root@deep]# mkdir -p /chroot/httpd/var/run
[root@deep]# mkdir -p /chroot/httpd/var/log/httpd
[root@deep]# chmod 750 /chroot/httpd/var/log/httpd/
[root@deep]# mkdir -p /chroot/httpd/home/httpd
```

拷贝主配置目录，配置文件，cgi-bin 目录，根目录和 httpd 程序：

```
[root@deep]# cp -r /etc/httpd /chroot/httpd/etc/
[root@deep]# cp -r /home/httpd/cgi-bin /chroot/httpd/home/httpd/
[root@deep]# cp -r /home/httpd/your-DocumentRoot /chroot/httpd/home/httpd/
[root@deep]# mknod /chroot/httpd/dev/null c 1 3
[root@deep]# chmod 666 /chroot/httpd/dev/null
[root@deep]# cp /usr/sbin/httpd /chroot/httpd/usr/sbin/
[root@deep]# cp -r /etc/ssl /chroot/httpd/etc/ ← require only if you use mod_ssl.
[root@deep]# chmod 600 /chroot/httpd/etc/ssl/certs/ca.crt
[root@deep]# chmod 600 /chroot/httpd//etc/ssl/certs/server.crt
[root@deep]# chmod 600 /chroot/httpd/etc/ssl/private/ca.key
[root@deep]# chmod 600 /chroot/httpd/etc/ssl/private/server.key
```

因为从 chroot 的角度看，我们在根上运行，所以我们需要创建“/chroot/httpd/etc”、“/chroot/httpd/dev”、“/chroot/httpd/lib”、“/chroot/httpd/usr/sbin”、“/chroot/httpd/var/run”、“/chroot/httpd/home/httpd”和“/chroot/httpd/var/log/httpd”这些目录。

因为我们用共享库编译的 Apache，所以我们需要把它们安装到 chroot 的目录结构里面。用“ldd /chroot/httpd/usr/sbin/httpd”来检查都需要哪些库文件。检查结果取决于我们在 Apache 中编译了哪些模块，大概是下面的样子：

```
libpam.so.0 => /lib/libpam.so.0 (0x40016000)
libm.so.6 => /lib/libm.so.6 (0x4001f000)
libdl.so.2 => /lib/libdl.so.2 (0x4003b000)
libcrypt.so.1 => /lib/libcrypt.so.1 (0x4003e000)
libnsl.so.1 => /lib/libnsl.so.1 (0x4006b000)
libresolv.so.2 => /lib/libresolv.so.2 (0x40081000)
libdb.so.3 => /lib/libdb.so.3 (0x40090000)
libc.so.6 => /lib/libc.so.6 (0x400cb000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

拷贝上面那些共享库：

```
[root@deep]# cp /lib/libpam.so.0 /chroot/httpd/lib/
[root@deep]# cp /lib/libm.so.6 /chroot/httpd/lib/
[root@deep]# cp /lib/libdl.so.2 /chroot/httpd/lib/
[root@deep]# cp /lib/libcrypt.so.1 /chroot/httpd/lib/
[root@deep]# cp /lib/libnsl* /chroot/httpd/lib/
[root@deep]# cp /lib/libresolv* /chroot/httpd/lib/
[root@deep]# cp /lib/libdb.so.3 /chroot/httpd/lib/
[root@deep]# cp /lib/libc.so.6 /chroot/httpd/lib/
[root@deep]# cp /lib/ld-linux.so.2 /chroot/httpd/lib/
```

为了某些网络功能如地址解析你还需要下面一些额外的库文件：

```
[root@deep]# cp /lib/libnss_compat* /chroot/httpd/lib/
[root@deep]# cp /lib/libnss_dns* /chroot/httpd/lib/
[root@deep]# cp /lib/libnss_files* /chroot/httpd/lib/
```

在被 chroot 的目录里（“/chroot/httpd/etc”）拷贝你的 passwd 和 group 文件。这里的概念同 ftpd 如何使用 passwd 和 group 文件一样。

```
[root@deep]# cp /etc/passwd /chroot/httpd/etc/
[root@deep]# cp /etc/group /chroot/httpd/etc/
```

接下来，在两个文件中（passwd 和 group）把和运行 apache 的用户无关的项全部删除。你还需要“/etc/resolv.conf”、“/etc/nsswitch.conf”和“/etc/hosts”文件。

编辑“passwd”文件（vi /chroot/httpd/etc/passwd）并把运行 apache 的用户外的其他项删除（在我们的设置中是“www”）：

设置“passwd”文件的“不可改变位”：

```
[root@deep]# cd /chroot/httpd/etc/
[root@deep]# chattr +i passwd
```

编辑“group”文件（vi /chroot/httpd/etc/group）并把运行 apache 的用户外的其他项删除（在我们的设置中是“www”）：

对“group”文件设置“不可改变位”：

```
[root@deep]# cd /chroot/httpd/etc/
[root@deep]# chattr +i group
```

对文件“httpd.conf”设置“不可改变位”：

```
[root@deep]# cd /chroot/httpd/etc/httpd/conf/
[root@deep]# chattr +i httpd.conf
```

通过设置“不可改变位”，文件不能被删除或改名，不能被链接，不能被写入内容，只有超级用户可以清除这一属性。

```
[root@deep]# cp /etc/resolv.conf /chroot/httpd/etc/
[root@deep]# cp /etc/hosts /chroot/httpd/etc/
[root@deep]# cp /etc/nsswitch.conf /chroot/httpd/etc/
```

对“resolv.conf”文件设置“不可改变位”：

```
[root@deep]# cd /chroot/httpd/etc/
[root@deep]# chattr +i resolv.conf
```

对“hosts”文件设置“不可改变位”：

```
[root@deep]# cd /chroot/httpd/etc/
[root@deep]# chattr +i hosts
```

对“nsswitch.conf”文件设置“不可改变位”：

```
[root@deep]# cd /chroot/httpd/etc/
[root@deep]# chattr +i nsswitch.conf
```

把“ocaltime”文件拷贝到监狱中，以便日志项可以按照你的本地时区进行调整：

```
[root@deep]# cp /etc/localtime /chroot/httpd/etc/
```

删除旧的文件和目录：

```
[root@deep]# rm -rf /var/log/httpd/
[root@deep]# rm -rf /etc/httpd/
[root@deep]# rm -rf /home/httpd/
[root@deep]# rm -f /usr/sbin/httpd
```

步骤三：

通知 syslogd 新的 chroot 服务。

通常，进程通过“/dev/log”同 syslogd 通讯。由于设置了 chroot 监狱，这样的通讯已经不可能了，所以需要通知 syslogd 去监听“/chroot/httpd/dev/log”。我们需要编辑 syslog 启动脚本以通知 syslogd 监听额外的地址：

编辑“syslog”脚本（vi /etc/rc.d/init.d/syslog）修改下面的行：

```
daemon syslogd -m 0
```

改为：

```
daemon syslogd -m 0 -a /chroot/httpd/dev/log
```

步骤四：

编辑“httpd”脚本（vi /etc/rc.d/init.d/httpd）修改下面的行：

```
daemon httpd
```

改为：

```
/usr/sbin/chroot /chroot/httpd/ /usr/sbin/httpd -DSSL
```

```
rm -f /var/run/httpd.pid
```

改为：

```
rm -f /chroot/httpd/var/run/httpd.pid
```

Red Hat 的“init”脚本守护进程不允许指定交替的 PID 文件，但是由于“httpd”启动脚本在 chroot 环境之外运行，所以它的启动和停止不受影响。

步骤五：

测试 chroot 的设置！重新启动 syslogd：

```
[root@deep]# /etc/rc.d/init.d/syslog stop
[root@deep]# /etc/rc.d/init.d/syslog start
```

现在，重新启动被 chroot 了的 Apache：

哇殴，我们终于完成了！试验一下，“**/etc/rc.d/init.d/httpd start**”。如果你没有收到错误信息，输入“**ps auwx | grep httpd**”看看 httpd 是否在运行。 如果在运行，我们挑选一个 httpd 的进程号来确认它被 chroot 了“**ls -la /proc/that_process_number/root/**”。如果你看到：

```
dev
etc
home
lib
usr
var
```

恭喜！

如上所述，如果你用了 Perl，你需要把系统库、perl 库、“/usr/lib/perl5”和其他库拷贝或建立硬链接到被 chroot 的区域，这同样适用于使用 SSL、PHP 等情况。

配置新的 /etc/logrotate.d/apache 文件

现在，Apache 日志位于“/chroot/var/log/httpd”目录而不再是“/var/log/httpd”，所以我们需要修改“/etc/logrotate.d/httpd”文件以指向新的被 chroot 了的目录。我们还需要用 mod_ssl 编译 Apache，增加一行以运行日志循环程序循环记录 ssl_request_log。设置你的“/etc/logrotate.d/apache”每周自动记录你的日志。

创建“**apache**”文件（touch /etc/logrotate.d/apache）并加入：

```
/chroot/httpd/var/log/httpd/access_log {
missingok
postrotate
/usr/bin/killall -HUP /chroot/httpd/usr/sbin/httpd
endscript
}
```

```
/chroot/httpd/var/log/httpd/error_log {
missingok
postrotate
/usr/bin/killall -HUP /chroot/httpd/usr/sbin/httpd
endscript
}

/chroot/httpd/var/log/httpd/ssl_request_log {
missingok
postrotate
/usr/bin/killall -HUP /chroot/httpd/usr/sbin/httpd
endscript
}

/chroot/httpd/var/log/httpd/ssl_engine_log {
missingok
postrotate
/usr/bin/killall -HUP /chroot/httpd/usr/sbin/httpd
endscript
}
```

优化 Apache

静态文件

对于静态文件，把“**mod_mmap_static**”（如果你按照我在上面编译时间描述的做，那么你已经做好了——增加模块-../mod_mmap_static.c）编译进 Apache（访问 http://www.apache.org/docs/mod/mod_mmap_static.html）并设置 Apache 对静态文档使用内存映象，例如以 root 身份创建一个配置文件：

```
[root@deep]# find /home/httpd/htdocs -type f -print | sed -e 's/.*/mmapfile &/'
> /etc/httpd/conf/mmap.conf
```

并象下面一样在你的 Apache 配置文件中包含“mmap.conf”：

```
[root@deep]# vi /etc/httpd/conf/httpd.conf
```

在“httpd.conf”文件中加入这一行：

```
Include conf/mmap.conf
```

noatime

Linux 有一个文件系统 mount 的选项叫 **noatime**。这一选项可以在“/etc/fstab”文件中加在 mount 选项中。当以这样的选项 mount 文件系统时，对文件的读取不会更新文件属性中的 atime 信息。由于这个信息（atime）没有什么用，所以缺少 atime 的更新没多大关系。

noatime 设置的重要性是消除了文件系统对文件的写操作，文件只是简单地被系统读取。由于写操作相对读来说要更消耗系统资源，所以这样设置可以明显提高服务器的性能。注意 wtime 信息仍然有效，任何时候文件被写，该信息仍被更新。

编辑 **fstab** 文件（vi /etc/fstab） 并加入 noatime 选项：

```
/dev/sda7 /chroot ext2 nosuid,nodev,noatime 1 2
```

假定“/dev/sda7”是你放置“/chroot”的分区。

重启你的系统并用下面的命令进行测试：

```
[root@deep]# reboot
[root@deep]# cat /proc/mounts
```

ip_local_port_range 参数

这里的信息对“/proc/sys/net/ipv4/ip_local_port_range”中的 sysctl 文件和 Linux 核心 2.2.ip_local_port_range –2 INTEGERS 有效。

定义被 TCP 和 UDP 使用的端口范围从本地端口中选择。第一个数字是第一个端口，第二个数字是最后一个端口，对于高负荷的系统，把端口移到 32768-61000。

```
echo “32768 61000” > /proc/sys/net/ipv4/ip_local_port_range
```

把上面的命令加入“/etc/rc.d/rc.local”文件，下次重起系统你就不需要再输入这些命令了。

安装到系统中的文件

```
> /etc/rc.d/init.d/httpd
> /etc/rc.d/rc0.d/K15httpd
> /etc/rc.d/rc1.d/K15httpd
> /etc/rc.d/rc2.d/K15httpd
> /etc/rc.d/rc3.d/S85httpd
> /etc/rc.d/rc4.d/S85httpd
> /etc/rc.d/rc5.d/S85httpd
> /etc/rc.d/rc6.d/K15httpd
> /etc/logrotate.d/apache
> /etc/httpd
> /etc/httpd/conf
> /etc/httpd/conf/httpd.conf.default
> /etc/httpd/conf/httpd.conf
> /etc/httpd/conf/mime.types.default
> /etc/httpd/conf/mime.types
> /etc/httpd/conf/magic.default
> /etc/httpd/conf/magic
> /etc/httpd/php.ini
> /home/httpd
> /usr/include/apache/ap_md5.h
> /usr/include/apache/ap_mm.h
> /usr/include/apache/ap_mmn.h
> /usr/include/apache/ap_sha1.h
> /usr/include/apache/buff.h
> /usr/include/apache/compat.h
> /usr/include/apache/conf.h
> /usr/include/apache/explain.h
> /usr/include/apache/fnmatch.h
> /usr/include/apache/hsregex.h
> /usr/include/apache/http_conf_globals.h
> /usr/include/apache/http_config.h
> /usr/include/apache/http_core.h
> /usr/include/apache/http_log.h
> /usr/include/apache/http_main.h
> /usr/include/apache/http_protocol.h
> /usr/include/apache/http_request.h
> /usr/include/apache/http_vhost.h
> /usr/include/apache/httpd.h
> /home/httpd/cgi-bin
> /home/httpd/cgi-bin/printenv
> /home/httpd/cgi-bin/test-cgi
```

```
> /usr/bin/htpasswd
> /usr/bin/htdigest
> /usr/bin/dbmmanage
> /usr/include/apache
> /usr/include/apache/xml
> /usr/include/apache/xml/asciitab.h
> /usr/include/apache/xml/hashtable.h
> /usr/include/apache/xml/iasciitab.h
> /usr/include/apache/xml/latin1tab.h
> /usr/include/apache/xml/nametab.h
> /usr/include/apache/xml/utf8tab.h
> /usr/include/apache/xml/xmldef.h
> /usr/include/apache/xml/xmlparse.h
> /usr/include/apache/xml/xmlrole.h
> /usr/include/apache/xml/xmltok.h
> /usr/include/apache/xml/xmltok_impl.h
> /usr/include/apache/alloc.h
> /usr/include/apache/ap.h
> /usr/include/apache/ap_compat.h
> /usr/include/apache/ap_config.h
> /usr/include/apache/ap_config_auto.h
> /usr/include/apache/ap_ctx.h
> /usr/include/apache/ap_ctype.h
> /usr/include/apache/ap_hook.h
> /usr/include/apache/multithread.h
> /usr/include/apache/rfc1413.h
> /usr/include/apache/scoreboard.h
> /usr/include/apache/util_date.h
> /usr/include/apache/util_md5.h
> /usr/include/apache/util_script.h
> /usr/include/apache/util_uri.h
> /usr/include/apache/os.h
> /usr/include/apache/os-inline.c
> /usr/lib/apache
> /usr/man/man1/htpasswd.1
> /usr/man/man1/htdigest.1
> /usr/man/man1/dbmmanage.1
> /usr/man/man8/ab.8
> /usr/man/man8/httpd.8
> /usr/man/man8/logresolve.8
> /usr/man/man8/rotatelogs.8
> /usr/man/man8/apxs.8
> /usr/sbin/httpd
> /usr/sbin/ab
```

```
> /usr/sbin/logresolve
> /usr/sbin/rotatelogs
> /usr/sbin/apxs
> /var/log/httpd
> /var/cache
> /var/cache/httpd
```

可选的 Apache 部件

Devel-Symdump

Perl 的模块 Devel::Symdump 提供一个很好的方式在运行的程序中检查 perl 的符号表和类的结构层次。从 2.00 版本开始，这个模块至少需要 perl5.003。要建立和安装该模块，请遵循下面的步骤：

软件包：

Devel-Symdump 主页： http://www.perl.com/CPAN/modules/by-module/Devel/。

```
[root@deep]# cp Devel-Symdump-version.tar.gz /var/tmp/
[root@deep]# tar xzpf Devel-Symdump-version.tar.gz
```

进入新的 devel-Symdump 目录并在你的终端输入下面的命令：

```
[root@deep]# perl Makefile.PL
[root@deep]# make
[root@deep]# make test
[root@deep]# make install
```

事后的清除：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf devel-Symdump.version/ Devel-Symdump-version.tar.gz
```

CGI.pm

CGI.pm 是一个很容易使用的 Perl5 的库，用来写 WWW 的 CGI 脚本。该软件的较老的版本 bug 较多，缺省就安装在你的系统中，请将该软件至少更新到版本 2.51。要安装该模块，请遵循下面的步骤：

软件包：

CGI.pm 主页：http://stein.cshl.org/WWW/software/CGI/cgi_docs.html。

```
[root@deep]# cp CGI.pm.tar.gz /var/tmp/
[root@deep]# tar xzpf CGI.pm.tar.gz
```

进入新的 CGI.pm 目录并在你的终端输入下面的命令：

```
[root@deep]# perl Makefile.PL
[root@deep]# make
[root@deep]# make test
[root@deep]# make install
```

事后的清除：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf CGI.pm/ CGI.pm.tar.gz
```

FormMail

FormMail 是一个全球性的 WWW 表单到电子邮件的网关。这里只需要一个表单的输入 tag（必须指定以便脚本和你当前的表单共同工作）。要安装该软件，请遵循以下步骤：

软件包：

Formmail 主页: http://www.worldwidemart.com/scripts/formmail.shtml。

```
[root@deep]# cp formmail.tar.gz /var/tmp/
[root@deep]# tar xzpf formmail.tar.gz
```

进入新的 Formmail 目录并在你的终端输入下面的命令：

```
[root@deep]# vi Formmail.pl
```

```
@referers = ('openarch.com','192.168.1.1');
```

这个数组允许你定义域，这些域可以保存表单并使用你的 FormMail 脚本。

“FormMail.pl”脚本需要放在你的服务器的“cgi-bin”目录里，并且 www 用户必须具有读和执行的权限。

```
[root@deep]# cp Formmail.pl /home/httpd/cgi-bin/ (你的放置 CGI 的目录).
```

进入“cgi-bin”目录

```
[root@deep]# chmod 750 Formmail.pl
[root@deep]# chown 0.99 Formmail.pl
```

事后的清除：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf formmail/ formmail.tar.gz
```

Webalizer

Webalizer 是一个 web 服务器的日志文件分析程序，以 HTML 的格式输出服务器的使用统计结果。结果同时具有表格和图形的方式，适于分析。

软件包：

Webalizer 主页： http://www.mrunix.net/webalizer/。

```
[root@deep]# cp webalizer-version-src.tgz /var/tmp/
[root@deep]# tar xzpf webalizer-version-src.tgz
```

Webalizer 需要 Tom Boutell 创建的 GD 图形库。如果你还没有该图形库，从 Red Hat 的 Linux 6.1 光盘上安装它：

```
[root@deep]# rpm -Uvh gd-devel-version.i386.rpm
```

进入新的 Webalizer 目录并在你的终端输入下面的命令：

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-pointer -fno-exceptions" \
./configure \
--prefix=/usr


[root@deep]# make
[root@deep]# make install
[root@deep]# mkdir /home/httpd/usage
```

在你的“httpd.conf”文件中加入下面的行：

```
Alias /usage/ "/home/httpd/usage/"
<Directory "/home/httpd/usage">
Order deny,allow
Deny from all
Allow from 192.168.1.3
</Directory>
```

事后清除：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf webalizer-version/ webalizer-version-src.tgz
```

FAQ-O-Matic

Faq-O-Matic 是一个基于 CGI 的系统，可以自动维护 FAQ（或称为经常问及的问题列表，Frequently Asked Questions list）。它允许访问者参加并更新你的 FAQ。要安装该程序，请遵循下面的步骤：

软件包：

FAQ-O-Matic 主页： http://www.dartmouth.edu/~jonh/ff-serve/cache/1.html。

```
[root@deep]# cp FAQ-O-Matic-version.tar.gz /var/tmp/
[root@deep]# tar xzpf FAQ-O-Matic-version.tar.gz
```

在安装“FAQ-O-Matic”前首先更新你的“CGI.pm”程序至少到版本 2.51 （见上），进入新的 FAQ-O-Matic 目录并在你的终端输入下面的命令：

```
[root@deep]# perl Makefile.PL
[root@deep]# make
[root@deep]# make install
[root@deep]# mv fom /home/httpd/cgi-bin/ (or wherever your CGIs live).
[root@deep]# mkdir -p /home/httpd/cgi-bin/fom-meta
[root@deep]# mkdir -p /home/httpd/faqomatic
[root@deep]# chown 0.www /home/httpd/cgi-bin/fom
[root@deep]# chown -R www.www /home/httpd/cgi-bin/fom-meta/
[root@deep]# chown -R www.www /home/httpd/faqomatic/
```

在你的“httpd.conf”文件中加入下面的行：

```
Alias /faqomatic/ "/home/httpd/faqomatic/"
<Directory "/home/httpd/faqomatic">
Order allow,deny
Allow from all
</Directory>

Alias /bags/ "/home/httpd/faqomatic/bags/"
<Directory "/home/httpd/faqomatic/bags">
Order allow,deny
Allow from all
</Directory>

Alias /cache/ "/home/httpd/faqomatic/cache/"
<Directory "/home/httpd/faqomatic/cache">
Order allow,deny
Allow from all
</Directory>

Alias /item/ "/home/httpd/faqomatic/item/"
<Directory "/home/httpd/faqomatic/item">
Order allow,deny
```

```
Allow from all
</Directory>
```

输入下面的命令重启你的 web 服务器：

```
[root@deep]# /etc/rc.d/init.d/httpd restart
```

```
Netscape http://localhost/cgi-bin/fom (或任何你喜欢的浏览器). (或任何 URL).
输入你的临时密码
创建 fom-meta 目录
首先点击"Define configuration parameters"并设置
如：在"Mandatory: Server directory configuration"下设置下列参数
$serverBase= http://www.openarch.com
$cgiURL= /cgi-bin/fom
$serveDir= /home/httpd/faqomatic/
$serveURL= /faqomatic/
```

配置其它你所需要的参数。

事后清除：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf FAQ-OMatic-version/ FAQ-OMatic-version.tar.gz
```

Webmail IMP

IMP 是一个 IMAP 的 Web 邮件 mail 客户。要安装该软件，请参考下面的步骤：

软件包：

Webmail IMP 主页：http://www.horde.org/imp/。

```
[root@deep]# cp horde-version.tar.gz /home/httpd/
[root@deep]# tar xzpf horde-version.tar.gz
[root@deep]# mv horde-version horde
[root@deep]# cp imp-version.tar.gz /home/httpd/horde/
```

转入“horde”目录(cd /home/httpd/horde/), 并 untar/gzip imp-version.tar.gz

```
[root@deep]# tar xzpf imp-version.tar.gz
[root@deep]# rm -f imp-version.tar.gz
[root@deep]# mv imp-version imp
```

在你的“httpd.conf”文件中加入下面的行：

```
Alias /horde/ "/home/httpd/horde/"
<Directory "/home/httpd/horde">
Order allow,deny
Allow from all
</Directory>

Alias /imp/ "/home/httpd/horde/imp/"
<Directory "/home/httpd/horde/imp">
Options None
Order allow,deny
Allow from all
</Directory>
```

用下面的命令重起你的 web 服务器：

```
[root@deep]# /etc/rc.d/init.d/httpd restart
```

现在启动叫做“setup.php3”的引擎，赋予用户通过 web 配置 IMP 的能力。考虑到安全，缺省是不允许的，但是你可以再激活它：

进入“horde”目录（cd /home/httpd/horde/） 并在终端输入下面的命令：

```
[root@deep]# sh ./install.sh
```

你应该可以把浏览器指向“http:// <your imp server>/<your horde home>/setup.php3”。在这里你可以进入图形化的设置程序并可以配置你的 IMP。

当你配置完毕后记得关闭该权限。

进入新的“horde”目录（cd /home/httpd/horde/）并在你的终端输入下面的命令：

```
[root@deep]# sh ./secure.sh
```

事后清除：

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf horde-version.tar.gz
```

Linux IPX Netware™客户

概述

IPX 是 Novell 公司使用的一种协议，为他们的产品 NetWare™提供网络互联支持。

ncpfs 是能够支持 Novell NetWare™ NCP 协议的文件系统。NCP 对于 NetWare 就像 NFS 对于 TCP/IP 那样。为了让 Linux 系统能够 mount 上 NetWare 的文件系统，需要特殊的 mount 程序。ncps 软件包就有这样的 mount 程序，还有其它用来配置和使用 ncpfs 文件系统的工具。可以用命令“rpm –q”可以检查一下在系统中是否已经安装了 ncpfs。

ipxutils 软件包包括一些在 Linux 下配置和调试 IPX 界面和网络的工具（ipx_configure、ipx_internal_net、ipx_interface 和 ipx_route）。IPX 是 Novell NetWare 文件服务器用来传输数据的底层协议。在系统中必须安装 ipxutils，用“rpm –q ipxutils”检查。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

ncpfs 的版本是 2.2.0.12。

ipxutils 的版本是 2.2..0.12。

编译一个支持 IPX 和 NCP 的内核

首先要保证内核支持 IPX 和 NCP 协议。在编译内核的时候，下面的这些问题都要回答“y”：

```
The IPX protocol (CONFIG_IPX) [N] Y
NCP filesystem support (CONFIG_NCP_PS) [N] Y
Packet signatures (CONFIG_NCPFS_PACKET_SIGNING) [N] Y
Clear remove/delete inhibit when needed (CONFIG_NCPFS_STRONG) [N] Y
```

```
Use NFS namespace if available (CONFIG_NCPFS_NFS_NS) [N] Y
Use LONG (OS/2) namespace if available (CONFIG_NCPFS_OS2_NS) [No] Y
Allow mounting of volume subdirectories (CONFIG_NCPFS_MOUNT_SUBDIR) [N] Y
Enable symbolic links and execute flags (CONFIG_NCPFS_EXTRAS) [N] Y
```

尝试建立只用 IPX 协议的网络

一个活动的网络界面可以不绑定任何协议。为了不用“ifconfig”命令设置 IP 地址，可以简单地用“ifconfig ethN up”这个命令。

假定想要把 eth1 设置成只支持 IPX 而不支持 TCP/IP，先要检查一下“ifcfg-eth1”文件是否存在并且确定 IPADDR、NETMASK、NETWORK、BROADCAST 的值为空：

```
[root@deep]# vi /etc/sysconfig/network-scripts/ifcfg-eth1
DEVICE=eth1
IPADDR=
NETMASK=
BROADCAST=
ONBOOT=no
BOOTPROTO=none
USERCTL=no
```

然后，用下面的命令重新启动 network daemon：

```
/etc/rc.d/init.d/network restart
```

接着用下面的命令使以太网卡 eth1 有效：

```
ifconfig eth1 up
```

因为在默认情况下 Linux 会在 eth0 和 eth1 两个网络界面上配置 IPX 协议，把第一块找到的网卡（eth0）设置程 IPX 主界面，因为我们只想在第二块网卡（eth1）上配置 IPX 协议，所以要把 eth0 上的 IPX 协议禁止掉。可以使用下面的命令：

```
ipx_interface del eth0 802.3
```

现在再用 ifconfig，就可以看到网络界面的配置情况：eth0 只支持 TCP/IP 协议，eth1 只支持 IPX 协议。

ncpfs 用户命令

用下面的命令自动配置网络界面并设定主界面：

```
[root@deep]# ipx_configure –auto_interface=on –auto_primary=on
```

用下面的命令 mount Novell™服务器：

```
[root@deep]# ncpmount –S DMS01 /mnt/netware –U username –P passwd
```

这个命令可以 mount 上文件服务器 DMS01，用 username 登录名和 passwd 口令，把文件系统 mount 到“/mnt/netware”目录下。如果没有设置“-P”参数，系统就会提示你输入口令。

用下面的命令 umount“/mnt/netware”目录：

```
[root@deep]# ncpumount /mnt/netware
```

用下面的命令列出网络上所有的 Novell 文件服务器：

```
[root@deep]# slist
```

用下面的命令拷贝文件：

```
[root@deep]# ncopy file1 [file2…] directory
```

在“/proc”目录下有一些和 Linux IPX 相关的文件：

/proc/net/ipx_interface

这个文件包含计算机上 IPX 界面的配置信息。这些信息有的是用手工配置的，有的是自动检测到并配置的。

/proc/net/ipx_route

这个文件包含在 IPX 路由表中出现的路由的列表。这些路由可以用命令手工加上，也可以由 IPX 路由 daemon 自动设置。

/proc/net/ipx

这个文件包含正在使用的 IPX socket 的列表。

Linux FTP 服务器

概述

使用文件传输协议（FTP）来通过网络在计算机之间传输文件是很普遍的一种方法。几乎在所有的计算机平台上面都有 FTP 的客户端和服务端的软件，因此用 FTP 来传送文件也是很方便的一个方法。

配置 FTP 服务器有很多不同的方法。其中一种是把 FTP 配置成只对系统中的用户开放的私有服务器，这也是 FTP 的默认配置。一个私有的 FTP 服务器只运行系统中的用户访问，而且可以对用户进行访问控制，这样可以给予或拒绝特定用户的访问权限。

另一种是把 FTP 服务器配置成匿名服务器。匿名 FTP 服务器允许任何人（不管有没有帐号）访问服务器并传输文件。因为可能存在潜在的安全问题，必须小心配置使得只允许访问系统中特定的目录。

在这一节里，我们把 FTP 配置成“chrooted”的方式，这种配置运行用户访问，例如：Web 站点的目录，但是不允许他们访问更高一层的目录。

注意事项

下面所有的命令都是 Unix 兼容的命令。

源路径都为“/var/tmp”（当然在实际情况中也可以用其它路径）。

安装在 RedHat Linux 6.1 下测试通过。

要用“root”用户进行安装。

wu-ftp 的版本号是 2.6.0。

软件包的来源

wu-ftp 的主页：http://www.wu-ftpd.org/。

下载：wu-ftpd-2.6.0.tar.gz

编译和安装

把软件包（tar.gz）解压缩：

```
[root@deep]# cp wu-ftpd-version.tar.gz /var/tmp
[root@deep]# cd /var/tmp
[root@deep]# tar xzpf wu-ftpd-version.tar.gz
```

编译和优化

转到 wu-ftpd 的新目录下，运行下面的命令：

编辑“ftpcount.c”文件（vi +241 src/ftpcount.c），改变下面这一行：

```
#if defined (LINUX)
```

改为：

```
#if defined (LINUX_BUT_NOT_REDHAT_6_0)
```

编辑“pathnames.h.in”文件（vi +42 src/pathnames.h.in），改变下面这一行：

```
#define _PATH_EXECPATH "/bin/ftp-exec"
```

改为：

```
#define _PATH_EXECPATH "/usr/bin/ftp-exec"
```

我们把“ftp-exec”从“/bin”目录改到“/usr/bin”目录下。

先设置编译器的编译参数：

```
CC="egcs" \
CFLAGS="-O9 -funroll-loops -ffast-math -malign-double -mcpu=pentiumpro
-march=pentiumpro -fomit-frame-
pointer -fno-exceptions" \
./configure \
--prefix=/usr \
--sysconfdir=/etc \
--localstatedir=/var \
--disable-dnsretry \
--enable-quota \
```

```
--enable-pam \
--disable-daemon \
--disable-newlines \
--disable-virtual \
--disable-plsm \
--disable-pasvip \
--disable-anonymous \
--enable-ls \
--enable-numericuid
```

这些编译参数告诉编译器如何编译 wu-ftpd：

- 不要用失败的 DNS 查询
- 加入对定额（QUOTA）的支持（如果 OS 支持）
- 加入对 PAM 的支持
- 不允许作为单独的 daemon 运行
- 删除过多的空行
- 不支持虚拟服务器
- 禁止 PID 加锁睡眠消息（用于繁忙的站点）
- 被动连接的时候不要求用同样的 IP
- 不允许匿名 ftp 访问
- 使用内部的“ls”命令（试验性的）
- 内部的“ls”命令显示 UID 而不显示用户名（为了提高速度）

用下面的命令编译和安装软件：

```
make
make install
install -m 755 util/xferstats /usr/sbin
touch /var/log/xferlog
chmod 600 /var/log/xferlog
cd /usr/sbin
ln -sf in.ftpd /usr/sbin/wu.ftpd
ln -sf in.ftpd /usr/sbin/in.wuftpd
strip /usr/bin/ftpcount
strip /usr/bin/ftpwho
strip /usr/sbin/in.ftpd
```

```
strip /usr/sbin/ftpshut
strip /usr/sbin/ckconfig
strip /usr/sbin/ftprestart
```

上面的“make”和“make install”可以配置软件以保证系统中有编译所需要的函数库，然后把所有的源文件都编译成可执行的二进制文件，最后把二进制文件和配置文件安装到相应的目录里。

“install –m”安装“xferstats”程序，用于统计传输了多少文件。“touch”命令为 xferstats 在“/var/log”目录下创建日志文件。“chomod”把“xferlog”的权限改为只对超级用户“root”可读和可写。接着，我们为“in.ftpd”二进制文件创建符号链接。最后，用“strip”命令减小所有二进制文件的大小以提高 wu-ftpd 的性能。

清除不必要的文件

```
[root@deep]# cd /var/tmp
[root@deep]# rm -rf wu-ftpd-version/ wu-ftpd-version.tar.gz
```

“rm”命令删除所有编译和安装 wu-ftpd 所需要的源程序，并且把 wu-ftpd 软件的压缩包删除掉。

为 FTP 站点的用户建立没有 shell 的帐号

首先，创建一个新的用户，这个用户被允许连接到 ftp 服务器上。因为要有“chroot”的环境，这个帐号不同于正常的用户帐号，不能受访问限制。“chroot”使用户产生这样的感觉好像自己已经在文件系统的最顶层了。

第一步

用下面的命令在“/etc/passwd”文件中创建用户。对于每个允许访问 ftp 服务器的新用户都要重复这个步骤。

```
[root@deep]# mkdir /home/ftp
[root@deep]# useradd -d /home/ftp/ftpadmin/ -s /dev/null ftpadmin > /dev/null
2>&1
[root@deep]# passwd ftpadmin


Changing password for user ftpadmin
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully
```

第二步

编辑“/etc/shells”文件并加入一个空 shell，如：null。这个假的 shell 可以限制用户对 ftp 服务器的访问。

```
[root@deep]# vi /etc/shells


/bin/bash
/bin/sh
/bin/ash
/bin/bsh
/bin/tcsh
/bin/csh
/dev/null ← This is our added no existent shell
```

第三步

现在编辑“/etc/passwd”文件，手工加上“/./”把“/home/ftp”目录和“/ftpadmin”目录分开，用户“ftpadmin”会自动转到（chdir）“/ftpadming”目录下。在“passwd”文件中每添加一个 ftp 用户都要重复这个步骤。

编辑“passwd”文件（vi /etc/passwd），把下面这一行改为：

```
ftpadmin:x:502:502::/home/ftp/ftpadmin/:/dev/null
```

改为：

```
ftpadmin:x:502:502::/home/ftp/./ftpadmin/:/dev/null
```

帐号为“ftpadmin”，这这个帐号的家目录有一些奇怪。第一部分“/home/ftp/”表示“chroot”时作为根目录的目录。被点号分开的“/ftpadmin”表示当登录 ftp 服务器的时候会自动转到这个目录。“/dev/null”这个空 shell 不允许“ftpadmin”像正常用户那样登录。经过这些改变，“ftpadmin”用户用的不是真正的 shell 而是伪 shell，这样访问 ftp 服务器就受到限制。

创建一个“chroot”用户环境

先要创建一个简单的根文件系统（root file system），包含有足够的文件，如果二进制程序、口令文件，等等。当用户登录的时候，Unix 就可以改变根文件系统（chroot）。注意一下，如果编译的时候象上面那样加上“--enable-ls”参数，“/home/ftp/bin”和“/home/ftp/lib”两个目录就可以不要了，因为 WU-FTP 会用自己带的“ls”。不过我们还是介绍一下旧的方法，也就是把“/bin/ls”拷贝到

“/home/ftp/bin”（chroot 之后就是“/bin”）目录下，然后把相关的运行库拷贝到“/home/ftp/lib”目录下。

第一步

创建改变根文件系统（chrooted）环境所需要的所有的目录：

```
[root@deep]# mkdir /home/ftp/dev
[root@deep]# mkdir /home/ftp/etc
[root@deep]# mkdir /home/ftp/bin (require only if you are not using the “--enable-ls” option)
[root@deep]# mkdir /home/ftp/lib (require only if you are not using the “--enable-ls” option)
```

第二步

把新目录的权限设成 0511：

```
[root@deep]# chmod 0511 /home/ftp/dev
[root@deep]# chmod 0511 /home/ftp/etc
[root@deep]# chmod 0511 /home/ftp/bin (require only if you are not using the “--enable-ls” option)
[root@deep]# chmod 0511 /home/ftp/lib (require only if you are not using the “--enable-ls” option)
```

上面这些“chmod”命令把 chrooted 之后的“dev”、“etc”、“bin”和“lib”目录设置成超级用户“root”可读、可执行，用户组和所有用户可执行。

第三步

把“/bin/ls”文件拷贝到“/home/ftp/bin”目录下，并把“ls”的权限改为 0111（不运行用户改变这个文件）。

```
[root@deep]# cp /bin/ls /home/ftp/bin (require only if you are not using the “--enable-ls” option)
[root@deep]# chmod 0111 /bin/ls /home/ftp/bin/ls (require only if you are not using the “--enable-ls” option)
```

第四步

找到“ls”程序所需的共享库：

```
[root@deep]# ldd /bin/ls (require only if you are not using the “--enable-ls” option)


libc.so.6 => /lib/libc.so.6 (0x00125000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x00110000)
```

把共享库拷贝到“/home/ftp/lib”目录下：

```
[root@deep]# cp /lib/libc.so.6 /home/ftp/lib/ (require only if you are not using the “--enable-ls” option)
[root@deep]# cp /lib/ld-linux.so.2 /home/ftp/lib/ (require only if you are not using the “--enable-ls” option)
```

注意：如果想用 Linux 的“ls”程序而不是用 WU-ftpd 自带的“ls”（编译时加上“--enable-ls”参数），才需要第三和第四步。

第五步

创建“/home/ftp/dev/null”文件：

```
[root@deep]# mknod /home/ftp/dev/null c 1 3
[root@deep]# chmod 666 /home/ftp/dev/null
```

第六步

把“group”和“passwd”文件拷贝到“/home/ftp/etc”目录下，然后再改变这两个文件。

```
[root@deep]# cp /etc/passwd /home/ftp/etc/
[root@deep]# cp /etc/group /home/ftp/etc/
```

编辑“passwd”文件（vi /home/ftp/etc/passwd）把除了“root”和允许使用 ftp 的用户之外的所有其它项删掉。这对于改变根文件系统的环境很重要，改变之后的“passwd”文件会是象下面这样的：

```
root:x:0:0:root:/:/dev/null
ftpadmin:x:502:502::/ftpadmin/:/dev/null
```

编辑“group”文件（vi /home/ftp/etc/group），把除了“root”和允许使用 ftp 的用户之外的所有其它项删掉。改变之后的“group”文件会是象下面这样的：

```
root:x:0:root
ftpadmin:x:502:
```

配置

可以到这去下载“floppy.tgz”文件：http://pages.infinit.net/lotus1/doc/opti/floppy.tgz。把“floppy.tgz”文件解开之后，可以在相应的目录下发现我们在这本书中介绍的所有软件的配置文件。这样就没有必要手工重新生成这些文件，或者用拷贝粘贴的方法把它们粘贴到配置文件中去。不管是打算自己动手生成配置文件还是拷贝现成的，你都要学会自己修改配置文件并且把配置文件拷贝到正确的目录下。下面将具体说明。

为了运行 FTP 服务器，必须创建或者把下面的文件拷贝到相应的目录下：

- 把“ftpaccess”文件拷贝到“/etc”目录下
- 把“ftpusers”文件拷贝到“/etc”目录下
- 把“ftphosts”文件拷贝到“/etc”目录下
- 把“ftpgroups”文件拷贝到“/etc”目录下
- 把“ftpconversion”文件拷贝到“/etc”目录下
- 把“ftp”文件拷贝到“/etc/pam.d”目录下
- 把“ftpd”文件拷贝到“/etc/logrotate.d”目录下

可以把“floppy.tgz”解压之后，找到上面列出来的文件，并拷贝到相应的目录下，或者用拷贝粘贴的方法从本书中直接粘贴出。

配置 /etc/ftpaccess 文件

“/etc/ftpaccess”文件是用来配置“ftpd”的。这个文件主要是设置允许哪个用户、多少用户访问 ftp 服务器，以及一些安全方面的问题。配置文件的每一行或者定义一个属性或者设定一个属性值，。对于非匿名的“chroot”访问，必须创建一些“guestgroup”，每一个都要对应“/home/ftp/etc/group”文件中的项。

创建 ftpaccess 文件（touch /etc/ftpaccess），在文件中加入：

```
class openarch guest 208.164.186.*
limit openarch 20 MoTuWeTh,Fr0000-1800 /home/ftp/.too_many.msg
email admin@openarch.com
loginfails 3
readme README* login
```

```
readme README* cwd=*
message /home/ftp/.welcome.msg login
message .message cwd=*
compress yes all
tar yes all
chmod yes guest
delete yes guest
overwrite yes guest
rename yes guest
log commands real,guest
log transfers real,guest inbound,outbound
guestgroup ftpadmin
guestgroup webmaster
# We don't want users being able to upload into these areas.
upload /home/ftp/* / no
upload /home/ftp/* /etc no
upload /home/ftp/* /dev no
# We'll prevent downloads with noretrieve.
noretrieve /home/ftp/etc
noretrieve /home/ftp/dev
log security real,guest
guest-root /home/ftp ftpadmin webmaster
restricted-uid ftpadmin webmaster
restricted-gid ftpadmin webmaster
greeting terse
Keepalive yes
noretrieve .notar
```

现在把文件的权限设成 600：

```
[root@deep]# chmod 600 /etc/ftpaccess
```

下面解释配置文件中的设置：

class

“class”用来定义一个允许访问 ftp 服务器的用户类别。可以定义任意多的类别（class）。每一个“class”行的格式如下：

```
class <classname> <typelist> <addrglob>
```

<classname>是 class 的名字，<typelist>是允许加到类别（class）中的用户类型，<addrglob>是这个 class 允许的 IP 地址范围。

<typelist>中的项是用逗号隔开的，每一个项有三种可能的取值：anonymous、guest 或 real。anonymous 用户是用 anonymous 或 ftp 帐号访问 ftp 服务器而且只需要访问公用文件的那些用户。guest 用户有一些特殊因为他们在系统中没有帐号，但是却又是 guest 组的成员。real 用户必须在 FTP 服务器上有帐号，而且需要经过服务器的安全验证。

<addrglob>可以用通配符，例如：“*”表示所有的站点。下面这一行：

```
class openarch guest 208.164.186.*
```

表示只有在系统中有自己帐号的 guest 用户而且 IP 地址为“208.164.186.*”才能访问 ftp 服务器。

limit

“limit”根据 class 和时间范围来限制登录 ftp 服务器的用户数。“limit”的格式是：

```
limit <class> <n> <times> <message_file>
```

其中<class>是受限制的类别（class），<n>表示在这个类别中受到限制的最多用户数，<times>表示“limit”有效的时间段，<message_file>是当达到最大的用户数目的时候，别的用户还想登录时显示的信息。

<times>中的字符串用逗号隔开，每一个字串表示一天。星期一到星期日分别用 Mo、Tu、We、Th、Fr、Sa 和 Su 表示，周末也可以用 Wk 表示。表示时间的小时和分钟之间不要用冒号隔开。“-”用来表示范围。

例如，限制“openarch”这个类别，最多可以有 20 个用户，访问时间是星期一道星期四全天，星期五从半夜到下午六点。用下面这一行来表示：

```
limit openarch 20 MoTuWeTh,Fr0000-1800 /home/ftp/.too_many.msg
```

如果一旦达到最大的用户数，还有别的用户想连接 ftp 服务器，就会把“/home/ftp/.too_mang.msg”中的信息传给用户。

loginfails

“loginfails”设置最多允许的登录失败的次数。可以用下面这一行来表示：

```
loginfails <n>
```

<n>表示最多允许的登录失败的次数。例如，只允许失败两次可以这样表示：

```
loginfails 3
```

readme

“readme”设置在什么条件下，一旦当前目录中的文件发生了变化需要提醒用户。

这个命令的格式为：

```
readme <path> <when>
```

<path>是用来提醒用户的文件的名字（例如：README），<when>设置出现这个提示信息的条件。

<when>可以为下面两种形式：LOGIN 或 CWD=<dir>。如果为 LOGIN，当用户成功登录的时候就会出现提示信息。如果为 CWD=<dir>，当用户进入<dir>目录的时候就会有提示信息。

请记住当给匿名（anonymous）用户设置提示信息的路径的时候，这个路径必须是相对于匿名 ftp 目录。

message

“message”设置当用户登录或转到一个目录的时候会收到什么消息。可以设定多个消息。命令的格式为：

```
message <path> <when>
```

<path>表示需要显示的文件的完整路径名，<when>和“readme”中的<when>意思一样。

还需要注意的是消息文件的路径名也是相对于匿名 ftp 目录的。

例如：

```
message /home/ftp/.welcome.msg LOGIN
```

compress、tar、chmod、delete、overwrite、rename

如果这些都不设置，那么就使用默认值，也就是对所有人都是“yes”。下面例子中的设置的意思是给 guest 组 chmod、delete、overwrite 和 rename 文件的权力，所有人都可以使用 compress 和 tar。

例如：

```
compress   yes    all
tar        yes    all
chmod      yes    guest
delete     yes    guest
overwrite  yes    guest
rename     yes    guest
```

log commands

因为安全上的原因需要记录用户使用的每一个命令。“log commands”的格式是：

```
log commands <typelist>
```

<typelist>是用逗号隔开的字串，表示哪些用户的命令需要记录下来，字串的取值可以是：anonymous、guest 或 real。

例如：要记录 real 和 guest 用户的每一个命令，可以这么表示：

```
log commands real,guest
```

这些记录都保存在“/var/log/message”文件中。

log transfers

因为安全文件需要把文件的传输都记录下来。“log transfers”的格式是：

```
log transfers <typelist> <directions>
```

<typelist>是用逗号隔开的字串，表示哪些用户的命令需要记录下来，字串的取值可以是：anonymous、guest 或 real。<direction>也是用逗号隔开的字串，设置需要记录的文件传输的方向，可以选择的两个传输方向是“inbound”（向内）和“outbound”（向外）。

例如，用下面表示记录所有 real 和 guest 用户的“inbound”和“outbound”方向的文件传输：

```
log transfers real,guest inbound,outbound
```

这些记录保存在“/var/log/xferlog”文件中。

guestgroup

这个命令用来设置 guest 组，每一行只能有一个成员。

例如：

```
guestgroup ftpadmin
guestgroup webmaster
```

log security

用来设置记录 real、guest 或 anonymous 用户违反安全规则的行为。

```
log security <typelist>
```

<typelist>是用逗号隔开的字串，字串的取值可以是：anonymous、guest 或 real。“real”表示真正在 ftp 服务器上有帐号的用户，“anonymous”表示匿名用户，“guest”表示 guest 用户。

例如：

```
log security real,guest
```

restricted-uid、restricted-gid、guest-root

这些用来设置是否允许 real 和 guest 用户访问家目录之外的目录。格式为：

```
guest-root <root-dir>
restricted-uid <uid-range>
restricted-gid <gid-range>
```

例如：

```
guest-root /home/ftp ftpadmin webmaster
restricted-uid ftpadmin webmaster
restricted-gid adminftp webmaster
```

<root-dir>设置 chroot 环境的用户路径。在一行里可以有多个 uid 地范围。如果为用户设置了 guest-root，那么该用户的家目录就在“<root-dir>/etc/passwd”文件中

设定，而“/etc/passwd”文件中的设定是无效的。当“ftpadmin”和“webmaster”被改变了根目录（chrooted）到“/home/ftp”目录下，他们就被限制在各自的家目录下而不能访问别人的文件。

greeting

设置用户登录时候的欢迎信息。格式为：

```
greeting full|brief|terse
```

“greeting full”是默认的设置显示主机名和 ftp daemon 地版本，“greeting brief”显示主机名，“greeting terse”简单地显示“FTP server ready”。

例如：

```
greeting terse
```

keepalive <yes|no>

设置 socket 的 TCP SO_KEEPALIVE 参数。这样在必要的时候可以断开网络连接。“yes”有效，“no”无效。最好设成“yes”：

```
Keepalive yes
```

配置 /etc/ftphosts 文件

“/etc/ftphosts”文件为每一个用户建立规则，指定允许用户从某个主机登录 ftp 服务器，或者不允许用户从某个主机登录 ftp 服务器。

创建“ftphosts”文件（touch /etc/ftphosts），加入下面这几行：

```
# Example host access file
#
# Everything after a '#' is treated as comment,
# empty lines are ignored
allow ftpadmin 208.164.186.1 208.164.186.2 208.164.186.4
```

把文件的权限改为 600：

```
[root@deep]# chmod 600 /etc/ftphosts
```

每一行可能是：

```
allow <username> <addrglob>
```

或

```
deny <username> <addrglob>
```

“allow”允许用户用<username>用户名，从<addrglob>地址访问 ftp 服务器。<addrglob>可以包含多个地址。

“deny”禁止用户名为<username>的用户，从<addrglob>地址访问 ftp 服务器。<addrglob>可以包含多个地址。

配置 /etc/ftpusers 文件

“/etc/ftpusers”文件设置哪些用户**不允许**连接到 ftp 服务器。

创建“ftpusers”文件（touch /etc/ftpusers），加入下面这几行：

```
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

把文件的权限设成 600：

```
[root@deep]# chmod 600 /etc/ftpusers
```

配置 /etc/ftpconversions 文件

“/etc/ftpconversions”是用来控制当传输文件的时候是否进行压缩。

创建“ftpconversions”文件（touch /etc/ftpconversions），在文件中加入：

```
:.Z: : :/bin/compress -d -c %s:T_REG|T_ASCII:O_UNCOMPRESS:UNCOMPRESS
: : :.Z:/bin/compress -c %s:T_REG:O_COMPRESS:COMPRESS
:.gz: : :/bin/gzip -cd %s:T_REG|T_ASCII:O_UNCOMPRESS:GUNZIP
: : :.gz:/bin/gzip -9 -c %s:T_REG:O_COMPRESS:GZIP
: : :.tar:/bin/tar -c -f - %s:T_REG|T_DIR:O_TAR:TAR
: : :.tar.Z:/bin/tar -c -Z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+COMPRESS
: : :.tar.gz:/bin/tar -c -z -f - %s:T_REG|T_DIR:O_COMPRESS|O_TAR:TAR+GZIP
: : :.crc:/bin/cksum %s:T_REG::CKSUM
: : :.md5:/bin/md5sum %s:T_REG::MD5SUM
```

把文件的属性改为 600：

```
[root@deep]# chmod 600 /etc/ftpconversions
```

配置 /etc/pam.d/ftp 文件

配置“/etc/pam.d/ftp”文件使其支持 PAM 安全验证。

创建“ftp”文件（touch /etc/pam.d/ftp）并加入：

```
#%PAM-1.0
auth required /lib/security/pam_listfile.so item=user sense=deny
file=/etc/ftpusers onerr=succeed
auth required /lib/security/pam_pwdb.so shadow nullok
auth required /lib/security/pam_shells.so
account required /lib/security/pam_pwdb.so
session required /lib/security/pam_pwdb.so
```

配置 /etc/logrotate.d/ftpd 文件

配置“/etc/logrotate.d/ftpd”文件使得日志文件每周自动循环更新。

创建“ftpd”文件（touch /etc/logrorate.d/ftpd）并加入：

```
/var/log/xferlog {
# ftpd doesn't handle SIGHUP properly
nocompress
}
```

配置 ftp 使其使用 inetd 超级服务器　用于实现 tcp-wrappers

tcp-wrappers 用来启动和中止 ftpd 服务。当 inetd 执行的时候，它会从默认为“/etc/inetd.conf”的配置文件读入配置信息。配置文件中每一行中的项用 TAB 或空格隔开。

编辑 inetd.conf 文件（vi /etc/inetd.conf），加入并验证是否存在下面这一行：

```
ftp stream tcp nowait root /usr/sbin/tcpd in.ftpd -l -a
```

注意：更新完“inetd.conf”文件之后要发给 inetd 一个 SIGNUP 信号，运行下面的命令：

```
[root@deep /root]# killall -HUP inetd
```

编辑“hosts.allow”文件（vi /etc/hosts.allow）加入这一行：

```
in.ftpd: 192.168.1.4 win.openarch.com
```

这表示 IP 地址为“192.168.1.4”并且主机名为“win.openarch.com”的计算机允许访问 ftp 服务器。

FTP 管理工具

ftpwho

ftpwho 显示当前连接到 ftp 服务器上的所有用户。这个命令菜单输出类似“/bin/ps”的输出，其格式为：

```
<pid> <time> <tty> <connection details>
```

其中<pid>表示 ftp daemon 用来处理这次文件传输的进程号，<time>表示用户什么时候连接到 ftp 服务器上，<tty>总是用问号（?）表示因为是通过 ftp 而不是 telnet 连接，<connection details>告诉连接是来自哪里、用户是谁以及用户现在在干什么。

下面是 ftpwho 输出的一个例子：

```
[root@deep]# ftpwho
```

```
Service class openarch:
5443 ? S 0:00 ftpd: win.openarch.com: admin: IDLE
- 1 users ( 20 maximum)
```

可以看到现在有一个用户登录（最多可以有 20 个用户同时登录），这个用户的用户名是 admin 来自 win.openarch.com。

ftpcount

ftpcount 是 ftpwho 的简化版，只显示登录到 ftp 服务器的用户数以及最多允许多少个用户登录。下面是一个例子：

```
[root@deep]# ftpcount


Service class openarch - 1 users ( 20 maximum)
```

保证 ftp 服务器的安全

首先确保已经创建了“/etc/ftpusers”文件，这个文件用来设置**不允许**哪些用户登录 ftp 服务器，其中至少要包括：root、bin、daemon、adm、lp、sync、shutdown、halt、mail、news、uucp、operator、games、nobody 以及所有 Linux 发行商在系统中提供的默认帐号。

如果想禁止匿名 ftp 服务，把 ftp 用户从 password 文件中移去，再用下面的命令确定在系统中没有安装 anonftp-version.i386.rpm 软件包：

```
[root@deep]# rpm -q anonftp.
```

upload 命令

在默认情况下，WU-FTPD 服务器给所有的 guest 用户上载的权限。当用户登录的时候，被改变根目录（chroot）到“/home/ftp”就不能访问这个目录之外的地方。但是“/home/ftp”目录中的一些地方还是需要保护，不能让用户随便访问。在我们配置的 ftp 服务器中为“/home/ftp”目录下的“bin”、“etc”、“dev”和“lib”目录。我们不允许用户上载文件到这些目录。所以我们要为这些目录设置访问权限，可以在“/etc/ftpaccess”文件中设置上载权限。在我们的例子中是这样设置的：

```
upload /home/ftp/* / no
upload /home/ftp/* /etc no
upload /home/ftp/* /dev no
```

```
upload /home/ftp/* /bin no (require only if you are not using the “--enable-ls” option)
upload /home/ftp/* /lib no (require only if you are not using the “--enable-ls” option)
```

noretrieve 命令

最好禁止某些用户从“/home/ftp”目录下的某些子目录中下载文件，可以用“noretrieve”命令在“/etc/ftpaccess”文件中设置。

```
noretrieve /home/ftp/etc
noretrieve /home/ftp/dev
noretrieve /home/ftp/bin (require only if you are not using the “--enable-ls” option)
noretrieve /home/ftp/lib (require only if you are not using the “--enable-ls” option)
```

.notar 文件

无论是否允许即时的目录打包（on-the-fly tar），都必须保证用户不能打包（tar）禁止上载的目录。在“/home/ftp”目录的每个子目录中都创建“.notar”文件。

```
[root@deep]# touch /home/ftp/.notar
[root@deep]# chmod 0 /home/ftp/.notar
[root@deep]# touch /home/ftp/etc/.notar
[root@deep]# chmod 0 /home/ftp/etc/.notar
[root@deep]# touch /home/ftp/dev/.notar
[root@deep]# chmod 0 /home/ftp/dev/.notar
[root@deep]# touch /home/ftp/bin/.notar (require only if you are not using the
“--enable-ls” option)
[root@deep]# chmod 0 /home/ftp/bin/.notar (require only if you are not using the
“--enable-ls” option)
[root@deep]# touch /home/ftp/lib/.notar (require only if you are not using the
“--enable-ls” option)
[root@deep]# chmod 0 /home/ftp/lib/.notar (require only if you are not using the
“--enable-ls” option)
```

这些长度为 0 的“.notar”文件会使一些浏览器和 ftp 代理（proxy）出现混乱，要解决这个问题必须把它们标识为禁止下载。在“/etc/ftpaccess”文件中加入这一行：

```
noretrieve .notar
```

安装到系统中的文件

> /etc/ftphosts
> /etc/ftpusers
> /etc/ftpaccess
> /etc/pam.d/ftp
> /etc/ftpconversions
> /etc/ftpgroups
> /etc/logrotate.d/ftpd
> /usr/bin/ftpcount
> /usr/bin/ftpwho
> /usr/man/man1/ftpcount.1
> /usr/man/man1/ftpwho.1
> /usr/man/man5/ftpaccess.5
> /usr/man/man5/ftphosts.5
> /usr/man/man5/ftpconversions.5
> /usr/man/man5/xferlog.5
> /usr/man/man8/ftpd.8
> /usr/man/man8/ftpshut.8
> /usr/man/man8/ftprestart.8
> /usr/sbin/in.ftpd
> /usr/sbin/ftpshut
> /usr/sbin/ckconfig
> /usr/sbin/ftprestart
> /usr/sbin/xferstats
> /usr/sbin/wu.ftpd
> /usr/sbin/in.wuftpd
> /var/log/xferlog

第六部分：与备份相关的参考资料

第十一章

备份和恢复的过程

备份和恢复的过程

服务器备份过程

服务器的安全性和可靠性是与备份密切相关的。定期备份是一件非常重要的事。错误随时会发生。有可能是硬件出错、电源问题或是人为的错误。如果你是系统管理员，很有可能有人会让你恢复一些被误删除的文件。

如果定期备份，就能把这种损失减到最小。最安全的备份方法是把它们备份到别的地方，如：网络上、磁带、可移动驱动器（removable drive）或可写光驱，等等。

在 linux 上有很多备份的方法。包括所有 Linux 发行版本都有的命令行工具，象“dd”、“dump”、“cpio”和“tar”。也有基于文本界面的工具，例如：“Amanda”，它提供更友好的用于备份和恢复的界面。而且，还有商业备份软件，如：“BRU”。

显然，备份和恢复的过程会因为不同的备份方案而有所区别。所以，我们介绍最常用的备份工具“tar”，它是*nix 平台上十分常见的命令行备份工具。

下面的命令把整个 Linux 服务器上的东西备份到“/archive”文件系统上。不过，这些不包括“/proc”这个虚拟的文件系统、任何 mount 到“/mnt”目录下的文件系统和“/archive”文件系统（没道理自己备份自己）。

注意：在备份文件系统时，不要包含“/proc” 虚拟文件系统！在“/proc”目录下的文件不是真正的文件，只不过是描述并指向内核数据结构的简单的文件链接。而且，也不要包含“/mnt”和“/archive”目录下的文件。

备份整个系统的命令如下：

```
[root@deep]# cd /
[root@deep]# tar -zcvpf /archive/full-backup-`date '+%d-%B-%Y'`.tar.gz \
--directory / --exclude=proc --exclude=mnt --exclude=archive \
--exclude=cache .
```

- z 表示备份的数据将使用“gzip”进行压缩。
- c 表示创建归档文件。
- v 显示文件列表。
- p 保存权限，文件的访问权限将被“记住”。
- f 说明下一个参数就是归档的文件名或设备名。

请注意一下带有当前日期的文件名是如何产生的，其方法是在两个后引号之间放入“date”命令。通常的命名习惯是给未压缩的文档加一个“tar”后缀，经过压缩后的加上“tar.gz”。

“--directory”选项告诉 tar 先转到规定的目录下（本例中为‘/’目录），然后进行备份。“--exclude”选项告诉 tar 不要备份指定的目录或文件。最后，命令末尾的“.”号告诉 tar 要备份当前目录下的所有内容。

再举一个例子，这一次只把指定的文件系统备份到 SCSI 磁带驱动器：

只备份指定的文件到一个 SCSI 磁带驱动器，使用如下命令：

```
[root@deep]# cd /
[root@deep]# tar -cvpf /dev/nst0 --label="Backup set created on `date
'+%d-%B-%Y'`." \
--directory / etc home chroot
```

注意：在上面的命令中没有用到“z”（压缩）选项，但是，我们建议把数据压缩后，再送到磁带上。因为磁带驱动器是一个设备，所以不能用文件名表示。因此，只能把设备名“/dev/nst0”作为 tar 的参数。“/dev/nst0”表示 SCSI 总线上的第一个磁带设备。

注意：“/dev/nst0”设备在备份集写完后不回卷；因此可以在一个磁带上写多个集合。你也可以指定设备为“/dev/st0”，这时磁带在备份集写完后自动回卷。

既然我们无法为备份集指定文件名，“--label”选项可以用来在归档文件里写入一些备份集的信息。最后，只有在“/etc”，“/home”和“/chroot”目录下的文件才被备份到磁带。

使用磁带时，可以用下面的命令回卷和弹出磁带：

```
[root@deep]# mt -f /dev/nst0 rewind
[root@deep]# mt -f /dev/nst0 offline
```

服务器恢复过程

当我们需要恢复一个重要文件时，正确地恢复就比定期备份更重要了！正如以上所讨论的，恢复的过程会因为备份方案的不同而有区别。在本节中，我们讨论如何恢复用 tar 备份过的文件。

以下的命令将从“full-backup-14-November-1999.tar.gz”档案中恢复所有的文件，该档案是在上面的服务器备份例子中生成的。

恢复整个系统的备份，命令如下：

```
[root@deep]# cd /
[root@deep]# tar –zxvpf /archive/full-backup-14-November-1999.tar.gz
```

此命令解开了压缩档案中的所有文件，同时保持了原来的文件属主和权限。

- **z** 表明档案是使用 gzip 压缩的。
- **x** 选项表示解包。
- **v** 显示得到的文件列表。
- **p** 保持权限；文件保护信息将被记住。
- **f** 后面的参数是档案的文件名或设备。

如果不需要恢复档案里的所有文件，可以参照以下的例子恢复指定的一个或多个文件：

恢复指定文件，命令如下：

```
[root@deep]# cd /
[root@deep]# tar -zxvpf /archive/full-backup-09-October-1999.tar.gz\
etc/passwd usr/sbin/chpasswd
```

该命令从档案中恢复文件“etc/passwd”和“usr/chpasswd”。

如果想从档案中恢复一个或少量的几个文件，就必须先找到文件名和它在档案中的具体路径。以下的例子将解决这个问题：

```
[root@deep]# cd /
[root@deep]# tar -ztvpf /archive/full-backup-09-October-1999.tar.gz \
| grep -i chpasswd
```

在这个例子里，档案中的所有文件名被列出。输出结果被重定向到 grep 命令，grep 的“i” 选项忽略了大小写，显示出路径或文件名中含“chpasswd”的所有文件。一旦找到了要恢复的文件，就可以指定文件名并使用上面的 tar 命令。

注意：当创建档案文件时，tar 会去掉文件路径开头的“/”斜线字符。这意味着文件恢复的位置可能和它备份时的位置不一样。因此，解决问题的办法就是在根目录下做所有的备份和恢复。另一种办法是先在不同的目录下恢复需要的文件，然后通过比较、移动或更新把文件恢复到原来的位置。

注意：如果你的系统里有文件被“chattr”命令设了不可变位，这些文件在恢复的时候将不会保持该位。你必须在备份结束后再使用命令“chattr”重新设置不可变位。

更多的资料

需要更多信息，你可以读 man 页：

tar(1) —GNU 版的 tar 归档工具

tar 备份的替代品

AMANDA

AMANDA（Advanced Maryland Automatic Network Disk Archiver 高级马里兰自动网络磁盘归档工具）是一个备份系统，它允许局域网的管理员用一个主备份服务器将多台主机的内容备份到一个大容量磁带驱动器上。AMANDA 使用本地 dump 和/或 GNU 的 tar 工具并能备份大量的多种版本的 UNIX 工作站。最新的版本还能使用 SAMBA 来备份微软的 Windows 95/NT 主机。

软件包

AMANDA 主页：http://www.cs.umd.edu/projects/amanda/

BRU

BRU 是为实现最大的可靠性和灵活性而设计的 Unix 备份和恢复工具。它让 Unix 的备份更方便、快捷和安全。BRU 远不止是 tar 和 cpio 的替代品，而是真正的多功能的备份系统。

软件包

BRU 的主页：http://www.bru.com/

第七部分：附录

附录 A

优化设置、小窍门和管理工作

这里的介绍的小窍门有些只能用于 Linux 系统，但是大多数对 Unix 系统都是适用的。

1.0 “du”命令

“du”命令是用来确定磁盘空间的。例如：为了确定“/var/log”和“/home”目录的大小，用下面的命令：

```
[root@deep]# du -sh /var/log /home
1.0M /var/log
66M /home
```

请注意上面的命令报告的是目录的实际大小。“/home”目录的大小为 66M，你可以进入“/home”目录（cd /home），然后用“du –sh *”命令可以确定这个目录中最大的文件。

注意：可以在“crontab”中加入上面的命令。这样不要登录系统，系统每天就能定期地报告系统磁盘空间的情况。

1.1 查询从你的计算机发出的 IP 包到远程计算机的路由

如果想知道从自己计算机发出的 IP 包到远程主机的路由情况，可以用这个命令：“traceroute <hostname>”，这里<hostname>是远程主机的主机名或 IP 地址。例如：

```
[root@deep]# traceroute www.redhat.com
```

1.2 显示主页被访问的次数

用下面的命令显示主页被访问的次数：

```
[root@deep]# grep "GET / HTTP" /var/log/httpd/access_log | wc -l
```

1.3 终止所有的服务

可以用下面的命令关闭所有的服务：

```
[root@deep]# killall httpd smbd nmbd sendmail named
```

1.4 在所有的终端窗口上显示时间

编辑“profile”文件（vi /etc/profile），加入下面这一行：

```
PROMPT_COMMAND='echo -ne
"\0337\033[2;999r\033[1;1H\033[00;44m\033[K"`date`"\033[00m\0338"'
```

1.5 你装了“lsof”吗？

如果没有安装，就安装一个。“lsof –i”这个命令可以列出计算机上已经打开的端口。尽管简单地用“netstat –an | grep LISTEN”也可以列出打开的 TCP 端口，但是“lsof”还可以显示出到底是哪一个进程在监听指定的端口，而“netstat”就不能。

```
[root@deep]# lsof -i
COMMAND PID USER FD TYPE DEVICE SIZE NODE NAME
Inetd 344 root 4u IPv4 327 TCP *:ssh (LISTEN)
sendmail 389 root 4u IPv4 387 TCP *:smtp (LISTEN)
smbd 450 root 5u IPv4 452 TCP deep.openarch.com:netbios-ssn (LISTEN)
nmbd 461 root 5u IPv4 463 UDP *:netbios-ns
nmbd 461 root 6u IPv4 465 UDP *:netbios-dgm
nmbd 461 root 8u IPv4 468 UDP deep.openarch.com:netbios-ns
nmbd 461 root 9u IPv4 470 UDP deep.openarch.com:netbios-dgm
named 2599 root 4u IPv4 3095 UDP *:32771
named 2599 root 20u IPv4 3091 UDP localhost.localdomain:domain
named 2599 root 21u IPv4 3092 TCP localhost.localdomain:domain (LISTEN)
named 2599 root 22u IPv4 3093 UDP deep.openarch.com:domain
named 2599 root 23u IPv4 3094 TCP deep.openarch.com:domain (LISTEN)
```

1.6 不用登录就可以通过 ssh 协议在远程主机上运行命令

“ssh”命令可以不用登录就在远程主机上运行命令。命令运行的结果显示在本地的计算机。下面的命令可以列出登录到远程主机的用户：

```
[admin@deep]$ ssh mail.openarch.com who
admin@mail.openarch.com's password:
root tty1 Dec 2 14:45
admin tty2 Dec 2 14:45
wahib pts/0 Dec 2 11:38
```

1.7 补全文件名

可以按[TAB]键来补全文件名或程序名，这样就可以少敲很多键。如果有多个文件的起始部分和已经输入的一样，计算机就会“嘟”一声，再按[TAB]键就会列出所有起始部分相同的文件。有机会的话，自己可以试试看，可以省调你很多时间。

附录 B

获取 RFC（Requests for Comments）

Requests for Comments (RFCs)是在 Network Information Center(NIC)的 Internet Engineering Task Force(IETF)发布的新协议和 Internet 协议标准的一系列文档。每一个文档都定义了 Internet 协议的某个方面。我们在下面列出了所有和这本书有关的 RFC。可以从 http://www.cis.ohio-state.edu/rfc/ 获得 RFC 文档。

RFC706

On the Junk Mail Problem.

RFC733

Standard for the Format of ARPA Network Text Messages.

RFC768

User Datagram Protocol (UDP).

RFC791

Internet Protocol (IP).

RFC792

Internet Control Message Protocol (ICMP).

RFC793

Transmission Control Protocol (TCP).

RFC805

Computer Mail Meting Notes.

RFC821

Simple Mail Transfert Protocol (SMTP).

RFC822

Standard for the Format of ARPA Internet Text Massages.

RFC934

Proposed Standard for Message Encapsulation.

RFC950

IP Subnet Extention.

RFC959

File Transfer Protocol (FTP).

RFC976

UUCP Mail Interchange Format Standard.

RFC1034

Domain Names: Concepts and Facilities.

RFC1036

Standard for Interchange of USENET Message.

RFC1058

Routing Information Protocol (RIP).

RFC1112

Internet Group Multicast Protocol (IGMP).

RFC1122

Requirement for Internet Host—Communication Layers.

RFC1123

Requirements for Internet Host—Application and Support.

RFC1137

Mapping Between Full RFC 822 and RFC 822 with Restricted Encoding.

RFC1153

Digest Message Format.

RFC1155

Structure of Management Information (SMI).

RFC1157

Simple Network Management Protocol (SNMP).

RFC1176

Interactive Mail Access Protocol: Version 2.

RFC1310

The Internet Standards Process.

RFC1319

MD2 Message-Digest Algorithm.

RFC1320

MD4 Message-Digest Algorithm.

RFC1321

MD5 Message-Digest Algorithm.

RFC1343

User Agent Configuration Mechanism for Multimedia Mail Format Information.

RFC1344

Implications of MIME for Internet Mail Gateways.

RFC1345

Character Mnemonics and Character Sets.

RFC1421

Privacy Enhancement for Internet Electronic Mail: Part I—Message Encipherment and authentication Procedures.

RFC1422

Privacy Enhancement for Internet Electronic Mail: Part II—Certificate-based key Management.

RFC1423

Privacy Enhancement for Internet Electronic Mail: Part III—Algorithms, modes, and identifiers [Draft].

RFC1428

Transmition of Internet Mail from Just-Send-8 to 8bit-SMTP/MIME.

RFC1492

An Access Control Protocol, Sometimes Called TACACS.

RFC1495

Mapping Between X.400(1988)/ISO 10021 and RFC 822.

RFC1496

X.400 1988 to 1984 Downgrading.

RFC1505

Encoding Header Field for Internet Messages.

RFC1510

The Kerberos Network Authentication Service (V5).

RFC1519

Classless Inter-Domain Routing (CIDR) Assignment and Aggregation Strategy.

RFC1521

MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the

Format of Internet Message Bodies (MIME).

RFC1522

Representation of Non-ASCII Text in Internet Message Headers.

RFC1566

Mail Monitoring MIB.

RFC1579

Firewall-Friendly FTP.

RFC1583

Open Shortest Path First Routing V2 (OSPF2).

RFC1625

WAIS over Z39.50-1988.

RFC1631

The IP Network Address Translator (NAT).

RFC1652

SMTP Service Extentions for 8bit-MIMEtransport.

RFC1661

Point-to-Point Protocol (PPP).

RFC1711

Classifications in E-mail Routing.

RFC1725

Post Office Protocol, Version 3 (POP)3.

RFC1738

Uniform Resource Locators (URL).

RFC1739

A Primer on Internet and TCP/IP Tools.

RFC1796

Not All RFCs are Standards.

RFC1830

SMTP Services Extentions for Transmission of Large and Binary MIME Messages.

RFC1844

Multimedia E-mail (MIME) User Agent checklist.

RFC1845

SMTP Service Extension for Checkpoint/Restart.

RFC1846

SMTP 521 Reply Code.

RFC1854

SMTP Service Extention for command pipelining.

RFC1855

Netiquette Guidelines.

RFC1864

The content-MD5 Header.

RFC1866

Hypertext Markup Language - 2.0.

RFC1869

附录 B

SMTP Service Extensions.

RFC1870

SMTP Service Extension for Message Size Declaration.

RFC1872

The MIME Multipart/Related Content-type.

RFC1873

Message/External-Body Content-ID Access-type.

RFC1883

Internet Protocol, Version 6 (Ipv6) Specification.

RFC1884

IP Version 6 Addressing Atchitecture.

RFC1886

DNS Extentions to support IP version 6.

RFC1891

SMTP Service Extension for Delivery Status Notifications.

RFC1892

The Multipart/Report Content Type for the Reporting of Mail System Administrative Messages.

RFC1893

Enhanced Mail System Status Codes.

RFC1894

An Extensible Message Format for Delivery Status Notifications.

RFC1918

Address Allocation for Private Internets.

RFC1928

SOCKS Protocol Version 5.

RFC1929

Username/Password Authentication for SOCKS V5.

RFC1961

GSS-API Authentication Method for SOCKS Version 5.

RFC2003

IP Encapsulation within IP.

RFC2028

The Organizations Involved in the IETF Standards Process.

附录 B

RFC2060

Internet Message Access Protocol – Version 4rev1 (IMAP4).

RFC2104

HMAC: Keyed-Hashing for Message Authentication.

RFC2138

Remote Authentication Dial In User Service (RADIUS).

RFC2200

Internet Official Protocol Standards.

RFC2305

A Simple Mode of Facsimile Using Internet Mail.

RFC2313

PKCS 1: RSA Encryption Version 1-5.

RFC2314

PKCS 10: Certification Request Syntax Version 1-5.

RFC2315

PKCS 7: Cryptographic Message Syntax Version 1-5.

附录C

参加翻译的工程师及翻译的相应章节

工程师	翻译的相应章节
brimmer	介绍 第一部分：与安装相关的参考资料 第三章：系统安全概要 第八章：编译器的功用 第九章：系统安全软件 第十章：服务器软件 Linux OPENSSL 服务器 Linux OpenLDAP 服务器 Linux IPX Netware 客户 Linux FTP 服务器 附录
davidliu	第十章：服务器软件 Linux Imap & Pop 服务器 Linux MM - 共享内存库 Linux Samba 服务器 Linux DNS and BIND 服务器 Linux Sendmail 服务器
ideal	第五章：配置和编译内核 第六章：TCP/IP 网络管理 第十章：服务器软件

附录 C

	Linux Squid Proxy 服务器
jwlinux	第十章：服务器软件 Linux Apache Web 服务器
laotang	第七章：网络防火墙
pwc	第十章：服务器软件 Linux PostgreSQL Database 服务器
sourcebox	第四章：系统优化概要
bulbul	第十一章：备份和恢复的过程