FreeBSD 使用手册

摘要

□迎使用 FreeBSD！ 本手册□用于安装 *FreeBSD 11.2-RELEASE* 和 *FreeBSD 12.0-RELEASE* 以及它□的日常使用。 □个手册目前由很多人 持□地 □□。 其中的内容需要不断地更新。 如果□有□趣参加□个□目，□□□件到 FreeBSD 文□□□□件列表。 此文□最新的英文原始版本可以从 FreeBSD 网站 上□得 (本手册的□早期版本可以在 http://docs.FreeBSD.org/doc/ □到)。 由 FreeBSD 中文□□ □□的最新□本可以在 FreeBSD 中文□□ 快照网站 □得， □一□本会持□地向主站同□。 此外， □也可以从 FreeBSD FTP 服□器 及其□多 □像站点 取得□□文□的各□其它格式， 以及□□形式的版本。 如果□希望得到一□印刷版本的手册， 可以从 FreeBSD Mall □□。 除此之外， □□可以 在手册中搜索内容。

目□

前言

□期的□者

作□ FreeBSD 的新用□，□将会在本手册第一部分□到 FreeBSD 的安装方法，同□逐□引入概念和□俗来加□ UNIX® 基□。 □□□部分只需要□有探索的精神和接受新概念的能力。

□完□些之后，手册中很漫□的第二部分是 FreeBSD 中系□管理□感□趣的所有主□的全面参考。 在□□□些章□的内容□所需要的背景知□都注□在□一章□的大□里面， 如果需要，可在□□前□行□□。

要□得附加的信息来源列表，□□□ 参考文献。

相□于第三版的改□

目前的在□手册代表了数百位□献者□去 10 年多所累□的努力成果。 以下是自 2004 年出版的□卷第三版之后的一些重要□更：

- DTrace， DTrace， □加了有□□大的 DTrace 性能分析工具有□的信息。
- 文件系□ Support， 文件系□支持， □加 FreeBSD 上非原生文件系□有□的信息， 比如 Sun™ 的 ZFS。
- 安全事件□□， 安全事件□□， □加了 FreeBSD 新的□□功能和使用方法。
- 虚□化， 虚□化， □加了在虚□化□件上安装 FreeBSD 有□的信息。

相□于第二版的改□ (2004)

□目前看到的□本手册的第三版是 FreeBSD 文□□□的成□□□□年完成的□峰之作。 □一版的内容已□□□到需要分成□卷才能印刷出版。 第三版包含了如下的的主要□□：

- □置和□整， 配置和□化， □行了□充并□加了□于 ACPI □源和□源管理， `cron` 系□□用程序，以及更多的内核□化□□的相□内容。
- 安全， 安全 一章□加了虚□□用网 (VPNs)， 文件□□控制表 (ACLs)， 以及安全公告的内容。
- □制□□控制， □制□□控制 (MAC) 是□一版新□的章□。 它解□了什□是 MAC， 以及□一机制如何使□的 FreeBSD 系□更安全。
- 存□， 存□， 在原有基□上□加了 USB 存□□□， 文件系□快照， 文件系□容限， 基于文件及网□的文件系□，以及与加密磁□分区有□的内容。
- Vinum 卷管理程序，Vinum， 是□一版中的新章□。 描述了如何使用□□提供了□□无□的□□磁□、□件 RAID-0, RAID-1 和 RAID-5 的卷管理系□——Vinum。
- 在 PPP 和 SLIP，PPP 和 SLIP 一章中□加了排除故障的□明。
- □子□件， □子□件 一章中□加了□于如何使用其它的□件□□代理、 SMTP □□、 UUCP、 fetchmail、procmail、 以及其它□□内容。
- 网□服□器，网□服□， 是新版中全新的一章。 □一章包括了如何架□ Apache HTTP 服□器、 ftpd，以及用于支持 Microsoft® Windows® 客□的 Samba。 一些段落来自原先的 高□网□， □□网□□用 一章。
- 高□网□，□□网□□用 一章□加了□于在 FreeBSD 中使用 Bluetooth® □□， 安装无□网□， 以及使用□□□□模式 (ATM) 网□的内容。

- □加了一□□□表， 用以□明整本□中出□的□□。
- □于全□中□表□行了□一□的美化工作。

相□于第一版的改□ (2001)

本手册的第二版是 FreeBSD 文□□□的成□□□□年完成的□峰之作。 第二版包含了如下的的主要□□：

- 添加了完整的索引。
- 用□形替□了以前所有用 ASCII □□。
- □个章□添加了□准大□，列出了□章所包含的信息和□者所□□了解的知□。
- 内容□□地分成三个部分： "起□"， "系□管理"和"附□"。
- 安装 FreeBSD ("安装 FreeBSD") 新版本中使用了□屏□片，使新用□更容易的□会正文。
- UNIX 基□ ("UNIX® 基□")□充了□程、守□□程和信号的附加信息。
- 安装□用程序. Packages 和 Ports ("安装□用程序") □充了二□制包管理的附加信息。
- X Window 系□ ("X Window 系□") 新版本中着重介□使用□代□面技□例如 XFree86™ 4.x 上的 KDE 和 GNOME
- FreeBSD 引□□程 ("FreeBSD □□□程") □第一版内容□行□充。
- 存□ ("存□") 由第一版中□个□独的章□"磁□"和"□□"合并而成。 我□□□□□部分作□一个整体比□容易理解。同□ RAID (包括硬件和□件 RAID) 部分也被添加□来。
- 串口通□ ("串口通信") □第一版□行完善，并□ FreeBSD 4.x/5.x 做了更新。
- PPP 和 SLIP ("PPP 和 SLIP") 全部更新。
- □多新的内容被添加到 高□网□ ("高□网□").
- □子□件 ("□子□件")□加了□于配置 sendmail 的信息。
- Linux® 二□制兼容模式("Linux® 兼容性") □加了□于安装 Oracle® 和 SAP® R/3® 的信息。
- 第二版中也涵□了下列主□：
  - 配置和□整(□置和□整)。
  - 多媒体(多媒体)

本手册的□□

□本手册分成了五个□□清晰的部分。第一部分 起□ 涵□了 FreeBSD 的安装和基本使用方法。□者可根据自己的情况按□序或者跳□一些熟悉的主□来□□。第二部分 常用操作 涵□了 FreeBSD 常用的功能，□部分可以不按□序□□。□个部分由一个□明的大□□始， □个大□描述本章□涵□的内容和□者□□已□知道的知□。□主要是□□者可以更好的□□感□趣的章□□□。第三部分 系□管理 涵□了 FreeBSD 高□用□所感□趣的广泛的□□。第四部分 网□通□ 包括了网□和服□的□□， 而第五部分□是□源信息的附□。

介□， 介□

向新用□介□ FreeBSD。它描述了 FreeBSD □□的□史、目□和□□模式。

安装 *FreeBSD*， 安装

本章将会□□用□完成安装□程。一些高□安装主□， 例如如何通□串行控制台安装，也涵□在内。

UNIX 基□， *UNIX*® 基□

本章涵□了 FreeBSD 操作系□基□命令和功能。 如果熟悉 Linux® 或者其他□ UNIX® 操作系□，□可以跳□□章。

安装□用程序. *Packages* 和 *Ports*， 安装□用程序

本章涵□如何用 FreeBSD 的 "Ports Collection" 和□准二□制□件包来安装第三方□件。

X Window 系□， *X Window* 系□

本章概要地描述了 X Window System 系□并□□地介□了如何在 FreeBSD 上使用它。此外他也描述了常用的□面□境， 例如 KDE 和 GNOME。

□面□用， □面□用

列出了一些常用的□面□用程序，比如 web □□器和□公套件， 描述了在 FreeBSD 上如何安装它□。

多媒体， 多媒体

展示了如何□□的系□□置声□和□□回放支持。 也描述了一些□□的音□和□□□用程序。

配置*FreeBSD*的内核， 配置 *FreeBSD* 内核

解□了□什□需要配置一个新内核并提供了配置、 □□、安装自定□内核的□□□明。

打印， 打印

描□了 FreeBSD 上打印机管理， 包括横幅□、打印□□，□有初始的□置。

Linux® 二□制兼容模式， *Linux*® 二□制兼容

描述了 FreeBSD 的 Linux® 兼容特性。 也提供了□多流行的 Linux® □用程序的□□的安装□明， 比如 Oracle® 和 Mathematica®。

□置和□整， 配置和□整

本章描述了管理□□整 FreeBSD 系□以□化性能□可能用到的一些参数。 也描述了 FreeBSD 中的各□配置文件以及它□所在的位置。

FreeBSD 引□□程， □□□程

本章描述 FreeBSD 的□□□程并且解□了如何用配置□□来控制□个□程。

用□和基本的□□管理， 用□和基本□号管理

本章描述了如何□建和操作用□□号， 同□也□述了□置用□□源限制和其他□号管理任□的方法。

安全， 安全

描述了保□ FreeBSD 系□安全可以使用的□多工具， □包括 Kerberos， IPsec 以及 OpenSSH。

Jails， *Jail*

介□了 jail 框架， 以及 jail 相□于 FreeBSD 中□□的 chroot 支持的改□。

□制□□控制， □制□□控制

解□了何□□制□□控制 (MAC) 以及如何利用□一机制来加□ FreeBSD 系□的安全。

安全事件□□， 安全事件□□

介□了 FreeBSD 事件□□是什□， 以及如何安装、 配置它， 并□□或□□□□□□信息。

存□， 存□

本章描述了□□用 FreeBSD 来管理存□介□和文件系□，包括物理磁□、RAID □列、光学和磁□媒体、后□存□磁□以及网□文件系□。

GEOM. 模□化磁□□□框架， ***GEOM***

介□了 FreeBSD 中的 GEOM 框架是什□， 以及如何配置它所支持的各□ RAID。

文件系□ ***Support***, 文件系□支持

探□了 FreeBSD □非原生文件系□的支持，比如 Sun™ 的 Z 文件系□。

Vinum 卷管理程序, ***Vinum***

本章描述了□□使用□□卷管理器 Vinum。它提供了□□无□的□□磁□和□件 RAID-0、RAID-1 以及 RAID-5。

虚□化， 虚□化

介□了虚□化系□提供的功能， 以及如何配合 FreeBSD 使用它□。

本地化－*I18N/L10N*使用和□置， 本地化

本章描述了如何在 FreeBSD 上使用非英□□言。它涵□了系□和□用程序□的本地化。

更新与升□ *FreeBSD*， 更新与升□ ***FreeBSD***

介□了 FreeBSD-STABLE、 FreeBSD-CURRENT 以及 FreeBSD □行版本之□的差□。 描述了一般用□如何□跟□□□程并从中受益。 涵□了如何更新用□的系□至□行版最新安全修正的方法。

DTrace, ***DTrace***

本章描述了如何在 FreeBSD 上配置和使用 Sun™ 的 DTrace 工具. □□跟踪可以通□□□的系□分析，□助□出系□性能瓶□。

串口通□， 串行通信

本章解□了如何□接□端和□制解□器到 FreeBSD 系□，包括□入和□出□接。

PPP 和 *SLIP*， ***PPP*** 和 ***SLIP***

本章描述了如何用 FreeBSD 通□使用 PPP，SLIP 或者基于以太网的 PPP (PPPoE) 来□接□程系□。

□子□件， □子□件

本章解□了一个 email 服□器的不同□成部分并且□□□□了□于最流行的 mail 服□器□件 sendmail 的配置。

网□服□， 网□服□

提供了□□的指引和示□配置文件以□明如何将一台 FreeBSD 机器作□网□文件系□服□器， 域名服□器， 网□信息服□器或□□同□服□器来使用的方法。

防火□， 防火□

解□了基于□件的防火□的原理， 并提供了□于配置 FreeBSD 上的几□防火□的□□□明。

高□网□□用， 高□网□□用

描述了□多□于网□的主□，包括如何在□的局域网中共享 Internet □接， 高□路由□□， 无□网□，

Bluetooth®， ATM， IPv6以及□多高□□□。

□取 *FreeBSD*， □取 ***FreeBSD***

列出了□得 FreeBSD 安装 CDROM 或 DVDROM 的不同□源，也提供了允□□自由下□ FreeBSD 的不同 Internet 站点。

参考□目， 参考□目

由于本手册触及到了很多不同的主□，因而可能引□□想要□取更多□□的□解。 参考□目列出了很多写作□本□□参考的好□。

Internet 上的□源， ***Internet*** 上的□源

□述了很多□ FreeBSD 用□有用的能□提出□□并□行技□交流的□于 FreeBSD 的□□。

PGP 公□， ***PGP*** 公□

列出了一些 FreeBSD □□者的 PGP □名公□。

本□中使用的一些□定

□了使本□保持一致性和易□性特做了以下□定：

排版□定

斜体

斜体 字用来表示文件名、URLs、□□文字和□□的主流用法。

等□

等□ 字体用来表示□□信息、 命令、 □境□量、 port 的名字、 主机名、 用□名、 □名、 □□名、 □量名，以及代□片断。

粗体

粗体 字用来表示□用程序、命令和□□字。

用□□入

按□用粗体来突出于其他文本。 □合□意味着字用+□接□，同□的按下它□，例如：

`Ctrl` + `Alt` + `Del`

表示□□□同□按下 `Ctrl`， `Alt` 和 `Del` □。

按□序依次□入的□□字通常是用逗号隔□，例如：

`Ctrl` + `X`, `Ctrl` + `S`

□意味着用□□□同□按 `Ctrl` 和 `X`，然后同□按 `Ctrl` 和 `S`。

示例

以 E:\> □□的例子代表一个 MS-DOS® 命令。除非□有□明，□些命令都可以在一个□代的 Microsoft®

Windows® "命令行"□□境被□行。

```
E:\> tools\fdimage floppies\kern.flp A:
```

以 # □□的例子代表必□以 FreeBSD 超□用□身□□行的命令。□可以用 root 身□登□来□入□些命令，或者以普通□号登□然后用 su(1) 来□得超□用□□限。

```
# dd if=kern.flp of=/dev/fd0
```

以 % □□的例子代表命令□□被普通□号□行。 除非□有□明， 在□置□境□量和使用的其他 shell 命令均□ C-shell □法。

```
% top
```

致□

□所看到的□本□是全球几百人努力的□果。 无□他□只是□正一些□□或提交完整的章□，所有的□献都是非常有用的。

一些公司通□提供□金□作者□注于文□□□、 提供出版□金等等方式来支持文□□□。 其中， BSDi (后并入http://www.windriver.com[Wind River Systems]) □助 FreeBSD 文□□□成□来□□改善□本□直到 2000 年三月第一个印刷版 (ISBN 1-57176-241-8) 的出版。 Wind River Systems 同□□助其他作者来□□出□□做很多改□和□文章添加一些附加章□。 □□工作□束于 2001 年 11 月印刷第二版 (ISBN 1-57176-303-1)。在 2003-2004 □年中，http://www.freebsdmall.com[FreeBSD Mall]， 向□改□□本手册以使其第三版印刷版本能□出版的志□者支付了□酬。

部分 I: 起□

手册的以下章□主要是□□□□始使用 FreeBSD 的用□及管理□:

- FreeBSD 入□。
- 安装□程向□。
- 教□ UNIX® 基本知□和基本原理。
- 展示如何在 FreeBSD 上安装大量的第三方□用程序。
- 介□使用 X，UNIX® □口系□， 以及□一些能□提高工作效率的□面□境配置□□。

我□□□用最少的□数来保持前言的索引，以至于可以用最少翻□次数将□手册从□至尾□□。

Chapter 1. 介□

1.1. 概述

非常感□□□ FreeBSD 感□趣！ 下面的章□涵□了 FreeBSD □目的各个方面， 比如它的□史、目□、□□模式，等等。

□□完□章，□将了解：

- FreeBSD 与其它□算机操作系□的□系。
- FreeBSD □目的□史。
- FreeBSD □目的目□。
- FreeBSD □放源代□□□模式的基□。
- 当然□有："FreeBSD" □个名称的由来。

1.2. □迎来到 FreeBSD 的世界!

FreeBSD 是一个支持 Intel (x86 和 Itanium®)，AMD64， Sun UltraSPARC® □算机的基于 4.4BSD-Lite 的操作系□。 到其他体系□□的移植也在□行中。 □也可以□□ FreeBSD 的□史， 或者最新的□行版本。 如果□有意捐助(代□， 硬件，基金)，□看□ FreeBSD 提供□助□篇文章。

1.2.1. FreeBSD 能做些什□？

FreeBSD 有□多非凡的特性。其中一些是：

- _□占式多任□_与□□□先□□整□保在□用程序和用□之□平滑公正的分享□算机□源， 即使工作在最大的□□之下。
- 多用□□□ 使得□多用□能□同□使用同一 FreeBSD 系□做各□事情。 比如， 像打印机和磁□□□器□□的系□外□，可以完全地在系□或者网□上的所有用□之□共享， 可以□用□或者用□□□行个□的□源限制， 以保□□界系□□源不被□用。
- 符合□界□准的□大 *TCP/IP* 网□ 支持， 例如 SCTP、 DHCP、 NFS、 NIS、 PPP， SLIP， IPsec 以及 IPv6。 □意味着□的 FreeBSD 主机可以很容易地和其他系□互□， 也可以作□企□的服□器，提供重要的功能， 比如 NFS(□程文件□□)以及 email 服□， 或将□的□□接入 Internet 并提供 WWW，FTP，路由和防火□(安全)服□。
- _内存保□_□保□用程序(或者用□)不会相互干□。 一个□用程序崩□不会以任何方式影□其他程序。
- FreeBSD 是一个 *32* 位_操作系□ *(*在 *Itanium*®，*AMD64*，和 *UltraSPARC*® 上是_*64* 位)， 并且从□始就是如此□□的。
- □界□准的 *X Window* 系□ (X11R7)□便宜的常□ VGA □示□和□□器提供了一个□形化的用□界面(GUI)，并且完全□放代□。
- 和□多 Linux，SCO，SVR4，BSDI 和 NetBSD 程序的_二□制代□兼容性_
- 数以千□的 *ready-to-run* □用程序可以从 FreeBSD *ports* 和 *packages* 套件中□到。 □可以□利地从□里□到， 何□搜索网□？
- 可以在 Internet 上□到成千上万其它 *easy-to-port* 的□用程序。 FreeBSD 和大多数流行的商□ UNIX®

代□□兼容， 因此大多数□用程序不需要或者只要很少的改□就可以□□。

- □式□求_虚□内存_和"集成的 VM/buffer □存"□□有效地□足了□用程序巨大的内存需求并依然保持其他用□的交互式□□。
- *SMP* 提供□多□理器的支持。
- 内建了完整的 *C*、 *C++*、 *Fortran* □□工具。 □多附加的用于高□研究和□□的程序□言， 也可以在通□ ports 和 packages 套件□得。
- 完整的系□_源代□_意味着□□□□境的最大程度的控制。 当□□有了一个真正的□放系□□， □什□□要受困于私有的解决方案， 任商□公司□布□？
- □富的_在□文□_。
- 不□如此！

FreeBSD 基于加州大学伯克利分校□算机系□研究□ (CSRG) □布的 4.4BSD-Lite， □承了 BSD 系□□□的□良□□。 除了 CSRG □秀的工作之外， FreeBSD □目花□了非常多的□□来□化□整系□， 使其在真□□□情况下□有最好的性能和可□性。 在□今， □多商□巨人正□□ PC 操作系□□加新功能、 提升和改善其可□性， 以便在其上展□激烈□争的同□， FreeBSD □在 已□能□提供所有□一切了！

FreeBSD 可以提供的□用事□上□局限于□的想象力。 从□件□□到工厂自□化，从存□控制到遥□的人造□星天□方位控制， 如果商□的 UNIX® □品可以做到， 那□就非常有可能□也可以用 FreeBSD 来做！ FreeBSD 也□大地受益于全世界的研究中心和大学□□的数以千□的高□量的□用程序， □些程序通常只需要很少的花□甚至免□。 可用的商□□用程序， □天也都在大量地□加。

因□ FreeBSD 自身的源代□是完全公□的， 所以□于特定的□用程序或□目，可以□系□□行最大限度的定制。 □□于大多数主流的商□生□商的操作系□来□几乎是不可能的。 以下是当前人□□用 FreeBSD 的某些程序的例子：

- *Internet* 服□： FreeBSD 内建的□大的 TCP/IP 网□使它得以成□各□ Internet 服□的理想平台， 比如：
  - FTP 服□器
  - World Wide Web 服□器(□准的或者安全的 [SSL])
  - IPv4 and IPv6 路由
  - 防火□和 NAT("IP □装") 网□
  - □子□件服□器
  - USENET 新□□和□子布告□系□
  - □有□多...

    使用 FreeBSD， □可以容易地从便宜的 386 □ PC 起□，并随着□的企□成□，一路升□到□有 RAID 存□的四路 Xeon 服□器。
- 教育： □是一名□算机科学或者相□工程□域的学生□？ 学□操作系□，□算机体系□□和网□没有比在 FreeBSD 可提供的体□下□手□践更好的□法了。□多可自由使用的 CAD、数学和□形□□包也使它□于那些主要□趣是在□算机上完成 _其他_工作的人非常有□助。
- 研究： 有完整的系□源代□，FreeBSD □于操作系□研究以及其他□算机科学分支都是一个□好的平台。 FreeBSD 可自由□得的本性， 同□可以使□在不同地方的□□□□在□放的□□上□□□□、 交流想法与合作□□成□可能， 且不必担心特□的版□□定或者限制。
- _网□：_需要一个新的路由器？ 一台域名服□器 (DNS)？ 一个隔□□的内部网□的防火□？ FreeBSD

可以容易的把□□在角落不用的 386 或者 486 PC □成一台完善的□包□□能力的高□路由器。

- *X Window* 工作站： FreeBSD 是廉价 X □端的一□□佳解决方案， □可以□□使用免□的 X11 服□器。 与 X □端不同，如果需要的□ FreeBSD 能□在本地直接□行程序， 因而□少了中央服□器的□担。 FreeBSD 甚至能□在 "无□" □境下□□， □使得□端更□便宜和易于管理。
- □件□□： 基本的 FreeBSD 系□□有包括著名的 GNU C/C++ □□器和□□工具在内的一整套□□工具。

FreeBSD 可以通□ CD-ROM、DVD， 以及匿名 FTP 以源代□和二□制方式□得。□□看□取 FreeBSDf 了解□取 FreeBSD 的更多□□。

1.2.2. □在使用 FreeBSD?

FreeBSD 被世界上最大的 IT 公司用作□□和□品的平台， 包括：

- Apple
- Cisco
- Juniper
- NetApp

FreeBSD 也被用来支持 Internet 上一些最大的站点， 包括：

- Yahoo!
- Yandex
- Apache
- Rambler
- 新浪网
- Pair Networks
- Sony Japan
- Netcraft
- NetEase
- Weathernews
- TELEHOUSE America
- Experts Exchange

等等□多。

1.3. □于 FreeBSD □目

下面的章□提供了□目的一些背景信息， 包括□要的□史、□目目□、以及□目□□模式。

1.3.1. FreeBSD 的□要□史

FreeBSD □目起源于 1993 年早期， 部分作□ "Unofficial 386BSD Patchkit" 的副□物，patchkit 的最后 3 个□□□□人是：Nate Williams，Rod Grimes 和我。

我□最初的目□是做出一□ 386BSD 的□□版以修正一些 Patchkit 机制无法解决的□□(bug)。 很多人可能□□得早期的□目名称叫做 "386BSD 0.5" 或者 "386BSD Interim" 就是□个原因。

386BSD 是 Bill Jolitz 的操作系□， 到那□已被□重地忽□了一年之久。 由于 Patchkit 在□去的□一天里都在急□膨□， 使得□其□行消化吸收□得越来越困□， 因此我□一致同意□□做些事情并决定通□提供□个□□的 "cleanup" 版本来□助 Bill。 然而，Bill 却在事先没有指出□个□目□□如何□展下去的情况下， 突然决定退出□个□目，最□□个□□只好被迫停止。

没□多久， 我□□□即便没有 Bill 的支持， □目仍有保留的价□， 因此，我□采用了 David Greenman 的意□，□其命名□ "FreeBSD"。在和当□的几个用□商量后， 我□提出了最初的目□， 而□件事明朗化后， □个□目就走上了正□，甚至可能成□□□。 □了拓展 FreeBSD 的□行渠道，我抱着□□看的心□， □系了光□商 Walnut Creek CDROM， 以便那些上网不方便的用□得到 FreeBSD。 Walnut Creek CDROM 不□支持□行 FreeBSD 光□版的想法， □□□个□□提供了所需的□算机和高速网□接入。 在那□， 若没有 Walnut Creek CDROM □一个完全未知的□目的空前信任， FreeBSD 不太可能像它今天□□，影□如此深□， □展如此快速。

第一个 CD-ROM (以及在整个互□网□□内□行的) □行版本是 FreeBSD 1.0，于 1993 年 10 月□布。□个版本基于 U.C. Berkeley 的 4.3BSD-Lite("Net/2")磁□， 也有□多□件是 386BSD 和自由□件基金会提供的。 □于第一次□行，□算是相当成功了。 在 1994 年 5 月，我□□布了更加成功的 FreeBSD 1.1 版。

在□段□□， □生了一些意外的情况。 Novell 和 U.C. Berkeley 就 Berkeley Net/2 磁□知□□□的□拉松式的官司□成了和解。 和解中的一部分是 U.C. Berkeley 作出的□□， 令 Net/2 中的一大部分内容成□ "受限的 (encumbered)" 和属于 Novell 知□□□的代□， 而后者在不久前□□从 AT&T 收□了□些□□； 作□回□， Berkeley 得到了来自 Novell 的 "□□"， 在 4.4BSD-Lite 版本正式□布□， 可以声明□不受限的 (unencumbered)，□有的 Net/2 用□□□烈建□□移到□个版本。 □包括了 FreeBSD， 而我□的□目□被允□在 1994 年 6 月底之前□□□行基于 Net/2 的□品。 根据和解□□， 在最后期限之前我□□布了一个最□版本， □个版本是 FreeBSD 1.1.5.1。

接下来， FreeBSD □始了□苦的从全新的、 不太完整的 4.4BSD-Lite 重新□写自己的□程。 "Lite" 版本中，Berkeley 的 CSRG □除了用于□系□能□引□的一大部分代□ (由于各□各□的法律需求)， 而当□ 4.4 在 Intel 平台的移植版本□有很多工作没有完成。 直到 1994 年 11 月， 我□的□目才完成了□□□渡， 并通□网□以及 CD-ROM (在 12 月底) 上□布了 FreeBSD 2.0。 尽管系□中□有很多比□粗□的地方， □个版本□是取得了巨大的成功， 并在 1995 年 6 月□布了更□大和易于安装的 FreeBSD 2.0.5 版本。

我□于 1996 年 8 月□布了 FreeBSD 2.1.5 版本， 它在 ISP 和商□□体中非常流行。 随后， 2.1-STABLE 分支的□一个版本□□而生，它就是 FreeBSD 2.1.7.1，在 1997 年 2 月□布并停止了 2.1-STABLE 的主流□□。□在，它□于□□状□， □□提供安全性的□□和其他□重的□□修□的□□(RELENG_2_1_0)。

FreeBSD 2.2 版作□ RELENG_2_2 分支，于 1996 年 11 月从□□主□ ("-CURRENT")分出来。它的第一个完整版(2.2.1)于 1997 年 4 月□布出来。 97 年夏秋之□，□着 2.2 分支的更□一□的版本在□□。其最后一版(2.2.8)于 1998 年 11 月□布出来。 第一个官方的 3.0 版本出□在 1998 年 10 月， 意味着 2.2 分支□束的□始。

1999 年 1 月 20 日又出□了新的分支，就是 4.0-CURRENT 和 3.X-STABLE 分支。从 3.X-STABLE 起，3.1 在 1999 年 2 月 15 日□行，3.2 在1999 年 5 月 15 日，3.3 在 1999 年 9 月 16 日，3.4 在 1999 年 12 月 20 日，3.5 在 2000 年 6 月 24 日，接下来几天后□布了很少的修□升□至 3.5.1，加入了□ Kerberos 安全性方面的修□。 □是 3.X 分支最后一个□行版本。

随后在 2000 年 3 月 13 日出□了一个新的分支， 也就是 4.X-STABLE。 □之后□布了□多的□行版本： 4.0-RELEASE 于 2000 年 3月□布， 而最后的 4.11-RELEASE □是在 2005 年 1 月□布的。

期待已久的 5.0-RELEASE 于 2003 年 1 月 19 日正式□布。 □是将近三年的□□的□峰之作， 同□也□志了 FreeBSD 在先□的多□理器和□用程序□程支持的巨大成就， 并引入了□于 UltraSPARC® 和 ia64 平台的支持。 之后于 2003 年 6 月□布了 5.1。 最后一个从 -CURRENT 分支的 5.X 版本是 5.2.1-RELEASE， 它在 2004 年 2 月正式□布。

RELENG_5 于 2004 年 8 月正式□建， □随其后的是 5.3-RELEASE， 它是 5-STABLE 分支的□志性□行版。 □个分支的最后一个版本， 5.5-RELEASE 是在 2006 年 5 月□布的。 RELENG_5 分支不会有后□的□行版了。

其后在 2005 年 7 月又建立了 RELENG_6 分支。 而 6.X 分支上的第一个版本， 即 6.0-RELEASE， □是在 2005 年 11 月□布的。 □个分支的最后一个版本， 6.4-RELEASE 是在 2008 年 11 月 □布的。 RELENG_6 分支上不再会有□布版本了。 □是最后一个支持 Alpha 硬件架□的版本。

RELENG_7 分支于 2007 年 10 月□建。 第一个□个分支的□行版是 7.0-RELEASE， □个版本是 2008 年 2 月 □布的。 最新的 11.2-RELEASE 是在 June 28, 2018 □布的。 RELENG_7 将不会有其它后□的□布版本。

其后在 2009 年 8 月又建立了 RELENG_8 分支。 8.X 分支的第一个版本， 8.0-RELEASE 是在 2009 年 11 月□布的。 最新的 12.0-RELEASE 于 December 11, 2018 □布。 RELENG_8 □将会有其它后□的□布版本。

目前， 中□期的□□□目□□在 9.X-CURRENT (主干, trunk) 分支中□行， 而 9.X 的 CD-ROM (当然， 也包括网□) 快照版本可以在 快照服□器 □到。

1.3.2. FreeBSD □目目□

FreeBSD □目的目□是无附加条件地提供能□用于任何目的的□件。 我□中的□多人□代□ (以及□目本身) 都有非常大的投入， 因此当然不介意偶□有一些□金上的□□， 但我□并没打算□决地要求得到□□□助。 我□□□我 □的首要 "使命" 是□任何人提供代□， 不管他□打算用□些代□做什□， 因□□□代□将能□被更广泛地使用， 从而最大限度地□□其价□。 我□□□是自由□件最基本的， 同□也是□我□所倡□的一个目□。

我□源代□□中， 以 GNU 公共□可□ (GPL) 或者 GNU 函数□公共□可□ (LGPL) □布的那些代□□有少 □的附加限制， □好只是□制性的要求□放代□而不是□的。 由于使用 GPL 的□件在商□用途上会□加若干□□性， 因此，如果可以□□的□， 我□更偏好使用限制相□更□松的 BSD 版□来□布□件。

1.3.3. FreeBSD □□模式

FreeBSD 的□□是一个非常□放且有有伸□性的□程， 就像从我□的 □ 献者列表里看到的，它是完全由来自全世界的数以百□的□献者□展起来的。 FreeBSD 的□□基□□□允□数以百 □的□□者通□互□网□同工作。 我□也□常□注着那些□我□的□□感□趣的新□□者和新的□意， 那些有□趣更□一□参与 □目的人只需要在 FreeBSD 技□□□□件列表 □系我□。 FreeBSD 公告□件列表 □那些希望了解我□工作所□及到 □些□域的人也是有用的。

无□是独立地工作或者封□式的□□工作， 了解 FreeBSD □□和它的□□□程都是有益的：

SVN 和 CVS 代□□

在□去的几年中 FreeBSD 的中央源代□□是由 CVS (并行版本控制系□)来□□的，CVS 是一个与 FreeBSD □□的可自由□得的源代□控制工具。自 2008 年六月起， □个□目□始□□使用SVN (Subversion)。 □次□□被 □□是非常必要的，因□ CVS 的□于快速□展源代□□和□史□□的限制越□明□。□在主源□□使用 SVN，客 □端的工具像 CVSup 和 csup □些依□于旧的 CVS 基□□□依然可以使用 - 因□□于 SVN 源□□的修改会被□回□ CVS。 目前只有中央原代□□是由 SVN 控制的。文□，万□网和 Ports □□仍旧使用着 CVS。 The primary repository resides on a machine in Santa Clara CA, USA 主 CVS 代□□放置在美国加利福尼□州 □克拉拉的一台机器上， 它被□制到全世界的大量□像站上。包含 -CURRENT 和 -STABLE 的 SVN □也同

□能非常容易的□的机器上。 □参□ 同□□的源□□ □得更多的相□信息。

committer 列表

committer 是那些□ CVS □有_写_□限的人， 他□被授□修改 FreeBSD 的源代□ (□□ "committer" 来自于 cvs(1) 的 `commit` 命令，□个命令用来把新的修改提交□ CVS 代□□)。提交修正的最好方法是使用 send-pr(1) 命令。如果□□□在系□中出□了一些□□的□， □也可以通□□件将它□□送至 FreeBSD committer 的□件列表。

FreeBSD 核心□□

如果把 FreeBSD □目看作一家公司，那□ _FreeBSD 核心□□_就相当于董事会。 核心□□的主要任□是提出□体上的□展□□，然后□定一个正□的方向。 □□那些富有献身精神和可□的□□者加入到 committer □伍中来也是核心□□的工作之一， □些新的成□将作□新核心□□成□和其他人一起□□前□。 当前的核心□□是 2010 年 7 月从 committer 中□□□生的。□□□□年一次。

一些核心□□的成□□□□特定的□任□□， 也就是□他□必□尽力□保某个子系□能工作正常。 FreeBSD □□者的完整列表和他□的□任□□，□参□□献者列表

核心□□的大部分成□加入 FreeBSD □□的□候都是志□的， 并没有从□目中□得任何□政上的□助， 所以"承□"不□□被理解□"支持保□"。 前面所述"董事会"的□推并不十分准□，或□更好的□法是，他□是一群□意放□他□的生活， 投身于 FreeBSD □目而非□□其个人更好的生活的人！

外□□献者

事□上，最大的□□□□正是□我□提供反□和□□修□的用□自己。 FreeBSD 的非集中式的□□者保持□系的主要方式就是□□ FreeBSD 技□□□□件列表，很多事情在那里□□。□看Internet上的□源了解□多 FreeBSD □件列表的更多信息。

FreeBSD □献者列表 很□并在不断□□， □什□不加入它来□ FreeBSD 做□献□？

提供代□不是□□个□□做□献的唯一方式； 有一个更完整的需要做的事情的列表，可以参□ FreeBSD □目网站。

□的来□，我□的□□模式好像是一□没有拘束的同心□。 □□集中式的□□模式，主要是考□到 FreeBSD _用□_的方便， 同□□他□能很容易地□□同一□□件， 而不会把潜在的□献者排除在外！ 我□的目□是提供一个包含有大量具有一致性 □用程序的□定的操作系□， 以利于用□的安装和使用，- □□模式在完成目□的□程中工作得非常有效。

我□□于那些想要加入，成□ FreeBSD □□者的期待是： 具有如同当前其他人一□的投入，来□保持□的成功！

1.3.4. 最新的 FreeBSD □行版本

FreeBSD 是一个免□使用且□有完整源代□的基于 4.4BSD-Lite 的系□， 它广泛□行于 Intel i386™、i486™、Pentium®、 Pentium® Pro、 Celeron®、 Pentium® II、 Pentium® III、 Pentium® 4(或者兼容系□)、 Xeon™、 和 Sun UltraSPARC® 的□算机系□上。 它主要以 加州大学伯克利分校 的 CSRG 研究小□的□件□基□，并加入了 NetBSD、OpenBSD、386BSD 以及来自 自由□件基金会 的一些□西。

自从 1994 年末我□的 FreeBSD 2.0 □行以来， FreeBSD 的性能，可定制性，□定性都有了令人注目的提高。最大的□化是通□整合虚□内存/文件系□中的高速□存改□的虚□内存系□， 它不□提升了性能，而且□少了 FreeBSD □内存的需要， 使得 5 MB 内存成□可接受的最小配置。 其他的改□包括完整的 NIS 客□端和服

□器端的支持， 事□式 TCP □□支持，按需□号的 PPP，集成的 DHCP 支持，改□的 SCSI 子系□， ISDN 的支持，ATM，FDDI，快速 Gigabit 以太网(1000 Mbit)支持， 提升了最新的 Adaptec 控制器的支持和修□了很多的□□。

除了最基本的系□□件，FreeBSD □提供了一个□有成千上万广受□迎的程序□成的□件的 Ports Collection。到本□付印□，已有超□ 36000 个 ports (ports 包括从 http(WWW) 服□器到游□、程序□□□言、□□器以及□能想到的几乎所有的□西)。 完整的 Ports Collection 大□需要 500 MB 的存□空□。所有的只提供□原始代□的"修正"。□使得我□能□容易地更新□件， 而且□少了老旧的 1.0 Ports Collection □硬□空□的浪□。 要□□一个 port，□只要切□到□想要安装的程序的目□， □入 `make install`，然后□系□去做剩下的事情。 □要□□的□一个程序完整的原始代□可以从 CD-ROM 或本地 FTP □得，所以□只需要□□□想要□件的足□的磁□空□。几乎大多数的□件都提供了事先□□好的 "package" 以方便安装，□于那些不希望从源代□□□他□自己的 ports 的人只要使用一个□□的命令 (`pkg_add`)就可以安装。 有□ package 和 ports 的更多信息可以在安装□用程序. Packages 和 Ports中□到。

□可以在最近的 FreeBSD 主机的 /usr/shared/doc 目□下□到□多有用的文件来□助□安装及使用 FreeBSD。□也可以用一个 HTML □□器来□□本地安装的手册， 使用下面的 URL：

FreeBSD 使用手册

/usr/shared/doc/handbook/index.html

FreeBSD FAQ

/usr/shared/doc/faq/index.html

□也可以□看在 http://www.FreeBSD.org/ 的主站上的副本。

Chapter 2. 安装 FreeBSD

2.1. 概述

FreeBSD 提供了一个以文字□主，□□好用的安装程序，叫做 sysinstall 。□是 FreeBSD 默□使用的安装程序；厂商如果想，也可以提供□合自己需要的安装程序。本章□明如何使用 sysinstall 来安装 FreeBSD。

学□完本章之后，□将会知道：

- 如何制作 FreeBSD 安装磁□
- FreeBSD如何参照及分割□的硬□
- 如何□□ sysinstall.
- 在□行 sysinstall □□将要回答的□□、 □□代表什□意□，以及□如何回答它□。

在□□本章之前，□□□：

- □□□要安装的 FreeBSD 版本所附的硬件支持列表以□定□的硬件有没有被支持。

一般来□，此安装□明是□□ i386™ ("PC 兼容机") 体系□□的□□。如果有其它体系□□的安装□明， 我□将一并列出。□然本文□□常保持更新， 但有可能与□安装版本上所□的□明文□有些□出入。 在□里建□□使用本□明文章作□一般性的安装指□参考手册。

2.2. 硬件需求

2.2.1. 最小配置

安装 FreeBSD 所需的最小硬件配置， 随 FreeBSD 版本和硬件架□不同而有所不同。

在接下来的几□中， □出了□些信息的一些□□。随□安装 FreeBSD 的方式不同， 可能需要使用□□或□ FreeBSD 支持的 CDROM □□器， 有□候也可能需要的是一□网□。 □将在 准□引□介□ 中□行介□。

2.2.1.1. FreeBSD/i386 和 FreeBSD/pc98

FreeBSD/i386 和 FreeBSD/pc98 版本， 都需要 486 或更高的□理器，以及至少 24 MB 的 RAM。 □需要至少 150 MB 的空□硬□空□， 才能完成最小的安装配置。

□于老旧的硬件而言， 多数□候， 装配更多的 RAM 和□出更多的硬□空□，要比使用更快的□理器更有用。

2.2.1.2. FreeBSD/amd64

有□□□理器同□能□支持□行 FreeBSD/amd64。 第一□是 AMD64 □理器， 包括 AMD Athlon™64、 AMD Athlon™64-FX、 AMD Opteron™ 以及更高□□的□理器。

能□使用 FreeBSD/amd64 的□一□□理器是包含了采用 Intel® EM64T 架□支持的□理器。 □□□理器包括 Intel® Core™ 2 Duo、 Quad、 以及 Extreme 系列□理器， 以及 Intel® Xeon™ 3000、 5000、 和 7000 系列

□理器。

如果□的□算机使用 nVidia nForce3 Pro-150， □ 必□ 使用 BIOS 配置， 禁用 IO APIC。 如果□没有□到□□的□□， 可能就只能□而禁用 ACPI 了。 Pro-150 芯片□存在一个 bug， 目前我□□没有□到□□□一□□的方法。

2.2.1.3. FreeBSD/sparc64

要安装 FreeBSD/sparc64， 必□使用它支持的平台 (参□ 支持的硬件)。

FreeBSD/sparc64 需要独占一□磁□。 目前□没有□法与其它操作系□共享一□磁□。

2.2.2. 支持的硬件

支持的硬件列表， 会作□ FreeBSD □行版本的 FreeBSD 兼容硬件□明提供。 □个文□通常可以在 CDROM 或 FTP 安装文件的□□目□□到， 它的名字是 HARDWARE.TXT， 此外， 在 sysinstall 的 documentation 菜□也可以□到。它□□特定的硬件架□列出了 FreeBSD 已知支持的硬件。 不同□行版本和架□上的硬件支持列表，可以在 FreeBSD 网站的 □行版信息 □面上□到。

2.3. 安装前的准□工作

2.3.1. 列出□□□的硬件清□

在安装 FreeBSD 之前，□□□□着将□□□中的硬件清□列出来。 FreeBSD 安装程序会将□些硬件(磁□、网□、光□等等) 以及型号及制造厂商列出来。FreeBSD 也会□□□□些□□□出最□当的 IRQ 及 IO 端口的□定。但是因□ PC 的硬件□□□在太□□□， □个□□不一定□是能成功。□□， □就可能需要手□更改有□□的□□的□定□。

如果□已□安装了其它的操作系□，如 Windows® 或 Linux， 那□□可以先由□些系□所提供的工具来□看□的□□□定□是□□分配的。 如果□真的没□法□定某些接口□用什□□定□，那□□可以□□看看， □不定它的□定已□□示在□上。常用的 IRQ 号□□ 3、5 以及 7； IO 端口的□通常以 16 □制位表示，例如 Ox330。

我□建□□在安装 FreeBSD 之前把□些信息打印或□□下来，做成表格 的□子也□会比□有□助，例如：

表 *1.* 硬件□□清□

□□名	**IRQ**	**IO** 端口号	□注
第一□硬□	N/A	N/A	40 GB，Seagate 制造，第一个 IDE 接口主□□
CDROM	N/A	N/A	第一个 IDE 接口从□□
第二□硬□	N/A	N/A	20 GB，IBM 制造, 第二个 IDE 接口主□□
第一个 IDE 控制器	14	0x1f0	
网□	N/A	N/A	Intel® 10/100
Modem	N/A	N/A	3Com® 56K faxmodem，位于 COM1 口

在清楚地了解了□□算机的配置之后， 需要□□它是否符合□希望安装的 FreeBSD 版本的硬件需求。

2.3.2. □□□的数据

如果□的□□上面存有重要的数据□料， 那□在安装 FreeBSD 前□□定□已□将□些□料□□了， 并且先□□□些□□文□是否有□□。FreeBSD 安装程序在要写入任何□料到□的硬□前都会先提醒□□□， 一旦□□定要写入，那□以后就没有反悔的机会。

2.3.3. 决定要将 FreeBSD 安装到□里

如果□想□ FreeBSD 使用整个硬□，那□□直接跳到下一□。

但是，如果□想□ FreeBSD 跟□已有的系□并存， 那□□必□□□数据存在硬□的分布方式有深入的了解，以及其所造成的影□。

2.3.3.1. FreeBSD/i386 体系□□的硬□分配方式

一个 PC 硬□可以被□分□□多□。 □些□被称□ *partitions* *(分区)*。 由于 FreeBSD 内部也有分区的概念，如此命名很容易□致混□， 因此我□在 FreeBSD 中，将其称□磁□ slice，或□称□ slices。 例如， FreeBSD 提供的用于操作 PC 磁□分区的工具 `fdisk` 就将其称□ slice 而不是 partition。由于□□的原因， □个硬□□支持四个分区； □些分区叫做 主分区*(Primary partion)*。 □了突破□个限制以便能使用更多的分区，就有了新的分区□型，叫做 □展分区*(Extended partition)*。 一个硬□可以□有一个□展分区。在□展分区里可以建立□多个所□的 □□分区*(Logical partitions)*。

□个分区都有其独立的 分区号*(partition ID)*, 用以区分□个分区的数据□型。FreeBSD 分区的分区号□ `165`。

一般而言，□□操作系□都会有自己独特的方式来区□分区。 例如 DOS 及其之后的 Windows®， 会分配□□个主分区及□□分区一个 □□器字符， 从 C: □始。

FreeBSD 必□安装在主分区。FreeBSD 可以在□个分区上面存放系□数据或是□建立的任何文件。 然而，如果□有多个硬□，□也可以在□些硬□上(全部或部分)建立 FreeBSD 分区。在□安装 FreeBSD 的□候，必□要有一个分区可以□ FreeBSD 使用。 □个分区可以是尚未□□的分区， 或是已□存在且存有数据但□不再需要的分区。

如果□已□用完了□硬□上的所有分区， 那□□必□使用其它操作系□所提供的工具 (如 DOS 或 Windows® 下的 `fdisk`) 来□出一个分区□ FreeBSD 使用。

如果□的某个分区有多余的空□，□可以使用它。 但是使用前□需要先整理一下□些分区。

FreeBSD 最小安装需要□ 100 MB 的空□，但是□□是 非常 基本的安装， 几乎没有剩下多少空□可以建立□自己的文件。一个□理想的最小安装是 250 MB，不含□形界面；或是 350 MB 以上，包含□形界面。 如果□□需要安装其它的第三方厂商的套件， 那□将需要更多的硬□空□。

□可以使用□似 PartitionMagic® □□的商□版本工具， 或□似 GParted □□的自由□件工具来□整分区尺寸，从而□ FreeBSD □出空□。 PartitionMagic® 和 GParted 都能改□ NTFS 分区的尺寸。 GParted 在□多 Live CD Linux □行版， 如 SystemRescueCD 中均有提供。

目前已□有□告□示改□ Microsoft® Vista 分区尺寸□会出□□□。 在□行此□操作□， 建□□准□一□ Vista 安装 CDROM。如同其他的磁□□□操作一□， □烈建□□事先□行□□。

不当的使用□些工具可能会□掉□硬□上的数据□料！ 在使用□些工具前□定□有最近的、没□□的□□数据。

例 *1.* 使用已存在的分区

假□□只有一个 4GB 的硬□，而且已□装了 Windows® 然后□将□个硬□分成□个分区 C: 跟 D:，□个分区大小□ 2 GB。在 C: 分区上存放有 1 GB 的数据、 D:分区上存放 0.5 GB 的数据。

□意味着□的□上有□个分区，一个□□器符号是一个分区 (如 c:、d:)。 □可以把所有存放在 D: 分区上的数据□制到 C: 分区, □□就空出了一个分区(d:)□ FreeBSD 使用。

例 *2.* □□已□在的分区

假□□只有一个 4 GB 的硬□，而且已□装了 Windows®。 □在安装 Windows® 的□候把 4 GB 都□了 C: 分区，并且已□使用了 1.5 GB 的空□。□想将剩余空□中的 2 GB □ FreeBSD 使用。

□了安装 FreeBSD，□必□从下面□□方式中□□一□：

1. □□ Windows® 的数据□料，然后重新安装 Windows®， 并□ Windows® 分配 2 GB 的空□。
2. 使用上面提及的 PartitionMagic® 来整理或切割□的分区。

2.3.4. 收集□的网□配置相□□料

如果□想通□网□(FTP 或是 NFS)安装 FreeBSD， 那□□就必□知道□的网□配置信息。在安装 FreeBSD 的□程中将会提示□□入□些□料，以□利完成安装□程。

2.3.4.1. 使用以太网或□□/DSL Modem

如果□通□局域网或是要通□网□使用□□/DSL 上网， 那□□必□准□下面的信息：

1. IP 地址。
2. 默□网□ IP 地址。
3. 主机名称。
4. DNS 服□器的 IP 地址。
5. 子网掩□。

如果□不知道□些信息， □可以□□系□管理□或是□的网□服□提供者。 他□可能会□□些信息会由 *DHCP* 自□分配；如果□□的□，□□住□一点就可以了。

2.3.4.2. 使用 Modem □接

如果□由 ISP 提供的□号服□上网，□仍然可以通□它安装 FreeBSD，只是会需要很□的□□。

□必□知道：

1. □号到 ISP 的□□号□。

2. □的 modem 是□接到□个 COM 端口。
3. □□号到 ISP 所用的□号和密□。

2.3.5. □□ FreeBSD □行勘□

□然我□尽力□保□个 FreeBSD □行版本的□定性， 但偶□也会有一些□□□入□行版。□少数情况下， □些□□甚至可能会影□安装。 当□□和修正□□之后，它□会列在 FreeBSD 网站中的 FreeBSD □行勘□ 中。 在□安装之前，□□首先看一看□□勘□表，以了解可能存在的□□。

□于所有□出版本的信息，包括勘□表，可以在 FreeBSD 网站 的 □行版信息 一□中□到。

2.3.6. 准□安装介□

FreeBSD 可以通□下面任何一□安装介□□行安装：

安装介□

- CDROM 或 DVD
- USB □□棒
- 在同一□算机上的 DOS 分区
- SCSI 或 QIC 磁□
- □□

网□

- 通□防火□的一个 FTP 站点，或使用 HTTP 代理。
- NFS 服□器
- 一个指定的并行或串行接口

如果□□□了 FreeBSD 的 CD 或 DVD，那□□可以直接□入下一□ (准□引□介□)。

如果□□没有 FreeBSD 的安装文件， □□按照 准□□自己的安装介□ 来准□。 □完那□之后， □就可以回到□□并从 准□引□介□ □□了。

2.3.7. 准□引□介□

FreeBSD 的安装□程□始于将□的□□□机□入 FreeBSD 安装□境 - －并非在其它的操作系□上□行一个程序。□算机通常使用安装在硬□上的操作系□□行引□， 也可以配置成使用一□"bootable(可引□)"的□□□行□□。大多数□代□算机也都可以从光□或 USB □来引□系□。

如果□有 FreeBSD 的安装光□或 DVD(或者是□□□的， 或者是□自己准□的。)并且□的□算机可以从光□□行□□ (通常在 BIOS 中会有 "Boot Order" 或□似的□□可以□置)，那□□就可以跳□此小□。 因□ FreeBSD 光□及 DVD 光□都是可以引□的， 用它□□机□不用做什□特□的准□。

要□建引□系□所需的□□棒， 需按下面的操作□行：

1. □取□□棒映像文件

   □□棒映像文件可以从 *arch* □□的 ISO-IMAGES 目□， 例如 ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/arch/ISO-IMAGES/version/FreeBSD-12.0-RELEASE-arch-memstick.img □得。 其中， *arch* 和 *version* 需要替□□□使用的平台和版本。 例如， FreeBSD/i386 12.0-RELEASE 的□□棒映像位于 ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/ISO-IMAGES/12.0/FreeBSD-12.0-RELEASE-i386-memstick.img。

   □□棒的映像文件□展名是 .img。 在 ISO-IMAGES/ 目□中提供了多□不同的映像， □需要根据使用的 FreeBSD 版本， 有□也包括硬件来□□合□的映像。

   在□□安装之前， □必 □□ □目前保存在 USB □□棒上的数据， 接下来的操作将会擦除 □些数据。

2. 准□□□棒

   下面的例子中， 目□□□棒□□的□□名是 /dev/da0。 □小心地□□□是希望覆□的□□，否□可能会□坏□的□有数据。

   □置 kern.geom.debugflags sysctl □允□写入目□□□的主引□□□。

   ```
   #  sysctl kern.geom.debugflags=16
   ```

3. 将映像文件写入□□棒

   .img 文件 不是 直接□制到□□棒中的那□普通文件。 □个映像是一□包含□□□全部内容的映像。 □意味着□□地从一个地方□制到□一个地方是 不能 □予其引□系□的能力的。 □必□使用 dd(1) 将映像文件直接写入磁□：

   ```
   # dd if=FreeBSD-12.0-RELEASE-i386-memstick.img of=/dev/da0 bs=64k
   ```

一般来□，要建立安装□(□□)□依照下列□□：

1. □取□机□□映像文件

□注意， 从 FreeBSD 8.0 □始， 我□不再提供□□映像了。 □参□前面□于如何使用 USB □□棒， 或 CDROM 和 DVD 来安装 FreeBSD 的介□。

□机□□映像文件可以在□的安装介□的 floppies/ 目□下□到， □外□也可以从下述网站的 floppies 目□下□： ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/架□名/版本-RELEASE/floppies/。 将 <架□名> 和 <版本> 替□□□使用的□算机体系□□和希望安装的版本号。 例如，用于安装 i386™ 上的 FreeBSD/i386 11.2-RELEASE 的文件的地址， □□是 ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/11.2-RELEASE/floppies/。

□□映像文件的□展名是 .flp。 在 floppies/ 目□中包括了□多不同的映像文件， 随□安装的 FreeBSD 版本， 某些□候也随硬件的不同， □需要使用的映像文件可能会有所不同。 □通常会需要四□□□， 即 boot.flp、 kern1.flp、 kern2.flp， 以及 kern3.flp。 □□□同一目□下的 README.TXT 文件以了解□于□些映像文件的最新信息。

□的 FTP 程序必□使用 二□制模式 来下□□些映 像文件。有些□□器只会用 *text* (或_ASCII _) 模式来□□数据， 用□些□□器下□的映像文件做成的□□将无法正常□机。

2. 准□□□

□必□□□下□的□一个映像文件准□一□□□。 并且□避免使用到坏掉的□□。 最□□的方式就是□先将□些□□格式化， 不要相信所□的已格式化的□□。在 Windows® 下的格式化程序不会告□□出□多少坏□，它只是□□的□□它□□ "bad" 并且忽略它□。 根据建□□□□使用全新的□□来存放安装程序。

如果□在安装 FreeBSD 的□程中造成当机、 □□或是其它怪□□象，第一个要□疑的就是引□□□。 □用其它的□□制作映像文件再□□看。

3. 将映像文件写入□□中

.flp 文件 并非 一般的文件，□不能直接将它□□制到□□上。 事□上它是一□包含完整磁□内容的映像文件。□表示□ 不能 □□的使用 DOS 的 copy 命令将文件写到□□上， 而必□使用特□的工具程序将映像文件直接写到□□中。

如果□使用 MS-DOS® 或 Windows® 操作系□来制作引□□， 那□□可以使用我□提供的 fdimage 程序来将映像文件写到□□中。

如果□使用的是光□，假□光□的□□器符号□ E:，那□□□行下面的命令：

```
E:\> tools\fdimage floppies\boot.flp A:
```

重□上述命令以完成□个 .flp 文件的写入， □□一个映像文件都必□更□□□； 制作好的□□□注明是使用□个映像文件做的。 如果□的映像文件存放在不同的地方，□自行修改上面的指令指向□存放 .flp 文件的地方。要是□没有 FreeBSD 光□， □可以到 FreeBSD 的 FTP 站点 tools目□ 中下□。

如果□在 UNIX® 系□上制作□□(例如其它 FreeBSD 机器)， □可以使用 dd(1) 命令来将映像文件写到□□中。 如果□用 FreeBSD,可以□行下面的命令：

```
# dd if=boot.flp of=/dev/fd0
```

在 FreeBSD 中，/dev/fd0 指的是第一个□□(即 A: □□器)； /dev/fd1 是 B: □□器,依此□推。其它的 UNIX® 系□可能会用不同的的名称， □□□就要□□□系□的□明文件。

□□在可以安装 FreeBSD 了

2.4. □始安装

默□情况下, 安装□程并不会改□任何□硬□中的数据， 除非□看到下面的□息：

```
Last Chance: Are you SURE you want continue the installation?

If you're running this on a disk with data you wish to save then WE
STRONGLY ENCOURAGE YOU TO MAKE PROPER BACKUPS before proceeding!

We can take no responsibility for lost disk contents!
```

在看到□最后的警告□息前□都可以随□□□， 安装程序界面不会□更□的硬□。如果□□□有任何□定□□， □□□可以直接将□源□掉而不会造成任何□害。

2.4.1. □机□□

2.4.1.1. 引□ i386™ 系□

1. 从□□尚未□机□始□起

2. 将□□□源打□。□□始的□候它□□会□示□入系□□置菜□或 BIOS 要按□个□，常□的是 `F2`、 `F10`、`Del` 或 `Alt` + `S`。不□是要按□个□，□按它□入 BIOS □置画面。 有□□的□算机可能会□示一个□形画面，典型的做法是按 `Esc` 将□掉□个□形画面， 以使□能□看到必要的□置信息。

3. □到□置□机□序的□□，它的□□□ "Boot Order" 通常会列出一些□□□□□□，例如：Floppy、 CDROM、First Hard Disk 等等。

   如果□要用光□安装，□□□ CDROM。 如果使用 USB □， 或者□□来引□系□， 也□□似地□□□□了正□的引□□□。 如有疑□， □参考□的主板□明手册。

   □存□定并□□，系□□□会重新□□。

4. 如果□根据 准□引□介□ 制作了 "可引□" 的 USB □□棒， 在□机前将其□到□算机上。

   如果□是从光□安装， 那□□机后□立即将 FreeBSD 光□放入光□中。

   □于 FreeBSD 7.3 和更早的版本， 可以使用□□引□， □些□□可以根据 准□引□介□ 来制作。 其中， boot.flp 是□□□。 引□系□□□使用□□□□。

   如果□□机后□□□算机引□了先前已□装好的其他操作系□， □□□： .. 是不是□□或光□太□放入而□失□机引□□□。 如果是， □将它□放入后重新□机。 .. BIOS □定不□，□重新□□ BIOS 的□定。 .. □的 BIOS 不支持从□些安装介□引□。

5. FreeBSD 即将□□。如果□是从光□引□， □会□到□似下面的画面：

```
Booting from CD-Rom...
645MB medium detected
CD Loader 1.2

Building the boot loader arguments
Looking up /BOOT/LOADER... Found
Relocating the loader and the BTX
Starting the BTX loader

BTX loader 1.00 BTX version is 1.02
Consoles: internal video/keyboard
BIOS CD is cd0
BIOS drive C: is disk0
BIOS drive D: is disk1
BIOS 636kB/261056kB available memory

FreeBSD/i386 bootstrap loader, Revision 1.1

Loading /boot/defaults/loader.conf
/boot/kernel/kernel text=0x64daa0 data=0xa4e80+0xa9e40 syms
=[0x4+0x6cac0+0x4+0x88e9d]
\
```

如果□从□□□□，□□看到□似下面的画面：

```
Booting from Floppy...
Uncompressing ... done

BTX loader 1.00  BTX version is 1.01
Console: internal video/keyboard
BIOS drive A: is disk0
BIOS drive C: is disk1
BIOS 639kB/261120kB available memory

FreeBSD/i386 bootstrap loader, Revision 1.1

Loading /boot/defaults/loader.conf
/kernel text=0x277391 data=0x3268c+0x332a8 |

Insert disk labelled "Kernel floppy 1" and press any key...
```

□根据提示将 boot.flp □□取出，□入 kern1.flp □□□，然后按 `Enter`。□只需从第一□□□□，然后再需要□根据提示□入其他□□就可以了。

6. 不□是从光□、USB □□棒或□□引□，接下来都会□入 FreeBSD 引□加□器菜□：

□ *1. FreeBSD Boot Loader Menu*

□可以等待十秒，或按 `Enter`。

2.4.1.2. 引□ sparc64

多数 sparc64 系□均配置□从硬□自□引□。 如果希望安装 FreeBSD，就需要从网□或 CDROM □□了，□需要首先□入 PROM (OpenFirmware)。

要完成□□工作，首先需要重□系□，并等待出□引□消息。 具体的信息取决于□使用的型号，不□它□会是□似下面□□：

```
Sun Blade 100 (UltraSPARC-IIe), Keyboard Present
Copyright 1998-2001 Sun Microsystems, Inc.  All rights reserved.
OpenBoot 4.2, 128 MB memory installed, Serial #51090132.
Ethernet address 0:3:ba:b:92:d4, Host ID: 830b92d4.
```

如果□的系□此□□始了从硬□引□的□程，□需要按下 `L1` + `A` 或 `Stop` + `A`， 或者在串口控制台上□送 `BREAK`(例如，在 tip(1) 或 cu(1) 中是 `~#`) 以便□入 PROM 提示符。 它□□是□似下面□□：

```
ok         ①
ok {0} ②
```

① □是在只有一□ CPU 的系□上的提示。

② □是用于 SMP 系□的□□， □里的数字， 是系□中可用的 CPU 数量。

□□， 将 CDROM □入□□器， 并在 PROM 提示符后面， □入 `boot cdrom`。

2.4.2. □看□□探□的□果

前面屏幕□示的最后几百行字会存在□冲区中以便□□□。

要□□□冲区，□可以按下 `Scroll Lock` □， □会□□画面的卷□功能。然后□就可以使用方向□或 `PageUp`、`PageDown` □来上下翻□。 再按一次 `Scroll Lock` □将停止画面卷□。

在□□□的□候会看到□似 典型的□□探□□果的画面。 真正的□果依照□的□□装置而有所不同。

典型的□□探□□果

```
avail memory = 253050880 (247120K bytes)
Preloaded elf kernel "kernel" at 0xc0817000.
Preloaded mfs_root "/mfsroot" at 0xc0817084.
md0: Preloaded image </mfsroot> 4423680 bytes at 0xc03ddcd4

md1: Malloc disk
Using $PIR table, 4 entries at 0xc00fde60
npx0: <math processor> on motherboard
npx0: INT 16 interface
pcib0: <Host to PCI bridge> on motherboard
pci0: <PCI bus> on pcib0
pcib1:<VIA 82C598MVP (Apollo MVP3) PCI-PCI (AGP) bridge> at device 1.0 on pci0
pci1: <PCI bus> on pcib1
pci1: <Matrox MGA G200 AGP graphics accelerator> at 0.0 irq 11
```

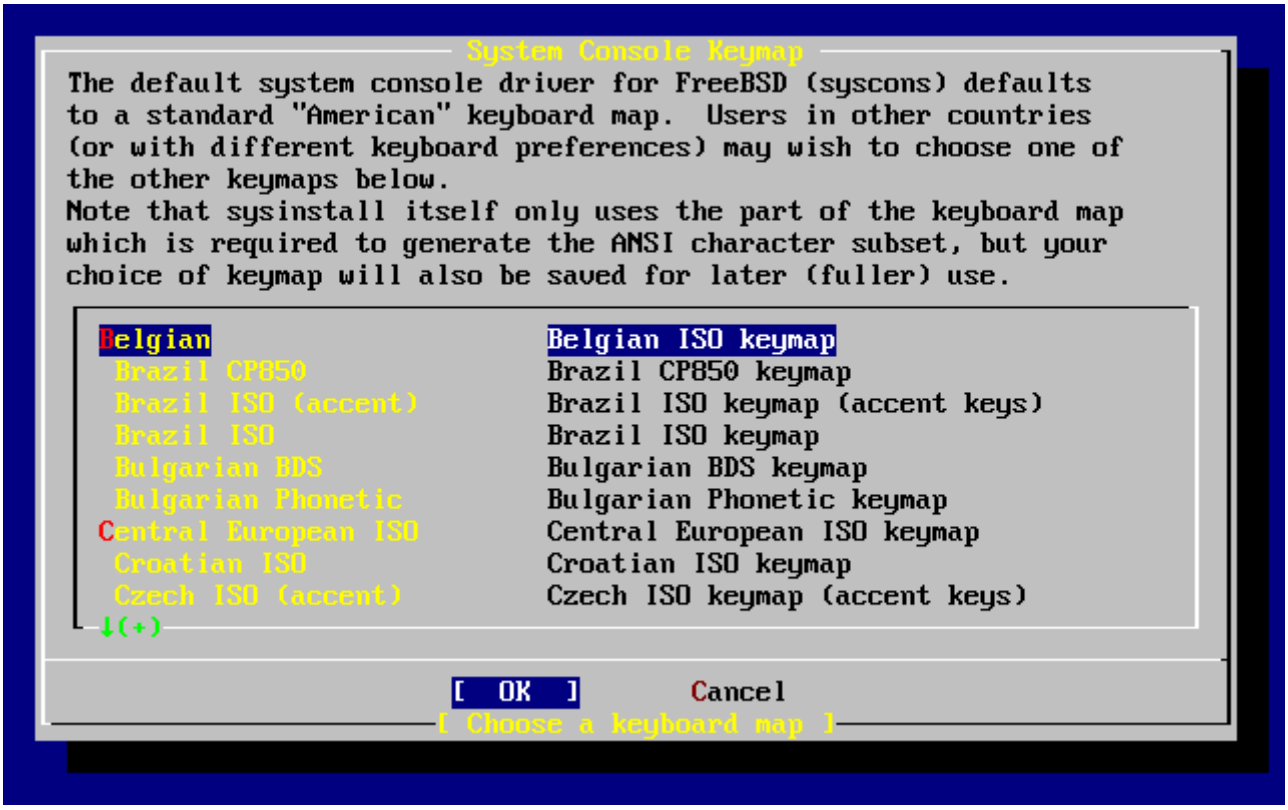
```
isab0: <VIA 82C586 PCI-ISA bridge> at device 7.0 on pci0
isa0: <iSA bus> on isab0
atapci0: <VIA 82C586 ATA33 controller> port 0xe000-0xe00f at device 7.1 on pci0
ata0: at 0x1f0 irq 14 on atapci0
ata1: at 0x170 irq 15 on atapci0
uhci0 <VIA 83C572 USB controller> port 0xe400-0xe41f irq 10 at device 7.2 on pci
0
usb0: <VIA 83572 USB controller> on uhci0
usb0: USB revision 1.0
uhub0: VIA UHCI root hub, class 9/0, rev 1.00/1.00, addr1
uhub0: 2 ports with 2 removable, self powered
pci0: <unknown card> (vendor=0x1106, dev=0x3040) at 7.3
dc0: <ADMtek AN985 10/100BaseTX> port 0xe800-0xe8ff mem 0xdb000000-0xeb0003ff ir
q 11 at device 8.0 on pci0
dc0: Ethernet address: 00:04:5a:74:6b:b5
miibus0: <MII bus> on dc0
ukphy0: <Generic IEEE 802.3u media interface> on miibus0
ukphy0: 10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
ed0: <NE2000 PCI Ethernet (RealTek 8029)> port 0xec00-0xec1f irq 9 at device 10.
0 on pci0
ed0 address 52:54:05:de:73:1b, type NE2000 (16 bit)
isa0: too many dependant configs (8)
isa0: unexpected small tag 14
orm0: <Option ROM> at iomem 0xc0000-0xc7fff on isa0
fdc0: <NEC 72065B or clone> at port 0x3f0-0x3f5,0x3f7 irq 6 drq2 on isa0
fdc0: FIFO enabled, 8 bytes threshold
fd0: <1440-KB 3.5" drive> on fdc0 drive 0
atkbdc0: <Keyboard controller (i8042)> at port 0x60,0x64 on isa0
atkbd0: <AT Keyboard> flags 0x1 irq1 on atkbdc0
kbd0 at atkbd0
psm0: <PS/2 Mouse> irq 12 on atkbdc0
psm0: model Generic PS/@ mouse, device ID 0
vga0: <Generic ISA VGA> at port 0x3c0-0x3df iomem 0xa0000-0xbffff on isa0
sc0: <System console> at flags 0x100 on isa0
sc0: VGA <16 virtual consoles, flags=0x300>
sio0 at port 0x3f8-0x3ff irq 4 flags 0x10 on isa0
sio0: type 16550A
sio1 at port 0x2f8-0x2ff irq 3 on isa0
sio1: type 16550A
ppc0: <Parallel port> at port 0x378-0x37f irq 7 on isa0
pppc0: SMC-like chipset (ECP/EPP/PS2/NIBBLE) in COMPATIBLE mode
ppc0: FIFO with 16/16/15 bytes threshold
plip0: <PLIP network interface> on ppbus0
ad0: 8063MB <IBM-DHEA-38451> [16383/16/63] at ata0-master UDMA33
acd0: CD-RW <LITE-ON LTR-1210B> at ata1-slave PIO4
Mounting root from ufs:/dev/md0c
/stand/sysinstall running as init on vty0
```

□仔□□□探□□果以□定 FreeBSD □到所有□期望出□的□□。 如果系□没有□到□□， □不会将其列出。 定制内核能□□□□系□添加默□的 GENERIC 内核所不支持的□□， 如声□等。

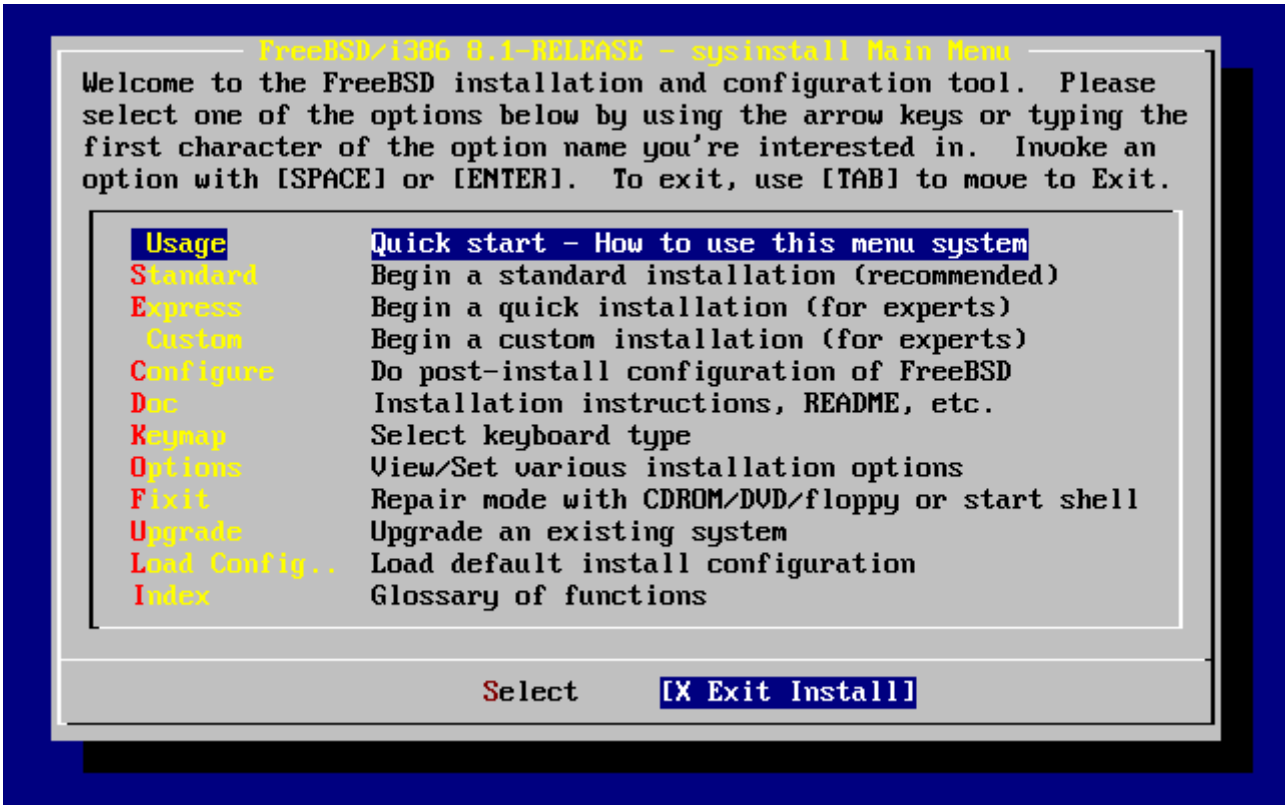
在 FreeBSD 6.2 和更高版本中， 在探□完系□□□之后， 将□示 □□国家及地区菜□。 □使用光□□来□□国家或地区。 接着按 Enter， 系□将自□□置地区。 □也可以很容易地退出 sysinstall 程序并从□来□。

□ *2.* □□国家及地区菜□

如果□在国家及地区菜□中□□了 United States (美国)， □系□会使用□准的美国□□映射； 如果□□了不同的国家，□会□示下面的菜□。 使用光□□□□正□的□□映射， 然后按 Enter 来□□。



□ *3.* □□□□菜□



□ *4. □□□□ Sysinstall*

在主界面使用方向□□□ Exit Install □会看到 如下的信息：

```
                User Confirmation Requested
      Are you sure you wish to exit? The system will reboot

                          [ Yes ]    No
```

如果此□□□了 **[yes]** 但 CDROM □留在光□里， □会再次□入安装程序。

如果□是从□□□□， □在重□系□之前， 需要将 boot.flp □□取出。

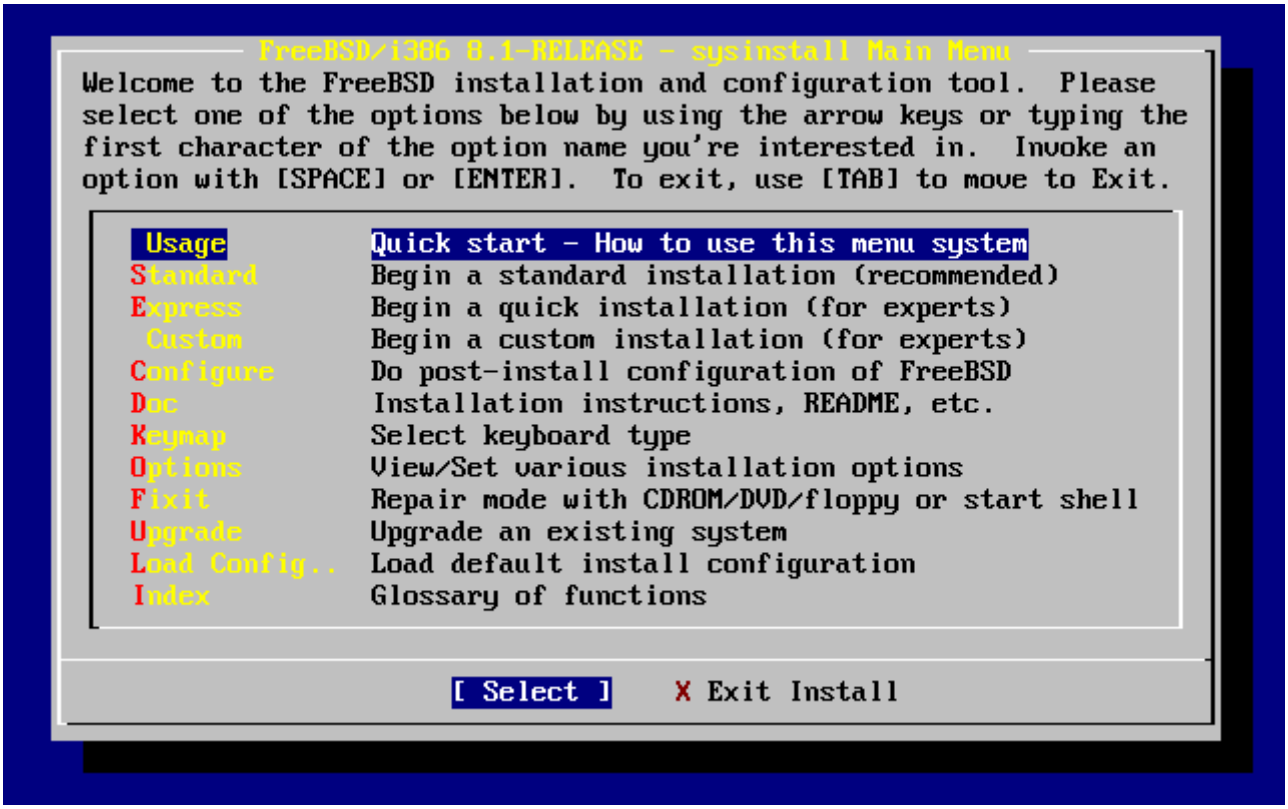
2.5. 介□ Sysinstall

sysinstall 是 FreeBSD □目所提供的安装程序。它以 console(控制台)□主， 分□多个菜□及画面□□配置及控制安装□程。

sysinstall 菜□画面由方向□、 `Enter`、`Tab`、`Space`， 以及其它按□所控制。在主画面的 Usage 菜□有□些按□的□明。

要□看□些□明，□将光□移到 Usage □目，然后 **[Select]** 按□被□□， □取 Sysinstall 主菜□的 Usage □目，然后按下 `Enter` □。

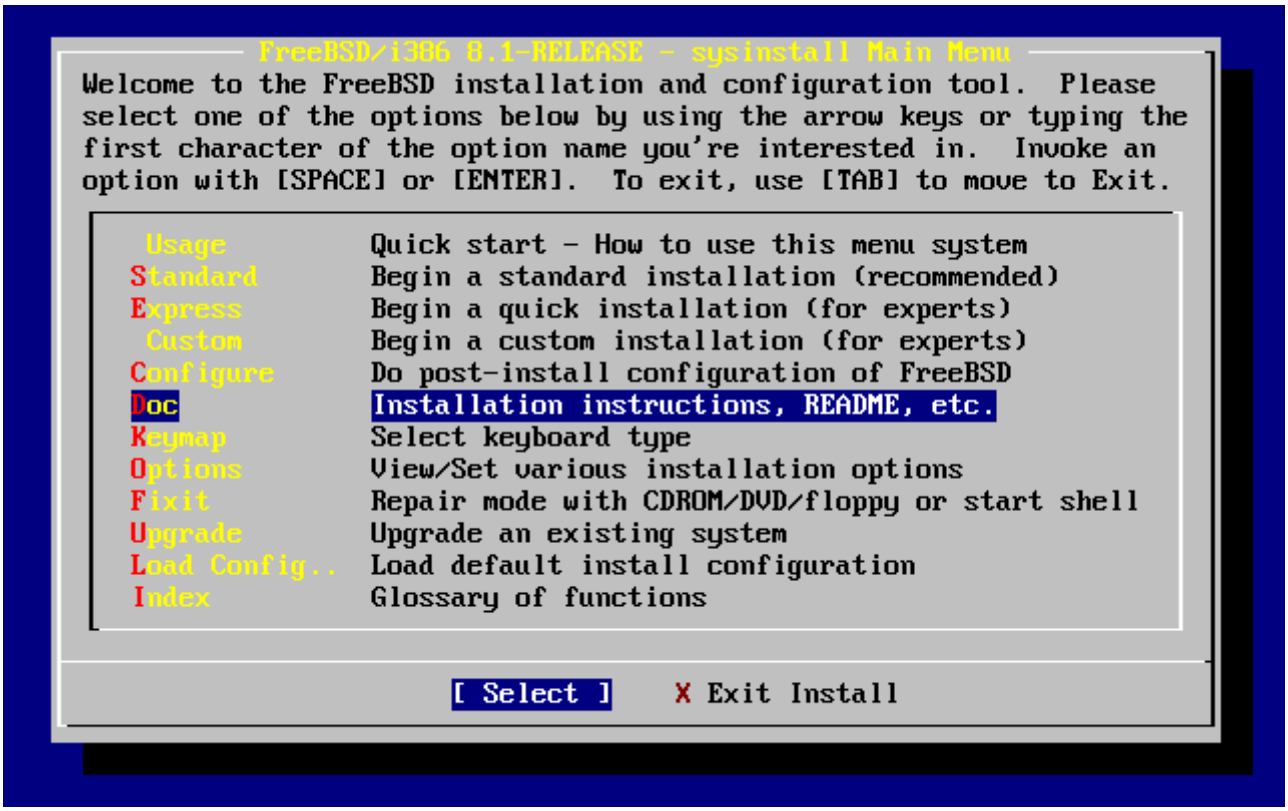
安装画面的使用□明会□示出来，□□完□□按 `Enter` □回到主画面。



□ *5.* □取 *Sysinstall* 主菜□的 *Usage* □目

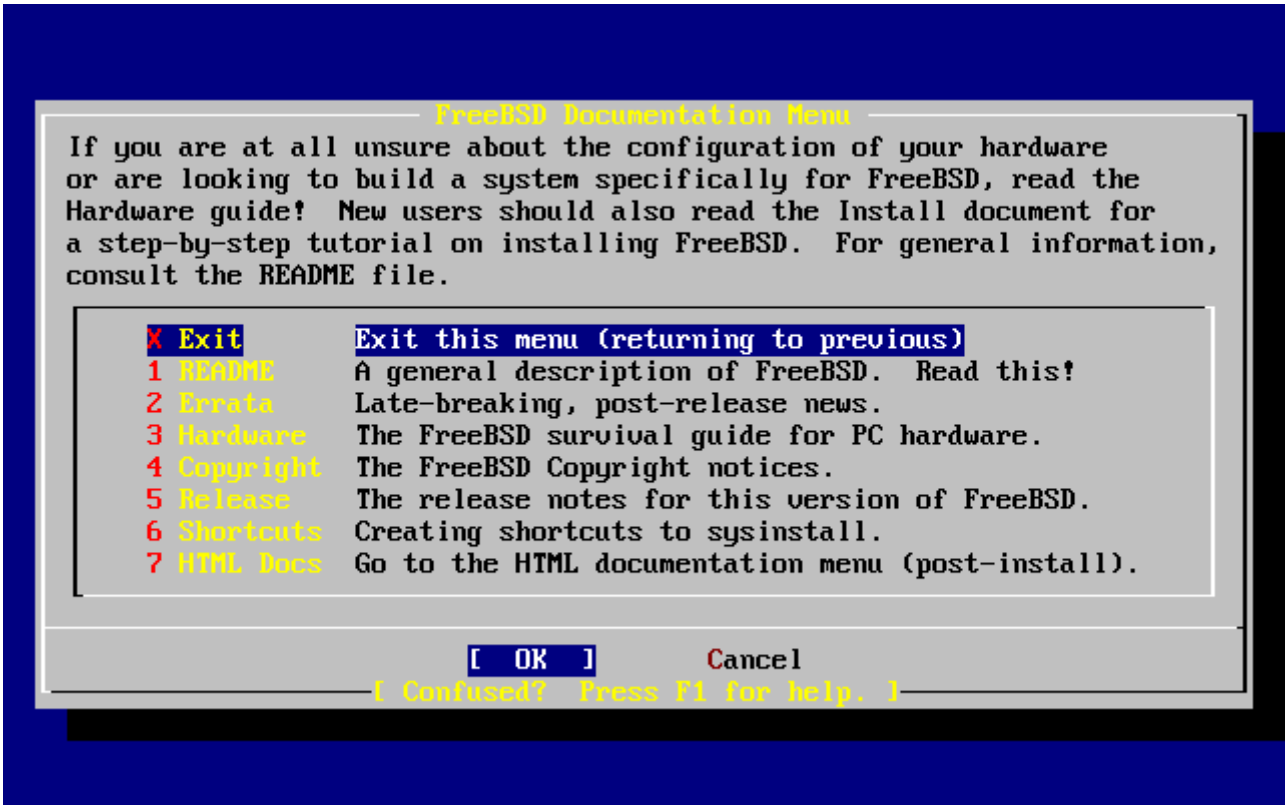
2.5.1. □□ Documentation(□明文件) 菜□

用方向□从主菜□□□ Doc 条目然后按 `Enter` □。



□ *6.* □□□明文件菜□

□将会□入□明文件菜□。



□ 7. *Sysinstall* □明文件菜□

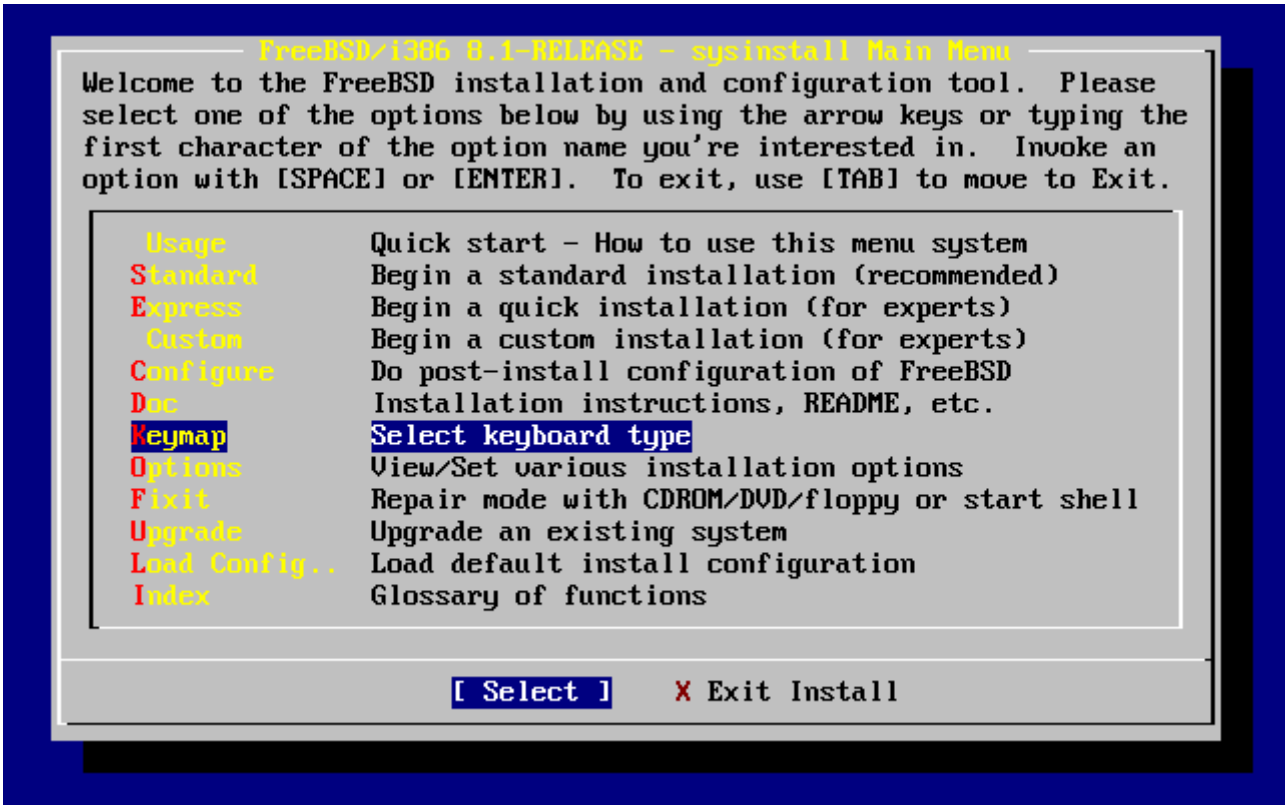
□□□些□明文件很重要。

要□□一篇文章，□用方向□□取要□□的文章然后按 `Enter` □。□□中再按一下 `Enter` 就会回到□明文件画面。

若要回到主菜□，用方向□□□ Exit 然后按下 `Enter` □。

2.5.2. □□□□□□(Keymap)菜□

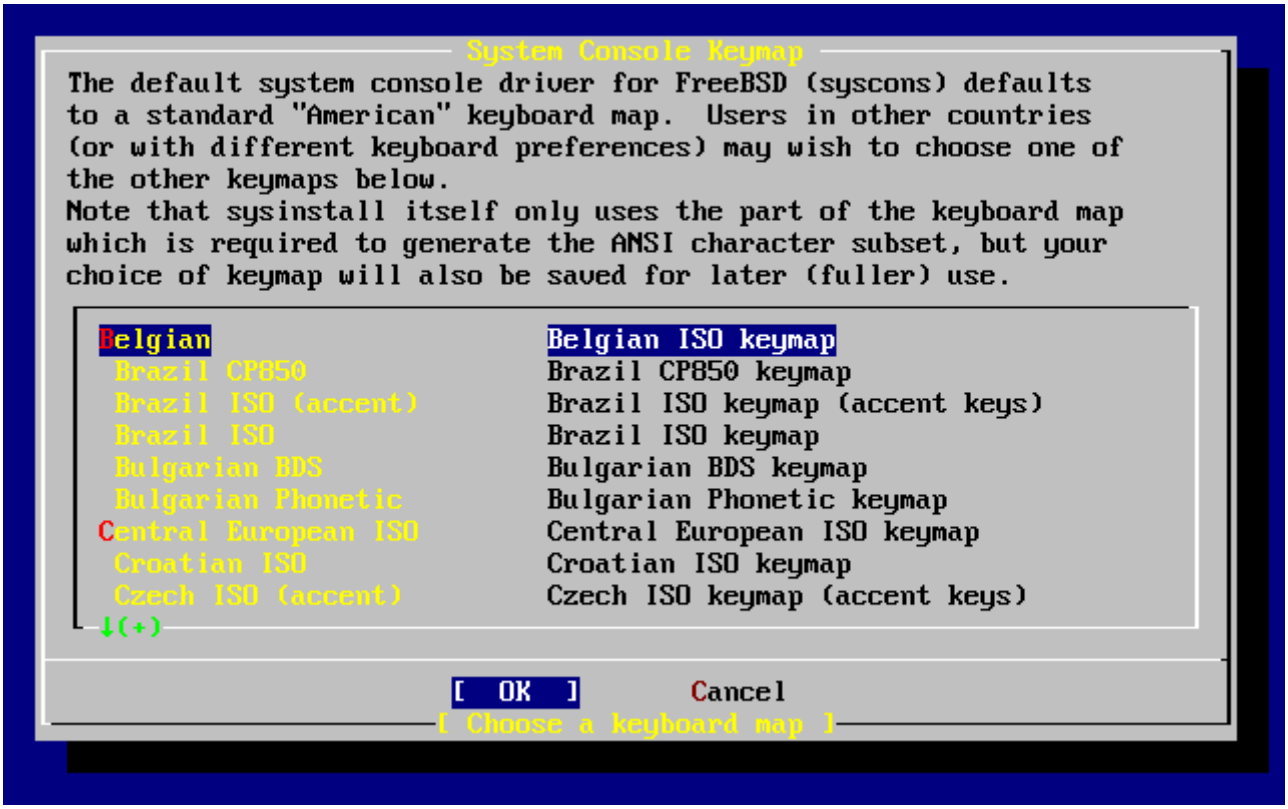
如果要改□□□按□的□□方式， □在主菜□□取 Keymap 然后按 `Enter` □。一般情况下不改□此□， 除非□使用了非□准□□或非美国□□。



□ *8. Sysinstall* 主菜□

□可以使用上下□移□到□想使用的□□□□方式， 然后按下 Space □以□取它；再按 Space □可以取消□取。当□完成后， □□□ **[OK]** 然后按 Enter □。

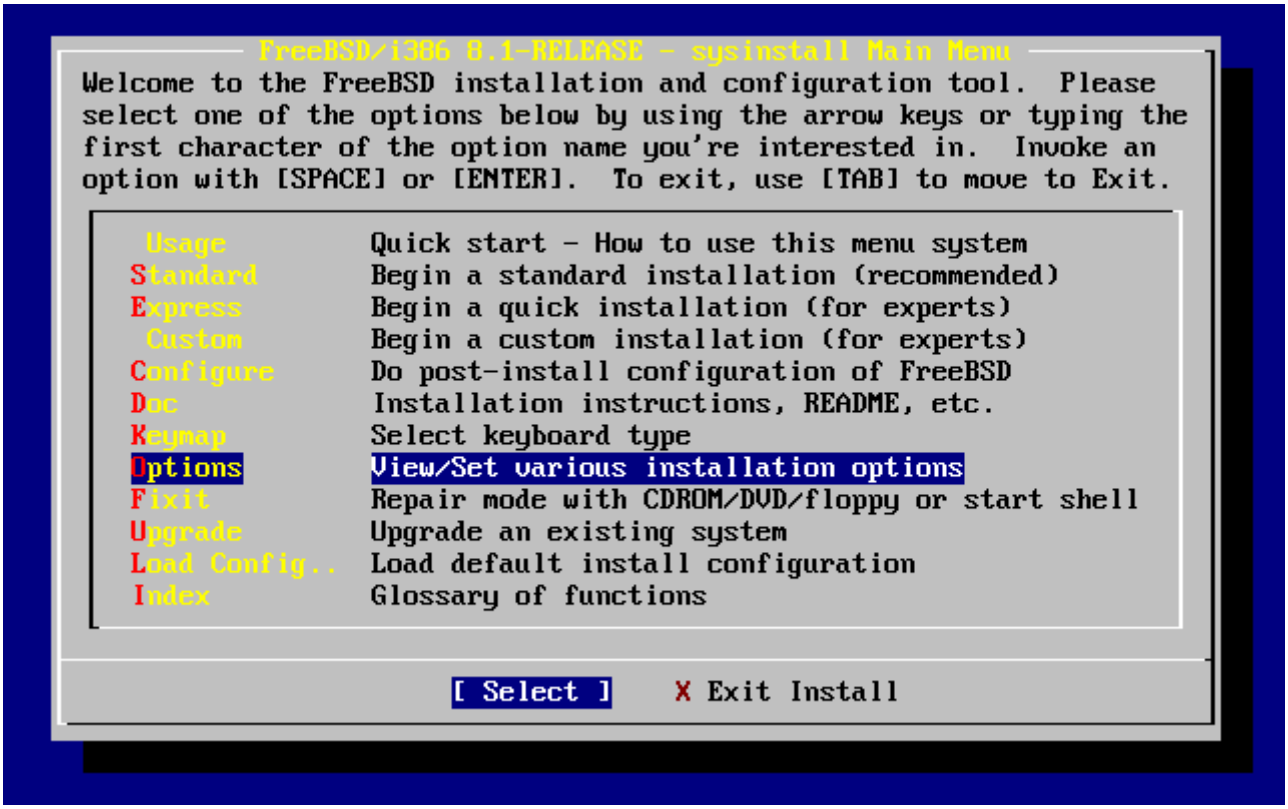
□一屏幕只□示出部分列表。□□ **[Cancel]** 按 Tab □将使用默□的□□□□， 并返回到主菜□



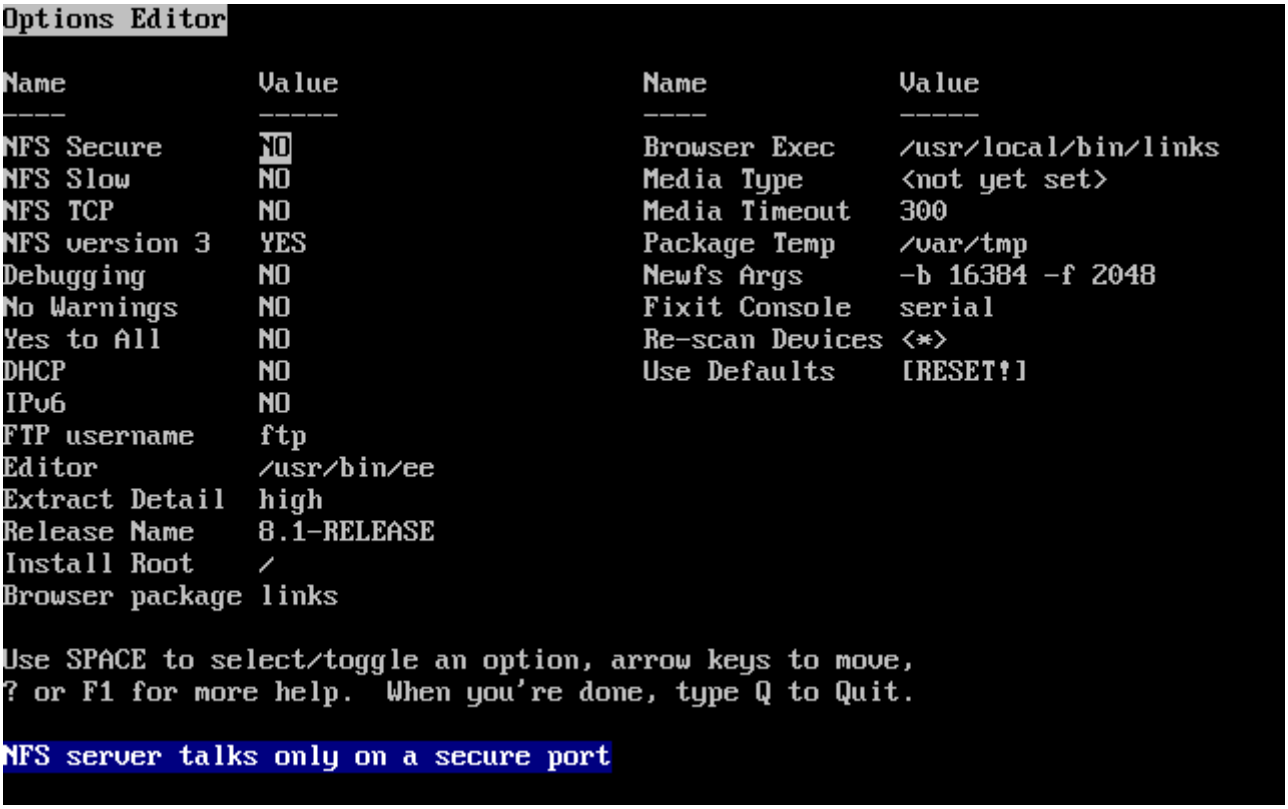
□ *9. Sysinstall* □□□□菜□

2.5.3. 安装□□□置画面

□□ Options 然后按 `Enter` □。



□ *10. Sysinstall* 主菜□



□ *11. Sysinstall* □□□置

□□□通常可以□用于大部分的使用者，□并不需要改□它□。 版本名称要根据安装的版本□行□化。

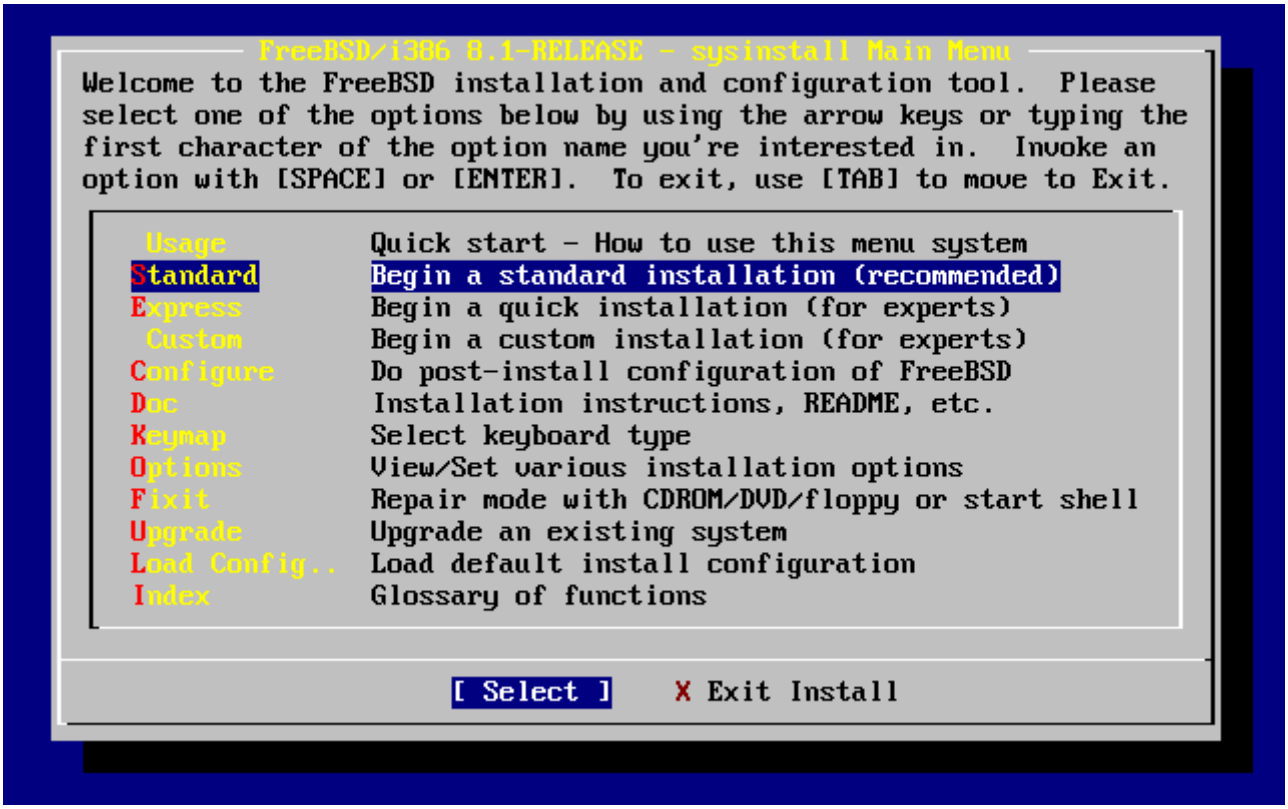
目前□□□目的描述会在屏幕下方以□底白字□示。 注意其中有一个□目是 Use Defaults(使用默□□) □可以由此□将所有的□定□原□□□。

可以按下 F1 来□□各□□的□明。

按 Q □可以回到主画面。

2.5.4. □始□行□准安装

Standard(□准) 安装□用于那些 UNIX® 或 FreeBSD 的初□使用者。用方向□□□ Standard 然后按 Enter □可□始□入□准安装。



□ *12.* □始□行□准安装

2.6. 分配磁□空□

□的第一个工作就是要分配 FreeBSD 用的硬□空□以便 sysinstall 先做好一些准□。 □了完成□个工作，□必□先□ FreeBSD 如何□到磁□信息做一个了解。

2.6.1. BIOS 磁□□号

当□在系□上安装配置 FreeBSD 之前， 有一个重要的事情一定要注意，尤其是当□有多个硬□的□候。

在 pc 架□，当□□像 MS-DOS® 或 Microsoft® Windows® □□跟 BIOS 相□的操作系□的□候，BIOS 有能力改□正常的磁□□序， 然后□些操作系□会跟着 BIOS 做改□。□□使用者不一定非要有所□的 "primary master" 硬□□机。 □多人□□最□□而便宜□□系□的方式就是再去□一□一模一□的硬□， 然后定期将数据从第一□硬□□制到第二个硬□，使用 Ghost 或 XCOPY。所以，当第一个硬□死了， 或者是被病毒破坏，或者有坏□道，他□可以□整 BIOS 中的□机□序而直接用第二□硬□□机。 就像交□硬□的数据□，但是无需打□机箱。

比□昂□，配有 SCSI 控制□的系□通常可以延伸 BIOS 的功能来□ SCSI □□ (可□七个) □到□似改□□序的功能。

□□于使用□□方式的使用者可能会感到□□， 因□在 FreeBSD 中并非如此。FreeBSD 不会参考 BIOS，而且也不知道所□的 "BIOS □□磁□□□" 是□□回事。□会□人感□很疑惑， 明明就是一□的硬□而且□料也完全从□一□□制□来的， □果却没□法像以前那□用。

当使用 FreeBSD 以前，□将 BIOS 中的硬□□机□序□回正常的□序， 并且以后不要再改□。 如果一定要交□硬□□序， 那□用硬件的方式， 打□机箱并□整□□。

> Bill 替 Fred 把旧的 Wintel 的机器装上了 FreeBSD。 他装了一台 SCSI 硬□，ID 是 0，然后把 FreeBSD 装在上面。
>
> Fred □始使用他新的 FreeBSD 系□；但是□了几天， 他□□□旧的 SCSI硬□□生了□多小□□。之后， 他就跟 Bill □起□件事。
>
> 又□了几天，Bill 决定是□解决□□的□候了， 所以他从后面房□的硬□ "收藏" 中□出了一个一模一□的硬□，并且□□表面□□后□示□□硬□没有□□。 因此，Bill 将它的 ID □成 4，然后安装到 Fred 的机器，并且将□料从磁□ 0 □制到磁□ 4。□在新硬□装好了， 而且看起来好像一切正常；所以，Bill □□□在□□可以□始用它了。 Bill 于是到 SCSI BIOS 中□定 SCSI ID 4 □□机□，用磁□ 4 重新□机后，一切□得很□利。
>
> □□用了几天后，Bill 跟 Fred 决定要来玩点新的： □将 FreeBSD 升□了。Bill 将 ID 0 的硬□移除 (因□有□□) 并且又从收藏区中拿了一□一□的硬□来。然后他用 Fred 神奇的网□ FTP 磁□将新版的 FreeBSD 安装在□□硬□上； 安装□程没什□□□□生。
>
> Fred 用了□新版本几天后，□得它很□合用在工程部□... 是□候将以前放在旧系□的工作□料□制□来了。因此， Fred 将 ID4 的 SCSI 硬□ (里面有放着旧系□中□制□来的最新□料) mount 起来，□果竟然□□在 ID4 的硬□上， 他以前的所有□料都不□了！
>
> □料□到□里去了□？
>
> 当初 Bill 将 ID0 硬□的□料□制到 ID4 的□候， ID4 即成□一个 "新的副本"。 而当他□ SCSI BIOS □定 ID4 □□机□，想□系□从 ID4 □机， □其□只是他自己□，因□大部分的系□可以直接□ BIOS 而改□□机□序， 但是 FreeBSD 却会把□机□序□原成正常的模式，因此，Fred 的 FreeBSD □是从原来那□ ID0 的硬□□机的。所有的□料都□在那□硬□上， 而不是在想象之中的 ID4 硬□。
>
> 幸□的是， 在我□□□□件事的□候那些□料都□在， 我□将□些□料从最早的那□ ID0 硬□取出来并交□□ Fred， 而 Bill 也由此了解到□算机□数是从 0 □始的。
>
> □然我□□里的例子使用 SCSI 硬□， 但是相同的概念也可以套用在 IDE 硬□上。

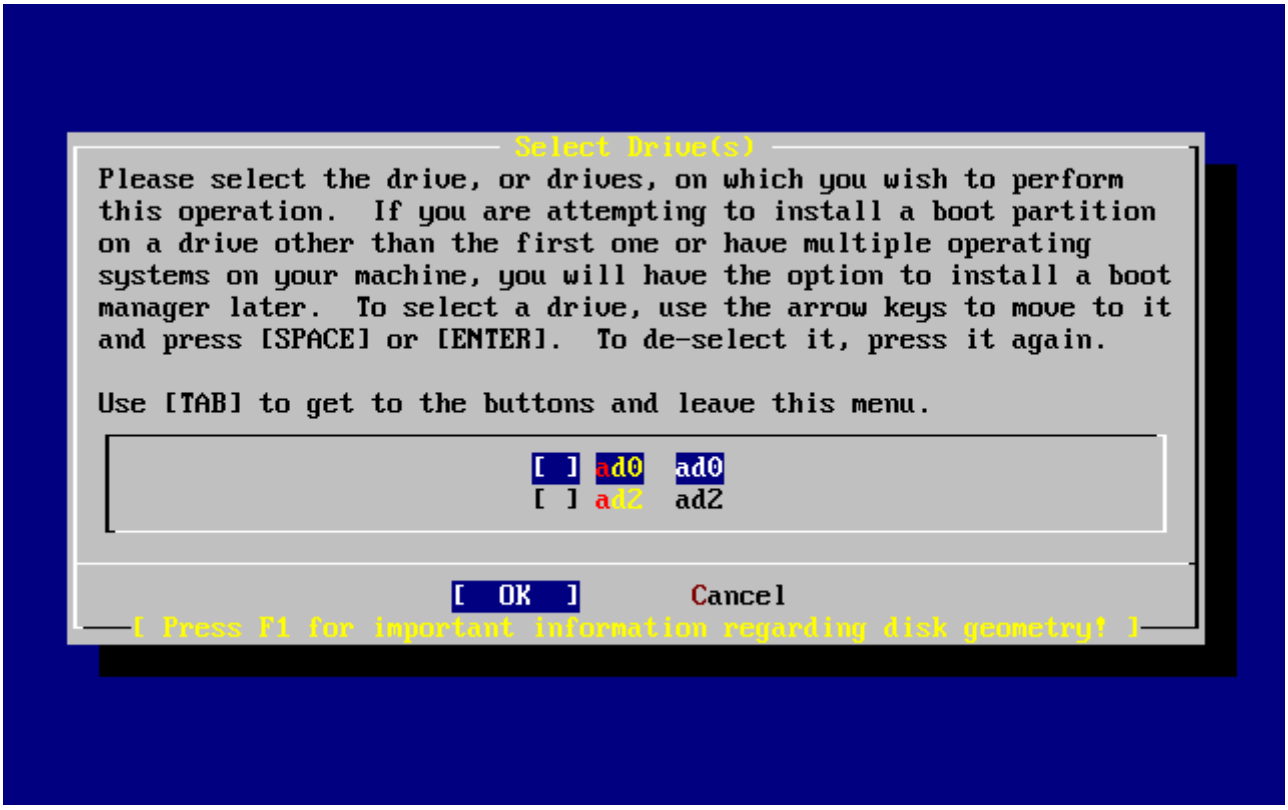
2.6.2. 使用 FDisk □建分区

如果不再做改□，数据将会写□硬□。如果□犯了一个□□想重新□始， □□□ sysinstall 安装程序的退出按□(exit)。或按 U □来 Undo 操作。 如果□的操作没有□果， □□可以重新□□□的□算机来□到□的目的。

当□在 sysinstall 主菜□□□使用□准安装后，□会看到下面的信息：

```
                                Message
In the next menu, you will need to set up a DOS-style ("fdisk")
partitioning scheme for your hard disk. If you simply wish to devote
all disk space to FreeBSD (overwriting anything else that might be on
the disk(s) selected) then use the (A)ll command to select the default
partitioning scheme followed by a (Q)uit. If you wish to allocate only
free space to FreeBSD, move to a partition marked "unused" and use the
(C)reate command.
                              [  OK  ]

                      [ Press enter or space ]
```

如屏幕指示，按 `Enter` ，然后就会看到一个列表列出所有在探测到的候选到的硬盘。 要分区的硬盘 例示的是有到个 IDE 硬盘的情形，个硬盘分别是 ad0 和 ad2。



13. 要分区的硬盘

可能正在奇怪，什么 ad1 没有列出来？ 什么消失了？

想，如果有个 IDE 硬盘，一个是在第一个 Primary master， 一个是 Secondary master，会生什么事？ 如果 FreeBSD 依照到的序来他命名，如 ad0 和 ad1 那就不会有什么。

但是，在来了。如果在想在 primary slave 加装第三个硬盘， 那个硬盘的名称就会是 ad1，之前的 ad1 就会成 ad2。 会造成什么？因的名称 （如 ad1s1a）是用来文件系统的， 因此可能会，突然，有些文件系统从此无法正地示出来， 必修改 FreeBSD 配置文件（注：/etc/fstab ）才可以正示。

了解决些，在配置内核的候可以叫 FreeBSD 直接用 IDE 所在的位置来命名，而不是依据到的序。使用方式的， 在 secondary master 的 IDE 就 永是ad2，即使的系中没有 ad0 或 ad1

也不受影□。

此□ FreeBSD 内核的默□□，□也是□什□上面的画面只□示 ad0 和 ad2 的原因。 画面上□台机器的□□硬□是装在 primary 及 secondary 的 master 上面； 并没有任何一个硬□安装在 slave □槽上。

□□□□□□想安装 FreeBSD 的硬□，然后按下 **[OK]**。之后 FDisk 就会□始，□会看到□似 典型的尚未□□前的 Fdisk 分区表的画面。

FDisk 的□示画面分□三个部分。

第一部分是画面上最上面□行，□示的是目前所□□的硬□的信息。 包含它的 FreeBSD 名称、硬□分布以及硬□的□容量。

第二部分□示的是目前□□的硬□上有□些分区， □个分区的□始及□束位置、所占容量、FreeBSD 名称、 它□的描述以及□□（sub-type）。此□例□示有□个未使用的小分区， □有一个大的 FAT 分区， （很可能是 MS-DOS® 或 Windows® 的 C: ）， 以及一个□展分区（在 MS-DOS® 或 Windows® 里面□可以包含□□分区）。

第三个部分□示 FDisk 中可用的命令。

```
Disk name:      ad0                                  FDISK Partition Editor
DISK Geometry:  16383 cyls/16 heads/63 sectors = 16514064 sectors (8063MB)

Offset       Size(ST)        End      Name  PType       Desc  Subtype    Flags

     0             63         62         -      6     unused        0
    63        4193217    4193279    ad0s1      2        fat       14    >
4193280           1008    4194287         -      6     unused        0    >
4194288       12319776   16514063    ad0s2      4   extended       15    >




The following commands are supported (in upper or lower case):

A = Use Entire Disk    G = set Drive Geometry   C = Create Slice   F = `DD' mode
D = Delete Slice       Z = Toggle Size Units    S = Set Bootable   I = Wizard m.
T = Change Type        U = Undo All Changes     Q = Finish


Use F1 or ? to get more help, arrow keys to select.
```

□ *14.* 典型的尚未□□前的 *Fdisk* 分区表

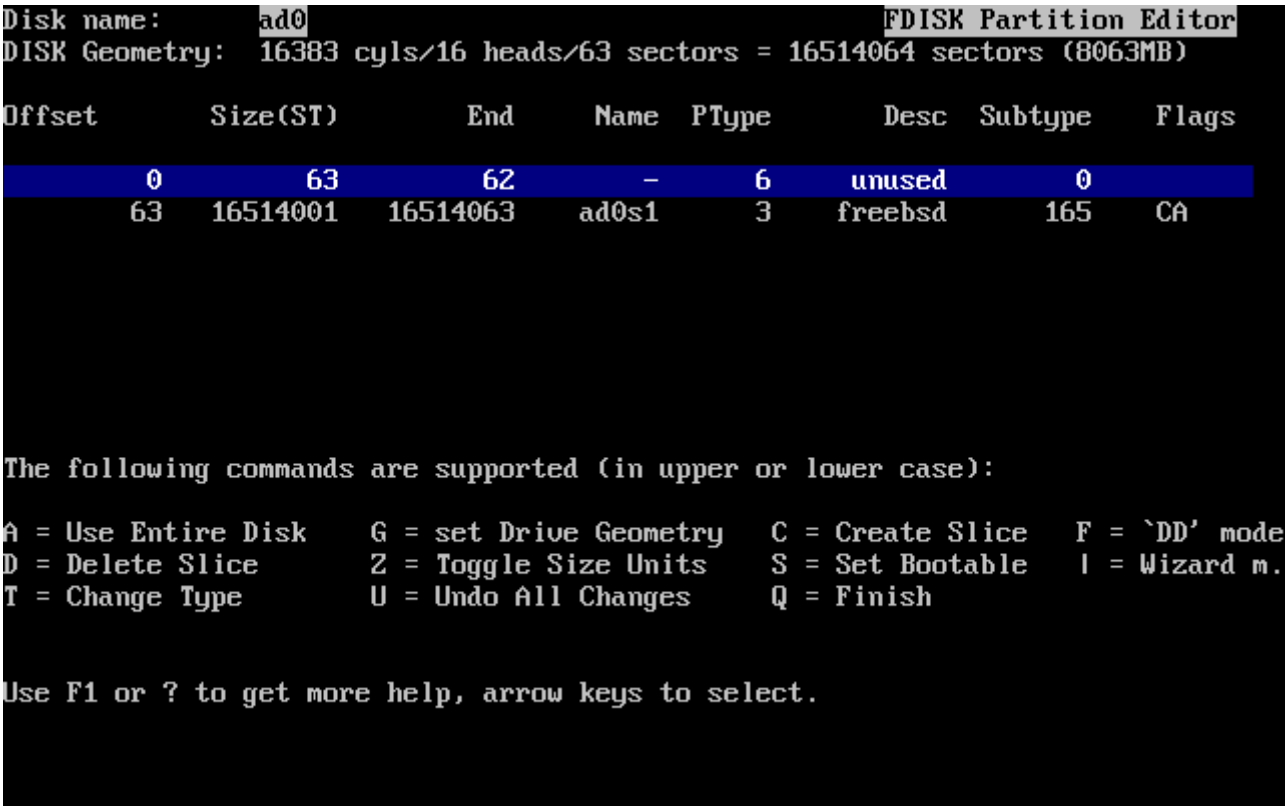
接下来要做的事跟□要□□□□的硬□分区有□。

如果□要□ FreeBSD 使用整个硬□（□后□□□要 sysinstall □□安装后会□除所有□个硬□上的□料），那□□就可以按 `A` □（Use Entire Disk ） 目前已有的分区都会被□除，取而代之的是一个小的，□示□ `unused` 的分区，以及一个大的 FreeBSD 分区。之后， □用方向□将光□移到□个 FreeBSD 分区，然后按 `S` 以将此分区□□□□□分区。 □会看到□似 Fdisk 分区使用整个硬□ 的画面。注意，在 `Flags` □中的 `A` □号表示此分区是 激活 的， 因而□□将从此分区□行。

要□除□有的分区以便□ FreeBSD □出空□， □可以将光□移□到要□除的分区后按 `D` □。 然后就可按 `C` □， 并在

□出的□□框中□入将要□建的分区的大小。 □入合□的大小后按 Enter □。 一般而言， □个□□框中的初始□是可以分配□□分区的最大□。 它可能是最大的□接分区或未分配的整个硬□大小。

如果□已□建立好□ FreeBSD 的分区 （使用像 PartitionMagic®□似的工具）， 那□□可以按下 C □来建立一个新的分区。同□的， 会有□□框□□□要建立的分区的大小。



□ *15. Fdisk* 分区使用整个硬□

完成后，按 Q □。□的□更会存在 sysinstall 中， 但是□不会真正写入□的硬□。

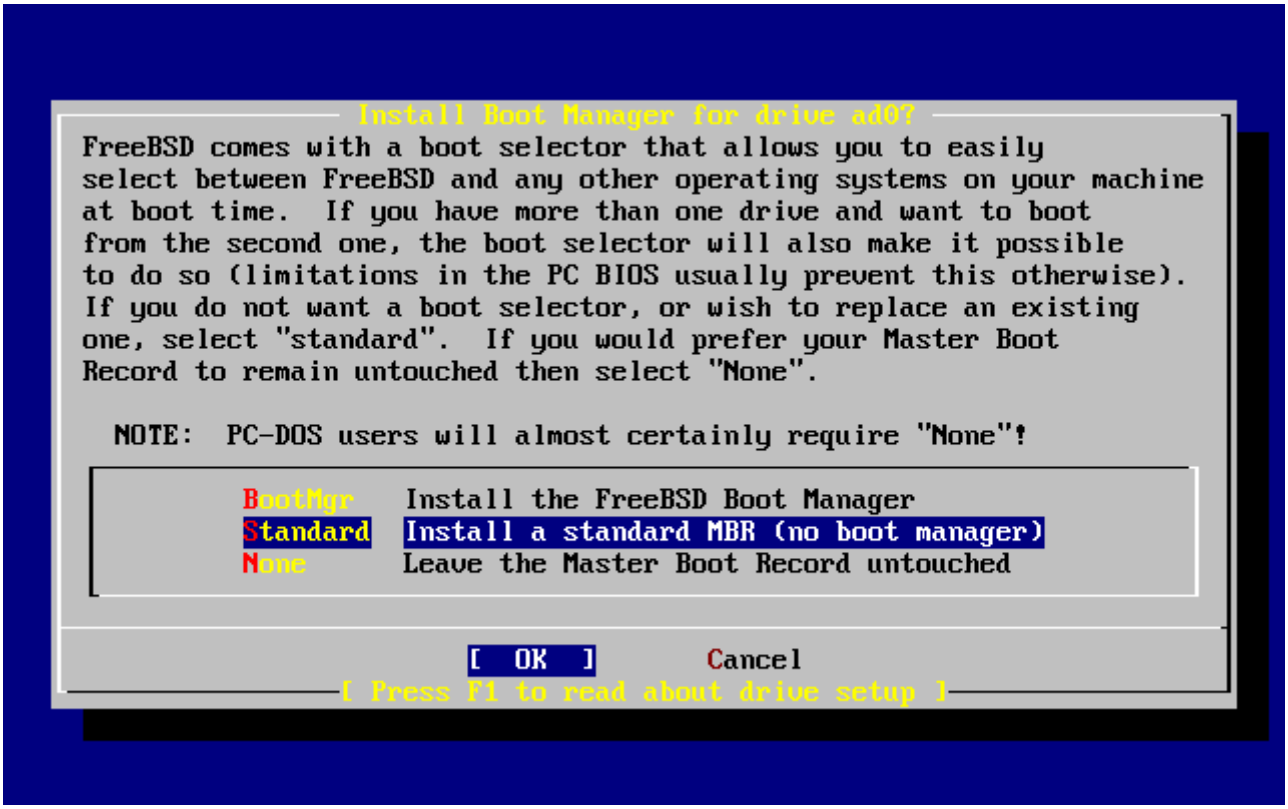
2.6.3. 安装多重引□

在□□□□可以□□要不要安装一个多重引□管理器。 一般而言，如果□到下列的情形， □□□□要安装多重引□管理程序。

- □有一个以上的硬□，并且 FreeBSD 并不是安装在第一个硬□上。
- 除了 FreeBSD，□□有其它的操作系□安装在同一□硬□上， 所以□需要在□机的□候□□要□入□一个系□。

如果□在□台机器上只安装一个 FreeBSD 操作系□， 并且安装在第一个硬□， 那□□□ Standard 安装就可以了。如果□已□使用了一个第三方的多重引□程序， 那□□□□ None。

□□好配置后□按 Enter。



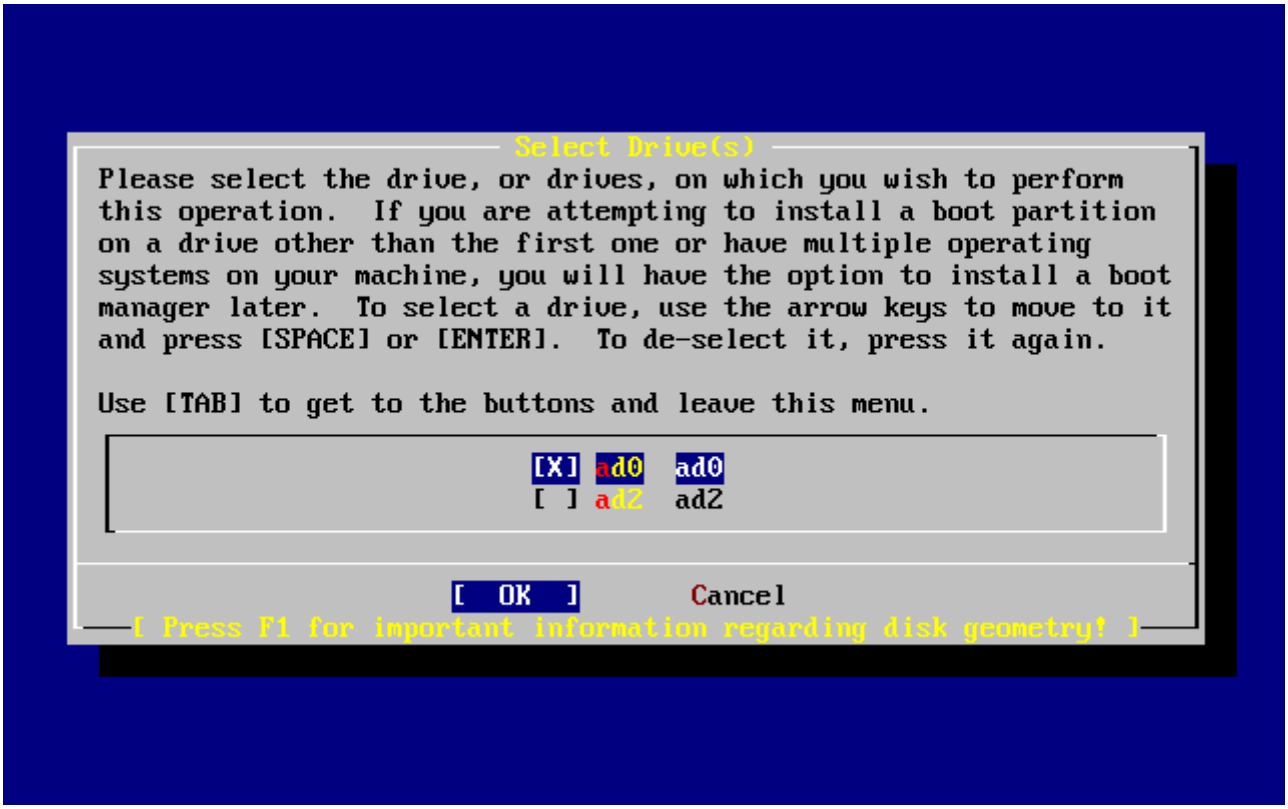
□ *16. Sysinstall* 多重引□管理程序

按下 F1 □所□示的在□□明中有□□一些操作系□共存可能□生的□□。

2.6.4. 在其它硬□上□建分区

如果□的系□上有一个以上的硬□， 在□□完多重引□管理程序后会再回到□□硬□的画面。 如果□要将 FreeBSD 安装在多个硬□上，那□□可以在□里□□其它的硬□， 然后重□使用 FDisk 来建立分区。

如果□想□ FreeBSD 来管理其它的硬□， 那□□个硬□都必□安装 FreeBSD 的多重引□管理程序。



□ *17.* □□□□硬□画面

Tab □可以在□最后□□的硬□、 **[OK]** 以及 **[Cancel]** 之□□行切□。

用 Tab □将光□移□到 **[OK]** 然后按 Enter □□□安装□程。

2.6.5. 使用 **bsdlabel** □建分区

□□在必□在□□建立好的 slice 中□□一些 label。 □注意，□个 label 的代号是 a 到 h，□外，□□上 b、 c 和 d 是有特殊用途的，不□□随意□□。

某些□用程序可以利用一些特殊的分区而□到□好的效果， 尤其是分区分散在不同的硬□的□候。但是，□在□是第一次安装FreeBSD， 所以不需要去□□如何分割□的硬□。最重要的是， 装好FreeBSD然后学□如何使用它。当□□FreeBSD有相当程度的熟悉后， □可以随□重新安装FreeBSD，然后改□□分区的方式。

下面的□例中有四个分区 - 一个是磁□交□分区，□外三个是文件系□。

表 *2.* □第一个硬□分区

分区	文件系□	大小	描述
a	/	1 GB	□是一个根文件系□（root filesystem）。 任何其它的文件系□都会挂在根目□（□注：用根目□比□□切） 下面。 1 GB □于此目□来□是合理的大小， 因□□往后并不会在□里存放太多的数据； 在安装 FreeBSD 后会用掉□ 128 MB 的根目□空□。 剩下的空□是用来存放□□文件用的，同□， □也□□□留一些空□，因□以后的FreeBSD版本可能会需要□多的 /（根目□）空□。
b	N/A	2-3 x RAM	b 分区□系□磁□交□分区（swap space）。□□正□的交□空□大小可是一□学□□。 一般来□，交□空□的大小□□是□系□上内存（RAM） 大小的2到3倍。 交□空□至少要有 64 MB。因此， 如果□的□□上的 RAM 比 32 MB 小， □将交□空□大小□□ 64 MB。 如果□有一个以上的硬□，□可以在□个硬□上都配置交□分区。FreeBSD 会利用□个硬□上的交□空□，□□做能□提高 swap 的性能。 如果是□□情形，先算出□□共需要的交□空□大小 （如128 MB），然后除以□□有的硬□数目（如2□），算出的□果就是□个硬□上要配置的交□空□的大小。在□个例子中， □个硬□的交□空□□ 64 MB。

分区	文件系□	大小	描述
e	/var	512 MB 至 4096 MB	/var 目□会存放不同□度的文件、日志以及其它管理用途的文件。大部分□些文件都是 FreeBSD □天在□行的□候会□取或是写入的。当□些文件放在□外的文件系□（□注：即/var）可以避免影□到其它目□下面□似的文件存取机制。
f	/usr	剩下的硬□空□ （至少 8 GB）	□所有的其它的文件通常都会存在/usr 目□以及其子目□下面。

上面例子中的数□□限于有□□的用□使用。 通常我□鼓励用□使用 FreeBSD 分区□□器中一个叫做 Auto Defaults的自□分区布局功能。

如果□要将FreeBSD安装在一个以上的硬□， 那□□必□在□配置的其它分区上再建立分区。 最□□的方式就是在□个硬□上建立□个分区，一个是交□分区， 一个是文件系□分区。

表 3. □其它磁□分区

分区	文件系□	大小	描述
b	N/A	□描述	之前提□，交□分区是可以跨硬□的。但是，即使 a 分区没有使用，□□上□是会把交□分区放在 b 分区上。
e	/disk_n_	剩下的硬□空□	剩下的空□是一个大的分区，最□□的做法是将之□□□ a分区而不是e分区。然而， □□上a分区是保留□根目□ (/) 用的。□不一定要遵守□个□□，但是 sysinstall 会，所以照着它做会使□的安装比□清爽、干□。□可以将□些文件系□挂在任何地方，本□例建□将它□挂在 /diskn 目□，*n* 依据□个硬□而有所不同，但是，□喜□的□也可将它□挂在□的地方。

分区的配置完成后，□可以用sysinstall. 来建立它□了。□会看到下面的信息：

```
                                Message
Now, you need to create BSD partitions inside of the fdisk
partition(s) just created. If you have a reasonable amount of disk
space (1GB or more) and don't have any special requirements, simply
use the (A)uto command to allocate space automatically. If you have
more specific needs or just don't care for the layout chosen by
(A)uto, press F1 for more information on manual layout.

                                [  OK  ]
                            [ Press enter or space ]
```

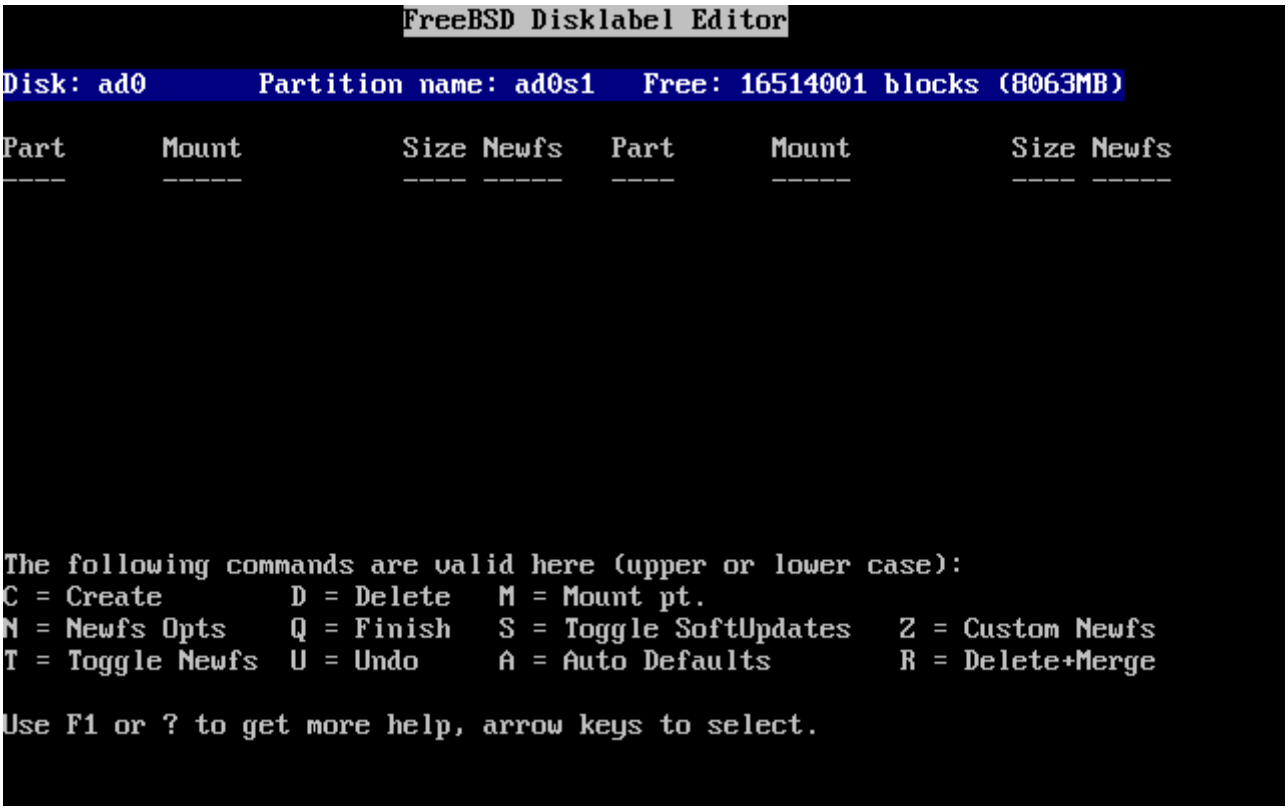
按下 `Enter` □□始FreeBSD分区表□□器，称做 Disklabel。

Sysinstall Disklabel □□器 □示□第一次□行 Disklabel的画面。 画面分□三个区域。

前几行□示的是□正在□□的硬□以及□正在建立的 slice 位于□个分区上。（在□里，Disklabel 使用的是分区名称 而不是 slice 名）。 此画面也会□示 slice □有多少空□可以使用；亦即，有多余的空□，但是尚未指派分区。

画面中□区域□示已建立的区区，□个分区的文件系□名称、 所占的大小以及一些□于建立□些文件系□的参数□□。

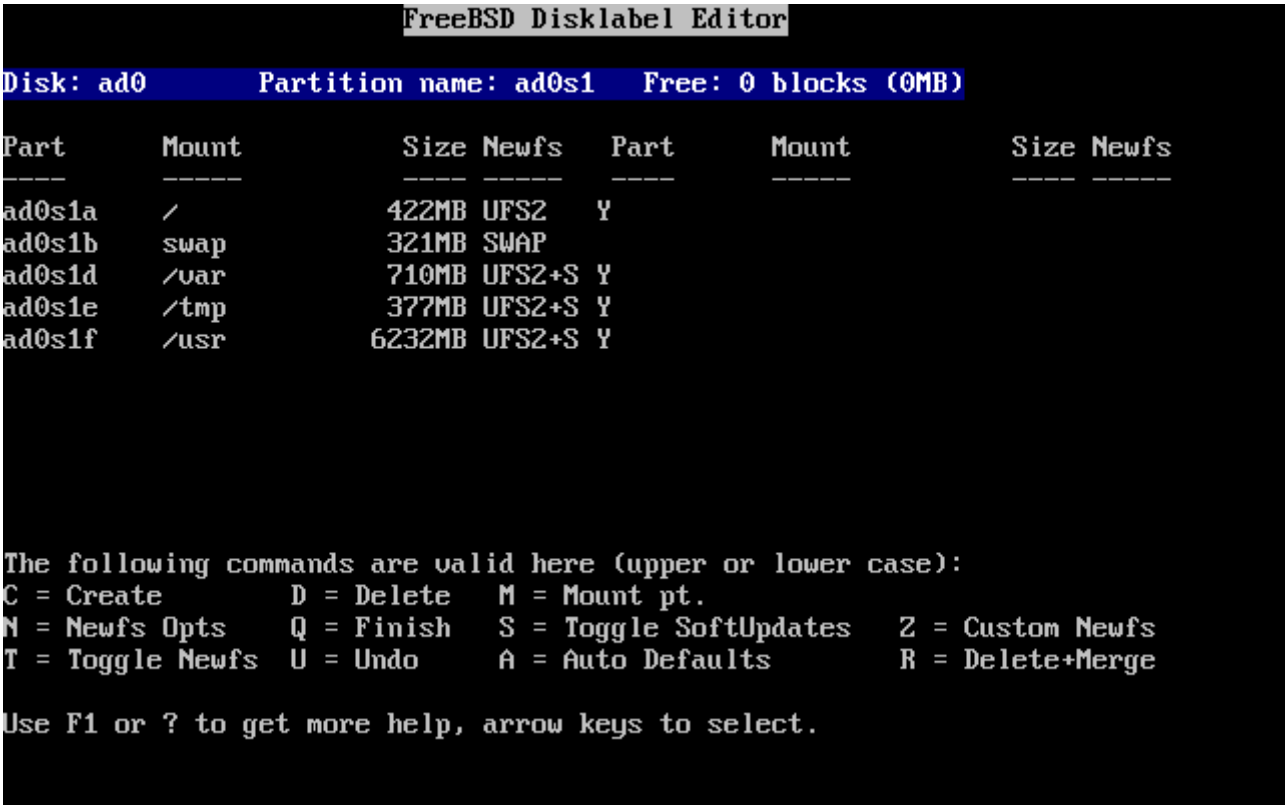
下方的第三区□示在 Disklabel 中可用的按□。



□ *18. Sysinstall Disklabel* □□器

Disklabel □可以自□配置分区以及□它□□□的大小。 □些默□的分区是由内部的分区尺寸算法根据磁□的大小□算出的。 □可以按 `A` □使用此功能。□会看到□似 Sysinstall Disklabel □□器-使用自□配置的画面。根据□硬□的大小， 自□分配所配置的大小不一定合□。但是没有□系， □并不一定要使用□□的大小。

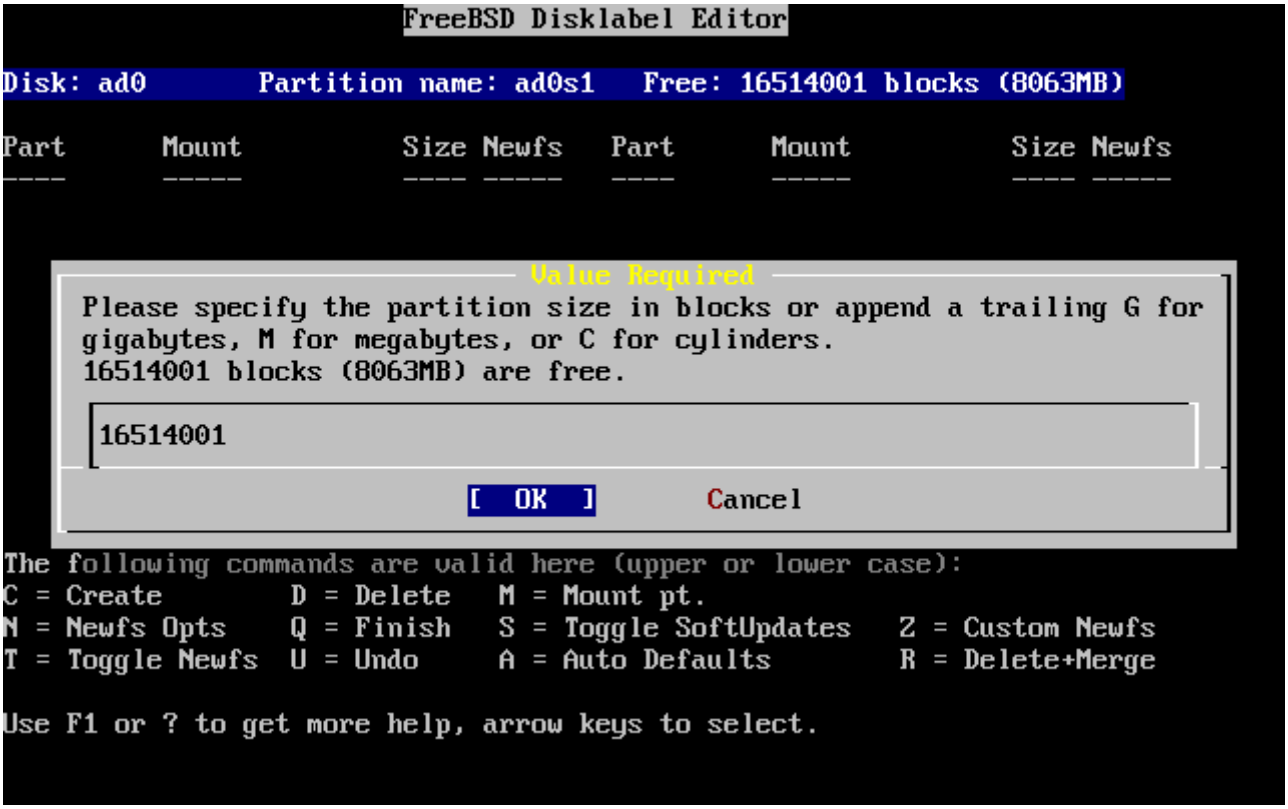
默□情况下会□/tmp 目□一个独立分区，而不是附属在 / 之下。 □□可以避免将一些□□文件放到根目□中（□注： 可能会用完根目□空□）。



□ *19. Sysinstall Disklabel* □□器-使用自□配置

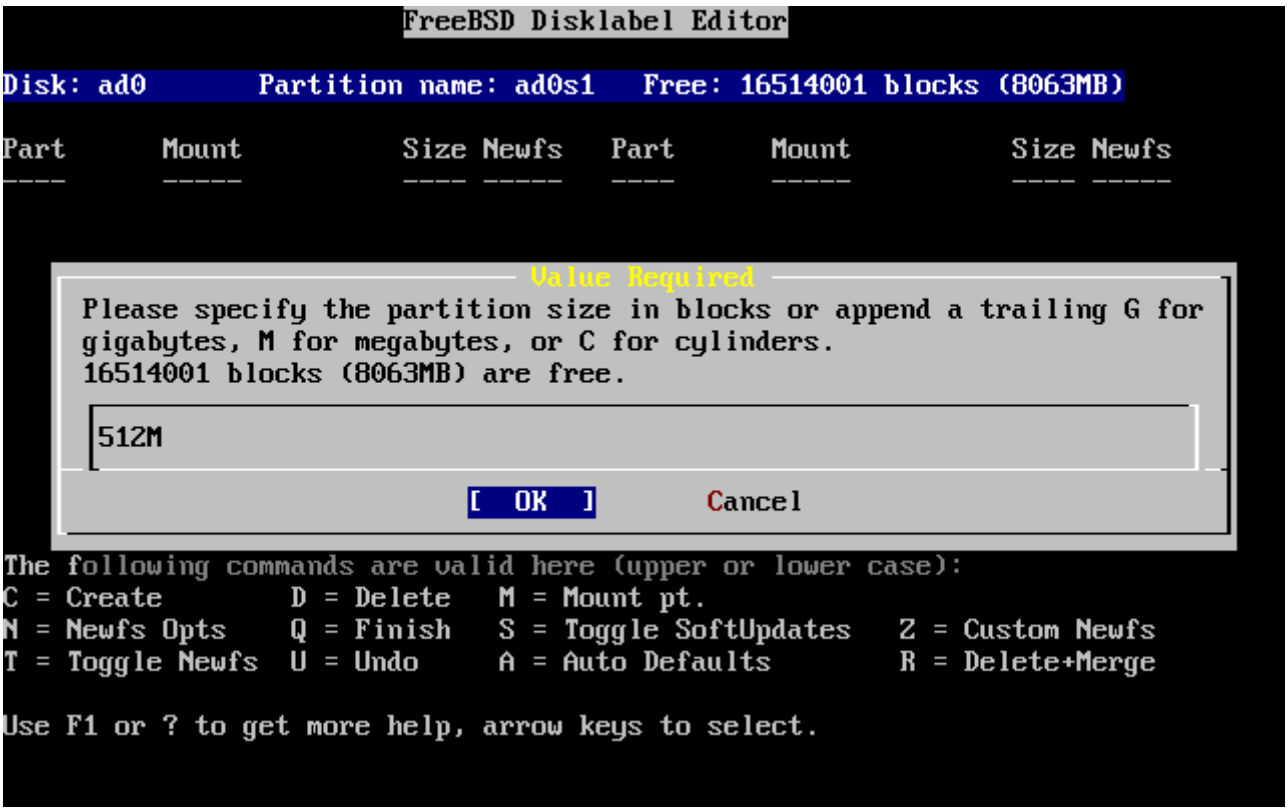
如果□不想使用默□的分区布局， □需要用方向□移□光□并□中第一个分区， 然后按 `D` 来□除它。 重□□一□程直到□除了所有推□的分区。

要建立第一个分区 (a，作□ / - 根文件系□)， □□□□已□在屏幕□部□中了正□的 slice， 然后按 `C`。 接下来将出□一个□□框， 要求□□入新分区的尺寸 (如 根目□使用空□ 所示)。 □可以□入以□□□位的尺寸，或以 `M` 表示MB、 `G` □尾表示GB， 或者 `C` 表示柱面数的方式来表□尺寸。



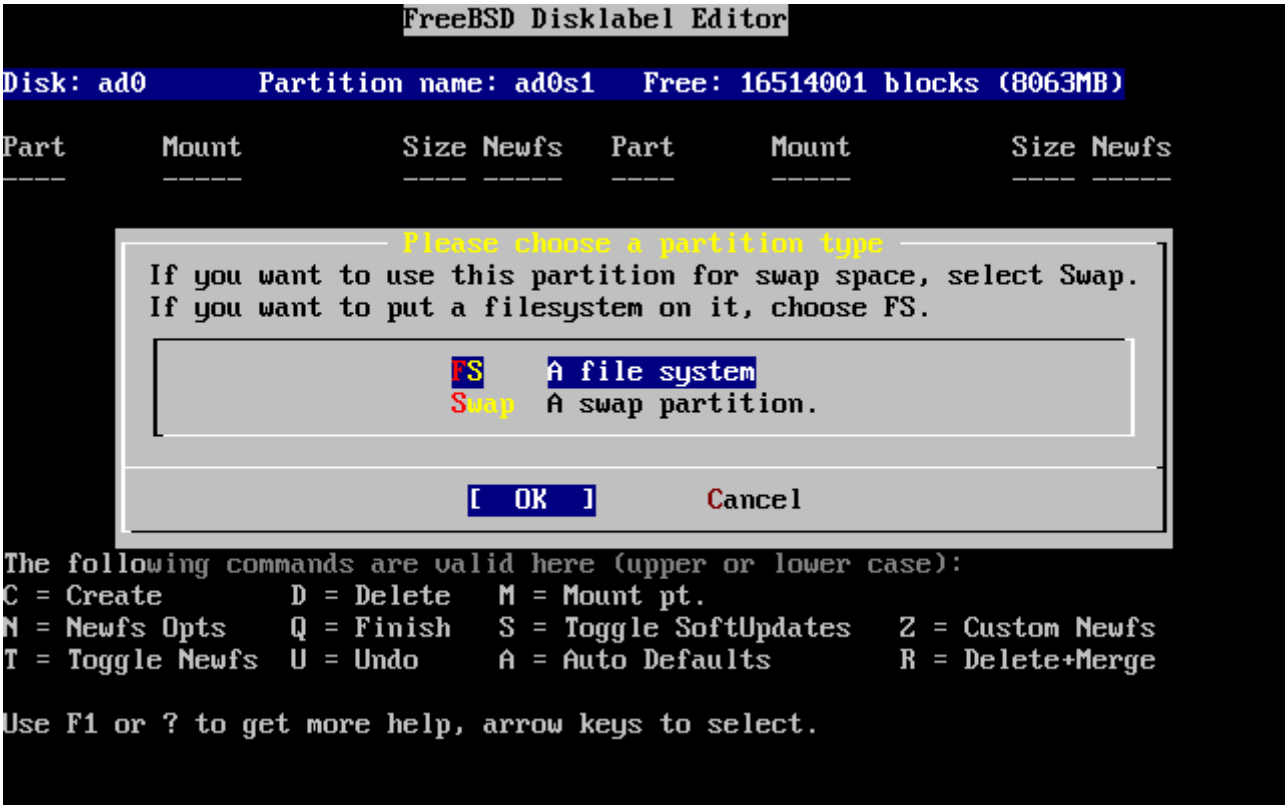
20. 根目使用空

如果使用此示的默尺寸， 会建一个占整个 slice 空余空的 partition。如果希望使用前面例子中描述的 partition 尺寸， 按 `Backspace` 除些数字， 并入 `512M`，如 要分区大小 所示。 然后， 按下 **[OK]**。



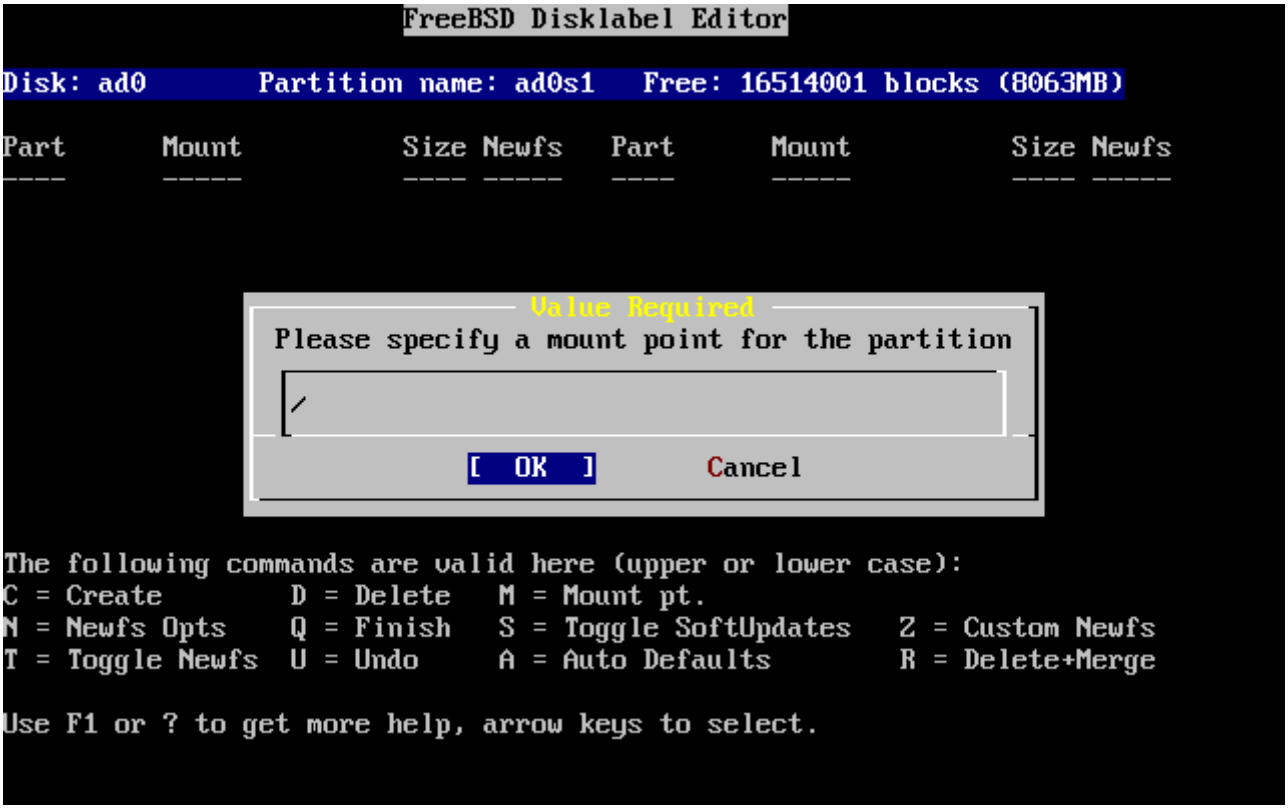
21. 要分区大小

入完大小后接着要建立的分区是文件系是交空，如 根分区型所示。第一个分区是文件系，所以 FS后按 `Enter` 。



□ *22.* □□根分区□型

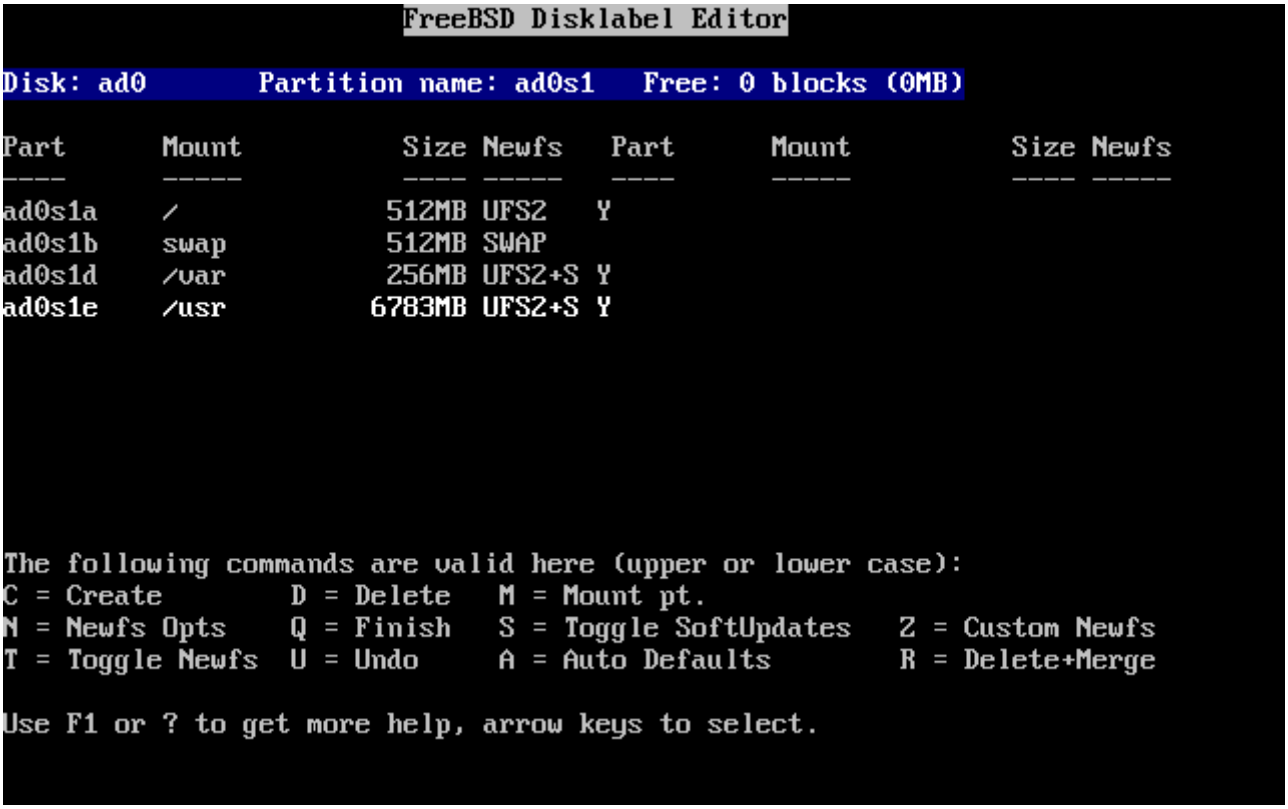
最后，因□□要建立的是一个文件系□，所以必□告□ Disklabel □个文件系□要挂接在什□地方，如 □□根挂接点所示。根文件系□的挂接点 /, 所以□□入 /,然后按 `Enter` □。



□ *23.* □□根挂接点

□□制作好的分区会□示在画面上。 □□□重□上述的□作以建立其它的分区。当建立交□空□的□候， 系□不会□□要将它挂接在□里，因□交□空□是不用挂在系□上的。 当□在建立最后一个分区/usr的□候， □可以直接使用默□的大小，即所有此分区剩余的空□。

□最□的 FreeBSD DiskLabel □□器画面会□似 Sysinstall Disklabel □□器, □□数字按□的□□而有所不同。 按下 Q □完成分区的建立。



□ *24. Sysinstall Disklabel □□器*

2.7. □□要安装的□件包

2.7.1. □□要安装的□件包

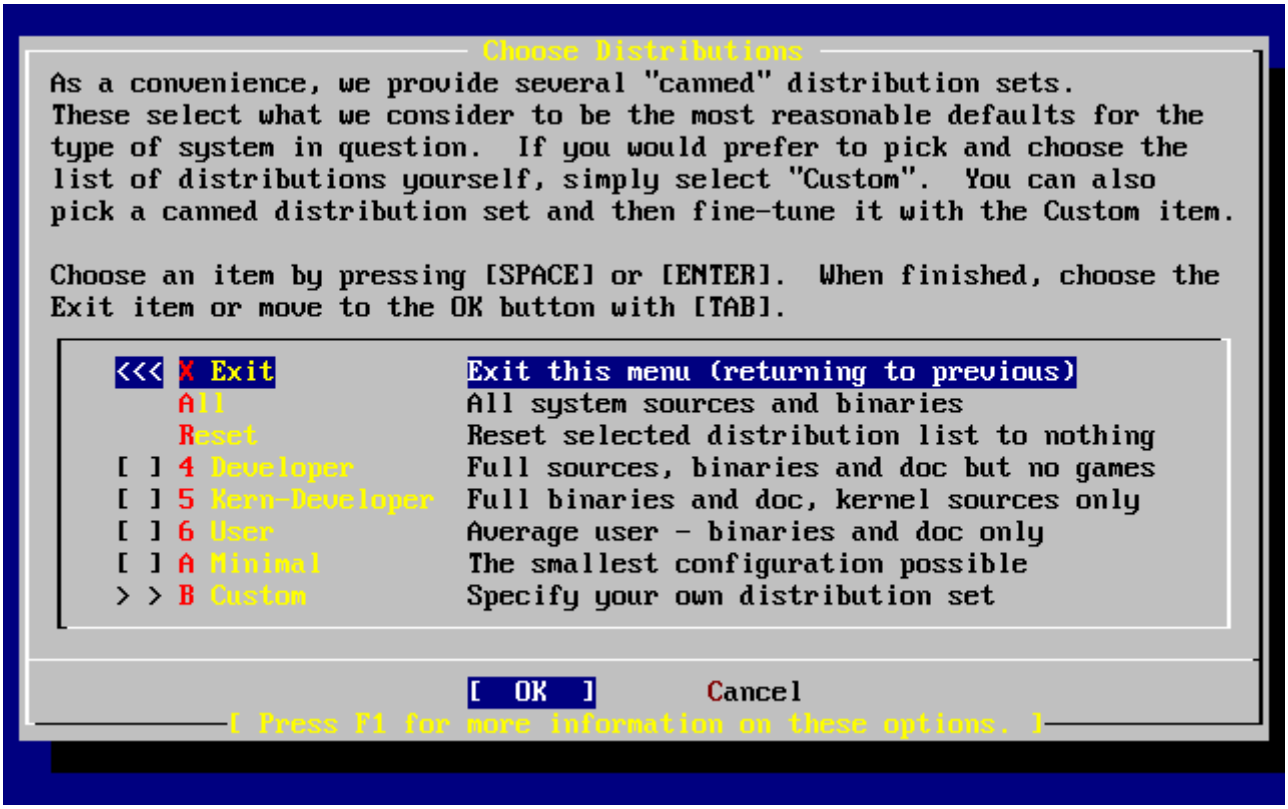
安装□些□件包在很大程度上取决于系□将被用来做什□， 以及有多少可用的磁□空□。内建的□□包括了□行所需要的最小系□， 到把所有□件包全都装上的常用配置。UNIX® 或 FreeBSD 新手通常直接□□一个□定好的□件包就可以了， 而有□□的使用者□可以考□自己□制安装□些□件包。

按下 F1 可以看到有□□件包的更多□□信息， 以及它□都包含了□些□件，之后，可以按 Enter 回到□件包□□画面。

如果需要□形用□界面， □配置 X 服□以及□□默□□面需要在完成 FreeBSD 之后完成。 □于安装和配置 X 服□的信息， 可以在 X Window 系□ □到。

如果需要定制内核， □□需要□□包含源代□的那个□□。 要了解□什□□□□和□建新的内核， □参□ 配置FreeBSD的内核。

□然， 包含所有□件的系□是最万能的。 如果磁□空□足□， 用光□□□□ □□□件包 中的 All 并按 Enter。如果担心磁□空□不□的□， □□□最合□的□□。 不要担心□□的是否是最合□的， 因□其他□件包可以在安装完□后再加入□来。



□ *25.* □□□件包

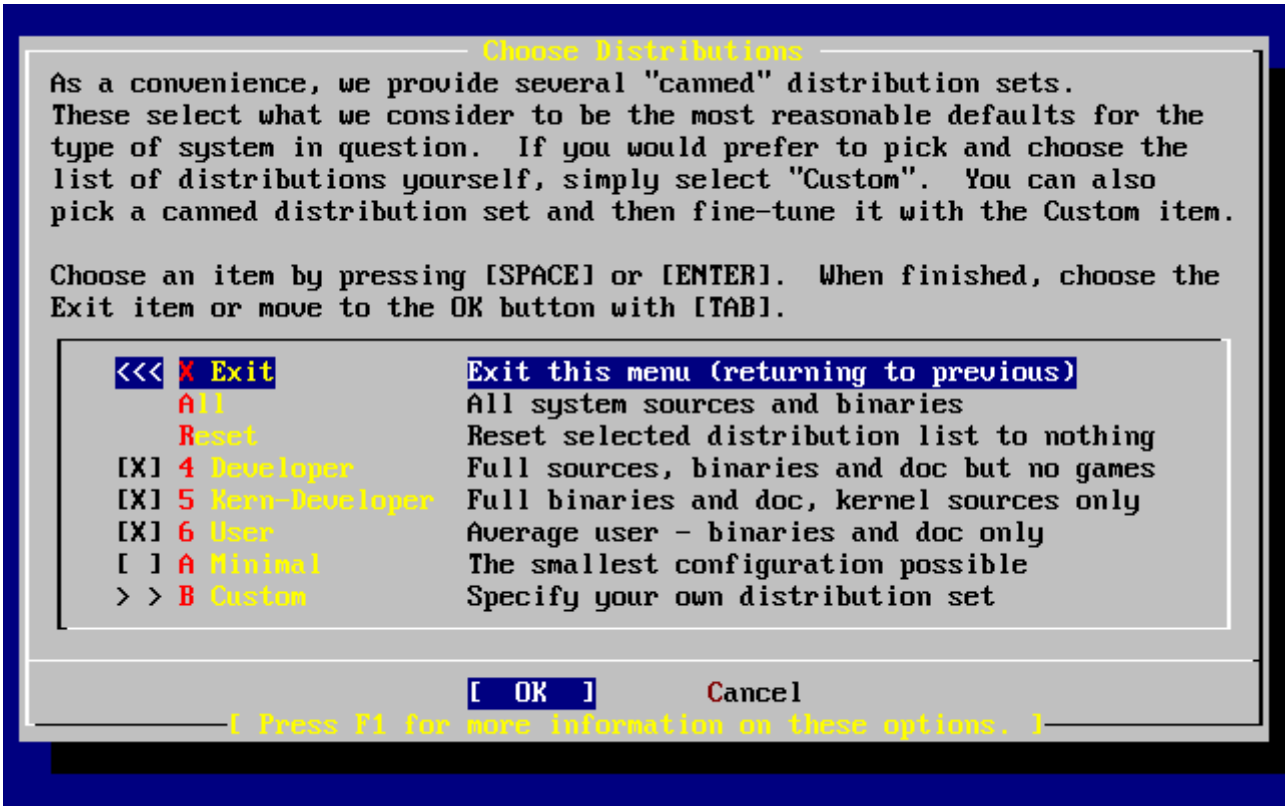
2.7.2. 安装ports□件包

当□□完□想要安装的部分后，接着会□□□要不要安装FreeBSD Ports □件包；Ports□件包可以□□□□方便地安装□件包。Ports本身并不包含□□ □件所需要的程序源代□，而是一个包含自□下□、□□以及安装的文□集合。 安装□用程序: Packages 和 Ports 一章□□如何使用Ports.

安装程序并不会□□□是否有足□的硬□空□， 在□□□一□之前□先□定□有足□的硬□空□。 目前 FreeBSD 12.0 版本中， FreeBSD Ports Collection 大□占用 500 MB 大小的硬□空□。 □于近期的版本□可能需要更多一些空□来安装他□。

```
                        User Confirmation Requested
Would you like to install the FreeBSD Ports Collection?

This will give you ready access to over 24,000 ported software packages,
at a cost of around 500 MB of disk space when "clean" and possibly much
more than that if a lot of the distribution tarballs are loaded
(unless you have the extra CDs from a FreeBSD CD/DVD distribution
available and can mount it on /cdrom, in which case this is far less
of a problem).

The Ports Collection is a very valuable resource and well worth having
on your /usr partition, so it is advisable to say Yes to this option.

For more information on the Ports Collection & the latest ports,
visit:
    http://www.FreeBSD.org/ports

                              [ Yes ]     No
```

□□ **[yes]** 将会安装 Ports Collection， 而□□ **[no]** □将跳□它。 □好后按 Enter □□。 此后， □□安装的□件包的屏幕将再次出□。



□ *26.* □□□要安装的□件包

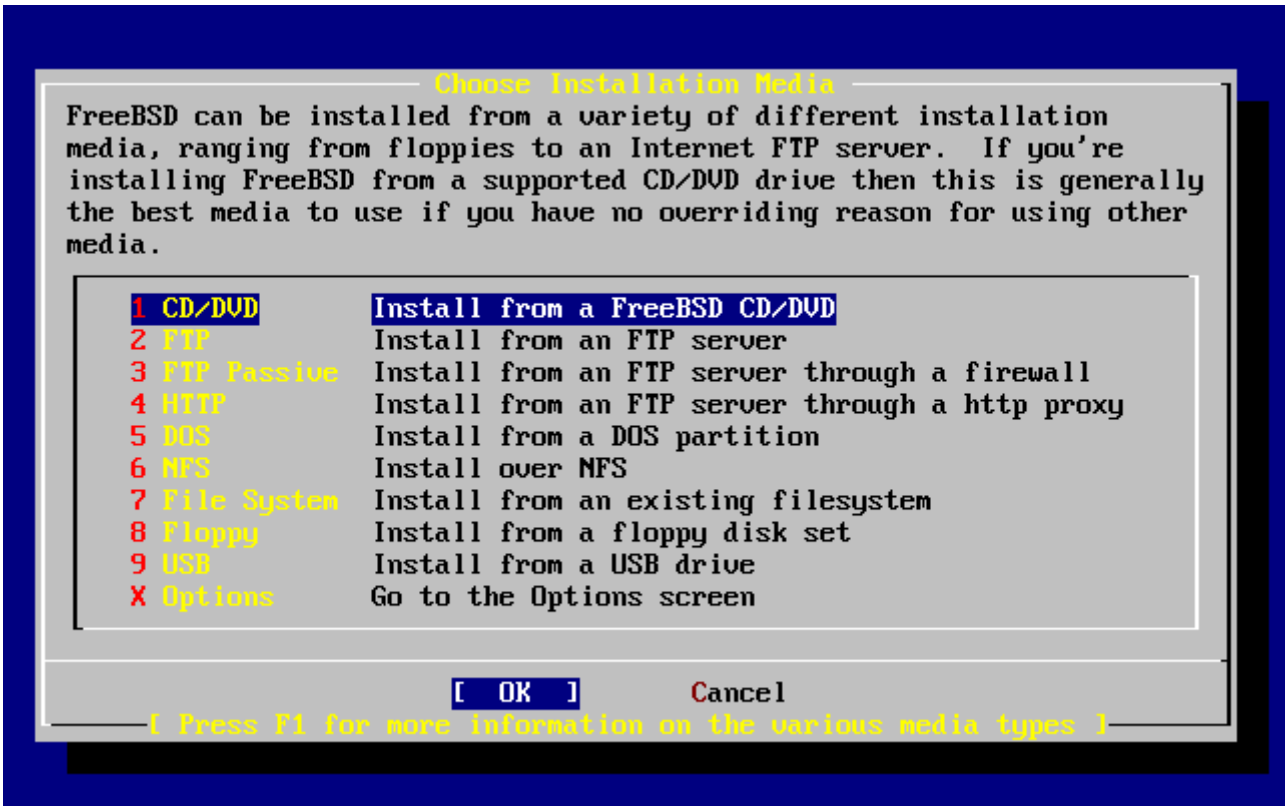
如果□□的□□感到□意，□□□Exit 退出，□保**[OK]** 被高亮□示，然后按 Enter □□。

2.8. □□□要使用的安装介□

如果要从 CDROM 或 DVD安装，使用方向□将光□移到 Install from a FreeBSD CD/DVD。□□ **[OK]** 被□取，然后按 `Enter` □始安装程序。

如果要使用其它的方式安装， □□□□当的安装介□然后按照屏幕指示□行安装。

按 `F1` 可以□示安装介□的在□□明。按一下 `Enter` 可返回□□安装介□画面。



□ *27.* □□安装介□

*FTP*安装模式

使用FTP安装，有三□方式：主□式（active）FTP、被□式（passive）FTP 或是透□HTTP代理服□器。

主□式**FTP**： 从**FTP**服□器安装

□个□□将会使所有的FTP□□使用 "Active"模式。 □将无法通□防火□，但是可以使用在那些比□早期， 不支持被□模式的FTP站。如果□的□接在使用被□（默□□） 模式□住了，□□主□模式看看！

被□模式**FTP**：通□防火□从**FTP**服□器安装

此□□会□ sysinstall 使用 "Passive"模式来安装。□使得使用者可以穿□ 不允□用非固定TCP PORTS□入的防火□。

FTP 透□ **HTTP** 代理服□器： 透□**HTTP**代理服□器， 由 **FTP** 服□器安装

此□□会□ sysinstall 通□HTTP□□ （像□□器一□）□到proxy服□器。 proxy服□器会解□送出的□求，然后通知FTP服□器。 因□通□HTTP□□，所以可以穿□防火□。 要用□□方式，□必□指定proxy服□器的地址。

□于一个 FTP 代理服□器而言， 通常在使用者登入名称中加入□要登入的服□器的用□名，加在 "@" 符号后面。然后代理服□器就会 "假装" 成一个真的服□器。例如，假□□要从 ftp.FreeBSD.org 安装，通□ FTP 代理服□器 foo.example.com， 使用1234端口。

在□□情况下，□可以到 options 菜□，将 FTP username □□ ftp@ftp.FreeBSD.org，密□□□□的□子□件地址。 安装介□部分，指定FTP (或是被□式 FTP，如果代理服□器支持的□)以及URL□ ftp://foo.example.com:1234/pub/FreeBSD。

因□ftp.FreeBSD.org的 /pub/FreeBSD 目□会被□取到 foo.example.com之下，□就可以从□台 机器 (会从 ftp.FreeBSD.org □取文件) 安装。

2.9. 安装□□

到此□止，可以□始□行安装了， □也是□避免更□到□的硬□的最后机会。

```
                    User Confirmation Requested
Last Chance! Are you SURE you want to continue the installation?

If you're running this on a disk with data you wish to save then WE
STRONGLY ENCOURAGE YOU TO MAKE PROPER BACKUPS before proceeding!

We can take no responsibility for lost disk contents!

                          [ Yes ]    No
```

□□ **[yes]** 然后按下 Enter □□安装

安装所需的□□会根据□所□□的□件、 安装介□以及□□□的速度而有所不同。 在安装的□程中会有一些信息来□示目前的□度。

当□看到下面的信息表示已□安装完成了：

```
                              Message

Congratulations! You now have FreeBSD installed on your system.

We will now move on to the final configuration questions.
For any option you do not wish to configure, simply select No.

If you wish to re-enter this utility after the system is up, you may
do so by typing: /usr/sbin/sysinstall.

                                 [ OK ]

                       [  Press enter or space  ]
```

按下 `Enter` 以□行安装后的配置。

□□ **[no]** 然后按 `Enter` 会取消安装，不会□□的系□造成更□。□会看到下面的信息：

```
                              Message
Installation complete with some errors.  You may wish to scroll
through the debugging messages on VTY1 with the scroll-lock feature.
You can also choose "No" at the next prompt and go back into the
installation menus to retry whichever operations have failed.

                                 [ OK ]
```

□生□个信息是因□什□□西也没有安装，按下 `Enter` 后会□□安装程序回到主安装界面。从主安装界面可以退出安装程序。

2.10. 安装后的配置

安装成功后，就可以□行□一□的配置了。引□新安装的 FreeBSD 系□之后，使用 `sysinstall` 并□□ Configure。

2.10.1. 配置网□

如果□之前配置用 PPP 通□ FTP 安装，那□□个画面将不会出□；正像所□的那□，□可以□后再做配置。

如果想更多的了解网□或将FreeBSD配置□网□或路由器，□参考 Advanced Networking 的相□文章。

```
                    User Confirmation Requested
   Would you like to configure any Ethernet or PPP network devices?

                           [ Yes ]   No
```

如果要配置网□，□□□ **[yes]** 然后按 `Enter`。 否□□□□ **[no]** □□。

□ *28.* □□网□□□

用方向□□□□要配置的网□接口，然后按`Enter`。

```
                  User Confirmation Requested
      Do you want to try IPv6 configuration of the interface?

                           Yes   [ No ]
```

目□私人区域网□IP□□IPv4已□足□，所以□□ **[no]** 然后按 `Enter`。

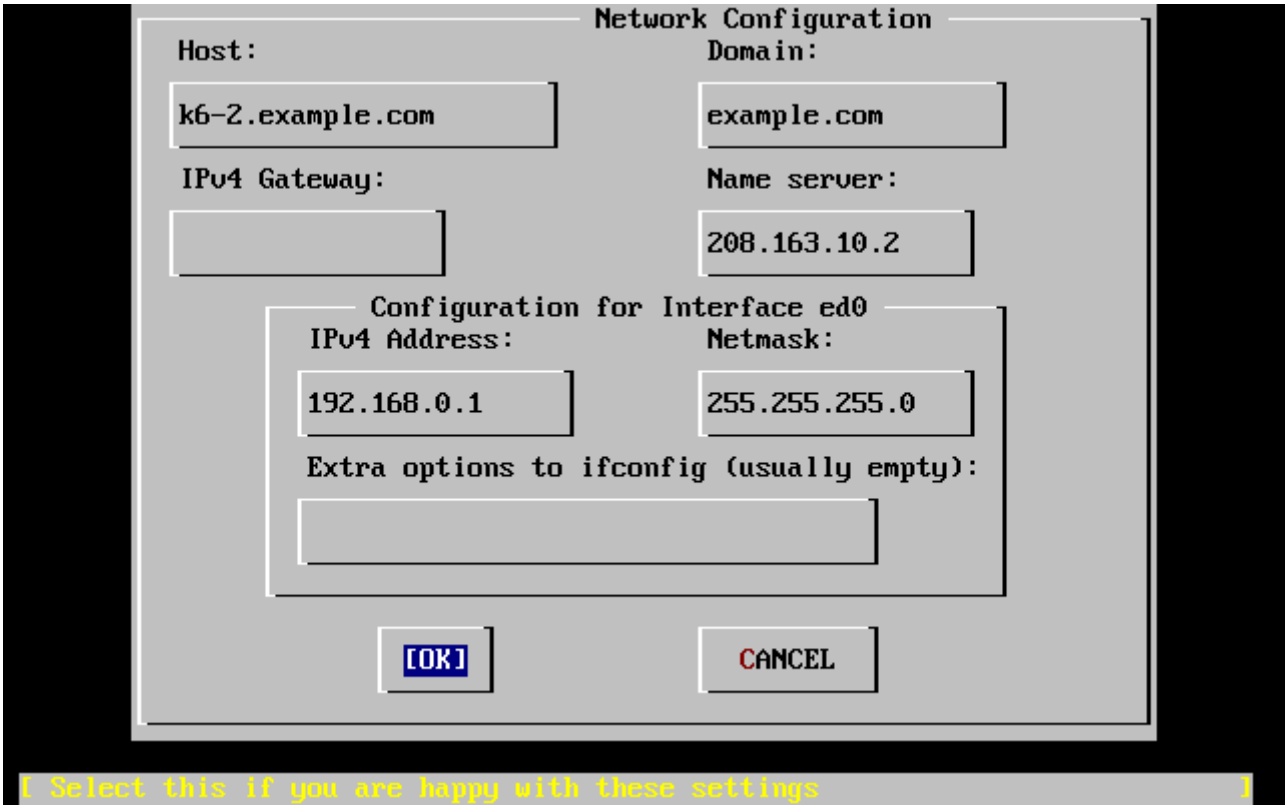
如果想□□新的IP通信□□ IPv6， 使用 RA 服□，□□□ **[yes]** 然后按 `Enter`。 □□ RA 服□器将会花□几秒的□□。

```
                          User Confirmation Requested
      Do you want to try DHCP configuration of the interface?

                           Yes   [ No ]
```

如果□不需要 DHCP (Dynamic Host Configuration Protocol □□主机配置□□)，□□ **[no]** 然后按`Enter`。

□□ **[yes]** 会□行dhclient， 如果成功，它会自□将网□配置信息填上。更多的信息□参考 网□自□配置 (DHCP)。

下面的网□配置□示了□□把以太网□□配置成区域网□网□的角色。



□ *29.* 配置 *ed0*接□

使用Tab □可以在各个□目之□□行切□，□□入□当 的信息：

Host（机器名称）

完整的机器名称，例如本例中的 k6-2.example.com 。

Domain（域名）

□机器所在的域名称，如本例的 example.com

IPv4 Gateway（**IPv4**网□）

□入将数据包□送到□端网□的机器IP地址。 只有当机器是网□上的一个□点□才要□入。 如果□台机器要作□□局域网的网□， □将此□□□空白。IPv4网□， 也被称作默□网□或默□路由器。

域名服□器

本地网□中的域名服□器的IP地址。 本例中假□机器所在的网□中没有域名服□器， 所以填入的是ISP提供的域名服□器地址 （208.163.10.2。）

IPv4 地址

本机所使用的IP地址。本例□ 192.168.0.1。

子网掩□

在□个局域网中所使用的地址□是 192.168.0.0 - 192.168.0.255， □□的子网掩□是 255.255.255.0。

ifconfig □外参数□定

任何ifconfig命令跟网□接口有□的参数。 本□例中没有。

使用 Tab □□□ **[OK]**然后按 Enter □。

```
                    User Confirmation Requested
       Would you like to bring the ed0 interface up right now?

                              [ Yes ]   No
```

□□ **[yes]** 然后按 `Enter` 将会将机器的网□□□□用状□。 机器下次□□的□候即可使用。

2.10.2. 配置网□

```
                    User Confirmation Requested
      Do you want this machine to function as a network gateway?

                              [ Yes ]    No
```

如果□台机器要作□本地网□和其它机器之□□送数据包的网□，□□□ **[yes]** 然后按 `Enter`。 如果□台机器只是网□上的普通□点，□□□ **[no]** 并按 `Enter` □□。

2.10.3. 配置网□服□

```
                    User Confirmation Requested
Do you want to configure inetd and the network services that it provides?

                               Yes   [ No ]
```

如果□□ **[no]**， □多网□服□，如 telnetd 将不会□用。 □□， □端用□将无法 telnet □入□台机器。 本机上的用□□是可以 telnet到□端机器的。

□些服□可以在安装完成后修改/etc/inetd.conf 配置文件来□用它□。□参□ □□ 以□得更多的信息。

如果□想□在就配置□些网□服□，□□□ **[yes]**， 然后会看到下面的信息：

```
                    User Confirmation Requested
The Internet Super Server (inetd) allows a number of simple Internet
services to be enabled, including finger, ftp and telnetd.  Enabling
these services may increase risk of security problems by increasing
the exposure of your system.

With this in mind, do you wish to enable inetd?

                              [ Yes ]   No
```

□□ **[yes]** □□。

```
                      User Confirmation Requested
inetd(8) relies on its configuration file, /etc/inetd.conf, to determine
which of its Internet services will be available.  The default FreeBSD
inetd.conf(5) leaves all services disabled by default, so they must be
specifically enabled in the configuration file before they will
function, even once inetd(8) is enabled.  Note that services for
IPv6 must be separately enabled from IPv4 services.

Select [Yes] now to invoke an editor on /etc/inetd.conf, or [No] to
use the current settings.

                               [ Yes ]   No
```

□□ **[yes]** 将允□□添加网□服□ (或将相□网□服□□行□□的 # 除掉即可。)

```
^[ (escape) menu  ^y search prompt  ^k delete line   ^p prev li   ^g prev page
^o ascii code     ^x search         ^l undelete line ^n next li   ^v next page
^u end of file    ^a begin of line  ^w delete word   ^b back 1 char
^t top of text    ^e end of line    ^r restore word  ^f forward 1 char
^c command        ^d delete char    ^j undelete char ^z next word
=====line 1 col 0 lines from top 1 ============================================
# $FreeBSD: src/etc/inetd.conf,v 1.73.10.2.4.1 2010/06/14 02:09:06 kensmith Exp
#
# Internet server configuration database
#
# Define *both* IPv4 and IPv6 entries for dual-stack support.
# To disable a service, comment it out by prefixing the line with '#'.
# To enable a service, remove the '#' at the beginning of the line.
#
#ftp    stream  tcp     nowait  root    /usr/libexec/ftpd       ftpd -l
#ftp    stream  tcp6    nowait  root    /usr/libexec/ftpd       ftpd -l
#ssh    stream  tcp     nowait  root    /usr/sbin/sshd          sshd -i -4
#ssh    stream  tcp6    nowait  root    /usr/sbin/sshd          sshd -i -6
#telnet stream  tcp     nowait  root    /usr/libexec/telnetd    telnetd
#telnet stream  tcp6    nowait  root    /usr/libexec/telnetd    telnetd
#shell  stream  tcp     nowait  root    /usr/libexec/rshd       rshd
#shell  stream  tcp6    nowait  root    /usr/libexec/rshd       rshd
#login  stream  tcp     nowait  root    /usr/libexec/rlogind    rlogind
#login  stream  tcp6    nowait  root    /usr/libexec/rlogind    rlogind
file "/etc/inetd.conf", 118 lines
```

□ *30. □□ inetd.conf*配置文件

在加入□想□用的服□后，按下 `Esc` □会出□一个 □□框可以□□□□以及保存修改。

2.10.4. □用 SSH 登□

```
                      User Confirmation Requested
                   Would you like to enable SSH login?
                           Yes       [  No  ]
```

□□ **[yes]** 便会□用 sshd(8)， 也就是 OpenSSH 服□程序。 它能□□□以安全的方式从□程□□机器。 如欲了解□于 OpenSSH 的□一□□情， □参□ OpenSSH。

2.10.5. 匿名 FTP

```
                      User Confirmation Requested
Do you want to have anonymous FTP access to this machine?

                               Yes    [ No ]
```

2.10.5.1. 不允□匿名 FTP□□

□□默□的 **[no]** 并按下 `Enter` □将仍然可以□在□台机器上有□号的用□□□ FTP。

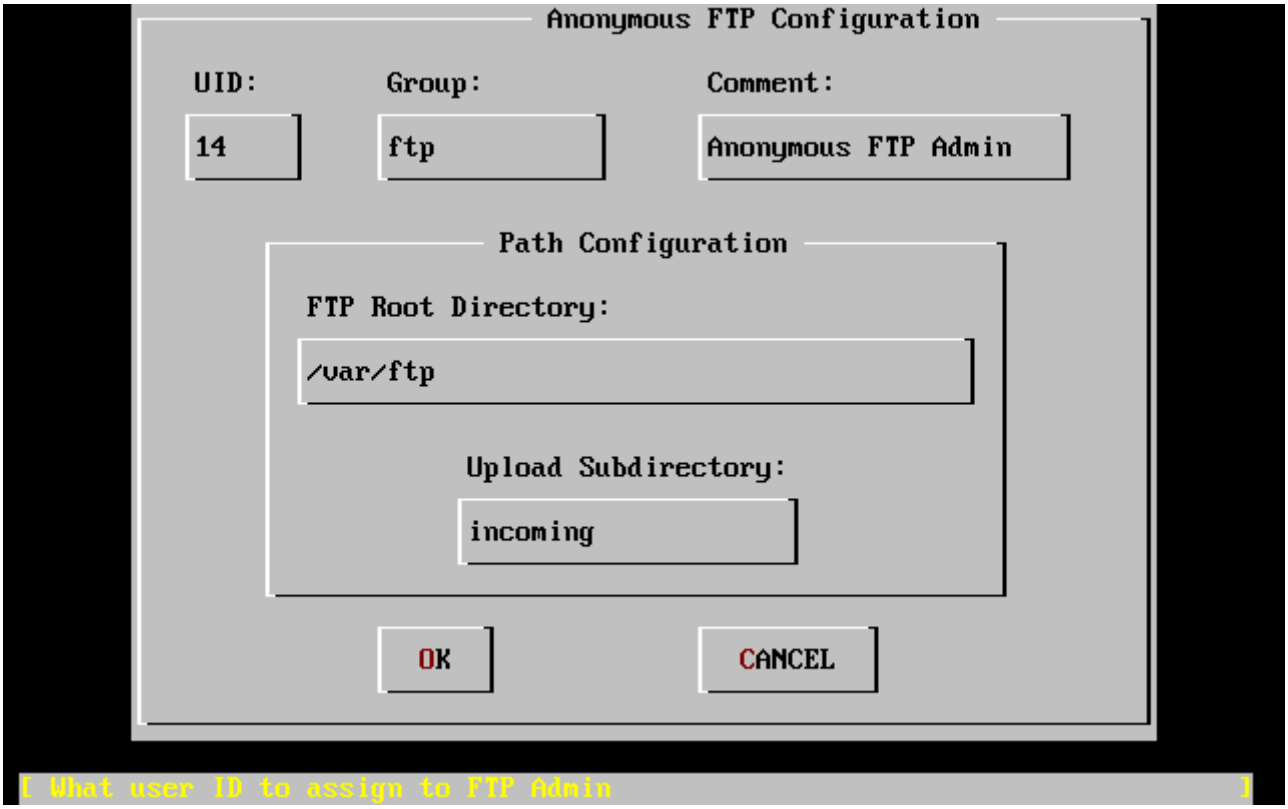
2.10.5.2. 允□匿名 FTP□□

如果□□□允□匿名 FTP 存取， 那□网□中任何人都可以使用FTP来□□□的机器。 在□用匿名□□之前□□考□网□的安全□□。 如果要知道更多有□网□安全的信息， □参□ 安全。

要□用FTP匿名□□，用方向□□□ **[yes]** 并按 `Enter` □。 系□会□出□一□的□□信息：

```
                      User Confirmation Requested
Anonymous FTP permits un-authenticated users to connect to the system
FTP server, if FTP service is enabled.  Anonymous users are
restricted to a specific subset of the file system, and the default
configuration provides a drop-box incoming directory to which uploads
are permitted.  You must separately enable both inetd(8), and enable
ftpd(8) in inetd.conf(5) for FTP services to be available.  If you
did not do so earlier, you will have the opportunity to enable inetd(8)
again later.

If you want the server to be read-only you should leave the upload
directory option empty and add the -r command-line option to ftpd(8)
in inetd.conf(5)

Do you wish to continue configuring anonymous FTP?

                          [ Yes ]         No
```

□些信息会告□□ FTP 服□□需要在 /etc/inetd.conf 中□用。 假如□希望允□匿名 FTP □接， □参□ 配置网□服□。 □□ **[yes]** 并按 `Enter` □□； 系□将□出下列信息：



□ *31.* 默□的匿名 *FTP* 配置

使用 `Tab` 在不同的信息字段之□切□， 并填写必要的信息：

UID

用于分配□匿名 FTP 用□的用□ ID。 所有上□的文件的属主都将是□个 ID。

Group

匿名 FTP 用□所在的□。

Comment

用于在 /etc/passwd 中描述□用□的□明性信息。

FTP Root Directory

可供匿名 FTP 用□使用的文件所在的根目□。

Upload Subdirectory

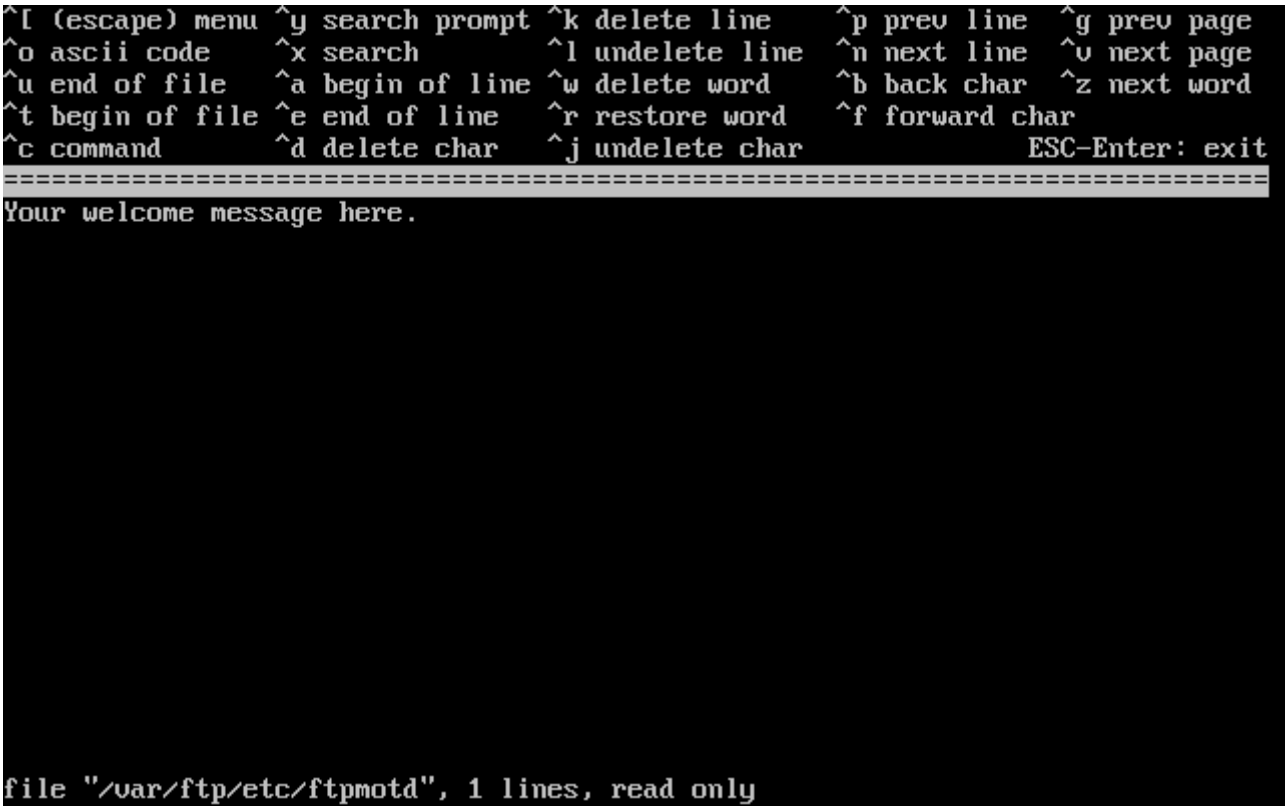
匿名 FTP 用□上□的文件的存放位置。

默□的 FTP 根目□将放在 /var 目□下。 如果□的 /var 目□空□不足以□付□的FTP需求， □可以将FTP的根目□改□ /usr 目□下的 /usr/ftp 目□。

当□□一切配置都□意后，□按 `Enter` □□□。

```
                    User Confirmation Requested
      Create a welcome message file for anonymous FTP users?

                         [ Yes ]    No
```

如果□□□ **[yes]** 并按下 `Enter`□， 系□会自□打□文本□□器□□□□FTP的□迎信息。



□ *32.* □□*FTP*□迎信息

此文本□□器叫做 ee。 按照指示修改信息文本或是□后再用□喜□的文本□□器来修改。 □□住画面下方□示的文件位置。

按 `Esc` 将□出一个默□□ a) leave editor的□□框。按 `Enter` 退出并□□。再次按 `Enter` 将保存修改。

2.10.6. 配置网□文件系□

网□文件系□ （NFS） 可以□□可以在网□上共享□的文件。 一台机器可以配置成NFS服□器、客□端或□者并存。□参考 网□文件系□（NFS) 以□得更多的信息。

2.10.6.1. NFS 服□器

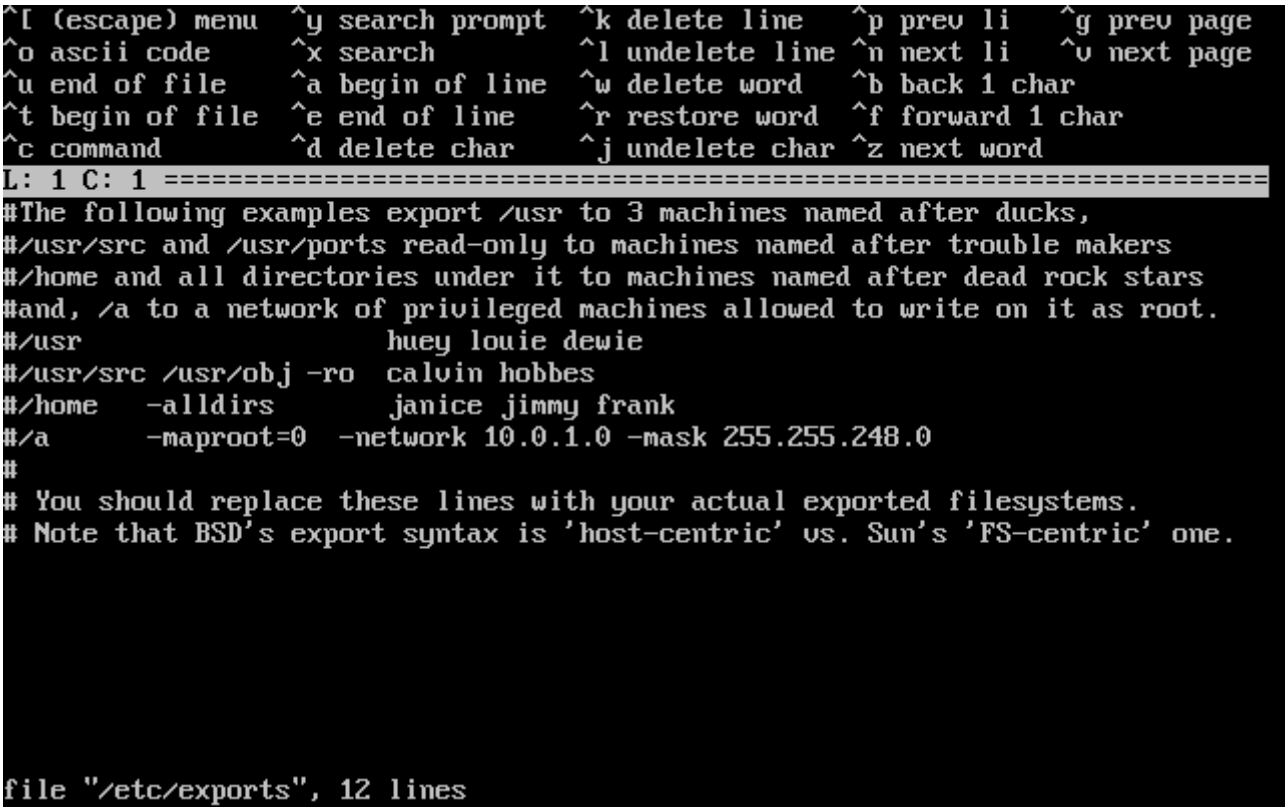
```
                    User Confirmation Requested
Do you want to configure this machine as an NFS server?

                              Yes    [ No ]
```

如果□不想安装网□文件系□，□□□ **[no]** 然后按 `Enter`□。

如果□□□ **[yes]** 将会出□一个□□框提醒□必□先建立一个 exports 文件。

```
                              Message
Operating as an NFS server means that you must first configure an
/etc/exports file to indicate which hosts are allowed certain kinds of
access to your local filesystems.
Press [Enter] now to invoke an editor on /etc/exports
                              [ OK ]
```

按 Enter □□□。系□会□□文本□□器□□□□ exports 文件。



□ *33.* □□ *exports*文件

按照指示加入真□□出的文件目□或是□后用□喜□的□□器自行□□。 □□下画面下方□示的文件名称及位置。

按下 Esc □会出□一具□□框，默□□□是 a) leave editor。按下 Enter □□并□□。

2.10.6.2. NFS 客□端

NFS 客□端允□□的机器□□NFS服□器。

```
                      User Confirmation Requested
 Do you want to configure this machine as an NFS client?

                              Yes   [ No ]
```

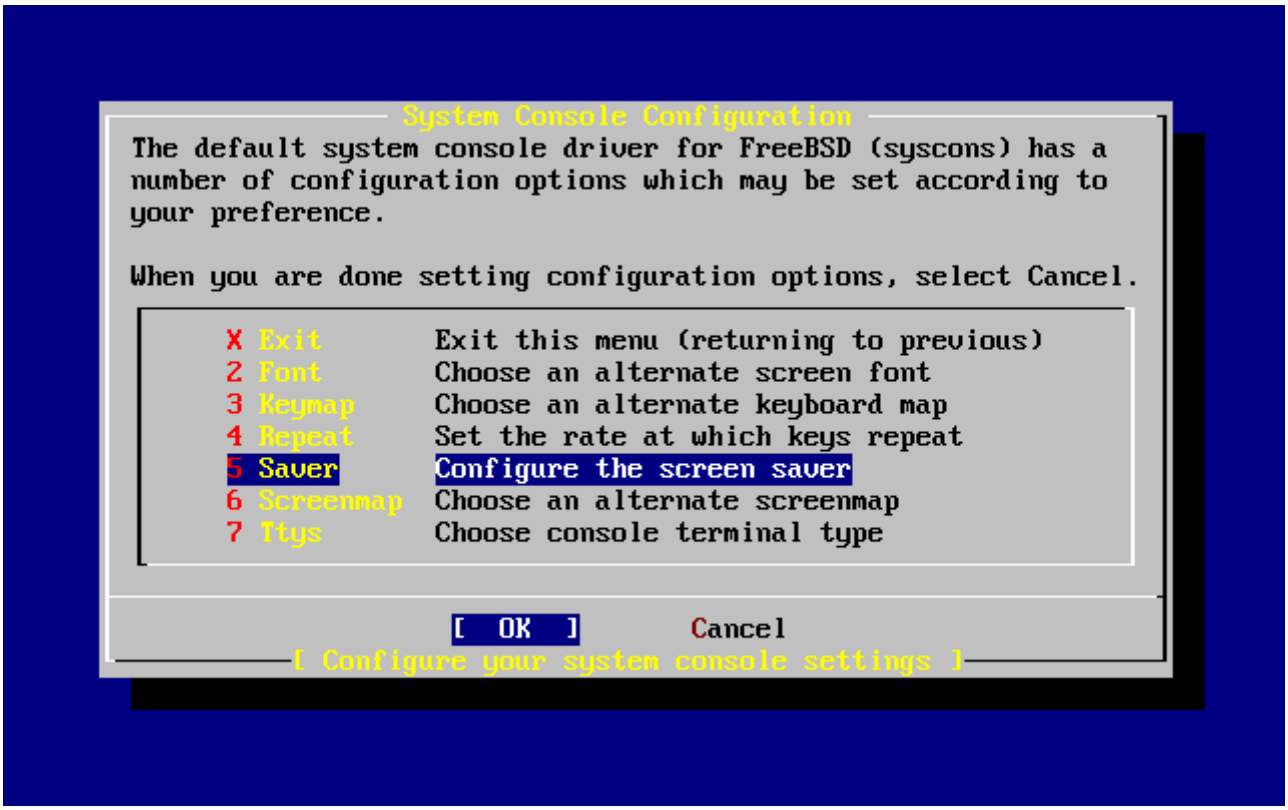
按照□的需要，□□ **[yes]** 或 **[no]** 然后按 Enter。

2.10.7. 配置系□□端

系□提供了几个□□可以□□配置□端的表□方式。

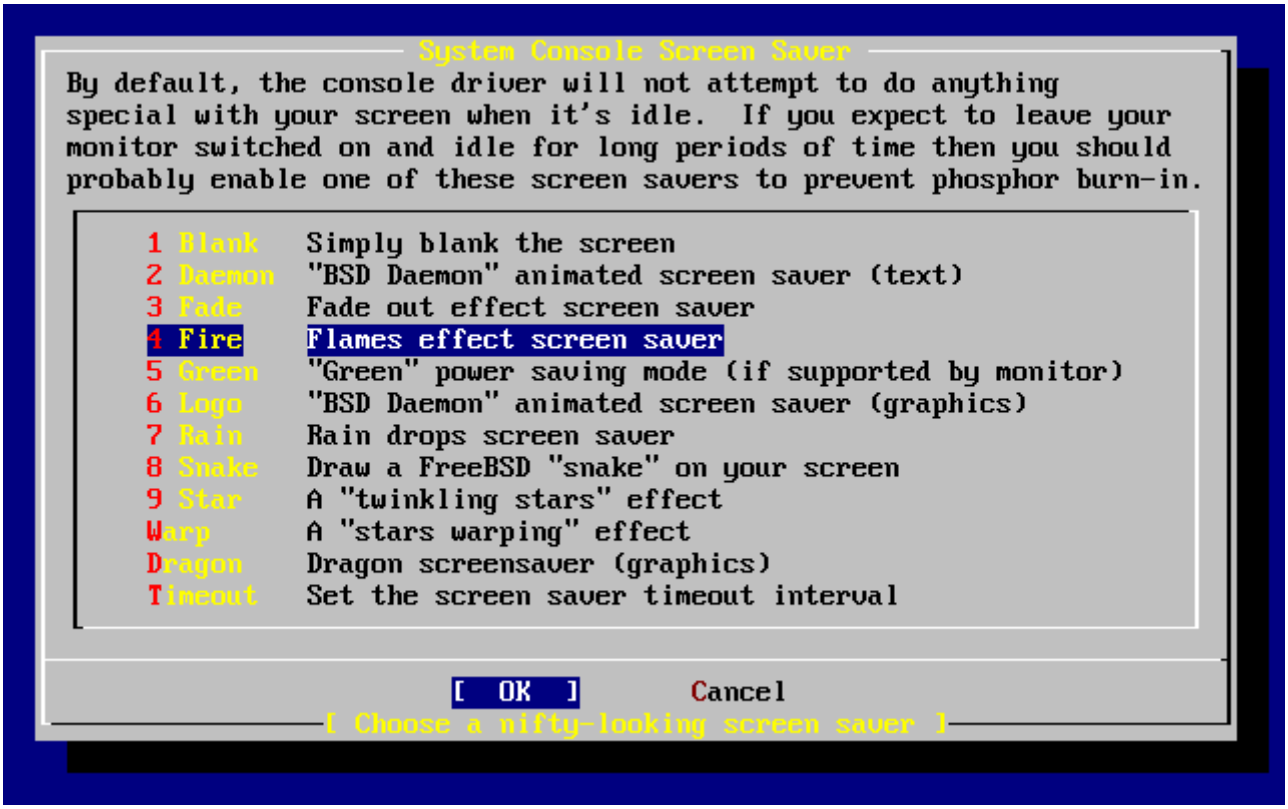
```
                    User Confirmation Requested
       Would you like to customize your system console settings?

                                [ Yes ]  No
```

要□□及配置□些□□，□□□ **[yes]** 并按 `Enter`。



□ *34.* 系□□端配置□□

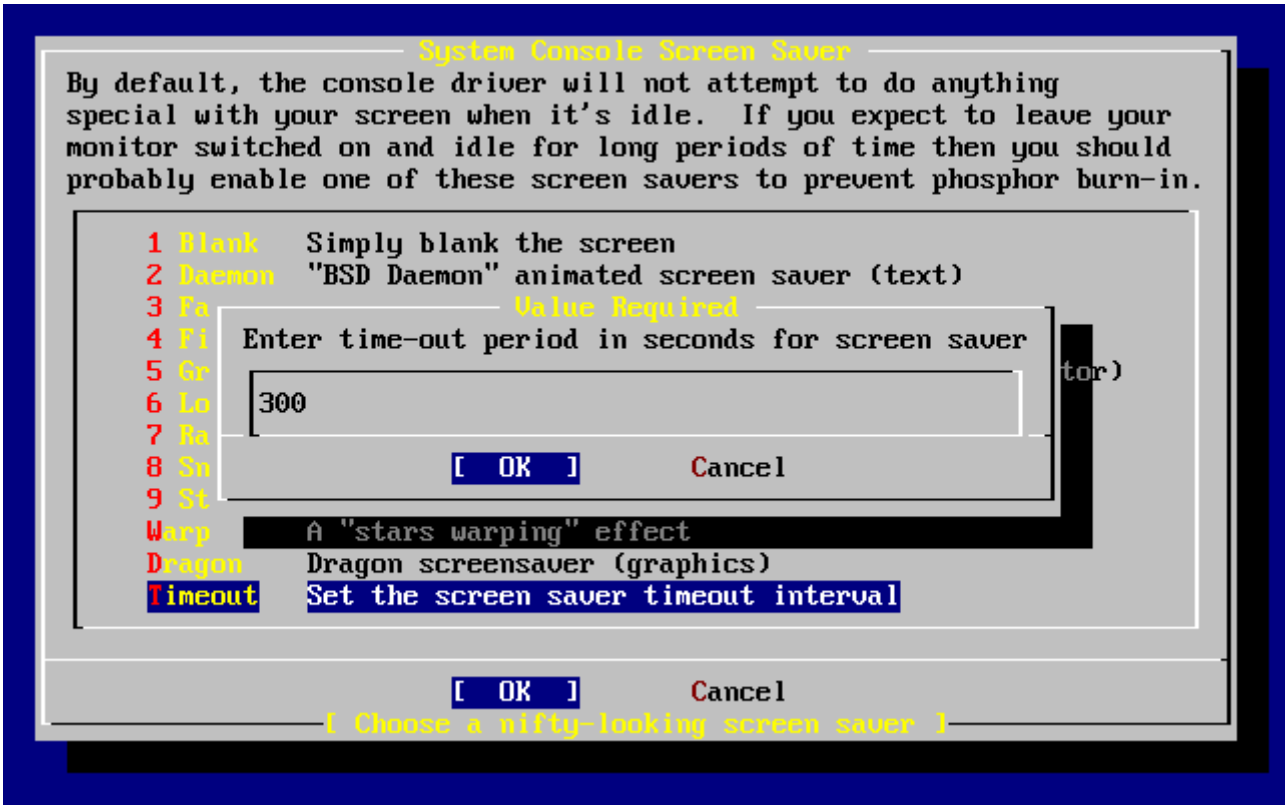
最常用的□□就是屏幕保□程序了。使用方向□将光□移□到 Saver 然后按 `Enter`。



35. 屏幕保□程序□□

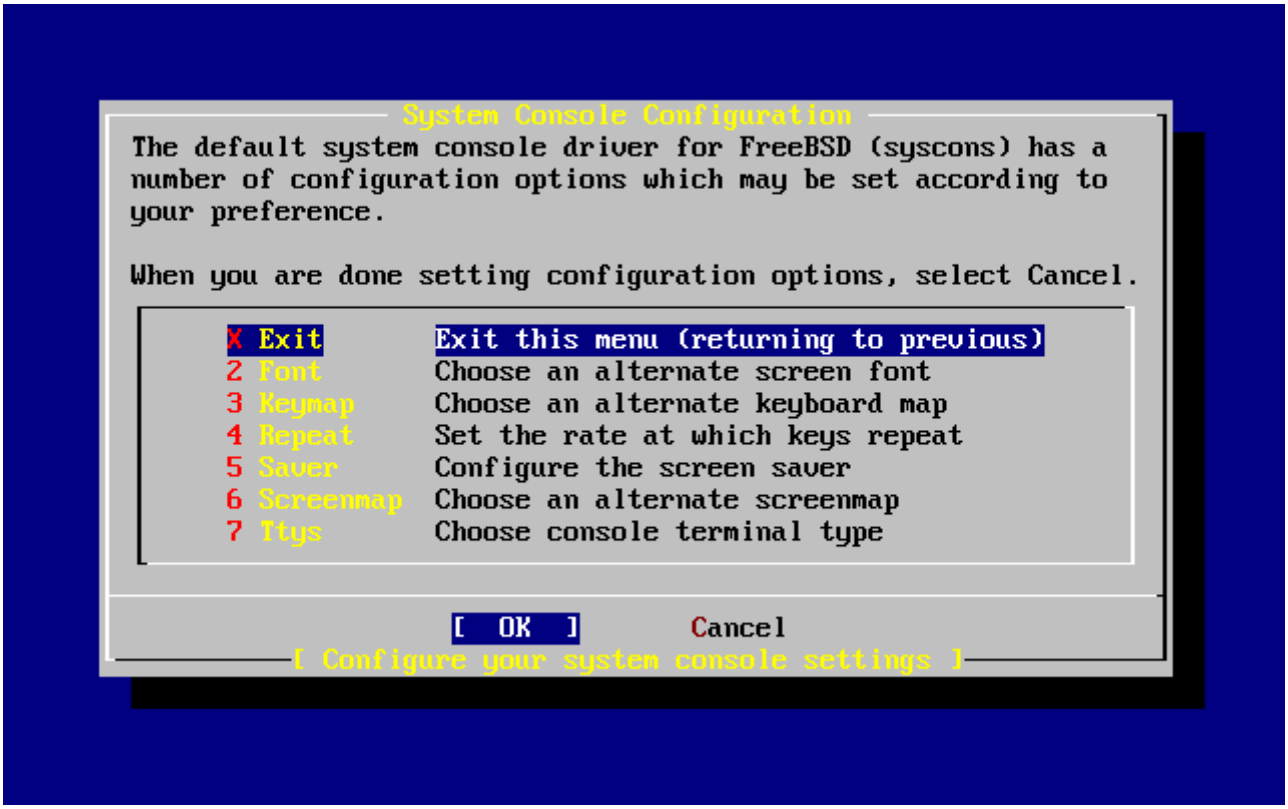
□□□想使用的屏幕保□程序，然后按 `Enter`。 之后回到系□□端配置画面。

默□□□屏幕保□程序的□□是300秒。如果要更改此□□，□再次□□ Saver 。然后□□ Timeout 并按 `Enter` □。系□会□出一个□□框如下：



36. 屏幕保□□□□置

□可以直接改□□个□，然后□ **[OK]**并按 `Enter` □回到系□□端配置画面。



37. 退出系□□端配置

□□ Exit 然后按下 `Enter` □会回到安装后的配置画面。

2.10.8. 配置□区

配置□机器的□区可以□系□自□校正任何区域□□的□更， 并且在□行一些跟□区相□的程序□不会出□。

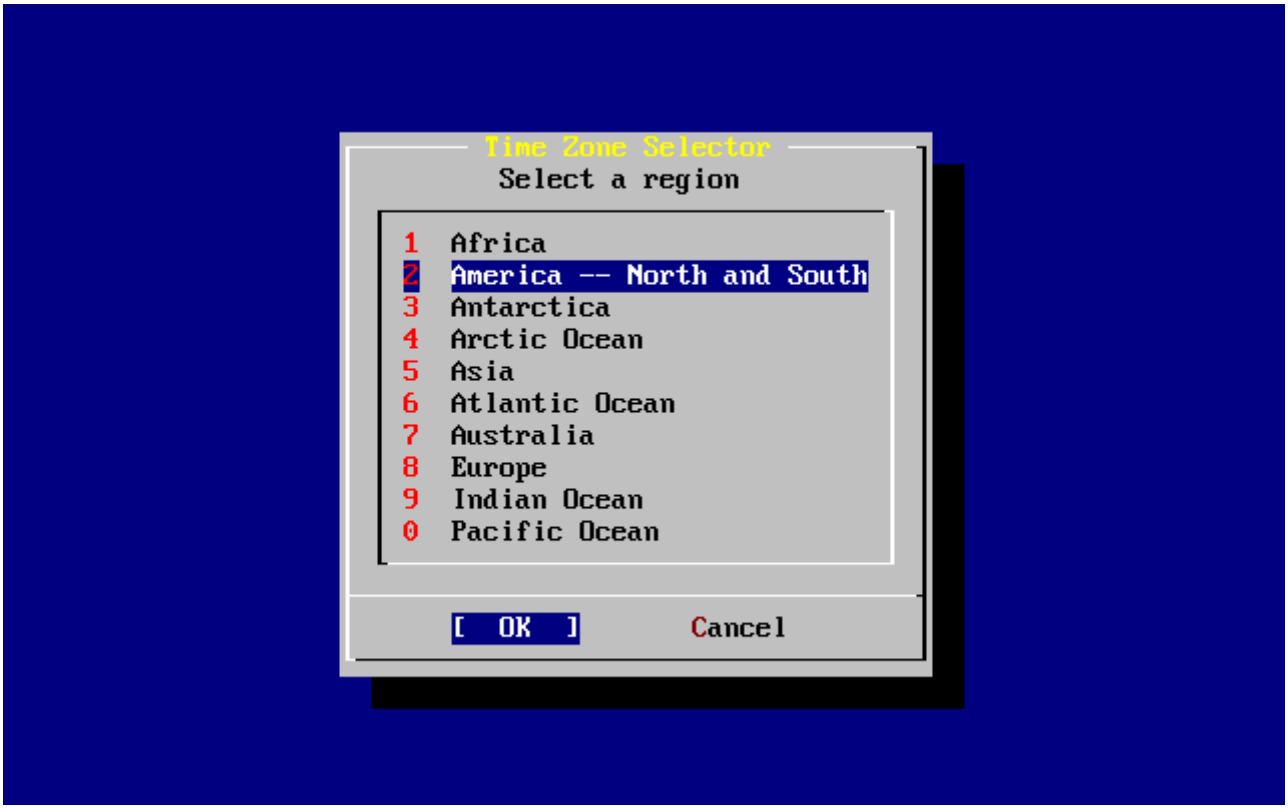
例子中假□此台机器位于美国□部的□区。 □参考□所在的地理位置来配置。

```
                    User Confirmation Requested
         Would you like to set this machine's time zone now?

                          [ Yes ]   No
```

□□ **[yes]** 并按下 `Enter` □以配置□区。

```
                      User Confirmation Requested
  Is this machine's CMOS clock set to UTC? If it is set to local time
  or you don't know, please choose NO here!

                               Yes   [ No ]
```

□里按照□机器□□的配置，□□ **[yes]** 或 **[no]** 然后按 `Enter`。

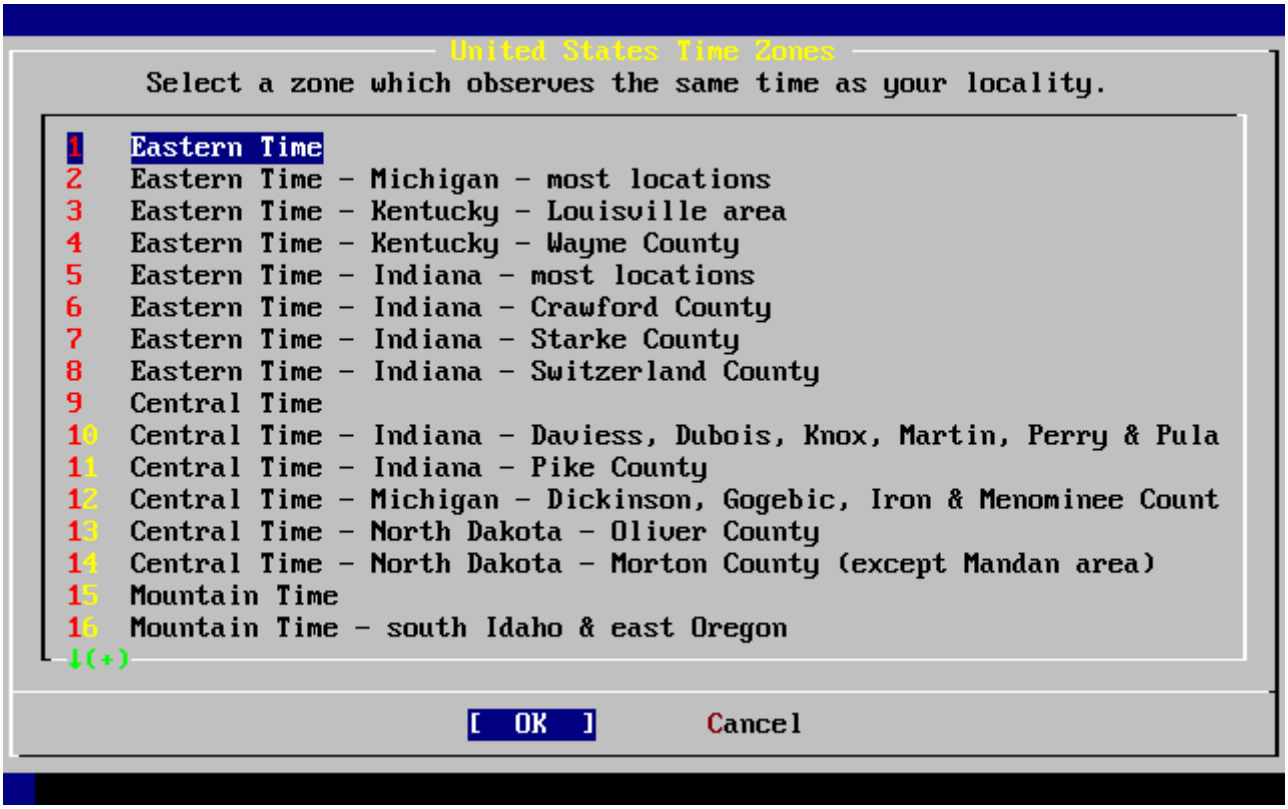


□ *38.* □□□所□的地理区域

□□□□当的区域然后按 `Enter`。

□ *39.* □□□所在的国家

□□□所在的国家然后按 `Enter`。



□ *40.* □□□所在的□区

□□□所在的□区然后按 `Enter`。

```
                    Confirmation
        Does the abbreviation 'EDT' look reasonable?

                    [ Yes ]   No
```

□□一下□区的□写是否正□，如果没□，□按 `Enter` 返回系□安装后的配置画面。

2.10.9. Linux 兼容性

□□内容只□用于 FreeBSD 7.X 安装□程， 如果□安装的是 FreeBSD 8.X 或更高版本， 系□不会□出□个提示。

```
                User Confirmation Requested
        Would you like to enable Linux binary compatibility?

                    [ Yes ]   No
```

□□ **[yes]** 并按下`Enter` □， 将允□□在FreeBSD中□行Linux的□件。安装程序会安装一些□了跟 Linux 兼容的□件包。

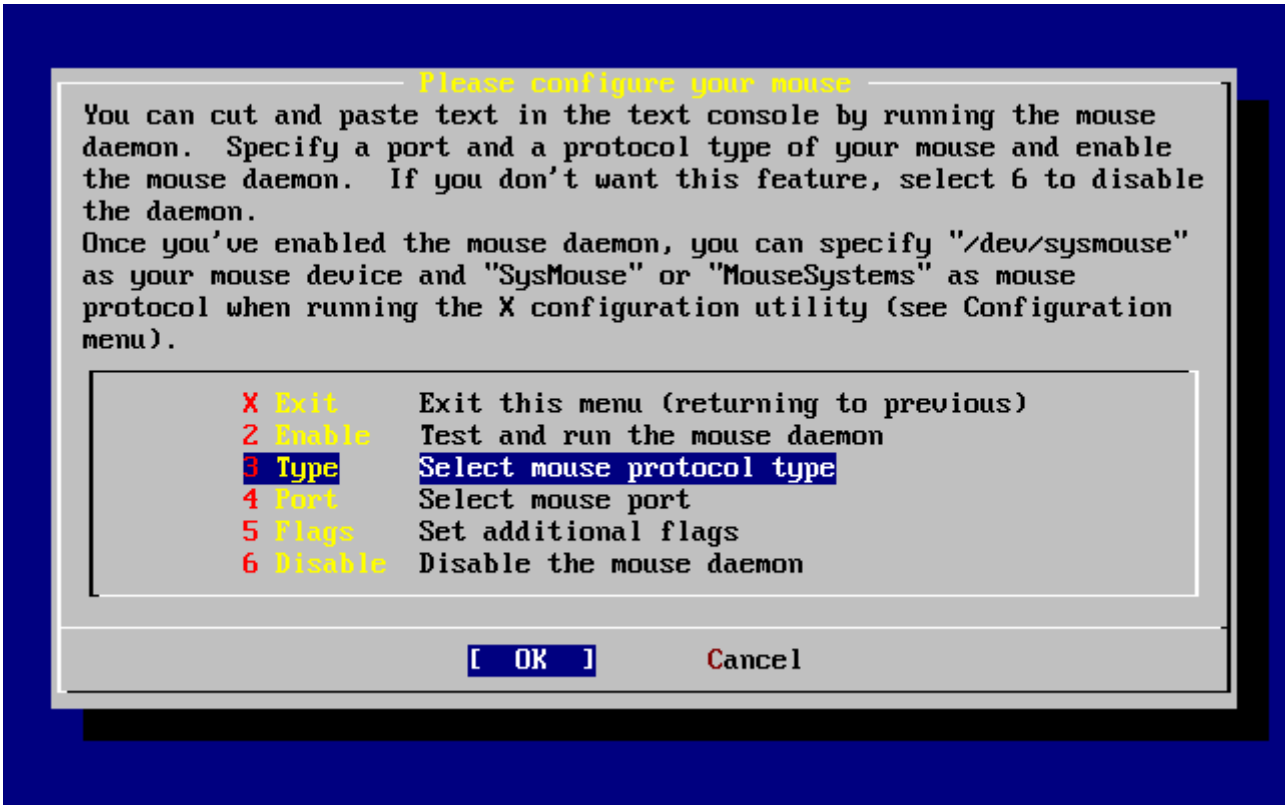
如果□是通□FTP安装，那□□必□□到网□上。 有□候FTP站并不会包含所有的安装□件包（例如Linux兼容□件包）；不□，□后□□可以再安装□个□目。

2.10.10. 配置鼠□

此□□可以□□在□端上使用三□鼠□剪□文字。 如果□用的鼠□是□个按□，□参考手册 moused(8)； 以取得有□模□三□鼠□的信息。□例中使用的鼠□不是USB接口。 （例如ps/2或com接口的鼠□）：

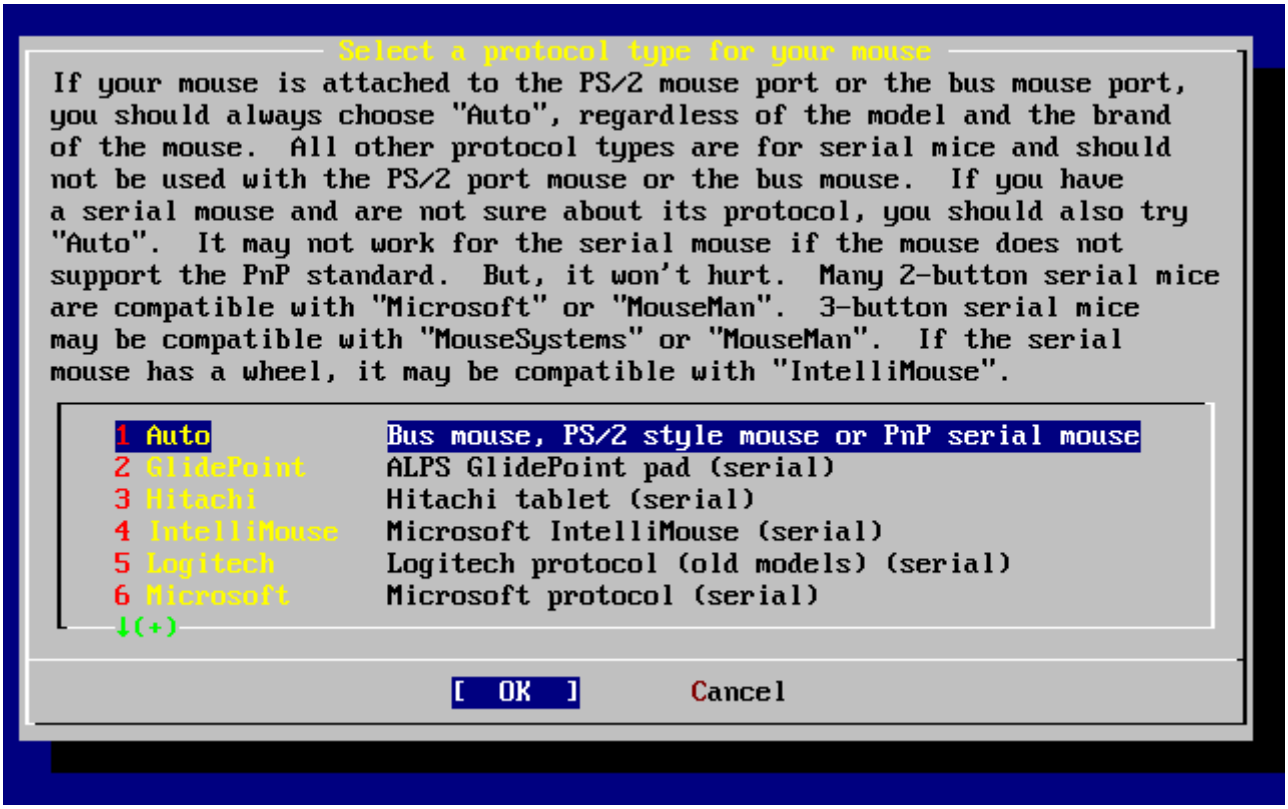
```
                    User Confirmation Requested
          Does this system have a PS/2, serial, or bus mouse?

                                [ Yes ]    No
```

如果□使用的是 PS/2、 串口或 Bus 鼠□，□□□ **[yes]**， 如果是 USB 鼠□， □□□□ **[no]** 并按 `Enter`。



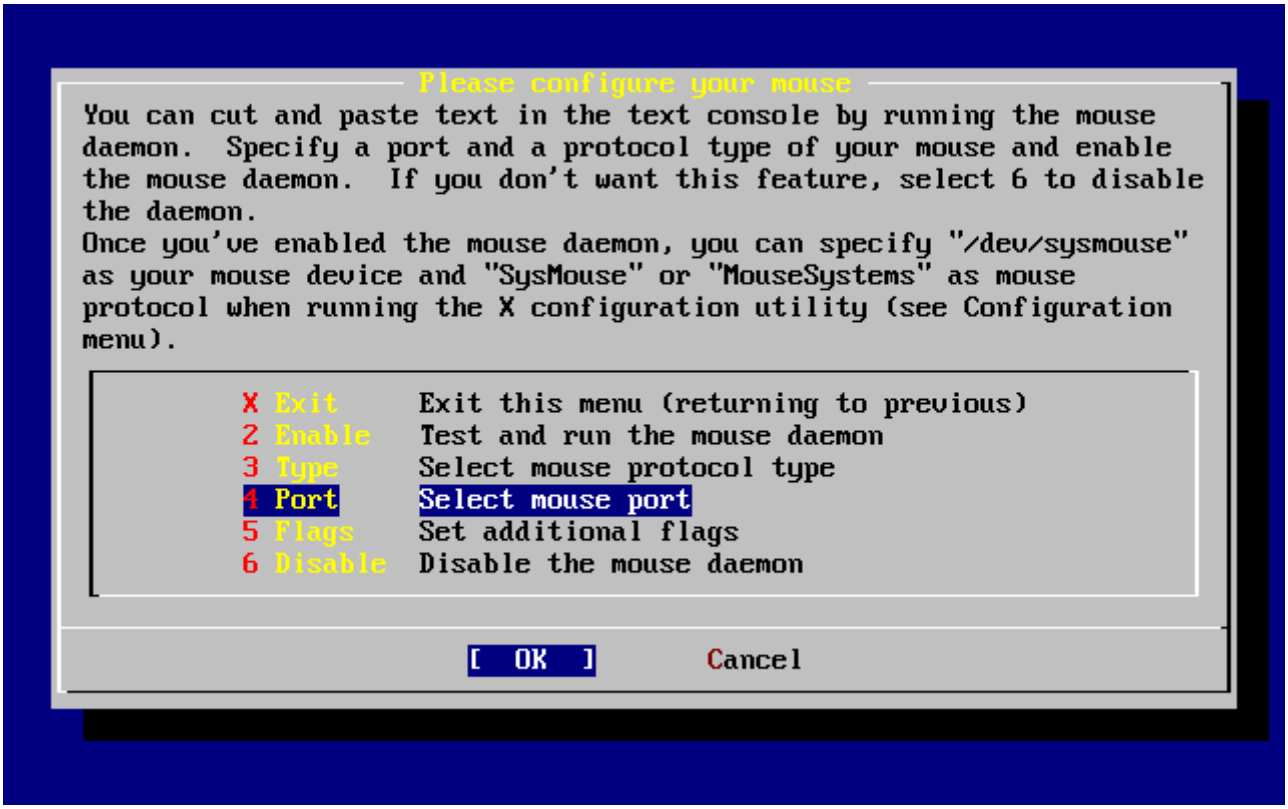
□ *41.* □□鼠□□型

使用方向□□□ Type 然后按 `Enter`。



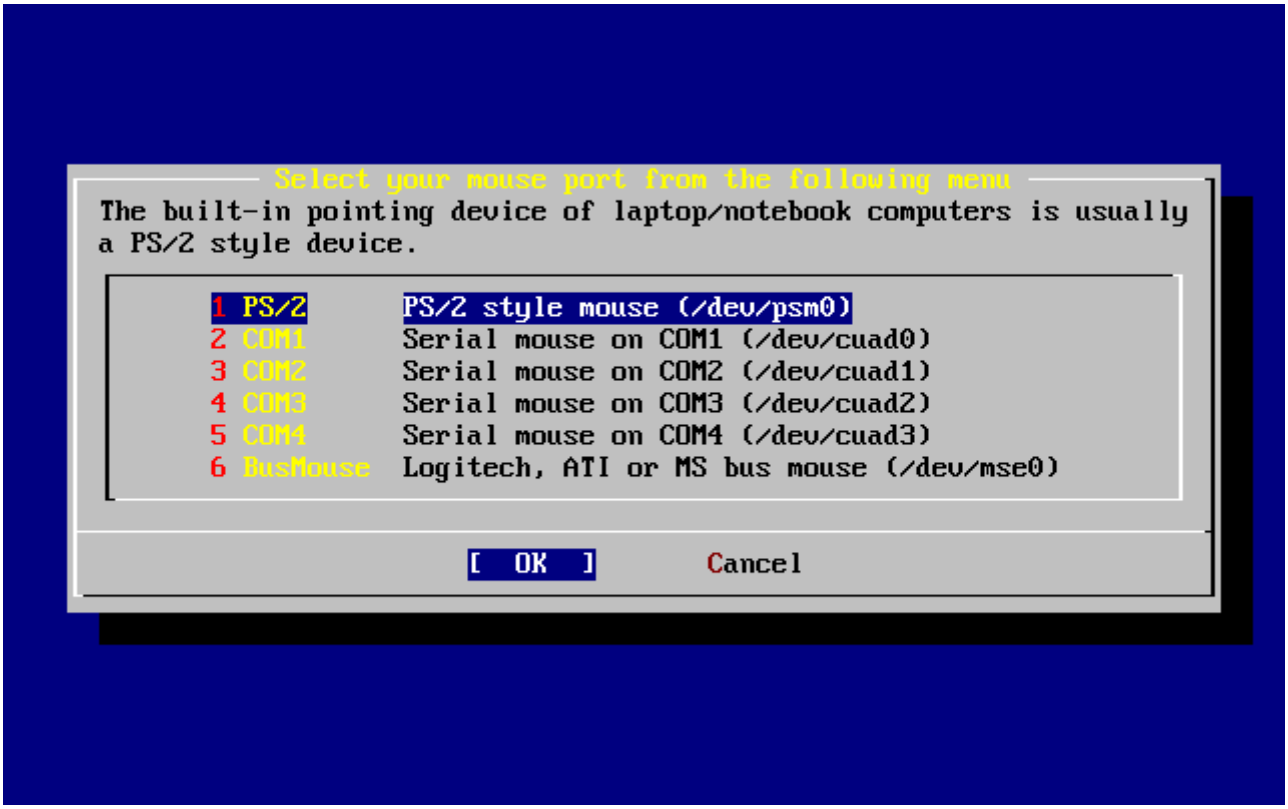
□ *42.* □置鼠□□□

在□个例子中使用的□型是ps/2鼠□，所以可以使用默□的 Auto（自□） 。 □可以用方向□□□合□的□目，□定□□了 **[OK]** 后按 `Enter` □□□此画面。



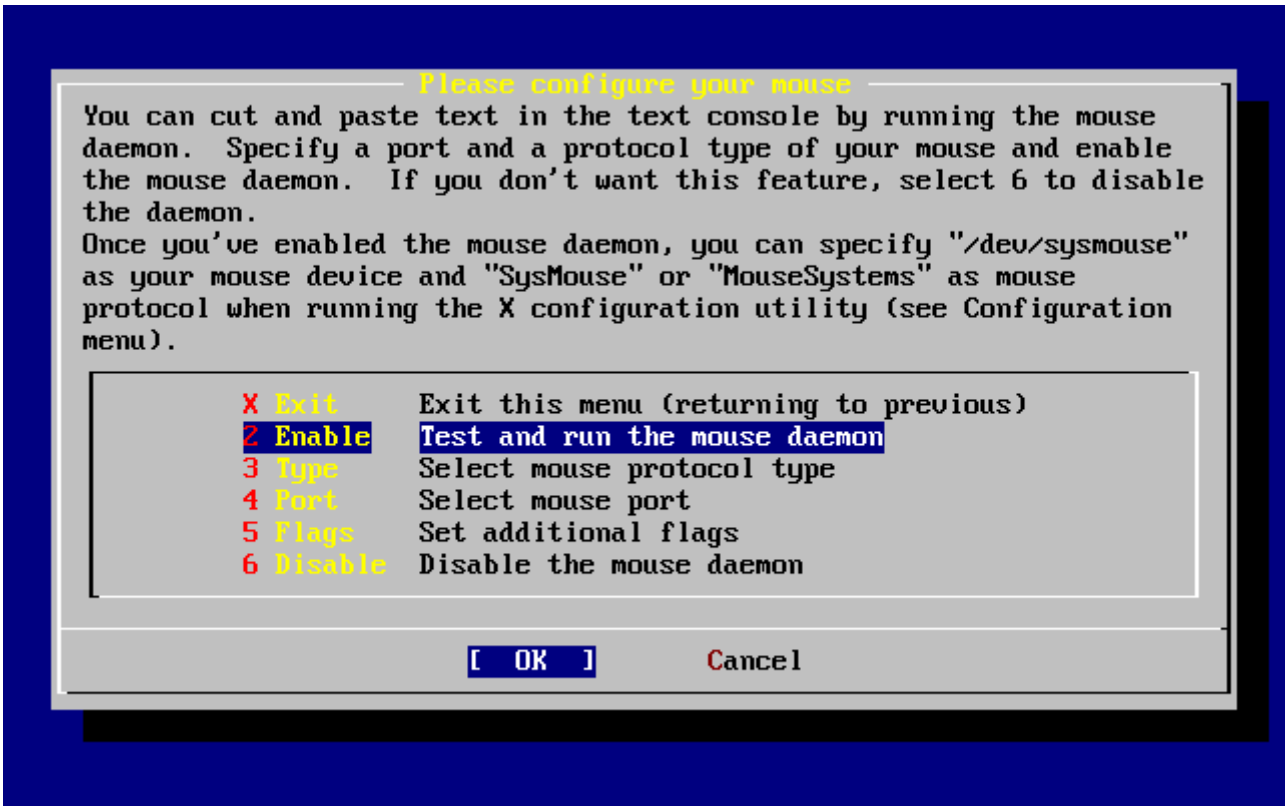
□ *43.* 配置鼠□端口

□□ Port 然后按 `Enter`。



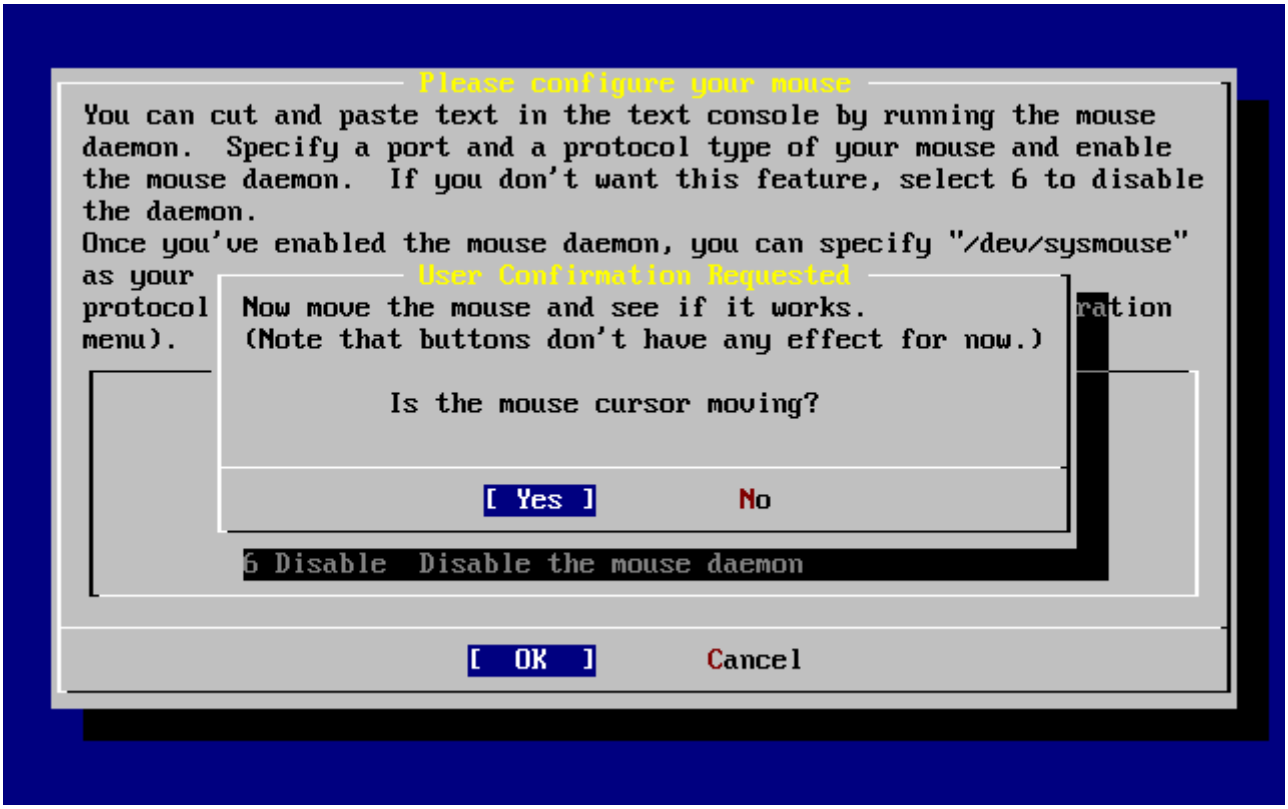
□ *44.* 配置鼠□端口

假□□台机器用的是ps/2鼠□，□可以采用默□的 PS/2 □□。□□□□当的□目然后按 `Enter`。



□ *45.* □□鼠□服□□程

□□Enable然后按 `Enter` 来□□和□□鼠□。



□ *46.* □□鼠□功能

鼠□指□可以在屏幕上移□，指明鼠□服□已□正常□用。那□□□□ **[yes]** 按 `Enter` □。否□鼠□没 有配置成功 - □□ **[no]** 并□□不同的配置 □□。

□□ Exit 并按 `Enter` 退回到系□安装完成后的配置画面。

2.10.11. 安装□□□的□件包 (package)

Package 是事先□□好的二□制文件， 因此， □是安装□件的一□便捷的方式。

在□里作□例子我□将□出安装一个 package 所需的□程。 如果需要， □可以在□一□段加入其他 package。 安装完成之后， `sysinstall` 依然可以用来安装其他 package。

```
                    User Confirmation Requested
The FreeBSD package collection is a collection of hundreds of
ready-to-run applications, from text editors to games to WEB servers
and more. Would you like to browse the collection now?

                             [ Yes ]   No
```

□□ **[yes]** 并按 `Enter` 将□入 package □□界面：

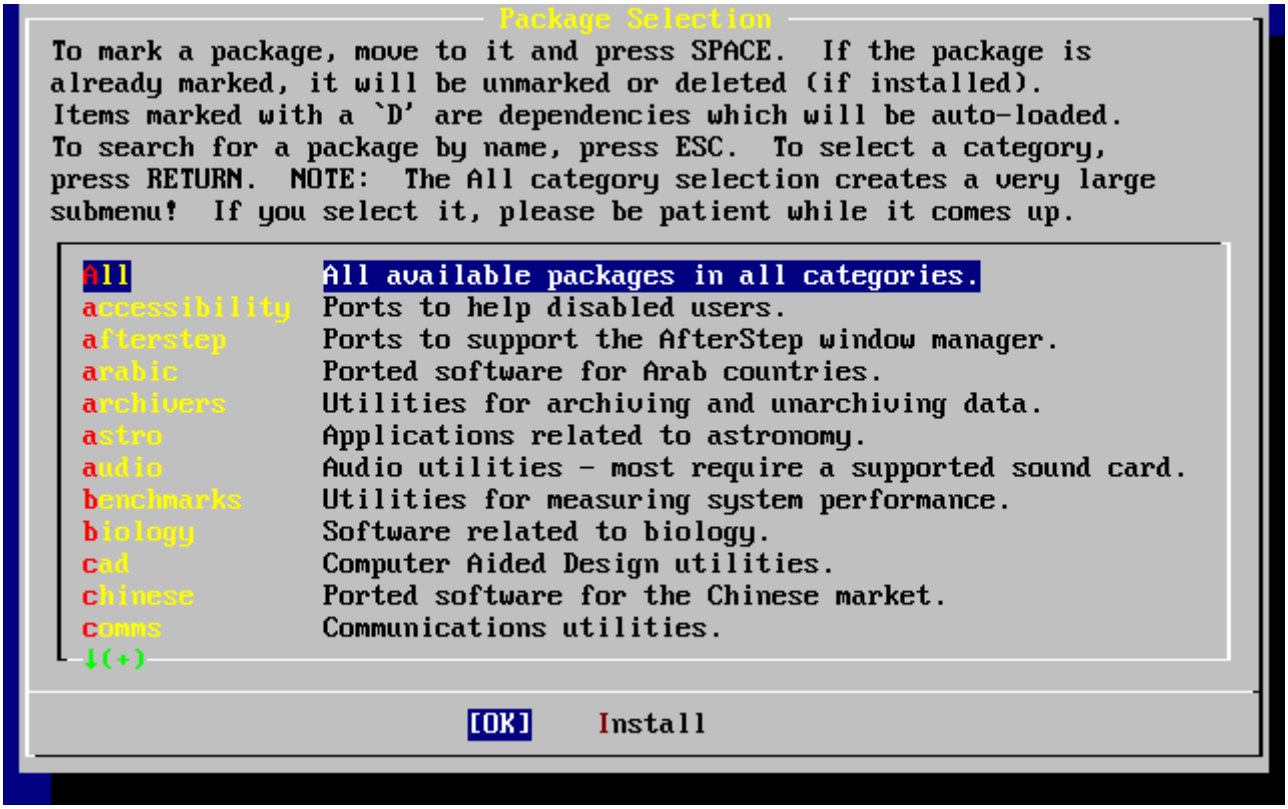


图 47. □□ *Package* □□

在任何□候， 只有当前安装介□上存在的 package 才可以安装。

如果□□了 All 或某个特定的分□， □系□会列出全部可用的 package。 用光□□移□光棒□中需要的 package，并按 `Enter`。

系□会□示可供□□的 package：

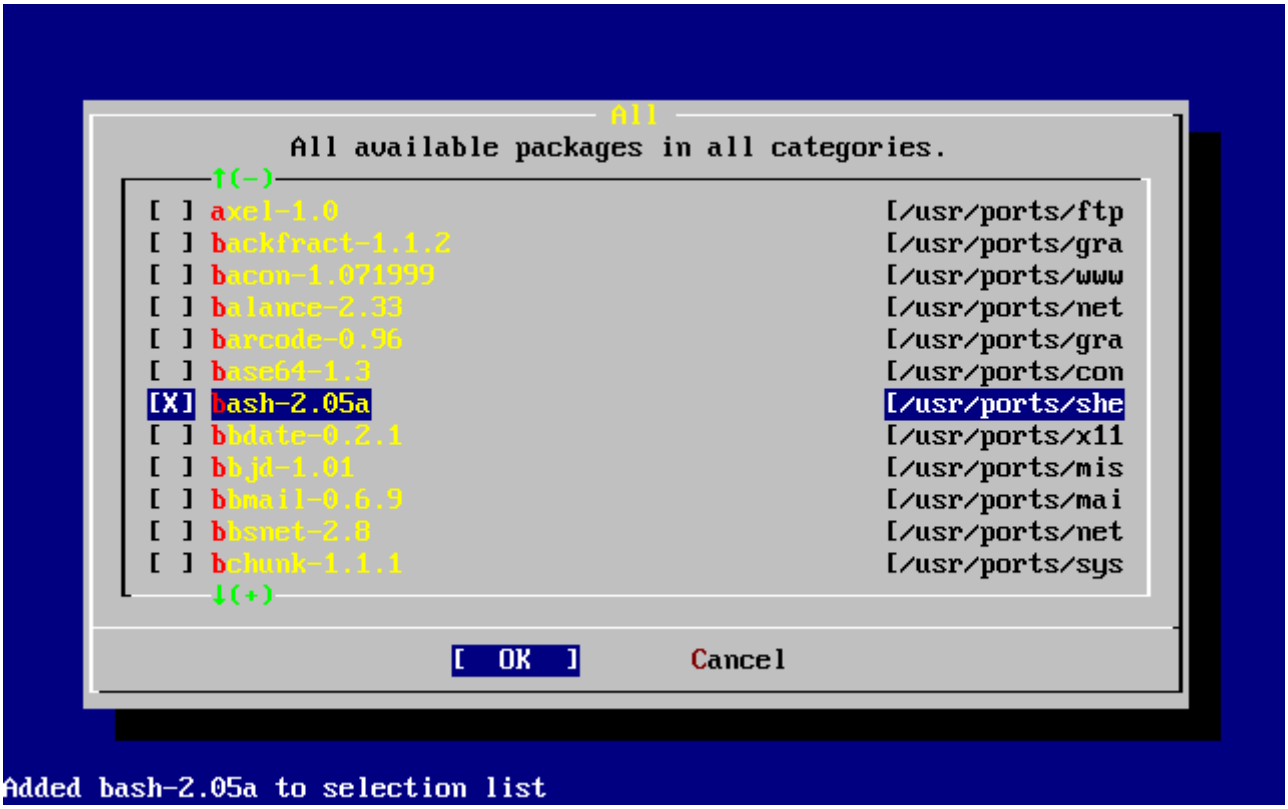


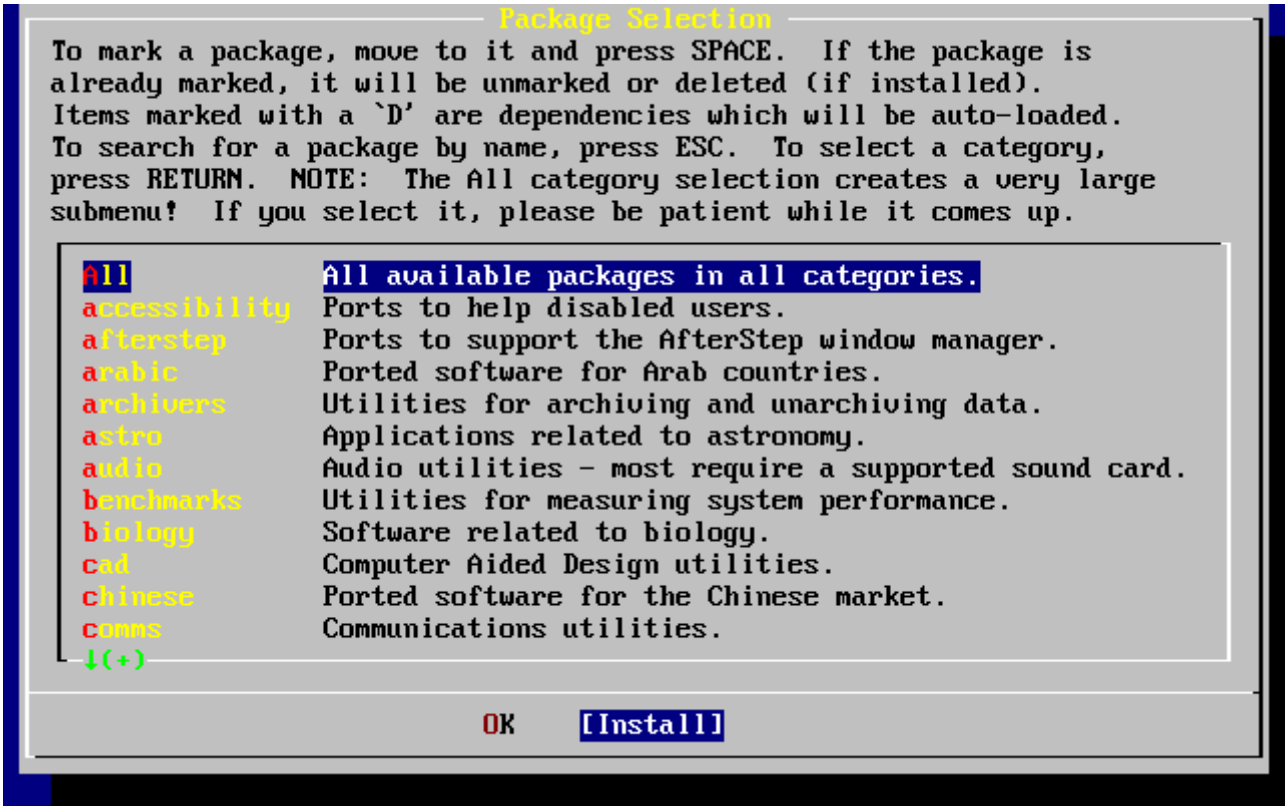
图 48. □□ *Package*

如□所示， 我□□□了 bash shell。 □可以根据需要使用 `Space` □来勾□□定的 package。 在屏幕左下角会□出 package 的□短□明。

反□按下 `Tab` □， 可以在最后□中的 package、 **[OK]** 和 **[Cancel]** 之□来回切□。

当□把需要的 package 都□□□安装之后， 按一下 `Tab` 切□到 **[OK]**， 随后按下 `Enter` 就可以回到 package □□菜□了。

左右方向□可以用于在 **[OK]** 和 **[Cancel]** 之□□行切□。 □□方法也可以用来□□ **[OK]**， 随后按下 `Enter` 也可以回到 package □□菜□。



□ *49.* 安装□□□□件包

使用 `Tab` 和左右方向□□□ **[Install]** 并按 `Enter`。 接下来需要□□将要安装的□□□包：

□ *50.* □□将要安装的□□□包

□□ **[OK]** 并按下 `Enter` 就可以□始□□□包的安装了。在□个□程中□会看到安装的相□信息， 直到安装完成□止。□留意□察是否有□□信息出□。

在完成□□□包的安装之后， 就□入了最后的配置□段。 如果□没有□□任何□□□包， 并希望直接□入最后的配置□段， □可以□□ **[Install]** 来跳□。

2.10.12. 添加用□和□

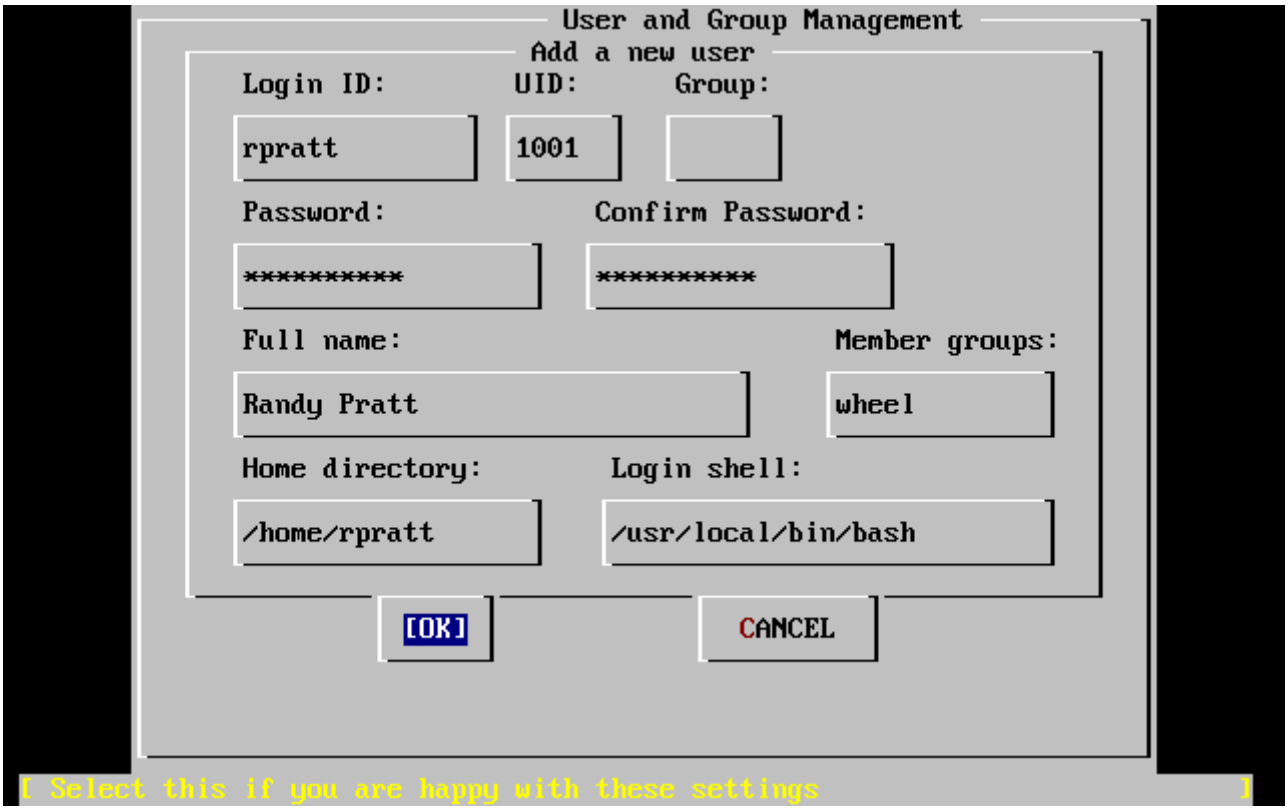
在安装系□的□程中， □□添加至少一个用□， 以避免直接以 `root` 用□的身□登□。 用以保存其用□数据的根分区通常很小， 因此用 `root` 身□□行程序可能将其迅速填□。 下面的提示信息介□了□□做可能□来的更大□患：

```
                    User Confirmation Requested
Would you like to add any initial user accounts to the system? Adding
at least one account for yourself at this stage is suggested since
working as the "root" user is dangerous (it is easy to do things which
adversely affect the entire system).

                           [ Yes ]   No
```

□□ **[yes]** 并按 `Enter` 即可□始□建用□的□程。

□ *51.* □□用□

用箭□□来□□ User 然后按 `Enter`。



□ *52.* 添加用□信息

下面的描述信息会出□在屏幕的下方，可以使用 `Tab` □来切□不同的□目，以便□入相□信息：

Login ID

新用□的登□名（□制性必□写）

UID

□个用□的ID□号（如果不写，系□自□添加）

Group

□个用□的登□□名（如果不写，系□自□添加）

Password

□个用□的密□（□入□个需要很仔□！）

Full name

用□的全名（解□、□注）

Member groups

□个用□所在的□

Home directory

用□的主目□（如果不写，系□自□添加）

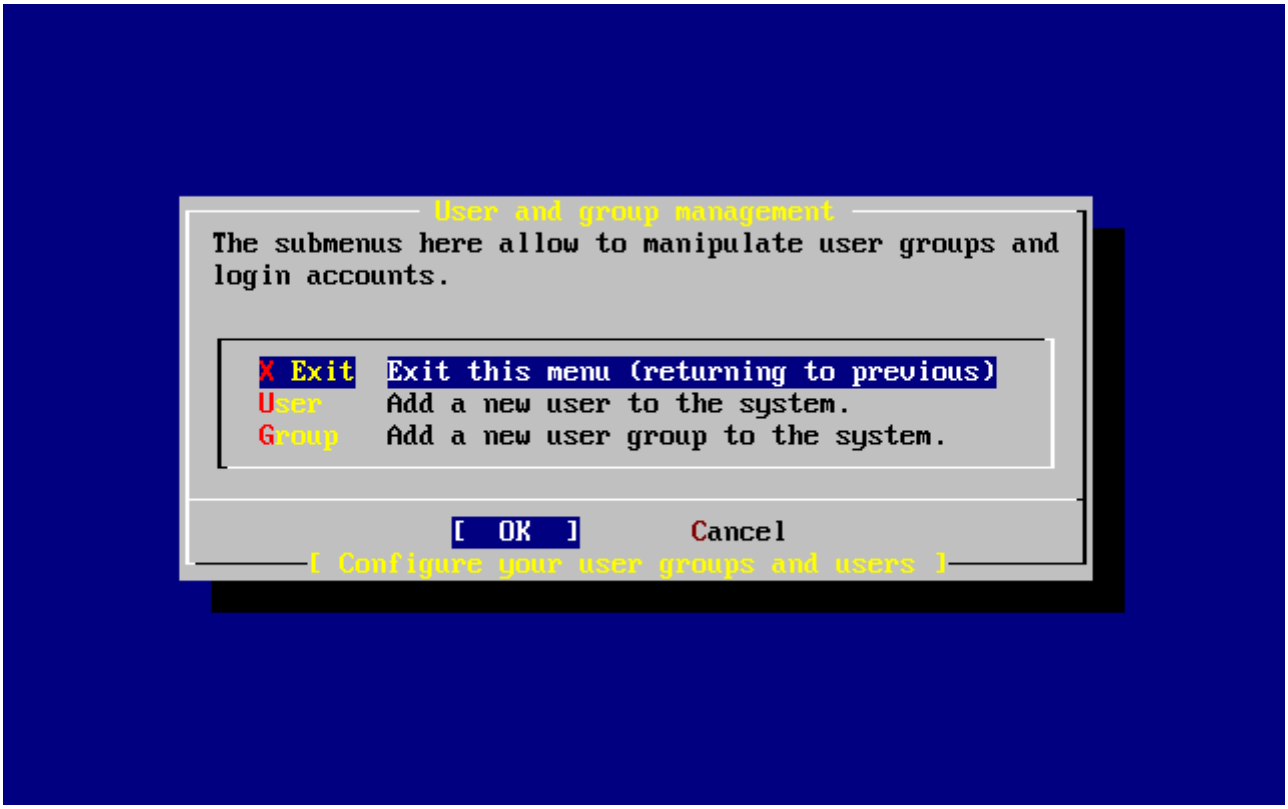
Login shell

用□登□的shell（默□是/bin/sh）。

□可以将登□ shell 由 /bin/sh 改□ /usr/local/bin/bash， 以便使用事先以 package 形式安装的 bash shell。不要使用一个不存在的或□不能登□的shell。 最通用的shell是使用 BSD-world 的 C shell， 可以通□指定/bin/tcsh来修改。

用□也可以被添加到 `wheel` □中成了一个超□用□，从而□有 `root` □限。

当□感□□意□，□入 **[OK]** □， 用□和□管理菜□将会重新出□。



□ *53.* 退出用□和□管理

如果有其他的需要， 此□□可以添加其他的□。 此外， □可以通□ `sysinstall` 在安装完成之后添加它□。

当□完成添加用□的□候，□□Exit 然后□入 `Enter` □□下面的安装。

2.10.13. □置 root 密□

```
                    Message
Now you must set the system manager's password.
This is the password you'll use to log in as "root".

                     [ OK ]

             [ Press enter or space ]
```

□入 `Enter` 来□置 root 密□。

密□必□正□地□入□次。 毋庸□言， □需要□□一个不容易忘□的口令。 □注意□□入的口令不会回□， 也不会□示星号。

```
New password:
Retype new password :
```

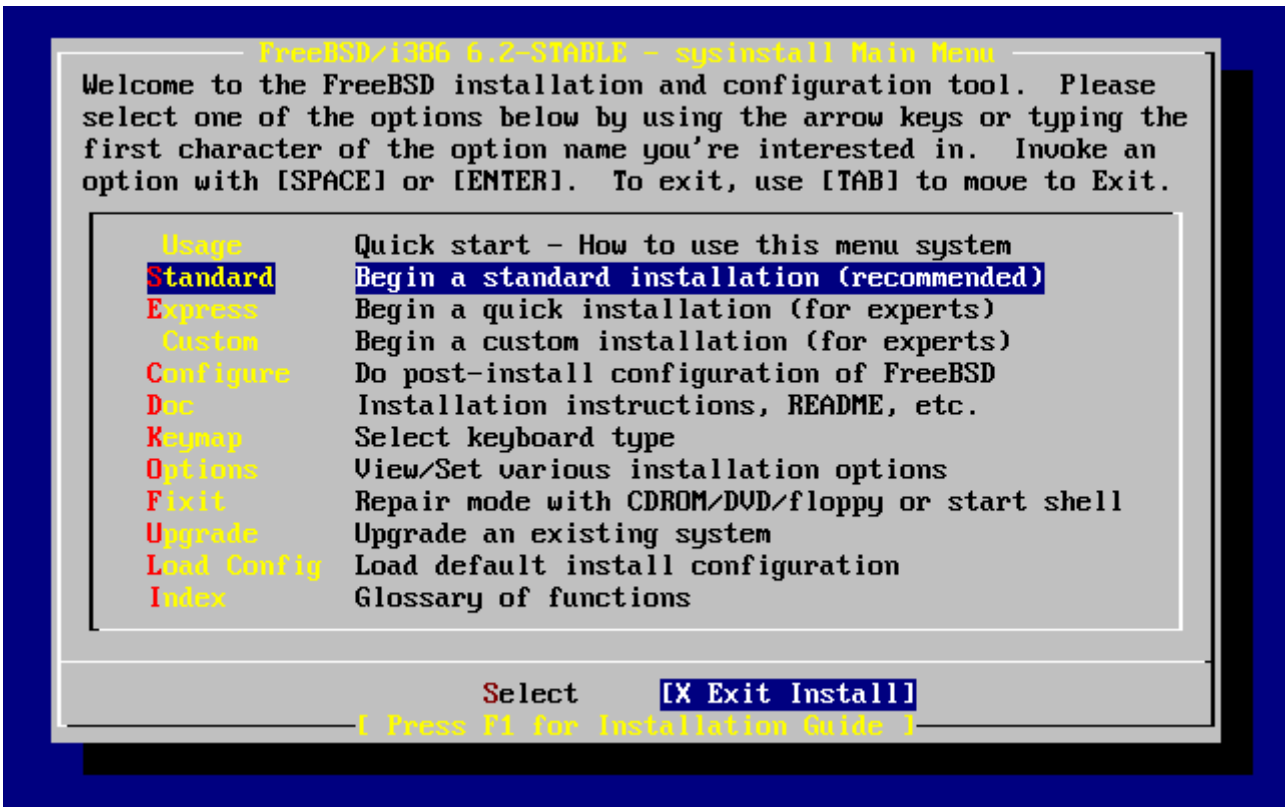
密□成功□入后，安装将□□。

2.10.14. 退出安装

如果□需要□置 其他网□□□， 或需要完成其他的配置工作， 可以在此□或者事后通□ sysinstall 来□行配置。

```
                    User Confirmation Requested
Visit the general configuration menu for a chance to set any last
options?

                              Yes    [ No ]
```

□□ **[no]** 然后□入 Enter 返回到主安装菜□。



□ *54.* 退出安装

□□ **[X Exit Install]** 然后□入 Enter。□可能需要□□是否真的退出安装：

```
                    User Confirmation Requested
Are you sure you wish to exit? The system will reboot.

                          [ Yes ]   No
```

□□ **[yes]**。 如果□是从 CDROM 引□的系□， □会出□下面的提示信息要求□取出光□：

```
                    Message
Be sure to remove the media from the drive.

                    [ OK ]
            [ Press enter or space ]
```

在系□□始重□之前， CDROM □□器是□住的。 CDROM 解□后就可以取出光□了 (□作要快)。 按 **[OK]** 重□系□。

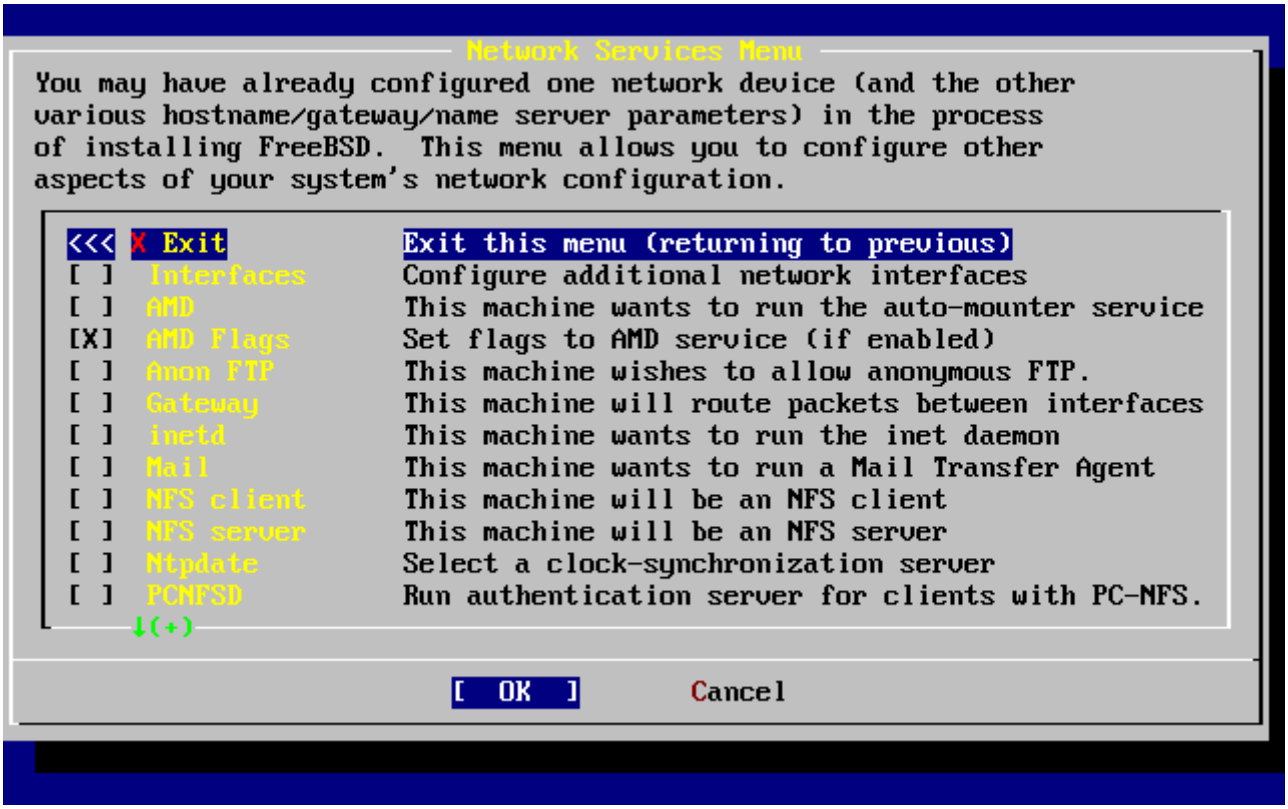
此后系□将重新□□， 因此□留意是否会出□一些□□信息。 □一□的□□， □参□ FreeBSD 的□□□程。

2.10.15. 配置其他网□服□

如果之前缺少□一□域的□□， 那□配置网□服□□于新手而言， 很可能会是一件很有挑□的事情。 网□， 包括 Internet， □于包括 FreeBSD 在内的所有□代操作系□而言都至□重要。 因此， 首先□ FreeBSD 提供的□富的网□性能加以了解会很有□助。 在安装□程中了解□些知□， 能□□保用□更好地理解他□可以用到的各□服□。

网□服□是一些可以接收来自网□上任何地方的人所提交的□入信息的程序。 人□一直都在努力□保□些程序不会做任何 "有害的" 事情。 不幸的是， 程序□□并不是十全十美的完人，因此，网□服□程序中的漏洞， 便有可能被攻□者利用来做一些坏事。因而， 只□用那些□知道自己需要的服□就很重要了。如果存在疑□， 那□就最好不要在□□□需要它之前□□任何网□服□。 □可以事后通□再次□行 sysinstall 或直接手工配置 /etc/rc.conf 来随□□用□些服□。

□□ **Networking** □□将下□示一个□似下面的菜□：



□ *55.* 网□配置之上□配置

第一个□□， Interfaces， 已□在前面的 配置网□ 中做□配置， 因此□在可以略□它。

□□ AMD □□， 将添加□于 BSD 自□挂接程序的支持。 □个程序通常会和 NFS □□ (□情参□下文)配合使用，以便自□挂□□程文件系□。 □用它不需要在此□□行特殊的□外配置。

下一行是 AMD Flags 的参数□□。 □□它之后，会□出一个□□□□ AMD 参数的子菜□。 菜□中包含一系列的□□：

```
-a /.amd_mnt -l syslog /host /etc/amd.map /net /etc/amd.map
```

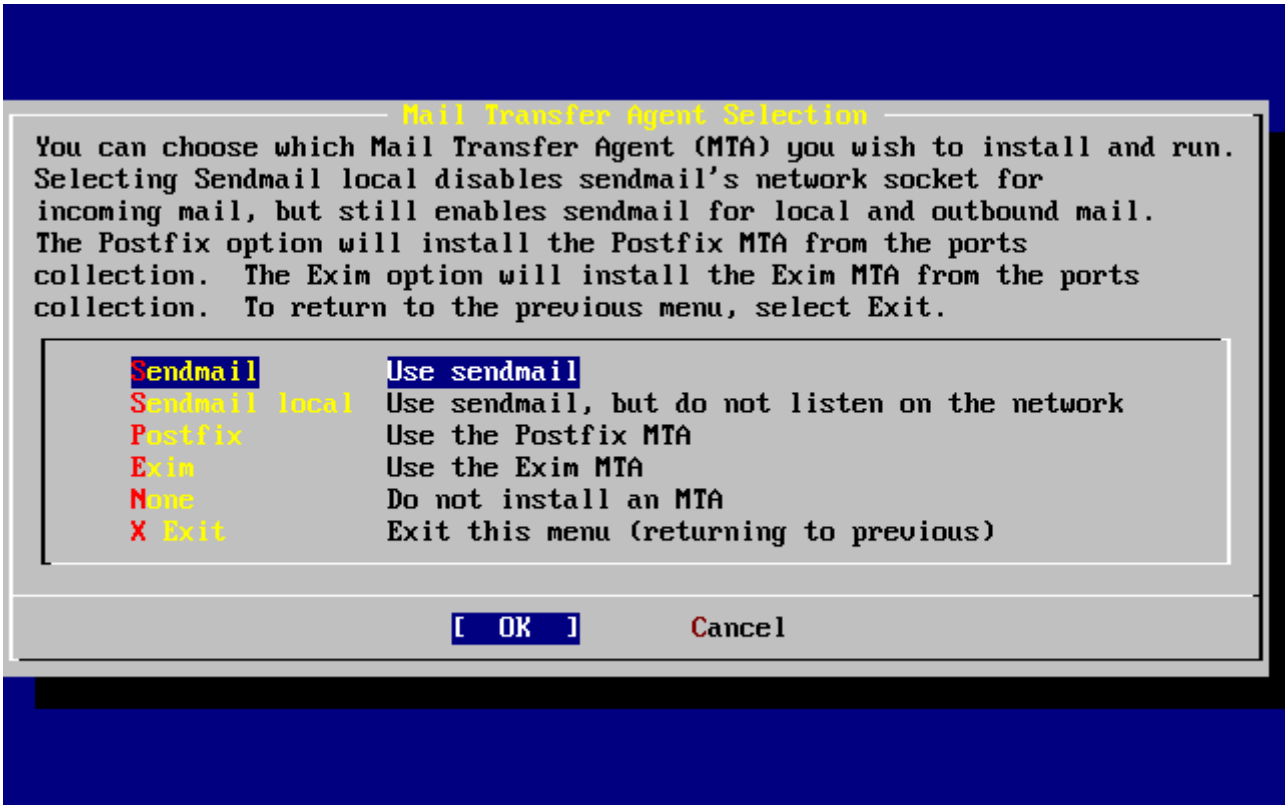
-a □□用来□置默□的挂接位置，□里使用的是 /.amd_mnt目□。-l 指定默□的 日志 文件； 但是，当使用 syslogd □， 所有在日志中□□的活□， 都会□送到系□日志服□去。 /host 用来挂接□程主机上□出的文件系□，而 /net 目□□用来挂接从特定 IP 地址□出的文件系□。 /etc/amd.map 文件定□了用于 AMD 的默□□出□□。

Anon FTP 允□匿名 FTP □□。 □中□个□□， 可以使□台机器成□一台匿名 FTP 服□器。 要注意□用□个□□的安全□□。 系□将使用□外的菜□来□明安全□□和□一□的配置。

Gateway □□可以使将本机配置成□一台以前我□介□□的网□。 如果□在安装□程中不小心□中了 Gateway，也可以在□里用□个□□来取消。

Inetd □□用来配置或完全禁用前面□□□的 inetd(8) 服□程序。

Mail 用来配置系□默□的 MTA 或□件□□代理。 □□□个□□将出□下面的菜□：



□ *56.* □□默□的 *MTA*

□里□□提供了一个安装MTA 并将其配置□默□□的机会。MTA 是一□能□将□件□□□本系□或互□网上的用□的□件服□。

□□ Sendmail 将会安装十分流行的 sendmail 服□， □也是 FreeBSD 的默□配置。Sendmail local □□表示将 sendmail □□默□的 MTA，但禁止其从 Internet 上接收□件的能力。 此外□有一些其他□□，Postfix 和 Exim 与 Sendmail 的功能□似。 它□□者也可以投□□件； 不□， 有些用□会喜□使用它□代替 sendmailMTA。

□□ MTA 或决定不挑□ MTA 之后， 网□配置菜□的下一□将是 NFS client。

NFS client 客□端可以使系□通□ NFS 与服□器□行通信。 NFS 服□器通□ NFS □□可以使其它在网□上的机器来□□自己的文件系□。 如果□台机器要作□一台独立的服□器，□个□□可以保留不□。 如果□用它， □在之后□需要□行更多的其他配置； □参□ 网□文件系□（NFS) 以了解□于配置客□机和服□器的□一□□情。

接下来的 NFS server □□， 可以□□将本机系□配置□ NFS 服□器。 □会自□将□□ RPC □程□程□用的信息写入配置文件。 RPC 是一□在多个主机和程序之□□行□接□□的机制。

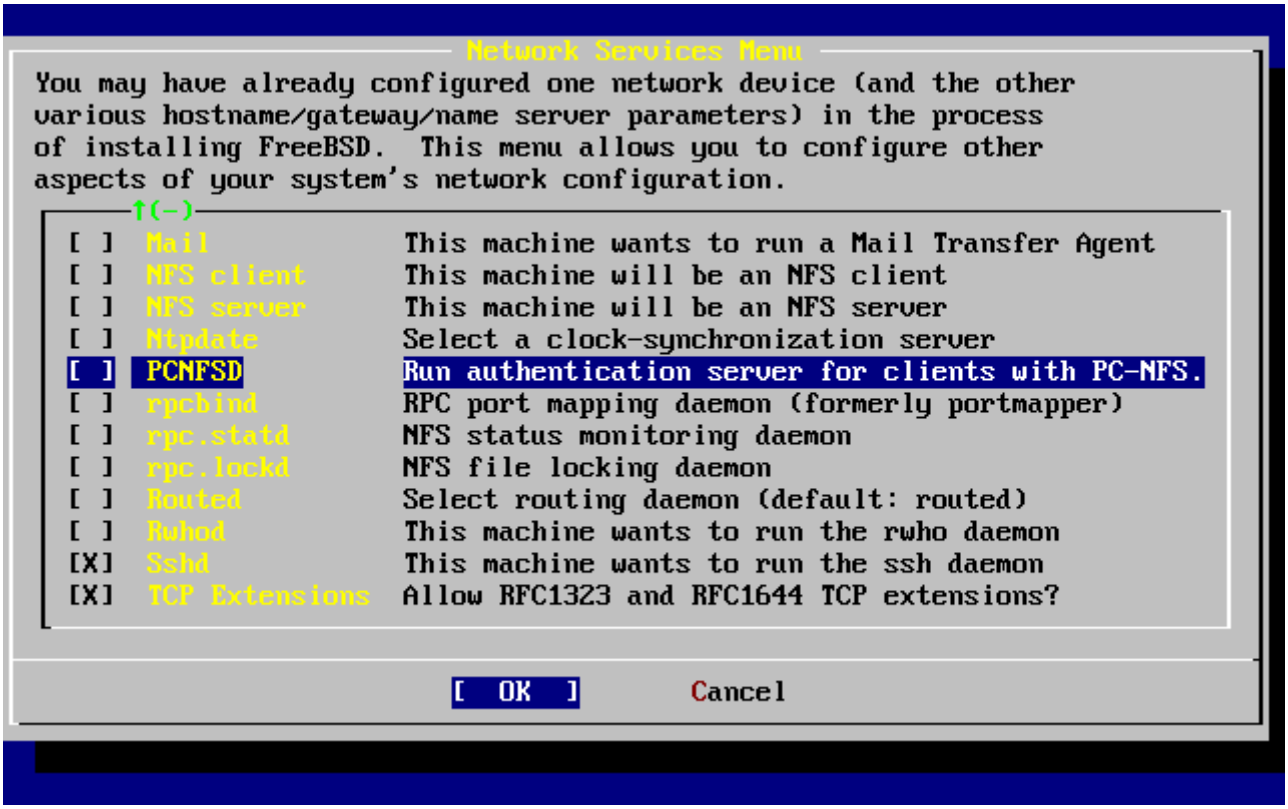
下一□是 Ntpdate □□， 它能□□理□□同□。 当□□它后， 会出□一个像下面所似的菜□：

□ *57. Ntpdate* 配置

从□个菜□□□一个□□最近的服□器。 □□□近的服□器，有助于提高□□同□的精度， 因□□□的服□器的□接延□可能会比□大。

下一个□□是 PCNFSD。 □个□□将安装第三方□件包 net/pcnfsd。 它可以用来□无法自行提供 NFS □□服□的操作系□， 如微□的 MS-DOS® 提供服□。

□屏到下一□看一下其它□□：



□ *58.* 网□配置之下□配置

rpcbind(8)， rpc.statd(8) 和 rpc.lockd(8) □三个程序是用来提供□程□程□用 (RPC) 服□的。 rpcbind 程序管理 NFS 服□器和客□端的通信， □是 NFS 正□工作的必要前提。rpc.statd 程序可以和其它主机上 rpc.statd 程序交互， 以提供状□□控。□些状□□告默□情况下会保存到 /var/db/statd.status 文件中。最后的一□是 rpc.lockd □□， 如果□用，□将提供文件上□服□。通常将它和 rpc.statd □用， 以□□□些主机会□求□文件□行上□操作， 以及□□操作的□繁程度。 尽管后□□功能□于□□非常有用， 但它□并不是 NFS 服□器和客□端正常□行所必需的。

下一个□目是Routed，□是一个路由程序。 routed(8) 程序管理网□路由表，□□多播路由， 并且支持在网□上与它物理相□的主机来□制它的路由表的□求。 它被广泛地□用在本地网□中并扮演着网□的角色。 当□□它后，一个子菜□会来□□□□个程序的默□位置。 默□的位置已□被定□□， □可以□□ Enter □，也可以按下其它的□。 □□会出来□一个菜□来□□□□□□ routed程序的参数。 默□的是 -q 参数。

接下来是 Rwhod □□， □中它会□用 rwhod(8) 程序在系□初□化的□候。 rwhod程序通□网□周期性的广播系□信息或以"客□"的身□来收集□些信息。 更多的信息可以□看 ruptime(1) 和 rwho(1) 手册□。

倒数第二个□□是sshd(8) 程序。它可以通□使用 OpenSSH 来提供安全的shell服□， 我□推□通□使用它来使用 telnet 和 FTP 服□。 sshd 服□通□使用加密技□来□建从一台机器到□一台机器的安全□接。

最后有一个 TCP □展□□。 □可以用来□展在 RFC 1323 和 RFC 1644 里定□的 TCP 功能。当□多主机以高速□接本机□，可能会引起某些□接被□□。 我□不推□使用□个□□， 但是当使用独立的主机□可以从它上面得到一些好□。

□在□已□配置完成了网□服□， □可以□□屏幕到□部□□ X Exit □， 退出□入下一个配置部分， 或□□地□□□次 X Exit 之后□□ **[X Exit Install]** 来退出 sysinstall。

2.10.16. FreeBSD 的□□□程

2.10.16.1. FreeBSD/i386 的□□□程

如果□□正常，□将看到在屏幕上有很多信息□□， 最后□会看到登□命令行。□可以通□□入 `Scroll-Lock` 和使用 `PgUp` 与 `PgDn` 来□看信息，再□入 `Scroll-Lock` 回到命令行。

□□信息可能不会□示（□冲区的限制）。□可以通□□入 `dmesg` 来□看。

使用□在安装□程中□置的用□名/密□来登□。（例子中使用 `rpratt`）。除非必□的□候□不要用 `root` 用□登□。

典型的□□信息：（忽略版本信息）

```
Copyright (c) 1992-2002 The FreeBSD Project.
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994
        The Regents of the University of California. All rights reserved.

Timecounter "i8254"  frequency 1193182 Hz
CPU: AMD-K6(tm) 3D processor (300.68-MHz 586-class CPU)
  Origin = "AuthenticAMD"  Id = 0x580  Stepping = 0
  Features=0x8001bf<FPU,VME,DE,PSE,TSC,MSR,MCE,CX8,MMX>
  AMD Features=0x80000800<SYSCALL,3DNow!>
real memory  = 268435456 (262144K bytes)
config> di sn0
config> di lnc0
config> di le0
config> di ie0
config> di fe0
config> di cs0
config> di bt0
config> di aic0
config> di aha0
config> di adv0
config> q
avail memory = 256311296 (250304K bytes)
Preloaded elf kernel "kernel" at 0xc0491000.
Preloaded userconfig_script "/boot/kernel.conf" at 0xc049109c.
md0: Malloc disk
Using $PIR table, 4 entries at 0xc00fde60
npx0: <math processor> on motherboard
npx0: INT 16 interface
pcib0: <Host to PCI bridge> on motherboard
pci0: <PCI bus> on pcib0
pcib1: <VIA 82C598MVP (Apollo MVP3) PCI-PCI (AGP) bridge> at device 1.0 on pci0
pci1: <PCI bus> on pcib1
pci1: <Matrox MGA G200 AGP graphics accelerator> at 0.0 irq 11
isab0: <VIA 82C586 PCI-ISA bridge> at device 7.0 on pci0
isa0: <ISA bus> on isab0
atapci0: <VIA 82C586 ATA33 controller> port 0xe000-0xe00f at device 7.1 on pci0
ata0: at 0x1f0 irq 14 on atapci0
ata1: at 0x170 irq 15 on atapci0
uhci0: <VIA 83C572 USB controller> port 0xe400-0xe41f irq 10 at device 7.2 on pci0
usb0: <VIA 83C572 USB controller> on uhci0
```

```
usb0: USB revision 1.0
uhub0: VIA UHCI root hub, class 9/0, rev 1.00/1.00, addr 1
uhub0: 2 ports with 2 removable, self powered
chip1: <VIA 82C586B ACPI interface> at device 7.3 on pci0
ed0: <NE2000 PCI Ethernet (RealTek 8029)> port 0xe800-0xe81f irq 9 at
device 10.0 on pci0
ed0: address 52:54:05:de:73:1b, type NE2000 (16 bit)
isa0: too many dependant configs (8)
isa0: unexpected small tag 14
fdc0: <NEC 72065B or clone> at port 0x3f0-0x3f5,0x3f7 irq 6 drq 2 on isa0
fdc0: FIFO enabled, 8 bytes threshold
fd0: <1440-KB 3.5" drive> on fdc0 drive 0
atkbdc0: <keyboard controller (i8042)> at port 0x60-0x64 on isa0
atkbd0: <AT Keyboard> flags 0x1 irq 1 on atkbdc0
kbd0 at atkbd0
psm0: <PS/2 Mouse> irq 12 on atkbdc0
psm0: model Generic PS/2 mouse, device ID 0
vga0: <Generic ISA VGA> at port 0x3c0-0x3df iomem 0xa0000-0xbffff on isa0
sc0: <System console> at flags 0x1 on isa0
sc0: VGA <16 virtual consoles, flags=0x300>
sio0 at port 0x3f8-0x3ff irq 4 flags 0x10 on isa0
sio0: type 16550A
sio1 at port 0x2f8-0x2ff irq 3 on isa0
sio1: type 16550A
ppc0: <Parallel port> at port 0x378-0x37f irq 7 on isa0
ppc0: SMC-like chipset (ECP/EPP/PS2/NIBBLE) in COMPATIBLE mode
ppc0: FIFO with 16/16/15 bytes threshold
ppbus0: IEEE1284 device found /NIBBLE
Probing for PnP devices on ppbus0:
plip0: <PLIP network interface> on ppbus0
lpt0: <Printer> on ppbus0
lpt0: Interrupt-driven port
ppi0: <Parallel I/O> on ppbus0
ad0: 8063MB <IBM-DHEA-38451> [16383/16/63] at ata0-master using UDMA33
ad2: 8063MB <IBM-DHEA-38451> [16383/16/63] at ata1-master using UDMA33
acd0: CDROM <DELTA OTC-H101/ST3 F/W by OIPD> at ata0-slave using PIO4
Mounting root from ufs:/dev/ad0s1a
swapon: adding /dev/ad0s1b as swap device
Automatic boot in progress...
/dev/ad0s1a: FILESYSTEM CLEAN; SKIPPING CHECKS
/dev/ad0s1a: clean, 48752 free (552 frags, 6025 blocks, 0.9% fragmentation)
/dev/ad0s1f: FILESYSTEM CLEAN; SKIPPING CHECKS
/dev/ad0s1f: clean, 128997 free (21 frags, 16122 blocks, 0.0% fragmentation)
/dev/ad0s1g: FILESYSTEM CLEAN; SKIPPING CHECKS
/dev/ad0s1g: clean, 3036299 free (43175 frags, 374073 blocks, 1.3% fragmentation)
/dev/ad0s1e: filesystem CLEAN; SKIPPING CHECKS
/dev/ad0s1e: clean, 128193 free (17 frags, 16022 blocks, 0.0% fragmentation)
Doing initial network setup: hostname.
ed0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet 192.168.0.1 netmask 0xffffff00 broadcast 192.168.0.255
        inet6 fe80::5054::5ff::fede:731b%ed0 prefixlen 64 tentative scopeid 0x1
```

```
        ether 52:54:05:de:73:1b
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x8
        inet6 ::1 prefixlen 128
        inet 127.0.0.1 netmask 0xff000000
Additional routing options: IP gateway=YES TCP keepalive=YES
routing daemons:.
additional daemons: syslogd.
Doing additional network setup:.
Starting final network daemons: creating ssh RSA host key
Generating public/private rsa1 key pair.
Your identification has been saved in /etc/ssh/ssh_host_key.
Your public key has been saved in /etc/ssh/ssh_host_key.pub.
The key fingerprint is:
cd:76:89:16:69:0e:d0:6e:f8:66:d0:07:26:3c:7e:2d root@k6-2.example.com
 creating ssh DSA host key
Generating public/private dsa key pair.
Your identification has been saved in /etc/ssh/ssh_host_dsa_key.
Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub.
The key fingerprint is:
f9:a1:a9:47:c4:ad:f9:8d:52:b8:b8:ff:8c:ad:2d:e6 root@k6-2.example.com.
setting ELF ldconfig path: /usr/lib /usr/lib/compat /usr/X11R6/lib
/usr/local/lib
a.out ldconfig path: /usr/lib/aout /usr/lib/compat/aout /usr/X11R6/lib/aout
starting standard daemons: inetd cron sshd usbd sendmail.
Initial rc.i386 initialization:.
rc.i386 configuring syscons: blank_time screensaver moused.
Additional ABI support: linux.
Local package initialization:.
Additional TCP options:.

FreeBSD/i386 (k6-2.example.com) (ttyv0)

login: rpratt
Password:
```

生成 RSA 和 DSA密□在比□慢的机器上可能要花很□□□。□只是一个 新安装后的首次□□，以后的□□会□得更快一点。

如果已□完成 X 服□器的配置， 且指定了默□的□面□口管理器， 就可以在命令行□入 `startx` 来□□它了。

2.10.17. FreeBSD □机

正□的□□操作系□是很重要的。不要□□□□□源。 首先，□需要成□一个超□用□，通□□入 `su` 命令来□□。然后□入 `root` 密□。□需要用□是 `wheel` □的一名成□。然后， 以root□入 `shutdown -h now`命令。

```
The operating system has halted.
Please press any key to reboot.
```

当shutdown命令□出后，屏幕上出□ "Please press any key to reboot" 信息□，□就可以安全的□□□算机了。如果按下任意一个□， □算机将重新□□。

□也能□使用 `Ctrl` + `Alt` + `Del` □合□来重新□□□算机，但是不推□使用□个操作。

2.11. 常□□□

下面将介□一些在安装□程中常□的□□，像如何□告□生的□□， 如何双重□□ FreeBSD 和 MS-DOS® 或 Windows®。

2.11.1. 当□遇到□□□，□□□□做？

由于 PC □□的限制， 硬件□□不可能 100% 地可□， 但是有些□□是□可以自己解决的。

首先□□一下□使用的 FreeBSD 版本的 硬件兼容□明 文□看看□使用的是否是被支持的硬件。

如果□使用的硬件是系□支持的，但仍然遇到了死机或其他□□， □需要□□ 定制的内核。 □能□支持默□的 GENERIC 内核所不支持的□□。 在引□□上的内核假定□大多数的硬件，均□按出厂□置的方式配置了 IRQ、 IO 地址和 DMA 通道。 如果□的硬件重新□行了配置， □可能需要□□内核配置， 并重新□□内核， 以便告□ FreeBSD 到□里去□□□□。

除此之外，也可能遇到□□情况□，即探□某□并不存在的□□□， 会干□到其他□□的□□并使其失□。 □□情况□下□禁止□□程序□□可能□致冲突的□□。

有些安装□□可以借助更新硬件的程序来解决，特□是主板的 BIOS 。 大部分的主板制造商都会提供网站□用□下□新的 BIOS 以及提供如何更新的□明。

也有□多制造商□烈建□，除非必要否□不要□易更新 BIOS 。因□更新的□程 可能 会□生□□，□而□害 BIOS 芯片。

2.11.2. 使用 MS-DOS® 和 Windows® 文件系□

目前， FreeBSD 尚不支持通□ Double Space™ 程序□□的文件系□。 因此，如果希望 FreeBSD □□数据，□□首先解□□□些文件系□。 □□工作，可以通□位于 Start> Programs > System Tools 菜□的 Compression Agent 来完成。

FreeBSD 可以支持基于 MS-DOS® 的文件系□ （有□被称□ FAT 文件系□）。 mount_msdosfs(8) 命令能□把□□的文件系□挂接到□有的目□□□中， 并允□□□ FAT 文件系□上的内容。 通常我□并不直接使用 mount_msdosfs(8) 程序，它一般会在 /etc/fstab 中的某一行被□用或者被 mount(8) 工具并配合□当的参数来□用。

/etc/fstab中一个典型的例子：

```
/dev/ad0sN  /dos  msdosfs rw  0 0
```

/dos 目□必□事先存在。 更多□于 /etc/fstab 的□□， □参□ fstab(5)。

一个使用 mount(8) 挂□ MS-DOS® 文件系□的例子：

```
# mount -t msdosfs /dev/ad0s1 /mnt
```

在此例子中， MS-DOS® 文件系□位于主硬□的第一个分区。 □的情况可能与引不同，□看命令 dmesg 和 mount 的□出。 它□□□可以□□得到足□的分区信息。

FreeBSD 可能使用和其他操作系□不同□数方法来□□磁□ slices， 特□需要指出的是， MS-DOS® 的□展分区通常会比 MS-DOS® 主分区被□□□更高的数□。 可以使用 fdisk(8) 工具来□助□定□些 slices 属于 FreeBSD □些是属于其他的操作系□。

NTFS 分区也可以通□□似 mount_ntfs(8) 命令挂接在FreeBSD上。

2.11.3. 排除故障□的常□□□和解决方法

2.11.3.1. 我的系□在引□到探□硬件□□生了死机、 安装□程中行□□常， 或没有□□到□□。

FreeBSD 在□□□程中广泛使用了 i386、 amd64 及 ia64 平台提供的 ACPI 服□来□□系□配置。 不幸的是， 在 ACPI □□和主板 BIOS 中存在一些 bug。 如果遇到□□情况， 可以在系□引□□禁用 ACPI， 其方法是在第三□段引□加□器□使用 hint hint.acpi.0.disabled：

```
set hint.acpi.0.disabled="1"
```

□一□置会在系□重□之后失效，因此，如果需要的□，□□在 /boot/loader.conf 文件中□加 hint.acpi.0.disabled="1"。 □于引□加□器的□一□□情， □参□ 概述。

2.11.3.2. 在硬□安装 **FreeBSD** 之后的首次□□□， 内核加□并□□了硬件， 但□出下列消息并停止□行：

系□在□理引□□非系□中的第一□□□有一个由来已久的□□。 BIOS 采用的□号方式有□和 FreeBSD 不一致， 而□法将其□□一□□很□正□地□□。

因而， 在□生□□情况□，FreeBSD 可能会需要一些□助才能□到磁□。有□□常□的情况， 在□些情况下□都需要手工告□ FreeBSD 根文件系□模□的位置。 □是通□告□引□加□器 BIOS 磁□□号、磁□□型以及 FreeBSD 中的□□磁□的□号来□□的。

第一□情况是有□□ IDE 硬□， 分□配置□□□ IDE □□上的主 (master) □□， 并希望 FreeBSD 从第二□硬□上□□。 BIOS 将□□硬□□□□磁□ 0 和磁□ 1， 而 FreeBSD □将其分□叫做 ad0 和 ad2。

FreeBSD 位于 BIOS 磁□ 1， 其□型是 ad 而 FreeBSD 磁□□号□是 2， 因此， □□□入：

```
1:ad(2,a)kernel
```

注意， 如果□的主□□上有从□□， □□一配置是不必要的 (因□□□配置是□的)。

第二□情况是从 SCSI 磁□□□，但系□中安装了一个或多个 IDE 硬□。□□，FreeBSD 磁□□号会比 BIOS 磁□□号小。如果□有□□ IDE 硬□， 以及一□ SCSI 硬□，□ SCSI 硬□将会是 BIOS 磁□ 2， □型□ da 而 FreeBSD 磁□□号是 0， 因此， □□□入：

```
2:da(0,a)kernel
```

来告□ FreeBSD □希望从 BIOS 磁□ 2 引□， 而它是系□中的第一□ SCSI 硬□。 假如只有一□ IDE 硬□， □□以 1: 代替。

一旦□□定了□□用的正□配置， 就可以用□准的文本□□器把它写到 /boot.config 文件中了。 除非□行指定，FreeBSD 将使用□个文件的内容， 作□□ boot: 提示的默□回□。

2.11.3.3. 在硬□安装 **FreeBSD** 之后的首次□□□， **Boot Manager** 只是□出了 **F?** 的菜□提示， 但并不□□引□□程。

在□安装 FreeBSD □行到分区□□器□所□置的磁□尺寸信息不□。 □回到分区□□器并指定正□的磁□尺寸。□□情况必□重新安装 FreeBSD。

如果□无法□定在□机器上的正□尺寸信息，可以用一个小技巧： 在磁□□始的地方安装一个小的 DOS 分区，并在其后安装 FreeBSD。 安装程序能□看到□个 DOS 分区， 并利用它推□磁□的尺寸信息， □通常会有所□助。

下面的技巧不再推□使用， 在□里□供参考：

如果□正准□建立只□行 FreeBSD 的服□器或工作站， 而无需考□（之后）与 DOS、 Linux 或其他操作系□的兼容性， 也可以使用整个硬□ (分区□□器中的 A)， □□ FreeBSD 独占整个硬□□一个扇区的非□准□□。 □会□除□于磁□尺寸的一切□□， 但会限制□以后□行 FreeBSD 以外的其他操作系□的能力。

2.11.3.4. 系□□到了 ed(4) 网□， 但□是□□□超□ **(device timeout)** □□。

□的网□可能使用了与 /boot/device.hints 文件中指定的 IRQ 不同的中断□求号。 ed(4) □□默□情况下并不支持 "□" 配置 (在 DOS 中使用 EZSETUP 配置的□)， 但如果□在网□的 hints 中指定 -1， 便会使用□配置。

□□使用网□的跳□□行硬配置（根据需要修改内核□置）或通□ hint hint.ed.0.irq="-1" 将 IRQ 指定□ -1。 □会告□内核使用□配置。

□一个可能是□的网□使用 IRQ 9， □会与 IRQ 2 共用同一中断□求□， 同□也是□致□□的一个常□原因 (特□是 VGA □使用 IRQ 2 的□候！)。 □□尽量避免使用 IRQ 2 或 9。

2.11.3.5. 当在 **X11** □端中□行 **sysinstall** 的□候， 黄色的字体相□于浅灰色的背景□得□以□□。 有没有什□能□□个□用程序提供高□比度的方法？ **colorcontrast**

如果□已□安装了 X11 并且 sysinstall 在 xterm(1) 或者 rxvt(1) 中默□的□色使得文字□以□□， 可以在□的 ~.Xdefaults 中加入 XTerm*color7: #c0c0c0 □得深灰色的背景。

2.12. 高□安装指南

□□主要描述在一些特殊情况下如何安装FreeBSD。

2.12.1. 在一个没有□示器或□□的系□上安装FreeBSD

□□□型的安装叫做 "headless install（无□安装）"， 因□正要安装FreeBSD的机器不是没□□示器，就是没有□□。 □可能会□那□□安装？ 可以使用一个串行控制台。 串行控制台基本上是使用□外一台机器来充当主□示

□□和□□。 要□□做，只要□行下面的□□： □建安装 USB □□棒，□看 准□引□介□一□□明； 此外， 也可下□ ISO 映像文件， 具体□参□ □建一□安装光□。

要将安装介□改□使用串口控制台， 需要按下面□些□□来操作 (如果使用 CDROM □可跳□第一□)：

1. 令安装 USB □□棒引□并□入串口控制台

   如果使用□□制作的 USB □□棒引□系□， □ FreeBSD 会□入正常的安装模式。 我□希望引□到串口控制台来完成安装。 □了做到□一点， 需要在 FreeBSD 中使用 mount(8) 挂□ USB □。

   ```
   # mount /dev/da0a /mnt
   ```

   □需要根据□□情况修改挂点的名称。

   □在挂好了□□棒， □需要□其□行配置令其□入串口控制台。 □此， 需要在 USB □□棒中的 loader.conf 文件中加入下面的□行配置：

   ```
   # echo 'console="comconsole"' >> /mnt/boot/loader.conf
   ```

   □□就完成了□ USB □□棒的配置， □□使用 umount(8) 命令将其卸下：

   ```
   # umount /mnt
   ```

   □在就可以拔下 USB □□棒并□入□一□程的第三□了。

2. 令安装 CD 引□并□入串口控制台

   如果□直接使用 ISO 映像 (see □建一□安装光□) 制作的 CD 引□， □ FreeBSD 会引□□入正常的安装模式。 我□希望引□到串口控制台来完成安装。 □了做到□一点， □需要展□、修改并重新生成 ISO 文件， 然后再刻□光□。

   在保存例如 FreeBSD-8.1-RELEASE-i386-disc1.iso ISO 的 FreeBSD 系□上用 tar(1) 工具提取全部文件：

   ```
   # mkdir /path/to/headless-iso
   # tar -C /path/to/headless-iso -pxvf FreeBSD-8.1-RELEASE-i386-disc1.iso
   ```

   接下来需要□其□行配置令其□入串口控制台。 □此， 需要在从 ISO 映像中提取的 loader.conf 文件中加入下面的□行配置：

   ```
   # echo 'console="comconsole"' >> /path/to/headless-iso/boot/loader.conf
   ```

   最后， 从修改好的目□□中□建新的 ISO 映像。 □里我□使用通□ sysutils/cdrtools port 安装的 mkisofs(8) 工具来完成：

   ```
   # mkisofs -v -b boot/cdboot -no-emul-boot -r -J -V "Headless_install" \
           -o Headless-FreeBSD-8.1-RELEASE-i386-disc1.iso /path/to/headless-iso
   ```

□□就完成了□ ISO 映像的配置， □可以使用□熟悉的工具将其刻□到 CD-R 上了。

3. □接 Null-modem □

   □在需要一根 null-modem □ 来□接□台机器。 只要□接□台机器的串口。 □里不能使用普通的串口□，而必□使用 null-modem □， 因□它需要一些内部交叉的□□。

4. □始□□安装

   □在可以□始安装了。 将 USB □□棒□到□准□□行 headless 安装的机器上， 然后□机。 如果□使用的是 CDROM， □在□机之后立即将光□放□光□。

5. □接□的无□机器

   □在□已□通□cu(1)□接到了那台机器。

   ```
   # cu -l /dev/cuau0
   ```

   在 FreeBSD 7.X 上□使用下面的命令：

   ```
   # cu -l /dev/cuad0
   ```

□□就可以了！ □□在可以通□ cu 会□来控制那台 headless 的机器了。 接着系□会提示□□□端□型。 □□ FreeBSD 彩色控制台并□□安装！

2.13. 准□□自己的安装介□

□了避免重□ "FreeBSD disc" 在□里指 FreeBSD CDROM or DVD 那即意味着□要□□或自己制做。

有好几个原因需要□□建自己的FreeBSD安装介□。 □可能是物理介□，如磁□，使用 sysinstall 程序□到的安装文件， FTP 站点或 MS-DOS®分区。

例如：

- □有□多机器□接到本地网□，使用一个FreeBSD光□。 □要使用FreeBSD来□建一个本地FTP站点，然后使用□个FTP站点来代替□接到Internet。
- □有一□ FreeBSD 光□， FreeBSD 不支持□的 CD/DVD □□器， 但 MS-DOS®/Windows® 支持。 □要□制安装文件到一个DOS分区， 然后使用□些文件□行安装。
- □要安装的□算机没有 CD/DVD□□器和网□，但□可以□接一个 "Laplink-style" 串口或并口□□到那台□算机。
- □要通□一个磁□机来安装FreeBSD.

2.13.1. □建一□安装光□

FreeBSD 的□个□行版本都□□一支持的平台提供至少□□ CDROM 映像 ("ISO images")。如果□有刻□机，

□些映像文件可以被("burned")　成FreeBSD的安装光□。　如果没有刻□机，而上网□□却很便宜，它也是一□很好的安装方式。

1. 下□正□的 ISO 映像文件

   □个版本的ISO映像文件都可以从 ftp://ftp.FreeBSD.org/pub/FreeBSD/ISO-IMAGES-架□名/版本 或最近的□像站点下□。□□合□的 架□ 和 版本 。

   目□中包含下面一些映像文件：

   表 *4. FreeBSD 7.X* 和 *8.X ISO* 映像文件名和含□

文件名	包含内容
FreeBSD-版本-RELEASE-架□-bootonly.iso	□个 CD 映像可以□□从光□□□并□入安装□程，但它并不提供用于支持从 CD 直接安装 FreeBSD 所需的文件。 在从 CD 引□之后， □需要通□网□ (例如从 FTP 服□器) 来完成安装。
FreeBSD-版本-RELEASE-架□-dvd1.iso.gz	□个 DVD 映像包括用于安装 FreeBSD 操作系□基本□件、 □□□包和文□所需的全部文件。它也支持引□□入基于 "livefs" 的修□模式。
FreeBSD-版本-RELEASE-架□-memstick.img	□个映像可以写□ USB □□棒， 用于引□系□并完成安装。 它也支持引□□入基于 "livefs" 的修□模式。 □个版本的映像中包含了文□所需要的全部文件， 但不提供其他包。FreeBSD 7.3 和更早版本中没有□个文件。
FreeBSD-版本-RELEASE-架□-disc1.iso	□个 CD 映像包含了 FreeBSD 操作系□的基本□件和文□包， 但不包括其它包。
FreeBSD-版本-RELEASE-架□-disc2.iso	□个 CD 映像包含了能填□光□的尽可能多的第三方□件包。 在 FreeBSD 8.0 和更高版本中不提供□个映像。
FreeBSD-版本-RELEASE-架□-disc3.iso	□一个包含了能填□光□的尽可能多的第三方□件包的 CD 映像。 在 FreeBSD 8.0 和更高版本中不提供□个映像。
版本-RELEASE-架□-docs.iso	FreeBSD 文□。
FreeBSD-版本-RELEASE-架□-livefs.iso	□个 CD 映像包含了用以支持引□□入基于 "livefs" 的修□模式， 但不包括直接从 CD 安装所需的文件。

FreeBSD 7.X 系列在 FreeBSD 7.3 之前的版本， 以及 FreeBSD 8.X 系列在 FreeBSD 8.1 之前的版本使用不同的命名□□。 它□的 ISO 文件名不使用 FreeBSD- 前□。

□ 必□ 下□ bootonly ISO 映像 (如果有) 或 disc1 的映像其中的一个。 没有必要都下□， 因□ disc1 映像包含了 bootonly ISO 映像中的全部内容。

如果□的 Internet □□很廉价， □□使用 bootonly ISO。 它能安装 FreeBSD， 而□可以根据需要使用 ports/packages 系□来下□并安装第三方□件 (参□ 安装□用程序: Packages 和 Ports)。

如果打算安装 FreeBSD 并安装常用的□件包， □□使用 dvd1。

其它的映像□也很有用， 但不是必□的， 尤其是在□有高速的网□□接□。

2. 刻□ CDs

□必□把□些映像文件刻□成光□。 如果□在其它的FreeBSD系□上完成此□工作，□看 □建和使用光学介□(CD) 得到更多的信息，（特□是 burncd 和 cdrecord）

如果□在其它的系□平台上□行，□需要相□的刻□□件。 映像文件使用的是□准的ISO格式，必□被□的刻□□件所支持。

如果有□趣制作一□定制的 FreeBSD 版本， □参考 Release Engineering Article。

2.13.2. □ FreeBSD 安装□建立局域网 FTP 站点

FreeBSD 光□的布局和 FTP 站点相同。 □□， 建立局域网 FTP 站点来用于网□上的其它□算机安装 FreeBSD，就十分的容易。

1. 在要作□FTP站点的那台FreeBSD机器上， □定FreeBSD磁□放入光□中并将它挂在 /cdrom 目□中。

```
# mount /cdrom
```

2. 在 /etc/passwd 文件中建立一个可匿名□□ FTP 服□器的□号。 □可以利用 vipw(8) 命令□□ /etc/passwd 文件， 加入下面□一行叙述：

```
ftp:*:99:99::0:0:FTP:/cdrom:/nonexistent
```

3. □定在 /etc/inetd.conf 配置文件中□□了FTP服□。

任何本地网□中的机器在安装 FreeBSD □□安装介□□就可以□□透□ FTP 站点，然后□取 "Other" 后□入 ftp://本地FTP服□器 即可以透□本地的FTP站点来安装FreeBSD。

如果用作 FTP 客□端的引□介□ (通常是□□) 与本地局域网的 FTP 站点上的版本不一致，sysinstall 会不允□□完成安装。 如果□使用的版本差距不很大， 并且希望□□□一判断，□□□入 **Options** 菜□， 并将安装包的名字改□ any。

此方式最好使用在有防火□保□的内部网□。 如果要将此FTP服□公□□外面的网□网□（非本地用□）， □的□□必□承担被侵入或其它的□□。 我□□烈建□□要有完善的安全机制才□□做。

2.13.3. □建安装□□

如果□从□□安装（我□_不_推□那□做）， 或者是由于不支持硬件或者更□□的理由是因□□□持要使用□□安装。 □必□准□几□□□。

至少□些□□必□是 1.44 MB 的，用来容□所有在 base (基本系□) 目□下的文件。如果□在 DOS 操作系□下准□就必□ 使用 MS-DOS® 的 FORMAT 命令来格式化□□。 如果□使用的是 Windows® 操作系□， 在□源管理器中就可以完成□个工作 (用右□□□ A: □□器，并□□ "Format")。

不要 指望厂家的□先格式化！ 最好□是□自□行格式化。 □去用□□告的很多□□都是由于不正□地使用格式化□□所造成的， 所以我□需要在□里着重提一下。

如果□在□外一台FreeBSD的机器上做了□□□的□， □行格式化是一个不□的主意。 □然□不需要把□□□都做成DOS文件系□。□也可以使用 bsdlabel 和 newfs 命令来□建一个UFS文件系□，具体操作按下面的□序□行：

```
# fdformat -f 1440 fd0.1440
# bsdlabel -w fd0.1440 floppy3
# newfs -t 2 -u 18 -l 1 -i 65536 /dev/fd0
```

然后□就可以像其它的文件系□一□挂上和写入□些磁□。

格式化□些磁□后，□必□把文件□制到磁□中。 □些□行文件被分割成□好可存□五□ 1.44 MB □□。 □□□所有的磁□，□出所有可能□合的文件。 直到□□到所有需要的配置并且将它□以□□方式安置。 第一个配置都□□有一个子目□在磁□上， 例如： a:\base\base.aa、 a:\base\base.ab， 等等。

base.inf 文件， 也□放在 base 的第一□□上， 因□安装程序需要□取□个文件， 以了解在□得□布包□需要下□多少文件。

一旦□□入□□安装介□的屏幕， □□ Floppy 将会看到后面的提示符。

2.13.4. 从 MS-DOS® 分区安装

如果从 MS-DOS® 分区安装， □需要将□布文件□制到□分区根目□下的 freebsd 目□中。 例如： c:\freebsd。□必□□制一部分 CDROM 或 FTP 上的目□□□， 因此， 如果□从光□□行□制， 建□使用 DOS 的 xcopy 命令。下面是准□□行 FreeBSD 最小系□安装的例子：

```
C:\> md c:\freebsd
C:\> xcopy e:\bin c:\freebsd\bin\ /s
C:\> xcopy e:\manpages c:\freebsd\manpages\ /s
```

假□ C: □是□的空□空□， E: □是□挂接的 CDROM。

如果□没有光□□□器，□可以从以下网站下□□行包。ftp.FreeBSD.org. □一个□行包都在一个目□中，例如 *base* □行包可以在 12.0/base/ 目□中□到。

□很多□行包来□，如果□希望从 MS-DOS®分区安装的□ （□有足□的空□），安装 c:\freebsd - 下的□个文件－□个 BIN □行包只是最低限度的要求。

2.13.5. □建一个安装磁□

从磁□安装也□是最□□的方式， 比在□使用 FTP 安装或使用 CDROM □快。安装的程序假□是□□地被□□在磁□上。 在□得到所有配置文件后，□□地解□它□，用下面的命令：

```
# cd /freebsd/distdir
# tar cvf /dev/rwt0 dist1 ... dist2
```

在□安装的□候，□要□定留有足□的空□□□□目□（允□□□□） 来容□磁□安装□ 全部 的内容。由于不是随机□□ 磁□的，所以□□安装方法需要很多□□空□。

□始安装□，在从□□□□ 之前， 磁□机必□已□放在□□□□中。否□， 安装□程中可能会□不到它。

2.13.6. 通□网□安装

可用的网□安装□型有三□。 以太网 (□准的以太网控制器)、 串口 (PPP) 以及 并口 (PLIP (laplink □□))。

如果希望以最迅速的方式完成网□安装， 那□以太网□配器当然就是首□！ FreeBSD 支持□大多数常□ PC 以太网□； 系□能□支持的网□ (以及所需的配置) 可以在 FreeBSD □行版附□的硬件兼容□明中□到。 如果□使用的是系□支持的 PCMCIA 以太网□， 在□□□本加□ 之前 之前一定要把它□好！ 很不幸， FreeBSD 目前并不支持在安装□程中□□ PCMCIA □。

此外， □□需要知道自己的 IP 地址、 网□□型□□的子网掩□， 以及机器名。 如果□正通□ PPP □接安装而没有固定的静□ IP， 不用怕， □个 IP 地址会由□的 ISP 自□分配。 □的系□管理□会告□□□行网□配置所需的信息。 如果□需要通□名字而不是 IP 地址来□□其他主机， □□需要配置一个域名服□器， 可能□需要一个网□地址 (在使用 PPP □， □个地址是服□提供商的 IP 地址)。 如果□希望通□ HTTP 代理服□器来完成 FTP 安装， □需要知道代理服□器的地址。 如果□不知道□些信息， □□在□行□□安装 之前 向系□管理□或 ISP □□。

如果□使用一个 MODEM，那□就只有 PPP □一□□□了。在□安装的□程中， 要□定□能很容易地□得完整且快速的□于□服□提供商的信息。

如果□使用 PAP 或 CHAP 方式□接到□的 ISP， （□句□□，如果□不使用脚本在Windows®中□接到□的ISP），那□□需要在 ppp 提示符下□入 `dial` 命令。否□，当 PPP □接者只提供一□最□□的□端模□器，□必□知道如何使用□□ MODEM 的 "AT commands"□号到□的 ISP。 想知道更深入的信息可以参考使用手册中的用□□PPP那□ 以及 FAQ 。 如果□有一些□□，可以使用 `set log local …` 命令将日志□示在屏幕上。

□也可以通□并口□□□接到□外一台 FreeBSD 机器上□行安装，□可以考□使用 "laplink" 并口□□□行安装。通□并口安装要比通□串口 （最高 50 kbytes/sec）安装快得多。

2.13.6.1. 通□NFS安装之前

NFS 安装方式是非常方便的。只需要□□地将 FreeBSD 文件□制到一台服□器上，然后在安装□□□NFS介□。

如果□个服□器要 "特□端口" 才能支持 （如SUN的工作站），□需要在安装前在 **Options** 菜□中□置 `NFS Secure`。

如果□使用了一□低□量的以太网□比□糟□， 速度很慢，□□考□ `NFS Slow`的□□。

□了□到NFS安装的目的，□个服□器必□支持 subdir 加□。 例如，如果□的 FreeBSD 12.0 目□存在：ziggy:/usr/archive/stuff/FreeBSD，然后 `ziggy` 将必□允□直接挂上 /usr/archive/stuff/FreeBSD，而不□□是 /usr 或 /usr/archive/stuff。

在 FreeBSD的 /etc/exports 配置文件中， 是由 -alldirs □□来控制的。其它 NFS 服□器也□有不同的方式。如果□从服□器得到 permission denied □个信息， 可能是因□□没有正□的□用它。

Chapter 3. 安装 FreeBSD（□用于 9.x 及以后版本）

3.1. 概述

FreeBSD 提供了一个以文字□主、 便于使用的安装程序： 从 FreeBSD 9.0-RELEASE □始是指 bsdinstall，而在之前□是指 sysinstall。 本章介□ bsdinstall 的使用， 有□ sysinstall 的使用参□ 安装 FreeBSD。

学□完本章之后， □将知道：

- 如何□建 FreeBSD 安装介□。
- FreeBSD 如何□分目□硬□。
- 如何□□ bsdinstall。
- □行 bsdinstall □需要回答的□□， □□的具体含□， 以及□□如何回答。

□□本章之前， □□□：

- □看将要安装的 FreeBSD 版本所附的硬件支持列表， 以□定□的硬件能□被支持。

一般来□， 此安装□明是□□ i386™（“PC 兼容机”） 架□的□算机； 同□也会尽可能地□其他架□下的安装予以□明。 □然本文□□常更新， 但仍可能与所安装版本上附□的□明文□有些□出入， 因此建□□□将其作□常□的安装指□。

3.2. 硬件需求

3.2.1. 最低配置

安装 FreeBSD 所需的最低配置， 随版本及硬件架□而有所不同。

以下几□□□些信息□行了□□。 根据所□的安装方式， 可能需要使用 FreeBSD 支持的 CDROM 或网□□配器， □□准□安装介□。

3.2.1.1. FreeBSD/i386

FreeBSD/i386 需要 486 或更快的□理器， 最小 64 MB 的内存， 以及至少 1.1 GB 的硬□空□。

通常情况下□于老旧的□算机而言， 安装更大的内存和□出更多的硬□空□，会比使用更快的□理器□性能的提升更加明□。

3.2.1.2. FreeBSD/amd64

FreeBSD/amd64 支持□□□理器。 第一□是 AMD64 □理器， 包括 AMD Athlon™64、 AMD Athlon™64-FX、 AMD Opteron™ 以及更高□□的□理器。

能□使用 FreeBSD/amd64 的□一□□理器是采用了 Intel® EM64 架□的□理器。 □□□理器包括 Intel® Core™ 2 Duo、 Quad 和 Extreme 家族， □包括 Intel® Xeon™ 3000、 5000 和 7000 系列， 以及 Intel® Core™

i3、 i5 和 i7。

□于使用了 nVidia nForce3 Pro-150 的机器， 必□ 在 BIOS □置中禁用 IO APIC， 如果没有□□的□□就只能□而禁用 ACPI。 因□ Pro-150 芯片□存在 bug，而目前□没有能□□避此□□的方法。

3.2.1.3. FreeBSD/powerpc Apple® Macintosh®

支持所有内建 USB 的 New World Apple® Macintosh® 系□， 同□也□配置多 CPU 的机器提供 SMP 支持。

注意 32 位的内核只能使用内存的前 2 GB，而 PowerMac G3 □白机上的 FireWire® 也不被支持。

3.2.1.4. FreeBSD/sparc64

有□ FreeBSD/sparc64 的系□支持， □□ FreeBSD/sparc64 □目。

FreeBSD/sparc64 需要独占一□磁□。 目前□不支持与其他操作系□共享同一□磁□。

3.2.2. 支持的硬件

FreeBSD □行版所支持的硬件架□及□□会列在硬件兼容□明文件中， 此文件通常名□ HARDWARE.TXT，位于□行版介□的根目□下。 □些内容也可以在 FreeBSD 网站的 □行版信息 □面上□到。

3.3. 安装前的准□工作

3.3.1. □□□的数据

在将 FreeBSD 安装至目□机器前， □首先□□其上的重要数据并□□□□行□□。 FreeBSD 安装程序□硬□做任何改□前都会□行□□， 而一旦操作□始就无法撤□。

3.3.2. 决定将 FreeBSD 安装在何□

如果整个硬□上□安装 FreeBSD 一个操作系□， 那□□直接跳□此□； 但如果需要□ FreeBSD 与其他操作系□并存， 那□首先□当了解 FreeBSD 的硬□布局□□。

3.3.2.1. FreeBSD/i386 与 FreeBSD/amd64 的硬□布局

硬□可以分割成多个区域， □些区域称作 *partition*（分区）。

有□□硬□分区方式。 □□的 *Master Boot Record* (MBR， 主引□□□) 的分区表中可以定□四个 *primary partitions (*主分区*)*。 (由于□史原因， FreeBSD 中将主分区称作 *slice*。) □了突破四个分区的限制，可以将其中一个主分区□建□ *extended partition (*□展分区*)*， 并在其中建立 *logical partitions (*□□分区*)*。正如□看到的那□， □□方法十分□拙。

新式的 *GUID Partition Table (GUID* 分区表*)* (GPT) 提供了更□□便的磁□分区方法。 与□□的 MBR 分区相比，GPT 功能更□□大。 常□的 GPT □□可以在一□磁□上支持多□ 128 个分区， 从而无需再采用□似□□分区□□迭床架屋的□□。

一些旧式的操作系□， 如 Windows® XP 并不兼容 GPT 分区格式。 如果需要□ FreeBSD 与□□的操作系□共用一□硬□， 就必□使用 MBR 分区了。

FreeBSD 的□准引□加□器需要使用一个主分区或 GPT 分区。 (有□ FreeBSD 引□□程的□情， □参□ FreeBSD 引□□程。) 如果所有的主分区或 GPT 分区都已在使用中， □必□□ FreeBSD □出一个来使用。

最小安装的 FreeBSD 只需 1 GB 磁□空□。 不□， □是 非常 基本的安装， 而且也不会留下多少可用的空□。 比□□用的情况下， 如果不使用□形界面， 最小安装□分配至少 3 GB 的空□， 而使用□形界面， □□分配至少 5 GB 的空□。 此外， 第三方□用程序可能□需要更多的空□。

有很多 免□或商□的分区□整工具 可供使用。 例如， 以 Live CD 形式提供的 GParted Live 中的 GParted 分区□□器。 此外， GParted 也可以在□多其它 Linux Live CD □行版中□到。

磁□分区程序有可能会破坏□有的数据。 在修改磁□分区之前， □先做一次完整的□□并校□其完整性。

□整 Microsoft® Vista 分区大小□可能会遇到一些□□。 如果要□□做， □提前准□好 Vista 安装光□。

例 *3.* 使用□有的分区

假□一台安装了 Windows® 的□算机上有一□ 40 GB 的硬□， 分成了□个 20 GB 的分区。 Windows® 将它□分□叫做 C: 和 D:。 C: 分区包含了 10 GB 数据， 而 D: 分区包含了 5 GB 数据。

将数据从 D: 移□到 C:， 就可将第二个分区□出来供 FreeBSD 使用了。

例 *4.* □小□有的分区

假□一台安装了 Windows® 的□算机上有一□ 40 GB 的硬□， 一个大的分区使用了整□磁□的全部空□。 Windows® 将□个 40 GB 分区叫做 C:。 目前占用了 15 GB 空□。 □希望将 Windows® 分区□少到 20 GB， 并将余下的 20 GB 分□ FreeBSD 使用。

可以在以下□□方法中任□一□：

1. □□ Windows® 数据。 接着， 重新安装 Windows®， 在安装□程中建立一个 20 GB 的分区。
2. 使用□似 GParted □□的分区□整工具来□小 Windows® 分区， 并□出空□□ FreeBSD 使用。

包含不同操作系□的磁□分区令□能□在任何□候使用其中的一□。 如果希望同□□行多□不同的操作系□， 可以使用在 虚□化 中介□的方法。

3.3.3. 收集网□配置信息

某些 FreeBSD 安装方式需要通□网□□接下□相□文件。 若要□接至以太网 (或□□□□/DSL □制解□器上的以太网接口)， □需要向安装程序提供必要的网□配置信息。

DHCP 可以用来提供自□配置网□的信息。 假如没有可用的 DHCP， □必□从局域网管理□， 或网□服□提供商那里□得必要的配置信息：

1. IP 地址
2. 子网掩□
3. 默□网□的 IP 地址

4. 本地网□域名
5. DNS 服□器的 IP 地址

3.3.4. □□ FreeBSD □行勘□

尽管 FreeBSD □目会□保□个□行版尽可能地□定， 但 bug □是在所□免。 □少数情况下， □些 bug 甚至会影□安装。 一旦□些□□被□□并修正后， 就会列在 FreeBSD 网站的 FreeBSD □行勘□ 中。 在安装之前， □首先□□□些勘□， 以□保安装可以□利□行。

有□所有□行版的信息及勘□， 可以在 FreeBSD 网站 的 □行版信息 一□中□到。

3.3.5. 准□安装介□

FreeBSD 的安装介□包括 CD、 DVD 及 USB □□棒。 若要□始安装， 只需使用安装介□引□□算机即可；注意不能通□在其他操作系□中□行安装程序□□方式□行安装。

□准的安装介□中包含了 FreeBSD 安装所需的全部文件， 除此之外， □有一□ *bootonly* 安装介□。 □□介□并不在其中直接包含安装所需的全部文件， 而是在需要□通□网□□行下□。 因此， 与□准的安装介□相比，bootonly安装介□体□更小。

FreeBSD 安装介□的副本可以从 FreeBSD 网站 □取。

如果□已□有 FreeBSD 的安装 CD、 DVD 或 USB □□棒， □可以跳□此□。

FreeBSD 的安装 CD 或 DVD 映像均□可引□的 ISO 文件。 只需要 CD 或 DVD 其中的一□即可完成安装操作。任□一□在当前操作系□中刻□成可引□光□即可。

若要□建可引□的□□棒， □□行以下操作：

1. □取□□棒映像

   FreeBSD 9.0-RELEASE 和更高版本的□□棒映像文件可以在 ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/arch/arch/ISO-IMAGES/version/FreeBSD-version-RELEASE-arch-memstick.img 中的 ISO-IMAGES/ 目□中□到， 其中， *arch* 是指要安装的架□， 而 *version* □是指要安装的版本号。□例来□， FreeBSD/i386 9.0-RELEASE 的□□棒映像位于 ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/i386/ISO-IMAGES/9.0/FreeBSD-9.0-RELEASE-i386-memstick.img □到。

   在 FreeBSD 8.*X* 以及更早的版本中， 映像文件的下□位置略有不同。 □于 FreeBSD 8.*X* 和更早版本的安装操作□参□ 安装 FreeBSD。

   □□棒映像的□展名□ .img。 在 ISO-IMAGES/ 目□中提供了多个不同的映像， 可以根据需要的 FreeBSD 版本， 有□也包括安装□象的硬件状况□行□□。

   □行以下□□前， □ □□ USB □□棒上的数据， 因□之后的操作将 擦除 □些数据。

2. 将映像文件写入□□棒

   Procedure: 在 **FreeBSD** 中操作

   在下面的例子中， 目□□□棒□□的□□□点是 /dev/da0。 操作前□仔□□□目□□□是否正□， 以免□坏□有的数据。

   a. 使用 dd(1) 写入映像

      □展名□ .img 的映像文件 不是 一□普通的文件。 它是□□□棒上完整内容所做的 映像， 因此 不能 只是像普通文件一□□□的□制， 而□使用 dd(1) 将其直接写入目□□□：

      ```
      # dd if=FreeBSD-9.0-RELEASE-i386-memstick.img of=/dev/da0 bs=64k
      ```

Procedure: 在 **Windows®** 中操作

操作前□□□是否□目□□□□□了正□的□□器号， 否□可能会覆□并□坏□的□有数据。

a. □取 Image Writer for Windows®

Image Writer for Windows® 是一□能将映像正□写入到□□棒中的免□□用程序。 从 https://launchpad.net/win32-image-writer/ 下□并将其提取至任意文件□后即可□始使用。

b. 使用 Image Writer 写入映像

双□□□ Win32DiskImager □行程序后， □定 `Device` 下面□示的□□器号所□□的是□□棒。 点□文件□□□以□□需要写入的映像文件， 然后点□ **[Save]** 接受□□。 在□□所有操作无□且没有其他□口□□□□棒后， 点□ **[Write]** 将映像文件写入□□棒。

系□不再支持从□□□行安装了。

□□在可以□始安装 FreeBSD 了。

3.4. □始安装

默□情况下， 在□看到下面□条信息之前， 安装程序不会□硬□数据做任何修改：

```
Your changes will now be written to disk.  If you
have chosen to overwrite existing data, it will
be PERMANENTLY ERASED. Are you sure you want to
commit your changes?
```

在此之前均可安全退出， 抑或□担心□行了某些□□的配置， 也可以直接□□□源。

3.4.1. □机□□

3.4.1.1. 引□ i386™ 及 amd64 系□

1. 若要使用 准□安装介□ 所述的 USB □□棒引□， □□在□机前将其□入□算机。

   若要使用 CDROM 引□， □□在□机后立刻将其放入□算机。

2. 根据所使用的安装介□， □□从 CDROM 或 USB□□。 在 BIOS □置中， 可以□□特定的引□□□。 大多数系□□可以在□□□□□引□□□， 通常需要按 `F10`、 `F11`、 `F12` 或 `Escape` □。

3. 如果□的□算机正常□□并加□了□有的操作系□， 那□□□□：

   a. USB □□棒□入□□或 CDROM 放入□□， □将其拔下或取出， 然后重新□□□算机并再次□□。

   b. BIOS □置□□， □重新□置。

   c. BIOS 不支持从当前介□□□； 可以使用 Plop Boot Manager， 它能□□老式□算机支持 CD 或 USB □□。

4. FreeBSD 将□始□□。 如果使用的是CDROM， □会看到□似□□的□示（版本信息可以忽略）：

```
Booting from CD-ROM...
645MB medium detected
CD Loader 1.2

Building the boot loader arguments
Looking up /BOOT/LOADER... Found
Relocating the loader and the BTX
Starting the BTX loader

BTX loader 1.00 BTX version is 1.02
Consoles: internal video/keyboard
BIOS CD is cd0
BIOS drive C: is disk0
BIOS drive D: is disk1
BIOS 636kB/261056kB available memory

FreeBSD/i386 bootstrap loader, Revision 1.1

Loading /boot/defaults/loader.conf
/boot/kernel/kernel text=0x64daa0 data=0xa4e80+0xa9e40 syms
=[0x4+0x6cac0+0x4+0x88e9d]
\
```

5. FreeBSD 引□加□器会□示：

59. FreeBSD 引加器菜

可以等待十秒或按 `Enter` 。

3.4.1.2. 引 Macintosh® PowerPC®

在大多数机器上， 机按住 `C` 可以从 CD 。 除此之外， 按住 `Command` + `Option` + `O` + `F`， 在非 Apple® 上是 `Windows` + `Alt` + `O` + `F`， 然后在出的提示符 `0 >` 下入

```
boot cd:,\ppc\loader cd:0
```

于不的 Xserves 机器，参考 Apple® 支持网站 以了解如何引至 Open Firmware。

3.4.1.3. 引 sparc64

多数 sparc64 系均置成了硬自。 若要安装 FreeBSD， 从网或 CDROM ， 就需要首先入 PROM（OpenFirmware）。

重系后等待引信息出， 然其具体内容取决于机器型号， 但会似：

```
Sun Blade 100 (UltraSPARC-IIe), Keyboard Present
Copyright 1998-2001 Sun Microsystems, Inc.  All rights reserved.
OpenBoot 4.2, 128 MB memory installed, Serial #51090132.
Ethernet address 0:3:ba:b:92:d4, Host ID: 830b92d4.
```

如果此系已始从硬， 那按下 `L1` + `A` 或 `Stop` + `A` 或在串口控制台送 BREAK（在 tip(1) 或 cu(1) 中是 `~#`）以入 PROM 提示符， 它如下所示：

```
ok        ①
ok {0} ②
```

① □是在□ CPU 系□上的提示符。

② □是在 SMP 系□上的提示符， 其中的数字表示可用的 CPU 个数。

□在， 放入 CDROM 并在 PROM 提示符后□入 `boot cdrom`。

3.4.2. □看□□探□□果

□了便于□□， 屏幕上所□示的最后几百行字符会始□保存在□冲区里。

若要□□□冲区， 可以按下 `Scroll Lock` □来□□屏幕的□□功能； □□后即可使用方向□、 `PageUp` □或 `PageDown` □□行翻□； 再次按下 `Scroll Lock` □将□□□□功能。

□□□将看到内核□行了□□探□， 其□果□似 典型的□□探□□果 中的文本， 但具体内容会因□算机中所包含的□□而有所不同。

典型的□□探□□果

```
Copyright (c) 1992-2011 The FreeBSD Project.
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994
        The Regents of the University of California. All rights reserved.
FreeBSD is a registered trademark of The FreeBSD Foundation.
FreeBSD 9.0-RELEASE #0 r225473M: Sun Sep 11 16:07:30 BST 2011
    root@psi:/usr/obj/usr/src/sys/GENERIC amd64
CPU: Intel(R) Core(TM)2 Duo CPU     T9400  @ 2.53GHz (2527.05-MHz K8-class CPU)
  Origin = "GenuineIntel"  Id = 0x10676  Family = 6  Model = 17  Stepping = 6
  Features
=0xbfebfbff<FPU,VME,DE,PSE,TSC,MSR,PAE,MCE,CX8,APIC,SEP,MTRR,PGE,MCA,CMOV,PAT,PSE36,CL
FLUSH,DTS,ACPI,MMX,FXSR,SSE,SSE2,SS,HTT,TM,PBE>
  Features2
=0x8e3fd<SSE3,DTES64,MON,DS_CPL,VMX,SMX,EST,TM2,SSSE3,CX16,xTPR,PDCM,SSE4.1>
  AMD Features=0x20100800<SYSCALL,NX,LM>
  AMD Features2=0x1<LAHF>
  TSC: P-state invariant, performance statistics
real memory  = 3221225472 (3072 MB)
avail memory = 2926649344 (2791 MB)
Event timer "LAPIC" quality 400
ACPI APIC Table: <TOSHIB A0064   >
FreeBSD/SMP: Multiprocessor System Detected: 2 CPUs
FreeBSD/SMP: 1 package(s) x 2 core(s)
 cpu0 (BSP): APIC ID:  0
 cpu1 (AP): APIC ID:  1
ioapic0: Changing APIC ID to 1
ioapic0 <Version 2.0> irqs 0-23 on motherboard
kbd1 at kbdmux0
acpi0: <TOSHIB A0064> on motherboard
acpi0: Power Button (fixed)
acpi0: reservation of 0, a0000 (3) failed
```

```
acpi0: reservation of 100000, b6690000 (3) failed
Timecounter "ACPI-safe" frequency 3579545 Hz quality 850
acpi_timer0: <24-bit timer at 3.579545MHz> port 0xd808-0xd80b on acpi0
cpu0: <ACPI CPU> on acpi0
ACPI Warning: Incorrect checksum in table [ASF!] - 0xFE, should be 0x9A
(20110527/tbutils-282)
cpu1: <ACPI CPU> on acpi0
pcib0: <ACPI Host-PCI bridge> port 0xcf8-0xcff on acpi0
pci0: <ACPI PCI bus> on pcib0
vgapci0: <VGA-compatible display> port 0xcff8-0xcfff mem 0xff400000-
0xff7fffff,0xe0000000-0xefffffff irq 16 at device 2.0 on pci0
agp0: <Intel GM45 SVGA controller> on vgapci0
agp0: aperture size is 256M, detected 131068k stolen memory
vgapci1: <VGA-compatible display> mem 0xffc00000-0xffcfffff at device 2.1 on pci0
pci0: <simple comms> at device 3.0 (no driver attached)
em0: <Intel(R) PRO/1000 Network Connection 7.2.3> port 0xcf80-0xcf9f mem 0xff9c0000-
0xff9dffff,0xff9fe000-0xff9fefff irq 20 at device 25.0 on pci0
em0: Using an MSI interrupt
em0: Ethernet address: 00:1c:7e:6a:ca:b0
uhci0: <Intel 82801I (ICH9) USB controller> port 0xcf60-0xcf7f irq 16 at device 26.0
on pci0
usbus0: <Intel 82801I (ICH9) USB controller> on uhci0
uhci1: <Intel 82801I (ICH9) USB controller> port 0xcf40-0xcf5f irq 21 at device 26.1
on pci0
usbus1: <Intel 82801I (ICH9) USB controller> on uhci1
uhci2: <Intel 82801I (ICH9) USB controller> port 0xcf20-0xcf3f irq 19 at device 26.2
on pci0
usbus2: <Intel 82801I (ICH9) USB controller> on uhci2
ehci0: <Intel 82801I (ICH9) USB 2.0 controller> mem 0xff9ff800-0xff9ffbff irq 19 at
device 26.7 on pci0
usbus3: EHCI version 1.0
usbus3: <Intel 82801I (ICH9) USB 2.0 controller> on ehci0
hdac0: <Intel 82801I High Definition Audio Controller> mem 0xff9f8000-0xff9fbfff irq
22 at device 27.0 on pci0
pcib1: <ACPI PCI-PCI bridge> irq 17 at device 28.0 on pci0
pci1: <ACPI PCI bus> on pcib1
iwn0: <Intel(R) WiFi Link 5100> mem 0xff8fe000-0xff8fffff irq 16 at device 0.0 on pci1
pcib2: <ACPI PCI-PCI bridge> irq 16 at device 28.1 on pci0
pci2: <ACPI PCI bus> on pcib2
pcib3: <ACPI PCI-PCI bridge> irq 18 at device 28.2 on pci0
pci4: <ACPI PCI bus> on pcib3
pcib4: <ACPI PCI-PCI bridge> at device 30.0 on pci0
pci5: <ACPI PCI bus> on pcib4
cbb0: <RF5C476 PCI-CardBus Bridge> at device 11.0 on pci5
cardbus0: <CardBus bus> on cbb0
pccard0: <16-bit PCCard bus> on cbb0
isab0: <PCI-ISA bridge> at device 31.0 on pci0
isa0: <ISA bus> on isab0
ahci0: <Intel ICH9M AHCI SATA controller> port 0x8f58-0x8f5f,0x8f54-0x8f57,0x8f48-
0x8f4f,0x8f44-0x8f47,0x8f20-0x8f3f mem 0xff9fd800-0xff9fdfff irq 19 at device 31.2 on
pci0
```

```
ahci0: AHCI v1.20 with 4 3Gbps ports, Port Multiplier not supported
ahcich0: <AHCI channel> at channel 0 on ahci0
ahcich1: <AHCI channel> at channel 1 on ahci0
ahcich2: <AHCI channel> at channel 4 on ahci0
acpi_lid0: <Control Method Lid Switch> on acpi0
battery0: <ACPI Control Method Battery> on acpi0
acpi_button0: <Power Button> on acpi0
acpi_acad0: <AC Adapter> on acpi0
acpi_toshiba0: <Toshiba HCI Extras> on acpi0
acpi_tz0: <Thermal Zone> on acpi0
attimer0: <AT timer> port 0x40-0x43 irq 0 on acpi0
Timecounter "i8254" frequency 1193182 Hz quality 0
Event timer "i8254" frequency 1193182 Hz quality 100
atkbdc0: <Keyboard controller (i8042)> port 0x60,0x64 irq 1 on acpi0
atkbd0: <AT Keyboard> irq 1 on atkbdc0
kbd0 at atkbd0
atkbd0: [GIANT-LOCKED]
psm0: <PS/2 Mouse> irq 12 on atkbdc0
psm0: [GIANT-LOCKED]
psm0: model GlidePoint, device ID 0
atrtc0: <AT realtime clock> port 0x70-0x71 irq 8 on acpi0
Event timer "RTC" frequency 32768 Hz quality 0
hpet0: <High Precision Event Timer> iomem 0xfed00000-0xfed003ff on acpi0
Timecounter "HPET" frequency 14318180 Hz quality 950
Event timer "HPET" frequency 14318180 Hz quality 450
Event timer "HPET1" frequency 14318180 Hz quality 440
Event timer "HPET2" frequency 14318180 Hz quality 440
Event timer "HPET3" frequency 14318180 Hz quality 440
uart0: <16550 or compatible> port 0x3f8-0x3ff irq 4 flags 0x10 on acpi0
sc0: <System console> at flags 0x100 on isa0
sc0: VGA <16 virtual consoles, flags=0x300>
vga0: <Generic ISA VGA> at port 0x3c0-0x3df iomem 0xa0000-0xbffff on isa0
ppc0: cannot reserve I/O port range
est0: <Enhanced SpeedStep Frequency Control> on cpu0
p4tcc0: <CPU Frequency Thermal Control> on cpu0
est1: <Enhanced SpeedStep Frequency Control> on cpu1
p4tcc1: <CPU Frequency Thermal Control> on cpu1
Timecounters tick every 1.000 msec
hdac0: HDA Codec #0: Realtek ALC268
hdac0: HDA Codec #1: Lucent/Agere Systems (Unknown)
pcm0: <HDA Realtek ALC268 PCM #0 Analog> at cad 0 nid 1 on hdac0
pcm1: <HDA Realtek ALC268 PCM #1 Analog> at cad 0 nid 1 on hdac0
usbus0: 12Mbps Full Speed USB v1.0
usbus1: 12Mbps Full Speed USB v1.0
usbus2: 12Mbps Full Speed USB v1.0
usbus3: 480Mbps High Speed USB v2.0
ugen0.1: <Intel> at usbus0
uhub0: <Intel UHCI root HUB, class 9/0, rev 1.00/1.00, addr 1> on usbus0
ugen1.1: <Intel> at usbus1
uhub1: <Intel UHCI root HUB, class 9/0, rev 1.00/1.00, addr 1> on usbus1
ugen2.1: <Intel> at usbus2
```

```
uhub2: <Intel UHCI root HUB, class 9/0, rev 1.00/1.00, addr 1> on usbus2
ugen3.1: <Intel> at usbus3
uhub3: <Intel EHCI root HUB, class 9/0, rev 2.00/1.00, addr 1> on usbus3
uhub0: 2 ports with 2 removable, self powered
uhub1: 2 ports with 2 removable, self powered
uhub2: 2 ports with 2 removable, self powered
uhub3: 6 ports with 6 removable, self powered
ugen2.2: <vendor 0x0b97> at usbus2
uhub8: <vendor 0x0b97 product 0x7761, class 9/0, rev 1.10/1.10, addr 2> on usbus2
ugen1.2: <Microsoft> at usbus1
ada0 at ahcich0 bus 0 scbus1 target 0 lun 0
ada0: <Hitachi HTS543225L9SA00 FBEOC43C> ATA-8 SATA 1.x device
ada0: 150.000MB/s transfers (SATA 1.x, UDMA6, PIO 8192bytes)
ada0: Command Queueing enabled
ada0: 238475MB (488397168 512 byte sectors: 16H 63S/T 16383C)
ada0: Previously was known as ad4
ums0: <Microsoft Microsoft 3-Button Mouse with IntelliEyeTM, class 0/0, rev 1.10/3.00,
addr 2> on usbus1
SMP: AP CPU #1 Launched!
cd0 at ahcich1 bus 0 scbus2 target 0 lun 0
cd0: <TEAC DV-W28S-RT 7.0C> Removable CD-ROM SCSI-0 device
cd0: 150.000MB/s transfers (SATA 1.x, ums0: 3 buttons and [XYZ] coordinates ID=0
UDMA2, ATAPI 12bytes, PIO 8192bytes)
cd0: cd present [1 x 2048 byte records]
ugen0.2: <Microsoft> at usbus0
ukbd0: <Microsoft Natural Ergonomic Keyboard 4000, class 0/0, rev 2.00/1.73, addr 2>
on usbus0
kbd2 at ukbd0
uhid0: <Microsoft Natural Ergonomic Keyboard 4000, class 0/0, rev 2.00/1.73, addr 2>
on usbus0
Trying to mount root from cd9660:/dev/iso9660/FREEBSD_INSTALL [ro]...
```

□仔□□□□□探□□果， 以□定 FreeBSD □到了所有□希望使用的□□。 没有□到的□□并不会在□里列出， 因□默□的 GENERIC 内核中不包含它□； 可以通□ 内核模□ □□些□□提供支持。

□□探□完成后， □将看到 □□安装介□的使用方式， 表明安装介□共有三□用途： 安装 FreeBSD 、 作□"`Live CD`"或引□至 FreeBSD 的命令行界面。 □使用方向□□□一□后按 `Enter` □□□。

□ *60.* □□安装介□的使用方式

在□里， □□□ **[Install]** 以□行安装程序。

3.5. 介□ bsdinstall

bsdinstall 是一个基于文本的 FreeBSD 安装程序， 作者是 Nathan Whitehorn <nwhitehorn@FreeBSD.org>， 于 2011 年被 FreeBSD 9.0 采用。

Kris Moore <kmoore@FreeBSD.org> □ PC-BSD □写的 pc-sysinstall 也可以用于 安装 FreeBSD。 □然有□会同 bsdinstall 混□， 但□□□者并不相□。

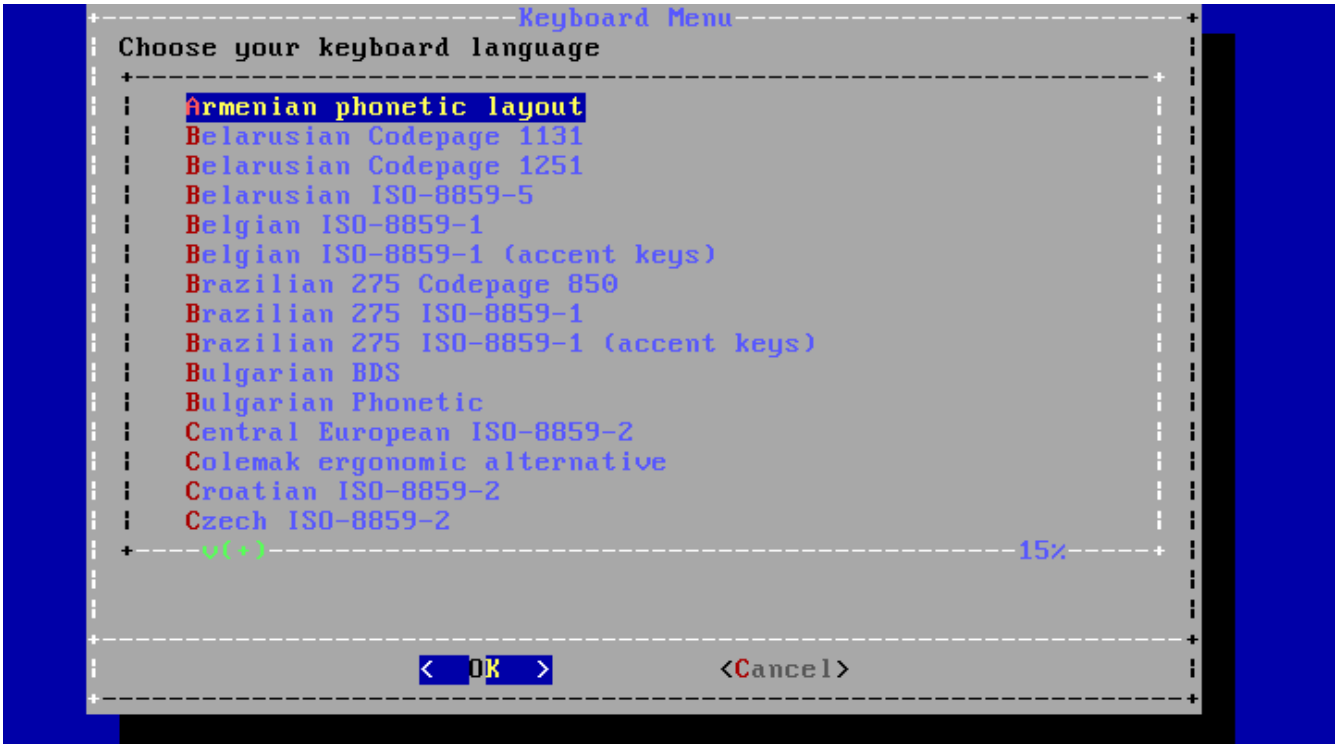
bsdinstall 菜□系□的主要控制□包括方向□、 `Enter` □、 `Tab` □、 `Space` □等。

3.5.1. □□□□映射

根据当前正在使用的系□控制台， bsdinstall 可能会首先提示□□一□非默□的□□布局。

□ *61.* □□映射□□

□□了 **[YES]** 后， 将□示下面的□□□□画面； 否□将不□示此画面而直接使用默□□□映射。



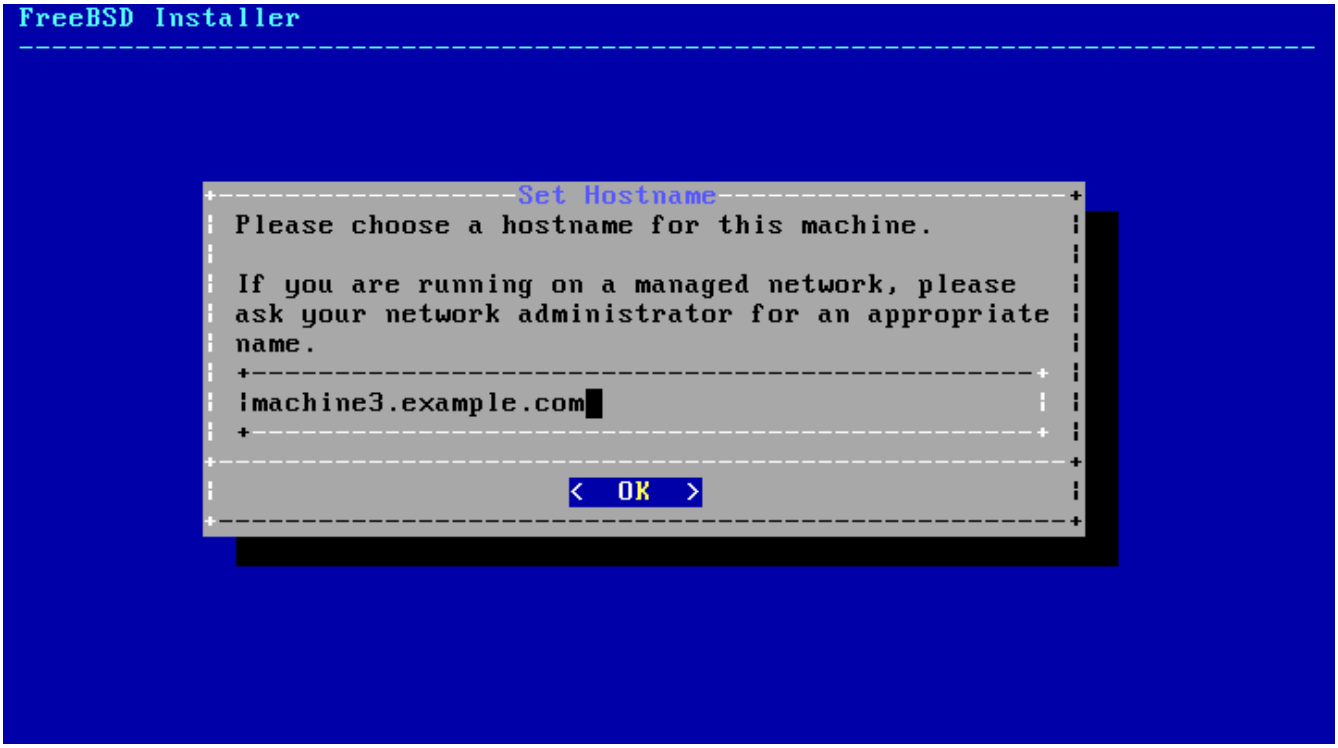
□ *62.* □□□□菜□

使用上/下方向□□□最□合当前系□的□□映射后， 按 `Enter` □□□。

按 `Esc` □以使用默□的□□映射。 如果不清楚□□□□一□， 推□ United States of America ISO-8859-1。

3.5.2. □置主机名

下面， bsdinstall 将提示□新安装的系□□置主机名。

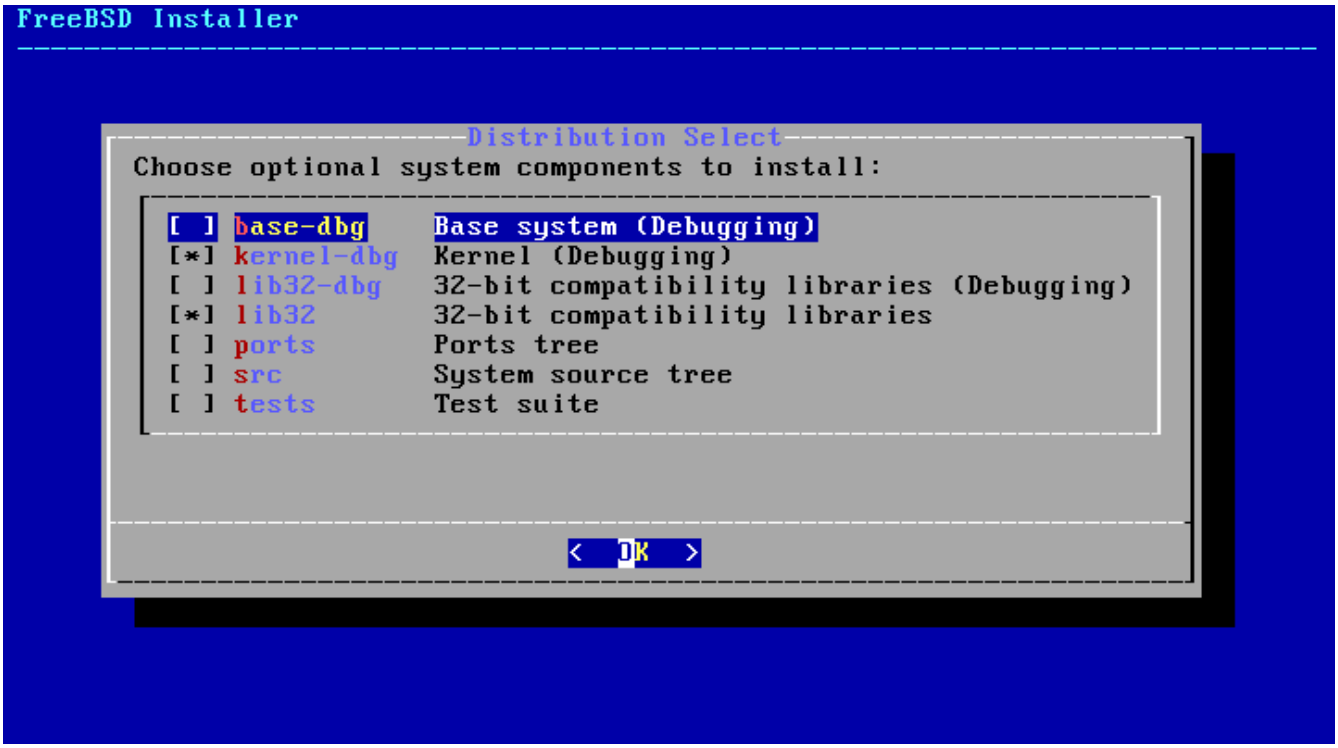


□ *63.* □置主机名

□□□入完整的主机名， 例如 `machine3.example.com`。

3.5.3. □□要安装的□件

下面， bsdinstall 将提示□□要安装的□件。



□ *64.* □□要安装的□件

安装□些□件很大程度取决于系□用途及可用磁□空□。 注意， 任何情况下都会安装 FreeBSD 内核及用□空□（□称"基系□"）。

根据安装□型的不同， 某些□件可能不会□示。

可□□件

- doc - 附加文□， 主要是与□目□史相□的内容。 □后□可以安装 FreeBSD 文□□□所提供的文□。
- games - 一些□□的 BSD 游□， 包括 fortune 与 rot13 等。
- lib32 - 兼容□文件， 用于在 64 位版本的 FreeBSD 上□行 32 位程序。
- ports - FreeBSD 的 ports 集。

  ports 集提供了一□□□而方便的途径来安装□件。 在 ports 集中， 并不包含□□□件所需的源代□， 取而代之的是一□能□自□下□、 □□并安装第三方□件包的文件。 安装□用程序. Packages 和 Ports 会□述如何使用 ports 集。

  □□此□□， 必□保□有足□的硬□空□， 注意安装程序并不会□此□行□□。 FreeBSD 9.0 的 ports 集□需 500 MB 的磁□空□； □也可以□□后的版本□留更大的空□。

- src - 系□源代□。

  FreeBSD 提供了与内核及用□空□有□的完整源代□。 大部分程序并不需要□些源代□， 它□主要用于□□特定□件（例如□□□□或内核模□）或者 FreeBSD 本身的□□。

  完整的源代□□需要 1 GB 的磁□空□， 而重新□□整个 FreeBSD 系□□□外□需要 5 GB 的空□。

3.6. 通□网□安装

bootonly 安装介□中并不会包含所有的安装文件。 如果使用□□介□□行安装， 那□需要的文件就必□通□网□下□。

□ *65.* 通□网□安装

根据 配置网□接□ 配置了网□□接后， 即可□始□□像站点。 □像站点上□存有 FreeBSD 的安装文件，□□一个更近的□像站点有助于更快的□取□些文件， 从而□少安装□□。

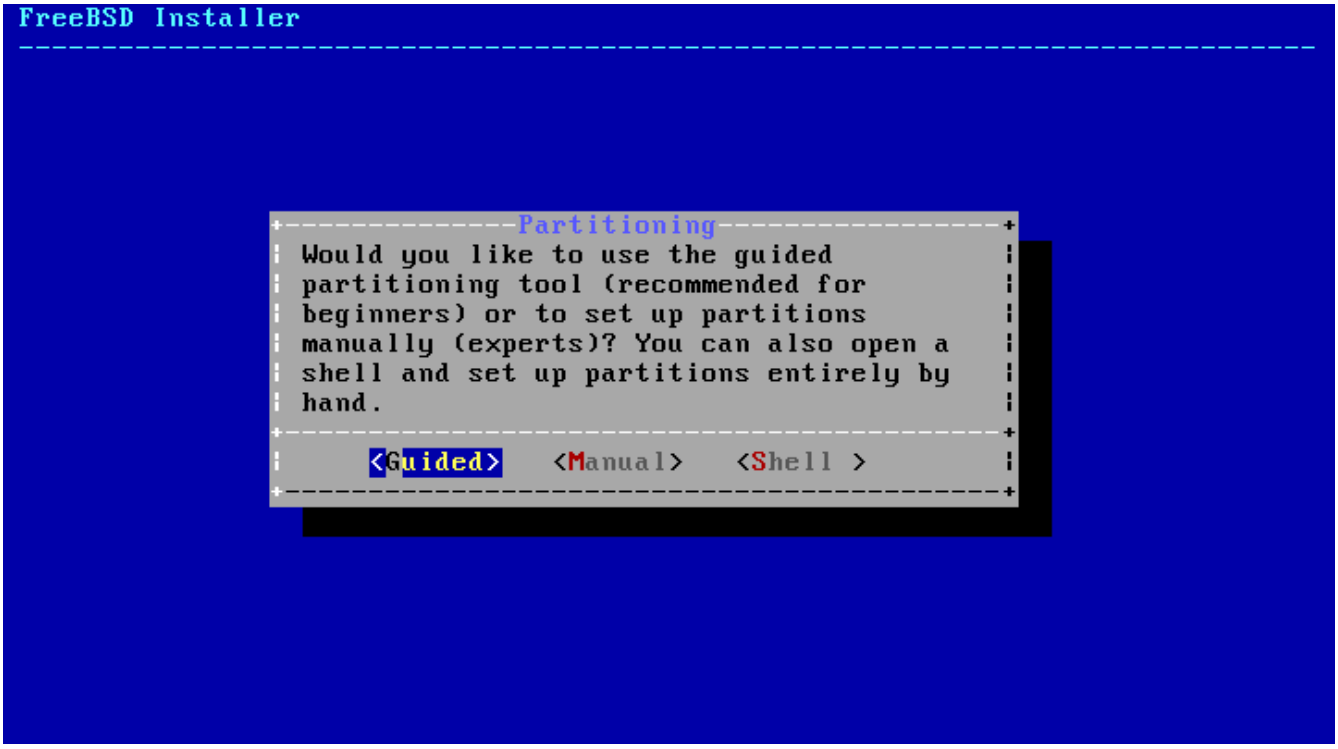
□ *66.* □□一个□像站点

□接至所□□像站点并□□到所需文件后， 安装将□□□行。

3.7. 分配磁□空□

FreeBSD 提供了三□方式来分配磁□空□： *Guided*（向□式） 分区能□自□□置磁□分区； 而 *Manual*（手□式）

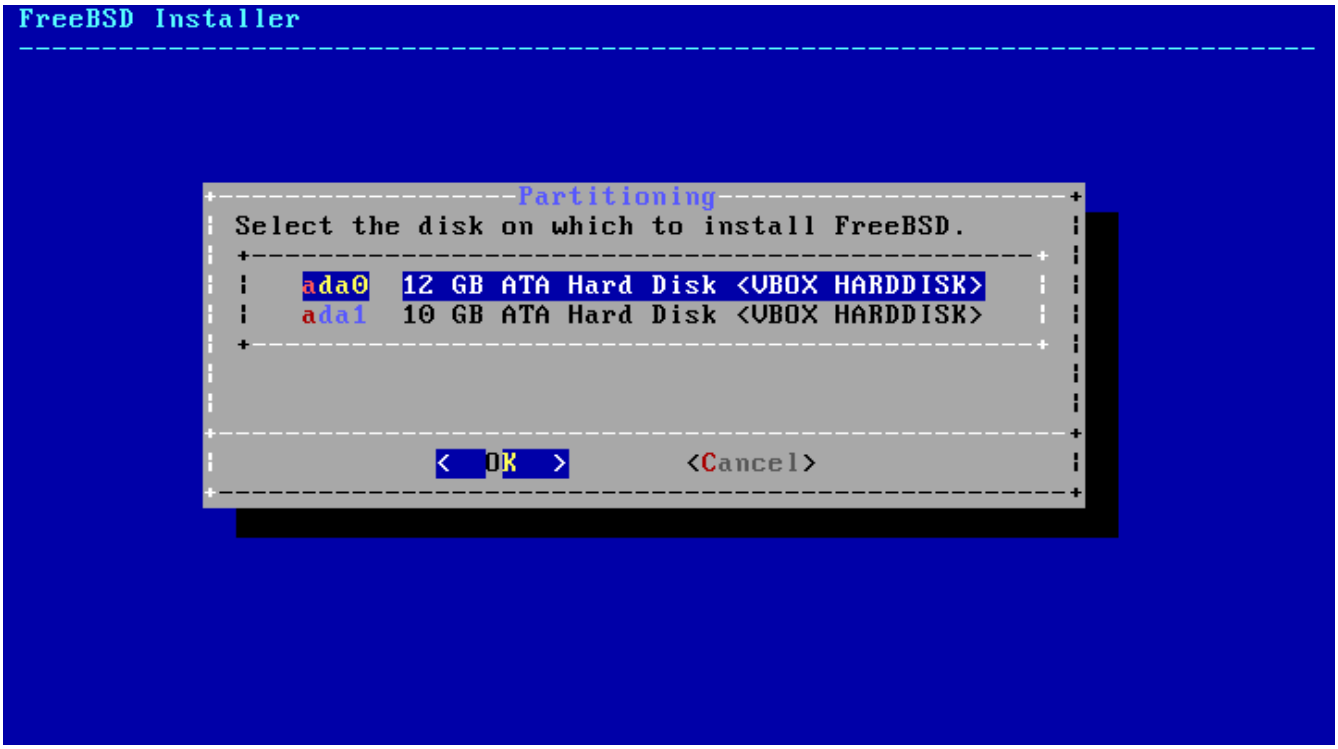
分区□允□高□用□□建自定□分区； □可以□入 shell 中直接使用□似 gpart(8)、 fdisk(8) 与 bsdlabel(8) □□的命令行程序。



□ *67.* □□分配磁□空□的方式

3.7.1. 向□式分区

如果机器上配有多□磁□， □需要□ FreeBSD 的安装指定目□磁□。

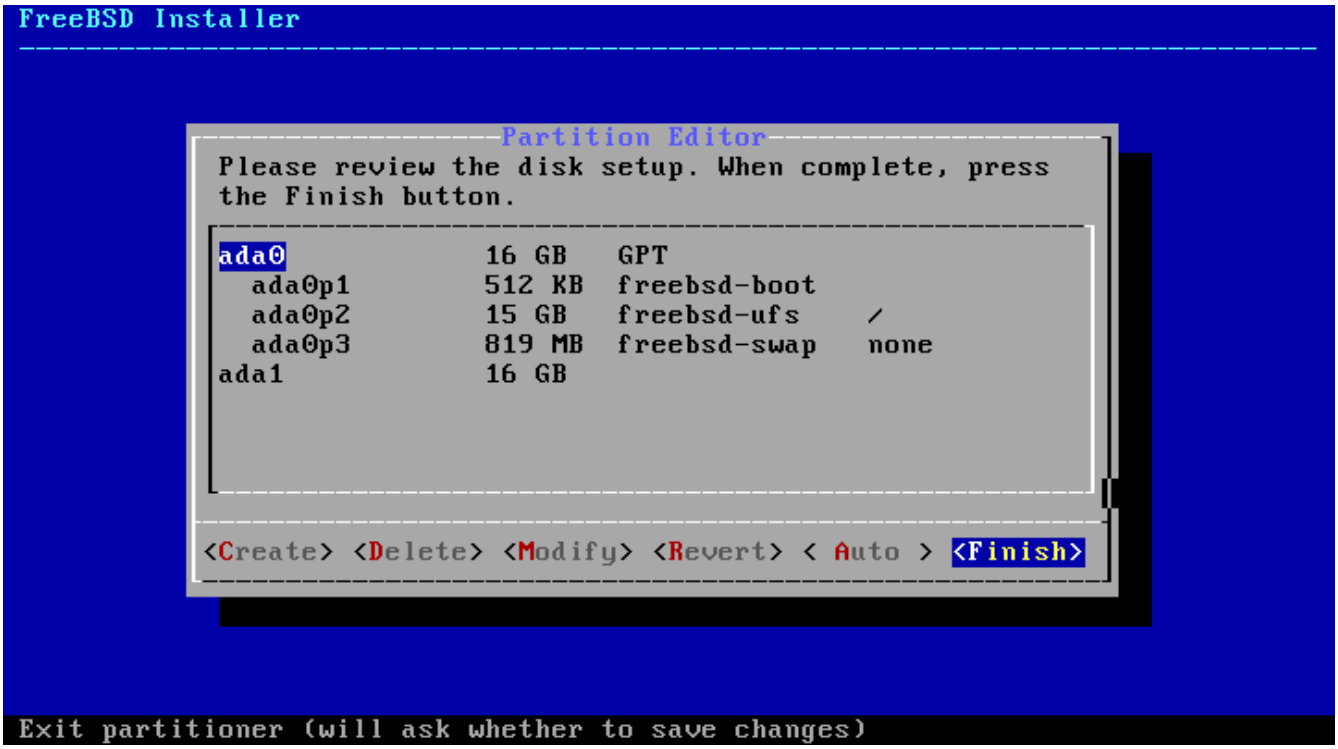


□ *68.* 从多□磁□中□行□□

可以将整个磁□都分配□ FreeBSD， 也可以只分配其中的一部分。 若□□的是 **[Entire Disk]**， □□建分区布局□会直接使用整个磁□； 若□□的是 **[Partition]**， □□建分区□□会使用磁□上的空□空□。

□ *69.* □□如何□建分区布局

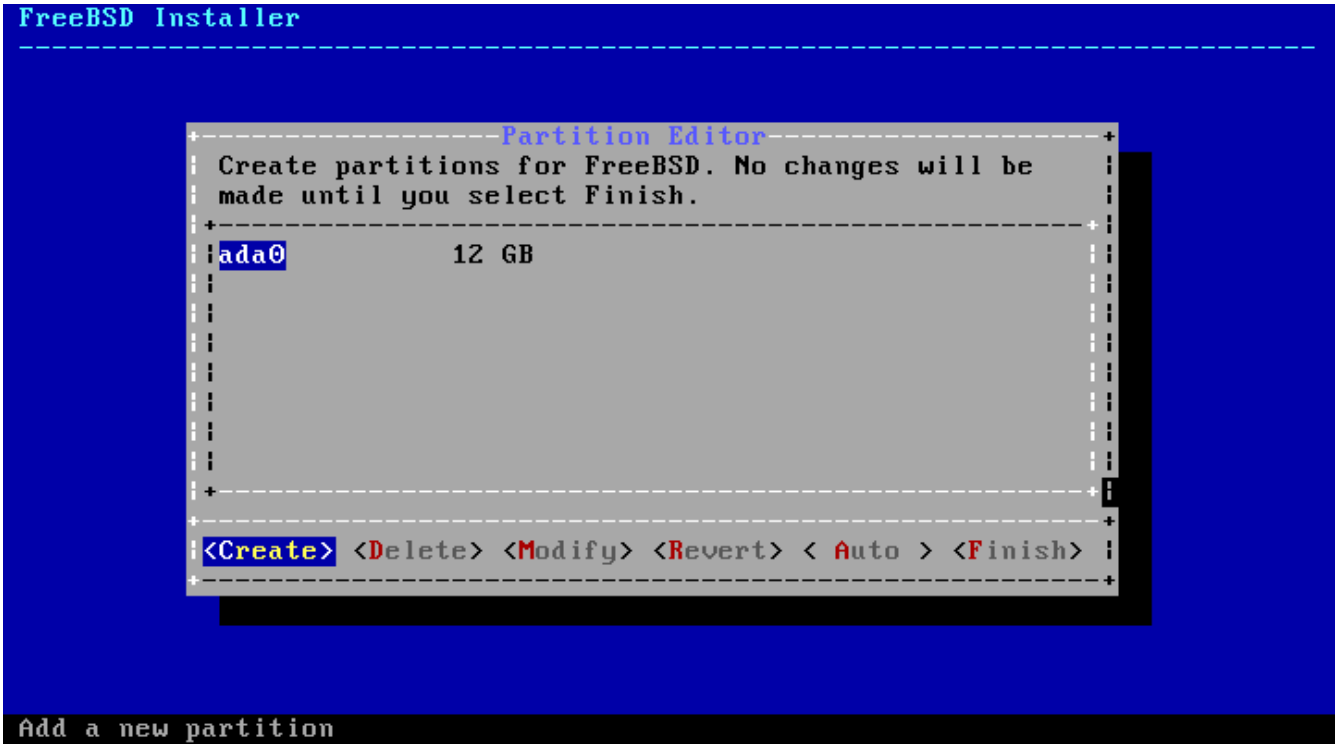
□仔□□□分区布局的□建□果。 如果□□有□□之□， 可以□□ **[Revert]** 来□原之前的分区； 此外， 也可以□□ **[Auto]** 重新□ FreeBSD 自□□建分区。 也可以手□□建、 修改或□除分区。 正□□建了分区之后， □□□ **[Finish]** 以□□安装。



□ *70.* □□已□建分区

3.7.2. 手□式分区

手□式分区将直接使用分区□□器□行操作。



□ *71.* 手□□建分区

高亮目□□□器（本例中□ ada0）并□□ **[Create]** 以□示 *partitioning scheme*（分区方案） 菜□。

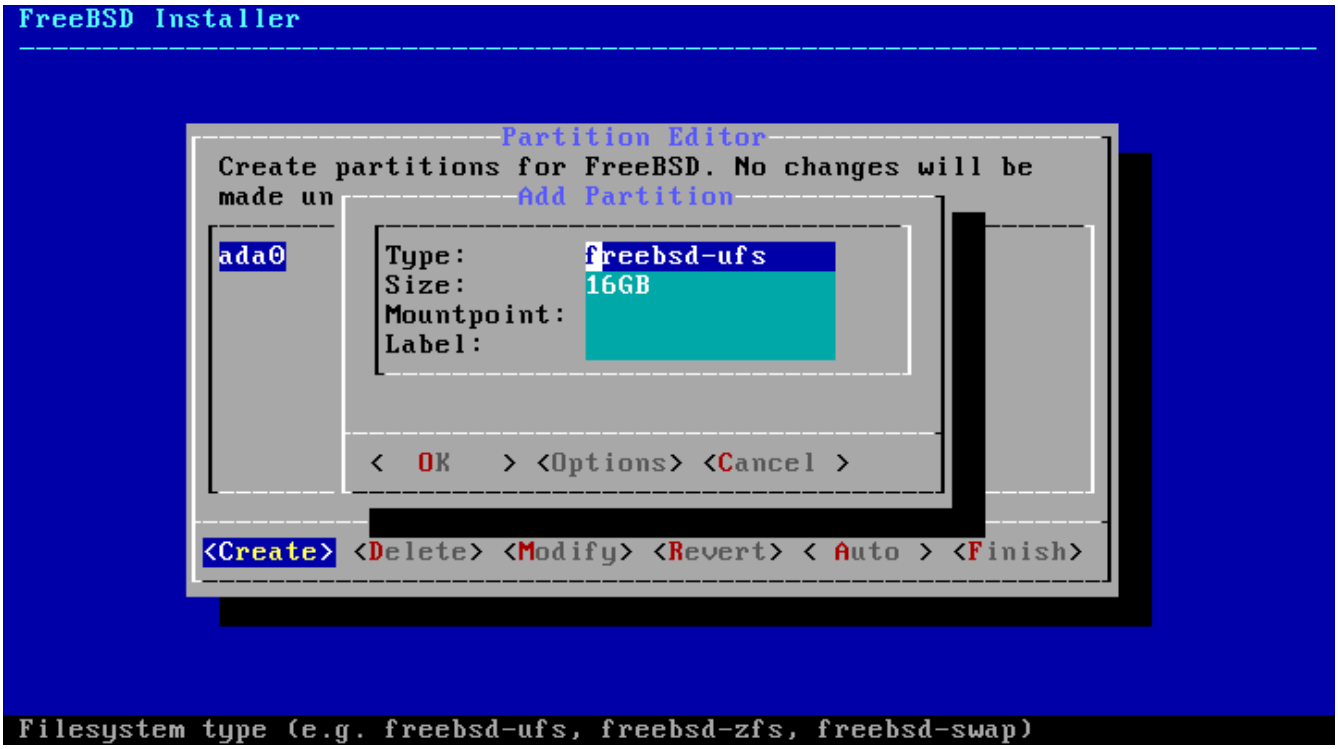
□ *72.* 手□□建分区

□于 PC 兼容机来□， GPT 分区通常是最合□的□□， 而某些不兼容 GPT 的老式操作系□□可能需要使用 MBR 分区。 除此之外的分区方案□用于一些不常□的或其他的老式操作系□。

表 *5.* 分区方案

□写	□明
APM	Apple Partition Map， 用于 PowerPC® Macintosh®。
BSD	不□ MBR 的 BSD Label， 有□也称作危□的□用模式，“dangerously dedicated mode”。 □参□ bsdlabel(8)。
GPT	GUID 分区表。
MBR	Master Boot Record， 主引□□□。
PC98	MBR □体， 用于 NEC PC-98 □算机。
VTOC8	Volume Table Of Contents， 用于 Sun SPARC64 和 UltraSPARC □算机。

□定了分区方案并□建完成后， 可再次□□ **[Create]** 以□建新的分区。



□ *73.* 手□□建分区

FreeBSD 的□准 GPT 安装至少会使用三个分区：

□准 *FreeBSD GPT* 分区

- `freebsd-boot` - FreeBSD 引□分区， 它必□□于首位。
- `freebsd-ufs` - FreeBSD 的 UFS 文件系□。
- `freebsd-swap` - FreeBSD 的交□空□。

也可以同□□建多个文件系□分区。 有些用□会喜□□□的分区格局， □ /、 /var、 /tmp， 以及 /usr 文件系□分□□建分区。 □参□ □建□□的分割式文件系□分区 中的例子。

可用的 GPT 分区□型可以在 gpart(8) 中□到。

在指定尺寸□， 可以使用常用的□写： *K* 表示 kilobytes、 *M* 表示 megabytes， 而 *G* 表示 gigabytes。

正□地□□磁□扇区能□□取最佳性能。 无□磁□的□个扇区□ 512 字□□是 4K 字□， 将分区大小□置□ 4K 字□的倍数都能□□保□□。 □□操作中， 只要使分区的大小等于 1M 或 1G 的倍数即可。 唯一的例外是 *freebsd-boot* 分区， 目前由于引□代□所限， 此分区不能大于 512K。

若分区包含文件系□，□需要在 Mountpoint □中□其□入挂□点； 若□□建了一个 UFS 分区， □□在此□中□入 /。

最后需要□入的是 *Label*（□□） □， 用于命名所□建的分区。 如果将□□器□接至不同的控制器或端口，其名称或□号会□生改□， 但□□的□□并不会□化。 在□似 /etc/fstab □□的文件中， 通□□□引用分区比通□□□器名加分区□号引用更加□活， 因□□□引用使系□□硬件的改□更加□容。 GPT 的□□会在磁□□接后出□在 /dev/gpt/ 中； 而其他分区方案中的□□也有不同的功能， 它□会出□在 /dev/ 中的不同目□里。

□避免冲突， □□□个文件系□指定独一无二的□□。 与□算机的名称、 用途或位置相□的字符均可添加至□□。 例如， □□室□算机的 UFS 根目□可以命名□ “labroot” 或 “rootfs-lab”。

例 *5.* □建□□的分割式文件系□分区

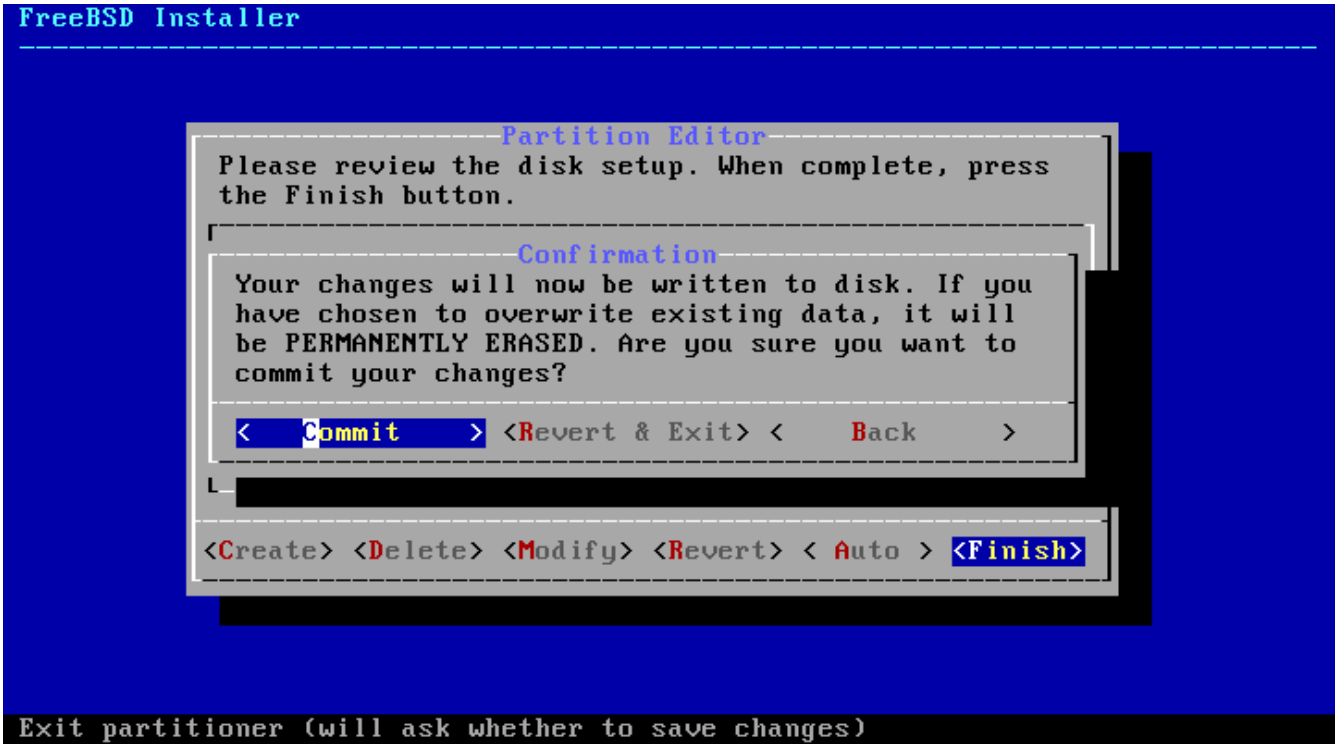
在□□的分区布局中， 目□ /、 /var、 /tmp 及 /user 都是位于自己分区上的独立文件系□； 在 GPT 分区方案中也可以□建□□的分区布局。 本例中所使用的是一□ 20G 的硬□， 如果使用更大的硬□， 建□□建更大的交□或 /var 分区。 □□的前□ `ex` 是指 “example”， 具体操作□□可以使用任何独一无二的字符。

分区□型	大小	挂□点	□□
`freebsd-boot`	`512K`		
`freebsd-ufs`	`2G`	/	`exrootfs`
`freebsd-swap`	`4G`		`exswap`
`freebsd-ufs`	`2G`	/var	`exvarfs`
`freebsd-ufs`	`1G`	/tmp	`extmpfs`
`freebsd-ufs`	接受默□□（剩余空□）	/usr	`exusrfs`

□建了自定□分区后， □□□ **[Finish]** 以□□安装。

3.8. 安装□□

下面， 安装程序将真正□硬□□行写操作， □也是取消安装的最后机会。



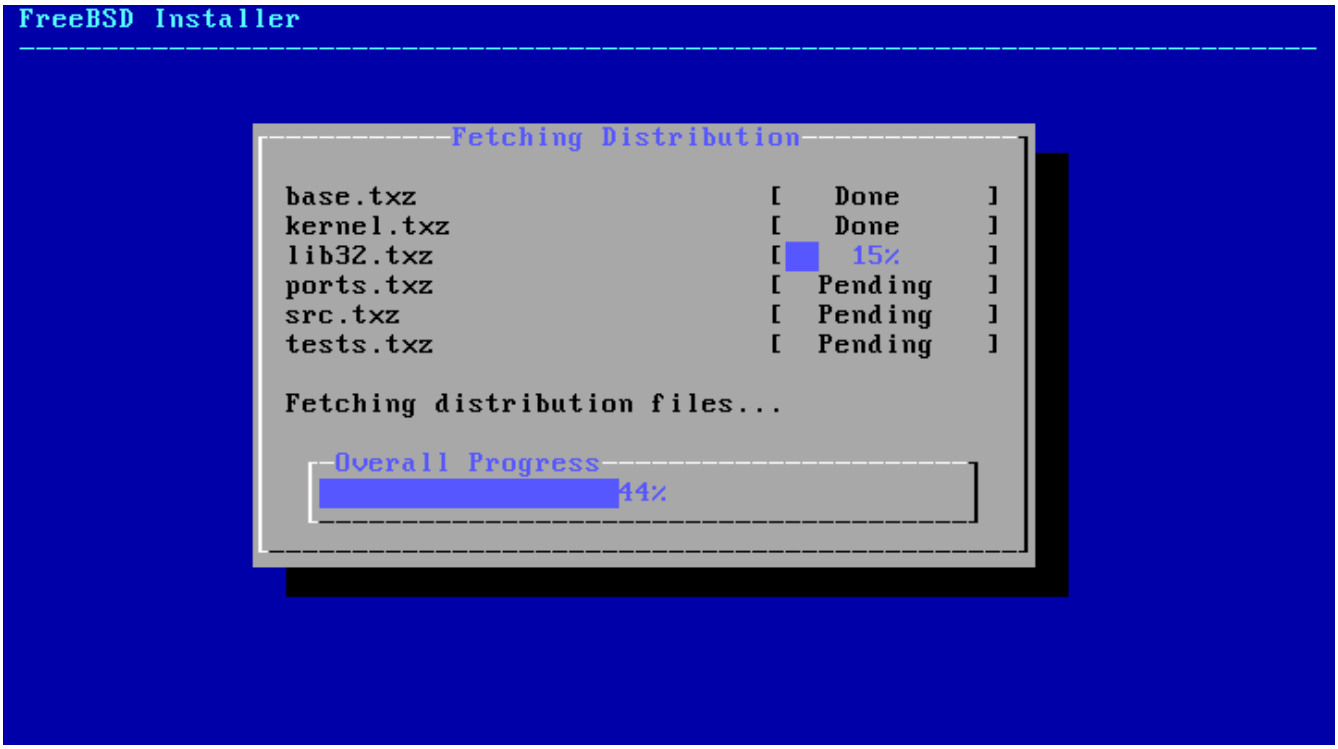
□ *74.* 最后□□

□□ **[Commit]** 并按 Enter □□□安装； □□ **[Back]** 以返回分区□□器□行修改； □□ **[Revert & Exit]**以退出安装而不修改任何硬□数据。

根据所□□件、 安装介□和机器速度的不同， 需要的□□会有所□化。 安装□会有一系列信息□示目前的□度。

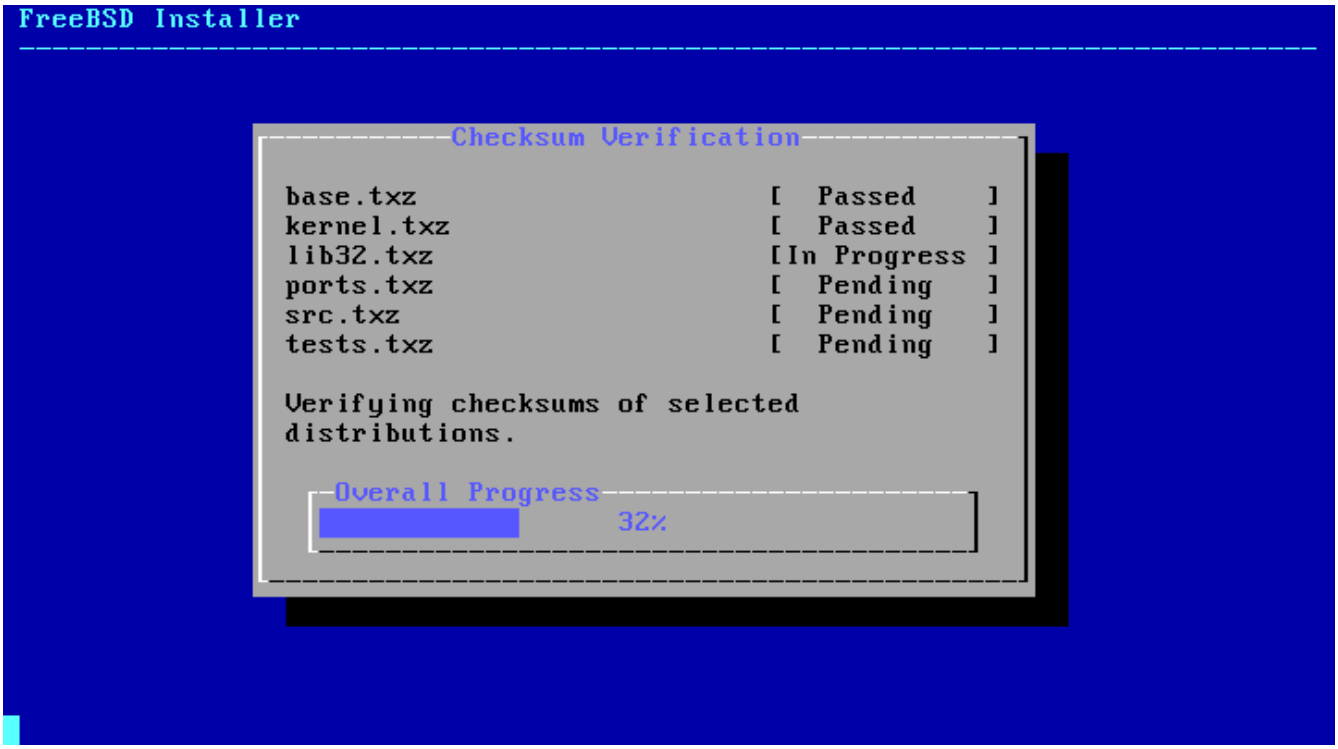
首先， 安装程序会将分区布局写入磁□， 并□行 newfs 初始化分区。

如果是通□网□安装， bsdinstall 将根据之前所□的□件下□□□的文件。



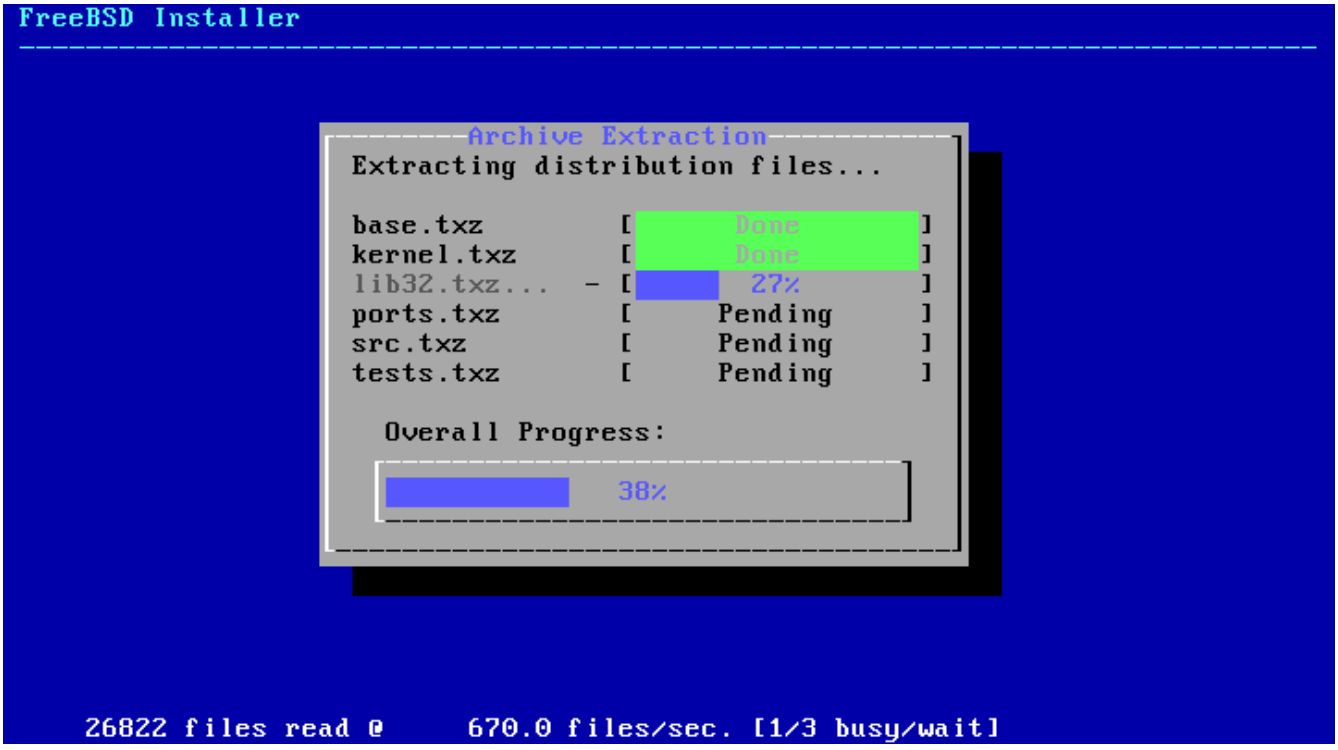
□ *75.* □取□件□□的文件

接下来， 会□□□些文件的完整性， 以防止其在下□□□坏或从安装介□中□□。



□ *76.* □□□件□□的文件

最后， □□□的□件文件会被提取至磁□。



□ *77.* 提取□件□□的文件

文件提取全部完成后， bsdinstall 将□始安装后的配置任□（参□ 安装后的配置）。

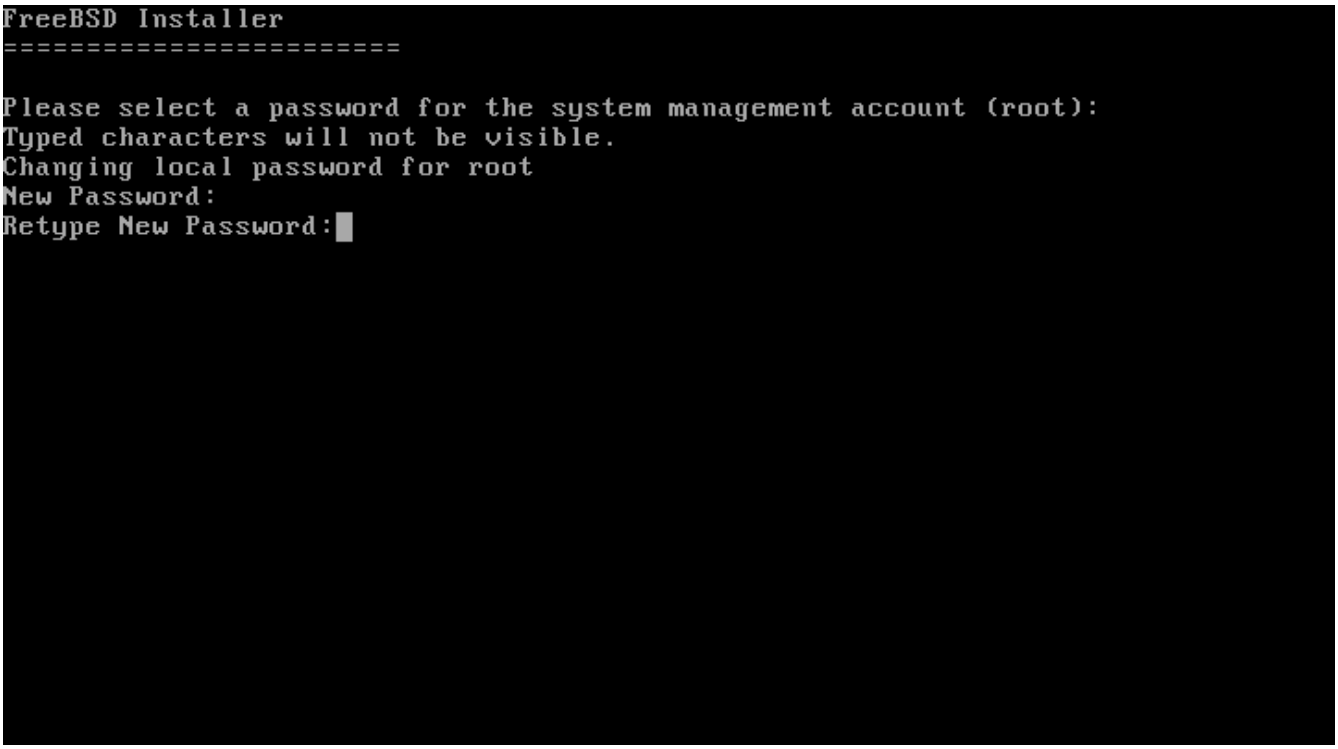
3.9. 安装后的配置

成功安装 FreeBSD 后， □需要依次□行一些配置。 在重□□入新系□前， □些配置始□可以通□最□的配置菜

□□行修改。

3.9.1. □置 root 密□

必□□置 root 密□。 □注意□入密□□， 被□入的字符并不会在屏幕上□示， 因此□防止□入□□， 必□再次□入相同的字符。



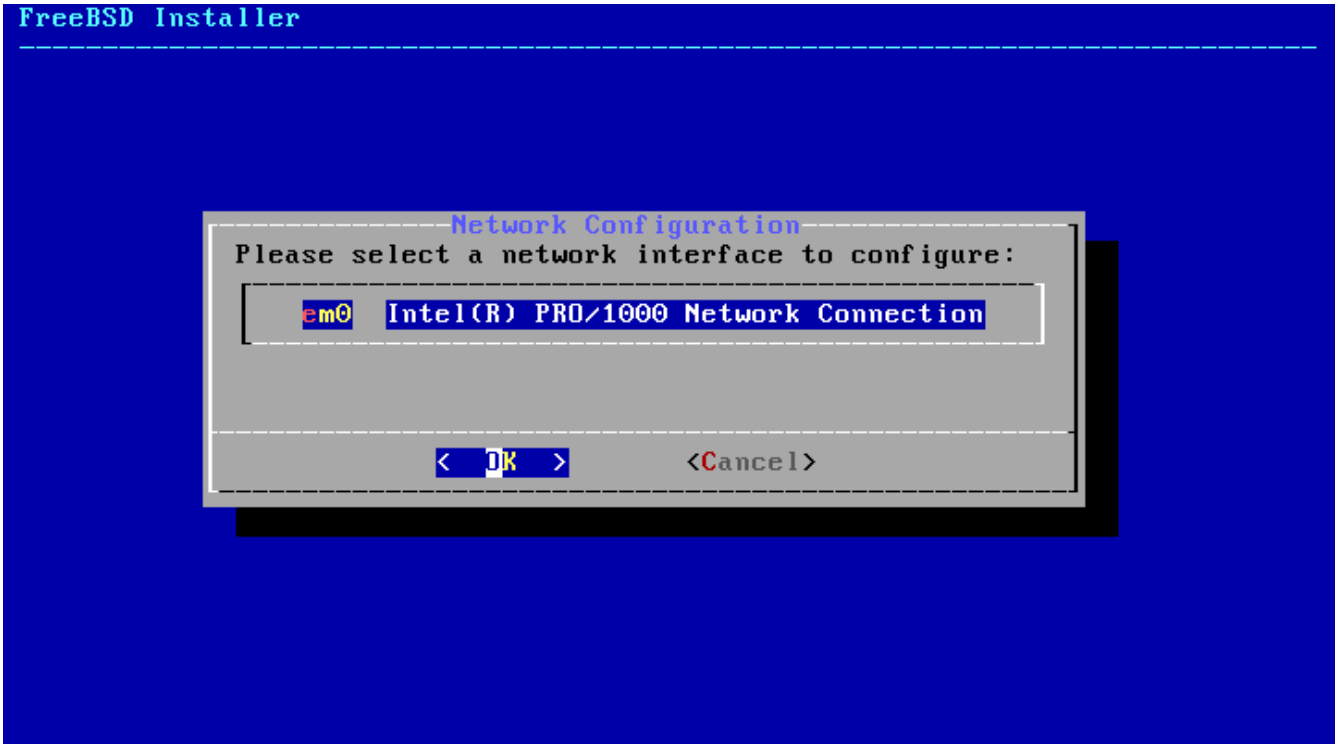
□ *78.* □置 root 密□

成功□置密□后， 安装将□□□行。

3.9.2. 配置网□接口

如果已□在 *bootonly* 安装□配置□网□接口， □可略□此□。

□里将□示一个网□接口列表， 其中的接口都是在当前□算机上□□到的， □□□一个□行配置。



□ *79.* □□一个网□接口

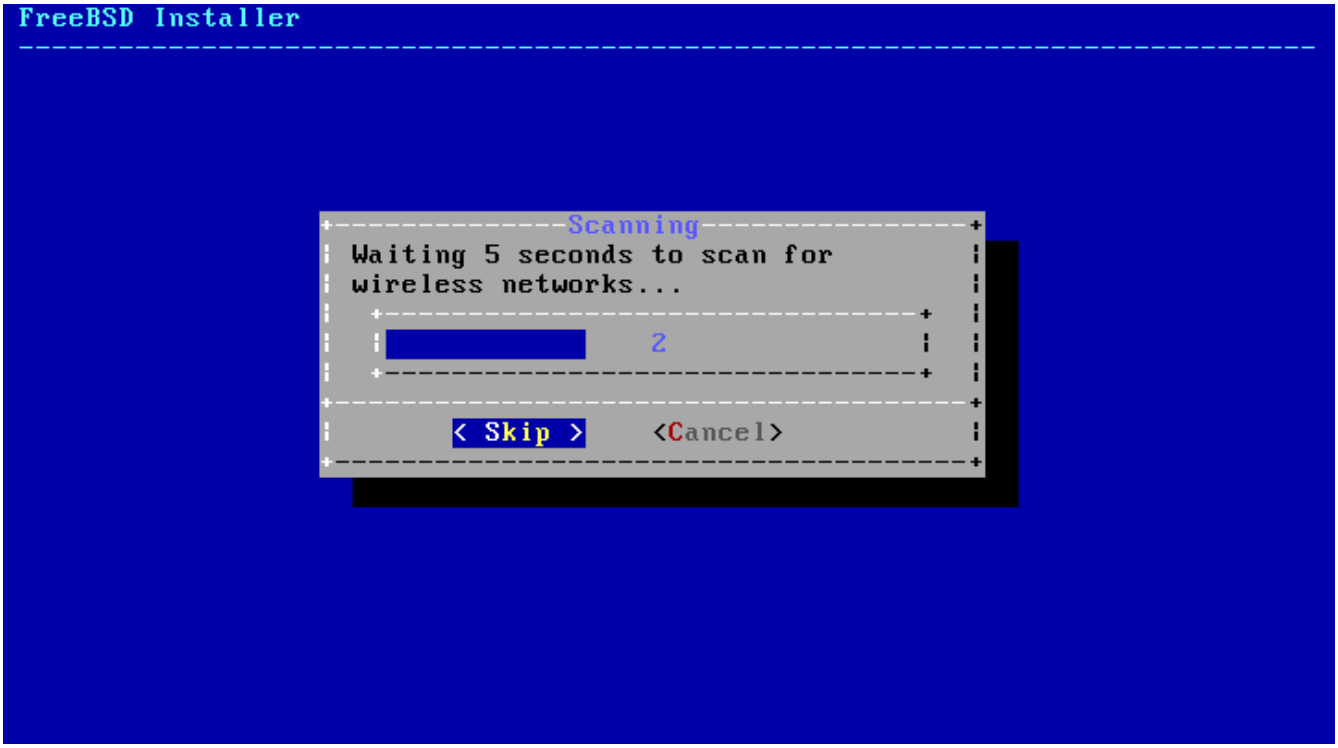
3.9.2.1. 配置无□网□接口

如果□□了无□网□接口， □必□□入相□的无□网□□□及安全参数， 以允□其□接至特定的网□。

无□网□是通□ Service Set Identifier（服□集□□符， □写□ SSID）来表示的， 它是唯一表示无□网□的短字符串。

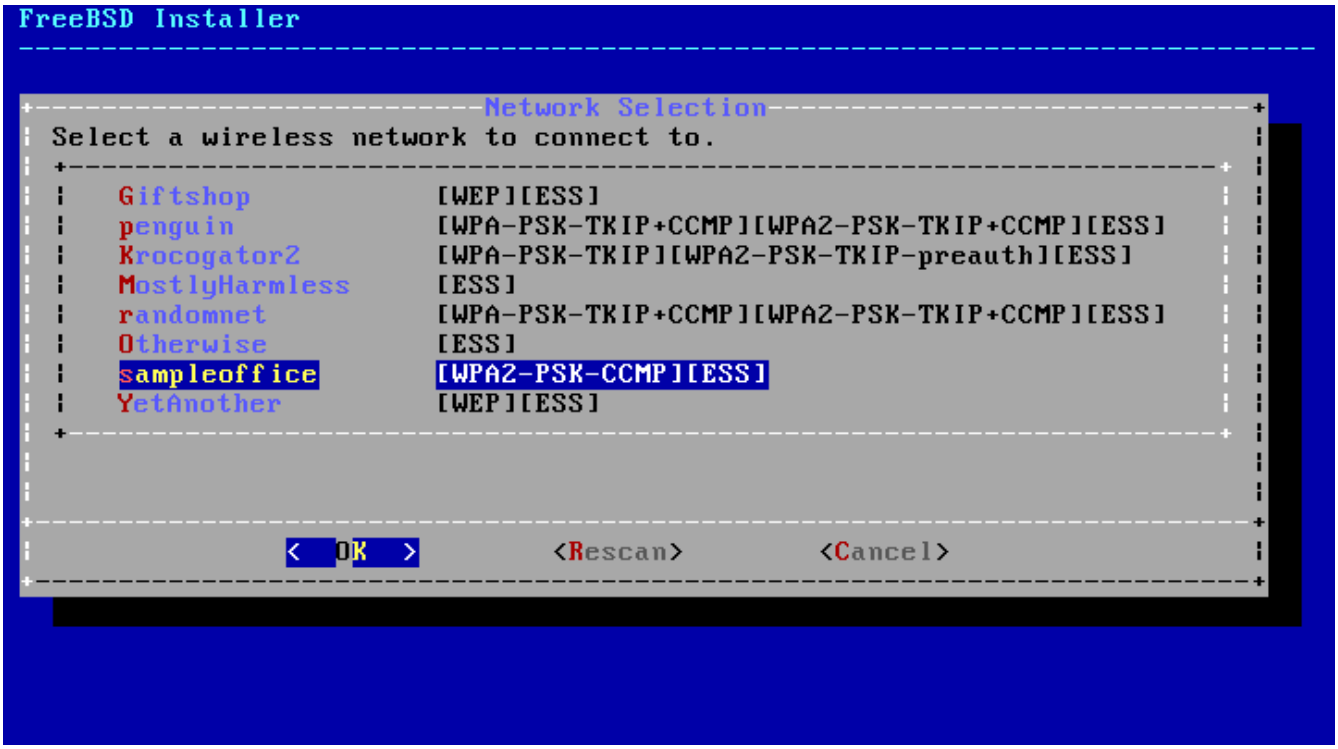
大多数无□网□都会以加密方式□□数据， 藉此保□信息不被未□授□者□看。 □烈建□采用 WPA2 加密。旧式的加密□型， 如 WEP， 几乎没有任何安全性可言。

若要□接至一个无□网□， 首先需要□描无□接入点。



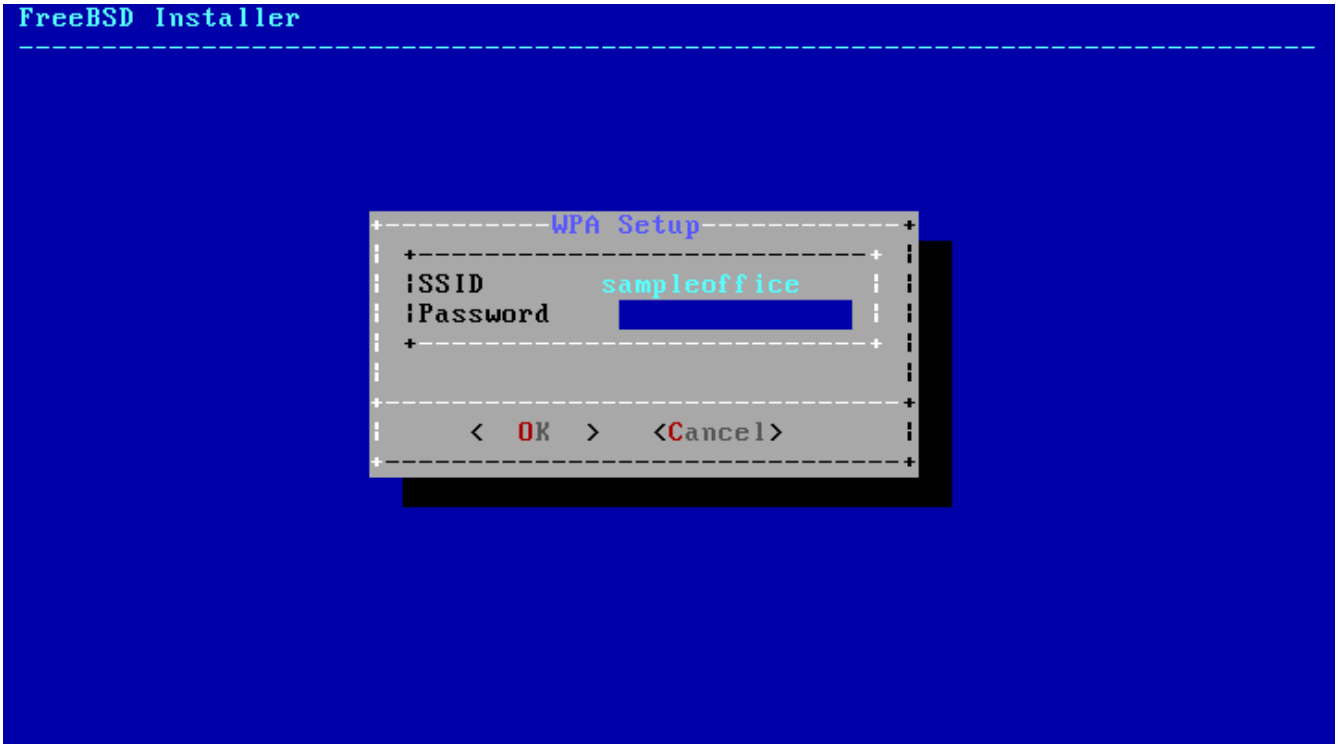
□ *80.* □描无□接入点

□描完成后， 会列出所有□□的 SSID 以及它□支持的加密□型□明。 如果需要□接的 SSID 没有列出， □□□ **[Rescan]** 再次□描。 如果□没有出□， □□□天□， 或将□算机移至更□近接入点的地方。 在做□□些改善措施之后， 再重新□描。



□ *81.* □□一个无□网□

□□所要□接的无□网□， 即可□入□接所需的加密信息。 □于 WPA2， 只需□入一个密□ （也叫□共享密□，(□称 PSK）。 □安全起□， 在□入框中□入的字符将□示□星号。



□ *82. WPA2 □置*

在□□了无□网□并□入了□接所需的信息后， 网□配置将□□□行。

3.9.2.2. 配置 IPv4 网□

□□是否使用 IPv4 网□。 □是最常□的网□□接□型。

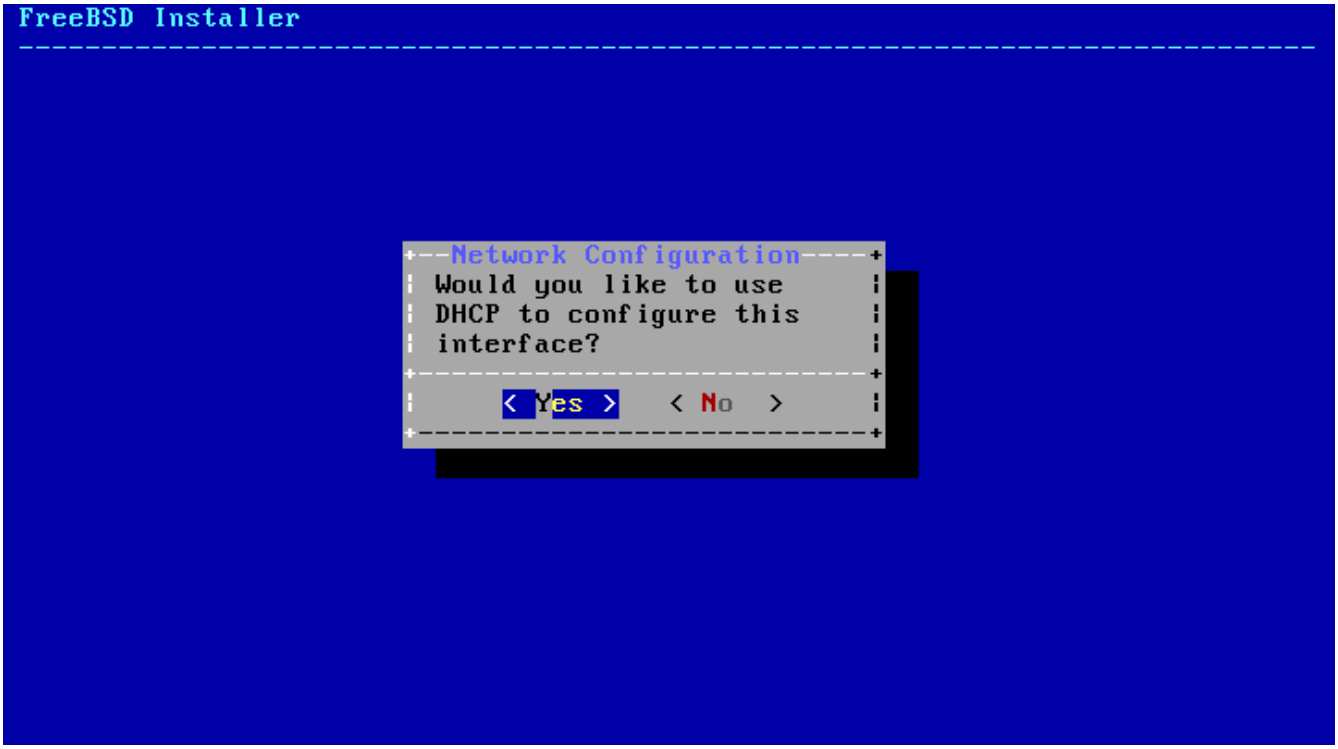
□ *83. □□ IPv4 网□*

有□□配置 IPv4 的方式。 *DHCP* 会自□地□网□接口□行正□的配置， 通常情况下， □是首□的方式。 而 *Static*（静□） 方式□需要手工□入网□的配置信息。

不要随意□入网□的配置信息，　　因□□□的□网□就无法正常工作。　　□向网□管理□或服□提供商那里取得 收集网□配置信息 所列出的配置信息。

3.9.2.2.1. 使用 DHCP 方式

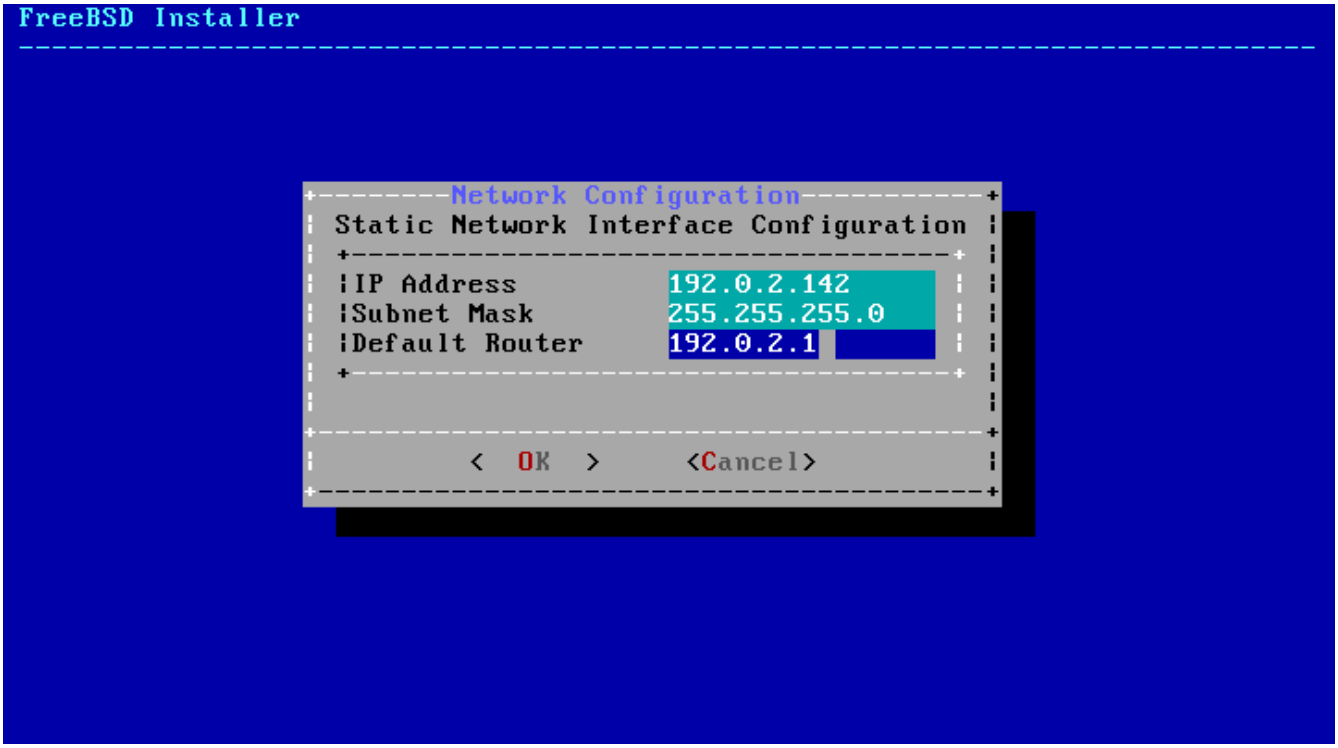
若存在可用的 DHCP 服□器，□□□ **[Yes]** 以自□配置网□接口。



□ *84.* □□ *DHCP* 配置 *IPv4*

3.9.2.2.2. 使用静□配置方式

网□接口的静□配置需要□入相□的 IPv4 配置信息。

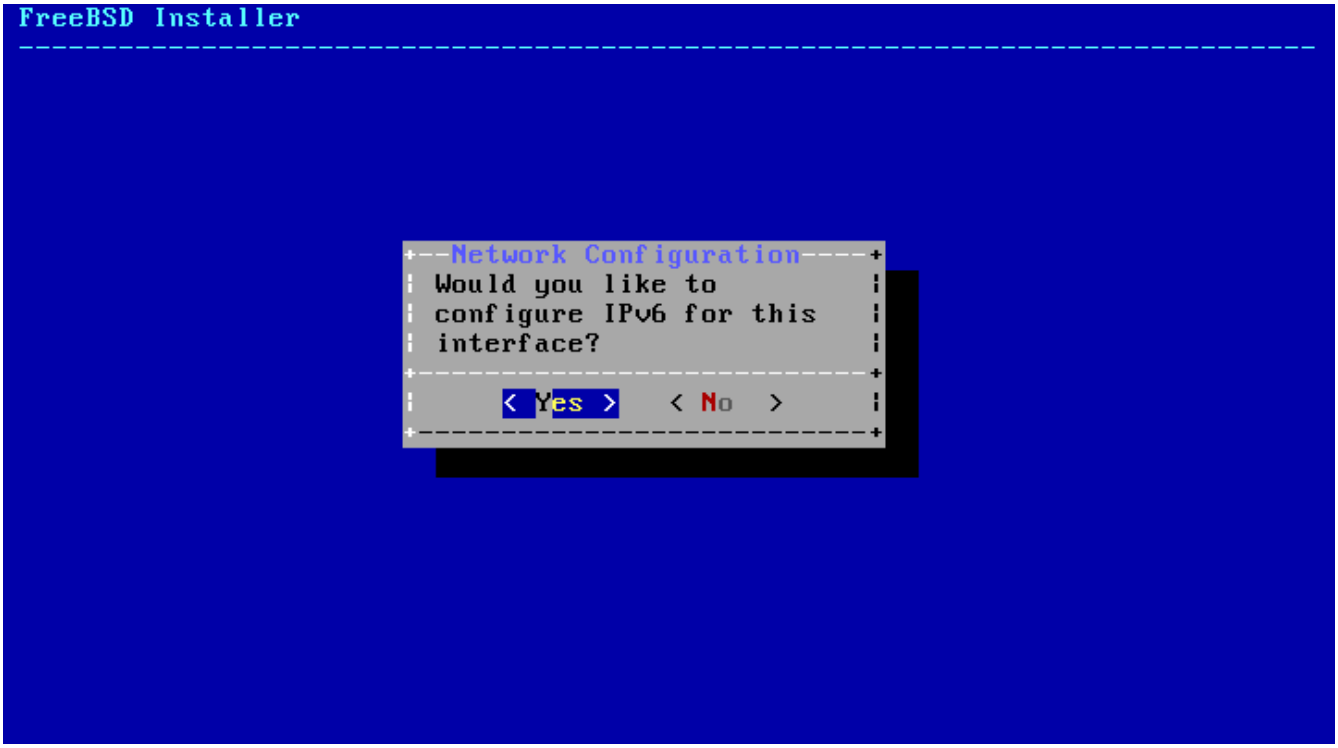


□ *85.* 静□配置 *IPv4*

- IP Address - IP 地址， 即□当前□算机手□分配的 IPv4 地址。 此地址必□是唯一的， 并且在本地网□上□没有被其他□□使用。
- Subnet Mask - 子网掩□， 用于本地网□。 通常是 255.255.255.0。
- Default Router（默□路由） - 网□上默□路由的 IP 地址。 通常， □是将本地网□□接至 Internet 的路由器或其他网□□□的地址。 也称作 *default gateway* （默□网□）。

3.9.2.3. 配置 IPv6 网□

IPv6 是一□新的网□配置方式。 如果□有可用的 IPv6 □接， 并需要使用它， □□ **[Yes]** 来□始配置。

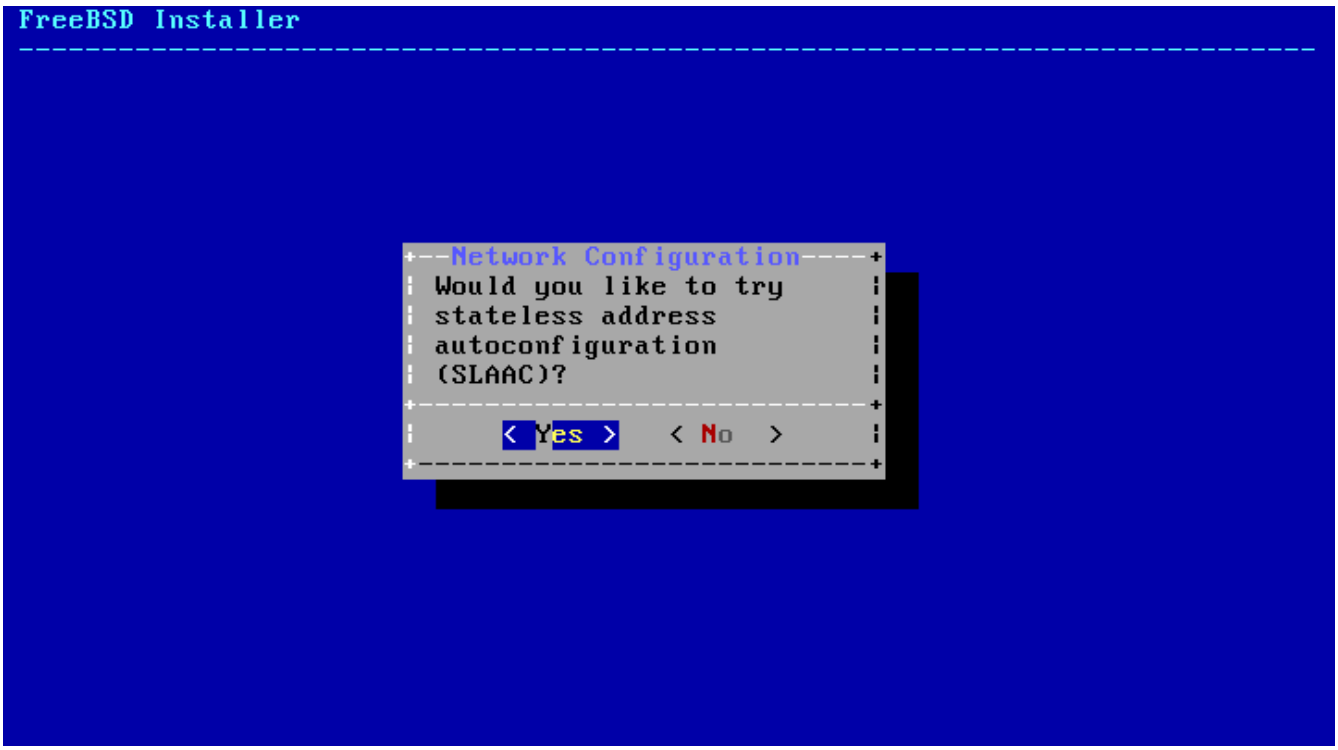


□ *86.* □□ *IPv6* 网□

IPv6 也有□□配置方式。 *SLAAC*， 或 *StateLess Address AutoConfiguration* （无状□地址自□配置）方式能□自□配置正□的网□接口， 而 *Static*（静□） 配置方式□需要手□□入网□信息。

3.9.2.3.1. 使用 Stateless Address Autoconfiguration 方式

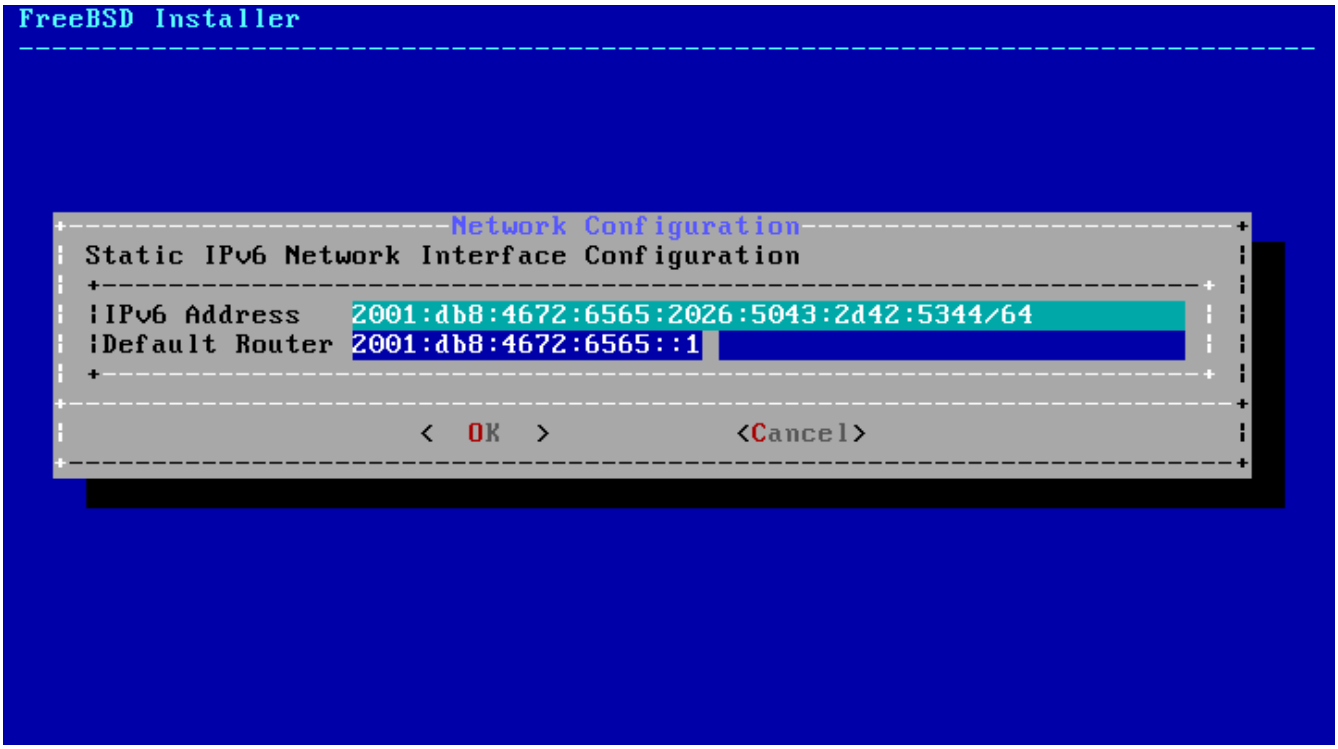
SLAAC 允□ IPv6 □件从本地路由器□求自□配置信息， □情参□ RFC4862。



□ *87.* □□ *SLAAC* 配置 *IPv6*

3.9.2.3.2. 使用静□配置方式

网□接口的静□配置需要□入相□的 IPv6 配置信息。

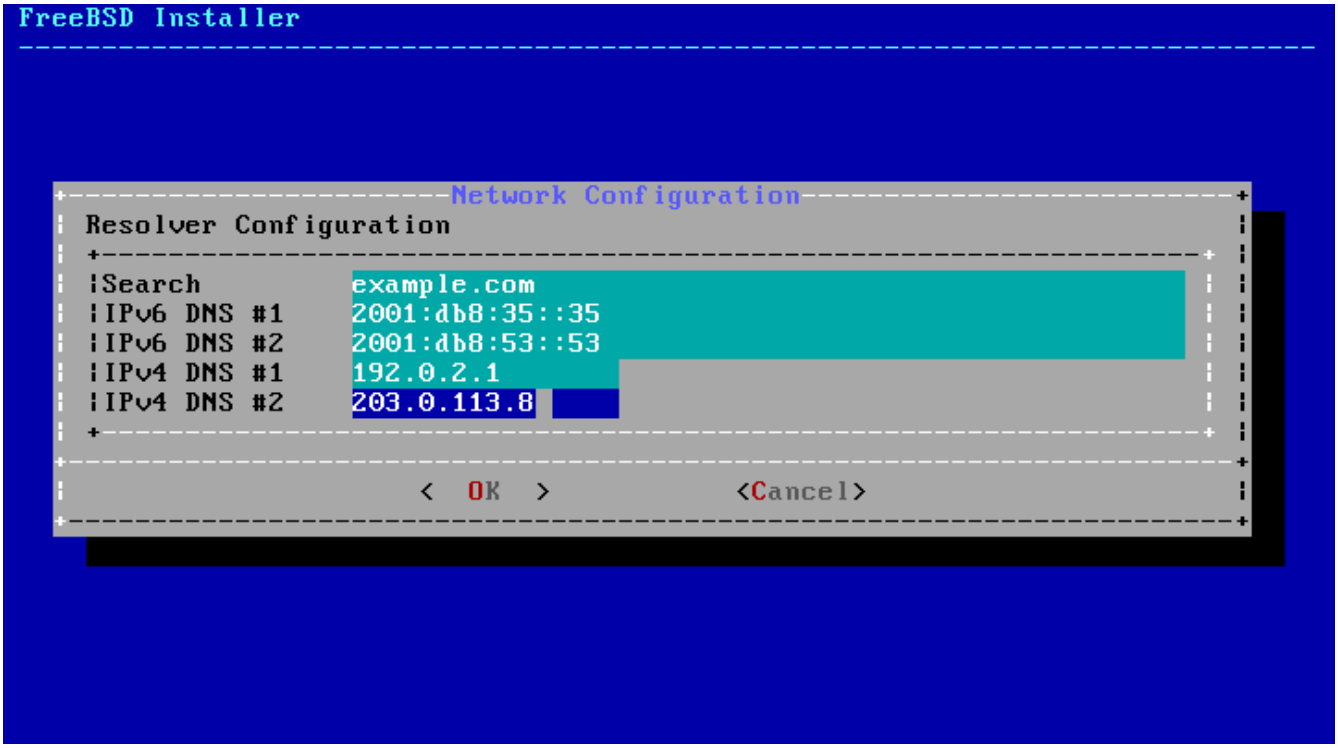


□ *88.* 静□配置 *IPv6*

- `IPv6 Address` （IPv6地址） - □当前□算机手工分配的 IP 地址。 □个地址必□是唯一的，并且没有被其他本地网□□□使用。
- `Default Router` （默□路由） - 网□上默□路由的地址。 通常， □是将本地网□□接至 Internet 的路由器或其他网□□□的地址。 也称作 *default gateway* （默□网□）。

3.9.2.4. 配置 DNS

Domain Name System （域名系□，□称 *DNS*） 解析器用于主机名和网□地址□的相互□□。 如果使用的是 DHCP 或 SLAAC， 那□其配置很可能已□存在； 否□， □在 Search 字段中□入本地网□的域名， 在 DNS #1 和 DNS #2 中□入本地 DNS 服□器的 IP 地址。 至少需要配置一个 DNS 服□器。



□ *89. DNS* 配置

3.9.3. □置□区

□□的机器□置□区将允□其自□校□， 并正□□行一些与□区相□的操作。

示例中的机器位于美国□部□区。 根据所□的地理位置， □的□□可能会有所不同。

□ *90.* □□本地或 *UTC* □□

□□ **[Yes]** 或 **[No]** 以□定机器□□的配置方式， 然后按 `Enter` □。 如果□并不知道系□使用的是 UTC □是本地□□， □□□ **[No]** 以使用更□常□的本地□□。

图 91. □□地区

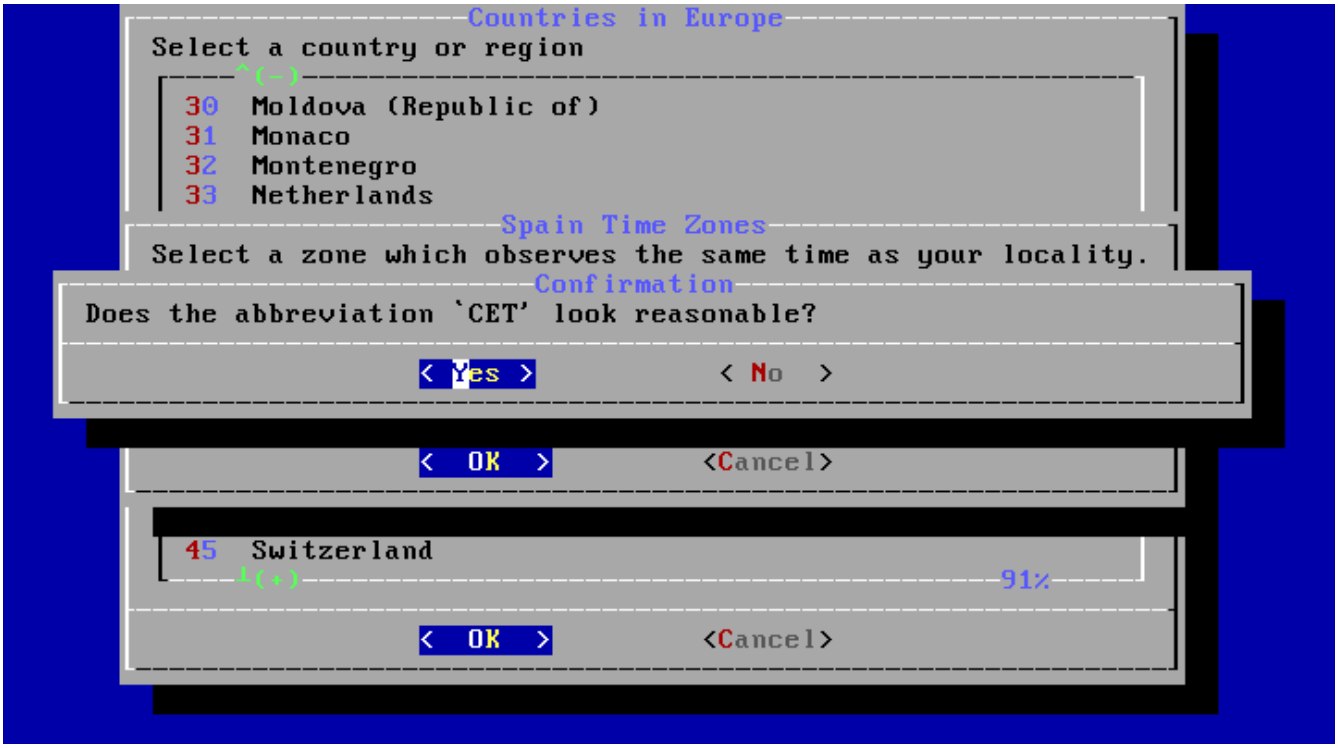
使用方向□□□合□的地区后按下 Enter □。

图 92. □□国家

用方向□□□合□的国家后按下 Enter □。

□ *93.* □□□区

用方向□□□合□的□区后按下 `Enter` □。

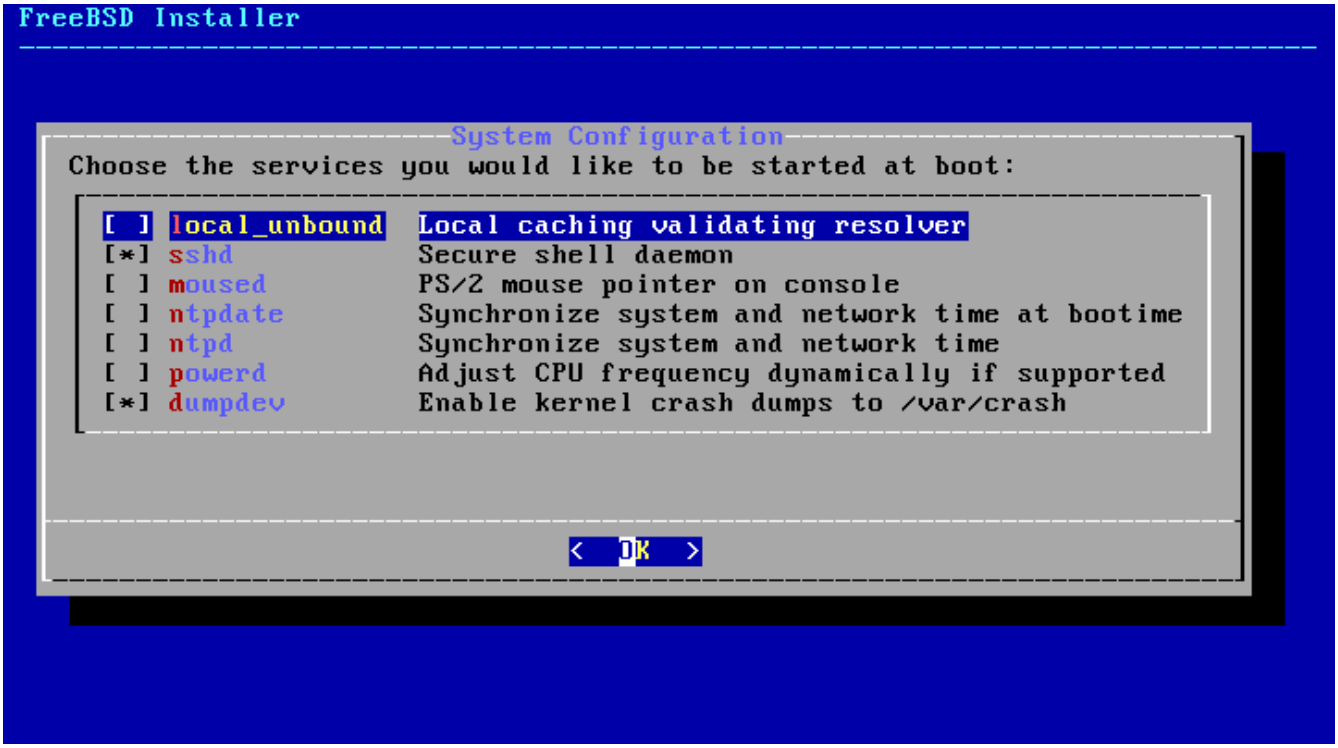


□ *94.* □□□区□□

□□□区的□写是正□的， 然后按 `Enter` □以□□安装后的配置。

3.9.4. □□需要□□的服□

可以□□□外的系□服□， 它□会在系□□□□自□□行。 所有□些服□都是可□的。



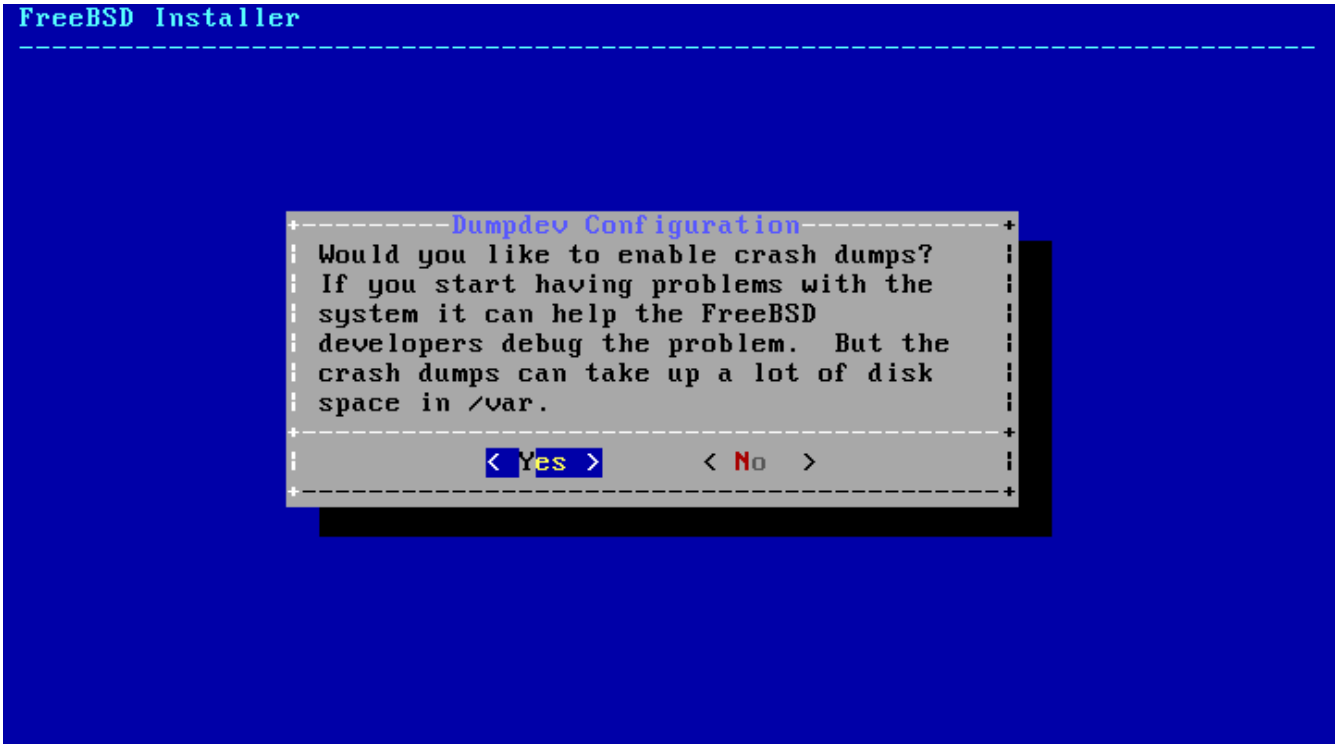
□ *95.* □□需要□□的服□

□外的系□服□

- `sshd` - Secure Shell（即 SSH） 守□□程， 提供安全的□程□□。
- `moused` - 支持在系□控制台中使用鼠□。
- `ntpd` - Network Time Protocol（网□□□□□， □称 NTP） 守□□程， 提供□□自□同□。
- `powerd` - 系□□量控制程序， 用于控制□量及□能。

3.9.5. □用崩□□□

bsdinstall 将□□是否在目□系□上□用崩□□□。 由于在□□系□□非常有用， 因此鼓励用□尽可能地□用崩□□□。 □□ **[Yes]** 以□用崩□□□， 或□□ **[No]** 以不□用崩□□□。

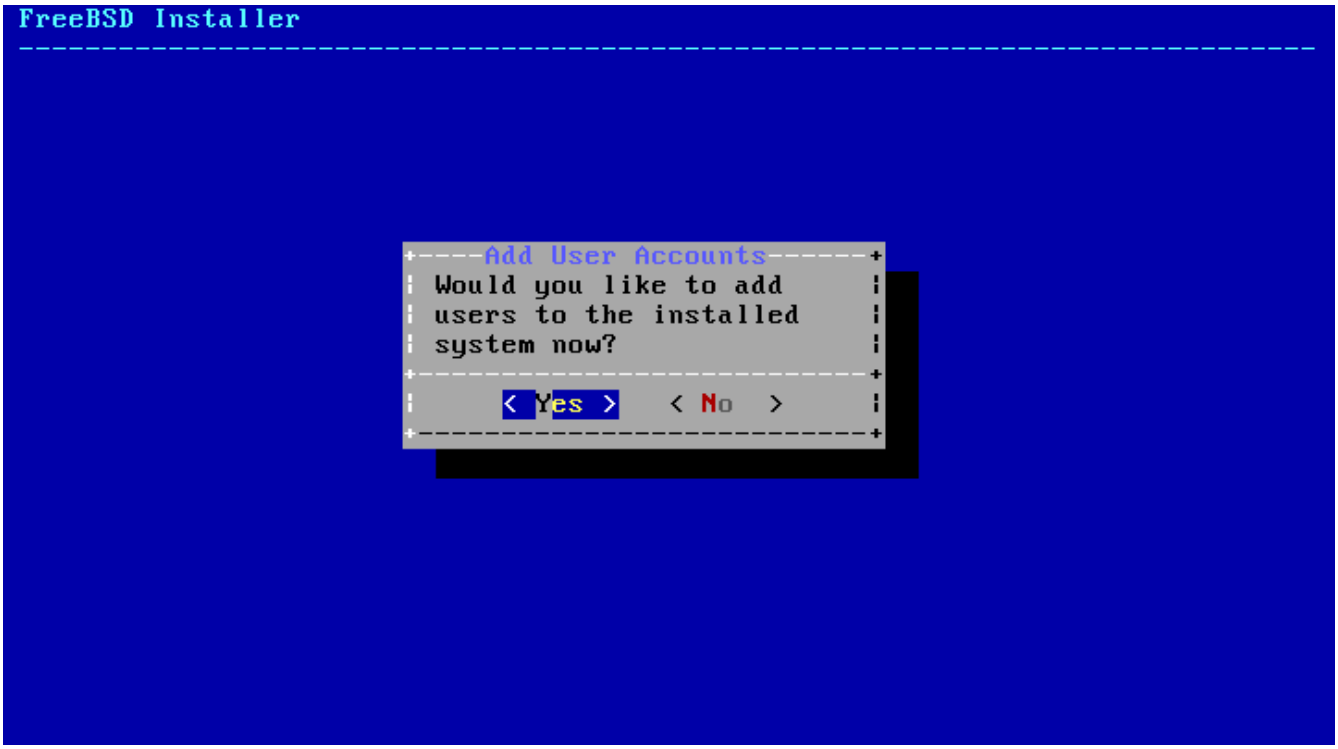


96. 用崩

3.9.6. 添加用

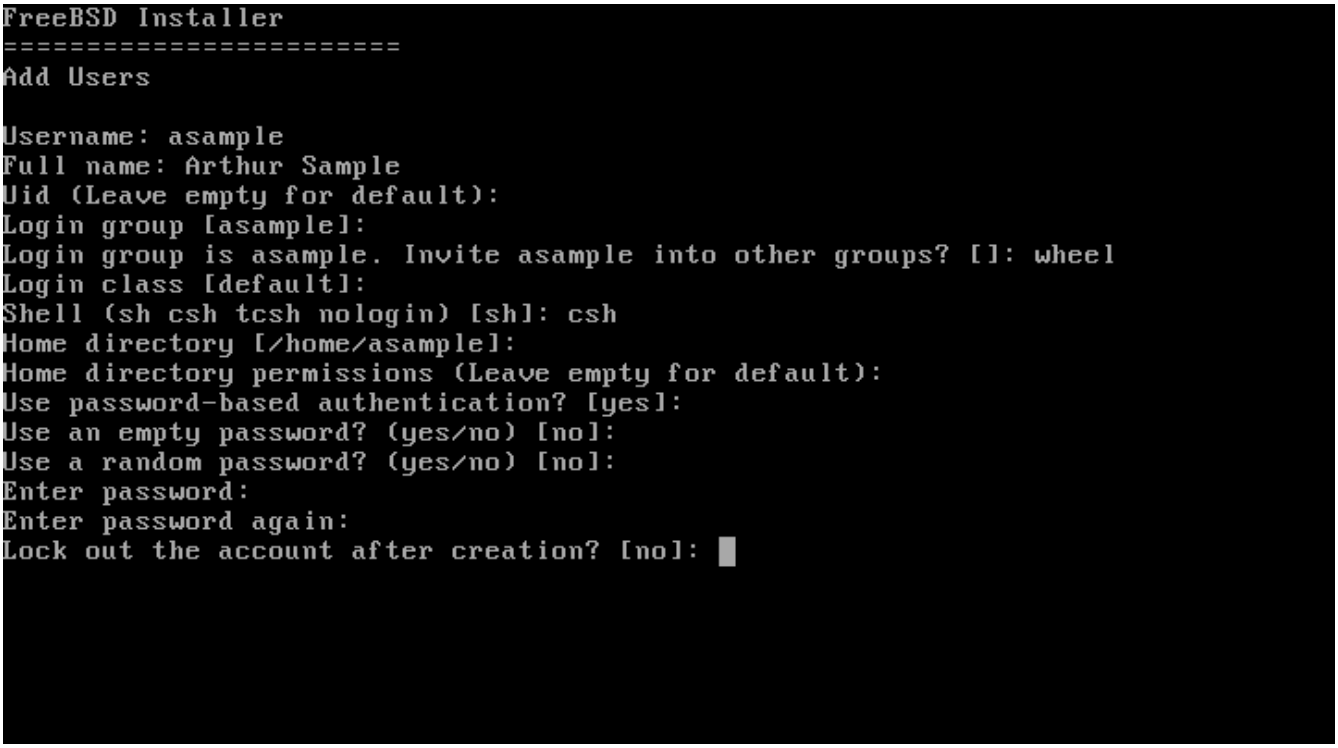
在安装程中， 至少添加一位普通用， 而不要始以 `root` 身登入。 当以 `root` 身登入系， 系几乎不会其操作提供任何限制或保。 以普通用身登更安全。

[Yes] 来添加新用。



97. 添加用号

需要添加的用入信息。



□ *98.* □入用□信息

用□信息

- Username - 用□名， 即登入□用□所□入的名称。 通常是名的首字母加姓的□合。
- Full name - 用□的全名。
- Uid - 用□ ID。 通常留空以自□分配。
- Login group - 用□□。 通常留空以接受默□取□。
- Invite user into other groups? - 是否同□将用□加入其他□限□？ 如果需要， □□入□限□名称。
- Login class - 登□□□。 通常留空以接受默□取□。
- Shell - 用□ shell。 在本例中□□的是 csh(1)。
- Home directory - 用□主目□。 通常留空以接受默□取□。
- Home directory permissions - 用□主目□的□限。 通常留空以接受默□取□。
- Use password-based authentication? - 是否使用基于密□的□□？ 通常□ “yes”。
- Use an empty password? - 是否使用空密□？ 通常□ “no”。
- Use a random password? - 是否使用随机密□？ 通常□ “no”。
- Enter password - 用□的□□密□。 □入的字符不会在屏幕上□示。
- Enter password again - 必□再次□入密□以□行□□。
- Lock out the account after creation? - □建后□定□号？ 通常□ “no”。

全部信息□入完成后， 系□会□示摘要并□□是否正□。 如果□□了□□， 可以□入 no 后□行修改； 如果没有□□， □□入 yes 以□建新用□。

```
Login group [asample]:
Login group is asample. Invite asample into other groups? []: wheel
Login class [default]:
Shell (sh csh tcsh nologin) [sh]: csh
Home directory [/home/asample]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]:
Enter password:
Enter password again:
Lock out the account after creation? [no]:
Username   : asample
Password   : *****
Full Name  : Arthur Sample
Uid        : 1001
Class      :
Groups     : asample wheel
Home       : /home/asample
Home Mode  :
Shell      : /bin/csh
Locked     : no
OK? (yes/no): yes
adduser: INFO: Successfully added (asample) to the user database.
Add another user? (yes/no):
```

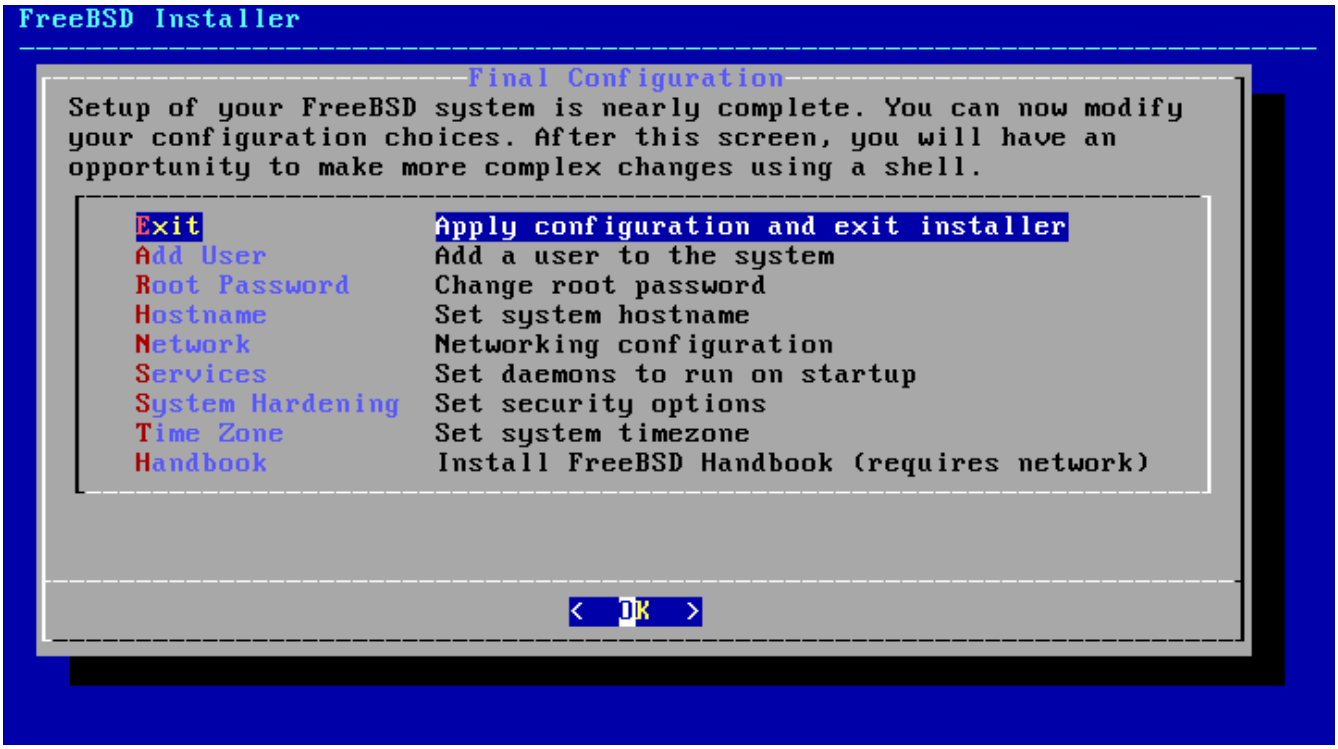
99. 退出用与管理

若需添加更多用，在"`Add another user?"后入 `yes`；入 no 以完成用添加并安装。

更多有用添加及管理的信息，参 用和基本的管理。

3.9.7. 最配置

所有的安装及配置完成后，仍有机会其行修改。



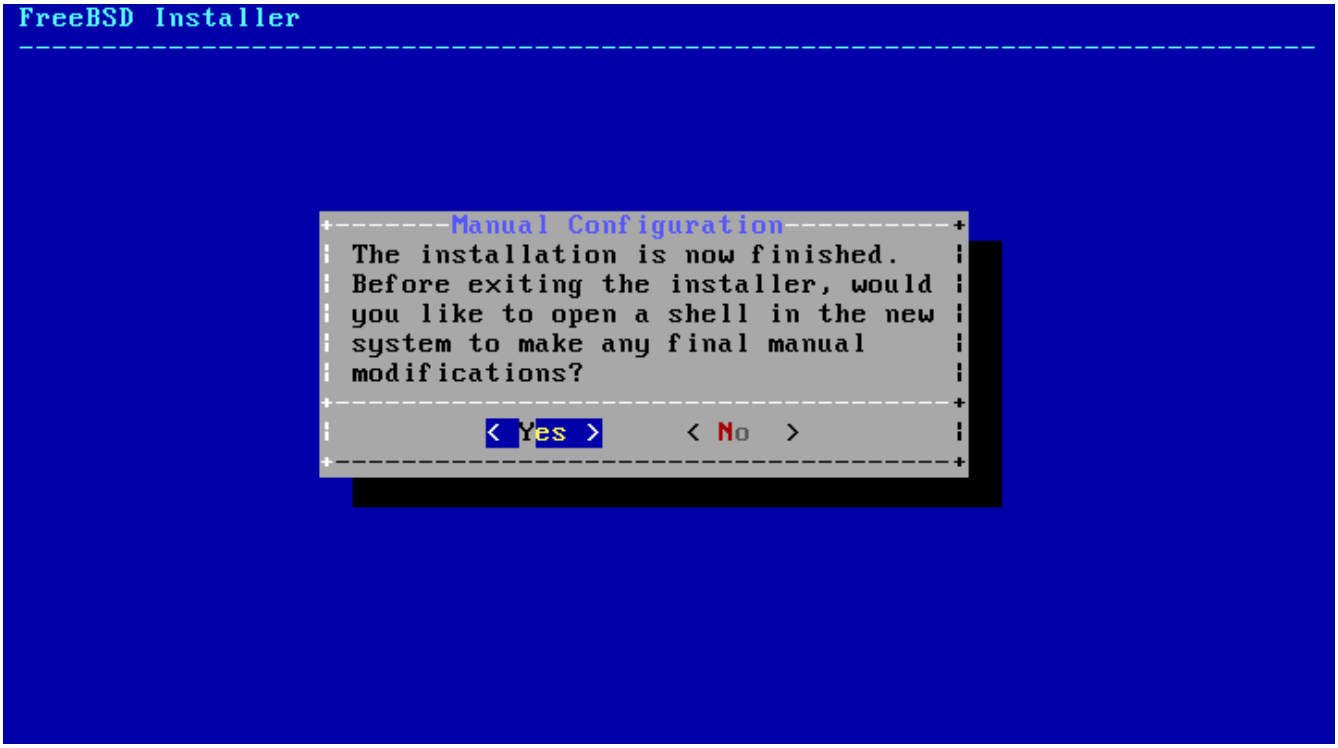
100. 最的配置菜

使用此菜，可以在完成安装前添加或修改任何配置。

最□的配置□□

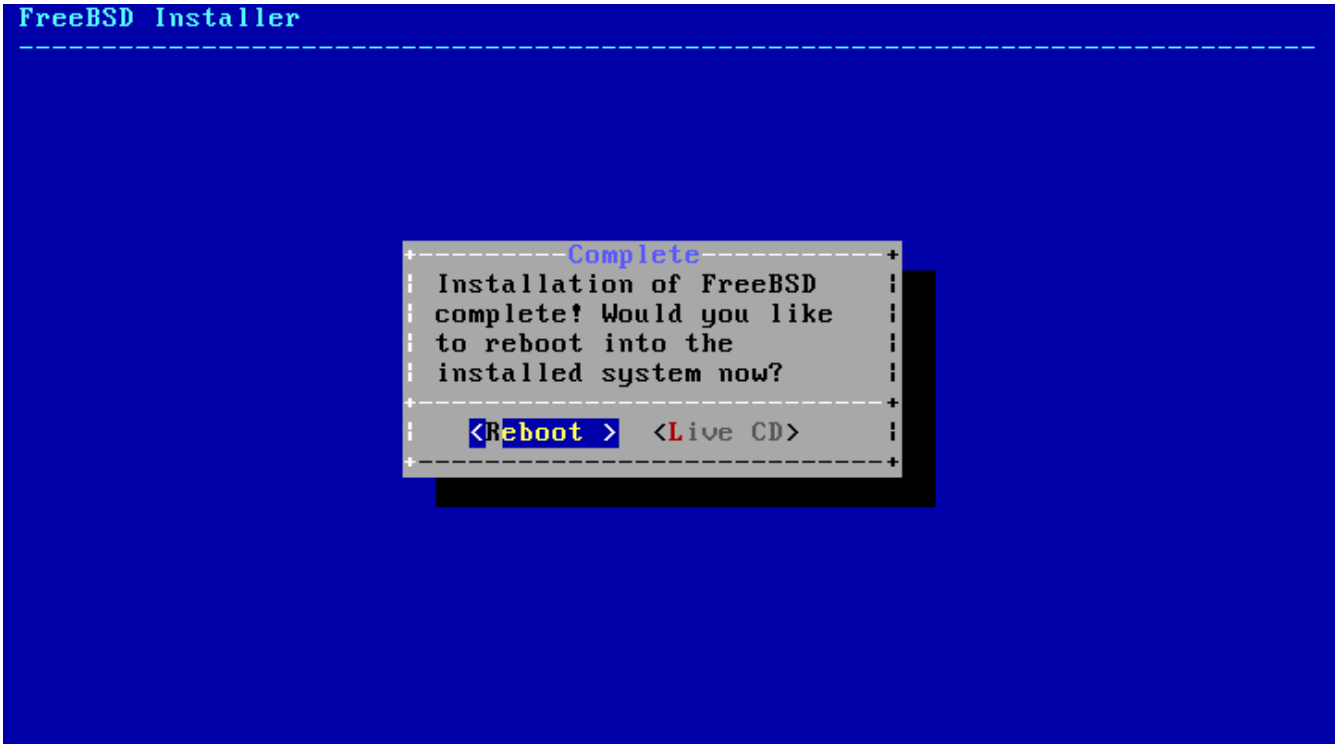
- Add User - 添加用□， □□ 添加用□。
- Root Password - root 密□， □□ □置 root 密□。
- Hostname - 主机名， □□ □置主机名。
- Network - 网□， □□ 配置网□接□。
- Services - 服□， □□ □□需要□□的服□.
- Time Zone - □区， □□ □置□区。
- Handbook - 手册，将下□并安装 FreeBSD 使用手册（即本□）。

完成了最□配置后， □□□ **[Exit]** 以□□安装。



□ *101.* 手□配置

bsdinstall 会□□重□前是否□需要□外的配置： □□ **[Yes]** □入 shell 做□些配置， □□ **[No]** 以□行安装的最后一□。



□ *102.* 完成安装

如果需要□一□的配置或特殊的□置， 可以□□ **[Live CD]** 来□入安装介□的 Live CD 模式。

安装完成后，□□ **[Reboot]** 重□□算机， 并□始使用全新的 FreeBSD 系□。 □不要忘□移除 FreeBSD 的安装 CD、 DVD 或 USB □□棒， 否□□算机可能会再次从□些介□□□。

3.9.8. FreeBSD 的□□与□□

3.9.8.1. FreeBSD/i386 的□□

FreeBSD □□□会□示□多相□信息， 正常情况下屏幕会不断□□， 而□□完成后□会□示一个登□提示符。 如果需要□看□□□的相□信息， 可以按下 `Scroll-Lock` □□□ *scroll-back buffer*（回□□存）， 然后使用 `PageUp` □、 `PageDown` □与方向□行翻□； 再次按下 `Scroll Lock` □将□□回□□存并返回正常的屏幕。

在 `login:` 提示符□□入安装□添加的用□名来登□系□， 本例中是 `asample`。 除非有必要，否□□勿作□ `root` 登□。

上述的回□□存大小有限， 因而未必全部可□。 登入系□后， 在提示符□□入 `dmesg | less`， 能□□看到□大部分的□□信息， □看后按 `q` □返回命令行。

典型的□□信息（此□略去了版本信息）：

```
Copyright (c) 1992-2011 The FreeBSD Project.
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1989, 1991, 1992, 1993, 1994
        The Regents of the University of California. All rights reserved.
FreeBSD is a registered trademark of The FreeBSD Foundation.

    root@farrell.cse.buffalo.edu:/usr/obj/usr/src/sys/GENERIC amd64
CPU: Intel(R) Core(TM)2 Duo CPU     E8400  @ 3.00GHz (3007.77-MHz K8-class CPU)
  Origin = "GenuineIntel"  Id = 0x10676  Family = 6  Model = 17  Stepping = 6
  Features
```

```
=0x783fbff<FPU,VME,DE,PSE,TSC,MSR,PAE,MCE,CX8,APIC,SEP,MTRR,PGE,MCA,CMOV,PAT,PSE36,MMX
,FXSR,SSE,SSE2>
  Features2=0x209<SSE3,MON,SSSE3>
  AMD Features=0x20100800<SYSCALL,NX,LM>
  AMD Features2=0x1<LAHF>
real memory  = 536805376 (511 MB)
avail memory = 491819008 (469 MB)
Event timer "LAPIC" quality 400
ACPI APIC Table: <VBOX   VBOXAPIC>
ioapic0: Changing APIC ID to 1
ioapic0 <Version 1.1> irqs 0-23 on motherboard
kbd1 at kbdmux0
acpi0: <VBOX VBOXXSDT> on motherboard
acpi0: Power Button (fixed)
acpi0: Sleep Button (fixed)
Timecounter "ACPI-fast" frequency 3579545 Hz quality 900
acpi_timer0: <32-bit timer at 3.579545MHz> port 0x4008-0x400b on acpi0
cpu0: <ACPI CPU> on acpi0
pcib0: <ACPI Host-PCI bridge> port 0xcf8-0xcff on acpi0
pci0: <ACPI PCI bus> on pcib0
isab0: <PCI-ISA bridge> at device 1.0 on pci0
isa0: <ISA bus> on isab0
atapci0: <Intel PIIX4 UDMA33 controller> port 0x1f0-0x1f7,0x3f6,0x170-
0x177,0x376,0xd000-0xd00f at device 1.1 on pci0
ata0: <ATA channel 0> on atapci0
ata1: <ATA channel 1> on atapci0
vgapci0: <VGA-compatible display> mem 0xe0000000-0xe0ffffff irq 18 at device 2.0 on
pci0
em0: <Intel(R) PRO/1000 Legacy Network Connection 1.0.3> port 0xd010-0xd017 mem
0xf0000000-0xf001ffff irq 19 at device 3.0 on pci0
em0: Ethernet address: 08:00:27:9f:e0:92
pci0: <base peripheral> at device 4.0 (no driver attached)
pcm0: <Intel ICH (82801AA)> port 0xd100-0xd1ff,0xd200-0xd23f irq 21 at device 5.0 on
pci0
pcm0: <SigmaTel STAC9700/83/84 AC97 Codec>
ohci0: <OHCI (generic) USB controller> mem 0xf0804000-0xf0804fff irq 22 at device 6.0
on pci0
usbus0: <OHCI (generic) USB controller> on ohci0
pci0: <bridge> at device 7.0 (no driver attached)
acpi_acad0: <AC Adapter> on acpi0
atkbdc0: <Keyboard controller (i8042)> port 0x60,0x64 irq 1 on acpi0
atkbd0: <AT Keyboard> irq 1 on atkbdc0
kbd0 at atkbd0
atkbd0: [GIANT-LOCKED]
psm0: <PS/2 Mouse> irq 12 on atkbdc0
psm0: [GIANT-LOCKED]
psm0: model IntelliMouse Explorer, device ID 4
attimer0: <AT timer> port 0x40-0x43,0x50-0x53 on acpi0
Timecounter "i8254" frequency 1193182 Hz quality 0
Event timer "i8254" frequency 1193182 Hz quality 100
sc0: <System console> at flags 0x100 on isa0
```

```
sc0: VGA <16 virtual consoles, flags=0x300>
vga0: <Generic ISA VGA> at port 0x3c0-0x3df iomem 0xa0000-0xbffff on isa0
atrtc0: <AT realtime clock> at port 0x70 irq 8 on isa0
Event timer "RTC" frequency 32768 Hz quality 0
ppc0: cannot reserve I/O port range
Timecounters tick every 10.000 msec
pcm0: measured ac97 link rate at 485193 Hz
em0: link state changed to UP
usbus0: 12Mbps Full Speed USB v1.0
ugen0.1: <Apple> at usbus0
uhub0: <Apple OHCI root HUB, class 9/0, rev 1.00/1.00, addr 1> on usbus0
cd0 at ata1 bus 0 scbus1 target 0 lun 0
cd0: <VBOX CD-ROM 1.0> Removable CD-ROM SCSI-0 device
cd0: 33.300MB/s transfers (UDMA2, ATAPI 12bytes, PIO 65534bytes)
cd0: Attempt to query device size failed: NOT READY, Medium not present
ada0 at ata0 bus 0 scbus0 target 0 lun 0
ada0: <VBOX HARDDISK 1.0> ATA-6 device
ada0: 33.300MB/s transfers (UDMA2, PIO 65536bytes)
ada0: 12546MB (25694208 512 byte sectors: 16H 63S/T 16383C)
ada0: Previously was known as ad0
Timecounter "TSC" frequency 3007772192 Hz quality 800
Root mount waiting for: usbus0
uhub0: 8 ports with 8 removable, self powered
Trying to mount root from ufs:/dev/ada0p2 [rw]...
Setting hostuuid: 1848d7bf-e6a4-4ed4-b782-bd3f1685d551.
Setting hostid: 0xa03479b2.
Entropy harvesting: interrupts ethernet point_to_point kickstart.
Starting file system checks:
/dev/ada0p2: FILE SYSTEM CLEAN; SKIPPING CHECKS
/dev/ada0p2: clean, 2620402 free (714 frags, 327461 blocks, 0.0% fragmentation)
Mounting local file systems:.
vboxguest0 port 0xd020-0xd03f mem 0xf0400000-0xf07fffff,0xf0800000-0xf0803fff irq 20
at device 4.0 on pci0
vboxguest: loaded successfully
Setting hostname: machine3.example.com.
Starting Network: lo0 em0.
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
        options=3<RXCSUM,TXCSUM>
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x3
        inet 127.0.0.1 netmask 0xff000000
        nd6 options=21<PERFORMNUD,AUTO_LINKLOCAL>
em0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=9b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM>
        ether 08:00:27:9f:e0:92
        nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
        media: Ethernet autoselect (1000baseT <full-duplex>)
        status: active
Starting devd.
Starting Network: usbus0.
DHCPREQUEST on em0 to 255.255.255.255 port 67
```

```
DHCPACK from 10.0.2.2
bound to 192.168.1.142 -- renewal in 43200 seconds.
add net ::ffff:0.0.0.0: gateway ::1
add net ::0.0.0.0: gateway ::1
add net fe80::: gateway ::1
add net ff02::: gateway ::1
ELF ldconfig path: /lib /usr/lib /usr/lib/compat /usr/local/lib
32-bit compatibility ldconfig path: /usr/lib32
Creating and/or trimming log files.
Starting syslogd.
No core dumps found.
Clearing /tmp (X related).
Updating motd:.
Configuring syscons: blanktime.
Generating public/private rsa1 key pair.
Your identification has been saved in /etc/ssh/ssh_host_key.
Your public key has been saved in /etc/ssh/ssh_host_key.pub.
The key fingerprint is:
10:a0:f5:af:93:ae:a3:1a:b2:bb:3c:35:d9:5a:b3:f3 root@machine3.example.com
The key's randomart image is:
+--[RSA1 1024]----+
|    o..          |
|   o . .         |
|  .   o          |
|       o         |
|    o   S        |
|   + + o         |
|o . + *          |
|o+ ..+ .         |
|==o..o+E         |
+-----------------+
Generating public/private dsa key pair.
Your identification has been saved in /etc/ssh/ssh_host_dsa_key.
Your public key has been saved in /etc/ssh/ssh_host_dsa_key.pub.
The key fingerprint is:
7e:1c:ce:dc:8a:3a:18:13:5b:34:b5:cf:d9:d1:47:b2 root@machine3.example.com
The key's randomart image is:
+--[ DSA 1024]----+
|       ..     . .|
|      o  .   . + |
|     . ..   . E .|
|    . .  o o . . |
|     +  S = .    |
|    +  . = o     |
|     +  . * .    |
|    . .  o .     |
|      .o. .      |
+-----------------+
Starting sshd.
Starting cron.
Starting background file system checks in 60 seconds.
```

```
Thu Oct  6 19:15:31 MDT 2011

FreeBSD/amd64 (machine3.example.com) (ttyv0)

login:
```

在□慢的机器上， 生成 RSA 和 DSA 密□可能需要一些□□。 □□情况只会在□□了 sshd 的新系□首次□□□□生，之后的□□速度不受影□。

FreeBSD 默□情况下并不会安装□形□境， 但提供了多□不同的□□。 □参□ X Window 系□ 了解□情。

3.9.9. □□ FreeBSD

正常□□ FreeBSD 有助于保□数据及系□硬件不受□坏。 不要直接□□□源。 如果用□是 `wheel` □的成□，首先在命令行中□入 `su` 后□入 `root` 密□成□超□用□。 此外， 也可作□ `root` 登□， 然后使用命令 `shutdown -p now`。 □□系□将安全地自行□□。

□然也可以使用□合□ `Ctrl` + `Alt` + `Del` 重□系□， 但正常情况下并不推□□□做。

3.10. 故障排除

下面将介□如何排除基本的安装故障， 例如用□□常□告的□□。

3.10.1. 遇到□□□□如何□理

由于 PC 架□的各□限制， 硬件□□不可能 100% 地可□探□， 然而， 当此□□象□生□， □有可能可以通□一些操作来自行解决它□。

首先□□根据所安装的 FreeBSD 版本核□ 硬件兼容□明 文□， 以□保其支持□的硬件。

如果使用被支持的硬件□仍遇到了死机或其他□□， □□□一个 自定□内核， □□即可□那些 GENERIC 内核中不存在的□□提供支持。 引□□上的内核假定□大多数硬件的 IRQ、 IO 地址和 DMA 通道均□出厂□置，如果□的硬件被重新配置□， 就很可能需要修改内核配置文件并重新□□内核， 以支持 FreeBSD □□□些硬件。

□可能出□一□情况， □□某个不存在的□□会□致□后□其他存在的□□□□失□。 在□□情况下， □□禁止□□引起冲突的□□所□□的□□程序。

有些安装□□可以通□更新硬件固件来避免或改善， 尤其是主板。 主板固件通常被称作 BIOS， 大多数主板和□算机制造商都□有提供升□和相□信息的网站。

制造商通常建□， 除非有□似□□更新□□必要的原因， 否□□避免升□主板 BIOS。 升□□程一旦出□□□， BIOS信息将遭到破坏， 从而□致□算机无法工作。

3.10.2. 故障排除□答

3.10.2.1. 在□□□， 我的系□在□□硬件□挂起， 或在安装□程中行□□常。

在 i386、 amd64 和 ia64 平台的□□□程中， FreeBSD 广泛使用了 ACPI 服□来□□系□配置， 不幸的是 ACPI

□□和主板 BIOS 中仍存在一些 bug。 在第三□段引□加□器中， 可以通□□置 hint.acpi.0.disabled 来禁用 ACPI：

```
set hint.acpi.0.disabled="1"
```

□一□置会在系□重□后失效， 因此必□将 hint.acpi.0.disabled="1" 添加至文件 /boot/loader.conf 中。□于引□加□器的更多信息， □参□ 概述。

Chapter 4. UNIX 基□

4.1. 概述

下列章□的命令和功能□用于FreeBSD操作系□。 同□□里□多内容和一些 □-UNIX® 操作系□相□。 假如□已□熟悉□些内容可跳□不□□。 假如□是FreeBSD新手， 那□□□□真□□地从□到尾□一遍□些章□。

□取□些内容，□将了解：

- □□在FreeBSD使用 "虚□控制台"。
- 在 UNIX® 中文件□限如何□作， 以及理解 FreeBSD 中的文件□志。
- FreeBSD 默□文件系□的架□。
- FreeBSD磁□架□。
- □□挂接或卸下文件系□。
- 什□是□程、守□□程、信号。
- 什□是shell，□当□□去改□登□□入的默□□境。
- □□使用基本的文本□□器。
- 什□是□□，什□是□□□点。
- FreeBSD 下，使用的是什□可□行文件格式。
- □□使用 man 手册并取得更多□□。

4.2. 虚□控制台和□端

可以用多□不同的方式使用 FreeBSD， 在文本□端□入命令是其中之一。 通□使用□□方式， □可以容易地使用 FreeBSD 来□得 UNIX® 操作系□的□活而□大的功能。 □一□将介□ "□端" 和 "控制台"， 以及如何在 FreeBSD 中使用它□。

4.2.1. 控制台

假如□没有□置 FreeBSD 在□□期□□□□形登□界面， 那□系□将在引□和□□脚本正□□行完成后，□□一个登□的提示。 □会看到□似□□的界面:

```
Additional ABI support:.
Local package initialization:.
Additional TCP options:.

Fri Sep 20 13:01:06 EEST 2002

FreeBSD/i386 (pc3.example.org) (ttyv0)

login:
```

□些信息可能和□的系□□微有点不同，但不会有很大差□。 最后□行是我□感□趣的， 理解□一行:

```
FreeBSD/i386 (pc3.example.org) (ttyv0)
```

□一行是□□才□□的系□信息其中一□， □所看到的是一个"FreeBSD"控制台， □行在一个Intel或兼容的x86体系架□上面。 □台□算机的名字 (□台 UNIX® □算机都有自己的名字) 叫 pc3.example.org， 就是□在□个系□控制台-□个 ttyv0 □端的□子。

在最后，最后一行一直保持□□:

```
login:
```

□里， □将可以□入用□名 "username" 并登□到 FreeBSD 系□中。 接下来的一□， 将介□如何登□系□。

4.2.2. □入FreeBSD

FreeBSD是一个多用□多任□的系□， □句□来□就是一个系□中可以容□□多不同的用□， 而□些用□都可以同□在□台机器中□行大量的程序。

□一个多用□系□都必□在某方面去区分 "user"， 在 FreeBSD 里 (以及 □-UNIX® 操作系□)， 完成□方面工作是有必要的， 因而， □位使用者在□行程序之前都必□首先 "登□"， 而□位用□都有与之□□的用□名 ("username") 和密□ ("password")。 FreeBSD 会在用□□入之前作出□□□□□信息。

当 FreeBSD 引□并□行完□□脚本之后， ， 它会□出一个提示， 并要求□入有效的用□名：

```
login:
```

□个例子更容易理解，我□假□□的用□名叫 john。 在提示符下□入 john 并按 Enter， 此□□□□看到□个提示 "password"：

```
login: john
Password:
```

□在□入 john的密□并按下 Enter。 □入密□□是 不回□的! 不必□此担心， □□做是出于安全考□。

假如□□入的密□是正□的， □□□□□已□入 FreeBSD， 并可以□始□□可用的命令了。

□□□看□ MOTD 或者出□一个命令提示符 (#、$ 或 % 字符). □表明□已成功登□□入FreeBSD。

4.2.3. 多个控制台

在一个控制台□行 UNIX® 命令□□很好， 但 FreeBSD 具有一次□行 多个程序的能力。 □使用一个控制台只会浪□ FreeBSD 同□□行多任□的能力。 而 "虚□控制台" 在□方面□□□大的功能。

FreeBSD 能配置出□足□不同需求的虚□控制台， 在□□上□用一□□就能从各个虚□控制台之□切□。 各个控制台有自己的□□通道， 当□在各个控制台切□□ FreeBSD 会切□到合□的□□□□通道和□示器□□通道。

FreeBSD 各个控制台之□可利用特殊□□切□并保留原有控制台，□可□□做: Alt + F1， Alt + F2， 一直到 Alt

+ F8 在FreeBSD里切□到其中一个虚□控制台。

同□地， □正在从其中某个控制台切□到□一个控制台的□候， FreeBSD 会保存正在使用和恢□将要使用屏幕□□通道。 □□□果形成一□ "□□"， □□有□多"虚□"屏幕和□□可以□入很多的命令。 □些程序需要在一个虚□控制台不能停止□行而又不需要□察它， 它□□□行而□可以切□到其他的虚□控制台。

4.2.4. /etc/ttys文件

FreeBSD 虚□控制台的默□配置□8个，但并不是硬性□置， □可以很容易□置虚□控制台的个数□多或□少。 虚□控制台的的□号和□置在 /etc/ttys 文件里。

□可以使用 /etc/ttys 文件在 FreeBSD 下配置虚□控制台。 文件里□一未加注□的行都能□置一个□端或虚□控制台 (当行里含有 # □个字符□不能使用) 。 FreeBSD 默□配置是配置出9个虚□控制台而只能□□8个， 以下□些行是 ttyv 一起□□:

```
# name  getty                           type    status          comments
#
ttyv0   "/usr/libexec/getty Pc"         cons25  on  secure
# Virtual terminals
ttyv1   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv2   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv3   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv4   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv5   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv6   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv7   "/usr/libexec/getty Pc"         cons25  on  secure
ttyv8   "/usr/X11R6/bin/xdm -nodaemon"  xterm   off secure
```

如果要了解□个文件中□一列的□□介□， 以及虚□控制台上所能使用的配置， □参考□机手册 ttys(5)。

4.2.5. □用□模式的控制台

□于 "□用□模式" □□介□在 □用□模式 □里可以□到。 当□□行□用□模式□只能使用一个控制台， 没有多个虚□控制台可使用。 □用□模式的控制台同也可以在 /etc/ttys 文件□置， 可在□行□到要□□的控制台：

```
# name  getty                           type    status          comments
#
# If console is marked "insecure", then init will ask for the root password
# when going to single-user mode.
console none                            unknown off secure
```

□个 console 已□注□掉, □可□□□行把 secure 改□ insecure。 □□， 当用□用□□入 FreeBSD □， 它仍然要求提供 root 用□的密□。

在把□个□□改□ insecure 的□候一定要小心， 如果□忘□了 root用□的密□， □入□用□会有点麻□。 尽管仍然能□入□用□模式， 但如果□不熟悉它就会非常令人□疼。

4.2.6. 改□控制台的□示模式

FreeBSD 控制台默□的□示模式可以被□整□ 1024x768， 1280x1024， 或者任何□的□□芯片和□示器所支持的其他尺寸。 要使用一个不同的□示模式， □必□首先重新□□内核并包含以下2个□□：

```
options VESA
options SC_PIXEL_MODE
```

在内核用□2个□□□□完成后，□就可以使用 vidcontrol(1) 工具来□定□的硬件支持何□□示模式了。 以 root 身□在控制台□入以下命令来□得一□所支持的□示模式列表。

```
# vidcontrol -i mode
```

□个命令的□出是一□□的硬件所支持的□示模式列表。 □可以在以 root 身□在控制台上□入 vidcontrol(1) 命令来改□□示模式：

```
# vidcontrol MODE_279
```

如果□□于新的□示模式□意，那□可以把它加入到 /etc/rc.conf 使机器在□次□□的□候都能生效， 我□使用了上一个例子中的模式：

```
allscreens_flags="MODE_279"
```

4.3. □限

FreeBSD，是 BSD UNIX® 的延□， 并基于几个□□的 UNIX® □念。 从一□始就多□提到 FreeBSD 是一个多用□的操作系□， 它能分□□理几个同□工作的用□所分配的毫无□□任□。 并□□□□位用□的硬件□□、 外□、内存和 CPU □理□□作出合理安排。

因□系□有能力支持多用□， 在□一方面系□都会作出□能□、 写和□行的□源□力限制。 □点□限以三个八位元的方式□存着， 一个是表示文件所属者， 一个是表示文件所属群□， 一个是表示其他人。□些数字以下列方式表示：

数□	□限	目□列表
0	不能□，不能写，不能□行	`---`
1	不能□，不能写，可□行	`--x`
2	不能□，可写，不能□行	`-w-`
3	不能□，可写，可□行	`-wx`
4	可□，不能写，不能□行	`r--`
5	可□，不能写，可□行	`r-x`
6	可□，可写，不能□行	`rw-`

数□	□限	目□列表
7	可□，可写，可□行	rwx

使用命令的 -l (ls(1)) 参数可以□示出文件的所属者、 所属□和其他人等属性。 □看以下的例子：

```
% ls -l
total 530
-rw-r--r--  1 root  wheel     512 Sep  5 12:31 myfile
-rw-r--r--  1 root  wheel     512 Sep  5 12:31 otherfile
-rw-r--r--  1 root  wheel    7680 Sep  5 12:31 email.txt
...
```

使用 ls -l 在□行的□始出□了：

```
-rw-r--r--
```

从左□起的第一个字，告□我□□个文件是一□□的文件: 普通文件?目□?特殊□□?socket?或是□□文件? 在□个例子，- 表示一个普通文件。 接下来三个字是 rw- 是文件□有者的□限。 再接下来的三个字是 r-- 是文件所属群□的□限。 最後三个字是 r-- 是其他人的□限。 以□一个文件□例，他的□限□定是□有者可以□写□个文件、群□可以□取、 其他使用者也能□取□个文件。 根据上面的表格， 用数字表示□个文件其三部分的□限□□是 644。

□□很好，但系□□□□□□□行□限控制的? 事□上 FreeBSD 将大部□硬件□□当作一个文件看待， 用程序能打□、□取、写入数据就如其他的文件一□。 而□□文件放在 /dev 目□。

目□也□□一□文件，也有□取、写入、□行的□限。 但目□的□行□限意□并不与普通文件相同， □□上□行□限是□入□限。 当一个目□是被□示可以□行的□， 表示可以□入它， 或者□言之， 利用 "cd" (改□当前目□) □入它。此外， □也表示有□□入目□的用□， 可以□□其下的已知名字的文件 (当然目□下的文件也受到□□限制)。

□□方面，想□取一个目□的列表就必□□□可□□限， 同□想□除一个已知的文件，就必□把目□下□个文件□□可写 和□行□限。

□有更多□限□定， 但是他□大多用在特殊状况下如一个setuid的□行文件和粘□性目□， 如果想要得知有□文件□限和如何□定的更多□□，□看手册chmod(1)。

4.3.1. □限的符号化表示

□限符号，某些□候就是指符号表□式， 使用八□制的字符□目□或文件分配□限。 □限符号的使用□法是 (□) (作用) (□限)。 看看下列数□的在那些地方所起什□□的作用:

□□	字母	介□
(□)	u	用□
(□)	g	所属群体
(□)	o	其他人
(□)	a	所有人 ("全部")
(作用)	+	□加□限

□□	字母	介□
(作用)	-	□少□限
(作用)	=	□定□限
(□限)	r	可□
(□限)	w	可写
(□限)	x	□行
(□限)	t	粘□位
(□限)	s	□置 UID 或 GID

□些数□ chmod(1) 以□□□定的。 □个例子，用以下命令阻止其他人□□ *FILE*文件:

```
% chmod go= FILE
```

如果需要□文件一次□行多□□， □可用逗号分□， 在下面的例子中， 将去掉 *FILE* 文件的群体和 "全体其他用□" 可写□限， 并□所有人□加可□行□限：

```
% chmod go-w,a+x FILE
```

4.3.2. FreeBSD 文件□志

在前面所介□的文件□限的基□之上， FreeBSD □支持使用 "文件□志"。 □些□志□文件提供了□一□的安全控制机制， 但□些控制并不□用于目□。

□些文件□志提供了□□文件的□一□控制， □助□保即使是 root 用□也无法□除或修改文件。

文件□志可以通□使用 chflags(1) 工具来修改， 其用□界面很□□。 例如， 要在文件 file1 上□用系□禁□□志，□使用下述命令：

```
# chflags sunlink file1
```

要禁用系□禁□□志， 只需在前述命令中的 sunlink □志前加 "no"。 例如：

```
# chflags nosunlink file1
```

要□示文件上的□志， □使用命令 ls(1) 的 -lo 参数：

```
# ls -lo file1
```

□出□果□□似于：

```
-rw-r--r--  1 trhodes  trhodes  sunlnk 0 Mar  1 05:54 file1
```

□多□志只可以由 root 用□来□加， 而□一些， □可以由文件的所有者来□加。 建□管理□仔□□□ chflags(1) 和 chflags(2) □机手册， 以□其加深理解。

4.3.3. setuid、 setgid 和 sticky □限

除了前面已□□□□的那些□限之外， □有三个管理□□□知道的□限配置。 它□是 setuid、 setgid 和 sticky。

□些配置□于一些 UNIX® 操作而言很重要， 因□它□能提供一些一般情况下不会授予普通用□的功能。 □了便于理解， 我□首先介□真□用□ ID (real user ID) 和生效用□ ID (effective user ID)。

真□用□ ID 是□有或□□□程的用□ UID。 生效 UID 是□程以其身□□行的用□ ID。 □例来□， passwd(1) 工具通常是以□起修改密□的用□身□□□， 也就是□其□程的真□用□ ID 是那个用□的 ID； 但是， 由于需要修改密□数据□， 它会以 root 用□作□生效用□ ID 的身□□行。 □□， 普通的非特□用□就可以修改口令， 而不是看到 Permission Denied □□了。

mount(8) 的 nosuid □□可以令系□在不□出任何□□提示的情况下不□行□些程序。 □一方面， □个□□并不是万无一失的， 正如 mount(8) □机手册所提到的那□， 如果系□中安装了□□ nosuid 的封装程序， 那□□□保□就可以被□□了。

setuid □限可以通□在普通□限前面加上一个数字四 (4) 来□置， 如下面的例子所示：

```
# chmod 4755 suidexample.sh
```

□□一来， suidexample.sh 的□限□□如下面□□：

```
-rwsr-xr-x   1 trhodes  trhodes    63 Aug 29 06:36 suidexample.sh
```

□会注意到， 在原先的属主□行□限的位置□成了 s。 □□， 需要提升特□的可□行文件， 例如 passwd 就可以正常□行了。

可以打□□个□端来□察□一情形。 在其中一个□端里面， 以普通用□身□□□ passwd □程。 在它等待□入新口令□， 在□一个□端中□看□程表中□于 passwd 命令的信息。

在□端 A 中：

```
Changing local password for trhodes
Old Password:
```

在□端 B 中：

```
# ps aux | grep passwd
```

```
trhodes  5232  0.0  0.2  3420  1608   0  R+    2:10AM   0:00.00 grep passwd
root     5211  0.0  0.2  3620  1724   2  I+    2:09AM   0:00.01 passwd
```

正如前面所□的那□， passwd 是以普通用□的身□□□的， 但其生效 UID 是 root。

与此□□， setgid □限的作用， 与 setuid □限□似， 只是当□用程序配合□一□定□行□， 它会被授予□有文件的那个□的□限。

如果需要在文件上配置 setgid □限， 可以在□限数□前面□加数字二 (2) 来□行 chmod 命令，如下面的例子所示：

```
# chmod 2755 sgidexample.sh
```

可以用与前面□似的方法来□□新□定的生效情况， 在□□限的地方的 s 表示□一配置已□生效：

```
-rwxr-sr-x   1 trhodes  trhodes    44 Aug 31 01:49 sgidexample.sh
```

在□些例子中， 尽管 shell 脚本也属于可□行文件的一□， 但它□不会以□配置的 EUID 或生效用□ ID 的身□□行。 □是因□ shell 脚本可能无法直接呼叫 setuid(2) □用。

我□已□□□了□个特殊□限位 (setuid 和 setgid □限位)， 它□□用□在使用程序□能□用到更高的□限， 有□□会削弱系□的安全性。 除了□□个之外， □有第三个特殊□限位： sticky bit， 它能□□□安全性。

当在目□上□置了 sticky bit 之后， 其下的文件就只能由文件的所有者□除了。 □个□限□置能□防止用□□除□似 /tmp □□的公共目□中不属于他□的文件。 要□用□□□限， 可以在□限□置前面加上数字一 (1)。 例如：

```
# chmod 1777 /tmp
```

□在， 可以用 ls 命令来□看效果：

```
# ls -al / | grep tmp
```

```
drwxrwxrwt  10 root  wheel         512 Aug 31 01:49 tmp
```

□里的□尾的 t 表示了 sticky bit □限。

4.4. 目□架□

理解 FreeBSD 的目□□次□□□于建立□系□整体的理解十分重要的基□。 其中， 最重要的概念是根目□， "/"。□个目□是系□引□□挂接的第一个目□， 它包含了用以准□多用□操作所需的操作系□基□□件。 根目□中也包含了用于在□□□□□到多用□模式之前挂接其他文件系□所需的挂接点。

挂接点 (mount point) 是新□的文件系□在接入□有系□□的起点位置 (通常是根目□)。 在 磁□□□ □此□行了□□的

□述。 □准的挂接点包括 /usr、 /var、 /tmp、 /mnt， 以及 /cdrom。 □些目□通常会在 /etc/fstab 文件中提及。 /etc/fstab 是一□包含系□中各个文件系□及挂接点的表。 在 /etc/fstab 中的□大多数文件系□都会在□□□由 rc(8) 脚本自□挂接， 除非特□指定了 noauto □□。 更多□□□参考 fstab 文件。

□可以通□ hier(7) 来了解完整的文件系□□次□明。 □在， □我□先来看一看□大多数的常□的目□以供参考。

目□	介□
/	文件系□的根目□。
/bin/	在□个用□和多用□□境下的基本工具目□。
/boot/	在操作系□在□□加□期□所用的程序和配置。
/boot/defaults/	默□□□引□□□的配置内容，□□□loader.conf(5)。
/dev/	□□□点，□□□ intro(4)。
/etc/	系□□□的配置和脚本。
/etc/defaults/	系□默□的□□配置和脚本，□参考 rc(8) 。
/etc/mail/	□系到□件系□□作的配置， □参考 sendmail(8)。
/etc/namedb/	named 配置文件，□参考 named(8)。
/etc/periodic/	□天、□星期和□月周期性地□行的脚本， □通□ cron(8)□□ periodic(8)。
/etc/ppp/	ppp配置文件，□□□ppp(8)。
/mnt/	由管理□□□使用挂接点的□□空目□。
/proc/	□行中的文件系□，□参□ procfs(5) 和 mount_procfs(8)。
/rescue/	用于□急恢□的一□静□□□的程序； 参□ rescue(8)。
/root/	root用□的Home(主)目□。
/sbin/	在□个用□和多用□□境下的存放系□程序和管理所需的基本□用目□。
/tmp/	□□文件。 /tmp 目□中的内容， 一般不会在系□重新□□之后保留。 通常会将基于内存的文件系□挂在 /tmp 上。 □一工作可以用一系列 tmpmfs 相□的 rc.conf(5) □量来自□完成。 (或者， 也可以在 /etc/fstab □加□□□； 参□ mdmfs(8))。
/usr/	存放大多数用□的□用□件。
/usr/bin/	存放□用命令，程序□□工具，和□用□件。
/usr/include/	存放□准 C include 文件.
/usr/lib/	存放□文件。
/usr/libdata/	存放各□□用工具的数据文件。
/usr/libexec/	存放系□□用或后台程序 (从□外的程序□□□行)。

目□	介□
/usr/local/	存放本地□行文件， □文件等等， 同□也是 FreeBSD ports 安装的默□安装目□。 /usr/local 在 /usr 中的目□布局大体相同， □□□ hier(7)。 但 man 目□例外， 它□是直接放在 /usr/local 而不是 /usr/local/share 下的， 而 ports □明文□在 share/doc/port。
/usr/obj/	通□□□ /usr/src 得到的目□文件。
/usr/ports/	存放 FreeBSD 的 Ports Collection (可□)。
/usr/sbin/	存放系□后台程序 和 系□工具 (由用□□行)。
/usr/shared/	存放架□独立的文件。
/usr/src/	存放 BSD 或者本地源□文件。
/usr/X11R6/	存放 X11R6 可□行文件、 □文件、 配置文件等的目□(可□)。
/var/	多用途日志、 □□或短期存放的， 以及打印假脱机系□文件。 有□会将基于内存的文件系□挂在 /var 上。 □一工作可以通□在 rc.conf(5) 中□置一系列 varmfs □量 (或在 /etc/fstab 中加入一行配置； 参□ mdmfs(8)) 来完成。
/var/log/	存放各□的系□□□文件。
/var/mail/	存放用□mailbox(一□□件存放格式)文件。
/var/spool/	各□打印机和□件系□spooling(回□)的目□。
/var/tmp/	□□文件。 □些文件在系□重新□□□通常会保留， 除非 /var 是一个内存中的文件系□。
/var/yp/	NIS 映射。

4.5. 磁□□□

FreeBSD □□文件的最小□位是文件名。 而文件名区分大小写，□就意味着 readme.txt 和 README.TXT 是□个不相同的文件。 FreeBSD 不凭文件□展名 (.txt) 去□□□个文件是 程序、 文□， 或是其他格式的数据。

各□文件存放在目□里。 一个目□可以□空， 也可以含有多个的文件。一个目□同□可以包含其他的目□， 允□□在一个目□里建立多个不同□次的目□。 □将□助□□松地□□□的数据。

文件或目□是由文件名或目□名，加上斜□符号 /， 再根据需要在目□名后面加上其他目□的名称。 如果□有一个名□ foo 的目□， 它包含□一个目□ bar， 后者包括一个叫 readme.txt 的文件， □全名， 或者□到文件的 路径 就是 foo/bar/readme.txt。

在文件系□里目□和文件的作用是存□数据。 □一个文件系□都有且只有一个□□目□ 根目□， □个根目□□可以容□其他目□。

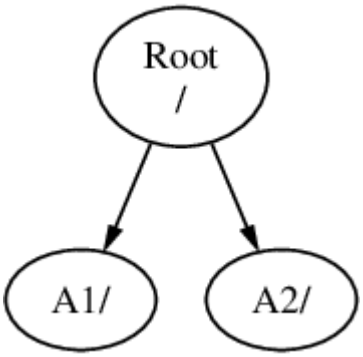
□也□在其他的一些操作系□□到□似□里的情况， 当然也有不同的情况。 □些例子， MS-DOS® 是用 \ 分隔文件名或目□名， 而 Mac OS® □使用:。

FreeBSD在路径方面不使用□□器名符号或□□器名称， 在FreeBSD里□不能□□使用： c:/foo/bar/readme.txt。

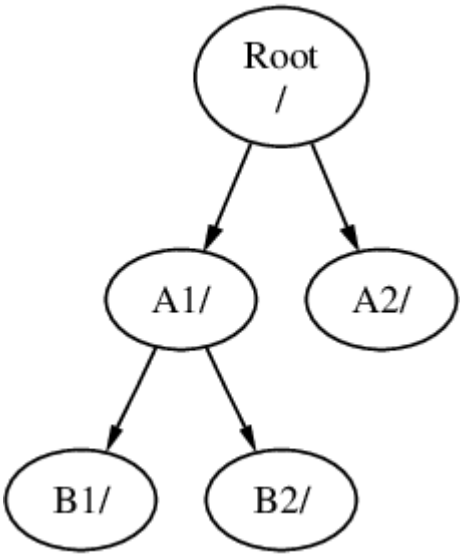
□了代替(□□器名符号)， 一个文件系□会指定 根 文件系□， 根文件系□的根目□是 /。 其他□一个文件系□挂接在根文件系□下。 无□有多少磁□在FreeBSD 系□里， □个磁□都会以目□的方式加上。

假□□有三个文件系□， 名□ A、 B 和 C。 □个文件系□有一个根目□， 而各自含有□个其他的目□， 名□ A1, A2 (B1, B2 和 C1, C2)。

看看 A □个根文件系□。 假如□用 ls 命令来□看□个目□□会□到□个子目□: A1 和 A2。 □个目□□是□个□子:



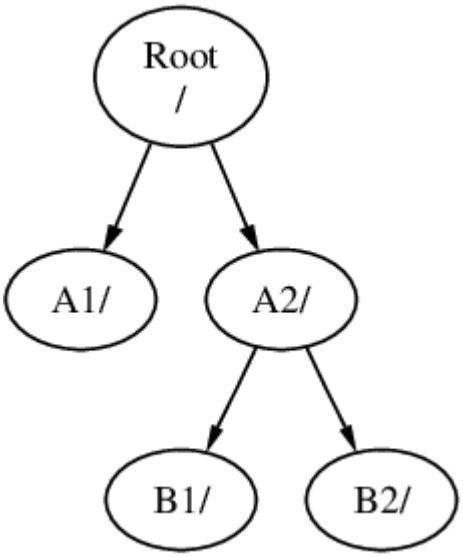
一个文件系□必□挂到□一个文件系□的某一目□， 所以□在假□把 B 文件系□挂到 A1目□， 那 B 根目□因此代替 了 A1， 而□示出 B 目□(的内容)：



无□B1 或 B2 目□在那里而延伸出来的路径必□□ /A1/B1 或 /A1/B2。 而在 /A1 里原有的文件会□□□藏。 想□些文件再出□把 B 从 A 挂接□放。

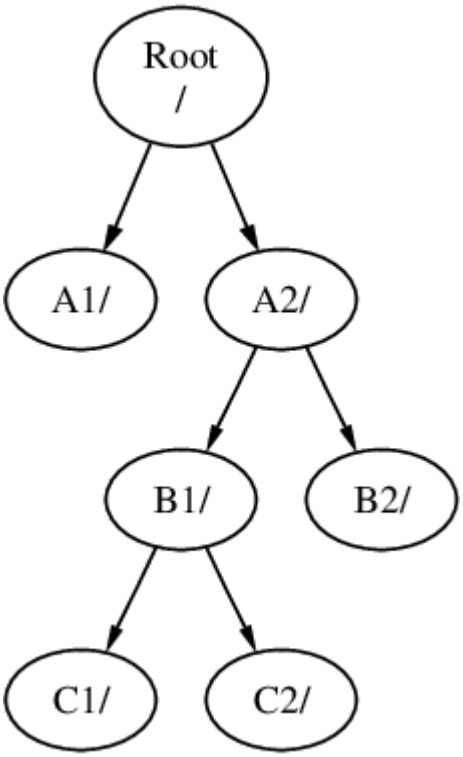
所有在B1 或 B2 目□里的文件都可以通□ /A1/B1 或 /A1/B2 □□。而在 /A1 中原有的文件会被□□□藏，直到 B 从 A 上被卸□ (unmout) □止。

把 B 挂接在 A2 那□表的□子就是□□子:

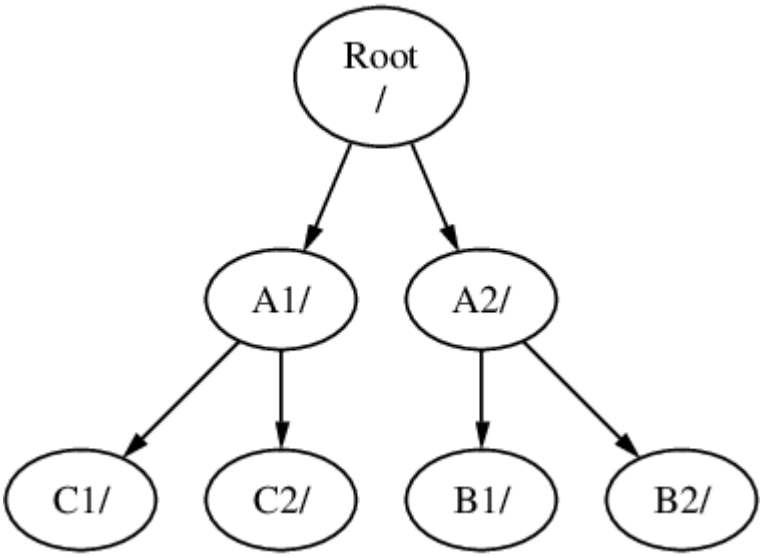


□个路径分□是 /A2/B1 和 /A2/B2 。

文件系□能把□部挂接在□一个文件系□上。 □□□个例子， 把 C 文件系□挂接在 B 文件系□里的 B1 目□，排列如下:



或者把 C 文件系□挂接在 A 文件系□里的A1目□：



假如□熟悉 MS-DOS® 并知道 join 命令， 尽管不相同，其□功能是相似的。

□方面不是普通知□而且□及到□自己所□心的， 当□安装FreeBSD并在以后添加新磁□□， □必□知到□如何新建文件系□和挂接上。

(FreeBSD系□)它有一个主要的根文件系□， 不需要□外新建立， 但当需要手工□理□，□是一个有用的知□。

多个文件系□的益□

- 不同的文件系□可用不同的 挂接参数。 □些例子， 仔□想一下， 根文件系□能用只□的方式挂接上，防止不□意□除或□□到一个危□的文件。 把各用□能写入的文件系□分□， 像/home□□， 由□外的文件系□分□用 *nosuid* 参数挂接，□个参数防止 *suid/guid* 在□行□个文件系□中的文件□生效， 从而□解了一些安全□□。
- FreeBSD 能根据一个文件系□使用的情况自□□化 □个文件系□上的文件布局。 所以□一个存□了大量小文件并会被□繁写入文件系□的□化与一个存□了少量大文件的□化是不同的。 而在一个大的□一文件系□上□无法体□□□的□化。
- FreeBSD 的文件系□能□在断□□尽可能避免□失。 然而， 在□□点□的□源失效仍然可能会破坏文件系□的□□。 将□的文件系□分成多个有助于分散□□， 并方便□□和恢□。

□一文件系□的益□

- 文件系□是固定大小的。 当安装FreeBSD□新建一个文件系□并□定一个大小， □会在□后□□到必□去建一个大的分区。 如果配置不当， □需要□□、 重新□建文件系□， 然后再恢□数据。

FreeBSD 提供了 growfs(8) 命令。 □使得能□□□地□整文件系□的大小，因而不再受其限制。

文件系□是和分区一一□□的。 □里的分区和常用的□□分区 (例如， MS-DOS® 分区) 的意思并不一□， □是由于 FreeBSD 的 UNIX® □□造成的。 □一个分区使用一个从 a 到 h 的字母来表示。 □个分区只能包含一个文件系□， □意味着文件系□通常可以由它□在文件系□目□□□中的挂接点， 或□□的分区字母来表示。

FreeBSD 的 交□分区 也需要使用磁□空□。 交□分区是□ FreeBSD 作 虚□内存 使用的， □□能令□的□算机有更多的内存可使用， 当FreeBSD在□行而内存不□的□候， 它会把其他一些可□移的数据□移到交□分区， 空出内存的位置以供使用。

某些 partitions 的用途是□定的。

分区	□定
a	通常指定□根文件系□
b	通常指定□交□分区
c	通常它和所在的 slice 大小相同。 c 分区上工作□必定会影□到事整个 slice (□个例子，坏□描器)。□通常不□意在□个partition建立文件系□。
d	分区 d 曾□有特殊的含□， 不□□意□在□□的系□上已不再□用， 因此 d 可以和任何其它普通的分区一□使用了。

□一个包含了文件系□的分区被保存在 FreeBSD 称□ *slice* 的部分上。 Slice 是一个 FreeBSD □□，通常被叫做分区， 再次□□， □是由于 FreeBSD 的 UNIX® 背景。 Slices 有其□号， 从1到4。

Slice □号在□□名后面， 并有一个 s 前□， 从 1 □始。 因此 "da0*s1*" 是第一个 SCSI □□器的第一个 slice。□个磁□上只能有四个物理的 slices， 但□可以在物理 slice 中使用□当的□型来□建□□ slice。 □些□展 slice □号从 5 □始， 因此 "ad0*s5*" 是第一个 IDE 磁□中的第一个 □展 slice。 文件系□所使用的□□□□占□ slice。

Slices, "□用指定" 物理□□器， 和其他□□器都包含 *partitions*， 那几个的 partitions 都是用字母从 a 到 h 来□定的， 而□些字母都在□□器名字之后，所以 "da0*a*" 是指首个da□□的 a partition， 而那个就是 "□□指定"。"ad1s3*e*" 是指IDE磁□上第三个slice的第五个partition。

最□，□个磁□都被系□□□。 一个磁□名字是用磁□□型代□和□号来□□的， 它不像slices，磁□的□号是由0□始的。□□代□□看□里所列出的磁□□□的代□。

当在 FreeBSD 中指定 partition 名字□， 必□同□包含□个分区的 slice 和磁□的名字； □似地， 在指定 slice □， 也□□□出包含□ slice 的磁□名字。 可□□列出： 磁□名称，s，slice □号，和partition□定字母。 例子□看□例磁□, Slice, 和 Partition 它□的命名。

一个磁□的布局 □里□示了一个磁□的布局，有更清楚的□助。

在安装FreeBSD□，□首先要配置好磁□slices， 然后在FreeBSD使用的slice上建立partitions。 并在□个partition上建立一个文件系□(或交□分区)， 和指定文件系□的挂接位置。

表 *6.* 磁□□□的代□

代□	□明
ad	ATAPI (IDE) 磁□
da	SCSI 直接存取磁□
acd	ATAPI (IDE) 光□
cd	SCSI 光□
fd	□□

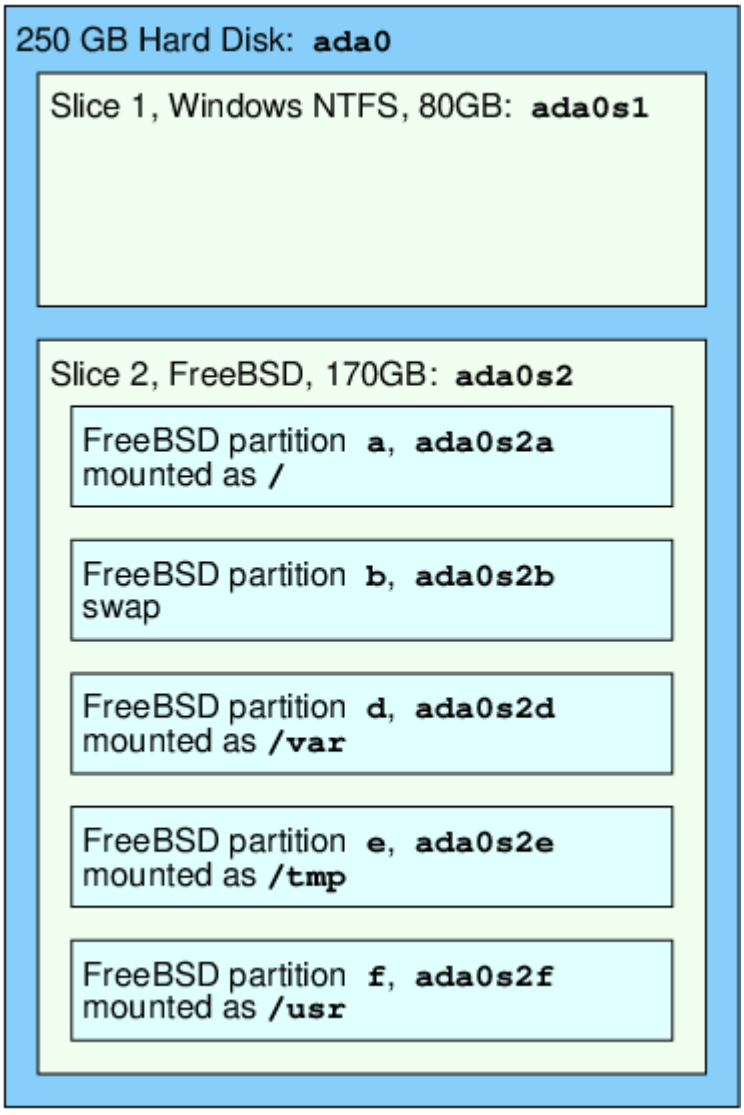
例 6. □例磁□, *Slice*, 和 *Partition* 它□的命名

命名	□明
ad0s1a	在首个IDE磁□(ad0)上的 第一个slice (s1)里的 第一个partition (a)。
da1s2e	在第二个SCSI磁□(da1)上的 第二个slice(s2)里的 第五个partition(e)。

例 7. 一个磁□的布局

从在系□里的首个IDE磁□□表可以□示出FreeBSD的□解。 假□磁□大小□4 GB，它里面包含了□个2 GB 大小的slices (但在MS-DOS®叫partitions)。 首个slice是一个MS-DOS®磁□叫C:， 而第二个slice 是FreeBSD配置好的slice。 FreeBSD配置好的slice有三个partitions和□一个交□分区。

□三个partitions各自控制一个文件系□。 partitiona 用于根文件系□， partitione 用于 /var 目□□， partitionf 用于 /usr 目□□。



4.6. 文件系□的挂接和卸下

□□文件系□就像一□□那□用/□立根部， 是比□理想的文件系□。 而/dev、 /usr 和其他目□就是根目□的分枝， □外□些目□可以再分枝，例如/usr/local。

□□考□□某些目□一些空□从而分散文件系□。 /var 之下包含目□ log/，目□spool/， 和不同□型的□□文件，很可能把它塞□。 把什□都塞□根文件系□不是一个好主意， 好的做法是□□把 /var 从 /分□出去。

□一个要考□的是，□物理□□或虚□磁□□些自□空□的文件系□□定目□□□□。 例如 网□文件系□ 或光□的挂接。

4.6.1. fstab 文件

在 引□□程 期□， 自□挂上/etc/fstab所列出的文件系□。 (除非他□注明□noauto □□)。

/etc/fstab 文件包含的各行的列表格式如下:

```
device        /mount-point fstype     options      dumpfreq     passno
```

device

□□名称(□□必□存在)， □明在 □□命名.

mount-point

目□ (目□必□存在)， 用在那个挂接上的文件系□上。

fstype

文件系□□型，□通□mount(8)□□。 默□的FreeBSD文件系□□型是ufs。

options

□□可□写文件系□的rw□□， 或□□只□文件系□的ro□□， 或其他一些□□，可随意□一个。 一个常用的□□ noauto 用在不需在引□□程期□挂接的文件系□。 其他的□□在 mount(8) 手册里列出。

dumpfreq

dump(8) 使用□□去决定那个文件系□必□移□。 假如缺少□□，默□的数□□0。

passno

□一□决定文件系□的□□□序， 文件系□想跳□□□□将passno□□0。 根文件系□(那个是在□方面□始之前必□□□的)□□将它的 passno □□1， 其他文件系□的 passno 必□把数□□到大于1。假如多个文件系□的passno的□相同，那□ fsck(8) 在允□的情况下将□□并行地去□□文件系□。

□参□ fstab(5) □机手册， 以□得□于 /etc/fstab 文件格式， 以及其中所包含的□□的□一□信息。

4.6.2. mount 命令

□个 mount(8) 命令是挂接文件系□的基本□用。

使用最多的基本格式:

```
# mount device mountpoint
```

它的□□非常多，而mount(8) 手册同□提及， 但常用的都在□里:

挂接的各□□□

`-a`

挂接/etc/fstab里所有列出的文件系□。 除非□□□ "noauto" 或作了排除在外的 `-t` □型□□，或者在□之前已挂上。

`-d`

除了□□上系□□用以外，可以完成任何事情，□个□□是和 `-v`参数一起□在一□使用，可以决定mount(8)所做的事情。

`-f`

□制去挂接一个未知的文件系□(会有危□)， 或当把一个文件系□挂接状□由可□写降□只□□，□制撤消可写通道。

`-r`

以只□方式挂接文件系□。 □和在指定了 `-o` □□配合 `ro` 参数的效果是一□的。

`-t` ***fstype***

根据□出的文件系□□型挂接文件系□， 假如□于`-a`□□，□挂接□个□型的文件系□。

"ufs" 是默□的文件系□□型。

`-u`

在文件系□上修改挂接□□。

`-v`

版本模式。

`-w`

以可□写方式挂接文件系□。

The `-o` □□采用一个逗号分□以下多个□□:

noexec

不允□文件系□上的二□制程序□行。□也是一个有用的安全□□。

nosuid

不允□文件系□上的 setuid 或 setgid □□生效。□也是一个有用的安全□□。

4.6.3. `umount` 命令

umount(8) 命令同□采用一个参数、一个挂接点、一个□□名。 或采用`-a`□□，又或采用`-A`□□。

所有格式都可采用 -f 去□行卸下， 或采用-v 用那□当的版本。 但警告，采用 -f并不是一个好主意，□行卸下文件系□可能□坏□算机或破坏文件系□上的数据。

-a 和 -A 会卸下所有已挂接的文件系， 可能通□-t后面列出的文件系□□行修改， 但无□如何，-A都不会□□去卸下根文件系□。

4.7. □程

FreeBSD 是一个多任□操作系□。 □就意味着好像一次可以□行一个以上的程序。 □个占用一定□□□行的程序就叫 □程 (process)。 □□行的□一个命令会至少□□一个新□程，□有很多一直□行着的系□□程，用以□持系□的正常□作。

□个□程用来□□的一个□号就叫 □程 *ID*， 或叫 *PID*。 而且，就像文件那□，□个□程也有所属用□和所属群体。所属用□和所属群体使用在□方面:□定□个□程可以打□那些文件和那些□□， 从而在初期使用文件的□限。多数的□程都有一个父□程， 而□程是依□父□程来□□的。 例如，假如□把命令□入到shell里那shell是一个□程，而□□行的各个命令同□是□程， 那□，shell就是□各个□行□程的父□程。 而□方面有一个例外的□程就叫init(8)。 init始□是首个□程,，所以他的PID始□是1， 而init在FreeBSD起□□由内核自□□□。

在系□上，有□个命令□□程□察非常有用:ps(1) 和 top(1)。 □个ps命令作用是□察当前□行□程的状□， □示他□的PID，使用了多少内存，它□□□的命令行。 而top命令□是□示所有□行□程，并在以秒□的短□内更新数据。□能交互式的□察□□算机的工作。

默□情况下， ps□□示出□自己所□行的命令。 例如:

```
% ps
  PID  TT  STAT      TIME COMMAND
  298  p0  Ss     0:01.10 tcsh
 7078  p0  S      2:40.88 xemacs mdoc.xsl (xemacs-21.1.14)
37393  p0  I      0:03.11 xemacs freebsd.dsl (xemacs-21.1.14)
48630  p0  S      2:50.89 /usr/local/lib/netscape-linux/navigator-linux-4.77.bi
48730  p0  IW     0:00.00 (dns helper) (navigator-linux-)
72210  p0  R+     0:00.00 ps
  390  p1  Is     0:01.14 tcsh
 7059  p2  Is+    1:36.18 /usr/local/bin/mutt -y
 6688  p3  IWs    0:00.00 tcsh
10735  p4  IWs    0:00.00 tcsh
20256  p5  IWs    0:00.00 tcsh
  262  v0  IWs    0:00.00 -tcsh (tcsh)
  270  v0  IW+    0:00.00 /bin/sh /usr/X11R6/bin/startx -- -bpp 16
  280  v0  IW+    0:00.00 xinit /home/nik/.xinitrc -- -bpp 16
  284  v0  IW     0:00.00 /bin/sh /home/nik/.xinitrc
  285  v0  S      0:38.45 /usr/X11R6/bin/sawfish
```

在□个例子里□可看到，从 ps(1) □出的□一列是有□律的。 PID 就是□程ID，□个□早前已□□□了。 PID号的分配由1一直上升直到99999， 当□□行到超□限制□，□些□号会回□分配 (仍在使用中的 PID 不会分配□其他□程)。 TT □一列□示了程序□行所在的□端， 目前可以安全地忽略。 STAT □示程序的状□，也可以安全地被忽略。TIME是程序在CPU□理□□-□行的□□量， 并不是指□程序□□到□在的所用的□□。 □多程序□巧遇到某方面在他□之前要花□大量CPU□理□□□，他□就必□等候。 最后， COMMAND 是□行程序□使所用的命令行。

ps(1)支持使用各□□□去改□□示出来的内容， 最有用的一个就是auxww。 a□□□示出所有□行□程的内容， 而不□□是□的□程。 u□□□示出□程所□属的用□名字以及内存使用， x □□□示出后台□程。 而 ww □□表示□ ps(1) 把□个□程的整个命令行全部□示完， 而不是由于命令行□□就把它从屏幕上截去。

下面和从top(1)□出是□似的，一个示例式□□就象□□子:

```
% top
last pid: 72257;  load averages:  0.13,  0.09,  0.03    up 0+13:38:33  22:39:10
47 processes:  1 running, 46 sleeping
CPU states: 12.6% user,  0.0% nice,  7.8% system,  0.0% interrupt, 79.7% idle
Mem: 36M Active, 5256K Inact, 13M Wired, 6312K Cache, 15M Buf, 408K Free
Swap: 256M Total, 38M Used, 217M Free, 15% Inuse

  PID USERNAME PRI NICE  SIZE    RES STATE    TIME   WCPU    CPU COMMAND
72257 nik       28   0  1960K  1044K RUN      0:00 14.86%  1.42% top
 7078 nik        2   0 15280K 10960K select   2:54  0.88%  0.88% xemacs-21.1.14
  281 nik        2   0 18636K  7112K select   5:36  0.73%  0.73% XF86_SVGA
  296 nik        2   0  3240K  1644K select   0:12  0.05%  0.05% xterm
48630 nik        2   0 29816K  9148K select   3:18  0.00%  0.00% navigator-linu
  175 root       2   0   924K   252K select   1:41  0.00%  0.00% syslogd
 7059 nik        2   0  7260K  4644K poll     1:38  0.00%  0.00% mutt
...
```

□个□出分成□部□。 前面部□(起始前五行) □示了:□行于最后□程的PID、 系□□□均衡 (那个是指系□繁忙□的□□方式)、 系□正常□行□□ (指从□□算起所用的□□) 和当前□□。 前面部□□外的□表 □及:多少□程在□行(□个情况是47)， 多少内存和多少交□分区在使用， 和在不同CPU状□里系□消耗多少□□。

在那下面一□串的□列和从ps(1)□出的的内存是相似的。 如以前ps(1)一□，□能□到:PID、用□名、CPU□理□□合□、□行的命令。 top(1)默□是□示□的□程所用内存空□的合□。 内存空□□里分成□列，一列□□体大小，□一列是必□□求□留大小是多少内存-□体大小。 而□留大小□□上是瞬□使用的多少。 在以上那个例子，□会看到那getenv(3)□□需要30 MB内存， 但□□只用了9 MB。

top(1) □□秒自□刷新一次，□可以用s改□刷新的秒数。

4.8. 守□□程，信号和□死□程

当□□行一个□□器□它是很容易控制的，告□它去加□文件它就加□。 □之所以能□□做，是因□□□器提供□□便利去□□做，和因□有□□器去附上的□端。 一些程序在□行中不需要□□的用□□入，一有机会就从□端里分□到后台去。例如，一个web系□整天都在作web□求的□□，他不需要□□入任何□西就能完成， □个□□的□一个例子就是把email的□送。

我□把那些程序叫 守□□程。 守□神是希□神□中的一些人物，非正非邪，他□是些守□小精□， 大体上□人□作出□献。 □多□似web服□或mail服□的系□□于今天仍有用途， □就是□什□在那□□的□□里，BSD的吉祥物保持□一双鞋加一把□叉的守□神模□。

守□□程的程序命名通常在最后加一个 "d"。 BIND 是伯克利互□网域名服□ (而□□□行的程序名称□是 named)，Apache web系□的程序就叫 httpd， 在行式打印机上的打印守□□程就是 lpd。 □只是一□□例，不是□准或硬性□定。 例如，□Sendmail而□用的主要mail守□□程就叫sendmail， 却不叫maild，□和□推□的一□。

有□可能会需要与守□□程□行通□。 而 信号 □是其中的一□通□机制。 可以□送信号□守□□程 (或相□的□一些□程)来与它□行通信， 不同的信号都有自己的数字□号-其中一些有特殊的含□， 其它的□可以被□用程序自己□行解□， 而一般来□， □用程序的文□会告□□些信号会被如何□理。 □只能□所属于□的□程□信号，假如□□其他人的□程□信号， □程就会用kill(1) 或 kill(2)□限□行拒□。 当然,root 用□会例外，它能把各□信号□送□□个□程。

在某些情况下，FreeBSD也会向□用□件□送信号。 假如一个□用□件含有□意写入并□□去□□内存，那是不可想象的，FreeBSD会向那个□程□送 段式□□ 信号 (SIGSEGV)。 假如一个□用□件使用alarm(3)系□去□行周期性□用□□功能，□当□到□□□， FreeBSD会向□用□件□送□□信号(SIGALRM)。

有□个信号可以停止□程:SIGTERM 和 SIGKILL。 SIGTERM比□友好，□程能捕捉□个信号， 根据□的需要来□□程序。在□□程序之前，□可以□束打□的□□文件和完成正在做的任□。 在某些情况下， 假如□程正在□行作□而且不能中断，那□□程可以忽略□个 SIGTERM信号。

□于SIGKILL信号，□程是不能忽略的。 □是一个 '"我不管□在做什□,立刻停止"'的信号。 假如□□送SIGKILL信号□□程， FreeBSD就将□程停止在那里。.

□可能会去使用 SIGHUP、 SIGUSR1 和 SIGUSR2信号。 □都是些通用的信号，各□□用程序都可以□用在各方面的信号□送。

假如□改□了web系□的配置文件-并想web系□去重□它的配置， □可以停止然后再□□httpd。但□□做web系□会□致一个短□ 的中断周期，那□是不受□迎的。几乎所有的守□□程在□写□，都会指定□SIGHUP 信号□行□□从而重□配置文件。 所以， 最好的方法， 就不是□死并重□ httpd， 而是□一个 SIGHUP 信号□它。 因□在□方面没有一个□准，不同的守□□程有不同的用法，所以不了解□□□一下守□□程的文□。

□送信号可用kill(1) 命令， □参考kill(1)所列出的例子。

Procedure: □送一个信号□□程

□个例子□示了□□去□一个信号□inetd(8)。 inetd配置文件是/etc/inetd.conf， 如果想inetd 去重□文件系□的□，可以□它□一个SIGHUP 信号。

1. □□□要□送信号的□程ID，可以用ps(1) 加 grep(1)来完成。 grep(1)命令被用在搜索□出方面，搜索□指定的字符串。 □命令是由普通用□来□行的，而inetd(8)是root用□□行的， 所以必□□ps(1)□上ax□□。

   ```
   % ps -ax | grep inetd
   198  ??  IWs    0:00.00 inetd -wW
   ```

   得出 inetd(8) PID号是198。 有□ grep inetd 命令也出□在□出中， □是因□在□方面 ps(1) 也是□□列表中□行□程。

2. 使用 kill(1) 去□送信号。 因□ inetd(8) 是由 root□□的， □必□使用 su(1) 去 □□ root 用□。

   ```
   % su
   Password:
   # /bin/kill -s HUP 198
   ```

   和大多数 UNIX® 命令一□， kill(1) 如果完成了任□， 就不会□出任何消息。 假如□□送信号□一个不属于□的□程， □会看到 kill: PID: Operation not permitted. 假如□□了PID号，把信号□送到其他□程，那是坏事。 或者□□幸，把信号□送到不存在的□程， □会看□ kill: PID: No such process.

   □什□使用 /bin/kill?

   □多shell提供了内建 kill 命令， □□， shell就能直接□送信号，而不是□行/bin/kill。 □点非常有用， 但不同shell有不同的□法来指定□送信号的名字，与其□□把它□学完倒不如□□地直接使用 /bin/kill …。

□送其他的信号也很相似， 只要在命令行替□ TERM 或 KILL 就行了。

在系□上随意□死□程是个坏主意，特□是init(8)， 它的□程ID是1，它非常特殊。可以□行 /bin/kill -s KILL 1 命令来□系□迅速□机。 当□按下 Return （回□）□之前， 一定要□□□□□行 kill(1) □所指定的参数。

4.9. Shells

在FreeBSD里，□日有一大堆工作是在命令行的界面完成的,那就叫做shell。 一个shell的主要功能就是从□入取得命令然后去□行他。 □多的shell同□能□我□完成内建的□日功能，例如:文件管理、文件□□、命令行□□、宏指令和□境□量。FreeBSD内含了一些shell，例如:sh、Bourne Shell、 tcsh和改良□的C-shell。 □外也有些shell也可在FreeBSD的Ports得到，例如:zsh和bash。

□想使用□一□shell取决于□的喜好， 假如□是C程序□□□，□可能□□一个C-like shell例如tcsh。 假如□是从Linux□来的或是一个命令行的新手，□可能会□一下bash。 □一点告□我□□一个shell都有各自的特性，可能□用于□的工作□境，也可能不□用于□的工作□境。

□个shell都有一个共通点就是文件名□全。 □入命令或文件名的前几个字，然后按Tab□，就能□shell的自□□全功能得出 命令或文件名。□里有一个例子，假□□有□个文件叫 foobar 和foo.bar，而□想□除 foo.bar，可□□在□□上□入 rm fo[Tab].[Tab]。

那□shell就会□出 rm foo[BEEP].bar。

□个[BEEP] 是□控制台□声， 那个是告□我□它不能完成文件名□全，因□有多个文件名符合。 foobar 和 foo.bar 都是以 fo□□， 它只可以□全到 foo。 □入 .并再按一次 Tab，shell才把其余的文件名全部□示出来。

□一个特点就是shell利用□境□量□行。□境□量是□存在shell□境空□上相□□的□和可□□， □个空□能□□程序从shell里□出，而且包含了□多程序的配置。 □个一个常用□境□量列和其含□的列表：

□量	□明
USER	当前登□□入的用□名。
PATH	搜索程序路径，以□点的冒号分隔□。
DISPLAY	假如有□个□量的□，就是X11□示器的网□名称。
SHELL	当前所用的shell。
TERM	用□□端的名字，通常用在□定□端的能力。
TERMCAP	各□□端功能所用□端分□□□的基本数据□目。
OSTYPE	操作系□□型，默□是FreeBSD。
MACHTYPE	是指系□上□行的CPU体系□□。
EDITOR	用□首□的文本□□器。
PAGER	用□首□的文本□面□度程序 。
MANPATH	搜索□机手册路径，以□点的冒号分隔□。

不同的shell□置□境□量也不相同。□个例子， 在如tcsh 和 csh□□的C-Style shell， □必□使用setenv去□置□境□量。 而在如sh和bash□□的Bourne shell， □必□使用export去□置当前□境□量。 再□个例子，要去□置或改□EDITOR□境□量， 在csh或tcsh下将EDITOR□□ /usr/local/bin/emacs:

```
% setenv EDITOR /usr/local/bin/emacs
```

而在Bourne shell下，□是:

```
% export EDITOR="/usr/local/bin/emacs"
```

□也可以在命令行上加一个$字符在□量之前从而取得□境□量。 □个例子，用echo $TERM 就会□示出$TERM的□定□， 其□就是shell取得$TERM并□□echo来□示的。

shell里有□多特□的字符代表着特□的□料，我□把叫做meta-characters。 最常用的就是

*字符，它可代表文件名的任何字符。 □些特□字符□用到文件名全域方面。假如，□入 echo *和□入 ls的效果是相同的，其□就是 shell 取得了全部符合 *的文件名，并□□ echo 在命令行下□示出来。

□了防止shell去分析□些特□字符， 我□可在它之前加一个 \字符去□明它只是普通字符。 echo $TERM就会□示出□的□端情况， 而 echo \$TERM 就会□示出 $TERM □几个字。

4.9.1. 改□□用的Shell

改□□的Shell的最□□方法是使用 chsh 命令。 □行 chsh 将根据□□定的EDITOR □境□量□入到那个□□器，假如没有□定，就会□入vi□□器。 □改□"Shell:"□行□□□。

□可使用chsh 的-s□□， □□就能□置□的shell却又不用□□器。假如□想把shell改□bash 可用下面的技巧。

```
% chsh -s /usr/local/bin/bash
```

□使用的shells必□ 在/etc/shells 文件里列出。 假如□从 ports里装一个shell，那就不用做□□了。 假如□手工装一个shell，那就要手工添加□去。

□个例了子，假如□手工把 bash装到 /usr/local/bin里，□□要□行□一□:

```
# echo "/usr/local/bin/bash" >> /etc/shells
```

然后□行chsh。

4.10. 文本□□器

FreeBSD 的很多配置都可以通□□□文本文件来完成。 因此， 最好能熟悉某□文本□□器。 FreeBSD 基本系□中提供了一些， □也可以从 Ports Collection 安装其它□□器。

最容易学的而又□□的□□器是 ee□□器， 是个□准的□易□□器。 要□□ ee，首先就要在命令行□入 ee filename，*filename* 是一个要□□的文件名。 例如，要□□ /etc/rc.conf就要□入 ee /etc/rc.conf，在 ee的控制内，□□器所有功能的操作方法都□示在最上方。 □个^ 字符代表 □□上的Ctrl □， 所以^e 就是 Ctrl + e □合□。假如想□□ee， 按Esc □，就可□□□□□□器。 当□修改了内容的□候，□□器会提示□保存。

FreeBSD本身也□□可多有□大功能的文本□□器， 例如 vi。□有其他在FreeBSD Ports里几□， 像 emacs 和 vim。 □些□□器有着□大的功能，但同□学□起来比□□□。 不管□□，假如□从事文字□□方面的工作， 学□如vim 或 emacs □些有□大功能的□□器用法， 在□□□工作里会□□□省不少的□□。

很多需要修改文件或打字□入的□用程序都会自□打□一个文本□□器。 更改默□使用的□□器， □□置 EDITOR □境□量。 参□ shells 以□取更多□□信息。

4.11. □□和□□□点

在一个系□里，硬件描述通常用法就是一个□□□□一个□□，包括磁□、打印机、□□和□□。 当 FreeBSD □□□程中，大多数的□□都能探□到并□示出来， □也可以□□/var/run/dmesg.boot， 引□□所有信息都在里面。

例如， acd0 就是 首个 IDE 光□□□， 而 kbd0 □代表□□。

在UNIX®操作系□里，大多数□□存在的特殊□□文件就是叫做□□□点， 他□都定位在/dev目□里。

4.11.1. 建立□□□点

当在系□中添加新□□或将附加□□的支持□□□内核之后， 都必□□其建立□□□点。

4.11.1.1. DEVFS (DEVice 文件系□)

□个□□文件系□， 或叫 DEVFS， □内核的□□命名在整体文件系□命名里提供通道， 并不是建立或更改□□□点，DEVFS只是□□的特□文件系□□行□□。

□参□ devfs(5) □机手册以了解更多□□。

4.12. 二□制文件格式

要理解□什□ FreeBSD 使用 elf(5) 格式， □必□首先了解一些 UNIX® 系□中的 三□ "主要" 可□行文件格式的有□知□：

- a.out(5)

  是最古老和"□典的" UNIX® 目□文件格式， □□格式在其文件的□始□有一个短小而又□□的首部， □首部□有一个魔幻数字，用来□□具体的格式(更多□情参□a.out(5))。 □□格式包含3个要装□入内存的段：.text，.data， 和 .bss，以及 一个符号表和一个字符串表。

- COFF

  SVR3目□文件格式。其文件□□在包括一个区段表(section table)， 因此除了.text，.data，和.bss区段以外，□□可以包含其它的区段。

- elf(5)

  COFF 的后□， 其特点是可以有多个区段， 并可以使用32位或64位的□。 它有一个主要的缺点： ELF在其□□□假□□个系□体系□□只有一□ ABI。 □□假□事□上相当□□， 甚至在商□化的SYSV世界中都是□□的 (它□至少有三□ABI: SVR4, Solaris, SCO)。

  FreeBSD□□在某□程度上解决□个□□，它提供一个工具，可以 □一个已知的ELF可□行文件 □□它所遵从的ABI的信息。 更多□方面的知□可以参□手册□brandelf(1)

FreeBSD从"□典"□□中来，因此使用了a.out(5)格式， □多BSD版本的□行(直到3.X分支的□始)也□明了□□格式的有效性。 □然在那以前的某段□□，在FreeBSD系□上□建和□行ELF格式 的二□制可□行文件(和内核)也是可能的，但FreeBSD一□始并不□□"□□" 到使用ELF作□其缺省的格式。□什□？□，当Linux□□完成了 □□到ELF格式的痛苦□程后，却□□并不足以由此而放□ a.out可□行文件格式，因□正是由于它□不□活的， 基于跳□表的共享□机制，使得□售商和□□者□□建共享□非常困□。 直到已有的ELF工具提供了一□解决共享□□□的□法，并被普遍□□是"前□方向"以后，□□的代价在FreeBSD界才被接受， 并由此完成了□□。FreeBSD的共享□机制其基□更□似于Sun SunOS™的共享□机制， 并且正因□此，其易用性很好。

那□，□什□会有□□多不同的格式□？

回溯到蒙昧和□暗的□去，那□只有□□的硬件。□□□□的硬件支□了一个□□ 和小型的系□。在□□的□□系□上(PDP-11)a.out格式 足以□任表□二□制文件的任□。当人□将UNIX®从□□□□的系□中移植出来的□候，

a.out格式被保留了下来，因□□于早期将UNIX®移植到 Motorola 68k，VAXen等系□来□，它□是足□可用的。

然后，一些□明的硬件工程□□□，如果可以□□件完成一些□□的□明操作，　　那□他□就可以在硬件□□中□少若干□□路，并可以□CPU核心□行得更快。　当a.out格式用于□□新型的硬件系□□(□在我□叫它　RISC)，□得并不合□。因此，人□□□了□多新的格式　　以便在□□的硬件系□上能□得比□□的a.out格式更□越　　的性能。□如COFF，ECOFF，□有其它　　一些晦□□□的格式正是在□个□段被□明出来的，人□也研究了□些格式的局限性，慢慢地最□落□到ELF格式。

同□，程序的大小□得越来越大，磁□空□(以及物理内存)相□来□却仍然□小，　　因此共享□的概念便□生了。VM系□也□得越来越□□了。当所有□些□□都建立在　　a.out格式的基□上的□候，它的可用性随着□个新特性　　的□生就受到了□重考□。并且，人□□希望可以在□行□□□装□某些□西，或者　　在初始化代□□行以后可以□□部分程序代□，以便□□主存□器和交□区。□程□言　　也□得越来越□□，人□希望可以在main()函数□行之前自□□行某些代□。□了□□　　所有□些功能，人□□a.out格式作了很多改□(hack)，　　他□在某个□段里基本也是可行的。随着□□的推移，a.out格式　不得不□加大量的代□和□□度来□足□些需求。□然ELF格式解决了□多□□的□□，但是从一个可用的系□□移到□一个系□却是痛苦的。因此　直到□□保留a.out格式的代价比□移到ELF格式 的代价□大的□候，人□才会最□□□到ELF格式。

然而，随着□□的推移，FreeBSD系□本身的□□工具(特□是□□器和装□器)　　□以派生的□□工具，其□展却形成了□个平行的分支。FreeBSD□个分支□加了共享□，　　并修改了一些□□。而原先□写了□些工具的GNU人□重写了□些工具，并□交叉□□提供了　　更□化的支持，□随意□入了不同格式的支持，等等。□然很多人希望□建□□FreeBSD的　交叉□□器，但他□却并未如□以□，因□FreeBSD的as　和ld的源代□更□老旧，所以无法完成□个任□。　新的GNU工具□(binutils)□□□支持交叉□□，ELF　格式，共享□，C++□展，等等。并且，由于很多供□商都□布ELF格式的 二□制文件，因而□FreeBSD能□□行它□将是一个很好的事情。

ELF格式比a.out格式□□要大些，同□也　　允□基□系□有更好的□展性。ELF格式的有□工具有着更好的□□，并且提供交叉□□支持，□□□多人来□是很重要的。ELF格式可能会□微　　慢一些，但很□□量出来。□外，在□□者之□，有□多□□也是不同的，比如它□映射□面的方式，　　□理初始化代□的方式，等等。所有□些都不太重要，但□也□□是不同之□。在将来的□当□候，　　GENERIC内核将不再支持a.out格式，并且，当不再需要□行□留的a.out格式程序□，内核也将不再提供□其的支持。

4.13. 取得更多的□□

4.13.1. □机手册

最□□的使用□明文□莫□于　FreeBSD　里的□机手册了。　几乎□一个程序都会附上一□□短□明，　以介□□个程序的的基本功能以及参数的用法。 我□能通□ `man` 命令来□□□些□明， 而使用 `man` 命令却是□□的事情:

```
% man command
```

`command` 就是□要了解的命令命称。 □个例子，想了解 `ls` 命令就□入:

```
% man ls
```

□些在□手册分下列章□:

1. 用□命令。

2. 系□□用以及□□代□。
3. C □文件里的函数□明。
4. □□□□程序。
5. 文件格式。
6. 游□以及其他□□。
7. 各□□□。
8. 系□□□以及命令。
9. 内核□□情况。

在某些情况下，同□的主□也会出□在在□手册的不同章□。 □个例子，系□里有chmod□个用□命令，而又有个chmod() 系□□用。 在□□情形下，□□当向 man 命令指定需要的内容:

```
% man 1 chmod
```

□□就会□示出手册里的用□ chmod 命令。 □□上，我□在写入文□□把特定□□参考内容在在□手册括号里注明。所以 chmod(1) 是指 chmod 用□命令， 而 chmod(2) 是指系□□用。

如果□已□知道命令的名字，只是不知道要□□使用的□，那就比□好□。 但□□名字都不知道□?□个□候□就可以利用 man 的搜□功能， 它会在手册的介□部□□□□要搜□的□□字，它的□□是 -k：

```
% man -k mail
```

当□使用□个命令的□候，man会把介□里含有"mail"□□字 的命令列出来，□□上□和apropos命令的功能是相同的。

有□□会看到/usr/bin 下有□多命令但不知他□的用途， □只需□□做:

```
% cd /usr/bin
% man -f *
```

或者□□做

```
% cd /usr/bin
% whatis *
```

□个命令是一□的。

4.13.2. GNU Info 文件

FreeBSD□多□用□件以及□用工具来自Free□件基金会(FSF)。 作□手册的□充，□些程序提供了一□更具有活力的超文□□明info， □可用info命令来□□他□。 假如□装上emacs，也能利用emacs 的info模式来□□。

使用 info(1) □个命令只需□□地□入：

```
% info
```

想得到□□介□， □按 h。 想快速得到的命令□明， □按 ?。

Chapter 5. 安装□用程序: Packages 和 Ports

5.1. 概述

FreeBSD 将□多系□工具□□作□基本系□的一部分。 然而， 要完成□□的工作， 可能□需要安装更多的第三方□用。 FreeBSD 提供了□□□充的技□， 用以在□的系□中安装第三方□件： FreeBSD Ports 套件 (用于从源代□安装)， 以及 packages (用以从□□□的二□制版本安装)。 □□□方法都可以用于从本地介□，或从网上直接安装□喜□的□用程序的最新版本。

□完□章，□将了解到:

- 如何安装第三方的二□制□件包。
- 如何使用 ports 套件从源代□□建第三方□件。
- 如何□除先前安装的□件包。
- 如何改□Ports Collection里面的一些参数，定制□件使用。
- 如何□到□需要的□件包。
- 如何升□□的□用□件。

5.2. □件安装□□

如果□以前使用□ UNIX® 系□，那典型的第三方□件安装的□□是像下面描述的：

1. 下□□个□件，□件的□行版可能是源代□格式，或是一个二□制包。
2. 解□□件(其中代表性的是用 compress(1), gzip(1), 或 bzip2(1) □□□的tar包)。
3. □□相□文□，了解如何安装。 (多半一个文件名是INSTALL或README， 或在doc/ 目□下的一些文□)
4. 如果□件是以源代□形式□布的，那就需要□□它。可能需要□□一个 Makefile文件， 或□行 `configure`脚本，和其他的一些工作。
5. □□和安装□件。

如果一切□利的□，就□□□□。如果□在安装一个□件包□□生一些□□， □可能需要□□一下它的代□，以使它能正常工作。

□可以□□使用 "□□的"方式安装□件。 然而， FreeBSD 提供了□□技□： packages 和 ports。 就在写□篇文章的□候， 已□有超□ 36000 个第三方的□用程序可以使用了。

□于任意一个□用程序包，是一个可以下□的FreeBSD package文件。□个 FreeBSD package包含了□□好的的副本， □有一些配置文件或文□。 一个下□的包文件可以用 FreeBSD 的包管理命令来操作， 例如 pkg_add(1)， pkg_delete(1), pkg_info(1) 等等。 可以使用一个□□的命令安装一个新的□用程序。

一个FreeBSD的port是一个可以自□从源代□□□成□用程序的文件集合。

□住，如果□自己来□□的□，需要□行很多□的操作 (解□， □丁， □□， 安装)。 □些整理 port 的文件集合包含了系□需要完成□个工作的必需信息。 □可以□行一些□□的命令， 那些源代□就可以自□地下□，

解□， 打□丁， □□， 直至安装完成。

□□上，ports 系□也能做出被 pkg_add 的程序包和不久就要□到的其他包管理命令来安装的□件包。

Packages 和 ports 是互相 依□ 的。 假□□想安装一个依□于已□安装的特定□的□用程序。 □用程序和那个□都已□□用于 FreeBSD ports 和 packages。 如果□使用 命令或 ports 系□来添加□用程序， □个都必□注意□是否被安装， 如果没有， 它会自□先安装□。

□里□出的□□技□是很相似的，□可能会奇怪□什□ FreeBSD 会弄出□□□技□。 其□， packages 和 ports 都有它□自己的□□， 使用□一□完全取决于□自己的喜好。

Package Benefits

- 一个□□的 package 通常要比一个□□的包含源代□的□用程序小得多。
- package 不需要□行□外的□□。 □于大型□用程序如 Mozilla， KDE 或 GNOME 来□□□得尤□重要， 特□是在□的系□□源比□差的情况下。
- package不需要□知道如何在FreeBSD上□□□件的□□□程。

Ports Benefits

- package 在□□□通常使用比□保守的□□， □是□了保□它□能□□行在大多数的系□上。 通□从 port 安装，□可以□微□整□□□□来□生□合于□理器的代□ (□□于 Pentium 4 或 AMD 的 Athlon CPU)。
- 一些□件包已□把与它□相□的能做和不能做的事情的□□都□□□去了。 例如， Apache 可能就配置了很多的□□。 从 port 中安装□， □不一定要接受默□的□□， 可以自己来□置。

  在一些例子中，一个□件有不同的配置存在多个package。 例如， Ghostscript存在 ghostscript package 和 ghostscript-nox11 package□个配置package， □取决于□是否安装了X11服□器。 □□的□整□package是可能的， 但如果一个□用程序有超□一个或□个不同的□□□□□□□， 就不行了。

- 一些□件的□可条件禁止采用二□制形式□行。 它□必□□上源代□。
- 一些人不信任二□制□行形式。 至少有了源代□， (理□上) 可以□自□□它，□□潜在的□□。
- 如果□要自己□□件打□丁，□就需要有源代□。
- 一些人喜□整天□着源代□□， 所以他□喜□□自□□源代□， 修改源代□等等。

保持更新 ports， □□□件列表 FreeBSD ports □件列表 和□交□□□告 FreeBSD ports bugs □件列表。

安装任何□用程序之前， □首先□□ http://vuxml.freebsd.org/ 上是否有□于□所安装的□用程序的安全□□□告。

□也可以安装 ports-mgmt/portaudit， 它能□自□地□□已□安装的□用程序的漏洞； 此外，在□安装程序之前它也会首先□□是否存在已知的漏洞。 □外， □也可以使用 portaudit -F -a □个命令在安装了某个□件包之后作出□□。

□章的其余部分将介□在 FreeBSD 上如何使用 packages 和 ports 来安装和管理第三方□件。

5.3. □□□要的□用程序

在□安装任何□用程序之前，需要知道□需要什□，那个□用程序叫什□。

FreeBSD中可用的□用程序正在不断地□□着。幸□的是， 有□多方法可以□到□所需要的程序:

- FreeBSD站点上有一个可以搜索到的当前所有可用的□用程序列表，在 http://www.FreeBSD.org/ports/。 它分很多□□，□既可以通□程序的名称来搜索(如果□知道名字)，也可以在分□中列出所有可用的□用程序。
- Dan Langille □□着网站 FreshPorts，在 http://www.FreshPorts.org/。 FreshPort□刻 "追踪" 着在 ports 中□用程序的□化。当有任何程序被升□□，他□就会□ email 提醒□。
- 如果□不知道□想要的□用程序的名字，可以通□ (http://www.freshmeat.net/) 网站来□□， 如果□到了□用程序， □可以回 FreeBSD 的主站去看一下□个□用程序是否已□被 port □去了。
- 如果□知道一个port的准□名字， 但需要知道在□个□□里面能□到它，□可以使用 whereis(1) □个命令。□□地□入 `whereis file`， *file* 就是□想安装的程序名字。 如果系□□到了它， □将被告知在它在□里， 例如:

  ```
  # whereis lsof
  lsof: /usr/ports/sysutils/lsof
  ```

  □果告□我□□个命令`lsof` (一个系□配置程序)可以在 /usr/ports/sysutils/lsof目□中□到。

- □可以使用□□的 echo(1) □句来□□某个 port 是否存在于 ports □中。 例如：

  ```
  # echo /usr/ports/*/*lsof*
  /usr/ports/sysutils/lsof
  ```

  Note that this will return any matched files downloaded into the /usr/ports/distfiles directory.

  □注意□条命令将会返回下□到 /usr/ports/distfiles 目□中所有符合条件的文件。

- □有□外的一个□□□需要的port的方法—是用ports collecton 内嵌的搜索机制。要使用□个搜索, □需要先到/usr/ports目□下面。 在那个目□里面， □行`make search name=program-name`， *program-name* 就是□想□□的程序名字。 □个例子， 如果□想□ `lsof`：

  ```
  # cd /usr/ports
  # make search name=lsof
  Port:   lsof-4.56.4
  Path:   /usr/ports/sysutils/lsof
  Info:   Lists information about open files (similar to fstat(1))
  Maint:  obrien@FreeBSD.org
  Index:  sysutils
  B-deps:
  R-deps:
  ```

  在□出的内容里面□要特□注意包含 "Path:" 的□行将告□□在□里可以□到□个 port。 如果要安装此 port，那其他□出的信息不是必□的， 但是□是□示□出了。

  □了更深入的搜索，□□可以用 `make search key=string`， *string*就是□想搜索的部分内容。 它将搜索port的名字、 注□， 描述和从属□系， 如果□不知道□想搜索的程序名字， 可以利用它搜索一些□□主□来□到□需要的。

上面的些方法， 搜索的字没有大小写区分的。 搜索 "LSOF"的果将和搜索"lsof"的果一。

5.4. 使用 Package 系

在 FreeBSD 系上有几不同的工具用来管理 package：

- `sysinstall` 工具可以在正在行的系上行， 以完成安装、 除和列出可用的以及已安装的件包的任。 如欲了解一信息， 参 安装的件包 (package)。
- 一余下的部分将介用于管理件包的命令行工具。

5.4.1. 一个 package 的安装

可以用 pkg_add(1) 个命令从本地文件或网上的服器来安装一个 FreeBSD 件包。

例 8. 在本地手下一个*package*,并安装它

```
# ftp -a ftp2.FreeBSD.org
Connected to ftp2.FreeBSD.org.
220 ftp2.FreeBSD.org FTP server (Version 6.00LS) ready.
331 Guest login ok, send your email address as password.
230-
230-     This machine is in Vienna, VA, USA, hosted by Verio.
230-         Questions? E-mail freebsd@vienna.verio.net.
230-
230-
230 Guest login ok, access restrictions apply.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd /pub/FreeBSD/ports/packages/sysutils/
250 CWD command successful.
ftp> get lsof-4.56.4.tgz
local: lsof-4.56.4.tgz remote: lsof-4.56.4.tgz
200 PORT command successful.
150 Opening BINARY mode data connection for 'lsof-4.56.4.tgz' (92375 bytes).
100% |***********************************************| 92375       00:00 ETA
226 Transfer complete.
92375 bytes received in 5.60 seconds (16.11 KB/s)
ftp> exit
# pkg_add lsof-4.56.4.tgz
```

如果没有本地package的安装 (如 FreeBSD CD-ROM)， 可以行 pkg_add(1) 命令并加上 `-r` 。 将迫使程序自决定目文件的正格式和版本， 然后自从一个 FTP 站点和安装 package。

```
# pkg_add -r lsof
```

上面的例子将下正的package， 而不需要用的干就可以安装。 如果想指定 FreeBSD package 的

□像站点， 替□主站点， 就必□相□地□置 PACKAGESITE □个□境□量， 覆□原来的□置。 pkg_add(1) 使用 fetch(3) 下□文件， 可以使用多□□境□量， 包含 FTP_PASSIVE_MODE、 FTP_PROXY， 和 FTP_PASSWORD。 如果□使用 FTP/HTTP 代理或在防火□后面， □可能需要□置□些□境□量。 □□的列表□参考 fetch(3)。上述例子中用 lsof 替代了 lsof-4.56.4。 当使用□程安装 Package 的□候□件名字不需要包含版本号。 pkg_add(1) 将自□的□到□个□件最新的版本。

如果□使用 FreeBSD-CURRENT 或 FreeBSD-STABLE版本的FreeBSD， pkg_add(1) 将下□□的□用□件的最新版本。 如果□使用 -RELEASE 版本的 FreeBSD， 它将会□得与□的版本相□的□件包版本。 □可以通□修改□境□量 PACKAGESITE 来改□□一行□。 例如， 如果□□行 FreeBSD 8.1-RELEASE 系□， 默□情况下 pkg_add(1) 将□□从 ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-8.1-release/Latest/ 下□□□□的□件包。 如果□希望□制 pkg_add(1) 下□ FreeBSD 8-STABLE 的□件包， □可以将 PACKAGESITE □置□ ftp://ftp.freebsd.org/pub/FreeBSD/ports/i386/packages-8-stable/Latest/。

□件包采用 .tgz 和 .tbz □□格式。□可以在 ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/packages/ 下面或从 FreeBSD 的□行光□□到， 它在□一个 4CD 的 FreeBSD □行版的 /packages目□中。 □件包的□□□□与 /usr/ports □一致。 □个分□都有自己的目□， 所有的□件包可以在目□ All中□到。

□件包系□的目□□□与ports的□□□□一致； 它□共同□成了整个 package/port。

5.4.2. □件包的管理

pkg_info(1) 是用于列出已安装的所有□件包列表和描述的程序。

```
# pkg_info
cvsup-16.1          A general network file distribution system optimized for CV
docbook-1.2         Meta-port for the different versions of the DocBook DTD
...
```

pkg_version(1)是一个用来□□所有安装的□件包版本的工具。 它可以用来比□本地 package 的版本与 ports 目□中的当前版本是否一致。

```
# pkg_version
cvsup                           =
docbook                         =
...
```

在第二列的符号指出了安装版本的相□□□和本地ports目□□中可用的版本。

符号	含□
=	在本地ports□中与已安装的□件包版本相匹配。
<	已安装的版本要比在ports□中的版本旧。
>	已安装的版本要比在ports□中的版本新 (本地的 port□可能没有更新)。

符号	含□
?	已安装的□件包无法在ports索引中□到。(可能□生□□事情，□个例子，□早先安装的一个 port 从 port □中移出或改名了)
*	□件包有很多版本。
!	已安装的□件包在索引中存有□□，但是由于某些原因 `pkg_version` 无法比□已安装的□件包与索引中相□□的版本号。

5.4.3. □除一个□件包

要□除先前安装的□件package，只要使用pkg_delete(1) 工具。

```
# pkg_delete xchat-1.7.1
```

需要注意的是，pkg_delete(1) 需要提供完整的包名；如果□只是指定了□似 *xchat* 而不是 *xchat-1.7.1* □□的名字，□它将拒□□行操作。不□，□可以使用 pkg_version(1) 来了解安装的 package 的版本。除此之外，也可以使用通配符：

```
# pkg_delete xchat\*
```

□□，所有名字以 `xchat` □□的 package 都会被□掉。

5.4.4. 其它

所有已安装的 package 信息都保存在 /var/db/pkg 目□下。安装文件的列表和□个 package 的内容和描述都能在□个目□的相□文件中□到。

5.5. 使用Ports Collection

下面的几个小□中，□出了□于如何使用 Ports 套件来在□的系□中安装或卸□程序的介□。□于可用的 `make` targets 以及□境□量的介□，可以在 ports(7) 中□到。

5.5.1. □得Ports Collection

在□能使用 ports 之前，□必□先□得 Ports Collection - 本□上是 /usr/ports 目□下的一堆 Makefile、□丁和描述文件。

在□安装 FreeBSD 系□的□候，sysinstall 会□□□是否需要安装 Ports Collection。如果□□□ no，那□可以用下面的指令来安装 Ports Collection：

Procedure: CVSup 方法

保持□本地 Ports 套件最新的一□快捷的方法， 是使用 CVSup □□来□行更新。 如果□希望了解更多□于 CVSup 的□□， □参□ 使用 CVSup。

在 FreeBSD 系□里□ CVSup 的□□叫作 csup。

在首次□行 csup 之前， □必□□ /usr/ports 是空的！ 如果□之前已□用其他地方安装了一□ Ports 套件， □ csup 可能不会自□□除已□在上游服□器上□除掉的□丁文件。

1. □行 `csup`:

```
# csup -L 2 -h cvsup.FreeBSD.org /usr/shared/examples/cvsup/ports-supfile
```

将 *cvsup.FreeBSD.org* 改□□□□近的 CVSup 服□器。 □参□ CVSup □像 (CVSup 站点) 中的□像站点完整列表。

有□可能希望使用自己的 ports-supfile，比如□， 不想□次都通□命令行来指定所使用的 CVSup 服□器。

a. □□情况下， 需要以 `root` 身□将 /usr/shared/examples/cvsup/ports-supfile □制到新的位置， 例如 /root 或□的主目□。
b. □□ ports-supfile。
c. 把 *CHANGE_THIS.FreeBSD.org* 修改成□□□近的 CVSup 服□器。可以参考 CVSup □像 (CVSup 站点) 中的□像站点完整列表。
d. 接下来按如下的方式□行 `csup`：

```
# csup -L 2 /root/ports-supfile
```

2. 此后□行 csup(1) 命令将下□最近所□行的改□， 并将它□□用到□的 Ports Collection 上， 不□□一□程并不重新□□□系□上的 ports。

Procedure: Portsnap 方式

Portsnap 是用于□布 Ports 套件的□一套系□。 □参□ 使用 Portsnap 以了解□于 Portsnap 功能更□□的介□。

1. 下□□□的 Ports 套件快照到 /var/db/portsnap。 □可以根据需要在□之后□□ Internet □接。

   ```
   # portsnap fetch
   ```

2. 假如□是首次□行 Portsnap， □需要将快照□放到 /usr/ports：

   ```
   # portsnap extract
   ```

   如果□已□有装好的 /usr/ports 而□只想更新， □□□行下面的命令：

   ```
   # portsnap update
   ```

Procedure: Sysinstall 方式

□□方法需要使用 sysinstall 从安装介□上安装 Ports 套件。 注意， 安装的将是□布□行版□的旧版 Ports 套件。 如果□能□□ Internet， □使用前面介□的方法之一。

1. 以 `root` 身□□行 `sysinstall`：

   ```
   # sysinstall
   ```

2. 用光□向下□□ Configure， 并按 `Enter`。
3. 向下并□□ Distributions， 按 `Enter`。
4. □□ ports， 并按 `Space`。
5. □□ Exit， 并按 `Enter`。
6. □□所希望的安装介□， 例如 CDROM、 FTP， 等等。
7. □□ Exit 并按 `Enter`。
8. 按 `X` 退出 sysinstall。

5.5.2. 安装 Ports

当提到 Ports Collection □， 第一个要□明的就是何□ "skeleton"。 □□地□， port skeleton 是□一个程序在 FreeBSD 上□□地□□并安装的所需文件的最小□合。 □个 port skeleton 包含：

- 一个 Makefile。 Makefile 包括好几个部分， 指出□用程序是如何□□以及将被安装在系□的□些地方。

- 一个 distinfo 文件。□个文件包括□些信息： □些文件用来□下□后的文件校□和□行□□ (使用 sha256(1))，来□保在下□□程中文件没有被破坏。
- 一个 files 目□。 □个目□包括在 FreeBSD 系□上□□和安装程序需要用到的□丁。 □些□丁基本上都是些小文件， 指出特定文件作了□些修正。 它□都是□文本的的格式，基本上是□□的 "□除第 10 行" 或 "将第 26 行改□□□ ..."， □丁文件也被称作 "diffs"， 他□由 diff(1) 程序生成。

  □个目□也包含了在□□ port □要用到的其它文件。

- 一个 pkg-descr 文件。 □是一个提供更多□□，有□件的多行描述。
- 一个 pkg-plist 文件。 □是即将被安装的所有文件的列表。它告□ ports 系□在卸□□需要□除□些文件。

一些ports□有些其它的文件， 例如 pkg-message。 ports 系□在一些特殊情况下会用到□些文件。 如果□想知道□些文件更多的□□以及 ports 的概要， □参□ FreeBSD Porter's Handbook。

port里面包含着如何□□源代□的指令， 但不包含真正的源代□。 □可以在网上或 CD-ROM 上□得源代□。 源代□可能被□□者□布成任何格式。 一般来□□□是一个被 tar 和 gzip □的文件， 或者是被一些其他的工具□□或未□□的文件。 ports中□个程序源代□□示文件叫 "distfile"， 安装 FreeBSD port的方法□不止□□□。

□必□使用 `root` 用□登□后安装 ports。

在安装任何 port 之前， □□首先□保已□更新到了最新的 Ports Collection， 并□□ http://vuxml.freebsd.org/ 中是否有与那个 port 有□的安全□□。

在安装□用程序之前， 可以使用 portaudit 来自□地□□是否存在已知的安全□□。 □个工具同□可以在 Ports Collection (ports-mgmt/portaudit) 中□到。 在安装新的 port 之前， 可以考□先□行一下 `portaudit -F` 来□取最新的漏洞数据□。 在□天的周期性系□安全□察□， 数据□会被自□更新， 并且会在□之后□施安全□□。 欲了解□一□的情况，□参□ portaudit(1) 和 periodic(8)。

Ports 套件假定□有可用的 Internet □接。 如果□没有， □需要将 distfile 手工放到 /usr/ports/distfiles 中。

要□始操作， 首先□入要安装 port 的目□：

```
# cd /usr/ports/sysutils/lsof
```

一旦□入了 lsof 的目□，□将会看到□个port的□□。 下一□就是 make，或□ "□□" □个 port。 只需在命令行□□地□入 `make` 命令就可□松完成□一工作。 做好之后，□可以看到下面的信息：

```
# make
>> lsof_4.57D.freebsd.tar.gz doesn't seem to exist in /usr/ports/distfiles/.
>> Attempting to fetch from ftp://lsof.itap.purdue.edu/pub/tools/unix/lsof/.
===>  Extracting for lsof-4.57
...
[extraction output snipped]
...
>> Checksum OK for lsof_4.57D.freebsd.tar.gz.
===>  Patching for lsof-4.57
===>  Applying FreeBSD patches for lsof-4.57
===>  Configuring for lsof-4.57
...
[configure output snipped]
...
===>  Building for lsof-4.57
...
[compilation output snipped]
...
#
```

注意，一旦□□完成，□就会回到命令行。 下一□安装 port， 要安装它只需要在 `make` 命令后跟上一个□□ `install` 即可：

```
# make install
===>  Installing for lsof-4.57
...
[installation output snipped]
...
===>   Generating temporary packing list
===>   Compressing manual pages for lsof-4.57
===>   Registering installation for lsof-4.57
===>  SECURITY NOTE:
      This port has installed the following binaries which execute with
      increased privileges.
#
```

一旦□返回到提示符，□就可以□行□□□安装的程序了。因□ `lsof` 是一个□予特殊□限的程序， 因此□示了一个安全警告。 在□□和安装 ports 的□候， □□□留意任何出□的警告。

□除工作目□是个好主意， □个目□中包含了全部在□□□程中用到的□□文件。 □些文件不□会占用宝□的磁□空□，而且可能会□升□新版本的 port □□来麻□。

```
# make clean
===>  Cleaning for lsof-4.57
#
```

使用 make install clean 可以一□完成 make、 make install 和 make clean □三个分□的□□的工作。

一些 shell 会□存□境□量 PATH 中指定的目□里的可□行文件， 以加速□□它□的速度。 如果□使用的是□□ shell， 在安装 port 之后可能需要□行 rehash 命令， 然后才能□行新安装的那些命令。 □个命令可以在□似 tcsh 的 shell 中使用。 □于□似 sh 的 shell，□□的命令是 hash -r。 □参□□的 shell 的文□以了解□一□的情况。

某些第三方 DVD-ROM □品， 如 FreeBSD Mall 的 FreeBSD Toolkit 中包含了 distfiles。 □些文件可以与 Ports 套件配合使用。 将 DVD-ROM 挂接到 /cdrom。 如果□使用不同的挂接点， □□□置 make □量 CD_MOUNTPTS。 如果□上有需要的 distfiles， □会自□使用。

□注意， 少数 ports 并不允□通□ CD-ROM □行。 □可能是由于下□之前需要填写注册表格，或者不允□再次□布， 或者有一些其它原因。 如果□希望安装在 CD-ROM 上没有的 port，就需要在□操作了。

ports 系□使用 fetch(1) 去下□文件， 它有很多可以□置的□境□量， 其中包括 FTP_PASSIVE_MODE、FTP_PROXY， 和 FTP_PASSWORD。 如果□在防火□之后，或使用 FTP/HTTP代理， □就可能需要□置它□。完整的□明□看 fetch(3)。

当使用者不是所有□□都能□接上网□， □可以利用 make fetch。 □只要在□□目□ (/usr/ports) 下□行□个命令，所有需要的文件都将被下□。 □个命令也同□可以在下□□□目□中使用， 例如： /usr/ports/net。 注意，如果一个port有一些依□的□或其他 port， 它将 不 下□□些依□的 port 的 distfile 文件， 如果□想□取所有依□的 port 的所有 distfile， □用 fetch-recursive 命令代替 fetch命令。

□可以在一个□□或在□□目□□□所有的 port， 或者使用上述提到的 make fetch命令。□□是非常危□的， 因□有一些port不能并存。 或者有□一□可能， 一些port会安装□个不同的文件， 但是却是相同的文件名。

在一些□□的例子中， 用□可能需要在除了 MASTER_SITES 以外的一个站点(本地已□下□下来的文件)去□得一个文件包。 □可以用以下命令不使用 MASTER_SITES:

```
# cd /usr/ports/directory
# make MASTER_SITE_OVERRIDE= \
ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/distfiles/ fetch
```

在□个例子中，我□把 MASTER_SITES□个□□改□了 ftp.FreeBSD.org/pub/FreeBSD/ports/distfiles/。

一些 port 允□ (或甚至要求) □指定□□□□来 □用/禁用 □用程序中非必需的功能， 一些安全□□， 以及其他可以□制的内容。 具有代表性的包括 www/mozilla、 security/gpgme、以及 mail/sylpheed-claws。 如果存在□□的□□， 通常会在□□□□出提示。

5.5.2.1. 改□默□的 Ports 目□

有□， 使用不同的工作□□目□和目□目□可能很有用 (甚至是必要的)。 可以用 WRKDIRPREFIX 和 PREFIX □□个□量来改□默□的目□。 例如：

```
# make WRKDIRPREFIX=/usr/home/example/ports install
```

将在 /usr/home/example/ports 中□□ port 并把所有的文件安装到 /usr/local。

```
# make PREFIX=/usr/home/example/local install
```

将在 /usr/ports □□它并安装到 /usr/home/example/local。

当然，

```
# make WRKDIRPREFIX=../ports PREFIX=../local install
```

将包含□□□置 (没有□法在□一□把它写完， 但□□□已□知道□□回事了)。

□外， □些□量也可以作□□境□量来□置。 □参考□的 shell 的□机手册上□于如何□置□境□量的□明。

5.5.2.2. □理 imake

一些 port 使用 imake (□是 X Window 系□的一部分) 不能正常地配合 PREFIX， 它□会□持把文件安装到 /usr/X11R6 下面。 □似地， 一些 Perl port 会忽略 PREFIX 并把文件安装到 Perl 的目□中。 □□些 port 尊重 PREFIX 是困□甚至是不可能的事情。

5.5.2.3. 重新配置 Ports

当□在□□某些 ports 的□候，可能会□出一个基于 ncurses 的菜□来□□来□□一些□□□□。 通常用□都希望能□在一个 port 被□□安装了以后□能再次□□□□菜□以添加□除或修改□些□□。 □□上有很多方法来做□件事情。 一个方法□入那个 port 的目□后□入 make config， 之后便会再次□示出菜□和已□□的□目。 □一个方法是用 make showconfig， □会□□□示出所有的配置□□。□有一个方法是□行 make rmconfig， □将□除所有已□□的□目。 有□□些□□更□□的内容□参□ ports(7)。

5.5.3. 卸□已□安装的 Ports

□在□已□了解了如何安装 ports， 并希望□一□了解如何卸□， 特□是在□□地安装了某个 port 之后。我□将卸□前面例子 (假如□没有注意的□， 是 lsof) 中安装的 port。 Ports 可以同 packages 以完全相同的方式 (在 Packages 一□ 中□行了介□) 卸□， 方法是使用 pkg_delete(1) 命令：

```
# pkg_delete lsof-4.57
```

5.5.4. 升□ Ports

首先， 使用 pkg_version(1) 命令来列出 Ports Collection 中提供了更新版本的那些 port：

```
# pkg_version -v
```

5.5.4.1. /usr/ports/UPDATING

在□更新了 Ports 套件之后， 在升□ port 之前， □□看 /usr/ports/UPDATING。 □个文件中介□了在升□□用□□注意的□□， 以及一些可能需要□行的操作。 □可能包括更改文件格式、 配置文件位置的□□，以及与先前版本的兼容性等等。

如果 UPDATING 与本□中介□的内容不同， □以 UPDATING □准。

5.5.4.2. 使用 Portupgrade 来更新 Ports

portupgrade 工具是□□来□化升□已安装的 port 的操作的。 它通□ ports-mgmt/portupgrade port 来提供。□可以像其它 port 那□， 使用 `make install clean` 命令来安装它：

```
# cd /usr/ports/ports-mgmt/portupgrade
# make install clean
```

使用 `pkgdb -F` 命令来□描已安装的 port 的列表， 并修正其所□告的不一致。 在□次升□之前， 有□律地□行它是个好主意。

□行 `portupgrade -a` □， portupgrade 将□始并升□系□中所安装的所有□□的 ports。 如果□希望在□个升□操作□得到□□， □指定 `-i` 参数。

```
# portupgrade -ai
```

如果□只希望升□某个特定的□用程序， 而非全部可用的 port， □使用 `portupgrade pkgname`。 如果 portupgrade □首先升□指定□用程序的□， □□指定 `-R` 参数。

```
# portupgrade -R firefox
```

要使用□□□的 package 而不是 ports 来□行安装， 需要指定 `-P`。 如果指定了□个□□， portupgrade 会搜索 `PKG_PATH` 中指定的本地目□， 如果没有□到， □从□程站点下□。 如果本地没有□到， 而且□程站点也没有成功地下□□□□包， □ portupgrade 将使用 ports。 要禁止使用 port， 可以指定 `-PP`。

```
# portupgrade -PP gnome2
```

如果只想下□ distfiles (或者， 如果指定了 `-P` 的□， 是 packages) 而不想□建或安装任何□西， 可以使用 `-F`。要了解更多□□， □参考 portupgrade(1)。

5.5.4.3. 使用 Portmanager 来升□ Ports

Portmanager 是□一个用以□化已安装 port 升□操作的工具。 它可以通□ ports-mgmt/portmanager port 安装：

```
# cd /usr/ports/ports-mgmt/portmanager
# make install clean
```

可以通□□个□□的命令来升□所有已安装的 port：

```
# portmanager -u
```

如果希望 Portmanager 在□行□□操作之前都□出提示， □使用 `-ui` 参数。 Portmanager 也可以用来在系□中安装新的 ports。 与通常的 `make install clean` 命令不同，它会在□□和安装□所□□的 port 之前升□所有依□包。

```
# portmanager x11/gnome2
```

如果□于所□ port 的依□有任何□□， 可以用 Portmanager 来以正□的□序重新□建它□。 完成之后， 有□□的 port 也将被重新□建。

```
# portmanager graphics/gimp -f
```

要了解更多信息， □参□ portmanager(1)。

5.5.4.4. 使用 **Portmaster** 升□ **Ports**

Portmaster 是□外一个用来升□已安装的 ports 的工具。 Portmaster 被□□成尽可能使用 "基本" 系□中能□到的工具 （它不依□于其他的 ports） 和 /var/db/pkg/ 中的信息来□□出需要升□的 ports。□可以在 ports-mgmt/portmaster □到它：

```
# cd /usr/ports/ports-mgmt/portmaster
# make install clean
```

Portmaster groups ports into four categories:

Portmaster 把 ports 分成4□：

- Root ports (不依□其他的 ports，也不被依□)
- Trunk ports (不依□其他的 ports，但是被其他的 ports 依□)
- Branch ports (依□于其他的 ports，同□也被依□)
- Leaf ports (依□于其他的 ports，但不被依□)

□可以使用 `-L` □□列出所有已安装的 ports 和□□存在更新的 ports：

```
# portmaster -L
===>>> Root ports (No dependencies, not depended on)
===>>> ispell-3.2.06_18
===>>> screen-4.0.3
        ===>>> New version available: screen-4.0.3_1
===>>> tcpflow-0.21_1
===>>> 7 root ports
...
===>>> Branch ports (Have dependencies, are depended on)
===>>> apache-2.2.3
        ===>>> New version available: apache-2.2.8
...
===>>> Leaf ports (Have dependencies, not depended on)
===>>> automake-1.9.6_2
===>>> bash-3.1.17
        ===>>> New version available: bash-3.2.33
...
===>>> 32 leaf ports

===>>> 137 total installed ports
        ===>>> 83 have new versions available
```

可以使用□个□□的命令升□所有已安装的 ports：

```
# portmaster -a
```

Portmaster 默□在□除一个□有的 port 前会做一个□□包。如果新的版本能□被成功安装，Portmaster 将□除□□。 使用 `-b` 后 Portmaster 便不会自□□除□□。加上 `-i` □□之后 Portmaster 将□入互□模式， 在升□□个 port 以前提示□□予□□。

如果□在升□的□程中□□了□□，□可以使用 `-f` □□升□/重新□□所有的 ports：

```
# portmaster -af
```

同□□也可以使用 Portmaster 往系□里安装新的 ports，升□所有的依□□系之后并安装新的 port：

```
# portmaster shells/bash
```

更多的□□信息□参□ portmaster(8)

5.5.5. Ports 和磁□空□

使用 Ports 套件会最□用完磁□空□。 在通□ ports □□和安装□件之后，□□□得清理□□的 work 目□，其方法是使用 `make clean` 命令。 □可以使用下面的命令来清理整个 Ports 套件：

```
# portsclean -C
```

随着□□的推移， □可能会在 distfiles 目□中□累下大量源代□文件。 □可以手工□除□些文件，也可以使用下面的命令来□除所有 port 都不引用的文件：

```
# portsclean -D
```

除此之外， 也可以用下列命令□去目前安装的 port 没有使用的源□包文件：

```
# portsclean -DD
```

□个 `portsclean` 工具是 portupgrade 套件的一部分。

不要忘□□除那些已□安装， 但已不再使用的 ports。 用于自□完成□□工作的一个好工具是 ports-mgmt/pkg_cutleaves port。

5.6. 安装之后□要做点什□？

通常，□通□port安装完一个□件后,可以□□它□的一些文□(如果它包含文□的□)， 或需要□□它的配置文件，来□保□个□件的□行， 或在机器□□的□候□□(如果它是一个服□的□)，等等。

□于不同的□件有着不同的配置□□。不管□□， 如果□装好了一个□件，但是不知道下一□□□□的□候，□些小技巧可能可以□助□:

- 使用 pkg_info(1) 命令，它能□到安装了□些文件，以及装在□里。 □个例子，如果□安装了 FooPackage version 1.0.0, 那□□个命令

  ```
  # pkg_info -L foopackage-1.0.0 | less
  ```

  将□示□个□件包安装的所有文件，□要特□注意在man/目□里面的文件， 它□可能是手册，etc/目□里面的配置文件，以及 doc/目□下面更多的文□。

  如果□不□定已□安装好的□件版本，□可以使用□□的命令

  ```
  # pkg_info | grep -i foopackage
  ```

  它将会□到所有已安装的□件包名字中包含*foopackage* 的□件包。 □于其他的□□， □只需要在命令行中替□*foopackage*。

- 一旦一些□件手册已被□□□安装，□可以使用 man(1) □看它。 同□的，如果有的□，□□可以完整的□看一遍配置文件的示例，以及任何□外的文□。
- 如果□用□件有网站，□□可以从网站上□到文□，常□□□的解答，或其他更多。 如果□不知道它□的网站地址，□使用下面的命令

```
# pkg_info foopackage-1.0.0
```

一个 WWW: 行, 如果它存在， 它将提供一个□个□用程序的网站URL.

- Ports 如果需要在服□器□□□□行(就像互□网服□器)， 它通常会把一个脚本的□例放入 /usr/local/etc/rc.d 目□。□了保□正□性， □可以□看□个脚本， 并□□或更改□个脚本的名字。 □情□看□□服□。

5.7. 如何□理坏掉的 Ports

如果□□□某个 port 无法正常工作， 有几件事□得□□， 包括：

1. 在 □□□告数据□ 中□□是否有尚未提交的修正。 如果有， 可以使用所提□的修正。
2. 要求 port 的□□人 (maintainer) 提供□助。 □入 make maintainer 或□□ Makefile □□□□人的□子□件地址。 □□得把 port 的名字和版本写在□件里 (Makefile 中的 $FreeBSD:□一行) 并把□□□出的□几行□□ maintainer。

某些 ports 并非一个人□□， 而是写了一个 □件列表。 □多， 但并非所有 port， 使用□似 freebsd-listname@FreeBSD.org □□的地址。 □在提出□□□考□□一点。

特□地， 由 ports@FreeBSD.org □□的 port， □□上并没有人□□。 □□□个□件列表的人□会感□□提供的修正和支持。 我□一直都需要更多志□者！

如果□没有得到回□， □可以使用 send-pr(1) 来提交□□□告 (□参□ 如何撰写 FreeBSD □□□告)。

3. 修正它！ Porter 手册 中提供了□于 "Ports" 基□□施的□□信息， 通□了解□些内容， □就能修正偶然坏掉的 port， 或甚至提交自己的 port 了！
4. 从□近的 FTP 站点下□一个□□好的安装包。 "中央的" package collection 在 ftp.FreeBSD.org 的 packages 目□中， 但 在此之前 □ 事先 □□一下是否存在□近的 □像网站！ 通常情况下□些安装包都可以直接使用， 而且□□比自行□□快一些。 安装□程本身可以通□ pkg_add(1) 来完成。

Chapter 6. X Window 系□

6.1. 概述

FreeBSD 使用 X11 来□用□提供功能□大的□形用□界面。 X11 是一□可以免□使用的 X □□系□， 其□□包括 Xorg FreeBSD 中默□使用并受官方支持的 X11 □□即是 Xorg， 它是由 X.Org 基金会□□的 X11 服□， 采用与 FreeBSD □似的授□。 此外， 也有一些用于 FreeBSD 的商□ X 服□器。

欲了解 X11 所支持的□示□等硬件， □□□ Xorg 网站。

在□□完□一章后，□将会了解：

- X □□系□的不同□件，它□是如何□同工作的。
- 如何安装和配置 X11。
- 如何安装和使用不同的□口管理器。
- 如何在 X11 中使用 TrueType® 字体。
- 如何□□的系□□置□形登□ (XDM)。

在□□□一章之前，□□□：

- 知道如何安装□外的第三方□用程序(安装□用程序: Packages 和 Ports)。

6.2. 理解 X

□于那些熟悉其他□形□境，比如 Microsoft® Windows® 或者 Mac OS® 的用□来□，第一次使用 X 可能会感□很□□。

通常□并不需要深入了解各□ X □件的作用以及它□之□的相互影□， 不□， 了解一些□于它□的基□知□，有助于更好地利用 X 的□大功能。

6.2.1. □什□要使用 X?

X 不是第一个□ UNIX® 而□□的□□系□， 但它是最流行的。 X 的原始□□□□在□□ X 之前就已□在□外一个□□系□上工作了。 那个系□的名字叫做 "W" (就是 "Window")。X 只是□□字母中 W 后面 的一个。

X 可以被叫做 "X", "X Window 系□", "X11", 等等。把 X11 称做 "X Windows" 可能会冒犯某些人； □看 X(7) 可以了解更多的信息。

6.2.2. X 客□机/服□器模型

X 一□始就是□□网□而□□的，所以 采用了 "client-server" 模型。在 X 模型中， "X server" □行在有□□，□示器，鼠□的□算机上。 服□器用来管理□示信息，□理来自□□和鼠□的□入信息， 并与其他□入□出□□交互(比如作□□入□□的 "tablet"， 或者作□□出□□的投影□)。 □一个 X □用程序 (比如 XTerm，或者 getenv(3)) 就是一个 "客□程序 (client)"。 客□程序□服□器□送信息，如 "□在□些坐□上画一个□口"， 而服□器□返回□理信息， 如 "用□□□点□了 OK 按□"。

如果□家或□公□境中只有一台使用 FreeBSD 的□算机， 就只能在同一台□算机上□行 X server 和 X client 了。

然而， 如果□有很多□行 FreeBSD 的机器， □可以在□的□面□算机上□行 X server， 而在比□高□的服□器上□行 X □用程序。 在□□的□境中， X server 和 X client 之□的通信就可以通□网□来□行。

□可能会□一些人感到困惑， 因□ X 的□□和他□料想的有些不同。 他□以□ "X server" 是□行在功能□大的大型机上的，而 "X client" 是□行在他□□面上的□算机上的。

□住，X server 是有□□和□示器的那台□算机，而 X client 是那些□示□口的程序。

Client 和 server 不一定都要□行在同一□操作系□上， 它□甚至无需在同一□□型的□算机上□行。 在 Microsoft® Windows® 或 Apple 公司的 Mac OS® 上□行 X server 也是可以的， 在它□上面也有很多免□的和商□化的□用程序。

6.2.3. □口管理器

X 的□□哲学很像 UNIX® 的□□哲学， "tools, not policy"。□就意味着 X 不会□□去□定任□□□如何去完成，而是，只□用□提供一些工具，至于决定如何使用□些工具是用□自己的 事情。

□套哲学□展了 X，它不会□定□口在屏幕上□□是什□□子，要如何移□鼠□， □□用什□□来切□□体 (比如， `Alt` + `Tab`按□，在 Microsoft® Windows® □境中的作用)，□个□口的工具条□□ 看起来像什□，他□是否□□有□□按□等等。

□□上，X 行使了一□叫做 "□口管理器"的□用程序的□□。有很多□□的程序可用： AfterStep, Blackbox, ctwm, Enlightenment, fvwm, Sawfish, twm, Window Maker，等等。□一个□口管理器 都提供了不同的界面和□感；其中一些□支持 "虚□□面"；有一些允□□可以定制一些□来管理□的□面； 一些有"□始" 按□，或者其他□似的□□；一些是 "可定制主□的(themeable)"， 通□安装新的主□， 可以完全改□外□。 □些以及很多其他的□口管理器， 都可以在 Ports Collection 的 x11-wm 分□目□里□到。

□外，KDE 和 GNOME □面□境都有他□自己的□口管理器 与□面集成。

□个□口管理器也有不同的配置机制；有些需要手工来写配置文件， 而□外一些□可以使用 GUI 工具来完成大部分的配置任□， □例而言， (Sawfish) 就使用 Lisp □言□写配置文件。

焦点策略

□口管理器的□一个特性是鼠□的 "focus policy"。 □个□口系□都需要有一个□□□口的方法来接受□□的□入信息，以及当前 □个□口□于可用状□。

□通常比□熟悉的是一个叫做 "click-to-focus" 的焦点策略。 □是 Microsoft® Windows® 使用的典型焦点策略，也就是□在一个□口上点□ 一下鼠□，□个□口就□于当前可用的状□。

X 不支持一些特殊的焦点策略。□切地□，□口管理器控制着在什□□候□个□口□有焦点。不同的□口管理器支持不同的焦点方案。它□都支持点□即□得焦点， 而且它□中的大多数都支持好几□方案。

最流行的焦点策略：

focus-follows-mouse

鼠□指示器下面的□口就是□得焦点的□口。 □个□口不一定位于其他所有□口之上。 通□将鼠□移到□一个□口就可以改□焦点， 而不需要在它上面点□。

sloppy-focus

□□方式是□ focus-follows-mouse 策略的一个小小□展。□于 focus-follows-mouse，如果□把鼠□移到了根□口（或□面背景）上， □所有的其它□口都会失去焦点， 而相□的全部□□□入也会□失。 如果□□了 sloppy-focus， □只有当指□□入新□口□，□口焦点才会□生□化， 而当退出当前□口□是不会□化的。

click-to-focus

当前□口由鼠□点□来□□。□口被"突出□示" ， 出□在所有其他□口的前面。即使指□被移向了□一个□口，所有的□□□入 仍会被□个□口接收。

□多□口管理器支持其他的策略，与□些相比又有些□化。□可以看具体 □口管理器的文□。

6.2.4. □口部件

提供工具而非策略的 X 方法使得在□个□用程序屏幕上看到的□口部件得到了 大大的□展。

"Widget" 只是□□用□接口中所有列□□目的一个□□，它 可以用某□方法来点□或操作；如按□，□□框，□□按□，□□，列表框等等。 Microsoft® Windows® 把□些叫做"控件"。

Microsoft® Windows® 和□果公司的 Mac OS® 都有一个□格的□口部件策略。 □用程序□□者被建□□保他□的□用程序共享一个普通的所□即所得的用□界面。 □于 X，它并不要求一个特殊的□形□格或一套相□合的□口部件集。

□□的□果是□不能期望 X □用程序只□有一个普通的所□即所得的界面。 有很多的流行的□口部件集□置，包括来自于 MIT 的 Athena， Motif® (模□ Microsoft® Windows® 的□口□格， 所有部件都具有斜□和3□灰色度)， OpenLook， 等等。

如今， □大多数比□新的 X □用程序采用一□新式的□口□□， □包括 KDE 所使用的 Qt， 以及 GNOME 所使用的 GTK+。 在□□一□□口系□下，UNIX® □面的一些所□即所得特性作了一些收□， 以使初学者感到更容易一些。

6.3. 安装 X11

Xorg 是 FreeBSD 上的默□ X11 □□。 Xorg 是由 X.Org 基金会□行的□放源代□ X Window 系□□□中的 X 服□。 Xorg 基于 XFree86™ 4.4RC2 和 X11R6.6 的代□。 从 FreeBSD Ports 套件可以安装 Xorg 的 7.7 版本。

如果需要从 Ports Collection □□和安装 Xorg：

```
# cd /usr/ports/x11/xorg
# make install clean
```

要完整地□□ Xorg □需要至少 4 GB 的剩余磁□空□。

□外 X11 也可以直接从 package 来安装。 我□提供了可以与 pkg_add(1) 工具配合使用的 X11 安装包。 如果从□程下□和安装， 在使用 pkg_add(1) □□不要指定版本号。 pkg_add(1) 会自□地下□最新版本的安装包。

想要从 package 安装 Xorg， □□地□入下面的命令：

```
# pkg_add -r xorg
```

上面的例子介□了如何安装完整的 X11 □件包， 包括服□器端，客□端，字体等等。 此外，也有一些□独的 X11 的 ports 和 packages.

□外， 如果需要最小化的 X11 □件， □也可以安装 x11/xorg-minimal。

□一章余下的部分将会□解如何配置 X11, 以及如何□置一个高效的□面□境。

6.4. 配置 X11

6.4.1. □始之前

在配置 X11 之前， □需要了解所安装的系□的下列信息：

- □示器□格
- □示□的芯片□型
- □示□的□存容量

□示器的□格被 X11 用来决定□示的分辨率和刷新率。 □些□格通常可以从□示器所□的文□中，以及制造商的网站□到。 需要知道□个数字□□： 垂直刷新率和水平刷新率。

□示□的芯片□型将决定 X11 使用什□模□来□□□形硬件。 尽管系□能自□□□出□大多数的硬件，但事先了解在自□□□出□的□候□是很有用□的。

□示□的□存大小决定了系□支持的分辨率和□色深度。 了解□些限制非常重要。

6.4.2. 配置 X11

□于 Xorg 7.3 □个版本， 可以不需要任何的配置文件就能□行，在提示符下□如下命令：

```
% startx
```

从 Xorg 7.4 □始， 可以使用 HAL 自□□□□□和鼠□。Ports sysutils/hal 和 devel/dbus 将被作□ x11/xorg 所依□的包安装□系□。 并且需要在 /etc/rc.conf 文件中□用：

```
hald_enable="YES"
dbus_enable="YES"
```

在更深入的配置 Xorg 以前， 需要□行□些服□ (手工□□或者重□机器)。

自□配置□于某些硬件可能不起作用或者无法做到期望的配置。 在□□情况下就有必要做一些手工配置。

□如 GNOME， KDE 或 Xfce 之□的□面□境， 大多都提供了一些允□用□非常易用的工具，来□置像分辨率□□的□示参数。 所以如果□□得默□的配置并不□合， 而且□打算安装一个□□的□面□境， 那□就□□□完成□面□境的安装， 并使用□合的□示□置工具。

配置 X11 需要一些□□。 第一□是以超□用□的身□建立初始的配置文件：

```
# Xorg -configure
```

□会在 /root 中生成一个叫做 xorg.conf.new 的配置文件 (无□□使用 su(1) 或直接登□， 都会改□默□的 $HOME 目□□量)。 X11 程序将□□探□系□中的□形硬件，并将探□到的硬件信息写入配置文件， 以便加□正□的□□程序。

下一□是□□□存的配置文件， 以□□ Xorg 能□同系□上的□形□□正常工作。 □于 Xorg 7.3 或者之前的版本， □入：

```
# Xorg -config xorg.conf.new
```

从 Xorg 7.4 和更高的版本□始， □个□□将□示出一个□色的屏幕，□于判断 X11 是否能正常工作会造成一些困□。 可以通□ `retro` □□使用旧的模式：

```
# Xorg -config xorg.conf.new -retro
```

如果看到□灰的格子以及 X 型鼠□指□， 就表示配置成功了。 要退出□□， 需要同□按下 `Ctrl` + `Alt` + `Fn` 来切□到用于□□ X 的虚□控制台 (`F1` 表示第一个虚□控制台) 之后按 `Ctrl` + `C`。

在Xorg 7.3 以及更早期的版本中， □使用 `Ctrl` + `Alt` + `Backspace` □合□来□制退出 Xorg。 如果需要在 7.4 和之后的版本中□用□个□合□， 可以在任意 X □端模□器中□入下面的命令：

```
% setxkbmap -option terminate:ctrl_alt_bksp
```

或者□ hald □建一个叫作 x11-input.fdi 的□□配置文件并保存至 /usr/local/etc/hal/fdi/policy 目□。 □个文件需包含以下□些：

```
<?xml version="1.0" encoding="utf-8"?>
<deviceinfo version="0.2">
  <device>
    <match key="info.capabilities" contains="input.keyboard">
      <merge key="input.x11_options.XkbOptions"
type="string">terminate:ctrl_alt_bksp</merge>
    </match>
  </device>
</deviceinfo>
```

□可能需要重□□的机器来使得 hald 重新□取□个文件。

此外， □需要在 xorg.conf.new 中的 ServerLayout 或 ServerFlags 小□中添加：

```
Option  "DontZap"   "off"
```

如果鼠□无法正常工作， 在□□深入之前需要先配置它。 参□ FreeBSD 安装一章中的 配置鼠□。 □外， 从 7.4 版本□始， xorg.conf 中的 InputDevice 部分将被忽略， □有助于自□□□硬件□□。 可以在□个文件中的 ServerLayout 或者 ServerFlags 加入以下□□使用旧的模式：

```
Option "AutoAddDevices" "false"
```

□入□□□同其他需要的□□ (比如， □□布局切□) 就可以像在之前的版本中的那□配置了。

正如前面所提到的， 自版本 7.4 □始 hald 守□□程默□自□□□□的□□。 可能□□出□的□□布局或型号有差□， 在□面□境中， 比如 GNOME， KDE 或者 Xfce 提供了工具来配置□□。 □一方面， 也可在 setxkbmap(1) 工具的□助下或者通□ hald 的配置文件来直接□置□□的属性。

□例来□， 如果某人想要使用一个 PC 102 □法□布局的□□， 我□就需要□ hald □建一个配置文件， 叫作 x11-input.fdi 并保存入 /usr/local/etc/hal/fdi/policy 目□。□个文件需要包含如下□些：

```
<?xml version="1.0" encoding="utf-8"?>
<deviceinfo version="0.2">
  <device>
    <match key="info.capabilities" contains="input.keyboard">
      <merge key="input.x11_options.XkbModel"
type="string">pc102</merge>
      <merge key="input.x11_options.XkbLayout" type="string">fr</merge>
    </match>
  </device>
</deviceinfo>
```

如果□个文件已□存在， 只要把□□配置相□的部分拷□加入即可。

□需要重□□的机器使 hald □入此文件。

也可以在 X 模□□端或一个脚本中使用以下的命令□到相同的效果:

```
% setxkbmap -model pc102 -layout fr
```

/usr/local/shared/X11/xkb/rules/base.lst 列出了各□不同的□□， 布局和可用的□□。

接下来是□整 xorg.conf.new 配置文件并作□□。 用文本□□器如 emacs(1) 或 ee(1) 打□□个文件。要做的第一件事是□当前系□的□示器□置刷新率。 □些□包括垂直和水平的同□□率。 把它□加到 xorg.conf.new 的 "Monitor" 小□中：

```
Section "Monitor"
        Identifier   "Monitor0"
        VendorName   "Monitor Vendor"
        ModelName    "Monitor Model"
        HorizSync    30-107
        VertRefresh  48-120
EndSection
```

在配置文件中也有可能没有 HorizSync 和 VertRefresh。 如果是□□的□， 就只能手□添加， 并在 HorizSync 和 VertRefresh 后面□置合□的数□了。 在上面的例子中， □出了相□的□示器的参数。

X 能□使用□示器所支持的 DPMS (能源之星) 功能。 xset(1) 程序可以控制超□□□， 并□制待机、挂起或□机。如果希望□用□示器的 DPMS 功能， □需要把下面的□置添加到 monitor □中：

```
        Option       "DPMS"
```

□□ xorg.conf.new 之前□□□□□默□的分辨率和色深。 □是在 "Screen" 小□中定□的：

```
Section "Screen"
        Identifier "Screen0"
        Device     "Card0"
        Monitor    "Monitor0"
        DefaultDepth 24
        SubSection "Display"
                Viewport  0 0
                Depth     24
                Modes     "1024x768"
        EndSubSection
EndSection
```

DefaultDepth □□字描述了要□行的默□色深。 □可以通□ Xorg(1) 的 -depth 命令行□□来替代配置文件中的□置。 Modes □□字描述了□定□色深度下屏幕的分辨率。 需要□明的是， 目□系□的□形硬件只支持由 VESA 定□的□准模式。 前面的例子中， 默□色深是使用 24位色。 在采用□个色深□， 允□的分辨率是 1024x768。

最后就是将配置文件存□， 并使用前面介□的□□模式□□一下。

在□□并解决□□的□程中， 包含了与 X11 服□器相□的各个□□的信息的 X11 日志文件会□□□□和排除□□有所□助。 Xorg 日志的文件名是 /var/log/Xorg.0.log □□的格式。 □□的日志文件名可能是 Xorg.0.log 到 Xorg.8.log 等等。

如果一切准□妥当， 就可以把配置文件放到公共的目□中了。 □可以在 Xorg(1) 里面□到具体位置。 □个位置通常是 /etc/X11/xorg.conf 或 /usr/local/etc/X11/xorg.conf。

```
# cp xorg.conf.new /etc/X11/xorg.conf
```

□在已□完成了 X11 的配置全□程。 Xorg 可以通□ startx(1) 工具来□□。 除此之外， X11 服□器也可以用 xdm(1) 来□□。

6.4.3. 高□配置主□

6.4.3.1. 配置 Intel® i810 □示芯片□

配置Intel i810芯片□的□示□需要有□□ X11 的能□用来□□□示□的 agpgart AGP程序接口。 □参□ agp(4) □□程序的□机手册了解更多□□。

□也□用于其他的□形□硬件配置。 注意如果系□没有将 agp(4) □□程序□□□内核，□□用 kldload(8) 加□模□是无效的。 □个□□程序必□□□□内核或者使用 /boot/loader.conf 在□□□加□□入内核。

6.4.3.2. 添加□屏平板□示器

□一□假定□了解一些□于高□配置的知□。 如果使用前面的□准配置工具不能□生可用的配置，□在日志文件中提供的信息□□足以修正配置使其正□工作。 如果需要的□， □□使用一个文本□□器来完成□□工作。

目前的□屏 (WSXGA、 WSXGA+、 WUXGA、 WXGA、 WXGA+， 等等) 支持 16:10 和 10:9 或一些支持不大好的□示比例。 常□的一些 16:10 比例的分辨率包括：

- 2560x1600
- 1920x1200
- 1680x1050
- 1440x900
- 1280x800

有□， 也可以□□地把□些分辨率作□ `Section "Screen"` 中的 `Mode` 来□行配置， □似下面□□：

```
Section "Screen"
Identifier "Screen0"
Device     "Card0"
Monitor    "Monitor0"
DefaultDepth 24
SubSection "Display"
    Viewport  0 0
    Depth     24
    Modes     "1680x1050"
EndSubSection
EndSection
```

Xorg 能□自□地通□ I2C/DDC 信息来自□□取□屏□示器的分辨率信息， 并□理□示器支持的□率和分辨率。

如果□□程序没有□□的 `ModeLines`， 就需要□ Xorg 一些提示了。 使用 /var/log/Xorg.0.log 能□提取足□的信息， 就可以写一个可用的 `ModeLine` 了。 □□信息如下所示：

```
(II) MGA(0): Supported additional Video Mode:
(II) MGA(0): clock: 146.2 MHz   Image Size:  433 x 271 mm
(II) MGA(0): h_active: 1680  h_sync: 1784  h_sync_end 1960 h_blank_end 2240 h_border:
0
(II) MGA(0): v_active: 1050  v_sync: 1053  v_sync_end 1059 v_blanking: 1089 v_border:
0
(II) MGA(0): Ranges: V min: 48  V max: 85 Hz, H min: 30  H max: 94 kHz, PixClock max
170 MHz
```

□些信息称做 EDID 信息。 从中建立 `ModeLine` 只是把□些数据重新排列□序而已：

```
ModeLine <name> <clock> <4 horiz. timings> <4 vert. timings>
```

如此， 本例中的 Section "Monitor" 中的 ModeLine □□似下面的形式：

```
Section "Monitor"
Identifier      "Monitor1"
VendorName      "Bigname"
ModelName       "BestModel"
ModeLine        "1680x1050" 146.2 1680 1784 1960 2240 1050 1053 1059 1089
Option          "DPMS"
EndSection
```

□□□□的□□□□之后， X 就可以在□的□屏□示器上□□了。

6.5. 在 X11 中使用字体

6.5.1. Type1 字体

X11 使用的默□字体不是很理想。 大型的字体□得参差不□，看起来很不□□， 并且， 在 getenv(3) 中，小字体□直无法看清。 有好几□免□、 高□量的字体可以很方便地用在 X11 中。 例如，URW字体集合 (x11-fonts/urwfonts) 就包括了高□量的 □准 type1 字体 (Times Roman™, Helvetica™、 Palatino™ 和其他一些)。 在 Freefont 集合中 (x11-fonts/freefonts) 也包括更多的字体， 但它□中的□大部分使用在□形□件中，如 Gimp，在屏幕字体中使用并不完美。□外， 只要花很少的功夫，可以将 XFree86™ 配置成能使用 TrueType® 字体：□参□后面的 TrueType® 字体一□。

如果希望使用 Ports Collection 来安装上面的 Type1 字体， 只需□行下面的命令：

```
# cd /usr/ports/x11-fonts/urwfonts
# make install clean
```

freefont 或其他的字□和上面所□的大体□似。 □了□ X 服□器能□□□到□些字体， 需要在 X 服□器的配置文件 (/etc/X11/xorg.conf) 中□加下面的配置：

```
FontPath "/usr/local/lib/X11/fonts/URW/"
```

或者，也可以在命令行□行：

```
% xset fp+ /usr/local/lib/X11/fonts/URW
% xset fp rehash
```

□□会起作用，但是当 X 会□□束后就会□失， 除非它被添加到□□文件 (~/.xinitrc 中， □□一个□常的 startx 会□，或者当□通□一个□似 XDM 的□形登□管理器登□□添加到 ~/.xsession 中)。 第三□方法是使用新的 /usr/local/etc/fonts/local.conf 文件： □看 anti-aliasing 章□。

6.5.2. TrueType® 字体

Xorg 已□内建了□ TrueType® 字体的支持。有□个不同的模□能□□用□个功能。 在□个例子中使用 freetype □个模□，因□它与其他的字体描□后端 是兼容的。要□用 freetype 模□，只需要将下面□行添加到 /etc/X11/xorg.conf 文件的 "Module" 部分。

```
Load "freetype"
```

□在，□ TrueType® 字体□建一个目□ (比如， /usr/local/lib/X11/fonts/TrueType) 然后把所有的 TrueType® 字体□制到□个目□。□住□不能直接从 Macintosh® □算机中提取 TrueType® 字体； 能被 X11 使用的必□是 UNIX®/MS-DOS®/Windows® 格式的。 一旦□已□将□些文件□制到了□个目□， 就可以用 ttmkfdir 来□建 fonts.dir 文件， 以便□X字体引□知道□已□安装了□些新文件。 ttmkfdir 可以在 FreeBSD Ports 套件中的 x11-fonts/ttmkfdir 中□到。

```
# cd /usr/local/lib/X11/fonts/TrueType
# ttmkfdir -o fonts.dir
```

□在把 TrueType® 字体目□添加到字体路径中。 □和上面 Type1 字体的□□是一□的， 那就是，使用

```
% xset fp+ /usr/local/lib/X11/fonts/TrueType
% xset fp rehash
```

或者把 FontPath □行加到 xorg.conf 文件中。

就是□□。□在 getenv(3)，Gimp，StarOffice™ 和其他所有的 X □用程序 □□可以□出安装的 TrueType® 字体。一些很小的字体(如在 Web □面上高分辨率□示的文本) 和一些很大的字体(在 StarOffice™ 下) □在看起来已□很好了。

6.5.3. Anti-Aliased 字体

□于所有支持 Xft 的□用程序， 所有放到 X11 /usr/local/lib/X11/fonts/ 和 ~/.fonts/ 中的字体都自□地被加入反走□支持。 □大多数□新的程序都提供了 Xft 支持， 包括 KDE、 GNOME 以及 Firefox。

要控制□些字体是 anti-aliased，或者配置 anti-aliased 特性， □建(或者□□，如果文件已□存在的□)文件 /usr/local/etc/fonts/local.conf。Xft 字体系□的几个 高□特性都可以使用□个文件来□□； □一部分只描述几□ 最□□的情况。要了解更多的□□，□□看 fonts-conf(5).

□个文件一定是 XML 格式的。注意□保所有的□□都完全的□□掉。 □个文件以一个很普通的 XML □□始，后跟一个 DOCTYPE 定□， 接下来是 <fontconfig> □□：

```
      <?xml version="1.0"?>
      <!DOCTYPE fontconfig SYSTEM "fonts.dtd">
      <fontconfig>
```

像前面所做的那□，在 /usr/local/lib/X11/fonts/ 和 ~/.fonts/ 目□下的所有字体已□可以被支持 Xft 的

□用程序使用了。如果□想添加□□个目□以外的其他路径，□□的添加下面□行到/usr/local/etc/fonts/local.conf文件中：

```
<dir>/path/to/my/fonts</dir>
```

添加了新的字体，尤其是添加了新的字体目□后，□□□□行下面的命令重建字体□存：

```
# fc-cache -f
```

Anti-aliasing 会□字体□□有些模糊，□□□加了非常小的文本的可□性， 并从大文本字体中□除 "□□"。但如果使用普通的文本，□可能引起眼疲□。 要禁止 14磅 以下字体的反走□， 需要□加如下配置：

```
        <match target="font">
            <test name="size" compare="less">
                <double>14</double>
            </test>
            <edit name="antialias" mode="assign">
                <bool>false</bool>
            </edit>
        </match>
        <match target="font">
            <test name="pixelsize" compare="less" qual="any">
                <double>14</double>
            </test>
            <edit mode="assign" name="antialias">
                <bool>false</bool>
            </edit>
        </match>
```

用 anti-aliasing 来□隔一些等□字体也是不□当的。 □似乎是 KDE 的一个□□。 要修□□个□□需要□保□个字体之□的□距保持在100。 加入下面□些行：

```
        <match target="pattern" name="family">
            <test qual="any" name="family">
                <string>fixed</string>
            </test>
            <edit name="family" mode="assign">
                <string>mono</string>
            </edit>
        </match>
        <match target="pattern" name="family">
            <test qual="any" name="family">
                <string>console</string>
            </test>
            <edit name="family" mode="assign">
                <string>mono</string>
            </edit>
        </match>
```

(□里把其他普通的修□的字体作□ "mono")，然后加入：

```
        <match target="pattern" name="family">
            <test qual="any" name="family">
                <string>mono</string>
            </test>
            <edit name="spacing" mode="assign">
                <int>100</int>
            </edit>
        </match>
```

某些字体，比如 Helvetica，当 anti-aliased 的□候可能存在□□。 通常的表□□字体本身似乎被垂直的切成□半。 糟□的□候，□可能□致□用程序崩□。 □了避免□□的□象，考□添加下面几行到 local.conf文件里面：

```
        <match target="pattern" name="family">
            <test qual="any" name="family">
                <string>Helvetica</string>
            </test>
            <edit name="family" mode="assign">
                <string>sans-serif</string>
            </edit>
        </match>
```

一旦□完成□ local.conf 文件的□□，□保□使用了 </fontconfig> □□来□束文件。 不□□做将会□致□的更改被忽略。

最后，用□可以通□他□个人的 .fonts.conf 文件来添加自己的□定。 要完成此□工作， 用□只需□□地□建 ~/.fonts.conf 并添加相□配置。 此文件也必□是 XML 格式的。

最后：□于LCD屏幕， 可能希望使用子像素的取□。 □□而言， □是通□分□控制 (水平方向分□的) □、□、□

像素， 来改善水平分辨率； □□做的效果一般会非常明□。 要□用它， 只需在 local.conf 文件的某个地方加入：

```
        <match target="font">
            <test qual="all" name="rgba">
                <const>unknown</const>
            </test>
            <edit name="rgba" mode="assign">
                <const>rgb</const>
            </edit>
        </match>
```

随□□示器的□□不同， 可能需要把 `rgb` 改□ `bgr`、 `vrgb` 或 `vbgr`： □□一下看看那个更好。

6.6. X □示管理器

6.6.1. 概要

X □示管理器(XDM) 是一个X□□系□用于□行登□会□管理的可□□。 □个可以□用于多□情况下，包括小 "X Terminals"， □面，大网□□示服□器。既然 X □□系□不受网□和□□的限制， 那□于通□网□□接起来的□行 X 客□端和服□器端的不同机器， 就会有很多的可配置□。 XDM 提供了一个□□要□接到□个□示服□器的□形接口，只要□入如登□用□名和密□□□的□□信息。

□也可以把 XDM 想象成与 getty(8) 工具一□(see 配置 for details)。□用□提供了同□功能。它可以完成系□的登□任□， 然后□用□□行一个会□管理器 (通常是一个 X □□管理器)。接下来 XDM 就等待□个程序退出，□出信号用□已□登□完成，□当退出屏幕。 □□， XDM 就可以□下一个登□用□□示登□和可□□屏幕。

6.6.2. 使用 XDM

如果希望使用 XDM 来□□， 首先需要安装 x11/xdm port (在□新版本的 Xorg 中它并不是默□安装的)。 XDM 服□程序位于 /usr/local/bin/xdm。 任何□候都可以 `root` 用□的身□来□行它， 以令其管理本地系□的 X □示。如果希望□ XDM 在系□□次□□□程中自□□行， 比□方便的做法是把它写到 /etc/ttys 的配置中。 有□□个文件的具体格式和使用方法□参□ 添加一个□□到/etc/ttys。 在默□的 /etc/ttys 文件中已□包含了在虚□□端上□行 XDM 服□的示□配置：

```
ttyv8   "/usr/local/bin/xdm -nodaemon"  xterm   off secure
```

默□情况下，□个□□是□□的，要□用它， □需要把第5部分的 `off` 改□ `on` 然后按照 重新□取/etc/ttys来□制init 的指□ 重新□□ init(8)。第一部分，□个程序将管理的□端名称是 `ttyv8`。□意味着 XDM 将□行在第9个虚□□端上。

6.6.3. 配置 XDM

XDM 的配置目□是在 /usr/local/lib/X11/xdm中。在□个目□中， □会看到几个用来改□ XDM 行□和外□的文件。□会□到□些文件：

文件	描述
Xaccess	客□端授□□□。
Xresources	默□的X□源□。
Xservers	□程和本地□示管理列表。
Xsession	用于登□的默□的会□脚本。
Xsetup_*	登□之前用于加□□用程序的脚本。
xdm-config	□行在□台机器上的所有□示的全局配置。
xdm-errors	服□器程序□生的□□。
xdm-pid	当前□行的 XDM 的□程 ID。

当 XDM □行□， 在□个目□中有几个脚本和程序可以用来□置□面。 □些文件中的□一个的用法都将被□要地描述。 □些文件的更□□的□法和用法在 xdm(1) 中将有□□描述。

默□的配置是一个矩形的登□□口，上面有机器的名称， "Login:" 和 "Password:"。如果□想□□□自己个性化的 XDM 屏幕，□是一个很好的起点。

6.6.3.1. Xaccess

用以□接由 XDM 所控制的□示□□的□□， 叫做 X □示管理器□接□□ (XDMCP)。 □个文件是一□用以控制来自□程□算机的 XDMCP □接的□□。 除非□修改 xdm-config 使其接受□程□接， 否□其内容将被忽略。 默□情况下，它不允□来自任何客□端的□接。

6.6.3.2. Xresources

□是一个默□的用来□示□□和登□屏幕的□用程序文件。 □可以在□个文件中□登□程序的外□□行定制。 其格式与 X11 文□中描述的默□□用程序文件是一□的。

6.6.3.3. Xservers

□是一个□□者□当提供的作□可□的□程□示列表。

6.6.3.4. Xsession

□是一个用□登□后□□ XDM 的默□会□脚本。通常，在 ~/.xsession 中□个用□将有一个可定制的会□脚本。

6.6.3.5. Xsetup_*

在□示□□者或登□接口之前，□些将被自□□行。 □是一个□个□示都要用到的脚本，叫做 Xsetup_，后面会跟一个本地□示的数字(比如 Xsetup_0)。典型的，□些脚本将在后台 (如 `xconsole`)□行一个或□个程序。

6.6.3.6. xdm-config

此文件以□用程序默□□的形式， 提供了在安装□所使用的普□的□示□置。

6.6.3.7. xdm-errors

□个文件包含了 XDM 正□法□行的的 X 服□器 的□出。 如果 XDM 正□法□行的□示由于某□原因被挂起， 那

□是一个□□□□信息的好地方。 □些信息会在□一个会□的基□上被写到用□的 ~/.xsession-errors 文件中。

6.6.4. □行一个网□□示服□器

□于其他客□端来□， 如果希望它□能□接到□示服□器，□就必□□□□□控制□□， 并□用□接□听。 默□情况下， □些都□□□比□保守的□。 要□ XDM 能□听□接， 首先要在 xdm-config 文件中注□掉一行：

```
! SECURITY: do not listen for XDMCP or Chooser requests ! Comment out this line if you
want to manage X terminals with xdm
DisplayManager.requestPort:     0
```

然后重新□□XDM。 □住默□□用程序文件的注□以"!" 字母□始，不是"\#"。 □需要□置□格的□□控制 - 看看在 Xaccess 文件中的□例， 并参考 xdm(1) 的□机手册， 以了解□一□的□□。

6.6.5. 替□ XDM

有几个替□默□ XDM程序的方案。 其中之一是 上一□已□描述□的kdm (与 KDE□□在一起)。 kdm 提供了□多□□上的改□和局部的修□， 同□能□用□在□□□能□□他□喜□的□口管理器。

6.7. □面□境

□□描述了 FreeBSD 上用于 X 的不同□面□境。 "□面□境" 可能□□是一个□□的□口管理器， 也可能是一个像 KDE 或者 GNOME□□的完整□面□用程序套件。

6.7.1. GNOME

6.7.1.1. 有□ GNOME

GNOME 是一个用□界面友好的□面□境， 能□使用□很容易地使用和配置他□的□算机。 GNOME 包括一个面板(用来□□□用程序和□示状□)， 一个□面(存放数据和□用程序的地方)， 一套□准的□面工具和□用程序，和一套与其他人相互□同工作的□□集。 其他操作系□的用□在使用 GNOME提供的□大的□形□□□境□会□得很好。更多的□于 FreeBSD 上 GNOME 的信息 可以在 FreeBSD GNOME Project 的网站上□到。 此外，□个网站也提供了相当□尽的□于安装、 配置和管理 GNOME 的常□□□解答 (FAQ)。

6.7.1.2. 安装 GNOME

□个□件可以很容易地通□□□□包或 Ports 套件来安装：

要从网□安装GNOME， 只要□入：

```
# pkg_add -r gnome2
```

从源代□□□GNOME，可以使用 ports□：

```
# cd /usr/ports/x11/gnome2
# make install clean
```

GNOME 需要挂□ /proc 文件系□才能正常□作。添加如下

```
proc            /proc       procfs  rw  0   0
```

到 /etc/fstab 以便在系□□□□自□挂□ procfs(5)。

一旦装好了 GNOME， 就必□告□ X server □□ GNOME 而不是默□的□口管理器。

最□□的□□ GNOME 的方法是使用 GDM， GNOME □示管理器。 随 GNOME □面一同安装的 GDM 尽管默□是禁用的。 可以在 /etc/rc.conf 中加入以下□行□用：

```
gdm_enable="YES"
```

□□在□重□机器的□候， GDM 将自□□行。

通常我□希望在 GDM □□□， 同□□用所有的 GNOME 服□， 可以将如下□行加入 /etc/rc.conf：

```
gnome_enable="YES"
```

GNOME 也可以通□□当地配置名□ .xinitrc 的文件来□□。 如果已□有了自定□的 .xinitrc， 将□□当前□口管理器的那一行改□□□ /usr/local/bin/gnome-session 就可以了。 如果□没有， 那□只需□□地：

```
% echo "/usr/local/bin/gnome-session" > ~/.xinitrc
```

接下来□入 startx， GNOME □面□境就□□了。

如果之前使用了一些旧式的□示管理器， 例如 XDM， □□□做是没用的。 此□□建立一个可□行的 .xsession 文件， 其中包含同□的命令。 要完成□□工作， 需要用 /usr/local/bin/gnome-session 取代□有的□口管理器：

```
% echo "#!/bin/sh" > ~/.xsession
% echo "/usr/local/bin/gnome-session" >> ~/.xsession
% chmod +x ~/.xsession
```

□有一□做法， 是配置□示管理器， 以便在登□□提示□□□□口管理器； 在 KDE □□ □□中介□了□于如何□ kdm（KDE 的□示管理器）□行□□的配置。

6.7.2. KDE

6.7.2.1. 有□ KDE

KDE 是一个容易使用的□代□面□境。 KDE 有很多很好的特性：

- 一个美□的□代的□面。

- 一个集合了完美网□□境的□面。
- 一个集成的□助系□，能□方便、高效地□助□使用 KDE □面和它的□用程序。
- 所有的KDE□用程序具有一致的所□即所得界面。
- □准的菜□和工具□，□□布局，□色配置等。
- 国□化：KDE 可以使用超□40□□言。
- 集中化、 □一的□□框□□的□面配置
- □多有用的 KDE□用程序。

KDE 附□了一个名□ Konqueror 的 web □□器， 它是其他□行于 UNIX® 系□上的 web □□器的一个□大的□争□手。 要了解□于 KDE 的更多□情， 可以□□ KDE 网站。 与 FreeBSD 相□的 KDE 信息和□源， 可以在 FreeBSD 上的 KDE □□ 的网站□到。

FreeBSD 上提供了□□版本的 KDE。 版本 3 已□推出了很□□□， 十分成熟。 而版本 4， 也就是下一代版本，也可以通□ Ports 套件来安装。 □□□版本甚至能□并存。

6.7.2.2. 安装 KDE

与 GNOME 和其他□面□境□似， □个□件可以很容易地通□□□□包或 Ports 套件来安装：

要从网□安装 KDE3 只需要：

```
# pkg_add -r kde
```

要从网□安装 KDE4 □需要：

```
# pkg_add -r kde4
```

pkg_add(1) 就会自□的下□最新版本的□用程序。

要从源代□□□ KDE3， 可以使用 ports □：

```
# cd /usr/ports/x11/kde3
# make install clean
```

而从 ports 提供的源代□□□ KDE4， □□的操作□是：

```
# cd /usr/ports/x11/kde4
# make install clean
```

安装好 KDE 之后， □需要告□X server □□□个□用程序来代替默□的□口管理器。 □可以通□□□ .xinitrc 文件来完成：

□于 KDE3：

```
% echo "exec startkde" > ~/.xinitrc
```

□于 KDE4：

```
% echo "exec /usr/local/kde4/bin/startkde" > ~/.xinitrc
```

□在，无□□什□□候用 startx□入 X □□系□， KDE 就将成□□的□面□境。

如果使用一个像 XDM□□的□示管理器， 那配置文件可能有点不同。需要□□一个 .xsession 文件，有□ kdm 的用法会在□章的后面介□。

6.7.3. 有□ KDE 的更多□□

□在 KDE 已□被安装在系□中了。 通□□助□面或点□多个菜□可以□□很多□西。 Windows® 或 Mac® 用□会有回到家的感□。

有□ KDE 的最好的参考□料是 它的在□文□。KDE □有它自己的 web □□器 Konqueror， □有很多其他的□用程序和□富文□。 □□的余下部分将□□一些很□用走□□花的方法来学□的技□□目。

6.7.3.1. KDE □示管理器

如果在同一系□上有多个用□， □管理□通常会希望使用□形化的登□界面。 前面已□提到， 使用 XDM 可以完成□□工作。 不□， KDE 本身也提供了□一个□□， 即 kdm， 它的外□更富吸引力，而且提供了更多的登□□□。 □得一提的是， 用□□能通□菜□很容易地□□希望使用的□面□境 (KDE、 GNOME 或其它)。

要□用 kdm， 需要根据 KDE 的版本修改不同的配置文件。

□于 KDE3， /etc/ttys 中的 ttyv8 □需被改写成如下的形式：

```
ttyv8 "/usr/local/bin/kdm -nodaemon" xterm on secure
```

□于 KDE4， □需要将如下□行加入 /etc/rc.conf：

```
local_startup="${local_startup} /usr/local/kde4/etc/rc.d"
kdm4_enable="YES"
```

6.7.4. Xfce

6.7.4.1. 有□Xfce

Xfce是以被GNOME 使用的 GTK+ 工具包□基□的□面□境， 但是更加□巧，□合于那些需要一个易于使用和配置并且□□而高效的□面的人。 看起来，它非常像使用在商□UNIX®系□上的CDE□境。 Xfce的主要特性有下面□些：

- 一个□□，易于使用的□面。

- 完全通□鼠□的□□和按□来控制等。
- 与CDE 相似的主面板，菜□，applets和□用launchers。
- 集成的□口管理器，文件管理器，声音管理器， GNOME □用模□等等。
- 可配置界面的主□。(因□它使用GTK+)
- 快速，□便，高效：□于比□老的/旧的机器或□有很少内存的机器仍然很理想。

更多有□Xfce 的信息可以参考Xfce 网站。

6.7.4.2. 安装Xfce

有一个二□制的Xfce □件包存在(在写作的□候)。要安装的□，□行下面的命令：

```
# pkg_add -r xfce4
```

□外， 也可以使用 Ports Collection 从源代□□□：

```
# cd /usr/ports/x11-wm/xfce4
# make install clean
```

□在，要告□X服□器在下次X□□□□行 Xfce。 只要□行下面的命令：

```
% echo "/usr/local/bin/startxfce4" > ~/.xinitrc
```

接下来就是□□ X， Xfce将成□□的□面。 与以前一□，如果使用像 XDM □□的□示管理器，需要□建一个.xsession文件，就像有□ GNOME 的那□描述的， 使用/usr/local/bin/startxfce4 命令，或者，配置□示管理器允□在□□□□一个□面， 就像有□kdm的那□描述的。

部分 II: 常□的任□

前面已□介□了必要的基□知□， 手册的□一部分将□□ FreeBSD 的一些最常用的功能。 □些章□包括：

- 向□介□流行和□用的□面□用程序： □□器、□品工具、文□察看程序，等等。
- 向□介□一系列可以在 FreeBSD 上使用的多媒体工具。
- 介□□□定制的 FreeBSD 内核以□用附加功能的方法。
- □□介□包括□面和网□打印机在内的打印系□□置。
- 向□展示如何在 FreeBSD 上□行 Linux □用程序。

某些章□希望□首先□□□其他部分， 在□些章的□□部分也会□出□似的提示。

Chapter 7. □面□用

7.1. 概述

FreeBSD 可以□行□□繁多的□面□用程序， □包括像□□器和字□理□□的□件。 □大多数□□的程序都可以通□ package 来安装， 或者从 Ports Collection 自□地□建。 □多新用□希望能□在它□的系□中□到□□的□用程序。 □一章将向□展示如何□松地使用 package 或者 Ports Collection 中安装□□的□件。

需要注意的是从 ports 安装意味着要□□源□。 根据□□的 ports 和□□速度的不同， □可能需要花□相当□的□□。 若是□□得□□源□太□耗□的□， □大多数 ports 也有□□□的版本可供安装。

因□ FreeBSD 提供的二□制兼容 Linux 的特性， □多原本□ Linux □□的程序都可以直接用在□的□面。 在安装任何的 Linux □用程序之前， □烈的推□□□□ Linux® 二□制兼容模式。 当□在□□特定的 ports □， 可以使用 whereis(1)。 一般来□， □多利用 Linux 二□制兼容特性的 ports 都以"linux-"□□。 在下面的介□中，都假□安装 Linux □用程序前已□□□了 Linux 二□制兼容功能。

本章涵□以下□□□用程序：

- □□器 (例如 Firefox、 Opera、 Konqueror)
- □公、□象□理 (例如 KOffice、 AbiWord、 GIMP、 OpenOffice.org、 LibreOffice)
- 文□□看 (例如 Acrobat Reader®、 gv、 Xpdf、 GQview)
- □□ (例如 GnuCash、 Gnumeric、 Abacus)

□□□章之前，□□□：

- 知道如何安装□外的第三方□件(安装□用程序. Packages 和 Ports)。
- 知道如何安装 Linux □件(Linux® 二□制兼容模式)。

想要□得更多的有□多媒体□境的信息，□□□ 多媒体。如果□想要建立和使用□子□件， □参考□子□件。

7.2. □□器

FreeBSD并没有□先安装特定的□□器。然而，在 ports 的目□ www 有□多□□器可以安装。如果□没有□□一一□□它□ (有些□候□可能需要花□相当□的□□) 大部分都有 package 可用。

KDE 和 GNOME 已□提供 HTML □□器。 □参考□面□境得到更多完整的有□□定□些□面□境的信息。

如果□要□小型的□□器， 可以□□看 www/dillo2、 www/links 或 www/w3m。

□一□□及如下程序：

程序名称	□源需求	安装□□	主要依□
Firefox	中等	□	Gtk+

程序名称	□源需求	安装□□	主要依□
Opera	少	□松	同□有可用的 FreeBSD 和 Linux 版本。 Linux 版本需要使用 Linux 二□制兼容模□和 linux-openmotif。
Firefox	中等	□	Gtk+
Konqueror	中等	□	需要 KDE □

7.2.1. Firefox

Firefox 是一个□代， 自由， □放源代□□定的□□器， 并完全移植到了 FreeBSD 上：它的特性包括有一个非常□准的 HTML □示引□， □□式□□， □出□口阻止， □展□件， 改□的安全性， 等等。Firefox 是基于 Mozilla 的代□。

□可以通□□入下面的命令来安装□□□的包：

```
# pkg_add -r firefox
```

□将会安装 Firefox 7.0， 如果希望□行 Firefox 3.6， □□使用下面的命令：

```
# pkg_add -r firefox36
```

如果□希望从源代□□□的□， 可以通□ Ports Collection 安装：

```
# cd /usr/ports/www/firefox
# make install clean
```

□于 Firefox 3.6， □□的命令中的 `firefox` □改□ `firefox36`。

7.2.2. Firefox 与 Java™ □件

在□一□和接下来的□□中， 我□均假定□已□安装了 Firefox。

通□ Ports 套件来安装 OpenJDK 6， □入下面的命令：

```
# cd /usr/ports/java/openjdk6
# make install clean
```

接下来安装 java/icedtea-web port：

```
# cd /usr/ports/java/icedtea-web
# make install clean
```

□□□在□□上述 port □使用的是系□□□的配置。

□□□□器并在地址□中□入 about:plugins 然后按 Enter。 □□器将会呈□一个列出所有已安装□件的□面； Java™ □件□在其中出□。

如果□□器□不到□件， □用□可能必□□行下面的命令， 并重□□□器：

```
% ln -s /usr/local/lib/IcedTeaPlugin.so \
  $HOME/.mozilla/plugins/
```

7.2.3. Firefox 与 Adobe® Flash™ □件

Adobe® Flash™ □件并没有直接提供其 FreeBSD 版本。 不□， 我□有一个□件□ (wrapper) 可以用来□行 Linux 版本的□件。 □个 wrapper 也支持 Adobe® Acrobat®、 RealPlayer 和很多其他□件。

根据□ FreeBSD 版本的不同□□相□的安装□□：

1. FreeBSD 7.X

   安装 www/nspluginwrapper port。 □个 port 需要安装一个□大的emulators/linux_base-fc4 port。

   下一□是安装 www/linux-flashplugin9 port。 □将会安装 Flash™ 9.X， 此版本目前能在 FreeBSD 7.X 上正常□行。

   在比 FreeBSD 7.1-RELEASE 更旧版本的系□上， □必□安装 www/linux-flashplugin7 并跳□以下 linprocfs(5) 的部□。

2. FreeBSD 8.X

   安装 www/nspluginwrapper port。 □个 port 需要安装一个□大的emulators/linux_base-f10 port。

   下一□是安装 www/linux-f10-flashplugin10 port。 □将会安装 Flash™ 10.X， 此版本目前能在 FreeBSD 8.X 上正常□行。

   □个版本需要□建一个符号□接：

   ```
   # ln -s /usr/local/lib/npapi/linux-f10-flashplugin/libflashplayer.so \
     /usr/local/lib/browser_plugins/
   ```

   如果系□中没有 /usr/local/lib/browser_plugins 目□， □□手工□建它。

按照 FreeBSD 版本， 在安装了正□的 Flash™ port 之后， □件必□由□个用□□行 nspluginwrapper 安装：

```
% nspluginwrapper -v -a -i
```

如果希望播放 Flash™ □画的□，Linux® 的□程文件系□， linprocfs(5) 必□挂□于 /usr/compat/linux/proc。可以通□以下的命令□□：

```
# mount -t linprocfs linprocfs /usr/compat/linux/proc
```

□也可以在机器□□□自□挂□， 把以下□行加入 /etc/fstab：

```
linproc /usr/compat/linux/proc  linprocfs   rw  0   0
```

然后就可以打□□□器， 并在地址□中□入 about:plugins 然后按下 Enter。 □将□示目前可用的□件列表。

7.2.4. Firefox and Swfdec Flash™ Plugin

Swfdec 是一个用以解□和□染 Flash™ □画的□。 Swfdec-Mozilla 是一个使用了 Swfdec □□ Firefox 能播放 SWF 文件的□件。它目前仍□于□□状□。

如果□不能或者不想□□安装，可以通□网□安装二□制包：

```
# pkg_add -r swfdec-plugin
```

如果二□制包□不可用，□可以通□ Ports Collection □□安装：

```
# cd /usr/ports/www/swfdec-plugin
# make install clean
```

然后重□□的□□器使得□个□件生效。

7.2.5. Opera

Opera 是一个功能□全， 并符合□准的□□器。 它□提供了内建的□件和新□□□器、 IRC 客□端， RSS/Atom feed □□器以及更多功能。 除此之外， Opera 是一个比□□量的□□器， 其速度很快。 它提供了□□不同的版本："native" FreeBSD 版本， 以及通□ Linux 模□□行的版本。

要使用 Opera 的 FreeBSD 版本来□□网□，安装以下的 package：

```
# pkg_add -r opera
```

有些 FTP 站点没有所有版本的 package， 但仍然可以通□ Ports 套件来安装 Opera：

```
# cd /usr/ports/www/opera
# make install clean
```

要安装 Linux 版本的 Opera， 将上面例子中的 `opera` 改□ `linux-opera` 即可。

Adobe® Flash™ □件目前并没有提供 FreeBSD □用的版本。 不□， 可以使用其 Linux® 版本的□件。 要安装□个版本， 需要安装 www/linux-f10-flashplugin10 port， 以及 www/opera-linuxplugins：

```
# cd /usr/ports/www/linux-f10-flashplugin10
# make install clean
# cd /usr/ports/www/opera-linuxplugins
# make install clean
```

然后可以□□□件是否可用了： 在地址□中□入 `opera:plugins` 然后按 `Enter`。 □□器将列出可用的□件列表。

添加 Java™ □件的方法， 与 □ Firefox 添加□件 的方法相同。

7.2.6. Konqueror

Konqueror 是 KDE 的一部分，不□也可以通□安装 x11/kdebase3 在非 KDE □境下使用。 Konqueror 不止是一个□□器， 也是一个文件管理器和多媒体播放器。

也有□□□富的□件能□配合 Konqueror 一起使用， □可以通□ misc/konq-plugins 来安装它□。

Konqueror 也支持 Flash™； □于如何□得用于 Konqueror 的 Flash™ 支持的 "How To" 文□ 可以在 http://freebsd.kde.org/howtos/konqueror-flash.php □到。

7.3. □公、□象□理

当需要□行□公或者□行□象□理□， 新用□通常都会□一些好用的□公套件或者字□理□件。 尽管目前有一些 □面□境， 如 KDE 已□提供了□公套件， 但目前□□没有一定之□。 无□□使用那□□面□境， FreeBSD 都能提供□需要的□件。

□□□及如下程序：

□件名称	□源需求	安装□□	主要依□
KOffice	少	多	KDE
AbiWord	少	少	Gtk+ 或 GNOME
The Gimp	少	□	Gtk+
OpenOffice.org	多	□	JDK™、 Mozilla
LibreOffice	□重	巨大	Gtk+ 或 KDE/ GNOME 或 JDK™

7.3.1. KOffice

KDE 社区提供了一套□公套件， 它能用在□面□境。它包含四个□准的□件，□些□件可以在其它□公套件中□到。KWord 是字□理程序、 KSpread 是□子表格程序、 KPresenter 是演示文□制作管理程序、 Kontour 是矢量□□□件。

安装最新的 KOffice 之前，先□定□是否安装了最新版的 KDE。

使用 package 来安装 KOffice，安装□□如下：

```
# pkg_add -r koffice
```

如果没有可用的 package，□可以使用 Ports Collection 安装。 安装 KDE3 的 KOffice 版本，如下：

```
# cd /usr/ports/editors/koffice-kde3
# make install clean
```

7.3.2. AbiWord

AbiWord 是一个免□的字□理程序，它看起来和 Microsoft® Word 的感□很相似。 它□合用来打印文件、信函、□告、□忘□等等， 它非常快且包含□多特性，并且非常容易使用。

AbiWord 可以□入或□出很多文件格式， 包括一些象 Microsoft® .doc □□□有格式的文件。

AbiWord 也有 package 的安装方式。□可以用以下方法安装：

```
# pkg_add -r abiword
```

如果没有可用的 package，它也可以从 Ports Collection □□。ports collection □□是最新的。它的安装方式如下：

```
# cd /usr/ports/editors/abiword
# make install clean
```

7.3.3. GIMP

□□象的□□或者加工， GIMP 是一个非常精通□象□理的□件。 它可以被用来当作□□的□□程序或者一个□□的照片□理套件。 它支持大量的□件和具有脚本界面的特性。 GIMP 可以□写□多的文件格式， 支持□描□和手写板。

□可以用下列命令安装：

```
# pkg_add -r gimp
```

如果□在 FTP 站点没有□到□个 package，□也可以使用 Ports Collection 的方法安装。ports 的 graphics 目□也包含有 Gimp 手册。 以下是安装它□的方法：

```
# cd /usr/ports/graphics/gimp
# make install clean
# cd /usr/ports/graphics/gimp-manual-pdf
# make install clean
```

Ports 中的 graphics 目□也有□□中的 GIMP 版本 graphics/gimp-devel。 HTML 版本的 Gimp 手册 可以在 graphics/gimp-manual-html □到。

7.3.4. OpenOffice.org

OpenOffice.org 包括一套完整的□公套件： 字□理程序、 □子表格程序、 演示文□管理程序和□□程序。 它和其它的□公套件的特征非常相似，它可以□入□出不同的流行的文件格式。 它支持□多□□言 - 国□化已□□透到了其界面、 □写□□和字典等各个□面。

OpenOffice.org 的字□理程序使用 XML 文件格式使它□加了可移植性和□活性。 □子表格程序支持宏□言和使用外来的数据□界面。 OpenOffice.org 已□可以平□的□行在 Windows®、Solaris™、Linux、FreeBSD 和 Mac OS® X 等各□操作系□下。 更多的有□ OpenOffice.org 的信息可以在 OpenOffice.org 网□ □到。 □于特定的 FreeBSD 版本的信息，□可以在直接在 FreeBSD OpenOffice 移植□□的□面下□。

安装 OpenOffice.org 方法如下：

```
# pkg_add -r openoffice.org
```

如果□正在使用 FreeBSD 的 -RELEASE 版本， 一般来□□□做是没□□的。 如果不是□□，□就可能需要看一看 FreeBSD OpenOffice.org 移植小□的网站， 并使用 pkg_add(1) 从那里下□并安装合□的□件包。 最新的□布版本和□□版本都可以在那里□到。

装好 package 之后， □只需□入下面的命令就能□行 OpenOffice.org 了：

```
% openoffice.org
```

在第一次□行□， 将□□□一些□□， 并在□的主目□中建立一个 .openoffice.org 目□。

如果没有可用的 OpenOffice.org package，□仍旧可以□□□□ port。然而， □必□□住它的要求以及大量的磁□空□和相当□的□□□□。

```
# cd /usr/ports/editors/openoffice.org-3
# make install clean
```

如果希望□□一套□行□本地化的版本， 将前述命令行改□：

```
# make LOCALIZED_LANG=your_language install clean
```

□需要将 *your_language* 改□正□的 ISO-代□。 所支持的□言代□可以在 files/Makefile.localized 文件中□到， □个文件位于 port 的目□。

一旦完成上述操作， 就可以通□下面的命令来□行 OpenOffice.org 了：

```
% openoffice.org
```

7.3.5. LibreOffice

LibreOffice 是由 The Document Foundation □□的自由□件□公套件， 它与其他平台上的主流□公系□兼容。□是 OpenOffice.org 的一个□牌的分支版本， 包含了完整□公效率套件中必□的□用： 文字□理、 □子表格、幻灯演示、 □□工具、 数据□管理程序， 以及用于□建和□□数学公式的程序。 它提供了□多不同□言的支持 - 国□化支持除了界面之外， □包括了□写□□器和字典。

LibreOffice 的字□理程序使用了内建的 XML 文件格式， 以期□得更好的可移植性和□活性。□子表格程序提供了一□可以与外部数据□交互的宏□言支持。 LibreOffice 目前已□可以□定□行于 Windows®、 Linux、 FreeBSD 和 Mac OS® X。 □于 LibreOffice 的更多信息可以在 LibreOffice 网站□到。

如果希望通□□□□的二□制包安装 LibreOffice， □行：

```
# pkg_add -r libreoffice
```

如果□行的是 FreeBSD 的 -RELEASE 版本， □个命令□□不会遇到任何□□。

装好□件包之后， 需要用下面的命令来安装 LibreOffice：

```
% libreoffice
```

在首次□行□， 系□会□□一系列□□， 并在当前用□的主目□中□建 .libreoffice 目□。

如果 LibreOffice □件包不可用， □□是可以通□ port 安装。 不□， □注意□□它需要相当多的磁□空□和□□。

```
# cd /usr/ports/editors/libreoffice
# make install clean
```

如果希望□□本地化的版本， 把前面的命令□成：

```
# make LOCALIZED_LANG=your_language install clean
```

□需要把 *your_language* □成正□的□言 ISO 代□。 可用的代□可以在 port 的 Makefile 中的 `pre-fetch` target 中□到。

完成□□和安装之后， 就可以用下面的命令□行 LibreOffice 了：

```
% libreoffice
```

7.4. 文□□看器

UNIX® 系□出□以来， 一些新的文□格式□始流行起来； 它□所需要的□准□看器可能不一定在系□内。 本□中，我□将了解如何安装它□。

□□涵□如下□用程序:

□件名称	□源需求	安装□□	主要依□
Acrobat Reader®	少	少	Linux二□制兼容
gv	少	少	Xaw3d
Xpdf	少	少	FreeType
GQview	少	少	Gtk+ 或 GNOME

7.4.1. Acrobat Reader®

□在□多文□都用 PDF 格式， 根据"□便小巧文□格式"的定□。一个被建□使用的□看器是 Acrobat Reader®，由 Adobe 所□行的 Linux 版本。因□ FreeBSD 能□□行 Linux 二□制文件， 所以它也可以用在 FreeBSD 中。

要从 Ports collection 安装 Acrobat Reader® 8， 只需：

```
# cd /usr/ports/print/acroread8
# make install clean
```

由于授□的限制， 我□不提供□□□的版本。

7.4.2. gv

gv 是 PostScript® 和 PDF 文件格式□看器。它源自 ghostview 因□使用 Xaw3d 函数□□它看起来更美□。它很快而且界面很干□。gv 有很多特性比如象□□大小、刻度或者抗□□。 大部分操作都可以只用□□或鼠□完成。

安装 gv package，如下：

```
# pkg_add -r gv
```

如果□无法□取□□□的包， □可以使用 Ports Collection：

```
# cd /usr/ports/print/gv
# make install clean
```

7.4.3. Xpdf

如果□想要一个小型的 FreeBSD PDF □看器， Xpdf 是一个小巧并且高效的□看器。 它只需要很少的□源而且非常□定。它使用□准的 X 字体并且不需要 Motif® 或者其它的 X 工具包。

安装 Xpdf package，使用如下命令：

```
# pkg_add -r xpdf
```

如果 package 不可用或者□□□使用 Ports Collection，如下：

```
# cd /usr/ports/graphics/xpdf
# make install clean
```

一旦安装完成，□就可以□□ Xpdf 并且使用鼠□右□来使用菜□。

7.4.4. GQview

GQview 是一个□片管理器。 □可以□□鼠□来□看一个文件、□□一个外部□□器、 使用□□和更多的功能。它也有幻灯片播放模式和一些基本的文件操作。 □可以管理采集的□片并且很容易□到重□的。 GQview 可以全屏幕□看并且支持国□化。

如果□想要安装 GQview package，如下：

```
# pkg_add -r gqview
```

如果□没有可用的 package 或者□□□使用 Ports Collection，如下：

```
# cd /usr/ports/graphics/gqview
# make install clean
```

7.5. □□

假如，基于任何的理由，□想要在 FreeBSD Desktop 管理□个人的□政，有一些□大并且易于使用的□件可以被□□□安装。 它□中的一些与流行的文件格式兼容象 Quicken 和 Excel 文件。

本□涵□如下程序：

□件名称	□源需求	安装□□	主要依□
GnuCash	少	□	GNOME
Gnumeric	少	□	GNOME
Abacus	少	少	Tcl/Tk
KMyMoney	少	□	KDE

7.5.1. GnuCash

GnuCash 是 GNOME 的一部分，GNOME 致力于□最□用□提供用□友好且功能□大的□件。使用 GnuCash，□可以□注□的收入和□支、□的□行□□， 或者□的股票。它的界面特性看起来非常的□□。

GnuCash 提供一个智能化的注册、□□分□系□、 很多□□快捷方式和自□完成方式。它能分□一个□个的□理到几个□□的部分。 GnuCash 能□入和合并 Quicken QIF 文件格式。 它也支持大部分的国□日期和流行的格式。

在□的系□中安装 GnuCash 所需的命令如下：

```
# pkg_add -r gnucash
```

如果 package 不可用，□可以使用 Ports Collection 安装：

```
# cd /usr/ports/finance/gnucash
# make install clean
```

7.5.2. Gnumeric

Gnumeric 是一个□子表格程序， GNOME □面□境的一部分。 它以通□元素格式和□多片断的自□填充系□来方便的自□"猜□"用□□入而著称。 它能□入一些流行的文件格式，比如象 Excel、 Lotus 1-2-3 或 Quattro Pro。 Gnumeric 凭借 math/guppi 支持□表。 它有大量的嵌入函数和允□所有通常比如象、数字、□□、日期、 □□等等的一些□元格式。

以 package 方式安装 Gnumeric 的方法如下：

```
# pkg_add -r gnumeric
```

如果 package 不可用，□可以使用 Ports Collection 安装：

```
# cd /usr/ports/math/gnumeric
# make install clean
```

7.5.3. Abacus

Abacus 是一个小巧易用的□子表格程序。 它包含□多嵌入函数在一些□域如□□学、□□和数学方面很有□助。它能□入和□出 Excel 文件格式。 Abacus 可以□生 PostScript® □出。

以 package 的方式安装 Abacus 的方法如下：

```
# pkg_add -r abacus
```

如果 package 不可用，□可以使用 Ports Collection 安装：

```
# cd /usr/ports/deskutils/abacus
# make install clean
```

7.5.4. KMyMoney

KMyMoney 是一个 KDE□境下的个人□□管理□件。 KMyMoney 旨在提供并融合各□商□□□管理□件所有的重要特性。 它也同□注重易用性和特有的□式□□功能。 KMyMoney 能从□准的 Quicken Interchange Format (QIF) 文件□入数据， 追踪投□，□理多□□□并能提供一个□□□告。 □有可用的□件支持□入 OFX 格式的数据。

以 package 的方式安装 KMyMoney 的方法如下：

```
# pkg_add -r kmymoney2
```

如果 package 不可用，□可以使用 Ports Collection 安装：

```
# cd /usr/ports/finance/kmymoney2
# make install clean
```

7.6. □□

尽管 FreeBSD 由于其高性能和可□性而□得了□多 ISP 的信□， 但它也完全可以用于□面□境。 □有数以千□的 packages 和 ports 能□□□迅速建立完美的□面□境。

下面是本章□及到的所有的□件的□要回□：

□件名称	**Package** 名称	**Ports** 名称
Opera	opera	www/opera
Firefox	firefox	www/firefox
KOffice	koffice	editors/koffice-kde3
AbiWord	abiword	editors/abiword

□件名称	**Package** 名称	**Ports** 名称
The GIMP	gimp	graphics/gimp
OpenOffice.org	openoffice	editors/openoffice.org-3
LibreOffice	libreoffice	editors/libreoffice
Acrobat Reader®	acroread	print/acroread8
gv	gv	print/gv
Xpdf	xpdf	graphics/xpdf
GQview	gqview	graphics/gqview
GnuCash	gnucash	finance/gnucash
Gnumeric	gnumeric	math/gnumeric
Abacus	abacus	deskutils/abacus
KMyMoney	kmymoney2	finance/kmymoney2

Chapter 8. 多媒体

8.1. 概述

FreeBSD 广泛地支持各□声□， □□可以从容地享受来自□的□算机的高保真□出。 □包括了□制和播放 MPEG Audio Layer 3 (MP3)、 WAV、 以及 Ogg Vorbis 等□多□格式声音的能力。 FreeBSD 同□也包括了□多的□用程序，□□可以□音、 □加声音效果以及控制附加的MIDI□□。

要是□于□手， FreeBSD 也能支持播放一般的□□文件和 DVD。 □各□□□媒体□行□□、 □□和播放的□用程序比起□理声音的□用程序略少一些。 例如， 在撰写□章□， FreeBSD Ports Collection 中□没有□似 audio/sox 那□好的重□□工具能□用来在不同的格式之□□□。 不□， □个□域的□件研□□展是很快的。

本章将介□配置声□的必要□□。 X11 的安装和配置 (X Window 系□) 里已□考□到了□□□的□□，但要想有更好的播放效果， 仍需要□整一些□西。

□了本章后，□将知道：

- 如何配置系□□□声□。
- □□声□是否正常工作的方法。
- 如何排除声□安装中的□□。
- 如何播放和□□MP3以及其它格式的音□。
- X 服□器如何支持□□。
- □些好的□□播放/□□"ports"。
- 如何播放 DVD、 .mpg 以及 .avi 文件。
- 如何从 CD 和 DVD 中提取文件。
- □□配置□□□。
- 如何配置□像□描□。

在□本章□前，□□□：

- 知道如何配置、安装一个新的内核 (配置FreeBSD的内核)

用mount(8) 命令去装□CD光□，至少会□生一个□□， 更糟的情况下会□生 *kernel panic*。□□媒体所用的□□与通常的ISO文件系□是不同的。

8.2. 安装声□

8.2.1. 配置系□

在□始之前，□□□清楚声□□型、所用的芯片以及它是 PCI □是 ISA □。 FreeBSD 支持□□繁多的 PCI 和 ISA □。□□硬件兼容□明 中支持的音□□□列表看看是否支持□的声□， 硬件兼容□明也会□明支持□声□的是□个□□程序。

要使用声□， 就□装□正□的□□程序。完成的方式有□□： 最□□的是使用命令 kldload(8) 来装□一个内核模□，在命令行□入

```
# kldload snd_emu10k1
```

或者在文件 /boot/loader.conf 里加入一行，内容如下

```
snd_emu10k1_load="YES"
```

上□□例用于 Creative SoundBlaster® Live! 声□。 其它可装□的模□列在文件 /boot/defaults/loader.conf 里□。 如果不知道□□使用□个□□， □可以□□加□ snd_driver module:

```
# kldload snd_driver
```

□是个 meta □□，一次加□了最常□的□□□□。 □会提高搜索正□□□的速度。也可以通□ /boot/loader.conf 工具来加□所有的声□□□。

如果希望在加□了 snd_driver meta □□之后了解到底□□了□□声□， 可以通□使用 `cat /dev/sndstat` 来□□ /dev/sndstat 文件。

□外，□也可以把支持□声□的代□静□地□□到内核里去。 下一□就采用□□方式支持硬件□出提示。 □于重新□□内核，□参考 配置FreeBSD的内核。

8.2.1.1. 定制内核使其支持声□

要做的第一件事情就是添加通用音□框架□□ sound(4) 到内核中， □需要添加下面□行到内核配置文件中：

```
device sound
```

接下来就是加入□我□所用声□的支持了。 首先需要□定我□的声□需要使用□一个□□。 □可以参考 硬件兼容列表 所列出的音□□□， 以□定□声□的□□。 例如， Creative SoundBlaster® Live! 声□由 snd_emu10k1(4) □□来支持。 要添加它， 需要在内核□□配置文件中加入下面一行：

```
device snd_emu10k1
```

一定要□□□□的□机手册了解如何使用它□。 □于内核配置文件中声□□□的具体写法， 也可以在 /usr/src/sys/conf/NOTES 文件中□到。

非即□即用的 ISA □可能需要□□内核提供一些□于声□配置的信息 (IRQ、 I/O 端口， 等等)， □一点与其他不支持即□即用的 ISA □□似。 □□工作可以通□ /boot/device.hints 文件来完成。 系□□□□， loader(8) 将□取□个文件， 并将其中的配置□□内核。 例如， 旧式的 Creative SoundBlaster® 16 ISA 非即□即用□需要使用 snd_sbc(4) □□并配合 snd_sb16(4)。 □可以在内核□□配置文件中□加如下配置：

```
device snd_sbc
device snd_sb16
```

□有下面□些到 /boot/device.hints中：

```
hint.sbc.0.at="isa"
hint.sbc.0.port="0x220"
hint.sbc.0.irq="5"
hint.sbc.0.drq="1"
hint.sbc.0.flags="0x15"
```

□□，声□使用 0x220 I/O 端口和 IRQ 5。

在 /boot/device.hints 文件中所使用的□法， 在 sound(4) □机手册中以及所用的具体声□□□的□机手册中，会□行□一□的□解。

上面所展示的是默□的配置。 有□候， □可能需要更改 IRQ 或其他配置， 以□□声□的□□情况。 □看 snd_sbc(4) □机手册了解更多信息。

8.2.2. □□声□

用修改□的内核重起，或者加□了需要的模□之后， 声□将会出□在□的系□消息□存中 (dmesg(8))，就像□□：

```
pcm0: <Intel ICH3 (82801CA)> port 0xdc80-0xdcbf,0xd800-0xd8ff irq 5 at device 31.5 on
pci0
pcm0: [GIANT-LOCKED]
pcm0: <Cirrus Logic CS4205 AC97 Codec>
```

声□的状□可以通□ /dev/sndstat 文件来□□：

```
# cat /dev/sndstat
FreeBSD Audio Driver (newpcm)
Installed devices:
pcm0: <Intel ICH3 (82801CA)> at io 0xd800, 0xdc80 irq 5 bufsz 16384
kld snd_ich (1p/2r/0v channels duplex default)
```

□系□的□出可能与此不同。如果没有看到 pcm □□，回□并□□一下前面做的。 重新□□□的内核配置文件并保□□□了正□的□□。 常□□□列在 常□□□ 一□。

如果一切正常，□□在□□□有一个多功能声□了。 如果□的 CD-ROM 或者 DVD-ROM □□器的音□□出□已□与声□□在一起， □可以把 CD 放入□□器并用 cdcontrol(1) 来播放：

```
% cdcontrol -f /dev/acd0 play 1
```

□多□用程序，比如 audio/workman 可以提供一个友好的界面。 □可能想要安装一个□用程序比如 audio/mpg123 来听 MP3 音□文件。

□一□快速□□声□的方法， 是将数据□送到 /dev/dsp， 像□□做：

```
% cat filename > /dev/dsp
```

□里 filename 可以是任意文件。 □行命令会□生一些噪音，□明声□果真在工作。

□□□点 /dev/dsp* 会在需要的□候自□□生。 如果没有使用它□， □它□不会出□在 ls(1) 的□出中。

声□混音□□可以通□ mixer(8) 命令更改。 更多□□可以在 mixer(8) □机手册中□到。

8.2.2.1. 常□□□

□□信息	解决方法
sb_dspwr(XX) timed out	I/O端口没有□置正□。
bad irq XX	IRQ□置不正□。□信□定的IRQ和声□的IRQ是一□的。
xxx: gus pcm not attached, out of memory	没有足□的内存空□供□置使用。
xxx: can't open /dev/dsp!	使用命令 `fstat

□一个□□是□多新式的□□本身包含它□自己的声音□□， 用以配合 HDMI □□的□□使用。 □个声音□□有□会在真正的声□之前被探□到， 从而成□默□的回放□□， 而使真正的声□无法□声。 要□□□□情况， □行 dmesg 并□察 pcm。 其□出□似下面□□：

```
...
hdac0: HDA Driver Revision: 20100226_0142
hdac1: HDA Driver Revision: 20100226_0142
hdac0: HDA Codec #0: NVidia (Unknown)
hdac0: HDA Codec #1: NVidia (Unknown)
hdac0: HDA Codec #2: NVidia (Unknown)
hdac0: HDA Codec #3: NVidia (Unknown)
pcm0: <HDA NVidia (Unknown) PCM #0 DisplayPort> at cad 0 nid 1 on hdac0
pcm1: <HDA NVidia (Unknown) PCM #0 DisplayPort> at cad 1 nid 1 on hdac0
pcm2: <HDA NVidia (Unknown) PCM #0 DisplayPort> at cad 2 nid 1 on hdac0
pcm3: <HDA NVidia (Unknown) PCM #0 DisplayPort> at cad 3 nid 1 on hdac0
hdac1: HDA Codec #2: Realtek ALC889
pcm4: <HDA Realtek ALC889 PCM #0 Analog> at cad 2 nid 1 on hdac1
pcm5: <HDA Realtek ALC889 PCM #1 Analog> at cad 2 nid 1 on hdac1
pcm6: <HDA Realtek ALC889 PCM #2 Digital> at cad 2 nid 1 on hdac1
pcm7: <HDA Realtek ALC889 PCM #3 Digital> at cad 2 nid 1 on hdac1
...
```

此□□□ (NVidia) 先于真正的声□ (Realtek ALC889) 被探□到。 要使用声□作□默□的回放□□， 将 hw.snd.default_unit 改□□□的□□□号：

```
# sysctl hw.snd.default_unit=n
```

□里的 n 是希望使用的声音□□□号， 在□个例子中是 4。 □可以在 /etc/sysctl.conf 中写上□个配置来令其永久性生效：

```
hw.snd.default_unit=4
```

8.2.3. 利用多个声源

通常而言， 会希望多个音源能□同□播放， 例如， esound 或者 artsd 就可能不支持与其它程序共享音□□□。

FreeBSD 可以通□ 虚□声道*(Virtual Sound Channels)* 来□到□□的效果， 它可以用 sysctl(8) 来□用。 虚□的声道可以能□在内核里混合声音来混合声□里播放的声道。

使用三条sysctl命令来□置虚□声道的数目。 如果□是 root 用□， □行下面的操作：

```
# sysctl dev.pcm.0.play.vchans=4
# sysctl dev.pcm.0.rec.vchans=4
# sysctl hw.snd.maxautovchans=4
```

上面的□例□定了4个虚□声道，□也是□□上所使用的数目。 dev.pcm.0.play.vchans=4 和 dev.pcm.0.rec.vchans=4 是 pcm0 用来播放与□音的虚□声道数， 一当□接上一个□□它就可配置了。 hw.snd.maxautovchans 是分配□新的音□□□的虚□声道数， 此□□个□□要用 kldload(8) 来□接。 因□ pcm 模□可以独立装□□多硬件□□程序， 因此 hw.snd.maxautovchans 也就可以存□分配□以后□接到的□□的虚□声道数。 可参□ pcm(4) 手册□□□取更多□□。

□不能在使用某个□□的□候改□其虚□通道数。 首先需要□□所有使用□□□的程序， 如音□播放器或声音服□。

当□用程序□求 /dev/dsp0 □， 系□会自□□其分配正□的 pcm □□。

8.2.4. 如何□置混音器通道□

不同的混音通道的默□音量是硬□□□ pcm(4) □□程序的。 同□， 也有很多□用或服□程序提供了允□用□直接□置并□住□些□的功能。 不□□并不是一个很好的解决方案， □可能希望在□□一□有一个可以□置的默□□。 □可以通□在 /boot/device.hints 定□□当的□来□□。 例如：

```
hint.pcm.0.vol="50"
```

□将在 pcm(4) 模□加□□， 将通道音量□置□默□的 50。

8.3. MP3音□

MP3 (MPEG Layer 3 Audio)□到□CD音□的效果，FreeBSD工作站没理由会缺少□□的好□□。

8.3.1. MP3播放器

目前□止， 最□流行的 X11 MP3 播放器是 XMMS (X 多媒体系□)。 Winamp 的□面可以直接用于 XMMS，因□它的 GUI 几乎和 Nullsoft 的 Winamp 完全一□。 □外， XMMS 也提供了内建的□件支持。

XMMS 可以通□ multimedia/xmms port 或 package 来安装。

XMMS 的界面很直□， 它提供了播放列表、 □形化均衡器等等。 如果□熟悉 Winamp， 就会感□ XMMS 很容易使用。

audio/mpg123 port 提供了一个命令行界面的 MP3 播放器。

mpg123 可以在□行□通□命令行指定声音□□和要播放的 MP3 文件， 假□□的声音□□是 /dev/dsp1.0 并且□想要播放的 MP3 文件□ *Foobar-GreatestHits.mp3* □可以□入以下的命令：

```
# mpg123 -a /dev/dsp1.0 Foobar-GreatestHits.mp3
High Performance MPEG 1.0/2.0/2.5 Audio Player for Layer 1, 2 and 3.
Version 0.59r (1999/Jun/15). Written and copyrights by Michael Hipp.
Uses code from various people. See 'README' for more!
THIS SOFTWARE COMES WITH ABSOLUTELY NO WARRANTY! USE AT YOUR OWN RISK!

Playing MPEG stream from Foobar-GreatestHits.mp3 ...
MPEG 1.0 layer III, 128 kbit/s, 44100 Hz joint-stereo
```

8.3.2. □取CD音□

在□CD或CD音□□□成MP3之前， CD上的音□数据□先□到硬□里。 □个可以通□□制原始的CDDA(CD数字音□)数据成□波形(WAV)文件。

工具 `cdda2wav` 是 sysutils/cdrtools 套件的一部□，可用来从CD中□取音□及其相□信息。

把CD放到光□里，下面的命令可以完成 (作□ `root`用□) 把整□ CD 分割成□个 (□个音□) 的WAV文件：

```
# cdda2wav -D 0,1,0 -B
```

cdda2wav 支持 ATAPI (IDE)光□。 从IDE光□中□取音□， 需要用□□名称代替SCSI的□元号。 例如， 想从 IDE 光□中□取第7道音□：

```
# cdda2wav -D /dev/acd0 -t 7
```

参数 `-D 0,1,0` 表示 SCSI □□ 0,1,0， 与命令 `cdrecord -scanbus` 的□出相□□。

□取□□，要使用□□ `-t`，如下所示：

```
# cdda2wav -D 0,1,0 -t 7
```

□个□例用于□取第七个音□。要□取一定□□的音□，如从1到7：

```
# cdda2wav -D 0,1,0 -t 1+7
```

利用dd(1)也可以从ATAPI光□中□取音□，从 □制音□ CD 可以了解更多。

8.3.3. MP3 □□

□今，可□的MP3□□器是 lame。 Lame 可以从ports□里的 audio/lame □□到。

利用□取的WAV文件，下□的命令就可以把 audio01.wav □□成 audio01.mp3：

```
# lame -h -b 128 \
--tt "Foo Song Title" \
--ta "FooBar Artist" \
--tl "FooBar Album" \
--ty "2001" \
--tc "Ripped and encoded by Foo" \
--tg "Genre" \
audio01.wav audio01.mp3
```

128 kbits 是□准的MP3位率(bitrate)。 □多人可能喜□更高的品□例如 160 或 192。 更高的位率， 会使 MP3 占用更多的磁□空□—但音□会更高。□□ -h 控制 "高品□但低速度 (higher quality but a little slower)" 模式的□□。 □□ --t 表示把 ID3 □□—通常包含了歌曲的信息， 植入到MP3文件里。 其它的□□□□可以□□ lame 的□机手册。

8.3.4. MP3 解□

要把MP3歌曲刻□成音□CD，就需要把它□□成非□□的波形(WAV)格式。 XMMS 和 mpg123 都支持把 MP3□出成非□□格式文件。

在 XMMS 中□出到磁□：

1. □□ XMMS.
2. 在□口里右□鼠□，□出 XMMS 菜□。
3. 在 □□(Options) 里□□ □定(Preference)。
4. 改□□出□件成 "写磁□□件(Disk Writer Plugin)"。
5. 按 配置(Configure)。
6. □入或□□一个目□用于存放解□的文件。
7. 象平常一□，把MP3文件装入到 XMMS 里□， 把音量□□到100%并且□掉EQ□定。
8. 按一下 播放(Play) - XMMS 如同在播放mp3一□，只是听不到声音。 □□上是在播放 mp3到一个文件里。
9. 要想再听MP3歌曲，□得把默□的□出□件□回原来的□。

用 mpg123 □行□准□出：

1. □行 ` mpg123 -s audio01.mp3 > audio01.pcm `

XMMS □出的文件是波形(WAV)格式， 而 mpg123 □把MP3□□成无□□的PCM 音□数据。□□格式都支持用 cdrecord 刻□成音□CD。 使用 burncd(8) □就必□使用无□□的PCM。 如果□□波形格式， 就要注意在□道□始□的一小点□音， □段声音是波形文件的□部□。 可以使用工具 SoX 来□松去除。 SoX 可从 audio/sox port 或包(package)中安装得到：

```
% sox -t wav -r 44100 -s -w -c 2 track.wav track.raw
```

□□ □建和使用光学介□(CD) □部□可以了解到更多在 FreeBSD 里刻□的信息。

8.4. □□回放

□□回放是个很新并且迅速□展中的□用□域。 一定要有耐心，因□不是所有的事情都象□音□那□□利。

在□始之前，□要了解□□的□型以及它所用的芯片的□型。 尽管 Xorg 支持大量的□□， 但能□到好的回放效果的却□□无几。 在X11□行□，□可以使用命令 xdpyinfo(1) □得使用□的□□的X服□器所支持的□展列表。

□了□估各□播放器和□置，□需要有一小段用作□□的MPEG文件。 由于一些DVD播放器会默□地在 /dev/dvd 里去□DVD文件， 因此， □会□□建立符号□接到恰当的□□会很有用：

```
# ln -sf /dev/acd0 /dev/dvd
# ln -sf /dev/acd0 /dev/rdvd
```

注意：由于 devfs(5) 本身的原因， 像□□手工建立的□接在重□后将不会存在。 想要无□什□□候□□□系□都能自□建立符号□接， 那就把下□□行加到 /etc/devfs.conf 里□：

```
link acd0 dvd
link acd0 rdvd
```

□外，DVD解密要求□用□用的DVD-ROM函数，要求把□可定到DVD□□里。

□了改善 X11 界面使用共享内存的能力， 建□提高一些 sysctl(8) □量的□：

```
kern.ipc.shmmax=67108864
kern.ipc.shmall=32768
```

8.4.1. □定□□的性能

在X11下有几□可以□示□像的方式。 到底□个能工作很大程度上依□于硬件。 首先， 下□描述的□一□方法在不同的硬件上都会有不同的品□。 其次， 在X11里的□像□示近来引起普遍的□注， 随着 Xorg 的□一个版本， 都会有很大的突破。

常□□像接口列表：

1. X11: 一般性的使用共享内存的X11□出。
2. XVideo: 一□X11接口□展，支持任何X11□像的可□拉。
3. SDL: □□直接媒体□。
4. DGA: 直接□片存取。
5. SVGAlib: 低□次掌控□片□。

8.4.1.1. XVideo

Xorg 有□□展叫做 *XVideo* (或称Xvideo，Xv，xv)， 它可以通□一个特殊的加速器直接把□像□示在可□拉的□象里。 即使在低端的□算机 (例如我的PIII 400 Mhz膝上□□)， □个□展也提供了很好的播放□量。

要了解□一□展是否在正常工作， 使用 `xvinfo` 命令：

```
% xvinfo
```

如果□示□果如下，那□的□□就支持XVideo：

```
X-Video Extension version 2.2
screen #0
  Adaptor #0: "Savage Streams Engine"
    number of ports: 1
    port base: 43
    operations supported: PutImage
    supported visuals:
      depth 16, visualID 0x22
      depth 16, visualID 0x23
    number of attributes: 5
      "XV_COLORKEY" (range 0 to 16777215)
              client settable attribute
              client gettable attribute (current value is 2110)
      "XV_BRIGHTNESS" (range -128 to 127)
              client settable attribute
              client gettable attribute (current value is 0)
      "XV_CONTRAST" (range 0 to 255)
              client settable attribute
              client gettable attribute (current value is 128)
      "XV_SATURATION" (range 0 to 255)
              client settable attribute
              client gettable attribute (current value is 128)
      "XV_HUE" (range -180 to 180)
              client settable attribute
              client gettable attribute (current value is 0)
    maximum XvImage size: 1024 x 1024
    Number of image formats: 7
      id: 0x32595559 (YUY2)
        guid: 59555932-0000-0010-8000-00aa00389b71
```

```
        bits per pixel: 16
        number of planes: 1
        type: YUV (packed)
      id: 0x32315659 (YV12)
        guid: 59563132-0000-0010-8000-00aa00389b71
        bits per pixel: 12
        number of planes: 3
        type: YUV (planar)
      id: 0x30323449 (I420)
        guid: 49343230-0000-0010-8000-00aa00389b71
        bits per pixel: 12
        number of planes: 3
        type: YUV (planar)
      id: 0x36315652 (RV16)
        guid: 52563135-0000-0000-0000-000000000000
        bits per pixel: 16
        number of planes: 1
        type: RGB (packed)
        depth: 0
        red, green, blue masks: 0x1f, 0x3e0, 0x7c00
      id: 0x35315652 (RV15)
        guid: 52563136-0000-0000-0000-000000000000
        bits per pixel: 16
        number of planes: 1
        type: RGB (packed)
        depth: 0
        red, green, blue masks: 0x1f, 0x7e0, 0xf800
      id: 0x31313259 (Y211)
        guid: 59323131-0000-0010-8000-00aa00389b71
        bits per pixel: 6
        number of planes: 3
        type: YUV (packed)
      id: 0x0
        guid: 00000000-0000-0000-0000-000000000000
        bits per pixel: 0
        number of planes: 0
        type: RGB (packed)
        depth: 1
        red, green, blue masks: 0x0, 0x0, 0x0
```

同□注意：列出来的格式(YUV2, YUV12, 等等) 并不□是随着 XVdieo的□一次□行而存在。没有它□可能会迷惑某些人。

如果□果看起来是□□：

```
X-Video Extension version 2.2
screen #0
no adaptors present
```

那□□的□□可以就不支持XVideo功能。

如果□的□不支持XVideo， □只是□明□的□示器在□足刷新□像的□算要求上存在更大的困□。 尽管□□和□理器很重要，□仍然会有个不□的□示效果。 此外， □也可以参考我□提供的文献， 在 □一□了解 中有所介□。

8.4.1.2. □□直接媒体□

□□直接媒体□(SDL)，原意是做□ Microsoft® Windows®、BeOS 以及 UNIX® 之□的端口□，允□跨平台□用□展，更高效地利用声□和□形□。SDL □可以在低□□□硬件， 有□□□做就比 X11 接口□更□高效。

□于 SDL， 可以参考 devel/sdl12。

8.4.1.3. 直接□形存取

直接□形存取 (Direct Graphics Access) 是一□ X11 □展， 通□它， □用程序能□□□ X 服□， 并直接修改画面□存 (framebuffer)。 由于它依□一□底□的内存映射来□□其功能， 因此使用它的程序必□以 `root` 身□来□行。

DGA □展可以通□ dga(1) 来完成□□和性能□量。 □行 `dga` □， 它将随按□改□□□的□色。 按 `q` 退出□个程序。

8.4.2. Ports 和 包(Packages) □□□的解决

□部□主要□□在 FreeBSD Ports 集中提供的可用于□□回放的□件。 □□回放在□件□展中是个很活□的□域，并且各□不同程序的功能可能与□里的描述不尽相同。

首先要弄清楚的重要一点是在 FreeBSD 上使用的□□程序其□展与在 Linux 里使用的是一□的。 大部□程序都□□在β□段。使用 FreeBSD 的包可能面□的□□：

1. 一个□用程序不能播放其它程序制作的文件。
2. 一个□用程序不能播放其自已制作的文件。
3. 不同机上的同□的程序，各自重新建立(rebuild)了一次， 播放同一个文件□果也会有不同。
4. 一个看起来没什□的□□器， 如□像尺寸的□整， 也有可能因□一个□整例程的□□□得很不象□。
5. □用程序□繁地留下□□(dumps core)。
6. 没有随 port 一起安装的文□可以在网上或者 port 的 work 目□中□到。

□些程序中□多也体□了 "Linux主□"。即， 有些□□来自于(程序)使用的□准□存在于Linux的□行版中，或者有些是 Linux 内核的功能， 而□程序的作者事先所假定了的是 Linux内核。□些□□并不□是被 port □□人□注意到或□理□， □也就可能□致如下□□：

1. 使用/proc/cpuinfo去□□□理器的特性。
2. □用□程可能□致一个程序□挂完成，而不是完全中止。
3. □件□不属于FreeBSD Ports集，而又与其它程序□常地一起使用。

□在，□些程序的□□人□也已同 port 的□□人□□行了□合， 以□少制作port□出□。

8.4.2.1. MPlayer

MPlayer 是近来□□的同□也正迅速□展着的一个□□播放器。 MPlayer □□的目□是在 Linux 和其它 UNIX 系□中的速度和机□性能。 在□□的□始人□在受不了当□可用的播放器的性能□， □个□□就□始了。 有人也□会

□□形接口已□成□新型□□的□牲品。 但是一旦□□□了命令行□□和按□控制方式，它就能表□得很好。

8.4.2.1.1. □建MPlayer

MPlayer 可以从 multimedia/mplayer □到。 MPlayer 在□□□程中会□行□多硬件□□， 而得到的可□行文件因此将无法移植到其他系□中使用。 因此， 从 ports 完成□□而不是安装□□□的包就很重要。 □外， 在 make 命令行□可以指定□多□□， 在 Makefile 中有所描述， 接下来我□□始□□：

```
# cd /usr/ports/multimedia/mplayer
# make
N - O - T - E

Take a careful look into the Makefile in order
to learn how to tune mplayer towards you personal preferences!
For example,
make WITH_GTK1
builds MPlayer with GTK1-GUI support.
If you want to use the GUI, you can either install
/usr/ports/multimedia/mplayer-skins
or download official skin collections from
http://www.mplayerhq.hu/homepage/dload.html
```

默□的 port □□□于□大多数用□来□是□用了。 不□， 如果□需要 XviD □解□器， □必□指定 WITH_XVID □个命令行□□。 默□的 DVD □□也可以用 WITH_DVD_DEVICE □□来定□， 其默□□是 /dev/acd0。

撰写□一章的□候， MPlayer port 的□□□程包括了 HTML 文□和□个可□行文件， mplayer 和 mencoder，后者是一个□□再□□工具。

MPlayer 的 HTML 文□提供了□富的内容。 如果□者□□本章中缺少□于□□硬件的一些信息， □ MPlayer 的文□将是十分□尽的□充。 如果□正在□□于 UNIX® 中的□□支持的□料， □□□□花一些□□来□□ MPlayer 的文□。

8.4.2.1.2. 使用MPlayer

任何 MPlayer 用□必□在其用□主目□下建立一个叫 .mplayer 的子目□。 □入下□的内容来建立□个必□的子目□：

```
% cd /usr/ports/multimedia/mplayer
% make install-user
```

在 mplayer 的手册里列出了它的命令□□。 HTML文□里有更□□□的信息。 □部□里， 我□只是描述了很少的常□□用。

要播放一个文件，如 testfile.avi， 可以通□各□□□接口当中的某一个去□置 -vo □□：

```
% mplayer -vo xv testfile.avi
```

```
% mplayer -vo sdl testfile.avi
```

```
% mplayer -vo x11 testfile.avi
```

```
# mplayer -vo dga testfile.avi
```

```
# mplayer -vo 'sdl:dga' testfile.avi
```

所有□些□□都是□得一□的， 因□它□的性能依□很多因素，并且都与硬件密切相□。

要播放 DVD， 需要把 testfile.avi 改□ dvd://N -dvd-device DEVICE。 □里 *N* 是要播放的□目□号， 而 DEVICE □是 DVD-ROM 的□□□点。 例如， 要播放 /dev/dvd 的第三个□目：

```
# mplayer -vo xv dvd://3 -dvd-device /dev/dvd
```

可以在□□ MPlayer □， 通□ WITH_DVD_DEVICE 来指定默□的 DVD □□。 系□内定的默□□□是 /dev/acd0。 更多□□， □参考 port 的 Makefile。

要停止、□停、前□等等，可以参考□定的按□---□些可以通□ mplayer -h 得到或□看手册。

□外，回放的重要□□是：用于全屏模式的 -fs -zoom 和起□助完成作用的`-framedrop`。

□了□ mplayer 的命令行不是太□，使用者可以通□建立一个文件 .mplayer/config 来□定如下默□□□：

```
vo=xv
fs=yes
zoom=yes
```

最后，mplayer 可以把DVD□目(title)□取成□ .vob 文件。□了从DVD中□出第二个□目，□□入：

```
# mplayer -dumpstream -dumpfile out.vob dvd://2 -dvd-device /dev/dvd
```

□出文件 out.vob 将是 MPEG 并且可以被□部□描述的其它 "包" 利用。

8.4.2.1.3. mencoder

在使用 mencoder 之前， 首先熟悉其 HTML 文□中所介□的□□是一个不□的主意。 它提供了□机手册，但如果没有 HTML 文□□□助不大。 有无数□方法来提高□□品□、 降低比特率、 修改格式， 而□些技巧可能会影□性能。 下面是几个例子， 第一个是□□地□制：

```
% mencoder input.avi -oac copy -ovc copy -o output.avi
```

不正□的命令□□□合可能使生成的文件不能被 mplayer 播放。因此，如果□只是想□取文件， 一定在 mplayer 里使用 “-dumpfile”。

□□ input.avi 成□□有MPEG3音□□□ (要求 audio/lame) 的MPEG4□□：

```
% mencoder input.avi -oac mp3lame -lameopts br=192 \
     -ovc lavc -lavcopts vcodec=mpeg4:vhq -o output.avi
```

□□就□生了可被 mplayer 和 `xine`播放的□出。

input.avi 可以□成 dvd://1 -dvd-device /dev/dvd 并以 root 的身□来□行， 以重新□ DVD □目□行□□。 由于□第一次做□□的工作□很可能会□□果不太□意， 建□□首先把□目□制成文件， 然后□它□行操作。

8.4.2.2. xine□□播放器

xine □□播放器是一个□注□□很广的□目， 它不□看准多合一的□□解决， 而且出品了一个可再用的基本□和一个可□展□件的可□行模□。 □行有 "包" 和port版本-- multimedia/xine。

xine 播放器仍然很粗□， 但□很□然与好□□无□。□□上 xine 要求□有快速的 CPU 和快速的□□来□行，或者需要支持 XVideo □展。 □形界面(GUI)可以使用，但很勉□。

到写□章□，□没有可用于播放CSS□□的DVD文件的□入模□随同 xine 一起□行。 第三方的建造(builds)里内建有□□的模□， 但都不属于FreeBSD Ports 集。

与MPlayer 相比， xine □用□考□得更多， 但同□，□用□来□也少了很多有条理的控制方式。 xine 播放器在XVideo接口上做得不□。

默□情况下，播放器 xine □□的□候会使用□形界面。那□就可以使用菜□打□指定的文件：

```
% xine
```

□外，没有□形界面也可以使用如下命令立即打□播放文件：

```
% xine -g -p mymovie.avi
```

8.4.2.3. 使用transcode

transcode □个□件并不是播放器， 而是一系列用于□□□和音□文件□行重新□□的工具。 通□使用 transcode，就可以□有使用□ stdin/stdout 接口的命令行工具来合并□□文件， 以及修□坏□文件的能力。

在□□ multimedia/transcode port □可以指定大量□□， 我□建□使用下面的命令行来□建 transcode：

```
# make WITH_OPTIMIZED_CFLAGS=yes WITH_LIBA52=yes WITH_LAME=yes WITH_OGG=yes \
WITH_MJPEG=yes -DWITH_XVID=yes
```

□于多数用□而言， 前述配置已□足□了。

□了□明 transcode 的功能， 下面的例子展示了如何将 DivX □□□ PAL MPEG-1 文件 (PAL VCD)：

```
% transcode -i input.avi -V --export_prof vcd-pal -o output_vcd
% mplex -f 1 -o output_vcd.mpg output_vcd.m1v output_vcd.mpa
```

生成的 MPEG 文件， output_vcd.mpg， 可以通 MPlayer 来播放。 甚至可以直接将个文件刻到 CD-R 介上来建 Video CD， 如果希望做的， 需要安装 multimedia/vcdimager 和 sysutils/cdrdao 个程序。

transcode 提供了机手册， 但仍参考 transcode wiki 以了解更多信息和例子。

8.4.3. 一了解

FreeBSD里不同的件包正迅速展中。 很可能在不久的将来，里所到的都将得到解决。 同，有些人想超越FreeBSD的音/像(A/V)能力， 那他就不得不从一些FAQ和指南里学知，并使用一些不同的用程序。 里就些者指出一些充信息。

MPlayer 文 是很技性的。 些文可以那些希望得于UNIX®高技的人提供参考。 MPlayer 件列表很不喜没耐心文的人， 如果什想告他，首先RTFM。

xine HOWTO 里有一章是于提高性能的，所有的播放器都很。

最后是一些很有前途的程序，者可以一下：

- Avifile，它就是 multimedia/avifile port。
- Ogle 它就是 multimedia/ogle port。
- Xtheater
- multimedia/dvdauthor， 一个制作 DVD 目的源放包。

8.5. 安装

8.5.1. 介

可以在的算机里看到无或有。 多是通RCA或S-video入接收合， 而且有些有广播接收器。

FreeBSD 通bktr(4)程序，提供了基于PCI的的支持， 要求些使用的是Brooktree Bt848/849/878/879 或 Conexant CN-878/Fusion 878a采集芯片。 要保个板上的有被支持的器，参考bktr(4)手册看所支持的器列表。

8.5.2. 加程序

要使用的，就要装bktr(4)程序。 个可以通往 /boot/loader.conf 里添加下一行来。象：

```
bktr_load="YES"
```

外，也可以把个内核， 要是的，就把下几行加到内核配置里去：

```
device   bktr
device  iicbus
device  iicbb
device  smbus
```

□些附加的□□□□程序是必□的，　　因□□的各□成部分是能□一根I2C□□相互□接在一起的。然后建立安装新的内核。

一旦□个支持被加到了□的系□里，□□要重□系□。 在□□□程中，□的□□□□□□示□up(□□)，象□□：

```
bktr0: <BrookTree 848A> mem 0xd7000000-0xd7000fff irq 10 at device 10.0 on pci0
iicbb0: <I2C bit-banging driver> on bti2c0
iicbus0: <Philips I2C bus> on iicbb0 master-only
iicbus1: <Philips I2C bus> on iicbb0 master-only
smbus0: <System Management Bus> on bti2c0
bktr0: Pinnacle/Miro TV, Philips SECAM tuner.
```

当然，□些信息可能因□的硬件不同而有所区□。　　但是□□□能□□那个□制器是否被正□□□到了，可能要忽略一些□□到的同sysctl(8)　MIB（管理系□□）和内核配置文件□□一起的参数。　例如，如果□想□制使用Philips(□利浦) SECAM制式的□□器，□就□把下列行加到内核配置文件里：

```
options OVERRIDE_TUNER=6
```

或者，□直接使用sysctl(8)：

```
# sysctl hw.bt848.tuner=6
```

□参□ bktr(4) 手册和 /usr/src/sys/conf/NOTES 文件， 以了解更多□□□于可用□□的□料。

8.5.3. 有用的□用程序

要使用□的□□□，□需要安装下列□用程序之一：

- multimedia/fxtv 提供 "□口□□(TV-in-a-window)" 功能和□像/声音/□像采集功能。
- multimedia/xawtv 也是一款□□□用程序，功能同 fxtv 一□。
- misc/alevt 解□和□示Videotext/Teletext。
- audio/xmradio， 一款用于一些□□□的□□□台□□器的程序。
- audio/wmtune， 一款用于□台□□器的便捷的□面程序。

更多的程序在FreeBSD Ports Collection(Ports 集)里。

8.5.4. □□解决

如果□的□□□遇到了什□□，　　□□□首先□□一下□的□□采集芯片和□□器是不是真正的被bktr(4)

□□程序支持，并且是不是使用了正□的配置□□。 想得到更多支持和□于□的□□□的各□□□， □可以接触和使用FreeBSD 多媒体□用□件列表 □件列表的□□包。

8.6. □象□描□

8.6.1. 介□

在 FreeBSD 中， □□□描□的能力， 是通□ SANE (Scanner Access Now Easy) API 提供的。 SANE 也会使用一些 FreeBSD □□□□来□□□描□硬件。

FreeBSD 支持 SCSI 和 USB □描□。 在做任何配置之前□□保□的□描□被 SANE 支持。 SANE 有一个 支持的□□列表， 可以□□提供有□□描□的支持情况和状□的信息。 在 FreeBSD 8.X 之前版本的系□中， uscanner(4) 手册□也提供了系□支持的 USB □描□列表。

8.6.2. 内核配置

上面提到 SCSI 和 USB 接口都是支持的。 取决于□的□描□接口， 需要不同的□□□□程序。

8.6.2.1. USB 接口

默□的 GENERIC 内核包含了支持 USB □描□需要的□□□□。 如果□决定使用一个定制的内核， □保下面在□的内核配置文件中存在下面□些行：

```
device usb
device uhci
device ohci
device ehci
```

在 FreeBSD 8.X 之前的版本中， □需要下面□行配置：

```
device uscanner
```

在□些 FreeBSD 版本中， 是通□□□□□程序 uscanner(4) 来提供□ USB □描□的支持的。 从 FreeBSD 8.0 □始，□些支持□直接由 libusb(3) 函数□提供。

使用正□的内核重新引□系□之后， □入 USB □描□。 系□消息□冲区 (使用 dmesg(8) □看) 中会出□下面的信息，表示□□到了□描□：

```
ugen0.2: <EPSON> at usbus0
```

或者， □于 FreeBSD 7.X 系□而言：

```
uscanner0: EPSON EPSON Scanner, rev 1.10/3.02, addr 2
```

随 FreeBSD 版本不同， □些信息表示□描□□□位于□□□点 /dev/ugen0.2 或 /dev/uscanner0。 在□个例子中，

我□使用的是 EPSON Perfection® 1650 USB □描□。

8.6.2.2. SCSI 接口

如果□的□描□是 SCSI 接口的， 重要的是要知道□使用□□ SCSI 控制器。 取决于所使用的 SCSI 芯片， □需要□整内核配置文件。 GENERIC 的内核支持最常用的 SCSI 控制器。 □□□ NOTES 文件并在□的内核配置文件中添加正□的行。 除了 SCSI □配器□□之外， □□需要在内核配置文件中□加下述配置：

```
device scbus
device pass
```

在正□地□□并安装了内核之后， 就□□可以在系□□□□， 从系□消息□冲中看到□些□□：

```
pass2 at aic0 bus 0 target 2 lun 0
pass2: <AGFA SNAPSCAN 600 1.10> Fixed Scanner SCSI-2 device
pass2: 3.300MB/s transfers
```

如果□的□描□没有在系□□□的□候加□， 很可能□需要□制手□□□一下，用 camcontrol(8) 命令□行一次 SCSI □□□描：

```
# camcontrol rescan all
Re-scan of bus 0 was successful
Re-scan of bus 1 was successful
Re-scan of bus 2 was successful
Re-scan of bus 3 was successful
```

然后□描□就会出□在 SCSI □□列表里：

```
# camcontrol devlist
<IBM DDRS-34560 S97B>              at scbus0 target 5 lun 0 (pass0,da0)
<IBM DDRS-34560 S97B>              at scbus0 target 6 lun 0 (pass1,da1)
<AGFA SNAPSCAN 600 1.10>           at scbus1 target 2 lun 0 (pass3)
<PHILIPS CDD3610 CD-R/RW 1.00>     at scbus2 target 0 lun 0 (pass2,cd0)
```

有□ SCSI □□的更多□□， 可□看 scsi(4) 和 camcontrol(8) 手册□。

8.6.3. SANE 配置

SANE 系□分□□部分： 后端 (graphics/sane-backends) 和前端 (graphics/sane-frontends)。后端部分提供到□描□自身的□□。 SANE 的 支持□□列表□□□明了□一个后端可以支持□的□象□描□。 如果□想使用□的□□，就必□□□的□描□□定正□的后端。 前端部分提供□形化的□描界面 (xscanimage)。

要做的第一□就是安装 graphics/sane-backends port 或者 package。然后，使用 sane-find-scanner 命令来□□ SANE 系□做的□描□□□：

```
# sane-find-scanner -q
found SCSI scanner "AGFA SNAPSCAN 600 1.10" at /dev/pass3
```

□出□示了□描□的接口□型和□描□□接到系□上的□□□点。 生□厂家和□品型号可能没有□示，不□不重要。

一些 USB □描□需要□加□固件，后端的手册□中有□方面的解□。 □也□□□□ sane-find-scanner(1) 和 linprocfs(7) 手册□。

□在我□需要□□□描□是否可以被□描前端□□。 默□情况下， SANE 后端自□一个叫做 sane(1) 的命令行工具。□个命令允□□列出□□以及从命令行□行□片□描。 -L □□用来列出□描□□□：

```
# scanimage -L
device `snapscan:/dev/pass3' is a AGFA SNAPSCAN 600 flatbed scanner
```

或者， 如果使用的是 USB 接口 中的 USB □描□：

```
# scanimage -L
device 'epson2:libusb:/dev/usb:/dev/ugen0.2' is a Epson GT-8200 flatbed scanner
```

上述□出来自于 FreeBSD 8.X 系□。 'epson2:libusb:/dev/usb:/dev/ugen0.2' □出了□描□所使用的后台名字 (epson2) 和□□□点 (/dev/ugen0.2)。

如果没有□出任何信息， 或提示没有□□到□描□， □□明 sane(1) 无法□□它。 如果□生□□情况， □就需要修改□描□支持后端的配置文件， 并定□所使用的□描□□。 /usr/local/etc/sane.d/ 目□中包含了所有的后端配置文件。 □□□□□□□常会在某些 USB □描□上□生。

linkend="scanners-kernel-usb"> 中所使用的 USB □描□， `sane-find-scanner` 会□出下面的信息：

例如， □于在 USB 接口， 在 FreeBSD 8.X 中， □描□已□被很好地□□并能□正常工作了；而□于更早版本的 FreeBSD 而言 (使用 uscanner(4) □□程序) sane-find-scanner □会□出□□的信息：

```
# sane-find-scanner -q
found USB scanner (UNKNOWN vendor and product) at device /dev/uscanner0
```

□描□被正□的探□到了，它使用 USB 接口，□接在 /dev/uscanner0 □□□点上。 我□□在可以□□看看□描□是否被正□的□□：

```
# scanimage -L
No scanners were identified. If you were expecting something different,
check that the scanner is plugged in, turned on and detected by the
sane-find-scanner tool (if appropriate). Please read the documentation
which came with this software (README, FAQ, manpages).
```

由于□描□没有□□成功， 我□就需要□□ /usr/local/etc/sane.d/epson2.conf 文件。 所用的□描□型号是 EPSON Perfection® 1650， □□我□知道□描□□使用 epson 后端。□保□□后端配置文件中的□助注□。 改□非常□□：注□掉□致□的□描□使用□□接口的所有行 (在我□□□情况下，我□将注□掉从 scsi □始的所有行，因□我□的□描□使用 USB 接口)，然后在文件的□尾添加指定的接口和所用的□□□点。 □□情况下， 添加下面□行：

```
usb /dev/uscanner0
```

□□保□□后端配置文件提供的注□以及后端手册□了解更多□□， 并使用正□的□法。我□□在可以□□□描□是否被□□到了：

```
# scanimage -L
device `epson:/dev/uscanner0' is a Epson GT-8200 flatbed scanner
```

我□的 USB □描□被□□到了。 此□如果商□和型号与□描□的□□情况不符， 并不会□来太大的麻□。 □需要□注的是 `epson:/dev/uscanner0' 字段， □个□了我□正□地后端名称和正□的□□□点。

一旦 scanimage -L 命令可以看到□描□， 配置就完成了。□□□在准□好等待□描了。

sane(1) 允□我□从命令行□行□片□描， 相比之下使用□形用□界面来□行□片□描会更好。 SANE 提供了一个□□但□用的□形界面： xscanimage (graphics/sane-frontends)。

Xsane (graphics/xsane)是□一个流行的□形□描前端。 □个前端提供了一些高□特性， 比如多□的□描模式(photocopy，fax，等。)， 色彩校正，批量□描，等等。□□个程序都可以作□ GIMP 的□件使用。

8.6.4. 授□其他用□□□□描□

前面所有的操作都是用 root □限来完成的。 然而□可能需要□其他的用□也可以□□□描□。 用□需要有□描□所用的□□□点的□和写□限。 比如，我□的 USB □描□使用□□□点 /dev/ugen0.2 □□上只是到□□□□□点 /dev/usb/0.2.0 的符号□接 (可以通□□看 /dev 目□的内容来□□□一点)。 □□□点本身和□个符号□接分□属于 wheel 和 operator □。 将用□ joe 添加到 □些□中， 就可以允□他使用□描□了， 不□， 出于□而易□的安全方面的原因， 在将用□加到特定的用□□， 特□是 wheel □□， 无疑需三思而后行。 更好的解决方法是□建一个□□用于□□ USB □□的□， 并□□个□的成□能□□□ USB □□。

□里作□示例， 我□将会使用名□ usb 的□。 第一□是借助 pw(8) 命令来□建它：

```
# pw groupadd usb
```

接下来， 令 /dev/ugen0.2 符号□接和 /dev/usb/0.2.0 □□□点能□以 usb □的身□来□□， 具体而言是配置正□的写□限 (0660 或 0664)， 因□默□情况下只有属主 (root) 才能写□些□□。 □些配置是通□在 /etc/devfs.rules 文件中添加如下的□置来□□的：

```
[system=5]
add path ugen0.2 mode 0660 group usb
add path usb/0.2.0 mode 0666 group usb
```

FreeBSD 7.X 用□需要将上面的配置改□使用与之□□的 /dev/uscanner0：

```
[system=5]
add path uscanner0 mode 660 group usb
```

随后□□需要在 /etc/rc.conf 中添加下面的内容并重新□□：

```
devfs_system_ruleset="system"
```

□于□些配置的□一□□□□参考□机手册 devfs(8)。

□在， 只需将用□添加到 usb □， 就可以使用□描□了：

```
# pw groupmod usb -m joe
```

更多□情， □参□□机手册 pw(8)。

Chapter 9. 配置FreeBSD的内核

9.1. 概述

内核是 FreeBSD 操作系□的核心。 它□□管理内存、 □行安全控制、 网□、 磁□□□等等。 尽管 FreeBSD 可以□□修改的□在已□越来越多， 但有□□□是需要重新配置和□□□的内核。

□完□章，□将了解：

- □什□需要建立定制的内核。
- 如何□写内核配置文件，或修改已存在的配置文件。
- 如何使用内核配置文件□建和□□新的内核。
- 如何安装新内核。
- 如何□理出□的□□。

□一章□出的命令□□以 `root` 身□□行， 否□可能会不成功。

9.2. □什□需要建立定制的内核?

□去， FreeBSD 采用的是被人□称作 "□片式" 的内核。 □□内核本身是一个大的程序， 它支持的□□不能□□□地加以改□， 而当希望改□内核的行□□， 就必□□□一个新的内核， 并重新□□□算机才可以使用它。

如今， FreeBSD 正在迅速地□移到一□新的模型， 其特点是将大量内核功能放□可以□□加□和卸□的内核模□来提供。 □使得内核能□□□硬件的□整 (例如□□本□算机中的 PCMCIA □)， 以及□内核引入新的功能，而无需在□□内核□就将其添加□去。 □□做法称□模□化内核。

尽管如此， 仍然有一些功能需要静□地□□□内核。 有□， □是由于□些功能与内核的□合非常□密而无法□□□□加□， □有一些情况是□□没有人将□些功能改写□可□□加□的模□。

□□定制的内核是成□高□ BSD 用□所必□□□的一□。 尽管□一□程需要花□一些□□， 但它能□□□的 FreeBSD 系□□来一些好□。 与必□支持大量硬件的 GENERIC 内核不同， 定制的内核可以只包含□于 □ PC 硬件的支持。□□做有很多好□， 例如：

- 更快地□□。 因□内核只需要□□□系□上的硬件， □□□所花□的□□将大大□短。
- 使用更少的内存。 由于可以□去不需要的功能和□□□□， 通常定制的内核会比 GENERIC 使用的内存更少。 □省内核使用的内存之所以重要是因□内核必□常□于物理内存中， 从而使□用程序能□用到更多的内存。 正因□□□， □ RAM □小的系□来□定制内核就更□重要了。
- 支持更多的硬件。 定制的内核允□□□加□似声□□□的 GENERIC 内核没有提供内建支持的硬件。

9.3. □□系□硬件

在□□配置内核以前，比□明智的做法是先□得一□机器硬件的清□。 当 FreeBSD 并不是主操作系□□，通□□看当前操作系□的配置可以很容易的 □建一□机器硬件的配置清□。□例来□， Microsoft® 的 □□管理器里通常含有□于已安装硬件的重要信息。 □□管理器 位于控制面板。

某些版本的 Microsoft® Windows® 有一个 系□ □□会指明 □□管理器 的位置。

如果机器上并不存在其他的操作系□， 系□管理□只能手□□□□些信息了。其中的一个方法是使用 dmesg(8) 工具以及 man(1) 命令。FreeBSD 上大多数的□□程序都有一□手册□（manual page）列出了所支持的硬件，在系□□□的□候，被□□的硬件也会被列出。□例来□， 下面的□几行表示 psm □□□到了一个鼠□：

```
psm0: <PS/2 Mouse> irq 12 on atkbdc0
psm0: [GIANT-LOCKED]
psm0: [ITHREAD]
psm0: model Generic PS/2 mouse, device ID 0
```

□个□□需要被包含在客□制定的内核配置文件里， 或着使用 loader.conf(5) 加□。

有□，dmesg 里只会□示来自系□消息的数据， 而不是系□□□□的□□信息。在□□的情况下，□可以□看文件 /var/run/dmesg.boot。

□一个□□硬件信息的方法是使用 pciconf(8) 工具， 它能提供更□□的□出，比如：

```
ath0@pci0:3:0:0:        class=0x020000 card=0x058a1014 chip=0x1014168c rev=0x01
hdr=0x00
    vendor     = 'Atheros Communications Inc.'
    device     = 'AR5212 Atheros AR5212 802.11abg wireless'
    class      = network
    subclass   = ethernet
```

□个片断取自于 pciconf -lv 命令的□出，□示 ath □□□到了一个无□以太网□□。□入命令 man ath 就能□□有□ ath(4) 的手册□（manual page）了。

□可以□□ man(1) 命令 -k □□， 同□能□得有用的信息。例如：

```
# man -k Atheros
```

能得到一□包含特定□□的手册□（manual page）：

```
ath(4)                  - Atheros IEEE 802.11 wireless network driver
ath_hal(4)              - Atheros Hardware Access Layer (HAL)
```

手□□有一□硬件的配置清□， 那□□□制定内核的□程就□得不那□困□了。

9.4. 内核□□，子系□和模□

在□□一个制定的内核之前□三思一下□□做的理由， 如果□是需要某个特定的硬件支持的□， 那□很可能已□存在一个□成的模□了。

内核模□存放在目□ /boot/kernel 中，并能由 kldload(8) 命令加□入正在□行的内核。 基本上所有的内核

□□都有特定的模□和手册□。比如， 下面提到的 ath 无□以太网□□。 在□个□□的□机手册中有以下信息：

```
Alternatively, to load the driver as a module at boot time, place the
following line in man:loader.conf[5]:

    if_ath_load="YES"
```

遵照示例，在 /boot/loader.conf 中加入 `if_ath_load="YES"` □能在机器□□的□候□□加□□个模□。

某些情况下，□没有相□的模□。通常是一些子系□和非常重要的□□， 比如，快速文件系□ (FFS) 就是一个内核必需的□□。 同□的□有网□支持 (INET)。不幸的是， 分辨一个□□是否必需的唯一方法就是□□□□以下那个模□本身。

去除某个□□的支持或某个□□会非常容易得到一个坏掉的内核。 □例来□，如果把 ata(4) □□从内核配置文件中去掉， 那□一个使用 ATA 磁□□□的系□可能就□得无法引□，除非有在 loader.conf 中加□。当□无法□定的□候， □□□一下那个模□并把它留在□的内核配置中。

9.5. 建立并安装一个定制的内核

首先□内核□建目□做一个快速的□□。 □里所提到的所有目□都在 /usr/src/sys 目□中； 也可以通□ /sys 来□□它。□里的□多子目□包含了内核的不同部分， 但□我□所要完成的任□最重要的目□是 arch/conf， □将在□里□□定制的内核配置； 以及 compile， □□□程中的文件将放置在□里。 *arch* 表示 i386、 amd64、 ia64、 powerpc、 sparc64， 或 pc98 (在日本比□流行的□一□ PC 硬件□□分支)。 在特定硬件架□目□中的文件只和特定的硬件有□； 而其余代□□是与机器无□的， □所有已□或将要移植并□行 FreeBSD 的平台上都共享□些代□。 文件目□是按照□□□□的， 所支持的硬件□□、 文件系□， 以及可□的□件通常都在它□自己的目□中。

□一章提供的例子假定□使用 i386 架□的□算机。 如果□的情况不是□□， 只需□目□名作相□的□整即可。

如果□的系□中 没有/usr/src/sys □□一个目□， □□明没有安装内核源代□。 安装它最□□的方法是通□以 root 身□□行 `sysinstall`， □□ Configure， 然后是 Distributions、 src， □中其中的 base 和 sys。 如果□不喜□ sysinstall 并且有一□ "官方的" FreeBSD CDROM， 也可以使用下列命令， 从命令行来安装源代□：

```
# mount /cdrom
# mkdir -p /usr/src/sys
# ln -s /usr/src/sys /sys
# cat /cdrom/src/ssys.[a-d]* | tar -xzvf -
# cat /cdrom/src/sbase.[a-d]* | tar -xzvf -
```

接下来， □入 arch/conf 目□下面， □制 GENERIC 配置文件， 并□□个文件起一个容易辨□的名称， 它就是□的内核名称。例如：

```
# cd /usr/src/sys/i386/conf
# cp GENERIC MYKERNEL
```

通常，□个名称是大写的，如果□正□□着多台不同硬件的FreeBSD机器，以□机器的域名来命名是非常好的主意。我□把它命名□ MYKERNEL就是□个原因。

将□的内核配置文件直接保存在 /usr/src 可能不是一个好主意。 如果□遇到□□， □掉 /usr/src 并重新□始很可能是一个□人的□□。 一旦□始做□件事， □可能几秒□之后才会意□到□同□会□除定制的内核配置文件。 □外， 也不要直接□□ GENERIC， 因□下次□ 更新代□□它会被覆□， 而□的修改也就随之□失了。

□也可以考□把内核配置文件放到□的地方， 然后再到 i386 目□中□建一个指向它的符号□接。

例如：

```
# cd /usr/src/sys/i386/conf
# mkdir /root/kernels
# cp GENERIC /root/kernels/MYKERNEL
# ln -s /root/kernels/MYKERNEL
```

必□以 root 身□□行□些和接下来命令， 否□就会得到 的□□提示。

□在就可以用□喜□的文本□□器来□□ MYKERNEL 了。 如果□□□□始使用 FreeBSD， 唯一可用的□□器很可能是 vi， 它的使用比□□□， 限于篇幅， □里不予介□， □可以在 参考□目 一章中□到很多相□□籍。 不□， FreeBSD 也提供了一个更好用的□□器， 它叫做 ee， □于新手来□， □很可能是一个不□的□□。 □可以修改配置文件中的注□以反映□的配置， 或其他与 GENERIC 不同的地方。

如果□在SunOS™或者其他BSD系□下定制□内核，那□个文件中的□大部分将□□非常熟悉。 如果□使用的是□如DOS□□的系□，那GENERIC配置文件看起来就非常困□， 所以在下面的 配置文件章□将慢慢地、仔□地□行介□。

如果□和 FreeBSD project □行了 代□同□， □一定要在□行任何更新之前□看 /usr/src/UPDATING。 □个文件中描述了更新□的代□中出□的重大□□或需要注意的地方。 /usr/src/UPDATING □是和□的 FreeBSD 源代□□□， 因此能□提供比手册更具□效性的新内容。

□在□□□□内核的源代□了。

Procedure: □□内核

1. □入 /usr/src 目□：

```
# cd /usr/src
```

2. □□内核：

```
# make buildkernel KERNCONF=MYKERNEL
```

3. 安装新内核：

```
# make installkernel KERNCONF=MYKERNEL
```

使用□□方法□□内核□， 需要安装完整的 FreeBSD 源代□。

默□情况下， 在□□□所定制的内核□， 全部 内核模□也会同□参与□建。 如果□希望更快地升□内核， 或者只希望□□□所需要的模□， □□在□□之前□□ /etc/make.conf：

```
MODULES_OVERRIDE = linux acpi sound/sound sound/driver/ds1 ntfs
```

□个□量的内容是所希望□建的模□列表。

```
WITHOUT_MODULES = linux acpi sound ntfs
```

□个□量的内容是将不在□□□程中□□的□□模□列表。 如果希望了解更多与□建内核有□的□量，□参□ make.conf(5) □机手册。

新内核将会被□制到 /boot/kernel 目□中成□ /boot/kernel/kernel 而旧的□被移到 /boot/kernel.old/kernel。□在□□系□， 然后用新的内核□□□算机。 如果出□□□， 后面的一些 故障排除方法 将□□□脱困境。 如果□的内核无法□□， □参考那一□。

其他与□□□程相□的文件， 如 loader(8) 及其配置， □放在 /boot。 第三方或定制的模□也可以放在 /boot/kernel， 不□□□注意保持模□和内核的同□□很重要的， 否□会□致不□定和□□。

9.6. 配置文件

配置文件的格式是非常□□的。 □一行都包括一个□□□， 以及一个或多个参数。 □□上，□大多数行都只包括一个参数。 在 \# 之后的内容会被□□是注□而忽略掉。 接下来几□，将以 GENERIC 中的□序介□所有□□字。 如果需要与平台有□的□□和□□的□□列表， □参考与 GENERIC 文件在同一个目□中的那个

NOTES， 而平台无□的□□， □可以在 /usr/src/sys/conf/NOTES □到。

配置文件中□可以使用 include □句。 □个□句能□在内核配置文件中直接引用其他配置文件的内容， 使得□能□使用□小的、 □包含相□于□存配置的□□而□少□□所需的工作。 例如， 如果□只需□ GENERIC 内核□行少量定制， 在其中添加几个□□程序和附加□□， □只要□□相□于 GENERIC 的□化就可以了：

```
include GENERIC
ident MYKERNEL

options         IPFIREWALL
options         DUMMYNET
options         IPFIREWALL_DEFAULT_TO_ACCEPT
options         IPDIVERT
```

□多系□管理□会□□， □□方法与先前从□□始写配置文件的方法相比， 可以□□相当多的好□：本地采用的配置文件只表□与 GENERIC 内核的差□， □□， 在升□的□候往往就不需要做任何改□， 而新加入 GENERIC 的功能就会自□加入到本地的内核， 除非使用 nooptions 或 nodevice □句将其排除。□一章余下的部分将着重介□典型的配置文件， 以及内核□□和□□的作用。

如果□需要一□包含所有□□的文件， 例如用于□□目的， □□以 root 身□□行下列命令：

```
# cd /usr/src/sys/i386/conf && make LINT
```

下面是一个 GENERIC 内核配置文件的例子， 它包括了一些需要解□的注□。 □个例子□□和□□制的 /usr/src/sys/i386/conf/GENERIC 非常接近。

```
machine     i386
```

□是机器的架□， 它只能是 amd64, i386, ia64, pc98, powerpc, 或 sparc64 中的一□。

```
cpu          I486_CPU
cpu          I586_CPU
cpu          I686_CPU
```

上面的□□指定了□系□中所使用的 CPU □型。 □可以使用多个 CPU □型 (例如， □不□定是□□指定 I586_CPU 或 I686_CPU)。 然而□于定制的内核， 最好能□只指定□使用的那□ CPU。 如果□□于自己使用的 CPU □型没有把握， 可以通□□看 /var/run/dmesg.boot 中的□□信息来了解。

```
ident          GENERIC
```

□是内核的名字。 □□□取一个自己的名字， 例如取名叫 MYKERNEL， 如果□一直在按照前面的□明做的□。 □放在 ident 后面的字符串在□□内核□会□示出来， 因此如果希望能□容易区分常用的内核和□□定制的内核， 就□□采取不同的名字 (例如， □想定制一个□□性的内核)。

```
#To statically compile in device wiring instead of /boot/device.hints
#hints          "GENERIC.hints"         # Default places to look for devices.
```

device.hints(5) 可以用来配置□□□□□□。 在□□的□候 loader(8) 将会□□缺省位置 /boot/devicehints。 使用 hints □□□就可以把□些 hints 静□□□□内核。 □□就没有必要在 /boot下□建devicehints。

```
makeoptions     DEBUG=-g          # Build kernel with gdb(1) debug symbols
```

一般的 FreeBSD □□□程， 在所□□的内核指定了 -g □□□， 由于此□□将□□□ gcc(1) 表示加入□□信息，因此会将□□符号也包含□来。

```
options         SCHED_ULE         # ULE scheduler
```

□是 FreeBSD 上使用的默□系□□度器。 □保留此□□。

```
options         PREEMPTION         # Enable kernel thread preemption
```

允□内核□程根据□先□的□占□度。 □有助于改善交互性， 并可以□中断□程更早地□行， 而无□等待。

```
options         INET              # InterNETworking
```

网□支持，即使□不打算□网，也□保留它，大部分的程序至少需要回□网□（就是和本机□行网□□接），所以□烈要求保留它。

```
options         INET6             # IPv6 communications protocols
```

□将打□IPv6□接□□。

```
options         FFS               # Berkeley Fast Filesystem
```

□是最基本的硬□文件系□，如果打算从本地硬□□□，□保留它。

```
options         SOFTUPDATES       # Enable FFS Soft Updates support
```

□个□□会□用内核中的 Soft Updates 支持， 它会□著地提高磁□的写入速度。 尽管□□功能是由内核直接提供的， 但仍然需要在□个磁□上□用它。 □□□ mount(8) 的□出， 以了解□系□中的磁□上是否已□□用了 Soft Updates。 如果没有看到 soft-updates □□， □需要使用 tunefs(8) (□于□存系□) 或 newfs(8) (□于新系□) 命令来激活它。

```
options         UFS_ACL           # Support for access control lists
```

□个□□将□用内核中的□□控制表的支持。 □依□于□展属性以及 UFS2， 以及在 文件系□□□控制表 中所介□的那些特性。 ACL 默□是□用的， 并且如果已□在文件系□上使用了□一特性， 就不□再□掉它， 因□□会去掉文件的□□控制表， 并以不可□期的方式改□受保□的文件的□□方式。

```
options         UFS_DIRHASH       # Improve performance on big directories
```

通□使用□外的内存，□个□□可以加速在大目□上的磁□操作。 □□□在大型服□器和□繁使用的工作站上打□□个□□，而在磁□操作不是很重要的 小型系□上□□它，比如防火□。

```
options         MD_ROOT           # MD is a potential root device
```

□个□□将打□以基于内存的虚□磁□作□根□□的支持。

```
options         NFSCLIENT         # Network Filesystem Client
options         NFSSERVER         # Network Filesystem Server
options         NFS_ROOT          # NFS usable as /, requires NFSCLIENT
```

网□文件系□。 如果□不打算通□ TCP/IP 挂接 UNIX® 文件服□器的分区， 就可以注□掉它。

```
options         MSDOSFS           # MSDOS Filesystem
```

MS-DOS® 文件系□。 只要□不打算在□□□挂接由 DOS 格式化的硬□分区， 就可以把它注□掉。 如前面所介□的那□， 在□第一次挂接 DOS 分区□， 内核会自□加□需要的模□。 此外， emulators/mtools □件提供了一个很方便的功能， 通□它□可以直接□□ DOS □□而无需挂接或卸下它□ (而且也完全不需要 MSDOSFS)。

```
options         CD9660            # ISO 9660 Filesystem
```

用于 CDROM 的 ISO 9660 文件系□。 如果没有 CDROM □□器或很少挂接光□数据 (因□在首次使用数据 CD □会自□加□)， 就可以把它注□掉。 音□ CD 并不需要□个□□。

```
options         PROCFS            # Process filesystem (requires PSEUDOFS)
```

□程文件系□。 □是一个挂接在 /proc 的一个 "假扮的" 文件系□， 其作用是允□□似 ps(1) □□的程序□出正在□行的□程的□一□信息。 多数情况下， 并不需要使用 PROCFS， 因□□大多数□□和□控工具， 已□□行了一系列修改， 使之不再依□ PROCFS： 默□安装的系□中并不会挂接□一文件系□。

```
options         PSEUDOFS          # Pseudo-filesystem framework
```

如果希望使用 PROCFS， 就必□加入 PSEUDOFS 的支持。

```
options         GEOM_GPT          # GUID Partition Tables.
```

□个□□提供了在磁□上使用大量的分区的能力。

```
options         COMPAT_43         # Compatible with BSD 4.3 [KEEP THIS!]
```

使系□兼容4.3BSD。不要去掉□一行，不然有些程序将无法正常□行。

```
options         COMPAT_FREEBSD4   # Compatible with FreeBSD4
```

如果希望支持在旧版 FreeBSD 上□□的使用旧式接口的□用程序， 就需要加入□一□□。 一般来□， 推□在所有的 i386™ 系□上□用□个□□， 因□□免可能会用到一些旧的□用； 到 5.X 才□始支持的平台， 如 ia64 和 sparc64，□不需要□个□□。

```
options         COMPAT_FREEBSD5   # Compatible with FreeBSD5
```

如果希望支持在 FreeBSD 5.X 版本上□□， 且使用 FreeBSD 5.X 系□□用接口的□用程序， □□加上□个□□。

```
options         COMPAT_FREEBSD6   # Compatible with FreeBSD6
```

如果希望支持在 FreeBSD 6.X 版本上□□， 且使用 FreeBSD 6.X 系□□用接口的□用程序， □□加上□个□□。

```
options         COMPAT_FREEBSD7   # Compatible with FreeBSD7
```

如果希望支持在 FreeBSD 8 以上版本的操作系□中□行在 FreeBSD 7.X 版本上□□， 且使用 FreeBSD 7.X 系□□用接口的□用程序， □□加上□个□□。

```
options         SCSI_DELAY=5000  # Delay (in ms) before probing SCSI
```

□将□内核在探□□个 SCSI □□之前等待 5 秒。 如果□只有 IDE 硬□□□器， 就可以不管它， 反之□可能会希望□□降低□个数□以加速□□□程。 当然， 如果□□□做之后 FreeBSD 在□□□的 SCSI □□□遇到□□，□□□需要再把它改回去。

```
options         KTRACE            # ktrace(1) support
```

□个□□打□内核□程跟踪，在□□□很有用。

```
options         SYSVSHM           # SYSV-style shared memory
```

提供System V共享内存(SHM)的支持，最常用到SHM的□□是X Window的XSHM延伸， 不少□□相□

程序会自□使用SHM来提供□外的速度。如果□要使用X Window，□最好加入□个□□。

```
options         SYSVMSG         # SYSV-style message queues
```

支持 System V 消息。 □只会在内核中□加数百字□的空□占用。

```
options         SYSVSEM         # SYSV-style semaphores
```

支持System V 信号量， 不常用到， 但只在kernel中占用几百个字□的空□。

ipcs(1) 命令的 -p □□可以□示出任何用到□些 System V 机制的□程。

```
options         _KPOSIX_PRIORITY_SCHEDULING # POSIX P1003_1B real-time extensions
```

在 1993 年 POSIX® 添加的□□□展。 在 Ports Collection 中某些□用程序会用到□些 （比如StarOffice™）。

```
options         KBD_INSTALL_CDEV  # install a CDEV entry in /dev
```

□个□□是在 /dev下建立□□□□□点必需的。

```
options         ADAPTIVE_GIANT    # Giant mutex is adaptive.
```

内核全局□ (Giant) 是一□互斥机制 (休眠互斥体) 的名字， 它用于保□□多内核□源。 □在， □已□成□了一□无法接受的性能瓶□， 它已□被越来越多地使用保□□个□源的□代替。 ADAPTIVE_GIANT □□将使得内核全局□作□一□自□□自旋□。 □意味着， 当有□程希望□住内核全局□互斥体， 但互斥体已□被□一个 CPU 上的□程□住的□候， 它将□□□行， 直到那个□程□放□□止。 一般情况下， □一个□程将□入休眠状□并等待下一次□度。 如果□不□定是否□□□□做的□， 一般□□打□它。

□注意在 FreeBSD 8.0-RELEASE 及以后的版本，所有的互斥体默□都是自□□的，除非在□□□使用 NO_ADAPTIVE_MUTEXES □□， 明□的指定□非自□□。因此，内核全局□（Giant）目前默□也是自□□的, 而且 ADAPTIVE_GIANT □□已□从内核配置文件中移出。

```
device          apic            # I/O APIC
```

apic □□将□用使用 I/O APIC 作□中断□送□□的能力。 apic □□可以被 UP 和 SMP 内核使用， 但 SMP 内核必□使用它。 要支持多□理器， □需要加上 options SMP。

只有在 i386 和 amd64 平台上才存在 apic □□， 在其他硬件平台上不□使用它。

```
device          eisa
```

如果□的主机板上有EISA□□，加入□个□置。使用□个□□可以自□□描并□置所有□接在EISA□□上的□□。

```
device          pci
```

如果□的主板有PCI□□，就加入□个□□。使用□个□□可以自□□描PCI□，并在PCI到ISA之□建立通路。

```
# Floppy drives
device          fdc
```

□是□□控制器。

```
# ATA and ATAPI devices
device          ata
```

□个□□器支持所有ATA和ATAPI□□。□只要在内核中加入device ata□□，就可以□内核支持□代□算机上的所有PCI ATA/ATAPI□□。

```
device          atadisk                 # ATA disk drives
```

□个是使用 ATAPI 硬□□□器□必□加入的□□。

```
device          ataraid                 # ATA RAID drives
```

□个□□需要 device ata， 它用于 ATA RAID □□。

```
device          atapicd                 # ATAPI CDROM drives
```

□个是ATAPI CDROM□□器所必□的。

```
device          atapifd                 # ATAPI floppy drives
```

□个是ATAPI □□□□器所必□的。

```
device          atapist                 # ATAPI tape drives
```

□个是ATAPI 磁□机□□器所必□的.

```
options         ATA_STATIC_ID           # Static device numbering
```

□指定□控制器使用其静□的□号； 如果没有□个□□， □会□□地分配□□的□号。

```
# SCSI Controllers
device          ahb        # EISA AHA1742 family
device          ahc        # AHA2940 and onboard AIC7xxx devices
options         AHC_REG_PRETTY_PRINT    # Print register bitfields in debug
                                        # output.  Adds ~128k to driver.
device          ahd        # AHA39320/29320 and onboard AIC79xx devices
options         AHD_REG_PRETTY_PRINT    # Print register bitfields in debug
                                        # output.  Adds ~215k to driver.
device          amd        # AMD 53C974 (Teckram DC-390(T))
device          isp        # Qlogic family
#device         ispfw      # Firmware for QLogic HBAs- normally a module
device          mpt        # LSI-Logic MPT-Fusion
#device         ncr        # NCR/Symbios Logic
device          sym        # NCR/Symbios Logic (newer chipsets + those of `ncr')
device          trm        # Tekram DC395U/UW/F DC315U adapters

device          adv        # Advansys SCSI adapters
device          adw        # Advansys wide SCSI adapters
device          aha        # Adaptec 154x SCSI adapters
device          aic        # Adaptec 15[012]x SCSI adapters, AIC-6[23]60.
device          bt         # Buslogic/Mylex MultiMaster SCSI adapters

device          ncv        # NCR 53C500
device          nsp        # Workbit Ninja SCSI-3
device          stg        # TMC 18C30/18C50
```

SCSI控制器。可以注□掉□系□中没有的□□。 如果□只有IDE□□，□可以把□些一起□掉。 *_REG_PRETTY_PRINT □□的配置， □是□□□□程序的□□□□。

```
# SCSI peripherals
device          scbus      # SCSI bus (required for SCSI)
device          ch         # SCSI media changers
device          da         # Direct Access (disks)
device          sa         # Sequential Access (tape etc)
device          cd         # CD
device          pass       # Passthrough device (direct SCSI access)
device          ses        # SCSI Environmental Services (and SAF-TE)
```

SSCSI外□□□。也可以像上面一□操作。

目前系□提供的 USB umass(4) 以及少量其它□□使用了 SCSI 子系□， 尽管它□并不是真的 SCSI □□。 因此， 如果在内核配置使用了□□□□程序， □□必不要□除 SCSI 支持。

```
# RAID controllers interfaced to the SCSI subsystem
device          amr         # AMI MegaRAID
device          arcmsr      # Areca SATA II RAID
device          asr         # DPT SmartRAID V, VI and Adaptec SCSI RAID
device          ciss        # Compaq Smart RAID 5*
device          dpt         # DPT Smartcache III, IV - See NOTES for options
device          hptmv       # Highpoint RocketRAID 182x
device          rr232x      # Highpoint RocketRAID 232x
device          iir         # Intel Integrated RAID
device          ips         # IBM (Adaptec) ServeRAID
device          mly         # Mylex AcceleRAID/eXtremeRAID
device          twa         # 3ware 9000 series PATA/SATA RAID

# RAID controllers
device          aac         # Adaptec FSA RAID
device          aacp        # SCSI passthrough for aac (requires CAM)
device          ida         # Compaq Smart RAID
device          mfi         # LSI MegaRAID SAS
device          mlx         # Mylex DAC960 family
device          pst         # Promise Supertrak SX6000
device          twe         # 3ware ATA RAID
```

支持RAID控制器。如果□没有□些，可以把它□注□掉或是□掉。

```
# atkbdc0 controls both the keyboard and the PS/2 mouse
device          atkbdc      # AT keyboard controller
```

□□控制器（atkbdc）提供AT□□□入以及PS/2指□□□的I/O服□。 □□□□程序（atkbd）与PS/2鼠□□□程序（psm）需要□个控制器，所以不要□除它。

```
device          atkbd       # AT keyboard
```

atkbd□□程序，与atkbdc控制器一起使用， 提供□接到AT□□控制器的AT 84□□与AT加□型□□的□□服□。

```
device          psm         # PS/2 mouse
```

如果□的鼠□□接到PS/2鼠□端口，就使用□个□□□□程序。

```
device          kbdmux         # keyboard multiplexer
```

□□□□多路□□器的基本支持。 如果□不打算使用多个□□， □可以放心地□除□一行。

```
device          vga         # VGA video card driver
```

□□□□。

```
device          splash     # Splash screen and screen saver support
```

□□□的 splash 画面！ 屏幕保□程序也需要□一□□。

```
# syscons is the default console driver, resembling an SCO console
device          sc
```

sc 是默□的控制台□□程序， □似 SCO 控制台。 由于□大部分全屏幕程序都通□□似 termcap □□的□端数据□函数□□□□控制台， 因此无□□使用□个或与 VT220 兼容的 vt 都没有什□□系。 如果□在□行□□控制台□使用全屏幕程序□□生□□， □在登□之后将 TERM □量□置□ scoansi。

```
# Enable this for the pcvt (VT220 compatible) console driver
#device          vt
#options         XSERVER          # support for X server on a vt console
#options         FAT_CURSOR       # start with block cursor
```

□是一个兼容 VT220 的控制台□□， 它同□能□向下兼容 VT100/102。 在同 sc 硬件不兼容的一些□□本上它能□□行的很好。 当然， 登□系□□□把 TERM □量□置□ vt100 或 vt220。 此□□在□接网□上大量不同的机器□也被□明非常有用， 因□此□ termcap 或 terminfo 通常没有可用的 sc □□ - 而 vt100 □几乎□□平台都支持。

```
device          agp
```

如果□的机器使用 AGP □， □把上面一行加入配置。 □将□用 AGP， 以及某些□上的 AGP GART 支持。

```
# □源管理支持（参□ NOTES 了解更多□□）
#device          apm
```

高□□源管理支持。□□□本有用，不□在 GENERIC 里默□禁用。

```
# □加 i8254 的 挂起/恢□ 支持。
device          pmtimer
```

用于□源管理事件， 例如 APM 和 ACPI 的□□□□□□。

```
# PCCARD (PCMCIA) support
# PCMCIA and cardbus bridge support
device          cbb             # cardbus (yenta) bridge
device          pccard          # PC Card (16-bit) bus
device          cardbus         # CardBus (32-bit) bus
```

PCMCIA支持。如果□使用膝上型□算机，□需要□个。

```
# Serial (COM) ports
device          sio             # 8250, 16[45]50 based serial ports
```

□些串口在 MS-DOS®/Windows® 的世界中称□ COM 口。

如果使用内置式的□制解□器， 并占用 COM4 而□□有一个串口在 COM2， □必□把□制解□器的 IRQ 改□ 2 (由于晦□的技□原因， IRQ2 = IRQ 9) 才能□在 FreeBSD 中□□它。如果有多口的串口□， □参考 sio(4) 以了解需要在 /boot/device.hints 中□行的□置。某些□□ (特□是基于 S3 芯片的□) 使用形如 0x*2e8 的 IO 地址， 而□多廉价的串口□不能□正□地□ 16-位 IO 地址空□□行解□， 因此它□会□生冲突， 并造成 COM4 □□上无法使用。

□一个串口都需要有一个唯一的 IRQ (除非□使用支持中断分享的串口□)， 因此默□的 COM3 和 COM4 IRQ 是不能使用的。

```
# Parallel port
device          ppc
```

ISA-bus并行接口。

```
device          ppbus       # Parallel port bus (required)
```

提供并行□□的支持。

```
device          lpt         # Printer
```

提供并口打印机的支持。

要使用并口打印机，就必□同□加入上面三行□置。

```
device          plip        # TCP/IP over parallel
```

□是□□并行网□接口的□□器。

```
device          ppi         # Parallel port interface device
```

普通用途的I/O ("geek port") + IEEE1284 I/O.

```
#device         vpo         # Requires scbus and da
```

□是□□Iomega Zip□□器的。它要求scbus和da的支持。 最好的□行效果是工作在EPP 1.9模式。

```
#device         puc
```

如果□有由 puc(4) 支持的 "□" 串行或并行 PCI □, □□去掉□一行的注□。

```
# PCI Ethernet NICs.
device          de          # DEC/Intel DC21x4x (Tulip)
device          em          # Intel PRO/1000 adapter Gigabit Ethernet Card
device          ixgb        # Intel PRO/10GbE Ethernet Card
device          txp         # 3Com 3cR990 (Typhoon)
device          vx          # 3Com 3c590, 3c595 (Vortex)
```

多□PCI网□□□器。注□或□除□系□中没有的□□.

```
# PCI Ethernet NICs that use the common MII bus controller code.
# NOTE: Be sure to keep the 'device miibus' line in order to use these NICs!
device          miibus      # MII bus support
```

MII□□支持□于一些PCI 10/100 Ethernet NIC来□是必需的。

```
device          bce         # Broadcom BCM5706/BCM5708 Gigabit Ethernet
device          bfe         # Broadcom BCM440x 10/100 Ethernet
device          bge         # Broadcom BCM570xx Gigabit Ethernet
device          dc          # DEC/Intel 21143 and various workalikes
device          fxp         # Intel EtherExpress PRO/100B (82557, 82558)
device          lge         # Level 1 LXT1001 gigabit ethernet
device          msk         # Marvell/SysKonnect Yukon II Gigabit Ethernet
device          nge         # NatSemi DP83820 gigabit ethernet
device          nve         # nVidia nForce MCP on-board Ethernet Networking
device          pcn         # AMD Am79C97x PCI 10/100 (precedence over 'lnc')
device          re          # RealTek 8139C+/8169/8169S/8110S
device          rl          # RealTek 8129/8139
device          sf          # Adaptec AIC-6915 (Starfire)
device          sis         # Silicon Integrated Systems SiS 900/SiS 7016
device          sk          # SysKonnect SK-984x & SK-982x gigabit Ethernet
device          ste         # Sundance ST201 (D-Link DFE-550TX)
device          stge        # Sundance/Tamarack TC9021 gigabit Ethernet
device          ti          # Alteon Networks Tigon I/II gigabit Ethernet
device          tl          # Texas Instruments ThunderLAN
device          tx          # SMC EtherPower II (83c170 EPIC)
device          vge         # VIA VT612x gigabit ethernet
device          vr          # VIA Rhine, Rhine II
device          wb          # Winbond W89C840F
device          xl          # 3Com 3c90x (Boomerang, Cyclone)
```

使用MII□□控制器代□的□□器。

```
# ISA Ethernet NICs.  pccard NICs included.
device          cs         # Crystal Semiconductor CS89x0 NIC
# 'device ed' requires 'device miibus'
device          ed         # NE[12]000, SMC Ultra, 3c503, DS8390 cards
device          ex         # Intel EtherExpress Pro/10 and Pro/10+
device          ep         # Etherlink III based cards
device          fe         # Fujitsu MB8696x based cards
device          ie         # EtherExpress 8/16, 3C507, StarLAN 10 etc.
device          lnc        # NE2100, NE32-VL Lance Ethernet cards
device          sn         # SMC's 9000 series of Ethernet chips
device          xe         # Xircom pccard Ethernet

# ISA devices that use the old ISA shims
#device         le
```

ISA 以太网□□□。 参□ /usr/src/sys/i386/conf/NOTES 以了解□于□个□□程序能□□□□的网□的□□。

```
# Wireless NIC cards
device          wlan            # 802.11 support
```

通用 802.11 支持。 □行配置是无□网□所必需的。

```
device          wlan_wep        # 802.11 WEP support
device          wlan_ccmp       # 802.11 CCMP support
device          wlan_tkip       # 802.11 TKIP support
```

□□ 802.11 □□的加密支持。 如果希望使用加密和 802.11i 安全□□， 就需要□些配置行。

```
device          an         # Aironet 4500/4800 802.11 wireless NICs.
device          ath             # Atheros pci/cardbus NIC's
device          ath_hal         # Atheros HAL (Hardware Access Layer)
device          ath_rate_sample # SampleRate tx rate control for ath
device          awi        # BayStack 660 and others
device          ral        # Ralink Technology RT2500 wireless NICs.
device          wi         # WaveLAN/Intersil/Symbol 802.11 wireless NICs.
#device         wl         # Older non 802.11 Wavelan wireless NIC.
```

用以支持多□无□网□。

```
# Pseudo devices
device   loop          # Network loopback
```

□是 TCP/IP 的通用回□□□。 如果□ telnet 或 FTP 到 localhost (也就是 127.0.0.1) □将通□□个□□回到本机。 □个□□是 必需的。

```
device    random        # Entropy device
```

Cryptographically secure random number generator.

```
device    ether         # Ethernet support
```

ether 只有在使用以太网□□才需要。 它包含了通用的以太网□□代□。

```
device    sl            # Kernel SLIP
```

sl 用以提供 SLIP 支持。 目前它几乎已□完全被 PPP 取代了， 因□后者更容易配置， 而且更□合□制解□器之□的□接， 并提供了更□大的功能。

```
device    ppp           # Kernel PPP
```

□一□□用以提供内核□的 PPP 支持， 用于□号□接。 也有以用□模式□行的 PPP □□， 使用 tun 并提供包括按需□号在内的更□□活的功能。

```
device    tun           # Packet tunnel.
```

它会被用□模式的 PPP □件用到。 参考本□的 PPP 以了解更多的□□。

```
device    pty           # Pseudo-ttys (telnet etc)
```

□是一个 "pseudo-terminal" 或模□登入端口。 它用来接收□入的 telnet 以及 rlogin 会□、 xterm，以及一些其它程序如 Emacs 等。

```
device    md            # Memory disks
```

内存□□□□。

```
device    gif           # IPv6 and IPv4 tunneling
```

它□□了在 IPv4 上的 IPv6 隧道、 IPv6 上的 IPv4 隧道、 IPv4 上的 IPv4 隧道、 以及IPv6 上的 IPv6隧道。 gif □□是 "自□克隆" 的， 它会根据需要自□□建□□□点。

```
device    faith         # IPv6-to-IPv4 relaying (translation)
```

□个□□□能捕捉□□它的数据包，并把它□□□□ IPv4/IPv6 翻□服□程序。

```
# The `bpf' device enables the Berkeley Packet Filter.
# Be aware of the administrative consequences of enabling this!
# Note that 'bpf' is required for DHCP.
device   bpf           # Berkeley packet filter
```

□是 Berkeley 包□□器。□个□□□允□网□接口被置于混□模式， 从而，截□广播网 (例如，以太网) 上的□一个数据包。 截□的数据□可以保存到磁□上，也可以使用 tcpdump(1) 程序来分析。

bpf(4) □□也被用于 dhclient(8) 来□取默□路由器(网□)的 IP 地址。如果使用 DHCP，就不要注□掉□行。

```
# USB support
device          uhci          # UHCI PCI->USB interface
device          ohci          # OHCI PCI->USB interface
device          ehci          # EHCI PCI->USB interface (USB 2.0)
device          usb           # USB Bus (required)
#device         udbp          # USB Double Bulk Pipe devices
device          ugen          # Generic
device          uhid          # Human Interface Devices
device          ukbd          # Keyboard
device          ulpt          # Printer
device          umass         # Disks/Mass storage - Requires scbus and da
device          ums           # Mouse
device          ural          # Ralink Technology RT2500USB wireless NICs
device          urio          # Diamond Rio 500 MP3 player
device          uscanner      # Scanners
# USB Ethernet, requires mii
device          aue           # ADMtek USB Ethernet
device          axe           # ASIX Electronics USB Ethernet
device          cdce          # Generic USB over Ethernet
device          cue           # CATC USB Ethernet
device          kue           # Kawasaki LSI USB Ethernet
device          rue           # RealTek RTL8150 USB Ethernet
```

支持各□ USB □□。

```
# FireWire support
device          firewire      # FireWire bus code
device          sbp           # SCSI over FireWire (Requires scbus and da)
device          fwe           # Ethernet over FireWire (non-standard!)
```

支持各□火□□□。

要了解 FreeBSD 所支持的□□的其他情况， □参考 /usr/src/sys/i386/conf/NOTES。

9.6.1. 大内存支持(PAE)

大内存配置的机器需要超□4 GB的虚□地址。 因□4GB的限制，Intel在Pentium®及后□的CPUs上□加了36位物理地址的支持。

物理地址□展 (PAE) 是 Intel® Pentium® Pro和后□的 CPU 提供的一□允□将内存地址□展到 64GB 的功能，FreeBSD 的所有最新版本均支持此功能， 并通□ PAE □□来□用□个能力。 因□Intel架□的限制， 高于或低于 4GB 都没有什□区□， 超□ 4GB 的内存分配只是□□地添加到可用内存池中。

□了□内核支持PAE，只要□加下面□一行到配置文件：

```
options         PAE
```

PAE在FreeBSD里面□在只能支持 Intel® IA-32□理器。 同□，□□□注意，FreeBSD的PAE支持没有□□广泛的□□， 和其他□定的特性相比只能当作是beta版。

PAE在FreeBSD下有如下的一些限制：

- □程不能接触大于4GB的VM空□。
- 没有使用 bus_dma(9) 接口的□□□□程序在打□了 PAE 支持的内核中会□致数据□坏。 因□□个原因， PAE 内核配置文件 会把所有在打□了 PAE 的内核上不能工作的□□程序排除在外。
- 一些系□打□了探□系□内存□源使用能力的功能，因□打□了 PAE支持，□些功能可能会被覆□掉。 其中一个例子就是内核参数kern.maxvnodes，它是控制 内核能使用的最大vnodes数目的，建□重新□整它及其他□似参数到合□的□。
- □了避免KVA的消耗，很有必要□加系□的内核虚□地址， 或者□少很耗系□□源的内核□□的□量（看上面）。KVA_PAGES□□ 可以用来□加KVA空□。

□了□定和高性能，建□□看tuning(7)手册□。pae(4)手册□包含 FreeBSD’sPAE支持的最新信息。

9.7. 如果出□□□□□□

在定制一个内核□，可能会出□四□□□。它□是：

config失□：

如果 config(8) 在□出□的内核描述□失□， □可能在某些地方引入了一□小的□□。 幸□的是， config(8) 会□示出它遇到□□的行号， □□□就能□迅速地定位□□。 例如， 如果□看到：

```
config: line 17: syntax error
```

可以通□与 GENERIC 或其他参考□料□比， 来□定□里的□□□是否□写正□。

make失□：

如果 make 命令失□， 它通常表示内核描述中□生了 config(8) 无法□出的的□□。 同□地， 仔□□□□的配置，如果仍然不能解决□□， □一封□件到 FreeBSD 一般□□□件列表 并附上□的内核配置， □□□□□很快就能解决。

内核无法□□：

如果□的内核无法□□， 或不□□□的□□， 千万□慌！ 非常幸□的是， FreeBSD 有一个很好的机制□助□从不兼容的内核恢□。 在 FreeBSD □□加□器那里□□地□□一下要□□的内核就可以了。 当系□在引□菜□的 10 秒倒□□□□入它， 方法是□□ "Escape to a loader prompt" □□， 其□号□ 6。 □入 `unload kernel`， 然后□入 `boot /boot/kernel.old/kernel`， 或者其他任何一个可以正□引□的内核即可。 当重新配置内核□，保持一个已□□明能□正常□□的内核永□是一个好□□。

当使用好的内核□□之后□可以□□配置文件并重新□□□□它。 比□有用的□源是 /var/log/messages 文件，它会□□□次成功□□所□生的所有内核消息。 此外， dmesg(8) 命令也会□示□次□□□□生的内核消息。

如果在□□内核□遇到麻□， □□必保留一个 GENERIC 或已知可用的其他内核， 并命名□□的名字以免在下次□□□被覆□。 不要依□ kernel.old 因□在安装新内核□， kernel.old 会被上次安装的那个可能不正常的内核覆□掉。 □外， 尽快把可用的内核□到 /boot/kernel 否□□似 ps(1) □□的命令可能无法正常工作。 □了完成□一点， 需要修改目□的名字：

```
# mv /boot/kernel /boot/kernel.bad
# mv /boot/kernel.good /boot/kernel
```

内核工作，但是**ps(1)**根本不工作

如果□安装了一个与系□中内建工具版本不同的内核， 例如在 -STABLE 系□上安装了 -CURRENT 的内核， □多用于□□系□状□的工具如 ps(1) 和 vmstat(8) 都将无法正常使用。 □□□ 重新□□一个和内核版本一致的系□。 □也是□什□一般不鼓励使用与系□其他部分版本不同的内核的一个主要原因。

Chapter 10. 打印

10.1. 概述

FreeBSD 可以支持多的打印机， 从最古老的式打印机到最新的激光打印机以及它之所有型的打印机， 令行的用程序生高量的打印出。

FreeBSD 也可以配置成网打印服器。 它可以从包括 FreeBSD、 Windows® 及 Mac OS® 在内的多其他算机上接收打印任。 FreeBSD 将保打印任之不会相互干并一次性完成， 而且能机器或用提交打印任的情况行并到其中用量最多的人， 以及生成用于打印任属于位用的 "" 等等。

在完章后，将知道：

- 配置 FreeBSD 后台打印。
- 安装打印器来特殊的打印任做特殊的理， 包括把来的文成打印机能理解的格式。
- 在打印出上或者横幅功能。
- 打印到接在其他算机上的打印机。
- 打印到直接接在网上的打印机。
- 控制打印机的限制， 包括限制打印任的大小和阻止某些用打印。
- 打印机表和使用情况。
- 解决打印故障。

在章之前， ：

- 知道配置并安装新内核 (配置FreeBSD的内核)。

10.2. 介

了在 FreeBSD 中使用打印机， 需要首先配置好伯克利行式打印机后台打印系即 LPD。 它是 FreeBSD 的准打印控制系。 章介 LPD 后台打印系， 在接下来将称 LPD， 并且将指完成其配置。

如果已熟悉了 LPD 或者其他后台打印系， 可以跳到 置后台打印系 部分。

LPD 完全控制一台算机上的打印机。 它多的事情：

- 它控制本地和接在网上其他算机上打印机的。
- 它允用提交要打印的文件; 些通常被是任。
- 它个打印机一个 列 来防止多个用在同一刻一台打印机。
- 它可以打印(也叫做*banner*或者 *burst*使用可以松的从一堆打印出中到它打印的任。
- 它来置接在串口上的打印机的通参数。
- 它能通网将任送到外一台算机的 LPD后台打印列中。
- 它可以根据不同的打印机言和打印机的性能行特殊的器来格式化任。
- 它打印机的使用情况。

通□配置文件 (/etc/printcap)和提供的特殊□□程序， □可以使LPD 系□在□多□□的打印机硬件上完成上面全部的或者一些子集的功能。

10.2.1. □什□要用后台打印

如果□是系□唯一的用□， □可能会奇怪□什□要在□不需要□□控制， □□□或者打印机使用□□□□后台打印□心。它可以□置成允□直接□□打印机， 但□□是□□使用后台打印， 因□：

- LPD在后台打印任□； □不用被迫等待数据被完全副本到打印机的□□。
- LPD可以可以方便的通□□□器□任□加上日期/ □□的□眉或者把一□特殊的文件格式 (比如TeX DVI 文件)□□成一□打印机可以理解的格式。 □不必去手□做□些□□。
- □多提供打印功能的免□和商□程序想要和□□算机上的的后台打印系□通□。 通□□置后台打印系□， □将更□松的支持其他以后要添加的或者□有的□件。

10.3. 基本□置

从 FreeBSD 8.0 起， 串口□□的□□名由 /dev/ttydN □□ /dev/ttyuN。 FreeBSD 7.X 用□□将□篇文□的示例中的□□名改□原先的□子。

要想在 LPD后台打印系□上使用打印机， □需要□置打印机硬件和 LPD□件。 □个 文□描述了□□□□置：

- 参□□□打印机 □置来了解□□□接一个打印机， 告□ LPD□□与 它通□， 并且打印□文本到 打印机。
- 参□ 高□打印机□置 来了解□□打印多□ 特殊格式的文件， □□打印□□□， □□通□网□ 打印， □□控制打印机的□□□限， 并且学会□打印 作□□□□。

10.3.1. □□打印机□置

□部分□解□□配置打印机硬件和 LPD使之与打印机配合。 □解的基□知□有：

- 硬件 □置部分将□解□□把一台打印机□接到 □□算机的一个端口上。
- □件 □置部分将□解□□配置 LPD后台打印的配置 文件 (/etc/printcap)。

如果□正在□置一台通□网□□□ 接收数据来打印而不是通□串口或者并口的打印机， 参□ 使用网□数据流界面的打印机。

尽管□部分叫"□□打印机 □置"， 但□是相当□□的。 使打印机 配合 LPD 后台打印系□在□算机上正常□□是最□的部分。 一旦□的打印机可以正常工作后，那些高□□□， 比如□文□和□□， 是相当□□的。

10.3.1.1. 硬件□置

□部分□述了打印机□接到□算机的多□ 途径。 主要□□了多□接口和 □接□， □有允□ FreeBSD 与打印机通□所需的内核配置。

如果□已□□接好了□的打印机而且已□ 用它在□外一个操作系□下成功的打印， □ 或□可以跳到□个部分□件□置。

10.3.1.1.1. 端口和□接□□

□在所出售的在 PC 上使用的打印机通常至少有 以下三□接口中的一个：

- 串口， 也叫 RS-232 或者 COM 口， 使用□□算机上的串口来□送数据到打印机。 串口在□算机上已□非常普遍， 而且□□也非常容易□到且容易制作。 串口有□需要特殊的□□， 而且可能需要□去配置□微有点儿□□的通□□□。 大多数 PC 的串口的最高□□速度只有 115200 bps， □使得打印很大的□像需要的□□很□。
- 并口 使用□算机上的并口来□送数据到打印机。 并口在□算机上也已□非常普遍， 而且速度高于 RS-232 串口。 □□非常容易□到， 但很□手工制作。 并口通常没有通□□□， □使得配置它相当□□。

  并口按打印机上的接□来命名也叫做 "Centronics"接口。

- USB 接口， 即通用串行□□， 可以□到比并口和串口高很多的速度。 其□□既□□又便宜。 USB 用来打印比串口和并口更有□□， 但 UNIX® 系□不能很好的支持它。 避免□个□□的方法就是□□一台像大多数打印机一□的既有 USB 接口又有并口的 打印机。

一般来□并口只提供□向通□ (□算机到打印机)， 而串口和 USB □可以提供双向通□。 新的并口 (EPP 和 ECP) 及打印机在使用了 IEEE-1284 □准的□□之后， 可以在 FreeBSD 下双向通□。

与打印机通□并口双向通□通常由□□□方法中的一□来完成。 第一个方法是使用□ FreeBSD □写的可以通□打印机使用的□言与打印机通□的□□程序。 □通常用在□墨打印机上， 且可以用来□告剩余墨水多少和其他状□信息。 第二□方法使用在支持 PostScript® 的打印机上。

PostScript® 任□事□上由程序□送□打印机； 但它并不□行打印而是直接将□果返回□□算机。 PostScript® 也采取双向通□来将打印中的□□□告□□算机， 比如 PostScript® 程序中的□□或者打印机□□。 □些信息□于用□来□也□是非常有价□的。 此外， 最好的在支持 PostScript® 的打印机上□□的方法需要双向通□： □□打印机打印□□数 (打印机从出厂一共打印□多少□)， 然后□送用□的任□， 之后再次□□□打印□数。 将打印前后得到的□个□相□就可以得到□用□要付多少□□。

10.3.1.1.2. 并口

用并口□接打印机需要用 Centronics □□把打印机与□算机□接起来。 具体□明指□在打印机， □算机的□明□上□□有， 或者干脆□个上面都有。

□住□用的□算机上的□个并口。 第一个并口在 FreeBSD 上叫 /dev/ppc0； 第二个叫 /dev/ppc1， 依此□推。打印机□□也用同□的方法命名： /dev/lpt0 是接在第一个并口上的打印机， 依此□推。

10.3.1.1.3. 串口

用串口□接打印机需要用 合□的串口□□把打印机与□算机□接起来。 具体 □明指□□□在打印机， □算机的□明□上有， 或者 同□干脆□个上面都有。

如果□不□定什□□儿的□□才是 " 合□的串口 □□ " ， □可以□□以下几□不同的 □□：

- □制解□器 □□□一端的 □一根引脚都直接□接到□一端 相□的引脚 上。 □□□□也叫做 "DTE-to-DCE" □□。
- 非□制解□器□□上□一端的有些引脚 是与□一端相□引脚直接□接的， 而有一些□是交叉□接的 (比如， □送数据引脚□接到 接收数据引脚)， □有一些引脚直接在□□□接□儿内 短接。 □□□□也叫做 "DTE-to-DTE" □□。
- 一些特殊的打印机需要的串口打印机 □□， 是一□和非□制解□器□□□似的□□， 只是一些信号□是送到了□一端， 而 不是直接在□接□儿内短路。

当然， □□得□打印机□置通□参数。 一般是通□打印机面板上的按□或者 DIP □□□行□置。 在□算机和打印机上都□□它□所支持的最高 波特 (□秒多少比特， 有□也叫 波特率) 的□□速率。 □□7或者

8个数据位； □□不校□， 偶校□或者奇校□； □□1个或2个停止位。 □要□□流量控制□□： 无， XON/XOFF (也叫做 "in-band" 或 "□件") 流量控制。 □住□的□件配置中的参数也要□成上面的数□。

10.3.1.2. □件□置

□部分描述了要使用 FreeBSD 系□中的 LPD 后台打印系□□行打印所需的□件□置。

包括□几个□□：

1. 在需要的□候配置内核来允□□□接 打印机的端口； 配置内核 部分会告□□ 需要做什□。
2. 如果□使用并口， □需要□置一下 并口的通□模式; □置 并口通□模式 部分会告□□具体的 □□。
3. □□操作系□是否能□□送数据到打印机。 □□打印机 □机状况 部分会告□□要□□ 做。
4. □ LPD □置与打印机匹配的参数□ 通□修改 /etc/printcap □个文件来完成。 □章后面 的部分将□解如何来完成□置。

10.3.1.2.1. 配置内核

操作系□的内核□了使某些特殊□□工作需要重新 □□。 打印机所用的串口、 并口就属于那些特殊□□。 因此，可能需要 添加□串口或并口的支持， 如果内核并没有配置它□的□。

要想知道□□在使用的内核是否支持串口， □入：

```
# grep sioN /var/run/dmesg.boot
```

其中 *N* 是串口的 □号， 从0□始。 如果□看到 □似下面的□出：

```
sio2 at port 0x3e8-0x3ef irq 5 on isa
sio2: type 16550A
```

□□明□□在使用的内核支持串口。

要想知道□□在使用的内核是否支持并口， □入：

```
# grep ppcN /var/run/dmesg.boot
```

其中 *N* 是并口的 □号， 同□从0□始。 如果得到□似 下面的□出：

```
ppc0: <Parallel port> at port 0x378-0x37f irq 7 on isa0
ppc0: SMC-like chipset (ECP/EPP/PS2/NIBBLE) in COMPATIBLE mode
ppc0: FIFO with 16/16/8 bytes threshold
```

那□□□在使用的内核支持并口。

□可能必□□了使操作系□支持□打印机需要的串口或 并口而 重新配置内核。

要□加□串口的支持， 参□ 内核配置□部分。 要□加□并口的支持， 除了参□ 上面提到的那部分之外， □要 参□下面的 部分。

10.3.1.3. □置并口的通□模式

在使用并口□， □可以□□ □ FreeBSD 用中断方式□是□□方式来 与打印机通□。 在 FreeBSD 上，通用的打印机□□ (lpt(4)) 使用 ppbus(4) 系□， 它利用 ppc(4) □□来控制端口芯片。

- 中断 方式是 GENERIC 核心的默□方式。 在□□方式下， 操作系□占用一条中断□求□来□□打印机是否已□做好接收数据的准□。
- □□ 方式是操作系□反□不断的□□打印机是否做好接收数据的准□。 当它返回就□□， 核心□始□送下面要□送的数据。

中断方式速度通常会快一些， 但却占用了一条宝□的中断□求□。 一些新出的 HP 打印机不能正常的工作在中断模式下， 是由于一些定□□□ (□没正□的理解) 造成的。 □些打印机需要使用□□方式。□□□使用 任何一□方式， 只要它能正常工作就行。 一些打印机□然在□□模式下都可以 工作，但在中断模式下会慢的要命。

□可以用以下□□方法□定通□模式： 通□ 配置内核或者使用 lptcontrol(8) □个程序。

要通□配置内核的方法□置 通□模式：

1. 修改内核配置文件。 □到 一个叫 ppc0 的□□。 如果□想要□置的是 第二个并口， 那□用 ppc1 代替。使用第三个并口的□候用 ppc2 代替， 依此□推。
   - 如果□希望使用中断□□模式， □□□□下面的配置：

     ```
     hint.ppc.0.irq="N"
     ```

     它在 /boot/device.hints □个文件中， 其中 *N* 用正□的中断 □号代替。 同□， 核心配置文件也必□包括 ppc(4) 的□□：

     ```
     device ppc
     ```

   - 如果□想要使用□□方式， 只需要把 /boot/device.hints □个文件中的下面□行□除掉：

     ```
     hint.ppc.0.irq="N"
     ```

     在 FreeBSD 下， 有□上面的方法并不能使并口工作在□□方式。 大多数情况是由于 acpi(4) □□造成的， 它可以自□□□到□□并将其挂□到系□上， 但也因此， 它控制着打印机端口的□□模式. □需要□□ acpi(4) 的配置来解决□个□□。
2. 保存文件。 然后配置， 建立， 并安装□配置的内核， 最后重新□□。 参□ 内核配置 □章来□得更多□□。

使用 lptcontrol(8) □置通□模式：

1. □入：

   ```
   # lptcontrol -i -d /dev/lptN
   ```

   将 lptN □置成中断方式。

2. □入：

   ```
   # lptcontrol -p -d /dev/lptN
   ```

   将 lptN □置成□□方式。

□可以把□些命令加入到 /etc/rc.local □个文件中， □□□次□□系□ □都会□置成□想要的方式。 参□ lptcontrol(8) 来□得 更多信息。

10.3.1.4. □□打印机的通□

在□置后台打印系□之前， □□□□保□的□算机可以把数据 □送到打印机上。 分□独立□□打印机的通□和后台打印系□会更□□。

我□□了□□打印机，将□送一些文本□它。 一个叫 lptest(1) 的程序能□任□□工作， 它可以□打印机立即打印出程序□□它的 字符： 它在□行打出 可以打印的 96 个 ASCII 字符。

当我□使用的是一台 PostScript® （ 或者以其他□言□基□的 ） 打印机， 那□ 需要更仔□的□□。 一段小小的 PostScript® 程序足以完成□□的任□， 比如下面□段程序：

```
%!PS
100 100 moveto 300 300 lineto stroke
310 310 moveto /Helvetica findfont 12 scalefont setfont
(Is this thing working?) show
showpage
```

可以把上面□段 PostScript® 代□写□一个文件里， 并且像下面部分的例子里那□ 使用。

上面的小程序是□□ PostScript® 而不是惠普的 PCL 写的。 由于 PCL □有□多其他打印机没有的□大功能， 比如它支持在打印□文本的同□□□特殊的命令， 而 PostScript® □不能直接打印□文本， 所以需要□□□打印机□言□行特殊的□理。

10.3.1.4.1. □□并口打印机

□部分内容将指□□□□□□ FreeBSD 是否可以与一台已□□接在并口上的打印机通□。

要□□并口上的打印机：

1. 用 su(1) 命令□□到 root 用□。
2. □送数据到打印机。
   - 如果打印机可以直接打印□文本， 可以用 lptest(1)。 □入：

     ```
     # lptest > /dev/lptN
     ```

     其中 *N* 是并口的□号， 从0□始。
   - 如果打印机支持 PostScript® 或其他打印机□言， 可以□送一段小程序到打印机。 □入：

     ```
     # cat > /dev/lptN
     ```

     然后， 一行一行地 □入 □入□段程序。 因□在按下 □行 或者 回□ 之后， □一行就不能再修改了。 当□□入完□段程序之后， 按 CONTROL+D， 或者其他表示文件□束的□。

     □外一□□法， □可以把□段程序写在一个文件里， 并□入：

     ```
     # cat file > /dev/lptN
     ```

     其中 *file* 是包含□□要□□打印机程序的文件名。

之后， □□□看到打印出了一些□西。 如果打印出的□西看起来并不正□， □不要着急； 我□将在后面指□□如何解决□□□□。

10.3.1.4.2. □□串口打印机

□部分将告□□如何□□ FreeBSD 是否可以与□接在串口上的打印机通□。

要□□□接在串口上的打印机：

1. 通□ su(1) 命令□□ root 用□。
2. 修改 /etc/remote □个文件。 □加下面□些内容：

   ```
   printer:dv=/dev/port:br#bps-rate:pa=parity
   ```

   其中 *port* 是串口的□□□点 (ttyu0、 ttyu1， 等等)， *bps-rate* 是与打印机通□□使用的速率， 而 *parity* 是通□□打印机要求的校□方法 (□□是 even、 odd、 none， 或 zero 之一)。

   □儿有一个串口打印机的例子， 它□接在第三个串口上， 速度□ 19200 波特， 不□行校□：

   ```
   printer:dv=/dev/ttyu2:br#19200:pa=none
   ```

3. 用 tip(1) □接打印机。 □入：

   ```
   # tip printer
   ```

   如果没能成功， □要再次修改 /etc/remote □个文件， 并且□□用 /dev/cuaaN 代替 /dev/ttydN。

4. □送数据到打印机。

   ◦ 如果打印机可以直接打印□文本， □用 lptest(1)。 □入：

   ```
   % $lptest
   ```

   ◦ 如果打印机支持 PostScript® 或者其他 打印机□言， □□送一段小程序到 打印机。 一行一行的□入程序， 必□ 非常仔□ 因□像退格 或者其他□□□也□□打印机来□有它的 意□。 □同□也需要按一个特殊的 文件□束□， □打印机知道它已□ 接收了整个程序。 □于 PostScript® 打印机， 按 CONTROL+D。

   或者， □同□也可以把程序存□在一个文件里 并□入：

   ```
   % >file
   ```

   其中 *file* 是 包含要□送程序的文件名。 在 tip(1) □送□个文件之后， 按代表 文件□束的□。

□□□看到打印出了一些□西。 如果它□看起来并不正□也不要着急； 我□将在□后的章□中介□如何解决□□□□。

10.3.1.5. □用后台打印： 文件 /etc/printcap

目前， □的打印机□□已□□好了□， 系□内核也□与打印机□机而重新配置好 (如果需要的□)， 而且□也已□可以□送一些□□的数据到打印机。 □在， 我□要配置 LPD 来使其控制□的打印机。

配置 LPD 要修改 /etc/printcap □个文件。 由于 LPD 后台打印系□在□次使用后台打印的□候， 都会□取

□个文件， 因此□□个文件的修改会立即生效。

printcap(5) □个文件的格式很□□。 □可以用□最喜□的文本□□器来修改 /etc/printcap □个文件。 □□格式和其他的像 /usr/shared/misc/termcap 和 /etc/remote □□文件是一□的。 要得到□于□□格式的□尽信息， □参□□机手册 cgetent(3)。

□□的后台打印配置包括下面的几□：

1. □打印机起一个名字 (□□和使用的□名)， 然后把它□写□文件 /etc/printcap； 参□ 如何□打印机命名 □章来得到更多的□于起名的□助。
2. 通□□加 sh □□掉□□□ (它默□是□用的)； 参□ 如何禁用□□□ 部分来得到更多信息。
3. 建立一个后台打印□列的目□， 并且通□ sd □目指定它的位置； □可参□ □建后台打印□列目□ 一□了解更多信息。
4. 在 /dev 下□置打印机□□□点， 并且在写在 /etc/printcap 文件中 lp □目里； 参□ □□打印机□□ □部分可以得到更多信息。 此外， 如果打印机□接在串口上， 通□参数的□置需要写在 ms# □中。 □些参数在 配置后台打印通□参数 □在前面已□□□□。
5. 安装□文本□□器； □情□参□ 安装文本□□器 小□。
6. 用 lpr(1) 命令来□□□置。 想得到更多信息可以参□ □□ 和 故障排除 部分。

使用打印机□言的打印机， 如 PostScript® 打印机， 通常是不能直接打印□文本的。 前面提到， 并且将在后面□□□行介□的□□的□置方法， 均假定□正在安装□□只能打印它能□□的文件格式的打印机。

用□通常会希望直接在系□提供的打印机上打印□文本。 采用 LPD 接口的程序也通常是□□□□的。 如果□正在安装□□一台打印机， 并且希望它不□能打印使用它支持的打印机□言的任□ 而且 □能打印□文本的任□的□， 那□□烈建□□在上面提到的□□□置的□□上□加一□： 安装从自□□文本到 PostScript® (或者其他打印机□言)的□□程序。 更多的□□， □参□ 在 PostScript® 打印机上打印□文本。

10.3.1.5.1. 打印机的命名

第一□ (□□) 就是□打印机起一个名字。 □是按功能起名字□是干脆起个古怪的名字都没有□系， 因□□可以□打印机□置□多的□名。

在 /etc/printcap 里至少有一个打印机必□指定， □名是 lp。 □是默□的打印机名。 如果用□既没有 PRINTER □境□量， 也没有在任何 LPD 命令的命令行中指定打印机名， □ lp 将是默□要使用的打印机。

□有， 我□通常把最后一个□名□置成能完全描述打印机的名字， 包括厂家和型号。

一旦□□好了名字或者一些□名， 把它□放□文件 /etc/printcap 里。 打印机的名字□□从最左□的一列写起。 用□杠来隔□□个□名， 并且在最后一个□名后面加上一个冒号。

在下面的例子中， 我□从一个基本的 /etc/printcap □始， 它只定□了□台打印机 (一台 Diablo 630 行式打印机和一台 Panasonic KX-P4455 PostScript® 激光打印机):

```
#
#  /etc/printcap for host rose
#
rattan|line|diablo|lp|Diablo 630 Line Printer:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:
```

在□个例子中， 第一台打印机被命名□ rattan 并且□置了 line， diablo, lp， 和 Diablo 630 Line Printer □几个□名。 因□它被□置了 lp □个□名， 所以它是默□打印机。 第二台 被命名□ bamboo， 并且□置了 ps， PS, S， panasonic， 和 Panasonic KX-P4455 PostScript v51.4 □几个□名。

10.3.1.5.2. 不打印□□□

LPD 后台打印系□默□ 会□□个任□打印 □□□。 □□□ 包含了□送□个任□的用□， □送□个任□的□算机， 任□的名字，并用大字母打出。 但不幸的是， 所有□些□外的文本， 都会□在□打印机□行最初的配置□排除故障□来困□，所以我□将先不打印□□□。

要□停打印□□□， □打印机的□□□加 sh □□， 在 /etc/printcap 文件中。 □儿有一个 /etc/printcap 文件中使用 sh 的例子:

```
#
#  /etc/printcap for host rose - no header pages anywhere
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:
```

注意我□的正□格式: 第一行从最左□一列□始， 而后的□一行用 TAB □□一次。 一行写不下需要□行□， 在□行前打一个反斜杠。

10.3.1.5.3. 建立后台打印□列目□

下一□□置就是要建立一个 后台打印□列目□， 也就是在打印任□最□完成之前用于存放□些任□的目□， □个目□中也会存放后台打印系□用到的其他一些文件。

由于后台打印□列目□的□量本□， 通常 把□些目□安排在 /var/spool 下。 □也没有必要去 □□后台打印□列目□里的内容。 重新建立它□只要□□的使用 mkdir(1) 命令。

通常， 我□□□将目□名起成和 打印机一□的名字， 像下面 □□：

```
# mkdir /var/spool/printer-name
```

然而， 如果□有很多网□打印机， □可能想要把□些后台打印的□列目□目□放在一个□独的□□使用 LPD 打印而准□的目□里。 我□将用我□的□台打印机 rattan 和 bamboo 作□例子：

```
# mkdir /var/spool/lpd
# mkdir /var/spool/lpd/rattan
# mkdir /var/spool/lpd/bamboo
```

如果担心用□任□的保密性， 可能会希望保□相□的后台打印□列目□， 使之不能被其他用□□□。 后台打印的□列目□的属主□□是 daemon 用□， 而 daemon 用□和 daemon □□有□写和搜索的□限，但其他用□没有。 接下来用我□的□台打印机作□例子:

```
# chown daemon:daemon /var/spool/lpd/rattan
# chown daemon:daemon /var/spool/lpd/bamboo
# chmod 770 /var/spool/lpd/rattan
# chmod 770 /var/spool/lpd/bamboo
```

最后， □需要通□/etc/printcap 文件告□ LPD □些目□。 □可以用 sd □□来指定后台打印□列目□的路径:

```
#
#  /etc/printcap for host rose - added spooling directories
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:sd=/var/spool/lpd/rattan:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:
```

注意打印机的名字要从第 1 列□始， 其他□□□行都要用 TAB □□□一次， 写不□需要□行在最后加上反斜杠。

如果□没用 sd □□指定后台打印□列目□， 后台打印系□会将 /var/spool/lpd 目□作□默□目□。

10.3.1.5.4. □□打印机□□

在 Hardware Setup 一□中，我□□明了 FreeBSD 与打印机通□将使用□个端口和 /dev 目□下的□点。 我□要告□ LPD □些信息。 当后台打印系□有任□需要打印，它将□□□程序 （□□□送数据到打印机） 打□指定的□□。

用 lp □□在 /etc/printcap 里列出 /dev 下的□□□点。

在我□的例子中， 假□打印机 rattan 在第一个并口上， 打印机 bamboo 在第六个串口上； 下面是 要□ /etc/printcap 文件里□加的内容 :

```
#
#  /etc/printcap for host rose - identified what devices to use
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:sd=/var/spool/lpd/rattan:\
        :lp=/dev/lpt0:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:\
        :lp=/dev/ttyu5:
```

如果□没在□的 /etc/printcap 文件中 用 lp □□指定□□□点， LPD 将默□使用 /dev/lp 。 /dev/lp 目前在 FreeBSD 中不存在。

如果□正在安装的打印机是□接在 并口上的， □跳到 安装文本 □□器 □章。 如果不是的□， □是最好按下面介□的□□做。

10.3.1.5.5. 配置后台打印通□参数

□于□在串口上的打印机， LPD 可以□□送数据到打印机的□□程序□置好波特率， 校□， 和其他串口通□参数 。□是有利的， 因□:

- 它可以□□只需□□的修改 /etc/printcap 就能□□不同的通□ 参数; □并不需要去重新□□□□器 程序。
- 它使得后台打印系□可以在 多台有不同串口通□□置的打印机上使用 相同的□□器程序。

下面□个 /etc/printcap 中 用 lp □□来控制列出□□的 串口通□参数 :

br#bps-rate

□置□□的通□速度□ *bps-rate*， □里 *bps-rate* 可以□ 50， 75， 110, 134， 150， 200， 300， 600， 1200， 1800， 2400， 4800， 9600, 19200， 38400， 57600， or 115200 比特□秒。

ms#stty-mode

□置已打□的中端□□的□□ 。 stty(1) 将□□ □述可用的□□。

当 LPD 打□ 用 lp 指定的□□□， 它会 将□□的特性□置成在 ms# □□后指定的那□。 特□是 parenb, parodd， cs5, cs6， cs7, cs8， cstopb, crtscts， 和 ixon □些模式， 它□在 stty(1) 手册中有□□□明。

我□□个例子来添加我□□在第6个串口上的 打印机。 我□将□波特□38400。 至于模式， 我□将用 -parenb □置成不校□， 用 cs8 □置成8位字符， 用 clocal □置成不要□制解□器控制， 用 crtscts □置成硬件流量控制：

```
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:\
        :lp=/dev/ttyu5:ms#-parenb cs8 clocal crtscts:
```

10.3.1.5.6. 安装文本□□器

我□□在准□告□ LPD 使用什□文本□□器 □打印机□送任□。 文本□□器， 也叫 □入□□器， 是一个 在 LPD 有一个任□要□□ 打印机□□行的程序。 当 LPD □打印机□行文本□□器□， 它□置□□器的 □准□入□要□□打印机的任

□， 而□准□出□ 用 lp □□指定的打印机 。 □□器先从□准□入□取 任□， □打印机□行一些□□ ， 并将□果写到□准□出， □些□果 将被打印。 想得到更多□于文本□□器的信息， □ □□器 □□。

□于□□的打印机□置， 文本□□器可以□□是一段 □行 /bin/cat 的 shell 脚本来 □送任□到打印机。 FreeBSD □提供了一个叫做 lpf 的□□器， 它可以□理退格和下□□来 使那些可能不能很好□理□□字符流的打印机正常工作。 而且， 当然， □可以用任何其他的 □想用的□□程序。 lpf □□器在 lpf: 一个文本□□器 □□将有□□描述。

首先， 我□来写一段叫做 /usr/local/libexec/if-simple 的□□ shell 脚本作□文本□□器。 用□熟悉的文本□□器将下面的内容放□ □个文件：

```
#!/bin/sh
#
# if-simple - Simple text input filter for lpd
# Installed in /usr/local/libexec/if-simple
#
# Simply copies stdin to stdout.  Ignores all filter arguments.

/bin/cat && exit 0
exit 2
```

使□个文件可以被□行：

```
# chmod 555 /usr/local/libexec/if-simple
```

然后用 if □□在 /etc/printcap 里告□ LPD 使用□个脚本。 我□将仍然□ 一直作□例子的□□台打印机在 /etc/printcap 里□加□个□□：

```
#
#  /etc/printcap for host rose - added text filter
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:sd=/var/spool/lpd/rattan:\ :lp=/dev/lpt0:\
        :if=/usr/local/libexec/if-simple:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:\
        :lp=/dev/ttyu5:ms#-parenb cs8 clocal crtscts:\
        :if=/usr/local/libexec/if-simple:
```

if-simple 脚本的副本可以在 /usr/shared/examples/printing 目□中□到。

10.3.1.5.7. □□ LPD

lpd(8) 在 /etc/rc 中被□行， 它是否被□行由 lpd_enable □个□量控制。 □个 □量默□是 NO。 如果□□没有修改， 那□□加□行：

```
lpd_enable="YES"
```

到 /etc/rc.conf 文件当中， 然后既可以重□□的 机器， 也可以直接□行 lpd(8)。

```
# lpd
```

10.3.1.5.8. □□

□在已□基本完成了 LPD 的基本□置。 但不幸的是， □不是□祝的□候， 因□我□□需要□□□置并且修正所有的□□。 要□□□置， □□打印一些□西。 要 用 LPD 系□打印， □可以 使用 lpr(1) 命令， 它可以提交一个任□来打印。

□可以□合使用 lpr(1) 和 lptest(1) 程序， 在 □□打印机 通□ □□介□□□生成一些□□文本。

要□□□□ *LPD* □置：

□入：

```
# lptest 20 5 | lpr -Pprinter-name
```

其中 *printer-name* 是 在 /etc/printcap 中指定的打印机的一个名字 (或者一个□名) 。 要□□默□ 打印机， □入 lpr(1) 不□任何 -P □□。 同□， 如果□正在□□一台使用 PostScript® 的打印机， □送一个 PostScript® 程序到打印机而不是 使用 lptest(1)。 □可以把程序放在一个 文件里， 然后□入： lpr file。

□于一台 PostScript® 打印机， □□□得到那段程序的 □果。 而如果□使用的 lptest(1)， □□得到的 □果□□看起来像下面□□：

```
!"#$%&'()*+,-./01234
"#$%&'()*+,-./012345
#$%&'()*+,-./0123456
$%&'()*+,-./01234567
%&'()*+,-./012345678
```

要更□一□的□□打印机， □□下□一些大的 程序 (□基于特定□言的打印机) 或者□行 lptest(1) 并使用不同的参数。比如， lptest 80 60 将生成 60 行 □行 80 个字符。

如果打印机不能工作， 参考 故障排除 □□。

10.4. 高□□置

从 FreeBSD 8.0 起， 串口□□的□□名由 /dev/ttydN □□ /dev/ttyuN。 FreeBSD 7.X 用□□将□篇文□的示例中的□□名改□原先的□子。

□部分将描述用来打印特□格式文件， □眉， 通□网□打印， 以及□打印机使用限制和 □□。

10.4.1. □□器

尽管 LPD □理网□□□， 任□排□， □□控制， 和打印的其他方面， 但大部分 □□ 工作□是由 □□器。 □□器是 一□与打印机通□并且□理□□依□和特殊需要的 程序。 在□□打印机□置□□里， 我□安装了一个□文本□□器 - 一个□□可以用在大多数打印机上的□□□的□□器 (安装文本□□器)。

然而，□了□行格式□□， 打印□□， □□特殊的打印机， 等等， □需要明白□□器是□□工作的。 在根本上□□器□□□理□些方面。 但坏消息是大多数□候 □ 必□自己提供□□器。 好消息是很多□□器通常都已□有了；当没有的□候，它□通常也是很好写的。

FreeBSD 也提供了一个□□器， /usr/libexec/lpr/lpf， 可以□大多数可以打印□文本的打印机工作。 （ 它□理文件里的退格和跳格，并且□行□□， 但□基本就是它所有能做的了。） □里□有几个□□器和□□器□件在 FreeBSD Ports Collection 里。

□是在□□里□将□到的内容：

- 在 □□器是如何工作的 小□中将介□在打印□程中□□器的作用。 如果希望了解在 LPD 使用□□器□， 在 "幕后" □生的事情， 便□□□□一小□。 了解□些知□能□□助□在□打印机安装□□器□更快地排□可能会遇到的各□□□。
- LPD 假定任何打印机在默□状□下均能打印□文本内容。 □于不能直接打印□文本的 PostScript® 打印机 (以及其他基于打印□言的打印机) 而言□会□来□□。 在 在 PostScript® 打印机上使用□文本任□ □□ 中将会介□如何解决□个□□的方法。 如果□使用 PostScript® 打印机， 就□□□□□内容。
- PostScript® □于□多程序来□都是一个非常受□迎的□出格式。 一些人甚至直接写 PostScript® 代□。 但不幸的是， PostScript® 打印机非常昂□。 模□ PostScript® 在 非 PostScript® 打印机上 □□将告□□□□一□修改 打印机的文本□□器， 使得一台 非 *PostScript*® 打印机接受 并打印 PostScript® 数据。 如果 □没有 PostScript® 打印机， 那□□□□□□个小□。
- □□□□器 □□□述了一个自□把指定格式文件， 比如□像或排版数据， □□成□打印机可以理解的格式的方法。 在□□了□□之后， □就□□可以配置打印机， □用□可以用 `lpr -t` 来打印 troff 数据、 用 `lpr -d` 来打印 TeX DVI 数据， 或用 `lpr -v` 来打印光□□像数据等工作了。 建□□□□□□。
- □出 □□器 □□□述了□个不是□常使用的 LPD: 的功能－□出□□器。 除非□要打印□眉 (□ □眉 □□)， □或□可以完全跳□□□。
- lpf: 一个文本□□器 描述了 `lpf`， 一个 FreeBSD 自□的相当完整而又□□的文本□□器，可以使用在行式打印机(和那些担当行式打印机功能的激光打印机)上。 如果□需要一个快速的方法来□打印机□□打印□文本的工作量， 或者□有一台遇到退格字符就冒□的打印机， □□□考□ `lpf`。

□可以在 /usr/shared/examples/printing 目□中□到下面将提到的那些脚本的副本。

10.4.1.1. □□器是□□工作的

前面□□， □□器是一个被 LPD □□， 用来□理与打印机通□□程中□□依□的部分 的可□行程序。

当 LPD 想要打印 一个任□中的文件， 它□□一个□□器 程序。 它把要打印的文件□置成□□器的□准□入， □准□出□置成打印机， 并且把□□信息定向到 □□日志文件 (在 `lf` □□里指定， 默□在 /etc/printcap， 或者 /dev/console 文件里)。

□□器被 LPD □□， 并且□□器的参数依□于 /etc/printcap 文件中所列出的和用□□任□用 lpr(1) 命令所指定的。例如， 如果用□□入 `lpr -t`， LPD 会□□ troff □□器， 即在目□打印机的 `tf` □□里所列出的□□器。 如果用□想要打印□文本， 它将会□□ `if` □□器 (□是通常的情况： 参□ □出□□器 来得到 □□)。

在 /etc/printcap 文件中， □可以指定三□□□器：

- The 文本□□器， 在 LPD 文□中也叫做 □入□□器 ， □理 常□的文本打印。 可以把它想象成默□□□器。 LPD 假定□台打印机默□情况下都可以打印□文本， 而文本□□器的任□就是来□定退格、 跳格， 或者其他在某□打印机上容易□□的特殊字符。 如果□所在的□境□打印机的使用情况□行□□， 那□文本□□器必□也□打印的□数□行□□， 通常是根据打印的行数和打印机在□□上能打印的行数□行□算得出。 文本□□器的□□命令□：

  `filter-name` [-c] -w *width* -l *length* -i *indent* -n *login* -h *host acct-file*

  □里

  `-c`

  当任□用 `lpr -l` □个命令提交□出□

  width

  □里取□在 /etc/printcap 文件中指定的 `pw` (□ □) □□的□， 默□□ 132。

  length

  □里取□的 `pl` (□ □) □□的□， 默□□ 66

  indent

  □里是来自 `lpr -i` 命令的□□□量， 默□□ 0

  login

  □里是正在打印文件的用□名

  host

  □里是提交打印任□的主机名

  acct-file

  □里是来自 `af` □量中指定的用于□□的文件名。

- □□□□器 的功能是， 将特定格式的文件□□成打印机能□□□并打印的格式。 例如， ditroff 格式的排版数据就是无法直接打印的， 但□可以安装一个□□□□器来将 ditroff 文件□□成一□打印机可以□□和打印的形式。 □参□ □□□□器 □一□来了解更多□□。 如果□需要□打印□行□□， 那□□□□□器也必□完成□□工作。 □□□□器的□□命令□：

  `filter-name` -x *pixel-width* -y *pixel-height* -n *login* -h *host acct-file*

  □其中 *pixel-width* 的□来自 `px` □□ (默□□ 0)， 而 *pixel-height* 的□来自 `py` □□ (默□□ 0)。

- □出□□器 □在没有文本□□器□， 或者□□□被打□□使用。 就我□的□□而言， □出□□器是很少用到的. 在 □出□□器□□中会介□它□。 □□□出□□器的命令行只有□个参数： `filter-name` -w *width* -l *length*

  它□的作用与文本□□器的 `-w` 和 `-l` 参数是一□的。

□□器也□□在 退出 □□出下面的几□退出状□:

exit 0

□□器已□成功的打印了文件.

exit 1

□□器打印失□了， 但希望 LPD □着再打印一次。 如果□□器返回了□个状□， LPD 将重新□□□□器。

exit 2

□□器打印失□并且不希望 LPD 重□。 □□情况下 LPD 会放□□个文件。

文本□□器随 FreeBSD 一起□布， 文件名□ /usr/libexec/lpr/lpf， 它利用□□和□□参数来决定何□□送送□指令，并提供位打印□□的方法。 它使用登□名、 主机名， 和□□文件参数来生成□□□□。

如果□想□□□□器， 要注意它是否是与 LPD 兼容。 如果兼容的□， □它□必□支持前面提到的那些参数。 如果□打算□写普通的□□器程序， □同□需要使之支持前面那些参数和退出状□□。

10.4.1.2. 在 PostScript® 打印机上打印□文本任□

如果□是□的□算机和 PostScript® (或其他□言的) 打印机的唯一用□， 而且□不打算□送□文本到打印机，并因此不打算从□用程序程序直接将□文本□到打印机的□， 就完全不需要再□心□□的内容了。

但是， 如果打印机同□需要接收 PostScript® 和□文本的任□， 就需要□打印机□行□置了。 要完成□□工作，我□需要一个文本□□器来□□到□的任□是□文本的□是 PostScript® 格式的。 所有 PostScript® 的任□必□以 `%!` (其他打印机□言□参□打印机的文□) □□。 如果任□的□□个字符是□□个， 就代表□是 PostScript® 格式的，并且可以直接略□任□剩余的部分。 如果任□□□的□个字符不是□□个， 那□□□器将把文本□□成 PostScript® 并打印□果。

我□□□去做？

如果□有一台串口打印机， 一个好□法就是安装 `lprps`。 `lprps` 是一个可以与打印机□行双向通信 PostScript® 打印机□□器。 它用打印机□来的□□信息来更新打印机的状□文件， 所以用□和管理□可以准□的看到打印机□在什□□的状□ (比如 缺墨 或者 □□)。 但更重要的是， 它包含了一个叫做 `psif` 的程序， 它可以□□接收到的文件是否是□文本的， 并且将使用 `textps` 命令 (也是由 `lprps` 提供的程序) □□文本到 PostScript®。 然后它会用 `lprps` 将任□□送到打印机。

`lprps` 可以在 FreeBSD Ports Collection (□□ The Ports Collection) 中□到。 □可以根据□面的尺寸□□安装 print/lprps-a4 和 print/lprps-letter。 在安装了 `lprps` 之后， 只需指定 `psif` □个程序的路径， □也是包含在 `lprps` 中的一个程序。 如果□已□用 ports 安装好了 `lprps`， 将下面的内容添加到 /etc/printcap 文件中 PostScript® 打印机的□□部分中：

```
:if=/usr/local/libexec/psif:
```

同□□需要指定 `rw` □□来告□ LPD 使用□-写模式打□打印机。

如果□有一台并口的 PostScript® 打印机 (因此不能与打印机□行 `lprps` 需要的双向通信)， 可以使用下面□段 shell 脚本来充当文本□□器:

```
#!/bin/sh
#
#  psif - Print PostScript or plain text on a PostScript printer
#  Script version; NOT the version that comes with lprps
#  Installed in /usr/local/libexec/psif
#

IFS="" read -r first_line
first_two_chars=`expr "$first_line" : '\(..\)'`

if [ "$first_two_chars" = "%!" ]; then
    #
    #  PostScript job,  print it.
    #
    echo "$first_line" && cat && printf "\004" && exit 0
    exit 2
else
    #
    #  Plain text,  convert it,  then print it.
    #
    ( echo "$first_line"; cat ) | /usr/local/bin/textps && printf "\004" && exit 0
    exit 2
fi
```

在上面的脚本中， `textps` 命令是一个独立安装的程序用来将□文本□□成 PostScript®。 □可以使用任何□喜□的文本到 PostScript® □□程序。 FreeBSD Ports Collection (□□ Ports Collection) 中包含了一个功能非常完整的文本到 PostScript® 的□□程序， 它叫做 `a2ps`。

10.4.1.3. 模□ PostScript® 在非 PostScript® 打印机上

PostScript® 是高□量排版和打印 事□上的 □准。 而 PostScript® 也是一个 昂□ 的□准。 幸好， Aladdin □□了一个和 PostScript® □似的叫做 Ghostscript 的程序可以用在 FreeBSD 上。 Ghostscript 可以□取大多数 PostScript® 的文件并□理其中的□面交□多□□，包括□多品牌的非 PostScript® 打印机。 通□安装 Ghostscript 并使用一个特殊的文本□□器，□可以使一台非 PostScript® 打印机用起来就像真的 PostScript® 打印机一□。

Ghostscript 被收□在 FreeBSD Ports Collection 中，有□多可用的版本， 比□常用的版本是 print/ghostscript-gpl。

要模□ PostScript®， 文本□□器要□□是否要打印一个 PostScript® 文件。 如果不是， 那□□□器将直接将文件□送到打印机； 否□， 它会用 Ghostscript 先将文件□□成打印机可以理解的格式。

□里有一个例子: 下面的脚本是一个□□ Hewlett Packard DeskJet 500 打印机的文本□□器。 □于其他打印机，替□ `gs` (Ghostscript) 命令中的 `-sDEVICE` 参数就可以了。 (□入 `gs -h` 来□得当前安装的 Ghostscript 所支持的□□列表。)

```
#!/bin/sh
#
#  ifhp - Print Ghostscript-simulated PostScript on a DeskJet 500
#  Installed in /usr/local/libexec/ifhp

#
#  Treat LF as CR+LF (to avoid the "staircase effect" on HP/PCL
#  printers):
#
printf "\033&k2G" || exit 2

#
#  Read first two characters of the file
#
IFS="" read -r first_line
first_two_chars=`expr "$first_line" : '\(..\)'`

if [ "$first_two_chars" = "%!" ]; then
    #
    #  It is PostScript; use Ghostscript to scan-convert and print it.
    #
    /usr/local/bin/gs -dSAFER -dNOPAUSE -q -sDEVICE=djet500 \
      -sOutputFile=- - && exit 0
else
    #
    #  Plain text or HP/PCL,  so just print it directly; print a form feed
    #  at the end to eject the last page.
    #
    echo "$first_line" && cat && printf "\033&l0H" &&
exit 0
fi

exit 2
```

最后， 需要告知 LPD 所使用的□□器， 通□ `if` □□完成:

```
:if=/usr/local/libexec/ifhp:
```

□可以□入 `lpr plain.text` 和 `lpr whatever.ps`， 它□都□□可以成功打印。

10.4.1.4. □□□□器

在完成了 打印机□□□置 □□中所描述的内容之后， □一件事 恐怕就是□□喜□的格式的文件安装□□□□器了 (除了□ ASCII 文本)。

10.4.1.4.1. □什□安装□□□□器?

□□□□器使打印□多格式的文件□得很容易。 比如， 假□我□大量使用 TeX 排版系□， 并且有一台 PostScript® 打印机。 □次从 TeX 生成一个 DVI 文件， 我□都不能直接打印它直到我□将 DVI 文件□□成 PostScript®。

□□的命令□□是下面的□子:

```
% dvips seaweed-analysis.dvi
% lpr seaweed-analysis.ps
```

通□安装 DVI 文件的□□□□器， 我□可以跳□□次手□□□□一□， 而□ LPD 来完成□个□□。 □在， □次要打印 DVI 文件， 我□只需要一□就可以打印它：

```
% lpr -d seaweed-analysis.dvi
```

我□要 LPD □□ DVI 文件是通□指定 -d □□完成的。 格式和□□ □□ □一□列出了所有的□□□□。

□于□□想要打印机支持的□□， 首先要安装 *f*□□□□器 然后在 /etc/printcap 中指定它的路径。 在□□打印□置中， □□□□器□似于文本□□器 (□□ 安装文本□□器) 不同的是它不是用来打印□文本， 而是将一个文件□□成打印机能□理解的格式。

10.4.1.4.2. 我□□安装□个□□□□器？

□□□安装□希望使用的□□□□器。 如果要打印很多 DVI 数据， 就需要 DVI □□□□器； 如果有大量的 troff 数据，就□□安装 troff □□器。

下面的表格□□了可以与 LPD配合 工作的□□器， 以及它□在 /etc/printcap文件中的□量名， □有如何在 lpr命令中□用它□：

文件□型	在**/etc/printcap**文件中的□量名	在**lpr**命令中□用使用的参数
cifplot	cf	-c
DVI	df	-d
plot	gf	-g
ditroff	nf	-n
FORTRAN text	rf	-f
troff	tf	-f
raster	vf	-v
plain text	if	none, -p, or -l

在例子中， lpr -d就是指 打印机需要在/etc/printcap文件中 df□量所指的□□器。

不管□人□□□， 像 FORTRAN 的文本 和 plot □些格式已□基本不用了。 所以在□的机器上，就可以安装其他的□□器来替□□些参数原有的意□。 例如， 假□想要能直接打印 Printerleaf 文件 (由 Interleaf desktop publishing 程序生成)， 而且不打算打印 plot 文件， 就可以安装一个 Printerleaf □□□□器并且用 gf □量指定它。 然后就可以告□□的用□使用 lpr -g 就可以 "打印 Printerleaf 文件。"

10.4.1.4.3. 安装□□□□器

以□安装的□□□□器不是 FreeBSD 基本系□的一部分， 所以它□可能是在 /usr/local 目□下。 通常目□ /usr/local/libexec 是保存它□的地方， 因□它□通常是通□ LPD □行的； 普通用□□□并不需要直接□行它□。

要□用一个□□□□器， 只需要在 /etc/printcap 文件中□目□打印机中合□的□量□上□□器所在的路径。

在接下来的例子当中， 我□将□ 一台叫做 bamboo 的打印机添加一个□□□□器。 下面是□个例子的 /etc/printcap 文件， 其中使用新□量 df 来□打印机 bamboo □置□□□□器：

```
#
#  /etc/printcap for host rose - added df filter for bamboo
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:sd=/var/spool/lpd/rattan:\
        :lp=/dev/lpt0:\
        :if=/usr/local/libexec/if-simple:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:\
        :lp=/dev/ttyu5:ms#-parenb cs8 clocal crtscts:rw:\
        :if=/usr/local/libexec/psif:\
        :df=/usr/local/libexec/psdf:
```

□里的 DVI □□器是一段 shell 脚本， 名字叫做 /usr/local/libexec/psdf。 下面是它的代□：

```
#!/bin/sh
#
#  psdf - DVI to PostScript printer filter
#  Installed in /usr/local/libexec/psdf
#
# Invoked by lpd when user runs lpr -d
#
exec /usr/local/bin/dvips -f | /usr/local/libexec/lprps "$@"
```

□段脚本以□□器模式□行 dvips (参数 -f) 并从□准□入□取要打印的任□。 然后□行 PostScript® 文本□□器 lprps (□□ 在 PostScript® 打印机上打印□文本任□ □一□)， 并且□着 LPD □□脚本的全部参数。 lprps 工具将利用□些参数来□打印□行□□。

10.4.1.4.4. 更多□□□□器□用□例

因□安装□□□□器的□□并不是固定的， 所以□□介□了一些可行的例子。 在以后的安装配置□程中可以以□些例子□参考。 甚至如果合□的□， 可以完全照搬□去。

□段例子中的脚本是一个 Hewlett Packard LaserJet III-Si 打印机的光□格式数据 (□□上也就是 GIF 文件)：

```
#!/bin/sh
#
#  hpvf - Convert GIF files into HP/PCL, then print
#  Installed in /usr/local/libexec/hpvf

PATH=/usr/X11R6/bin:$PATH; export PATH
giftopnm | ppmtopgm | pgmtopbm | pbmtolj -resolution 300 \
    && exit 0 \
    || exit 2
```

它的工作原理就是将 GIF 文件□□成 portable anymap， 再□□成 portable graymap， 然后再□□成 portable bitmap， 最后再□□成 LaserJet/PCL- 兼容的数据。

下面是□打印机配置上上述□□器的 /etc/printcap 文件：

```
#
#  /etc/printcap for host orchid
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:\
        :lp=/dev/lpt0:sh:sd=/var/spool/lpd/teak:mx#0:\
        :if=/usr/local/libexec/hpif:\
        :vf=/usr/local/libexec/hpvf:
```

下面的脚本是一个在名叫 bamboo 的□台 PostScript® 打印机上打印用 groff 排版□件生成的 troff 数据的打印□□器：

```
#!/bin/sh
#
#  pstf - Convert groff's troff data into PS, then print.
#  Installed in /usr/local/libexec/pstf
#
exec grops | /usr/local/libexec/lprps "$@"
```

上面□段脚本□是用 lprps 来与打印机□行通□。 如果打印机是接在并口上的， 那□就□□使用下面的□段脚本：

```
#!/bin/sh
#
#  pstf - Convert groff's troff data into PS, then print.
#  Installed in /usr/local/libexec/pstf
#
exec grops
```

□里是我□要□用□□器需要在 /etc/printcap 里□加的内容：

```
:tf=/usr/local/libexec/pstf:
```

下面的例子也□会□□多 FORTRAN 老手羞愧。 它是一个 FORTRAN- 文本 的□□器， 能在任意一台 可以打印□文本的打印机上使用。 我□将□打印机 teak 安装□个□□器：

```
#!/bin/sh
#
# hprf - FORTRAN text filter for LaserJet 3si:
# Installed in /usr/local/libexec/hprf
#

printf "\033&k2G" && fpr && printf "\033&l0H" &&
 exit 0
exit 2
```

然后我□要在 /etc/printcap 中□打印机能□ teak □用□个□□器添加下面的内容：

```
:rf=/usr/local/libexec/hprf:
```

最后， 再□出一个有些□□的例子。 我□将□以前介□□的 teak □台激光打印机添加一个 DVI □□器。 首先，最容易的部分： 更新 /etc/printcap 加入 DVI □□器的路径：

```
:df=/usr/local/libexec/hpdf:
```

□在， □困□的部分了： □写□□器。 □了□□□□器， 我□需要一个 DVI-到-LaserJet/PCL □□程序。 FreeBSD Ports Collection (□□ Ports Collection □一□) 中有一个： print/dvi2xx。 安装□个 port 就会得到我□需要的程序，dvilj2p， 它可以将 DVI 数据□□成 LaserJet IIp， LaserJet III， 和 LaserJet 2000 兼容的数据。

dvilj2p 工具使得□□器 hpdf □得十分□□， 因□ dvilj2p 不能□取□准□入。 它需要从文件中□取数据。 更糟□的是， □个文件的名字必□以 .dvi □尾。 所以使用 /dev/fd/0 作□□准□入是有□□的。 我□可以通□□接 (符号□接) 来解决□个□□。 □接一个□□的文件名 (一个以 .dvi □尾的文件名) 到 /dev/fd/0， 从而□制 dvilj2p 从□准□入□取。

□在迎面而来的是□外一个□□， 我□不能使用 /tmp 存放□□□接。 符号□接是被用□和□ bin □有的。 而□□器□是以 daemon 用□□行的。 并且 /tmp 目□□置了 sticky 位。 所以□□器只能建立符号□接，但它不能在用完之后清除掉□些□接。 因□它□属于不同的用□。

所以□□器将在当前工作目□下建立符号□接， 即后台打印□列目□ (用□量 sd 在 /etc/printcap 中指定)。□是一个非常好的□□□器完成它工作的地方， 特□□是因□ (有□) □个目□比起 /tmp 来有更多的可用磁□空□。

最后， □出□□器的代□：

```
#!/bin/sh
#
#  hpdf - Print DVI data on HP/PCL printer
#  Installed in /usr/local/libexec/hpdf

PATH=/usr/local/bin:$PATH; export PATH
```

```
#
#  Define a function to clean up our temporary files.  These exist
#  in the current directory, which will be the spooling directory
#  for the printer.
#
cleanup() {
   rm -f hpdf$$.dvi
}

#
#  Define a function to handle fatal errors: print the given message
#  and exit 2.  Exiting with 2 tells LPD to do not try to reprint the
#  job.
#
fatal() {
    echo "$@" 1>&2
    cleanup
    exit 2
}

#
#  If user removes the job, LPD will send SIGINT, so trap SIGINT
#  (and a few other signals) to clean up after ourselves.
#
trap cleanup 1 2 15

#
#  Make sure we are not colliding with any existing files.
#
cleanup

#
#  Link the DVI input file to standard input (the file to print).
#
ln -s /dev/fd/0 hpdf$$.dvi || fatal "Cannot symlink /dev/fd/0"

#
#  Make LF = CR+LF
#
printf "\033&k2G" || fatal "Cannot initialize printer"

#
#  Convert and print.  Return value from dvilj2p does not seem to be
#  reliable, so we ignore it.
#
dvilj2p -M1 -q -e- dfhp$$.dvi

#
#  Clean up and exit
#
cleanup
```

```
exit 0
```

10.4.1.4.5. 自□□□： 一□替代□□□□器的方法

以上□些□□□□器基本上建成了□的打印□境， 但也有不足就是必□由用□来指定 (在 lpr(1) 命令行中) 要使用□一个□□器。 如果□的用□不是□□算机很在行， 那□□用□□器将是一件麻□的事情。 更糟的是， 当□□器□定的不正□□， □□器被用在了不它□□□型的文件上， 打印机也□会□出上百□□。

比只安装□□□□器更好的方法， 就是□文本□□器 (因□它是默□的□□器) 来□□要打印文件的□型， 然后自□□行正□的□□□□器。 像 file □□的工具可以□我□一定的□助。 当然， 要区分□ 有些 文件的□型□是有困□的 - 但是，当然， □可以□□它□提供□□□□器。

FreeBSD 的 Ports 套件提供了一个可以自□□行□□的文本□□器， 名字叫做 apsfilter (print/apsfilter)。它可以□□□文本、 PostScript®、 DVI 以及几乎任何格式的文件， 并在□行相□的□□之后完成打印工作。

10.4.1.5. □出□□器

LPD 后台打印系□□支持一□我□□没有□□□的□□器： □出□□器。 □出□□器只是用来打印□文本的， □似于文本□□器， 但□化了□多地方。 如果□正在使用□出□□器而不是文本□□器， 那□：

- LPD □整个任□□□一个□出□□器， 而不是□任□中的□个文件都□□一次。
- LPD 不会提供任□中文件□始和□束的信息□□出□□器。
- LPD 不会提供用□名或者主机名□□□器， 所以它是无法做打印□□的。 事□上它只有□个参数：

  □□器-名字 -w□度 -l□度

  □度 来自于 pw □量， 而 *length* 来自于 pl □量， □些□都是□□□□中□打印机□置的。

不要□□出□□器的□化所耽□。 如果想要□出□□器完成□任□中的□个文件都重新□始一□打印是 不可能 的。□使用文本□□器 (也叫□入□□器)； □□ 安装文本□□器。 此外， □□上， □出□□器 更□□ ， 它要□□□□它的字□流中是否有特殊的□志字符， 并且□自己□送信号来代替 LPD 的。

可是， 如果打算要□□□或者需要□送控制字符或者其他的初始化字符串来完成打印□□□， 那□□出□□器□是必需的。 (但是它也是 无用的 如果打算□打印的用□□□， 因□ LPD 不会□□出□□器任何用□或者主机的信息。)

在一台□个的打印机上， LPD 同□允□□出□□器、 文本□□器和其他的□□器。 在某些情况下， LPD 将□会□□□出□□器来打印□□□ (□□ □□□)。 然后 LPD 会要求□出□□器 自己停止□行 ， 它□送□□□器□个字□： ASCII 031跟着一个 ASCII 001。 当□出□□器看□□□个字□ (031, 001)， 它□□通□□送 SIGSTOP 信号来停止自己的□行。当 LPD 已□□行好了其他的□□器， 它会通□□□出□□器□送 SIGCONT 信号来□□出□□器重新□行。

如果□有一个□出□□器而 没有 文本□□器， 并且 LPD 正在□理一个□文本任□， LPD 会使用□出□□器来完成□个任□。 像以前□行一□， □出□□器会按□序打印任□中的文件， 而不会□入送□或其他□□的命令， 但□也□并 不是□想要的□果。 在大多数情况下， □□是需要一个文本□□器。

lpf □个我□前面介□□的文本□□器程序， 也可以用来做□出□□器。 如果需要使用快速且混乱的□出□□器，但又不想写字□□□和信号□送代□， 那□□□ lpf。 lpf 也可以包含在一个 shell 脚本中来□理任何打印机可能需要的初始化代□。

10.4.1.6. lpf： 一个文本□□器

/usr/libexec/lpr/lpf □个程序包含在 FreeBSD 的二□制程序中， 它是一个文本□□器 (□入□□器)。 它可以□排□出 (用 lpr -i 命令提交的任□)， 可以打印控制字符禁止断□用 lpr -l 提交的任□)， 可以□整任□中退格和制表符打印的位置， □可以□打印□行□□。 它同□可以像□出□□器一□工作。

lpf □用于很多打印□境。 尽管它本身没有向打印机□送初始化代□的功能， 但写一个 shell 脚本来完成所需的初始化并□行 lpf 是很容易的。

□了□ lpf 可以正□的□行打印□□， 那□需要 /etc/printcap 中的 pw 和 pl □量都填入正□的□。 它用□些□来□定一□能打印多少文本， 并□算出任□有多少□。 想得到更多□于打印□□的信息， □参□ □打印机使用□行□□。

10.4.2. □□□

如果□有 很多 用□， 他□正在使用各式各□的打印机， 那□□或□要考□一下把 □□□ 当作无可避免之□□了。

□□□， 也叫 *banner* 或者 *burst* □， 可以用来辨□打印出的文件是□打印的。 它□通常用大号的粗体字母打印出来， 也可能用装□□□□四周， 所以在一堆打印出的文件中， 突出的□示了□个文件属于□个用□的□个任□。 □可以□用□快速的□到他□的任□。 而□□□一个明□的缺点就是， 在□个任□中都要有一□或者几□□作□□□□印出来， 可是它□的有用的地方只□□几分□的作用， 最后它□会被放□回收站或者□□□□堆。 (注意□□□只是一个任□一个， 而不是任□中的□个文件都有一个， 所以可能□□□□不算很浪□。)

LPD 系□可以自□□□的打印提供□□□， 如果 □的打印机可以直接打印□文本。 如果□的打印机是一台 PostScript® 打印机， □将需要一个外部的程序来生成□□□； □□ 在 PostScript® 打印机上打印□□□。

10.4.2.1. 打□□□□

在 □□打印□置 □□， 我□通□在 /etc/printcap 文件中指定 sh ("禁止□□□") 来把□□□功能□掉了。 要重新□打印机□□□□□功能， 只需要□除掉 sh。

听起来很容易， 不是□？

是的。 □ 可能 不得不□□出□□器来□打印机□送初始化字符串。 下面是一个用在 Hewlett Packard PCL-兼容打印机上的□出□□器的例子：

```
#!/bin/sh
#
#  hpof - Output filter for Hewlett Packard PCL-compatible printers
#  Installed in /usr/local/libexec/hpof

printf "\033&k2G" || exit 2
exec /usr/libexec/lpr/lpf
```

用 of □量指定□出□□器的路径。 参□ □出□□器 □一□来得到更多信息。

下面是一个□我□以前介□的叫做 teak 的打印机配置的 /etc/printcap 文件； 在配置当中我□□□了□□□并且加入了上述的打印□□器：

```
#
#  /etc/printcap for host orchid
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:\
        :lp=/dev/lpt0:sd=/var/spool/lpd/teak:mx#0:\
        :if=/usr/local/libexec/hpif:\
        :vf=/usr/local/libexec/hpvf:\
        :of=/usr/local/libexec/hpof:
```

□在， 当用□再□任□□打印机 teak 的□候， □个任□都会有一个□□□。 如果用□想要花□□来□□他□自己打印的文件， 那□他□可以通□ lpr -h 命令来提交任□； 参考 □□□□□ □一□来得到更多□于 lpr(1) 的□□。

LPD 在□□□之后□出一个□□字符。 如果□的打印机使用一个不同的字符或者字符串当作退□指令， 在 /etc/printcap 中用 ff □量指定即可。

10.4.2.2. 控制□□□

通□□用□□□， LPD 将生成出一个 □□□， 一整□的大字母， □着用□， 主机和任□名。 下面是一个例子 (kelly 从主机 rose 打印了一个叫做 "outline" 的任□)：

```
      k                 ll       ll
      k                  l        l
      k                  l        l
      k   k     eeee     l        l     y    y
      k  k     e    e    l        l     y    y
      k k      eeeeee    l        l     y    y
      kk k     e         l        l     y    y
      k   k    e    e    l        l     y   yy
      k    k    eeee     lll      lll    yyy y
                                           y
                                          y    y
                                           yyyy

                                ll
                       t         l        i
                       t         l
       oooo    u    u   ttttt    l       ii     n nnn     eeee
      o    o   u    u     t      l        i     nn   n   e    e
      o    o   u    u     t      l        i     n    n   eeeeee
      o    o   u    u     t      l        i     n    n   e
      o    o   u   uu     t  t    l        i     n    n   e    e
       oooo     uuu u      tt     lll      iii    n    n    eeee

      r rrr     oooo     ssss     eeee
      rr   r   o    o   s    s   e    e
      r        o    o    ss      eeeeee
      r        o    o      ss    e
      r        o    o   s    s   e    e
      r         oooo     ssss     eeee


                                        Job:  outline
                                        Date: Sun Sep 17 11:04:58 1995
```

LPD 会附加一个□□符在□段文本之后， 所以任□会在新的一□上□始 (除非□置了 `sf` (禁止□□) 在 /etc/printcap 文件里目□打印机的□□中)。

如果□喜□， LPD 可以生成一个 短□□； 指定 `sb` (短 banner) 在文件 /etc/printcap 中。 □□□就会看起来像下面□□：

```
rose:kelly  Job: outline  Date: Sun Sep 17 11:07:51 1995
```

同□是默□的， LPD 也是先打印□□□， 然后才是任□。 要想反□来， 在 /etc/printcap 中指定 `hl` (最后□□)。

10.4.2.3. □□□□□的任□□□

使用 LPD 内置的□□□会在□行打印□□的□候□生一□特殊情况： □□□肯定是 免□ 的。

□什□？

因□□出□□器是□有的一个在打印□□□□能□行□□的外部程序， 但却没有提供□它任何 用□或者主机 的信息或者□□文件， 所以它无法知道□□□□打印机的使用付□。 如果□□是 "□加一□□数" □文本□□器或者其他□□器 (它□有用□和主机的信息) 是不□的， 因□用□可以用 lpr -h 命令跳□□□□。 他□是需要□自己并没有打印的□□□付□。 基本上， lpr -h 是明知用□的首□， 但也不能□制□□人使用它。

□□个□□器生成自己的□□□ (因此可以□它□□□) 是 仍然不□的。 如果用□想要用 lpr -h 命令禁止□□□， 它□将仍然印出□□□并且□它□付□。 因□ LPD 不会把 -h □个参数□□任何□□器。

□□， □□□□□□？

□可以：

- □可 LPD 的□个□□， 并且免□提供□□□打印。
- 安装一个替代 LPD 的□件， 比如 LPRng。 参考 替□□准的后台打印□件 来得到更多□于可以替代 LPD 的□件的信息。
- 写一个 □明的 □出□□器。 通常， □出□□器不□□去完成除了初始化打印机或者□行一些□□字符□□以外的任何事情。 它□合完成□□□和□文本任□ (当没有文本 (□入) □□器□)。 但是， 如果有文本□□器□□文本任□服□， 那□ LPD 将□□打印□□□□□□出□□器。 而且， □个□出□□器可以理解□□□里 LPD 生成的信息，然后决定□位用□和主机□□□□□□付□。 □□方法□有的□□是□出□□器仍然不知道□□使用什□□□文件 (af□量的内容并没有被□□□来)， 但是如果□有一个□所周知的□□文件， 就可以直接把文件名写□□出□□器。 □了□化解□□□的□□， 我□定□ sh (短□□) □量在 /etc/printcap 文件中。 但□些□是太麻□了， 而且用□也更喜□□他□免□打印□□□的慷慨的系□管理□。

10.4.2.4. 在 PostScript® 打印机上打印□□□

像上面描述的那□，LPD 可以生成一个□文本的□□□来□□多□打印机。 当然， PostScript® 不能直接打印□文本， 所以 LPD 没什□用-或者□大多□候是□□。

一个□而易□的方法来得到□□□就是□□个□□□□器和文本□□器都来生成□□□。 □些□□器□□用用□名和主机的参数来生成一个相□□的□□□。 □□方法的缺点就是用□□是打印出□□□， 无□他□是否用 lpr -h 命令来提交的任□。

□我□来深入深入的研究一下□个方法。 下面的脚本□入三个参数 (用□登□名， 主机名， 和任□名) 然后生成一个□□的 PostScript® □□□：

```
#!/bin/sh
#
#  make-ps-header - make a PostScript header page on stdout
#  Installed in /usr/local/libexec/make-ps-header
#

#
#  These are PostScript units (72 to the inch).  Modify for A4 or
#  whatever size paper you are using:
#
page_width=612
page_height=792
border=72
```

```
#
#  Check arguments
#
if [ $# -ne 3 ]; then
    echo "Usage: `basename $0` <user> <host> <job>" 1>&2
    exit 1
fi

#
#  Save these, mostly for readability in the PostScript, below.
#
user=$1
host=$2
job=$3
date=`date`

#
#  Send the PostScript code to stdout.
#
exec cat <<EOF
%!PS

%
%  Make sure we do not interfere with user's job that will follow
%
save

%
%  Make a thick, unpleasant border around the edge of the paper.
%
$border $border moveto
$page_width $border 2 mul sub 0 rlineto
0 $page_height $border 2 mul sub rlineto
currentscreen 3 -1 roll pop 100 3 1 roll setscreen
$border 2 mul $page_width sub 0 rlineto closepath
0.8 setgray 10 setlinewidth stroke 0 setgray

%
%  Display user's login name, nice and large and prominent
%
/Helvetica-Bold findfont 64 scalefont setfont
$page_width ($user) stringwidth pop sub 2 div $page_height 200 sub moveto
($user) show

%
%  Now show the boring particulars
%
/Helvetica findfont 14 scalefont setfont
/y 200 def
[ (Job:) (Host:) (Date:) ] {
200 y moveto show /y y 18 sub def }
```

```
forall

/Helvetica-Bold findfont 14 scalefont setfont
/y 200 def
[ ($job) ($host) ($date) ] {
        270 y moveto show /y y 18 sub def
} forall

%
% That is it
%
restore
showpage
EOF
```

□在， □个□□□□器和文本□□器都能□用□段脚本来生成□□□， 然后打印用□的任□。 下面是我□早些□候在□个文□中提到的 DVI □□□□器， 被修改之后来生成一个□□□：

```
#!/bin/sh
#
#  psdf - DVI to PostScript printer filter
#  Installed in /usr/local/libexec/psdf
#
#  Invoked by lpd when user runs lpr -d
#

orig_args="$@"

fail() {
    echo "$@" 1>&2
    exit 2
}

while getopts "x:y:n:h:" option; do
    case $option in
        x|y)  ;; # Ignore
        n)    login=$OPTARG ;;
        h)    host=$OPTARG ;;
        *)    echo "LPD started `basename $0` wrong." 1>&2
              exit 2
              ;;
    esac
done

[ "$login" ] || fail "No login name"
[ "$host" ] || fail "No host name"

( /usr/local/libexec/make-ps-header $login $host "DVI File"
  /usr/local/bin/dvips -f ) | eval /usr/local/libexec/lprps $orig_args
```

□□器是□□解□参数列表来决定用□名和主机名的。 解□的方法□于其他□□□□器来□也是一□的。 尽管文本□□器需要□入的参数有些小的不同， (参□ □□器是□□工作的)。

像我□以前提到的那□， 上面的配置， 尽管相当□□， □掉了 "禁止□□□" 的□□ (-h □□) 在 lpr 中。 如果用□想要保□□木 (或者是几便士， 如果□□打印□□□收□的□)， 它□不能完成□件事情， 因□□个□□器都要□□个任□打印一个□□□。

要允□用□□于□个任□都可以□□□□□， □需要使用在 □□□□□□ □□中介□的那□技巧： 写一个□出□□器来解□ LPD-生成的□□□并且生成一个 PostScript® 的版本。 如果用□用 lpr -h 命令提交任□， 那□ LPD 将不会生成□□□，并且□出□□器也不会生成□□□。 否□， □出□□器将从 LPD □取文本， 然后□送□当的□□□的 PostScript® □□□打印机。

如果□有的是一台□在串口上的 PostScript® 打印机， □可以使用 lprps 里的一个□出□□器， psof ，它可以完成上述任□。 但注意 psof 不□□□□□□。

10.4.3. 网□打印

FreeBSD 支持网□打印： □送任□□□程打印机。 网□打印通常指□□不同的方式：

- □□一台□接在□程主机上的打印机。 在一台主机上安装一台常□的串口或并口打印机。 然后， □置 LPD 来通□网□□□其他主机上的打印机。 具体□ 安装在□程主机上的打印机 □□。
- □□一台直接□接在网□上的打印机。 打印机□有一个网□接口 (或者替代常□的串口或者并口)。□□的打印机可能像下面□□工作：
  - 它或□可以理解 LPD 的□□， 并且甚至可以接收□程主机□来的任□排□□列。 □□，它就像一个普通的主机□行着 LPD 一□。 做在 安装在□程主机上的打印机 里介□的□□， 可以□置好□□的打印机。
  - 它或□支持网□数据流。 □□， 把打印机 "接" 在一台网□上的主机上， 由□台主机□□安排任□并□送任□到打印机。 参□ □网□数据流接□的打印机 □□来得到更多安装□□打印机的建□。

10.4.3.1. 安装在□程主机上的打印机

LPD 后台打印系□内建了□□其他也□行着 LPD (或者是与 LPD 兼容的) 的主机□送任□的功能。 □个功能使□可以在一台主机上安装打印机， 并□它可以在其他主机上□□。 □个功能同□□用在那些有网□接□并且可以理解 LPD □□的打印机上。

要□□□□程打印的功能， 首先在一台主机上安装打印机， 就是 打印服□器， 可以使用在 □□打印机□置 □□中□□□置的方法。 高□的□置可以参考 高□打印机□置 □□中□需要的部分。 一定要□□一下打印机，看看它是不是所有□□□的 LPD 的功能都正常工作。 此外□需要□□ 本地主机 允□使用 □程主机 上的 LPD 服□ (参□ 限制□程主打印任□)。

如果□正在使用一台□网□接□并与 LPD 兼容的打印机， 那□我□那下面□□中的 打印服□器 就是打印机本身，而 打印机名 就是□□打印机配置的名字。 参考随打印机和/或者打印机-网□接□供□的文□。

如果□正使用惠普的 Laserjet， □打印机名 text 将自□地□□完成 LF 到 CRLF 的□□，因而也就不需要 hpif 脚本了。

然后， 在□外一台□想要□□打印机的主机上的 /etc/printcap 文件中加入它□的□□， 像下面□□：

1. 可以随意□□个□□起名字。 □□起□， □可以□打印服□器使用相同的名字或者□名。

2. 保留 lp □量□空， (:lp=:)。
3. 建立一个后台打印□列目□， 并用 sd □量指明其位置。 LPD 将把任□提交□打印服□器之前， 会把□些任□保存在□里。
4. 在 rm □量中放入打印服□器的名字。
5. 在 rp 中放入打印服□器上打印机的名字。

就是□□。 不需要列出□□□□器， □面大小， 或者其他的一些□西在 /etc/printcap 文件中。

□有一个例子。 主机 rose 有□台打印机， bamboo 和 rattan。 我□要□主机 orchid 的用□可以使用□□台打印机。 下面是 /etc/printcap 文件， 用在主机 orchid (□□ □□□□□) 上的。 文件中已□有了打印机 teak 的□□； 我□在主机 rose 上□加了□台打印机：

```
#
#  /etc/printcap for host orchid - added (remote) printers on rose
#

#
#  teak is local; it is connected directly to orchid:
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:\
        :lp=/dev/lpt0:sd=/var/spool/lpd/teak:mx#0:\
        :if=/usr/local/libexec/ifhp:\
        :vf=/usr/local/libexec/vfhp:\
        :of=/usr/local/libexec/ofhp:

#
#  rattan is connected to rose; send jobs for rattan to rose:
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :lp=:rm=rose:rp=rattan:sd=/var/spool/lpd/rattan:

#
#  bamboo is connected to rose as well:
#
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :lp=:rm=rose:rp=bamboo:sd=/var/spool/lpd/bamboo:
```

然后， 我□只需要在主机 orchid 上建立一个后台打印□列目□：

```
# mkdir -p /var/spool/lpd/rattan /var/spool/lpd/bamboo
# chmod 770 /var/spool/lpd/rattan /var/spool/lpd/bamboo
# chown daemon:daemon /var/spool/lpd/rattan /var/spool/lpd/bamboo
```

□在， 主机 orchid 上的用□可以打印到 rattan 和 bamboo 了。 如果， 比如， 一个用□在主机 orchid 上□入了：

```
% lpr -P bamboo -d sushi-review.dvi
```

LPD 系□在主机 orchid 上会□制□个任□到后台打印□列目□ /var/spool/lpd/bamboo 并且□下□是一个 DVI 任□。 当主机 rose 上的打印机 bamboo 的后台打印□列目□有空□的□， □□个 LPD 系□将会□□□个文件到主机 rose 上。 文件将排在主机 rose 的□列中直到最□被打印出来。 它将被从 DVI □□成 PostScript® (因□ bamboo 是一台 PostScript® 打印机) 在主机 rose。

10.4.3.2. □有网□数据流接口的打印机

通常， 当□□打印机□□了一□网□， 可以得到□个版本： 一个是模□后台打印 (□一些的版本)， 或者一个只□送数据□打印机就像在使用串口或者并口一□ (便宜一些的版本)。 □□□述如何使用□个便宜一些的版本。 要得到□一些版本的更多信息， 参□前面章□ 安装在□程主机上的打印机。

/etc/printcap 文件的格式□□指定使用□个串口或并口， 并且□要指定 (如果□正在使用串口)， 使用多快的波特， 是否使用流量控制， □制表符延□， □□□行， 等等。 但是没有一□方法指定一个□接到一台正在□听 TCP/IP 的或者其他网□接口的打印机。

要□送数据到网□打印机， 就需要□□一个通□程序， 它可以被文本或者□□□□器□用。 下面是一些例子： 脚本 netprint 将□准□入的所有数据□送到一个□在网□上的打印机。 我□将打印机的名字作□第一个参数， 端口号跟在后面作□第二个参数， □□ netprint。 注意它只支持□向通□ (FreeBSD 到打印机)； 很多网□打印机支持双向通□， 并且□是□可能利用到的 (得到打印机状□， □行打印□□， 等等的□候。)。

```
#!/usr/bin/perl
#
#  netprint - Text filter for printer attached to network
#  Installed in /usr/local/libexec/netprint
#
$#ARGV eq 1 || die "Usage: $0 <printer-hostname> <port-number>";

$printer_host = $ARGV[0];
$printer_port = $ARGV[1];

require 'sys/socket.ph';

($ignore, $ignore, $protocol) = getprotobyname('tcp');
($ignore, $ignore, $ignore, $ignore, $address)
    = gethostbyname($printer_host);

$sockaddr = pack('S n a4 x8', &AF_INET, $printer_port, $address);

socket(PRINTER, &PF_INET, &SOCK_STREAM, $protocol)
    || die "Can't create TCP/IP stream socket: $!";
connect(PRINTER, $sockaddr) || die "Can't contact $printer_host: $!";
while (<STDIN>) { print PRINTER; }
exit 0;
```

然后我□就可以在多□□器里使用□个脚本了。 加入我□有一台 Diablo 750-N 行式打印机□在网□上。 打印机在 5100 端口上接收要打印的数据。 打印机的主机名是 scrivener。 □里是□□个打印机写的文本□□器：

```
#!/bin/sh
#
#  diablo-if-net - Text filter for Diablo printer `scrivener' listening
#  on port 5100.   Installed in /usr/local/libexec/diablo-if-net
#
exec /usr/libexec/lpr/lpf "$@" | /usr/local/libexec/netprint scrivener 5100
```

10.4.4. 限制打印机的使用

□□将□述□于限制打印机使用的□□。 LPD 系□□□可以控制□可以□□打印机， 无□本地或是□程的， 是否他□可以打印机多□副本， 任□可以有多大， 以及打印□列的尺寸等。

10.4.4.1. 限制多□副本

LPD 系□能□□化用□在打印多□副本□的工作。 用□可以用 `lpr -#5` (□例) 来提交打印任□， □会将任□中□个文件都打印五□副本。 □是不是一件很棒的事情□。

如果□感□多□副本会□打印机造成不必要的磨□和□耗， □可以屏蔽掉 lpr(1) 的 `-#` □□， □可以通□在 /etc/printcap 文件中□加 `sc` □量来完成。 当用□用 `-#` □□提交任□□， 他□将看到：

```
lpr: multiple copies are not allowed
```

注意当□一台□程打印机□行□置□ (参□ 安装在□程主机上的打印机 □一□) □□需要同□在□程主机的 /etc/printcap 文件中 □加`sc` □量， 否□用□□是可以从其他主机上提交使用多□副本的任□。

下面是一个例子。 □个是 /etc/printcap 文件在主机 `rose` 上。 打印机 `rattan` 非常□□， 所以我□将允□多□副本， 但是激光打印机 `bamboo` □有些忙， 所以我□禁止多□副本， 通□□加 `sc` □量：

```
#
#  /etc/printcap for host rose - restrict multiple copies on bamboo
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:sd=/var/spool/lpd/rattan:\
        :lp=/dev/lpt0:\
        :if=/usr/local/libexec/if-simple:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:sc:\
        :lp=/dev/ttyu5:ms#-parenb cs8 clocal crtscts:rw:\
        :if=/usr/local/libexec/psif:\
        :df=/usr/local/libexec/psdf:
```

□在， 我□□需要□机 `sc` □量在主机 `orchid` 的 /etc/printcap 文件中 (□便我□也禁止打印机 `teak` 多□打印) ：

```
#
#  /etc/printcap for host orchid - no multiple copies for local
#  printer teak or remote printer bamboo
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:\
        :lp=/dev/lpt0:sd=/var/spool/lpd/teak:mx#0:sc:\
        :if=/usr/local/libexec/ifhp:\
        :vf=/usr/local/libexec/vfhp:\
        :of=/usr/local/libexec/ofhp:

rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :lp=:rm=rose:rp=rattan:sd=/var/spool/lpd/rattan:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :lp=:rm=rose:rp=bamboo:sd=/var/spool/lpd/bamboo:sc:
```

通□使用 sc □量， 我□阻止了 lpr -# 命令的使用， 但仍然没有禁止用□多次□行 lpr(1) ， 或者多次提交任□中同□的文件， 像下面□□：

```
% lpr forsale.sign forsale.sign forsale.sign forsale.sign forsale.sign
```

□里有很多□方法可以阻止□□行□ (包括忽略它)， 并且是免□的。

10.4.4.2. 限制□打印机的□□

□可以控制□可以打印到□台打印机通□ UNIX® 的□机制和文件 /etc/printcap 中的 rg □量。 只要把可以□□打印机的用□放□□当的□中， 然后在 rg □量中写上□的名字。

如果□□以外的用□ (包括 root) □□打印到被限制的打印机，将会得到□□的提示：

```
lpr: Not a member of the restricted group
```

像使用 sc (禁止多□副本) □量一□， □需要指定 rg 在□程同□□打印机有□□限制的主机上， 如果□感□合□的□ (参考 安装在□程主机上的打印机 □一□)。

比如， 我□将□任何人都可以□□打印机 rattan， 但只有在 artists □中的人可以使用打印机 bamboo。 □里是□似的主机 rose 上的 /etc/printcap 文件：

```
#
#  /etc/printcap for host rose - restricted group for bamboo
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:sd=/var/spool/lpd/rattan:\
        :lp=/dev/lpt0:\
        :if=/usr/local/libexec/if-simple:

bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:sc:rg=artists:\
        :lp=/dev/ttyu5:ms#-parenb cs8 clocal crtscts:rw:\
        :if=/usr/local/libexec/psif:\
        :df=/usr/local/libexec/psdf:
```

Let us leave the other example /etc/printcap file (for the host orchid) alone. Of course, anyone on orchid can print to bamboo. It might be the case that we only allow certain logins on orchid anyway, and want them to have access to the printer. Or not.

□里□台□能有一个限制的□。

10.4.4.3. 控制提交的任□大小

如果□有很多用□□□打印机， 可能需要□用□可以提交的文件尺寸□置一个上限。 □竟， 文件系□中后台打印□列目□的空□是有限的， □需要保□□里有空□来存放其他用□的任□。

LPD 允□通□使用 mx □量来限制任□中文件的最大字□数， 方法是指定□位□□的 BUFSIZ 数， □□表示 1024 字□。 如果在□个□量的□是 0， □表示不□行限制； 不□， 如果不指定 mx □量的□， □会使用默□□ 1000 □。

□个限制是□于任□中 文件 的， 而 不是 任□□共的大小。

LPD 不会拒□比限制大小大的文件。 但它是将限制大小以内的部分排入□列， 并且打印出来的只有□些。 剩下的部分将被□□。 □个行□是否正□□需□□。

□我□来□例子打印机 rattan 和 bamboo □加限制。 由于那些 artists 的 PostScript® 文件可能会很大， 我□将限制大小□ 5 兆字□。 我□将不□□文本行式打印机做限制：

```
#
#  /etc/printcap for host rose
#

#
#  No limit on job size:
#
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:mx#0:sd=/var/spool/lpd/rattan:\
        :lp=/dev/lpt0:\
        :if=/usr/local/libexec/if-simple:

#
#  Limit of five megabytes:
#
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:sc:rg=artists:mx#5000:\
        :lp=/dev/ttyu5:ms#-parenb cs8 clocal crtscts:rw:\
        :if=/usr/local/libexec/psif:\
        :df=/usr/local/libexec/psdf:
```

同□， 限制只□本地用□起作用。 如果□置了允□□程用□使用□的打印机， □程用□将不会受到□些限制。 □也需要指定 `mx` □量在□程主机的 /etc/printcap 文件中。 参□ 安装在□程主机上的打印机 □一□来得到更多有□□程打印的信息。

除此之外， □有□一□限制□程任□大小的方法； 参□ 限制□程主机打印任□。

10.4.4.4. 限制□程主机打印任□

LPD 后台打印系□提供了多□方法来限制从□程主机提交的任□：

主机限制

□可以控制本地 LPD 接收□台□程主机□来的□求， 通□ /etc/hosts.equiv 文件和 /etc/hosts.lpd 文件。 LPD □看是否到来的任□□求来自被□□个文件中列出的主机。 如果没有， LPD 会拒□□个□求。

□些文件的格式非常□□： □行一个主机名。 注意 /etc/hosts.equiv 文件也被 ruserok(3) □□使用， 并影□着 rsh(1) and rcp(1) 等程序， 所以要小心。

□个例子， 下面是 /etc/hosts.lpd 文件在主机 `rose` 上：

```
orchid
violet
madrigal.fishbaum.de
```

意思是主机 `rose` 将接收来自 `orchid`， `violet`， 和 `madrigal.fishbaum.de` 的□求。 如果任何其他的主机□□□□主机 `rose` 的 LPD， 任□将被拒□。

大小限制

□可以控制后台打印□列目□需要保留多少空□。 建立一个叫做 minfree 的文件在后台打印□列目□下□本地打印机。 在□个文件中□入一个数字来代表多少磁□□数 (512 字□) 的剩余空□来接收□程任□。

□□□可以保□□程用□不会填□□的文件系□。 □也可以用它来□本地用□一个□先： 他□可以在磁□剩余空□低于 minfree 文件中的指定□后仍然可以提交任□。

比如， □我□□加一个 minfree 文件□打印机 bamboo。 我□□□ /etc/printcap 文件来□到□个打印机的后台打印□列目□； □里是打印机 bamboo 的□□：

```
bamboo|ps|PS|S|panasonic|Panasonic KX-P4455 PostScript v51.4:\
        :sh:sd=/var/spool/lpd/bamboo:sc:rg=artists:mx#5000:\
        :lp=/dev/ttyu5:ms#-parenb cs8 clocal crtscts:rw:mx#5000:\
        :if=/usr/local/libexec/psif:\
        :df=/usr/local/libexec/psdf:
```

后台打印□列目□在 sd □量中□出。 我□□置 3 兆字□ (6144 磁□□) □文件系□上必□存在的□共剩余空□， □ LPD 可以接受□程任□：

```
# echo 6144 > /var/spool/lpd/bamboo/minfree
```

用□限制

□可以控制□些□程用□可以打印到本地打印机， 通□指定 rs □量在 /etc/printcap 文件中。 当 rs 出□在一个本地打印机的□□中□， LPD 将接收来自□程主机 并 在本地有同□登□名的用□提交的任□。 否□， LPD 会拒□□个任□。

□个功能在一个 (比如) 有□多部□共享一个网□的□境中特□有用， 并且有些用□可以越□部□的□界。 通□□他□在□的系□上建立□号， 他□可以他□自己的部□的系□里使用□的打印机。 如果 只 允□他□□的打印机，而不是□的□算机□源， □可以□他□ "象征" □□， 不□主目□并且□置一个没用的 shell， 比如 /usr/bin/false。

10.4.5. □打印机使用□□

当然， □需要□打印付□。 □什□不？ □□和墨水都需要花□的。 并且□里□有□□的□用 - 打印机是由很多部件□装成的， 并且零件会坏掉。 □可以□□□的打印机， 使用形式， 和□□□用来得出□□ (或者□尺， □米， 或者□什□) 的□用。 □在， □□□□□打印□□□？

好了， 坏消息是 LPD 后台打印系□在□个部分没有提供很多□助。 □□是一个□使用的打印机的□□，打印的格式， 和 □的 在□打印机的使用□□的需求依□性很高的。

要□□□□， □必□更改打印机的文本□□器 (□□文本任□□□) 和□□□□器 (□其他格式的文件□□)， 要□□□数或者□□打印了多少□的□。 □不可以通□使用□□的□出□□器来逃脱□□， 因□它不能□行□□。 参□ □□器 □□。

通常， 有□□方法来□行□□：

- 定期□□ 是更常用的方法， 可能因□它更□□。 无□合□何人打印一个任□， □□器都将□□用□名， 主机名，和打印的□数到一个□□文件。 □个月， 学期， 年， 或者任何□想□定的□□段， 收集□些不同打印机上的□□文件， 按用□□打印的□数□行□算， 并□使用□行付□。 然后□掉所有□□文件， □始一个新的□□周期。

- □□□□ 不太常用， 可能因□它比□□。 □□方法□□□器□用□的打印□行□□的□□。 像磁□配□， □□是□□的。 □可以□□用□打印当他□的□□超□的□候， 并且可能提供一□方法□用□□□并□整他□的 "打印配□。" 但□个方法需要一些数据□代□来跟踪用□和他□的配□。

LPD 后台打印系□□□□方法都支持且很□□： 所以□需要提供□□器 (大多数□候)， □要提供□□代□。 但□好的方面是： □可以有非常□活的□□方法。 比如， □可以□□使用□段□□□是□□□□。 □可以□□□□□些信息： 用□名， 主机名， 任□□型， 打印□数， 使用了多少平方尺的□， 任□打印了多□□□， 等等。 □可以通□修改□□器来存□□些信息。

10.4.5.1. 快速并且混乱的打印□□

FreeBSD 包含□个可以□□立刻可以建立起□□的□段□□的程序。 它□是文本□□器 lpf， 在 lpf： 一个文本□□器□□中描述， 和 pac(8)， 一个收集并□□打印机□□文件中□□的程序。

像在前面章□提到的□□器一□ (□□器)， LPD □□文本或者□□□□器并在□□器命令行里□上□□文件的名字。 □□器可以使用□个参数知道□往□写□□□□。 □个文件的名字来自于 af □量在 /etc/printcap 文件里， 并且如果没有指定□□路径， □默□是相□于后台打印□列目□的。

LPD □□ lpf □着□□和□□的参数 (通□ pw 和 pl □量)。 lpf 使用□些参数来判定将使用多少□□。 在文件□送到打印机之后， 它就会在□□文件中写入□□。 □□像下面□个□子：

```
2.00 rose:andy
3.00 rose:kelly
3.00 orchid:mary
5.00 orchid:mary
2.00 orchid:zhang
```

□□□□□个打印机都使用一个独立的□□文件， 像 lpf 就没有内建文件□□□， □□□个 lpf 可能会□生彼此□□混合的情况， 如果它□同□要在同一个文件写入内容的□候。 一个最□□的保□□个打印机都使用一个独立的□□文件的方法就是将 af=acct 写在 /etc/printcap 文件中。 然后， □个打印机的□□文件都会在□台打印机的后台打印□列目□中， 文件的名字叫做 acct。

当□准□□用□的打印□行收□□， □行 pac(8) 程序。 只要□□到要收集信息的□台打印机的后台打印□列目□， 然后□入 pac。 □将会得到一个美元□□的摘要像下面□□：

```
  Login               pages/feet   runs    price
orchid:kelly               5.00     1   $  0.10
orchid:mary               31.00     3   $  0.62
orchid:zhang               9.00     1   $  0.18
rose:andy                  2.00     1   $  0.04
rose:kelly               177.00   104   $  3.54
rose:mary                 87.00    32   $  1.74
rose:root                 26.00    12   $  0.52

total                    337.00   154   $  6.74
```

□些是 pac(8) 需要的参数：

`-P` 打印机

□台 打印机 要□□。 □个□□□在用 `af` □量在 /etc/printcap 文件中指定了□□路径的情况下起作用。

`-c`

以金□来排序□出来代替以用□名字字母排序。

`-m`

忽略□□文件中的主机名。 □上□个□□， 用□ `smith` 在主机 `alpha` 上与同□的用□ `smith` 在主机 `gamma` 上一□。 不□□个□□的□， 他□□是不同的用□。

`-p` □价

使用 *price* 作□□□或□尺美元的□价来替代 `pc` □量指定的□价在 /etc/printcap 文件中， 或者□分 (默□)。 *price* 可以用一个浮点数来指定。

`-r`

反向排序。

`-s`

建立一个□□摘要文件， 并且截短□□文件。

名字 ...

只打印指定 名字 用□的□□信息。

在 pac(8) 默□□生的摘要中， 可以看到在不同主机上的□个用□打印了多少□。 如果在□□里， 主机不考□ (因□用□可以使用任何主机)， □行 `pac -m`， 来得到下面的摘要：

```
  Login               pages/feet   runs    price
andy                        2.00     1   $  0.04
kelly                     182.00   105   $  3.64
mary                      118.00    35   $  2.36
root                       26.00    12   $  0.52
zhang                       9.00     1   $  0.18

total                     337.00   154   $  6.74
```

要以美元□算□付□数， pac(8) 指定 `pc` □量在 /etc/printcap 文件中 (默□是 200， 或者 2 分□□). □个参数的□位是百分之一分， 在□个□量中指定□□或者□尺的价格。 □可以覆□□个□当□行 pac(8) □着参数 `-p` 的□候。 参数 `-p` 的□位是美元， 而不是百分之一分。 例如，

```
# pac -p1.50
```

□定□□的价格是 1 美元 5 美分。 □可以通□□个□□来□到目□利□。

最□， □行 `pac -s` 将存□□些信息在一个□□文件里， 文件名和打印机□□的名字相同， 但是□着 `_sum` 的后□。 然后截短□□文件。 当□再次□行 pac(8) 的□候， 它再次□取□□文件来得到初始的□□， 然后在□□文件中□加信息。

10.4.5.2. □□□打印的□数□行□数？

□了□行□程的精□□，需要判断一个任□将会消耗多少□□。□是打印□□□□的□□。

□于□文本任□，□个□□不是太□解决：□任□中的行数□行□数然后与打印机支持的□□行数□行比□。□忘了也□添印的行，或者很□的□□上的一行但在打印机上会折成□行的□□□行□□。

文本□□器 `lpf` (在 lpf：一个文本□□器 □□中介□) 会在□□□考□□些□□。如果正在□写一个可以□行□□的文本□□器，□可能需要□看 `lpf` 的源代□。

□□□理其他格式的文件？

好，□于 DVI- 到 -LaserJet 或者 DVI- 到 -PostScript® □□，可以□□的□□器□出□断信息，□于 `dvilj` 或者 `dvips` 命令，并且看到多少□被□□了。□也□可以□于其他□型的文件和□□程序□行□似操作。

但是□些方法的弱点就是事□上打印机并不是打印了所有的□。比如，□□，缺墨，或者炸掉了 - 但用□□是要□没有打印的部分付□。

□□□□做？

只有一条 肯定 的方法来□行 精□ 的□□。 □□一台可以告□□它使用了多少□的打印机， 并且将它□接到串口或者网□上。几乎所有 PostScript® 打印机都支持□个小功能。其他制造厂或其他型号也可以有□个功能 (比如 Imagen 激光网□打印机)。□□些打印机更改□□器使它在打印完□个任□之后接收□□用量，并□基于□个□□行□□。不需要□算行数，也不需要容易出□的文件□□。

当然，□也□是可以大方的使打印免□。

10.5. 使用打印机

□□将□述如何使用在 FreeBSD 下□置好的打印机。下面是一个用□□命令的□□：

lpr(1)
打印任□

lpq(1)
□□打印□列

lprm(1)
从打印机的□列中移除任□

□有一个管理命令，lpc(8)，在 管理打印机 一□中有所介□，它可以用于控制打印机及其□列。

lpr(1), lprm(1), and lpq(1) □三个命令都接受 `-P printer-name` □□来指定□□个打印机 / □列□行操作，在 /etc/printcap 文件中列出的打印机。□允□□提交，□除，并□□任□在多个打印机上。如果□不使用 `-P` □□，那□□些命令会使用在 □境□量 `PRINTER` 中指定的打印机。最□，如果□也没有 `PRINTER` □个□境□量，□些命令的默□□是叫做 `lp` 的□台打印机。

从此以后，□□ 默□打印机 就是指 `PRINTER` □境□量中指定的□台，或者叫做 `lp` 的□一台当没有□境□量 `PRINTER` 的□候。

10.5.1. 打印任□

要打印文件，□入：

```
% lpr filename ...
```

□个命令会打印所有列出的文件到默□打印机。 如果没有列出文件， lpr(1) 会从□准□入□取打印数据。 比如，□个命令打印一些重要的系□文件：

```
% lpr /etc/host.conf /etc/hosts.equiv
```

要□□一个指定的打印机，□入：

```
% lpr -P printer-name filename ...
```

□个例子打印一个当前目□的□□的列表到叫做 rattan 的□台打印机：

```
% ls -l | lpr -P rattan
```

因□没有□ lpr(1) 命令列出文件， lpr 从□准□入□入数据， 在□里是 ls -l 命令的□出。

lpr(1) 命令同□可以接受多□控制格式的□□， □用文件□□， 生成多□副本， 等等。 要得到更多信息， 参考打印□□ □□。

10.5.2. □□任□

当使用 lpr(1) □行打印□， □希望打印的所有数据被放在一起打包成了一个 "打印任□"， 它被□送到 LPD 后台打印系□。 □台打印机都有一个任□□列， 并且□的任□在□列中等待其他用□的其他任□打印。 打印机按照先来先印的□□打印□些任□。

要□示默□打印机的□列， □入 lpq(1)。 要指定打印机， 使用 -P □□。 例如， 命令

```
% lpq -P bamboo
```

会□示打印机 bamboo 的□列。 下面是命令 lpq □出的一个例子：

```
bamboo is ready and printing
Rank    Owner    Job  Files                              Total Size
active kelly    9    /etc/host.conf, /etc/hosts.equiv   88 bytes
2nd    kelly    10   (standard input)                   1635 bytes
3rd    mary     11   ...                                78519 bytes
```

□里□示了□列中有三个任□在 bamboo 中。 第一个任□， 用□ kelly 提交的， □□ "任□□号" 9。 □个要打印的任□都会□得一个不同的任□□号。 大多□候可以忽略□个任□□号， 但在□需要取消任□□会用到□个号□； 参考

移除任□ □□得到更多信息。

□号□ 9 的任□包含了□个文件； 在 lpr(1) 命令行中指定的多个文件被看作是一个□个的任□。 它是当前激活的任□ (注意□个□ 激活 在 "Rank" □列下面)， 意思是打印机当前正在打印那个任□。 第二个任□包含了□准□入□□ lpr(1) 命令的数据。 第三个任□来自用□ mary； ， 它是一个比□大的任□。 □要打印的文件的路径名太□了， 所以 lpq(1) 命令只□示了三个点。

lpq(1) □出的□一行也很有用： 它告□我□打印机正在做什□ (或者至少是 LPD □□打印机□□正在做的)。

lpq(1) 命令同□支持 -l □□来生成一个□□的□列表。 下面是一个 lpq -l 命令的例子：

```
waiting for bamboo to become ready (offline ?)
kelly: 1st              [job 009rose]
       /etc/host.conf                   73 bytes
       /etc/hosts.equiv                 15 bytes

kelly: 2nd              [job 010rose]
       (standard input)                 1635 bytes

mary: 3rd                               [job 011rose]
      /home/orchid/mary/research/venus/alpha-regio/mapping 78519 bytes
```

10.5.3. 移除任□

如果□□一个打印任□改□了主意， 可以用 lprm(1) 将任□从□列中□除。 通常， □甚至可以用 lprm(1) 命令来移除一个当前激活的任□， 但是任□的一部分或者所有□是可能打印出来。

要从默□打印机中移除一个任□， 首先使用 lpq(1) □到任□□号。 然后□入：

```
% lprm job-number
```

要从指定打印机中□除任□， □加 -P □□。 下面的命令会□除□号□ 10 的任□从 bamboo □台打印机：

```
% lprm -P bamboo 10
```

lprm(1) 命令有一些快捷方式：

lprm -

□除所有属于□的任□ (默□打印机的)。

lprm *user*

□除所有属于用□ *user* 的任□ (默□打印机的)。 超□用□可以□除用□的任□； □只可以□除自己的任□。

lprm

命令行中不□任□□号， 任□名， 或者 - □□， lprm(1) 会□除默□打印机上当前激活的任□， 如果它属于□。 超□用□可以□除任□激活的任□。

使用参数 -P 和上面的快捷方式来用指定打印机替代默□打印机。 例如， 下面的命令会□除当前用□在打印机 rattan □列中的所有任□：

```
% lprm -P rattan -
```

如果□正工作在一个网□□境中， lprm(1) 将只允□在提交任□的主机上□除任□， 甚至是同一台打印机也可以在其他主机上使用□。 下面的命令□明了□个：

```
% lpr -P rattan myfile
% rlogin orchid
% lpq -P rattan
Rank   Owner      Job  Files                         Total Size
active seeyan     12    ...                           49123 bytes
2nd    kelly      13   myfile                         12 bytes

% lprm -P rattan 13
rose: Permission denied

% logout
% lprm -P rattan 13
dfA013rose dequeued
cfA013rose dequeued
```

10.5.4. 超越□文本：打印□□

lpr(1) 支持□多控制文本格式的参数， □□□形和其他格式文件， 生成多□副本， □理任□， 等等。 □一□将描述 □些□□。

10.5.4.1. 格式与□□□□

下面的 lpr(1) 参数控制任□中文件的格式。 使用□些参数， 如果任□不含□文本， 或者□想□□文本通□ pr(1) 格式化。

例如， 下面的命令打印一个 DVI 文件 (来自 TeX 排版系□) 文件名□ fish-report.dvi 到打印 bamboo：

```
% lpr -P bamboo -d fish-report.dvi
```

□些□□□用到任□中的□个文件， 所以□不能混合 (□) DVI 和 ditroff 文件在同一个任□中。 替代的方法是， 用独立的任□提交□些文件， 使用不同的□□□□□不同的任□。

所有□些□□除了 -p 和 -T 都需要□□□□器安装□目□打印机。 例如， -d □□需要 DVI □□□□器。 参考 □□□□器 □□得到更多□□。

-c

打印 cifplot 文件。

`-d`

打印 DVI 文件。

`-f`

打印 FORTRAN 文本文件。

`-g`

打印 plot 数据。

`-i number`

□□ *number* 列； 如果没有指定 *number*， □□□ 8 列。 □个□□□可以工作在某些□□器上。

不要在□□ `-i` 和数字之□加入空格。

`-l`

打印文字数据， 包括控制字符。

`-n`

打印 ditroff (无□□依□ troff) 数据。

-p

打印之前用 pr(1) 格式化□文本。 参考 pr(1) 得到更多信息。

`-T title`

使用 *title* 在 pr(1) 上来替代文件名。 □个□□□在使用 `-p` □□□起作用。

`-t`

打印 troff 数据。

`-v`

打印 raster 数据。

下面是一个例子： □个命令打印了一个很好的 ls(1) □机手册到默□打印机：

```
% zcat /usr/shared/man/man1/ls.1.gz | troff -t -man | lpr -t
```

zcat(1) 命令解□□ ls(1) 的手册并且将内容□□ troff(1) 命令， 它将格式化□些内容并且生成 GNU troff □出□ lpr(1) ， 它提交任□到 LPD 后台打印。 因□使用了 `-t` □□□ lpr(1) ， 后台打印将会□□ GNU troff □出到默□打印机可以理解的格式当任□被打印□。

10.5.4.2. 任□□理□□

下面的 lpr(1) □□告□ LPD □任□特殊□理：

-# copies

生成 *copies* 个副本□任□中的□个文件， 替代□个文件一□副本。 管理□可以禁止□个□□来□少打印机的浪□和鼓励□印机的使用。 参考 限制多□副本。

□个例子打印三□副本的文件 parser.c 跟着三□副本的文件 parser.h 到默□打印机：

```
% lpr -#3 parser.c parser.h
```

-m

打印完成后□信。 使用□个□□， LPD 系□将会□送□件到□的□□， 当它完成了□理□的任□后。 在信中，它将会告□□任□是否成功完成或者出□了□□， 并且 (通常) 指明是什□□□。

-s

不要□制文件到后台打印□列目□， 要使用符号□接。

如果□正在打印一个很大的任□， □可能需要□个□□。 它□省后台打印□列目□的空□ (□的任□可能使后台打印□列目□所在的文件系□剩余空□超出)。 它同□也□省了□□， 因□ LPD 将不会副本任□的□个字□到后台打印□列目□。

□也有一个缺点： 因□ LPD 将直接指向源文件， □不能修改或者□除它□直到它□被打印出来。

如果□打印到一台□程打印机， LPD 将最□将文件从本地主机副本到□程主机上， 所以□□ `-s` 只能□省本地后台打印□列目□的空□， 而不是□程的。 □然如此， 但它□是很有用。

-r

移除任□中的文件在它□被□制到后台打印□列目□之后， 或者在用 `-s` □□打印它□之后。 □慎使用□个□□！

10.5.4.3. □□□□□

□些 lpr(1) 的□□□整了通常出□在任□□□□上的文本。 如果□□□被跳□了在目□打印机上， □些□□将不会起作用。参考 □□□ 得到更多□于□置□□□的信息。

-C *text*

替□□□□上的主机名□ *text*。 主机名通常都是提交任□的主机名称。

-J *text*

替□□□□上的任□名□ *text*。 任□名通常是任□中□一个文件的名字， 或者 stdin 如果□正在打印□准□入。

-h

不打印任何□□□。

在某些地点， □个□□可能无效， 与□□□的□生方法有□。 参考 □□□ 得到□□信息。

10.5.5. 管理打印机

作□一个打印机的管理者， □必□要安装， □置， 并且□□它□。 使用 lpc(8) 命令，□可以与打印机以更多的方式交流。 用 lpc(8)， □可以

- □□或停止打印机
- □用或禁止它□的□列
- 重新安排□个□列中的任□。

首先，　一个□于□□的解□：　如果一个打印机被　停止　了，　它将不会打印它□列中的任何□西。　但用□□是可以提交任□，　它□会在□列中等待直到打印机被 □□ 或者□列被清空。

如果一个□列被　禁止，　没有用□（除了　root）可以提交任□到打印机。　一个　□用　的□列允□任□被提交。一个打印机可以被 □□ 但它的□列被禁止，　在□□情况下打印机将打印□列中的任□，　直到□列□空。

通常，　□必□有　root　□限来使用　lpc(8)　命令。　普通用□可以使用　lpc(8)　命令来□得打印机状□并且重□一台挂了的打印机。

□里是一个□于　lpc(8)　命令的摘要。　大部分命令□着一个　*printer-name*　参数来知道要□□台打印机操作。□可以用 `all` 填在 *printer-name* 的位置来代表所有在 /etc/printcap 文件中列出的打印机。

`abort printer-name`

取消当前任□并停止打印机。 用□仍然可以提交任□，　如果□列□是□用的。

`clean printer-name`

从打印机的后台打印□列目□移除旧的文件。　有□，　□成任□的文件没有被　LPD　正□的□除，　特□是在打印中出□□□或者管理活□比□多的□候。 □个命令□□不属于后台打印□列目□的文件并□除它□。

`disable printer-name`

禁止新任□入□。 如果打印机正在工作，　它将会□□打印□列中剩余的任□。 超□用□ (`root`) □是可以提交任□，甚至提交到一个禁止的□列。

□个命令在□□一台新打印机或者安装□□器□非常有用：　禁止□列并提交以　root　提交任□。　其他用□将不能提交任□直到□完成了□□并用命令 `enable` 重新□用了□列的□候。

`down printer-name message`

打印机下□。 等于 `disable` 命令后跟一个 `stop` 命令。 *message* 将作□打印机状□，　当用□使用 lpq(1) 或者 `lpc status` 命令□看打印机□列状□的□候□示出来。

`enable printer-name`

□打印机□□□列。 用□可以提交任□到打印机但是在打印机□□之前不会打印出任何□西。

`help command-name`

打印□于 *command-name* 命令的□助。 不□ *command-name*，　□打印可用命令的摘要。

`restart printer-name`

□□打印机。　普通用□可以使用□个命令，　当一些特□的□境□致　LPD　□死□，　但他□不能□用一台使用　stop　或者 `down` 命令停用的打印机。 `restart` 命令等同于 `abort` 后跟着一个 `start`。

`start printer-name`

□用打印机。 打印机将□始打印□列中的任□。

`stop printer-name`

停止打印机。　打印机将完成当前任□并且将不再打印□列中的任□任□。　尽管打印机被停用，　但用□仍然可以提交任□到一个□□的□列。

`topq printer-name job-or-username`

重新以　*printer-name*　安排□列，　通□将列出的　*job*　□号或者指定的所属　*username*　的任□放在

□列的最前面。 □于□个命令， □不可以使用 `all` 当作 *printer-name*。

`up printer-name`

打印机上□； 相□于 `down` 命令。 等同于 `start` 后跟着一个 `enable` 命令。

lpc(8) 的命令行接受上面的命令。 如果□不□入任何命令， lpc(8) □□入一个交互模式， 在□里□可以□入命令直到□入 `exit`， `quit`， 或者文件□束符。

10.6. 替□□准后台打印

如果□已□通□□了□个手册， 那□到□在□□□已□了解了□于 FreeBSD 包含的后台打印系□ LPD 的一切。 □可能□□了它很多的缺点， 它□很自然的□□提出□□的□□： "□里□有什□后台打印系□□ (并且可以工作在 FreeBSD 上)？"

LPRng

LPRng， 它的意思是 "LPR： 下一代"， 是一个完全重写的 PLP。 Patrick Powell 和 Justin Mason (PLP □□的主要□□人) 合作完成了 LPRng。 LPRng 的主站是 http://www.lprng.org/。

CUPS

CUPS， 通用 UNIX 打印系□， 提供了一个□便的打印□□ UNIX®-基□的操作系□。 它是由 Easy Software Products □□的， 并且成□了 UNIX® 供□商和用□的□准打印解决方案。

CUPS 使用 Internet 打印□□ (IPP) 作□管理打印任□和□列的基□。 行式打印机守□程序 (LPD) 服□器消息□ (SMB)， 和 AppSocket (a.k.a. JetDirect) □□的部分功能也被支持。 CUPS □加了基于□□网□打印机和 PostScript 打印机描述 (PPD) 的打印□□来支持 UNIX® 下的真□打印。

CUPS 的主站是 http://www.cups.org/。

HPLIP

HPLIP， HP Linux® 成像及打印系□ (Imaging and Printing system)， 是一套由 HP □□的用于支持 HP 的打印、 □描和□真□□的工具。 □套程序利用 CUPS 打印系□作□后端来提供一些打印方面的功能。

HPLIP 的主□位于 http://hplipopensource.com/hplip-web/index.html。

10.7. 疑□□□

在使用 lptest(1) □行□□的□□之后， □可能得到了下面的□果， 而不是正□的□果：

□了一会儿， 它工作了； 或者， 它没有退出一整□□。

打印机□行了打印， 但在□之前它呆了一段而且什□都没做。 事□上， □可能需要按一下打印机上的打印剩余 或者 送□ 按□来□□果出□。

如果□是□□所在， 打印机可能在等待， 看看在打印之前， □的任□是否□有更多的数据。 要修正□个□□， □可以□文本□□器□送一个送□字符 (或者其他需要的) 到打印机。 □通常足□□打印机立即打印出内部□存内剩余的文本。 它同□可以用来□保□个任□的□尾都占用一整□□， □□下一个任□才不会在前一个任□最后一□□的中□□始。

接下来的 shell 脚本 /usr/local/libexec/if-simple 的脚本打印了一个送□符在它□送任□到打印机之后：

```
#!/bin/sh
#
# if-simple - Simple text input filter for lpd
# Installed in /usr/local/libexec/if-simple
#
# Simply copies stdin to stdout.  Ignores all filter arguments.
# Writes a form feed character (\f) after printing job.

/bin/cat && printf "\f" && exit 0
exit 2
```

它的□出□生了 "楼梯效果"。

□可能在□上得到下面□些：

```
!"#$%&'()*+,-./01234
               "#$%&'()*+,-./012345
                              #$%&'()*+,-./0123456
```

□也成□了 楼梯效果 的受害者， □是由□新行的□志字符的解□不一致造成的。 UNIX® □格的操作系□使用一个□个字符： ASCII □ 10， 即□行 (LF)。 MS-DOS®， OS/2®， 和其他的系□使用一□儿字符，ASCII □ 10 和 ASCII □ 13 (回□ CR)。 □多打印机使用 MS-DOS® 的□□来代表新行。

当□在 FreeBSD 上打印□， □的文本□用了□行字符。 打印机， 打印机看到□行字符后， 走一行□， 但□光□位置□是在□□□上要打印的下一个字符□。 □就是回□的作用： 将下一个要打印的字符的位置移到□□的左□□。

□里是 FreeBSD 想要打印机做的：

打印机收到 CR	打印机打印 CR
打印机收到 LF	打印机打印 CR + LF

下面有几□完成□个的□法：

- 使用打印机的配置□□或者控制面板来更改它□□些字符的解□。 □看打印机的手册来□到□□更改。

如果□引□□的系□到其他除了 FreeBSD 之外的操作系□， □可能不得不 重新配置打印机使用 □个操作系□□ CR 和 LF 字符的解□。 □可能更喜□下面□□一□解决方案。

- □ FreeBSD 的串口□□自□□□ LF 到 CR+LF。 当然， □ □□ 工作在串口打印机上。 要□□□个功能， 定□ `ms#` □量并 □置 `onlcr` 模式在 /etc/printcap 文件中相□打印机□。
- □送一个 □□□ 到打印机来□它□□□ LF 字符做不同的□理。 参考□的打印机手册来了解□的打印机支持□些□□□。 当□□到合□的□□□， 修改文本□□器□其先□送□个□□□， 然后再□送打印任□。

  □里是一个□□得 Hewlett-Packard PCL □□□的打印机□写的文本□□器。 □个□□器使得打印机将 LF 作□一个 LF 和一个 CR 来□待； 然后它□送任□； 最后□送一个送□符□出任□的最后一□□。 它□□可以在几乎所有 Hewlett Packard 打印机上工作。

```
#!/bin/sh
#
# hpif - Simple text input filter for lpd for HP-PCL based printers
# Installed in /usr/local/libexec/hpif
#
# Simply copies stdin to stdout.  Ignores all filter arguments.
# Tells printer to treat LF as CR+LF.  Ejects the page when done.

printf "\033&k2G" && cat && printf "\033&l0H" && exit 0
exit 2
```

下面是一个 /etc/printcap 文件的例子在叫做 orchid 的主机上。 它只有一台打印机□接在第一个并口上， 一台 Hewlett Packard LaserJet 3Si 名字叫做 teak。 它使用上面那段脚本作□文本□□器：

```
#
#  /etc/printcap for host orchid
#
teak|hp|laserjet|Hewlett Packard LaserJet 3Si:\
        :lp=/dev/lpt0:sh:sd=/var/spool/lpd/teak:mx#0:\
        :if=/usr/local/libexec/hpif:
```

行行覆□。

打印机从来不□□□行。 所有的文本都打印在□一行文本的上面。

□个□□是 "相反" 于楼梯效果， 像上面描述的那□， 并且更少□。 一些地方， LF □个 FreeBSD 用来□束一行的字符被作□ CR □个将打印位置返回到□的左□的字符□待。 而没有向下走□一行。

使用打印机的配置□□或者控制面板来□制□ LF 和 CR □行下面的□□：

打印机收到	打印机打印
CR	CR
LF	CR + LF

打印□掉字符。

当打印□， □行里打印机都□掉一些字符没有打。 □个□□可能随着打印的□行越□□重， □掉越来越多的字符。

□个□□是由打印机跟不上□算机通□串口□送数据的速度造成的 (□个□□□□不会□生在并口打印机上)。 有□□方法能克服□个□□：

- 如果打印机支持 XON/XOFF 流量控制， 那就□ FreeBSD 使用它， 通□加入 ixon 模式在 ms# □量里。
- 如果打印机支持□求/清除硬件握手信号 （通常□ RTS/CTS）， 指定 crtscts 模式在 ms# □量里。 并且要□定□接打印机和□算机的□是支持硬件流量控制的。

它打印出□□。

打印机打印出的□西看起来是一些随机的字符， 而不是想要打印的□西。

□通常意味着□一□串口打印机通□参数□置不正□的□□。 □□ br □量中□定的波特， 和 ms# 中的校□□置；□定打印机也在使用和 /etc/printcap 文件中相同的□置。

没有反□。

如果没有反□， □□就可能出在 FreeBSD 而不是硬件上了。 □加日志文件 (lf) □量到 /etc/printcap 文件里出□□□的打印机的□□□。 比如， 下面是打印机 rattan 的□□， 使用了 lf □量：

```
rattan|line|diablo|lp|Diablo 630 Line Printer:\
        :sh:sd=/var/spool/lpd/rattan:\
        :lp=/dev/lpt0:\
        :if=/usr/local/libexec/if-simple:\
        :lf=/var/log/rattan.log
```

然后， 再次打印。 □□日志文件 (在我□的例子当中， 是 /var/log/rattan.log □个文件) 来看是否有□□信息出□。 根据出□的信息， □着来修正□□。

如果□没有指定 lf □量， LPD 会使用 /dev/console 作□默□□。

Chapter 11. Linux® 二□制兼容模式

11.1. 概述

FreeBSD 提供了与 Linux® 32-bit 二□制兼容， 允□用□在 FreeBSD 系□上安装和□行大多数的 32-bit Linux® 二□制程序而无需做任何修改。 据□在某些情况下， FreeBSD 上□行的 32-bit Linux® 二□制程序能有更好的表□。

然而， 仍然有一些 Linux® 操作系□特有的功能在 FreeBSD 上并不被支持。 例如， 要是 Linux® 程序□度地使用了□如□用虚□ 8086 模式 i386™ 特有的□用， □无法在 FreeBSD 上□行。 □外， 目前□不支持 64-bit 的 Linux® 二□制程序。

□完□章，□将了解到：

- 如何在 FreeBSD 系□中□用 Linux® 二□制兼容模式。
- 如何安装□外的 Linux® 共享□。
- 如何在 FreeBSD 上安装 Linux® □用程序。
- FreeBSD 上 Linux® 兼容模式的□□□□。

在□□□章之前，□□□知道：

- 知道如何安装 □外的第三方□件。

11.2. 配置 Linux® 二□制兼容模式

默□情况下， Linux® □并没有被安装而且 Linux® 二□制兼容模式也没有被□□。 Linux® □可以通□手□安装或者使用 FreeBSD 的 Ports Collection。

安装 emulators/linux-base-f10 包或者 port 是最容易在 FreeBSD 系□上□得一套基本的 Linux® □的方法。使用如下方法安装 port：

```
# cd /usr/ports/emulators/linux_base-f10
# make install distclean
```

安装完成以后， 加□ linux 模□□用 Linux® 二□制兼容模式：

```
# kldload linux
```

□看模□是否已□被加□：

```
% kldstat
Id Refs Address    Size     Name
 1    2 0xc0100000 16bdb8   kernel
 7    1 0xc24db000 d000     linux.ko
```

在 /etc/rc.conf 中加入以下□行后 Linux® 兼容模式便会在系□□□□自□□□：

```
linux_enable="YES"
```

想要在自制内核中静□□接 Linux® 二□制兼容支持的用□可以在自定□的内核配置文件中加入 options COMPAT_LINUX。 然后按照 配置FreeBSD的内核 中所描述的方法□□并安装新内核。

11.2.1. 手□安装□外的□

在配置了 Linux® 兼容模式之后， 如果某个 Linux® □用程序依然提示□不到共享□， 需先□出此 Linux® 二□制程序需要的共享□再手□安装。

在 Linux® 系□上使用 ldd □出□用程序所需的共享□文件。 比如， 在安装有 Doom 的 Linux® 系□上□行如下的命令列出 linuxdoom 所需用到的共享□文件：

```
% ldd linuxdoom
libXt.so.3 (DLL Jump 3.1) => /usr/X11/lib/libXt.so.3.1.0
libX11.so.3 (DLL Jump 3.1) => /usr/X11/lib/libX11.so.3.1.0
libc.so.4 (DLL Jump 4.5pl26) => /lib/libc.so.4.6.29
```

然后把上面□出中最后一列中的所有文件从 Linux® 系□□制到 FreeBSD 上的 /compat/linux。□制完成之后， 建立指向第一□中文件名的符号□接。 □□在 FreeBSD 系□上将会有如下的文件：

```
/compat/linux/usr/X11/lib/libXt.so.3.1.0
/compat/linux/usr/X11/lib/libXt.so.3 -> libXt.so.3.1.0
/compat/linux/usr/X11/lib/libX11.so.3.1.0
/compat/linux/usr/X11/lib/libX11.so.3 -> libX11.so.3.1.0
/compat/linux/lib/libc.so.4.6.29
/compat/linux/lib/libc.so.4 -> libc.so.4.6.29
```

如果已□有了一个与 ldd □出中第一列的主修□号相同的 Linux® 共享□文件， □不再需要□制最后那列文件，□有的共享□□□可以正常使用。 如果是更新版本的共享□通常建□□制。 只要有符号□接指向新的版本， 那□就可以□除旧版的了。

比如， FreeBSD 系□中□有□些共享□文件：

```
/compat/linux/lib/libc.so.4.6.27
/compat/linux/lib/libc.so.4 -> libc.so.4.6.27
```

并且 ldd 指出某个二□制程序需要之后版本：

```
libc.so.4 (DLL Jump 4.5pl26) -> libc.so.4.6.29
```

既然□有文件最后的版本号只相差一到□个版本， 程序□□可以正常使用□旧些的版本。 不管□□，使用新版本替□□有 libc.so 都是安全的。

```
/compat/linux/lib/libc.so.4.6.29
/compat/linux/lib/libc.so.4 -> libc.so.4.6.29
```

通常最初几次在 FreeBSD 上安装 Linux® 程序□需要□□ Linux® 二□制程序所依□的共享□文件。 在此之后，系□里便会有足□多的 Linux® 共享□文件来□行新安装的 Linux® 二□制程序而无需□外操作。

11.2.2. 安装 Linux® ELF 二□制程序

ELF 二□制程序有□需要□外的□□。 当未被□□的 ELF 二□制程序被□行的□候， 会生成如下的□□信息：

```
% ./my-linux-elf-binary
ELF binary type not known
Abort
```

□了□助 FreeBSD 内核分辨 FreeBSD ELF 二□制程序和 Linux® 二□制程序， □使用 brandelf(1)：

```
% brandelf -t Linux my-linux-elf-binary
```

由于□在的 GNU 工具□能自□把□当的□□信息写入 ELF 二□制程序中，□个□□通常不是必□做的。

11.2.3. 安装基于 Linux® RPM 的□用程序

安装基于 Linux® RPM 的□用程序， 首先需要安装 archivers/rpm 包或者 port。 安装好之后 root 用□就能使用此命令安装 .rpm 了：

```
# cd /compat/linux
# rpm2cpio < /path/to/linux.archive.rpm | cpio -id
```

如有必要的□使用 brandelf □□安装好的 ELF 二□制程序。 注意此□安装将无法干□卸□。

11.2.4. 配置主机名解析器

如果 DNS 不能正常工作或是出□以下的□□信息：

```
resolv+: "bind" is an invalid keyword resolv+:
"hosts" is an invalid keyword
```

□参照此方法配置 /compat/linux/etc/host.conf：

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

□里指定了先□□ /etc/hosts 再□□ DNS。 如果 /compat/linux/etc/host.conf 不存在的□， Linux® 程序便会

□取 /etc/host.conf 并提示与 FreeBSD 的□法不兼容。 如果没有在 /etc/resolv.conf 文件中配置域名服□器，可以□除 bind。

11.3. 高□主□

此章□将□述是 Linux® 二□制兼容如何工作的， 内容基于 Terry Lambert tlambert@primenet.com (Message ID: <199906020108.SAA07001@usr09.primenet.com>) □表在 FreeBSD □聊□件列表 的□件。

FreeBSD 有一个叫 "execution class loader" 的抽象□。 它被嵌入□了 execve(2) 系□□用。

□史上 UNIX® 加□器会依□□看魔数 （通常是文件的□□ 4 至 8 个字□）来□□是否是系□已知的的二□制程序，如果是的□， 就会□用二□制程序加□器。

如果它不是二□制□型的程序， execve(2) □用会返回一个□□， shell □会把它当作 shell 命令□行。 "不□当前是□一□ shell" 都会默□做出此□假□。

随后， sh(1) 会□□□□的□个字符， 如果它□是 :\n， 那□就□用 csh(1)。

FreeBSD 有一□加□器列表而不是一个□一的加□器， 并能回退到 #! 加□器来□行 shell 解□器或者 shell 脚本。

□了支持 Linux® ABI， FreeBSD 看到了二□制 ELF 程序的魔数。 ELF 加□器会□□一个□用的 □□， 那是在 ELF □像中的一个注□部分， 此区域在 SVR4/Solaris™ ELF 二□制中并不存在。

要□行 Linux® 二□制程序， 必□先使用 brandelf(1) 命令 □□ □ Linux □型：

```
# brandelf -t Linux file
```

当 ELF 加□器看到了 Linux □□，便会替□ proc □□中的一个指□。 所有的系□□用都通□此指□来索引。除此以外， □程被□□以便□ signal trampoline 代□的陷□向量做特殊□理， □有一些其他由 Linux® 内核模□来□理的（□微）修□。

Linux® 系□□用向量包含一个 sysent[] □□的列表， 它的地址位于内核模□之中。

当一个系□□用被 Linux® 二□制程序□用□， 陷□代□会把系□□用函数指□从 proc 解引用至 Linux® 而不是 FreeBSD 的系□□用入口。

Linux® 模式会□□地 *reroots* □□。 □与 union 文件系□□□是等效的。 首先会□□在 /compat/linux/original-path 目□□□文件。 如果失□了， 就会在 /original-path 目□下□□。 □使得需要其它程序的程序得以□行。例如，Linux® 工具□都可以在 Linux® ABI 的支持下□行。 也就是□ Linux® 二□制程序可以加□并□行 FreeBSD 二□制程序， 如果当前没有相□的 Linux® 二□制程序， 可以在 /compat/linux 目□□中放置一个 uname(1) 命令， 使 Linux® 程序不易察□它□并没有□行在 Linux® 系□上。

事□上， 在 FreeBSD 内核中有一个 Linux® 内核。 所有由内核提供的服□的各□底□功能在 FreeBSD 系□□用表的□□和 Linux® 系□□用表的□□是一□的： 文件系□操作， 虚□内存操作， 信号□送， 和 System V IPC。 唯一的不同是 FreeBSD 会得到 FreeBSD 的 *glue* 功能， 而 Linux® 程序会得到 Linux® 的 *glue* 功能。 FreeBSD 的 *glue* 功能是静□□接入内核的， 而 Linux® 的 *glue* 功能可以静□□接， 或者通□内核模□□□。

□格□来其□并没有真正的模□， □是一□ ABI 的□□。 有□□被称□ "Linux® 模□" 是因□在□□的□候□没有其他□合的

□用来描述。 要□ FreeBSD □行 Linux® 二□制程序并不□切， 因□当□代□并□没有被□□□去。

部分 III: 系□管理

FreeBSD　手册中其余章□的内容都是□于系□管理。　□一章□都从描述□章将要介□的内容□始，　由浅入深□相□内容□行介□。

□些章□在撰写□，　已□□□成了□多相互独立的部分，　如果□需要了解某部分内容，　直接□□□部分内容即可，而无需按照□序，　也不必在□□始使用 FreeBSD 之前完整地□□它□。

Chapter 12. □置和□整

12.1. 概述

使用 FreeBSD 的一个重要□□是系□配置。 正□地配置系□能充分地□少以后□□和升□系□所需的工作量。 □章将解□一些 FreeBSD 的配置□程，包括一些可以□整的 FreeBSD 系□的一些参数。

□完本章， □将了解：

- 如何有效地利用文件系□和交□分区。
- rc.conf 的基本□置以及 /usr/local/etc/rc.d □□体系。
- 如何□置和□□网□。
- 如何在□的网□□□上配置虚□主机。
- 如何使用 /etc 下的各配置文件。
- 如何通□ `sysctl` □量来□ FreeBSD 系□□行□□。
- □□□整磁□性能和修改内核限制。

在□□本章之前，□□□了解：

- 了解 UNIX® 和 FreeBSD 的基□知□ (UNIX 基□)。
- 熟悉内核配置□□的基□知□ (配置FreeBSD的内核)。

12.2. 初□配置

12.2.1. 分区□□

12.2.1.1. 基本分区

当使用 bsdlabel(8) 或者 sysinstall(8) 来分割□的文件系□的□候， 要□住硬□□□器外磁道□□数据要比从内磁道□□数据快。 因此□□将小的和□常□□的文件系□放在□□器□外的位置， 一些大的分区比如 /usr □□放在磁□比□□里的位置。 以□似□□的□序建立分区是一个不□的主意：root，swap， /var， /usr。

/var 分区的大小能反映□的机器使用情况。 /var 文件系□用来存□□件， 日志文件和打印□列□存， 特□是□箱和日志文件可能会□到无法□料的大小， □主要取决于在□的系□上有多少用□和□的日志文件可以保存多□□□。 大多数用□很少需要 /var 有 1GB 以上的□置空□。

有□候 /var/tmp 需要很多的磁□空□。 在使用 pkg_add(1) 安装新的□件□，包管理工具会在 /var/tmp 中解□出一□□□拷□。 大的□件包，像 Firefox， OpenOffice 或者 LibreOffice 在安装□如果 /var/tmp 中没有足□的空□就可能需要一些技巧了。

/usr 分区存□很多用来系□□行所需要的文件例如 ports(7) (建□□□做) 和源代□ (可□的)。 ports 和基本系□的源代□在安装□都是可□的， 但我□建□□□个分区至少保留 2GB 的可用空□。

当□□分区大小的□候，□住保留一些空□。 用完了一个分区的空□而在□一个分区上□有很多， 可能会□致出□一些□□。

一些用□会□□ sysinstall(8) 的 Auto-defaults 自□分区有□会分配□ /var 和 / □小的分区空□。 分区□□精□一些并且大一些。

12.2.1.2. 交□分区

一般来□，交□分区□□大□是系□内存 (RAM) 的□倍。 例如，如果机器有 128M 内存，交□文件□□是 256M。 □小内存的系□可以通□多一点地交□分区来提升性能。 不建□小于 256 兆的交□分区，并且□充□的内存□□被考□一下。 当交□分区最少是主内存的□倍的□候，内核的 VM (虚□内存) □面□度算法可以将性能□整到最好。如果□□机器添加更多内存， 配置太小的交□分区会□致 VM □面□描的代□效率低下。

在使用多□SCSI磁□(或者不同控制器上的IDE磁□)的大系□上， 建□在□个□□器上建立交□分区(直到四个□□器)。 交□分区□□大□一□大小。内核可以使用任意大小， 但内部数据□□□是最大交□分区的 4 倍。保持交□分区同□的大小， 可以允□内核最佳地□度交□空□来□□磁□。 即使不太使用，分配大的交□分区也是好的， 在被迫重□之前它可以□□更容易的从一个失□的程序中恢□□来。

12.2.1.3. □什□要分区？

一些用□□□一个□独的大分区将会很好， 但是有很多原因会□明□什□□是个坏主意。首先，□个分区有不同的分区特性，因此分□可以□文件系□□整它□。 例如，根系□和 /usr 一般只是□取，写入很少。很多□写□繁的被放在 /var 和 /var/tmp中。

□当的□分一个系□， 在其中使用□小的分区， □□， 那些以写□主的分区将不会比以□□主的分区付出更高的代价。 将以写□主的分区放在□近磁□的□□， 例如放在□□的大硬□的前面代替放在分区表的后面，将会提高□需要的分区的 I/O 性能。□在可能也需要在比□大的分区上有很好的 I/O 性能， 把他□移□到磁□外□不会□来多大的性能提升，反而把 /var 移到外面会有很好的效果。最后□及到安全□□。 一个主要是只□的小的、整□的根分区可以提高从一个□重的系□崩□中恢□□来的机会。

12.3. 核心配置

系□的配置信息主要位于 /etc/rc.conf。 □个文件包含了配置信息很大的一部分，主要在系□□□的□候来配置系□， □个名字直接□明了□点；它也是 rc* 文件的配置信息。

系□管理□□□在 rc.conf 文件中建立□□来覆□ /etc/defaults/rc.conf 中的默□□置。 □个默□文件不□□被逐字的□制到 /etc —— 它包含的是默□□而不是一个例子。 所有特定的改□□□在 rc.conf 中。

在集群□用中，□了降低管理成本， 可以采用多□策略把□及全站□□的□置从特定于系□的□置中分□出来。 推□的方法是把系□□□的配置放到 /etc/rc.conf.local 文件中。 例如：

- /etc/rc.conf:

```
sshd_enable="YES"
keyrate="fast"
defaultrouter="10.1.1.254"
```

- /etc/rc.conf.local:

```
hostname="node1.example.org"
ifconfig_fxp0="inet 10.1.1.1/8"
```

rc.conf 文件可以通□ rsync 或□似的程序来分□到所有的机器上， 而各自的 rc.conf.local 文件□保持不□。

使用 sysinstall(8) 或者 make world 来升□系□不会覆□ rc.conf 文件， 所以系□配置信息不会□失。

配置文件 /etc/rc.conf 是通□ sh(1) 解析的。 □使得系□管理□可以在其中添加一些□□，从而□建能□□□非常□□的□景的配置。 □参□□机手册 rc.conf(5) 来了解□于□一□□的□一□信息。

12.4. □用程序配置

典型的，被安装的□用程序有他自己的配置文件、□法等等。 从基本系□中分□他□是很重要的以至于他□可以容易的被 package 管理工具定位和管理

一般来□，□些文件被安装在 /usr/local/etc。□个例子中， 一个□用程序有很多配置文件并且□建了一个子目□来存放他□。

通常，当一个 port 或者 package 被安装的□候， 配置文件示例也同□被安装了。它□通常用 .default 的后□来□□。如果不存在□个□用程序的配置文件， 它□会通□□制 .default 文件来□建。

例如，看一下□个目下的内容 /usr/local/etc/apache：

```
-rw-r--r--  1 root  wheel   2184 May 20  1998 access.conf
-rw-r--r--  1 root  wheel   2184 May 20  1998 access.conf.default
-rw-r--r--  1 root  wheel   9555 May 20  1998 httpd.conf
-rw-r--r--  1 root  wheel   9555 May 20  1998 httpd.conf.default
-rw-r--r--  1 root  wheel  12205 May 20  1998 magic
-rw-r--r--  1 root  wheel  12205 May 20  1998 magic.default
-rw-r--r--  1 root  wheel   2700 May 20  1998 mime.types
-rw-r--r--  1 root  wheel   2700 May 20  1998 mime.types.default
-rw-r--r--  1 root  wheel   7980 May 20  1998 srm.conf
-rw-r--r--  1 root  wheel   7933 May 20  1998 srm.conf.default
```

文件大小□示了只有 srm.conf 改□了。以后 Apache 的升□就不会改□□个文件。

12.5. □□服□

□多用□会□□使用 Ports Collection 来在 FreeBSD 上安装第三方□件。 很多情况下□可能需要□行一些配置以便□□些□件能□在系□初始化的□程中□□。 服□， 例如 mail/postfix 或 www/apache13 就是□些需要在系□初始化□□□的□件包中的□个典型代表。 □一□解□了□□第三方□件所需要的□□。

FreeBSD 包含的大多数服□，例如 cron(8)， 就是通□系□□□脚本□□的。 □些脚本也□会有些不同， □取决于 FreeBSD 版本。 但是不管□□， 需要考□的一个重要方面是他□的□□配置文件要能被基本□□脚本□□捕□。

12.5.1. □展□用程序配置

□在 FreeBSD 提供了 rc.d， □使得□□用□件的□□□行配置□得更加方便， 并提供了更多的其他功能。 例如，使用在 rc.d 一□中所介□的□□字， □用程序就可以□置在某些其他服□， 例如 DNS 之后□□； 除此之外，□可以通□ rc.conf 来指定一些□外的□□参数， 而不再需要将它□硬□□到□□脚本中。 基本的□□脚本如下所示：

```
#!/bin/sh
#
# PROVIDE: utility
# REQUIRE: DAEMON
# KEYWORD: shutdown

. /etc/rc.subr

name=utility
rcvar=utility_enable

command="/usr/local/sbin/utility"

load_rc_config $name

#
# DO NOT CHANGE THESE DEFAULT VALUES HERE
# SET THEM IN THE /etc/rc.conf FILE
#
utility_enable=${utility_enable-"NO"}
pidfile=${utility_pidfile-"/var/run/utility.pid"}

run_rc_command "$1"
```

□个脚本将保□ utility 能□在 DAEMON 服□之后□□。 它同□也提供了□置和跟踪 PID， 也就是□程 ID 文件的方法。

可以在 /etc/rc.conf 中加入：

```
utility_enable="YES"
```

□个方法也使得命令行参数、包含 /etc/rc.subr 中所提供的功能， 兼容 rcorder(8) 工具并提供更□□的通□ rc.conf 文件来配置的方法。

12.5.2. 用服□来□□服□

其他服□， 例如 POP3 服□器， IMAP， 等等， 也可以通□ inetd(8) 来□□。 □一□程包括从 Ports Collection 安装相□的□用程序， 并把配置加入到 /etc/inetd.conf 文件， 或去掉当前配置中的某些注□。 如何使用和配置 inetd 在 inetd 一□中□行了更□深入的□述。

一些情况下， 通□ cron(8) 来□□系□服□也是一□可行的□□。 □□方法有很多好□， 因□ cron 会以 crontab 的文件属主身□□行那些□程。 □使得普通用□也能□□行他□的□用。

cron 工具提供了一个独有的功能， 以 @reboot 来指定□□。 □□的□置将在 cron(8) □□□□行， 通常□也是系□初始化的□候。

12.6. 配置 cron

FreeBSD 最有用的□件包(utilities)中的一个是 cron(8)。 cron □件在后台□行并且□常□□ /etc/crontab 文件。cron □件也□□ /var/cron/tabs 目□，搜索新的 crontab 文件。□些 crontab 文件存□一些 cron 在特定□□□行任□的信息。

cron 程序使用□□不同□型的配置文件， 即系□ crontab 和用□ crontabs。 □□格式的唯一区□是第六个字段。在系□ crontab 中，第六个字段是用于□行命令的用□名。 □□予了系□ crontab 以任意用□身□□行命令的能力。在用□ crontab 中， 第六个字段是要□行的命令， 所有的命令都会以□个用□自己的身□□行； □是一□重要的安全功能。

同其他用□一□， root 用□也可以有自己的 crontab。 它不同于 /etc/crontab (也就是系□ crontab)。 由于有系□ crontab 的存在， 通常并不需要□ root 建立□独的用□ crontab。

□我□来看一下 /etc/crontab 文件：

```
# /etc/crontab - root's crontab for FreeBSD
#
# $FreeBSD: src/etc/crontab,v 1.32 2002/11/22 16:13:39 tom Exp $
#①
#
SHELL=/bin/sh
PATH=/etc:/bin:/sbin:/usr/bin:/usr/sbin ②
HOME=/var/log
#
#
#minute hour    mday    month   wday    who command ③
#
#
*/5 *   *   *   *   root    /usr/libexec/atrun ④
```

① 像大多数 FreeBSD 配置文件一□，# 字符是注□。 □□， 就可以□写注□来□明要□行什□操作， 以及□□做的原因。 需要注意的是， 注□□□□起一行， 而不能跟命令放在同一行上， 否□它□会被看成命令的一部分。 □个文件中的空行会被忽略。

② 首先□□定□□境□量。等号 (=) 字符用来定□任何□境□量，像□个例子用到了 SHELL，PATH 和 HOME □量。如果 shell 行被忽略掉，cron 将会用默□□ sh。如果 PATH □量被忽略， 那□就没有默□□并且需要指定文件□□位置。如果 HOME 被忽略，cron 将用用□行者的 home 目□。

③ □一行定□了七个字段。它□是 minute、 hour、mday、 month、wday、 who 和 command。 它□差不多已□□明了各自的用□。Minute 是命令要□行□的分□，Hour 跟 minute 差不多，只是用小□来表示。Mday 是□个月的天。Month 跟 hour □有 minute 都差不多，用月□来表示。wday 字段表示星期几。 所有□些字段的□必□是数字并且用24小□制来表示。"who" 字段是特□的，并且只在 /etc/crontab 文件中存在。□个字段指定了命令□□以□个用□的身□来□行。当一个用□添加了他(□)的 crontab 文件的□候，他□就会没有□个字段□□。最后，是 command 字段。□是最后的一个字段， 所以自然就是它指定要□行的程序。

④ 最后一行定□了上面所□的□。注意□里我□有一个 /5 列表，□跟着是一些 字符。* 字符代表"□始到最后"，

也可以被解□成 □次。所以，根据□行， □然表明了无□在何□□隔 5 分□以 root 身□来□行 atrun 命令。□看 atrun(8) 手册□以□得 atrun 的更多信息。命令可以有任意多个□□□它□的□志。无□□□， □展到多行的命令□□用反斜□("\")来□行。

□是□个 crontab 文件的基本□置， □然它□有一个不同。第六行我□指定的用□名只存在于系□ /etc/crontab 文件。□个字段在普通用□的 crontab 文件中□□被忽略。

12.6.1. 安装 Crontab

□□不要用□□方法来□□/安装系□ crontab。 □需要做的只是使用自己喜□的□□器： cron 程序会注意到文件□生了□化， 并立即□始使用新的版本。参□ □个 FAQ □目 以了解□一□的情况。

要安装□写好的用□ crontab， 首先使用最□□的□□器来□建一个符合要求格式的文件，然后用 crontab 程序来完成。最常□的用法是：

```
% crontab crontab-file
```

在前面的例子中， crontab-file 是一个事先写好的 crontab。

□有一个□□用来列出安装的 crontab 文件： 只要□□ -l □□□ crontab 然后看一下□出。

用□想不用模板(已□存在的文件)而直接安装他的 crontab 文件，用 crontab -e □□也是可以的。 它将会□□一个□□器并且□建一个新文件，当□个文件被保存的□候， 它会自□的用 crontab 来安装□个文件。

如果□□后想要□底□除自己的用□ crontab 可以使用 crontab 的 -r □□。

12.7. 在 FreeBSD 中使用 rc

在 2002 年， FreeBSD 整合了来自 NetBSD 的 rc.d 系□， 并通□它来完成系□的初始化工作。 用□要注意在 /etc/rc.d 目□下的文件。 □里面的□多文件是用来管理基□服□的， 它□可以通□ start、 stop， 以及 restart □□来控制。 □例来□， sshd(8) 可以通□下面的命令来重□：

```
# /etc/rc.d/sshd restart
```

□其它服□的操作与此□似。 当然， □些服□通常是在□□□根据 rc.conf(5) 自□□□的。 例如， 要配置使系□□□□□□网□地址□□服□， 可以□□地通□在 /etc/rc.conf 中加入如下□置来完成：

```
natd_enable="YES"
```

如果 natd_enable="NO" 行已□存在， 只要□□的把 NO 改成 YES 即可。 rc 脚本在下次重新□□的□候会自□的装□所需要的服□， 像下面所描述的那□。

由于 rc.d 系□在系□□□/□□□首先□□/停止服□，如果□置了□当的 /etc/rc.conf □量，□准的 start、stop 和 restart □□将会□行他□的□作。例如 sshd restart 命令只在 /etc/rc.conf 中的 sshd_enable □置成 YES 的□候工作。不管是否在 /etc/rc.conf 中□置了，要 start、stop 或者 restart 一个服□

，命令前可以加上一个"one"前。例如要不当前 /etc/rc.conf 的置重新 sshd，行下面的命令：

```
# /etc/rc.d/sshd onerestart
```

用 rcvar 可以来的 /etc/rc.conf 中用当的 rc.d 脚本的服是否被用。从而管理可以行的程序来 sshd 是否真的在 /etc/rc.conf 中被了：

```
# /etc/rc.d/sshd rcvar
# sshd
$sshd_enable=YES
```

第二行 (# sshd) 是从 sshd 命令中出的，而不是 root 控制台。

了定一个服是否真的在行，可以用 status 。例如 sshd 是否真的了：

```
# /etc/rc.d/sshd status
sshd is running as pid 433.
```

有些候也可以 reload 服。 一操作上是向服送一个信号， 来制其重新加配置。 多数情况下， 服的会是 SIGHUP 信号。 并非所有服都支持一功能。

rc.d 系不用于网服， 它也系初始化中的多数程提供支持。 比如 bgfsck 文件， 当它被行， 将会出下述信息：

```
Starting background file system checks in 60 seconds.
```

个文件用做后台文件系，系初始化的候完成。

很多系服依其他服提供的相功能。例如，NIS 和其他基于 RPC 的服可能在 rpcbind 服之前失。要解决个，依系信息和其他信息当作注被包含在个脚本文件的前面。 程序在系初始化分析些注以决定用其他系服来足依系。

下面的字句必被包含在所有的脚本文件里， （他都是 rc.subr(8) 用来 "enable" 脚本必需的）：

- PROVIDE: 指定此文件所提供的服的名字。

以下的字句可以被包含在文件的部。格来他不是必需的， 但作于 rcorder(8) 有一定的提示作用：

- REQUIRE: 列出此服之前所需要的其他服。 此脚本提供的服会在指定的那些服 之后 。
- BEFORE: 列出依此服的其他服。 此脚本提供的服将在指定的那些服 之前 。

通在脚本中仔定些字， 系管理可以很有条理的控制脚本的序， 而避免使用像其他 UNIX® 操作系那混乱的 "runlevels"。

更多于 rc.d 系的信息， 可以在 rc(8) 和 rc.subr(8) 机手册中到。 如果有意撰写自己的 rc.d 脚本，或有的脚本行一些改， 也可以参考 篇文章。

12.8. □置网□

□在我□不可想象一台□算机没有网□□接的情况。 添加和配置一□网□是任何 FreeBSD 系□管理□的一□基本任□。

12.8.1. □□正□的□□程序

在□始之前，□□□知道□的网□□型，它用的芯片和它是 PCI □是 ISA 网□。FreeBSD 支持很多□ PCI 和 ISA 网□。可以□看□的版本硬件兼容性列表以□定□的网□被支持。

□□系□能□支持□的网□之后， □□需要□它□□合□的□□程序。 /usr/src/sys/conf/NOTES 和 /usr/src/sys/arch/conf/NOTES 将□□提供所支持的一些网□和芯片□的信息。 如果□□疑□□程序是否使所要□的那一个， □参考□□程序的□机手册。 □机手册将提供□于所支持的硬件更□□的信息， 甚至□包括可能□生的□□。

如果□的网□很常□的□， 大多数□候□不需要□□□浪□精力。 常用的网□在 GENERIC 内核中已□支持了， 所以□的网□在□□□就会□示出来，像是：

```
dc0: <82c169 PNIC 10/100BaseTX> port 0xa000-0xa0ff mem 0xd3800000-0xd38
000ff irq 15 at device 11.0 on pci0
miibus0: <MII bus> on dc0
bmtphy0: <BCM5201 10/100baseTX PHY> PHY 1 on miibus0
bmtphy0:  10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
dc0: Ethernet address: 00:a0:cc:da:da:da
dc0: [ITHREAD]
dc1: <82c169 PNIC 10/100BaseTX> port 0x9800-0x98ff mem 0xd3000000-0xd30
000ff irq 11 at device 12.0 on pci0
miibus1: <MII bus> on dc1
bmtphy1: <BCM5201 10/100baseTX PHY> PHY 1 on miibus1
bmtphy1:  10baseT, 10baseT-FDX, 100baseTX, 100baseTX-FDX, auto
dc1: Ethernet address: 00:a0:cc:da:da:db
dc1: [ITHREAD]
```

在□个例子中，我□看到有□□使用 dc(4) □□的网□在系□中。

如果□的网□没有出□在 GENERIC 中， □需要手工加□合□的□□程序。 要完成□□工作可以使用下面□□方法之一：

- 最□□的□法是用 kldload(8) 加□网□□□的内核模□。 除此之外， 通□在 /boot/loader.conf 文件中加入□当的□置， 也可以□系□在引□□自□加□□些模□。 不□， 并不是所有的网□都能□通□□□方法提供支持； ISA 网□是比□典型的例子。
- □外， □也可以将网□的支持静□□□□内核。 察看 /usr/src/sys/conf/NOTES，/usr/src/sys/arch/conf/NOTES 以及□□程序的□机手册以了解需要在□的内核配置文件中加一些什□。要了解□于重新□□内核的□一□□□， □参□ 配置FreeBSD的内核。 如果□的□在引□□可以被内核 (GENERIC) □□， □□□不需要□□新的内核。

12.8.1.1. 使用 Windows® NDIS □□程序

不幸的是， □多厂商由于□□□□程序会□及□多敏感的商□机密， 至今仍不□意将把□□程序作□□放源代□形式□布列入他□的□□表。 因此， FreeBSD 和其他操作系□的□□者就只剩下了□□□□： 要□□□□□□的痛苦□程来

□□□□行逆向工程， 要□使用□存的□ Microsoft® Windows® 平台提供的□□□版本的□□程序。 包括参与 FreeBSD □□的□大多数□□人□， 都□□了后一□方法。

得益于 Bill Paul (wpaul) 的工作， 已□可以 "直接地" 支持 网□□□接口□准 (NDIS, Network Driver Interface Specification) 了。 FreeBSD NDISulator (也被称□ Project Evil) 可以支持二□制形式的 Windows® □□程序， 并□它相信正在□行的是 Windows®。 由于 ndis(4) □□使用的是用于 Windows® 的二□制形式的□□， 因此它只能在 i386™ 和 amd64 系□上使用。

ndis(4) □□在□□□主要提供了 PCI、 CardBus 和 PCMCIA □□的支持， 而 USB □□目前□没有提供支持。

要使用 NDISulator， □需要三件□西：

1. 内核的源代□
2. 二□制形式的 Windows® XP □□程序 (□展名□ .SYS)
3. Windows® XP □□程序配置文件 (□展名□ .INF)

□需要□到用于□的□的□些文件。 一般而言， □些文件可以在随□附送的 CD 或制造商的网站上□到。 在下面的例子中， 我□用 W32DRIVER.SYS 和 W32DRIVER.INF 来表示□些文件。

不能在 FreeBSD/amd64 上使用 Windows®/i386 □□程序。 必□使用 Windows®/amd64 □□才能在其上正常工作。

接下来的□□是将二□制形式的□□程序□装成内核模□。 要完成□一任□， 需要以 `root` 用□的身□□行 ndisgen(8)：

```
# ndisgen /path/to/W32DRIVER.INF /path/to/W32DRIVER.SYS
```

ndisgen(8) 是一个交互式的程序， 它会提示□□入所需的一些其他的□外信息； □些工作完成之后， 它会在当前目□生成一个内核模□文件， □个文件可以通□下述命令来加□：

```
# kldload ./W32DRIVER_SYS.ko
```

除了□□生成的内核模□之外， □必□加□ ndis.ko 和 if_ndis.ko □□个内核模□， 在□加□需要 ndis(4) 的模□□， 通常系□会自□完成□一操作。 如果希望手工加□它□， □可以使用下列命令：

```
# kldload ndis
# kldload if_ndis
```

第一个命令会加□ NDIS 袖珍端口□□封装模□， 而第二条命令□加□□□的网□接口。

□在□□看 dmesg(8) 来了解是否□生了□□。 如果一切正常， □会看到□似下面的□出：

```
ndis0: <Wireless-G PCI Adapter> mem 0xf4100000-0xf4101fff irq 3 at device 8.0 on pci1
ndis0: NDIS API version: 5.0
ndis0: Ethernet address: 0a:b1:2c:d3:4e:f5
ndis0: 11b rates: 1Mbps 2Mbps 5.5Mbps 11Mbps
ndis0: 11g rates: 6Mbps 9Mbps 12Mbps 18Mbps 36Mbps 48Mbps 54Mbps
```

□之后， 就可以像使用其它网□接口 (例如 dc0) 一□来使用 ndis0 □□了。

与任何其它模□一□， □也可以配置系□， 令其在□□□自□加□ NDIS 模□。 首先， 将生成的模□ W32DRIVER_SYS.ko □制到 /boot/modules 目□中。 接下来， 在 /boot/loader.conf 中加入：

```
W32DRIVER_SYS_load="YES"
```

12.8.2. 配置网□

□在正□的网□□□程序已□装□，那□就□□配置它了。 跟其他配置一□，网□可以在安装□用 sysinstall 来配置。

要□示□系□上的网□接口的配置，□入下列命令：

```
% ifconfig
dc0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=80008<VLAN_MTU,LINKSTATE>
        ether 00:a0:cc:da:da:da
        inet 192.168.1.3 netmask 0xffffff00 broadcast 192.168.1.255
        media: Ethernet autoselect (100baseTX <full-duplex>)
        status: active
dc1: flags=8802<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=80008<VLAN_MTU,LINKSTATE>
        ether 00:a0:cc:da:da:db
        inet 10.0.0.1 netmask 0xffffff00 broadcast 10.0.0.255
        media: Ethernet 10baseT/UTP
        status: no carrier
plip0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> metric 0 mtu 1500
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> metric 0 mtu 16384
        options=3<RXCSUM,TXCSUM>
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x4
        inet6 ::1 prefixlen 128
        inet 127.0.0.1 netmask 0xff000000
        nd6 options=3<PERFORMNUD,ACCEPT_RTADV>
```

在□个例子中，□示出了下列□□：

- dc0: 第一个以太网接口
- dc1: 第二个以太网接口
- plip0： 并口 (如果系□中有并口的□)
- lo0: 回□□□

FreeBSD 使用内核引□□□□到的网□□□□序来命名网□。例如 sis2 是系□中使用 sis(4) □□的第三□网□。

在□个例子中，dc0 □□□用了。主要表□在：

1. UP 表示□□网□已□配置完成准□工作。
2. □□网□有一个 Internet (inet) 地址 (□个例子中是 192.168.1.3)。
3. 它有一个有效的子网掩□ (netmask； 0xffffff00 等同于 255.255.255.0)。
4. 它有一个有效的广播地址 (□个例子中是 192.168.1.255)。
5. 网□的 MAC (ether) 地址是 00:a0:cc:da:da:da
6. 物理□□媒介模式□于自□□□状□ (media: Ethernet autoselect (100baseTX <full-duplex>))。我□看到 dc1 被配置成□行在 10baseT/UTP 模式下。 要了解□□媒介□型的更多信息， □□□它□的使用手册。
7. □接状□ (status)是 active，也就是□□接信号被□□到了。□于 dc1，我□看到 status: no carrier。□通常是网□没有□好。

如果 ifconfig(8) 的□出□示了□似于：

```
dc0: flags=8843<BROADCAST,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=80008<VLAN_MTU,LINKSTATE>
        ether 00:a0:cc:da:da:da
        media: Ethernet autoselect (100baseTX <full-duplex>)
        status: active
```

的信息，那□就是□没有配置网□。

要配置网□，□需要 root □限。 网□配置可以通□使用 ifconfig(8) 命令行方式来完成， 但是□□□次□□都要做一遍。放置网□配置信息的文件是 /etc/rc.conf。

用□自己喜□的□□器打□ /etc/rc.conf。 并且□需要□□一□系□中存在的网□添加一行， 在我□的例子中，添加如下几行：

```
ifconfig_dc0="inet 192.168.1.3 netmask 255.255.255.0"
ifconfig_dc1="inet 10.0.0.1 netmask 255.255.255.0 media 10baseT/UTP"
```

用自己正□的□□名和地址来替□例子中的 dc0，dc1 等内容。□□□□□□□网□□□和 ifconfig(8) 的手册□来了解各□□，也要□看一下 rc.conf(5) □助□来了解 /etc/rc.conf 的□法。

如果在安装的□候配置了网□，□于网□的一些行可能已□存在了。 所以在添加新行前仔□□□一下 /etc/rc.conf。

□也可能需要□□ /etc/hosts 来添加局域网中不同的机器名称和 IP 地址， 如果它□不在那里的□。 □□看□机手册 hosts(5) 和 /usr/shared/examples/etc/hosts 以了解更多信息。

如果□□通□□台机器□□ Internet， □□需要手工配置默□网□和域名解析服□器：

```
# echo 'defaultrouter="your_default_router"' >> /etc/rc.conf
# echo 'nameserver your_DNS_server' >> /etc/resolv.conf
```

12.8.3. □□和□□

□ /etc/rc.conf 做了必要的修改之后□□重□系□以□用□接口的修改， 并且□□系□重□后没有任何配置□□。 □外□也可以重□网□系□：

```
# /etc/rc.d/netif restart
```

如果在 /etc/rc.conf 中配置了默□网□， □需要□行下面的命令：

```
# /etc/rc.d/routing restart
```

网□系□重□之后， □□□网□接口。

12.8.3.1. □□以太网□

□了□□网□被正□的配置了，在□里我□要做□件事情。首先， ping 自己的网□接口，接着 ping 局域网内的其他机器。

首先□□本地接口：

```
% ping -c5 192.168.1.3
PING 192.168.1.3 (192.168.1.3): 56 data bytes
64 bytes from 192.168.1.3: icmp_seq=0 ttl=64 time=0.082 ms
64 bytes from 192.168.1.3: icmp_seq=1 ttl=64 time=0.074 ms
64 bytes from 192.168.1.3: icmp_seq=2 ttl=64 time=0.076 ms
64 bytes from 192.168.1.3: icmp_seq=3 ttl=64 time=0.108 ms
64 bytes from 192.168.1.3: icmp_seq=4 ttl=64 time=0.076 ms

--- 192.168.1.3 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.074/0.083/0.108/0.013 ms
```

□在我□□□ ping 局域网内的其他机器：

```
% ping -c5 192.168.1.2
PING 192.168.1.2 (192.168.1.2): 56 data bytes
64 bytes from 192.168.1.2: icmp_seq=0 ttl=64 time=0.726 ms
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=0.766 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=64 time=0.700 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=64 time=0.747 ms
64 bytes from 192.168.1.2: icmp_seq=4 ttl=64 time=0.704 ms

--- 192.168.1.2 ping statistics ---
5 packets transmitted, 5 packets received, 0 packet loss
round-trip min/avg/max/stddev = 0.700/0.729/0.766/0.025 ms
```

□如果□□置了 /etc/hosts 文件，也可以用机器名来替□ 192.168.1.2。

12.8.3.2. □□

□□硬件和□件配置一直是一件□痛的事情，　　从最□□的□始可以□□一些痛苦。　　例如网□是否□好了？是否配置好了网□服□？防火□配置正□□？　是否使用了被　FreeBSD　支持的网□？　在□送□□□告之前□□□□看一下硬件□明，　升□ FreeBSD 到最新的 STABLE 版本，　看一下□件列表或者在 Internet 上搜索一下。

如果网□工作了，　但性能低下，□□好好□□一下 tuning(7) □机手册。　□也可以□□一下网□配置，　不正□的□置会□致慢速的网□□接。

一些用□可能会在一些网□上□□一到□次　device　timeouts，　□通常是正常□象。　如果□常□□甚至引起麻□，□□□定一下它跟其他□□没有冲突。仔□□□网□□接，　或者□一□网□。

有□用□会看到少量 watchdog timeout □□。　□□情况要做的第一件事就是□□□□□接。　一些网□需要支持□□控制的 PCI □槽。　在一些老的主板上，只有一个 PCI □槽支持 (一般是 slot 0)。　□□网□和主板□明□来□定是不是□个□□。

No route to host 通常□生在如果系□不能□送一个路由到目的主机的包的□候。　□在没有指定默□路由或者网□没有□上□会□生。　□□ netstat -rn 的□出并□□有一个有效的路由能到□相□的主机。　如果没有，□□□ 高□网□□用。

ping: sendto: Permission denied　□□信息□常由防火□的配置□□引起。　如果　ipfw　在内核中□用了但是没有定□□□，　那□默□的□□就是拒□所有通□，甚至 ping □求！　□□ 防火□ 以了解更多信息。

有□网□性能低下或者低于平均水平，　□□情况最好把□□媒介模式从 autoselect 改□□正□的□□介□模式。　□通常□大多数硬件有用，　但可能不会解决所有人的□□。　接着，□□所有网□□置，并且□□ tuning(7) 手册□。

12.9. 虚□主机

FreeBSD　　的一个很普通的用途是虚□主机站点，　　一个服□器虚□成很多服□器一□提供网□服□。　　□通□在一个接口上□定多个网□地址来□□。

一个特定的网□接口有一个"真□"的地址，　　也可能有一些"□名"地址。□些□名通常用　　/etc/rc.conf　　中的□□来添加。

一个 fxp0 的□名□□□似于：

```
ifconfig_fxp0_alias0="inet xxx.xxx.xxx.xxx netmask xxx.xxx.xxx.xxx"
```

□住□名□□必□从 alias0 □始并且按□序□□(例如 _alias1、 _alias2)。 配置程序将会停止在第一个缺少的数字的地方。

f□算□名的子网掩□是很重要的，幸□的是它很□□。 □于一个接口来□，必□有一个描述子网掩□的地址。 任何在□个网段下的地址必□有一个全是 1 的子网掩□(通常表示□ 255.255.255.255 或 0xffffffff。

□例来□， 假□使用 fxp0 □接到□个网□， 分□是 10.1.1.0， 其子网掩□□ 255.255.255.0， 以及 202.0.75.16， 其子网掩□□ 255.255.255.240。 我□希望从 10.1.1.1 到 10.1.1.5 以及从 202.0.75.17 到 202.0.75.20 的地址能□互相□□。 如前所述， 只有□个网段中的第一个地址 (本例中， 10.0.1.1 和 202.0.75.17) □使用真□的子网掩□； 其余的 (10.1.1.2 到 10.1.1.5 以及 202.0.75.18 到 202.0.75.20) □必□配置□使用 255.255.255.255 作□子网掩□。

下面是根据上述描述所□行的 /etc/rc.conf 配置：

```
ifconfig_fxp0="inet 10.1.1.1 netmask 255.255.255.0"
ifconfig_fxp0_alias0="inet 10.1.1.2 netmask 255.255.255.255"
ifconfig_fxp0_alias1="inet 10.1.1.3 netmask 255.255.255.255"
ifconfig_fxp0_alias2="inet 10.1.1.4 netmask 255.255.255.255"
ifconfig_fxp0_alias3="inet 10.1.1.5 netmask 255.255.255.255"
ifconfig_fxp0_alias4="inet 202.0.75.17 netmask 255.255.255.240"
ifconfig_fxp0_alias5="inet 202.0.75.18 netmask 255.255.255.255"
ifconfig_fxp0_alias6="inet 202.0.75.19 netmask 255.255.255.255"
ifconfig_fxp0_alias7="inet 202.0.75.20 netmask 255.255.255.255"
```

12.10. 配置文件

12.10.1. /etc 布局

在配置信息中有很多的目□，□些包括：

/etc	一般的系□配置信息。□儿的数据是与特定系□相□的。
/etc/defaults	系□配置文件的默□版本。
/etc/mail	□外的 sendmail(8) 配置信息，其他 MTA 配置文件。
/etc/ppp	用于用□□和内核□ ppp 程序的配置。
/etc/namedb	named(8) 数据的默□位置。通常 named.conf 和区域文件存放在□里。
/usr/local/etc	被安装的□用程序配置文件。可以参考□个□用程序的子目□。
/usr/local/etc/rc.d	被安装程序的 □□/停止 脚本。
/var/db	特定系□自□□生的数据□文件，像 package 数据□，位置数据□等等。

12.10.2. 主机名

12.10.2.1. /etc/resolv.conf

/etc/resolv.conf 指示了 FreeBSD 如何□□域名系□(DNS)。

resolv.conf 中最常□的□□是：

`nameserver`	按□序要□□的名字服□器的 IP 地址，最多三个。
`search`	搜索机器名的列表。□通常由本地机器名的域决定。
`domain`	本地域名。

一个典型的 resolv.conf 文件：

```
search example.com
nameserver 147.11.1.11
nameserver 147.11.100.30
```

只能使用一个 `search` 和 `domain` □□。

如果□在使用 DHCP，dhclient(8) □常使用从 DHCP 服□器接受来的信息重写 resolv.conf。

12.10.2.2. /etc/hosts

/etc/hosts 是 Internet 早期使用的一个□□文本数据□。 它□合 DNS 和 NIS 提供名字到 IP 地址的映射。 通□局域网□接的机器可以用□个□□的命名方案来替代□置一个 named(8) 服□器。□外，/etc/hosts 也可以提供一个 Internet 名称的本地□□以□□需要从外部□□□来的□担。

```
# $FreeBSD$
#
#
# Host Database
#
# This file should contain the addresses and aliases for local hosts that
# share this file.  Replace 'my.domain' below with the domainname of your
# machine.
#
# In the presence of the domain name service or NIS, this file may
# not be consulted at all; see /etc/nsswitch.conf for the resolution order.
#
#
::1         localhost localhost.my.domain
127.0.0.1       localhost localhost.my.domain
#
# Imaginary network.
#10.0.0.2       myname.my.domain myname
#10.0.0.3       myfriend.my.domain myfriend
#
# According to RFC 1918, you can use the following IP networks for
# private nets which will never be connected to the Internet:
#
#   10.0.0.0    -   10.255.255.255
#   172.16.0.0  -   172.31.255.255
#   192.168.0.0 -   192.168.255.255
#
# In case you want to be able to connect to the Internet, you need
# real official assigned numbers.  Do not try to invent your own network
# numbers but instead get one from your network provider (if any) or
# from your regional registry (ARIN, APNIC, LACNIC, RIPE NCC, or AfriNIC.)
#
```

/etc/hosts 用□□的格式：

```
[Internet address] [official hostname] [alias1] [alias2] ...
```

例如：

```
10.0.0.1 myRealHostname.example.com myRealHostname foobar1 foobar2
```

参考 hosts(5) 以□得更多信息。

12.10.3. 日志文件配置

12.10.3.1. syslog.conf

syslog.conf 是 syslogd(8) 程序的配置文件。 它指出了的 syslog □□信息□型被存□在特定的日志文件中。

```
# $FreeBSD$
#
#       Spaces ARE valid field separators in this file. However,
#       other *nix-like systems still insist on using tabs as field
#       separators. If you are sharing this file between systems, you
#       may want to use only tabs as field separators here.
#       Consult the syslog.conf(5) manual page.
*.err;kern.debug;auth.notice;mail.crit          /dev/console
*.notice;kern.debug;lpr.info;mail.crit;news.err /var/log/messages
security.*                                      /var/log/security
mail.info                                       /var/log/maillog
lpr.info                                        /var/log/lpd-errs
cron.*                                          /var/log/cron
*.err                                           root
*.notice;news.err                               root
*.alert                                         root
*.emerg                                         *
# uncomment this to log all writes to /dev/console to /var/log/console.log
#console.info                                   /var/log/console.log
# uncomment this to enable logging of all log messages to /var/log/all.log
#*.*                                            /var/log/all.log
# uncomment this to enable logging to a remote log host named loghost
#*.*                                            @loghost
# uncomment these if you're running inn
# news.crit                                     /var/log/news/news.crit
# news.err                                      /var/log/news/news.err
# news.notice                                   /var/log/news/news.notice
!startslip
*.*                                             /var/log/slip.log
!ppp
*.*                                             /var/log/ppp.log
```

参考 syslog.conf(5) 手册□以□得更多信息

12.10.3.2. newsyslog.conf

newsyslog.conf 是一个通常用 cron(8) □□□行的 newsyslog(8) 程序的配置文件。 newsyslog(8) 指出了什□□候日志文件需要打包或者重新整理。 比如 logfile 被移□到 logfile.0，logfile.0 被移□到 logfile.1 等等。□外，日志文件可以用 gzip(1) 来□□，它□是□□的命名格式： logfile.0.gz，logfile.1.gz 等等。

newsyslog.conf 指出了□个日志文件要被管理，要保留多少和它□什□□候被□建。 日志文件可以在它□□到一定大小或者在特定的日期被重新整理。

```
# configuration file for newsyslog
# $FreeBSD$
#
# filename          [owner:group]    mode count size when [ZB] [/pid_file] [sig_num]
/var/log/cron                           600  3     100  *     Z
/var/log/amd.log                        644  7     100  *     Z
/var/log/kerberos.log                   644  7     100  *     Z
/var/log/lpd-errs                       644  7     100  *     Z
/var/log/maillog                        644  7     *    @T00  Z
/var/log/sendmail.st                    644  10    *    168   B
/var/log/messages                       644  5     100  *     Z
/var/log/all.log                        600  7     *    @T00  Z
/var/log/slip.log                       600  3     100  *     Z
/var/log/ppp.log                        600  3     100  *     Z
/var/log/security                       600  10    100  *     Z
/var/log/wtmp                           644  3     *    @01T05 B
/var/log/daily.log                      640  7     *    @T00  Z
/var/log/weekly.log                     640  5     1    $W6D0 Z
/var/log/monthly.log                    640  12    *    $M1D0 Z
/var/log/console.log                    640  5     100  *     Z
```

参考 newsyslog(8) 手册□以□得更多信息。

12.10.4. sysctl.conf

sysctl.conf 和 rc.conf □□个文件的□格很接近。 其中的配置均□ □量=□ □□的形式。 在□个文件中配置的□，均会在系□□入多用□模式之后□行□□的修改操作。 需要注意的是， 并不是所有的□量都能□在多用□模式下修改。

如果希望□□□收到致命的信号退出的□程□行□□， 并阻止普通用□看到其他用□的□程， 可以在 sysctl.conf 中□行下列配置：

```
# 不□□由于致命信号□致的□程退出（例如信号 11，□□越界）
kern.logsigexit=0

# 阻止用□看到以其他用□ UID 身□□行的□程。
security.bsd.see_other_uids=0
```

12.11. 用 sysctl □行□整

sysctl(8) 是一个允□□改□正在□行中的 FreeBSD 系□的接口。它包含一些 TCP/IP 堆□和虚□内存系□的高□□，□可以□有□□的管理□提高引人注目的系□性能。用 sysctl(8) 可以□取□置超□五百个系□□量。

基于□点， sysctl(8) 提供□个功能：□取和修改系□□置。

□看所有可□□量：

```
% sysctl -a
```

□一个指定的□量，例如 kern.maxproc：

```
% sysctl kern.maxproc
kern.maxproc: 1044
```

要□置一个指定的□量，直接用 *variable=value* □□的□法：

```
# sysctl kern.maxfiles=5000
kern.maxfiles: 2088 -> 5000
```

sysctl □量的□置通常是字符串、数字或者布□型。 (布□型用 1 来表示'yes'，用 0 来表示'no')。

如果□想在□次机器□□□自□□置某些□量， 可将它□加入到文件 /etc/sysctl.conf 之中。更多信息，□参□手册□ sysctl.conf(5) 及 sysctl.conf。

12.11.1. 只□的 sysctl(8)

有□可能会需要修改某些只□的 sysctl(8) 的□。 尽管有□不得不□□做， 但只有通□(重新)□□才能□到□□的目的。

例如一些膝上型□□的 cardbus(4) □□不会探□内存□□，并且□生看似于□□的□□：

```
cbb0: Could not map register memory
device_probe_and_attach: cbb0 attach returned 12
```

像上面的□□通常需要修改一些只□的 sysctl(8) 默□□置。要□□□点，用□可以在本地的 /boot/loader.conf.local 里面放一个 sysctl(8) "OIDs"。那些□置定位在 /boot/defaults/loader.conf 文件中。

修□上面的□□用□需要在□才所□的文件中□置 hw.pci.allow_unsupported_io_range=1。□在 cardbus(4) 就会正常的工作了。

12.12. □整磁□

12.12.1. Sysctl □量

12.12.1.1. vfs.vmiodirenable

vfs.vmiodirenable sysctl □量可以□置成0(□)或者1(□)；默□是1。 □个□量控制目□是否被系□□存。大多数目□是小的， 在系□中只使用□个片断(典型的是1K)并且在□存中使用的更小 (典型的是512字□)。当□个□量□置□□□(0) □， □存器□□□存固定数量的目□，即使□有很大的内存。 而将其□□ (□置□1) □， □允□□存器用 VM □面□存来□存□些目□，□所有可用内存来□存目□。 不利的是最小的用来□存目□的核心内存是大于 512 字□的物理□面大小(通常是 4k)。 我□建□如果□在□行任何操作大量文件的程序□保持□个□□打□的默□□。 □些服□包括 web □存，大容量□件系□和新□系□。 尽管可能会浪□一些内存，但打□□个□□通常不会降低性能。 但□是□□□□一下。

12.12.1.2. vfs.write_behind

vfs.write_behind sysctl □量默□是 1 (打□)。 它告□文件系□□被收集□的□候把内容写□介□，典型的是在写入大的□□的文件□。 主要的想法是， 如果可能□ I/O 性能会□生□面影□□， □尽量避免□□冲□存被未同□□冲区充□。 然而它可能降低□理速度并且在某些情况下□可能想要□□它。

12.12.1.3. vfs.hirunningspace

vfs.hirunningspace sysctl □量决定了在任何□定情况下， 有多少写 I/O 被排□□列以□系□的磁□控制器。 默□□一般是足□的，但是□有很多磁□的机器来□□可能需要把它□置成 4M 或 5M。注意□个□置成很高的□(超□□存器的写□限)会□致坏的性能。 不要盲目的把它□置太高！高的数□会□致同□□生的□操作的□延。

sysctl 中□有□多与 buffer cache 和 VM□面 cache 有□的□， 一般不推□修改它□。 虚□内存系□已□能□很好地□行自□□整了。

12.12.1.4. vm.swap_idle_enabled

vm.swap_idle_enabled sysctl □量在有很多用□□入、□□系□和有很多空□□程的大的多用□系□中很有用。 □些系□注重在空□的内存中□□生□□□力的□理。通□ vm.swap_idle_threshold1 和 vm.swap_idle_threshold2 打□□个特性并且□整交□滞后 (在空□□)允□□降低内存□中空□□程的□先□，从而比正常的出□ (pageout) 算法更快。□□出□守□□程□来了□助。 除非□需要否□不要把□个□□打□，因□□所□衡的是更快地□入内存，因而它会吃掉更多的交□和磁□□□。在小的系□上它会有决定性的效果， 但是在大的系□上它已□做了合□的□面□度□个□□允□ VM 系□容易的□全部的□程□出内存。

12.12.1.5. hw.ata.wc

FreeBSD 4.3 中默□将 IDE 的写□存□掉了。 □会降低到 IDE 磁□用于写入操作的□□， 但我□□□□有助于避免硬□厂商所引入的， 可能引致□重的数据不一致□□。 □□□□□□上是由于 IDE 硬□就写操作完成□件事的不□□□致的。当□用了 IDE 写入□存□， IDE 硬□□□器不但不会按□序将数据写到□上， 而且当磁□承受重□□，它甚至会自作主□地□推□某些□的□□写操作。 □□一来， 在系□□生崩□或掉□□， 就会□致□重的文件系□□坏。基于□些考□， 我□将 FreeBSD 的默□配置改成了更□安全的禁用 IDE 写入□存。 然而不幸的是， □□做□致了性能的大幅降低， 因此在后来的□行版中□个配置又改□默□□用了。 □可以通□□察 hw.ata.wc sysctl □量，来□□□的系□中所采用的默□□。 如果 IDE 写□存被禁用， □可以通□将内核□量□置□ 1 来□用它。 □一操作必□在□□□通□ boot loader 来完成。 在内核□□之后□□□□做是没有任何作用的。

要了解更多的信息，□□□ ata(4)。

12.12.1.6. SCSI_DELAY (kern.cam.scsi_delay)

SCSI_DELAY 内核配置会□短系□□□□□。 默□□在系□□□□程中有 15 秒的□延□□， □是一个足□多且可□的□。把它□少到 5 通常也能工作(特□是□代的□□器)。 □可以在系□引□□□整引□加□器□量 kern.cam.scsi_delay 来改□它。需要注意的是， 此□使用的□位是 毫秒 而 不 是 秒 。

12.12.2. Soft Updates

tunefs(8) 程序能□用来很好的□整文件系□。 □个程序有很多不同的□□，但是□在只介□ Soft Updates 的打□和□□，□□做：

```
# tunefs -n enable /filesystem
# tunefs -n disable /filesystem
```

在文件系□被挂□之后不能用 tunefs(8) 来修改。打□ Soft Updates 的最佳□机是在□用□模式下任何分区被挂□前。

Soft Updates □大地改善了元数据修改的性能， 主要是文件□建和□除，通□内存□存。我□建□□在所有的文件系□上使用 Soft Updates。□□知道 Soft Updates 的□点：首先， Soft Updates 保□了崩□后的文件系□完整性，但是很可能有几秒□ (甚至一分□！) 之前的数据没有写到物理磁□。如果□的系□崩□了□可能会□失很多工作。 第二，SoftUpdates 推□文件系□□的□放□□。如果在文件系□ (例如根文件系□)快□了的情况下□系□□行大□模的升□比如 `make installworld`， 可能会引起磁□空□不足从而造成升□失□。

12.12.2.1. Soft Updates 的□□□料

有□□□□的方法来把文件系□的元数据 (meta-data) 写入磁□。 (Meta-data更新是更新□似 inodes 或者目□□些没有内容的数据)

从前，默□方法是同□更新□些元数据(meta-data)。 如果一个目□改□了，系□在真正写到磁□之前一直等待。文件数据□存(文件内容)在□之后以非同□形式写入。 □□做有利的一点是操作安全。如果更新□□生□□，元数据(meta-data) 一直□于完整状□。文件要不就被完整的□建要不根本就不□建。 如果崩□□□不到文件的数据□，fsck(8) 可以□到并且依□把文件大小□置□ 0 来修□文件系□。 □外，□□做既清楚又□□。缺点是元数据(meta-data)更新很慢。例如 `rm -r` 命令，依次触及目□下的所有文件， 但是□个目□的改□(□除一个文件)都要同□写入磁□。 □包含它自己更新目□，inode 表和可能□文件分散的□的更新。 同□□□出□大的文件操作上(比如 `tar -x`)。

第二□方法是非同□元数据更新。□是 Linux/ext2fs 和 *BSD ufs 的 `mount -o async` 默□的方法。所有元数据更新也是通□□存。 也就是它□会混合在文件内容数据更新中。 □个方法的□点是不需要等待□个元数据更新都写到磁□上， 所以所有引起元数据更新大的操作比同□方式更快。同□，□个方法也是清楚且□□的，所以代□中的漏洞□□很小。 缺点是不能保□文件系□的状□一致性。如果更新大量元数据□失□ (例如掉□或者按了重□按□)，文件系□会□在不可□知的状□。 系□再□□□没有机会□□文件系□的状□；inode 表更新的□候可能文件的数据□已□写入磁□了但是相□□的目□没有，却不能用 `fsck` 命令来清理(因□磁□上没有所需要的信息)。 如果文件系□修□后□坏了，唯一的□□是使用 newfs(8) 并且从□□中恢□它。

□个□□通常的解决□法是使用 *dirty region logging* 或者 *journaling* 尽管它不是一□的被使用并且有□候□用到其他的事□□□中更好。 □□方法元数据更新依然同□写入，但是只写到磁□的一个小区域。 □后他□将会被移□到正□的位置。因□□□区很小， 磁□上接近的区域磁□不需要移□很□的距□，所以□些比写同□快一些。 □外□个方法的□□性有限，所以出□□□的机会也很少。缺点是元数据要写□次 (一次写到□□区域，一次写到正□的区域)。正常情况下， 悲□的性能可能会□生。从□一方面来□， 崩□的□候所有未□生的元数据操作可以很快的在系□□□之后从□□中恢□□来。

Kirk McKusick，伯克利 FFS 的□□者，用 Soft Updates 解决了□个□□：元数据更新保存在内存中并且按照排列的□序写入到磁□ ("有序的元数据更新")。□□的□果是，在繁重的元数据操作中， 如果先前的更新□在内存中没有被写□磁□，后来的更新就会捕捉到。所以所有的目□操作在写□磁□的□候首先在内存中□行 (数据□按照它□的位置来排列，所以它□不会在元数据前被写入)。 如果系□崩□了□将□致一个固定的 "日志回朔"： 所有不知如何写入磁□的操作都像没有□生□一□。文件系□的一致性保持在 30 到 60 秒之前。它保□了所有正在使用的□源被□□例如□和 inodes。崩□之后， 唯一的□源分配□□是一个□□是"空□"的□源的□源被□□□"使用"。 fsck(8) 可以□出

□□情况并且□放不再使用的□源。它□于忽略崩□后用 mount -f □制挂上的文件系□的□□状□是安全的。 □了□放可能没有使用的□源，fsck(8) 需要在□后的□□□行。一个主意是用 后台 *fsck*：系□□□的□候只有一个文件系□的 快照 被□□下来。fsck 可以在□后□行。所有文件系□可以在"有□□"的□候被挂接， 所以系□可以在多用□模式下□□。接着，后台 fsck 可以在所有文件系□需要的□候□□来□放可能没有使用的□源。 (尽管□□，不用 Soft Updates 的文件系□依然需要通常的 fsck。)

f它的□点是元数据操作几乎跟非同□一□快 (也就是比需要□次元数据写操作的 *logging* 更快)。缺点是代□的□□性(意味着□于□失用□敏感数据有更多的□□) 和高的内存使用量。□外它有些特点需要知道。崩□之后，文件系□状□会"落后"一些。同□的方法用 fsck 后在一些地方可能□生一些零字□的文件， □些文件在用 Soft Updates 文件系□之后不会存在， 因□元数据和文件内容根本没有写□磁□(可能□生在□行 rm 之后)。□可能在文件系□上安装大量数据□候引□□□， 没有足□的剩余空□来□次存□所有文件。

12.13. □整内核限制

12.13.1. 文件/□程限制

12.13.1.1. kern.maxfiles

kern.maxfiles 可以根据系□的需要□当□□。 □个□量用于指定在系□中允□的文件描述符的最大数量。当文件描述符表□的□候， file: table is full 会在系□消息□冲区中反□出□， □可以使用 dmesg 命令来□察□一□象。

□个打□的文件、 套接字和管道， 都会占用一个文件描述符。 在大型生□服□器上， 可能会□易地用掉数千个文件描述符， 具体用量取决于服□的□型和并行□□的服□数量。

在早期版本的 FreeBSD 中， kern.maxfiles 的默□□， 是根据□内核配置文件中的 maxusers □□□算的。kern.maxfiles □个数□， 会随 maxusers 成比例地□□。 当□□定制的内核□， 按照□系□的用途来修改□个□是个好主意。 □个数字同□□决定内核的□多□□的限制□。 有□， 尽管并不会真的有 256 个用□同□□接一台生□服□器， 但□于高□□的 web 服□器而言， 却可能需要与之□似的□源。

□量 kern.maxusers 会在系□□□□， 根据可用内存的尺寸□行□算， 在内核□始□行之后， 可以通□只□的 kern.maxusers sysctl □量□来□行□察。 有些情况下， 可能会希望使用更大或更小一些的 kern.maxusers，它可以以加□器□量的形式□行配置； □似 64、 128 和 256 □□的□都并不□□。 我□不推□使用超□ 256 的□，除非□需要巨量的文件描述符； 根据 kern.maxusers 推算默□□的那些□量， 一般都可以在引□甚至□行□通□/boot/loader.conf (□参□ loader.conf(5) □机手册或 /boot/defaults/loader.conf 文件来□得相□的指□) 或□篇文□的其余部分所介□的方式来□整。

在□早的版本中， 如果□明□地将 maxusers □置□ 0， □系□会自□地根据硬件配置来□定□个□。。 在 FreeBSD 5.X 和更高版本中， maxusers 如果不指定的□， 就会取默□□ 0。 如果希望自行管理 maxusers，□□配置一个不低于 4 的□， 特□是使用 X Window System 或□□□件的□候。 □□做的原因是， maxusers 所决定的一个最□重要的表的尺寸会影□最大□程数， □个数□将是 20 + 16 * maxusers。 因此如果将 maxusers □置□ 1， □就只能同□□行 36 个□程， □□包括了 18 个左右的系□引□□□□的□程， 以及 15 个左右的，在□□□ X Window System □所引□的□程。 即使是□□的任□， 如□□□机手册， 也需要□□多至九个的□程，用以□□、 解□□， 并□示它。 将 maxusers □□ 64 将允□□同□□行最多 1044 个□程， □几乎足以□足任何需要了。不□， 如果□看在□□其它程序， 或□行用以支持大量用□的服□ (例如 ftp.FreeBSD.org) □， 看到令人担□的□□， 就□□提高□一数□， 并重新□□内核。

maxusers 并 不能 限制□□能□登□到□系□上来的用□的数量。 它的主要作用是根据□可能支持的用□数量来□一系列系□数据表□置合理的尺寸， 以便提供支持他□所需□行的□程□源。

12.13.1.2. kern.ipc.somaxconn

kern.ipc.somaxconn sysctl □量 限制了接收新 TCP □接□听□列的大小。□于一个□常□理新□接的高□□ web服□□境来□，默□的 128 太小了。 大多数□境□个□建□□加到 1024 或者更多。 服□□程会自己限制□听□列的大小(例如 sendmail(8) 或者 Apache)， 常常在它□的配置文件中有□置□列大小的□□。 大的□听□列□防止拒□服□攻□也会有所□助。

12.13.2. 网□限制

NMBCLUSTERS 内核配置□□指出了系□可用的网□Mbuf的数量。 一个高流量的服□器使用一个小数目的网□□存会影□ FreeBSD 的性能。 □个 cluster 可能需要2K内存，所以一个1024的□需要在内核中□网□□存保留2M内存。 可以用□□的方法□算出来需要多少网□□存。 如果□有一个同□□生1000个以上□接的web服□器， 并且□个□接用掉16K接收和□送□存， 就需要大概32M网□□存来□保web服□器的工作。 一个好的□□□算方法是乘以2，所以2x32Mb/2Kb=64MB/2kb=32768。 我□建□在有大量内存的机器上把□个□□置在4096到32768之□。 没有必要把它□置成任意太高的□，它会在□□□引起崩□。 netstat(1) 的 -m □□可以用来□察网□cluster使用情况。

kern.ipc.nmbclusters 可以用来在□□□刻□□□个。 □□在旧版本的 FreeBSD 需要使用 NMBCLUSTERS config(8) □□。

□常使用 sendfile(2) 系□□用的繁忙的服□器， 有必要通□ NSFBUFS 内核□□或者在 /boot/loader.conf (□看 loader(8) 以□得更多□□) 中□置它的□来□□ sendfile(2) □存数量。 □个参数需要□□的普通原因是在□程中看到 sfbufa 状□。sysctl kern.ipc.nsfbufs □量在内核配置□量中是只□的。 □个参数是由 kern.maxusers 决定的，然而它可能有必要因此而□整。

即使一个套接字被□□成非阻塞，在□个非阻塞的套接字上呼叫 sendfile(2) 可能□致 sendfile(2) 呼叫阻塞直到有足□的 struct sf_buf 可用。

12.13.2.1. net.inet.ip.portrange.*

net.inet.ip.portrange.* sysctl □量自□的控制□定在 TCP 和 UDP 套接字上的端口□□。 □里有三个□□：一个低端□□，一个默□□□和一个高端□□。 大多数网□程序分□使用由 net.inet.ip.portrange.first 和 net.inet.ip.portrange.last 控制的从 1024 到 5000 的默□□□。端口□□用作□外□接，并且某些情况可能用完系□的端口， □□常□生在□行一个高□荷 web 代理服□器的□候。 □个端口□□不是用来限制主要的例如 web 服□器□入□接或者有固定端口例如□件□□□外□接的。 有□□可能用完了端口，那就建□□当的□加 net.inet.ip.portrange.last。 10000、20000 或者 30000 可能是□当的□。 更改端口□□的□候也要考□到防火□。 一些防火□会阻止端口的大部分□□ (通常是低□□的端口)并且用高端口□行□外□接(-)。 基于□个□□建□不要把 net.inet.ip.portrange.first □的太小。

12.13.2.2. TCP □□□延(Bandwidth Delay Product)

限制 TCP □□延□□和 NetBSD 的 TCP/Vegas □似。 它可以通□将 sysctl □量 net.inet.tcp.inflight.enable □置成 1 来□用。 系□将□□□算□一个□接的□□延□□， 并将排□的数据量限制在恰好能保持最□吞吐量的水平上。

□一特性在□的服□器同□向使用普通□制解□器， 千兆以太网， 乃至更高速度的光与网□□接 (或其他□□延

□□很大的□接) 的□候尤□重要， 特□是当□同□使用滑□□□放， 或使用了大的□送□口的□候。 如果□用了□个□□，□□□□把 net.inet.tcp.inflight.debug □置□ 0 (禁用□□)， □于生□□境而言， 将 net.inet.tcp.inflight.min □置成至少 6144 会很有好□。 然而， 需要注意的是， □个□□置□大事□上相当于禁用了□接□□延□□限制功能。□个限制特性□少了在路由和交□包□列的堵塞数据数量， 也□少了在本地主机接口□列阻塞的数据的数量。在少数的等候□列中、 交互式□接，尤其是通□慢速的□制解□器，也能用低的 往返□□操作。但是，注意□只影□到数据□送 (上□/服□端)。□数据接收(下□)没有效果。

□整 net.inet.tcp.inflight.stab 是 不 推□的。 □个参数的默□□是 20， 表示把 2 个最大包加入到□□延□□□口的□算中。 □外的□口似的算法更□□定， 并改善□于多□网□□境的相□能力， 但也会□致慢速□接下的 ping □□□□ (尽管□是会比没有使用 inflight 算法低□多)。 □于□些情形， □可能会希望把□个参数□少到 15， 10， 或 5； 并可能因此而不得不□少 net.inet.tcp.inflight.min (比如□， 3500) 来得到希望的效果。 □少□些参数的□， 只□作□最后不得已□的手段来使用。

12.13.3. 虚□内存

12.13.3.1. kern.maxvnodes

vnode 是□文件或目□的一□内部表□。 因此， □加可以被操作系□利用的 vnode 数量将降低磁□的 I/O。一般而言， □是由操作系□自行完成的， 也不需要加以修改。 但在某些□候磁□ I/O 会成□瓶□， 而系□的 vnode 不足， □□一配置□被□加。 此□需要考□是非活□和空□内存的数量。

要□看当前在用的 vnode 数量：

```
# sysctl vfs.numvnodes
vfs.numvnodes: 91349
```

要□看最大可用的 vnode 数量：

```
# sysctl kern.maxvnodes
kern.maxvnodes: 100000
```

如果当前的 vnode 用量接近最大□， □将 kern.maxvnodes □□大 1,000 可能是个好主意。 □□□□□看 vfs.numvnodes 的数□， 如果它再次攀升到接近最大□的程度， 仍需□□提高 kern.maxvnodes。 在 top(1) 中□示的内存用量□有□著□化， 更多内存会□于活□ (active) 状□。

12.14. 添加交□空□

不管□□□得如何好，有□候系□并不像□所期待的那□□行。 如果□□□需要更多的交□空□，添加它很□□。 有三□方法□加交□空□：添加一□新的硬□□□器、通□ NFS 使用交□空□和在一个□有的分区上□建一个交□文件。

要了解□于如何加密交□区， 相□配置， 以及□什□要□□做， □参□手册的 □交□区□行加密。

12.14.1. 在新的硬□□□器上使用交□空□

□是添加交□空□最好的方法， 当然□了□到□个目的需要添加一□硬□。 □竟□□是可以使用□一□磁□。如果能□□做， 重新□□一下手册中□于交□空□的 初□配置 来了解如何最□地安排交□空□。

12.14.2. 通□ NFS 交□

除非没有可以用作交□空□的本地硬□□， 否□不推□□使用 NFS 来作□交□空□使用。 NFS 交□会受到可用网□□□限制并且□加 NFS 服□器的□担。

12.14.3. 交□文件

□可以□建一个指定大小的文件用来当作交□文件。 在我□的例子中我□将会使用叫做 /usr/swap0 的 64MB 大小的文件。当然□也可以使用任何□所希望的名字。

例 *9.* 在 *FreeBSD* 中□建交□文件

1. □□□的内核配置包含虚□磁□(Memory disk)□□ (md(4))。它在 GENERIC 内核中是默□的。

```
device   md   # Memory "disks"
```

2. □建一个交□文件(/usr/swap0)：

```
# dd if=/dev/zero of=/usr/swap0 bs=1024k count=64
```

3. □予它(/usr/swap0)一个□当的□限：

```
# chmod 0600 /usr/swap0
```

4. 在 /etc/rc.conf 中□用交□文件：

```
swapfile="/usr/swap0"   # Set to name of swapfile if aux swapfile desired.
```

5. 通□重新□□机器或下面的命令使交□文件立刻生效：

```
# mdconfig -a -t vnode -f /usr/swap0 -u 0 && swapon /dev/md0
```

12.15. □源和□源管理

BIOS 接口管理，例如可□拔 *BIOS (PNPBIOS)*或者高□□源管理*(APM)* 等等。□源和□源管理是□代操作系□的□□□成部分。 例如□可能当系□温度□高的□候□□的操作系□能□□到 (并且可能提醒□)。

以有效的方式利用硬件□源是非常重要的。 在引入 ACPI 之前， 管理□源使用和系□散□□操作系□是很困□的。 硬件由 BIOS □行管理， 因而用□□□源管理配置的控制和□看都比□困□。 一些系□通□ 高□□源管理 *(APM)* 提供了有限的配置能力。 □源和□源管理是□代操作系□的一个□□□件。 例如， □可能希望操作系□□□系□的一些限制， 例如系□的温度是否超出了□期的□□速度 (并在需要□□出警告)。

在 FreeBSD 使用手册的□一章□，我□将提供 ACPI 全面的信息。 参考□料会在末尾□出。

12.15.1. 什□是 ACPI？

高□配置和□源接口 (ACPI) 是一个□界□准的硬件□源和□源管理接口 (因此而得名) 。它是 操作系□控制的配置和□源管理*(Operating System-directed configuration and Power Management)*，也就是□，它□操作系□(OS)提供了更多的控制和□性。 在引入 ACPI 之前， □代操作系□使得目前即□即用接口的局限性更加 "凸□" 出来。 ACPI 是 APM(高□□源管理) 的直接□承者。

12.15.2. 高□□源管理 (APM) 的缺点

高□□源管理 *(APM)* 是一□基于系□目前的活□控制其□源使用的机制。 APM BIOS 由 (系□的) 制造商提供，并且是硬件平台□属的。 在 OS 中的 APM □□作□中介来□□ *APM* □件接口， 从而□□□□源使用的管理。 在 2000 年或更早的□期生□的□算机系□， 仍需要使用 APM。

APM 有四个主要的□□。 首先， □源管理是通□ (制造商□属的) BIOS □□的， 而 OS □完全不了解其□□。 例如，用□在 APM BIOS 中□置了硬□□□器的空□等待数□， 当超□□一空□□□的限制□， 它 (BIOS) 将会□慢硬□□□器的速度， 而不会征求 OS 的同意。 第二， APM □□是嵌入 BIOS 的， 因此它是在 OS 的控制之外□□的。□意味着用□只能通□通□刷新他□ ROM 中的 APM BIOS 才能□解决某些□□； 而□是一个很危□的操作， 因□它可能使系□□入一个无法恢□的状□。 第三， APM 是一□制造商□属的技□， 也就是□有很多第三方的 (重□的工作) 以及 bugs， 如果在一个制造商的 BIOS 中有， 也未必会在其他的□品中解决。 最后但□不是最小的□□， APM BIOS 没有□□□□□的□源策略提供足□的余地， 也无法□□能□非常□合具体机器的策略。

即□即用 *BIOS (PNPBIOS)* 在很多□候都是不可□的。 PNPBIOS 是 16-位 的技□， 因此 OS 不得不使用 16-位模□才能□与 PNPBIOS 的方法 "接口"。

FreeBSD APM □□在 apm(4) 手册□中有描述。

12.15.3. 配置 ACPI

默□情况下， acpi.ko □□， 会在系□引□□由 loader(8) 加□， 而 不□ 直接□□□内核。 □□做的原因是模□操作起来更方便， 例如， 无需重新□□内核就可以切□到□一个 acpi.ko 版本。 □□可以□□□□得更□□一些。□一个原因是， □多□候在□□已□□□之后再□□ ACPI 可能会有些□□。 如果□遇到了□□， 可以全面禁用 ACPI。 □个□□不□， 目前也无法卸□， 因□系□□□通□它与□多不同的硬件□行交互。 ACPI 可以通□在 /boot/loader.conf 中配置或在 loader(8) 提示符□配置 `hint.acpi.0.disabled="1"` 来禁用。

ACPI 和 APM 不能共存， 相反， 它□□分□使用。 后加□的□□如果□□系□中已□□行了其中的一个， 便会停止□行。

ACPI 可以用来□系□□入休眠模式， 方法是使用 acpiconf(8) 的 `-s` 参数， 加上一个 `1-5` 的数字。 多数用□会希望使用 `1` 或 `3` (挂起到 RAM)。 而 `5` □会□系□□行与下列命令效果□似的□□机：

```
# halt -p
```

除此之外， □有一些通□ sysctl(8) 提供的□□。 □参□□机手册 acpi(4) 和 acpiconf(8) 以□得更多信息。

12.16. 使用和□□ FreeBSD ACPI

ACPI 是一□全新的□□□□、 管理□源使用、 以及提供□去由 BIOS 管理的□□不同硬件的□准化方法。 □ ACPI 在各□系□上都能正□使用的工作一直在□行， 但□多主板的 *ACPI* 机器□言 (AML) 字□代□中的 bug， FreeBSD

的内核中子系□□□的不完善， 以及 Intel® ACPI-CA 解□器中的 bug 仍然不□会出□。

□□文□期望能□□助□□助 FreeBSD ACPI 的□□人□来□到□所□察到的□□的根源， 并通□□□□到其解决方法。 感□□□□□□文□， 我□也希望能□解决□的系□上的□□。

12.16.1. 提交□□信息

在提交□□之前， □□□□已□在□行最新的 BIOS 版本， 此外， 也包括嵌入式控制器的固件版本。

如果□希望提交一个□□， □□保将下述信息□到 freebsd-acpi@FreeBSD.org:

- □□行□的描述， 包括系□□型、型号，以及任何触□□□的相□信息。 □外， □注意尽可能准□地描述□一□□是否□□是陌生的。
- 在 "boot -v" 之后得到的 dmesg(8) □出， 以及任何在重□ bug □出□的□□信息。
- 在禁用了 ACPI 之后的 "boot -v" 的 dmesg(8) □出， 如果□□□禁用 ACPI 能□□助消除□□。
- 来自 `fsysctl hw.acpi`的□出。 □也是□到□的系□所提供的功能的一□好□法。
- 能□得到□的 *ACPI Source Language* (ASL) 的 URL。 不要 把 ASL 直接□到□件列表中， 因□它□可能非常大。 □了得到 ASL □可以□行□个命令：

```
# acpidump -dt > name-system.asl
```

(把 *name* 改□□的登□名， 并把 *system* 改□□的硬件制造商及其型号。 例如： njl-FooCo6000.asl)

□多□□者也会□□ FreeBSD-CURRENT □件列表 但□是□□到 FreeBSD ACPI □件列表 □□它会被更多人看到。 □耐心等待， 因□我□都有全□的其他工作。 如果□的 bug 不是□而易□的， 我□可能会要求□通□ send-pr(1) 来提交一个 PR。 在□入 PR □，□将同□的信息包含□去。 □将□助我□来追踪和解决□□。 不要在□ FreeBSD ACPI □件列表 写信之前□送 PR 因□我□把它当作已知文体的□忘□而不是□告机制。 □的□□很可能已□被其他人□告□了。

12.16.2. 背景

ACPI 存在于采用 ia32 (x86)、 ia64 (安□)、 以及 amd64 (AMD) 架□的所有□代□算机上。 完整的□准具有大量的各式功能， 包括 CPU 性能管理、 □源控制、 温度□控、 □池系□、 嵌入式控制器以及□□枚□。 □大多数系□□□比完整□准的功能要少一些。 例如， □面系□通常只□□□□枚□部分， 而□□本□通常支持降温和□源管理功能。 □□本通常□提供休眠和□醒支持， 并提供与此□□的□□功能。

符合 ACPI 的系□中有□多□件。 BIOS 和芯片□制造商提供一些固定的表 (例如， FADT) 在存□器中， 以提供□似 APIC 映射 (用于 SMP)、 配置寄存器、 以及□□的配置□等等。 □外， 一个字□代□ (bytecode) 表 (系□区□描述表DSDT) □提供了通□□状命名空□来指定□□及其功能的方法。

ACPI □□必□要□理固定表， □□字□□解□器， 并修改□□程序和内核， 以接受来自 ACPI 子系□的信息。 □于 FreeBSD， Intel® 提供了一个解□器 (ACPI-CA)， 它在 Linux 和 NetBSD 也可以使用。 ACPI-CA 源代□可以在 src/sys/contrib/dev/acpica □到。 用于在 FreeBSD 中允□ ACPI-CA 正□□□的代□□在 src/sys/dev/acpica/Osd。 最后， 用于□□ ACPI □□的□□可以在 src/sys/dev/acpica □到。

12.16.3. 常□□□

要□ ACPI 正常工作， 它的□一部分都必□工作正常。 下面是一些常□的□□， 按照出新的□繁程度排序， 并□出了一些□□或修正它□的方法。

12.16.3.1. 鼠□□□

某些□候， □醒操作会□致鼠□不再正常工作。 已知的□□□一□□的方法， 是在 /boot/loader.conf 文件中添加 hint.psm.0.flags="0x3000" □置。 如果□□做不能解决□□， □考□按前面介□的方法提交□□□告。

12.16.3.2. 休眠/□醒

ACPI 提供了三□休眠到 RAM (STR) 的状□， S1-S3， 以及一个休眠到磁□的状□ (STD)， 称作 S4。 S5 是 "□□机" 同□也是系□接好□源但没有□机□的正常状□。 S4 □□上可以用□□不同的方法来□□。 S4BIOS 是一□由 BIOS □助的挂起到磁□方法， 而 S4OS □是完全由操作系□□□的。

可以使用 sysctl hw.acpi 来□看与休眠有□的□目。 □里是我的 Thinkpad 上得到的□果。

```
hw.acpi.supported_sleep_state: S3 S4 S5
hw.acpi.s4bios: 0
```

□表示我可以使用 acpiconf -s 来□□ S3， S4OS， 以及 S5。 如果 s4bios 是一 (1)， □可以使用 S4BIOS 来代替 S4OS。

当□□休眠/□醒□， 从 S1 □始， 如果它被支持的□。 □个状□是最可能正常工作的状□， 因□它不需要太多的□□支持。 没有人□□ S2 但如果□有它的支持， □□□和 S1 □似。 下一件□得□□的是 S3。 □是最深的 STR 状□， 并需要一系列□□的支持才能□正常地重新初始化□的硬件。 如果□在□醒系□□遇到□□， □不要吝惜□□件□ FreeBSD ACPI □件列表 □件列表， 尽管不要指望□□一定会很快解决， 因□有□多□□程序/硬件需要□行更多的□□和改□。

休眠和□醒操作最常□的□□是某些□□□□程序不会保存、 恢□或正□地重新初始化其固件、 寄存器或□□内存。 □□□□□些□□□， 首先可以□□：

```
# sysctl debug.bootverbose=1
# sysctl debug.acpi.suspend_bounce=1
# acpiconf -s 3
```

□个□□会模□休眠和恢□□程而不真的□入 S3 状□。 有□， □会用□□方式很容易地□住□□ (例如， □失固件状□、 □□ watchdog 超□， 以及一直重□等)。 注意系□不会真的□入 S3 状□， □意味着□些□□可能不会掉□， 而□多□□在完全不提供休眠和恢□方法□仍可正常工作， 而不像使用真的 S3 状□那□。

□□的情况□需要更多的硬件， 例如用于串口控制台的串口/□， 以及用于 dcons(4) 的火□口/□和内核□□技能。

□了□助隔□□□， □在内核中□去尽可能多的□□。 如果□□做能□解决□□， □□□逐个加□□□直到□□再次出□。 通常□□□的□□程序如 nvidia.ko、 X11 □示□□， 以及 USB 的□□最多， 而以太网□的□□□通常工作的很好。 如果□能□通□加□和卸□□□使系□正常工作， □可以通□将□当的命令放到 /etc/rc.suspend 和 /etc/rc.resume 来将□个□程自□化。 在□□个文件中有一个注□掉的卸□和加□□□程序的例子供□参考。 □外□□可以将 hw.acpi.reset_video □置□零 (0)， 如果□的□示在□醒之后□得很混乱。 此外□□可以□□更□或更短的

hw.acpi.sleep_delay □看看是否有所助益。

□一件□得一□的事情是使用一个比□新的包含 ACPI 支持的 Linux □行版来□□看他□的 休眠/□醒 功能是否在同□的硬件上能□正常工作。 如果在 Linux 下正常， □很可能是 FreeBSD □□程序的□□， 而隔□□□并□到存在□□的□□有助于解决它。 需要注意的是 ACPI 的□□人□通常并不□□其他□□ (例如 声音、 ATA， 等等) 因此如果最□□□是□□的□□最好□是□到 FreeBSD-CURRENT □件列表 □件列表并□□□□程序的□□者。 如果□喜□冒□， □可以加一些 printf(3) 到有□□的□□中， 以□到它的恢□功能□生□□的位置。

最后， □□看禁用 ACPI 并代之以□用 APM。 如果 休眠/□醒 能□在 APM 下正常工作， 使用 APM 可能会更好， 特□是□于□老的硬件 (2000年以前)。 硬件制造商需要一些□□来□老硬件的 ACPI 工作正常， 而 ACPI 的□□十之八九是 BIOS 中的毛病引□的。

12.16.3.3. 系□停止□□ (□□或永久性地)

□大多数系□停止□□是由于未能及□□□中断或□生了中断□暴□致的。 芯片□有很多□□最□会溯源到 BIOS 如何在引□系□之前配置中断， APIC (MADT) 表的正□性， 以及 系□控制中断 (SCI) 如何路由。

通□察看 vmstat -i 的□出中包括 acpi0 的那一行可以区分中断□暴和未能及□□□中断。 如果□秒□数器□□的速度多于一□个， □□是遇到了中断□暴。 如果系□停止了□□， □可以□□停止内核并□入 DDB (在控制台上按 CTRL + ALT + ESC) 并□入 show interrupts。

□理中断□□的救命稻草是□□禁用 APIC 支持， □是通□在 loader.conf 中加入 hint.apic.0.disabled="1" 完成的。

12.16.3.4. 崩□

崩□□于 ACPI 是比□□□的情况， 如果□□， 我□将会非常重□并很快修□它。 □要做的第一件事是□法隔□出能□重□崩□ (如果可能的□) 的操作并□取一□□用堆□。 □□用 options DDB 并□置串行控制台 (参□ 通□串口□□入DDB□□器) 或配置一个 dump(8) 分区。 □将在 DDB 中通□ tr 得到□用堆□。 如果□只能用手抄的方法□□它， 一定要□下□五 (5) 行和最后五 (5) 行。

然后， □□通□在□□□禁用 ACPI 来隔□故障。 如果□□做能□正常工作， □通□□置 debug.acpi.disable 的那□数□来隔□具体是□个 ACPI 子系□的□□。 □参□ acpi(4) □机手册中□出的那些例子。

12.16.3.5. 系□在休眠或□机之后又□□了

首先□□□在 loader.conf(5) 中□置 hw.acpi.disable_on_poweroff="0"。 □将□ ACPI 不再在□机□程中禁用一些事件。 基于同□的原因， 一些系□需要把□个□□置□ "1" (□是默□□)。 □通常能□修□在休眠或□机□立即再次□□的□□。

12.16.3.6. 其他□□

如果□有 ACPI 的其他□□ (同 docking station □同工作、 无法□□□□， 等等)， □把描述□□□件列表； 不□， □些□□也有可能和 ACPI 中尚未完成的部分有□， 它□可能需要□□才能被□□。 □□点耐心， 并准□□□我□可能会□□□的□丁。

12.16.4. ASL、acpidump， 以及 IASL

最常□的□□是 BIOS 制造商提供的不正□ (甚至完全□□的!) 字□代□。 □通常会以□似下面□□的内核消息□示在控制台上：

```
ACPI-1287: *** Error: Method execution failed [\\_SB_.PCI0.LPC0.FIGD._STA] \\
(Node 0xc3f6d160), AE_NOT_FOUND
```

□多□候， □可以通□将 BIOS 升□到最新版本来解决此□□□。 □大多数控制台消息是无害的， 但如果□有其他□□例如□池工作不正常， □从 AML □始□□□将是一条捷径。 字□代□， 或常□的 AML， 是从一□叫做 ASL 的□言写成的源代□□行□□得到的□果。 AML 一般存放在 DSDT 表中。 要得到□系□的 ASL， 需要使用 acpidump(8)。 需要同□指定 -t (□示固定□的内容) 和 -d (将 AML 反□□成 ASL) □个□□。 □参□ 如何提交□□信息 一□了解如何使用它。

最方便的初□□□是□□重新□□ ASL 来看看是否有□□。 通常可以忽略□一□程中□生的警告， 但□□一般就都是 bug， 它□通常就是□致 ACPI 无法正常工作的原因。 要重新□□□的 ASL， 可以使用下面的命令：

```
# iasl your.asl
```

12.16.5. 修□ ASL

我□的□期目□是□□一个人都能□在不需要任何用□干□的情况下使用 ACPI。 然而， 目前我□仍然在□□□□ BIOS 制造商常□□□的方法。 Microsoft® 解□器 (acpi.sys 和 acpiec.sys) 并不会□格地□□是否遵守了□准， 因此□多只在 Windows® 中□□ ACPI 的 BIOS 制造商很可能永□不会修正他□的 ASL。 我□希望不断地□出并用文□□明 Microsoft® 的解□器到底允□那些不□准的行□， 并在 FreeBSD □行□□的修改使它能□正常工作而不需要用□修正 ASL。 作□一□□□□解□□的方法， 并□助我□□□其行□， □可以手工修正 ASL。 如果□□能□解决□□， □把新旧 ASL 的 diff(1) □□我□， □□我□就有可能□□ ACPI-CA 中的□□行□， 从而不再需要□来手工修正。

下面是一些常□的□□信息， 它□的原因， 以及如何修正。

12.16.5.1. _OS dependencies (_OS 依□)

某些 AML 假定世界是由不同版本的 Windows® □成的。 □可以□ FreeBSD 声称自己是任意 OS 来看一看是否能□修正□□。 比□□□的□法是□置 hw.acpi.osname="Windows 2001" 到 /boot/loader.conf 中，或使用□在 ASL 中□到的其他字符串。

12.16.5.2. Missing Return statements (缺少返回□句)

一些方法可能没按照□准要求的那□□式地返回□。 尽管 ACPI-CA 无法□理它， 但 FreeBSD 提供了一个□□它并允□其暗含地返回□的方法。 □也可以□加一个□式的 Return □句， 如果□知道那里需要返回一个□的□。 要□制 iasl □□ ASL， 需要使用 -f □志。

12.16.5.3. 替□默□的 AML

在定制 your.asl 之后， □可以通□下面的命令□□它：

```
# iasl your.asl
```

可以使用 -f □志来□制□建 AML， 即使在□□□程中□生了□□。 □注意某些□□ (例如， 缺少 Return □句) 会自□被解□器忽略掉。

DSDT.aml 是 iasl 命令的默□□出文件名。 可以加□它来取代□ BIOS 中存在□□的副本 (它仍然存在于□存中), 其方法是按下面的□明□□ /boot/loader.conf：

```
acpi_dsdt_load="YES"
acpi_dsdt_name="/boot/DSDT.aml"
```

一定要把□的 DSDT.aml □制到 /boot 目□中。

12.16.6. 从 ACPI 中□取□□□出信息

ACPI □□程序提供了非常□活的□□机制。 □允□□指定一□子系□， 以及所需要的□□信息。 需要□□的子系□可以按 "layers(□)" 来指定， 并分□ ACPI-CA □件 (ACPI_ALL_COMPONENTS) 和 ACPI 硬件支持 (ACPI_ALL_DRIVERS)。 □□□出的□□程度可以通□ "level(□□度)" 来指定， 其□□是 ACPI_LV_ERROR (只□告□□) 到 ACPI_LV_VERBOSE (□示所有)。 "level" 是一个位掩□因此可以一次□置多个□□， 中□用空格分□。 □□使用中□□□考□使用串行控制台来□□□出， 如果它太□以至于冲掉了控制台消息□冲的□。 不同的□和□出□□度的完整列表可以在 acpi(4) □机手册中□到。

□□□出默□并不□□。 要起用它， □需要在内核□置中添加 options ACPI_DEBUG， 如果□的内核中□入了 ACPI 的□。 □□可以在 /etc/make.conf 中加入 ACPI_DEBUG=1 来在全局起用它。 如果它只是模□， □可以用下面的方法来重新□□ acpi.ko：

```
# cd /sys/modules/acpi/acpi
&& make clean &&
make ACPI_DEBUG=1
```

安装 acpi.ko 到 /boot/kernel and add your 并把所需的□□度和□在 loader.conf 中指定。 □个例子将□用所有 ACPI-CA □件以及所有 ACPI 硬件□□ (CPU、 LID， 等等) 的消息。 只□出□□信息， 也就是最低的□□度。

```
debug.acpi.layer="ACPI_ALL_COMPONENTS ACPI_ALL_DRIVERS"
debug.acpi.level="ACPI_LV_ERROR"
```

如果□需要的信息是由某个特定的事件触□的 (比如□， 休眠之后的□醒)， □可以不修改 loader.conf 而□而使用 sysctl 来在□□和□那个事件准□系□之后再指定□和□□度。 □些 sysctl 的名字和 loader.conf 中的一致。

12.16.7. 参考文献

□于 ACPI 的更多信息可以从下面□些地方□到：

- The FreeBSD ACPI □件列表
- ACPI □件列表存□ http://lists.freebsd.org/pipermail/freebsd-acpi/
- 旧的 ACPI □件列表存□ http://home.jp.FreeBSD.org/mail-list/acpi-jp/
- The ACPI 2.0 □准 http://acpi.info/spec.htm
- FreeBSD 手册□: acpi(4), acpi_thermal(4), acpidump(8), iasl(8), acpidb(8)
- DSDT □□□源. (使用 Compaq 作□例子但通常情况下都很有用。)

Chapter 13. FreeBSD 引□□程

13.1. 概述

□□□□以及加□操作系□的□程被称□"引□□程"， 或者□称□"引□"。 FreeBSD 的引□□程□用□自定□□□提供了很大的伸□性， □可以□□□□不同的操作系□，或者是同一系□的不同版本及内核。

本章将□□介□□能在 FreeBSD 引□□程中□置的配置□□。 □包括了引□内核、探□□□并□□ init(8) 等等之前所□生的所有事情。 □些事□一般□生在文本由白□灰□。

□完□章□将会知道：

- FreeBSD 引□系□里的各□□件， 以及它□之□的交互方式.
- 在 FreeBSD 引□□□各□件配置□□以控制引□□程。
- device.hints(5)的基本知□。

只□用于*x86*

本章只描述了□行于 Intel x86 体系之上的 FreeBSD 的引□□程。

13.2. 引□□□

□□□□及□□和引□操作系□□成了一个有趣的□□境地。 按照定□在操作系□被□□之前□算机是无法完成任何任□的，包括□行磁□上的程序。 如果□算机在没有操作系□的情况下不能□行来自于磁□上的程序而操作系□又是放在磁□上的， 那操作系□是如何□□的□？

在 Munchausen男爵□□□ (The Adventures of Baron Munchausen) □本□中有一个和□个□程□似的故事，一个人掉到了下水管道里， 然后□着拉自己的靴□ (bootstrap) 克服重重困□爬了出来。 在早期文献中，多以□□ *bootstrap* 来指代操作系□的加□机制， 如今它逐□被□写□ "booting"。

在 x86 硬件体系中，基本□入/□出系□ (BIOS) □□加□操作系□， □了做到□一点，BIOS 在磁□上□□主引□□□(MBR)，而 MBR 必□在放置的磁□的特定位置。BIOS 有足□的能力来□入和□行 MBR， 且假使地□□ MBR 能完成加□操作系□的剩余任□， MBR可能需要BIOS的□助。

在MBR中的代□通常被提□引□管理器， 尤其是与用□交互的那□。□一□引□器通常有更多代□位于磁□第一 □道或在操作系□的文件系□中。 (引□管理器有□也被称□*boot loader*， 但是FreeBSD□后面的引□□段才使用□个□□。) 流行的引□管理器包括boot0(亦称Boot Easy，□准的 FreeBSD 引□管理器)、 Grub、GAG，以及LILO。 (只有boot0能装得□MBR。)

如果□只安装了一个操作系□，那□一个□准的 MBR 就足□了。 □个 MBR 先在磁□上搜索可引□的(亦称"活□的")分区， 然后□行分区上的代□以加□操作系□的其它部分。 MBR由fdisk(8)安装，是一个缺省的MBR。相□文件□ /boot/mbr。

如果□在磁□上安装了多个操作系□那□□可以安装一个不同的 引□管理器，它能□示一□操作系□的列表，□能从中□□□□个。 □□的□□引□器将在下一小□中□□。

□□系□的剩余部分被分□三个□段。第一□段由 MBR □行,它只是使□算机□入特定的状□然后□行第二□段。 第二□段□微干得多一些。第三□段完成加□操作系□的任□。 工作被分□三个□段是因□ PC □准□第一第二□段

□行的程序的大小有所限制。 把□些任□□在一起使得 FreeBSD 可以提供更大伸□性的加□器 (loader)。

然后内核□□，它□始探□□□并初始化它□。 一旦内核引□□程完成任□，内核将控制□交□用□□程 init(8)， 它□□磁□是否□于可用状□。init(8) 然后□始用□□□源配置： 加□文件系□□□网□，及粗略地□□所有 FreeBSD 系□加□□□常□行的□程。

13.3. 引□管理器和各引□□段

13.3.1. The Boot Manager

在MBR或引□管理器中的代□有□被提□引□□程的 □段*0*。□一小□便是前面提到引□器中的□□： boot0和LILO。

boot0引□管理器： 由 FreeBSD 的安装程序以及 boot0cfg(8) 所安装的 MBR， 默□基于 /boot/boot0。(程序boot0非常□□， 由于在中的程序只能有446字□□， 分区表和MBR末端的0x55AA□□也要□占一些空□。) 如果□已□安装boot0 并且有多个操作系□在□的硬□上， 那□□如果□安装了 FreeBSD MBR 而且安装了多个操作系□， □会在系□□□□看到□似下面的提示：

例 10. boot0 截屏

```
F1 DOS
F2 FreeBSD
F3 Linux
F4 ??
F5 Drive 1

Default: F2
```

目前已□知道一些其它操作系□，特□是 Windows® ， 会以自己的 MBR 覆□□有 MBR。 如果□生了□□事情，或者□想用 FreeBSD 的 MBR 覆□□有的 MBR，□可以使用以下的命令：

```
# fdisk -B -b /boot/boot0 device
```

device 是要写入 MBR 的□□名，比如 ad0 代表第一个 IDE 磁□，ad2 代表第二个 IDE 控制器上的第一个 IDE 磁□， da0 代表第一个 SCSI 磁□，等等。 抑或，如果□需要一个自行配置的MBR，□使用boot0cfg(8)。

The LILO Boot Manager: 要想安装□个引□管理器并也用来引□FreeBSD， 首先□□Linux，并将以下□□加入到已有的配置文件 /etc/lilo.conf：

```
other=/dev/hdXY
table=/dev/hdX
loader=/boot/chain.b
label=FreeBSD
```

在上面的内容里，使用Linux的□示符指定了FreeBSD的主分区和□□器， 将*X*替□□Linux□□器字母， 将*Y*替□□Linux主分区号。 如果□使用的是 SCSI □□器，□需要将 */dev/hd* 改成 */dev/sd*， □里再次使用了 *XY* 的

□法。 如果□安装的□个系□在同一□□器上，loader=/boot/chain.b □□可以去掉。□在□可以□行 /sbin/lilo -v 使修改生效；□□□屏幕上的消息□□修改。

13.3.2. 第一□段，/boot/boot1，和第二□段， /boot/boot2

概念上，第一，第二□段同属于一个程序，□于磁□的相同区域。但由于空□限制， 它□被分□□部分。可是□□是会一起安装它□。它□由安装器或 bsdlabel(□下文)□制自被□合而成的 /boot/boot。

它□位于文件系□外，引□分区的第一□道，从第一扇区□始。在□里boot0，或者任何其它引□管理器， 期望□到一个程序□行，□□引□□程。 所使用的扇区数可由/boot/boot的大小□定。

boot1 非常□□，因□它再多也只能有 512 字□， 只能□□□存着分区信息的 *bsdlabel*， 及□□□行 boot2。

boot2 □微有点加□，能□理解 FreeBSD 的文件系□以便于□□里面的文件， 能提供□□内核和加□器的□□界面。

因□ loader 有着更□的功能， 提供了一套易于使用的引□配置，boot2 一般都□行 loader， 但以前它的任□是直接□行内核。

例 *11. boot2* 的屏幕□出

```
>> FreeBSD/i386 BOOT
Default: 0:ad(0,a)/boot/loader
boot:
```

如果□要更改已安装的 boot1 和 boot2，□使用命令 bsdlabel(8)。

```
# bsdlabel -B diskslice
```

diskslice 是用于引□的磁□和分区， 比如 ad0s1 代表第一个 IDE 磁□上的第一个分区。

dangerously dedicated

如果□在 bsdlabel(8) 命令中只使用了磁□名，比如 ad0，就会破坏磁□上的所有分区。□当然不是□所希望的，所以在按下 回 □ 之前 一定要□命令□行多次□□。

13.3.3. 第三□段，/boot/loader

加□器 (loader) 是三个□段中的最后□段， 且是放置在文件系□之中的，一般是文件 /boot/loader。

loader 被作□一□友好的配置方式，使用了一□内建且易用的命令集。 □些命令由一个□大的多的解□器支持□建，其本身□有□□得多的命令集。

13.3.3.1. Loader 程序流程

初始□，loader 会探□控制台和磁□，□□是从□□□引□的。 它会根据□些信息□置□量， □□解□器以接受通□脚本或交互方式□来的用□命令。

loader 然后会□取并□行 /boot/loader.rc， 默□地□取 /boot/defaults/loader.conf 以□置可□的默□□量，□取

/boot/loader.conf □□些□量作本地修改。loader.rc 依据□些□量□行□作，加□任何被□□的模□和内核。

最后，默□地，loader 会停留 10 秒等待按□， 若没有□生中断，就□始引□内核。如果被中断，用□会得到一个命令行提示符， 在□里用□得更改□量、卸□所有模□、加□模□、最后引□ 或重新引□。

13.3.3.2. Loader 内建的命令

□些是最常用的 loader 命令.□所有可用命令的解□□参□ loader(8)。

autoboot*seconds*

在□定的□□内如果没有中断□生就引□内核。它□示一个倒数□□， 默□的□□□□是 10 秒。

boot [-options] [kernelname]

立即按指定的□□□□指定名字的内核 (如果有指定的□)。 只有首先□行□ *unload* 命令之后指定的内核名字才会生效， 否□， □□的将是先前已□加□的内核。

boot-conf

基于□量□各□模□□行自□配置 (和引□内核□□生的一□)。 □只□□住要先使用 `unload` 命令， 然后修改一些□量，比如 `kernel`。

help [topic]

□示从文件 /boot/loader.help □取的□助信息。如果□定的主□是 `index`， 那□列出来的是所有可用的主□。

include *filename* ...

通□□定的文件名□理文件。文件被□入，然后被一行一行地解□。 任何□□都会立即中止 include 命令。

load [-t type] *filename*

加□内核、内核模□，或者是□定□型的文件 (通□□定的文件名)。 任何在文件名后面的参数都会被□□文件。

ls [-l] [path]

□示□定路径或者是根目□ (如果路径没有指定) 下面的文件列表。 如果指定了 `-l` □□，文件大小也会□示。

lsdev [-v]

列出所有可以加□模□的□□。 如果指定了`-v` □□，会□示出更多的□□。

lsmod [-v]

□示已被加□的模□。如果指明了 `-v` □□， 会□示更多的□□。

more *filename*

□示指定的文件，□隔 `LINES` 停□一次。

reboot

立即重□系□。

set *variable*

□置 loader 的□境□量。

unload

移除所有已被加□的模□。

13.3.3.3. Loader 示例

□里有一些□□中 loader 用法的示例

- 只是□□的引□默□内核，不同的是□入□用□模式：

```
boot -s
```

- 卸□默□内核和模□，然后加□旧的 (或者其它) 的内核：

```
unload

load kernel.old
```

□可以使用被称□通用内核的 kernel.GENERIC， 或者□以前安装的内核 kernel.old (当□升□或配置了□自己的内核等□候)。

使用以下命令加□常用的模□和□一个内核：

```
unload
set kernel="kernel.old"
boot-conf
```

- 加□内核配置脚本：

```
load -t userconfig_script /boot/kernel.conf
```

13.3.3.4. □□□的 Splash □像

在□□□出□的 splash □像比起原本的□□信息更加可□□。 □个□像将被始□□示在屏幕上直到出□控制台的登□提示或者 X □示管理器提供了登□画面。

在 FreeBSD 系□中有□个基本的□境。 第一个是默□□□的控制台命令行□境。 在系□□□之后， 会在控制台上出□一个登□提示。 第二个□境是 X11 □面□形□境。 在安装了 X11 和一□□形 □面□境， 比如 GNOME， KDE，或者 XFce， X11 □面可以用 `startx` 命令□行。

比起□□基于字符的登□提示，有些用□可能更喜□ X11 □形化的登□界面。 □形化的登□管理器像 Xorg 的 XDM，GNOME 的 gdm， KDE 的 kdm (□有其他 Port Collection 中的) 基本上都提供了一个□形化的登□界面代替控制台上的登□提示符。 在成功登□之后， 它□展□□用□一个□形化的□面。

在命令行□境， splash □像将在□示登□提示符之前□藏所有□□□的□□与任□□□的消息。 在 X11 □境， 用□将会□得一个□□上更加清爽□□体□， □似于某些像 (Microsoft® Windows® 或者非 UNIX® □型的系□) 用□所希望体□到的。

13.3.3.4.1. Splash □像功能

目前的 splash □像的功能□限于支持 256 色的位□ (.bmp) 或者 ZSoft PCX (.pcx) 文件。 此外， splash □像文件的分辨率必□是 320x200 像素或者更少， 才□能在□准 VGA □配器上使用。

要使用尺寸更大的□像， □到最大分辨率 1024x768 像素， □需□□ FreeBSD 的 VESA 支持。 □可以通□在系□□□□加□ VESA 模□完成， 或者在内核配置文件中加入 `VESA` □□并□□ (参□ 配置FreeBSD的内核)。 VESA 支持□予了用□□示覆□整个□示器的□□画面能力。

在□□的□候 splash □像就会被□示在屏幕上， 它可以在任何□候都按任意□□□。

Splash □像同□也会是 X11 之外默□的屏幕保□。 在一段□□的□置后，屏幕便会□□周期性的□□□示 splash □像，从明亮至暗淡， 周而□始。 默□的 splash □像 (屏幕保□) 可由 /etc/rc.conf 中的 `saver=` □□控制。 `saver=` □□有一些内置的屏幕保□可供□□， 完整的列表可以再 splash(4) 手册□中□到。 默□的屏幕保□被称□ "warp"。 □注意在 /etc/rc.conf 中所指定 `saver=` □□□限□用于虚□控制台。 □于 X11 □形化的登□管理器无效。

一些有□□□引□器的信息， 包括□□□□菜□和一个定□倒数提示符都会在□□□□示， 即是□□了 splash □像功能。

splash □像文件□本可以从 http://artwork.freebsdgr.org 下□。 安装了 sysutils/bsd-splash-changer port 之后， □次□□的□候便能从集合中随机□□ splash □像。

13.3.3.4.2. □□ Splash □像功能

Splash □像 (.bmp) 或者 (.pcx) 文件必□放置在 root 分区上， 比如 /boot 目□。

□于默□的□示分辨率 (256 色，320x200 像素或更少) □□ /boot/lodaer.conf， 添加如下的□置：

```
splash_bmp_load="YES"
bitmap_load="YES"
bitmap_name="/boot/splash.bmp"
```

□于更高的分辨率，最大至 1024x768 像素， □□ /boot/lodaer.conf， 添加如下的□置：

```
vesa_load="YES"
splash_bmp_load="YES"
bitmap_load="YES"
bitmap_name="/boot/splash.bmp"
```

以上□些□置假□ /boot/splash.bmp □需要被使用的 splash □像。 当需要使用 PCX 文件的□候， 添加入下列□置， 根据分辨率的高低添加 `vesa_load="YES"`。

```
splash_pcx_load="YES"
bitmap_load="YES"
bitmap_name="/boot/splash.pcx"
```

文件名并不限于以上例子中的 "splash"。 它可以是任何名称，只要是 BMP 或者 PCX □型的文件， 比如 splash_640x400.bmp 或者 blue_wave.pcx.

一些有趣的 loader.conf □□：

`beastie_disable="YES"`

□将□□□示□□□□菜□，　但是倒数□□仍然会出□。　即是在□□菜□□□被禁用的□候，　在倒数□□段□入相□的□□□□仍然有效。

`loader_logo="beastie"`

□将替□□□□□菜□右□默□□示的 "FreeBSD" □彩色的小魔鬼□志，　就像以往的□行版那□。

□参□ splash(4)，　loader.conf(5) 和 vga(4) 手册□□取更多□□信息。

13.4. 内核在引□□的交互

一旦内核被　loader　(一般情况下)　或者　boot2　(越□　loader)　加□，　它将□□引□□志，如果有的□，就会□行必要的□作□整。

13.4.1. 内核引□□志

□里是一些常用的引□□志：

`-a`

在内核初始化□，□□作□根加□的□□。

`-C`

从 CDROM 引□。

`-c`

□行 UserConfig (引□□的内核配置器)

`-s`

引□□入□用□模式

`-v`

在内核引□□程中□示更有的信息

□有更多的引□□志，□□ boot(8) 以□取有□它□的信息。

13.5. Device Hints

在初始化系□□□□，loader(8)　会□取　device.hints(5)　文件。□个文件以□量的形式□存着内核引□信息，　有□被称□ "device hints"。 □□□□程序用"device hints" □□□□行配置。

Device hints 也可以在 第三□段的boot loader 的命令行提示符中指定。□量可以用 `set` 命令添加，`unset` 命令□除，　`show` 命令□看。在文件 /boot/device.hints □置的□量亦可以在□里被覆□。□入 boot loader 中的□量不是永久性的，在下次□□□就会被忘□。

一旦系□引□成功，kenv(1) 命令可以用来清楚所有的□量。

文件 /boot/device.hints 的□法是一行一个□量， 使用"#"作□注□□。 □行是按照如下方式□□的：

```
hint.driver.unit.keyword="value"
```

第三□段 boot loader 的□法是：

```
set hint.driver.unit.keyword=value
```

driver 是□□□□程序名， unit 是□□□□程序□位名， keyword 是 hint □□字。 □□字可以由以下□□□成：

- at：指明□□所□定的□□
- port：指明所使用 I/O 的起始地址。
- irq：指明所使用的中断□求号。
- drq：指明 DMA channel 号。
- maddr：指明□□占用的物理内存地址。
- flags：□□□□置各□□志位。
- disabled：如果□成 1， □□被禁用。

□□□□程序能□接受更多的 hints，推□□参看它□的□机手册。参看 device.hints(5)、kenv(1)、loader.conf(5) 和 loader(8) □机手册以□取更多的信息。

13.6. Init：□程控制及初始化

一旦内核完成引□，它就把控制□交□了用□□程 init(8)，它放置在 /sbin/init， 或者 init_path □量指定的程序路径中。 □个□量是在 loader 里面□置的。

13.6.1. 自□重□□程

自□重□□程会□□系□中可用的文件系□□于健康的状□。 如果不是， 而且使用 fsck(8) 也无法修□□些□□， init(8) 会□入 □用□模式 以便系□管理□直接修正□些□□。

13.6.2. □用□模式

此模式可以通□ 自□重□□程 或者通□□有 -s □□的用□引□或通□在 loader 里□置 boot_single □量等多□方式来□到。

也可以在多用□模式下□□无重□ (-r) □□和停机 (-h) □□的 shutdown(8) 命令来□入□用□模式。

如果系□ 控制台 在文件 /etc/ttys 中被□置□ 不安全(insecure)， 在初始化□用□模式前会出□要求□入 root 密□的命令行提示符。

例 12. 在 /etc/ttys 文件中的不安全控制台

```
# name  getty                           type    status          comments
#
# If console is marked "insecure", then init will ask for the root password # when
going to single-user mode.
console none                            unknown off insecure
```

把控制台□置成 不安全 (insecure) 使只知道 root 密□的人才能□入□用□模式， 因□□□□控制台在物理上是不安全的。因此如果□考□到安全性， □□□ 不安全 (insecure)，而非 安全 (secure)。

13.6.3. 多用□模式

如果 init(8) □□□的文件系□一切正常，又或者用□在□用□模式完成了工作， 系□就会□入多用□模式，□始系□的□源配置。

13.6.3.1. □源配置 (rc)

□源配置分□从文件 /etc/defaults/rc.conf、 /etc/rc.conf 中□取默□配置和□□配置， 然后加□在文件 /etc/fstab 中提及的文件系□、 □□网□服□、□□各□系□守□□程，最后□□本地安装包的□□脚本。

rc(8) □机手册是□于□源配置的很好的参考。

13.7. □机 (shutdown) □程

由命令 shutdown(8) 的□起的□机□程中， init(8) 会□着□行 /etc/rc.shutdown 脚本， □所有□程□送 TERM 信号， 最后□不按□停止的□程□送 KILL 信号。

在支持□源管理的平台上□□ FreeBSD 系□的□源， 只要□□地使用命令 shutdown -p now 即可。 此外，可以用命令 shutdown -r now 来重□ FreeBSD。 要□行 shutdown(8) □必□是 root 用□或 operator □的成□。也可以使用 halt(8) 和 reboot(8) 命令来□□系□， □参看它□的□机手册以□得更多的信息。

□源管理需要支持， □要求内核支持 acpi(4) 或以模□形式加□它。

Chapter 14. 用□和基本的□□管理

14.1. 概述

FreeBSD允□多个用□同□使用□算机. 当然,□些用□中不是很多人同□坐在同一台□算机前. ,而是其他用□可以通□网□来使用同一台□算机以完成他□的工作.要使用系□,□个人都□□有一个□□.

□完□章，□将了解到:

- 在一个FreeBSD系□上不同用□□□之□的区□.
- 如何添加用□□□.
- 如何□除用□□□.
- 如何改□□□□□，如用□的全名，或首□的shell.
- 如何在□个□□基□上□置限制，来控制像内存，CPU□□□□的□源.
- 如何使用□来使□□管理更容易.

在□□□章之前，□□当了解:

- 了解UNIX®和FreeBSD的基□知□ (UNIX 基□).

14.2. 介□

所有□□系□的用□都是通□□□完成的，所以用□和□□管理是FreeBSD系□不可或缺的重要部分.

□个FreeBSD系□的□□都有一些和它相□□的信息去□□它.

用□名

用□名在`login:` 提示符的后面□入。 用□名□于一台□算机来□是唯一的； □不可以使用□个相同的用□名来登□。 有很多用来□建正□用□名的□□， 具体□参考 passwd(5)； □使用的用□名通常需要8个或更少的小写字母。

口令

□个□□都有一个口令与它□□。 口令可以是空的， □□不需要口令就可以□□系□。 □通常不是一个好主意； □个□□都□□有口令。

用□ ID (UID)

UID是系□用来□□用□的数字，□□上它的□□是0到65536之□，用以唯一地□□用□。 FreeBSD在内部使用UID来□□用□ - 在工作以前。 任何允□□指定一个用□名的 FreeBSD 命令都会把它□□成UID。 □意味着□可以用不同的用□名使用多个□□， 但它□的UID是一□的。 FreeBSD 会把□些□□□定是同一个用□。

□ID (GID)

GID是用来□□用□所在的□的， □□上□□在0到65536之□的数字。 □是一□基于用□GID而不是它□的UID的用来控制用□□□□源的机制。 □可以□少一些配置文件的大小。 一个用□也可以属于多个□。

登□□

登□□是□□机制的□展,当把系□分配□不同用□□,它提供了□外的□活性.

口令的定期更改

默□情况下， FreeBSD 并不□制用□去改□他□的口令。 □可以以用□□□位□制要求一些或所有的用□定期改□他□的口令。

□□的到期□□

默□情况下 FreeBSD 不会自□完成□□□期操作。 如果□正在□建□□， □□□知道一个□□的有效使用期限。例如， 在学校里□会□□个学生建立一个□□， □可以指定它□何□□期。 □□□期后， □然□□的目□和文件仍然存在， 但□□已□不能□□使用了。

用□的全名

用□名可以唯一地□□FreeBSD的□□， 但它不会反映用□的全名。 □些信息可能与□□是相□的。

主目□

主目□是用□用来□□的目□的完全路径。 一个通常的□□是把所有用□的主目□都放在 /home/username 下，或者 /usr/home/username 下。 用□将把他□的个人文件放在自己的主目□下， 他□可以在那里□建任何目□.

用□ **shell**

Shell提供了用□用来操作系□的默□□境。 有很多不同的shell， 有□□的用□会根据他□的□□来□□自己喜好的shell。

有三□□型的□□: 超□用□， 系□用□， 以及 普通用□。 超□用□□□通常叫做 `root`， 可以没有限制地管理系□。 系□用□□行服□。 最后， 普通用□□那些登□系□以及□□□件的人使用。

14.3. 超□用□□□

超□用□□□， 通常叫做 `root`， 可以重新配置和管理系□， 在收□□件， 系□□□或□程□□的日常工作中，尽量不要使用root□限。

□是因□不象普通用□□□， 超□用□能□无限制地操作系□， 超□用□□□的□用可能会引起无法想象的□□。普通的用□□□不会由于出□而破坏系□， 所以要尽可能的使用普通□□， 除非□需要□外的特□。

在使用超□用□命令□要再三□□， 因□一个□外的空格或缺少某个字符的命令都可能会引起数据□失。

所以， 在□□完□章后□第一件要做的事就是， 在平□使用的□候， □建一个没有特□的用□□□。 无□□使用的是□用□□是多用□系□□□的申□都是相同的。 在□章的后面， 我□将□□如何□建一个□外的□□和如何在普通用□和超□用□之□□行切□。

14.4. 系□□□

系□用□是那些要使用□如DNS、 □件， web等服□的用□。 使用□□的原因就是安全； 如果所有的用□都由超□用□来□行， 那它□就可以不受□束地做任何事情。

典型的系□□□包括 `daemon`、 `operator`、 `bind` (供 域名服□ 使用)、 `news`， 以及 `www`。

`nobody` 是普通的没有特□的系□用□。 然而， 大多数与用□□系很密切的服□是使用 `nobody`的， □的□点非常重要， □□可能使用□□的非常有特□。

14.5. 用□□□

用□□□是□真□的用□□□系□的主要方式， □些□□把用□和□境隔□， 能阻止用□□坏系□和其他用□， 在不影□其他用□的情况之下定制自己的□境。

任何人□□□的系□必□要有他□自己唯一的□□。 □可以□□□到□做了什□事， 并且阻止人□破坏其他用□的□置和□□其他人的□件等等。

□个用□能□□置他□自己的□境， 以利于他□通□改□shell， □□器， □□□定和□言等□□并且更好的使用□个系□。

14.6. 修改□□

在UNIX® 的□理用□□□的□境中有很多不同的命令可用. 最普通的命令如下， 接下来是□□使用它□的例子。

命令	摘要
adduser(8)	在命令行添加新用□.
rmuser(8)	在命令行□除用□.
chpass(1)	一个□活的用于修改用□数据□信息的工具.
passwd(1)	一个用于修改用□口令的□□的命令行工具.
pw(8)	一个□大□活修改用□□□的工具.

14.6.1. 添加用□

adduser(8) 是一个□□的添加新用□的命令. 它□用□□建 passwd 和 group 文件。 它也□新用□□建一个主目□，之后， 它会□制一□默□的配置文件 ("dotfiles") 从 /usr/shared/skel □个目□， 然后□新用□□送一封□□迎信息的□件。

例 13. 在 FreeBSD 中添加一个新用

```
# adduser
Username: jru
Full name: J. Random User Uid (Leave empty for default):
Login group [jru]:
Login group is jru. Invite jru into other groups? []: wheel
Login class [default]:
Shell (sh csh tcsh zsh nologin) [sh]: zsh
Home directory [/home/jru]:
Home directory permissions (Leave empty for default):
Use password-based authentication? [yes]:
Use an empty password? (yes/no) [no]:
Use a random password? (yes/no) [no]:
Enter password:
Enter password again:
Lock out the account after creation? [no]:
Username   : jru
Password   : ****
Full Name  : J. Random User
Uid        : 1001
Class      :
Groups     : jru wheel
Home       : /home/jru
Shell      : /usr/local/bin/zsh
Locked     : no
OK? (yes/no): yes
adduser: INFO: Successfully added (jru) to the user database.
Add another user? (yes/no): no Goodbye!
#
```

入的口令并不会回到屏幕上， 此外系也不会示星号。 必保没有口令。

14.6.2. 除用

可以使用rmuser(8) 从系中完全除一个用. rmuser(8) 行如下:

1. □除用□的 crontab(1) □□ (如果有的□).
2. □除属于用□的at(1) 工作.
3. □掉属于用□的所有□程.
4. □除本地口令文件中的用□.
5. □除用□的主目□ (如果他有自己的主目□).
6. □除来自 /var/mail属于用□的□件.
7. □除所有□如 /tmp的□□文件存□区中的文件.
8. 最后, □除 /etc/group中所有属于□的□用□名.

如果一个□□成空，而□名和用□名一□,□将被□除. adduser(8)命令建立□个用□唯一的□.

rmuser(8) 不能用来□除超□用□的□□, 因□那□做是□系□□大的破坏.

默□情况下, 使用交互模式, □□能□□□清楚的知道□在做什□.

例 *14.* □除用□ 交互模式下的□□□除

```
# rmuser jru Matching password entry:
jru:*:1001:1001::0:0:J. Random User:/home/jru:/usr/local/bin/zsh Is this the entry
you wish to remove? y Remove user's home directory (/home/jru)? y Updating
password file, updating databases, done.
Updating group file: trusted (removing group jru -- personal group is empty) done.
Removing user's incoming mail file /var/mail/jru: done.
Removing files belonging to jru from /tmp: done.
Removing files belonging to jru from /var/tmp: done.
Removing files belonging to jru from /var/tmp/vi.recover: done.
#
```

14.6.3. chpass

chpass(1) 可以改□用□的口令, shells, 和包括个人信息在内的数据□信息.

只有系□管理□， 即超□用□， 才可以用 chpass(1) 改□其他用□口令和信息。

除了可□□的用□名， 不需要任何□□， chpass(1) 将□示一个包含用□信息的□□器. 可以□□改□用□在数据□中的信息.

如果□不是超□用□的□， 在退出□□状□之后， 系□会□□□口令。

例 *15.* 以超□用□交互□行 chpass 命令

```
#Changing user database information for jru.
Login: jru
Password: *
Uid [#]: 1001
Gid [# or name]: 1001
Change [month day year]:
Expire [month day year]:
Class:
Home directory: /home/jru
Shell: /usr/local/bin/zsh
Full Name: J. Random User
Office Location:
Office Phone:
Home Phone:
Other information:
```

普通用□只能改□他□自己很少的一部分信息.

例 *16.* 以普通用□交互□行 chpass 命令

```
#Changing user database information for jru.
Shell: /usr/local/bin/zsh
Full Name: J. Random User
Office Location:
Office Phone:
Home Phone:
Other information:
```

chfn(1) 和 chsh(1) 只是到 chpass(1) 的符号□接， □似地， ypchpass(1), ypchfn(1) 以及 ypchsh(1) 也是□□。 NIS 是自□支持的， 不一定要在命令前指定 yp。 如果□□□有点不太明白， 不必担心， NIS 将在 介□。

14.6.4. passwd命令

passwd(1) 是改□□自己作□一个普通用□口令或者作□超□用□口令常用的方法.

用□改□口令前必□□入原来的口令, 防止用□□□□端□非授□的用□□入改□合法用□的口令。

例 17. 改□□的口令

```
% passwd Changing local password for jru.
Old password:
New password:
Retype new password:
passwd: updating the database...
passwd: done
```

例 18. 改□其他用□的口令同超□用□的一□

```
# passwd jru Changing local password for jru.
New password:
Retype new password:
passwd: updating the database...
passwd: done
```

就象 chpass(1)一□, yppasswd(1) 只是一个到 passwd(1)的□接, 所以NIS用任何一个命令都可以正常工作.

14.6.5. pw命令

pw(8) 是一个用来□建、□除、修改、□示用□和□的命令行工具。 它□有系□用□和□文件□□器的功能。 pw(8)有一个非常□大的命令行□□□置, 但新用□可能会□得它比□里□的其它命令要□□很多。

14.7. 限制用□使用系□□源

如果□有一些用□, 并想要□他□所使用的系□□源加以限制, FreeBSD 提供了一些系□管理□限制用□□□系□□源的方法。 □些限制通常被分□□□: 磁□配□, 以及其它□源限制。

磁□配□限制用□□磁□的使用, 而且它□提供一□快速□□用□使用磁□数量而不需要□刻□算的方法。 配□将在文件系□配□□□.

其它□源限制包括CPU、 内存以及用□可能会使用的其它□源。 □些是通□□登□□行分□完成的, 下面将做□□。

登□的□由 /etc/login.conf 文件定□。 比□精□的描述超出了本章的□□, 但 login.conf(5) □机手册会有比□□□的描述。 可以□□个用□都分配到一个登□□ (默□是 defalut), □个登□□都有一套和它相□□的功能。 登□功能是 名字=□ □□的一□□, 其中名字 是一个□所周知的□□符, □ 是一个根据名字□□□理得到的任意字符串。□置登□□和功能相当□□, 在 login.conf(5) □机手册会有比□□□的描述。

系□并不直接□取 /etc/login.conf 中的配置， 而是采用数据□文件 /etc/login.conf.db 以提供更快的□□能力。 要从 /etc/login.conf 文件生成 /etc/login.conf.db， □使用下面的命令：

```
# cap_mkdb /etc/login.conf
```

□源限制与普通登□限制是有区□的。 首先， □于□□限制， 有□限制 (比□常□) 和硬限制之分。 一个□限制可能被用□□整□， 但不会超□硬限制。 越往后可能越低， 但不会升高。 其次， □大多数□源限制会分配特定用□的□个□程， 而不是□用□的全部□程。 注意， □些区□是□源限制的特殊操作所□定的， 不是登□功能框架的完成 (也就是□, 他□□□上 不是一个登□功能的特例)。

不再□□了， 下面是□大多数□源限制的例子 (□可以在 login.conf(5) □到其它与登□功能相□的内容)。

coredumpsize

很明□， 由程序□生的核心文件大小的限制在磁□使用上是属于其它限制的 (例如， 文件大小， 磁□配□)。 不□， 由于用□自己无法□生核心文件， 而且通常并不□除它□， □置□个可以尽量避免由于一个大型□用程序的崩□所造成的大量磁□空□的浪□。 (例如, emacs) 崩□。

cputime

□是一个用□□程所能消耗的最□ CPU □□。 □反限制的□程， 将被内核□掉。

□是一个有□CPU消耗的□□ 限制， 不是top(1) 和 ps(1) 命令□屏幕上□示的 CPU消耗的百分比。 在写此□明□， 后者的限制是不太可能和没有价□的： □□器 - □□一个可能是合法的工作 - 可以在某一□刻□易的用掉 100% 的 CPU。

filesize

□是用□可以□理一个文件的最大□。 不象 磁□配□， □个限制是□□个文件□制□行的， 不是用□自己的所有文件。

maxproc

□是一个用□可以□行的最大□程数。 □包括前台和后台□程。 很明□， □不可能比系□指定 kern.maxproc sysctl(8) 的限制要大。 同□也要注意， □置的□小会妨碍用□的□理能力： 可能需要多次登□或□行多个管道。 一些任□， 例如□□一些大的程序， 也可能会□生很多□程 (例如， make(1)， cc(1) 以及其它一些□□理程序)。

memorylocked

□是一个□程允□□到主存中的最大内存容量 (参□ mlock(2))。 大型程序， 例如像 amd(8) 在遇到□□□， 它□得到的巨大交□量无法□□□系□□行□理。

memoryuse

□是在□定□□内一个□程可能消耗的最大内存数量。 它包括核心内存和交□内存。 在限制内存消耗方面， □不是一个完全的限制，但它是一个好的□始。

openfiles

□是一个□程可以打□的最大文件数。 在FreeBSD中， 文件可以被表□□套接字和IPC通道； 注意不要把□个数□置的太小。 系□□的限制是由 kern.maxfiles 定□的， □情参□ sysctl(8)。

sbsize

□是网□内存数量的限制， □主要是□□通□□建□多套接字的老式 DoS 攻□的， 但也可以用来限制网□通信。

stacksize

□是一个□程堆□可能□到的最大□。 它不能□独的限制一个程序可能使用的内存数量； 所以，需要与其它的限制手段配合使用。

在□置□源限制□， 有一些其他的事需要注意。 下面是一些通常的技巧、 建□和注意事□。

- 系□□□的□程/etc/rc会被指派□ 守□程序 的登□□.
- □然 /etc/login.conf 文件是一个□□大多数限制做合理配置的□源文件， 但只有□也就是系□管理□，才知道什□最□合□的系□。 □置的太高可能会因□□于□放而□致系□被□用， 而□置□低， □可能降低效率。
- 使用 X Window 的用□可能要比其他用□使用更多的□源。 因□X11本身就使用很多□源， 而且它鼓励用□同□□行更多的程序。
- □必注意， □多限制措施是□□□个□程来□施的， 它□并不限制某一用□所能用到的□量。 例如， 将 openfiles □置□ 50 表示以□用□身□□行的□程最多只能打□ 50 个文件。 因而， 用□□□可以打□的文件□数就□□是 maxproc 和 openfiles □的乘□。 □内存用量的限□与此□似。

有□□源限制,登□□的更深入信息可以□看相□□机手册: cap.mkdb(1), getrlimit(2), login.conf(5).

14.8. □

□□□的□就是一个用□列表. □通□□名和GID (□ ID) 来□□。 在 FreeBSD (以及□大多数其他 UNIX® 系□) 中，内核用以决定一个□程是能□完成一□□作的□个因素是它所属的用□ ID 和□ ID。 与用□ ID 不同， □个□程都有一个和它相□□的□的列表。 □可能听□□用□或□程的 "□ ID"； 大多数情况下， □表示列表中的第一个□。

与□ID□□的□名在/etc/group中。 □是一个由冒号来界定的文本文件。 第一部分是□名，第二部分是加密后的口令， 第三部分是□ID， 第四部分是以逗号相隔的成□列表。 它可以用手工方式□行□□(当然， 如果□能保□不出□法□□的□！)。 □于更完整的□法描述， 参□ group(5) □机手册.

如果不想手工□□ /etc/group， 也可以使用 pw(8) 添加和□□□。 例如， 要添加一个叫 teamtwo 的□，□定它存在：

例 *19.* 使用*pw(8)*添加一个□

```
# pw groupadd teamtwo
# pw groupshow teamtwo
teamtwo:*:1100:
```

上面的数字 1100 是□ teamtwo 的□ ID。 目前， teamtwo □没有成□， 因此也就没有多大用□。 接下来， 把 jru 加入到 teamtwo □。

例 20. 使用 *pw(8)* □置□的成□列表

```
# pw groupmod teamtwo -M jru
# pw groupshow teamtwo
teamtwo:*:1100:jru
```

`-M` 所需的参数是一个用逗号分隔的□中将要成□成□的用□列表。 前面我□已□知道， □令文件中， □个用□已□指定了一个所属□。 之后用□被自□地添加到□列表里； 当我□使用 `groupshow` 命令□ pw(8) 用□列表不被□示出来。 但当通□ id(1) 或者□似工具□看□， 就会看到用□列表。 □言之， pw(8) 命令只能□取 /etc/group 文件； 它从不□□从 /etc/passwd 文件□取更多信息。

例 21. 使用 *pw(8)* □□添加新的成□

```
# pw groupmod teamtwo -m db
# pw groupshow teamtwo
teamtwo:*:1100:jru,db
```

`-m` □□的参数是一个由逗号分隔的即将被添加□□的用□列表。 与先前那个例子的不同之□在于， □个列表中的用□将被添加□□而非取代□中的□有用□。

例 22. 使用*id(1)*来决定□成□

```
% id jru
uid=1001(jru) gid=1001(jru) groups=1001(jru), 1100(teamtwo)
```

正如□所看到的， `jru` 是□ `jru` 和□ `teamtwo`的成□.

有□pw(8)的更多信息， □参看其它□机手册。 更多的□于 /etc/group 文件格式的信息， □参考 group(5) □机手册。

Chapter 15. 安全

15.1. 概述

□一章将□系□安全的基本概念□行介□， 除此之外， □将介□一些好的□□， 以及 FreeBSD 下的一些更深入的□□。 □章的□多内容□于一般的系□和 Internet 安全也□用。 如今， Internet 已□不再像以前那□是一个人人都□意与□作好□居的 "友善" 的地方。 □系□更加安全， 将保□□的数据、 智力□□、 □□， 以及其他很多□西不至于被入侵者或心存□意的人所窃取。

FreeBSD 提供了一系列工具和机制来保□□的系□和网□的完整及安全。

□完□章，□将了解：

- 基本的 FreeBSD 系□安全概念。
- FreeBSD 中□多可用的密□学□施，例如 DES 和 MD5。
- 如何□置一次性口令□□机制。
- 如何配置 TCP Wrappers 以便与 inetd 配合使用。
- 如何在 FreeBSD 上□置 Kerberos5。
- 如何配置 IPsec 并在 FreeBSD/Windows® 机器之□建□ VPN。
- 如何配置并使用 OpenSSH，以及 FreeBSD 的 SSH □行方式。
- 系□ ACL 的概念，以及如何使用它□。
- 如何使用 Portaudit 工具来□核从 Ports Collection 安装的第三方□件包的安全性。
- 如何从 FreeBSD 的安全公告中□得有用信息并采取相□措施。
- □于□程□□功能的感性□□， 并了解如何在 FreeBSD 中□用它。

在□始□□□章之前，□需要：

- 理解基本的 FreeBSD 和 Internet 概念。

其他安全方面的□□， □□穿本□的始□。 例如， □制性□□控制 (MAC) 在 □制□□控制 中□行了介□， 而 Internet 防火□□在 防火□ 中□行了□□。

15.2. 介□

安全是系□管理□自始至□的基本要求。 由于所有的 BSD UNIX® 多用□系□都提供了与生俱来的安全性，因此建立和□□□外的安全机制， □保用□的 "□□" 可能也就是最需要系□管理□考□的□巨的工作了。机器的安全性取决于□□置的安全□施， 而□多安全方面的考□， □会与人□使用□算机□的便利性相矛盾。一般来□， UNIX® 系□能□□任数目□多□程并□地□理各□任□， □其中的□多□程是以服□身□□行的 - □意味着，外部□体能□与它□互□并□生会□交互。 如今的□面系□， 已□能□□到□多昔日的小型机甚至主机的性能， 而随着□些□算机的□网和在更大□□内完成互□， 安全也成□了一个日益□峻的□□。

系□的安全也□能□□付各□形式的攻□， □也包括那些使系□崩□， 或阻止其正常□□， 并不□限于□□窃取 `root` □号 ("破□ root") 的攻□形式。 安全□□大体可分□以下几□：

1. 拒□服□攻□。
2. 窃取其他用□的□□。
3. 通□可□□服□窃取root□□。
4. 通□用□□□窃取root□□。
5. 建立后□。

拒□式服□攻□是侵占机器所需□源的一□行□。 通常， DoS 攻□采用暴力(brute-force)手段通□□倒性的流量来破坏服□器和网□□， 以使机器崩□或无法使用。 某些 DoS 攻□□利用在网□□中的□□， □用一个□□的信息包就可以□机器崩□， □□情况通常只能通□□内核打□丁来修□。 在一些不利的条件下， □服□器的攻□能□被修□， 只要□当地修改一下系□的□□来限制系□□服□器的□荷。 □□的网□攻□是很□□付的。例如，一个欺□性信息包的攻□， 无法阻止入侵者切断□的系□与Internet的□接。 它不会使□的机器死掉，但它会把Internet□接占□。

窃取用□□□要比D.o.S.攻□更加普遍。 □多系□管理□仍然在他□的服□器上□行着基本的 telnetd，rlogind， rshd 和 ftpd 服□。 □些服□在默□情况下不会以加密□接来操作。 □果是如果□的系□有中等□模大小的用□群， 在通□□程登□的方式登□到□系□的用□中， 一些人的口令会被人窃取。 仔□的系□管理□会从那些成功登□系□的□程□□日志中□□可疑的源地址。

通常必□假定，如果一个入侵者已□□□到了一个用□的□□， 那□它就可能使自己成□ root。 然而， 事□是在一个安全和□□做得很好的系□中， □□用□的□□不一定会□入侵者成□ root。 □个差□是很重要的，因□没有成□ root □入侵者通常是无法□藏它的□迹的， 而且， 如果走□的□， 除了□用□的文件乱掉和系□崩□之外， 它不能做什□□的事情。 窃取用□□□是很普遍的事情， 因□用□往往不会□系□管理□的警告采取措施。

系□管理□必□牢牢□住，可能有□多潜在的方法会使他□机器上的 root 用□受到威□。入侵者可能知道 root 的口令，而如果在以 root □限□行的服□器上□到一个缺陷 (bug)， 就可以通□网□□接到那台服□器上□到目的；□外， 一旦入侵者已□侵入了一个用□的□□， 可以在自己的机器上□行一个 suid-root 程序来□□服□器的漏洞， 从而□他侵入到服□器并□取 root。 攻□者□到了入侵一台机器上 root 的途径之后， 他□就不再需要安装后□了。□多 root 漏洞被□□并修正之后， 入侵者会想尽□法去□除日志来消除自己的□□痕迹，所以他□会安装后□。 后□能□入侵者提供一个□□的方法来重新□取□□系□的 root □限， 但它也会□□明的系□管理□一个□□入侵的□便方法。 □入侵者无法安装后□事□上□□的系□安全是有害的， 因□□□并不会修□那些侵入系□的入侵者所□□的新漏洞。

安全的管理方法□当使用像 "洋葱皮" 一□多□次的方法来□□， □些措施可以按下面的方式□行分□：

1. □保 root 和□□人□□□的安全。
2. □保 root - 以root用□□限□行的服□器和suid/sgid可□行程序的安全。
3. □保用□□□的安全。
4. □保口令文件的安全。
5. □保内核中核心□件、直接□□□□和文件系□的安全。
6. 快速□□系□中□生的不□当的□化。
7. 做个偏□狂。

□一章的下一□将比□深入地□述上面提到的□一个条目。

15.3. □保 FreeBSD 的安全

命令与□□

在□□文□中，我□使用 粗体 来表示□用程序， 并使用 □倍距 字体来表示命令。 □□的排版区分能□有效地区分□似 ssh □□的概念， 因□它既可以表示命令，又可以表示□□。

接下来的几□中， 将介□在□一章中 前一□ 中所介□的那些加□ FreeBSD 系□安全性的手段。

15.3.1. □保 root 和□□人□□□的安全

首先，如果□没有□保 root □□的安全， 就没必要先□神□保用□□□的安全了。□大多数系□都会指派一个口令□ root □□。 我□的第一个假定是，口令 □是 不安全的。 □并不意味着□要把口令□掉。 口令通常□□□机器的控制台来□是必□的。 也就是□， □□□避免允□在控制台以外的地方使用口令， 甚至包括使用 su(1) 命令的情形。 例如，□信□的 pty □端在 /etc/ttys 文件中被指定□ insecure (不安全)，□将使直接通□ telnet 或 rlogin 登□ root 会不被接受。 如果使用如 sshd □□的其他登□服□， 也要□□直接登□ root 是□□的。□可以通□□□ /etc/ssh/sshd_config 文件来做到□一点，□信 PermitRootLogin 被□置成 no。 考□到□一□□□方法 - 如 FTP□□的服□， 以免因□它□而□致安全性的□失。 直接登□ root 只有通□系□控制台才被允□。

当然， 作□一个系□管理□， □□当□得 root身□， 因此， 我□□了一些后□来允□自己□入。 但□些后□只有在□□了□外的口令□□之后才能使用。 一□□ root 可□□的方法是□加□当的用□□□到 wheel □ (在 /etc/group 中)。wheel □中的用□成□可以使用 su 命令来成□ root。 □□不□□通□在口令□中□行□置来□予□□人□天然的 wheel □成□身□。 □□人□□被放置在 staff □中，然后通□ /etc/group 文件加入到 wheel □。事□上，只有那些需要以 root 身□□行操作的用□才需要放□ wheel □中。 当然，也可以通□ 某□其它的□□手段，例如 Kerberos，可以通□ root □□中的 .k5login 文件来允□□行 ksu(1) 成□ root ， 而不必把它□放□ wheel □。 □可能是一□更好的解决方案，因□ wheel 机制仍然可能□致入侵者□得 root ，如果他拿到了口令文件，并能□□入□□的□□。 尽管有 wheel 比什□都没有要□一些， 但它并不是一□□□安全的□法。

可以使用 pw(8) 命令来完全禁止某一个□号：

```
# pw lock staff
```

□将阻止用□使用任何方法登□，包括 ssh(1)。

□一个阻止某个□□□□的方法是使用一个 “*” 字符替□掉加密后的口令。 □将不会与任何加密后的口令匹配，从而阻止了用□的□□。 □例□明：

```
foobar:R9DT/Fa1/LV9U:1000:1000::0:0:Foo Bar:/home/foobar:/usr/local/bin/tcsh
```

□被改□：

```
foobar:*:1000:1000::0:0:Foo Bar:/home/foobar:/usr/local/bin/tcsh
```

□会阻止用□ foobar 使用□□的方式登□。 但是□于使用了 Kerberos 或者配置了 ssh(1) 公□/密□□的情况下，用□依然可以□□。

□些安全机制同□假定， 从□格受限的机器向限制更□松的机器上登□。 例如， 如果□的服□器□行了所有的服□，那□，工作站□□什□都不□行。 □了□工作站尽可能地安全，□□避免□行任何没有必要的服□， 甚至不□行任何服□。 □外， 也□□考□使用□口令保□功能的屏幕保□程序。 毋庸置疑， 如果攻□者能□物理地接触□的工作站， 那□他就有能力破坏任何安全□施，□□□是我□需要考□的一个□□，但同□地， 真正能□物理接触□的工作站或服□器并□施攻□的人在□□生活中并不常□， □大多数攻□来自于网□， 而攻□者往往无法物理地接触服□器或工作站。

使用□似 Kerberos □□的工具，也□我□提供了使用一个工具来禁用某个用□， 或修改他□的口令，并在所有机器上立即生效的方法。 如果□工的□号被窃取， 能□在所有的其他机器上生效的口令□更将很有意□。如果口令分散地保存在多个机器上， 一次修改 N 台机器上的口令很可能是一件痛苦的事情。 此外，Kerberos □能□提供更多的限制，除了 Kerberos 令牌有很好的□期机制之外， 它□能□□制用□在某个特定的期限内修改口令(比如□，□月一次).

15.3.2. □保以root用□□限□行的服□器和suid/sgid可□行程序的安全

□慎的管理□只□行他□需要的服□， 不多， 不少。 要当心第三方的服□程序很可能有更多的□□。 例如，□行旧版的 imapd 或 popper 无□于将 root 令牌拱手送□全世界的攻□者。 永□不要□行那些□没有仔□□□□的服□程序， □外也要知道， □多服□程序并不需要以 root 的身□□行。 例如， ntalk、 comsat， 以及 finger□些服□， 都能□以一□被称作 沙盒 的特殊用□的身□□行。 除非□已□解决掉了□多麻□的□□， 否□沙盒就不是完美的， 但洋葱式安全□□仍然成立： 如果有人□法攻破了在沙盒中□行的程序， 那□在做更多坏事之前， 他□□必□想□法攻破沙盒本身的限制。 攻□者需要攻破的□次越多， 他□成功的可能性就越小。 □去， 破解 root 的漏洞几乎在所有以 root 身□□行的服□上都□□□，包括那些基本的系□服□。 如果□的机器只打算向外界提供 sshd 登□， 而用□不会使用 telnetd 或 rshd 甚至 rlogind 登□， 就□□毫不□豫地□□它□！

FreeBSD □在默□在沙盒中□行 ntalkd, comsat, 以及 finger。此外， named(8) 也可以□□□行。/etc/defaults/rc.conf 中包括了如何如此□行 named 的方法，只是□些内容被注□掉了。 如何升□或安装系□将决定□些沙盒所使用的特殊用□是否被自□安装。 □慎的系□管理□将根据需要研究并□□沙盒。

此外，□有一些服□通常并不在沙盒中□行： sendmail, popper, imapd, ftpd, 以及一些其他的服□。当然，它□有一些替代品，但安装那些服□可能需要做更多□外的工作。 可能必□以 root 身□□行□些程序，并通□其他机制来□□入侵。

系□中□一个比□大的 root 漏洞 是安装在其中的 suid-root 和 sgid 的可□行文件。 □大多数□□程序， 例如 rlogin 会放在 /bin、 /sbin、 /usr/bin， 或 /usr/sbin 目□中。 尽管并没有 100% 的安全保□，但系□默□的 suid 和 sgid 可□行文件通常是相□安全的。 当然，偶□也会□□一些存在于□些可□行文件中的 root 漏洞。1998年，Xlib 中□□了一□ root 漏洞，□使得 xterm (通常是做了suid的) □得可以入侵。 做得安全些， □比出□□□再后悔要□。 因此，□慎的管理□通常会限制 suid 可□行文件， 并保□只有□工□号能□□行它□，或只□放□特定的用□□，甚至□底干掉 (chmod 000) 任何 suid 可□行文件， 以至于没有人能□□行它□。没有□示□□的服□器通常不会需要 xterm 可□行文件。 sgid 可□行文件通常同□地危□。 一旦入侵者攻克了sgid-kmem，那□他就能□□取 /dev/kmem 并□而□取□□加密的口令文件， 从而窃取任何包含口令的□号。□外，攻破了 kmem 的入侵者能□□□通□ pty □送的按□序列，即使用□使用的是安全的登□方式。 攻破了 tty □的用□□能□向几乎所有用□的 tty 写入数据。如果用□正在□行一个□端程序，或包含了□□模□功能的□端□真程序， 那□，入侵者能□以那个用□的身□□行任何命令。

15.3.3. □保用□□□的安全

用□□号的安全通常是最□保□的。□然□可以□□的□工□置□苛的登□限制， 并用 "星号" 替□掉他□的口令， 但□可能无法□普通的用□□□做。 如果有足□的决策□， 那□在保□用□□号安全的斗争中或□会□于□□， 但如果不是

□□， □能做的只是警□地□控□些□号的□□。 □用□使用 ssh 或 Kerberos 可能会有更多的□□， 因□需要更多的管理和技□支持， 尽管如此， 与使用加密的口令文件相比， □仍不失□一个好□法。

15.3.4. □保口令文件的安全

能□□保起作用的唯一一□方法， 是将口令文件中尽可能多的口令用星号代替， 并通□ ssh 或 Kerberos 来使用□些□号。 即使只有 root 用□能□□取加密□的口令文件 (/etc/spwd.db)， 入侵者仍然可能□法□到它的内容， 即使他□□□无法写入□个文件。

□的安全脚本□□□常□□并□告口令文件的□□ (参□后面的 □□文件完整性 一□)。

15.3.5. □保内核中内核□□、直接□□□□和文件系□的安全

如果攻□者已□拿到了 root 那□他就有能力作任何事情， 当然， 有一些事情是他□比□喜□干的。 例如， □大多数□代的内核都包括一个内建的听包□□。 在 FreeBSD 中，□个□□被称作 bpf 。攻□者通常会□□在攻克的系□上□行它。 如果□不需要 bpf □□提供的功能，那□，就不要把它□入内核。

但是， 即使□□□了 bpf □□， 仍需要□注 /dev/mem 和 /dev/kmem。 就事□事地□， 入侵者仍然能通□直接□□的方式写入磁□□□。 □外， □有一个称作模□加□器的内核机制， kldload(8)。 有□取心的入侵者， 可以□由□一机制， 在正在□行的内核中通□ KLD 模□来安装自己的 bpf， 或其它听包□□。 □了避免□些□□， □必□将内核的安全□□提高到至少 1。

内核的安全□□可以通□多□方式来□置。 最□□的□置正在□行的内核安全□的方法， 是使用 sysctl 来□置内核□量 kern.securelevel：

```
# sysctl kern.securelevel=1
```

默□情况下， FreeBSD 内核□□□的安全□□是 -1。 除非管理□或 init(8) 由于□□脚本加以改□， 安全□□会□□保持□ -1。 在系□□□□程中， 可以在 /etc/rc.conf 文件中， 将□量 kern_securelevel_enable □量□置□ YES 并将 kern_securelevel □量□置□希望的安全□□来提高它。

默□情况下， 在□□脚本□行完之后， FreeBSD 的安全□□□置是 -1。 □称作 "不安全模式"， 因□文件的不可修改□□ (immutable flag) 可以改□□□， 而且全部□□可以直接□行□写， 等等。

一旦将安全□□□置□ 1 或更高， □只允□追加 (append-only) 和不可修改□□会被□行， 而且不可以□□。 直接□□裸□□□会被拒□。 更高的安全□□会施加□一□的□□限制。 □于安全□□的完整介□， □参□□机手册 security(7) (□于 FreeBSD 7.0 之前的版本， □是□机手册 init(8))。

将安全□□□整到 1 或更高可能会□致 X11 (□□ /dev/io 会被阻止)， 或从源代□□□ FreeBSD (□一□程中的 installworld 部分需要□□取消一些文件上的只允□追加和不可修改□□) 出□一些□□， 并□致一些其他小□□。 有些□候， 例如 X11 的情况， 可以通□在引□□程中□早的□段□□ xdm(1) 来□□， 因□□□安全□□□很低。 □似□□的方法， □于某些安全□□或限制有可能不可用。 提前做好□□可能会是个好主意。 理解不同的安全□□所施加的限制非常重要， 因□一些限制可能□系□□得很□使用。 □外， 了解它□也有助于理性地配置默□□定。

如果内核的安全□□□□ 1 或更高， 在重要的□□程序、 目□和脚本文件上□置 schg □□ (也就是在系□□□到□置安全□□之前□行的程序和它□的配置) 就有意□了。 然而， □□做也可能有些□火， 而由于系□□行于□

高的安全□□， 升□系□也会□得困□的多。 作□妥□， 可以□系□以□高的安全□□□行， 但并不将所有的□□文件都配置 schg □□。 □一□方法是将 / 和 /usr 以只□模式挂□。 □注意， □分□苛的安全配置很可能限制□□□入侵的能力。

15.3.6. □□文件完整性: 可□行文件，配置文件和其他文件

当□施□格的限制□，往往会在使用的方便性上付出代价。例如，使用 chflags 来把 schg □□ □用到 / 和 /usr 中的□大多数文件上可能会起到反作用， 因□尽管它能□保□那些文件， 但同□也使入侵□□无法□行。□次化安全的最后一□可能也是最重要的 - □□。 如果无法□□出潜在的入侵行□， 那□安全的其他部分可能相□来□意□可能就不那□大了 (或者，更糟□的事情是， 那些措施会□□安全的假象)。 □次化安全最重要的功能是□□入侵者， 而不是□底不□他□入侵， □□才可能当□□住入侵者。

□□入侵的一□好□法是□□那些被修改、 □除或添加的文件。 □□文件修改的最佳方法是与某个 (通常是中央的)受限□□的系□上的文件□行比□。 在一台□格限制□□的系□上撰写□的安全脚本通常不能□被入侵者察□， 因此，□非常重要。□了最大限度地□□□一策略的□□，通常会使用只□的 NFS， 或者□置 ssh □匙□以便□其他机器提供□□。除了网□交互之外， NFS可能是一□很□被察□的方法 - 它允□□□控□一台客□机上的文件系□， 而□□□控几乎是无法察□的。如果一台□格受限的服□器和客□机是通□交□机□接的， 那□ NFS 将是一□非常好的方式。 不□，如果那台□控服□器和客□机之□通□集□器 (Hub)，或□□□多□的路由来□接，□□□方式就很不安全了， 此□，□考□使用 ssh ，即使□可以在□□□□中□到。

一旦□□个受限的机器□予了至少□取它□□控的客□系□的□限， 就□□□□□的□控撰写脚本。以 NFS 挂接□例，可以用□似 find(1) 和 md5(1) □□的命令□基□来完成我□所需的工作。 最好能□□天□被控机的所有□行文件□算一遍 md5，同□，□□以更高的□率□□那些 /etc 和 /usr/local/etc 中的控制文件。一旦□□了不匹配的情形，□控机□立即通知系□管理□。 好的安全脚本也□□□□在系□分区，如 / 和 /usr 中是否有新□或□除的可□行文件，以及不□宜的 suid 。

如果打算使用 ssh 来代替 NFS，那□撰写安全脚本将□得困□□多。 本□上，需要在脚本中使用 scp 在客□端□制文件， □一方面，用于□□的□行文件 (例如 find) 也需要使用 scp □到客□端，因□ ssh 客□程序很可能已□被攻陷。□之，在一条不□安全的□路上 ssh 可能是必□的， 但也必□□付它所□来的□□。

安全脚本□□□□□用□以及□□成□的□限□置文件： .rhosts、 .shosts、 .ssh/authorized_keys 等等。□些文件可能并非通□ MD5 来□行□□。

如果□的用□磁□空□很大， □□□□分区上面的文件可能非常耗□。 □□情况下， 采用□志来禁止使用 suid 可□行文件将是一个好主意。 □可能会想看看 nosuid □□ （参□ mount(8)）。 尽管如此， □些□描仍然□□至少□周□行一次， □□做的意□并不是□□有效的攻□， 而是□□攻□企□。

□程□□ (参□ accton(8)) 是一□相□成本□低的， 可以□助□在被入侵后□估□失的机制。 □于□出入侵者是如何□入系□的□件事情来□， 它会非常的有所助益，特□是当入侵者什□文件都没有修改的情况下。

最后， 安全脚本□□□理日志文件， 而日志文件本身□□通□尽可能安全的方法生成 - □程 syslog 可能非常有用。入侵者会□□掩□他□的踪迹， 而日志文件□于希望了解入侵□生□□的系□管理□来□□□得尤□重要。保持日志文件的永久性□□的一□方法是在串口上□行系□控制台， 并在一台安全的机器上收集□些信息。

15.3.7. 偏□

□点偏□不会□来□害。作□一□□例， 系□管理□在不影□使用的便利的前提下可以□用任何安全特性，此外， 在□□深思熟□之后，也可以□加一些 □□会 □使用□得不那□方便的安全特性。 更重要的是，有安全意□的管理□□□学会混合不同的安全策略 - 如果□逐字逐句地按照□□文□来配置□的机器， 那无□于向那些同□能得到□□文□的攻□者透露了更多的信息。

15.3.8. 拒□服□攻□

□一□将介□拒□服□攻□。 DoS 攻□通常是基于数据包的攻□， 尽管几乎没有任何□法来阻止大量的□造数据包耗尽网□□源， 但通常可以通□一些手段来限制□□攻□的□害，使它□无法□□服□器：

1. 限制服□□程 fork。
2. 限制 springboard 攻□ (ICMP □□攻□， ping 广播，等等)。
3. 使内核路由□存□□。

一□比□常□的 DoS 攻□情形， 是通□攻□□制□程 (fork) 的服□， 使其□生大量子□程， 从而是其□行的机器耗尽内存、 文件描述符等□源， 直到服□器□底死掉。 inetd (参□ inetd(8)) 提供了□多□□来限制□□攻□。 需要注意的是， 尽管能□阻止一台机器□底□掉， 但通常无法防止服□本身被□□。 □仔□□ inetd 的□机手册， 特□是它的 -c、 -C 以及 -R □三个□□。 □造 IP 攻□能□□□ inetd 的 -C □□， 因此， □些□□需要配合使用。 某些独立的服□器也有□似的限制参数。

例如， Sendmail 就提供了自己的 -OMaxDaemonChildren □□， 它通常比 Sendmail 的□□限制□□更□有效，因□服□器□□的□算有滞后性。 □可以在□□ sendmail □指定一个 MaxDaemonChildren 参数； 把它□的足□高以便承□□所需要的□荷， 当然， 不要高到足以□□行 Sendmail 的机器死掉。 此外， 以□列模式 (-ODeliveryMode=queued) □行 Sendmail 并把服□程序 (sendmail -bd) 和□列□行程序分□□行 (sendmail -q15m) 也是一个好主意。 如果□希望保□□列的□□性， 可以考□使用更短的□隔， 例如 -q1m， 但同□也需要指定一个合理的子□程数， 也就是通□ MaxDaemonChildren □□以免 那个Sendmail 造成重□的故障。

Syslogd 可以被直接地攻□，因此， □烈建□只要可行，就在□□它的□候加上 -s 参数， 其他情况下，□至少□□加上 -a。

□于基于□接的服□，例如 TCP Wrapper 的 reverse-identd， 都□□格外的小心， 因□它□都可能直接遭受攻□。 一般情况下， 基于安全考□， 不□使用 TCP Wrapper 所提供的 reverse-ident □□的功能。

此外， 将内部服□保□起来， 阻止来自其他主机的□□也十分重要， □些工作可以通□□置□界路由器来完成。主要的想法， 是阻止来自□的 LAN 以外的□□， □有助于避免 root 受到攻□。 尽可能配置排他式的防火□，例如， "用防火□阻止所有的网□流量 除了 端口 A、B、 C、D，以及 M-Z"。 通□采用□□方法，□可以很容易地将低端口的□□阻止在外， 而又不□配置使防火□放□那些明□需要□放的服□， 例如 named (如果□的机器准□作□域的主要解析服□器)， ntalkd， sendmail，以及其他可以从 Internet □□的服□。如果□□□以其他方式配置防火□ - 采用比□□松的策略， 那□□将很有可能忘□ "□掉" 一□个服□， 或者在□加了一些服□之后忘□更新防火□策略。 尽管如此， 仍然可以考□允□□数据□入□号□高的那一部分端口， □将保□那些需要□□特性的服□能□正常工作， 而又不影□低端口服□的安全性。 此外， □□注意到 FreeBSD 允□□来控制□□□定的端口的□□， 即一系列 net.inet.ip.portrange □量，通□ sysctl 来完成□置。 (sysctl -a | fgrep portrange)。 □使得□完成□□□的防火□策略□得易如反掌。 例如， □可能希望普通的高段端口的起止□□是 4000 到 5000， 而更高□□□是 49152 到 65535， 随后在防火□中阻止低于 4000 的所有端口 (当然，除了那些特地□ Internet □□而□□的端口)。

□一□常被称作 springboard 的攻□也是非常常□的 DoS 攻□ - 它通□使服□器□生其无法□理的□□来□到目的。最常□的攻□就是 *ICMP ping* 广播攻□。 攻□者通□□造 ping 包， 将其源 IP □置□希望攻□的机器的 IP。 如果□的□界路由器没有□行禁止 ping 广播地址的□置， □□的网□将最□陷于□□□造的 ping 包之中， 特□是当攻□者同□使用了多个不同的网□□。 广播攻□能□□生超□ 120 兆位的瞬□流量。 □一□常□的□□ ICMP □□□告系□的 springboard 攻□， 通□建立可以生成 ICMP 出□□□的包， 攻□者能□攻□服□器的网□下行□源， 并□致其上行□源耗尽。 □□□型的攻□也可以通□耗尽内存来使得使得被攻□的服□器崩□， 特□是当□些服□器无法足□快地完成 ICMP □□的□候。 □新的内核可以通□□整 sysctl □量 net.inet.icmp.icmplim 来限制□□攻□。 最后一□主要的 springboard 是□□某些 inetd 的内部服□， 例如 udp echo 服□□行的。 攻□者□□地□造一个来自服□器 A 的

echo □的 UDP 包， 然后将□个包□到 B 的 echo □。 于是， □台服□器将不停地将包□□□方。 攻□者能□将□台服□器的□□服□都耗竭， 并且通□□□方式， 只需要很少的包就可以□ LAN 超□。 □似的□□□ chargen □也是存在的。 好的系□管理□□□□□□些 inetd 的□□服□。

□造的包攻□也可以用来使内核的路由□存□□。 □参考 net.inet.ip.rtexpire， rtminexpire， 以及 rtmaxcache sysctl 参数。 □造的包可以用随机的源 IP 攻□， 使得内核在路由表中□生一个□□的□存□，它可以通□ netstat -rna | fgrep W3 看到。 □些路由通常需要 1600 秒才会□期。 如果内核□□路由表□得太大， 它会□□地降低 rtexpire 但以 rtminexpire □限。 □引□了□个□□：

1. 在□□量不大的服□器上， 内核□于突然□□的反□不□快。
2. rtminexpire 的□没有低到□内核在此□攻□□活下去的程度。

如果□的服□器通□ T3 或更快的□路接入 Internet， 那□通□ sysctl(8) 来手□地降低 rtexpire 和 rtminexpire 就非常必要。 当然，□不要把它□□置□零 (除非□想□机器崩□) 将□□个参数□置□ 2 通常已□足以抵御□□攻□了。

15.3.9. Kerberos 和 SSH 的□□□□

如果□打算使用， 那□ Kerberos 和 ssh 都有一些需要解决的□□。 Kerberos 5 是一个很棒的□□□□，但使用了它的 telnet 和 rlogin □用程序有一些 bug， 使得它□不□合□理二□制流。 而且， 除非使用了 -x □□，否□默□情况下 Kerberos 并不加密会□。 ssh 在默□□加密所有的会□内容。

除了默□□□加密密□之外， ssh 在所有的其他方面都做得很好。 □意味着如果□持有供□□□系□其他部分密□的工作站作了很好的安全防□， 而□□到了一台不安全的机器上， □□的密□可能被□人□得。 尽管□□的密□并没有被泄漏， 但由于 ssh 会在□登□的□程中□用一个□□端口， 如果攻□者拿到那台不安全的机器上的 root 那□他将能□利用那个端口来使用□的密□， 从而□□□能□□□的那些机器。

我□建□□在使用 ssh □配合 Kerberos 来完成工作人□的登□□程。 Ssh 在□□□可以加入 Kerberos 支持。 在□少了潜在地暴露 ssh 密□的机会的同□， 它□能□通□ Kerberos 来保□口令。 Ssh 密□只有在做□安全防□的机器上□行自□操作□才□使用 (□是 Kerberos 不□合的情形)。 此外，我□□建□□要□在 ssh 配置中□□密□□□，要□在 authorized_keys 中□加 from=IP/DOMAIN □□， 来限制□些密□能□登□的来源机器。

15.4. DES、 Blowfish、 MD5， 以及 Crypt

UNIX® 系□上的□个用□都有一个与其□□□□的口令。 很□然， 密□只需要被□个用□和操作系□知道。 □了保□口令的私密性， 采用了一□称□ "□向散列" 的方法来□理口令， □□地□， 很容易从口令推算出散列□，反之却很□。 其□， □才那句□可能并不十分□切： 因□操作系□本身并不 真的 知道□的口令。 它只知道口令□□加密的形式。 □取口令□□ "明文" 的唯一□法是采用暴力在口令可能的区□内□□。

不幸的是，当 UNIX® □□出□□，安全地加密口令的唯一方法基于DES， 数据加密□准 (the Data Encryption Standard)。 于是□□那些非美国居民□来了□□， 因□ DES 的源代□在当□不能被出口到美国以外的地方，FreeBSD 必□□到符合美国法律，但又要与其他那些使用 DES 的 UNIX® 版本兼容的□法。

解决方案是把加密函数□分割□□个， 于是美国的用□可以安装并使用 DES 函数□， 而国□用□□使用□外一套□提供的一□可以出口的加密算法。 □就是 FreeBSD □什□使用 MD5 作□它的默□加密算法的原因。 MD5 据信要比 DES 更安全，因此，安装 DES 更多地是出于兼容目的。

15.4.1. □□□采用的加密算法

□在□个□支持 DES、 MD5 和 Blowfish 散列函数。默□情况下， FreeBSD 使用 MD5 来加密口令。

可以很容易地□□ FreeBSD 使用□□加密方法。 □□ /etc/master.passwd 文件中的加密密□是一□方法。 用 MD5 散列加密的密□通常要比用 DES 散列得到的□一些， 并且以 1 字符□始。 以 $2a$ □始的口令是通□ Blowfish 散列函数加密的。 DES 密□字符没有任何可以用于□□的特征， 但他□要比 MD5 短， 并且以不包括 $ 在内的 64 个可□示字符来表示， 因此相□比□短的、没有以美元符号□□的字符串很可能是一个 DES 口令。

新口令所使用的密□格式是由 /etc/login.conf 中的 passwd_format 来控制的， 可供□□的算法包括 des, md5 和 blf。 □参考 login.conf(5) □机□助以□得更□一□的□情。

15.5. 一次性口令

默□情况下， FreeBSD 提供了 OPIE (One-time Passwords In Everything) 支持， 它默□使用 MD5 散列。

下面将介□三□不同的口令。 第一□是□常用的 UNIX® □格或 Kerberos 口令； 我□在后面的章□中将称其□ "UNIX® 口令"。 第二□是使用 OPIE 的 opiekey(1) 程序生成， 并□ opiepasswd(1) 以及登□提示所接受的一次性口令，我□称其□ "一次性口令"。 最后一□口令是□□入□ opiekey 程序 (有些□候是 opiepasswd 程序) 用以□生一次性口令的秘密口令，我□称其□ "秘密口令" 或通俗地□称□ "口令"。

秘密口令和□的 UNIX® 口令毫无□系， 尽管可以□置□相同的， 但不推□□做。 OPIE 秘密口令并不像旧式的 UNIX® 口令那□只能限于8位以内。 □想要用多□的口令都可以。 有六、七个□的短句是很常□的□□。 在□大多数□候， OPIE 系□和 UNIX® 口令系□完全相互独立地工作。

除了口令之外， □于 OPIE □有□□至□重要的数据。 其一被称作 "□子" 或 "key"， 它包括□个字符和五个数字。 □一个被称作 "迭代□数"， □是一个 1 到 100 之□的数字。 OPIE 通□将□子加到秘密口令后面， 并□行迭代□数那□多次的 MD4/MD5 散列□算来得到□果， 并将□果表示□ 6 个短的英文□□。 □ 6 个英文□□就是□的一次性口令。 □□系□ (主要是 PAM) 会□□上次使用的一次性口令， 如果用□提供的口令的散列□与上次一致， □可以通□身□□。 由于使用了□向的散列函数， 因此即使截□了上次使用的口令， 也没有□法恢□出下次将要使用的口令； □次成功登□都将□致迭代□数□□， □□用□和登□程序将保持同□。 □当迭代□数□少到 1 □， 都必□重新初始化 OPIE。

接下来将□□和□个系□有□的三个程序。 opiekey 程序能□接收□迭代□数， □子和秘密口令，并生成一个一次性口令， 或一□包含□□的一□一次性口令的表格。 opiepasswd 程序用于初始化 OPIE，并修改口令、 迭代次数、□子和一次性口令。 和 opieinfo 程序可以用于□□相□的□□数据文件 (/etc/opiekeys) 并□示□行命令的用□当前的迭代□数和□子。

我□将介□四□不同的操作。 在安全的□接上通□ opiepasswd 来第一次□置一次性口令， 或修改口令及□子。 第二□操作是在不安全的□接上使用 opiepasswd □以在安全□接上□行的 opiekey 来完成同□的工作。 第三□操作是在不安全的□接上使用 opiekey 来登□。 最后一□操作是采用 opiekey 来生成大批的密□，以便抄下来或打印出来，在没有安全□接的地方使用。

15.5.1. 安全□接的初始化

第一次初始化 OPIE □， 可以使用 opiepasswd 命令：

```
% opiepasswd -c
[grimreaper] ~ $ opiepasswd -f -c
Adding unfurl:
Only use this method from the console; NEVER from remote. If you are using
telnet, xterm, or a dial-in, type ^C now or exit with no password.
Then run opiepasswd without the -c parameter.
Using MD5 to compute responses.
Enter new secret pass phrase:
Again new secret pass phrase:
ID unfurl OTP key is 499 to4268
MOS MALL GOAT ARM AVID COED
```

在 Enter new secret pass phrase: 或 Enter secret password: 提示之后， □□入一个密□或口令字。□留意， □并不是□用于登□的口令， 它用于生成一次性的登□密□。 "ID" □一行□出了所需的参数： □的登□名，迭代□数， 以及□子。 登□系□□， 它能□□住□些参数并呈□□□， 因此无需□□它□。 最后一行□出了与□的秘密口令□□的、用于登□的一个一次性口令； 如果□立即重新登□， □它将是□需要使用的那个口令。

15.5.2. 不安全□接初始化

如果□需要通□一个不安全的□接来初始化， □□首先在安全□接上□行□一次 opiekey； □可能希望在可信的机器的 shell 提示符下完成。 此外□需要指定一个迭代□数 (100 也□是一个□好的□□) 也可以□□一个自己的□子， 或□□算机随机生成一个。 在不安全的□接上 (当然是□到□希望初始化的机器上)，使用 opiepasswd 命令：

```
% opiepasswd

Updating unfurl:
You need the response from an OTP generator.
Old secret pass phrase:
        otp-md5 498 to4268 ext
        Response: GAME GAG WELT OUT DOWN CHAT
New secret pass phrase:
        otp-md5 499 to4269
        Response: LINE PAP MILK NELL BUOY TROY

ID mark OTP key is 499 gr4269
LINE PAP MILK NELL BUOY TROY
```

□了接受默□的□子， 按下 Return （回□）。 在□入□□口令之前， 到一个有安全□接的机器上， 并□它同□的参数：

```
% opiekey 498 to4268
Using the MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret pass phrase:
GAME GAG WELT OUT DOWN CHAT
```

□在回到不安全的□接， 并将生成的一次性口令粘□到相□的□用程序中。

15.5.3. 生成一个一次性密□

一旦初始化□ OPIE， 当□登□□将看到□似□□的提示：

```
% telnet example.com
Trying 10.0.0.1...
Connected to example.com
Escape character is '^]'.

FreeBSD/i386 (example.com) (ttypa)

login: <username>
otp-md5 498 gr4269 ext
Password:
```

□外， OPIE 提示有一个很有用的特性 (□里没有表□出来)： 如果□在口令提示□按下 `Return` (回□) 系□将回□□□入的口令， □可以藉此看到自己所□入的内容。 如果□□手工□入一个一次性密□， □会非常有用。

此□□需要生成一个一次性密□来回答□一提示。 □□工作必□在一个可信的系□上□行 `opiekey` 来完成。 (也可以□到 DOS、 Windows® 以及 Mac OS® 等操作系□上□行的版本)。 □个程序需要将迭代□数和□子提供□它。 □可以从登□提示那里□制和粘□它□。

在可信的系□上：

```
% opiekey 498 to4268
Using the MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret pass phrase:
GAME GAG WELT OUT DOWN CHAT
```

□在就可以用□□□得的一次性口令登□了。

15.5.4. □生多个一次性口令

有□，会需要到不能□□可信的机器或安全□接的地方。 □□情形下， 可以使用 `opiekey` 命令来一次生成□多一次性口令。 例如：

```
% opiekey -n 5 30 zz99999
Using the MD5 algorithm to compute response.
Reminder: Don't use opiekey from telnet or dial-in sessions.
Enter secret pass phrase: <secret password>
26: JOAN BORE FOSS DES NAY QUIT
27: LATE BIAS SLAY FOLK MUCH TRIG
28: SALT TIN ANTI LOON NEAL USE
29: RIO ODIN GO BYE FURY TIC
30: GREW JIVE SAN GIRD BOIL PHI
```

`-n 5` 按□序□求 5 个口令， `30` □指定了最后一个迭代□数□□是多少。 注意□些口令将按与使用□序相反的□序来□示。 如果□比□偏□， 可以手工写下□些□果； 一般来□把它粘□到 `lpr` 就可以了。 注意，□一行都□示迭代□数及其□□的一次性的密□； 一些人建□用完一个就□掉一个。

15.5.5. 限制使用 UNIX® 口令

OPIE 可以□ UNIX® 口令的使用□行基于 IP 的登□限制。 □□的文件是 /etc/opieaccess， □个文件默□情况下就是存在的。 □参□ opieaccess(5) 以了解□于□个文件□一□的情况， 以及安全方面需要□行的一些考□。

下面是一个示□的 opieaccess 文件：

```
permit 192.168.0.0 255.255.0.0
```

□行允□指定 IP 地址的用□ (再次□□□□地址容易被□造) 在任何□候使用 UNIX® 口令登□。

如果 opieaccess 中没有匹配的□□， □将默□拒□任何非 OPIE 登□。

15.6. TCP Wrappers

□一个熟悉 inetd(8) 都□□听□□ TCP Wrappers， 但几乎没有人□它在网□□境中的作用有全面的理解。 几乎□个人都会安装防火□来□理网□□接， 然而□然防火□有非常广泛的用途， 它却不是万能的， 例如它无法□理□似向□接□起者□送一些文本□□的任□。 而 TCP Wrappers □件能□完成它以及更多的其他事情。 接下来的几段中将□□□多 TCP Wrappers 提供的功能， 并且， □□出了一些配置□例。

TCP Wrappers □件□展了 inetd □受其控制的服□程序□施控制的能力。 通□使用□□方法， 它能□提供日志支持、 返回消息□□入的□接、 使得服□程序只接受内部□接， 等等。 尽管防火□也能□完成其中的某些功能， 但□不□□加了一□□外的保□， 也提供了防火□无法提供的功能。

然而， 由 TCP Wrappers 提供的一些□外的安全功能， 不□被□□好的防火□的替代品。 TCP Wrappers □□合防火□或其他安全加□□施一并使用， □系□多提供一□安全防□。

由于□些配置是□于 inetd 的□展， 因此， □者□首先□□ 配置 inetd □□。

尽管由 inetd(8) □行的程序并不是真正的 "服□程序"， 但□□上也把它□称□服□程序。 下面仍将使用□一□□。

15.6.1. 初始配置

在 FreeBSD 中使用 TCP Wrappers 的唯一要求是□保 inetd 在从 rc.conf 中□□□包含了 `-Ww` □□； □是默□的□置。 当然， □需要□ /etc/hosts.allow □行□当的配置， 但 syslogd(8) 在配置不当□会在系□日志中□□相□消息。

与其它的 TCP Wrappers □□不同， 使用 hosts.deny 在□里被□□是不推□和□□的做法。 所有的配置□□□放到 /etc/hosts.allow 中。

在最□□的配置中， 服□程序的□接策略是根据 /etc/hosts.allow 允□或阻止。 FreeBSD 中的默□配置是允□一切□到由 inetd 所□□的服□的□接□求。 在基本配置之后将□□更□□的情况。

基本配置的形式通常是 服□ : 地址 : □作。 □里 服□ 是从 inetd □□的服□程序的名字。 而 地址 可以是任何有效的主机名、 一个 IP 或由方括号 ([ ]) 括起来的 IPv6 地址。 □作 字段可以使 allow 或 deny， 分□用于允□和禁止相□的□□。 在配置□□需要注意所有的配置都是按照第一个匹配的□□□□的， □表示配置文件将按照□序□□匹配□□， 而一旦□到匹配， □搜索也就停止了。

□外也有□多其他□□， □些将在后面介□。 □□的配置行从上面□些描述之中可以很容易得出。 例如， 允□ POP3 □接通□ mail/qpopper 服□， □把下面的行添加到 hosts.allow：

```
# This line is required for POP3 connections:
qpopper : ALL : allow
```

□加□□之后， 需要重新□□ inetd。 可以通□使用 kill(1) 命令来完成□□工作， 或使用 /etc/rc.d/inetd 的 restart parameter 参数。

15.6.2. 高□配置

TCP Wrappers 也有一些高□的配置□□； 它□能□用来□如何□理□接□施更多的控制。 一些□候， 返回一个□明到特定的主机或□求服□的□接可能是更好的□法。 其他情况下， □□日志或者□送□件□管理□可能更□□合。 □外， 一些服□可能只希望□本机提供。 □些需求都可以通□使用 通配符， □展字符以及外部命令来□□。 接下来的□□将介□□些。

15.6.2.1. 外部命令

假□由于□生了某□状况， 而□致□接□□被拒□掉， 而将其原因□送□□起□接的人。 如何完成□□的任□□？ □□的□作可以通□使用 twist □□来□□。 当□起了□接□求□， twist 将□用一个命令或脚本。 在 hosts.allow 文件中已□□出了一个例子：

```
# The rest of the daemons are protected.
ALL : ALL \
        : severity auth.info \
        : twist /bin/echo "You are not welcome to use %d from %h."
```

□个例子将把消息 "You are not allowed to use daemon from hostname." 返回□□□先前没有配置□允□□□的服□客□。 □于希望把消息反□□□接□起者， 然后立即切断□□的需求来□， □□的配置非常有用。 □注意所有反□信息必□ 被引号 " 包□； □一□□是没有例外的。

如果攻□者向服□程序□送大量的□接□求， □可能□□一次成功的拒□服□攻□。

□一□可能是□□□□情况使用 spawn。 □似 twist， spawn □□也暗含拒□□接， 并可以用来□行外部命令或服□。 与 twist 不同的是， spawn 不会向□接□起者□送回□。 考□下面的配置：

```
# We do not allow connections from example.com:
ALL : .example.com \
    : spawn (/bin/echo %a from %h attempted to access %d >> \
      /var/log/connections.log) \
    : deny
```

□将拒□来自 *.example.com 域的所有□接； 同□□将□□主机名， IP 地址， 以及□方所□□□接的服□名字到 /var/log/connections.log 文件中。

除了前面已□介□□的□□字符， 例如 %a 之外， □有一些其它的□□符。 参考 hosts_access(5) □机手册可以□得完整的列表。

15.6.2.2. 通配符□□

前面的例子都使用了 ALL。 其它□□能□将功能□展到更□。 例如， ALL 可以被用来匹配□一个服□、 域，或 IP 地址。 □一些可用的通配符包括 PARANOID， 它可以用来匹配任何来自可能被□造的 IP 地址的主机。 □言之， paranoid 可以被用来定□来自 IP 与其主机名不符的客□。 下面的例子将□□更多的感性□□：

```
# Block possibly spoofed requests to sendmail:
sendmail : PARANOID : deny
```

在□个例子中， 所有□接 sendmail 的 IP 地址与其主机名不符的主机都将被拒□。

如果服□器和客□机有一方的 DNS 配置不正□， 使用 PARANOID 可能会□重地削弱服□。 在□置之前， 管理□□□□慎地考□。

要了解□于通配符和他□的功能， □参考 hosts_access(5) □机手册。

□了使□置能□生效， □□首先把 hosts.allow 的第一行配置注□掉。 □□的□始部分已□□明了□一点。

15.7. Kerberos5

Kerberos 是一□附加的网□系□/□□， 用以□用□通□一台安全服□器提供的服□来□□身□。 包括□程登□、□程□制、在系□□安全地□制文件， 以及其它高危□性的操作， 由于其存在而□著地提高了安全型并且更加可控。

Kerberos 可以理解□一□身□□□代理系□。 它也被描述□一□以受信第三方□主□的身□□□系□。 Kerberos 只提供一□功能 - 在网□上安全地完成用□的身□□□。 它并不提供授□功能 (也就是□用□能□做什□操作) 或□□功能 (□□用□作了什□操作)。 一旦客□和服□器都使用了 Kerberos 来□明各自的身□之后， 他□□可以加密全部的通□以保□□□数据的私密性和完整性。

因此， □烈建□将 Kerberos 同其它提供授□和□□服□的安全手段□用。

接下来的□明可以用来指□如何安装 FreeBSD 所附□的 Kerberos。 不□， □仍然需要参考相□的□机手册以□得完整的描述。

□了展示 Kerberos 的安装□程， 我□□定：

- DNS 域 ("zone") □ example.org。
- Kerberos □域是 EXAMPLE.ORG。

在安装 Kerberos □□使用□□的域名即使□只是想在内部网上用一用。 □可以避免 DNS □□并保□了同其它 Kerberos 之□的互操作性。

15.7.1. □史

Kerberos 最早由 MIT 作□解决网□安全□□的一个方案提出。 Kerberos □□采用了□加密， 因此客□能□在不安全的网□上向服□器 (以及相反地) □□自己的身□。

Kerberos 是网□□□□□名字， 同□也是用以表□□□了它的程序的形容□。 (例如 Kerberos telnet)。 目前最新的□□版本是 5，在 RFC 1510 中有所描述。

□□□有□多免□的□□， □些□□涵□了□多□不同的操作系□。 最初研制 Kerberos 的麻省理工学院 (MIT) 也仍然在□□□□他□的 Kerberos □件包。 在 US 它被作□一□加密□品使用， 因而□史上曾□受到 US 出口管制。 MITKerberos 可以通□ port (security/krb5) 来安装和使用。 Heimdal Kerberos 是□一□第 5 版□□， 并且明□地在 US 之外的地区□□， 以避免出口管制 (因此在□多非商□的□ UNIX® 系□中非常常用。 Heimdal Kerberos □件包可以通□ port (security/heimdal) 安装， 最新的 FreeBSD 的最小安装也会包含它。

□使尽可能多的□者从中受益， □□□明以 FreeBSD 附□的 Heimdal □件包□准。

15.7.2. 配置 Heimdal KDC

密□分□中心 (KDC) 是 Kerberos 提供的集中式□□服□ - 它是□□ Kerberos tickets 的那台□算机。 KDC 在 Kerberos □域中的其它机器看来是 "受信的"， 因此必□格外注意其安全性。

需要□明 Kerberos 服□器只需要非常少的□算□源， 尽管如此， 基于安全理由仍然推□使用独占的机器来扮演 KDC 的角色。

要□始配置 KDC， 首先□□□□的 /etc/rc.conf 文件包含了作□一个 KDC 所需的□置 (□可能需要□当地□整路径以□□自己系□的情况)：

```
kerberos5_server_enable="YES"
kadmind5_server_enable="YES"
```

接下来需要修改 Kerberos 的配置文件， /etc/krb5.conf：

```
[libdefaults]
    default_realm = EXAMPLE.ORG
[realms]
    EXAMPLE.ORG = {
        kdc = kerberos.example.org
        admin_server = kerberos.example.org
    }
[domain_realm]
    .example.org = EXAMPLE.ORG
```

□注意□个 /etc/krb5.conf 文件假定□的 KDC 有一个完整的主机名， 即 kerberos.example.org。 如果□的 KDC 主机名与它不同， □□添加一条 CNAME (□名) □到 zone 中去。

□于有正□地配置□的 BINDDNS 服□器的大型网□， 上述例子可以精□□：

```
[libdefaults]
      default_realm = EXAMPLE.ORG
```

将下面的内容加入到 example.org zone 数据文件中：

```
_kerberos._udp      IN  SRV     01 00 88 kerberos.example.org.
_kerberos._tcp      IN  SRV     01 00 88 kerberos.example.org.
_kpasswd._udp       IN  SRV     01 00 464 kerberos.example.org.
_kerberos-adm._tcp  IN  SRV     01 00 749 kerberos.example.org.
_kerberos           IN  TXT     EXAMPLE.ORG
```

要□客□机能□□到 Kerberos 服□， 就 必□ 首先配置完整或最小配置的 /etc/krb5.conf并且正□地配置 DNS 服□器。

接下来需要□建 Kerberos 数据□。 □个数据□包括了使用主密□加密的所有□体的密□。 □并不需要□住□个密□，它会保存在一个文件 (/var/heimdal/m-key) 中。 要□建主密□， 需要□行 kstash 并□入一个口令。

主密□一旦建立， □就可以用 kadmin 程序的 -l 参数 (表示 "local") 来初始化数据□了。 □个□□□ kadmin 直接地修改数据□文件而不是通□ kadmind 的网□服□。 □解决了在数据□□建之前□接它的□生蛋的□□。 □入 kadmin 提示符之后， 用 init 命令来□建□域的初始数据□。

最后， 仍然在 kadmin 中， 使用 add 命令来□建第一个 principal。 □□使用全部的默□□置， 随后可以在任何□候使用 modify 命令来修改□些□置。 □外， 也可以用 ? 命令来了解可用的□□。

典型的数据□□建□程如下：

```
# kstash
Master key: xxxxxxxx
Verifying password - Master key: xxxxxxxx

# kadmin -l
kadmin> init EXAMPLE.ORG
Realm max ticket life [unlimited]:
kadmin> add tillman
Max ticket life [unlimited]:
Max renewable life [unlimited]:
Attributes []:
Password: xxxxxxxx
Verifying password - Password: xxxxxxxx
```

□在是□□ KDC 服□的□候了。 □行 /etc/rc.d/kerberos start 以及 /etc/rc.d/kadmind start 来□□□些服□。 尽管此□□没有任何正在□行的 Kerberos 服□， 但□仍然可以通□□取并列出□□□□建的那个 principal (用□) 的 ticket 来□□ KDC □□在正常工作， 使用 KDC 本身的功能：

```
% kinit tillman
tillman@EXAMPLE.ORG's Password:

% klist
Credentials cache: FILE:/tmp/krb5cc_500
    Principal: tillman@EXAMPLE.ORG

  Issued           Expires          Principal
Aug 27 15:37:58  Aug 28 01:37:58  krbtgt/EXAMPLE.ORG@EXAMPLE.ORG
```

完成所需的操作之后， 可以撤消□一 ticket：

```
% kdestroy
```

15.7.3. □ Kerberos □用 Heimdal 服□

首先我□需要一□ Kerberos 配置文件 /etc/krb5.conf 的副本。 只需□□地用安全的方式 (使用□似 scp(1) 的网□工具， 或通□□□) □制 KDC 上的版本， 并覆□掉客□机上的□□文件就可以了。

接下来需要一个 /etc/krb5.keytab 文件。 □是提供 Kerberos 服□的服□器和工作站的一个主要区□ - 服□器必□有 keytab 文件。 □个文件包括了服□器的主机密□， □使得 KDC 得以□□它□的身□。 此文件必□以安全的方式□到服□器上， 因□如果密□被公之于□， □安全也就□于一旦。 也就是□， 通□明文的通道， 例如 FTP 是非常糟□的想法。

一般来□， □会希望使用 kadmin 程序来把 keytab □到服□器上。 由于也需要使用 kadmin 来□主机建立 principal (KDC 一端的 krb5.keytab)， 因此□并不□□。

注意□必□已□□得了一个 ticket 而且□个 ticket 必□□可使用 kadmind.acl 中的 kadmin 接口。 □参考 Heimdal info 中的 "Remote administration(□程管理)" 一□ (info heimdal) 以了解如何□□□□控制表。 如果不希望□

用□程的 kadmin 操作， □可以□□地采用安全的方式□接 KDC (通□本机控制台， ssh(1) 或 Kerberos telnet(1)) 并使用 kadmin -l 在本地□行管理操作。

安装了 /etc/krb5.conf 文件之后， □就可以使用 Kerberos 上的 kadmin 了。 add --random-key 命令可以用于添加主机 principal， 而 ext 命令□允□□出服□器的主机 principal 到它的 keytab 中。 例如：

```
# kadmin
kadmin> add --random-key host/myserver.example.org
Max ticket life [unlimited]:
Max renewable life [unlimited]:
Attributes []:
kadmin> ext host/myserver.example.org
kadmin> exit
```

注意 ext 命令 (□是 "extract" 的□写) 默□会把□出的密□放到 /etc/krb5.keytab 中。

如果□由于没有在 KDC 上□行 kadmind (例如基于安全理由) 因而无法□程地使用 kadmin □可以直接在 KDC 上添加主机 principal (host/myserver.EXAMPLE.ORG) 随后将其□出到一个□□文件中 (以免覆□ KDC 上的 /etc/krb5.keytab)， 方法是使用下面的命令：

```
# kadmin
kadmin> ext --keytab=/tmp/example.keytab host/myserver.example.org
kadmin> exit
```

随后需要把 keytab □制到服□器上 (例如使用 scp 或□□)。 一定要指定一个不同于默□的 keytab 名字以免覆□ KDC 上的 keytab。

到□在□的服□器已□可以同 KDC 通□了 (因□已□配置了 krb5.conf 文件)， 而且它□能□□明自己的身□ (由于配置了 krb5.keytab 文件)。 □在可以□用一些 Kerberos 服□。 在□个例子中， 我□将在 /etc/inetd.conf 中添加下面的行来□用 telnet 服□， 随后用 /etc/rc.d/inetd restart 重□ inetd(8) 服□来使□置生效：

```
telnet    stream  tcp     nowait  root    /usr/libexec/telnetd  telnetd -a user
```

□□的部分是 -a (表示□□) □型□置□用□ (user)。 □参考 telnetd(8) □机手册以了解□□。

15.7.4. 使用 Heimdal 来□用客□端 Kerberos

□置客□机是非常□□的。 在正□配置了 Kerberos 的网□中， 只需要将位于 /etc/krb5.conf 的配置文件□行一下□置就可以了。 □一□□可以□□地通□安全的方式将文件从 KDC □制到客□机上来完成。

□□在客□机上□行 kinit、 klist， 以及 kdestroy 来□□□取、 □示并□除 □□□ principal 建立的 ticket 是否能□正常□行， 如果能， □用其它的 Kerberos □用程序来□接□用了 Kerberos 的服□。 如果□用程序不能正常工作而□取 ticket 正常， □通常是服□本身， 而非客□机或 KDC 有□□。

在□□□似 telnet 的□用程序□， □考□使用□包程序 (例如 tcpdump(1)) 来□□□的口令没有以明文方式□□。 □□使用 telnet 的 -x 参数， 它将加密整个数据流 (□似 ssh)。

□多非核心的 Kerberos 客□□用程序也是默□安装的。 在 Hemidal 的 "最小" 安装理念下， telnet 是唯一一个采用了 Kerberos 的服□。

Heimdal port □提供了一些默□不安装的客□□用程序， 例如□用了 Kerberos 版本的 ftp、 rsh、 rcp、 rlogin 以及一些更不常用的程序。 MIT port 也包括了一整套 Kerberos 客□□用程序。

15.7.5. 用□配置文件： .k5login 和 .k5users

在某个□域中的用□往往都有自己的 Kerberos principal (例如 tillman@EXAMPLE.ORG) 并映射到本机用□□□ (例如本机上名□ tillman 的□□)。 客□端□用程序， 如 telnet 通常并不需要用□名或 principal。

不□， 有□□可能会需要□予某些没有匹配 Kerberos principal 的人使用本地用□□□的□限。 例如 tillman@EXAMPLE.ORG 可能需要□□本地的 webdevelopers 用□□号。 其它 principal 可能也会需要□□□个本地□号。

用□ home 目□中的 .k5login 和 .k5users □□个文件可以配合 .hosts 和 .rhosts 来有效地解决□个□□。 例如，如果 .k5login 中有如下内容：

```
tillman@example.org
jdoe@example.org
```

并放到了本地用□ webdevelopers 的 home 目□中， □列出的□个 principals 都可以使用那个□号， 而无□共享口令。

建□□在□始□施之前首先□□□些命令的□机□助。 特□地， ksu 的□机手册包括了 .k5users 的相□内容。

15.7.6. Kerberos 提示、技巧和故障排除

- 当使用 Heimdal 或 MITKerberos ports □， 需要□□ PATH □境□量把 Kerberos 客□□用列在系□自□的版本之前。
- 同一□域内的所有□算机的□□□置是否同□？ 如果不是的□， □身□□□可能会失□。 通□ NTP □行□□同□描述了如何使用 NTP 来同□□□。
- MIT 和 Heimdal 能□很好地互操作。 一个例外是 kadmin， 因□□个□□没有被□准化。
- 如果□改□了主机名， □□需要修改□的 host/ principal 并更新 keytab。 □一□律也□用于□似 Apache 的 www/mod_auth_kerb 所使用的 www/ principal □□的特殊 keytab □。
- □的□域中的□一台主机必□在 DNS (或至少在 /etc/hosts 中) 可以解析 (同□包括正向和反向)。 CNAME 能□正常使用， 但必□有正□的□□ A 和 PTR □□。 此□□出的□□信息可能很□人困惑： Kerberos5 refuses authentication because Read req failed: Key table entry not found。
- 某些作□客□使用□的 KDC 的操作系□可能没有将 ksu □置□ setuid root 的□限。 □意味着 ksu 将不能□正常工作， 从安全角度□□是一个不□的主意， 但可能令人□□。 □□□□并不是 KDC 的□□。
- 使用 MITKerberos □， 如果希望允□一个 principal □有超□默□的十小□有效期的 ticket □必□使用 kadmin 中的 modify_principal 来修改 principal 本身以及 krbtgt 的 maxlife(最大有效期)。 此后， principal 可以使用 kinit 的 -l 参数来□求一个有更□有效期的 ticket。 *

如果在 KDC 上□行了听包程序， 并在工作站上□行 kinit， □可能会注意到 TGT 是在 kinit 一□始□行的□候就□出了的 - 甚至在□□入口令之前！ □于□个□象的解□是 Kerberos 服□器可以无限制地收□ TGT (Ticket Granting Ticket) □任何未□授□的□求； 但是， □一个 TGT 都是使用用□的口令派生出来的密□□行加密的。 因此， 当用□□入口令□它并不会□送□ KDC， 而是直接用于解密 kinit 所拿到的 TGT。 如果解密□程得到了一个包含合法的□□□的有效 ticket， □□明用□的 Kerberos 凭据有效。 □些凭据包含了一个会□密□用以在随后建立 Kerberos 服□器的加密通□， □□由服□器自己的私□加密的□□的 ticket-granting ticket。 □个第二□加密□于用□来□是看不到的， 但它使得 Kerberos 服□器能□□□□一个 TGT 的真□性。

- 如果需要有效期更□的 ticket (例如一周) 而且□使用 OpenSSH □接保存□的 ticket 的机器， □□□ sshd_config 中的 Kerberos TicketCleanup 被□置□ no 否□在注□□会自□□除所有的 ticket。
- 切□主机的 principals 的 ticket 有效期一定要比用□的□。 如果□的用□ principal 的有效期是一周， 而所□接的主机的有效期是九个小□， □□存的主机 principal 将先行□期， □果是 ticket □存无法正常工作。
- 当配置 krb5.dict 文件来防止使用特定的□□口令 (kadmind 的□机手册中□要介□了它)， □切□只有指定了口令策略的 principals 才会使用它□。 krb5.dict 文件的格式很□□： □个串占一行。 □建一个到 /usr/shared/dict/words 的符号□接会很有用。

15.7.7. 与 MIT port 的区□

MIT 和 Heimdal 主要的区□在于 kadmin 程序使用不同 (尽管等价) 的命令和□□。 如果□的 KDC 是 MIT 的， □其影□是不能使用 Heimdal 的 kadmin 程序来□程管理 KDC (或相反)。

完成同□工作的命令可能会有些□的不同。 推□按照 MITKerberos 的网站 (http://web.mit.edu/Kerberos/www/) 上的□明来操作。 □小心□于路径的□□， MIT port 会默□安装到 /usr/local/， □因此可能会□行 "普通的" 系□□用程序而非 MIT, 如果□的 PATH □境□量把 把系□目□放在前面的□。

如果使用 FreeBSD 提供的 MITsecurity/krb5 port， 一定要仔□□□ port 所安装的 /usr/local/shared/doc/krb5/README.FreeBSD， 如果□想知道□什□通□ telnetd 和 klogind 登□□会出□一些□□的□象的□。 最重要地， "incorrect permissions on cache file(□存文件□限不正□)" 行□需要使用 login.krb5 来□行□□， 才能□正□地修改□□凭据的属主。

除此之外， □□修改 rc.conf 并加入下列配置：

```
kerberos5_server="/usr/local/sbin/krb5kdc"
kadmind5_server="/usr/local/sbin/kadmind"
kerberos5_server_enable="YES"
kadmind5_server_enable="YES"
```

□□做的原因是， MIT kerberos 会将可□行文件装到 /usr/local 之下。

15.7.8. □解 Kerberos 的限制

15.7.8.1. Kerberos 是一□ all-or-nothing 方式

在网□上□用的□个服□都必□□行修改以便□其能□配合 Kerberos 工作 (否□就只能使用其它方法来保□它□不受网□攻□的侵害)， 如果不是□□， □用□的凭据就有可能被窃取并再次使用。 一个例子是□所有的□程 shell (例如通□ `rsh` 和 `telnet`) □用了 Kerberos 但没有将使用明文□□的 POP3 □件服□器 Kerberos化。

15.7.8.2. Kerberos 是□□用□工作站□□的

在多用□□境中 Kerberos 的安全性会被削弱。 □是因□它把 ticket 保存到 /tmp 目□中， 而□个目□可以被任何用□□取。 如果有用□与其它人同□共享一台□算机 (也就是 multi-user)， □□个用□的 ticket 就可能被其它用□窃取 (□制)。

可以通□使用 `-c` 文件名 □□的命令行□□， 或者(推□的)改□ `KRB5CCNAME` □境□量来避免□个□□， 但很少有人□□做。原□上， 将 ticket 保存到用□的 home 目□并□□地□置□限就能□□解□个□□。

15.7.8.3. KDC 会成□□点崩□故障点

根据□□， KDC 必□是安全的， 因□主密□数据□保存在它上面。 决不□□在 KDC上面□行其它服□， 而且□□□保它的物理安全。 由于 Kerberos 使用同一个密□ (□□中的那个 "主" 密□) 来加密所有的密□， 而将□个文件保存在 KDC， 因此其安全尤□重要

不□， 主密□的泄露并没有想象中的那□可怕。 主密□只用来加密 Kerberos 数据□以及□生随机数□生器的□子。只要 KDC 是安全的， 即使攻□者拿到了主密□也做不了什□。

□外， 如果 KDC 不可用 (例如由于拒□服□攻□或网□故障) □网□服□将由于□□服□无法□行而不能使用， 从而□致更大□□的拒□服□攻□。 通□部署多个 KDC (一个主服□器， 配合一个或多个从服□器) 并采用□□仔□□□和□□的□用□□方式可以避免□□□□ (PAM 是一个不□的□□)。

15.7.8.4. Kerberos 的不足

Kerberos 允□用□、主机和服□之□□行相互□□。 但它并没有提供机制来向用□、主机或服□□□ KDC。 □意味着□□木□的程序，例如 `kinit` 有可能□□用□所有的用□名和密□。 尽管如此， 可以用□似 security/tripwire □□的文件系□完整性□□工具来避免此□情况的□生。

15.7.9. 相□□源和其它□料

- The Kerberos FAQ
- Designing an Authentication System: a Dialog in Four Scenes
- RFC 1510, The Kerberos Network Authentication Service (V5)
- MIT Kerberos home page
- Heimdal Kerberos home page

15.8. OpenSSL

□多用□可能并没有注意到 FreeBSD 所附□的 OpenSSL 工具包的功能。 OpenSSL 提供了建立在普通的通□□基□上的加密□□□； □些功能□□多网□□用和服□程序所广泛使用。

□ OpenSSL 的一些常□用法包括加密□件客□的身□□□□程， 基于 Web 的交易如信用□等等。 □多 ports 如 www/apache13-ssl， 以及 mail/claws-mail 等等都提供了□□□ OpenSSL 支持的方法。

□大多数情况下 Ports Collection 会□□使用 security/openssl 除非明□地将 WITH_OPENSSL_BASE make □量□置□ "yes"。

FreeBSD 中附□的 OpenSSL 版本能□支持 安全套接字□ v2/v3 (SSLv2/SSLv3) 和 安全□□□ v1 (TLSv1) 三□网□□□， 并可作□通用的密□学函数□使用。

尽管 OpenSSL 支持 IDEA 算法， 但由于美国□利， 它在默□情况下是不□□的。 如果想使用它， □□□相□的授□， 如果□□授□可以接受， □可以在 make.conf 中□置 MAKE_IDEA。

□□用□件提供□□是 OpenSSL 最□常用的功能之一。 □□是一□能□□保公司或个人有效身□不被□造的凭据。 如果□□没有被□多 "□威□□机□"， 或 CA 中的某一个□□， □会□生一个警告。 □威□□机□通常是一家公司， 例如 VeriSign， 它能□通□□署来□明个人或公司□□的有效性。 □个□程是需要付□的， 当然， □不是使用□□的必要条件； 然而， □□做会□那些比□偏□的用□感到□松。

15.8.1. 生成□□

□了生成□□， 需要使用下面的命令：

```
# openssl req -new -nodes -out req.pem -keyout cert.pem
Generating a 1024 bit RSA private key
.................++++++
.........................................++++++
writing new private key to 'cert.pem'
-----
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) [AU]:US
State or Province Name (full name) [Some-State]:PA
Locality Name (eg, city) []:Pittsburgh
Organization Name (eg, company) [Internet Widgits Pty Ltd]:My Company
Organizational Unit Name (eg, section) []:Systems Administrator
Common Name (eg, YOUR name) []:localhost.example.org
Email Address []:trhodes@FreeBSD.org

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:SOME PASSWORD
An optional company name []:Another Name
```

□注意， 在 "Common Name" 提示后面我□□入的是一个域名。 □个提示要求□入服□器的名字，

□个名字今后将用于完成□□□程； 如果在□里□入域名以外的内容， 那□□□也就失去其意□了。 □□可以指定一些其他的□□， 比如□□的有效期， 以及使用的加密算法等等。 □些□□的完整列表， 可以在 openssl(1) □机手册中□到。

在□□行前述命令的目□中将生成□个文件。 □□申□， 即 req.pem， 可以□□一家□□机□， 它将□□□□入的凭据的真□性， 并□申□□行□名， 再把□□返□□□。 第二个文件的名字将是 cert.pem， 它包含了□□的私□，□被全力保□； 如果它落入□人手中， □可以被用来□造□ (或□的服□器)。

如果不需要来自 CA 的□名， 也可以□建自行□名的□□。 首先， 需要生成 RSA 密□：

```
# openssl dsaparam -rand -genkey -out myRSA.key 1024
```

接下来， 生成 CA 密□：

```
# openssl gendsa -des3 -out myca.key myRSA.key
```

然后用□个密□来□建□□：

```
# openssl req -new -x509 -days 365 -key myca.key -out new.crt
```

上述□□将在当前目□中生成□个新文件： 一个是□威□□机□的□名文件， myca.key； □一个是□□本身，new.crt。 □些文件□□放到同一个目□中， 一般而言， 推□放到 /etc， 并且只允□ root □取。 建□把□限□置□0700， □可以通□ chmod 工具来完成。

15.8.2. 使用□□的一个例子

那□有了□些文件可以做些什□□？ 一个比□典型的用法是用来加密 SendmailMTA 的通□□接。 □可以解决用□通□本地 MTA □送□件□使用明文□行身□□□的□□。

□个用法可能并不完美， 因□某些 MUA 会由于没有在本地安装□□而向用□□出警告。□参考那些□件的□明了解□于安装□□的信息。

下面的□置□添加到本地的 .mc 文件

```
dnl SSL Options
define(`confCACERT_PATH',`/etc/certs')dnl
define(`confCACERT',`/etc/certs/new.crt')dnl
define(`confSERVER_CERT',`/etc/certs/new.crt')dnl
define(`confSERVER_KEY',`/etc/certs/myca.key')dnl
define(`confTLS_SRV_OPTIONS', `V')dnl
```

□里， /etc/certs/ 是准□用来在本地保存□□和密□的位置。 最后， 需要重新生成本地的 .cf 文件。 □一工作可以□□地通□在 目□中□行 makeinstall 来完成。 接下来， 可以使用 makerestart 来重新□□Sendmail 服□程序。

如果一切正常的□， 在 /var/log/maillog 中就不会出□□□提示， Sendmail 也□□出□在□程列表中。

做一个□□的□□， 使用 telnet(1) 来□接□件服□器：

```
# telnet example.com 25
Trying 192.0.34.166...
Connected to example.com.
Escape character is '^]'.
220 example.com ESMTP Sendmail 8.12.10/8.12.10; Tue, 31 Aug 2004 03:41:22 -0400 (EDT)
ehlo example.com
250-example.com Hello example.com [192.0.34.166], pleased to meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-8BITMIME
250-SIZE
250-DSN
250-ETRN
250-AUTH LOGIN PLAIN
250-STARTTLS
250-DELIVERBY
250 HELP
quit
221 2.0.0 example.com closing connection
Connection closed by foreign host.
```

如果□出中出□了 "STARTTLS" □□明一切正常。

15.9. IPsec 上的 VPN

使用 FreeBSD 网□在□个被 Internet 分□的网□之□架□ VPN。

15.9.1. 理解 IPsec

□一□将指□□完成架□ IPsec。□了配置 IPsec， □□当熟悉如何□□一个定制的内核的一些概念 (参□ 配置FreeBSD的内核)。

IPsec 是一□建立在 Internet □□ (IP) □之上的□□。 它能□□□个或更多主机以安全的方式来通□ (并因此而得名)。 FreeBSD IPsec "网□□□□" 基于 KAME 的□□， 它支持□□□□族， IPv4 和 IPv6。

IPsec 包括了□个子□□：

- *Encapsulated Security Payload (ESP)*, 保□ IP 包数据不被第三方介入， 通□使用□称加密算法 (例如 Blowfish、 3DES)。
- *Authentication Header (AH)*, 保□ IP 包□不被第三方介入和□造， 通□□算校□和以及□ IP 包□的字段□行安全散列来□□。 随后是一个包含了散列□的附加□， 以便能□□□包。

ESP 和 AH 可以根据□境的不同， 分□或者一同使用。

IPsec 既可以用来直接加密主机之□的网□通□ (也就是 □□模式)； 也可以用来在□个子网之□建造 "虚□隧道"

用于□个网□之□的安全通□ (也就是 隧道模式)。 后一□更多的被称□是 虚□□用网 *(VPN)*。 ipsec(4) □机手册提供了□于 FreeBSD 中 IPsec 子系□的□□信息。

要把 IPsec 支持放□内核， □□在配置文件中加入下面的□□：

```
options   IPSEC        IP security
device    crypto
```

如果需要 IPsec 的□□支持， □□□加：

```
options   IPSEC_DEBUG  debug for IP security
```

15.9.2. □□

由于□如何建立 VPN 并不存在□准， 因此 VPN 可以采用□多□不同的技□来□□， □□技□都有其□□和弱点。 □篇文章将展□一个具体的□用情景， 并□它□□了□合的 VPN。

15.9.3. 情景： □个网□，一个家庭的网□和一个公司的网□。 都接入了 Internet，并且通□□条 VPN 就像在同一个网□一□。

□有条件如下：

- 至少有□个不同的站点
- □个站点都使用内部的 IP
- □个站点都通□□行 FreeBSD 的网□接入 Internet。
- □个网□上的网□至少有一个公网的 IP 地址。
- 网□的内部地址可以是公网或私有的 IP 地址， □并不是□□。它□并不冲突，比如它□不同□使用 `192.168.1.x` □□的地址。

15.9.4. 在 FreeBSD 上配置 IPsec

□始需先从 Ports Collection 安装 security/ipsec-tools。 □个第三方□件提供了一些能□□助配置的□用程序。

下一□是□建□个 gif(4) □□□用来在□个网□□□□数据包的 "隧道"。 使用 `root` 身□□行以下命令， 并用真□的内部外部网□替□命令中的 *internal* 和 *external* □：

```
# ifconfig gif0 create
```

```
# ifconfig gif0 internal1 internal2
```

```
# ifconfig gif0 tunnel external1 external2
```

比如，公司 LAN □外的 IP 地址是 172.16.5.4， 内部的 IP 地址□ 10.246.38.1。 家庭 LAN □外的 IP 地址是 192.168.1.12， 内部的 IP 地址□ 10.0.0.5。

□看起来可能有些混乱，所以我□通□ ifconfig(8) 命令□出再回□一下：

```
Gateway 1:

gif0: flags=8051 mtu 1280
tunnel inet 172.16.5.4 --> 192.168.1.12
inet6 fe80::2e0:81ff:fe02:5881%gif0 prefixlen 64 scopeid 0x6
inet 10.246.38.1 --> 10.0.0.5 netmask 0xffffff00

Gateway 2:

gif0: flags=8051 mtu 1280
tunnel inet 192.168.1.12 --> 172.16.5.4
inet 10.0.0.5 --> 10.246.38.1 netmask 0xffffff00
inet6 fe80::250:bfff:fe3a:c1f%gif0 prefixlen 64 scopeid 0x4
```

一旦完成以后，□个私有的 IP 地址都□□能像下面 ping(8) 命令□出那□互相□□。

```
priv-net# ping 10.0.0.5
PING 10.0.0.5 (10.0.0.5): 56 data bytes
64 bytes from 10.0.0.5: icmp_seq=0 ttl=64 time=42.786 ms
64 bytes from 10.0.0.5: icmp_seq=1 ttl=64 time=19.255 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=64 time=20.440 ms
64 bytes from 10.0.0.5: icmp_seq=3 ttl=64 time=21.036 ms
--- 10.0.0.5 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 19.255/25.879/42.786/9.782 ms

corp-net# ping 10.246.38.1
PING 10.246.38.1 (10.246.38.1): 56 data bytes
64 bytes from 10.246.38.1: icmp_seq=0 ttl=64 time=28.106 ms
64 bytes from 10.246.38.1: icmp_seq=1 ttl=64 time=42.917 ms
64 bytes from 10.246.38.1: icmp_seq=2 ttl=64 time=127.525 ms
64 bytes from 10.246.38.1: icmp_seq=3 ttl=64 time=119.896 ms
64 bytes from 10.246.38.1: icmp_seq=4 ttl=64 time=154.524 ms
--- 10.246.38.1 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 28.106/94.594/154.524/49.814 ms
```

正如□期的那□，□□都有从私有地址□送和接受 ICMP 数据包的能力。下面， □个网□都必□配置路由□□以正□□□□□的网□流量。 下面的命令可以□□□个：

```
# corp-net# route add 10.0.0.0 10.0.0.5 255.255.255.0
```

```
# corp-net# route add net 10.0.0.0: gateway 10.0.0.5
```

```
# priv-net# route add 10.246.38.0 10.246.38.1 255.255.255.0
```

```
# priv-net# route add host 10.246.38.0: gateway 10.246.38.1
```

此刻，不□从网□□是网□后的机器都能□□内部的网□。 □很容易通□以下的例子□□：

```
corp-net# ping 10.0.0.8
PING 10.0.0.8 (10.0.0.8): 56 data bytes
64 bytes from 10.0.0.8: icmp_seq=0 ttl=63 time=92.391 ms
64 bytes from 10.0.0.8: icmp_seq=1 ttl=63 time=21.870 ms
64 bytes from 10.0.0.8: icmp_seq=2 ttl=63 time=198.022 ms
64 bytes from 10.0.0.8: icmp_seq=3 ttl=63 time=22.241 ms
64 bytes from 10.0.0.8: icmp_seq=4 ttl=63 time=174.705 ms
--- 10.0.0.8 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 21.870/101.846/198.022/74.001 ms

priv-net# ping 10.246.38.107
PING 10.246.38.1 (10.246.38.107): 56 data bytes
64 bytes from 10.246.38.107: icmp_seq=0 ttl=64 time=53.491 ms
64 bytes from 10.246.38.107: icmp_seq=1 ttl=64 time=23.395 ms
64 bytes from 10.246.38.107: icmp_seq=2 ttl=64 time=23.865 ms
64 bytes from 10.246.38.107: icmp_seq=3 ttl=64 time=21.145 ms
64 bytes from 10.246.38.107: icmp_seq=4 ttl=64 time=36.708 ms
--- 10.246.38.107 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 21.145/31.721/53.491/12.179 ms
```

配置 "隧道" 是比□容易的部分。 配置一条安全□接□是个更加深入的□程。 下面的配置是使用 pre-shared（PSK） RSA 密□。除了 IP 地址外，□□的 /usr/local/etc/racoon/racoon.conf 也几乎相同。

```
path    pre_shared_key  "/usr/local/etc/racoon/psk.txt"; #location of pre-shared key
file
log     debug;  #log verbosity setting: set to 'notify' when testing and debugging is
complete

padding # options are not to be changed
{
        maximum_length  20;
        randomize       off;
        strict_check    off;
        exclusive_tail  off;
}
```

```
timer   # timing options. change as needed
{
        counter         5;
        interval        20 sec;
        persend         1;
#       natt_keepalive  15 sec;
        phase1          30 sec;
        phase2          15 sec;
}

listen  # address [port] that racoon will listening on
{
        isakmp          172.16.5.4 [500];
        isakmp_natt     172.16.5.4 [4500];
}

remote  192.168.1.12 [500]
{
        exchange_mode   main,aggressive;
        doi             ipsec_doi;
        situation       identity_only;
        my_identifier   address 172.16.5.4;
        peers_identifier        address 192.168.1.12;
        lifetime        time 8 hour;
        passive         off;
        proposal_check  obey;
#       nat_traversal   off;
        generate_policy off;

                        proposal {
                                encryption_algorithm    blowfish;
                                hash_algorithm          md5;
                                authentication_method   pre_shared_key;
                                lifetime time           30 sec;
                                dh_group                1;
                        }
}

sainfo  (address 10.246.38.0/24 any address 10.0.0.0/24 any)    # address
$network/$netmask $type address $network/$netmask $type ( $type being any or esp)
{                               # $network must be the two internal networks you are
joining.
        pfs_group       1;
        lifetime        time    36000 sec;
        encryption_algorithm    blowfish,3des,des;
        authentication_algorithm        hmac_md5,hmac_sha1;
        compression_algorithm   deflate;
}
```

解□所有可用的□□， □同□些例子里列出的都超越了□□文□的□□。 在 racoon 配置手册□中有着□富的相□信息。

SPD 策略也需要配置一下， □□ FreeBSD 和 racoon 就能□加密和解密主机□的网□流量了。

□可以通□在公司的网□上□行一个□似下面□□的 shell 脚本□□。保存到 /usr/local/etc/racoon/setkey.conf，□个文件会被在系□初始化的□候用到。

```
flush;
spdflush;
# To the home network
spdadd 10.246.38.0/24 10.0.0.0/24 any -P out ipsec esp/tunnel/172.16.5.4-
192.168.1.12/use;
spdadd 10.0.0.0/24 10.246.38.0/24 any -P in ipsec esp/tunnel/192.168.1.12-
172.16.5.4/use;
```

一旦完成后，便使用下面的命令在□□的网□上都□□ racoon：

```
# /usr/local/sbin/racoon -F -f /usr/local/etc/racoon/racoon.conf -l
/var/log/racoon.log
```

□出将会□似□□的：

```
corp-net# /usr/local/sbin/racoon -F -f /usr/local/etc/racoon/racoon.conf
Foreground mode.
2006-01-30 01:35:47: INFO: begin Identity Protection mode.
2006-01-30 01:35:48: INFO: received Vendor ID: KAME/racoon
2006-01-30 01:35:55: INFO: received Vendor ID: KAME/racoon
2006-01-30 01:36:04: INFO: ISAKMP-SA established 172.16.5.4[500]-192.168.1.12[500]
spi:623b9b3bd2492452:7deab82d54ff704a
2006-01-30 01:36:05: INFO: initiate new phase 2 negotiation:
172.16.5.4[0]192.168.1.12[0]
2006-01-30 01:36:09: INFO: IPsec-SA established: ESP/Tunnel 192.168.1.12[0]-
>172.16.5.4[0] spi=28496098(0x1b2d0e2)
2006-01-30 01:36:09: INFO: IPsec-SA established: ESP/Tunnel 172.16.5.4[0]-
>192.168.1.12[0] spi=47784998(0x2d92426)
2006-01-30 01:36:13: INFO: respond new phase 2 negotiation:
172.16.5.4[0]192.168.1.12[0]
2006-01-30 01:36:18: INFO: IPsec-SA established: ESP/Tunnel 192.168.1.12[0]-
>172.16.5.4[0] spi=124397467(0x76a279b)
2006-01-30 01:36:18: INFO: IPsec-SA established: ESP/Tunnel 172.16.5.4[0]-
>192.168.1.12[0] spi=175852902(0xa7b4d66)
```

□□一下 "隧道" 能正常工作， 切□到□外一个控制台用如下的 tcpdump(1) 命令□看网□流量。根据需要替□掉下面的 em0 网□界面。

```
# tcpdump -i em0 host 172.16.5.4 and dst 192.168.1.12
```

控制台上能看到如下□似的□出。如果不是□□的□， 可能就有些□□了，□□的□需要用到返回的数据。

```
01:47:32.021683 IP corporatenetwork.com > 192.168.1.12.privatenetwork.com:
ESP(spi=0x02acbf9f,seq=0xa)
01:47:33.022442 IP corporatenetwork.com > 192.168.1.12.privatenetwork.com:
ESP(spi=0x02acbf9f,seq=0xb)
01:47:34.024218 IP corporatenetwork.com > 192.168.1.12.privatenetwork.com:
ESP(spi=0x02acbf9f,seq=0xc)
```

此刻，□个网□就好像是同一个网□的一部分一□。 而且□□个网□很可能也□□有防火□的保□。 要使得□□个网□能互相□□，就需要添加一些□出包的□□。 就 ipfw(8) 来□，加入下面的几行□配置文件：

```
ipfw add 00201 allow log esp from any to any
ipfw add 00202 allow log ah from any to any
ipfw add 00203 allow log ipencap from any to any
ipfw add 00204 allow log udp from any 500 to any
```

□□号可能需要根据□有机器上的配置做相□的修改。

□于 pf(4) 或者 ipf(8) 的用□， 下面的几行□□□□可行：

```
pass in quick proto esp from any to any
pass in quick proto ah from any to any
pass in quick proto ipencap from any to any
pass in quick proto udp from any port = 500 to any port = 500
pass in quick on gif0 from any to any
pass out quick proto esp from any to any
pass out quick proto ah from any to any
pass out quick proto ipencap from any to any
pass out quick proto udp from any port = 500 to any port = 500
pass out quick on gif0 from any to any
```

最后，要允□机器初始化的□候□始 VPN 支持，在 /etc/rc.conf 中加入以下的几行：

```
ipsec_enable="YES"
ipsec_program="/usr/local/sbin/setkey"
ipsec_file="/usr/local/etc/racoon/setkey.conf" # allows setting up spd policies on
boot
racoon_enable="yes"
```

15.10. OpenSSH

OpenSSH 是一□用于安全地□□□程□算机的□接工具。 它可以作□ `rlogin`、 `rsh` `rcp` 以及 `telnet` 的直接替代品使用。 更□一□， 其他任何 TCP/IP □接都可以通□ SSH 安全地□行隧道/□□。 OpenSSH □所有的□□□行加密， 从而有效地阻止了窃听、 □接劫持， 以及其他网□□的攻□。

OpenSSH 由 OpenBSD project □□， 它基于 SSH v1.2.12 并包含了最新的□□修□和更新。 它同□兼容 SSH □□的 1 和 2 □个版本。

15.10.1. 使用 OpenSSH 的好□

一般□来， 在使用 telnet(1) 或 rlogin(1) □， 数据是以未□加密的明文的形式□送的。 □□一来， 在客□机和服□器之□的网□上□行的听包程序， 便可以在会□中窃取到□□的用□名/密□和数据。 OpenSSH 提供了多□的身□□□和加密方法来防止□□情况的□生。

15.10.2. □用 sshd

sshd 的□用是作□ FreeBSD 安装中 Standard 安装□程中的一□来□行的。 要□看 sshd 是否已被□用， □□□ rc.conf 文件中的：

```
sshd_enable="YES"
```

□表示在下次系□□□□加□ OpenSSH 的服□程序 sshd(8)。 此外， 也可以手□使用 rc(8) 脚本 /etc/rc.d/sshd 来□□ OpenSSH：

```
/etc/rc.d/sshd start
```

15.10.3. SSH 客□

ssh(1) 的工作方式和 rlogin(1) 非常□似。

```
# ssh user@example.com
Host key not found from the list of known hosts.
Are you sure you want to continue connecting (yes/no)? yes
Host 'example.com' added to the list of known hosts.
user@example.com's password: *******
```

登□□程和使用 rlogin 或 telnet 建立的会□非常□似。 在□接□， SSH 会利用一个密□指□系□来□□服□器的真□性。 只有在第一次□接□， 用□会被要求□入 yes。 之后的□接将会□□□先保存下来的密□指□。 如果保存的指□与登□□接收到的不符， □将会□出警告。 指□保存在 ~/.ssh/known_hosts 中， □于 SSH v2 指□， □是 ~/.ssh/known_hosts2。

默□情况下， □新版本的 OpenSSH 只接受 SSH v2 □接。 如果能用版本 2 □客□程序会自□使用， 否□它会返回使用版本 1 的模式。 此外， 也可以通□命令行参数 -1 或 -2 来相□地□制使用版本 1 或 2。 保持客□端的版本 1 能力是□了考□□早版本的兼容性。

15.10.4. 安全□制

scp(1) 命令和 rcp(1); 的用法□似， 它用于将文件□制到□程的机器上， 或□制□来， 区□是它是安全的。

```
#  scp user@example.com:/COPYRIGHT COPYRIGHT
user@example.com's password: *******
COPYRIGHT            100% |*****************************|  4735
00:00
#
```

由于先前的例子中已□保存了指□， 使用 scp(1) □会自□地加以□□。

scp(1) 使用的参数同 cp(1) □似。 第一个参数是一个或一□文件， 然后是□制的目□。 由于文件是通□ SSH 在网上□□的， 因此某些文件的名字需要写成 用□名@主机名:<□程文件路径>。

15.10.5. 配置

□□ OpenSSH 服□程序和客□端的系□□配置文件在 /etc/ssh 目□中。

ssh_config 用于配置客□端的□定， 而 sshd_config □用于配置服□器端。

□外 sshd_program (默□是 /usr/sbin/sshd)， 以及 sshd_flags □□个 rc.conf □□提供了更多的配置□□。

15.10.6. ssh-keygen

用于取代口令的一□方法是使用 ssh-keygen(1) 来生成 DSA 或 RSA 密□□用于□□用□的身□：

```
% ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_dsa):
Created directory '/home/user/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user/.ssh/id_dsa.
Your public key has been saved in /home/user/.ssh/id_dsa.pub.
The key fingerprint is:
bb:48:db:f2:93:57:80:b6:aa:bc:f5:d5:ba:8f:79:17 user@host.example.com
```

ssh-keygen(1) 会生成一个包含公私□□用于□□身□。 私□将保存到 ~/.ssh/id_dsa 或 ~/.ssh/id_rsa， 而公□□被存放到 ~/.ssh/id_dsa.pub 或 ~/.ssh/id_rsa.pub， 文件名取决于□□□的 DSA 和 RSA 密□□型。 RSA 或者 DSA 公□必□被存放到□程机器上的 ~/.ssh/authorized_keys 才能□使系□正□□□。

□将允□从□程□接□以基于 SSH 密□的□□来代替口令□□。

如果在 ssh-keygen(1) 中使用了通行字， □□次使用私□□都需要□入它。 ssh-agent(1) 能□□解多次□入□通行字的□力， 并将在接下来的 ssh-agent 和 ssh-add 予以□述。

□□和配置文件可能随 OpenSSH 的版本不同而不同； □了避免出□□， □□参考 ssh-keygen(1) □机手册。

□将使到□程机器的□接基于 SSH 密□而不是口令。

如果在□行 ssh-keygen(1) □使用了通行字， □次使用私□的□候用□都将被要求□入通行字。 ssh-agent(1) 能□□□重□□入□□通行字的□担， 有□更□□的探究在 ssh-agent 和 ssh-add 下一□ .

随着□系□上的 OpenSSH 版本的不同，各□□□和配置文件也会不同； □了避免此□□□，□需要参□ ssh-keygen(1) □机手册。

15.10.7. ssh-agent 和 ssh-add

ssh-agent(1) 和 ssh-add(1) □□个工具， 提供了一□将 SSH 秘□加□到内存中以便使用， 而不必□次都□入通行字的方法。

ssh-agent(1) 工具能□使用加□到其中的私□来□理□□□程。 ssh-agent(1) □被用于□□□一个□用程序。最基本的用法是， 使用它来□□ shell， 而高□一些的用法□是用它来□□□口管理器。

要在 shell 中使用 ssh-agent(1)， 首先□把 shell 作□参数来□□它。 随后， □通□ ssh-add(1) 并□入通行字，来向它提供身□□□信息。 一旦□些□□都做完了， 用□就□□能□ ssh(1) 到任何一个安装了□□公□的机器了。例如：

```
% ssh-agent csh
% ssh-add
Enter passphrase for /home/user/.ssh/id_dsa:
Identity added: /home/user/.ssh/id_dsa (/home/user/.ssh/id_dsa)
%
```

要在 X11 中使用 ssh-agent(1)， □用 ssh-agent(1) 的□程□置于 ~/.xinitrc 之中。 □将把 ssh-agent(1) 服□提供□所有在 X11 中□行的程序。 下面是一个 ~/.xinitrc 文件的□例：

```
exec ssh-agent startxfce4
```

□将□□ ssh-agent(1)， 而后者将在□次 X11 □□□□行 XFCE。 作完□些之后就可以重□ X11 以便使修改生效。随后□就可以□行 ssh-add(1) 来加□全部 SSH 密□了。

15.10.8. SSH 隧道

OpenSSH 能□□建隧道以便用加密的会□来封装其他□□。

下面的命令告□ ssh(1) □ telnet □建一个隧道：

```
% ssh -2 -N -f -L 5023:localhost:23 user@foo.example.com
%
```

上述 ssh 命令使用了下面□些□□：

-2

□制 ssh 使用第2版的□□ (如果需要和□老的 SSH 一同工作□不要使用□个□□)。

`-N`

表示不使用命令行， 或者只使用隧道。 如果省略， `ssh` 将同初始化会。

`-f`

制 `ssh` 在后台行。

`-L`

表示生一条 本地端口:程主机:程端口 形式的隧道。

`user@foo.example.com`

程 SSH 服器。

SSH 隧道通听 `localhost` 上面指定端口来完成工作。 它将把本机主机/端口上接收到的接通 SSH 接到程主机/端口。

本例中， 位于 `localhost` 的 *5023* 端口 被用于 `localhost` 的接到程主机的 *23* 端口。 由于 *23* 是 telnet 使用的， 因此它将通 SSH 隧道完成 telnet 会。

可以用来封装任意不安全的 TCP , 例如 SMTP、 POP3、 FTP等等。

例 *23.* 使用 *SSH* *SMTP* 建安全隧道

```
% ssh -2 -N -f -L 5025:localhost:25 user@mailserver.example.com
user@mailserver.example.com's password: *****
% telnet localhost 5025
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mailserver.example.com ESMTP
```

可以与 ssh-keygen(1) 以及外的用号配合来建立一个更透明的 SSH 隧道境。 密可以被用在需要入口令的地方， 而且可以不同的用配置不同的隧道。

15.10.8.1. 用的 SSH 通道例子

15.10.8.1.1. 加 POP3 服的安全

工作， 有一个允外来接的 SSH 服器。 同一个公网中有一个件服器提供 POP3 服。 个网， 或从家到公室的网可能不， 或不完全可信。 基于的原因， 需要以安全的方式来看件。 解决方法是建一个到公室 SSH 服器的接， 并通个接来 POP3 服：

```
% ssh -2 -N -f -L 2110:mail.example.com:110 user@ssh-server.example.com
user@ssh-server.example.com's password: ******
```

当个通道上， 可以把 POP3 求到 `localhost` 端口 2110。 个接将通通道安全地到 `mail.example.com`。

15.10.8.1.2. □□□□的防火□

一些大□□包的网□管理□会使用一些□端的防火□策略， 不□□□□入的□接， 而且也□□□出的□接。 一些□候□可能只能□接□程机器 22 端口，以及 80 端口用来□行 SSH 和网□□□。

□可能希望□□一些其它的（也□与工作无□的）服□， 例如提供音□的 Ogg Vorbis 流媒体服□器。 如果 Ogg Vorbis server 在 22 或 80 端口以外的端口播放音□， □□将无法□□它。

解决方法是建立一个到□的网□的防火□之外的网□上的 SSH 服□器， 并通□它提供的通道□接到 Ogg Vorbis 服□器上。

```
% ssh -2 -N -f -L 8888:music.example.com:8000 user@unfirewalled-system.example.org
user@unfirewalled-system.example.org's password: *******
```

□在□可以把客□程序指定到 localhost 的 8888 端口， 它将把□求□□□ music.example.com 的 8000 端口，从而□□防火□。

15.10.9. 允□用□登□ AllowUsers □□

通常限制□些用□能□登□， 以及从何□登□会是好主意。 采用 AllowUsers □□能□方便地□到□一目的。 例如，想要只允□ root 用□从 192.168.1.32 登□， 就可以在 /etc/ssh/sshd_config 文件中加入下述□置：

```
AllowUsers root@192.168.1.32
```

要允□用□ admin 从任何地方登□， □只需列出用□名：

```
AllowUsers admin
```

可以在同一行指定多个用□， 例如：

```
AllowUsers root@192.168.1.32 admin
```

列出需要登□机器的用□很重要； 否□他□将被□在外面。

在完成□ /etc/ssh/sshd_config 的修改之后□必□告□ sshd(8) 重新加□其配置文件， 方法是□行：

```
# /etc/rc.d/sshd reload
```

15.10.10. □一□的□料

OpenSSH

ssh(1) scp(1) ssh-keygen(1) ssh-agent(1) ssh-add(1) ssh_config(5)

sshd(8) sftp-server(8) sshd_config(5)

15.11. 文件系□□□控制表

与文件系□在其他方面的加□， 如快照等一道， FreeBSD 提供了通□文件系□□□控制表 (ACL) □□的安全机制。

□□控制表以高度兼容 (POSIX®.1e) 的方式□展了□准的 UNIX® □限模型。 □一特性使得管理□能□利用其□□□□更□□□的安全模型。

如果想□ UFS 文件系□□用 ACL 支持， □需要添加下列□□：

```
options UFS_ACL
```

并重新□□内核。 如果没有将□个□□□□□内核， □在挂接支持 ACL 的文件系□□将会收到警告。 □个□□在 GENERIC 内核中已□包含了。 ACL 依□于在文件系□上□用□展属性。 在新一代的 UNIX® 文件系□， UFS2 中内建了□□支持。

在 UFS1 上配置□展属性需要比 UFS2 更多的管理□□。 而且， 在 UFS2 上的□展属性的性能也有□大的提高。 因此， 如果想要使用□□控制表， 推□使用 UFS2 而不是 UFS1。

ACL 可以在挂接□通□□□ `acls` 来□□， 它可以加入 /etc/fstab。 □外， 也可以通□使用 tunefs(8) 修改超□□中的 ACL □□来持久性地□置自□的挂接属性。 一般而言， 后一□方法是推□的做法， 其原因是：

- 挂接□的 ACL □□无法被重挂接 (mount(8) `-u`) 改□， 只有完整地 umount(8) 并做一次新的 mount(8) 才能改□它。 □意味着 ACL 状□在系□□□之后就不可能在 root 文件系□上□生□化了。 □外也没有□法改□正在使用的文件系□的□个状□。
- 在超□□中的□置将使得文件系□□被以□用 ACL 的方式挂接， 即使在 fstab 中的□□□目没有作□置， 或□□□序□生□化□也是如此。 □避免了不慎将文件系□以没有□用 ACL 的状□挂接， 从而避免没有□制 ACL □□的安全□□。

可以修改 ACL 行□， 以允□在没有□行一次全新的 mount(8) 的情况下□用它， 但我□□□，不鼓励在未□用 ACL □□□做是有必要的， 因□如果□用了 ACL， 然后□掉它，然后在没有刷新□展属性的情况下重新□用它是很容易造成□□的。 一般而言， 一旦□用了文件系□的 ACL 就不□□再□掉它， 因□此□的文件系□的保□措施可能和用□所期待的□子不再兼容， 而重新□用 ACL 将重新把先前的 ACL 附着到文件上， 而由于它□的□限□生了□化， 就很可能造成无法□期的行□。

在□看目□□， □用了 ACL 的文件将在通常的属性后面□示 + (加号)。 例如：

```
drwx------  2 robert  robert  512 Dec 27 11:54 private
drwxrwx---+ 2 robert  robert  512 Dec 23 10:57 directory1
drwxrwx---+ 2 robert  robert  512 Dec 22 10:20 directory2
drwxrwx---+ 2 robert  robert  512 Dec 27 11:57 directory3
drwxr-xr-x  2 robert  robert  512 Nov 10 11:54 public_html
```

□里我□看到了 directory1、 directory2， 以及 directory3 目□使用了 ACL。 而 public_html □没有。

15.11.1. 使用 ACL

文件系□ ACL 可以使用 getfacl(1) 工具来□看。 例如， 如果想□看 test 的 ACL □置， 所用的命令是：

```
% getfacl test
    #file:
    #owner:1001
    #group:1001
    user::rw-
    group::r--
    other::r--
```

要修改□个文件上的 ACL □置， □需要使用 setfacl(1) 工具。 例如：

```
% setfacl -k test
```

`-k` 参数将把所有当前定□的 ACL 从文件或文件系□中□除。 一般来□□□使用 `-b` 因□它会保持□ ACL 正常工作的那些□不□。

```
% setfacl -m u:trhodes:rwx,group:web:r--,o::--- test
```

在前面的命令中， `-m` □□被用来修改默□的 ACL □。由于已□被先前的命令 □除，因此没有□先定□的□，于是默□的□□被恢□，并附加上指定的□□。 □小心地□□，如果□加入了一个不存在的用□或□，那□将会在 stdout 得到一条 `Invalid argument` 的□□提示。

15.12. □□第三方安全□□

□去几年中， 安全□域在如何□理漏洞的□估方面取得了□足的□□。 几乎□一个操作系□都越来越多地安装和配置了第三方工具， 而系□被入侵的威□也随之□加。

漏洞的□估是安全的一个□□因素， 尽管 FreeBSD 会□布基本系□的安全公告， 然而□□一个第三方工具都□布安全公告□超出了 FreeBSD Project 的能力。 在□一前提下， 一□□□第三方漏洞的威□， 并警告管理□存在已知的安全□□的方法也就□□而生。 名□ Portaudit 的 FreeBSD 附加工具能□□助□□成□一目的。

ports-mgmt/portaudit port 会下□一个数据□， □一数据□是由 FreeBSD Security Team 和 ports □□人□□□的， 其中包含了已知的安全□□。

要□始使用 Portaudit， 需要首先从 Ports Collection 安装它：

```
# cd /usr/ports/ports-mgmt/portaudit && make install clean
```

在安装□程中， periodic(8) 的配置文件将被修改， 以便□ Portaudit 能□在□天的安全□□□程中□行。 一定要保□□到 `root` □号的□日安全□□□件□□有人在□。 除此之外不需要□行更多的配置了。

安装完成之后， 管理□可以通□下面的命令来更新数据□， 并□看目前安装的□件包中所存在的已知安全漏洞：

```
# portaudit -Fda
```

由于□天□行 periodic(8) □都会自□更新数据□， 因此， □行□条命令是可□的。 在□里只是作□例子□出。

在任何□候， 如果希望□通□ Ports Collection 安装的第三方□件工具□行□□， 管理□都可以使用下面的命令：

```
# portaudit -a
```

□□存在漏洞的□件包， Portaudit 将生成□似下面的□出：

```
Affected package: cups-base-1.1.22.0_1
Type of problem: cups-base -- HPGL buffer overflow vulnerability.
Reference: <http://www.FreeBSD.org/ports/portaudit/40a3bca2-6809-11d9-a9e7-
0001020eed82.html>

1 problem(s) in your installed packages found.

You are advised to update or deinstall the affected package(s) immediately.
```

通□□□上面□出的 URL， 管理□能□了解□于那个漏洞的□一□信息。 □些信息通常包括受到影□的 FreeBSD Port 版本， 以及其他可能包含安全公告的网站。

□而言之， Portaudit 是一个□大的工具， 并能□配合 Portupgrade port 来非常有效地工作。

15.13. FreeBSD 安全公告

像其它具有□品□品□的操作系□一□， FreeBSD 会□布 "安全公告"。 通常□□公告会只有在相□的□行版本已□正□地打□□丁之后□到安全□件列表并在勘□中□明。 本□将介□什□是安全公告， 如何理解它， 以及□系□打□丁的具体□□。

15.13.1. 安全公告看上去是什□□子？

FreeBSD 安全公告的□式□似下面的□例， □一例子来自 FreeBSD 安全□□通知□件列表 □件列表。

```
=============================================================================
FreeBSD-SA-XX:XX.UTIL                                          Security Advisory
                                                            The FreeBSD Project

Topic:          denial of service due to some problem①

Category:       core②
Module:         sys③
Announced:      2003-09-23④
Credits:        Person⑤
Affects:        All releases of FreeBSD⑥
                FreeBSD 4-STABLE prior to the correction date
Corrected:      2003-09-23 16:42:59 UTC (RELENG_4, 4.9-PRERELEASE)
                2003-09-23 20:08:42 UTC (RELENG_5_1, 5.1-RELEASE-p6)
                2003-09-23 20:07:06 UTC (RELENG_5_0, 5.0-RELEASE-p15)
                2003-09-23 16:44:58 UTC (RELENG_4_8, 4.8-RELEASE-p8)
                2003-09-23 16:47:34 UTC (RELENG_4_7, 4.7-RELEASE-p18)
                2003-09-23 16:49:46 UTC (RELENG_4_6, 4.6-RELEASE-p21)
                2003-09-23 16:51:24 UTC (RELENG_4_5, 4.5-RELEASE-p33)
                2003-09-23 16:52:45 UTC (RELENG_4_4, 4.4-RELEASE-p43)
                2003-09-23 16:54:39 UTC (RELENG_4_3, 4.3-RELEASE-p39)⑦
CVE Name:       CVE-XXXX-XXXX⑧

For general information regarding FreeBSD Security Advisories,
including descriptions of the fields above, security branches, and the
following sections, please visit
http://www.FreeBSD.org/security/.

I.   Background⑨

II.  Problem Description⑩

III. Impact⑪

IV.  Workaround⑫

V.   Solution⑬

VI.  Correction details⑭

VII. References⑮
```

① Topic(□□) 一□□明了□□到底是什□。 它基本上是□所□□的安全□□及其所□及的工具的描述。

② Category (分□) 是指系□中受到影□的□件， □一□可能是 core、 contrib， 或者 ports 之一。 core 分□表示安全弱点影□到了 FreeBSD 操作系□的某个核心□件。 contrib 分□表示弱点存在于某个捐□□ FreeBSD Project 的□件， 例如 sendmail。 最后是 ports， 它表示□弱点影□了 Ports Collection 中的某个第三方□件。

③ Module(模□) 一□□出了□件的具体位置， 例如 sys。 在□个例子中， 可以看到 sys 模□是存在□□的； 因此， □个漏洞会影□某个在内核中的□件。

④ Announced(□布□□) 一□反映了与安全公告有□的数据是什□□候公之于□的。 □□明安全□□已□□□□□□□存在， 而□丁已□写入了 FreeBSD 的代□□。

⑤ Credits(作者) 一□□出了注意到□□存在并□告它的个人或□体。

⑥ The Affects(影□□□) 一□□出了 FreeBSD 的□些版本存在□个漏洞。 □于内核来□， □□受影□的文件上□行的 ident □出可以□助□□文件版本。 □于 ports， 版本号在 /var/db/pkg 里面的 port 的名字后面列出。 如果系□没有与 FreeBSD CVS 代□□同□并□日□建， 它很可能是有□□的。

⑦ Corrected(修正□□) 一□□出了□行版本中修正□□的具体日期、□□和□差。 在公共漏洞数据□ (Common Vulnerabilities Database) 系□中□留的， 用于□看漏洞的□□信息。

⑧ Background(技□背景) 一□提供了受影□的□件的作用。 多数□候□一部分会□明□什□ FreeBSD 中包含了它， 它的作用， 以及它的一些原理。

⑨ Problem Description(□□描述) 一□深入□述安全漏洞的技□□□。 □部分有□会包括有□□的代□相□的□□情况， 甚至是□个部件如何能□被□意利用并打□漏洞的□□。

⑩ Impact(影□) 一□描述了□□能□造成的影□□型。 例如， 可能□致拒□服□攻□， □限提升， 甚至□致得到超□用□的□限。

⑪ Workaround(□急方案) 一□□出了系□管理□在□□无法升□系□□可以采取的□□性□策。 □些原因可能包括□□限制， 网□□源的限制， 或其它因素。 不□无□如何， 安全不能□被□□， 有□□的系□要□□□打□丁， 要□□□□施□□□急方案。

⑫ Solution(解决方案) 一□提供了如何□有□□的系□打□丁的方法。 □是□□逐□□□和□□□的□系□打□丁并□其安全地工作的方法。

⑬ =Correction Details(修正□□) 一□展示了□□ CVS 分支或某个□行版的修正特征。 同□也提供了□个分支上相□文件的版本号。

⑭ References(文献) 一□通常会□出其它信息的来源。 □可能包括 URL， □籍、 □件列表以及新□□。

15.14. □程□□

□程□□是一□管理□可以使用的跟踪系□□源使用情况的手段， 包括它□分配□了□些用□、 提供系□□□手段， 并且可以精□到用□□行的□一个命令。

当然， □□做法是兼有利弊的。 它的好□是， □□入侵□可以迅速把□□□小到攻□者□入的□刻； 而□□做的缺点， □是□□会□生大量的日志， 因而需要很多磁□空□来存□它□。 □一□将□□管理□一□一□地配置基本的□程□□。

15.14.1. □用并利用□程□□

在使用□程□□之前， 必□先□用它。 要完成□□工作， 需要□行下面的命令：

```
# touch /var/account/acct

# accton /var/account/acct

# echo 'accounting_enable="YES"' >> /etc/rc.conf
```

一旦□用之后， □□就会□始跟踪 CPU □□数据、 命令， 等等。 所有的□□日志不是以可□的方式□□的， 要□看它□， 需要使用 sa(8) □个工具。 如果没有□出其他参数， □ sa 将按用□， 以分□□□位□示他□所使用的□□、 □共的

CPU 和用□□□， 以及平均的 I/O 操作数目， 等等。

要□示□于□□□出的命令的相□信息， □□使用 lastcomm(1) 工具。 lastcomm 命令可以用来□示在某一 ttys(5) 上的用□信息， 例如：

```
# lastcomm ls
    trhodes ttyp1
```

将会□示出所有已知的 trhodes 在 ttyp1 □端上□行 ls 的情况。

更多的可用□□在□机手册 lastcomm(1)、 acct(5) 和 sa(8) 中有所介□。

Chapter 16. Jails

16.1. 概述

□一章将□□介□ FreeBSD jail 是什□， 以及如何使用它□。 Jail， 有□也被□□是□ *chroot* □境 的一□□□型替代品， □于管理□而言是非常□大的工具， 同□， 它的一些基本用法， □高□用□而言也相当有用。

□完□章， □将了解：

- jail 是什□， 以及它在□安装的 FreeBSD 中所能□□的作用。
- 如何□□、 □□和停止 jail。
- 如何从 jail 内部或主机上□行管理的一些基□知□。

其他一些能□□□提供□于 jail 的有用信息的地方□有：

- jail(8) □机手册。 □是□于 `jail` - 用于在 FreeBSD 中□□、 停止和控制 FreeBSD jails - 工具的完整□明□。
- □件列表及其存□。 由 FreeBSD □件列表服□器 提供的 FreeBSD 一般□□□件列表 和其他□件列表的存□， 已□包含了一系列□于 jails 的有价□的信息。 通常搜索存□或□□ freebsd-questions □件列表能□□□□来很多有用的信息。

16.2. 与 Jail 相□的一些□□

□了□助□更好地理解与 jail 有□的 FreeBSD 系□知□， 以及它□如何与 FreeBSD 的其它部分相互作用， □□理解下列□□：

chroot(8) (命令)

□个工具使用 FreeBSD 的系□□用 chroot(2) FreeBSD 来改□□程， 以及□程的所有衍生□程所能看到的根目□。

chroot(2) (□境)

在 "chroot" 中□行的□程□境。 □包括□似文件系□中的可□部分、 可用的用□及用□□ ID、 网□接口以及其他 IPC 机制等□源。

jail(8) (命令)

用以在 jail □境中□行□程的系□管理工具。

宿主 (系□、 □程、 用□等等)

能□控制 jail □境的系□。 宿主系□能□□□全部可用的硬件□源， 并能□控制 jail □境内外的□程。 宿主系□与 jail 的一□重要区□是， 在宿主系□中的超□用□□程， 并不像在 jail 中那□受到一系列限制。

hosted (系□、 □程、 用□等等)

可□□□源受 FreeBSD jail 限制的□程、 用□或其他□体。

16.3. 介□

由于系□管理是一□困□而又令人□解的任□， 因此人□□□了一系列□大的工具， 来□管理□的工作□得更加□□。

□些改□通常是□系□能□以更□□的方式安装、 配置， 并毫无□□地持□□□。 □其中， □多管理□希望能□□系□正□地□行安全方面的配置， 使其能□用于真正的用途， 而阻止安全方面的□□。

FreeBSD 系□提供的一□用于改善安全的工具就是 *jail*。 jail 是在 FreeBSD 4.X 中由 Poul-Henning Kamp <phk@FreeBSD.org> 引入的， 它在 FreeBSD 5.X 中又□行了一系列改□， 使得它成□了一个□大而□活的系□。 目前仍然在□其□行持□的□□， 以提高其可用性、 性能和安全性。

16.3.1. Jail 是什□

BSD-□的操作系□从 4.2BSD □始即提供了 chroot(8)。 chroot(2) 工具能□改□一□□程的根目□的位置， 从而建立一个与系□中其他部分相隔□的安全□境： 在 chroot □境中的□程， 将无法□□其外的文件或其他□源。 正是由于□□能力， 即使攻□者攻破了某一个□行于 chroot □境的服□， 也不能攻破整个系□。 chroot(8) □于那些不需要很多□活性或□□的高□功能的□□□用而言相当好用。 □外， 在引入 chroot 概念的□程中， 曾□□□□□多跳出 chroot □境的方法， 尽管□些□□在□新的 FreeBSD 版本中已□修正， 但很明□地， chroot(8) 并不是一□用于加固服□安全的理想解决方案。 因此， 必□□□一个新的子系□来解决□些□□。

□就是□什□要□□ *jail* 最主要的原因。

Jail 以多□方式改□了□□的 chroot(2) □境概念。 在□□的 chroot(2) □境中， 只限制了□程能□□□文件系□的□些部分。 其他部分的系□□源 (例如系□用□、 正在□行的□程， 以及网□子系□) 是由 chroot □程与宿主系□中的其他□程共享的。 jail □展了□个模型， 它不□将文件系□的□□虚□化， 而且□将用□、 FreeBSD 的网□子系□， 以及一些其他系□□源虚□化。 □于□些精□控制以及□整 jail □境□□能力的更具体的介□， 可参□ 微□和管理。

jail 具有以下四□特点：

- 目□子□ - □入 jail 的起点。 一旦□入了 jail， □程就不再被允□□□□□子□以外的□象。 □□上影□到最初 chroot(2) □□的安全□□不会影□ FreeBSD jail。
- 主机名 - 将用于 jail 的主机名。 jail 主要用于存放网□服□， 因此在□个 mail 上能□□注一个有意□的主机名， 能□在很大程度上□化系□管理□的工作。
- IP 地址 - □个地址是指定□ jail 的， 在 jail 的生命周期内都无法改□。 通常 jail 的 IP 地址是某一个网□接口上的□名地址， 但□并不是必需的。
- 命令 - 准□在 jail 中□行的可□行文件的完整路径名。 □个命令是相□于 jail □境的根目□的， 随 jail □境的□型不同， 可能会有很多不同之□。

除了□些之外， jail 也可以□有自己的用□和自己的 `root` 用□。 自然， □里的 `root` 用□的□力会受限于 jail □境， 并且， 从宿主系□的□点看来， jail `root` 用□并不是一个无所不能的用□。 此外， jail 中的 `root` 用□不能□行除了其□□ jail(8) □境之外的系□中的一些□□操作。 □于 `root` 用□的能力和限制， 在后面的 微□和管理 中将加以介□。

16.4. 建立和控制 jail

一些系□管理□喜□将 jail 分□□□： "完整的" jail， 通常包含真正的 FreeBSD 系□， 以及 "服□" jail， □用于□行一个可能使用特□的□用或服□。 □只是一□概念上的区分， 并不影□如何建立 jail 的□程。 在□机手册 jail(8) 中□如何□建 jail □行了清晰的□述：

```
# setenv D /here/is/the/jail
# mkdir -p $D ①
# cd /usr/src
# make buildworld ②
# make installworld DESTDIR=$D ③
# make distribution DESTDIR=$D ④
# mount -t devfs devfs $D/dev ⑤
```

① 第一□就是□ jail □□一个位置。 □个路径是在宿主系□中 jail 的物理位置。 一□常用的□□是 /usr/jail/jailname， 此□ *jailname* 是 jail 的主机名。 /usr/ 文件系□通常会有足□的空□来保存 jail 文件系□， □于 "完整" 的 jail 而言， 它通常包含了 FreeBSD 默□安装的基本系□中□个文件的副本。

② 如果□已□通□使用 `make world` 或者 `make buildworld` 重新□□□了□的 userland， □可以跳□□一□□并把□有的 userland 安装□新的 jail。

③ □个命令将在 jail 目□中安装所需的可□行文件、 函数□以及□机手册等。

④ `distribution` □个 make target 将安装全部配置文件， 或者□句□□， 就是将 /usr/src/etc/ □制到 jail □境中的 /etc： $D/etc/。

⑤ 在 jail 中不是必□要挂接 devfs(8) 文件系□。 而□一方面， 几乎所有的□用程序都会需要□□至少一个□□， □主要取决于□用程序的性□和目的。 控制 jail 中能□□□的□□非常重要， 因□不正□的配置， 很可能允□攻□者在 jail 中□行一些□意的操作。 通□ devfs(8) □施的控制， 可以通□由□机手册 devfs(8) 和 devfs.conf(5) 介□的□□集配置来□□。

一旦装好了 jail， 就可以使用 jail(8) 工具来安装它了。 jail(8) 工具需要四个必填参数， □些参数在 Jail 是什□ 中□行了介□。 除了□四个参数之外， □□可以指定一些其他参数， 例如， 以特定用□身□来在 jail 中□行程序等等。 □里， `command` 参数取决于□希望建立的 jail 的□型； □于 虚□系□， 可以□□ /etc/rc， 因□它会完成真正的 FreeBSD 系□□□所需的操作。 □于 服□ jail， □行的命令取决于将在 jail 中□行的□用程序。

Jail 通常□在系□□□□□□， 因此， FreeBSD rc 机制提供了一些很方便的机制来□化□些工作。

1. 在引□□需要□□的 jail 列表□写入 rc.conf(5) 文件：

```
jail_enable="YES"   # 如果□□ NO □表示不自□□□ jail
jail_list="www"     # 以空格分隔的 jail 名字列表
```

在 jail_list 中的名字中， 可以使用字母和数字， 而不□使用其他字符。

2. □于 jail_list 中列出的 jail， □□指定一系列□□的 rc.conf(5) □置， 用以描述具体的 jail：

```
jail_www_rootdir="/usr/jail/www"     # jail 的根目□
jail_www_hostname="www.example.org"   # jail 的主机名
jail_www_ip="192.168.0.10"          # jail 的 IP 地址
jail_www_devfs_enable="YES"          # 在 jail 中挂接 devfs
jail_www_devfs_ruleset="www_ruleset" # 在 jail 中□用的devfs □□集
```

默□情况下， 在 rc.conf(5) 中配置□□的 jail 会□行其中的 /etc/rc 脚本， 也就是□， 默□情况下将 jail 作□虚□系□方式来□□。 □于服□ jail， □□□外指定□□命令， 方法是□置□□的 jailjailnameexec_start 配置。

如欲了解全部可用的□□， □参□□机手册 rc.conf(5)。

/etc/rc.d/jail 脚本也可以用于手工□□或停止 rc.conf 中配置的 jail：

```
# /etc/rc.d/jail start www
# /etc/rc.d/jail stop www
```

目前， 尚没有一□方法来很干□地□□ jail(8)。 □是因□通常用于正常□□系□的命令， 目前尚不能在 jail 中使用。 目前， □□ jail 最好的方式， 是在 jail 外通□ jexec(8) 工具， 在 jail 中□行下列命令：

```
# sh /etc/rc.shutdown
```

更□一□的□□□明， □参□□机手册 jail(8)。

16.5. 微□和管理

□可以□ jail □置□多不同的□□， 并□ FreeBSD 宿主系□以不同的方式与 jail 交互， 以支持更高□□的□用。 □一□将介□：

- 一些用于微□ jail 行□和安全限制的□□。
- 一些可以通□ FreeBSD Ports 套件安装的高□ jail 管理□用程序， □些程序可以用于□□一般的基于 jail 的解决方案。

16.5.1. FreeBSD 提供的用于微□ jail 的系□工具

□于 jail 的配置微□， 基本上都是通□□置 sysctl(8) □量来完成的。 系□提供了一个特殊的 sysctl 子□， 全部相□的□□均在□□子□中； □就是 FreeBSD 内核的 security.jail.* □□子□。 下面是与 jail 有□的主要 sysctl，以及□些□量的默□□。 □些名字都比□容易理解， 如欲了解□一□的□情， □参□□机手册 jail(8) 和 sysctl(8)。

- security.jail.set_hostname_allowed: 1
- security.jail.socket_unixiproute_only: 1
- security.jail.sysvipc_allowed: 0
- security.jail.enforce_statfs: 2
- security.jail.allow_raw_sockets: 0
- security.jail.chflags_allowed: 0
- security.jail.jailed: 0

系□管理□可以在 宿主系□ 中， 透□□置□些□量的□来默□□ root 用□□加或取消限制。 需要注意的是，某些限制是不能□取消的。 在 jail(8) 中的 root 用□， 无法挂□或卸下文件系□， 此外在 jail 中的 root 用□也不能加□或卸□ devfs(8) □□集、 配置防火□□□， 或□行其他需要修改内核数据的管理操作， 例如□置内核的 securelevel 等等。

FreeBSD 的基本系□包含一系列用于□看目前在使用的 jail 信息， 以及接入 jail 并□行管理命令所需的基本工具。 jls(8) 和 jexec(8) 命令都是 FreeBSD 基本系□的一部分， 并可用于□行□□的任□：

- 列出在用的 jail 以及□□的 jail □□ (JID)、 IP 地址、 主机名和路径。
- 从宿主系□中接入正在□行的 jail， 并在其中□行命令， 以完成一系列 jail 管理任□。 □在 root 希望干□地□□ jail □非常有用。 jexec(8) 工具也可以用于在 jail 中□□ shell 以便□其□行管理； 例如：

```
# jexec 1 tcsh
```

16.5.2. 由 FreeBSD Ports 套件提供的高□管理工具

在□多第三方 jail 管理工具中， sysutils/jailutils 是最完整和好用的。 它是一系列方便 jail(8) 管理的小工具。 □参□其网站以了解□一□的□情。

16.6. Jail 的□用

16.6.1. 服□ Jail

□一□主要基于 Simon L. B. Nielsen <simon@FreeBSD.org> 的 http://simon.nitro.dk/service-jails.html 中的思路， 以及由 Ken Tom locals@gmail.com 更新的文□。 □一□中描述了如何配置 FreeBSD 系□的 jail(8) 功能□其□加一个安全□次。 □部分假定□□行 RELENG_6_0 或更新版本， 并理解本章之前部分的内容。

16.6.1.1. □□

jail 的一个主要□□是如何□它□□行升□和管理。 由于□个 jail 都是从□□□的， □于□个 jail 而言升□也□□不是个很□重的□□， 因□升□不会太□麻□， 而□于多个 jail 而言， 升□不□会耗□大量□□， 并且是十分乏味的□程。

□个配置□程需要□□ FreeBSD 有□多的配置和使用□□。 如果□些□程□得太□□□， □□考□使用□□□的系□， 例如 sysutils/ezjail， 它提供了更□□的管理 FreeBSD jail 的方法。

基本的想法是， 在不同的 jail 中尽可能多地以安全的方式使用共享的□源 - 使用只□的 mount_nullfs(8) 挂接， □会□升□□□□多， 从而使□□个服□建立不同的 jail □□方案□得更加可行。 □外， 它也□□加、□除以及升□ jail 提供了更□便捷的方法。

在□里服□的常□例子包括： HTTP 服□、 DNS 服□、 SMTP 服□等等， □如此□。

□□介□的配置的目的包括：

- 建立□□并易于理解的 jail □□。 也就是□ 不必 □□个 jail □行完整的 installworld 操作。
- 使□□ jail 更容易。
- 使更新或升□ jail 更容易。
- 使□行自□的 FreeBSD 分支成□可能。
- □安全的更偏□的追求， 尽可能□少被攻陷的可能。
- 尽可能□省空□和 inode。

如前面提到的那□， □个□□□大程度上依□于将一□只□的主模板 (known as nullfs) 挂接到□一个 jail 中， 并□□个 jail 配置一个可□写的□□。 □□□□可以是物理磁□、 分区， 或以 vnode □后端的 md(4) □□。 在□个例子中， 我□将使用可□写的 nullfs 挂接。

下面的表中描述了文件系□格局：

- □个 jail 挂接到 /home/j 目□下的一个目□。
- /home/j/mroot 是□个 jail 共用的模板， □于所有的 jail 而言都是只□的。
- 在 /home/j 目□中， □个 jail 有一个□□的空目□。
- □个 jail 中都有一个 /s 目□， □个目□将□接到系□中的可□写部分。
- □个 jail □基于 /home/j/skel 建立其可□写空□。
- □个 jailspace (jail 中的可□写部分) □□建到 /home/js。

□假定所有的 jail 都放置于 /home 分区中。 当然， □可以根据需要将□个配置改□需要的任何□子， 但在接下来的例子中， 也□相□地加以□□。

16.6.1.2. 建立模板

□一□将介□□建 jail 所需的只□主模板所需的□□。

一般来□， □□将系□升□到最新的 FreeBSD -RELEASE 分支， 具体做法□参□本手册的相□ 章□。 当更新不可行□， □需要完成 buildworld □程， □外， □□需要 sysutils/cpdup □件包。 我□将使用

portsnap(8) 工具来下 FreeBSD Ports 套件。 在使用手册的 Portsnap 章 中， 提供了初学者的介。

1. 首先， 需要□将要存放只□的 FreeBSD □行文件的文件系□建立一个目□， 接着□入 FreeBSD 源代□的目□， 并在其中安装 jail 模板：

```
# mkdir /home/j /home/j/mroot
# cd /usr/src
# make installworld DESTDIR=/home/j/mroot
```

2. 接着， 准□一□ FreeBSD Ports 套件， 以及用于□行 mergemaster 的 FreeBSD 源代□：

```
# cd /home/j/mroot
# mkdir usr/ports
# portsnap -p /home/j/mroot/usr/ports fetch extract
# cpdup /usr/src /home/j/mroot/usr/src
```

3. □建系□中可□写部分的骨架：

```
# mkdir /home/j/skel /home/j/skel/home /home/j/skel/usr-X11R6
/home/j/skel/distfiles
# mv etc /home/j/skel
# mv usr/local /home/j/skel/usr-local
# mv tmp /home/j/skel
# mv var /home/j/skel
# mv root /home/j/skel
```

4. 使用 mergemaster 安装缺失的配置文件。 接下来， □除 mergemaster □建的多余目□：

```
# mergemaster -t /home/j/skel/var/tmp/temproot -D /home/j/skel -i
# cd /home/j/skel
# rm -R bin boot lib libexec mnt proc rescue sbin sys usr dev
```

5. □在， 将可□写文件系□□接到只□文件系□中。 □□保□在 s/ 目□中建立了□当的符号□接。 如果没有建立目□或建立的位置不正□， 可能会□致安装失□。

```
# cd /home/j/mroot
# mkdir s
# ln -s s/etc etc
# ln -s s/home home
# ln -s s/root root
# ln -s ../s/usr-local usr/local
# ln -s ../s/usr-X11R6 usr/X11R6
# ln -s ../../s/distfiles usr/ports/distfiles
# ln -s s/tmp tmp
# ln -s s/var var
```

6. 最后， □建一个默□的包含下列配置的 /home/j/skel/etc/make.conf：

```
WRKDIRPREFIX?=  /s/portbuild
```

配置 WRKDIRPREFIX 使得在□个 jail 中分□□□ FreeBSD 成□可能。 □注意 ports 目□是只□系□的一部分。 而自□的 WRKDIRPREFIX □使得□□□程得以在 jail 中的可□写部分完成。

16.6.1.3. 建立 Jail

□在我□已□有了完整的 FreeBSD jail 模板， 可以在 /etc/rc.conf 中安装并配置它□了。 □个例子中演示了建立 3 个 jail： "NS"、 "MAIL" 和 "WWW"。

1. 在 /etc/fstab 文件中加入下列配置， 以便□系□自□挂接 jail 的只□模板和□写空□：

```
/home/j/mroot   /home/j/ns     nullfs  ro  0   0
/home/j/mroot   /home/j/mail   nullfs  ro  0   0
/home/j/mroot   /home/j/www    nullfs  ro  0   0
/home/js/ns     /home/j/ns/s   nullfs  rw  0   0
/home/js/mail   /home/j/mail/s nullfs  rw  0   0
/home/js/www    /home/j/www/s  nullfs  rw  0   0
```

□描批次号 (pass number) □ 0 的分区不会在□□□使用 fsck(8) □行□□， 而□存批次号 (dump number) □ 0 的分区□不会在 dump(8) □□□。 我□不希望 fsck □□ nullfs 挂接， 或□ dump □□ jail 中的只□ nullfs 挂接。 □就是□什□在□个 fstab 条目的最后□列是 "0 0" 的原因。

2. 在 /etc/rc.conf 中配置 jail：

```
jail_enable="YES"
jail_set_hostname_allow="NO"
jail_list="ns mail www"
jail_ns_hostname="ns.example.org"
jail_ns_ip="192.168.3.17"
jail_ns_rootdir="/usr/home/j/ns"
jail_ns_devfs_enable="YES"
jail_mail_hostname="mail.example.org"
jail_mail_ip="192.168.3.18"
jail_mail_rootdir="/usr/home/j/mail"
jail_mail_devfs_enable="YES"
jail_www_hostname="www.example.org"
jail_www_ip="62.123.43.14"
jail_www_rootdir="/usr/home/j/www"
jail_www_devfs_enable="YES"
```

□把 jailnamerootdir □量□置成 /usr/home 而不是 /home 的原因是 /home 目□在默□安装的 FreeBSD 上是指向 /usr/home 的一个符号□接。 而 jail namerootdir □量必□是一个 不 包含符号□接的路径， 否□ jail 将拒□□。 可以使用 realpath(1) 工具来决定□一□量□被□予一个什□□的□。 更□□的信息□参□安全公告 FreeBSD-SA-07:01.jail

3. □□个 jail □建所需的只□文件系□挂接点：

```
# mkdir /home/j/ns /home/j/mail /home/j/www
```

4. 在 jail 中安装可□写的模板。 注意□需要使用 sysutils/cpdup， 它能□□助□□保□个目□都是正□地□制的：

```
# mkdir /home/js
# cpdup /home/j/skel /home/js/ns
# cpdup /home/j/skel /home/js/mail
# cpdup /home/j/skel /home/js/www
```

5. □□， 就完成了 jail 的制作， 可以□行了。 首先□ jail 挂接文件系□， 然后使用 /etc/rc.d/jail 脚本来□□它□：

```
# mount -a
# /etc/rc.d/jail start
```

□在 jail □□就□□起来了。 要□□它□是否□行正常， 可以使用 jls(8) 命令。 它的□出□□□似□□：

```
# jls
   JID  IP Address      Hostname                      Path
     3  192.168.3.17    ns.example.org                /home/j/ns
     2  192.168.3.18    mail.example.org              /home/j/mail
     1  62.123.43.14    www.example.org               /home/j/www
```

□□， 就可以登入 jail 并□加用□和配置服□了。 JID 列□出了正在□行的 jail 的□□□号。 □可以使用下面的命令来在 JID □号□ 3 的 jail 中□行管理任□：

```
# jexec 3 tcsh
```

16.6.1.4. 升□

有□， 由于安全□□， 或新□功能有用， 会希望将系□升□到一个新版本的 FreeBSD。 □□安装方式的□□使得升□□有 jail □得很容易。 □外， 它也能最大限度地□小停机□□， 因□ jail 只在最后□刻才需要□□。 □外， 它也提供了□□的回退到先前版本的方法。

1. 第一□是按通常的方法升□主机的系□。 接着， 在 /home/j/mroot2 中建立一个新的□□模板：

```
# mkdir /home/j/mroot2
# cd /usr/src
# make installworld DESTDIR=/home/j/mroot2
# cd /home/j/mroot2
# cpdup /usr/src usr/src
# mkdir s
```

   在□行 `installworld` □会□建一些不需要的目□， □将它□□除：

```
# chflags -R 0 var
# rm -R etc var root usr/local tmp
```

2. 重建到主系□中的可□写符号□接：

```
# ln -s s/etc etc
# ln -s s/root root
# ln -s s/home home
# ln -s ../s/usr-local usr/local
# ln -s ../s/usr-X11R6 usr/X11R6
# ln -s s/tmp tmp
# ln -s s/var var
```

3. □在是□候□□ jail 了：

```
# /etc/rc.d/jail stop
```

4. 卸下原先的文件系□：

```
# umount /home/j/ns/s
# umount /home/j/ns
# umount /home/j/mail/s
# umount /home/j/mail
# umount /home/j/www/s
# umount /home/j/www
```

   可□写的文件系□ (/s) 会在只□系□之后挂接， 因此□首先卸□。

5. 将先前的只□文件系□□走， □成新的系□。 □□做也同□保留了先前系□的□□， 从而可以在出□□□□从中恢□。 □里我□根据新系□的□建□□来命名。 此外我□把先前的 FreeBSD Ports 套件直接移□到新的文件系□中， 以□省磁□空□和 inode：

```
# cd /home/j
# mv mroot mroot.20060601
# mv mroot2 mroot
# mv mroot.20060601/usr/ports mroot/usr
```

6. □在新的只□模板就可以用了， 剩下的事情是重新挂接文件系□并□□ jails：

```
# mount -a
# /etc/rc.d/jail start
```

最后用 jls(8) □□ jail □□是否正常。 不要忘□在 jail 中□行 mergemaster。 配置文件和 rc.d 脚本在升□□□□行更新。

Chapter 17. □制□□控制

17.1. 概要

FreeBSD 5.X 在 POSIX®.1e 草案的基□上引入了 TrustedBSD □目提供的新的安全性□展。新安全机制中最重要的□个， 是文件系□□□控制列表 (ACL) 和□制□□控制 (MAC) 机制。 □制□□控制允□加□新的□□控制模□， 并借此□施新的安全策略， 其中一部分□一个很小的系□子集提供保□并加□特定的服□，其他的□□所有的主体和客体提供全面的□□式安全保□。 定□中有□□制的部分源于如下事□， 控制的□□由管理□和系□作出， 而不像自主□□控制 (DAC，FreeBSD 中的□准文件以及 System V IPC □限) 那□是按照用□意□□行的。

本章将集中□述□制□□控制框架 (MAC 框架) 以及一套用以□施多□安全策略的□件式的安全策略模□。

□□本章之后， □将了解：

- 目前 FreeBSD 中具有□些 MAC 安全策略模□， 以及与之相□的机制。
- MAC 安全策略模□将□施何□策略， 以及□□式与非□□式策略之□的差□。
- 如何高效地配置系□令使其使用 MAC 框架。
- 如何配置 MAC 框架所提供的不同的安全策略模□。
- 如何用 MAC 框架□建更□安全的□境， 并□例□明。
- 如何□□ MAC 配置以□保正□□建了框架。

□□本章之前， □□□：

- 了解 UNIX® 和 FreeBSD 的基□ (UNIX 基□)。
- 熟悉内核配置/□□ (配置FreeBSD的内核) 的基□。
- □安全及其如何与 FreeBSD 相配合有些了解； (安全)。

□本章信息的不当使用可能□致□失系□□□□， 激怒用□， 或者无法□□ X11 提供的特性。更重要的是， MAC 不能用于□底保□一个系□。 MAC 框架□用于□□□有安全策略；如果没有健全的安全条例以及定期的安全□□， 系□将永□不会□□安全。

此外□需要注意的是， 本章中所包含的例子□□是例子。 我□并不建□在一个生□用系□上□行□些特□的□置。 □施各□安全策略模□需要□慎的考□与□□， 因□那些并不完全理解所有机制如何工作的人， 可能会□□需要□整个系□中很多的文件或目□□行重新配置。

17.1.1. 未□及的内容

本章涵□了与 MAC 框架有□的□多方面的安全□□； 而新的 MAC 安全策略模□的□□成果□不会□及。 MAC 框架中所包含的一部分安全策略模□， 具有一些用于□□及新模□□□的特定属性， 其中包括 mac_test(4)、mac_stub(4) 以及 mac_none(4)。 □于□些安全策略模□及其提供的□多机制的□□信息，□参□□机手册中的内容。

17.2. 本章出□的重要□□

在□□本章之前， 有些□□□□需要解□， 希望能藉此□清可能出□的疑惑， 并避免在文中□新□□、 新信息□行生硬的介□。

- 区□(compartment)： (□注： 区□ □一□□， 在一些文献中也称做□□ (category)。 此外， 在其它一些翻□文献中， □□□也翻□□ "象限"。) 指一□被□分或隔□的程序和数据， 其中， 用□被明□地□予了□□特定系□□件的□限。 同□， 区□也能□表□分□， 例如工作□、 部□、 □目， 或□□。 可以通□使用区□来□施 need-to-know 安全策略。
- 高水位□(high water mark)： 高水位□策略是一□允□提高安全□□， 以期□□更高□□的信息的安全策略。 在多数情况下， 当□程□束□， 又会回到原先的安全□□。 目前， FreeBSD MAC 框架尚未提供□□的策略， 在□里介□其定□主要是希望□□一个完整的概念。
- 完整性(integrity)： 作□一个□□概念， 完整性是数据可信性的一□程度。 若数据的完整性提高， □数据的可信性相□提高。
- □□(label)： □□是一□可□用于文件、 目□或系□其他客体的安全属性， 它也可以被□□是一□机密性印□。 当一个文件被施以□□□， 其□□会描述□一文件的安全参数， 并只允□□有相似安全性□置的文件、 用□、 □源等□□□文件。 □□□的涵□及解□取决于相□的策略配置： 某些策略会将□□当作□某一客体的完整性和保密性的表述， 而其它一些策略□会用□□保存□□□□。
- 程度(level)： □某□安全属性加□或削弱的□定。 若程度□加， 其安全性也相□□加。
- 低水位□(low water mark)： 低水位□策略允□降低安全□□， 以□□安全性□差的信息。 多数情况下， 在□程□束□， 又会回到原先的安全□□。 目前在 FreeBSD 中唯一□□□一安全策略的是 mac_lomac(4)。
- 多重□□(multilabel)： `multilabel` 属性是一个文件系□□□。 □□□可在□用□模式下通□ tunefs(8) 程序□行□置。 可以在引□□使用的 fstab(5) 文件中， 也可在□建新文件系□□□行配置。 □□□将允□管理□□不同客体施以不同的 MAC □□。 □□□□□用于支持□□的安全策略模□。
- 客体(object)： 客体或系□客体是一□□体， 信息随 主体 的□向在客体内部流□。 客体包括目□、 文件、 区段、 □示器、 □□、 存□器、 磁存□器、 打印机及其它数据存□/□移□□。 基本上， 客体就是指数据容器或系□□源。 □ 客体 的□□□□上意味着□数据的□□。
- 策略(policy)： 一套用以□定如何□成目□的□□。 策略 一般用以描述如何□特定客体□行操作。 本章将在安全策略的□畴内□□策略， 一套用以控制数据和信息流并□定其□□者的□□， 就是其中一例。
- 敏感性(sensitivity)： 通常在□□ MLS □使用。 敏感性程度曾被用来描述数据□□有何等的重要或机密。 若敏感性程度□加， □保密的重要性或数据的机密性相□□□。
- □一□□(single label)： 整个文件系□使用一个□□□数据流□施□□控制， 叫做□一□□。 当文件系□使用此□置□， 即无□何□当 多重□□ □□未被□定□， 所有文件都将遵守相同□□□定。
- 主体(subject)： 主体就是引起信息在□个 客体 □流□的任意活□□体， 比如用□， 用□□程(□注：原文□processor)， 系□□程等。 在 FreeBSD 中， 主体几乎□是代表用□活□在某一□程中的一个□程。

17.3. □于 MAC 的□明

在掌握了所有新□□之后， 我□从整体上来考□ MAC 是如何加□系□安全性的。 MAC 框架提供的□多安全策略模□可以用来保□网□及文件系□， 也可以禁止用□□□某些特定的端口、 套接字及其它客体。 将策略模□□合在一起以□建一个□有多□次安全性的□境， 也□是其最佳的使用方式， □可以通□一次性加□多个安全策略模□来□□。 在多□次安全□境中， 多重策略模□可以有效地控制安全性， □一点与□化型(hardening) 策略， 即那□通常只□化系□中用于特定目的的元素的策略是不同的。 相比之下，

多重策略的唯一不足是需要系□管理□先期□置好参数， 如多重文件系□安全□志、 □一位用□的网□□□□限等等。

与采用框架方式□□的□期效果相比， □些不足之□是微不足道的。 例如， □系□具有□特定配置挑□必需的策略的能力， 有助于降低性能□□。 而□少□无用策略的支持， 不□可以提高系□的整体性能，而且提供了更□活的□□空□。 好的□施方案中□□考□到整体的安全性要求， 并有效地利用框架所提供的□多安全策略模□。

□□一个使用 MAC 特性的系□， 至少要保□不允□用□任意更改安全属性； 所有的用□□用工具、程序以及脚本， 必□在所□安全策略模□提供的□□□□的□束下工作； 并且系□管理□□掌握 MAC □□□□的一切控制□。

□心□□正□的安全策略模□是系□管理□□有的□□。 某些□境也□需要限制网□的□□控制□， 在□□情况下， 使用 mac_portacl(4)、 mac_ifoff(4) 乃至 mac_biba(4) 安全策略模□都会是不□的□始； 在其他情况下， 系□客体也□需要□格的机密性， 像 mac_bsdextended(4) 和 mac_mls(4) □□的安全策略模□就是□此而□。

□安全策略模□的决定可依据网□配置□行， 也□只有特定的用□才□□被允□使用由 ssh(1) 提供的程序以□□网□或互□网， mac_portacl(4) 安全策略模□□□成□□□情况下的□□。 但□文件系□又□作些什□□？ 是由特定的用□或群□来□定某些目□的□□□限， 抑或是将特定客体□□保密以限制用□或□件□□特定文件？

在文件系□的例子中， 也□□□客体的□限□某些用□是保密的， 但□其他□不是。 比如， 一个□大的□□□□， 也□会被分成□多由几人□成的小□， A □目中的□□人□可能不被允□□□ B □目□□人□□作的客体， 但同□他□□需要□□由 C □目□□人□□作的客体， □正符合上述情形。 使用由 MAC 框架提供的不同策略， 用□就可以被分成□□小□，然后被□予□当区域的□□□， 由此， 我□就不用担心信息泄漏的□□了。

因此， □一□安全策略模□都有其□理系□整体安全□□的独特方法。 □安全策略模□的□□□在□安全策略深思熟□的基□之上□行。 很多情况下， 整体安全策略需要重新修正并在系□上□施。 理解 MAC 框架提供的不同安全策略模□会□助管理□就其面□的情形□□最佳的策略模□。

FreeBSD 的默□内核并不包含 MAC 框架□□， 因此， 在□□使用本章中的例子或信息之前， □□□添加以下内核□□：

```
options MAC
```

此外， 内核□需要重新□□并且重新安装。

尽管有□ MAC 的□多□机手册中都声明它□可以被□□到内核中， 但□□些策略模□的使用仍可能□致□死系□的网□及其他功能。 使用 MAC 就像使用防火□一□， 因此必□要小心防止将系□完全□死。 在使用 MAC □， □□考□是否能□回退到之前的配置， 在□程□行配置更□加倍小心。

17.4. 理解 MAC □□

MAC □□是一□安全属性， 它可以被□用于整个系□中的主体和客体。

配置□□□， 用□必□能□□切理解其所□行的操作。 客体所具有的属性取决于被加□的策略模□， 不同策略模□解□其属性的方式也差□很大。 由于缺乏理解或无法了解其□□系而□致的配置不当， 会引起意想不到的， 也□是不□看到的系□□常。

客体上的安全□□是由安全策略模□决定的安全□□控制的一部分。 在某些策略模□中，□□本身所包含的所有信息足以使其作出决策， 而在其它一些安全策略模□中， □□□可能被作□一个□大□□体系的一部分□行□理。

□例来□， 在文件上□定 biba/low □□， 意味着此□□隶属 Biba 策略模□， 其□□ "low"。

某些在 FreeBSD 中支持□□特性的策略会提供三个□定□的□□， 分□是 low、 high 及 equal □□。 尽管□些□□在不同安全策略模□中会□□□控制采取不同措施， 但有一点是可以肯定的， 那就是 low □□表示最低限度的□定， equal □□会将主体或客体□定□被禁用的或不受影□的， high □□□会□用 Biba 及 MLS 安全策略模□中允□的最高□□的□定。

在□一□□文件系□的□境中， 同一客体上只会□用一个□□， 于是， 一套□□□限将被□用于整个系□， □也是很多□境所全部需要的。 □一些□用□景中， 我□需要将多重□□□用于文件系□的客体或主体， 如此一来，就需要使用 tunefs(8) 的 multilabel □□。

在使用 Biba 和 MLS □可以配置数□□□， 以□示分□控制中的□□程度。 数□的程度可以用来□分或将信息按□分□， 从而只允□同程度或更高程度的□□其□行□□。

多数情况下， 管理□将□□整个文件系□□定□一□□。

等一下， □看起来很像 *DAC*！ 但我□□ *MAC* □□只将控制□□予了管理□。 此句□依然是正□的。 在某□程度上，root 是□施控制的用□， 他配置安全策略模□以使用□□被分配到□当的□□/□□ levels 中。 □， 很多安全策略模□同□可以限制 root 用□。 □于客体的基本控制可能会下放□群□， 但 root 用□随□可以□除或更改□些□定。□就是如 Biba 及 MLS □□一些安全策略模□所包含的 hierarchal/clearance 模型。

17.4.1. 配置□□

□□上， 有□□□式安全策略模□配置的各□□□都是用基□系□□件□□的。 □些命令□客体和主体配置以及配置的□施和□□提供了一个□便的接口。

所有的配置都□□通□ setfmac(8) 及 setpmac(8) □件□施。 setfmac 命令是用来□系□客体□置 MAC □□的， 而setpmac □是用来□系□主体□置□□的。 例如：

```
# setfmac biba/high test
```

若以上命令不□生□□□会直接返回命令提示符， 只有当□生□□□， □些命令才会□出提示， □和 chmod(1) 和chown(8) 命令□似。 某些情况下， 以上命令□生的□□可能是 Permission denied， 一般在受限客体上□置或修改□置□会□生此□□。 系□管理□可使用以下命令解决此□□：

```
# setfmac biba/high test
Permission denied
# setpmac biba/low setfmac biba/high test
# getfmac test
test: biba/high
```

如上所示， 通□ setpmac □被□用的□程□予不同的□□， 以覆□安全策略模□的□置。 getpmac □件通常用于当前□行的□程， 如 sendmail： 尽管其使用□程□号来替代命令， 其□□是相同的。 如果用□□□□其无法□□的文件□行操作， 根据所加□的安全策略模□的□□， 函数 mac_set_link 将会□出 Operation not permitted 的

□□提示。

17.4.1.1. 一般□□□型

mac_biba(4)、 mac_mls(4) 及 mac_lomac(4) 策略模□提供了□定□□□□的功能， 其□□□是 high、 equal 及 low 之一。 以下是□□些□□功能的□□描述：

- `low` □□被□□是主体或客体所具有的最低□次的□□□定。 □主体或客体采用此□定， 将阻止其□□□□□ high 的客体或主体。
- `equal` □□只能被用于不希望受策略控制的客体上。
- `high` □□□客体或主体采用可能的最高□定。

至于□个策略模□， □□□定都会□生不同的信息流指令。 □□□机手册中相□的章□将□一□□明□些一般□□配置的特点。

17.4.1.1.1. □□高□配置

如下所示， 用于 比□方式:区□+区□ (`comparison:compartment+compartment`) 的□□等□数：

```
biba/10:2+3+6(5:2+3-20:2+3+4+5+6)
```

其含□□：

"Biba 策略□□"/"等□ 10" ："区□ 2、 3及6"： ("等□5 ...")

本例中， 第一个等□将被□□是 "有效区□" 的 "有效等□"， 第二个等□是低□等□， 最后一个□是高□等□。 大多数配置中并不使用□些□置， □□上， 它□是□更高□的配置准□的。

当把它□□用在系□客体上□， □只有当前的等□/区□， 因□它□反映可以□施□□控制的系□中可用的□□， 以及网□接口。

等□和区□， 可以用来在一□主体和客体之□建立一□称□ "支配 (dominance)" 的□系， □□□系可能是主体支配客体， 客体支配主体， 互不支配或互相支配。 "互相支配" □□情况会在□个□□相等□□生。 由于 Biba 的信息流特性， □可以□置一系列区□， "need to know"， □可能□生于□目之□， 而客体也由其□□的区□。 用□可以使用 `su` 和 `setpmac` 来将他□的□限□一□□分， 以便在没有限制的区□里□□客体。

17.4.1.2. 用□和□□□置

用□本身也需要□置□□， 以使其文件和□程能□正□地与系□上定□的安全策略互□， □是通□使用登□分□在文件 login.conf 中配置的。 □个使用□□的策略模□都会□行用□分□□定。

以下是一个使用所有策略模□的例子：

```
default:\
    :copyright=/etc/COPYRIGHT:\
    :welcome=/etc/motd:\
    :setenv=MAIL=/var/mail/$,BLOCKSIZE=K:\
    :path=~/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin:\
    :manpath=/usr/shared/man /usr/local/man:\
    :nologin=/usr/sbin/nologin:\
    :cputime=1h30m:\
    :datasize=8M:\
    :vmemoryuse=100M:\
    :stacksize=2M:\
    :memorylocked=4M:\
    :memoryuse=8M:\
    :filesize=8M:\
    :coredumpsize=8M:\
    :openfiles=24:\
    :maxproc=32:\
    :priority=0:\
    :requirehome:\
    :passwordtime=91d:\
    :umask=022:\
    :ignoretime@:\
    :label=partition/13,mls/5,biba/10(5-15),lomac/10[2]:
```

label □□用以□定用□分□默□□□， □□□将由 MAC □行。 用□□不会被允□更改□□， 因此其从用□的□点看不是可□的。 当然， 在真□情况的配置中， 管理□不会希望□用所有策略模□。 我□建□□在□施以上配置之前□□本章的其余部分。

用□也□会在首次登□后更改其□□， 尽管如此， □□□是策略的主□局限性。 上面的例子告□ Biba 策略， □程的最小完整性是□5， 最大完整性□15， 默□且有效的□□□10。 □程将以 10的完整性□行直至其决定更改□□， □可能是由于用□使用了 setpmac 命令 (□操作将在登□□被 Biba 限制在一定用□□□之内)。

在所有情况下， 修改 login.conf 之后， 都必□使用 cap_mkdb 重□□登□分□ capability 数据□，□在接下来的例子和□□中就会有所体□。

很多站点可能□有数目可□的用□需要不同的用□分□， 注意到□点是大有裨益的。 深入来□就是需要事先做好□□， 因□管理起来可能十分困□。

在 FreeBSD 以后的版本中， 将包含一□将用□映射到□□的新方式， 尽管如此， □也要到 FreeBSD 5.3 之后的某个□□才能□□。

17.4.1.3. 网□接口和□□□定

也可以在网□接口上配置□□， 以控制□出网□的数据流。 在所有情况下， 策略都会以□□客体的方式□作。 例如， 在 biba 中□置□高的用□， 就不能□□□□□低的网□接口。

maclabel 可以作□ ifconfig 的参数用于□置网□接口的 MAC □□。 例如：

```
# ifconfig bge0 maclabel biba/equal
```

将在 bge(4) 接口上□置 biba/equal 的 MAC □□。 当使用□似 biba/high(low-high) □□的□□□， 整个□□□使用引号括起来； 否□将□生□□。

□一个支持□□的策略模□都提供了用于在网□接口上禁用□ MAC □□的系□控制□量。 将□□□置□ equal 的效果与此□似。 □参□ sysctl 的□出、 策略模□的□机手册， 或本章接下来的内容， 以了解更□一□的□情。

17.4.2. 用□一□□□是多重□□？

默□情况下， 系□采用的是 singlelabel □□。 但□□管理□意味着什□□？ □□策略之□存在很多的不同之□， 它□在系□安全模型的□活性方面， 提供了不同的□□。

singlelabel 只允□在□个主体或客体上使用一个□□， 如 biba/high。 □降低了管理的□□， 但也同□降低了支持□□的策略的□活性。 □多管理□可能更希望在安全策略中使用 multilabel。

multilabel □□允□□一个主体或客体□有各自独立的 MAC □□， 起作用与□准的、 只允□整个分区上使用一个的 singlelabel □□□似。 multilabel 和 single □□□□只有□□□了□□功能的那些策略， 如 Biba、 Lomac、 MLS 以及 SEBSD 才有意□。

很多情况下是不需要□置 multilabel 的。 考□下列情形和安全模型：

- 使用了 MAC 以及□多混合策略的 FreeBSD web-服□器。
- □台机器上的整个系□中只需要一个□□， 即 biba/high。 此□的文件系□并不需要 multilabel □□， 因□有效的 label 只有一个。
- 因□□台机器将作□ Web 服□器使用， 因此□□以 biba/low □行 Web 服□， 以杜□向上写。 Biba 策略以及它如何□作将在□后予以□□， 因此， 如果□感□前面的□明□以理解的□， □□□□下面的内容， 再回来□□□些内容就会有□□清晰的□□了。 服□器可以使用□置□ biba/low 的□独的分区， 用于保持其□行□境的状□。 □个例子中□省略了□多内容， 例如， 如何□数据配置□□限制、 参数配置和用□的□置； 它只是□前述的内容提供一个□□的例子。

如果打算使用非□□式策略， 就不需要 multilabel □□了。 □些策略包括 seeotheruids、 portacl 和 partition。

□一个需要注意的事情是， 在分区上使用 multilabel 并建立基于 multilabel 可能会提高系□管理的□□， 因□文件系□中的所有客体都需要指定□□。 □包括□目□、 文件， 甚至□□□点。

接下来的命令将在需要使用多个□□的文件系□上□置 multilabel。 □一操作只能在□用□模式下完成：

```
# tunefs -l enable /
```

交□区不需要如此配置。

某些用□可能会在根分区上配置 multilabel □志□遇到困□。 如果□生□□的情况， □□□本章的 MAC 框架的故障排除。

17.5. □□安全配置

在□施新技□□， 首先□行□□都是非常好的□□。 在□段□□， 管理□一般都□ "□行全面的考察"， □至少□包括下列因素：

- 方案□施的必要条件；
- 方案□施的目□；

就□施 MAC 而言， □包括：

- 如何在目□系□上□信息和□源□行分□。
- 需要限制□□信息或□源的□□， 以及□采用何□限制。
- 需要使用□些 MAC 模□来完成□些目□。

尽管重新配置并修改系□□源和安全配置是可行的， 但□□整个系□并修□□存的文件和用□□号并不是一件□而易□的事情。 □□有助于完成无□□且有效的可信系□□施。 事先 □采用 MAC 的可信系□， 以及其配置做□□行十分有益， 因□□□□施的成□至□重要。 草率散漫地配置 MAC 通常是□致失□的□根。

不同的□境可能会有不同的需求。 建立多□次而完□的安全配置， 可以□少系□正式□□之后所需要的微□。 同□地， 接下来的章□将介□管理□能□使用的各□不同的模□； 描述它□的使用和配置； 除此之外□有一些□于它□最□合的情景的介□。 例如， web 服□器可能希望使用 mac_biba(4) 和 mac_bsdextended(4) 策略，而其他情况下， 例如一台机器上只有少量的本地用□□， mac_partition(4) □是不□的□□。

17.6. 模□配置

在 MAC 框架中的□个模□， 都可以像前述那□□□入内核， 或作□□行□内核模□加□。 推□的用法， 是通□在 /boot/loader.conf 加入□当的□置， 以便在系□□□□的初始化操作□程中加□□些模□。

接下来的一些小□， 将□□□多 MAC 模□， 并□□介□它□的功能。 此外， □一章□将介□一些具体□境中的用例。某些模□支持一□称□□□ (labeling) 的用法， 它可以通□使用□似 "允□做□个而不允□做那个" 的□□来□□□□控制。□□配置文件可以控制允□的文件□□方式、 网□通□， 以及□多其他□限。 在前一□中， 我□已□展示了文件系□中如何通□ `multilabel` □志来□用基于文件或分区的□□控制的方法。

□□□配置在整个系□中只□制一个□□的限制， □也是 `tunefs` □□□什□是 `multilabel` 的原因。

17.7. MAC seeotheruids 模□

模□名： mac_seeotheruids.ko

□□的内核配置： `options MAC_SEEOTHERUIDS`

引□□□： `mac_seeotheruids_load="YES"`

mac_seeotheruids(4) 模□模□并□展了 `security.bsd.see_other_uids` 和 `security.bsd.see_other_gids` `sysctl` □量。 □一模□并不需要□先配置□□， 它能□透明地与其他模□□同工作。

加□模□之后， 下列 `sysctl` □量可以用来控制其功能：

- security.mac.seeotheruids.enabled 将□用模□的功能， 并使用默□的配置。 □些默□□置将阻止用□看到其他用□的□程和 socket。
- security.mac.seeotheruids.specificgid_enabled 将允□特定的□从□一策略中和面。 要将某些□排除在□一策略之外， 可以用 security.mac.seeotheruids.specificgid=XXX sysctl □量。 前述例子中， *XXX* □替□□希望不受限的□ ID 的数□形式。
- security.mac.seeotheruids.primarygroup_enabled 可以用来将特定的主要□排除在策略之外。 使用□一□量□， 不能同□□置 security.mac.seeotheruids.specificgid_enabled。

17.8. MAC bsdextended 模□

模□名： mac_bsdextended.ko

□□的内核配置： options MAC_BSDEXTENDED

引□□□： mac_bsdextended_load="YES"

mac_bsdextended(4) 模□能□□制文件系□防火□策略。 □一模□的策略提供了□准文件系□□限模型的一□□展，使得管理□能□建立一□□似防火□的□□集， 以文件系□□次□□中的保□文件、 □用程序，以及目□。 在□□□□文件系□客体□， 会遍□□□表， 直至□到匹配的□□， 或到□表尾。 □一行□可以通□修改 sysctl(8) 参数，security.mac.bsdextended.firstmatch_enabled 来□行□置。 与 FreeBSD 中的其他防火□□置□似，也可以建一个文件来配置□□控制策略， 并通□ rc.conf(5) □量的配置在系□引□□加□它。

□□表可以通□工具 ugidfw(8) 工具来□入， 其□法□似 ipfw(8)。 此外□可以通□使用 libugidfw(3) □来□□其他的工具。

当使用□一模□模□□□□其小心； 不正□的使用将□致文件系□的某些部分无法□□。

17.8.1. 例子

在加□了 mac_bsdextended(4) 模□之后， 下列命令可以用来列出当前的□□配置：

```
# ugidfw list
0 slots, 0 rules
```

如希望的那□， 目前□没有定□任何□□。 □意味着一切都□可以□□。 要□建一个阻止所有用□， 而保持 root 不受影□的□□， 只需□行下面的命令：

```
# ugidfw add subject not uid root new object not uid root mode n
```

□本身可能是一个很糟□的主意， 因□它会阻止所有用□□行□怕最□□的命令， 例如 ls。 更富于□心的□□可能是：

```
# ugidfw set 2 subject uid user1 object uid user2 mode n
# ugidfw set 3 subject uid user1 object gid user2 mode n
```

□将阻止任何 user1 □ user2 的主目□的全部□□， 包括目□列表。

user1 可以用 not uid user2 代替。 □将同□的□制□□控制□施在所有用□， 而不是□个用□上。

root 用□不会受到□些□□的影□。

我□已□□出了 mac_bsdextended(4) 模□如何□助加□文件系□的大致介□。 要了解更□一□的信息， □参□ mac_bsdextended(4) 和 ugidfw(8) □机手册。

17.9. MAC ifoff 模□

模□名： mac_ifoff.ko

□□的内核配置： options MAC_IFOFF

引□□□： mac_ifoff_load="YES"

mac_ifoff(4) 模□完全是□了立即禁止网□接口， 以及阻止在系□初□□□用网□接口而□□的。 它不需要再系□中配置任何□□， 也不依□于其他 MAC 模□。

□大多数特性都可以通□□整下面的 sysctl 来加以控制。

- security.mac.ifoff.lo_enabled 表示 □用/禁用 □回接口 (lo(4)) 上的全部流量。
- security.mac.ifoff.bpfrecv_enabled 表示 □用/禁用 伯克利包□□器 (bpf(4)) 接口上的全部流量。
- security.mac.ifoff.other_enabled 将在所有其他接口 □用/禁用 网□。

最□常用的 mac_ifoff(4) 用法之一是在不允□引□□程中出□网□流量的□境中□□网□。 □一个建□的用法是撰写一个使用 security/aide 的脚本， 以便自□地在受保□的目□中□□新的或修改□的文件□切断网□。

17.10. MAC portacl 模□

模□名： mac_portacl.ko

□□的内核配置： MAC_PORTACL

引□□□： mac_portacl_load="YES"

mac_portacl(4) 模□可以用来通□一系列 sysctl □量来限制□定本地的 TCP 和 UDP 端口。 本□上 mac_portacl(4) 使得 非-root 用□能□□定到它所指定的特□端口， 也就是那些□号小于 1024 的端口。

在加□之后， □个模□将在所有的 socket 上□用 MAC 策略。 可以□整下列一些配置：

- security.mac.portacl.enabled 将完全 □用/禁用 策略。
- security.mac.portacl.port_high 将□置□ mac_portacl(4) 所保□的最高端口号。
- security.mac.portacl.suser_exempt 如果□置□非零□， 表示将 root 用□排除在策略之外。
- security.mac.portacl.rules 将指定□□的 mac_portacl 策略； □参□下文。

□□的 mac_portacl 策略， 是在 security.mac.portacl.rules sysctl 所指定的一个下列形式的字符串：

rule[,rule,…] 其中可以□出任意多个□□。 □一个□□的形式都是： idtype:id:protocol:port。 □里的 idtype 参数可以是 uid 或 gid， 分□表示将 id 参数解□□用□ id 或□ id。 protocol 参数可以用来□定希望□用到 TCP 或 UDP □□上， 方法是把□一参数□置□ tcp 或 udp。 最后的 port 参数□□出了所指定的用□或□能□□定的端口号。

由于□□集会直接由内核加以解□， 因此只能以数字形式表示用□ ID、 □ ID， 以及端口等参数。 □言之， □不能使用用□、 □， 或端口服□的名字来指定它□。

默□情况下， 在 □-UNIX® 系□中， □号小于 1024 的端口只能□特□□程使用或□定， 也就是那些以 root 身□□行的□程。 □了□ mac_portacl(4) 能□允□非特□□程□定低于 1024 的端口， 就必□首先禁用□准的 UNIX® 限制。 □可以通□把 sysctl(8) □量 net.inet.ip.portrange.reservedlow 和 net.inet.ip.portrange.reservedhigh □置□ 0 来□□。

□参□下面的例子， 或 mac_portacl(4) □机手册中的□明， 以了解□一□的信息。

17.10.1. 例子

下面的例子更好地展示了前面□□的内容：

```
# sysctl security.mac.portacl.port_high=1023
# sysctl net.inet.ip.portrange.reservedlow=0 net.inet.ip.portrange.reservedhigh=0
```

首先我□需要□置使 mac_portacl(4) 管理□准的特□端口， 并禁用普通的 UNIX® □定限制。

```
# sysctl security.mac.portacl.suser_exempt=1
```

□的 root 用□不□因此策略而失去特□， 因此□把 security.mac.portacl.suser_exempt □置□一个非零的□。 □在□已□成功地配置了 mac_portacl(4) 模□， 并使其默□与 □-UNIX® 系□一□□行了。

```
# sysctl security.mac.portacl.rules=uid:80:tcp:80
```

允□ UID □ 80 的用□ (正常情况下， □□是 www 用□) □定到 80 端口。 □□ www 用□就能□□行 web 服□器， 而不需要使用 root □限了。

```
# sysctl security.mac.portacl.rules=uid:1001:tcp:110,uid:1001:tcp:995
```

允□ UID □ 1001 的用□□定 TCP 端口 110 ("pop3") 和 995 ("pop3s")。 □□用□就能□□□接受来□到 110 和 995 的□接□求的服□了。

17.11. MAC partition (分区) 模□

模□名： mac_partition.ko

□□的内核配置： options MAC_PARTITION

引□□□： mac_partition_load="YES"

mac_partition(4) 策略将把□程基于其 MAC □□放到特定的 "partitions" (分区) 中。 □是一□特殊□型的 jail(8)， 但□□者□行比□意□不大。

□个模□□加到 loader.conf(5) 文件中， 以便在□□□程中□用□些□□。

□大多数□一策略的配置是通□ setpmac(8) 工具来完成的， 它将在后面介□。 □个策略可以使用下面的 sysctl：

- security.mac.partition.enabled 将□用□制的 MAC □程 partitions。

当□用了□个□□□， 用□将只能看到他□自己的， 以及其他与他□同□一个 partition 的□程， 而不能使用能□越□ partition 的工具。 例如， insecure class 中的用□， 就无法使用 top 命令， 以及其他需要□生新□程的工具。

要□置或□除 partition □□中的工具， 需要使用 setpmac：

```
# setpmac partition/13 top
```

□将把 top 命令加入到 insecure class 中的用□的□□集。 注意， 所有由 insecure class 中的用□□生的□程， 仍然会留在 partition/13 □□中。

17.11.1. 例子

下面的命令将□示 partition □□以及□程列表：

```
# ps Zax
```

接下来的□个命令将允□察看其他用□的□程 partition □□， 以及那个用□正在□行的□程：

```
# ps -ZU trhodes
```

除非加□了 mac_seeotheruids(4) 策略， 否□用□就看不到 root 的□□。

非常手工化的□□， 可能会在 /etc/rc.conf 中禁用所有的服□， 并用脚本来按不同的□□来□□它□。

下面的几个策略支持基于所□出的三□□□的完整性□定。 □些□□， □同它□的限制， 在模□的□机手册中□行了□一□介□。

17.12. MAC 多□ (Multi-Level) 安全模□

模□名： mac_mls.ko

□□的内核配置： options MAC_MLS

引□□□： mac_mls_load="YES"

mac_mls(4) 策略， 通□□格控制信息流向来控制系□中主体和客体的□□。

在 MLS □境中， "□可 (clearance)" □□会在□一个主体或客体□□上□行□置， □同□□的区□。 由于□些透明度或敏感度可以有六千多个□次， 因此□□一个主体或客体□行配置将是一件□任何系□管理□都感到□疼的任□。 所幸的是， □个策略中已□包含了三个 "立即可用的" □□。

□些□□是 mls/low、 mls/equal 以及 mls/high。 由于□些□□已□在□机手册中□行了介□， □里只□出□要的□明：

- mls/low □□包含了最低配置， 从而允□其他客体支配它。 任何□□□ mls/low 的客体将是地透明度的，从而不允□□□更高□□的信息。 此外， □个□□也阻止□有□高透明度的客体向其写入或□□信息。
- mls/equal □□□放到不希望使用□一策略的客体上。
- mls/high □□是允□的最高□□透明度。 指定了□个□□的客体将支配系□中的其他客体； 但是， 它□将不允□向□低□□的客体泄露信息。

MLS 提供了：

- 提供了一些非□次分□的□次安全模型；
- 固定□□： 不允□向上□， 不允□向下写 (主体可以□取同□或□低□□的客体， 但不能□取高□□的。 □似地，主体可以向同□或□高□写， 而不能向下写)；
- 保密 (防止不□当的数据透露)；
- 系□□□的基□要点， 是在多个敏感□□之□并行地□理数据 (而不泄露秘密的和机密的信息)。

下列 sysctl 可以用来配置特殊服□和接口：

- security.mac.mls.enabled 用来□用/禁用 MLS 策略。
- security.mac.mls.ptys_equal 将所有的 pty(4) □□□□□ mls/equal。
- security.mac.mls.revocation_enabled 可以用来在□□□□□低 grade □撤□客体□□□。
- security.mac.mls.max_compartments 可以用来□置客体的最大区□□次； 基本上， □也就是系□中所允□的最大区□数。

要管理 MLS □□， 可以使用 setfmac(8) 命令。 要在客体上指定□□， 需要使用下面的命令：

```
# setfmac mls/5 test
```

下述命令用于取得文件 test 上的 MLS □□：

```
# getfmac test
```

以上是□于 MLS 策略提供功能的概要。 □一□做法是在 /etc 中建立一个主策略文件， 并在其中指定 MLS 策略信息， 作□ setfmac 命令的□入。 □□方法， 将在其他策略之后□行介□。

17.12.1. □□托管敏感性

通□使用多□安全策略模□， 管理□可以□□如何控制敏感信息的流向。 默□情况下， 由于其默□的禁止向上□以及向下写的性□， 系□会默□将所有客体置于□低的状□。 □□， 所有的客体都可以□□， 而管理□□可以在配置□段慢慢地□行提高信息的敏感度□□的修改。

除了前面介□的三□基本□□□□之外， 管理□□可以根据需要将用□和用□□□行分□， 以阻止它□之□的信息流。一些人□比□熟悉的信息限界□□， 如 机密、 秘密， 以及 □密 可以方便□理解□一概念。 管理□也可以□□地根据□目□□建不同的分□。 无□采用何□分□方法， 在□施限制性的策略之前， 都必□首先想好如何□行□□。

□个安全策略模□最典型的用例是□子商□的 web 服□器， 其上的文件服□保存公司的重要信息以及金融机□的情况。 □于只有□三个用□的个人工作站而言， □可能不甚□用。

17.13. MAC Biba 模□

模□名： mac_biba.ko

□□的内核配置： `options MAC_BIBA`

引□□□： `mac_biba_load="YES"`

mac_biba(4) 模□将加□ MAC Biba 策略。 □个策略与 MLS 策略非常□似， 只是信息流的□□有些相反的地方。通俗地□， □就是防止敏感信息向下□播， 而 MLS 策略□是防止敏感信息的向上□播； 因而， □一□的□多内容都可以同□□用于□□策略。

在 Biba □境中， "integrity" (完整性) □□， 将□置在□一个主体或客体上。 □些□□是按照□次□□建立的。如果客体或主体的□□被提升， 其完整性也随之提升。

被支持的□□是 `biba/low`， `biba/equal` 以及 `biba/high`； 解□如下：

- `biba/low` □□是客体或主体所能□有的最低完整性□□。 在客体或主体上□置它， 将阻止其在更高□□客体或主体□其□行的写操作， □然□仍被允□。
- `biba/equal` □□只□在那些希望排除在策略之外的客体上□置。
- `biba/high` 允□向□低□□的客体上写， 但不允□□那些客体。 推□在那些可能影□整个系□完整性的客体上□置□个□□。

Biba 提供了：

- □次式的完整性□□， 并提供了一□非□次式的完整性分□；
- 固定□□： 不允□向上写， 不允□向下□ (与 MLS 相反)。 主体可以在它自己和□低的□□写，但不能向更高□□□施写操作。 □似地， 主体也可以□在其自己的， 或更高□□的客体， 但不能□取□低□□的客体；
- 完整性 (防止□数据□行不正□的修改)；
- 完整性□□ (而不是 MLS 的敏感度□□)。

下列 `sysctl` 可以用于□□ Biba 策略。

- `security.mac.biba.enabled` 可以用来在机器上□用/禁用是否□施 Biba 策略。
- `security.mac.biba.ptys_equal` 可以用来在 pty(4) □□上禁用 Biba 策略。
- `security.mac.biba.revocation_enabled` 将在支配主体□生□化□□制撤□□客体的□□□。

要操作系□客体上的 Biba 策略， 需要使用 `setfmac` 和 `getfmac` 命令：

```
# setfmac biba/low test
# getfmac test
test: biba/low
```

17.13.1. □□托管完整性

与敏感性不同， 完整性是要□保不受信方不能□信息□行□改。 □包括了在主体和客体之□□□的信息。 □能□□保用□只能修改甚至□□需要他□的信息。

mac_biba(4) 安全策略模□允□管理□指定用□能□看到和□行的文件和程序， 并□保□些文件能□□系□及用□或用□□所信任， 而免受其他威□。

在最初的□□□段， 管理□必□做好将用□分成不同的等□、 □□和区域的准□。 在□□前后，包括数据以及程序和使用工具在内的客体， 用□都会无法□□。 一旦□用了□个策略模□， 系□将默□使用高□□的□□， 而□分用□□□和等□的工作□交由管理□来□行配置。 与前面介□的□□限界不同， 好的□□方法可能□包括 topic。 例如， 只允□□□人□修改代□□、 使用源代□□□器， 以及其他□□工具， 而其他用□□分入其他□□， 如□□人□、 □□人□， 以及普通用□， □些用□可能只□有□□些□料的□限。

通□其自然的安全控制， 完整性□□□低的主体， 就会无法向完整性□□高的主体□行写操作； 而完整性□□□高的主体， 也不能□察或□□低完整性□□的客体。 通□将客体的□□□□最低□， 可以阻止所有主体□其□行的□□操作。 □一安全策略模□□期的□用□合包括受限的 web 服□器、 □□和□□机， 以及源代□□。 而□于个人□端、作□路由器的□算机， 以及网□防火□而言， 它的用□就不大了。

17.14. MAC LOMAC 模□

模□名： mac_lomac.ko

□□的内核配置： `options MAC_LOMAC`

引□□□： `mac_lomac_load="YES"`

和 MAC Biba 策略不同， mac_lomac(4) 策略只允□在降低了完整性□□之后， 才允□在不破坏完整性□□的前提下□□□低完整性□□的客体。

MAC 版本的 Low-watermark 完整性策略不□与□早的 lomac(4) □□相混□， 除了使用浮□的□□来支持主体通□□助□□区□降□之外， 其工作方式与 Biba 大体相似。 □一次要的区□以 `[auxgrade]` 的形式出□。当指定包含□助□□的 lomac 策略□， 其形式□□似于： `lomac/10[2]` □里数字二 (2) 就是□助□□。

MAC LOMAC 策略依□于系□客体上存在普□的□□， □□就允□主体从□低完整性□□的客体□取， 并□主体的□□降□， 以防止其在之后写高完整性□□的客体。 □就是前面□□的 `[auxgrade]` □□， 因此□个策略能□提供更大的兼容性， 而所需要的初始配置也要比 Biba 少。

17.14.1. 例子

与 Biba 和 MLS 策略□似； `setfmac` 和 `setpmac` 工具可以用来在系□客体上放置□□：

```
# setfmac /usr/home/trhodes lomac/high[low]
# getfmac /usr/home/trhodes
lomac/high[low]
```

注意， □里的□助□□是 low， □一特性只由 MAC LOMAC 策略提供。

17.15. MAC Jail 中的 Nagios

下面□出了通□多□ MAC 模□， 并正□地配置策略来□□安全□境的例子。 □只是一个□□， 因此不□被看作四海一家的解决之道。 □□□□一个策略， 而忽略它不能解决任何□□， 并可能在生□□境中□生□□性的后果。

在□始□些操作之前， 必□在□一个文件系□上□置 multilabel □□， □些操作在□一章□始的部分□行了介□。 不完成□些操作， 将□致□□的□果。 首先， □□□已□安装了 net-mngt/nagios-plugins、 net-mngt/nagios， 和 www/apache13 □些 ports， 并□其□行了配置， 且□□正常。

17.15.1. □建一个 insecure (不安全) 用□ Class

首先是在 /etc/login.conf 文件中加入一个新的用□ class：

```
insecure:\
:copyright=/etc/COPYRIGHT:\
:welcome=/etc/motd:\
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K:\
:path=~/bin:/sbin:/bin:/usr/sbin:/usr/bin:/usr/local/sbin:/usr/local/bin
:manpath=/usr/shared/man /usr/local/man:\
:nologin=/usr/sbin/nologin:\
:cputime=1h30m:\
:datasize=8M:\
:vmemoryuse=100M:\
:stacksize=2M:\
:memorylocked=4M:\
:memoryuse=8M:\
:filesize=8M:\
:coredumpsize=8M:\
:openfiles=24:\
:maxproc=32:\
:priority=0:\
:requirehome:\
:passwordtime=91d:\
:umask=022:\
:ignoretime@:\
:label=biba/10(10-10):
```

并在 default 用□ class 中加入：

```
:label=biba/high:
```

一旦完成上述操作， 就需要□行下面的命令来重建数据□：

```
# cap_mkdb /etc/login.conf
```

17.15.2. 引□配置

□在□□□不要重新□□， 我□□需要在 /boot/loader.conf 中□加下面几行， 以便□模□随系□初始化一同加□：

```
mac_biba_load="YES"
mac_seeotheruids_load="YES"
```

17.15.3. 配置用□

使用下面的命令将 root □□属于默□的 class：

```
# pw usermod root -L default
```

所有非 root 或系□的用□， □在需要一个登□ class。 登□ class 是必□的， 否□□些用□将被禁止使用□似 vi(1) □□的命令。 下面的 sh 脚本□能完成□个工作：

```
# for x in `awk -F: '($3 >= 1001) && ($3 != 65534) { print $1 }' \

# /etc/passwd`; do pw usermod $x -L default; done;
```

将 nagios 和 www □□个用□□入不安全 class：

```
# pw usermod nagios -L insecure
```

```
# pw usermod www -L insecure
```

17.15.4. □建上下文文件

接下来需要□建一个上下文文件； □可以把下面的□例放到 /etc/policy.contexts 中。

```
# This is the default BIBA policy for this system.

# System:
/var/run                        biba/equal
/var/run/*                      biba/equal

/dev                            biba/equal
/dev/*                          biba/equal

/var                biba/equal
/var/spool                      biba/equal
/var/spool/*                    biba/equal

/var/log                        biba/equal
/var/log/*                      biba/equal

/tmp                biba/equal
/tmp/*              biba/equal
/var/tmp            biba/equal
/var/tmp/*          biba/equal

/var/spool/mqueue       biba/equal
/var/spool/clientmqueue     biba/equal

# For Nagios:
/usr/local/etc/nagios
/usr/local/etc/nagios/*         biba/10

/var/spool/nagios               biba/10
/var/spool/nagios/*             biba/10

# For apache
/usr/local/etc/apache           biba/10
/usr/local/etc/apache/*         biba/10
```

□个策略通□在信息流上□置限制来□化安全。 在□个配置中， 包括 `root` 和其他用□在内的用□， 都不允□□□ Nagios。 作□ Nagios 一部分的配置文件和□程， 都是完全独立的， 也称□ jailed。

接下来可以用下面的命令将其□入系□：

```
# setfsmac -ef /etc/policy.contexts /
# setfsmac -ef /etc/policy.contexts /
```

随□境不同前述的文件系□布局可能会有所不同； 不□无□如何， 都只能在一个文件系□上□行它。

在 /etc/mac.conf 文件中的 main 小□需要□行下面的修改：

```
default_labels file ?biba
default_labels ifnet ?biba
default_labels process ?biba
default_labels socket ?biba
```

17.15.5. □用网□

在 /boot/loader.conf 中□加下列内容：

```
security.mac.biba.trust_all_interfaces=1
```

将下述内容加入 rc.conf 中的网□接口配置。 如果主 Internet 配置是通□ DHCP 完成的， □需要在□次系□□□之后手工□行□似的配置：

```
maclabel biba/equal
```

17.15.6. □□配置

首先要□□ web 服□以及 Nagios 不会随系□的初始化和重□□程而自□□□。 在此之前， □在此□□ root 用□不能□□ Nagios 配置目□中的任何文件 如果 root 能□在 /var/spool/nagios 中□行 ls(1)， □表示配置有□。 如果配置正□的□， □会收到一条 "permission denied" □□信息。

如果一切正常， Nagios、 Apache， 以及 Sendmail 就可以按照□□安全策略的方式□□了。 下面的命令将完成此工作：

```
# cd /etc/mail && make stop && \
setpmac biba/equal make start && setpmac biba/10\(10-10\) apachectl start && \
setpmac biba/10\(10-10\) /usr/local/etc/rc.d/nagios.sh forcestart
```

再次□□是否一切正常。 如果不是的□， □□□日志文件和□□信息。 此外， □可以用 sysctl(8) 来□□禁用 mac_biba(4) 安全策略模□的□制措施， 并象之前那□□行配置和□□服□。

root 用□可以放心大胆地修改安全□制措施， 并□□配置文件。 下面的命令可以□安全策略□行降□， 并□□一个新的 shell：

```
# setpmac biba/10 csh
```

要阻止□□情况□生， 就需要配置 login.conf(5) 中□可的命令□□了。 如果 setpmac(8) □□□行超越□可□□的命令， □会返回一个□□， 而不是□行命令。 在□个例子中， 可以把 root □□ biba/high(high-high)。

17.16. User Lock Down

□个例子□□的是一个相□□小的存□系□， 其用□数少于五十。 用□能□在其上登□， 除了存□数据之外， □可以□□一些其他□源。

在□个□景中， mac_bsdextended(4) 可以与 mac_seeotheruids(4) 并存， 以□到禁止□□非授□□源， 同□□藏其他用□的□程的目的。

首先， 在 /boot/loader.conf 中加入：

```
mac_seeotheruids_load="YES"
```

随后， 可以通□下述 rc.conf □量来□用 mac_bsdextended(4) 安全策略模□：

```
ugidfw_enable="YES"
```

默□□□保存在 /etc/rc.bsdextended 中， 并在系□初始化□加□； 但是， 其中的默□□可能需要□行一些改□。 因□□台机器只□□得了授□的用□提供服□， 因此除了最后□□之外， 其它内容都□保持注□的状□。 □□□□□将默□□制加□属于用□的系□客体。

在□台机器上添加需要的用□并重新□□。 出于□□的目的， □在□个控制台上分□以不同的用□身□登□。 □行 ps aux 命令来看看是否能看到其他用□的□程。 此外， 在其他用□的主目□中□行 ls(1) 命令， 如果配置正□，□□个命令会失□。

不要□□以 root 用□的身□□行□□， 除非□已□修改了特定的 sysctl 来阻止超□用□的□□。

在添加新用□□， 他□的 mac_bsdextended(4) □□不会自□出□在□□集表中。 要迅速更新□□集， 只需□□地使用 kldunload(8) 和 kldload(8) 工具来卸□并重新加□安全策略模□。

17.17. MAC 框架的故障排除

在□□□程中， 有一些用□□告了正常配置下出□的□□。 其中的一些□□如下所示：

17.17.1. 无法在 / 上□用 multilabel □□

multilabel □志在根 (/) 分区上没有保持□用状□！

看起来□五十个用□中就有一个遇到□□的□□， 当然， 在我□的初始配置□程中也出□□□□的□□。 更□一□的□察使得我相信□个所□的 "bug" 是由于文□中不□切的描述， 或□其□生的□解造成的。 无□它是因□什□引□的，下面的□□□□能□解决此□□：

1. □□ /etc/fstab 并将根分区□置□ ro， 表示只□。
2. 重新□□并□入□用□模式。
3. 在 / 上□行 tunefs -l enable
4. 重新□□并□入正常的模式。
5. □行 mount -urw/ 并把 /etc/fstab 中的 ro 改回 rw， 然后再次重新□□。
6. 再次□□来自 mount 的□出， 已□□根文件系□上正□地□置了 multilabel。

17.17.2. 在 MAC 之后无法□□ X11 了

在使用 MAC 建立安全的□境之后， 就无法□□ X 了！

□可能是由于 MAC partition 策略， 或者□某个 MAC □□策略□行了□□的配置□致的。 要□□□个□□， □□□：

1. □□□□信息； 如果用□是在 insecure class 中， □ partition 策略就可能□致□□。 □□将用□的 class 重新改□ default class， 并使用 cap_mkdb 命令重建数据□。 如果□无法解决□□， □□入第二□。
2. 仔□□□□□策略。 □□□□有□□的用□的策略是正□的， 特□是 X11 □用， 以及 /dev □。
3. 如果□些都无法解决□□， 将出□消息和□□的□境的描述， □送到 TrustedBSD 网站上的 TrustedBSD □□□件列表， 或者 FreeBSD 一般□□□件列表 □件列表。

17.17.3. Error: _secure_path(3) cannot stat .login_conf

当我□□从 root 用□切□到其同中的其他用□□， 出□了□□提示 _secure_path: unable to state .login_conf。

□个提示通常在用□□有高于它将要成□的那个用□的 □□□定□出□。 例如， 如果系□上的一个用□ joe □有默□的 biba/low □□， 而 root 用□□有 biba/high， 它也就不能□看 joe 的主目□， 无□ root 是否使用了 su 来成□ joe。 □□情况下， Biba 完整性模型， 就不会允□ root □看在□低完整性□□中的客体。

17.17.4. root 用□名被破坏了！

在普通模式， 甚至是□用□模式中， root 不被□□。 whoami 命令返回了 0 (零) 而 su □提示 who are you?。 到底□生了什□？

□□策略被禁用可能会□致□□的□□， 无□是通□ sysctl(8) 或是卸□了策略模□。 如果打算禁用策略， 或者□□禁用它， □登□性能数据□需要重新配置， 在其中□除 label □□。 仔□□□ login.conf 以□保所有的 label □□都已□□除， 然后使用 cap_mkdb 命令来重建数据□。

□□情况也可能在通□策略来限制□□ master.passwd 文件或□□的那个数据□□□生。 □主要是由于管理□修改受某一 label 限制的文件， 而与系□□的通用策略□生了冲突。 □□， 用□信息将由系□直接□取， 而在文件□承了新的 label 之后□会拒□□□。 此□， 只需使用 sysctl(8) 禁用□一策略， 一切就会恢□正常了。

Chapter 18. 安全事件□□

18.1. 概述

FreeBSD 中包含了□于□粒度安全事件□□的支持。 事件□□能□支持可□的、 □粒度且可配置的， □于各□与安全有□的系□事件， 包括登□、 配置□更， 以及文件和网□□□等的日志□□。 □些日志□□□于在正在□行的系□上□施□控、 入侵□□和事后分析都十分重要。 FreeBSD □□了 Sun 所□布的 BSM API 和文件格式， 并且与 Sun™ 的 Solaris™ 和 Apple® 的 Mac OS® X □□□□兼容。

□一章的重点是安装和配置事件□□。 它介□了事件策略， 并提供了一个□□的配置例子。

□完□章， □将了解：

- 事件□□是什□， 以及它如何工作。
- 如何在 FreeBSD 上□用□和□程配置事件□□。
- 如何使用□□□□摘要和□□工具来□□□□□□行□□。

□□□章之前， □□□：

- 理解 UNIX® 和 FreeBSD 的基□知□ (UNIX 基□)。
- 熟悉□于内核配置和□□的基本方法 (配置FreeBSD的内核)。
- 熟悉安全知□以及如何在 FreeBSD □用它□ (安全)。

□□机制中存在一些已知的限制， 例如并不是所有与安全有□的系□事件都可以□□， □外某些登□机制， 例如基于 X11 □示管理器， 以及第三方服□的登□机制， 都不会在用□的登□会□中正□配置□□。

安全□□机制能□□系□活□生成非常□□的□□信息： 在繁忙的系□中， □□数据如果配置不当会非常的大， 并在一周内迅速超□几个 GB 的尺寸。 管理□□考□□□配置中的□致磁□空□需求的□些□□。 例如， 可能需要□ /var/audit 目□□独分配一个文件系□， 以防止在□□日志所用的文件系□被填□□影□其它文件系□。

18.2. 本章中的一些□□□□

在□始□□□章之前， 我□需要解□一下与□□有□的一些□□的□□：

- 事件 *(event)*： 可□□事件是指能□被□□子系□□□的任何事件。 □例□来， 与安全有□的事件包括□建文件、 建立网□□接， 以及以某一用□身□登□， 等等。 任何事件必要□是 "有主 (attributable)" 的， 即可以最□□于某一已通□□□的用□的身□， 反之， □称□事件是 "无主 (non-attributable)" 的。 无主事件可以是□生在登□□程成功之前的任何事件， 例如□□一次无效密□等。
- □ *(class)*： 事件□是指相□事件的一个命名集合， 通常在□□表□式中使用。 常用的事件□包括 "□建文件" (fc)、 "□行" (ex) 和 "登入和注□" (lo)。
- □□ *(record)*： □□是指描述一个安全事件的日志□。 □□包括□□事件□型、 □行操作的主体 (用□) 信息、 日期和事件信息， 以及与之相□的□象或参数信息， 最后是操作成功或失□。
- □目 *(trail)*： □□□目， 或日志文件， 包含了一系列描述安全事件的□□□□。 典型情况下，

□□□目基本上是以事件□生的□□□序□□的。 只有□得授□的□程， 才能□向□□□目中提交□□。

- □□表□式 *(selection expression)*： □□表□式是包含一系列前□和□□事件□名字， 用以匹配事件的字符串。
- □□ *(preselection)*： 系□通□□一□程来□□事件是否是管理□所感□趣的， 从而避免□他□不感□趣的事件生成□□。 □□配置使用一系列□□表□式， 用以□□事件□□、 要□□的用□， 以及□用于□□□用□身□， 以及未□□用□身□的□程的全局配置。
- □□ *(reduction)*： 从□有的□□□□中□□出用于保留、 打印或分析的□程。 除此之外， 它也表示从□□□□中□去不需要的□□□□的□程。 通□使用□□操作， 管理□可以□□□留□□数据的策略。 例如， □□的□□□□信息， 可能会保留一个月之久， 但在□之后， □□□些□□信息□行□□操作， 只保留登□信息用于存□。

18.3. 安装□□支持

□于事件□□的支持， 已□随□准的 `installworld` □程完成。 管理□可以通□□看 /etc/security 的内容来□□□一点。 □□能看到一些名字以 *audit* □□的文件， 例如 audit_event。

□于□□功能的用□□支持目前是作□ FreeBSD 基本系□的一部分来安装的。 默□内核中也包含了□于事件□□的内核支持， 但如果□使用的是定制内核， 就必□在内核配置文件中明□指定希望添加□一支持：

```
options AUDIT
```

接下来， □□按照 配置FreeBSD的内核 中所介□的□□来完成一次内核的□□和安装。

在□□好并安装了内核， 并重新□□了系□之后， 就可以在 rc.conf(5) 中□加下列配置来□用□□服□了：

在□□、安装了□□□□功能的内核，并重新□□□算机之后， 就可以在 rc.conf(5) 中□加下列配置来□用□□服□了：

```
auditd_enable="YES"
```

此后， 必□重新□□系□， 或通□下面的命令手工□□□□服□来□□□□支持：

```
/etc/rc.d/auditd start
```

18.4. □□□□行配置

所有用于安全□□的配置文件， 都可以在 /etc/security □到。 要□□□□服□， 下面□些文件必□存在：

- audit_class - 包含□于□□□的定□。
- audit_control - 控制□□子系□的特性， 例如默□□□□、 在□□日志所在的卷上保留的最小空□、 □□日志的最大尺寸， 等等。
- audit_event - 文字化的系□□□事件名称和描述， 以及□个事件属于□个□□。
- audit_user - □□特定用□的□□需求， □些配置在登□□会与全局的默□□合并。
- audit_warn - 由 auditd □用的一个可定制的 shell 脚本， 用于在意外情况， 如用于□□日志的空□□少， 或□□日志文件被翻□□， 生成警告信息。

在□□和□□□□配置文件□一定要小心， 因□配置文件中的□□会□致□□事件不正□。

18.4.1. 事件□□表□式

在□□配置文件中的□多地方会用到□□表□式来□定□些事件是需要□□的。 表□式中需要指定要匹配的事件□型，并使用前□指定是否□接受或忽略匹配的事件， 此外， □可以指定一个可□□指明匹配成功或失□的操作。 □□表□式是按从左到右的□序□算的， 而□于□个表□式的情形， □是通□将后一个追加到前一个之后来□□的。

下面列出了在 audit_class 中的默□事件□型：

- `all` - *all* (全部) - 表示匹配全部事件□。
- `ad` - *administrative* (管理) - 所有在系□上所□行的管理性操作。
- `ap` - *application* (□用) - □用程序定□的□作。
- `cl` - *file close* (文件□□) - □□□ `close` 系□□用的操作。
- `ex` - *exec* (□行) - □□程序的□行。 □于命令行参数和□境□量的□□是通□在 audit_control(5) 中 `policy` 的 `argv` 和 `envv` 参数来控制的。
- `fa` - *file attribute access* (造□文件属性) - □□□□□象属性， 例如 stat(1)、 pathconf(2) 以及□似事件。
- `fc` - *file create* (□建文件) - □□□建了文件的事件。
- `fd` - *file delete* (□除文件) - □□所□生的文件□除事件。
- `fm` - *file attribute modify* (修改文件属性) - □□文件属性□生□化的事件， 例如 chown(8)、 chflags(1)、 flock(2)， 等等。
- `fr` - *file read* (□文件数据) - □□□取数据、 文件以□方式打□等事件。
- `fw` - *file write* (写文件数据) - □□写入数据、 文件以写方式打□等事件。
- `io` - *ioctl* - □□□ ioctl(2) 系□□用的使用。
- `ip` - *ipc* - □□各□形式的□程□通信 (IPC)， 包括 POSIX 管道和 System V IPC 操作。
- `lo` - *login_logout* - □□系□中□生的 login(1) 和 logout(1) 事件。
- `na` - *non attributable* (无主) - □□无法□□的事件。
- `no` - *invalid class* (无效□) - 表示不匹配任何事件。
- `nt` - *network* (网□) - 与网□操作有□的事件， 例如 connect(2) 和 accept(2)。
- `ot` - *other* (其它) - □□各□□□事件。
- `pc` - *process* (□程) - □□□程操作， 例如 exec(3) 和 exit(3)。

□些□□事件， 可以通□修改 audit_class 和 audit_event □□个配置文件来□行定制。

□个列表中， □个□□□均包含一个表示匹配成功/失□操作的前□， 以及□一□是否是□加或□去□事件□或□型的匹配。

- (none) □□事件的成功和失□□例。
- `+` □□□一□的成功事件。
- `-` □□□一□的失□事件。

- ^ 不□□本□中的成功或失□事件。
- ^+ 不□□本□中的成功事件。
- ^- 不□□本□中的失□事件。

下面例子中的□□字符串表示□□成功和失□的登□/注□事件， 而□□行事件， □只□□成功的：

```
lo,+ex
```

18.4.2. 配置文件

多数情况下， 在配置□□系□□， 管理□只需修改□个文件： audit_control 和 audit_user。 前者控制系□□的□□属性和策略， 而后者□用于□□具体的用□来微□。

18.4.2.1. audit_control 文件

audit_control 文件指定了一系列用于□□子系□的默□□置。 通□□看□个文件， 我□可以看到下面的内容：

```
dir:/var/audit
flags:lo
minfree:20
naflags:lo
policy:cnt
filesz:0
```

□里的 dir □□可以用来□置用于保存□□日志的一个或多个目□。 如果指定了多个目□， □将在填□一个之后□用下一个。 一般而言， □□通常都会配置□保存在一个□用的文件系□之下， 以避免□□系□与其它子系□在文件系□□的□候所□生的冲突。

flags 字段用于□有主事件配置系□□的□□条件。 在前面的例子中， 所有用□成功和失□的登□和注□都会被□□。

minfree 参数用于定□保存□□日志的文件系□上剩余空□的最小百分比。 当超□□一□□□， 将□生一个警告。 前面的例子中， 最小剩余空□比例□置成了□成。

naflags □□表示□□□□□无主事件， 例如作□登□□程和系□服□的那些□程的事件。

policy □□用于指定一个以逗号分隔的策略□志表， 以控制一系列□□行□。 默□的 cnt □志表示系□□在□□失□□□□□行 (□烈建□使用□个□志)。 □一个常用的□志是 argv， 它表示在□□命令□行操作□， 同□□□□□ execve(2) 系□□用的命令行参数。

filesz □□指明了□□日志在自□停止□□和翻□之前允□的最大尺寸。 默□□ 0 表示禁用自□日志翻□。 如果配置的□不是零， 但小于最小□ 512k， □□个配置会被忽略， 并在日志中□□□一消息。

18.4.2.2. audit_user 文件

audit_user 文件允□管理□□特定用□指定□一□的□□需求。 □一行使用□个字段来配置用□的□□： 第一个是 alwaysaudit 字段， 它指明了一□□□用□□会□行□□的事件； 而第二个□是 neveraudit 字段， 它指明了一系列□□用□不□□的事件。

在下述 audit_user 示例文件中， □□了 root 用□的 登□/注□ 事件， 以及成功的命令□行事件， 此外， □□□了 www 用□的文件□建和成功的命令□行事件。 如果与前面的示□ audit_control 文件配合使用， □ root 的 lo □就是多余的， 而□ www 用□而言， 其登□/注□事件也会被□□：

```
root:lo,+ex:no
www:fc,+ex:no
```

18.5. 管理□□子系□

18.5.1. □看□□日志

□□□□是以 BSM 二□制格式保存的， 因此必□使用工具来□其□行修改， 或将其□□□文本。 praudit(1) 命令能□将□□文件□□□□□的文本格式； 而 auditreduce(1) 命令□可以□分析、 存□或打印目的来□□□□日志文件。 auditreduce 支持一系列□□参数， 包括事件□型、 事件□、 用□、 事件的日期和□□， 以及文件路径或操作□象。

例如， praudit 工具会将指定的□□□□□存□□□文本格式的□□日志：

```
# praudit /var/audit/AUDITFILE
```

此□ AUDITFILE 是要□存的□□日志文件。

□□□□中包括一系列□□□□， □些□□由一系列短□ (token) □成， 而 praudit 能把它□□序□示□一行。 □个短□都属于某个特定的□型， 例如 header 表示□□□□□， 而 path □表示在一次名字□□中的文件路径。 下面是一个 execve 事件的例子：

```
header,133,10,execve(2),0,Mon Sep 25 15:58:03 2006, + 384 msec
exec arg,finger,doug
path,/usr/bin/finger
attribute,555,root,wheel,90,24918,104944
subject,robert,root,wheel,root,wheel,38439,38032,42086,128.232.9.100
return,success,0
trailer,133
```

□个□□□□表示一次成功的 execve □用， □行了 finger doug。 在参数短□中是由 shell 提交□内核的命令行。 path 短□包含了由内核□□得到的可□行文件路径。 attribute 短□中包含了□可□行文件的描述， 特□地， 它包括了文件的□限模式， 用以□定□用程序是否是 setuid 的。 subject(主体) 短□描述了主体□程， 并□序□□了□□用□ ID、 生效用□ ID 和□ ID、 □□用□ ID 和□ ID、 □程 ID、 会□ ID、 端口 ID， 以及登□地址。 注意□□用□ ID 和□□用□ ID 是不同的： 用□ robert 在□行□个命令之前已□切□□ root □□， 但它会以最初□行身□□□的用□身□□行□□。 最后， return 短□表示□行成功， 而 trailer 表示□□□一□□。

praudit 可以□□使用 -x 参数来支持 XML 格式的□出。

18.5.2. □□□□□□

由于□□日志可能会很大， 管理□可能会希望□□□□的一个子集来使用， 例如与特定用□相□的□□：

```
# auditreduce -u trhodes /var/audit/AUDITFILE | praudit
```

□将□□保存在 AUDITFILE 中的所有由 trhodes □生的□□日志。

18.5.3. 委派□□□□□限

在 audit □中的用□， □有□取 /var/audit 下的□□□□的□限； 默□情况下， □个□是空的， 因此只有 root 用□可以□取□□□□。 如果希望□某个用□指定□□□□□， □可以将其加入 audit。 由于□看□□日志的内容可以提供□于用□和□程行□的大量深度信息， 在□委派□些□力□， □□必□慎行事。

18.5.4. 通□□□管道来□□□控

□□管道是位于□□文件系□中的自□□制 (cloning) 的虚□□□， 用于□□用程序控制正在□行的□□□□流， □主要是□了□足入侵□□和系□□控□件作者的需要。 不□， □管理□而言， □□管道□□也提供了一□无需冒□□□□文件属主出□□□的麻□， 或由于日志翻□而打断事件流的麻□， 而□□□□□控的方便途径。 要跟踪□□事件流，使用下面的命令行：

```
# praudit /dev/auditpipe
```

默□情况下， □□管道□□□点只有 root 用□才能□□。 如果希望 audit □的成□能□□□它， □在 devfs.rules 中加入下述 devfs □□：

```
add path 'auditpipe*' mode 0440 group audit
```

□参□ devfs.rules(5) 以了解□于配置 devfs 文件系□的□一□信息。

很容易配置出□□事件反□循□， 也就是□看事件的操作本身会□生更多的事件。 例如，如果所有的网□ I/O 均被□□， 又在 SSH 会□中□行 praudit(1)， 就会以很高的速率□生持□的□□事件流， 因□□□示一个事件都会□生新的事件。 建□□在需要在□□管道□□上□行 praudit □， □□一个没有□行□粒度 I/O □□的会□来□行。

18.5.5. □□□□文件的□□

□□□□只由内核写入， 且只能由 auditd 管理。 管理□不□□□使用 newsyslog.conf(5) 或其它工具来完成□□日志的□□工作。 □可以使用 audit 管理工具来□□□□、 重新配置□□系□， 并完成日志□□。 下面的命令将□□□服□□建新的□□日志， 并□信号□内核要求其使用新的日志。 旧日志将□止并被改名， 此□， 管理□就可以操作它了。

```
# audit -n
```

如果 auditd 服□程序没有在□行， □□个命令将失□并□出□□提示。

在 /etc/crontab 加入如下□置， 将使 cron(8) □十二小□将日志□□一次。

```
0     */12       *       *       *       root    /usr/sbin/audit -n
```

□些修改会在□保存 /etc/crontab 后生效。

□于□□□□文件基于尺寸的自□翻□，　可以通□ audit_control(5) 中的 filesz □□来配置，　□个□□在□一章的配置文件一□中已□介□□。

18.5.6. □□□□□□

由于□□□□文件会□得很大，　通常会希望在□□服□□□它□，　□其□行□□或□□。　audit_warn 脚本可以用来在一系列与□□有□的事件□生□，　□行一些用□定□的操作，　□也包括在□□□□翻□□□行清理操作。□例而言，可以在 audit_warn 脚本中加入下列内容来在□□□□□□□□□它：

```
#
# Compress audit trail files on close.
#
if [ "$1" = closefile ]; then
        gzip -9 $2
fi
```

其它存□操作也包括将□□□□□制到一个中央的服□器，　□除旧的□□文件，　或□□□□□□并□除不需要的□□等。□个脚本会在□□□□文件正常□□□□行一次，因此在非正常□□系□□，就不会□行它了。

Chapter 19. 存□

19.1. 概述

□章介□了 FreeBSD 中磁□的使用方法。包括内存□， 网□附属磁□和□准的 SCSI/IDE 存□□□，以及使用 USB 的□□。

□完□章，□将了解到：

- FreeBSD 中用来描述硬□上数据□□的□□ (partitions and slices)。
- 如何在□的系□上□加硬□。
- 如何配置 FreeBSD 来使用 USB 存□□□。
- 如何□置虚□文件系□，例如内存磁□。
- 如何使用配□来限制磁□空□的使用。
- 如何□加磁□安全来□防功□。
- 如何刻□ CD 和 DVD 。
- 用于□□的多□存□媒介。
- 如何在 FreeBSD 上使用□□程序。
- 如何□□到□磁□。
- 文件系□快照是什□， 以及如何有效地使用它□。

在□□章之前，□□□：

- 知道□□去配置和安装新的 FreeBSD 内核 (配置FreeBSD的内核).

19.2. □□命名

下面是在 FreeBSD 上被支持的物理存□□□和它□被分配的□□名。

表 7. 物理磁□命名□□

□□器□型	□□□□命名
IDE 硬□□□器	ad
IDE CDROM □□器	acd
SCSI 硬□以及 USB 大容量存□□□	da
SCSI CDROM □□器	cd
各□非□准 CDROM □□器	用于 Mitsumi CD-ROM 的 mcd 以及用于 Sony CD-ROM □□器的 scd
Floppy drives	fd
SCSI tape drives	sa
IDE tape drives	ast

□□器□型	□□□□命名
Flash drives	`fla` for DiskOnChip® Flash device
RAID drives	`aacd` for Adaptec® AdvancedRAID, `mlxd` and `mlyd` for Mylex®, `amrd` for AMI MegaRAID®, `idad` for Compaq Smart RAID, `twed` for 3ware® RAID.

19.3. 添加磁□

下面□□将会介□如何在一台只有一□磁□的机器上新□一□ SCSI 磁□。 首先 需要□掉□算机，然后按操作□程来安装□□器，控制器和□□程序。由于 各厂家生□的□品各不相同，具体的安装□□不在此文□介□之内。

以 `root` 用□登□。安装完□□后，□□一下 /var/run/dmesg.boot 有没有□到新的磁□。在我□ 的例子中新□加的磁□就是 da1，我□从 /1 挂上它。 (如果□正添加 IDE □□器， □□□名□□是 ad1)。

因□ FreeBSD □行在 IBM-PC 兼容机上，它必□遵循 PC BIOS 分区□□。 □与□□的 BSD 分区是不同的。一个 PC 的磁□最高只能有四个 BIOS 主分区。如果磁□只安装 FreeBSD □可以使用 *dedicated* 模式。□外， FreeBSD 必□安装在 PC BIOS 支持的分区内。FreeBSD 把分区叫作 *slices* □可能会把人□糊□。□也可以在只安装 FreeBSD 的磁□上使用 slices，也可以在安装有其它操作系□的磁□上使用 slices。□不会影□其它操作系□的 `fdisk` 分区工具。

在 slice 方式表示下，□□器被添加到 /dev/da1s1e。 可以□作：SCSI 磁□，□号□ 1 (第二个SCSI 磁□)， slice 1 (PC BIOS 分区 1)， 的 BSD 分区 e 。在有些例子中，也可以□化□ /dev/da1e。

由于 bsdlabel(8) 使用 32-位 的整数来表示扇区号， 因此在多数情况下它的表□力限于□个磁□ 2^32-1 个扇区， 或 2TB。 fdisk(8) 格式允□的起始扇区号不能高于 2^32-1， 而分区尺寸也不能超□ 2^32-1，□□一来通常情况下分区尺寸不能超□ 2TB， 而磁□尺寸□不能超□ 4TB。 sunlabel(8) 格式的限制是□个分区 2^32-1 个扇区， 但可以有 8 个分区， 因而可以支持最大 16TB 的磁□。 □于更大的磁□， 可以使用 gpart(8) 来□建 GPT 分区。 GPT 除了支持大磁□之外， □不受 4 个 slice 的限制。

19.3.1. 使用 sysinstall(8)

1. 使用 Sysinstall

   □可以使用 sysinstall 命令的菜□来分区和□□一个新的磁□。 □一操作需要有 root □限，□可以直接使用 root □□登□或者使用 su 命令来切□到 root 用□。□行 sysinstall ，然后□□ Configure 菜□。在 FreeBSD Configuration Menu 下，上下□□， □□ Fdisk 条目。

2. fdisk 分区□□器

   □入 fdisk 分区□□器后，□□ A ，FreeBSD 将使用全部的磁□。当被告知 "remain cooperative with any future possible operating systems"□，回答 YES。使用 W 保存□才的修改。□在使用 Q 退出 FDISK □□器。下面会看到有□ "主引□区" 的信息。 □在□已□在□行的系□上添加了一个磁□， 因此□□□□ None。

3. Disk Label □□器

   接下来，□□□退出 sysinstall 并且再次□□它，并按照上面的□□直接□入 Label □□。□入 磁□□□□□器。□就是□要□建的 BSD 分区。一个磁□最多可以有 8 个分区，□□□ a-h。有几个分区□□有特殊的用途。a 分区被用来作□根分区(/)。 系□磁□（例如：从那儿□□的分区）必□有一个 a 分区。b 分区被用作交□分区，可以用很多磁□用作交 □分区。 c 分区代表整个硬□，或在 FreeBSD slice 模式下代表整个 slice。其它分区作□一般分区来使用。

   sysinstall 的□□□□器用 e 表示非 root 和非 swap 分区。在□□□□器中，可以使用□入 C □建一个文件系□。当提示□是否是一个 FS（文件系□）或 swap □，□□ FS，然后□出一个加□点（如：/mnt）。 当在 post-install 模式□添加一个磁□， sysinstall 不会在 /etc/fstab 中□建□□，所以是否指定加□点并不重要。

   □在已□准□把新□□写到磁□上，然后□建一个文件系□，可以按下 W。出□任何□□都会不能□建新的分区。可以退出□□□□ 器然后重新□行 sysinstall 。

4. 完成

   下面一□就是□□ /etc/fstab，□□的磁□添加一个新 □□。

19.3.2. 使用命令行工具

19.3.2.1. 使用 Slices

□□安装将允□磁□与可能安装在□□算机上的其它操作系□一起 正□工作，而不会□乱其它操作系□的分区。推□使用□□方法来安装 新磁□，除非□有更好的理由再使用 dedicated 模式！

```
# dd if=/dev/zero of=/dev/da1 bs=1k count=1
# fdisk -BI da1 #初始化新磁□
# bsdlabel -B -w da1s1 auto #加上□□
# bsdlabel -e da1s1 # □在□□□□才□建的磁□分区
# mkdir -p /1
# newfs /dev/da1s1e # □□□建的□个分区重□□个操作
# mount /dev/da1s1e /1 # 挂上分区
# vi /etc/fstab # 完成之后，添加合□的□□到□的 /etc/fstab文件。
```

如果有一个 IDE 磁□，□得要用 ad 替□前面的 da。

19.3.2.2. □用模式

如果□并没有安装其它的操作系□，可以使用 `dedicated` 模式。□住□□模式可能会弄乱 Microsoft 的操作系□，但不会□它□行破坏。 它不□□□到的 IBM OS/2® 的 "appropriate" 分区。

```
# dd if=/dev/zero of=/dev/da1 bs=1k count=1
# bsdlabel -Bw da1 auto
# bsdlabel -e da1               # □建 `e' 分区
# newfs /dev/da1e
# mkdir -p /1
# vi /etc/fstab             # □ /dev/da1e添加一个□□
# mount /1
```

□一□方法：

```
# dd if=/dev/zero of=/dev/da1 count=2
# bsdlabel /dev/da1 | bsdlabel -BR da1 /dev/stdin
# newfs /dev/da1e
# mkdir -p /1
# vi /etc/fstab                 # □ /dev/da1e添加一个□□
# mount /1
```

19.4. RAID

19.4.1. □件 RAID

19.4.1.1. □接磁□□□器配置 (CCD)

□□一个大容量存□比□好的解决方案，最重要的因素是□品的速度、 性能和成本。 通常□三者不可能都□足;要□得比□快和可□的大容量存□□□， 就比□昂□。但如果将成本降下来，那它的速度或可□性就会打折扣。

在□□下面描述的系□□， 价格被□□最重要的因素， 接下来是速度和性能。 □个系□的数据□□速度基本上受限于网□。 性能也非常重要， CCD □□器上的所有数据都被□□到了 CD-R □， 可以很容易地□数据□行恢□。

在□□一个大容量的存□解决方案□，第一□是要□□□自己的需求。 如果□的需求更偏重于速度和性能，那□□的解决方案将就不同于上面的□□。

19.4.1.1.1. 安装硬件

除了 IDE 系□磁□外，□有三个 Western Digital 30GB、5400 RPM 的 IDE 磁□□成了大□ 90G 的□接磁□□□存□空□。 理想情况是□个 IDE 硬□都独占 IDE 控制器和数据□， 但□了尽可能降低成本，通常并不会安装更多的控制器， 而是通□配置跳□，使□个 IDE 控制器都管理一个主□和一个从□。

重□□后，系□ BIOS 被配置成自□□□硬□。FreeBSD □□到它□：

```
ad0: 19574MB <WDC WD205BA> [39770/16/63] at ata0-master UDMA33
ad1: 29333MB <WDC WD307AA> [59598/16/63] at ata0-slave UDMA33
ad2: 29333MB <WDC WD307AA> [59598/16/63] at ata1-master UDMA33
ad3: 29333MB <WDC WD307AA> [59598/16/63] at ata1-slave UDMA33
```

如果 FreeBSD 没有□□到它□，□□定它□的跳□是否□置 正□。大多数 IDE 磁□有一个 "Cable Select" 跳□。□个 不是 □置 master/slave 硬□的跳□。□□文□ 信息来□定正□的跳□□置。

接下来考□的是，如何□建文件系□。□□好好研究一下 vinum(4) (Vinum 卷管理程序)和 ccd(4) □□方式，在□里我□□□ ccd(4)

19.4.1.1.2. 安装 CCD

ccd(4) 允□用□将几个相同的的磁□通□一个□□文件系□ □接起来。要使用 ccd(4)，□需要在内核中配置 ccd(4) 支持□□。把□行加入到内核配置文件中，然后重建内核：

```
device   ccd
```

□ ccd(4) 的支持也可以内核模□的形式□入。

要安装 ccd(4), 首先需要使用 bsdlabel(8) 来□□硬□：

```
bsdlabel -w ad1 auto
bsdlabel -w ad2 auto
bsdlabel -w ad3 auto
```

此□将整个硬□□建□ ad1c, ad2c 和 ad3c。

下一□是改□ disklable 的□型。也可以使用 bsdlabel(8) 来□□：

```
bsdlabel -e ad1
bsdlabel -e ad2
bsdlabel -e ad3
```

□儿在□个已□□置了 EDITOR □境□量的磁□上打□了 disklable，在我我例子中使用的是 vi(1)。

可以看到：

```
8 partitions:
#        size   offset    fstype   [fsize bsize bps/cpg]
  c: 60074784        0    unused        0     0     0   # (Cyl.    0 - 59597)
```

添加一个新的 e 分区□ ccd(4) 用。□可以是 c 分区的一个副本， 但 fstype 必□ 是 4.2BSD。做完之后，□会看到一面□些：

```
8 partitions:
#        size   offset    fstype   [fsize bsize bps/cpg]
  c: 60074784        0    unused        0     0     0   # (Cyl.    0 - 59597)
  e: 60074784        0    4.2BSD        0     0     0   # (Cyl.    0 - 59597)
```

19.4.1.1.3. 建立文件系□

□在已□□个磁□都加上了□□，下面需要建立 ccd(4)。要□□做， 需要使用 ccdconfig(8) 工具，同□要提供□似下面的□□：

```
ccdconfig ccd0 32 0 /dev/ad1e /dev/ad2e /dev/ad3e
```

□个□□的意□和用法如下所示： 配置□□的第一个参数，在□是 /dev/ccd0c。 /dev/ 部分是任□□。下一个参数是文件系□的□入□(interleave)。□入□定□了一个 磁□□中一个分段或条□(stripe)的大小，通常是 512 个字□。所以一个□ 32 的□入□将是 16,384 字□。 □入□□ ccdconfig(8) 附□了□□。如果□要□用□□器□像，需要在□儿指定它。在□个配置中没有做 ccd(4) 的□像，所以把它 □□ 0 (zero)。 ccdconfig(8) 的最后配置是□□的排列□□。使用完整的□□ 路径名。

□行 ccdconfig(8) 后 ccd(4) 就配置好了。□在要□建文件 系□了，参考 newfs(8) □□，□行下同的命令：

```
newfs /dev/ccd0c
```

19.4.1.1.4. 自□□建

最后，要挂上 ccd(4) ，需要先配置它。把当前的配置文件写入 /etc/ccd.conf 中，使用下面的命令：

```
ccdconfig -g > /etc/ccd.conf
```

当重新□□系□□，如果 /etc/ccd.conf 存在， 脚本 /etc/rc 就□行 ccdconfig -C。 □□就能自□配置 ccd(4) 以到它能被挂上。

如果□□□入了□用□模式，在 mount(8) 上 ccd(4) 之前，需要□行下面的命令来配置□列：

```
ccdconfig -C
```

要自□挂接 ccd(4),需要□ ccd(4) 在 /etc/fstab 中配置一个□□，以便在□□□它能被挂上。 如下所示：

```
/dev/ccd0c              /media      ufs     rw      2       2
```

19.4.1.2. Vinum 卷管理

Vinum 卷管理是一个□□虚□磁□的□□□□□工具。它使磁□从 □□□的接口和数据映射中独立出来。与□□的存□□□相比，□加了 □活性、性能和可□性。 vinum(4) □□了 RAID-0、RAID-1 和 RAID-5 三□模式，它

□既可以独立使用，也可□合使用。

参考 Vinum 卷管理程序 得到更多 vinum(4) 的信息。

19.4.2. 硬件 RAID

FreeBSD 支持很多硬件 RAID 控制器。 □些硬件不需要 FreeBSD 指定□件来管理 RAID 系□。

使用 BIOS 支持的硬件，一般情况下□些硬件可以自行操作。 下面是一个□明的描述□置一个 Promise IDERAID 控制器。 当硬件□□装好且系□重□后，屏幕上□示一个□□信息。接着□入硬件□置屏幕。在□里，□可以把所有的磁□□合在一起使用。□□ FreeBSD 将磁□看作一个□□器。其它 □□的 RAID 也可以相□的□行□置。

19.4.3. 重建 ATA RAID1 □列

FreeBSD 允□□□□拔□列中□坏的磁□。 在□重新□□系□之前□注意□一点。

□可能会在 /var/log/messages 或者在 dmesg(8) 的□出中看到□似下面□些的内容：

```
ad6 on monster1 suffered a hard error.
ad6: READ command timeout tag=0 serv=0 - resetting
ad6: trying fallback to PIO mode
ata3: resetting devices .. done
ad6: hard error reading fsbn 1116119 of 0-7 (ad6 bn 1116119; cn 1107 tn 4 sn 11)\\
status=59 error=40
ar0: WARNING - mirror lost
```

使用 atacontrol(8)，□看更多的信息：

```
# atacontrol list
ATA channel 0:
    Master:      no device present
    Slave:   acd0 <HL-DT-ST CD-ROM GCR-8520B/1.00> ATA/ATAPI rev 0

ATA channel 1:
    Master:      no device present
    Slave:       no device present

ATA channel 2:
    Master:  ad4 <MAXTOR 6L080J4/A93.0500> ATA/ATAPI rev 5
    Slave:       no device present

ATA channel 3:
    Master:  ad6 <MAXTOR 6L080J4/A93.0500> ATA/ATAPI rev 5
    Slave:       no device present

# atacontrol status ar0
ar0: ATA RAID1 subdisks: ad4 ad6 status: DEGRADED
```

1. 首先□□将包含故障□的 ata 通道卸下， 以便安全地将其拆除：

```
# atacontrol detach ata3
```

2. □上磁□
3. 重新挂接 ata 通道：

```
# atacontrol attach ata3
Master:  ad6 <MAXTOR 6L080J4/A93.0500> ATA/ATAPI rev 5
Slave:   no device present
```

4. 将新□作□□□□加入□列：

```
# atacontrol addspare ar0 ad6
```

5. 重建□列：

```
# atacontrol rebuild ar0
```

6. 可以通□下面的命令来□看□度：

```
# dmesg | tail -10
[output removed]
ad6: removed from configuration
ad6: deleted from ar0 disk1
ad6: inserted into ar0 disk1 as spare

# atacontrol status ar0
ar0: ATA RAID1 subdisks: ad4 ad6 status: REBUILDING 0% completed
```

7. 等待操作完成。

19.5. USB 存□□□

到目前□止，有□多外部外部存□解决方案， 例如：通用串行□□ (USB)：硬□、USB thumbdrives、CD-R burners 等等。 FreeBSD □□些□□提供了支持。

19.5.1. 配置

USB 大容量存□□□□，在 umass(4), 中提供了□ USB 存□□□的支持。如果□使用 GENERIC 内核，□不必要改□配置文件里的任何内容。 如果□使用了定制的内核，就要□定下面的行出□在□的内核配置文件里：

```
device scbus
device da
device pass
device uhci
device ohci
device ehci
device usb
device umass
```

umass(4) □□程序使用 SCSI 子系□来□□ USB 存□□□， □的 USB □□将被系□看成□一个 SCSI □□。依□□主板上的 USB 芯片， □只□□□ device uhci 或用于 USB 1.X 支持的 device ohci 二者之一即可， 但是□者都加入内核配置文件当中也是无害的。 □于 USB 2.X 控制器的支持由 ehci(4) 提供 (device ehci □一行)。不要忘了如果□加入了上面的几行要重新□□和安装内核。

如果□的 USB □□是一个 CD-R 或 DVD 刻□机， SCSI CD-ROM □□程序， cd(4), 就必□加入内核中通□下面□行：

```
device cd
```

由于刻□机被□□ SCSI □□， 因此， 不□□在内核配置文件中使用 atapicam(4) □□程序。

19.5.2. □□配置

配置好后准□□行□□：□入□的 USB □□， 在系□信息中 (dmesg(8)), □□会出□像下面的□□：

```
umass0: USB Solid state disk, rev 1.10/1.00, addr 2
GEOM: create disk da0 dp=0xc2d74850
da0 at umass-sim0 bus 0 target 0 lun 0
da0: <Generic Traveling Disk 1.11> Removable Direct Access SCSI-2 device
da0: 1.000MB/s transfers
da0: 126MB (258048 512 byte sectors: 64H 32S/T 126C)
```

当然□，商□，□□□□ (da0) 和其它的□□信息会根据□的配置不同 而有所不同。

因□ USB □□被看作 SCSI □□中的一个， camcontrol 命令也能□用来列出 USB 存□□□和系□的□□：

```
# camcontrol devlist
<Generic Traveling Disk 1.11>      at scbus0 target 0 lun 0 (da0,pass0)
```

如果□□上已□包含了文件系□， □在□□就可以挂接它了。 如果需要， □参□ 添加磁□ 来了解如何在 USB□□器上格式化和□建分区。

允□非可信用□挂□任意介□， 例如通□使用前面介□的 vfs.usermount 来□用的功能，从安全角度来看是很不保□的。 FreeBSD 中的□大多数文件系□并不提供□□□意□□的内建防□能力。

如果希望□□能□被普通用□挂接， □需要做一些其它操作。 首先， 在 USB 存□□□接到□算机上□， 系□自□生成的□□文件， 必□是□用□能□□写的。 一□做法是□所有属于 operator □的用□都可以□□□□。 要完成□□工作， 首先需要用 pw(8) 来□用□指定□。 其次， 在生成□□文件□， operator □□能□写它□。 □可以通□在 /etc/devfs.rules 中□加一些相□的□置来□□：

```
[localrules=5]
add path 'da*' mode 0660 group operator
```

如果系□中已□有其它 SCSI 磁□， □上述操作必□做一些□化。 例如， 如果系□中已□存在了□□名□ da0 到 da2 的磁□， □第二行□改□：

```
add path 'da[3-9]*' mode 0660 group operator
```

□会将系□中已□存在的磁□， 排除在属于 operator □的□□之外。

□外， □□需要在 /etc/rc.conf 文件中， □用 devfs.rules(5) □□集：

```
devfs_system_ruleset="localrules"
```

接下来， 需要配置内核， 令普通用□能□挂接文件系□。 最□□的方法是将下面的配置加入到 /etc/sysctl.conf：

```
vfs.usermount=1
```

注意， □个□置只有在下次重□系□□才会生效。 □外， □也可以使用 sysctl(8) 来□置□个□量。

最后一□是□建将要挂接文件系□的目□。 □个目□必□是属于将要挂接文件系□的用□的。 以 root 身□□用□建立属于□用□的 /mnt/username (此□ *username* □替□成用□的登□名， 并把 *usergroup* 替□成用□所属的□)：

```
# mkdir /mnt/username
# chown username:usergroup /mnt/username
```

假□已□□入了一个 USB □□□□， 并且系□将其□□□ /dev/da0s1， 由于□些□□通常是 FAT 文件系□， 用□可以□□挂接它□：

```
% mount -t msdosfs -o -m=644,-M=755 /dev/da0s1 /mnt/username
```

如果拔出□□ (必□首先将其□□的磁□卷卸下)， □□会在系□消息□冲区中看到□似下面的信息：

```
umass0: at uhub0 port 1 (addr 2) disconnected
(da0:umass-sim0:0:0:0): lost device
(da0:umass-sim0:0:0:0): removing device entry
GEOM: destroy disk da0 dp=0xc2d74850
umass0: detached
```

19.5.3. 深入□□

除了 Adding Disks 和 Mounting and Unmounting File Systems 章之外， □□各□手册□也是有益的：umass(4), camcontrol(8), 和 FreeBSD 8.X 的 usbconfig(8) 或者□于更早期 FreeBSD 版本的 usbdevs(8)。

19.6. □建和使用光学介□(CD)

19.6.1. 介□

CD 与普通的磁□相比有很多不同的特性。最初它□是不能被用□写入的。 由于没有磁□和磁道移□□的延□，所以它□可以□□的□行□取。 方便的在□个系□之□□行数据的□□，比起相同大小的存□介□来□。

CD 有磁道，□□系到数据□取□的□□性而不是物理磁□的性能。 要在 FreeBSD 中制作一个 CD，□要准□好要写到 CD 上的数据文件， 然后根据□个 tracks 写入到 CD。

ISO 9660 文件系□被□□用来□理□些差□。 但令人□憾的是， 它也有一些其他文件系□所没有的限制， 不□幸□的是， 它提供了一□□展机制， 使得正□写入的 CD 能□超越□些限制， 而又能在不支持□些□展的系□上正常使用。

sysutils/ port 包括了 mkisofs(8)， □是一个可以用来生成包含 ISO 9660 文件系□的数据文件的程序。他也提供了□于一些□展的支持□□，下面将□□介□。

使用□个工具来刻□ CD 取决于□的 CD 刻□机是 ATAPI 的， □是其他□型的。 □于 ATAPI CD 刻□机，可以使用基本系□附□的 `burncd` 程序。 SCSI 和 USB CD 刻□机， □需要配合 `cdrecord` 程序使用， 它可以通□ sysutils/cdrtools port 安装。 除此之外， 在 ATAPI 接口的刻□机上， 也可以配合 ATAPI/CAM 模□ 来使用 `cdrecord` 以及其它□ SCSI 刻□机撰写的工具。

如果□想使用□□形界面的 CD 刻□□件， 可以考□一下 X-CD-Roast 或 K3b。 □些工具可以通□使用□□□安装包，或通□ sysutils/xcdroast 和 sysutils/k3b ports 来安装。 X-CD-Roast 和 K3b 需要 ATAPI/CAM 模□ 配合 ATAPI 硬件。

19.6.2. mkisofs

mkisofs(8) 程序作□ sysutils/cdrtools port 的一部分， 将生成 ISO 9660 文件系□，其中包含 UNIX® 命名空□中的文件名。 最□□的用法是：

```
# mkisofs -o imagefile.iso /path/to/tree
```

□个命令将□建一个包含 ISO9660 文件系□的 *imagefile.iso* 文件，它是目□□ */path/to/tree* 的一个副本。 在□理□程中， 它将文件名称映射□□准的 ISO9660 文件系□的文件名，将排除那些不典型的 ISO 文件系□的文件。

有很多□□能□用来克服那些限制。特□的，-R □□能□□用 Rock Ridge □展一般的 UNIX® 系□，-J □□能□用用于 Microsoft 系□的 Joliet □展，-hfs □□能用来□建用于 Mac OS® 系□的 HFS 文件系□。

□于那些即将要在 FreeBSD 系□中使用 CD 的人来□，-U □□能用来消除所有文件名的限制。当使用 -R □□□，它会□生一个 文件系□映像，它与□从那儿□□ FreeBSD □是一□的，□然它在□多方面也□反了 ISO 9660 的□准。

最后一个常用的□□是 -b。 它用来指定□□映像的位置， 用以生成 "El Torito" □□ CD。 □个□□使用一个参数，用以指定将写入 CD 的目□的根。 默□情况下， mkisofs(8) 会以常□的 "□□模□" 方式来□建 ISO，因此它希望引□映像文件的尺寸恰好是 1200， 1440 或 2880 KB。 某些引□加□器， 例如 FreeBSD □行版磁□， 并不使用模□模式； □□情况下， 需要使用 -no-emul-boot □□。 因此， 如果 /tmp/myboot 是一个包含了□□映像文件 /tmp/myboot/boot/cdboot 的可引□的 FreeBSD 系□，□就可以使用下面的命令生成 ISO 9660 文件系□映像 /tmp/bootable.iso：

```
# mkisofs -R -no-emul-boot -b boot/cdboot -o /tmp/bootable.iso /tmp/myboot
```

完成□些工作之后， 如果□的内核中配置了 md， 就可以用下列命令来挂接文件系□了：

```
# mdconfig -a -t vnode -f /tmp/bootable.iso -u 0
# mount -t cd9660 /dev/md0 /mnt
```

可以□□ /mnt 和 /tmp/myboot 是一□的。

□可以使用 mkisofs(8) 的其它□□来□整它的行□。特□是修改 ISO 9660 的□分格式，□建 Joliet 和 HFS 格式的磁□。□看 mkisofs(8) □机手册得到更多的□助。

19.6.3. burncd

如果用的是 ATAPI 的 CD 刻□机，可以使用 burncd 命令来刻□□的 CD ISO 映像文件。 burncd 命令是基本系□的一部分，中以使用 /usr/sbin/burncd 来安装。 用法如下：

```
# burncd -f cddevice data imagefile.iso fixate
```

在 *cddevice* 上刻□一□ *imagefile.iso* 的副本。 默□的□□是 /dev/acd0。 □参考 burncd(8) 以了解□置写入速度的参数，如何在刻□完成之后自□□出CD，以及刻□音□数据。

19.6.4. cdrecord

如果没有一个 ATAPI CD 刻□机，必□使用 cdrecord 来刻□□的 CD 。 cdrecord 不是基本系□的一部分;必□ 从 sysutils/cdrtools 或□当的 package 安装它。基本系□的□化可能会引起□个程序的□□。可能是由 "coaster" 引起的。当升□系□□，同□需要升□ port， 或者如果□ 使用 -STABLE， 那□在升□到新版本□也要升□ port。

cdrecord 有□多□□，基本用法与 burncd 相似。刻□一个 ISO 9660 映像文件只需□□做：

```
# cdrecord dev=device imagefile.iso
```

使用 cdrecord 的比□巧妙的方法是□到使用的 dev 。要□到正□的□置，可以使用 cdrecord 的 -scanbus □□，□会□生□□的□果：

```
# cdrecord -scanbus
Cdrecord-Clone 2.01 (i386-unknown-freebsd7.0) Copyright (C) 1995-2004 Jörg Schilling
Using libscg version 'schily-0.1'
scsibus0:
        0,0,0     0) 'SEAGATE ' 'ST39236LW       ' '0004' Disk
        0,1,0     1) 'SEAGATE ' 'ST39173W        ' '5958' Disk
        0,2,0     2) *
        0,3,0     3) 'iomega  ' 'jaz 1GB         ' 'J.86' Removable Disk
        0,4,0     4) 'NEC     ' 'CD-ROM DRIVE:466' '1.26' Removable CD-ROM
        0,5,0     5) *
        0,6,0     6) *
        0,7,0     7) *
scsibus1:
        1,0,0   100) *
        1,1,0   101) *
        1,2,0   102) *
        1,3,0   103) *
        1,4,0   104) *
        1,5,0   105) 'YAMAHA  ' 'CRW4260        ' '1.0q' Removable CD-ROM
        1,6,0   106) 'ARTEC   ' 'AM12S          ' '1.06' Scanner
        1,7,0   107) *
```

□个列表列出了□□的的□当的 dev □。□到□的 CD burner ,使用三个用逗号分隔的数□来表示 dev.在□个例子中，CRW 是 dev=1,5,0，所以正□的□入□是 dev=1,5,0 。有一个很容易的方法可以指定□个□;看看 cdrecord(1) 的介□了解有□音□，控制速度和其他的□西。

19.6.5. □制音□ CD

□可以□□□制 CD，把 CD 上面的音□数据解□□出一系列的文件， 再把□些文件写到一□空白 CD 上。 □个□程□于 ATAPI 和 SCSI □□器来□有些微的不同。

Procedure: SCSI □□器

1. 使用 cdda2wav 来解□□音□。

   ```
   % cdda2wav -vall -D2,0 -B -Owav
   ```

2. 使用 cdrecord 来写 .wav 文件。

   ```
   % cdrecord -v dev=2,0 -dao -useinfo  *.wav
   ```

   □保 *2,0* 被□当地□置了， 具体方法在 cdrecord 中有所描述。

Procedure: ATAPI □□器

借助于 ATAPI/CAM 模□， cdda2wav 同□也能在 ATAPI □□上使用。此工具比起下面推□的方法通常是个更好的□□(□□修正， 字□序□□， 等等)。

1. ATAPI CD □□用 /dev/acddtnn表示□个□道， □里 *d* 是□□器号， *nn* 是□道号，由□位小数位□成，省略前□零。 所以第一个□片上的第一个□道就是 /dev/acd0t01，第二个就是 /dev/acd0t02，第三个就是 /dev/acd0t03，等等。

   □□必□□在 /dev 中出□了□□的文件。 如果□□□有某些□目缺失， □□□制系□重新□□介□：

   ```
   # dd if=/dev/acd0 of=/dev/null count=1
   ```

2. 使用 dd(1) 解□□□个□道。当解□□文件的□候□也必□使用 一个特殊的□大小。

   ```
   # dd if=/dev/acd0t01 of=track1.cdr bs=2352
   # dd if=/dev/acd0t02 of=track2.cdr bs=2352
   ...
   ```

3. 使用 burncd 把解□□的文件刻□到光□上。□必□指定 □些文件是音□文件，□□ burncd 会在刻□完成□□束光□。

   ```
   # burncd -f /dev/acd0 audio track1.cdr track2.cdr ... fixate
   ```

19.6.6. □制数据 CD

□可以把数据 CD □制成一个与之等价的映像文件， 可以使用 mkisofs(8) □建□□文件， 或使用它来□制任何数据 CD。 □里□出的例子假定□的 CDROM □□是 acd0， □□将其替□□□□□使用的 CDROM □□。

```
# dd if=/dev/acd0 of=file.iso bs=2048
```

□在□有一个映像文件了，□可以像上面描述的那□把它刻□成 CD。

19.6.7. 使用数据 CD

□在□已□□建了一□□准的数据 CDROM，□或□想要 挂□来□取上面的□□。 默□情况下，mount(8) 假定文件系□是 ufs □型的。如果□□□下面的命令：

```
# mount /dev/cd0 /mnt
```

□会得到一条 Incorrect super block 的□□信息，没有挂□成功。CDROM 不是 UFS 文件系□，所以□□□□挂□它是 是不行的。□需要告□ mount(8) 文件系□是 ISO9660 □型的，□□ 就可以了。只需要指定 mount(8) 的 -t

cd9660 □□。例如， 如果□想要挂□ CDROM □□， /dev/cd0 到 /mnt 目□，□需要□行：

```
# mount -t cd9660 /dev/cd0 /mnt
```

注意□的□□名 (在□个例子中是 /dev/cd0)可能 有所不同，取决于□的 CDROM 使用的接口。□外， -t cd9660 □□等同于□行 mount_cd9660(8)。上面的例子可以□短 □：

```
# mount_cd9660 /dev/cd0 /mnt
```

用□□方法□基本可以使用任何□到的数据 CDROM。 然而某些有 ISO 9660 □展的光□可能会行□古怪。例如，joliet 光□用□个字□的 unicode 字符存□所有的文件名。 FreeBSD 内核并不使用 Unicode， 但 FreeBSD CD9660 □□可以将 Unicode 字符自□□□□内核可以□□的形式。 如果□□□有些非英文字符□示□□号，就□要使用 -C □□来指定字符集了。 欲了解□一□的□情， □参□□机手册 mount_cd9660(8)。

如果希望通□ -C □□来□行字符集□□， □内核会需要加□ cd9660_iconv.ko 模□。□□工作可以通□在 loader.conf 中加入下列配置：

```
cd9660_iconv_load="YES"
```

并重新□□□算机来完成， 除此之外， 也可以通□ kldload(8) 来手□加□。

有□候，当□□□挂□ CDROM 的□候，会得到一条 Device not configured 的□□信息。□通常 表明 CDROM □□□□托□里没有光□， 或者□□器在□□上不可□。 需要几秒□□□等待 CDROM □□器辨□已□接到反□的信息，□耐心等待。

有□候，SCSI CDROM 可能会□不到，因□没有足□的 □□来□答□□的 reset 信号。如果□有一个 SCSI CDROM □将下面的□□添加到□的内核 配置文件并重建□的内核。

```
options SCSI_DELAY=15000
```

□个告□□的 SCSI □□□□□□停 15 秒□， □□的 CDROM □□器足□的机会来□答 □□ reset 信号。

19.6.8. 刻□原始数据 CD

□可以□□把一个文件目□刻□到 CD 上而不用 □建 ISO 9660 文件系□。有些人□□做是□了□□的 目的。□个□行的比刻□一个□准 CD 速度要快得多：

```
# burncd -f /dev/acd1 -s 12 data archive.tar.gz fixate
```

要重新□回□□刻□到 CD 上的数据， □必□从原始□□□点□取数据：

```
# tar xzvf /dev/acd1
```

□不能像挂□一个通常的 CDROM 一□挂□□□光□。 □□的 CDROM 也不能在除了 FreeBSD 之外的任何操作系□上□出。 如果□想要可以挂□ CD，或者 和□一□操作系□共享数据，□必□像上面描述的那□使用 mkisofs(8)。

19.6.9. 使用 ATAPI/CAM □□

□个□□允□ ATAPI □□(CD-ROM, CD-RW, DVD □□器等...)通□ SCSI 子系□□□， □□允□使用像 sysutils/cdrdao 或者 cdrecord(1) □□的程序。

要使用□个□□， □需要把下面□行添加到 /boot/loader.conf 文件中：

```
atapicam_load="YES"
```

接下来， 重新□□□算机。

如果□希望将 atapicam(4) 以静□□□的形式加入内核， □需要在内核配置文件中加入□行：

```
device atapicam
```

此外□需要在内核配置文件中加入：

```
device ata
device scbus
device cd
device pass
```

□些□□已□有了。 然后， 重新□□并安装新内核， 并重新□□□算机。

在引□□程中， □的刻□机将会出□在内核的提示信息中， 就像□□：

```
acd0: CD-RW <MATSHITA CD-RW/DVD-ROM UJDA740> at ata1-master PIO4
cd0 at ata1 bus 0 target 0 lun 0
cd0: <MATSHITA CDRW/DVD UJDA740 1.00> Removable CD-ROM SCSI-0 device
cd0: 16.000MB/s transfers
cd0: Attempt to query device size failed: NOT READY, Medium not present - tray closed
```

□□器□在可以通□ /dev/cd0 □□名□□了，例如要 挂□ CD-ROM 到 /mnt，只需要□入下面的 命令：

```
# mount -t cd9660 /dev/cd0 /mnt
```

作□ root，□可以□行下面的 命令来得到刻□机的 SCSI 地址：

```
# camcontrol devlist
<MATSHITA CDRW/DVD UJDA740 1.00>   at scbus1 target 0 lun 0 (pass0,cd0)
```

□□ 1,0,0 就是 SCSI 地址了，可以被 cdrecord(1) 和其他的 SCSI 程序使用。

有□ ATAPI/CAM 和 SCSI 系□的更多信息， 可以参□ atapicam(4) 和 cam(4) 手册 □。

19.7. □建和使用光学介□(DVD)

19.7.1. 介□

和 CD 相比，DVD 是下一代光学存□介□技□。 DVD 可以容□比任何 CD 更多的数据，已□成□□今□□出版□的□准。

我□称作可□□ DVD 的有五□物理□□格式：

- DVD-R：□是第一□可用的 DVD 可□□格式。 DVD-R □准由 DVD Forum 定□。 □□格式是一次可写的。
- DVD-RW：□是 DVD-R □准的可覆写版本。 一□ DVD-RW 可以被覆写大□ 1000 次。
- DVD-RAM：□也是一□被 DVD Forum 所支持的可覆写格式。 DVD-RAM 可以被看作一□可移□硬□。然而，□□介□和大部分 DVD-ROM □□器以及 DVD-Video 播放器不兼容； 只有少数 DVD 刻□机支持 DVD-RAM。 □参□ 使用 DVD-RAM 以了解□于如何使用 DVD-RAM 的□一□□情。
- DVD+RW：□是一□由 DVD+RW Alliance 定□的可覆写格式。一□ DVD+RW 可以被覆写大□ 1000 次。
- DVD+R：□□格式是 DVD+RW 格式的一次可写□□。

一□□□的可□□ DVD 可以存□ 4,700,000,000 字□，相当于 4.38 GB 或者□ 4485 MB (1 千字□等于 1024 字□)。

必□□明一下物理介□与□用程序的分□。 例如 DVD-Video 是一□特殊的文件系□，可以被覆写到任何可□□的 DVD 物理介□上： DVD-R、DVD+R、DVD-RW 等等。在□□介□□型之前， □一定要□□刻□机和 DVD-Video 播放器 (一□□独的播放器或者□算机上的 DVD-ROM □□器) 是和□□介□兼容的。

19.7.2. 配置

growisofs(1) 将被用来□施 DVD 刻□。 □个命令是 dvd+rw-tools 工具集 (sysutils/dvd+rw-tools) 的一部分。 dvd+rw-tools 支持所有的 DVD 介□□型。

□些工具将使用 SCSI 子系□来□□□，因此 ATAPI/CAM 支持 必□加入内核。 如果□的刻□机采用 USB 接口□不需要□□做，□参考 USB 存□□ 来了解 USB □□配置的□一□□情。

此外，□需要□用 ATAPI □□的 DMA 支持。 □一工作可以通□在 /boot/loader.conf 文件中加入下面的行来完成：

```
hw.ata.atapi_dma="1"
```

□□使用 dvd+rw-tools 之前□□□参考 dvd+rw-tools 硬件兼容性列表 是否有与□的 DVD 刻□机有□的信息。

如果□想要一个□形化的用□界面，□□□看一看 K3b (sysutils/k3b)，它提供了 growisofs(1) 的一个友好界面和□多其他刻□工具。

19.7.3. 刻□数据 DVD

growisofs(1) 命令是 mkisofs 的前端，它会□用 mkisofs(8) 来□建文件系□布局，完成到 DVD 上的刻□。□意味着□不需要在刻□之前□建数据映像。

要把 /path/to/data 目□的数据刻□到 DVD+R 或者 DVD-R 上面，使用下面的命令：

```
# growisofs -dvd-compat -Z /dev/cd0 -J -R /path/to/data
```

-J -R □□□□□ mkisofs(8) 用于文件系□□建 (□表示□建□有□有 joliet 和 Rock Ridge □展的 ISO 9660 文件系□)，参考 mkisofs(8) □机手册了解更多□□。

□□ -Z 用来在任何情况下初始刻□会□： 不管多会□与否。 DVD □□，*/dev/cd0*， 必□依照□的配置做出改□。-dvd-compat 参数会□束光□， 光□成□不可附加的。□会提供更多的和 DVD-ROM □□器的介□兼容性。

也可以刻□成一个 pre-mastered 映像，例如□□一个映像文件 *imagefile.iso*，我□可以□行：

```
# growisofs -dvd-compat -Z /dev/cd0=imagefile.iso
```

刻□的速度可以被□□到并自□□行□整， 根据介□和□□器的使用情况。如果□想□制改□速度， 可以使用 -speed= 参数。更多的信息，□看 growisofs(1) □机手册。

如果需要在刻□的□□中添加超□ 4.38GB 的□个文件， 就必□使用 mkisofs(8) 或其他相□工具 (例如 growisofs(1)) 的 -udf -iso-level 3 参数来□建 UDF/ISO-9660 混合文件系□。只有在□建 ISO 映像文件或直接在□上写数据□才需要□□做。 以□□方式□建的光□必□通□ mount_udf(8) 工具以 UDF 文件系□挂□， 因此只有操作系□支持 UDF □才可以□□做， 否□□上的文件数据可能会无法正□□出。

要□建□□的 ISO 文件：

```
% mkisofs -R -J -udf -iso-level 3 -o imagefile.iso /path/to/data
```

直接将文件刻□到光□上：

```
# growisofs -dvd-compat -udf -iso-level 3 -Z /dev/cd0 -J -R
/path/to/data
```

假如只是使用包含巨型文件的 ISO 映像文件□， 就不需要在□行 growisofs(1) 来将映像文件刻□成光□□指定任何□外的□□了。

□外， 在映像文件中□加或直接刻□巨型文件□， □需要注意使用最新的 sysutils/cdrtools (包含了 mkisofs(8))， 因□旧版并不提供巨型文件支持。 如果□遇到□□， 也可以□□一下□□版本的□件包， 例如 sysutils/cdrtools-devel 并参□ mkisofs(8) □机手册。

19.7.4. 刻□ DVD-Video

DVD-Video 是一□特殊的基于 ISO 9660 和 micro-UDF (M-UDF) □□的文件系□。 DVD-Video 也呈□了一个特殊的数据格式， □就是□什□□需要一个特殊的程序像 multimedia/dvdauthor 来制作 DVD 的原因。

如果□已□有了 DVD-Video 文件系□的映像， 就可以以同□的方式制作□一个映像，可以参看前面章□的例子。 如果□想制作 DVD 并想放在特定的目□中，如在目□ /path/to/video 中， 可以使用下面的命令来刻□ DVD-Video：

```
# growisofs -Z /dev/cd0 -dvd-video /path/to/video
```

-dvd-video □□将□□□ mkisofs(8) 并指示它□建一个 DVD-Video 文件系□布局。 除此之外。 -dvd-video □□也包含了 -dvd-compat growisofs(1) □□。

19.7.5. 使用 DVD+RW

不像 CD-RW，一个空白的 DVD+RW 在□一次使用前必□先格式化。 growisofs(1) 程序将会□□的自□□其□行□当的□理， □是 *recommended* 的方式。□也可以使用 dvd+rw-format 来□ DVD+RW □行格式化：

```
# dvd+rw-format /dev/cd0
```

□只需要□行□□的操作一次，牢□只有空白的 DVD+RW 介□才需要格式化。□可以以前面章□同□的方式来刻□ DVD+RW。

如果□想刻□新的数据 (刻□一个新的完整的文件系□ 而不□□是追加一些数据) 到 DVD+RW，□不必再将其格式化成空白□， □只□要直接覆□掉以前的□□即可。 (□行一个新的初始化□□), 像□□：

```
# growisofs -Z /dev/cd0 -J -R /path/to/newdata
```

DVD+RW 格式化程序□□□的向以前的□□追加数据提供了可能性。 □个操作有一个新的会□和一个已□存在的会□合并而成。 它不需要多个写会□□程， growisofs(1) 将在介□上 □加 ISO 9660 文件系□。

例如，我□想追加一些数据到到我□以前的 DVD+RW 上，我□可以使用下面的命令：

```
# growisofs -M /dev/cd0 -J -R /path/to/nextdata
```

在以后的写操作□， □使用与最初的刻□会□□相同的 mkisofs(8) □□。

如果□想□得与 DVD-ROM □□更好的兼容性，可以使用 -dvd-compat □□。 在 DVD+RW □□情况下， □□做并不妨碍□添加数据。

如果出于某□原因□真的想要空白介□□， 可以□行下面的命令：

```
# growisofs -Z /dev/cd0=/dev/zero
```

19.7.6. 使用 DVD-RW

DVD-RW 接受□□光□格式：□□□序写入和受限式覆写。默□的 DVD-RW □是□序写入格式。

空白的 DVD-RW 能□直接□行刻□而不需要格式化操作， 然而非空的□序写入格式的 DVD-RW 需要格式化才能写入新的初始区段。

要格式化一□ DVD-RW □□序写入模式，□行：

```
# dvd+rw-format -blank=full /dev/cd0
```

一次完全的格式化 (`-blank=full`) 在 1x 倍速的介□上将会花□大□ 1 个小□。快速格式化可以使用 `-blank` □□来□行，如果 DVD-RW 要以 Disk-At-Once (DAO) 模式刻□的□。要以 DAO 模式刻□ DVD-RW，使用命令：

```
# growisofs -use-the-force-luke=dao -Z /dev/cd0=imagefile.iso
```

`-use-the-force-luke=dao` □□不是必需的， 因□ growisofs(1) □□最低限度的□□(快速格式化) 介□并□行 DAO 写入。

事□上□于任何 DVD-RW 都□□使用受限式覆写模式， □□格式比默□的□□□序写入更加□活。

在一□□序 DVD-RW 上写入数据，使用和其他 DVD 格式相同的指令：

```
# growisofs -Z /dev/cd0 -J -R /path/to/data
```

如果□想在□以前的刻□上附加数据，□必□使用 growisofs(1) 的 `-M` □□。然而， 如果□在一□□□□序写入模式的 DVD-RW 上附加数据， 将会在□上□建一个新的区段，□果就是一□多区段光□。

受限式覆写格式的 DVD-RW 在新的初始化区段前不需要格式化， □只是要用 `-Z` □□覆写光□，□和 DVD+RW 的情形是相似的。也可以用和 DVD+RW 同□方式的 `-M` □□把□存的 ISO 9660 文件系□写入光□。 □果会是一□□区段 DVD。

要把 DVD-RW 置于受限式覆写格式， 必□使用下面的命令：

```
# dvd+rw-format /dev/cd0
```

更改回□序写入模式使用：

```
# dvd+rw-format -blank=full /dev/cd0
```

19.7.7. 多区段

几乎没有□个 DVD-ROM □□器支持多区段 DVD，它□大多数□候都只□取第一个区段。 □序写入格式的 DVD+R、DVD-R 和 DVD-RW 可以支持多区段， DVD+RW 和 DVD-RW 受限式覆写格式不存在多区段的概念。

在 DVD+R、DVD-R 或者 DVD-RW 的□序写入格式下， 一次初始化 (未□□) 区段之后使用下面的命令，将会在光□上添加一个新的区段：

```
# growisofs -M /dev/cd0 -J -R /path/to/nextdata
```

□ DVD+RW 或者 DVD-RW 在受限式覆写模式下使用□条命令， 会合并新区段到存在的区段中来附加数据。□果就是一□□区段光□。 □是在□些介□上用于在最初的写操作之后添加数据的方式。

介□上的一些空□用于区段之□区段的□始与□束。 因此，□□用大量的数据添加区段来□化介□空□。 □于 DVD+R 来□区段的数量限制□ 154， □于 DVD-R 来□大□是 2000，□于双□ DVD+R 来□是 127。

19.7.8. 更多的信息

要□得更多的□于 DVD 的信息 `dvd+rw-mediainfo /dev/cd0` 命令可以□行来□得 更多的信息。

更多的□于 dvd+rw-tools 的信息可以在 growisofs(1) □机手册□到，在 dvd+rw-tools web site 和 cdwrite mailing list □接中也可□到。

`dvd+rw-mediainfo` 命令的□出□果□□， 以及介□的□□会被用来做□□□告。 如果没有□些□出，就很□□□解决□□。

19.7.9. 使用 DVD-RAM

19.7.9.1. 配置

DVD-RAM 刻□机通常使用 SCSI 或 ATAPI □□接口之一。 □于 ATAPI □□， DMA □□模式必□手工□用。□一工作可以通□在 /boot/loader.conf 文件中□加下述配置来完成：

```
hw.ata.atapi_dma="1"
```

19.7.9.2. 初始化介□

如本章前面的介□所言， DVD-RAM 可以□□一移□硬□。 与任何其它型号的移□硬□□似， 首次使用它之前，□首先 "初始化" DVD-RAM。 在下面的例子中， 我□将在全部空□上使用□准的 UFS2 文件系□：

```
# dd if=/dev/zero of=/dev/acd0 bs=2k count=1
# bsdlabel -Bw acd0
# newfs /dev/acd0
```

□□根据□□情况将 acd0 改□□所使用的□□名。

19.7.9.3. 使用介□

一旦□在 DVD-RAM 上完成了前面的操作， 就可以像普通的硬□一□挂接它了：

```
# mount /dev/acd0 /mnt
```

然后就可以正常地□ DVD-RAM □行□写了。

19.8. □建和使用□□

把数据存□在□□上有□也是十分有用的。 例如， 在没有其它可□的存□介□， 或只需将少量数据□到其他□算机□。

□一章将介□□□在 FreeBSD 上使用□□。 在使用 DOS 3.5 英寸□□□首要要□及的就是格式化，但其概念与其它的□□格式化□□□似。

19.8.1. 格式化□□

19.8.1.1. □□

□□的□□像其它□□一□是通□在 /dev 中的条目来□□的。 直接□□□□□， 只需□□地使用 /dev/fdN 来表示。

19.8.1.2. 格式化

一□□□在使用□前必□先被低□格式化。 通常□主已□做□了，但格式化是□□介□完整性的一□好方法。 尽管□有可能会□取大量（或少量）的硬□大小，但 大部分磁□都能被格式化□□□ 1440kB 。

低□格式化□□□需要使用 fdformat(1) 命令。□个程序需要□□名作□参数。

要留意一切□□信息，□些信息能□□助□□定 磁□的好与坏。

19.8.1.2.1. □□的格式化

使用 /dev/fdN □□来格式化□□。□入一□新的 3.5 英寸的□□在□的□□中：

```
# /usr/sbin/fdformat -f 1440 /dev/fd0
```

19.8.2. 磁□□□

□□低□格式化后， □需要□它分配一个□□。 □个磁□□□以后会被□去， 但系□需要使用它来□定磁□的尺寸。

新的磁□□□将会接管整个磁□，会包括所有合□的□于□□的 geometry 信息。 磁□□□的 geometry □列在 /etc/disktab中。

□在可以用下面的方法来使用 bsdlabel(8) 了：

```
# /sbin/bsdlabel -B -w /dev/fd0 fd1440
```

19.8.3. 文件系□

□在□□□□行高□格式化。 □会在它上面安置一个新的文件系□，可使 FreeBSD 来□它□行□写。 在□建完新的文件系□后，磁□□□将被消□，所以如果□想重新格式化磁□， □必□重新□建磁□□□。

□□的文件系□可以□□ UFS 或 FAT 。 FAT 是通常情况下□□比□好的□□。

要制作新的文件系□在□□上，可以使用下面的命令：

```
# /sbin/newfs_msdos /dev/fd0
```

□在磁□已□可以□行□取和使用。

19.8.4. 使用□□

要使用□□，需要先使用 mount_msdosfs(8) 挂接它。 除此之外， 也可以使用在 ports 套件中的 emulators/mtools 程序。

19.9. 用磁□机□□

主流的磁□机有 4mm, 8mm, QIC, mini-cartridge 和 DLT。

19.9.1. 4mm (DDS: Digital Data Storage)

4mm 磁□机正在逐□取代 QIC 成□工作站□□数据的首□□。 在 Conner 收□了 QIC 磁□机□域□先的制造商 Archive 之后不久， 即不再生□□□磁□机， □使得□一□□□得愈加明□。 4mm 的□□器更加小和安静，但□于数据保存的可□性仍不及 8mm □□器。它要比 8mm 的便宜和小得多 (3 x 2 x 0.5 inches, 76 x 51 x 12 mm) 。和 8mm 的一□，□写□的寿命都不□，因□它□同□使用螺旋式 的方式来□写。

□些□□的数据□□的速度□在 ~150 kB/s 到 ~500 kB/s 之□， 存□空□从 1.3 GB 到 2.0 GB 之□，硬件□□可使空□加倍。磁□□ □元可以有 6 台磁□机，120 个磁□匣，以自□切□的方式使用同一个磁□□， 磁□□的容量可□ 240 GB 。

DDS-3 □准□在支持的磁□机容量最高可□到 12 GB (或□□的 24 GB)。

4mm 和 8mm 同□都使用螺旋式□写的方式，所有螺旋式□写的□点及缺点， 都可以在 4mm 和 8mm 磁□机上看到。

磁□在□□ 2,000 次的使用或 100 次的全部□□后，就□退休了。

19.9.2. 8mm (Exabyte)

8mm 磁□机是最常□的 SCSI 磁□机，也是磁□交□的最佳□□。几乎□个 工作站都有一台 2 GB 8mm 磁□机。8mm 磁□机可信度高、方便、安静。 □匣小 (4.8 x 3.3 x 0.6 inches; 122 x 84 x 15 mm)而且不□。8mm 磁□机 的下□是一个短短的□写□，而□写□的寿命取决于磁□□□写□□，相□高 速□□情况。

数据□□速度□在 250 kB/s 到 500 kB/s 之□，可存□的空□从 300 MB 到 7 GB，硬件□□可使空□加倍。磁□□□元可以有 6 台磁 □机，120 个磁□匣，以自□切□的方式使用同一个磁□□，磁□□的容量可□ 840+ GB。

Exabyte "Mammoth" 模型支持 12 GB 的容量在一个磁□ 上(□□后可□ 24 GB)相当于普通磁□的二倍。

数据是使用螺旋式□写的方式□□在磁□上的，□写□和磁□□相差 6 度， 磁□以 270 度□□着□，并抵住□写□，□□□地旋□，使得磁□具有高密度， 从一端到□一端并可使磁道□密地分布。

19.9.3. QIC

QIC-150 磁□和磁□机可能是最常□的磁□机和介□了。 QIC 磁□机是最便宜的 "正□" □□□□。 它的缺点在于介□的价格□高。 QIC 磁□要比 8mm 或 4mm 磁□□， □ GB 的数据存□价格可能最高高出 5 倍。 但是， 如果□的需求能□□半打磁□所□足的□， 那□ QIC 可能是明智之□。 QIC 是 最 常□的磁□机。 □个站点都会有某□密度的 QIC。 □有□是一□麻□， QIC 有很多在外□上相似（有□一□），但是密度不同的磁□。 QIC 磁□机噪音很大。 它□在□址以及□写□都会□出声音。 QIC 磁□的□格是 6 x 4 x 0.7 英寸 (152 x 102 x 17 毫米)。

数据□□的速度介于 150 kB/s 到 500 kB/s 之□，可存□的空□ 从 40 MB 到 15 GB。□新的 QIC 磁□机具有硬件□□的功能。 QIC 的使用率愈来愈低，□□被 DAT 所取代。

数据以磁道的方式□□在磁□上，磁道数及磁道的□度会根据容量而有所不同。 通常新的磁□机具有的向后兼容的□取功能（通常也具□写入的功能）。□于数据 的安全性，QIC 具有不□的□价。

磁□机在□□ 5,000 次的使用后，就□退休了。

19.9.4. DLT

在□一章列出的磁□机中 DLT 具有最快的数据□□率。 1/2" (12.5mm) 的 磁□包含在□□的磁□匣 (4 x 4 x 1 inches; 100 x 100 x 25 mm)中。磁□匣 的一□是一个旋□匣道，通□匣道的□合，可以□磁□卷□。磁□匣内只有一个 □，而本章中所提到的其他磁□匣都是有□个□的（9磁道磁□机例外）。

数据□□的速度□ 1.5 MB/s，是 4mm, 8mm, 或 QIC 磁□机的三倍。 可存□的空□从 10 GB 到 20 GB，具有磁□机数据□。磁□机数据□ □元可以有 1 to 20 台磁□机，5 到 900 个磁□匣，磁□机数据□的容量可□ 50 GB 到 9 TB 。

如果要□□的□，DLT 型 IV 格式的磁□机最高可支持 70 GB 的存□ 容量。

数据存□在平行于磁□□行方向的磁道上（就像 QIC 磁□），一次写入□个 磁道。□写□的寿命相当□，□当磁□停止前□，磁□与□写□之□没有相□□□。

19.9.5. AIT

AIT 是 Sony □□的一□新格式，□个磁□最高可以存□ 50 GB。磁□ 机使用内存芯片来保存磁□上的索引内容。□个索引能□被磁□机□□器快速□□ 来搜索磁□机上文件所□的位置，而不像其他的磁□机需要花几分□的□□才能□到文件。像 SAMS:Alexandria □□的□件：能□操 作四十或者更多的 AIT 磁□□，直接使用内存芯片来□行通信把内容□示在屏幕上， 以决定把什□文件□□到□个磁□上，加□和恢□数据。

像□□的□成本大概在 $20,000 美元左右，零售市□可能□要□一点。

19.9.6. 第一次使用新的磁□机

当在一□完全空白的磁□上□□定入数据□，会得到□似下面□□的□□信息：

```
sa0(ncr1:4:0): NOT READY asc:4,1
sa0(ncr1:4:0):  Logical unit is in process of becoming ready
```

信息指出□□磁□没有□□号 (block □号□ 0)。在 QIC-525 之后的所有 QIC 磁□，都采用 QIC-525 □准，必□写入一个 Identifier Block 。□于□□□□， 有以下□□解决的□法：

- 用mt fsf 1 可以□磁□机□磁□写入 Identifier Block 。
- 使用面板上的按□磁□。

  再□入一次，并存□ dump 数据到磁□上。

  □□dump 将□回 DUMP: End of tape detected ，然后□会得到□□的□□信息： HARDWARE FAILURE info:280 asc:80,96。

  □□用 mt rewind 来倒□磁□。

  磁□操作的后□操作就完成了。

19.10. 用□□□□

19.10.1. 能□使用□□来□□数据□

□磁□通常是用来□□的□□中不太合□的□□：

- □□□□不太可□，特□是□期使用。
- □□和恢□都很慢
- 它□只有非常有限的存□容量。

然而，如果没有其它的□□数据的方法，那□□□□□比没有□□要好。

如果必□使用□□的□，必□□保□片的□量。□□在□公室中使用已□有□多 年了。最好使用一些名牌厂商的□品以□保□量。

19.10.2. 如何□□数据到□□

最好的□□数据到□□的方法是使用 tar(1) 程序加上 -M □□， 它可以允□数据□□到多□□□上。

要□□当前目□中所有的文件可以使用□个命令 (需要有 root□限)：

```
# tar Mcvf /dev/fd0 *
```

当第一□□□的□候， tar(1) 会指示□□入下一□□，□入第二□□之后就按回□。

```
Prepare volume 2 for /dev/fd0 and hit return:
```

□个□□可能需要重□很多次，直到□些文件□□完成□止。

19.10.3. 可以□□□□□

不幸的是，tar(1) 在□多卷文件作□□□是不允□使用 -z □□的。当然，可以用 gzip(1) □□所有的文件，把它□打包到磁□，以后在用 gunzip(1) 解□。

19.10.4. 如何恢□□□

要恢□所有文件：

```
# tar Mxvf /dev/fd0
```

有□□方法来恢□□□中的个□文件。首先，就要用第一□□□□：

```
# tar Mxvf /dev/fd0 filename
```

tar(1) 程序会提示□□入后面的□□，直到它□到所需要的文件。

如果□知道□个文件在□个□上，□就可以□入那□□，然后使用上同同□的命令。 如果□□上的第一个文件与前面的文件是□□的，那 tar(1) 命令会警告□它无法 恢□，即使□不要求它□□做。

19.11. □□策略

□□□□□□的第一要□是□□以下□□皆已考□到：

- 磁□故障
- 文件的意外□除
- 随机的文件□□
- 机器完全□□ (例如火□)， 包括破坏全部在□□□。

□□上述的□个□□采用完全不同的技□来解决是完全可行的。 除了只包含少量几乎没有价□数据的个人系□之外，一般来□很少有一□技□能□同□兼□前面所有的需要。

可以采用的技□包括：

- □整个系□的数据□行存□， □□到永久性的□□介□上。 □□方法□□上能□提供□□前面所有□□的保□， 但□□做通常很慢， 而且恢□□会比□麻□。 □可以将□□置于近□或在□的状□， 然而恢□文件仍然是一个□□， 特□是□没有特□的那些用□而言。
- 文件系□快照。 □□技□□□上只□无意中□除文件□一□情况有用， 但在□□情况下它会提供 非常大 的□助，而且□□迅速， 操作容易。
- 直接□制整个文件系□和/或磁□ (例如周期性地□整个机器做 rsync(1))。 通常□□于在网□上的□一需求最

□□用。 要□磁□故障提供更□通用的保□， 通常□□方法要□于 RAID。 □于恢□无意中□除的文件来□，□□方法基本上与 UFS 快照属于同一□次， 使用□一个取决于□的喜好。

- RAID。 它能□最大限度地□少磁□故障□致的停机□□。 其代价是需要□理更□□繁的磁□故障 (因□磁□的数量□加了)， 尽管□□故障不再需要作□非常□急的事□来□理。
- □□文件的指□。 mtree(8) 工具□于□□操作非常有用。 尽管□并不是一□□□的技□， 但它能□□保□有机会注意到那些□需要求助于□□□□的事情。 □□于□□□□非常重要， 而且□有□□地加以□□。

很容易列□更多的技□， 它□中有□多□□上是前面所列出的方法的□□。 特□的需求通常会需要采用特□的技□(例如， □□在□□行的数据□， 往往需要数据□□件提供某□方法来完成中□□□) 来□足。 最重要的事情是，一定要了解需要将数据保□起来免受何□□□， 以及□生□□□□□如何□理。

19.12. □□程序

有三个主要的□□程序 dump(8)、tar(1) 和 cpio(1)。

19.12.1. Dump 和 Restore

dump 和 restore 是 UNIX® □□的□□程序。 它以 block 而不是以文件□□位来□□数据、□接或目□。 dump □□的是□□上的整个文件系□， 不能只□□一个文件系□的部分或是用到□个以上文件系□的目□□。 与其他□□□件不同的是， dump 不会写文件和目□到磁□机， 而是写入包含文件 和目□的原始数据□。 当需要恢□数据的□候，restore 默□在 /tmp/ 下保存□□数据 - 如果□正在操作的恢□□只有比□小的 /tmp 的□，□可能需要把□境□量 TMPDIR □置到一个有更多空□的目□， 使得此□程更容易成功。

如果在□的 root 目□使用 dump， 将不需要□□ /home、/usr 或其他目□， 因□□些是典型的其他文件系□或符号□接到那些文件系□的加□点。

dump 是最早出□于 AT&T UNIX 的 Version 6 (□ 1975)。 默□的参数□用于 9-track 磁□(6250 bpi)，所以如果要用高密度的磁□（最高可□ 62,182 ftpi）， 就不能用默□的参数， 而要□外指定参数。 □些默□□必□在命令行被修改以更好地利用当前磁□机的功能。

rdump 和 rrestore 可以通□网□在□一台□算机的磁□机上□□数据。 □□个程序都是依□ rcmd(3) 和 ruserok(3) 来□□□程的磁□机。 因此，□行□□的用□必□要有□程主机的 .rhosts □□□。 rdump 和 rrestore 的参数必□□用于□程主机 例如，当□从 FreeBSD □到一台 SUN 工作站 knomodo 去使用磁□机□，使用：

```
# /sbin/rdump 0dsbfu 54000 13000 126 komodo:/dev/nsa8 /dev/da0a 2>&1
```

要注意的是：必□□□□在使用 .rhosts □的安全情况。

也可以通□使用 ssh 用一个更安全的方式来使用 dump 和 restore 。

例 *24.* 通□ *ssh* 使用 dump

```
# /sbin/dump -0uan -f - /usr | gzip -2 | ssh -c blowfish \
          targetuser@targetmachine.example.com dd of=/mybigfiles/dump-usr-l0.gz
```

或使用 dump 的 built-in 方法，□置□境□量 RSH：

例 25. 通□□置 *ssh* □境□量 RSH 使用 dump

```
# RSH=/usr/bin/ssh /sbin/dump -0uan -f
targetuser@targetmachine.example.com:/dev/sa0 /usr
```

19.12.2. tar

tar(1) 也同□是在第 6 版 AT&T UNIX (大□是 1975 前后) 出□的。 tar □文件系□直接操作；其作用是把文件和目□写入磁□。 tar 并不支持 cpio(1) 所提供的全部功能， 但也不需要 cpio 所需要使用的□□的命令行管道。

要 tar 到□接在名□ komodo 的 Sun 机器上的 Exabyte 磁□机， 可以使用：

```
# tar cf - . | rsh komodo dd of=tape-device obs=20b
```

如果□担心通□网□□□会有安全□□，□当使用 ssh， 而不是 rsh。

19.12.3. cpio

cpio(1) 是 UNIX® 最早用来作文件交□的磁□机程序。它有□行字□ 交□的□□，可以用几□不同的格式写入，并且可以将数据用管道□□其他程序。 cpio 没□法自□□□目□□内的文件列表，必□通□□准 □入 stdin 来指定。

cpio 不支持通□网□的□□方式。可以使用 pipeline 和 rsh 来□送数据□□程的磁□机。

```
# for f in directory_list; do
find $f >> backup.list
done
# cpio -v -o --format=newc < backup.list | ssh user@host "cat > backup_device"
```

□里的 *directory_list* 是要□□的目□列表， *user@host* □合了将 要□行□□的用□名和主机名，*backup_device* 是写 入□□的□□（如 /dev/nsa0）。

19.12.4. pax

pax(1) 是符合 IEEE/POSIX® □准的程序。多年来各□不同版本 的 tar 和 cpio □有些不兼容。 □了防止□□情况，并使其□准化，POSIX® 出了□套新的工具程序。 pax □□可以□写各□ cpio 和 tar 的格式，并可以自己□加新的格式。它的命令 集比 tar 更接近 cpio。

19.12.5. Amanda

Amanda (Advanced Maryland Network Disk Archiver) 并非□一的程序，而是一个客□机/服□器模式的□□系□ 。一台 Amanda 服□器可以□□任意数量□行 Amanda 的客□机或是将□上 Amanda 服□器的

□算机上的数据□□到一台磁□机上。一个常□的□□是，数据写入磁□机的□□将超 □取行数据的□□，而 Amanda 解决了□个□□。它使用一个 "holding disk" 来同□□□几个文件系□。 Amanda 建立 "archive sets" 的一□磁□，用来□□在 Amanda 的配置文件中所列出的完整的文件系□。

Amanda 配置文件提供完整的□□控制及 Amanda □生的网□□□。 Amanda 可以使用上述任何一个□□程序来向磁□写入数据。Amanda 可以从 port 或 package 取得，它并非系□默□安装的。

19.12.6. Do Nothing □□策略

"Do nothing" 不是一个程序，而是被广泛使用的□□策略。 不需要□算，不需要□□的□□表，全部都不用。如果□的数据□生了什□□□， 忽略它！

如果□的□□和数据不□得□做□些事，那□ "Do nothing" 将是最好的□□程序。要注意的是，UNIX® 是相当好用的工具，□可能在几个月 内，就□□□已□收集了不少□□来□相当具有价□的文件和程序。

"Do nothing" □于像 /usr/obj 和其他 可由□的□算机□生的文件来□，是最好的方法。例如□本手册包含有 HTML 或 PostScript® 格式的文件。□些文□格式是从 SGML □入文件□建的。□建 HTML 或 PostScript® 格式的文件的□□就没有必要了。只要□常□□ SGML 文件就□了。

19.12.7. □个□□程序最好？

在dump(8) □期 Elizabeth D. Zwicky □□了所有以上列出的□□程序。在各□各□怪□的文件系□中， dump 是□明智的□□。Elizabeth 建立起各□各□、 奇怪或常□的文件系□，并用各□□□程序，□□在各□文件系□上□□ 及恢□数据。□些怪□之□包括：具有 holes 和一个 nulls block 的文件， 文件名具有有趣字符，无法□写的文件及□□，在□□□改□文件大小，在 □□□建立或□除的文件。□将□果写在： LISA V in Oct. 1991. 参□ torture-testing Backup and Archive Programs.

19.12.8. □急恢□程序

19.12.8.1. 在出□□□前

在遇到□□前，只需要□行以下四个□□：

第一，打出□的□个磁□□□器的磁□□□ (例如： `bsdlabel da0 | lpr`)，文件系□表， (/etc/fstab) ，以及所有□□信息， 并将其□制□□。

第二， 刻□一□ "livefs" CDROM。 □个 CDROM 包含了用于引□□入 FreeBSD "livefs" 修□模式的支持，□□模式允□用□□行□多任□， 例如□行 dump(8)、 restore(8)、 fdisk(8)、 bsdlabel(8)、 newfs(8)、 mount(8)， 等等。 Livefs CD 映像文件随 FreeBSD/i386 12.0-RELEASE 提供， 可以从 ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/ISO-IMAGES/12.0/FreeBSD-12.0-RELEASE-i386-livefs.iso □得。

第三，定期将数据□□到磁□。任何在上次□□后的改□都无法恢□。□得将 磁□写保□。

第四， □□在第二□所建立的 "livefs" CDROM 及□□的磁□。 写下□□， 并和□□ CDROM、 打印副本以及磁□放在一起。 □在需要恢□数据□可能正心慌意乱， 而□些□□可能会□助□避免□掉□□磁□ （□□会□生□□情况□？□例来□， 本□□行 `tar xvf /dev/sa0` 命令□， □可能会不小心□入 `tar cvf /dev/sa0`， 从而覆□□□磁□）。

保□起□， □可以制作□□ "livefs" CDROM 和□□磁□。 其中一□□放到其它地方， □里□的其他地方当然不是指同一□□公楼的地下室， 世□中心的一大批公司已□学到了血的教□。 保存□□□□

的位置□□与□的□算机和磁□□□器越□越好。

19.12.8.2. 出□□□后

□□□□是： □的硬件是否幸免于□？ 由于已□做好了定期的□□工作， 因此并不需要担心□件的□□。

如果硬件已□□坏， □些部分□□在□□使用□算机之前□掉。

如果硬件□能用， 将 "livefs" CDROM □入 CDROM □□器并引□系□。 □将看到最初安装系□□的菜□。 □□正□的国家之后， □□ Fixit — Repair mode with CDROM/DVD/floppy or start a shell □□， 然后再□□ CDROM/DVD — Use the live filesystem CDROM/DVD □□。 □可以使用 `restore` 以及其他位于 /mnt2/rescue 的工具。

分□恢□□一个文件系□

□着 `mount` （例如： `mount /dev/da0a /mnt`） 第一个磁□上的 root 分区。 如果 bsdlabel 已□□坏， □需要使用 `bsdlabel` 根据□先前打印存□的□□来重新分区并分配磁□□□。 接着使用 `newfs` 重建文件系□。 以□写方式重新挂□磁□的根分区 (`mount -u -o rw /mnt`)。 使用□的□□程序以及□□磁□恢□文件系□数据 （例如 `restore vrf /dev/sa0`）。 最后卸下文件系□ （例如 `umount /mnt`）。 □于□掉的其他文件系□， 重□□行前面□些操作。

当□的系□正常□□后， 将□的数据□□到新的磁□。 任何造成数据□失的的□□都可能再次□生。 □在花一些□□， 也□可以在下次□生□□□救□一把。

19.13. 网□、内存和 和以及映像文件□介□的虚□文件系□

除了□在□□算机上的物理磁□： □□、 CD、 硬□□□器， 等等之外， FreeBSD □能□□一些其他的磁□形式 - 虚□磁□。

□□包括， 如 网□文件系□ (Network File System) 和 Coda一□的网□文件系□、 内存以及映像文件□介□的虚□文件系□。

随□行的 FreeBSD 版本不同， 用来□建和使用以映像文件介□文件系□和内存文件系□的工具也不尽相同。

系□会使用 devfs(5) 来□建□□□点， □□用□来□是透明的。

19.13.1. 以映像文件□介□的文件系□

在 FreeBSD 系□中， 可以用 mdconfig(8) 程序来配置和□用内存磁□， md(4)。 要使用 mdconfig(8)， 就需要在内核配置文件中添加 md(4) 模□来支持它：

```
device md
```

mdconfig(8) 命令支持三□□型的虚□文件系□： 使用 malloc(9)，来分配内存文件系□，内存文件系□作□文件或作□ □用的交□分区。一□使用方式是在文件中来挂□一个□□和 CD □像。

将一个□存的映像文件作□文件系□挂□：

例 26. 使用 mdconfig 挂□已□存在的映像文件

```
# mdconfig -a -t vnode -f diskimage -u 0
# mount /dev/md0 /mnt
```

使用 mdconfig(8) 来□建新的映像文件:

例 27. 使用 mdconfig 将映像文件作□文件系□挂□

```
# dd if=/dev/zero of=newimage bs=1k count=5k
5120+0 records in
5120+0 records out
# mdconfig -a -t vnode -f newimage -u 0
# bsdlabel -w md0 auto
# newfs md0a
/dev/md0a: 5.0MB (10224 sectors) block size 16384, fragment size 2048
        using 4 cylinder groups of 1.25MB, 80 blks, 192 inodes.
super-block backups (for fsck -b #) at:
 160, 2720, 5280, 7840
# mount /dev/md0a /mnt
# df /mnt
Filesystem 1K-blocks Used Avail Capacity  Mounted on
/dev/md0a       4710    4  4330     0%    /mnt
```

如果没有通□ -u □□指定一个□□号 mdconfig(8) 将使用 md(4) □它自□□□一个未用的□□□□号。 分配□它的□□名将被□出到□准□出□□， 其形式是与 md4 □似。 如果希望了解更多相□信息， □参□□机手册 mdconfig(8)。

mdconfig(8) 功能很□大， 但在将映像文件作□文件系□挂□□， 仍需使用□多行的命令。 □此 FreeBSD 也提供了一个名□ mdmfs(8) 的工具， □程序使用 mdconfig(8) 来配置 md(4) □□， 并用 newfs(8) 在其上□建 UFS 文件系□， 然后用 mount(8) 来完成挂□操作。 例如， 如果想□建和挂接像上面那□的文件系□映像，只需□□地□行下面的□□：

例 28. 使用 mdmfs 命令配置和挂□一个映像文件□文件系□

```
# dd if=/dev/zero of=newimage bs=1k count=5k
5120+0 records in
5120+0 records out
# mdmfs -F newimage -s 5m md0 /mnt
# df /mnt
Filesystem 1K-blocks Used Avail Capacity  Mounted on
/dev/md0        4718    4  4338     0%    /mnt
```

如果□使用没有加□□号的 md □□， mdmfs(8) 将使用 md(4) 的自□□示号特性来自□□其 □□一个未使用的□□。更□□的 mdmfs(8)，□参考□机手册。

19.13.2. 以内存□介□的文件系□

一般来□， 在建立以内存□介□的文件系□□， □使用 "交□区作□介□ (swap backing)"。 使用交□区作□介□，并不意味着内存□将被无条件地□出到交□区， 它只是表示将根据需要从可□出的内存池中分配内存。 此外，也可以使用 malloc(9) □建以内存作□介□的文件系□。 不□在内存不足□， □□方式可能引致系□崩□。

例 *29.* 用 mdconfig □建新的内存□□□

```
# mdconfig -a -t swap -s 5m -u 1
# newfs -U md1
/dev/md1: 5.0MB (10240 sectors) block size 16384, fragment size 2048
        using 4 cylinder groups of 1.27MB, 81 blks, 192 inodes.
        with soft updates
super-block backups (for fsck -b #) at:
 160, 2752, 5344, 7936
# mount /dev/md1 /mnt
# df /mnt
Filesystem 1K-blocks Used Avail Capacity  Mounted on
/dev/md1        4718    4  4338     0%    /mnt
```

例 *30.* 使用 mdmfs 来新建内存介□文件系□

```
# mdmfs -s 5m md2 /mnt
# df /mnt
Filesystem 1K-blocks Used Avail Capacity  Mounted on
/dev/md2        4846    2  4458     0%    /mnt
```

19.13.3. 从系□中移除内存□□□

当不再使用内存□□□□， □将其□源□放回系□。 第一□操作是卸下文件系□， 然后使用 mdconfig(8) 把虚□磁□从系□中分□， 以□放□源。

例如， 要分□并□放所有 /dev/md4 使用的□源， □使用命令：

```
# mdconfig -d -u 4
```

mdconfig -l 命令可以列出□于配置 md(4) □□的信息。

19.14. 文件系□快照

FreeBSD 提供了一个和 Soft Updates □□的新功能: 文件系□快照

快照允□用□□建指定文件系□的映像，并把它□当做一个文件来□待。 快照文件必□在文件系□正在使用□□建，一个用□□□个文件系□□建的 快照不能大于20个。活□的快照文件被□□在超□□中，所以它□可以在系□

□□的□候一□□行挂接后摘掉。当一个快照不再需要□，可以使用□准的 rm(1) 使用来使其□除。快照可以以任何□序□行移除，但所有使用 的快照不可能同□□行移除，因□其它的快照将有可能互相引用一些□。

不可改的 snapshot 文件□志， 是由 mksnap_ffs(8) 在完成□建快照文件□□置的。 unlink(1) 命令是一个特例， 以允□□除快照文件。

快照可以通□ mount(8) 命令□建。 将文件系□ /var 的快照放到 /var/snapshot/snap 可以使用下面的命令：

```
# mount -u -o snapshot /var/snapshot/snap /var
```

作□□□，□也可以使用 mksnap_ffs(8) 来□建一个快照：

```
# mksnap_ffs /var /var/snapshot/snap
```

可以□□文件系□中的快照文件 (例如 /var)， 方法是使用 find(1) 命令：

```
# find /var -flags snapshot
```

当快照文件被□建好后，可以用于下面一些目的：

- 有些管理□用文件快照来□行□□， 因□快照可以被□移到 CD 或磁□上。
- 文件系□一致性□□程序 fsck(8) 可以用来□□快照文件。 如果文件系□在挂接前是一致的，□□□□果也一定是一致的 (也就是不会做任何修改)。 □□上□也正是后台 fsck(8) 的操作□程。
- 在快照上□行 dump(8) 程序。 dump 将返回包含文件系□和快照的□□□。dump(8) 也能□□取快照，使用 -L □志可以首先□建快照， 完成 dump 映像之后再自□□除它。
- 用 mount(8) 来挂接快照作□文件系□的一个□□的□像。 要 mount(8) 快照 /var/snapshot/snap □行：

  ```
  # mdconfig -a -t vnode -f /var/snapshot/snap -u 4
  # mount -r /dev/md4 /mnt
  ```

□在□就可以看到挂接在 /mnt 目□下的 /var 文件系□的快照。 □一□□西都保存的像它□建□的状□一□。唯一例外的是更早的快照文件将表□□□度□ 0 的文件。 用完快照文件之后可以把它卸下，使用：

```
# umount /mnt
# mdconfig -d -u 4
```

想了解更多□于 softupdates 和 文件系□快照的信息， 包括技□□明， 可以□□ Marshall Kirk McKusick 的 WWW 站点 http://www.mckusick.com/。

19.15. 文件系□配□

配□是操作系□的一个可□的功能， 它允□管理□以文件系□□□元， 限制分派□用□或□成□所使用的磁□空□大小或是使用的□文件数量。 □□常被用于那些分□操作的系□上， □于□些系□而言， 通常希望限制分派到□一个用□或□的□源□量， 从而可以防止某个用□占用所有可用的磁□空□。

19.15.1. 配置系□来□用磁□配□

在决定使用磁□配□前，□信磁□配□已□在内核中配置好了。只要在在内核 中配置文件中添加下面一行就行了：

```
options QUOTA
```

在默□情况下 GENERIC 内核是不会□用□个功能的， 所以必□配置、重建和安装一个定制的内核。□参考 FreeBSD 内核配置 配置FreeBSD的内核 □章了解更多有□内核配置的信息。

接下来，需要在 /etc/rc.conf 中□用磁□配□。可以 通□添加下面□行来完成：

```
enable_quotas="YES"
```

□了更好的控制配□□的□□，□有□外一个可配置的□量。通常 □□□，集成在□个文件系□上的配□会被配□□□程序 quotacheck(8) 自□□□。配□□□功能能□□保在配□数据□中 的数据正□地反映了文件系□的数 据情况。□是一个很耗□□的□理□程，它会影□系□的□□□□。如果 想跳□□一□，可以在文件 /etc/rc.conf 加入 下面□一行来□到目的：

```
check_quotas="NO"
```

最后，要□□ /etc/fstab 文件，以在□一个 文件系□基□上□用磁□配□。□是□用用□和□配□，或同□□用用□ 和□配□的地方。

要在一个文件系□上□用□个用□的配□，可以在 /etc/fstab 里添加 userquota □□在要雇用配□文件的系□上。例如：

```
/dev/da1s2g   /home    ufs rw,userquota 1 2
```

同□的，要□用□配□，使用 groupquota □□来代替 userquota □□。要同□□用用□和□配□，可以□□做：

```
/dev/da1s2g    /home    ufs rw,userquota,groupquota 1 2
```

默□情况下，配□文件是存放在文件系□的以 quota.user 和 quota.group 命名的根目□下。可以□看 fstab(5) □机手册了解更多信息。 尽管□机手册 fstab(5) 提到， 可以□配□文件指定其他的位置， 但并不推□□□做， 因□不同的配□工具并不一定遵循此□□。

到□儿，可以用新内核重新□□系□。 /etc/rc 将自□ □行□当的命令来□建最初的配□文件，所以并不需要手□来□建任何零□度的配□ 文件。

在通常的操作□程中，并不要求手□□行 quotacheck(8)、 quotaon(8)，或 quotaoff(8) 命令，然而可能需要□□与他□的操作 相似的□机手册。

19.15.2. □置配□限制

一旦□配置好了□用配□的系□，可以□□一下它□是真的有用。 可以□□做：

```
# quota -v
```

□□□能□看到一行当前正在使用的□个文件系□□用的磁□配□ 使用情况的摘要信息。

□在可以使用 edquota(8) 命令准□□用配□限制。

有几个有□如何□制限制用□或□可以分配到的磁□空□大小的□□。 □可以限制磁□存□□的配□， 或文件的数量，甚至同□限制□者。 □些限制最□可分□□□： 硬限制和□限制。

硬性限制是一□不能越□的限制。 一旦用□□到了系□指定的硬性限制， 他就无法在□□的文件系□分配到更多的□源。 例如， 如果文件系□上分□用□的硬性限制是 500 KB， 而□在已□用掉了 490 KB， 那□□个用□最多□能再分配 10 KB 的空□。 □言之， 如果□□□□再分配 11 KB， □会失□。

而与此相反， □性限制在一段□□内是允□越□的。 □段□□也称□□限期， 其默□□是一周。 如果一个用□延□□□太□的□，□限制将会□成硬限制， 而□□分配磁□空□的操作将被拒□。 当用□占用的空□回到□性限制□以下□，□限期将重新□始□算。

下面是一个□行 edquota(8) □看到的例子。当 edquota(8) 命令被□用□，会被□移□ EDITOR □境□量指派的□□器中，允□□□配□限制。如果□境□量没有□置，默□在 vi □□器上□行。

```
# edquota -u test
```

```
Quotas for user test:
/usr: kbytes in use: 65, limits (soft = 50, hard = 75)
        inodes in use: 7, limits (soft = 50, hard = 60)
/usr/var: kbytes in use: 0, limits (soft = 50, hard = 75)
        inodes in use: 0, limits (soft = 50, hard = 60)
```

在□一个□用了磁□配□的文件系□上，通常会看到□行。一行是 block 限制，□一行是 inode 限制。□□地改□要修改的配□限制的□。 例如，提高□个用□□限制的数□到 500 ，硬限制到 600 ：

```
/usr: kbytes in use: 65, limits (soft = 50, hard = 75)
```

to:

```
/usr: kbytes in use: 65, limits (soft = 500, hard = 600)
```

当□□□□器的□候，新的配□限制□置将会被保存。

有□，在 UIDs 的□□上□置配□限制是非常必要的。□可以通□在 edquota(8) 命令后面加上 -p □□来完成。首先，□用□分配所需要的配□限制，然后□行命令 edquota -p protouser startuid-enduid。例如，如果 用□ test 已□有了所需要的配□限制，下面的命令 可以被用来□制那些 UIDs □10,000 到 19,999 的配□限制：

```
# edquota -p test 10000-19999
```

更多□□□参考 edquota(8) □机手册。

19.15.3. □□配□限制和磁□使用

既可以使用 quota(1) 也可以使用 repquota(8) 命令来□□ 配□限制和磁□使用情况。 quota(1) 命令能□□□□个用□和□的配置 使用情况。只有超□用□才可以□□其它用□的配□和磁□使用情况。 repquota(8) 命令可以用来了解所有配□和磁□的使用情况。

下面是一个使用 quota -v 命令后的□出情况：

```
Disk quotas for user test (uid 1002):
     Filesystem  usage    quota   limit   grace   files   quota   limit   grace
           /usr      65*     50      75   5days       7      50      60
       /usr/var       0      50      75               0      50      60
```

前面以 /usr 作□例子。 此用□目前已□比□限制 50 KB 超出了 15 KB， □剩下 5 天的□限期。 □注意， 星号 * □明用□已□超出了其配□限制。

通常， 如果用□没有使用文件系□上的磁□空□， 就不会在 quota(1) 命令的□出中□示， 即使已□□那个用□指定了配□。 而使用 -v □□□会□示它□， 例如前面例子中的 /usr/var。

19.15.4. 通□ NFS 使用磁□配□

配□能□在 NFS 服□器上被配□子系□□迫使用。在 NFS 客□端， rpc.rquotad(8) 命令可以使用 quota 信息用于 quota(1) 命令， 可以允□用□□看它□的 quota □□信息。

可以□□在 /etc/inetd.conf 中□用 rpc.rquotad：

```
rquotad/1      dgram rpc/udp wait root /usr/libexec/rpc.rquotad rpc.rquotad
```

□在重□ inetd：

```
# /etc/rc.d/inetd restart
```

19.16. 加密磁□分区

FreeBSD 提供了□好的数据保□措施，防止未受□的数据□□。 文件□限和□制□□控制(MAC)(看 □制□□控制) 可以□助□防在操作系□□于□行状□和□算机加□□未受□的第三方□□数据。 但是，和操作系□□制受□不相□的是，如果□客有物理上□□□算机的可能， 那他就可以□□的把□算机的硬件安装到□一个系□上□制出敏感的数据。

无□攻□者如何取得停机后的硬件或硬□□□器本身， FreeBSD GEOM Based Disk Encryption (基于 GEOM 的磁□加密， gbde) 和 geli 加密子系□都能□保□□算机上的文件系□数据， 使它□免受□怕是□□有素的攻□者□得有用的□源。 与那些只能加密□个文件的□重的加密方法不同， gbde 和 geli 能□透明地加密整个文件系□。 明文数据不会出□在硬□的任何地方。

19.16.1. 使用 gbde □磁□□行加密

1. 成□ root

   配置 gbde 需要超□用□的□力。

   ```
   % su -
   Password:
   ```

2. 在内核配置文件中添加□ gbde(4) 的支持

   在□的内核配置中加入下面一行：

   ```
   options GEOM_BDE
   ```

   按照 配置FreeBSD的内核 所□行的介□重新□□并安装内核。

   重新引□□入新的内核。

3. □一□无需重新□□内核的方法， 是使用 kldload 来加□ gbde(4)：

   ```
   # kldload geom_bde
   ```

19.16.1.1. 准□加密□

下面□个例子假□□添加了一个新的硬□在□的系□并将□有一个□独的加密分区。 □个分区将挂接在 /private目□下。 gbde 也可以用来加密 /home 和 /var/mail， 但是□需要更多的□□命令来□行。

1. 添加新的硬□

   添加新的硬□到系□中可以□看在 添加磁□ 中的□明。 □个例子的目的是□明一个新的硬□分区已□添加到系□中如： /dev/ad4s1c。在例子中 /dev/ad0s1* □□代表系□中存在的□准 FreeBSD 分区。

   ```
   # ls /dev/ad*
   /dev/ad0        /dev/ad0s1b     /dev/ad0s1e     /dev/ad4s1
   /dev/ad0s1      /dev/ad0s1c     /dev/ad0s1f     /dev/ad4s1c
   /dev/ad0s1a     /dev/ad0s1d     /dev/ad4
   ```

2. □建一个目□来保存 gbde Lock 文件

   ```
   # mkdir /etc/gbde
   ```

   gbde lock 文件包含了 gbde 需要□□的加密分区的信息。 没有 lock 文件， gbde 将不能解密包含在加密分区上的数据。 □个加密分区使用一个独立的 lock 文件。

3. 初始化 gbde 分区

   一个 gbde 分区在使用前必□被初始化， □个初始化□程只需要□行一次：

   ```
   # gbde init /dev/ad4s1c -i -L /etc/gbde/ad4s1c.lock
   ```

   gbde(8) 将打□□的□□器， 提示□去□置在一个模板文件中的配置□量。 使用 UFS1 或 UFS2， □置扇区大小□ 2048：

   ```
   $FreeBSD: src/sbin/gbde/template.txt,v 1.1 2002/10/20 11:16:13 phk Exp $
   #
   # Sector size is the smallest unit of data which can be read or written.
   # Making it too small decreases performance and decreases available space.
   # Making it too large may prevent filesystems from working.  512 is the
   # minimum and always safe.  For UFS, use the fragment size
   #
   sector_size     =       2048
   [...]
   ```

   gbde(8) 将□□□入□次用来加密数据的密□短□。 □次□入的密□必□相同。 gbde 保□□数据的能力依□□□□□入的密□的□量。

   gbde init 命令□□的 gbde 分区□建了一个 lock 文件， 在□个例子中存□在 /etc/gbde/ad4s1c.lock中。 gbde lock 文件必□使用 ".lock" □展名才能□被 /etc/rc.d/gbde □□脚本正□□□。

gbde lock 文件 必□ 和加密分区上的内容同□□□。 如果□生只有 lock 文件遭到□除的情况□， 就没有□法□定 gbde 分区上的数据是否是解密□的。 □外，如果没有 lock 文件， 即使磁□的合法主人， 不□□大量□致的工作也无法□□加密分区上的数据， 而□是在□□ gbde(8) □完全没有考□□的。

4. 把加密分区和内核□行□□

```
# gbde attach /dev/ad4s1c -l /etc/gbde/ad4s1c.lock
```

在加密分区的初始化□程中□将被要求提供一个密□短□。 新的加密□□将在 /dev 中□示□ /dev/device_name.bde：

```
# ls /dev/ad*
/dev/ad0        /dev/ad0s1b     /dev/ad0s1e     /dev/ad4s1
/dev/ad0s1      /dev/ad0s1c     /dev/ad0s1f     /dev/ad4s1c
/dev/ad0s1a     /dev/ad0s1d     /dev/ad4        /dev/ad4s1c.bde
```

5. 在加密□□上□建文件系□

当加密□□和内核□行□□后， □就可以使用 newfs(8) 在此□□上□建文件系□， 使用 newfs(8) 来初始化一个 UFS2 文件系□比初始化一个 UFS1 文件系□□要快，摧□使用 -O2 □□。

```
# newfs -U -O2 /dev/ad4s1c.bde
```

newfs(8) 命令必□在一个 gbde 分区上□行， □个分区通□一个存在的 *.bde □□名□行□□。

6. 挂接加密分区

□加密文件系□□建一个挂接点。

```
# mkdir /private
```

挂接加密文件系□。

```
# mount /dev/ad4s1c.bde /private
```

7. 校□加密文件系□是否有效

加密的文件系□□在□于 df(1) □□可□并可以使用。

```
% df -H
Filesystem        Size   Used  Avail Capacity  Mounted on
/dev/ad0s1a      1037M    72M   883M     8%    /
/devfs            1.0K   1.0K     0B   100%    /dev
/dev/ad0s1f       8.1G    55K   7.5G     0%    /home
/dev/ad0s1e      1037M   1.1M   953M     0%    /tmp
/dev/ad0s1d       6.1G   1.9G   3.7G    35%    /usr
/dev/ad4s1c.bde   150G   4.1K   138G     0%    /private
```

19.16.1.2. 挂接已有的加密文件系□

□次系□□□后， 在使用加密文件系□前必□和内核重新□行□□， 校□□□和再次挂接。使用的命令必□由 root用□来□行。

1. □□ gbde 分区到内核

   ```
   # gbde attach /dev/ad4s1c -l /etc/gbde/ad4s1c.lock
   ```

   接下来系□将提示□□入在初始化加密的 gbde 分区□所用的密□短□。

2. 校□文件系□□□

   加密文件系□不能列在 /etc/fstab 文件中□行自□加□， 在加□前必□手□□行 fsck(8) 命令□文件系□□行□□□□。

   ```
   # fsck -p -t ffs /dev/ad4s1c.bde
   ```

3. 挂接加密文件系□

   ```
   # mount /dev/ad4s1c.bde /private
   ```

   加密后的文件系□□在可以有效使用。

19.16.1.2.1. 自□挂接加密分区

可以□建脚本来自□地附加、 □□， 并挂接加密分区， 然而， □于安全考□， □个脚本不□包含 gbde(8) 密□。因而， 我□建□□□脚本在控制台或通□ ssh(1) □行并要求用□□入口令。

除此之外， 系□□提供了一个 rc.d 脚本。 □个脚本的参数可以通□ rc.conf(5) 来指定， 例如：

```
gbde_autoattach_all="YES"
gbde_devices="ad4s1c"
gbde_lockdir="/etc/gbde"
```

在□□□将要求□入 gbde 的口令。 在□入正□的口令之后， gbde 加密分区将被自□挂接。 □于将 gbde 用在□□本□□上□， □就很有用了。

19.16.1.3. gbde 提供的密□学保□

gbde(8) 采用 CBC 模式的 128-位 AES 来加密扇区数据。 磁□上的□个扇区都采用不同的 AES 密□来加密。要了解□于 gbde 的密□学□□， 包括扇区密□如何从用□提供的口令字中生成等□□， □参考 gbde(4)。

19.16.1.4. 兼容性□□

sysinstall(8) 是和 gbde 加密□□不兼容的。 在□□ sysinstall(8) □必□将 *.bde □□和内核□行分□，否□在初始化探□□□□将引起冲突。 与加密□□□行分□在我□的例子中使用如下的命令：

```
# gbde detach /dev/ad4s1c
```

□需要注意的是， 由于 vinum(4) 没有使用 geom(4) 子系□， 因此不能同□使用 gbde 与 vinum 卷。

19.16.2. 使用 geli □磁□□行加密

□有□一个可用于加密的 GEOM class - geli。 它目前由 Paweł Jakub Dawidek <pjd@FreeBSD.org> □□。Geli 工具与 gbde 不同； 它提供了一些不同的功能， 并采用了不同的方式来□行密□学□算。

geli(8) 最重要的功能包括：

- 使用了 crypto(9) 框架 - 如果系□中有加解密硬件加速□□， □ geli 会自□加以利用。
- 支持多□加密算法 (目前支持 AES、 Blowfish， 以及 3DES)。
- 允□□根分区□行加密。 在系□□□□， 将要求□入用于加密根分区的口令。
- 允□使用□个不同的密□ (例如， 一个 "个人密□" 和一个 "公司密□")。
- geli 速度很快 - 它只□行□□的扇区到扇区的加密。
- 允□□□和恢□主密□。 当用□必□□□其密□□， 仍然可以通□从□□中恢□密□来存取数据。
- 允□使用随机的一次性密□来挂接磁□ - □□于交□区和□□文件系□非常有用。

更多 geli 功能介□可以在 geli(8) □机手册中□到。

下面的□□介□了如何□用 FreeBSD 内核中的 geli 支持， 并解□了如何□建新和使用 geli 加密 provider。

由于需要修改内核， □需要□有超□用□□限。

1. 在内核中加入 geli 支持

   在内核配置文件中加入下面□行：

   ```
   options GEOM_ELI
   device crypto
   ```

   按照 配置FreeBSD的内核 介□的□□重新□□并安装内核。

   □外， geli 也可以在系□引□□加□。 □是通□在 /boot/loader.conf 中□加下面的配置来□□的：

   ```
   geom_eli_load="YES"
   ```

   geli(8) □在□□已□□内核所支持了。

2. 生成主密□

   下面的例子□描述如何生成密□文件， 它将作□主密□ (Master Key) 的一部分， 用于挂接到 /private 的加密 provider。 □个密□文件将提供一些随机数据来加密主密□。 同□， 主密□也会使用一个口令字来保□。 Provider 的扇区尺寸□ 4kB。 此外， □里的□□将介□如何挂□ geli provider， 在其上□建文件系□， 如何挂接并在其上工作， 最后将其卸下。

   建□□使用□大的扇区尺寸 (例如 4kB)， 以□得更好的性能。

   主密□将由口令字保□， 而密□文件的数据来源□将是 /dev/random。 我□称之□ provider 的 /dev/da2.eli 的扇区尺寸将是 4kB。

   ```
   # dd if=/dev/random of=/root/da2.key bs=64 count=1
   # geli init -s 4096 -K /root/da2.key /dev/da2
   Enter new passphrase:
   Reenter new passphrase:
   ```

   同□使用口令字和密□文件并不是必□的； □也可以只使用其中的一□来加密主密□。

   如果密□文件写作 "-"， □表示使用□准□入。 下面是□于如何使用多个密□文件的例子：

   ```
   # cat keyfile1 keyfile2 keyfile3 | geli init -K - /dev/da2
   ```

3. 将 provider 与所生成的密□□□

   ```
   # geli attach -k /root/da2.key /dev/da2
   Enter passphrase:
   ```

   新的明文□□将被命名□ /dev/da2.eli。

```
# ls /dev/da2*
/dev/da2  /dev/da2.eli
```

4. □建新的文件系□

```
# dd if=/dev/random of=/dev/da2.eli bs=1m
# newfs /dev/da2.eli
# mount /dev/da2.eli /private
```

□在加密的文件系□□□已□可以被 df(1) 看到， 并□于可用状□了：

```
# df -H
Filesystem     Size   Used  Avail Capacity  Mounted on
/dev/ad0s1a    248M    89M   139M    38%    /
/devfs         1.0K   1.0K     0B   100%    /dev
/dev/ad0s1f    7.7G   2.3G   4.9G    32%    /usr
/dev/ad0s1d    989M   1.5M   909M     0%    /tmp
/dev/ad0s1e    3.9G   1.3G   2.3G    35%    /var
/dev/da2.eli   150G   4.1K   138G     0%    /private
```

5. 卸下卷并断□ provider

一旦在加密分区上的工作完成， 并且不再需要 /private 分区， 就□考□将其卸下并将 geli 加密分区从内核上断□。

```
# umount /private
# geli detach da2.eli
```

□于如何使用 geli(8) 的更多信息， 可以在其□机手册中□到。

19.16.2.1. 使用 geli rc.d 脚本

geli 提供了一个 rc.d 脚本， 它可以用于□化 geli 的使用。 通□ rc.conf(5) 配置 geli 的方法如下：

```
geli_devices="da2"
geli_da2_flags="-p -k /root/da2.key"
```

□将把 /dev/da2 配置□一个 geli provider， 其主密□文件位于 /root/da2.key， 而 geli 在□接 provider □将不使用口令字 (注意只有在 geli init □段使用了 -P 才可以□□做)。 系□将在□□之前将 geli provider 断□。

□于如何配置 rc.d 的□□信息可以在使用手册的 rc.d 一□中□到。

19.17. □交□区□行加密

FreeBSD 提供了易于配置的交□区加密机制。 随所用的 FreeBSD 版本， 可用的配置□□会有所不同，而配置方法也会有一些差□。 可以使用 gbde(8) 和 geli(8) □□加密系□来□行交□区的加密操作。 前面所□的□□□加密系□， 都用到了 encswap □个 rc.d 脚本。

在前面的小□ 如何加密磁□分区 中， 已□就不同的加密系□之□的区□□行了□□的□□。

19.17.1. □什□需要□交□区□行加密？

与加密磁□分区□似， 加密交□区有助于保□敏感信息。 □此， 我□不妨考□一个需要□理敏感信息的程序，例如， 它需要□理口令。 如果□些口令一直保持在物理内存中， □一切相安无事。 然而， 如果操作系□□始将内存□□出到交□区， 以便□其他□用程序□出内存□， □些口令就可能以未加密的形式写到磁□上， 并□攻□者所□易□得。 加密交□区能□有效地解决□□□□。

19.17.2. 准□

在本□余下的部分中， 我□□定使用 ad0s1b 作□交□区。

到目前□止， 交□区仍是未加密的。 很可能其中已□存有明文形式的口令或其他敏感数据。 要□正□一□□，首先□使用随机数来覆□交□分区的数据：

```
# dd if=/dev/random of=/dev/ad0s1b bs=1m
```

19.17.3. 使用 gbde(8) 来加密交□区

/etc/fstab 中与交□区□□的行中， □□名□追加 .bde 后□：

```
# Device                Mountpoint      FStype  Options         Dump    Pass#
/dev/ad0s1b.bde         none            swap    sw              0       0
```

19.17.4. 使用 geli(8) 来加密分区

□一□方法是使用 geli(8) 来□到加密交□区的目的， 其□程与使用 gbde(8) 大体相似。 此□， 在 /etc/fstab 中交□区□□的行中， □□名□追加 .eli 后□：

```
# Device                Mountpoint      FStype  Options         Dump    Pass#
/dev/ad0s1b.eli         none            swap    sw              0       0
```

geli(8) 默□情况下使用密□□度□ 256-位的 AES 加密算法。

当然， □些默□□是可以通□ /etc/rc.conf 中的 geli_swap_flags □□来修改的。 下面的配置表示□ rc.d 脚本 encswap □建一个 geli(8) 交□区， 在其上使用密□□度□ 128-位 的 Blowfish 加密算法， 4 kilobytes 的扇区尺寸， 并采用 "最后一次□□□卸下" 的策略：

```
geli_swap_flags="-e blowfish -l 128 -s 4096 -d"
```

□参□ geli(8) □机手册中□于 onetime 命令的□明， 以了解其他可用的□□。

19.17.5. □□所作的配置能□□□作用

在重□系□之后， 就可以使用 swapinfo 命令来□□加密交□区是否已□在正常□□了。

如果使用了 gbde(8)， □：

```
% swapinfo
Device          1K-blocks     Used    Avail Capacity
/dev/ad0s1b.bde    542720        0   542720     0%
```

如果使用了 geli(8)， □：

```
% swapinfo
Device          1K-blocks     Used    Avail Capacity
/dev/ad0s1b.eli    542720        0   542720     0%
```

19.18. 高可用性存□ (HAST)

19.18.1. 概述

高可用性是担□□□□□的□用的一□主要需求， 而高可用存□□是□□□境中的一□□□□件。 高可用存□ Highly Available STorage， 或 HAST， 是由 Paweł Jakub Dawidek <pjd@FreeBSD.org> □□的一□用于提供在□台物理上隔□的系□之□以透明的方式， 通□ TCP/IP 网□□□数据的高可用性框架。 HAST 可以看作通□网□□行的 RAID1 (□像)， □似于 GNU/Linux® 平台上的 DRBD® 存□系□。 配合 FreeBSD 提供的其他高可用性基□□施， 如 CARP， HAST 可以用来□建可以抗御硬件故障的高可用存□集群。

□完□□， □将了解：

- 何□ HAST， 它如何工作以及提供□些功能。
- 如何在 FreeBSD 上配置和使用 HAST。
- 如何与 CARP 及 devd(8) 配合□建可□的存□系□。

在□□□□之前， □□：

- 了解 UNIX® 和 FreeBSD 的基□知□ (UNIX 基□)。
- 知道如何配置网□接口以及其他核心 FreeBSD 子系□ (□置和□整)。
- 理解 FreeBSD 的网□功能 (网□通□)。
- 使用 FreeBSD 8.1-RELEASE 或更新版本。

HAST □目是由 FreeBSD 基金会□助完成的， 并得到了来自 OMCnet Internet Service GmbH 和 TransIP

BV 的支持。

19.18.2. HAST 的功能

HAST 系□提供的功能主要包括：

- 可以掩□本地硬□的 I/O □□。
- 文件系□无□， 因而可以配合 FreeBSD 支持的任何文件系□使用。
- 高效率的快速重新同□机制， 令系□只同□在□一□点停机□修改□的□。
- 可以在已□部署好的□境中添加冗余。
- 配合 CARP、 Heartbeat 或其他□似的工具， 可以□□健壮的可□存□系□。

19.18.3. HAST 的□行机制

由于 HAST 本□上是在多个机器□同□地□行□□□制， 因此它需要至少□个□点 (物理的机器) - 其一作□ 主 (也称作 `master`) □点， □一个作□ 从 (`slave`) □点。 □□台机器会共同□成一个集群。

目前 HAST 只能使用最多□个集群□点。

由于 HAST 是配置成以主从□点的方式□行， 在任何□刻都只能有唯一的一个□点是主□点。 主 □点， 也称作活□ □点， □□□理由 HAST 管理的□□的全部 I/O □求。 而 从 □点□会自□从 主 □点同□数据的□更操作。

在 HAST 系□中的物理□□包括：

- 本地磁□ (在主□点上)
- □程磁□ (在从□点上)

HAST 在□的□□上同□□行， □使其□文件系□和□用程序透明。 HAST 在 /dev/hast/ 目□中提供□准的 GEOM □□供其他工具或□用程序使用， 因此， 在使用上， □□用程序或文件系□而言， HAST 提供的□□与普通的裸□或分区等没有任何区□。

□到本地磁□的□次写、 □除或□存刷写操作， 都会同□通□ TCP/IP □到□程磁□上。 □操作是由本地磁□完成，除非本地磁□上的数据不是最新的， 或□生了 I/O □□。 在□□情况下， □操作会在从□点上完成。

19.18.3.1. 同□及□制模式

HAST 希望提供快速的故障恢□能力。 基于□一考量， □少在某个□点停机后需要的同□□□就十分重要。□了提供快速的同□能力， HAST 会□□一□保存在磁□上的□区段位映射表 (bitmap of dirty extents)，在普通的同□模式中， 它只同□□些部分的数据 (初始的同□除外)。

□理同□有多□不同的方式， HAST □□□□以下几□同□方式：

- *memsync*： 当本地的写操作已□完成， 并且□程□点□□已□收到数据□， 便□□数据的写操作已□完成，而不是等待□程□点完成数据的写操作。 □程□点在□出回□之后， 会立即□始□行写操作。 □□模式的目□是□少□□□□， 但在同□仍然保持很好的可□性。 目前 *memsync* □制模式尚未□□。
- *fullsync*： 只有在本地写操作完成， 并且□程的写操作也已□完成的情况下， 才□□数据的写操作已□完成。 □□模式是最保□， 同□也是最慢的一□□制模式。 □是目前系□□□的□制模式。

- *async*： 在本地写操作完成□， 即□□数据已□写完。 □是最快， 同□也是□□最大的□制模式， 一般而言只有在□一□点的延□□大□才□考□使用。 目前 *async* □制模式尚未□□。

目前， 只支持 *fullsync* □制模式。

19.18.4. HAST 的配置

HAST 需要 GEOM_GATE 支持才能正常工作。 系□自□的□□ GENERIC 内核 并不 包含 GEOM_GATE， 但默□的 FreeBSD 安装包含了 geom_gate.ko 内核模□。 如果□系□□行了裁剪， □□□□□个模□是否可用。 此外， GEOM_GATE 也可以静□□□□内核， 方法是在内核的□□配置中添加下面的□置：

```
options GEOM_GATE
```

从操作系□的角度， HAST 框架包含了下面□些部件：

- □□□行数据同□的 hastd(8) 服□程序，
- 用于□行管理操作的 hastctl(8) 用□□管理工具，
- 配置文件 hast.conf(5)。

下面的例子将介□使用 HAST 在□个□点之□以 主-从 模式□制数据的方法。 □个□点的名字分□是 hasta 其 IP 地址□ *172.16.0.1*， 以及 hastb， 其 IP 地址□ *172.16.0.2*。 □□台机器都使用尺寸相同的磁□ /dev/ad6 来□用于 HAST 的□行。 HAST 存□池 (有□也称□□源， 例如位于 /dev/hast/ 的□□文件) 将命名□ test。

HAST 的配置文件是 /etc/hast.conf。 在□个□点上， □个文件的内容□□是完全一□的。 最□配置如下：

```
resource test {
    on hasta {
        local /dev/ad6
        remote 172.16.0.2
    }
    on hastb {
        local /dev/ad6
        remote 172.16.0.1
    }
}
```

如果需要更高□的配置， □参□□机手册 hast.conf(5)。

在 remote □句中也可以使用主机名。 □□情况下需要□保□些主机名是可以解析的， 例如在 /etc/hosts 文件中， 或在本地 DNS 中□行了定□。

□在在□个□点上都有同□的配置了， 接下来我□需要□建 HAST 存□池。 在□个□点上分□□行下面的命令来初始化本地此怕， 并□□ hastd(8) 服□：

```
# hastctl create test
# /etc/rc.d/hastd onestart
```

没有 □法使用已□包含文件系□的 GEOM □□来□建存□池 (□言之， 已□存在的文件系□无法□□□ HAST 管理的存□池)， □是因□□建存□池的□程需要保存一些元数据， 而已□写入文件系□的□□不再能提供保存□些元数据所需的空□。

HAST 并不□□□□□点的角色 (主 或 从)。 □点的角色是由管理□手工， 或由□似 Heartbeat □□的□件通□ hastctl(8) 来完成配置的。 在希望成□主□点的系□ (hasta) 上□行下面的命令令其成□主□点：

```
# hastctl role primary test
```

□似地， 用下面的命令来指明从□点 (hastb)：

```
# hastctl role secondary test
```

有可能会出□□个□点之□无法正常通□， 但又都配置□主□点□□的情况； □□称作 □分裂 的状□是十分危□的。 在 从□分裂状□恢□ 中介□了如何从□□状□中恢□的方法。

接下来， 可以在□个□点上分□用 hastctl(8) 工具来□□□点身□是否正□：

```
# hastctl status test
```

□其中比□重要的是 status(状□) □行， 在□个□点上， 其□出均□□ complete(完好)。 如果系□□出的□出是 degraded (降□)， □表示出□了□□。 正常情况下， □点□的同□已□□始。 当 hastctl status 命令□告的 dirty 数据□数量□ 0 字□□， 表示□个□点的数据已□完全同□。

最后一□是在 GEOM □□ /dev/hast/test 上□建文件系□。 □□工作必□在 主 □点上□行 (因□ /dev/hast/test 只在 主 □点上出□)， 随硬□尺寸的不同， □可能需要花□数分□的□□：

```
# newfs -U /dev/hast/test
# mkdir /hast/test
# mount /dev/hast/test /hast/test
```

一旦完成了 HAST 框架的配置， 最后一□就是□保 HAST 在系□引□□程中会自□□□了。 □了□到□个目的， □在 /etc/rc.conf 文件中添加□行配置：

```
hastd_enable="YES"
```

19.18.4.1. 故障□移配置

□个例子的目的在于建立一套健壮的存□系□， 令其能□抵御在任何一个□点上□生的故障。 □其中的□□任□是

□集群中的 主 □点□生故障的情形□行及□的□救□理。 当□生□□情况□， 从 □点可以无□地接手主□点的工作，□文件系□□行□□并挂接， 从而□□□行， 而不□失任何数据。

□了□成□一任□， 需要使用 FreeBSD 提供的□一□功能 - CARP 所提供的 IP □自□故障□移能力。 CARP 是共用地址冗余□□ Common Address Redundancy Protocol 的□写， 它允□多个同网段的主机共享同一 IP 地址。 □根据 Common Address Redundancy Protocol (CARP， 共用地址冗余□□ 的介□在□个□点上都配置 CARP。 完成□些配置之后， □个□点都会有自己的 carp0 网□接口， 共用 IP 地址 *172.16.0.254*。 □然，集群中的 HAST 主□点也必□是 CARP 主□点。

前面一□中□建的 HAST 存□池□在可以提供□网□上的其他主机使用了。 其上的文件系□可以通□ NFS、 Samba 等等， 以共用 IP 地址 *172.16.0.254* 来□□。 □在余下的唯一□□是自□化□主□点故障的□理。

当 CARP 网□接口的□路状□□生□化□， FreeBSD 操作系□会□生一个 devd(8) 消息， □□就可以□□ CARP 网□接口的状□了。 CARP 接口的状□□化表示□点□生故障， 或重新回到了网□中。 □些情况下需要□行特定的脚本来完成□□的□理。

□了截□ CARP 网□接口的状□□化， 需要在□个□点的 /etc/devd.conf 文件中添加如下的□置：

```
notify 30 {
    match "system" "IFNET";
    match "subsystem" "carp0";
    match "type" "LINK_UP";
    action "/usr/local/sbin/carp-hast-switch master";
};

notify 30 {
    match "system" "IFNET";
    match "subsystem" "carp0";
    match "type" "LINK_DOWN";
    action "/usr/local/sbin/carp-hast-switch slave";
};
```

□使□□的配置生效， 需要在□个□点上□行下面的命令：

```
# /etc/rc.d/devd restart
```

当网□接口 carp0 的状□□生□化□， 系□会□生一个通知消息， □允□ devd(8) 子系□□行管理□指定的任意脚本，在□个例子中是 /usr/local/sbin/carp-hast-switch。 □个脚本的作用是自□化故障□移。 □于前面 devd(8) 配置的具体含□， □参□□机手册 devd.conf(5)。

下面是一个□□脚本的示例：

```
#!/bin/sh

# Original script by Freddie Cash <fjwcash@gmail.com>
# Modified by Michael W. Lucas <mwlucas@BlackHelicopters.org>
# and Viktor Petersson <vpetersson@wireload.net>
```

```
# The names of the HAST resources, as listed in /etc/hast.conf
resources="test"

# delay in mounting HAST resource after becoming master
# make your best guess
delay=3

# logging
log="local0.debug"
name="carp-hast"

# end of user configurable stuff

case "$1" in
    master)
        logger -p $log -t $name "Switching to primary provider for ${resources}."
        sleep ${delay}

        # Wait for any "hastd secondary" processes to stop
        for disk in ${resources}; do
            while $( pgrep -lf "hastd: ${disk} \(secondary\)" > /dev/null 2>&1 ); do
                sleep 1
            done

            # Switch role for each disk
            hastctl role primary ${disk}
            if [ $? -ne 0 ]; then
                logger -p $log -t $name "Unable to change role to primary for resource
${disk}."
                exit 1
            fi
        done

        # Wait for the /dev/hast/* devices to appear
        for disk in ${resources}; do
            for I in $( jot 60 ); do
                [ -c "/dev/hast/${disk}" ] && break
                sleep 0.5
            done

            if [ ! -c "/dev/hast/${disk}" ]; then
                logger -p $log -t $name "GEOM provider /dev/hast/${disk} did not
appear."
                exit 1
            fi
        done

        logger -p $log -t $name "Role for HAST resources ${resources} switched to
primary."

        logger -p $log -t $name "Mounting disks."
```

```
        for disk in ${resources}; do
            mkdir -p /hast/${disk}
            fsck -p -y -t ufs /dev/hast/${disk}
            mount /dev/hast/${disk} /hast/${disk}
        done

    ;;

    slave)
        logger -p $log -t $name "Switching to secondary provider for ${resources}."

        # Switch roles for the HAST resources
        for disk in ${resources}; do
            if ! mount | grep -q "^/dev/hast/${disk} on "
            then
            else
                umount -f /hast/${disk}
            fi
            sleep $delay
            hastctl role secondary ${disk} 2>&1
            if [ $? -ne 0 ]; then
                logger -p $log -t $name "Unable to switch role to secondary for
resource ${disk}."
                exit 1
            fi
            logger -p $log -t $name "Role switched to secondary for resource ${disk}."
        done
    ;;
esac
```

□而言之， 在□点成□网□的 `master` / `primary` □点□， 脚本会□行下面的操作：

- 在本□点升格□ HAST 存□池的主□点。
- □□ HAST 存□池上的文件系□。
- 挂接存□池中的文件系□到□当的位置。

当□点成□ `backup` / `secondary` □点□：

- 卸下 HAST 存□池。
- 将本□点降格□ HAST 存□池的从□点。

□必注意， 上面的脚本只是概念性的介□。 它并不能□理所有可能□生的情况， 因此□根据□□情况□行修改， 例如□□/停止必要的服□， 等等。

在前面的例子中， 出于示□的目的我□使用的是□准的 UFS 文件系□。 □了□少恢□所需的□□，可以使用□日志的 UFS 文件系□， 或者使用 ZFS 文件系□。

更具体的信息和例子□参□ HAST Wiki □面。

19.18.5. 故障排除

19.18.5.1. 一般故障排除提示

HAST 通常都能□无故障地□行， 不□， 和任何其他□件□品一□， 有□它也可能会无法以希望的方式□□。 □致□□的可能性有很多， 但一般来□， 首先要□保集群中所有□点的□□是同□的。

当□□排除 HAST 故障□， □提高 hastd(8) 的□□□□。 □可以通□在□□ hastd(8) 服□□指定 -d 参数来□□。 需要□明的是， 可以多次指定□一参数来□一□提高□□□□。 此外， □可以考□使用 -F 参数来□□服□， 它会令 hastd(8) 服□在前台□行。

19.18.5.2. 从□分裂状□恢□

当集群中的□个□点之□无法相互通□□， □个□点都会□□自己是主□点， 从而□致 □分裂 的状□。 □□情形十分危□，因□□个□点会□生互相无法合并的数据。 □□情形需要系□管理□□施手工干□。

从□□状□中恢□□， 管理□必□决定□一个□点包含最重要的数据□□ (或者手工合并□些改□) 并□ HAST □行一次完整的同□操作， 覆□有□□的那个□点的数据。 要完成□个工作，在有□□的□点上□行下面的命令：

```
# hastctl role init <resource>
# hastctl create <resource>
# hastctl role secondary <resource>
```

Chapter 20. GEOM: 模□化磁□□□框架

20.1. 概述

本章将介□以 FreeBSD GEOM 框架来使用磁□。 □包括了使用□一框架来配置的主要的 RAID 控制工具。 □一章不会深入□□ GEOM 如何□理或控制 I/O、 其下□的子系□或代□。 □可以从 geom(4) □机手册及其□多 SEE ALSO 参考文献中得到□些信息。 □一章也不是□ RAID 配置的□威介□， 它只介□由 支持GEOM 的 RAID □□。

□完□章， □将了解：

- 通□ GEOM 支持的 RAID □型。
- 如何使用基本工具来配置和管理不同的 RAID □□。
- 如何通□ GEOM 使用□像、 条□、 加密和挂接在□程的磁□□□。
- 如何排除挂接在 GEOM 框架上的磁□□□的□□。

□□□章之前， □□：

- 理解 FreeBSD 如何□理磁□□□ (存□)。
- 了解如何配置和安装新的 FreeBSD 内核 (配置FreeBSD的内核)。

20.2. GEOM 介□

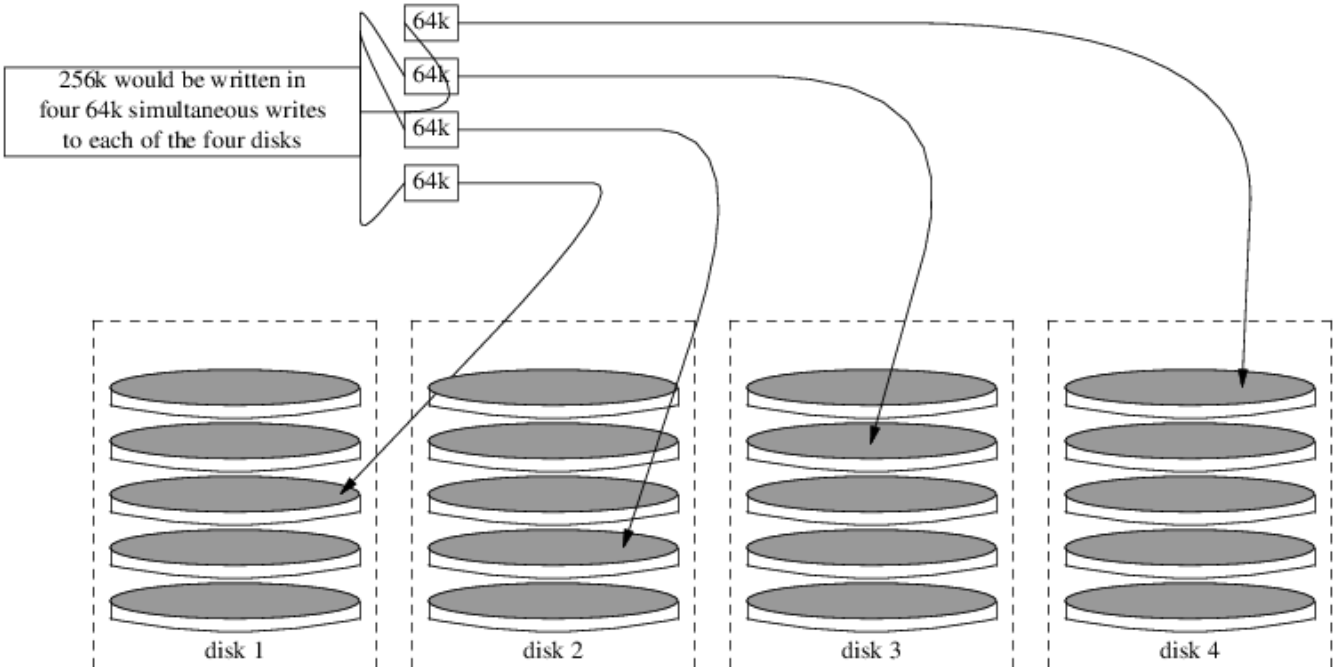
GEOM 允□□□和控制□ (classes) - 主引□□□、 BSD □□ (label)， 等等 - 通□使用 provider， 或在 /dev 中的特殊文件。 它支持□多□件 RAID 配置， GEOM 能□向操作系□， 以及在其上□行的工具提供透明的□□方式。

20.3. RAID0 - 条□

条□是一□将多个磁□□□器合并□一个卷的方法。 □多情况下， □是通□硬件控制器来完成的。 GEOM 磁□子系□提供了 RAID0 的□件支持， 它也成□磁□条□。

在 RAID0 系□中， 数据被分□多个□， □些□将分□写入□列的所有磁□。 与先前需要等待系□将 256k 数据写到一□磁□上不同， RAID0 系□， 能□同□分□将打碎的 64k 写到四□磁□上， 从而提供更好的 I/O 性能。 □一性能提升□能□通□使用多个磁□控制器来□一□改□。

在 RAID0 条□中的□一个□的尺寸必□一□， 因□ I/O □求是分散到多个□上的， 以便□□些□上的□写并行完成。


256k would be written in
four 64k simultaneous writes
to each of the four disks
64k
64k
64k
64k
disk 1
disk 2
disk 3
disk 4

Procedure: 在未格式化的 **ATA** 磁□上建立条□

1. 加□ geom_stripe.ko 模□：

```
# kldload geom_stripe
```

2. □信存在合□的挂接点 (mount point)。 如果□个卷将成□根分区， 那□□□把它挂接到其他位置i， 如 /mnt：

```
# mkdir /mnt
```

3. □定将被做成条□卷的磁□的□□名， 并□建新的条□□□。 □例而言， 要将□个未用的、 尚未分区的 ATA 磁□ /dev/ad2 和 /dev/ad3 做成一个条□□□：

```
# gstripe label -v st0 /dev/ad2 /dev/ad3
Metadata value stored on /dev/ad2.
Metadata value stored on /dev/ad3.
Done.
```

4. 接着需要写□准的 label， 也就是通常所□的分区表到新卷上， 并安装□准的引□代□：

```
# bsdlabel -wB /dev/stripe/st0
```

5. 上述□程将在 /dev/stripe 目□中的 st0 □□基□上建立□个新□□。 □包括 st0a 和 st0c。 □□， 就可以在 st0a □□上用下述 `newfs` 命令来建立文件系□了：

```
# newfs -U /dev/stripe/st0a
```

   在屏幕上将□□一些数字， 整个操作□□能在数秒内完成。 □在可以挂接□□做好的卷了。

要挂接□□建的条□□：

```
# mount /dev/stripe/st0a /mnt
```

要在□□□程中自□挂接□个条□上的文件系□， 需要把□于卷的信息放到 /etc/fstab 文件中。□□到此目的， 需要□建一个叫 stripe 的永久的挂□点：

```
# mkdir /stripe
# echo "/dev/stripe/st0a /stripe ufs rw 2 2" \
    >> /etc/fstab
```

此外， geom_stripe.ko 模□也必□通□在 /boot/loader.conf 中□加下述□置， 以便在系□初始化□程中自□加□：

```
# echo 'geom_stripe_load="YES"' >> /boot/loader.conf
```

20.4. RAID1 - □像

□像是□多公司和家庭用□使用的一□无□中断的□□技□。 □□地□， □像的概念就是 磁□B 是同□□制 (replicate) 的磁□A 的副本， 或者 磁□C+D 是 diskA+B 的同□□制副本， 等等。 无□磁□配置如何， □□技□的共同特点都是一□磁□或分区的内容会同□□制到□外的地方。 □□， 除了能□很容易地恢□信息之外， □能□在无□中断服□或□□的情况下□行□□， 甚至直接将副本送到数据保安公司□地□存。

在□始做□件事之前， 首先□准□□个容量相同的磁□□□器， 下面的例子假定它□都是使用直接□□方式 (Direct Access， da(4)) 的 SCSI 磁□。

20.4.1. □主磁□□行□像

假定□□有系□中的 FreeBSD 安装到了第一个， 也就是 da0 □上， □□告□ gmirror(8) 将主要数据保存在□里。

在□始□建□像卷之前， 可以□用更多的□□信息， 并□□放□□□的完全□□。 □可以通□将 sysctl(8) □量 kern.geom.debugflags □置□下面的□来□□：

```
# sysctl kern.geom.debugflags=17
```

接下来需要□建□像。 □个□程的第一□是在主磁□上保存元数据信息， 也就是用下面的命令来□建 /dev/mirror/gm □□：

在引□用的□□基□上新建□像□， 有可能会□致保存在磁□上最后一个扇区的数据□失。 在新安装 FreeBSD 之后立即□建□像可以□低此□□。 下面的操作与默□的 FreeBSD 9.*X* 安装□程不兼容， 因□它采用了新的 GPT 分区格式。 GEOM 会覆□ GPT 元数据， □会□致数据□失， 并有可能□致系□无法引□。

```
# gmirror label -vb round-robin gm0 /dev/da0
```

系□□□出下面的回□：

```
Metadata value stored on /dev/da0.
Done.
```

初始化 GEOM， □□操作会加□内核模□ /boot/kernel/geom_mirror.ko：

```
# gmirror load
```

当□个命令□行完之后， 系□会在 /dev/mirror 目□中□建□□□点 gm0。

配置在系□初始化□程中自□加□ geom_mirror.ko：

```
# echo 'geom_mirror_load="YES"' >> /boot/loader.conf
```

□□ /etc/fstab 文件， 将其中先前的 da0 改□新的□像□□ gm0。

如果 vi(1) 是□喜□的□□器， 以下□是完成此□任□的一个□便方法：

```
# vi /etc/fstab
```

在 vi(1) 中□□□有的 fstab 内容， 具体操作是 `:w /etc/fstab.bak`。 接着， 把所有旧的 da0 替□成 gm0， 也就是□入命令 `:%s/da/mirror\/gm/g`。

修改完后的 fstab 文件□□是下面的□子。 磁□□□器是 SCSI 或 ATA 甚至 RAID 都没有□系， 最□的□果都是 gm。

```
# Device        Mountpoint  FStype  Options     Dump    Pass#
/dev/mirror/gm0s1b  none        swap    sw      0   0
/dev/mirror/gm0s1a  /       ufs rw      1   1
/dev/mirror/gm0s1d  /usr        ufs rw      0   0
/dev/mirror/gm0s1f  /home       ufs rw      2   2
#/dev/mirror/gm0s2d /store      ufs rw      2   2
/dev/mirror/gm0s1e  /var        ufs rw      2       2
/dev/acd0       /cdrom      cd9660  ro,noauto   0   0
```

重□系□：

```
# shutdown -r now
```

在系□初始化□程中， 新建的 gm0 会代替 da0 □□工作。 系□完成初始化之后， 可以通□□□ `mount` 命令的□出来□看效果：

```
# mount
Filesystem         1K-blocks    Used    Avail Capacity  Mounted on
/dev/mirror/gm0s1a   1012974  224604   707334    24%    /
devfs                      1       1        0   100%    /dev
/dev/mirror/gm0s1f  45970182   28596 42263972     0%    /home
/dev/mirror/gm0s1d   6090094 1348356  4254532    24%    /usr
/dev/mirror/gm0s1e   3045006 2241420   559986    80%    /var
devfs                      1       1        0   100%    /var/named/dev
```

□个□出是正常的。 最后， 使用下面的命令将 da1 磁□加到□像卷中， 以□始同□□程：

```
# gmirror insert gm0 /dev/da1
```

在□建□像卷的□程中， 可以用下面的命令□看状□：

```
# gmirror status
```

一旦□像卷的□建操作完成， □个命令的□出就会□成□□：

```
      Name    Status  Components
mirror/gm0  COMPLETE  da0
                      da1
```

如果有□□或者□建仍在□行， □出中的 COMPLETE 就会是 DEGRADED。

20.4.2. 故障排除

20.4.2.1. 系□拒□引□

如果系□引□□出□□似下面的提示：

```
ffs_mountroot: can't find rootvp
Root mount failed: 6
mountroot>
```

□□情况□使用□源或□位按□重□机器。 在引□菜□中， □□第六 (6) 个□□。 □将□系□□入 loader(8) 提示符。 在此□手工加□内核模□：

```
OK? load geom_mirror
OK? boot
```

如果□□做能解决□□， □□明由于某□原因模□没有被正□加□。 □□ /boot/loader.conf 中相□条目是否正□。 如果□□仍然存在，可以在内核配置文件中加入：

```
options GEOM_MIRROR
```

然后重新□□和安装内核来解决□个□□。

20.4.3. 从磁□故障中恢□

磁□□像的一大好□是在当其中一个磁□出□故障□， 可以很容易地将其替□掉， 并且通常不会□失数据。

考□前面的 RAID1 配置， 假□ da1 出□了故障并需要替□， 要替□它， 首先□定□个磁□出□了故障， 并□□系□。 此□， 可以用□上新的磁□， 并重新□□系□。 □之后可以用下面的命令来完成磁□的替□操作：

```
# gmirror forget gm0
```

```
# gmirror insert gm0 /dev/da1
```

在重建□程中可以用 `gmirror status` 命令来□看□度。 就是□□□□。

20.5. RAID3 - 使用□用校□□□的字□□条□

RAID3 是一□将多个磁□□成一个卷的技□， 在□个配置中包含一个□用于校□的□。 在 RAID3 系□中，数据会以字□□□位拆分并写入除校□□之外的全部□□器中。 □意味着从 RAID3 中□取数据□将会□□所有的□□器。采用多个磁□控制器可以□一□改善性能。 RAID3 □列最多可以容忍其中的 1 个□□器出□故障，它可以提供全部□□器□容量的 1 - 1/n， 此□ n 是□列中的磁□数量。 □□配置比□□合保存大容量的数据，例如多媒体文件。

在建立 RAID3 □列□， 至少需要 3 □磁□。 所有的□的尺寸必□一致， 因□ I/O □求会并□分派到不同的□上。□外， 由于 RAID3 本身的□□， □的数量必□恰好是 3, 5, 9, 17, 等等 (2^n + 1)。

20.5.1. 建立□用的 RAID3 □列

在 FreeBSD 中， RAID3 是通□ graid3(8) GEOM class □□的。 在 FreeBSD 中建立□用的 RAID3 □列需要下述□□。

□然理□上从 RAID3 □列□□ FreeBSD 是可行的， 但□并不常□， 也不推□□□□做。

1. 首先， 在引□加□器中用下面的命令加□ geom_raid3.ko 内核模□：

   ```
   # graid3 load
   ```

   此外， 也可以通□命令行手工加□ geom_raid3.ko 模□：

   ```
   # kldload geom_raid3.ko
   ```

2. □建用于挂□卷的挂点目□：

   ```
   # mkdir /multimedia/
   ```

3. □定将要加入□列的磁□□□名， 并□建新的 RAID3 □□。 最□， □个□□将代表整个□列。 下面的例子使用三个未□分区的 ATA 磁□： ada1 和 ada2 保存数据， 而 ada3 用于校□。

   ```
   # graid3 label -v gr0 /dev/ada1 /dev/ada2 /dev/ada3
   Metadata value stored on /dev/ada1.
   Metadata value stored on /dev/ada2.
   Metadata value stored on /dev/ada3.
   Done.
   ```

4. □新建的 gr0 □□分区， 并在其上□建 UFS 文件系□：

   ```
   # gpart create -s GPT /dev/raid3/gr0
   # gpart add -t freebsd-ufs /dev/raid3/gr0
   # newfs -j /dev/raid3/gr0p1
   ```

   屏幕上会□□□多数字， □个□程需要一段□□才能完成。 此后， □就完成了□建卷的全部操作， 可以挂□它了。

5. 最后一□是挂□文件系□：

   ```
   # mount /dev/raid3/gr0p1 /multimedia/
   ```

   □在可以使用 RAID3 □列了。

□了□上述配置在系□重□后□□可用， □需要□行一些□外的配置操作。

1. 在挂□卷之前必□首先加□ geom_raid3.ko 模□。 将下面的配置添加到 /boot/loader.conf 文件中，可以□系□在引□□程中自□加□□个模□：

```
geom_raid3_load="YES"
```

2. □需要在 /etc/fstab 文件中加入下列配置， 以便□系□引□□自□挂□□列上的文件系□：

```
/dev/raid3/gr0p1    /multimedia ufs rw  2   2
```

20.6. GEOM Gate 网□□□

通□ gate 工具， GEOM 支持以□程方式使用□□， 例如磁□、 CD-ROM、 文件等等。 □和 NFS □似。

在□始工作之前， 首先要□建一个□出文件。 □个文件的作用是指定□可以□□□出的□源， 以及提供何□□□的□□授□。 例如， 要把第一□ SCSI □的第四个 slice □出， □□的 /etc/gg.exports 会是□似下面的□子：

```
192.168.1.0/24 RW /dev/da0s4d
```

□表示允□同属私有子网的所有机器□□ da0s4d 分区上的文件系□。

要□出□个□□， 首先□□□它没有被挂接， 然后是□□ ggated(8) 服□：

```
# ggated
```

□在我□将在客□机上 mount □□□， 使用下面的命令：

```
# ggatec create -o rw 192.168.1.1 /dev/da0s4d
ggate0
# mount /dev/ggate0 /mnt
```

到此□止， □□□□已□可以通□挂接点 /mnt □□了。

□注意， 如果□□已□被服□器或网□上的任何其他机器挂接， □前述操作将会失□。

如果不再需要使用□个□□， 就可以使用 umount(8) 命令来安全地将其卸下了， □一点和其他磁□□□似。

20.7. □磁□□□添加卷□

在系□初始化的□程中， FreeBSD 内核会□□□到的□□□建□□□点。 □□□□方式存在一些□□， 例如， 在通□ USB 添加□□□□如何□理？ 很可能有□存□□□最初被□□□ da0 而在□之后， □由 da0 □成了 da1。 而□□会在挂接 /etc/fstab 中的文件系□□造成□□， □些□□， □可能在系□引□□□致无法正常□□。

解决□个□□的一个方法是以□接拓□方式□式地□行 SCSI □□命名， □□， 当在 SCSI □上□加新□□□， □些□□将使用一个未用的□号。 但如果 USB □□取代了主 SCSI 磁□的位置□？ 由于 USB 通常会在 SCSI □之前□□到， 因此很可能出□□□□象。 当然， 可以通□在系□引□之后再□入□些□□来□□□个□□。 □一□□□□个□□的方法，□是只使用 ATA □□器， 并避免在 /etc/fstab 中列出 SCSI □□。

□有一□更好的解决方法。 通□使用 glabel 工具， 管理□或用□可以□磁□□□打上□□， 并在 /etc/fstab 中使用□些□□。 由于 glabel 会将□□保存在□□ provider 的最后一个扇区， 在系□重□之后， 它仍会持□存在。 因此，通□将具体的□□替□□使用□□表示， 无□□□□点□成什□， 文件系□都能□□利地完成挂接。

□并不是□□□一定是永久性的。 glabel 工具既可以□建永久性□□， 也可以□建□□性□□。在重□□， 只有永久性□□会保持。 □参□□机手册 glabel(8) 以了解□者之□的差□。

20.7.1. □□□型和使用示□

有□□□型的□□， 一□是普通□□， □一□是文件系□□□。 □□可以是永久性的或□□性的。永久性的□□可以通□ tunefs(8) 或 newfs(8) 命令□□。根据文件系□的□型， 它□将在 /dev 下的一个子目□中被□建。例如， UFS2 文件系□的□□会□建到 /dev/ufs 目□中。永久性的□□□可以使用 glabel label □建。它□不再是文件系□特定的，而是会在 /dev/label 目□中被□建。

□□性的□□在系□下次重□□会消失， □些□□会□建到 /dev/label 目□中， 很□合□□之用。可以使用 glabel create □建□□性的□□。□参□ glabel(8) 手册□以□取更多□□信息。

要□一个 UFS2 文件系□□建永久性□□， 而不破坏其上的数据，可以使用下面的命令：

```
# tunefs -L home /dev/da3
```

如果文件系□□了， □可能会□致数据□坏； 不□， 如果文件系□快□了， 此□□首先□除一些无用的文件， 而不是□加□□。

□在， □□可以在 /dev/ufs 目□中看到□□， 并将其加入 /etc/fstab：

```
/dev/ufs/home       /home           ufs     rw              2       2
```

当□行 tunefs □， □首先卸下文件系□。

□在可以像平□一□挂接文件系□了：

```
# mount /home
```

□在， 只要在系□引□□通□ /boot/loader.conf 配置加□了内核模□ geom_label.ko， 或在□□内核□指定了 GEOM_LABEL □□， □□□点由于□□□□而□序□生□化□， 就不会影□文件系□的挂接了。

通□使用 newfs 命令的 -L 参数， 可以在□建文件系□□□其添加默□的□□。 □参□□机手册 newfs(8) 以了解□一□的□情。

下列命令可以清除□□：

```
# glabel destroy home
```

以下的例子展示了如何□一个□□磁□打上□□。

例 *31.* □□□磁□打上□□

□□□磁□打上永久性□□， 系□□□能□正常□□， 即使磁□被移□到了□外一个控制器或者□移到了一个不同的系□上。 此例中我□假□使用了一个 ATA 磁□， 当前□个□□被系□□□□ ad0。 □假□使用了□准的 FreeBSD 分区□分方案， /, /var, /usr 和 /tmp 文件系□， □有一个 swap 分区。

重□系□，在 loader(8) 提示符下□入 `4` □□到□用□模式。 然后□入以下的命令：

```
# glabel label rootfs /dev/ad0s1a
GEOM_LABEL: Label for provider /dev/ad0s1a is label/rootfs
# glabel label var /dev/ad0s1d
GEOM_LABEL: Label for provider /dev/ad0s1d is label/var
# glabel label usr /dev/ad0s1f
GEOM_LABEL: Label for provider /dev/ad0s1f is label/usr
# glabel label tmp /dev/ad0s1e
GEOM_LABEL: Label for provider /dev/ad0s1e is label/tmp
# glabel label swap /dev/ad0s1b
GEOM_LABEL: Label for provider /dev/ad0s1b is label/swap
# exit
```

系□加□□□□□入多用□模式。 在□□完□后， □□ /etc/fstab 用各自的□□替□下常□的□□名。 最□ /etc/fstab 看起来差不多是□□的：

```
# Device                Mountpoint      FStype  Options         Dump    Pass#
/dev/label/swap         none            swap    sw              0       0
/dev/label/rootfs       /               ufs     rw              1       1
/dev/label/tmp          /tmp            ufs     rw              2       2
/dev/label/usr          /usr            ufs     rw              2       2
/dev/label/var          /var            ufs     rw              2       2
```

□在可以重□系□了。 如果一切□利的□， 系□可以正常□□并且 `mount` 命令□示：

```
# mount
/dev/label/rootfs on / (ufs, local)
devfs on /dev (devfs, local)
/dev/label/tmp on /tmp (ufs, local, soft-updates)
/dev/label/usr on /usr (ufs, local, soft-updates)
/dev/label/var on /var (ufs, local, soft-updates)
```

从 FreeBSD 7.2 □始， glabel(8) class 新□了一□用于 UFS 文件系□唯一□□符， `ufsid` 的□□支持。 □些□□可以在 /dev/ufsid 目□中□到， 它□会在系□引□□自□□建。 在 /etc/fstab 机制中， 也可以使用 `ufsid` □□。 □可以使用

glabel status 命令来□得与文件系□□□的 ufsid □□列表：

```
% glabel status
                  Name  Status  Components
ufsid/486b6fc38d330916     N/A  ad4s1d
ufsid/486b6fc16926168e     N/A  ad4s1f
```

在上面的例子中 ad4s1d 代表了 /var 文件系□， 而 ad4s1f □代表了 /usr 文件系□。 □可以使用□些 ufsid □来挂□它□， 在 /etc/fstab 中配置□似□□：

```
/dev/ufsid/486b6fc38d330916        /var        ufs        rw        2      2
/dev/ufsid/486b6fc16926168e        /usr        ufs        rw        2      2
```

所有包含了 ufsid 的□□都可以用□□方式挂□， 从而消除了需要手工□建永久性□□的麻□， 而又能□提供提供与□□名无□的挂□方式的便利。

20.8. 通□ GEOM □□ UFS 日志

随着 FreeBSD 7.0 的□布， 提供了□期□人□所期待的日志功能的□□。 □个□□采用了 GEOM 子系□，可以很容易地使用 gjournal(8) 工具来□行配置。

日志是什□？ 日志的作用是保存文件系□事□的□□， □言之， 完成一次完整的磁□写入操作所需的□□， □些□□会在元数据以及文件数据写□之前， 写入到磁□中。 □□事□日志可以在随后用于重放并完成文件系□事□，以避免文件系□出□不一致的□□。

□□方法是□一□阻止文件系□□失数据并□生不一致的方法。 与 Soft Updates 追踪并□保元数据更新□序□□方法不同， 它会□□地将日志保存到指定□此□任□保留的磁□空□上， 在某些情况下可全部存放到□外一□磁□上。

与其他文件系□的日志□□不同， gjournal 采用的是基于□， 而不是作□文件系□的一部分的方式 - 它只是作□一□ GEOM □展□□。

如果希望□用 gjournal， FreeBSD 内核需要下列□□ - □是 FreeBSD 7.0 以及更高版本系□上的默□配置：

```
options UFS_GJOURNAL
```

如果使用日志的卷需要在□□的□候被挂□， □需加□ geom_journal.ko 内核模□， 将以下□行加入 /boot/loader.conf：

```
geom_journal_load="YES"
```

□个功能也可被□□□一个定制的内核， 需在内核配置文件中加入以下□行：

```
options GEOM_JOURNAL
```

□在， 可以□空□的文件系□□建日志了。 □于新□的 SCSI 磁□ da4， 具体的操作□□□：

```
# gjournal load
# gjournal label /dev/da4
```

□□， 就会出□一个与 /dev/da4 □□□点□□的 /dev/da4.journal □□□点。 接下来， 可以在□个□□上建立文件系□：

```
# newfs -O 2 -J /dev/da4.journal
```

□个命令将建立一个包含日志□□的 UFS2 文件系□。

然后就可以用 `mount` 命令来挂接□□了：

```
# mount /dev/da4.journal /mnt
```

当磁□包含多个 slice □， □个 slice 上都会建立日志。 例如， 如果有 ad4s1 和 ad4s2 □□个 slice， □ `gjournal` 会建立 ad4s1.journal 和 ad4s2.journal。

出于性能考□， 可能会希望在其他磁□上保存日志。 □于□□情形， □□在□用日志的□□后面，□出日志提供者或存□□□。 在□存的文件系□上， 可以用 `tunefs` 来□用日志； 不□， 在□□修改文件系□之前，□□□其□行□□。 多数情况下， 如果无法□建□□的日志， `gjournal` 就会失□， 并且不会防止由于不当使用 `tunefs` 而造成的数据□失。

□于 FreeBSD 系□的□□磁□使用日志也是可能的。 □参□ Implementing UFS Journaling on a Desktop PC 以□得更多□□信息。

Chapter 21. 文件系□ Support

21.1. 概述

文件系□□于任何操作系□来□都是一个不可缺的部分。 它□允□用□上□和存□文件，提供□数据的□□，当然，是使硬□能具有□□的用途。不同的操作系□通常都有一个共同的主要方面， 那就是它□原生的文件系□。在 FreeBSD 上□个文件系□通常被称□快速文件系□或者 FFS， □是基于原来的 Unix™ 文件系□，通常也被称□ UFS。 □是 FreeBSD 用于在磁□上□□数据的原生的文件系□。

FreeBSD 也支持数量繁多的不同的文件系□， 用于提供本地从其他操作系□上□□数据的支持，那些就是指存放在本地挂□的 USB 存□□□，□存□□和硬□上的数据。□支持一些非原生的文件系□。 □些文件系□是在其他的操作系□上□□的，像 Linux® 的□展文件系□ （EXT），和 Sun™ 的 Z 文件系□ （ZFS）。

FreeBSD 上□于各□文件系□的支持分成不同的□次。 一些要求加□内核模□，□外的可能要求安装一系列的工具。 □一章□旨在□助 FreeBSD 用□在他□的系□上□□其他的文件系□， 由 Sun™ 的 Z 文件系□□始。

在□□了□一章□之后，□将了解：

- 原生与被支持的文件系□之□的区□。
- FreeBSD 支持□些文件系□。
- 如何起用，配置，□□和使用非原生的文件系□。

在□□□章以前，□□□：

- 了解 UNIX® 和 FreeBSD 基本知□ (UNIX 基□)。
- 熟悉基本的内核配置/□□方法 (配置FreeBSD的内核)。
- 熟悉在 FreeBSD 上安装第三方□件 (安装□用程序. Packages 和 Ports)。
- 熟悉 FreeBSD 上的磁□，存□和□□名 (存□)。

21.2. Z 文件系□ (ZFS)

Z 文件系□是由 Sun™ □□使用存□池方法的新技□。 □就是□只有在需要存□数据的□候空□才会被使用。 它也□保□数据最大完整性而□□的，支持数据快照， 多□拷□和数据校□。□加了被称□ RAID-Z 的新的数据□制□型。RAID-Z 是□□似于 RAID5□型, 但被□□成防止写入漏洞。

21.2.1. □整 ZFS

ZFS 子系□需利用到大量的系□□源， 所以可能需要一些□校来□日常□用提供最大化的效能。 作□ FreeBSD 的一□□□性的特性，□可能在不久的将来有所□化； 无□如何，下面的□些□□是我□推□的：

21.2.1.1. 内存

□共的系□内存至少□有 1GB，推□ 2GB 或者更多。 在此□所有的例子中，我□使用了 1GB 内存的系□并配合了一些恰当的□校。

有些人在少于 1GB 内存的□境有幸正常使用， 但是在□□有限的物理内存的条件下，当系□的□□很高□，FreeBSD □有可能因于内存耗尽而崩□。

21.2.1.2. 内核配置

我□建□把未使用的□□和□□从内核配置文件中去除。 既然大部□的□□都有以模□的形式存在，它□就可以很容易的通□ /boot/loader.conf 加□。

i386™ □架的用□□在内核配置文件中加入以下的□□， 重新□□内核并重□机器：

```
options     KVA_PAGES=512
```

□个□□将□展内核的地址空□， 因而允□ vm.kvm_size 能□超越 1 GB 的限制(PAE□ 2 GB)。 □了□出□个□□最合□的□， 把以兆(MB)□□位所需的地址空□除以 4 得到。 在□个例子中，512 □□ 2 GB。

21.2.1.3. Loader 可□参数

所有□架上 FreeBSD 都□□加大 kmem 地址空□。在有 1GB 物理内存的□□系□上，在 /boot/loader.conf 中加入如下的参数并且重□后通□了□□。

```
vm.kmem_size="330M"
vm.kmem_size_max="330M"
vfs.zfs.arc_max="40M"
vfs.zfs.vdev.cache.size="5M"
```

更多 ZFS 相□推□□校的□□□参□ http://wiki.freebsd.org/ZFSTuningGuide.

21.2.2. 使用 ZFS

FreeBSD 有一□□□机制能在系□初始化□挂□ ZFS 存□池。 可以通□以下的命令□置：

```
# echo 'zfs_enable="YES"' >> /etc/rc.conf
# /etc/rc.d/zfs start
```

□□文□剩余的部分假定系□中有 3 □ SCSI 磁□可用， 它□的□□名分□□ da0， da1 和 da2。 IDE 硬件的用□可以使用 ad 代替 SCSI。

21.2.2.1. □个磁□存□池

在□个磁□上□建一个□□， 非冗余的 ZFS， 使用 zpool 命令：

```
# zpool create example /dev/da0
```

可以通□ df 的□出□看新的存□池：

```
# df
Filesystem  1K-blocks    Used    Avail Capacity  Mounted on
/dev/ad0s1a   2026030  235230  1628718    13    /
devfs               1       1        0   100    /dev
/dev/ad0s1d  54098308 1032846 48737598     2    /usr
example      17547136       0 17547136     0    /example
```

□□□出清楚的表明了 example 存□池不□□建成功而且被 挂□ 了。 我□能像□□普通的文件系□那□□□它，就像以下例子中演示的那□，用□能□在上面□建文件并□□：

```
# cd /example
# ls
# touch testfile
# ls -al
total 4
drwxr-xr-x   2 root  wheel    3 Aug 29 23:15 .
drwxr-xr-x  21 root  wheel  512 Aug 29 23:12 ..
-rw-r--r--   1 root  wheel    0 Aug 29 23:15 testfile
```

□憾的是□个存□池并没有利用到 ZFS 的任何特性。 在□个存□池上□建一个文件系□，并□用□□：

```
# zfs create example/compressed
# zfs set compression=gzip example/compressed
```

□在 example/compressed 是一个□用了□□的 ZFS 文件系□了。 可以□□□制一些大的文件到 /example/compressed。

使用□个命令可以禁用□□：

```
# zfs set compression=off example/compressed
```

使用如下的命令卸□□个文件系□，并用 df 工具□□：

```
# zfs umount example/compressed
# df
Filesystem  1K-blocks    Used    Avail Capacity  Mounted on
/dev/ad0s1a   2026030  235232  1628716    13    /
devfs               1       1        0   100    /dev
/dev/ad0s1d  54098308 1032864 48737580     2    /usr
example      17547008       0 17547008     0    /example
```

重新挂在□个文件系□使之能被□□， 并用 df □□：

```
# zfs mount example/compressed
# df
Filesystem         1K-blocks    Used    Avail Capacity  Mounted on
/dev/ad0s1a          2026030  235234  1628714    13    /
devfs                      1       1        0   100    /dev
/dev/ad0s1d         54098308 1032864 48737580     2    /usr
example             17547008       0 17547008     0    /example
example/compressed  17547008       0 17547008     0    /example/compressed
```

存□池与文件系□也可通□ mount 的□出□看：

```
# mount
/dev/ad0s1a on / (ufs, local)
devfs on /dev (devfs, local)
/dev/ad0s1d on /usr (ufs, local, soft-updates)
example on /example (zfs, local)
example/data on /example/data (zfs, local)
example/compressed on /example/compressed (zfs, local)
```

正如前面所提到的，ZFS 文件系□，在□建之后就能像普通的文件系□那□使用。然而，□有很多其他的特性是可用的。在下面的例子中，我□将□建一个新的文件系□，data。并要在上面存□些重要的文件，所以文件系□需要被□置成把□一个数据□都保存□□拷□：

```
# zfs create example/data
# zfs set copies=2 example/data
```

□在可以再次使用 df □看数据和空□的使用状况：

```
# df
Filesystem         1K-blocks    Used    Avail Capacity  Mounted on
/dev/ad0s1a          2026030  235234  1628714    13    /
devfs                      1       1        0   100    /dev
/dev/ad0s1d         54098308 1032864 48737580     2    /usr
example             17547008       0 17547008     0    /example
example/compressed  17547008       0 17547008     0    /example/compressed
example/data        17547008       0 17547008     0    /example/data
```

□注意存□池上的□一个文件系□都有着相同数量的可用空□。□就是我□在□些例子中使用 df 的原因，是□了文件系□都是从相同的存□池取得它□所需的空□。ZFS 去掉了□如卷和分区此□的概念，并允□多个文件系□占用同一个存□池。不再需要文件系□与存□池的□候能像□□□□它□：

```
# zfs destroy example/compressed
# zfs destroy example/data
# zpool destroy example
```

磁□无法避免的会坏掉和停止□□。 当□□磁□坏掉的□候，上面的数据都将□失。 一个避免因磁□□坏而□失数据的方法是使用 RAID。ZFS 在它的存□池□□中支持□□的特性， □便是下一□将探□的。

21.2.2.2. ZFS RAID-Z

正如前文中所提到的，□一章□将假□存在 3 个 SCSI □□， da0， da1 和 da2 (或者 ad0 和超出此例使用了 IDE 磁□)。 使用如下的命令□建一个 RAID-Z 存□池:

```
# zpool create storage raidz da0 da1 da2
```

Sun™ 推□在一个 RAID-Z 配置中使用的磁□数量□ 3 至 9 □。 如果□要求在□独的一个存□池中使用 10 □或更多的磁□， □考□分拆成更小 RAID-z □。 如果□只有 2 □磁□，并仍然需要冗余， □考□使用 ZFS 的 mirror 特性。 更多□□□参考 zpool(8) 手册□。

zpool `storage` 至此就□建好了。 可以如前文提到的那□使用 mount(8) 和 df(1) □□。 如需配□更多的磁□□□□把它□加□个列表的后面。 在存□池上□建一个叫 `home` 的文件系□， 用□的文件最□都将被保存在上面：

```
# zfs create storage/home
```

像前文中提到的那□，用□的目□与文件也可□用□□并保存多□拷□， 可通□如下的命令完成：

```
# zfs set copies=2 storage/home
# zfs set compression=gzip storage/home
```

把用□的数据都拷□□来并□建一个符号□接， □他□□始使用□个新的目□：

```
# cp -rp /home/* /storage/home
# rm -rf /home /usr/home
# ln -s /storage/home /home
# ln -s /storage/home /usr/home
```

□在用□的数据□□都保存在新□建的 /storage/home 上了。 □□添加一个新用□并以□个身□登□。

□□□建一个可日后用来回退的快照：

```
# zfs snapshot storage/home@08-30-08
```

□注意快照□□将只会□取一个真□的文件系□， 而不是某个用□目□或文件。`@` 字符□文件系□名或卷名的分隔符。当用□目□被□坏□，可用如下命令恢□：

```
# zfs rollback storage/home@08-30-08
```

□得所有可用快照的列表，可使用 `ls` 命令□看文件系□的 .zfs/snapshot 目□。例如，□行如下命令来□看之前

□取的快照：

```
# ls /storage/home/.zfs/snapshot
```

可以□写一个脚本来□月定期□取用□数据的快照，久而久之， 快照可能消耗掉大量的磁□空□。 之前□建的快照可用以下命令□除：

```
# zfs destroy storage/home@08-30-08
```

在所有□些□□之后，我□没有理由再把 /store/home □□放置了。□它称□真正的 /home 文件系□：

```
# zfs set mountpoint=/home storage/home
```

使用 df 和 mount 命令将□示□在系□把我□的文件系□真正当作了 /home：

```
# mount
/dev/ad0s1a on / (ufs, local)
devfs on /dev (devfs, local)
/dev/ad0s1d on /usr (ufs, local, soft-updates)
storage on /storage (zfs, local)
storage/home on /home (zfs, local)

# df
Filesystem   1K-blocks    Used    Avail Capacity  Mounted on
/dev/ad0s1a    2026030  235240  1628708    13    /
devfs                1       1        0   100    /dev
/dev/ad0s1d   54098308 1032826 48737618     2    /usr
storage       26320512       0 26320512     0    /storage
storage/home  26320512       0 26320512     0    /home
```

□□就基本完成了 RAID-Z 的配置了。使用夜□ periodic(8) □取有□文件系□□建之□的状□更新，□行如下的命令：

```
# echo 'daily_status_zfs_enable="YES"' >> /etc/periodic.conf
```

21.2.2.3. 修□ RAID-Z

□一□□ RAID 都有□□它□ 状□ 的方法。 ZFS 也不例外。 可以使用如下的命令□看 RAID-Z □□：

```
# zpool status -x
```

如果所有的存□池□于健康状□并且一切正常的□， 将返回如下信息：

```
all pools are healthy
```

如果存在□□，可能是一个磁□□□下□了， 那□返回的存□池的状□将看上去是□似□个□子的：

```
pool: storage
 state: DEGRADED
status: One or more devices has been taken offline by the administrator.
    Sufficient replicas exist for the pool to continue functioning in a
    degraded state.
action: Online the device using 'zpool online' or replace the device with
    'zpool replace'.
 scrub: none requested
config:

    NAME        STATE     READ WRITE CKSUM
    storage     DEGRADED     0     0     0
      raidz1    DEGRADED     0     0     0
        da0     ONLINE       0     0     0
        da1     OFFLINE      0     0     0
        da2     ONLINE       0     0     0

errors: No known data errors
```

在□个例子中，□是由管理□把此□□下□后的状□。 可以使用如下的命令将磁□下□：

```
# zpool offline storage da1
```

□在切断系□□源之后就可以替□下 da1 了。 当系□再次上□□，使用如下的命令替□磁□：

```
# zpool replace storage da1
```

至此可用不□ -x □志的命令再次□□状□：

```
# zpool status storage
pool: storage
 state: ONLINE
 scrub: resilver completed with 0 errors on Sat Aug 30 19:44:11 2008
config:

    NAME        STATE     READ WRITE CKSUM
    storage     ONLINE       0     0     0
      raidz1    ONLINE       0     0     0
        da0     ONLINE       0     0     0
        da1     ONLINE       0     0     0
        da2     ONLINE       0     0     0

errors: No known data errors
```

在□个例子中，一切都□示正常。

21.2.2.4. 数据校□

正如前面所提到的，ZFS 使用 校□和(checksum) 来□□存□数据的完整性。 □□在文件系□□建□自□□用的，可使用以下的命令禁用：

```
# zfs set checksum=off storage/home
```

□不是个明智的□□，因□校□和 不□非常有用而且只需占用少量的存□空□。 并且□用它□也不会明□的消耗□多□源。 □用后就可以□ ZFS 使用校□和校□来□□数据的完整。 □个□程通常称□ "scrubbing"。可以使用以下的命令□□ `storage` 存□池里数据的完整性：

```
# zpool scrub storage
```

□个□程需花□相当□的□□，取决于存□的数据量。 而且 I/O 非常密集， 所以在任何□□只能□行一个□□的操作。在 scrub 完成之后，状□就会被更新， 可使用如下的命令□看：

```
# zpool status storage
pool: storage
 state: ONLINE
 scrub: scrub completed with 0 errors on Sat Aug 30 19:57:37 2008
config:

    NAME        STATE     READ WRITE CKSUM
    storage     ONLINE       0     0     0
      raidz1    ONLINE       0     0     0
        da0     ONLINE       0     0     0
        da1     ONLINE       0     0     0
        da2     ONLINE       0     0     0

errors: No known data errors
```

□个例子中完成□□非常的清楚。 □个特性可以□助□在很□的一段□□内□保数据的完整。

Z 文件系□有更多的□□，□参□ zfs(8) 和 zpool(8) 手册□。

Chapter 22. Vinum 卷管理程序

22.1. 概述

无□□有什□□的磁□，□会有一些潜在□□：

- 它□可能容量太小。
- 它□可能速度太慢。
- 它□可能也太不可□。

□□□些□□， 人□提出并□□了□多不同的解决方案。 □了□□□些□□， 一些用□采用了多个， 有□甚至是冗余的磁□□□方法。 除了支持□多□不同的硬件 RAID 控制器之外， FreeBSD 的基本系□中包括了 Vinum 卷管理器，它是一个用以□□虚□磁□□□器的□□□。 *Vinum* 是一□称□ 卷管理器， 或者□用于解决前面□三□□□的虚□磁□□□程序。 Vinum 能□提供比□□磁□系□更好的□活性、 性能和可□性， 并□□了能□□独或配合使用 RAID-0、RAID-1 和 RAID-5 模型。

□一章□□□磁□存□的潜在□□□行了□要□明，并介□了 Vinum 卷管理器。

从 FreeBSD 5 □始， □ Vinum □行了重写， 以便使其符合 GEOM 架□ (GEOM. 模□化磁□□□框架)， 同□保留其原有的□□□意、 □□， 以及保存在磁□上的元数据格式。□一重写的版本称□ *gvinum* (表示 *GEOM vinum*)。 接下来的文字中 *Vinum* 是一个抽象的名字， 通常并不具体指某一特定的□□。 新版本中所有的指令都□通□ `gvinum` 命令来操作， 而□□的内核模□的名字， 也由 vinum.ko 改□了 geom_vinum.ko， 而在 /dev/vinum 中的所有□□□点， 也改□放到了 /dev/gvinum。 从 FreeBSD 6 □始， 旧版的 Vinum □□已不再提供。

22.2. 磁□容量太小

磁□越大，存□的数据也就越多。□□常会□□□需要 一个比□可使用的磁□大得多的文件系□。 无可否□，□个□□ 已□没有十年前那□□峻了，但它仍然存在。通□□建一个在□多 磁□上存□数据的抽象□□，一些系□可以解决□个□□。

22.3. □□瓶□

□代系□□常需要用一个高度并□的方式来□□数据。 例如，巨大的FTP 或HTTP 服□器可以支持数以千□的并□会□， 可以有多个□到外部世界的100 Mbit/s , □□□地超□了□大多数磁□的数据□□速率。

当前的磁□□□器最高可以以70 MB/s的速度□□数据， 但□个□在一个有□多不受□束的□程□□一个□□器的□境中□得并不重要， 它□可能只完成了□些□的一小部分。□□一□情况下，从磁□子 系□的角度来看□□就更加有趣：重要的参数是在子系□上的□ 荷，□句□□是□□占用了□□器多少□□。

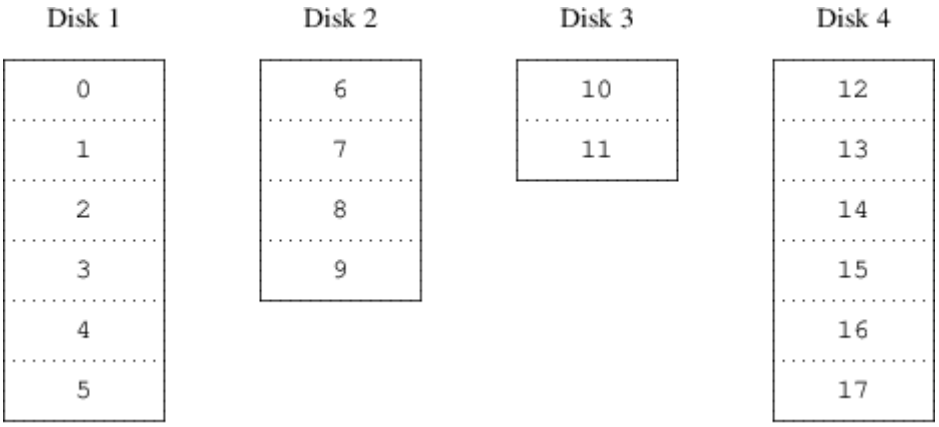
在任何磁□□□中， □□器必□先□道， 等待磁□□□第一个扇区， 然后□行□□. □些□作看起来可能很□小: 我□不会感有任何中断。

假□□□10 kB数据， ：□在的高性能磁□平均□道□□是3.5ms。 最快的□□器可以旋□在15,000 rpm，，所以平均□址□□□2ms. 在70 MB/s的速度□□□，数据的□□□□大□150 µs，几乎无法和□址□□相比. 在□□一□情况下，高效的□□也会降低到 1 MB/s □然□□的快慢依□与所□□数据的大小。

□于□个瓶□的一般和明□的解决方法是采用 "多个磁□":而不是只使用一个大磁□, 它使用几个比□小的磁□□合起来形成一个大的磁□. □个磁□都可以独立地□行□□数据，所以通□使用多个磁□ 大大提高了数据吞吐量。

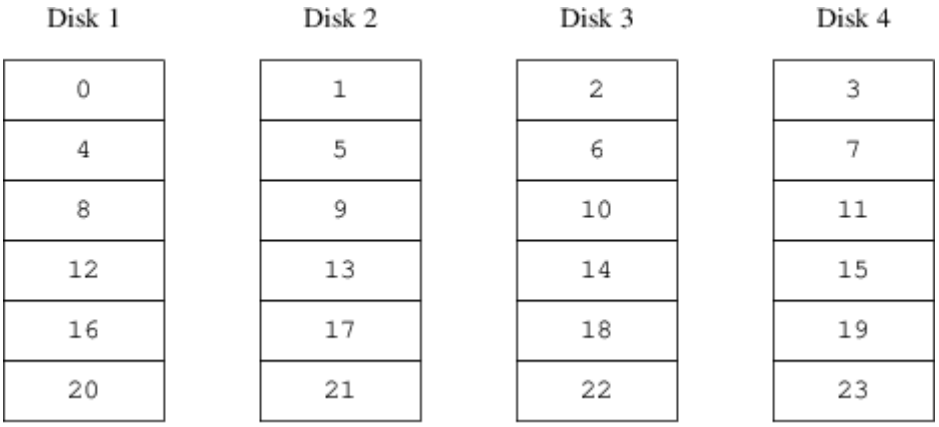
当然，所要求的吞吐量的提高要比磁□的数量小得多。 尽管□个□□器并行□□数据，但没有□法□保□求能□平均分配到□个□□器上。不可避免一个□□器的□□可能比□一个要高得多。

磁□的□□平衡很大程度依□于□□器上数据的共享方式. 在下面的□□中, 将磁□存□想象成一个巨大的数据扇区，像一本□的□ 那□用□号来□定地址. 最明□的方法是把虚□磁□分成□多□□的扇区□，□个扇区大小就是独立的磁□大小，用□□方法来存□数据, 就像把一本厚厚的□分成很多小的章□。□个方法叫做 串□ 它有一个□点就是磁□不需要有任何特定的大小□系。 当□□到的虚□磁□根据它的地址空□来分布的□候， 它能工作得很好。 当□□集中在一个比□小的区域的□候，性能的提高没有□著的改□。 串□□□例□明了用串□□□的方式来分配存□□元的□序。



□ *103.* 串□□□

□外一□影射方法是把地址空□分布在比□小的容量相同的磁□上， 从而能□在不同的□□上存□它□。例如，前256个扇区可能存□在第一 个磁□上，接着的256 个扇区存□在□一个磁□上等等。 写□最后一个磁 □后，□程会重□以前的工作，直到所有的磁□被写□。□个影射叫做 分段*(striping)* 或者 RAID-0. 分段要求很精□地□址，通□多个磁□□行数据□□的□候，它 可能会引起□外的I/O □□，但它也可能提供更多的□□□□。 分段□□□示了用分段形式分配的存□□元的□序。



□ *104.* 分段□□

22.4. 数据的完整性

□□磁□的最后一个□□是它□不太可□。 □然磁□□□器的可□性在□去几年有了很大的提高， 但它□仍然是服□器中最容易□坏的核心□件。 当它□□生故障的□候， □果可能是□□性的： 替□坏的磁□□□器并恢□数据可能要花□几天□□。

解决□个□□的□□方法是建立 □象， 在不同的物理硬件上□数据做□个副本。 根据 RAID □□出□的□□□序， □个技□也被叫做 RAID □□1 或者 RAID-1。 任何写到卷的数据也会被写到□象上， 所以可以从任何一个副本□取数据， 如果其中有一个出□故障， 数据也□可以从其他□□器上□□到。

□象有□个□□：

- 价格. 它需要□倍的存□容量。
- 性能影□。 写入操作必□在□个□□器上□行，所以它□ 花□□倍的□□。□取数据并不会影□性能： 它□甚至看起来会更快。

一个可□的方案采用 奇偶校□ 的方式， 用以□□ RAID 2、 3、 4 和 5。 □其中， RAID-5 是我□最感□趣的。 在 Vinum 的□□中， □是一个条□□□□□的□体， 其中， □一个条□中都以一个□用的□， 来保存其它□的奇偶校□□。□□， RAID-5 plex 除了在□个□中都包含了一个奇偶校□□之外， □□ RAID-5 □也就和普通的条□ plex 一□了。作□ RAID-5 的一□要求， 奇偶校□□在□一个条□中的□序都是不同的。 数据□的□号， 决定了它的相□□号。

Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	Parity
3	4	Parity	5
6	Parity	7	8
Parity	9	10	11
12	13	14	Parity
15	16	Parity	17

□ *105. RAID-5* 的□□

与□像相比， RAID-5 最□著的□□在于只需使用少得多的存□空□。 □取□似于条□式存□的□□， 但写入会慢得多，大□□相当于□性能的 25%。 如果一个□□器失效， □□列仍然可以在降□的模式□行： □取来自正常的□□器数据的操作照常□行， 但□取失效的□□器的数据， □来自于余下□□器上相□的□算□果。

22.5. Vinum 目□

□了解决□些□□，Vinum 提出了一个四□的目□□□：

- 最□著的目□是虚□磁□， 叫做 卷*(volume)*. 卷本□上与一个UNIX 磁□ □□器有同□的属性，□然它□是有些不太一□。它□没有大小的限制。
- 卷下面是 *plexes*, □一个表示卷的所有地址空□。在□次□□中的□个水平能□提供 冗余功能。可以把plex想象成用一个□象排列的方式□□起来的 独立磁□，□个都包含同□的数据。
- 由于Vinum 存在于UNIX 磁□存□框架中,所以它也可能 使用UNIX 分区作□多个磁□plex 的□成部分, 但事□上□并不可□:UNIX 磁□只能有有限数量的分区。 取而代之，Vinum 把一个□□的UNIX 分区 (the *drive*) 分解成叫做*subdisks*的相□区域, 它可以使用□个 来□plex 建立□。
- Subdisks 位于 Vinum □□器上, 当前的UNIX 分区。Vinum □□器可以包含很多的subdisks。 除了□□器□始的一小□区域用来存□配置和描述信息以外，整个 □□器都可以用于存□数据。

下面的章□描述了□些目□提供了Vinum 所要求的功能的方法。

22.5.1. 卷的大小要求

在Vinum的配置中，Plex可以把多个subdisk 分布在所有的□□上。 □果， □个独立的□□器的大小都不会限制plex 的大小，从而不会限制卷的大小

22.5.2. 多余的数据存□

Vinum 通□□一个卷□上多个plex 来完成□象的功能。 □个plex 是一个在一个卷中的数据的描述。一个卷可以包含一个 到八个plex。

□然一个plex 描述了一个卷的所有数据，， 但可能描述的部分被物理地□失了。可能是□□的□□ （没有□plex 部分定□一个subdisk）也可能是意外的故障 （由于□□器的故障□致）。只要至少有一个plex 能□□卷的完全地址□□提供数据，卷就能□正常工作。

22.5.3. 性能□□

Vinum 在plex 水平既□行串□也□行分段：

- 一个串□的*plex*□流使用 □个subdisk 的地址空□。
- 一个 分段的*plex* 在□个subdisk 上 □分数据. Subdisk 必□是大小一□的，□了从一个□接的plex 中 区分□它，必□至少有□个subdisk。

22.5.4. □□**plex** □□更有效？

FreeBSD 12.0提供的Vinum 版本能□□□□plex:

- 串□的plex 更加□活：它□可以包含任何数量的subdisk， subdisk 也可能有不同的□度。Plex 可以通□添加□外的subdisk 来得到□展。 与分段 plex 不同， 它□需要的 CPU □□更少， 尽管 CPU 上的□□差□是不可□量的。 □一方面，它□的□□可能不平衡，一个磁□可能□□很重， 而其他的可能很空□。
- 分段(RAID-0) plexes 的最大□点是 它□□少了□□不平衡的情况: 通□□□一个最合□大小的分段 (大□是256 kB), □甚至可以在各个□成的□□器上降低□□ . □□方法的缺点是在subdisk 上受到非常□□的□□限制 ： 它□必□是同□大小, 通□添加新的subdisk 来□展一个plex 是非常□□的,以至Vinum 当前没有□□它. Vinum 利用一个□外的，代价不高的限制：一个分段的plex 必□有至少□个subdisk， 否□， 它就无法区分□接的 plex 了。

Vinum Plex□□□ □□一下□个plex □□ 的□点和缺点.

表 *8. Vinum Plex*□□□

Plex □型	最少**subdisks**	可否添加**subdisks**	尺寸相同	□用
串□	1	可以	不必□	□有很大□性和□中性能的大数据量存□。
分段	2	不可以	必□	大量并□□□,具有□高性能。

22.6. 一些例子

Vinum □□着一个描述本系□中□象的 配置数据□。 □始□， 用□可以在 gvinum(8) 工具来从若干配置文件生成配置数据□。 Vinum 在其控制的□个磁□分区 (在 Vinum 中称□ *device*)

上都保存配置数据□的副本。 □一数据□在□次状□□化□均会更新， 因而重□□个 Vinum □象□， 都能□恢□其状□。

22.6.1. 配置文件

配置文件描述了独立的 Vinum.一个□□卷的定□可能是□□的:

```
    drive a device /dev/da3h
    volume myvol
      plex org concat
        sd length 512m drive a
```

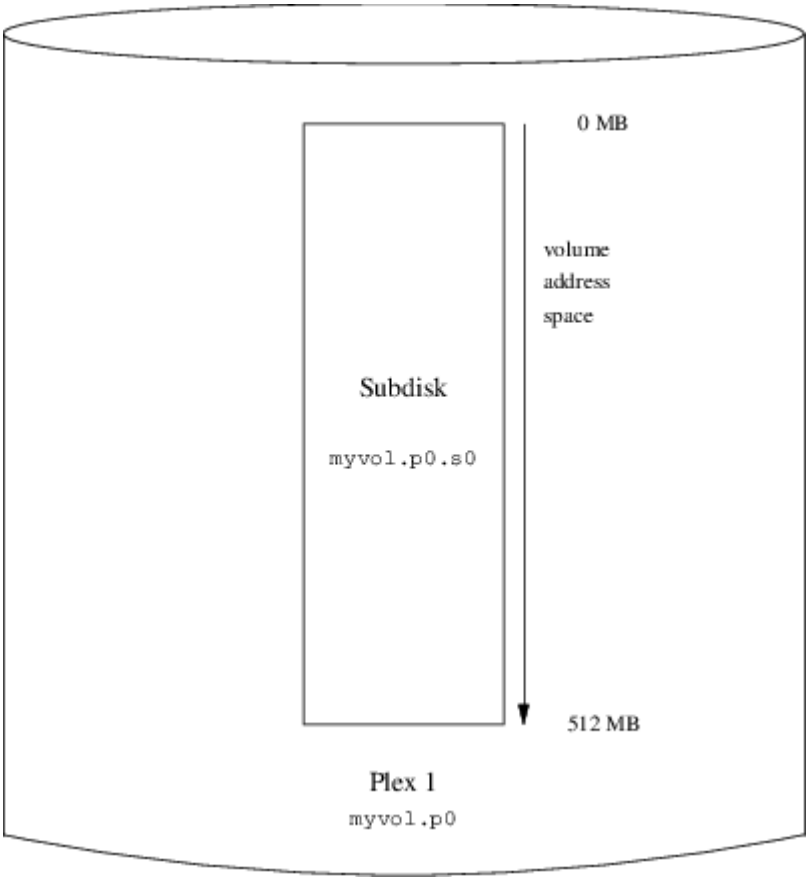
□个文件描述了四个Vinum 目□:

- *drive* 行描述了一个磁□分区（□□器） 和与下面的硬件相□的它的位置。它□出了一个符号名 *a*. □个与□□名称分□的符号名允□ 磁□从一个位置移□到□一个位置而不会□混。
- *volume* 行描述了一个卷。 唯一的必□属性是名称，在□个例子中是 *myvol*.
- *plex* 行定□了一个plex。 唯一需要的参数是□□,在□个例子中是 *concat*. 没有名称是必然的: 系□自□通□添加suffix *.px* 来从卷名称□生一个名字,□里的*x* 是在卷中的plex 的□号。而□个plex 将被 叫做*myvol.p0*。
- *sd* 行描述了一个subdisk。 最小的□明是存□subdisk 的□□器名称，和subdisk 的□度。 □于plex，没有名称也是必然的：系□自□通□添加 suffix *.sx* 来分配源自plex 的名称，□里 *x*是plex 中subdisk 的□号。 Vinum □□个subdisk 命名□*myvol.p0.s0*。

□理完□个文件后， gvinum(8) 会□生下面的□出：

```
      #  gvinum -> create config1
      Configuration summary
      Drives:         1 (4 configured)
      Volumes:        1 (4 configured)
      Plexes:         1 (8 configured)
      Subdisks:       1 (16 configured)

    D a                     State: up       Device /dev/da3h        Avail: 2061/2573
MB (80%)

    V myvol                 State: up       Plexes:       1 Size:        512 MB

    P myvol.p0            C State: up       Subdisks:     1 Size:        512 MB

    S myvol.p0.s0           State: up       PO:        0  B Size:        512 MB
```

□些□出内容展示了 gvinum(8) 的□要列表格式。 在 一个□□的Vinum 卷 中用□形展示了□个配置。



□ *106.* 一个□□的*Vinum* 卷

下面□个□□示了一个由按□序排列的subdisk □成的plex。 在□个小小的例子中，卷包含一个plex，plex 包含一个subdisk。

□个卷本身和普通的磁□分区相比并没有什□特□的□越性， 它包含了一个 plex， 因此不是冗余的。 □个 plex 中包括了一个子磁□， 因此□和从磁□分区分配存□没什□□□。 接下来的几□， 将介□一些更有用的配置方法。

22.6.2. 提高容□性： □像

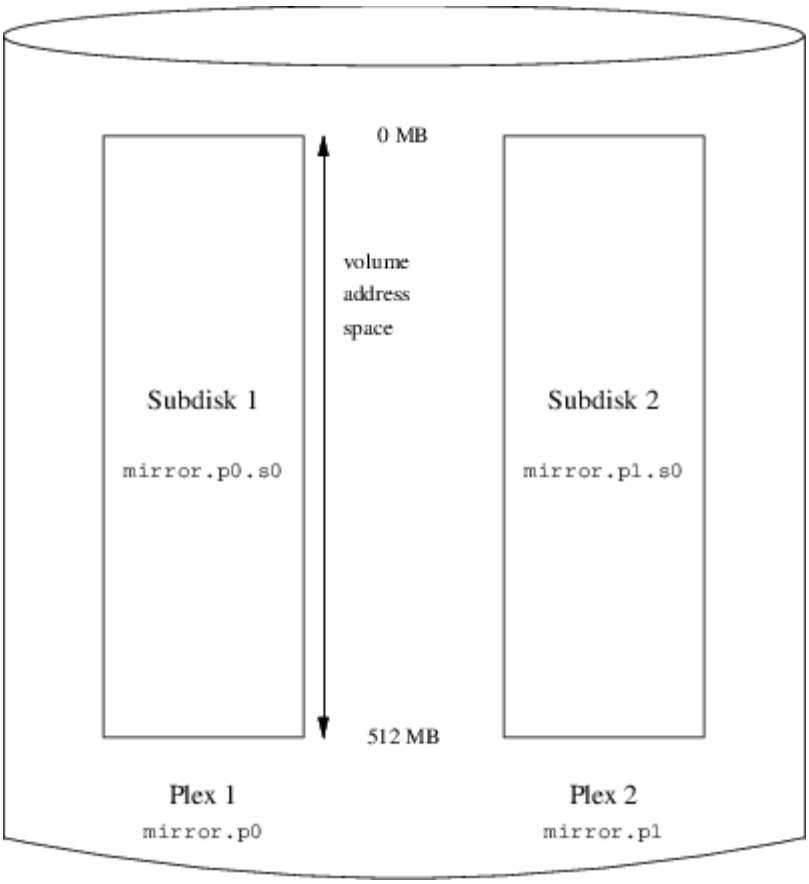
卷的容□性可以通□□像来提高。 在配置□像卷□， □保 plex 分布在不同的□□器上十分重要， □□一个□□器坏掉□， 就不会同□影□□个 plex。 下面的配置将映射卷：

```
drive b device /dev/da4h
volume mirror
  plex org concat
    sd length 512m drive a
  plex org concat
    sd length 512m drive b
```

上面的例子中， 并不需要再次指定□□器 *a*， 因□ Vinum □控所有其配置数据□的□象。 完成定□之后，配置如下所示：

```
    Drives:         2 (4 configured)
    Volumes:        2 (4 configured)
    Plexes:         3 (8 configured)
    Subdisks:       3 (16 configured)

    D a                     State: up       Device /dev/da3h        Avail: 1549/2573
MB (60%)
    D b                     State: up       Device /dev/da4h        Avail: 2061/2573
MB (80%)

    V myvol                 State: up       Plexes:       1 Size:        512 MB
    V mirror                State: up       Plexes:       2 Size:        512 MB

    P myvol.p0            C State: up       Subdisks:     1 Size:        512 MB
    P mirror.p0           C State: up       Subdisks:     1 Size:        512 MB
    P mirror.p1           C State: initializing     Subdisks:     1 Size:        512
MB

    S myvol.p0.s0           State: up       PO:        0  B Size:        512 MB
    S mirror.p0.s0          State: up       PO:        0  B Size:        512 MB
    S mirror.p1.s0          State: empty    PO:        0  B Size:        512 MB
```

□像 Vinum 卷 以□形方式展示了其□□。



□ *107.* □像 *Vinum* 卷

□个例子中， □一个 plex 包含了完整的 512 MB 地址空□。 在前面的例子中， plex □只包括一个子□。

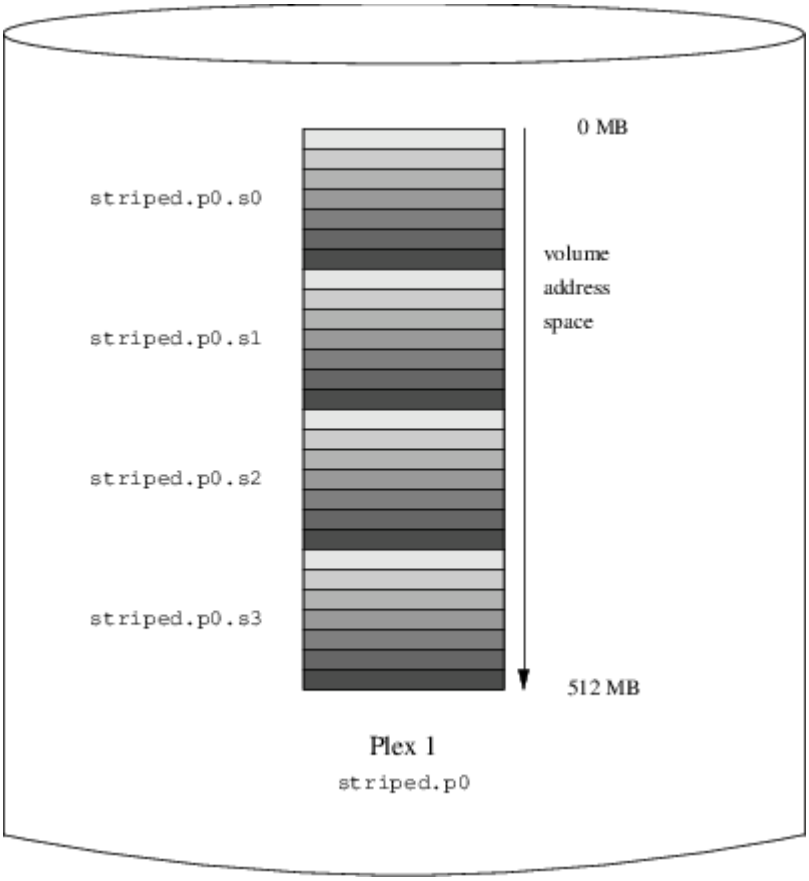
22.6.3. □化性能

前面例子中的□像卷要比没有□像的卷具有更好的容□能力， 但它的性能要差一些： □一次写入卷□， 需要同□写到□个□□器上， 因而也就需要更大的磁□□□□。 如果希望非常好的性能， □需要□外一□方式： 不做□像，而将数据分成条□放到尽可能多的、不同的磁□上。 下面□出了一个跨越四个磁□□□器的 plex 卷：

```
    drive c device /dev/da5h
    drive d device /dev/da6h
    volume stripe
    plex org striped 512k
      sd length 128m drive a
      sd length 128m drive b
      sd length 128m drive c
      sd length 128m drive d
```

和之前□似， 并不需要定□ Vinum 已□知道的□□器。 在完成定□之后， 将得到如下配置：

```
    Drives:         4 (4 configured)
    Volumes:        3 (4 configured)
    Plexes:         4 (8 configured)
    Subdisks:       7 (16 configured)

    D a                     State: up       Device /dev/da3h        Avail: 1421/2573
MB (55%)
    D b                     State: up       Device /dev/da4h        Avail: 1933/2573
MB (75%)
    D c                     State: up       Device /dev/da5h        Avail: 2445/2573
MB (95%)
    D d                     State: up       Device /dev/da6h        Avail: 2445/2573
MB (95%)

    V myvol                 State: up       Plexes:       1 Size:        512 MB
    V mirror                State: up       Plexes:       2 Size:        512 MB
    V striped               State: up       Plexes:       1 Size:        512 MB

    P myvol.p0            C State: up       Subdisks:     1 Size:        512 MB
    P mirror.p0           C State: up       Subdisks:     1 Size:        512 MB
    P mirror.p1           C State: initializing     Subdisks:     1 Size:        512
MB
    P striped.p1            State: up       Subdisks:     1 Size:        512 MB

    S myvol.p0.s0           State: up       PO:        0  B Size:        512 MB
    S mirror.p0.s0          State: up       PO:        0  B Size:        512 MB
    S mirror.p1.s0          State: empty    PO:        0  B Size:        512 MB
    S striped.p0.s0         State: up       PO:        0  B Size:        128 MB
    S striped.p0.s1         State: up       PO:      512 kB Size:        128 MB
    S striped.p0.s2         State: up       PO:     1024 kB Size:        128 MB
    S striped.p0.s3         State: up       PO:     1536 kB Size:        128 MB
```



□ *108.* 条□化的 *Vinum* 卷

□个卷在 条□化的 Vinum 卷 中□出。 条□的□影部分， 表示在 plex 地址空□中的位置： □色最浅的在最前面，而最深的在最后。

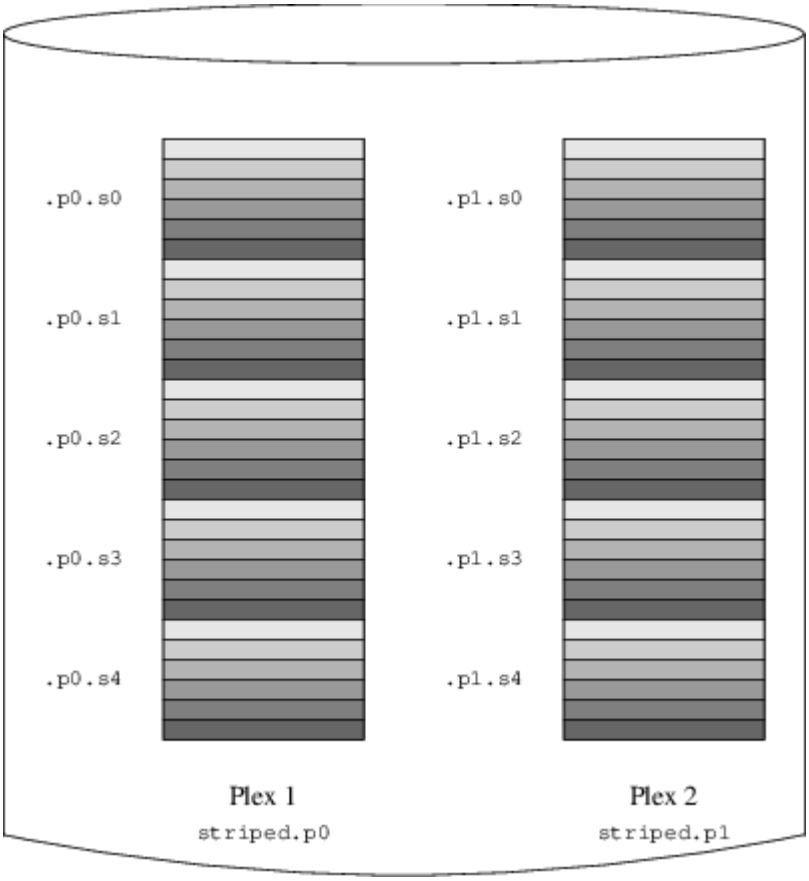
22.6.4. 高性能容在

如果硬件足□多， 也能□□建比□准 UNIX® 分区同□提高了容□性和性能的卷。 典型的配置文件□似：

```
volume raid10
  plex org striped 512k
    sd length 102480k drive a
    sd length 102480k drive b
    sd length 102480k drive c
    sd length 102480k drive d
    sd length 102480k drive e
  plex org striped 512k
    sd length 102480k drive c
    sd length 102480k drive d
    sd length 102480k drive e
    sd length 102480k drive a
    sd length 102480k drive b
```

第二个 plex 中的子□和第一个 plex 中的□□了□个□□器： □能□□助□保即使同□□□□个□□器， 写操作也不会同□□生在同一个□上。

□像并条□化的 Vinum 卷 □出了□卷的□□。



□ *109.* □像并条□化的 *Vinum* 卷

22.7. □象命名

如前面所描述的那□， Vinum 会□ plex 和子□指定默□的名字， 而□些名字也是可以定制的。 不推□修改默□的名字： 使用允□□□象任意命名的 VERITAS 卷管理器的□□□明， □一□活性并没有□来太多的好□， 相反，它很容易□致□象的混□。

名字中可以包括任何非空白的字符， 但一般来□， 建□只使用字母、 数字和下□□。 卷、 plex， 以及子□的名字， 可以包含最多 64 个字符， 而□□器的名字， □最□可以使用 32 个字符。

Vinum □象会在 /dev/gvinum 之下生成□□□点。 前述的配置将使 Vinum □建以下□□□点：

- □个卷□□的□□□。 □些是 Vinum 使用的主要□□。 因此， 前述配置包括下列□□： /dev/gvinum/myvol、/dev/gvinum/mirror、 /dev/gvinum/striped、 /dev/gvinum/raid5 以及 /dev/gvinum/raid10。
- 所有卷的直接□都存放在 /dev/gvinum/ 中。
- 目□ /dev/gvinum/plex， 以及 /dev/gvinum/sd 中相□地存放了□个 plex 以及子□的□□□点。

例如， 考□下面的配置文件：

```
drive drive1 device /dev/sd1h
drive drive2 device /dev/sd2h
drive drive3 device /dev/sd3h
drive drive4 device /dev/sd4h
volume s64 setupstate
  plex org striped 64k
    sd length 100m drive drive1
    sd length 100m drive drive2
    sd length 100m drive drive3
    sd length 100m drive drive4
```

□理□个文件之后， gvinum(8) 将在 /dev/gvinum 中建立下面的□□：

```
drwxr-xr-x  2 root  wheel       512 Apr 13 16:46 plex
crwxr-xr--  1 root  wheel   91,   2 Apr 13 16:46 s64
drwxr-xr-x  2 root  wheel       512 Apr 13 16:46 sd

/dev/vinum/plex:
total 0
crwxr-xr--  1 root  wheel   25, 0x10000002 Apr 13 16:46 s64.p0

/dev/vinum/sd:
total 0
crwxr-xr--  1 root  wheel   91, 0x20000002 Apr 13 16:46 s64.p0.s0
crwxr-xr--  1 root  wheel   91, 0x20100002 Apr 13 16:46 s64.p0.s1
crwxr-xr--  1 root  wheel   91, 0x20200002 Apr 13 16:46 s64.p0.s2
crwxr-xr--  1 root  wheel   91, 0x20300002 Apr 13 16:46 s64.p0.s3
```

□然 plex 和子□一般并不推□指定名字， 但□是必□□ Vinum □□器命名。 □□， 当把□□器□移到不同的地方□，它仍然能□被自□地□□出来。 □□器名最□可以包含 32 个字符。

22.7.1. □建文件系□

□于系□而言， 卷和磁□是一□的。 唯一的例外是， 与 UNIX® □□器不同， Vinum 并不□卷□行分区，因而它也就不包含分区表。 □要求修改某些磁□工具， 特□是 newfs(8)， 它会□□将 Vinum 卷名当作分区□□。例如， 磁□□□器的名字可能是 /dev/ad0a 或 /dev/da2h。 □些名字分□表□在第一个 (0) IDE (ad) 磁□上的第一个分区 (a)， 以及第三个 (2) SCSI 磁□ (da) 上的第八个分区 (h)。 而相比而言， Vinum 卷可能叫做 /dev/gvinum/concat， □个名字和分区名没有什□□系。

要在□个卷上□建文件系□， □需要使用 newfs(8)：

```
# newfs /dev/gvinum/concat
```

22.8. 配置 Vinum

在 GENERIC 内核中， 并不包含 Vinum。 可以□□一个定制的包含 Vinum 的内核， 然而并不推□□□做。 □□

Vinum 的□准方法， 是使用内核模□ (kld)。 甚至不需要使用 kldload(8) 来□□ Vinum： 在□□ gvinum(8) □，它会□□□一模□是否已□加□， 如果没有， □会自□地加□它。

22.8.1. □□

Vinum 将配置信息， 采用与配置文件一□的形式来存放到磁□分区上。 当从配置数据□中□取□， Vinum 会□□一系列在配置文件中不可用的□□字。 例如， 磁□配置文件可能包含下面的文字：

```
volume myvol state up
volume bigraid state down
plex name myvol.p0 state up org concat vol myvol
plex name myvol.p1 state up org concat vol myvol
plex name myvol.p2 state init org striped 512b vol myvol
plex name bigraid.p0 state initializing org raid5 512b vol bigraid
sd name myvol.p0.s0 drive a plex myvol.p0 state up len 1048576b driveoffset 265b
plexoffset 0b
sd name myvol.p0.s1 drive b plex myvol.p0 state up len 1048576b driveoffset 265b
plexoffset 1048576b
sd name myvol.p1.s0 drive c plex myvol.p1 state up len 1048576b driveoffset 265b
plexoffset 0b
sd name myvol.p1.s1 drive d plex myvol.p1 state up len 1048576b driveoffset 265b
plexoffset 1048576b
sd name myvol.p2.s0 drive a plex myvol.p2 state init len 524288b driveoffset 1048841b
plexoffset 0b
sd name myvol.p2.s1 drive b plex myvol.p2 state init len 524288b driveoffset 1048841b
plexoffset 524288b
sd name myvol.p2.s2 drive c plex myvol.p2 state init len 524288b driveoffset 1048841b
plexoffset 1048576b
sd name myvol.p2.s3 drive d plex myvol.p2 state init len 524288b driveoffset 1048841b
plexoffset 1572864b
sd name bigraid.p0.s0 drive a plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 0b
sd name bigraid.p0.s1 drive b plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 4194304b
sd name bigraid.p0.s2 drive c plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 8388608b
sd name bigraid.p0.s3 drive d plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 12582912b
sd name bigraid.p0.s4 drive e plex bigraid.p0 state initializing len 4194304b driveoff
set 1573129b plexoffset 16777216b
```

□里最明□的区□是， 指定了配置的位置信息、名称 (□些在配置文件中□是可用的， 但不鼓励用□自行指定) 以及状□信息 (□是用□不能指定的)。 Vinum 并不在配置信息中保存□于□□器的信息： 它会□描已□配置的磁□□□器上包含 Vinum □□的分区。 □使得 Vinum 能□在 UNIX® □□器被指定了不同的 ID □也能□正□□□它□。

22.8.1.1. 自□□□

Gvinum 在通□ loader.conf(5) 加□了内核模□之后就能自□□□。 在□□□加□ *Gvinum* 模□， 需在 /boot/loader.conf 中加入 `geom_vinum_load="YES"`。

当使用 gvinum start 命令来□□ Vinum □， Vinum 会从某一个 Vinum □□器中□取配置数据□。 正常情况下，□个□□器上都包含了同□的配置数据□副本， 因此从□个□□器上□取是无所□的。 但是， 在系□崩□之后， Vinum 就必□□□□一个□□器上的配置数据□是最新的， 并从上面□取配置。 如果需要， 它会更新其它□□器上的配置。

22.9. 使用 Vinum 作□根文件系□

如果文件系□使用完全□像的 Vinum 配置， 有□也会希望根文件系□也作了□像。 □□配置要比□像其它文件系□麻□一些， 因□：

- 根文件系□在引□□程中很早的□候就必□□于可用状□， 因此 Vinum 的基□□施在□一□刻就□□可用了。
- 包含根文件系□的卷， 同□也保存了系□的引□程序和内核， 因此它□必□能□被宿主系□的内建工具 (例如 PC 机的 BIOS) □□， 而通常是没□法□它□了解 Vinum 的□□的。

下面几□中， □□ "根卷" □□包含根文件系□的 Vinum 卷。 把□个卷命名□ "root" 可能是个不□的主意， 不□从技□上□， 并不□格地要求□□做。 不□， 接下来的命令例子都使用□个名字。

22.9.1. 及早□□ Vinum 以□□□根文件系□的要求

有□多□于它的尺度：

- Vinum 必□在□□□可以被内核使用。 因此， 在 自□□□ 中所介□的方法， 也就无法□□□一任□的需要了。 在接下来的配置中， 也 不能 □置 start_vinum 参数。 第一□方法是通□将 Vinum 静□□□到内核中来□□，□□， 它就在任何□候都可用了， □然一般并不需要□□。 □一□方法是通□ /boot/loader (第三□段，/boot/loader) 来尽早加□ vinum 内核模□， □一操作□生在内核加□之前。 □可以通□将下面的配置：

  ```
  geom_vinum_load="YES"
  ```

  加入到 /boot/loader.conf 文件中来□□。

- □ *Gvinum* 而言， 所有的□□□程都是在内核模□加□□自□□行的， 因此上面的操作， 也就是所要□行的全部工作了。

22.9.2. □基于 Vinum 的卷在引□□可以□□

因□目前的 FreeBSD 引□程序只有 7.5 KB 的代□， 并且已□承担了从 UFS 文件系□中□取文件 (例如 /boot/loader) 的重任， 因此完全没有□法再□它去分析 Vinum 配置数据中的 Vinum □□， 并□到引□卷本身的信息。 因此， 需要一些技巧来□引□代□提供□准的 "a" 分区， 而它□包含了根文件系□。

要□□些得以□□， 根卷需要□足下面的条件：

- 根卷不能是条□卷或 RAID-5 卷。
- 根卷 plex 不能包含□接的子□。

需要□明的是， 使用多个 plex， □个 plex 都□制一□根文件系□的副本， 是需要而且是可行的。 然而， 引□□程只能使用□些副本中的一个来引□系□， 直到内核最□自行挂接根文件系□□止。 □些 plex 中的□个子□， 在□之后会有它□自己的 "a" 分区， 以表□□一个可以引□的□□。 □一个 "a" 分区， 尽管并不需要和其它包含根卷的 plex □于各自□□器的同一位置。 但是， □□□建 Vinum 卷使得□像卷相互□称， 从而能□避免了混□。

□了□建□一个根卷的 "a" 分区， 需要完成下面的操作：

1. 使用下面的命令来了解根卷成□子□的位置 (从□□□始的偏移量) 和尺寸：

   ```
   # gvinum l -rv root
   ```

   需要注意的是， Vinum 偏移量和尺寸的□位是字□。 它□必□是 512 的整数倍， 才能得到 bsdlabel 命令所需的□号。

2. 在□一个根卷成□□□上， □行命令：

   ```
   # bsdlabel -e devname
   ```

   □其中， □于没有 slice (也就是 fdisk) 表的磁□， *devname* 必□是磁□的名字 (例如 da0)， 或者是 slice 的名字 (例如 ad0s1)。

   如果□□上已□有了 "a" 分区 (比如□， 包含 Vinum 之前的根文件系□)， □□改□其它的名字， 以便□□□□ (如果需要的□)， 但它并不会□□用于□□系□。 注意， 活□的分区 (□似正挂接的根文件系□) 不能被改名， 因此， 要完成□□工作， 必□从 "Fixit" □□□， 或者分□□操作， 并 (在□像情形中) 首先操作那些非引□□。

   然后， □□上 Vinum 分区的偏移 (如果有的□) 必□加到□个□□上根卷□□的子□上。 其□果□， 将成□新的 "a" 分区的 "offset" □。 □个分区的 "size" □， 可以根据前面的配置□算得出。 "fstype" □□是 4.2BSD。 "fsize"、 "bsize"， 以及 "cpg" □， □□与文件系□的□□情况匹配， 尽管在配置 Vinum □并不重要。

   □□， 新的 "a" 分区， 将□建并覆□□一□□上的 Vinum 分区的□□。 注意， bsdlabel 只有在 Vinum 分区的 fstype 被□□□ "vinum" □， 才允□□□做。

3. □就成了！ 所有的 "a" 分区□在都已存在， 而且是根卷的一□副本。 □烈建□□再次□□其□果，方法是：

   ```
   # fsck -n /dev/devnamea
   ```

□必注意， 所有包含控制信息的文件， 都必□放到 Vinum 卷上的根文件系□。 在□□新的 Vinum 根卷□， 它□可能和□□在用的根文件系□不匹配。 因此， /etc/fstab 和 /boot/loader.conf □□个文件需要特□地注意。

在下次重□□， 引□程序需要从新的基于 Vinum 的根文件系□中□取□当的控制信息， 并据此工作。在内核初始化□程的□尾部分， 在所有的□□都被宣示之后， 如果□示了下面的信息， □表示配置成功：

```
Mounting root from ufs:/dev/gvinum/root
```

22.9.3. 基于 Vinum 的根文件系□的配置□例

在 Vinum 根卷配置好之后， `gvinum l -rv root` 的□出可能□似下面的□子：

```
...
Subdisk root.p0.s0:
        Size:        125829120 bytes (120 MB)
        State: up
        Plex root.p0 at offset 0 (0  B)
        Drive disk0 (/dev/da0h) at offset 135680 (132 kB)

Subdisk root.p1.s0:
        Size:        125829120 bytes (120 MB)
        State: up
        Plex root.p1 at offset 0 (0  B)
        Drive disk1 (/dev/da1h) at offset 135680 (132 kB)
```

需要注意的□是 135680， 也就是偏移量 (相□于 /dev/da0h 分区)。 □相当于 bsdlabel □法中的 265 个 512-字□的磁□□。 □似地， 根卷的尺寸是 245760 个 512-字□的磁□□。 /dev/da1h 中， 包含了根卷的第二个副本，采用了同□的配置。

□些□□的 bsdlabel □似下面的□子：

```
...
8 partitions:
#        size   offset    fstype   [fsize bsize bps/cpg]
  a:   245760      281    4.2BSD     2048 16384     0   # (Cyl.    0*- 15*)
  c: 71771688        0    unused        0     0         # (Cyl.    0 - 4467*)
  h: 71771672       16     vinum                       # (Cyl.    0*- 4467*)
```

可以看到， □装的 "a" 分区的 "size" 参数和前面的一□， 而 "offset" 参数□是 Vinum 分区 "h"， 以及□□中□一分区 (或 slice) 的偏移量之和。 □是一□典型的配置， 它能□避免在 无法□□， 引□程序□生 panic 中介□的□□。 此外， 我□也看到整个 "a" 分区完全□于□□上包含了 Vinum 数据的 "h" 分区之中。

注意， 在上面的配置中， 整个□□都是 Vinum □用的， 而且没有留下 Vinum 之前的根分区， 因□它永久性地成□了新建的 Vinum 配置中的一个子□。

22.9.4. 故障排除

如果遇到了□□， □需要从中恢□的□法。 下面列出了一些常□的缺陷， 及其解决方法。

22.9.4.1. 系□的引□程序加□了， 但无法□□

如果由于某□原因系□不再□□□□， 引□程序可以在 10-秒 倒□□的□候， 按 `space` □来停止。 加□器□量 (例如 `vinum.autostart`) 可以通□使用 `show` 命令来□看， 并使用 `set` 和 `unset` 命令来□置。

如果遇到的□□是由于 Vinum 的内核模□没有列入□加□的列表， 而没有正□加□， □□□使用 `load geom_vinum` 会有所□助。

此后， 可以使用 boot -as 来□□□□□程。 □□ -as 会要求内核□□所挂接的根文件系□ (-a)， 并使引□□程在□用□模式停止 (-s)， 此□根文件系□是以只□方式挂接的。 □□， 即使只挂接了多 plex 卷中的一个 plex，也不会引致 plex 之□数据不一致的□□。

当提示□入要挂接的根文件系□□， 可以□入任何一个包含根文件系□的□□。 如果正□地配置了 /etc/fstab， □默□的□□是□似 ufs:/dev/gvinum/root。 一般可以使用□似 ufs:da0d □□的□□来代替它， 因□它通常包括了 Vinum 之前的根文件系□。 需要注意的是， 如果在□里□入了 "a" 分区， □它可能表□的□□上是 Vinum 根□□的一个子□， 而在□像式配置中， □只会挂接□像的根□□中的一个。 如果之后将□个文件系□以□写方式挂接，□需要从 Vinum 根卷中□去其他的 plex， 否□□些卷中可能会包含不一致的数据。

22.9.4.2. 只加□了主引□程序

如果 /boot/loader 加□失□， 而主引□程序加□正常 (在□□□， 屏幕最左□一列有一个旋□的□)， □可以□□在此□中断主引□程序的□程， 方法是按 space □。 □将在引□的第二□段□停， 具体可以参□ 第一□段，/boot/boot1，和第二□段， /boot/boot2。 此□， 可以□□从□一个分区， 例如原先包含根文件系□，并不再叫作 "a" 的那个分区， □□。

22.9.4.3. 无法□□， 引□程序□生 panic

□□情况一般是由于 Vinum 安装□程中破坏了引□程序造成的。 不幸的是， Vinum 目前只在分区□始的地方保留了 4 KB 的空□， 之后就□始写 Vinum □信息了。 然而， 目前第一□段和第二□段的引□程序，加上 bsdlabel 嵌入的内容□需要 8 KB。 因此， 如果 Vinum 分区从偏移量 0 □始， 而□个 slice 或磁□能□□□， □ Vinum 的安装将□掉引□程序。

□似地， 如果从上述情形中恢□， 例如， 从 "Fixit" □□□， 并通□ bsdlabel -B 按照 第一□段，/boot/boot1，和第二□段， /boot/boot2 中介□的方法来恢□引□程序， □引□程序会覆□掉 Vinum □， □□ Vinum 也就□不到它的磁□了。 尽管□并不会真的□掉 Vinum 的配置数据， 或者 Vinum 卷上的数据，并且可以通□□入一模一□的 Vinum 配置数据来恢□， 但从□□状况中完全恢□是非常困□的。 要真正解决□□，必□将整个 Vinum 分区向后移□至少 4 KB， 以便使 Vinum □和系□的引□程序不再冲突。

Chapter 23. 虚□化

23.1. 概述

虚□化□件能□□同一台机器上同□□行多个操作系□。 在 PC 上， □□系□通常由一个□行虚□化□件的宿主操作系□，以及一系列客□操作系□□成。

□完□章， □将了解：

- 宿主操作系□与客□操作系□的区□。
- 如何在采用 Intel® □理器的 Apple® Macintosh® □算机上安装 FreeBSD。
- 如何在 Microsoft® Windows® 以 Virtual PC 安装 FreeBSD。
- 如何□□虚□化□境□ FreeBSD 系□□行性能□□。

在□□□章之前， □□：

- 理解 UNIX® 和 FreeBSD 的基□知□ (UNIX 基□)。
- 了解如何安装 FreeBSD (安装 FreeBSD)。
- 了解如何配置网□□接 (高□网□)。
- 了解如何安装第三方□件 (安装□用程序. Packages 和 Ports).

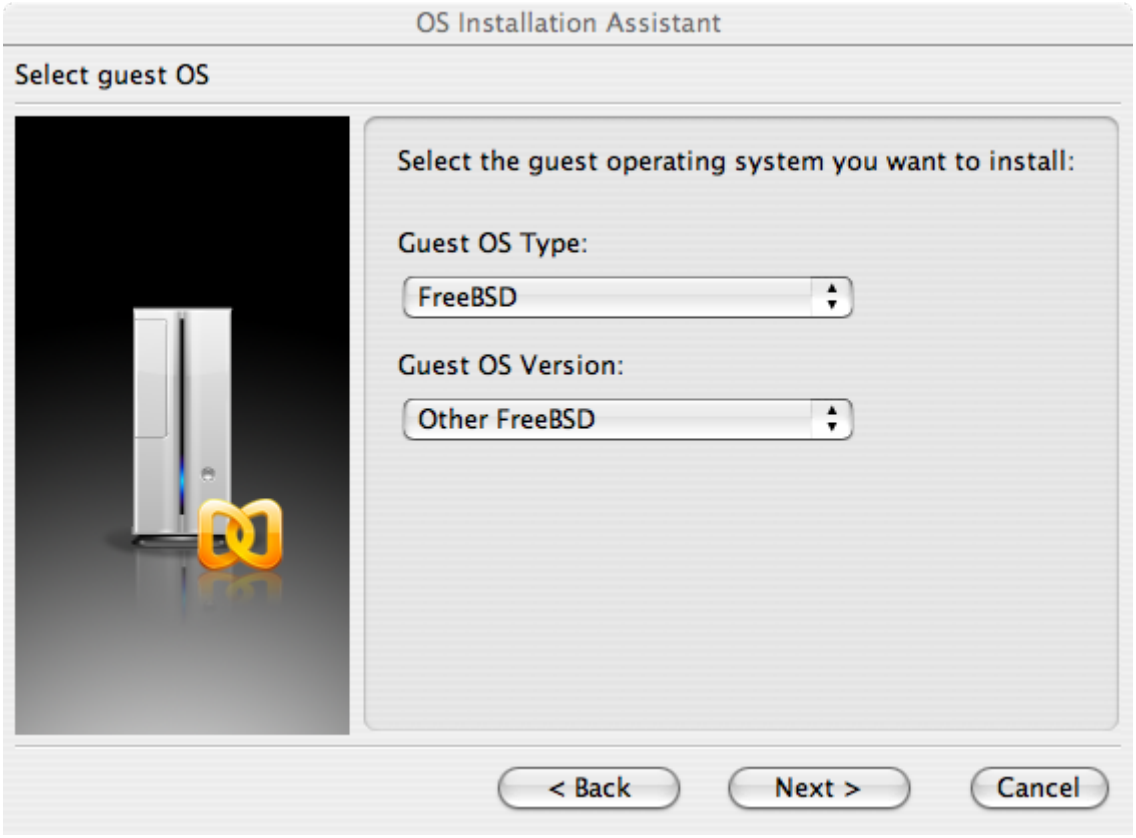
23.2. 作□客□ OS 的 FreeBSD

23.2.1. MacOS 上的 Parallels

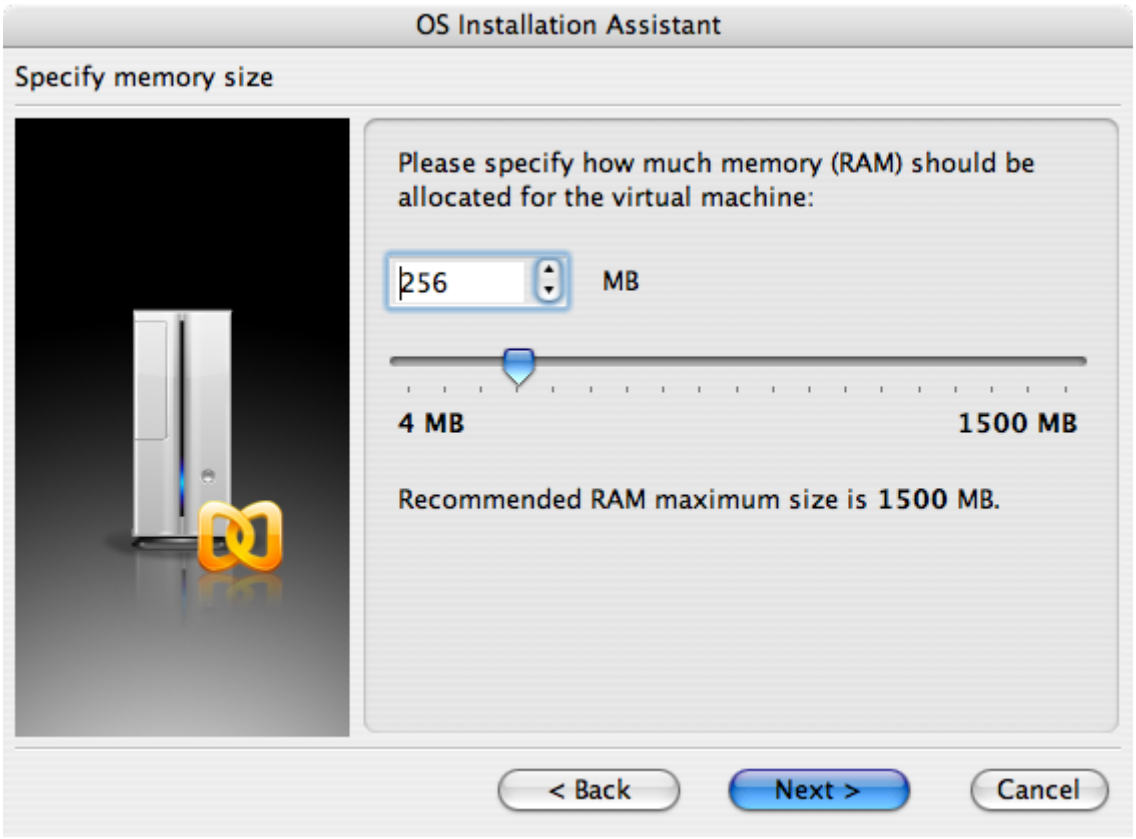
□ Mac® □□的 Parallels Desktop 是一□可用于采用 Intel® □理器， 并□行 Mac OS® 10.4.6 或更高版本的 Apple® Mac® □算机的商□□件。 它□ FreeBSD 系□提供了完整的支持。 在 Mac OS® X 上安装了□个□件之后， 用□需要配置虚□机并安装所需的客□操作系□。

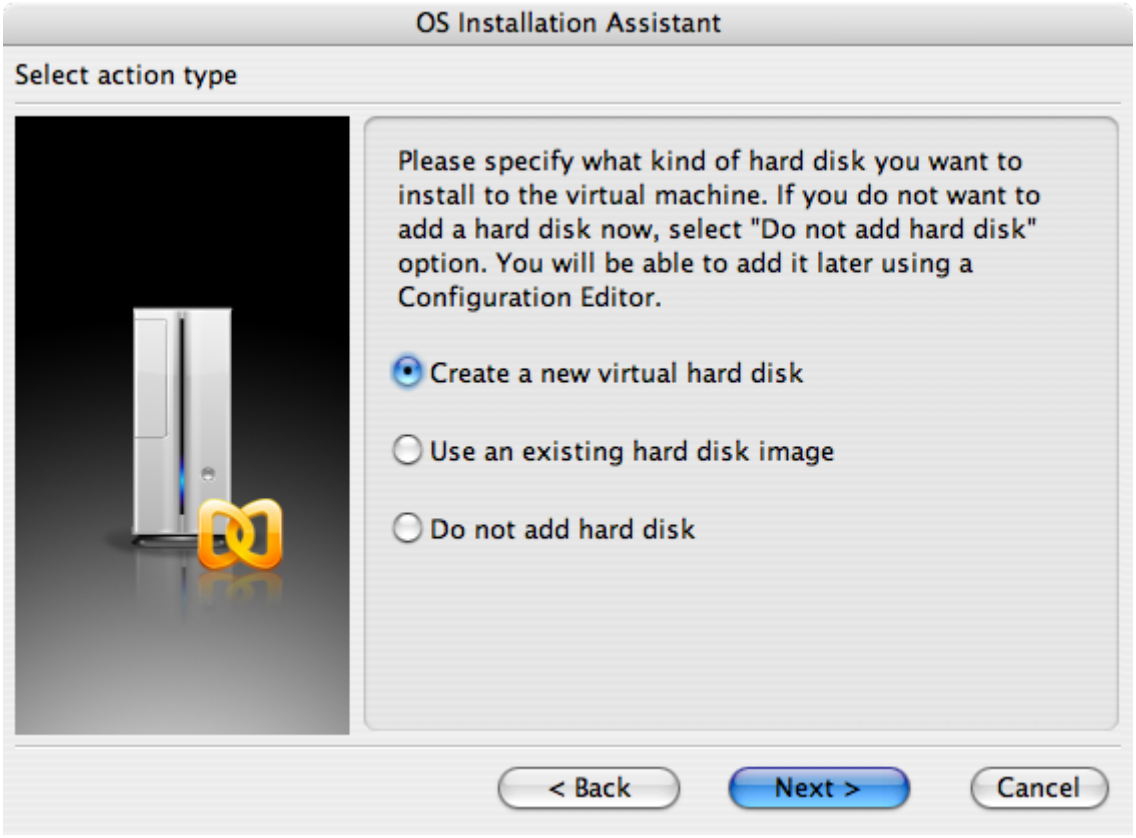
23.2.1.1. 在 Parallels/Mac OS® X 上安装 FreeBSD

在 Mac OS® X/Parallels 上安装 FreeBSD 的第一□是□建一个新的虚□机。 在系□提示□□客□ OS □型 (**Guest OS Type**) □□□ FreeBSD， 并根据□使用 FreeBSD 虚□□例的需要分配磁□和内存：

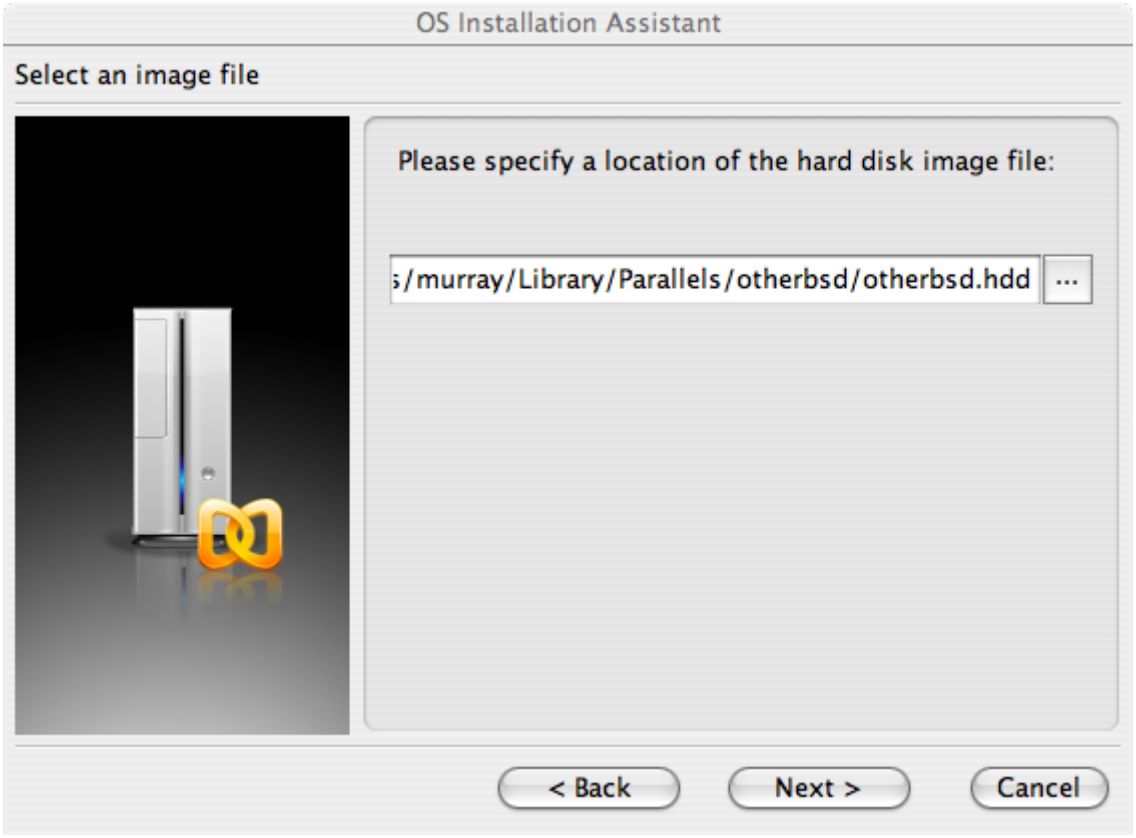


□多数在 Parallels 上使用 FreeBSD 的情形而言， 4GB 磁□空□和 512MB 的 RAM 就□用了：

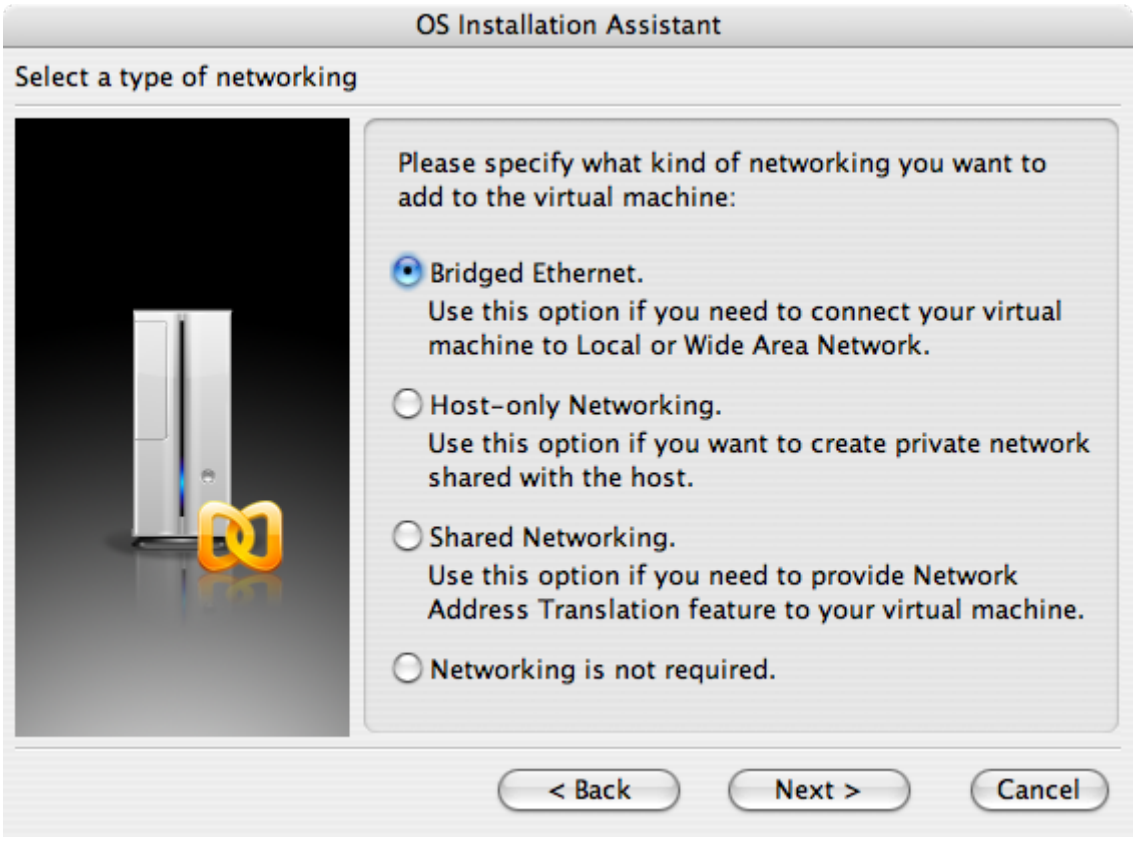


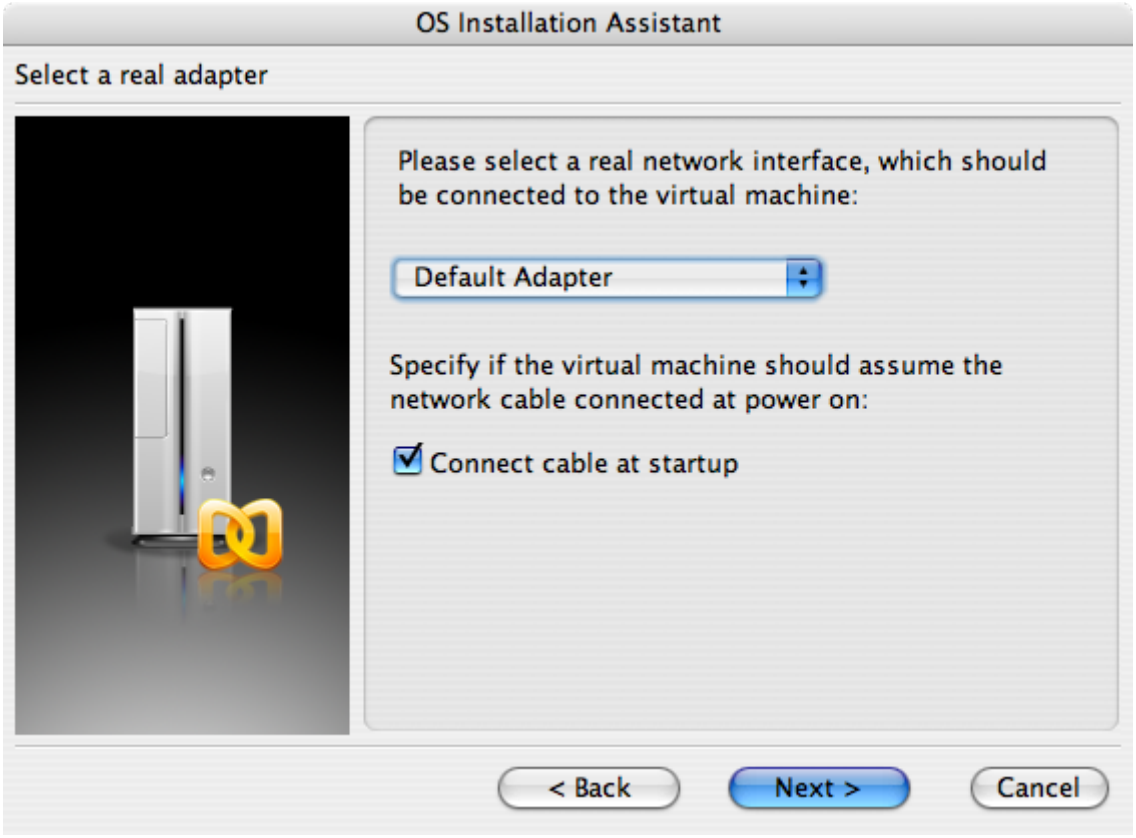

OS Installation Assistant
Select action type
Please specify what kind of hard disk you want to install to the virtual machine. If you do not want to add a hard disk now, select "Do not add hard disk" option. You will be able to add it later using a Configuration Editor.
Create a new virtual hard disk
Use an existing hard disk image
Do not add hard disk
< Back
Next >
Cancel


OS Installation Assistant
Specify hard disk options
Please specify a preferred virtual disk size:
8000
MB
Select disk format:
Expanding (recommended).
Disk image file is small initially and grows as you add more data to the virtual machine. This disk format takes less time to create and saves disk space on the host.
Plain.
Disk image file consumes all the allocated space right from the start. It takes more time to create but allows guest OS to operate faster.
< Back
Next >
Cancel

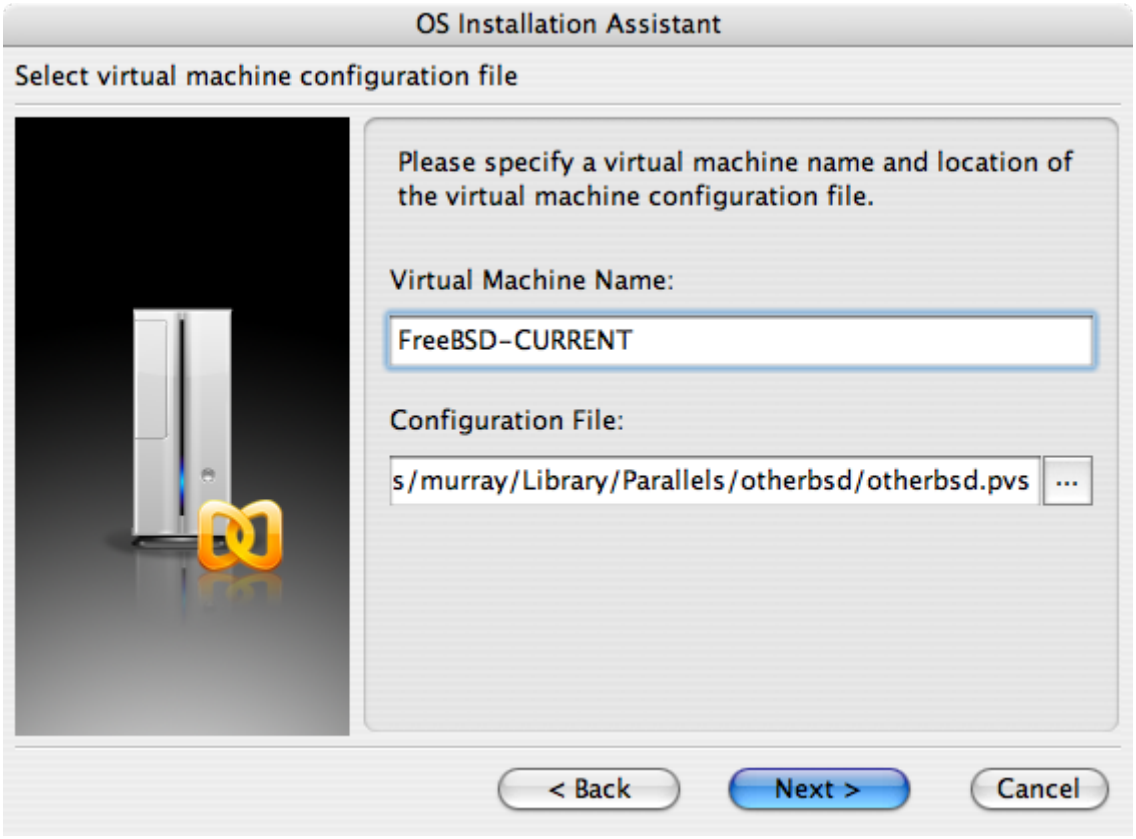

OS Installation Assistant
Select an image file
Please specify a location of the hard disk image file:
s/murray/Library/Parallels/otherbsd/otherbsd.hdd
...
< Back
Next >
Cancel

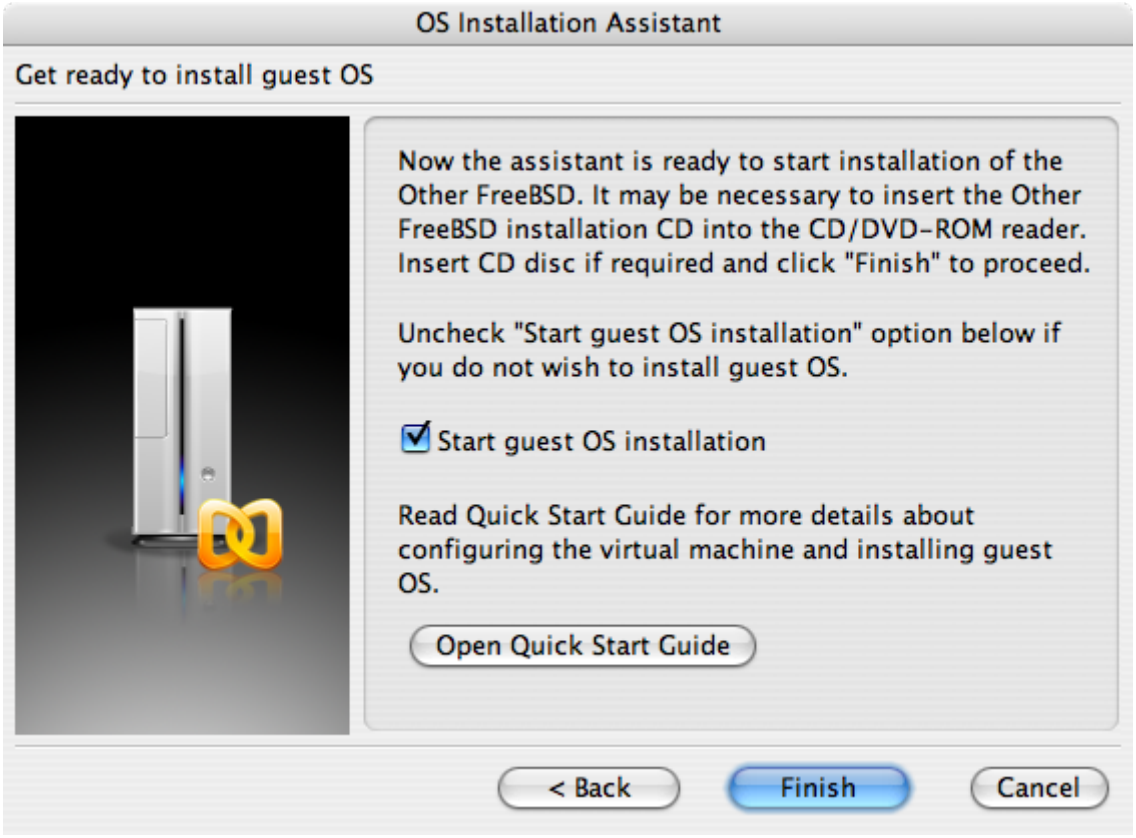
□□使用的网□和网□□型：


OS Installation Assistant
Select a type of networking
Please specify what kind of networking you want to add to the virtual machine:
Bridged Ethernet.
Use this option if you need to connect your virtual machine to Local or Wide Area Network.
Host-only Networking.
Use this option if you want to create private network shared with the host.
Shared Networking.
Use this option if you need to provide Network Address Translation feature to your virtual machine.
Networking is not required.
< Back
Next >
Cancel

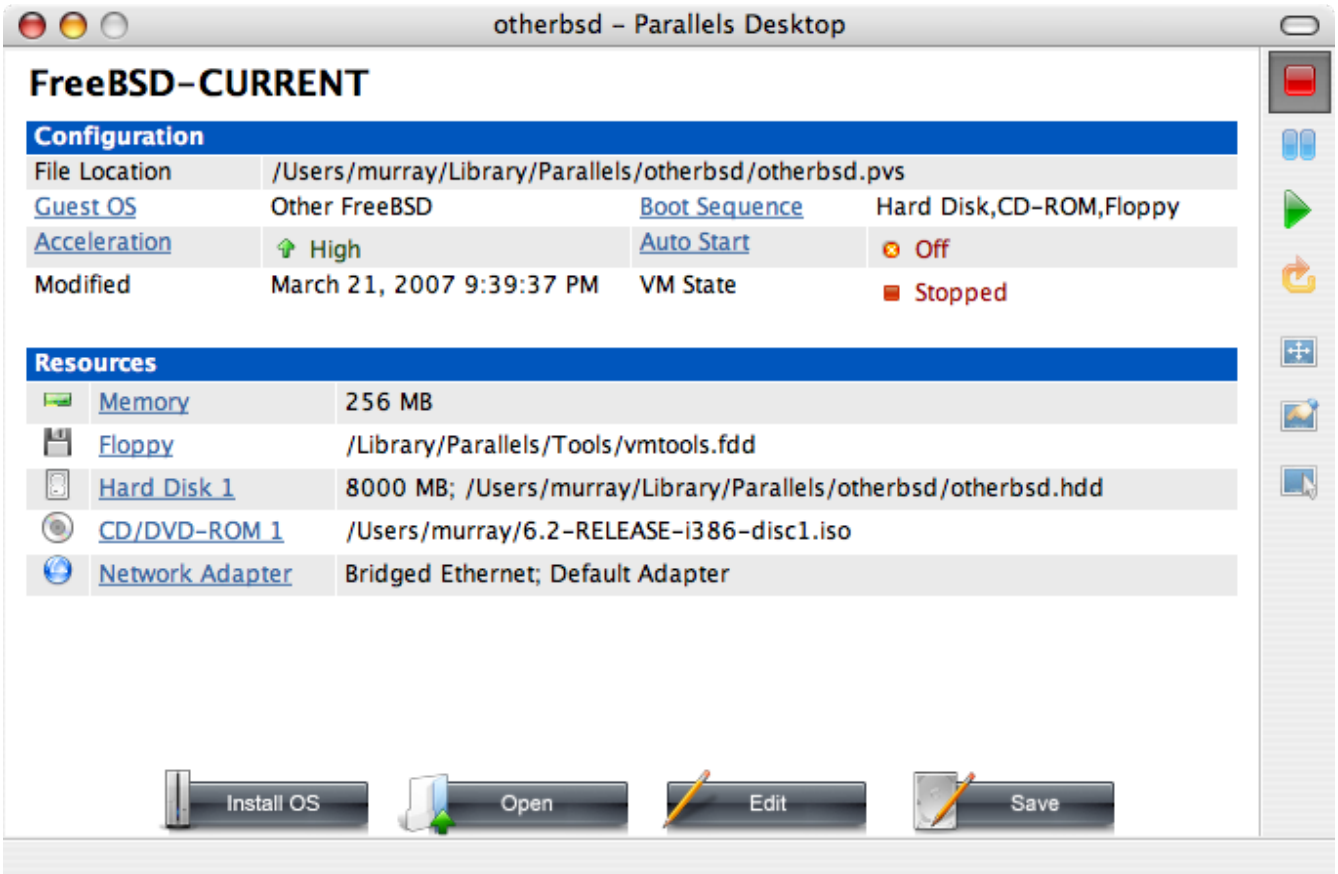

OS Installation Assistant
Select a real adapter
Please select a real network interface, which should be connected to the virtual machine:
Default Adapter
Specify if the virtual machine should assume the network cable connected at power on:
Connect cable at startup
< Back
Next >
Cancel

保存并完成配置：


OS Installation Assistant
Select virtual machine configuration file
Please specify a virtual machine name and location of the virtual machine configuration file.
Virtual Machine Name:
FreeBSD-CURRENT
Configuration File:
s/murray/Library/Parallels/otherbsd/otherbsd.pvs
...
< Back
Next >
Cancel

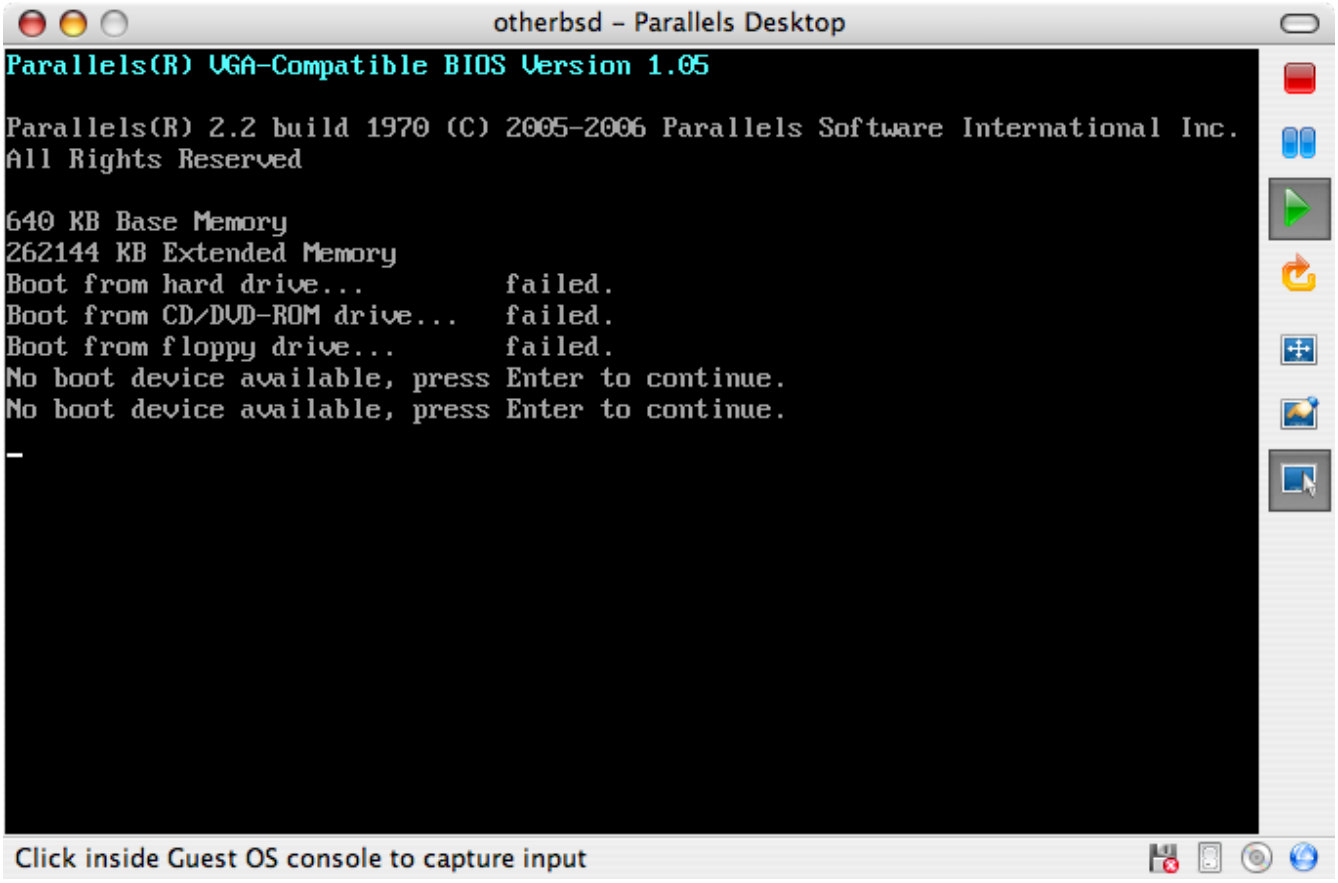


在□建了 FreeBSD 虚□机之后， □需要在其中安装 FreeBSD。 最好的做法是使用官方的 FreeBSD CDROM 或从官方 FTP 站点下□的 ISO □像来完成□个任□。 如果□的本地 Mac® 文件系□中有 ISO 映像文件， 或□的 Mac® 的 CD □□器中有 CDROM， 就可以在 FreeBSD Parallels □口的右下角点□光□□□。 之后， 系□将□出一个□口， 供□完成将虚□机中的 CDROM □□器□接到本地的 ISO 文件或真正的 CDROM □□器上。

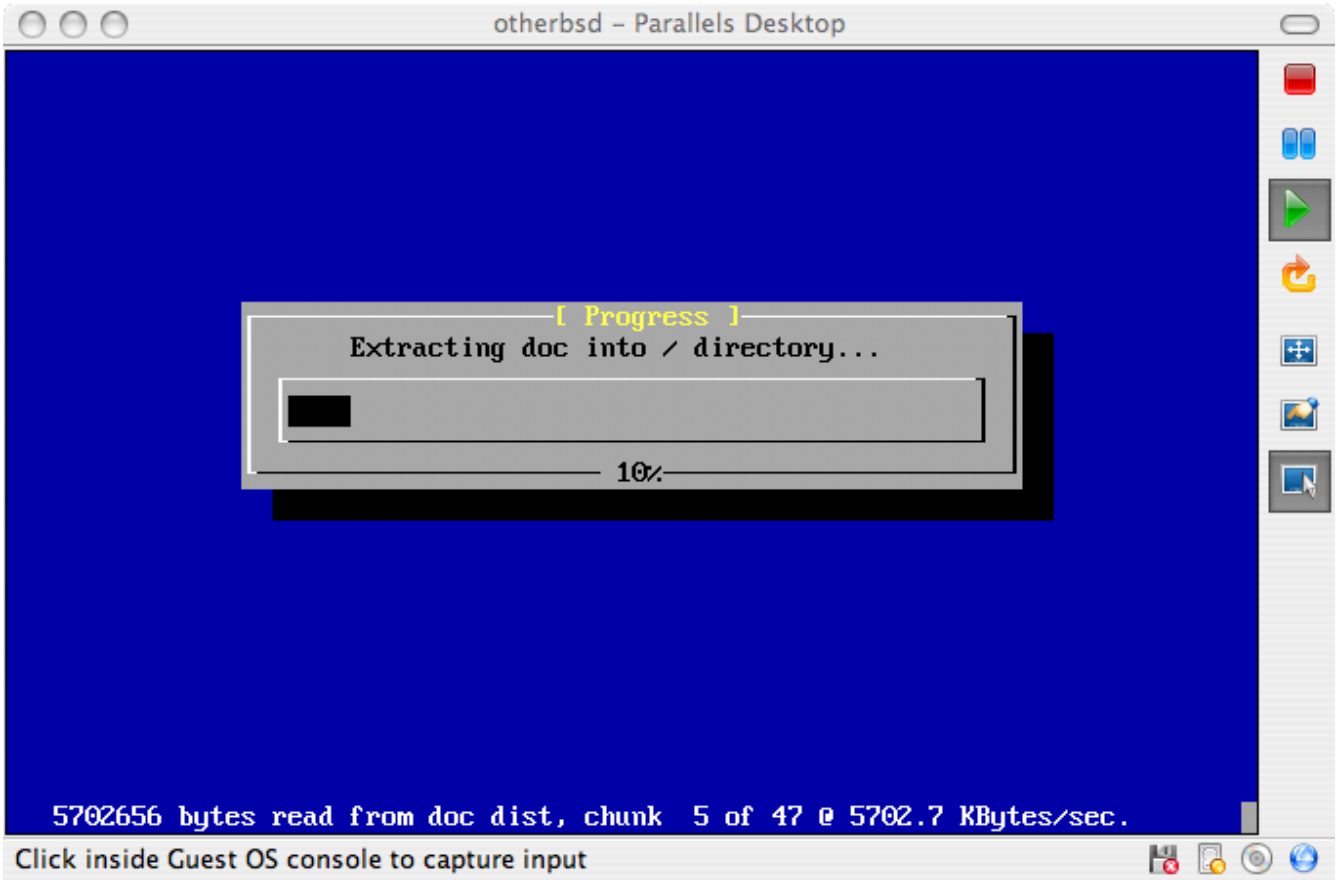


在完成了将 CDROM 与□的安装源完成□□之后， 就可以按重□ (reboot) □□来重□ FreeBSD 虚□机了。

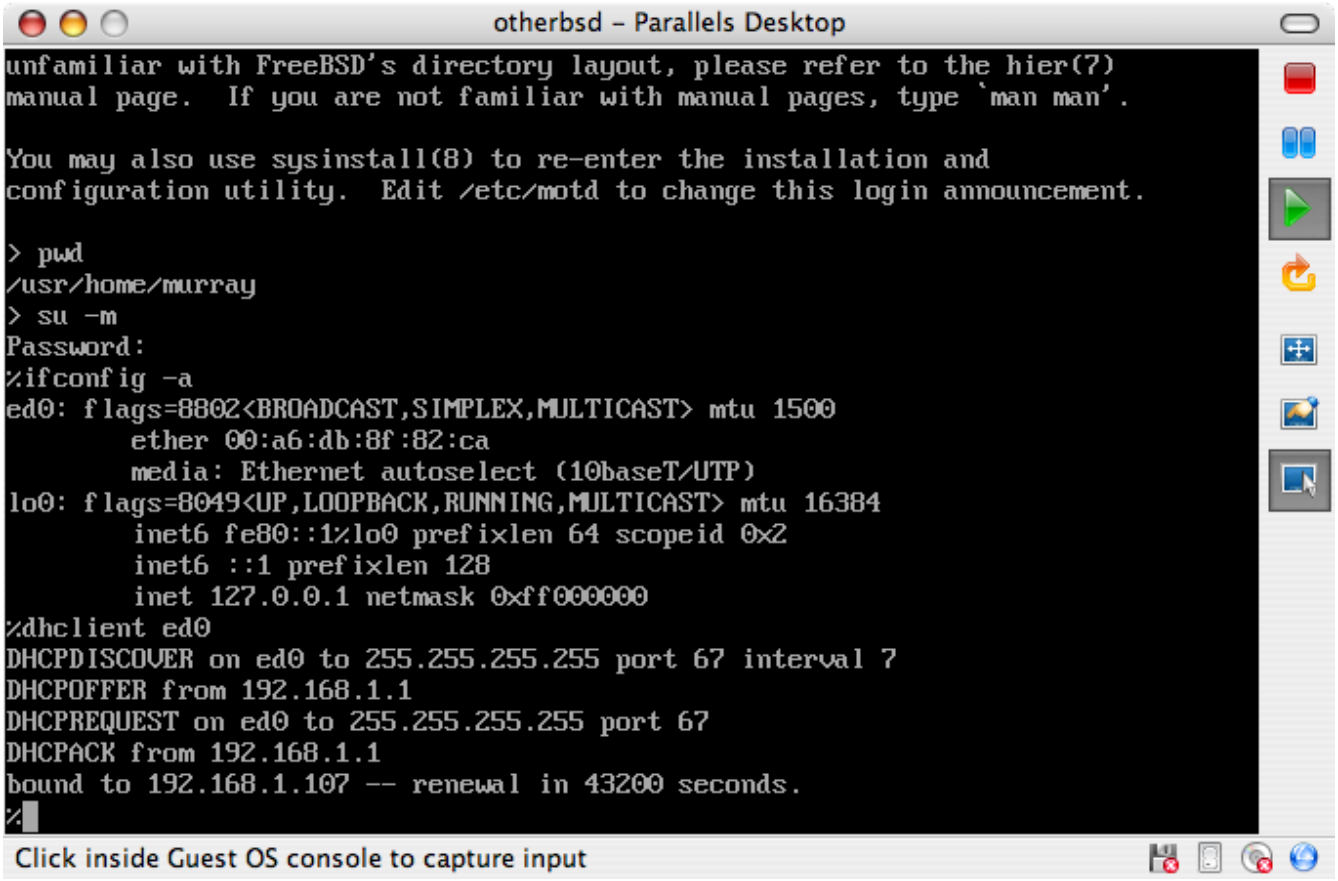
Parallels 将配合一个特殊的 BIOS □□， 后者能□像普通的 BIOS 一□□□系□中是否有 CDROM □□器。



此□， 它就能□□到 FreeBSD 安装介□并□始 安装 FreeBSD 中所介□的□准的基于 sysinstall 安装的□程。



此□□可以安装 X11， 但□□不要□它□行配置。 在完成安装之后， 重□并□入新安装的 FreeBSD 虚□机。



23.2.1.2. 在 Mac OS® X/Parallels 上配置 FreeBSD

在□将 FreeBSD 安装到 Mac OS® X 的 Parallels 上之后， □需要□行一系列的配置， 以便□系□的虚□化操作□行□化。

1. 配置引□加□器□量

   最重要的一□是通□□低 `kern.hz` □量来降低 Parallels □境中的 FreeBSD □ CPU 的使用。 □可以通□在 /boot/loader.conf 中□加下述配置来完成：

   ```
   kern.hz=100
   ```

   如果不使用□个配置， □置的 FreeBSD Parallels 客□ OS 会在□□理器的 iMac® 上使用大□ 15% 的 CPU。 如此修改之后， 空□□的使用量就□少到大□ 5% 了。

2. □建新的内核配置文件

   □可以□去全部 SCSI、 FireWire， 以及 USB □□□□程序。 Parallels 提供了一个由 ed(4) □□的虚□网□， 因此， 除了 ed(4) 和 miibus(4) 之外的其他网□接口□□都可以从内核中□去。

3. 配置网□

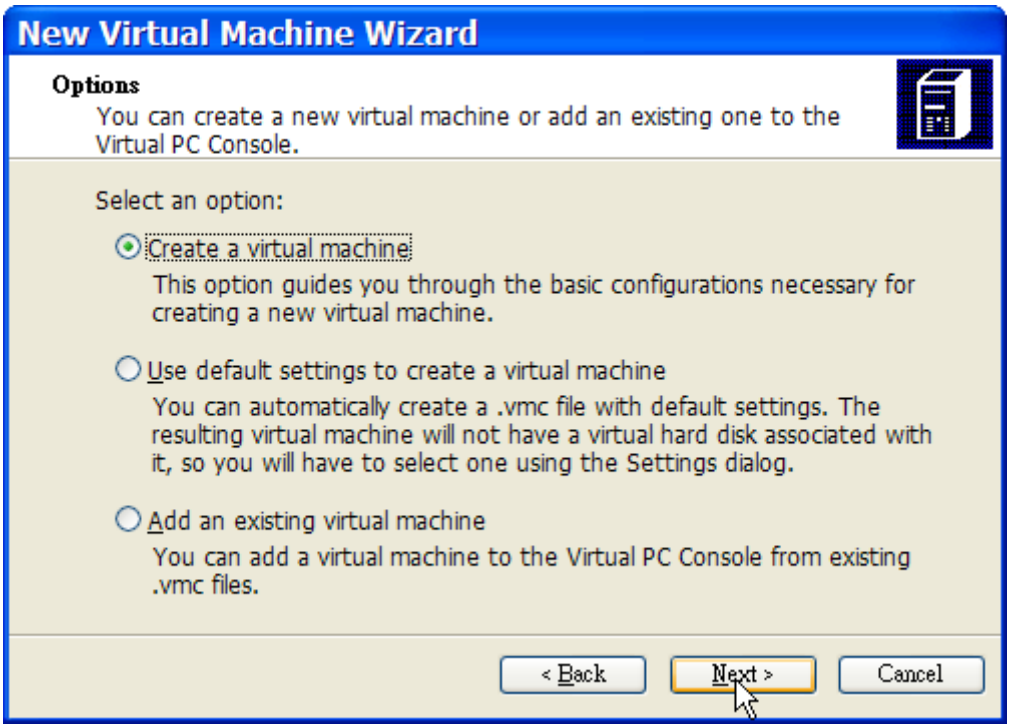
   最基本的网□配置， 是通□使用 DHCP 来将□的虚□机与宿主 Mac® 接入同一个局域网。 □可以通□在 /etc/rc.conf 中加入 `ifconfig_ed0="DHCP"` 来完成。 更高□一些的网□配置方法， □参□ 高□网□ 中的介□。

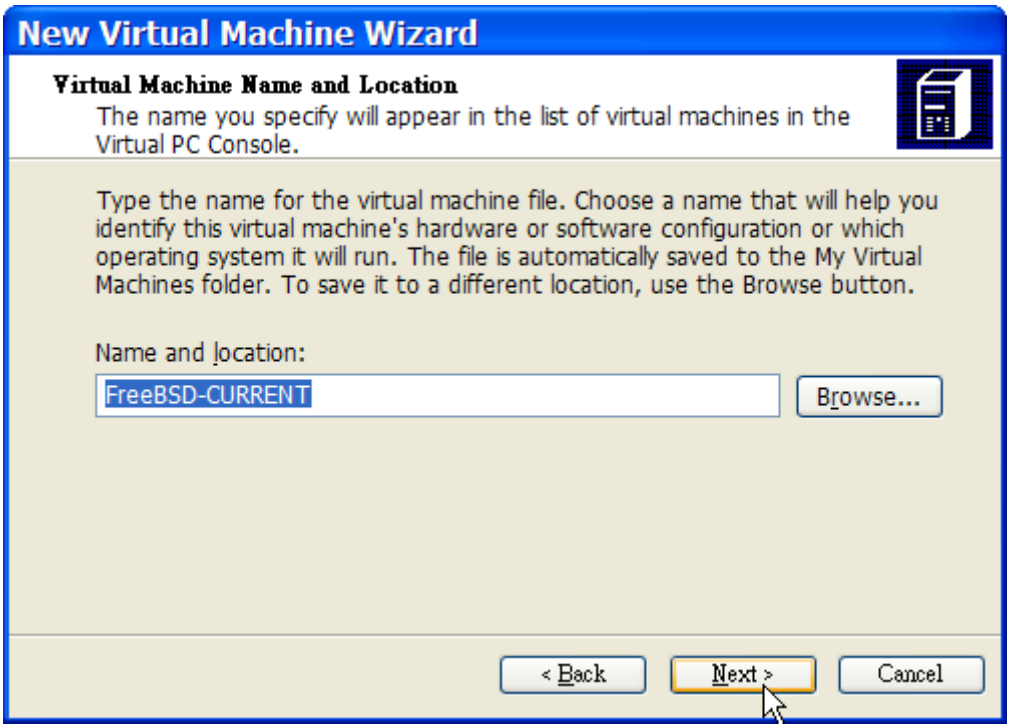
23.2.2. Windows® 上的 Virtual PC

Virtual PC 是 Microsoft® 上的 Windows® □件□品， 可以免□下□使用。 相□系□要求，□参□ system requirements □明。 在 Microsoft® Windows® 装完 Virtual PC 之后， 必□□□所安装的虚□机器来做相□□定。

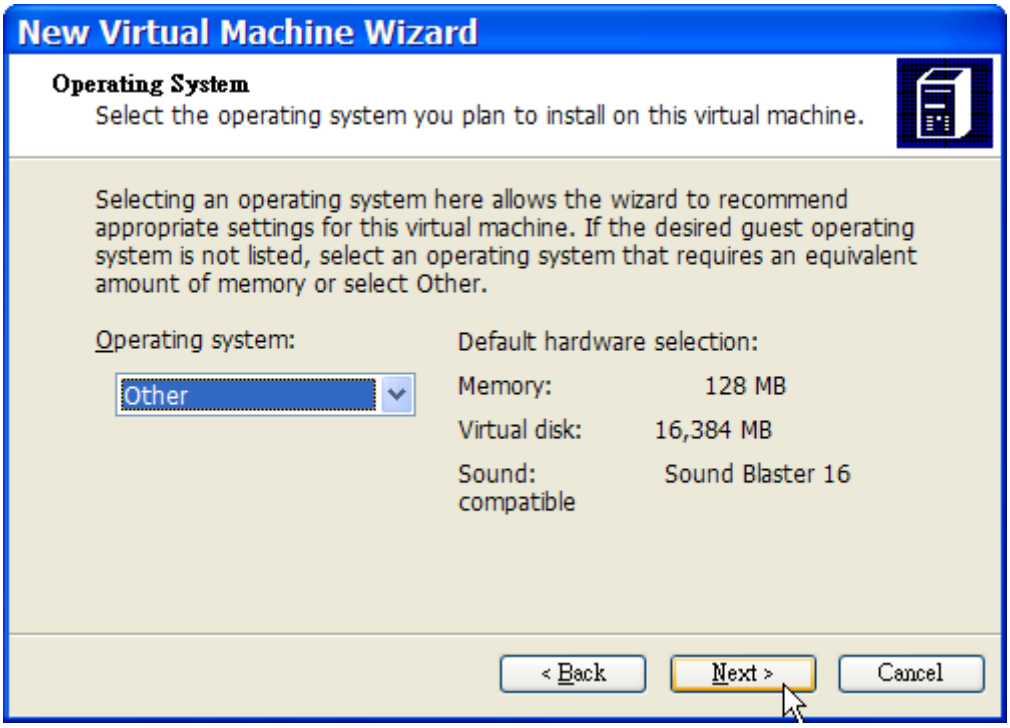
23.2.2.1. 在 Virtual PC/Microsoft® Windows® 上安装 FreeBSD

在 Microsoft® Windows®/Virtual PC 上安装 FreeBSD 的第一□是新□虚□器。 如下所示，在提示向□中□□□ Create a virtual machine：

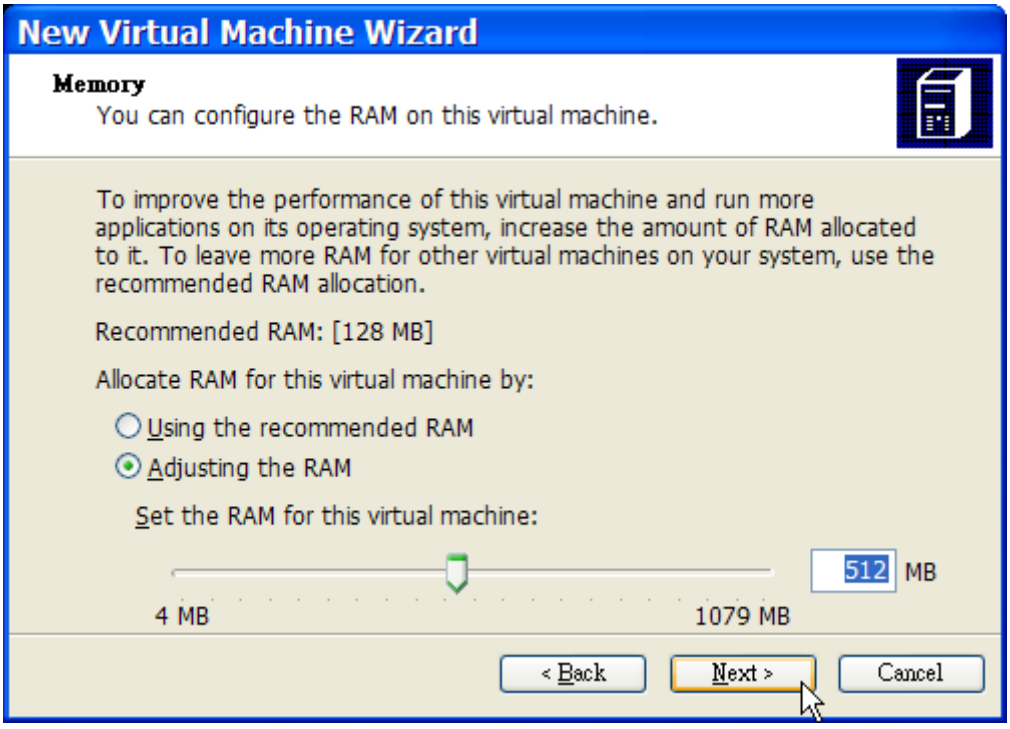


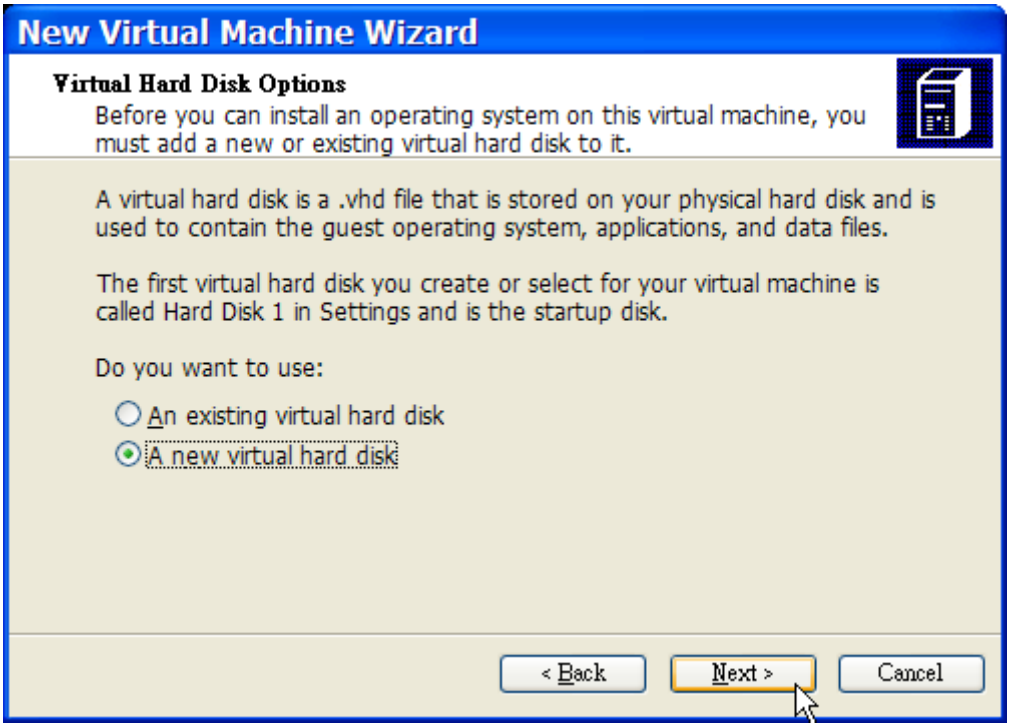


然后在 Operating system □□ Other：

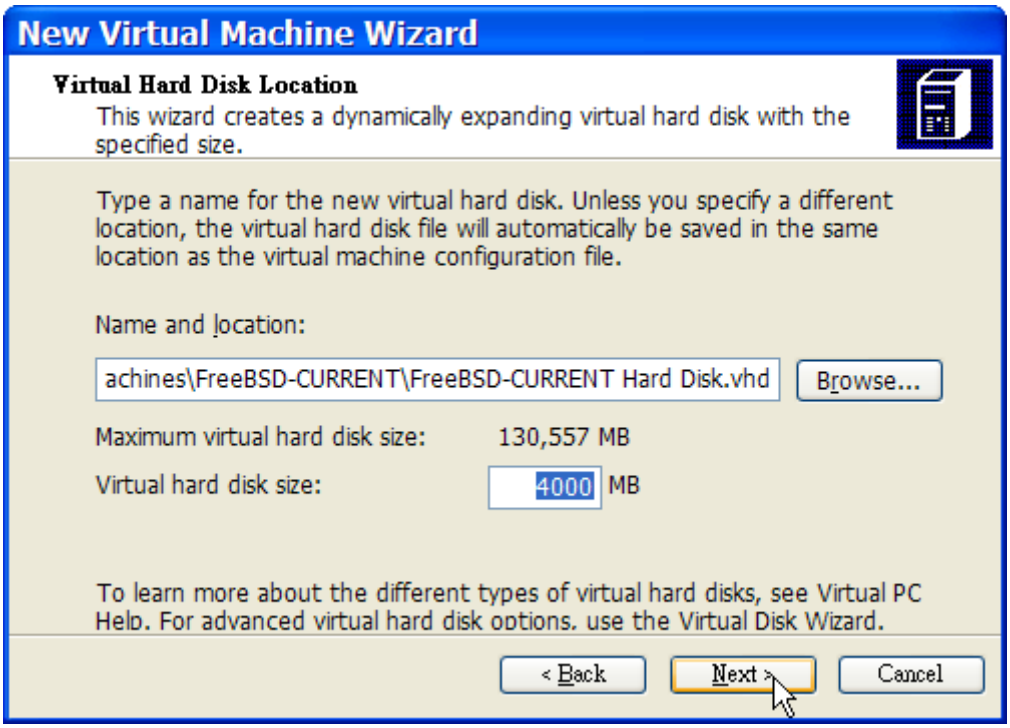


并依据自身需求来□□硬□容量和内存的分配。□大多数在 Virtual PC 使用 FreeBSD 的情况而言， 大□ 4GB 的硬□空□以及 512MB 的内存就□用了。

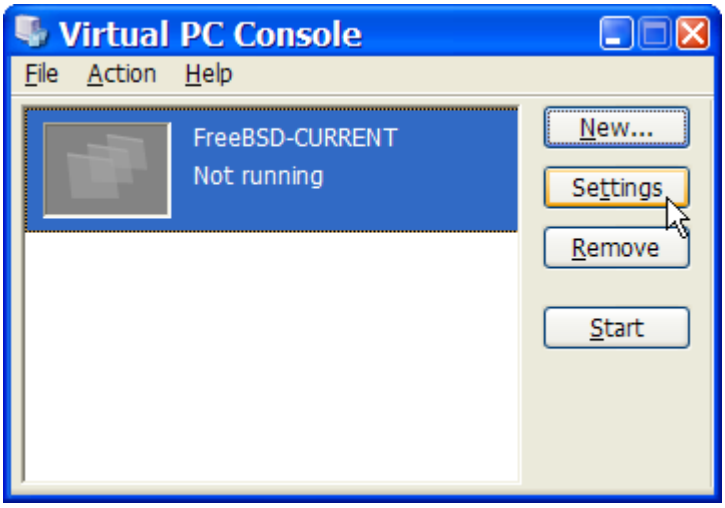


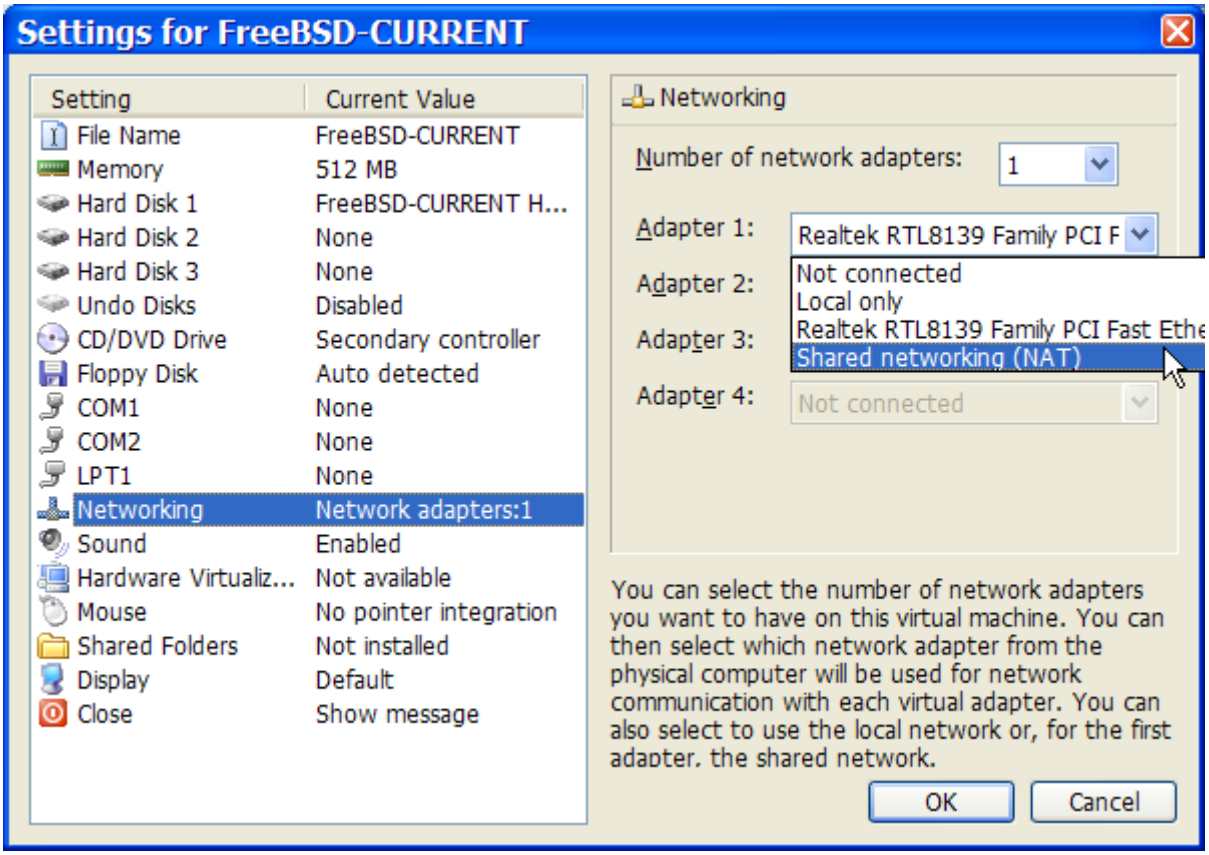


保存并完成配置：

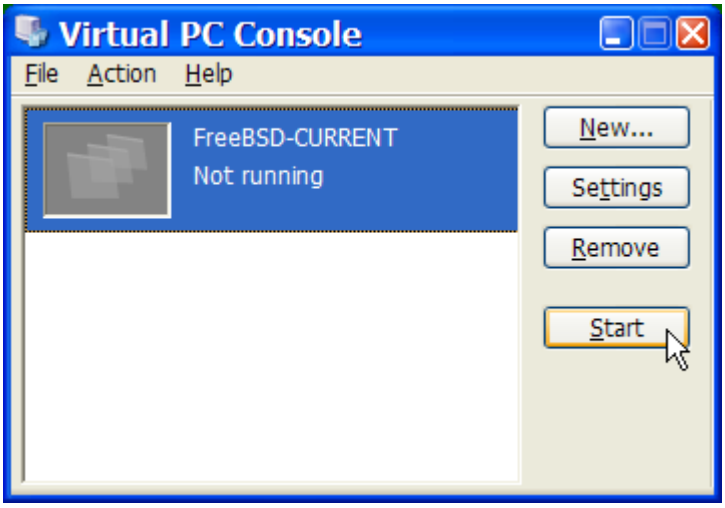


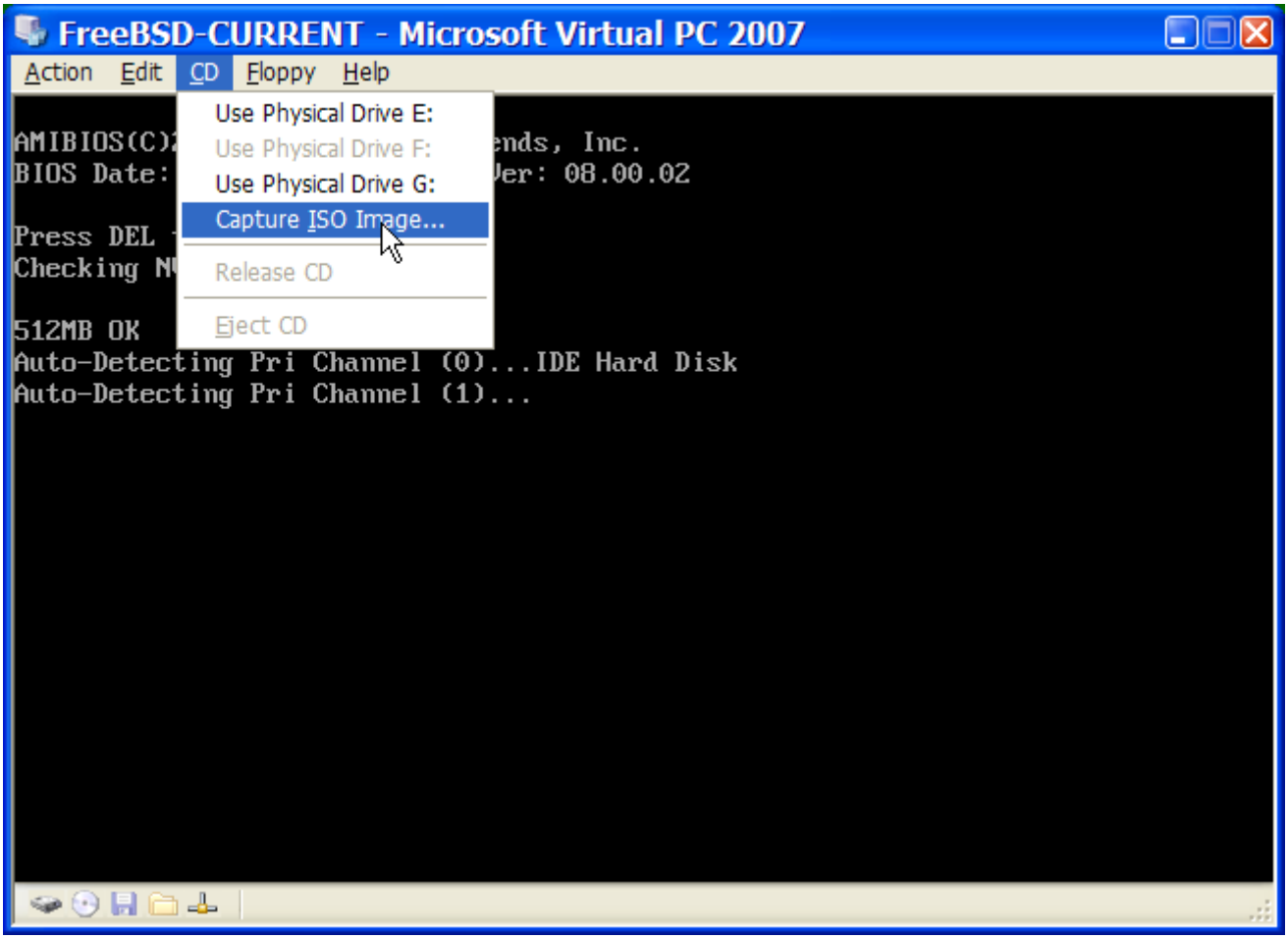
接下来□□新建的 FreeBSD 虚□机器，并□□ **Settings**， 以□定网□□□以及网□：



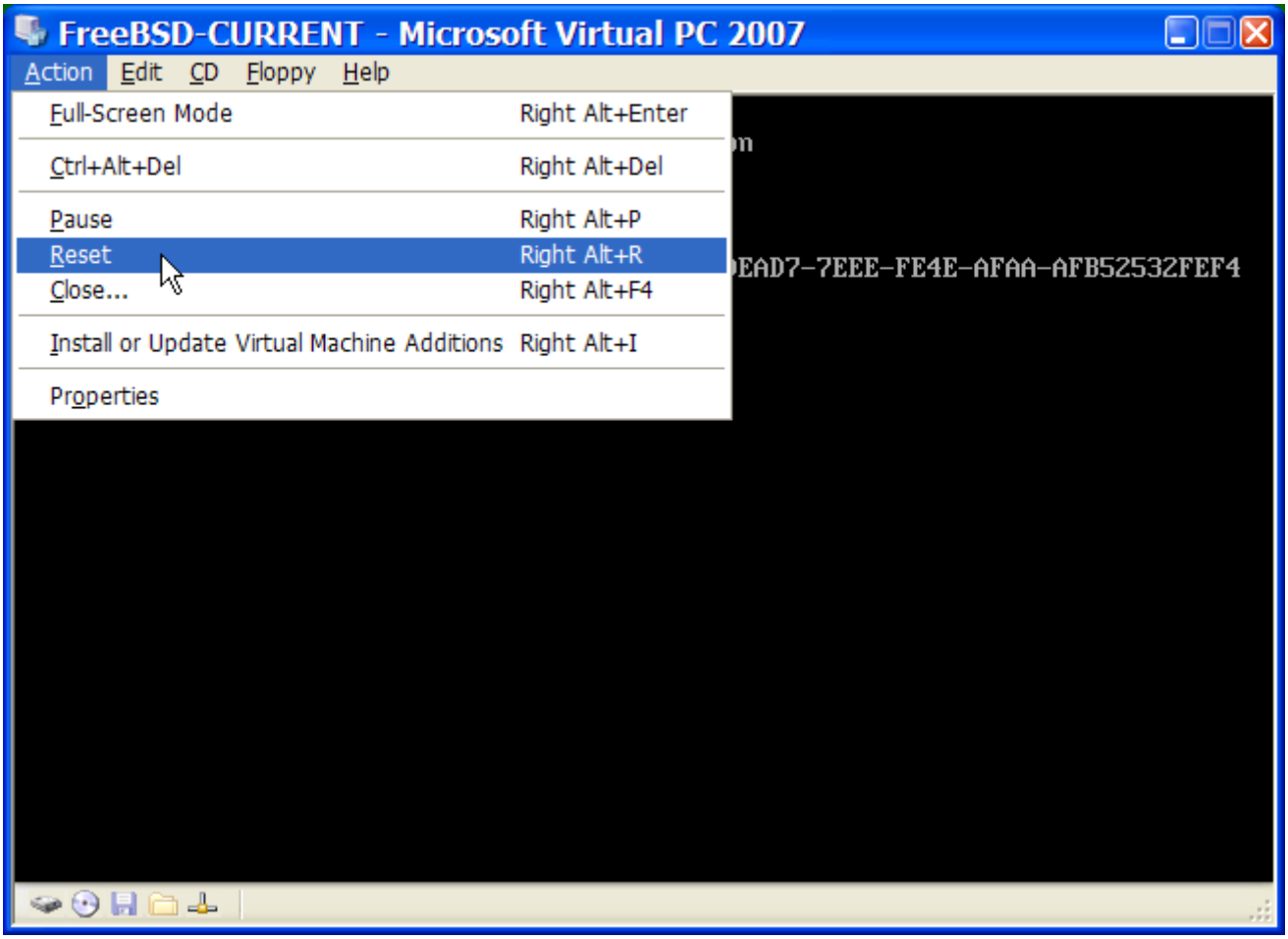


在新建 FreeBSD 虚□机器以后， 就可以□□以其安装 FreeBSD。 安装方面， 比□好的作法是使用官方的 FreeBSD 光□或从官方 FTP 站下□ ISO □像。 若□的 Windows® 系□ 内已有□ ISO □像， 那□就可以在 FreeBSD 虚□机器上双□， 以□始□□。 接着在 Virtual PC □口内按 **CD** 再按 **Capture ISO Image...**。 接着出□一个□□框， 可以把虚□机器内的光□□定到□ ISO □像， 或者是真□的光□。

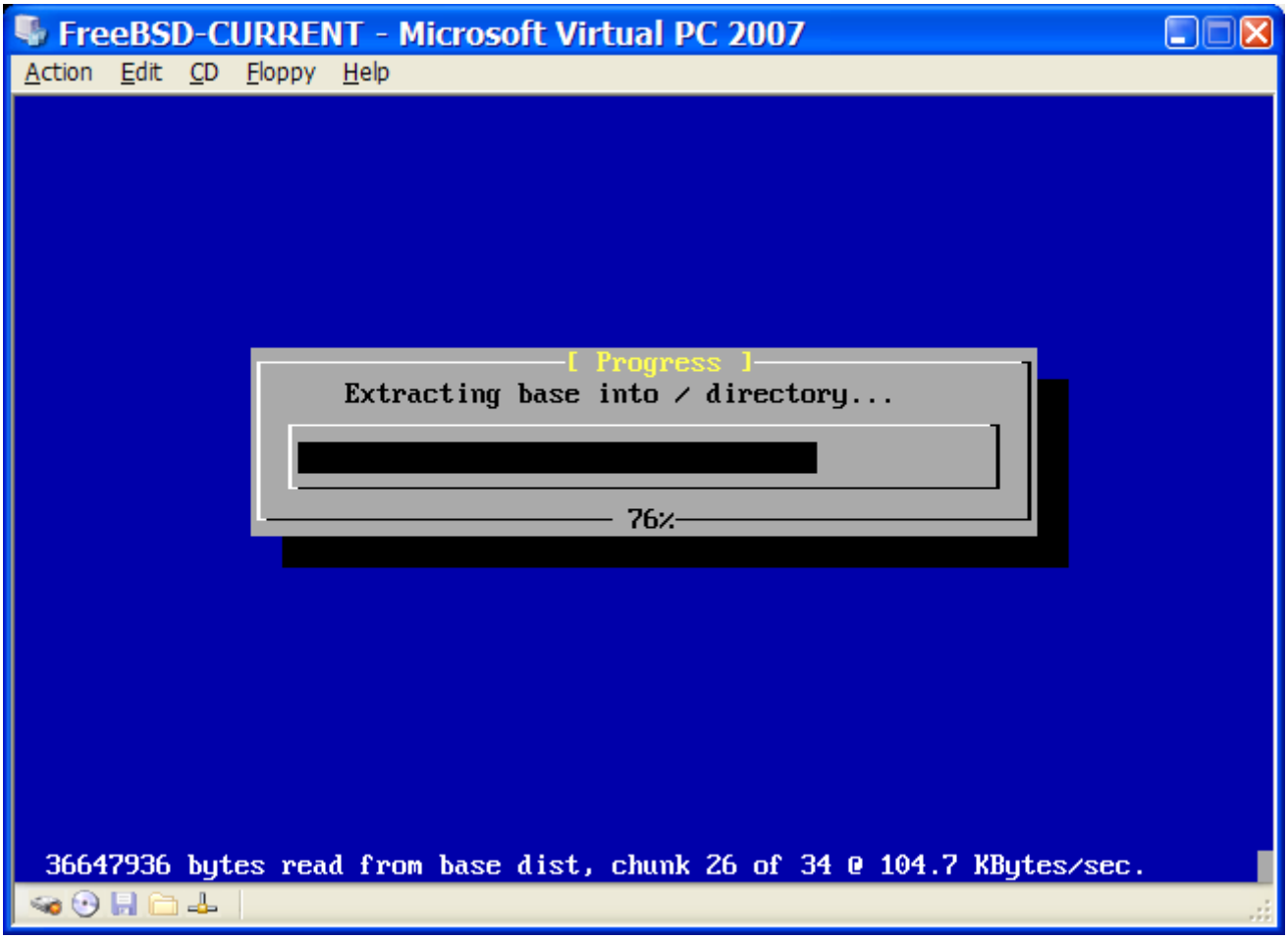




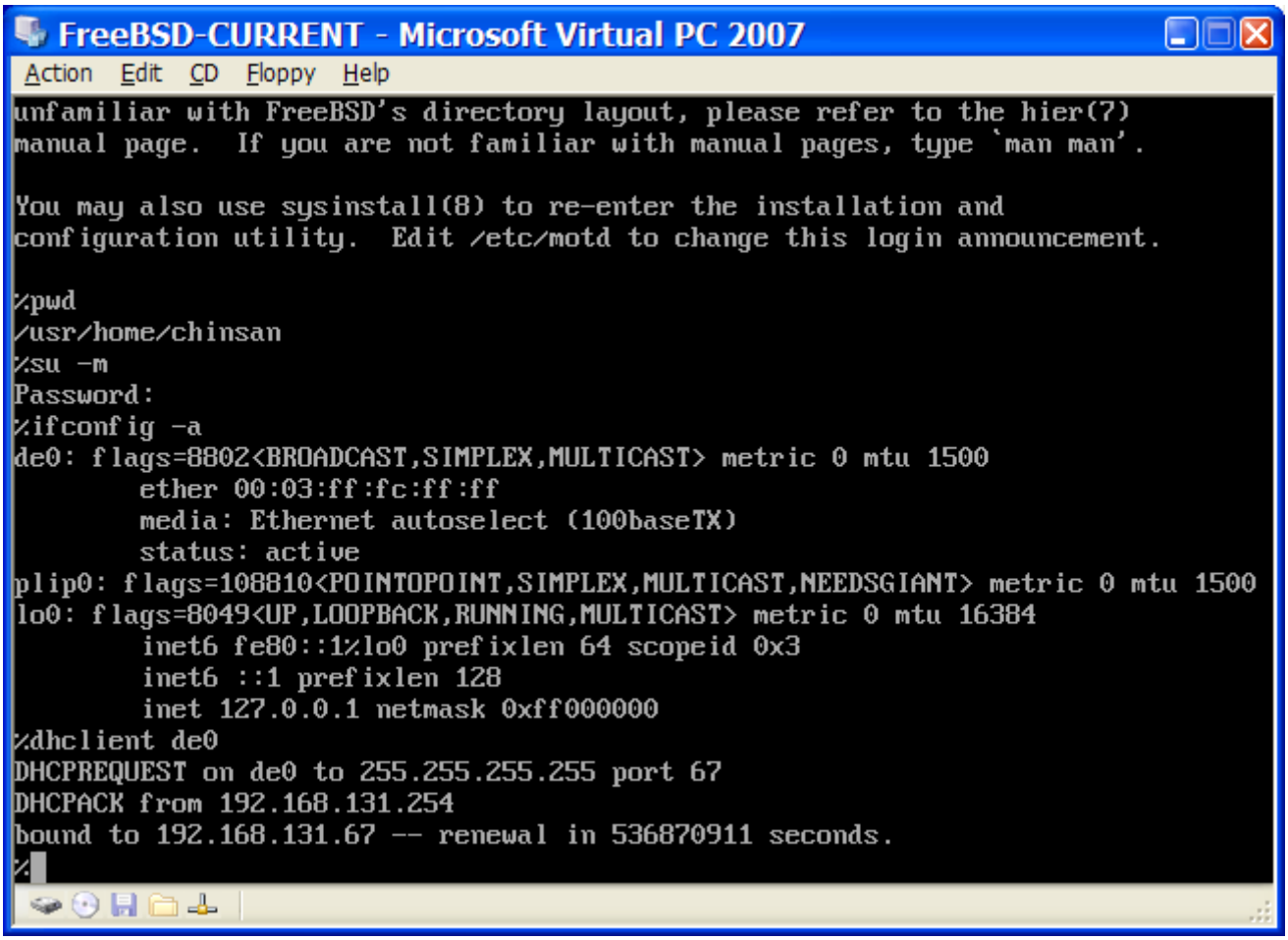
□好光□来源之后，就可以重新□机， 也就是先按 **Action** 再按 **Reset** 即可。 Virtual PC 会以特殊 BIOS □机，并与普通 BIOS 一□会先□□是否有光□□□器。



此□， 它会□到 FreeBSD 安装光□， 并□始在 安装 FreeBSD 内所介□的 sysinstall 安装□程。 □□候也可以□便安装 X11， 但不要□行相□□定。



完成安装之后， 得把安装光或者 ISO 像退出。 最后， 把装好的 FreeBSD 虚机器重新机即可。



23.2.2.2. □整 Microsoft® Windows®/Virtual PC 上的 FreeBSD

在 Microsoft® Windows® 上以 Virtual PC 装好 FreeBSD 后， □需要做一些□定□□， 以便将虚□机内的 FreeBSD 最佳化。

1. □定 boot loader 参数

   最重要的□□乃是藉由□降 `kern.hz` 来降低 Virtual PC □境内 FreeBSD 的 CPU 占用率。 在 /boot/loader.conf 内加上下列□定即可：

   ```
   kern.hz=100
   ```

   若不作□□定， 那□光是 idle 状□的 FreeBSD Virtual PC guest OS 就会在□一□理器的□□上大□有 40% 的 CPU 占用率。 作了上述修改之后, 占用率大□会降至 3%。

2. 建立一个新的内核配置文件

   可以放心把所有的 SCSI， FireWire 和 USB □□□□都移除。 Virtual PC 有提供 de(4) 的虚□网□, 因此除了 de(4) 以及 miibus(4) 以外其他的网□也都可以从内核的配置文件中移除。

3. □定网□

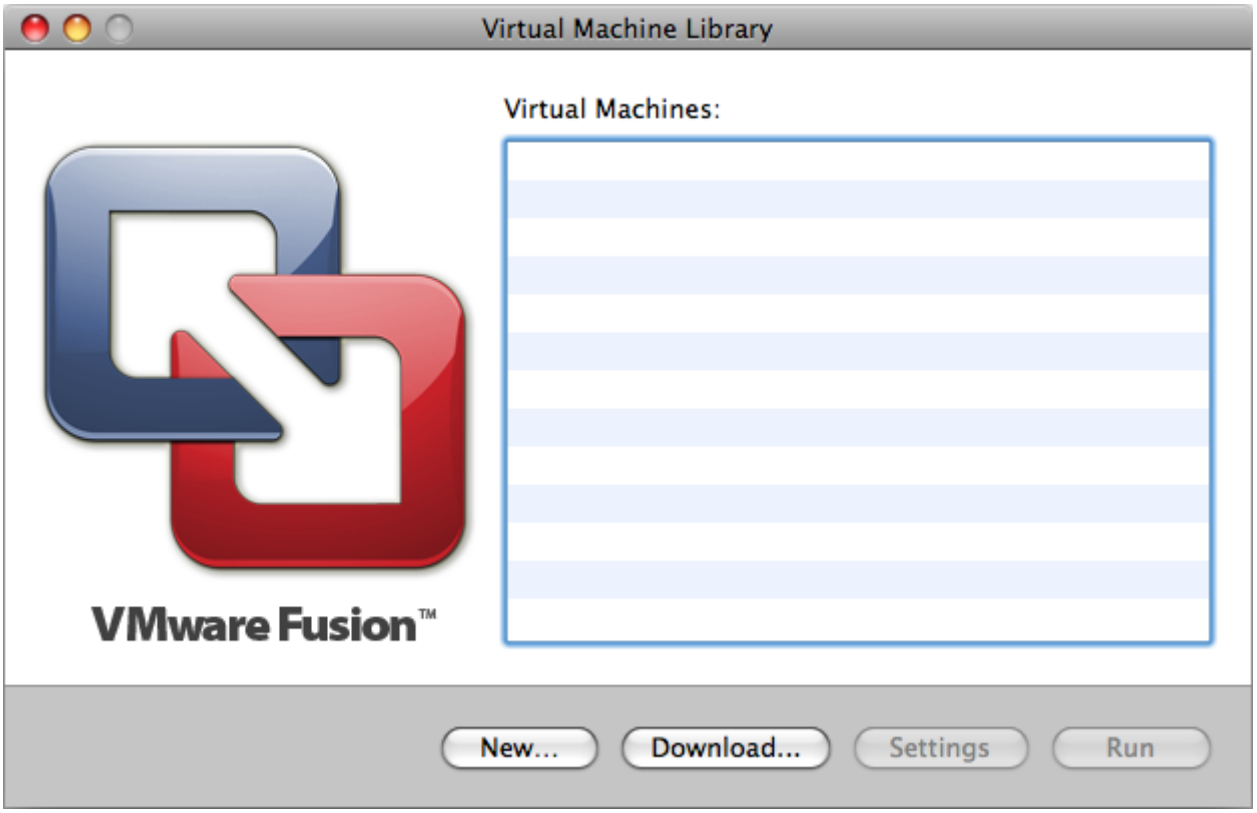
   可以□虚□机器□□得使用 DHCP 来□定与 host (Microsoft® Windows®) 相同的本地网□□境， 只要在 /etc/rc.conf 加上 `ifconfig_de0="DHCP"` 即可完成。 其他的高□网□□置， 可参□ 高□网□.

23.2.3. □行于 MacOS 的 VMware

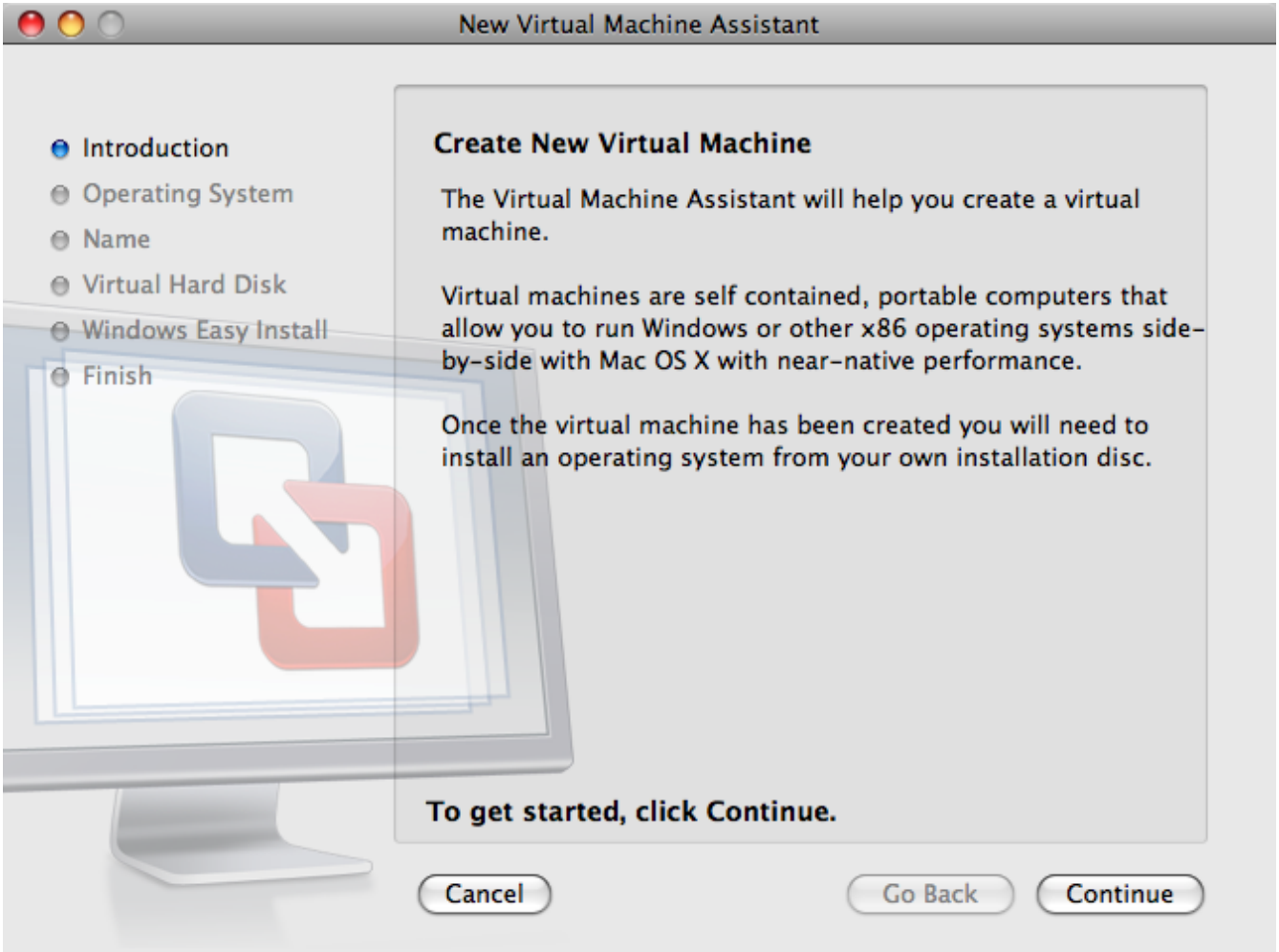
Mac® 版本的 VMware Fusion 是一个商□□件，□行在基于 Intel® 的 Apple® Mac® □算机的 Mac OS® 10.4.9 或更版本的操作系□上。 FreeBSD 是一个完全被支持的客□操作系□。 在 Mac OS® X 上安装了 VMware Fusion 之后， 用□就可以着手配置一个虚□机器并安装客□操作系□。

23.2.3.1. 在 VMware/Mac OS® X 上安装 FreeBSD

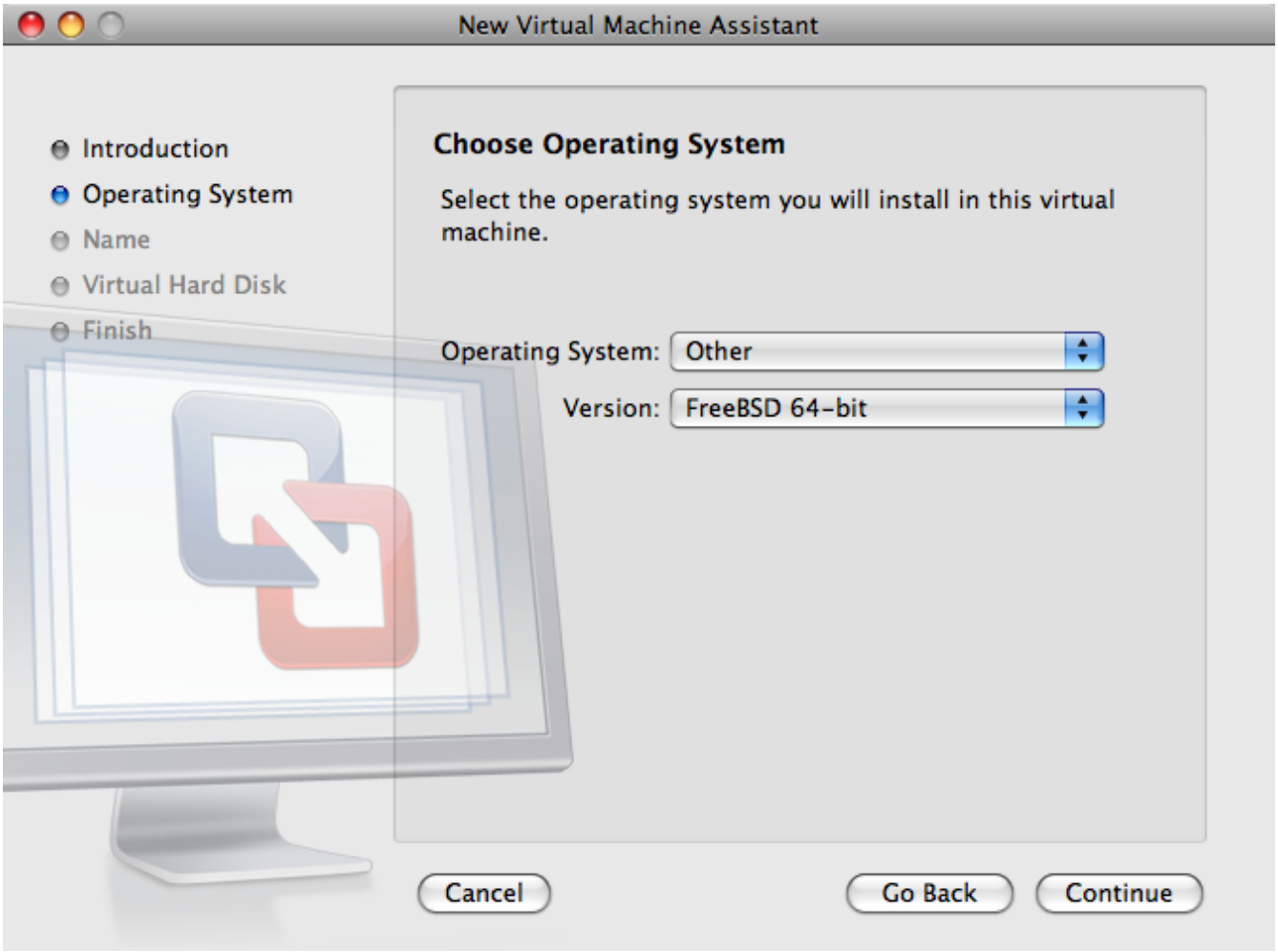
第一□是□行 VMware Fusion， 虚□机器□将被装□。 □□ "New" □建 VM：



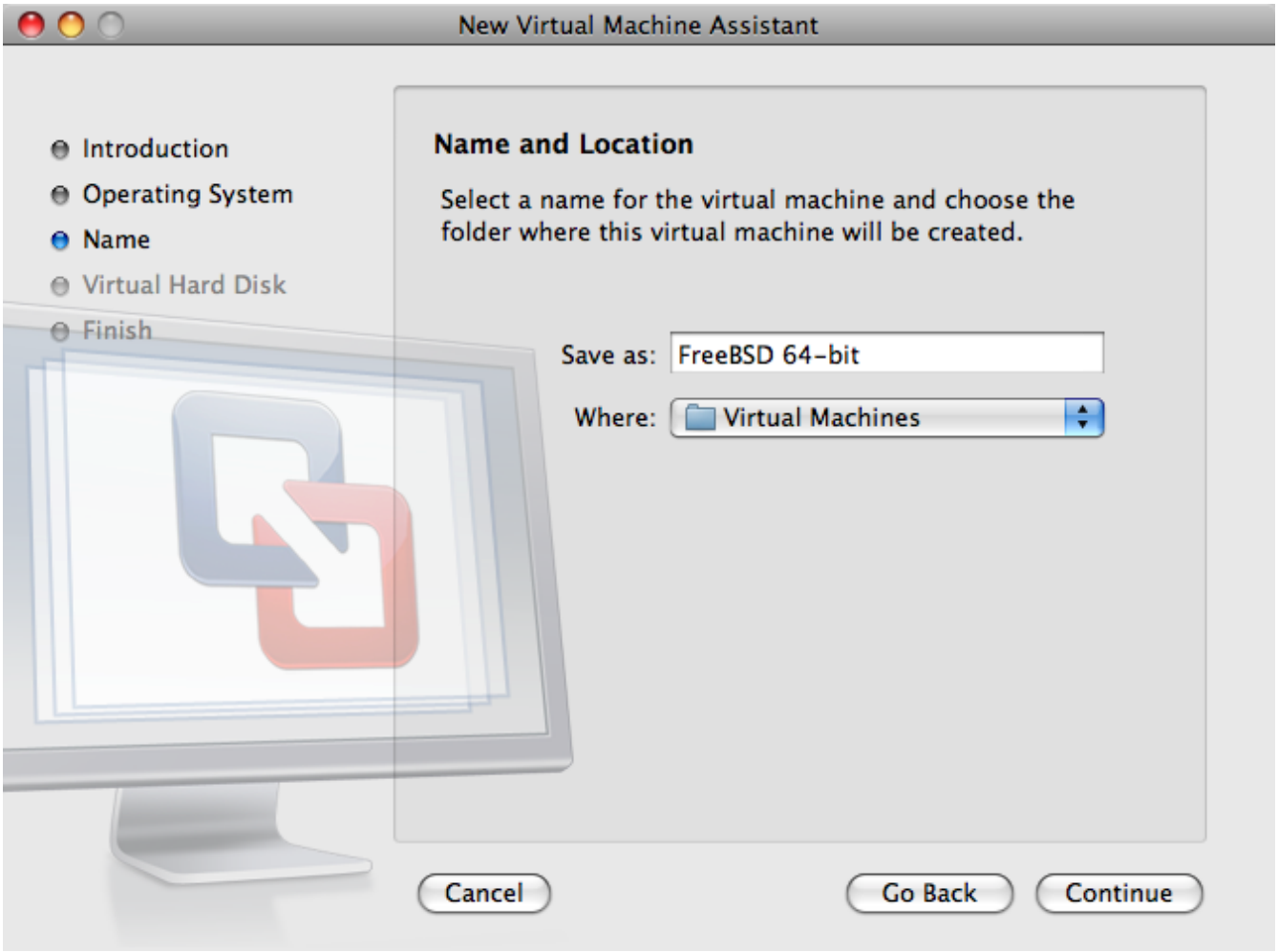
New Virtual Machine Assistant 将被□行来□助□□建 VM， □□ Continue □□：



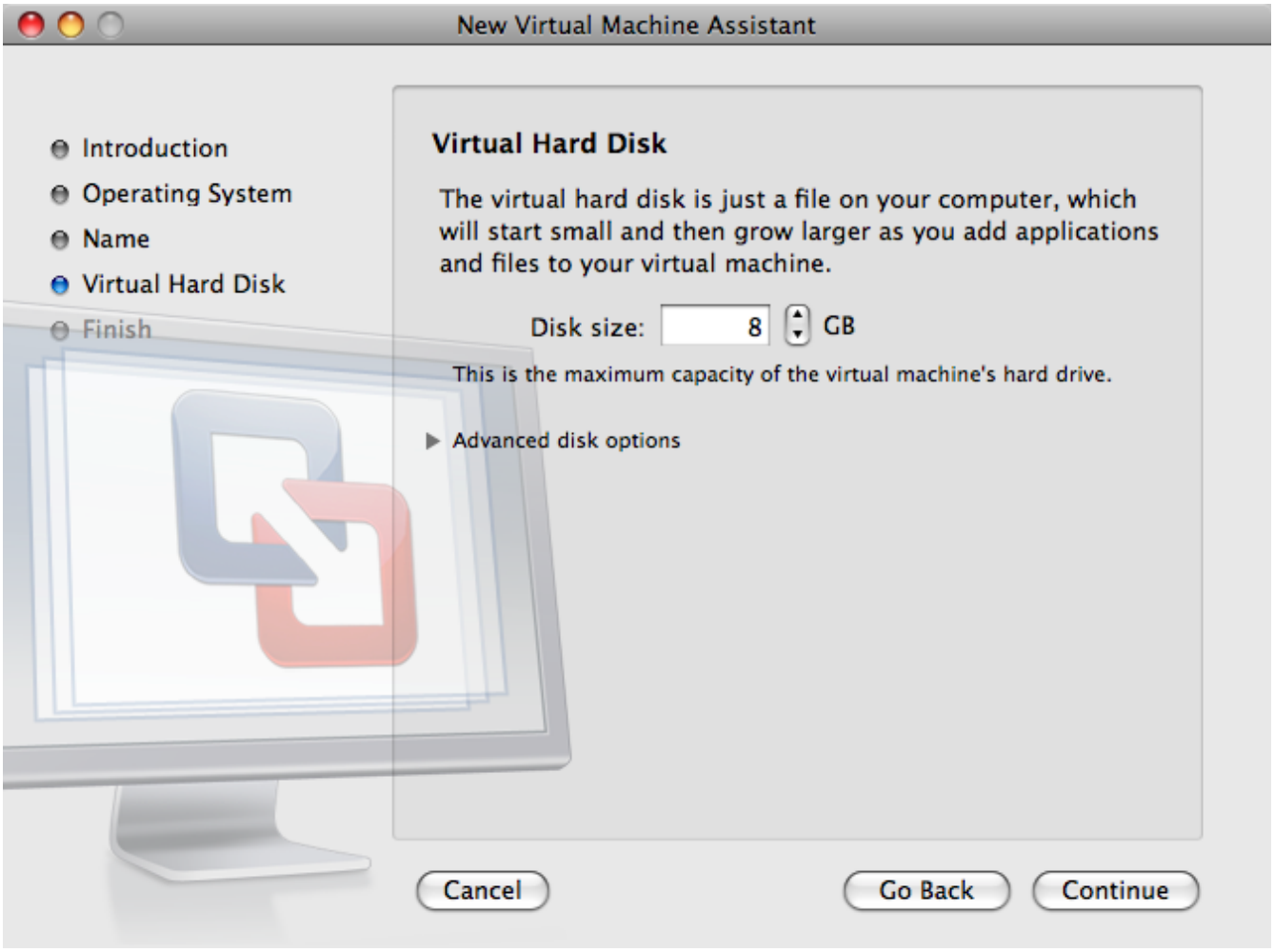
在 Operatiing System □□□ Other，Version □可□ FreeBSD 或 FreeBSD 64-bit。



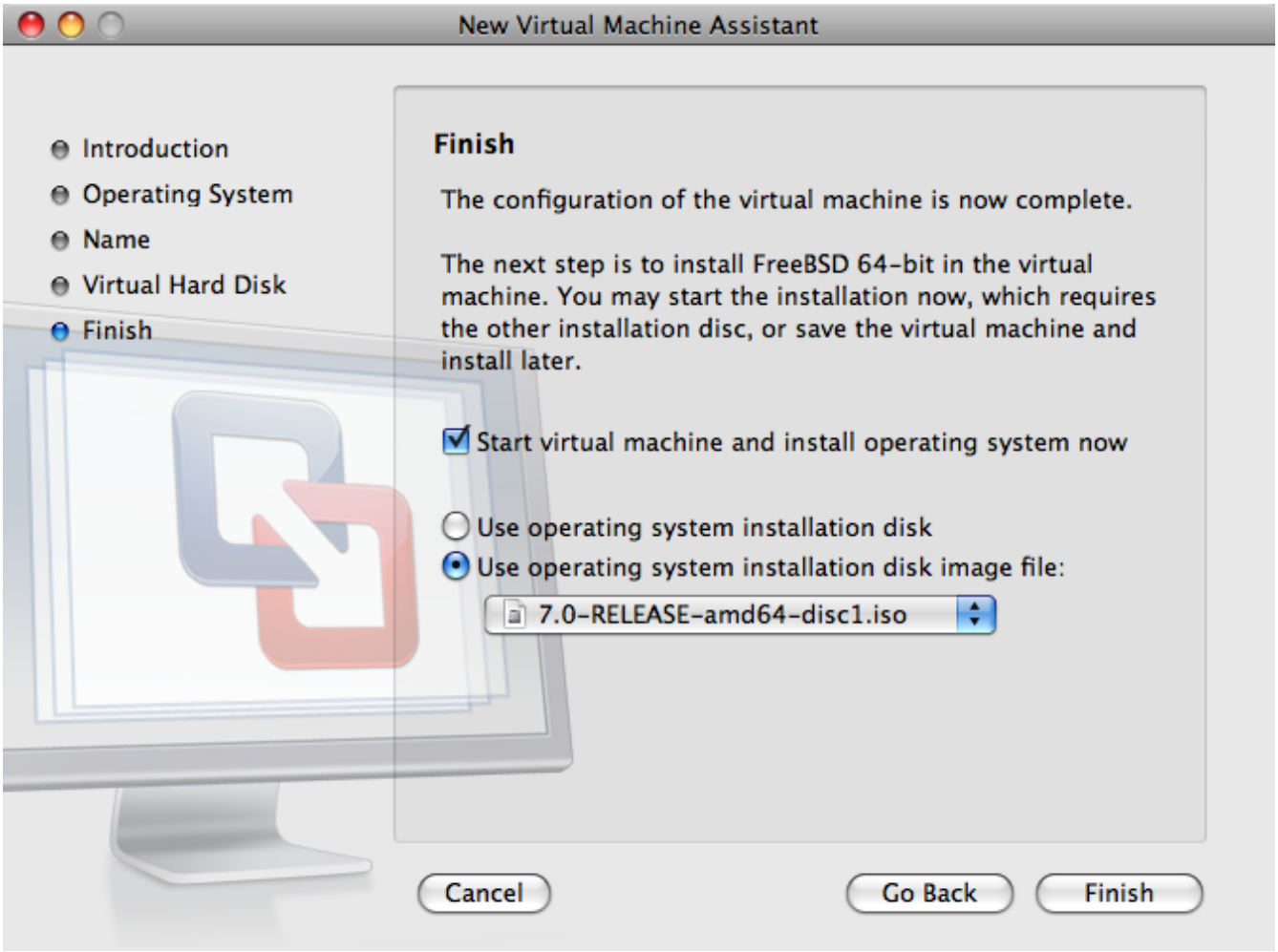
□一个□想要的 VM □像名字和存□的目□位置。


New Virtual Machine Assistant
Introduction
Operating System
Name
Virtual Hard Disk
Finish
Name and Location
Select a name for the virtual machine and choose the folder where this virtual machine will be created.
Save as: FreeBSD 64-bit
Where: Virtual Machines
Cancel
Go Back
Continue

□□ VM 虚□硬□的大小：

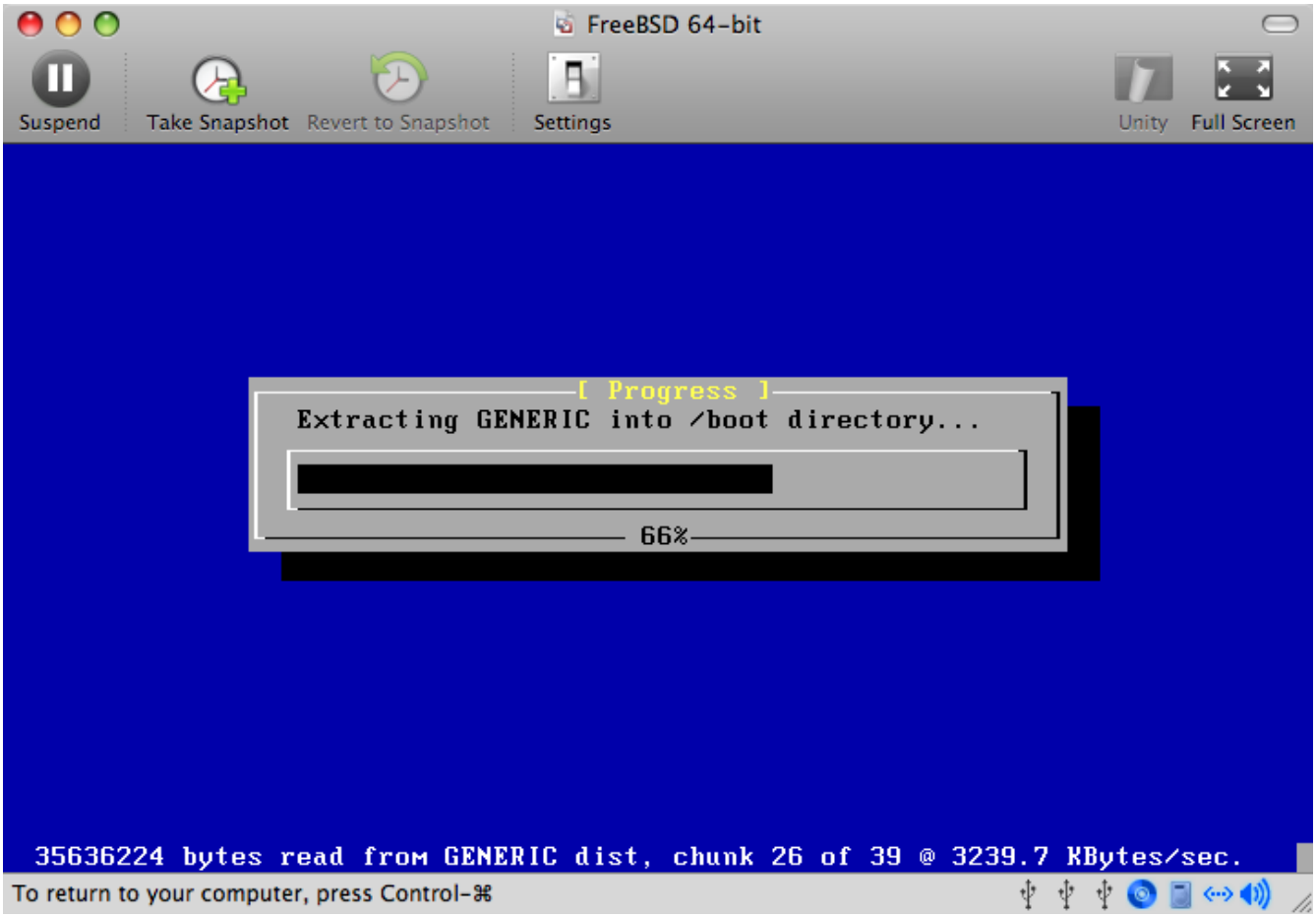


□□安装 VM 的方式， 从一个 ISO □像或一□ CD 安装：



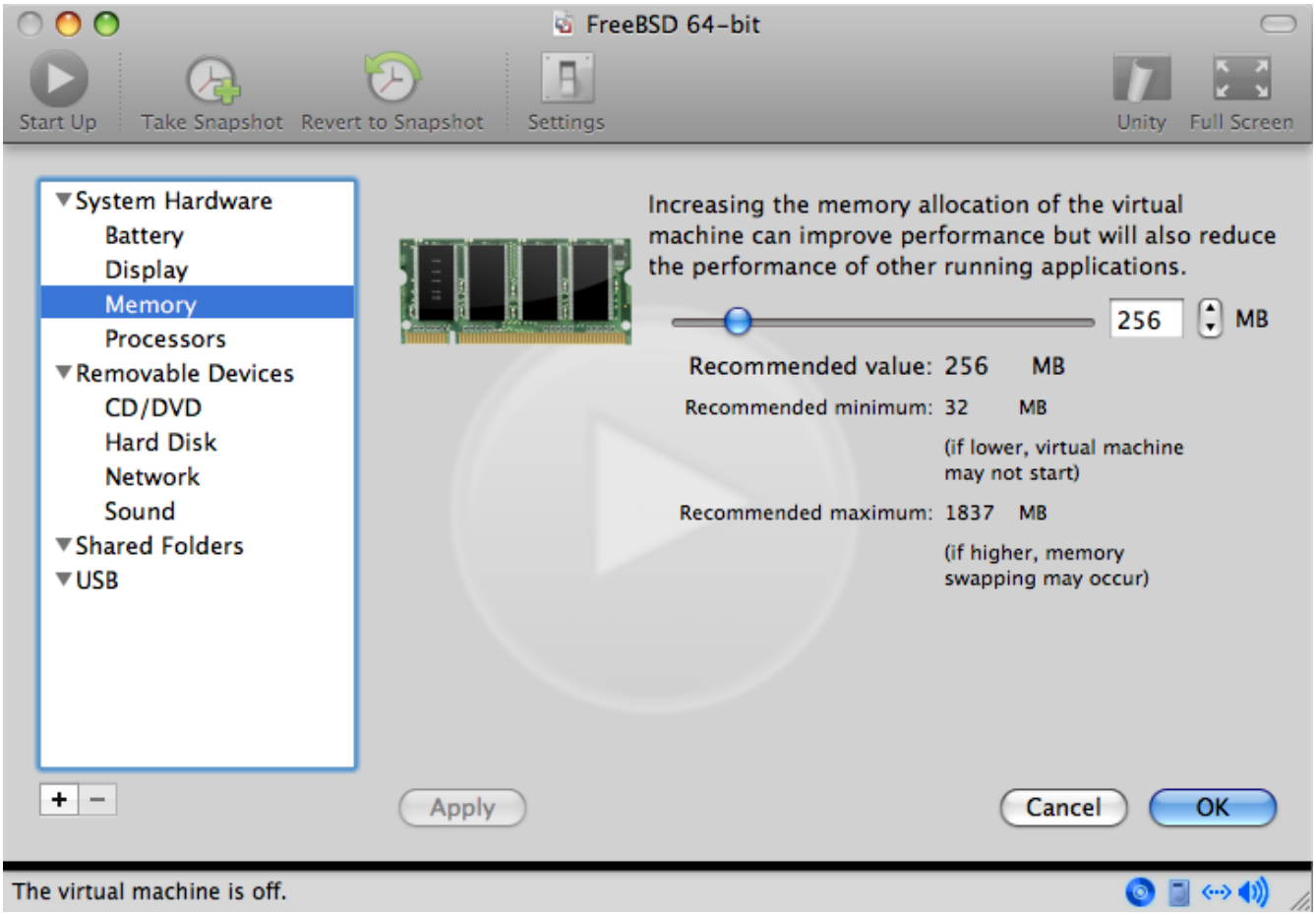
一旦□点□了 Finish， VM 就会□□了：

以□通常的方式安装 FreeBSD 或者参照 安装 FreeBSD 中的□□：

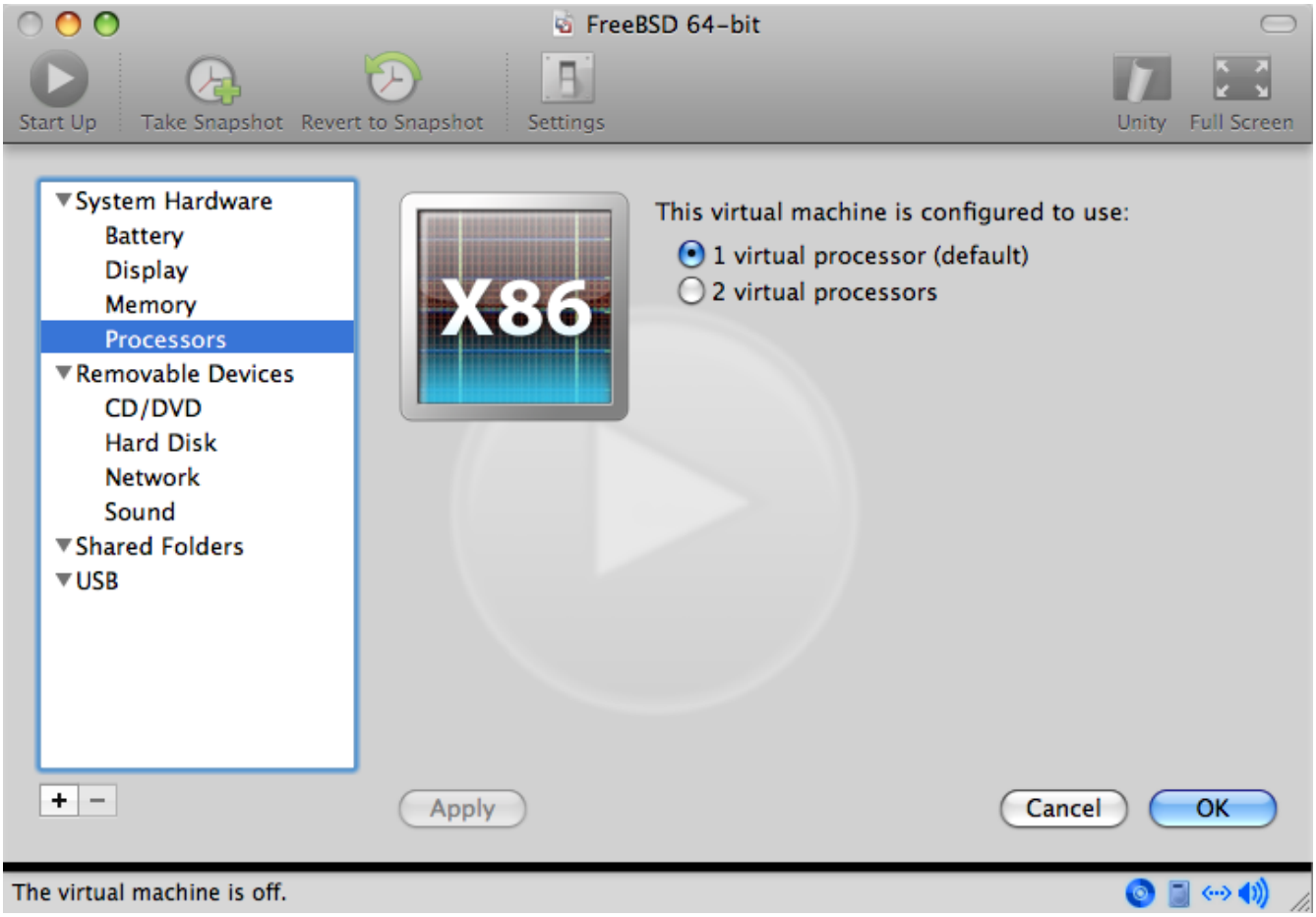


安装完成之后，□就可以修改一些 VM 的□定，比如内存大小：

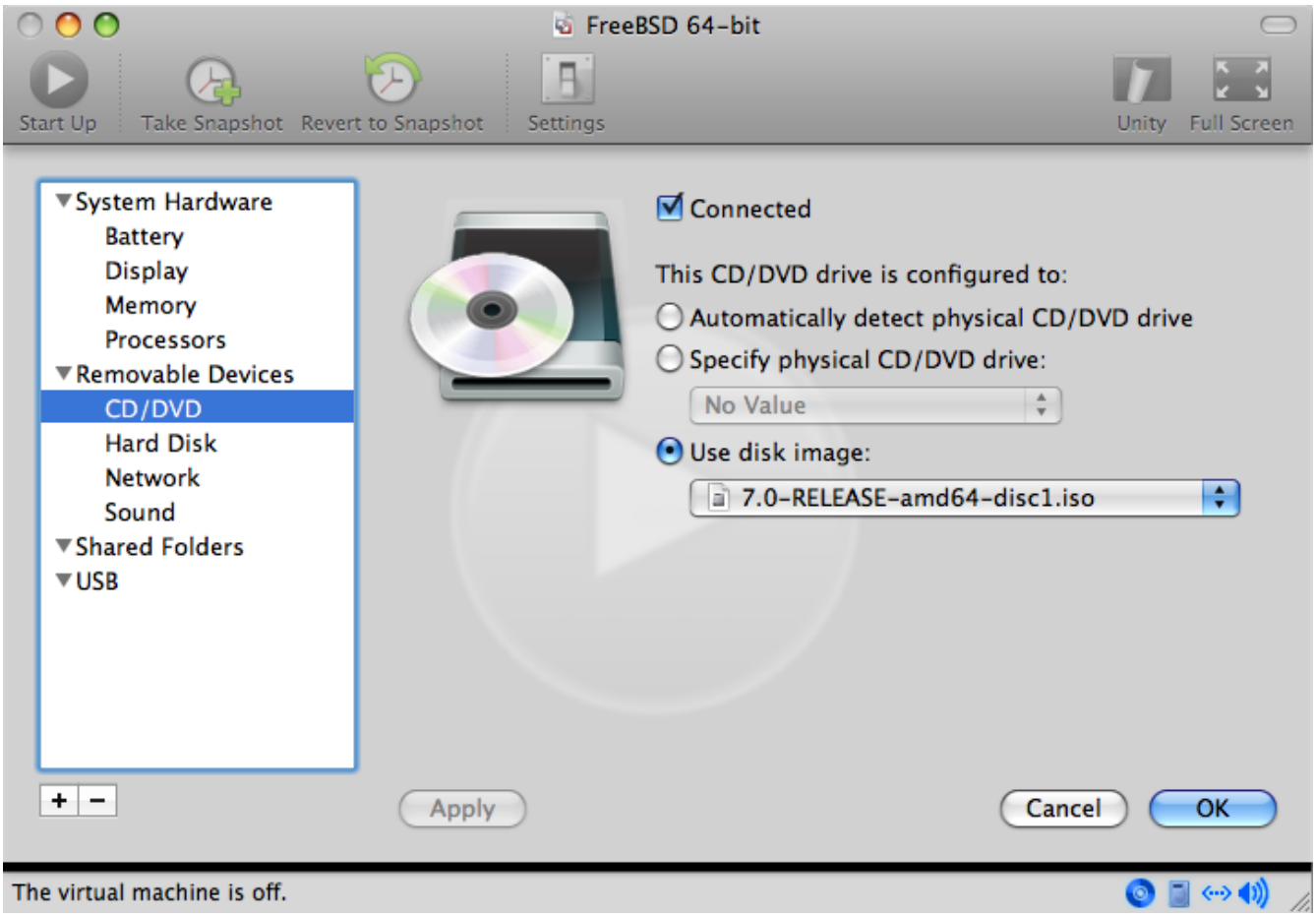
在 VM □行的□候，VM 系□硬件的□置是无法修改的。


FreeBSD 64-bit
Start Up
Take Snapshot
Revert to Snapshot
Settings
Unity
Full Screen
System Hardware
Battery
Display
Memory
Processors
Removable Devices
CD/DVD
Hard Disk
Network
Sound
Shared Folders
USB
Increasing the memory allocation of the virtual machine can improve performance but will also reduce the performance of other running applications.
256
MB
Recommended value: 256 MB
Recommended minimum: 32 MB
(if lower, virtual machine may not start)
Recommended maximum: 1837 MB
(if higher, memory swapping may occur)
Apply
Cancel
OK
The virtual machine is off.

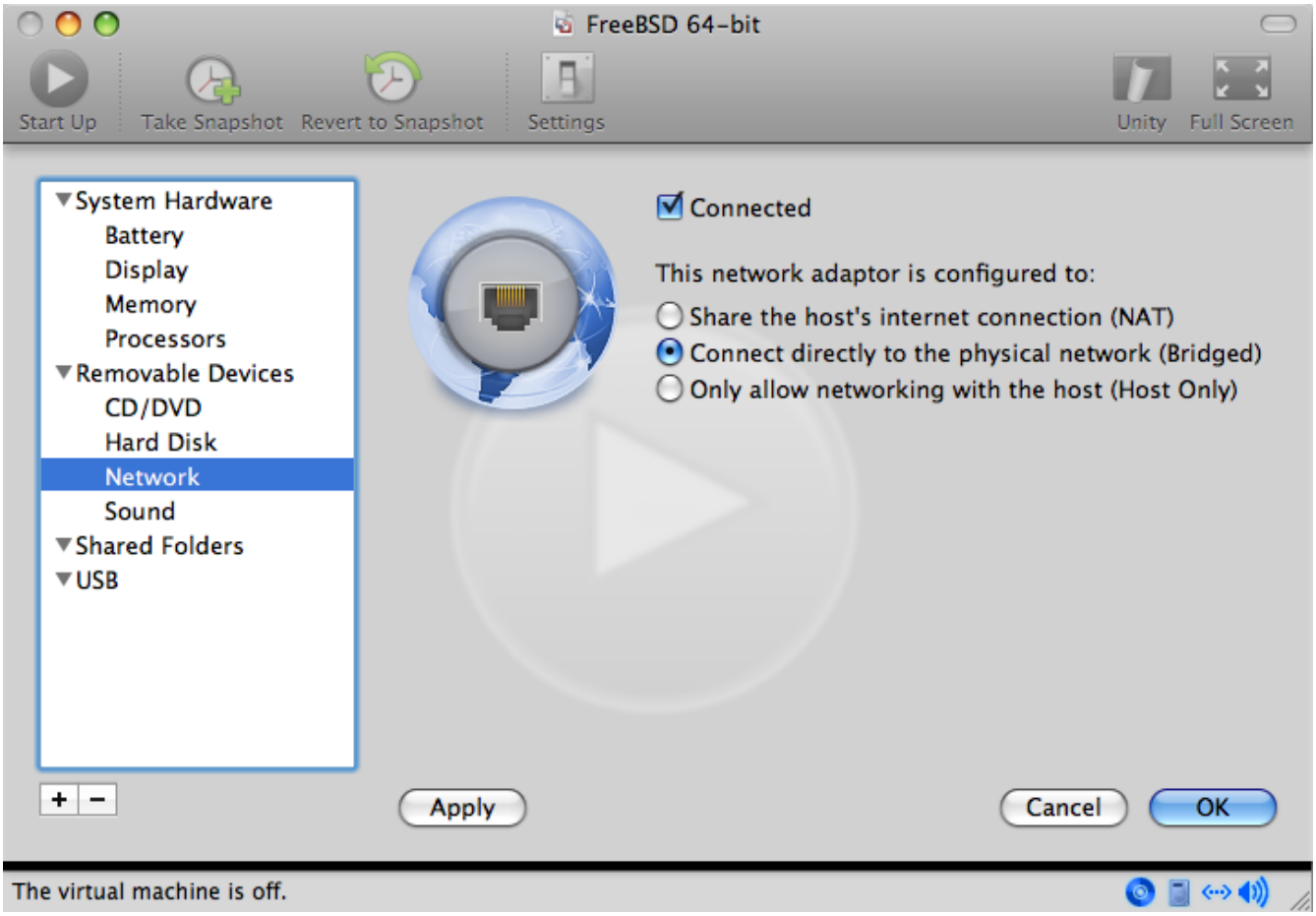
配置 VM 的 CPU 数量：


FreeBSD 64-bit
Start Up
Take Snapshot
Revert to Snapshot
Settings
Unity
Full Screen
System Hardware
Battery
Display
Memory
Processors
Removable Devices
CD/DVD
Hard Disk
Network
Sound
Shared Folders
USB
X86
This virtual machine is configured to use:
1 virtual processor (default)
2 virtual processors
Apply
Cancel
OK
The virtual machine is off.

CD-ROM□□的状□。通常当□不在需要 CDROM/ISO 的□候可以切断他□跟 VM 的□接。



最后一□需要修改的是 VM 与网□□接的方式。 如果□希望除了宿主以外的机器也能□接到 VM， □□□ Connect directly to the physical network (Bridged)。□□ Share the host's internet connection (NAT) 的□， VM 可以□接上网□，但是不能从外面□□。



在□修改完□定之后，就可以从新安装的 FreeBSD 虚□机器□□了。

23.2.3.2. 配置□行于 Mac OS® X/VMware 上的 FreeBSD

在 Mac OS® X 上的 VMware 上安装完 FreeBSD 之后，有些配置的□□可用来□化虚□系□。

1. □置 boot loader □量

   最重要的□□是降低 `kern.hz` 来□少 VMware 上 FreeBSD 的 CPU 使用率。□需要在 /boot/loader.conf 里加入以下□行□定：

   ```
   kern.hz=100
   ```

   如果没有□□□定，VMware 上的 FreeBSD 客□ OS 空□□将占用 iMac® 上一个 CPU 大□ 15% 的□源。在修改此□□定之后□□ 5%。

2. □建一个新的内核配置文件

   □可以去掉所有的 FireWire, USB □□的□□程序。 VMware 提供了一个 em(4) 支持的虚□网□□配器，所以除了 em(4) 之外的网□□□都可以被剔除。

3. □置网□

   最基本的网□□定包括□□的使用 DHCP 把□的虚□机器□接到宿主 Mac® 相同的本地网□上。 在 /etc/rc.conf 中加入： `ifconfig_em0="DHCP"`。 更多有□网□的□置可以参□ 高□网□。

23.3. 作□宿主 OS 的 FreeBSD

在□去的几年中 FreeBSD 并没有任何可用的并被官方支持的虚□化解决方案。 一些用□曾□使用□利用 Linux® 二□制兼容□□行的 VMware □旧并多半已□□的版本 (比如 emulators/vmware3)。 在 FreeBSD 7.2 □布不久，Sun □源版本 (Open Source Edition OSE) 的 VirtualBox™ 作□一个 FreeBSD 原生的程序出□在了 Ports Collection 中。

VirtualBox™ 是一个□□非常活□， 完全虚□化的□件， 并且可在大部□的操作系□上使用， 包括 Windows®, Mac OS®, Linux® 和 FreeBSD。同□也能把 Windows® 或 UNIX® 作□客□系□□行。 它有一个□源和一个私有□□版本。 从用□的角度来看， OSE 版本最主要的限制也□是缺乏 USB 的支持。 其他更多的差□可以通□□接 http://www.virtualbox.org/wiki/Editions □看 "Editions" □面。 目前， FreeBSD 上只有 OSE 版本可用。

23.3.1. 安装 VirtualBox™

VirtualBox™ 已作□一个 FreeBSD port 提供， 位于 emulators/virtualbox-ose， 可使用如下的命令安装：

```
# cd /usr/ports/emulators/virtualbox-ose
# make install clean
```

在配置□□框中的一个有用的□□是 `GusetAdditions` 程序套件。 □些在客□操作系□中提供了一些有用的特性，比如集成鼠□指□ (允□在宿主和客□系□□使用鼠□， 而不用事先按下某个特定的快捷□来切□) 和更快的□□□染，特□是在 Windows® 客□系□中。 在安装了客□操作系□之后， 客□附加□件可在 **Devices** 菜□中□到。

在第一次□行 VirtualBox™ 之前□需要做一些配置上的修改。port 会安装一个内核模□至 /boot/modules 目□， 此模□需要事先加□：

```
# kldload vboxdrv
```

可以在 /boot/loader.conf 中加入以下的配置使此模□在机器重□之后能自□加□：

```
vboxdrv_load="YES"
```

在 3.1.2 之前版本的 VirtualBox™ 需要挂接 proc 文件系□。 在新版本中不再有此要求， 因□它□使用了由 sysctl(3) □提供的功能。

当使用旧版本的 port □， 需要使用下面的□□来挂□ proc：

```
# mount -t procfs proc /proc
```

□了使配置能在重□后始□生效， 需要在 /etc/fstab 中加入以下□行：

```
proc    /proc   procfs  rw  0   0
```

如果在□行 VirtualBox™ 的□端中□□了□似如下的□□消息：

```
VirtualBox: supR3HardenedExecDir: couldn't read "", errno=2 cchLink=-1
```

此故障可能是由 proc 文件系□□致的。 □使用 `mount` 命令□□文件系□是否正□挂□。

在安装 VirtualBox™ □会自□□建 `vboxusers` □。 所有需要使用 VirtualBox™ 的用□必□被添加□此□中的成□。 可以使用 `pw` 命令添加新的成□：

```
# pw groupmod vboxusers -m yourusername
```

□行 VirtualBox™， 可以通□□□□当前□形□境中的 Sun VirtualBox， 也可以在虚□□端中□入以下的命令:

```
% VirtualBox
```

□得更多有□配置和使用 VirtualBox™ 的信息， □□□官方网站 http://www.virtualbox.org。 □于 FreeBSD port 非常新， 并仍□于□□状□。□□看 FreeBSD wiki 上的相□□面 http://wiki.FreeBSD.org/VirtualBox 以□取最新的信息和故障排□□□。

Chapter 24. 本地化－I18N/L10N使用和□置

24.1. 概述

FreeBSD是一个由分布于全世界的用□和□献者支持的□目。 □章将□□FreeBSD的国□化和本地化的□□,允□非英□用□也能使用FreeBSD很好地工作。 在系□和□用水平上，主要是通□□行i18N□准来□□的，所以□里我□将□□者提供□□的介□。

□完□一章，□将了解：

- 不同的□言和地域是如何在□代操作系□上□行□□的。
- 如何□□的登入shell□置本地化。
- 如何配置□的控制台□非英□□言。 languages.
- 如何使用不同的□言来有效地使用X Windows。
- 在□里可以□到更多有□□□符合i18N□准的□用程序的信息。

□□□章之前，□□当了解：

- □□安装□外的第三方程序（安装□用程序. Packages 和 Ports）。

24.2. 基□知□

24.2.1. I18N/L10N 是什□？

□□人□把internationalization□写成I18N,中□的数字是前后□个字母□的字母个数。 L10N依据"localization"使用同□的命名□□。 I18N/L10N方法、□□和□用□合在一起，允□用□使用他□自己所□□的□言。

I18N□用程序使用I18N工具来□程。它允□□□人□写一个□□的文件， 就可以将□示的菜□和文本翻□成本地□言。我□非常鼓励程序□遵循□□□□。

24.2.2. □什□要使用I18N/L10N?

I18N/L10N□准能□很好地支持□□看、□入或□理非英□□言。

24.2.3. I18N支持□些□言？

I18N和L10N不是FreeBSD特有的。当前，它能支持世界上□大部分主力□言，
包括但不限于：中文，德文，日文，朝□文，法文，俄文，越南文等等。

24.3. 使用本地化□言

I18N不是FreeBSD特有的，它是一个□□。我□鼓励□□助FreeBSD完善□一□□。

本地化□置需要具□三个条件：□言代□ (Language Code)、 国家代□ (Country Code) 和□□(Encoding)。本地名字可以用下面□些部分来□造：

□言代□_国家代□.□□

24.3.1. □言和国家代□

□了用特殊的□言来□FreeBSD系□□行本地化（或其他□UNIX®系□），　　用□必□要知道相□的国家和□言代□（国家代□告□□用程序使用□一□□言□□）。　此外，WEB□□器，SMTP/POP服□器，web服□器等都是以□个□基□的。下面就是一个国家和□言代□的例子:

□言/国家代□	描述
en_US	美国英□
ru_RU	俄□
zh_CN	□体中文

24.3.2. □□

一些□言不使用 ASCII □□，它□使用8-位，□或多字□的字符，更多的信息□参考 multibyte(3)。比□老的□用程序可能会无法□□它□，并□□□是控制字符。比□新的□用程序通常会□出 8-位字符。随□□的不同，用□可能不得不将□或多字□字符支持□入□用程序，或□行一些□外的配置，才能□正常使用它□。要□入和□理□或多字□字符，FreeBSD Ports Collection 已□□□□言提供了不同的程序。□参考各个 FreeBSD Port 中的 I18N 文□。

特□需要指出的是，用□可能需要□看□用程序的文□，以□定如何正□地配置它，或需要□configure/Makefile/□□器 指定什□□的参数。

□住下面□些:

- 特定□言的□□C字符集 (参□ multibyte(3))，例如 ISO8859-1, ISO8859-15, KOI8-R, CP437。
- □字□或多字□□□，如EUC, Big5。

□可以在IANA Registry□□一下□行的字符集列表。

与此不同的是， FreeBSD 使用与 X11-兼容的本地□□模式。

24.3.3. I18N□用程序

在FreeBSD Ports和Package系□里面，I18N□用程序已□使用I18N 来命名。然而它□不是□支持需要的□言。

24.3.4. 本地化□置

通常只要在登入shell里面□置LANG□本地化，一般通□□置用□的 ~/.login_conf 或用□shell的□□文件（~/.profile，~/.bashrc，~/.cshrc）。没有必要□置 LC_CTYPE，LC_CTIME。更多的信息□参考特定□言的FreeBSD文□。

□□当在□的配置文件中□置下面□个□量：

- LANG □POSIX®□置本地化□言功能。

- MM_CHARSET□用程序的MIME字符集。

□包括用□的shell配置，特定的□用配置和X11配置。

24.3.4.1. □置本地化的方法

有□□方法来□置本地化，接下来都会描述。 第一□ (推□) 就是在 登入分□里面指定□境□量。 第二□方法是把□境□量加到shell的□□文件里面。

24.3.4.1.1. 登入分□方法

□□方法允□把本地化名称和MIME字符集的□境□量□□可能的shell， 而不是加到□个特定shell的□□文件里面。用□□□置 Level Setup 允□普通用□自己完成□个□置，而管理□□□置需要超□用□□限。

24.3.4.1.1.1. 用□□□置

□有一个□置用□根目□文件.login_conf的小例子， 它□上述□个□量□置了Latin-1□□。

```
me:\
    :charset=ISO-8859-1:\
    :lang=de_DE.ISO8859-1:
```

□是一个□.login_conf□置繁体中文的BIG-5□□的例子。□□□置下面的大部分□量， 因□很多□件都没有□中文，日文和□文□置正□的本地化□量。

```
#Users who do not wish to use monetary units or time formats
#of Taiwan can manually change each variable
me:\
    :lang=zh_TW.Big5:\
    :setenv=LC_ALL=zh_TW.Big5:\
    :setenv=LC_COLLATE=zh_TW.Big5:\
    :setenv=LC_CTYPE=zh_TW.Big5:\
    :setenv=LC_MESSAGES=zh_TW.Big5:\
    :setenv=LC_MONETARY=zh_TW.Big5:\
    :setenv=LC_NUMERIC=zh_TW.Big5:\
    :setenv=LC_TIME=zh_TW.Big5:\
    :charset=big5:\
    :xmodifiers="@im=gcin": #Set gcin as the XIM Input Server
```

更多的信息参考管理□□□置和login.conf(5)

24.3.4.1.2. 管理□□□置

□□用□的登入分□在 /etc/login.conf里面是否□置了正□的□言。主要□定下面的几个□置：

```
language_name|Account Type Description:\
    :charset=MIME_charset:\
    :lang=locale_name:\
    :tc=default:
```

再次使用前面的Latin-1□□的例子：

```
german|German Users Accounts:\
    :charset=ISO-8859-1:\
    :lang=de_DE.ISO8859-1:\
    :tc=default:
```

在修改用□的登入□型之前， □首先□行下面的命令：

```
# cap_mkdb /etc/login.conf
```

以便使在 /etc/login.conf 中新□的配置生效。

24.3.4.1.3. 使用 vipw(8) 改□登入□型。

使用vipw添加新用□，看起来像下面□□：

```
user:password:1111:11:language:0:0:User Name:/home/user:/bin/sh
```

24.3.4.1.4. 用adduser(8)改□登入□型。

用adduser添加新用□看起来像下面□□：

- 在/etc/adduser.conf里面□置defaultclass = □言。□□□住，□必□□使用其它□言的所有用□□置 缺省□□。
- □一次使用adduser(8)的□候，一个特定□言的可□□性回答会像下面□□□出：

  ```
  Enter login class: default []:
  ```

- 如果□打算□□一个用□使用□外一□□言，□□□□□：

  ```
  # adduser -class language
  ```

24.3.4.1.5. 使用pw(8)改□登入□型。

如果□使用pw(8)来添加新用□，□□□□使用：

```
# pw useradd user_name -L language
```

24.3.4.1.6. Shell□□文件方法

不推□使用□□方法，因□它需要□□一个可能的shell程序一个不同的□□文件。 □□用登入分□方法来代替□□方法。

□了□置本地化名称和MIME字符集，只要在/etc/profile或 /etc/csh.login□□文件里面□置□□个□量。下面我□使用德□做例子：

在/etc/profile里面：

```
LANG=de_DE.ISO8859-1; export LANG
MM_CHARSET=ISO-8859-1; export MM_CHARSET
```

或在/etc/csh.login里面：

```
setenv LANG de_DE.ISO8859-1
setenv MM_CHARSET ISO-8859-1
```

□外，□可以把上面的□置添加到/usr/shared/skel/dot.profile （和前面的/etc/profile一□），或者/usr/shared/skel/dot.login （和前面的/etc/csh.login一□）。

□于X11：

在$HOME/.xinitrc里面：

```
LANG=de_DE.ISO8859-1; export LANG
```

或者：

```
setenv LANG de_DE.ISO8859-1
```

依□□的shell(看上面）。

24.3.5. 控制台□置

□于所有的□□C字符集，在/etc/rc.conf中用正在□□的□言□置正□的控制台字符：

```
font8x16=font_name
font8x14=font_name
font8x8=font_name
```

□儿的*font_name*来自于/usr/shared/syscons/fonts目□， 不□.fnt后□。

如果需要的□， □□通□ `sysinstall` 来配置与□字□ C 字符集□□的 keymap 和 screenmap。 在 sysinstall 中，□□ Configure 之后□□ Console 即可□行配置。 除此之外， □也可以在 /etc/rc.conf 中加入□似下面的配置：

```
scrnmap=screenmap_name
keymap=keymap_name
keychange="fkey_number sequence"
```

□儿的*screenmap_name*是来自/usr/shared/syscons/scrnmaps目□，　　不□.scm后□。　　一个□影射字体的屏幕布局通常被作□一个工作区，　　用来在VGA□配器字体矩□上□展8位到9位。如果屏幕字体是使用一个8位的排列，要移□□些字母□□□些区域。

如果□在/etc/rc.conf里面□用了moused daemon：

```
moused_enable="YES"
```

那□需要在下一段□□鼠□指□信息。

默□情况下，　syscons(4)□□程序的鼠□指□在字符集中占用0xd0-0xd3的□□。　如果□的□言使用□个□□，□必□把指□□□移出□个□□。 要□□□个□□， 需要在 /etc/rc.conf 中加入：

```
mousechar_start=3
```

□里， *keymap_name* 来自于 /usr/shared/syscons/keymaps 目□， 但去掉了 .kbd 后□。 如果不□定□□使用□一个□□布局， □可以使用 kbdmap(1) 来□□， 而无需反□重□。

通常， `keychange` 是□定功能□□， 匹配□定的□端□型来□是必需的， 因□功能□序列无法在□□布局中定□。

此外□□□□□□并□□在 /etc/ttys 中已□□所有的 `ttyv*` □配置了正□的□端□型。 目前， 相□的默□定□是：

字符集□置	□端□型
ISO8859-1 or ISO8859-15	`cons25l1`
ISO8859-2	`cons25l2`
ISO8859-7	`cons25l7`
KOI8-R	`cons25r`
KOI8-U	`cons25u`
CP437 (VGA default)	`cons25`
US-ASCII	`cons25w`

□于多字□字符□言，可以□的在 /usr/ports/language 目□中使用正□的FreeBSD port。一些port以控制台出□，　　而系□把它作□串行vtty□端，因此，　　必□□　　X11　　和□串行控制台准□足□的vtty□端。下面是在控制台中使用其他□言的□用程序的部分列表：

□言	特定区域
Traditional Chinese (BIG-5)	chinese/big5con
Japanese	japanese/kon2-16dot or japanese/mule-freewnn

□言	特定区域
Korean	korean/han

24.3.6. X11□置

□然X11不是FreeBSD□□的一部分， 但我□已□□FreeBSD用□包含了一些信息。 具体□□可以参考Xorg Web 站点 或是□使用的 X11 Server 的网站。

在~/.Xresources里面，□可以□当□整特定□用程序的I18N□置（如字体，菜□等）。

24.3.6.1. □示字体

安装 Xorg 服□器 (x11-servers/xorg-server)， 然后安装□□□言的 TrueType® 字体。 □□置正□的地区信息，□将□□能□在菜□和其它地方看到所□□的□言。

24.3.6.2. □入非英□字符

X11□入方法（XIM）□□是所有X11客□端的一个新□准。 所有将作□XIM客□端来写的X11□用程序从XIM□入服□器□入。 不同的□言有几□XIM服□器可用。

24.3.7. 打印机□置

一些□□的C字符集通常是用硬□□来□□□打印机的。更□或多位的字符集需要特定的□置， 我□推□使用apsfilter。□也可以使用特定□言□□器把文□□□□ PostScript®或PDF格式。

24.3.8. 内核和文件系□

FreeBSD 的快速文件系□ (FFS) 是完全支持 8-位 字符的， 因此它可以被用于任何□□的 C 字符集 (参□ multibyte(3))， 但在文件系□中不会保存字符集的名字； 也就是□， 它不加修改地保存 8-位信息，而并不知道如何□□。 正式□来， FFS 目前□不支持任何形式的□或多字□字符集。 不□， 某些□或多字符集提供了独立的□□ FFS 的□丁来□助□用□于它□的支持。 目前□些要□是无法移植的， 要□□于粗□，因此我□不打算把它□加入到源代□中。 □参考相□□言的 Web 站点， 以了解□于□些□丁的□一□情况。

FreeBSD MS-DOS®已□能□配置成用在MS-DOS®上，Unicode字符集和可□的FreeBSD文件系□字符集的更多信息， □参考 mount_msdosfs(8) □机手册。

24.4. □□I18N程序

□多FreeBSD Ports已□支持I18N了。他□中的一些都用-I18N作□□。 □些和其他很多程序已□内建I18N的支持，不需要考□其他的事□了。

然而一些像MySQL□□的□用程序需要重新配置字符集，可在 Makefile里面□置，或者直接把参数□□□configure。

24.5. 本地化FreeBSD

24.5.1. 俄□（KOI8-R□□）

□于KOI8-R□□的更多信息□□□KOI8-R参考（Russian Net Character Set）。

24.5.1.1. 本地□置

把下面的行加入到□的~/.login_conf文件：

```
me:My Account:\
    :charset=KOI8-R:\
    :lang=ru_RU.KOI8-R:
```

参看前面的□置本地化的例子。

24.5.1.2. 控制台□置

- 把下面一行加到 /etc/rc.conf：

  ```
  mousechar_start=3
  ```

- 并在 /etc/rc.conf 里面□加如下□置：

  ```
  keymap="ru.utf-8"
  scrnmap="utf-82cp866"
  font8x16="cp866b-8x16"
  font8x14="cp866-8x14"
  font8x8="cp866-8x8"
  ```

- □于/etc/ttys里面的ttyv*□□，要使用 cons25r作□□端□型。

参看前面的□置控制台的例子。

24.5.1.3. 打印机□置

既然□大多数□俄□字符的打印机遵循CP866的□准，　　那□需要一个□□KOI8-R到CP866□□的特定□出□□器。□□的一个□□器默□的安装在　　/usr/libexec/lpr/ru/koi2alt。　　一个支持俄□的打印机的/etc/printcap□□看起来是□□的：

```
lp|Russian local line printer:\
    :sh:of=/usr/libexec/lpr/ru/koi2alt:\
    :lp=/dev/lpt0:sd=/var/spool/output/lpd:lf=/var/log/lpd-errs:
```

更多信息参考printcap(5)手册□。

24.5.1.4. MS-DOS®文件系□和俄□文件名

下面的例子是在挂上MS-DOS® 文件系□后，□用□俄□文件名支持的fstab(5)□□：

```
/dev/ad0s2      /dos/c  msdos   rw,-Wkoi2dos,-Lru_RU.KOI8-R 0 0
```

□□ -L 用于□□地区名称， 而 -W □用于□置字符□□表。 要使用 -W □□， □一定要首先挂接 /usr， 然后再挂接 MS-DOS® 分区， 因□□□表是放在 /usr/libdata/msdosfs 的。 要了解□一□的□□， □参考 mount_msdosfs(8) □机手册。

24.5.1.5. X11□置

1. 首先□□行前面介□的 非-X 的本地化□置。
2. 如果□正使用 Xorg， □安装 x11-fonts/xorg-fonts-cyrillic package。

   □□□ /etc/X11/xorg.conf 文件中的 "Files" 小□。 下面的行， □加到任何其它 FontPath □之前：

   ```
   FontPath   "/usr/local/lib/X11/fonts/cyrillic"
   ```

   □□看 ports 中的其它西里□字体。

3. 要激活俄□□□， 需要在 xorg.conf 文件的 "Keyboard" 小□中加入下列内容：

   ```
   Option "XkbLayout"   "us,ru"
   Option "XkbOptions"  "grp:toggle"
   ```

   要□信XkbDisable 已□□□ (注□掉) 了。

   RUS/LAT的切□用CapsLock。老的CapsLock功能可以通□ Shift + CapsLock 来模□（只有在LAT模式的□候）。

   使用 grp:toggle □， RUS/LAT 切□□将是 右 Alt， 而使用 grp:ctrl_shift_toggle □表示切□□是 Ctrl + Shift。 使用 grp:caps_toggle □， RUS/LAT 切□□□是 CapsLock。 旧的 CapsLock 功能仍可通□ Shift + CapsLock (只□ LAT 模式有效)。 由于不明原因， grp:caps_toggle 在 Xorg 中无法使用。

   如果□的□□上有 "Windows®" □， 但□□ RUS 模式下， 某些非字母□映射不正常， □□在□的 xorg.conf 文件中加入下面□行：

   ```
   Option "XkbVariant" ",winkeys"
   ```

   俄□的 XKB □□可能并不□某些不具□本地化功能的□用程序所支持。

本地化程序最低限度□在程序□□□□用 `XtSetLanguageProc (NULL, NULL, NULL);` 函数。

参□ KOI8-R for X Window 以□得□于□ X11 □用□行本地化的指□。

24.5.2. □置繁体中文

FreeBSD-Taiwan□□有一个使用很多中文ports的中文化指南在 http://netlab.cse.yzu.edu.tw/~statue/freebsd/zh-tut/。 目前， `FreeBSD 中文化指南` 的□□人□是 沈俊□ statue@freebsd.sinica.edu.tw。

沈俊□ statue@freebsd.sinica.edu.tw 利用 FreeBSD-Taiwan 的 `zh-L10N-tut`建立了 Chinese FreeBSD Collection (CFC)。 相□的 packages 和脚本等可以在 ftp://freebsd.csie.nctu.edu.tw/pub/taiwan/CFC/ □到。

24.5.3. 德□本地化（□合所有的ISO 8859-1□言）

Slaven Rezic eserte@cs.tu-berlin.de 写了一个在 FreeBSD 机器下如何使用日□曼□言的德□指南。 □□德□教程可以在 http://user.cs.tu-berlin.de/~eserte/FreeBSD/doc/umlaute/umlaute.html □到。

24.5.4. 希□□本地化

Nikos Kokkalis nickkokkalis@gmail.com 撰写了□于在 FreeBSD 上支持希□□的完整文章， 在 http://www.freebsd.org/doc/el/articles/greek-language-support/。 □注意□篇文章 只有 希□□的版本。

24.5.5. 日□和□□本地化

日□本地化□参考http://www.jp.FreeBSD.org/，□□参考 http://www.kr.FreeBSD.org/。

24.5.6. 非英□的FreeBSD文□

一些 FreeBSD 的□献者已□将部分 FreeBSD 文□翻□成了其他□言。 □可在 主站 以及 /usr/shared/doc □到。

Chapter 25. 更新与升□ FreeBSD

25.1. 概述

FreeBSD 在□行版之□始□是持□□□的。 一些人喜□使用官方□行的版本， □一些喜□与最新的□□保持同□。 然而， 即使是官方的□行版本也常常需要安全□丁和重大修正方面的更新。 不□□使用了何□版本， FreeBSD 都提供了所有更新系□所需的工具， □□□松的在不同版本□升□。 □一章□将□助□决定是跟踪□□系□□是□持使用某个□行的版本。 同□□列出了一些保持系□更新所需的基本工具。

□了本章后，□将了解到：

- 使用□些工具来更新系□与 Ports Collection。
- 如何使用 freebsd-update, CVSup, CVS, or CTM □□的系□保持更新。
- 如何比□已安装的系□与原来已知拷□的状□。
- 如何使用 CVSup 或者文□ ports 来更新本地的文□。
- □个□□分支 FreeBSD-STABLE 和 FreeBSD-CURRENT 的区□。
- 如何通□ `make buildworld` 重新□□安装整个基本系□(等等)。

在□本章□前，□□□了解的：

- 正□□置网□□接 (高□网□)。
- 知道□□安装附加的第三方□件(安装□用程序. Packages 和 Ports)。

整个□一章中，`cvsup` 命令都被用来□取 FreeBSD 源代□的更新。 □需要安装 net/cvsup port 或者二□制包(如果□不想要安装□形界面的 `cvsup` 客□端的□， □可以安装 net/cvsup-without-gui port)。 □也可以使用 csup(1) 代替， 它□在已□是基本系□的一部分了。

25.2. FreeBSD 更新

打安全□丁是□于□□□算机□件的一个重要部分， 特□是□于操作系□。□于 FreeBSD 来□， 很□的一段□□以来□都不是一件容易的事情。 □丁打在源代□上，代□需要被重新□□□二□制， 然后再重新安装□□后的程序。

FreeBSD 引入了 `freebsd-update` 工具之后□便不再是□□了。□个工具提供了 2 □功能。 第一，它可以把二□制的安全和勘□更新直接□用于 FreeBSD 的基本系□，而不需要重新□□和安装。第二， □个工具□支持主要跟次要的□行版的升□。

由安全小□支持的各□体系□□和□行版都可使用二□制更新。 在升□到一个新的□行版本之前，□先□□一下当前□行版的声明， 因□它□可能包含有□于□期望升□版本的重要消息。 □些□行声明可以通□以下□接□□： http://www.FreeBSD.org/releases/。

如果 `crontab` 中存在有用到 `freebsd-update` 特性的部分， 那□□些在□始以下操作前必□先被禁止。

25.2.1. 配置文件

有些用□可能希望通□□整配置文件 /etc/freebsd-update.conf 中的默□配置来更好地控制升□的□程。

可用的参数在文□中介□的很□□， 但下面的□些可能需要□一□的解□：

```
# Components of the base system which should be kept updated.
Components src world kernel
```

□个参数是控制 FreeBSD 的□一部分将被保持更新。 默□的是更新源代□，整个基本系□□有内核。□些部件跟安装□的那些相同，□例来□， 在□里加入 world/games 就会允□打入游□相□的□丁。 使用 src/bin □是允□更新 src/bin 目□中的源代□。

最好的□□是把□个□□保留□默□□， 因□如果要修改它去包含一些指定的□□， 就需要用□列出□一个想要更新的□目。 □可能会引起可怕的后果， 因□部分的源代□和二□制程序得不到同□。

```
# Paths which start with anything matching an entry in an IgnorePaths
# statement will be ignored.
IgnorePaths
```

添加路径，比如 /bin 或者 /sbin □□些指定的目□在更新□程中不被修改。 □个□□能□防止本地的修改被 freebsd-update 覆□。

```
# Paths which start with anything matching an entry in an UpdateIfUnmodified
# statement will only be updated if the contents of the file have not been
# modified by the user (unless changes are merged; see below).
UpdateIfUnmodified /etc/ /var/ /root/ /.cshrc /.profile
```

更新指定目□中的未被修改的配置文件。 用□的任何修改都会使□些文件的自□更新失效。 □有□外一个□□，KeepModifiedMetadata， □个能□ freebsd-update 在合并□保存修改。

```
# When upgrading to a new FreeBSD release, files which match MergeChanges
# will have any local changes merged into the version from the new release.
MergeChanges /etc/ /var/named/etc/
```

一个 freebsd-update □□□□合并的配置文件的列表。文件合并的□程是 一系列的 diff(1) □丁□似于更少□□的 mergemaster(8) 合并的□□是接受，打□一个文本□□器，或者 freebsd-update 会被中止。 在不能□定的□候，□先□□ /etc 然后接受合并。更多□于 mergemaster 的信息□参□ mergemaster。

```
# Directory in which to store downloaded updates and temporary
# files used by FreeBSD Update.
# WorkDir /var/db/freebsd-update
```

□个目□是放置所有□丁和□□文件的。 用□做一个版本升□的□，□□□此□至少有 1 GB 的可用磁□空□。

```
# When upgrading between releases, should the list of Components be
# read strictly (StrictComponents yes) or merely as a list of components
# which *might* be installed of which FreeBSD Update should figure out
# which actually are installed and upgrade those (StrictComponents no)?
# StrictComponents no
```

当□置成 yes □， freebsd-update 将假□□个 Components 列表□完整的， 并且□此列表以外的□目不会修改。□□上就是 freebsd-update 会□□更新 Componets 列表里的□一个文件。

25.2.2. 安全□丁

安全□丁存□在□程的机器上， 可以使用如下的命令下□并安装：

```
# freebsd-update fetch
# freebsd-update install
```

如果□内核打了□丁，那□系□需要重新□□。 如果一切都□展□利，系□就□□被打好了□丁而且 freebsd-update 可由夜□ cron(8) □行。在 /etc/crontab 中加入以下条目足以完成□□任□：

```
@daily                                  root    freebsd-update cron
```

□条□□是□明□天□行一次 freebsd-update 工具。 用□□方法， 使用了 cron 参数， freebsd-update □□□是否存在更新。 如果有了新的□丁，就会自□下□到本地的磁□， 但不会自□□系□打上。root 会收到一封□子□件告知需手□安装□丁。

如果出□了□□，可以使用下面的 freebsd-update 命令回退到上一次的修改：

```
# freebsd-update rollback
```

完成以后如果内核或任何的内核模□被修改的□， 就需要重新□□系□。□将使 FreeBSD 装□新的二□制程序□内存。

freebsd-update 工具只能自□更新 GENERIC 内核。 如果□使用自行□□的内核， □在 freebsd-update 安装完更新的其余部分之后需要手工重新□□和安装内核。 不□， freebsd-update 会□□并更新位于 /boot/GENERIC (如果存在) 中的 GENERIC 内核， 即使它不是当前 (正在□行的) 系□的内核。

保存一□ GENERIC 内核的副本到 /boot/GENERIC 是一个明智的主意。 在□断□多□□，以及在 重大和次要的更新 中介□的使用 freebsd-update 更新系□□会很有用。

除非修改位于 /etc/freebsd-update.conf 中的配置， freebsd-update 会随其他安装一起□内核的源代□□行更新。 重新□□并安装定制的内核可以以通常的方式来□行。

通□ freebsd-update □布的更新有□并不会□及内核。 如果在□行 freebsd-update install 的□程中内核代□没有□行□□， 就没有必要重新□□内核了。 不□， 由于 freebsd-update □次都会更新 /usr/src/sys/conf/newvers.sh 文件， 而修□版本 (uname -r □告的 -p 数字) 来自□个文件， 因此， 即使内核没有□生□化， 重新□□内核也可以□ uname(1) □告准□的修□版本。 在□□□多系□□□□做会比□有□助， 因□□一信息可以迅速反映机器上安装的□件更新情况。

25.2.3. 重大和次要的更新

□个□程会□除旧的目□文件和□， □将使大部分的第三方□用程序无法□除。 建□将所有安装的 ports 先□除然后重新安装，或者□后使用 ports-mgmt/portupgrade 工具升□。 大多数用□将会使用如下命令□□□□：

```
# portupgrade -af
```

□将□保所有的□西都会被正□的重新安装。 □注意□境□量 BATCH □置成 yes 的□将在整个□程中□所有□□回答 yes，□会□助在□□□程中免去人工的介入。

如果正在使用的是定制的内核， □升□操作会□□一些。 □会需要将一□ GENERIC 内核的副本放到 /boot/GENERIC。 如果系□中没有 GENERIC 内核， 可以用以下□□方法之一来安装：

- 如果只□□□一次内核， □位于 /boot/kernel.old 中的内核， 就是 GENERIC 的那一个。 只需将□个目□改名□ /boot/GENERIC 即可。
- 假如能□直接接触机器， □可以通□ CD-ROM 介□来安装 GENERIC 内核。 将安装□□入光□， 并□行下列命令：

  ```
  # mount /cdrom
  # cd /cdrom/X.Y-RELEASE/kernels
  # ./install.sh GENERIC
  ```

  □需要将 X.Y-RELEASE 替□□□正在使用的版本。 GENERIC 内核默□情况下会安装到 /boot/GENERIC。

- 如果前面的方法都不可用， □可以使用源代□来重新□□和安装 GENERIC 内核：

  ```
  # cd /usr/src
  # env DESTDIR=/boot/GENERIC make kernel
  # mv /boot/GENERIC/boot/kernel/* /boot/GENERIC
  # rm -rf /boot/GENERIC/boot
  ```

  如果希望 freebsd-update 能□正□地将内核□□□ GENERIC， □必□□保没有□ GENERIC 配置文件□行□任何□□。 此外， 建□□取消任何其他特殊的□□□□ (例如使用空的 /etc/make.conf)。

上述□□并不需要使用□个 GENERIC 内核来引□系□。

重大和次要的更新可以由 freebsd-update 命令后指定一个□行版本来□行， □例来□，下面的命令将□助□升□到 FreeBSD 8.1：

```
# freebsd-update -r 8.1-RELEASE upgrade
```

在□行□个命令之后，freebsd-update 将会先解析配置文件和□估当前的系□以□得更新系□所需的必要信息。然后便会□示出一个包含了已□□到与未□□到的□件列表。 例如：

```
Looking up update.FreeBSD.org mirrors... 1 mirrors found.
Fetching metadata signature for 8.0-RELEASE from update1.FreeBSD.org... done.
Fetching metadata index... done.
Inspecting system... done.

The following components of FreeBSD seem to be installed:
kernel/smp src/base src/bin src/contrib src/crypto src/etc src/games
src/gnu src/include src/krb5 src/lib src/libexec src/release src/rescue
src/sbin src/secure src/share src/sys src/tools src/ubin src/usbin
world/base world/info world/lib32 world/manpages

The following components of FreeBSD do not seem to be installed:
kernel/generic world/catpages world/dict world/doc world/games
world/proflibs

Does this look reasonable (y/n)? y
```

此□，freebsd-update 将会□□下□所有升□所需的文件。在某些情况下， 用□可能被□及需安装些什□和如何□行之□的□□。

当使用定制内核□， 前面的□□会□生□似下面的警告：

```
WARNING: This system is running a "MYKERNEL" kernel, which is not a
kernel configuration distributed as part of FreeBSD 8.0-RELEASE.
This kernel will not be updated: you MUST update the kernel manually
before running "/usr/sbin/freebsd-update install"
```

此□□可以□□安全地无□□个警告。 更新的 GENERIC 内核将在升□□程的中□□□中使用。

在下□完□□本地系□的□丁之后， □些□丁会被□用到系□上。 □个□程需要消耗的□□取决于机器的速度和其□□。□个□程中将会□配置文件所做的□□□行合并 - □一部分需要用□的参与， 文件可能会自□合并， 屏幕上也可能会□出一个□□器， 用于手工完成合并操作。 在□理□程中， 合并成功的□果会□示□用□。 失□或被忽略的合并，□会□致□一□程的□止。 用□可能会希望□□一□ /etc 并在□之后手工合并重要的文件， 例如 master.passwd 和 group。

系□至此□没有被修改，所有的□丁和合并都在□外一个目□中□行。 当所有的□丁都被成功的打上了以后，所有的配置文件都被合并后， 我□就已□完成了整个升□□程中最困□的部分， 下面就需要用□来安装□些□更了。

一旦□个□□完成后，使用如下的命令将升□后的文件安装到磁□上。

```
# freebsd-update install
```

内核和内核模□会首先被打上□丁。 此□必□重新□□□算机。 如果□使用的是定制的内核， □使用 nextboot(8) 命令来将下一次用于引□系□的内核 /boot/GENERIC (它会被更新)：

```
# nextboot -k GENERIC
```

在使用 GENERIC 内核□□之前， □□信它包含了用于引□系□所需的全部□□程序 (如果□是在□程□行升□操作， □□□信网□□□也是存在的)。 特□要注意的情形是，如果之前的内核中静□□□了通常以内核模□形式存在的□□程序， 一定要通□ /boot/loader.conf 机制来将□些模□加□到 GENERIC 内核的基□上。 此外， □可能也希望□□取消不重要的服□、 磁□和网□挂□等等， 直到升□□程完成□止。

□在可以用更新后的内核引□系□了：

```
# shutdown -r now
```

在系□重新上□后，需要再次□行 freebsd-update。 升□的状□被保存着，□□ freebsd-update 就无需重□□始，但是会□除所有旧的共享□和目□文件。 □行如下命令□□□个□段的升□：

```
# freebsd-update install
```

取决与是否有□的版本更新，通常只有 2 个而不是 3 个安装□段。

□在需要重新□□和安装第三方□件。 □□做的原因是某些已安装的□件可能依□于在升□□程中已□除的□。 可使用 ports-mgmt/portupgrade 自□化□个□□，以如下的命令□始：

```
# portupgrade -f ruby
# rm /var/db/pkg/pkgdb.db
# portupgrade -f ruby18-bdb
# rm /var/db/pkg/pkgdb.db /usr/ports/INDEX-*.db
# portupgrade -af
```

一旦□个完成了以后，再最后一次□行 freebsd-update 来□束升□□程。 □行如下命令□理升□中的所有□□：

```
# freebsd-update install
```

如果□□□用□ GENERIC 内核来引□系□， □在是按照通常的方法重新□□并安装新的定制内核的□候了。

重新□□机器□入新版本的 FreeBSD 升□□程至此就完成了。

25.2.4. 系□状□□照

freebsd-update 工具也可被用来□着一个已知完好的 FreeBSD 拷□□□当前的版本。 □个□□□估当前的系□工具，□和配置文件。 使用以下的命令□始□照：

```
# freebsd-update IDS >> outfile.ids
```

□个命令的名称是 IDS， 它并不是一个像 security/snort □□的入侵□□系□的替代品。因□ freebsd-update 在磁□上存□数据， 很□然它□有被□改的可能。 当然也可以使用一些方法来降低被□改的可能性，比如□置 kern.securelevel 和不使用□把 freebsd-update 数据放在只□文件系□上，例如 DVD 或 安全存放的外置 USB 磁□上。

□在系□将会被□□，生成一□包含了文件和它□的 sha256(1) 哈希□的清□，已知□行版中的□与当前系□中安装的□将会被打印到屏幕上。 □就是□什□□出被送到了 outfile.ids 文件。 它□□的太□无法用肉眼□照，而且会很快填□控制台的□冲区。

□个文件中有非常□的行，但□出的格式很容易分析。 □例来□，要□得一□与□行版中不同哈希□的文件列表，已可使用如下的命令：

```
# cat outfile.ids | awk '{ print $1 }' | more
/etc/master.passwd
/etc/motd
/etc/passwd
/etc/pf.conf
```

□□□出□□□□短后的，其□是有更多的文件。 其中有些文件并非人□修改，比如 /etc/passwd 被修改是因□添加了用□□系□。在某些情况下， □有□外的一些文件，□如内核模□与 freebsd-update 的不同是因□它□被更新□了。 □了指定的文件或目□排除在外，把它□加到 /etc/freebsd-update.conf 的 IDSIgnorePaths □□中。

除了前面□□□的部分之外， □也能被当作是□升□方法的□□□充。

25.3. Portsnap： 一个 Ports Collection 更新工具

FreeBSD 基本系□也包括了一个更新 Ports Collection 的工具： portsnap(8)。在□行之后，它会□上一个□程网站， 校□安全密□，然后下□一□ Ports Collection 的拷□。 密□是用来校□所有下□文件的完整性，□保它□在□□是未被修改。 使用以下的命令下□最新的 Ports Collection：

```
# portsnap fetch
Looking up portsnap.FreeBSD.org mirrors... 3 mirrors found.
Fetching snapshot tag from portsnap1.FreeBSD.org... done.
Fetching snapshot metadata... done.
Updating from Wed Aug  6 18:00:22 EDT 2008 to Sat Aug 30 20:24:11 EDT 2008.
Fetching 3 metadata patches.. done.
Applying metadata patches... done.
Fetching 3 metadata files... done.
Fetching 90 patches.....10....20....30....40....50....60....70....80....90. done.
Applying patches... done.
Fetching 133 new ports or files... done.
```

□个例子展示的是 portsnap(8) □□并校□了几个用于当前 ports 的□丁。□□表明以前□行□， 如果是第一次□行的□，那□□□只会下□ Ports Collection。

在 portsnap(8) 成功地完成一次 `fetch` 操作之后， 会将校□□的 Ports 套件和后□的□丁保存在本地。 首次□行 `portsnap` 之后， □必□使用 `extract` 安装下□的文件：

```
# portsnap extract
/usr/ports/.cvsignore
/usr/ports/CHANGES
/usr/ports/COPYRIGHT
/usr/ports/GIDs
/usr/ports/KNOBS
/usr/ports/LEGAL
/usr/ports/MOVED
/usr/ports/Makefile
/usr/ports/Mk/bsd.apache.mk
/usr/ports/Mk/bsd.autotools.mk
/usr/ports/Mk/bsd.cmake.mk
...
```

使用 `portsnap update` 命令更新已安装的 Ports：

```
# portsnap update
```

至此更新就完成了，然后便可以使用更新后的 Ports Collection 来安装或升□□用程序。

`fetch` 和 `extract` 或 `update` 可以作□□□的□作□行， 如下例所示：

```
# portsnap fetch update
```

□个命令将会下□最新版本的 Ports 并更新本地位于 /usr/ports 的拷□。

25.4. 更新系□附□的文□

除了基本系□和 Ports 套件之外， 文□也是 FreeBSD 操作系□的一个□成部分。 尽管□□是可以通□ FreeBSD 网站 来□□最新的 FreeBSD 文□， 一些用□的网□□接可能很慢， 甚至完全没有网□□接。 幸□的是，有很多方法可以用来更新随□行版本附□的 FreeBSD 文□的本地副本。

25.4.1. 使用 CVSup 来更新文□

FreeBSD 文□的源代□和安装版本都可以通□ CVSup 来以与基本系□ (参考 重新□□ "world") □似的方法来升□。□一□中将会介□：

- 如何安装□□文□所需的工具集， 用于从源代□来□□ FreeBSD 文□所需的那些工具。
- 如何使用 CVSup 将文□下□到 /usr/doc。
- 如何从源代□□□ FreeBSD 文□， 并将其安装到 /usr/shared/doc。
- □□文□的□程中支持的一些□□□□， 例如只□□某些□言的版本， 或只□□特定的□出格式。

25.4.2. 安装 CVSup 和文□工具集

从源代□□□ FreeBSD 文□需要大量的工具。 □些工具并不是 FreeBSD 基本系□的一部分， 因□□些工具需要占用大量的磁□空□， 而且并不是□所有 FreeBSD 用□都有用； 只有活□地撰写 FreeBSD 新文□， 或□常从源代□更新文□的用□才需要□些工具。

全部所需的工具， 均可通□ Ports 套件来安装。 textproc/docproj port 是由 FreeBSD 文□□□□□的方便安装和更新□些工具的主 port。

如果不需要 PostScript® 或 PDF 文□的□， 也可以考□安装 textproc/docproj-nojadetex port。 □套文□工具集包含除了 teTeX typesetting 引□之外的其他全部工具。 teTeX 是一个很大的工具集， 因此如果不需要 PDF □出的□， 排除它会□省很多□□和磁□空□。

如欲了解□于安装和使用 CVSup 的□一□信息， □参□ 使用 CVSup。

25.4.3. 更新文□源代□

CVSup 工具能□下□文□源代□的原始副本， □可使用 /usr/shared/examples/cvsup/doc-supfile 文件作□配置模板来修改。 在 doc-supfile 中的默□主机名是一个无效的占位主机名， 但 cvsup(1) 能□通□命令行来指定主机名， 因此文□源代□可以使用下面的命令从 CVSup 服□器□得：

```
# cvsup -h cvsup.FreeBSD.org -g -L 2 /usr/shared/examples/cvsup/doc-supfile
```

□□将 *cvsup.FreeBSD.org* 改□最近的 CVSup 服□器。 参□ CVSup 站点 □于□像站点的完整列表。

初始的文□源代□下□需要一些□□， □需要耐心等待它完成。

后□的更新可以用同□的命令来□行。 由于 CVSup 工具只下□上次□行之后所□生□的更新， 因此在首次□行之后再□行 CVSup □□是很快的。

在□出源代□之后， □可以使用□一□由 /usr/doc 目□中的 Makefile 支持的方法来更新它。 通□在

/etc/make.conf 中配置 SUP_UPDATE、 SUPHOST 和 DOCSUPFILE， 可以通□□行：

```
# cd /usr/doc
# make update
```

来完成更新。 典型的 /etc/make.conf 中的 make(1) □□是：

```
SUP_UPDATE= yes
SUPHOST?= cvsup.freebsd.org
DOCSUPFILE?= /usr/shared/examples/cvsup/doc-supfile
```

将 SUPHOST 和 DOCSUPFILE 的□使用 ?= 来指定的好□是使 make 命令行能□覆□□些□□。 在向 make.conf 中□加□□□推□□□做， 以避免在□□□反□修改□个文件。

25.4.4. 文□源代□中可□的□□

FreeBSD 文□的更新和□□系□支持一些方便只更新一部分文□， 或只□□特定格式及□文的□□。 □些□□可以在 /etc/make.conf 文件中配置， 也可以通□ make(1) 工具来指定。

□些□□包括：

DOC_LANG
: 准□□□和安装的□言列表。 例如， 指定□ en_US.ISO8859-1 表示只□□英文版的文□。

FORMATS
: 准□□出的格式列表。 目前， 系□支持 html、 html-split、 txt、 ps、 pdf、 和 rtf。

SUPHOST
: 用于用来更新的 CVSup 服□器的主机名。

DOCDIR
: 用于安装文□的目□。 默□□ /usr/shared/doc。

如欲了解 FreeBSD 中其他可供配置的全局 make □量， □参□ make.conf(5)。

□于 FreeBSD 文□□□系□的其他□情， □参□ FreeBSD 文□□□入□之新手必□部分。

25.4.5. 从源代□安装 FreeBSD 文□

在 /usr/doc 中下□了最新的文□源代□快照之后， 就可以□始□手□□文□了。

要更新全部 DOC_LANG 中定□的□言的文□， 需要□行下面的命令：

```
# cd /usr/doc
# make install clean
```

如果在 make.conf 中配置了正□的 DOCSUPFILE、 SUPHOST 和 SUP_UPDATE □□， □可以将更新源代□和安装一□完成：

```
# cd /usr/doc
# make update install clean
```

如果只需要更新某个特定□言的文□， 可以在 /usr/doc 中与之□□的目□中□行 make(1)：

```
# cd /usr/doc/en_US.ISO8859-1
# make update install clean
```

此外， □可以透□ make □量 FORMATS 来控制□出格式， 例如：

```
# cd /usr/doc
# make FORMATS='html html-split' install clean
```

25.4.6. 使用文□ Ports

在之前的章□中， 我□已展示了从源代□更新 FreeBSD 文□的方法。 基于源代□的更新的方法可能并不是□于所有的 FreeBSD 系□都可行有效。 □□文□源代□需要一大堆的工具， 文□工具□， □于 CVS 的一定了解和从□□中□出源代□， □有一些□□已□出代□的手工□□。 □一章□我□将介□一□使用 Ports 来更新已安装的 FreeBSD 文□：

- 下□并安装□□□好的文□快照， 而不用在本地□□任何部□ (□□便不再需要安装整个文□工具□了)。
- 下□文□的源代□并使用 ports 框架□□ (使得□出和□□的□□更容易些)。

□□□更新 FreeBSD 文□的方法都由一□ 文□工程□ <doceng@FreeBSD.org> □月更新的 文□ *ports* 提供支持。□些都列在了 FreeBSD Ports docs 虚□分□下面。

25.4.6.1. □□和安装文□ Ports

文□ ports 使用 ports 的□建框架使得文□的□□□得更加容易。 自□化了□出文□源代□， 配以□合的□境□置和命令行参数□行 make(1)， 它□使得安装或卸□文□□得就像安装 FreeBSD 其他 port 或二□制包那□容易。

□一个特性便是当在本地□□文□ ports □， 文□工具□ ports 会被列入依□□系， 并自□安装。

文□ ports 按以下的方式□□：

- 一个 "主 port"， 在 misc/freebsd-doc-en 下可以□到□个文□ port。 它是所有文□ ports 的基□。 在默□的情况下， 它只安装英文版文□。
- 一个 "合集 port"， misc/freebsd-doc-all， 它将□建并安装所有□言版本的所有文□。
- 最后是各□翻□的 "从属 port"， 比如： misc/freebsd-doc-hu 是匈牙利文版的文□。 所有□些都基于主 port 并会安装上□□□言的翻□文□。

以 root 用□身□□行如下的命令安装文□：

```
# cd /usr/ports/misc/freebsd-doc-en
# make install clean
```

□将会安装分章□的英文版本 HTML 格式文□ (与http://www.FreeBSD.org 上的相同) 到 /usr/local/shared/doc/freebsd 目□。

25.4.6.1.1. 常□的□□□□

文□ ports 有□多用来修改默□行□的□□。 以下是一段□要列表：

WITH_HTML

允□□建 HTML 格式： □□文□□一个□一的 HTML 文件。 此□文□的文件名□情况而定通常是 article.html，或 book.html， □外附加一些□片。

WITH_PDF

允□□建 Adobe® Portable Document Format， 可使用 Adobe® Acrobat Reader®， Ghostscript 或者其他的 PDF □□器□□。 此□文□的文件名□情况而定通常是 article.pdf 或 book.pdf。

DOCBASE

文□将被安装到的目□。默□□ /usr/local/shared/doc/freebsd。

□注意默□的目□与 CVSup 方法□所使用的目□不同。 □是因□我□正在安装的是一个 port， 而 ports 通常会被安装到 /usr/local 目□。 □可以指定 PREFIX □量覆□默□□。

□是一□□短的□于如何使用以上提到□量来安装 PDF 格式的匈牙利文□：

```
# cd /usr/ports/misc/freebsd-doc-hu
# make -DWITH_PDF DOCBASE=share/doc/freebsd/hu install clean
```

25.4.6.2. 使用文□ Packages

正如上文所述， 从 ports □建文□需要在本地安装一□文□工具□和一些□□所需的磁□空□。 当不□□源安装文□工具□， 或者从源代□□□需要太多的磁□空□□， 我□仍然可以安装□□□好的文□快照的 ports。

文□工程□ <doceng@FreeBSD.org> □个月都会制作 FreeBSD 文□快照的包。 □些二□制包可以通□包工具来操作， 比如 pkg_add(1)， pkg_delete(1)， 等等。

当使用二□制包□， 将安装所指定□言相□的 FreeBSD 文□的 所有 可用格式。

□例来□， 以下的命令将安装最新□□□的匈牙利□文□：

```
# pkg_add -r hu-freebsd-doc
```

二□制包使用了以下与□□ ports 名称不同的命名格式: lang-freebsd-doc。 □里的 *lang* 是□言代□的□短形式， 比如 hu 表示匈牙利□， 或者 zh_cn 表示□体中文。

25.4.6.3. 更新文□ Ports

任何用于更新 ports 的工具都可以被用来更新已安装的文□ port。 □例来□， 下面的命令通□ ports-mgmt/portupgrade 工具来更新已安装的匈牙利□文□二□制包。

```
# portupgrade -PP hu-freebsd-doc
```

25.5. 追踪□□分支

FreeBSD 有□个□□分支： FreeBSD-CURRENT 和 FreeBSD-STABLE。 □一章□将□□个分支作相□介□与如何保持□的系□更新。 我□将先介□ FreeBSD-CURRENT 然后是 FreeBSD-STABLE。

25.5.1. 使用最新的 FreeBSD CURRENT

□里再次□□， FreeBSD-CURRENT 是 FreeBSD □□的 "最前沿"。 FreeBSD-CURRENT 用□要有□高的技□能力， 并且□□有能力自已解决困□的系□□□。 如果□是个 FreeBSD 新手， 那□在安装之前最好三思。

25.5.1.1. FreeBSD-CURRENT 是什□？

FreeBSD-CURRENT 是 FreeBSD 的□展前沿。 包括了在下一个官方□行的□件中可能存在， 也可能不存在的□展、 □□性改□、 以及□渡性的机制。 尽管□多 FreeBSD □□者□天都会□□ FreeBSD-CURRENT 源代□， 但有□□些代□仍然会是不能□□的。 □然□些□□会很快解决， 但 FreeBSD-CURRENT 是□来破坏□是□正希望的功能性改善， 很可能完全取决于□□取源代□的的□机！

25.5.1.2. □需要 FreeBSD-CURRENT？

FreeBSD-CURRENT □合下□三□主要□趣□体：

1. FreeBSD 社区的成□： □□工作在源□□的某部分的人和□保持 "最新" □□□需求的人。
2. FreeBSD 社区的成□： □促使 FreeBSD-CURRENT 保持尽可能的健全而□花□□去解决□□的□□的□□者；以及那些□意提出□于 FreeBSD □化和□体方向的建□性建□并且提供□丁□□它□的人□。
3. 那些只是想□注或□了参考目的使用当前 (current) 源□的人□ (如，□了□□，而不是□行)。 □些人也偶□做做注□或□献代□。

25.5.1.3. FreeBSD-CURRENT 不是什□？

1. 追求最新功能， □听□里面有一些很酷的新功能， 并希望成□□周□的人中第一个□□它□的人。 尽管□能□因此首先了解到最新的功能， 但□也意味着在出□新的 bug □□也首当其冲。
2. 修□□漏的快捷方式。任何 FreeBSD-CURRENT 的既定版本在修□已知□漏的同□又可能会□生新的□漏。
3. 无所不在的"官方支持"。 我□尽最大努力在3个"合法的" FreeBSD-CURRENT □之一真□□人□提供□助，但是我□ 没有□□提供技□支持。 □并不是因□我□是那□不喜□□助人解困的无耻之徒 (如果我□是的□，就不会制作 FreeBSD 了)。 我□不能□天□□地回□上百的消息，而且 我□□□□展 FreeBSD！ 在改善 FreeBSD 和回□大量□于□□代□的□□之□如果要做个□□的□， □□人□会□□前者。

25.5.1.4. 使用 FreeBSD-CURRENT

1. 加入 FreeBSD-CURRENT □件列表 和 SVN src □ head/-current 分支的修□□息 列表。 □个不

□□是个好主意，而且很 重要。如果□不去 *FreeBSD-CURRENT* □件列表， □就不会看到人□所做的□于系□当前状□的□明， □□□就有可能在□人已□□□并解决了的一大堆□□面前□倒。 更重要的是□会□□一些重要的公告---□于□的系□安全可能是至□重要的。

SVN src □ head/-current 分支的修□□息 列表允□□看到□个□化的提交□□， 因□□些□□与其它相□信息是同□的。

要加入□些列表，或其它可能的列表，□□□ https://lists.freebsd.org ，并且点□□想□□的列□。 □于其它□□的□明那里有提供。 如果□有□趣追踪整个原代□□的□更□□， 我□建□□□□ SVN 整个 src □的修□□息 (除了 "user" 与 "projects") □件列表。

2. 从FreeBSD □像站点 □取源□。 □有□□方式□□： ..

   与称作 standard-supfile 的 supfile 一起使用 cvsup，□个可以从 /usr/shared/examples/cvsup得到。□是最被推□的方式，因□它允□□一次□取整个集合， 以后就只取更改□的部分。□多人从 `cron` □行 `cvsup`，以保持他□的源□自□更新。 □□要定制上□的 supfile □本，并且配置 cvsup 以□□□的□境。

   standard-supfile 例子是□追踪指定的 FreeBSD 安全分支而指定的， 而不是 FreeBSD-CURRENT。 □需要□□□个文件并把如下□行：

   ```
   *default release=cvs tag=RELENG_X_Y
   ```

   替□□：

   ```
   *default release=cvs tag=.
   ```

   可以参□手册中的 CVS Tags 章□□得更多可用 tag 的□□□明。

   使用工具 CTM。 如果□的□接性能不太好(高价□接或只能通□□子□件存取)， CTM 是个□□。 但□也□有争□并且常常得到到坏文件。因此很少使用它， □也注定了不能□期用它来工作。□于使用 9600 bps 或更快□接的人，我□推□使用 CVSup。

3. 如果□□取源□是用于□行，而不只是看看，那□就□取 整个 FreeBSD-CURRENT，不要□部分。□□做的原因是源□的大部分都依□于其他部分， 要是□□着只□□其中一部分的□，保□□会陷入麻□。

   在□□ FreeBSD-CURRENT 之前，□仔□□□ /usr/src 里的 Makefile 文件。 尽管是部分的升□□程，□至少也要首先安装新的内核和重建系□。□□ FreeBSD-CURRENT □件列表 □件列表和 /usr/src/UPDATING， 会□□在其它循序□□的□程中保持最新， □□于我□向下一个□行版□移是很有必要的。

4. □心一点！如果□正□行 FreeBSD-CURRENT， 我□很想知道□□于它的一些想法， 尤其是□于□漏修□或□□的建□。 非常□迎□有代□的建□！

25.5.2. 使用最新的 FreeBSD STABLE

25.5.2.1. FreeBSD-STABLE 是什□?

FreeBSD-STABLE 是我□的□展分支，我□的主要□行版就由此而来。 □个分支会以不同速度□化，并且假定□些是第一次□入 FreeBSD-CURRENT □行□□。然而，□ 仍然 是个□展中的分支，□意味着在一定的□

候，FreeBSD-STABLE 源□可能或不可能□足一些特殊的要求。 它只不□是□一个工程□展途径，并不是□端用□的□源。

25.5.2.2. □需要 FreeBSD-STABLE？

如果□有□趣追随 FreeBSD 的□□□程或□其做点□献， 尤其是和下一个 "非□□" 的 FreeBSD □行版有□□， □□□考□采用 FreeBSD-STABLE。

尽管安全更新也会□入 FreeBSD-STABLE 分支，但□并不 必□ 使用 FreeBSD-STABLE 来□到□□的目的。 □一个 FreeBSD 的安全公告都会解□如何修□受到影□的□行版中的□□ ，而因□安全原因而去采用一个□□分支□然可能会同□引入一些不希望的修改。

尽管我□尽力□保 FreeBSD-STABLE 分支在任何□候都能□正□□□和□行， 但没有人能□担保它在任何□候都□可以。 此外， 尽管代□在□入 FreeBSD-STABLE 之前都是在 FreeBSD-CURRENT 上完成□□， 但使用 FreeBSD-STABLE 的人要比使用 FreeBSD-CURRENT 的更多。 有□据□示， □角□□里的各□□□有些□候仍然会由于在 FreeBSD-CURRENT 不那□明□ 而在 FreeBSD-STABLE 暴露出来。

基于□些原因， 不 推□□盲目地追随 FreeBSD-STABLE， 并且， 在粗略地□□□代□之前不要更新任何生□服□器到 FreeBSD-STABLE 也非常重要。

如果□没有用于完成□些工作的□源， 我□推□□使用最新的 FreeBSD □行版， 并使用□行版提供的二□制更新机制来在□行版之□完成□移。

25.5.2.3. 使用FreeBSD-STABLE

1. 加入 FreeBSD-STABLE; □件列表 列表。□□随□了解可能出□在 FreeBSD-STABLE 里的"build 依□性"或其它需要特□注意的□□。 当□□□正在考□某些有争□的修□或更新□， 他□就会在□个□件列表里□表声明，□用□机会回□， 看他□□于提出的□化是否□有什□□□。

   加入相□的 SVN 列表来追踪□所□心的分支。比如，如果□在追踪 7-STABLE 分支，加入 svn-src-stable-7 列表。 □□□次□个分支上有改□的□候就能□□看到提交□□, □包括了修改可能引起的副作用之□的相□信息。

   要加入□些列表或其他可用的，□□ https://lists.freebsd.org 并点□□希望□□的列表。□于其它□□的□明可以在那里看到。 如果□有□趣追踪整个原代□□的□更□□， 我□建□□□□ SVN 整个 src □的修□□息 (除了 "user" 与 "projects") □件列表。

2. 如果□正安装一个新系□， 并希望它□行□月从 FreeBSD-STABLE □□的快照， □察看 Snapshots 网□以了解更多信息。 □外， 也可以从 □像站点 安装最新的 FreeBSD-STABLE □行版， 并按照其中的□明将系□更新到最新的 FreeBSD-STABLE 源代□。

   如果□已□在□行□早的 FreeBSD 版本， 并希望通□源代□方式升□， □可以通□ FreeBSD □像站点 来完成。□可以通□□□方式来□行： ..

   与称作 stable-supfile 的 supfile 一起使用 cvsup，□个可以从 /usr/shared/examples/cvsup 得到。□是最被推□的方式，因□它允□□一次□取整个集合， 以后就只取更改□的部分。□多人从 cron □行 cvsup，以保持他□的源□自□更新。 □□要定制上□的 supfile □本，并且配置 cvsup 以□□□的□境。 ..

   使用工具 CTM。 如果□的□接性能不太好(高价□接或只能通□□子□件存取)， CTM 是个□□。 但□也□有争□并且常常得到到坏文件。因此很少使用它， □也注定了不能□期用它来工作。□于使用 9600 bps 或更快□接的人，我□推□使用 CVSup。

3. 本□上□，如果□需要快速存取源□并且不□□通信□□的□，可以使用 cvsup 或 ftp。否□，就使用 CTM。 .

   在□□ FreeBSD-STABLE 之前，□仔□□□ /usr/src 里的 Makefile。 □至少□□安装一个新的内核并重建系□，首先做□升□□程的一部分。□□ FreeBSD-STABLE; □件列表 □件列表和 /usr/src/UPDATING， 可能□□在其它循序□□的□程中保持更新， □在我□向下一□行版□移□是很有必要的。

25.6. 同□□的源□

有□多方式通□互□网(或□子□件)与 FreeBSD □目源□特定□域或所有□域保持更新，主要依□于□的□趣。 我□提供的主要服□是匿名 CVS、 CVSup，和 CTM。

□然只更新源□□中的部分是可能的， 唯一被支持的更新□程是更新整个□、并且重□□用□区(如：在用□空□□行的所有程序，像 /bin 和 /sbin下□的)和内核源□。 只更新源□□中的部分，或只有内核，或只有用□区 (userland) 通常会出□□□。□些□□包括有□□□□、内核崩□ (kernel panics)、数据出□。

匿名 CVS 和 CVSup 使用 下拉*(pull)* 模式来更新源代□。 在 CVSup 中， 用□ (或者 cron 脚本) 会□用 cvsup 程序， 后者会同某一个 cvsupd 服□□行交互， 以更新□的文件。 □接到的更新是更新□刻最新的， 并且□只会收到那些需要的更新。 □可以很容易地限制更新的□□， 只更新那些□需要的文件。 服□器端会根据□手□已□有的文件即□地生成更新内容。 匿名 CVS 相□于 CVSup 而言要□□一些， 因□它只是□ CVS 的一□□展，□□可以从□程的 CVS 代□□得到更新。 CVSup 相□而言， 要比 匿名 CVS 更有效率， 然而后者却更容易使用。

□一□方法是 CTM。 □□方法并不能将□手□的代□与中央代□□中的版本□行比□， 也不能下□它□。 在主 CTM 服□器上□行的脚本会□天□行多次， □次□行都能□自□地□□所有文件自上次□行以来所□生的□化， 如果□□有文件□生了□□， 就会□□、 □上一个序列号， 并□行便于使用□子□件□行□送的□□操作 (其中只包括可打印的 ASCII 字符)。 一旦接收到， □些"CTM deltas"就会被□送□ ctm_rmail(1) 工具---可以自□□行解□、校□和□用□些□化到用□的□制的源□里。 □个□程比 CVSup 更□有效， 而且更少占用我□的服□器□源，因□它不□□采用下拉*(pull)* 模式，□采用 上推*(push)* 模式。

当然， □□做也会□来一些不便。 如果□不□意□除了□的□□包的部分内容， CVSup 会□□到并□□重建破坏的部分。 CTM 是不会□□做的， 如果□□除了□的源□□中的某部分(并已不能恢□)， 那□□就必□从破坏□ (从最新的CVS "base delta") □始，使用 CTM 或 匿名 CVS □行重建，□□□除坏的数据并再同□。

25.7. 重新□□ "world"

只要□根据一定版本的 FreeBSD (FreeBSD-STABLE、FreeBSD-CURRENT 等等)， 已□同□了□本地的源□□，那□□就可以使用□些源□□来重建系□。

做好□□

无需□□在行□ 之前 □□整个系□是多□的重要。 尽管重新□□系□是 (如果□按照文□的指示做的□)一件很容易完成的工作， 但出□也是在所□免的， □外， □人在源□里面引入的□□也可能造成系□无法引□。

□□信自己已□做□□□， 并且在手□有恢□□□或可以引□的光□。 □可能永□也不会用到它，但安全第一嘛！

□□恰当的□件列表

FreeBSD-STABLE 和 FreeBSD-CURRENT 分支自然是 □展中的。□ FreeBSD 做□献的都是人，偶□也会犯□□。

有□□些□□没什□危害，只是引起□的系□生成新的□断警告。 有□是□□性的，并□致□的系□不能□□或破坏□的文件系□ (甚至更糟)。

如果出□了□似的□□， □一封"小心(heads up)"帖到相□的□件列表里， □清□□的本□以及受影□的系□。在□□解决后，再□封"解除(all clear)"声明。

如果使用 FreeBSD-STABLE 或 FreeBSD-CURRENT 而又不□□ FreeBSD-STABLE; □件列表 和 FreeBSD-CURRENT □件列表 各自的□件列表， 那□□是自□麻□。

不要使用 `make world`

□多□早的文□推□使用 `make world` 来完成□□工作。 □□做会跳□一些必要的□□， 因此只有在□知道自己在做什□的□候才可以□□做。 几乎所有的情况下 `make world` 都是不□□做的事情，□□□使用□里描述的方法。

25.7.1. 更新系□的□□方法

在更新系□□， 一定要首先□看 /usr/src/UPDATING 文件， 以便了解在 buildworld 之前需要□行的操作，然后按照下面列出的□□□行操作：

□些更新□□假定□使用的是包含旧□□器、 内核以及用□□工具及配置的旧版 FreeBSD。 我□使用 "world" 来表示系□中的核心□行文件、 函数□和程序文件。 □□器是 "world" 的一部分， 但有其特殊性。

此外， 我□□假定□已□□得了□新版本操作系□的源代□。 如果□正更新的系□中的源代□也是旧版系□所附□的，□□需要参□ 同□□的源□ 来把代□同□到□新的版本。

从源代□更新系□， 有□会比初看上去的□候更麻□一些， □一方面， FreeBSD 的□□人□有□会不得不修改推□的更新□□， 特□是当出□了一些无法避免的依□□系的□候。 □一□余下的部分， 将介□目前推□的更新□□背后的原理。

成功的更新操作必□解决下面的□些□□：

- 旧的□□器可能无法□□新的内核。 (□一方面， 旧的□□器很可能有 bug。) 因此， 新的内核□□以新的□□器□□。 更具体地□， 新的□□器□在新内核□始□□之前已□完成了□□□□。 □注意， 新的□□器并不一定需要在□□新内核之前 安装 到系□中。
- 新的 world 有可能依□一些新的内核特性。 因此， 新内核必□在新的 world 之前安装。

□□个□□就是□什□我□将在后面的章□中介□的， 需要按照 `buildworld`、 `buildkernel`、 `installkernel`、 `installworld` 的□序来更新系□的原因。 □并不是□需要遵守推□的更新操作的全部原因， 除了□□个最重要的理由之外， □有一些并不那□□而易□的原因：

- 旧的 world 可能无法配合新的内核正常工作， 因此， □在安装完新内核之后， □尽快将 world 也随之更新。
- 有些配置文件的□□必□在安装新的 world 之前完成， 而□一些配置文件的□□□有可能□致旧 world 工作不正常。 因此， 通常而言会需要□次不同的配置文件更新□□。

- 多数情况下， 更新□□只会替□或□加文件； □言之， □有的旧文件并不会被□除。 有□， □可能会□致一些其他□□。 因此， 有□安装操作会指明， 必□在某些操作之前手工□除一些文件。□些在未来可能会被自□化， 也可能不会自□化。

由于有□些考□， 因此一般情况下我□建□使用下列更新□□。 □注意，具体的更新操作中可能会需要一些附加的□□， 但核心的□程□□是不会□易□生□化的：

1. `make buildworld`

   □□操作会□□新的□□器， 以及少量相□工具， 并在随后使用新的□□器来□□ world。 □□的□果会存放在 /usr/obj。

2. `make buildkernel`

   与旧式的、 使用 config(8) 和 make(1) 的方法不同， □□做法会使用存放于 /usr/obj 中的 新的 □□器。□□做法使得□免去了由于□□器与内核源代□不一致□致的□□。

3. `make installkernel`

   安装新的内核及其模□， 使系□能□以更新后的内核□□。

4. 重□系□并□入□用□模式。

   □用□模式使得更新正在□行的□件可能□致的□□□到最少。 此外， 它也使配合新内核□行旧 world 可能出□的□□□到最少。

5. `mergemaster -p`

   □□操作会□行完成安装新的 world 所需的配置文件更新操作。 例如， 它可能会在系□的密□数据□中添加新的用□□或用□。 □些操作通常在上次更新之后□加了新的用□□或特殊系□用□之后是需要的， 因□`installworld` □□操作会需要□些用□或□才能□利完成。

6. `make installworld`

   从 /usr/obj 中□制 world。 □□操作之后， □在□上的系□， 包括内核和 world 就都是新的了。

7. `mergemaster`

   更新余下的配置文件， 因□□的 world 已□更新完成了。

8. 重□系□。

   □□操作将加在新的内核， 以及新的 world 和更新□的配置文件。

注意， 如果□正从同一 FreeBSD 版本分支升□， 例如， 从 7.0 到 7.1， □上述□程可能没有那□必要， 因□□不太可能遇到□重的□□器、 内核源代□、 用□□程序源代□或配置文件不匹配的情形。 旧式的 `make world` 然后再□□新内核的升□方法， 很可能有机会能□正常□作而完成升□工作。

但是， 在大版本升□的□程中， 不按照前面所介□的操作来□行升□□， 便很可能遇到一些□□。

此外， □需要注意的是， 有些□候升□的□程中 (例如从 4.*X* 到 5.0) 可能会需要一些□外的□□ (例如在 installworld 之前更名或□除一些文件)。 □仔□□□ /usr/src/UPDATING □个文件， 特□是它的□尾部分所介

□的推□的升□操作□序。

由于□□人□□□不可能完全避免一些不匹配方面的□□， □个□程一直在演化□程中。 不□幸□的是， 目前推□的□个升□□□， □□能□在很□一段□□内不需要做任何□整。

□□一下， 目前推□的从源代□升□ FreeBSD 的方法是：

```
# cd /usr/src
# make buildworld
# make buildkernel
# make installkernel
# shutdown -r now
```

有□， 可能需要□外地□行一次 `mergemaster -p` 才能□完成 `buildworld` □□。 □些要求， 会在 UPDATING 中□行描述。 一般而言， □可以□□地跳□□一□， 只要□行的不是大跨度的 FreeBSD 版本升□。

在 `installkernel` 成功完成之后， □需要引□到□用□模式 (□例而言， 可以在加□器提示后□入 `boot -s`)。接下来□行：

```
# adjkerntz -i
# mount -a -t ufs
# mergemaster -p
# cd /usr/src
# make installworld
# mergemaster
# reboot
```

□□□一□的□明

前面所□出的， 只是□助□□始工作的□要□明。 要清楚地理解□一□， 特□是如果打算自行定制内核配置， 就□□□下面的内容。

25.7.2. □□ /usr/src/UPDATING

在□做其它事之前，□□□ /usr/src/UPDATING (或在□的源□里的等效的文件)。 □个文件要包含有□于□可能遇到的□□的重要信息， 或指定了□可能使用到的命令的□行□序。如果 UPDATING 与□□里□到相矛盾，那就先依据 UPDATING。

正如先前所述，□□ UPDATING 并不能替代□□正□的□件列表。□都是互□的，并不彼此排斥。

25.7.3. □□ /etc/make.conf

□□ /usr/shared/examples/etc/make.conf 以及 /etc/make.conf。 第一个文件包含了一些默□的定□ - 它□中的□大多数都注□掉了。 □了在重新□□系□□能□使用它□， □把□些□□加入到 /etc/make.conf。 □注意在 /etc/make.conf 中的任何□置同□也会影□□次□行 `make` 的□果， 因此□置一些□合自己系□的□□是一个好□□。

一般的用□通常会从 /usr/shared/examples/etc/make.conf □制 CFLAGS 和 NO_PROFILE □□的□置到 /etc/make.conf 中并令它□生效。

□考□其他的一些□□ (例如 COPTFLAGS、 NOPORTDOCS 等等)， 看看是否合用。

25.7.4. 更新 /etc 里的文件

/etc 目□包含有除了□的系□□□□□行的脚本外大部分的系□配置信息。 有些脚本随 FreeBSD 的版本而不同。

有些配置文件在天天□行的系□里也是要使用到的。尤其是 /etc/group。

偶□， 作□安装□程的一部分， make installworld 会要求事先□建某些特定的用□或□。 在□行升□□， 它□可能并不存在。 □会□升□造成□□。 有□， make buildworld 会□□它□是否已□存在。

最近就有个□□的例子， 当□新□了 smmsp 用□。 当用□□□完成安装操作□， 在 mtree(8) □□建立 /var/spool/clientmqueue □失□了。

解决□法是通□使用 -p □□以□建前 (pre-buildworld) 模式□行 mergemaster(8)。 □表示只□比那些□于成功□行 buildworld 或 installworld 起□□作用的文件。 在第一次□□做□， 如果使用的是早期的不支持 -p 的 mergemaster 版本的□， 使用源□中的新版本即可。

```
# cd /usr/src/usr.sbin/mergemaster
# ./mergemaster.sh -p
```

如果□是个偏□狂 (paranoid)， □可以□□□的系□看看□个文件属于□已更名或□除了的那个□。

```
# find / -group GID -print
```

将□示所有 *GID* □ (可以是□名也可以是数字地□ ID)所有的文件。

25.7.5. 改□□用□模式

□可能想在□用□模式下□□系□。 除了□更快□理事情□然有好□外， 重装系□将触及□多重要的系□文件，包括所有□准系□二□制文件、□文件、包含 (include) 文件等等。 在正□行的系□ (尤其是在有活□的用□的□候)中更改□些文件是自□□□。

□一□模式是在多用□模式下□□系□，然后□□到□用□模式下安装。 如果□喜□□□方式，只需在建立 (build)完成后才□行下□的□□。 □推□□□到□用□模式下直到□必□ installkernel 或 installworld。

从□行的系□里，以超□用□方式□行：

```
# shutdown now
```

□□就会□□到□用□模式。

除此之外， 也可以重□系□， 并在□□菜□□□□ "single user"(□用□) □□。 □□系□将以□用□模式□□。 接着， 在 shell 提示符□□行：

```
# fsck -p
# mount -u /
# mount -a -t ufs
# swapon -a
```

□会□□文件系□，重新将 / 以□/写模式挂接， 参考 /etc/fstab 挂接其它所有的 UFS 文件系□，然后□用交□区。

如果□的 CMOS □□是□置□本地□□，而不是 GMT (如果 date(1) 命令□出不能□示正□的□□和地区也□有其事)， □可能也需要□行下□的命令：

```
# adjkerntz -i
```

□□可以□定□正□的本地□区□置-不□□做， □以后可能会□到一些□□。

25.7.6. □除 /usr/obj

随着重新□建系□的□行， □□□果会放到 (默□情况下) /usr/obj 下。 □些目□会映射到 /usr/src。

通□□除□个目□， 可以加速 make buildworld 的□程， 并避免相互依□□系等□□的□□。

/usr/obj 中的某些文件可能□置了不可改□□ (□情参□ chflags(1))， 需要首先去掉□些□志。

```
# cd /usr/obj
# chflags -R noschg *
# rm -rf *
```

25.7.7. 重新□□基本系□

25.7.7.1. 保存□出

建□把□行 make(1) 后得到的□出存成一个文件。 如果什□地方出了□，□就会有个□□信息的□□。 尽管□□不能□□分析□里出了□， 但如果□把□的□□□到某个□件列表里就能□助其他的人。

□□做最□□的□法是使用 script(1) 命令，同是□上参数指定存放□出的文件名。 □□在重建系□之前立即□□做，然后在□程完成□□入 exit。

```
# script /var/tmp/mw.out
Script started, output file is /var/tmp/mw.out
# make TARGET
… compile, compile, compile …
# exit
Script done, …
```

如果□□□做，就 不要 把文件存到 /tmp 里□。下次□□□，□个目□就会被清除掉。 存放的最好地方是 /var/tmp (如上个□例)或 root 的主目□。

25.7.7.2. □□基本系□

□必□在/usr/src目□里□：

```
# cd /usr/src
```

(当然，除非□的源□是在其它地方，真是□□的□更□成那个目□就行了)。

使用 make(1) 命令重建系□。□个命令会从 Makefile (描述□成 FreeBSD 的程序□□□□被重建， 以什□□的□序建立等等) 里□取指令。

□入的一般命令格式如下：

```
# make -x -DVARIABLE target
```

□个例子里，-x 是会□□□ make(1) 的一个□□。□看 make(1) 手册有□可用的□□例子。

-D_VARIABLE_ □□一个□量□ Makefile。□些□量控制了 Makefile 的行□。□些同 /etc/make.conf □置的□量一□，只是提供了□一□□置它□的方法。

```
# make -DNO_PROFILE target
```

是□一□指定不被建立 (built) 的先定□ (profiled libraries) 的方式，□同 /etc/make.conf 里的

```
NO_PROFILE=    true     #    避免□□性能分析□
```

一起使用。

目□ *(target)* 告□ make(1) 什□□做。□个 Makefile 定□了一定数量不同的"目□ (targets)"， 然后□□□的目□就决定了什□会□生。

有些目□列在 Makefile 里的，但并不意味着□要□行。相反，建立□程 (build process) 利用它□把重建系□的一些必要的□□分割成几个子□□。

大部分的□□不需要向 make(1) □□参数，因此□的命令看起来可能象□□：

```
# make target
```

此□ *target* 表示的是若干□□□□。 多数情况下， 第一个 target 都□□是 buildworld。

正如名字所暗示的，buildworld 在 /usr/obj 下□建立了一个全新的□， 然后使用□一个 target， installworld 在当前的机器里安装它。

将□些□□分□有□个□点。 首先， 它允□□安全地完成建立 (build)， 而不□正在□行的系□的□件□生影□。 □建□程是"自主的 (self hosted)"。 因□□□， □可以安全地在以多用□模式□行的机器里□行 buildworld，而不用当心不良影□。 但是依然推□□在□用□模式□□行 installworld。

第二，允□□使用 NFS 挂接 (NFS mounts) 升□□网□里的多台□算机。如果□有三台 A、B 和 C 想□行升□，在A □行 make buildworld 和 make installworld。 然后将 A 上的 /usr/src 和 /usr/obj 通□ NFS 挂接到 B 和 C 上， 接下来， 只需在 B 和 C 上使用 make installworld 来安装□建的□果就可以了。

尽管 world target 仍然存在，□烈建□□不要用它。

□行

```
# make buildworld
```

我□提供了一个□□性的功能， 可以在□建□程中□ make 指定 -j 参数， 令其在□建□程中同□□□多个并□的□程。□于多 CPU 的机器而言， □□做有助于□□其性能。 不□， 由于□□□程中的瓶□主要是在 IO 而不是 CPU 上，因此它也会□□ CPU 的机器□来好□。

□典型的□ CPU 机器， 可以使用：

```
# make -j4 buildworld
```

□□， make(1) 会最多同□□□ 4 个□程。 从□到□件列表中的□□看， □□做能□来最佳的性能。

如果□使用的机器有多□ CPU， 并且配置了 SMP 的内核， 也可以□□看 6 到 10 的数□， 并□察是否能□来□建性能上的改善。

25.7.7.3. 耗□

□□基本系□所需的□□会受到很多因素的影□， 不□， □新的机器□□都能在一□个小□之内完成 FreeBSD-STABLE 源代□的□建， 而无□任何技巧或捷径。 完成 FreeBSD-CURRENT 源代□的□□， □通常需要更□一些的□□。

25.7.8. □□和安装新内核

要充分利用□的新系□，□□□重新□□内核。 □是很有必要的，因□特定的内存□□已□□生了改□，像 ps(1) 和 top(1) □□的程序会不能工作， 除非内核同源□□的版本是一□的。

最□□、最安全的方式是 build 并安装一个基于 GENERIC 的内核。□然 GENERIC 可能没有□合□的系□的所有必要的□□， 但它包括了□□□的系□到□用□模式所必需的内容。 □是个不□的□□新系□是否工作正常的□□。在从 GENERIC □□、核□系□可以工作后， □就可以建立 (build) 一个基于□的正常内核配置文件的新的内核了。

在 FreeBSD 中， 首先完成 build world 然后再□□新内核非常重要。

如果□想建立一个定制内核，而且已□有了配置文件， 只需象□□使用 KERNCONF=MYKERNEL：

```
# cd /usr/src
# make buildkernel KERNCONF=MYKERNEL
# make installkernel KERNCONF=MYKERNEL
```

注意，如果□已把 内核安全□□(kern.securelevel) □高到了 1 以上，而且□□置了 noschg 或相似的□□到了

□的内核二□制里□，□可能会□□□□到□用□模式里使用 installkernel 是很有必要的。 如果□没有□置它，□□□也能毫无□□地在多用□模式□行□□个命令。 □参考 init(8) 以了解更多□于 内核安全□(kern.securelevel) 的信息；□看 chflags(1) 了解更多□于不同文件□□的信息。

25.7.9. 重□到□用□模式

□□□□用□模式□□新内核。照改□□用□模式□的□明去做。

25.7.10. 安装□□好的新系□

□□在□使用 installworld 来安装新的系□二□制。

□行

```
# cd /usr/src
# make installworld
```

如果在 make buildworld 的命令行指定了□量，□就必□在 make installworld 命令行里指定同□的□量。 □于其它的□□并不是必需的，如，-j 就不能同 installworld 一起使用。

□例，□□行了：

```
# make -DNO_PROFILE buildworld
```

□就必□使用：

```
# make -DNO_PROFILE installworld
```

来安装□果，否□就要□着安装先定 (profiled) 的在 make buildworld □段没有建立 (built) 的二□制文件。

25.7.11. 不是由 make installworld 更新的更新文件

重新□□整个系□不会使用新的或改□的配置文件更新某些目□ (尤其像 /etc、/var 和 /usr)

更新□些文件最□□的方式就是使用 mergemaster(8)，手工去做也是可以的，只要□□意。 不管□□□□一□，一定□得□□ /etc 以防出□。

25.7.11.1. mergemaster

mergemaster(8) 工具是个 Bourne 脚本，用于□□ /etc 和 /usr/src/etc 源□□里□的配置文件的不同点。□是保持系□配置文件同源□□里的一起更新的推□方式。

在提示符里□□地□入 mergemaster 就可以□始，并□看它的□始□程。mergemaster 会建立一个□□的根 (root)□境，在 / 下， 放置各□系□配置文件。□些文件然后同当前安装到□系□里的□行比□。 此□，不同的文件会以 diff(1) 格式□行□示，使用 + 符号□□□加或修改的行，- □□将完全□除的行或将被替

□成新行。□看 diff(1) 手册可以得到更多□于 diff(1) □法和文件不同点□□□示的信息。

mergemaster(8) 会□□□示□个文件的不同□， □□□就可以□□是□除新文件 (相□□□文件)， 是以未改状□安装□□文件，是以当前安装的文件合并□□文件， □是再看一次 diff(1) □果。

"□□□除□□文件"将使 mergemaster(8) 知道我□希望保留我□当前的文件不改，并□除新的。 并不推□□个□□，除非□没有更改当前文件的理由。任何□候在 mergemaster(8) 提示符里□入 `?`，□就会得到□助。 如果□□跳□文件，将在其它文件□理完后再次□行。

"□□安装未修改□□文件"将会使新文件替□当前的。 □大部分未改的文件，□是个最好的□□。

"□□合并文件"将□□打□一个文本□□器， 里□是□个文件的内容。□□在就可以一□合并它□， 一□在屏幕里□看，同□从□者中□取部分生成最□文件。 当□个文件一起比□□，`l` □会□□左□的内容， `r` 会□□右□的。最□的□出是由□个部分□成的一个文件， 用它就可以安装了。□个□□通常用于用□修改了□置的文件。

"□□再次□看 diff(1) □果"将会在提供□□□之前， □示文件的不同□，就象 mergemaster(8) 所做的一□。

在 mergemaster(8) 完成了□系□文件的□理后， □会得到其它的□□。mergemaster(8) 可能会□□是否要重建密□文件， 并在最后提示□是否要□除余下的□□文件。

25.7.11.2. 手□更新

如果想要手工更新，但不要只是从 /usr/src/etc 把文件□制到 /etc 就了事。有些文件是必□先"安装"的。 □是因□ /usr/src/etc 目□并 不是 想像的那□是 /etc 目□的一个□制。事□上，有些是文件是 /etc 有的，而 /usr/src/etc 里□没有。

如果□使用 mergemaster(8) (作□推□)，□可以向前跳到 下一□。

手工做最□□的方式是安装□些文件到一个新的目□，完成后再来□□不同□。

□□□已有的 */etc*

□然，理□上，没有什□会自□□□个目□， 事情□是做□操□当一点。□制已有 /etc 到一个安全的地方，如：

```
# cp -Rp /etc /etc.old
```

`-R` 完成□□□制 (□者注：即可以□制目□以下的所有内容)，`-p` 保留文件的□□、所属等等。

□需要建立一个虚目□ (a dummy set of directories) 来安装新的 /etc 和其它文件。 /var/tmp/root 是个不□的□□， 除此之外，□有一些子目□是需要的。

```
# mkdir /var/tmp/root
# cd /usr/src/etc
# make DESTDIR=/var/tmp/root distrib-dirs distribution
```

□□就建好了需要的目□□□，然后安装文件。在 /var/tmp/root 下建立的大部分子目□是空的， 而且要□除掉。最□□的方式是：

```
# cd /var/tmp/root
# find -d . -type d | xargs rmdir 2>/dev/null
```

□□会□除所有的空目□。(□准的□□信息被重定向到了 /dev/null，以防止□于非空目□的警告。)

/var/tmp/root □在包含了□放在 / 下某个位置的所有文件。 □□在必□仔□□□一个文件，□□它□与□已有的文件有多大不同。

注意，有些已□安装在 /var/tmp/root 下的文件有个"."在□□。在写的□候，像□□唯一的文件是 /var/tmp/root/ 和 /var/tmp/root/root/ 里 shell □□文件，尽管可能有其它的(依□于□什□□候□取□个)。 □信使用 `ls -a` 可以看到它□。

最□□的方式是使用 diff(1) 去比□□个文件：

```
# diff /etc/shells /var/tmp/root/etc/shells
```

□会□示出 /etc/shells 文件和新的 /var/tmp/root/etc/shells 文件的不同□。 用□些来决定是合并□已做的□化□是□制□的旧文件□来。

使用日□ *(Time Stamp)* 命名新的 *Root(*根*)*目□*(/var/tmp/root)*，□□□可以□松地比□□个版本的不同

□繁重建系□意味着必□□繁更新 /etc，而□可能会有点□□。

在合并到 /etc 的文件里， 最新更改的□可以做个□制，由此加快□个(指更新)□程。 下□就□出了一个□□做的主意。

1. 像平常一□建立系□ (Make the world)。当□想更新 /etc 和其它目□里， □目□目□一个含有当前日期的名字。假如□是 1998 年 2 月 14 日做的，□可以□行下□的：

   ```
   # mkdir /var/tmp/root-19980214
   # cd /usr/src/etc
   # make DESTDIR=/var/tmp/root-19980214 \
       distrib-dirs distribution
   ```

2. 如上□列出的，从□个目□合并□化。

   在□完成后，不要 □除 /var/tmp/root-19980214 目□。

3. 在□下□了最新版的源□并改□后，□行第一□。 □□将得到一个新的目□，可能叫做 /var/tmp/root-19980221 (如果等了一周做的升□)。

4. □□在能看到□个目□□的不同了---在隔周的□□里使用 diff(1) 建立□□ diff □生的不同：

   ```
   # cd /var/tmp
   # diff -r root-19980214 root-19980221
   ```

   一般情况下，□□□□的不同□比 /var/tmp/root-19980221/etc 和 /etc 之□的不同要小很多。 因□不同点更小，也就更容易把□些□化移到□的 /etc 目□里□。

5. □□在可以□除早先的□个 /var/tmp/root-* 目□：

   ```
   # rm -rf /var/tmp/root-19980214
   ```

6. □次□需要合并□些□化到 /etc 里，就重□□个流程。

□可以使用 date(1) 自□□生目□的名称：

```
# mkdir /var/tmp/root-`date "+%Y%m%d"`
```

25.7.12. 重□

□在完成了。在□□□所有内容都放置正□后， □可以重□系□了。只是□□的 shutdown(8) 可以□□做：

```
# shutdown -r now
```

25.7.13. □束

恭喜！□□在成功升□了□的 FreeBSD 系□。

如果□有□微的□□，可以□易地重建系□的□定部分。 例如，在部分升□或合并 /etc □，□不小心□除了 /etc/magic，file(1) 命令就会停止工作。□□情况下，□行下□□行修□：

```
# cd /usr/src/usr.bin/file
# make all install
```

25.7.14. □□

25.7.14.1. □个□化□都□要重建系□□？

□个不好□，因□要看□化的情况。如，如果□□□行了 CVSup，并得到下□更新的文件：

```
src/games/cribbage/instr.c
src/games/sail/pl_main.c
src/release/sysinstall/config.c
src/release/sysinstall/media.c
src/shared/mk/bsd.port.mk
```

□就不必重建整个系□。□只需到相□的子目□里□行 `make all install`，□此而已。 但是，如果有重大□化，如 src/lib/libc/stdlib， 那□□就要重建系□或至少静□□接的那些部分 (除了□□加的部分都是静□□接的)。

在□天后，就是□的事了。要是□□□个星期重建一下系□的□， □可能会高□。或者□可能只想重做改□□的部分，□信□能□出所有依□□系。

当然，所有□些依□于□想升□的□率，和□是否想跟踪 FreeBSD-STABLE 或 FreeBSD-CURRENT。

25.7.14.2. 我的□□失□，并伴随有□多 11信号 11 (或其它的数字信息) 号□□。是□□回事呀？

□个通常表示硬件□□。 (重)建系□是个□□□□系□硬件的有效地方式， 并且常常□生内存□□。 □些正好表示它□自已做□□□器□奇地死于收到的奇怪信息。

一个□信的指示器是如果重新□始 make，并且整个□程中会死在不同的点上。

□于□□情况，□没有什□可做的，除了更□机器里的部件，看是□一个坏了。

25.7.14.3. 我完成后可以□除 /usr/obj □？

□短地□，可以。

/usr/obj 包含了所有在□□□段生成的目□文件。通常， 在 `make buildworld` □程中第一□之一就是□除□个目□重新□始。 □□情况下，在□完成后，保留 /usr/obj 没有多大意□，□可□放一大堆磁□空□(目前是 2 GB 左右)。

不□， 如果□很了解整个□程， 也可以□ make buildworld 跳□□一□。 □会□后□的□□□程□行得更快， 因□大部分的源□都不必再□行□□了。 □□做的□面效果是它可能会触□一些由于敏感的依□□系□致的□□， □些□□会□致□□以奇怪的方式出□并失□。 □在 FreeBSD □件列表里□常引起沸□， 当有人抱怨他□ build 失□□，并没意□到□是因□自已是想抄近路。

25.7.14.4. 中断的 build 可以被恢□□？

依□于□在□□到□□之前整个□程□行了多□。

一般而言 (当然□并不是硬性□定)， make buildworld 的□程中将会首先□建新版的基本□建工具 (例如 gcc(1)，以及 make(1)) 和系□□。 随后会安装□些工具和□。 □些新版本的工具和□在随后将被用于重新□□和□接它□本身。 整个系□ (□在包括了常□的用□程序， 例如 ls(1) 或 grep(1)) 会同新版的系□文件一起被重新□建。

如果□正□于最后一个□段， 并且了解它 (因□□已□看□了所保存的□出) □可以 (相当安全地) 做：

```
… □□修□ …
# cd /usr/src
# make -DNO_CLEAN all
```

□□就不会取消先前的 make buildworld 所做的工作了。

在"make buildworld"的□出中如果看到如下信息：

```
--------------------------------------------------------------
Building everything..
--------------------------------------------------------------
```

出□在 make buildworld 的□出中， □□□做□□不会有什□□□。

如果没有看到□□的信息， 或者□不□定， □从□□始□建将是万无一失的做法。

25.7.14.5. 我□□加快建立系□的速度？

- 以□用□模式□行
- 把 /usr/src 和 /usr/obj 目□放到不同磁□里的独立文件系□里。如果可能，□些磁□在不同的磁□控制器里。
- 更好的，是把□些文件系□放置到多个使用 ccd(4) (□接磁□□□器—concatenated disk driver)□□的磁□里。
- □掉 profiling (在 /etc/make.conf 里□置 "NO_PROFILE=true")。□差不多用不了它。
- 在 /etc/make.conf 里也□ CFLAGS □置上 -O -pipe。 最佳□化 -O2 会更慢，而且 -O 和 -O2 之□的□化差□基本上可以忽略。 -pipe □□□器使用管道而不用□□文件□行通信， □□可以□少磁□存取 (以内存作□代价)。
- □□ -jn □□□ make(1) 以便并□□行多个□程。 □□就不会考□□的是否是□个或多个□理器机器。
- 存放 /usr/src 的文件系□可以使用 noatime □□来挂接 (或重新挂接)。 □□会防止文件系□□□文件的存取□□。□可能并不需要□些信息。

  ```
  # mount -u -o noatime /usr/src
  ```

□个例子里假定 /usr/src 是在它自已的文件系□里。如果不是 (例如假□它是 /usr 的部分)，那□□就需要那个文件系□挂接点， 而不是 /usr/src。

- 存放 /usr/obj 的文件系□可以使用 async □□被挂接 (或重新挂接)。 □□做将□用□□写□。 □句□□，□□用程序而言写会立即完成， 而数据□延□几秒才会写到□里。 □□做能□成批地写下数据， 从而□大地改善性能。

注意， □个□□会使□的文件系□□得脆弱。 使用□个□□会提高在□源断掉或机器非正常重□□， 文件系□□入不可恢□状□的概率。

如果在□个文件系□里 /usr/obj 是很□□的，□不是□□。如果□有其它有价□的数据在同一个文件系□， 那□在□使用□个□□□前，□□□□一下。

```
# mount -u -o async /usr/obj
```

同上，如果 /usr/obj 不在自已的文件系□里，使用相□挂接点的名字把它从例子里□替□掉。

25.7.15. 如果出□了□□我□□□□？

□□□信□的□境没有先前 build 留下的残余。□点□□□。

```
# chflags -R noschg /usr/obj/usr
# rm -rf /usr/obj/usr
# cd /usr/src
# make cleandir
# make cleandir
```

不□，make cleandir 真的要□行□次。

然后重新□始整个□程，使用 make buildworld □始。

如果□□有□□，就把□□和 uname -a 的□出□送到 FreeBSD 一般□□□件列表 □件列表。准□回答其它□于□的□置的□□！

25.8. □除□□的文件、 目□和函数□

在 FreeBSD 的□□□程中， 随□可能会出□一些文件或其内容□□的情况。□□情况有可能是由于其功能在其它地方□□了， 函数□的版本号□加， 或完全从基本系□中□去， 等等。一般的□□和更新□程并不会□去□些旧的文件、 函数□或目□， 在更新系□之后， □及□予以清理。 清理的好□是□些文件不会再□□占用存□ (以及□□) 空□， □外， 如果旧的函数□或文件中存在安全或可□性□□， □也□更新到新的函数□， 以避免安全□患或崩□情形的□生。 □□的文件、 目□和函数□会列在 /usr/src/ObsoleteFiles.inc 中。 接下来将介□在系□更新□程中如何□去□些□□的文件。

我□假定□已□按照 更新系□的□□方法 介□的□□完成了更新操作。 在 make installworld 和 mergemaster

命令完成之后，□□使用下面的命令□□系□中是否存在□□的文件或□：

```
# cd /usr/src
# make check-old
```

如果有□□的文件，□可以用下面的命令来□除：

```
# make delete-old
```

参□ /usr/src/Makefile 可以了解其他 target 的功用。

在□除文件□，系□会□□□个文件都□出提示。□可以跳□□些提示，并□系□自□完成□除操作，方法是使用 make □量 BATCH_DELETE_OLD_FILES，具体做法如下：

```
# make -DBATCH_DELETE_OLD_FILES delete-old
```

□也可以用 yes 命令和管道来□到□似的目的：

```
# yes|make delete-old
```

□去□□的文件，有可能会破坏□有的依□□些文件的□用程序。□于旧的函数□来□，□□□□出□的可能性更大。□大多数情况下，□□重新□□使用旧□的所有的程序、port 或函数□之后再□行 make delete-old-libs。

在 Ports Collection 中提供了一些□□□□□接□依□□系的工具，例如 sysutils/libchk 和 sysutils/bsdadminscripts。

□□的□□□接□可能会与新□冲突，□致□似□□的警告消息：

```
/usr/bin/ld: warning: libz.so.4, needed by /usr/local/lib/libtiff.so, may conflict
with libz.so.5
/usr/bin/ld: warning: librpcsvc.so.4, needed by /usr/local/lib/libXext.so, may
conflict with librpcsvc.so.5
```

要解决□□的□□，需要□□安装□个□的 port：

```
# pkg_info -W  /usr/local/lib/libtiff.so
/usr/local/lib/libtiff.so was installed by package tiff-3.9.4
# pkg_info -W /usr/local/lib/libXext.so
/usr/local/lib/libXext.so was installed by package libXext-1.1.1,1
```

接着卸□、重新□□并安装 port。□可以使用 ports-mgmt/portmaster 或 ports-mgmt/portupgrade 工具来自□完成□些操作。在□□所有的 port 都重新□□，并且不再使用旧□以后，□就可以用下面的命令来

□除它□了：

```
# make delete-old-libs
```

25.9. 跟踪多台机器

如果□有多台机器想跟踪同□的源□□， 那□□它□都下□源□并重建所有□西，看起有点浪□□源： 磁□空□、网□□□以及 CPU 周期。 解决的□法是□一台机器□理大部分的工作，而其它的机器通□ NFS 挂接 (mount) □些工作。□部分列□了一□□□做的方法。

25.9.1. 准□

首先，□定一批机器，□行的二□制代□是同一套---我□称作 □建集群 *(build set)*。□台机器可以使用不同的定制内核， 但它□□行的是相同的用□区二□制文件(userland binaries)。 从□批机器中□□一台机器做□ □建机器*(build machine)*。 □将是用于□建(build)系□和内核的机器。想像一下，它□□是一台快速的机器， 有足□的空余的 CPU 来□行`make buildworld`。 □也想要□一台机器做□ □□机器*(test machine)*， □个将用于□件的更新生成□品之前□他□□行□□。□个 必□ 是一台□能提供的平□也可使用的机器。 它可以是"□建机器"，但没□个必要。

在□个"□建集群"里的所有机器需要从同一台机器、 同一个点上挂接 /usr/obj 和 /usr/src。理想地， 它□在"□建机器"上的□个不同的□□器里， 但是在那台机器上可以□行 NFS 挂接。如果□有多个"□建集群"， /usr/src □□在某个"□建机器"上， 而在其它机器上□行 NFS 挂接。

最后，□□"□建集群"里所有机器上的 /etc/make.conf 和 /etc/src.conf 与"□建机器"里的相同。 □意味着"□建机器"必□□建部分基本系□用于 "□建集群"里所有机器的安装。同□， □台"□建机器"要有它自已的内核名字，使用 /etc/make.conf 里的 `KERNCONF` □行□置，并且□台"□建机器"□□把它□列在`KERNCONF` 里，同□把自已的内核列在最前。 "□建机器"的 /usr/src/sys/arch/conf 里一定要有□台机器的内核配置文件，如果它想□建它□的内核的□。

25.9.2. 基本系□

既然所有的妥当了，就准□□建所有的□西。如□□基本系□中描述的一□在"□建机器"上□建内核和系□， 但是什□也不安装。在□建□束后，□到"□□机器"上， 安装□□□建的内核。如果□台机器通□ NFS 挂接了 /usr/src 和 /usr/obj， 在□重□到□用□模式里，□需要□□网□然后挂接他□。 最□□的方式是□□到多用□模式下，然后□行`shutdown now` □到□用□模式。一旦□入，□就可以安装新的内核和系□，并□行 `mergemaster`，就像平常一□。完成后， 重□返回到一般多用□模式操作□台机器。

在□□信所有在 "□□机"里都工作正常后， 就使用相同的□程在 "□建集群"里的其它机器里安装新的□件。

25.9.3. Ports

□似的想法是使用 ports □。 第一个□□的□□是从同一台□算机上挂接 /usr/ports 到 "□建集群" 里的全部□算机。然后正□□置 /etc/make.conf 共享 distfiles。□□把 `DISTDIR` □置到一个共享的目□里， 那里可以被任何一个`root` 用□写入， 并且是由□的 NFS 挂接映射的。 □置□一台机器的 `WRKDIRPREFIX` 到一个本地□建 (build) 目□。最后，如果□要□建和□布包 (packages)，那□□□□置 `PACKAGES` 到一个□似于 `DISTDIR` 的目□。

Chapter 26. DTrace

26.1. 概述

DTrace，也称□□□跟踪，是由 Sun™ □□的一个用来在生□和□□性生□系□上□出系□瓶□的工具。在任何情况下它都不是一个□□工具， 而是一个□□系□分析□□出性能及其他□□的工具。

DTrace 是个特□好的分析工具，□有大量的□助□断系□□□的特性。 □可以使用□先写好的脚本利用它的功能。用□也可以通□使用 DTrace D □言□建他□自己定制的分析工具， 以□足特定的需求。

在□□了□一章□之后，□将了解：

- DTrace 是什□，它提供了些□些特性。
- DTrace 在 Solaris™ 与 FreeBSD 上的□□的差□。
- 如何在 FreeBSD 上□□和使用 DTrace。

在□□□一章□之前，□□□了解：

- 了解 UNIX® 和 FreeBSD 的基本知□ (UNIX 基□)。
- 熟悉基本的内核配置/□□ (配置FreeBSD的内核).
- 熟悉 FreeBSD 有□的安全知□ (安全)。
- 了解如何□取和重新□□ FreeBSD 源代□ (更新与升□ FreeBSD)。

□□特性目前仍被□□是□□性的。 有些□□功能性缺失，□有一些可能□无法□行。最□，□个特性会□合用于生□，届□□篇文□也会做些□当的修改。

26.2. □□上的差□

□然 FreeBSD 上的 DTrace 与 Solaris™ 上的非常相似， 在□□深入之前我□需要□明一下存在的差□。 用□首先会注意到的便是 FreeBSD 上的 DTrace 需要明□地被□用。 DTrace 相□的内核□□和模□必□□□后才能正常工作。 □后我□会作□□介□。

有一个 `DDB_CTF` 内核□□用来□□从内核与内核模□加□ `CTF` 数据。 CTF 是 Solaris™ Compact C Type Format 封装了□似于 DWARF 和 venerable stabs □化的□□信息。CTF 数据是由 `ctfconvert` 和 `ctfmerge` 工具加入二□制文件的。`ctfconvert` 工具分析由□□器生成的 DWARFELF □□ section， `ctfmerge` 合并目□文件的 CTFELF section 到可□行文件或共享□。更多□于在□用 FreeBSD 内核上□用此□的□□内容即将完成。

比起 Solaris™， FreeBSD 有几个不同提供器。 最□得注意的是 `dtmalloc` 提供器， 可以□□根据□型追踪 FreeBSD 内核中的 `malloc()`。

只有 `root` 可以使用 FreeBSD 上的 DTrace。 □是由系□安全上的差□造成的，Solaris™ 提供了一些 FreeBSD 上□未□□的低□的安全□□。 同□， /dev/dtrace/dtrace 也被□格的限制□□供 `root` 用□□□。

最后，DTrace □ Sun™ CDDL □可下□布的□件。随 FreeBSD □行的 `Common Development and Distribution License` 可以在□□ /usr/src/cddl/contrib/opensolaris/OPENSOLARIS.LICENSE 或者通□ http://www.opensolaris.org/os/licensing □看在□版本。

□个□可表示□有 DTrace □□的 FreeBSD 内核仍□ BSD □可； 然而， 以二□制□布模□， 或者加□二□制模□□需遵守 CDDL。

26.3. □用 DTrace 支持

在内核配置文件中加入以下几行来□□□ DTrace 的支持：

```
options         KDTRACE_HOOKS
options         DDB_CTF
```

使用 AMD64 架□的需要在内核配置文件中加入如下□行：

```
options         KDTRACE_FRAME
```

此□□提供了□ FBT 特性的支持。 DTrace 可以在没有此□□的情况下正常工作， 但是函数□界跟踪便会有所限制。

所有的源代□都必□重新使用 CTF □□□□安装。重新□□ FreeBSD 源代□可以通□以下的命令完成：

```
# cd /usr/src

# make WITH_CTF=1 kernel
```

系□需要重新□□。

在重新□□和新内核□入内存之后，需要添加 Korn shell 的支持。因□ DTrace 工具包有一些工具是由 `ksh` 写的。安装 shells/ksh93。 同□也可以通□ shells/pdksh 或者 shells/mksh 使用□些工具。

最后是□得最新的 DTrace 工具包。 当前版本可以通□下面的□接□到 http://www.opensolaris.org/os/community/dtrace/dtracetoolkit/。
□个工具包含有一个安装机制，尽管如此，并不需要安装便可使用它□。

26.4. 使用 DTrace

在使用 DTrace 的功能之前，DTrace □□必□存在。 使用如下的命令装□此□□：

```
# kldload dtraceall
```

DTrace 支持□在□□可以使用了。 管理□□在可以使用如下的命令□看所有的探□器：

```
# dtrace -l | more
```

所有的□出都□□□ `more` 工具， 因□它□会很快超出屏幕的□示区域。此□，DTrace □□被□□是能□正常工作的了。

□在是□考察工具包的□候了。

工具包是□□写好的一堆脚本，与 DTrace 一起□行来收集系□信息。 有脚本用来□□已打□的文件，内存，CPU 使用率和□多□西。使用如下的命令解□脚本：

```
# gunzip -c DTraceToolkit* | tar xvf -
```

使用 cd 命令切□到那个目□， 并修改所有文件的可□行□限，把那些名字小写的文件□限改□ 755。

所有□些脚本都需要修改它□的内容。那些指向 /usr/bin/ksh 需要修改成 /usr/local/bin/ksh，□外使用 /usr/bin/sh 需要□更□ /bin/sh，最后□有使用 /usr/bin/perl 的需要□更□ /usr/local/bin/perl。

此刻□需□慎提醒一下□者 FreeBSD 的 DTrace 支持仍是 不完整的 和 □□性 的。□些脚本中的大多数都无法□行，因□它□□于□□ Solaris™ 或者使用了目前□不支持的探□器。

在撰写□篇文章的□候，DTrace 工具包中只有□个脚本在 FreeBSD 上是完全支持的： hotkernel 和 procsystime 脚本。□□个脚本便是我□下一部分将要探□的：

hotkernel 被□□成□明□个函数占用了内核□□。 正常□行的□，它将生成□似以下的□出：

```
# ./hotkernel
Sampling... Hit Ctrl-C to end.
```

系□管理□必□使用 Ctrl + C □合□停止□个□程。 □接着中止之后，脚本便会一□内核函数与□定□□的列表， 使用□量排序□出：

```
kernel`_thread_lock_flags                                2   0.0%
0xc1097063                                               2   0.0%
kernel`sched_userret                                     2   0.0%
kernel`kern_select                                       2   0.0%
kernel`generic_copyin                                    3   0.0%
kernel`_mtx_assert                                       3   0.0%
kernel`vm_fault                                          3   0.0%
kernel`sopoll_generic                                    3   0.0%
kernel`fixup_filename                                    4   0.0%
kernel`_isitmyx                                          4   0.0%
kernel`find_instance                                     4   0.0%
kernel`_mtx_unlock_flags                                 5   0.0%
kernel`syscall                                           5   0.0%
kernel`DELAY                                             5   0.0%
0xc108a253                                               6   0.0%
kernel`witness_lock                                      7   0.0%
kernel`read_aux_data_no_wait                             7   0.0%
kernel`Xint0x80_syscall                                  7   0.0%
kernel`witness_checkorder                                7   0.0%
kernel`sse2_pagezero                                     8   0.0%
kernel`strncmp                                           9   0.0%
kernel`spinlock_exit                                    10   0.0%
kernel`_mtx_lock_flags                                  11   0.0%
kernel`witness_unlock                                   15   0.0%
kernel`sched_idletd                                    137   0.3%
0xc10981a5                                           42139  99.3%
```

□个脚本也能与内核模□一起工作。要使用此特性， 用 -m □志□行脚本：

```
# ./hotkernel -m
Sampling... Hit Ctrl-C to end.
^C
MODULE                                                COUNT   PCNT
0xc107882e                                                1   0.0%
0xc10e6aa4                                                1   0.0%
0xc1076983                                                1   0.0%
0xc109708a                                                1   0.0%
0xc1075a5d                                                1   0.0%
0xc1077325                                                1   0.0%
0xc108a245                                                1   0.0%
0xc107730d                                                1   0.0%
0xc1097063                                                2   0.0%
0xc108a253                                               73   0.0%
kernel                                                  874   0.4%
0xc10981a5                                           213781  99.6%
```

procsystime 脚本捕捉并打印□定 PID 的系□□用□□。 在下面的例子中，新生成了一个 /bin/csh □例。procsystime □行后□等待在新□行的 csh 上□入一些命令。 □是□□的□果：

```
# ./procsystime -n csh
Tracing... Hit Ctrl-C to end...
^C

Elapsed Times for processes csh,

         SYSCALL          TIME (ns)
          getpid               6131
       sigreturn               8121
           close              19127
           fcntl              19959
             dup              26955
         setpgid              28070
            stat              31899
       setitimer              40938
           wait4              62717
       sigaction              67372
     sigprocmask             119091
    gettimeofday             183710
           write             263242
          execve             492547
           ioctl             770073
           vfork            3258923
      sigsuspend            6985124
            read         3988049784
```

正如□示的那□，`read` 系□□用似乎使用了最多的□秒□位□□， `getpid()` 系□□用使用了最少的□□。

26.5. D □言

DTrace 工具包包括了很多由 DTrace 特殊□言写成的脚本。 在 Sun™ 的文□中称□□□言□ "D □言"， 它与 C++ 非常□似。□此□言更深入的□□□超出了□篇文章的□□。 更多相□的□□可以在 http://wikis.sun.com/display/DTrace/Documentation □到。

部分 IV: 网□通□

FreeBSD 是目前以高性能网□服□□目的而部署□□最广的操作系□之一。 □□□些□□的章□包括：

- 串口通□
- PPP 和以太网上的 PPP
- □子□件
- □行网□服□
- 防火□
- 其他□□网□□□

□些章□主要供□在需要□参考。 不必按特定的□序来□□它□， 此外， □□始在网□中使用 FreeBSD 之前也不需要先把它□都□完。

Chapter 27. 串口通

27.1. 概述

UNIX® 一直都是支持串口通的。事上， 早期的 UNIX® 系就是利用串口来入和出数据的。 那常的 "端" 包括一个秒10个字符的串口打印机和， 在些已生了很大的化。 章将介一些利用 FreeBSD 行串口通的方法。

完章，将了解到：

- 如何通端接到的FreeBSD系。
- 如何使用modem号到程主机。
- 如何允程用通modem登到的系。
- 如何从串口控制台引的系。

章之前，当了解：

- 如何配置和安装一个新的内核 (配置FreeBSD的内核)。
- 理解 UNIX® 的限和程 (UNIX 基)。
- 准打算在 FreeBSD 中使用的串口 (modem 或多口)的技参考手册。

27.2. 介

从 FreeBSD 8.0 始， 用于串口的点从 /dev/cuadN 改了 /dev/cuauN； 从 /dev/ttydN 改了 /dev/ttyuN。 FreeBSD 7.X 用需要根据情况文中的例子行必要的整。

27.2.1.

bps

秒位- 数据的速度

DTE

数据端 - 如的算机

DCE

数据通 - 如的modem

RS-232

用于硬件串口通的EIA准

当通数据速度的候，不会使用 "baud"。Baud指气准率，它已使用了很， 而 "bps" (bits per second) 才是正使用的 (至少它不会打那些争的家)。

27.2.2. □□和端口

要将 modem 或□端与□的 FreeBSD 系□相□， □的□算机需要一个串口， 以及用于□接串口□□所需的□□。 如果□比□熟悉硬件及所需要的□□， □可以跳□□□。

27.2.2.1. □□

串口□□有□多不同的□□。 最常□的□□□型是 null-modem □□和□准 ("直□") RS-232 □□。 □的硬件□明□中会介□□使用的□□□□。

27.2.2.1.1. Null-modem□□

null-modem □□会直接□送某些信号， 如 "Signal Ground" (信号地)， 但□其他信号□行交□。 例如， "Transmitted Data" (数据□送) 引脚是□到□一端 "Received Data" (数据接收) 引脚的。

也可以自行制作 null-modem □□□□端使用 (例如， □了品□的要求)。 下面的表格展示了 RS-232C 信号， 以及 DB-25 □接器上的引脚。 注意， □准也要求一根直通引脚 1 到引脚 1 的 保□地 *(Protective Ground)* □， 但□通常都被省掉。 某些□端在只有引脚 2、 3 和 7 的□候， 就已□能□正常使用了， 而其他一些，□需要下面例子中所展示的不同的配置。

表 *9. DB-25 to DB-25 Null-Modem Cable*

信号	引脚 #		引脚 #	信号
SG	7	□接到	7	SG
TD	2	□接到	3	RD
RD	3	□接到	2	TD
RTS	4	□接到	5	CTS
CTS	5	□接到	4	RTS
DTR	20	□接到	6	DSR
DTR	20	□接到	8	DCD
DSR	6	□接到	20	DTR
DCD	8	□接到	20	DTR

□里□有□□目前比□流行的其他接□方式。

表 *10. DB-9* 到 *DB-9 Null-Modem* □□

信号	引脚 #		引脚 #	信号
RD	2	接到	3	TD
TD	3	接到	2	RD
DTR	4	接到	6	DSR
DTR	4	接到	1	DCD
SG	5	接到	5	SG
DSR	6	接到	4	DTR

信号	引脚 #		引脚 #	信号
DCD	1	接到	4	DTR
RTS	7	接到	8	CTS
CTS	8	接到	7	RTS

表 *11. DB-9* 到 *DB-25 Null-Modem* □□

信号	引脚 #		引脚 #	信号
RD	2	DB-9 到 DB-25 Null-Modem □□	2	TD
TD	3	接到	3	RD
DTR	4	接到	6	DSR
DTR	4	接到	8	DCD
SG	5	接到	7	SG
DSR	6	接到	20	DTR
DCD	1	接到	20	DTR
RTS	7	接到	5	CTS
CTS	8	接到	4	RTS

当某一段□接器上的一个引脚需要□接到□端的一□引脚□， 通常是将那一□引脚使用一短□□接， 而使用□□接到□一端的那个引脚。

上面的□□似乎更□流行。 在其他□□中 (在 *RS-232 Made Easy* □本□中□行了□□介□) □是 SG 接 SG， TD 接 RD、RTS 和 CTS 接 DCD、 DTR 接 DSR， 反之亦然。

27.2.2.1.2. □准RS-232C□□

□准的串口□□会直接□送所有 RS-232C 信号。 也就是□， 一□的 "Transmitted Data" 引脚， 会直接接到□一□的 "Transmitted Data" 引脚。 □包括将□制解□器接到□的 FreeBSD 系□上的那□□□， 同□也□用于某些型号的□端。

27.2.2.2. 端口

串口是FreeBSD主机与□端□□数据的□□。 □□描述了端口的□□和它□在 FreeBSD 上是如何□址的。

27.2.2.2.1. 端口的□□

有好几□串口。 在采□或制作□□之前， □□□□它能□□合□的□端以及 FreeBSD 系□。

□大多数□端都提供 DB-25 端口。 个人□算机， 也包括□行 FreeBSD 的 PC 机， 通常会有 DB-25 或 DB-9 口。 如果□的 PC 上有多□口串口□， □可能有 RJ-12 或 RJ-45 口。

□参□□硬件的文□以了解所用接口的□格。 此外， □也可以通□□察外□来了解所用的端口。

27.2.2.2.2. 端口名称Port Names

在FreeBSD中，□可以通□ /dev 目□中的一个□□来□□□个串口。有□□不同的□□：

- 呼入端口的名字是 /dev/ttyuN， 其中 *N* 是端口的□号， 从零□始□数。 一般来□， □使用呼入端口作□□端。 呼入端口要求数据□使用□波□□ (DCD) 信号来工作。
- 呼出端口的名字是 /dev/cuauN。 通常并不使用呼出端口作□□端， 而只用于□制解□器。 如果串口□□或□端不支持□波□□信号， □可能必□要使用呼出端口。

如果□已□□接一个□端到第一个串口 (在 MS-DOS® 上是COM1)， □可以使用 /dev/ttyu0 来作□□端。 如果它是在第二个串口 (COM2)， 那就是 /dev/ttyu1，等等。

27.2.3. 内核配置

FreeBSD默□支持4个串口。 在MS-DOS®下，□些是 COM1， COM2， COM3， 和 COM4。 FreeBSD 目前支持 "dumb" 多口串口□，如 BocaBoard 1008 和 2016， 以及□多 Digiboard 和 Stallion Technologies 制造的智能多接口□。 不□， 默□的内核只会□□□准的COM端口。

要看看□的内核是否支持□的串口，只要在内核□□□□看一下□□信息， 或使用 /sbin/dmesg 命令重新□□内核□□信息。 特□的，□□以sio字符□□的信息。

如果想只察看包含 sio 一□的消息， 可以使用下面的命令：

```
# /sbin/dmesg | grep 'sio'
```

例如，在一个□有4个串口的系□上，□些是串口特定的内核□□信息：

```
sio0 at 0x3f8-0x3ff irq 4 on isa
sio0: type 16550A
sio1 at 0x2f8-0x2ff irq 3 on isa
sio1: type 16550A
sio2 at 0x3e8-0x3ef irq 5 on isa
sio2: type 16550A
sio3 at 0x2e8-0x2ef irq 9 on isa
sio3: type 16550A
```

如果内核未能□出所有的串口， 可能需要通□修改 /boot/device.hints 文件来□行一些配置。 此外，也可以注□或完全□除掉□没有的□□。

□参□ sio(4) □机手册来了解□于串口， 以及多□口□配置的□一□□□。 如果□正使用一个在不同版本的 FreeBSD 上的文件□□必小心， 因□□□参数和□法□生了□化。

□里端口 IO_COM1 代替了 0x3f8，端口 IO_COM2 代替了 0x2f8，端口 IO_COM3 代替了 0x3e8，端口 IO_COM4 代替了 0x2e8，□些都是各自端口相□的端口地址。 中断4，3，5，9都是□常用的中断。也要注意有些正常的串口可能 无法 在一些ISA□□的PC上共享中断 (多□口板□有板□的□子□□，允□在板上所有 16550A 的□□共享一个或□个中断□求)。

27.2.4. □□特殊文件

在内核中， 大多数□□都是通□ "□□特殊文件" 来□□的， □些文件一般位于 /dev 目□中。 sio 是通□ /dev/ttyuN (呼入) 和 /dev/cuauN (呼出) □□来□□的。 此外， FreeBSD 也提供了初始化□□ （/dev/ttyuN.init 和 /dev/cuauN.init） 以及□□□ （/dev/ttyuN.lock 和 /dev/cuauN.lock）。 初始化□□用于在打□端口□初始化其通□参数， 例如使用 RTS/CTS 信号□行流控制的□制解□器的 crtscts。 □□□□用于在端口上提供一个□□志， 防止用□或程序改□特定的参数； □参□ termios(4)、 sio(4)， 以及 stty(1) 的□机手册， 以了解□于□端配置、 □和初始化□□， 以及配置□端参数的□□信息。

27.2.5. 串口配置

ttyuN （或 cuauN） □□是□将要打□的□用程序的一般□□。 当□程打□某个□□□， 它将有一个□端 I/O □置的默□配置。 □可以在命令行看看□些□置：

```
# stty -a -f /dev/ttyu1
```

当□修改了□个□□的□置，□个□置会生效，除非□□被□□。 当它被重新打□□，它将回到默□□置。 要修改默□□置，□可以打□和□整 "初始状□" □□的□置。例如， 要□ttyu5 打□ CLOCAL 模式，8位通□和默□的 XON/XOFF 流控制， □入：

```
# stty -f /dev/ttyu5.init clocal cs8 ixon ixoff
```

串口□□的系□□初始化， 是由 /etc/rc.d/serial 来控制的。 □个文件会影□串口□□的默□□置。

□了防止□用程序修改某些□置， □修改 "lock state"(□状□) □□。 例如， 要把 ttyu5 的速率□定□ 57600 bps，□入：

```
# stty -f /dev/ttyu5.lock 57600
```

□在，一个打□ttyu5 和□法改□端口速度的□用程序将被固定在57600bit/s。很自然地， □需要□定初始状□，然后用root□□□定状□□□的写入功能。

很□然，□□□只□ root 用□可以初始化或□定□□的状□。

27.3. □端

从 FreeBSD 8.0 □始， 用于串口的□□□点从 /dev/cuadN 改□了 /dev/cuauN； 从 /dev/ttydN 改□了 /dev/ttyuN。 FreeBSD 7.X 用□需要根据□□情况□□□文□中的例子□行必要的□整。

当□在□算机控制台或是在一个□接的网□上□， □端提供了一个方便和低成本的□□ FreeBSD 系□的方法。□□描述了如何在 FreeBSD 上使用□端。

27.3.1. □端的用法和□型

早期的 UNIX® 系□没有控制台。 人□通□将□端□接到□算机的串口来登□和使用程序。 它很像用 modem 和一些□端□件来□号□入一个□程的系□， 只能□行文本的工作。

今天的 PC 已□可以使用高□量的□形了， 但与今天的其他UNIX®操作系□一□，建立一个登□会□的能力仍然存在。 通□使用一个□端□接到一个没有使用的串口， □就能登□和□行任何文本程序或在 X □□系□中□行一个 `xterm` □口程序。

□于商□用□，□可以把任何□端□接到 FreeBSD 系□， 然后把它□放在□工的□面上。 □于家庭用□，□可以使用一台比□老的 IBM PC 或 Macintosh □行一个□端□接到一台□行 FreeBSD 的高性能机器上。

□于FreeBSD，有三□□端：

- □□端
- 充当□端的PC
- X □端

下面一小□将描述□一□□端。

27.3.1.1. □□端

□□端需要□□的好几□硬件，□□通□串口□□接到□算机。 它□被叫做 "□" 是因□它□只能□用来□示，□送和接收文本。 □不能在它上面□行任何程序。

有好几百□□□端，包括Digital Equipment Corporation 的VT-100和Wyse的WY-75。只有几□可以在FreeBSD上工作。 一些高端的□端可以□示□形，但只有某些□件包可以使用□些高□特性。

□□端被广泛用于那些不需要□形□用的工作中。

27.3.1.2. 充当□端的PC

假如 □□端 的功能□限于□示、 □送和接收文本的□， 那□□然任何一台□置的个人□算机， 都完全能□□任□□端的工作。 因此□需要的是合□的□□， 以及一些在□台□算机上□行的 □端□真 □件。

□□配置在家庭中□用十分广泛。 例如， 如果□的□人正忙于在□的 FreeBSD 系□的控制台上工作□，□就可以将一台功能□弱的□算机挂在□个 FreeBSD 系□上来同□完成一些文本界面的工作。

在 FreeBSD 的基本系□中至少有□个能用于□行串口□接的工具： cu(1) 和 tip(1)。

如果要从□行 FreeBSD 的□算机上通□串口□接到□一系□， 可以使用：

```
# cu -l 串口□□
```

此□ "串口□□" 表示□□算机上某个串口□□的□□名。 /dev/cuauN。

此□的 "N" 表示串口的□号。

□注意在 FreeBSD 中□□的□号是从零而非一□始的 (□一点与□一些系□， 如基于 MS-DOS® 的系□不同)。 因此， 在基于 MS-DOS® 系□中的 COM1 在 FreeBSD 中通常叫做 /dev/cuau0。

其他一些人可能喜□使用□一些来自 Ports 套件的程序。 Ports 中提供了几个与 cu(1) 和 tip(1) □似的工具， 例如 comms/minicom。

27.3.1.3. X □端

X□端是最□□的□端系□。它□通常需要使用以太网来□接。 它□能□示任何X□用程序。

我□介□X□端只是□了感□趣。然而， □章_不会_□及X□端的安装，配置或使用。

27.3.2. 配置

□□描述了□在一个□端上□用一个登□会□□， 需要在 FreeBSD 系□上□行的配置。 假□已□配置好了内核来支持串口， 就可以直接□始□接了。

在 FreeBSD 引□□程 中曾□提到， init □程依□于系□□□□所有的□理控制和初始化。 通□ init 来□行的一些任□将先□取 /etc/ttys文件， 然后在可用的□端上□用一个 getty □程。 getty □程可用来□□一个登□名和□□ login程序。

然而，要□□的FreeBSD系□配置□端，□需要以 root 身□□行下面的□□：

1. 如果它不在那里， □需要□串口在 /dev 目□下添加一行□□到 /etc/ttys。
2. 指定 /usr/libexec/getty 在端口上□行， 然后从 /etc/gettytab 文件指定□当的 *getty* □型。
3. 指定默□的□端□型。
4. □置端口□ "on"。
5. □定端口是否□ "secure"。
6. 迫使init 重新□取 /etc/ttys文件。

作□可□的□□，□可以通□在 /etc/gettytab 中建立一个□□，在第2□□建一个定制的 *getty* □型来使用。□章不会介□如何做。 □可以参考 gettytab(5) 和 getty(8) 的□机手册了解更多信息。

27.3.2.1. 添加一个□□到/etc/ttys

/etc/ttys文件列出了□ FreeBSD系□上允□登□的所有端口。 例如， 第一个虚□控制台 ttyv0 在□个文件中有一个□□。 □可以使用□个□□登□□控制台。 □个文件也包含其他虚□控制台的□□，串口，和□ ttys □端。 □于一个硬□□的□端， 只要列出串口的 /dev □□而不需要 /dev 部分 (例如， /dev/ttyv0 可以被列□ ttyv0)。

默□的 FreeBSD 安装包括一个支持最初四个串口 ttyu0 到 ttyu3 的/etc/ttys 文件。 如果□从那些端口中某一个使用□端，□不需要添加□一个□□。

例 32. 在 /etc/ttys 中□加□端□□

假□我□□接□个□端□系□： 一个 Wyse-50 和一个老的□行 Procomm □端□件模□一个 VT-100 □端的286IBM PC。 在 /etc/ttys 文件中的相□的□□是□□的：

```
ttyu1  "/usr/libexec/getty std.38400"  wy50  on  insecure
ttyu5   "/usr/libexec/getty std.19200"  vt100  on  insecure
```

- 第一部分指定了□端指定文件的名称， 它可以在 /dev中□到。
- 第二部分是在□行□行的命令，通常是 getty(8)。 getty 初始化然后打□一行，□置速度， 用□名的命令和□行登□程序。getty 程序在它的命令行接收一个参数 (可□)， *getty* □型。 一个 *getty* □型会在□端行描述一个特征， 像波特率和奇偶校□。 getty 程序从 /etc/gettytab 文件□取□些特征。文件/etc/gettytab 包含了□多老的和新的□端行□□。 在很多例子中，□□文本 std 的□□将用硬□□□端来工作。 □些□□忽略了奇偶性。 □是一个从110到115200 bit/s的 std □□。 当然，□可以添加□自己的□□到□个文件。 gettytab 的□机手册提供了更多的信息。当在/etc/ttys中□置 *getty* □型的□候， □信在□端上的通□□置匹配。 在我□的例子中， Wyse-50 不使用奇偶性， 用38400 bit/s来□接。286 PC不使用奇偶性，用19200bit/s来□接。
- 第三部分是通常□接到那个tty行的□端□型。□于□号端口， unknown 或 dialup 通常被用在□个地方。□于硬□□的□端，□端□型不会改□， 所以□可以从termcap数据□文件中放一个真正的□端□型。在我□的例子中， Wyse-50 使用真正的□端□型， 而□行 Procomm 的286 PC将被□置成在VT-100上的模□。
- 如果端口被□用，可以指定第四个部分。在第二部分， 把它放在□儿将□行初始化□程来□□程序 getty。如果□在□部分□延， 将没有getty，在端口上因此就没有登□。
- 最后部分被用来指定端口是否安全。 □□一个安全的端口意味着□信任它允□用 root □□从那个端口登□。 不安全的端口不允□ root登□。 在一个不安全的端口上， 用□必□用无特□的□□登□， 然后使用 su 或一个相似的机制来□得超□用□的□限。

27.3.2.2. 重新□取/etc/ttys来□制init

□/etc/ttys文件做一个必要的修改后，□必□□送一个 SIGHUP 信号□初始化□程来迫使它重新□取配置文件，例如：

```
# kill -HUP 1
```

init □是系□□行□的第一个□程，因此它□是PID 1。

如果能□正□□置，所有的□□都是□当的，□端将可以□用了， 然后一个 getty □程将在□个□端□行， □将在□的□端上看到登□命令行。

27.3.3. □的□接可能出□的□□

即使□小心翼翼地注意□□，□仍然可能会在□置□端□出□。 □有一个有□□□和解决□法的列表：

27.3.3.1. 没有登□命令出□：

□定□端被嵌入和打□了。如果把一台个人□算机充当一个□端， □信□端模□□件□行在正□的串口上。

□信□□被□固地□接在□端和 FreeBSD □算机上。 □信用了正□的□□。

□定□端和 FreeBSD 的□□速度和奇偶□置已□一致了。 如果□有一个□像□示□端，□信□比度已□□□好了。如果它是一个可打印的□端，□信□□和墨水已□就□了。

□定一个 getty □程正在□行和服□□端。 例如， 可以用ps 命令得到□行 getty 程序的列表，□入：

```
# ps -axww|grep getty
```

□将看到一个□端的□□。例如，下面的□示表明一个getty 正在第二个串口 ttyu1 □行， 正在 /etc/gettytab 中使用 std.38400 的□□：

```
22189  d1  Is+    0:00.03 /usr/libexec/getty std.38400 ttyu1
```

如果没有 getty □程□行， □信□已□在/etc/ttys中□用了端口。 在修改完/etc/ttys文件后，□得□行 kill -HUP 1。

如果 getty □程□□在□行， 但□端上仍然没有□示出登□提示， 或者□然□示了□缺不允□□□入， □的□端或□□可能不支持硬件握手。 □□□将 /etc/ttys 中的 std.38400 改□ 3wire.38400 (注意在改完 /etc/ttys 之后要 kill -HUP 1)。 3wire □□和 std □似， 但忽略硬件握手。 □可能需要在使用 3wire □□少波特率或□用□件流控制以避免□冲区溢出。

27.3.3.2. 出□一个 "□□" 而不是一个登□命令行

□信□端和 FreeBSD 使用相同的 bit/s □□率和奇偶校□□置。 □□一下 getty □程□信当前使用正□的 *getty* □型。如果没有， □□/etc/ttys然后□行kill -HUP 1。

27.3.3.3. 当□入密□□，字符□个□个出□

将□端 (或□端模□□件) 从 "半双工" 或 "本地回□" □成 "全双工"。

27.4. □入服□

从 FreeBSD 8.0 □始， 用于串口的□□□点从 /dev/cuadN 改□了 /dev/cuauN； 从 /dev/ttydN 改□了 /dev/ttyuN。 FreeBSD 7.X 用□需要根据□□情况□□□文□中的例子□行必要的□整。

□□入服□配置FreeBSD系□与□接到□端是非常相似的，除非□正在使用 modem来□号而不是□端。

27.4.1. 外置vs.内置modem

外置modem看起来很容易□号。 因□，外置 modem 可以通□□存在非易失性的RAM中的参数来配置， 它□通常提供指示器来□示重要的RS-232信号的状□。 不停□光的信号灯能□用□留下比□深刻的印象，而且指示器也可以用来□看modem是否正常地工作。

内置modem通常缺乏非易失性的RAM， 所以它的配置可能会限制在通 DIP 来置。 如果的内置modem有指示灯，也很看得到。

27.4.1.1. Modem和

如果使用一个外置的 modem，那将需要当的。 一个准的串口当足以至普通的信号能接上：

表 *12.* 信号名称

写	全名
RD	收到数据 (Received Data)
TD	出数据 (Transmitted Data)
DTR	数据端就 (Data Terminal Ready)
DSR	数据集就 (Data Set Ready)
DCD	数据波 (Data Carrier Detect) (RS-232 的收到路信号器)
SG	信号地 (Signal Ground)
RTS	要求送数据 (Request to Send)
CTS	允方送数据 (Clear to Send)

FreeBSD 速度超 2400 bps 的情形需要通 RTS 和 CTS 信号来完成流控制， 通 CD 信号来呼叫和挂机， 并通 DTR 信号来在会束制解器行位。 某些在接没有提供全部需要的信号， 会来，例如在挂断登会不消失， 就有可能是的。

与其它 UNIX® 操作系似， FreeBSD 使用硬件信号来呼叫， 以及在挂断挂断并位制解器。 FreeBSD 避免送命令制解器， 或其状。 如果熟悉通制解器来接基于 PC 的 BBS 系，可能看起来有点用。

27.4.2. 串口的考

FreeBSD支持基于 NS8250， NS16450， NS16550 和 NS16550A 的EIA RS-232C通接口。 8250和16450有字符冲。 16550提供了一个 16 个字符的冲， 可以提高更多的系性能。 因字符冲比 16 个字符的冲需要更多的系源来工作， 所以基于16550A的接口可能更好。 如果系没有活的串口，或有大的， 16 字符冲的于低率的通来更好。

27.4.3. 快速

于端， `init` 会在个配置串口上个入接生一个 `getty` 程。 例如， 如果一个 modem 被附在 /dev/ttyu0 中，用命令`ps ax`可以示下面些：

```
4850 ??  I      0:00.09 /usr/libexec/getty V19200 ttyu0
```

当用上modem， 并使用它行接， CD 就会被 modem 出。 内核注意到波信号已被到，需要完成 `getty` 端口的打。 `getty` 送一个登：在指定的初始速度上的命令行。 Getty 会合法的字符是否被接收， 在典型的配置中， 如果 "", `getty` 就会法

□□□速度，直到它接收到合理的字符。

用□在□入他/□的登□名称后， getty□行/usr/bin/login， □会要求用□□入密□来完成登□， 然后□□用□的shell。

27.4.4. 配置文件

如果希望允□□入□的 FreeBSD 系□， 在 /etc 目□中有三个系□配置文件需要□□注。 其一是 /etc/gettytab，其中包含用于 /usr/libexec/getty 服□的配置信息。 其二是 /etc/ttys， 它的作用是告□ /sbin/init □些 tty □□上□□□行 getty。 最后， □于端口的初始化命令， □放到 /etc/rc.d/serial 脚本中。

□于在 UNIX® 上配置□入□制解□器有□□主要的流派。 一□是将本地□算机到□制解□器的 RS-232 接口配置□固定速率。 □□做的好□是， □程用□□能立即□到系□的登□提示符， 而其缺点□是， 系□并不知道用□真□的数据速率是多少， 因而， □似 Emacs □□的程序， 也就无法□整它□□制屏幕的方式， 以便□慢速□接改善□□□□。

□一□流派将□制解□器的 RS-232 接口速率配置□随□程用□的□接速率□化。 例如， □ V.32bis (14.4 Kbps) □接，□制解□器会□自己的 RS-232 接口以 19.2 Kbps 的速率□行， 而 2400 bps □接， □会使□制解□器的 RS-232 接口以 2400 bps 的速率□行。 由于 getty 并不能□□具体的□制解□器的□接速率反□信息， 因此， getty 会以初始速度□出一个 login: 提示， 并□□用□的□□字符。 如果用□看到乱□， □他□□知道此□□按下 Enter □，直到看到可以辨□的提示符□止。 如果数据速率不匹配， □ getty 会将用□□入的任何信息均□□ "乱□"， 并□□以下一□速率来再次□出 login: 提示符。 □一□程可能需要令人作□地重□下去， 不□一般而言， 用□只要敲一□下□□就能看到正□的提示符了。 □然， □□登□□程看起来不如前面所介□的 "□定速率" 方法那□□□明了， 但使用低速□接的用□， 却可以在□行全屏幕程序□得到更好的交互□□。

□一□将尽可能公平地介□□于配置的信息， 但更着力于介□□制解□器速率随□接速率□化的配置方法。

27.4.4.1. /etc/gettytab

/etc/gettytab是一个用来配置 getty 信息的 termcap □格的文件。 □看看 gettytab 的□机手册了解完整的文件格式和功能列表。

27.4.4.1.1. □定速度的配置

如果□把□的modem的数据通□率□定在一个特殊的速度上， □不需要□ /etc/gettytab 文件作任何□化。

27.4.4.1.2. 匹配速度的配置

□将需要在 /etc/gettytab 中□置一个□□来告□ getty □希望在 modem 上使用的速度。 如果□的 modem 的速率是 2400 bit/s， □可以使用□有的 D2400 的□□。

```
#
# Fast dialup terminals, 2400/1200/300 rotary (can start either way)
#
D2400|d2400|Fast-Dial-2400:\
        :nx=D1200:tc=2400-baud:
3|D1200|Fast-Dial-1200:\
        :nx=D300:tc=1200-baud:
5|D300|Fast-Dial-300:\
        :nx=D2400:tc=300-baud:
```

如果□有一个更高速度的 modem， 必□在 /etc/gettytab 中添加一个□□。 下面是一个□□可以以最高 19.2 Kbit/s 的用在 14.4 Kbit/s的modem上的接口□□：

```
#
# Additions for a V.32bis Modem
#
um|V300|High Speed Modem at 300,8-bit:\
        :nx=V19200:tc=std.300:
un|V1200|High Speed Modem at 1200,8-bit:\
        :nx=V300:tc=std.1200:
uo|V2400|High Speed Modem at 2400,8-bit:\
        :nx=V1200:tc=std.2400:
up|V9600|High Speed Modem at 9600,8-bit:\
        :nx=V2400:tc=std.9600:
uq|V19200|High Speed Modem at 19200,8-bit:\
        :nx=V9600:tc=std.19200:
```

□□做的□果是 8-数据位， 没有奇偶校□的□接。

上面使用19.2 Kbit/s的□接速度的例子，也可以使用 9600 bit/s (for V.32)， 2400 bit/s， 1200 bit/s，300 bit/s， 直到 19.2 Kbit/s。 通□率的□□使用 nx= ("next table") 来□□。 □条□使用一个 tc= ("table continuation") 的□□来加速□于一个特殊□□率的□准□置。

如果□有28.8 Kbit/s的modem，或□想使用它的 14.4Kbit/s 模式， 就需要使用一个更高的超□ 19.2 Kbit/s 的通□速度的 modem。 □是一个□□ 57.6 Kbit/s 的 gettytab □□的例子：

```
#
# Additions for a V.32bis or V.34 Modem
# Starting at 57.6 Kbps
#
vm|VH300|Very High Speed Modem at 300,8-bit:\
        :nx=VH57600:tc=std.300:
vn|VH1200|Very High Speed Modem at 1200,8-bit:\
        :nx=VH300:tc=std.1200:
vo|VH2400|Very High Speed Modem at 2400,8-bit:\
        :nx=VH1200:tc=std.2400:
vp|VH9600|Very High Speed Modem at 9600,8-bit:\
        :nx=VH2400:tc=std.9600:
vq|VH57600|Very High Speed Modem at 57600,8-bit:\
        :nx=VH9600:tc=std.57600:
```

如果□的 CPU 速度□低， 或系□的□荷很重， 而且没有 16550A 的串口，□可能会在57.6 Kbit/s 上得到 sio "silo"□□。

27.4.4.2. /etc/ttys

/etc/ttys文件的配置在 在 /etc/ttys 中□加□端□□中介□□。 配置 modem 是相似的， 但我□必□指定一个不同的□端□型。 □定速度和匹配速度配置的通用格式是：

```
ttyu0   "/usr/libexec/getty xxx"   dialup on
```

上面的第一条是□个□□的□□特定文件 - ttyu0 表示 /dev/ttyu0 是□个 getty 将被□□的文件。 第二条 "/usr/libexec/getty xxx" 是将□行在□□上的□程 init。 第三条，dialup，是默□的□端□型。 第四个参数，on， 指出了□路是可操作的 init。 也可能会有第五个参数， secure， 但它将只被用作□有物理安全的□端(如系□□端)。

默□的□端□型可能依□于本地参考。 □号是□□的默□□端□型， 以至用□可以定制它□的登□脚本来注意□端什□□候□号， 和自□□□它□的□端□型。 然而， 作者□□它很容易在它的站点上指定 vt102 作□默□的□端□型， 因□用□□才在它□的□程系□上使用的是VT102模□器。

□□/etc/ttys作修改之后，□可以□送 init □程□一个 HUP 信号来重□文件。□可以使用下面的命令来□送信号：

```
# kill -HUP 1
```

如果□是□的第一次□置系□， □可能要在□信号 init 之前等一下， 等到□的 modem 正□地配置并□接好。

27.4.4.2.1. □定速度的配置

□于一个□定速度的配置，□的 ttys □□必□有一个□ getty 提供固定速度的□□。 □于一个速度被□定在 19.2kbit/s 的 modem， ttys □□是□□的：

```
ttyu0   "/usr/libexec/getty std.19200"   dialup on
```

如果□的 modem 被□定在一个不同的数据速度， □ std.speed 使用□当的速度来代替 std.19200。 □信□使用了一个在 /etc/gettytab 中列出的正□的□型。

27.4.4.2.2. 匹配速度的□置

在一个匹配速度的□置中，□的 ttys □需要参考在 /etc/gettytab □当的起始 "auto-baud" □□。 例如， 如果□□一个以 19.2 Kbit/s □始的可匹配速度的 modem 添加上面建□的□□， □的 ttys □□可能是□□的：

```
ttyu0   "/usr/libexec/getty V19200"   dialup on
```

27.4.4.3. /etc/rc.d/serial

高速□制解□器， 如使用 V.32、 V.32bis， 以及 V.34 的那些， 需要使用硬件 (RTS/CTS) 流控制。 □可以在 /etc/rc.d/serial 中□加 stty 命令来在 FreeBSD 内核中， □□制解□器□置硬件流控制□志。

例如， 在 1 号串口 (COM2) □入和□出□□上配置 termios □志 crtscts， 可以通□在 /etc/rc.d/serial □加下面的□置来□□：

```
# Serial port initial configuration
stty -f /dev/ttyu1.init crtscts
stty -f /dev/cuau1.init crtscts
```

27.4.5. Modem □置

如果□有一个 modem， 它的参数能被存□在非易失性的 RAM 中， □将必□使用一个□端程序来□置参数 （比如 MS-DOS® 下的 Telix 或者 FreeBSD 下的 `tip`）。 使用同□的通□速度来□接 modem 作□初始速度 `getty` 将使用和配置 modem 的非易失性 RAM 来□□□些要求：

- □接□宣告 CD
- 操作□宣告 DTR； DTR 消失□挂断□路并□位□制解□器
- CTS □□数据流控制
- 禁用 XON/XOFF 流控制
- RTS 接收数据流控制
- □静模式 (无返回□)
- 无命令回□

□□□□ modem 的文□□到□需要用什□命令和 DIP 接口□置。

例如，要在一个 U.S. Robotics® Sportster® 14400 的外置 modem 上□置上面的参数，可以用下面□些命令：

```
ATZ
AT&C1&D2&H1&I0&R2&W
```

□也可能想要在 modem 上□□机会□□□个□置， 例如它是否使用 V.42bis 和 MNP5 □□。

外置 modem 也有一些用来□置的 DIP □□， 也□□可以使用□些□置作□一个例子：

- Switch 1: UP - DTR Normal
- Switch 2: N/A (Verbal Result Codes/Numeric Result Codes)
- Switch 3: UP - Suppress Result Codes
- Switch 4: DOWN - No echo, offline commands
- Switch 5: UP - Auto Answer
- Switch 6: UP - Carrier Detect Normal
- Switch 7: UP - Load NVRAM Defaults
- Switch 8: N/A (Smart Mode/Dumb Mode)

在□号 modem 上的□果代□□□被 禁用/抑制， 以避免当 `getty` 在 modem □于命令模式并回□□入□□□地□出 `login:` 提示□可能造成的□□。 □□可能□致 `getty` 与 modem 之□□生更□的不必要交互。

27.4.5.1. □定速度的配置

□于□定速度的配置， □需要配置 modem 来□得一个不依□于通□率的□定的 modem到□算机 的□□率。 在一个 U.S. Robotics® Sportster® 14400 外置 modem 上， □些命令将□定 modem 到□算机 的□□率：

```
ATZ
AT&B1&W
```

27.4.5.2. 匹配速度的配置

□于一个□速的配置， □需要配置 modem □□它的串口□□率匹配接收的□□率。 在一个 U.S. Robotics® Sportster® 14400 的外置 modem 上， □些命令将□定 modem 的□□修正□□率□合命令要求的速度， 但允□串口速度□□没有□□的□接：

```
ATZ
AT&B2&W
```

27.4.5.3. □□modem的配置

大多数高速的modem提供了用来□看当前操作参数的命令。 在USR Sportster 14400外置modem上， 命令 ATI5 □示了存□在非易失性RAM中的□置。 要看看正□的 modem 操作参数， 可以使用命令 ATZ 然后是 ATI4。

如果□有一个不同牌子的 modem， □□ modem 的使用手册看看如何双重□□□的 modem 的配置参数。

27.4.6. □□解答

□儿是几个□□□号modem的□□。

27.4.6.1. □□FreeBSD系□

把□的modem□接到FreeBSD系□， □□系□， 然后， 如果□的 modem 有一个指示灯， 当登□□看看 modem 的 DTR 指示灯是否亮： 会在系□控制台出□命令行——如果它亮， 意味着 FreeBSD 已□在□当的通□端口□□了一个 getty □程， 等待 modem 接收一个呼叫。

如果DTR指示灯不亮， 通□控制台登□到 FreeBSD系□，然后□行一个 ps ax 命令来看 FreeBSD 是否正在正□的端口□行 getty□程。 □将在□程□示中看到像□□的一行：

```
  114 ??  I      0:00.10 /usr/libexec/getty V19200 ttyu0
  115 ??  I      0:00.10 /usr/libexec/getty V19200 ttyu1
```

如果□看到是□□的：

```
  114 d0  I      0:00.10 /usr/libexec/getty V19200 ttyu0
```

modem 不接收呼叫， □意味着 getty 已□在通□端口打□了。 □可以指出□□有□□或 modem □□配置， 因□ getty 无法打□通□端口。

如果□没有看到任何 getty □程等待打□想要的 ttyuN 端口， 在 /etc/ttys 中双□□的□□看看那儿是否有□□。 □外，□□日志文件 /var/log/messages 看看是否有一些来自 init 或 getty 的□□日志。 如果有任何信息，仔□□□配置文件 /etc/ttys 和 /etc/gettytab，□有相□的□□文件 /dev/ttyuN， 是否有□□，□失□□，或□失了□□指定文件。

27.4.6.2. □□接入Try Dialing In

□法□入系□。 □信使用8位， 没有奇偶□□， 在□程系□上的1阻止位。 如果□不能立刻得到一个命令行，□□□隔一秒按一下 `Enter`。 如果□仍没有看到一个登□： □法□送一个 BREAK。 如果□正使用一个高速的 modem 来□号， □在□定□号 modem 的接口速度后再□□。

如果□不能得到一个登□：prompt，再□□一下 /etc/gettytab，重□□□：

- 在/etc/ttys 中指定的初始可用的名称与 /etc/gettytab 的一个可用的相匹配。
- □个 nx= □□与□一个 gettytab 可用名称匹配。
- □个 tc= □□与□一个 gettytab可用名称相匹配。

如果□□号但 FreeBSD 系□上的 modem 没有回□， □信 modem 能回□□□。 如果 modem 看起来配置正□了，通□□□ modem 的指示灯来□□ DTR □□接正□。

如果□做了好几次，它仍然无法工作，打断一会，等会再□□。 如果□不能工作， 也□□□□□一封□子□件□ FreeBSD 一般□□□件列表 □求□助。

27.5. □出□□

从 FreeBSD 8.0 □始， 用于串口的□□□点从 /dev/cuadN 改□了 /dev/cuauN； 从 /dev/ttydN 改□了 /dev/ttyuN。 FreeBSD 7.X 用□需要根据□□情况□□□文□中的例子□行必要的□整。

下面将□□的主机通□ modem □接到□一台□算机上。 □只要□当地建立一个□端作□□程主机就可以。

□可以用来登□□一个BBS。

如果□用 PPP 有□□， 那□□□接可以用来从 Internet 上下□一个文件。 如果□必□ FTP 一些□西， 而 PPP 断了，使用□端会□来 FTP 它□。 然后使用 zmodem 来把它□□□到□的机器上。

27.5.1. 我的Stock Hayes Modem不被支持，我□□□□?

事□上， □机手册□于□个的描述已□□□了。 一个通用的 Hayes□号已□内建其中。 只要在□的 /etc/remote 文件中使用 at=hayes。

Hayes □□不□ "□明" 只能□出一些比□新的 modem 的高□特性 - 如 BUSY、 NO DIALTONE， 或 CONNECT 115200 的信息将被□乱。 当□使用的□候， □必□把□些信息□掉。(通□ ATX0&W)。

□外，□号的延□是 60 秒。 □的 modem 可能使用□外的□□或提示□□有其他的通□□□。 □□ ATS7=45&W。

27.5.2. 我如何□入□些 AT 命令?

在 /etc/remote 文件中□加一个 "direct" □。 □例而言， 如果□的□制解□器挂在第一个串口， 即 /dev/cuau0 上， □□添加下面□行：

```
cuau0:dv=/dev/cuau0:br#19200:pa=none
```

此□□使用□的 modem 所支持的最高 br bps 速率。 接下来， □入 `tip cuau0` 就可以□到 modem 上了。

此外， 也可以 `root` 身□□行 `cu` 命令：

```
# cu -lline -sspeed
```

line 是串口 (例如 /dev/cuau0) 而 *speed* □是速率 (如 `57600`)。 当□□入完 AT 之后， 按 `~.` 即可退出。

27.5.3. □在pn @□□不能工作？

在□□号□中的 `@` □□告□□算机在 /etc/phones 文件中□□一个□□号□。 但 `@` □□也是一个在像 /etc/remote □□的可用文件中的特殊字符。 用一个反斜□符号退出：

```
pn=\@
```

27.5.4. 我如何在命令行□□□号□?

在□的 /etc/remote 文件中通常放着一个叫做 "generic" 的□□。 例如：

```
tip115200|Dial any phone number at 115200 bps:\
        :dv=/dev/cuau0:br#115200:at=hayes:pa=none:du:
tip57600|Dial any phone number at 57600 bps:\
        :dv=/dev/cuau0:br#57600:at=hayes:pa=none:du:
```

然后， 可以□行：

```
# tip -115200 5551234
```

如果□更喜□cu而不是tip，使用一个通用的cu□□：

```
cu115200|Use cu to dial any number at 115200bps:\
        :dv=/dev/cuau1:br#57600:at=hayes:pa=none:du:
```

然后□入：

```
# cu 5551234 -s 115200
```

27.5.5. □□做□是否□次都需要重新□入 bps 速率?

添加一□ `tip1200` 或 `cu1200`， 并将 bps 速率□成更合□的□。 `tip` 的默□□是1200 bps， 也就是□什□会有 `tip1200` □条□□的原因。 □然□并不需要使用 1200 bps。

27.5.6. 我通□一个□端服□器□□了很多主机。

除非□次都要等到□□接到主机然后□入 `CONNECT host`， 否□使用 `tip` 的 `cm` 功能。 例如， 在 /etc/remote 中的□些□□：

```
pain|pain.deep13.com|Forrester's machine:\
        :cm=CONNECT pain\n:tc=deep13:
muffin|muffin.deep13.com|Frank's machine:\
        :cm=CONNECT muffin\n:tc=deep13:
deep13:Gizmonics Institute terminal server:\
        :dv=/dev/cuau2:br#38400:at=hayes:du:pa=none:pn=5551234:
```

将□□□入 `tip pain` 或 `tip muffin` □接到主机 `pain` 或 `muffin`， 和 `tip deep13` □接到□端服□器。

27.5.7. tip能□□个站点□用多个□路□？

□常有一个□□， 一个大学有几个modem□路， 几千个学生□法使用它□。

在 /etc/remote 中□□的大学添加一个□□， 然后□ `pn` 功能使用 `@` □□：

```
big-university:\
        :pn=\@:tc=dialout
dialout:\
        :dv=/dev/cuau3:br#9600:at=courier:du:pa=none:
```

接着， 在 /etc/phones 中列出大学的□□号□：

```
big-university 5551111
big-university 5551112
big-university 5551113
big-university 5551114
```

`tip` 将按□序□用□一个，然后就停止。 如果想□□□□， 隔一段□□再□行 `tip`。

27.5.8. □什□我必□□入 `Ctrl` + `P` □次才能□出 `Ctrl` + `P` 一次?

`Ctrl` + `P` 是默□的"□制"字符，被用来告□ `tip` 下一个字符是文字的数据。□可以用 `~s` □任何其他的字符□置□制字符，□意思是 "□置一个□量"。

在新的一行□入 `~sforce=single-char`。 *single-char* 是任何□□的字符。 如果□□漏了 *single-char*， 那□制字符就是空字符，□可以□入 `Ctrl` + `2` 或 `Ctrl` + `Space` 来完成。更好的*single-char*是 `Shift` + `Ctrl` + `6`，□只用在一些□端服□器上。

通□在□的 $HOME/.tiprc 文件中指定下面□行， 就可以得到□想要的任何□制字符：

```
force=single-char
```

27.5.9. 突然我□入的□一□□西都□成了大写??

□一定是□入了 `Ctrl` + `A`， 即 `tip` 的 "raise character"， 会□□地指定成坏掉的 caps-lock□。 使用上面的 `~s` 来合理地□置各□ `raisechar`。 事□上， 如果□不想使用□些特性的□，□可以用同□的方法□置□制字符。

□儿有一个很好的示例 .tiprc 文件，□ Emacs用□来□，需要□常按 `Ctrl` + `2` 和 `Ctrl` + `A`：

```
force=^^
raisechar=^^
```

`^^` 是 `Shift` + `Ctrl` + `6`.

27.5.10. 如何用 `tip` 做文件□□？

如果□正在与□一台 UNIX® 系□□□， □可以用 `~p`(put) 和 `~t` (take) □送和接收文件。 □些命令可以在□程系□上□行 `cat` 和 `echo` 来接收和□送文件。 □法是□□的：

`~p` local-file [remote-file]

`~t` remote-file [local-file]

由于没有□□校□， 所以□需要使用其他□□， 如 zmodem。

27.5.11. 我如何用`tip`□行zmodem？

要接收□些文件，可以在□程□端□□□送程序。然后，□入 `~C rz` 在本地□始接收它□。 要□送文件， 可以在□程□端□□接收程序。 然后， □入 `~C sz files` 把它□□送到□程系□。

27.6. □置串口控制台

从 FreeBSD 8.0 □始， 用于串口的□□□点从 /dev/cuadN 改□了 /dev/cuauN； 从 /dev/ttydN 改□了 /dev/ttyuN。 FreeBSD 7.X 用□需要根据□□情况□□□文□中的例子□行必要的□整。

27.6.1. 介□

FreeBSD可以通□一个串口只使用一个□ (dumb) □端就可以□□一个系□。 □□一□配置只有□□人能使用：希望在机器上安装 FreeBSD 的系□管理□， 他没有□□或□示器， □有就是要□□内核或□□□□程序的□□人□。

就象 FreeBSD 引□□程 描述的， FreeBSD 采用一个三□的□□□程。 最先□□□存在 FreeBSD □□磁□的□□ slice 的□□代□□中。 引□□然后就被加□， 接着□行第三□□□引□器 (/boot/loader)。

□了□置串口控制台， □必□配置□□代□□， □□引□器代□和内核。

27.6.2. 串口控制台的配置， □明版

□一□假定□使用默□的配置， 只希望迅速地□得□于配置串口控制台的概□。

1. 使用串口□□□接 COM1 和控制□端。
2. 要在串口控制台上□示所有的引□信息， 需要以超□用□的身□□行下面的命令：

```
# echo 'console="comconsole"' >> /boot/loader.conf
```

3. □□ /etc/ttys 并把 ttyu0 的 `off` 改□ `on`， `dialup` 改□ `vt100`。 否□通□串口控制台上将不会提示□入口令， 从而□致潜在的安全漏洞。
4. 重新□□并□察是否生效。

如果需要不同的配置， 更□一□的配置□□可以在 串口控制台的□置 □到。

27.6.3. 串口控制台的□置

1. 准□一根串口□□。

   □需要使用一个 null-modem 的□□或□准的串口□和一个 null-modem □配器。 □参考 □□和端口 中有□串口□的□□。

2. 拔掉□□。

   □大多数的PC在□机□□的□候会□□到□□， 如果没有□□到□□， □会出□□□。 一些机器会提示缺少□□，就不会□□引□系□。

   如果□的□算机出□□□， 但仍能□□□□， □可以不必理它。

   如果□的□算机没有□□拒□□□， 那□需要配置 BIOS 来避免它。 □参考□的主板的使用□明了解更多□□。

   在 BIOS 中将□□□□ "Not installed" (未安装)。 □在□仍然无法使用□□。 □□做只是告□ BIOS 在□□□不要探□□□。 □的 BIOS 不□抱怨□□不存在。 即使□一□志□置□ "Not installed" □， 只要把□□□上， 它就仍可使用。 如果以上的□□不存在于 BIOS 中， 可□□□□ "Halt on Error" □□。 把□一□□置□ "All but Keyboard" 或者是 "No Errors"， 都能器到相同的作用。

   如果系□有 PS/2 鼠□， 如果幸□的□， □也可以象□□一□把它拔下来， □是因□ PS/2 鼠□与□□的一些硬件是共享的， □的鼠□□上去， 系□会□□□□仍在那儿。

3. □一个□ (dumb) □端到COM1：（sio0）。

   如果□没有□□端， 可以使用一个比□老的□有一个 modem 程序的PC/XT机器， 或在其他 UNIX® 机器上的串口。 如果□没有 COM1：(sio0)， 去□一个。 □□， □就不能只能□□ COM1：来□□系□。 如果□已□在□一台□□上使用 COM1， □必□□□□除那个□□， 然后安装一个新的系□引□□和内核。

4. □信□的内核配置文件已□□ COM1：(sio0) □置了□当的□□：

   有□的□□是：

   `0x10`

   □用控制台支持。 如果没有□置它， □其他的控制台□□都会被忽略。 □在， □大多数的□置都有控制台的支持。 □个□□的第一个就是首□的。 □个□独□□是不能□保串口□用于控制台的，□置下面的□□或加上下面描述的 `-h` □□， 和□个放在一起。

   `0x20`

   无□是否使用了下面将要□□的 `-h` □□， 都□制□个□元作□控制台 (除非使用了更高□先□的控制台)。□志 `0x20` 必□与 `0x10` 一起使用。

   `0x40`

   □留□个□元 (配合 `0x10`) 并□它不能用于普通的使用。 □不□在希望作□控制台的串口□元上□置□个□志。 □一□志是□内核□程□□准□的。 参□ □□者手册 以了解□于□程□□更□一□的情况。

   例如：

```
device sio0 flags 0x10
```

看看 sio(4) 的□机手册了解更多信息。

如果□□没有被□置， □必□□行UserConfig或重新□□内核。

5. 在□□磁□的 a 分区的根目□□建 boot.config 文件。

□个文件将指□引□□代□如何□□系□。 □了激活串口控制台， □必□有一个或多个下面的□□——如果□要多个□□， 在同一行必□都包含它□：

-h

切□内部和串口控制台。 □使用□个来交□控制台□□。 例如， 如果□从内部控制台□□， □可以使用 -h 来直接使用□□引□器和内核来使用串口作□它的控制台□□。 □外， 如果□从串口□□， □可以使用 -h 来告□□□引□器和内核使用□示□□作□控制台。

-D

切□□一和双重控制台配置。 在□一配置中， 控制台将是本机的控制台 (□示□□) 或串口。 在双重控制台配置中， □示□□和串口将同□成□控制台， 无□ -h 的□□的情形。 然而， 双控制台配置只在引□□□行的□程中起作用。 一旦□□引□器□得控制， 由 -h □□指定的控制台将成□唯一的控制台。

-P

在□□□，探□□□。如果□□□不到， -D 和 -h □□会自□□置。

由于当前版本引□□的空□限制， -P □□只能探□□展的□□。 少于101□的□□将无法被探□到。 如果□□到□个情况， □必□避免使用 -P □□。 目前□没有□□□个□□的□法。

使用 -P □□来自□□□控制台， 或使用 -h □□来激活控制台。

□也可以使用boot□机文□中所描述的其他□□。

除了 -P □□， 所有□□将被□□□□引□器 (/boot/loader)。 □□引□器将通□□□ -h □□的状□来决定是□示□□成□控制台， □是串口成□控制台。 □表示如果□指定 -D □□， 但在 /boot.config 中没有 -h □□， □在□□代□□□使用串口作□控制台。 □□引□器将使用内部□示□□作□控制台。

6. □□机器

当□□□□的FreeBSD□，引□□将把 /boot.config 的内容□□控制台。例如：

```
/boot.config: -P
Keyboard: no
```

如果□把 -P 放在 /boot.config 中并指出□□存在或不存在， 那将只出□第二行。□些信息会被定位到串口或内部控制台， 或□者同□， □完全取决于 /boot.config 中的□□。

□□	送出消息的□□
none	内部控制台
-h	串口控制台
-D	串口控制台和内部控制台
-Dh	串口控制台和内部控制台
-P，有□□	内部控制台
-P，无□□	串口控制台

出□上面信息后，在引□□加□□□引□器和更多信息被映到屏幕之前将有一个小小的停□。在通常情况下，□不需要打断□□□程，但□了□信□置是否正□，□也可以□□做。

在控制台上按 `Enter` 以外的任意□就能打断□□□程。引□□将□入命令行模式。□将看到：

```
>> FreeBSD/i386 BOOT
Default: 0:ad(0,a)/boot/loader
boot:
```

□□上面出□的信息，可能是串口，或内部控制台，或□个同□，完全取决于□在 /boot.config 中的□□。如果信息出□在正□的控制台，按 `Enter` □□□□□程。

如果□要使用串口控制台，但□没有看到命令行，那可能□置有□□。□□，□入 `-h` 然后按 `Enter` 或 `Return` 来告□引□□（然后是□□引□器和内核）□□串口作□控制台。一旦系□起来了，就可以回去□□一下是什□出了□□。

□□引□器加□完后，□将□入□□□程的第三□，□仍然可以在□□引□器通□□定□喜□的□境来切□内部控制台和串口控制台。参考 从□□引□器修改控制台。

27.6.4. 摘要

□是几个在□章要□□的几个□置和□□的控制台的摘要。

27.6.4.1. 例1： □□ sio0 □置□□ 0x10

```
device sio0 flags 0x10
```

在 /boot.config 中的□□	引□□□行□所用的控制台	引□加□器□行□所用的控制台	内核所用的控制台
无	内部	内部	内部
-h	串口	串口	串口
-D	串口和内部	内部	内部
-Dh	串口和内部	串口	串口
-P，有□□	内部	内部	内部

在 /boot.config 中的□□	引□□□行□所用的控制台	引□加□器□行□所用的控制台	内核所用的控制台
-P， 没有□□	串口和内部	串口	串口

27.6.4.2. 例2：□□ sio0 □置□□□ 0x30

```
device sio0 flags 0x30
```

在 /boot.config 中的□□	引□□□行□所用的控制台	引□加□器□行□所用的控制台	内核所用的控制台
无	内部	内部	串口
-h	串口	串口	串口
-D	串口和内部	内部	串口
-Dh	串口和内部	串口	串口
-P， 有□□	内部	内部	串口
-P， 没有□□	串口和内部	串口	串口

27.6.5. 串口控制台的提示

27.6.5.1. □置更高的串口速度

在默□配置中， 串口的□置是： 速率 9600 波特、 8 数据位、 无奇偶校□位、 1 停止位。 如果□希望修改默□的控制台速率， 可以采用下列几□方法之一：

- 将 BOOT_COMCONSOLE_SPEED 配置□希望的速率， 并重新□□引□□。 □参□ 使用 sio0 以外的串口 作□控制台 以了解如何□□和安装新的引□□。

  如果串口控制台已配置□使用 -h 以外的其它方式引□， 或者内核使用的速率与引□□不同， □必需在内核配置文件中加入下述□置， 并重新□□新内核：

  ```
  options CONSPEED=19200
  ```

- 使用内核引□□□ -S. -S □个命令行□□可以加到 /boot.config 中。 □参□□机手册 boot(8) 以□得如何在 /boot.config 中□加□□， 以及其它的可用□□。

- 在□的 /boot/loader.conf 文件中□用 comconsole_speed □□。

  使用□个□□□，□□需要在 /boot/loader.conf 中配置 console、 boot_serial， 以及 boot_multicons。 下面是一个利用 comconsole_speed 改□串口控制台速率的例子：

  ```
  boot_multicons="YES"
  boot_serial="YES"
  comconsole_speed="115200"
  console="comconsole,vidconsole"
  ```

27.6.5.2. 使用 sio0 以外的串口 作□控制台

使用串口而不是 sio0 作□控制台需要做一些重□□。 如果□无□如何都要使用□一个串口， 重新□□引□□， □□引□器和内核。

1. 取得内核源代□ (参考 更新与升□ FreeBSD)。
2. □□ /etc/make.conf 文件， 然后□置 BOOT_COMCONSOLE_PORT作□□要使用 (0x3f8、 0x2f8、 0x3E8 或 0x2E8) 端口的地址。 只有 sio0 到 sio3 (COM1 到 COM4) 都可以使用； 但多口串口□将不会工作。 不需要任何中断□置。
3. □建一个定制的内核配置文件， 在□要使用的串口添加合□的□□。 例如， 如果要将 sio1 (COM2) 作□控制台：

   ```
   device sio1 flags 0x10
   ```

   或

   ```
   device sio1 flags 0x30
   ```

   其他端口的控制台□□也不要□。
4. 重新□□和安装引□□：

   ```
   # cd /sys/boot
   # make clean
   # make
   # make install
   ```
5. 重建和安装内核。
6. 用 bsdlabel(8) 将引□□写到□□□上，然后从新内核□□。

27.6.5.3. 通□串口□□入DDB□□器

```
options BREAK_TO_DEBUGGER
options DDB
```

27.6.5.4. 在串口控制台上得到一个登□命令行

□可能希望通□串口□□入登□提示， □在□可以看到□□信息， 通□串口控制台□入内核□□信息。可以□□做。

用一个□□器打□ /etc/ttys 文件， 然后□到下面的行：

```
ttyu0 "/usr/libexec/getty std.9600" unknown off secure
ttyu1 "/usr/libexec/getty std.9600" unknown off secure
ttyu2 "/usr/libexec/getty std.9600" unknown off secure
ttyu3 "/usr/libexec/getty std.9600" unknown off secure
```

ttyu0 到 ttyu3 相当于 COM1 到 COM4。 可以打□或□□某个端口。 如果□已□改□了串口的速度， □必□改掉□准的 9600 与当前的例如 19200 相匹配。

□也可以改□□端的□型从不知名的到□串口□端的真□□型。 □□完□个文件， □必□ kill -HUP 1 来使□个修改生效。

27.6.6. 从□□引□器修改控制台

前面一□描述了如何通□□整引□□来□定串口控制台。 □□将□到在□□引□器中通□□入一些命令和□境□量来指定控制台。 由于□□引□器会被□□□程的第三□所□用， 引□□以后， 在□□引□器中的□置将忽略在引□□中的□置。

27.6.6.1. 配置串口控制台

□可以很容易地指定□□引□器和内核来使用串口控制台， 只需要在 /boot/loader.ronf中写入下面□行：

```
console="comconsole"
```

无□前一□中的引□□如何配置， □个□置都会生效。

□最好把上面一行放在 /boot/loader.conf 文件的第一行，以便尽早地在□□□看到串口控制台的□□信息。

同□地，□可以指定内部控制台□：

```
console="vidconsole"
```

如果□不□置□□引□□境□量控制台， □□引□器和内核将使用在引□□□用 -h □□指定的控制台。

控制台可以在 /boot/loader.conf.local 或者是在 /boot/loader.conf 中指定。

看看 loader.conf(5) 的□机手册了解更多信息。

目前， 引□□尚不提供与引□加□器的 -P □□等价的□□， □外， 它也不能根据是否有□□存在自□决定□□使用内部控制台□是串口控制台。

27.6.6.2. 使用串口而不是sio0作□控制台

要使用一个串口而不是 sio0 作□串口控制台 需要重新□□□□引□器。下面的□□跟 使用 sio0 以外的串口 作□控制台 描述的相似。

27.6.7. 警告

□篇文章本意是想告□人□如何□定没有□示□□或□□的□用服□器。　不幸的是，　□大多数系□没有□□可以□□□□，而没有□示□□就不□□□□。 使用 AMI BIOS 的机器可以通□在 CMOS 中将 "graphics adapter" □□□ "Not installed" 来在□□□不要求□示□配器。

然而， □多机器并不支持□个□□， 如果□的系□没有□示硬件就拒□□□。 □于□些机器， 即使□没有□示器， 也必□在机器上□上□示□配器。 建□□□□采用 AMI BIOS 的机器。

Chapter 28. PPP 和 SLIP

28.1. 概述

FreeBSD 有很多方法可以将□算机与□算机□接起来。 通□使用□号 modem 来建立网□或 Internet □接， 或允□其他人通□□的机器来□上网□， □些都要求使用 PPP 或 SLIP。 □章将□□介□□置□些基于 modem 的通信服□的方法。

□完□一章， □将了解：

- 如何□置用□□ PPP。
- 如何□置内核□ PPP。 (□限 FreeBSD 7.X)。
- 如何□置 PPPoE (PPP over Ethernet)。
- 如何□置 PPPoA (PPP over ATM)。
- 如何配置和安装 SLIP 客□端和服□器。 (□限 FreeBSD 7.X)。

在□□□章之前， □□：

- 熟悉基本的网□□□。
- 理解□号□接和 PPP、 SLIP 的基□知□。

□可能想知道用□□ PPP 与内核□ PPP 之□的不同之□。 回答很□□： 用□□ PPP □理用□□的□入和□出数据，而不是内核□。 在内核与用□区之□□制数据的花□要大一些， 但它能提供具有更多特性的PPP□□。 用□□PPP使用 tun □□与外界通信而内核□ PPP 使用 ppp □□。

在□章中， 如果没有特殊□明， □ ppp 指的是用□□ PPP， 除非需要和其它 PPP □件， 例如 pppd (□限 FreeBSD 7.X) 加以区分。 □外， 若没有□外的注明， 本章所介□的所有命令都需要以 `root` 身□来□行□限。

28.2. 使用用□□ PPP

从 FreeBSD 8.0 □始， uart(4) □□取代了 sio(4) □□。 用以表示串口的□□□点由分□ /dev/cuadN 改□了 /dev/cuauN， 并从 /dev/ttydN 改□了 /dev/ttyuN。 FreeBSD 7.X 用□在升□□需要因□之□配置文件□行必要的更改。

28.2.1. 用□□ PPP

28.2.1.1. 前提条件

本章假定□具□如下条件：

- □有一个 ISP 提供的用于□接使用 PPP 的□号。
- □需要有□接在系□上， 并做了正□配置的 modem， 或其他能□□接□ ISP 的□□。
- ISP 的□号号□。

- □的登□名称和密□ (可能是一般的 UNIX □格的登□名和密□□， 也可能是 PAP 或 CHAP 登□名和密□□)。
- 一个或多个域名服□器 IP 地址。 通常， □会从ISP□得到□个□□的IP地址。 如果□至少得到了一个，就可以在文件 ppp.conf 中加入 `enable dns` 命令使 ppp □置域名服□。 □个功能取决于 ISP □支持 DNS □商的具体□□。

下面的信息由□的 ISP 提供， 但不是必需的：

- ISP的网□IP地址。 网□是□准□□接， 并□□ 默□路由 的主机。 如果□没有□个信息， □可以虚□一个， 在□接□ ISP 的 PPP 服□器会自□告□□正□的□。

  □个虚□的 IP 地址在 ppp 中□做 `HISADDR`。

- 准□使用的子网掩□。 如果ISP没有提供， 一般使用 `255.255.255.255` 是没有□□的。
- 如果 ISP 提供了静□的IP地址和主机名， 可以□入它□。 反之， □□□□方主机指定它□□合□的 IP 地址。

如果□不知道□些信息， □与□的 ISP □系。

在□□中， 所有作□例子展示的配置文件中都有行号。 □些行号只是□了使解□和□□□得方便，在真□的文件中并不存在。 此外， 在必要□□使用 Tab 和空格来□行□□。

28.2.1.2. PPP自□化配置

ppp和pppd(PPP的内核□□□， □限 FreeBSD 7.X) 都使用 /etc/ppp 目□中的配置文件。 用□□ PPP 的例子可以在 /usr/shared/examples/ppp/ 中□到。

配置ppp要求根据□的需要□□几个文件。 □□□几个文件取决于□的 IP 是静□分配 (□次都使用同一个地址) □是□□分配的 (□次□接到 ISP 都会□得不同的 IP 地址)。

28.2.1.2.1. PPP和静□IP地址

□需要□□配置文件/etc/ppp/ppp.conf， 如下所示。

以冒号:□尾的行从第一列 (行首)□始， 其它所有的行都要使用空格或制表符 (Tab) 来□□。

```
1     default:
2       set log Phase Chat LCP IPCP CCP tun command
3       ident user-ppp VERSION (built COMPILATIONDATE)
4       set device /dev/cuau0
5       set speed 115200
6       set dial "ABORT BUSY ABORT NO\\sCARRIER TIMEOUT 5 \
7                 \"\" AT OK-AT-OK ATE1Q0 OK \\dATDT\\T TIMEOUT 40 CONNECT"
8       set timeout 180
9       enable dns
10
11    provider:
12      set phone "(123) 456 7890"
13      set authname foo
14      set authkey bar
15      set login "TIMEOUT 10 \"\" \"\" gin:--gin: \\U word: \\P col: ppp"
16      set timeout 300
17      set ifaddr x.x.x.x y.y.y.y 255.255.255.255 0.0.0.0
18      add default HISADDR
```

行**1**：

指定默□的□。 当PPP□行□□个□中的命令将自□□行。

行**2**：

□用登□参数。 工作正常后， □避免□生□多的日志文件， □行□□□化□：

```
set log phase tun
```

行 **3**：

告□ PPP □□向□方□□自己。 如果在建立或使用□接□遇到任何麻□， PPP就会向□方主机自我□□。 □方主机管理□在□理□个□□□， □些信息会有用。

行 **4**：

□明modem要□接的端口号。 COM1 □□的□□是 /dev/cuau0 而 COM2 □□的□是 /dev/cuau1。

行 **5**：

□置□接的速度。 如果 115200 有□□， □□ 38400。

行 **6 & 7**：

□号字符串。 用□□ PPP 使用一□与 chat(8)程序相似的□法。 □参考□机手册了解□□□言的相□信息。

注意， □了便于□□此命令□行了□行。 任何 ppp.conf 里的命令都可以□□做， 前提是行的最后一个字符必□是 \。

行 **8**：

□置□接的□□□隔。 默□是 180 秒， 所以□一行是多余的。

行 **9**：

告□PPP向□方主机□□本地域名解析□置。 如果□□行了本地的域名服□器， 要注□或□除掉□一行。

行 **10**：

□了可□性的需要□置一个空行。 空行会被PPP忽略。

行 **11**：

□ "provider"指定一个□。 可以改成 ISP的名字， □□□以后就可以使用 `load ISP` 来□□□接。

行 **12**：

□置提供商的□□号□。 多个□□号□可以使用冒号 (`:`) 或管道符号 (`|`) 隔□。 □□个字符的区□在ppp(8)的□机手册中有介□。 □的来□， 如果□要循□使用□些号□， 可以使用冒号。 如果□想使用第一个号□，当第一个号□失□了再用第二个号□， 就使用管道符号。 如所示的那□， 要□整个□□号□加上引号(")。

如果□□号□里有空格， 必□用引号(")将其括起来。 否□会造成□□却□以察□的□□。

行 **13 & 14**：

指定用□名和密□。 当使用 UNIX® □格的命令提示符登□□， □些□可以用□有 \U \P 参数的 `set login` 命令□行修改。 当使用PAP或CHAP□行□接□， □些□在□□使用。

行 **15**：

如果□使用的是PAP或者CHAP， 在□里就不会有登□。 要注□或□除掉□一行。 □参考 PAP 和 CHAP□□以了解更多□□。

登□命令是的□法是chat□型的。 在□个例子中是□□的：

```
J. Random Provider
login: foo
password: bar
protocol: ppp
```

□需要改□□个脚本以□合□自己的需要。 当□第一次写□个脚本□， □当□保已□□用 "chat" 并□于登□状□，□□□才能□□通信是否正在按□□□行。

行**16**：

□置默□的超□□□。 □里， □接若在 300 秒内无□□将被断□。如果□不想□置成超□， 将□个□□置成0，或在命令行使用 `-ddial` □□。

行 **17**：

□置接口地址。 □需要用 ISP 提供□□的 IP 地址替□字符串 *x.x.x.x*， 用 ISP 的网□ IP 地址 (即□要□接的主机)替□字符串 *y.y.y.y*。 如果ISP没有□□提供网□地址， 可以使用 `10.0.0.2/0`。 如果□需要使用一个 "猜到"的地址， □□保在 /etc/ppp/ppp.linkup 中□□个 PPP和□□IP地址 指令□建了□一□。 如果没有□一行， `ppp` 将无法以 `-auto` 模式□行。

第**18**行：

添加一个到ISP网□的默□路由。 `HISADDR`□个□□字会被第17行所指定的网□地址替□。 □行必□出□在第17行之后，以免在 `HISADDR` 初始化之前使用它的□。

如果□不想使用 -auto 的 PPP，□□行□□到 ppp.linkup 文件中。

若□有一个静□IP地址，且使用-auto模式□行ppp(因□在□接之前已□正□□置了路由表□)，那就不需要再向ppp.linkup添加□。□可能希望在□接以后□建一个□来□用程序。□在以后的sendmail的例子中会解□。

示例配置文件可以在目□ /usr/shared/examples/ppp/ 中□到。

28.2.1.2.2. PPP和□□IP地址

如果ISP没□□指定静□的IP地址，ppp要被配置成能□与□方□商□定本地和□程地址。要完成□□工作，先要"猜"一个IP地址，然后允□ ppp在□接后使用IP配置□□(IPCP)□行正□配置。ppp.conf的配置是与PPP和静□IP地址一□的，除了以下的改□：

```
17      set ifaddr 10.0.0.1/0 10.0.0.2/0 255.255.255.255 0.0.0.0
```

再次□□，不要包括行号，它只是一个引用□□。□排一个空格是必需的。

行**17**：

/ 字符后面是 PPP 所要求的地址掩□。□可以根据需要使用不同 IP 地址，但以上的例子永□是可行的。

最后的参数(0.0.0.0)告□ PPP从0.0.0.0而不是10.0.0.1□始□商地址，□于有些ISP，□是必需的。不要将 0.0.0.0 作□ set ifaddr 的第一个参数，因□□使得 PPP 在 -auto 模式□不能□置初始路由。

如果□不□行-auto模式，就需要在/etc/ppp/ppp.linkup中□建一个□。□接建立之后，ppp.linkup被□用。□□候，ppp将指派接口地址，接着再添加路由表□：

```
1     provider:
2        add default HISADDR
```

行 **1**：

□了建立□接，ppp会按照如下□□在ppp.linkup□□□:首先，□□□□相同的□□（如同在ppp.conf一□）。如果失□了，□□作□网□ IP 地址的□，此□是四个八位字□的□格。如果依旧没有□到，就□□ MYADDR □

行 **2**：

□行告□ ppp添加指向 HISADDR的默□路由。HISADDR由通□IPCP□商得到的IP号替□。

参考/usr/shared/examples/ppp/ppp.conf.sample和/usr/shared/examples/ppp/ppp.linkup.sample中的pmdemand□以□取□□化的例子。

28.2.1.2.3. 接收□入

当要配置 ppp接受来自LAN上的□入□，□需要决定是否将包□□LAN。如果是的□，□就必□从 LAN 子网中□□方分配一个IP，需要在文件 /etc/ppp/ppp.conf 中使用命令 enable proxy。□□□□□定文件 /etc/rc.conf 中包含以下内容：

```
gateway_enable="YES"
```

28.2.1.2.4. 使用□个getty？

配置 FreeBSD 的□号服□ 描述了如何用 getty(8) 来□□□号服□。

除了 getty 之外□有 mgetty (可通□ comms/mgetty+sendfax port 来安装)， 它是 getty 的智能版本，是按照□号□的思想□□的。

使用 mgetty 的好□是它能□□地与 modem □行 会□， □就意味着如果在/etc/ttys中的端口被□□， □的moderm就不会回□□入。

□新版本的 mgetty (从 0.99beta 起) 也支持自□□□ PPP 数据流， □□即便客□端不使用脚本也能□□服□器了。

参考Mgetty 和 AutoPPP的□机手册了解更多信息。

28.2.1.2.5. PPP □限

ppp 命令通常必□以 root 用□的身□□行。 如果希望以普通用□的身□□□ ppp 服□ (就像下面描述的那□)， 就必□把此用□加入 network □， 使其□得□行 ppp 的□限。

□□需要使用allow命令使用□能□□配置文 件的一个或多个部分：

```
allow users fred mary
```

如果□个命令被用在 default 部分中， □可以□指定的用□□□任何□西。

28.2.1.2.6. □□IP用□的PPP Shell

□建一个名□/etc/ppp/ppp-shell文件， 加入以下内容：

```
#!/bin/sh
IDENT=`echo $0 | sed -e 's/^.*-\(.*\)$/\1/'`
CALLEDAS="$IDENT"
TTY=`tty`

if [ x$IDENT = xdialup ]; then
        IDENT=`basename $TTY`
fi

echo "PPP for $CALLEDAS on $TTY"
echo "Starting PPP for $IDENT"

exec /usr/sbin/ppp -direct $IDENT
```

□个脚本要有可□行属性。 然后通□如下命令□建一个指向此脚本且名□ ppp-dialup的符号□接：

```
# ln -s ppp-shell /etc/ppp/ppp-dialup
```

□□□将□个脚本作□所有□入用□的 *shell*。 以下是在文件 /etc/passwd 中□于 PPP 用□ pchilds 的例子 (切□，

不要直接修改□个密□文件， 用 vipw(8) 来修改它)。

```
pchilds:*:1011:300:Peter Childs PPP:/home/ppp:/etc/ppp/ppp-dialup
```

□建一个名□ /home/ppp 的目□作□□入用□的主目□， 其中包含以下□些空文件：

```
-r--r--r--   1 root     wheel           0 May 27 02:23 .hushlogin
-r--r--r--   1 root     wheel           0 May 27 02:22 .rhosts
```

□□就可以防止/etc/motd被□示出来。

28.2.1.2.7. 静□IP用□的Shell

像上面那□□建ppp-shell文件， □□个静□分配IP用□□建一个到 ppp-shell的 符号□接。

例如， 如果□希望□三个□号用□， fred， sam， 和 mary 路由 /24 CIDR 的网□， □需要□入以下内容：

```
# ln -s /etc/ppp/ppp-shell /etc/ppp/ppp-fred
# ln -s /etc/ppp/ppp-shell /etc/ppp/ppp-sam
# ln -s /etc/ppp/ppp-shell /etc/ppp/ppp-mary
```

□个用□的Shell必□被□成一个符号□接(例如用□ mary的Shell□□是/etc/ppp/ppp-mary)。

28.2.1.2.8. □□□IP用□□置ppp.conf

/etc/ppp/ppp.conf文件□□包含下面 □些行：

```
default:
  set debug phase lcp chat
  set timeout 0

ttyu0:
  set ifaddr 203.14.100.1 203.14.100.20 255.255.255.255
  enable proxy

ttyu1:
  set ifaddr 203.14.100.1 203.14.100.21 255.255.255.255
  enable proxy
```

□□得必□的。

default:□在□次会□□都会加□。 □个在 /etc/ttys 中□用的行都必□□其□建一个相似于 ttyu0: 的□。 □一行□□从□□ IP 地址池中取得唯一的IP地址。

28.2.1.2.9. □静□ IP 用□配置 ppp.conf

根据上面 /usr/shared/examples/ppp/ppp.conf 文件的内容， □必□□□个静□□号用□添加一个□。 我□□□以 fred、 sam 以及 mary□例。

```
fred:
  set ifaddr 203.14.100.1 203.14.101.1 255.255.255.255

sam:
  set ifaddr 203.14.100.1 203.14.102.1 255.255.255.255

mary:
  set ifaddr 203.14.100.1 203.14.103.1 255.255.255.255
```

如果需要， /etc/ppp/ppp.linkup 也□□包括□个静□IP用□的的路由信息。 下面□一行□客□□接添加了到 203.14.101.0/24 网□的路由。

```
fred:
  add 203.14.101.0 netmask 255.255.255.0 HISADDR

sam:
  add 203.14.102.0 netmask 255.255.255.0 HISADDR

mary:
  add 203.14.103.0 netmask 255.255.255.0 HISADDR
```

28.2.1.2.10. mgetty和AutoPPP

默□情况下， comms/mgetty+sendfax port 在□□□□用了 AUTO_PPP □□， 它使 mgetty 能□□□ PPP □接的 LCP 状□， 并自□□生 PPP shell。 不□， 由于在默□配置中的 login/password 序列并不出□， 因此， 就必□使用 PAP 或 CHAP 来□重用□身□。

□□假定用□已□在系□中成功地□□并安装了 comms/mgetty+sendfax。

□□□的 /usr/local/etc/mgetty+sendfax/login.config 文件中包含以下内容：

```
/AutoPPP/ -     -             /etc/ppp/ppp-pap-dialup
```

□行告□mgetty□行 ppp-pap-dialup脚本来□听PPP□接。

□建/etc/ppp/ppp-pap-dialup文件写入以下内容 (此文件□□是可□行的)：

```
#!/bin/sh
exec /usr/sbin/ppp -direct pap$IDENT
```

□□于□个在/etc/ttys的□用行， 都要在/etc/ppp/ppp.conf 中□建相□的□。 □和上面的定□是相同的。

```
pap:
  enable pap
  set ifaddr 203.14.100.1 203.14.100.20-203.14.100.40
  enable proxy
```

□个以□□方式登□的用□， 都必□在 /etc/ppp/ppp.secret 文件中□出用□名/口令， 或者使用以下□□， 来通□ PAP 方式以 /etc/passwd 文件提供的信息来完成身□□□。

```
enable passwdauth
```

如果□想□某些用□分配静□IP， 可以在 /etc/ppp/ppp.secret 中将IP号作□第三个参数指定。 □参□ /usr/shared/examples/ppp/ppp.secret.sample 中的例子。

28.2.1.2.11. MS Extensions

可以配置PPP以提供DNS和NetBIOS域名服□器地址。

要在 PPP 1.x 版本中□用□些□展， 需要在 /etc/ppp/ppp.conf 的□□□中加入下列配置：

```
enable msext
set ns 203.14.100.1 203.14.100.2
set nbns 203.14.100.5
```

PPP版本2及以上：

```
accept dns
set dns 203.14.100.1 203.14.100.2
set nbns 203.14.100.5
```

□将告□客□端首□域名服□器和□用域名服□器。

在版本2及以上版本中， 如果省略了 `set dns`， PPP会使用 /etc/resolv.conf中的□。

28.2.1.2.12. PAP 和 CHAP □□

一些 ISP 将系□配置□使用 PAP 或 CHAP 机制来完成□接□□。 如果遇到□□情况， 在□□接□ ISP 就不会看到 `login:` 提示符， 而是立即□始 PPP □□。

PAP 安全性要比 CHAP 差一些， 但在□里安全性并不是□□， 因□密□ (即使用明文□送) 只是通□串行□□送， 攻□者并没有太多机会去 "窃听" 它。

参考 PPP 与静□ IP 地址 或 PPP 与□□ IP 地址 小□， 并完成下列改□：

```
13      set authname MyUserName
14      set authkey MyPassword
15      set login
```

第 **13** 行：

□一行指明□的PAP/CHAP用□名。 □需要□_MyUserName_□入正□的□。

第 **14** 行：

□一行指明□的 PAP/CHAP password密□。 □需要□ *MyPassword* □入正□的□。 □外，□可能希望加入一些□外的□□，例如：

```
16      accept PAP
```

或

```
16      accept CHAP
```

以明□□的意□， 不□， 默□情况下 PAP 和 CHAP 都会被接受。

行 **15**：

如果□使用的是 PAP 或 CHAP， 一般来□ ISP 就不会要求□登□服□器了。 □□， 就必□禁用 "set login" □置。

28.2.1.2.13. 即□改□□的ppp 配置

与后台□行的ppp程序□行□□是可能的， 前提是□置了一个合□的□断端口。 做到□一点，需要把下面的行加入到□的配置中：

```
set server /var/run/ppp-tun%d DiagnosticPassword 0177
```

□行告□ PPP在指定的UNIX®域socket中□听， 当用□□接□需要□出指定的密□。 %d用tun□□号替□。

一旦□用了socket， 就可以在脚本中□用程序pppctl(8)来□理正在□行的 的PPP。

28.2.1.3. 使用PPP网□地址翻□

PPP 可以使用内建的 NAT， 而无需内核支持。 □可以在 /etc/ppp/ppp.conf 中加入如下配置来□用它：

```
nat enable yes
```

PPP NAT也可以使用命令行□□ -nat□□。 在 /etc/rc.conf 文件中也有 ppp_nat □， 并默□□用。

如果□使用了□个特性， □□会□□在 /etc/ppp/ppp.conf中以下 □□□于□用incoming connections forwarding是有用的：

```
nat port tcp 10.0.0.2:ftp ftp
nat port tcp 10.0.0.2:http http
```

或者完全不信任外来的□求

```
nat deny_incoming yes
```

28.2.1.4. 最后的系□配置

□在□已配置了ppp， 但在真正工作之前□有一些事情要做。 即修改 /etc/rc.conf。

从上依次往下看， □□已□正□地配置了 hostname=， 例如：

```
hostname="foo.example.com"
```

如果□的ISP提供□□一个静□的IP和名字， 将□个名字□□hostname是最合□的。

□□ network_interfaces □量。 如果要配置系□通□□号□入ISP， 一定要将tun0□□加入□个列表， 否□就□除它。

```
network_interfaces="lo0 tun0"
ifconfig_tun0=
```

ifconfig_tun0□量□□是空的， 且要□建一个名□ /etc/start_if.tun0的文件。 □个文件□□包含□一行：

```
ppp -auto mysystem
```

此脚本在网□配置□被□行， □□PPP守□□程□入自□模式。 如果□台机子充当一个LAN的网□，□可能希望使用 -alias。 参考相□□机手册了解更多□□。

□必在 /etc/rc.conf 中， 把路由程序□置□ NO：

```
router_enable="NO"
```

不□□ routed 服□程序非常重要， 因□ routed □会□掉由 ppp 所建立的默□路由。

此外， 我□建□□□□一下 sendmail_flags □一行中没有指定 -q 参数， 否□ sendmail 将会不断地□□□□网□， 而□□做将会□致机器不断地□行□号。 可以考□：

```
sendmail_flags="-bd"
```

替代的做法是当□次 PPP □接建立□□必□通□□入以下命令□制 sendmail 重新□□□件□列：

```
# /usr/sbin/sendmail -q
```

□也可以在ppp.linkup使用!bg命令自□完成□些工作：

```
1     provider:
2       delete ALL
3       add 0 0 HISADDR
4       !bg sendmail -bd -q30m
```

如果□不喜□□□做， 可以□立一个 "dfilter" 以阻止 SMTP □□。 参考相□文件了解更多□□。

□在□唯一要做的事是重新□□□算机。 重□之后，可以□入：

```
# ppp
```

然后是dial provider以□□ PPP会□。 或者如果□想□ppp自□建立会□， 因□□有一条广域网□接 (且没有□建 start_if.tun0 脚本)， □入：

```
# ppp -auto provider
```

28.2.1.5. □□

当第一次□置PPP□， 下面几□是必□的：

客□端：

1. □保 tun□□□了□核。
2. □保 /dev 目□中名□ tunN 的□□文件是可用的。
3. 在 /etc/ppp/ppp.conf中□建一个□。 pmdemand示例□□□合于□大多数ISP。
4. 如果□使用□□IP地址， 在/etc/ppp/ppp.linkup□建一个□。
5. 更新/etc/rc.conf 文件。
6. 如果□要求按需□号， □建一个start_if.tun0脚本。

服□器端：

1. □保tun□□已□□入内核。
2. □保 /dev 目□中名□ tunN 的□□文件是可用的。
3. 在/etc/passwd中□建一个□ (使用vipw(8)程序)。
4. 在用□的home目□□建一个□行 `ppp -direct direct-server`或相似命令的profile。
5. 在/etc/ppp/ppp.conf中□建一个□。 direct-server示例□□能□足要求。
6. 在 /etc/ppp/ppp.linkup中□建一个□。
7. 更新 /etc/rc.conf 文件。

28.3. 使用内核□PPP

□□内容只在 FreeBSD 7.X 上可用。

28.3.1. □立内核□PPP

在□始配置 PPP 之前， □□□ pppd 已□存放在 /usr/sbin 中， 并且 /etc/ppp 目□是存在的。

pppd能在□□模式下工作：

1. 作□一个 "客□" - □要通□PPP串行□或modem□把□的机器□接到互□网上。
2. 作□"服□器" -□算机已□位于网□上， 且被用于通□PPP与其它□算机□接。

□□情况□都需要□立一个□□文件， (/etc/ppp/options 或者是 ~/.ppprc 如果□的□算机有多个用□使用PPP)。

□□需要一些modem/serial□件(comms/kermit就很□合)， 使□能□□号并与□程主机建立□接。

28.3.2. 使用pppd作□客□端

下面□个 /etc/ppp/options□□文件能□被用来与CISCO□端服□器的 PPP□□接。

```
crtscts         # enable hardware flow control
modem           # modem control line
noipdefault     # remote PPP server must supply your IP address
                # if the remote host does not send your IP during IPCP
                # negotiation, remove this option
passive         # wait for LCP packets
domain ppp.foo.com      # put your domain name here

:remote_ip    # put the IP of remote PPP host here
                # it will be used to route packets via PPP link
                # if you didn't specified the noipdefault option
                # change this line to local_ip:remote_ip

defaultroute    # put this if you want that PPP server will be your
                # default router
```

□接：

1. 使用 Kermit (或其他 modem 程序来□号)， 然后□入□的用□名和口令 (或在□程主机上□用 PPP 所需的其他信息)。
2. 退出 Kermit (并不挂断□接)。
3. □入下面□行：

   ```
   # /usr/sbin/pppd /dev/tty01 19200
   ```

   一定要使用正□的速度和□□名。

□在□的□算机已□用PPP□接。 如果□接失□， □可在文件 /etc/ppp/options 中添加 debug □□， 并□看控制台信息以跟踪□□。

下面□个/etc/ppp/pppup脚本能自□完成□三个□□：

```
#!/bin/sh
pgrep -l pppd
pid=`pgrep pppd`
if [ "X${pid}" != "X" ] ; then
        echo 'killing pppd, PID=' ${pid}
        kill ${pid}
fi
pgrep -l kermit
pid=`pgrep kermit`
if [ "X${pid}" != "X" ] ; then
        echo 'killing kermit, PID=' ${pid}
        kill -9 ${pid}
fi

ifconfig ppp0 down
ifconfig ppp0 delete

kermit -y /etc/ppp/kermit.dial
pppd /dev/tty01 19200
```

/etc/ppp/kermit.dial 是一个 Kermit 脚本， 它会完成□号， 并在□程主机上完成所有需要的身□□□□程 (□□文□的最后有一个脚本□例)。

使用下面□个脚本/etc/ppp/pppdown断□PPP□□：

```
#!/bin/sh
pid=`pgrep pppd`
if [ X${pid} != "X" ] ; then
        echo 'killing pppd, PID=' ${pid}
        kill -TERM ${pid}
fi

pgrep -l kermit
pid=`pgrep kermit`
if [ "X${pid}" != "X" ] ; then
        echo 'killing kermit, PID=' ${pid}
        kill -9 ${pid}
fi

/sbin/ifconfig ppp0 down
/sbin/ifconfig ppp0 delete
kermit -y /etc/ppp/kermit.hup
/etc/ppp/ppptest
```

通□□行/usr/etc/ppp/ppptest， 看看pppd 是否仍在□行：

```
#!/bin/sh
pid=`pgrep pppd`
if [ X${pid} != "X" ] ; then
        echo 'pppd running: PID=' ${pid-NONE}
else
        echo 'No pppd running.'
fi
set -x
netstat -n -I ppp0
ifconfig ppp0
```

□行脚本 /etc/ppp/kermit.hup以挂起moderm， □个文件包含：

```
set line /dev/tty01 ; put your modem device here
set speed 19200
set file type binary
set file names literal
set win 8
set rec pack 1024
set send pack 1024
set block 3
set term bytesize 8
set command bytesize 8
set flow none

pau 1
out +++
inp 5 OK
out ATH0\13
echo \13
exit
```

也可以用chat 代替kermit：

以下□个文件用以建立pppd□接。

/etc/ppp/options：

```
/dev/cuad1 115200

crtscts     # enable hardware flow control
modem       # modem control line
connect "/usr/bin/chat -f /etc/ppp/login.chat.script"
noipdefault # remote PPP serve must supply your IP address
            # if the remote host doesn't send your IP during
                # IPCP negotiation, remove this option
passive         # wait for LCP packets
domain your.domain  # put your domain name here

:       # put the IP of remote PPP host here
            # it will be used to route packets via PPP link
                # if you didn't specified the noipdefault option
                # change this line to local_ip:remote_ip

defaultroute    # put this if you want that PPP server will be
            # your default router
```

/etc/ppp/login.chat.script：

以下的内容□□放在一行内。

```
ABORT BUSY ABORT 'NO CARRIER' "" AT OK ATDTphone.number
  CONNECT "" TIMEOUT 10 ogin:-\\r-ogin: login-id
  TIMEOUT 5 sword: password
```

一旦□些被安装且修改正□，□所要做的就是□行pppd，就像□□：

```
# pppd
```

28.3.3. 使用pppd作□服□器

/etc/ppp/options要包括下面□些内容：

```
crtscts                         # Hardware flow control
netmask 255.255.255.0           # netmask (not required)
192.114.208.20:192.114.208.165  # IP's of local and remote hosts
                                # local ip must be different from one
                                # you assigned to the Ethernet (or other)
                                # interface on your machine.
                                # remote IP is IP address that will be
                                # assigned to the remote machine
domain ppp.foo.com              # your domain
passive                         # wait for LCP
modem                           # modem line
```

下面□个脚本/etc/ppp/pppserv 使pppd以服□器方式□□：

```
#!/bin/sh
pgrep -l pppd
pid=`pgrep pppd`
if [ "X${pid}" != "X" ] ; then
        echo 'killing pppd, PID=' ${pid}
        kill ${pid}
fi
pgrep -l kermit
pid=`pgrep kermit`
if [ "X${pid}" != "X" ] ; then
        echo 'killing kermit, PID=' ${pid}
        kill -9 ${pid}
fi

# reset ppp interface
ifconfig ppp0 down
ifconfig ppp0 delete

# enable autoanswer mode
kermit -y /etc/ppp/kermit.ans

# run ppp
pppd /dev/tty01 19200
```

使用脚本/etc/ppp/pppservdown停止服□器：

```
#!/bin/sh
pgrep -l pppd
pid=`pgrep pppd`
if [ "X${pid}" != "X" ] ; then
        echo 'killing pppd, PID=' ${pid}
        kill ${pid}
fi
pgrep -l kermit
pid=`pgrep kermit`
if [ "X${pid}" != "X" ] ; then
        echo 'killing kermit, PID=' ${pid}
        kill -9 ${pid}
fi
ifconfig ppp0 down
ifconfig ppp0 delete

kermit -y /etc/ppp/kermit.noans
```

下面的 Kermit 脚本 (/etc/ppp/kermit.ans) 能□□用/禁用□ modem 的自□□答模式。 其内容□似下面□□：

```
set line /dev/tty01
set speed 19200
set file type binary
set file names literal
set win 8
set rec pack 1024
set send pack 1024
set block 3
set term bytesize 8
set command bytesize 8
set flow none

pau 1
out +++
inp 5 OK
out ATH0\13
inp 5 OK
echo \13
out ATS0=1\13   ; change this to out ATS0=0\13 if you want to disable
                ; autoanswer mode
inp 5 OK
echo \13
exit
```

一个名□/etc/ppp/kermit.dial的脚本用于向□程主机　□行□号和□□。　□要根据需要定制它。　要加入□的登□名和密□，□□要根据 modem 和□程主机的反□修改□入□句。

```
;
; put the com line attached to the modem here:
;
set line /dev/tty01
;
; put the modem speed here:
;
set speed 19200
set file type binary            ; full 8 bit file xfer
set file names literal
set win 8
set rec pack 1024
set send pack 1024
set block 3
set term bytesize 8
set command bytesize 8
set flow none
set modem hayes
set dial hangup off
set carrier auto                ; Then SET CARRIER if necessary,
set dial display on             ; Then SET DIAL if necessary,
set input echo on
```

```
set input timeout proceed
set input case ignore
def \%x 0                       ; login prompt counter
goto slhup

:slcmd                          ; put the modem in command mode
echo Put the modem in command mode.
clear                           ; Clear unread characters from input buffer
pause 1
output +++                      ; hayes escape sequence
input 1 OK\13\10                ; wait for OK
if success goto slhup
output \13
pause 1
output at\13
input 1 OK\13\10
if fail goto slcmd              ; if modem doesn't answer OK, try again

:slhup                          ; hang up the phone
clear                           ; Clear unread characters from input buffer
pause 1
echo Hanging up the phone.
output ath0\13                  ; hayes command for on hook
input 2 OK\13\10
if fail goto slcmd              ; if no OK answer, put modem in command mode

:sldial                         ; dial the number
pause 1
echo Dialing.
output atdt9,550311\13\10               ; put phone number here
assign \%x 0                    ; zero the time counter

:look
clear                           ; Clear unread characters from input buffer
increment \%x                   ; Count the seconds
input 1 {CONNECT }
if success goto sllogin
reinput 1 {NO CARRIER\13\10}
if success goto sldial
reinput 1 {NO DIALTONE\13\10}
if success goto slnodial
reinput 1 {\255}
if success goto slhup
reinput 1 {\127}
if success goto slhup
if < \%x 60 goto look
else goto slhup

:sllogin                        ; login
assign \%x 0                    ; zero the time counter
pause 1
```

```
echo Looking for login prompt.

:slloop
increment \%x                   ; Count the seconds
clear                           ; Clear unread characters from input buffer
output \13
;
; put your expected login prompt here:
;
input 1 {Username: }
if success goto sluid
reinput 1 {\255}
if success goto slhup
reinput 1 {\127}
if success goto slhup
if < \%x 10 goto slloop         ; try 10 times to get a login prompt
else goto slhup                 ; hang up and start again if 10 failures

:sluid
;
; put your userid here:
;
output ppp-login\13
input 1 {Password: }
;
; put your password here:
;
output ppp-password\13
input 1 {Entering SLIP mode.}
echo
quit

:slnodial
echo \7No dialtone.  Check the telephone line!\7
exit 1

; local variables:
; mode: csh
; comment-start: "; "
; comment-start-skip: "; "
; end:
```

28.4. PPP □接故障排除

从 FreeBSD 8.0 □始， uart(4) □□取代了 sio(4) □□。 用以表示串口的□□□点由分□ /dev/cuadN 改□了 /dev/cuauN， 并从 /dev/ttydN 改□了 /dev/ttyuN。 FreeBSD 7.X 用□在升□□需要因□之□配置文件□行必要的更改。

本□将□述通□modem□接使用PPP□可能出□的□□。 例如， □可能需要□切地知道□□入的系□会出□一个

□□的命令行提示符。 有些 ISP 会提供 ssword提示符， 而其它的可能会出□ password；如果没有根据情况的不同相□地□写 ppp 脚本， 登□就会失□。 □断 ppp 最常用的方法是手□□行□接。以下的信息会一□一□地□□完成手□□接。

28.4.1. □□□□□点

如果使用的是定制内核， □□在其□□配置中包含下列配置：

```
device   uart
```

默□的 GENERIC 内核中包含了 uart □□， 因此如果□使用的是它的□， 就不需要担心了。 只要□看 dmesg □出中是否有 modem □□：

```
# dmesg | grep uart
```

□□□□到与 uart □□有□的□出。 □些就是我□需要的 COM 端口。 如果□的 modem 按照□准串行端口工作，□就会在 uart1 或 COM2 上□到它。 如果 modem □□□接在 uart1 接口 (在 DOS 中称□COM2)， 那□□的 modem 将会是 /dev/cuau1。

28.4.2. 手□□接

通□手□控制ppp来□接Internet 是□断□接及□知ISP□理PPP客□端方式的一个快速， □□的方法。 □我□从PPP命令行□始， 在所有的例子中我□使用 *example* 表示□行 PPP 服□的主机名。 □入ppp 命令打□ ppp：

```
# ppp
```

□在我□已□打□了ppp。

```
ppp ON example> set device /dev/cuau1
```

□置modem□□， 在本例子中是 cuau1。

```
ppp ON example> set speed 115200
```

□置□接速度， 在本例中我□使用15,200 kbps。

```
ppp ON example> enable dns
```

使ppp配置域名服□， 在文件/etc/resolv.conf中添加域名服□器行。 如果 ppp不能□定我□的主机名， 可以在□后□置。

```
ppp ON example> term
```

切□到 "□端"□我□就能手□地控制□台 modem 的模式。

```
deflink: Entering terminal mode on /dev/cuau1
type '~h' for help
```

```
at
OK
atdt123456789
```

使用命令at初始化modem， 然后使用atdt和ISP□□的号□□行□号。

```
CONNECT
```

□接配置， 如果我□遇到了与硬件无□的□接□□， 可以在□里□□解决。

```
ISP Login:myusername
```

□里提示□□入用□名， □入ISP提供的用□名然后按回□。

```
ISP Pass:mypassword
```

□□提示我□□入密□， □入 ISP提供的密□。 如同登□入FreeBSD， 密□不会□示。

```
Shell or PPP:ppp
```

由于ISP的不同， □个提示符可能不会出□。 □里我□需要考□： 是使用□行于提供商端的 Shell， □是□□ ppp？ □本例中， 我□□□使用 ppp， 因□我□希望得到 Internet □接。

```
Ppp ON example>
```

注意在□个例子中， 第一个 p已□大写。 □表示我□已□成功地□接上了 ISP。

```
PPp ON example>
```

我□已□成功通□了 ISP的□□， 正在等待分配IP地址。

```
PPP ON example>
```

我□得到了一个 IP 地址， 成功地完成了□接。

```
PPP ON example>add default HISADDR
```

□□就完成了添加默□路由所需的配置。 □是与外界通信所必需的。 因□之前我□只是与服□器端建立了□接。 如果由于已存在的路由而□致操作失□， □可以在 add 前加 !号。 除此之外， □也可以在真正□接之前□置□些 (指 add default HISADDR)， ppp 会根据□□□定□商取得新的路由。

如果一切□利， □在我□□□能得到一个活□的 Internet □接， 可以使用 CTRL + z 使其□入后台。 如果□□□ PPP 重新□□ ppp， □表示□接被断□。 大写的 P 表明建立了到 ISP 的□接， 而小写的 p □表示□接由于某□原因被断□， □有助于□助我□了解□接的状□。 ppp 只有□□个状□。

28.4.2.1. □断排□

如果□有一根直□□且似乎不能建立□接， 要使用set ctsrts off以□□字□流的CTS/RTS。 □□情况一般□生在□接兼容 PPP 的□端服□器□。 当它向通信□接写入数据□， PPP就会挂起， 一直等待一个CTS，或者一个不可能出□的 Clear to Send 信号。 如果使用了□个□□， □□□使用 set accmap□□，某些存在缺陷的硬件在完成端□端□送特定字符， 特□是 XON/XOFF □可能会遇到困□。 □参□ ppp(8) □机手册以了解□于可用□□的更多□□， 以及如何使用它□。

如果□的 modem 比□旧， 就需要使用 set parity even 了。 奇偶校□的默□□置是 none， 但在旧式的 (当流量大量□加□) □制解□器和某些 ISP 被用来□□。 □需要使用□个□□才能使用 Compuserve ISP。

PPP 可能并不返回命令模式， □通常是 ISP 等待□□一端□起□商□□生了□□。 此□， 使用 ~p 命令将□制 ppp □始□送配置信息。

如果□没有看到登□提示， □很可能需要使用 PAP 或 CHAP □□来代替前面例子中的 UNIX® □格□□。 要使用 PAP 或 CHAP 只需在□入□端模式之前把下面的□□加入 PPP：

```
ppp ON example> set authname myusername
```

此□ *myusername* □改□□的 ISP 分配□□的用□名。

```
ppp ON example> set authkey mypassword
```

此□ *mypassword* □□□□的 ISP 分配□□的口令。

如果□接正常， 但无法□□域名， □□□ ping(8) 某个 IP 地址来看看是否返回了信息。 如果□□□百分之百 (100%) □包， 那□□很可能没有分配默□路由。 □仔□□□□□ add default HISADDR 是否在□接□被□置了。 如果□能□接到□程的 IP 地址□有可能域名解析服□器的地址没有被加入到 /etc/resolv.conf。 □个文件□□是下面的□子：

```
domain example.com
nameserver x.x.x.x
nameserver y.y.y.y
```

此□ *x.x.x.x* 和 *y.y.y.y* □□改□□的 ISP 的 DNS 服□器的 IP 地址。 □一信息在□注册□可能会提供□□， 不□通常只需□ ISP 打个□□就能知道了。

□□可以□ syslog(3) □□的 PPP □接提供日志。 只需□加：

```
!ppp
*.*     /var/log/ppp.log
```

到 /etc/syslog.conf 中。 □大多数情况下， □个功能默□已□打□了。

28.5. 使用基于以太网的PPP(PPPoE)

本□将介□如何建立基于以太网的PPP (PPPoE)。

28.5.1. 配置内核

□于PPPOE， 并没有必□的内核配置。 如果必需的 netgraph 支持没有□□入内核， 它可以由 ppp □□加□。

28.5.2. □置ppp.conf

以下是一个ppp.conf的例子：

```
default:
  set log Phase tun command # you can add more detailed logging if you wish
  set ifaddr 10.0.0.1/0 10.0.0.2/0

name_of_service_provider:
  set device PPPoE:xl1 # replace xl1 with your Ethernet device
  set authname YOURLOGINNAME
  set authkey YOURPASSWORD
  set dial
  set login
  add default HISADDR
```

28.5.3. □行ppp

以 root 身□□行：

```
# ppp -ddial name_of_service_provider
```

28.5.4. □□□□行ppp

在 /etc/rc.conf 中加入以下内容：

```
ppp_enable="YES"
ppp_mode="ddial"
ppp_nat="YES"   # if you want to enable nat for your local network, otherwise NO
ppp_profile="name_of_service_provider"
```

28.5.5. 使用 PPPoE 服□□□

在某些□候， 有必要使用一个服□□□来建立□的□接。 服□□□用于区分同一网□中的不同服□器。

□可以在ISP提供的文□中□到必要的服□□□信息。 若不能□到， □□向□的 ISP □求技□支持。

作□最后的方法， □可以□□ Roaring Penguin PPPoE， 它可以在 Ports Collection 中□到。 然而需要注意的是， 它可能会清楚 modem 的固件， 并使其无法正常工作， 因此一定要仔□考□之后再做□个操作。 □□地安装由服□提供商随 modem 提供的程序。 随后， □□ **System** 菜□。 □的配置文件□□会在□里列出。 一般来□它的名字□□是 *ISP*。

配置文件名 (service tag， 服□□□) 将被用于 PPPoE 在 ppp.conf 中的配置□， 作□服□商 `set device` 命令的一部分 (参□ ppp(8) □机手册以了解更多□□)。 它□□□似下面的□子：

```
set device PPPoE:xl1:ISP
```

□住将_xl1_□成□□的以太网□□。

□住将 *ISP* □成□□□□到的profile名。

□得更多的信息， □参考：

- Cheaper Broadband with FreeBSD on DSL by Renaud Waldura.
- Nutzung von T-DSL und T-Online mit FreeBSD by Udo Erdelhoff (in German).

28.5.6. □有一个3Com® HomeConnect™ ADSL Modem的PPPOE双重□接

□个 modem 不遵循 RFC 2516 (*A Method for transmitting PPP over Ethernet (PPPoE)*， 其作者□ L. Mamakos、 K. Lidl、 J. Evarts、 D. Carrel、 D. Simone 以及 R. Wheeler)。 而是使用不同的数据包格式作□以太网的框架。 □向 3Com 抱怨， 如果□□□它□□遵守 PPPoE 的□□。

□了□FreeBSD能□与□个□□通信， 必□□置sysctl。 通□更改/etc/sysctl.conf， □一□可以在□□□自□完成：

```
net.graph.nonstandard_pppoe=1
```

或者， 也可以直接□行下面的命令：

```
# sysctl net.graph.nonstandard_pppoe=1
```

很不幸，由于□是系□全局□置， 无法同□与正常的PPP客□端(或服□器) 和3Com®HomeConnect™ ADSL Modem通信。

28.6. 使用 ATM 上的 PPP (PPPoA)

以下将介□如何□置基于ATM的PPP(PPPoA)。 PPPoA是欧洲DSL提供商的普遍□□。

28.6.1. 使用 Alcatel SpeedTouch™USB 的 PPPoA

□□□一□□的 PPPoA 支持， 在 FreeBSD 中是作□ port 提供的， 因□其固件使用了 阿□□特□可□□， 因而不能与 FreeBSD 的基本系□一起免□地再□布。

使用 Ports 套件 可以非常方便地安装 net/pppoa port， 之后按照它提供的指示操作就可以了。

和□多 USB □□□似， 阿□□特的 SpeedTouch™ USB 需要从主机上下□固件才能□正常工作。 在 FreeBSD 中□可以将此操作自□化， 在有□□□到某个 USB 口的□候自□下□固件。 可以在 /etc/usbd.conf 文件中加入下面的信息来□它自□完成固件的□送。 注意， 必□以 root 用□的身□□它。

```
device "Alcatel SpeedTouch USB"
    devname "ugen[0-9]+"
    vendor 0x06b9
    product 0x4061
    attach "/usr/local/sbin/modem_run -f /usr/local/libdata/mgmt.o"
```

要□□USB守□□程usbd， 在/etc/rc.conf加入以下行：

```
usbd_enable="YES"
```

也可以将ppp□置成□□□□号。 向 /etc/rc.conf加入以下□几行。 同□地□需要以root用□登□。

```
ppp_enable="YES"
ppp_mode="ddial"
ppp_profile="adsl"
```

□了使其正常工作， □需要使用net/pppoa port提供的ppp.conf□例。

28.6.2. 使用mpd

可以使用 mpd 来□接多□□型的服□， 特□是 PPTP 服□。 □可以在 Ports Collection 中□到 mpd， 它的位置是 net/mpd。 □多 ADSL modem 需要在 modem 和□算机之□建立一条 PPTP 隧道， 而阿□□特 SpeedTouch™ Home 正是其中的一□。

首先需要从 port 完成安装， 然后才能配置 mpd 来□足□的需要， 并完成服□商的配置。 port 会把一系列包括了□□注解的配置文件□例放到 PREFIX/etc/mpd/。 注意， □里的 *PREFIX* 表示 ports 安装的目□， 默□情况下， □□是 /usr/local/。 □于配置 mpd 的完整□明， 会以 HTML 格式随 port 一起安装。 □些文件将放在 PREFIX/shared/doc/mpd/。 下面是通□ mpd □接 ADSL 服□的一个□□例子。 配置被分□放到了□个文件中， 第一个是 mpd.conf：

```
default:
    load adsl

adsl:
    new -i ng0 adsl adsl
    set bundle authname username ①
    set bundle password password ②
    set bundle disable multilink

    set link no pap acfcomp protocomp
    set link disable chap
    set link accept chap
    set link keep-alive 30 10

    set ipcp no vjcomp
    set ipcp ranges 0.0.0.0/0 0.0.0.0/0

    set iface route default
    set iface disable on-demand
    set iface enable proxy-arp
    set iface idle 0

    open
```

① username用来向□的ISP□行□□。

② password用来向□的ISP□行□□。

mpd.links包含□接的信息：

```
adsl:
    set link type pptp
    set pptp mode active
    set pptp enable originate outcall
    set pptp self 10.0.0.1 ①
    set pptp peer 10.0.0.138 ②
```

① □行mpd的主机的IP地址。

② ADSL modem的IP地址。 Alcatel SpeedTouch™ Home 默□的是 10.0.0.138。

初始化□接：

```
# mpd -b adsl
```

□可以通□以下命令□看□接状□：

```
% ifconfig ng0
ng0: flags=88d1<UP,POINTOPOINT,RUNNING,NOARP,SIMPLEX,MULTICAST> mtu 1500
     inet 216.136.204.117 --> 204.152.186.171 netmask 0xffffffff
```

使用mpd□接ADSL服□是推□的方式。

28.6.3. 使用pptpclient

也可以使用net/pptpclient□接其它的 PPPoA。

要使用 net/pptpclient □接 DSL 服□， 需要安装 port 或 package 并□□ /etc/ppp/ppp.conf。 □需要有 root □限才能完成□□□操作。 以下是 ppp.conf 中的一个示例□。 参考 ppp 的□机手册 ppp(8)， 以了解更多有□ ppp.conf □□的信息。

```
adsl:
 set log phase chat lcp ipcp ccp tun command
 set timeout 0
 enable dns
 set authname username ①
 set authkey password ②
 set ifaddr 0 0
 add default HISADDR
```

① □在 DSL 服□提供商那里的用□名

② □□□的口令。

由于□必□将□号密□以明文的方式放入ppp.conf □□□□保没有任何人能看到此文件的内容。 以下一系列命令将会□保此文件只□ root用□可□。 □参□ chmod(1) 和 chown(8) 的 □机手册以了解有□如何操作的□一□信息。

```
# chown root:wheel /etc/ppp/ppp.conf
# chmod 600 /etc/ppp/ppp.conf
```

以下将□到 DSL 路由器的会□打□一个 tunnel。 以太网DSL modem有一个□置的局域网IP地址。 以 Alcatel SpeedTouch™ Home □例， □个地址是 10.0.0.138。 路由器的文□□□会告□□它使用的地址。 □行以下命令以打□ tunnel 并□始会□：

```
# pptp address adsl
```

□□□在命令的最后加上("&")号， 否□ pptp 无法返回到命令行提示符。

要□建一个 tun虚□□□用于□程pptp 和ppp 之□的交互。 一旦□回到了命令行， 或者 pptp □程□□了一个□接， □可以□□□□tunnel□□：

```
% ifconfig tun0
tun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1500
        inet 216.136.204.21 --> 204.152.186.171 netmask 0xffffff00
        Opened by PID 918
```

如果□无法□接， 一般可以通□telnet或者web□□器□□路由器(modem)的配置。 如果依旧无法□接， □□□□□ pptp的□出及ppp的日志文件 /var/log/ppp.log 以□得□索。

28.7. 使用SLIP

□□内容只在 FreeBSD 7.X 上可用。

28.7.1. □置 SLIP 客□端

下面是在静□主机网□上配置 FreeBSD 机器使用 SLIP 的方法。 □于□□主机名分配 (□的地址会随□次□号而不同)， □可能需要□□□一些的□置。

首先， □需要□□□制解□器所□接的串口。 □多人会□置一个符号□接， 例如 /dev/modem， 用以指向□□的□□名， 如 /dev/cuadN。 □□□就可以□□□的□□名□行抽象， 以□□制解□器□到其他串口□方便□整之用。 不然，修改 /etc 和遍布于系□中的 .kermrc 文件将是一件很麻□的事情！

/dev/cuad0 □□ COM1， 而 /dev/cuad1 □□□ COM2， 等等。

□保□的内核文件包含以下内容：

```
device   sl
```

□包含在GENERIC内核， 所以□□□不会是个□□， 除非□ 已□□除了它。

28.7.1.1. 只需做一次的事情

1. 把□本地网□上的机器、 网□以及域名服□器， 都加入到 /etc/hosts 文件中。 我□的是下面□个□子：

```
127.0.0.1               localhost loghost
136.152.64.181          water.CS.Example.EDU water.CS water
136.152.64.1            inr-3.CS.Example.EDU inr-3 slip-gateway
128.32.136.9            ns1.Example.EDU ns1
128.32.136.12           ns2.Example.EDU ns2
```

2. □□保在□的 /etc/nsswitch.conf 中的 `hosts:` 小□里面， `files` 先于 `dns` 出□。 如果不是□□的□，可能会□生一些不希望的□象。
3. □□/etc/rc.conf。
   a. □□以下□行□置主机名(hostname)：

   ```
   hostname="myname.my.domain"
   ```

   □□用□主机的Internet全名代替。 ..

   改□□一行以指明默□的路由：

   ```
   defaultrouter="NO"
   ```

   改□：

   ```
   defaultrouter="slip-gateway"
   ```

4. □建文件/etc/resolv.conf， 写入以下内容：

```
domain CS.Example.EDU
nameserver 128.32.136.9
nameserver 128.32.136.12
```

   正如□看到的， □些行□置了域名服□器。 当然， □□的域名和IP地址取决于□的□境。

5. □置`root`和 `toor`的密□(其它任何没有密□的□号)。
6. 重□□算机， 然后□□使用了正□的主机名。

28.7.1.2. □建一个SLIP□接

1. 在命令提示符之后□入 slip □行□号， □入□的机器名和口令。 具体需要□入什□， 与□的□境密切相□。 如果使用 Kermit， □可以使用□似下面的脚本：

```
# kermit setup
set modem hayes
set line /dev/modem
set speed 115200
set parity none
set flow rts/cts
set terminal bytesize 8
set file type binary
# The next macro will dial up and login
define slip dial 643-9600, input 10 =>, if failure stop, -
output slip\x0d, input 10 Username:, if failure stop, -
output silvia\x0d, input 10 Password:, if failure stop, -
output ***\x0d, echo \x0aCONNECTED\x0a
```

当然， □□需要修改用□名和口令来□足□□需要。 完成□些操作之后， 只需在 Kermit 提示符之后□入 slip 就可以□接了。

将密□以□文本的形式存放在文件系□无□如何都是个 坏 主意。 □考□□□做的□□。

2. 在□里退出 Kermit (也可以用 Ctrl + z 将其挂起)， 以 root 用□□入：

```
# slattach -h -c -s 115200 /dev/modem
```

如果□能ping通路由器□一端的主机， 就是□接好了！ 如果不行， □可以使用-a□□代替 -c作□ slattach的参数。

28.7.1.3. □□□接

按下面的□□做：

```
# kill -INT `cat /var/run/slattach.modem.pid`
```

来□掉 slattach。 切□上述操作只有以 root 身□才能完成。 接下来回到 kermit (如果之前是将它挂起了， □使用 fg) 并退出 (q)。

在 slattach(8) □机手册中提到， 必□使用 ifconfig sl0 down 才能将接口□□□□□， 但和□□做似乎没有什□区□。 (ifconfig sl0 仍然□告同□的□西。)

有□， □的 modem 可能会拒□挂断。 □□情况下， 只需重新□□ kermit 并再次退出它就可以了。 一般来□□二次就可以了。

28.7.1.4. □□解答

如果□不行， 尽管□□件到 freebsd-net □件列表来提□。 常□的□□包括：

- □行 slattach □不使用 -c和-a□□ (□□□不是□□的， 但有些用□□告□□做解决了□□)。
- 使用s10替□ sl0 (在一些字体下很□看出不同)。
- □□ifconfig sl0来□看□的接口状□。 例如， □可以□□做：

```
# ifconfig sl0
sl0: flags=10<POINTOPOINT>
        inet 136.152.64.181 --> 136.152.64.1 netmask ffffff00
```

- 如果在使用 ping(8) □得到了 no route to host □□的提示， □□明□的路由表可能有□□。 可以用 netstat -r 命令来□示当前的路由：

```
# netstat -r
Routing tables
Destination      Gateway            Flags     Refs     Use  IfaceMTU    Rtt
Netmasks:

(root node)
(root node)

Route Tree for Protocol Family inet:
(root node) =>
default          inr-3.Example.EDU  UG          8   224515  sl0 -      -
localhost.Exampl localhost.Example. UH          5    42127  lo0 -       0.438
inr-3.Example.ED water.CS.Example.E UH          1        0  sl0 -      -
water.CS.Example localhost.Example. UGH        34 47641234  lo0 -       0.438
(root node)
```

前述的例子来自于一个非常繁忙的系□。 □系□上的□些数字会因网□活□的不同而改□。

28.7.2. □置SLIP服□器

本文提供了在 FreeBSD 上□置 SLIP 服□， 也就是如何配置□的系□， 使其能在□程 SLIP 客□端登□□自□地□□□接的建□。

28.7.2.1. 前提条件

□一□技□性很□， 所以要求□有一定的背景知□。 本□假定□熟悉 TCP/IP 网□□□， 特□是网□和□点□址、 子网掩□、 子网□分、 路由、 路由□□ (如RIP) 等知□。 在□号服□器上配置 SLIP 需要□些概念性的知□。 如果□不熟悉它□， □先□□ Craig Hunt 的 *TCP/IP* 网□管理 由O'Reilly & Associates, Inc. 出版 (ISBN 0-937175-82-X)， 或 Douglas Comer 有□ TCP/IP □□的□籍。

此外□假定□已□配置好了□的□制解□器以及相□的系□文件， 以允□通□□制解□器□行登□。 如果□□没有□此配置好系□， □参□ □入服□ 以了解□于如何□行□号服□的配置。 □可能也会想看一看 sio(4) 的□机手册，

以了解□于串口□□□□的□一□信息， 以及 ttys(5)、 gettytab(5)、 getty(8) & init(8) 上□于□□配置系□来接受来自□制解□器的登□□求的具体情况， □有 stty(1) 以了解□于□置串口参数 (例如 clocal 表示串口直□) 等。

28.7.2.2. 快速□□

使用FreeBSD作□SLIP服□器， 在典型配置□， 它是□□工作的： 一个SLIP客□□号并以□用的login ID登□到FreeBSD SLIP服□器系□。 □个用□使用 /usr/sbin/sliplogin 作□ shell。 sliplogin 程序会在文件 /etc/sliphome/slip.hosts 中□□□个用□的□， 如果□到了匹配□， 就将串行□□接到一个可用的 SLIP 接口， 然后□行 shell 脚本 /etc/sliphome/slip.login 以配置 SLIP 接口。

28.7.2.2.1. 一个SLIP服□器登□的例子

例如， 如果一个SLIP用□的ID是Shelmerg， 在/etc/master.passwd中Shelmerg的□如下的所示：

```
Shelmerg:password:1964:89::0:0:Guy Helmer -
SLIP:/usr/users/Shelmerg:/usr/sbin/sliplogin
```

Shelmerg登□□， sliplogin在文件 /etc/sliphome/slip.hosts中搜索与用□ID匹配的行;如下所示：

```
Shelmerg        dc-slip sl-helmer       0xfffffc00        autocomp
```

sliplogin□到□条区配行， 并将串行□与□一个可用的SLIP接口□起来， 然后□行 /etc/sliphome/slip.login脚本：

```
/etc/sliphome/slip.login 0 19200 Shelmerg dc-slip sl-helmer 0xfffffc00 autocomp
```

如果一切□利 /etc/sliphome/slip.login 将在 sliplogin □定的 SLIP 接口上□出 ifconfig (前述的例子中是 SLIP 接口 0， □是 slip.login 的第一个参数)， 以□置本地 IP 地址 (dc-slip)、 □程 IP 地址 (sl-helmer)、 □一 SLIP 接口的子网掩□ (0xfffffc00)， 以及任何其他□志 (autocomp)。 如果□生□□， sliplogin 通常会通□ syslogd 的 daemon facility □下有用的信息， 前者会把□些信息保存到 /var/log/messages (参□ syslogd(8) 和 syslog.conf(5) 以及 /etc/syslog.conf 的□机手册， 以了解 syslogd 在□□什□， 以及□些内容将被□在□里)。

28.7.2.3. 内核配置

FreeBSD 的默□内核 (GENERIC) 提供了 SLIP (sl(4)) 支持； 使用定制的内核□， □必□把下面的□置加入到配置文件：

```
device    sl
```

默□情况下， □的 FreeBSD □算机不会□□包。 如果□希望将 FreeBSD SLIP 服□器作□路由器使用， 就需要修改 /etc/rc.conf 文件， 将 gateway_enable □量□□ YES。 □□下次系□引□□就能□保持□一配置了。

要立即□用□些配置， 可以 root 的身□□行：

```
# /etc/rc.d/routing start
```

□参□ 配置FreeBSD的内核 以了解如何配置 FreeBSD 内核， 并□得在重新配置内核方面的指□。

28.7.2.4. Sliplogin配置

正如先前所提到的， /etc/sliphome 目□中有三个文件， 它□共同□成 /usr/sbin/sliplogin 的配置 (参考 sliplogin 的□机手册 sliplogin(8))： 用于定□ SLIP 用□和相□的 IP 地址的 slip.hosts、 通常□用于配置 SLIP 接口的 slip.login， 以及 (可□的) slip.logout， 用以撤□由 slip.login 所□行的□作。

28.7.2.4.1. 配置 slip.hosts

/etc/sliphome/slip.hosts里的□行包含至少四个元素， 元素之□由空格隔□：

- SLIP用□的登□ID
- SLIP□接的本地地址(指SLIP服□器)
- SLIP□接的□程地址
- 网□掩网

本地和□程地址可以是主机名 (通□文件/etc/hosts或者域名服□解析□IP地址， □取决于文件/etc/nsswitch.conf 中的□置)， 网□掩网可以是一个 能通□文件/etc/networks解析的名字。 在一个□例系□中， /etc/sliphome/slip.hosts是□□的：

```
#
# login local-addr      remote-addr     mask            opt1    opt2
#                                                (normal,compress,noicmp)
#
Shelmerg  dc-slip       sl-helmerg      0xfffffc00      autocomp
```

在□行末尾是一或多个□□：

- normal -不□□□□
- compress - □□□□
- autocomp -如果□程端允□， □□□□
- noicmp -禁用ICMP数据包 (□□就会□□所有的"ping"数据包， 不占用□的□□)

□SLIP□接的本地及□程地址的□□取决是□是准□在SLIP服□器上使用 TCP/IP 子网□是使用"ARP代理" (它并不是 "真正的"ARP代理， 而是我□在本□用于介□的□□)。 如果□不能□定□□何□方式或者如何分配地址， □参考 "前提条件"(前提条件)里列出的TCP/IP□籍 或者向□的IP网□管理□□教。

如果打算□□的 SLIP 客□使用一个独立的子网， 就需要先从分配得到的网□号中取出一个子网号， 然后再在 □个子网里□□个 SLIP 客□分配 IP 地址。 接下来， □□需要通□ SLIP 服□器在最近的 IP 路由器上配置一个指向 SLIP 子网的静□路由。

如果要使用 "代理 ARP" 的方式， □□需要从 SLIP 服□器的以太子网中□□个 SLIP 客□分配IP地址， □必□ 修改/etc/sliphome/slip.login 和 /etc/sliphome/slip.logout脚本以使用 arp(8)来管理在 SLIP 服□器 ARP

表中的 "代理 ARP" □。

28.7.2.4.2. slip.login Configuration

典型的/etc/sliphome/slip.login 如下所示：

```
#!/bin/sh -
#
#       @(#)slip.login  5.1 (Berkeley) 7/1/90

#
# generic login file for a slip line.  sliplogin invokes this with
# the parameters:
#      1        2         3        4          5         6     7-n
#   slipunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 inet $4 $5 netmask $6
```

□个slip.login脚本□□□□有相□本地及□程地址和掩□的SLIP接口□行 `ifconfig`。

如果□决定使用"ARP代理" 方式(而非□□的SLIP客□使用独立的子网)， □的/etc/sliphome/slip.login □□是□□：

```
#!/bin/sh -
#
#       @(#)slip.login  5.1 (Berkeley) 7/1/90

#
# generic login file for a slip line.  sliplogin invokes this with
# the parameters:
#      1        2         3        4          5         6     7-n
#   slipunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 inet $4 $5 netmask $6
# Answer ARP requests for the SLIP client with our Ethernet addr
/usr/sbin/arp -s $5 00:11:22:33:44:55 pub
```

slip.login新加的行`arp -s $5 00:11:22:33:44:55 pub` 在 SLIP 服□器的 ARP 表中加入了一个表□。 □个ARP□使得□当□个以太网上的其它 IP □点□ SLIP 客□端 IP 地址□行 ARP □求□， SLIP 服□器会以自已的以太网MAC地址作□回□。

当使用以上的例子□， 一定要将 以太网MAC地址 （`00:11:22:33:44:55`） 替□成□系□网□的MAC地址，否□"ARP代理" 将完全无法工作！ □可以□看 `netstat -i` □出□果以取得以太网 MAC 地址; □出的第二行□□是□□：

```
ed0   1500  <Link>0.2.c1.28.5f.4a         191923    0   129457     0   116
```

□行表明□个系□的以太网MAC地址是`00:02:c1:28:5f:4a` `-netstat` `-i`□出的以太网MAC地址必□

改成用冒号隔□， 并且要□个十六□数前加上。 □是arp(8)要求的格式; 参考arp(8) 的□机手册以□取完整的使用方法。

在□写 /etc/sliphome/slip.login 和 /etc/sliphome/slip.logout □， 一定要□置 "可□行" (execute) 位 (□言之， chmod 755 /etc/sliphome/slip.login /etc/sliphome/slip.logout)， 否□ sliplogin将无法□行它。

28.7.2.4.3. slip.logout配置

/etc/sliphome/slip.logout并不是必需的 (除非□使用了"ARP代理")， 如果□准□□建它， □里有一个基本的 slip.logout 脚本的例子：

```
#!/bin/sh -
#
#       slip.logout

#
# logout file for a slip line.  sliplogin invokes this with
# the parameters:
#      1        2         3        4          5         6     7-n
#   slipunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 down
```

如果使用了 "代理 ARP"， □可能希望 /etc/sliphome/slip.logout 在用□注□□自□□ SLIP 客□端□除 ARP □：

```
#!/bin/sh -
#
#       @(#)slip.logout

#
# logout file for a slip line.  sliplogin invokes this with
# the parameters:
#      1        2         3        4          5         6     7-n
#   slipunit ttyspeed loginname local-addr remote-addr mask opt-args
#
/sbin/ifconfig sl$1 down
# Quit answering ARP requests for the SLIP client
/usr/sbin/arp -d $5
```

arp -d $5 将□除由 "代理 ARP"slip.login 在 SLIP 客□程序登□□所生成的 ARP □。

再次□□： 建立 /etc/sliphome/slip.logout 之后， 一定要□置可□行位 (也就是□， chmod 755 /etc/sliphome/slip.logout)。

28.7.2.5. 路由考□

如果没有使用 "代理 ARP" 的方法来在□的 SLIP 客□机和网□的其余部分 (也可能是 Internet) 之

□路由数据包， □可能需要□加□□最近的默□路由器的静□路由， 以便通□ SLIP 服□器来在 SLIP 客□机子网上□行路由。

28.7.2.5.1. 静□路由

向□最近的默□路由添加一个静□路由可以□是很麻□ (或者□是不可能， 如果□没有□限□□做)。 如果在□的□□中使用多路由器网□， 有些路由器 (比如 Cisco 和 Proteon 生□的) 不但要配置指向 SLIP 子网的路由，而且□需要配置将□些静□路由□□其它的路由器。 所以一些□家意□和□□解答□于使基于静□路由表的路由正常工作很有必要。

Chapter 29. □子□件

29.1. 概述

"□子□件"，或通常所□的 email，是□今使用最广泛的通信方式之一。 本章将□如何在 FreeBSD 上□行□件服□，以及如何使用 FreeBSD 来收□□子□件作基本的介□； 然而，它并不是一□完整的参考手册，□□上，□多需要考□的重要事□都没有提及。 我□推□□者□□ 参考文献 中的参考□籍，以□得□于□部分的全面□□。

□完□章，□将了解：

- □些□件与收□□子□件有□。
- FreeBSD 下的基本 sendmail 配置文件在□里。
- 本地和□程□箱之□的区□。
- 如何阻止□□□件制造者非法地使用□的□件服□器作□□□中□。
- 如何安装和配置用于替代 sendmail 的其他□件□□代理。
- 如何□理常□的□件服□器□□。
- 如何使用 SMTP 和 UUCP。
- 如何□置系□使其只能□送□件。
- 如何在□号□接□使用□件。
- 如何配置 SMTP □□以□加安全性。
- 如何安装并使用用□□件代理，如 mutt 来收□□件。
- 如何从□程的 POP 或 IMAP 服□器上下□□件。
- 如何在□入的□件上自□地□用□□器和□□。

□□本章之前，□需要：

- 正□地配置□的网□□接 (高□网□).
- 正□地□□的□件服□器配置 DNS 信息 (网□服□).
- 知道如何安装第三方□件 (安装□用程序. Packages 和 Ports).

29.2. 使用□子□件

□件交□可以分□ 5 部分。它□是： 用□端程序、服□端守□□程、DNS、□程或本地的□箱、 当然，□有□件主机自己。

29.2.1. 用□端程序

□包括一些基于命令行的程序，例如 mutt、 alpine、elm 和 `mail`，以及□似 balsa、 xfmail □□的 GUI 程序。 此外，□有我□更"熟悉的"WWW □□器□□的程序。 □些程序□□地通□□用服□守□□程把□件事□交□本地的"□件主机"，或者通□ TCP 把□件□出去。

29.2.2. □件主机上使用的服□程序

FreeBSD 默□情况下采用 sendmail， 但它也支持□数□多的其它□件服□程序， □其中包括：

- exim;
- postfix;
- qmail.

□件服□器后台守□程序通常有□个功能 - 接收外面□来的□件和把□件□送出去。 但它 不 □□使用□似 POP 或 IMAP □□的□□来□□□□件， 也不□□□接到本地的 mbox 或 Maildir 信箱。 □可能需要其它的 服□程序 来完成□些任□。

□早版本的 sendmail 有一些□重的安全□□， 他□可能□致攻□者从本地和/或□程操作□的□□。 □□□□□自己使用的是最新版本以避免□些□□。 □外， 也可以从 FreeBSD Ports Collection 来安装其它的 MTA。

29.2.3. Email 和 DNS

域名系□ (DNS) 及其服□程序 named 在email的投□□程当中扮演着很重要的角色。 □了能□从□的站点向其它的站点□□□件， 服□程序需要通□ DNS □□接收□件的□程站点的位置。 □似地， 在□程站点向□的主机投□□件□也会□生□□的□□。

DNS □□将主机名映射□ IP 地址， 同□， 也需要保存□送□件□所需要的信息， □些信息称作 MX □□。 MX (Mail eXchanger，□件交□) □□指定了□个， 或□些主机能□接收特定域下的□件。 如果□没有□主机名或域名□置 MX □□， □□件将被直接□交□主机名□□ IP 所在的主机。

□可以通□ host(1) 命令来□□任何域或主机名□□的 MX □□， 如下面的例子所示：

```
% host -t mx FreeBSD.org
FreeBSD.org mail is handled (pri=10) by mx1.FreeBSD.org
```

29.2.4. 接收□件

□□的域接收□件是通□□件服□器来完成的。 它收集□送□□的域的那些□件，并保存到 mbox (存□□件默□的方法) 或 Maildir 格式， □取决于□采用的配置。 一旦□件被保存下来， 就可以在本地通□□似 mail(1) 或 mutt □□的程序， 或在□程通□ POP 或 IMAP □□的□□来□取了。 □□地□， 如果□只在本地□□□件，那就没有必要安装 POP 或 IMAP 服□。

29.2.4.1. 通□ POP 和 IMAP □□□程的□件

如果希望在□程□□□箱， 就需要□□ POP 或 IMAP 服□器。 □些□□允□用□从□程方便地□□他□的信箱。 尽管 POP 和 IMAP 都允□用□从□程□□信箱， 但 IMAP 有很多□点， □包括：

- IMAP 既可以从□程服□器上□取□件， 也可以把□件放上去。
- IMAP 支持并□更新。
- IMAP □于使用低速网□的用□尤□有用， 因□它能□□□些用□把□件的□□下□下去， 而无需立即下□整个□件。它□可以在服□器端□行□似□□□□的操作， 以□少客□机和服□器之□的通□量。

□可以按照下面的□□来安装和配置 POP 或 IMAP 服□器：

1. □□一个最符合需要的 IMAP 或 POP 服□器。 下列 POP 和 IMAP 服□器是最著名的，而且都有很多成功案例：
   - qpopper;
   - teapop;
   - imap-uw;
   - courier-imap;
   - dovecot;
2. 通□ ports collection 安装 POP 或 IMAP 服□。
3. 根据需要修改 /etc/inetd.conf 来加□ POP 或 IMAP 服□。

此外□□注意的是 POP 和 IMAP □□的信息， 包括用□名和口令等等， 通常都是明文的。□意味着如果□希望加密□□□程中的信息， 可能需要考□使用 ssh(1) 隧道或者使用SSL。□于如何□施隧道在 SSH 隧道 中□行了□□□述， SSL 部分在OpenSSL。

29.2.4.2. 操作本地的信箱

信箱可以在□件服□器本地直接用 MUA 来□行操作。 □通常是通□ mutt 或 mail(1) □□的□用程序□□的。

29.2.5. □件服□器

□件服□器是通□服□器□的一个名字 (□注：来□□主机)， □也正是它能在□的主机和网□上□送和接收□件的原因。

29.3. sendmail 配置

sendmail(8) 是 FreeBSD 中的默□□件□□代理 (MTA)。 sendmail 的任□是从□件用□代理 (MUA) 接收□件然后根据配置文件的定□把它□送□配置好的的寄送程序。 sendmail 也能接受网□□接， 并且□送□件到本地□箱或者□送它到其它程序。

sendmail 使用下列配置文件：

文件名	功能
/etc/mail/access	sendmail □□数据□文件
/etc/mail/aliases	□箱□名
/etc/mail/local-host-names	sendmail 接收□件主机列表
/etc/mail/mailer.conf	□寄配置程序
/etc/mail/mailertable	□件分□列表
/etc/mail/sendmail.cf	sendmail的主配置文件
/etc/mail/virtusertable	虚□用□和域列表

29.3.1. /etc/mail/access

□□数据□定□了什□主机或者 IP 地址可以□□本地□件服□器和它□是□□□型的□□。 主机可能会列出 OK、 REJECT 、RELAY 或者□□的通□ sendmail 的出□□理程序□□一个□定的□件□□。 主机默□列出 OK，允□□送□件到主机，只要□件的最后目的地是本地主机。列出 REJECT 将拒□所有的□件□接。如果□有 RELAY □□的主机将被允□通□□个□件服□器□送□件到任何地方。

例 *33.* 配置 *sendmail* 的□□□可数据□

```
cyberspammer.com                550 We do not accept mail from spammers
FREE.STEALTH.MAILER@            550 We do not accept mail from spammers
another.source.of.spam          REJECT
okay.cyberspammer.com           OK
128.32                          RELAY
```

在上面的例子中我□有 5 条□□。 与左□列表匹配的□件人受到右□列表□作的影□。 前□的□个例子□出了 sendmail 的出□□理程序□□到的□□代□。 当一个□件与左□列表相匹配□，□个信息会被打印到□程主机上。下一条□□拒□来自 Internet 上的一个特□主机的□件 another.source.of.spam。接下来的□□允□来自 okay.cyberspammer.com 的□件□接， □条□□比上面那行 cyberspammer.com 更准□。更多的准□匹配使不准□的匹配无效。最后一行允□□子□件从主机和 128.32 □□的 IP 地址□□。 □些主机将被允□通□□台□件服□器前往其它□件服□器□送□件。

当□个文件被升□的□候，□必□在 /etc/mail/ □行 make 升□数据□。

29.3.2. /etc/mail/aliases

□名数据□包含一个□展到用□，程序或者其它□名的虚□□箱列表。 下面是一些在 /etc/mail/aliases 中使用的例子：

例 *34.* □件□名

```
root: localuser
ftp-bugs: joe,eric,paul
bit.bucket:  /dev/null
procmail: "|/usr/local/bin/procmail"
```

□个文件的格式很□□； 冒号左□的□箱名， 会被展□成右□的形式。 第一个例子□□地将 root □箱□展□ localuser， 之后将□□在□名数据□中□行□□。 如果没有□到匹配的□□， □□件会被□□本地用□ localuser。第二个例子展示了一个□件列表。 □送到 ftp-bugs 的□件会被展□成 joe， eric 和 paul □三个□箱。注意也可以通□ user@example.com □□的形式来指定□程的□箱。 接下来的例子展示了如何把□件写入到文件中， □个例子中是 /dev/null。 最后一个例子展示了如何将□件□□一个程序， 具体而言是通□ UNIX® 管道□到 /usr/local/bin/procmail 的□准□入。

更新此文件□， □需要在 /etc/mail/ 中使用 make 来更新数据□。

29.3.3. /etc/mail/local-host-names

□是一个 sendmail(8) 被接受□一个本地主机名的主机名列表。 可以放入任何 sendmail 将从那里收□□件的域名或主机。例如，如果□个□件服□器从域 example.com 和主机 mail.example.com 接收□件，它的 local-host-names 文件，可以看起来象如下□□：

```
example.com
mail.example.com
```

当□个文件被升□，sendmail(8) 必□重新□□，以便更新□置。

29.3.4. /etc/mail/sendmail.cf

sendmail的主配置文件 sendmail.cf 控制着 sendmail 的所有行□， 包括从重写□件地址到打印拒□□程□件服□器信息等所有事。 当然，作□一个不同的角色，□个配置文件是相当□□的， 它的□□部分已□超出了本□的□□。幸□的是， □个文件□于□准的□件服□器来□很少需要被改□。

sendmail 主配置文件可以用 m4(1) 宏定□ sendmail 的特性和行□。它的□□□看 /usr/src/contrib/sendmail/cf/README。

当□个文件被修改□， sendmail 必□重新□□以便□新修改生效。

29.3.5. /etc/mail/virtusertable

virtusertable 映射虚□域名和□箱到真□的□箱。 □些□箱可以是本地的、□程的、/etc/mail/aliases 中定□的□名或一个文件。

例 35. 虚□域□件映射的例子

```
root@example.com                root
postmaster@example.com          postmaster@noc.example.net
@example.com                    joe
```

在上面□个例子中， 我□映射了一个域 example.com。 □个文件是按照从上到下， 首个匹配的方式来□理的。第一□将 root@example.com 映射到本地□箱 root。 下一□□将 postmaster@example.com 映射到位于 noc.example.net 的 postmaster。 最后， 如果没有来自 example.com 的匹配， □将使用最后一条映射，它表示将所有的其它□件□□ example.com 域的某个人。 □□， 将映射到本地信箱 joe。

29.4. 改□□的□件□□代理程序

先前已□提到，FreeBSD 中的 sendmail 已□安装了□的 MTA (□件□□代理程序)。因此它□□着□的收□□件的工作。

然而，基于不同的理由，一些系□管理□想要改□他□系□的 MTA。□些理由从□□的想要□□□一个 MTA，到需要一个特殊的特性或者 package 依□某个□寄程序等等。幸□的是，不管是什□理由，FreeBSD 都能容易的改□它。

29.4.1. 安装一个新的 MTA

□于可用的 MTA □有很多的□□。一个好的出□点是 FreeBSD Ports Collection，在那里□能□到很多。 当然□可以从任何位置不受任何限制的使用 MTA，只要□能□它□行在 FreeBSD 下。

□始安装□的新 MTA。一旦它被安装， 它可以□□有机会判定它是否能□足□的需要， 并且在它接管 sendmail 之前□□有机会配置□的新□件。 当完成□些之后，□□□□信安装的新□件不会□□更改系□的二□制文件例如 /usr/bin/sendmail。 除此以外， □的新□件□件□用之前要已□配置好它。

具体配置□参考□所□□的 MTA □件的配置文□或其它相□□料。

29.4.2. 禁用 sendmail

如果□打算禁用 sendmail 的□件□出服□， 保持系□中有一个替代它的、 可用的□件□送系□就非常重要。 如果□不□□做的□， □似 periodic(8) □□的系□功能就无法如□期的那□， 通□□件来□送其□行□果。 □系□中的□多部分可能都假定有可用的 sendmail-兼容 系□。 如果□些□用程序□□使用 sendmail 的□行文件来□送□件， 而□又禁用了它， □□件将□入 sendmail 的非活□ (inactive) □列， 而永□不会被送□。

要□底禁用包括□件送出服□在内的所有 sendmail 功能， 必□将

```
sendmail_enable="NO"
sendmail_submit_enable="NO"
sendmail_outbound_enable="NO"
sendmail_msp_queue_enable="NO"
```

写入 /etc/rc.conf。

如果只是想要停止 sendmail 的接收□件服□， □□□在 /etc/rc.conf 文件中□置

```
sendmail_enable="NO"
```

更多的有□ sendmail 可用的□□□□，参看 rc.sendmail(8) □机手册.

29.4.3. 机器引□□□行□的新 MTA

可以向 /etc/rc.conf 中加入配置□使新的 MTA 在系□□□□□行， 下面是一个 postfix 的例子：

```
# echo 'postfix_enable="YES"' >> /etc/rc.conf
```

□□ MTA 就能在系□□□是自□□行了。

29.4.4. 替□系□默□的□寄程序 sendmail

因□ sendmail 程序是一个在 UNIX® 系□下普遍存在的一个□准的□件，一些□件就假定它已□被安装并且配置好。 基于□个原因，□多其它的 MTA 提供者都提供了兼容 sendmail 的命令行界面来□行。

□使它□像"混入"sendmail 一□□的很容易掌握。

因此，如果□使用其它的□寄程序， □必□□定□个□件是去□□□行□准的 sendmail 二□制，就象 /usr/bin/sendmail，□是□行□自己□□的替□□寄程序。 幸□的是，FreeBSD 提供了一个系□□用 mailwrapper(8)，它能□□做□件工作。

当 sendmail 安装后被□行，□可以在 /etc/mail/mailer.conf 中□到如下行：

```
sendmail     /usr/libexec/sendmail/sendmail
send-mail    /usr/libexec/sendmail/sendmail
mailq        /usr/libexec/sendmail/sendmail
newaliases   /usr/libexec/sendmail/sendmail
hoststat     /usr/libexec/sendmail/sendmail
purgestat    /usr/libexec/sendmail/sendmail
```

□个的意思就是当□些公共命令 (例如 sendmail 它本身) □行□， 系□□□上□用了一个 sendmail 指定的 mailwrapper 的副本，它□□ mailer.conf 并且□行 /usr/libexec/sendmail/sendmail 做□替代。当默□的 sendmail 功能被□用， 系□将很容易的改□□□上□行的二□制文件。

因此如果□想要 /usr/local/supermailer/bin/sendmail-compat 替□ sendmail 被□行，□□□改□ /etc/mail/mailer.conf 文件□：

```
sendmail     /usr/local/supermailer/bin/sendmail-compat
send-mail    /usr/local/supermailer/bin/sendmail-compat
mailq        /usr/local/supermailer/bin/mailq-compat
newaliases   /usr/local/supermailer/bin/newaliases-compat
hoststat     /usr/local/supermailer/bin/hoststat-compat
purgestat    /usr/local/supermailer/bin/purgestat-compat
```

29.4.5. 最后

一旦做完□想要配置的□件事，□□□□掉 sendmail □程并且□□属于□的新□件的□程， 或者□□的重□。 重□也将□□提供了□□□的系□已□□行了正□的配置的机会。 在引□的□候自□的□行□新的 MTA。

29.5. 疑□解答

29.5.1. □什□必□在我的站点的主机上使用 FQDN？

□可能会□□主机□□上是在□外一个域里面， 例如，如果□是在 foo.bar.edu 里，而□要□一台叫 mumble 的主机，它在 bar.edu 域里，□就必□用完整的域名 mumble.bar.edu，而不是用 mumble。

□□上，□在 BSD BIND resolvers 中是可行的。 然而目前随 FreeBSD 附□的 BIND 已不□同一域外提供□写服□。所以，□个不完整的主机名 mumble 必□以 mumble.foo.bar.edu □□形式才能被□到，或者将在根域中搜索它。

□跟以前的□理是不同的，以前版本将会□□□□ mumble.bar.edu 和 mumble.edu。 如果□想要了解□□方式是否是好，或者它有什□安全方面的漏洞，□参□ RFC 1535 文□。

如果□想要一个好的工作□境，□可以使用如下行：

```
search foo.bar.edu bar.edu
```

替□先前旧的版本：

```
domain foo.bar.edu
```

把□行放在□的 /etc/resolv.conf 文件中。然而，□一定要□定□□的搜□□序不会造成 RFC 1535 里提到的"boundary between local and public administration" □□。

29.5.2. sendmail 提示信息 mail loops back to myself

下面是 sendmail FAQ 中的回答：

```
我得到了如下的信息：

553 MX list for domain.net points back to relay.domain.net
554 <user@domain.net>... Local configuration error

我如何解决□个□□？

□已□通□ MX □□指定把□送□特定的域（例如，domain.net）
的□件被□寄到指定的主机（在□个例子中，relay.domain.net），
而□台机器并不□□它自己是 domain.net。□把 domain.net 添加到
/etc/mail/local-host-names 文件中 [在 8.10 版之前是 /etc/sendmail.cw]
(如果□使用 FEATURE(use_cw_file) 的□) 或者在 /etc/mail/sendmail.cf
中添加Cw domain.net。
MX record
```

sendmail 的 FAQ 可以在 http://www.sendmail.org/faq/ □到， 如果□想要□□的□件做任何的"□整"， □推□首先看一看它。

29.5.3. 我如何在一个□号主机上□行一个□件服□？ PPP

□想要把局域网上的 FreeBSD 主机□接到互□网上，而□台 FreeBSD 主机将会成□□个局域网的□件网□， □个□号□接不必一直保持在□接状□。

最少有□□方法可以□足□的要求。一□方法就是使用 UUCP。

□一□方法是□到一个□□的服□器来□□的域提供副 MX 主机服□。 例如，如果□公司的域名是 example.com，□的互□网服□提供者把 example.net 作□□域的副 MX 服□：

```
example.com.          MX        10      example.com.
                      MX        20      example.net.
```

只有一台主机被指定当做□的最□收信主机（在 example.com 主机的 /etc/mail/sendmail.cf 文件中添加 Cw example.com）。

当 sendmail □□分□□件的□候， 它会□□通□ modem □接到□ (example.com)。 因□□并不在□，所以□是会得到一个超□的□□。 sendmail 将会把□件□送到副 MX 主机，也就是□，□的互□网服□提供者 (example.net)。副 MX 主机将周期性的□□□接并□送□件到□的主机 (example.com)。

□也□想要使用下面的□个登□脚本：

```
#!/bin/sh
# Put me in /usr/local/bin/pppmyisp
( sleep 60 ; /usr/sbin/sendmail -q ) &
/usr/sbin/ppp -direct pppmyisp
```

如果□想要□一个用□建立一个分□登□的脚本， □可以使用 sendmail -qRexample.com 替□上面的脚本。□□将使所有的□件按照□的 example.com □列立即被□理。

更深入的方法可以参考下面□段：

□段信息是从 FreeBSD Internet 服□提供商的□件列表 拿来的。

```
> 我□□用□提供副 MX 主机服□。用□□天都会上□好几次
> 并且自□把信件取回主 MX 主机
> (当有他□的□件□我□并没有通知他□)。
> 我□的 mailqueue 程序□ 30 分□清一次□件□列。那段□□他□
> 就必□上□ 30 分□以□保他□的信件送□他□的主 MX 主机。
>
> 有任何指令可以用 sendmail 寄出所有□件□？
> 普通用□在我□的机器上当然没有 root □限。

在 sendmail.cf 的privacy flags部分，有□□的□定
Opgoaway,restrictqrun

移除 restrictqrun 可以□非 root 用□□□列□理的程序。
□可能也要重新安排□的 MX □定。我□是用□的 MX 主机，
而且我□□□定了□个：

# If we are the best MX for a host, try directly instead of generating # local config
error.
OwTrue

□□的□□程机器会直接把信送□□，而不会□□□接□的用□的机器。
然后□就可以把□件□送到□的用□。□个□定只□
主机有效，所以□必□要□□的用□在 DNS 中把他□的□件主机□置□
customer.com或者
hostname.customer.com。只要□customer.com在 DNS
里添加一个 A □□就可以了。
```

29.5.4. □什□当我□送□件到其它主机□是有 Relaying Denied 出□信息？

默□的 FreeBSD 安装中， sendmail 会配置□只□送来自它所在主机上的□件。 例如，如果有可用的 POP 服□器，□用□将可以从学校、 公司或其他什□□的地方□□□件，但他□仍然无法从□程直接□送□件。 通常，在几次□□之后， MAILER-DAEMON 将□出一封包含 5.7 Relaying Denied □□信息的□件。

有很多方法可以避免□□□象。 最直截了当的方法是把□的 ISP 的地址放到 /etc/mail/relay-domains 文件中。 完成□□工作的□□的方法是：

```
# echo "your.isp.example.com" > /etc/mail/relay-domains
```

建立或□□□个文件以后□必□重新□□ sendmail。 如果□是一个管理□并且不希望在本地□送□件，或者想要在其它的机器甚至其它的 ISP 上使用一个客□端系□， □个方法是很方便的。如果□□有一到□个□件□□它也非常的有用。 如果有大量的地址需要添加， □可以很□□的使用□喜□的文本□□器打□□个文件添加域名，□行一个：

```
your.isp.example.com
other.isp.example.net
users-isp.example.org
www.example.org
```

□在□件可以通□□的系□□送， □个列表中存在的主机（前提是用□在□的系□上已□有一个□□）将可以成功的□送。□是一个允□正常的□程用□从□的系□□送□件， 并且阻止其它非法用□通□□系□□送□□□件的好方法。

29.6. 高□主□

下面□□将介□□件配置和□整个域安装□件。

29.6.1. 基本配置

在□箱外，只要□□置 /etc/resolv.conf 或者□行□自己的名字服□器，□就可以□送□件到外部的主机。 如果□想要□的□件□送□某个特定的 MTA(例如， sendmail) 在□的 FreeBSD 主机上，有□个方法：

- □行□自己的域名服□器和□自己的域。例如， FreeBSD.org
- □得直接分□□□主机的□件。□可以直接使用□当前的 DNS 名称。例如，example.FreeBSD.org。

不管□□□上面那□方法，□了直接在□的主机上□送□件， 必□有一个静□的 IP 地址(不是象 PPP □号一□的□□地址)。 如果□在防火□后面，它必□□ SMTP □□通□。 如果□想要在□的主机上直接的收取□件， □必□□定□件事：

- □定在□ DNS 中的 MX □□(最小□号的)指向□的 IP 地址。
- □定在□ DNS 中的 MX □□没有禁止□的主机。

上面的□条□□都允□□在□的主机直接接收□件。

□□□个：

```
# hostname
example.FreeBSD.org
# host example.FreeBSD.org
example.FreeBSD.org has address 204.216.27.XX
```

如果□看到□些， □直接□往 yourlogin@example.FreeBSD.org □□已□可以正常工作了 (假□ sendmail 已□在 example.FreeBSD.org 上正□□□了)。

如果□看到□些：

```
# host example.FreeBSD.org
example.FreeBSD.org has address 204.216.27.XX
example.FreeBSD.org mail is handled (pri=10) by hub.FreeBSD.org
```

所有□送到主机 (example.FreeBSD.org) 的□件在相同的用□名下将会被 hub □止的收集，而不是直接□送到□的主机。

上面的信息是通□□的 DNS 服□器来□理的。支持□件路由信息的 DNS □□是 □件 交□ □□。如果 MX □□不存在，□件将通□它自己的 IP 地址被直接的□送到主机。

freefall.FreeBSD.org的MX□□如下所示:

```
freefall        MX  30  mail.crl.net
freefall        MX  40  agora.rdrop.com
freefall        MX  10  freefall.FreeBSD.org
freefall        MX  20  who.cdrom.com
```

正如□□看到的，freefall 有很多 MX □□。 最小□号的 MX □□是直接接收□件的主机。如果因□一些原因它不可用，其它 (有□会□□"backup MXes")接收信息将会□□接替并做□□的排列。

□了有效的使用交□式 MX 站点，□当从□的机器上分□一些 Internet □接。□的 ISP 或者其它友好的站点可以没有任何□□的□□提供□个服□。

29.6.2. Mail for Your Domain

□了□置一个"□件主机"(又称□件服□器) □必□要把□多□件□送到与它相□的几个工作站中。 基本上，□想要"要求"在□域的□个主机的所有□件 (在□个例子里是 *.FreeBSD.org) □向到□的□件服□器，从而使□的用□可以在主□件服□器里接收他□的□件。

要使工作最□□，□有同□ 用□名 的□□□□同□存在于□台机器上。使用 adduser(8) 来□□做。

□将使用的□件主机必□□□个工作站指定一个□件交□。□可以在 DNS 中□□配置：

```
example.FreeBSD.org A   204.216.27.XX       ; Workstation
            MX  10 hub.FreeBSD.org  ; Mailhost
```

无□ A □□指向□，□将□工作站重新定位到□件主机。□件将被□送到 MX 主机。

□不能自己□□做除非□□行着一个 DNS 服□器。 如果不是□□，或者不能□行□自己的 DNS 服□器，告□□的 ISP 或者□□提供 DNS 服□的人。

如果□正在使用虚□□件主机，下面的信息将会□□有用。 在□个例子里，我□假定□有一个客□并且他有自己的域， □个例子中是 customer1.org，□要把 customer1.org 所有的□件□送到□的□件主机 mail.myhost.com。 □的 DNS □□□□是□□：

```
customer1.org       MX  10  mail.myhost.com
```

□ 不 需要有个 A □□， 如果□只□域 customer1.org □理□件。

必□清楚 customer1.org 将不能工作，除非存在一个 A □□。

最后一件□必□要做的事是告□ sendmail 接受□件的是什□域和(或)主机名。 □里有好几□方法。下面方法可以任□一□：

- 添加□的主机到 /etc/mail/local-host-names 文件中，如果□使用的是 FEATURE(use_cw_file)。如果□使用 sendmail 8.10 或者更高版本，文件是 /etc/sendmail.cw。
- 添加一行 Cwyour.host.com 到□的 /etc/sendmail.cf 或 /etc/mail/sendmail.cf 文件，如果□使用 sendmail 8.10 或者更高版本。

29.7. SMTP 与 UUCP

sendmail 的配置，在 FreeBSD 中已□配置好□□的站点直接的□接 Internet。 如果站点希望他□的□件通□ UUCP 交□，□必□安装其它的 sendmail 配置文件。

手工的配置 /etc/mail/sendmail.cf 是一个高□主□。sendmail 8 版本通□ m4(1) □□理生成一个配置文件，□□上□个配置□生在一个比□高的抽象□。 m4(1) 配置文件可以在 /usr/shared/sendmail/cf 下□到。 cf 目□中的 README 文件是□于 m4(1) 配置的基本的介□。

最好的支持 UUCP □送的方法是使用 mailertable 的特点。建立一个□料□□ sendmail 可以使用它自己的路由决策。

首先，□必□建立□自己的 .mc 文件。 /usr/shared/sendmail/cf/cf 目□包含一些例子。 假定□已□命名自己的文件叫做 foo.mc， □要做的只是把它□□成一个有效的 sendmail.cf：

```
# cd /etc/mail
# make foo.cf
# cp foo.cf /etc/mail/sendmail.cf
```

一个典型的 .mc 文件看起来可能象□□：

```
VERSIONID(`Your version number') OSTYPE(bsd4.4)

FEATURE(accept_unresolvable_domains)
FEATURE(nocanonify)
FEATURE(mailertable, `hash -o /etc/mail/mailertable')

define(`UUCP_RELAY', your.uucp.relay)
define(`UUCP_MAX_SIZE', 200000)
define(`confDONT_PROBE_INTERFACES')

MAILER(local)
MAILER(smtp)
MAILER(uucp)

Cw    your.alias.host.name
Cw    youruucpnodename.UUCP
```

accept_unresolvable_domains、 nocanonify 和 confDONT_PROBE_INTERFACES 特性将避免在□送□件□使用DNS的机会。UUCP_RELAY □是支持 UUCP □送所必□的。□□的放入一个 Internet 上可以□理 UUCP 虚□域地址的主机名。通常，□在□里填入□ ISP □件的回□□。

一旦□做完□些，□□需要□个 /etc/mail/mailertable 文件。 如果□只有一个用来□□所有□件的□外通道的□，以下的文件就足□了：

```
#
# makemap hash /etc/mail/mailertable.db < /etc/mail/mailertable
.                           uucp-dom:your.uucp.relay
```

一个更□□点的例子象□□：

```
#
# makemap hash /etc/mail/mailertable.db < /etc/mail/mailertable
#
horus.interface-business.de   uucp-dom:horus
.interface-business.de        uucp-dom:if-bus
interface-business.de         uucp-dom:if-bus
.heep.sax.de                  smtp8:%1
horus.UUCP                    uucp-dom:horus
if-bus.UUCP                   uucp-dom:if-bus
.                             uucp-dom:
```

□三行□理域地址□件，不□□被□送出默□的路由， 而由某些 UUCP □居取代的特殊情况，□是□了走"捷径"。下一行□理本地网的□件□它可以使用 SMTP 来□送。 最后，UUCP □居提起。UUCP 虚□域的□□， 允□一个 uucp-neighbor !recipient 推翻默□□□。最后一行□以一个□独的句点最□□束， 以 UUCP □送到提供□所有的□件网□的 UUCP □居。 所有在 uucp-dom: □□字里的□点名称必□是有效的 UUCP □居，□可以用 uuname 去□□。

提醒□□个文件在使用前必□被□□成 DBM 数据□文件。最好在 mailertable 最上面用注解写出命令行来完成

□个工作。 当□□次更□□的 mailertable 后□□是需要□行□个命令。

最后提示：如果□不□定某个特定的路径可用， □得把 `-bt` □□加到 sendmail。□会将 sendmail □□在 地址□□模式。只要按下 `3,0`，接着□入□希望□□的□件路径位置。 最后一行告□□使用□件代理程序，代理程序会通知目的主机以及 (可能□□) 地址。 要□□此模式□按 `Ctrl` + `D`。

```
% sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> 3,0 foo@example.com
canonify           input: foo @ example . com
...
parse            returns: $# uucp-dom $@ your.uucp.relay $: foo < @ example . com . >
> ^D
```

29.8. 只□送□件的配置

□多□候， 可能只希望通□□□服□器来□送□件。 典型的情况包括：

- 使用□面机， 但希望通□□似 send-pr(1) □□的程序□送□件。 □□就需要使用 ISP 的□件□□服□器。
- 不在本地□理□件的服□器， 但它需要把□件交□□□服□器来□行□理。

几乎任何一个 MTA 都能□□任□□的工作。 然而不幸的是， 要把一个全功能的 MTA 正□地配置□只把□件交□其他服□器是一件很困□的事情。 使用 sendmail 以及 postfix □□的程序， 多少有些□□用牛刀的感□。

此外， 如果□使用典型的 Internet □□服□， □的□□可能会包含禁止□行 "□件服□器" 的条款。

□足□些需要最□□的□法是安装 mail/ssmtp port。 以 `root` 身□□行下面的命令：

```
# cd /usr/ports/mail/ssmtp
# make install replace clean
```

一旦装好， mail/ssmtp 就可以用四行 /usr/local/etc/ssmtp/ssmtp.conf 来配置：

```
root=yourrealemail@example.com
mailhub=mail.example.com
rewriteDomain=example.com
hostname=_HOSTNAME_
```

□□□□□ `root` 使用了真□的□子□件地址。 用□的 ISP 提供的外□□件□□服□器名称， 替□掉 `mail.example.com` (某些 ISP 可能将其称□ "外□□件服□器" 或 "SMTP 服□器")。

接下来需要□□禁用了 sendmail， 包括□件□出服□在内。 □参□ 禁用 sendmail 以了解□一□的□□。

mail/ssmtp 也提供了一些其他□□。 □参□在 /usr/local/etc/ssmtp 中的示例配置， 或者 ssmtp 的□机手册来得到一些例子和更多的其他信息。

以□□方式配置 ssmtp， 能□□□□算机上的任何需要□送□件的□件都正常□□， 而不必冒□反 ISP 的使用政策，或使□的□□被劫持用于□送□□□件的□□。

29.9. □号□接□使用□件□送

如果□有静□的 IP 地址， 就□□不用修改任何默□的配置。 将主机名□置□分配□□的 Internet 名称，其他的事情 sendmail 都会替□做好。

如果□的 IP 地址是□□分配的， 并使用 PPP □接□入 Internet， □□可能会从 ISP 的□件服□器上得到一个信箱。□里我□假□□的 ISP 的域名是 example.net， □的用□名是 user， □把自己的机器称作 bsd.home， 而□的 ISP 告□□可以使用 relay.example.net 来□□□件。

□了从□箱收取□件， 需要安装一个收信代理。 fetchmail 是一个能□支持□多□不同□□的不□的□□。□个程序可以通□ package 或 Ports Collection (mail/fetchmail) 来安装。 通常， □的 ISP 会提供 POP。如果□使用用□ PPP，□□可以在 Internet □接建立□自□地□取□件， □可以通□在 /etc/ppp/ppp.linkup 中□加如下的□来□□：

```
MYADDR:
!bg su user -c fetchmail
```

如果□正使用 sendmail (如下所示) 来□非本地用□□送□件， □可能需要□ sendmail 在□的 Internet □接建立□立即□送□件□列。 要完成□□工作， □□把下面的命令放到 /etc/ppp/ppp.linkup 中的 fetchmail 之后

```
  !bg su user -c "sendmail -q"
```

假□□在 bsd.home 上有一个 user 用□。 在 bsd.home 上的 user 主目□中□建一个 .fetchmailrc 文件：

```
poll example.net protocol pop3 fetchall pass MySecret
```

因□包含了密□ MySecret， □个文件□□只有 user 可□。

要使用正□的 from: □来□送文件， □必□告□ sendmail 使用 user@example.net 而不是i user@bsd.home。□外， □可能也需要要求 sendmail 通□ relay.example.net 来□送□件， 以便更快地□送它□。

以下的 .mc 文件□□可以□足□的需求：

```
VERSIONID(`bsd.home.mc version 1.0')
OSTYPE(bsd4.4)dnl
FEATURE(nouucp)dnl
MAILER(local)dnl
MAILER(smtp)dnl
Cwlocalhost
Cwbsd.home
MASQUERADE_AS(`example.net')dnl
FEATURE(allmasquerade)dnl
FEATURE(masquerade_envelope)dnl
FEATURE(nocanonify)dnl
FEATURE(nodns)dnl
define(`SMART_HOST', `relay.example.net')
Dmbsd.home
define(`confDOMAIN_NAME',`bsd.home')dnl
define(`confDELIVERY_MODE',`deferred')dnl
```

如何□□□个 .mc 文件到 sendmail.cf 文件的□□，□参考前面的章□。 □外，在更新 sendmail.cf 文件后，不要忘□重□ sendmail。

29.10. SMTP □□

在□的□件服□器上□用 SMTP □□有很多好□。 SMTP □□可以□ sendmail 多一重安全保障，而且也使得使用不同机器的漫游用□能□使用同一个□件服□器， 而不需要□次都修改它□的□件客□端配置。

1. 从 ports 安装 security/cyrus-sasl2。 □个 port 位于 security/cyrus-sasl2。 security/cyrus-sasl2 port 支持很多可以在□□□指定的可□□。 由于我□要使用 SMTP 身□□□， 因此要□□没有禁用 LOGIN □□。

2. 安装完 security/cyrus-sasl2 之后， □□ /usr/local/lib/sasl2/Sendmail.conf (如果不存在□建立一个) 并在其中□加下列配置：

```
pwcheck_method: saslauthd
```

3. 接下来， 安装 security/cyrus-sasl2-saslauthd， □□ /etc/rc.conf 并加入下列配置：

```
saslauthd_enable="YES"
```

   最后□用 saslauthd 服□：

```
# /usr/local/etc/rc.d/saslauthd start
```

   □个服□将充当 sendmail 使用 FreeBSD 的 passwd 数据□来完成身□□□□的代理人角色。 □避免了□□个需要使用 SMTP 身□□□的用□建立□□的用□名和口令的麻□， 也□保了登□与□件的口令一致。

4. □在□□ /etc/make.conf 文件，添加如下行：

```
SENDMAIL_CFLAGS=-I/usr/local/include/sasl -DSASL
SENDMAIL_LDFLAGS=-L/usr/local/lib
SENDMAIL_LDADD=-lsasl2
```

   □些配置将告□系□在□□ sendmail □使用□当的配置□□来在□□□程中□入 cyrus-sasl2. 在重新□□ sendmail 之前， □□□已□安装了 cyrus-sasl2。

5. 重新□□ sendmail □行如下命令：

```
# cd /usr/src/lib/libsmutil
# make cleandir && make obj && make
# cd /usr/src/lib/libsm
# make cleandir && make obj && make
# cd /usr/src/usr.sbin/sendmail
# make cleandir && make obj && make && make install
```

   如果 /usr/src 和共享□没有大的□化并且它□都必□可用，sendmail □□□□没有任何□□。

6. sendmail 被重新□□和安装后， □□□的 /etc/mail/freebsd.mc 文件 (或者无□□□□使用的□的□个 .mc 文件。□多管理□□□使用跟 hostname(1) 一□的唯一的 .mc 文件□出)。添加□些行在□个文件：

```
dnl set SASL options
TRUST_AUTH_MECH(`GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN')dnl
define(`confAUTH_MECHANISMS', `GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN')dnl
```

□些□□配置有不同的方法，□于 sendmail □□用□。 如果□想要使用除 pwcheck 之外的方法，□参考相□文□。

7. 最后，在 /etc/mail □行 make(1)。 它将建立□的新 .mc 文件并建立一个 .cf 文件命名□ freebsd.cf (或者□想使用□的其它名字的 .mc文件)。接着使用命令 `make install restart`，□将□制文件到 sendmail.cf，并且正□的重新□□ sendmail。 更多有□□个□程的信息，□可以参考 /etc/mail/Makefile 文件。

如果所□个□□都做□了， □□□可以通□□的□件客□端□入□的登□信息并且□送一个□□信息。 更多的分析，□置 sendmail 的 `LogLevel` 到 13 并且□看 /var/log/maillog 中的信息。

如欲了解更多的信息， □参看 sendmail 网站上的 □于 SMTP □□ 的介□。

29.11. □件用□代理

□件用□代理 (MUA) 是一个用于收□□件的□用程序。 更□一□， 随着□子□件的 "演化" 并愈□□□， MUA 在和□子□件相□合方面□得日□□大； □□用□提供了更多的功能和□活性。 FreeBSD 包含了□于□多□件用□代理的支持，所有□些都可以通□ FreeBSD Ports Collection 来□松安装。 用□可以□□□似 evolution 以及 balsa □□的□形界面程序， 也可以□□□似 mutt、 alpine 或 `mail` □□的控制台程序， 或者某些大型机□使用的 web 界面。

29.11.1. mail

mail(1) 是 FreeBSD 中默□的□件用□代理 (MUA)。 它是一个基于控制台的 MUA， 提供了所有用于收□文本形式的□子□件所需的基本功能， □然它□理附件的能力有限， 而且只支持本地的信箱。

□然 `mail` 没有内建的 POP 或 IMAP 服□器支持， 然而□些信箱可以通□□似 fetchmail □□的□用程序， 来下□到本地的 mbox 文件中。 □一□用程序在本章的□后部分 (使用 fetchmail) □行了介□。

要收□□件， 只需□□地使用 `mail` 命令， 如下所示：

```
% mail
```

用□保存在 /var/mail 中的信箱的内容会被 `mail` 程序自□地□取。 如果信箱是空的， 程序会退出并□出一个消息表示没有□件。 一旦□完了信箱， 将□□□用程序的界面， 并列出□件。 所有的□件会被自□□号，□似下面的□子：

```
Mail version 8.1 6/6/93.  Type ? for help.
"/var/mail/marcs": 3 messages 3 new
>N  1 root@localhost        Mon Mar  8 14:05  14/510   "test"
 N  2 root@localhost        Mon Mar  8 14:05  14/509   "user account"
 N  3 root@localhost        Mon Mar  8 14:05  14/509   "sample"
```

□在， □通□使用 mail 的 t 命令， 并□出□件的□号， 就可以看到□件了。 在□个例子中， 我□将□□第一封□件：

```
& t 1
Message 1:
From root@localhost  Mon Mar  8 14:05:52 2004
X-Original-To: marcs@localhost
Delivered-To: marcs@localhost
To: marcs@localhost
Subject: test
Date: Mon,  8 Mar 2004 14:05:52 +0200 (SAST)
From: root@localhost (Charlie Root)

This is a test message, please reply if you receive it.
```

正如在上面的例子中所看到的， t □将□示完整的□件□。 要再次□看□件的列表， 可以使用 h □。

如果需要回□□件， 也可以使用 mail 来完成， 方法是使用 R 或 r □□个 mail□。 R □会要求 mail 只回□□送□件的人， 而 r 不□回□□送□件的人， 而且也会将回□抄送□原来□件的其他接收者。 如果需要， 也可以在□些命令后面指定□件的□号。 做完□些之后， 就可以□入回□了， 在□件的最后□□有一个只有一个 . 的行，例如：

```
& R 1
To: root@localhost
Subject: Re: test

Thank you, I did get your email.
.
EOT
```

要□出新□件， 可以使用 m， 后面接收件人的□件地址。 多个收件人之□， □□使用 , 隔□。 接下来需要□入□件的主□， 然后是正文。 同□的， 在□件最后需要一个只有 . 的空行表示□束。

```
& mail root@localhost
Subject: I mastered mail

Now I can send and receive email using mail ... :)
.
EOT
```

在 mail 工具中， 可以用 ? 来□示□助， 而参考 mail(1) □机手册□可以□得更多□于 mail 的□助信息。

正如前面所提到的那□， mail(1) 命令在□□□没有考□到要□理附件， 因而在□方面他的功能很弱。 新的 MUA， 如 mutt， 能□更好地□理附件。 但如果□仍然希望使用 mail 命令， 那□ converters/mpack port □是一个□得考□的附加工具。

29.11.2. mutt

mutt 是一个短小精悍的□件用□代理， 它提供了□多卓越的功能， 包括：

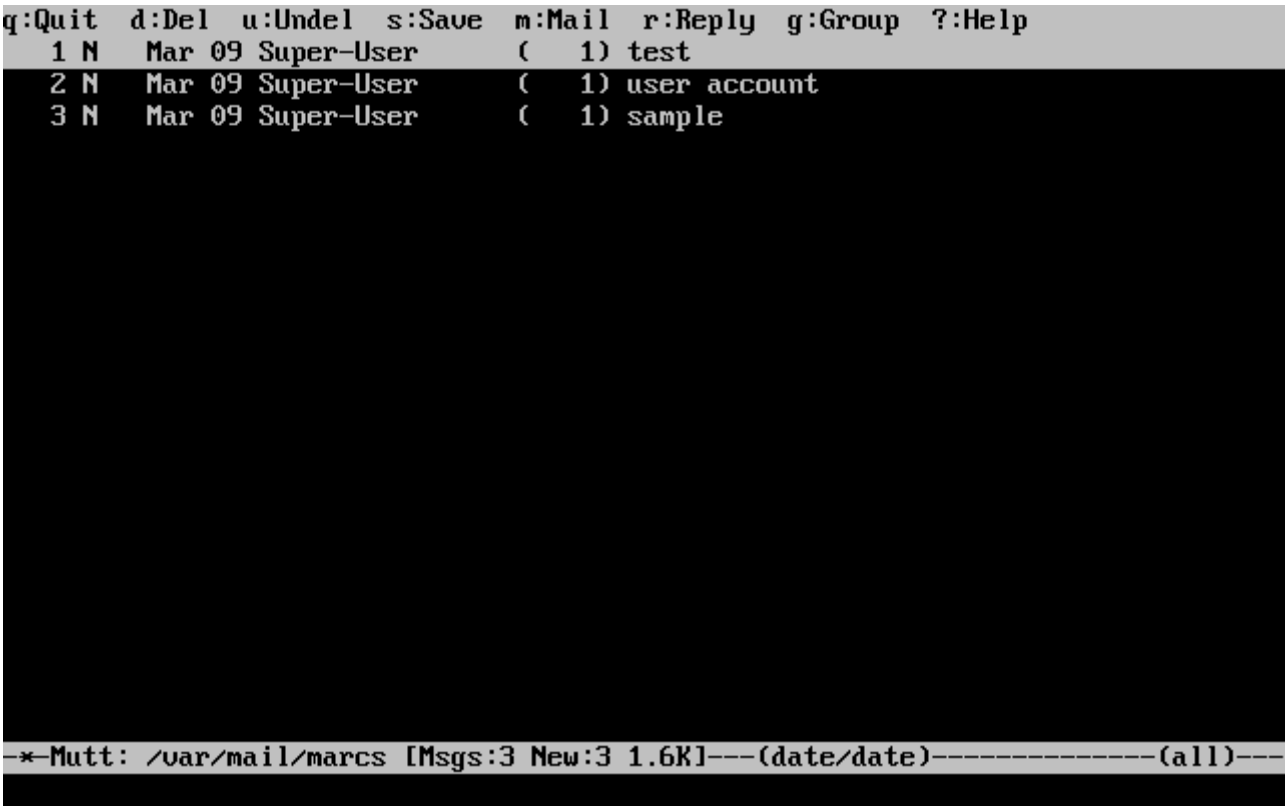
- 能□按□索□□□件；
- 支持使用 PGP □□件□行数字□名和加密；
- 支持 MIME；
- 支持 Maildir；
- 高度可定制。

所有□些特性， 都使得 mutt 得以□身于目前最先□的□件用□代理的行列。 □参考 http://www.mutt.org 以了解更多□于 mutt 的□料。

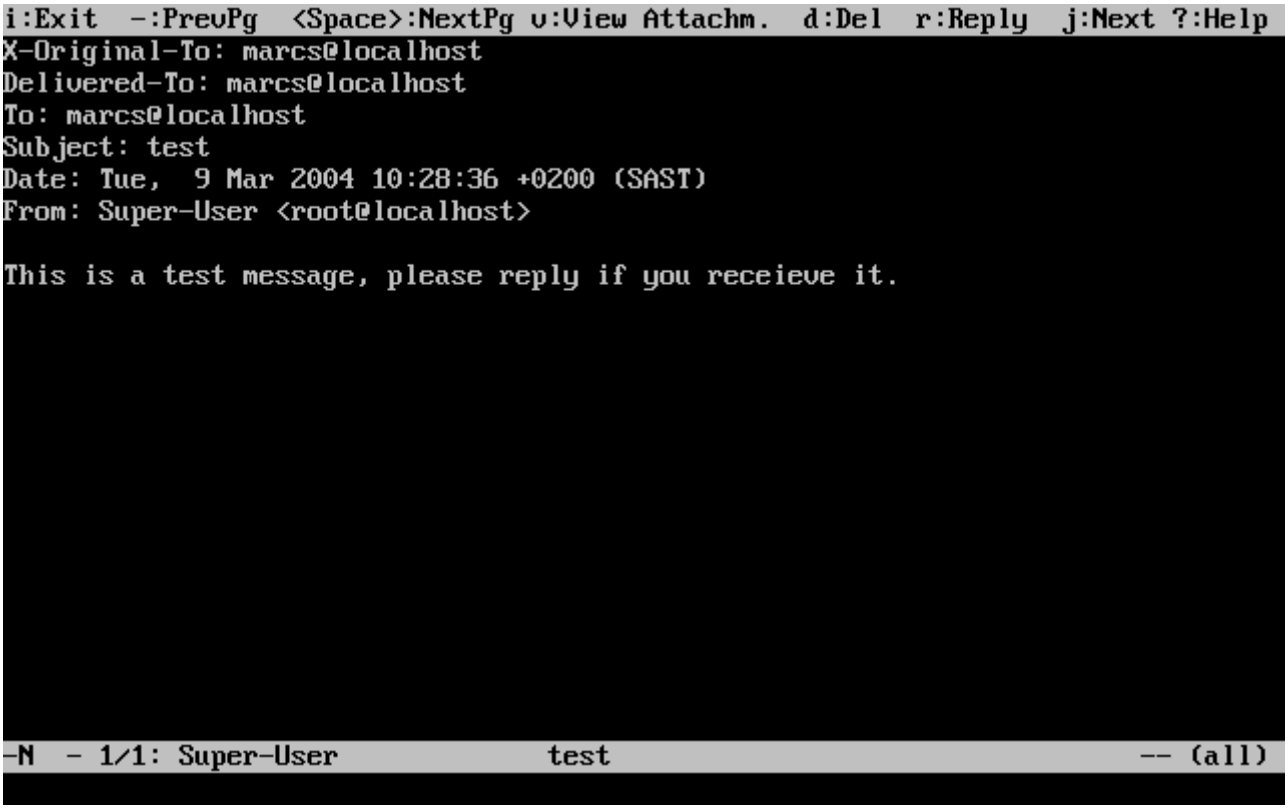
□定版本的 mutt 可以通□ mail/mutt port 来安装， 而□□版本， □可以通□使用 mail/mutt-devel port 安装。 通□ port 安装之后，可以通□下面的命令来□□ mutt：

```
% mutt
```

mutt 会自□□取 /var/mail 中的用□信箱， 并□示其内容。 如果用□信箱中没有□件， □ mutt 将等待来自用□的命令。 下面的例子展示了 mutt 列出□件的情形：



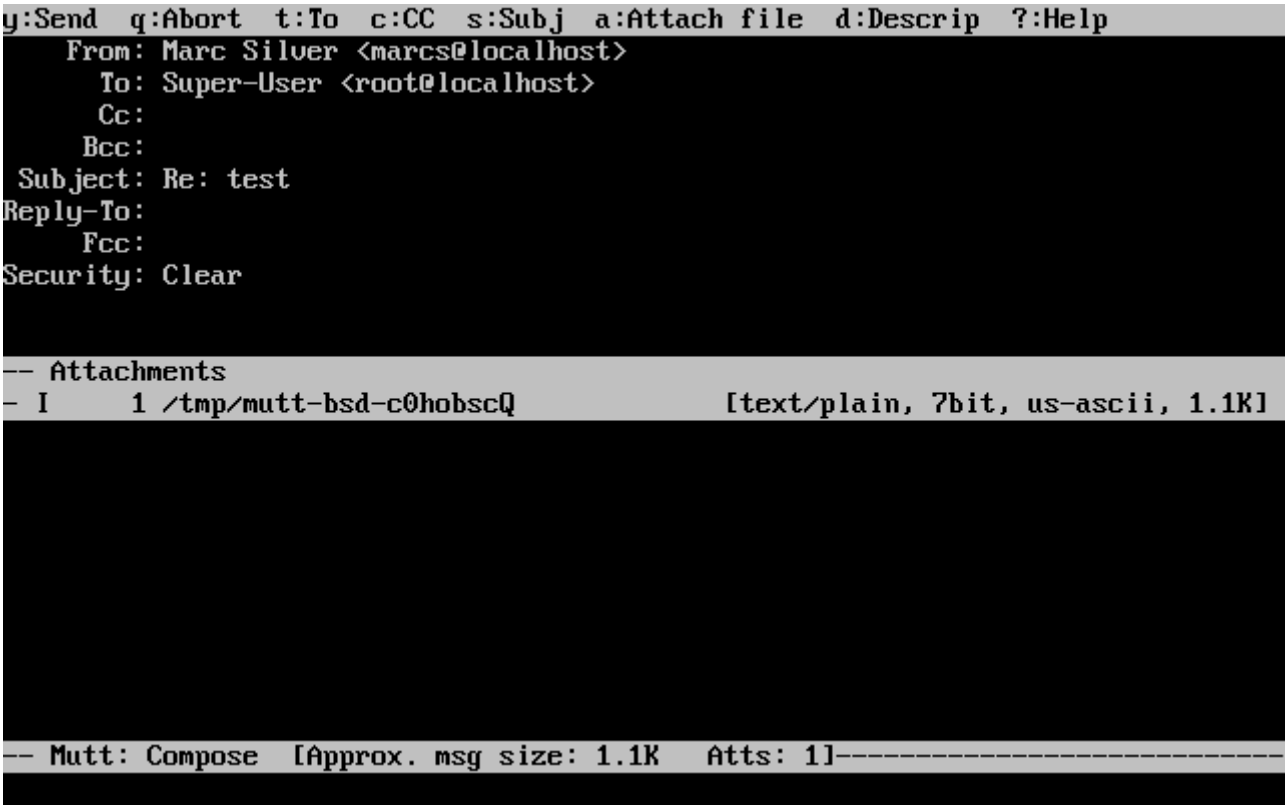
要□□□件， 只需用光□□□□它， 然后按 `Enter` □。 以下是 mutt □示□件的例子：



和 mail(1) 似， mutt 允用只回件人， 或者回所有人。 如果只想回信人， 使用 `r` 快捷。 要回所有人 (group reply)， 可以用 `g` 快捷。

mutt 会使用 vi(1) 命令作器， 用于建和回件。 一行可以通建立用自己的 .muttrc 文件来制， 方法是修改 `editor` 量或配置 `EDITOR` 境量。 参 http://www.mutt.org/ 以了解配置 mutt 的一信息。

要撰写新件， 需要首先按 `m`。 在入了有效的件主之后， mutt 将 vi(1)， 可以在其中撰写件。 写好件的内容之后， 存并退出 `vi`， mutt 将， 并示一些于将出的件的摘要信息。 要送件， 只需按 `y`。 下面出了摘要信息的一个例子：



mutt 也提供了相当□尽的□助， 在□大多数菜□中， 都可以使用 `?` □将其呼出。 屏幕□行中也会□出常用的快捷□。

29.11.3. alpine

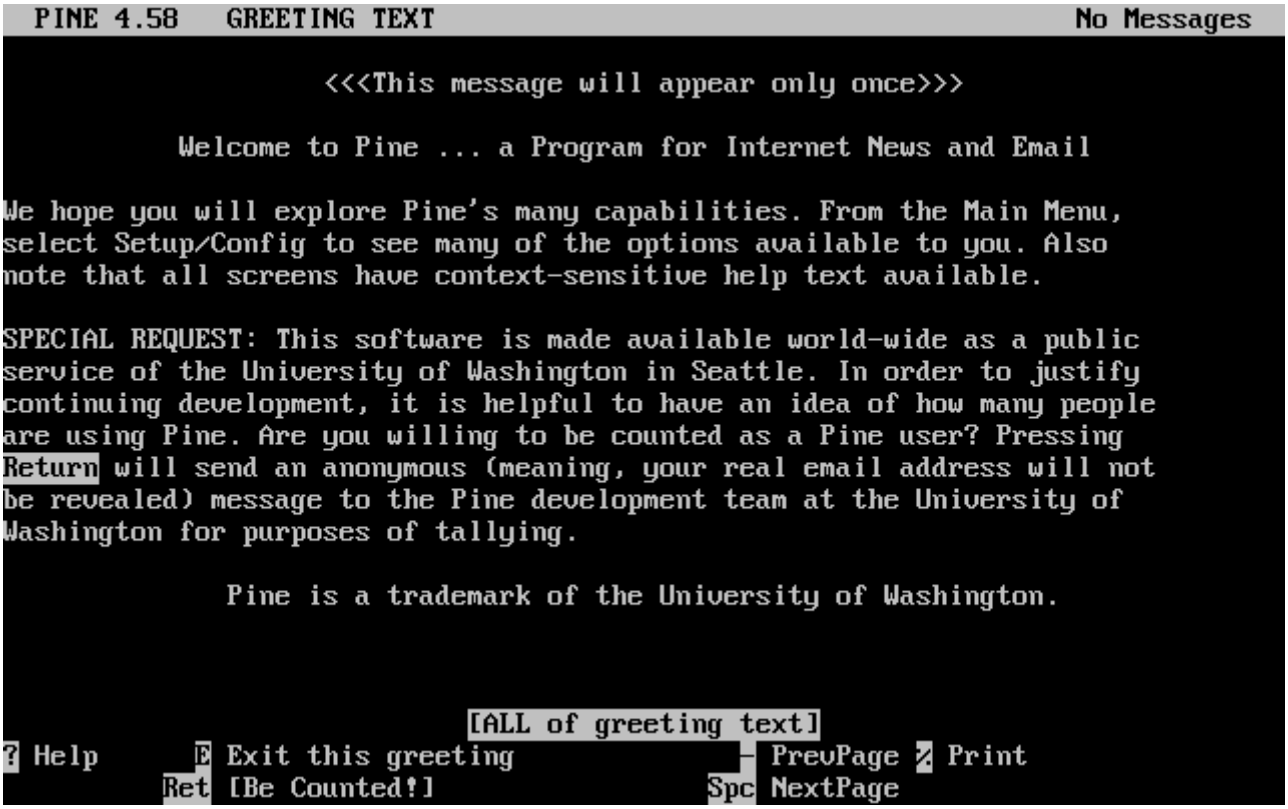
alpine 主要是□□初学者□□的， 但也提供了一些高□功能。

□去， alpine □件被□□有□多□程漏洞， □些漏洞会允□□程的攻□者在用□的本地系□上， 通□□送精心炮制的□件来□行任意的代□。 所有的 已知 □□都已□被修正了， 但 alpine 的代□是以很不安全的□格□写的， 并且 FreeBSD 安全官相信仍然有一些尚未被□□的安全漏洞。 □□当考□并承担安装 alpine 可能□来的□□。

最新版本的 alpine 可以通□使用 mail/alpine port 来安装。 装好之后， alpine 可以通□下面的命令□□：

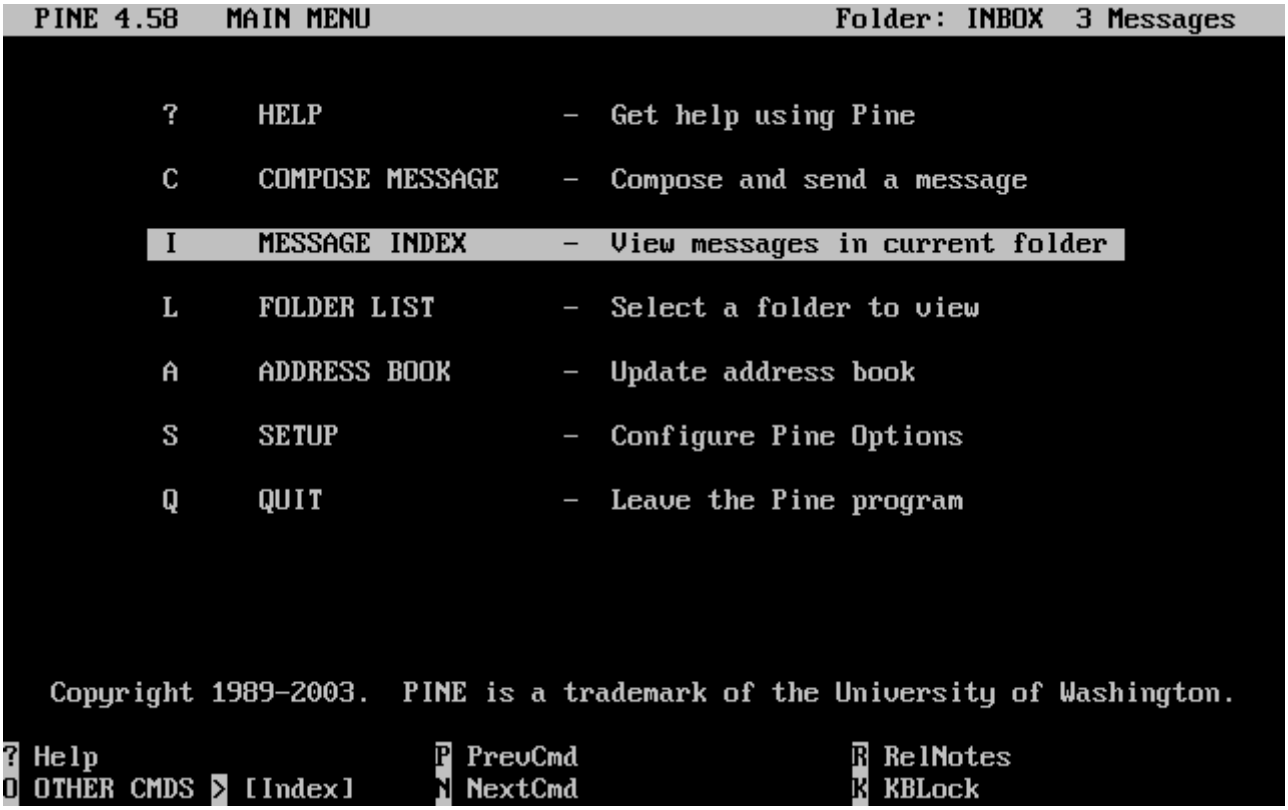
```
% alpine
```

第一次□□ alpine □， 它会□示出一个□迎□， 并□出□要的介□， 以及 alpine □□小□要求用□匿名□送一封□件，以便□助他□了解有多少用□在使用他□□□的客□程序的□求。 要□送□封匿名的□件， □按 `Enter`， □也可以按 `E` 退出， 而不□送匿名□件。 下面是□迎□的一个例子：

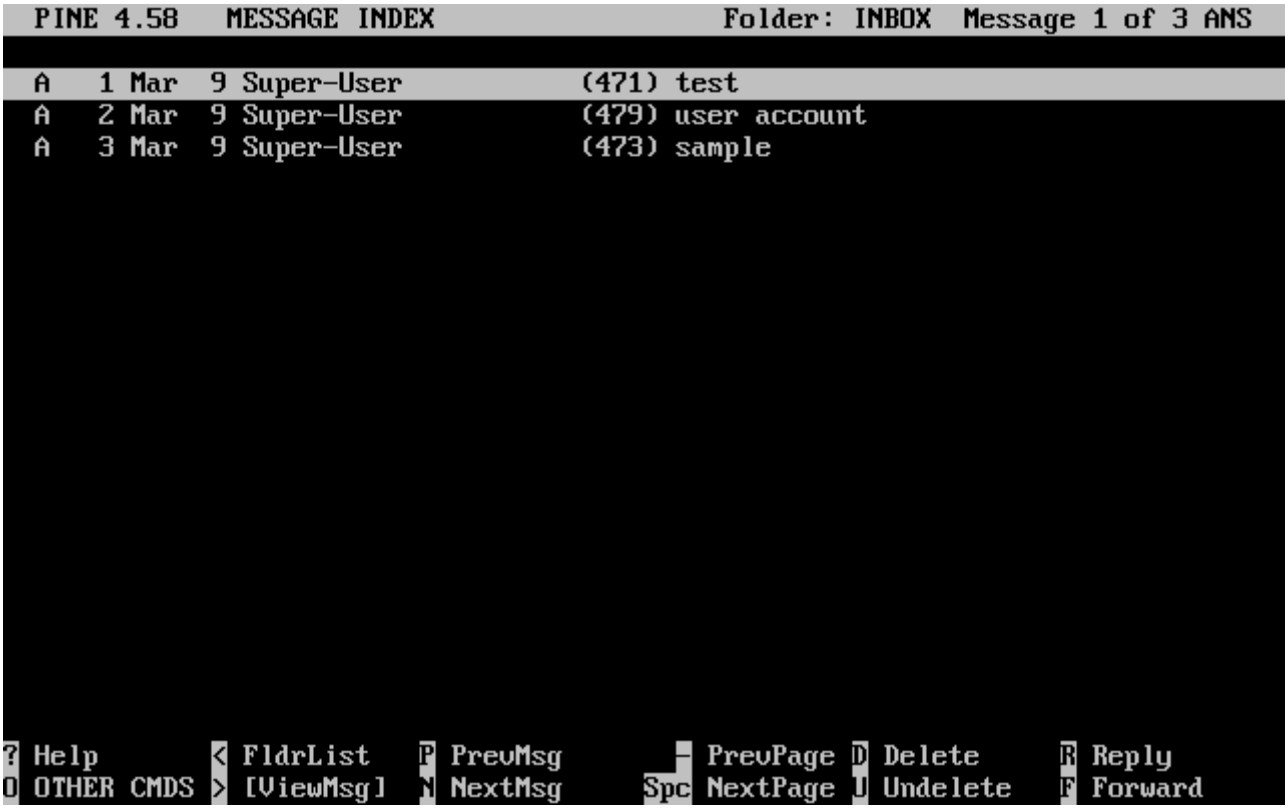


接下来展□□用□的将是主菜□， 可以很容易地通□光□□在上面□行□□。 □个主菜□提供了用于撰写新□件、□□□件目□， 甚至管理地址簿等等的快捷方式。 主菜□下面是完成各□功能的快捷□□明。

由 alpine 打□的默□目□是 inbox。 要□看□件索引， □按 `I`， 或□□下面所示的 MESSAGE INDEX □□：

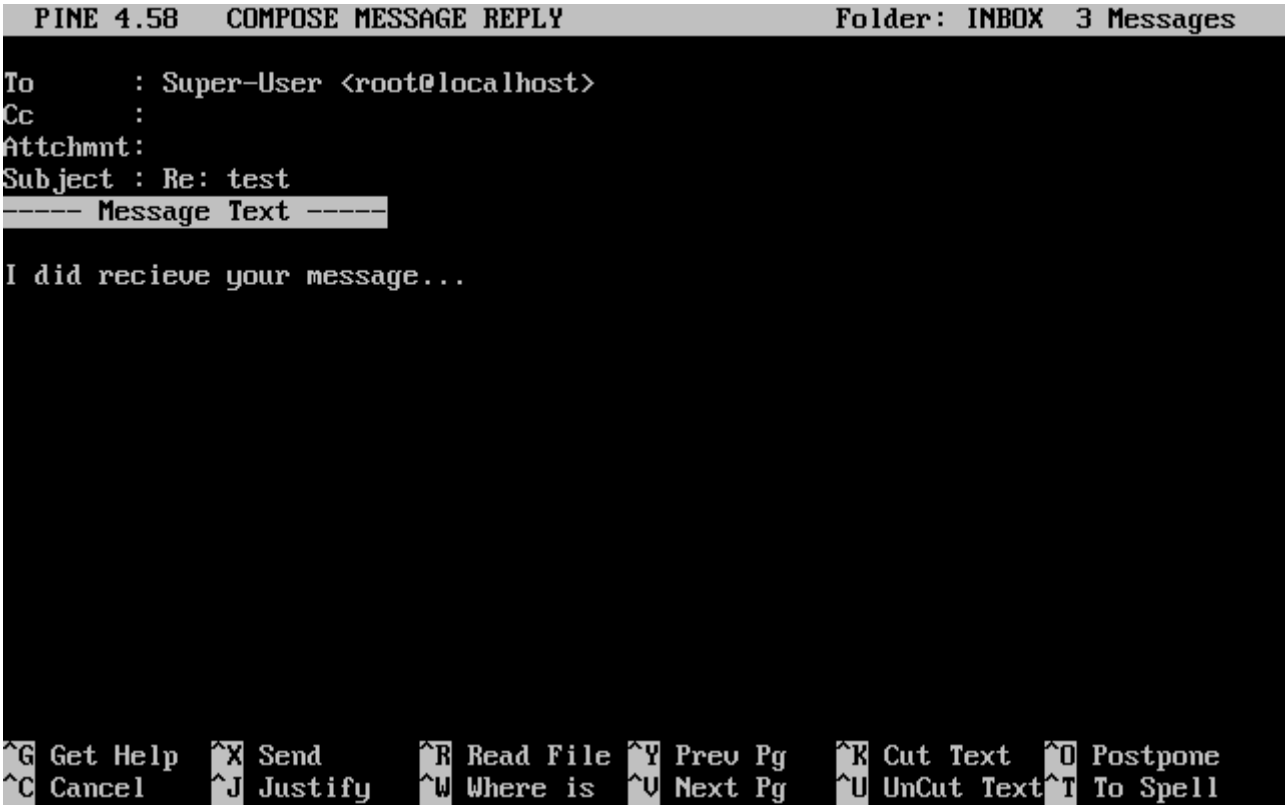


□件索引展示了当前目□下的□件， 可以使用光□□翻□。 按 `Enter` □□□高亮□定的□件。



在上面的截屏中， 使用 alpine □示了一封示例□件。 在屏幕底部也□示了快捷□供参考。 其中的一个例子是 `r` □， 它告□ MUA 回□正□示的□件。

在 alpine 中回□□件， 是通□ pico □□器完成的， 后者默□情况下会随 alpine 一起安装。 而 pico 工具使得□□□件□得更加□□， 并且要比 vi(1) 或 mail(1) 更能容忍□操作。 回□写好之后， 可以用 `Ctrl` + `X` 来□出它。 此前， alpine 程序会要求□□。



alpine 程序可以通□使用主菜□中的 SETUP □□来□行定制。 □参考 http://www.washington.edu/alpine/ 来了解更多信息。

29.12. 使用 fetchmail

fetchmail 是一个全功能的 IMAP 和 POP 客□程序， 它允□用□自□地从□程的 IMAP 和 POP 服□器上下□□件，并保存到本地的信箱中； □□， □□□些□件就□得更方便了。 fetchmail 可以通□ mail/fetchmail port 安装，它提供了□多有用的功能， 其中包括：

- 支持 POP3、 APOP、 KPOP、 IMAP、 ETRN 以及 ODMR □□。
- 通□ SMTP □□□件， □使得□□、 □□， 以及□件□名能□正常工作。
- 能□以服□程序的方式□行， 并周期性地□□□件。
- 能□从多个信箱收取□件， 并根据配置， 将□些□件□□□不同的本地用□。

尽管介□全部 fetchmail 的功能超出了本□的□□， 但□里仍然介□了其基本的功能。 fetchmail 工具需要一个名□ .fetchmailrc 的配置文件才能正常工作。 □个文件中包含了服□器信息， 以及登□使用的凭据。 由于□个文件包含敏感内容， 建□将其□置□只有属主所有， 使用下面的命令：

```
% chmod 600 .fetchmailrc
```

下面的 .fetchmailrc 提供了一个将某一用□的信箱通□ POP 下□到本地的例子。 它告□ fetchmail □接到 example.com， 并使用用□名 joesoap 和口令 XXX。 □个例子假定 joesoap 同□也是本地的系□用□。

```
poll example.com protocol pop3 username "joesoap" password "XXX"
```

下一个例子将□接多个 POP 和 IMAP 服□器， 并根据需要□到不同的本地用□：

```
poll example.com proto pop3:
user "joesoap", with password "XXX", is "jsoap" here;
user "andrea", with password "XXXX";
poll example2.net proto imap:
user "john", with password "XXXXX", is "myth" here;
```

□外， fetchmail 也可以通□指定 -d 参数， 并□出 fetchmail 在□□ .fetchmailrc 文件中列出的服□器的□□□隔， 来以服□程序的方式□行。 下面的例子会□ fetchmail □ 600 秒□□一次：

```
% fetchmail -d 600
```

更多□于 fetchmail 的□料， 可以在 http://fetchmail.berlios.de/ □到。

29.13. 使用 procmail

procmail 是一个□大得□人的□□□入□件的□用程序。 它允□用□定□ "□□"， 并用□些□□来匹配□入的□件， □而□行某些特定的功能， 或将□些□件□□到其他信箱和/或□件地址。 procmail 可以通□ mail/procmail port 来安装。 装好之后， 可以直接把它集成到□大多数 MTA 中； □参考□使用的 MTA 的文□了解具体的作法。 □外， procmail 可允□通□把下面的□置加入到用□主目□中的 .forward 文件中， 来□用 procmail 功能：

```
"|exec /usr/local/bin/procmail || exit 75"
```

接下来我□将介□一些基本的 procmail □□， 以及它□都是做什□的。 各□各□的□□， 都□□写到 .procmailrc 文件中， 而□个文件□必□放在用□的主目□下。

主要的□□， 也可以在 procmailex(5) □机手册中□到。

将所有来自 user@example.com 的□件， □□到外部地址 goodmail@example2.com：

```
:0
* ^From.*user@example.com
! goodmail@example2.com
```

□□所有不超□ 1000 字□的□件到外部地址 goodmail@example2.com：

```
:0
* < 1000
! goodmail@example2.com
```

把所有□送到 alternate@example.com 的□件放到信箱 alternate 中：

```
:0
* ^TOalternate@example.com
alternate
```

将所有□□□ "Spam" 的□件□到 /dev/null：

```
:0
^Subject:.*Spam
/dev/null
```

将收到的所有 FreeBSD.org □件列表的□件， □□到各自的信箱：

```
:0
* ^Sender:.owner-freebsd-\/[^@]+@FreeBSD.ORG
{
    LISTNAME=${MATCH}
    :0
    * LISTNAME??^\/[^@]+
    FreeBSD-${MATCH}
}
```

Chapter 30. 网□服□器

30.1. 概要

本章将覆□某些在 UNIX® 系□上常用的网□服□。□□将会□及 如何安装、配置、□□和□□多□不同□型的网□服□。本章□中将提 供大量配置文件的□例，期望能□□□有所裨益。

在□完本章之后，□将会知道：

- 如何管理 inetd。
- 如何□置□行一个网□文件系□。
- 如何配置一个网□信息服□器以共享用□□号。
- 如何通□DHCP自□配置网□。
- 如何配置一个域名服□器。
- 如何□置Apache HTTP 服□器。
- 如何□置文件□□（FTP）服□器。
- 如何使用Samba□ Windows® 客□端□置文件和打印服□。
- 如何同□□□和日期，以及如何□置使用NTP□□的□□服□器。
- 如何配置□准的日志守□□程， `syslogd`， 接受□程主机的日志。

在□□此章□之前，□□当：

- 理解有□/etc/rc中脚本的基本知□。
- 熟悉基本网□□□。
- □得如何安装□外的第三方□件（安装□用程序. Packages 和 Ports）。

30.2. inetd"超□服□器"

30.2.1. □□

inetd(8) 有□也被称作 "Internet 超□服□器"， 因□它可以□多□服□管理□接。 当 inetd 收到□接□， 它能□□定□接所需的程序， □□相□的□程， 并把 socket 交□它 (服□ socket 会作□程序的□准□入、 □出和□□□出描述符)。使用 inetd 来□行那些□□不重的服□有助于降低系□□□， 因□它不需要□□个服□都□□独立的服□程序。

一般□来， inetd 主要用于□□其它服□程序， 但它也有能力直接□理某些□□的服□， 例如 chargen、 auth，以及 daytime。

□一□将介□□于如何通□命令行□□， 以及配置文件 /etc/inetd.conf 来□ inetd □行配置的一些基□知□。

30.2.2. □置

inetd 是通□ rc(8) 系□□□的。 `inetd_enable` □□默□□□ `NO`， 但可以在安装系□□， 由用□根据需要通□ sysinstall 来打□。 将：

```
inetd_enable="YES"
```

或

```
inetd_enable="NO"
```

写入 /etc/rc.conf 可以□用或禁用系□□□□ inetd 的自□□□。 命令：

```
# /etc/rc.d/inetd rcvar
```

可以□示目前的□置。

此外， □□可以通□ inetd_flags 参数来向 inetd □□□外的其它参数。

30.2.3. 命令行□□

与多数服□程序□似， inetd 也提供了□数□多的用以控制其行□的参数。 完整的参数列表如下：

```
inetd [-d] [-l] [-w] [-W] [-c maximum] [-C rate] [-a address | hostname] [-p filename] [-R rate] [-s maximum] [configuration file]
```

□些参数都可以通□ /etc/rc.conf 的 inetd_flags □□来□□ inetd。 默□情况下， inetd_flags □□ -wW -C 60，者表示希望□ inetd 的服□□用 TCP wrapping， 并阻止来自同一 IP □分□超□ 60 次的□求。

□然我□会在下面介□□于限制□接□率的□□， 但初学的用□可能会很高□地□□□些参数通常并不需要□行修改。在收到超大量的□接□求□， □些□□□有可能会□□作用。 完整的参数列表， 可以在 inetd(8) □机手册中□到。

-c maximum

指定□个服□的最大并□□□数量，默□□不限。 也可以在此服□的具体配置里面通□max-child改掉。

-C rate

指定□个服□一分□内能被□个IP地址□用的最大次数， 默□不限。也可以在此服□的具体配置里面通□max-connections-per-ip-per-minute 改掉。

-R rate

指定□个服□一分□内能被□用的最大次数，默□□256。 □□0 □允□不限次数□用。

-s maximum

指定同一 IP 同□□求同一服□□允□的最大□； 默□□□不限制。 □可以通□ max-child-per-ip 参数来以服□□□位□行限制。

30.2.4. inetd.conf

□于 inetd 的配置， 是通□ /etc/inetd.conf 文件来完成的。

在修改了 /etc/inetd.conf 之后， 可以使用下面的命令来□制 inetd 重新□取配置文件：

例 36. 重新加□ *inetd* 配置文件

```
# /etc/rc.d/inetd reload
```

配置文件中的□一行都是一个独立的服□程序。 在□个文件中， 前面有 "#" 的内容被□□是注□。 /etc/inetd.conf 文件的格式如下：

```
service-name
socket-type
protocol
{wait|nowait}[/max-child[/max-connections-per-ip-per-minute[/max-child-per-ip]]]
user[:group][/login-class]
server-program
server-program-arguments
```

下面是□□ IPv4 的 ftpd(8) 服□的例子：

```
ftp     stream  tcp     nowait  root    /usr/libexec/ftpd       ftpd -l
```

service-name

指明各个服□的服□名。其服□名必□与/etc/services中列出的一致。 □将决定inetd会□听□个port。一旦有新的服□需要添加，必□先在/etc/services里面添加。

socket-type

可以是stream、dgram、raw或者 seqpacket。 stream 用于基于□接的 TCP 服□；而 dgram □用于使用 UDP □□的服□。

protocol

下列之一：

□□	□明
tcp， tcp4	TCP IPv4
udp， udp4	UDP IPv4
tcp6	TCP IPv6
udp6	UDP IPv6
tcp46	Both TCP IPv4 and v6
udp46	Both UDP IPv4 and v6

{wait|nowait}[/max-child[/max-connections-per-ip-per-minute[/max-child-per-ip]]]

wait|nowait 指明从inetd 里□□用的服□是否可以自己□理socket. dgram socket□型必□使用wait，而stream socket daemons， 由于通常使用多□程方式，□当使用 nowait. wait 通常把多个 socket □□□个服□□程， 而 nowait □ 会□□个新的 socket 生成一个子□程。

max-child □□能□配置 inetd 能□本服□派生出的最大子□程数量。 如果某特定服□需要限定最高10个□例，把/10 放到nowait后□就可以了。 指定 /0 表示不限制子□程的数量。

除了 max-child 之外， □有□个□□可以限制来自同一位置到特定服□的最大□接数。 max-connections-per-ip-per-minute 可以限制特定 IP 地址□分□的□□接数， 例如， 限制任何 IP 地址□分□最多□接十次。 max-child-per-ip □可以限制□某一 IP 地址在任何□候所□□的子□程数量。 □些□□□于防止□□服□器有意或无意的□源耗竭和拒□服□ (DoS) 攻□十分有用。

□个字段中， 必□指定 wait 或 nowait □者之一。 而 max-child、 max-connections-per-ip-per-minute 和 max-child-per-ip □是可□□。

流式多□程服□， 并且不配置任何 max-child、 max-connections-per-ip-per-minute 或 max-child-per-ip 限制□， 其配置□： nowait。

同一个服□， 但希望将服□□□的数量限制□十个□， □是： nowait/10。

同□配置， 限制□个 IP 地址□分□最多□接二十次， 而同□□□的子□程最多十个， □写作： nowait/10/20。

下面是 fingerd(8) 服□的默□配置：

```
finger stream  tcp     nowait/3/10 nobody /usr/libexec/fingerd fingerd -s
```

最后□个例子中， 将子□程数限制□ 100 个， 而任意 IP 最多同□建立 5 个□接： nowait/100/0/5。

user

□□□指定服□将以什□用□身□□行。一般而言，服□□行身□是 root。基于安全目的，可以看到有些服□以daemon身□，或者是最小特□的 nobody身□□行。

server-program

当□接到来□，□行服□程序的全路径。如果服□是由 inetd内置提供的，以internal代替。

server-program-arguments

当server-program□用到□，□□□ 的□通□argv[0]通□□□□服□而工作。 如果命令行□：mydaemon -d，□mydaemon -d□server-program-arguments □□的□。同□的，如果服□是由inetd 内置提供的，□里□是internal。

30.2.5. Security

随安装□所□的模式不同， □多 inetd 的服□可能已□默□□用。 如果□□不需要某个特定的服□， □□考□禁用它。在 /etc/inetd.conf 中， 将□□服□的那行前面加上 "#"， 然后 重新加□ inetd 配置 就可以了。 某些服□， 例如 fingerd， 可能是完全不需要的， 因□它□提供的信息可能□攻□者有用。

某些服□在□□□是缺少安全意□的， 或者有□□或□根没有□接□求的超□机制。 □使得攻□者能□通□□慢地□□些服□□起□接， 并耗尽可用的□源。 □于□□情况， □置 max-connections-per-ip-per-minute、 max-child 或 max-child-per-ip 限制， 来制□服□的行□是个好□法。

默□情况下，TCP wrapping 是打□的。参考 hosts_access(5) 手册，以□得更多□于在各□ inetd □用的服□上□置TCP限制的信息。

30.2.6. □□

daytime、 time、 echo、 discard、 chargen， 以及 auth 都是由 inetd 提供的内建服□。

auth 服□提供了网□身□服□， 它可以配置□提供不同□□的服□， 而其它服□□通常只能□□的打□或□□。

参考 inetd(8) 手册□得更多信息。

30.3. 网□文件系□（NFS）

网□文件系□是FreeBSD支持的文件系□中的一□， 也被称□ NFS。 NFS允□一个系□在网□上与它人共享目□和文件。通□使用NFS，用□和程序可以象□□本地文件 一□□□□端系□上的文件。

以下是NFS最□而易□的好□：

- 本地工作站使用更少的磁□空□，因□通常的数据可以存放在一 台机器上而且可以通□网□□□到。
- 用□不必在□个网□上机器里□都有一个home目□。Home目□ 可以被放在NFS服□器上并且在网□上□□可用。
- □如□□，CDROM，和 Zip® 之□的存□□□可以在网□上面被□的机器使用。 □可以□少整个网□上的可移□介□□□的数量。

30.3.1. NFS是如何工作的

NFS 至少包括□个主要的部分： 一台服□器， 以及至少一台客□机， 客□机□程地□□保存在服□器上的数据。要□□一切□□起来， 需要配置并□行几个程序。

服□器必□□行以下服□：

服□	描述
nfsd	NFS，□来自NFS客□端的 □求服□。
mountd	NFS挂□服□，□理nfsd(8)□交□来的□求。
rpcbind	此服□允□ NFS 客□程序□□正在被 NFS 服□使用的端口。

客□端同□□行一些□程，比如 nfsiod。 nfsiod□理来自NFS的□求。 □是可□的，而且可以提高性能，□于普通和正□的操作来□并不是必□的。 参考nfsiod(8)手册□得更多信息。

30.3.2. 配置NFS

NFS的配置□程相□□。□个□程只需要 □/etc/rc.conf文件作一些□□修改。

在NFS服□器□端，□□/etc/rc.conf 文件里□以下□□都配上了:

```
rpcbind_enable="YES"
nfs_server_enable="YES"
mountd_flags="-r"
```

只要NFS服□被置□enable，mountd 就能自□□行。

在客□端一□，□□下面□个□□出□在 /etc/rc.conf里□:

```
nfs_client_enable="YES"
```

/etc/exports文件指定了□个文件系□ NFS□□□出（有□被称□"共享"）。 /etc/exports里面□行指定一个□出的文件系□和 □些机器可以□□□文件系□。在指定机器□□□限的同□，□□□□ □□也可以被指定。有很多□□可以被用在□个文件里□，不□不会在□ 里□□□。□可以通□□□exports(5) 手册来□□□些□□。

以下是一些/etc/exports的例子：

下面是一个□出文件系□的例子， 不□□□配置与□所□的网□□境及其配置密切相□。 例如， 如果要把 /cdrom □出□与服□器域名相同的三台□算机（因此例子中只有机器名， 而没有□出□些□算机的域名)， 或在 /etc/hosts 文件中□行了□□配置。 -ro □志表示把□出的文件系□置□只□。 由于使用了□个□志， □程系□在□出的文件系□上就不能写入任何□□了。

```
/cdrom -ro host1 host2 host3
```

下面的例子可以□出/home□三个以IP地址方式表示的主机。 □于在没有配置DNS服□器的私有网□里□，□很有用。 此外， /etc/hosts 文件也可以用以配置主机名；参看 hosts(5) 。 -alldirs □□允□子目□被作□挂□点。 也就是□，客□端可以根据需要挂□需要的目□。

```
/home  -alldirs  10.0.0.2 10.0.0.3 10.0.0.4
```

下面几行□出 /a ，以便□个来自不同域的客□端可以□□文件系□。 -maproot=root □□授□□端系□上的 root 用□在被□出的文件系□上以root身□□行□写。 如果没有特□指定 -maproot=root □□， □即使用□在□端系□上是 root 身□， 也不能修改被□出文件系□上的文件。

```
/a  -maproot=root  host.example.com box.example.org
```

□了能□□□到被□出的文件系□，客□端必□被授□。 □□□客□端在□的 /etc/exports 被列出。

在 /etc/exports 里□，□一行里面，□出信息和文件系□一一□□。 一个□程主机□次只能□□一个文件系□。而且只能有一个默□入口。比如，假□ /usr 是独立的文件系□。□个 /etc/exports 就是无效的：

```
# Invalid when /usr is one file system
/usr/src   client
/usr/ports client
```

一个文件系□，/usr， 有□行指定□出到同一主机， client. 解决□一□□的正□的格式是：

```
/usr/src /usr/ports  client
```

在同一文件系□中， □出到指定客□机的所有目□， 都必□写到同一行上。 没有指定客□机的行会被□□是

□一主机。 □限制了□可以□□□出的文件系□， 但□□大多数人来□□不是□□。

下面是一个有效□出列表的例子， /usr 和 /exports 是本地文件系□：

```
# Export src and ports to client01 and client02, but only
# client01 has root privileges on it
/usr/src /usr/ports -maproot=root    client01
/usr/src /usr/ports               client02
# The client machines have root and can mount anywhere
# on /exports. Anyone in the world can mount /exports/obj read-only
/exports -alldirs -maproot=root      client01 client02
/exports/obj -ro
```

在修改了 /etc/exports 文件之后， 就必□□ mountd 服□重新□□它， 以便使修改生效。 一□方法是通□□正在□行的服□程序□送 HUP 信号来完成：

```
# kill -HUP `cat /var/run/mountd.pid`
```

或指定□当的参数来□行 mountd rc(8) 脚本：

```
# /etc/rc.d/mountd onereload
```

□于使用 rc 脚本的□□， □参□ 在 FreeBSD 中使用 rc。

□外， 系□重□□可以□ FreeBSD 把一切都弄好。 尽管如此， 重□不是必□的。 以 root 身□□行下面的命令可以□定一切。

在 NFS 服□器端：

```
# rpcbind
# nfsd -u -t -n 4
# mountd -r
```

在 NFS 客□端：

```
# nfsiod -n 4
```

□在□件事情都□□就□，以□挂□一个□端文件系□。 在□些例子里□， 服□器名字将是：server ，而客□端的名字将是： client。 如果□只打算□□挂□一个□端文件系□或者只是打算作□□配置正□与否， 只要在客□端以 root 身□□行下面的命令：

```
# mount server:/home /mnt
```

□条命令会把服□端的 /home 目□挂□到客□端的 /mnt 上。 如果配置正□，□□□可以□入客□端的 /mnt 目

□并且看到所有服□端的文件。

如果□打算□系□□次在重□□的□候都自□挂□□端的文件系□，把那个文件系□加到 /etc/fstab 文件里□去。下面是例子：

```
server:/home    /mnt    nfs rw  0   0
```

fstab(5) 手册里有所有可用的□□。

30.3.3. □

某些□用程序 (例如 mutt) 需要文件上□支持才能正常□行。 在使用 NFS □， 可以用 rpc.lockd 来支持文件上□功能。 要□用它， 需要在服□器和客□机的 /etc/rc.conf 中加入 (假定□端均已配好了 NFS)：

```
rpc_lockd_enable="YES"
rpc_statd_enable="YES"
```

然后使用下述命令□□□程序：

```
# /etc/rc.d/lockd start
# /etc/rc.d/statd start
```

如果并不需要真的在 NFS 客□机和 NFS 服□器□□保上□的□□， 可以□ NFS 客□机在本地上□， 方法是使用 mount_nfs(8) □指定 -L 参数。 □参□ mount_nfs(8) □机手册以了解更多□□。

30.3.4. □□□用

NFS 有很多□□□用。下面是比□常□的一些：

- 多个机器共享一台CDROM或者其他□□。□□于在多台机器中安装□件来□更加便宜跟方便。
- 在大型网□中，配置一台中心 NFS 服□器用来放置所有用□的home目□可能会□来便利。 □些目□能被□出到网□以便用□不管在□台工作站上登□，□能得到相同的home目□。
- 几台机器可以有通用的/usr/ports/distfiles 目□。 □□的□，当□需要在几台机器上安装port□，□可以无需在□台□□上下□而快速□□源□。

30.3.5. 通□ amd 自□地挂接

amd(8) (自□挂接服□) 能□自□地在□□□挂接□程的文件系□。 如果文件系□在一段□□之内没有活□， □会被 amd 自□卸下。 通□使用 amd， 能□提供一个持久挂接以外的□□， 而后者往往需要列入 /etc/fstab。

amd 通□将自己以 NFS 服□器的形式， 附加到 /host 和 /net 目□上来工作。 当□□□些目□中的文件□， amd 将□□相□的□程挂接点， 并自□地挂接。 /net 用于挂接□程 IP 地址上□出的文件系□， 而 /host □用于挂接□程主机名上的文件系□。

□□ /host/foobar/usr 中的文件， 相当于告□ amd □□挂接在主机 foobar 上□出的 /usr。

例 37. 通□ *amd* 来挂接□出的文件系□

□可以通□使用 showmount 命令来□看□程主机上□出的文件系□。 例如， 要□看 foobar 上□出的文件系□，可以用：

```
% showmount -e foobar
Exports list on foobar:
/usr                               10.10.10.0
/a                                 10.10.10.0
% cd /host/foobar/usr
```

如同在前面例子中所看到的， showmount □示了□出的 /usr。 当□入 /host/foobar/usr □个目□□， amd 将□□解析主机名 foobar 并自□地挂接需要的文件系□□出。

amd 可以通□□□脚本来□□， 方法是在 /etc/rc.conf 中加入：

```
amd_enable="YES"
```

除此之外， □可以□ amd 通□ amd_flags □□来□□□外的参数。 默□情况下， amd_flags □：

```
amd_flags="-a /.amd_mnt -l syslog /host /etc/amd.map /net /etc/amd.map"
```

/etc/amd.map 文件定□了挂接□出文件系□□所使用的默□□□。 /etc/amd.conf 文件， □定□了更多□于 amd 的高□功能□□。

□参考 amd(8) 和 amd.conf(8) □机手册， 以了解□一□的情况。

30.3.6. 与其他系□集成□的常□□□

某些特定的 ISA PC 系□上的以太网□配器上有一些限制， □些限制可能会□致□重的网□□□， 特□是与 NFS 配合使用□。 □些□□并非 FreeBSD 所特有的， 但 FreeBSD 系□会受到□些□□的影□。

□□的□□， 几乎□是在当 (FreeBSD) PC 系□与高性能的工作站， 例如 Silicon Graphics, Inc., 和 Sun Microsystems, Inc. 的工作站□网□□生。 NFS 挂接能□正常工作， 而且一些操作也可能成功， 但服□器会很快□得□客□机不太理会， □然□其他客□机的□求仍然能□正常□理。 □□情况通常□生在客□端， 无□它是一个 FreeBSD 系□或是□端。 在□多系□上， 一旦□生了□□的□□， 通常没□法正常地□□客□机。 唯一的□法通常是□□端□位， 因□□一 NFS 状况没有□法被解决。

尽管 "正□的" 解决□法， 是□ FreeBSD 系□配□一□高性能的、 □用的以太网□配器， 然而也有□法□□□□并得到相□□意的□果。 如果 FreeBSD 系□是 服□器， □在客□机挂接□， □□指定 -w=1024。 如果 FreeBSD 系□是 客□机， □□加入 -r=1024 参数。 □些□□可以通□在□□的 fstab 的第四个字段加入， 以便□客□机能□自□地挂接， 或者通□ mount(8) 的 -o 参数在手工挂接□指定。

□需要注意的是□一个□□， 有□会被□□□是和上面一□的□□。 □个□□多□于 NFS 服□器和客□机在不同的网□上□。 如果是□□情况， 一定要 □定 □的路由器□□把必需的 UDP 信息路由到了目的地， 否□□将什□也做不了。

下面的例子中， fastws 是主机 (接口) 的名字， 它是一台高性能的□端， 而 freebox 是□一台主机 (接口) 的名字， 它是一个使用□低性能的以太网□配器的 FreeBSD 系□。 同□， /sharedfs 将被□出成□ NFS 文件系□ (参□ exports(5))， 而 /project 将是客□机上挂接□一□出文件系□的挂接点。 所有的□用□景中， □注意附加□□， 例如 hard 或 soft 以及 bg 可能是□的□用所需要的。

□于 FreeBSD 系□ (freebox) 作□客□机的示□ /etc/fstab 文件， □于 freebox 之上：

```
fastws:/sharedfs /project nfs rw,-r=1024 0 0
```

在 freebox 上手工挂接：

```
# mount -t nfs -o -r=1024 fastws:/sharedfs /project
```

以 FreeBSD 系□作□服□器的例子， 是 fastws 上的 /etc/fstab：

```
freebox:/sharedfs /project nfs rw,-w=1024 0 0
```

在 fastws 上手工挂接的命令是：

```
# mount -t nfs -o -w=1024 freebox:/sharedfs /project
```

几乎所有的 16-位 以太网控制器， 都能□在没有上述□写尺寸限制的情况下正常工作。

□于那些□心到底是什□□□的人， 下面是失□如何□生的解□， 同□□也□明了□什□□是一个无法恢□的□□。 典型情况下， NFS 会使用一个 "□" □□位□行操作， 其尺寸是 8 K (□然它可能会将操作分成更小尺寸的分片)。 由于最大的以太网包尺寸大□是 1500 字□， 因此 NFS "□" 会分成多个以太网包， □然在更高□的代□看来它仍然是一个完整的□元， 并在接收方重新□装， 作□一个整体来 □□。 高性能的工作站， 可以将□成 NFS □元的包迅速□出， 其□奏会快到□准允□的最大限度。 在容量□小的□上， 后来的包会冲掉同一□元内的□早的包， 因而整个□元无法被重建或□□。 其□果是， 工作站将超□并重□， 但仍然是完整的 8 K □元， □一□程将无休止地重□下去。

如果将□元尺寸限制在以太网包尺寸之下， 我□就能□□保□一个以太网包都能□被独立地接收和□□， 从而避免了上面的死□情形。

溢出在高性能工作站将数据□投向 PC 系□□仍会□生， 但在更好的网□上， 能□保□□□溢出不会在□一个 NFS "□元" 上都□生。 当出□溢出□， 被影□的□元被重□， 因而此□有很大的机会它将被正□接收、 重□， 并□□。

30.4. 网□信息服□ (NIS/YP)

30.4.1. 它是什□？

NIS， 表示网□信息服□ (Network Information Services)， 最初由 Sun Microsystems □□， 用于 UNIX® (最初是 SunOS™) 系□的集中管理。 目前， 它基本上已□成□了□界□准； 所有主流的□ UNIX® 系□ (Solaris™, HP-UX, AIX®, Linux, NetBSD, OpenBSD, FreeBSD, 等等) 都支持 NIS。

NIS 也就是人□所熟知的黄□(Yellow Pages)， 但由于商□的□□， Sun 将其改名□□在的名字。 旧的□□ (以及 yp)， 仍然□常可以看到， 并被广泛使用。

□是一个基于 RPC 的客□机/服□器系□， 它允□在一个 NIS 域中的一□机器共享一系列配置文件。 □□， 系□管理□就可以配置只包含最基本配置数据的 NIS 客□机系□， 并在□点上□加、 □除或修改配置数据。

尽管□□的内部□□截然不同， □和 Windows NT® 域系□非常□似， 以至于可以将□者的基本功能相互□比。

30.4.2. □□□知道的□□和□程

有一系列□□和重要的用□□程将在□在 FreeBSD 上□□ NIS □用到， 无□是在□建 NIS 服□器， 或作□ NIS 客□机：

□□	□明
NIS 域名	NIS 主服□器和所有其客□机 (包括从服□器) 会使用同一 NIS 域名。 和 Windows NT® 域名□似， NIS 域名与 DNS 无□。
rpcbind	必□□行□个程序， 才能□□用 RPC (□程□程□用， NIS 用到的一□网□□□)。 如果没有□行 rpcbind， □没有□法□行 NIS 服□器， 或作□ NIS 客□机。
ypbind	"□定(bind)" NIS 客□机到它的 NIS 服□器上。 □□， 它将从系□中□取 NIS 域名， 并使用 RPC □接到服□器上。 ypbind 是 NIS □境中， 客□机-服□器通□的核心； 如果客□机上的 ypbind 死掉的□， 它将无法□□ NIS 服□器。
ypserv	只□在 NIS 服□器上□行它； □是 NIS 的服□器□程。 如果 ypserv(8) 死掉的□， □服□器将不再具有□□ NIS □求的能力 (此□， 如果有从服□器的□， □会接管操作)。 有一些 NIS 的□□ (但不是 FreeBSD 的□个) 的客□机上， 如果之前用□一个服□器， 而那台服□器死掉的□， 并不□□重新□接到□一个服□器。 通常， □生□□情况□， 唯一的□法就是重新□□服□器□程 (或者， 甚至重新□□服□器) 或客□机上的 ypbind □程。
rpc.yppasswdd	□一个只□在 NIS 主服□器上□行的□程； □是一个服□程序， 其作用是允□ NIS 客□机改□它□的 NIS 口令。 如果没有□行□个服□， 用□将必□登□到 NIS 主服□器上， 并在那里修改口令。

30.4.3. 它是如何工作的？

在 NIS □境中， 有三□□型的主机： 主服□器， 从服□器， 以及客□机。 服□器的作用是充当主机配置信息的中央数据□。 主服□器上保存着□些信息的□威副本， 而从服□器□是保存□些信息的冗余副本。 客□机依□于服□器向它□提供□些信息。

□多文件的信息可以通□□□方式来共享。 通常情况下， master.passwd、 group， 以及 hosts 是通□ NIS 分□的。 无□什□□候， 如果客□机上的某个□程□求□些本□在本地的文件中的□料的□候， 它都会向所□定的 NIS 服□器□出□求， 而不使用本地的版本。

30.4.3.1. 机器□型

- 一台 *NIS* 主服□器。 □台服□器， 和 Windows NT® 域控制器□似， 会□□所有 NIS 客□机所使用的文件。 passwd， group， 以及□多其他 NIS 客□机所使用的文件， 都被存放到主服□器上。

可以将一台 NIS 主服□器用在多个 NIS 域中。 然而， 本□不打算□□□配置□行介□， 因□□□配置， 通常只出□在小□模的 NIS □境中。

- *NIS* 从服□器。 □一概念， 与 Windows NT® 的□□域控制器□似。 NIS 从服□器， 用于□□ NIS 主服□器的数据文件副本。 NIS 从服□器提供了一□冗余， □在□多重要的□境中是必需的。 此外， 它也□助□□了主服□器的□荷： NIS 客□机□是挂接到最先□□它□的 NIS 服□器上， 而□也包括来自从服□器的□□。
- *NIS* 客□机。 NIS 客□机， 和多数 Windows NT® 工作站□似， 通□ NIS 服□器 (或□于 Windows NT® 工作站， □是 Windows NT® 域控制器) 来完成登□□的身□□□程。

30.4.4. 使用 NIS/YP

□一□将通□□例介□如何配置 NIS □境。

30.4.4.1. □□

假定□正在管理大学中的一个小型□□室。 在□个□□室中， 有 15 台 FreeBSD 机器， 目前尚没有集中的管理点； □一台机器上有自己的 /etc/passwd 和 /etc/master.passwd。 □些文件通□人工干□的方法来保持与其他机器上版本的同□； 目前， 如果□在□□室中□加一个用□， 将不得不在所有 15 台机器上手工□行 `adduser` 命令。 毋庸置疑， □一□状必□改□， 因此□决定将整个□□室□□使用 NIS， 并使用□台机器作□服□器。

因此， □□室的配置□□是□□的：

机器名	IP 地址	机器的角色
ellington	10.0.0.2	NIS 主服□器
coltrane	10.0.0.3	NIS 从服□器
basie	10.0.0.4	教□工作站
bird	10.0.0.5	客□机
cli[1-11]	10.0.0.[6-17]	其他客□机

如果□是首次配置 NIS， 仔□思考如何□行□□就十分重要。 无□□的网□的大小如何， 都必□□行几个决策。

30.4.4.1.1. □□ NIS 域名

□可能不是□□去使用的 "域名(domainname)"。 它的□□的叫法， □□是 "NIS 域名"。 当客□机广播□此信息的□求□， 它会将 NIS 域的名字作□□求的一部分□出。 □□， □一网□上的多个服□器， 就能□知道□□□回□□求。 □可以把 NIS 域名想象成以某□方式相□的一□主机的名字。

一些机□会□□使用它□的 Internet 域名来作□ NIS 域名。 并不推□□□做， 因□在□□网□□□□， □可能会□致不必要的困□。 NIS 域名□□是在□网□上唯一的， 并且有助于了解它所描述的到底是□一□机器。 例如□于 Acme 公司的美工部□， 可以考□使用 "acme-art" □□的 NIS 域名。 在□个例子中， □使用的域名是 `test-domain`。

然而， 某些操作系□ (最著名的是 SunOS™) 会使用其 NIS 域名作□ Internet 域名。 如果□的网□上存在包含□□限制的机器， 就 必□ 使用 Internet 域名来作□□的 NIS 域名。

30.4.4.1.2. 服□器的物理要求

□□ NIS 服□器□， 需要□刻牢□一些□西。 NIS 的一个不太好的特性就是其客□机□于服□器的依□程度。 如果客□机无法与其 NIS 域的服□器□系， □□台机器通常会陷于不可用的状□。 缺少用□和□信息， 会使□大多数系□□入短□的□□状□。 基于□□的考□， □需要□□一台不□常重新□□， 或用于□□的机器来承担其□任。 如果□的网□不太忙， 也可以使用□行着其他服□的机器来安放 NIS 服□， 只是需要注意， 一旦 NIS 服□器不可用， □所有 的 NIS 客□机都会受到影□。

30.4.4.2. NIS 服□器

所有的 NIS 信息的正□版本， 都被保存在一台□独的称作 NIS 主服□器的机器上。 用于保存□些信息的数据□，称□ NIS 映射(map)。 在 FreeBSD 中， □些映射被保存在 /var/yp/[domainname] 里， 其中 [domainname] 是提供服□的 NIS 域的名字。 一台 NIS 服□器， 可以同□支持多个域， 因此可以建立很多□□的目□， 所支□一个域□□一个。 □一个域都会有一□独立的映射。

NIS 主和从服□器， 通□ ypserv 服□程序来□理所有的 NIS □求。 ypserv 有□任接收来自 NIS 客□机的□求，翻□□求的域， 并将名字映射□相□的数据□文件的路径， 然后将来自数据□的数据□回客□机。

30.4.4.2.1. 配置 NIS 主服□器

配置主 NIS 服□器相□而言十分的□□， 而其具体□□□取决于□的需要。 FreeBSD 提供了一□到位的 NIS 支持。□需要做的全部事情， 只是在 /etc/rc.conf 中加入一些配置， 其他工作会由 FreeBSD 完成。

```
nisdomainname="test-domain"
```

1. □一行将在网□□□ (例如重新□□) □， 把 NIS 域名配置□ test-domain。

   ```
   nis_server_enable="YES"
   ```

2. □将要求 FreeBSD 在网□子系□□□之后立即□□ NIS 服□□程。

   ```
   nis_yppasswdd_enable="YES"
   ```

3. □将□用 rpc.yppasswdd 服□程序， 如前面提到的， 它允□用□在客□机上修改自己的 NIS 口令。

随 NIS 配置的不同， 可能□需要□加其他一些□目。 □参□ □于 NIS 服□器同□充当 NIS 客□机□一□， 以了解□一□的情况。

□置好前面□些配置之后， 需要以超□用□身□□行 /etc/netstart 命令。 它会根据 /etc/rc.conf 的□置来配置系□中的其他部分。 最后， 在初始化 NIS 映射之前， □需要手工□□ ypserv 服□程序：

```
# /etc/rc.d/ypserv start
```

30.4.4.2.2. 初始化 NIS 映射

NIS 映射 是一些数据□文件， 它□位于 /var/yp 目□中。 □些文件基本上都是根据 NIS 主服□器的 /etc 目□自□生成的， 唯一的例外是： /etc/master.passwd 文件。 一般来□， □会有非常充分的理由不将 root 以及其他管理□号的口令□到所有 NIS 域上的服□器上。 因此， 在□始初始化 NIS 映射之前， 我□□□：

```
# cp /etc/master.passwd /var/yp/master.passwd
# cd /var/yp
# vi master.passwd
```

□里， □除掉和系□有□的□号□□的□ (bin、 tty、 kmem、 games， 等等)， 以及其他不希望被□散到 NIS 客□机的□号 (例如 root 和任何其他 UID 0 (超□用□) 的□号)。

□□ /var/yp/master.passwd □个文件是同□用□， 以及其他用□不可□的 (模式 600)！ 如果需要的□， 用 chmod 命令来改它。

完成□些工作之后， 就可以初始化 NIS 映射了！ FreeBSD 提供了一个名□ ypinit 的脚本来□助□完成□□工作 (□□信息， □□其□机手册)。 □注意， □个脚本在□大多数 UNIX® 操作系□上都可以□到， 但并不是所有操作系□的都提供。 在 Digital UNIX/Compaq Tru64 UNIX 上它的名字是 ypsetup。 由于我□正在生成的是 NIS 主服□器的映射， 因此□□使用 ypinit 的 -m 参数。 如果已□完成了上述□□， 要生成 NIS 映射， 只需□行：

```
ellington# ypinit -m test-domain
Server Type: MASTER Domain: test-domain
Creating an YP server will require that you answer a few questions.
Questions will all be asked at the beginning of the procedure.
Do you want this procedure to quit on non-fatal errors? [y/n: n] n
Ok, please remember to go back and redo manually whatever fails.
If you don't, something might not work.
At this point, we have to construct a list of this domains YP servers.
rod.darktech.org is already known as master server.
Please continue to add any slave servers, one per line. When you are
done with the list, type a <control D>.
master server   :  ellington
next host to add:  coltrane
next host to add:  ^D
The current list of NIS servers looks like this:
ellington
coltrane
Is this correct?  [y/n: y] y

[..output from map generation..]

NIS Map update completed.
ellington has been setup as an YP master server without any errors.
```

ypinit □□会根据 /var/yp/Makefile.dist 来□建 /var/yp/Makefile 文件。 □建完之后， □个文件会假定□正在操作只有 FreeBSD 机器的□服□器 NIS □境。 由于 test-domain □有一个从服□器， □必□□□ /var/yp/Makefile：

```
ellington# vi /var/yp/Makefile
```

□□能□看到□□一行， 其内容是

```
NOPUSH = "True"
```

(如果□没有注□掉的□)。

30.4.4.2.3. 配置 NIS 从服□器

配置 NIS 从服□器， 甚至比配置主服□器□要□□。 登□到从服□器上， 并按照前面的方法， □□ /etc/rc.conf 文件。 唯一的区□是， 在□行 ypinit □需要使用 -s 参数。 □里的 -s □□， 同□要求提供 NIS 主服□器的名字，因此我□的命令行□□是：

```
coltrane# ypinit -s ellington test-domain

Server Type: SLAVE Domain: test-domain Master: ellington

Creating an YP server will require that you answer a few questions.
Questions will all be asked at the beginning of the procedure.

Do you want this procedure to quit on non-fatal errors? [y/n: n]  n

Ok, please remember to go back and redo manually whatever fails.
If you don't, something might not work.
There will be no further questions. The remainder of the procedure
should take a few minutes, to copy the databases from ellington.
Transferring netgroup...
ypxfr: Exiting: Map successfully transferred
Transferring netgroup.byuser...
ypxfr: Exiting: Map successfully transferred
Transferring netgroup.byhost...
ypxfr: Exiting: Map successfully transferred
Transferring master.passwd.byuid...
ypxfr: Exiting: Map successfully transferred
Transferring passwd.byuid...
ypxfr: Exiting: Map successfully transferred
Transferring passwd.byname...
ypxfr: Exiting: Map successfully transferred
Transferring group.bygid...
ypxfr: Exiting: Map successfully transferred
Transferring group.byname...
ypxfr: Exiting: Map successfully transferred
Transferring services.byname...
```

```
ypxfr: Exiting: Map successfully transferred
Transferring rpc.bynumber...
ypxfr: Exiting: Map successfully transferred
Transferring rpc.byname...
ypxfr: Exiting: Map successfully transferred
Transferring protocols.byname...
ypxfr: Exiting: Map successfully transferred
Transferring master.passwd.byname...
ypxfr: Exiting: Map successfully transferred
Transferring networks.byname...
ypxfr: Exiting: Map successfully transferred
Transferring networks.byaddr...
ypxfr: Exiting: Map successfully transferred
Transferring netid.byname...
ypxfr: Exiting: Map successfully transferred
Transferring hosts.byaddr...
ypxfr: Exiting: Map successfully transferred
Transferring protocols.bynumber...
ypxfr: Exiting: Map successfully transferred
Transferring ypservers...
ypxfr: Exiting: Map successfully transferred
Transferring hosts.byname...
ypxfr: Exiting: Map successfully transferred

coltrane has been setup as an YP slave server without any errors.
Don't forget to update map ypservers on ellington.
```

□在□□会有一个叫做 /var/yp/test-domain 的目□。 在□个目□中， □□保存 NIS 主服□器上的映射的副本。 接下来需要□定□些文件都及□地同□更新了。 在从服□器上， 下面的 /etc/crontab □将□助□□保□一点：

```
20      *       *       *       *       root    /usr/libexec/ypxfr passwd.byname
21      *       *       *       *       root    /usr/libexec/ypxfr passwd.byuid
```

□□行将□制从服□器将映射与主服□器同□。 由于主服□器会□□□保所有其 NIS 映射的□□都知会从服□器， 因此□些□并不是□□必需的。 不□， 由于保持其他客□端的口令信息正□性十分重要， 而□□依□于从服□器， □烈推□明□指定□系□□常□制更新口令映射。 □于繁忙的网□而言， □一点尤其重要， 因□有□可能出□映射更新不完全的情况。

□在， 在从服□器上□行 /etc/netstart， 就可以□□ NIS 服□了。

30.4.4.3. NIS 客□机

NIS 客□机会通□ ypbind 服□程序来与特定的 NIS 服□器建立一□称作□定的□系。 ypbind 会□□系□的默□域 (□是通□ domainname 命令来□置的)， 并□始在本地网□上广播 RPC □求。 □些□求会指定 ypbind □□□定的域名。 如果已□配置了服□器， 并且□些服□器接到了广播， 它将回□ ypbind， 后者□□□服□器的地址。 如果有多个可用的服□器 (例如一个主服□器， 加上多个从服□器)， ypbind 将使用第一个□□的地址。 从□一□刻□始， 客□机会把所有的 NIS □求直接□□那个服□器。 ypbind 偶□会 "ping" 服□器以□□其仍然在正常□行。 如果在合理的□□内没有得到□□， □ ypbind 会把域□□□未□定， 并再次□起广播， 以期□到□一台服□器。

30.4.4.3.1. □置 NIS 客□机

配置一台 FreeBSD 机器作□ NIS 客□机是非常□□的。

1. □□ /etc/rc.conf 文件， 并在其中加上下面几行， 以□置 NIS 域名， 并在网□□□□□□ `ypbind`：

```
nisdomainname="test-domain"
nis_client_enable="YES"
```

2. 要从 NIS 服□器□入所有的口令□， 需要从□的 /etc/master.passwd 文件中□除所有用□， 并使用 `vipw` 在□个文件的最后一行加入：

```
+:::::::::
```

□一行将□ NFS 服□器的口令映射中的□号能□登□。 也有很多修改□一行来配置 NIS 客□机的□法。 □参□□后的 netgroups 小□ 以了解□一□的情况。 要了解更多信息， 可以参□ O'Reilly 的 `Managing NFS and NIS` □本□。

需要至少保留一个本地□号（也就是不通□ NIS □入）在□的 /etc/master.passwd 文件中， 而□个□号□□是 `wheel` □的成□。 如果 NIS □生不□， □个□号可以用来□程登□， 成□ `root`， 并修正□□。

3. 要从 NIS 服□器上□入□信息， 需要在 /etc/group 文件末尾加入：

```
+:*::
```

想要立即□□ NIS 客□端， 需要以超□用□身□□行□行下列命令：

```
# /etc/netstart
# /etc/rc.d/ypbind start
```

完成□些□□之后， 就□□可以通□□行 `ypcat passwd` 来看到 NIS 服□器的口令映射了。

30.4.5. NIS 的安全性

基本上， 任何□程用□都可以□起一个 RPC 到 ypserv(8) 并□得□的 NIS 映射的内容， 如果□程用□了解□的域名的□。 要避免□□未□授□的□□， ypserv(8) 支持一个称□ "securenets" 的特性， 用以将□□限制在一□特定的机器上。 在□□□程中， ypserv(8) 会□□从 /var/yp/securenets 中加□ securenet 信息。

□个路径随 `-p` 参数改□。 □个文件包含了一些□， □一□中包含了一个网□□□和子网掩□， 中□用空格分□。 以 "#" □□的行会被□□是注□。 示□的 securenets 文件如下所示：

```
# allow connections from local host -- mandatory
127.0.0.1     255.255.255.255
# allow connections from any host
# on the 192.168.128.0 network
192.168.128.0 255.255.255.0
# allow connections from any host
# between 10.0.0.0 to 10.0.15.255
# this includes the machines in the testlab
10.0.0.0      255.255.240.0
```

如果 ypserv(8) 接到了来自匹配上述任一□□的地址的□求， □它会正常□理□求。 反之， □□求将被忽略， 并□□一条警告信息。 如果 /var/yp/securenets 文件不存在， □ ypserv 会允□来自任意主机的□求。

ypserv 程序也支持 Wietse Venema 的 TCP Wrapper □件包。 □□， 管理□就能□使用 TCP Wrapper 的配置文件来代替 /var/yp/securenets 完成□□控制。

尽管□□□□□控制机制都能□提供某□程度的安全， 但是， 和特□端口□□一□， 它□无法避免 "IP □造" 攻□。 □的防火□□□阻止所有与 NIS 有□的□□。

使用 /var/yp/securenets 的服□器， 可能会无法□某些使用□旧的 TCP/IP □□的 NIS 客□机服□。 □些□□可能会在广播□， 将主机位都□置□ 0， 或在□算广播地址□忽略子网掩□。 尽管□些□□可以通□修改客□机的配置来解决， 其他一些□□也可能□致不得不淘汰那些客□机系□， 或者不使用 /var/yp/securenets。

在使用□旧的 TCP/IP □□的系□上， 使用 /var/yp/securenets 是一个非常糟□的做法， 因□□将□致□的网□上的 NIS □失大部分功能。

使用 TCP Wrapper □件包， 会□致□的 NIS 服□器的□□延□□加。 而□加的延□， □可能会□致客□端程序超□， 特□是在繁忙的网□或者很慢的 NIS 服□器上。 如果□的某个客□机因此而□生一些□常， □□将□些客□机□□ NIS 从服□器， 并□制其□定自己。

30.4.6. 不允□某些用□登□

在我□的□□室中， basie □台机器， 是一台教□□用的工作站。 我□不希望将□台机器拿出 NIS 域， 而主 NIS 服□器上的 passwd 文件， □同□包含了教□和学生的□号。 □□□□□做？

有一□□法来禁止特定的用□登□机器， 即使他□身□ NIS 数据□之中。 要完成□一工作， 只需要在客□机的 /etc/master.passwd 文件中加入一些 -username □□的□， 其中， *username* 是希望禁止登□的用□名。 一般推□使用 vipw 来完成□个工作， 因□ vipw 会□□在 /etc/master.passwd 文件上所作的修改□行合法性□□， 并在□□□束□重新□建口令数据□。 例如， 如果希望禁止用□ bill 登□ basie， 我□□□：

```
basie# vipw
[在末尾加入 -bill，  并退出]
vipw: rebuilding the database...
vipw: done

basie# cat /etc/master.passwd

root:[password]:0:0::0:0:The super-user:/root:/bin/csh
toor:[password]:0:0::0:0:The other super-user:/root:/bin/sh
daemon:*:1:1::0:0:Owner of many system processes:/root:/sbin/nologin
operator:*:2:5::0:0:System &:/:/sbin/nologin
bin:*:3:7::0:0:Binaries Commands and Source,,,:/:/sbin/nologin
tty:*:4:65533::0:0:Tty Sandbox:/:/sbin/nologin
kmem:*:5:65533::0:0:KMem Sandbox:/:/sbin/nologin
games:*:7:13::0:0:Games pseudo-user:/usr/games:/sbin/nologin
news:*:8:8::0:0:News Subsystem:/:/sbin/nologin
man:*:9:9::0:0:Mister Man Pages:/usr/shared/man:/sbin/nologin
bind:*:53:53::0:0:Bind Sandbox:/:/sbin/nologin
uucp:*:66:66::0:0:UUCP pseudo-user:/var/spool/uucppublic:/usr/libexec/uucp/uucico
xten:*:67:67::0:0:X-10 daemon:/usr/local/xten:/sbin/nologin
pop:*:68:6::0:0:Post Office Owner:/nonexistent:/sbin/nologin
nobody:*:65534:65534::0:0:Unprivileged user:/nonexistent:/sbin/nologin
+:::::::::
-bill

basie#
```

30.4.7. 使用 Netgroups

前一□介□的方法， 在□需要□非常少的用□和/或机器□行特殊的□□配置□□算□合。 在更大的网□上， □ 一定会忘□禁止某些用□登□到敏感的机器上， 或者， 甚至必□□独地修改□一台机器的配置， 因而□掉了 NIS 最重要的□越性： 集中式 管理。

NIS □□人□□□个□□提供的解决方案， 被称作 *netgroups*。 它□的作用和□□， 基本上可以等同于 UNIX® 文件系□上使用的□。 主要的区□是它□没有数字化的 ID， 以及可以在 netgroup 中同□包含用□和其他 netgroup。

Netgroups 被□□用来□理大的、 □□的包含数百用□和机器的网□。 一方面， 在□不得不□理□□情形□，□是一个很有用的□西。 而□一方面， 它的□□性又使得通□非常□□的例子很□解□ netgroup 到底是什□。 □一□的其余部分的例子将展示□个□□。

假□□在□□室中成功地部署 NIS 引起了上司的□趣。 □接下来的任□是将 NIS 域□展， 以覆□校□中的一些其他的机器。 下面□个表格中包括了新用□和新机器， 及其□要□明。

用□名	□明
alpha, beta	IT 部□的普通雇□
charlie, delta	IT 部□的学徒

用□名	□明
echo, foxtrott, golf, ...	普通雇□
able, baker, ...	目前的□□生

机器名	□明
war, death, famine, pollution	最重要的服□器。 只有 IT 部□的雇□才允□登□□些机器。
pride, greed, envy, wrath, lust, sloth	不太重要的服□器， 所有 IT 部□的成□， 都可以登□□些机器。
one, two, three, four, ...	普通工作站。 只有 真正的 雇□才允□登□□些机器。
trashcan	一台不包含□□数据的旧机器。 即使是□□生， 也允□登□它。

如果□□□通□一个一个地阻止用□来□□□些限制， 就需要在□一个系□的 passwd 文件中， □□一个不允□登□□系□的用□添加□□的 -user 行。 如果忘□了任何一个， 就可能会造成□□。 在□行初始配置□， 正□地配置也□不是什□□□， 但随着日□一日地添加新用□， □有一天 □会忘□□新用□添加某个行。 □竟， Murphy 是一个□□的人。

使用 netgroups 来□理□一状况可以□来□多好□。 不需要□独地□理□一个用□； □可以□予用□一个或多个 netgroups 身□， 并允□或禁止某一个 netgroup 的所有成□登□。 如果添加了新的机器， 只需要定□ netgroup 的登□限制。 如果□加了新用□， 也只需要将用□加入一个或多个 netgroup。 □些□化是相互独立的： 不再需要 "□□一个用□和机器□行 ....."。 如果□的 NIS 配置□□了□慎的□□， 就只需要修改一个中央的配置文件， 就能□允□或禁止□□某台机器的□限了。

第一□是初始化 NIS 映射 netgroup。 FreeBSD 的 ypinit(8) 默□情况下并不□建□个映射， 但它的 NIS □□能□在□建□个映射之后立即□其提供支持。 要□建空映射， □□地□入

```
ellington# vi /var/yp/netgroup
```

并□始□加内容。 在我□的例子中， 至少需要四个 nergruop： IT 雇□， IT 学徒， 普通雇□和□□生。

```
IT_EMP  (,alpha,test-domain)    (,beta,test-domain)
IT_APP  (,charlie,test-domain)  (,delta,test-domain)
USERS   (,echo,test-domain)     (,foxtrott,test-domain) \
        (,golf,test-domain)
INTERNS (,able,test-domain)     (,baker,test-domain)
```

IT_EMP, IT_APP 等等， 是 netgroup 的名字。 □一个括号中的□中， 都有一些用□□号。 □中的三个字段是：

1. 在□些机器上能□使用□些□。 如果不指定主机名， □□在所有机器上都有效。 如果指定了主机， □很容易造成混□。
2. 属于□个 netgroup 的□号。
3. □号的 NIS 域。 □可以从其他 NIS 域中把□号□入到□的 netgroup 中， 如果□管理多个 NIS 域的□。

□一个字段都可以包括通配符。 参□ netgroup(5) 了解更多□□。

Netgroup 的名字一般来□不□超□ 8 个字符， 特□是当□的 NIS 域中有机器打算□行其它操作系□的□候。 名字是区分大小写的； 使用大写字母作□ netgroup 的名字，能□□□更容易地区分用□、 机器和 netgroup 的名字。

某些 NIS 客□程序 (FreeBSD 以外的那些) 可能无法□理含有大量□的 netgroup。 例如，某些早期版本的 SunOS™ 会在 netgroup 中包含多于 15 个 □ □出□□□。 要□□□个□□，可以□建多个 子netgroup，□一个中包含少于 15 个用□， 以及一个包含所有 子netgroup 的真正的 netgroup：

```
BIGGRP1  (,joe1,domain)  (,joe2,domain)  (,joe3,domain) [...]
BIGGRP2  (,joe16,domain)  (,joe17,domain) [...]
BIGGRP3  (,joe31,domain)  (,joe32,domain)
BIGGROUP  BIGGRP1 BIGGRP2 BIGGRP3
```

如果需要超□ 225 个用□， 可以□□重□上面的□程。

激活并分□新的 NIS 映射非常□□：

```
ellington# cd /var/yp
ellington# make
```

□个操作会生成三个 NIS 映射， 即 netgroup、 netgroup.byhost 和 netgroup.byuser。 用 ypcat(1) 可以□□□些 NIS 映射是否可用了：

```
ellington% ypcat -k netgroup
ellington% ypcat -k netgroup.byhost
ellington% ypcat -k netgroup.byuser
```

第一个命令的□出， □□与 /var/yp/netgroup 的内容相近。 第二个命令， 如果没有指定本机□有的 netgroup， □□□没有□出。 第三个命令， □用于□示某个用□□□的 netgroup 列表。

客□机的□置也很□□。 要配置服□器 war， 只需□入 vipw(8) 并把

```
+:::::::::
```

改□

```
+@IT_EMP:::::::::
```

□在， 只有 netgroup IT_EMP 中定□的用□会被□入到 war 的口令数据□中， 因此只有□些用□能□登□。

不□， □个限制也会作用于 shell 的 ~， 以及所有在用□名和数字用□ ID 之□□施□□的函数的功能。 □言之， cd

~user 将不会正常工作， 而 ls -l 也将□示数字的 ID 而不是用□名， 并且 find . -user joe -print 将失□，并□出 No such user 的□□信息。 要修正□个□□， □需要□入所有的用□□， 而 不允□他□登□服□器。

□可以通□在 /etc/master.passwd 加入□一行来完成。 □行的内容是：

+:::::::::/sbin/nologin， 意思是 "□入所有的□， 但□入□的 shell □替□□ /sbin/nologin"。 通□在 /etc/master.passwd 中□加默□□， 可以替□掉 passwd 中的任意字段。

□必□□ +:::::::::/sbin/nologin □一行出□在 +@IT_EMP::::::::: 之后。 否□， 所有从 NIS □入的用□□号将以 /sbin/nologin 作□登□ shell。

完成上面的修改之后， 在 IT 部□有了新□工□， 只需修改一个 NIS 映射就足□了。 □也可以用□似的方法，在不太重要的服□器上， 把先前本地版本的 /etc/master.passwd 中的 +::::::::: 改□：

```
+@IT_EMP:::::::::
+@IT_APP:::::::::
+:::::::::/sbin/nologin
```

相□的用于普通工作站的配置□□是：

```
+@IT_EMP:::::::::
+@USERS:::::::::
+:::::::::/sbin/nologin
```

一切平安无事， 直到数周后， 有一天策略□生了□化： IT 部□也□始招收□□生了。 IT □□生允□使用普通的□端， 以及不太重要的服□器； 而 IT 学徒， □可以登□主服□器。 □□加了新的 netgroup IT_INTERN，以及新的 IT □□生到□个 netgroup 并□始修改□一台机器上的配置..... 老□□得好："□一□， □全身"。

NIS 通□ netgroup 来建立 netgroup 的能力， 正可以避免□□的情形。 一□可能的方法是建立基于角色的 netgroup。 例如， □可以□建称□ BIGSRV 的 netgroup， 用于定□最重要的服□器上的登□限制， 以及□一个成□ SMALLSRV 的 netgroup， 用以定□次重要的服□器， 以及第三个， 用于普通工作站的 netgroup USERBOX。 □三个 netgroup 中的□一个， 都包含了允□登□到□些机器上的所有 netgroup。 □的 NIS 映射中的新□如下所示：

```
BIGSRV    IT_EMP  IT_APP
SMALLSRV  IT_EMP  IT_APP  ITINTERN
USERBOX   IT_EMP  ITINTERN USERS
```

□□定□登□限制的方法， 在□能□将机器分□并加以限制的□候可以工作的相当好。 不幸的是， □是□例外，而非常□情况。 多数□候， 需要按机器去定□登□限制。

与机器相□的 netgroup 定□， 是□理上述策略改□的□一□可能的方法。 此□， □台机器的 /etc/master.passwd 中， 都包含□个 "+" □□的行。 第一个用于添加允□登□的 netgroup □号， 而第二个□用于□加其它□号， 并把 shell □置□ /sbin/nologin。 使用 "全大写" 的机器名作□ netgroup 名是个好主意。 □言之， □些行□□□似于：

```
+@BOXNAME:::::::::
+:::::::::/sbin/nologin
```

一旦在所有机器上都完成了□□的修改，　就再也不需要修改本地的　/etc/master.passwd　了。所有未来的修改都可以在 NIS 映射中□行。□里是一个例子，其中展示了在□一□用情景中所需要的 netgroup 映射，以及其它一些常用的技巧：

```
# Define groups of users first
IT_EMP    (,alpha,test-domain)    (,beta,test-domain)
IT_APP    (,charlie,test-domain)  (,delta,test-domain)
DEPT1     (,echo,test-domain)     (,foxtrott,test-domain)
DEPT2     (,golf,test-domain)     (,hotel,test-domain)
DEPT3     (,india,test-domain)    (,juliet,test-domain)
ITINTERN  (,kilo,test-domain)     (,lima,test-domain)
D_INTERNS (,able,test-domain)     (,baker,test-domain)
#
# Now, define some groups based on roles
USERS     DEPT1   DEPT2     DEPT3
BIGSRV    IT_EMP  IT_APP
SMALLSRV  IT_EMP  IT_APP    ITINTERN
USERBOX   IT_EMP  ITINTERN  USERS
#
# And a groups for a special tasks
# Allow echo and golf to access our anti-virus-machine
SECURITY  IT_EMP  (,echo,test-domain)  (,golf,test-domain)
#
# machine-based netgroups
# Our main servers
WAR       BIGSRV
FAMINE    BIGSRV
# User india needs access to this server
POLLUTION  BIGSRV  (,india,test-domain)
#
# This one is really important and needs more access restrictions
DEATH     IT_EMP
#
# The anti-virus-machine mentioned above
ONE       SECURITY
#
# Restrict a machine to a single user
TWO       (,hotel,test-domain)
# [...more groups to follow]
```

如果□正使用某□数据□来管理□号，□□可以使用□的数据□的□告工具来□建映射的第一部分。□□，新用□就自□地可以□□□些机器了。

最后的提醒：使用基于机器的 netgroup 并不□是□用的。如果正在□学生□□室部署数十台甚至上百台同□的机器，□□□使用基于角色的 netgroup，而不是基于机器的 netgroup，以便把 NIS

映射的尺寸保持在一个合理的□□内。

30.4.8. 需要牢□的事□

□里是一些其它在使用 NIS □境□需要注意的地方。

- □次需要在□□室中□加新用□□， 必□ 只 在 NIS 服□器上加入用□， 而且 一定要□得重建 *NIS* 映射。 如果□忘□了□□做， 新用□将无法登除 NIS 主服□器之外的任何其它机器。 例如， 如果要在□□室□加新用□ `jsmith`， 我□需要：

  ```
  # pw useradd jsmith
  # cd /var/yp
  # make test-domain
  ```

  也可以□行 `adduser jsmith` 而不是 `pw useradd jsmith`.

- 将管理用的□号排除在 *NIS* 映射之外。 一般来□， □不希望□些管理□号和口令被□散到那些包含不□使用它□的用□的机器上。

- □保 *NIS* 主和从服□器的安全， 并尽可能□少其停机□□。 如果有人攻入或□□地□□□些机器， □整个□□室的任也就无法登□了。

  □是集中式管理系□中最薄弱的□□。 如果没有保□好 NIS 服□器， □就有大批□怒的用□需要□付了！

30.4.9. NIS v1 兼容性

FreeBSD 的 ypserv 提供了某些□ NIS v1 客□提供服□的支持能力。 FreeBSD 的 NIS □□， 只使用 NIS v2 □□， 但其它□□可能会包含 v1 □□， 以提供□旧系□的向下兼容能力。 随□些系□提供的 ypbind 服□将首先□□□定 NIS v1 服□器， 即使它□并不真的需要它 (有些甚至可能会一直广播搜索□求， 即使已□从某台 v2 服□器得到了回□也是如此)。 注意， 尽管支持一般的客□机□用， □个版本的 ypserv 并不能□理 v1 的映射□送□求； 因而， 它就不能与□早的支持 v1 □□的 NIS 服□器配合使用， 无□是作□主服□器□是从服□器。 幸□的是， □今□□已□没有仍然在用的□□的服□器了。

30.4.10. 同□作□ NIS 客□机的 NIS 服□器

在多服□器域的□境中， 如果服□器同□作□ NIS 客□， 在□行 ypserv □要特□小心。 一般来□， □制服□器□定自己要比允□它□广播□定□求要好， 因□□□情况下它□可能会相互□定。 某些怪□的故障， 很可能是由于某一台服□器停机， 而其它服□器都依□其服□所□致的。 最□， 所有的客□机都会超□并□定到其它服□器， 但□个延□可能会相当可□， 而且恢□之后仍然存在再次□生此□□□的□患。

□可以□制一台机器□定到特定的服□器， □是通□ `ypbind` 的 `-S` 参数来完成的。 如果不希望□次□□ NIS 服□器□都手工完成□□工作， 可以在 /etc/rc.conf 中加入：

```
nis_client_enable="YES" # run client stuff as well
nis_client_flags="-S NIS domain,server"
```

参□ ypbind(8) 以了解更多情况。

30.4.11. 口令格式

在□□ NIS □， 口令格式的兼容性□□是一□最□常□的□□。 假如□的 NIS 服□器使用 DES 加密口令，□它只能支持使用 DES 的客□机。 例如， 如果□的网□上有 Solaris™ NIS 客□机， □几乎肯定需要使用 DES 加密口令。

要□□□的服□器和客□机使用的口令格式， 需要□看 /etc/login.conf。 如果主机被配置□使用 DES 加密的口令，□ `default` class 将包含□似□□的□：

```
default:\
    :passwd_format=des:\
    :copyright=/etc/COPYRIGHT:\
    [Further entries elided]
```

其他一些可能的 `passwd_format` 包括 `blf` 和 `md5` (分□□□于 Blowfish 和 MD5 加密口令)。

如果修改了 /etc/login.conf， 就必□重建登□性能数据□， □是通□以 `root` 身□□行下面的程序来完成的：

```
# cap_mkdb /etc/login.conf
```

已□在 /etc/master.passwd 中的口令的格式不会被更新， 直到用□在登□性能数据□重建之后 首次修改口令□止。

接下来， □了□保所有的口令都按照□□□的格式加密了， □需要□□ /etc/auth.conf 中 `crypt_default` □出的□先□□的口令格式。 要完成此工作， 将□□□的格式放到列表的第一□。 例如， 当使用 DES 加密的口令□，□□□□□：

```
crypt_default   =   des blf md5
```

在□一台基于 FreeBSD 的 NIS 服□器和客□机上完成上述工作之后， 就可以肯定□的网□上它□都在使用同□的口令格式了。 如果在 NIS 客□机上做身□□□□生□□， □也是第一个可能出□□□的地方。 注意： 如果□希望在混合的网□上部署 NIS 服□器， 可能就需要在所有系□上都使用 DES， 因□□是所有系□都能□支持的最低限度的公共□准。

30.5. 网□自□配置 (DHCP)

30.5.1. 什□是 DHCP？

DHCP， □□主机配置□□， 是一□□系□得以□接到网□上， 并□取所需要的配置参数手段。 FreeBSD 使用来自 OpenBSD 3.7 的 OpenBSD `dhclient`。 □里提供的所有□于 `dhclient` 的信息， 都是以 ISC 或 OpenBSD DHCP 客□端程序□准的。 DHCP 服□器是 ISC □件包的一部分。

30.5.2. □一□都介□□些内容

□一□描述了 ISC 和 DHCP 系□中的客□端， 以及和 ISC DHCP 系□中的服□器端的□件。 客□端程序，`dhclient`， 是随 FreeBSD 作□它的一部分提供的； 而服□器部分， □可以通□ net/isc-dhcp31-server port

得到。 dhclient(8)、 dhcp-options(5)、 以及 dhclient.conf(5) □机手册， 加上下面所介□的参考文献，都是非常有用的□源。

30.5.3. 它如何工作

当 DHCP 客□程序， dhclient 在客□机上□行□， 它会□始广播□求配置信息的消息。 默□情况下， □些□求是在 UDP 端口 68 上。 服□器通□ UDP 67 □出□□， 向客□机提供一个 IP 地址， 以及其他有□的配置参数，例如子网掩□、 路由器， 以及 DNS 服□器。 所有□些信息都会以 DHCP "lease" 的形式□出，并且只在一段特定的□□内有效 (□是由 DHCP 服□器的□□者配置的)。 □□， 那些已□断□网□的客□机使用的□旧的 IP 地址就能被自□地回收了。

DHCP 客□程序可以从服□器端□取大量的信息。 □于能□得的信息的□□列表， □参考 dhcp-options(5)。

30.5.4. FreeBSD 集成

FreeBSD 完全地集成了 OpenBSD 的 DHCP 客□端， dhclient。 DHCP 客□端支持在安装程序和基本系□中均有提供， □使得□不再需要去了解那些已□□行了 DHCP 服□器的网□的具体配置参数。

sysinstall 能□支持 DHCP。 在 sysinstall 中配置网□接口□， 它□□的第二个□□便是： "Do you want to try DHCP configuration of the interface? (□是否希望在此接口上□□ DHCP 配置?)"。 如果做肯定的回答， □将□行 dhclient， 一旦成功， □将自□地填写网□配置信息。

要在系□□□□使用 DHCP， □必□做□件事：

- □的内核中， 必□包含 bpf □□。 如果需要□□做， 需要将 device bpf 添加到内核的□□配置文件中，并重新□□内核。 要了解□于□□内核的□一□信息， □参□ 配置FreeBSD的内核。

  bpf □□已□是 FreeBSD □行版中默□的 GENERIC 内核的一部分了， 因此如果□没有□内核□行定制， □不用□建一□新的内核配置文件， DHCP 就能工作了。

  □于那些安全意□很□的人来□， □□□知道 bpf 也是包□听工具能□正□工作的条件之一(当然， 它□□需要以 root 身□□行才行)。 bpf是 使用 DHCP 所必□的， 但如果□□安全非常敏感， □很可能会有理由不把 bpf 加入到□的内核配置中， 直到□真的需要使用 DHCP □止。

- □□□的 /etc/rc.conf 并加入下面的□置：

```
ifconfig_fxp0="DHCP"
```

  □必将 fxp0 替□□□希望自□配置的网□接口的名字， □可以在 □置网□ □到更□一□的介□。

  如果□希望使用□一位置的 dhclient， 或者需要□ dhclient □□其他参数， □可以添加下面的配置(根据需要□行修改)：

```
dhclient_program="/sbin/dhclient"
dhclient_flags=""
```

DHCP 服□器， dhcpd， 是作□ net/isc-dhcp31-server port 的一部分提供的。 □个 port 包括了 ISC DHCP 服□器及其文□。

30.5.5. 文件

- /etc/dhclient.conf

  dhclient 需要一个配置文件， /etc/dhclient.conf。 一般□来， □个文件中只包括注□， 而默□□基本上都是合理的。 □个配置文件在 dhclient.conf(5) □机手册中□行了□一□的□述。

- /sbin/dhclient

  dhclient 是一个静□□□的， 它被安装到 /sbin 中。 dhclient(8) □机手册□出了□于 dhclient 的□一□□□。

- /sbin/dhclient-script

  dhclient-script 是一个 FreeBSD □用的 DHCP 客□端配置脚本。 在 dhclient-script(8) 中□它□行了描述， 但一般来□， 用□不需要□其□行任何修改， 就能□□一切正常□□了。

- /var/db/dhclient.leases

  DHCP 客□程序会□□一个数据□来保存有效的 lease， 它□被以日志的形式保存到□个文件中。 dhclient.leases(5) □出了更□□致的介□。

30.5.6. □□□物

DHCP □□的完整描述是 RFC 2131。 □于它的其他信息□源的站点 http://www.dhcp.org/ 也提供了□尽的□料。

30.5.7. 安装和配置 DHCP 服□器

30.5.7.1. □一章包含□些内容

□一章提供了□于如何在 FreeBSD 系□上使用 ISC (Internet 系□□会) 的 DHCP □□套件来架□ DHCP 服□器的信息。

DHCP 套件中的服□器部分并没有作□ FreeBSD 的一部分来提供， 因此□需要安装 net/isc-dhcp31-server port 才能提供□个服□。 □参□ 安装□用程序. Packages 和 Ports 以了解□于如何使用 Ports Collection 的□一□□情。

30.5.7.2. 安装 DHCP 服□器

□了在□的 FreeBSD 系□上□行配置以便作□ DHCP 服□器来使用， 需要把 bpf(4) □□□□□内核。 要完成□□工作， 需要将 device bpf 加入到□的内核配置文件中， 并重新□□内核。 要得到□于如何□□内核的□一□信息， □参□ 配置FreeBSD的内核。

bpf □□是 FreeBSD 所附□的 GENERIC 内核中已□□入的□件， 因此□并不需要□了□ DHCP 正常工作而特□地定制内核。

如果□有□□的安全意□， □□注意 bpf 同□也是□听包程序能□正□工作的□□ (尽管□□程序仍然需要以特□用□身□□行)。 bpf是 使用 DHCP 所必需的， 但如果□□安全非常敏感， □可能会不希望将 bpf 放□内核， 直到□真的□□ DHCP 是必需的□止。

接下来要做的是□□示□的 dhcpd.conf， 它由 net/isc-dhcp31-server port 安装。 默□情况下， 它的名字□□是 /usr/local/etc/dhcpd.conf.sample， 在□始修改之前， □需要把它□制□ /usr/local/etc/dhcpd.conf。

30.5.7.3. 配置 DHCP 服□器

dhcpd.conf 包含了一系列□于子网和主机的定□， 下面的例子可以□助□理解它：

```
option domain-name "example.com";①
option domain-name-servers 192.168.4.100;②
option subnet-mask 255.255.255.0;③

default-lease-time 3600;④
max-lease-time 86400;⑤
ddns-update-style none;⑥

subnet 192.168.4.0 netmask 255.255.255.0 {
  range 192.168.4.129 192.168.4.254;⑦
  option routers 192.168.4.1;⑧
}

host mailhost {
  hardware ethernet 02:03:04:05:06:07;⑨
  fixed-address mailhost.example.com;⑩
}
```

① □个□□指定了提供□客□机作□默□搜索域的域名。 □参考 resolv.conf(5) 以了解□于□一概念的□情。

② □个□□用于指定一□客□机使用的 DNS 服□器， 它□之□以逗号分隔。

③ 提供□客□机的子网掩□。

④ 客□机可以□求租□的有效期， 而如果没有， □服□器将指定一个租□有效期， 也就是□个□ (□位是秒)。

⑤ □是服□器允□租出地址的最大□□。 如果客□机□求了更□的租期， □它将得到一个地址， 但其租期□限于 max-lease-time 秒。

⑥ □个□□用于指定 DHCP 服□器在一个地址被接受或□放□是否□□□□□更新 DNS。 在 ISC □□中， □一□□是 必□指定的。

⑦ 指定地址池中可以用来分配□客□机的 IP 地址□□。 在□个□□之□， 以及其□界的 IP 地址将分配□客□机。

⑧ 定□客□机的默□网□。

⑨ 主机的硬件 MAC 地址 (□□ DHCP 服□器就能□在接到□求□知道□求的主机身□)。

⑩ 指定□是得到同一 IP 地址的主机。 □注意在此□使用主机名是□的， 因□ DHCP 服□器会在返回租借地址信息之前自行解析主机名。

在配制好 dhcpd.conf 之后， □在 /etc/rc.conf 中□用 DHCP 服□器， 也就是□加：

```
dhcpd_enable="YES"
dhcpd_ifaces="dc0"
```

此□的 dc0 接口名□改□ DHCP 服□器需要□听 DHCP 客□端□求的接口 (如果有多个， □用空格分□)。

接下来， 可以用下面的命令来□□服□：

```
# /usr/local/etc/rc.d/isc-dhcpd start
```

如果未来□需要修改服□器的配置， □□必牢□□送 SIGHUP 信号□ dhcpd 并 不会 □致配置文件的重新加□， 而□在其他服□程序中□是比□普遍的□定。 □需要□送 SIGTERM 信号来停止□程， 然后使用上面的命令来重新□□它。

30.5.7.4. 文件

- /usr/local/sbin/dhcpd

  dhcpd 是静□□接的， 并安装到 /usr/local/sbin 中。 随 port 安装的 dhcpd(8) □机手册提供了□于 dhcpd 更□□尽的信息。

- /usr/local/etc/dhcpd.conf

  dhcpd 需要配置文件， 即 /usr/local/etc/dhcpd.conf 才能□向客□机提供服□。 □个文件需要包括□提供□客□机的所有信息， 以及□于服□器□行的其他信息。 此配置文件的□□描述可以在随 port 安装的 dhcpd.conf(5) □机手册上□到。

- /var/db/dhcpd.leases

  DHCP 服□器会□□一个它□□的租用地址数据□， 并保存在□个文件中， □个文件是以日志的形式保存的。 随 port 安装的 dhcpd.leases(5) □机手册提供了更□□的描述。

- /usr/local/sbin/dhcrelay

  dhcrelay 在更□□□的□境中， 可以用来支持使用 DHCP 服□器□□□求□□一个独立网□上的 DHCP 服□器。 如果□需要□个功能， 需要安装 net/isc-dhcp31-relay port。 dhcrelay(8) □机手册提供了更□□尽的介□。

30.6. 域名系□ (DNS)

30.6.1. □□

FreeBSD 在默□情况下使用一个版本的 BIND (Berkeley Internet Name Domain)， □是目前最□流行的 DNS □□□□。 DNS 是一□□□， 可以通□它将域名同 IP 地址相互□□。 例如， □□ www.FreeBSD.org 将得到 FreeBSD Project 的 web 服□器的 IP 地址， 而□□ ftp.FreeBSD.org □将得到□□的 FTP 机器的 IP 地址。 □似地， 也可以做相反的事情。 □□ IP 地址可以得到其主机名。 当然， 完成 DNS □□并不需要在系□中□行域名服□器。

目前， 默□情况下FreeBSD 使用的是 BIND9 DNS 服□□件。 我□内建于系□中的版本提供了□□的安全特性、

新的文件目□□□， 以及自□的 chroot(8) 配置。

在 Internet 上的 DNS 是通□一套□□□□的□威根域名系□， □□域名 (TLD)， 以及一系列小□模的，提供少量域名解析服□并□域名信息□行□存的域名服□器□成的。

目前， BIND 由 Internet Systems Consortium https://www.isc.org/ □□。

30.6.2. □□

要理解□□文□， 需要首先了解一些相□的 DNS □□。

□□	定□
正向 DNS	将域名映射到 IP 地址
原点 (Origin)	表示特定域文件所在的域
named, BIND	在 FreeBSD 中 BIND 域名服□器□件包的常□叫法。
解析器 (Resolver)	□算机用以向域名服□器□□域名信息的一个系□□程
反向 DNS	将 IP 地址映射□主机名
根域	Internet 域□次的起点。 所有的域都在根域之下，□似文件系□中， 文件都在根目□之下那□。
域 (Zone)	独立的域， 子域， 或者由同一机□管理的 DNS 的一部分。

域的例子：

- `.` 在本文□中通常指代根域。
- `org.` 是根域之下的一个□□域名 (TLD)。
- `example.org.` 是在 `org.` TLD 之下的一个域。
- `1.168.192.in-addr.arpa` 是一个表示所有 `192.168.1.*` IP 地址空□中 IP 地址的域。

如□所□， 域名中越□□的部分会越□左出□。 例如， `example.org.` 就比 `org.` □□更小， □似地 `org.` 又比根域更小。 域名各个部分的格局与文件系□十分□似： /dev 目□在根目□之下， 等等。

30.6.3. □行域名服□器的理由

域名服□器通常会有□□形式： □威域名服□器， 以及□存域名服□器。

下列情况需要有□威域名服□器：

- 想要向全世界提供 DNS 信息， 并□□求□出□威□答。
- 注册了□似 `example.org` 的域， 而需要将 IP 指定到其下的主机名上。
- 某个 IP 地址□需要反向 DNS □ (IP 到主机名)。
- □□服□器， 或常□的从 (slave) 服□器， 会在主服□器出□□□或无法□□□来□答□□□求。

下列情况需要有□存域名服□器：

- 本地的 DNS 服□器能□□存， 并比直接向外界的域名服□器□求更快地得到□答。

当有人□□ www.FreeBSD.org □，解析器通常会向上□ ISP 的域名服□器□出□求， 并□得回□。 如果有本地的□存 DNS 服□器， □□只有在第一次被□存 DNS 服□器□到外部世界。 其他的□□不会□向局域网外， 因□它□已□有在本地的□存了。

30.6.4. DNS 如何□作

在 FreeBSD 中， BIND 服□程序被称□ named。

文件	描述
named(8)	BIND 服□程序
rndc(8)	域名服□控制程序
/etc/namedb	BIND 存放域名信息的位置。
/etc/namedb/named.conf	域名服□配置文件

随在服□器上配置的域的性□不同， 域的定□文件一般会存放到 /etc/namedb 目□中的 master、 slave， 或 dynamic 子目□中。 □些文件中提供了域名服□器在□□□□□所需要的 DNS 信息。

30.6.5. □□ BIND

由于 BIND 是默□安装的， 因此配置它相□而言很□□。

默□的 named 配置， 是在 chroot(8) □境中提供基本的域名解析服□， 并且只限于□听本地 IPv4 回□地址 (127.0.0.1)。 如果希望□□□一配置， 可以使用下面的命令：

```
# /etc/rc.d/named onestart
```

如果希望 named 服□在□次□□的□候都能□□□， 需要在 /etc/rc.conf 中加入：

```
named_enable="YES"
```

当然， 除了□□文□所介□的配置□□之外， 在 /etc/namedb/named.conf 中□有很多其它的□□。 不□， 如果□需要了解 FreeBSD 中用于□□ named 的那些□□的□， □可以□看 /etc/defaults/rc.conf 中的 named_* 参数， 并参考 rc.conf(5) □机手册。 除此之外， 在 FreeBSD 中使用 rc 也是一个不□的起点。

30.6.6. 配置文件

目前， named 的配置文件存放于 /etc/namedb 目□， 在使用前□根据需要□行修改， 除非□只打算□它完成□□的域名解析服□。 □个目□同□也是□□行□大多数配置的地方。

30.6.6.1. /etc/namedb/named.conf

```
// $FreeBSD$
//
```

```
// Refer to the named.conf(5) and named(8) man pages, and the documentation
// in /usr/shared/doc/bind9 for more details.
//
// If you are going to set up an authoritative server, make sure you
// understand the hairy details of how DNS works.  Even with
// simple mistakes, you can break connectivity for affected parties,
// or cause huge amounts of useless Internet traffic.

options {
    // Relative to the chroot directory, if any
    directory   "/etc/namedb";
    pid-file    "/var/run/named/pid";
    dump-file   "/var/dump/named_dump.db";
    statistics-file "/var/stats/named.stats";

// If named is being used only as a local resolver, this is a safe default.
// For named to be accessible to the network, comment this option, specify
// the proper IP address, or delete this option.
    listen-on   { 127.0.0.1; };

// If you have IPv6 enabled on this system, uncomment this option for
// use as a local resolver.  To give access to the network, specify
// an IPv6 address, or the keyword "any".
//  listen-on-v6    { ::1; };

// These zones are already covered by the empty zones listed below.
// If you remove the related empty zones below, comment these lines out.
    disable-empty-zone "255.255.255.255.IN-ADDR.ARPA";
    disable-empty-zone
"0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.IP6.ARPA";
    disable-empty-zone
"1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.IP6.ARPA";

// If you've got a DNS server around at your upstream provider, enter
// its IP address here, and enable the line below.  This will make you
// benefit from its cache, thus reduce overall DNS traffic in the Internet.
/*
    forwarders {
        127.0.0.1;
    };
*/

// If the 'forwarders' clause is not empty the default is to 'forward first'
// which will fall back to sending a query from your local server if the name
// servers in 'forwarders' do not have the answer.  Alternatively you can
// force your name server to never initiate queries of its own by enabling the
// following line:
//  forward only;

// If you wish to have forwarding configured automatically based on
// the entries in /etc/resolv.conf, uncomment the following line and
```

```
// set named_auto_forward=yes in /etc/rc.conf.  You can also enable
// named_auto_forward_only (the effect of which is described above).
//  include "/etc/namedb/auto_forward.conf";
```

正如注□所言， 如果希望从上□□存中受益， 可以在此□□用 forwarders。 正常情况下， 域名服□器会逐□地□□ Internet 来□到特定的域名服□器， 直到得到答案□止。 □个□□将□它首先□□上□域名服□器 (或□外提供的域名服□器)， 从而从它□的□存中得到□果。 如果上□域名服□器是一个繁忙的高速域名服□器，□□用它将有助于改善服□品□。

127.0.0.1不会 正常工作。 一定要把地址改□□上□服□器的 IP 地址。

```
    /*
       Modern versions of BIND use a random UDP port for each outgoing
       query by default in order to dramatically reduce the possibility
       of cache poisoning.  All users are strongly encouraged to utilize
       this feature, and to configure their firewalls to accommodate it.

       AS A LAST RESORT in order to get around a restrictive firewall
       policy you can try enabling the option below.  Use of this option
       will significantly reduce your ability to withstand cache poisoning
       attacks, and should be avoided if at all possible.

       Replace NNNNN in the example with a number between 49160 and 65530.
    */
    // query-source address * port NNNNN;
};

// If you enable a local name server, don't forget to enter 127.0.0.1
// first in your /etc/resolv.conf so this server will be queried.
// Also, make sure to enable it in /etc/rc.conf.

// The traditional root hints mechanism. Use this, OR the slave zones below.
zone "." { type hint; file "named.root"; };

/*  Slaving the following zones from the root name servers has some
    significant advantages:
    1. Faster local resolution for your users
    2. No spurious traffic will be sent from your network to the roots
    3. Greater resilience to any potential root server failure/DDoS

    On the other hand, this method requires more monitoring than the
    hints file to be sure that an unexpected failure mode has not
    incapacitated your server.  Name servers that are serving a lot
    of clients will benefit more from this approach than individual
    hosts.  Use with caution.

    To use this mechanism, uncomment the entries below, and comment
    the hint zone above.
*/
```

```
/*
zone "." {
    type slave;
    file "slave/root.slave";
    masters {
        192.5.5.241;    // F.ROOT-SERVERS.NET.
    };
    notify no;
};
zone "arpa" {
    type slave;
    file "slave/arpa.slave";
    masters {
        192.5.5.241;    // F.ROOT-SERVERS.NET.
    };
    notify no;
};
zone "in-addr.arpa" {
    type slave;
    file "slave/in-addr.arpa.slave";
    masters {
        192.5.5.241;    // F.ROOT-SERVERS.NET.
    };
    notify no;
};
*/

/*  Serving the following zones locally will prevent any queries
    for these zones leaving your network and going to the root
    name servers.  This has two significant advantages:
    1. Faster local resolution for your users
    2. No spurious traffic will be sent from your network to the roots
*/
// RFC 1912
zone "localhost"    { type master; file "master/localhost-forward.db"; };
zone "127.in-addr.arpa" { type master; file "master/localhost-reverse.db"; };
zone "255.in-addr.arpa" { type master; file "master/empty.db"; };

// RFC 1912-style zone for IPv6 localhost address
zone "0.ip6.arpa"   { type master; file "master/localhost-reverse.db"; };

// "This" Network (RFCs 1912 and 3330)
zone "0.in-addr.arpa"       { type master; file "master/empty.db"; };

// Private Use Networks (RFC 1918)
zone "10.in-addr.arpa"      { type master; file "master/empty.db"; };
zone "16.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "17.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "18.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "19.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "20.172.in-addr.arpa"  { type master; file "master/empty.db"; };
```

```
zone "21.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "22.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "23.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "24.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "25.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "26.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "27.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "28.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "29.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "30.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "31.172.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "168.192.in-addr.arpa" { type master; file "master/empty.db"; };

// Link-local/APIPA (RFCs 3330 and 3927)
zone "254.169.in-addr.arpa" { type master; file "master/empty.db"; };

// TEST-NET for Documentation (RFC 3330)
zone "2.0.192.in-addr.arpa" { type master; file "master/empty.db"; };

// Router Benchmark Testing (RFC 3330)
zone "18.198.in-addr.arpa"  { type master; file "master/empty.db"; };
zone "19.198.in-addr.arpa"  { type master; file "master/empty.db"; };

// IANA Reserved - Old Class E Space
zone "240.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "241.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "242.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "243.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "244.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "245.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "246.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "247.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "248.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "249.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "250.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "251.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "252.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "253.in-addr.arpa"     { type master; file "master/empty.db"; };
zone "254.in-addr.arpa"     { type master; file "master/empty.db"; };

// IPv6 Unassigned Addresses (RFC 4291)
zone "1.ip6.arpa"       { type master; file "master/empty.db"; };
zone "3.ip6.arpa"       { type master; file "master/empty.db"; };
zone "4.ip6.arpa"       { type master; file "master/empty.db"; };
zone "5.ip6.arpa"       { type master; file "master/empty.db"; };
zone "6.ip6.arpa"       { type master; file "master/empty.db"; };
zone "7.ip6.arpa"       { type master; file "master/empty.db"; };
zone "8.ip6.arpa"       { type master; file "master/empty.db"; };
zone "9.ip6.arpa"       { type master; file "master/empty.db"; };
zone "a.ip6.arpa"       { type master; file "master/empty.db"; };
zone "b.ip6.arpa"       { type master; file "master/empty.db"; };
```

```
zone "c.ip6.arpa"       { type master; file "master/empty.db"; };
zone "d.ip6.arpa"       { type master; file "master/empty.db"; };
zone "e.ip6.arpa"       { type master; file "master/empty.db"; };
zone "0.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "1.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "2.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "3.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "4.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "5.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "6.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "7.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "8.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "9.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "a.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "b.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "0.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "1.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "2.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "3.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "4.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "5.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "6.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "7.e.f.ip6.arpa"       { type master; file "master/empty.db"; };

// IPv6 ULA (RFC 4193)
zone "c.f.ip6.arpa"     { type master; file "master/empty.db"; };
zone "d.f.ip6.arpa"     { type master; file "master/empty.db"; };

// IPv6 Link Local (RFC 4291)
zone "8.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "9.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "a.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "b.e.f.ip6.arpa"       { type master; file "master/empty.db"; };

// IPv6 Deprecated Site-Local Addresses (RFC 3879)
zone "c.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "d.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "e.e.f.ip6.arpa"       { type master; file "master/empty.db"; };
zone "f.e.f.ip6.arpa"       { type master; file "master/empty.db"; };

// IP6.INT is Deprecated (RFC 4159)
zone "ip6.int"          { type master; file "master/empty.db"; };

// NB: Do not use the IP addresses below, they are faked, and only
// serve demonstration/documentation purposes!
//
// Example slave zone config entries.  It can be convenient to become
// a slave at least for the zone your own domain is in.  Ask
// your network administrator for the IP address of the responsible
// master name server.
//
```

```
// Do not forget to include the reverse lookup zone!
// This is named after the first bytes of the IP address, in reverse
// order, with ".IN-ADDR.ARPA" appended, or ".IP6.ARPA" for IPv6.
//
// Before starting to set up a master zone, make sure you fully
// understand how DNS and BIND work.  There are sometimes
// non-obvious pitfalls.  Setting up a slave zone is usually simpler.
//
// NB: Don't blindly enable the examples below. :-)  Use actual names
// and addresses instead.

/* An example dynamic zone
key "exampleorgkey" {
    algorithm hmac-md5;
    secret "sf87HJqjkqh8ac87a02lla==";
};
zone "example.org" {
    type master;
    allow-update {
        key "exampleorgkey";
    };
    file "dynamic/example.org";
};
*/

/* Example of a slave reverse zone
zone "1.168.192.in-addr.arpa" {
    type slave;
    file "slave/1.168.192.in-addr.arpa";
    masters {
        192.168.1.1;
    };
};
*/
```

在 named.conf 中， □□出了从域、□□域和反解析域的例子。

如果新□了域， 就必需在 named.conf 中加入□□的□目。

例如， 用于 example.org 的域文件的描述□似下面□□：

```
zone "example.org" {
    type master;
    file "master/example.org";
};
```

如 type □句所□示的那□， □是一个主域， 其信息保存在 /etc/namedb/master/example.org 中， 如 file □句所示。

```
zone "example.org" {
    type slave;
    file "slave/example.org";
};
```

在从域的情形中， 所指定的域的信息会从主域名服□器□□□来， 并保存到□□的文件中。 当主域服□器□生□□或不可□□， 从域名服□器就有一□可用的域名信息， 从而能□□外提供服□。

30.6.6.2. 域文件

下面的例子展示了用于 example.org 的主域文件 (存放于 /etc/namedb/master/example.org)：

```
$TTL 3600        ; 1 hour default TTL
example.org.    IN      SOA      ns1.example.org. admin.example.org. (
                                2006051501      ; Serial
                                10800           ; Refresh
                                3600            ; Retry
                                604800          ; Expire
                                300             ; Negative Reponse TTL
                        )

; DNS Servers
                IN      NS      ns1.example.org.
                IN      NS      ns2.example.org.

; MX Records
                IN      MX 10   mx.example.org.
                IN      MX 20   mail.example.org.

                IN      A       192.168.1.1

; Machine Names
localhost       IN      A       127.0.0.1
ns1             IN      A       192.168.1.2
ns2             IN      A       192.168.1.3
mx              IN      A       192.168.1.4
mail            IN      A       192.168.1.5

; Aliases
www             IN      CNAME   example.org.
```

□注意以 "." □尾的主机名是全称主机名， 而□尾没有 "." 的□是相□于原点的主机名。 例如， ns1 将被□□□ ns1.example.org.

域信息文件的格式如下：

```
□□名          IN □□□型     □
```

最常用的 DNS □□：

SOA

□域□威□始

NS

□威域名服□器

A

主机地址

CNAME

□名□□的正□名称

MX

□件□□服□器

PTR

域名指□ (用于反向 DNS)

```
example.org. IN SOA ns1.example.org. admin.example.org. (
                          2006051501      ; Serial
                          10800           ; Refresh after 3 hours
                          3600            ; Retry after 1 hour
                          604800          ; Expire after 1 week
                          300 )           ; Negative Reponse TTL
```

`example.org.`

域名， 同□也是□个域信息文件的原点。

`ns1.example.org.`

□域的主/□威域名服□器。

`admin.example.org.`

此域的□□人的□子□件地址， 其中 "@" 需要□掉 (admin@example.org □□ `admin.example.org`)

`2006051501`

文件的序号。 □次修改域文件□都必□□加□个数字。 □今， □多管理□会考□使用 `yyyymmddrr` □□的格式来表示序号。 `2006051501` 通常表示上次修改于 05/15/2006， 而后面的 `01` □表示在那天的第一次修改。 序号非常重要， 它用于通知从域服□器更新数据。

```
       IN NS           ns1.example.org.
```

□是一个 NS □。 □个准□提供□威□答的服□器都必□有一个□□□。

```
localhost       IN      A       127.0.0.1
ns1             IN      A       192.168.1.2
ns2             IN      A       192.168.1.3
mx              IN      A       192.168.1.4
mail            IN      A       192.168.1.5
```

A □□指明了机器名。 正如在前面所看到的， ns1.example.org 将解析□ 192.168.1.2。

```
                IN      A       192.168.1.1
```

□一行把当前原点 example.org 指定□使用 IP 地址 192.168.1.1。

```
www             IN CNAME        @
```

正□名 (CNAME) □□通常用于□某台机器指定□名。 在□个例子中， 将 www 指定成了 "主" 机器的一个□名，后者的名字与域名 example.org (192.168.1.1) 相同。 CNAME 不能同与之有相同名字的任何其它□□并存。

```
              IN MX    10      mail.example.org.
```

MX □□表示□个□件服□器□□接收□到□个域的□件。 mail.example.org 是□件服□器的主机名， 而 10 □是它的□先□。

可以有多台□件服□器， 其□先□分□是 10、 20 等等。 □□向 example.org 投□□件的服□器， 会首先□□□先□最高的 MX (□先□数□最小的□□)、 接着□□次高的， 并重□□一□程直到□件□□□止。

in-addr.arpa 域名信息文件 (反向 DNS)， 采用的格式是同□的， 只是 PTR □代替了 A 或 CNAME 的位置。

```
$TTL 3600

1.168.192.in-addr.arpa. IN SOA ns1.example.org. admin.example.org. (
                        2006051501      ; Serial
                        10800           ; Refresh
                        3600            ; Retry
                        604800          ; Expire
                        300 )           ; Negative Reponse TTL

        IN      NS      ns1.example.org.
        IN      NS      ns2.example.org.

1       IN      PTR     example.org.
2       IN      PTR     ns1.example.org.
3       IN      PTR     ns2.example.org.
4       IN      PTR     mx.example.org.
5       IN      PTR     mail.example.org.
```

□个文件□出了上述假想域中 IP 地址到域名的映射□系。

需要□明的是， 在 PTR □□右□的名字必□是全称域名 (也就是必□以 "." □束)。

30.6.7. □存域名服□器

□存域名服□器是一□主要承担解析□□□□角色的域名服□器。 它□□地自行□行□□， 并将□□□果□住以□后□使用。

30.6.8. 安全

尽管 BIND 是最□常用的 DNS □□， 但它□是有一些安全□□。 □常会有人□□一些可能的甚至可以利用的安全漏洞。

尽管 FreeBSD 会自□将 named 放到 chroot(8) □境中□行， 但仍有一些其它可用的安全机制来□助□□避潜在的□□ DNS 服□的攻□。

□□ CERT 的安全公告， 并□□ the FreeBSD 安全□□通知□件列表 是一个有助于□助□了解最新 Internet 及 FreeBSD 安全□□的好□□。

如果□□了□□， □保源代□是最新的， 并重新□□一□ named 有可能会有所□助。

30.6.9. □一□□□

BIND/named □机手册： rndc(8) named(8) named.conf(8)

- 官方的 ISC BIND □面
- Official ISC BIND Forum
- O'Reilly DNS 和 BIND 第 5 版
- RFC1034 - 域名 - 概念和工具
- RFC1035 - 域名 - □□及其□准

30.7. Apache HTTP 服□器

30.7.1. □□

FreeBSD 被用于□行□多全球最□繁忙的 web 站点。 大多数 Internet 上的 web 服□器， 都使用 Apache HTTP 服□器。 Apache □件包可以在□的 FreeBSD 安装□上□到。 如果没有在首次安装□附□安装 Apache， □可以通□ www/apache13 或 www/apache22 port 来安装。

一旦成功地安装了 Apache， 就必□□其□行配置。

□一□介□了 1.3.X 版本的 Apache HTTP 服□器 的配置， 因□它是随 FreeBSD 一同使用的最多的版本。 Apache 2.X 引入了很多新技□， 但在此并不□□。 要了解□于 Apache 2.X 的更多□料， □参□ http://httpd.apache.org/。

30.7.2. 配置

主要的 Apache HTTP Server 配置文件， 在 FreeBSD 上会安装□ /usr/local/etc/apache/httpd.conf。□是一个典型的 UNIX® 文本配置文件， 它使用 # 作□注□符。 □于全部配置□□的□尽介□超出了本□的□□，□里将只介□最常被修改的那些。

`ServerRoot "/usr/local"`

□指定了 Apache 安装的□□目□。 □行文件被放到服□器根目□ (server root) 的 bin 和 sbin 子目□中，而配置文件□位于 etc/apache。

`ServerAdmin you@your.address`

□个地址是在服□器□生□□□□送□子□件的地址， 它会出□在服□器生成的□面上， 例如□□□面。

`ServerName www.example.com`

ServerName 允□□配置□送回客□端的主机名， 如果□的服□器被用□以□的名字□□ (例如， 使用 www 而不是主机本身的真□名字)。

`DocumentRoot "/usr/local/www/data"`

DocumentRoot： □个目□是□的文□所在的目□。 默□情况下， 所有的□求都会从□个位置去□取， 但也可以通□符号□接和□名指定其它的位置。

在修改配置之前□□ Apache 的配置文件永□是一个好□□。 一旦□初始配置□意了， 就可以□始□行 Apache 了。

30.7.3. □行 Apache

与□多其它网□服□不同， Apache 并不依□ inetd 超□服□器来□行。 一般情况下会把它配置□一个独立的服□器， 以期在客□的 web □□器□入 HTTP □求□， 能□□得更好的性能。 它提供了一个 shell 脚本来使□□、停止和重新□□服□器□得尽可能地□□。 首次□□ Apache， 只需□行：

```
# /usr/local/sbin/apachectl start
```

可以在任何□候使用下面的命令来停止服□：

```
# /usr/local/sbin/apachectl stop
```

当由于某□原因修改了配置文件之后， 需要重□服□器：

```
# /usr/local/sbin/apachectl restart
```

要在重□ Apache 服□器□不中断当前的□接， □□□行：

```
# /usr/local/sbin/apachectl graceful
```

更多的信息， 可以在 apachectl(8) □机手册中□到。

要在系□□□□□□ Apache， □□在 /etc/rc.conf 中加入：

```
apache_enable="YES"
```

或者□于Apache 2.2：

```
apache22_enable="YES"
```

如果□希望在系□引□□□□ Apache `httpd` 程序并指定其它一些□□， □可以把下面的行加到 rc.conf：

```
apache_flags=""
```

□在 web 服□器就□始□行了， □可以使用 web □□器打□ `http://localhost/`。 默□□示的 web □面是 /usr/local/www/data/index.html。

30.7.4. 虚□主机

Apache 支持□□不同□型的虚□主机。 第一□方法是基于名字的虚□主机。 基于名字的虚□主机使用客□机□来的 HTTP/1.1 □来辨□主机名。 □使得不同的域得以共享同一个 IP 地址。

要配置 Apache 来使用基于名字的虚□主机， 需要把□似下面的□加到□的 httpd.conf 中：

```
NameVirtualHost *
```

如果□的 web 服□器的名字是 `www.domain.tld`， 而□希望建立一个 `www.someotherdomain.tld` 的虚□域， □□在 httpd.conf 中加入：

```
<VirtualHost *>
ServerName www.domain.tld
DocumentRoot /www/domain.tld
</VirtualHost>

<VirtualHost *>
ServerName www.someotherdomain.tld
DocumentRoot /www/someotherdomain.tld
</VirtualHost>
```

□需要把上面的地址和文□路径改□所使用的那些。

要了解□于虚□主机的更多信息， □参考官方的 Apache 文□， □些文□可以在 http://httpd.apache.org/docs/vhosts/ □到。

30.7.5. Apache 模□

有□多不同的 Apache 模□， 它□可以在基本的服□器基□上提供□多附加的功能。 FreeBSD 的 Ports Collection □安装 Apache 和常用的附加模□提供了非常方便的方法。

30.7.5.1. mod_ssl

mod_ssl □个模□使用 OpenSSL □， 来提供通□ 安全套接字□ (SSL v2/v3) 和 □□□安全 (TLS v1) □□的□加密能力。 □个模□提供了从某一受信的□□□署机□申□□名□□所需的所有工具， □可以藉此在 FreeBSD 上□行安全的 web 服□器。

如果□未曾安装 Apache， 也可以直接安装一□包含了 mod_ssl 的版本的 Apache 1.3.X， 其方法是通□ www/apache13-modssl port 来□行。 SSL 支持已□作□ Apache 2.X 的一部分提供， □可以通□ www/apache22 port 来安装后者。

30.7.5.2. □言□定

Apache□于一些主要的脚本□言都有相□的模□。 □些模□使得完全使用某□脚本□言来写 Apache 模□成□可能。他□通常也被嵌入到服□器作□一个常□内存的解□器， 以避免□□一个外部解□器□于下一□将描述的□□网站所需□□和□源上的□□。

30.7.6. □□网站

在□去的十年里，越来越多的企□□了□加收益和暴光率而□向了互□网。 □也同□□□了□于互□网□内容的需求。有些公司，比如 Microsoft® 推出了基于他□□有□品的解决方案，□源社区也做出了□□的回□。比□□尚的□□包括 Django，Ruby on Rails， mod_perl, and mod_php.

30.7.6.1. Django

Django 是一个以 BSD □可□□布的 framework， 能□□□者快速写出高性能高品□的 web □用程序。 它提供□一个□象□系映射□件，数据□型可以被当 Python 中的□象，和一□□富的□□数据□□□ API， 使□□者避免了写 SQL □句。它同□□提供了可□展的模板系□， □□用程序的□□部分与 HTML 的表□□分□。

Django 依□与 mod_python， Apache, 和一个可□的 SQL 数据□引□。 在□置了一些恰当的□志后，FreeBSD 的 Port 系□将会□助□安装□些必需的依□□。

例 *38.* 安装 *Django*，*Apache2*， *mod_python3*，和 *PostgreSQL*

```
# cd /usr/ports/www/py-django; make all install clean -DWITH_MOD_PYTHON3
-DWITH_POSTGRESQL
```

在安装了 Django 和那些依□的□件之后， □需要□建一个 Django □目的目□，然后配置 Apache，当有□于□网站上□用程序的某些指定的 URL □□用内嵌的 Python 解□器。

例 39. Django/mod_python 有□ Apache 部分的配置

□需要在 Apache 的配置文件 httpd.conf 加入以下□几行， 把□某些 URL 的□求□□□的 web □用程序：

```
<Location "/">
    SetHandler python-program
    PythonPath "['/dir/to/your/django/packages/'] + sys.path"
    PythonHandler django.core.handlers.modpython
    SetEnv DJANGO_SETTINGS_MODULE mysite.settings
    PythonAutoReload On
    PythonDebug On
</Location>
```

30.7.6.2. Ruby on Rails

Ruby on Rails 是□外一个□源的 web framework， 提供了一个全面的□□框架，能□助 web □□者工作更有成效和快速写出□大的□用。 它能非常容易的从 posts 系□安装。

```
# cd /usr/ports/www/rubygem-rails; make all install clean
```

30.7.6.3. mod_perl

Apache/Perl 集成□□， 将 Perl 程序□□□言的□大功能， 与 Apache HTTP 服□器 □密地□合到了一起。 通□ mod_perl 模□， 可以完全使用 Perl 来撰写 Apache 模□。 此外， 服□器中嵌入的持久性解□器，消除了由于□□外部的解□器□ Perl 脚本的□□所造成的性能□失。

mod_perl 通□多□方式提供。 要使用 mod_perl， □□注意 mod_perl 1.0 只能配合 Apache 1.3 而 mod_perl 2.0 只能配合 Apache 2.X 使用。 mod_perl 1.0 可以通□ www/mod_perl 安装， 而以静□方式□□的版本，□可以通□ www/apache13-modperl 来安装。 mod_perl 2.0 □可以通□ www/mod_perl2 安装。

30.7.6.4. mod_php

PHP， 也称□ "PHP: Hypertext Preprocessor"， 是一□特□□合于 Web □□的通用脚本□言。 它能□很容易地嵌入到 HTML 之中， 其□法接近于 C、 Java™， 以及 Perl， 以期□ web □□人□的一迅速撰写□□生成的□面。

要□得用于 Apache web 服□器的 PHP5 支持， 可以从安装 lang/php5 port □始。

在首次安装 lang/php5 port 的□候， 系□会自□□示可用的一系列 OPTIONS (配置□□)。 如果□没有看到菜□，例如由于□去曾□安装□ lang/php5 port 等等， 可以用下面的命令再次□示配置菜□， 在 port 的目□中□行：

```
# make config
```

在配置□□□□框中， □中 APACHE □一□， 就可以□□出用于与 Apache web 服□器配合使用的可□□加□的 mod_php5 模□了。

由于各式各□的原因（例如， 出于已□部署的 web □用的兼容性考□）， □多网站仍在使用 PHP4。 如果□需要 mod_php4 而不是 mod_php5， □使用 lang/php4 port。 lang/php4 port 也支持□多 lang/php5 port 提供的配置和□□□□□。

前面我□已□成功地安装并配置了用于支持□□ PHP □用所需的模□。 □□□并□□□已将下述配置加入到了 /usr/local/etc/apache/httpd.conf 中：

```
LoadModule php5_module        libexec/apache/libphp5.so
```

```
AddModule mod_php5.c
    <IfModule mod_php5.c>
        DirectoryIndex index.php index.html
    </IfModule>
    <IfModule mod_php5.c>
        AddType application/x-httpd-php .php
        AddType application/x-httpd-php-source .phps
    </IfModule>
```

□些工作完成之后， □需要使用 apachectl 命令来完成一次 graceful restart 以便加□ PHP 模□：

```
# apachectl graceful
```

在未来□升□ PHP □， make config □□操作就不再是必需的了； □所□□的 OPTIONS 会由 FreeBSD 的 Ports 框架自□保存。

在 FreeBSD 中的 PHP 支持是高度模□化的， 因此基本安装的功能十分有限。 □加其他功能的支持非常□□，只需通□ lang/php5-extensions port 即可完成。 □个 port 提供了一个菜□□□的界面来□助完成 PHP □展的安装。 □外， 也可以通□□□的 port 来□独安装□展。

例如， 要将□于 MySQL 数据□服□器的支持加入 PHP5， 只需□□地安装 databases/php5-mysql。

安装完□展之后， 必□重新□□ Apache 服□器， 来令其□□新的配置□更：

```
# apachectl graceful
```

30.8. 文件□□□□ (FTP)

30.8.1. □□

文件□□□□ (FTP) □用□提供了一个□□的， 与 FTP 服□器交□文件的方法。 FreeBSD 系□中包含了 FTP 服□□件，ftpd。 □使得在 FreeBSD 上建立和管理 FTP 服□器□得非常□□。

30.8.2. 配置

最重要的配置□□是决定允□□些□号□□ FTP 服□器。 一般的 FreeBSD 系□包含了一系列系□□号分□用于□行不同的服□程序， 但未知的用□不□被允□登□并使用□些□号。 /etc/ftpusers 文件中， 列出了不允□通□ FTP □□的用□。 默□情况下， □包含了前述的系□□号， 但也可以在□里加入其它不□通□ FTP □□的用□。

□可能会希望限制通□ FTP 登□的某些用□， 而不是完全阻止他□使用 FTP。 □可以通□ /etc/ftpchroot 文件来完成。 □一文件列出了希望□ FTP □□□行限制的用□和□的表。 而在 ftpchroot(5) □机手册中， 已□□此□行了□尽的介□， 故而不再□述。

如果□想要在服□器上□用匿名的 FTP □□， □必□建立一个名□ ftp 的 FreeBSD 用□。 □□， 用□就可以使用 ftp 或 anonymous 和任意的口令 (□□上， □□是以那个用□的□件地址作□口令) 来登□和□□□的 FTP 服□器。 FTP 服□器将在匿名用□登□□□用 chroot(2)， 以便将其□□限制在 ftp 用□的主目□中。

有□个文本文件可以用来指定□示在 FTP 客□程序中的□迎文字。 /etc/ftpwelcome 文件中的内容将在用□□接上之后， 在登□提示之前□示。 在成功的登□之后， 将□示 /etc/ftpmotd 文件中的内容。 □注意后者是相□于登□□境的， 因此□于匿名用□而言， 将□示 ~ftp/etc/ftpmotd。

一旦正□地配置了 FTP 服□器， 就必□在 /etc/inetd.conf 中□用它。 □里需要做的全部工作就是将注□符 "#" 从已有的 ftpd 行之前去掉：

```
ftp stream  tcp nowait  root    /usr/libexec/ftpd   ftpd -l
```

如 重新加□ inetd 配置文件 所介□的那□， 修改□个文件之后， 必□□ inetd 重新加□它， 才能使新的□置生效。□参□ □置 以□取更多有□如何在□系□上□用 inetd 的□□信息。

ftpd 也可以作□一个独立的服□□□。 □□的□就需要在 /etc/rc.conf 中□置如下的□量：

```
ftpd_enable="YES"
```

在□置了上述□量之后，独立的服□将在下次系□重□的□候□□， 或者通□以 root 身□手□□行如下的命令□□：

```
# /etc/rc.d/ftpd start
```

□在可以通□□入下面的命令来登□□的 FTP 服□器了：

```
% ftp localhost
```

30.8.3. □□

ftpd 服□程序使用 syslog(3) 来□□消息。 默□情况下， 系□日志将把和 FTP 相□的消息□□到 /var/log/xferlog 文件中。 FTP 日志的位置， 可以通□修改 /etc/syslog.conf 中如下所示的行来修改：

```
ftp.info      /var/log/xferlog
```

一定要小心□待在匿名 FTP 服□器中可能遇到的潜在□□。 一般而言， 允□匿名用□上□文件□三思。 □可能□□自己的 FTP 站点成□了交易未□授□的商□□件的□□， 或□生更糟□的情况。 如果不需要匿名的 FTP 上□，可以在文件上配置□限， 使得□能□在其它匿名用□能□下□□些文件之前□□它□。

30.9. □ Microsoft® Windows® 客□机提供文件和打印服□ (Samba)

30.9.1. □□

Samba 是一个流行的□源□件包， 它提供了□□ Microsoft® Windows® 客□机的文件和打印服□。 □□客□机可以□接并使用 FreeBSD 系□上的文件空□， 就如同使用本地的磁□一□， 或者像使用本地打印机一□使用 FreeBSD 上的打印机。

Samba □件包可以在□的 FreeBSD 安装□上□到。 如果□没有在初次安装 FreeBSD □安装 Samba， □可以通□ net/samba34 port 或 package 来安装。

30.9.2. 配置

默□的 Samba 配置文件会以 /usr/local/shared/examples/samba34/smb.conf.default 的名字安装。□个文件必□□制□ /usr/local/etc/smb.conf 并□行定制， 才能□始使用 Samba。

smb.conf 文件中包含了 Samba 的□行□配置信息， 例如□于打印机的定□， 以及希望共享□ Windows® 客□机的 "共享文件系□"。 Samba □件包包含了一个称□ swat 的 web 管理工具， 后者提供了配置 smb.conf 文件的□□方法。

30.9.2.1. 使用 Samba Web 管理工具 (SWAT)

Samba Web 管理工具 (SWAT) 是一个通□ inetd □行的服□程序。 因此， 需要把 /etc/inetd.conf 中下面几行的注□去掉， 才能□使用 swat 来配置 Samba：

```
swat    stream  tcp     nowait/400      root    /usr/local/sbin/swat    swat
```

如 重新加□ inetd 配置文件 中所介□的那□， 在修改了□个配置文件之后， 必□□ inetd 重新加□配置，才能使其生效。

一旦在 inetd.conf 中□用了 swat， 就可以用□□器□□ connect to http://localhost:901 了。 □将首先使用系□的 root □号登□。

只要成功地登□□了 Samba 配置□面， 就可以□□系□的文□， 或从 **Globals**(全局) □□□□始配置了。 **Globals** 小□□□于 [global] 小□中的□量， 前者位于 /usr/local/etc/smb.conf 中。

30.9.2.2. 全局配置

无□是使用 swat， □是直接□□ /usr/local/etc/smb.conf， 通常首先要配置的 Samba □□都是：

workgroup
: NT 域名或工作□名， 其他□算机将通□□些名字来□到服□器。

`netbios name`

□个□□用于□置 Samba 服□器的 NetBIOS 名字。 默□情况下， □是所在主机的 DNS 名字的第一部分。

`server string`

□个□□用于□置通□ `net view` 命令， 以及某些其他网□工具可以□看到的□于服□器的□明性文字。

30.9.2.3. 安全配置

在 /usr/local/etc/smb.conf 中的□个最重要的配置， 是□定的安全模型， 以及客□机上用□的口令存放后端。 下面的□句控制□些□□：

`security`

最常□的□□形式是 `security = share` 和 `security = user`。 如果□的客□机使用用□名， 并且□些用□名与□的 FreeBSD 机器一致， 一般□□□用□□ (user) 安全。 □是默□的安全策略， 它要求客□机首先登□， 然后才能□□共享的□源。

如果采用共享□ (share) 安全， □客□机不需要用有效的用□名和口令登□服□器， 就能□□接共享的□源。 □是□早版本的 Samba 中的默□□。

`passdb backend`

Samba 提供了若干□不同的□□后端模型。 □可以通□ LDAP、 NIS+、 SQL 数据□， 或□□修改的口令文件， 来完成客□端的身□□□。 默□的□□模式是 `smbpasswd`， □也是本章将介□的全部内容。

假□□使用的是默□的 `smbpasswd` 后端， □必□首先□建一个 /usr/local/etc/samba/smbpasswd 文件， 来允□ Samba □客□□行身□□□。 如果□打算□ UNIX® 用□□号能□从 Windows® 客□机上登□， 可以使用下面的命令：

```
# smbpasswd -a username
```

目前推□使用的后端是 `tdbsam`， □□使用下面的命令来添加用□□号：

```
# pdbedit -a -u username
```

□参考 官方的 Samba HOWTO 以了解□于配置□□的□一□信息。 按照前面□出的基本描述， □□□已□可以□□ Samba 了。

30.9.3. □□ Samba

net/samba34 port 会□加一个新的用于控制 Samba 的□□脚本。 要□用□个脚本， 以便用它来完成□□、 停止或重□ Samba 的任□， 需要在 /etc/rc.conf 文件中加入：

```
samba_enable="YES"
```

此外， 也可以□行更□粒度的控制：

```
nmbd_enable="YES"
```

```
smbd_enable="YES"
```

□也同□配置了在系□引□□□□ Samba。

配置好之后， 就可以在任何□候通□下面的命令来□□ Samba 了：

```
# /usr/local/etc/rc.d/samba start
Starting SAMBA: removing stale tdbs :
Starting nmbd.
Starting smbd.
```

□参□ 在 FreeBSD 中使用 rc 以了解□于使用 rc 脚本的□一□信息。

Samba 事□上包含了三个相互独立的服□程序。 □□□能□看到 nmbd 和 smbd □个服□程序都是通□ samba 脚本□□的。 如果在 smb.conf 中□用了 winbind 名字解析服□， □□□可以看到 winbindd 服□被□□起来。

可以在任何□候通□下面的命令来停止□行 Samba：

```
# /usr/local/etc/rc.d/samba stop
```

Samba 是一个□□的□件包， 它提供了用于与 Microsoft® Windows® 网□□行集成的各式各□的功能。 要了解□于□里所介□的基本安装以外的其它功能， □□□ http://www.samba.org。

30.10. 通□ NTP □行□□同□

30.10.1. □□

随着□□的推移， □算机的□□会□向于漂移。 网□□□□□ (NTP) 是一□□保□的□□保持准□的方法。

□多 Internet 服□依□、 或□大地受益于本地□算机□□的准□性。 例如， web 服□器可能会接收到一个□求， 要求如果文件在某一□刻之后修改□才□送它。 在局域网□境中， 共享文件的□算机之□的□□是否同□至□重要， 因□□□才能使□□□保持一致。 □似 cron(8) □□的程序， 也依□于正□的系□□□， 才能□准□地□行操作。

FreeBSD 附□了 ntpd(8) NTP 服□器， 它可以用于□□其它的 NTP 服□器， 并配置本地□算机的□□， 或者□其它机器提供服□。

30.10.2. □□合□的 NTP 服□器

□了同□□的系□□□， 需要首先□到至少一个 NTP 服□器以供使用。 网□管理□， 或 ISP 都可能会提供用于□□目的的 NTP 服□器-□□看他□的文□以了解是否是□□。 □外， 也有一个在□的 公□的 NTP 服□器列表， □可以从中□一个□近的 NTP 服□器。 □□□□□□的服□器的□□策略， 如果需要的□， 申□一下所需的□可。

□□多个相互不□接的 NTP 服□器是一个好主意， □□在某个服□器不可□， 或者□□不可□□就可以有□的□□。 □

是因□， ntpd(8) 会智能地□□它收到的□□-它会更□向于使用可□的服□器。

30.10.3. 配置□的机器

30.10.3.1. 基本配置

如果只想在系□□□□同□□□， □可以使用 ntpdate(8)。 □于□常重新□□， 并且不需要□常同□的□面系□来□□比□□合， 但□大多数机器都□□□行 ntpd(8)。

在引□□使用 ntpdate(8) 来配合□行 ntpd(8) 也是一个好主意。 ntpd(8) □□地修正□□， 而 ntpdate(8) □直接□置□□， 无□机器的当前□□和正□□□有多大的偏差。

要□用引□□的 ntpdate(8)， 需要把 ntpdate_enable="YES" 加到 /etc/rc.conf 中。 此外， □需要通□ ntpdate_flags 来□置同□的服□器和□□， 它□将□□□ ntpdate(8)。

30.10.3.2. 一般配置

NTP 是通□ /etc/ntp.conf 文件来□行配置的， 其格式在 ntp.conf(5) 中□行了描述。 下面是一个例子：

```
server ntplocal.example.com prefer
server timeserver.example.org
server ntp2a.example.net

driftfile /var/db/ntp.drift
```

□里， server □□指定了使用□一个服□器， □一个服□器都独立一行。 如果某一台服□器上指定了 prefer (偏好) 参数， 如上面的 ntplocal.example.com， □会□先□□□个服□器。 如果偏好的服□器和其他服□器的□□存在□著的差□， □□□它的□□， 否□将使用来自它的□□， 而不理会其他服□器。 一般来□， prefer 参数□□□注在非常精□的 NTP □源， 例如那些包含特殊的□□□控硬件的服□器上。

而 driftfile □□， □指定了用来保存系□□□□率偏差的文件。 ntpd(8) 程序使用它来自□地□□□□的自然漂移， 从而使□□即使在切断了外来□源的情况下， 仍能保持相当的准□度。

□外， driftfile □□也保存上一次□□所使用的 NTP 服□器的信息。 □个文件包含了 NTP 的内部信息， 它不□被任何其他□程修改。

30.10.3.3. 控制□的服□器的□□

默□情况下， NTP 服□器可以被整个 Internet 上的主机□□。 如果在 /etc/ntp.conf 中指定 restrict 参数， □可以控制允□□些机器□□□的服□器。

如果希望拒□所有的机器□□□的 NTP 服□器， 只需在 /etc/ntp.conf 中加入：

```
restrict default ignore
```

□□做会禁止□的服□器□□在本地配置中列出的服□器。 如果□需要令 NTP 服□器与外界的 NTP 服□器同□□□， □□允□指定服□器。 □参□□机手册 ntp.conf(5) 以了解□一□的□□。

如果只希望子网内的机器通□□的服□器同□□□， 而不允□它□配置□服□器， 或作□同□□□的□点来□用， □加入

```
restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap
```

□里， 需要把 192.168.1.0 改□□网□上的 IP 地址， 并把 255.255.255.0 改□□的子网掩□。

/etc/ntp.conf 可能包含多个 restrict □□。 要了解□一□的□□， □参□ ntp.conf(5) 的 Access Control Support(□□控制支持) 小□。

30.10.4. □行 NTP 服□器

要□ NTP 服□器在系□□□□随之□□， 需要把 ntpd_enable="YES" 加入到 /etc/rc.conf 中。 如果希望向 ntpd(8) □□更多参数， 需要□□ /etc/rc.conf 中的 ntpd_flags。

要在不重新□□机器的前提下□□服□器， 需要手工□行 ntpd， 并□上 /etc/rc.conf 中的 ntpd_flags 所指定的参数。 例如：

```
# ntpd -p /var/run/ntpd.pid
```

30.10.5. 在□□性的 Internet □接上使用 ntpd

ntpd(8) 程序的正常工作并不需要永久性的 Internet □接。 然而， 如果□的□□性□接是配置□按需□号的， 那□防止 NTP 通□□繁触□□号， 或保持□接就有必要了。 如果□使用用□□ PPP， 可以使用 filter □句， 在 /etc/ppp/ppp.conf 中□行必要的□置。 例如：

```
set filter dial 0 deny udp src eq 123
# Prevent NTP traffic from initiating dial out
set filter dial 1 permit 0 0
set filter alive 0 deny udp src eq 123
# Prevent incoming NTP traffic from keeping the connection open
set filter alive 1 deny udp dst eq 123
# Prevent outgoing NTP traffic from keeping the connection open
set filter alive 2 permit 0/0 0/0
```

要了解□一□的信息， □参考 ppp(8) 的 PACKET FILTERING(包□□) 小□， 以及 /usr/shared/examples/ppp/ 中的例子。

某些 Internet □□提供商会阻止低□号的端口， □会□致 NTP 无法正常工作， 因□□□无法到□□的机器。

30.10.6. □一□的信息

□于 NTP 服□器的文□， 可以在 /usr/shared/doc/ntp/ □到 HTML 格式的版本。

30.11. 使用 syslogd □□□程主机的日志

□理系□日志□于系□安全和管理是一个重要方面。 当有多台分布在中型或大型网□的机器，再或者是□于各□不同□型的网□中， □□他□上面的日志文件□□得非常□以操作， 在□□情况下， 配置□程日志□□能使整个□理□程□得更加□松。

集中□□日志到一台指定的机器能□□□一些日志文件管理的□担。 日志文件的收集， 合并与循□可以在一□配置， 使用 FreeBSD 原生的工具， 比如 syslogd(8) 和 newsyslog(8)。 在以下的配置示例中， 主机 A，命名□ logserv.example.com， 将用来收集本地网□的日志信息。 主机 B， 命名□ logclient.example.com 将把日志信息□送□服□器。 在□□中， □□个主机都需要配置正□的正向和反向的 DNS 或者在 /etc/hosts 中□□。否□， 数据将被服□器拒收。

30.11.1. 日志服□器的配置

日志服□器是配置成用来接收□程主机日志信息的机器。 在大多数的情况下□是□了方便配置， 或者是□了更好的管理。 不□是何原因， 在□□深入之前需要提一些必需条件。

一个正□配置的日志服□器必□符合以下几个最基本的条件：

- 服□器和客□端的防火□□□允□ 514 端口上的 UDP □文通□。
- syslogd 被配置成接受从□程客□□来的消息。
- syslogd 服□器和所有的客□端都必□有配有正□的正向和反向 DNS， 或者在 /etc/hosts 中有相□配置。

配置日志服□器， 客□端必□在 /etc/syslog.conf 中列出, 并指定日志的 facility：

```
+logclient.example.com
*.*     /var/log/logclient.log
```

更多□于各□被支持并可用的 *facility* 能在 syslog.conf(5) 手册□中□到。

一旦加入以后， 所有此□ facility 消息都会被□□到先前指定的文件 /var/log/logclient.log。

提供服□的机器□需要在其 /etc/rc.conf 中配置：

```
syslogd_enable="YES"
syslogd_flags="-a logclient.example.com -v -v"
```

第一个□□表示在系□□□□□用 syslogd 服□， 第二个□□表示允□服□器接收来自指定日志源客□端的数据。第二行配置中最后的部分， 使用 -v -v， 表示□加日志消息的□□程度。 在□整 facility 配置的□候，□个配置非常有用， 因□管理□能□看到□些消息将作□□个 facility 的内容来□□。

可以同□指定多个 -a □□来允□多个客□机。 此外， □可以指定 IP 地址或网段， □参□ syslog(3) □机手册以了解可用配置的完整列表。

最后， 日志文件□□被□建。 不□□用何□方法□建， 比如 touch(1) 能很好的完成此□任□：

```
# touch /var/log/logclient.log
```

此□， □□重□并□□一下 syslogd 守□□程：

```
# /etc/rc.d/syslogd restart
# pgrep syslog
```

如果返回了一个 PIC 的□， 服□端□□被成功重□了， 并□□□始配置客□端。 如果服□端没有重□的□， □在 /var/log/messages 日志中□□相□□出。

30.11.2. 日志客□端配置

日志客□端是一台□送日志信息到日志服□器的机器， 并在本地保存拷□。

与日志服□器□似， 客□端也需要□足一些最基本的条件：

- syslogd(8) 必□被配置成□送指定□型的消息到能接收他□的日志服□器。
- 防火□必□允□ 514 端口上的 UDP 包通□；
- 必□配置正向与反向 DNS， 或者在 /etc/hosts 中有正□的□□。

相比服□器来□配置客□端更□松一些。 客□端的机器在 /etc/rc.conf 中做如下的□置：

```
syslogd_enable="YES"
syslogd_flags="-s -v -v"
```

和前面□似， □些□□会在系□□□□程中□用 syslogd 服□， 并□加日志消息的□□程度。 而 -s □□□表示禁止服□接收来自其他主机的日志。

Facility 是描述某个消息由系□的□部分生成的。 □例来□， ftp 和 ipfw 都是 facility。 当□□□服□生成日志消息□， 它□通常在日志消息中包含了□□□工具。 Facility 通常□有一个□先□或等□， 就是用来□□一个日志消息的重要程度。 最普通的□ warning 和 info。 □参□ syslog(3) 手册□以□得一个完整可用的 facility 与□先□列表。

日志服□器必□在客□端的 /etc/syslog.conf 中指明。 在此例中， @ 符号被用来表示□送日志数据到□程的服□器， 看上去差不多如下□□：

```
*.*     @logserv.example.com
```

添加后， 必□重□ syslogd 使得上述修改生效：

```
# /etc/rc.d/syslogd restart
```

□□日志消息是否能通□网□□送， 在准□□出消息的客□机上用 logger(1) 来向 syslogd □出信息：

```
# logger "Test message from logclient"
```

□段消息□在□□同□出□在客□机的 /var/log/messages 以及日志服□器的 /var/log/logclient.log 中。

30.11.3. □□日志服□器

在某些情况下， 如果日志服□器没有收到消息的□就需要□□一番了。 有几个可能的原因， 最常□的□个是网□□接的□□和 DNS 的□□。 □了□□□些□□， □□□□□的机器都能使用 /etc/rc.conf 中所□定的主机名□□到□方。 如果□个能正常工作的□， 那□就需要□ /etc/rc.conf 中的 `syslogd_flags` □□做些修改了。

在以下的示例中， /var/log/logclient.log 是空的， /var/log/message 中也没有表明任何失□的原因。 □了□加□□的□出， 修改 `ayalogd_flags` □□至□似于如下的示例， 并重□服□：

```
syslogd_flags="-d -a logclien.example.com -v -v"
```

```
# /etc/rc.d/syslogd restart
```

在重□服□之后， 屏幕上将立刻□□□似□□的□□数据：

```
logmsg: pri 56, flags 4, from logserv.example.com, msg syslogd: restart
syslogd: restarted
logmsg: pri 6, flags 4, from logserv.example.com, msg syslogd: kernel boot file is
/boot/kernel/kernel
Logging to FILE /var/log/messages
syslogd: kernel boot file is /boot/kernel/kernel
cvthname(192.168.1.10)
validate: dgram from IP 192.168.1.10, port 514, name logclient.example.com;
rejected in rule 0 due to name mismatch.
```

很明□，消息是由于主机名不匹配而被拒收的。 在一点一点的□□了配置文件之后， □□了 /etc/rc.conf 中如下□行有□入□□：

```
syslogd_flags="-d -a logclien.example.com -v -v"
```

□行□□包涵有 `logclient`， 而不是 `logclien`。 在做了正□的修改并重□之后便能□到□期的效果了：

```
# /etc/rc.d/syslogd restart
logmsg: pri 56, flags 4, from logserv.example.com, msg syslogd: restart
syslogd: restarted
logmsg: pri 6, flags 4, from logserv.example.com, msg syslogd: kernel boot file is
/boot/kernel/kernel
syslogd: kernel boot file is /boot/kernel/kernel
logmsg: pri 166, flags 17, from logserv.example.com,
msg Dec 10 20:55:02 <syslog.err> logserv.example.com syslogd: exiting on signal 2
cvthname(192.168.1.10)
validate: dgram from IP 192.168.1.10, port 514, name logclient.example.com;
accepted in rule 0.
logmsg: pri 15, flags 0, from logclient.example.com, msg Dec 11 02:01:28 trhodes: Test
message 2
Logging to FILE /var/log/logclient.log
Logging to FILE /var/log/messages
```

此刻， 消息能□被正□接收并保存入文件了。

30.11.4. 安全性方面的思考

就像其他的网□服□一□， 在□□配置之前需要考□安全性。 有□日志文件也包含了敏感信息，比如本地主机上所□用的服□， 用□□号和配置数据。 从客□端□出的数据□□网□到□服□器， □期□既没有加密也没有密□保□。 如果有加密需要的□， 可以使用 security/stunnel，它将在一个加密的隧道中□□数据。

本地安全也同□是个□□。 日志文件在使用中或循□□后都没有被加密。 本地用□可能□取□些文件以□得□系□更深入的了解。 □于□□情况， □□些文件□置正□的□限是非常有必要的。 newsyslog(8) 工具支持□新□建和循□的日志□置□限。 把日志文件的□限□置□ 600 能阻止本地用□不必要的□探。

Chapter 31. 防火□

31.1. 入□

防火□的存在， 使得□□出入系□的数据流成□可能。 防火□可以使用一□或多□ "□□ (rules)"， 来□□出入□的网□□接的数据包， 并决定允□或阻止它□通□。 □些□□通常可以□□数据包的某个或某些特征， □些特征包括，但不必限于□□□型、 来源或目的主机地址， 以及来源或目的端口。

防火□可以大幅度地改善主机或网□的安全。 它可以用来完成下面的任□：

- 保□和隔□□用程序、 服□程序， 以及□内部网□上的机器， 不受那些来自公共的 Internet 网□上□所不希望的数据流量的干□。
- 限制或禁止从内部网□□公共的 Internet 上的服□。
- 支持网□地址□□ (NAT)， 它使得□的内部网□能□使用私有的 IP 地址， 并分享一条通往公共的 Internet 的□接 (使用一个 IP 地址， 或者一□公网地址)。

□完□章， □将了解：

- 如何正□地定□包□□□。
- FreeBSD 中内建的几□防火□之□的差□。
- 如何使用和配置 OpenBSD 的 PF 防火□。
- 如何使用和配置 IPFILTER。
- 如何使用和配置 IPFW。

□□□章之前， □需要：

- 理解基本的 FreeBSD 和 Internet 概念。

31.2. 防火□的概念

建立防火□□□集的基本方法有□□： "明示允□ (inclusive)"型 或 "明示禁止 (exclusive)"型。明示禁止的防火□□□， 默□允□所有数据通□防火□， 而□□□□集中定□的， □是不允□通□防火□的流量， □言之，与□些□□不匹配的数据， 全部是允□通□防火□的。 明示允□的防火□正好相反， 它只允□符合□□集中定□□□的流量通□， 而其他所有的流量都被阻止。

明示允□型防火□能□提供□于□出流量更好的控制， □使其更□合那些直接□ Internet 公网提供服□的系□的需要。 它也能□控制来自 Internet 公网到□的私有网□的□□□型。 所有和□□不匹配的流量都会被阻止并□□在案。 一般来□明示允□防火□要比明示禁止防火□更安全， 因□它□□著地□少了允□不希望的流量通□可能造成的□□。

除非特□□明， □一章的配置和示□的□□集都是□建明示允□防火□的。

使用了 "□状□功能的防火□ (stateful firewall)"， 可以□一□地收□安全机制。 □□防火□能□□□通□防火□的□接，□而只允□与□有□接匹配的□接， 或□建新的□接。 □状□功能的防火□的缺点是， 在很短□□内有大量的□接□求□，它□可能会受到拒□服□ (DoS) 攻□。 □大多数防火□都提供了同□□用□□防火□的能力， 以便□站点提供更好的保□。

31.3. 防火□□件包

FreeBSD 的基本系□内建了三□不同的防火□□件包。 它□是 *IPFILTER* (也被称作 IPF)、 *IPFIREWALL* (也被称作 IPFW)， 以及 *OpenBSD* 的 *PacketFilter* (也被称□ PF)。 FreeBSD 也提供了□个内建的、用于流量整形 (基本上是控制□□占用) 的□件包： altq(4) 和 dummynet(4)。 Dummynet 在□去一直和 IPFW □密集成， 而 ALTQ □需要配合 PF 使用。 IPFILTER 的流量整形功能可以使用 IPFILTER 的 NAT 和□□功能以及 IPFW 的 dummynet(4) 配合，或者 使用 PF 跟 ALTQ 的□合。 IPFW， 以及 PF 都是用□□来控制是否允□数据包出入□的系□， □然它□采取了不同的□□方法和□□□法。

FreeBSD 包含多个内建的防火□□件包的原因在于， 不同的人会有不同的需求和偏好。 任何一个防火□□件包都很□□是最好的。

作者□向于使用 IPFILTER， 因□它提供的状□式□□， 在 NAT 的□境中要□□□多， 而且它内建了 ftp 代理，□□化了使用外部 FTP 服□□所需的配置。

由于所有的防火□都基于□□所□定的包控制字段来□□功能， 撰写防火□□□集□， 就必□了解 TCP/IP 是如何工作的， 以及包的控制字段在正常会□交互中的作用。 □可以在□个网站□到一□很好的解□文□：http://www.ipprimer.com/overview.cfm.

31.4. OpenBSD Packet Filter (PF) 和 ALTQ

2003 年 7 月， OpenBSD 的防火□， 也就是常□的 PF 被成功地移植到了 FreeBSD 上， 并可以通□ FreeBSD Ports Collection 来安装了； 第一个将 PF 集成到基本系□中的版本是 2004 年 11 月□行的 FreeBSD 5.3。 PF 是一个完整的提供了大量功能的防火□□件， 并提供了可□的 ALTQ (交□□列， Alternate Queuing) 功能。 ALTQ 提供了服□品□ (QoS) □□整形功能。

OpenBSD □目非常□出的□□着一□ PF FAQ。 就其本身而言，□一□注重于 FreeBSD 的 PF 和提供一些□于使用方面的一般常□。更□□的使用信息□参□ PF FAQ。

更多的□□信息， 可以在 FreeBSD 版本的 PF 网站上□到： http://pf4freebsd.love2party.net/。

31.4.1. 使用 PF 可加□的内核模□

要加□ PF 内核模□， 可以在 /etc/rc.conf 中加入下面的□置：

```
pf_enable="YES"
```

然后使用□□脚本来加□模□：

```
# /etc/rc.d/pf start
```

需要□明的是， 如果系□中没有□□集配置文件， □上述操作不会加□ PF 模□。 配置文件的默□位置是 /etc/pf.conf。 如果 PF □□集在其他位置， 可以用下面的 /etc/rc.conf 配置来告□ PF：

```
pf_rules="/path/to/pf.conf"
```

pf.conf 的例子可以在 /usr/shared/examples/pf/ 到。

PF 模也可以手工从命令行加：

```
# kldload pf.ko
```

PF 的日志功能是由 pflog.ko 提供的， 通在 /etc/rc.conf 中加入下面的置：

```
pflog_enable="YES"
```

然后使用脚本来加模：

```
# /etc/rc.d/pflog start
```

如果需要其他 PF 特性， 需要将 PF 支持内核。

31.4.2. PF 内核

然不必自把 PF 的支持 FreeBSD 内核，但是有仍然需要做来使用到 PF 的某些没有被收可加模的高特性，比如 pfsync(4) 用来送某些改到PF 状表。 它能配合 carp(4) 使用 PF 建立支持故障移的防火。 更多有 CARP 的信息可以参本手册的 Common Address Redundancy Protocol (CARP， 共用地址冗余)。

The PF kernel options can be found in /usr/src/sys/conf/NOTES and are reproduced below:

有 PF 的内核可以在 /usr/src/sys/conf/NOTES 中到， 以下也略有述：

```
device pf
device pflog
device pfsync
```

device pf 用于用 "Packet Filter" 防火的支持 （pf(4)）。

device pflog 用可的 pflog(4) 网， 用以通 bpf(4) 描述符来流量。 pflogd(8) 服可以用来存信息， 并把它以日志形式到磁上。

device pfsync 用可的 pfsync(4) 支持，是用于 "状更" 的网。

31.4.3. 可用的 rc.conf

The following rc.conf(5) statements configure PF and pflog(4) at boot:

以下 rc.conf(5) 中的句用于配置 PF 和 pflog(4)

```
pf_enable="YES"                 # □用 PF (如果需要的□,  自□加□内核模□)
pf_rules="/etc/pf.conf"         # pf 使用的□□定□文件
pf_flags=""                     # □□□□□□ pfctl 的其他□□
pflog_enable="YES"              # □□ pflogd(8)
pflog_logfile="/var/log/pflog"  # pflogd 用于□□日志的文件名
pflog_flags=""                  # □□□□□□ pflogd 的其他□□
```

如果□的防火□后面有一个 LAN， 而且需要通□它来□□ LAN 上的包， 或□行 NAT， □需要同□□用下述□□：

```
gateway_enable="YES"            # □用□ LAN 网□
```

31.4.4. 建立□□□□

PF 会从 pf.conf(5) (默□□ /etc/pf.conf) 文件中□取配置□□， 并根据那里的□□修改、□□或□数据包通□。 默□安装的 FreeBSD 已□提供了一些□□的例子放在 /usr/shared/examples/pf/ 目□下。 □参□ PF FAQ □取完整的 PF □□信息。

在□□ PF FAQ □， □□刻注意不同版本的 FreeBSD 可能会使用不同版本的 PF。 目前， FreeBSD 8.*X* 和之前的系□使用的是与 OpenBSD 4.1 相同版本的 PF。 FreeBSD 9.*X* 和之后的系□使用的是与 OpenBSD 4.5 相同版本的 PF。

FreeBSD packet filter □件列表 是一个提有□配置使用 PF 防火□□□的好地方。□在提□之前□□□件列表的□□！

31.4.5. 使用 PF

使用 pfctl(8) 可以控制 PF。 以下是一些□用的命令 （□□□ pfctl(8) □得全部可用的□□）：

命令	作用
`pfctl -e`	□用 PF
`pfctl -d`	禁用 PF
`pfctl -F all -f /etc/pf.conf`	清除所有□□ (nat, filter, state, table, 等等。) 并□取 /etc/pf.conf
`pfctl -s [ rules | nat | state ]`	列出 filter □□, nat □□, 或状□表
`pfctl -vnf /etc/pf.conf`	□□ /etc/pf.conf 中的□□，但不加□相□的□□

31.4.6. □用 ALTQ

ALTQ 只有在作□□□□□加入到 FreeBSD 内核□才能使用。ALTQ 目前□不是所有的可用网□□□都能□支持的。□参□ altq(4) □机手册了解□正使用的 FreeBSD 版本中的□□支持情况。

下面□些□□将□用 ALTQ 以及一些附加的功能：

```
options         ALTQ
options         ALTQ_CBQ        # 基于分□的排列 (CBQ)
options         ALTQ_RED        # 随机先期□□ (RED)
options         ALTQ_RIO        # □□入和□出的包□行 RED
options         ALTQ_HFSC       # □等□的包□度器 (HFSC)
options         ALTQ_PRIQ       # 按□先□的排列 (PRIQ)
options         ALTQ_NOPCC      # 在□□ SMP 内核□必□使用，禁止□□□
```

options ALTQ 将□用 ALTQ 框架的支持。

options ALTQ_CBQ 用于□用 基于分□的□列 (CBQ) 支持。 CBQ 允□□将□接分成不同的□□， 或者□， □列，以便在□□中□它□指定不同的□先□。

options ALTQ_RED 将□用 随机□□□ (RED)。 RED 是一□用于防止网□□塞的技□。 RED 度量□列的□度，并将其与□列的最大和最小□度□□□行比□。 如果□列□□， □新的包将被□□。 如名所示， RED 从不同的□接中随机地□□数据包。

options ALTQ_RIO 将□用 出入的随机□□□。

options ALTQ_HFSC □用 □次式公平服□平滑包□度器。 要了解□于 HFSC □一□的信息， □参□ http://www-2.cs.cmu.edu/~hzhang/HFSC/main.html。

options ALTQ_PRIQ □用 □先□列 (PRIQ)。 PRIQ 首先允□高□先□□列中的包通□。

options ALTQ_NOPCC □用 ALTQ 的 SMP 支持。 如果是 SMP 系□， □必□使用它。

31.5. IPFILTER (IPF) 防火□

IPFILTER 的作者是 Darren Reed。 IPFILTER 是独立于操作系□的： 它是一个□放源代□的□用， 并且已□被移植到了 FreeBSD、 NetBSD、 OpenBSD、 SunOS、 HP/UX， 以及 Solaris 操作系□上。 IPFILTER 的支持和□□都相当活□， 并且有□律地□布更新版本。

IPFILTER 提供了内核模式的防火□和 NAT 机制， □些机制可以通□用□模式□行的接口程序□行□□和控制。 防火□□□可以使用 ipf(8) 工具来□□地□置和□除。 NAT □□可以通□ ipnat(1) 工具来□□。 ipfstat(8) 工具□可以用来□示 IPFILTER 内核部分的□□数据。 最后， 使用 ipmon(8) 程序可以把 IPFILTER 的□作□□到系□日志文件中。

IPF 最初是使用一□ "以最后匹配的□□□准" 的策略来□□的， □□方式只能支持无状□的□□。 随着□代的□□， IPF 被逐□□□， 并加入了 "quick" □□， 以及支持状□的 "keep state" □□， □使得□□□理□□□得更富有□代气息。 IPF 的官方文□只介□了□□的□□□写方法和文件□理□□。 新□的功能只是作□一些附加的□□出□， 如果能完全理解□些功能， □□于建立更安全的防火□就很有好□。

□一□中主要是□□ "quick" □□， 以及支持状□的 "keep state" □□的介□。 □是明示允□防火□□□集最基本的□写要素。

要□得□于□□□□理方式的□□信息， □参考： http://www.obfuscation.org/ipf/ipf-howto.html#TOC_1 以及 http://coombs.anu.edu.au/~avalon/ip-filter.html。

IPF FAQ 可以在 http://www.phildev.net/ipf/index.html □到。

除此之外， □□可以在 http://marc.theaimsgroup.com/?l=ipfilter □到□放源代□的 IPFilter 的□件列表存□，并□行搜索。

31.5.1. □用 IPF

IPF 作□ FreeBSD 基本安装的一部分， 以一个独立的内核模□的形式提供。 如果在 rc.conf 中配置了 `ipfilter_enable="YES"`， 系□就会自□地□□加□ IPF 内核模□。 □个内核模□在□建□□用了日志支持， 并加入了 `default pass all` □□。 如果只是需要把默□的□□□置□ `block all` 的□， 就不需要把 IPF □□到内核中。 □□地通□把 `block all` □条□□加入自己的□□集来□到同□的目的。

31.5.2. 内核□□

下面□些 FreeBSD 内核□□□□并不是□用 IPF 所必需的。 □里只是作□背景知□来加以□述。 如果将 IPF □入了内核， □□□的内核模□将不被使用。

□于 IPF □□□句的内核□□配置的例子， 可以在内核源代□中的 /usr/src/sys/conf/NOTES □到。 此□列□如下：

```
options IPFILTER
options IPFILTER_LOG
options IPFILTER_DEFAULT_BLOCK
```

`options IPFILTER` 用于□用 "IPFILTER" 防火□的支持。

`options IPFILTER_LOG` 用于□用 IPF 的日志支持， 所有匹配了包含 `log` 的□□的包， 都会被□□到 ipl □个包□□□-□□中。

`options IPFILTER_DEFAULT_BLOCK` 将改□防火□的默□□作， □而， 所有不匹配防火□的 `pass` □□的包都会被阻止。

□些□□只有在□重新□□并安装了上述配置的内核之后才会生效。

31.5.3. 可用的 rc.conf □□

要在□□□激活 IPF， 需要在 /etc/rc.conf 中□加下面的□置：

```
ipfilter_enable="YES"             # □□ ipf 防火□
ipfilter_rules="/etc/ipf.rules"   # 将被加□的□□定□， □是一个文本文件
ipmon_enable="YES"                # □□ IP □□日志
ipmon_flags="-Ds"                 # D = 作□服□程序□□
                                  # s = 使用 syslog □□
                                  # v = □□ tcp □口大小、 ack 和□序号(seq)
                                  # n = 将 IP 和端口映射□名字
```

如果在防火□后面有使用了保留的私有 IP 地址□□的 LAN， □需要□加下面的一些□□来□用 NAT 功能：

```
gateway_enable="YES"               # □用作□ LAN 网□的功能
ipnat_enable="YES"                 # □□ ipnat 功能
ipnat_rules="/etc/ipnat.rules"     # 用于 ipnat 的□□定□文件
```

31.5.4. IPF

ipf(8) 命令可以用来加□□自己的□□文件。 一般情况下， □可以建立一个包括□自定□的□□的文件， 并使用□个命令来替□掉正在□行的防火□中的内部□□：

```
# ipf -Fa -f /etc/ipf.rules
```

-Fa 表示清除所有的内部□□表。

-f 用于指定将要被□取的□□定□文件。

□个功能使得□能□修改自定□的□□文件， 通□□行上面的 IPF 命令， 可以将正在□行的防火□刷新□使用全新的□□集， 而不需要重新□□系□。 □□于□□新的□□来□就很方便， 因□□可以任意□行上面的命令。

□参考 ipf(8) □机手册以了解□个命令提供的其它□□。

ipf(8) 命令假定□□文件是一个□准的文本文件。 它不能□理使用符号代□的脚本。

也□□有□法利用脚本的非常□大的符号替□能力来□建 IPF □□。 要了解□一□的□□， □参考 □建采用符号替□的□□脚本。

31.5.5. IPFSTAT

默□情况下， ipfstat(8) 会□取并□示所有的累□□□， □些□□是防火□□□以来用□定□的□□匹配的出入流量，□可以通□使用 ipf -Z 命令来将□些□数器清零。

□参□ ipfstat(8) □机手册以了解□一□的□□。

默□的 ipfstat(8) 命令□出□似于下面的□子：

```
input packets: blocked 99286 passed 1255609 nomatch 14686 counted 0
 output packets: blocked 4200 passed 1284345 nomatch 14687 counted 0
 input packets logged: blocked 99286 passed 0
 output packets logged: blocked 0 passed 0
 packets logged: input 0 output 0
 log failures: input 3898 output 0
 fragment state(in): kept 0 lost 0
 fragment state(out): kept 0 lost 0
 packet state(in): kept 169364 lost 0
 packet state(out): kept 431395 lost 0
 ICMP replies: 0 TCP RSTs sent: 0
 Result cache hits(in): 1215208 (out): 1098963
 IN Pullups succeeded: 2 failed: 0
 OUT Pullups succeeded: 0 failed: 0
 Fastroute successes: 0 failures: 0
 TCP cksum fails(in): 0 (out): 0
 Packet log flags set: (0)
```

如果使用了 -i (□入流量) 或者 -o (□出流量)， □个命令就只□取并□示内核中所安装的□□□□器□□的□□数据。

ipfstat -in 以□□号的形式□示□入的内部□□表。

ipfstat -on 以□□号的形式□示流出的内部□□表。

□出和下面的□似：

```
@1 pass out on xl0 from any to any
@2 block out on dc0 from any to any
@3 pass out quick on dc0 proto tcp/udp from any to any keep state
```

ipfstat -ih □示内部□□表中的□入流量， □一个匹配□□前面会同□□示匹配的次数。

ipfstat -oh □示内部□□表中的流出流量， □一个匹配□□前面会同□□示匹配的次数。

□出和下面的□似：

```
2451423 pass out on xl0 from any to any
354727 block out on dc0 from any to any
430918 pass out quick on dc0 proto tcp/udp from any to any keep state
```

ipfstat 命令的一个重要的功能可以通□指定 -t 参数来使用， 它会以□似 top(1) 的□示 FreeBSD 正□行的□程表的方式来□示□□数据。 当□的防火□正在受到攻□的□候， □个功能□□得以□□、 □□， 并□看攻□的数据包。□个□□提□提供了□□□□希望□□的目的或源 IP、 端口或□□的能力。 □参□ ipfstat(8) □机手册以了解□□信息。

31.5.6. IPMON

□了使 ipmon 能□正□工作， 必□打□ IPFILTER_LOG □个内核□□。 □个命令提供了□□不同的使用模式。

内建模式是默□的模式， 如果□不指定 -D 参数， 就会采用□□模式。

服□模式是持□地通□系□日志来□□的工作模式， □□， □就可以通□□看日志来了解□去曾□□生□的事情。 □□模式是 FreeBSD 和 IPFILTER 配合工作的模式。 由于在 FreeBSD 中提供了一个内建的系□日志自□□□功能， 因此， 使用 syslogd(8) 比默□的将日志信息□□到一个普通文件要好。 在默□的 rc.conf 文件中， ipmon_flags □句会指定 -Ds □志：

```
ipmon_flags="-Ds"                # D = 作□服□程序□□
                # s = 使用 syslog □□
                # v = □□ tcp □口大小、 ack 和□序号(seq)
                # n = 将 IP 和端口映射□名字
```

□□日志的好□是很明□的。 它提供了在事后重新□□相□信息， 例如□些包被□□， 以及□些包的来源地址等等。 □将□□□攻□者提供非常有用的第一手□料。

即使□用了日志机制， IPF 仍然不会□其□□□行任何日志□□工作。 防火□管理□可以决定□□集中的□些□□□日志， 并在□些□□上加入 log □□字。 一般来□， 只□□□拒□性的□□。

作□□例， 通常会有一条默□的、拒□所有网□流量的□□， 并指定 log □□字， 作□□的□□集的最后一条。 □□就能□看到所有没有匹配任何□□的数据包了。

31.5.7. IPMON 的日志

Syslogd 使用特殊的方法□日志数据□行分□。 它使用称□ "facility" 和 "level" 的□。 以 -Ds 模式□行的 IPMON 采用 local0 作□默□的 "facility" 名。 如果需要， 可以用下列 levels 来□一□区分数据：

```
LOG_INFO - 使用 "log" □□字指定的通□或阻止□作
LOG_NOTICE - 同□□□通□的那些数据包
LOG_WARNING - 同□□□阻止的数据包
LOG_ERR - □一□□□含不完整的包□的数据包
```

要□置 IPFILTER 来将所有的数据□□到 /var/log/ipfilter.log， 需要首先建立□个文件。 下面的命令可以完成□个工作：

```
# touch /var/log/ipfilter.log
```

syslogd(8) 功能可以通□在 /etc/syslog.conf 文件中的□句来定□。 syslog.conf 提供了相当多的用以控制 syslog 如何□理□似 IPF □□的用用程序所□生的系□消息的方法。

□需要将下列□句加到 /etc/syslog.conf：

```
local0.* /var/log/ipfilter.log
```

□里的 local0.* 表示把所有的相□日志信息写到指定的文件中。

要□ /etc/syslog.conf 中的修改立即生效， 可以重新□□□算机， 或者通□□行 /etc/rc.d/syslogd reload 来

□它重新□取 /etc/syslog.conf。

不要忘了修改 /etc/newsyslog.conf 来□□□建的日志□行□□。

31.5.8. □□消息的格式

由 ipmon 生成的消息由空格分隔的数据字段□成。 所有的消息都包含的字段是：

1. 接到数据包的日期。
2. 接到数据包的□□。 其格式□ HH:MM:SS.F， 分□是小□、 分□、 秒， 以及分秒 (□个数字可能有□多位)。
3. □理数据包的网□接口名字， 例如 dc0。
4. □和□□的□号， 例如 @0:17。

可以通□ ipfstat -in 来□看□些信息。

1. □作： p 表示通□， b 表示阻止， S 表示包□不全， n 表示没有匹配任何□□， L 表示 log □□。 □示□些□志的□序是： S, p, b, n, L。 大写的 P 或 B 表示□□包的原因是某个全局的日志配置，而不是某个特定的□□。
2. 地址。 □□□上包括三部分： 源地址和端口 (以逗号分□)， 一个 → 符号， 以及目的地址和端口， 例如： 209.53.17.22,80 → 198.73.220.17,1722。
3. PR， 后跟□□名称或□号， 例如： PR tcp。
4. len， 后跟包□的□度， 以及包的□□度， 例如： len 20 40。

□于 TCP 包， □□会包括一个附加的字段， 由一个□字号□始， 之后是表示所□置的□志的一个字母。 □参□ ipf(5) □机手册， 以了解□些字母所□□的□志。

□于 ICMP 包， □在最后会有□个字段。 前一个□是 "ICMP"， 而后一个□是 ICMP 消息和子消息的□型， 中□以斜□分□， 例如 ICMP 3/3 表示端口不可□消息。

31.5.9. □建采用符号替□的□□脚本

一些有□□的 IPF 会□建包含□□的文件， 并把它□写成能□与符号替□脚本兼容的方式。 □□做最大的好□是能□在修改□只修改符号名字所代表的□， 而在脚本□行□直接替□掉所有的名符。 作□脚本， 可以使用符号替□来把那些□常使用的□直接用于多个□□。 下面将□出一个例子。

□个脚本所使用的□法与 sh(1)、 csh(1)， 以及 tcsh(1) 脚本。

符号替□的前□字段是美元符号： $。

符号字段不使用 $ 前□。

希望替□符号字段的□， 必□使用双引号 (") 括起来。

□的□□文件的□□□似□□：

```
############## IPF □□脚本的□□ ########################
oif="dc0"            # 外网接口的名字
odns="192.0.2.11"    # ISP 的 DNS 服□器 IP 地址
myip="192.0.2.7"     # 来自 ISP 的静□ IP 地址
ks="keep state"
fks="flags S keep state"

# 可以使用□个脚本来建立 /etc/ipf.rules 文件,
# 也可以 "直接地" □行它。
#
# □□除□个注□号之一。
#
# 1) 保留下面一行, □□建 /etc/ipf.rules：
#cat > /etc/ipf.rules << EOF
#
# 2) 保留下面一行, □ "直接地" □行脚本：
/sbin/ipf -Fa -f - << EOF

# 允□□出到我的 ISP 的域名服□器的□□
pass out quick on $oif proto tcp from any to $odns port = 53 $fks
pass out quick on $oif proto udp from any to $odns port = 53 $ks

# 允□□出未加密的 www □□□求
pass out quick on $oif proto tcp from $myip to any port = 80 $fks

# 允□□出使用 TLS SSL 加密的 https www □□□求
pass out quick on $oif proto tcp from $myip to any port = 443 $fks
EOF
################### IPF □□脚本的□束 ########################
```

□就是所需的全部内容。 □个□□本身并不重要， 它□主要是用于体□如何使用符号代□字段， 以及如何完成□的替□。 如果上面的例子的名字是 /etc/ipf.rules.script， 就可以通□□入下面的命令来重新加□□□：

```
# sh /etc/ipf.rules.script
```

在□□文件中嵌入符号有一个□□： IPF 无法□□符号替□， 因此它不能直接地□取□□的脚本。

□个脚本可以使用下面□□方法之一来使用：

- 去掉 `cat` 之前的注□， 并注□掉 `/sbin/ipf` □□的那一行。 像其他配置一□， 将 `ipfilter_enable="YES"` 放到 /etc/rc.conf 文件中， 并在此后立刻□行脚本， 以□建或更新 /etc/ipf.rules。
- 通□把 `ipfilter_enable="NO"` (□是默□□) 加到 /etc/rc.conf 中， 来禁止系□□□脚本□□ IPFILTER。

  在 /usr/local/etc/rc.d/ □□目□中□加一个□似下面的脚本。 □□□它起一个□而易□的名字， 例如 ipf.loadrules.sh。 □注意， .sh □展名是必需的。

```
#!/bin/sh
sh /etc/ipf.rules.script
```

脚本文件必□□置□属于 root， 并且属主可□、 可写、 可□行。

```
# chmod 700 /usr/local/etc/rc.d/ipf.loadrules.sh
```

□□， 在系□□□□， 就会自□加□□的 IPF □□了。

31.5.10. IPF □□集

□□集是指一□□写好的依据包的□决策允□通□或阻止 IPF □□。 包的双向交□□成了一个会□交互。 防火□□□集会作用于来自于 Internet 公网的包以及由系□□出来回□□些包的数据包。 □一个 TCP/IP 服□ (例如 telnet, www, □件等等) 都由□□□先定□了其特□ (□听) 端口。 □到特定服□的包会从源地址使用非特□ (高□号) 端口□出， 并□到特定服□在目的地址的□□端口。 所有□些参数 (例如： 端口和地址） 都是可以□防火□□□所利用的， 判□是否允□服□通□的□准。

IPF 最初被写成使用一□称作 "以最后匹配的□□□准" 的□理□□， 且只能□理无状□的□□。 随着□代的□展， IPF □行了改□， 并提供了 "quick" □□， 以及一个有状□的 "keep state" □□。 后者使□理□□迅速地跟上了□代的□伐。

□一□中提供的一些指□， 是基于使用包含 "quick" □□和有状□的 "keep state" □□来□行□述的。 □些是□写明示允□防火□□□集的基本要素。

当□防火□□□□行操作□， □ □慎行事。 某些配置可能会 将□反□在 服□器外面。 保□起□， □可以考□在第一次□行防火□配置□在本地控制台上， 而不是□程， 如通□ ssh 来□行。

31.5.11. □□□法

□里□出的□□□法已□□化到只□理那些新式的□状□□□， 并且都是 "第一个匹配的□□□□" □□的。 要了解完整的□□□□□法描述， □参□ ipf(8) □机手册。

以 # 字符□□的内容会被□□是注□。 □些注□可以出□在一行□□的末尾， 或者独占一行。 空行会被忽略。

□□由□□字□成。 □些□□字必□以一定的□序， 从左到右出□在一行上。 接下来的文字中□□字将使用粗体表示。 某些□□字可能提供了子□□， □些子□□本身可能也是□□字， 而且可能会提供更多的子□□。 下面的文字中， □□□法都使用粗体的小□□□呈□， 并介□了其上下文。

ACTION IN-OUT OPTIONS SELECTION STATEFUL PROTO SRC_ADDR,DST_ADDR OBJECT PORT_NUM TCP_FLAG STATEFUL

ACTION = block | pass

IN-OUT = in | out

OPTIONS = log | quick | on 网□接口的名字

SELECTION = proto □□名称 | 源/目的 IP | port = 端口号 | flags □志□

PROTO = tcp/udp | udp | tcp | icmp

SRC_ADD,DST_ADDR = all | from □象 to □象

OBJECT = IP地址 | any

PORT_NUM = port 端口号

TCP_FLAG = S

STATEFUL = keep state

31.5.11.1. ACTION (□作)

□作□表示匹配□□的包□采取什□□作。 □一个□□ 必□ 包含一个□作。 可以使用下面□□□作之一：

`block` 表示如果□□与包匹配， □□□包。

`pass` 表示如果□□与包匹配， □允□包通□防火□。

31.5.11.2. IN-OUT

□个□□器□□都必□明□地指定是流入□是流出的□□。 下一个□□字必□要□是 `in`， 要□是 `out`， 否□将无法通□□法□□。

`in` 表示□□□被□用于□□从 Internet 公网上收到的数据包。

`out` 表示□□□被□用于即将□出到 Internet 的数据包。

31.5.11.3. OPTIONS

□些□□必□按下面指定的□序出□。

`log` 表示包□□被写入到 ipl 日志 (如前面 LOGGING 小□所介□的那□)， 如果它与□□匹配的□。

`quick` 表示如果□出的参数与包匹配， □以□个□□□准， □使得能□ "短路" 掉后面的□□。 □个□□□于使用新式的□理□□是必需的。

`on` 表示将网□接口的名称作□□□参数的一部分。 接口的名字会在 ifconfig(8) 的□出中□示。 使用□个□□，□□□只会□用到某一个网□接口上的出入数据包上。 要配置新式的□理□□， 必□使用□个□□。

当□□包□， 包的□会被写入到 IPL 包日志□□□中。 □跟 `log` □□字， 可以使用下面几个修□符 (按照下列□序)：

`body` 表示□同□□□包的前 128 字□的内容。

`first` 如果 `log` □□字和 `keep state` □□同□使用， □□个□□只在第一个包上触□， □□就不用□□□一个 "keep state" 包信息了。

31.5.11.4. SELECTION

□一□所介□的□□字可以用于所□察的包的属性。 有一个□□字主□， 以及一□子□□□□字， □必□从他□中□□一个。 以下是一些通用的属性， 它□必□按下面的□序使用：

31.5.11.5. PROTO

`proto` 是一个主□□□字， 它必□与某个相□的子□□□□字配合使用。 □个□的作用是匹配某个特定的□□。 要使用新式的□□□理□□， 就必□使用□个□□。

`tcp/udp | udp | tcp | icmp` 或其他在 /etc/protocols 中定□的□□。 特殊的□□□□字 `tcp/udp` 可以用于匹配 TCP 或 UDP 包， 引入□个□□字的作用是是避免大量的重□□□的麻□。

31.5.11.6. SRC_ADDR/DST_ADDR

使用 `all` □□□， 基本上相当于 "from any to any" 在没有配合其他□□字的情形。

`from src to dst`： `from` 和 `to` □□字主要是用来匹配 IP 地址。 所有的□□都必□ 同□ □出源和目的□个参数。 `any` 是一个可以用于匹配任意 IP 地址的特殊□□字。 例如， □可以使用 `from any to any` 或 `from 0.0.0.0/0 to any` 或 `from any to 0.0.0.0/0` 或 `from 0.0.0.0 to any` 以及 `from any to 0.0.0.0`。

如果无法使用子网掩□来表示 IP 的□， 表□地址就会很麻□。 使用 net-mgmt/ipcalc port 可以□助□行□算。 □参□下面的网□了解如何撰写□度掩□： http://jodies.de/ipcalc。

31.5.11.7. PORT

如果□源或目的指定了匹配端口， □□就只能□用于 TCP 和 UDP 包了。 当□写端口比□□□□， 可以指定 /etc/services 中所定□的名字， 也可以直接用端口号来指定。 如果端口号出□在源□象一□， □被□□是源端口号； 反之， □被□□是目的端口号。 要使用新式的□□□理□□， 就必□与 `to` □象配合使用□个□□。 使用的例子： `from any to any port = 80`

□□个端口的比□可以多□方式□行， 并可使用不同的比□算符。 此外， □可以指定端口的□□。

port "=" | "!=" | "<" | ">" | "⇐" | ">=" | "eq" | "ne" | "lt" | "gt" | "le" | "ge".

要指定端口□□， 可以使用 "<>" | "><"。

在源和目的匹配参数之后， 需要使用下面□个参数， 才能□使用新式的□□□理□□。

31.5.11.8. TCP_FLAG

□志只□ TCP □□有用。 □些字母用来表□ TCP 包□的□志。

新式的□□□理□□使用 `flags S` 参数来□□ tcp 会□□始的□求。

31.5.11.9. STATEFUL

`keep state` 表示如果有一个包与□□匹配， □其□□参数□激活有状□的□□机制。

如果使用新式的□理□□， □□个□□是必需的。

31.5.12. 有状□□□

有状□□□将网□流量当作一□双向的包交□来□理。 如果激活它， keep-state 会□□地□□一个相□的包在双向会□交互□程中□生内部□□。 它能□□□□起者和包的目的地之□的会□是有效的双向包交□□程的一部分。 如果包与□些□□不符， □将自□地拒□。

状□保持也使得 ICMP 包能□与 TCP 或 UDP 会□相□。 因此， 如果□在□□网站□收到允□的状□保持□□匹配的 ICMP □型 3 代□ 4 □□， □□些□□会被自□地允□□入。 所有 IPF 能□□理的包， 都可以作□某□活□会□的一部分，即使它是□一□□□的， 也会被允□□入。

所□生的事情是：

将要通□□入 Internet 公网的网□接口□出的包， 首先会□□□□状□表的□□。 如果包与会□中□期的下一个包匹配，防火□就会允□包通□， 并更新状□表中的会□的交互流信息。 不属于活□会□的包， □□□地交□□出□□集去□□。

□到□入 Internet 公网接口的包， 也会先□□□□状□表的□□。 如果包与会□中□期的下一个包匹配， 防火□就会允□包通□， 并更新状□表中的会□的交互流信息。 不属于活□会□的包， □□□地交□□入□□集去□□。

当会□□束□， □□的□会在□□状□表中□除。

有状□□□使得□能□集中于阻止/允□新的会□。 一旦新会□被允□通□， □所有后□的包就都被自□地允□通□， 而□造的包□被自□地拒□。 如果新的会□被阻止， □后□的包也都不会被允□通□。 有状□□□从技□角度而言，在阻止目前攻□者常用的洪水式攻□来□， 具有更好的抗御能力。

31.5.13. 明示允□□□集的例子

下面的□□集是如何□写非常安全的明示允□防火□□□集的一个□例。 明示允□防火□只□允□的服□ pass (通□)，而所有其他的□□都会被默□地拒□。 期望用来保□其他机器的防火□， 通常也叫做 "网□防火□"， □使用至少□个网□接口， 并且通常只有一个接入到受信的一端 (LAN)， 而□一□□接入不受信的一端 (Internet 公网)。□外， 防火□也可以配置□只保□它所□行的那个系□ - □□□型称作 "主机防火□"， 通常在接入不受信网□的服□器上使用。

包括 FreeBSD 在内的所有□ UNIX® 系□通常都会使用 lo0 和 IP 地址 127.0.0.1 用于操作系□中内部的通□。防火□□□必□允□□些包无阻碍地通□。

接入 Internet 公网的网□接口， 是放置□□并允□将□□□求□到 Internet 以及接收□□的地方。 □有可能是用□模式的 PPP tun0 接口， 如果□的网□同 DSL 或□□□制解□器相□的□。

如果有网□是直接接入私有网段的， □些网□接口就可能需要配置允□来自□些 LAN 的包在彼此之□，以及到外界 (Internet) 上的□□的通□□□。

一般□来， □□□被□□□三个主要的小□： 所有允□自由通□的接口□□， □到公网接口的□□， 以及□入公网接口的□□。

□一个公网接口□□中， □常会匹配到的□□□□放置在尽可能□前的位置。 而最后一个□□□□是阻止包通□， 并□□它□。

下面防火□□□集中， Outbound 部分是一些使用 pass 的□□， □些□□指定了允□□□的公网 Internet 服□，并且指定了 quick、 on、 proto、 port， 以及 keep state □些□□。 proto tcp □□□指定了 flag □个□□， □□会□的第一个包将出□状□机制。

接收部分□首先阻止所有不希望的包， □□做有□个不同的原因。 其一是□意的包可能和某些允□的流量□□存在部分匹配， 而我□希望阻止， 而不是□□些包□□与 allow □□部分匹配就允□它□□入。 其二是， 已□□信要阻止的包被拒□□件事， 往往并不是我□需要□注的， 因此只要□□地予以阻止即可。 防火□□□集中的□个部分的最后一条□□都是阻止并□□包， □有助于□逮捕攻□者留下法律所要求的□据。

□外一个需要注意的事情是□保系□□不希望的数据包不做回□。 无效的包□被□□和消失。 □□， 攻□者便无法知道包是否到□了□的系□。 攻□者□系□了解的越少， 攻陷系□所需的□□也就越多。 包含 log first □□的□□只会□□它□第一次被触□□的包， 在例子中□个□□被用于□□ nmap OS 指□探□ □□。 security/nmap 是攻□者常用的一□用于探□目□系□所用操作系□的工具。

如果□看到了 log first □□的日志， 就□□用 ipfstat -hio 命令来看看那个□□被匹配的次数。 如果数目□大，□表示系□正在受到洪水式攻□。

如果□□的包的端口号并不是□所知道的， 可以在 /etc/services 或 http://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers 了解端口号通常的用途。

参考下面的网□， 了解木□使用的端口： http://www.sans.org/security-resources/idfaq/oddports.php。

下面是我在自己的系□中使用的完整的， 非常安全的 明示允□ 防火□□□集。 直接使用□个□□集不会□□造成□□，□所要做的只是注□掉那些□不需要 pass(允□通□) 的服□。

如果在日志中□□了希望 阻止 的□□， 只需在 inbound 小□中□加一条阻止□□集可。

□必□将□一个□□中的 dc0 替□□□系□上接入 Internet 的网□接口名称， 例如， 用□□境下的 PPP □□是 tun0。

在 /etc/ipf.rules 中加入下面的内容：

```
######################################################################
# No restrictions on Inside LAN Interface for private network
# Not needed unless you have LAN
######################################################################

#pass out quick on xl0 all
#pass in quick on xl0 all

######################################################################
# No restrictions on Loopback Interface
######################################################################
pass in quick on lo0 all
pass out quick on lo0 all

######################################################################
# Interface facing Public Internet (Outbound Section)
# Match session start requests originating from behind the
# firewall on the private network
# or from this gateway server destined for the public Internet.
######################################################################

# Allow out access to my ISP's Domain name server.
# xxx must be the IP address of your ISP's DNS.
# Dup these lines if your ISP has more than one DNS server
```

```
# Get the IP addresses from /etc/resolv.conf file
pass out quick on dc0 proto tcp from any to xxx port = 53 flags S keep state
pass out quick on dc0 proto udp from any to xxx port = 53 keep state

# Allow out access to my ISP's DHCP server for cable or DSL networks.
# This rule is not needed for 'user ppp' type connection to the
# public Internet, so you can delete this whole group.
# Use the following rule and check log for IP address.
# Then put IP address in commented out rule & delete first rule
pass out log quick on dc0 proto udp from any to any port = 67 keep state
#pass out quick on dc0 proto udp from any to z.z.z.z port = 67 keep state

# Allow out non-secure standard www function
pass out quick on dc0 proto tcp from any to any port = 80 flags S keep state

# Allow out secure www function https over TLS SSL
pass out quick on dc0 proto tcp from any to any port = 443 flags S keep state

# Allow out send & get email function
pass out quick on dc0 proto tcp from any to any port = 110 flags S keep state
pass out quick on dc0 proto tcp from any to any port = 25 flags S keep state

# Allow out Time
pass out quick on dc0 proto tcp from any to any port = 37 flags S keep state

# Allow out nntp news
pass out quick on dc0 proto tcp from any to any port = 119 flags S keep state

# Allow out gateway & LAN users' non-secure FTP ( both passive & active modes)
# This function uses the IPNAT built in FTP proxy function coded in
# the nat rules file to make this single rule function correctly.
# If you want to use the pkg_add command to install application packages
# on your gateway system you need this rule.
pass out quick on dc0 proto tcp from any to any port = 21 flags S keep state

# Allow out ssh/sftp/scp (telnet/rlogin/FTP replacements)
# This function is using SSH (secure shell)
pass out quick on dc0 proto tcp from any to any port = 22 flags S keep state

# Allow out insecure Telnet
pass out quick on dc0 proto tcp from any to any port = 23 flags S keep state

# Allow out FreeBSD CVSup
pass out quick on dc0 proto tcp from any to any port = 5999 flags S keep state

# Allow out ping to public Internet
pass out quick on dc0 proto icmp from any to any icmp-type 8 keep state

# Allow out whois from LAN to public Internet
pass out quick on dc0 proto tcp from any to any port = 43 flags S keep state
```

```
# Block and log only the first occurrence of everything
# else that's trying to get out.
# This rule implements the default block
block out log first quick on dc0 all

#####################################################################
# Interface facing Public Internet (Inbound Section)
# Match packets originating from the public Internet
# destined for this gateway server or the private network.
#####################################################################

# Block all inbound traffic from non-routable or reserved address spaces
block in quick on dc0 from 192.168.0.0/16 to any    #RFC 1918 private IP
block in quick on dc0 from 172.16.0.0/12 to any     #RFC 1918 private IP
block in quick on dc0 from 10.0.0.0/8 to any        #RFC 1918 private IP
block in quick on dc0 from 127.0.0.0/8 to any       #loopback
block in quick on dc0 from 0.0.0.0/8 to any         #loopback
block in quick on dc0 from 169.254.0.0/16 to any    #DHCP auto-config
block in quick on dc0 from 192.0.2.0/24 to any      #reserved for docs
block in quick on dc0 from 204.152.64.0/23 to any   #Sun cluster interconnect
block in quick on dc0 from 224.0.0.0/3 to any       #Class D & E multicast

##### Block a bunch of different nasty things. ############
# That I do not want to see in the log

# Block frags
block in quick on dc0 all with frags

# Block short tcp packets
block in quick on dc0 proto tcp all with short

# block source routed packets
block in quick on dc0 all with opt lsrr
block in quick on dc0 all with opt ssrr

# Block nmap OS fingerprint attempts
# Log first occurrence of these so I can get their IP address
block in log first quick on dc0 proto tcp from any to any flags FUP

# Block anything with special options
block in quick on dc0 all with ipopts

# Block public pings
block in quick on dc0 proto icmp all icmp-type 8

# Block ident
block in quick on dc0 proto tcp from any to any port = 113

# Block all Netbios service. 137=name, 138=datagram, 139=session
# Netbios is MS/Windows sharing services.
# Block MS/Windows hosts2 name server requests 81
```

```
block in log first quick on dc0 proto tcp/udp from any to any port = 137
block in log first quick on dc0 proto tcp/udp from any to any port = 138
block in log first quick on dc0 proto tcp/udp from any to any port = 139
block in log first quick on dc0 proto tcp/udp from any to any port = 81

# Allow traffic in from ISP's DHCP server. This rule must contain
# the IP address of your ISP's DHCP server as it's the only
# authorized source to send this packet type. Only necessary for
# cable or DSL configurations. This rule is not needed for
# 'user ppp' type connection to the public Internet.
# This is the same IP address you captured and
# used in the outbound section.
pass in quick on dc0 proto udp from z.z.z.z to any port = 68 keep state

# Allow in standard www function because I have apache server
pass in quick on dc0 proto tcp from any to any port = 80 flags S keep state

# Allow in non-secure Telnet session from public Internet
# labeled non-secure because ID/PW passed over public Internet as clear text.
# Delete this sample group if you do not have telnet server enabled.
#pass in quick on dc0 proto tcp from any to any port = 23 flags S keep state

# Allow in secure FTP, Telnet, and SCP from public Internet
# This function is using SSH (secure shell)
pass in quick on dc0 proto tcp from any to any port = 22 flags S keep state

# Block and log only first occurrence of all remaining traffic
# coming into the firewall. The logging of only the first
# occurrence avoids filling up disk with Denial of Service logs.
# This rule implements the default block.
block in log first quick on dc0 all
##################### End of rules file ########################################
```

31.5.14. NAT

NAT 是 网□地址□□(Network Address Translation) 的□写。 □于那些熟悉 Linux® 的人来□， □个概念叫做 IP □装 (Masquerading)； NAT 和 IP □装是完全一□的概念。 由 IPF 的 NAT 提供的一□功能是， 将防火□后的本地局域网 (LAN) 共享一个 ISP 提供的 IP 地址来接入 Internet 公网。

有些人可能会□， □什□需要□□做。 一般而言， ISP 会□非商□用□提供□□的 IP 地址。 □□地址意味着□次登□到 ISP 都有可能得到不同的 IP 地址， 无□是采用□□□号登□， 或使用 cable 以及 DSL □制解□器的方式。 □个 IP 是□与 Internet 公网交互□使用的身□。

□在考□家中有五台 PC 需要□□ Internet 的情形。 □可能需要向 ISP □□一台 PC 所使用的独立的 Internet □号付□， 并且□有五根□□□。

有了 NAT， □就只需要一个 ISP □号， 然后将□外四台 PC 的网□通□交□机□接起来， 并通□□行 FreeBSD 系□的那台机器作□网□□接出去。 NAT 会自□地将□一台 PC 在内网的 LAN IP 地址， 在□□防火□□□□□公网的 IP 地址。 此外， 当数据包返回□， 也将□行逆向的□□。

在 IP 地址空□中， 有一些特殊的□□是保留供□□ NAT 的内网 LAN IP 地址使用的。 根据 RFC 1918，可以使用下面□些 IP □□用于内网， 它□不会在 Internet 公网上路由：

起始 IP 10.0.0.0	-	□束 IP 10.255.255.255
起始 IP 172.16.0.0	-	□束 IP 172.31.255.255
起始 IP 192.168.0.0	-	□束 IP 192.168.255.255

31.5.15. IPNAT

NAT □□是通□ ipnat 命令加□的。 默□情况下， NAT □□会保存在 /etc/ipnat.rules 文件中。 □参□ ipnat(1) 了解更多的□情。

如果在 NAT 已□□□之后想要修改 NAT □□， 可以修改保存 NAT □□的那个文件， 然后在□行 ipnat 命令□加上 -CF 参数， 以□除在用的 NAT 内部□□表， 以及所有地址翻□表中已有的□。

要重新加□ NAT □□， 可以使用□似下面的命令：

```
# ipnat -CF -f /etc/ipnat.rules
```

如果想要看看□系□上 NAT 的□□信息， 可以用下面的命令：

```
# ipnat -s
```

要列出当前的 NAT 表的映射□系， 使用下面的命令：

```
# ipnat -l
```

要□示□□的信息并□示与□□□理和当前的□□/表□：

```
# ipnat -v
```

31.5.16. IPNAT □□

NAT □□非常的□活， 能□□□商□用□和家庭用□的各□不同的需求。

□里所介□的□□□法已□被□化， 以□□非商用□境中的一般情况。 完整的□□□法描述， □参考 ipnat(5) □机手册中的介□。

NAT □□的写法与下面的例子□似：

```
map IF LAN_IP_RANGE -> PUBLIC_ADDRESS
```

□□□ map 出□在□□的最前面。

将 *IF* 替□□□外的网□接口名。

LAN_IP_RANGE 是内网中的客□机使用的地址□□。 通常情况下， □□□是□似 192.168.1.0/24 的地址。

PUBLIC_ADDRESS 既可以是外网的 IP 地址， 也可以是 0/32 □个特殊的□□字， 它表示分配到 *IF* 上的所有地址。

31.5.17. NAT 的工作原理

当包从 LAN 到□防火□， 而目的地址是公网地址□， 它首先会通□ outbound □□□□。 接下来， NAT 会得到包， 并按自□向下的□序□理□□， 而第一个匹配的□□将生效。 NAT 接下来会根据包□□的接口名字和源 IP 地址□□所有的□□。 如果包和某个 NAT □□匹配， □会□□包的 (源 IP 地址， 例如， 内网的 IP 地址) 是否在 NAT □□中箭□左□指定的 IP 地址□□匹配。 如果匹配， □包的原地址将被根据用 0/32 □□字指定的 IP 地址重写。 NAT 将向它的内部 NAT 表□送此地址， □□， 当包从 Internet 公网中返回□， 就能□把地址映射回原先的内网 IP 地址， 并在随后使用□□器□□来□理。

31.5.18. □用 IPNAT

要□用 IPNAT， 只需在 /etc/rc.conf 中加入下面一些□句。

使机器能□在不同的网□接口之□□行包的□□， 需要：

```
gateway_enable="YES"
```

□次□机□自□□□ IPNAT：

```
ipnat_enable="YES"
```

指定 IPNAT □□集文件：

```
ipnat_rules="/etc/ipnat.rules"
```

31.5.19. 大型 LAN 中的 NAT

□于在一个 LAN 中有大量 PC， 以及包含多个 LAN 的情形， 把所有的内网 IP 地址都映射到同一个公网 IP 上会□致□源不□的□□， 因□同一个端口可能在□多做了 NAT 的 LAN PC 上被多次使用， 并□致□撞。 有□□方法来□解□个□□。

31.5.19.1. 指定使用□些端口

普通的 NAT □□□似于：

```
map dc0 192.168.1.0/24 -> 0/32
```

上面的□□中， 包的源端口在包通□ IPNAT □□不会□生□化的。 通□使用 portmap □□字， □可以要求 IPNAT 只使用指定□□内的端口地址。 比如□， 下面的□□将□ IPNAT 把源端口改□指定□□内的端口：

```
map dc0 192.168.1.0/24 -> 0/32 portmap tcp/udp 20000:60000
```

使用 auto □□字可以□配置□得更□□一些， 它会要求 IPNAT 自□地□□可用的端口并使用：

```
map dc0 192.168.1.0/24 -> 0/32 portmap tcp/udp auto
```

31.5.19.2. 使用公网地址池

□很大的 LAN 而言， □有一天会□到□□一个□界□， 此□的 LAN 地址已□多到了无法只用一个公网地址表□的程度。 如果有可用的一□公网 IP 地址， □可以将□些地址作□一个 "地址池" 来使用， □ IPNAT 来从□些公网 IP 地址中挑□用于□包的地址， 并将其□□些包□建映射□系。

例如， 如果将下面□个把所有包都映射到同一公网 IP 地址的□□：

```
map dc0 192.168.1.0/24 -> 204.134.75.1
```

□作修改， 就可以用子网掩□来表□ IP 地址□□：

```
map dc0 192.168.1.0/24 -> 204.134.75.0/255.255.255.0
```

或者用 CIDR □法来指定的一□地址了：

```
map dc0 192.168.1.0/24 -> 204.134.75.0/24
```

31.5.20. 端口重定向

非常流行的一□做法是， 将 web 服□器、 □件服□器、 数据□服□器以及 DNS 分□放到 LAN 上的不同的 PC 上。 □□情况下， 来自□些服□器的网□流量仍然□□被 NAT， 但必□有□法把□入的流量□到□□的局域网的 PC 上。 IPNAT 提供了 NAT 重定向机制来解决□个□□。 考□下面的情况， □的 web 服□器的 LAN 地址是 10.0.10.25， 而□的唯一的公网 IP 地址是 20.20.20.5， □可以□写□□的□□：

```
rdr dc0 20.20.20.5/32 port 80 -> 10.0.10.25 port 80
```

或者：

```
rdr dc0 0.0.0.0/0 port 80 -> 10.0.10.25 port 80
```

□外， 也可以□ LAN 地址 10.0.10.33 上□行的 LAN DNS 服□器来□理公网上的 DNS □求：

```
rdr dc0 20.20.20.5/32 port 53 -> 10.0.10.33 port 53 udp
```

31.5.21. FTP 和 NAT

FTP 是一个在 Internet 如今天□□□人所熟知之前就已□出□的恐□， 那□， 研究机□和大学是通□租用的□路□到一起的， 而 FTP □被用于在科研人□之□共享大文件。 那□， 数据的安全性并不是需要考□的事情。 若干年之后， FTP □□□被埋□了正在形成中的 Internet 骨干， 而它使用明文来交□用□名和口令的缺点，并没有随着新出□的一些安全需求而得到改□。 FTP 提供了□□不同的□格， 即主□模式和被□模式。 □者的区□在于数据通道的建立方式。 被□模式相□而言要更加安全， 因□数据通道是由□起 ftp 会□的一方建立的。 □于 FTP 以及它所提供的不同模式， 在 http://www.slacksite.com/other/ftp.html □行了很好的□述。

31.5.21.1. IPNAT □□

IPNAT 提供了一个内建的 FTP 代理□□， 它可以在 NAT map □□中指定。 它能□□□所有外□的 FTP 主□或被□模式的会□□始□求， 并□□地□建□□性的□□器□□， 只打□用于数据通道的端口号。 □□， 就消除了 FTP 一般会□防火□□来的， 需要大□□地打□高端口所可能□来的安全□患。

下面的□□可以□理来自内网的 FTP □□：

```
map dc0 10.0.10.0/29 -> 0/32 proxy port 21 ftp/tcp
```

□个□□能□□理来自网□的 FTP □□：

```
map dc0 0.0.0.0/0 -> 0/32 proxy port 21 ftp/tcp
```

□个□□理所有来自内网的非 FTP 网□流量：

```
map dc0 10.0.10.0/29 -> 0/32
```

FTP map □□□□在普通的 map □□之前出□。 所有的包会从最上面的第一个□□□始□行□□。 匹配的□序是网□名称， 内网源 IP 地址， 以及它是否是 FTP 包。 如果所有□些□□都匹配成功， □ FTP 代理将建立一个□□的□□□□， 以便□ FTP 会□的数据包能□正常出入， 同□□□些包□行 NAT。 所有的 LAN 数据包，如果没有匹配第一条□□， □会□□□□匹配下面的□□， 并最□被 NAT。

31.5.21.2. IPNAT FTP □□□□

如果使用了 NAT FTP 代理， □只需要□ FTP □建一个□□。

如果不使用 FTP 代理， 就需要下面□三个□□：

```
# Allow out LAN PC client FTP to public Internet
# Active and passive modes
pass out quick on rl0 proto tcp from any to any port = 21 flags S keep state

# Allow out passive mode data channel high order port numbers
pass out quick on rl0 proto tcp from any to any port > 1024 flags S keep state

# Active mode let data channel in from FTP server
pass in quick on rl0 proto tcp from any to any port = 20 flags S keep state
```

31.6. IPFW

IPFIREWALL (IPFW) 是一个由 FreeBSD □起的防火□□用□件， 它由 FreeBSD 的志□者成□□写和□□。 它使用了□□的无状□□□和□□□写方式， 以期□到□□状□□□所期望的目□。

□准的 FreeBSD 安装中， IPFW 所□出的□□集□例 (可以在 /etc/rc.firewall 和 /etc/rc.firewall6 中□到) 非常□□， 建□不要不加修改地直接使用。 □□例中没有使用状□□□， 而□功能在大部分的配置中都是非常有用的， 因此□一□并不以系□自□的□例作□基□。

IPFW 的无状□□□□法， 是由一□提供□□的□□能力的技□支持的， □□技□□□超出了一般的防火□安装人□的知□水平。 IPFW 是□□足□□用□， 以及掌握先□技□的□□□好者□□于高□的包□□需求而□□的。 要完全□放 IPFW 的□□所□有的□大能力， 需要□不同的□□的□□有深入的了解， 并根据它□独特的包□信息来□写□□。 □一□□的□□□述超出了□本手册的□□。

IPFW 由七个部分□成， 其主要□件是内核的防火□□□□□理器， 及其集成的数据包□□工具、 日志工具、 用以触□ NAT 工具的 divert (□□) □□、 高□特殊用途工具、 dummynet 流量整形机制， fwd rule □□工具， □接工具， 以及 ipstealth 工具。 IPFW 支持 IPv4 和 IPv6。

31.6.1. □用 IPFW

IPFW 是基本的 FreeBSD 安装的一部分， 以□独的可加□内核模□的形式提供。 如果在 rc.conf 中加入 firewall_enable="YES" □句， 就会自□地加□□□的内核模□。 除非□打算使用由它提供的 NAT 功能， 一般情况下并不需要把 IPFW □□ FreeBSD 的内核。

如果将 firewall_enable="YES" 加入到 rc.conf 中并重新□□系□， □下列信息将在□□□程中， 以高亮的白色□示出来：

```
ipfw2 initialized, divert disabled, rule-based forwarding disabled, default to deny,
logging disabled
```

可加□内核模□在□□□加入了□□日志的能力。 要□用日志功能， 并配置□□日志□□的限制， 需要在 /etc/sysctl.conf 中加入一些配置。 □些□置将在重新□□之后生效：

```
net.inet.ip.fw.verbose=1
net.inet.ip.fw.verbose_limit=5
```

31.6.2. 内核□□

把下列□□在□□ FreeBSD 内核□就加入， 并不是□用 IPFW 所必需的， 除非□需要使用 NAT 功能。 □里只是将□些□□作□背景知□来介□。

```
options    IPFIREWALL
```

□个□□将 IPFW 作□内核的一部分来□用。

```
options    IPFIREWALL_VERBOSE
```

□个□□将□用□□通□ IPFW 的匹配了包含 log □□字□□的□一个包的功能。

```
options    IPFIREWALL_VERBOSE_LIMIT=5
```

以□□的方式， 限制通□ syslogd(8) □□的包的个数。 如果在比□□劣的□境下□□防火□的活□可能会需要□个□□。它能□避免潜在的□□ syslog 的洪水式拒□服□攻□。

```
options    IPFIREWALL_DEFAULT_TO_ACCEPT
```

□个□□默□地允□所有的包通□防火□， 如果□是第一次配置防火□， 使用□个□□将是一个不□的主意。

```
options    IPDIVERT
```

□一□□□用 NAT 功能。

如果内核□□中没有加入 IPFIREWALL_DEFAULT_TO_ACCEPT， 而配置使用的□□集中也没有明□地指定允□□接□入的□□， 默□情况下， □到本机和从本机□出的所有包都会被阻止。

31.6.3. /etc/rc.conf Options

□用防火□：

```
firewall_enable="YES"
```

要□□由 FreeBSD 提供的几□防火□□型中的一□来作□默□配置， □需要□□ /etc/rc.firewall 文件并□出合□的□型，然后在 /etc/rc.conf 中加入□似下面的配置：

```
firewall_type="open"
```

□□可以指定下列配置□□之一：

- `open` - 允□所有流量通□。
- `client` - 只保□本机。
- `simple` - 保□整个网□。
- `closed` - 完全禁止除回□□□之外的全部 IP 流量。
- `UNKNOWN` - 禁止加□防火□□□。
- filename - 到防火□□□文件的□□路径。

有□□加□自定□ ipfw 防火□□□的方法。 其一是将□量 `firewall_type` □□包含不□ ipfw(8) 命令行□□的 防火□□□文件的完整路径。 下面是一个□□的□□集例子：

```
add deny in
add deny out
```

除此之外， 也可以将 `firewall_script` □量□□包含 `ipfw` 命令的可□行脚本， □□□个脚本会在□□□自□□行。 与前面□□集文件等价的□□脚本如下：

`ipfw` 命令是在防火□□行□， 用于在其内部□□表中手工逐条添加或□除防火□□□的□准工具。 □一方法的□□在于，一旦□的□□□算机或停机， □所有□加或□除或修改的□□也就□掉了。 把所有的□□都写到一个文件中， 并在□□□使用□个文件来加□□□， 或一次大批量地替□防火□□□， 那□推□使用□里介□的方法。

`ipfw` 的□一个非常□用的功能是将所有正在□行的防火□□□□示出来。 IPFW 的□□机制会□□一个□□□□地□建□数器， 用以□□与它□匹配的包的数量。 在□□□□的□程中， 列出□□及其□数器是了解它□是否工作正常的重要手段。

按□序列出所有的□□：

```
# ipfw list
```

列出所有的□□， 同□□出最后一次匹配的□□□：

```
# ipfw -t list
```

列出所有的□□信息、 匹配□□的包的数量， 以及□□本身。 第一列是□□的□号， 随后是□出包匹配的数量，□入包的匹配数量， 最后是□□本身。

```
# ipfw -a list
```

列出所有的□□□□和静□□□：

```
# ipfw -d list
```

同□□示已□期的□□□□：

```
# ipfw -d -e list
```

将□数器清零：

```
# ipfw zero
```

只把□□号□ *NUM* 的□数器清零：

```
# ipfw zero NUM
```

31.6.4. IPFW □□集

□□集是指一□□写好的依据包的□决策允□通□或阻止 IPFW □□。 包的双向交□□成了一个会□交互。 防火□□□集会作用于来自于 Internet 公网的包以及由系□□出来回□□些包的数据包。 □一个 TCP/IP 服□ (例如 telnet, www, □件等等) 都由□□□先定□了其特□ (□听) 端口。 □到特定服□的包会从源地址使用非特□ (高□号) 端口□出， 并□到特定服□在目的地址的□□端口。 所有□些参数 (例如： 端口和地址） 都是可以□防火□□□所利用的， 判□是否允□服□通□的□准。

当有数据包□入防火□□， 会从□□集里的第一个□□□始□行比□， 并自□向下地□行匹配。 当包与某个□□□□参数相匹配□， 将会□行□□所定□的□作， 并停止□□集搜索。 □□策略， 通常也被称作 "最先匹配者□□" 的搜索方法。 如果没有任何与包相匹配的□□， 那□它就会根据□制的 IPFW 默□□□， 也就是 65535 号□□截□。 一般情况下□个□□是阻止包， 而且不□出任何回□。

如果□□定□的□作是 `count`、 `skipto` 或 `tee` □□的□， 搜索会□□。

□里所介□的□□， 都是使用了那些包含状□功能的， 也就是 `keep state`、 `limit`、 `in`、 `out` 以及 `via` □□的□□。 □是□写明示允□防火□□□集所需的基本框架。

在操作防火□□□□□□慎行事， 如果操作不当， 很容易将自己反□在外面。

31.6.4.1. □□□法

□里所介□的□□□法已□□□了□化， 只包括了建立□准的明示允□防火□□□集所必需的那些。 要了解完整的□□□法□明， □参□ ipfw(8) □机手册。

□□是由□□字□成的： □些□□字必□以特定的□序从左到右□写。 下面的介□中， □□字使用粗体表示。 某些□□字□包括了子□□， □些子□□本身可能也是□□字， 有些□可以包含更多的子□□。

用于表示□始一段注□。 它可以出□在一个□□的后面， 也可以独占一行。 空行会被忽略。

CMD RULE_NUMBER ACTION LOGGING SELECTION STATEFUL

31.6.4.1.1. CMD

□一个新的□□都□以 add 作□前□， 它表示将□□加入内部表。

31.6.4.1.2. RULE_NUMBER

□一条□□都与一个□□在 1 到 65535 之□的□□□号相□□。

31.6.4.1.3. ACTION

□一个□□可以与下列的□作之一相□□， 所指定的□作将在□入的数据包与□□所指定的□□□准相匹配□□行。

allow | accept | pass | permit

□些□□字都表示允□匹配□□的包通□防火□， 并停止□□搜索□□。

check-state

根据□□□□表□□数据包。 如果匹配， □□行□□所指定的□作， 亦即生成□□□□； 否□， □移到下一个□□。 check-state □□没有□□□准。 如果□□集中没有 check-state □□， □会在第一个 keep-state 或 limit □□□， □□□□□表□施□□。

deny | drop

□□个□□字都表示□□匹配□□的包。 同□， 停止□□搜索□□。

31.6.4.1.4. LOGGING

log or logamount

当数据包与□ `log` □□字的□□匹配□， 将通□名□ SECURITY 的 facility 来把消息□□到 syslogd(8)。 只有在□□的次数没有超□ logamount 参数所指定的次数□， 才会□□日志。 如果没有指定 `logamount`， □会以 sysctl □量 `net.inet.ip.fw.verbose_limit` 所指定的限制□准。 如果将□□□限制□之一指定□零， □表示不作限制。 如果□到了限制数， 可以通□将□□的日志□数或包□数清零来重新□用日志， □参□ `ipfw reset log` 命令来了解□□。

日志是在所有其他匹配条件都□□成功之后， 在□□包□施最□□作 (accept, deny) 之前□行的。 □可以自行决定□些□□□□用日志。

31.6.4.1.5. SELECTION

□一□所介□的□□字主要用来描述□□包的□些属性， 用以判断包是否与□□相匹配。 下面是一些通用的用于匹配包特征的属性， 它□必□按□序使用：

udp | tcp | icmp

也可以指定在 /etc/protocols 中所定□的□□。 □个□定□的是匹配的□□， 在□□中必□指定它。

from src to dst

`from` 和 `to` □□字用于匹配 IP 地址。 □□中必□ 同□ 指定源和目的□个参数。 如果需要匹配任意 IP 地址， 可以使用特殊□□字 `any`。 □有一个特殊□□字， 即 `me`， 用于匹配□的 FreeBSD 系□上所有网□接口上所配置的 IP 地址， 它可以用于表□网□上的其他□算机到防火□ (也就是本机)， 例如 `from me to any` 或 `from any to me` 或 `from 0.0.0.0/0 to any` 或 `from any to 0.0.0.0/0` 或 `from 0.0.0.0 to any` 或 `from any to 0.0.0.0` 以及 `from me to 0.0.0.0`。 IP 地址可以通□ □点的 IP 地址/掩□□度 (CIDR □法)， 或者一个□点的 IP 地址的形式来指定。 □是□写□□□所必需的。 使用 net-mgmt/ipcalc port 可以用来□化□算。 □于

□个工具的更多信息， 也可参考它的主□： http://jodies.de/ipcalc。

port number

□个参数主要用于那些支持端口号的□□ (例如 TCP 和 UDP)。 如果要通□端口号匹配某个□□， 就必□指定□个参数。 此外， 也可以通□服□的名字 (根据 /etc/services) 来指定服□， □□会比使用数字指定端口号直□一些。

in | out

相□地， 匹配□入和□出的包。 □里的 in 和 out 都是□□字， 在□写匹配□□□， 必需作□其他条件的一部分来使用。

via IF

根据指定的网□接口的名称精□地匹配□出的包。 □里的 via □□字将使得接口名称成□匹配□程的一部分。

setup

要匹配 TCP 会□的□起□求， 就必□使用它。

keep-state

□是一个必□使用的□□字。 在□生匹配□， 防火□将□建一个□□□□， 其默□行□是， 匹配使用同一□□的、从源到目的 IP/端口 的双向网□流量。

limit {src-addr | src-port | dst-addr | dst-port}

防火□只允□匹配□□□， 与指定的参数相同的 *N* 个□接。 可以指定至少一个源或目的地址及端口。 limit 和 keep-state 不能在同一□□中同□使用。 limit 提供了与 keep-state 相同的功能， 并□加了一些独有的能力。

31.6.4.2. 状□□□□□

有状□□□将网□流量当作一□双向的包交□来□理。 它提供了一□□外的□□能力， 用以□□会□中的包是否来自最初的□送者， 并在遵循双向包交□的□□□行会□。 如果包与□些□□不符， □将自□地拒□它□。

check-state 用来□□在 IPFW □□集中的包是否符合□□□□机制的□□。 如果匹配， □允□包通□， 此□防火□将□建一个新的□□□□来匹配双向交□中的下一个包。 如果不匹配， □将□□□□□集中的下一个□□。

□□□□机制在 SYN-flood 攻□下是脆弱的， 因□□□情况会□生大量的□□□□， 从而耗尽□源。 □了抵抗□□攻□， 从 FreeBSD 中加入了一个叫做 limit 的新□□。 □个□□可以用来限制符合□□的会□允□的并□□接数。 如果□□□□表中的□□数超□ limit 的限制数量， □包将被□□。

31.6.4.3. □□防火□消息

□□日志的好□是□而易□的： 它提供了在事后□□所□生的状况的方法， 例如□些包被□□了， □些包的来源和目的地， 从而□□提供□到攻□者所需的□据。

即使□用了日志机制， IPFW 也不会自行生成任何□□的日志。 防火□管理□需要指定□□集中的□些□□□□□□日志， 并在□些□□上□加 log □作。 一般来□， 只有 deny □□□□□日志， 例如□于□入的 ICMP ping 的 deny □□。 □外， □制 "默□的 ipfw □□ deny □□"， 并加入 log □作来作□□的□□集的最后一条□□也是很常□的用法。 □□， □就能看到没有匹配任何一条□□的那些数据包。

日志是一把双刃□， 如果不□慎地加以利用， □可能会陷入□多的日志数据中， 并□致磁□被日志塞□。 将磁□填□是 DoS 攻□最□老套的手法之一。 由于 syslogd 除了会将日志写入磁□之外， □会□出到 root 的控制台屏幕上， 因此有□多的日志信息是很□人□火的事情。

IPFIREWALL_VERBOSE_LIMIT=5 内核□□将限制同一个□□□到系□日志程序 syslogd(8) 的□□消息的数量。 当内核□用了□个□□□， 某一特定□□所□生的□□消息的数量将封□□□个数字。 一般来□， 没有□法从□□ 200 条一模一□的日志信息中□取更多有用的信息。 □例来□， 如果同一个□□□生了 5 次消息并被□□到 syslogd，余下的相同的消息将被□数， 并像下面□□□□ syslogd：

```
last message repeated 45 times
```

所有□□的数据包包消息， 默□情况下会最□写到 /var/log/security 文件中， 后者在 /etc/syslog.conf 文件里□行了定□。

31.6.4.4. □写□□脚本

□大多数有□□的 IPFW 用□会□建一个包含□□的文件， 并且， 按能□以脚本形式□行的方式来□写。□□做最大的一个好□是， 可以大批量地刷新防火□□□， 而无□重新□□系□就能□激活它□。 □□方法在□□新□□□会非常方便， 因□同一□程在需要□可以多次□行。 作□脚本， □可以使用符号替□来撰写那些□常需要使用的□， 并用同一个符号在多个□□中反□地表□它。 下面将□出一个例子。

□个脚本使用的□法同 sh(1)、 csh(1) 以及 tcsh(1) 脚本兼容。 符号替□字段使用美元符号 $ 作□前□。符号字段本身并不使用 $ 前□。 符号替□字段的□必□使用 "双引号" 括起来。

可以使用□似下面的□□文件：

```
################ start of example ipfw rules script #############
#
ipfw -q -f flush       # Delete all rules
# Set defaults
oif="tun0"             # out interface
odns="192.0.2.11"      # ISP's DNS server IP address
cmd="ipfw -q add "     # build rule prefix
ks="keep-state"        # just too lazy to key this each time
$cmd 00500 check-state
$cmd 00502 deny all from any to any frag
$cmd 00501 deny tcp from any to any established
$cmd 00600 allow tcp from any to any 80 out via $oif setup $ks
$cmd 00610 allow tcp from any to $odns 53 out via $oif setup $ks
$cmd 00611 allow udp from any to $odns 53 out via $oif $ks
################### End of example ipfw rules script #############
```

□就是所要做的全部事情了。 例子中的□□并不重要， 它□主要是用来表示如何使用符号替□。

如果把上面的例子保存到 /etc/ipfw.rules 文件中。 下面的命令来会重新加□□□。

```
# sh /etc/ipfw.rules
```

/etc/ipfw.rules □个文件可以放到任何位置， 也可以命名□随便什□□的名字。

也可以手工□行下面的命令来□到□似的目的：

```
# ipfw -q -f flush
# ipfw -q add check-state
# ipfw -q add deny all from any to any frag
# ipfw -q add deny tcp from any to any established
# ipfw -q add allow tcp from any to any 80 out via tun0 setup keep-state
# ipfw -q add allow tcp from any to 192.0.2.11 53 out via tun0 setup keep-state
# ipfw -q add 00611 allow udp from any to 192.0.2.11 53 out via tun0 keep-state
```

31.6.4.5. □状□□□集

以下的□□非-NAT □□集， 是如何□写非常安全的 '明示允□' 防火□的一个例子。 明示允□防火□只允□匹配了 pass □□的包通□， 而默□阻止所有的其他数据包。 用来保□整个网段的防火□， 至少需要有□个网□接口， 并且其上必□配置□□， 以便□防火□正常工作。

所有□ UNIX® 操作系□， 也包括 FreeBSD， 都□□□允□使用网□接口 lo0 和 IP 地址 127.0.0.1 来完成操作系□内部的通□。 防火□必□包含一□□□， 使□些数据包能□无障碍地收□。

接入 Internet 公网的那个网□接口上， □□配置授□和□□控制， 来限制□外的□□， 以及来自 Internet 公网的□□。 □个接口很可能是□的用□□ PPP 接口， 例如 tun0， 或者□接在 DSL 或□□ modem 上的网□。

如果有至少一个网□接入了防火□后的内网 LAN， □必□□□些接口配置□□， 以便□□些接口之□的包能□□□地通□。

所有的□□□被□□□三个部分， 所有□无阻碍地通□的□□， 公网的□出□□， 以及公网的接收□□。

公网接口相□的□□的□序， □□是最□常用到的放在尽可能□前的位置， 而最后一个□□， □□□是阻止那个接口在那一方向上的包。

□出部分的□□只包含一些 allow □□， 允□□定的那些唯一区分□□的端口号所指定的□□通□， 以允□□□ Internet 公网上的□些服□。 所有的□□中都指定了 proto、 port、 in/out、 via 以及 keep state □些□□。 proto tcp □□同□指定 setup □□， 来区分□始□□会□的包， 以触□将包放入 keep state □□表中的□作。

接收部分□首先阻止所有不希望的包， □□做有□个不同的原因。 其一是□意的包可能和某些允□的流量□□存在部分匹配， 而我□希望阻止， 而不是□□些包□□与 allow □□部分匹配就允□它□□入。 其二是， 已□□信要阻止的包被拒□□件事， 往往并不是我□需要□注的， 因此只要□□地予以阻止即可。 防火□□□集中的□个部分的最后一条□□都是阻止并□□包， □有助于□逮捕攻□者留下法律所要求的□据。

□外一个需要注意的事情是□保系□□不希望的数据包不做回□。 无效的包□被□□和消失。 □□， 攻□者便无法知道包是否到□了□的系□。 攻□者□系□了解的越少， 其攻□的□度也就越大。 如果不知道端口号， 可以□□ /etc/services/ 或到 http://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers 并□□一下端口号， 以了解其用途。 □外， □也可以在□个网□上了解常□木□所使用的端口： http://www.sans.org/security-resources/idfaq/oddports.php。

31.6.4.6. 明示允□□□集的例子

下面是一个非-NAT 的□□集， 它是一个完整的明示允□□□集。 使用它作□□的□□集不会有什□□□。 只需把那些不需要的服□□□的 pass □□注□掉就可以了。 如果□在日志中看到消息， 而且不想再看到它□，

只需在接收部分□加一个一个 deny □□。 □可能需要把 dc0 改□接入公网的接口的名字。 □于使用用□□ PPP 的用□而言， □□是 tun0。

□些□□遵循一定的模式。

- 所有□求 Internet 公网上服□的会□□始包， 都使用了 keep-state。
- 所有来自 Internet 的授□服□□求， 都采用了 limit □□来防止洪水式攻□。
- 所有的□□都使用了 in 或者 out 来□明方向。
- 所有的□□都使用了 via接口名 来指定□□匹配通□□一个接口的包。

□些□□都□放到 /etc/ipfw.rules。

```
################ Start of IPFW rules file ###################################
# Flush out the list before we begin.
ipfw -q -f flush

# Set rules command prefix
cmd="ipfw -q add"
pif="dc0"     # public interface name of NIC
              # facing the public Internet

######################################################################
# No restrictions on Inside LAN Interface for private network
# Not needed unless you have LAN.
# Change xl0 to your LAN NIC interface name
######################################################################
#$cmd 00005 allow all from any to any via xl0

######################################################################
# No restrictions on Loopback Interface
######################################################################
$cmd 00010 allow all from any to any via lo0

######################################################################
# Allow the packet through if it has previous been added to the
# the "dynamic" rules table by a allow keep-state statement.
######################################################################
$cmd 00015 check-state

######################################################################
# Interface facing Public Internet (Outbound Section)
# Interrogate session start requests originating from behind the
# firewall on the private network or from this gateway server
# destined for the public Internet.
######################################################################

# Allow out access to my ISP's Domain name server.
# x.x.x.x must be the IP address of your ISP.s DNS
# Dup these lines if your ISP has more than one DNS server
```

```
# Get the IP addresses from /etc/resolv.conf file
$cmd 00110 allow tcp from any to x.x.x.x 53 out via $pif setup keep-state
$cmd 00111 allow udp from any to x.x.x.x 53 out via $pif keep-state

# Allow out access to my ISP's DHCP server for cable/DSL configurations.
# This rule is not needed for .user ppp. connection to the public Internet.
# so you can delete this whole group.
# Use the following rule and check log for IP address.
# Then put IP address in commented out rule & delete first rule
$cmd 00120 allow log udp from any to any 67 out via $pif keep-state
#$cmd 00120 allow udp from any to x.x.x.x 67 out via $pif keep-state

# Allow out non-secure standard www function
$cmd 00200 allow tcp from any to any 80 out via $pif setup keep-state

# Allow out secure www function https over TLS SSL
$cmd 00220 allow tcp from any to any 443 out via $pif setup keep-state

# Allow out send & get email function
$cmd 00230 allow tcp from any to any 25 out via $pif setup keep-state
$cmd 00231 allow tcp from any to any 110 out via $pif setup keep-state

# Allow out FBSD (make install & CVSUP) functions
# Basically give user root "GOD" privileges.
$cmd 00240 allow tcp from me to any out via $pif setup keep-state uid root

# Allow out ping
$cmd 00250 allow icmp from any to any out via $pif keep-state

# Allow out Time
$cmd 00260 allow tcp from any to any 37 out via $pif setup keep-state

# Allow out nntp news (i.e., news groups)
$cmd 00270 allow tcp from any to any 119 out via $pif setup keep-state

# Allow out secure FTP, Telnet, and SCP
# This function is using SSH (secure shell)
$cmd 00280 allow tcp from any to any 22 out via $pif setup keep-state

# Allow out whois
$cmd 00290 allow tcp from any to any 43 out via $pif setup keep-state

# deny and log everything else that.s trying to get out.
# This rule enforces the block all by default logic.
$cmd 00299 deny log all from any to any out via $pif

######################################################################
# Interface facing Public Internet (Inbound Section)
# Check packets originating from the public Internet
# destined for this gateway server or the private network.
######################################################################
```

```
# Deny all inbound traffic from non-routable reserved address spaces
$cmd 00300 deny all from 192.168.0.0/16 to any in via $pif  #RFC 1918 private IP
$cmd 00301 deny all from 172.16.0.0/12 to any in via $pif     #RFC 1918 private IP
$cmd 00302 deny all from 10.0.0.0/8 to any in via $pif          #RFC 1918 private IP
$cmd 00303 deny all from 127.0.0.0/8 to any in via $pif        #loopback
$cmd 00304 deny all from 0.0.0.0/8 to any in via $pif            #loopback
$cmd 00305 deny all from 169.254.0.0/16 to any in via $pif   #DHCP auto-config
$cmd 00306 deny all from 192.0.2.0/24 to any in via $pif       #reserved for docs
$cmd 00307 deny all from 204.152.64.0/23 to any in via $pif  #Sun cluster interconnect
$cmd 00308 deny all from 224.0.0.0/3 to any in via $pif         #Class D & E multicast

# Deny public pings
$cmd 00310 deny icmp from any to any in via $pif

# Deny ident
$cmd 00315 deny tcp from any to any 113 in via $pif

# Deny all Netbios service. 137=name, 138=datagram, 139=session
# Netbios is MS/Windows sharing services.
# Block MS/Windows hosts2 name server requests 81
$cmd 00320 deny tcp from any to any 137 in via $pif
$cmd 00321 deny tcp from any to any 138 in via $pif
$cmd 00322 deny tcp from any to any 139 in via $pif
$cmd 00323 deny tcp from any to any 81 in via $pif

# Deny any late arriving packets
$cmd 00330 deny all from any to any frag in via $pif

# Deny ACK packets that did not match the dynamic rule table
$cmd 00332 deny tcp from any to any established in via $pif

# Allow traffic in from ISP's DHCP server. This rule must contain
# the IP address of your ISP.s DHCP server as it.s the only
# authorized source to send this packet type.
# Only necessary for cable or DSL configurations.
# This rule is not needed for .user ppp. type connection to
# the public Internet. This is the same IP address you captured
# and used in the outbound section.
#$cmd 00360 allow udp from any to x.x.x.x 67 in via $pif keep-state

# Allow in standard www function because I have apache server
$cmd 00400 allow tcp from any to me 80 in via $pif setup limit src-addr 2

# Allow in secure FTP, Telnet, and SCP from public Internet
$cmd 00410 allow tcp from any to me 22 in via $pif setup limit src-addr 2

# Allow in non-secure Telnet session from public Internet
# labeled non-secure because ID & PW are passed over public
# Internet as clear text.
# Delete this sample group if you do not have telnet server enabled.
```

```
$cmd 00420 allow tcp from any to me 23 in via $pif setup limit src-addr 2

# Reject & Log all incoming connections from the outside
$cmd 00499 deny log all from any to any in via $pif

# Everything else is denied by default
# deny and log all packets that fell through to see what they are
$cmd 00999 deny log all from any to any
################## End of IPFW rules file ###################################
```

31.6.4.7. 一个 NAT 和□状□□□集的例子

要使用 IPFW 的 NAT 功能， □需要□行一些□外的配置。 除了其他 IPFIREWALL □句之外， □需要在内核□□配置中加上 `option IPDIVERT` □句。

在 /etc/rc.conf 中， 除了普通的 IPFW 配置之外， □需要加入：

```
natd_enable="YES"                   # Enable NATD function
natd_interface="rl0"                # interface name of public Internet NIC
natd_flags="-dynamic -m"            # -m = preserve port numbers if possible
```

将□状□□□与 `divert natd` □□ (网□地址□□) 会使□□集的□写□得非常□□。 `check-state` 的位置， 以及 `divert natd` □□将□得非常□□。 □□一来， 就不再有□□的□序□理□□流程了。 提供了一□新的□作□型， 称□ `skipto`。 要使用 `skipto` 命令， 就必□□□一个□□□行□号， 以□定 `skipto` □□号是□希望跳□到的位置。

下面□出了一些未加注□的例子来□明如何□写□□的□□， 用以□助□理解包□理□□集的□理□序。

□理流程从□□文件最上□的第一个□□□始□理， 并自□向下地□□□一个□□， 直到□到匹配的□□， 且数据包从防火□中放出□止。 □注意□□号 100 101， 450， 500， 以及 510 的位置非常重要。 □些□□控制□出和接收的包的地址□□□程， □□它□在 keep-state □□表中的□□□中就能□与内网的 LAN IP 地址□□。 □一个需要注意的是， 所有的 allow 和 deny □□都指定了包的方向 (也就是 outbound 或 inbound) 以及网□接口。 最后， □注意所有□出的会□□求都会□求 `skipto rule 500` 以完成网□地址□□。

下面以 LAN 用□使用 web □□器□□一个 web □面□例。 Web □面使用 80 来完成通□。 当包□入防火□□， □□ 100 并不匹配， 因□它是□出而不是收到的包。 它能□通□□□ 101， 因□□是第一个包， 因而它□没有□入□□状□保持表。 包最□到□□□ 125， 并匹配□□□。 最□， 它会通□接入 Internet 公网的网□□出。 □之前， 包的源地址仍然是内网 IP 地址。 一旦匹配□个□□， 就会触□□个□作。 `keep-state` □□会把□个□□□到 keep-state □□□□表中， 并□行所指定的□作。 □作是□到□□表中的信息的一部分。 在□个例子中， □个□作是 `skipto rule 500`。 □□ 500 NAT 包的 IP 地址， 并将其□出。 □□必牢□， □一□非常重要。 接下来， 数据包将到□目的地， 之后返回并从□□集的第一条□□□始□理。 □一次， 它将与□□ 100 匹配， 其目的 IP 地址将被映射回□□的内网 LAN IP 地址。 其后， 它会被 `check-state` □□□理， □而在□存会□表中□到□□□， 并□到 LAN。 数据包接下来□到了内网 LAN PC 上， 而后者□会□送从□程服□器□求下一段数据的新数据包。 □个包会再次由 `check-state` □□□□， 并□到□出的表□， 并□行其□□的□作， 即 `skipto 500`。 包跳□到□□ 500 并被 NAT 后□出。

在接收一□， 已□存在的会□的数据包会被 `check-state` □□自□地□理， 并□到 `divert nat` □□。 我□需要解决的□□是， 阻止所有的坏数据包， 而只允□授□的服□。 例如在防火□上□行了 Apache 服□， 而我□希望人□在□□ Internet 公网的同□， 也能□□□本地的 web 站点。 新的接入□始□求包将匹配□□ 100， 而 IP

地址□□防火□所在的服□器而映射到了 LAN IP。 此后， 包会匹配所有我□希望□□的那些令人生□的□西， 并最□匹配□□ 425。 一旦□生匹配， 会□生□件事。 数据包会被□到 keep-state □□表， 但此□， 所有来自那个源 IP 的会□□求的数量会被限制□ 2。 □一做法能□挫□□□指定端口上服□的 DoS 攻□。 □作同□指定了 `allow` 包□被□到 LAN 上。 包返回□， `check-state` □□会□□出包属于某一已□存在的会□交互， 并直接把它□到□□ 500 做 NAT，并□到□出接口。

示□□□集 #1:

```
#!/bin/sh
cmd="ipfw -q add"
skip="skipto 500"
pif=rl0
ks="keep-state"
good_tcpo="22,25,37,43,53,80,443,110,119"

ipfw -q -f flush

$cmd 002 allow all from any to any via xl0  # exclude LAN traffic
$cmd 003 allow all from any to any via lo0  # exclude loopback traffic

$cmd 100 divert natd ip from any to any in via $pif
$cmd 101 check-state

# Authorized outbound packets
$cmd 120 $skip udp from any to xx.168.240.2 53 out via $pif $ks
$cmd 121 $skip udp from any to xx.168.240.5 53 out via $pif $ks
$cmd 125 $skip tcp from any to any $good_tcpo out via $pif setup $ks
$cmd 130 $skip icmp from any to any out via $pif $ks
$cmd 135 $skip udp from any to any 123 out via $pif $ks

# Deny all inbound traffic from non-routable reserved address spaces
$cmd 300 deny all from 192.168.0.0/16  to any in via $pif  #RFC 1918 private IP
$cmd 301 deny all from 172.16.0.0/12   to any in via $pif  #RFC 1918 private IP
$cmd 302 deny all from 10.0.0.0/8      to any in via $pif  #RFC 1918 private IP
$cmd 303 deny all from 127.0.0.0/8     to any in via $pif  #loopback
$cmd 304 deny all from 0.0.0.0/8       to any in via $pif  #loopback
$cmd 305 deny all from 169.254.0.0/16  to any in via $pif  #DHCP auto-config
$cmd 306 deny all from 192.0.2.0/24    to any in via $pif  #reserved for docs
$cmd 307 deny all from 204.152.64.0/23 to any in via $pif  #Sun cluster
$cmd 308 deny all from 224.0.0.0/3     to any in via $pif  #Class D & E multicast

# Authorized inbound packets
$cmd 400 allow udp from xx.70.207.54 to any 68 in $ks
$cmd 420 allow tcp from any to me 80 in via $pif setup limit src-addr 1

$cmd 450 deny log ip from any to any

# This is skipto location for outbound stateful rules
$cmd 500 divert natd ip from any to any out via $pif
$cmd 510 allow ip from any to any

########################## end of rules  ###################
```

下面的□个□□集基本上和上面一□，　但使用了易于□□的□写方式，　并□出了相当多的注解，　以□助□□□少的 IPFW □□□写者更好地理解□些□□到底在做什□。

示□□□集 #2：

```
#!/bin/sh
################ Start of IPFW rules file ######################################
# Flush out the list before we begin.
ipfw -q -f flush

# Set rules command prefix
cmd="ipfw -q add"
skip="skipto 800"
pif="rl0"     # public interface name of NIC
              # facing the public Internet

########################################################################
# No restrictions on Inside LAN Interface for private network
# Change xl0 to your LAN NIC interface name
########################################################################
$cmd 005 allow all from any to any via xl0

########################################################################
# No restrictions on Loopback Interface
########################################################################
$cmd 010 allow all from any to any via lo0

########################################################################
# check if packet is inbound and nat address if it is
########################################################################
$cmd 014 divert natd ip from any to any in via $pif

########################################################################
# Allow the packet through if it has previous been added to the
# the "dynamic" rules table by a allow keep-state statement.
########################################################################
$cmd 015 check-state

########################################################################
# Interface facing Public Internet (Outbound Section)
# Check session start requests originating from behind the
# firewall on the private network or from this gateway server
# destined for the public Internet.
########################################################################

# Allow out access to my ISP's Domain name server.
# x.x.x.x must be the IP address of your ISP's DNS
# Dup these lines if your ISP has more than one DNS server
# Get the IP addresses from /etc/resolv.conf file
$cmd 020 $skip tcp from any to x.x.x.x 53 out via $pif setup keep-state

# Allow out access to my ISP's DHCP server for cable/DSL configurations.
$cmd 030 $skip udp from any to x.x.x.x 67 out via $pif keep-state

# Allow out non-secure standard www function
$cmd 040 $skip tcp from any to any 80 out via $pif setup keep-state
```

```
# Allow out secure www function https over TLS SSL
$cmd 050 $skip tcp from any to any 443 out via $pif setup keep-state

# Allow out send & get email function
$cmd 060 $skip tcp from any to any 25 out via $pif setup keep-state
$cmd 061 $skip tcp from any to any 110 out via $pif setup keep-state

# Allow out FreeBSD (make install & CVSUP) functions
# Basically give user root "GOD" privileges.
$cmd 070 $skip tcp from me to any out via $pif setup keep-state uid root

# Allow out ping
$cmd 080 $skip icmp from any to any out via $pif keep-state

# Allow out Time
$cmd 090 $skip tcp from any to any 37 out via $pif setup keep-state

# Allow out nntp news (i.e., news groups)
$cmd 100 $skip tcp from any to any 119 out via $pif setup keep-state

# Allow out secure FTP, Telnet, and SCP
# This function is using SSH (secure shell)
$cmd 110 $skip tcp from any to any 22 out via $pif setup keep-state

# Allow out whois
$cmd 120 $skip tcp from any to any 43 out via $pif setup keep-state

# Allow ntp time server
$cmd 130 $skip udp from any to any 123 out via $pif keep-state

################################################################
# Interface facing Public Internet (Inbound Section)
# Check packets originating from the public Internet
# destined for this gateway server or the private network.
################################################################

# Deny all inbound traffic from non-routable reserved address spaces
$cmd 300 deny all from 192.168.0.0/16  to any in via $pif  #RFC 1918 private IP
$cmd 301 deny all from 172.16.0.0/12   to any in via $pif  #RFC 1918 private IP
$cmd 302 deny all from 10.0.0.0/8      to any in via $pif  #RFC 1918 private IP
$cmd 303 deny all from 127.0.0.0/8     to any in via $pif  #loopback
$cmd 304 deny all from 0.0.0.0/8       to any in via $pif  #loopback
$cmd 305 deny all from 169.254.0.0/16  to any in via $pif  #DHCP auto-config
$cmd 306 deny all from 192.0.2.0/24    to any in via $pif  #reserved for docs
$cmd 307 deny all from 204.152.64.0/23 to any in via $pif  #Sun cluster
$cmd 308 deny all from 224.0.0.0/3     to any in via $pif  #Class D & E multicast

# Deny ident
$cmd 315 deny tcp from any to any 113 in via $pif
```

```
# Deny all Netbios service. 137=name, 138=datagram, 139=session
# Netbios is MS/Windows sharing services.
# Block MS/Windows hosts2 name server requests 81
$cmd 320 deny tcp from any to any 137 in via $pif
$cmd 321 deny tcp from any to any 138 in via $pif
$cmd 322 deny tcp from any to any 139 in via $pif
$cmd 323 deny tcp from any to any 81  in via $pif

# Deny any late arriving packets
$cmd 330 deny all from any to any frag in via $pif

# Deny ACK packets that did not match the dynamic rule table
$cmd 332 deny tcp from any to any established in via $pif

# Allow traffic in from ISP's DHCP server. This rule must contain
# the IP address of your ISP's DHCP server as it's the only
# authorized source to send this packet type.
# Only necessary for cable or DSL configurations.
# This rule is not needed for 'user ppp' type connection to
# the public Internet. This is the same IP address you captured
# and used in the outbound section.
$cmd 360 allow udp from x.x.x.x to any 68 in via $pif keep-state

# Allow in standard www function because I have Apache server
$cmd 370 allow tcp from any to me 80 in via $pif setup limit src-addr 2

# Allow in secure FTP, Telnet, and SCP from public Internet
$cmd 380 allow tcp from any to me 22 in via $pif setup limit src-addr 2

# Allow in non-secure Telnet session from public Internet
# labeled non-secure because ID & PW are passed over public
# Internet as clear text.
# Delete this sample group if you do not have telnet server enabled.
$cmd 390 allow tcp from any to me 23 in via $pif setup limit src-addr 2

# Reject & Log all unauthorized incoming connections from the public Internet
$cmd 400 deny log all from any to any in via $pif

# Reject & Log all unauthorized out going connections to the public Internet
$cmd 450 deny log all from any to any out via $pif

# This is skipto location for outbound stateful rules
$cmd 800 divert natd ip from any to any out via $pif
$cmd 801 allow ip from any to any

# Everything else is denied by default
# deny and log all packets that fell through to see what they are
$cmd 999 deny log all from any to any
################## End of IPFW rules file ###################################
```

Chapter 32. 高□网□

32.1. 概述

本章将就一系列与网□有□的高□□□□行□□。

□完□章，□将了解：

- □于网□和路由的基□知□。
- 如何配置 IEEE® 802.11 和 Bluetooth® □□。
- 如何用 FreeBSD 做网□。
- 如何□无□机上配置网□□□。
- 如何配置从网□ PXE □□一个 NFS 根文件系□。
- 如何配置网□地址□□ (NAT)。
- 如何使用 PLIP □接□台□算机。
- 如何在□行 FreeBSD 的□算机上配置 IPv6。
- 如何配置 ATM。
- 如何利用 CARP， FreeBSD 支持的 Common Address Redundancy Protocol (共用地址冗余□□)

在□□章之前， □□：

- 理解 /etc/rc 脚本的基本知□。
- 熟悉基本的网□□□。
- 了解如何配置和安装新的 FreeBSD 内核 (配置FreeBSD的内核)。
- 了解如何安装第三方□件 (安装□用程序. Packages 和 Ports)。

32.2. 网□和路由

要□网□上的□台□算机能□相互通□， 就必□有一□能□描述如何从一台□算机到□一台□算机的机制， □一机制称作路由□□*(routing)*。 "路由□" 是一□□先定□的地址： "目的地(destination)" 和 "网□(gateway)"。 □个地址□所表□的意□是， 通□ 网□ 能□完成与 目的地 的通信。 有三□□型的目的地址： □个主机、 子网、 以及 "默□"。 如果没有可用的其它路由， 就会使用 "默□路由"， 有□默□路由的内容， 将在□后的章□中□行□□。 网□也有三□□型： □个主机， 网□接口 (也叫 "□路 (links)") 和以太网硬件地址 (MAC 地址)。

32.2.1. □例

□了□明路由□□的各个部分， 首先来看看下面的例子。 □是 `netstat` 命令的□出：

```
% netstat -r
Routing tables

Destination      Gateway            Flags     Refs     Use     Netif Expire

default          outside-gw         UGSc       37      418      ppp0
localhost        localhost          UH          0      181       lo0
test0            0:e0:b5:36:cf:4f   UHLW        5    63288       ed0     77
10.20.30.255     link#1             UHLW        1     2421
example.com      link#1             UC          0        0
host1            0:e0:a8:37:8:1e    UHLW        3     4601       lo0
host2            0:e0:a8:37:8:1e    UHLW        0        5       lo0 =>
host2.example.com link#1             UC          0        0
224              link#1             UC          0        0
```

□□行□出了当前配置中的默□路由 (将在 下一□ 中□行介□) 和 localhost (本机) 路由。

□里的路由表中□出的用于 localhost 的接口 (Netif 列) 是 lo0, 也就是大家熟知的 "回□□□"。它表示所有以此□ "目的地" 的通信都留在本机, 而不通□ LAN □出, 因□□些流量最□会回到起点。

接着出□的是以 0:e0: □□的地址。□些是以太网硬件地址, 也称□ MAC 地址。FreeBSD 会自□□□在同一个以太网中的任何主机 (如 test0), 并□其新□一个路由, 并通□那个以太网接口 - ed0 直接与它通□ (□者注：那台主机)。与□□路由表相□的也有一个超□□ (Expire列), 当我□在指定□□内没有收到从那个主机□来的信息, □□就派上用□了。□□情况下, 到□个主机的路由就会被自□□除。□些主机被使用一□叫做RIP(路由信息□□—Routing Information Protocol)的机制所□□, □□机制利用基于"最短路径□□ (shortest path determination)"的□法□算出到本地主机的路由。

FreeBSD 也会□本地子网添加子网路由(10.20.30.255 是子网 10.20.30 的广播地址, 而 example.com 是□个子网相□的域名)。名称 link#1 代表主机上的第一□以太网□。□会□□, □于它□没有指定□外的接口。

□□个□(本地网□主机和本地子网)的路由是由守□□程 routed 自□配置的。如果它没有□行, 那就只有被静□定□(例如, 明□□入的) 的路由才存在了。

host1 行代表我□的主机, 它通□以太网地址来□□。因□我□是□送端, FreeBSD知道使用回□接口 (lo0) 而不是通□以太网接口来□行□送。

□个 host2 行是我□使用 ifconfig(8) □名 (□看□于以太网的那部分就会知道我□□什□□□做) □□生的一个□例。在 lo0 接口之后的 ⇒ 符号表明我□不□使用了回□ (因□□个地址也□及了本地主机), 而且明□指出它是个□名。□□路由只有在支持□名的主机上才能□□出来。所有本地网上的其它的主机□于□□路由只会□□□有 link#1。

最后一行 (目□子网224) 用于□理多播——它会覆□到其它的区域。

最后, □个路由的不同属性可以在 Flags 列中看到。下□是个□于□些□志和它□的含□的一个□表：

U	Up: 路由□于活□状□。
H	Host: 路由目□是□个主机。
G	Gateway: 所有□到目的地的网□□到□一□程系□上, 并由它决定最后□到□里。

S	Static: □个路由是手工配置的，不是由系□自□生成的。
C	Clone: 生成一个新的路由， 通□□个路由我□可以□接上□些机子。 □□□型的路由通常用于本地网□。
W	WasCloned: 指明一个路由——它是基于本地区域网□ (克隆) 路由自□配置的。
L	Link: 路由□及到了以太网硬件。

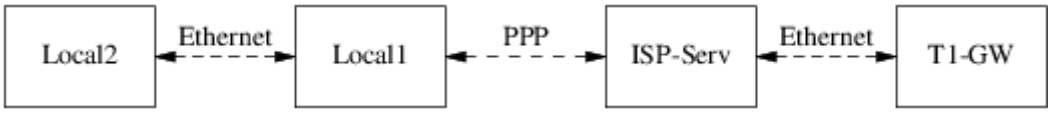
32.2.2. 默□路由

当本地系□需要与□程主机建立□接□， 它会□□路由表以决定是否有已知的路径存在。 如果□程主机属于一个我□已知如何到□ (克隆的路由) 的子网内，那□系□会□□看沿着那个接口是否能□□接。

如果所有已知路径都失□，系□□有最后一个□□： "默□"路由。□个路由是特殊□型的网□路由 (通常只有一个存在于系□里)，并且□是在□志□使用一个 c来□行□□。□于本地区域网□里的主机， □个网□被□置到任何与外界有直接□接的机子里 (无□是通□ PPP、DSL、cable modem、T1 或其它的网□接口□接)。

如果□正□某台本身就做□网□□接外界的机子配置默□路由的□， 那□□默□路由□□是□的"互□网服□商 (ISP)"那方的网□机子。

□我□来看一个□于默□路由的例子。□是个很普遍的配置：



主机 Local1 和 Local2 在□那□。Local1 通□ PPP □号□接到了 ISP。□个 PPP 服□器通□一个局域网□接到□一台网□机子——它又通□一个外部接口□接到 ISP 提供的互□网上。

□的□一台机子的默□路由□□是：

Host	Default Gateway	Interface
Local2	Local1	Ethernet
Local1	T1-GW	PPP

一个常□的□□是"我□□什□ (或□□) 能将 T1-GW □置成□ Local1 默□网□，而不是它所□接 ISP 服□器？"

□住，因□ PPP 接口使用的一个地址是在 ISP 的局域网里的，用于□那□的□接，□于 ISP 的局域网里的其它机子，其路由会自□□生。 因此，□就已□知道了如何到□机子 T1-GW， 那□也就没必要中□那一□了——□送通信□ ISP 服□器。

通常使用地址 X.X.X.1 做□一个局域网的网□。 因此 (使用相同的例子)，如果□本地的 C □地址空□是 10.20.30，而□的 ISP 使用的是 10.9.9， 那□默□路由表将是：

Host	Default Route
Local2 (10.20.30.2)	Local1 (10.20.30.1)
Local1 (10.20.30.1, 10.9.9.30)	T1-GW (10.9.9.1)

□可以很□易地通□ /etc/rc.conf 文件□定默□路由。在我□的□例里，在主机 Local2 里，我□在文件 /etc/rc.conf 里□加了下□内容：

```
defaultrouter="10.20.30.1"
```

也可以直接在命令行使用 route(8) 命令:

```
# route add default 10.20.30.1
```

要了解□于如何手工□□网□路由表的□一□□□， □参考 route(8) □机手册。

32.2.3. 重宿主机(Dual Homed Hosts)

□有一□其它的□型的配置是我□要提及的， □就是一个主机□于□个不同的网□。技□上，任何作□网□ (上□的□例中，使用了 PPP □接) 的机子就算作是重宿主机。 但□个□□□上□用来指那□□于□个局域网之中的机子。

有一□情形，一台机子有□个网□， □于各个子网都有各自的一个地址。□一□情况， □台机子□有一□网□，但使用 ifconfig(8) 做了□名。如果有□个独立的以太网在使用的情形就使用前者，如果只有一个物理网段，但□□上分成了□个独立的子网， 就使用后者。

□□情况都要□置路由表以便□子网都知道□台主机是到其它子网的网□——入站路由 (inbound route)。将一台主机配置成□个子网□的路由器， □□配置□常在我□需要□□□向或双向的包□□或防火□□被用到。

如果想□主机在□个接口□□□数据包，□需要激活 FreeBSD 的□□功能。至于□□做，□看下一部分了解更多。

32.2.4. 建立路由器

网□路由器只是一个将数据包从一个接口□□到□一个接口的系□。 互□网□准和良好的工程□践阻止了 FreeBSD □□在 FreeBSD 中把它置成默□□。□在可以在 rc.conf(5) 中改□下列□量的□□ YES，使□个功能生效：

```
gateway_enable="YES"          # Set to YES if this host will be a gateway
```

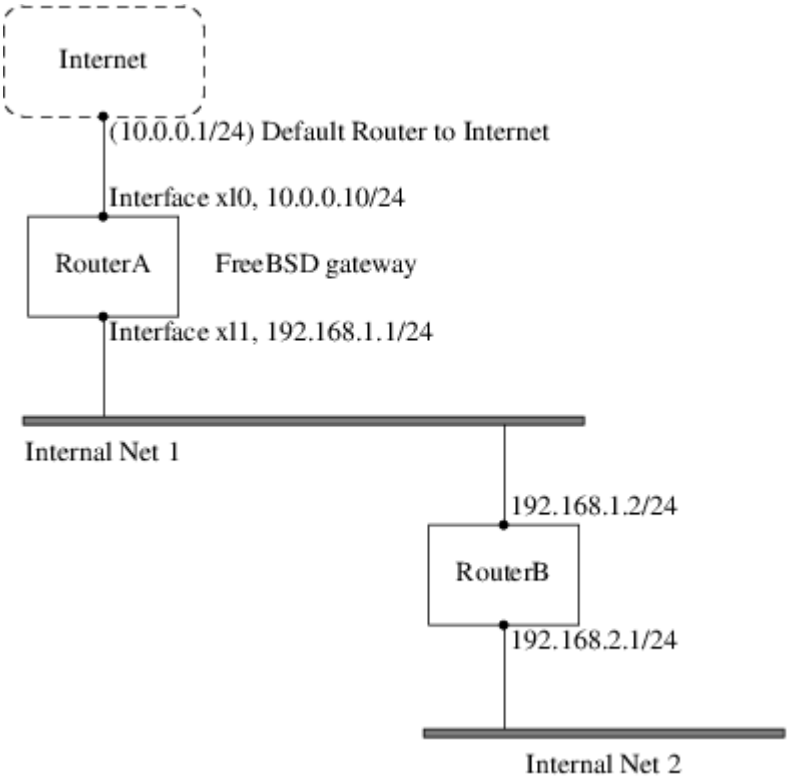
□个□□会把sysctl(8) □量——net.inet.ip.forwarding □置成 1。如果□要□□地停止路由， □可以把它重□□ 0。

新的路由器需要有路由才知道将数据□向何□。 如果网□□□□，□可以使用静□路由。FreeBSD 也自□一个□准的BSD路由□□守□□程 routed(8)， 称之□ RIP (version 1和 version 2) 和 IRDP。□ BGP v4，OSPF v2 和其它□□路由□□□□的支持可以从 net/zebra 包中得到。 像 GateD® 一□的商□□品也提供了更□□的网□路由解决方案。

32.2.5. □置静□路由

32.2.5.1. 手□配置

假□如下□□一个网□：



在□里，RouterA 是我□的 FreeBSD 机子，它充当□接到互□网其它部分的路由器的角色。 默□路由□置□10.0.0.1， 它就允□与外界□接。我□假定已□正□配置了 RouterB，并且知道如何□接到想去的任何地方。 (在□个□里很□□。只□在 RouterB 上□加默□路由，使用 192.168.1.1 做□网□。)

如果我□□看一下RouterA的路由表， 我□就会看到如下一些内容：

```
% netstat -nr
Routing tables

Internet:
Destination        Gateway            Flags    Refs      Use  Netif  Expire
default            10.0.0.1           UGS         0    49378    xl0
127.0.0.1          127.0.0.1          UH          0        6    lo0
10.0.0.0/24        link#1             UC          0        0    xl0
192.168.1.0/24     link#2             UC          0        0    xl1
```

使用当前的路由表，RouterA 是不能到□我□的内网——Internal Net 2 的。它没有到 192.168.2.0/24 的路由。 一□可以接受的方法是手工□加□条路由。以下的命令会把 Internal Net 2 网□加入到 RouterA 的路由表中，使用192.168.1.2 做□下一个跳□：

```
# route add -net 192.168.2.0/24 192.168.1.2
```

□在 RouterA 就可以到□ 192.168.2.0/24 网□上的任何主机了。

32.2.5.2. 永久配置

上面的□例□于□行着的系□来□配置静□路由是相当不□了。 只是，有一个□□——如果□重□□的 FreeBSD 机子，路由信息就会消失。 □理附加的静□路由的方法是把它放到□的 /etc/rc.conf 文件里去。

```
# Add Internal Net 2 as a static route
static_routes="internalnet2"
route_internalnet2="-net 192.168.2.0/24 192.168.1.2"
```

配置□量 static_routes 是一串以空格隔□的字符串。□一串表示一个路由名字。 在上面的例子中我□中有一个串在 static_routes 里。□个字符串中 *internalnet2*。 然后我□新□一个配置□量 route_internalnet2， □里我□把所有□□ route(8)命令的参数拿了□来。 在上面的□例中的我使用的命令是：

```
# route add -net 192.168.2.0/24 192.168.1.2
```

因此，我□需要的是 "-net 192.168.2.0/24 192.168.1.2"。

前□已□提到， 可以把多个静□路由的名称， 放到 static_routes 里□。 接着我□就来建立多个静□路由。下面几行所展示的， 是在一个假想的路由器上□加 192.168.0.0/24 和 192.168.1.0/24 之□静□路由的例子：

```
static_routes="net1 net2"
route_net1="-net 192.168.0.0/24 192.168.0.1"
route_net2="-net 192.168.1.0/24 192.168.1.1"
```

32.2.6. 路由□播

我□已□□□了如何定□通向外界的路由， 但未□及外界是如何□到我□的。

我□已□知道可以□置路由表， □□任何指向特定地址空□ (在我□的例子中是一个 C □子网) 的数据都会被送往网□上特定的主机， 然后由□台主机向地址空□内部□□数据。

当□得到一个分配□□的网□的地址空□□， ISP(网□服□商)会□置它□的路由表， □□指向□子网的数据就会通□ PPP □接下□到□的网□。 但是其它跨越国界的网□是如何知道将数据□□□的 ISP 的□？

有一个系□(很像分布式 DNS 信息系□)， 它一直跟踪被分配的地址空□， 并□明它□□接到互□网骨干(Internet backbone)的点。 "骨干(Backbone)" 指的是□□全世界和跨国的□□的主要干□。 □一台骨干主机(backbone machine)有一□主要表集的副本， 它将□送□特定网□的数据□向相□的骨干□体上(backbone carrier)， 从□点往下遍□服□提供商□，直到数据到□□的网□。

服□提供商的任□是向骨干网□广播，以声明它□就是通向□的网点的□接□点 (以及□入的路径)。□就是路由□播。

32.2.7. □□解答

有□候，路由□播会有一个□□，一些网□无法与□□接。 或□能□□□出路由是在□里中断的最有用的命令就是 traceroute(8)了。当□无法与□程主机□接□， □个命令一□有用(例如 ping(8) 失□)。

traceroute(8) 命令将以□想□接的主机的名字作□参数□行。 不管是到□了目□，□是因□没有□接而□止， 它都会□示所□□的所有网□主机。

想了解更多的信息，□看 traceroute(8) 的手册。

32.2.8. 多播路由

FreeBSD 一□始就支持多播□用□件和多播路由□□。 多播程序并不要求FreeBSD的任何特殊的配置，就可以工作得很好。多播路由需要支持被□□入内核：

```
options MROUTING
```

□外，多播路由守□□程——mrouted(8) 必□通□ /etc/mrouted.conf 配置来□□通道和 DVMRP。 更多□于多播路由配置的信息可以在 mrouted(8) 的手册里□到。

多播路由服□ mrouted(8) □□了 DVMRP 多播路由□□， 在□多采用多播的□合， 它已被 pim(4) 取代。 mrouted(8) 以及相□的 map-mbone(8) 和 mrinfo(8) 工具可以在 FreeBSD 的 Ports Collection net/mrouted 中□到。

32.3. 无□网□

32.3.1. 无□网□基□

□大多数无□网□都采用了 IEEE® 802.11 □准。 基本的无□网□中， 都包含多个以 2.4GHz 或 5GHz □段的无□□波广播的站点 (不□， 随所□地域的不同， 或者□了能□更好地□行通□， 具体的□率会在 2.3GHz 和 4.9GHz 的□□内□化)。

802.11 网□有□□□□方式： 在 *infrastructure* 模式 中， 一个通□站作□主站， 其他通□站都与其□□； □□网□称□ BSS， 而主站□成□无□□□点 (AP)。 在 BSS 中， 所有的通□都是通□ AP 来完成的； 即使通□站之□要相互通□，也必□将消息□□ AP。 在第二□形式的网□中， 并不存在主站， 通□站之□是直接通□的。 □□网□形式称作 IBSS，通常也叫做 *ad-hoc* 网□。

802.11 网□最初在 2.4GHz □段上部署， 并采用了由 IEEE® 802.11 和 802.11b □准所定□的□□。 □些□准定□了采用的操作□率、 包括分□和□□速率 (通□□程中可以使用不同的速率) 在内的 MAC □特性等。 □后的 802.11a □准定□了使用 5GHz □段□行操作， 以及不同的信号机制和更高的□□速率。 其后定□的 802.11g □准□用了在 2.4GHz 上如何使用 802.11a 信号和□□机制， 以提供□□早的 802.11b 网□的向前兼容。

802.11 网□中采用的各□底□□□机制提供了不同□型的安全机制。 最初的 802.11 □准定□了一□称□ WEP 的□□安全□□。 □个□□采用固定的□□布密□， 并使用 RC4 加密算法来□在网□上□□的数据□行□□。 全部通□站都必□采用同□的固定密□才能通□。 □一格局已□被□明很容易被攻破， 因此目前已□很少使用了， 采用□□方法只能□那些接入网□的用□迅速断□。 最新的安全□践是由 IEEE® 802.11i □准□出的， 它定□了新的加密算法， 并通□一□附加的□□来□通□站向无□□□点□□身□， 并交□用于□行数据通□的密□。 更□一□， 用于加密的密□会定期地刷新， 而且有机制能□□□入侵的□□ (并阻止□□□□)。 无□网□中□一□常用的安全□□□准是 WPA。 □是在 802.11i 之前由□界□□定□的一□□渡性□准。 WPA 定□了在 802.11i 中所□定的要求的子集， 并被□□用来在旧式硬件上□施。 特□地， WPA 要求只使用由最初 WEP 所采用的算法派生的 TKIP 加密算法。 802.11i □不但允□使用 TKIP， 而且□要求支持更□的加密算法 AES-CCM 来用于加密数据。 (在 WPA 中并没有要求使用 AES 加密算法， 因□在旧式硬件上□施□□算法□所需的□算□□性太高。)

除了前面介□的那些□□□准之外， □有一□需要介□的□准是 802.11e。 它定□了用于在 802.11 网□上□行多媒体□用， 如□□流和使用 IP □送的□音 (VoIP) 的□□。 与 802.11i □似， 802.11e 也有一个前身□准， 通常称作 WME (后改名□ WMM)， 它也是由□界□□定□的 802.11e 的子集， 以便能□在旧式硬件中使用多媒体□用。□于 802.11e 与 WME/WMM 之□的□一□重要区□是， 前者允□□流量通□服□品□ (QoS) □□和□□媒体□□□□来安排

□先□。 □于□些□□的正□□□， 能□□□高速突□数据和流量分□。

FreeBSD 支持采用 802.11a, 802.11b 和 802.11g 的网□。 □似地， 它也支持 WPA 和 802.11i 安全□□ (与 11a、 11b 和 11g 配合)， 而 WME/WMM 所需要的 QoS 和流量分□， □在部分无□□□上提供了支持。

32.3.2. 基本安装

32.3.2.1. 内核配置

要使用无□网□， □需要一□无□网□， 并□当地配置内核令其提供无□网□支持。 后者被分成了多个模□， 因此□只需配置使用□所需要的□件就可以了。

首先□需要的是一个无□□□。 最□常用的一□无□配件是 Atheros 生□的。 □些□□由 ath(4) □□程序提供支持，□需要把下面的配置加入到 /boot/loader.conf 文件中：

```
if_ath_load="YES"
```

Atheros □□分□三个部分： □□部分 (ath(4))、 用于□理芯片□有功能的支持□ (ath_hal(4))， 以及一□用以□□□□速率的算法 (ath_rate_sample here)。 当以模□方式加□□一支持□， 所需的其它模□会自□加□。 如果□使用的不是 Atheros □□， □□□□□的模□； 例如：

```
if_wi_load="YES"
```

表示使用基于 Intersil Prism □品的无□□□ (wi(4) □□)。

在□篇文□余下的部分中， 我□将以 ath(4) □来□行示□， 如果要套用□些配置的□，可能需要根据□□□的配置情况来修改示例中的□□名称。 在 FreeBSD 兼容硬件□明中提供了目前可用的无□网□□□， 以及兼容硬件的列表。 □□不同版本和硬件平台的□明可以在 FreeBSD 网站的 Release Information □面□到。 如果□的无□□□没有与之□□的 FreeBSD □用□□程序， 也可以□□使用 NDIS □□封装机制来直接使用 Windows® □□。

□于 FreeBSD 7.X， 在配置好□□□□之后， □□需要引入□□程序所需要的 802.11 网□支持。 □于 ath(4) □□而言，至少需要 wlan(4) wlan_scan_ap 和 wlan_scan_sta 模□； wlan(4) 模□会自□随无□□□□□一同加□， 剩下的模□必□要在系□引□□加□， 就需要在 /boot/loader.conf 中加入下面的配置：

```
wlan_scan_ap_load="YES"
wlan_scan_sta_load="YES"
```

从 FreeBSD 8.0 起， □些模□成□了 wlan(4) □□的基□□件， 并会随□配器□□一起□□加□。

除此之外， □□需要提供□希望使用的安全□□所需的加密支持模□。 □些模□是□□来□ wlan(4) 模□根据需要自□加□的， 但目前□必□手工□行配置。 □可以使用下面□些模□： wlan_wep(4)、 wlan_ccmp(4) 和 wlan_tkip(4)。 wlan_ccmp(4) 和 wlan_tkip(4) □□个□□都只有在□希望采用 WPA 和/或 802.11i 安全□□□才需要。 如果□的网□不采用加密， 就不需要 wlan_wep(4) 支持了。 要在系□引□□加□□些模□， 需要在 /boot/loader.conf 中加入下面的配置：

```
wlan_wep_load="YES"
wlan_ccmp_load="YES"
wlan_tkip_load="YES"
```

通□系□引□配置文件（也就是 /boot/loader.conf）中的□些信息生效， □必□重新□□□行 FreeBSD 的□算机。 如果不想立刻重新□□， 也可以使用 kldload(8) 来手工加□。

如果不想加□模□， 也可以将□些□□□□到内核中， 方法是在内核的 □□配置文件中加入下面的配置：

```
device wlan              # 802.11 support
device wlan_wep          # 802.11 WEP support
device wlan_ccmp         # 802.11 CCMP support
device wlan_tkip         # 802.11 TKIP support
device wlan_amrr         # AMRR transmit rate control algorithm
device ath               # Atheros pci/cardbus NIC's
device ath_hal           # pci/cardbus chip support
options AH_SUPPORT_AR5416 # enable AR5416 tx/rx descriptors
device ath_rate_sample   # SampleRate tx rate control for ath
```

使用 FreeBSD 7.X □， □需要配置下面□□行； FreeBSD 的其他版本不需要它□。

```
device wlan_scan_ap      # 802.11 AP mode scanning
device wlan_scan_sta     # 802.11 STA mode scanning
```

将□些信息写到内核□□配置文件中之后， □需要重新□□内核， 并重新□□□行 FreeBSD 的 □算机。

在系□□□之后， □会在引□□□出的信息中， □到□似下面□□的□于无□□□的信息：

```
ath0: <Atheros 5212> mem 0x88000000-0x8800ffff irq 11 at device 0.0 on cardbus1
ath0: [ITHREAD]
ath0: AR2413 mac 7.9 RF2413 phy 4.5
```

32.3.3. Infrastructure 模式

通常的情形中使用的是 infrastructure 模式或称 BSS 模式。 在□□模式中， 有一系列无□□□点接入了有□网 □。 □个无□网都会有自己的名字， □个名字称作网□的 SSID。 无□客□端都通□无□□□点来完成接入。

32.3.3.1. FreeBSD 客□机

32.3.3.1.1. 如何□□无□□□点

□可以通□使用 `ifconfig` 命令来□描网□。 由于系□需要在操作□程中切□不同的无□□率并探□可用的无□□□点， □□□求可能需要数分□才能完成。 只有超□用□才能□□□□□描：

```
# ifconfig wlan0 create wlandev ath0
# ifconfig wlan0 up scan
SSID/MESH ID    BSSID              CHAN RATE   S:N     INT CAPS
dlinkap         00:13:46:49:41:76   11   54M -90:96   100 EPS  WPA WME
freebsdap       00:11:95:c3:0d:ac    1   54M -83:96   100 EPS  WPA
```

在□始□描之前， 必□将网□接口□□ up。 后□的□描□求就不需要再将网□接口□□ up 了。

在 FreeBSD 7.X 中， 会直接□配器□□， 例如 ath0， 而不是 wlan0 □□。 因此□需要把前面的命令行改□：

```
# ifconfig ath0 up scan
```

在□□文□余下的部分中， □也需要注意 FreeBSD 7.X 上的□些差□， 并□命令行示例□行□似的改□。

□描会列出所□求到的所有 BSS/IBSS 网□列表。 除了网□的名字 SSID 之外， 我□□会看到 BSSID 即无□□□点的 MAC 地址。 而 CAPS 字段□□出了网□□型及其提供的功能， 其中包括：

表 *13.* 通□站功能代□

功能代□	含□
E	Extended Service Set (ESS)。 表示通□站是 infrastructure 网□ (相□于 IBSS/ad-hoc 网□) 的成□。
I	IBSS/ad-hoc 网□。 表示通□站是 ad-hoc 网□ (相□于 ESS 网□) 的成□。
P	私密。 在 BSS 中交□的全部数据□均需保□数据保密性。 □表示 BSS 需要通□站使用加密算法，例如 WEP、 TKIP 或 AES-CCMP 来加密/解密与其他通□站交□的数据□。
S	短前□□ (Short Preamble)。 表示网□采用的是短前□□ (由 802.11b High Rate/DSSS PHY 定□， 短前□□采用 56-位 同□字段， 而不是在□前□□模式中所采用的 128-位 字段)。
s	短□撞槽□□ (Short slot time)。 表示由于不存在旧式 (802.11b) 通□站， 802.11g 网□正使用短□撞槽□□。

要□示目前已知的网□， 可以使用下面的命令：

```
# ifconfig wlan0 list scan
```

□些信息可能会由无□□配器自□更新， 也可使用 scan 手□更新。 快取□存中的旧数据会自□□除， 因此除非

□行更多□描， □个列表会逐□□小。

32.3.3.1.2. 基本配置

在□一□中我□将展示一个□□的例子来介□如何□无□网□□配器在 FreeBSD 中以不加密的方式工作。 在□熟悉了□些概念之后， 我□□烈建□□在□□的使用中采用 WPA 来配置网□。

配置无□网□的□程可分□三个基本□□： □□无□□□点、 □□□的通□站身□， 以及配置 IP 地址。 下面的几□中将分□□地介□它□。

32.3.3.1.2.1. □□无□□□点

多数□候□系□以内建的探□方式□□无□□□点就可以了。 □是在□将网□接口置□ up 或在 /etc/rc.conf 中配置 IP 地址□的默□方式， 例如：

```
wlans_ath0="wlan0"
ifconfig_wlan0="DHCP"
```

如前面提到的那□， FreeBSD 7.X 只需要一行配置：

```
ifconfig_ath0="DHCP"
```

如果存在多个无□□□点， 而□希望从中□□具体的一个， □可以通□指定 SSID 来□□：

```
wlans_ath0="wlan0"
ifconfig_wlan0="ssid your_ssid_here DHCP"
```

在某些□境中， 多个□□点可能会使用同□的 SSID (通常， □□做的目的是□化漫游)， □□可能就需要与某个具体的□□□□了。 □□情况下， □□□指定无□□□点的 BSSID (□□可以不指定 SSID)：

```
wlans_ath0="wlan0"
ifconfig_wlan0="ssid your_ssid_here bssid xx:xx:xx:xx:xx:xx DHCP"
```

除此之外， □有一些其它的方法能□□束□□无□□□点的□□， 例如限制系□□描的□段， 等等。 如果□的无□网□支持多个□段， □□做可能会非常有用， 因□□描全部可用□段是一个十分耗□的□程。 要将操作限制在某个具体的□段， 可以使用 `mode` 参数； 例如：

```
wlans_ath0="wlan0"
ifconfig_wlan0="mode 11g ssid your_ssid_here DHCP"
```

就会□制□使用采用 2.4GHz 的 802.11g， □□在□描的□候， 就不会考□那些 5GHz 的□段了。 除此之外， □可以通□ `channel` 参数来将操作□定在特定□率， 以及通□ `chanlist` 参数来指定□描的□段列表。 □于□些参数的□一□信息， 可以在□机手册 ifconfig(8) 中□到。

32.3.3.1.2.2. □□身□

一旦□□定了无□□□点， □的通□站就需要完成身□□□， 以便□始□送和接收数据。 身□□□可以通□□多方式□行，最常用的一□方式称□□放式□□， 它允□任意通□站加入网□并相互通信。 □□□□方式只□在□第一次配置无□网□□行□□□使用。 其它的□□方式□需要在□行数据通□之前， 首先□行密□□商握手； □些方式要□使用□先分□的密□或密□， 要□是用更□□一些的后台服□， 如 RADIUS。 □大多数用□会使用默□的□放式□□， 而第二多的□是 WPA-PSK， 它也称□个人 WPA， 在 下面 的章□中将□行介□。

如果□使用 Apple® AirPort® Extreme 基站作□无□□□点， □可能需要同□在□端配置 WEP 共享密□□□。 □可以通□在 /etc/rc.conf 文件中□行□置， 或使用 wpa_supplicant(8) 程序来手工完成。 如果□只有一个 AirPort® 基站， □可以用□似下面的方法来配置：

```
wlans_ath0="wlan0"
ifconfig_wlan0="authmode shared wepmode on weptxkey 1 wepkey 01234567
DHCP"
```

一般而言， □尽量避免使用共享密□□□□□方法， 因□它以非常受限的方式使用 WEP 密□，使得攻□者能□很容易地破解密□。 如果必□使用 WEP (例如， □了兼容旧式的□□) 最好使用 WEP 配合 open □□方式。 □于 WEP 的更多□料□参□ WEP。

32.3.3.1.2.3. 通□ DHCP □取 IP 地址

在□□定了无□□□点， 并配置了□□参数之后， □必□□得 IP 地址才能真正□始通□。 多数□候， □会通□ DHCP 来□得无□ IP 地址。 要□到□个目的， 需要□□ /etc/rc.conf 并在配置中加入 DHCP：

```
wlans_ath0="wlan0"
ifconfig_wlan0="DHCP"
```

□在□已□完成了□用无□网□接口的全部准□工作了， 下面的操作将□用它：

```
# /etc/rc.d/netif start
```

一旦网□接口□始□行， 就可以使用 ifconfig 来□看网□接口 ath0 的状□了：

```
# ifconfig wlan0
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        ether 00:11:95:d5:43:62
        inet 192.168.1.100 netmask 0xffffff00 broadcast 192.168.1.255
        media: IEEE 802.11 Wireless Ethernet OFDM/54Mbps mode 11g
        status: associated
        ssid dlinkap channel 11 (2462 Mhz 11g) bssid 00:13:46:49:41:76
        country US ecm authmode OPEN privacy OFF txpower 21.5 bmiss 7
        scanvalid 60 bgscan bgscanintvl 300 bgscanidle 250 roam:rssi 7
        roam:rate 5 protmode CTS wme burst
```

□里的 status: associated 表示□已□□接到了无□网□ (在□个例子中， □个网□的名字是 dlinkap)。 bssid 00:13:46:49:41:76 是指□所用无□□□点的 MAC 地址； authmode OPEN 表示□通□的内容将将不加密。

32.3.3.1.2.4. 静□ IP 地址

如果无法从某个 DHCP 服□器□得 IP 地址， □可以配置一个静□ IP 地址， 方法是将前面的 DHCP □□字替□□地址信息。 □□必保持其他用于□接无□□□点的参数：

```
wlans_ath0="wlan0"
ifconfig_wlan0="inet 192.168.1.100 netmask 255.255.255.0 ssid your_ssid_here"
```

32.3.3.1.3. WPA

WPA (Wi-Fi 保□□□) 是一□与 802.11 网□配合使用的安全□□， 其目的是消除 WEP 中缺少身□□□能力的□□，以及一些其它的安全弱点。 WPA 采用了 802.1X □□□□， 并采用从多□与 WEP 不同的加密算法中□□一□来保□数据保密性。 WPA 支持的唯一一□加密算法是 TKIP (□□密□完整性□□)， TKIP 是一□□ WEP 所采用的基本 RC4 加密算法的□展， 除此之外□提供了□□□到的入侵的□□机制。 TKIP 被□□用来与旧式硬件一同工作，只需要□行部分□件修改； 它提供了一□改善安全性的折衷方案， 但仍有可能受到攻□。 WPA 也指定了 AES-CCMP 加密作□ TKIP 的替代品， 在可能□□向于使用□□加密； 表□□一□□的常用□□是 WPA2 (或 RSN)。

WPA 定□了□□和加密□□。 □□通常是使用□□方法之一来完成的： 通□ 802.1X 或□似 RADIUS □□的后端□□服□，或通□在通□站和无□□□点之□通□事先分□的密□来□行最小握手。 前一□通常称作企□ WPA，而后者通常也叫做个人 WPA。 因□多数人不会□无□网□配置 RADIUS 后端服□器， 因此 WPA-PSK 是在 WPA 中最□常□的一□。

□无□□接的控制和身□□□工作 (密□□商或通□服□器□□) 是通□ wpa_supplicant(8) 工具来完成的。 □个程序□行□需要一个配置文件， /etc/wpa_supplicant.conf。 □于□个文件的更多信息， □参考□机手册 wpa_supplicant.conf(5)。

32.3.3.1.3.1. WPA-PSK

WPA-PSK 也称作 个人-WPA， 它基于□先分□的密□ (PSK)， □个密□是根据作□无□网□上使用的主密□的密□生成的。 □表示□个无□用□都会使用同□的密□。 WPA-PSK 主要用于小型网□， 在□□网□中，通常不需要或没有□法架□□□服□器。

无□何□， 都□使用足□□， 且包括尽可能多字母和数字的□口令， 以免被猜出和/或攻□。

第一□是修改配置文件 /etc/wpa_supplicant.conf， 并在其中加入在□网□上使用的 SSID 和事先分□的密□：

```
network={
  ssid="freebsdap"
  psk="freebsdmall"
}
```

接下来， 在 /etc/rc.conf 中， 我□将指定无□□□的配置， 令其采用 WPA， 并通□ DHCP 来□取 IP 地址：

```
wlans_ath0="wlan0"
ifconfig_wlan0="WPA DHCP"
```

下面□用无□网□接口：

```
# /etc/rc.d/netif start
Starting wpa_supplicant.
DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 5
DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 6
DHCPOFFER from 192.168.0.1
DHCPREQUEST on wlan0 to 255.255.255.255 port 67
DHCPACK from 192.168.0.1
bound to 192.168.0.254 -- renewal in 300 seconds.
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.254 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet OFDM/36Mbps mode 11g
      status: associated
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode WPA2/802.11i privacy ON deftxkey UNDEF
      AES-CCM 3:128-bit txpower 21.5 bmiss 7 scanvalid 450 bgscan
      bgscanintvl 300 bgscanidle 250 roam:rssi 7 roam:rate 5 protmode CTS
      wme burst roaming MANUAL
```

除此之外， □也可以手□地使用 above 中那□ /etc/wpa_supplicant.conf 来配置， 方法是□行：

```
# wpa_supplicant -i wlan0 -c /etc/wpa_supplicant.conf
Trying to associate with 00:11:95:c3:0d:ac (SSID='freebsdap' freq=2412 MHz)
Associated with 00:11:95:c3:0d:ac
WPA: Key negotiation completed with 00:11:95:c3:0d:ac [PTK=CCMP GTK=CCMP]
CTRL-EVENT-CONNECTED - Connection to 00:11:95:c3:0d:ac completed (auth) [id=0 id_str=]
```

接下来的操作， 是□行 dhclient 命令来从 DHCP 服□器□取 IP：

```
# dhclient wlan0
DHCPREQUEST on wlan0 to 255.255.255.255 port 67
DHCPACK from 192.168.0.1
bound to 192.168.0.254 -- renewal in 300 seconds.
# ifconfig wlan0
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.254 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet OFDM/36Mbps mode 11g
      status: associated
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode WPA2/802.11i privacy ON deftxkey UNDEF
      AES-CCM 3:128-bit txpower 21.5 bmiss 7 scanvalid 450 bgscan
      bgscanintvl 300 bgscanidle 250 roam:rssi 7 roam:rate 5 protmode CTS
      wme burst roaming MANUAL
```

如果在 /etc/rc.conf 中把 ifconfig_wlan0 □置成了 DHCP (像 ifconfig_wlan0="DHCP" □□), 那□在 wpa_supplicant □上了无□接入点 (AP) 之后，□会自□□行 dhclient。

如果不打算使用 DHCP 或者 DHCP 不可用， □可以在 wpa_supplicant □通□站完成了身□□□之后， 指定静□ IP 地址：

```
# ifconfig wlan0 inet 192.168.0.100 netmask 255.255.255.0
# ifconfig wlan0
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.100 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet OFDM/36Mbps mode 11g
      status: associated
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode WPA2/802.11i privacy ON deftxkey UNDEF
      AES-CCM 3:128-bit txpower 21.5 bmiss 7 scanvalid 450 bgscan
      bgscanintvl 300 bgscanidle 250 roam:rssi 7 roam:rate 5 protmode CTS
      wme burst roaming MANUAL
```

如果没有使用 DHCP， □需要手工配置默□网□， 以及域名服□器：

```
# route add default your_default_router
# echo "nameserver your_DNS_server" >> /etc/resolv.conf
```

32.3.3.1.3.2. 使用 EAP-TLS 的 WPA

使用 WPA 的第二□方式是使用 802.1X 后端□□服□器。 在□个例子中， WPA 也称作 企□-WPA, 以便与安全性□差、 采用事先分□密□的 个人-WPA 区分□来。 在 企□-WPA 中， □□操作是采用 EAP 完成的 (可□展□□□□)。

EAP 并未附□加密方法。 因此□□者决定将 EAP 放在加密信道中□行□送。 目前有□多 EAP □□方法，

最常用的方法是 EAP-TLS、 EAP-TTLS 和 EAP-PEAP。

EAP-TLS (□ □□□安全 的 EAP) 是一□在无□世界中得到了广泛支持的□□□□, 因□它是 Wi-Fi □盟 核准的第一个 EAP 方法。 EAP-TLS 需要使用三个□□： CA □□ (在所有□算机上安装)、 用以向□□明服□器身□的服□器□□, 以及□个无□客□端用于□明身□的客□机□□。 在□□ EAP 方式中， □□服□器和无□客□端均通□自己的□□向□方□明身□， 它□均□□□方的□□是本机□的□□□□机□ (CA) □□的。

与之前介□的方法□似， 配置也是通□ /etc/wpa_supplicant.conf 来完成的：

```
network={
  ssid="freebsdap" ①
  proto=RSN  ②
  key_mgmt=WPA-EAP ③
  eap=TLS ④
  identity="loader" ⑤
  ca_cert="/etc/certs/cacert.pem" ⑥
  client_cert="/etc/certs/clientcert.pem" ⑦
  private_key="/etc/certs/clientkey.pem" ⑧
  private_key_passwd="freebsdmallclient" ⑨
}
```

① □个字段表示网□名 (SSID)。

② □里， 我□使用 RSN (IEEE® 802.11i) □□， 也就是 WPA2。

③ key_mgmt □行表示所用的密□管理□□。 在我□的例子中， 它是使用 EAP □□的 WPA： WPA-EAP。

④ □个字段中， 提到了我□的□接采用 EAP 方式。

⑤ identity 字段包含了 EAP 的□体串。

⑥ ca_cert 字段□出了 CA □□文件的路径名。 在□□服□器□□□， □个文件是必需的。

⑦ client_cert □行□出了客□机□□的路径名。 □□个无□客□端而言， □个□□都是在全网□□内唯一的。

⑧ private_key 字段是客□机□□私□文件的路径名。

⑨ private_key_passwd 字段是私□的口令字。

接着， 把下面的配置写入 /etc/rc.conf：

```
wlans_ath0="wlan0"
ifconfig_wlan0="WPA DHCP"
```

下一□是使用 rc.d 机制来□用网□接口：

```
# /etc/rc.d/netif start
Starting wpa_supplicant.
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 7
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 15
DHCPACK from 192.168.0.20
bound to 192.168.0.254 -- renewal in 300 seconds.
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.254 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet DS/11Mbps mode 11g
      status: associated
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode WPA2/802.11i privacy ON deftxkey UNDEF
      AES-CCM 3:128-bit txpower 21.5 bmiss 7 scanvalid 450 bgscan
      bgscanintvl 300 bgscanidle 250 roam:rssi 7 roam:rate 5 protmode CTS
      wme burst roaming MANUAL
```

如前面提到的那□， 也可以手工通□ wpa_supplicant 和 ifconfig 命令□到□似的目的。

32.3.3.1.3.3. 使用 EAP-TTLS 的 WPA

在使用 EAP-TLS □， 参与□□□程的服□器和客□机都需要□□， 而在使用 EAP-TTLS (□□□□安全隧道的 EAP) □， 客□机□□□是可□的。 □□方式与某些安全 web 站点更□接近， 即使□□者没有客□端□□， □些 web 服□器也能建立安全的 SSL 隧道。 EAP-TTLS 会使用加密的 TLS 隧道来□送□□信息。

□于它的配置， 同□是通□ /etc/wpa_supplicant.conf 文件来□行的：

```
network={
  ssid="freebsdap"
  proto=RSN
  key_mgmt=WPA-EAP
  eap=TTLS ①
  identity="test" ②
  password="test" ③
  ca_cert="/etc/certs/cacert.pem" ④
  phase2="auth=MD5" ⑤
}
```

① □个字段是我□的□接所采用的 EAP 方式。

② identity 字段中是在加密 TLS 隧道中用于 EAP □□的身□串。

③ password 字段中是用于 EAP □□的口令字。

④ ca_cert 字段□出了 CA □□文件的路径名。 在□□服□器□□□， □个文件是必需的。

⑤ □个字段中□出了加密 TLS 隧道中使用的□□方式。 在□个例子中， 我□使用的是□ MD5-加密口令 的 EAP。 "inner authentication" (□注：内部□定) 通常也叫 "phase2"。

□□必□把下面的配置写入 /etc/rc.conf：

```
wlans_ath0="wlan0"
ifconfig_wlan0="WPA DHCP"
```

下一□是□用网□接口：

```
# /etc/rc.d/netif start
Starting wpa_supplicant.
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 7
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 15
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 21
DHCPACK from 192.168.0.20
bound to 192.168.0.254 -- renewal in 300 seconds.
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.254 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet DS/11Mbps mode 11g
      status: associated
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode WPA2/802.11i privacy ON deftxkey UNDEF
      AES-CCM 3:128-bit txpower 21.5 bmiss 7 scanvalid 450 bgscan
      bgscanintvl 300 bgscanidle 250 roam:rssi 7 roam:rate 5 protmode CTS
      wme burst roaming MANUAL
```

32.3.3.1.3.4. 使用 EAP-PEAP 的 WPA

PEAPv0/EAP-MSCHAPv2 是最常□的 PEAP 方法。 此文□的以下部分将使用 PEAP 指代□些方法。

PEAP (受保□的 EAP) 被□□用以替代 EAP-TTLS， 并且是在 EAP-TLS 之后最□常用的 EAP □准。 □言之，如果□的网□中有多□不同的操作系□， PEAP 将是□次于 EAP-TLS 的支持最广的□准。

PEAP 与 EAP-TTLS 很像： 它使用服□器端□□， 通□在客□端与□□服□器之□建立加密的 TLS 隧道来向用□□□身□， □保□了□□信息的交□□程。 在安全方面， EAP-TTLS 与 PEAP 的区□是 PEAP 会以明文广播用□名，只有口令是通□加密 TLS 隧道□送的。 而 EAP-TTLS 在□送用□名和口令□， 都使用 TLS 隧道。

我□需要□□ /etc/wpa_supplicant.conf 文件， 并加入与 EAP-PEAP 有□的配置：

```
network={
  ssid="freebsdap"
  proto=RSN
  key_mgmt=WPA-EAP
  eap=PEAP ①
  identity="test" ②
  password="test" ③
  ca_cert="/etc/certs/cacert.pem" ④
  phase1="peaplabel=0" ⑤
  phase2="auth=MSCHAPV2" ⑥
}
```

① □个字段的内容是用于□接的 EAP 方式。

② identity 字段中是在加密 TLS 隧道中用于 EAP □□的身□串。

③ password 字段中是用于 EAP □□的口令字。

④ ca_cert 字段□出了 CA □□文件的路径名。 在□□服□器□□□， □个文件是必需的。

⑤ □个字段包含了第一□段□□ (TLS 隧道) 的参数。 随□使用的□□服□器的不同， □需要指定□□的□□。 多数□候， □□□□是 "客□端 EAP 加密"， □可以通□使用 peaplabel=0 来指定。 更多信息可以在□机手册 wpa_supplicant.conf(5) 中□到。

⑥ □个字段的内容是□□□□在加密的 TLS 隧道中使用的信息。 □ PEAP 而言， □是 auth=MSCHAPV2。

□□必□把下面的配置加入到 /etc/rc.conf：

```
wlans_ath0="wlan0"
ifconfig_wlan0="WPA DHCP"
```

下一□是□用网□接口：

```
# /etc/rc.d/netif start
Starting wpa_supplicant.
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 7
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 15
DHCPREQUEST on wlan0 to 255.255.255.255 port 67 interval 21
DHCPACK from 192.168.0.20
bound to 192.168.0.254 -- renewal in 300 seconds.
wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.254 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet DS/11Mbps mode 11g
      status: associated
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode WPA2/802.11i privacy ON deftxkey UNDEF
      AES-CCM 3:128-bit txpower 21.5 bmiss 7 scanvalid 450 bgscan
      bgscanintvl 300 bgscanidle 250 roam:rssi 7 roam:rate 5 protmode CTS
      wme burst roaming MANUAL
```

32.3.3.1.4. WEP

WEP (有□等效□□) 是最初 802.11 □准的一部分。 其中没有提供身□□□机制， 只提供了弱□□控制，而且很容易破解。

WEP 可以通□ `ifconfig` 配置：

```
# ifconfig wlan0 create wlandev ath0
# ifconfig wlan0 inet 192.168.1.100 netmask 255.255.255.0 \
        ssid my_net wepmode on weptxkey 3 wepkey 3:0x3456789012
```

- `weptxkey` 指明了使用□个 WEP 密□来□行数据□□。 □里我□使用第三个密□。 它必□与无□接入点的配置一致。 如果□不清楚□的无□接入点， □□用 `1` （就是□第一个密□）来□置□个□量。
- `wepkey` 用于□□ WEP 密□。 其格式□□ *index:key*， key 默□□ `1`; 如果需要□置的密□不是第一个，就必需指定 index 了。

□需要将 `0x3456789012` 改□在无□接入点上配置的那个。

我□建□□□□□机手册 ifconfig(8) 来了解□一□的信息。

`wpa_supplicant` 机制也可以用来配置□的无□网□使用 WEP。 前面的例子也可以通□在 /etc/wpa_supplicant.conf 中加入下述□置来□□：

```
network={
  ssid="my_net"
  key_mgmt=NONE
  wep_key3=3456789012
  wep_tx_keyidx=3
}
```

接着：

```
# wpa_supplicant -i wlan0 -c /etc/wpa_supplicant.conf
Trying to associate with 00:13:46:49:41:76 (SSID='dlinkap' freq=2437 MHz)
Associated with 00:13:46:49:41:76
```

32.3.4. Ad-hoc 模式

IBSS 模式， 也称□ ad-hoc 模式， 是□点□点□接□□的。 例如， 如果希望在□算机 `A` 和 `B` 之□建立 ad-hoc 网□，我□只需□□□个 IP 地址和一个 SSID 就可以了。

在□算机 `A` 上：

```
# ifconfig wlan0 create wlandev ath0 wlanmode adhoc
# ifconfig wlan0 inet 192.168.0.1 netmask 255.255.255.0 ssid freebsdap
# ifconfig wlan0
  wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
      ether 00:11:95:c3:0d:ac
      inet 192.168.0.1 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet autoselect mode 11g <adhoc>
      status: running
      ssid freebsdap channel 2 (2417 Mhz 11g) bssid 02:11:95:c3:0d:ac
      country US ecm authmode OPEN privacy OFF txpower 21.5 scanvalid 60
      protmode CTS wme burst
```

此□的 adhoc 参数表示无□网□接口□以 IBSS 模式□□。

此□， 在 B 上□□能□□□到 A 的存在了：

```
# ifconfig wlan0 create wlandev ath0 wlanmode adhoc
# ifconfig wlan0 up scan
  SSID/MESH ID    BSSID              CHAN RATE   S:N     INT CAPS
  freebsdap       02:11:95:c3:0d:ac    2   54M -64:-96  100 IS   WME
```

在□出中的 I 再次□□了 A 机是以 ad-hoc 模式□行的。 我□只需□ B 配置一不同的 IP 地址：

```
# ifconfig wlan0 inet 192.168.0.2 netmask 255.255.255.0 ssid freebsdap
# ifconfig wlan0
  wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.2 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet autoselect mode 11g <adhoc>
      status: running
      ssid freebsdap channel 2 (2417 Mhz 11g) bssid 02:11:95:c3:0d:ac
      country US ecm authmode OPEN privacy OFF txpower 21.5 scanvalid 60
      protmode CTS wme burst
```

□□， A 和 B 就可以交□信息了。

32.3.5. FreeBSD 基于主机的（无□）□□接入点

FreeBSD 可以作□一个（无□）□□接入点（AP）， □□可以不必再去□一个硬件 AP 或者使用 ad-hoc 模式的网□。 当□的 FreeBSD 机器作□网□□接到□外一个网□的□候将非常有用。

32.3.5.1. 基本配置

在把□的 FreeBSD 机器配置成一个 AP 以前， □首先需要先在内核配置好□□的无□网□的无□网□支持。 当然□□需要加上□想用的安全□□。想□得更□□的信息， □参□ 基本安装。

目前□不支持使用 Windows® □□和 NDIS □□包装的网□做□ AP 使用。只有 FreeBSD 原生的无□□□能□支持 AP 模式。

一旦装□了无□网□的支持， □就可以□□一下看看□的无□□□是否支持基于主机的无□□□接入模式 （通常也被称□ hostap 模式）：

```
# ifconfig wlan0 create wlandev ath0
# ifconfig wlan0 list caps
drivercaps=6f85edc1<STA,FF,TURBOP,IBSS,HOSTAP,AHDEMO,TXPMGT,SHSLOT,SHPREAMBLE,MONITOR,
MBSS,WPA1,WPA2,BURST,WME,WDS,BGSCAN,TXFRAG>
cryptocaps=1f<WEP,TKIP,AES,AES_CCM,TKIPMIC>
```

□段□出□示了网□所支持的各□功能； 其中的□□字 HOSTAP 表示□□网□可以作□无□网□接入点来使用。 此外， □里□会□出所支持的加密算法： WEP、 TKIP、 AES， 等等。 □些信息□于知道在□□接入点上使用何□安全□□非常重要。

只有□建网□□□□□能□配置无□□□是否以 hostap 模式□行， 如果之前已□存在了相□的□□， □需要首先将其□□：

```
# ifconfig wlan0 destroy
```

接着， 在配置其它参数前， 以正□的□□重新生成□□：

```
# ifconfig wlan0 create wlandev ath0 wlanmode hostap
# ifconfig wlan0 inet 192.168.0.1 netmask 255.255.255.0 ssid freebsdap mode 11g
channel 1
```

再次使用 ifconfig □□ wlan0 网□接口的状□：

```
# ifconfig wlan0
  wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
      ether 00:11:95:c3:0d:ac
      inet 192.168.0.1 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet autoselect mode 11g <hostap>
      status: running
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode OPEN privacy OFF txpower 21.5 scanvalid 60
      protmode CTS wme burst dtimperiod 1 -dfs
```

hostap 参数指定了接口以主机接入点的方式□行。

通□在 /etc/rc.conf 中加入下面的配置， 也可以在系□引□的□程中自□完成□于网□接口的配置：

```
wlans_ath0="wlan0"
create_args_wlan0="wlanmode hostap"
ifconfig_wlan0="inet 192.168.0.1 netmask 255.255.255.0 ssid freebsdap mode 11g channel
1"
```

32.3.5.2. 不使用□□或加密的（无□）□□接入点

尽管我□不推□□行一个不使用任何□□或加密的 AP， 但□是一个非常□□的□□ AP 是否正常工作的方法。 □□配置□于□□客□端□□也非常重要。

一旦 AP 被配置成了我□前面所展示的那□， 就可以在□外一台无□机器上初始化一次□描来□到□个 AP：

```
# ifconfig wlan0 create wlandev ath0
# ifconfig wlan0 up scan
SSID/MESH ID    BSSID              CHAN RATE   S:N     INT CAPS
freebsdap       00:11:95:c3:0d:ac    1   54M -66:-96  100 ES   WME
```

在客□机上能看到已□□接上了（无□）□□接入点：

```
# ifconfig wlan0 inet 192.168.0.2 netmask 255.255.255.0 ssid freebsdap
# ifconfig wlan0
  wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
      ether 00:11:95:d5:43:62
      inet 192.168.0.2 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet OFDM/54Mbps mode 11g
      status: associated
      ssid freebsdap channel 1 (2412 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode OPEN privacy OFF txpower 21.5 bmiss 7
      scanvalid 60 bgscan bgscanintvl 300 bgscanidle 250 roam:rssi 7
      roam:rate 5 protmode CTS wme burst
```

32.3.5.3. 使用 WPA 的（无□）□□接入点

□一段将注重介□在 FreeBSD （无□）□□接入点上配置使用 WPA 安全□□。 更多有□ WPA 和配置基于 WPA 无□客□端的□□ □参□ WPA。

hostapd 守□□程将被用于□理与客□端的□□和在□用 WPA （无□）□□接入点上的密□管理。

接下来，所有的配置操作都将在作□ AP 的 FreeBSD 机器上完成。 一旦 AP 能□正□的工作了，便把如下□行加入 /etc/rc.conf 使得 hostapd 能在机器□□的□候自□□行：

```
hostapd_enable="YES"
```

在配置 hostapd 以前， □□保□已□完成了基本配置中所介□的□□ 基本配置。

32.3.5.3.1. WPA-PSK

WPA-PSK 旨在□没有□□服□器的小型网□而□□的。

配置文件□ /etc/hostapd.conf file：

```
interface=wlan0 ①
debug=1 ②
ctrl_interface=/var/run/hostapd ③
ctrl_interface_group=wheel ④
ssid=freebsdap ⑤
wpa=1 ⑥
wpa_passphrase=freebsdmall ⑦
wpa_key_mgmt=WPA-PSK ⑧
wpa_pairwise=CCMP TKIP ⑨
```

① □一□□明了□□接入点所使用的无□接口。

② □一□□置了□行 hostapd □候□示相□信息的□□程度。 1 表示最小的□□。

③ ctrl_interface □□□出了 hostapd 存□与其他外部程序（比如 hostapd_cli(8)）通信的域套接口文件路径。□里使用了默□□。

④ ctrl_interface_group □行□置了允□□□控制界面文件的□属性 （□里我□使用了 wheel □）。

⑤ □一□是□置网□的名称。

⑥ wpa □□表示□用了 WPA 而且指明要使用何□ WPA □□□□。 □ 1 表示 AP 将使用 WPA-PSK。

⑦ wpa_passphrase □□包含用于 WPA □□的 ASCII 密□。

⑧ wpa_key_mgmt □行表明了我□所使用的密□管理□□。 在□个例子中是 WPA-PSK。

⑨ wpa_pairwise □□表示（无□）□□接入点所接受的加密算法。 在□个例子中，TKIP(WPA) 和 CCMP(WPA2) 密□都会被接受。 CCMP 密□是除 TKIP 外的□一□□□， CCMP 一般作□首□密□； □有在 CCMP 不能被使用的□境中□□ TKIP。

接下来的一□就是□行 hostapd：

```
# /etc/rc.d/hostapd forcestart
```

```
# ifconfig wlan0
  wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 2290
       inet 192.168.0.1 netmask 0xffffff00 broadcast 192.168.0.255
       inet6 fe80::211:95ff:fec3:dac%ath0 prefixlen 64 scopeid 0x4
       ether 00:11:95:c3:0d:ac
       media: IEEE 802.11 Wireless Ethernet autoselect mode 11g <hostap>
       status: associated
       ssid freebsdap channel 1 bssid 00:11:95:c3:0d:ac
       authmode WPA2/802.11i privacy MIXED deftxkey 2 TKIP 2:128-bit txpowmax 36
protmode CTS dtimperiod 1 bintval 100
```

□在客□端能□□接上□行的（无□）□□接入点了， 更多□□可以参□ WPA。 □看有□些客□□接上了 AP 可以□行命令 `ifconfig wlan0 list sta`。

32.3.5.4. 使用 WEP 的（无□）□□接入点

我□不推□使用 WEP 来□置一个（无□）□□接入点， 因□没有□□的机制并容易被破解。 一些□史□留下的无□网□□支持 WEP 作□安全□□， □些网□□允□搭建不含□□或 WEP □□的 AP。

在□置了正□的 SSID 和 IP 地址后，无□□□就可以□入 hostap 模式了：

```
# ifconfig wlan0 create wlandev ath0 wlanmode hostap
# ifconfig wlan0 inet 192.168.0.1 netmask 255.255.255.0 \
    ssid freebsdap wepmode on weptxkey 3 wepkey 3:0x3456789012 mode 11g
```

- `weptxkey` 表示□□中使用□一个 WEP 密□。 □个例子中用了第3把密□（□注意密□的□号从 1□始）。□个参数必□□置以用来加密数据。
- `wepkey` 表示□置所使用的 WEP 密□。 它□□符合 *index:key* □□的格式。 如果没有指定 index，那□默□□□ 1。 □就是□如果我□使用了除第一把以外的密□， 那□就需要指定 index。

再使用一次 `ifconfig` 命令□看 wlan0 接口的状□：

```
# ifconfig wlan0
  wlan0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
      ether 00:11:95:c3:0d:ac
      inet 192.168.0.1 netmask 0xffffff00 broadcast 192.168.0.255
      media: IEEE 802.11 Wireless Ethernet autoselect mode 11g <hostap>
      status: running
      ssid freebsdap channel 4 (2427 Mhz 11g) bssid 00:11:95:c3:0d:ac
      country US ecm authmode OPEN privacy ON deftxkey 3 wepkey 3:40-bit
      txpower 21.5 scanvalid 60 protmode CTS wme burst dtimperiod 1 -dfs
```

□在可以从□外一台无□机器上初始化一次□描来□到□个 AP 了：

```
# ifconfig wlan0 create wlandev ath0
# ifconfig wlan0 up scan
SSID            BSSID              CHAN RATE  S:N   INT CAPS
freebsdap       00:11:95:c3:0d:ac    1   54M 22:1   100 EPS
```

□在客□机能□使用正□的参数（密□等） □到并□上（无□）□□接入点了， 更多□□□参□WEP。

32.3.6. 同□使用有□和无□□接

一般而言， 有□网□的速度更快而且更可□， 而无□网□□提供更好的□活及机□性， 使用□□本的用□，往往会希望□合□者的□点， 并能□在□□□接之□无□切□。

在 FreeBSD 上可以将多个网□接口合并到一起， 并以 "故障□移" 的方式自□切□， 也就是□， □一□网□接口有一定的□先□序， 而操作系□在□路状□□生□化□□自□□行切□， 例如当同□存在有□和无□□接的□候

□先使用有□网□， 而当有□网□断□□， □自□切□到无□网□。

我□将在□后的 □路聚合与故障□移 中介□□路聚合和故障□移， 并在 有□网□和无□网□接口□的自□切□ 中□□□配置方式□行示□。

32.3.7. 故障排除

如果□在使用无□网□□遇到了麻□， 此□提供了一系列用以□助排除故障的□□。

- 如果□在列表中□不到无□□□点， □□□□没有将无□□□配置□使用有限的一□□段。
- 如果□无法□□到无□□□点， □□□□的通□站配置与无□□□点的配置一致。 □包括□□模式以及安全□□。 尽可能□化□的配置。 如果□正使用□似 WPA 或 WEP □□的安全□□， □将无□□□点配置□□放□□和不采用安全措施，并□□是否数据能□通□。
- 一旦□能□□□到无□□□点之后， 就可以使用□□的工具如 ping(8) 来□断安全配置了。

  wpa_supplicant 提供了□多□□支持； □□手工□行它， 在□□□指定 -dd □□， 并察看□出□果。

- 除此之外□有□多其它的底□□□工具。 □可以使用 /usr/src/tools/tools/net80211 中的 wlandebug 命令来□用 802.11 □□支持□的□□功能。 例如：

  ```
  # wlandebug -i ath0 +scan+auth+debug+assoc
    net.wlan.0.debug: 0 => 0xc80000<assoc,auth,scan>
  ```

  可以用来□用与□描无□□□点和 802.11 □□在安排通□□与握手有□的控制台信息。

  □有□多有用的□□信息是由 802.11 □□□的； wlanstats 工具可以□示□些信息。 □些□□数据能□指出由 802.11 □□□出来的□□。 □注意某些□□可能是由□□□□在 802.11 □之下□□出来的， 因此□些□□可能并不□示。 要□断与□□有□的□□， □需要参考□□□□程序的文□。

如果上述信息没能□助□□到具体的□□所在， □提交□□□告， 并在其中附上□些工具的□出。

32.4. □牙

32.4.1. □介

Bluetooth (□牙) 是一□无□技□， 用于建立□□□ 2.4GHZ，波□□ 10 米的私有网□。 网□一般是由便携式□□，比加手机 (cellular phone)， 掌上□□ (handhelds) 和膝上□□ (laptops)) 以 ad-hoc 形式□成。不象其它流行的无□技□——Wi-Fi，Bluetooth 提供了更高□的服□□面，像□ FTP 的文件服□、文件推送 (file pushing)、□音□送、串行□模□等等。

在 FreeBSD 里，□牙□ (Bluetooth stack) 通□使用 Netgraph 框架 (□看 netgraph(4)) 来的□□。大量的"Bluetooth USB dongle"由 ng_ubt(4) □□程序支持。 基于 Broadcom BCM2033 芯片□的 Bluetooth □□可以通□ ubtbcmfw(4) 和 ng_ubt(4) □□程序支持。 3Com Bluetooth PC □ 3CRWB60-A 由 ng_bt3c(4) □□程序支持。 基于 Serial 和 UART 的□牙□□由 sio(4)、ng_h4(4) 和 hcseriald(8)。本□介□ USB Bluetooth dongle 的使用。

32.4.2. □入□□

默□的 Bluetooth □□□□程序已存在于内核模□里。 接入□□前，□需要将□□程序加□入内核：

```
# kldload ng_ubt
```

如果系□□□□ Bluetooth □□已□存在于系□里， 那□从 /boot/loader.conf 里加□□个模□：

```
ng_ubt_load="YES"
```

□入USB dongle。控制台(console)(或syslog中)会出□□似如下的信息：

```
ubt0: vendor 0x0a12 product 0x0001, rev 1.10/5.25, addr 2
ubt0: Interface 0 endpoints: interrupt=0x81, bulk-in=0x82, bulk-out=0x2
ubt0: Interface 1 (alt.config 5) endpoints: isoc-in=0x83, isoc-out=0x3,
      wMaxPacketSize=49, nframes=6, buffer size=294
```

脚本 /etc/rc.d/bluetooth 是用来□□和停止 Bluetooth stack (□牙□)的。 最好在拔出□□前停止 stack(stack)，当然也不是非做不可。 □□ stack (□) □，会得到如下的□出：

```
# /etc/rc.d/bluetooth start ubt0
BD_ADDR: 00:02:72:00:d4:1a
Features: 0xff 0xff 0xf 00 00 00 00 00
<3-Slot> <5-Slot> <Encryption> <Slot offset>
<Timing accuracy> <Switch> <Hold mode> <Sniff mode>
<Park mode> <RSSI> <Channel quality> <SCO link>
<HV2 packets> <HV3 packets> <u-law log> <A-law log> <CVSD>
<Paging scheme> <Power control> <Transparent SCO data>
Max. ACL packet size: 192 bytes
Number of ACL packets: 8
Max. SCO packet size: 64 bytes
Number of SCO packets: 8
```

32.4.3. 主控制器接口 (HCI)

主控制器接口 (HCI) 提供了通向基□控制器和□接管理器的命令接口及□□硬件状□字和控制寄存器的通道。□个接口提供了□□□牙基□ (Bluetooth baseband) 功能的□一方式。 主机上的 HCI □与□牙硬件上的 HCI 固件交□数据和命令。 主控制器的□□□ (如物理□□) □□程序提供□个 HCI □交□信息的能力。

□□个□牙 (Bluetooth) □□□建一个 *hci* □型的 Netgraph □点。 HCI □点一般□接□牙□□的□□□点 (下行流) 和 L2CAP □点 (上行流)。 所有的HCI操作必□在 HCI □点上□行而不是□□□□□点。HCI □点的默□名是 "devicehci"。更多□□□参考 ng_hci(4) 的□机手册。

最常□的任□是□□在 RF proximity 中的□牙 (Bluetooth) □□。□个就叫做 □□*(inquiry)*。□□及 HCI 相□的操作可以由 hccontrol(8) 工具来完成。 以下的例子展示如何□出□□内的□牙□□。 在几秒□内□□□得到一□□□列表。 注意□程主机只有被置于 *discoverable(*可□□*)* 模式才能答□□□。

```
% hccontrol -n ubt0hci inquiry
Inquiry result, num_responses=1
Inquiry result #0
       BD_ADDR: 00:80:37:29:19:a4
       Page Scan Rep. Mode: 0x1
       Page Scan Period Mode: 00
       Page Scan Mode: 00
       Class: 52:02:04
       Clock offset: 0x78ef
Inquiry complete. Status: No error [00]
```

BD_ADDR 是□牙□□的特定地址， □似于网□的 MAC 地址。需要用此地址与某个□□□一□地通信。 可以□ BD_ADDR 分配由人可□的名字 (human readable name)。 文件 /etc/bluetooth/hosts 包含已知□牙主机的信息。 下面的例子展示如何□得分配□□程□□的可□名。

```
% hccontrol -n ubt0hci remote_name_request 00:80:37:29:19:a4
BD_ADDR: 00:80:37:29:19:a4
Name: Pav's T39
```

如果在□程□牙上□行□□，□会□□□的□算机是 "your.host.name (ubt0)"。 分配□本地□□的名字可随□改□。

□牙系□提供点□点□接 (只有□个□牙□□参与) 和点□多点□接。在点□多点□接中，□接由多个□牙□□共享。以下的例子展示如何取得本地□□的活□基□ (baseband) □接列表。

```
% hccontrol -n ubt0hci read_connection_list
Remote BD_ADDR    Handle Type Mode Role Encrypt Pending Queue State
00:80:37:29:19:a4     41  ACL    0 MAST    NONE       0     0 OPEN
```

connection handle(□接柄*)* 在需要□止基□□接□有用。注意：一般不需要手□完成。 □ (stack) 会自□□止不活□的基□□接。

```
# hccontrol -n ubt0hci disconnect 41
Connection handle: 41
Reason: Connection terminated by local host [0x16]
```

参考 hccontrol help □取完整的 HCI 命令列表。大部分 HCI 命令不需要超□用□□限。

32.4.4. □□□接控制和□配□□(L2CAP)

□□□接控制和□配□□ (L2CAP) □上□□□提供面向□接和无□接的数据服□， 并提供多□□功能和分割重□操作。L2CAP 充□上□□□和□用□件□□和接收最大□度□ 64K 的 L2CAP 数据包。

L2CAP 基于 通道*(channel)* 的概念。 通道 (Channel) 是位于基□ (baseband) □接之上的□□□接。□个通道以多□一的方式□定一个□一□□ (single protocol)。 多个通道可以□定同一个□□，但一个通道不可以□定多个□□。 □个在通道里接收到的 L2CAP 数据包被□到相□的上□□□。 多个通道可共享同一个基□□接。

□□个□牙 (Bluetooth) □□□建一个 *l2cap* □型的 Netgraph □点。 L2CAP □点一般□接 HCI □点(下行流)和□牙□□的□□□点(上行流)。 L2CAP □点的默□名是 "devicel2cap"。 更多□□□参考 ng_l2cap(4) 的□机手册。

一个有用的命令是 l2ping(8)， 它可以用来 ping 其它□□。 一些□牙□□可能不会返回所有□送□它□的数据，所以下例中的 `0 bytes` 是正常的。

```
# l2ping -a 00:80:37:29:19:a4
0 bytes from 0:80:37:29:19:a4 seq_no=0 time=48.633 ms result=0
0 bytes from 0:80:37:29:19:a4 seq_no=1 time=37.551 ms result=0
0 bytes from 0:80:37:29:19:a4 seq_no=2 time=28.324 ms result=0
0 bytes from 0:80:37:29:19:a4 seq_no=3 time=46.150 ms result=0
```

l2control(8) 工具用于在 L2CAP 上□行多□操作。 以下□个例子展示如何取得本地□□的□□□接 (通道) 和基□□接的列表：

```
% l2control -a 00:02:72:00:d4:1a read_channel_list
L2CAP channels:
Remote BD_ADDR     SCID/ DCID   PSM  IMTU/ OMTU State
00:07:e0:00:0b:ca    66/   64     3   132/  672 OPEN
% l2control -a 00:02:72:00:d4:1a read_connection_list
L2CAP connections:
Remote BD_ADDR    Handle Flags Pending State
00:07:e0:00:0b:ca     41 O           0 OPEN
```

□一个□断工具是 btsockstat(1)。 它完成与 netstat(1) □似的操作， 只是用了□牙网□相□的数据□□。 以下□个例子□示与 l2control(8) 相同的□□□接。

```
% btsockstat
Active L2CAP sockets
PCB      Recv-Q Send-Q Local address/PSM       Foreign address   CID   State
c2afe900      0      0 00:02:72:00:d4:1a/3     00:07:e0:00:0b:ca 66    OPEN
Active RFCOMM sessions
L2PCB    PCB      Flag MTU   Out-Q DLCs State
c2afe900 c2b53380 1    127   0     Yes  OPEN
Active RFCOMM sockets
PCB      Recv-Q Send-Q Local address     Foreign address   Chan DLCI State
c2e8bc80      0    250 00:02:72:00:d4:1a 00:07:e0:00:0b:ca 3    6    OPEN
```

32.4.5. RFCOMM □□

RFCOMM □□提供基于 L2CAP □□的串行端口模□。 □□□基于 ETSI TS 07.10 □准。RFCOMM 是一个□□的□□□□， 附加了摸□ 9 □ RS-232(EIATIA-232-E) 串行端口的定□。 RFCOMM □□最多支持 60 个并□□接 (RFCOMM通道)。

□了□□ RFCOMM， □行于不同□□上的□用程序建立起一条□于它□之□通信段的通信路径。 RFCOMM□□上□用于使用串行端口的□用□件。 通信段是一个□□到□一个□□的□牙□接 (直接□接)。

RFCOMM □心的只是直接□接□□之□的□接， 或在网□里一个□□与 modem 之□的□接。RFCOMM 能支持其它的配置， 比如在一端通□□牙无□技□通□而在□一端使用有□接口。

在FreeBSD，RFCOMM □□在□牙套接字□ (Bluetooth sockets layer) □□。

32.4.6. □□的□□(Pairing of Devices)

默□情况下，□牙通信是不需要□□的， 任何□□可与其它任何□□□□。一个□牙□□ (比如手机) 可以□□通□□□以提供某□特殊服□ (比如□号服□)。 □牙□□一般使用 *PIN□(PIN codes)*。 一个 PIN □是最□□ 16 个字符的 ASCII 字符串。 用□需要在□个□□中□入相同的PIN□。用□□入了 PIN □后， □个□□会生成一个 □接密匙*(link key)*。接着□接密□可以存□在□□或存□器中。 □接□□个□□会使用先前生成的□接密□。 以上介□的□程被称□□□*(pairing)*。 注意如果任何一方□失了□接密□，必□重新□行□□。

守□□程 hcsecd(8) □□□理所有□牙□□□求。 默□的配置文件是 /etc/bluetooth/hcsecd.conf。 下面的例子□示一个手机的 PIN □被□□□"1234"：

```
device {
        bdaddr  00:80:37:29:19:a4;
        name    "Pav's T39";
        key     nokey;
        pin     "1234";
      }
```

PIN □没有限制(除了□度)。有些□□ (例如□牙耳机) 会有一个□置的 PIN □。-d □□□制 hcsecd(8) 守□□程□于前台，因此很容易看清□生了什□。 □置□端□□准□接收□□ (pairing)，然后□□□牙□接到□端□□。 □端□□□□回□接收了□□并□求PIN□。□入与 hcsecd.conf 中一□的 PIN □。 □在□的个人□算机已□与□程□□□□了。 □外□也可以在□程□□上初始□点。

可以通□在 /etc/rc.conf 文件中□加下面的行， 以便□ hcsecd 在系□□□□自□□行：

```
hcsecd_enable="YES"
```

以下是□□的 hcsecd 服□□出□本：

```
hcsecd[16484]: Got Link_Key_Request event from 'ubt0hci', remote bdaddr
0:80:37:29:19:a4
hcsecd[16484]: Found matching entry, remote bdaddr 0:80:37:29:19:a4, name 'Pav's T39',
link key doesn't exist
hcsecd[16484]: Sending Link_Key_Negative_Reply to 'ubt0hci' for remote bdaddr
0:80:37:29:19:a4
hcsecd[16484]: Got PIN_Code_Request event from 'ubt0hci', remote bdaddr
0:80:37:29:19:a4
hcsecd[16484]: Found matching entry, remote bdaddr 0:80:37:29:19:a4, name 'Pav's T39',
PIN code exists
hcsecd[16484]: Sending PIN_Code_Reply to 'ubt0hci' for remote bdaddr 0:80:37:29:19:a4
```

32.4.7. 服□□□□□ (SDP)

服□□□□□ (SDP) 提供□客□端□件一□方法， 它能□□由服□器□件提供的服□及属性。 服□的属性包括所提供服□的□型或□□， 使用□服□所需要的机制或□□。

SDP 包括 SDP 服□器和 SDP 客□端之□的通信。 服□器□□一□服□□□列表，它介□服□器上服□的特性。 □个服□□□包含□于□个服□的信息。通□□出 SDP □求， 客□端会得到服□□□列表的信息。如果客□端 (或者客□端上的□用□件) 决定使用一个服□，□了使用□个服□它必□与服□提供都建立一个独立的□接。 SDP 提供了□□服□及其属性的机制，但它并不提供使用□些服□的机制。

一般地，SDP客□端按照服□的某□期望特征来搜索服□。 但是，即使没有任何□于由 SDP 服□端提供的服□的□□信息， 有□也能令人□意地□□它的服□□□里所描述的是□□服□□型。 □□□□所提供服□的□程称□ □□*(browsing)*。

□牙 SDP 服□端 sdpd(8) 和命令行客□端 sdpcontrol(8) 都包括在了□准的 FreeBSD 安装里。下面的例子展示如何□行 SDP □□□□。

```
% sdpcontrol -a 00:01:03:fc:6e:ec browse
Record Handle: 00000000
Service Class ID List:
        Service Discovery Server (0x1000)
Protocol Descriptor List:
        L2CAP (0x0100)
                Protocol specific parameter #1: u/int/uuid16 1
                Protocol specific parameter #2: u/int/uuid16 1

Record Handle: 0x00000001
Service Class ID List:
        Browse Group Descriptor (0x1001)

Record Handle: 0x00000002
Service Class ID List:
        LAN Access Using PPP (0x1102)
Protocol Descriptor List:
        L2CAP (0x0100)
        RFCOMM (0x0003)
                Protocol specific parameter #1: u/int8/bool 1
Bluetooth Profile Descriptor List:
        LAN Access Using PPP (0x1102) ver. 1.0
```

等等。注意□个服□有一个属性 (比如 RFCOMM 通道)列表。 根据服□□可能需要□一些属性做个注□。 有些"□牙□□ (Bluetooth implementation)"不支持服□□□， 可能会返回一个空列表。□□情况，可以搜索指定的服□。下面的例子展示如何搜索 OBEX Object Push (OPUSH) 服□：

```
% sdpcontrol -a 00:01:03:fc:6e:ec search OPUSH
```

要在 FreeBSD 里□□牙客□端提供服□，可以使用 sdpd(8) 服□。 □可以通□在 /etc/rc.conf 中加入下面的行：

```
sdpd_enable="YES"
```

然后用下面的命令来□□ sdpd 服□：

```
# /etc/rc.d/sdpd start
```

需要□□端提供□牙服□的本地的服□程序会使用本地 SDP □程注册服□。像□□的程序就有 rfcomm_pppd(8)。一旦□□它，就会使用本地 SDP □程注册□牙 LAN 服□。

使用本地 SDP □程注册的服□列表，可以通□本地控制通道□出 SDP □□□□□得：

```
# sdpcontrol -l browse
```

32.4.8. □号网□ (DUN) 和使用 PPP(LAN) □面的网□接入

□号网□ (DUN) 配置通常与 modem 和手机一起使用。 如下是□一配置所□及的内容：

- □算机使用手机或 modem 作□无□ modem 来□接□号因特网□入服□器， 或者使用其它的□号服□；
- □算机使用手机或 modem 接收数据□求。

使用 PPP(LAN) □面的网□接入常使用在如下情形：

- □个□牙□□的局域网□入；
- 多个□牙□□的局域网接入；
- PC 到 PC (使用基于串行□模□的 PPP 网□)。

在 FreeBSD 中，□个□面使用 ppp(8) 和 rfcomm_pppd(8) (一□封装器，可以将 RFCOMM □牙□接□□□ PPP 可操作的□西) 来□□。 在使用任何□面之前，一个新的 PPP □□必□在 /etc/ppp/ppp.conf 中建立。 想要□例□参考 rfcomm_pppd(8)。

在下面的例子中，rfcomm_pppd(8) 用来在 NUN RFCOMM 通道上打□一个到 BD_ADDR □ 00:80:37:29:19:a4 的□□的 RFCOMM □接。具体的 RFCOMM 通道号要通□ SDP 从□端□□□得。也可以手□指定通 RFCOMM，□□情况下 rfcomm_pppd(8) 将不能□行 SDP □□。使用 sdpcontrol(8) 来□□□端□□上的 RFCOMM 通道。

```
# rfcomm_pppd -a 00:80:37:29:19:a4 -c -C dun -l rfcomm-dialup
```

□了提供 PPP(LAN) 网□接入服□，必□□行 sdpd(8) 服□。一个新的 LAN 客□端条目必□在 /etc/ppp/ppp.conf 文件中建立。 想要□例□参考 rfcomm_pppd(8)。 最后，在有效地通道号上□始 RFCOMM PPP 服□。RFCOMM PPP 服□会使用本地 SDP □程自□注册□牙 LAN 服□。下面的例子展示如何□□ RFCOMM PPP 服□。

```
# rfcomm_pppd -s -C 7 -l rfcomm-server
```

32.4.9. OBEX □象推送 (OBEX Object Push - OPUSH) □面

OBEX□□被广泛地用于移□□□之□□□的文件□□。 它的主要用□是在□外□通信□域， 被用于□□本或手持□□之□的一般文件□□。

OBEX 服□器和客□端由第三方□件包 obexapp□□，它可以从 comms/obexapp port 安装。

OBEX 客□端用于向 OBEX 服□器推入或接出□象。 一个□像可以是(□个例子)商□□片或□会。 OBEX 客□能通□ SDP 从□程□□取得 RFCOMM 通道号。□可以通□指定服□名代替 RFCOMM 通道号来完成。支持的服□名是有：IrMC、FTRN 和 OPUSH。 也可以用数字来指定 RFCOMM 通道号。下面是一个 OBEX 会□的例子，一个□□信息□像从手机中被拉出， 一个新的□像被推入手机的目□。

```
% obexapp -a 00:80:37:29:19:a4 -C IrMC
obex> get telecom/devinfo.txt devinfo-t39.txt
Success, response: OK, Success (0x20)
obex> put new.vcf
Success, response: OK, Success (0x20)
obex> di
Success, response: OK, Success (0x20)
```

□了提供 OBEX 推入服□，sdpd(8) 必□□于□行状□。必□□建一个根目□用于存放所有□入的□象。 根文件□的默□路径是 /var/spool/obex。 最后，在有效的 RFCOMM 通道号上□始 OBEX 服□。OBEX 服□会使用 SDP □程自□注册 OBEX □象推送 (OBEX Object Push) 服□。 下面的例子展示如何□□ OBEX 服□。

```
# obexapp -s -C 10
```

32.4.10. 串口(SP)□面

串口(SP)□面允□□牙□□完成 RS232 (或□似) 串口□的□真。 □个□面所□及到情形是， 通□虚□串口使用□牙代替□□来□理以前的程序。

工具 rfcomm_sppd(1) 来□□串口□。 "Pseudo tty" 用来作□虚□的串口。 下面的例子展示如何□接□程□□的串口服□。 注意□不必指定 RFCOMM 通道——rfcomm_sppd(1) 能□通□ SDP 从□端□□那里□得。 如果□想代替它的□，可以在命令行里指定 RFCOMM 通道来□□：

```
# rfcomm_sppd -a 00:07:E0:00:0B:CA -t /dev/ttyp6
rfcomm_sppd[94692]: Starting on /dev/ttyp6...
```

一旦□接上，"pseudo tty"就可以充当串口了：

```
# cu -l ttyp6
```

32.4.11. □□解答

32.4.11.1. 不能□接□端□□

一些□老的□牙□□并不支持角色□□ (role switching)。默□情况下，FreeBSD 接受一个新的□接□， 它会□□□行角色□□并成□主控端 (master)。 不支持角色□□的□□将无法□接。 注意角色□□是在新□接建立□□行的，因此如果□程□□不支持角色□□，就不可能向它□出□求。 一个 HCI □□用来在本地端禁用角色□□。

```
# hccontrol -n ubt0hci write_node_role_switch 0
```

32.4.11.2. 如果有□， 能否知道到底正在□生什□？

可以。 需要借助第三方□件包 hcidump， 它可以通□ comms/hcidump port 来安装。 hcidump 工具和 tcpdump(1) 非常相像。 它可以用来□示□牙数据包的内容， 并将其□□到文件中。

32.5. □接

32.5.1. □介

有□， 会有需要将一个物理网□分成□个独立的网段， 而不是□建新的 IP 子网， 并将其通□路由器相□。 以□□方式□接□个网□的□□称□ "网□ (bridge)"。 有□个网□接口的 FreeBSD 系□可以作□网□来使用。

网□通□学□□个网□接口上的 MAC □地址 (以太网地址) 工作。 只当数据包的源地址和目□地址□于不同的网□□，网□才□行□□。

在很多方面，网□就像一个□有很少端口的以太网交□机。

32.5.2. □合□接的情况

□合使用网□的， 有□多□不同的情况。

32.5.2.1. 使多个网□相互□通

网□的基本操作是将□个或多个网段□接在一起。 由于各式各□的原因， 人□会希望使用一台真正的□算机，而不是网□□□来充任网□的角色， 常□的原因包括□□的限制、 需要□行防火□， 或□虚□机网□接口□接虚□网□。网□也可以将无□网□以 hostap 模式接入有□网□。

32.5.2.2. □□/数据整形防火□

使用防火□的常□情形是无需□行路由或网□地址□□的情况 (NAT)。

□例来□， 一家通□ DSL 或 ISDN □接到 ISP 的小公司， □有 13 个 ISP 分配的全局 IP 地址和 10 台 PC。 在□□情况下， 由于□分子网的□□， 采用路由来□□防火□会比□困□。

基于网□的防火□可以串接在 DSL/ISDN 路由器的后面， 而无需考□ IP □制的□□。

32.5.2.3. 网□□□

网□可以用于□接□个不同的网段， 并用于□□往返的以太网□。 □可以通□在网□接口上使用 bpf(4)/tcpdump(1)， 或通□将全部以太网□□制到□一个网□接口 (span 口) 来□□。

32.5.2.4. 2□ VPN

通□ IP □接的网□， 可以利用 EtherIP 隧道或基于 tap(4) 的解决方案， 如 OpenVPN 可以将□个以太网□接到一起。

32.5.2.5. 2□ 冗余

网□可以通□多条□路□接在一起， 并使用生成□□□ (Spanning Tree Protocol) 来阻止多余的通路。□使以太网能□正□工作， □个□□之□□□只有一条激活通路， 而生成□能□□□□路， 并将多余的□路置□阻断状□。当激活通路断□□， □□能□□算□外一□□， 并重新激活阻断的通路， 以恢□到网□各点的□通性。

32.5.3. 内核配置

□一□主要介□ if_bridge(4) 网□□□。 除此之外， □有一个基于 netgraph 的网□□□， 如欲了解□一□□□， □参□□机手册 ng_bridge(4)。

网□□□是一个内核模□， 并会随使用 ifconfig(8) □建网□接口□自□加□。 □也可以将 device if_bridge 加入到内核配置文件中， 以便将其静□□□□内核。

包□□可以通□使用了 pfil(9) 框架的任意一□防火□□件包来完成。 □些防火□可以以模□形式加□， 也可以静□□□□内核。

通□配合 altq(4) 和 dummynet(4)， 网□也可以用于流量控制。

32.5.4. □用网□

网□是通□接口□制来□建的。 □可以使用 ifconfig(8) 来□建网□接口， 如果内核不包括网□□□， □它会自□将其□入。

```
# ifconfig bridge create
bridge0
# ifconfig bridge0
bridge0: flags=8802<BROADCAST,SIMPLEX,MULTICAST> metric 0 mtu 1500
        ether 96:3d:4b:f1:79:7a
        id 00:00:00:00:00:00 priority 32768 hellotime 2 fwddelay 15
        maxage 20 holdcnt 6 proto rstp maxaddr 100 timeout 1200
        root id 00:00:00:00:00:00 priority 0 ifcost 0 port 0
```

如此就建立了一个网□接口， 并□其随机分配了以太网地址。 maxaddr 和 timeout 参数能□控制网□在□□表中保存多少个 MAC 地址， 以及表□中主机的□期□□。 其他参数控制生成□的□□方式。

将成□网□接口加入网□。 □了□网□能□□所有网□成□接口□□包， 网□接口和所有成□接口都需要□于□用状□：

```
# ifconfig bridge0 addm fxp0 addm fxp1 up
# ifconfig fxp0 up
# ifconfig fxp1 up
```

网□□在会在 fxp0 和 fxp1 之□□□以太网□。 等效的 /etc/rc.conf 配置如下， 如此配置将在系□□□□□建同□的网

□。

```
cloned_interfaces="bridge0"
ifconfig_bridge0="addm fxp0 addm fxp1 up"
ifconfig_fxp0="up"
ifconfig_fxp1="up"
```

如果网□主机需要 IP 地址， □□将其□在网□□□本身， 而不是某个成□□□上。 □可以通□静□□置或 DHCP 来完成：

```
# ifconfig bridge0 inet 192.168.0.1/24
```

除此之外， 也可以□网□接口指定 IPv6 地址。

32.5.5. 防火□

当□用包□□□， 通□网□的包可以分□在□入的网□接口、 网□接口和□出的网□接口上□行□□。 □些□段均可禁用。 当包的流向很重要□， 最好在成□接口而非网□接口上配置防火□。

网□上可以□行□多配置以决定非 IP 及 ARP 包能否通□， 以及通□ IPFW □□二□防火□。 □参□ if_bridge(4) □机手册以了解□一□的□□。

32.5.6. 生成□

网□□□□了快速生成□□□ (RSTP 或 802.1w)， 并与□早的生成□□□ (STP) 兼容。 生成□可以用来在网□拓□中□□并消除□路。 RSTP 提供了比□□ STP 更快的生成□覆□速度， □□□□会在相□的交□机之□交□信息， 以迅速□入□□状□， 并避免□生□路。 FreeBSD 支持以 RSTP 和 STP 模式□行， 而 RSTP 是默□模式。

使用 stp 命令可以在成□接口上□用生成□。 □包含 fxp0 和 fxp1 的网□， 可以用下列命令□用 STP：

```
# ifconfig bridge0 stp fxp0 stp fxp1
bridge0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        ether d6:cf:d5:a0:94:6d
        id 00:01:02:4b:d4:50 priority 32768 hellotime 2 fwddelay 15
        maxage 20 holdcnt 6 proto rstp maxaddr 100 timeout 1200
        root id 00:01:02:4b:d4:50 priority 32768 ifcost 0 port 0
        member: fxp0 flags=1c7<LEARNING,DISCOVER,STP,AUTOEDGE,PTP,AUTOPTP>
                port 3 priority 128 path cost 200000 proto rstp
                role designated state forwarding
        member: fxp1 flags=1c7<LEARNING,DISCOVER,STP,AUTOEDGE,PTP,AUTOPTP>
                port 4 priority 128 path cost 200000 proto rstp
                role designated state forwarding
```

网□的生成□ ID □ 00:01:02:4b:d4:50 而□先□□ 32768。 其中 root id 与生成□相同， 表示□是作□生成□根的网□。

□一个网□也□用了生成□：

```
bridge0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        ether 96:3d:4b:f1:79:7a
        id 00:13:d4:9a:06:7a priority 32768 hellotime 2 fwddelay 15
        maxage 20 holdcnt 6 proto rstp maxaddr 100 timeout 1200
        root id 00:01:02:4b:d4:50 priority 32768 ifcost 400000 port 4
        member: fxp0 flags=1c7<LEARNING,DISCOVER,STP,AUTOEDGE,PTP,AUTOPTP>
                port 4 priority 128 path cost 200000 proto rstp
                role root state forwarding
        member: fxp1 flags=1c7<LEARNING,DISCOVER,STP,AUTOEDGE,PTP,AUTOPTP>
                port 5 priority 128 path cost 200000 proto rstp
                role designated state forwarding
```

□里的 `root id 00:01:02:4b:d4:50 priority 32768 ifcost 400000 port 4` 表示根网□是前面的 `00:01:02:4b:d4:50`， 而从此网□出□的通路代价□ `400000`, 此通路到根网□是通□ `port 4` 即 fxp0 □接的。

32.5.7. 网□的高□用法

32.5.7.1. 重建流量流

网□支持□□模式， 在 bpf(4) □理之后会将包□□， 而不是□□□理或□□。 □可以用于将□个或多个接口上的□入□化□一个 bpf(4) 流。 在将□个独立的接口上的□□的 RX/TX 信号重整□一个□， □会非常有用。

如果希望将四个网□接口上的□入□成一个流：

```
# ifconfig bridge0 addm fxp0 addm fxp1 addm fxp2 addm fxp3 monitor up
# tcpdump -i bridge0
```

32.5.7.2. □像口 (Span port)

网□收到的□个以太网□都可以□到□像口上。 网□上的□像口数量没有限制， 如果一个接口已□被配置□□像口，□它就不能再作□网□的成□口来使用。 □□用法主要是□与网□□像口相□的□听机配合使用。

如果希望将所有□□到名□ fxp4 的接口上：

```
# ifconfig bridge0 span fxp4
```

32.5.7.3. □用接口 (Private interface)

□用接口不会□□流量到除□用接口之外的其他端口。 □些流量会无条件地阻断， 因此包括 ARP 在内的以太网□均不会被□□。 如果需要□□性地阻断流量， □□使用防火□。

32.5.7.4. 自学□接口 (Sticky Interfaces)

如果网□的成□接口□□□自学□， □□□学□的地址□一旦□入□□快取□存， 即被□□是静□□。 自学□□不会从快取□存中□期或替□掉， 即使地址在□一接口上出□也是如此。 □使得不必事先□布□□表， 也能根据学□□果得到静□□的有点， 但在□些网段被网□看到的客□机， 就不能漫游至□一网段了。

□一□用法是将网□与 VLAN 功能□用， □□客□网□会被隔□在一□， 而不会浪□ IP 地址空□。 考□ CustomerA 在 vlan100 上， 而 CustomerB □在 vlan101 上。 网□地址□ 192.168.0.1， 同□作□ internet 路由器使用。

```
# ifconfig bridge0 addm vlan100 sticky vlan100 addm vlan101 sticky vlan101
# ifconfig bridge0 inet 192.168.0.1/24
```

□台客□机均将 192.168.0.1 作□默□网□， 由于网□快取□存是自学□的， 因而它□无法□造 MAC 地址来截取其他客□机的网□流量。

在 VLAN 之□的通□可以通□□用接口 (或防火□) 来阻断：

```
# ifconfig bridge0 private vlan100 private vlan101
```

□□□些客□机就完全相互隔□了。 可以使用整个的 /24 地址空□， 而无需□分子网。

32.5.7.5. 地址限制

接口后的源 MAC 地址数量是可以控制的。 一旦到□了限制未知源地址的包将会被□□， 直至□有□存中的一□□期或被移除。

下面的例子是□置 CustomerA 在 vlan100 上可□接的以太网□□最大□□ 10。

```
# ifconfig bridge0 ifmaxaddr vlan100 10
```

32.5.7.6. SNMP 管理

网□接口和 STP 参数能□由 FreeBSD 基本系□的 SNMP 守□□程□行管理。□出的网□ MIB 符和 IETF □准，所以任何 SNMP 客□端或管理包都可以被用来接收数据。

在网□机器上从/etc/snmp.config 文件中去掉以下□行的注□ begemotSnmpdModulePath."bridge" = "/usr/lib/snmp_bridge.so" 并□□ bsnmpd 守□□程。 其他的配置□□□如 community names 和 access lists 可能也□也需要修改。 参□ bsnmpd(1) 和 snmp_bridge(3) □取更多信息。

以下的例子中使用了 Net-SNMP □件 (net-mgmt/net-snmp) 来□□一个网□，当然同□也能□使用port net-mgmt/bsnmptools。 在 SNMP 客□端 Net-SNMP 的配置文件 $HOME/.snmp/snmp.conf 中加入以下几行来□入网□的 MIB 定□：

```
mibdirs +/usr/shared/snmp/mibs
mibs +BRIDGE-MIB:RSTP-MIB:BEGEMOT-MIB:BEGEMOT-BRIDGE-MIB
```

通□ IETF BRIDGE-MIB(RFC4188) □□一个□独的网□

```
% snmpwalk -v 2c -c public bridge1.example.com mib-2.dot1dBridge
BRIDGE-MIB::dot1dBaseBridgeAddress.0 = STRING: 66:fb:9b:6e:5c:44
BRIDGE-MIB::dot1dBaseNumPorts.0 = INTEGER: 1 ports
BRIDGE-MIB::dot1dStpTimeSinceTopologyChange.0 = Timeticks: (189959) 0:31:39.59 centi-
seconds
BRIDGE-MIB::dot1dStpTopChanges.0 = Counter32: 2
BRIDGE-MIB::dot1dStpDesignatedRoot.0 = Hex-STRING: 80 00 00 01 02 4B D4 50
...
BRIDGE-MIB::dot1dStpPortState.3 = INTEGER: forwarding(5)
BRIDGE-MIB::dot1dStpPortEnable.3 = INTEGER: enabled(1)
BRIDGE-MIB::dot1dStpPortPathCost.3 = INTEGER: 200000
BRIDGE-MIB::dot1dStpPortDesignatedRoot.3 = Hex-STRING: 80 00 00 01 02 4B D4 50
BRIDGE-MIB::dot1dStpPortDesignatedCost.3 = INTEGER: 0
BRIDGE-MIB::dot1dStpPortDesignatedBridge.3 = Hex-STRING: 80 00 00 01 02 4B D4 50
BRIDGE-MIB::dot1dStpPortDesignatedPort.3 = Hex-STRING: 03 80
BRIDGE-MIB::dot1dStpPortForwardTransitions.3 = Counter32: 1
RSTP-MIB::dot1dStpVersion.0 = INTEGER: rstp(2)
```

dot1dStpTopChanges.0的□□2　意味着　STP　网□拓□改□了2次，拓□的改□表示1个或多个　网□中的□接改□或失效并且有一个新□生成。 dot1dStpTimeSinceTopologyChange.0 的□□能□□示□是何□改□的。

□□多个网□接口可以使用 private BEGEMOT-BRIDGE-MIB：

```
% snmpwalk -v 2c -c public bridge1.example.com
enterprises.fokus.begemot.begemotBridge
BEGEMOT-BRIDGE-MIB::begemotBridgeBaseName."bridge0" = STRING: bridge0
BEGEMOT-BRIDGE-MIB::begemotBridgeBaseName."bridge2" = STRING: bridge2
BEGEMOT-BRIDGE-MIB::begemotBridgeBaseAddress."bridge0" = STRING: e:ce:3b:5a:9e:13
BEGEMOT-BRIDGE-MIB::begemotBridgeBaseAddress."bridge2" = STRING: 12:5e:4d:74:d:fc
BEGEMOT-BRIDGE-MIB::begemotBridgeBaseNumPorts."bridge0" = INTEGER: 1
BEGEMOT-BRIDGE-MIB::begemotBridgeBaseNumPorts."bridge2" = INTEGER: 1
...
BEGEMOT-BRIDGE-MIB::begemotBridgeStpTimeSinceTopologyChange."bridge0" = Timeticks:
(116927) 0:19:29.27 centi-seconds
BEGEMOT-BRIDGE-MIB::begemotBridgeStpTimeSinceTopologyChange."bridge2" = Timeticks:
(82773) 0:13:47.73 centi-seconds
BEGEMOT-BRIDGE-MIB::begemotBridgeStpTopChanges."bridge0" = Counter32: 1
BEGEMOT-BRIDGE-MIB::begemotBridgeStpTopChanges."bridge2" = Counter32: 1
BEGEMOT-BRIDGE-MIB::begemotBridgeStpDesignatedRoot."bridge0" = Hex-STRING: 80 00 00 40
95 30 5E 31
BEGEMOT-BRIDGE-MIB::begemotBridgeStpDesignatedRoot."bridge2" = Hex-STRING: 80 00 00 50
8B B8 C6 A9
```

通□ mib-2.dot1dBridge 子□改□正在被□□的网□接口：

```
% snmpset -v 2c -c private bridge1.example.com
BEGEMOT-BRIDGE-MIB::begemotBridgeDefaultBridgeIf.0 s bridge2
```

32.6. □路聚合与故障□移

32.6.1. 介□

使用 lagg(4) 接口， 能□将多个网□接口聚合□一个虚□接口， 以提供容□和高速□接的能力。

32.6.2. □行模式

Failover (故障□移)

只通□主网口收□数据。 如果主网口不可用， □使用下一个激活的网口。 □在□里加入的第一个网口便会被□□主网口； 此后加入的其他网口， □会被□□故障□移的□用网口。 如果□生故障□移之后， 原先的网口又恢□了可用状□， □它仍会作□主网口使用。

Cisco® Fast EtherChannel®

Cisco® Fast EtherChannel® (FEC) 是一□静□配置， 并不□行□点□□商或交□以太网□来□控□路情况。 如果交□机支持 LACP， □□使用后者而非□□配置。

FEC 将□出流量在激活的网口之□以□□□散列信息□依据分拆， 并接收来自任意激活网口的入流量。 散列信息包含以太网源地址、 目的地址， 以及 (如果有的□) VLAN tag 和 IPv4/IPv6 源地址及目的地址信息。

LACP

支持 IEEE® 802.3ad □路聚合控制□□ (LACP) 和□□□□。 LACP 能□在□点与若干□路聚合□之□□商□路。 □一个□路聚合□ (LAG) 由一□相同速度、 以全双工模式□行的网口□成。 流量在 LAG 中的网口之□， 会以□速度最大的原□□行分□。 当物理□路□生□化□， □路聚合会迅速□□□□形成新的配置。

LACP 也是将□出流量在激活的网口之□以□□□散列信息□依据分拆， 并接收来自任意激活网口的入流量。 散列信息包含以太网源地址、 目的地址， 以及 (如果有的□) VLAN tag 和 IPv4/IPv6 源地址及目的地址信息。

Loadbalance (□□均衡)

□是 *FEC* 模式的□名。

Round-robin (□□)

将□出流量以□□方式在所有激活端口之□□度， 并从任意激活端口接收□入流量。 □□模式□反了以太网□排序□□， 因此□小心使用。

32.6.3. 例子

例 *40.* 与 *Cisco®* 交□机配合完成 *LACP* □路聚合

在□个例子中， 我□将 FreeBSD 的□个网口作□一个□□均衡和故障□移□路聚合□接到交□机上。 在此基□上， □可以□加更多的网口， 以提高吞吐量和故障容□能力。 由于以太网□路上□□点□的□序是□制性的，因此□个□点之□的□接速度， 会取决于一□网□的最大速度。 □□算法会尽量采用更多的信息，以便将不同的网□流量分□到不同的网□接口上， 并平衡不同网口的□□。

在 Cisco® 交□机上将 *FastEthernet0/1* 和 *FastEthernet0/2* □□个网口添加到 channel-group *1*：

```
interface FastEthernet0/1
 channel-group 1 mode active
 channel-protocol lacp
!
interface FastEthernet0/2
 channel-group 1 mode active
 channel-protocol lacp
```

使用 *fxp0* 和 *fxp1* □建 lagg(4) 接口， □用□个接口并配置 IP 地址 *10.0.0.3/24*：

```
# ifconfig fxp0 up
# ifconfig fxp1 up
# ifconfig lagg0 create
# ifconfig lagg0 up laggproto lacp laggport fxp0 laggport fxp1 10.0.0.3/24
```

用下面的命令□看接口状□：

```
# ifconfig lagg0
```

□□□ *ACTIVE* 的接口是激活据合□的部分， □表示它□已□完成了与□程交□机的□商， 同□， 流量将通□□些接口来收□。 在 ifconfig(8) 的□□□出中会□出 LAG 的□□。

```
lagg0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=8<VLAN_MTU>
        ether 00:05:5d:71:8d:b8
        media: Ethernet autoselect
        status: active
        laggproto lacp
        laggport: fxp1 flags=1c<ACTIVE,COLLECTING,DISTRIBUTING>
        laggport: fxp0 flags=1c<ACTIVE,COLLECTING,DISTRIBUTING>
```

如果需要□看交□机上的端口状□， □□使用 `show lacp neighbor` 命令：

```
switch# show lacp neighbor
Flags:  S - Device is requesting Slow LACPDUs
        F - Device is requesting Fast LACPDUs
        A - Device is in Active mode       P - Device is in Passive mode

Channel group 1 neighbors

Partner's information:

                LACP port                       Oper    Port     Port
Port      Flags   Priority  Dev ID         Age     Key     Number   State
Fa0/1     SA      32768     0005.5d71.8db8  29s    0x146   0x3      0x3D
Fa0/2     SA      32768     0005.5d71.8db8  29s    0x146   0x4      0x3D
```

如欲□看□一□的□情，□需要使用 `show lacp neighbor detail` 命令。

如果希望在系□重□□保持□些□置，□在 /etc/rc.conf 中□加如下配置：

```
ifconfig_fxp0="up"
ifconfig_fxp1="up"
cloned_interfaces="lagg0"
ifconfig_lagg0="laggproto lacp laggport fxp0 laggport fxp1 10.0.0.3/24"
```

例 *41.* 故障□移模式

故障□移模式中， 当首□□路□生□□□， 会自□切□到□用端口。 首先□用成□接口， 接着是配置 lagg(4) 接口， 其中， 使用 *fxp0* 作□首□接口， *fxp1* 作□□用接口， 并在整个接口上配置 IP 地址 *10.0.0.15/24*：

```
# ifconfig fxp0 up
# ifconfig fxp1 up
# ifconfig lagg0 create
# ifconfig lagg0 up laggproto failover laggport fxp0 laggport fxp1 10.0.0.15/24
```

□建成功之后， 接口状□会是□似下面□□， 主要的区□是 MAC 地址和□□名：

```
# ifconfig lagg0
lagg0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=8<VLAN_MTU>
        ether 00:05:5d:71:8d:b8
        inet 10.0.0.15 netmask 0xffffff00 broadcast 10.0.0.255
        media: Ethernet autoselect
        status: active
        laggproto failover
        laggport: fxp1 flags=0<>
        laggport: fxp0 flags=5<MASTER,ACTIVE>
```

系□将在 *fxp0* 上□行流量的收□。 如果 *fxp0* 的□接中断， □ *fxp1* 会自□成□激活□接。 如果主端口的□接恢□， □它又会成□激活□接。

如果希望在系□重□□保持□些□置， □在 /etc/rc.conf 中□加如下配置：

```
ifconfig_fxp0="up"
ifconfig_fxp1="up"
cloned_interfaces="lagg0"
ifconfig_lagg0="laggproto failover laggport fxp0 laggport fxp1 10.0.0.15/24"
```

例 42. 有□网□和无□网□接口□的自□切□

□于使用□□本的用□来□， 通常会希望使用无□网□接口作□□用接口， 以便在有□网□不可用□□□保持网□□接。 通□使用 lagg(4)， 我□可以只使用一个 IP 地址的情况下， □先使用性能和安全性都更好的有□网□， 同□保持通□无□网□□接来□□数据的能力。

要□□□□的目的， 就需要将用于□接无□网□的物理接口的 MAC 地址修改□与所配置的 lagg(4) 一致，后者是从主网□接口， 也就是有□网□接口， □承而来。

在□个配置中， 我□将□先使用有□网□接口 *bge0* 作□主网□接口， 而将无□网□接口 *wlan0* 作□□用网□接口。 □里的 *wlan0* 使用的物理□□是 *iwn0*， 我□需要将它的 MAC 地址修改□与有□网□接口一致。 □了□到□个目的首先要得到有□网□接口上的 MAC 地址：

```
# ifconfig bge0
bge0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
    options=19b<RXCSUM,TXCSUM,VLAN_MTU,VLAN_HWTAGGING,VLAN_HWCSUM,TSO4>
    ether 00:21:70:da:ae:37
    inet6 fe80::221:70ff:feda:ae37%bge0 prefixlen 64 scopeid 0x2
    nd6 options=29<PERFORMNUD,IFDISABLED,AUTO_LINKLOCAL>
    media: Ethernet autoselect (1000baseT <full-duplex>)
    status: active
```

□可能需要将 *bge0* 改□□系□上□□使用的接口， 并从□出□果中的 ether □行□出有□网□的 MAC 地址。接着是修改物理的无□网□接口， *iwn0*：

```
# ifconfig iwn0 ether 00:21:70:da:ae:37
```

□用无□网□接口， 但不在其上配置 IP 地址：

```
# ifconfig wlan0 create wlandev iwn0 ssid my_router up
```

□用 *bge0* 接口。 □建 lagg(4) 接口， 其中 *bge0* 作□主网□接口， 而以 *wlan0* 作□□□接口：

```
# ifconfig bge0 up
# ifconfig lagg0 create
# ifconfig lagg0 up laggproto failover laggport bge0 laggport wlan0
```

新□建的接口的状□如下， □系□上的 MAC 地址和□□名等可能会有所不同：

```
# ifconfig lagg0
lagg0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> metric 0 mtu 1500
        options=8<VLAN_MTU>
        ether 00:21:70:da:ae:37
        media: Ethernet autoselect
        status: active
        laggproto failover
        laggport: wlan0 flags=0<>
        laggport: bge0 flags=5<MASTER,ACTIVE>
```

接着用 DHCP 客□端来□取 IP 地址：

```
# dhclient lagg0
```

如果希望在系□重□□保持□些□置， □在 /etc/rc.conf 中□加如下配置：

```
ifconfig_bge0="up"
ifconfig_iwn0="ether 00:21:70:da:ae:37"
wlans_iwn0="wlan0"
ifconfig_wlan0="WPA"
cloned_interfaces="lagg0"
ifconfig_lagg0="laggproto failover laggport bge0 laggport wlan0 DHCP"
```

32.7. 无□操作

FreeBSD 主机可以从网□□□而无需本地磁□就可操作， 使用的是从 NFS 服□器装□的文件系□。 除了□准的配置文件，无需任何的系□修改。 很容易□置□□的系□因□所有必要的元素都很容易得到：

- 至少有□□可能的方法从网□加□内核：
  - PXE：Intel® 的先□□□行□境 (Preboot eXecution Environment) 系□是一□□活的引□ ROM 模式，□个 ROM 内建在一些网□或主板的中。□看 pxeboot(8) 以□取更多□□。
  - Etherboot port (net/etherboot) □生通□网□加□□内核的可 ROM 代□。□些代□可以□入网□上的 PROM 上，或从本地□□ (或硬□) □□器加□，或从□行着的 MS-DOS® 系□加□。它支持多□网□。
- 一个□板脚本 (/usr/shared/examples/diskless/clone_root) □化了□服□器上的工作站根文件系□的□建和□□。 □个脚本需要少量的自定□，但□能很快的熟悉它。
- /etc 存在□准的系□□□文件用于□□和支持无□的系□□□。
- 可以向 NFS 文件或本地磁□□行交□(如果需要的□)。

□置无□工作站有□多方法。 有很多相□的元素大部分可以自定□以□合本地情况。 以下将介□一个完整系□的安装，□□的是□□性和与□准 FreeBSD □□脚本的兼容。介□的系□有以下特性：

- 无□工作站使用一个共享的只□ / 文件系□和一个共享的只□/usr。

root 文件系□是一□□准的 FreeBSD 根文件系□ (一般是服□器的)，只是一些配置文件被特定于无□操作的配置文件覆□。

root 文件系□必□可写的部分被 md(4) 文件系□覆□。 任何的改写在重□后都会□失。

- 内核由 etherboot 或 PXE □送和加□， 有些情况可能会指定使用其中之一。

如上所述，□个系□是不安全的。 它□□□于网□的受保□区域并不被其它主机信任。

□部分所有的信息均在 5.2.1-RELEASE 上□□□。

32.7.1. 背景信息

□置无□工作站相□要□□而又易出□。 有□分析一些原因是很□的。例如：

- □□□□□在□行□可能□生不同的行□。
- 出□信息□常是加密了的或根本就没有。

在□里， □及到的一些背景知□□于可能出□的□□的解决是很有□助的。

要成功地引□系□□有些操作需要做。

- 机子需要□取初始的参数，如它的 IP 地址、□行文件、服□器名、根路径。□个可以使用 或 BOOTP □□来完成。 DHCP 是 BOOTP 的兼容□展， 并使用相同的端口和基本包格式。

  只使用 BOOTP 来配置系□也是可行的。 bootpd(8) 服□程序被包含在基本的 FreeBSD 系□里。

  不□，DHCP 相比 BOOTP 有几个好□ (更好的配置文件，使用 PXE 的可能性，以及□多其它并不直接相□的无□操作)， 接着我□会要描述一个 DHCP 配置， 可能的□会利用与使用 bootpd(8) 相同的例子。□个□板配置会使用ISC DHCP □件包 (3.0.1.r12 □行版安装在□□服□器上)。

- 机子需要□送一个或多个程序到本地内存。 TFTP 或 NFS 会被使用。□□TFTP □是 NFS 需要在几个地方的"□□□□"□□里□置。 通常的□□源是□文件名指定了□□的□□：TFTP 通常从服□器里的一个□一目□□送所有文件，并需要相□□个目□的文件名。 NFS 需要的是□□文件路径。

- 介于□□程序和内核之□的可能的部分需要被初始化并□行。 在□部分有几个重要的□量：

  - PXE 会装入 pxeboot(8)——它是 FreeBSD 第三□段装□器的修改版。 loader(8) 会□得□多参数用于系□□□， 并在□送控制之前把它□留在内核□境里。 在□□情况下，使用 GENERIC 内核就可能了。
  - Etherboot 会做很少的准□直接装□内核。 □要使用指定的□□建立 (build) 内核。

    PXE 和 Etherboot 工作得一□的好。 不□， 因□一般情况下内核希望 loader(8) 做了更多的事情，PXE 是推□的方法。

    如果□的 BIOS 和网□都支持 PXE， 就□□使用它。

- 最后，机子需要□□它的文件系□。 NFS 使用在所有的情况下。

□看 diskless(8) 手册□。

32.7.2. 安装□明

32.7.2.1. 配置使用ISC DHCP

ISC DHCP 服□器可以回□ BOOTP 和 DHCP 的□求。

ISC DHCP 4.2 并不属于基本系□。首先□需要安装 net/isc-dhcp42-server port 或相□的"包"。

一旦安装了 ISC DHCP， □需要一个配置文件才能□行 (通常名叫 /usr/local/etc/dhcpd.conf)。 □里有个注□□的例子，里□主机 margaux 使用 Etherboot， 而主机corbieres 使用 PXE：

```
default-lease-time 600;
max-lease-time 7200;
authoritative;

option domain-name "example.com";
option domain-name-servers 192.168.4.1;
option routers 192.168.4.1;

subnet 192.168.4.0 netmask 255.255.255.0 {
  use-host-decl-names on; ①
  option subnet-mask 255.255.255.0;
  option broadcast-address 192.168.4.255;

  host margaux {
    hardware ethernet 01:23:45:67:89:ab;
    fixed-address margaux.example.com;
    next-server 192.168.4.4; ②
    filename "/data/misc/kernel.diskless"; ③
    option root-path "192.168.4.4:/data/misc/diskless"; ④
  }
  host corbieres {
    hardware ethernet 00:02:b3:27:62:df;
    fixed-address corbieres.example.com;
    next-server 192.168.4.4;
    filename "pxeboot";
    option root-path "192.168.4.4:/data/misc/diskless";
  }
}
```

① □个□□告□ dhcpd □送host 里声明的用于无□主机的主机名的□。 □外可能会□加一个 option host-name margaux 到 host 声明里。

② next-server 正式指定 TFTP 或 NFS 服□用于□入装□器或内核文件 (默□使用的是相同的主机作□DHCP 服□器)。

③ filename 正式定□□□的文件——etherboot 或 PXE □□行下一□将装□它。 根据使用的□□方式，它必□要指定。 Etherboot 可以被□□来使用 NFS 或 TFTP。 FreeBSD port 默□配置了NFS。 PXE 使用 TFTP， □就是□什□在□里使用相□文件名 (□可能依□于 TFTP 服□器配置，不□会相当典型)。 同□，PXE 会装□ pxeboot， 而不是内核。□外有几个很有意思的可能，如从 FreeBSD CD-ROM 的 /boot 目□装□ pxeboot (因□ pxeboot(8) 能□装□ GENERIC 内核，□就使得可以使用 PXE 从□程的 CD-ROM 里□□)。

④ root-path □□定□到根 (root) 文件系□的路径，通常是 NFS 符号。当使用 PXE □，只要□不□用内核里的 BOOTP □□，可以不管主机的IP。NFS 服□器然后就如同 TFTP 一□。

32.7.2.2. 配置使用BOOTP

□里□跟的是一个等效的 bootpd 配置 (□少到一个客□端)。□个可以在 /etc/bootptab 里□到。

□注意：□了使用BOOTP，etherboot 必□使用非默□□□ NO_DHCP_SUPPORT 来□行□□，而且 PXE 需要 DHCP 。bootpd 的唯一可□的好□是它存在于基本系□中。

```
.def100:\
  :hn:ht=1:sa=192.168.4.4:vm=rfc1048:\
  :sm=255.255.255.0:\
  :ds=192.168.4.1:\
  :gw=192.168.4.1:\
  :hd="/tftpboot":\
  :bf="/kernel.diskless":\
  :rp="192.168.4.4:/data/misc/diskless":

margaux:ha=0123456789ab:tc=.def100
```

32.7.2.3. 使用Etherboot准□□□程序

Etherboot 的网站 包含有更多的文□ ——主要□准的是 Linux 系□，但无疑包含有有用的信息。如下列出的是□于在 FreeBSD 系□里使用 Etherboot。

首先□必□安装net/etherboot 包或 port。

□可以改□ Etherboot 的配置 (如使用 TFTP 来代替 NFS)， 方法是修改 Config 文件——在 Etherboot 源目□里。

□于我□的□置，我□要使用一□□□□。 □于其它的方法(PROM，或 MS-DOS®程序)， □参考 Etherboot 文□。

想要使用□□□□，先□入一□□到安装有 Etherboot 的机器的□□器里， 然后把当前路径改到 src 目□——在 Etherboot □下， 接着□入：

```
# gmake bin32/devicetype.fd0
```

devicetype 依□于无□工作站上的以太网□的□型。 参考在同一个目□下的 NIC 文件□□正□的 *devicetype*。

32.7.2.4. 使用PXE□□

默□地，pxeboot(8) 装□器通□ NFS 装□内核。它可以□□来使用 TFTP——通□在文件 /etc/make.conf 里指定 LOADER_TFTP_SUPPORT □□来代替。 □参□ /usr/shared/examples/etc/make.conf 里的注□ 了解如何配置。

除此之外□有□个未□明的 make.conf □□——它可能□于□置一系列控制台无□机器会有用：BOOT_PXELDR_PROBE_KEYBOARD和 BOOT_PXELDR_ALWAYS_SERIAL。

当机器□□里，要使用 PXE， 通常需要□□ Boot from network □□——在 BIOS □置里， 或者在 PC 初始化的□

候□入一个功能□ (function key)。

32.7.2.5. 配置 TFTP 和 NFS 服□器

如果□正在使用 PXE 或 Etherboot——配置使用了 TFTP，那□□需要在文件服□器上□用 tftpd：

1. 建立一个目□——从那里 tftpd 可以提供文件服□，如 /tftpboot。
2. 把□一行加入到 /etc/inetd.conf里：

```
tftp    dgram   udp wait    root    /usr/libexec/tftpd  tftpd -l -s /tftpboot
```

好像有一些版本的 PXE 需要 TCP 版本的 TFTP。在□□情况下，加入第二行，使用 `stream tcp` 来代替 `dgram udp`。

3. □ inetd 重□其配置文件。要正□□行□个命令，在 /etc/rc.conf 文件中必□加入 `inetd_enable="YES"`：

```
# /etc/rc.d/inetd restart
```

□可把 tftpboot 目□放到服□器上的什何地方。□定□个位置□置在 inetd.conf 和 dhcpd.conf 里。

在所有的情况下，□都需要□用 NFS， 并且 NFS 服□器上□出相□的文件系□。

1. 把□一行加入到/etc/rc.conf里：

```
nfs_server_enable="YES"
```

2. 通□往 /etc/exports 里加入下面几行(□整"□入点"列， 并且使用无□工作站的名字替□ *margaux corbieres*)， □出文件系□——无□根目□存在于此：

```
/data/misc -alldirs -ro margaux corbieres
```

3. □ mountd 重□它的配置文件。如果□真的需要□用第一□的 /etc/rc.conf 里 NFS， □可能就要重□系□了。

```
# /etc/rc.d/mountd restart
```

32.7.2.6. 建立无□内核

如果□在使用 Etherboot， □需要□无□客□端建立内核配置文件， 使用如下□□(除了常使用的外)：

```
options     BOOTP          # Use BOOTP to obtain IP address/hostname
options     BOOTP_NFSROOT  # NFS mount root filesystem using BOOTP info
```

□可能也想使用 BOOTP_NFSV3， BOOT_COMPAT 和 BOOTP_WIRED_TO (参考 NOTES 文件)。

□些名字具有□史性，并且有些有些□□， 因□它□□□上□用了内核里 (它可能□制限制 BOOTP 或 DHCP 的使用)，与 DHCP 和 BOOTP 的无□的□用。

□□内核(参考配置FreeBSD的内核)， 然后将它□制到 dhcpd.conf 里指定的地方。

当使用 PXE 里， 使用以上□□建立内核并不做□格要求(尽管建□□□做)。 □用它□会在内核□□□引起更多的 DHCP 提及□的□求，□来的小小的□□是在有些特殊情况下新□和由 pxeboot(8) 取回的□之□的不一致性。 使用它□的好□是主机名会被附□□置。否□，□就需要使用其它的方法来□置主机名，如在客□端指定的 rc.conf 文件里。

□了使□有 Etherboot 的内核可引□，就需要把□□提示 (device hint) □□□去。通常要在配置文件(□看 NOTES 配置注□文件) 里□置下列□□：

```
hints       "GENERIC.hints"
```

32.7.2.7. 准□根(root)文件系□

□需要□无□工作站建立根文件系□， 它就是 dhcpd.conf 里的 root-path 所指定的目□。

32.7.2.7.1. 使用 make world 来□制根文件系□

□□方法可以迅速安装一个□底干□的系□ (不□□是根文件系□) 到 DESTDIR。 □要做的就是□□地□行下面的脚本：

```
#!/bin/sh
export DESTDIR=/data/misc/diskless
mkdir -p ${DESTDIR}
cd /usr/src; make buildworld && make buildkernel
make installworld && make installkernel
cd /usr/src/etc; make distribution
```

一旦完成，□可能需要定制 /etc/rc.conf 和 /etc/fstab——根据□的需要放到 DESTDIR里。

32.7.2.8. 配置 swap(交□)

如果需要，位于服□器上的交□文件可以通□ NFS 来□□。

32.7.2.8.1. NFS 交□区

内核并不支持在引□□□用 NFS 交□区。 交□区必□通□□□脚本□用， 其□程是挂接一个可写的文件系□，并在其上□建并□用交□文件。 要建立尺寸合□的交□文件， 可以□□做：

```
# dd if=/dev/zero of=/path/to/swapfile bs=1k count=1 oseek=100000
```

要□用它，□□要把下面几行加到 rc.conf里：

```
swapfile=/path/to/swapfile
```

32.7.2.9. □□□□

32.7.2.9.1. □行□ /usr 是只□在

如果无□工作站是配置来支持 X， 那□□就必□□整 XDM 配置文件，因□它默□把□□信息写到 /usr。

32.7.2.9.2. 使用非 FreeBSD 服□器

当用作根文件系□的服□器□行的是不 FreeBSD，□□要在 FreeBSD 机器上建立根文件系□， 然后把它□制到它的目的地，使用的命令可以是 `tar` 或 `cpio`。

在□□情况下，有□□于 /dev 里的一些特殊的文件会有□□，原因就是不同的 "最大/最小"整数大小。 一□解决的方法就是从非 FreeBSD 服□里□出一个目□， 并把它□入 FreeBSD 到机子上， 并使用 devfs(5) 来□用□透明地分派□□□点。

32.8. 从 PXE □□一个 NFS 根文件系□

Intel® □□□□行□境 （PXE） 能□操作系□从网□□□。 通常由近代主板的 BIOS 提供 PXE 支持，它可以通□在 BIOS □置里□□从网□□□□□。 一个功能完整的 PXE 配置□需要正□地□置 DHCP 和 TFTP 服□。

当□算机□□的□候， 通□ DHCP □取□于 从 TFTP 得到引□加□器（boot loader）的信息。 在□算机接受此信息以后， 便通□ TFTP 下□并□行引□加□器。 □些□□于 □□□□行□境 (PXE) □□ 的 2.2.1 章□中。 在 FreeBSD 中， 在 PXE □程中□取的引□加□器□ /boot/pxeboot。 在 /boot/pxeboot □行之后， FreeBSD 的内核被加□， 接着是其他的 FreeBSD 相□引□部分依次被□行。 更多□于 FreeBSD □□□程的□□信息□参□ FreeBSD 引□□程。

32.8.1. 配置用于 NFS 根文件系□的 `chroot` □境

1. Choose a directory which will have a FreeBSD installation which will be NFS mountable. For example, a directory such as /b/tftpboot/FreeBSD/install can be used.

   □□一个可被用□ NFS 挂□并安装有 FreeBSD 的目□。 比如可以使用像 /b/tftpboot/FreeBSD/install □□的一个目□。

   ```
   # export NFSROOTDIR=/b/tftpboot/FreeBSD/install
   # mkdir -p ${NFSROOTDIR}
   ```

2. 使用如下的命令□□ NFS 服□ 配置NFS.

3. 将下面□行加入 /etc/exports 用以通□ NFS □出此目□：

   ```
   /b -ro -alldirs
   ```

4. 重起 NFS 服□：

   ```
   # /etc/rc.d/nfsd restart
   ```

5. 按照 □置 中□明的□□□用 inetd(8)。

6. 将如下□行加入到 /etc/inetd.conf：

   ```
   tftp dgram udp wait root /usr/libexec/tftpd tftpd -l -s /b/tftpboot
   ```

7. 重□ inetd：

   ```
   # /etc/rc.d/inetd restart
   ```

8. 重新□□ FreeBSD 内核和用□□：

   ```
   # cd /usr/src
   # make buildworld
   # make buildkernel
   ```

9. 把 FreeBSD 安装到 NFS 挂□目□：

   ```
   # make installworld DESTDIR=${NFSROOTDIR}
   # make installkernel DESTDIR=${NFSROOTDIR}
   # make distribution DESTDIR=${NFSROOTDIR}
   ```

10. □□ TFTP 服□是否能下□将从 PXE □取的引□加□器：

```
# tftp localhost
tftp> get FreeBSD/install/boot/pxeboot
Received 264951 bytes in 0.1 seconds
```

11. □□ ${NFSROOTDIR}/etc/fstab 并加入以下□行挂□ NFS 根文件系□：

```
# Device                                        Mountpoint    FSType    Options
Dump Pass
myhost.example.com:/b/tftpboot/FreeBSD/install       /         nfs       ro
0    0
```

用□的 NFS 服□器主机名或者 IP 地址替□ *myhost.example.com*。 在此例中， 根文件系□是以"只□"的方式挂□用来防止 NFS 客□端可能意外□除根文件系□上的文件。

12. □置 chroot(8) □境中的 root 密□。

```
# chroot ${NFSROOTDIR}
# passwd
```

此□□置从 PXE □□的客□机的 root 密□。

13. 允□ ssh root 登□从 PXE □□的客□机， □□ ${NFSROOTDIR}/etc/ssh/sshd_config 并□□ PermitRootLogin □□。 □于此□□的□明□参□ sshd_config(5)。

14. □ ${NFSROOTDIR} 的 chroot(8) □境做些其他的定制。 □可以是像使用 pkg_add(1) 安装二□制包，使用 vipw(8) 修改密□， 或者□□ amd.conf(8) 映射自□挂□等。例如：

```
# chroot ${NFSROOTDIR}
# pkg_add -r bash
```

32.8.2. 配置 /etc/rc.initdiskless 中用到的内存文件系□

如果□从一个 NFS 根卷□□， /etc/rc 如果□□到是从 NFS □□便会□行 /etc/rc.initdiskless 脚本。□□□此脚本中的注□部分以便了解到底□生了什□。 我□需要把 /etc 和 /var 做成内存文件系□的原因是□些目□需要能被写入， 但 NFS 根文件系□是只□的。

```
# chroot ${NFSROOTDIR}
# mkdir -p conf/base
# tar -c -v -f conf/base/etc.cpio.gz --format cpio --gzip etc
# tar -c -v -f conf/base/var.cpio.gz --format cpio --gzip var
```

当系□□□的□候， /etc 和 /var 内存文件系□就会被□建并挂□， cpio.gz 就会被□制□去。

32.8.3. 配置 DHCP 服□

PXE 需要配置一个 TFTP 服□器和一个 DHCP 服□器。 DHCP 服□并不要求与 TFTP 服□在同一台机器上，但是必□能□从□的网□□□到它。

1. 按照此文□□ 安装和配置 DHCP 服□器 方法安装 DHCP 服□。 □保 /etc/rc.conf 和 /usr/local/etc/dhcpd.conf 都配置正□。
2. 在 /usr/local/etc/dhcpd.conf 中配置 `next-server`， `filename`， `option root-path` □□指向□的 TFTP 服□器的 IP 地址， 以及 TFTP 上 /boot/pxeboot 文件的路径， 和 NFS 根文件系□的路径。□里一□ dhcpd.conf □例：

```
subnet 192.168.0.0 netmask 255.255.255.0 {
   range 192.168.0.2 192.168.0.3 ;
   option subnet-mask 255.255.255.0 ;
   option routers 192.168.0.1 ;
   option broadcast-address 192.168.0.255 ;
   option domain-name-server 192.168.35.35, 192.168.35.36 ;
   option domain-name "example.com";

   # IP address of TFTP server
   next-server 192.168.0.1 ;

   # path of boot loader obtained
   # via tftp
   filename "FreeBSD/install/boot/pxeboot" ;

   # pxeboot boot loader will try to NFS mount this directory for root FS
   option root-path "192.168.0.1:/b/tftpboot/FreeBSD/install/" ;

}
```

32.8.4. 配置 PXE 客□端与□□□接□□

1. 当客□端□□的□候， □入 BIOS 配置菜□。 □置 BIOS 从网□□□。 如果之前□所有的配置□□都正□的□，那□所有部分□□能 "正常工作"。
2. 使用 net/wireshark port □看 DHCP 和 TFTP 的网□流量来□□各□□□。
3. □保 pxeboot 能从 TFTP □取。 在□的 TFTP 服□器上□□ /var/log/xferlog 日志□保 pxeboot 被从正□的位置□取。 可以□□□□上面例子 dhcpd.conf 中所□置的：

   ```
   # tftp 192.168.0.1
   tftp> get FreeBSD/install/boot/pxeboot
   Received 264951 bytes in 0.1 seconds
   ```

   □□□ tftpd(8) 和 tftp(1)。 其中的 BUGS 列出了 TFTP 的一些限制。
4. □保根文件系□能□从 NFS 挂□。 可以□□□□上面例子 dhcpd.conf 中所□置的：

   ```
   # mount -t nfs 192.168.0.1:/b/tftpboot/FreeBSD/install /mnt
   ```
5. □□ src/sys/boot/i386/libi386/pxe.c 中的代□以了解 pxeboot 加□器如何□置□如 boot.nfsroot.server 和 boot.nfsroot.path 之□的□量。 □些□量被用在了 src/sys/nfsclient/nfs_diskless.c 的 NFS 无□根挂□代□中。
6. Read pxeboot(8) and loader(8).

32.9. ISDN

□于 ISDN 技□和硬件的一个好的□源是Dan Kegel 的 ISDN 主□。

一个快速□□的到 ISDN 的路□□如下：

- 如果□住在欧洲，□可能要□看一下 ISDN □部分。
- 如果□正□□首要地使用 ISDN 基于□号非□用□路□接到□有提供商的互□网， □可能要了解一下□端□配器。如果□更改提供商的□， □会□□□来最大的□活性、最小的麻□。
- 如果□□接了□个局域网 (LAN)，或使用了□用的 ISDN □□□接到互□网，□可能要考□□□□独的路由器/网□。

在决定□□□一□方案的□候，价格是个很□□的因素。 下面列有从不算□到最□的□□：

32.9.1. ISDN □

FreeBSD 的 ISDN 工具通□被□□ (passive card) □支持 DSS1/Q.931(或 Euro-ISDN) □准。 此外也支持一些 active card， 它□的固件也支持其它信号□□， □其中包括最先得到支持的 "Primary Rate (PRI) ISDN"□。

isdn4bsd □件允□□接到其它 ISDN 路由器，使用的是原始的 HDLC 上的 IP 或利用同□ PPP：使用□有 isppp (一个修改□的 sppp(4) □□程序)的 PPP 内核，或使用用□区 (userland) ppp(8)。通□使用 userland ppp(8)，□个或更多 ISDN 的 B 通道□□□得可能。 除了□多如 300 波特 (Baud) 的□ modem 一□的工具外， □可以□□□□□答机□用。

在 FreeBSD 里，正有更多的 PC ISDN 被支持； 告示在整个欧洲及世界的其它多地区可以成功使用。

被支持的主型 ISDN 主要是有 Infineon (以前的 Siemens) ISAC/HSCX/IPAC ISDN 芯片，外有有 Cologne (只有 ISA) 芯片的 ISDN 、有 Winbond W6692 芯片的 PCI 、一部分有 Tiger300/320/ISAC 芯片的以及有一些商家有的芯片的 (如 AVM FritzCard PCI V.1.0 和 the AVM FritzCard PnP)。

当前的支持的 ISDN 有 AVM B1 (ISA 和 PCI) BRI 和 AVM T1 PCI PRI 。

于 isdn4bsd 的文，看 isdn4bsd的主， 那里也有提示、勘表以及更多的文 (如 isdn4bsd手册)。

要是有趣加不同 ISDN 的支持，当前不支持的 ISDN PC 的支持或想 isdn4bsd 的性能，系 Hellmuth Michaelis <hm@FreeBSD.org>。

于安装、配置以及 isdn4bsd 故障排除的，可以利用 freebsd-isdn 件列表。

32.9.2. ISDN 端配器

端配器 (TA) 于 ISDN 就好比 modem 于常。

多 TA 使用准的 Hayes modem AT 命令集，并且可以降来代替 modem。

TA 基本的作同 modem 一，不同之是接和整个速度更比老 modem 更快。同 modem 的安装一，也需要配置 PPP。的串口速度已足高。

使用 TA 接互网提供商的主要好是可以做的 PPP。 由于 IP 地址空得越来越，多提供商都不再提供静 IP。多的独立的路由器是不支持 IP 分配的。

TA 完全依于在行的 PPP 程， 以完成它的功能和定的接。可以在 FreeBSD 机子里易地从使用 modem 升到 ISDN，要是已安装了 PPP 的。只是，在使用 PPP 程序所体到任何同也存在。

如果想要最大的定性，使用 PPP 内核，而不要使用 userland PPP。

下面的 TA 就可以同 FreeBSD 一起工作：

- Motorola BitSurfer 和 Bitsurfer Pro
- Adtran

大部分其它的 TA 也可能工作，TA 提供商他的品可以接受大部分的准 modem AT 命令集。

于外置 TA 的是：象 modem 要一，机子需要有一个好的串行。

想要更深入地理解串行以及和同串口的不同点， 就要 FreeBSD 串行硬件教程了。

TA 将准的 PC 串口 (同的) 限制到了 115.2 Kbs，即使有 128 Kbs 的接。 想要完全利用 ISDN 有能力到的 128 Kbs，就需要把 TA 移到同串行上。

当心被去一个内置的 TA 以及自可以避免同/。内置的 TA 只是地将一准 PC 串口芯片内建在里。 所做的些只是省去一根串行以及省去一个空的孔。

有 TA 的同至少和一个独立的路由器同一快地， 而且使用一个的 386 FreeBSD 盒它。

□□同□□/TA □是独立的路由器，是个要高度□慎的□□。 在□件列表里有些相□的□□。我□建□□去搜索一下□于完整□□的□□。

32.9.3. □独的 ISDN □/路由器

ISDN □或路由器根本就没有指定要 FreeBSD 或其它任何的操作系□。更多完整的□于路由和□接技□的描述，□参考网□指南的□籍。

□部分的内容里，路由器和□接□□个□□将会交替地使用。

随着 ISDN 路由器/□的价格下滑，□它□的□□也会□得越来越流行。 ISDN 路由器是一个小盒子，可以直接地接入□的本地以太网， 并且自我管理到其它□/路由器的□接。它有个内建的□件用于与通信——通□ PPP 和其它流行的□□。

路由器有比□准 TA 更快的吞吐量，因□它会使用完全同□的 ISDN □接。

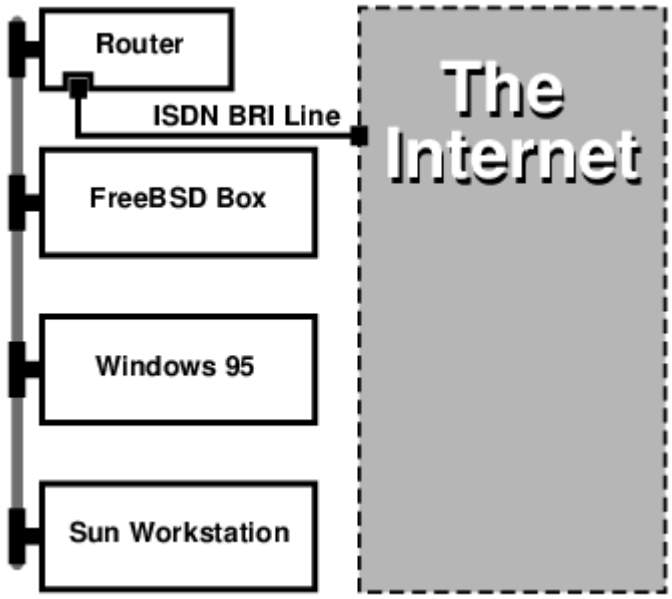
使用 ISDN 路由器和□的主要□□是□个生□商之□的□同性仍存在□□。 如果□□□□接到互□网提供商，□□□跟他□□行交□。

如果□□□□接□个局域网网段，如□的家庭网和□公网， □将是最□□最低□□的解决方案。因□□□的□□是用于□接□□的， 可以保□□□□接一定会成功。

例如□接到家里的□算机，或者是□公网里的一个分支□接到□公主网， 那□下面的□置就可能用到：

例 *43.* □公室局部或家庭网

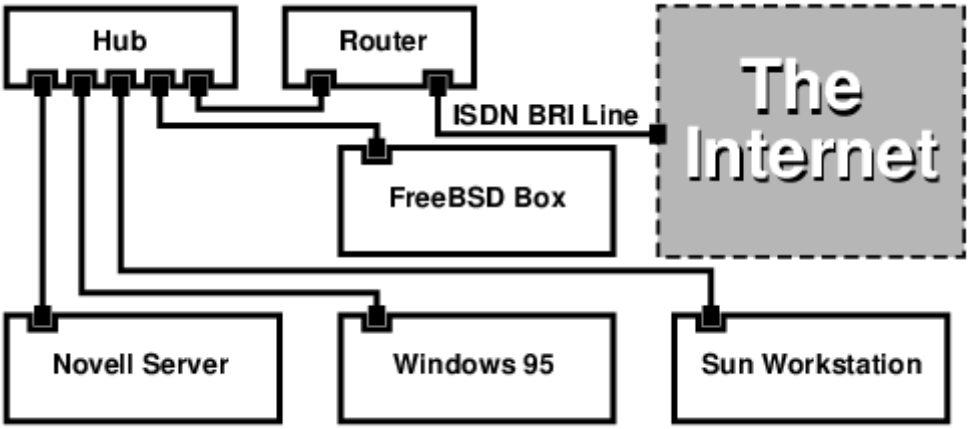
网□使用基于□□拓□的 10 base 2 以太网 ("瘦网(thinnet)")。如果有必要，用网□□接路由器和 AUI/10BT 收□器。



如果□的家里或□公室支部里只有一台□算机， □可以使用一根交叉的双□□直接□接那台独立路由器。

例 44. 主□公室或其它网□

网□使用的是星形拓□的 10 base T 以太网("双□□")。



大部分路由器/网□有一大好□就是，它□允□□在 同一 □□，有□个 分□独立的 PPP □接到□个分□的点上。□点在□多的 TA 上是不支持的， 除非□有□个串口的特定模式(通常都很□)。□不要把它与通道□接、MPP 等相混□。

□是个非常有用的功能，例如，如果在□的□公室里□有个□有的 ISDN □接，而且□想接入到里□，但休想□□一根 ISDN □也能工作。 □公室里的路由器能□管理□有的B通道□接到互□网 (64 Kbps) 以及使用□一个通道 B 来完成□独的数据□接。 第二个 B 通道可以用于□□、□出或□□与第一个B通道□行□接 (MPP等)，以□取更大□□。

以太网□也允□□□□的不□□是 IP 通信。□也可以□送 IPX/SPX 或其它任何□所使用的□□。

32.10. 网□地址□□

32.10.1. 概要

FreeBSD 的网□地址□□服□， 通常也被叫做 natd(8)， 是一个能□接收□入的未□理 IP 包， 将源地址修改□本□地址然后重新将□些包注入到□出 IP 包流中。 natd(8) 同□修改源地址和端口， 当接收到□□数据□，它作逆向□□以便把数据□回原先的□求者。

NAT 最常□的用途是□人□所熟知的 Internet □接共享。

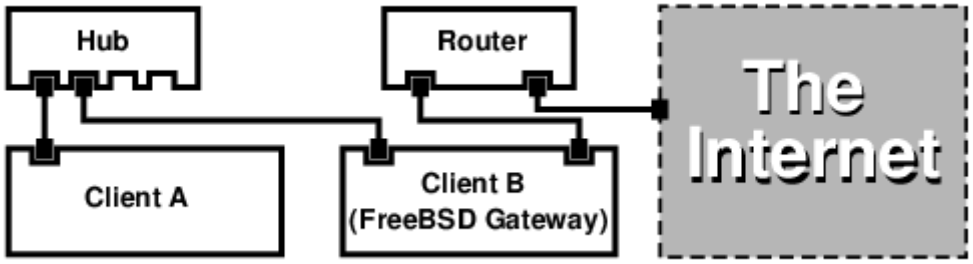
32.10.2. 安装

随着 IPv4 的 IP 地址空□的日益枯竭， 以及使用如 DSL 和□□等高速□接的用□的逐□□多， 越来越多的人□始需要 Internet □接共享□□的解决方案。 由于能□将□多□算机通□一个□外的 IP 地址□行接入， natd(8) 成□了一个理想的□□。

更□常□的情况， 一个用□通□□□或者 DSL □路 接入，并□有一个 IP 地址，同□，希望通□□台接入 Internet 的□算机来□ LAN 上更多的□算机提供接入服□。

□了完成□一任□， 接入 Internet 的 FreeBSD 机器必□扮演网□的角色。 □台网□必□有□□网□ - 一□用于□接 Internet 路由器， □一□用来□接 LAN。 所有 LAN 上的机器通□ Hub 或交□机□行□接。

有多□方法能□通□ FreeBSD 网□将 LAN 接入 Internet。 □个例子只介□了有至少□□网□的网□。



上述配置被广泛地用于共享 Internet □接。 LAN 中的一台机器□接到 Internet 中。 其余的□算机□通□那台"网□" 机来□接 Internet。

32.10.3. 引□加□器配置

在默□的 GENERIC 内核中， 并没有□用通□ natd(8) □行网址翻□的功能， 不□， □一功能可以通□在 /boot/loader.conf 中添加□□配置来在引□□自□予以加□：

```
ipfw_load="YES"
ipdivert_load="YES"
```

此外， □可以将引□加□器□量 net.inet.ip.fw.default_to_accept □□ 1：

```
net.inet.ip.fw.default_to_accept="1"
```

在□□始配置防火□和 NAT 网□□， □加□个配置是个好主意。 默□的 ipfw(8) □□将是 allow ip from any to any 而不是默□的 deny ip from any to any， □□， 在系□重□□，也就不太容易被反□在外面。

32.10.4. 内核配置

当不能使用内核模□， 或更希望将全部需要的功能□□□内核□， 可以在内核配置中添加下面的□置来□□：

```
options IPFIREWALL
options IPDIVERT
```

此外， 下列是一些可□的□□：

```
options IPFIREWALL_DEFAULT_TO_ACCEPT
options IPFIREWALL_VERBOSE
```

32.10.5. 系□引□□的配置

如果希望在系□引□□程中□用防火□和 NAT 支持， □在 /etc/rc.conf中添加下列配置：

```
gateway_enable="YES" ①
firewall_enable="YES" ②
firewall_type="OPEN" ③
natd_enable="YES"
natd_interface="fxp0" ④
natd_flags="" ⑤
```

① 将机器配置□网□。 □行 `sysctl net.inet.ip.forwarding=1` 效果相同。

② 在□□□□用 /etc/rc.firewall 中的防火□□□。

③ 指定一个□定□的允□所有包□入的防火□□□集。 参□ /etc/rc.firewall 以了解其他□型的□□集。

④ 指定通□□个网□接口□□包 (接入 Internet 的那一个)。

⑤ 其他希望在□□□□□□ natd(8) 的参数。

在 /etc/rc.conf 中加入上述□□将在系□□□□□行 `natd -interface fxp0`。 □一工作也可以手工完成。

当有太多□□要□□□，也可以使用一个 natd(8) 的配置文件来完成。□□情况下，□个配置文件必□通□在 /etc/rc.conf 里□加下面内容来定□：

```
natd_flags="-f /etc/natd.conf"
```

/etc/natd.conf 文件会包含一个配置□□列表， □行一个。在□跟部分的例子里将使用下面的文件：

```
redirect_port tcp 192.168.0.2:6667 6667
redirect_port tcp 192.168.0.3:80 80
```

□于配置文件的更多信息，参考 natd(8) 手册□中□于 `-f` □□那一部分。

在LAN后面的□一台机子和接口□□被分配私有地址空□(由RFC 1918定□) 里的 IP 地址，并且默□网□□成 natd 机子的内□ IP 地址。

例如：客□端 A 和 B 在 LAN 后面，IP 地址是 192.168.0.2 和 192.168.0.3，同□ natd 机子的 LAN 接口上的 IP 地址是 192.168.0.1。客□端 A 和 B 的默□网□必□要□成 natd 机子的 IP——192.168.0.1。natd 机子外□，或互□网接口不需要□了 natd(8) 而做任何特□的修改就可工作。

32.10.6. 端口重定向

使用 natd(8) 的缺点就是 LAN 客□不能从互□网□□。LAN 上的客□可以□行到外面的□接，而不能接收□来的□接。如果想在 LAN 的客□端机子上□行互□网服□，□就会有□□。 □此的一□□□方法是在 natd 机子上重定向□定的互□网端口到 LAN 客□端。

例如：在客□端 A 上□行 IRC 服□，而在客□端 B 上□行 web 服□。 想要正□的工作，在端口 6667 (IRC) 和 80 (web) 上接收到的□接就必□重定向到相□的机子上。

-redirect_port 需要使用□当的□□□送□ natd(8)。□法如下：

```
 -redirect_port proto targetIP:targetPORT[-targetPORT]
            [aliasIP:]aliasPORT[-aliasPORT]
            [remoteIP[:remotePORT[-remotePORT]]]
```

在上面的例子中，参数□□是：

```
-redirect_port tcp 192.168.0.2:6667 6667
-redirect_port tcp 192.168.0.3:80 80
```

□就会重定向□当的 *tcp* 端口到 LAN 上的客□端机子。

-redirect_port 参数可以用来指出端口□□来代替□个端口。例如， *tcp 192.168.0.2:2000-3000 2000-3000* 就会把所有在端口 2000 到 3000 上接收到的□接重定向到主机 A 上的端口 2000 到 3000。

当直接□行 natd(8) □，就可以使用□些□□， 把它□放到 /etc/rc.conf 里的 natd_flags="" □□上， 或通□一个配置文件□行□送。

想要更多配置□□，□参考 natd(8)。

32.10.7. 地址重定向

如果有几个 IP 地址提供，那□地址重定向就会很有用， 然而他□必□在一个机子上。使用它，natd(8) 就可以分配□□一个 LAN 客□端它□自己的外部 IP 地址。natd(8) 然后会使用□当的□部 IP 地址重写从 LAN 客□端外出的数据包， 以及重定向所有□来的数据包——一定的 IP 地址回到特定的 LAN 客□端。□也叫做静□ NAT。例如，IP 地址 128.1.1.1、 128.1.1.2 和 128.1.1.3 属于 natd 网□机子。 128.1.1.1 可以用来作 natd 网□机子的外□ IP 地址，而 128.1.1.2 和 128.1.1.3 用来□□回 LAN 客□端 A 和 B。

-redirect_address □法如下：

```
-redirect_address localIP publicIP
```

localIP	LAN 客□端的内部 IP 地址。
publicIP	相□ LAN 客□端的外部 IP 地址。

在□个例子里，参数是：

```
-redirect_address 192.168.0.2 128.1.1.2 -redirect_address 192.168.0.3 128.1.1.3
```

象 -redirect_port 一□，□些参数也是放在 /etc/rc.conf 里的 natd_flags="" □□上， 或通□一个配置文件□送□它。使用地址重定向， 就没有必要用端口重定向了，因□所有在某个 IP 地址上收到的数据都被重定向了。

在 natd 机子上的外部 IP 地址必□激活并且□名到 (aliased) 外□接口。要□做就看看 rc.conf(5)。

32.11. 并口□□ IP (PLIP)

PLIP 允□我□在□个并口□□行 TCP/IP。 在使用□□本□□， 或没有网□的□算机□， □会非常有用。 □一□中， 我□将□□：

- 制作用于并口的 (laplink) □□。
- 使用 PLIP □接□台□算机。

32.11.1. 制作并口□□。

□可以在□多□算机供□店里□到并口□□。 如果□不到， 或者希望自行制作， □可以参□下面的表格， 它介□了如何利用普通的打印机并口□□来改制：

表 *14.* 用于网□□接的并口□□接□方式

A-name	A 端	B 端	描述	Post/Bit
.... DATA0 -ERROR	.... 2 15	.... 15 2	数据	.... 0/0x01 1/0x08 ....
.... DATA1 +SLCT	.... 3 13	.... 13 3	数据	.... 0/0x02 1/0x10 ....
.... DATA2 +PE	.... 4 12	.... 12 4	数据	.... 0/0x04 1/0x20 ....
.... DATA3 -ACK	.... 5 10	.... 10 5	□冲 (Strobe)	.... 0/0x08 1/0x40 ....
.... DATA4 BUSY	.... 6 11	.... 11 6	数据	.... 0/0x10 1/0x80 ....
GND	18-25	18-25	GND	-

32.11.2. □置 PLIP

首先，□需要一根 laplink □。然后， □□□台□算机的内核都有□ lpt(4) □□程序的支持：

```
# grep lp /var/run/dmesg.boot
lpt0: Printer on ppbus0
lpt0: Interrupt-driven port
```

并口必□是一个中断□□的端口， □□在 /boot/device.hints 文件中配置：

```
hint.ppc.0.at="isa"
hint.ppc.0.irq="7"
```

然后□□内核配置文件中是否有一行 device plip 或加□了 plip.ko 内核模□。 □□□情况下， 在使用 ifconfig(8) 命令□都会□示并口□□的网□接口， □似□□：

```
# ifconfig plip0
plip0: flags=8810<POINTOPOINT,SIMPLEX,MULTICAST> mtu 1500
```

用 laplink □接通□台□算机的并口。

在□□以 root 身□配置通□参数。 例如， 如果□希望将 host1 通□□一台机器 host2 □接：

```
                host1 ----- host2
IP Address    10.0.0.1      10.0.0.2
```

配置 host1 上的网□接口，照此做：

```
# ifconfig plip0 10.0.0.1 10.0.0.2
```

配置 host2 上的网□接口，照此做：

```
# ifconfig plip0 10.0.0.2 10.0.0.1
```

□□在□□有个工作的□接了。想要更□□的信息， □□□ lp(4) 和 lpt(4) 手册□。

□□□□□加□个主机到 /etc/hosts：

```
127.0.0.1               localhost.my.domain localhost
10.0.0.1                host1.my.domain host1
10.0.0.2                host2.my.domain host2
```

要□□□接是否工作，可以到□一台机子上，然后 ping □外一台。例如，在 host1 上：

```
# ifconfig plip0
plip0: flags=8851<UP,POINTOPOINT,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        inet 10.0.0.1 --> 10.0.0.2 netmask 0xff000000
# netstat -r
Routing tables

Internet:
Destination        Gateway           Flags     Refs     Use      Netif Expire
host2              host1             UH          0       0       plip0
# ping -c 4 host2
PING host2 (10.0.0.2): 56 data bytes
64 bytes from 10.0.0.2: icmp_seq=0 ttl=255 time=2.774 ms
64 bytes from 10.0.0.2: icmp_seq=1 ttl=255 time=2.530 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=255 time=2.556 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=255 time=2.714 ms

--- host2 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 2.530/2.643/2.774/0.103 ms
```

32.12. IPv6

IPv6 (也被称作 IPng "下一代 IP") 是□所周知的 IP □□ (也叫 IPv4) 的新版本。 和其他□代的 *BSD 系□一□, FreeBSD 包含了 KAME 的 IPv6 参考□□。 因此， □的 FreeBSD 系□包含了□□ IPv6 所需要的所有工具。 □一□主要集中□□如何配置和使用 IPv6。

在 1990 年代早期， 人□□始担心可用的 IPv4 地址空□在不断地□小。 随着 Internet 的爆炸式□展， 主要的□个担心是：

- 用尽所有的地址。 当然□在□个□□已□不再那□尖□， 因□ RFC1918 私有地址空□ (10.0.0.0/8、172.16.0.0/12， 以及 192.168.0.0/16) 和网□地址□□ (NAT) 技□已□被广泛采用。
- 路由表条目□得太大。□点今天仍然是焦点。

IPv6 解决□些和其它□多的□□：

- 128 位地址空□。□句□，理□上有 340,282,366,920,938,463,463,374,607,431,768,211,456 个地址可以使用。□意味着在我□的星球上□平方米大□有 6.67 * 10^27 个 IPv6 地址。
- 路由器□在它□的路由表里存放网□地址集， □就□少路由表的平均空□到 8192 个条目。

IPv6 □有其它□多有用的功能，如：

- 地址自□配置 (RFC2462)
- Anycast (任意播) 地址("一□多")
- □制的多播地址
- IPsec (IP 安全)
- □□的□□□

- 移□的 (Mobile) IP
- IPv6 到 IPv4 的□□机制

要更多信息，□□看：

- IPv6 概□，在playground.sun.com
- KAME.net

32.12.1. □于 IPv6 地址的背景知□

有几□不同□型的 IPv6 地址：Unicast，Anycast 和 Multicast。

Unicast 地址是□人□所熟知的地址。一个被□送到 unicast 地址的包□□上会到□属于□个地址的接口。

Anycast 地址□□上与 unicast 地址没有差□， 只是它□□□一□接口。指定□ anycast 地址的包会到□最近的 (以路由□□位) 接口。Anycast 地址可能只被路由器使用。

Multicast 地址□□一□接口。指定□ multicast 地址的包会到□属于 multicast □的所有的接口。

IPv4 广播地址 (通常□ `xxx.xxx.xxx.255`) 由 IPv6 的 multicast 地址来表示。

表 *15.* 保留的 *IPv6* 地址

IPv6 地址	□定□度 **(bits)**	描述	□注
`::`	128 bits	未指定	□似 IPv4 中的 `0.0.0.0`
`::1`	128 bits	□回地址	□似 IPv4 中的 `127.0.0.1`
`::00:xx:xx:xx:xx`	96 bits	嵌入的 IPv4	低 32 bits 是 IPv4 地址。□也称作 "IPv4 兼容 IPv6 地址"
`::ff:xx:xx:xx:xx`	96 bits	IPv4 影射的 IPv6 地址	低的 32 bits 是 IPv4 地址。 用于那些不支持 IPv6 的主机。
`fe80::` - `feb::`	10 bits	□路□回	□似 IPv4 的□回地址。
`fec0::` - `fef::`	10 bits	站点□回	
`ff::`	8 bits	多播	
`001` (base 2)	3 bits	全球多播	所有的全球多播地址都指定到□个地址池中。前三个二□制位是 "001"。

32.12.2. IPv6 地址的□法

□□形式被描述□：`x:x:x:x:x:x:x:x`， □一个"x"就是一个 16 位的 16 □制□。当然， □个十六□制□以三个"0"□始□的也可以省略。如 `FEBC:A574:382B:23C1:AA49:4592:4EFE:9982`

通常一个地址会有很□的子串全部□零， 因此□个地址的□□子串常被□写□"::"。 例如：`fe80::1` □□的□□形式是 `fe80:0000:0000:0000:0000:0000:0000:0001`。

第三□形式是以□所周知的用点"."作□分隔符的十□制 IPv4 形式，写出最后 32 Bit 的部分。例如 2002::10.0.0.1 □□的十□制正□表□方式是 2002:0000:0000:0000:0000:0000:0a00:0001 它也相当于写成 2002::a00:1.

到□在，□者□□能理解下面的内容了：

```
# ifconfig
```

```
rl0: flags=8943UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MULTICAST mtu 1500
        inet 10.0.0.10 netmask 0xffffff00 broadcast 10.0.0.255
        inet6 fe80::200:21ff:fe03:8e1%rl0 prefixlen 64 scopeid 0x1
        ether 00:00:21:03:08:e1
        media: Ethernet autoselect (100baseTX )
        status: active
```

fe80::200:21ff:fe03:8e1%rl0 是一个自□配置的□路□回地址。它作□自□配置的一部分由 MAC 生成。

□于 IPv6 地址的□□的更多信息，□参看 RFC3513。

32.12.3. □行□接

目前，有四□方式可以□接到其它 IPv6 主机和网□：

- 咨□□的互□网服□提供商是否提供 IPv6。
- SixXS 向全球□□提供通道。
- 使用 6-to-4 通道 (RFC3068)
- 如果□使用的是□号□接， □可以使用 net/freenet6 port。

32.12.4. IPv6 世界里的 DNS

□于 IPv6 有□□□型的 DNS □□：IETF 已□宣布 A6 是□□□准；□行的□准是 AAAA □□。

使用AAAA□□是很□□的。通□□加下面内容， □□的主机分配置□□才接收到的新的 IPv6 地址：

```
MYHOSTNAME           AAAA    MYIPv6ADDR
```

到□的主域 DNS 文件里，就可以完成。要是□自已没有 DNS 域服□，□可以□□□的 DNS 提供商。目前的 bind 版本 (version 8.3 与 9) 和 dns/djbdns(含IPv6□丁) 支持 AAAA □□。

32.12.5. 在 /etc/rc.conf 中□行所需的修改

32.12.5.1. IPv6 客□机□置

□些□置将□助□把一台□ LAN 上的机器配置□一台客□机， 而不是路由器。 要□ rtsol(8) 在□□□自□配置□的网□，只需添加：

```
ipv6_enable="YES"
```

要自□地静□指定 IP 地址， 例如 2001:471:1f11:251:290:27ff:fee0:2093， 到 fxp0 上， □写上：

```
ipv6_ifconfig_fxp0="2001:471:1f11:251:290:27ff:fee0:2093"
```

要指定 2001:471:1f11:251::1 作□默□路由， 需要在 /etc/rc.conf 中加入：

```
ipv6_defaultrouter="2001:471:1f11:251::1"
```

32.12.5.2. IPv6 路由器/网□配置

□将□助□从隧道提供商那里取得必要的□料， 并将□些□料□化□在重□□能□保持住的□置。 要在□□□恢□□的隧道，需要在 /etc/rc.conf 中□加：

列出要配置的通用隧道接口， 例如 gif0：

```
gif_interfaces="gif0"
```

配置□接口使用本地端地址 *MY_IPv4_ADDR* 和□程端地址 *REMOTE_IPv4_ADDR*：

```
gifconfig_gif0="MY_IPv4_ADDR REMOTE_IPv4_ADDR"
```

□用分配□□用于 IPv6 隧道□端的 IPv6 地址， 需要□加：

```
ipv6_ifconfig_gif0="MY_ASSIGNED_IPv6_TUNNEL_ENDPOINT_ADDR"
```

此后十□置 IPv6 的默□路由。 □是 IPv6 隧道的□一端：

```
ipv6_defaultrouter="MY_IPv6_REMOTE_TUNNEL_ENDPOINT_ADDR"
```

32.12.5.3. IPv6 隧道配置

如果服□器将□的网□通□ IPv6 路由到世界的其他角落， □需要在 /etc/rc.conf 中添加下面的配置：

```
ipv6_gateway_enable="YES"
```

32.12.6. 路由宣告和主机自□配置

□□将□助□配置 rtadvd(8) 来宣示默□的 IPv6 路由。

要□用 rtadvd(8) □需要在 /etc/rc.conf 中添加：

```
rtadvd_enable="YES"
```

指定由□个网□接口来完成 IPv6 路由□求非常重要。 □例来□， □ rtadvd(8) 使用 fxp0：

```
rtadvd_interfaces="fxp0"
```

接下来我□需要□建配置文件， /etc/rtadvd.conf。 示例如下：

```
fxp0:\
    :addrs#1:addr="2001:471:1f11:246::":prefixlen#64:tc=ether:
```

将 fxp0 改□□打算使用的接口名。

接下来， 将 2001:471:1f11:246:: 改□分配□□的地址前□。

如果□□有□用的 /64 子网， □不需要修改其他□置。 反之， □需要把 prefixlen# 改□正□的□。

32.13. □□□□模式 (ATM)

32.13.1. 配置 classical IP over ATM (PVCs)

Classical IP over ATM (CLIP) 是一□最□□的使用□ IP 的 ATM 的方法。 □□方法可以用在交□式□接 (SVC) 和永久□接 (PVC) 上。□部分描述的就是配置基于 PVC 的网□。

32.13.1.1. 完全互□的配置

第一□使用PVC来□置 CLIP 的方式就是通□□用的 PVC □网□里的□一台机子都互□在一起。 尽管□□配置起来很□□，但□于数量更多一点的机子来□就有些不切□□了。 例如我□有四台机子在网□里，□一台都使用一□ ATM □配器□□接到 ATM 网□。第一□就是□□ IP 地址和机子□的 ATM □接。我□使用下面的：

主机	IP 地址
hostA	192.168.173.1
hostB	192.168.173.2
hostC	192.168.173.3
hostD	192.168.173.4

□了建造完全交□的网□，我□需要在第一□机子□有一个 ATM □接：

机器	VPI.VCI □
hostA - hostB	0.100
hostA - hostC	0.101
hostA - hostD	0.102

机器	VPI.VCI □
hostB - hostC	0.103
hostB - hostD	0.104
hostC - hostD	0.105

在□一个□接端 VPI 和 VCI 的□都可能会不同， 只是□了□□起□，我□假定它□是一□的。 下一□我□需要配置□一个主机上的 ATM 接口：

```
hostA# ifconfig hatm0 192.168.173.1 up
hostB# ifconfig hatm0 192.168.173.2 up
hostC# ifconfig hatm0 192.168.173.3 up
hostD# ifconfig hatm0 192.168.173.4 up
```

假定所有主机上的 ATM 接口都是 hatm0。 □在 PVC 需要配置到 hostA 上 (我□假定它□都已□配置在了 ATM 交□机上，至于□□做的， □就需要参考一下□交□机的手册了)。

```
hostA# atmconfig natm add 192.168.173.2 hatm0 0 100 llc/snap ubr
hostA# atmconfig natm add 192.168.173.3 hatm0 0 101 llc/snap ubr
hostA# atmconfig natm add 192.168.173.4 hatm0 0 102 llc/snap ubr

hostB# atmconfig natm add 192.168.173.1 hatm0 0 100 llc/snap ubr
hostB# atmconfig natm add 192.168.173.3 hatm0 0 103 llc/snap ubr
hostB# atmconfig natm add 192.168.173.4 hatm0 0 104 llc/snap ubr

hostC# atmconfig natm add 192.168.173.1 hatm0 0 101 llc/snap ubr
hostC# atmconfig natm add 192.168.173.2 hatm0 0 103 llc/snap ubr
hostC# atmconfig natm add 192.168.173.4 hatm0 0 105 llc/snap ubr

hostD# atmconfig natm add 192.168.173.1 hatm0 0 102 llc/snap ubr
hostD# atmconfig natm add 192.168.173.2 hatm0 0 104 llc/snap ubr
hostD# atmconfig natm add 192.168.173.3 hatm0 0 105 llc/snap ubr
```

当然，除 UBR 外其它的通信□定也可□ ATM □配器支持□些。 此□情况下，通信□定的名字要跟人通信参数后□。工具 atmconfig(8) 的□助可以□□得到：

```
# atmconfig help natm add
```

或者在 atmconfig(8) 手册□里得到。

相同的配置也可以通□ /etc/rc.conf 来完成。□于 hostA，看起来就象□□：

```
network_interfaces="lo0 hatm0"
ifconfig_hatm0="inet 192.168.173.1 up"
natm_static_routes="hostB hostC hostD"
route_hostB="192.168.173.2 hatm0 0 100 llc/snap ubr"
route_hostC="192.168.173.3 hatm0 0 101 llc/snap ubr"
route_hostD="192.168.173.4 hatm0 0 102 llc/snap ubr"
```

所有 CLIP 路由的当前状□可以使用如下命令□得：

```
hostA# atmconfig natm show
```

32.14. Common Address Redundancy Protocol (CARP，共用地址冗余□□)

Common Address Redundancy Protocol， 或□称 CARP 能□使多台主机共享同一 IP 地址。在某些配置中， □□做可以提高可用性， 或□□□□均衡。 下面的例子中， □些主机也可以同□使用其他的不同的 IP 地址。

要□用 CARP 支持， 必□在 FreeBSD 内核配置中□加下列□□， 并按照 配置FreeBSD的内核 章□介□的方法重新□□内核：

```
device  carp
```

□外的一个方法是在□□□加□ if_carp.ko 模□。 把如下的□行加入到 /boot/loader.conf：

```
if_carp_load="YES"
```

□□就可以使用 CARP 功能了， 一些具体的参数， 可以通□一系列 sysctlOID 来□整。

OID	描述
net.inet.carp.allow	接受□来的 CARP 包。 默□□用。
net.inet.carp.preempt	当主机中有一个 CARP 网□接口失去□□□， □个□□将停止□台主机上所有的 CARP 接口。 默□禁用。
net.inet.carp.log	当□□ 0 表示禁止□□所有日志。 □□ 1 表示□□□坏的 CARP 包。任何大于 1 表示□□ CARP 网□接口的状□□化。默□□□ 1。
net.inet.carp.arpbalance	使用 ARP 均衡本地网□流量。 默□禁用。
net.inet.carp.suppress_preempt	此只□ OID □示抑制□占的状□。 如果一个接口上的□接失去□□, □□占会被抑制。 当□个□量的□□ 0 □，表示□占未被抑制。 任何□□都会使 OID □□。

CARP □□可以通□ ifconfig 命令来□建。

```
# ifconfig carp0 create
```

在真□□境中， □些接口需要一个称作 VHID 的□□□号。 □个 VHID 或 Virtual Host Identification (虚□主机□□) 用于在网□上区分主机。

32.14.1. 使用 CARP 来改善服□的可用性 (CARP)

如前面提到的那□， CARP 的作用之一是改善服□的可用性。 □个例子中， 将□三台主机提供故障□移服□， □三台服□器各自有独立的 IP 地址， 并提供完全一□的 web 内容。 三台机器以 DNS □□的方式提供服□。 用于故障□移的机器有□个 CARP 接口， 分□配置□外□台服□器的 IP 地址。 当有服□器□生故障□， □台机器会自□得到故障机的 IP 地址。 □□以来， 用□就完全感□不到□生了故障。 故障□移的服□器提供的内容和服□， □与其□之提供□□□的服□器一致。

□台机器的配置， 除了主机名和 VHID 之外□完全一致。 在我□的例子中， □□台机器的主机名分□是 hosta.example.org 和 hostb.example.org。 首先， 需要将 CARP 配置加入到 rc.conf。 □于 hosta.example.org 而言， rc.conf 文件中□包含下列配置：

```
hostname="hosta.example.org"
ifconfig_fxp0="inet 192.168.1.3 netmask 255.255.255.0"
cloned_interfaces="carp0"
ifconfig_carp0="vhid 1 pass testpass 192.168.1.50/24"
```

在 hostb.example.org 上， □□的 rc.conf 配置□是：

```
hostname="hostb.example.org"
ifconfig_fxp0="inet 192.168.1.4 netmask 255.255.255.0"
cloned_interfaces="carp0"
ifconfig_carp0="vhid 2 pass testpass 192.168.1.51/24"
```

在□台机器上由 ifconfig 的 pass □□指定的密□必□是一致的， □一点非常重要。 carp □□只会□听和接受来自持有正□密□的机器的公告。 此外， 不同虚□主机的 VHID 必□不同。

第三台机器， provider.example.org 需要□行配置， 以便在□外□台机器出□□□□接管。 □台机器需要□个 carp □□， 分□□理□个机器。 □□的 rc.conf 配置□似下面□□：

```
hostname="provider.example.org"
ifconfig_fxp0="inet 192.168.1.5 netmask 255.255.255.0"
cloned_interfaces="carp0 carp1"
ifconfig_carp0="vhid 1 advskew 100 pass testpass 192.168.1.50/24"
ifconfig_carp1="vhid 2 advskew 100 pass testpass 192.168.1.51/24"
```

配置□个 carp □□， 能□□ provider.example.org 在□台机器中的任何一个停止□□□， 立即接管其 IP 地址。

默□的 FreeBSD 内核 可能 □用了主机□□占。 如果是□□的□， provider.example.org 可能在正式的内容服□器恢□□不□放 IP 地址。 此□， 管理□必□手工□制 IP 回到原来内容服□器。 具体做法是在 provider.example.org 上使用下面的命令：

```
# ifconfig carp0 down  ifconfig carp0 up
```

□个操作需要在与出□□□的主机□□的那个 carp 接口上□行。

□在□已□完成了 CARP 的配置， 并可以□始□□了。 □□□程中， 可以随□重□或切断□台机器的网□。

如欲了解更多□□， □参□ carp(4) □机手册。

部分 V: 附□

附□ A: □取 FreeBSD

A.1. CDROM 和 DVD □行商

A.1.1. 零售盒装□品

可以从下面几个零售商那里□到 FreeBSD 的盒装□品 (FreeBSD CD， 附加□件，印刷文□)：

- CompUSA
  WWW: http://www.compusa.com/
- Frys Electronics
  WWW: http://www.frys.com/

A.1.2. CD 和 DVD 光□

FreeBSD CD 和 DVD 光□可以从□多在□零售商那里□到：

- FreeBSD Mall, Inc.
  700 Harvest Park Ste F
  Brentwood, CA 94513
  USA
  Phone: +1 925 240-6652
  Fax: +1 925 674-0821
  Email: <info@freebsdmall.com>
  WWW: http://www.freebsdmall.com/
- Dr. Hinner EDV
  St. Augustinus-Str. 10
  D-81825 München
  Germany
  Phone: (089) 428 419
  WWW: http://www.hinner.de/linux/freebsd.html
- Ikarios
  22-24 rue Voltaire
  92000 Nanterre
  France
  WWW: http://ikarios.com/form/#freebsd
- JMC Software
  Ireland
  Phone: 353 1 6291282
  WWW: http://www.thelinuxmall.com
- The Linux Emporium
  Hilliard House, Lester Way
  Wallingford
  OX10 9TA
  United Kingdom

Phone: +44 1491 837010
Fax: +44 1491 837016
WWW: http://www.linuxemporium.co.uk/products/bsd/

- Linux+ DVD Magazine
  Lewartowskiego 6
  Warsaw
  00-190
  Poland
  Phone: +48 22 860 18 18
  Email: <editors@lpmagazine.org>
  WWW: http://www.lpmagazine.org/
- Linux System Labs Australia
  21 Ray Drive
  Balwyn North
  VIC - 3104
  Australia
  Phone: +61 3 9857 5918
  Fax: +61 3 9857 8974
  WWW: http://www.lsl.com.au
- LinuxCenter.Ru
  Galernaya Street, 55
  Saint-Petersburg
  190000
  Russia
  Phone: +7-812-3125208
  Email: <info@linuxcenter.ru>
  WWW: http://linuxcenter.ru/shop/freebsd

A.1.3. □行人

如果□是□售商并且想□售 FreeBSD CDROM □品， □和□行人□系：

- Cylogistics
  809B Cuesta Dr., #2149
  Mountain View, CA 94040
  USA
  Phone: +1 650 694-4949
  Fax: +1 650 694-4953
  Email: <sales@cylogistics.com>
  WWW: http://www.cylogistics.com/
- Ingram Micro
  1600 E. St. Andrew Place
  Santa Ana, CA 92705-4926
  USA
  Phone: 1 (800) 456-8000
  WWW: http://www.ingrammicro.com/

- Kudzu, LLC
  7375 Washington Ave. S.
  Edina, MN 55439
  USA
  Phone: +1 952 947-0822
  Fax: +1 952 947-0876
  Email: <sales@kudzuenterprises.com>
- LinuxCenter.Kz
  Ust-Kamenogorsk
  Kazakhstan
  Phone: +7-705-501-6001
  Email: <info@linuxcenter.kz>
  WWW: http://linuxcenter.kz/page.php?page=fr
- LinuxCenter.Ru
  Galernaya Street, 55
  Saint-Petersburg
  190000
  Russia
  Phone: +7-812-3125208
  Email: <info@linuxcenter.ru>
  WWW: http://linuxcenter.ru/freebsd
- Navarre Corp
  7400 49th Ave South
  New Hope, MN 55428
  USA
  Phone: +1 763 535-8333
  Fax: +1 763 535-0341
  WWW: http://www.navarre.com/

A.2. FTP 站点

官方的 FreeBSD 源代□可以从遍布全球的□像站点 通□匿名 FTP 下□。 站点 ftp://ftp.FreeBSD.org/pub/FreeBSD/ 有着良好的网□□接并且允□大量的并□□接， 但是 □或□更想□一个 "更近的" □像站点 (特□是当□想□行某□形式的□像的□候)。

FreeBSD 可以从下面□些□像站点通□匿名 FTP 下□。如果□□□了通□匿名 FTP □取 FreeBSD， □尽量使用□□比□近的站点。被列□ "主□像站点" 的□像站点一般都有完整的 FreeBSD 文件 (□□□□体系□□的所有当前可用的版本)， □或□从□所在的国家或地区的站点下□会得到更快的下□速度。□个站点提供了最流行的体系□□的最近的版本而有可能不提供完整的 FreeBSD 存□。所有的站点都提供匿名 FTP □□而有些站点也提供其他的□□方式。□□个站点可用的□□方式 在其主机名后有所□明。

Central Servers, Primary Mirror Sites, Armenia, Australia, Austria, Brazil, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hong Kong, Ireland, Japan, Korea, Latvia, Lithuania, Netherlands, New Zealand, Norway, Poland, Russia, Saudi Arabia, Slovenia, South Africa, Spain, Sweden, Switzerland, Taiwan, Ukraine, United Kingdom, United States of America.

(as of UTC)

Central Servers

ftp://ftp.FreeBSD.org/pub/FreeBSD/ (ftp / ftpv6 / http://ftp.FreeBSD.org/pub/FreeBSD/ / http://ftp.FreeBSD.org/pub/FreeBSD/)

Primary Mirror Sites

In case of problems, please contact the hostmaster <mirror-admin@FreeBSD.org> for this domain.

- ftp://ftp1.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp3.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp4.FreeBSD.org/pub/FreeBSD/ (ftp / ftpv6 / http://ftp4.FreeBSD.org/pub/FreeBSD/ / http://ftp4.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp5.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp6.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp7.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp10.FreeBSD.org/pub/FreeBSD/ (ftp / ftpv6 / http://ftp10.FreeBSD.org/pub/FreeBSD/ / http://ftp10.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp11.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp13.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp14.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp14.FreeBSD.org/pub/FreeBSD/)

Armenia

In case of problems, please contact the hostmaster <hostmaster@am.FreeBSD.org> for this domain.

- ftp://ftp1.am.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp1.am.FreeBSD.org/pub/FreeBSD/ / rsync)

Australia

In case of problems, please contact the hostmaster <hostmaster@au.FreeBSD.org> for this domain.

- ftp://ftp.au.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.au.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp3.au.FreeBSD.org/pub/FreeBSD/ (ftp)

Austria

In case of problems, please contact the hostmaster <hostmaster@at.FreeBSD.org> for this domain.

- ftp://ftp.at.FreeBSD.org/pub/FreeBSD/ (ftp / ftpv6 / http://ftp.at.FreeBSD.org/pub/FreeBSD/ / http://ftp.at.FreeBSD.org/pub/FreeBSD/)

Brazil

In case of problems, please contact the hostmaster <hostmaster@br.FreeBSD.org> for this domain.

- ftp://ftp2.br.FreeBSD.org/FreeBSD/ (ftp / http://ftp2.br.FreeBSD.org/)
- ftp://ftp3.br.FreeBSD.org/pub/FreeBSD/ (ftp / rsync)
- ftp://ftp4.br.FreeBSD.org/pub/FreeBSD/ (ftp)

Czech Republic

In case of problems, please contact the hostmaster <hostmaster@cz.FreeBSD.org> for this domain.

- ftp://ftp.cz.FreeBSD.org/pub/FreeBSD/ (ftp / ftp://ftp.cz.FreeBSD.org/pub/FreeBSD/ / http://ftp.cz.FreeBSD.org/pub/FreeBSD/ / http://ftp.cz.FreeBSD.org/pub/FreeBSD/ / rsync / rsyncv6)
- ftp://ftp2.cz.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp2.cz.FreeBSD.org/pub/FreeBSD/)

Denmark

In case of problems, please contact the hostmaster <hostmaster@dk.FreeBSD.org> for this domain.

- ftp://ftp.dk.FreeBSD.org/pub/FreeBSD/ (ftp / ftpv6 / http://ftp.dk.FreeBSD.org/pub/FreeBSD/ / http://ftp.dk.FreeBSD.org/pub/FreeBSD/)

Estonia

In case of problems, please contact the hostmaster <hostmaster@ee.FreeBSD.org> for this domain.

- ftp://ftp.ee.FreeBSD.org/pub/FreeBSD/ (ftp)

Finland

In case of problems, please contact the hostmaster <hostmaster@fi.FreeBSD.org> for this domain.

- ftp://ftp.fi.FreeBSD.org/pub/FreeBSD/ (ftp)

France

In case of problems, please contact the hostmaster <hostmaster@fr.FreeBSD.org> for this domain.

- ftp://ftp.fr.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp1.fr.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp1.fr.FreeBSD.org/pub/FreeBSD/ / rsync)
- ftp://ftp3.fr.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp6.fr.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp6.fr.FreeBSD.org/pub/FreeBSD/ (ftp / rsync)
- ftp://ftp7.fr.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp8.fr.FreeBSD.org/pub/FreeBSD/ (ftp)

Germany

In case of problems, please contact the hostmaster <de-bsd-hubs@de.FreeBSD.org> for this domain.

- ftp://ftp.de.FreeBSD.org/pub/FreeBSD/ (ftp)

- ftp://ftp1.de.FreeBSD.org/freebsd/ (ftp / http://www1.de.FreeBSD.org/freebsd/ / rsync://rsync3.de.FreeBSD.org/freebsd/)
- ftp://ftp2.de.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp2.de.FreeBSD.org/pub/FreeBSD/ / rsync)
- ftp://ftp4.de.FreeBSD.org/FreeBSD/ (ftp / http://ftp4.de.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp5.de.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp7.de.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp7.de.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp8.de.FreeBSD.org/pub/FreeBSD/ (ftp)

Greece

In case of problems, please contact the hostmaster <hostmaster@gr.FreeBSD.org> for this domain.

- ftp://ftp.gr.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.gr.FreeBSD.org/pub/FreeBSD/ (ftp)

Hong Kong

ftp://ftp.hk.FreeBSD.org/pub/FreeBSD/ (ftp)

Ireland

In case of problems, please contact the hostmaster <hostmaster@ie.FreeBSD.org> for this domain.

- ftp://ftp3.ie.FreeBSD.org/pub/FreeBSD/ (ftp / rsync)

Japan

In case of problems, please contact the hostmaster <hostmaster@ie.FreeBSD.org> for this domain.

- ftp://ftp.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp3.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp4.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp5.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp6.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp7.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp8.jp.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp9.jp.FreeBSD.org/pub/FreeBSD/ (ftp)

Korea

In case of problems, please contact the hostmaster <hostmaster@kr.FreeBSD.org> for this domain.

- ftp://ftp.kr.FreeBSD.org/pub/FreeBSD/ (ftp / rsync)
- ftp://ftp2.kr.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp2.kr.FreeBSD.org/pub/FreeBSD/)

Latvia

In case of problems, please contact the hostmaster <hostmaster@lv.FreeBSD.org> for this domain.

- ftp://ftp.lv.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.lv.FreeBSD.org/pub/FreeBSD/)

Lithuania

In case of problems, please contact the hostmaster <hostmaster@lt.FreeBSD.org> for this domain.

- ftp://ftp.lt.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.lt.FreeBSD.org/pub/FreeBSD/)

Netherlands

In case of problems, please contact the hostmaster <hostmaster@nl.FreeBSD.org> for this domain.

- ftp://ftp.nl.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.nl.FreeBSD.org/os/FreeBSD/ / rsync)
- ftp://ftp2.nl.FreeBSD.org/pub/FreeBSD/ (ftp)

New Zealand

- ftp://ftp.nz.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.nz.FreeBSD.org/pub/FreeBSD/)

Norway

In case of problems, please contact the hostmaster <hostmaster@no.FreeBSD.org> for this domain.

- ftp://ftp.no.FreeBSD.org/pub/FreeBSD/ (ftp / rsync)

Poland

In case of problems, please contact the hostmaster <hostmaster@pl.FreeBSD.org> for this domain.

- ftp://ftp.pl.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp2.pl.FreeBSD.org

Russia

In case of problems, please contact the hostmaster <hostmaster@ru.FreeBSD.org> for this domain.

- ftp://ftp.ru.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.ru.FreeBSD.org/FreeBSD/ / rsync)
- ftp://ftp2.ru.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp2.ru.FreeBSD.org/pub/FreeBSD/ / rsync)
- ftp://ftp5.ru.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp5.ru.FreeBSD.org/pub/FreeBSD/ / rsync)
- ftp://ftp6.ru.FreeBSD.org/pub/FreeBSD/ (ftp)

Saudi Arabia

In case of problems, please contact the hostmaster <ftpadmin@isu.net.sa> for this domain.

- ftp://ftp.isu.net.sa/pub/ftp.freebsd.org (ftp)

Slovenia

In case of problems, please contact the hostmaster <hostmaster@si.FreeBSD.org> for this domain.

- ftp://ftp.si.FreeBSD.org/pub/FreeBSD/ (ftp)

South Africa

In case of problems, please contact the hostmaster <hostmaster@za.FreeBSD.org> for this domain.

- ftp://ftp.za.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.za.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp4.za.FreeBSD.org/pub/FreeBSD/ (ftp)

Spain

In case of problems, please contact the hostmaster <hostmaster@es.FreeBSD.org> for this domain.

- ftp://ftp.es.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.es.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp3.es.FreeBSD.org/pub/FreeBSD/ (ftp)

Sweden

In case of problems, please contact the hostmaster <hostmaster@se.FreeBSD.org> for this domain.

- ftp://ftp.se.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.se.FreeBSD.org/pub/FreeBSD/ (ftp / rsync://ftp2.se.FreeBSD.org/)
- ftp://ftp3.se.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp4.se.FreeBSD.org/pub/FreeBSD/ (ftp / ftp://ftp4.se.FreeBSD.org/pub/FreeBSD/ / http://ftp4.se.FreeBSD.org/pub/FreeBSD/ / http://ftp4.se.FreeBSD.org/pub/FreeBSD/ / rsync://ftp4.se.FreeBSD.org/pub/FreeBSD/ / rsync://ftp4.se.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp6.se.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp6.se.FreeBSD.org/pub/FreeBSD/)

Switzerland

In case of problems, please contact the hostmaster <hostmaster@ch.FreeBSD.org> for this domain.

- ftp://ftp.ch.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.ch.FreeBSD.org/pub/FreeBSD/)

Taiwan

In case of problems, please contact the hostmaster <hostmaster@tw.FreeBSD.org> for this domain.

- ftp://ftp.ch.FreeBSD.org/pub/FreeBSD/ (ftp / ftp://ftp.tw.FreeBSD.org/pub/FreeBSD/ / rsync / rsyncv6)
- ftp://ftp2.tw.FreeBSD.org/pub/FreeBSD/ (ftp / ftp://ftp2.tw.FreeBSD.org/pub/FreeBSD/ / http://ftp2.tw.FreeBSD.org/pub/FreeBSD/ / http://ftp2.tw.FreeBSD.org/pub/FreeBSD/ / rsync / rsyncv6)

- ftp://ftp4.tw.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp5.tw.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp6.tw.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp6.tw.FreeBSD.org/ / rsync)
- ftp://ftp7.tw.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp8.tw.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp11.tw.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp11.tw.FreeBSD.org/FreeBSD/)
- ftp://ftp12.tw.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp13.tw.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp14.tw.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp15.tw.FreeBSD.org/pub/FreeBSD/ (ftp)

Ukraine

- ftp://ftp.ua.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp.ua.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp6.ua.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp6.ua.FreeBSD.org/pub/FreeBSD / rsync://ftp6.ua.FreeBSD.org/FreeBSD/)
- ftp://ftp7.ua.FreeBSD.org/pub/FreeBSD/ (ftp)

United Kingdom

In case of problems, please contact the hostmaster <hostmaster@uk.FreeBSD.org> for this domain.

- ftp://ftp.uk.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.uk.FreeBSD.org/pub/FreeBSD/ (ftp / rsync://ftp2.uk.FreeBSD.org/ftp.freebsd.org/pub/FreeBSD/)
- ftp://ftp3.uk.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp4.uk.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp5.uk.FreeBSD.org/pub/FreeBSD/ (ftp)

United States of America

In case of problems, please contact the hostmaster <hostmaster@us.FreeBSD.org> for this domain.

- ftp://ftp1.us.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp2.us.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp3.us.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp4.us.FreeBSD.org/pub/FreeBSD/ (ftp / ftpv6 / http://ftp4.us.FreeBSD.org/pub/FreeBSD/ / http://ftp4.us.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp5.us.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp6.us.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp8.us.FreeBSD.org/pub/FreeBSD/ (ftp)

- ftp://ftp10.us.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp11.us.FreeBSD.org/pub/FreeBSD/ (ftp)
- ftp://ftp13.us.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp13.us.FreeBSD.org/pub/FreeBSD/ / rsync)
- ftp://ftp14.us.FreeBSD.org/pub/FreeBSD/ (ftp / http://ftp14.us.FreeBSD.org/pub/FreeBSD/)
- ftp://ftp15.us.FreeBSD.org/pub/FreeBSD/ (ftp)

A.3. BitTorrent

基本□行版 CD 的 ISO □像也可以通□ BitTorrent □得。 用□下□□像的 torrent 文件能□在□里□到 http://torrents.freebsd.org:8080

BitTorrent 客□端□件可以从□个 port net-p2p/py-bittorrent 或□□□的二□制包安装。

在通□ BitTorrent 下□了 ISO □像之后， □可以照着 burncdburncd 中的所描述的方法□□到 CD 或 DVD 介□上。

A.4. 匿名 CVS

A.4.1. 概述

匿名 CVS(或人□常□的 *anoncvs*)是由和 FreeBSD 附□的 CVS □用工具提供的用于和□程的 CVS 代□□同□的一□特性。 尤其是， 它允□ FreeBSD 用□不需要特殊的□限□任何一台 FreeBSD □目的官方 anoncvs 服□器□行只□的 CVS 操作。 要使用它，□□的□置 CVSROOT □境□量指向□当的 anoncvs 服□器， □入 cvs login 命令并提供广□人知的密□"anoncvs"，然后使用 cvs(1) 命令像□□任何本地□□一□来□□它。

cvs login 命令把用来登□ CVS 服□器的密□□存在□的 HOME 目□中一个叫 .cvspass 的文件里。 如果□个文件不存在， 第一次使用 cvs login 的□候可能会出□。 □□建一个空的 .cvspass 文件，然后□□重新登□。

也可以□□□ CVSup 和 *anoncvs* 服□本□上提供了同□的功能，但是有各□各□ 不同的□合可以影□用□□同□方式的□□。□□来□， CVSup 在网□□源利用方面 更加有效，而且是到目前□止在□者之中技□上更成熟的除了成本方面。要使用 CVSup，在下□任何□西之前 必□首先安装配置特定的客□端， 而且只能用于下□相当大□的 CVSup 称作 *collections*。

相比之下，anoncvs 可以通□ CVS 模□名来从□个文件里□出任何□西并□□特定的程序（比如 ls 或者 grep)。当然，anoncvs 也只□用于□ CVS □□的只□操作，所以如果□是想用和 FreeBSD □目共享的□□提供本地□□的□，CVSup 几乎是□唯一的□□。

A.4.2. 使用匿名 CVS

配置 cvs(1) 使用匿名 CVS □□可以□□的□定 CVSROOT □境□量指向 FreeBSD □目的 *anoncvs* 服□器之一。 到此□写作□止， 下面的服□器都是可用的：

- 法国: :pserver:anoncvs@anoncvs.fr.FreeBSD.org:/home/ncvs (使用 pserver 模式□， 用 cvs login 配合口令 "anoncvs" 来登□。 如果使用的是 ssh， □不需要口令。)
- 台湾地区: :pserver:anoncvs@anoncvs.tw.FreeBSD.org:/home/ncvs (使用 pserver 模式□， 用 cvs

login 配合口令 "anoncvs" 来登。 如果使用的是 ssh， 不需要口令。)

```
SSH2 HostKey: 1024 02:ed:1b:17:d6:97:2b:58:5e:5c:e2:da:3b:89:88:26
/etc/ssh/ssh_host_rsa_key.pub
SSH2 HostKey: 1024 e8:3b:29:7b:ca:9f:ac:e9:45:cb:c8:17:ae:9b:eb:55
/etc/ssh/ssh_host_dsa_key.pub
```

- 美国: anoncvs@anoncvs1.FreeBSD.org:/home/ncvs (使用 ssh ， 使用版本 2， 不需要口令。)

```
SSH2 HostKey: 2048 53:1f:15:a3:72:5c:43:f6:44:0e:6a:e9:bb:f8:01:62
/etc/ssh/ssh_host_dsa_key.pub
```

因 CVS 上允 "出" 曾存在的 (或者，某情况下将会存在) FreeBSD 源代的任意版本， 需要熟悉 cvs(1) 的版本 (-r) 参数， 以及在 FreeBSD 代中可用的。

有，修和分支。 修特指一个特定的修版本。含始是不的。 分支，一方面， 指代定定分支的最新修， 因分支不及特定的修版本， 它明天所代表的含就可能和今天的不同。

CVS 包括了用可能感趣的 修。 注意， 些并不用于 Ports Collection， 因它并不包含多个分支。

当指定一个分支，通常会得到那个分支的文件的最新版本。 如果希望得到一些旧的版本，可以用 -D date 制定一个日期。 察看 cvs(1) 手册了解更多。

A.4.3. 示例

在之前烈建通 cvs(1) 的手册， 里有一些的例子来展示如何使用匿名 CVS：

例 *45.* 从 *-CURRENT* 出些西 *(ls(1))*：

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.tw.FreeBSD.org:/home/ncvs
% cvs login
>在提示符，入任意密 “password”.
% cvs co ls
```

例 *46.* 通 *SSH* 出整个 *src/* 代：

```
% cvs -d anoncvs@anoncvs1.FreeBSD.org:/home/ncvs co src
The authenticity of host 'anoncvs1.freebsd.org (216.87.78.137)' can't be
established.
DSA key fingerprint is 53:1f:15:a3:72:5c:43:f6:44:0e:6a:e9:bb:f8:01:62.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'anoncvs1.freebsd.org' (DSA) to the list of known
hosts.
```

例 47. □出 8-STABLE 分支中的 ls(1) 版本：

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.tw.FreeBSD.org:/home/ncvs
% cvs login
在提示符□，□入任意密□ “password”。
% cvs co -rRELENG_8 ls
```

例 48. □建 ls(1) 的□化列表(用□准的 diff)

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.tw.FreeBSD.org:/home/ncvs
% cvs login
在提示符□，□入任意密□ “password”。
% cvs rdiff -u -rRELENG_8_0_0_RELEASE -rRELENG_8_1_0_RELEASE ls
```

例 49. □出可以使用的其它的模□名：

```
% setenv CVSROOT :pserver:anoncvs@anoncvs.tw.FreeBSD.org:/home/ncvs
% cvs login
在提示符□，□入任意密□ “password”。
% cvs co modules
% more modules/modules
```

A.4.4. 其他□源

下面附加的□源可能□学□ CVS 有□助：

- CVS 教程， 来自加州州立理工大学。
- CVS 主□， CVS □□和支持社区。
- CVSweb 是 FreeBSD □目的 CVS web 界面。

A.5. 使用 CTM

CTM 是保持□程目□□和中央服□器目□□同□的一□方法。 它被□□用于 FreeBSD 的源代□□，□然其他人随着□□推移会□□它可以用于其他目的。 当前几乎没有，也或者只有很少的文□□述□建 deltas 的□□， 所以如果□希望使用 CTM 去做其它事情， □□系 ctm-users-desc □件列表了解更多信息。

A.5.1. □什□我□使用 CTM?

CTM 会□□一□ FreeBSD 源代□□的本地副本。 代□□有很多的 "flavors" 可用。不管□是希望跟踪完整的 CVS □□是只是一个分支， CTM 都会□□提供信息。 如果□是 FreeBSD 上的一个活□的□□者，但是缺乏或者不存在 TCP/IP □接，或者只是希望把□化自□□送□□， CTM 就是□合□的。□于最□□的分支 □将会□天□得三个以上的 deltas。 然而，□□□考□通□□件来自□□送。 升□的大小□是保□尽可能的小。 通常小于 5K，也偶然

(十分之一可能)会有 10-50K，也不□地有个大的 100K+ 甚至更大的。

□也需要□自己了解直接和□□代□而不是□□行版本打交道的各□警告。□□情况会很□著， 如果□□□了 "current" 代□的□。□烈建□□□□和 FreeBSD 保持同□。

A.5.2. 使用 CTM 我需要做什□？

□需要□□□西：CTM 程序，□有初始的 deltas 来 feed it(□到 "current" □□)。

CTM 程序从版本 2.0 □布以来 已□是 FreeBSD 的一部分了，如果□安装了源代□副本的□， 它位于 /usr/src/usr.sbin/ctm。

□□□ CTM 的 "deltas" 可以有□□方式，FTP 或者 email。 如果□有普通的□□ Internet 的 FTP □限， 那□下面的 FTP 站点支持□□ CTM：

ftp://ftp.FreeBSD.org/pub/FreeBSD/CTM/

或者看看□一小□□像。

FTP □□相□的目□并取得 README 文件，从那里□始。

如果□希望通□ email 得到□的 deltas：

□□一个 CTM 分□列表。 ctm-src-cur-desc 支持最新的□□分支。 ctm-src-7-desc 支持 7.X □行分支，等等。。(如果□不知道如何□□□件列表， 点□上面的列表名或者到 https://lists.freebsd.org 点□□希望□□的列表。 列表□包含了所有必要的□□指□。)

当□□始接收到□□件中的 CTM 升□□，□可以使用 `ctm_rmail` 程序来解□并□用它□。 事□上如果□想要□□程以全自□的形式□行的□，□可以通□在 /etc/aliases 中□置直接使用 `ctm_rmail` 程序。 □看 `ctm_rmail` 手册□了解更多□□。

不管□使用什□方法得到 CTM deltas，□都□□□□ ctm-announce-desc □件列表。 以后会有□独的地方提交有□ CTM 系□的操作的公告。 点□上面的□件列表名并按照指示□□□件列表。

A.5.3. 第一次使用 CTM

在□□始使用 CTM delta 之前，□需要□得一个起始点。

首先□□□□定□已□有了什□。□个人都可以从一个"空"目□□始。 □必□用一个初始的 "空的" delta 来□始□的 CTM 支持□。曾□□了□的便利□些 "起始" deltas 被有意的通□ CD 来□行， 然而□在已□不□□做了。

因□代□□有数十兆字□，□□□更喜□从手□上已□有的□西□始。如果□有一□ -RELEASE CD 光□，□可以从里面□制或者解□□一□初始代□出来。 □会□省非常多的数据□□量。

□会□□□些"初始的" deltas 名字的数字后面都有个 `X` (比如 src-cur.3210XEmpty.gz)。 后面加一个 `X` 的□□符合□的初始 "seed" 的由来。 Empty 是一个空目□。通常一个基本的从 `Empty` □始的□□由 100 个 deltas □成。□便□一下，他□都很大！70 到 80 兆字□的 `gzip` □□的数据□于 XEmpty deltas 是很平常的。

一旦□已□□定了一个基本的 delta □始，□就需要比□个数高的所有的 delta。

A.5.4. 在□的日常生活中使用 CTM

要□用 deltas，□□的□入：

```
# cd /where/ever/you/want/the/stuff
# ctm -v -v /where/you/store/your/deltas/src-xxx.*
```

CTM 能□理解被 `gzip` □□的 deltas，所以□不需要先 `gunzip` 他□，□可以□省磁□空□。

除非□得整个□程非常可□， CTM 不会□及到□的代□□的。□也可以使用 `-c` □□来校□ delta， □□ CTM 就不会□及代□□； 它会只校□ delta 的完整性看看是否可以安全的用于□的当前代□□。

CTM □有其他的一些参数， □看手册□或者源代□了解更多信息。

□真的就是全部的事情了。□次得到一个新的 delta，就通□ CTM □行它来保□□的代□是最新的。

如果□些 deltas 很□重新下□的□不要□除它□。 有些□西坏掉的□候□会想到保留它□的。 即使□只有□□，也□考□使用 `fdwrite` 来做一□副本。

A.5.5. □持□本地的□□

作□一名□□者喜□□□，改□代□□中的文件。 CTM 用一□受限的方式支持本地修改：再□□文件 foo 存在之前，首先□□ foo.ctm。如果□个文件存在， CTM 会□它操作而不是 foo。

□□行□□我□提供了一□□□的方式来□持本地的改□： 只要□制□□□修改的文件并用 .ctm 的后□重新命名。然后就可以自由的修改代□了，CTM 会更新 .ctm 文件到最新版本。

A.5.6. 其他有趣的 CTM □□

A.5.6.1. 正□的□出□些将被更新

□可以□定□□列表， CTM 可以做到，在□的代□□上使用 CTM 的 `-l` □□。

□很有用如果□想要保存改□的日志， pre- 或者 post- 用各□□格□理修改的文件的□□， 或者□□是想感受一下孩子般的□狂。

A.5.6.2. 在升□前制作□□

有□□可能想□□将要被 CTM 升□所改□的所有文件。

指定 `-B backup-file` □□会□致 CTM □□将要被□定的 CTM delta 改□的所有文件到 backup-file。

A.5.6.3. 限定受升□影□的文件

有□□可能□限定一个□定的 CTM 升□的□□感□趣，也有可能想知道□□从一列 deltas 中解□□一部分文件。

□可以通□使用 `-e` 和 `-x` □□指定□□□□表□式来控制 CTM 即将□之操作的文件列表。

例如，要从□保存的CTM deltas 集里解□□出一个最新的 lib/libc/Makefile 文件，□行□个命令：

```
# cd /where/ever/you/want/to/extract/it/
# ctm -e '^lib/libc/Makefile' ~ctm/src-xxx.*
```

□于□一个在 CTM delta 中指定的文件，`-e` 和 `-x` □□按照命令行□定的□序□用。 文件只有在所有的 `-e` 和 `-x` 被□用之后□□□合格之后 才能被 CTM 操作。

A.5.7. CTM 未来的□□

其中几□：

- 在 CTM 中使用一些□□方式， □□来允□察□冒充的 CTM □丁。
- 整理 CTM 的□□， 它□□得□乱而□反直□了。

A.5.8. □□

也有一系列的 `ports` collection 的 deltas，但是人□□它的□致□没有那□高。

A.5.9. CTM □像

CTM/FreeBSD 可以在下面的□像站点通□匿名 FTP 下□。如果□□□通□匿名 FTP □取 CTM， □□着使用一个□□□近的站点。

如果有□□，□□系 ctm-users-desc □件列表。

加利福尼□州，旧金山湾区，官方源代□

- ftp://ftp.FreeBSD.org/pub/FreeBSD/development/CTM/

南非，旧的 **deltas** 的□□服□器

- ftp://ftp.za.FreeBSD.org/pub/FreeBSD/CTM/

中国台湾

- ftp://ctm.tw.FreeBSD.org/pub/FreeBSD/development/CTM/
- ftp://ctm2.tw.FreeBSD.org/pub/FreeBSD/development/CTM/
- ftp://ctm3.tw.FreeBSD.org/pub/FreeBSD/development/CTM/

如果□在□附近□不到□像或者□像不完整， □着使用搜索引□比如 alltheweb.

A.6. 使用 CVSup

A.6.1. 概述

CVSup 是一个用于从□程服□器主机上的主 CVS □□□布和升□源代□□的□件包。 FreeBSD 的源代□□□在加利福尼□州一台主□□服□器的 CVS □□里。 有了 CVSup， FreeBSD 用□可以很容易的保持他□自己的源代□□更新。

CVSup 使用所□的升□ *pull* 模式。在 pull 模式下，客□端在需要的□候向服□器端□求更新。 服□器被□

的等待客□端的升□□求。 因此所有的升□都是客□端□起的。 服□器决不会□送未□求的升□。用□必□手□□行 CVSup 客□端□取更新， 或者□置一个 cron 作□来□它以固定的□律自□□行。

□□ CVSup用大写字母写正是表示， 代表了完整的□件包。 它的主要□件是□行在□个用□机器上的客□端 cvsup， 和□行在□个 FreeBSD □像站点上的服□器端 cvsupd。

当□□□ FreeBSD 文□和□件列表□，□可能会看□ sup。 Sup 是 CVSup 的前身，有着相似的目的。 CVSup 使用很多和 sup 相同的方式， 而且， 它□是用使用和 sup 的兼容的配置文件。 Sup 已□不再被 FreeBSD □目使用了， 因□ CVSup 既快又有更好的□活性。

csup 是用 C □言□ CVSup □件的重写。 它最大的好□是， □个程序更快一些，并且也不需要依□于 Modula-3 □言， 因此也就不需要安装后者。 □外，□可以直接使用它， 因□它是基本系□的一部分。 假如□决定使用 csup， □可以跳□安装 CVSup □一□， 并在文章中余下部分提到的 CVSup 改□ csup。

A.6.2. 安装

安装 CVSup 最□□的方式就是使用 FreeBSD packages collection 中□□□的 net/cvsup 包。 如果□想从源代□□建 CVSup， □可以使用 net/cvsup port。但是□先警告一下： net/cvsup port 依□于 Modula-3 系□，会花□相当的□□和磁□空□来下□□□。

如果想在没有安装 Xorg 的□算机， 例如服□器上使用 CVSup， □只能使用不包含 CVSupGUI 的 net/cvsup-without-gui。

A.6.3. CVSup 配置

CVSup 的操作被一个叫做 supfile 的配置文件所控制。 在目□ /usr/shared/examples/cvsup/ 下面有一些示例的 supfiles。

supfile 中的信息解答了 CVSup 下面的几个□□：

- □想接收 □些文件？
- □想要它□的 □个版本？
- □想从□里 □取它□？
- □想把它□ 放在□自己机器的什□地方？
- □想把 □的状□文件放在□？

在下面的章□里，我□通□依次回答□些□□来□建一个典型的 supfile 文件。首先，我□描述一下 supfile 的整体□成。

supfile 是个文本文件。注□用 # □□，至行尾有效。 空行和只包含注□的行会被忽略。

□个保留行描述一批用□希望接收的文件。 □行以 "collection"， 由服□器端定□的合理的文件分□，的名字□□。collection 的名字告□服□器□想要的文件。 collection 名字□束或者有更多的字段，用空格分隔。□些字段回答了上面列出的□□。 字段□型有□□：□□字段和□字段。 □□字段由独立的□□字□成，比如， delete 或者 compress。□字段也用□□字□□， □□字后面跟 = 和第二个□而没有空格。 例如，release=cvs 是一个□字段。

一个典型的 supfile 往往接收多于一个的 collection。□建 supfile 的一□方式是明□的□□一个 collection 指定相□的字段。然而，□□使得 supfile 的行□得特□□，很不方便， 因□ supfile 中的所有 collection 的大部分字段都是相同的。 CVSup 提供了一个默□机制来避免 □些□□。用特定的□ collection 名 *default □□的行可以被用来□置□□和□□ supfile 中随后的 collection 中的默□□。 默□□可以通□□□个 collection 自身指定不同的□来□□个的 collection 覆□□置， 也可以在 mid-supfile 中通□附加的 *default 行改□或□充。

知道了□些，我□□在就可以□始□建一个 用于接收和升□ FreeBSD-CURRENT 主源代□□的 supfile 文件了。

- □想接收□些文件？

  通□ CVSup 可用的文件□□成叫做 "collections" 的名称□。 □些可用的 collection 在 随后的章□ 中描述。在□个例子里， 我□希望接收 FreeBSD 系□的完整的主代□□。 有一个□独的大的 collection src-all □我□完成□个。 □建我□的 supfile 的第一□， 我□□□的列出□些 collection，□个一行(在□个例子里， 只有一行)：

  ```
  src-all
  ```

- □想要他□的 □个版本？

  通□ CVSup，□□□上可以接收 曾□存在的源代□的任何版本。 □是有可能的，因□ cvsupd 服□器直接通□ CVS □□工作，那包含了所有的版本。□可以 用 tag= 和 date= □字段 指定一个□想要的版本。

  仔□的正□指定任何 tag= 字段。有一些 tag 只□特定的 collection 文件合法。 如果□指定了一个不正□的或者 □写□□的 tag，CVSup 会□除□可能不想□除的文件。 特□地，□ ports-* collection 只使用 tag=.。

  tag= 字段在□□中表示□一个符号□□。 有□□□□，修□□□和分支□□。 修□□□代表一个特定的修□版本。 它的含□是一成不□的。 分支□□，□一方面，代表□定□□□上□定□□的最新修□。 因□分支□□不代表一个特定的修□版本， 它明天的含□就可能和今天的有所不同。

  CVS □□ 包含了用□可能感□趣的分支□□。 当在 CVSup 的配置文件中指定□□的□候，必□用 tag= □□(RELENG_8 会□成 tag=RELENG_8)。 □住只有 tag=. 可以用于 Ports Collection。

  注意像看到的那□正□的□入□□名。 CVSup 不能辨□合法和不合法□□。 如果□□写□了□□名， CVSup 会像□指定了一个没有任何文件的合法□□一□工作， 那会□除□已□存在的代□。

  当□指定一个分支□□的□候，□通常会收到□□□上文件的最新版本。 如果□希望接收一些□□的版本，□可以通□用 date= □字段指定一个日期来做到。 cvsup(1) 手册□解□了如何来做。

  □于我□的示例来□，我□希望接收 FreeBSD-CURRENT。 我□在我□的 supfile 的□□添加□行：

  ```
  *default tag=.
  ```

  有一个重要的特例， 如果□既没指定 tag= 字段也没指定 date= 字段的情况。□□情况下，□会收到直接来自于服□器 CVS □□的真□的 RCS 文件， 而不是某一特定版本。 □□人□一般喜□□□操作模式。通□在他□的系□上□□一□□□自身的副本， 他□可以□□修□□史以及□□文件□去的版本。 然而，□个好

□是以大量的磁□空□□代价的。

- □想从□里□取他□？

  我□使用 host= 字段来告□ cvsup 从□里□取更新。 任何一个 CVSup □像站点都可以， □然□□□□□一个□□比□近的站点。 在□个例子里我□将使用一个虚□的 FreeBSD □布站点， cvsup99.FreeBSD.org：

  ```
  *default host=cvsup99.FreeBSD.org
  ```

  □需要在□行 CVSup 之前把□个改成一个□□存在的站点。 在任何 cvsup □行的特定□刻，□都可以在命令行上使用 -h hostname □□来覆□主机□置。

- □想把它□放在 □自己机器的什□地方？

  prefix= 字段告□ cvsup 把接收的文件放在□里。 在□个例子里，我□把源代□文件直接放□我□的主源代□□，/usr/src。 src 目□已□□含在我□□□接收的 collection 里了， 所以正□的写法是：

  ```
  *default prefix=/usr
  ```

- cvsup 在□里□□它的状□文件？

  CVSup 客□端在被叫做 "base" 的目□里□□了几个状□文件。 □些文件□助 CVSup 更有效的工作， 通□跟踪□已□接收到□些更新的方式。 我□将使用□准的 base 目□， /var/db：

  ```
  *default base=/var/db
  ```

  如果□的 base 目□□不存在，□在最好□建它。 如果 base 目□不存在，cvsup 客□端会拒□工作。

- 其他的 supfile □置：

  在 supfile 中有一些其他□□需要介□一下：

  ```
  *default release=cvs delete use-rel-suffix compress
  ```

  release=cvs □示服□器□□从 FreeBSD 的主 CVS □□中□取信息。 事□上□是□□的，但是也有可能会超出□个□□的□□。

  delete □ CVSup □限□除文件。 □□□□是指定□个，□□ CVSup 可以保□□的源代□□完全更新。CVSup 很小心的只□除那些不再依□的文件。 □□有的任何□外的文件会被□格的保留。

  use-rel-suffix 是 ... 不可思□的。 如果□真的想了解它，□看 cvsup(1) 手册□。 否□，就指定而不用担心□个。

  compress □用 gzip □格的信道□□。 如果□的网□□接是 T1 或者更快， □可能不想使用□□。 否□，它非常有□助。

- 把它□放在一起：

□是我□的示例的完整 supfile 文件：

```
*default tag=.
*default host=cvsup99.FreeBSD.org
*default prefix=/usr
*default base=/var/db
*default release=cvs delete use-rel-suffix compress

src-all
```

A.6.3.1. refuse 文件

像上面提到的，CVSup 使用一□ *pull* 方法。基本上，□意味着□要□接到 CVSup 服□器，服□器□， "□有些□能下□的□西 ..."，然后□的客□端反□"好，我要□个， □个，□个，□有□个。"在默□的配置中， CVSup 客□端会取回□在配置文件中□定的 collection 和□□的□个文件。 然而，并不□是□想要的， 尤其是□在同□ doc，ports，或者 www □ - 大部分人都不能□□四□或者五□ □言，因此他□不需要下□特定□言的文件。 如果□在 CVSup Ports Collection，□ 可以通□□独指定□个 collection 来避免□个 (比如，*ports-astrology*， *ports-biology*，等等取代□□的□明 *ports-all*)。然而，因□ doc 和 www □没有特定□言的 collection，□必□ 使用 CVSup □多□好的特性之一： refuse 文件。

refuse 文件本□上是告□ CVSup 它不□□从 collection 中取得某些文件；□句□□，它告□客□端 拒□ 来自服□器的特定的文件。 refuse 文件可以在 base/sup/ 中□到(或者，如果□没有，□□□建一个)。 *base* 在□的 supfile 中定□； 默□情况下，*base* 就是 /var/db， □意味着默□的 refuse 文件就是 /var/db/sup/refuse。

refuse 文件的格式很□□； 它□□包含□不希望下□的文件和目□名。 例如，如果□除了英□和德□之外不会□其他□言， 而且也不打算□□德文的文□翻□版本， □可以把下面□些放在□的 refuse 文件里：

```
doc/bn_*
doc/da_*
doc/de_*
doc/el_*
doc/es_*
doc/fr_*
doc/hu_*
doc/it_*
doc/ja_*
doc/mn_*
doc/nl_*
doc/no_*
doc/pl_*
doc/pt_*
doc/ru_*
doc/sr_*
doc/tr_*
doc/zh_*
```

等等其他□言(□可以通□□□ FreeBSD CVS □□□到完整的列表)。

有□个非常有用的特性，那些慢速□接或者要□他□的 Internet □接按□付□的用□就可以□省宝□的□□因□他□不再需要 下□那些从来不用的文件。要了解 refuse 文件的更多信息以及其它 CVSup 的□雅的特性，□□□它的手册□。

A.6.4. □行 CVSup

□□在准□□□升□了。命令很□□：

```
# cvsup supfile
```

supfile 的位置当然就是□□□□建的 supfile 文件名□。 如果□在 X11 下面□行，cvsup 会□示一个有一些可以做平常事情的按□的 GUI □口。 按 **[go]** 按□，然后看着它□行。

在□个例子里□将要升□□目前的 /usr/src □，□将需要 用 root 来□行程序，□□ cvsup 有需要的□限来更新□的文件。 □□□建了□的配置文件，又从来没有使用□□个程序， □□不安是可以理解的。有一个□□的方法不改□□当前的文件 来做一次□□性的□行。只要在方便的地方□建一个 空目□，并在命令行上作□一个□外的参数□明：

```
# mkdir /var/tmp/dest
# cvsup supfile /var/tmp/dest
```

□指定的目□会作□所有文件更新的目的路径。 CVSup 会□□□在 /usr/src 中的文件，但是不会修改或□除。任何文件更新都会被放到 /var/tmp/dest/usr/src 里了。 在□□方式下□行 CVSup 也会把它的 base 目□状□文件保持原□。□些文件的新版本 会被写到指定的目□。 因□□有 /usr/src 目□的□□限，所以□行□□□□性的□行甚至不需要使用 root 用□。

如果□没有□行 X11 或者不喜□ GUI， 当□□行 cvsup 的□候需要在命令行添加 □个□□：

```
# cvsup -g -L 2 supfile
```

-g 告□ CVSup 不要使用 GUI。如果□ 没在□行 X11 □个是自□的，否□□必□指定它。

-L 2 告□ CVSup □出所有正在升□的文件的□□。 有三个等□可以□□，从 -L 0 到 -L 2。默□是 0，意味着除了□□消息 什□都不□出。

□有□多其它的□□可用。想要一个□短的列表， □入 cvsup -H。要□看更□□的描述， □□看手册□。

一旦□□升□工作的方式□意了，□就 可以使用 cron(8) 来安排□□的□行 CVSup。 很□然的，□不□□□ CVSup 通□ cron(8) □行的□候使用它的 GUI。

A.6.5. CVSup 文件 collection

CVSup 可用的文件 collection 是分□□□的。 有几个大的 collection，然后它□有分成更小的子 collection。接收一个大的 collection 等同于 接收它的□一个子 collection。 collection 的等□□系在下面列表中通□□□的使用 反映出来。

最常用的 collection 是 src-all，和 ports-all。其它的 collection 只被有着特定 目的的小部分人使用，

有些站点可能不全部支持。

`cvs-all release=cvs`

FreeBSD 主 CVS □□，包含 密□系□的代□。

`distrib release=cvs`

FreeBSD □行版本和□像相□的 文件。

`doc-all release=cvs`

FreeBSD 使用手册和其它文□的源代□。 其中不包含 FreeBSD web 站点的文件。

`ports-all release=cvs`

FreeBSD Ports Collection。

如果□不想升□全部的 `ports-all`(整个 ports □)， 而只是使用下面列出的一个子集，□□保□□是升□了 `ports-base` 子 collection！ 无□何□在 ports □建下□□造有所改□的□候都会通□ `ports-base` 表□出来，事□上某些 改□会很快的被"□□的" ports 使用，因此，如果□只升□了 "□□的" ports 而他□使用了一些新的特性， 就有□大的可能□□会因一些神秘的□□信息而失□。 □□情况下非常快速的要做的事情 就是□保□的 `ports-base` 子 collection 更新到 最新。

要自行□建 ports/INDEX， □ 必□ 接受 `ports-all` (完整的 ports tree)。 在部分 ports tree 上□建 ports/INDEX 是不被支持的。 □参□ FAQ。

`ports-accessibility release=cvs`

用以□助残疾用□的□件。

`ports-arabic release=cvs`

阿拉伯□支持。

`ports-archivers release=cvs`

存□工具。

`ports-astro release=cvs`

天文相□的 ports。

`ports-audio release=cvs`

声音支持。

`ports-base release=cvs`

Ports Collection □建下部□造 - 位于 /usr/ports 的 Mk/ 和 Tools/ 子目□的 各□各□的文件。

□□看重要警告：□□□ □是更新□个 子 collection，无□□更新 FreeBSD Ports Collection 的任何部分的□候！

`ports-benchmarks release=cvs`

基准。

`ports-biology release=cvs`

生物学。

`ports-cad release=cvs`

□算机□助□□工具。

`ports-chinese release=cvs`

中文□言支持。

`ports-comms release=cvs`

通信□件。

`ports-converters release=cvs`

字符□□□□。

`ports-databases release=cvs`

数据□

`ports-deskutils release=cvs`

□算机□明前常出□在□面上的□西。

`ports-devel release=cvs`

□□工具。

`ports-dns release=cvs`

DNS 相□□件。

`ports-editors release=cvs`

□□器

`ports-emulators release=cvs`

其它操作系□的模□器

`ports-finance release=cvs`

□□，金融相□□用程序。

`ports-ftp release=cvs`

FTP 客□端和服□器端工具。

`ports-games release=cvs`

游□

`ports-german release=cvs`

德□支持。

`ports-graphics release=cvs`

□形□像工具。

`ports-hebrew release=cvs`
: 希伯来□支持。

`ports-hungarian release=cvs`
: 匈牙利□言支持。

`ports-irc release=cvs`
: Internet 多□交□(IRC)工具。

`ports-japanese release=cvs`
: 日□支持。

`ports-java release=cvs`
: Java™ 工具。

`ports-korean release=cvs`
: □国□言支持。

`ports-lang release=cvs`
: □程□言。

`ports-mail release=cvs`
: □件□件。

`ports-math release=cvs`
: 数□□算□件。

`ports-misc release=cvs`
: □□工具。

`ports-multimedia release=cvs`
: 多媒体□件。

`ports-net release=cvs`
: 网□□件。

`ports-net-im release=cvs`
: 即□消息□件。

`ports-net-mgmt release=cvs`
: 网管□件。

`ports-net-p2p release=cvs`
: □等网 (peer to peer network) □用。

`ports-news release=cvs`
: USENET 新□□件。

`ports-palm release=cvs`

Palm™ 系列□件支持。

`ports-polish release=cvs`

波□□支持。

`ports-ports-mgmt release=cvs`

用于管理 ports 和□□□包的工具。

`ports-portuguese release=cvs`

葡萄牙□支持。

`ports-print release=cvs`

打印□件。

`ports-russian release=cvs`

俄□支持。

`ports-science release=cvs`

科学□算。

`ports-security release=cvs`

安全工具。

`ports-shells release=cvs`

命令行 shell。

`ports-sysutils release=cvs`

系□□用工具。

`ports-textproc release=cvs`

文本□理工具(不 包含□面出版)。

`ports-ukrainian release=cvs`

□克□□支持。

`ports-vietnamese release=cvs`

越南□支持。

`ports-www release=cvs`

万□网(WWW)相□□件。

`ports-x11 release=cvs`

支持 X window 系□的 ports。

`ports-x11-clocks release=cvs`

X11 □□。

`ports-x11-drivers release=cvs`
: X11 □□程序。

`ports-x11-fm release=cvs`
: X11 文件管理器。

`ports-x11-fonts release=cvs`
: X11 字体和字体工具。

`ports-x11-toolkits release=cvs`
: X11 工具包。

`ports-x11-servers release=cvs`
: X11 服□器。

`ports-x11-themes release=cvs`
: X11 主□。

`ports-x11-wm release=cvs`
: X11 □口管理器。

`projects-all release=cvs`
: FreeBSD 内部□目的代□□。

`src-all release=cvs`
: FreeBSD 主代□，包含密□系□的代□。

`src-base release=cvs`
: /usr/src □□的各式各□的文件。

`src-bin release=cvs`
: □用□模式下可能用到的用□工具 (/usr/src/bin)。

`src-cddl release=cvs`
: 采用了 CDDL 授□的□用工具和函数□ (/usr/src/cddl)。

`src-contrib release=cvs`
: FreeBSD □目之外的工具和□，通常在 FreeBSD 中不作修改 (/usr/src/contrib)。

`src-crypto release=cvs`
: FreeBSD □目之外的 密□系□工具和□，通常在 FreeBSD 中不作修改 (/usr/src/crypto)。

`src-eBones release=cvs`
: Kerberos 和 DES (/usr/src/eBones)。 目前的 FreeBSD 中不再使用使用。

`src-etc release=cvs`
: 系□配置文件 (/usr/src/etc)。

`src-games release=cvs`

游□ (/usr/src/games)。

`src-gnu release=cvs`

GNU 公共□可□□的工具 (/usr/src/gnu)。

`src-include release=cvs`

□文件 (/usr/src/include)。

`src-kerberos5 release=cvs`

Kerberos5 安全包 (/usr/src/kerberos5)。

`src-kerberosIV release=cvs`

KerberosIV 安全包 (/usr/src/kerberosIV)。

`src-lib release=cvs`

□ (/usr/src/lib)。

`src-libexec release=cvs`

通常被其它程序□用的系□程序 (/usr/src/libexec)。

`src-release release=cvs`

生成 FreeBSD 版本必需的文件 (/usr/src/release)。

`src-rescue release=cvs`

用于□急修□的静□□□的程序； □参□ rescue(8) (/usr/src/rescue)。

`src-sbin release=cvs`

□用□模式的系□工具 (/usr/src/sbin)。

`src-secure release=cvs`

密□相□□和命令 (/usr/src/secure)。

`src-share release=cvs`

跨多个平台的共享的文件 (/usr/src/share)。

`src-sys release=cvs`

内核 (/usr/src/sys)。

`src-sys-crypto release=cvs`

内核密□系□代□ (/usr/src/sys/crypto)。

`src-tools release=cvs`

□□ FreeBSD 的各□各□的工具 (/usr/src/tools)。

`src-usrbin release=cvs`

用□工具 (/usr/src/usr.bin)。

`src-usrsbin release=cvs`

系□工具 (/usr/src/usr.sbin)。

`www release=cvs`

FreeBSD WWW 站点的源代□。

`distrib release=self`

CVSup 服□器的 配置文件。用于 CVSup □像站点。

`gnats release=current`

GNATS bug 跟踪数据□。

`mail-archive release=current`

FreeBSD □件列表存□。

`www release=current`

□□理□的 FreeBSD WWW 站点文件(不是源文件)。 用于 WWW □像站点。

A.6.6. 更多信息

CVSup FAQ 以及□于 CVSup 的其他信息， □□看 CVSup 主□。

多数与 FreeBSD 有□的 CVSup □□会在 FreeBSD 技□□□□件列表 □行。 □个□件的新版本会在那里和 FreeBSD 公告□件列表 公布。

如果□于 CVSup 有任何□□， 或希望提交 bug □告， □参□ CVSup FAQ。

A.6.7. CVSup 站点

FreeBSD 的 CVSup 服□器□行于 下列站点：

A.7. CVS □□

当使用 cvs 或者 CVSup □取和升□源代□的□候，必□指定一个修□□。 修□□代表 FreeBSD □□的一个特定分支， 或者一个特定的□□点。第一□叫做 "分支□□"，第二□叫做 "版本□□"。

A.7.1. 分支□□

所有□些，除了 `HEAD` (□个□是 合法□□)以外，只□用于 src/ □。ports/，doc/，和 www/ □没有分支。

HEAD

主□的符号名，或者□ FreeBSD-CURRENT。 当没有指定修□版本的□候也是默□的。

在 CVSup 里，□个□□通□ 一个 `.` 来反映出来(不是□点，而是一个 `.` 字符)。

在 CVS 里，当没有修□□□指定□□是默□的。 在一台 STABLE 机器上□出或者升□到 CURRENT 源代□ 通常不是 一个好主意，除非□是□的本意。

RELENG_8

□是 FreeBSD-8.X 的□□分支， 也被称作 FreeBSD 8-STABLE。

RELENG_8_2

□是 FreeBSD-8.2 □行版分支，只用于安全公告， 以及其他重要更新。

RELENG_8_1

FreeBSD-8.1 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_8_0

FreeBSD-8.0 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_7

□是 FreeBSD-7.X 的□□分支， 也被称作 FreeBSD 7-STABLE。

RELENG_7_4

FreeBSD-7.4 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_7_3

FreeBSD-7.3 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_7_2

FreeBSD-7.2 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_7_1

FreeBSD-7.1 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_7_0

FreeBSD-7.0 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_6

□是 FreeBSD-6.X 的□□分支， 也被称作 FreeBSD 6-STABLE。

RELENG_6_4

FreeBSD-6.4 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_6_3

FreeBSD-6.3 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_6_2

FreeBSD-6.2 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_6_1

FreeBSD-6.1 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_6_0

FreeBSD-6.0 的□行版分支， 只用于安全公告， 以及其他重要更新。

RELENG_5

□是 FreeBSD-5.X 的□□分支， 也被称作 FreeBSD 5-STABLE。

RELENG_5_5

FreeBSD-5.5 安全分支。 只被安全公告和其它重要更新使用。

RELENG_5_4

FreeBSD-5.4 安全分支。 只被安全公告和其它重要更新使用。

RELENG_5_3

FreeBSD-5.3 安全分支。 只被安全公告和其它重要更新使用。

RELENG_5_2

□□ FreeBSD-5.2 和 FreeBSD-5.2.1 的安全分支， 只被安全公告和其它重要更新使用。

RELENG_5_1

□□ FreeBSD-5.1 的□行版本分支， 只被安全公告和其它重要更新使用。

RELENG_5_0

□□ FreeBSD-5.0 的□行版本分支， 只被安全公告和其它重要更新使用。

RELENG_4

FreeBSD-4.X □□□，也被叫做 FreeBSD-STABLE。

RELENG_4_11

FreeBSD-4.11 安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_10

FreeBSD-4.10 安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_9

FreeBSD-4.9 安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_8

FreeBSD-4.8 安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_7

FreeBSD-4.7 安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_6

FreeBSD-4.6 和 4.6.2 的安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_5

FreeBSD-4.5 安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_4

FreeBSD-4.4 安全分支。 只被安全公告和其它重要更新使用。

RELENG_4_3

FreeBSD-4.3 安全分支。 只被安全公告和其它重要更新使用。

RELENG_3

FreeBSD-3.X 的□□□，也被叫做 3.X-STABLE。

RELENG_2_2

FreeBSD-2.2.X 的□□□， 也被叫做 2.2-STABLE。 □个分支基本上已□□□了。

A.7.2. 版本□□

当一个特定的 FreeBSD 版本□行□， □些□□代表了一个指定的□□点。□布工程□程在 Release Engineering Information 和 Release Process 文□中被□□描述。 src □使用以 RELENG_ □□的□□。 ports 和 doc □使用以 RELEASE □□的□□。 最后， www □上不会有任何特定□行版的□□。

RELENG_8_2_0_RELEASE

FreeBSD 8.2

RELENG_8_1_0_RELEASE

FreeBSD 8.1

RELENG_8_0_0_RELEASE

FreeBSD 8.0

RELENG_7_4_0_RELEASE

FreeBSD 7.4

RELENG_7_3_0_RELEASE

FreeBSD 7.3

RELENG_7_2_0_RELEASE

FreeBSD 7.2

RELENG_7_1_0_RELEASE

FreeBSD 7.1

RELENG_7_0_0_RELEASE

FreeBSD 7.0

RELENG_6_4_0_RELEASE

FreeBSD 6.4

RELENG_6_3_0_RELEASE

FreeBSD 6.3

RELENG_6_2_0_RELEASE

FreeBSD 6.2

RELENG_6_1_0_RELEASE
: FreeBSD 6.1

RELENG_6_0_0_RELEASE
: FreeBSD 6.0

RELENG_5_5_0_RELEASE
: FreeBSD 5.5

RELENG_5_4_0_RELEASE
: FreeBSD 5.4

RELENG_4_11_0_RELEASE
: FreeBSD 4.11

RELENG_5_3_0_RELEASE
: FreeBSD 5.3

RELENG_4_10_0_RELEASE
: FreeBSD 4.10

RELENG_5_2_1_RELEASE
: FreeBSD 5.2.1

RELENG_5_2_0_RELEASE
: FreeBSD 5.2

RELENG_4_9_0_RELEASE
: FreeBSD 4.9

RELENG_5_1_0_RELEASE
: FreeBSD 5.1

RELENG_4_8_0_RELEASE
: FreeBSD 4.8

RELENG_5_0_0_RELEASE
: FreeBSD 5.0

RELENG_4_7_0_RELEASE
: FreeBSD 4.7

RELENG_4_6_2_RELEASE
: FreeBSD 4.6.2

RELENG_4_6_1_RELEASE
: FreeBSD 4.6.1

RELENG_4_6_0_RELEASE
FreeBSD 4.6

RELENG_4_5_0_RELEASE
FreeBSD 4.5

RELENG_4_4_0_RELEASE
FreeBSD 4.4

RELENG_4_3_0_RELEASE
FreeBSD 4.3

RELENG_4_2_0_RELEASE
FreeBSD 4.2

RELENG_4_1_1_RELEASE
FreeBSD 4.1.1

RELENG_4_1_0_RELEASE
FreeBSD 4.1

RELENG_4_0_0_RELEASE
FreeBSD 4.0

RELENG_3_5_0_RELEASE
FreeBSD-3.5

RELENG_3_4_0_RELEASE
FreeBSD-3.4

RELENG_3_3_0_RELEASE
FreeBSD-3.3

RELENG_3_2_0_RELEASE
FreeBSD-3.2

RELENG_3_1_0_RELEASE
FreeBSD-3.1

RELENG_3_0_0_RELEASE
FreeBSD-3.0

RELENG_2_2_8_RELEASE
FreeBSD-2.2.8

RELENG_2_2_7_RELEASE
FreeBSD-2.2.7

RELENG_2_2_6_RELEASE

FreeBSD-2.2.6

RELENG_2_2_5_RELEASE

FreeBSD-2.2.5

RELENG_2_2_2_RELEASE

FreeBSD-2.2.2

RELENG_2_2_1_RELEASE

FreeBSD-2.2.1

RELENG_2_2_0_RELEASE

FreeBSD-2.2.0

A.8. AFS 站点

FreeBSD 的 AFS 服□器□行于下面的站点：

瑞典

文件的路径是： /afs/stacken.kth.se/ftp/pub/FreeBSD/

```
stacken.kth.se          # Stacken Computer Club, KTH, Sweden
130.237.234.43          #hot.stacken.kth.se
130.237.237.230         #fishburger.stacken.kth.se
130.237.234.3           #milko.stacken.kth.se
```

□□者 ftp@stacken.kth.se

A.9. rsync 站点

下面的站点□ FreeBSD 可以通□ rsync □□下□。 rsync □用程序和 rcp(1) 的工作方式很相像， 但是有更多的□□，使用 rsync □程更新□□只□□ □□文件的不同之□， 因此能□大幅度的提高网□同□速率。 如果□是 FreeBSD FTP 服□器或者 CVS □□的□像站点， □一点非常有用。 rsync 套件可以工作在□多□ 操作系□上，在 FreeBSD 上，□看 net/rsync port 或者使用 package。

捷克共和国

rsync://ftp.cz.FreeBSD.org/

可用的 collection：

- ftp： FreeBSD FTP 服□器的 部分□像。
- FreeBSD： FreeBSD FTP 服□器的 完整□像。

荷□

rsync://ftp.nl.FreeBSD.org/

可用的 collection：

- FreeBSD: □于 FreeBSD FTP 服□器的完整□像。

俄□斯

rsync://ftp.mtu.ru/

可用的 collections:

- FreeBSD: 完整的 FreeBSD FTP 服□器□像。
- FreeBSD-gnats: GNATS □□追踪数据□。
- FreeBSD-Archive: FreeBSD □案的 FTP 服□器□像。

瑞典

rsync://ftp4.se.freebsd.org/

可用的 collections：

- FreeBSD： FreeBSD FTP 服□器的完整□像。

台湾地区 (中国)

rsync://ftp.tw.FreeBSD.org/

rsync://ftp2.tw.FreeBSD.org/

rsync://ftp6.tw.FreeBSD.org/

可用的 collection：

- FreeBSD： FreeBSD FTP 服□器的完整□像。

英国

rsync://rsync.mirrorservice.org/

可用的 collection：

- sites/ftp.FreeBSD.org: FreeBSD FTP 服□器 的完整□像。

美国

rsync://ftp-master.FreeBSD.org/

服□器只供 FreeBSD 主□像站点使用。

可用的 collection：

- FreeBSD：FreeBSD FTP 服□器的主要存□。
- acl：FreeBSD 主 ACL 列表。

  rsync://ftp13.FreeBSD.org/

  可用的 collection：

- FreeBSD：FreeBSD FTP 服□器的完整 □像。

附□ B: 参考文献

尽管手册□能□提供□于 FreeBSD 操作系□最□□威的参考□料， 它□有□却不能告□我□如何□整个系□很好地□□起来。 因此， 一本□于 UNIX® 系□管理的好□， 以及一□好的用□手册是不可或缺的。

B.1. □于 FreeBSD 的□□□籍与□志

非英文的□籍和□志：

- FreeBSD 入□与□用 (繁体中文)， 出版商： Drmaster， 1997. ISBN 9-578-39435-7.
- FreeBSD 技□内幕 (□体中文□本)， 机械工□出版社。 ISBN 7-111-10201-0。
- FreeBSD 使用大全 第一版 （□体中文）， 机械工□出版社。ISBN 7-111-07482-3。
- FreeBSD 使用大全 第二版 （□体中文）， 机械工□出版社。ISBN 7-111-10286-X。
- FreeBSD Handbook (第二版□体中文□本)， 人民□□出版社。ISBN 7-115-10541-3。
- FreeBSD 3.x Internet 高□服□器的架□与管理 （□体中文）， 清□大学出版社。ISBN 7-900625-66-6。
- FreeBSD & Windows 集成□网□□ （□体中文）， 中国□道出版社。 ISBN 7-113-03845-X。
- FreeBSD 网站架□□□ （□体中文）， 中国□道出版社。ISBN 7-113-03423-3。
- FreeBSD for PC 98’ers (日文, 出版商：SHUWA System Co, LTD. ISBN 4-87966-468-5 C3055 P2900E。
- FreeBSD (日文, 出版商：CUTT. ISBN 4-906391-22-2 C3055 P2400E.
- Complete Introduction to FreeBSD （日文）, 出版商：Shoeisha Co., Ltd. ISBN 4-88135-473-6 P3600E.
- Personal UNIX Starter Kit FreeBSD （日文）, 出版商：ASCII. ISBN 4-7561-1733-3 P3000E.
- FreeBSD Handbook （日文□本）, 出版商：ASCII. ISBN 4-7561-1580-2 P3800E.
- FreeBSD mit Methode （德文）, 出版商：Computer und Literatur Verlag/Vertrieb Hanser, 1998. ISBN 3-932311-31-0.
- FreeBSD 4 - Installieren, Konfigurieren, Administrieren （德文）， 出版商：Computer und Literatur Verlag, 2001. ISBN 3-932311-88-4.
- FreeBSD 5 - Installieren, Konfigurieren, Administrieren （德文）， 出版商： Computer und Literatur Verlag, 2003. ISBN 3-936546-06-1.
- FreeBSD de Luxe (德文), 出版商： Verlag Modere Industrie, 2003. ISBN 3-8266-1343-0.
- FreeBSD Install and Utilization Manual （日文）, 出版商： Mainichi Communications Inc.， 1998. ISBN 4-8399-0112-0.
- Onno W Purbo, Dodi Maryanto, Syahrial Hubbany, Widjil Widodo *Building Internet Server with FreeBSD* （印尼文）, 出版商： Elex Media Komputindo.
- Absolute BSD: The Ultimate Guide to FreeBSD （繁体中文） 出版商： GrandTech Press, 2003. ISBN 986-7944-92-5.
- The FreeBSD 6.0 Book （繁体中文）， 出版商：Drmaster, 2006. ISBN 9-575-27878-X.

英文版的□籍和□志：

- Absolute FreeBSD, 2nd Edition: The Complete Guide to FreeBSD, 出版商： No Starch Press, 2007. ISBN: 978-1-59327-151-0
- The Complete FreeBSD, 出版商： O'Reilly, 2003. ISBN: 0596005164
- The FreeBSD Corporate Networker's Guide, 出版商： Addison-Wesley, 2000. ISBN: 0201704811
- FreeBSD: An Open-Source Operating System for Your Personal Computer, 出版商： The Bit Tree Press, 2001. ISBN: 0971204500
- Teach Yourself FreeBSD in 24 Hours, 出版商： Sams, 2002. ISBN: 0672324245
- FreeBSD 6 Unleashed, 出版商： Sams, 2006. ISBN: 0672328755
- FreeBSD: The Complete Reference, 出版商： McGrawHill, 2003. ISBN: 0072224096
- BSD Magazine, 出版商： Software Press Sp. z o.o. SK. ISSN 1898-9144

B.2. 用□指南

- Computer Systems Research Group, UC Berkeley. *4.4BSD User's Reference Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-075-9
- Computer Systems Research Group, UC Berkeley. *4.4BSD User's Supplementary Documents*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-076-7
- *UNIX in a Nutshell*. O'Reilly & Associates, Inc., 1990. ISBN 093717520X
- Mui, Linda. *What You Need To Know When You Can't Find Your UNIX System Administrator*. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-104-6
- Ohio State University □写了一□ UNIX 入□教程 并提供了在□的 HTML 和 PostScript 格式的版本。
- □□文□的意大利文 翻□ 是 FreeBSD Italian Documentation Project 的一部分。
- Jpman Project, Japan FreeBSD Users Group. FreeBSD User's Reference Manual (日文□本). Mainichi Communications Inc., 1998. ISBN4-8399-0088-4 P3800E.
- Edinburgh University has written an Online Guide for newcomers to the UNIX environment.

B.3. 管理□指南

- Albitz, Paul and Liu, Cricket. *DNS and BIND*, 4th Ed. O'Reilly & Associates, Inc., 2001. ISBN 1-59600-158-4
- Computer Systems Research Group, UC Berkeley. *4.4BSD System Manager's Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-080-5
- Costales, Brian, et al. *Sendmail*, 2nd Ed. O'Reilly & Associates, Inc., 1997. ISBN 1-56592-222-0
- Frisch, Æleen. *Essential System Administration*, 2nd Ed. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-127-5
- Hunt, Craig. *TCP/IP Network Administration*, 2nd Ed. O'Reilly & Associates, Inc., 1997. ISBN 1-56592-322-7
- Nemeth, Evi. *UNIX System Administration Handbook*. 3rd Ed. Prentice Hall, 2000. ISBN 0-13-020601-6

- Stern, Hal *Managing NFS and NIS* O'Reilly & Associates, Inc., 1991. ISBN 0-937175-75-7
- Jpman Project, Japan FreeBSD Users Group. FreeBSD System Administrator's Manual (日文□本). Mainichi Communications Inc., 1998. ISBN4-8399-0109-0 P3300E.
- Dreyfus, Emmanuel. Cahiers de l'Admin: BSD 2nd Ed. (in French), Eyrolles, 2004. ISBN 2-212-11463-X

B.4. □□指南

- Asente, Paul, Converse, Diana, and Swick, Ralph. *X Window System Toolkit*. Digital Press, 1998. ISBN 1-55558-178-1
- Computer Systems Research Group, UC Berkeley. *4.4BSD Programmer's Reference Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-078-3
- Computer Systems Research Group, UC Berkeley. *4.4BSD Programmer's Supplementary Documents*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-079-1
- Harbison, Samuel P. and Steele, Guy L. Jr. *C: A Reference Manual*. 4th ed. Prentice Hall, 1995. ISBN 0-13-326224-3
- Kernighan, Brian and Dennis M. Ritchie. *The C Programming Language*. 2nd Ed. PTR Prentice Hall, 1988. ISBN 0-13-110362-8
- Lehey, Greg. *Porting UNIX Software*. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-126-7
- Plauger, P. J. *The Standard C Library*. Prentice Hall, 1992. ISBN 0-13-131509-9
- Spinellis, Diomidis. Code Reading: The Open Source Perspective. Addison-Wesley, 2003. ISBN 0-201-79940-5
- Spinellis, Diomidis. Code Quality: The Open Source Perspective. Addison-Wesley, 2006. ISBN 0-321-16607-8
- Stevens, W. Richard and Stephen A. Rago. *Advanced Programming in the UNIX Environment*. 2nd Ed. Reading, Mass. : Addison-Wesley, 2005. ISBN 0-201-43307-9
- Stevens, W. Richard. *UNIX Network Programming*. 2nd Ed, PTR Prentice Hall, 1998. ISBN 0-13-490012-X
- Wells, Bill. "Writing Serial Drivers for UNIX". *Dr. Dobb's Journal*. 19(15), December 1994. pp68-71, 97-99.

B.5. 操作系□原理

- Andleigh, Prabhat K. *UNIX System Architecture*. Prentice-Hall, Inc., 1990. ISBN 0-13-949843-5
- Jolitz, William. "Porting UNIX to the 386". *Dr. Dobb's Journal*. 1991年1月 - 1992年6月
- Leffler, Samuel J., Marshall Kirk McKusick, Michael J Karels and John Quarterman *The Design and Implementation of the 4.3BSD UNIX Operating System*. Reading, Mass. : Addison-Wesley, 1989. ISBN 0-201-06196-1
- Leffler, Samuel J., Marshall Kirk McKusick, *The Design and Implementation of the 4.3BSD UNIX Operating System: Answer Book*. Reading, Mass. : Addison-Wesley, 1991. ISBN 0-201-54629-9

- McKusick, Marshall Kirk, Keith Bostic, Michael J Karels, and John Quarterman. *The Design and Implementation of the 4.4BSD Operating System*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-54979-4

  (□本□的第二章的 在□版本 是 FreeBSD Documentation Project 的一部分。)

- Marshall Kirk McKusick, George V. Neville-Neil *The Design and Implementation of the FreeBSD Operating System*. Boston, Mass. : Addison-Wesley, 2004. ISBN 0-201-70245-2
- Stevens, W. Richard. *TCP/IP Illustrated, Volume 1: The Protocols*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-63346-9
- Schimmel, Curt. *Unix Systems for Modern Architectures*. Reading, Mass. : Addison-Wesley, 1994. ISBN 0-201-63338-8
- Stevens, W. Richard. *TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP and the UNIX Domain Protocols*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-63495-3
- Vahalia, Uresh. *UNIX Internals — The New Frontiers*. Prentice Hall, 1996. ISBN 0-13-101908-2
- Wright, Gary R. and W. Richard Stevens. *TCP/IP Illustrated, Volume 2: The Implementation*. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-63354-X

B.6. 安全方面的参考文献

- Cheswick, William R. and Steven M. Bellovin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-63357-4
- Garfinkel, Simson and Gene Spafford. *Practical UNIX & Internet Security*. 2nd Ed. O'Reilly & Associates, Inc., 1996. ISBN 1-56592-148-8
- Garfinkel, Simson. *PGP Pretty Good Privacy* O'Reilly & Associates, Inc., 1995. ISBN 1-56592-098-8

B.7. 硬件参考

- Anderson, Don and Tom Shanley. *Pentium Processor System Architecture*. 2nd Ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40992-5
- Ferraro, Richard F. *Programmer's Guide to the EGA, VGA, and Super VGA Cards*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-62490-7
- Intel 公司在他□的 □□人□网站上, 提供了□于他□的 CPU，芯片□，以及□准的文□。多数是PDF文件.
- Shanley, Tom. *80486 System Architecture*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40994-1
- Shanley, Tom. *ISA System Architecture*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40996-8
- Shanley, Tom. *PCI System Architecture*. 4th ed. Reading, Mass. : Addison-Wesley, 1999. ISBN 0-201-30974-2
- Van Gilluwe, Frank. *The Undocumented PC*, 2nd Ed. Reading, Mass: Addison-Wesley Pub. Co., 1996. ISBN 0-201-47950-8
- Messmer, Hans-Peter. *The Indispensable PC Hardware Book*, 4th Ed. Reading, Mass: Addison-

Wesley Pub. Co., 2002. ISBN 0-201-59616-4

B.8. UNIX® 史

- Lion, John *Lion's Commentary on UNIX, 6th Ed. With Source Code*. ITP Media Group, 1996. ISBN 1573980137
- Raymond, Eric S. *The New Hacker's Dictionary, 3rd edition*. MIT Press, 1996. ISBN 0-262-68092-0. 它也被称作 Jargon File
- Salus, Peter H. *A quarter century of UNIX*. Addison-Wesley Publishing Company, Inc., 1994. ISBN 0-201-54777-5
- Simon Garfinkel, Daniel Weise, Steven Strassmann. *The UNIX-HATERS Handbook*. IDG Books Worldwide, Inc., 1994. ISBN 1-56884-203-1. Out of print, but available online.
- Don Libes, Sandy Ressler *Life with UNIX* - special edition. Prentice-Hall, Inc., 1989. ISBN 0-13-536657-7
- *BSD* 族. http://www.FreeBSD.org/cgi/cvsweb.cgi/src/shared/misc/bsd-family-tree 或在 FreeBSD 机器上的 /usr/shared/misc/bsd-family-tree 。
- *Networked Computer Science Technical Reports Library*. http://www.ncstrl.org/
- *Old BSD releases from the Computer Systems Research group (CSRG)*. http://www.mckusick.com/csrg/: The 4CD set covers all BSD versions from 1BSD to 4.4BSD and 4.4BSD-Lite2 (but not 2.11BSD, unfortunately). The last disk also holds the final sources plus the SCCS files.

B.9. 各期刊

- *The C/C++ Users Journal*. R&D Publications Inc. ISSN 1075-2838
- *Sys Admin - The Journal for UNIX System Administrators* Miller Freeman, Inc., ISSN 1061-2688
- *freeX - Das Magazin für Linux - BSD - UNIX* (德文) Computer- und Literaturverlag GmbH, ISSN 1436-7033

附□ C: Internet上的□源

□展迅猛的FreeBSD使得□有的印刷、平面媒体跟不上它的□展□度！ 而□子版的也□是最好的，通常是唯一一个可以跟上最新□展方向的。FreeBSD来自于志□者的成果， 用□社区通常也扮演着是"技□支持部□"的角色。通□□子□， Web □□件和 USENET 新□□可以很快的□到他□。

以下列出了尽量多的□系FreeBSD用□社区的方式。如果□□□有其他的□源没有被包括在□儿， □告□FreeBSD文□□□□件列表，以便将它□加入到□里。

C.1. □件列表

□件列表通常是提□或是□起有□ FreeBSD 某一方面的□□技□□□最直接的途径。 有多□□□于不同 FreeBSD □□的□件列表。 把□的□□□送到最合□的□件列表通常能□得更加快速准□的回□。

本文的最后□出了各个不同的□件列表的使用□□。 在□□其中任何一个列表之前，□先□□使用条文。 □在□□□些□件列表的人□天都会收到上百封□于FreeBSD的信件。 □立列表的使用条文有助于□□□□□量。否□□些□□□□的列表将失去其意□。

如果□想要□□□送一封□件到 *FreeBSD* □件列表，□可以把□件□往 *FreeBSD* □□□件列表。□不要往其他的列表□送□□□件。

如果不知道□个□件列表□合于□送□的□□， □参□ 如何从 FreeBSD-questions □件列表中更快地得到答案。

在列表中□送任何□□之前， □首先学□使用□件列表的最佳方式， 例如如何通□□□ □件列表常□□□回答集 (FAQ)文□， 来避免□常重□的□□。

全部的□件列表□□都可以在FreeBSD World Wide Web服□器上□到。此服□器提供了很棒的□□□搜□功能，可□□□到FAQ的解答。 而在□件列表上提□之前，□先搜□是否已有答案。 □注意□意味着所有□往FreeBSD □件列表的消息都会被永久□□保存。 当□及到□私保□的□， 可以考□使用一个可使用后□□的□子□件地址并只□送公□的信息。

C.1.1. 列表摘要

一般性的列表*:* 以下的列表都是一般性的， 而且可以自由地加入，鼓励大家加入他□:

目□	用途
freebsd-advocacy	FreeBSD鼓吹者
FreeBSD 公告□件列表	重要的事件和里程碑
freebsd-arch	架□和□□的□□
freebsd-bugbusters	与FreeBSD□□□告数据□和有□工具□□相□的□□
freebsd-bugs	□告FreeBSD的Bug
freebsd-chat	和技□无□的FreeBSD□□区
freebsd-chromium	FreeBSD Chromium 相□的□□
FreeBSD-CURRENT □件列表	□□使用 FreeBSD-CURRENT 有□的一些□□

目□	用途
freebsd-isp	ISP使用FreeBSD的□□
freebsd-jobs	与FreeBSD有□的工作机会
freebsd-questions	用□□□和技□支持
FreeBSD 安全□□通知□件列表	安全通知
FreeBSD-STABLE; □件列表	□□使用 FreeBSD-STABLE 有□的一些□□
FreeBSD □□□件列表	在真正□送一个□件到□件列表之前可以先□送到□里□□

技□性的□件列表:　以下的□件列表是用来□□技□性□□的。　在加入□□及□□之前□□必□真□□□个列表主□，因□他□□□的内容都是□格地被限制着的。

目□	用途
FreeBSD ACPI □件列表	ACPI 和□源管理的□□
freebsd-afs	将 AFS 移植到 FreeBSD
freebsd-aic7xxx	□ Adaptec® AIC 7xxx □□□□
freebsd-amd64	将 FreeBSD 移植到 AMD64 系□
freebsd-apache	□于与 Apache 有□的 ports 的□□
freebsd-arm	将 FreeBSD 移植到 ARM® □理器
freebsd-atm	在 FreeBSD 上使用 ATM 网□
freebsd-bluetooth	在 FreeBSD 上使用 Bluetooth® 技□
freebsd-cluster	在集群□境中使用 FreeBSD
freebsd-cvsweb	CVSweb □□
freebsd-database	□□ FreeBSD 下□□和使用数据□
freebsd-doc	□建 FreeBSD 相□文□
freebsd-drivers	□ FreeBSD 撰写□□
freebsd-eclipse	FreeBSD 上的 Eclipse IDE、工具、 富客□□用，以及 ports 的用□□□。
freebsd-embedded	在嵌入式□用中使用 FreeBSD
freebsd-eol	□于与 FreeBSD 有□， 但已不再□ FreeBSD Project 所□□的□件的互助支持。
freebsd-emulation	在 FreeBSD 上模□其它系□， 如 Linux/MS-DOS®/Windows®
freebsd-firewire	FreeBSD 的 FireWire® (iLink, IEEE 1394) 技□□□
freebsd-fs	文件系□
freebsd-gecko	Gecko □染引□ issues
freebsd-geom	□□ GEOM 的□□和□□
freebsd-gnome	移植 GNOME 和 GNOME□用程序

目□	用途
freebsd-hackers	一般性的技□□□
freebsd-hardware	一般性的支持 FreeBSD 的硬件的□□
freebsd-i18n	FreeBSD 的国□化
freebsd-ia32	在 IA-32 (Intel® x86) 平台上□行 FreeBSD
freebsd-ia64	将 FreeBSD 移植到 Intel® 即将推出的 IA64 系□
freebsd-ipfw	□于 IP 防火□代□再□□的技□性□□
freebsd-isdn	ISDN□□人□
freebsd-jail	□于 jail(8) 机制的□□
freebsd-java	Java™ □□人□以及移植 JDK™s 到 FreeBSD 的人□
freebsd-kde	移植 KDE 和 KDE □用程序
freebsd-lfs	移植 LFS 到FreeBSD上
freebsd-mips	移植 FreeBSD 到 MIPS®
freebsd-mobile	□于便携式□算机的□□
freebsd-mono	FreeBSD 上的 Mono 和 C# □用
FreeBSD 多媒体□用□件列表	多媒体□用程序
freebsd-new-bus	技□□□□于□□架□
freebsd-net	网□子系□和 TCP/IP 源代□的□□
freebsd-office	FreeBSD 上的□公套件
freebsd-performance	高性能、□□下安装后的性能□整□□
freebsd-perl	□多与 perl 相□的 ports 的□□
freebsd-pf	□于 packet filter 防火□系□的□□
freebsd-platforms	□于向非 Intel® 架□的平台上移植的□□
freebsd-ports	□于 Ports Collection 的□□
freebsd-ports-bugs	ports bugs/PRs□□
freebsd-ppc	移植 FreeBSD 到 PowerPC®
freebsd-proliant	□于 FreeBSD 在 HP ProLiant 服□器平台上的技□□□
freebsd-python	FreeBSD □属的 Python □□
freebsd-rc	□于 rc.d 系□及其□□的□□
freebsd-realtime	FreeBSD □□□展的□□
freebsd-ruby	□于 FreeBSD 上 Ruby 的□□
freebsd-scsi	SCSI 子系□
FreeBSD 安全□□□件列表	系□安全
freebsd-small	在嵌入式系□上使用 FreeBSD (已□□； □使用 freebsd-embedded 代替)

目□	用途
freebsd-sparc64	移植 FreeBSD 到 SPARC® 系□
freebsd-standards	□ FreeBSD □□ C99 以及 POSIX® □准
freebsd-sysinstall	sysinstall(8) 的□□
freebsd-threads	□程
freebsd-testing	FreeBSD 性能和□定性□□
freebsd-tilera	□□将 FreeBSD 移植到 Tilera 系列 CPU
freebsd-tokenring	在 FreeBSD 中支持 Token Ring
freebsd-toolchain	□□在 FreeBSD 中集成的□□工具集
freebsd-usb	□于 FreeBSD 的 USB 支持的□□
freebsd-virtualization	□□各□ FreeBSD 支持的虚□化技□
freebsd-vuxml	□于 VuXML 的□□□□
freebsd-x11	□□和支持在 FreeBSD 上□行的 X11
freebsd-xen	□□ FreeBSD Xen™ 上的移植 - □□和使用
freebsd-xfce	FreeBSD 上 XFCE 的移植和□□

限制□□的列表: 以下的列表是□□某些特定的□者而□的， 而且并不□合被当成是一般公□□□区。□最好在某一技□□□区参与□□后再□□□□□些 限制□□的□件列表，因□□□□可以了解到在□些□□区□言所需要的礼□。

目□	用途
freebsd-hubs	□行□象站点的成□(支持基本服□)
freebsd-user-groups	用□□□整
freebsd-vendors	商家在□布之前的□整
freebsd-wip-status	FreeBSD □目□度状□
freebsd-wireless	□□ 802.11 □， 工具和□□□□□□
freebsd-www	www.FreeBSD.org的□□

分□列表: 所有以上的列表在一个分□格式里面是可利用的。 一旦□□了一个列表，□可以在□的□号□□里面□置□的分□□□。

CVS 和 *SVN* 列表: 以下的□件是□□FreeBSD源代□的□更□□有□趣的人看的， 而且它□是只□的□件列表，□不能□Email□他□。

列表	源位置	描述
cvs-all	/usr/(CVSROOT|doc|ports)	所有□源代□的改□□□ (其他 CVS commit 列表的超集)
cvs-doc	/usr/(doc|www)	所有□ doc 和 www 源代□的改□□□
cvs-ports	/usr/ports	所有□ ports 源代□的改□□□
cvs-projects	/usr/projects	所有□ projects 源代□的改□□□

列表	源位置	描述
cvs-src	/usr/src	所有□ src 源代□的改□□□ （由 svn-to-cvs 提交□入程序生成）
SVN 整个 src □的修□□息 (除了 "user" 与 "projects")	/usr/src	所有□ Subversion □□的改□□□ (除了 user 和 projects)
SVN src □ head/-current 分支的修□□息	/usr/src	所有□ Subversion □□ "head" 分支的改□□□ (FreeBSD-CURRENT 分支)
svn-src-projects	/usr/projects	所有□ Subversion 源□□□中有□ projects 部分的改□□□
svn-src-release	/usr/src	所有□ Subversion 源□□□中有□ releases 部分的改□□□
svn-src-releng	/usr/src	所有□ Subversion 源□□□中有□ releng 部分的改□□□ (security / release engineering 分支)
svn-src-stable	/usr/src	所有□ Subversion 源□□□中有□ stable 分支的改□□□
svn-src-stable-6	/usr/src	所有□ Subversion 源□□□中有□ stable/6 分支的改□□□
svn-src-stable-7	/usr/src	所有□ Subversion 源□□□中有□ stable/7 分支的改□□□
svn-src-stable-8	/usr/src	所有□ Subversion 源□□□中有□ stable/8 分支的改□□□
SVN commit messages for only the 9-stable src tree	/usr/src	所有□ Subversion 源□□□中有□ stable/9 分支的改□□□
svn-src-stable-other	/usr/src	所有□ Subversion 源□□□中早期 stable 分支的改□□□
svn-src-svnadmin	/usr/src	所有□ Subversion 源 □□□中管理用脚本, hook 和其他配置数据的改□□□
svn-src-user	/usr/src	所有□ Subversion 源□□□中有□ user 部分的改□□□
svn-src-vendor	/usr/src	所有□ Subversion 源□□□中有□ vender 部分的改□□□

C.1.2. 如何□□

□□一个列表，点□上面的列表名字或到 https://lists.freebsd.org 并点□□入□感□趣的列表，□个列表的□面包含了所必需的□□操作指南。

其□□只需□送□件到 列表名@FreeBSD.org。 它将被再次□□到全世界的□个□件列表的成□。

点□上面的 URL，在列表的底部可以从□□的列表中退出。 也可以□送一个□子□件到 列表名-unsubscribe@FreeBSD.org 来退□。

此外，我□要求□必□保持在技□性的□件列表中只是□□技□。 如果□只是□一些重要的公告感□趣，建□□加入 FreeBSD 公告□件列表， 它的通信量比□低。

C.1.3. 列表□章

所有 FreeBSD 的□件列表都有同□的基本□□， 所有人必□按照□□来做。 □反□些□□□， FreeBSD Postmaster postmaster@FreeBSD.org 会在前□次□送警告， 如果第三次□反， FreeBSD Postmaster 将从所有 FreeBSD 的□件列表中□除□□的人， 并□□来自□信人之后的所有□件。 我□很□憾必□要遵守□□的□□，但今天的互□网是一个很混乱的□境， 它上面的很多□束机制， 都相当脆弱。

具体□□:

- 任何□表的主□都□当附合基本的列表概况。例如，如果列表是有□技□□□的， 那□□表的文章包含技□□□。不要把不相□的□□放在一起。 □于没有主□的自由形式的□□，可以使用 FreeBSD-chat

  freebsd-chat@FreeBSD.org。

- 不要将同一个□□□送到超□□个的□件列表上，当有一个清晰和明□的必□要 □表到□个列表的要求□，也只能是□个。□于大多数的列表，已□有相当多的□□了， 除了一些比□深奥的□□(如"-stable & -scsi")，没有必要同□将一个□□□到多个列表上。 如果一个信息以□□方式（多个□件列表在Cc行出□）被□送□□，那Cc行在把它再□送出去之前也将被整理。 无□□是最初□表者，都会□致□自己的交叉□送。

- 不容□□行人身攻□和□□（在前后的争□中），包括用□和□□人□。 □当遵守最起□的网□礼□，象需要征得同意才可以引用或□□私人□件等。 然而，也有非常少的情况下，□□的内容会符合列表□章，因此，它会在最初□予警告（或禁止）。

- □格的禁止非FreeBSD相□□品或服□的广告，一旦□□将□上取□。

□独的列表□章:

FreeBSD ACPI □件列表

*ACPI*和□源管理□□

freebsd-afs

*Andrew*文件系□

□个列表是用来□□porting和从CMU/Transarc使用AFS。

FreeBSD 公告□件列表

重要事件/里程碑

□是一个□布FreeBSD重大事件的□件列表。□包括有□snapshots和其他版本的公告， 新的FreeBSD的性能的公告，□可以用于指派志□者等等。□个列表比□小。

freebsd-arch

架□和□□□□

□个列表是□□FreeBSD的架□。本□上□保□内容的□技□性。例如主□是：

- 如何重新□建系□使其同□有几个自己□造的系□□行。
- 需要什□才能修□VFS来使Heidemann□工作。
- 我□□□改□□□□□程序接口以便能□在多□□□和体系□□上使用同□的□□程序。
- 如何写一个网□□□。

freebsd-bluetooth

FreeBSD 上的 *Bluetooth®*

□是一个 FreeBSD 的 Bluetooth® 用□聚集的□□区。 □里□迎□于□□□□、 □□□□、 □丁、 □□□告、 □□□度□告，功能需求以及其他与 Bluetooth® 相□的□□。

freebsd-bugbusters

同等□□□告□理□果

□个列表的目的是作□一个□整和□□□□来服□于Bug列表的成□，Bugbuster列表成□ 和其他任何的□PR数据□真正的有□趣的成□。□个列表不是□了□□□于Bug□□，□丁或PRs。

freebsd-bugs

Bug□告

□是一个□告FreeBSD的Bug的□件列表。可以随□通□ send-pr(1) 命令或WEB□面来提交Bug。

freebsd-chat

与*FreeBSD*社区相□的非技□性□目

□个列表超出了其他有□非技□、社会信息的内容。 包括□□Jordan看起来是否像一个机敏的□探，是否句首的字母要大写， □喝了很多□□， □儿的□酒□造的最好，□在他□的地下室里□造了□酒等等。□于偶然宣布重大的事件 （例如：将要□行的聚会，婚礼，生日，新工作等等）也能使用□□技□列表，除上述列□之外 任何事情都可以□布在-chat列表上。

freebsd-chromium

FreeBSD 上的 *Chromium*

□是一个□□ FreeBSD 上 Chromium 相□□□的□件列表。 □是一个□□□□和安装 Chromium 的技□□列表。

Core Team

*FreeBSD*核心□□

□是一个只供核心成□内部使用的□件列表，只有当一个与FreeBSD相□的□重的事情需要裁决或□格□核□，才能□送消息到□个□件列表。

FreeBSD-CURRENT □件列表

□于使用*FreeBSD-CURRENT*版的□□

□是一个□□FreeBSD-CURRENT用□的□件列表。 它包括一些可能影□用□的新特性的警告，使用FreeBSD-current的一些指□。 任何□行"CURRENT"的人必□同意□个列表，□是一个□技□的□件列表。

freebsd-cvsweb

FreeBSD CVSweb□□

□于FreeBSD-CVSweb的使用，□□和□□的技□性□□。

freebsd-doc

文□□□

□个□件列表是与FreeBSD□建的文□的出版和□□的□□。 □个□件列表的成□都会提交到"The FreeBSD Documentation Project"。 它是一个□放的列表，可以自由地加入和做□献！

freebsd-drivers

□ *FreeBSD* 撰写□□□□

□是□于 FreeBSD 上的□□□□的技□□。 它主要供□写□□□□的□□人□提出□于如何使用 FreeBSD 内核提供的 API 来□写□□□□程序的□□。

freebsd-eclipse

FreeBSD 上的 *Eclipse IDE*、工具、 富客□□用， 以及 *ports* 的用□□□。

□个□件列表的目的， 是□在 FreeBSD 平台上□□、 安装、 使用、 □□和□□ Eclipse IDE、 工具、 富客□□用的用□， 提供互助式支持， 以及□将 Eclipse IDE 和□件移植到 FreeBSD □境中提供□助。

□一个目的是建立一个在 Eclipse 社区和 FreeBSD 社区之□的交流管道， 以□到互惠互利。

尽管□个列表主要□注的是 Eclipse 用□的□求， 它也□使用 Eclipse 框架□□ FreeBSD □用的□用提供了□□。

freebsd-embedded

在嵌入式□用中使用 *FreeBSD*

□个列表□□□于在嵌入式系□中如何使用 FreeBSD 的□□。 □是一个技□性的□件列表， 其主要内容是技□□□。 □□□一□件列表， 我□将嵌入式系□定□□那些不作□□面系□、 只完成某些□一任□的□算□□。 □些□例包括路由器交□机和 PBX □□的网□□□、 □程□量□□、 PDA、 PoS 系□，等等。

freebsd-emulation

模□其他系□， 例如 *Linux/MS-DOS®/Windows®*

□是一个□□□于如何在 FreeBSD 上□行□其他操作系□所撰写的程序的□□。

freebsd-eol

□于与 *FreeBSD* 有□， 但已不再□ *FreeBSD Project* 所□□的□件的互助支持。

□个□件列表主要用于那些有□趣提供或使用□□已不再□ FreeBSD Project 官方所支持 (例如，以安全更新或□丁的形式) 的 FreeBSD 相□□件的用□或公司□□。

freebsd-firewire

FireWire® (iLink, IEEE 1394)

□个□件列表是□于FreeBSD子系□FireWire® (aka IEEE 1394 aka iLink)的□□和□行。相□特定的主□包括□准，□□□□和他□的□□， □配器板/□/芯片□置，及他□的正□的代□的□□和□施。

freebsd-fs

文件系□

□于FreeBSD文件系□的□□。□是一个□技□的□件列表。

freebsd-gecko

Gecko □染引□

□是一个□□ FreeBSD 上 Gecko 有□的□用程序的□件列表。

□□ FreeBSD 上 Gecko Ports □用程序的□□， 以及它□的安装， □□和支持。

freebsd-geom

GEOM

□□GEOM和相□□行的□□。□是一个□技□的□件列表。

freebsd-gnome

GNOME

□□□于在FreeBSD系□上的GNOME□面□境 □是一个□技□的□件列表。

freebsd-ipfw

*IP*防火□

□是□于在FreeBSD里重新□□IP防火□代□的技□□□□。

freebsd-ia64

移植*FreeBSD*到*IA64*

□是一个有□将FreeBSD移植到Intel® IA64架□上的技□□□列表， □□一些相□的□□与解决方案。也□迎□□些□□感□趣的个□□□者。

freebsd-isdn

*ISDN*通信

□是一个FreeBSD支持的ISDN系□□□的□件列表。

freebsd-java

Java™□□

□是一个□□Java™ □用□□和 JDK™s的porting与□□的□件列表。

freebsd-jobs

工作的提供和□□

□个□□是□□与 FreeBSD 相□的雇□信息和个人□□， 比如： 如果□想□一个与 FreeBSD 相□的工作或有一个工作需要 FreeBSD □是一个□□来广告的好地方。 □ 不 是□一般性雇□□□的□件列表， □□个□□已□有了足□多的□□。

注意□个列表，像其他的 FreeBSD.org □件列表一□是会分□□全世界的□□者的。 因此，□需要明白□于位置和地域□□，□定之□是容易□系和可合作的。

Email最好□□使用 -□文本格式，不□基本的PDF,HTML和 很少其他的能被更多□者接受的格式也是可以的。Microsoft® Word (.doc) 格式是被□件列表服□器拒□的。

freebsd-kde

KDE

□□□于在FreeBSD系□上使用KDE。 □是一个□技□的□件列表。

freebsd-hackers

技□□□

□是一个与FreeBSD相□的技□□□□□，是一个主要的技□性□件列表。 他是□□个□的工作在FreeBSD上的人来提出□□或□□相□的解决方案, 也□迎□□些□□感□趣的个□的□□者。□是一个□技□的□件列表。

freebsd-hardware

*FreeBSD*硬件的普通□□

有□FreeBSD□行的硬件□型的普通□□，包括是否□□的一些□□和建□。

freebsd-hubs

□象站点

人□□行FreeBSD的□象站点的公告和□□。

freebsd-isp

*ISP*供□商□□

□是一个□□使用FreeBSD的ISP供□商的□件列表。□是一个□技□的□件列表。

freebsd-mono

FreeBSD 上的 *Mono* 和 *C#* □用

□是一个□□ FreeBSD 上的 Mono □□框架的□件列表。 □是一个□技□的□件列表。 它是□将 Mono 或 C# □用移植到 FreeBSD， 以及提出□□及□□其他解决方案的人准□的。 此外， 也□迎有□趣参与□□的其他人。

freebsd-office

FreeBSD 上的□公套件□用

□于□公套件□用， 它□的安装、 □□和 FreeBSD 支持的□□中心。

freebsd-performance

□□□于□整及高速□行*FreeBSD*

□个□件列表提供了一个□□客，管理□和有□的□体去□□与FreeBSD性能相□的主□的空□。 可以在□里□行□□的包括在任意高□□下，体□版下或者是有限制的条件下安装FreeBSD。 非常鼓励自□地□了改□FreeBSD性能的相□□体去□□□个列表。 □是个高技□含量的列表理□上□□合有□富□□的FreeBSD用□，□客，或□FreeBSD的速度、性能 、升□感□趣的管理□。□不是一个□答式的列表，□于□些□□去□相□文

□，但他是 一个可以投稿的地方，或者了解□于待解决的与性能相□的主□。

freebsd-pf

□于 *packet filter* 防火□系□的□□和□□

□于 FreeBSD □境下 packet filter (pf) 防火□系□的□□。 □里□迎技□□□， 以及一般的□用□□。 此外，□里也是□□ ALTQ QoS 框架的合□□所。

freebsd-platforms

移植到非 *Intel®* 平台上

跨平台的 FreeBSD □□， □于非 Intel® FreeBSD 移植版本的□□和提□。 □是一个□技□性的□件列表， 其□□内容□格限制□技□。

freebsd-ports

*"ports"*的□□

□于FreeBSD的"ports collection" (/usr/ports)的□□， ports的基□□造和□整□的ports□□。□是一个□技□的□件列表。

freebsd-ports-bugs

*"ports" bugs*的□□

□□□于FreeBSD的"ports collection" (/usr/ports),□□□告 ports建□，或者ports的修正。□是一个□技□的□件列表。

freebsd-proliant

□于 *FreeBSD* 在 *HP ProLiant* 服□器平台上的技□□□

□个□件列表用来□□在 HP ProLiant 服□器上使用 FreeBSD， 包括□□ ProLiant □用的□□、 管理□件、配置工具， 以及 BIOS 更新等。 同□地， □里也是□□ hpasmd、 hpasmcli， 以及 hpacucli 模□的主要□所。

freebsd-python

FreeBSD 上的 *Python*

□是一个□□□于如何在 FreeBSD 上改善 Python 支持的□件列表。 □是一个□技□的□件列表。 它是□那些移植 Python、 其第三方模□， 以及 Zope 相□□件到 FreeBSD 上的人准□的。 □里也□迎参与技□□□的人。

freebsd-questions

用□□□

□是一个有□FreeBSD□□的□件列表。□不□当□送"how to" □□□技□列表，除非□□□□个□□是非常可□的技□□□。

freebsd-ruby

有□ *FreeBSD* 上 *Ruby* 的□□

□是一个□□□于 Ruby 在 FreeBSD 上支持的□件列表。 □是一个□技□的□件列表。它是□那些移植 Ruby、第三方□以及 各□ framework 准□的。

□里也□迎参与技□□□的人。

freebsd-scsi

*SCSI*子系□

□是一个□□FreeBSD的SCSI子系□的□件列表。□是一个□技□的列表。

FreeBSD 安全□□□件列表

安全□□

FreeBSD的□算机安全□□（DES,Kerberos,已知的安全漏洞和修□等）。 □是一个□技□的□件列表。注意：□不是一个□和答的列表，但是同□□出 □□和答案到FAQ是□迎的。

FreeBSD 安全□□通知□件列表

安全通知

FreeBSD安全□□和修□的通知。□不是一个□□列表，□□的列表□当是FreeBSD-security

freebsd-small

在嵌入式□用程序中使用*FreeBSD*

□个列表□□了与□小的和嵌入的FreeBSD安装的□□主□。 □是一个□技□的列表。

□一列表已被 freebsd-embedded 代替。

FreeBSD-STABLE; □件列表

□□□于*FreeBSD-STABLE*版的使用

□是一个FreeBSD-STABLE用□的□件列表。它包括-STABLE的新特性可能会影□用□的警告。 任何□行"STABLE"的人□当□常□注□个列表。□是一个□技□的列表。

freebsd-standards

*C99 & POSIX*一致

□是□于FreeBSD□□C99和POSIX□准的技□□□□。

freebsd-toolchain

□□ *FreeBSD* 中集成的□□工具集

□是□于□□ FreeBSD 中集成的□□工具集的□□。 □里有包括 Clang 和 GCC， 以及其他□似□□器、 □接器和□□器等□件的□□。

freebsd-usb

□□ *FreeBSD* 的 *USB* 支持

□个□件列表是□于 FreeBSD 上的 USB 支持的技□性□□。

freebsd-user-groups

用□□□整列表

□个□件列表□□□从各地的使用群体到彼此相互□□□□和 从核心□□中指定个人。□个□件列表□被限制到大□和□□用□□ □□的□□之内。

freebsd-vendors

商家

□□FreeBSD□□和FreeBSD□硬件商家的□□。

freebsd-virtualization

□□各□ *FreeBSD* 支持的虚□化技□

□□ FreeBSD 所支持的各□虚□化技□的□件列表。 在注重□□基本功能，加入新特性的同□， 也□用□提供了一个□求□助和□□他□的使用□□的□所。

freebsd-wip-status

FreeBSD □目□度状□

□个□件列表是用来□布 FreeBSD 相□□目的□建和工作□度的。 □至□个列表的消息将会先被□核。通常建□把消息用 "To:" □□一个更典型的 FreeBSD 列表，而只□□ "BCC:" □□个列表。 □□□的工作□度就能在典型的列表上□□， 因□□个列表是不允□□□□□的。

□看一下□□中合□的消息作□例子。

可能□隔几个月， 会从□个列表中的消息中提取出一个□□性的消息摘要□到 FreeBSD 网站做□状□□告的一部分 。 □也能从那里□到更多的例子和以往的□告。

freebsd-wireless

□□ *802.11* □， 工具□□□□

FreeBSD-wireless □件列表集中□□ 802.11 □ (sys/net80211)， □□程序和工具的□□。

freebsd-xen

□□ *FreeBSD* 有□ *Xen™* 上的移植 - □□和使用

□个□件列表集中□□ FreeBSD 的 Xen™ 移植。 □期的流量会很小，所以□个列表旨在同□□ □□与□□□□的技□□□和管理部属□□ 提供一个□□的□所。

freebsd-xfce

XFCE

□是□□□于向 FreeBSD 移植 XFCE 的□□。 □是一个技□性的□件列表。 其成□是目前正活□地□行 FreeBSD XFCE 移植的□□人□， 主要用于提出□□或□□其他解决方法。 此外， 也□迎希望□注相□技□□□的其他人士。

C.1.4. □□□件列表

FreeBSD□件列表是使用了多□□□方法去消除□□□件、病毒和其他没用的□子□件。
□部分所描述的并不包括所有常用的保□□件列表的消除方法。

□件列表只包含一些允□的附件□型。所有在列表中有MIME□型的附件的□子□件在 □件列表中被□□之前将被□□掉。

- application/octet-stream
- application/pdf
- application/pgp-signature
- application/x-pkcs7-signature
- message/rfc822
- multipart/alternative
- multipart/related
- multipart/signed
- text/html
- text/plain
- text/x-diff
- text/x-patch

一些□件列表可以允□附件□其他MIME□型，但是以上列出的 □□被多数的□件列表所采用。

如果一个□子□件包含HTML和□文本形式，HTML的形式将被□除。 如果一个□子□件内容只是HTML形式，他将被□□□□文本格式。

C.2. Usenet新□□

除了FreeBSD□个特殊的新□□，□有很多□□FreeBSD或与FreeBSD用□相□的其他□□□。 一些新□□的□□□搜索□案是可以使用的， 有什□□□可以与Warren Toomey wkt@cs.adfa.edu.au□系。

C.2.1. BSD特殊的新□□

- comp.unix.bsd.freebsd.announce
- comp.unix.bsd.freebsd.misc
- de.comp.os.unix.bsd (德□)
- fr.comp.os.bsd (法□)
- it.comp.os.freebsd (意大利□)
- tw.bbs.comp.386bsd (繁体中文)

C.2.2. Internet上其他的UNIX®新□□

- comp.unix
- comp.unix.questions
- comp.unix.admin
- comp.unix.programmer
- comp.unix.shell
- comp.unix.user-friendly

- comp.security.unix
- comp.sources.unix
- comp.unix.advocacy
- comp.unix.misc
- comp.bugs.4bsd
- comp.bugs.4bsd.ucb-fixes
- comp.unix.bsd

C.2.3. X Window系□

- comp.windows.x.i386unix
- comp.windows.x
- comp.windows.x.apps
- comp.windows.x.announce
- comp.windows.x.intrinsics
- comp.windows.x.motif
- comp.windows.x.pex
- comp.emulators.ms-windows.wine

C.3. World Wide Web服□器

C.3.1. □□， 部落格， 社会性网□

- The FreeBSD Forums 提供了一个基于 web 的□□用以□□ FreeBSD 相□□□与技□。
- Planet FreeBSD 提供了□多由 FreeBSD □□者部落格摘要的集合。 很多的□□者都在上面□表有□他□工作□要的□□， 新的□丁和工作□度。
- The BSDConferences YouTube Channel 提供了一□世界各地 BSD 峰会的高□量□□。 □个是一个不□的□看重要□□者展示最新 FreeBSD 有□成果的方法。

C.3.2. Official Mirrors

Central Servers, Armenia, Australia, Austria, Czech Republic, Denmark, Finland, France, Germany, Hong Kong, Ireland, Japan, Latvia, Lithuania, Netherlands, Norway, Russia, Slovenia, South Africa, Spain, Sweden, Switzerland, Taiwan, United Kingdom, United States of America.

(as of UTC)

Central Servers

- https://www.FreeBSD.org/

Armenia

- http://www.at.FreeBSD.org/ (IPv6)

Australia

- http://www.au.FreeBSD.org/
- http://www2.au.FreeBSD.org/

Austria

- http://www.at.FreeBSD.org/ (IPv6)

Czech Republic

- http://www.cz.FreeBSD.org/ (IPv6)

Denmark

- http://www.dk.FreeBSD.org/ (IPv6)

Finland

- http://www.fi.FreeBSD.org/

France

- http://www1.fr.FreeBSD.org/

Germany

- http://www.de.FreeBSD.org/

Hong Kong

- http://www.hk.FreeBSD.org/

Ireland

- http://www.ie.FreeBSD.org/

Japan

- http://www.jp.FreeBSD.org/www.FreeBSD.org/ (IPv6)

Latvia

- http://www.lv.FreeBSD.org/

Lithuania

- http://www.lt.FreeBSD.org/

Netherlands

- http://www.nl.FreeBSD.org/

Norway

- http://www.no.FreeBSD.org/

Russia

- http://www.ru.FreeBSD.org/ (IPv6)

Slovenia

- http://www.si.FreeBSD.org/

South Africa

- http://www.za.FreeBSD.org/

Spain

- http://www.es.FreeBSD.org/
- http://www2.es.FreeBSD.org/

Sweden

- http://www.se.FreeBSD.org/

Switzerland

- http://www.ch.FreeBSD.org/ (IPv6)
- http://www2.ch.FreeBSD.org/ (IPv6)

Taiwan

- http://www.tw.FreeBSD.org/
- http://www2.tw.FreeBSD.org/
- http://www4.tw.FreeBSD.org/
- http://www5.tw.FreeBSD.org/ (IPv6)

United Kingdom

- http://www1.uk.FreeBSD.org
- http://www3.uk.FreeBSD.org/

United States of America

- http://www5.us.FreeBSD.org/ (IPv6)

C.4. Email地址

下面的用□□提供了与FreeBSD相□的□件地址。如果他被□用的□， □个列表的管理□有收回的□利。

域	工具	用□□	管理□
ukug.uk.FreeBSD.org	Forwarding only	ukfreebsd@uk.FreeBSD.org	Lee Johnston lee@uk.FreeBSD.org

附□ D: PGP公□

有些□候，□可能需要校□□名或者□送加密的□件□官□或者□□者，　　□里□了方便□而提供了一些密□。完整的 FreeBSD.org 用□密□可以在 pgpkeyring.txt 下□。

D.1. Officers

D.1.1. Security Officer □□ <security-officer@FreeBSD.org>

```
pub   rsa4096/D39792F49EA7E5C2 2017-08-16 [SC] [expires: 2023-01-02]
      Key fingerprint = FC0E 878A E5AF E788 028D  6355 D397 92F4 9EA7 E5C2
uid                           FreeBSD Security Officer <security-officer@FreeBSD.org>
sub   rsa4096/6DD0A349F26ADEFD 2017-08-16 [E] [expires: 2023-01-02]
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBFmT2+ABEACrTVJ7Z/MuDeyKFqoTFnm5FrGG55k66RLeKivzQzq/tT/6RKO9
K8DaEvSIqD9b0/xgK02KgLSdp0Bucq8HLDFYUk3McFa6Z3YwjobNCWkxc72ipvVl
uAOGN4H6fuoYOpeg4cLK1H9pktUIrzONTCixaZzc/Bu6X+aX4ywGeCfsuu8g5v03
fLCPBLLgf3Bm5wsyZ6ZaGmsmILrWzd+d/rbr35Mcc5BekdgywUI4R191qo1bdrw9
mEJP1V7Ik3jpExOsNnuhMTvm5OQMeCTfUvVEOtBU15QtbT+1LXF5FIOgML0LwS5v
RHZN+5w/xvzSnEULpj24UuMKLDs/u9rj8U/zET8QaE+oG7m/mr4jJWZEmdX8HKdO
WrpnVj6UAppk72qdBIEfLsOW2xB/NOjJppbCQH3+sw7DRYA2UnKE9Mptj/KKiE4
cs4c8Cupo2WSu93lEZDC5rCrULpT2lFeEXnRYlC/5oIgY5w9sFide9VI4CzHkkWX
Z2NPW/i1w3mFhoXjvnNLGOYMfAMKPxsRC2/Bn3bY0IhKvuIZ4rAeu7FTmKDDqFKQ
YEcrUOW74ZVng17AB29xzjWr4zNJVvp/CybFiUb8JoKkwtVWRqAVZIEgenAjU40d
G5+W4e+ccL0mfTQfEBbXRjnL2BL2tnaoBR42cTfbZGRucPHz7MrlKBEeZQARAQAB
tDdGcmVlQlNEIFNlY3VyaXR5IE9mZmljZXIgPHNlY3VyaXR5LW9mZmljZXJARnJl
ZUJTRC5vcmc+iQJUBBMBCgA+FiEE/A6HiuWv54gCjWNV05eS9J6n5cIFAlmT2+AC
GwMFCQoek4AFCwkIBwMFFQoJCAsFFgIDAQACHgECF4AACgkQ05eS9J6n5cKd9A/9
Fz3uGjNy28D0ALT1d/JJGzdQ2R3YwspHk9KHBr1LePkog9wf1WRalwCeNtPmA+g5
cn24psuzOeh1tRElImTZ2eE2ENPZ9XzK/J0ok0nK42MvmIwmMCyz+CaWv9GXW+FK
0oXnFmHi4YaQUVN3p+45TGkD9T+O5biVww7P47n/NnWsTfhLx0bzC7LyjPKXINai
/LgPgtlcOgY65/YhW/qhADCkoU7qMp9is41jMjTu1WB3OBPJkUkNpHfu6r15y8FN
Wqsk7K4W6Obr/WQ6VKGGXgh/a5mTcaEoFGMO16uHijAY4nXeb2HGZlBKxgmPH9Ur
aT4A9Pz/n+rIRMrK+rs+msFPemQHHNBYxy+x99uBpRBNyT2Su6GouZIxu5J16aIM
V0ZyOy/dy7m/uJ4sMhJPqKkd8a+MoQs/2L1M1y1EAzsO/QZqIrKrCluaftNN9k/B
qU0XClSDqB6sRMF7HFzYqb+f+M6cwSL/3Cp1Yx4rZ/onEE/MdWp64+3R87dETTXd
5tWXQw04qOhfPri5cBTI7r3t/qMO1iNXCGSG5RJbGkas6N6t6Mj83L4ItjI8doLf
aSIWZjj1XP3/me2hFJ6h2G5y5A+khO4ZwhC0ATFSq1fYbVGHw5AtfthIgNn8FoWu
+Sb8h7/RqTr7F6LgWagAoAh0GtVj02SVABZjcNZz/AKJAjcEEAEKACEWIQQc9/9v
rfXKn74bjLLtZ+zWXc9q5wUCWZPcTAMFAngACgkQ7Wfs1l3PauflkRAAgYcaBX0Y
ic4btxKoP/eOVpgUciOPPKEhDCiloQDyf4XQnZFDoMfjgcHpbLTBZ6kiAz2UzDGr
fJ4yUqrD+xfixUfCd5YpwzsaSpCGzDzSxOBcP/SpuAFhe40awSOIf5MruQar9Mlf
33JyslDLULXXeewAq2pcGk0/WrrOragI6Cs2vPGy9XP96VvLxyhjrWjlKmnO+//w
UF8oIO5hhKoqbtoxxlcqJgsWVyHch0mnPzvr6GWwoPhFXocnh1oPdbLjX1AwmGm9
ltEYMge4QxONIXlXJR0TvuDuJOaLNvTOC3OI8L97fdBcZS7eNJrG5FAYR5Ft3ISf
```

```
KJowIsSLGDt/cYApqpyP2pv7FpCvnwHgXHYar7/q4zhngCFRxQ2DPUx1cIJQ3Bgh
HZolKyK1X7XE5ZVDfZ3s3gcHSVKS89pipgHHZNr4sSmOanA8rXHcyHS4o2zSi1ie
r4iBwnOk6cCd6UNzEIiq0y/XhP/sc7xeL0mn3wDuV7jDBP9sp65sexL1qtIAfnzL
pLQevm0z41ifrUH5nNeL6RdbXpaoXc8M4PJJeQKJDu04KzLcQpZdUdCJsbS6QO9w
srWR8enQXPEhz2CO4L77bM9TgYO29222jTqEPcbXcmxF/klxO1rpssTTHUnHHi1Z
LUGYCbZPjt+laTJ2YPHTjUtN1Jw85vSKCEuJATMEEAEKAB0WIQS7KNQLNg7uk2rt
FW/l97zLo73d+AUCWjSYRwAKCRDl97zLo73d+JKyB/9N5Ytao12nD5QzMLvceGh5
otCLN99TUryYiDVDLoNkBivq3jHQA/hOX2rwEueFq0+LF8/2DnglJuUICNtCxIzL
WXXf/Hr5iWBUQ0JxYNPQzzjdMSXGE0WMwYVpAbCGxHpIsetKLdHUCwneYhaywe3I
KzmRJSDJGV1IJB0sAfoFtgybZXHgIR61jQjtnNmmyYXliYCd0wmIhXQDFN91tzzG
+EZdJ3Fao9JsMC+x55jO6EOLVySZgRF5E8vCeKUWemQciKFC7EhKcljILPYAA21u
NmHCAgRHKWU9JMdFK0w9lQuN2HQaNfkahjarTNM/Q6LwxY0dLG0vVYifE085WFAf
uQINBFmT2+ABEACxi39m5nQZexzY3c9sg/w5mUYCD89ZNSkj427gduQMYYGn7YW6
jSPfVJ/V3+PDK824c0a0XasyDapQFY1CPTZYrReRPoyjb8tJjsSVGXXCTFpJZlFU
br6kS9mgcx58Sypke2PMVk73+W1N1Yco+nahfTECRuM2/T2zHHr0AdKuBPF28U+H
TxyLatKoIgQwHDs4E/f4ZTbAoHvu3PixAl7XHVXCgz0cHaLhRljXizbZDXngOdGm
lqdFlAIpL6/l8E3m1Er0m3IfFo6qSzWRHg/KaBGIL4YKetJ6ACjlkCe5qbatDpmk
gWlg3Ux4RBVjyCK834Xh7eZpEcNf2iwpm28glWh7XMHGUplTHkU3PWQ4vGfNxXB8
HBOd9r02/cHL6MiHwhCAfIzZGVtqR0i9Ira57TMdXTpJWNXUcgsCMsi/Bg2a+hsn
aiYLrZc18uNL5nqOqsqKG3c1TcmeN7nbxVgnrNST4AjteulkhmB9p8tNOXA3u979
OO0T5LPwdqIpobdZ0lfw4URnAGw4Wd4Sm9PtRw0RvuAk2M2e5KXNyxPWAuMVkoRR
a7wG6h/R8pki54Gexyc+JkfB4ZcOrzHNLurw6DhxroyfRs8WEgX0wNIGmJvCXSBG
54jb5w9qudYwzIg4YPfvuX8sfeY8MTNhal3rF0tvVloGj3l709wlaWlBYwARAQAB
iQI8BBgBCgAmFiEE/A6HiuWv54gCjWNV05eS9J6n5cIFAlmT2+ACGwwFCQoek4AA
CgkQ05eS9J6n5cKhWw/+PT0R4r2gPAxI8ESEe380BYOmneNAH24MFOgWXqWCj4zX
Uz992BVnW2aL5nH4O5d822LGeCrYUC7SCpQvlifdHZHjobgtizLTwuu40bc3gSOz
cxWlx2jKfx3Ezn6QQz2mhhK6fZ1AO0ObiQxQq25ldURep95L78E/C8XkCe11YlUR
ng3wQKeHM7awZWRw/QBC92haHuVtU3cx7At+zQL7jTBKSZqd34zzs0uoXIhk2h94
O07MMDZ8z8MeU337vdL+RKYtD2bljLwpf7/kqg1D/q44RJ4ZpZcha9G0GvtLaQg2
+MAPlLg1vOWZ8wOTLaQHm+uzYRpkqxkIV8OuVd4UikCd8t3VNjNG5rG/YRNIAX0A
UEzs6oMF5YOFE8LmykesbUHAbC07Vcb0AsT5u3XKixDiIpPdnYSwGlkvoOVVLdeh
q/aXLK9V8BpViG5+a8xP2fdF1eMqdnrKAsiO4GEiq193PN/FA049VeIs3fd0izAa
x7+ag1MGtoF5Pij5iTVJm6phH5SUd1P3FY3OmclxWj/MbL4ba/G/6FWcy5NXxdw9
L1bRqaM2KEHJ67aF6NZz7UMldwExAWzFbUon1LUpKysAukxVf0EnntydBeVOQ+JO
HdqEpirrVLMpxPttUB2xxbo947nMj7/Bnme2gvb0vxaC9xSGVxrpW9cg5iCwSdc=
=8rds
-----END PGP PUBLIC KEY BLOCK-----
```

D.1.2. Security Team Secretary <secteam-secretary@FreeBSD.org>

```
pub   4096R/3CB2EAFCC3D6C666 2013-09-24 [expires: 2018-01-01]
      Key fingerprint = FA97 AA04 4DF9 0969 D5EF  4ADA 3CB2 EAFC C3D6 C666
uid                          FreeBSD Security Team Secretary <secteam-
secretary@FreeBSD.org>
sub   4096R/509B26612335EB65 2013-09-24 [expires: 2018-01-01]
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
```

```
mQINBFJBjIIBEADadvvpXSkdnBOGV2xcsFwBBcSwAdryWuLk6v2VxjwsPcY6Lwqz
NAZr2Ox1BaSgX7106Psa6v9si8nxoOtMc5BCM/ps/fmedFU48YtqOTGF+utxvACg
Ou6SKintEMUa1eoPcww1jzDZ3mxx49bQaNAJLjVxeiAZoYHe9loTe1fxsprCONnx
Era1hrI+YA2KjMWDORcwa0sSXRCI3V+b4PUnbMUOQa3fFVUriM4QjjUBU6hW0Ub0
GDPcZq45nd7PoPPtb3/EauaYfk/zdx8Xt0OmuKTi9/vMkvB09AEUyShbyzoebaKH
dKtXlzyAPCZoH9dihFM67rhUg4umckFLc8vc5P2tNblwYrnhgL8ymUaOIjZB/fOi
Z2OZLVCiDeHNjjK3VZ6jLAiPyiYTG1Hrk9E8NaZDeUgIb9X/K06JXVBQIKNSGfX5
LLp/j2wr+Kbg3QtEBkcStlUGBOzfcbhKpE2nySnuIyspfDb/6JbhD/qYqMJerX0T
d5ekkJ1tXtM6aX2iTXgZ8cqv+5gyouEF5akrkLi1ySgZetQfjm+zhy/1x/NjGd0u
35QbUye7sTbfSimwzCXKIIpy06zIO4iNA0P/vgG4v7ydjMvXsW8FRULSecDT19Gq
xOZGfSPVrSRSAhgNxHzwUivxJbr05NNdwhJSbx9m57naXouLfvPAMeJYwARAQAB
tD9GcmVlQlNEIFNlY3VyaXR5IFRlYW0gU2VjcmV0YXJ5IDxzZWN0ZWFtLXNlY3Jl
dGFyeUBGcmVlQlNELm9yZz6JAj0EEwEKACcFAlJBjIICGwMFCQgH7b8FCwkIBwMF
FQoJCAsFFgIDAQACHgECF4AACgkQPLLq/MPWxmYt8Q/+IfFhPIbqglh4rwFzgR58
8YonMZcq+5Op3qiUBh6tE6yRz6VEqBqTahyCQGIk4xGzrHSIOIj2e6gEk5a4zYtf
0jNJprk3pxu2Og05USJmd8lPSbyBF20FVm5W0dhWMKHagL5dGS8zInlwRYxr6mMi
UuJjj+2Hm3PoUNGAwL1SH2BVOeAeudtzu80vAlbRlujYVmjIDn/dWVjqnWgEBNHT
SD+WpA3yW4mBJyxWil0sAJQbTlt5EM/XPORVZ2tvETxJIrXea/Sda9mFwvJ02pJn
gHi6TGyOYydmbu0ob9Ma9AvUrRlxv8V9eN7eZUtvNa6n+IT8WEJj2+snJlO4SpHL
D3Z+l7zwfYeM8FOdzGZdVFgxeyBU7t3AnPjYfHmoneqgLcCO0nJDKq/98ohz5T9i
FbNR/vtLaEiYFBeX3C9Ee96pP6BU26BXhw+dRSnFeyIhD+4g+/AZ0XJ1CPF19D+5
z0ojanJkh7lZn4JL+V6+mF1eOExiGrydIiiSXDA/p5FhavMMu8Om4S0sn5iaQ2aX
wRUv2SUKhbHDqhIILLeQKlB3X26obx1Vg0nRhy47qNQn/xc9oSWLAQSVOgsShQeC
6DSzrKIBdKB3V8uWOmuM7lWAoCP53bDRW+XIOu9wfpSaXN2VTyqzU7zpTq5BHX1a
+XRw8KNHZGnCSAOCofZWnKyJAhwEEAEKAAYFAlJBjYgACgkQ7Wfs1l3PaudFcQ//
UiM7EXsIHLwHxez32TzA/0uNMPWFHQN4Ezzg4PKB6Cc4amva5qbgbhoeCPuP+XPI
2ELfRviAHbmyZ/zIgqplDC4nmyisMoKlpK0Yo1w4qbix9EVVZr2ztL8F43qN3Xe/
NUSMTBgt/Jio7l5lYyhuVS3JQCfDlYGbq6NPk0xfYoYOMOZASoPhEquCxM5D4D0Z
3J3CBeAjyVzdF37HUw9rVQe2IRlxGn1YAyMb5EpR2Ij612GFad8c/5ikzDh5q6JD
tB9ApdvLkr0czTBucDljChSpFJ7ENPjAgZuH9N5Dmx2rRUj2mdBmi7HKqxAN9Kdm
+pg/6vZ3vM18rBlXmw1poQdc3srAL+6MHmIfHHrq49oksLyHwyeL8T6BO4d4nTZU
xObP7PLAeWrdrd1Sb3EWlZJ9HB/m2UL9w9Om1c6cb6X2DoCzQAStVypAE6SQCMBK
pxkWRj90L41BS62snja+BlZTELuuLTHULRkWqS3fFkUxlDSMUn96QksWlwZLcxCv
hKxJXOX+pHAiUuMIImaPQ0TBDBWWf5d8zOQlNPsyhSGFR5Skwzlg+m9ErQ+jy7Uz
UmNCNztlYgRKeckXuvr73seoKoNXHrn7vWQ6qB1IRURj2bfphsqlmYuITmcBhfFS
Dw0fdYXSDXrmG9wad98g49g4HwCJhPAl0j55f93gHLGIRgQQEQoABgUCUkGO5gAK
CRAV1ogEymzfsol4AKCI7rOnptuoXgwYx2Z9HkUKuugSRwCgkyW9pxa5EovDijEF
j1jG/cdxTOaJAhwEEAEKAAYFAlJBkdUACgkQkshDRW2mpm6aLxAAzpWNHMZVFt7e
wQnCJnf/FMLTjduGTEhVFnVCkEtI+YKarveE6pclqKJfSRFDxruZ6PHGG2CDfMig
J6mdDdmXCkN//TbIlRGowVgsxpIRg4jQVh4S3D0Nz50h+Zb7CHbjp6WAPVoWZz7b
Myp+pN7qx/miJJwEiw22Eet4Hjj1QymKwjWyY146V928BV/wDBS/xiwfg3xIVPZr
RqtiOGN/AGpMGeGQKKplkeITY7AXiAd+mL4H/eNf8b+o0Ce2Z9oSxSsGPF3DzMTL
kIX7sWD3rjy3Xe2BM20stIDrJS2a1fbnIwFvqszS3Z3sF5bLc6W0iyPJdtbQ0pt6
nekRl9nboAdUs0R+n/6QNYBkj4AcSh3jpZKe82NwnD/6WyzHWtC0SDRTVkcQWXPW
EaWLmv8VqfzdBiw6aLcxlmXQSAr0cUA6zo6/bMQZosKwiCfGl3tR4Pbwgvbyjoii
pF+ZXfz7rWWUqZ2C79hy3YTytwIlVMOnp3MyOV+9ubOsFhLuRDxAksIMaRTsO7ii
5J4z1d+jzWMW4g1B50CoQ8W+FyAfVp/8qGwzvGN7wxN8P1iR+DZjtpCt7J+Xb9Pt
L+lRKSO/aOgOfDksyt2fEKY4yEWdzq9A3VkRo1HCdUQY6SJ/qt7IyQHumxvL90F6
vbB3edrR/fVGeJsz4vE10hzy7kI1QT65Ag0EUkGMggEQAMTsvyKEdUsgEehymKz9
MRn9wiwfHEX5CLmpJAvnX9MITgcsTX8MKiPyrTBnyY/QzA0rh+yyhzkY/y55yxMP
INdpL5xgJCS1SHyJK85HOdN77uKDCkwHfphlWYGlBPuaXyxkiWYXJTVUggSjuO4b
```

```
jeKwDqFl/4Xc0XeZNgWVjqHtKF91wwgdXXgAzUL1/nwN3IglxiIR31y10GQdOQEG
4T3ufx6gv73+qbFc0RzgZUQiJykQ3tZK1+Gw6aDirgjQYOc90o2Je0RJHjdObyZQ
aQc4PTZ2DC7CElFEt2EHJCXLyP/taeLq+IdpKe6sLPckwakqtbqwunWVoPTbgkxo
Q1eCMzgrkRu23B2TJaY9zbZAFP3cpL65vQAVJVQISqJvDL8K5hvAWJ3vi92qfBcz
jqydAcbhjkzJUI9t44v63cIXTI0+QyqTQhqkvEJhHZkbb8MYoimebDVxFVtQ3I1p
EynOYPfn4IMvaItLFbkgZpR/zjHYau5snErR9NC4AOIfNFpxM+fFFJQ7W88JP3cG
JLl9dcRGERq28PDU/CTDH9rlk1kZ0xzpRDkJijKDnFIxT2ajijVOZx7l2jPL1njx
s4xa1jK0/39kh6XnrCgK49WQsJM5IflVR2JAi8BLi2q/e0NQG2pgn0QL695Sqbbp
NbrrJGRcRJD9sUkQTpMsLlQTABEBAAGJAiUEGAEKAA8FAlJBjIICGwwFCQgH7b8A
CgkQPLLq/MPWxmZAew//et/LToMVR3q6/qP/pf9ob/QwQ3MgejkC0DY3Md7JBRl/
6GWfySYnO0Vm5IoJofcv1hbhc/y3OeZTvK4s+BOQsNokYe34mCxZG4dypNaepkQi
x0mLujeU/n4Y0p0LTLjhGLVdKina2dM9HmllgYr4KumT58g6eGjxs2oZD6z5ty0L
viU5tx3lz3o0c3I9soH2RN2zNHVjXNW0EvWJwFLxFeLJbk/Y3UY1/kXCtcyMzLua
S5L5012eUOEvaZr5iYDKjy+wOxY4SUCNYf0GPmSej8CBbwHOF2XCwXytSzm6hNb3
5TRgCGbOSFTIy9MxfV5lpddQcdzijmuFSl8LySkL2yuJxjlI7uKNDN+NlfODIPMg
rdH0hBSyKci6Uz7Nz/Up3qdE+aISq68k+Hk1fiKJG1UcBRJidheds29FCzj3hoyZ
VDmf6OL60hL0YI1/4GjIkJyetlPzjMp8J7K3GweOUkfHcFihYZlbiMe7z+oIWEc7
0fNScrAGF/+JN3L6mjXKB6Pv+ER5ztzpfuhBJ/j7AV5BaNMmDXAVO4aTphWl7Dje
iecENuGTpkK8Ugv5cMJc4QJaWDkj/9sACc0EFgigPo68KjegvKg5R8jUPwb8E7T6
lIjBtlclVhaUrE2uLx/yTz2Apbm+GAmD8M0dQ7IYsOFlZNBW9zjgLLCtWDW+p1A=
=5gJ7
-----END PGP PUBLIC KEY BLOCK-----
```

D.1.3. 核心□□秘□ <core-secretary@FreeBSD.org>

```
pub   rsa4096/D8C8C83B49F26F17 2020-06-26 [SC] [expires: 2022-06-30]
      Key fingerprint = 4B64 E9E0 BDE9 B3EC C06B  5C66 D8C8 C83B 49F2 6F17
uid                           FreeBSD Core Team Secretary <core-
secretary@freebsd.org>
sub   rsa4096/377C937536E4821B 2020-06-26 [E] [expires: 2022-06-30]
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBF72HwABEAC5hl4kfh8DyRpp0WE5rwbnuS+wQ51EVTGs1vLho8OZ2XruzlQT
AezCnKLsqMgD/UEaBcn9kbKoeqp2sIwuEUX+P79KhRc4C8RJ8TMfDH0OtC091QVp
MYWbIsvZYCO04K+rN1Dbk2En3BOJVgTowqbZzR3hPvzeU2/P+Y3zMtpQGea2DB5d
24Q/tIuPMh89evEXOx0K5eM/4P2awSmA3J+h+r09UYjKejJ5OBUJQsMervWAHgCA
TxJQHoPXw+ZKpJB3dzyHKTMukVZhdCjK6Zt2tih/rO/CHDsitMgYRIl3w2X6pDfV
JOpvOBlzg7nooIw94v6Uxr2y/JWgOGh2qy07u4qE//y6uSl55s+Vq5TrFr79VSwB
GhY9As/0Dk1lyFisKp1/yiet2W7Pu4c99Z5dsrQPSTLFvkvonVRX8wgxRZwk6gWA
LEYklwoR0NXiqlrpBT10Tsnsa4aoUvZW6eyOWZrKsdsVn05sgRmvlfpiqBbwqldJ
0EeF/MztPuhmq4Hgn+DmmYnx/P85pZpThcfJx16VxS8nB7ExYljeC9LF8V8/1d7e
tfgAj8ezzNtr2TXSZ5gblQtYLjKdgBiBZqsxHPYHzfG8Zx3eYs2Myklf9p4lt7nv
atTroDt8pUGXfhGfoqSHSLXODfYAO9/7DOPqTy5Pan4i7aWBPP+gfK0kgQARAQAB
tDhGcmVlQlNEIENvcmUgVGVhbSBTZWNyZXRhcnkgPGNvcmUtc2VjcmV0YXJ5QGZy
ZWVic2Qub3JnPokCVAQTAQoAPhYhBEtk6eC96bPswGtcZtjIyDtJ8m8XBQJe9h8A
AhsDBQkDx60ABQsJCAcDBRUKCQgLBRYDAgEAAh4BAheAAAoJENjIyDtJ8m8XQFwP
/RqHPMSsLlTcq5NfK2MAVGmdtpL5wf84bchVWtcXUUEwXW1wI2cdDwu9SoqudDbP
```

```
2lrbMpxWeUWAgCpPCF/vCVo4Nzd0zb1cEGKRKFiZe/4EQ8dfvqr03YyupSQvx6+P
oY+8y3kl7iHJKBkwrASraB2p+N9XDAJDgqz+1M2Xbo7rcJx64wBOCyPAxd9JWsge
d8mXyAqZlrLihsTjLbhuYbJxpKM5YjGubVaQZaNIDxUduqc8Pt9VgHvWJBc9VPPA
3B6E9/PUFZYZeZQSROkYniN9NE7keitxj/rvZkpzcaXfAoDMC7CSoLBzlP+CJZ+i
Kk7IWz4JpxiYkE/IY4VvMMYms9tRP8fVv0+R7r7yKEA9SSlH+e9qC++OoWg4b+wV
OrWtVIWvaJCtj5ZAPCutGZxBdvXEbHd/Gv6uCzG86n4huz23U+Y4iLzoAlVelnQs
Hqu1wSAUBNpplyeZ1TvrGg2pufxLh8iXfh0npDP/6J+u0GUfeX4JoAzvxlatXMYI
fBmqmcZI6ShJN8qQtCUa5OMqbnieo7Fmpf8BsLegjAsQ+8w21ATD2boinStntLzF
/yoL/z9WYxmoOdHYcQ8bildjCvtbAKrZie8sI4SgWQz2UX6KX9sc/WOmWUEtjdqB
WfGratZNoxuQLUvEDftt7r9ts1jKVUl3dMPTCfU4wcj5iQIzBBABCgAdFiEEVbCT
pybDiFVxIrrVNqQMg7DW754FAl72J74ACgkQNqQMg7DW756LaA//Z3CCF5fQ08tx
RLeqHNsS5xCYS97TjZxY6xAMBjebkS+ABkgdbedSH+YNGfdaGSD/SMtvMAmnx55t
18DDdA4pqC5x2USaHjXFdbDdxKuKMAoSAtOpipVASVmW0FkZI5C5FDe3MF8+mfGb
EPhVPwKbo7R5tk4jUPyX8wUaOAyUX9fyQnwDxN+zTHvKwnX/+qwpoKaY2N4ZOI0w
rOF1kkczibbfwvjVYcpPovGALmTccnWo1XvpkhllG93Y21mH+T2Ub/BK3GhvgJQi
WwiDtMwelUnPLp4W1451OU1OyGzeT/XwuMPH9dsKz5Iw4/g1zqQEtZj2Gc0DP5we
HM50doTn+dVIF+WCFLhPYm0RSf8Zj8ngbX/HV2UYLB5k+uNT9YTnBVEdKVydx7Cp
IplC7XApJEfTUk7wl7YCGn5P5YolC7DSJlwcAjxdbffXLowBhgyOq+EJJgnqerZl
r4db58h2epIHRKgnSl5z4KoAGW1O5dFShBz1UYPj4cZdeE+twpcgEg3/7LMzPzF/
xQAQZ89axxXBaCPl+YVsuMJSerbNdPp1SjCs9e8Vev91tLFmt/sY4IpvbPHZavGl
/4ealh8E1zPgf8lVW9TPrUY6mjN/uDI2y39tk2EoFzOcSQhlEM6gRW8uV4q92cWM
V55hu7Vs2RrKA7fve9y+YBi3DdTwwHSJATMEEAEKAB0WIQSfAoNvUNOtWrdaxYgM
tAPk6VuW7AUCXvY98wAKCRAMtAPk6VuW7CDLB/9PSUSMV/pnC+X4ougpjpqfSJf8
5bozjkKSkNqXZmt2vJVImc/oSK13awq46FC4rAhk59lT3kaH6EKvDHQ5G8Twi07u
VotcOdtfMjXgPV6RLmo6Hps0E1nzmbsum6xeemRDf3D3n1kAdUteXNBxHTIdAbeY
p4Wxu46CC/SqD6HbnUF2o+/6dXXyV1lTnViIj6m5eFD2OQ4Jdq7GPsSjSS2XL4f9
jHZUOUJyyA0aFWjJ+SCzMkXSUnyiOCl4uUHdCgivLIRyZ/giWoQpr8sAgHXCh82h
T3BmbHgmcMgMh+wNxH878IPwUU0CKRd2dL5kOSZVCFuMnFsc9eIie5kMEJwPuQIN
BF72HwABEADT9l4GIYiFaYg2QbQ3wsmmFnP/pAZiHDxXI6wL6xCKj6o2sc1/b5j3
ILEiAoqZ5ZenXX6T7Epjal0ASkfsGo/n3vF18grSudIkXJPQXcb61fXU7xfmGAEU
HWABQG+OD/HTvUPAITVckl4LxVFkz3oqRnq13rxDk1XZYvLVWeBn8vfWF4/glz9k
etfLw71Pk9f86BuNb0vCPnWpOpZaOxKlabdGpMKDD+1RYC/L+ZEwKiLBfgXTzK3g
IWAX3kTrQjKBZzsQ0s5TFWkm+z80GVUq8HKlXUOuF8s7cX+KXGU2kYcC8DQrxPdL
jYm6N8axOn4RR8eP5ZFA0W7qMieFSHAjqCs4srdN1bGC3nS0zGsQCvtTRBbu0nen
O6uwzWQgTzWVfV+dqaEH2crnhn5CUI0A8jdbFBGDiBbWJz/QfRray1CEc8q+hZFM
OLBsVXrDVe6hUXTveGc9xAnXC+0o3nnc7WhWr1caTbbhnzlEbME8u2oLif7rkhc7
FanuQEyKa76J1zou08ZeLK/pUFXTbRCoyUEVL+VIxLESCWi1ptkDpiZey3l6fe0Q
WWRMLFMpbu3WTNl21bEwfRL03+fP1q+yGAV5hyJv/EMldd76v577dAolIsTh+aDP
PMJ7mJ5NwOuiC20HIlCjuVT5A2pBIzFfraZY/v4dzoaOpXZjEz9wIwARAQABiQI8
BBgBCgAmFiEES2Tp4L3ps+zAa1xm2MjIO0nybxcFAl72HwACGwwFCQPHrQAACgkQ
2MjIO0nybxcflQ/9FYvM/lBSzy4VFOjNsUkRtjmPtyw2dJmQOCbWoSHmibRCG26a
Upt5lp1n4LG/qEtDlus5mDETL+/TnYhCG+hhnHADc87goLwBwl37yK1NAYvOy2rm
TddjDT5vZW0yzHjHqIJlNxQ4OjMi/XjyHIzb0PGNayFVi3XkLVxWZI+lWON1btWk
gpFfEgqRqQbJxM2cSEQimkfrrE+b2/M4cGX9rThpTtpfpbyHjTsS6juo4/eIdnBA
UXpKce4Q9LB5zxDaakKoDVxxkc9R0HAAoIH4u+Fu8az+CuH2sJcVJWK7Nxct++N8
Xhj+FUS+Ay8siu+ScQjsOHOHRwr6a+6NT58eylwR5hwotmnzJHLZReqknoAjLEGT
d33jzKM/y6OqPe/oPGj2b13RkA2vRnCPm33+T57sLMonNe6hhlXs9VTgXxSAzfMa
cmVOdP+nxUsoc3MtqjE2z2BcI9WMmmJFeEgE2BOj703CQuot+8jcZFXGUW+i6V1a
k7dZEMDsbALNzxaRNGeJC6HiM1+dXFGLNHEIgBLGwdvFAxTfNauvK0p7skDWEx44
giaUjZYpQ21+SHjVKTUnFQiiIDORvs3jdZDaxK/Y/vSoLRUiLBiHZWa6mxQY4uc6
5nAzLZB2BiBRfdL8fEO154nWjAZBLbKhK+ke2DBoPvSWubLPJqZyh+GmZAE=
```

```
=3AI7
-----END PGP PUBLIC KEY BLOCK-----
```

D.1.4. Ports 管理□□秘□ <portmgr-secretary@FreeBSD.org>

```
pub   rsa2048/D8294EC3BBC4D7D5 2012-07-24 [SC]
      Key fingerprint = FB37 45C8 6F15 E8ED AC81  32FC D829 4EC3 BBC4 D7D5
uid                           FreeBSD Ports Management Team Secretary <portmgr-
secretary@FreeBSD.org>
sub   rsa2048/5CC117965F65CFE7 2012-07-24 [E]
sub   rsa4096/CA20328577064EB7 2013-10-05 [S]
sub   rsa4096/8B114B3613867E00 2013-10-05 [E]
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBFAOzqYBCACYd+KGv0/DduIRpSEKWZG2yfDILStzWfdaQMD+8zdWihB0x7dd
JDBUpV0o0Ixzt9mvu5CHybx+9lOHeFRhZshFXc+bIJOPyi+JrSs100o7Lo6jg6+c
Si2vME0ixG4x9YjCi8DisXIGJ1kZiDXhmVWwCvL+vLInpeXrtJnK8yFkmszCOr4Y
Q3GXuvdU0BF2tL/Wo/eCbSf+3U9syopVS2L2wKcP76bbYU0ioO35Y503rJEK6R5G
TchwYvYjSXuhv4ec7N1/j3thrMC9GNpoqjVninTynOk2kn+YZuMpO3c6b/pfoNcq
MxoizGlTu8VT4OO/SF1y52OkKjpAsENbFaNTABEBAAG0R0ZyZWVCU0QgUG9ydHMg
TWFuYWdlbWVudCBUZWFtIFNlY3JldGFyeSA8cG9ydG1nci1zZWNyZXRhcnlARnJl
ZUJTRC5vcmc+iQE4BBMBAgAiBQJQDs6mAhsDBgsJCAcDAgYVCAIJCgsEFgIDAQIe
AQIXgAAKCRDYKU7Du8TX1QW2B/0coHe8utbTfGKpeM4BY9IyC+PFgkE58Hq50o8d
shoB9gfommcUaK9PNwJPxTEJNlwiKPZy+VoKs/+dO8gahovchbRdSyP1ejn3CFy+
H8pol0hDDU4n7Ldc50q54GLuZijdcJZqlgOloZqWOYtXFklKPZjdUvYN8KHAntgf
u361rwM4DZ40HngYY9fdGc4SbXurGA5m+vLAURLzPv+QRQqHfaI1DZF6gzMgY49x
qS1JBF4kPoicpgvs3o6CuX8MD9ewGFSAMM3EdzV6ZdC8pnpXC8+8Q+p6FjNqmtjk
GpW39Zq/p8SJVg1RortCH6qWLe7dW7TaFYov7gF1V/DYwDN5iEYEEBECAAYFAlN2
WksACgkQtzkaJjSHbFtuMwCg0MXdQTcGMMOma7LC3L5b4MEoZ+wAn0WyUHpHwHnn
pn2oYDlfAbwTloWIiQEiBBABAgAMBQJQGiE/BQMAEnUAAAoJEJcQuJvKV618AhwH
/ie5fi+wL0aapUHHs454xUv8xtdPfKpA35U4R2ZaVZ6wTCWWl3by+i81YIiGFVmQ
uQkkms1vRFhY6fVYzOxQR3VhWTTFexgLlLdI88eoBlzlsIvv7/4bNT9dUcg2TeYS
Ah3TzZsmVbqXIg8XvrCBD/WhG1cXGonHKszP6RcyFSDDD+bQogONqujM94dIcuoZ
7VCqbUvFKJ+rI3uXA1XFZSgFI9cDtnKYQqpGJcDEH2VeSfX/4XFuTg64g93AcCvE
qHANgquSWBfJGAy9fqV51LrRp7wB1GLfO0Vr0HSpn148wXDBSdi5T1AdfiBb8ENd
1fUfI/QfxHS5klkGE/akQNqJAhwEEwEKAAYFAldVzKcACgkQrbv4YQo3ibdgsg//
SS3vgpNtwiZHkUhfKAvuEhTlHUgJno0s55JT8JhApX3wAWHf5HJDtU+rpvthVHF7
iWUBNZyilDb2fKQVsLJtUQx7Mq6Cov1wddGtU3cf675VPPMrW3i31Ai86Aq3+qZD
EFqFOv96kunpVdFsDpQ7n5Pl4w9bI2tsLrLIRvMvtVggOSYz962XQ/TVXc6OQ4JK
LU0VH7mDdSySDeM5Q4B4+xVT+8Fu/kdHGRaDojbCyQgNBTfEoslPfFMrI2sC4iGP
XUyc+l1QNA4AdE2Vjt5log4yE3iq3RnKnzzYiYZr1CgZJ9O2KUzzX5KE0/6f6uQo
WJrtLmxnsktRCyo5pJSfg88p32S87512FzYKOXJHrqhKlDcTNCet7/JZn4Q0UThb
3qh1nadFzMV1ZkeWzLp0pXYORGbKC8kGzfknCD5w0wBDxqwrX9LfOEUMTUHhE+v1
gW2X83kngBZqQRQnYnLGejI8LcI6+PqNeZ9UDE2XKNTWk6n/L18M5fcDOGmLYNTp
BeM/B3R7sudAhwnreJk+XiYMGVHxb13WzKR16axCNGPTji+GjIpwtXoBd3V7f9J+
e4ughPhNpN1CTzbzK7adB2gxJymfHH5AVBAc5FGIucMKp7R1diUYrKqEZx3byU+y
JpRzHu1mibInwSvO+D506GioPD+itU6Yfhci8VC2mI2JAhwEEgEIAAYFAlQYNs8A
```

```
CgkQ6rA8WL/cR49ZJRAA12CXRwt87Hce8A7CbLbVncpHGaDnf1VClHTdAQFO9py1
dLS5+LhYxEwOjIR2Vcc8CA03iwfJKshKXH2Z6FW6iSFCc9bCVzHTrcuwsGc5WZQM
JlGiHxYmsyWNad/fCW8mH0q1SjTqcPcApg2+1j/XQa4rcL42i87vB0LIs7TtEv+Y
WvxA301N6qHAVE42wrRkfS6gJjxOF/PpYhdhH2TBKCXd6yVrGL9sv5OqUoAvzKUH
sJ8q+n/iUP+/NVKGQej6cQsX7uA+sZTsnREQVIPhXwwFmEfil/gxjmpUZVMmks/D
b8ZfP29CJvwEIcfWn6shGcSA6YOds7fIybd8Mdt2D/BEBlIMBfKSK498BUgUht2v
3LJIgTVCIQoSEfYUK1T1GoTf48ewxgBsZPymvarERmEx30q/Y401EPobX3hQUQ0d
J5GSjKOflpP3AHASlWz5QacL580934fsC3BdxKwW8eY6fgaBXnvA+gE3lL0MTHIF
Wwz1JIDLMtxLSjFTtdI4SKF6FVfkoPRmfMzBMtbTkSp9M7HdmYFwtrESTpubbQkn
WEOHXWzNQPY4OrYkceIREtehg8sL5C/D9jPfqsiH40MYxlN93/6/M6fQ8jU9VZa7
cIRcbTqDHZq7G2Dgk42uF4NeIStYUWnNJBge24+sPe6fLjK0E9z5UjLaMq0cqO25
AQ0EUA7OpgEIAMzcSWKX+T66lfKXUV5J9+hXuOauu9WLW2dlq0+w9AlWfGfgGbMZ
z/7YmNUqzNjHqQngNsQQIShrFgVqNWkZOZn2TpLCrIl7Itl3gVYXWZQpKzjCgU1Y
mp+yNZR+fWZzSDF9DZdLZY9y9SI3lljb33o+N1EVfNUvRJksKY2lm6Gra9gPtaCZ
LuZu5zw/SSDnJBNU8wbMoce8yFDfcOjCq+4BcPAws8qcFCsmICmE6FZvEJZE/y9w
dH7kB1oU+kThpOKPEgu9YmubXNU81MxhGdQEDvG2525iSwQyr3lhwqGngJEcyDc6
A/cAJ/c+KZ4pc3ufULSgtXDqlJLE7wUg4esAEQEAAYkBHwQYAQIACQUCUA7OpgIb
DAAKCRDYKU7Du8TX1Th6B/4uVcoIFXcKaj1/GAZjeZl0M4mGTUKIjjfCITrMaV+F
tp1wo+ZujAAlepyQ1VesMlB/P1bw0AvZx2qwntaGjbB1yc9B2pmLeYBydBtW2LgX
LiafxRIy2uPFxSO510IyR5/K5cGTIyvz1o9ZcNlmYnpL4rs0xAx/HN2ErRiUmvoG
HvBMqt1yVQHtlUDoSJqFoBg0d7Vn4K8mvyHRBW/45ox0huV1H2yNLzcKpX/N9gak
kCjxv2YseDJgSYLwM+Ee9Tx/FgZm5jRYw3pcVcn8/NPojIBG/Wi/dTH3gTQtigda
RZoPdZkE7Iqu2o8FS1SSvDaZRwJHS5pAQjvDIDI8xoZRuQINBFJPlYgBEAC5Ao/e
SZr5n4MrTsZ0dcPMZ0AblMbWMX/gVsKlNCHzjAcQJlw0X233vQXNww2coJo62dEQ
g3FSeDpgzUXP776E7LfO9LKOPXTNYdoumllIYQ+B8nDY5MdMACMmfY14nU9y8gIQ
k0n/xo7aXDkQr5gLF3CR84YWyeXjNkgZF5jXZ+DHgv4Qm31H8zeuH5rBkeLSiiJT
z2sskGkL/3Tj96HxIeElkXPCrYxAOriMh87VjFhCUCjf+50nlbsA6VSS/ZeMlBgd
E6pStKf53nKEJDqJZx1F+aCuxJop/U4z2AhqmABFrjNQW1l4mCHYr5RJt2VPr6qu
gm0sP0ig2dmbiuFMwxkJ1u1snLG16mdfGnsIdZnX/ufmAcXl2s0Sb3xWqQf1r7Va
ovOCzloqK9SATDINzwsgYgo4R8F76FeVejN5bYa51YMogk63i170yzYCCkEBVL8F
tibHLsGtx4QYJIZCQUgj5qVgCGfXgNUboLxUPPd14bfL8vs991qsYFM4Q/GaJygg
RGWtvWY+0niGF3GZlZsQm0beD2JIBaKulQvvd6rV7jmjrwvtG/apJjY+RKr6M0be
kjb9LA3vOOE9C2nEekEvT6v2DIwLPGqN4MNIsDgJVb6YmX6BG+qbq39E89If1EUR
1GZ4BAbBZoXImcbgoPYPxmg/KP/j19J8zlh+DwARAQABiQM+BBgBAgAJBQJST5WI
AhsCAikJENgpTsO7xNfVwV0gBBkBAgAGBQJST5WIAAoJEMogMoV3Bk63xfYP/Aui
Ybn8sZ2DtA5j1EQhVdHpt1o13MUJIUGtN7v7FPDhTcqdlDgfI8SN1wLBnuEZ3JTx
du9yuRuTn1ACdoY1hRqzKlxjS5bLibwyQlJnR5yOnLMNUbzp9psjM1Ek6sT0Mp5L
57boIaosl2xkGqMUCVVo67oaIHCJFWLzk+JzBlwspPUPrqKCzcPMac7GQcPN851o
Y483prvwWzn412396nSvTVjJAX3GkfbA4UQPmGOG3KO6EylxX4d4DDdaak7gDPHW
xnNP20uefMjULhWifnF95W0i18+aImyYupLnci/RPwpFbzBVky3EEFgj0uaytHXL
aHVtcfCZ8FZ2hvNLaxSMZkPsJOEteegnWME7/9S21hinFc/ToNXvcCC9i7FWiw/T
iEZ8eBvHS6mSc+KZhhBEuwl+q6ybkqV8TToOD7YYj7NM/6vXXqISiZFII+rZZ9Dg
oCPNtNxkki6txUwBXq9lUvptVzJANpqpNVr+rWXsmDJ2AGQgc3NehdszIOXVSpld
deGx5kU19G+UEDQeyVNwt+JHU2oIy6+Hc/+QYV4KH84z/R4FUCgqeyb03VE+zdMv
5PpkTJpyvL51nWzmR+PCDhCw4TJFOUCIih6CWl7MO8FaOvC7UkCLXyKAP5uhDkRm
cVFgov/BvpASujRBMUdXZQFvjPn0YczYKr6qC63H0eMIAJEBqtzNv4RnVP2dY+gg
yhLwNf4kkaZwZ31TPClEeutUKxHLZiKNWMElO8efoB+ZXw1I1NnCJQ/h4H8RESsJ
foW9imPTsi6tgFIYX/KwLxtnVPXYw6e01iLxd8HsS5YruAuhFnMJZxHnAqZD1GL+
/zym9IdMQMd9lJm+nc5H3duJolmUyPxoO0dUMpdQwsATPZID0YBIZ1d7pprA9yj1
3+VMMxz7JYalkztadojC+HsZ15ZJy+5iSo4+Hd4MouS8VVUs7dsYuTMgpBYYhxZE
nyLYoXEaR3c8bAVvu2FsNRC5XCjHjpGUHMgZt1aRP2Q+A0Cus8Bn5OMvFkuwOUiR
```

```
Iya5Ag0EUk+VvwEQAMG0+JaBKCu2WPH87fIcoXQYEm6VxVgAesnshg2KfbnBuMko
5X2SAHsPxDsyEf+vKSohsyBNPpMSo5b1Q3CaKln664KJqJ3RitqMit+ajrGDpWnu
rbGZu9yJybhkgvn2dVb9BwSJ/CWaSVxVe8+BaLWhuYmig3WQo8WsFRFg5e89Mw0i
yoDy7SuLkrBFbECIT6OWC0qk+5gp5XgqnTbt0uHLp7V2mZpkmZ3796p5IjzTTUmo
GazFCt+PX1Qnb5yeISg2zABIG8WVcBHgeBxYnI8ADVEEztaknPofG8JKzf0YGk9G
c952pPgyTjqykVeVVlnpseDstqE9PEqjEW4Yqf73iC761fqyh/pW/bsadL2Fp0b1
qHTpRJWtDXgc38+kr/GSNg/sSydutJiAeH+AVGAgYkg7vFCkw332G2vudCPUYFff
rpidlk3KfzwOf+fTPXukdhd3XDNEHsvIVQ3H/qz2NrYV8pkvOSB3D2rg4746R6cd
cb4aBLtol9MicAQ1F/8VVsB/5Xouw09HHdXQdaSoj8zb1kqD6es4anAerVLxOvQi
0c6LZm9jogJBTkXhFZHAEy/mxvTwTdM1rUlybgki+VgG/rMkcwRsCwO/okDmNMlw
mfhLZegRvK7MkJAUcKUxuY18wylE5doZihhZlO9sXHD3jcYW6tbRjBnw1VZpABEB
AAGJAR8EGAECAAkFAlJPlb8CGwwACgkQ2ClOw7vE19XBDgf9FsF98hfzAvLqSNGA
G44bfihuHC4FOqqNcnWYEMh6QH/HLbc6v7gg7uMCxINAg5q3k16IjUfVtFC4ZJ9S
xZGUCOmdrkh/8KkFCaxOD/TqOtU1wpSM74waBLn5Et24N8LgCokOHA9V8ck4VBt6
muJ2xlbyouHlmLrm6cyaH+0A++S7nR5i53mOJrQjQGYJoBFgNlLZ9BljdUDX6MtT
Rni7dJM2tFRY5upeinjHNvFC2Wc7YNCm7hZkCK3jfSR0Gya66l/Hr+e0qhLWp6D8
l+CI8VSRRJk5+MVUpUV2NrhWWbZQdUrshD2Jq3Z426QzfWVIFTpP3PtPubvhtyQU
WLuj9Q==
=e71t
-----END PGP PUBLIC KEY BLOCK-----
```

D.1.5. <doceng-secretary@FreeBSD.org>

```
pub   rsa2048/E1C03580AEB45E58 2019-10-31 [SC] [expires: 2022-10-30]
      Key fingerprint = F24D 7B32 B864 625E 5541  A0E4 E1C0 3580 AEB4 5E58
uid                           FreeBSD Doceng Team Secretary <doceng-
secretary@freebsd.org>
sub   rsa2048/9EA8D713509472FC 2019-10-31 [E] [expires: 2022-10-30]
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBF27FFcBCADeoSsIgyQUY8vREwkTikwFFlNg31MVy5s/Nq1cNK1PRfRMnprS
yfB62KqbYuz16bmQKaA9zHN4FGfiTvR6tl66LVHm1s/5HPiLv8sP14GsruLro9zN
v72dO7a9i68bMw+jarPOnu9dGiDFEI0dACOkdCGEYKEUapQeNpmWRrQ46BeXyFwF
JcNx76bJJUkwk6fWC0W63D762e6lCEX6ndoaPjjLBnFvtx13heNGUc8RukBwe2mA
U5pSGHj47J05bdWiRSwZaXa8PcW+20zTWaP755w7zWe4h60GANY7OsT9nuOqsioJ
QonxTrJuZweKRV8fNQ1EfDws3HZr7/7iXvO3ABEBAAG0PEZyZWVCU0QgRG9jZW5n
IFRlYW0gU2VjcmV0YXJ5IDxkb2Nlbmctc2VjcmV0YXJ5QGZyZWVic2Qub3JnPokB
VAQTAQoAPhYhBPJNezK4ZGJeVUGg5OHANYCutF5YBQJduxRXAhsDBQkFo5qABQsJ
CAcDBRUKCQgLBRYDAgEAAh4BAheAAAoJEOHANYCutF5YB2IIALw+EPYmOz9qlqIn
oTFmk/5MrcdzC5iLEfxubbF6TopDWsWPiOh5mAuvfEmROSGf6ctvdYe9UtQV3VNY
KeeyskeFrIBOFo2KG/dFqKPAWef6IfhbW3HWDWo5uOBg01jHzQ/pB1n6SMKiXfsM
idL9wN+UQKxF3Y7S/bVrZTV0isRUolO9+8kQeSYT/NMojVM0H2fWrTP/TaNEW4fY
JBDAl5hsktzdl8sdbNqdC0GiX3xb4GvgVzGGQELagsxjfuXk6PfOyn6Wx2d+yRcI
FrKojmhihBp5VGFQkntBIXQkaW0xhW+WBGxwXdaAl0drQlZ3W+edgdOl705x73kf
Uw3Fh2a5AQ0EXbsUVwEIANEPAsltM4vFj2pi5xEuHEcZIrIX/ZJhoaBtZkqvkB+H
4pu3/eQHK5hg0Dw12ugffPMz8mi57iGNI9TXd8ZYMJxAdvEZSDHCKZTX9G+FcxWa
/AzKNiG25uSISzz7rMB/lV1gofCdGtpHFRFTiNxFcoacugTdlYDiscgJZMJSg/hC
GXBdEKXR5WRAgAGandcL8llCToOt1lZEOkd5vJM861w6evgDhAZ2HGhRuG8/NDxG
r4UtlnYGUCFof/Q4oPNbDJzmZXF+8OQyTNcEpVD3leEOWG1Uv5XWS2XKVHcHZZ++
ISo/B5Q6Oi3SJFCVV9f+g09YF+PgfP/mVMBgif2fT20AEQEAAYkBPAQYAQoAJhYh
BPJNezK4ZGJeVUGg5OHANYCutF5YBQJduxRXAhsMBQkFo5qAAAoJEOHANYCutF5Y
kecIAMTh2VHQqjXHTszQMsy3NjiTVVITI3z+pzY0u2EYmLytXQ2pZMzLHMcklmub
5po0X4EvL6bZiJcLMI2mSrOs0Gp8P3hyMI40IkqoLMp7VA2LFlPgIJ7K5W4oVwf8
khY6lw7qg2l69APm/MM3xAyiL4p6MU8tpvWg5AncZ6lxyy27rxVflzEtCrKQuG/a
oVaOlMjH3uxvOK6IIxlhvWD0nKs/e2h2HIAZ+ILE6ytS5ZEg2GXuigoQZdEnv71L
xyvE9JANwGZLkDxnS5pgN2ikfkQYlFpJEkrNTQleCOHIIIp8vgJngEaP51xOIbQM
CiG/y3cmKQ/ZfH7BBvlZVtZKQsI=
=MQKT
-----END PGP PUBLIC KEY BLOCK-----
```