中标麒麟可信操作系统 V6.0

文件系统与存储管理指南

中标软件有限公司

上海市徐汇区番禺路 1028 号数娱大厦 10 层（200030）

北京市海淀区北四环西路 9 号银谷大厦 20 层（100190）

广州市天河北路 898 号信源大厦 16 层 1604 室（510898）

目 录

中标麒麟软件使用许可协议

尊敬的中标麒麟用户：

首先感谢您选用由中标软件有限公司开发并制作发行的中标麒麟产品。

请在打开本软件介质包之前，仔细阅读本协议条款以及所提供的所有补充许可条款（统称“协议”）。一旦您打开本软件介质包，即表明您已接受本协议的条款，本协议将立即生效，对您和本公司双方具有法律约束力。

1. 使用许可

按照已为之支付费用的用户数目及计算机硬件类型，中标软件有限公司（下称“中标软件”） 向您授予非排他、不可转让的许可，仅允许内部使用由中标软件提供的随附软件和文档以及任何错误纠正（统称“本软件”）。

— 软件使用许可

在遵守本协议的条款和条件的情况下，中标软件给予贵机构非独占、不可转让、有限的许可，允许贵机构至多使用软件的五（5）份完整及未经修改的二进制格式副本，而此种软件副本仅可安装于贵机构操作的电脑中。

— 教育机构使用许可

在遵守本协议的条款和条件的情况下，如果贵机构是教育机构，中标软件给予贵机构非独占、不可转让的许可，允许贵机构仅在内部使用随附的未经修改的二进制格式的软件。此处的“在内部使用”是指由在贵机构入学的学生、贵机构教员和员工使用软件。

— 字型软件使用

软件中包含生成字体样式的软件（“字型软件”）。贵机构不可从软件中分

离字型软件。贵机构不可改动字型软件，以新增此等字型软件被作为软件的一部分交付予贵机构时所不具备的任何功能。贵机构不可将字型软件嵌入作为商业产品提供以换取收费或其他报酬的文件。

2. 限制

本软件受到版权（著作权）法、商标法和其他法律及国际知识产权公约的保护。中标软件和/或其许可方保留对本软件的所有权及所有相关的知识产权。对于中标软件或其许可方的任何商标、服务标记、标识或商号的任何权利、所有权或利益，本协议均不作任何授权。

3.关于复制、修改及分发

如果在所有复制品中维持本协议书不变，您可以且必须根据《GNU GPL-GNU 通用公共许可证》复制、修改及分发中标麒麟产品中遵守《GNU GPL-GNU 通用公共许可证》协议的软件，其他不遵守《GNU GPL-GNU 通用公共许可证》协议的中标麒麟产品必须根据符合相关法律之其他许可协议进行复制、修改及分发，但任何以中标麒麟产品为基础的衍生发行版未经中标软件有限公司的书面授权不能使用任何中标软件有限公司的商标或其他任何标志。

特别注意：该复制、修改及分发不包括本产品中包含的任何不适用《GNU GPL-GNU 通用公共许可证》的软件，如中标麒麟产品中包含的输入法软件、字库软件、第三方应用软件等。除非适用法律禁止实施，否则您不得对上述软件进行复制、修改（包括反编译或反向工程）、分发。

4. 有限担保

中标软件向您担保，自购买之日起九十（90）天内（以收据副本为凭证），本软件的存储介质（如果有的话）在正常使用的情况下无材料和工艺方面的缺陷。除上述内容外，本软件按“原样”提供。在本有限担保项下，您的所有补偿及中标软件的全部责任为由中标软件选择更换本软件介质或退还本软件的购买费用。

5. 担保的免责声明

除非在本协议中有明确规定，否则对于任何明示或默示的条件、陈述及担保，包括对适销性、对特定用途的适用性或非侵权性的任何默示的担保，均不予负责，但上述免责声明被认定为法律上无效的情况除外。

6. 责任限制

在法律允许范围内，无论在何种情况下，无论采用何种有关责任的理论，无论因何种方式导致，对于因使用或无法使用本软件引起的或与之相关的任何收益损失、利润或数据损失，或者对于特殊的、间接的、后果性的、偶发的或惩罚性的损害赔偿，中标软件或其许可方均不承担任何责任（即使中标软件已被告知可能出现上述损害赔偿）。根据本协议，在任何情况下，无论是在合同、侵权行为（包括过失）方面，还是在其他方面，中标软件对您的责任将不超过您就本软件所支付的金额。即使上述担保未能达到其基本目的，上文所述的限制仍然适用。

7. 终止

本协议在终止之前有效。您可以随时终止本协议，但必须销毁本软件的全部正本和副本。如果您未遵守本协议的任何规定，则本协议将不经中标软件发出通知立即终止。终止时，您必须销毁本软件的全部正本和副本。

8. 管辖法律

与本协议相关的任何诉讼均受适用的中华人民共和国法律管辖。任何其它国家和地区的选择法律的规则不予适用。

9. 可分割性

如果本协议中有任何规定被认定为无法执行，则删除相应规定，本协议仍然有效，除非删除防碍各方愿望的实现（在这种情况下，本协议将立即终止）

10. 完整性

本协议是您与中标软件就其标的达成的完整协议。它取代此前或同期的所有口头或书面往来信息、建议、陈述和担保。在本协议期间，有关报价、订单、回

执或各方之间就本协议标的进行的其他往来通信中的任何冲突条款或附加条款，均以本协议为准。对本协议的任何修改均无约束力，除非通过书面进行修改并由每一方的授权代表签字。

11. 商标和标识

贵机构承认并与中标软件有着以下共识，即中标软件拥有中标软件、中标麒麟商标，以及所有与中标软件、中标麒麟相关的商标、服务标记、标识及其他品牌标识（“中标软件标记”）。贵机构对中标软件标记的任何使用都应有利于中标软件。

12. 源代码

本软件可能包含源代码，其提供之唯一目的是在符合本协议条款之规定时供参考之用。源代码不可再分发，除非在本协议中有明确规定。

13. 因侵权而终止

如果本软件成为或在任一方看来可能成为任何知识产权侵权索赔之标的，则任一方可立即终止本协议。

14. Java 技术限制

贵机构不可更改“Java 平台界面”（简称“JPI”，即指明为“java”包或“java”包的任何子包中的类），无论通过在 JPI 中创建额外的类，还是通过其他方式导致对 JPI 中的类进行增添或更动，均为不可。如果贵机构创建一个额外的类以及一个或多个相关的 API，而它们（i）扩展 Java 平台的功能；并且（ii）可供第三方软件开发者用于开发可调用上述额外 API 的额外软件，则贵机构必须迅即广泛公布对此种 API 的准确说明，以供所有开发者免费使用。贵机构不可创建、或授权贵机构的被许可人创建以任何方式标示为“java”、“javax”、“sun”的额外的类、界面、子包或 Sun 在任何命名约定中指明的类似约定。参见 Java 运行时环境二进制代码许可的适当版本（目前位于 http://www.java.sun.com/jdk/index.html），以了解可与 Java 小程序和应用程序共同分发的运行时代码的可供情况。

中标麒麟可信操作系统 V6.0 产品介绍

为满足政府、国防、金融、电力、机要、保密等领域对操作系统的高安全性需求，中标软件有限公司（以下简称“中标软件”）基于多年来在操作系统安全和可信计算方面的技术积累，研制推出了国内首款自主可控、高安全等级的可信操作系统软件产品-中标麒麟可信操作系统 V6.0。

结合可信计算技术和操作系统安全技术，中标麒麟可信操作系统 V6.0 通过信任链的建立及传递实现对平台软硬件的完整性度量；提供基于三权分立机制的多项安全功能（身份鉴别、访问控制、数据保护、安全标记、可信路径、安全审计等）和统一的安全控制中心；全面支持国内外可信计算规范（TCM/TPCM、TPM2.0）；兼容主流的软硬件和自主 CPU 平台；提供可持续性的安全保障，防止软硬件被篡改和信息被窃取，系统免受攻击；为业务应用平台提供全方位的安全保护，保障关键应用安全、可信和稳定的对外提供服务。

中标软件还提供基于 Linux 操作系统的安全评估、安全优化、安全加固等安全服务和系统安全定制开发业务。

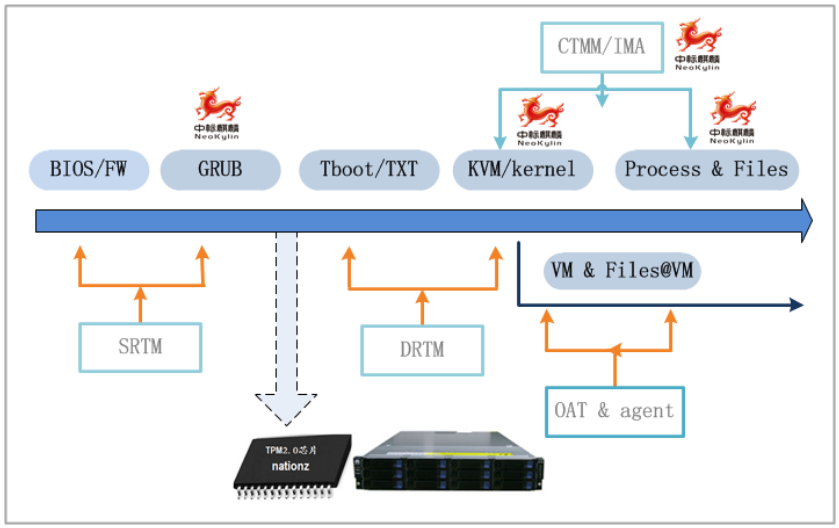
主要特性

■ 操作系统高安全等级

中标麒麟可信操作系统 V6.0 严格遵照可信计算技术规范（TCM/TPCM、TPM2.0）、GB/T 20272-2006 技术要求和国际 CC 标准等进行研制开发。通过操作系统安全的国家标准 GB/T 20272-2006 第四级（结构化保护级）测评认证并获得销售许可。

■ 可信计算实现内核级

国内首款全面支持 TCM/TPCM 和 TPM2.0 可信计算规范的可信操作系统，支持通用和专用可信密码芯片/模块；基于中标软件可信度量模块 CTMM（CS2C Trusted Measure Module）提供可信引导、可信启动和可信运行控制等功能；通过信任链的创建传递过程，实现对平台软硬件的完整性度量；提供基于可信芯片的上层可信功能和图形化的可信管理中心；并实现信任链从物理主机到虚拟化平台的拓展，提供对虚拟机的完整性度量。



■ 安全功能和机制全面

基于 LSM 的安全子系统框架，提供基于三权分立机制的多项安全功能，包括身份鉴别、自主访问控制、强制访问控制、数据机密性和完整性保护、安全标记、可信路径、安全审计等。针对不同的应用场景，系统支持细粒度的强制访问控制 SELinux 和轻量级强制访问控制 SMACK。

■ 系统管理配置灵活

内置主流数据库、中间件和应用服务器的安全策略，同时提供多种图形化安全策略配置和管理工具；基于图形化的安全控制中心实现系统安全可信功能模块化的集中配置和管理，界面友好，简洁易用；用户可以方便快捷完成系统的安全管理。

■ 良好的兼容性

中标麒麟可信操作系统 V6.0 适用于从服务器应用到桌面办公等各种环境，支持各类通用和专业应用；并内置默认的安全策略，实现系统安全和易用的结合，具有良好的软、硬件兼容性。系统支持 64 位应用程序，提供丰富的硬件驱动程序，中标软件有限公司还可协助第三方硬件厂商完成驱动程序的研发和移植，实现专用和特定硬件设备的支持。

系统要求

512MB 物理 RAM（推荐使用 1G 以上 RAM）

5G 以上可用磁盘空间

800x600 以上显示分辨率（推荐采用 1024x768 或更高分辨率）

硬件平台

Intel x86-64（AMD64）

自主 CPU 平台（龙芯、申威、兆芯、众志、Arm64 等）

获得更多的信息

如果出现了本手册不能解决的问题，可以通过如下的方式获得帮助：

阅读和打印 man 页以及 info 页。(man 页和 info 页是系统文档，可以帮助您了解系统提供了哪些可用命令以及如何使用它们)；

- 使用 GNOME 帮助浏览器；
- 登录 www.cs2c.com.cn 网站，查阅相关资料。

技术支持

请您按照中标麒麟可信操作系统 V6.0 产品包装或以下联系方式获取中标软件提供的技术支持服务，包括：

- 所有服务均以远程方式执行；
- 产品安装支持；
- 5＊8 小时电话，邮件，网站、传真等支持；
- 同版本补丁升级服务；
- 远程电话、邮件、网站、传真等支持服务只针对中标麒麟相关产品的安装、使用的问题提供支持，不包含对第三方软硬件的支持服务；
- 服务期按照合同规定起止日期内提供服务。

如果您有其它额外的技术支持需求，请致电中标软件有限公司，我们承诺为您提供优质的服务。

公司网址：www.cs2c.com.cn

客户热线：400-706-1825

电子邮件：support@cs2c.com.cn

公司电话：上海(021)51098866　北京(010)51659955　广州(020)38182526
公司传真：上海(021)51062866　北京(010)62800607　广州(020)38182529

1. 概述

存储管理指南中含有大量关于中标麒麟可信操作系统 V6.0 支持的文件系统和数据存储功能的信息。本指南的目的是作为一个系统管理员管理单节点（即非集群）存储解决方案时的快速参考。

1.1 中标麒麟可信操作系统 V6.0 的新功能

中标麒麟可信操作系统 V6.0 具有如下文件系统改进的特点：

1） 文件系统加密(技术预览)

现在，您可以在挂载时使用 eCryptfs1 来加密一个文件系统，它在一个实际的文件系统之上提供了一个加密层。这种“伪文件系统”为每个文件和文件名加密，它提供了比加密的块设备更精细的加密。关于文件系统加密的更多信息，请参考章节 14 加密文件系统。

2） 文件系统高速缓存(技术预览)

FS-Cache 允许您使用本地存储来缓存来自网络上的文件系统（例如通过 NFS）的数据。这有助于最大限度地减少网络流量，但它并不能保证服务器能更快地访问网络上的数据。FS-Cache 允许服务器上的文件系统直接与客户端的本地高速缓存进行交互，而无需创建一个 overmounted 文件系统。关于 FS-Cache 的更多信息，请参考章节 0FS-Cache。

3） 输入/输出限制处理

Linux 的 I/O 栈现在可以处理设备提供的 I/O 限制信息。这使得存储管理工具可以更好地为某些设备优化 I/O。更多信息，请参考章节 0 存储 I/O 对齐和尺寸。

4） ext4 支持

在此版本中完全支持 ext4 文件系统。它现在是中标麒麟可信操作系统 V6.0 的默认文件系统，支持无限多个子目录。它还具有更精细的时间戳，扩展属性支持和定额日志处理。关于 ext4 的更多信息，请参考章节 0 Ext4 文件系统。

5） 网络块存储

现在支持以太网光纤通道。这允许一个光纤通道接口使用 10 千兆以太网的同时保留光纤通道协议。有关如何设置的说明，请参考章节 23.7 配置以太网接口的光纤通道。

2. 安装过程中的存储注意事项

许多存储设备和文件系统设置只能在安装时进行配置。其它设置，如文件系统类型，只能修改到某一点，而不需要格式化。因此，在安装中标麒麟可信操作系统 V6.0 之前相应地计划存储配置是明智之举。

在为您的系统规划存储配置时，本章讨论了几方面的考量因素。对于实际安装说明（包括在安装过程中的存储配置），请参考中标麒麟可信操作系统 V6.0 提供的安装指南。

2.1 在安装过程中更新存储配置

以下设备或设置的安装配置已经根据中标麒麟可信操作系统 V6.0 要求进行了更新。

1） 以太网光纤通道(FCoE)

Anaconda 现在能够在安装过程中配置 FCoE 存储设备。

2） 存储设备过滤接口

Anaconda 现在已经提高了对安装过程中存储设备的控制。

现在，除了实际用于系统存储的设备之外，您可以控制哪些设备对安装程序可用/可见。通过设备过滤有两种途径可以实现。

1） 基本途径

对于只使用本地连接磁盘和固件 RAID 阵列作为存储设备的系统。

2） 高级途径

对于使用 SAN(如 multipath、iSCSI、FCoE)设备的系统。

3） 自动分区和/home

当可用的 LVM 物理卷容量为 50GB 或以上时，自动分区会为/home 文件系统创建一个单独的逻辑卷。当创建一个单独的/home 逻辑卷时，根文件系统（/）将被限制到 50GB，但/home 逻辑卷将扩展到占据卷组中所有剩余空间。

2.2 支持的文件系统

本节介绍了中标麒麟可信操作系统 V6.0 支持的每个文件系统的基本技术信息。

表 2-1 支持的文件系统技术规格

文件系统	最大支持尺寸	最大文件尺寸	最大子目录（每个子目录）	符号链接的最大深	ACL 支持	细节
Ext2	8TB	2TB	32,000	8	是	N/A
Ext3	16TB	2TB	32,000	8	是	第 8 章 Ext3 文件系统
Ext4	16TB	8TB	65,000[1]	8	是	第九章 Ext4 文件系统
XFS	100TB	8TB	65,000[1]	8	是	第十一章 XFS 文件系统

当连接数超过 65000 时，重置为 1，不再增加。

备注：并非所有中标麒麟可信操作系统 V6.0 支持的文件系统都记录在本指南中。此外，技术预览文件系统（如 BTRFS）也没有记录在本指南中。

2.3 特殊注意事项

本节列举了几个问题和因素用来针对特定存储配置进行考量。

1） /home、/opt、/usr/local 单独分区

如果你未来可能升级您的系统，请把/home，/opt 和/usr/local 放在一个单独的设备上。这将允许您重新格式化包含这个操作系统的设备/文件系统，同时保留您的用户和应用程序数据。

2） IBM 系统 Z 平台上的 DASD 和 zFCP 设备

在 IBM 系统 Z 平台上，DASD 和 zFCP 设备的配置是通过通道命令字（CCW）机制来完成的。CCW 路径必须被明确地添加到系统中，然后联机。对于 DASD 的设备，这仅仅是列出当 DASD=在启动命令行或在 CMS 配置文件的参数时的设备编号（或设备编号范围）。

对于 zFCP 设备，您必须列出设备编号，逻辑单元号（LUN），和全球端口名称（WWPN）。一旦 zFCP 设备初始化后，它被映射到一个 CCW 路径。FCP_x=启动命令行（或 CMS 配置文件）允许您为安装程序指定这一信息。

3） 使用 LUKS 加密块设备

使用 LUKS/的 dm-crypt 格式化一个用于加密块设备会破坏该设备上的所有现有的格式。因此，在新系统的存储配置被激活为安装过程的一部分之前，您应该决定为哪个设备进行加密（如果有的话）。

4） 过期 BIOS RAID 元数据

从一个为固件 RAID 配置的系统上移动一个硬盘而没有从该磁盘上删除 RAID 元数据会使 Anaconda 无法准确检测该磁盘。

警告：从磁盘中卸载/删除 RAID 元数据可能摧毁任何存储的数据。中标麒麟可信操作系统 V6.0 建议您备份您的数据，然后再继续。

为了从磁盘中删除 RAID 元数据，请使用下面的命令：

```
dmraid -r -E /device/
```

关于管理 RAID 设备的详细信息，请参考 man dmraid 和章节 15 独立磁盘冗余阵列(RAID)。

5） iSCSI 检测和配置

对于 iSCSI 驱动器的即插即用检测，请在 iBFT 启动功能的网络接口卡（NIC）的固件中对他们进行配置。安装过程支持 iSCSI 目标的 CHAP 身份验证。然而，安装过程并不支持 iSNS 搜索。

6） FCoE 检测和配置

对于以太网光纤通道(FCoE)驱动器的即插即用检测，请在 EDD 启动功能的网络接口卡（NIC）的固件中对他们进行配置。

7） DASD

在安装过程中，直接访问存储设备(DASD)不能被添加/配置。这种装置在 CMS 配置文件中被指定。

8） 启用 DIF/DIX 的块设备

DIF/DIX 是由某个 SCSI 主机总线适配器和块设备提供的硬件校验功能。DIF/DIX 启用时，如果该块设备被用作一个通用的块设备时，会发生错误。缓冲 I/O 或基于 mmap（2）的 I/O 将不能稳定地工作，因为缓冲的写入路径没有联锁来防止已缓冲的数据在 DIF/DIX 校验被计算出来后被重写。

正因为如此，I/O 稍后会随着校验错误而失败。这个问题是所有块设备（或基于文件系统的）缓冲 I/O 或 mmap（2）I/O 所常见的，所以它不可能解决重载所导致的这些错误。

因此，启用 DIF/DIX 的块设备应仅可用于使用 O_DIRECT 的应用程序。这样的应用程序应该使用原始块设备。另外，只要 O_DIRECT I/O 是通过文件系统发出的，您也可以放心使用启用 DIF/DIX 的块设备上的 XFS 文件系统。XFS 是

在执行某些分配操作时唯一不能返回缓冲 I/O 的文件系统。

为确保 I/O 数据在 DIF/DIX 校验已经被计算出来后不会发生改变，这一责任总是在于应用程序，因此，只有设计使用 O_DIRECT I/O 和 DIF/DIX 的应用程序应该使用 DIF/DIX 硬件。

3. Btrfs

Btrfs 是一个新的正在积极开发中的本地文件系统。它的目的是提供更好的性能和可扩展性，这将反过来惠及用户。

 备注：Btrfs 在这一点上不是一个达到生产质量的文件系统。中标麒麟可信操作系统 V6.0 尚在技术预览阶段，因此它只能用于开发英特尔 64 和 AMD64。

3.1 Btrfs 特点

Btrfs 被植入几个实用程序以便系统管理员易于管理。

1） 内置系统回滚功能

在应用系统更新前快照文件系统，如果出现错误，他们可以轻松地回滚到先前的系统状态。这是一种便利的操作。

2） 内置的压缩功能

这使得节省空间变得更容易。

3） 校验

这将提高错误检测效率。

具体功能包括集成的 LVM 操作，如：

a） 动态，在线添加或删除新的存储设备。

b） 对整个元器件 RAID 的内部支持。

c） 对元数据或用户数据使用不同 RAID 水平的能力。

d） 全面校验所有的元数据和用户数据。

4. LVM(逻辑卷管理)

LVM 是逻辑卷管理工具，其中包括分配磁盘，条带化，镜像和调整逻辑卷。

使用 LVM，硬盘驱动器或硬盘驱动器集被分配到一个或多个物理卷。LVM 物理卷可以放在其他可能跨越两个或多个磁盘的块设备上。

物理卷被合并成逻辑卷，/boot/分区除外。/boot/分区不能在逻辑卷组上，因

为引导装载程序无法读取它。如果 root(/)分区在逻辑卷上，则创建一个不属于卷组一部分的单独/boot/分区。

由于物理卷无法跨越多个驱动器，为了跨越多个驱动器，请为每个驱动器创建一个或多个物理卷。

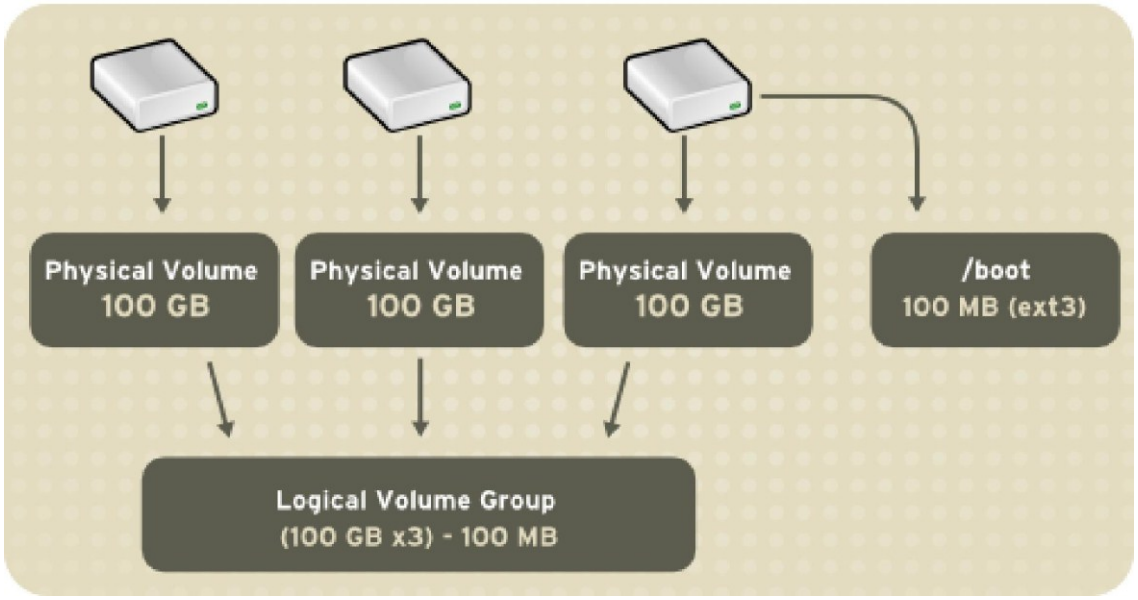


图 4-1 逻辑卷

卷组被分为逻辑卷，这些逻辑卷被分配了挂载点，如/home 和/；文件系统类型，如 ext2 或 ext3。当“分区”充分发挥其能力时，卷组中的自由空间可以被添加到逻辑卷来增加分区的大小。当一个新的硬盘驱动器被添加到系统中，它可以被添加到卷组，并且逻辑卷分区的大小可以增加。

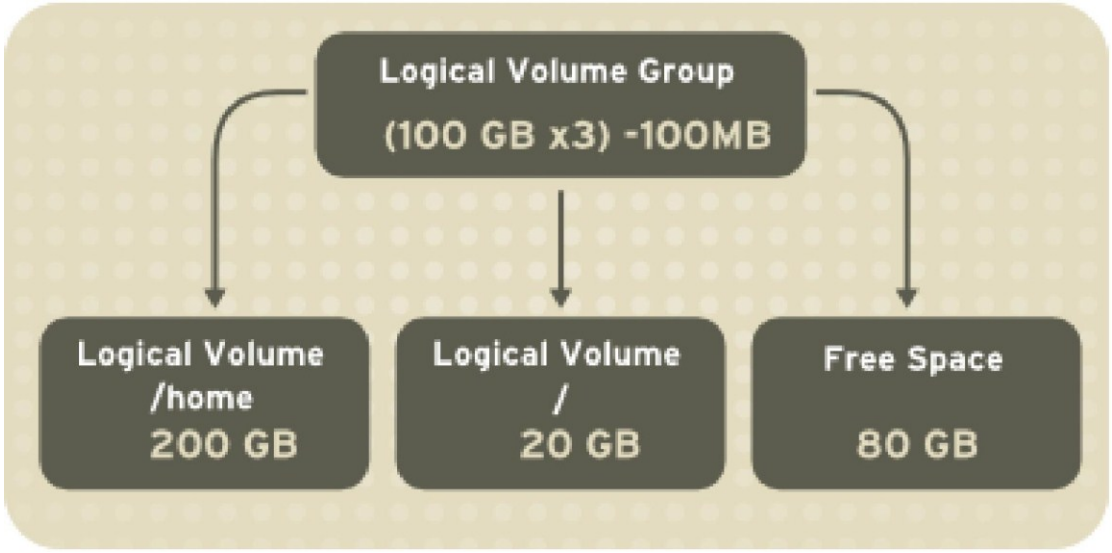


图 4-2 逻辑卷

另一方面，如果一个系统用 ext3 文件系统来分区，硬盘驱动器可分为确定大小的分区。如果一个分区已满，则不容易扩展分区的大小。即使分区移动到另一个硬盘驱动器，原来的硬盘空间必须被视为一个不同的分区或未使用的分区进行重新分配。

 重要：本章关于 LVM/LVM2 侧重于使用 LVM GUI 管理工具，即 system-config-lvm。关于集群和非集群存储中的 LVM 分区的创建和配置的全面信息，请参阅中标麒麟可信操作系统 V6.0 提供的逻辑卷管理器管理指南。

此外，中标麒麟可信操作系统 V6.0 安装指南还介绍了如何在安装过程中创建和配置 LVM 逻辑卷。

4.1 什么是 LVM2?

LVM 版本 2 或 LVM2 是中标麒麟可信操作系统 V6.0 所默认的，他们使用 2.6 内核中的设备映射驱动器。LVM2 可以从运行 2.4 内核的中标麒麟可信操作系统 V6.0 进行升级。

4.2 使用系统配置 LVM（system-config-lvm）

LVM 实用程序允许您在 X 窗口或图形界面管理逻辑卷。您可以通过从你的菜单面板选择**【系统】=>【管理】=>【逻辑卷管理】**来访问应用程序。另外，您可以通过从终端输入 system-config-lvm 来开始逻辑卷管理实用程序。

本节中使用的例子中，以下是在安装过程中创建的卷组的详细信息：

```
/boot - (Ext3) file system. Displayed under 'Uninitialized Entities'. (DO NOT initialize this partition).
LogVol00 - (LVM) contains the (/) directory (312 extents).
LogVol02 - (LVM) contains the (/home) directory (128 extents).
LogVol03 - (LVM) swap (28 extents).
```

以上逻辑卷在磁盘实体/dev/hda2 被创建，而/boot 创建于/dev/hda1 中。该系统还包括图 4-7 所示的“未初始化的实体”。说明了 LVM 工具的主窗口。以上配置的逻辑和物理视图如下所示。三个逻辑卷存在于相同的物理卷（hda2）上。

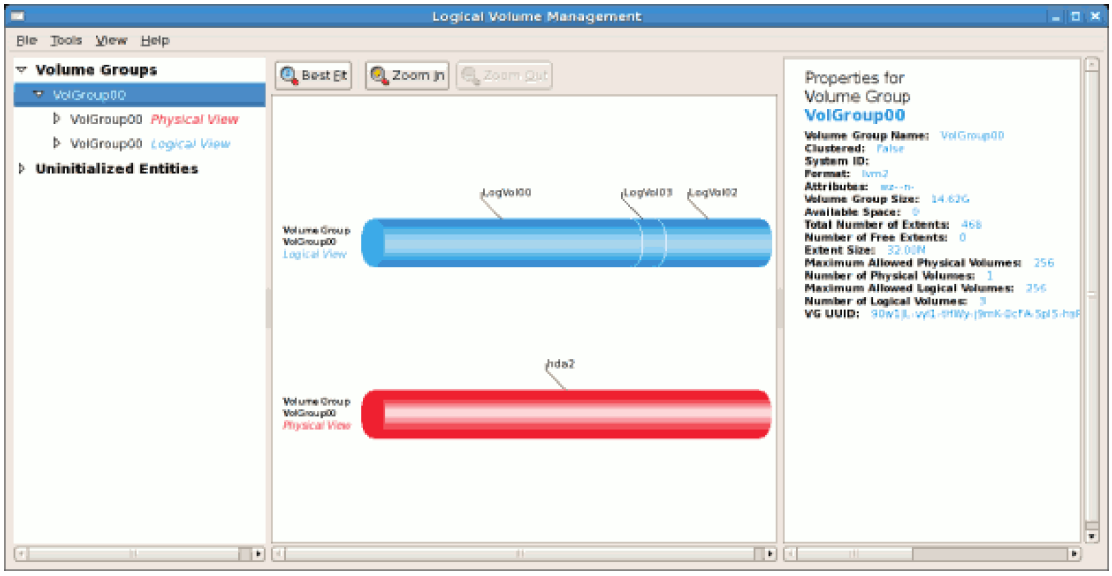


图 4-3 主要 LVM 窗口

下图说明了该卷的物理视图。在此窗口中，你可以从该卷组选择和删除一个卷或将区间从该卷迁移到另一个卷组。图 4-12 详细讨论了迁移区间的步骤。

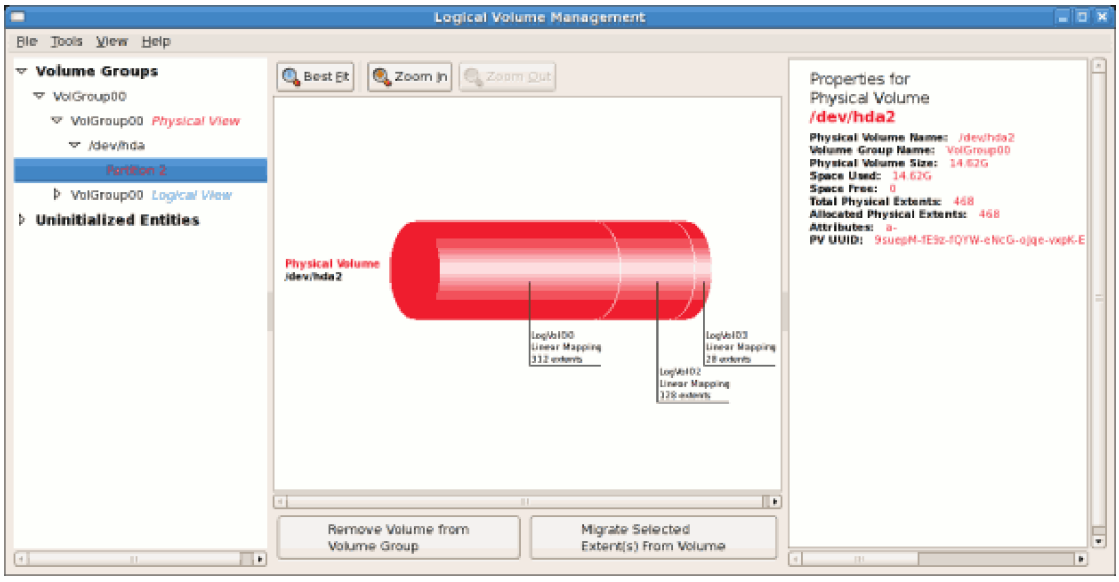


图 4-4 物理视图窗口

下图说明了所选卷组的物理视图。逻辑卷的大小也用所指示的单个逻辑卷的大小进行说明。

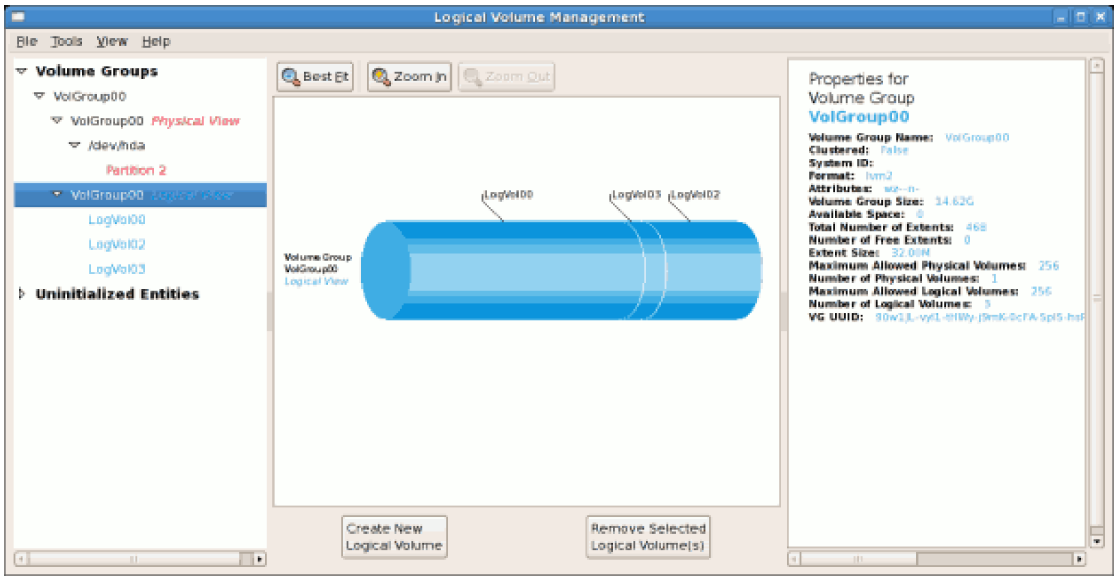


图 4-5 逻辑视图窗口

在左侧栏中，您可以在卷组中选择单个逻辑卷来查看每一卷的详细信息。本例子的目的是将逻辑卷名称“LogVol03”重命名为“Swap”。要执行此操作，请选择相应的逻辑卷，并单击**【编辑属性】**按钮。这将显示逻辑卷编辑窗口，从中可以修改逻辑卷的名称，大小（以 extent 为单位），还可以使用逻辑卷组的可用剩余空间。下图说明了这一点。

请注意，这个逻辑卷不能改变大小，因为目前卷组中还没有自由空间。如果有剩余空间，将启用此选项。点击**【确定】**按钮保存更改（这将重新挂载该卷）。要取消更改，请单击**【取消】**按钮。要恢复到上次快照设置，点击**【还原】**按钮。单击 LVM 实用程序窗口上的**【创建快照】**按钮，可以创建一个快照。如果所选的逻辑卷正为系统（例如）the/(root)目录所时用，因为该卷无法卸载，这个任务将不会成功。

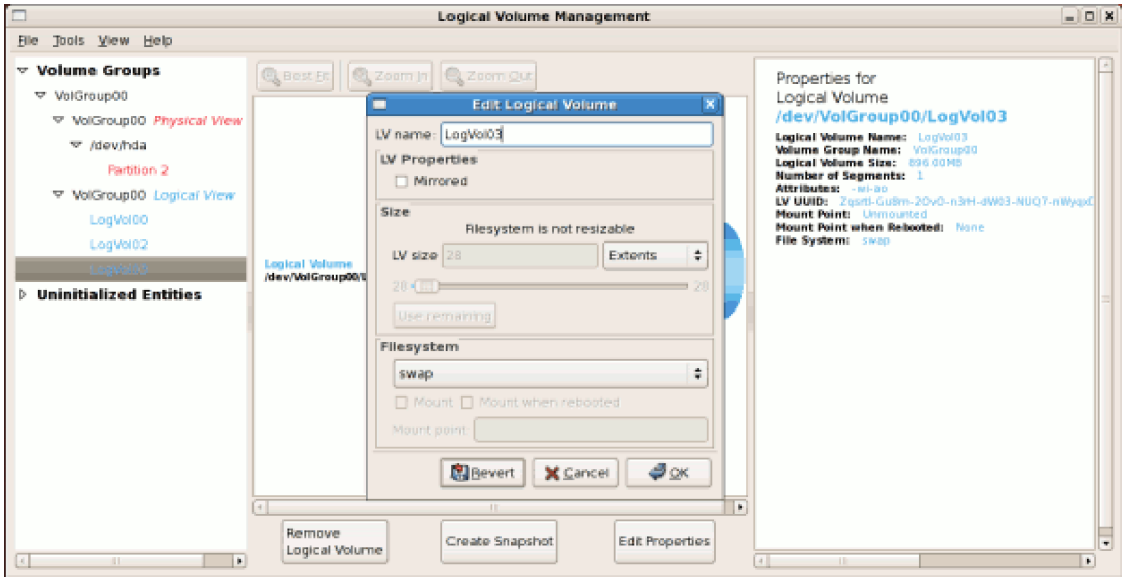


图 4-6 编辑逻辑卷

4.2.1 使用未初始化实体

“未初始化实体”包括未分区的空间和非 LVM 文件系统。在这个例子中，分区 3，4，5，6 和 7 在安装过程中被创建，并在硬盘上留下一些未分区的空间。请查看每个分区，并确保你阅读了**【磁盘实体属性】**窗口的右栏，以确保您不会删除关键数据。在这个例子中的第 1 分区不能被初始化，因为它是/boot 未初始化实体，说明如下。

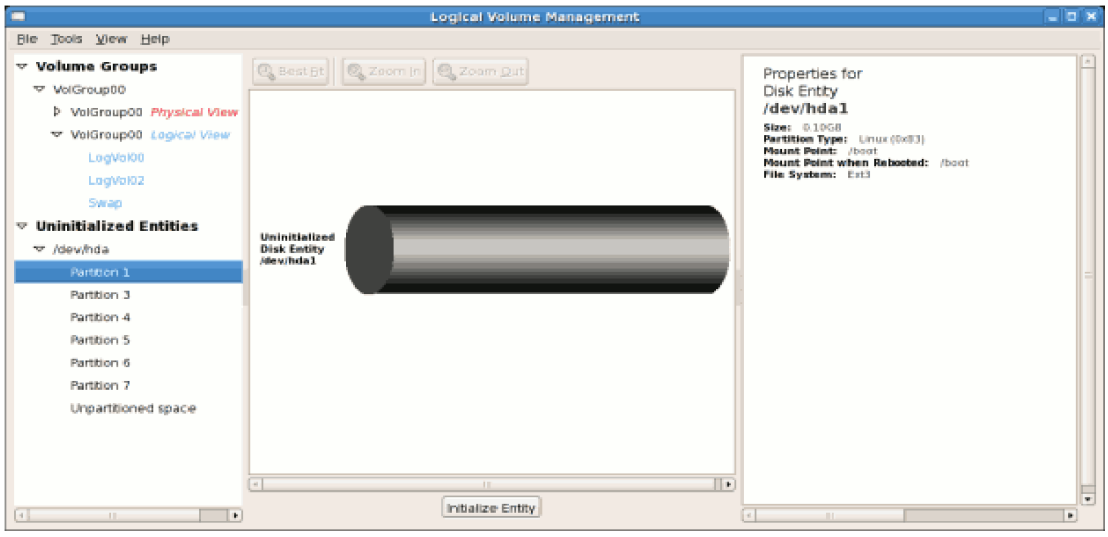


图 4-7 未初始化实体

在这个例子中，分区 3 将被初始化并添加到现有卷组。为了初始化一个分区或未分区的空间，请选择分区，然后单击**【初始化实体】**按钮。一旦初始化，该卷将被列在**【未分配卷】**列表中。

4.2.2 将未分配卷添加到卷组中

一旦初始化，该卷将被列在【**未分配卷**】列表中。下图说明了一个未分配的分区（分区 3）。在窗口底部的相应按钮可让您：

1） 创建一个卷组；

2） 添加一个未分配卷到一个现存卷组中；

3） 从 LVM 删除该卷。

要添加该卷到现有卷组，点击【**添加到现有卷组**】按钮。

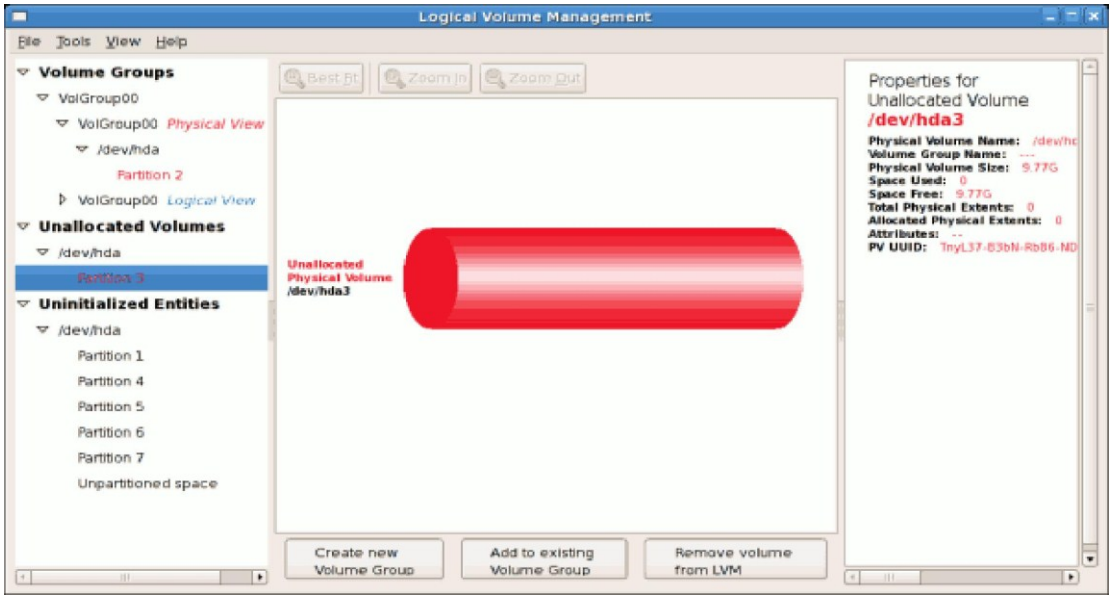


图 4-8 未分配卷组

点击【**添加到现有卷组**】按钮，将显示一个列出现有卷组的弹出窗口，您可以添加你想要初始化的物理卷到该卷组。一个卷组可能跨越一个或多个硬盘。在这个例子中，只有一个卷组中存在，如下所示。

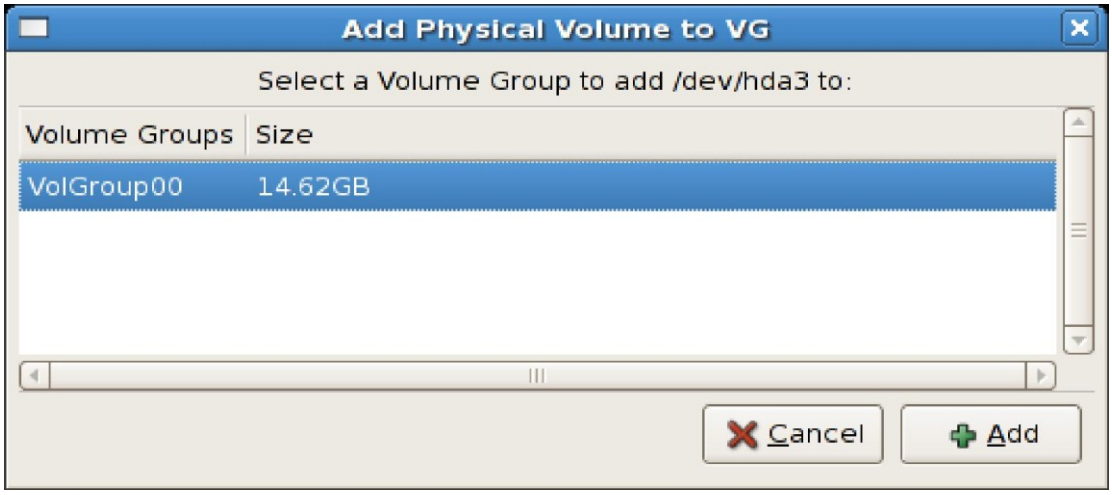


图 4-9 添加物理卷到卷组

一旦添加到现有卷组，新逻辑卷将自动添加到所选的卷组的未使用空间。您可以使用未使用的空间：

1） 创建一个新的逻辑卷（点击**【创建新的逻辑卷】**按钮），

2） 选择一个现有的逻辑卷并扩展区间（见章节 4.2.6 扩展卷组），

3） 选择一个现有的逻辑卷并点击**【删除所选的逻辑卷】**按钮将其从卷组中删除。请注意，你不能选择未使用的空间来执行此操作。

下图说明了添加新卷组后的“VolGroup00”逻辑视图。

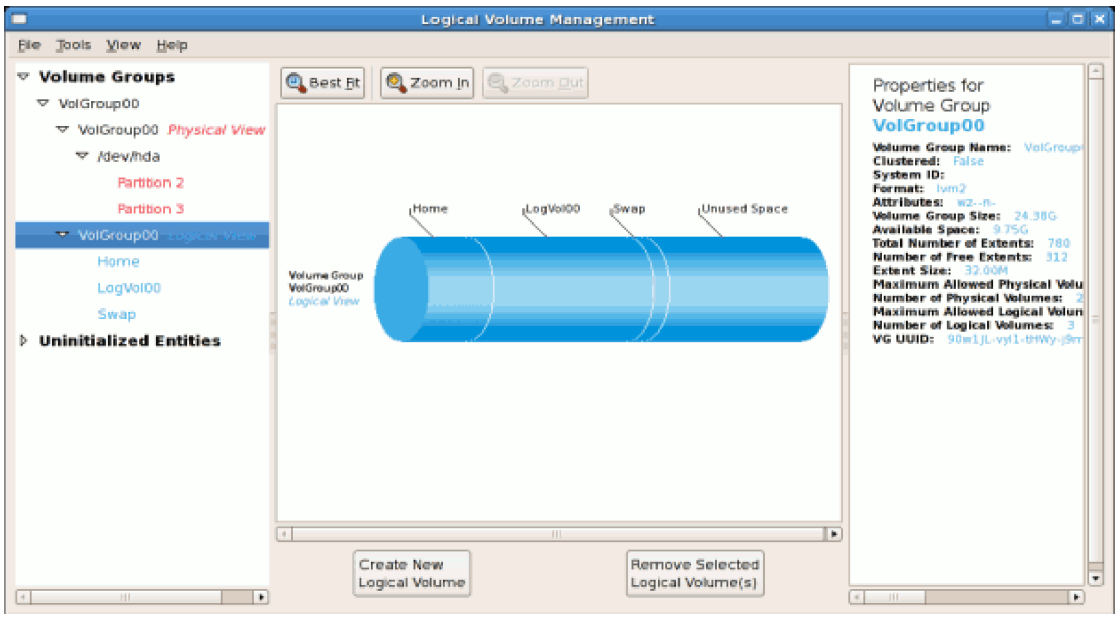


图 4-10 卷组的逻辑视图

在下图中，未初始化的实体（分区 3，5，6 和 7）被添加到“VolGroup00”。

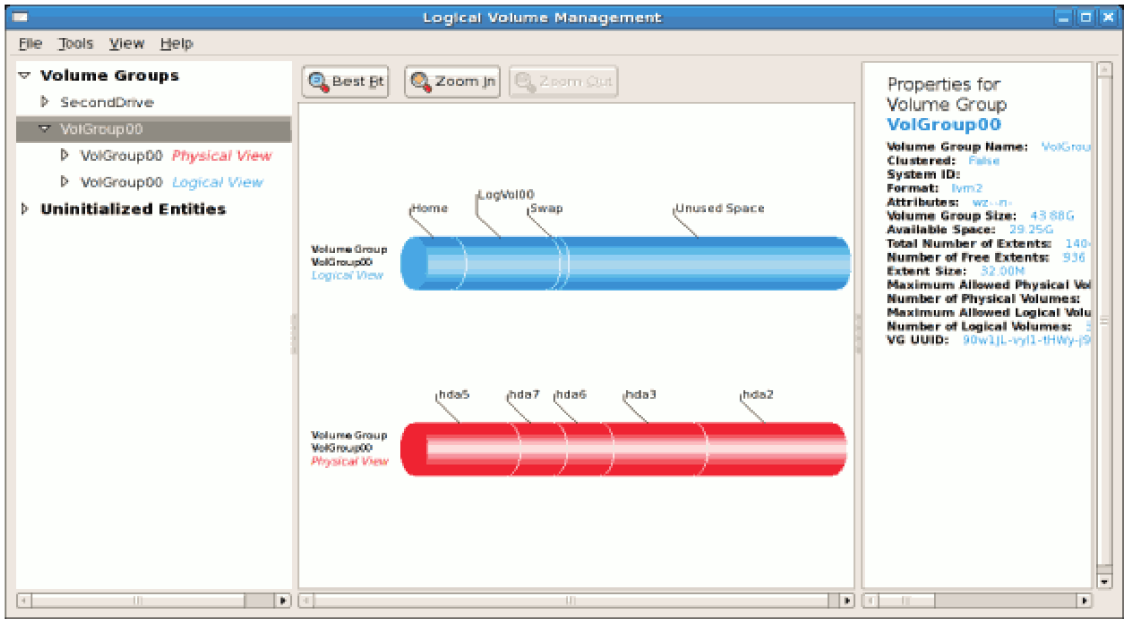


图 4-11 卷组的逻辑视图

4.2.3 迁移区间

若要从一个物理卷迁移区间，请选择该卷并点击**【卷组迁移所选区间】**钮。请注意，您需要有足够数量的自由区间才可以迁移卷组内的区间。如果你没有足够数量的自由区间，将显示一条错误消息。要解决这个问题，请扩展您的卷组（见“4.2.6 扩展卷组”）如果卷组中检测到足够数量的自由区间，将显示一个弹出窗口，从中您可以选择该区间的目的地或自动让 LVM 选择物理卷（PVs）以便将其迁移。如下所示：

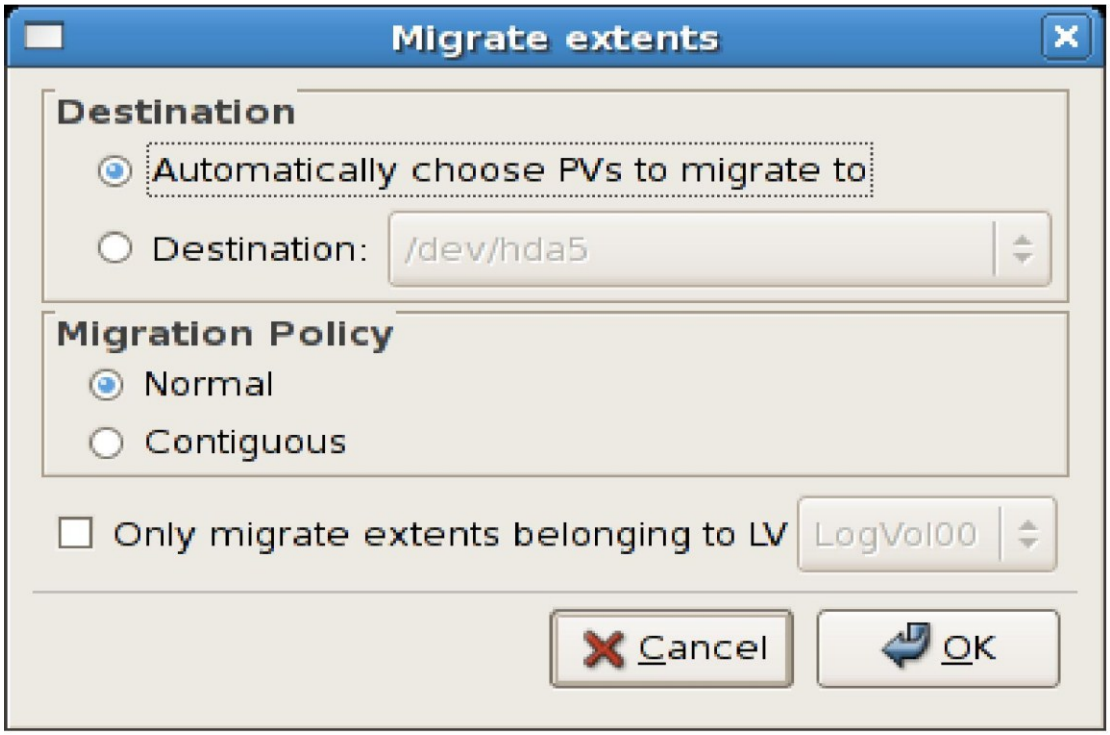


图 4-12 迁移区间

下图说明了一个进展中的区间迁移。在这个例子中，区间被迁移到“分区 3”。

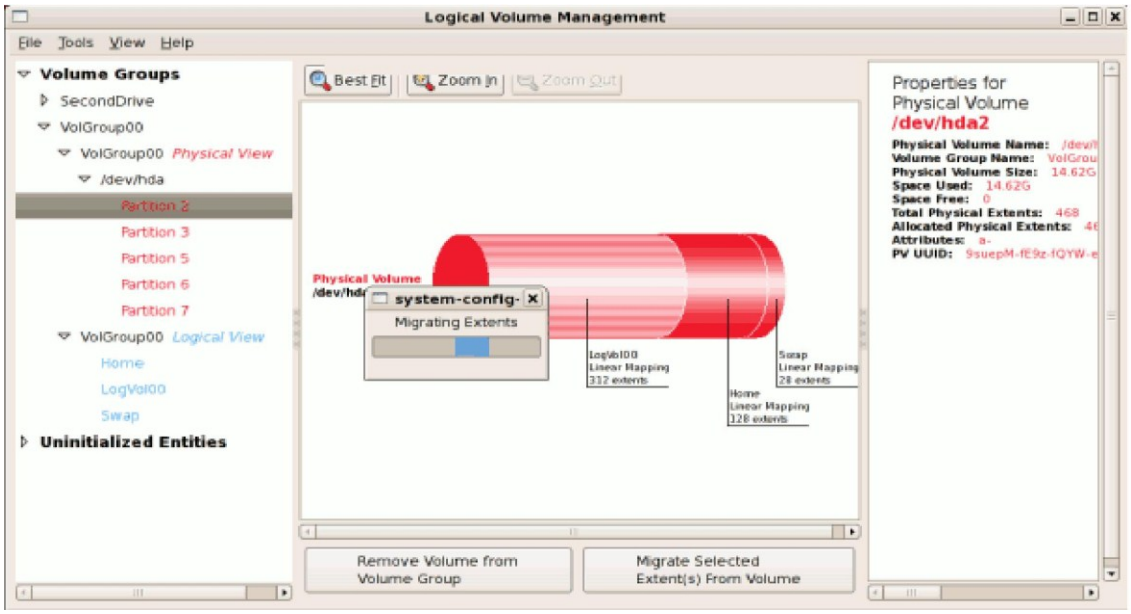


图 4-13 进行中的区间迁移

一旦区间已被迁移，未使用的空间留在物理卷上。下图说明了该卷组的物理视图和逻辑视图。请注意 LogVol00 的区间

迁移区间允许您在硬盘升级的情况下移动逻辑卷或更好地管理您的磁盘空

间。

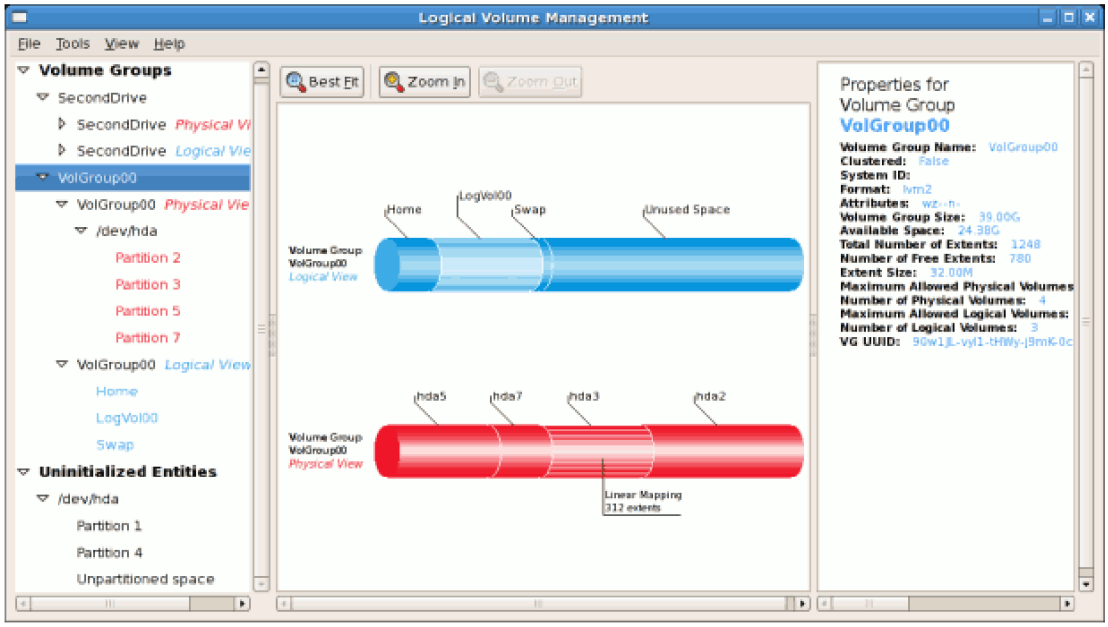


图 4-14 卷组的逻辑和物理视图

4.2.4 用逻辑卷管理 LVM 来添加新硬盘

在这个例子中，添加一个新的 IDE 硬盘。下图说明了新硬盘的详细信息。从下图可知，磁盘未被初始化也未被安装。要初始化一个分区，点击**【初始化实体】**按钮。更详细信息，见 4.2.1 使用未初始化实体。一旦初始化，LVM 将新卷组添加到图 4-8 所示的未分配卷列表中。

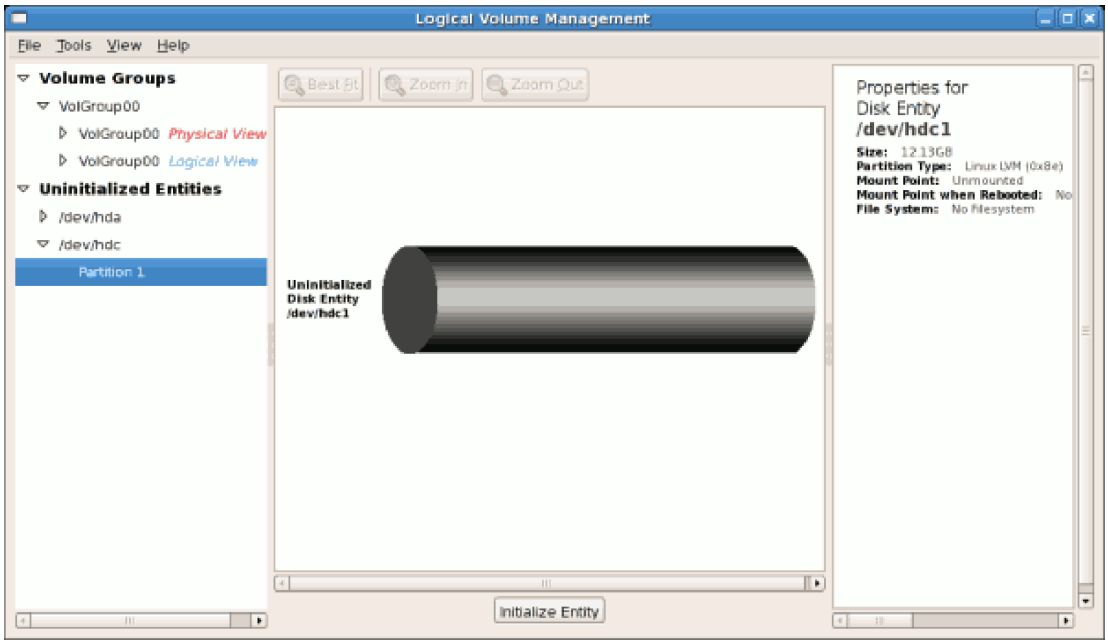


图 4-15 未初始化硬盘

4.2.5 添加一个新的卷组

一旦初始化，LVM 将添加新卷到未分配卷的列表，您可以把它添加到现有卷组或创建一个新卷组。您还可以从 LVM 删除该卷。如果从 LVM 中删除，该卷将在“未初始化实体列表”中列出，如图 4-7 所示。在这个例子中，一个新卷组将被创建，如下所示。

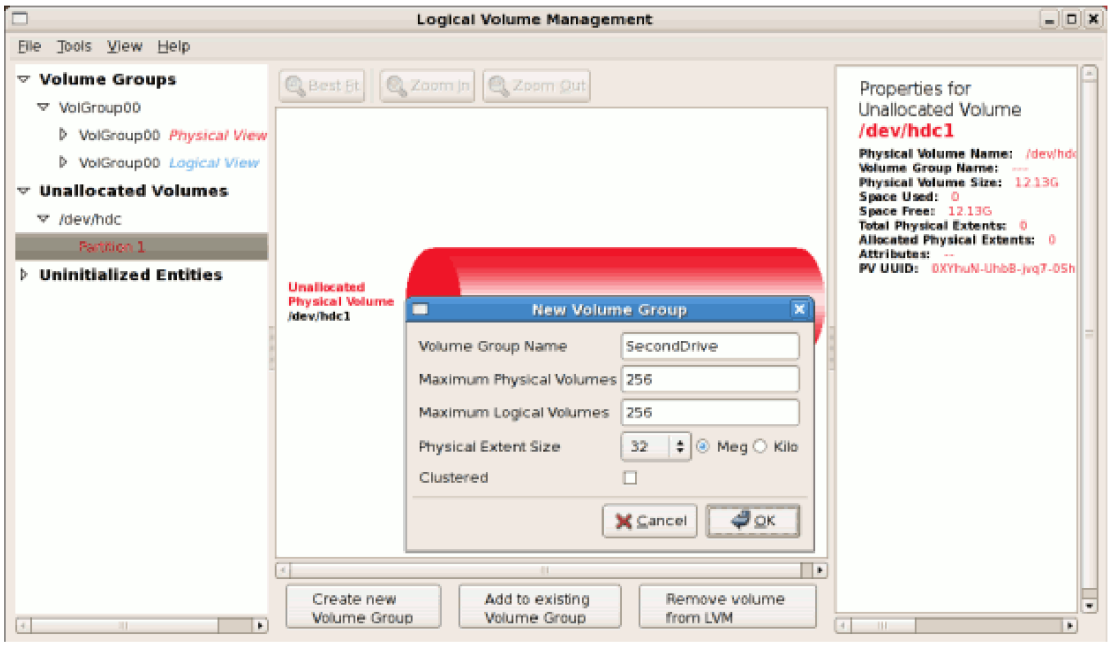


图 4-16 创建新卷组

一旦创建了一个新的卷组，它将显示在现有卷组列表中，如下所示。因为没有已创建的逻辑卷，该逻辑视图将显示带有未使用空间的卷组。要创建逻辑卷，选择卷组并单击**【创建新的逻辑卷】**按钮，如下图所示。请选择您要在卷组上使用的区间。在这个例子中，卷组中的所有区间被用来创建新的逻辑卷。

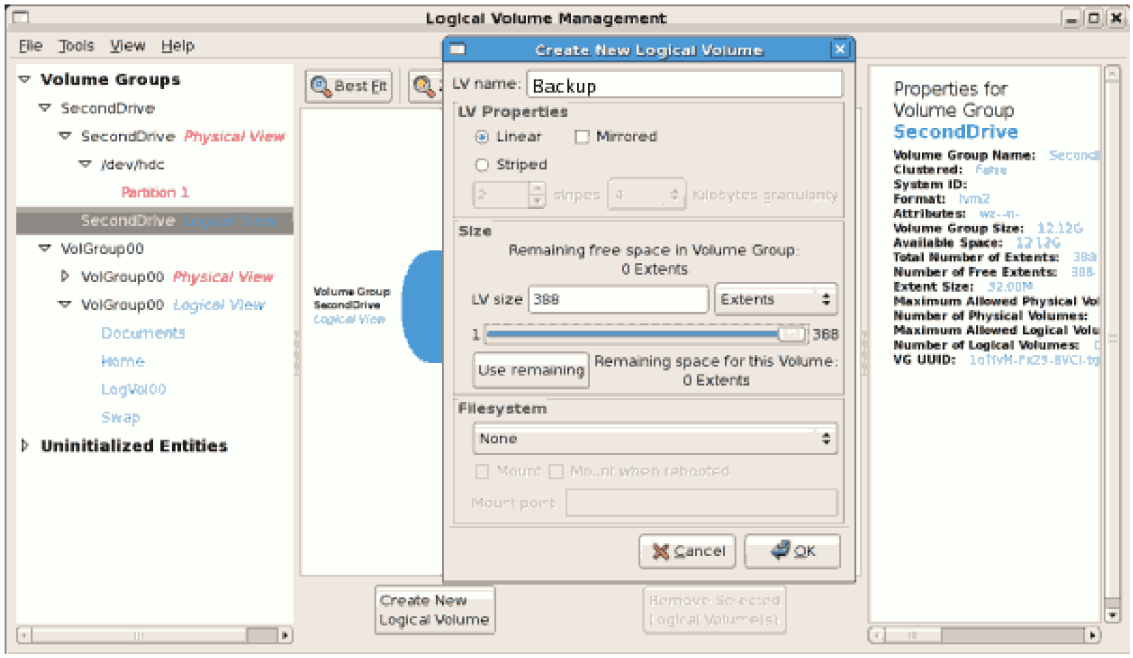


图 4-17 创建新逻辑卷

下图说明了新卷组的物理视图。名为“备份”的新逻辑卷也列出在此卷组中。

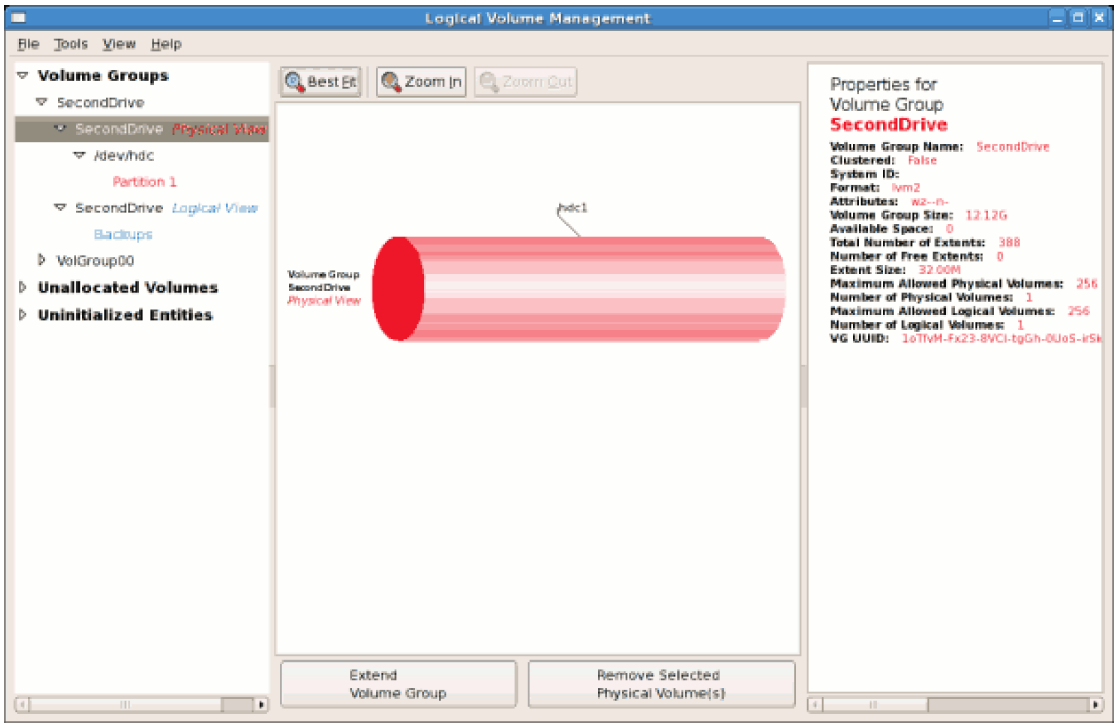


图 4-18 新卷组的物理视图

4.2.6 扩展卷组

在这个例子中，其目的是扩展新卷组，包括未初始化的实体（分区）。这旨

在增加卷组区间的大小或数量。要扩展卷组中，请单击**【扩展卷组】**按钮。

这将显示**【扩展卷组】**窗口，如下图所示。在**【扩展卷组】**窗口，你可以选择磁盘实体（分区）添加到该卷组。请确保您检查过所有“未初始化的磁盘实体”（分区）的内容，以避免删除任何关键数据（见图 4-15）。在这个例子中，磁盘实体（分区）/dev/hda6 的选择如下所示。

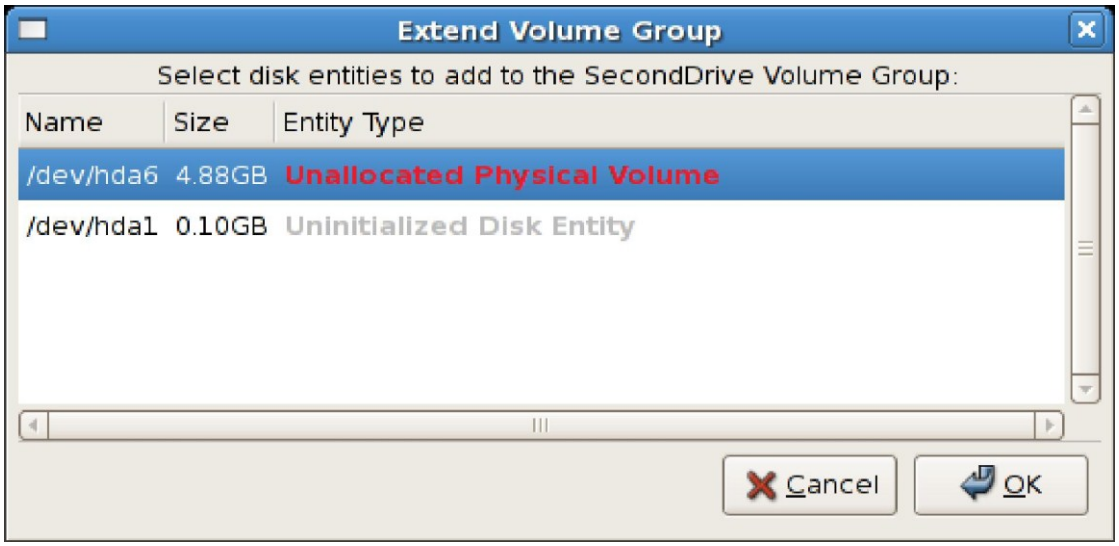


图 4-19 选择磁盘实体

添加完毕后，新卷将被添加到卷组中的“未使用的空间”。下图说明扩展后卷组的逻辑和物理视图。

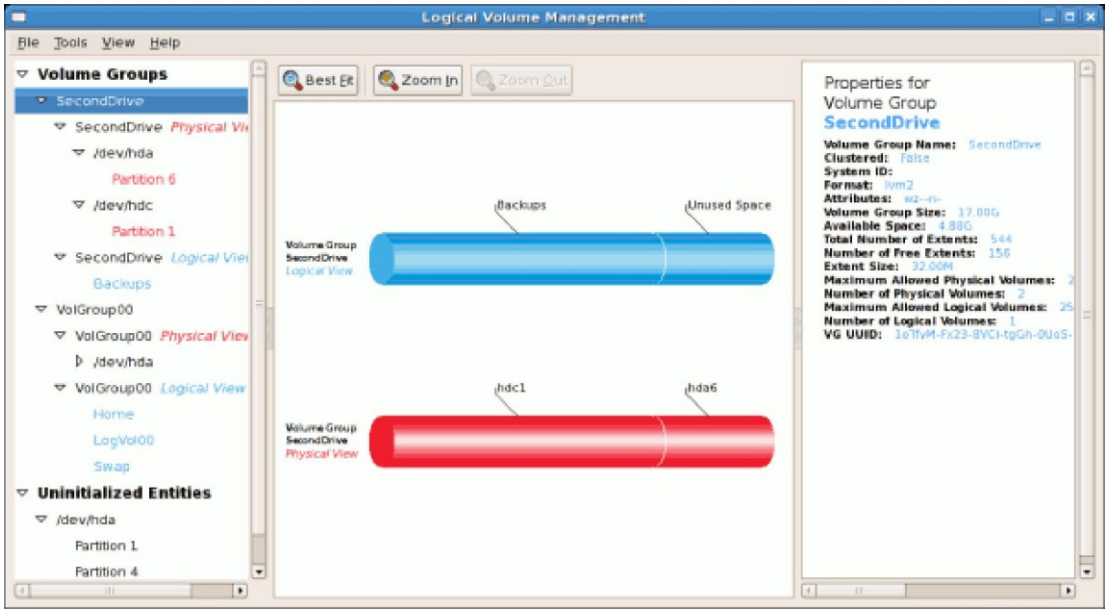


图 4-20 扩展卷组的逻辑和物理视图

4.2.7 编辑逻辑卷组

LVM 实用程序允许您选择卷组中的一个逻辑卷，并修改其名称，大小和指定文件系统选项。在这个例子中，名为“备份”的逻辑卷扩展到卷组的剩余空间。

点击【**编辑属性**】按钮，将显示【**编辑逻辑卷**】弹出窗口，从中可以编辑逻辑卷的属性。在此窗口中，你也可以在作出更改之后安装此卷并在系统重新启动时挂载它。请注意，您应该指明挂载点。如果您指定的挂载点并不存在，将显示一个弹出窗口，提示你去创建。【**修改逻辑卷**】窗口如下所示。

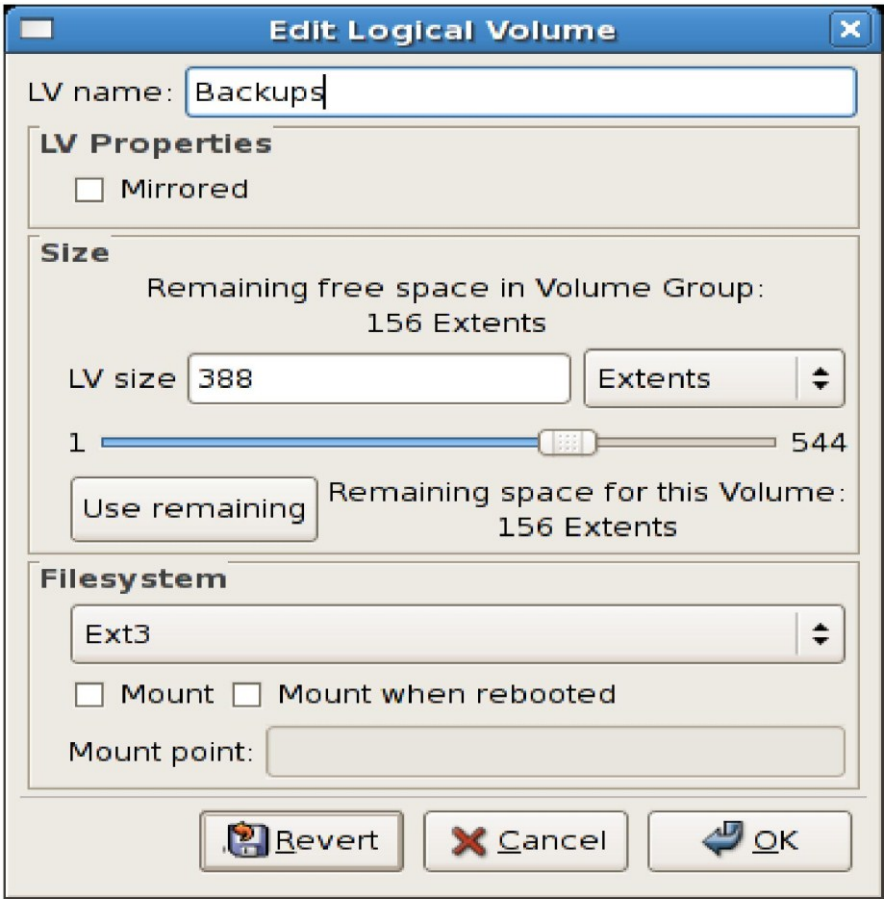


图 4-21 编辑逻辑卷

如果你想挂载卷，选择【**挂载**】复选框指明首选的挂载点。要在系统重新启动时挂载卷，请选择【**重启时挂载**】复选框。在这个例子中，新卷将被挂载在 /mnt/backups 中。如**错误!未找到引用源。**所示

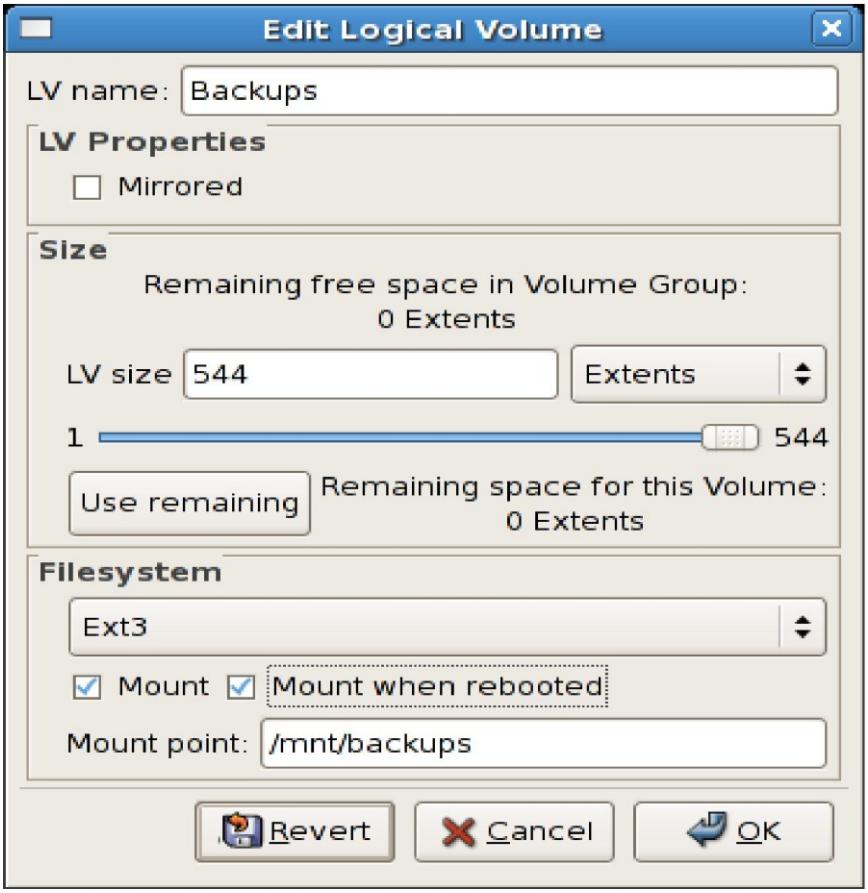


图 4-22 编辑逻辑卷-指定挂载选项

下图给出了逻辑卷被扩展到未使用空间后的该卷组的逻辑和物理视图。请注意在这个例子中，名为“备份”的逻辑卷跨越两个硬盘。通过使用 LVM 一个卷能够以带状跨越两个或两个以上的物理设备。

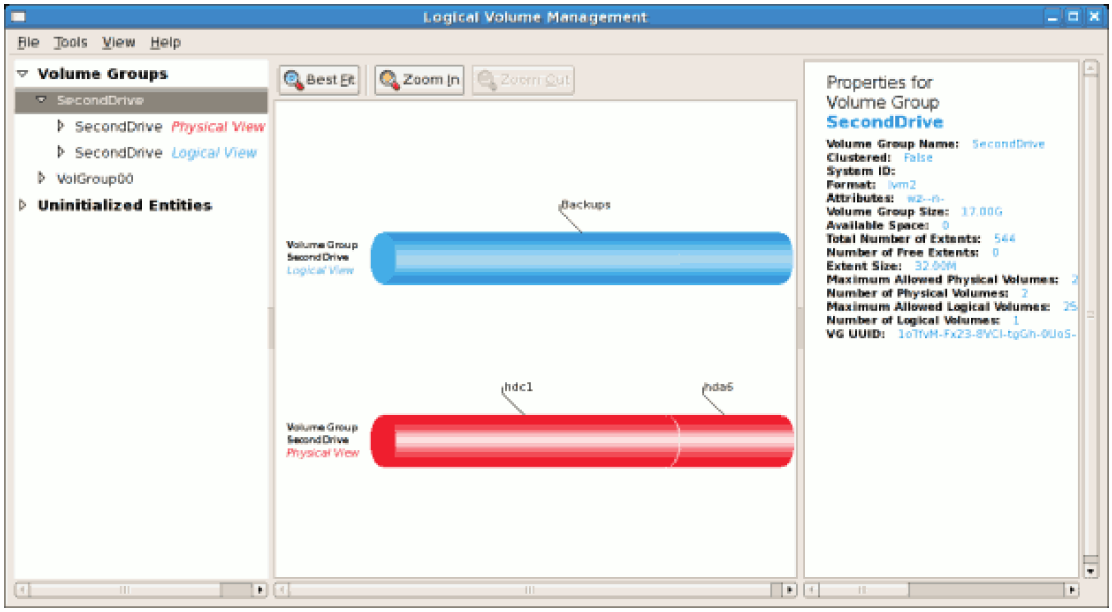


图 4-23 编辑逻辑卷

4.3 参考

使用这些源代码，以更多了解 LVM。

1） 安装了的文档

rpm -qd lvm2 — 此命令显示 LVM 包中所有文件，包括手册页。

lvm --help —此命令将显示所有可用的 LVM 命令。

2） 有用的网站

http://tldp.org/HOWTO/LVM-HOWTO/ —来自 Linux 文档项目的 LVM HOWTO。

5. 分区

实用程序 parted 允许用户：

1） 查看现有的分区表

2） 更改现有分区的大小

3） 从自由空间或附加硬盘驱动器来添加分区

默认情况下，在安装中标麒麟可信操作系统 V6.0 时 parted 软件包包括在内。要开始 parted，登录为根用户，在 shell 提示符后键入 parted /dev/sda 命令（这里的/dev/sda 是你要配置的驱动器的设备名称）。

如果你想删除或调整分区，该分区所在的设备不得使用。可能并不建议在使用中的设备上创建一个新的分区。

对于不在使用中的设备，设备上的任何分区都不可以安装，并且设备上的任何交换空间都禁止启用。

同时，在使用时分区表不应该被修改，这是因为内核可能无法正确识别这些更改。如果分区表不匹配已挂载分区的实际状态，信息可能写入错误的分区，造成数据丢失和被重载。

实现这一目标最简单的方式是在救援模式下启动您的系统。当提示挂载文件系统时，请选择“跳过”。

另外，如果驱动器不包含任何使用中的分区（使用或锁定的文件系统防止被卸载的系统进程），您可以使用 umount 命令卸载，并使用 swapoff 命令关闭所有交换空间的硬盘驱动器。

表 5-1 parted 命令包含了经常使用的 parted 命令列表。接下来的部分更详细

地解释了这些命令和参数。

表 5-1 parted 命令

命令	描述
check minor-num	执行文件系统的简单校验
cp from to	从一个分区复制文件系统到另一个分区；from and to 是分区的次号码
help	显示可用命令的列表
mklabel label	为分区表创建一个磁盘标签
mkfs minor-num file-system-type	创建一个类型 file-systemtype 的文件系统
mkpart part-type fs-type start-mb end-mb	分区而无需创建一个新的文件系统
mkpartfs part-type fs-type start-mb end-mb	建立一个分区，并创建指定的文件系统
move minor-num start-mb end-mb	移动分区
name minor-num name	仅命名 Mac 分区和 PC98 磁盘标签
print	显示分区表
quit	取消 parted
rescue start-mb end-mb	从 start-mb end-mb 抢救丢失的分区
resize minor-num start-mb end-mb	从 start-mb end-mb 调整分区
rm minor-num	删除分区
select device	选择不同的设备来进行配置
set minor-num flag state	设置分区标志，打开或关闭状态
toggle [NUMBER [FLAG]	切换分区 NUMBER 标志的状态
unit UNIT	为 UNIT 设置缺省单元

5.1 查看分区表

开始 parted 后，使用 print 打印命令来查看分区表。出现一个类似以下的表：

```
Model: ATA ST3160812AS (scsi)
Disk /dev/sda: 160GB
Sector size (logical/physical): 512B/512B
Partition Table:       msdos
Number   Start End  Size TypeFile system      Flags
```

```
1     32.3kB  107MB   107MB   primary   ext3 boot
2     107MB   105GB   105GB   primary   ext3
3     105GB   107GB   2147MB  primary   linux-swap
4     107GB   160GB   52.9GB  extended  root
5     107GB   133GB   26.2GB  logical   ext3
6     133GB   133GB   107MB   logical   ext3
7     133GB   160GB   26.6GB  logical         lvm
```

第一行包含的磁盘类型，制造商，型号和接口，第二行显示了磁盘标签类型。第四行下面剩余的输出显示了分区表。

在分区表中，Minor number 是分区编号。例如，号码为 1 的分区对应于 /dev/sda1。Start 开始和 End 结束值以 MB 为单位。有效的 Type 是元数据，自由的，初级的，扩展的或逻辑的。Filesystem 是该文件系统类型，它可以是以下的任何一项：

- ext2
- ext3
- fat16
- fat32
- hfs
- jfs
- linux-swap
- ntfs
- reiserfs
- hp-ufs
- sun-ufs
- xfs

如果一个设备的 Filesystem 显示为空值，这意味着它的文件系统类型是未知的。

Flags（标志）栏列出了为分区设置的标志。可用标志包括启动、根、交换、隐藏、raid、lvm 或 lba。

 提示：要选择不同的设备而无需重新启动 parted，在设备名称（例如，/dev/sda）后使用选择命令。这样做使您可以查看或配置设备的分区表。

5.2 创建一个分区

警告：不要尝试在一个正在使用中的设备创建分区。

在创建一个分区之前，启动进入救援模式（或卸载设备上的所有分区并关闭设备上所有交换空间）。

开始 parted，这里的/dev/sda 是要创建分区的设备：parted /dev/sda。

查看当前的分区表，以确定是否有足够的可用空间：print。

如果没有足够的自由空间，您可以调整现有分区。详细信息请参见“5.4 调整分区”。

5.2.1 进行分区

从分区表中，确定新的分区的起点和终点，确定它应该是什么分区类型。您只能让一个设备有四个主分区（没有扩展分区）。如果您需要四个以上的分区，您可以有三个主分区，一个扩展分区，和扩展分区内的多个逻辑分区。

例如，要在硬盘驱动器上创建一个 1024 兆到 2048 兆字节 ext3 文件系统主分区，键入以下命令：

```
mkpart primary ext3 1024 2048
```

提示：如果您改为使用 mkpartfs 命令，文件系统的创建会在分区创建之后完成。然而，parted 并不支持创建一个 ext3 文件系统。因此，如果你想创建一个 ext3 文件系统，请使用 mkpart 并且用下文描述的 mkfs 命令来创建该文件系统。

一旦你按下 Enter 键，变化就开始发生，因此在执行命令之前一定要仔细审查。

创建分区后，使用 print 命令以确认它是在分区表中，且分区类型，文件系统类型以及大小都正确。也请记住新分区的编号，以便您可以标签分区上的任何文件系统。您还应该查看执行 cat /proc/分区的输出，以确保该内核识别新的分区。

5.2.2 格式化并标记分区

该分区仍然没有一个文件系统。创建文件系统。

```
/sbin/mkfs -t ext3 /dev/sda6
```

警告：格式化分区会永久破坏分区上目前存在的的任何数据。

下一步，给分区上的文件系统贴上一个标签。例如，如果新分区上的文件系统是/dev/sda6，而且你想要为它贴上标签/work，请使用：

```
e2label /dev/sda6 /work
```

默认情况下，安装程序使用分区挂载点作为标签，以确保该标签是独一无二的。您可以使用你想要的任何标签。

之后，以 root 身份创建一个挂载点（如/work）。

5.2.3 添加到/etc/fstab 中

以 root 身份，编辑/etc/fstab 文件，包括使用分区 UUID 的新分区。使用 blkid -L label 命令检索分区的 UUID。新行应看起来类似于以下：

```
UUID=93a0429d-0318-45c0-8320-9676ebf1ca79   /work    ext3  defaults 1 2
```

第一栏应该包含文件系统 UUID，第二栏应该包含新分区的挂载点，下一列应该是文件系统类型（例如，ext3 或交换）。如果你需要更多关于格式的信息，用命令 man fstab 读取手册页。

如果第四栏是缺省的，该分区在启动的时候安装。无需重新启动安装分区，以 root 身份，键入命令：

```
mount /work
```

5.3 删除分区

警告：不要尝试在一个正在使用中的设备上删除分区。

在删除一个分区之前，开机进入救援模式（或卸载设备上所有分区并关闭设备上的所有交换空间）。

开始 parted，这里的/dev/sda 是所要删除分区上的设备：parted /dev/sda。

查看当前的分区表，以确定要删除分区的 minor 号：print。

用 rm 命令删除分区。例如，用 minor 号 3 删除分区：rm 3。

一旦你按下 Enter 键，变化就开始发生，因此在执行命令之前一定要仔细审查。

删除分区后，使用 print 命令以确认它已经从分区表中删除。您还应该查看下面命令的输出：

```
cat /proc/partitions
```

确保内核知道该分区已被删除。

最后一步是将它从/etc/fstab 文件中删除。找到声明所删除分区的这一行，并将它从文件中删除。

5.4 调整分区

 警告：不要尝试在一个正使用中的设备上调整分区。

在调整一个分区前，开机进入救援模式（或卸载设备上的所有分区并关闭设备上的所有交换空间）。

开始 parted，这里的/dev/sda 是所要调整分区上的设备：parted /dev/sda。

查看当前的分区表，以确定要调整分区的 minor 号以及该分区的起点和终点：print。

要调整分区的大小，紧随该分区 minor 号使用 resize 命令，以兆字节开始并以兆字节结束。例如：

resize 3 1024 2048

 警告：一个分区不能大于设备上的可用空间。

调整分区后，使用 print 命令来确认该分区大小已被正确地调整，且分区类型正确，文件系统类型正确。

重新启动系统进入正常模式后，使用 df 命令，以确保分区能被正确挂载并且识别出新的大小。

6. 文件系统结构

6.1 为什么要共享一个统一的结构？

文件系统结构是一个操作系统中的最基本的组织级别。几乎所有的操作系统与其用户之间的交互方式，应用程序和安全模式都依赖于操作系统如何来组织存储设备上的文件。提供一种通用的文件系统结构能确保用户和程序可以访问和写入文件。

文件系统将文件分为两种逻辑类型：

1） 共享与非共享文件

2） 可变与静态文件

可共享的文件既可以在本地访问也可以被远程主机访问，非共享文件仅适用于本地。可变文件，如文档，可以随时更改，而静态文件，如二进制文件，只有系统管理员才可以进行更改。

以这种方式分类文件有助于同每个文件的功能发生关联，每个文件都有包含这些文件的目录所具有的权限。操作系统和它的用户与文件的交互方式确定了文件所存放的目录，该目录是否具有只读或读/写权限，以及每个用户对该文件的访问级别。这个组织的顶层是至关重要的，对底层目录的访问可能会受到限制，否则如果自顶层以下的访问规则不遵守严格的结构，可能会出现安全问题。

6.2 文件系统层次标准概述(FHS)

中标麒麟可信操作系统 V6.0 使用文件系统层次标准（FHS）的文件系统结构，它为许多文件类型和目录定义了名称，位置和权限。

FHS 文档是任何 FHS 标准兼容文件系统的权威参考，但该标准留下了许多未定义或可扩展的领域。本节是该标准的一个概述同时也描述了本标准未涉及的文件系统部分。

FHS 兼容的两个最重要的元素是：

1） 与其他 FHS 兼容系统的兼容性

2） 挂载只读/usr/分区的能力。

这一点特别重要，因为/usr/包含常见的可执行文件，而且不应该由用户进行更改。此外，由于/usr/安装为只读，它应该能够从 CD- ROM 驱动器或其他机器通过 NFS 只读挂载来进行安装。

6.2.1 FHS 组织

这里指出的目录和文件是 FHS 文档所指定的目录和文件的一个很小的子集。http://www.pathname.com/fhs/。最完整的信息，请参阅最新的 FHS 文档。

6.2.1.1 收集文件系统信息

df 命令报告该系统的磁盘空间使用情况。它的输出类似于以下界面：

```
Filesystem   1K-blocks    Used Available Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
11675568       6272120       4810348       57% /    /dev/sda1
100691 9281      86211         10%    /boot
none     322856  0     322856           0%    /dev/shm
```

默认情况下，df 显示以千字节块为单位的分区大小，和以千字节为单位的已用的/可用的磁盘空间数量。要查看兆字节和千兆字节的信息，使用命令 df -h 参数代表“容易理解的”格式。df -h 的输出看起来类似于以下内容：

```
Filesystem   Size Used Avail Use% Mounted on
/dev/mapper/VolGroup00-LogVol00
12G    6.0G    4.6G    57% /    /dev/sda1
     99M    9.1M   85M    10%    /boot
none           316M   0    316M      0%      /dev/shm
```

备注：挂载的分区/dev/shm 表示系统的虚拟内存文件系统。

du 命令显示目录中的文件所正在使用的空间的估计大小，显示每个子目录的磁盘使用情况。du 命令输出的最后一行显示该目录总的磁盘使用情况，如要使输出更容易理解，请使用 du –hs 命令。对于更多的选择，请参考 man du 命令。

要以视图方式查看系统的分区和磁盘空间使用情况，请通过点击**【应用程序】=>【系统工具】=>【系统监视器】**或使用 gnome-system-monitor 命令来使用 Gnome 系统监控器（Gnome System Monitor）。选择**【文件系统】**选项卡以查看系统的分区。下图说明了**【文件系统】**选项卡。

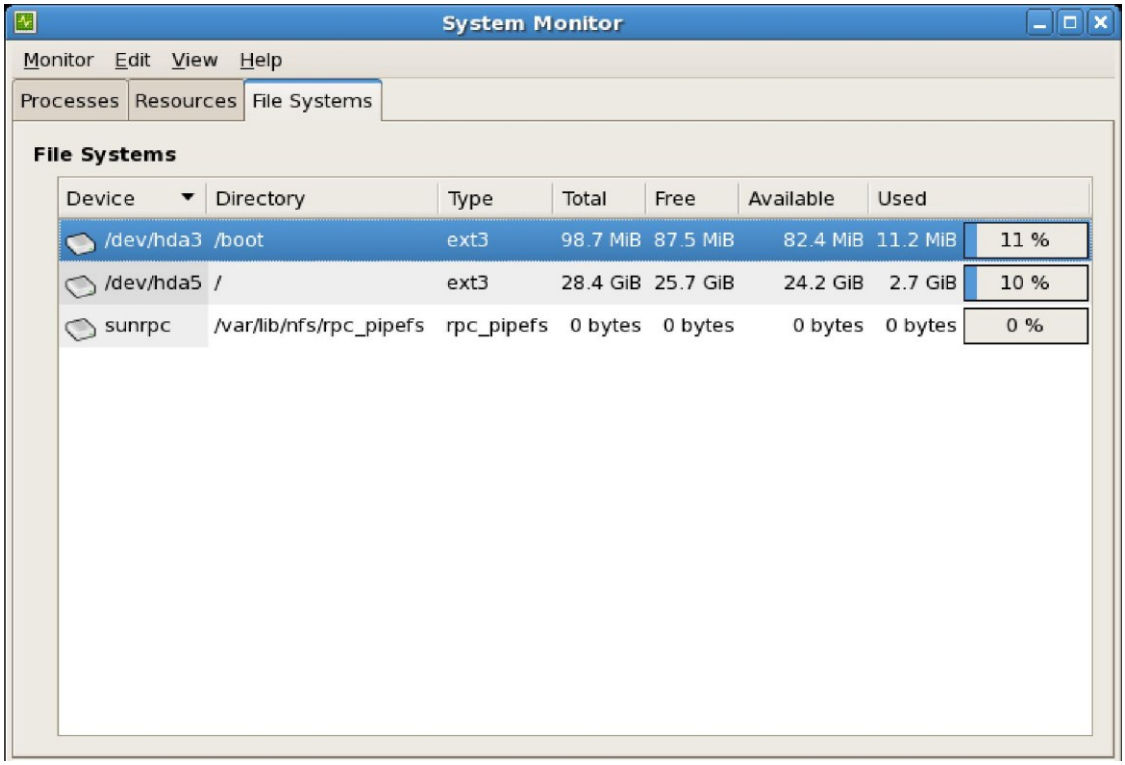


图 6-1 GNOME 系统监控文件系统选项卡

6.2.1.2 /boot/目录

/boot/目录包含启动系统所需的静态文件，如 Linux 内核。这些文件对系统的正确启动至关重要。

警告：不要删除/boot/目录。这样做可能使系统无法启动。

6.2.1.3 /dev/目录

/dev/目录包含代表以下设备类型的设备节点：

1） 连接到系统的设备

2） 由内核提供的虚拟设备

这些设备节点对系统功能的正常发挥起着至关重要。udevd 守护进程在/dev/按要求创建和删除设备节点。

在/dev/目录和子目录的设备要么是字符设备（只提供一串输入/输出串行数据流，例如鼠标或键盘）要么就是块设备（随机访问，如硬盘驱动器，软盘驱动器）。如果你安装了 GNOME 或 KDE，某些存储设备在连接（例如通过 USB）或插入（例如，通过 CD 或 DVD 驱动器）时可以被自动检测到，并且出现一个显示该内容的弹出窗口。

表 6-1 /dev/中常用文件的例子

文件	描述
/dev/hda	主要 IDE 通道上的主设备。
/dev/hdb	主要 IDE 通道上的从设备。
/dev/tty0	第一个虚拟控制台。
/dev/tty1	第二个虚拟控制台。
/dev/sda	主 SCSI 或 SATA 通道上的第一台设备。
/dev/lp0	第一个并口。

6.2.1.4 /etc/目录

为本地机器的配置文件保留/etc/目录。它应该不包含任何二进制文件，任何二进制文件应该被移到/bin/或/sbin/。

例如，/etc/skel/目录存储“框架”用户档案，用来在用户首次创建时填充主目录。应用程序也在这个目录中存储配置文件并且在执行时可以参考。/etc/exports 文件控制将哪些文件系统导出到远程主机。

6.2.1.5 /dev/lib 目录

/lib/目录应该只包含执行/bin/和/sbin/目录下的二进制文件所需要的库。这些共享的图片库是用来启动系统，或执行根目录文件系统中的命令。

6.2.1.6 /media/目录

/media/目录包含子目录作为移动介质，如 USB 存储介质，DVD 光盘，光盘，Zip 磁盘的挂载点。

6.2.1.7 /mnt/目录

/mnt/目录保留给暂时安装的文件系统，如 NFS 文件系统挂载。对于所有可移动的存储介质，请使用/media/目录。自动检测到的移动介质将被安装在/media 目录。

备注：/mnt 目录不得用于安装程序。

6.2.1.8 /opt/目录

/opt/目录通常为不属于默认安装部分的软件和附加包而保留。一个安装到/opt/的软件包创建一个同名的目录，如/opt/packagename/。在大多数情况下，这样的软件包都遵循一个可预见的子目录结构，大多数会将它们的二进制文件存储在/opt/packagename/bin/以及将它们的手册页 man pages 存储在/opt/packagename/man/中，等等。

6.2.1.9 /proc/目录

/proc/目录包含特殊的文件，这些特殊的文件可以从内核中提取信息或将信息发送到内核。这些信息的例子包括系统内存，CPU 信息和硬件配置。更多关于/proc/的信息，请参考”6.4/proc 虚拟文件系统”。

6.2.1.10 /sbin/目录

/sbin/目录中存储对于启动，恢复，还原或修复系统至关重要的二进制文件。/sbin/中的二进制文件要有 root 权限才能使用。此外，/sbin/目录包含了/usr/目录安装前系统所使用的二进制文件，/usr/安装之后所使用的任何系统实用程序通常被放置在/usr/sbin/中。

至少，以下程序，应存放在/sbin/中：

1） arp
2） clock
3） halt
4） init
5） fsck.*
6） grub
7） ifconfig
8） mingetty
9） mkfs.*
10） mkswap
11） reboot
12） route
13） shutdown
14） swapoff
15） swapon

6.2.1.11 /srv/目录

/srv/目录包含中标麒麟可信操作系统 V6.0 所服务的站点具体数据。此目录为用户提供了用于特定服务数据文件的位置，如 FTP，WWW 或 CVS。仅涉及到一个特定用户的数据，应该在/home/目录中。

6.2.1.12 /sys/目录

/sys/目录采用了 2. 6 内核特定的新的 sysfs 虚拟文件系统。随着对 2.6 内核

中热插拔硬件设备支持的增加，/sys/目录包含类似于/proc/所含有的信息，但却显示了热插拔设备具体设备信息的分层视图。

6.2.1.13 /usr/目录

/usr/目录适用于可以跨多台机器共享的文件。/usr/目录往往只是在它自己的分区上，并以只读方式挂载。至少，/usr/应包含以下子目录：

1） /usr/bin，用于二进制文件

2） /usr/etc，用于全系统的配置文件

3） /usr/games

4） /usr/include，用于 C 头文件

5） /usr/kerberos，用于与 Kerberos 相关的二进制文件和相关文件

6） /usr/lib，用于未被设计成可为 shell 脚本或用户可直接使用的目标文件和库

7） /usr/libexec，包含由其他程序调用的程序小帮手

8） /usr/sbin，存储系统管理不属于/sbin/的二进制文件

9） /usr/share，存储非特定体系结构的文件

10） /usr/src，存储源代码

11） /usr/tmp->/var/tmp

/usr/目录还应该包含一个/local/子目录。按照 FHS，这个子目录是由系统管理员在本地安装软件时所使用，并且应该在系统更新时不被重写。/usr/local 目录有一个类似于/usr/的结构，并且包含以下子目录：

1） /usr/local/bin

2） /usr/local/etc

3） /usr/local/games

4） /usr/local/include

5） /usr/local/lib

6） /usr/local/libexec

7） /usr/local/sbin

8） /usr/local/share

9） /usr/local/src

中标麒麟可信操作系统 V6.0/usr/local/用法与 FHS 略有不同。FHS 说明/usr/local/应该用来存储在系统软件升级时应保持安全的软件。由于 RPM 软件包

管理器可以安全地进行软件升级，因此没有必要通过把文件保存在/usr/local/中来保护文件。

相反，中标麒麟可信操作系统 V6.0 使用/usr/local/将软件安装到本地机器。例如，如果/usr/目录以一个只读 NFS 共享的方式从远程主机进行安装，它仍然有可能将软件包或程序安装在/usr/local/目录。

6.2.1.14 /var/目录

由于 FHS 需要 Linux 以只读方式安装/usr/，写入日志文件，或需要/spool/或 lock/目录的任何程序应写入到/var/目录。FHS 说明/var/适合可变数据文件，其中包括/spool/目录文件，日志数据，瞬态/临时文件之类的。

下面是在/var/目录中发现一些目录：

1） /var/account/

2） /var/arpwatch/

3） /var/cache/

4） /var/crash/

5） /var/db/

6） /var/empty/

7） /var/ftp/

8） /var/gdm/

9） /var/kerberos/

10） /var/lib/

11） /var/local/

12） /var/lock/

13） /var/log/

14） /var/mail->/var/spool/mail/

15） /var/mailman/

16） /var/named/

17） /var/nis/

18） /var/opt/

19） /var/preserve/

20） /var/run/

21） /var/spool/

22） /var/tmp/

23） /var/tux/

24） /var/www/

25） /var/yp/

系统日志文件，如 messages 和 lastlog，保存在/var/log/目录中。/var/lib/rpm/目录包含 RPM 系统数据库，将文件锁定在/var/lock/目录中。

/var/spool/目录下包含为某些程序保存数据文件的子目录。这些子目录包括：

1） /var/spool/at/

2） /var/spool/clientmqueue/

3） /var/spool/cron/

4） /var/spool/cups/

5） /var/spool/exim/

6） /var/spool/lpd/

7） /var/spool/mail/

8） /var/spool/mailman/

9） /var/spool/mqueue/

10） /var/spool/news/

11） /var/spool/postfix/

12） /var/spool/repackage/

13） /var/spool/rwho/

14） /var/spool/samba/

15） /var/spool/squid/

16） /var/spool/squirrelmail/

17） /var/spool/up2date/

18） /var/spool/uucp

19） /var/spool/uucppublic/

20） /var/spool/vbox/

6.3 中标麒麟可信操作系统 V6.0 特殊文件位置

中标麒麟可信操作系统 V6.0 略微扩展了 FHS 结构以适应特殊文件。

大多数涉及到 RPM 的文件存放在/var/lib/rpm/目录中。关于 RPM 的更多信息，请参阅 man rpm。

/var/cache/yum/目录包含 Package Updater 所使用的文件，包括系统的 RPM 头信息。这个位置也可能用于暂时存放更新系统时下载的 RPMs。

中标麒麟可信操作系统 V6.0 的另一个特定位置是在/etc/sysconfig/目录中。这个目录存储各种配置信息。许多在启动时运行的脚本使用此目录中的文件。

6.4 /proc 虚拟文件系统

与大多数文件系统不同，/proc 既不包含文本也不包括二进制文件。相反，它包含虚拟文件，因此，/proc 通常称为虚拟文件系统。这些虚拟文件的大小通常是零字节，即使它们包含了大量的信息。

就其本身而论，/proc 文件系统不用于存储。其主要目的是为硬件，内存和正在运行的进程，以及其他系统组件提供一个基于文件的接口。您可以通过查看相应的/proc 文件来检索多个系统组件的实时信息。/proc 中的某些文件也可以被（用户和应用程序）操纵来配置内核。

在管理和监控系统存储方面，相关/proc 文件如下：

1） /proc/devices

显示当前配置的各种字符和块设备

2） /proc/filesystems

列出所有目前由内核支持的文件系统类型

3） /proc/mdstat

包含系统上的多个磁盘或 RAID 配置的目前信息，如果它们存在的话

4） /proc/mounts

列出系统目前使用的所有挂载设备

5） /proc/partitions

包含分区块的分配信息

关于/proc 文件系统的更多信息，请参阅中标麒麟可信操作系统 V6.0 系统管理员手册。

7. 使用 mount 命令

在 Linux，UNIX 和类似的操作系统中，不同的分区和可移动设备（例如：光盘，DVD 或 USB 闪存驱动器）上的文件系统可以附加到目录树中的某一个点（挂载点），然后再分离。附加或分离文件系统，请使用 mount 或 umount 命令。

本章介绍了这些命令的基本用法，以及一些高级主题，如移动挂载点或创建共享子树。

7.1 列出当前安装的文件系统

要显示所有当前附加的文件系统，请运行 mount 命令而无需额外的参数：

```
mount
```

此命令显示了已知挂载点的列表。每一行提供下面表格中所列的设备名称，文件系统类型，文件所安装的目录以及相关挂载选项的重要信息：

目录装置的类型键入（选项）

中标麒麟可信操作系统 V6.0 还包括 findmnt 实用工具，它允许用户列出树状列表中的挂载文件系统。要显示所有当前附加的文件系统，请运行 findmnt 命令而无需额外的参数：

```
findmnt
```

7.1.1 指出文件系统类型

默认情况下，mount 命令的输出包括各种虚拟文件系统，如 sysfs 和 tmpfs。如果要仅显示具有一定的文件系统类型的设备，请在命令行选择-t 选项：

```
mount -t type
```

同样，如果要只显示具有一定的文件系统类型的设备，请使用 findmnt 命令，键入：

```
findmnt -t type
```

对于常见的文件系统类型清单，请参阅表 7.1“常见文件系统类型”。关于用法的例子，参见示例 7.1“列出当前挂载的 ext4 文件系统”。

示例 7.1，列出当前安装的 ext4 文件系统

通常情况下，/和/boot 分区被格式化来使用 ext4。如果仅仅要显示使用该文件系统的挂载点，请在 shell 提示后下键入以下内容：

```
—]$ mount -t ext4
/dev/sda2 on / type ext4 (rw)
/dev/sda1 on /boot type ext4 (rw)
```

使用 findmnt 命令列出挂载点，键入以下内容：

```
—]$ findmnt -t ext4
TARGET SOURCE FSTYPE OPTIONS
/ /dev/sda2 ext4 rw, realtime, seclabel, barrier=1, data=ordered
```

```
/boot  /dev/sda1  ext4    rw, realtime, seclabel, barrier=1, data=ordered
```

7.2 安装文件系统

要附加一定的文件系统，请在下表中使用 mount 命令：

```
mount [option...] device directory
```

该设备可以通过块设备的完整路径（如“/dev/sda3”），通用的唯一标识符（UUID，例如，“UUID=34795a28-ca6d-4fd8-a347-73671d0c19cb”）或卷标（例如，“LABEL=home”）来进行标识。

当所有需要的信息都不可用时（即无设备名称，无目标目录，或无文件系统类型）时运行 mount 命令，它会读取/etc/fstab 配置文件的内容看是否给定的文件系统被列在其中。此文件列出了设备名称和选定文件系统所应安装的目录，以及文件系统类型和挂载选项。正因为如此，当安装在该文件中指定的文件系统时，您可以使用下列命令的变体之一：

```
mount [option...] directory
mount [option...] device
```

请注意挂载该文件系统需要的权限，除非该命令以 root 身份运行（见 7. 2. 2 节，“指定安装选项”）。

 备注：确定一个特定设备的 UUID 和标签

要确定一个特定设备的 UUID 和标签（如果设备使用的话），请以下列形式使用 blkid 命令：

```
blkid device
```

例如，要显示/dev/sda3 的信息，键入以下内容：

```
—]# blkid /dev/sda3
/dev/sda3: LABEL="home" UUID="34795a28-ca6d-4fd8-a347-73671d0c19cb"
TYPE="ext3"
```

7.2.1 指定文件系统类型

在大多数情况下，mount 自动检测文件系统。不过，也有某些文件系统不能被认可并且需要手动指定，如 NFS（网络文件系统）或 CIFS（通用互联网文件系统）。要指定文件系统类型，请在下表中使用 mount 命令：

```
mount -t type device directory
```

表 7-1 提供了可以使用 mount 命令的常用文件系统类型的列表。对于所有可用文件系统类型的完整列表，请查阅章节 7. 4. 1 手册页文档相关的手册页作为

参考。

表 7-1 常用文件系统类型

类型	描述
ext2	ext2 文件系统
ext3	ext3 文件系统
ext4	ext4 文件系统
iso9660	ISO 9660 文件系统。经常被光学介质所使用，通常为 CD。
jfs	由 IBM 创建的 JFS 文件系统。
nfs	NFS 文件系统。它通常用来访问网络上的文件。
nfs4	NFSv4 文件系统。它通常用来访问网络上的文件。
ntfs	NTFS 文件系统。它经常被用在运行 Windows 操作系统的机器上。
udf	UDF 文件系统。经常被光学介质所使用，通常为 CD。
vfat	FAT 文件系统。它被经常用在运行 Windows 操作系统的机器上以及某些数字媒体，如 USB 闪存驱动器或软盘。

用法实例请参见例 7. 2“安装 USB 闪存驱动器”。

示例 7.2，安装 USB 闪存驱动器

旧的USB闪存驱动器经常使用FAT文件系统。假设这种驱动器使用/dev/sdc1设备并且/media/flashdisk/目录存在，通过以 root 用户身份在 shell 提示后键入以下内容把它安装到这个目录中：

```
~]# mount -t vfat /dev/sdc1 /media/flashdisk
```

7.2.2 指定 Mount 选项

要指定文件系统类型，请在下表中使用该命令：

```
mount -o options device directory
```

提供多个选项时，不要在逗号后插入空格，否则mount会将空格后的值错误地解释为额外的参数。

表 7-2 提供常见挂载选项的列表。对于所有可用选项的完整列表，请查阅“7. 4. 1 手册页文档”相关的手册页作为参考。

表 7-2 常见的 mount 选项

选项	描述
Async	允许文件系统的异步输入/输出操作。
auto	允许使用 mount -a 命令自动挂载文件系统。
defaults	为 async、auto、dev、exec、nouser、rw、suid 提供别名。
exec	允许二进制文件在特定的文件系统上执行。

选项	描述
loop	挂载一个映像作为循环设备。
noauto	不允许使用 mount -a 命令的文件系统进行自动挂载。
noexec	不允许二进制文件在特定的文件系统上执行。
nouser	不允许普通用户（即 root 用户除外）挂载和卸载文件系统。
remount	在已经挂载的情况下，重新挂载文件系统。
ro	以只读方式挂载文件系统。
rw	以读写模式挂载文件系统。
user	允许普通用户（即 root 用户除外）挂载和卸载该文件系统。

用法实例请参见例 7.3“安装 ISO 映像”。

示例 7.3，挂载 ISO 映像

ISO 映像（或一般的磁盘映像）可以通过使用循环设备进行挂载。假设 Fedora14 挂载光盘的 ISO 映像出现在当前工作目录中而且/media/cdrom/目录存在，通过以 root 身份运行以下命令来挂载此映像到该目录中：

```
~]# mount -o ro,loop Fedora-14-x86_64-Live-Desktop.iso /media/cdrom
```

请注意，ISO9660 是一个只读文件系统的设计。

7.2.3 共享 Mounts

有时，某些系统管理任务需要从目录树中多个地方访问相同的文件系统（例如，当准备 chroot 环境时）。为了满足这些要求，mount 命令执行--bind 选项以提供一种复制某些挂载的方法。用法如下：

```
mount --bind old_directory new_directory
```

虽然这个命令允许用户从两个位置来访问文件系统，但它并不适用于安装在原始目录内的文件系统。如果还要包括这些挂载，请键入以下内容：

```
mount --rbind old_directory new_directory
```

此外，为了提供尽可能多的灵活性，中标麒麟可信操作系统 V6.0V6.0 执行名为共享子树的功能。此功能允许使用以下四个挂载类型：

共享挂载

共享挂载允许为给定的挂载点创建一个精确副本。当挂载点被标记为共享挂载时，原来挂载点的任何挂载在其中都得以体现，反之亦然。要将一个挂载点的类型更改为共享挂载，在 shell 提示下键入以下内容：

```
mount --make-shared mount_point
```

另外，要改变选定的挂载点以及其下的所有挂载点的挂载类型，请输入：

```
mount --make-rshared mount_point
```

用法实例请参见例 7.4“安装共享挂载点”。

示例 7.4：创建共享挂载点

通常有两个位置用来挂载其他文件系统：/media 目录用于可移动介质，而/mnt 目录用于暂时挂载的文件系统。通过使用一个共享的挂载，您可以使这两个目录共享相同的内容。要做到这一点，请以 root 的身份将/media 目录标记为“共享”：

```
-]# mount --bind /media /media
-]# mount --make-shared /media
```

然后使用以下命令在/mnt 中创建副本：

```
-]# mount --bind /media /mnt
```

现在可以验证/media 中的挂载也出现在/mnt 内。例如，如果 CD-ROM 驱动器中包含非空媒体而且/media/cdrom/目录存在，请运行以下命令：

```
-]# mount /dev/cdrom /media/cdrom
-]# ls /media/cdrom
EFI GPL isolinux LiveOS
-]# ls /mnt/cdrom
EFI GPL isolinux LiveOS
```

同样，也可以验证/mnt 目录中挂载的任何一个文件系统在/media 中都有反映。例如，如果一个使用/dev/sdc1 设备的非空 USB 闪存驱动器被插入而且/mnt/flashdisk/目录存在，请键入以下命令：

```
-]# mount /dev/sdc1 /mnt/flashdisk
~]# ls /media/flashdisk
en-US publican.cfg
~]# ls /mnt/flashdisk
en-US publican.cfg
```

从挂载

从挂载允许为给定的挂载点创建一个受限副本。当挂载点被标记为从挂载时，原始挂载点的任何挂载在其中都得以体现，但从挂载内没有一个挂载会被映射到原始挂载点中。要将一个挂载点的类型更改为从挂载，在 shelll 提示下键入以下内容：

```
mount --make-slave mount_point
```

另外，要改变选定的挂载点以及其下的所有挂载点的挂载类型，请输入：

```
mount --make-rslave mount_point
```

用法实例请参见例 7.5“创建从挂载点”。

示例 7.5：创建从挂载点

这个例子显示了如何获得/media 目录的内容以及出现在/mnt 中，但/mnt 目录中没有任何挂载被映射到/media 目录中。要做到这一点，以 root 的身份，首先将/media 目录标记为“共享”：

```
~]# mount --bind /media /media
~]# mount --make-shared /media
```

然后使用以下命令在/mnt 中创建副本，将其标记为“slave”

```
~]# mount --bind /media /mnt
~]# mount --make-slave /mnt
```

现在可以验证/media 中的挂载也出现在/mnt 内。例如，如果 CD-ROM 驱动器中包含非空介质而且/media/cdrom/目录存在，请运行以下命令：

```
~]# mount /dev/cdrom /media/cdrom
~]# ls /media/cdrom
EFI GPL isolinux LiveOS
~]# ls /mnt/cdrom
EFI GPL isolinux LiveOS
```

同时，验证挂载在/mnt 目录中的文件系统并没有在 media 中得以体现。例如，如果一个使用/dev/sdc1 设备的非空 USB 闪存驱动器被插入而且/mnt/flashdisk/目录存在，请键入以下命令：

```
~]# mount /dev/sdc1 /mnt/flashdisk
~]# ls /media/flashdisk
~]# ls /mnt/flashdisk
en-US publican.cfg
```

私有挂载

私有挂载是挂载的默认类型，不同于共享挂载或从挂载，它不接收或转发任何传播活动。要想清楚地标记一个挂载点的类型为私有挂载，在 shelll 提示下键入以下内容：

```
mount --make-private mount_point
```

另外，要改变选定的挂载点以及其下的所有挂载点的挂载类型，请输入：

```
mount --make-rprivate mount_point
```

用法实例请参见例 7.6“创建私有挂载点”。

示例 7.6：创建私有挂载点

在考虑到例 7.4 方案的情况下，假设先前已经使用 root 身份通过执行下列命令创建了一个共享挂载点：

```
~]# mount --bind /media /media
~]# mount --make-shared /media
~]# mount --bind /media /mnt
```

要将/mnt 目录标记为“私有”，键入以下命令：

```
~]# mount --make-private /mnt
```

现在可以验证/media 中没有挂载出现在/mnt 内。例如，如果 CD-ROM 驱动器中包含非空介质而且/media/cdrom/目录存在，请运行以下命令：

```
-]# mount /dev/cdrom /media/cdrom
-]# ls /media/cdrom
EFI GPL isolinux LiveOS
-]# ls /mnt/cdrom
-]#
```

同时，验证挂载在/mnt 目录中的文件系统并没有在 media 中得以体现。例如，如果一个使用/dev/sdc1 设备的非空 USB 闪存驱动器被插入而且/mnt/flashdisk/目录存在，请键入以下命令：

```
-]# mount /dev/sdc1 /mnt/flashdisk
-]# ls /media/flashdisk
-]# ls /mnt/flashdisk
en-US publican.cfg
```

不可绑定的挂载

为了防止一个给定的挂载点被进行任何复制，请使用不可绑定的挂载点。要将一个挂载点的类型更改为不可绑定挂载，在 shelll 提示下键入以下内容：

```
mount --make-unbindable mount_point
```

另外，要改变选定的挂载点以及其下的所有挂载点的挂载类型，请输入：

```
mount --make-runbindable mount_point
```

用法实例请参见例 7.7“创建不可绑定挂载点”。

示例 7.7：创建不可绑定挂载点

要防止/media 目录被共享，以 root 的身份在 shell 提示下键入以下内容：

```
~]# mount --bind /media /media
~]# mount --make-unbindable /media
```

这样一来，任何为这个挂载点创建副本的后续尝试都将因错误而失败：

```
~]# mount --bind /media /mnt
mount: wrong fs type, bad option, bad superblock on /media,
missing codepage or helper program, or other error
In some cases useful info is found in syslog - try
dmesg | tail or so
```

7.2.4 移动 Mount 点

要改变文件系统所挂载的目录，使用下面的命令：

```
mount --move old_directory new_directory
```

用法实例请参见例 7.8。

示例 7.8：移动一个现存的 NFS 挂载点

NFS 存储包含用户目录，并且已经挂载在/mnt/userdirs/中。以 root 身份，通过使用下面的命令将这个挂载点移动到/home：

```
~]# mount --move /mnt/userdirs /home
```

要验证挂载点已经被移动，请列出两个目录的内容：

```
-]# ls /mnt/userdirs
-]# ls /home
jill joe
```

7.3 卸载文件系统

要分离以前挂载的文件系统，可以使用以下 umount 命令变种的其中之一：

```
umount directory
umount device
```

请注意，除非以 root 身份登录执行此命令，您必须有正确的权限来卸载文件系统（见“7.2.2 指定挂载选项”）。用法实例请参见例 7.9 卸载 CD。

 提示：确定该文件系统不在使用中

当一个文件系统正在使用中（例如，当一个进程正在读取该文件系统上的一个文件时，或当它正在被内核使用时），运行 umount 命令将因错误而失败。要确定哪些过程正在访问该文件系统，请在下表中使用 fuser 命令：

```
fuser -m directory
```

例如，列出正在访问挂载到 media/cdrom/目录的文件系统的进程列表，请键入：

```
]$ fuser -m /media/cdrom
/media/cdrom:   1793     2013     2022     2435     10532c     10672c
```

示例 7.9：卸载 CD

要卸载一个先前挂载到/media/cdrom/目录的 CD，请在 shell 提示下键入以下内容：

```
]$ umount /media/cdrom
```

7.4 文档

以下资源提供了一个关于深入这个主题的文档。

7.4.1 手册页文档

man 8 mount —规定其用法的完整文档的 mount 命令手册页。

man 8 umount —提供了其用法完整文档的 umount 命令手册页。

man 8 findmnt —提供了其用法完整文档的 findmnt 命令手册页。

man 5 fstab —手册页为/etc/fstab 文件格式提供了一个详细描述。

7.4.2 有用的网站

Shared subtrees —一篇 LWN 文章，包含共享子树的概念。

sharedsubtree.txt —一共享子树的补丁所附带的扩展文档。

8. Ext3 文件系统

ext3 文件系统本质上是 ext2 文件系统的一个增强版。这些改进提供了以下优点：

1） 可用性

意外电源故障或系统崩溃（也称为不洁系统关机）之后，每个机器上挂载的 ext2 文件系统必须接受检查以保持与 e2fsck 程序的一致性。这是一个耗时的过程，可能会大大延迟系统启动时间，特别是含有大量文件的大卷。在这段时间内，卷上的任何数据都不可访问。

ext3 文件系统提供的日志意味着这种文件系统检查不再是不洁系统关机后所必要的步骤。在某些罕见硬件故障的情况下，比如硬盘驱动器故障，唯一一次使用 ext3 的一致性检查才会发生。不洁系统关机后恢复 ext3 文件系统所需的时间并不依赖于文件系统的大小或文件数量，而是取决于用来保持一致性的日志的大小。默认的日志大小需要大约一秒钟来进行恢复，这取决于硬件的速度。

 备注：ext3 中唯一支持的日志模式是 data=ordered（默认）。

2） 数据完整性

ext3 文件系统防止在发生不洁系统关机时丢失数据的完整性。ext3 文件系统允许您选择您的数据受保护的类型和等级。默认情况下，ext3 卷配置保持文件系统状态方面高度的数据一致性。

3） 速度

尽管不止一次地写入一些数据，ext3 在大多数情况下比 ext2 具有更高的吞吐量，因为 ext3 日志优化了硬盘磁头的动作。您可以从三个日志模式中来选择优化速度，但是如果系统可能失败，这样做意味着需要在数据完整性方面做出权衡。

4） 易于转换

很容易从 ext2 迁移到 ext3，并且从一个强大的日志文件系统中收益，而无需重新格式化。关于如何执行此任务的详细信息，请参见“8.2 转换为 Ext3 文件系统”。

ext3 的中标麒麟可信操作系统 V6.0V6.0 具有以下更新特性：

1） 默认索引节点大小的改变

为适应扩展属性的更高效存储，磁盘上索引节点的默认大小已经被增加，例如 ACL 或 SELinux 属性。随着这种变化，一个给定大小的文件系统上创建的索引节点的默认数量已经减少。索引节点的大小可以在 mke2fs -I 选项中来选定或在/etc/mke2fs.conf 目录中进行指定，以进行系统范围的 mke2fs 缺省设置。

 备注：如果您在升级到中标麒麟可信操作系统 V6.0V6.0 时想让任何 ext3 文件系统保持完好，您不必重做该文件系统。

2） 新挂载点：data_err

一个新的挂载点已经被添加 data_err=abort。如果一个 data=ordered 方式的文件数据（而不是元数据）缓冲发生错误，此选项指示 ext3 中止该日志。此选项默认是关闭的（即设置为 data_err=ignore）。

3） 更高效的存储使用

当创建一个文件系统（即 mkfs）时，mke2fs 将尝试“抛弃”或“修剪”文件系统元数据所不使用的块。这有助于优化固态硬盘或精简存储。为了制止这种行为，请使用 mke2fs -K 选项。

以下各节会给您详细解释用于创建和调整 ext3 分区的各个步骤。ext2 分区，跳过以下分区和格式化部分，并直接进入“8.2 转换为 Ext3 文件系统”。

8.1 创建 Ext3 文件系统

挂载后，创建一个新的 ext3 文件系统有时是很必要的。例如，如果添加了一个新的磁盘驱动器系统，您可能需要对硬盘进行分区，并使用 ext3 文件系统。

创建一个 ext3 文件系统的步骤如下：

1） 使用 mkfs 来格式化带有 ext3 文件系统的分区。

2） 使用 e2label 来标记该文件系统。

8.2 转换为一个 Ext3 文件系统

tune2fs 让您将 ext2 文件系统转换为 ext3。

 备注：在使用 tune2fs 之前和之后始终用 e2fsck 实用程序来检查您的文件系统。中标麒麟可信操作系统 V6.0 的默认安装对所有文件系统都使用 ext4。在试图转换之前，备份您的所有文件系统以防发生任何错误。此外，我们建议您应该尽可能创建一个新的 ext3 文件系统并向其迁移您的数据，而不是从 ext2 转换为 ext3。

要将 ext2 文件系统转换为 ext3，以 root 身份登录，并在终端键入以下命令：

```
tune2fs -j block_device
```

其中 block_device 包含要转换的 ext2 文件系统。

一个有效的块设备可能是两种类型的记录其中之一：

1） 映射设备 - 卷组中的逻辑卷，例如，/dev/mapper/VolGroup00-LogVol02。

2） 静态设备 - 传统的存储卷，例如，/dev/sdbX，其中 sdb 是一个存储设备的名称，X 是分区号。

使用 df 命令显示已挂载的文件系统。

8.3 恢复到一个 Ext2 文件系统

为方便起见，本节中的示例命令对于块设备使用以下值：
/dev/mapper/VolGroup00-LogVol02

如果因为某种原因您想将一个分区从 ext3 恢复到 ext2，您必须先通过以 root 身份登录并输入以下命令来卸载该分区，

```
umount /dev/mapper/VolGroup00-LogVol02
```

接下来，通过以 root 用户身份键入以下命令将文件系统类型改为 ext2：

```
tune2fs -O ^has_journal /dev/mapper/VolGroup00-LogVol02
```

通过以 root 用户身份键入以下命令进行分区的错误检查：

```
e2fsck -y /dev/mapper/VolGroup00-LogVol02
```

然后再通过键入以下命令将分区挂载为 ext2 文件系统：

```
mount -t ext2 /dev/mapper/VolGroup00-LogVol02 /mount/point
```

在上面的命令中，用分区挂载点来替换/mount/point。

 备注：如果.journal 日志文件存在于分区的 root 层，则删除它。

你现在有一个 ext2 分区。如果你想永久地将分区更改为 ext2，记得要更新/etc/fstab 文件。

9. Ext4 文件系统

ext4 文件系统是 ext3 文件系统的可扩展延伸，ext3 是中标麒麟可信操作系统 V6.0V6.0 的默认文件系统。现在 Ext4 是中标麒麟可信操作系统 V6.0V6.0 的默认文件系统，并且可以支持 8TB 的文件和文件系统。它还支持无限多个子目录（ext3 文件系统只支持 32000 个子目录）。

主要特点：

ext4 使用区间（相对于传统的用于 ext2 和 ext3 的块映射方案），从而提高使用大文件时的性能并减少大文件的元数据开销。此外，ext4 也对未分配的块组和相应的 inode（索引节点）表部分进行了标记，这使得它们能够在文件系统检查过程中被跳过。随着文件系统大小的增长，这使得较快的文件系统检查变得更加有利。

分配特点

ext4 文件系统具有以下特点的分配方案：

1） 持久性预分配

2） 延迟分配

3） 多块分配

4） Stripe-aware 分配

由于延迟分配和其他性能优化，ext4 写入文件到磁盘的行为不同于 ext3。在 ext4 中，程序写入到文件系统并不能保证写在磁盘上的，除非该程序随后发出 fsync()调用。

默认情况下，ext3 几乎立即自动强迫新创建的文件进入到磁盘，即使在没有发出 fsync()调用的情况下。此行为隐藏了没有使用 fsync()的程序中的错误，以确保数据写入到磁盘上。另一方面，ext4 文件系统经常等待几秒钟的时间写出磁

盘上的变化，允许它结合并重新排列写入以获得比 EXT3 更好的磁盘性能。

警告：不同于 ext3，ext4 文件系统在提交事务时并不强制把数据写到磁盘。因此，它需要更长的时间将缓冲写入刷新到磁盘。如同其他的文件系统，使用数据完整性调用如 fsync()，以确保数据写入到永久存储中。

其它 Ext4 特点：

ext4 文件系统也支持以下内容：

1） 扩展属性（xattr），它可以使该系统与每个文件的一些额外名称/值对相关联。

2） 配额日志，这就避免了系统崩溃之后冗长的配额一致性检查的需要。

3） 亚秒时间戳

9.1 创建一个 Ext4 文件系统

要创建一个 ext4 文件系统，使用 mkfs.ext4 命令。在一般情况下，默认选项对于大多数使用场景是最佳的，如：

```
mkfs.ext4 /dev/device
```

下面是此命令的示例输出，它显示了生成的文件系统的几何特征和特点：

```
mke2fs 1.41.9 (22-Aug-2009)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
1954064 inodes, 7813614 blocks
390680 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=4294967296
239 block groups
32768 blocks per group, 32768 fragments per group
8176 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
4096000
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

对于条状块设备（如 RAID5 阵列），可以在文件系统创建时指定条带形状。

使用适当的条带形状，大大提高了 ext4 文件系统的性能。

当在 lvm 或 md 卷上创建文件系统时，mkfs.ext4 选择一个最佳的几何形状。将某些几何信息导出到操作系统的硬件 RAID 上，这一点也可能是真实的。

要指定条状几何图形，请使用带有如下子选项的 mkfs.ext4（即扩展文件系统选项）的-E 选项：

stride=value

指定 RAID 大块量度。

stripe-width=value

指定一个 RAID 设备中的数据磁盘数量，或条带中条带化单位的数量。

对于两个子选项，value 一定要在文件系统的块单元中被指定。例如，要在 4K 块文件系统中创建一个 64K 条带化单位（即 16×4096）的文件系统，请使用以下命令：

```
mkfs.ext4 -E stride=16,stripe-width=64 /dev/device
```

关于创建文件系统的更多信息，请参阅 man mkfs.ext4。

 备注：可以使用 tune2fs 开启 ext3 文件系统上的某些 ext4 功能，以及使用 ext4 驱动程序来挂载一个 ext3 文件系统。然而，这些行动不为中标麒麟可信操作系统 V6.0 所支持，因为他们未经完全测试。正因为如此，对于因此转换或挂载的 ext3 文件系统，我们不能保证一致的性能和可预测的行为。

9.2 挂载一个 Ext4 文件系统

ext4 文件系统可以在没有额外选项的情况下挂载。例如：

```
mount /dev/device /mount/point
```

ext4 文件系统还支持影响行为的多个挂载选项。例如，acl 参数启用访问控制列表，而 user_xattr 参数开启用户扩展属性。要启用这两个选项，使用它们各自的参数-o，如：

```
mount -o acl,user_xattr /dev/device /mount/point
```

tune2fs 实用工具还允许管理员设置文件系统超级块中的默认挂载选项。关于此方面的更多信息，请参阅 man tune2fs。

写屏障

默认情况下，ext4 使用写屏障以确保文件系统的完整性，即使启用写入缓存的设备断电。对于没有写入缓存的设备或电池供电的写缓存设备，请用 nobarrier 选项来禁用写障碍：

```
mount -o nobarrier /dev/device /mount/point
```

关于写屏障的更多信息，请参考第十九章写屏障。

9.3 调整 Ext4 文件系统

扩展 ext4 文件系统之前，确保底层块设备大小适当以保存后来的文件系统。对于受影响的块设备，请使用适当的方法来调整大小。

在使用 resize2fs 命令进行挂载时，ext4 文件系统可能会被扩展，如：resize2fs /mount/point size

resize2fs 命令也可以减少已卸载的 ext4 文件系统的大小，例如：resize2fs /dev/device size

当调整 ext4 文件系统时，resize2fs 实用程序以文件系统块为单位读取文件的大小，除非使用后缀来表明特定的单位。以下后缀表明特定单位：

1） s —512kb 扇区

2） K —千字节

3） M —兆字节

4） G —千兆字节

关于调整 ext4 文件系统的更多信息，请参阅 man resize2fs。

9.4 其他 Ext4 文件系统实用程序

中标麒麟可信操作系统 V6.0 还具有管理 ext4 文件系统的其他实用程序的特点：

1） e2fsck

用于修复 ext4 文件系统。此工具对 ext4 文件系统进行了比 ext3 更为有效的检查和修复，这得益于 ext4 磁盘结构的更新。

2） e2label

更改 ext4 文件系统上的标签。此工具还可以在 ext2 和 ext3 文件系统上起作用。

3） quota

ext4 文件系统用户和组关于磁盘空间（块）和文件（索引节点）用法的报告和控制。关于使用 quota 的更多信息，请参考 man quota 和 17.1 节“配置磁盘定额”。

正如早在 9.2 节“挂载 Ext4 文件系统”，tune2fs 实用程序也可以为 ext2，

ext3 和 ext4 文件系统调整配置文件系统参数。此外，下面的工具在调试和分析 ext4 文件系统时也有用：

4） debugfs

调试 ext2、ext3 或 ext4 文件系统。

5） e2image

将关键的 ext2，ext3 或 ext4 文件系统元数据保存到文件中。

有关这些实用程序的更多信息，请参阅各自的手册页。

10. GFS2

中标麒麟可信操作系统 V6.0GFS2 文件系统是一个直接与 Linux 内核文件系统接口（VFS 层）相互联结的本地文件系统接口。当作为一个集群文件系统执行时，GFS2 采用分布式元数据和多种日志。

GFS2 基于 64 位架构，它在理论上可以容纳 8 EB 文件系统。然而，GFS2 文件系统当前支持的最大尺寸为 100 TB。如果您的系统需要超过 100 TB 的 GFS2 文件系统，请与您的中标麒麟服务代表取得联系。

确定文件系统的大小时，您应该考虑您的恢复需求。在一个非常大的文件系统上运行 fsck 命令可能需要很长的时间而且消耗大量的内存。此外，在磁盘或磁盘子系统故障的情况下，恢复时间是受限于备份介质的速度。

当在中标麒麟可信操作系统 V6.0 集群套件中进行配置时，中标麒麟可信操作系统 V6.0GFS2 节点可以用中标麒麟可信操作系统 V6.0 集群套件配置和管理工具进行配置和管理。然后中标麒麟可信操作系统 V6.0GFS2 提供中标麒麟可信操作系统 V6.0 集群 GFS2 节点之间的数据共享，采用整个 GFS2 节点的文件系统名称空间的单一且一致的视图。这使得不同节点上的进程共享 GFS2 文件，就像同一节点上的进程共享本地文件系统上的文件一样，没有明显的区别。有关中标麒麟可信操作系统 V6.0 集群套件的信息，请参阅配置和管理中标麒麟集群。

GFS2 文件系统必须创建在一个线性或镜像 LVM 逻辑卷上。中标麒麟可信操作系统 V6.0 集群套件中的 LVM 逻辑卷由 CLVM 进行管理，这是一个集群范围内的 LVM 实施，由 CLVM 守护进程和 clvmd 所启用，运行在中标麒麟可信操作系统 V6.0 集群套件集群中。该守护进程使得应用 LVM2 来管理跨集群逻辑卷成为可能，使集群中的所有节点共享逻辑卷。LVM 卷管理器上的信息，请参阅

逻辑卷管理器管理。

gfs2.ko 内核模块实现 GFS2 文件系统并加载在 GFS2 集群节点上。

 提示：关于集群和非集群存储中 GFS2 文件系统的创建和配置的全面信息，请参阅中标麒麟可信操作系统 V6.0 提供的 GFS2 指南。

11. XFS 文件系统

XFS 是一个高度可扩展的，高性能的文件系统，最初是由 Silicon Graphics 公司设计的，它的创建是为了支持非常大的文件系统（最多 16 个 E 字节），文件（8E 字节）和目录结构（数以千万个条目）。

主要特点：

XFS 支持元数据日志，这有利于系统崩溃的更快恢复。XFS 文件系统在挂载和活动状态下也可以同时进行碎片整理和扩大。此外，中标麒麟可信操作系统 V6.0V6.0 支持备份和恢复 XFS 特定的实用程序。

分配特点

XFS 具有以下分配方案的特点：

1） 基于区间的分配

2） 支持条带化分配政策

3） 延迟分配

4） 空间预分配

延迟分配和其他性能优化以影响 ext4 的同样方式来影响 XFS。也就是说，程序写入到XFS 文件系统是不能保证写在磁盘上的，除非该程序随后发出 fsync() 调用。

关于文件系统上延迟分配含义的更多信息，请参阅章节 0Ext4 文件系统中的分配功能。确保写入到磁盘的解决方法也适用于 XFS。

其它 XFS 特点

ext4 文件系统也支持以下内容：

1） 扩展属性（xattr），它可以使该系统与每个文件的一些额外名称/值对相关联。

2） 定额日志，这就避免了系统崩溃之后进行冗长定额一致性检查的必要。

3） 项目/目录定额，允许定额限制超过一个目录树。

4） 亚秒时间戳

11.1 创建一个 XFS 文件系统

要创建一个 ext4 文件系统，请使用 mkfs.xfs /dev/device 命令。在一般情况

下，默认选项对于常见使用环境是最佳的，如：

当使用包含现有文件系统块设备上的 mkfs.xfs 时，使用-f 选项强制覆盖该文件系统。

下面是一个 mkfs.xfs 命令的示例输出：

```
meta-data=/dev/device  isize=256    agcount=4,  agsize=3277258  blks
=  sectsz=512     attr=2
data    =  bsize=4096      blocks=13109032,  imaxpct=25
=  sunit=0  swidth=0  blks
naming    =version 2    bsize=4096      ascii-ci=0
log =internal  log     bsize=4096      blocks=6400,  version=2
=  sectsz=512   sunit=0  blks,  lazy-count=1
realtime   =noneextsz=4096  blocks=0,  rtextents=0
```

备注：XFS 文件系统创建后，它的大小不能减小。然而，通过使用 xfs_growfs 命令它仍然可以被放大（参见 “11.4 增加 XFS 文件系统的大小”）。

对于条状块设备（如 RAID5 阵列），可以在文件系统创建时指定条带状。使用适当的条带状，大大提高了 XFS 文件系统的性能。

当在 lvm 或 md 卷上创建文件系统时，mkfs.ext4 选择一个最佳的几何形状。在某些将几何信息导出到操作系统的硬件 RAID 上，这一点也可能是真实的。

要指定条带状，请使用以下 mkfs.xfs 子选项：

su=value

指定条带状单元或 RAID 数据块大小。该值必须以字节为单位进行指定，带有可选的 k、m 或 g 后缀。

sw=value

指定一个 RAID 设备中的数据磁盘数量，或条带中条带化单位的数量。

下面的例子指定一个包含了 4 个条带化单位的 RAID 设备，该数据块大小为 64K：

```
mkfs.xfs -d su=64k,sw=4 /dev/device
```

关于创建文件系统的更多信息，请参阅 man mkfs.ext4。

11.2 挂载 XFS 文件系统

XFS 文件系统可以在没有额外选项的情况下挂载，例如。

```
mount /dev/device /mount/point
```

XFS 文件系统还支持影响行为的多个挂载选项。

默认情况下，XFS 分配索引节点以反映其在磁盘上的位置。然而，由于一些 32 位用户空间应用程序不兼容大于 2^{32} 的索引节点号，XFS 会分配产生 32 位索引节点编号的磁盘位置中的所有索引节点。这可能会导致非常大的文件系统（即大于 2 TB 的）的性能下降，因为索引节点偏向于块设备的开端，而数据偏向于结尾。

解决此问题，请使用 inode64 挂载选项。此选项配置 XFS 以分配整个文件系统的索引节点和数据，这样可以提高性能：

```
mount -o inode64 /dev/device /mount/point
```

写屏障

默认情况下，XFS 使用写屏障以确保文件系统的完整性，即使当写入缓存启用设备断电时。对于没有写入缓存的设备或电池供电的写缓存设备，请用 nobarrier 选项来禁用写屏障：

```
mount -o nobarrier /dev/device /mount/point
```

关于写屏障的更多信息，请参考章节 19 写屏障。

11.3 XFS 定额管理

XFS 定额子系统管理磁盘对空间（块）和文件（索引节点）使用所施加的限制。XFS 定额控制或报告用户，组或目录/项目层上的这些项的使用方法。此外，请注意虽然用户、组和目录/项目定额被单独启用，组和项目的定额互斥。

在按目录或按项目管理的基础上，XFS 管理与特定项目相关的目录层级的磁盘使用情况。这样做，XFS 认识到项目之间跨组织的“群”界限。这样一来可以提供比用户或组可用的定额管理更宽泛的控制水平。

XFS 定额在挂载时启用，采用特定的挂载选项。每个挂载选项也可以被指定为 noenforce，在不强加任何限制的情况下，这将允许报告使用情况。有效的定额挂载选项包括：

1） uquota/uqnoenforce – 用户定额

2） gquota/gqnoenforce - 组定额

3） pquota/pqnoenforce - 项目定额

一旦启用定额，xfs_quota 工具可以用来设置磁盘使用情况的限制和报告。默认情况下，xfs_quota 在基本模式下交互运行。基本模式子命令只是报告使用

情况并提供给所有用户。基本 xfs_quota 子命令包括：

quota username/userID

显示给定的用户名或数字用户 ID 的使用和限制

df

显示块和索引节点的未使用的和已使用的计数。

相比之下，xfs_quota 也有一个专家模式。该模式的子命令允许限制的实际配置，并且只提供给高特权用户。要交互使用专家模式的子命令，运行 xfs_quota -x。专家模式的子命令包括：

report /path

为特定的文件系统报告定额信息。

limit

修改定额限制。

对于基本模式或专家模式的子命令的完整列表，请使用子命令的帮助。

所有子命令也可以使用- c 选项直接从命令行运行，-x 用于运行专家子命令。例如，要为/home（/dev/ blockdevice）显示一个样本配额报告，请使用命令 xfs_quota -cx 'report -h' /home。显示类似于以下界面的输出：

```
User quota on /home (/dev/blockdevice)
Blocks
User     ID  Used     SoftHard    Warn/Grace
----------------------------------------
0    0     0     00  [     ]
103.4G 0     0     00  [     ]
```

要为用户 John（其主目录是/home/john）设置分别为 500 和 700 的软、硬索引节点数限制，请使用下面的命令：

```
xfs_quota -x -c 'limit isoft=500 ihard=700 /home/john'
```

默认情况下，limit 子命令将目标识别为用户。当为一组配置限制时，使用-g 选项（如在前面的例子中）。同样，对项目使用-p 选项。

软、硬块限制也可以使用 bsoft/bhard 而不是 isoft/ihard 来进行配置。例如，设置一个 1000 兆和 1200 兆的软、硬块限制，按/target/path 文件系统上的 accounting 分组，使用下面的命令：

```
xfs_quota -x -c 'limit -g bsoft=1000m bhard=1200m accounting' /target/path
```

 备注：虽然实时块（rtbhard/rtbsoft）在 man xfs_quota 中被描述为设置配额时的有效单元，实时分卷并没有在此版本中启用。因此，rtbhard 和 rtbsoft 选项并不适用。

设置项目限制

在为项目控制目录配置限额之前，首先将它们添加到/etc/projects 中。项目名称可以添加到/etc/projectid 以便将项目 ID 映射到项目名称中。一旦项目被添加到/etc/projects，使用下面的命令初始化项目目录：

```
xfs_quota -c 'project -s projectname'
```

初始化目录项目的配额可以随后进行配置，如：

```
xfs_quota -x -c 'limit -p bsoft=1000m bhard=1200m projectname'
```

通用配额配置工具（如 quota、repquota、edquota）也可用于操纵 XFS 配额。然而，这些工具不能与 XFS 项目配额共同使用。

关于设置 XFS 配额的更多信息，请参阅 man xfs_quota。

11.4 增加 XFS 文件系统的大小

在使用 xfs_growfs 命令进行挂载的同时 XFS 文件系统可能会增长：

```
xfs_growfs /mount/point -D size
```

-D size 选项将文件系统增长到指定的大小（文件系统块所表示的）。如果没有-D size 选项，xfs_growfs 会使文件系统大小增长到设备支持的最大尺寸。

用-D size 选项扩展 xfs 文件系统之前，确保底层块设备大小适当以保存后来的文件系统。对于受影响的块设备，请使用适当的方法来调整大小。

 备注：虽然文件系统在挂载时可以扩展，但它们的大小根本不能缩减。

关于扩展文件系统的更多信息，请参阅 man xfs_growfs。

11.5 修复 XFS 文件系统

要修复一个 xfs 文件系统，请使用 xfs_repair 命令，如下：

xfs_repair /dev/device

xfs_repair 实用程序是高度可扩展的，并且是为修复多索引节点特大文件系统而设计的。请注意，不像其他的 Linux 文件系统，即使当一个 XFS 文件系统遭遇不洁卸载时，xfs_repair 也不会在启动时运行。在不洁卸载的情况下，xfs_repair 在挂载时简单地重放日志，确保一致稳定的文件系统。

 备注：xfs_repair 实用工具不能修复带有脏日志的 XFS 文件系统。要清除日志，挂

载和卸载 XFS 文件系统。如果日志损坏，无法重播，请使用-L 选项（“强制日志归零”）以清除日志，即 xfs_repair -L /dev/device。但是请注意，这可能导致进一步的损坏或数据丢失。

关于创建文件系统的更多信息，请参阅 man mkfs.ext4。

11.6 挂起 XFS 文件系统

若要挂起或恢复写活动到一个文件系统，请使用 xfs_freeze。挂起写活动允许基于硬件的设备快照被用于捕获处于一致状态下的文件系统。

备注：xfs_freeze 实用程序由 xfsprogs 软件包提供，仅在 x86_64 上可用。

要挂起 XFS 文件系统（即冻结），请使用：

```
xfs_freeze -f /mount/point
```

要解冻 XFS 文件系统，请使用：

```
xfs_freeze -u /mount/point
```

当拍摄 LVM 快照时，没有必要使用 xfs_freeze 先挂起文件系统。相反，LVM 管理工具会在快照之前自动挂起 XFS 文件系统。

备注：您还可以使用 xfs_freeze 实用程序来冻结/解冻 ext3，ext4 的，GFS2 的，XFS，和 BTRFS 文件系统。这样做的句法也是一样的。

关于冻结和解冻文件系统的更多信息，请参阅 man xfs_freeze。

11.7 XFS 文件系统的备份和恢复

XFS 文件系统的备份和恢复涉及两个实用程序： Xfsdump 和 xfsrestore。

备份或转储 XFS 文件系统，使用 xfsdump 实用工具。中标麒麟可信操作系统 V6.0V6.0 支持到磁带驱动器或常规文件图像的备份，并且还允许多个转储写入到同一个磁带。xfsdump 实用程序还允许跨越多个磁带的转储，虽然只有一个转储可以被写入到正规文件中。此外，xfsdump 支持增量备份，并且可以使用大小，子树或索引节点从备份中排除这些文件，以便将让们过滤。

为了支持增量备份，xfsdump 使用转储级别来确定一个与特定转储相对的基本转储。-l 选项指定了一个转储级别(0-9)。要执行完全备份，执行一个 0 级转储文件系统（即/path/to/filesystem），如：

```
xfsdump -l 0 -f /dev/device /path/to/filesystem
```

备注：-f 选项指定了备份目的地址。例如，/dev/st0 目的地址通常被用做磁带驱动器。xfsdump 目的地址可以是磁带驱动器，正规文件或远程磁带设备。

与此相反，增量备份只会转储上次 0 级转储之后改变的文件。1 级转储是完

全转储后的第一个增量转储，下一个增量转储是 2 级，依此类推，最大的转储是 9 级。因此，执行 1 级转储到磁带驱动器：

```
xfsdump -l 1 -f /dev/st0 /path/to/filesystem
```

相反，xfsrestore 实用程序从 xfsdump 产生的转储中恢复文件系统。xfsrestore 实用程序有两种模式：一个默认的简单模式，和一个累积模式。具体转储用会话 ID 或会话标签进行标识。因此，恢复转储要求其相应的会话 ID 或标签。要显示所有转储（完整的和增量的）的会话 ID 和标签，请使用- I 选项，如：

```
xfsrestore -I
```

将显示类似于以下界面的输出：

```
file system 0:
fs id: 45e9af35-efd2-4244-87bc-4762e476cbab
session 0:
mount point: bear-05:/mnt/test
device: bear-05:/dev/sdb2
time: Fri Feb 26 16:55:21 2010
session label: "my_dump_session_label"
session id: b74a3586-e52e-4a4a-8775-c3334fa8ea2c
level: 0
resumed: NO
subtree: NO
streams: 1
stream 0:
pathname: /mnt/test2/backup
start: ino 0 offset 0
end: ino 1 offset 0
interrupted: NO
media files: 1
media file 0:
mfile index: 0
mfile type: data
mfile size: 21016
mfile start: ino 0 offset 0
mfile end: ino 1 offset 0
media label: "my_dump_media_label"
media id: 4a518062-2a8f-4f17-81fd-bb1eb2e3cb4f
xfsrestore: Restore Status: SUCCESS
```

XFS 恢复简单模式

简单模式，允许用户从 0 级转储中恢复整个文件系统。在确定一个 0 级转储的会话 ID（即 session-ID）后，使用以下命令将它完全恢复到/path/to/destination：

```
xfsrestore -f /dev/st0 -S session-ID /path/to/destination
```

 备注：- f 选项指定转储的位置，而- S 或- L 选项指定要恢复哪些具体的转储。- S 选项是用来指定一个会话 ID，而- L 选项用来指定一个会话标签。- I 选项显示每个转储的会话标签和会话 ID。

XFS 回复简单模式

xfsrestore 累计模式允许从一个特定的增量备份中恢复文件系统，例如 1 级到 9 级。若要从增量备份中恢复文件系统，只需添加- r 选项，如：

```
xfsrestore -f /dev/st0 -S session-ID -r /path/to/destination
```

交互式操作

xfsrestore 实用程序还允许从一个要解压，添加或删除的转储中恢复具体文件。要以交互方式使用 xfsrestore ，请使用- i 选项，如：

```
xfsrestore -f /dev/st0 -i
```

互动对话将在 xfsrestore 完成读取指定装置后开始。此对话中可用命令包括 cd、ls、add、delete 和 extract，完整的命令列表，使用帮助。

关于转储和恢复文件系统的更多信息，请参阅 man xfsdump 和 man xfsrestore。

11.8 其它 XFS 文件系统实用程序

中标麒麟可信操作系统 V6.0 还具有其他管理 XFS 文件系统的实用程序的特点：

1） xfs_fsr

用于挂载的 XFS 文件系统进行碎片整理。当不带参数调用时，**xfs_fsr** 重新整理所有已挂载的 XFS 文件系统中所有正规文件。此实用程序还允许用户在指定的时间暂停碎片整理，并从它所中止的地方恢复。

此外，xfs_fsr 还允许仅针对一个文件的碎片整理，如 xfs_fsr /path/ to/file。我们建议不要对整个文件系统进行定期碎片整理，因为这通常是不合理的。

2） xfs_bmap

打印一个 XFS 文件系统中的文件所使用的磁盘块的映射。这个映射列出了指定文件所使用的每一个区间，以及没有相应的块（即孔）的文件中的区域。

3） xfs_info

打印 XFS 文件系统的信息。

4） xfs_admin

更改 XFS 文件系统的参数。xfs_admin 实用工具只能修改、卸载设备/文件系统的参数。

5） xfs_copy

复制整个 XFS 文件系统的内容到一个或多个并行的目标中。

此外，下面的实用工具在调试和分析 XFS 文件系统时也有用：

6） xfs_metadump

复制 XFS 文件系统元数据到一个文件。xfs_metadump 实用程序应该只用来复制已卸载的，只读的，或冻结的/暂停的文件系统，否则，生成转储可能会被损坏或不一致。

7） xfs_mdrestore

恢复 XFS 元转储映像（使用 xfs_metadump 来产生）为一个文件系统映像。

8） xfs_db

调试 XFS 文件系统

有关这些实用程序的更多信息，请参阅各自的手册页。

12. 网络文件系统 (NFS)

网络文件系统（NFS）允许远程主机通过网络挂载文件系统，并与这些文件系统进行交互，就好像它们是本地挂载的一样。这使得系统管理员可以将资源整合到网络上的集中服务器上。

本章重点介绍 NFS 的基本概念和补充信息。

12.1 NFS 是如何工作的？

目前，有三个版本的 NFS。NFS 版本 2（NFSv2）发行的时间比较长并得到了广泛支持。NFS 版本 3（NFSv3）支持安全异步写入，拥有比 NFSv2 更强大的错误处理功能，它也支持 64 位的文件大小和偏移，允许客户端访问超过 2GB 的文件数据。

NFS 版本 4（NFSv4）通过防火墙在互联网上起作用，不再需要 rpcbind 服务，它支持 ACL 并利用状态操作。中标麒麟可信操作系统 V6.0 支持 NFSv2，

NFSv3 和 NFSv4 客户。通过 NFS 文件系统挂载时，如果服务器支持，中标麒麟可信操作系统 V6.0 在默认情况下使用 NFSv4。

NFS 的所有版本都可以使用在 IP 网络上运行的传输控制协议（TCP），NFSv4 有此要求。NFSv2 和 NFSv3 可以使用运行在 IP 网络上的用户数据报协议（UDP）提供客户端和服务器之间的无状态网络连接。

当使用带有 UDP 的 NFSv2 或 NFSv3，无状态的 UDP 连接（正常情况下）的协议开销小于 TCP，这可以转化为非常洁净而且不拥挤的网络上的更好表现。但是，由于 UDP 是无状态的，如果服务器意外停机，UDP 客户仍发出大量服务器请求致使网络继续饱和。此外，当一个帧丢失 UDP，整个 RPC 请求必须重传，而在 TCP 情况下，仅丢失帧需要重新发送。由于这些原因，TCP 是连接 NFS 服务器时的首选协议。

挂载和锁定协议已纳入 NFSv4 协议中。服务器也在众所周知的 TCP 端口 2049 上监听。因此，NFSv4 用户不需要与 rpcbind、lockd 和 rpc.statd 守护进程交互。NFS 服务器仍然需要 rpc.mountd 守护进程，但不涉及任何线上操作。

备注：TCP 是中标麒麟可信操作系统 V6.0 下 NFS 版本 2 和 3 的默认传输协议。UDP 可用于满足所需要的兼容目的，但不推荐广泛使用。NFSv4 需要 TCP。

所有的 RPC/NFS 守护进程都有一个- p 命令行选项，可以设置端口，使防火墙的配置更容易。

TCP 包装授予客户端访问权后，NFS 服务器参考/etc/exports 配置文件，以确定是否允许客户端访问任何导出的文件系统。一旦查实，所有文件和目录操作都将提供给用户。

重要：为了让 NFS 在启用了防火墙的中标麒麟可信操作系统 V6.0 默认安装状态下工作，用默认的 TCP 端口 2049 来配置 IPTables。没有适当的 IPTables 配置的情况下，NFS 将无法正常工作。

NFS 初始化脚本和 rpc. nfsd 过程现在允许绑定到在系统启动时指定的任何端口。不过，如果端口是不可用的，或者与其他守护冲突，这样很容易出错。

12.1.1 所需的服务

中标麒麟可信操作系统 V6.0 使用内核级的支持组合和守护进程来提供 NFS 文件共享。所有的 NFS 版本都依靠客户端和服务器之间的远程过程调用（RPC）。在中标麒麟可信操作系统 V6.0V6.0 下 RPC 服务被 rpcbind 服务所控制。为了共

享或挂载 NFS 文件系统，下列服务共同发挥作用，取决于实施的是哪个版本的 NFS：

 备注：portmap 服务被用来将 RPC 程序号映射到中标麒麟可信操作系统 V6.0 早期版本的 IP 地址端口号组合中。现在，这项服务被中标麒麟可信操作系统 V6.0V6.0 中的 rpcbind 所取代用来启用 IPv6 支持。有关此变化的更多信息，请参阅以下各链接：

TI-RPC /rpcbind support: http://nfsv4.bullopensource.org/doc/tirpc_rpcbind.php

NFS 中 IPv6 支持: http://nfsv4.bullopensource.org/doc/nfs_ipv6.php

nfs

service nfs start 启动 NFS 服务器和适当的 RPC 进程来满足共享 NFS 文件系统的服务请求。

nfslock

service nfslock start 激活一个强制性的服务，启动相应的 RPC 进程以允许 NFS 客户端锁定服务器上的文件。

rpcbind

rpcbind 接受来自本地 RPC 服务的端口保留。然后这些端口变为可用（或被告知），所以相应的远程 RPC 服务可以访问它们。rpcbind 响应 RPC 服务的请求，并设置到所请求的 RPC 服务的连接。这种情况不用在 NFSv4 环境下。

以下 RPC 进程促进了 NFS 服务：

rpc.mountd

这个过程接收来自 NFS 客户端的挂载请求，并验证所请求的文件系统目前正在导出。这个过程被 nfs 服务自动启动，不需要用户的配置。

rpc.nfsd

rpc. nfsd 提供服务器通告要定义的详细的 NFS 版本和协议。它与 Linux 内核协同工作以满足 NFS 客户端的动态需求，如在每次 NFS 客户端连接时提供服务器线程。这一过程与 nfs 服务对应。

lockd

lockd 是一个运行在客户端和服务器上的内核线程。它实现了网络锁管理器（NLM）协议，它允许 NFSv2 和 NFSv3 客户锁定服务器上的文件。每当运行 NFS 服务器和挂载 NFS 文件系统时，它就自动启动。

rpc.statd

这个过程执行了网络状态监视器（NSM）RPC 协议，该协议在 NFS 服务器

不正常移入情况下重新启动时通知 NFS 客户端。rpc.statd 是由 nfslock 服务自动启动，并且不需要用户配置。这种情况不用在 NFSv4 环境下。

rpc.rquotad

这个过程为远程用户提供用户配额信息。rpc.rquotad 是由 NFS 服务自动启动并且不需要用户配置。

rpc.idmapd

rpc.idmapd 提供 NFSv4 客户端和服务器向上调用，进行线上 NFSv4 名称（这是 user@domain 形式的字符串）和本地 UID 和 GID 之间的映射。idmapd 与 NFSv4 一起执行功能，/etc/idmapd.conf 必须进行配置。这项服务需要在 NFSv4 环境下使用，虽然不是所有主机共享相同的 DNS 域名。

12.2 NFS 客户端配置

mount 命令挂载客户端上的 NFS 共享。其格式如下：

```
mount -t nfs -o options host:/remote/export /local/directory
```

该命令使用如下变量：

选项

一组挂载选项的逗号分隔列表，有效的 NFS 挂载选项的详细信息，请参阅“0 常用 NFS 挂载选项”。

服务器

导出您想挂载文件系统的服务器的主机名，IP 地址，或完全合格的域名。

/remote/export

从服务器导出的文件系统/目录，即您想挂载的目录

/local/directory

客户端上/remote/export 应挂载的位置

中标麒麟可信操作系统 V6.0V6.0 使用带有 mount -t nfs 命令的 NFSv4。如果服务器不支持 NFSv4 客户端将自动下降到服务器所支持的版本。如果您使用 nfsvers/vers 选项通过一个服务器不支持的特定版本，挂载将失败。

有关详细信息，请参考 man mount。

如果手动挂载一个 NFS 共享，重新启动后共享将不会自动挂载。中标麒麟可信操作系统 V6.0 提供了两种开机时自动挂载远程文件系统的方法： /etc/fstab 文件和 autofs 服务。更多信息，参见“12.2.1 使用 /etc/fstab 挂载 NFS 文件系统”

和“12.3 autofs”。

12.2.1 使用/etc/fstab 来挂载 NFS 文件系统

从另一台机器挂载 NFS 共享的另一种方法是添加一行命令到/etc/fstab 文件中。该行命令必须声明 NFS 服务器的主机名、服务器上正要导出的目录、以及挂载 NFS 共享的本地计算机上的目录。您必须以 root 身份修改 /etc/fstab 文件。

/etc/fstab 中命令行的通用语法如下：

```
server:/usr/local/pub /pub nfs rsize=8192,wsize=8192,timeo=14,intr
```

在执行此命令之前，挂载点/pub 必须存在于客户机上。将这一行添加到客户端系统上的 /etc/fstab 之后，使用命令 mount /pub 并从服务器中挂载挂载点 /pub 。

/etc/fstab 文件在开机时为 netfs 中所引用，所以引用 NFS 共享的命令行与开机过程中手动键入 mount 命令有同样的效果。

挂载 NFS 导出的一个有效/etc/fstab 项应该包含以下信息：

```
server:/remote/export /local/directory nfs options 0 0
```

变量服务器， /remote/export、/local/directory 以及 options 与手动挂载 NFS 共享时使用的相同。每个变量的定义，请参见“12.2 NFS 客户端配置”

 备注：在读取 /etc/fstab 之前，挂载点/local/directory 必须存在于客户端上。否则，挂载将会失败。

关于/etc/fstab 的更多信息，请参阅 man fstab。

12.3 自动挂载（autofs）

使用/ etc / fstab 的一个缺点是，不管用户访问 NFS 挂载的文件系统的频率如何小，该系统必须耗费资源来保持挂载的文件系统就位。一个或两个挂载并不是问题，但是当系统一次性维护挂载的多个系统，系统的整体性能可能会受到影响。/etc/fstab 中的另一种方法是使用基于内核的挂载工具。自动挂载程序由两部分组成：

1） 一个执行文件系统的内核模块

2） 一个执行所有的其他功能的用户空间守护进程

自动挂载实用程序可以自动地挂载和卸载 NFS 文件系统（按需挂载），因此节省了系统资源。它也可用于挂载其他文件系统，包括 AFS，SMBFS，CIFS 和本地文件系统等。

提示：自动挂载 NFS 共享

nfs-utils 包现在既是“NFS 文件服务器”的一部分也是“网络文件系统客户端” 组的一部分。因此，它不再由基本组默认挂载。在尝试挂载 NFS 共享之前确保 nfsutils 已经安装在系统上。需要注意的是 autofs 也是“网络文件系统客户端”组的一部分。

autofs 使用/etc/auto.master（主映射）作为其默认的主配置文件。这可以改为配合名称服务开关（NSS）机制来使用另一个使用 autofs 配置（在/etc/sysconfig/autofs）的支持的网络源代码和名称。autofs 版本 4 守护进程的一个实例是为主映射中配置的每个挂载点而运行的，所以它可以为任何给定的挂载点从命令行手动运行。使用 autofs 第 5 版是不可能的，因为它使用一个单独的守护进程来管理所有配置的挂载点，正因为如此，所有自动挂载必须配置在主映射中。这是与其他行业标准自动挂载程序的一般要求相符合的。挂载点，主机名，导出的目录和选项都可以在一组文件（或其他支持的网络资源）中进行指定，而不是对每台主机进行手动配置。

12.3.1 autofs 第五版的改进之处

Autofs 第 5 版在第 4 版的基础上进行了以下功能改进：

1） 直接映射支持

autofs 的直接映射提供了一种机制，可以在文件系统中层次结构的任意点上自动挂载文件系统。直接映射表征为主映射中/ -的一个挂载点。直接映射中的条目包含一个作为键的绝对路径名称（而不是间接映射中使用的相对路径名称）。

2） 支持空闲挂载和卸载

多挂载映射条目描述了单键下的一个挂载点层。一个很好的例子是-hosts 映射，常用于在多挂载映射 “/ net/host” 项下从主机挂载所有输出。当使用“-hosts”映射时，"/net/host"的"/s"将为每一个输出从主机安装 autofs 来触发脚本程序挂载，并且在他们被访问时将其挂载或终止。当用大量导出来访问服务器时，可以大大减少所需的活动挂载数量。

3） 增强的 LDAP 支持

Autofs 第 5 版中的轻量目录访问协议（LDAP）支持已在 autofs 第 4 版的基础上通过几种方法得到了增强。autofs 配置文件（/ etc/sysconfig/autofs）提供了一种机制来指定站点执行的 autofs 模式，从而排除了应用程序本身通过试错法进

行确定的必要。此外，经过身份验证的到 LDAP 服务器的绑定现在得到支持，可以使用常见的 LDAP 服务器执行所支持的大多数机制。这种支持已添加新的配置文件： /etc/autofs_ldap_auth.conf。默认的配置文件是自记录的，并使用 XML 格式。

4） 名称服务开关（nsswitch）配置的正确使用。

名称服务开关配置文件的存在是为了提供一种方法来确定特定的配置数据来源。这种配置是为了让管理员获得使用后端数据库选择的灵活性，同时保持一个统一的软件接口来访问数据。虽然第 4 版挂载在处理 NSS 配置方面的性能正变得越来越好，但它仍然是不完整的。Autofs 的第 5 版是一个完整的实现。

该文件支持语法的更多信息，请参阅 man nsswitch.conf。请注意，并非所有的 NSS 数据库都是有效的映射源并且语法分析器将拒绝无效的 NSS 数据库。有效源文件是 yp、nis、nisplus、ldap 和 hesiod。

5） 每个 autofs 挂载点有多个主映射条目

有一样东西经常使用但还没有提到，是对直接挂载点/-的多主映射条目处理。每个条目的映射键被合并并且作为一个映射来发挥作用。

直接挂载的 connectathon 测试映射如下例所示：

```
/- /tmp/auto_dcthon
/- /tmp/auto_test3_direct
/- /tmp/auto_test4_direct
```

12.3.2 自动挂载（autofs）配置

自动挂载程序的主要配置文件是/etc/auto.master，也称为主映射，它可以按照 12.3.1 节“autofs 第 5 版较第 4 版的改进之处”所描述的进行更改。主映射列出系统上的 autofs 控制的挂接点，而且其相应的配置文件或网络资源被称为自动加载映射。主映射的格式如下：

```
mount-point map-name options
```

这种格式所使用的变量如下：

mount-point

autofs 挂载点 如 /home。

map-name

包含挂载点列表的映射源名称，以及这些挂载点应被挂载在哪个文件系统位置。映射项的句法如下所述。

选项

如果提供，这些都将适用于给定映射中的所有条目，前提是它们本身并不具有指定的选项。这种行为不同于选项在其中累计的第 4 版 autofs。这已被更改以实现混合环境的兼容性。

下面是一个来自/etc/auto.master 文件的示例行（以 cat /etc/ auto.master 显示）：

```
/home /etc/auto.misc
```

映射的一般格式类似于主映射，但是 “选项” 出现在挂载点和位置之间，而不像主映射中出现在条目结尾处：

```
mount-point [options] location
```

这种格式所使用的变量如下：

mount-point

这是指 autofs 挂载点。这可以是间接挂载的单一目录名称或直接挂载的挂载点的完整路径。每一个直接和间接的映射输入键（mount-point 以上）可能后跟一个空格隔开的偏移目录列表（每个子目录名称以“/”开始）使他们成为一个多重挂载条目。

选项

只要提供，这些都是不指定自己的选项的映射条目的挂载选项。

位置

这是指文件系统位置，如本地文件系统路径（对于以"/"开头的映射名称以 sun 映射格式转义字符":"继续)， NFS 文件系统或其他有效的文件系统位置。

以下是从映射文件（即/etc/auto.misc）内容的一个示例：

```
payroll -fstype=nfs personnel:/dev/hda3
sales -fstype=ext3 :/dev/hda4
```

映射文件中的第一栏表示 autofs 挂载点（来自名为 personnel 的服务器的 sales 和 payroll）。第二栏表示 autofs 挂载的选项，而第三栏表示挂载的来源。上述配置之后，autofs 挂载点会变为/home/payroll 和 /home/sales。-fstype=选项经常被忽略，一般情况下正确的操作不需要该选项。

如果目录不存在，自动挂载程序将创建目录。如果挂载开始前目录就已经存在，自动挂载程序在退出时将不会删除它们。通过发出以下两个命令之一，您可以启动或重新启动 automount 守护进程：

```
service autofs start
```

```
service autofs restart
```

使用上面的配置，如果一个进程需要访问 autofs 卸载目录，如/ /home/payroll/2006/July.sxc，automount 守护进程自动挂载该目录。在指定超时情形下，如果目录不能在超时期间被访问，将自动被卸载。

您可以通过发出以下命令来查看 automount 守护进程的状态：

```
service autofs status
```

12.3.3 覆盖或增加网站配置文件

对于客户端系统上的一个特定的挂载点来说，覆盖站点默认值可能是有用的。例如，考虑下列条件：

挂载程序映射存储在 NIS 和/etc/nsswitch.conf 文件中有以下指令：

```
automount:  files nis
```

auto.master 文件中包含以下：

```
+auto.master
```

NIS auto.master map 文件中包含以下：

```
/home auto.home
```

NIS auto.home map 文件中包含以下：

```
beth      fileserver. example. com:/export/home/beth
joe fileserver. example. com:/export/home/joe
*    fileserver. example. com:/export/home/&
```

文件映射 /etc/auto.home 并不存在。

鉴于这些条件，让我们假设客户端系统需要从不同的服务器覆盖 NIS 映射 auto.home 和挂载主目录。在这种情况下，客户端将需要使用下面的 /etc/auto.master 映射：

```
/home -/etc/auto.home
+auto.master
```

而且，/etc/auto.home 映射包含下列项目：

```
*    labserver.example.com:/export/home/&
```

由于自动挂载只处理第一次出现的一个挂载点，/home 将包含/etc/auto.home 的内容，而不是 NIS auto.home 映射。

另外，如果您只是想增加几个条目的站点范围内的 auto.home 映射，请创建一个/ etc/auto.home 文件映射，并把您的新条目放在文件映射中而且其结束处包

括 NIS auto.home 映射。然后/etc/auto.home 文件映射可能类似于：

```
mydir someserver:/export/mydir
+auto.home
```

鉴于上面列出的 NIS auto.home 映射，ls /home 现在将输出：

```
beth joe mydir
```

最后一个例子如期运作，因为 autofs 知道不包括一个与正在读取的文件名称相同的文件映射的内容。因此，autofs 将移动到 nsswitch 配置中的下一个映射源。

12.3.4 使用 LDAP 存储自动挂载映射

LDAP 客户端库必须安装在所有配置好的系统上来检索来自 LDAP 的自动挂载程序映射。在中标麒麟可信操作系统 V6.0 上，openldap 包应该作为自动挂载程序的依赖进行自动挂载。要配置 LDAP 访问，请修改/etc/openldap/ldap.conf。确保您的网站架构正确地设置了 BASE、URI 和 schema。

rfc2307bis 描述了最近成立的在 LDAP 中存储自动加载映射的模式。要使用此模式，有必要从架构定义中删除注释字符以便在 autofs 配置（/etc/sysconfig/autofs）中对其进行设置。例如：

```
DEFAULT_MAP_OBJECT_CLASS="automountMap"
DEFAULT_ENTRY_OBJECT_CLASS="automount"
DEFAULT_MAP_ATTRIBUTE="automountMapName"
DEFAULT_ENTRY_ATTRIBUTE="automountKey"
DEFAULT_VALUE_ATTRIBUTE="automountInformation"
```

确保这些是配置中没有注释的唯一架构记录。请注意，automountKey 取代了 rfc2307bis 架构中的 cn 属性。示例配置的 LDIF 描述如下：

```
# extended LDIF
#
# LDAPv3
# base <> with scope subtree
# filter: (&(objectclass=automountMap)(automountMapName=auto.master))
# requesting: ALL
#
# auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com
objectClass: top
objectClass: automountMap
automountMapName: auto.master
# extended LDIF
#
```

```
# LDAPv3
# base <automountMapName=auto.master,dc=example,dc=com> with scope subtree
# filter: (objectclass=automount)
# requesting: ALL
#
# /home, auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com objectClass: automount
cn: /home
automountKey: /home
automountInformation: auto.home
# extended LDIF
#
# LDAPv3
# base <> with scope subtree
# filter: (&(objectclass=automountMap)(automountMapName=auto.home))
# requesting: ALL
#
# auto.home, example.com
dn: automountMapName=auto.home,dc=example,dc=com
objectClass: automountMap automountMapName: auto.home
# extended LDIF
#
# LDAPv3
# base <automountMapName=auto.home,dc=example,dc=com> with scope subtree
# filter: (objectclass=automount)
# requesting: ALL
#
# foo, auto.home, example.com
dn: automountKey=foo,automountMapName=auto.home,dc=example,dc=com
objectClass: automount
automountKey: foo
automountInformation: filer.example.com:/export/foo

# /, auto.home, example.com
dn: automountKey=/,automountMapName=auto.home,dc=example,dc=com
objectClass: automount
automountKey: /
automountInformation: filer.example.com:/export/&
```

12.4 常见 NFS 挂载选项

除了通过 NFS 在远程主机上挂载文件系统外，您也可以在挂载时指定其他选项以便使挂载共享更容易使用。这些选项可以和手动 mount 命令、/etc/fstab 设置以及 autofs 一起使用。

以下是 NFS 挂载常用的选项：

intr

如果服务器出现故障或无法到达， 允许 NFS 请求被打断。

lookupcache=mode

对于一个给定的挂载点， 指定内核如何管理目录条目的缓存。模式的有效参数包括 all、none 或 pos/positive。

nfsvers=version

指定 NFS 协议使用的版本，包括版本 2，3 或 4。对于运行多重 NFS 服务器的主机来说这是有用的。如果没有指定版本，NFS 将会使用内核和 mount 命令支持的最高版本。

选项 vers 与 nfsvers 相同，出于兼容性的原因都被包括在此版本中。

noacl

关闭所有 ACL 处理。在与旧版本的中标麒麟可信操作系统 V6.0 或 Solaris 接口时，可能需要这样处理，因为最新开发的 ACL 技术与旧系统不兼容。

nolock

禁用文件锁。当连接到旧的 NFS 服务器时，偶尔需要此设置。

noexec

防止执行所挂载文件系统上的二进制文件。如果该系统是一个正在挂载包含不兼容二进制文件的非 Linux 文件系统，这是非常有用的。

nosuid

禁止 set-user-identifier 或 set-group-identifier 比特位。这可以防止远程用户通过运行 setuid 程序来获得更高的权限。

port=num

port=num —指定 NFS 服务器端口的数值。如果 num 为 0（默认），那么 mount 查询端口号使用的远程主机 rpcbind 服务。如果远程主机的 NFS 守护进程不是用 RPCBIND 服务来注册的，TCP2049 的标准 NFS 端口号可用来作为替代。

rsize=num and wsize=num

通过设置一个较大的数据扇区（num，以字节为单位）加快在同一时间完成传输的 NFS 读取 (rsize) 和写入 (wsize)通讯。改变这些值时要小心，一些旧的 Linux 内核和网卡不能与较大的扇区协同工作。对于 NFSv2 或 NFSv3，这两个参数的默认值设置为 8192。对于 NFSv4，这两个参数的默认值设置为 32768。

sec=mode

在进行 NFS 连接的身份验证时，请指定要使用的安全类型。默认设置是 sec=sys，它通过利用 AUTH_SYS 验证 NFS 操作来使用本地 UNIX UID 和 GID。

sec=krb5 使用 Kerberos V6.0，而不是本地的 UNIX UID 和 GID 对用户进行身份验证。

sec=krb5i 使用 Kerberos V6.0 的用户身份验证，并执行完整的 NFS 操作检查，使用安全校验以防止数据被篡改。

sec=krb5p 使用 Kerberos V6.0 用于用户身份验证，完整性检查和加密 NFS 流量，以防止流量嗅探。这是最安全的设置，但它也涉及到了大部分的性能消耗。

tcp

命令 NFS 挂载使用 TCP 协议。

udp

命令 NFS 挂载使用 UDP 协议。

对于完整的选项列表和每个选项的更详细信息，请参阅 man mount 和 man nfs。关于通过 TCP 或 UDP 协议使用 NFS 的详细信息，请参阅第 12. 9 节，“通过 TCP 使用 NFS”。

12.5 启动或停止 NFS

要运行 NFS 服务器，rpcbind[1] 服务必须正在运行。要验证 rpcbind 是活动的，请使用以下命令：

```
service rpcbind status
```

备注：使用 service 命令启动，停止或重新启动一个守护进程需要 root 权限。

如果 rpcbind 服务正在运行中，则可以启动 NFS 服务。要启动 NFS 服务器，请使用 root 身份执行下列命令：

```
service nfs start
```

备注：要使 NFS 客户端和服务器都正常发挥作用，必须启动 nfslock。启动 NFS 锁，请使用下面的命令：service nfslock start

如果 NFS 被设置为在开机时启动，请通过运行 chkconfig -- list nfslock 确保 nfslock 也可以启动。如果 nfslock 不是设置为 on，这意味着每次启动计算机时您将需要手动运行 service nfslock start。要设置 nfslock 在开机时自动启动，请使用 chkconfig nfslock on。

只有 NFSv2 和 NFSv3 需要 nfslock 服务。

要停止服务器，请使用：

```
service nfs stop
```

restart 选项是重启 NFS 的快捷途径。这是编辑 NFS 配置文件后令配置更改生效的最有效方式。要重启服务器，请以 root 身份，键入：

```
service nfs restart
```

如果它目前正在运行，condrestart（有条件重新启动）选项只启动 nfs。此选项对于脚本是有用的，因为如果它没有运行，它不会启动守护进程。要有条件地重启服务器，请以 root 身份，键入：

```
service nfs condrestart
```

要重新加载 NFS 服务器配置文件，而无需重新启动服务，请以 root 身份，键入：

```
service nfs reload
```

12.6 NFS 服务器配置

配置 NFS 服务器有两种方法：

1） 通过手动编辑 NFS 配置文件，即/etc/exports。

2） 通过命令行，即通过 exportfs。

12.6.1 /etc/exports 配置文件

/etc/exports 文件控制将哪些文件系统导出到远程主机并指定选项。它遵循下列句法规则：

1） 空白行会被忽略。

2） 若要添加注释，请以 hash 标记（＃）开始。

3） 您可以用反斜杠（\）来打断比较长的表达式。

4） 每个导出的文件系统应该在自己的独立表达式上。

5） 放置在导出文件系统后面的授权主机的任何列表必须由空格字符分开。

6） 每个主机的选项必须直接放在主机标识符后的括号中，无须任何一个空格将主机和第一个括号分离开来。

导出文件系统的每个条目都具有以下结构：

```
export host(options)
```

上述结构使用下列变量：

export

正被导出的目录

host

共享导出文件的主机或网络

options

主机要使用的选项

您可以指定多个主机，以及为每个主机指定特定选项。要做到这一点，请把它们列在同一个以空格分隔的表达式上，每个主机名称附带其各自的选项（括号内），如：

```
export host1(options1) host2(options2) host3(options3)
```

关于指定主机名称的不同方法的更多信息，请参阅第 “12.6.4 主机名称格式”。

在最简单的形式中，/etc/exports 文件仅指定导出的目录和允许访问的主机，如下面的例子：

```
/exported/directory bob.example.com
```

在这里，bob.example.com 可以从 NFS 服务器挂载/exported/directory/。因为在这个例子中没有指定任何选项，NFS 将使用默认设置，其中：

ro

导出文件系统是只读的。远程主机不能改变文件系统上所共享的数据。要允许主机对文件系统(i.e. read/write)做出变更，请指定 rw 选项。

sync

在先前请求作出的更改写入到磁盘之前，NFS 服务器将不会对请求作出任何回应。反之，如果要启用异步写入操作，请指定选项 async。

wdelay

如果它怀疑另一个写入请求迫在眉睫，NFS 服务器将延迟写入磁盘。这可以提高性能，因为它减少了磁盘必须由单独写命令访问的次数，从而降低了写入开销。要禁用此功能，指定 no_wdelay，请注意，如果还指定了默认的 sync 选项，no_wdelay 也是唯一可用的。

root_squash

这可以防止远程连接的 root 用户获得 root 权限，相反，NFS 服务器将分配给他们用户 ID nfsnobody。这将有效地将远程 root 用户的权限“压缩”到最低的本地用户权限，防止远程服务器上未经授权写入的可能。要禁用 root 权限压缩，请指定 no_root_squash。

要压缩每个远程用户（包括 root）的权限，请使用 all_squash 命令。要指定 NFS 服务器应该从一个特定的主机分配给远程用户的用户和组 ID，请使用 anonuid 和 anongid 选项，分别如下：

```
export host(anonuid=uid,anongid=gid)
```

在这里，uid 和 gid 分别是用户 ID 号码和组 ID 号码。anonuid 和 anongid 选项使您可以创建一个特殊的用户/组帐户用于远程 NFS 用户共享。

 提示：默认情况下，访问控制列表（ACL）为中标麒麟可信操作系统 V6.0 下的 NFS 所支持。要禁用此功能，请在导出文件系统时指定 no_acl 选项。

每个导出文件系统的各个默认值必须被明确重写。例如，如果未指定 **rw** 选项，则导出文件系统共享为只读。以下是来自重写了两个默认选项的/etc/exports 文件的一个示例行：

```
/another/exported/directory 192.168.0.3(rw,async)
```

在这个例子中，192.168.0.3 可以挂载/another/exported/directory/ read/write，而且所有到磁盘的写入都是异步的。关于导出选项的更多信息，请参阅 man exportfs。

此外，其他选项都可用于没有指定默认值的位置。这些措施包括：禁用子树检查的能力，允许访问不安全端口的能力，以及允许不安全的文件锁定的能力（对某些早期 NFS 客户端实现很有必要）。这些较少使用的选项的详细信息，请参阅 man exports。

 警告：/etc/exports 文件格式非常精确，特别是在空格字符的使用方面。记住要始终将导出的文件系统与主机分开，并且要用一个空格字符将主机彼此分开。除了注释行之外，该文件中应该没有其他的空格字符。

例如，下面的两行并不意味着同样的事情：

```
/home bob.example.com(rw)
/home bob.example.com (rw)
```

第一行只允许来自 bob.example.com read/write 的用户访问/home 目录。第二行允许来自 bob.example.com 的用户以只读方式挂载该目录（默认的），而其他用户可以以读/写方式挂载。

12.6.2 exportfs 命令

每个经由 NFS 正被导出到远程用户的文件系统，以及对这些文件系统的访问级别都被列在/etc/exports 文件中。NFS 服务启动时，/usr/sbin/ exportfs 命令启

动并读取这个文件，将控制权传递到 rpc.mountd（如果 NFSv2 或 NFSv3 的）用于实际挂载过程中，然后传递到 rpc.nfsd 文件，此时文件系统对远程用户可用。

当手动发出命令时，/usr/sbin/exportfs 命令允许 root 用户选择性地导出或不导出某些目录，而无需重新启动 NFS 服务。当给出适当的选项时，/usr/sbin/exportfs 命令将导出的文件系统写入到/var/lib/nfs/xtab 中。由于在决定对文件系统的访问权限时，rpc. mountd 指的是 xtab 文件，因此，对导出的文件系统列表的修改将立即生效。

下面是可用于/usr/sbin/exportfs 的常用选项列表：

-r

通过在/etc/lib/nfs/xtab 中构建新的导出清单，使/etc/exports 中列出的所有目录都可以被导出。这个选项有效地刷新了导出列表中对/etc/exports 所做的任何更改。

-a

所有目录被导出或不导出，取决于哪些其他选项被传递给了/usr/sbin/exportfs。如果没有指定其他选项，/usr/sbin/exportfs 导出/etc/exports 中指定的所有文件系统。

-o 文件系统

指定/etc/exports 中未列出的但却要导出的目录。用需要导出的附加文件系统替换该文件系统。这些文件系统必须以/etc/exports 中指定的方式进行格式化。关于/etc/exports 句法的更多信息，请参见 12.6.1 节，“ etc/exports 配置文件”。在将其永久添加到要导出的文件系统列表之前，此选项通常用于测试导出的文件系统。

-i

忽略/etc/exports，命令行给出的唯一选项用来定义导出的文件系统。

-u

不导出所有的共享目录。命令/usr/sbin/exportfs -ua 暂停 NFS 文件共享，同时保留所有 NFS 守护进程。重新启用 NFS 共享，使用 exportfs -r。

-v

冗长操作，执行 exportfs 命令时，导出的或未导出的文件系统将被显示得更加详尽。

如果没有选项被传递给 exportfs 命令，它会显示当前导出的文件系统列表。关于 exportfs 命令的更多信息，请参阅 man exportfs。

12.6.2.1 NFSv4 环境下使用 exportfs

exportfs 命令被用于维护导出文件系统的 NFS 表。无参数使用时，exportfs 显示所有导出的目录。

由于 NFSv4 不再使用 MOUNT 协议，但是 NFSv2 和 NFSv3 协议仍在使用，这样文件系统的挂载就发生了改变。

一个 nfsv4 客户端现在有能力将所有的 nfsv4 服务器的导出共享目录信息视为单一的文件系统，这就是所谓的 nfsv4 伪文件系统。在中标麒麟可信操作系统 V6.0 中，伪文件系统被认定为一个单一的、真正的文件系统，在导出信息中以 fsid=0 选项进行确定。

12.6.3 在防火墙下运行 NFS

NFS 需要 rpcbind，为 RPC 服务动态地分配端口，并可能导致配置防火墙规则方面的问题。要允许客户端访问防火墙后面的 NFS 共享，请编辑/etc/sysconfig/nfs 配置文件来控制所需的 RPC 服务在哪些端口上运行。

默认情况下，/etc/sysconfig/nfs 可能不存在于所有系统上。如果它不存在，创建它，并添加以下变量，用一个未使用的端口号来取代端口（另外，如果该文件存在，取消注释，并根据需要更改默认项）：

MOUNTD_PORT=port

控制 mountd (rpc.mountd) 使用哪一个 TCP 和 UDP 端口。

STATD_PORT=port

控制 status (rpc.statd)使用哪一个 TCP 和 UDP 端口。

LOCKD_TCPPORT=port

控制 nlockmgr (lockd)使用哪个 TCP 端口。

LOCKD_UDPPORT=port

控制 nlockmgr (lockd)使用哪个 UDP 端口。

如果 NFS 无法启动，请检查/var/log/messages。在通常情况下，如果你指定一个已在使用中的端口号，NFS 将无法启动。编辑/etc/sysconfig/nfs 后，使用 service nfs restart 重新启动 NFS 服务。运行 rpcinfo -p 命令确认更改。

要配置一个允许 NFS 的防火墙，请执行以下步骤：

1） 允许用于 NFS 的 TCP 和 UDP 端口 2049。

2） 允许 TCP 和 UDP 端口 111(rpcbind/sunrpc)。

3） 允许以 MOUNTD_PORT="port" 指定的 TCP 和 UDP 端口。

4） 允许以 MOUNTD_PORT="port" 指定的 TCP 和 UDP 端口。

5） 允许以 LOCKD_TCPPORT="port"指定的 TCP 端口。

6） 允许以 LOCKD_UDPPORT="port"指定的 UDP 端口。

12.6.4 主机名格式

主机可以有以下几种形式：

1） 单机

一个完全合格的域名（能够被服务器解析），主机名（服务器可以解析），或一个 IP 地址。

2） 通过通配符指定的系列机型

使用*或？字符指定字符串匹配。通配符一般不会和 IP 地址一起使用，然而，如果反向 DNS 查找失败，他们可能会意外地发挥作用。当以完全合格的域名指定通配符时，点（.）不包括在通配符内。例如， *.example.com 包括 one.example.com 但并不包括 one.two.example.com。

3） IP 网络

使用 a.b.c.d/z，其中 abcd 代表网络，z 是掩码中的比特数（例如 192. 168. 0. 0/24）。另一个可接受的格式是 abcd/netmask，其中 abcd 代表网络，netmask 是子网掩码（例如，192.168.100.8/255.255.255.0）。

4） 网络组

使用格式@group-name，group-name 是 NIS 网络组（netgroup）名称。

12.7 保护 NFS 的安全

NFS 非常适合于大量的已知主机以透明的方式分享整个文件系统。然而，易于使用带来各种潜在的安全问题。在将 NFS 文件系统导出到服务器上或安装在客户端上时，请考虑以下的部分。这样做，以便最大限度地减少 NFS 的安全隐患，更好地保护服务器上的数据。

12.7.1 NFSv2 或 NFSv3 环境下的主机访问

NFS 控制谁可以在发出挂载请求的主机上挂载导出的文件系统，而不是实

际上使用该文件系统的用户。主机必须被给予明确权利以便挂载导出的文件系统。除了通过文件和目录权限以外，访问控制对于用户来说是不可能的。换句话说，一旦一个文件系统是通过 NFS 导出，任何连接到 NFS 服务器上的远程主机上的任何用户都可以访问共享数据。要限制潜在的风险，管理员通常允许只读访问或将用户权限压缩到一个普通的用户和组 ID。

不幸的是，这些解决方案阻止 NFS 共享以最初计划的方式被使用。

此外，如果攻击者获得导出 NFS 文件系统的系统所使用的 DNS 服务器的控制，与特定的主机名或完全限定的域名相关联的系统可以指向未经授权的计算机。在这一点上，未经授权的计算机是允许挂载 NFS 共享的系统，因为没有用户名或密码的信息被交换用来为 NFS 挂载提供额外的安全性。

当通过 NFS 导出目录时，应尽量少使用通配符，因为通配符范围包括比预期更多的系统也是可能的。

您还可以通过 TCP wrapper 来限制对 rpcbind[1] 服务的访问。用 iptables 创建规则也可以限制对 rpcbind、rpc.mountd 和 rpc.nfsd 所使用端口的访问。

确保 NFS 和 rpcbind 的更多信息，请参阅 man iptables。

12.7.2 NFSv4 中的主机访问

NFSv4 的发布为 NFS 认证和安全带来了一场革命。NFSv4 强制执行 RPCSEC_GSS 内核模块，Kerberos 版本 5 GSS - API 机制，SPKM- 3 以及 LIPKEY。随着 NFSv4 的诞生，强制性的安全机制用于对个人用户进行身份验证，而不是用于在 NFSv2 和 NFSv3 环境下使用的客户端机器。因此，出于安全原因，我们建议只要有可能就使用高于其他版本的 NFSv4。

 备注：假设在配置 NFSv4 服务器之前，Kerberos 票证授予服务器（KDC）已经被正确地安装和配置。Kerberos 是一种网络身份验证系统，它允许客户端和服务器通过使用对称加密和受信任的第三方即 KDC 进行相互验证。

鉴于前者的功能和广泛部署，NFSv4 中包括的 ACL 支持是基于 Microsoft Windows NT 模式的而不是 POSIX 模式。NFSv2 和 NFSv3 没有提供对原始 ACL 属性的支持。

NFSv4 的另一个重要的安全功能是取消为挂载文件系统而使用的 MOUNT 协议。由于它处理文件句柄的方式，该协议提出了可能的安全漏洞。

关于 RPCSEC_GSS 框架的更多信息，包括 rpc.svcgssd 和 rpc.gssd 之间如何互相操作，请参考 http://www.citi.umich.edu/projects/nfsv4/gssd/。

12.7.3 文件权限

一旦 NFS 文件系统由远程主机以读/写方式进行挂载，每个共享文件拥有的唯一保护就是其权限。如果共享相同用户 ID 值的两个用户装入相同的 NFS 文件系统，他们可以修改对方的文件。此外，客户端系统上任何以 root 身份登录的人都可以使用 su - 命令通过 NFS 共享来访问任何文件。

默认情况下，访问控制列表（ACL）为中标麒麟可信操作系统 V6.0 下的 NFS 所支持。我们建议您启用此功能。

默认情况下，NFS 导出文件系统时，使用 root 权限压缩（Root squashing）。在本地机器上以 root 用户身份访问 NFS 共享的任何人的用户 ID 都被设为 nobody。root 权限压缩由默认选项 root_squash 所控制，有关此选项的更多信息，请参见 12.6.1/etc/exports 配置文件。如果可能的话，不要禁用 root 权限压缩。

当以只读身份导出 NFS 共享时，可以考虑使用 all_squash 选项。这个选项使得每个访问导出文件系统的用户取得 nfsnobody 用户的用户 ID。

12.8 NFS 和 rpcbind

备注：以下章节仅适用于 NFSv2 或 NFSv3，要求为逆兼容性而启动 rpcbind 服务。

rpcbind 实用程序会在 RPC 服务与 RPC 监听的端口之间构建一种映射关系。RPC 进程在启动时，会通知 rpcbind， 注册它们（RPC 程序或进程）所监听的端口以及 RPC 程序对应的编号。然后，客户端系统用一个特定的 RPC 程序号来联系服务器上的 rpcbind。rpcbind 服务将客户端重新定向到正确的端口号，所以它可以和所要求的服务取得通信。

由于基于 RPC 的服务依靠 rpcbind 与传入的客户端请求取得连接，这些服务的任何一项启动之前 rpcbind 服务必须可用。

rpcbind 服务使用 TCP wrapper 进行访问控制，rpcbind 的访问控制规则影响所有基于 RPC 的服务。另外，也可以为每个 NFS RPC 守护进程指定访问控制规则。rpc.mountd 和 rpc.statd 的手册页中包含关于这些规则的精确语法的信息。

12.8.1 NFS 和 rpcbind 的故障排除

由于 rpcbind1 提供了 RPC 服务和用于他们之间通信的端口号码之间的协调机制，在故障排除时，使用 rpcbind 来查看当前的 RPC 服务状态也是有用的。rpcinfo 命令显示每个基于 RPC 的服务使用的端口号，RPC 程序号，版本号，和

IP 协议类型（TCP 或 UDP）。

为了确保为 rpcbind 启用正确的 NFS 基于 RPC 的服务，请以 root 身份发出以下命令：

```
rpcinfo -p
```

以下是这个命令的范例输出：

```
program  vers    proto    port
100021   1   udp  32774   nlockmgr
100021   3   udp  32774   nlockmgr
100021   4   udp  32774   nlockmgr
100021   1   tcp  34437   nlockmgr
100021   3   tcp  34437   nlockmgr
100021   4   tcp  34437   nlockmgr
100011   1   udp  819  rquotad
100011   2   udp  819  rquotad
100011   1   tcp  822  rquotad
100011   2   tcp  822  rquotad
100003   2   udp  2049    nfs
100003   3   udp  2049    nfs
100003   2   tcp  2049    nfs
100003   3   tcp  2049    nfs
100005   1   udp  836  mountd
100005   1   tcp  839  mountd
100005   2   udp  836  mountd
100005   2   tcp  839  mountd
100005   3   udp  836  mountd
100005   3   tcp  839  mountd
```

如果一项 NFS 服务没有正常启动，rpcbind 将无法把来自客户端的用于该项服务的 RPC 请求映射到正确的端口。在许多情况下，如果 NFS 没有出现在 rpcinfo 输出中，重新启动 NFS 将导致该项服务正确注册 rpcbind 并开始工作。启动 NFS 的说明，请参阅 “12.5 启动和停止 NFS”。

欲了解更多信息和 rpcinfo 选项列表，请参阅其手册页。

12.9 通过 TCP 使用 NFS

NFS 的默认传输协议是 TCP， 然而，中标麒麟可信操作系统 V6.0 内核包括对 UDP 上 NFS 的支持。要通过 UDP 使用 NFS，在客户端系统上挂载 NFS 导出文件系统时，须将 mount 选项-o udp 包括在内。注意：UDP 上的 NFSv4 是不符合标准的，因为 UDP 没有拥塞控制的特点，正因为如此，UDP 上的 NFSv4 是不被支持的。

配置 NFS 文件系统导出有三种方式：

1） 通过命令行（客户端）按需导出

2） 通过/etc/fstab 文件（客户端）自动导出

3） 通过 autofs 的配置文件如/etc/auto.master and /etc/auto.misc（带有 NIS 的服务器端）自动导出

例如，通过命令行（客户端）按需导出

```
mount -o udp shadowman.example.com:/misc/export /misc/local
```

当 NFS 挂载在/etc/fstab（客户端）中被指定时：

```
server:/usr/local/pub    /pub    nfs    rsize=8192,wsize=8192,timeo=14,intr,udp
```

当 NFS 挂载为了 NIS 服务器而在 autofs 配置文件中被指定时，可用于 NIS 启用工作站：

```
myproject    -rw,soft,intr,rsize=8192,wsize=8192,udp penguin.example.net:/proj52
```

由于默认是 TCP，如果未指定-o udp 选项，通过 TCP 可以访问 NFS 导出的文件系统。

使用 TCP 的优势包括以下内容：

1） UDP 只确认数据包完成，而 TCP 确认每一个数据包。这导致在安装共享文件时，使用 TCP 的高负载网络在性能方面获得增益。

2） TCP 比 UDP 具有更好的拥塞控制。在一个非常拥挤的网络上，UDP 数据包是第一个被删除的数据包。这意味着，如果 NFS 以 8k 数据块为单位写入数据，但所有这 8k 必须基于 UDP 进行重传。由于 rcp 的可靠性机制，每次只有 8k 的数据被传递。

3） TCP 也有较好的错误检测功能。当一个 TCP 连接中断时（由于服务器不可用），此时客户端会停止发送数据，一旦服务恢复正常，它会重新开启连接进程。但对于 UDP 因为它是无连接的， 导致客户端会继续发送数据，直至服务重新建立一个连接。

使用 tcp 最大的不足在于，由于与 tcp 协议相关联的一些附加成本，性能会受到轻微的影响。

12.10 参考

管理 NFS 服务器也是一个挑战。许多选项，其中包括不少本章中没有提到的，可用于导出或挂载 NFS 共享文件。更多信息，请查阅以下源文件。

安装了的文档

/usr/share/doc/nfs-utils-version/ —这个目录包含了大量关于 Linux 的 NFS 执行情况的信息，包括查看各种 NFS 配置和文件传输性能方面的影响。

man mount —包含全面审视用于 NFS 服务器和客户端配置的挂载选项。

man fstab —给出用于在启动时挂载文件系统的**/etc/fstab** 文件格式的详细信息。

man nfs — 提供关于 NFS 特定的文件系统的导出和挂载选项的详细信息。

man exports —显示导出 NFS 文件系统时/etc/exports 文件中使用的常见选项。

有用的网站

http://nfs.sourceforge.net/ —Linux 的 NFS 项目的主页和项目状态更新的理想场所。

http://www.citi.umich.edu/projects/nfsv4/linux/ —Linux 2.6 内核资源的 NFSv4。

http://www.nfsv4.org[2] —NFS 版本 4 和所有相关标准的首页。

http://www.vanemery.com/Linux/NFSv4/NFSv4-no-rpcsec.html —描述带有 Fedora Core2 的 NFSv4 的详细信息，包括 2.6 内核。

http://www.nluug.nl/events/sane2000/papers/pawlowski.pdf —关于 NFS 版本 4 协议的特性和增强功能的一本出色的白皮书。

http://wiki.autofs.net —autofs wiki 索引，讨论，文档和增强功能。

相关书籍

Hal Stern、Mike Eisler 和 Ricardo Labiaga 编著的 Managing NFS and NIS；O'Reilly & Associates 为许多不同的可用 NFS 导出和挂载选项提供了一个很好的参考指南。

Brent Callaghan 编著的 NFS Illustrated Addison – Wesley 出版公司 – 提供 NFS 与其他网络文件系统之间的对比，并显示关于 NFS 通讯如何发生的详细信息。

13. FS-Cache

FS – Cache 是一个持久性的本地高速缓存，可以被文件系统用来获取通过网络检索到的数据并将其缓存在本地磁盘上。对于访问网络挂载的文件系统（例

如，NFS）中数据的用户来说，这将有助于最大限度地减少网络流量。

下图是关于 FS 高速缓存工作原理的一个高层次例证：

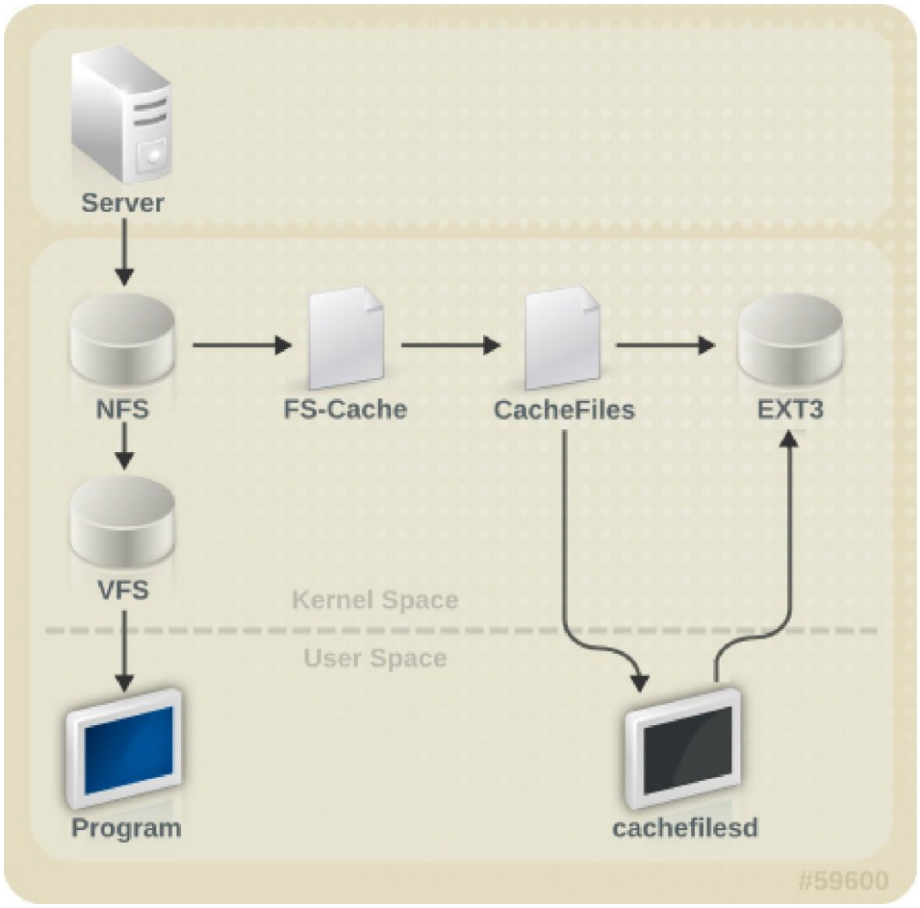


图 13-1 FS-Cache 概述

FS – Cache 的设计对用户和系统管理员来说要尽可能透明。不同于 Solaris 上的 cachefs ，FS- Cache 允许服务器上的文件系统直接与客户端的本地高速缓存进行交互，而无需创建一个 overmounted 文件系统。使用 NFS，挂载选项命令客户端在 FS 缓存启用的情况下安装 NFS 共享。

FS缓存不会改变通过网络进行工作的文件系统的基本操作-它只是为该文件系统提供了一个可以持久缓存数据的地方。例如，一个客户端仍然可以挂载 NFS 共享，无论 FS - Cache 是否启用。此外，缓存 NFS 可以处理不适合缓存（不论个别或集体）的文件，因为文件可以部分缓存并且不一定必须从最前面的位置开始读取。FS– Cache 还隐藏着发生在客户端文件系统驱动程序中缓存中的所有 I / O 错误。

要提供缓存服务，FS– Cache 需要后台程序缓存。后台程序缓存是一个存储驱动程序，用来配置以提供缓存服务（即 cachefiles）。在这种情况下，FS– Cache

需要一个挂载的基于块的文件系统，可以支持 bmap 和扩展属性（如 EXT3）作为其后台程序缓存。

FS- Cache 不能随意缓存任何文件系统，无论是通过网络还是其他方式：共享文件系统的驱动程序必须改变，让 FS 缓存，数据存储/检索和元数据的设置和验证之间实现互动。FS- Cache 需要来自高速缓存文件系统的索引键和相关性数据来支持持久性存储：索引键匹配文件系统对象与缓存对象，相关性数据确定缓存对象是否仍然有效。

13.1 性能保证

FS 缓存并不能保证提高性能。相反，使用后台程序缓存会带来性能上的损失：例如，缓存 NFS 共享添加磁盘访问到跨网络查找。而 FS- Cache 力图尽可能实现异步，在不可能实现异步的情况下，采取同步路径（如读取）。

例如，使用 FS- Cache 去缓存其它空载千兆网络上的两台计算机之间的 NFS 共享不会显示任何文件访问方面的性能改进。相反，NFS 请求将从服务器的内存而不是从本地磁盘更快地得到满足。

因此，FS- Cache 的使用是各因素之间作用的结果。如果 FS - Cache 被用来缓存 NFS 流量，例如，它可能会使客户端的速度慢下来，可以通过满足本地读取请求来大量减少网络和服务器负载而无需耗费网络带宽。

13.2 设置高速缓存

目前，中标麒麟可信操作系统 V6.0V6.0 只提供 cachefiles 后台缓存程序。cachefilesd 守护进程启动和管理 cachefiles。/etc/cachefilesd.conf 文件控制 cachefiles 如何提供缓存服务。要配置这种类型的后台缓存程序，必须安装 cachefilesd 软件包。

配置后台缓存程序的第一个设置是用哪个目录作为高速缓存。要完成此配置，请使用如下参数：

```
dir /path/to/cache
```

通常情况下，后台缓存程序目录设置在/etc/cachefilesd.conf 作为/var/cache/fscache，如：

```
dir   /var/cache/fscache
```

FS-Cache 将缓存存储在容纳/path/to/cache 的文件系统中。在笔记本电脑上，最好使用根文件系统（/）作为主机文件系统，但对于一个台式机，它会更加谨

慎地专门为高速缓存挂载缓存磁盘分区。

支持 FS-Cache 后台缓存程序所需功能的文件系统包括中标麒麟可信操作系统 V6.0V6.0 下列文件系统：

1） ext3 (启用扩展属性)

2） ext4

3） BTRFS

4） XFS

主机文件系统必须支持用户自定义的扩展属性，FS- Cache 使用这些属性来存储相关性维护信息。为 ext3 文件系统使用户自定义的扩展属性（i.e. device），请使用：

```
tune2fs -o user_xattr /dev/device
```

另外，文件系统的扩展属性可以在安装时启用，如：

```
mount /dev/device /path/to/cache -o user_xattr
```

后台缓存程序通过让容纳缓存的分区保持一定量的可用空间来进行工作。扩充和收缩缓存以回应耗尽自由空间的系统中的其他元素，这使得它可以在根文件系统上安全使用（例如，在一台笔记本电脑上）。FS- Cache 设置这种行为的默认值，它可以通过缓存剔除限制来进行配置。有关配置缓存剔除限制的详细信息，请参阅 “13.4 设置高速缓存剔除限制”。

一旦配置文件就位，启动 cachefilesd 守护进程：

```
service cachefilesd start
```

要配置 cachefilesd 以便在开机时启动，请以 root 身份执行以下命令：

```
chkconfig cachefilesd on
```

13.3 在 NFS 环境下使用高速缓存

NFS 不会使用缓存，除非明确指示。要配置 NFS 挂载来使用 FS- Cache，包括了 mount 命令的-o fsc 选项，如：

```
mount nfs-share:/ /mount/point -o fsc
```

所有到/mount/point 下文件的访问将通过缓存，除非该文件是直接 I / O 打开或写入形式（更多信息，参见 “13.3.2 NFS 环境下的缓存限制”）。NFS 索引缓存使用 NFS 文件句柄的内容，而不是文件名，这意味着，硬链接文件正确地共享缓存。

NFS 的版本 2，3，和 4 都支持高速缓存。然而，每个版本使用不同的分支

版本来实现缓存。

13.3.1 缓存共享

NFS 缓存共享有几个潜在的问题需要处理。因为缓存是持久的，在高速缓存中的数据块以四级密匙序列进行索引：

第一层： 服务器的详细信息

第二层： 某些挂载选项， 安全类型：FSID， 唯一标志（uniquifier）

第三层： 文件柄

第四层： 文件中的页码

为了避免超级块之间的相关性管理中存在的问题，所以希望缓存数据的 NFS 超级块有独特第二层密匙。通常情况下，两个具有相同的源卷和选项的 NFS 挂载将共享一个超级块，从而共享缓存，即使他们在该卷装入不同的目录。采取以下两种 mount 命令：

```
mount home0:/disk0/fred /home/fred -o fsc
mount home0:/disk0/jim /home/jim -o fsc
```

在这里，/home/fred 和 /home/jim 可能会共享超级块，因为他们有相同的选项，尤其是当他们来自 NFS 服务器上(home0)相同的卷源/ 分区时。现在，请考虑以下两个并发的 mount 命令：

```
mount home0:/disk0/fred /home/fred -o fsc,rsize=230
mount home0:/disk0/jim /home/jim -o fsc,rsize=231
```

在这种情况下，/home/fred 和 /home/jim 不会分享超级块，因为他们有不同的网络接入参数，它们是 2 级密匙的一部分。这同样适用于以下的挂载序列：

```
mount home0:/disk0/fred /home/fred1 -o fsc,rsize=230
mount home0:/disk0/fred /home/fred2 -o fsc,rsize=231
```

在这里，两个子树（/home/fred1 和 /home/fred2）的内容将被缓存两次。

避免超级块共享的另一种方式是用 nosharecache 参数明确地将其抑制。使用同样的例子：

```
mount home0:/disk0/fred /home/fred -o nosharecache,fsc
mount home0:/disk0/jim /home/jim -o nosharecache,fsc
```

然而，在这种情况下，只有一个超级块将被允许使用高速缓存，因为没有什么可以用来区分 home0:/disk0/fred 和 home0:/disk0/jim 的 2 级密匙。为了解决这个问题，至少有一个挂载要添加上唯一识别符，即 i.e：fsc=unique-identifier。例如：

```
mount home0:/disk0/fred /home/fred -o nosharecache,fsc
mount home0:/disk0/jim /home/jim -o nosharecache,fsc=jim
```

在这里，唯一的标识符 jim 将被添加到/home/jim 缓存所用的 2 级密匙中。

13.3.2 NFS 环境下的缓存限制

从一个直接 I/O 共享文件系统打开一个文件将自动绕过缓存。这是因为这种类型的访问必须直接指向服务器。

从一个共享的写入文件系统打开一个文件将无法在 NFS 版本 2 和 3 环境下起作用。这些版本的协议没有为客户端提供足够的相关性管理信息来检测从另一个客户端到同一文件的并发写入。

因此，从一个共享文件系统打开一个文件，无论是直接 I/O 还是写入都将刷新该文件的缓存副本。FS－Cache 不会再次缓存该文件，直到它不再以直接 I/O 或写入方式打开。

此外，此版本 FS－ Cache 只缓存普通的 NFS 文件。FS 缓存将不缓存目录，符号链接，设备文件，FIFO 和 sockets。

13.4 设置缓存剔除范围

cachefilesd 守护进程将来自共享文件系统的远程数据缓存在磁盘上的可用空间。这可能会消耗所有可用空间，如果根分区包含在磁盘中，结果可能更糟。为了控制这种情况，cachefilesd 试图通过从缓存中删除旧的对象（即最近很少访问的），以保持一定量的自由空间。这种行为被称为高速缓存剔除。

当处理文件系统的大小时，CacheFiles 剔除行为通过/etc/cachefilesd.conf 中的三种设置来控制：

brun N%

如果可用空间量上升到磁盘总容量的 N%以上，cachefilesd 禁用剔除。

bcull N%

如果可用空间量下降到磁盘总容量的 N%以下，cachefilesd 开始剔除。

bstop N%

如果可用空间量下降到低于 N%，cachefilesd 将不再分配磁盘空间，直到剔除可用空间量提高到 N%以上。

有些文件系统对它们实际上可以支持的文件数量有限制（例如，ext3 最多只能支持 32000 个文件）。这使得 CacheFiles 在不触发 bcull 或 bstop 的情况下可能

达到文件系统支持的最大文件数量。为了解决这个问题，cachefilesd 也试图保持文件数量在文件系统的限额以下。这种行为由以下设置来控制：

frun N%

如果文件系统可以进一步容纳的文件的数量降低到其最大文件限额的 N%以下，cachefilesd 禁用剔除。例如，frun 5%，如果文件系统可以容纳超过 1600 个文件，或者如果文件的数量低于其限额的 95%，即 30400 个文件，cachefilesd 将禁用 ext3 文件系统上的剔除。

fcull N%

如果文件系统可以进一步容纳的文件的数量上升到其最大文件限额的 N%以上，cachefilesd 开始剔除。例如，fcull 5%，如果文件系统可以容纳超过 1600 个文件，或者如果文件的数量超过其限额的 95%，即 30400 个文件，cachefilesd 将开始 ext3 文件系统上的剔除。

fstop N%

如果文件系统可以进一步容纳的文件数量上升到其最大文件限额的 N%以上，cachefilesd 将不再分配磁盘空间直到剔除行为将文件的数量下降到限额的 N%以下。例如，fstop 5%，cachefilesd 将不再容纳磁盘空间，直到剔除行为将文件数量下降到限额的 95%以下，即 30400 个文件。

每个设置的默认值 N 如下：

brun/frun — 10%

bcull/fcull — 7%

bstop/fstop — 3%

配置这些设置时，以下条件必须成立：

0 <= bstop < bcull < brun < 100

0 <= fstop < fcull < frun < 100

13.5 统计信息

FS- Cache 还不断跟踪一般性统计资料。查看此信息，请使用：

```
cat /proc/fs/fscache/stats
```

FS- Cache 统计信息包括决策点和对象计数器。FS 缓存提供的统计数据的更多细节，请参阅下面的内核文件：

/usr/share/doc/kernel-doc-version/Documentation/filesystems/caching/fscache.txt。

13.6 参考

关于 cachefilesd 以及如何对其进行配置的详细信息， 请参考 man cachefilesd 和 man cachefilesd.conf。下面的内核文件还提供了额外的信息：

/usr/share/doc/cachefilesd-0.5/README

/usr/share/man/man5/cachefilesd.conf.5.gz

/usr/share/man/man8/cachefilesd.8.gz

有关 FS-Cache 的一般信息，包括其设计的约束条件细节，现有的统计资料和具备的能力，请参阅下面的内核文件：

/usr/share/doc/kernel-doc-version/Documentation/filesystems/caching/fscache.txt

14. 加密文件系统

中标麒麟可信操作系统 V6.0V6.0 现在支持 eCryptfs，即“伪文件系统”它以每个文件为基础来提供数据和文件名加密。“伪文件系统”一词是指 eCryptfs 没有盘上格式这一事实，相反，它是一个驻留在实际文件系统上的文件系统层。eCryptfs 层提供加密功能。

eCryptfs 工作起来像绑定的挂载一样，因为它拦截写入到底层（即加密）文件系统的文件操作。Cryptfs 层在底层文件系统中的文件的元数据中添加了一个头。此元数据描述了该文件的加密，在它被传递到加密文件系统之前，eCryptfs 对文件数据进行了加密。此外，eCryptfs 还可以加密文件名称。

eCryptfs 不是一个盘上的文件系统，因此，没有必要通过 mkfs 之类的工具创建它。相反，eCryptfs 通过发出一种特殊的 mount 命令开始实施。要管理 eCryptfs 保护的文件系统，必须先安装 ecryptfs- utils 包。

14.1 挂载一个加密的文件系统

用 eCryptfs 和挂载来加密文件系统最简单的方法是交互方式。启动此进程，请执行以下命令：

```
mount -t ecryptfs /source /destination
```

用 eCryptfs 来加密一个目录层次结构（即/source）是指把它挂载到一个被 eCryptfs 加密的挂载点（即/destination）上。所有到/destination 的文件操作将通过加密传递到底层/source 文件系统。然而，在某些情况下，对于一个文件操作

而言，在不通过 eCryptfs 层的情况下直接修改/source 也是可能的。但这样一来可能导致某些冲突。

这是为什么在大多数环境中，我们建议/source 和 /destination 应该尽量相同。例如：

```
mount -t ecryptfs /home /home
```

这实际上意味着加密一个文件系统并将其挂载在它本身上。这样做有助于确保所有到/home 的文件操作通过 eCryptfs 层。

在互动的加密/挂载过程中，mount 将允许配置以下设置：

加密密匙类型， openssl、tspi 或 passphrase。当选择 passphrase 时，mount 将要求其一。

1） 密码：aes、blowfish、des3_ede、cast6 或 cast5。

2） 密码字节数：16、32、24。

3） 是否 plaintext passthrough 被启用。

4） 是否 filename encryption 被启用。

互动挂载的最后一步后，mount 命令将显示所有做出的选择并执行挂载。此输出包括每个选定的设置的命令行选项等值。例如， 在 plaintext passthrough 和 filename encryption 都禁用的情况下， 采用密匙类型 passphrase 来挂载/home，密码 aes ， 密匙字节数 16 ，将会有如下输出：

```
Attempting to mount with the following options:
ecryptfs_unlink_sigs
ecryptfs_key_bytes=16
ecryptfs_cipher=aes
ecryptfs_sig=c7fed37c0a341e19
Mounted eCryptfs
```

在此显示中的选项可以随后被直接传递给命令行来加密和挂载使用相同配置的文件系统。要做到这一点，使用每个选项作为 mount 的- o 选项的参数。例如：

```
mount -t ecryptfs /home /home -o ecryptfs_unlink_sigs \ecryptfs_key_bytes=16
ecryptfs_cipher=aes ecryptfs_sig=c7fed37c0a341e19[1]
```

14.2 其他信息

eCryptfs 及其挂载选项的更多信息，请参阅 man ecryptfs（由 ecryptfs-utils 软件包提供）。下面的内核文件（kernel-doc 软件包所提供的）还提供关于 eCryptfs

其他信息：

/usr/share/doc/kernel-doc-version/Documentation/filesystems/ecryptfs.txt

15. 独立磁盘冗余阵列(RAID)

RAID 的基本思想是将多个小型廉价的磁盘驱动器合并成一个阵列来完成一个大型的昂贵的驱动器所能达到的性能或冗余目标。驱动器的阵列就好像是一台作为一个单一逻辑存储单元或驱动器的计算机。

15.1 什么是 RAID?

RAID 允许跨越多个磁盘传播信息。RAID 使用一些技术如磁盘条带化（RAID 级别 0），磁盘镜像（RAID 级别 1）和部分磁盘条带化（RAID 级别 5）来实现冗余校验，降低延迟，增加带宽，并将从硬盘崩溃恢复的能力最大化。

通过把它分解成大小一致的数据块（虽然其它值也是可以接受的，但通常是 256K 或 512K），RAID 把数据分布在阵列中的每个驱动器上。每个数据块随后根据所采用的 RAID 级别被写入到 RAID 阵列中的硬盘驱动器上。当读取数据时，这个过程是相反的，给人的错觉就是阵列中的多个驱动器实际上是一个大的驱动器。

15.2 谁应该使用 RAID?

系统管理员和管理大量数据的其他人将受益于 RAID 技术的使用。部署 RAID 的主要理由包括：

1） 提高速度

2） 使用一个单一虚拟磁盘，提高存储容量

3） 降低磁盘故障的几率

15.3 RAID 类型

有三种可能的 RAID 方法： 固件的 RAID，硬件 RAID 和软件 RAID。

固件 RAID

固件 RAID（又名作为 ATARAID）是一种类型的软件 RAID，其中 RAID 集可以使用一个基于固件的菜单来进行配置。这种类型的 RAID 中使用的固件也被植入 BIOS，允许您从 RAID 集进行启动。不同的厂商使用不同的盘上元数据格式以标记 RAID 集组件。英特尔的 Matrix RAID 是固件 RAID 系统的一个很好

的例子。

硬件 RAID

基于硬件的阵列独立地从主机管理 RAID 子系统。它为每个主机 RAID 阵列提出了一个单一的磁盘。

硬件 RAID 设备可能在系统内部或外部，内部设备通常包括一个可以透明地处理该操作系统 RAID 任务的专门的控制器卡，而外部设备通常通过 SCSI，光纤通道，iSCSI， InfiniBand 或其他高速网络互连连接到系统，并提出系统的逻辑卷。

RAID 控制器卡的功能类似于操作系统的一个 SCSI 控制器，并处理所有实际的驱动器通信。用户将驱动器插入 RAID 控制器中（就像一个正常的 SCSI 控制器），然后将它们添加到 RAID 控制器配置中，操作系统不会知道其中的差别。

软件 RAID

软件 RAID 实现内核磁盘（块设备）代码中的各种 RAID 级别。它提供最便宜可行的解决方案，因为它并没有要求昂贵的磁盘控制器卡或热插拔机箱[1]。软件 RAID 也可以用便宜的 IDE 磁盘以及 SCSI 磁盘。随着今天速度更快的 CPU 的诞生，软件 RAID 的性能普遍优于硬件 RAID。

Linux 内核包含一个多重磁盘（MD）驱动程序，允许完全独立于硬件的 RAID 解决方案。一种基于软件的阵列的性能取决于服务器的 CPU 性能和负载。

下面是 Linux 软件 RAID 栈的一些主要特点：

1） 多线程设计

2） 无须重建条件下，Linux 机器之间阵列的可移植性

3） 利用空闲的系统资源，重建后台阵列

4） 热插拔驱动器支持

5） 利用某些 CPU 功能优势如单指令多数据流扩展支持（streaming SIMD support）来进行 CPU 检测

6） 阵列中的磁盘上的坏扇区的自动校正

7） 定期进行 RAID 数据一致性检查，以确保阵列的健康

8） 通过发送到重要事件指定的邮件地址上的邮件警报来实施对阵列的主动监控

9） 通过允许内核确切地知道磁盘的哪些分区需要重新同步，而不是将整个

阵列都重新同步，以写入为目的的位图将大大加快重新同步事件（Resync events）。

10） 重新同步检查点。因此，如果您在重新同步期间重新启动您的计算机，在启动时重新同步将从它中断的地方开始恢复，而不是从头再来

11） 在安装后能够更改阵列的参数。例如，当您有一个新的磁盘要添加时，您可以把 4 磁盘 RAID5 阵列扩展到 5 磁盘 RAID5 阵列。这种扩展操作在现场完成，并不需要重新安装到新的阵列上。

15.4 RAID 级别和线性支持

RAID 支持各种配置，包括 0 级，1 级，4 级，5 级，6 级，10 级和线性。RAID 类型定义如下：

0 级：

0 级 RAID 通常被称为“条带化”，是一种以性能为导向的条带化数据映射技术。这意味着被写入到阵列中的数据将被分解成条状并写入到构成该阵列的所有磁盘上，允许在不提供冗余的情况下，以较低的固有成本实现较高的 I / O 性能。

许多 0 级 RAID 的执行只会将整个成员设备上的数据条带化为阵列中最小设备的大小。这意味着，如果您有多台尺寸略有不同的设备，每台设备将得到处理，就好像它与最小的驱动器的大小相同。因此，0 级阵列的一般存储容量等于硬件 RAID 中最小成员磁盘的容量或者是软件 RAID 中最小成员分区的容量乘以阵列中的磁盘或分区数量。

1 级：

1 级 RAID，或“镜像”， 比任何其他形式的 RAID 使用的时间都长。1 级 RAID 通过将相同的数据写入到阵列中的每个成员磁盘上来提供冗余，使得每个磁盘上有一个“镜像”副本。由于其简单性和高数据可用性，镜像仍然很流行。1 级 RAID 采用两个或多个磁盘进行操作，提供了非常好的数据可靠性而且提高了读取密集型应用方面的性能，但成本相对较高。[2]

1 级阵列的存储容量等于硬件 RAID 中最小镜像硬盘的容量或软件 RAID 中最小镜像分区的容量。1 级冗余是所有 RAID 类型中最高的，该阵列能够在只有一个单一的磁盘出现的情况下进行操作。

4 级：

4 级 RAID 使用集中在一个单一的磁盘驱动器上的奇偶校验 [3] 来保护数据。因为专用的奇偶校验磁盘代表所有到 RAID 阵列的写入处理上的一个固有瓶颈，如果没有附带的技术，如回写高速缓存，或在系统管理员考虑到这个瓶颈来故意设计软件 RAID 设备的情况下（如一个阵列，一旦为数据所填充，该阵列将很难获得多少写入处理）。4 级 RAID 使用得如此之少以至于在 Anaconda 中它不是一个可用的选项。但是，如果真正需要的话，可以由用户手动创建。

4 级硬件 RAID 的存储容量等于最小成员分区的容量乘以分区数减一。4 级 RAID 阵列的性能永远是不对称的，这意味着读取的表现将优于写入。这是因为写入生成校验时消耗了额外的 CPU 和主内存带宽，然后在将实际数据写入到磁盘时，还要消耗额外的总线带宽，因为你正在写入的不仅是数据，还有奇偶校验值。读取操作只需要读取数据，而不是奇偶校验值，除非阵列处于降级状态。因此，对于正常运行条件下相同数量的数据传输，读取产生更少的到驱动器和跨计算机总线的流量。

5 级：

这是最为普通的 RAID 类型。通过在阵列的所有成员磁盘驱动器上分布奇偶校验值，5 级 RAID 取消了 4 级固有的瓶颈。唯一的性能瓶颈是奇偶校验计算过程本身。随着现代 CPU 和软件 RAID 的诞生，这通常根本算不上是一个瓶颈，因为现代的 CPU 可以非常快速地生成校验。但是，如果在软件 RAID5 的阵列中，您有足够大量的成员设备以保证合并的数据集在所有设备上的传输速度足够高，那么这个瓶颈就开始发挥作用。

至于 4 级，5 级 RAID 具有不对称的性能，其读取性能大大超越写入。5 级 RAID 的存储容量与 4 级的计算方式相同。

6 级：

这是一种数据冗余和保存时常见的 RAID 等级，而性能并不是最重要的， 1 级 RAID 的低空间效率是不能接受的。6 级 RAID 使用一个复杂的校验计划以便能够恢复阵列中任何两个驱动器的损失。这种复杂的校验计划在软件 RAID 设备上创建了一个相当高的 CPU 负荷，还在写入处理期间施加了增加的负荷。因此，不仅是 5 级和 4 级，6 级也同样在性能上是不对称的，而且它相对来说极其不对称。

6 级 RAID 阵列的总容量计算方式类似于 4 级和 5 级 RAID，除了您必须从额外的校验存储空间的器件数量中减去 2 台设备（而不是 1 台）。

10 级：

10 级 RAID 尝试将 0 级的性能优势与 1 级的冗余结合起来。它还有助于在 2 个以上设备的情况下减少一些在 1 级阵列上浪费的空间。使用 10 级 RAID 情况下，可以创建一个 3 驱动器阵列进行配置以便为每一条数据仅存储 2 个副本，然后使整个阵列的大小变为最小设备的 1.5 倍，而不是只等于最小设备的规模（就像是用一个 3 驱动设备，1 级阵列）。

创建 10 级阵列时的可用选项数目（以及为某一特定用途选择正确选项的复杂性）使得无法在安装期间完成创建。使用命令行 mdadm 工具以手动方式创建一个也是可能的。关于选项和各自性能权衡的详细信息，请参阅 man md。

线性 RAID

线性 RAID 是驱动器的简单分组以便创建一个更大的虚拟驱动器。在线性 RAID 中，数据块以顺序方式从一个成员驱动器开始分配，只有当第一个被完全填充后才会到下一个驱动器。这种分组没有提供性能上的益处，因为任何 I／O 操作在成员驱动器之间分配是不可能的。线性 RAID 也没有提供冗余，而事实上，这样降低了可靠性 - 如果任何一个成员驱动器发生故障，整个阵列将无法使用。容量是所有成员磁盘容量的总和。

15.5 Linux RAID 子系统

Linux 中的 RAID 是由以下子系统组成的：

Linux 硬件 RAID 控制器驱动程序

硬件 RAID 控制器在 Linux 中没有特定的 RAID 子系统。因为他们使用特殊的 RAID 芯片组，硬件 RAID 控制器有自己的驱动程序，这些驱动程序允许系统检测作为常规磁盘的 RAID 集。

mdraid

mdraid 子系统被设计成为一个 Linux 的软件 RAID 解决方案，它也是 Linux 下软件 RAID 的首选解决方案。该子系统使用自己的元数据格式，一般简称为本地 mdraid 元数据。

mdraid 还支持其他元数据格式，被称为外部元数据。中标麒麟可信操作系统 V6.0V6.0 在外部元数据环境下使用 mdraid 来访问 ISW / IMSM 集（英特尔固件

RAID）。mdraid 集通过 mdadm 实用工具来进行配置和控制。

dmraid

设备映射 RAID 或 dmraid 指的是 device-mapper 的内核代码，提供一种机制将磁盘拼凑在一起整合到 RAID 集中。这个相同的内核代码不提供任何 RAID 配置机制。

dmraid 被整体配置到用户空间，使得很容易地支持各种盘上的元数据格式。因此，dmraid 用在各种各样的固件 RAID 执行中。dmraid 软件包还支持英特尔固件 RAID，尽管中标麒麟可信操作系统 V6.0V6.0 使用 mdraid 来访问英特尔固件 RAID 集。

15.6 1 安装程序中的 RAID 支持

Anaconda 安装程序会自动检测系统上的任何硬件和固件 RAID 集，使它们可用于安装。Anaconda 也支持使用 mdraid 的软件 RAID，并可以识别现有 mdraid 集。

Anaconda 提供了用于在安装过程中创建 RAID 集的实用工具，但这些工具仅允许分区（而不是整个磁盘）成为新集的成员。要为一个集使用整个磁盘，只需在其上创建一个跨越整个磁盘的分区，并将该分区作为 RAID 集的成员。

当 root 文件系统使用 RAID 集时，Anaconda 会添加特殊的内核命令行选项到 bootloader 配置，告诉 initrd 在搜寻 root 文件系统之前需要激活哪个 RAID 集。

在安装过程中配置 RAID 的说明，请参阅中标麒麟可信操作系统 V6.0V6.0 安装指南。

15.7 配置 RAID 集

大多数 RAID 集在创建过程中被配置，通常是通过固件菜单或从安装程序中进行。在某些情况下，您可能需要在安装系统后创建或修改 RAID 集，而无需重新启动机器，进入固件菜单，这样做效果最佳。

有些硬件 RAID 控制器允许您配置邻近备用设备 RAID 集或者甚至增加额外的磁盘后完全定义新集。因为在这方面没有标准的 API，这就要求使用驱动程序特定的实用工具。这方面的资料请参阅您的硬件 RAID 控制器的驱动程序文档。

mdadm

mdadm 命令行工具用于管理 Linux 软件中的 RAID，即 mdraid。关于不同

mdadm 模式和选项的信息，请参考 man mdadm。手册页还包含了常见的操作，如创建，监控和装配软件 RAID 阵列的有用的例子。

dmraid

顾名思义，dmraid 用于管理 device – mapper RAID 集。dmraid 工具发现 ATARAID 设备使用多个元数据格式的处理程序，每个程序都支持多种格式。对于所支持格式的完整列表，运行 dmraid -l

正如前面在第 15.5 节，“Linux RAID 子系统”所提到的，dmraid 工具不能在创建后配置 RAID 集。关于使用 dmraid 的更多信息，请参阅 man dmraid。

15.8 高级 RAID 设备创建

在某些情况下，您可能想要在一个安装完成后无法创建的阵列上安装该操作系统。通常，这意味着在复杂的 RAID 设备上设置/ boot 或根文件系统阵列，在这种情况下，您可能需要使用 Anaconda 所不支持的阵列选项。为了解决这个问题，请执行以下步骤：

1） 像往常一样插入安装盘。

2） 在最初的启动阶段，请选择救援模式，而不是安装或升级。当系统完全进入救援模式启动时，用户将会被派送一个命令行终端。

3） 从该终端，使用 parted 在目标硬盘驱动器上来创建 RAID 分区。然后，使用 mdadm 从使用任何及所有可用设置和选项的分区手动创建 RAID 阵列。欲了解更多有关如何执行这些操作的信息，请参阅第 5 章“分区”， man parted 和 man mdadm。

4） 一旦创建了阵列，您也可以有选择地在阵列上创建文件系统。中标麒麟可信操作系统 V6.0V6.0 支持的文件系统的基本技术信息，请参阅第 2. 2 节 “支持的文件系统概述”。

5） 重新启动计算机，这时候选择安装或升级（Install 或 Upgrade）像往常一样进行安装。当 Anaconda 搜索系统中的磁盘时，它会发现既存的 RAID 设备。

6） 当问到有关如何使用系统中的磁盘时，选择自定义布局（Custom Layout ），然后单击下一步。在设备列表中，既有的 MD RAID 设备将陆续被列出。

7） 选择一个 RAID 设备，单击“编辑”，并配置其挂载点以及（可选）它应

该使用的文件系统类型（如果你没有创造一个早期版本），然后单击“完成”。Anaconda 将执行到这个既存的 RAID 设备的安装，当您将其创建到救援模式中时，请保持您选择的自定义选项。

备注：安装程序的有限的救援模式，不包括手册页。man mdadm 和 man md 都包含用于创建自定义 RAID 阵列的有用信息，并且可能整个工作区都需要 man mdadm 和 man md 。因此，它可能有助于在手册页可用的情况下访问一台机器或在启动进入救援模式下和创建自定义的阵列之前把它们打印出来。

16. 交换空间

16.1 什么是交换空间？

物理内存（RAM）已满时使用 Linux 中的交换空间。如果系统需要更多的内存资源而且 RAM 已满，内存中的无效页面被移动到交换空间。虽然交换空间可以帮助计算机缓解少量的 RAM，但它不应该被视为可以置换 RAM。交换空间位于硬盘驱动器，比物理内存的存取时间略慢。

交换空间可以是一个专用的交换分区（推荐），交换文件，或交换分区和交换文件的组合。

当物理内存小于 2G 时，swap 大小为物理内存的 2 倍。超过 2G 的部分，swap 大小跟物理内存相等，但从来不应少于 32 MB。

因此，如果

M = GB 级 RAM 数量， S= GB 级 swap 大小，那么

```
If M < 2
S = M *2
Else
S = M + 2
```

利用这个公式，2G 物理内存需要 4G swap，而 3G 物理内存则需要 5G swap。如果你打算添加内存的话，创建一个大些的 swap 区是很有帮助的。

对于物理内存确实较大的系统（超过 32G），使用小于等于物理内存容量的 swap 区会更好(约 1 倍，或更少的物理 RAM)。

提示：文件系统和分配为交换空间的 LVM2 卷在使用时不应该被修改。如果一个系统进程或内核正在使用交换空间，任何修改交换空间的尝试都将失败。使用 free 和 cat /proc/swaps 命令来验证正在使用中的 swap 的大小和位置。

我们建议您在该系统的救援模式下启动时修改交换空间，就如何在救援模式

下启动的说明，请参阅安装指南。当提示挂载文件系统时，请选择“跳过”。

16.2 添加交换空间

有时需要在安装后添加更多的交换空间。例如，您可以在您的系统将 RAM 的大小从 128 MB 升级到 256 MB，但目前只有 256 MB 的交换空间。如果你执行内存密集的操作或运行需要大量内存的应用程序，将交换空间的大小增加到 512 MB 可能是有利的。

您有三种选择： 创建一个新的交换分区，创建一个新的交换文件，或在现有的 LVM2 逻辑卷上扩展交换分区。建议您扩展现有的逻辑卷。

16.2.1 在 LVM2 逻辑卷上扩展交换

默认情况下，中标麒麟可信操作系统 V6.0V6.0 在安装过程中使用所有可用空间。如果您的系统属于这种情况，那么您必须首先添加一个新的物理卷到交换空间使用的卷组。关于如何进行此操作的说明，请参见 4. 2. 2 节，“添加未分配的卷到卷组”。

添加额外的存储到交换空间的卷组后，可以把它扩展。要做到这一点，请执行以下步骤（假设/dev/VolGroup00/LogVol01 是你想扩展 256MB 的卷）：

1） 禁用交换相关的逻辑卷：

```
swapoff -v /dev/VolGroup00/LogVol01
```

2） 调整 LVM2 逻辑卷为增加 256 MB：

```
lvresize /dev/VolGroup00/LogVol01 -L +256M
```

3） 格式化新的交换空间

```
mkswap /dev/VolGroup00/LogVol01
```

4） 禁用扩展的逻辑卷

```
swapon -v /dev/VolGroup00/LogVol01
```

为了测试逻辑卷是否被成功地扩展，请使用 cat /proc/swaps 或 free 来检查交换空间。

16.2.2 为 Swap 创建 LVM2 逻辑卷

要添加一个交换卷组（假设/dev/VolGroup00/LogVol02 是你想添加的交换卷）：

1） 创建大小为 256 MB 的 LVM2 逻辑卷

```
lvcreate VolGroup00 -n LogVol02 -L 256M
```

2） 格式化新的交换空间

```
mkswap /dev/VolGroup00/LogVol02
```

3） 添加以下条目到/etc/fstab 文件：

```
/dev/VolGroup00/LogVol02 swap swap defaults 0 0
```

4） 禁用扩展的逻辑卷

```
swapon -v /dev/VolGroup00/LogVol02
```

为了测试逻辑卷是否被成功地扩展，使用 cat /proc/swaps 或 free 来检查交换空间。

16.2.3 创建交换文件

添加一个交换文件：

1） 确定以兆字节为单位的新的交换文件的大小并乘以 1024 以确定块的数量。例如，64 MB 的交换文件的块大小为 65536。

2） 在 shell 提示下以 root 用户身份键入以下命令，计数与所需的块大小相等：

```
dd if=/dev/zero of=/swapfile bs=1024 count=65536
```

3） 用此命令设置交换文件：

```
mkswap /swapfile
```

4） 要在开机时立即但不是自动地启用交换文件：

```
swapon /swapfile
```

5） 要在开机时启用，编辑/etc/fstab 来包括以下条目：

```
/swapfile swap swap defaults 0 0
```

下一次系统启动时，它启用新的交换文件。

为了测试逻辑卷是否被成功地扩展，使用 cat /proc/swaps 或 free 来检查交换空间。

16.3 删除交换空间

在安装后减少交换空间也须审慎。例如，您可以在您的系统将 RAM 的大小从 1 GB 降到 512 MB，但分配的交换空间仍是 2 GB。把交换空间的大小减少到 1 GB 可能是有利的，因为较大 2 GB 可能浪费磁盘空间。

您有三种选择： 删除整个用于交换的 LVM2 逻辑卷，删除交换文件，或减少现有的 LVM2 逻辑卷上的交换空间。

16.3.1 减少 LVM2 逻辑卷上的交换空间

要减少 LVM2 交换逻辑卷（假设/dev/VolGroup00/LogVol01 是您想要减少的

卷）：

1） 禁用交换相关的逻辑卷：

```
swapoff -v /dev/VolGroup00/LogVol01
```

2） 将 LVM2 逻辑卷减少 512 MB：

```
lvreduce /dev/VolGroup00/LogVol01 -L -512M
```

3） 格式化新的交换空间

```
mkswap /dev/VolGroup00/LogVol01
```

4） 禁用扩展的逻辑卷

```
swapon -v /dev/VolGroup00/LogVol01
```

为了测试逻辑卷是否被成功地缩减，使用 cat /proc/swaps 或 free 来检查交换空间。

16.3.2 删除用于交换的 LVM2 逻辑卷

要删除一个交换卷组（假设/dev/VolGroup00/LogVol02 是你想删除的交换卷）：

1） 禁用交换相关的逻辑卷：

```
swapoff -v /dev/VolGroup00/LogVol02
```

2） 删除大小为 512 MB 的 LVM2 逻辑卷

```
lvremove /dev/VolGroup00/LogVol02
```

3） 从/etc/fstab 文件中删除以下条目：

```
/dev/VolGroup00/LogVol02 swap swap defaults 0 0
```

为了测试逻辑卷是否被成功地删除，使用 cat /proc/swaps 或 free 来检查交换空间。

16.3.3 删除交换文件

删除一个交换文件：

1） 在 shell 提示下以 root 用户身份执行以下命令来禁用交换文件（其中/swapfile 是交换文件）：

```
swapoff -v /swapfile
```

2） 从/etc/fstab 文件中删除其条目：

3） 删除实际文件：

```
rm /swapfile
```

16.4 移动交换空间

为了将交换空间从一个位置移动到另一个位置，请按照以下步骤删除交换空间，然后按步骤添加交换空间。

17. 磁盘定额

用户消耗了太多的磁盘空间或分区已满之前，通过实施提醒系统管理员的磁盘定额来限制磁盘空间的大小。

磁盘定额可以根据个人用户以及用户群体进行配置。这使得它可以分别管理分配给用户特有文件（如电子邮件）的空间和分配到用户所执行项目上的空间（假定项目有自己的群组）。

此外，定额的设置不仅要控制消耗的磁盘块数，还要控制索引节点数（包含 UNIX 文件系统中的文件信息的数据结构）。因为索引节点用于包含文件相关的信息，这使得可创建文件的数量得到控制。

必须安装 quota RPM 以实现磁盘定额。

17.1 配置磁盘定额

要实现磁盘定额，使用以下步骤：

1） 通过修改/etc/fstab 文件，启用每个文件系统的定额。

2） 重新挂载文件系统。

3） 创建定额数据库文件，并生成磁盘使用情况表。

4） 分配定额的政策。

下面的章节中详细讨论了这些步骤中的每一步。

17.1.1 启用定额

以 root 身份，使用文本编辑器，编辑/etc/fstab 文件。添加 usrquota 和/或用 grpquota 选项到需要定额的文件系统中：

```
/dev/VolGroup00/LogVol00 /      ext3 defaults              1  1
LABEL=/boot /boot       ext3 defaults              1  2
none/dev/pts  devpts    gid=5,mode=620   0  0
none/dev/shm       tmpfs      defaults             0  0
none/proc      proc       defaults             0  0
none/sys sysfs      defaults             0  0
/dev/VolGroup00/LogVol02  /home     ext3 defaults,usrquota,grpquota 1 2
/dev/VolGroup00/LogVol01  swap      swap       defaults             0  0 . . .
```

在这个例子中，/home 文件系统启用用户和组定额。

备注：下面的例子假设一个单独的/home 分区是在中标麒麟可信操作系统 V6.0 的安装过程中创建的。root (/)分区可于在/ etc / fstab 文件中设置定额政策。

17.1.2 重新安装文件系统

添加 usrquota 和/或 grpquota 选项后，重新挂载 fstab 条目到已被修改的每个文件系统。如果文件系统没有在任何进程中使用，请使用下列方法之一：

1） umount 命令后紧接着发出 mount 命令来重新挂载文件系统。挂载和卸载各种文件系统类型的 umount 和 mount 的特定语法，请参阅手册页。

2） 发出 mount -o remount file-system 命令（其中文件系统的名称是 file-system）以重新挂载文件系统。例如， 重新挂载/home 文件系统，需发出的命令是 mount -o remount /home。

如果该文件系统目前正在使用中，重新挂载文件系统的最简单的方法是重新启动系统。

17.1.3 创建定额数据库文件

每个启用了定额的文件系统重新安装后，该系统可以使用磁盘定额来工作。然而，文件系统本身尚未准备支持定额。下一步是运行 quotacheck 命令。

quotacheck 命令检查启用了定额的文件系统，并建立每个文件系统的当前磁盘使用情况表。然后此表被用来更新操作系统的磁盘使用情况的副本。此外，文件系统的磁盘定额文件被更新。

要创建文件系统上的定额文件（aquota.user 和 aquota.group），使用 quotacheck 命令的- c 选项。例如，如果启用用户和组定额用于/ home 文件系统，在/ home 目录中创建该文件：

```
quotacheck -cug /home
```

- c 选项指明应该为每个启用定额的文件系统创建定额文件，- u 选项指定检查用户定额，- g 选项指定检查组定额。

如果- u 或- g 选项都没有被指定，只需创建用户定额文件。如果只有-g 选项被指定，只需创建组定额文件。

创建文件后，运行以下命令生成启用定额的每个文件系统的当前磁盘使用情况表：

```
quotacheck -avug
```

使用的选项如下：

a

检查所有启用定额的，本地挂载的文件系统

v

定额检查进行时显示详细的状态信息

u

检查用户的磁盘定额信息

g

检查群组的磁盘定额信息

quotacheck 运行完毕后，将每个启用定额的本地挂载的文件系统如/ home 的数据填充到与启用的定额（用户和/或组）相对应的定额文件中。

17.1.4 为每个用户分配定额

最后一步是用 edquota 命令来分配磁盘定额。

要为用户配置定额，在 shell 提示下以 root 用户身份执行命令：

```
edquota username
```

为每个需要定额的用户执行此步骤。例如，如果定额是在/home 分区的/etc/fstab 目录中（下例中的/dev/VolGroup00/LogVol02）被启用并且执行 edquota testuser 命令，以下信息显示在系统默认配置的编辑器中：

```
Disk quotas for user testuser (uid 501):
Filesystem                   blocks  soft      hard    inodes  soft  hard
/dev/VolGroup00/LogVol02     440436     0         0     37418     0     0
```

备注：edquota 使用 EDITOR 环境变量定义的文本编辑器。要改变编辑器，把您的~/.bash_profile 文件中 EDITOR 环境变量设置为您选择的编辑器的完整路径。

第一栏是为它启用了定额的文件系统的名称。第二栏显示用户当前正在使用多少个数据块。接下来的两栏用来设置文件系统上用户的软，硬数据块限度。inodes 栏显示用户当前正在使用多少个索引节点。最后的两栏用来设置文件系统上用户的软、硬索引节点限度。

硬数据块限制是一个用户或组可以使用的磁盘空间的绝对最高额度。一旦达到此限额，将没有进一步的磁盘空间可以使用。

软数据块限额定义了可以使用的最大磁盘空间量。然而，不同于硬数据块限额，在某一时间段软数据块限额可以被超出。这个时间被称为宽限期。宽限期可

由秒，分钟，小时，天，周，或月来表示。

如果任何值都设置为 0，即没有设置限制。在文本编辑器中，更改所需的限制。例如：

```
Disk quotas for user testuser (uid 501):
Filesystem                    blocks    soft    hard   inodes    soft
hard
/dev/VolGroup00/LogVol02   440436   500000   550000    37418       0
0
```

为了验证用户的定额已被设置，使用命令：

```
quota testuser
```

17.1.5 为每一组分配定额

定额也可以按组来进行分配。例如，为 devel 组（设置组定额前组必须存在）设置组定额，使用命令：

```
edquota -g devel
```

此命令将显示文本编辑器中该组的现有定额：

```
Disk quotas for group devel (gid 505):
Filesystem      blocks     soft  hard inodes     soft  hard
/dev/VolGroup00/LogVol02     440400 0      0      37418      0      0
```

修改限制，然后保存该文件。

为了验证组的定额已设置，使用命令：

```
quota -g devel
```

17.1.6 为软限制设置宽限期

如果一个给定的定额有软限制，你可以用下面的命令编辑宽限期（即软限制可以被超出的时间量）：

```
edquota -t
```

此命令对用户或组的索引节点或块的定额起作用。其它 edquota 命令在一个特定的用户或组的定额上运行，-t 选项只在每个启用定额的文件系统上生效。

17.2 管理磁盘定额

如果实施定额，它们需要一些维护 – 主要目的是看是否超出定额，并确保定额准确。

当然，如果用户多次超过其定额或始终如一地达到他们的软限制，一个系统管理员有几种选择取决于什么类型的用户，以及有多少磁盘空间影响他们的工作。管理员可以帮助用户确定如何使用更少的磁盘空间，或增加用户的磁盘定额。

17.2.1 启用和禁用

在不将它们设置为 0 的情况下，也可以禁用定额。要关闭所有用户和组定额，请使用以下命令：

```
quotaoff -vaug
```

如果- u 或- g 选项都没有被指定，只需禁用用户定额文件。如果只有-g 是指定的，只需禁用组定额。-v 开关用详细状态信息显示命令执行情况。

若要再次启用定额，请使用带有相同选项的 quotaon 命令。

例如，若要启用所有文件系统的用户和组定额，请使用以下命令：

```
quotaon -vaug
```

若要启用一个特定的文件系统的定额，如/ home，请使用以下命令：

```
quotaon -vug/home
```

如果- u 或- g 选项都没有被指定，只需启用用户定额。如果只有-g 是指定的，只需启用组定额。

17.2.2 磁盘定额报告

创建磁盘使用情况报告需要运行 repquota 实用程序。例如命令 repquota /home 产生这样的输出：

```
*** Report for user quotas on device /dev/mapper/VolGroup00-LogVol02
Block grace time: 7days;  Inode grace time: 7days
Block limits File limits
User used soft hard grace used soft hard grace
----------------------------------------------------------------------------------------------------
---
root --  36  0  0  4  0
kristin  --  540 0  0  125 0
testuser  --  440400  500000  550000  37418  0
0
0
0
```

要查看所有启用定额的文件系统的磁盘使用情况报告（- **a** 选项），请使用命令：

```
repquota -a
```

虽然该报告便于阅读，但有几点应加以解释。每个用户后面显示的-是判断数据块或索引节点限制是否已被超出的一个快捷方法。无论软限制是否被超出，a +出现在相应-的位置，第一个 - 代表块限制，第二个 - 代表索引节点限制。

宽限期栏通常是空白的。如果已超过软限制，该栏包含与宽限期剩余的时间

段相等的一个时间规范。如果宽限期已过期，在这个位置上什么符号都不出现。

17.2.3 保持定额精确

当一个文件系统没有卸载干净（例如，由于系统崩溃）时，有必要运行 quotacheck。然而，即使系统没有崩溃，也可以定期运行 quotacheck。定期安全运行 quotacheck 的方法包括：

确保在下次重新启动时执行 quotacheck。

 提示：对于大多数系统的最佳方法

对于定期重新启动的（繁忙）多用户系统，此方法效果最佳。

以 root 身份，将一个 shell 脚本放到/etc/cron.daily/ 或 /etc/cron.weekly/目录中，或安排使用包含 touch /forcequotacheck 命令的 crontab -e 命令。这将在根目录中创建一个空的 forcequotacheck 文件，系统的 init 脚本在启动时寻找该文件。如果发现 init 脚本正在运行 quotacheck。随后，init 脚本会删除/forcequotacheck 文件，因此，安排用 cron 定期创建这个文件确保在下次重新启动时运行 quotacheck。

关于 cron 的更多信息，请参阅 man cron。

在单用户模式下运行 quotacheck

另一种安全运行 quotacheck 的方式是（重新）启动系统进入到单用户模式以防止定额文件中的数据遭到损坏，并运行以下命令：

```
quotaoff -vaug /file_system
quotacheck -vaug /file_system
quotaon -vaug /file_system
```

在运行中的系统上运行 quotacheck

如果有必要，在没有用户登录期间在一台机器上运行 quotacheck 也是可能的，因此文件系统上没有已打开的文件正在接受检查。运行命令 quotacheck -vaug file_system，如果执行 quotacheck 不能以只读方式重新挂载给定的 file_system，此命令将失败。请注意，根据检查结果，文件系统将被以读写方式重新挂载。

 警告：由于定额文件存在遭到损坏的可能性，我们并不不建议在一个以读写方式挂载的 live 文件系统上运行 quotacheck。

关于配置 cron 的更多信息，请参阅 man cron。

17.3 参考

磁盘定额的详细信息，请参阅以下命令的手册页：

quotacheck

edquota

repquota

quota

quotaon

quotaoff

18. 访问控制列表

文件和目录拥有的权限集适用于文件的所有者，与该文件关联的组以及系统的所有其他用户。然而，这些权限集有其局限性。例如，不同的权限不能配置给不同的用户。因此，访问控制列表（ACL）可以得到实施。

中标麒麟可信操作系统 V6.0 内核为 ext3 文件系统和 NFS 导出的文件系统提供 ACL 支持。通过 Samba 访问的 ext3 文件系统也支持 ACL。

随着内核的支持，acl 软件包需要实施 ACL。它包含用于添加、修改、删除和检索 ACL 信息的工具。

cp 和 mv 命令复制或移动与文件和目录相关的任何 ACL。

18.1 安装文件系统

在使用文件或目录的 ACL 时，文件或目录的分区必须挂载 ACL 支持。如果它是一个本地的 ext3 文件系统，可以用下面的命令安装：

```
mount -t ext3 -o acl device-name partition
```

例如：

```
mount -t ext3 -o acl /dev/VolGroup00/LogVol02 /work
```

另外，如果分区在 /etc/fstab 文件中列出，分区的条目可以包括 acl 选项：

```
LABEL=/work       /work       ext3 acl   1      2
```

如果一个 ext3 文件系统是通过 Samba 访问的，并且 ACL 已被用于该文件，ACL 能够被识别，因为 Samba 已经用-with-acl-support 选项进行了编译。访问或挂载 Samba 共享时，不需要进行特殊的标记。

18.1.1 NFS

默认情况下，如果正被 NFS 服务器导出的文件系统支持 ACL 并且 NFS 客

户端能够读取 ACL，ACL 将为客户端系统所利用。

要在配置服务器时禁用 NFS 共享上的 ACL，需要包括/ etc/ exports 文件中的 no_acl 选项。若要在客户端上安装 ACL 时禁用 NFS 共享上的 ACL，可以通过命令行或 /etc/fstab file 文件用 no_acl 选项完成对它的安装。

18.2 设置访问 ACLs

有两种类型的 ACLs： 访问 ACL 和默认 ACL。访问 ACL 是一个特定的文件或目录的访问控制列表。默认 ACL 只能与目录相关联，如果目录内的文件没有存取 ACL，它使用目录的默认 ACL 规则。默认 ACL 是可选的。

ACL 可配置如下：

1） 每个用户

2） 每个组

3） 通过有效权限掩码

4） 适合于未列在文件用户组中的用户

setfacl 实用工具为文件和目录设置 ACL。使用- m 选项，添加或修改文件或目录的 ACL：

```
setfacl -m rules files
```

以下格式必须指定规则(rules)。如果他们是由逗号分隔开的， 在同一个命令中可以指定多个规则。

u:uid:perms

为用户设置存取 ACL。可以指定用户名或 UID。该用户可以是系统上的任何有效用户。

g:gid:perms

为一个组设置存取 ACL。可以指定组名或 GID。该组可以是系统上的任何有效组。

m:perms

设置有效权限掩码。掩码是所属组和所有的用户和组条目的所有权限的并集。

o:perms

为文件组中用户之外的用户设置存取 ACL。

权限 (perms)必须结合用于读，写和执行的字符 r、w 和 x。

如果一个文件或目录已经有一个 ACL，而且 setfacl 命令已被使用，额外的规则会被添加到现有的 ACL 或现有的规则将被修改。

例如，为了给用户 andrius 读写权限：

```
setfacl -m u:andrius:rw /project/somefile
```

若要删除用户，组或其他人的所有的权限，请使用- x 选项，并且不指定任何权限：

```
setfacl -x rules files
```

例如，从 UID 为 500 的用户中删除所有权限：

```
setfacl -x u:500 /project/somefile
```

18.3 1 设置默认的 ACL

要设置一个默认 ACL，请在规则之前添加 d: ，并指定一个目录，而不是一个文件名。

例如，对于不在用户组中的用户，设置 /share/ 目录默认的 ACL 来读取和执行（个别文件的访问 ACL 可以覆盖它）：

```
setfacl -m d:o:rx /share
```

18.4 检索 ACL

要确定一个文件或目录的现有 ACL，使用 getfacl 命令。在下面的例子中，getfacl 被用来确定一个文件的现有 ACL。

```
getfacl home/john/picture.png
```

上述命令将返回下面的输出信息：

```
# file: home/john/picture.png
# owner: john
# group: john
user::rw-
group::r--
other::r--
```

如果指定一个默认 ACL 的目录，默认的 ACL 也会被显示，如下图所示。例如，getfacl home/sales/ 将显示类似的输出：

```
# file: home/sales/
# owner: john
# group: john
user::rw-
user:barryg:r--
group::r--
```

```
mask::r--
other::r--
default:user::rwx
default:user:john:rwx
default:group::r-x
default:mask::rwx
default:other::r-x
```

18.5 用 ACL 归档文件系统

默认情况下，现在 dump 命令在备份操作期间会保存 ACL。使用 tar 归档文件或文件系统时，请使用 --acls 选项来保存 ACL。同样，当使用 cp 来复制带有 ACL 的文件时，须包括 --preserve=mode 选项，以确保 ACL 被复制过来。此外，cp 的- a 选项（相当于-dR --preserve=all）在备份过程中还保留 ACL，伴随着时间戳，SELinux 上下文，以及类似的其他信息的。有关 dump、tar 或 cp 的更多信息，请参阅各自的手册页。

Star 实用程序类似于 tar 实用程序，它可以被用来生成文件档案， 然而，它的一些选项是有区别的。常用选项列表，请参见表 18.1star 的命令行选项 。对于所有可用选项，请参见 man star。star 软件包需要使用此实用程序。

表 18-1 star 的命令行选项

选项	描述
-c	创建一个归档文件
-n	不要解压缩文件， 配合-x 使用以表明文件解压缩进。程。
-r	替换归档文件。文件被写入到归档文件的结尾处，取代具有相同路径和文件名的任何文件。
-t	显示归档文件的内容。
-u	更新存档文件。如果他们在归档中不存在，或如果这些文件比在存档中的同名文件还要新，这些文件将被写入到归档结尾处。如果存档是一个可以退格的文件或一个未锁定的磁带，此选项才会适用。
-x	从归档中解压缩文件， 如果与-U 选项一起使用并且归档中的文件比文件系统上相应的文件还要旧，该文件不会被解压缩。

-help	显示最重要的选项。
-xhelp	显示最不重要的选项。
-/	从归档中解压缩文件时，不要去掉文件名中开头的斜线。默认情况下，解压缩文件时，开头斜线被去掉。
-acl	当创建或解压缩时，存档或恢复任何与文件和目录相关的 ACL。

18.6 与旧系统的兼容性

如果 ACL 已设置在给定文件系统的任何一个文件上，该文件系统便会有 ext_attr 属性。要查看这个属性，使用下面的命令：

```
tune2fs -l filesystem-device
```

已经获得 ext_attr 属性的文件系统可以使用老版本的内核来进行挂载，但这些内核并不执行任何已设置的 ACL。

1.22 版和更高的 e2fsprogs 软件包所包括的 e2fsck 的实用工具版本可以检查带有 ext_attr 属性的文件系统。旧版本拒绝检查。

18.7 1 参考

更多信息，请参考以下手册页

man acl — ACLs 描述

man getfacl — 讨论如何获得文件访问控制列表

man setfacl — 解释如何设置文件访问控制列表

man star — 解释 star 实用程序及其许多选项的更多信息。

19. 写屏障

写屏障是一个内核机制，以确保该文件系统元数据可以在永续性存储体上被正确地写入和排序 ，即使是当带有不稳定写缓存的存储设备断电时。启用写屏障的文件系统也能够确保通过 fsync() 传送的数据在断电过程中保持持久不变。

启用写屏障导致了某些应用程序的大量性能损失。具体来说，大量使用 fsync（）或创建和删除许多小文件的应用程序运行起来可能会慢得多。

19.1 写屏障的重要性

文件系统在安全更新元数据方面表现得非常谨慎，以确保一致性。日志文件

系统将元数据更新捆绑到事务记录，并以下列方式发送到持久性存储体：

1） 首先，文件系统发送事务主体到存储设备。

2） 然后，文件系统发送一个提交块。

3） 如果事务和其相应的提交块被写入磁盘，文件系统假设该事务会在任何断电情况下幸存下来。

然而，电源故障期间的文件系统完整性对于带有额外高速缓存的存储设备来说变得更加复杂。存储目标设备如本地 S - ATA 或 SAS 驱动器可能有 32MB—64MB 大小（带有现代驱动器的）的写入缓存。硬件 RAID 控制器通常包含内部写入缓存。此外，高端阵列，例如那些来自 NetApp，IBM，日立和 EMC（除了别的之外），也有大容量高速缓存。

当数据在高速缓存中时，带有写入缓存的存储设备报告 I / O 是“完整的”；如果缓存断电，数据也会丢失。更糟糕的是，当缓存转移到持久性存储体，它可能改变原有的元数据排序。如果发生这种情况，提交块会出现在磁盘上而无需完整的关联事务就位。因此，在断电后恢复期间，该日志可能会将这些未初始化的事务块重播到文件系统，这将导致数据不一致和数据腐蚀。

写屏障是如何工作的？

写屏障是通过 I / O 之前和之后的存储写入缓存刷新而在 Linux 内核中实现的，这是一个紧急命令 order-critical 。事务写入完成后，存储缓存将被刷新，提交块被写入，然后缓存被再次刷新。这将确保：

1） 磁盘包含所有的数据。

2） 没有发生重新排序。

启用写屏障的环境下，fsync() 调用也将发出一个存储缓存刷新命令。这样可以保证文件数据在磁盘上的持久性，即使是在 fsync() 返回后不久发生断电的情况下。

19.2 启用/禁用 写屏障

为了降低断电期间数据损坏的风险，一些存储设备使用电池供电的写入高速缓存。一般来说，高端阵列和一些硬件控制器使用电池供电的写入缓存。然而，由于缓存的波动性对内核来说是不可见的，中标麒麟可信操作系统 V6.0V6.0 在所有支持的日志文件系统上以默认的方式启用写屏障。

 备注：写入缓存的目的是增加 I / O 性能。然而，启用写屏障是指不断刷新这些高速缓存，它可以显著降低性能。

对于具有非易失性，电池供电的写入高速缓存和禁用写缓存的设备，您可以在挂载时使用 mount 的-o nobarrier 选项来安全地禁用写屏障。然而，有些设备不支持写屏障，这些设备将错误消息记录到/var/log/ messages 中（参阅表 19-1“每个文件系统的写屏障错误消息”）。

表 19-1 每个文件系统的写屏障错误信息

文件系统	错误信息
ext3/ext4	JBD： 基于写屏障的同步操作在禁用写屏障的设备上失败。
XFS	文件系统设备 – 禁用写屏障，试用屏障写入失败。
btrfs	btrfs： 禁用 dev 设备上的写屏障。

19.3 写屏障注意事项

某些系统配置不需要写屏障来保护数据。在大多数情况下，其他方法均优于写屏障，因为启用写入障碍会导致显著的性能损失。

禁用写缓存

避免数据完整性问题的另一种方法是确保在断电的情况下没有任何写入缓存数据丢失。如果可能的话，最好的配置办法是简单地禁用写入缓存。在一个简单的服务器或带有一个或多个 SATA 驱动器的台式机上（关闭本地 SATA 控制器 Intel AHCI 部分），您可以用 hdparm 命令禁用目标 SATA 驱动器上的写入缓存，如：

```
hdparm -W0 /device/
```

电池供电的写入缓存

无论系统在何时使用电池供电的带有写高速缓存的硬件 RAID 控制器，写屏障都是不必要的。如果系统配有这样的控制器并且它的组建驱动器禁用写入缓存，该控制器将提醒自身成为一个透写式高速缓存，这将通知内核在断电时不丢失写入缓存数据。

大多数控制器使用供应商特定的工具来查询和操作目标驱动器。例如，LSI Megaraid SAS 控制器使用电池供电的写高速缓存，这种类型的控制器需要 MegaCli64 工具来管理目标驱动器。若要显示所有 SI Megaraid SAS 的后端驱动

器的状态，请使用命令：

```
MegaCli64 -LDGetProp -DskCache -LAll -aALL
```

若要禁用所有 SI Megaraid SAS 后端驱动器的写入缓存，请使用命令：

```
MegaCli64 -LDSetProp -DisDskCache -Lall –aALL
```

 备注：在系统正常运行时，硬件 RAID 卡为其电池充电。如果一个系统长时间断电，电池将失去电荷，使其中存储的数据在停电时容易损坏。高端阵列

高端阵列在电源故障时有多种保护数据的方式。因此，没有必要验证外部 RAID 存储体中内部驱动器的状态。

NFS

NFS 客户端不需要启用写屏障，因为数据的完整性是由 NFS 服务器端进行处理的。因此，应配置 NFS 服务器以确保整个断电期间数据的持久性（无论是通过写屏障或其他手段）。

20. 存储 I / O 对齐和大小

SCSI 和 ATA 标准的最新增强功能允许存储设备指出他们的首选（在某些情况下，所需的）I / O 对齐和 I / O 大小。这个信息对于将物理扇区大小从 512 字节增加到 4K 字节的新的磁盘驱动器来说特别有用。此信息也可能有利于 RAID 设备，其中数据块大小和条带大小可能会影响性能。

Linux 的 I / O 栈已得到增强以便处理供应商提供的 I / O 对齐和 I / O 大小的信息，使存储管理工具（parted、lvm、mkfs.*等等）优化数据布局和访问。如果旧设备不导出 I / O 对齐方式和大小数据，那么中标麒麟可信操作系统 V6.0 中的存储管理工具将保守地在一个 4K（或更大的概率为 2）边界上对齐 I / O。这将确保 4K 扇区设备正常运行，即使他们没有指定任何所需/首选的 I / O 对齐方式和大小。

备注：中标麒麟可信操作系统 V6.0 支持 4K 扇区设备作为数据磁盘，而不是作为启动盘。计划在以后的版本设计对 4K 扇区设备的启动支持。

参见“20.2 用户空间访问”来学习如何确定操作系统从设备获得的信息。这个数据随后被用于存储管理工具以确定数据的位置。

20.1 存储访问的参数

操作系统使用以下信息来确定 I / O 对齐和大小：

physical_block_size

该设备可以操作的最小内部单元

logical_block_size

外部使用，以解决设备上的位置

alignment_offset

Linux 块设备（分区/ MD/ LVM 设备）的开头部分从底层物理对齐偏移的字节数

minimum_io_size

该设备对于随机 I / O 的首选最小单元

optimal_io_size

该设备对于流式 I / O 的首选单元

例如，某些 4K 扇区设备内部使用 4K physical_block_size 但在外部提供更细化的 512 字节 logical_block_size 给 Linux。这种差异可能导致 I / O 没有对齐。为了解决这个问题，中标麒麟可信操作系统 V6.0V6.0I / O 栈将尝试在自然对齐的边界上（physical_block_size）启动所有数据区，如果块设备的开头部分偏移底层物理对齐位置，确保它可以为任何 alignment_offset 作出解释。

存储厂商也可以提供有关设备的随机 I / O(minimum_io_size) 和流式 I / O(optimal_io_size) 首选最小单元的 I / O 提示。例如，minimum_io_size 和 optimal_io_size 可能分别对应一个 RAID 设备的数据块大小和条带大小。

20.2 用户空间访问

始终照顾到使用正确对齐的和大小精确的 I / O。这一点对直接 I / O 访问尤为重要。直接 I / O 应在 logical_block_size 边界上对齐，并以 logical_block_size 倍数为单位。

由于本地设备是 4K（即 logical_block_size 是 4K），现在关键的是应用程序以设备的 logical_block_size 倍数为单位执行直接 I/O。这意味着应用程序将失败，因为本地 4K 设备将执行 512 字节对齐的 I / O，而不是 4K 对齐的 I / O。

为了避免这种情况，应用程序应咨询设备的 I / O 参数，以确保它使用的是正确的 I / O 对齐方式和大小。如前所述，I / O 参数通过 sysfs 和块设备 ioctl 接口进行输出。

对于更多细节，请参考 man libblkid。本手册页由 libblkid-devel 软件包提供。

sysfs 接口

- /sys/block/disk/alignment_offset
- /sys/block/disk/partition/alignment_offset
- /sys/block/disk/queue/physical_block_size
- /sys/block/disk/queue/logical_block_size
- /sys/block/disk/queue/minimum_io_size
- /sys/block/disk/queue/optimal_io_size

对于不提供 I / O 参数信息的“遗留”设备，内核仍将输出这些 sysfs 属性，例如：

```
alignment_offset:   0
physical_block_size:    512
logical_block_size: 512
minimum_io_size:  512
optimal_io_size:    0
```

块设备 ioctls

- BLKALIGNOFF: alignment_offset
- BLKPBSZGET: physical_block_size
- BLKSSZGET: logical_block_size
- BLKIOMIN: minimum_io_size
- BLKIOOPT: optimal_io_size

20.3 标准

本节描述了 ATA 和 SCSI 设备所使用的 I/O 标准。

ATA

ATA 设备必须通过 IDENTIFY DEVICE 命令报告适当的信息。ATA 设备仅对 physical_block_size、logical_block_size 和 alignment_offset 报告 I / O 参数。附加 I / O 提示在 ATA 命令集范围以外。

SCSI

中标麒麟可信操作系统 V6.0V6.0 的 I / O 参数支持要求至少 SCSI Primary Commands (SPC-3)协议第 3 版。内核将只发送一个 extended inquiry 扩展查询（获得 BLOCK LIMITS VPD 页面的访问权）和 READ CAPACITY(16) 命令到要求符合 SPC- 3 协议的设备上。

READ CAPACITY(16)命令提供块大小和对齐偏移：

1）LOGICAL BLOCK LENGTH IN BYTES 被用来驱动

/sys/block/disk/queue/ physical_block_size

2） LOGICAL BLOCKS PER PHYSICAL BLOCK EXPONENT 被用来驱动 /sys/block/disk/ queue/logical_block_size

3） LOWEST ALIGNED LOGICAL BLOCK ADDRESS 被用来驱动：

/sys/block/disk/alignment_offset

/sys/block/disk/partition/alignment_offset

BLOCK LIMITS VPD 页面 (0xb0) 提供 I/O 提示信息。同时也使用 OPTIMAL TRANSFER LENGTH GRANULARITY 和 OPTIMAL TRANSFER LENGTH 来驱动：

/sys/block/disk/queue/minimum_io_size

/sys/block/disk/queue/optimal_io_size

sg3_utils 软件包提供 sg_inq 实用程序， 该程序可以用来访问 BLOCK LIMITS VPD 页面。要做到这一点，请运行：

```
sg_inq -p 0xb0 disk
```

20.4 I/O 堆栈参数

Linux I / O 堆栈的所有层已被设计用来将各种 I / O 参数传播到堆栈上。当一个层消耗了属性或聚集了许多设备时，该层必须公开相应的 I / O 参数，使上层的设备或工具对存储转换有一个准确的视图。实例如下：

1） I / O 堆栈的唯一一层应该对非零 alignment_offset 作出调整， 一旦某一层作出了相应调整，将导出一个 alignment_offset 为零的设备。

2） 用 LVM 创建的条带化设备映射器（DM）必须导出一个相对于条带计数（磁盘数）和用户提供的数据块大小的 minimum_io_size 和 optimal_io_size。

在中标麒麟可信操作系统 V6.0V6.0 中，设备映射和软件 RAID（MD）设备驱动程序可以被用来任意地组合不同 I / O 参数的设备。内核的块层将尝试合理地把各个设备的 I / O 参数结合起来。内核不会阻止异类设备的相结合， 然而，请注意这样做所带来的风险。

例如，一个 512 字节的设备和一个 4K 设备可能合并成一个单一的逻辑 DM 设备，这将有一个 4K 的 logical_block_size。在这样一个混合设备上分层的文件系统假设 4K 将被以原子级写入，但在现实中，当被发送到 512 字节的设备上时，

它将跨越 8 个逻辑块地址。如果有一个系统故障，对更高级别的 DM 设备使用一个 4K logical_block_size 会增加只把部分数据写入到 512 字节设备上的可能性。

如果整合多台设备的 I / O 参数导致冲突，块层可能会发出警告即该设备很容易受局部写入操作的影响和/或没有对齐。

20.5 LVM (逻辑卷管理)

LVM 提供用于管理内核的 DM 设备的用户空间工具。LVM 将转换数据区（一个给定的 DM 设备将使用）的启动部分来解释与 LVM 管理的任何设备相关的非零 alignment_offset。这意味着逻辑卷将被正确地对齐（alignment_offset=0）。

默认情况下，LVM 将为任何 alignment_offset 作出调整，但通过在 /etc/lvm/lvm.conf 中设置 data_alignment_offset_detection 为 0，这种行为可能被禁用。不建议禁用此行为。

LVM 还将检测设备的 I / O 提示。设备数据区的开头部分将是 sysfs 中公开的 minimum_io_size 或 optimal_io_size 的倍数。如 optimal_io_size 未被定义（即 0），LVM 将使用 minimum_io_size。

默认情况下，LVM 将自动确定这些 I/O 提示，但通过在/etc/lvm/lvm.conf 中设置 data_alignment_detection 为 0，这种行为可能被禁用。不建议禁用此行为。

20.6 分区和文件系统工具

本节介绍不同分区和文件系统管理工具如何与设备的 I / O 参数进行交互。

util-linux-ng's libblkid and fdisk

util-linux-ng 软件包的 libblkid 库包括一个纲领性的 API 来访问设备的 I / O 参数。libblkid 允许应用程序，特别是那些使用直接 I / O 的应用程序正确地估计其 I / O 请求。来自 util-linux-ng 的 fdisk 实用程序使用 libblkid 为所有分区的最佳布局确定设备的 I / O 参数。fdisk 实用工具将调整 1MB 边界上的所有分区。

parted and libparted

来自 parted 的 libparted 库使用 libblkid 的 I / O 参数 API。中标麒麟可信操作系统 V6.0V6.0 安装程序（Anaconda）使用 libparted，这意味着所有安装程序或 parted 创建的分区都将被正确地对齐。对于似乎不会提供 I / O 参数的设备上创建的所有分区，默认进行 1MB 的调整。

parted 使用的经验如下：

1） 始终使用报告的 alignment_offset 作为第一个主分区开头部分的偏移。

2） 如果 optimal_io_size 已被定义（即不为 0），请在 optimal_io_size 边界上对齐所有分区。

3） 如果 optimal_io_size 未被定义（即 0），alignment_offset 为 0，并且 minimum_io_size 是 2 的乘方，请使用 1MB 的默认对齐方式。

这对于那些似乎不会提供 I / O 提示的“遗留”设备都是有用的。因此，默认情况下所有分区会在 1MB 的边界上对齐。

 备注：中标麒麟可信操作系统 V6.0 无法区分那些不提供 I / O 提示的设备和那些以 alignment_offset=0 和 optimal_io_size=0 提供 I / O 提示的设备。这种设备可能是单一的 SAS 4K 设备，正因为如此，在磁盘开始时，最多也就会丢失 1MB 空间。

文件系统工具

不同的 mkfs.filesystem 实用程序也得到了增强以消耗设备的 I / O 参数。这些实用程序将不会允许文件系统被格式化用来使用一个比底层存储设备 logical_block_size 小的块大小。

除 mkfs.gfs2 之外，所有其他的 mkfs.filesystem 实用程序也可以使用 I/O 提示来布置相对于 minimum_io_size 和 optimal_io_size 的底层存储设备的盘上数据结构和数据区。这使得文件系统为适应各种 RAID（条带化的）布局以最优化的方式进行格式化。

21. 设置远程无盘系统

网络启动服务(由 system-config-netboot 提供)在中标麒麟可信操作系统 V6.0V6.0 中不再可用。无需使用 system-config-netboot，在此版本中部署无盘系统也是可能的。

若要建立一个基本的通过 PXE 启动的远程无盘系统，您需要下面的软件包：

tftp-server

xinetd

dhcp

syslinux

dracut-network

远程无盘系统启动需要一个 tftp 服务（由 tftp 服务器提供）和 DHCP 服务（由 DHCP 提供）。tftp 服务用于通过 PXE 加载程序来检索网络上的内核映像和 initrd。

tftp 和 DHCP 服务都必须提供在同一台主机上。

以下各节概述了在网络环境中部署远程无盘系统的必要程序。

21.1 为无盘客户端配置 tftp 服务

默认情况下 tftp 服务是禁用的。若要启用它，并允许通过网络进行 PXE 启动，请设置/etc/xinetd.d/tftp 中 Disabled 选项为 no。若要配置 tftp，请执行以下步骤：

1） tftp 根目录 (chroot) 位于 /var/lib/tftpboot 。复制 /usr/share/syslinux/pxelinux.0 到 /var/lib/tftpboot/， 如：

```
cp /usr/share/syslinux/pxelinux.0 /var/lib/tftpboot/
```

2） 在 tftp 根目录中创建一个 pxelinux.cfg 目录：

```
mkdir -p /var/lib/tftpboot/pxelinux.cfg/
```

您还需要正确配置防火墙规则，允许 tftp 流量， 由于 tftp 支持 TCP wrapper 包，您可以通过/etc/hosts.allow 来配置主机到 tftp 的访问。man hosts_access 提供了/etc/ hosts.allow 文件的信息。

为无盘客户端配置 tftp 后，相应地配置 DHCP，NFS 和导出的文件系统。参见 “21.2 为无盘客户端配置 DHCP”和“21.3 为无盘客户端配置导出文件系统”来获得相应指导。

21.2 为无盘客户端配置 DHCP

配置 tftp 服务器后，您需要在同一台主机上建立一个 DHCP 服务。此外，您应该在 DHCP 服务器上启用 PXE 启动功能， 要做到这一点，须添加以下配置到 /etc/dhcp/dhcp.conf：

```
allow booting;
allow bootp;
class "pxeclients" {
match if substring(option vendor-class-identifier, 0, 9) = "PXEClient";
next-server server-ip;
filename "linux-install/pxelinux.0";
}
```

用 tftp 和 DHCP 服务驻留的主机的 IP 地址来替换 server-ip。现在，tftp 和 DHCP 已完成配置，所剩下的工作就是配置 NFS 和导出的文件系统， 具体说明，请参见 “21.3 为无盘客户端配置导出文件系统”。

21.3 为无盘客户端配置导出文件系统

导出文件系统（由网络中的无盘客户端使用）的根目录通过 NFS 共享。配置 NFS 服务以便通过把它添加到/etc/exports 来导出根目录。关于如何进行此操作的说明，请参见 “12.6.1etc/exports 配置文件”。

为了完全适应无盘客户端，根目录应该包含一个完整的中标麒麟可信操作系统 V6.0 安装。您可以通过 rsync 与运行的系统同步，如：

```
rsync -a -e ssh --exclude='/proc/*' --exclude='/sys/*' hostname.com:/ / exported/root/directory
```

通过 rsync 进行同步的运行中系统的主机名被用来替换 hostname.com. /exported/root/directory 是导出文件系统的路径。

另外，您还可以使用带有- installroot 选项的 yum 来安装中标麒麟可信操作系统 V6.0 到特定的位置。例如：

```
yum groupinstall Base --installroot=/exported/root/directory
```

在可以用于无盘客户端之前，要导出的文件系统仍然需要进行进一步配置。为了解决这个问题，请执行以下步骤：

1） 配置导出的文件系统的/etc/fstab 包含（至少）以下配置：

```
none /tmp tmpfs defaults 0 0
tmpfs /dev/shm tmpfs defaults 0 0
sysfs /sys sysfs defaults 0 0
proc /proc proc defaults 0 0
```

2） 选择无盘客户端应该使用的内核（vmlinuz-kernel-version），并将其复制到 tftp 启动目录：

```
cp /boot/vmlinuz-kernel-version /var/lib/tftpboot/
```

3） 创建网络支持的 initrd（即 initramfs-kernel-version.img）：dracut initramfs-kernel-version.img vmlinuz-kernel-version 同样复制生成的 initramfs-kernel-version.img 到 tftp 启动目录中。

4） 编辑默认的启动配置以便使用/var/lib/tftpboot 目录中的 initrd 和 内核。此项配置应该指导无盘客户端的根目录以读-写方式挂载导出文件系统（/exported/root/directory）。若要完成这一步， 请用以下命令配置 /var/lib/tftpboot/ pxelinux.cfg/default：

```
default NS6
label NS6
kernel vmlinuz-kernel-version
proc /proc proc defaults 0 0
```

```
append initrd=initramfs-kernel-version.img root=nfs:server-ip:/exported/root/directory
rw
```

用 tftp 和 DHCP 服务驻留的主机的 IP 地址来替换 server-ip。

NFS 共享现在已准备好导出到无盘客户端。这些客户端可以通过 PXE 从网络启动。

22. 固态硬盘部署指南

固态磁盘（SSD）是使用 NAND 闪存芯片持久性地存储数据的存储设备。这一点使得他们有别于前几代使用旋转的磁性盘片来存储数据的磁盘。在一个 SSD 中，跨整个逻辑块地址（LBA）数据的访问时间范围是不变的，然而对于较老的使用旋转介质的磁盘，跨度大的地址范围的访问模式产生搜索成本。因此，SSD 设备有更好的延迟和吞吐量。

然而，并不是所有的 SSD 表现出同样的性能配置。事实上，相对于旋转介质，许多第一代设备很少或根本没有优势。因此，重要的是定义固态存储体的类别以便在本节中进一步讨论。

根据吞吐量，固态硬盘可以分为三类：

1） 第一类 SSD 使用 PCI – Express 连接，它提供了比其他任何类别都快的 I / O 吞吐量。该类也有一个非常低的随机存取延迟。

2） 第二类使用传统的 SATA 连接，并具有快速随机读写访问的操作（尽管没有使用 PCI - Express 连接的固态硬盘那么快）。

3） 第三类也采用了 SATA，但此类 SSD 的性能与使用 7200 转转速的磁盘设备没有实质性区别。

对于所有这三个类别，当使用的块数接近磁盘容量时，性能下降。由于供应商不同，性能的影响程度差别很大。然而，所有的设备都会遇到某些方面的性能下降。

为了解决性能下降问题，主机系统（例如，Linux 内核）可能使用丢弃请求通知存储器不再使用规定范围的块。一个固态盘可以使用此信息以释放内部空间，使用空闲块进行平均读写。只有存储体支持其存储协议（ATA 或 SCSI）时，丢弃请求才会发出。丢弃请求被发送到使用存储协议特定的 discard 命令的存储体上（TRIM 命令用于 ATA 和带有 UNMAP 集的 WRITE SAME 或 用于 SCSI 的 UNMAP 命令）。

当文件系统上还有可用的自由空间时，启用 discrad 支持是最有用的，但文件系统已经被写入到底层存储设备的大多数逻辑块上。有关 TRIM 的更多信息，请参阅各以下链接中的数据集管理 T13 规范：

http://t13.org/Documents/UploadedDocuments/docs2008/e07154r6-Data_Set_Management_Proposal_for_ATA-ACS2.doc

有关 UNMAP 的更多信息，请参阅各以下链接中的第 4. 7. 3. 4 节 SCSI 块命令 3 T10 规范：

http://www.t10.org/cgi-bin/ac.pl?t=f&f=sbc3r26.pdf

 备注：并非市场上的所有固态设备都支持 discard。

22.1 部署注意事项

由于 SSD 的内部布局和操作，最好是在内部擦除块边界上对设备进行分区。如果 SSD 导出拓扑信息，中标麒麟可信操作系统 V6.0V6.0 中的分区实用程序将会选择合乎情理的默认值。

但是，如果该设备没有导出拓扑信息，我们建议应该在 1MB 边界上创建第一个分区。

此外，请记住，MD（软件 RAID）并不支持 discard。相比之下，LVM 使用的逻辑卷管理器（LVM）和设备映射（DM）目标确实支持 discard。唯一不支持 discard 的 DM 目标是 dm-snapshot、dm-crypt 和 dm-raid45。dm-映像的 discard 支持被添加到中标麒麟可信操作系统 V6.0V6.0 中。

我们不建议您将 1，4，5，6 级软件 RAID 使用在 SSD 上。在这些 RAID 级别的初始化阶段，一些 RAID 管理实用程序（如 mdadm）会向存储设备上的所有块写入数据以确保校验正常运行。这将导致 SSD 性能迅速下降。

目前，EXT4 是唯一完全支持 discard 的文件系统。要在设备上启用 discard 命令，请使用 mount 选项 discard。例如，挂载/dev/sda2 到启用 discard 的/mnt 上，请运行：

```
mount -t ext4 -o discard /dev/sda2 /mnt
```

默认情况下，EXT4 不发出 discard 命令。这主要是避免可能无法正确执行 discard 命令的设备出现问题。Linux 交换代码会发出 discard 命令到启用 discard 功能的设备上，而且没有选项来控制这种行为。

22.2 调优注意事项

本节介绍了配置设置时可能会影响 SSD 性能的几个要考虑的因素。

I/O 调度程序

任何 I / O 调度程序都应与大多数 SSD 一起执行。然而，至于任何其他存储类型，我们建议设立基准以便为一个给定的工作量确定最优配置。

使用固态硬盘时，我们建议改变 I / O 调度程序只为了给特定工作负载设立基准。以下的内核文件还包含有关 I / O 调度程序之间如何切换的说明：

```
/usr/share/doc/kernel-version/Documentation/block/switching-sched.txt
```

虚拟文件系统

同 I / O 调度程序一样，虚拟内存（VM）子系统不需要特殊的调优。鉴于 SSD 上 I / O 的快速性，它应该有可能调低 vm_dirty_background_ratio 和 vm_dirty_ratio 设置，由于增加的写入活动不应对磁盘上其他操作的恢复时间产生负面影响。然而，这可以产生更全面的 I / O，因此未经特定工作量的测试，一般不建议进行此操作。

Swap

SSD 也可以被用来作为交换设备，可能产生良好的 page-out/page-in 性能。

23. 在线存储管理

当操作系统运行时，它往往需要在不重启的情况下添加，删除或调整存储设备的大小。本章概述了在系统运行时重新配置中标麒麟可信操作系统 V6.0V6.0 主机系统上存储设备的程序。它涵盖了 iSCSI 和光纤通道存储互连， 其它互连类型可能在未来会被增加。

本章侧重于存储设备的添加，删除，修改和监控。本章并没有详细讨论光纤通道或 iSCSI 协议。关于使用这些协议的更多信息，请参阅其他文件。

本章参考了各种 sysfs 对象。我们表示，sysfs 对象名称和目录结构均以中标麒麟可信操作系统 V6.0 主要发行版中的变更为准。这是因为上游 Linux 内核不提供稳定的内部 API。有关如何以可移植方式引用 sysfs 对象的说明，请参阅内核源代码树中文件/usr/share/doc/kerneldoc-version/Documentation /sysfs-rules.txt 来获得相应指导。

⚠警告：必须小心进行在线存储重新配置。该过程中的系统故障或中断可能会导致意

想不到的结果。我们建议您在更改操作时最大限度地减少系统负荷。这将减少配置变更过程中发生 I / O 错误、内存溢出错误或类似错误的机会。以下各节就此提供了更具体的说明。

此外，我们建议您应首先备份所有数据，然后重新配置在线存储。

23.1 光纤通道

本节讨论了光纤通道 API，中标麒麟可信操作系统 V6.0V6.0 本地光纤通道驱动程序以及这些驱动程序的光纤通道性能。

23.1.1 光纤通道 API

下面是一个/sys/class/ 目录列表，该列表包含用于提供用户空间 API 的文件。在每个项目中，主机号是由 H 指定，总线号码由 B 指定，目标由 T 指定，逻辑单元号码（LUN）由 L 指定，以及远程端口号码由 R 指定。

 提示：如果您的系统正在使用多路径软件，我们建议您在更改本节中所述的任何值之前，应咨询您的硬件供应商。

传输： /sys/class/fc_transport/targetH:B:T/

1） port_id — 24 位的端口 ID/地址

2） node_name — 64 位节点的名称

3） port_name — 64 位端口的名称

远程端口： /sys/class/fc_remote_ports/rport-H:B-R/

1） port_id

2） node_name

3） port_name

dev_loss_tmo — 标记一个链接为“坏”之前等待的秒数。一旦链接被标记为坏，运行于相应路径的 I/O（以及该路径上任何新的 I/O）将失败。

默认 dev_loss_tmo 值各不相同，这取决于使用的是哪个驱动程序/设备。如果使用 QLogic 适配器，默认是 35 秒，而如果使用一个 Emulex 适配器，默认是 30 秒。通过 scsi_transport_fc 模块参数 dev_loss_tmo，dev_loss_tmo 值是可以改变的，尽管驱动程序可以覆盖此超时值。

最高 dev_loss_tmo 值是 600 秒。然而，当 dev_loss_tmo 设置为零或任何大于 600 的值，则将使用驱动程序的内置的超时值。

fast_io_fail_tmo - 发现一个链接出现问题时，失败 I/O 执行前的等待时长。达到驱动程序的 I/O 将失败。如果 I/O 在一个阻塞队列中，直到 dev_loss_tmo

到期并且该队列变得畅通，它才会失败。

主机： /sys/class/fc_host/hostH/

port_id

issue_lip — 指示驱动程序重新搜索远程端口。

23.1.2 本机光纤通道驱动程序和性能

中标麒麟可信操作系统 V6.0V6.0 装载本地光纤通道驱动器：

1） lpfc

2） qla2xxx

3） zfcp

4） mptfc

表 23-1 描述了每个本地中标麒麟可信操作系统 V6.0 驱动程序的不同光纤通道 API 功能。X 表示对功能的支持。

表 23-1 光纤通道 API 功能

	lpfc	qla2xxx	zfcp	mptfc
传输：port_id	X	X	X	X
传输：node_name	X	X	X	X
传输：port_name	X	X	X	X
远程端口：dev_loss_tmo	X	X	X	X
远程端口：fast_io_fail_tmo	X	X [1]	X [2]	
主机：port_id	X	X	X	X
主机：issue_lip	X	X		

23.2 iSCSI

本节介绍了 iSCSI API 和 iscsiadm 实用工具。使用 iscsiadm 实用程序之前，请首先运行 yum install iscsiinitiator-utils 来安装 iscsi-initiator-utils 包。

此外，iSCSI 服务必须正在运行，以便及时发现或登录到目标方。要启动 iSCSI 服务，请运行 service iscsi start

23.2.1 iSCSI API

要获取有关正在运行会话的信息，请运行：

```
iscsiadm -m session -P 3
```

此命令显示会话/设备状态，会话 ID(SID)，一些协商参数，以及可通过会话访问的 SCSI 设备。

对于较短的输出（例如，只显示 sid 到节点的映射），请运行：

```
iscsiadm -m session -P 0
```

或

```
iscsiadm -m session
```

这些命令用此格式打印运行会话列表：

```
driver [sid] target_ip:port,target_portal_group_tag proper_target_name
```

例如：

```
iscsiadm -m session
tcp [2] 10.15.84.19:3260,2 iqn.1992-08.com.netapp:sn.33615311
tcp [3] 10.15.85.19:3260,3 iqn.1992-08.com.netapp:sn.33615311
```

关于 iSCSI API 的更多信息，请参考 /usr/share/doc/iscsi-initiator-utils-version/README。

23.3 持久性命名

通过引用可到达一个存储设备的路径，操作系统发出 I/O 到此存储设备上。对于 SCSI 设备，路径包括以下内容：

1） 主机总线适配器（HBA）的 PCI 标识符

2） 该 HBA 的通道号码

3） 远程 SCSI 目标方地址

4） 逻辑单元号（LUN）

这个基于路径的地址不是永久性的。系统重新配置时它可能会改变（无论是通过本手册中所述的在线重新配置，或当系统关机时重新配置，并重新启动）。甚至当没有物理重新配置发生时，路径标识符也可能改变；这是由于当系统启动时或者当一个总线被重新扫描时发现过程中的时间变化而引起的。

操作系统提供了一些非持久的名称，代表这些到存储设备的访问路径。一个是 /dev/sd 名称，另一个是 major:minor 号码。第三个是在/dev/disk/by-path/目录中保持的一个符号链接。这个符号链接将路径标识符映射到当前 /dev/sd 名称。例如，对于一个光纤通道设备，PCI 信息和 Host:BusTarget:LUN 信息可能会显示如下：

```
pci-0000:02:0e.0-scsi-0:0:0:0 -> ../../sda
```

对于 iSCSI 设备，by-path/ 名称将目标方名称和门户的信息映射到 sd 名称。

一般来说应用程序不适合使用这些基于路径的名称。这是因为这些路径引用的存储设备可能会改变，也许会导致把不正确的数据写入到设备。基于路径的名称也不适合多路径设备，因为基于路径的名称可能会被误认为是单独的存储设备，导致不协调的访问和意外修改数据。

此外，基于路径的名称是系统特定的。当多个系统访问该设备时，这可能会导致意想不到的数据更改，如在一个集群中。

由于这些原因，一些用于识别设备的持久的、独立于系统的方法已被开发出来。下面的章节中详细地讨论了这些内容。

23.3.1 WWID

万维网标识符（WWID）可以用来可靠地识别设备。这是一个 SCSI 标准要求的持久的，与所有 SCSI 设备无关的独立于系统的 ID。对每一个存储设备，都要保证 WWID 标识符是唯一的，并且不依赖于于访问设备的路径。

通过发出 SCSI Inquiry 查询命令来检索设备标识的重要产品数据（页 0x83）或单元序列号（页 0x80 的），可获取此标识符。在/dev/disk/by-id/ 目录中保持的符号链接中，可以看出从这些 WWIDs 到当前/dev/sd 名称的映射。

例如，带有页 0x83 标识符的设备会具有如下信息：

```
scsi-3600508b400105e21000090000049000 -> ../../sda
```

或者，带有页 0x80 标识符的设备会具有如下信息：

```
scsi-SSEAGATE_ST373453LW_3HW1RHM6 -> ../../sda
```

中标麒麟可信操作系统 V6.0 自动保持从基于 WWID 的设备名称到该系统上当前 /dev/sd 名称的适当映射。即使设备的路径发生变化，甚至从不同系统访问设备时，应用程序可以使用/dev/disk/by-id/ 名称来引用磁盘上的数据。

如果从系统到设备有多重路径，device-mapper-multipath 使用 WWID 检测这些路径。然后，Device-mapper-multipath 给出/dev/mapper/ wwid 中单独一个 "伪设备" ， 如/dev/mapper/3600508b400105df70000e00000ac0000。

multipath -l 命令显示了到非持久性标识符的映射： 主机：通道：目标方：LUN，/dev/sd 名称，以及 major:minor 号码。

3600508b400105df70000e00000ac0000 dm-2 vendor, product

```
[size=20G][features=1 queue_if_no_path][hwhandler=0][rw]
```

```
\_ round-robin 0 [prio=0][active]
\_ 5:0:1:1 sdc 8:32 [active][undef]
\_ 6:0:1:1 sdg 8:96 [active][undef]
\_ round-robin 0 [prio=0][enabled]
\_ 5:0:0:1 sdb 8:16 [active][undef]
\_ 6:0:0:1 sdf 8:80 [active][undef]
```

Device-mapper-multipath 自动保持基于 WWID 的设备名称到该系统上对应的/dev/sd 名称的适当映射。这些名称在路径变化后也保持不变，并且从不同系统访问设备时，它们也是一致的。

当使用（device-mapper-multipath 的）user_friendly_names 功能时，WWID 被映射到一个格式为/dev/mapper/mpathn 的名称。默认情况下，这个映射被保存在文件 /var/lib/multipath/bindings 中。只要该文件被保持不变，这些 mpathn 名称就是永久的。

警告：多路径绑定的文件（默认情况下，var/lib/multipath/bindings）必须在启动时可用。如果/ var 是一个来自/的单独的文件系统，那么您必须改变文件的默认位置。

提示：如果你使用 user_friendly_names，那么需要额外的步骤是在集群中获得一致的名称。参考使用设备映射多重路径 2 一书中 的一致的多重路径名称 1 一节 。

除了系统提供的这些永久的名称外，您也可以使用 udev 规则来实施你自己的永久性名称，映射到存储体的 WWID。

23.3.2 UUID 和其他持久性标识符

如果一个存储设备包含一个文件系统，那么该文件系统可以提供下列中的一个或两个：

1） 通用唯一识别码(UUID)。

2） 文件系统标签。

这些标识符是永久的，并且基于由某些应用程序写入到设备上的元数据。通过使用/dev/disk/by-label/ (如 boot -> ../../sda1) 和 /dev/disk/by-uuid/ (如 f8bf09e3-4c16-4d91-bd5e-6f62da165c08 -> ../../sda1)目录中操作系统所保持的符号链接，也可使用它们来访问设备。

md 和 LVM 将元数据写入到存储设备上，并且在扫描设备时读取数据。在每一种情况下，元数据都包含一个 UUID，因此，无论该设备的路径（或系统）访问该设备的路径如何，都可以识别该设备。因此，只要元数据保持不变，这些设施提供的设备名称是永久的。

23.4 删除存储设备

在删除到存储设备本身的访问之前，最好先从设备备份数据。之后，刷新 I / O 并删除所有引用该设备的操作系统（如下所述）。如果该设备使用多路径，那么为多路径“伪设备”和每个代表设备路径的标识符执行此操作。如果你只删除到多路径设备的一个路径，其他的路径将保持，那么这个过程就很简单，如 “23.6 添加存储设备或路径”所述。

当系统处于内存压力下时，不建议删除存储设备，因为 I/O 刷新将增加负载。要确定内存压力的水平，请运行 vmstat 1 100 命令；如果出现以下状况，不建议删除设备：

1） 10％以上样本的可用内存小于总内存的 5％（free 命令也可以用来显示总内存）。

2） 交换处于活动状态（vmstat 输出中的非零 si 和 so 栏）。

删除所有对设备的访问的一般过程如下：

确保清洁的设备删除步骤：

1） 关闭设备的所有用户并且按需备份设备数据。

2） 使用 umount 命令卸载安装该设备的任何文件系统。

3） 从使用它的任何 md 和 LVM 卷中删除该设备。如果该设备是一个 LVM 卷组的成员，那么可能有必要使用 pvmove 命令从设备上删除数据，然后使用 vgreduce 命令删除物理卷，和（可选）pvremove 命令从磁盘删除 LVM 元数据。

4） 如果设备使用多路径，运行 multipath -l 并注意所有到设备的路径。之后，使用 multipath -f device 删除多路径设备。

5） 运行 blockdev –flushbufs device 刷新任何未决 I/O 到该设备所有路径。这对于原始设备尤其重要，因为那里没有 umount 或 vgreduce 操作来引起 I/O 刷新。

6） 删除设备的基于路径的名称的任何引用，如系统中应用程序，脚本或实用程序中的/dev/sd、/dev/disk/by-path 或 major:minor 号码。确保将来添加的不同设备不会被误认为是当前设备，这一点很重要。

7） 最后，从 SCSI 子系统删除到设备的每个路径。要做到这一点，使用命令 echo 1 > /sys/block/device-name/device/delete，其中设备的名称可能是

sde。此操作的另一个变形是 echo 1 > /sys/class/scsi_device/h:c:t:l/device/delete，其中 h 是 HBA 号码，c 是 HBA 上的通道，t 是 SCSI 目标方 ID，以及 l 是 LUN。

 备注：这些命令的旧形式 echo "scsi remove-single-device 0 0 0 0" > /proc/scsi/scsi，不宜使用。

您可以为各种命令中的设备确定设备的名称，HBA 号码，HBA 通道，SCSI 目标方 ID 和 LUN，如 lsscsi、 scsi_id、multipath –l 和 ls -l /dev/ disk/by-*。

执行步骤 “确保清洁的设备删除”后，设备可以从正在运行的系统中安全地被物理删除。执行此操作时，没有必要停止到其他设备 I/O 活动。

不推荐其他过程，如物理删除设备后是重新扫描 SCSI 总线（如 “23.9 扫描存储连接”所述），从而导致操作系统的状态被更新以反映所发生的变化。由于 I/O 超时会导致延误，并且设备可能会被意外删除。如果有必要对一个连接执行重新扫描，这只能当 I/O 暂停时完成，如“23.9 扫描存储连接”所述。

23.5 删除到存储设备的路径

如果要删除到多路径设备的一个路径（不影响其他到此设备的路径），那么一般过程如下：

删除到存储设备的路径

1） 移除对设备的基于路径的名称的任何引用，如系统中应用程序，脚本或实用程序中的/dev/sd、/dev/disk/by-path 或 major:minor 号码。确保将来添加的不同设备不会被误认为是当前设备，这一点很重要。

2） 使用 echo offline > /sys/block/sda/device/state 来选择脱机路径。

3） 这将导致任何发送到该路径上设备的后续 I/O 立即失败。Device-mapper-multipath 将继续使用到该设备的剩余路径。

4） 删除 SCSI 子系统的路径。要做到这一点，请使用命令 echo 1 > /sys/block/device-name/device/delete，其中设备名称 device-name 可能是 sde，例如 (如 步骤 “确保清洁的设备删除”)。

执行步骤“删除到存储设备的路径”后，该路径可以安全地从正在运行的系统中删除。执行此操作过程中，没有必要停止 I / O，因为 devicemapper-multipath 根据配置的路径分组和故障切换策略将 I /O 重新路由到剩余的路径。

不推荐其他步骤，如物理删除设备后重新扫描 SCSI 总线，将导致操作系统的状态被更新以反映所发生的变化。由于 I/O 超时会导致延误，并且设备可能会被意外删除。如果有必要对一个连接执行重新扫描，这必须在 I/O 暂停时完成，如 “23.9 扫描存储连接”所述。

23.6 添加存储设备或路径

添加设备时，您应该知道系统分配给新设备的基于路径的设备名称（/dev/sd 名称，major:minor 号码，以及/dev/disk/by-path 名）。您应确保所有过去的基于路径的设备名称的引用已被删除。否则，新设备可能被误认为是旧设备。

添加存储设备或路径的第一步是在物理上启用到新存储设备的访问，或开启到现有设备的一个新路径。这一步需要使用供应商特定的命令在光纤通道或 iSCSI 存储服务器上完成。执行过程中，请注意将被提交到主机的新存储的 LUN 值。如果存储服务器是光纤通道，还需要注意的是存储服务器的万维网节点名称（WWNN），并确定是否有一个单一的 WWNN 适用于存储服务器上的所有端口。如果情况并非如此，请注意将用于访问新 LUN 的每个端口的万维网端口名称（WWPN）。

下一步，使操作系统知道新的存储设备，或现有设备的新路径。建议使用的命令是：

```
echo "c t l" > /sys/class/scsi_host/hosth/scan
```

在前面的命令中，h 代表 HBA，c 代表 HBA 上的通道，t 代表 SCSI 目标方 ID，l 代表 LUN。

备注：这些命令的旧格式 echo "scsi add-single-device 0 0 0 0" > /proc/ scsi/scsi， 不宜使用。

对所有端口实施单一 WWNN 的光纤通道存储服务器，您可以通过搜索 sysfs 中的 WWNN 来确定正确的 h、c 和 t 值（即 HBA 号码，HBA 通道和 SCSI 目标方 ID）。例如， 如果存储服务器的 WWNN 是 0x5006016090203181， 请使用：

```
grep 5006016090203181 /sys/class/fc_transport/*/node_name
```

将显示类似于以下界面的输出：

```
/sys/class/fc_transport/target5:0:2/node_name:0x5006016090203181
/sys/class/fc_transport/target5:0:3/node_name:0x5006016090203181
/sys/class/fc_transport/target6:0:2/node_name:0x5006016090203181
/sys/class/fc_transport/target6:0:3/node_name:0x5006016090203181
```

这表明有四个光纤通道路由到此目标方（两个单通道 HBA，每个都通向两个存储端口）。假设一个 LUN 值是 56，那么下面的命令将配置第一条路径：

```
echo "0 2 56" > /sys/class/scsi_host/host5/scan
```

必读对每个到新设备的路径执行此操作。

对于并不对所有端口执行单一 WWNN 的光纤通道存储服务器，您可以通过搜索 sysfs 中的每一个 WWPNs 来确定正确 HBA 号码，HBA 通道和 SCSI 目标方 ID。

确定 HBA 号码，HBA 通道和 SCSI 目标方 ID 的另一种方式是引用与新设备在相同路径上已配置好的另一设备。这可以通过各种命令来完成，如 lsscsi、scsi_id、multipath -l 和 ls -l /dev/disk/by-*。

可以使用此信息以及新设备的 LUN 编号来探测和配置到新设备的路径。

添加所有到设备的 SCSI 路径后，执行 multipath 命令，请检查是否设备已正确配置。此时，设备可添加到 md、LVM、mkfs 或 mount。

如果遵循上述步骤，那么设备可以安全地被添加到正在运行的系统。执行此操作时，没有必要停止到其他设备的 I/O 活动。在存储 I/O 进行时，不建议使用其他涉及到 SCSI 总线重新扫描（或复位）的步骤，从而导致操作系统更新其状态以反映当前设备的连接。

23.7 配置以太网光纤通道接口

建立和部署以太网光纤通道（FCoE）接口需要两个包：

fcoe-utils

lldpad

一旦安装了这些软件包，请执行以下步骤，启用虚拟局域网（VLAN）的 FCoE：

步骤配置以太网光纤通道 FCoE 接口，

1） 通过将现有网络脚本（例如/ etc/fcoe/cfg-eth0）复制到到支持 FCoE 的以太网设备名称，配置一个新的 VLAN。这将提供一个默认的文件来进行配置。假设 FCoE 设备是 ethX，请运行：

```
cp /etc/fcoe/cfg-eth0 /etc/fcoe/cfg-ethX
```

2） 如果你想在开机时使设备自动加载，在相应的/etc/sysconfig/network-scripts/ifcfg-ethX 文件中设置 ONBOOT=yes。例如，如果 FCoE 设备是 eth2，

那么相应地编辑/etc/sysconfig/network-scripts/ifcfg-eth2。

3） 使用以下命令来启动数据中心桥接守护进程（dcbd）：

```
/etc/init.d/lldpad start
```

4） 使用以下命令来启用以太网接口数据中心桥接：

```
dcbtool sc ethX dcb on
```

5） 然后，运行以下命令来启用以太网接口 FCoE：

```
dcbtool sc ethX app:fcoe e:1
```

备注：只有当以太网接口的 dcbd 设置没有改变时，这些命令才会生效。

6） 现在使用以下命令，加载 FCoE 设备：

```
ifconfig ethX up
```

7） 使用如下命令，启动 FCoE

```
/etc/init.d/fcoe start
```

假设光纤的所有其他设置都正确，则 FCoE 设备应立刻出现。要查看已配置的 FCoE 设备，请运行：fcoeadmin -i

正确配置以太网接口来使用 FCoE 后，我们建议您设置 FCoE 和 lldpad 在启动时运行。若要 执行此操作， 请使用 chkconfig 命令， 如：

```
chkconfig lldpad on
chkconfig fcoe on
```

警告：不要在实施 DCB 的 CNAs 上运行基于软件的 DCB 或 LLDP。

一些组合网络适配器（CNA）在固件中实现数据中心桥接（DCB）协议。DCB 协议假设在特定网络链接上只有一个 DCB 发生器。这意味着必须在实施 DCB 的 CNAs 上禁用任何较高层次的基于软件的 DCB，或链路层发现协议（LLDP）。

23.8 配置 FCoE 接口在开机时自动挂载

备注：本节中的指令自中标麒麟可信操作系统 V6.0V6.0 发行起才出现在 /usr/share/doc/fcoe-utils-version/ README 中。关于次版本的任何可能的变更，请参阅该文件。

通过 udev 规则、autofs 和其他类似的方法，您可以安装新搜索到的磁盘。然而，有时一个特定的服务可能需要在开机时挂载 FCoE 磁盘。在这种情况下，在任何需要 FCoE 磁盘的服务启动之前，fcoe 服务一运行就应立即挂载 FCoE 磁盘。

要配置一个 FCoE 磁盘在开机时自动挂载，请将合适的 FCoE 挂载代码添加

到 fcoe 服务的启动脚本。fcoe 启动脚本是 /etc/init.d/fcoe。

无论您是使用一个简单的格式化的 FCoE 磁盘、LVM、或者多路径设备节点，FCoE 挂载代码根据每个系统配置的不同而不同。以下是一个 FCoE 挂载代码的示例，用于挂载/etc/fstab 中由通配符指定的文件系统：

```
mount_fcoe_disks_from_fstab()
{
local timeout=20
local done=1
local fcoe_disks=($(egrep 'by-path\/fc-.*_netdev' /etc/fstab | cut -d ' ' -f1))
test -z $fcoe_disks && return 0
echo -n "Waiting for fcoe disks . "
while [ $timeout -gt 0 ]; do
for disk in ${fcoe_disks[*]}; do
if ! test -b $disk; then
done=0
break
fi
done
test $done -eq 1 && break;
sleep 1
echo -n ". "
done=1
let timeout-¬
done
if test $timeout -eq 0; then
echo "timeout!"
else
echo "done!"
fi
# mount any newly discovered disk
mount -a 2>/dev/null
}
```

fcoe 服务脚本启动 fcoemon 守护进程后，应该唤醒 mount_fcoe_disks_from_fstab 函数。这将以下路径中指定的 FCoE 磁盘挂载到/ etc / fstab 目录中：

```
/dev/disk/by-path/fc-0xXX:0xXX /mnt/fcoe-disk1 ext3 defaults,_netdev    0    0
/dev/disk/by-path/fc-0xYY:0xYY /mnt/fcoe-disk2 ext3 defaults,_netdev    0    0
```

带有 fc- 和_netdev 子串的条目启用 mount_fcoe_disks_from_fstab 函数以确定 FCoE 磁盘挂载点。关于/etc/fstab 条目的更多信息，请参阅 man 5 fstab。

 备注：对于 FCoE 磁盘发现，fcoe 服务并不设置超时。因此，FCoE 挂载代码应该

执行自己的超时期限。

23.9 扫描存储互连

您可以使用几个命令来重置/或扫描一个或多个互连，可能在一次操作中添加和删除多个设备。这种类型的扫描可能是破坏性的的，因为 I / O 操作超时情况下，它可以导致延误，并且意外删除设备。因此，我们建议只在必要时才可使用这种类型的扫描。此外，扫描存储互连时，必须遵守下列限制：

1） 在执行过程之前，必须暂停和刷新受影响互连上的所有 I / O，并且恢复在 I / O 之前检查的扫描结果。

2） 同删除设备一样，当系统处在内存压力下时，不建议进行互连扫描。要确定内存压力的水平，请运行命令 vmstat 1 100， 如果 10％以上样本的可用内存低于总内存的 5％，不建议互连扫描。如果交换空处于活动状态（vmstat 输出中的非零 si 和 so 栏），也不建议进行此操作。命令 free 还可以显示总内存。

可以使用以下命令来扫描存储互连。

```
echo "1" > /sys/class/fc_host/host/issue_lip
```

此操作执行环路初始化协议（LIP），然后将扫描互连，并更新 SCSI 层，以反映当前总线上的设备。LIP，从本质上讲，是一个总线复位，将引起设备的添加和删除。此过程在光纤通道互连上配置一个新的 SCSI 目标方时是非常必要的。

记住，issue_lip 是一个异步操作。该命令可能会在整个扫描已完成之前结束。您必须监视/var/log/messages 文件，以确定当它完成的时间。

lpfc 和 qla2xxx 驱动程序支持 issue_lip。对于中标麒麟可信操作系统 V6.0 中每个驱动程序支持的 API 功能的更多信息，请参阅表 23.1，“光纤通道 API 功能”。

/usr/bin/rescan-scsi-bus.sh

默认情况下，该脚本将扫描系统上的所有 SCSI 总线，更新 SCSI 层，以反映总线上的新设备。脚本提供了更多的选项，允许删除设备和发出 LIP。欲了解更多有关此脚本的（包括已知问题）详细信息，请参阅 23.15 节通过 rescan-scsi-bus.sh 命令添加/删除逻辑单元。

```
echo "- - -" > /sys/class/scsi_host/hosth/scan
```

和第 23. 6 节，“添加存储设备或路径”中描述的是同一个命令，用于添加

存储设备或路径。然而，在这种情况下，通配符取代了通道号，SCSI 目标方 ID 和 LUN 值。允许使用标识符和通配符的任意组合，使您可以根据需要编写更具体或更宽泛的命令。此过程将添加 LUN，但不会删除它们。

```
rmmod driver-name 或 modprobe driver-name
```

这些命令完全重新初始化驱动器控制的所有互连的状态。虽然这种做法有些极端，但在某些情况下，这种做法是恰当的。例如，这可以用于以不同模块参数值来重启驱动程序。

23.10 iSCSI 发现配置

默认 iSCSI 配置文件是/etc/iscsi/iscsid.conf。此文件包含 iscsid 和 iscsiadm 使用的 iSCSI 设置。

在目标方发现期间，iscsiadm 工具使用/etc/iscsi/iscsid.conf 中的设置来创建两种类型的记录：

/var/lib/iscsi/nodes 中的节点记录

登录到目标方时，iscsiadm 使用此文件中的设置。

/var/lib/iscsi/discovery_type 中的搜索记录

当执行发现相同目标方时，iscsiadm 使用此文件中的设置。

在使用不同的发现设置时，请首先删除当前的发现记录（即/var/lib/iscsi/discovery_type）。欲完成此操作，请执行以下命令：

iscsiadm -m discovery -t discovery_type -p target_IP:port -o delete[3] 此处，discovery_type 可以是 sendtargets、isns 或 fw。有两种重新配置发现记录设置的方式：

1） 执行发现操作前，直接编辑/etc/iscsi/iscsid.conf 文件。搜索设置使用前缀 discovery，要查看这些设置，请运行：

```
iscsiadm -m discovery -t discovery_type -p target_IP:port
```

2） 另外，iscsiadm 也可以直接改变搜索记录设置，如：

```
iscsiadm -m discovery -t discovery_type -p target_IP:port -o update -n setting -v %value
```

关于可用设置和每个可用设置有效值的更多信息，请参阅 man iscsiadm 。

配置完发现设置后，在后续尝试发现新目标方的操作中将使用新设置。关于如何扫描新的iSCSI目标方的详细介绍，请参阅第23. 12节，“扫描iSCSI连接”。

配置 iSCSI 目标方发现的更多信息，请参阅 iscsiadm 和 iscsid 的手册页。

/etc/iscsi/iscsid.conf 文件中也包含了正确配置的语法示例。

23.11 配置 iSCSI 卸载和接口绑定

本章介绍如何设置 iSCSI 接口以便在使用软件 iSCSI 时将一个会话绑定到 NIC 端口。还介绍了如何设置接口来使用支持卸载的网络设备，即来自 Chelsio，Broadcom 和 ServerEngines 的设备。

您可以配置网络子系统来确定 iSCSI 接口应该用于绑定的路径 NIC。例如，如果门户和 NIC 设置在不同的子网上，那么没有必要手动配置用于绑定的 iSCSI 接口。

配置用于绑定的 iSCSI 接口之前，请先运行下面的命令：

```
ping -I ethX target_IP[3]
```

如果 ping 失败，那么您将不能将会话绑定到 NIC 上。如果是这种情况，请首先检查网络设置。

23.11.1 查看可用 iface 配置

支持以下 iSCSI 发起方实现的 iSCSI 卸载和接口绑定：

1） 软件 iSCSI – 如 scsi_tcp 和 ib_iser 模块，这个栈为每个会话分配一个 iSCSI 主机实例（即 scsi_host），每个会话只有一个连接。因此，/sys/class_scsi_host 和 /proc/scsi 将对您登录的每个连接/会话报告一个 scsi_host。

2） 卸载 iSCSI – 如 Chelsio cxgb3i，Broadcom bnx2i 和 ServerEngines be2iscsi 模块，这个栈为每个 PCI 设备分配 scsi_host。因此，主机总线适配器上的每个端口会显示为不同的 PCI 设备，每个 HBA 端口有不同 scsi_host。

要管理两种类型的发起方执行，iscsiadm 使用 iface 结构。使用这种结构，iface 配置必须输入到每个 HBA 端口的/var/lib/iscsi/ifaces ，软件 iSCSI，或者用于绑定会话的网络设备(ethX) 。

要查看可用的 iface 配置，请运行 iscsiadm -m iface。这将显示以下格式的 iface 信息：

```
iface_name transport_name,hardware_address,ip_address,net_ifacename,initiator_name
```

关于每个值/设置的说明，请参阅下表。

表 23-2 iface 设置

设置	描述
iface_name	iface 配置名称。
transport_name	驱动程序名称
hardware_address	MAC 地址
ip_address	此端口使用的 IP 地址
net_iface_name	用于 vlan 的名称或软件 iSCSI 会话绑定的别名。对于 iSCSI 卸载，net_iface_name 将会是<空>，因为这个值在重新启动后并非保持不变。
initiator_name	此设置用于重载默认的发起人名称，该名称定义在 /etc/iscsi/initiatorname.iscsi 中

下面是 iscsiadm -m iface 命令输出的一个示例：

```
iface0  qla4xxx, 00:c0:dd:08:63:e8, 20. 15. 0. 7, default, iqn. 2005-06. com. NS:madmax
iface1  qla4xxx, 00:c0:dd:08:63:ea, 20. 15. 0. 9, default, iqn. 2005-06. com. NS:madmax
```

对于软件 iSCSI，每个 iface 配置必须有一个唯一的名称（少于 65 个字符）。支持卸载的网络设备 iface_name 格式为 transport_name.hardware_name。

例如，使用 Chelsio 网卡的系统上的 iscsiadm -m iface 输出可能显示为：

```
default tcp,<empty>,<empty>,<empty>,<empty>
iser iser,<empty>,<empty>,<empty>,<empty>
cxgb3i.00:07:43:05:97:07 cxgb3i,00:07:43:05:97:07,<empty>,<empty>,<empty>
```

也可以以更友好的的方式来显示特定 iface 配置的设置。要这样做，请使用选项-I iface_name。这将以如下格式显示设置：

```
iface.setting = value
```

使用前面的例子，相同 Chelsio 视频卡（即 iscsiadm -m iface -I cxgb3i.00:07:43:05:97:07）的 iface 设置将显示为：

```
# BEGIN RECORD 2.0-871
iface.iscsi_ifacename = cxgb3i.00:07:43:05:97:07
iface.net_ifacename = <empty>
iface.ipaddress = <empty>
iface.hwaddress = 00:07:43:05:97:07
iface.transport_name = cxgb3i
iface.initiatorname = <empty>
# END RECORD
```

23.11.2 软件 iSCSI 的 iface 配置

如前所述，每个与会话绑定的网络对象都需要 **iface** 配置。

要为软件 iSCSI 创建 iface 配置，请运行以下命令：

```
iscsiadm -m iface -I iface_name --op=new
```

这将创建一个指定 iface_name 的新的空 iface 配置。如果已存在有相同 iface_name 的 iface 配置，那么它将被新的空 iface 配置所覆盖。

要配置一个 iface 配置的特定设置，请使用下面的命令：

```
iscsiadm -m iface -I iface_name --op=update -n iface.setting -v hw_address
```

例如，要设置 iface0 的 MAC 地址 (hardware_address) 为 00:0F:1F:92:6B:BF，请运行:

```
iscsiadm -m iface -I iface0 - -op=update -n iface.hwaddress -v 00:0F:1F:92:6B:BF
```

警告：不要使用 default 或 iser 作为 iface 的名称。两个字符串是 iscsiadm 用于保持向后兼容的特殊值。任何手动创建的名为 default 或 iser 的 iface 配置将禁用反向兼容性。

23.11.3 为 iSCSI 卸载配置 iface

默认情况下，iscsiadm 将为每个 Chelsio，Broadcom 和 ServerEngines 端口创建一个 iface 配置。要查看可用的 iface 配置，使用相同的命令在软件 iSCSI 中执行此操作，即 iscsiadm -m iface。

在使用网卡的 iface 进行 iSCSI 卸载之前，请首先设置该设备应该使用的 IP 地址（target_IP[3]）。对于使用 be2iscsi 驱动程序的 ServerEngines 设备（即 ServerEngines iSCSI HBA），IP 地址在 ServerEngines BIOS 设置屏幕上进行配置。

对于 Chelsio 和 Broadcom 设备，配置 IP 地址的过程与其他 iface 设置过程相同。因此，配置 iface 的 IP 地址，请使用：

```
iscsiadm -m iface -I iface_name -o update -n iface.ipaddress -v target_IP
```

例如，要设置一个 Chelsio 卡的 iface IP 地址 (iface 名为 cxgb3i.00:07:43:05:97:07) 为 20.15.0.66，请使用：

```
iscsiadm -m iface -I cxgb3i.00:07:43:05:97:07 -o update -n iface.ipaddress -v 20.15.0.66
```

23.11.4 绑定/解除到门户的 iface

每当 iscsiadm 用于扫描互连时，它都会首先检查 /var/lib/iscsi/ifaces 中每个 iface 配置的 iface.transport 设置。iscsiadm 实用程序然后会将发现的门户绑定到 iface.transport 设置为 tcp 的任何 iface 上。

这样做是由于兼容性的原因。要重载此设置，请使用-I iface_name 来指定将

哪一个门户绑定到一个 iface，如：

```
iscsiadm -m discovery -t st -p target_IP:port -I iface_name -P 13
```

默认情况下，iscsiadm 实用程序不会将任何门户自动绑定到使用卸载的 iface 配置。因为这样的 iface 配置不会使 iface.transport 的设置为 tcp。因此，Chelsio、Broadcom 和 ServerEngines 端口的 iface 配置需要手动绑定到发现的门户上。

它也可以防止一个门户被绑定到任何现有的 iface 上。若要执行此操作， 请使用 default 作为 iface_name，如：

```
iscsiadm -m discovery -t st -p IP:port -I default -P 1
```

要删除目标方和 iface 之间的绑定，请使用：

```
iscsiadm -m node -targetname proper_target_name -I iface0 --op=delete[5]
```

要删除特定 iface 的所有绑定，请使用：

```
iscsiadm -m node -I iface_name --op=delete
```

要删除特定门户的所有绑定（如 Equalogic 目标方），请使用：

```
iscsiadm -m node -p IP:port -I iface_name --op=delete
```

备注：如果 /var/lib/iscsi/iface 中没有定义 iface 配置，并且未使用-I 选项，iscsiadm 将允许网络子系统决定一个特定门户应该使用哪个设备。

23.12 扫描 iSCSI 互连

对于 iSCSI，如果目标方发送一个 iSCSI 异步事件，该事件暗示添加了新的存储，则将自动完成扫描。CMDS TMMDSTM 和 EMC CelerraTM 支持此功能。

然而，如果目标方没有发送一个 iSCSI 异步事件，您需要使用 iscsiadm 实用工具对它们进行手动扫描。然而，在这样做之前，你首先需要检索合理的 --targetname 和--portal 值。如果您的设备型号只支持单一逻辑单元和每个目标的门户，请使用 iscsiadm 向主机发出 sendtargets 命令：

```
iscsiadm -m discovery -t sendtargets -p target_IP:port[3]
```

输出信息将以如下格式显示：

```
target_IP:port,target_portal_group_tag proper_target_name
```

例如，一个 proper_target_name 为 iqn.1992-08.com.netapp:sn.33615311 、target_IP: port 为 10. 15. 85. 19:3260 的目标方，其输出信息可能会显示如下内容：

```
10.15.84.19:3260,2 iqn.1992-08.com.netapp:sn.33615311
10.15.85.19:3260,3 iqn.1992-08.com.netapp:sn.33615311
```

在这个例子中，该目标有两个门户，每个门户使用的 target_ip: port 分别为 10. 15. 84. 19:3260 和 10. 15. 85. 19:3260。

要查看每个会话将使用哪一个 iface 配置，然后添加-P 1 选项。此选项也将以树状格式打印会话信息，如：

```
Target: proper_target_name
Portal: target_IP:port,target_portal_group_tag
Iface Name: iface_name
```

例如，对于 iscsiadm -m discovery -t sendtargets -p 10.15.85.19:3260 -P 1，输出信息可能显示为：

```
Target: iqn.1992-08.com.netapp:sn.33615311
Portal: 10.15.84.19:3260,2
Iface Name: iface2
Portal: 10.15.85.19:3260,3
Iface Name: iface2
```

这意味着，目标方 iqn.1992-08.com.netapp:sn.33615311 将使用 iface2 作为其 iface 配置。

然而，对于某些型号的设备（如：来自 EMC 和 NetApp），单目标方可能有多个逻辑单元和/或门户。在这种情况下，首先要向主机发出 sendtargets 命令来发现目标方上的新门户。然后，使用以下命令重新扫描现有会话：

```
iscsiadm -m session --rescan
```

通过指定会话的 SID 值，您还可以重新扫描一个特定的会话，如： iscsiadm -m session -r SID --rescan[6]

如果您的设备支持多个目标方，您需要向主机发出 sendtargets 命令以便发现每个目标上的新门户。然后，重新扫描现有会话，发现现有会话上的新的逻辑单元（即使用--rescan 选项）。

 重要：用于检索--targetname 和--portal 值的 sendtargets 命令将覆盖/var/lib/iscsi/nodes 数据库中的内容。通过使用/etc/iscsi/iscsid.conf 中的设置，该数据库将被重新填充。然而，如果一个会话目前已被登录并正在使用中，则不会发生这种情况。

为了安全地添加新的目标/门户或删除旧的目标/门户，请分别使用-o new 或 -o delete 选项。例如，欲添加新的目标/门户，而不覆盖/var/lib/iscsi/ nodes，请使用下面的命令：

```
iscsiadm -m discovery -t st -p target_IP -o new
```

要删除目标方在发现过程中没有显示的/var/lib/iscsi/nodes 条目，请使用：

```
iscsiadm -m discovery -t st -p target_IP -o delete
```

您也可以同时执行这两项任务，如：

```
iscsiadm -m discovery -t st -p target_IP -o delete -o new
```

sendtargets 命令将产生以下输出：

```
ip:port,target_portal_group_tag proper_target_name
```

例如，如果一个设备有一个目标方，逻辑单元和门户，以 equallogic-iscsi1 为您的 target_name，输出信息应该类似如下：

```
10.16.41.155:3260,0
iqn.2001-05.com.equallogic:6-8a0900-ac3fe0101-63aff113e344a4a2-dl585-03-1
```

proper_target_name 和 ip:port,target_portal_group_tag 与 “23.2.1 Iscsi API”中同名的值是一致的。

这时，您拥有手动扫描 iSCSI 设备所需的适当的--targetname 和—portal 值。欲完成此操作，请执行以下命令：

```
iscsiadm --mode node --targetname proper_target_name --portal
ip:port,target_portal_group_tag \ --login 7
```

使用我们先前的例子（其中 proper_target_name 是 equallogic-iscsi1），完整的命令是：

```
iscsiadm --mode node --targetname \ iqn.2001-05.com.equallogic:6-8a0900-
ac3fe0101-63aff113e344a4a2-dl585-03-1 \ --portal 10.16.41.155:3260,0 - login[7]
```

23.13 登录 iSCSI 目标方

如 “23.2 iSCSI”所述，为及时发现或登录到目标方，iSCSI 服务必须保持运行。要启动 iSCSI 服务，请运行：

```
service iscsi start
```

执行此命令时，iSCSI 的 init 脚本会自动登录到 node. startup 设置被配置为 automatic 的目标方上。这是对所有目标方 node. startup 的默认值。

为了防止自动登录到目标，请将 node.startup 设置为 manual。欲完成此操作，请执行以下命令：

```
iscsiadm -m node --targetname proper_target_name -p target_IP:port -o update -n
node.startup -v manual
```

删除整个记录也可以防止自动登录。要执行此操作，请运行：

```
iscsiadm -m node --targetname proper_target_name -p target_IP:port -o delete
```

从网络上的 iSCSI 设备自动挂载文件系统，请使用_netdev 选项为/etc/fstab 中的挂载添加一个分区项。例如，要在启动过程中将 iSCSI 设备 sdb 自动挂载到 /mount/iscsi，请添加以下行到/etc/fstab 中：

```
/dev/sdb /mnt/iscsi ext3 _netdev    0 0
```

要手动登录到 iSCSI 目标，请使用下面的命令：

```
iscsiadm -m node --targetname proper_target_name -p target_IP:port –l
```

备注：proper_target_name 和 target_IP:port 引用目标方的全名和 IP 地址/端口的组合。有关详细信息，请参阅第 23.2.1 节，“iSCSI API”和第 23.12 节“扫描 iSCSI 互连”。

23.14 调整在线逻辑单元

大多数情况下，全面调整在线逻辑单元包括两件事情： 调整逻辑单元本身并反映相应多路径中设备大小的变化（如果系统上启用多路径）。

要调整在线逻辑单元，请首先通过您的存储设备阵列管理接口来修改逻辑单元的大小。此过程对因阵列各异而各不相同，因此，更多关于这方面的信息，请咨询存储阵列供应商文档。

备注：为了调整一个在线文件系统，该文件系统不准驻留在一个分区的设备上。

23.14.1 调整光纤通道逻辑单元

修改在线逻辑单元大小后，请重新扫描逻辑单元，以确保系统能够检测到更新后的大小。要对光纤逻辑单元完成此操作，请执行以下命令：

```
echo 1 > /sys/block/sdX/device/rescan
```

提示：要重新扫描使用多路径的系统上的光纤通道逻辑单元，对代表多路径逻辑单元路径的每个 sd 设备（即 sd1，sd2 等），请执行上述命令。为了确定哪些设备是多路径逻辑单元的路径，请使用 multipath -ll，然后，找到与背调整逻辑单元相匹配的项。建议您访问每个条目的 WWID，使其更容易发现与正被调整的逻辑单元相匹配的项。

23.14.2 调整 iSCSI 逻辑单元

修改在线逻辑单元大小后，请重新扫描逻辑单元，以确保系统能够检测到更新后的大小。欲完成对 iSCSI 设备的操作，请执行以下命令：

```
iscsiadm -m node --targetname target_name -R
```

用设备所在目标方得名称来代替 target_name。

备注：您还可以使用下面的命令来重新扫描 iSCSI 逻辑单元：

```
iscsiadm -m node -R -I interface
```

使用被调整逻辑单元(例如，iface0)的相应接口名称来代替 interface。此命令执行两个操作：

1） 它会以与命令 echo "- - -" > / sys/class/scsi_host/host/scan 同样的方式来扫描新的设备（参见 “23.2 扫描 iSCSI 互连”）。

2） 它以与命令 echo 1 > /sys/ block/sdX/device/rescan 同样的方式重新扫描新的/被修改的逻辑单元 。请注意，这个命令与重新扫描光纤通道逻辑单元使用的命令相同。

23.14.3 更新多路径设备的大小

如果您的系统上启用多路径，您还需要将逻辑单元的大小变化反映到逻辑单元相应的多路径设备上（在调整逻辑单元大小后）。要做到这一点，首先使用 service multipathd status 来确保 multipathd 正在运行一旦验证了 multipathd 正在运行，请运行以下命令：

```
multipathd -k"resize map multipath_device"
```

multipath_device 变量是/dev/ mapper 中设备对应的多路径项。根据多路径在您系统上的设置方式，multipath_device 可以是两种格式其一：

1） mpathX，其中 X 是您设备对应的条目（例如，mpath0）

2） 一个 WWID；例如，3600508b400105e210000900000490000

要确定哪个多路径条目对应您调整大小后的逻辑单元，请运行 multipath -ll。这显示系统中所有现存多路径挂载点的列表，以及其相应设备的主要和次要号码。

提示：如果存在到 multipath_device 的任何命令队列，不要使用 multipathd -k"resize map multipath_device" 。也就是说，当 no_path_retry 参数为 "queue"并且没有到设备的活动路径时，请不使用此命令。

如果您的系统正在使用中标麒麟可信操作系统 V6.0V6.0，您将需要执行以下步骤来指示 multipathd 后台程序识别出（并适应）您对被调整逻辑单元所做出的更改：

调整相应的多路径设备

1） 为多路径设备转储设备映射表：请使用：

```
dmsetup table multipath_device
```

2） 保存转储设备映射表为 table_name。此表随后将被重新加载和编辑。

3） 检查设备映射表。注意每一行的前两个数字分别对应于磁盘的开始和末尾扇区。

4） 挂起设备映射目标方：

```
dmsetup suspend multipath_device
```

5） 打开以前保存（即 table_name）的设备映射表。改变第二个数字（即磁

盘的末尾扇区），以反映磁盘的 512 字节扇区的新数量。例如，如果新磁盘的大小是 2GB，将第二个号码改为 4194304。

6） 重新加载修改后的设备映射表。

```
dmsetup reload multipath_device table_name
```

7） 恢复设备映射目标方：

```
dmsetup resume multipath_device
```

23.15 通过 rescanscsi-bus.sh 添加/删除逻辑单元

sg3_utils 软件包提供了 rescan-scsi-bus.sh 脚本，它可以按照需要自动更新主机的逻辑单元配置（设备已添加到系统后）。rescan-scsi-bus.sh 脚本也可以在所支持设备上执行 issue_lip。欲了解更多有关如何使用此脚本的详细信息，请参阅 rescan-scsi-bus.sh --help。

要安装 sg3_utils 软件包，请运行

```
yum install sg3_utils。
```

使用 rescan-scsi-bus.sh 的已知问题

当使用 rescan-scsi-bus.sh 脚本时，需要注意以下已知问题：

1） 为了使 rescan-scsi-bus.sh 正常工作，LUNG 必须是第一个映射逻辑单元。如果它是 LUNG，rescan-scsi-bus.sh 只能探测到的第一个映射逻辑单元。rescan-scsi-bus.sh 将无法扫描任何其他的逻辑单元，除非它检测到第一个映射逻辑单元，即使你使用--nooptscan 选项。

2） 一个竞争条件要求， 如果逻辑单元首次映射，rescan-scsi-bus.sh 应该运行两次。第一次扫描期间，rescan-scsi-bus.sh 只会添加 LUNG， 所有其他逻辑单元在第二次扫描时添加。

3） 使用- remove 选项时， rescan-scsi-bus.sh 脚本的错误将导致不能正确地识别逻辑单元大小的变化。

4） rescan-scsi-bus.sh 脚本不能识别出 iSCSI 逻辑单元的删除。

23.16 修改链路损耗性能

本节介绍如何修改使用光纤通道或 iSCSI 协议设备的链接损耗性能。

23.16.1 光纤通道

如果一个驱动程序执行了传输 dev_loss_tmo 回调，当检测到传输问题时，通过链接的设备访问将被阻塞。为了验证设备是否被堵塞，请运行以下命令：

```
cat /sys/block/device/device/state
```

如果设备被阻塞，此命令将返回到 blocked。如果设备正常运行，该命令将返回 running。

确定远程端口的状态

1） 要确定一个远程端口的状态，运行以下命令：

```
cat /sys/class/fc_remote_port/rport-H:B:R/port_state
```

2） 远程端口（以及通过它进行访问的设备）受阻时，此命令将返回到 Blocked 。如果远程端口运行正常，该命令将返回到 Online。

3） 如果 dev_loss_tmo 秒这个问题得不到解决，rport 和设备将被解锁并且该设备上运行的所有 I / O（连同发送到该设备的任何新 I / O）都将失败。

更改 dev_loss_tmo

更改文件中 dev_loss_tmo 值，期望值的 echo。如，若设置 dev_loss_tmo 为 30 秒，请运行：

```
echo 30 > /sys/class/fc_remote_port/rport-H:B:R/dev_loss_tmo
```

关于 dev_loss_tmo 的更多信息， 请参阅 “23 光纤通道 API”。

当一个设备被阻塞时，光纤通道类将按现状离开该设备；即/dev/sdx 将保留当前的/dev/sdx。这是因为 dev_loss_tmo 过期了。如果链接问题在后期被修复，操作将继续使用相同的 SCSI 设备和设备节点名称。

光纤通道： remove_on_dev_loss

当链接标记为坏时（即 dev_loss_tmo 秒后过期），如果您希望设备在 SCSI 层中删除，则可以使用 scsi_transport_fc 模块参数 remove_on_dev_loss。当一个设备在 SCSI 层被删除，而 remove_on_dev_loss 正在实行中时，一旦所有传输问题得到纠正后，设备将被添加回来。

警告：不推荐使用 remove_on_dev_loss ，因为删除 SCSI 层设备不会自动卸载该设备的任何文件系统。当被删除设备的文件系统仍保持已挂载状态时，可能无法从多路径或 RAID 设备正确删除该设备。

如果上层不是热插拔系统，可能出现更多的问题。这是因为上层可能仍然保持对设备被删除之前状态的引用。再次添加设备时，可能会导致意外的行为。

23.16.2 使用 dm-multipath 的 iSCSI 设置

如果实施 dm-multipath，最好是设置 iSCSI 计时器将命令立即推迟到多路径

层。要对此进行配置，在 device { in /etc/ multipath.conf 下嵌套下面的命令行：

```
features "1 queue_if_no_path"
```

如果所有路径都在 dm-multipath 层失败，这将确保 I / O 错误重试和排列。

您可能需要进一步调整 iSCSI 定时器以便对于更好地监测您的 SAN 问题。您可以配置的可用 iSCSI 定时器包括 NOP-Out 间隔/超时 和 replacement_timeout，这将在下面的章节中讨论。

23.16.2.1 NOP-Out 间隔/超时

为了帮助监测 SAN 问题，iSCSI 层向每个目标方发送 NOP- OUT 请求。如果一个 NOP 请求超时，iSCSI 层将通作出反应，即，使任何运行中的命令执行失败并指示 SCSI 层在可能的情况下重新排列这些命令。

当正在使用 dm-multipath 时 ，SCSI 层将使正在运行的命令执行失败，并将它们推迟到多路径层。然后，多路径层在另一个路径上重试那些命令。如果现在没有使用 dmmultipath ，这些命令在完全失败之前将被重试五次。

NOP - OUT 请求之间的时间间隔是 10 秒。若要对此作出调整，请打开 /etc/iscsi/ iscsid.conf 并编辑以下命令行：

```
node.conn[0].timeo.noop_out_interval = [interval value]
```

一旦设定，iSCSI 层每[interval value]秒将向每个目标方发送 NOP- OUT 请求。

默认情况下，NOP-Out 请求超时在 10 seconds[9] 之内。若要对此作出调整，请打开 /etc/iscsi/ iscsid.conf 并编辑以下命令行：

```
node.conn[0].timeo.noop_out_timeout = [timeout value]
```

这个命令设置 iSCSI 层 在[timeout value] 秒之后将一个 NOP-OUT 请求设为超时。

SCSI 错误处理程序

如果 SCSI 错误处理程序正在运行，当一个 NOP-Out 请求在该路径上超时的时候，该路径上正在运行的命令不会立即失败。相反，replacement_timeout 秒后，这些命令将失败。关于 replacement_timeout 的更多信息， 请参考 23.16.2.2 节 replacement_timeout。

若要验证是否 SCSI 错误处理程序正在运行，请运行如下命令：

```
iscsiadm -m session -P 3
```

23.16.2.2 replacement_timeout

replacement_timeout 控制 iSCSI 层应该等待超时路径/会话在运行在其上的所有命令失败之前重建自身的时间。 默认的 replacement_timeout 值是 120 秒。

若要对 replacement_timeout 作出调整，请打开 /etc/iscsi/ iscsid.conf 并编辑以下命令行：

```
node.session.timeo.replacement_timeout = [replacement_timeout]
```

/etc/multipath.conf 中的 1 queue_if_no_path 选项设置 iSCSI 计时器将命令立即推迟到多路径层（参考 23.16.2 节 dmmultipath iSCSI 设置）。此设置可防止 I / O 错误传播到应用程序，正因为如此，您可以设置 replacement_timeout 为 15-20 秒。

通过配置较低的 replacement_timeout，当 iSCSI 层试图重新建立失败的路径/会话时，I / O 被迅速发送到一个新的路径并被执行（NOP-Out 超时情况下）。如果所有路径都超时，那么多路径和设备映射层将根据 /etc/multipath.conf 设置而不是/etc/iscsi/iscsid.conf 设置对 I / O 进行内部排列。

重要：无论您所考虑的是故障转移的速度或安全性，推荐的 replacement_timeout 值将取决于其他因素。这些因素包括网络，目标方和系统负载。因此，将它应用到一个关键任务系统之前，建议您彻底测试 replacements_timeout 的任何新配置 。

23.16.3 iSCSI Root

当直接通过 iSCSI 磁盘访问根分区时，应设置 iSCSI 定时器以便于 iSCSI 层有几次机会尝试重新建立一个路径/会话。此外，命令不应该被迅速重新排队列到 SCSI 层。这与执行 dmmultipath 时所应该做的相反。

首先，NOP-Outs 应该被禁用。您可以通过设置 NOP-Out 间隔和超时为零来执行此操作。若要对此作出设置，请打开 /etc/iscsi/ iscsid.conf 并编辑以下命令行：

```
node.conn[0].timeo.noop_out_interval = 0
node.conn[0].timeo.noop_out_timeout = 0
```

与此一致，replacement_timeout 应被设置为一个较大的数字。这将指示该系统等待很长一段时间来重新建立路径/会话。若要对 replacement_timeout 作出调整，请打开 /etc/iscsi/ iscsid.conf 并编辑以下命令行：

```
node.session.timeo.replacement_timeout = replacement_timeout
```

配置完 /etc/iscsi/iscsid.conf 后，您必须重新发现受影响存储。这将允许系统

加载和使用/etc/iscsi/iscsid.conf 中的任何新值。更多关于如何发现 iSCSI 设备的信息，请参阅“扫描 iSCSI 互连”。

为特定会话配置超时

您还可以为一个特定的会话配置超时，并且使他们表现非持久性（而不是使用/etc/iscsi/iscsid.conf）。欲完成此操作，请执行以下命令（相应地替换变量）：

```
iscsiadm -m node -T target_name -p target_IP:port -o update -n
node.session.timeo.replacement_timeout -v $timeout_value
```

 重要：建议为涉及根分区访问的 iSCSI 会话提供此处描述的配置。对于涉及访问其他类型存储的 iSCSI 会话（即，在使用 dm-multipath 的系统中），请参阅第 23.16.2 节，“dm-multipath iSCSI 设置”。

23.17 控制 SCSI 命令定时器和设备状态

Linux SCSI 层为每个命令设置一个定时器。此计时器到期时，SCSI 层将静止主机总线适配器（HBA），并等待所有未完成的命令或者超时或者完成。随后，SCSI 层将激活驱动程序的错误处理程序。

错误处理程序被触发时，它会按顺序尝试以下操作（直到其中一个被成功执行）：

1） 中止命令。

2） 重置设备。

3） 重置总线。

4） 重置主机。

如果所有这些操作都失败，该设备将被设置为 offline 脱机状态。当发生这种情况时，该设备的所有 I / O 都将失败，直到问题被解决并且用户设置此设备为 running。

然而，如果设备使用光纤通道协议而且 rport 被阻塞时，这个处理过程是不同的，。在这种情况下，激活错误处理程序之前，驱动程序等待几秒钟，从而使 rport 再次上线。这可以防止设备由于临时传输问题而离线。

设备状态

要显示设备的状态，请使用：

```
cat /sys/block/device-name/device/state
```

要设置要设置一个设备为 running 状态， 请使用：

```
echo running > /sys/block/device-name/device/state
```

命令计时器

要控制命令计时器， 您可以写入到 /sys/block/device-name/device/timeout。要做到这一点，请运行：

```
echo value /sys/block/device-name/device/timeout
```

此处， value 是您想实施的超时值(几秒内) 。

23.18 故障排除

本节提供在线存储重新配置过程中用户常见问题的解决方案。

逻辑单元删除状态没有体现在主机上。

当一个逻辑单元在一个配置文件管理器中被删除，更改没有体现在主机上。在这种情况下，当使用 dm-multipath 时，LVM 命令将无限期挂起，因为逻辑单元现在已经变为过时。

为了解决这个问题，请执行以下步骤：

解决过时的逻辑单元

1） 确定 /etc/lvm/cache/.cache 中的哪个 mpath 链接条目是针对过时逻辑单元的。欲完成此操作，请执行以下命令：

```
ls -l /dev/mpath | grep stale-logical-unit
```

2） 例如， 如果 stale-logical-unit 是 3600d0230003414f30000203a7bc41a00，可能出现以下结果：

```
lrwxrwxrwx 1 root root 7 Aug 2 10:33 /3600d0230003414f30000203a7bc41a00 -> ../dm-4
lrwxrwxrwx 1 root root 7 Aug 2 10:33 /3600d0230003414f30000203a7bc41a00p1 -> ../dm-5
```

这意味着，3600d0230003414f30000203a7bc41a00 被映射到两个 mpath 链接：dm-4 和 dm-5。

3） 接下来， 打开 /etc/lvm/cache/.cache。删除所有包含 stale-logical-unit 的命令行，以及 stale-logical-unit 映射到的 mpath 链接。

使用上一步中的例子，您需要删除的命令行，如下：

```
/dev/dm-4
/dev/dm-5
/dev/mapper/3600d0230003414f30000203a7bc41a00
/dev/mapper/3600d0230003414f30000203a7bc41a00p1
/dev/mpath/3600d0230003414f30000203a7bc41a00
/dev/mpath/3600d0230003414f30000203a7bc41a00p1
```

24. 设备映射多路径管理和虚拟存储

24.1 虚拟存储

中标麒麟可信操作系统 V6.0 支持下列文件系统/虚拟存储的在线存储方法：

1） 光纤通道

2） iSCSI

3） NFS

4） GFS2

中标麒麟可信操作系统 V6.0V6.0 的虚拟技术使用 libvirt 来管理虚拟实例。libvirt 实用程序使用存储池（storage pools）的概念来管理虚拟化客户的存储。存储池是可以划分成较小的卷或直接分配给客户的存储体。存储池卷可以分配给虚拟客户。可用的存储池分为两类：

1） 本地存储池

本地存储包括存储设备，直接附属于主机的文件或目录。本地存储包括本地目录，直接附属磁盘和 LVM 卷组。

2） 网络（共享）存储池

网络存储包括使用标准协议的网络共享存储设备。网络存储包括使用光纤通道 iSCSI，NFS GFS2 和 SCSI RDMA 协议的共享存储设备。网络存储是在主机之间迁移虚拟客户的必要条件。

24.2 DM-Multipath

设备映射多路径管理（DM-Multipath）功能，使您可以将服务器节点和存储阵列之间的多个 I / O 路径配置到一个设备上。这些 I ／ O 路径可以是包括独立电缆，开关和控制器的物理 SAN 连接。多重路径聚合 I ／ O 路径，从而创建了一个包括聚合路径的新设备。

使用 DM-Multipath 主要出于以下几个原因：

1） 冗余

DM-Multipath 可以提供主动/被动配置中的故障转移。在主动/被动配置中，只有一半的路径在任何时间都用于 I / O。如果 I / O 路径的任何一个元素（电缆，开关或控制器）失败，DM-Multipath 将切换到备用路径。

2） 改进的性能

DM-Multipath 可以在主动/主动模式下进行配置，其中 I ／ O 以轮流的方式遍布在路径上。在某些配置中，DM-Multipath 可以检测到 I ／ O 路径上的负载并且动态实现负载平衡。

附录A 词汇表

此术语表定义了整个存储管理指南 中与文件系统和存储有关的常用术语。

碎片整理（Defragmentation）	重组文件的数据块，使它们在磁盘上占有更为连续的物理空间。
延迟分配（Delayed Allocatin）	当数据被刷新到磁盘时而不是在写入磁盘时选择磁盘位置的行为因为空间分配器并不经常被调用而且需求较大，这通常可以导致更有效的分配方式。
扩展属性（Extended Attributes）	可能与文件关联的名称//值元数据对。
区间（Extent）	一个文件分配单元，以偏移量，长度对形式存储在文件的元数据中。一个区间记录可以描述一个文件中许多连续块。
文件系统修复（fsck）	一个验证和修复文件系统元数据一致性的方法。非日志文件系统崩溃后，或硬件故障或内核错误后可能需要此功能。
碎片（Fragmentation）	由于文件内连续的逻辑偏移，一个文件的数据块没有被分配到连续的物理（磁盘）位置上。 由于磁盘寻道时间，文件碎片在某些情况下可能会导致性能下降。
元数据日志（Metadata Journaling）	确保文件系统的元数据即使在系统崩溃后也保持一致的方法。元数据日志可以采取不同的形式，但在每种情况下，日记或记录在系统崩溃后可以被重建，只将一致的事务变化写入到磁盘。
持久性预分配（Persistent Preallocation）	一种文件分配类型，选择磁盘位置并且标记这些块为已使用，而不考虑何时或是否写入这些块。直到这些数据写入到块，读操作才将返回 0。可用 fallocate()glibc 函数执行预分配。
POSIX 访问控制列表（ACLs）	附加到文件上的元数据，这些元数据允许更细粒度的访问（ACL）控制。ACL 经常被作为一种特殊类型的扩展属性。
限额（Quota）	文件系统中由管理员设置的单个用户和组的块或索引节点的使用限制。
条带单元（Stripe Unit）	有时也被称为步幅或数据块大小。条带单元是指在移动到下一个单元之前，写入到条带

	化存储的一个组件的数据量。以 字节数或文件系统块单元数对其进行指定。
条带宽度(Stripe Width)	条带存储体中单个数据条带单元的数目(不包括校验)。根据所使用的管理工具，可以字节数或文件系统的块单元数，或者条带单元的倍数对其进行指定。
条带化分配(Stripe-aware allocation)	一种空间分配器行为，其中分配和 I / O 与底层条带存储完全一致。 这也取决于 mkfs 时间可用的条带信息。有序的 I / O 分配可避免底层存储低效的读 - 改 - 写 循环。
写屏障（Write Barriers）	加强具有不稳定写缓存的存储设备上一致 I / O 有序性的方法。必须使用屏障来保证：断电后，不会因存储硬件以不同于操作系统所要求的顺序从其不稳定写入缓存中写出块而违反元数据日志所需的有序性保证。 另请参阅元数据日志。