Sun StorEdge™ SAM-FS
存储和归档管理指南

Version 4, Update 4

Sun Microsystems, Inc.
www.sun.com

文件号码 819-4782-10
2005 年 12 月，修订版 A

请将有关本文档的意见和建议提交至： http://www.sun.com/hwdocs/feedback

请回收

目录

表

前言

本手册《Sun StorEdge™ SAM-FS 存储和归档管理指南》介绍了 Sun StorEdge SAM-FS 版本 4 更新 4 (4U4) 发行版中所支持的存储和归档管理软件。Sun StorEdge SAM-FS 软件可以自动将文件从联机磁盘复制到归档介质中。归档介质可由联机磁盘或可移除介质卡盒组成。

以下 Sun Solaris™ 操作系统 (Operating System, OS) 平台支持 Sun StorEdge SAM-FS 4U4 发行版：

- Solaris 9, update 3, 4/03
- Solaris 10

本手册的读者对象是负责配置和维护 Sun StorEdge SAM-FS 软件的系统管理员。并假定系统管理员非常熟悉 Solaris OS 过程，包括创建帐户、执行系统备份和其他基本的 Solaris 系统管理任务。

注 – 您可以购买 Sun StorEdge QFS 和 Sun StorEdge SAM-FS 软件的许可证，以便将 Sun StorEdge QFS 文件系统和 Sun StorEdge SAM-FS 软件中的存储及归档管理器配合运行。此类系统称之为 *SAM-QFS*。

本手册并未列出 SAM-QFS 配置，除非必须明确以避免混淆。在本手册中谈及存储及归档管理时，您可以假定对 Sun StorEdge SAM-FS 的参考同样适用于各种 SAM-QFS 配置。同样，当谈及文件系统设计和性能时，您可以假定对 Sun StorEdge QFS 的参考同样适用于各种 SAM-QFS 配置。

本书的结构

本书包括以下章节：

- 第 1 章提供概述信息。
- 第 2 章介绍基本操作。本章所述的信息适用于大多数自动化库和手动载入的设备。
- 第 3 章介绍归档过程。
- 第 4 章介绍释放过程。
- 第 5 章介绍登台过程。
- 第 6 章介绍回收过程。
- 第 7 章介绍如何使用 Sun SAM-Remote 软件。
- 第 8 章介绍有关 Sun StorEdge SAM-FS 操作的高级主题。
- 附录 A 介绍如何依据各类库的专用操作说明来管理相应库中的卡盒。这一章还介绍了这些库及其相应的基本操作步骤。

词汇表定义了本手册及其他 Sun StorEdge QFS 和 Sun StorEdge SAM-FS 文档中所用到的术语。

使用 UNIX 命令

本文档不会介绍基本的 UNIX® 命令和操作过程，如关闭系统、启动系统和配置设备等。欲获知此类信息，请参阅以下文档：

- 系统附带的软件文档
- Solaris™ 操作系统的有关文档，其 URL 如下：

  `http://docs.sun.com`

shell 提示符

表 P-1 列出了本手册中使用的 shell 提示符。

表 P-1 shell 提示符

shell	提示符
C shell	*machine-name*%
C shell 超级用户	*machine-name*#
Bourne shell 和 Korn shell	$
Bourne shell 和 Korn shell 超级用户	#

印刷约定

表 P-2 列出了本手册中使用的印刷约定。

表 P-2 印刷约定

字体或符号	含义	示例
AaBbCc123	命令、文件和目录的名称；计算机屏幕输出	编辑 .login 文件。 使用 ls -a 列出所有文件。 % You have mail.
AaBbCc123	用户键入的内容，与计算机屏幕输出的显示不同	% **su** Password:
AaBbCc123	保留未译的新词或术语以及要强调的词。要使用实名或值替换的命令行变量。	这些称为 *class* 选项。 要删除文件，请键入 rm *filename*。
新词术语强调	新词或术语以及要强调的词。	您**必须**成为超级用户才能执行此操作。
《书名》	书名	阅读 《用户指南》 的第 6 章。

表 **P-2** 印刷约定 （续）

字体或符号	含义	示例
[ ]	在命令语句内，方括号内的参数表示可选参数。	`scmadm [-d sec] [-r n[:n][,n]...] [-z]`
{ *arg* | *arg* }	在命令语句内，大括号和竖线表示必须指定其中一个参数。	`sndradm -b { phost | shost }`
\	命令行末尾的反斜杠 (\) 表示此命令续接下一行。	`atm90 /dev/md/rdsk/d5 \ /dev/md/rdsk/d1`

相关文档

本手册是介绍 Sun StorEdge QFS 和 Sun StorEdge SAM-FS 软件产品操作文档系列的一部分。表 P-3 列出了这些 4U4 版产品的完整文档系列。

表 **P-3** 相关文档

书名	文件号码
《Sun StorEdge SAM-FS 文件系统配置和管理指南》	819-4807-10
《Sun StorEdge SAM-FS Installation and Upgrade Guide》	819-4776-10
《Sun StorEdge SAM-FS 故障排除指南》	819-4787-10
《Sun StorEdge QFS 配置和管理指南》	819-4797-10
《Sun StorEdge QFS 安装和升级指南》	819-4792-10
《Sun StorEdge QFS 和 Sun StorEdge SAM-FS 4.4 发行说明》	819-4802-10

访问 Sun 联机文档

Sun StorEdge QFS 和 Sun StorEdge SAM-FS 软件销售套件中包括了这些产品文档的 PDF 文件。用户可从以下位置查看这些 PDF 文件：

- Sun Network Storage 文档网站

此网站包含许多存储软件产品的文档。

a. 要访问该 Web 站点，请输入以下 URL：

```
http://www.sun.com/products-n-solutions/hardware/docs/
Software/Storage_Software
```

屏幕上会出现 "Storage Software" 页面。

b. 从以下列表中单击适当的链接：

- Sun StorEdge QFS 软件
- Sun StorEdge SAM-FS 软件

- `docs.sun.com`

  此 Web 站点包含 Solaris 和其他多个 Sun 软件产品的文档。

a. 要访问该 Web 站点，请输入以下 URL：

```
http://docs.sun.com
```

屏幕上会出现 `docs.sun.com` 页面。

b. 通过在搜索框中搜索以下项目之一来查找适用的产品文档：

- Sun StorEdge QFS
- Sun StorEdge SAM-FS

第三方 Web 站点

Sun 对本文档中提到的第三方 Web 站点的可用性不承担任何责任。对于此类站点或资源中的（或通过它们获得的）任何内容、广告、产品或其他资料， Sun 并不表示认可，也不承担任何责任。对于因使用或依靠此类站点或资源中的（或通过它们获得的）任何内容、产品或服务而造成的或连带产生的实际或名义损坏或损失， Sun 概不负责，也不承担任何责任。

联系 Sun 技术支持

如果您遇到通过本文档无法解决的技术问题，请访问以下网址：

```
http://www.sun.com/service/contacting
```

使用许可

有关获取 Sun StorEdge SAM-FS 软件许可证的信息，请与 Sun 销售代表或授权的服务供应商 (Authorized Service Provider, ASP) 联系。

安装帮助

要获得安装和配置服务，请拨打 1-800-USA4SUN 联系 Sun 企业服务部门，或联系当地的企业服务销售代表。

Sun 欢迎您提出意见

Sun 致力于提高其文档的质量，并十分乐意收到您的意见和建议。您可以通过以下网址提交您的意见和建议：

`http://www.sun.com/hwdocs/feedback`

请在您的反馈信息中包含本文档的书名和文件号码：《Sun StorEdge SAM-FS 存储和归档管理指南》，文件号码 819-4782-10。

第 1 章

概述

Sun StorEdge SAM-FS 环境提供了具有存储、归档管理以及检索功能的可配置文件系统。Sun StorEdge SAM-FS 软件对文件进行归档的方法是，将文件从联机磁盘高速缓存复制到归档介质。归档介质可以是另一文件系统中的磁盘分片，也可以是自动或手动载入的存储设备中的可移除磁带或磁光盘卡盒。另外， Sun StorEdge SAM-FS 软件可自动将联机磁盘空间大小维持在站点指定的使用阈值上。它可以释放与已归档的文件数据相关联的磁盘空间，并在需要时将文件恢复到联机磁盘。

本章从技术层面对 Sun StorEdge SAM-FS 组件进行概述。它包括下列主题：

性能

Sun StorEdge SAM-FS 环境存储和归档管理软件可与文件系统 （如 Sun StorEdge QFS）配合运行。这种文件系统是位于服务器磁盘高速缓存中的高性能 UNIX 文件系统。有关文件系统自身的更多信息，请参见 《Sun StorEdge QFS 配置和管理指南》。

Sun StorEdge SAM-FS 环境中还包含以下其他的组件：

- 归档程序自动将联机磁盘高速缓存中的文件复制到归档介质中。归档介质可以由联机磁盘文件组成，也可以由多个可移除的介质卡盒组成。默认情况下，归档程序会自动为 Sun StorEdge SAM-FS 文件系统中的所有文件创建一个归档副本，并将此归档副本写入归档介质。您可以配置归档程序，使其在不同归档介质上最多可创建 4 份归档副本。如果文件被分成几段，则每段都会被视为一个文件，且单独进行归档。当基于磁盘的文件符合站点定义的一套选择标准后，归档进程将会启动。

  有关归档程序的更多信息，请参见第 31 页 “归档”。有关分段文件的更多信息，请参见第 194 页 “使用分段文件”。

- 释放程序通过释放那些符合条件的已归档文件占用的磁盘块，自动将文件系统的联机磁盘高速缓存大小维持在站点指定的使用百分比阈值上。

  释放是指释放由已归档文件的数据占用的主 （磁盘）存储空间的过程。它使用两个阈值 （用磁盘总空间的百分比表示）来管理联机磁盘高速缓存的可用空间。这两个阈值分别是上限和下限。当联机磁盘占用空间超出上限时，系统会自动开始释放那些符合条件的已归档文件占用的磁盘空间。当联机磁盘占用空间到达下限时，系统会自动停止释放由已归档文件的数据占用的磁盘空间。选择释放文件的依据是文件的大小和文件在联机磁盘中的时间。另外，文件的第一部分可保留在磁盘上，以加快访问速度和掩饰登台过程的延迟。如果某个文件被分段归档，则可以单独释放该文件的各个部分。有关释放程序的更多信息，请参见第 101 页 “释放”。

- 登台程序将文件数据恢复到磁盘高速缓存中。当用户或进程请求那些已从磁盘高速缓存中释放的文件数据时，登台程序会自动将文件数据复制回联机磁盘高速缓存中。

  如果某个文件的数据块已被释放，则在系统访问该文件时，登台程序会自动将该文件或文件段的数据重新登台到联机磁盘高速缓存中。一旦开始登台操作，读取操作也将随即开始，从而使应用程序可以立即使用文件，而不必等到整个文件完全登台到磁盘高速缓存中。

  Sun StorEdge SAM-FS 软件可自动处理登台请求错误。如果返回登台错误，则系统会尝试查找文件的下一个可用归档副本。系统可自动处理的登台错误包括介质错误、介质不可用、自动化库不可用以及其他错误。有关登台的更多信息，请参见第 117 页 “登台”。

- 回收程序清除包含已过期归档副本的归档卷，以使这些卷可重新使用。

  一旦用户修改了文件，归档介质上与这些文件的旧版本相关联的归档副本将被视为 *expired*。这些副本不再有用，因此可以从系统中清除。回收程序可以识别那些其中绝大部分是过期归档副本的归档卷，并将这些卷中的非过期副本移动到其他卷中进行保存。

  如果可移除介质卷中仅包含过期副本，则可以执行以下某一操作：

  - 重新标记该卷，以便可立即重用。
  - 将该卷的内容导出到离站存储设备中，作为文件变更的历史记录。可使用标准的 UNIX 实用程序从过期的归档副本中恢复文件的以前版本。

  由于回收过程与最终用户的数据文件有关，因此它对最终用户是完全透明的。有关回收的更多信息，请参见第 131 页 “回收”。

存储设备

Sun StorEdge SAM-FS 环境可支持的磁带存储设备和磁光设备种类非常多。Sun StorEdge SAM-FS 所支持的自动化库可根据其与环境连接的方式，分为以下几组：

- 直接连接。直接连接的自动化库使用小型计算机系统接口 (Small Computer System Interface, SCSI) 直接连接到主机系统。这种连接可以是直接连接，也可以是光纤通道连接。例如，Sun StorEdge 库使用直接连接方式。Sun StorEdge SAM-FS 系统直接使用自动化库的 SCSI 标准控制这些库。
- 网络连接。 Sun StorEdge SAM-FS 软件可配置为库的主机系统的客户机。网络连接自动化库包括 StorageTek 库、 ADIC/Grau 库、 IBM 库和 Sony 库等。这些库使用供应商提供的软件包。这时， Sun StorEdge SAM-FS 软件使用自动化库的专用守护进程，与供应商软件进行接口。

表 1-1 列出了不同自动化库专用的守护进程。

表 1-1 自动化库守护进程

守护进程	描述
`sam-robotsd`	监视传输器控制守护进程的执行情况。 `sam-robotsd` 守护进程由 `sam-amld` 守护进程自动启动。
`sam-genericd`	控制直接连接的库和介质更换器。还通过 DAS 接口控制 ADIC 库。
`sam-stkd`	通过 ACSAPI 接口控制 StorageTek 介质更换器。
`sam-ibm3494d`	通过 `lmcpd` 接口控制 IBM 3494 磁带库。
`sam-sonyd`	通过 DZC-8000S 接口控制 Sony 网络连接自动化库。

有关受支持的存储设备列表，请与 Sun Microsystems 销售代表或授权的服务供应商 (Authorized Service Provider, ASP) 联系。

Sun StorEdge SAM-FS 环境中所管理的各设备之间的关系，是在主配置文件 `/etc/opt/SUNWsamfs/mcf` 中定义的。`mcf` 文件指定了 Sun StorEdge SAM-FS 环境中所包括的可移除介质设备、自动化库和文件系统。在 `mcf` 文件中，每一个设备均分配有唯一的设备标识。`mcf` 文件中的条目还可定义手动安装的归档设备和自动化库目录文件。

系统将尽可能使用标准的 Solaris 磁盘和磁带设备驱动程序。对于 Solaris 操作系统 (OS) 不可直接支持的设备 （例如某些特定的库和光盘设备）， Sun StorEdge SAM-FS 软件包中提供了此类设备的专用驱动程序。

File System Manager

File System Manager 软件是基于浏览器的图形用户界面，用户可使用它从一个中心位置配置和控制一个或多个 Sun StorEdge QFS 或 Sun StorEdge SAM-QFS 服务器。您可使用所在网络中任何一台主机上的 Web 浏览器访问这个中心位置。

该软件的目标是为执行与 Sun StorEdge QFS 或 Sun StorEdge SAM-QFS 服务器相关的最常用的任务提供一种相对简单的操作方式。要对服务器进行进一步的配置或管理，请使用该服务器的命令行界面、脚本、配置文件等。有关更多信息以及安装 File System Manager 的指导，请参见 《Sun StorEdge SAM-FS 安装和升级指南》。

在安装 File System Manager 后，您可以使用两个可能的用户名 （`samadmin` 和 `samuser`）以及两个不同的角色 （`SAMadmin` 或 `no role`）登录这个软件。您使用 File System Manager 所能执行的任务，因您登录时使用的用户名和角色而异。这些差异是：

- 如果以 `samadmin` 身份登录，则可以从上述两种角色中任选一种角色。
  - `SAMadmin` 角色可赋予您完全的管理员权限，您能够对 Sun StorEdge QFS 环境中的设备进行配置、监视、控制和重新配置。

    只有 Sun StorEdge QFS 管理员才应使用 `SAMadmin` 角色登录。其他所有用户应该以 `samuser` 身份登录。
  - 如果以 `no role` 角色登录，则您只能对环境进行监视，而无法对环境进行任何更改或重新配置。
- 如果以 `samuser` 身份登录，则您只能对环境进行监视，而无法对环境进行任何更改或重新配置。

至于在系统管理方面，请记住，在安装了 File System Manager 的服务器上， Solaris 操作系统的超级用户不必是 File System Manager 的管理员。只有 `samadmin` 具有 File System Manager 应用程序的管理员权限。而超级用户则是管理站的管理员。

▼ 调用 File System Manager

执行本过程将 File System Manager 载入浏览器。

1. **登录到管理站 Web 服务器。**
2. **通过 Web 浏览器调用 File System Manager 软件。**

   输入以下 URL：

```
https://hostname:6789
```

其中的 *hostname*，用于指定主机名称。如果除指定主机名外，还需要指定域名，请按以下格式指定 *hostname*：*hostname*.*domainname*。

请注意，此 URL 的开始部分为 `https`，而不是 `http`。此后，将会显示 Sun Java Web Console 登录屏幕。

3. **在 `User Name` 提示符下，输入 `samadmin`。**
4. **在 `Password` 提示符下，输入在安装 File System Manager 软件时所选择的密码。**
5. **单击 `SAMadmin` 角色。**

   只有 Sun StorEdge SAM-FS 管理员才应使用 `SAMadmin` 角色登录。
6. **在提示输入 `Role Password` 时，输入在步骤 4 中输入的密码。**
7. **单击 `Log In`。**
8. **单击 `File System Manager 2.0`。**

   您现在已登录到 File System Manager。

创建附加的管理员和用户帐户

在完成 File System Manager 的初始配置后，您可以随时创建附加的管理员和 guest 帐户。 guest 帐户是管理站的本地帐户。

安装 File System Manager 后，File System Manager 将创建以下两个 Solaris 操作系统 (OS) 登录帐户和以下角色：

- 帐户：`samadmin`、 `samuser`
- 角色：`SAMadmin`

用户帐户 `samadmin` 分配给 `SAMadmin` 角色。该用户具有管理 File System Manager、Sun StorEdge QFS、 Sun StorEdge SAM-FS 软件的管理员权限 （即读写权限）。

用户帐户 `samuser` 仅具有 `Guest` 权限。该用户对 Sun StorEdge QFS 和 Sun StorEdge SAM-FS 操作仅具有只读访问权限。

如果删除 File System Manager 软件，系统将删除 `samadmin` 和 `samuser` Solaris 帐户以及 `SAMadmin` 角色。但是，删除脚本并不会删除您手工创建的任何附加帐户。您必须使用以下一个或两个过程来管理您手工添加的所有帐户。

▼ 创建附加的管理员帐户

管理员帐户的持有者具有管理 File System Manager、 Sun StorEdge QFS、 Sun StorEdge SAM-FS 软件的管理员权限 （即读写权限）。

1. **登录到管理站。**
2. **键入 `useradd` *username*。**

3. **键入** `passwd` *username*。
4. **按屏幕上的提示键入密码。**
5. **键入** `usermod -R SAMadmin` *username*。

注 – 不要将 `root` 用作 *username*。

▼ 创建附加的 Guest 帐户

guest 帐户持有者对 Sun StorEdge QFS 和 Sun StorEdge SAM-FS 操作仅具有只读访问权限。

1. **登录到管理站。**
2. **键入** `useradd` *account_name*。
3. **键入** `passwd` *account_name*。
4. **按屏幕上的提示键入密码。**

通过 File System Manager 管理其他服务器

默认情况下， File System Manager 只设置用于管理其所在的服务器。它还可以用于对其他运行 Sun StorEdge QFS 或 Sun StorEdge SAM-FS 软件的服务器进行管理，但是首先应设置这些服务器，使其允许 File System Manager 访问。添加其他的服务器后，用户可通过浏览器界面管理这些服务器上的文件系统、归档进程和介质。

添加服务器：

1. **除了使用浏览器界面外，用户还可以使用 telnet 连接到希望添加的服务器。请以超级用户身份登录。**
2. **使用** `fsmadm`**(1M)** `add` **命令将管理站 （File System Manager 软件所在的系统）添加到可远程管理此服务器的主机的列表。**

   使用此命令添加到列表中的所有主机均可以远程管理该服务器。其他的所有主机则无法远程管理该服务器。

   例如：
   # fsmadm add 管理站名称 . 域名

   要确保管理站添加成功，请运行 `fsmadm`(1M) `list` 命令，并检验输出中是否列出了该管理站。
3. **以管理员用户身份登录到 File System Manager 浏览器界面。**

4. **在 "Servers" 页面，单击 "Add"。**

   屏幕上将显示 "Add Server" 窗口。

5. **在 "Server Name" 或 "IP Address" 字段中，键入服务器的名称或 IP 地址。**

6. **单击 "OK"。**

第 2 章

使用自动化库和手动载入的驱动器

自动化库是一种自动控制的设备，它可在无操作人员参与的情况下，自动载入和卸载可移除卡盒。卡盒可以导入自动化库或从中导出。系统能够自动载入或卸载卡盒。归档和登台进程根据站点定义的方案来分配要使用的驱动器数量。自动化库也可称为介质更换器、自动光盘存储器、传输器、资料库或介质库。

以下几节从各个方面介绍了如何在 Sun StorEdge SAM-FS 环境中使用自动化库。《Sun StorEdge SAM-FS 安装和升级指南》中提供了初始配置指导，本章将介绍自动化库和手动载入的驱动器的操作指导。此外，本章还介绍在被请求的卷不在库中时，用于提示操作员载入该卷的通知工具。

注 – Sun StorEdge SAM-FS 软件可与许多厂商的自动化库进行交互。有关自动化库型号、固件级别以及其他兼容性方面的信息，请与 Sun 的客户支持部门联系。

某些自动化库的功能所产生的操作可能不同于本章介绍的操作。要确定对于 Sun StorEdge SAM-FS 环境中所使用的自动化库，其供应商是否提供有额外的操作指导，请查看附录 A：第 201 页 “具有供应商特定操作过程的自动化库的基本操作”。

本章包括以下主题：

约定

执行本章所述的基本操作过程时，通常需要使用 samcmd(1M) 命令、samu(1M) 操作员实用程序以及以下命令：

- tplabel(1M)
- odlabel(1M)
- auditslot(1M)
- cleandrive(1M)
- chmed(1M)
- import(1M)
- set_state(1M)
- samexport(1M)

但是，在许多情况下，您可以选用多种方法来执行所述的任务。您可以在 File System Manager 中执行其中许多任务，这个管理器是 Sun StorEdge QFS 和 Sun StorEdge SAM-FS 软件基于 Web 的图形用户界面 (Graphical User Interface, GUI)。可通过这个界面来配置、控制、监视和重新配置 Sun StorEdge QFS 和 Sun StorEdge SAM-FS 环境中的组件。有关如何安装 File System Manager 的信息，请参见 《Sun StorEdge SAM-FS 安装和升级指南》。有关如何使用 File System Manager 的信息，请参见其联机帮助。有关如何使用 samu(1M) 操作员实用程序的信息，请参见 《Sun StorEdge QFS 配置和管理指南》。

命令参数

许多命令接受一组通用的参数。表 2-1 列出了这些参数。

表 2-1 命令参数

参数	含义
eq	所访问的设备在 mcf 文件中定义的设备序号。可以识别的设备包括自动化库、驱动器或文件系统。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。
partition	磁光盘的一面。 partition 必须为 1 或 2。
media_type	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
vsn	分配给卷的卷序列名。

某些命令可以接受多种参数组合，这视您的环境而定。例如，在 samu(1M) 操作员实用程序中， load 命令具有以下两种格式：

```
:load eq:slot
:load media_type.vsn
```

请注意以下细节：

- 第一种格式使用冒号 (:) 分隔 *eq* 和 *slot*。
- 第二种格式使用句点 (.) 分隔 *media_type* 和 *vsn*。

术语

本章所使用的某些术语对您来说可能是新术语，因此，表 2-2 中列出了一些常用术语及其含义。

表 2-2 术语

术语	含义
自动化库	用于存储磁带或光盘卡盒的自动设备。
卡盒	磁带或磁光盘卡盒。磁光盘卡盒可以包含一个或多个卷或分区。
分区	整个磁带或磁光盘的一面。一个分区只能包含一个卷。
卷	卡盒中用于存储数据的命名区域。一个卡盒中可以有一个或多个卷。双面卡盒有两个卷，每一面为一个卷。系统采用卷序列名 (Volume Serial Name, VSN) 来标识卷。

自动化库操作

有几项基本操作在所有自动化库中基本上都是相同的。本节介绍了以下基本操作：

- 第 13 页 “启动可移除介质的操作”
- 第 12 页 “停止可移除介质的操作”
- 第 13 页 “打开自动化库”
- 第 13 页 “关闭自动化库”
- 第 14 页 “将卡盒载入自动化库”
- 第 15 页 “从驱动器中卸载卡盒”
- 第 15 页 “标记卡盒”

- 第 17 页 “审计卷”
- 第 18 页 “审计自动化库 （仅限于直接连接）”
- 第 18 页 “使用清洁卡盒”
- 第 20 页 “清洁磁带机”
- 第 22 页 “清除介质错误”
- 第 23 页 “从驱动器中取出卡住的卡盒”
- 第 24 页 “目录操作和卡盒的导入与导出”
- 第 28 页 “启用载入通知”

▼ 停止可移除介质的操作

可以停止可移除介质的操作，并保持 Sun StorEdge SAM-FS 系统的安装状态。例如，在您希望手动操作库中的卡盒时，可能需要执行此操作。恢复介质操作之后，未完成的登台操作将被重新执行，而归档操作也将随之恢复。

- **要停止可移除介质的操作，可使用 `samcmd`(1M) `idle` 和 `samd`(1M) `stop` 命令。**

  这些命令的使用格式如下：

```
samcmd idle eq
samd stop
```

  其中的 *eq*，用于输入所访问的设备在 `mcf` 文件中定义的设备序号。要将驱动器置于空闲状态，请对 `mcf` 文件中配置的每一个 *eq* 运行 `samcmd idle` *eq* 命令。

  也可以使用 `samu`(1M) 操作员实用程序或使用 File System Manager 将驱动器置于空闲状态。

注 – 在运行 `samd`(1M) `stop` 命令之前，Sun StorEdge SAM-FS 环境中的驱动器应该处于空闲状态。其目的是使归档程序、登台程序和其他进程结束当前的任务。如果未能成功运行 `samd`(1M) `stop` 命令，则在恢复归档、登台或其他活动时可能导致意外的结果。

▼ 启动可移除介质的操作

通常，当安装了 Sun StorEdge SAM-FS 文件系统后，可移除介质的操作便已启动。

- **要手动启动可移除介质操作，而不安装文件系统，请输入 `samd`(1M) `start` 命令。**

  此命令的使用格式如下：

```
# samd start
```

  如果在输入上述命令时可移除介质操作已在运行，则会生成以下消息：

```
SAM-FS sam-amld daemon already running
```

  有关 `samd`(1M) 命令的更多信息，请参见 `samd`(1M) 手册页。

▼ 打开自动化库

当某个库处于 `on` 状态时，则它已经处于 Sun StorEdge SAM-FS 系统的控制之下，且可以继续执行一般性的操作。打开库时，Sun StorEdge SAM-FS 软件会执行以下操作：

- 根据设备的内部状态来查询设备。它可以发现磁带的位置、是否使用了条码等。
- 更新目录和其他内部结构。

- **使用 `samcmd`(1M) `on` 命令打开自动化库。**

  此命令的使用格式如下：

```
samcmd on eq
```

  其中的 *eq*，用于指定所访问的自动化库在 `mcf` 文件中定义的设备序号。

  另外，还可使用 `samu`(1M) 或 File System Manager 执行此任务。

▼ 关闭自动化库

将自动化库置于 `off` 状态可以停止 I/O 操作，并使该库不再受 Sun StorEdge SAM-FS 控制。此时，卡盒将不能自动移动。请注意，自动化库中的驱动器仍处于 `on` 状态。若要执行以下任务，可能需要关闭自动化库：

- 仅停止 Sun StorEdge SAM-FS 自动化库的操作。
- 关闭自动化库的电源。

- **使用 samcmd(1M) off 命令关闭自动化库。**

  此命令的使用格式如下：

```
samcmd off eq
```

  其中的 *eq*，用于指定所访问的自动化库在 mcf 文件中定义的设备序号。

  另外，还可使用 samu(1M) 或 File System Manager 执行此任务。

▼ 将卡盒载入自动化库

当请求某个 VSN 以进行归档或登台操作时，系统会自动将卡盒载入驱动器。载入是指将卡盒从存储插槽移入驱动器，并使之处于就绪状态的过程。

- **使用 samcmd(1M) load 命令手动载入卡盒。**

  即使驱动器处于 unavail 状态，也可使用此命令。此命令具有以下两种格式：

```
samcmd load eq:slot[:partition]
samcmd load media_type.vsn
```

表 2-3 samcmd(1M) load 的参数

参数	含义
eq	所访问的驱动器在 mcf 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。
media_type	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
partition	磁光盘的一面。 partition 必须为 1 或 2。此参数不适用于磁带卡盒。
vsn	分配给卷的卷序列名。

另外，还可使用 samu(1M) 或 File System Manager 执行此任务。

当手动载入卡盒时，它通常会载入库中的下一个可用驱动器。如果不希望某个驱动器这样自动载入，请使用 samu(1M) 实用程序的 :unavail 命令，或使用 File System Manager 更改该设备的状态。例如，在故障恢复操作或对磁带进行分析期间，就可能会执行此操作。

▼ 从驱动器中卸载卡盒

当不再需要使用卷时，系统会自动卸载卡盒。也可手动卸载驱动器。卸载是指从驱动器中取出卡盒。

- **使用 samcmd(1M) unload 命令手动卸载卡盒。**

  即使驱动器处于 unavail 状态，也可以使用此命令。此命令的使用格式如下：

```
samcmd unload eq
```

其中的 *eq*，用于指定所访问的自动化库在 mcf 文件中定义的设备序号。

另外，还可使用 samu(1M) 或 File System Manager 执行此任务。

标记卡盒

标记卡盒的过程取决于您是标记磁带还是标记光盘卡盒。以下两小节介绍了这些过程。

注意 – 标记和重新标记卡盒可使任何软件都无法再访问卡盒中当前存储的数据。应该仅在确定卡盒中存储的数据不再有用时，才重新标记卡盒。

▼ 标记或重新标记磁带

以下 tplabel(1M) 命令行格式显示了在标记或重新标记磁带时最常用的选项：

```
tplabel [ -new | -old vsn ] -vsn vsn eq:slot
```

表 2-4 tplabel(1M) 的参数

参数	含义
vsn	卷序列名。如果是进行重新标记，新 VSN 名可以与旧 VSN 名相同。
eq	所访问的自动化库或手动载入的驱动器在 mcf 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。此参数不适用于手动载入的驱动器。

- **要标记新磁带，请使用 `tplabel`(1M) 命令。**

  此命令的使用格式如下：

```
tplabel -new -vsn vsn eq:slot
```

- **要重新标记现有磁带，请使用 `tplabel`(1M) 命令。**

  此命令的使用格式如下：

```
tplabel -old vsn -vsn vsn eq:slot
```

发出上述用以标记或重新标记磁带的命令后，系统会载入磁带并确定其位置，然后写入磁带标签。有关 `tplabel`(1M) 命令的更多信息，请参见 `tplabel`(1M) 手册页。

另外，还可使用 File System Manager 执行此任务。

▼ 标记或重新标记光盘

以下 `odlabel`(1M) 命令行格式显示了在标记或重新标记光盘时最常用的选项：

```
odlabel [ -new | -old vsn ] -vsn vsn eq:slot:partition
```

表 2-5 `odlabel`(1M) 的参数

参数	含义
vsn	卷序列名。如果是进行重新标记，新 VSN 名可以与旧 VSN 名相同。
eq	所访问的自动化库或手动载入的驱动器在 `mcf` 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。此参数不适用于手动载入的驱动器。
partition	磁光盘的一面。 partition 必须为 `1` 或 `2`。此参数不适用于磁带卡盒。

- **要标记新光盘，请使用 `odlabel`(1M) 命令。**

  此命令的使用格式如下：

```
odlabel -new -vsn vsn eq:slot:partition
```

- **要重新标记现有光盘，请使用 `odlabel`(1M) 命令。**

  此命令的使用格式如下：

```
odlabel -old vsn -vsn vsn eq:slot:partition
```

  发出上述用以标记或重新标记光盘的命令后，系统会载入光盘并确定其位置，然后写入光盘标签。有关 `odlabel`(1M) 命令的更多信息，请参见 `odlabel`(1M) 手册页。

  另外，还可使用 File System Manager 执行此任务。

▼ 审计卷

有时，需要在库目录中更新以前所报告的磁带或光盘卡盒上的剩余空间。`auditslot`(1M) 命令用于载入包含卷的卡盒、读取标签并更新存储插槽的库目录条目。

- **使用 `auditslot`(1M) 命令审计卷。**

  此命令的使用格式如下：

```
auditslot [-e] eq:slot[:partition]
```

表 2-6 `auditslot`(1M) 的参数

参数	含义
`-e`	如果指定了 `-e` 选项，且介质为磁带，则将会更新剩余的空间。否则，将不会进行更新。
eq	所访问的自动化库或手动载入的驱动器在 `mcf` 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。此参数不适用于手动载入的驱动器。
partition	磁光盘的一面。 partition 必须为 `1` 或 `2`。此参数不适用于磁带卡盒。

  有关 `auditslot`(1M) 命令的更多信息，请参见 `auditslot`(1M) 手册页。

  也可以使用 `samu`(1M) 实用程序的 `:audit` 命令或使用 File System Manager 执行此任务。

▼ 审计自动化库 （仅限于直接连接）

注 – 此任务不能在通过网络连接的自动化库上执行。

执行全面审计时，会将每一个卡盒载入驱动器、读取标签并更新库目录。在以下情况下应审计库：

- 移动了自动化库中的卡盒，但没有使用 Sun StorEdge SAM-FS 的命令。
- 如果怀疑库目录的状态有问题，且希望更新库目录 （例如在意外断电后）。
- 如果在没有配备邮箱的自动化库中添加、取出或移动了卡盒。

- **可使用 `samcmd`(1M) `audit` 命令执行自动化库的全面审计。**

  此命令的使用格式如下：

```
samcmd audit eq
```

其中的 *eq*，用于指定所访问的自动化库在 `mcf` 文件中定义的设备序号。

也可以使用 `samu`(1M) 实用程序的 `:audit` 命令或使用 File System Manager 执行此任务。

使用清洁卡盒

Sun StorEdge SAM-FS 系统允许您导入清洁卡盒来清洁磁带机。此过程的具体步骤取决于清洁卡盒是否编有条码。以下几节介绍了使用清洁卡盒的不同情况。

清洁方法因生产商不同而不同。如果遇到这方面的问题，请参见第 201 页 “具有供应商特定操作过程的自动化库的基本操作”，以确定是否需要为您的设备执行特殊步骤。

注 – 此任务不能在通过网络连接的自动化库上执行。

▼ 重置清洁循环次数

清洁磁带仅在一定的清洁循环次数内有效。可以使用 `samu`(1M) 实用程序的 `:v` 命令或使用 File System Manager 来显示剩余的清洁次数。

Sun StorEdge SAM-FS 系统可以跟踪每个清洁磁带的清洁循环次数，并在剩余循环次数等于零时弹出磁带。例如， DLT 清洁磁带的循环次数为 20，而 Exabyte 清洁磁带的循环次数则为 10。每次导入清洁磁带时，系统均会自动将清洁循环次数重置为该类型磁带的最高循环次数。

如果系统支持自动清洁功能，但自动化库中所有清洁磁带的剩余循环次数均为零，则系统会将驱动器设置为关闭状态，并在 Sun StorEdge SAM-FS 日志中记录一条消息。

- **可使用 chmed(1M) 命令将清洁磁带的循环次数重置为零。**

  此命令的使用格式如下：

```
chmed -count count media_type.vsn
```

表 2-7 chmed(1M) 的参数

参数	含义
count	您为清洁磁带重置的清洁循环次数。
media_type	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
vsn	分配给卷的卷序列名。

▼ 使用具有条码的清洁卡盒

如果清洁卡盒编有条码，则可以使用 import(1M) 命令导入清洁卡盒。

1. **确保清洁卡盒的条码为 CLEAN，或以字母 CLN 开头。**
2. **使用 import(1M) 命令导入清洁卡盒。**

   此命令的使用格式如下：

```
import eq
```

其中的 *eq*，用于指定所访问的自动化库在 mcf 文件中定义的设备序号。

Sun StorEdge SAM-FS 系统会将卡盒从邮箱移至存储插槽，并更新每个卡盒的库目录。另外，发出此命令后，系统将设置清洁介质标志，并根据介质的类型将访问计数设置为适当的清洁循环次数。每使用介质清洁一次驱动器，访问计数便减少一次。

例如，以下命令是将一个在 mcf 文件中编号为 50 的清洁磁带导入自动化库：

```
# import 50
```

另外，还可使用 samu(1M) 或 File System Manager 执行此任务。

▼ 使用没有条码的清洁卡盒

如果清洁卡盒没有条码，则必须首先将其导入。此时，它未被标记为清洁卡盒。请执行以下步骤：

1. **使用 `import`(1M) 命令导入卡盒。**

   此命令的使用格式如下：

   ```
   import eq
   ```

   其中的 *eq*，用于指定所访问的自动化库在 `mcf` 文件中定义的设备序号。

2. **使用 `chmed`(1M) 命令将设备类型更改为清洁卡盒。**

   您必须知道自动化库的设备序号以及清洁卡盒被载入的存储插槽。

   在以下命令行示例中，自动化库的设备序号为 50，且清洁卡盒位于存储插槽 77 中：

   ```
   # chmed +C 50:77
   ```

   上述命令将卡盒类型更改为清洁卡盒类型。

3. **再次使用 `chmed`(1M) 命令，设置清洁循环次数。**

   以下命令示例为上一步骤中的清洁卡盒设置了清洁循环次数：

   ```
   # chmed -count 20 50:77
   ```

   有关 `chmed`(1M) 命令的更多信息，请参见 `chmed`(1M) 手册页。

▼ 清洁磁带机

注 – Sun StorEdge SAM-FS 系统不支持自动地清洁通过网络连接的库。您应使用供应商提供的库管理软件来自动清洁此类库。

如果硬件支持清洁磁带的使用，则 Sun StorEdge SAM-FS 环境也支持使用清洁磁带。如果某个磁带机要求清洁，系统会自动载入清洁磁带。

如果您的系统使用编有条码的标签，则在条码标签中，清洁磁带的 VSN 必须为 `CLEAN` 或以字母 `CLN` 开头。另外，可以使用 `chmed`(1M) 命令将 VSN 标记为清洁磁带并设置清洁循环计数。系统允许安装多个清洁磁带。

注 – 某些驱动器错误可能会导致重复载入清洁卡盒，直到执行完所有的清洁循环。为防止发生此类情况，您可以使用 chmed(1M) 命令限制清洁卡盒的清洁循环次数。例如：

```
# chmed -count 20 50:77
```

如果不能进行自动清洁，而且系统使用条码，则可手动执行以下步骤来请求清洁驱动器：

- **使用 cleandrive(1M) 命令。**

  此命令的使用格式如下：

```
cleandrive eq
```

其中的 *eq*，用于指定所访问的自动化库在 mcf 文件中定义的设备序号。这是要载入清洁卡盒的驱动器。

磁带机自动清洁

从 Sun StorEdge SAM-FS 4U4 开始，通过软件启动的磁带机清洁功能在默认情况下被设置为 off，因此用户必须选择是通过硬件启动还是通过软件启动磁带机清洁。

通过硬件启动清洁的方式使用了介质更换器内置的自动清洁特性。要使用此特性，清洁卡盒可能需要插入特殊插槽。有关指导，请参见生产商提供的文档。

通过软件启动清洁的方式使用了 Sun StorEdge SAM-FS 的自动清洁特性。 此现有的特性新添加了 logsense 增强选项，用于防止驱动器使用无效的清洁介质。要启用 Sun StorEdge SAM-FS 的自动清洁特性，应首先禁用通过硬件启动清洁，并在 defaults.conf 中输入以下行：

```
tapeclean = all autoclean on logsense on
```

要调用传统的 Sun StorEdge SAM-FS 自动清洁特性 （基于检测驱动器清洁状态的数据），请在 defaults.conf 文件中输入以下行：

```
tapeclean = all autoclean on logsense off
```

注 – 在具有两个以上驱动器的库上使用自动清洁特性时，建议每个 Sun StorEdge SAM-FS 目录至少具有两个清洁卡盒。如果需要清洁多个驱动器，但没有足够的清洁卡盒，则系统会将这些驱动器置于 DOWN 状态。

▼ 清除介质错误

当卡盒发生硬件或软件错误时，Sun StorEdge SAM-FS 系统会在 VSN 目录中设置 `media error` 标志。对于任何生成 `media error` 信号的给定卡盒，均可使用 `chmed`(1M) 命令清除其错误，然后尝试使用该卡盒。可使用 `samu`(1M) 实用程序的 `v` 命令或 File System Manager 显示 `media error` 标志。

1. **使用 `chmed`(1M) 命令清除 `media error` 标志。**

   按以下格式使用此命令清除 `media error` 标志：

```
chmed -E media_type.vsn
```

表 2-8 `chmed`(1M) 的参数

参数	含义
media_type	介质类型。有关有效介质类型的列表，请参见 `mcf`(4) 手册页。
vsn	分配给卷的卷序列名。

2. **运行 `auditslot`(1M) 命令更新剩余空间信息。**

   此命令的使用格式如下：

```
auditslot -e eq:slot[:partition]
```

表 2-9 `auditslot`(1M) 的参数

参数	含义
`-e`	如果指定了 `-e` 选项，且介质为磁带，则将会更新剩余的空间。否则，将不会进行更新。
eq	所访问的自动化库或手动载入的驱动器在 `mcf` 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。此参数不适用于手动载入的驱动器。
partition	磁光盘的一面。 partition 必须为 `1` 或 `2`。此参数不适用于磁带卡盒。

有关 `auditslot`(1M) 命令的更多信息，请参见 `auditslot`(1M) 手册页。

也可以使用 `samu`(1M) 实用程序的 `:audit` 命令或使用 File System Manager 执行此任务。

▼ 从驱动器中取出卡住的卡盒

如果卡盒卡在驱动器中，请执行以下步骤。

1. **使用 samcmd(1M) off 命令关闭自动化库中的驱动器。**
   此命令的使用格式如下：

```
samcmd off eq
```

   其中的 *eq*，用于指定所访问的驱动器在 mcf 文件中定义的设备序号。
   另外，还可使用 samu(1M) 或 File System Manager 执行此任务。

2. **使用 samcmd(1M) off 命令关闭自动化库。**
   此命令的使用格式如下：

```
samcmd off eq
```

   其中的 *eq*，用于指定所访问的库在 mcf 文件中定义的设备序号。
   另外，还可使用 samu(1M) 或 File System Manager 执行此任务。

3. **从驱动器中物理取出卡盒。**
   确保不要损坏卡盒或驱动器。

4. **使用 samcmd(1M) on 命令打开自动化库和驱动器。**
   为驱动器和库分别运行一条此命令。此命令的使用格式如下：

```
samcmd on eq
```

   其中的 *eq*，用于指定所访问的自动化库或驱动器在 mcf 文件中定义的设备序号。
   如果自动化库在打开后执行了审计，则本过程结束。如果未执行审计，请执行下一步骤。

5. **如果将卡盒放回其存储插槽，请使用 chmed(1M) 命令调整库目录，以便为损坏的磁带设置占用标志。**
   此命令的使用格式如下：

```
chmed +o eq:slot
```

表 **2-10** chmed(1M) 的参数

参数	含义
eq	所访问的自动化库或驱动器在 mcf 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。此参数不适用于手动载入的驱动器。

有关 chmed(1M) 命令的更多信息，请参见 chmed(1M) 手册页。

如果您取出卡盒，并希望在以后将其放回，则必须将卡盒导入自动化库。

目录操作和卡盒的导入与导出

在自动化库中物理地添加 （导入）和从中取出 （导出）卡盒使您可以执行多项功能，这其中包括：

- 替换卡盒。
- 将卡盒重新定位为离站存储，以备今后进行故障恢复时使用。如果要执行此任务，则您可以使用 chmed(1M) 命令的 -I 选项，来指定诸如卡盒存储位置等其他信息。

导入和导出卡盒时，还可更新库目录。在 Sun StorEdge SAM-FS 系统中，可使用 import(1M) 与 samexport(1M) 命令完成此任务。另外，还可使用 File System Manager 执行这些任务。

库**目录**是一个中央仓库，其中包含了 Sun StorEdge SAM-FS 环境在查找自动化库中的卡盒时所需的所有信息。库目录文件是一个二进制 UFS 驻留文件，其中包括自动化库中每一个插槽的有关信息。该文件中的信息包括：与插槽中的卡盒相关联的一个或多个卷序列名 (Volume Serial Names, VSN)；该卡盒的容量和剩余空间；以及指示卡盒只读、写保护、回收和其他状态信息的标志。

Sun StorEdge SAM-FS 环境处理目录的方式不尽相同，这取决于自动化库连接到服务器的方式，具体如下：

- 如果自动化库采用直接连接方式，则库目录的条目与自动化库中的物理插槽之间是一一对应关系。库目录中的第一个条目是自动化库中的第一个插槽。需要某个卡盒时，系统将首先查询库目录，以确定哪一个插槽包含该 VSN，然后发出命令以将该插槽中的卡盒载入驱动器。
- 如果自动化库采用网络连接方式，则库目录中的条目与自动化库中的插槽不是直接的对应关系。它是自动化库中已知 VSN 的列表。请求某个卡盒时，系统将向供应商的软件发送请求以将该 VSN 载入驱动器。供应商的软件可以确定包含该 VSN 的存储插槽。

每个自动化库处理卡盒导入和导出操作的方式，都因系统特性和供应商提供的软件不同而存在差异。例如，对于 ACL 4/52 库，您需要先运行 move 命令将卡盒移至导入 / 导出设备中，然后才能从自动化库中导出卡盒。

注 – 通过网络连接的自动化库则使用它们自己的实用程序来导入和导出卡盒，因此 `import`(1M) 和 `samexport`(1M) 命令只更新 Sun StorEdge SAM-FS 系统使用的库目录条目。如果您使用的是通过网络连接的自动化库，请参见第 201 页 “具有供应商特定操作过程的自动化库的基本操作”以了解有关导入和导出卡盒的信息。

跟踪导出的介质 — 历史记录

Sun StorEdge SAM-FS 历史记录可以跟踪记录从自动化库或手动安装的设备中导出的卡盒。历史记录类似于一个虚拟库，但它没有已定义的硬件设备。与自动化库一样，它也在 `mcf` 文件中进行配置；具有用于记录与其关联的所有卡盒的目录；可以导入和导出卡盒；而且在 File System Manager 中显示为另一自动化库。

可以在 `mcf` 文件中，使用 `hy` 设备类型来配置历史记录。如果没有在 `mcf` 文件中配置历史记录，它将按以下方式创建：

```
historian  n+1  hy  -  on  /var/opt/SUNWsamfs/catalog/historian
```

在以上条目中，*n+1* 是 `mcf` 文件中的最后一个设备序号加 1。如果希望目录使用不同的设备序号或路径名，则只需在 `mcf` 中对历史记录进行定义。

历史记录第一次启动时，历史记录库目录将被初始化为具有 32 个条目。请确保文件系统中的目录足以容纳整个目录。您可能希望跟踪已从库中导出的现有 Sun StorEdge SAM-FS 卡盒。在这种情况下，您需要按 `build_cat`(1M) 手册页中的说明，根据现有的卡盒建立历史记录目录。

`defaults.conf` 文件中的以下两个配置指令将影响历史记录的操作：

- 如果使用了 `exported_media = unavailable` 指令，则从自动化库中导出的任何卡盒均会被标记为不能用于历史记录。请求已标记为不可用的卡盒将产生 EIO 错误。
- 如果使用了 `attended = no` 指令，则它将通知历史记录没有操作员处理载入请求。如果已通知历史记录载入卡盒，但实际并未载入，则会产生 EIO 错误。

有关配置的更多信息，请参见 `historian`(7) 和 `defaults.conf`(4) 手册页。

对自动化库执行导入和导出操作

邮箱是自动化库中的一个区域，它用于在自动化库中添加和取出卡盒。 `import`(1M) 命令用于将卡盒从邮箱移至存储插槽。 `samexport`(1M) 命令用于将卡盒从存储插槽移至邮箱。对于大多数库，如果在 Sun StorEdge SAM-FS 软件启动时，邮箱中含有卡盒，则软件会在启动期间自动导入此卡盒。

导入和导出的具体操作过程因生产商而异。如果遇到这方面的问题，请参见第 201 页“具有供应商特定操作过程的自动化库的基本操作”以确定是否需要为您的设备执行特殊过程。

以下几节介绍如何导入和导出卡盒：

- 第 26 页 “向使用邮箱的库中导入卡盒”
- 第 26 页 “从使用邮箱的库中导出卡盒”
- 第 27 页 “向不使用邮箱的库中导入卡盒”
- 第 27 页 “从不使用邮箱的库中导出卡盒”

▼ 向使用邮箱的库中导入卡盒

要将卡盒导入使用邮箱的自动化库中，请执行以下步骤。

1. **使用生产商建议的操作打开邮箱。**

   邮箱的旁边通常配有一个按钮。有时，邮箱可能只配有一个插槽 （在供应商的文档中称为**邮箱插槽**）。

2. **将卡盒手动放入邮箱。**

3. **关闭邮箱。**

4. **使用 `import`(1M) 命令导入卡盒。**

   此命令的使用格式如下：

```
import eq
```

   其中的 *eq*，用于指定所访问的库在 `mcf` 文件中定义的设备序号。

   系统将卡盒从邮箱移至存储插槽，并更新每个卡盒的库目录。

   另外，还可使用 `samu`(1M) 或 File System Manager 执行此任务。

▼ 从使用邮箱的库中导出卡盒

本过程用于将卡盒从存储插槽移至邮箱或邮箱插槽。要从使用邮箱的自动化库中导出（弹出）卡盒，请执行以下步骤。

1. **使用 `samexport`(1M) 命令将卡盒从存储插槽移至邮箱。**

   按以下某一种格式使用此命令：

```
samexport eq:slot
samexport media_type.vsn
```

表 2-11 samexport(1M) 的参数

参数	含义
eq	所访问的自动化库在 mcf 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。
media_type	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
vsn	分配给卷的卷序列名。

另外，还可使用 samu(1M) 或 File System Manager 执行此步骤。

2. **使用生产商建议的操作打开邮箱或邮槽。**

   邮箱的旁边通常配有一个按钮。

▼ 向不使用邮箱的库中导入卡盒

1. **使用 samcmd(1M) unload 命令。**

   此命令的使用格式如下：

```
samcmd unload eq
```

   其中的 *eq*，用于指定所访问的库在 mcf 文件中定义的设备序号。

   等待系统完成其当前任务，将自动化库的状态设置为关闭，然后将当前活动的目录转送至历史记录。

2. **解除自动化库挡门的锁定，打开挡门。**

3. **将卡盒载入可用的插槽。**

4. **关闭自动化库的挡门，锁定挡门。**

   自动化库将重新初始化并扫描库中的卡盒。 Sun StorEdge SAM-FS 软件将所导入的卡盒的 VSN 添加到库目录中，从而更新该目录。此时，自动化库的状态会设置为 on。

▼ 从不使用邮箱的库中导出卡盒

1. **使用 samcmd(1M) unload 命令。**

   此命令的使用格式如下：

```
samcmd unload eq
```

   其中的 *eq*，用于指定所访问的库在 mcf 文件中定义的设备序号。

   等待系统完成其当前任务，将自动化库的状态设置为关闭，然后将当前活动的目录转送至历史记录。

2. **解除自动化库挡门的锁定，打开挡门。**

3. **从各个插槽中取出所需的卡盒。**

4. **关闭自动化库的挡门，锁定挡门。**

   自动化库将重新初始化并扫描自动化库中的卡盒。系统使用库插槽中当前卡盒的 VSN 来更新库目录。已取出的卡盒的 VSN 会从库目录中删除，并且只保留在历史记录文件中。此时，自动化库的状态会设置为 `on`。

▼ 启用载入通知

Sun StorEdge SAM-FS 软件会定期请求载入卡盒，以满足归档和登台需求。如果被请求的卡盒位于库中，则系统会自动处理该请求。如果被请求的卡盒不在库中，则需要操作员载入卡盒。如果已启用载入通知，则当需要获取不在库中的卡盒时，`load_notify.sh`(1M) 脚本会向有关人员发送电子邮件。

1. **成为超级用户。**

2. **使用 `cp`(1) 命令，将载入通知脚本从其安装位置复制到可操作的位置。**

   例如：

```
# cp /opt/SUNWsamfs/examples/load_notify.sh
/etc/opt/SUNWsamfs/scripts/load_notify.sh
```

3. **使用 `more`(1) 或其他命令检查 `defaults.conf` 文件。**

   确保此文件中包含以下指令：

   - `exported_media=available`
   - `attended=yes`

   默认情况下，系统已设置了这些指令。如果要启用载入通知功能，请确保未更改这些指令。

4. **修改 `load_notify.sh` 脚本以便将通知发送给操作员。**

   默认情况下，此脚本将向 `root` 用户发送电子邮件。不过，您可以编辑此脚本以将电子邮件发送给其他人员、拨打寻呼机，或提供其他通知方式。

手动载入驱动器操作

本节介绍在您配有手动载入的独立驱动器 （而不是自动化库）时所执行的操作。每一个手动载入的驱动器都有自己的单插槽库目录。

▼ 载入卡盒

- **要将卡盒载入手动载入设备，请根据生产商的说明将卡盒放入驱动器。**

  Sun StorEdge SAM-FS 系统会识别已载入的卡盒，读取标签并更新手册 （即单插槽目录）。无需进行其他操作。

▼ 卸载卡盒

- **使用 `samcmd`(1M) `idle` 命令将驱动器置于空闲状态。**

  此命令可以确保无任何归档或登台进程处于活动状态。此命令的使用格式如下：

```
samcmd idle eq
```

  其中的 *eq*，用于指定所访问的驱动器在 `mcf` 文件中定义的设备序号。

  完成所有 I/O 活动后，驱动器的状态将从 `idle` 切换为 `off`，然后弹出磁带。

  如果是磁带，则先进行倒带，然后才能取出卡盒。如果是光盘卡盒，则会自动弹出。有关取出特定卡盒的说明，请参见生产商提供的指导。

  另外，还可使用 `samu`(1M) 或 File System Manager 执行此任务。

▼ 查看库目录

- **使用 `samu`(1M) 实用程序的 `:v` 命令。**

  此命令的使用格式如下：

```
:v eq
```

  其中的 *eq*，用于指定所访问的库在 `mcf` 文件中定义的设备序号。

第 3 章

归档

归档是指将文件从 Sun StorEdge SAM-FS 文件系统复制到可移除介质卡盒上的卷或另一个文件系统中磁盘分区的过程。本章中，术语**归档介质**是指向其中写入归档卷的各种卡盒或磁盘分片。Sun StorEdge SAM-FS 的归档功能可完成许多操作，例如可指定哪些文件需要立即归档，哪些文件从不需要归档，还可执行其他任务。

本章介绍归档程序的操作原理、制订站点归档策略的通用原则，以及如何通过创建 `archiver.cmd` 文件来实现这些归档策略。

它包括以下主题：

归档过程概述

归档程序自动将 Sun StorEdge SAM-FS 文件写入归档介质中。归档和登台文件并不需要操作人员的参与。文件归档至归档介质上的卷，每个卷由称为**卷序列名** (Volume Serial Name, VSN) 的唯一标识来识别。归档介质中可以包含一个或多个卷。识别单个卷时，必须指定介质类型和 VSN。

归档程序在安装 Sun StorEdge SAM-FS 文件系统时自动启动。通过在以下文件中插入归档指令，您可以为您的站点自定义归档程序的操作：

```
/etc/opt/SUNWsamfs/archiver.cmd
```

执行归档时，并不一定需要有 `archiver.cmd` 文件。当没有这个文件时，归档程序使用以下默认设置：

- 将所有文件归档至可用卷。
- 所有文件的**归档时限**为 4 分钟。归档时限是指从最后一次修改文件到开始归档文件所经历的时间。
- **归档时间间隔**为 10 分钟。归档时间间隔是指两次完整的归档过程之间相隔的时间。

以下几节介绍归档集的概念，并说明在归档过程中所执行的操作。

归档集

归档集表示一组要归档的文件。用户可以在任意多个文件系统上定义归档集。同一个归档集中的文件在大小、所有权、组和目录位置等方面遵守相同的标准。归档集控制归档副本的目的地、归档副本的保留时间以及数据归档之前的等待时间。归档集中的所有文件均被复制到与该归档集关联的卷。文件系统中的一个文件能且只能属于一个归档集。

文件在创建或修改之后，归档程序会将其复制到归档介质中。归档文件与标准 UNIX `tar`(1) 格式兼容。这确保了 Sun Solaris 操作系统 (Operating System, OS) 与其他 UNIX 系统之间的数据兼容性。此格式包括文件访问数据 （即 inode 信息）和文件的路径。由于采用了 `tar`(1) 格式，这样在 Sun StorEdge SAM-FS 环境中的所有数据都丢失时，就可使用标准的 UNIX 工具和命令来恢复文件。归档过程还会复制执行 Sun StorEdge SAM-FS 文件系统操作所必需的数据。此类数据包括目录、符号链接、分段文件的索引和归档介质信息。

在本节的后半部分，术语**文件**既指文件数据，也指元数据。仅在需要加以区分时，才会使用术语**文件数据**和**元数据**。而术语**文件系统**是指已安装的 Sun StorEdge SAM-FS 文件系统。

通常，管理员可为归档集取任意名称，但以下情况除外：

- 有两个保留的归档集名称：`no_archive` 和 `allsets`。

  `no_archive` 归档集是系统默认定义的归档集。选入此归档集的文件永远不会被归档。临时目录 （例如 `/sam1/tmp`）中的文件可能会包括在 `no_archive` 归档集中。

  `allsets` 归档集用于定义适用于所有归档集的参数。
- 每个 Sun StorEdge SAM-FS 文件系统的归档集名称都会被保留，以便控制结构信息。 Sun StorEdge SAM-FS 文件系统为每个文件系统都提供了一个默认归档集。对于每个文件系统，归档程序不仅为其归档元数据，而且还归档文件数据。文件系统归档集包括目录和链接信息以及不属于其他归档集的任何文件。默认归档集的名称与其关联的文件系统的名称相同，并且不能更改。例如，如果一个文件系统的名称为 `samfs1`，则其归档集的名称应为 `samfs1`。
- 归档集名最长不得超过 29 个字符。所用的字符仅限于 26 个大小写字母、数字 0 至 9 以及下划线字符 (_)。

归档操作

默认情况下，归档程序将为每一个归档集创建一份副本，但您可以要求为每一个归档集创建多达四份副本。归档集和副本份数意味着占用一系列卷。归档副本在其他卷中提供文件的复件。

为确保完整无缺地归档文件，归档程序会在文件修改之后等待一段指定的时间，然后再对其进行归档。正如前文所述，这段时间称为**归档时限**。

文件中的数据须经更改后，该文件才被视为归档或重新归档的对象。如果只是访问文件，则不会对其归档。例如，对某个文件运行 touch(1) 或 mv(1) 命令，并不会导致系统对该文件进行归档或重新归档。运行 mv(1) 命令只是改变文件的名称，而并没有更改文件数据。在故障恢复时，如果通过 tar(1) 文件进行恢复，则运行此命令可能会产生其他后果。有关故障恢复的更多信息，请参见 《Sun StorEdge SAM-FS 故障排除指南》。

归档程序依据文件的归档时限来选择要归档的文件。用户可以为每一个归档副本定义归档时限。

用户可以使用 touch(1) 命令将其文件的默认时间参考，更改成过去较早的时间或将来较远的时间。但是，这可能会导致意外的归档结果。为了避免此类问题，归档程序会调节时间参考以使它们始终处于以下所示的范围：

creation_time < *time_ref* < *time_now*

以下几节介绍归档程序执行的步骤，包括从最初的文件扫描进程到文件复制进程。

步骤 1：识别要归档的文件

每一个已安装的文件系统均有单独的 sam-arfind 进程。 sam-arfind 进程将监视每一个文件系统，以确定哪些文件需要归档。当某文件的更改会影响它自身的归档状态时，文件系统都会向其 sam-arfind 进程发出通知。这类更改的示例有：对文件进行修改、重新归档、取消归档以及重命名。 sam-arfind 进程在接到通知后会检查该文件，以确定是否需要对它执行归档操作。

sam-arfind 进程根据文件的属性说明，来确定文件所属的归档集。用于确定文件所属归档集的特征如下：

- 文件名中的目录路径部分，以及采用了正则表达式的完整文件名 （可选）
- 文件所有者的用户名
- 文件所有者的组名
- 最小文件大小
- 最大文件大小

如果文件的一份或多份副本到达或超出归档时限， sam-arfind 会将文件添加到归档集的一个或多个归档请求中。**归档请求**是属于同一个归档集的所有文件的集合。对于要重新归档的文件，则采用单独的归档请求。这样可以分别控制尚未归档文件以及重新归档文件的时间调度。归档请求是一个文件，它位于以下目录中：

/var/opt/SUNWsamfs/archiver/file_sys/ArchReq

这个目录中的文件是二进制文件，可以使用 showqueue(1M) 命令显示。

归档请求有时也称为 *ArchReq*。

如果文件的一份或多份副本未达到归档时限，该文件所在的目录以及将到达归档时限的时间会被添加到扫描列表中。一旦到达扫描列表中列出的时间，将对目录进行扫描。并将到达归档时限的文件添加到归档请求中。

如果文件处于脱机状态，则 sam-arfind 进程会选择将用作归档副本来源的卷。如果文件副本需要重新归档，则 sam-arfind 进程将选择包含这个需要重新归档的归档副本的卷。

如果文件已被分成数段，此进程将仅选择已发生更改的文件段进行归档。分段文件的索引不含用户数据，因此这些分段文件将被视为文件系统归档集的成员而单独进行归档。

归档优先级是根据文件属性特征以及与归档集关联的文件属性乘数计算出来的。事实上，其计算公式如下所示：

archive_priority = (*file_property_value* * *property_multiplier*) 之和

大多数 *file_property_value* 数值为 1 或 0，相当于属性是 TRUE 还是 FALSE。例如，如果正在创建第 1 个归档副本，则第 1 个属性副本的值为 1。而第 2 个副本、第 3 个副本和第 4 个副本的属性值为 0。

归档时限和文件大小等其他文件属性的值可以是 0 或 1 之外的其他数值。

property_multiplier 的值由归档集的 -priority 参数决定。由于可为文件的各个方面（例如归档时间或大小）设定值，因此您的站点可以改变归档请求的优先级。有关 -priority 参数的更多信息，请参见 archiver.cmd(4) 手册页。

archive_priority 和属性乘数是浮点数值。所有属性乘数的默认值均为 0.0。此时，此归档请求被设置为归档请求中的最高文件优先级。

确定文件是否需要归档的方法有两种：连续归档和扫描。连续归档即为，归档程序会对文件系统进行处理，以确定哪些文件需要归档。而扫描方法则是，归档程序定期细查文件系统，并选择需要归档的文件。以下几节将介绍这两种方法。

连续归档

连续归档是默认的归档方法 (examine=noscan)。在连续归档方法中，可以通过 -startage、-startcount 以及 -startsize 参数来指定归档集开始归档的条件。这些条件允许您根据已完成的归档工作优化归档时效。

- 示例 1。如果需要花费一个小时来创建应该同时归档的文件，则可以将 -startage 参数设置为 1 小时 (-startage 1h)，以确保所有文件在调度归档请求前都已创建完毕。
- 示例 2。可以将 -startsize 指定为 150 GB (-startsize 150g)，来命令归档程序在需要归档的数据达到 150 GB 时，才进行归档。

- 示例 3。如果知道要生成 3000 个需要归档的文件，则可以通过指定 -startcount 3000，来确保同时对这些文件进行归档。

在满足任何一个启动条件后， sam-arfind 进程就会发送归档请求至归档守护进程 sam-archiverd，以调度将文件复制到归档介质。

扫描归档

如果不使用连续归档方法，则可以指定 examine=scan 来指示 sam-arfind 进行扫描，以检查需要归档的文件。需要归档的文件会被放入归档请求中。 sam-arfind 进程将定期扫描每一个文件系统，以确定哪些文件需要归档。 sam-arfind 进程执行的第一个扫描过程是目录扫描。在此扫描期间， sam-arfind 按从上到下的顺序逐层遍历目录树。它将检查每一个文件，如果文件不需要归档，则为其设置 archdone 文件状态标志。接下来，对 .inodes 文件进行扫描。此时，只检查那些没有 archdone 标志的 inode。

完成文件系统扫描之后， sam-arfind 进程会将每一个归档请求发送至归档程序的守护进程 sam-archiverd，以调度将文件复制到归档介质。 sam-arfind 进程随后进入休眠状态，休眠的时间为 interval=*time* 指令所指定的时间。在时间间隔结束时，sam-arfind 进程将重新开始扫描。

步骤 2：编辑归档请求

sam-archiverd 守护进程收到归档请求后，将对其进行**编辑**。此步骤介绍了这个编辑过程。

可能无法一次对归档请求中的所有文件进行归档。这取决于归档介质的容量或归档程序命令文件中指定的控制条件。**编辑**是指从归档请求中选择一次要对哪些文件进行归档的过程。完成归档请求的归档复制操作后，如果仍有需要归档的文件，则会重新编辑归档请求。

sam-archiverd 守护进程会根据某些默认标准和站点特定的标准，来排列归档请求中的文件。默认操作是根据文件系统扫描期间文件的发现顺序，依次将归档请求中的所有文件归档到同一个归档卷中。您可以通过指定站点的特定标准，来控制按什么顺序对文件进行归档，以及如何将它们分布到不同的卷中。这些标准称为**归档集参数**，其检验顺序依次为：-reserve、-join、-sort、-rsort（执行逆向排序）以及 -drives。有关这些参数的更多信息，请参见 archiver.cmd(4) 手册页。

如果归档请求属于已指定 -reserve *owner* 的归档集，则 sam-archiverd 守护进程将依据文件的目录路径、用户名或组名来排列归档请求中的文件。此操作由归档集的 -reserve 参数控制。它首先选择属于第一个 *owner* 的文件进行归档。剩余文件将在以后进行归档。

如果归档请求属于已指定 -join *method* 的归档集，则 sam-archiverd 守护进程将根据指定的 -join *method* 对文件进行分组。如果还指定了 -sort 或 -rsort *method* ，则 sam-archiverd 守护进程将根据 -sort 或 -rsort *method* 排列每个组中的文件。此时，归档请求既进行了组合，也进行了排序。

对于以后的编辑和调度进程，每一个组内的所有组合文件均被视为单个文件。

如果归档请求属于已指定 -sort 或 -rsort *method* 的归档集，则 sam-archiverd 守护进程将根据 -sort 或 -rsort 参数指定的排序方法排列文件。 sam-archiverd 守护进程通常会根据排序方法、归档时限、大小或目录位置将文件组合在一起，具体视排序方法而定。默认情况下，归档请求不会进行排序，因此归档程序将根据在文件系统扫描期间发现文件的顺序对文件进行归档。

sam-archiverd 守护进程可以确定文件是处于联机状态还是脱机状态。如果归档请求中既包含联机文件，也包含脱机文件，则会首先选择联机文件进行归档。

如果未要求归档请求按一定的排序方法进行组合或排序，则脱机文件会按归档副本所在的卷来排列。这可确保同一卷中每一个归档集内的所有文件可以按它们在介质上的排列顺序同时登台。在为脱机文件创建多份归档副本期间，脱机文件不会被释放，直到创建完所有要求的副本。与第一个文件处于同一卷中且要登台的所有文件都将被选作首先进行归档的文件。

请注意，在对脱机文件进行归档时，使用 -join、 -sort 或 -rsort 参数可能会对性能造成负面影响。这是因为要归档的文件的顺序可能与脱机文件所需的归档卷顺序不符。因此，建议您仅在创建第一份归档副本时使用 -join、 -sort 或 -rsort 参数。开始创建其他副本时，如果归档介质有足够的空间，则其他副本会尽可能地采用第一份副本的归档顺序进行归档。

归档请求将输入至 sam-archiverd 守护进程的调度队列中。

步骤 3：调度归档请求

sam-archiverd 守护进程中的调度程序将在出现以下情况之一时立即执行：

- 归档请求已输入至调度队列中。
- 已完成某个归档请求的归档。
- 从目录服务器收到介质状态发生变化的消息。
- 收到更改归档程序状态的消息。

调度队列中的归档请求按优先级排列。调度程序每次执行时，均会检查所有归档请求，以确定是否可以将它们分配至 sam-arcopy 进程，从而将文件复制到归档介质中。

此时，必须存在用于创建文件副本的驱动器。必须存在可供归档集使用的卷，并且它们有足够的空间来容纳归档请求中的文件。

驱动器

如果归档集指定了 -drives 参数，则 sam-archiverd 守护进程会将归档请求中选定的文件分配至多个驱动器中。如果此时可用的驱动器数量少于 -drives 参数指定的数量，则使用实际可用的数量。

如果归档请求中的文件大小总量小于 -drivemin 值，则只使用一个驱动器。-drivemin 值可以是 -drivemin 参数指定的值，也可以是 archmax 值。

archmax 值是由 -archmax 参数指定的值或为此介质定义的值。有关 -archmax 参数和 archmax= 指令的更多信息，请参见 archiver.cmd(4) 手册页。

如果归档请求中的文件大小总量大于 -drivemin 值，则会执行以下计算：*drive_count* = *total_size* / *drivemin*。如果 *drive_count* 小于 -drives 参数所指定的驱动器数，则使用 *drive_count* 所指定的驱动器数。

不同的驱动器归档文件所花的时间各不相同。可以使用 -drivemax 参数以获得更好的驱动器利用率。 -drivemax 参数要求您在重新调度驱动器接受更多数据前，指定将写入该驱动器的最大字节数。

卷

系统必须存在一个或多个具有足够空间的卷，来容纳归档请求中的全部文件或至少一部分文件。如果归档集最近使用过的卷上有足够的空间，归档程序将使用这个卷。另外，该卷不应是归档程序正在使用的卷。

如果可用于归档集的卷正在使用中，则归档程序会选择其他卷。但只有在未指定 -fillvsns 参数时，此原则才适用。如果指定了该参数，则归档请求不能另行调度。

如果归档请求太大，无法装入一个卷中，则系统会选择只将这个卷所能容纳的文件归档于其中。如果归档请求包含的文件太大，无法装入一个卷中，并且未为归档请求选择卷溢出功能，则这些文件无法归档。此时，系统会将一则说明此情况的消息发送至日志中。

您可以使用 -ovflmin 参数为归档集指定卷溢出功能，或使用 ovflmin= 指令为介质指定卷溢出功能。有关 -ovflmin 参数和 ovflmin= 指令的更多信息，请参见 archiver.cmd(4) 手册页。此参数 ovflmin 用于确定文件溢出介质的最小大小。为归档集指定的 ovflmin 优先于为介质定义的 ovflmin。如果文件的大小小于 ovflmin，则文件无法归档。此时，系统会将一则说明此情况的消息发送至日志中。

如果文件的大小大于 ovflmin，则会根据需要分配其他卷。系统将按容量逐渐减少的顺序选择其他卷，以尽可能地减少文件所占用的卷数量。

如果没有可用于归档请求的卷，则归档请求会等待。

在确定某个归档请求的调度优先级时，除依据步骤 1 中得出的归档优先级之外，系统还将参照某些其他属性，例如文件是处于联机状态还是处于脱机状态。有关自定义属性乘数的更多信息，请参见 archiver.cmd(4) 手册页中所述的 -priority 参数。

对于每一个归档请求，sam-archiverd 守护进程均会通过计算归档优先级和各种与系统资源属性相关联的乘数之和，来确定其调度优先级。这些资源属性与以下各项相关：归档请求排队的时间 （秒数）；归档进程中使用的第一个卷是否已载入驱动器；以及其他方面。

使用经调整的优先级， sam-archiverd 守护进程指定每一个可进行复制的归档请求。

步骤 4：对归档请求中的文件进行归档

当归档请求已经准备就绪，可以进行归档时， sam-archiverd 守护进程将检查每一个归档请求，标记归档文件 (tarball) 的界限，以使每一个归档文件的大小不超过 -archmax *target_size* 参数指定的值。如果单个文件大于 *target_size*，它将成为归档文件中的唯一文件。

对于每一个归档请求和要使用的每一个驱动器， sam-archiverd 守护进程均会将归档请求分配至 sam-arcopy 进程，以便将文件复制到归档介质。如果单个文件大于 *target_size*，它将成为归档文件中的唯一文件。归档信息会输入至 inode。

如果已启用归档日志功能，则会创建归档日志条目。

如果文件已登台，系统会释放其磁盘空间。此进程会持续运行，直到列表中的所有文件归档完毕。

多种错误以及文件状态改变都会导致文件复制失败。这其中包括读取磁盘高速缓存或向卷写入数据时发生的错误。另外，文件状态的改变也会导致文件复制失败，其中包括在选择文件后对文件进行了修改，打开文件并写入了数据，以及文件被删除等。

sam-arcopy 进程退出后， sam-archiverd 守护进程将检查归档请求。如果有文件尚未归档，则会重新编辑归档请求。

默认输出范例

代码示例 3-1 显示了运行 archiver(1M) -l 命令的输出范例。

代码示例 3-1 archiver(1M) -l 命令的输出

```
# archiver

Archive media:
default:mo
media:mo archmax:5000000
media:lt archmax:50000000
Archive devices:
device:mo20 drives_available:1 archive_drives:1
device:lt30 drives_available:1 archive_drives:1
Archive file selections:
```

代码示例 3-1 archiver(1M) -l 命令的输出 （续）

```
Filesystem samfs1:
samfs1  Metadata
    copy:1  arch_age:240
big  path:. minsize:512000
    copy:1  arch_age:240
all  path:
    copy:1  arch_age:30
Archive sets:
all
    copy:1 media:mo
big
    copy:1 media:lt
samfs1
    copy:1 media:mo
```

归档程序的守护进程

sam-archiverd 守护进程用于调度归档活动。sam-arfind 进程用于将需要归档的文件分配至归档集。 sam-arcopy 进程用于将需要归档的文件复制到选定的卷。

一旦 Sun StorEdge SAM-FS 开始活动， sam-fsd 就会启动 sam-archiverd 守护进程。sam-archiver 守护进程将执行 archiver(1M) 命令，以读取 archiver.cmd 文件并建立用于控制归档操作的表。它将为每一个已安装的文件系统启动 sam-arfind 进程；同样，如果未安装文件系统，则会停止相关的 sam-arfind 进程。然后， sam-archiverd 进程将监视 sam-arfind 的运行状态，并处理来自操作员或其他进程的信号。

归档日志文件和事件日志

sam-arfind 和 sam-arcopy 进程可以生成日志文件，该文件中包含每一个已归档文件或自动取消归档的文件的有关信息。日志文件连续地记录归档操作。您可以使用日志文件查找文件的早期副本，这些副本是传统备份的产物。

默认情况下，系统不会生成此文件。您可以在 archiver.cmd 文件中插入 logfile= 指令，以指明创建日志文件，并指定其名称。有关日志文件的更多信息，请参见本章第 46 页 “使用归档程序指令” 和 archiver.cmd(4) 手册页。

归档程序使用 syslog 工具和 archiver.sh，在日志文件中记录警告及信息消息。

代码示例 3-2 显示了一些摘自归档程序日志文件的示例行，这些行包括了每个字段的定义。

代码示例 3-2 归档程序日志文件的内容

```
A 2001/03/23 18:42:06 mo 0004A arset0.1 9a089.1329 samfs1
118.51162514 t0/fdn f 0 56
A 2001/03/23 18:42:10 mo 0004A arset0.1 9aac2.1 samfs1 189.53
1515016 t0/fae f 0 56
A 2001/03/23 18:42:10 mo 0004A arset0.1 9aac2.b92 samfs1 125.53
867101 t0/fai f 0 56
A 2001/03/23 19:13:09 lt SLOT22 arset0.2 798.1 samfs1 71531.14
1841087 t0/fhh f 0 51
A 2001/03/23 19:13:10 lt SLOT22 arset0.2 798.e0e samfs1 71532.12
543390 t0/fhg f 0 51
A 2003/10/23 13:30:24 dk DISK01/d8/d16/f216 arset4.1 810d8.1 qfs2
119571.301 1136048 t1/fileem f 0 0
A 2003/10/23 13:30:25 dk DISK01/d8/d16/f216 arset4.1 810d8.8ad
qfs2 119573.295 1849474 t1/fileud f 0 0
A 2003/10/23 13:30:25 dk DISK01/d8/d16/f216 arset4.1 810d8.16cb
qfs2 119576.301 644930 t1/fileen f 0 0
A 2003/10/23 13:30:25 dk DISK01/d8/d16/f216 arset4.1 810d8.1bb8
qfs2 119577.301 1322899 t1/fileeo f 0 0
```

表 3-1 按从左至右的顺序列出了上面示例中各字段的内容。

表 3-1 归档程序日志文件的字段

字段	内容
1	归档活动，如下所示： • A 表示已归档。 • R 表示已重新归档。 • U 表示已取消归档。
2	归档操作发生的日期，格式为 *yyyy*/*mm*/*dd*。
3	归档活动发生的时间，格式为 *hh*:*mm*:*ss*。
4	归档介质类型。有关介质类型的信息，请参见 mcf(4) 手册页。
5	VSN。对于可移除介质卡盒，这是卷序列名。对于磁盘归档，这是磁盘卷名以及 tar(1) 归档文件的路径。
6	归档集和副本份数。

表 3-1 归档程序日志文件的字段 （续）

字段	内容
7	归档文件 （tar(1) 文件）在介质上的起始物理位置和归档文件中的文件偏移量 （采用十六进制表示）。
8	文件系统名。
9	Inode 编号和世代编号。世代编号是在索引编号被重新使用后生成的一个附加编号，它与索引编号一起用来标识使用的唯一性。
10	在文件仅写入一个卷时，表示文件大小。在文件写入多个卷时，表示部分文件的大小。
11	相对于文件系统安装点的文件路径以及名称。
12	文件类型，如下所示： • d 表示目录。 • f 表示普通文件。 • l 表示符号链接。 • R 表示可移除的介质文件。 • I 表示段索引。 • S 表示数据段。
13	溢出文件的一部分或段。如果这个文件是溢出文件，则该值不为零而对于其他所有文件类型，该值均为零。
14	文件归档至的驱动器的设备序号。

关于 archiver.cmd 文件

archiver.cmd 文件用于控制归档程序的行为方式。默认情况下，只要一启动 sam-fsd 并安装 Sun StorEdge SAM-FS 文件系统，归档程序就开始运行。归档程序默认情况下执行以下操作：

- 将所有文件归档至所有可用的卷。
- 所有文件的归档时限为 4 分钟。
- 归档时间间隔为 10 分钟。

您很可能希望对归档程序的操作进行自定义，以满足站点的特殊归档要求。这些操作由归档程序命令文件 (archiver.cmd) 中的指令控制。

注 – 如果不采用以下过程中介绍的方法，则还可以使用 File System Manager 软件来创建或修改 `archiver.cmd` 文件。在 File System Manager 中对归档配置所做的更改将会自动对 `archiver.cmd` 文件进行相应的更改。有关更多信息，请参见 File System Manager 联机帮助。

▼ 创建或修改 archiver.cmd 文件并传播更改

1. **确定是否需要编辑 `archiver.cmd` 文件，或是否需要编辑临时的 `archiver.cmd` 文件。（可选）**

   如果您具有 `/etc/opt/SUNWsamfs/archiver.cmd` 文件，且系统已在对文件进行归档，请执行此步骤。可考虑将 `archiver.cmd` 文件复制到一个临时位置，以利于对它进行编辑并在正式使用前进行测试。

2. **使用 `vi`(1) 或其他编辑器来编辑 `archiver.cmd` 文件或其临时文件。**

   为了控制您的站点的归档操作，可根据您的需要添加指令。有关可在这个文件中包括哪些指令的信息，请参见第 46 页 “使用归档程序指令”和第 81 页 “关于磁盘归档”。

3. **保存并关闭 `archiver.cmd` 文件或其临时文件。**

4. **使用 `archiver`(1M) `-lv` 命令检验文件的正确性。**

   无论您何时更改 `archiver.cmd` 文件，均应使用 `archiver`(1M) 命令检查其中是否存在语法错误。使用如下所示的 `archiver`(1M) 命令，在当前 Sun StorEdge SAM-FS 系统中检验 `archiver.cmd` 文件：

   ```
   # archiver -lv
   ```

   上面的命令将列出所有选项，并将 `archiver.cmd` 文件、卷、文件系统内容和错误的列表写入标准输出文件 (`stdout`)。出现错误时，归档程序将停止运行。

   默认情况下， `archiver`(1M) 命令会检验 `/etc/opt/SUNWsamfs/archiver.cmd` 文件中的错误。如果您修改的是 `archiver.cmd` 文件的临时文件，则可在 `archiver`(1M) 命令中使用 `-c` 选项并提供这个临时文件的名称，以在正式使用之前对其进行检验。

5. **重复步骤 2、步骤 3 以及步骤 4，直到文件没有错误。**

   在执行下一步骤前，必须更正所有错误。如果归档程序发现 `archiver.cmd` 文件中有错误，则它不会对任何文件进行归档。

6. **保存并关闭 `archiver.cmd` 文件。**

7. **将这个临时文件移至** `/etc/opt/SUNWsamfs/archiver.cmd`。**（可选）**

   仅当您使用了临时文件时，才需要执行此步骤。

8. **使用** `samd`**(1M)** `config` **命令传播文件更改并重新启动系统。**

```
# samd config
```

archiver.cmd 文件

`archiver.cmd` 文件由以下类型的指令组成：

- 常规指令
- 归档集分配指令
- 归档集指令
- VSN 池指令
- VSN 关联指令

这些指令由从 `archiver.cmd` 文件读取的文本行组成。每一行指令包含一个或多个字段，它们由空格或制表符隔开。井字符 (#) 后面的任何文本均被视为注释，并且不会受到检查。在行的末尾添加一个反斜杠 (\)，可使该行续接至下一行。

`archiver.cmd` 文件中的某些指令会要求您指定时间单元或字节单元。要指定这些单元，请将第 43 页“表 3-2 archiver.cmd 文件指令单元”中用于代表单元的字母作为数字的后缀。

表 3-2 `archiver.cmd` 文件指令单元

单元后缀	含义
时间后缀：	
`s`	秒。
`m`	分钟， 60 秒。
`h`	小时， 3,600 秒。
`d`	天， 86,400 秒。
`w`	周， 604,800 秒。
`y`	年， 31,536,000 秒。

表 **3-2** archiver.cmd 文件指令单元 （续）

单元后缀	含义
大小后缀：	
b	字节。
k	千字节 (KB)， 2**10 或 1,024 字节。
M	兆字节 (MB)， 2**20 或 1,048,576 字节。
G	千兆字节 (GB)， 2**30 或 1,073,741,824 字节。
T	兆兆字节 (TB)， 2**40 或 1,099,511,627,776 字节。
P	千兆兆字节 (PB)， 2**50，或 1,125,899,906,842,624 字节。
E	兆兆兆字节 (EB)， 2**60，或 1,152,921,504,606,846,976 字节。

archiver.cmd 文件示例

代码示例 3-3 显示了 archiver.cmd 文件示例。右侧的注释指出了各种指令类型。

代码示例 3-3 archiver.cmd 文件示例

```
interval = 30m                      # General directives
logfile = /var/opt/SUNWsamfs/archiver/archiver.log

fs = samfs1                         # Archive Set Assignments
no_archive tmp
work work
    1 1h
    2 3h
images images -minsize 100m
    1 1d
    2 1w
samfs1_all  .
    1 1h
    2 1h

fs = samfs2                         # Archive Set Assignments
no_archive tmp
system . -group sysadmin
    1 30m
    2 1h
samfs2_all  .
    1 10m
    2 2h

params                              # Archive Set Directives
allsets -drives 2
images.1 -join path -sort size
endparams
vsns                                # VSN Associations
samfs1.1   mo      optic-2A
samfs1.2   lt      TAPE01
work.1     mo      optic-[3-9][A-Z]
work.2     lt      .*
images.1   lt      TAPE2[0-9]
images.2   lt      TAPE3[0-9]
samfs1_all.1       mo.*
samfs1_all.2       lt.*
samfs2.1   mo      optic-2A
samfs2.2   lt      TAPE01
system.1   mo      optic08a optic08b
system.2   lt      ^TAPE4[0-1]
samfs2_all.1       mo.*
samfs2_all.2       lt.*
endvsns
```

使用归档程序指令

以下几节对 archiver.cmd 指令进行介绍。这些指令是：

全局归档指令

全局指令用于控制归档程序的整体操作，并可用于优化站点配置的归档程序操作。可以在 archiver.cmd 文件中直接添加全局指令，或使用 File System Manager 软件指定全局指令。有关使用 File System Manager 设置全局指令的更多信息，请参见 File System Manager 联机帮助。

archiver.cmd 文件中的全局指令可以通过第二个字段中的等号 (=) 或缺少其他字段来识别。

在 archiver.cmd 文件中，全局指令必须位于所有 fs= 指令前面。fs= 指令是那些专用于特定文件系统的指令。如果归档程序检测到全局指令位于 fs= 指令后面，则会发出一则消息。

archivemeta 指令：控制是否对元数据进行归档

archivemeta 指令控制是否对文件系统元数据进行归档。如果您文件系统中的文件经常移动，且目录结构经常发生更改，则应该对元数据进行归档。但是，如果目录结构非常稳定，则可以禁用元数据归档，并减少可移除介质驱动器所执行的操作，因为载入和卸载卡盒会对元数据进行归档。默认情况下，会对元数据进行归档。

此指令的格式如下：

```
archivemeta = state
```

其中的 *state*，用于指定状态为 on 还是 off。默认设置是 on。

元数据的归档方式不尽相同，这取决于您所使用的超级块是第 1 版还是第 2 版，具体如下：

- 如果使用的是第 1 版的文件系统，归档程序会将目录、可移除的介质文件、段 inode 以及符号链接归档为元数据。
- 如果使用的是第 2 版的文件系统，可移除的介质文件和符号链接会存储在 inode 中，而不是存储在数据块中。归档程序不会对它们进行归档。归档程序只将目录和段 inode 归档为元数据。符号链接则归档为数据。

archmax 指令：控制归档文件的大小

archmax 指令用于指定归档文件的最大容量。将用户文件组合在一起，形成归档文件。达到 *target_size* 值后，将不能再向归档文件中添加用户文件。大型用户文件将写入单个归档文件中。

要更改默认值，请使用以下指令：

```
archmax=media target_size
```

表 3-3 archmax 指令的参数

参数	含义
media	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
target_size	指定归档文件的最大大小。归档文件的最大大小与介质有关。默认情况下，写入光盘的归档文件不得超过 5 MB。对于磁带，归档文件的最大默认大小为 512 MB。

将归档文件的大小设置为较大的值或较小的值都各有优缺点。例如，在使用磁带进行归档时，将 archmax 设置成较大的值，可以减少磁带机停止和启动的次数。但是，在写入较大的归档文件时，有可能提前到达磁带末尾，因而浪费了大量磁带空间。一般而言， archmax 的设置值不应超过介质容量的百分之五。例如，可以使用如下所示的 archmax 指令来设置 20 GB 的磁带：

```
archmax=sg 1G
```

此外，您还可为单个归档集设置 archmax 指令。

bufsize 指令：设置归档程序缓冲区大小

默认情况下，系统使用内存缓冲器将需要归档的文件复制到归档介质。可以使用 bufsize 指令来设置非默认的缓冲区大小和锁定缓冲区 （可选）。这些操作可以改善系统的性能。您可以尝试不同的 *buffer_size* 值以确定最适合的缓冲区大小。

此指令的格式如下：

```
bufsize=media  buffer_size  [ lock ]
```

表 3-4 bufsize 指令的参数

参数	含义
media	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
buffer_size	指定一个 2 至 32 的值。默认值为 4。这个值乘以该介质类型的 *dev*_blksize 值，计算结果即为所使用的缓冲区大小。您可以在 defaults.conf 文件中指定 *dev*_blksize 的值。有关此文件的更多信息，请参见 defaults.conf(4) 手册页。
lock	lock 参数指明归档程序在创建归档副本时是否使用锁定的缓冲区。如果指定 lock，则归档程序将在 sam-arcopy(1M) 操作期间在内存中的归档缓冲区上设置文件锁定。这可以避免由于为每一个 I/O 请求锁定和取消锁定缓冲区而造成的开销，从而减少占用系统 CPU 的时间。 仅在配有大量内存的大型系统上，才有必要指定 lock 参数。如果内存不足，则可能会导致内存用尽。 只有已为需要归档的文件启用直接 I/O 时，lock 参数才有效。默认情况下，不会指定 lock 参数，文件系统会在所有直接 I/O 缓冲区上设置锁定（也包括用于归档的缓冲区）。有关启用直接 I/O 的更多信息，请参见 setfa(1) 手册页、sam_setfa(3) 库例程手册页，或 mount_samfs(1M) 手册页中介绍的 -O forcedirectio 选项。

例如，您可以按以下方式在 archiver.cmd 文件的指令行中指定该指令：

```
bufsize=od  7  lock
```

您可以使用 -bufsize 和 -lock 归档集副本参数，以归档集为单元设置缓冲区的大小和锁定。有关更多信息，请参见第 64 页 “归档集副本参数”。

drives 指令：控制用于归档的驱动器数

默认情况下，归档程序使用自动化库中的所有驱动器进行归档。要限制归档程序使用的自动化库中的驱动器数量，请使用 drives 指令。

此指令的格式如下：

```
drives=auto_lib  count
```

表 3-5 drives 指令的参数

参数	含义
auto_lib	mcf 文件中定义的自动化库的系列集名。
count	用于归档活动的驱动器数量。

另请参见第 66 页 “指定归档请求的驱动器数：-drivemax、 -drivemin 和 -drives”，其中介绍了 -drivemax、 -drivemin 和 -drives 归档集副本参数。

examine 指令：控制归档扫描

新文件以及已更改的文件都是备选的归档文件。归档程序会通过以下某一方法来查找此类文件：

- 连续归档。当进行连续归档时，归档程序对文件系统进行处理，检测其中刚刚发生更改的文件。
- 基于扫描的归档。而对于基于扫描的归档，归档程序则会定期扫描文件系统，寻找需要归档的文件。

examine 指令控制归档程序是执行连续归档，还是执行基于扫描的归档，具体如下：

```
examine=method
```

可以为 *method* 指定的关键字如表 3-6 所示。

表 3-6 examine 指令 *method* 参数的值

method 值	含义
noscan	指定连续归档。在执行初次扫描后，仅当目录内容发生更改以及需要进行归档时，才会对其进行扫描。但不对目录和 inode 信息进行扫描。这种归档方法的性能要好于基于扫描的归档，尤其是当文件系统中文件的数量大于 1,000,000 时。默认设置。
scan	指定基于扫描的归档。第一次的文件系统扫描是执行目录扫描。接下来对 inode 进行扫描。
scandirs	指定仅对目录执行基于扫描的归档。指定后，如果归档程序查找到具有 no_archive 属性的目录，将不对其进行扫描。这种目录中含有未更改的文件，跳过这种目录这可以大大缩短归档扫描时间。
scaninodes	指定仅对 inode 执行基于扫描的归档。

interval 指令：指定归档时间间隔

归档程序定期检查所有已安装的 Sun StorEdge SAM-FS 文件系统的状态。检查时间由归档时间间隔控制。**归档时间间隔**是指在每一个文件系统上执行扫描操作的时间间隔。要更改时间间隔，请使用 interval 指令。

注 – 在未设置连续归档时， interval 指令仅启动完全扫描。如果未设置连续归档，并且未指定 startage、 startsize 或 startcount 参数，则归档程序将使用 interval 指令预定扫描。如果已设置连续归档 (examine=noscan)，则 interval 指令将作为默认的 startage 值。

此指令的格式如下：

```
interval=time
```

其中的 *time*，用于指定对文件系统执行扫描操作的时间间隔，单元为秒。默认情况下，*time* 以秒计。默认设置为 interval=600，即 10 分钟。也可以将时间单元指定为分钟、小时等等。有关指定时间单元的信息，请参见第 43 页“表 3-2 archiver.cmd 文件指令单元”。

如果归档程序接收到 samu(1M) 实用程序的 :arrun 命令，则会立即扫描所有文件系统。如果 archiver.cmd 文件中还指定了examine=scan 指令，则会在运行 :arrun 或 :arscan 后进行扫描。

如果为文件系统设置了 hwm_archive 安装选项，则可以自动缩短归档时间间隔。此安装选项指定归档程序在文件系统填满且超过空间占用上限时即开始进行扫描。high=*percent* 安装选项用于设置文件系统空间占用的上限。

有关指定归档时间间隔的更多信息，请参见 archiver.cmd(4) 手册页。有关设置安装选项的更多信息，请参见 mount_samfs(1M) 手册页。

logfile 指令：指定归档程序日志文件

归档程序可以生成一个日志文件，其中包含每一个归档、重新归档或自动取消归档的文件的有关信息。日志文件连续地记录归档操作。要指定日志文件，请使用 logfile 指令。此指令的格式如下：

```
logfile=pathname
```

其中的 *pathname* 用于指定日志文件的绝对路径和名称。默认情况下，系统不会生成此文件。

此外，您还可为单个文件系统设置 logfile 指令。

▼ 备份归档程序日志文件

假定您希望通过将前一天的日志文件复制到另一位置，以实现归档程序日志文件的每日备份。如果您确保在归档程序日志文件关闭时执行复制，则可以达到此目的。换言之，在系统打开归档程序日志文件以向其中写入信息时，您不能执行复制操作。您需要执行以下步骤：

1. **使用 mv(1) 命令在 UFS 中移动归档程序日志文件。**

   这可给予 sam-arfind(1M) 或 sam-arcopy(1M) 一定的操作时间，以完成向归档程序日志文件写入信息。

2. **使用 mv(1) 命令将前一天的归档程序日志文件移至 Sun StorEdge SAM-FS 文件系统。**

notify 指令：重命名事件通知脚本

notify 指令用于将归档程序的事件通知脚本文件的名称设置为 *filename*。此指令的格式如下：

```
notify=filename
```

其中的 *filename*，用于指定包含归档程序事件通知脚本的文件名称，或指定该文件的完整路径。

默认文件名如下所示：

```
/etc/opt/SUNWsamfs/scripts/archiver.sh
```

归档程序执行此脚本，来根据特定站点的要求处理各种事件。此脚本通过第一个参数的关键字进行调用。其关键字包括：emerg、alert、crit、err、warning、notice、info 和 debug。

其他参数在默认脚本中加以说明。有关更多信息，请参见 archiver.sh(1M) 手册页。

ovflmin 指令：控制卷溢出功能

卷溢出是指允许在多个卷上归档文件的过程。在 archiver.cmd 文件中使用 ovflmin 指令即可启用卷溢出功能。当文件的大小超过 ovflmin 指令的 *minimum_file_size* 值时，归档程序会将文件的另一部分写入至另一个同类型的可用卷中 （如有必要）。写入至每一个卷的文件部分称为**片段**。

注 – 使用卷溢出功能之前，请务必充分理解卷溢出的含义。只有在全面检验卷溢出对您的站点所产生的影响后，才可以使用卷溢出功能。文件归档在多个卷上会严重地加大故障恢复操作和回收操作的难度。

归档程序通过 ovflmin 指令来控制卷溢出功能。ovflmin 指令用于指定允许文件溢出卷的最小大小。默认情况下，归档程序会禁用卷溢出功能。

此指令的格式如下：

```
ovflmin = media  minimum_file_size
```

表 3-7 ovflmin 指令的参数

参数	含义
media	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
minimum_file_size	指定文件溢出时的最小大小。

示例 1。假设许多文件归档在 mo 介质卡盒中，并且产生严重的碎片 （例如 25%）。由于这些文件不能占用卷的整个空间，因而导致每个卷中存有大量的未用空间。为了更好地利用卷内空间，您需要将 mo 介质的 ovflmin 设置成略小于最小文件大小的值。以下指令将其设置为 150 MB：

```
ovflmin=mo 150m
```

请注意，在本示例中启用卷溢出功能会造成在归档和登台文件时需要载入两个卷。

此外，您还可为单个归档集设置 ovflmin 指令。

示例 2。 sls(1) 命令可列出归档副本，并显示每个 VSN 上文件的每个片断。代码示例 3-4 显示了一个文件的归档程序日志文件以及 sls -D 命令的输出，这个文件名为 file50，它是一个归档在多个卷上的大型文件。

代码示例 3-4 归档程序日志文件示例

```
A  97/01/13  16:03:29  lt  DLT000  big.1  7eed4.1  samfs1  13.7
477609472  00 big/file50  0  0

A  97/01/13  16:03:29  lt  DLT001  big.1  7fb80.0  samfs1  13.7
516407296  01 big/file50  0  1

A  97/01/13  16:03:29  lt  DLT005  big.1  7eb05.0  samfs1  13.7
505983404  02 big/file50  0  2
```

代码示例 3-4 显示了 file50 归档在三个卷上，其 VSN 分别为 DLT000、 DLT001 和 DLT005。每个片段在卷上的位置和大小分别显示在第七个和第十个字段中，它们的值与另外显示的 sls -D 输出中的值相匹配。有关归档日志条目的完整描述，请参见 archiver(1M) 手册页。

代码示例 3-5 显示了 sls -D 命令及其输出。

代码示例 3-5 sls(1M) -D 命令及其输出

```
# sls -D file50
file50:
  mode: -rw-rw----  links:   1  owner: gmm    group: sam
  length: 1500000172  admin id:      7  inode: 1407.5
  offline;   archdone;  stage -n
  copy1: ---- Jan 13 15:55            lt
    section 0:  477609472     7eed4.1    DLT000
    section 1:  516407296     7fb80.0    DLT001
    section 2:  505983404     7eb05.0    DLT005
  access:      Jan 13 17:08  modification: Jan 10 18:03
  changed:     Jan 10 18:12  attributes:   Jan 13 16:34
  creation:    Jan 10 18:03  residence:    Jan 13 17:08
```

卷溢出文件不能生成校验和。有关使用校验和的更多信息，请参见 ssum(1) 手册页。

注 – 请记住，当使用卷溢出功能时，一旦出现故障，恢复卷溢出数据是非常困难的。有关如何恢复这类文件的信息，请参见 《Sun StorEdge SAM-FS 故障排除指南》中的示例。有关更多信息，请参见 request(1) 手册页。

---

wait 指令：延迟归档程序的启动

wait 指令命令归档程序等待来自 samu(1M) 或 File System Manager 的启动信号。当收到此信号后，归档程序才开始典型的归档操作。默认情况下，归档程序在由 sam-fsd(1M) 启动后即开始归档操作。要延迟归档操作，请使用 wait 指令。此指令的格式如下：

```
wait
```

此外，您还可为单个文件系统设置 wait 指令。

文件系统指令

可在 archiver.cmd 文件的常规指令后面添加专用于特定文件系统的 fs= 指令。遇到 fs= 指令后，归档程序假定后面的所有指令均仅适用于特定的文件系统。

可以按照以下几节所述通过编辑 archiver.cmd 文件来指定 fs= 指令，或者通过使用 File System Manager 软件来指定它们。有关更多信息，请参见 File System Manager 联机帮助。

fs 指令：指定文件系统

默认情况下，归档控制指令适用于所有文件系统。不过，您可以将某些控制指令的适用范围限定在特定的文件系统上。要指定特定的文件系统，可使用 fs 指令。此指令的格式如下：

```
fs=fsname
```

其中的 *fsname*，用于指定在 mcf 文件中定义的文件系统名。

这些指令后面出现的常规指令和归档集相关指令仅适用于指定的文件系统，直到出现下一个 fs= 指令。例如，您可以使用此指令为每一个文件系统指定不同的日志文件。

其他文件系统指令

有几个指令既可以指定为适用于所有文件系统的全局指令，也可以指定为专用于某一文件系统的指令。无论在何处指定，它们的效用都是相同的。这几个指令如下：

- interval 指令。有关此指令的更多信息，请参见第 50 页 “interval 指令：指定归档时间间隔”。
- logfile 指令。有关此指令的更多信息，请参见第 50 页 “logfile 指令：指定归档程序日志文件”。
- wait 指令。有关此指令的更多信息，请参见第 53 页 “wait 指令：延迟归档程序的启动”。

归档集分配指令

默认情况下，文件将作为根据文件系统命名的归档集的一部分进行归档。不过，您可以指定其他归档集以包含具有类似特征的文件。如果某个文件不属于您指定的任何归档集，则它将作为根据文件系统命名的默认归档集的一部分进行归档。

可以按照以下几节所述通过直接编辑 archiver.cmd 文件来创建归档集，或者使用 File System Manager 软件来创建归档集。在 File System Manager 中， *archive policy* 用于定义归档集。有关更多信息，请参见 File System Manager 联机帮助。

分配归档集

归档集成员指令可以将具有类似特征的文件分配至同一个归档集。这些指令的语法与 find(1) 命令类似。每一个归档集分配指令均采用以下格式：

```
archive_set_name  path  [search_criteria1  search_criteria2  ...  ]  [file_attributes]
```

表 **3-8** 归档集分配指令的参数

参数	含义
archive_set_name	您的站点为归档集定义的名称。它必须是归档集分配指令中的第一个字段。归档集名通常暗示属于该归档集的文件的特征。指定归档集名时，只能使用 26 个英文字母、数字 (0-9) 和下划线字符 (_)，而不得使用其他特殊字符或空格。归档集名的第一个字符必须是字母。 要阻止对某些文件进行归档，请将 *archive_set_name* 指定为 `no_archive`。
path	相对于文件系统安装点的路径。这使得归档集成员指令可适用于多个 Sun StorEdge SAM-FS 文件系统。如果要使该路径包括文件系统中的所有文件，请在路径字段中输入句点 (.)。不允许在路径的开头使用斜杠 (/)。 *path* 所指定目录中的文件及其子目录均视为属于该归档集。
search_criteria1 *search_criteria2*	可以指定零个、一个或多个 *search_criteria* 参数。指定搜索标准的目的是根据文件大小、文件所有权以及其他要素限制归档集的范围。有关可用的 *search_criteria* 参数的信息，请参见以下几节。
file_attributes	可以指定零个、一个或多个 *file_attributes* 参数。当 `sam-arfind` 进程在归档期间扫描文件系统时，将为文件设置这些文件属性。

示例 1。代码示例 3-6 显示了典型的归档集成员指令。

代码示例 3-6 归档集成员指令

```
hmk_files     net/home/hmk      -user hmk
datafiles     xray_group/data   -size 1M
system        .
```

示例 2。将文件归入名为 `no_archive` 归档集中，可以阻止归档程序对这些文件进行归档。代码示例 3-7 显示了阻止对 `tmp` 目录的各级子目录下的文件进行归档的指令行，而且不管 `tmp` 目录在文件系统中的位置如何，都是如此。

代码示例 3-7 阻止归档的归档指令

```
fs = samfs1
no_archive tmp
no_archive . -name .*/tmp/
```

以下几节介绍了可以指定的 *search_criteria* 参数。

文件大小 *search_criteria*：-access 和 -nftv

可使用 -access *age* 特征来指定用于确定归档集成员的文件时限。如果使用这个 *search_criteria*，访问时间早于 *age* 的文件将被重新归档至其他介质中。其中的 *age*，用于指定一个整数，这个整数的后缀是表 3-9 中所示的某个字母。

表 3-9 -access *age* 的后缀

字母	含义
s	秒
m	分钟
h	小时
d	天
w	周
y	年

例如，可使用这个指令来指定将长时间未使用的文件重新归档至一个较为便宜的介质中。

在确定时限时，归档程序将验证文件的访问和修改时间，以确保这些时间大于或等于文件的创建时间，且小于或等于文件的检查时间。这样做是为了提供正确的归档功能或取消归档功能。不过，对于已“迁移”至某个目录中的文件，此验证可能不会产生预期的效果。在这种情况下，可以使用 -nftv（No File Time Validation, 非文件时间验证）参数阻止对文件的访问和修改时间进行验证。

文件大小 *search_criteria*：-minsize 和 -maxsize

可使用 -minsize *size* 和 -maxsize *size* 特征，通过文件的大小来确定归档集成员。其中的 *size*，用于指定一个整数，这个整数的后缀是表 3-10 中所示的某一字母。

表 3-10 -minsize 和 -maxsize *size* 的后缀

字母	含义
b	字节
k	千字节 (KB)
M	兆字节 (MB)
G	千兆字节 (GB)
T	兆兆字节 (TB)
P	千兆兆字节 (PB)
E	兆兆兆字节 (EB)

示例。本示例中的命令行指定所有介于 500 KB 和 100 MB 之间的文件均属于归档集 `big_files`。超过 100 MB 的文件属于归档集 `huge_files`。代码示例 3-8 显示了这些命令行。

代码示例 3-8 `-minsize` 和 `-maxsize` 指令的使用示例

```
big_files  .  -minsize 500k  -maxsize 100M
huge_files .  -minsize 100M
```

所有者和组 *search_criteria*：`-user` 和 `-group`

可使用 `-user` *name* 和 `-group` *name* 特征，通过所有权和组关系来确定归档集成员。代码示例 3-9 显示了这些指令的示例。

代码示例 3-9 使用 -user 和 -group 指令的示例

```
adm_set     .    -user sysadmin
mktng_set   .    -group marketing
```

所有属于用户 `sysadmin` 的文件均属于归档集 `adm_set`，所有组名称为 `marketing` 的文件均属于归档集 `mktng_set`。

使用模式匹配的文件名 *search_criteria*：`-name` *regex*

使用正则表达式可以指定属于某个归档集的文件的名称。作为一项 *search_criteria*，`-name` *regex* 参数指定任何与正则表达式 *regex* 相匹配的完整路径均为归档集的成员。

regex 参数遵守 `regexp`(5) 手册页中列出的约定。请注意，正则表达式不遵守与 UNIX 通配符相同的约定。

在系统内部，所有位于选定目录之下的文件 （及其相对于文件系统安装点的指定路径）均会被列出并一起传递，以进行模式匹配。这使得您可以在 `-name` *regex* 字段中创建模式，以匹配文件名和路径名。

示例

1. 以下指令将归档集 `images` 中的文件限定为那些以 `.gif` 结尾的文件：

```
images  . -name \.gif$
```

2. 以下指令选择以字符 GEO 开头的文件：

```
satellite  . -name /GEO
```

3. 您可以将正则表达式与 no_archive 归档集结合使用。以下指令可阻止对任何以 .o 结尾的文件进行归档：

```
no_archive  . -name \.o$
```

4. 假定您的 archiver.cmd 文件中包含代码示例 3-10 中所示的指令行：

代码示例 3-10 正则表达式示例

```
# File selections.
fs = samfs1
     1 1s
     2 1s
no_archive share/marketing -name fred\.
```

对于此 archiver.cmd 文件，归档程序不会对用户目录或其子目录下的 fred.* 文件进行归档。文件的归档情况如下所述：

- 如果指定了如代码示例 3-10 中所示的指令，则将不对代码示例 3-11 中所示的文件进行归档。

代码示例 3-11 不进行归档的文件 （假定所指定的指令如代码示例 3-10 所示）

```
/sam1/share/marketing/fred.anything
/sam1/share/marketing/first_user/fred.anything
/sam1/share/marketing/first_user/first_user_sub/fred.anything
```

- 如果指定了如代码示例 3-10 中所示的指令，则将对代码示例 3-12 中所示的文件进行归档。

代码示例 3-12 进行归档的文件 （假定所指定的指令如代码示例 3-10 所示）

```
/sam1/fred.anything
/sam1/share/fred.anything
/sam1/testdir/fred.anything
/sam1/testdir/share/fred.anything
/sam1/testdir/share/marketing/fred.anything
/sam1/testdir/share/marketing/second_user/fred.anything
```

5. 假定您的 archiver.cmd 文件中包含代码示例 3-13 中所示的指令行：

代码示例 3-13 archiver.cmd 文件示例

```
# File selections.
fs = samfs1
     1 1s
     2 1s
no_archive share/marketing -name ^share/marketing/[^/]*/fred\.
```

代码示例 3-13 中的 archiver.cmd 文件不会对用户根目录下的 fred.* 文件进行归档。但会对用户子目录和 share/marketing 目录下的 fred.* 文件进行归档。在此情况下，用户根目录刚好是 first_user。本示例中，从 share/marketing/ 到下一个斜杠字符 (/) 之间的任何目录均被视为用户的根目录。文件的归档情况如下所述：

- 如下所示的文件不会被归档：

```
/sam1/share/marketing/first_user/fred.anything
```

- 如果指定了如代码示例 3-13 中所示的指令，则将对代码示例 3-14 中所示的文件进行归档。

代码示例 3-14 进行归档的文件 （假定所指定的指令如代码示例 3-13 所示）

```
/sam1/share/fred.anything
/sam1/share/marketing/fred.anything
/sam1/share/marketing/first_user/first_user_sub/fred.anything
/sam1/fred.anything
/sam1/testdir/fred.anything
/sam1/testdir/share/fred.anything
/sam1/testdir/share/marketing/fred.anything
/sam1/testdir/share/marketing/second_user/fred.anything
/sam1/testdir/share/marketing/second_user/sec_user_sub/fred.any
```

释放和登台 *file_attributes*：-release 和 -stage

您可以通过分别使用 -release 和 -stage 选项，设置与归档集内文件相关联的释放和登台属性。这两种设置都会取代用户以前可能设置的释放或登台属性。

-release 选项的格式如下所示：

```
-release attributes
```

-release 指令的 *attributes* 所遵守的约定与 release(1) 命令相同，具体如表 3-11 所示。

表 3-11 -release 选项

属性	含义
a	完成第一个归档副本后，释放文件占用的磁盘空间。
d	重置为默认设置。
n	不释放文件占用的磁盘空间。
p	释放文件占用的部分磁盘空间。

-stage 选项的格式如下所示：

```
-stage attributes
```

-stage 指令的 *attributes* 所遵守的约定与 stage(1) 命令相同，具体如表 3-12 所示。

表 3-12 -stage 指令的 *attributes*

属性	含义
a	联合登台本归档集中的文件。
d	重置为默认设置。
n	不登台本归档集中的文件。

以下示例显示了如何使用文件名参数和文件属性，来部分释放 Macintosh 资源目录：

```
MACS  . -name .*/\.rscs/  -release p
```

归档集成员冲突

有时，在选择路径和其他文件特征以将文件归入归档集时，可能会造成归档集成员混乱。系统采用以下方式来解决此类情况：

1. 选择归档集中最初的成员定义。
2. 首先选择文件系统本地的成员定义，然后选择全局性的成员定义。
3. 如果某个成员定义与先前的成员定义完全相同，则系统会发出错误通知。

因此，用户应在指令文件的前部设置更加严格的成员定义。

如果对特定文件系统的归档操作进行了控制 （使用 fs=*fsname* 指令），则归档程序在检验全局指令之前，会先检验文件系统的特定指令。这样，文件可以分配至本地归档集（包括 no_archive 归档集），而不是分配至全局归档集。在设置 no_archive 等全局归档集分配时，就默认采用此规则。

代码示例 3-15 显示了一个 archiver.cmd 文件。

代码示例 3-15 可能存在成员冲突的 archiver.cmd 文件

```
no_archive   . -name .*\.o$
fs = samfs1
    allfiles  .
fs = samfs2
    allfiles  .
```

从代码示例 3-15 可以看出，管理员本不希望对以上两个文件系统中的 .o 文件进行归档。但是，由于对本地归档集分配 allfiles 的检验先于全局归档集分配 no_archive，因此仍会对 samfs1 和 samfs2 文件系统中的 .o 文件进行归档。

代码示例 3-16 显示了确保能够不对以上两个文件系统中的 .o 文件进行归档的指令。

代码示例 3-16 更正后的 archiver.cmd 文件

```
fs = samfs1
    no_archive   . -name .*\.o$
    allfiles  .
fs = samfs2
    no_archive   . -name .*\.o$
    allfiles  .
```

归档副本指令

如果不指定归档副本，则归档程序将为归档集中的文件写入单独的归档副本。默认情况下，创建文件副本的归档时限为四分钟。如果您需要多份副本，则必须使用归档副本指令指定所有副本 （包括第一个副本）。

归档副本指令以一个整数 *copy_number* 开头。该数字 （1、 2、 3 或 4）是副本份数。数字后面是一个或多个用于指定该副本归档特征的参数。

归档副本指令必须紧跟在与它们相关的归档集分配指令的后面。每一个归档副本指令均采用以下格式：

```
copy_number [ -release | -norelease ] [archive_age] [unarchive_age]
```

可以按照这里介绍的方法通过编辑 archiver.cmd 文件指定归档副本份数，或者通过使用 File System Manager 软件来指定。有关更多信息，请参见 File System Manager 联机帮助。

以下几节介绍这些归档副本指令参数。

归档之后释放磁盘空间：-release

可以在副本份数后面使用 -release 指令，来指定在创建归档副本之后自动释放的文件所占用的磁盘空间。此选项的格式如下：

```
-release
```

在代码示例 3-17 中，属于组 images 的文件在其归档时限达到 10 分钟时进行归档。在创建第一个归档副本后，系统会释放文件占用的磁盘高速缓存空间。

代码示例 3-17 使用了 -release 指令的 archiver.cmd 文件

```
ex_set . -group images
    1 -release 10m
```

延迟释放磁盘空间：-norelease

您可能希望在创建多份归档副本之后再释放文件占用的磁盘空间。可以使用 -norelease 选项，它将使得系统在所有标记了 -norelease 的副本创建完毕之后，才自动释放磁盘高速缓存。此选项的格式如下：

```
-norelease
```

通过 -norelease 选项可以在所有副本均已归档后使归档集释放其占用的磁盘空间，但是归档集中的所有文件只有在系统调用释放程序并且将它们选择为释放备选文件后才会释放其占用的磁盘空间。

代码示例 3-18 指定了一个名为 vault_tapes 的归档集。系统将为其创建两份副本，并且只有在这两份副本创建完毕之后，才释放与该归档集相关的磁盘高速缓存。

代码示例 3-18 使用了 -norelease 指令的 archiver.cmd 文件

```
vault_tapes
    1 -norelease 10m
    2 -norelease 30d
```

请注意，仅需创建单个归档副本时 -norelease 参数对自动释放磁盘空间无效，因为至少在创建一个副本后才会释放磁盘空间。

同时使用 -release 和 -norelease

如果要确保归档集在所有副本完成归档之后立即释放其占用的磁盘空间，则您可以同时使用 -release 和 -norelease 选项。-release 和-norelease 的组合使用可以使归档程序在使用了这两个选项组合的所有副本创建之后释放该文件。使用这种方法，系统会立即释放磁盘空间，而在单独使用 -norelease 选项的情况下，将会等待调用释放程序。

设置归档时限

您可以通过在该指令的下一个字段中指定归档时限来设置文件的归档时限。归档时限的后缀字符可以指定为 h（小时）或 m（分钟）。第 43 页“表 3-2 archiver.cmd 文件指令单元”完整列出了这些后缀字符及其含义。

在代码示例 3-19 中，目录 data 中的文件将在其归档时限到达 1 个小时后进行归档。

代码示例 3-19 指定了归档时限的 archiver.cmd 文件

```
ex_set data
    1 1h
```

自动取消归档

如果您为文件指定多个归档副本，则可能需要只保留一个归档副本，而自动取消对其他所有副本进行归档。当使用不同归档时限将文件归档至不同介质时，可能发生此类情况。

代码示例 3-20 显示了指定取消归档时限的指令。

代码示例 3-20 指定了取消归档时限的 archiver.cmd 文件

```
ex_set home/users
    1 6m 10w
    2 10w
    3 10w
```

路径 home/users 中文件的第一个副本将在文件修改后六分钟进行归档。当文件的归档时间达到 10 周时，系统会创建第二个和第三个归档副本，并且会取消对第一个副本进行归档。

有关控制取消归档的其他方法，请参见第 71 页 “控制取消归档”。

为元数据指定多份副本

如果需要多份元数据副本，则可以在指令文件中放置副本份数定义，其位置是紧跟在 fs= 指令后。

代码示例 3-21 显示了一个指定了多份元数据副本的 archiver.cmd 文件。

代码示例 3-21 指定了多份元数据副本的 archiver.cmd 文件

```
fs = samfs7
    1 4h
    2 12h
```

在本示例中，系统将在四小时后创建 samfs7 文件系统的元数据的第一份副本，然后在 12 小时后创建第二份副本。

文件系统元数据包括对文件系统中路径名的更改。因此，如果您经常更改目录，则系统会创建新的归档副本。这会造成系统频繁地载入您为元数据指定的卷。

归档集副本参数

archiver.cmd 文件中，归档集参数部分以 params 指令开头，以 endparams 指令结尾。代码示例 3-22 给出了归档集指令的格式。

代码示例 3-22 归档集副本参数的格式

```
params
archive_set_name.copy_number[R] [ -param1 -param2 ...]
.
.
.
endparams
```

表 3-13 归档集副本参数的参数

参数	含义
archive_set_name	您的站点为归档集定义的名称。通常可暗示属于该归档集的文件的特征。可以是 allsets。指定归档集名时，只能使用 26 个英文字母、数字 (0-9) 和下划线字符 (_)，而不得使用其他特殊字符或空格。归档集名的第一个字符必须是字母。
.	句点 (.) 字符。用于分隔 *archive_set_name* 和 *copy_number*。

表 3-13 归档集副本参数的参数 （续）

参数	含义
copy_number	用于定义归档副本份数的整数。可以是 1、 2、 3 或 4。
R	指定所定义的参数是用于定义此归档集要重新归档的副本数。例如，可以在 *-param1* 参数中，通过使用 R 和指定 VSN 将重新归档的副本定向到特定的卷。
-param1 *-param2*	一个或多个参数。以下几小节介绍 params 和 endparams 指令之间可以指定的参数。

可以按照这里介绍的方法通过编辑 archiver.cmd 文件指定归档集副本参数，或者也可以通过使用 File System Manager 软件来指定。有关更多信息，请参见 File System Manager 联机帮助。

伪归档集 allsets 提供了一种为所有归档集设置默认归档集指令的方法。所有 allsets 指令必须位于实际归档集副本指令的前面，其原因是为单个归档集副本设置的参数可以取代由 allsets 指令设置的参数。有关 allsets 归档集的更多信息，请参见 archiver.cmd(4) 手册页。

本部分介绍除磁盘归档参数之外的其他所有归档集处理参数。有关磁盘归档的信息，请参见第 81 页 “关于磁盘归档”。

控制归档文件的大小：-archmax

-archmax 指令用于指定归档集文件的最大容量。格式如下：

```
-archmax target_size
```

这条指令与 archmax 全局指令非常相似。有关这条指令以及 *target_size* 的输入值的信息，请参见第 47 页 “archmax 指令：控制归档文件的大小”。

设置归档程序缓冲区大小：-bufsize

默认情况下，需要归档的文件先存储在缓冲区的内存中，然后再写入归档介质。可以使用 -bufsize 参数指定非默认的缓冲区大小。这些操作可以改善系统的性能。您可以尝试不同的 *buffer_size* 值以确定最适合的缓冲区大小。

此参数的格式如下：

```
-bufsize=buffer_size
```

其中的 *buffer_size*，可用于指定一个 2 至 32 的值。默认值为 4。这个值乘以该介质类型的 *dev*_blksize 值，计算结果即为所使用的缓冲区大小。 *dev*_blksize 的值可以在 defaults.conf 文件中指定。有关此文件的更多信息，请参见 defaults.conf(4) 手册页。

例如，可以按以下方式在 archiver.cmd 文件的命令行中指定该参数：

```
myset.1 -bufsize=6
```

此外，通过指定全局 bufsize=*media buffer_size* 指令，也可以达到与该指令同样的效果。有关此主题的更多信息，请参见第 47 页 “bufsize 指令：设置归档程序缓冲区大小”。

指定归档请求的驱动器数：-drivemax、 -drivemin 和 -drives

默认情况下，归档程序只使用一个介质驱动器对归档集中的文件进行归档。如果归档集中的文件数量众多或比较大，则使用多个驱动器可以帮助您更有效地归档。另外，如果自动化库中驱动器的运行速度不同，使用这些指令可以使归档更为高效。

代码示例 3-23 和表 3-14 显示了可用于将归档请求拆分至多个磁带机，以及平衡磁带机传输速度差异的参数。

代码示例 3-23 -drivemax、 -drivemin 和 -drives 指令的格式

```
-drivemax max_size
-drivemin min_size
-drives number
```

表 3-14 -drivemax、 -drivemin 和 -drives 参数的参数

参数	含义
maxsize	一个驱动器中可归档的最大数据量。
minsize	一个驱动器中可归档的最小数据量。其默认值是 -archmax *target_size* 的值 （如果已指定）或是该介质类型的默认值。 如果指定了 -drivemin *minsize* 参数，则 Sun StorEdge SAM-FS 只在数据量足够大时才使用多个驱动器。作为一个指导原则，可以将 *minsize* 设置为足够大，以使传输时间远远大于卡盒更改时间 （载入、定位、卸载）。
number	用于对此归档集进行归档的驱动器数量。默认设置为 1。

系统会根据所指定的参数来检验归档请求，具体如下：

- 如果归档请求小于 *min_size*，则只使用一个驱动器来写入归档请求。
- 如果归档请求大于 *min_size*，则根据 *min_size* 检验归档请求，并调度适当数量的驱动器，但最多不超过指定的最大驱动器数。

■ 如果 *min_size* 为零，则尝试将归档请求中的文件归档在所指定最大数量的驱动器上。

当使用 -drives 参数时，仅在一次归档的数据量大于 *min_size* 时，才使用多个驱动器。实际并行使用的驱动器数是 *arch_req_total_size*/*min_size* 和 -drives 参数指定的驱动器数这两者当中的较小者。

如果希望将一个归档请求分割至各个驱动器，但是您又不希望将较小的归档请求也分割在全部驱动器上，则可以使用 -drivemin 和 -drives 参数。这适用于大型文件的操作。

要设置这些参数，用户需要考虑文件的创建速度、载入和卸载驱动器所需的时间以及驱动器的传输速率。

示例 1。假设您在五个驱动器上分割一个名为 big_files 的归档集。根据归档集的大小，表 3-15 中显示了可能的分割方法。

表 3-15 归档集分割示例

归档集大小	驱动器数
< 20 GB	1
≥ 20 GB 至 < 30 GB	2
≥ 30 GB 至 < 40 GB	3
≥ 40 GB 至 < 50 GB	4
≥ 50 GB	5

代码示例 3-24 显示了archiver.cmd 文件中用于将归档请求分割至多个驱动器的指令行。

代码示例 3-24 用于将归档请求分割至多个驱动器的指令

```
params
bigfiles.1 -drives 5 -drivemin 10G
endparams
```

示例 2。下行是在 archiver.cmd 文件中指定的：

```
huge_files.2 -drives 2
```

当归档集 huge_files.2 中的文件总大小等于或大于介质的 drivemin 的两倍时，使用两个驱动器来归档文件。

最大化卷上的空间：-fillvsns

默认情况下，归档程序在写入归档副本时，将使用分配给归档集的所有卷。而且归档程序在写入归档副本时，会选择一个空间足以容纳所有文件的卷。这将导致卷不会被完全占用。如果指定了 -fillvsns，归档程序则将归档请求拆分为一个较小的组。

设置归档缓冲区锁定：-lock

默认情况下，需要归档的文件先存储在缓冲区的内存中，然后再写入归档介质。如果已启用直接 I/O，则可以使用 -lock 参数锁定此缓冲区。此操作可以改善系统的性能，并且可以尝试不同的参数值，以达到最佳性能。

此参数的格式如下：

```
-lock
```

-lock 参数指明归档程序在创建归档副本时是否使用锁定的缓冲区。如果指定 -lock，归档程序将在 sam-arcopy(1M) 操作期间，在内存中的归档缓冲区上设置文件锁定。这样可避免对缓冲区进行分页，因而提高了系统性能。

仅在配有大量内存的大型系统上，才有必要指定 -lock 参数。如果内存不足，则可能会导致内存用尽。

只有已为需要归档的文件启用直接 I/O 时， -lock 参数才有效。默认情况下，不会指定 -lock 参数，并且文件系统会在所有直接 I/O 缓冲区上设置锁定 （包括用于归档的缓冲区）。有关启用直接 I/O 的更多信息，请参见 setfa(1) 手册页、 sam_setfa(3) 库例程手册页，或 mount_samfs(1M) 手册页中介绍的 -O forcedirectio 选项。

例如，可以按以下方式在 archiver.cmd 文件的命令行中指定该参数：

```
yourset.3 -lock
```

另外，还可通过指定 bufsize=*media buffer_size* [lock] 指令的 lock 参数，来指定全局性的、与该参数等效的参数。有关此主题的更多信息，请参见第 47 页 “bufsize 指令：设置归档程序缓冲区大小”。

创建脱机文件的归档副本：-offline_copy

为文件创建了一份归档副本后，此文件即成为可释放的备选文件。如果在创建所有归档副本前，释放文件并使其处于脱机状态，则归档程序会使用此参数来确定创建其他归档副本时所应使用的方法。在选择所使用的 *method* 时，要考虑 Sun StorEdge SAM-FS 系统的可用驱动器数以及可用的磁盘高速缓存空间。此参数的格式如下：

```
-offline_copy method
```

为 *method* 指定如下表所示的某个关键字。

表 3-16 -offline_copy 指令中 *method* 参数的值

method	含义
none	在将文件复制到归档卷中之前，根据需要登台每个文件。默认设置。
direct	不使用高速缓存，而是将文件从脱机卷直接复制到归档卷。此方法假定来源卷和目标卷是不同的卷，而且有两个可用驱动器。如果指定了此方法，则需将 stage_n_window 安装选项的值增大，使其大于 256 KB 的默认值。有关安装选项的更多信息，请参见 mount_samfs(1M) 手册页。
stageahead	在对一个文件进行归档的同时，登台另一个文件。如果指定了这种方法，系统会在将某文件写入目标位置的同时，登台下一个归档文件。
stageall	在归档前，将所有文件登台到磁盘高速缓存。这种方法仅使用一个驱动器，并假定磁盘高速缓存的空间足以容纳所有文件。

指定回收

回收进程可用于收回由过期归档映像所占用的归档卷中的空间。默认情况下，不进行回收。

要进行回收，可在 archiver.cmd 文件和 recycler.cmd 文件中设置指令。有关 archiver.cmd 文件中支持的回收指令的更多信息，请参见第 131 页 “回收”。

联合归档：-join path

如果指定了 -join path 参数，归档程序将使用联合归档。如果您希望将整个目录归档至一个卷，并且您知道归档文件实际只占用一个卷，则使用联合归档可以满足您的要求。否则，如果希望将目录保存在一起，请使用 -sort path 或 -rsort path 参数将文件连续地保存在一起。 -rsort 可执行逆向排序。

当归档程序将归档文件写入卷时，它会使用用户文件有效地压缩卷。将来，当从同一目录中访问文件时，会出现一定的延迟，因为登台进程需要对卷进行重新定位以读取下一个文件。为了缩短延迟，可以将来自同一目录路径的文件连续地归档至一个归档文件中。联合归档进程将取代空间效率运算法则，从而将来自同一目录的文件归档在一起。-join path 参数可以使这些文件在归档集副本中连续地归档在一起。

当无需更改文件内容但始终需要同时访问一组文件时，使用联合归档可以满足此类要求。例如在医院里，用户可能使用联合归档方法来访问医疗图片。与同一位患者关联的图片可以保存在同一个目录中，以便医生可以同时访问这些图片。如果根据静态图片的目录位置连续归档图片，则可以更为方便、快捷地访问它们。例如：

```
patient_images.1 -join path
```

注 – -join path 参数可将来自相同目录的数据文件写入至同一个归档文件。如果目录很多，但其中的文件较小且数量不多，归档程序会创建很多较小的归档文件。由于数据文件对每个归档文件的 tar(1) 头文件来说相对较小，因此这些较小的分散文件会降低系统的写入性能。这会降低高速磁带机的写入性能。

另外，由于 -join path 参数指定将来自同一目录中的所有文件都归档在单一卷中，因此有可能找不到合适的可用卷。在这种情况下，如果不为归档集分配更多的卷，这些文件将无法归档。另一种可能是一组要归档的文件太大，以至于永远不能归档在单个卷上。在这种情况下，这些文件永远无法归档。

对于大多数应用场合而言，如果不需要对 -join path 操作进行更为严格的限制，则 -sort path 或 -join path 参数是最佳选择。

另外，也可以按文件归档时限、大小或路径对归档集副本中的文件进行排序。 age 和 size 参数互相排斥，不可同时使用。代码示例 3-25 显示了如何使用 -sort 参数以及 age 或 size 参数，来对归档集进行排序。

代码示例 3-25 对归档集进行排序的指令

```
cardiac.1 -sort path
cardiac.2 -sort age
catscans.3 -sort size
```

第一行强制归档程序按路径名对归档请求进行排序。第二行强制归档程序根据文件的归档时限，按从旧到新的顺序，对名为 cardiac.2 的归档集副本进行排序。第三行强制归档程序根据文件的大小，按照从小到大的顺序，对名为 catscans 的归档集副本进行排序。如果希望逆向排序，可以用 -rsort 替代 -sort。

控制取消归档

取消归档是指删除文件或目录的归档条目的过程。默认情况下，文件永远不会被取消归档。归档程序根据上一次访问文件的时间来确定是否取消归档。所有经常访问的数据都可以存储在磁盘等快速介质中，而其他所有不经常访问的旧数据可以存储在磁带中。

示例 1。代码示例 3-26 显示了一个 `archiver.cmd` 文件。

代码示例 3-26 控制取消归档的指令

```
arset1 dir1
   1    10m    60d
   2    10m
   3    10m
vsns
arset1.1    mo       OPT00[0-9]
arset1.2    lt       DLTA0[0-9]
arset1.3    lt       DLTB0[0-9]
```

如果代码示例 3-26 所示的 `archiver.cmd` 文件所控制的文件常被访问，则它将始终保留在磁盘中，即使它的归档时间超过 60 天。只有在该文件未被访问的时间超过 60 天时才会删除副本 1 信息。

如果归档程序因文件未被访问的时间超过 60 天而删除其副本 1 信息，则当从副本 2 登台文件时，需要从磁带中读取文件。该文件恢复联机后，归档程序会在磁盘上为其创建新的副本 1，并且由此时开始计算为期 60 天的访问周期。如果该文件再次被访问，则 Sun StorEdge SAM-FS 归档程序会为其重新生成新的副本 1。

示例

假设某位患者在医院里进行为期四周的治疗。在此期间，该患者的所有文件位于快速介质中（副本 1=mo）。四周后，该患者出院。在该患者出院后的 60 天内，如果该患者的数据从未被访问过，则会取消归档 inode 中的副本 1 条目，而只保留副本 2 和副本 3 条目。此时，用户可以回收卷以便为新患者腾出空间，从而避免增加磁盘库。如果该患者六个月后到医院复查，则首先从磁带（副本 2）访问数据。现在，归档程序会在磁盘中自动创建一个新的副本 1，以确保在复查期间（可能持续数天或数周）可以从快速介质中访问数据。

控制归档文件的写入方式：`-tapenonstop`

默认情况下，归档程序会在归档文件之间写入一个磁带标记（EOF 标签）和多个磁带标记。当启动下一个归档文件时，驱动程序会返回到第一个磁带标记后面的位置，因而会造成性能降低。`-tapenonstop` 参数可以指示归档程序只写入初始的磁带标记。这

样，驱动程序只需返回到上一个磁带标记（而不是第一个磁带标记）后面的位置，因而提高了性能。另外，如果指定 -tapenonstop 参数，归档程序将在复制操作的最后输入归档信息。

有关 -tapenonstop 参数的更多信息，请参见 archiver.cmd(4) 手册页。

保留卷：-reserve

默认情况下，归档程序会将归档集副本写入由 archiver.cmd 文件中卷关联部分所述的正则表达式所指定的卷。但是，有时可能需要归档集卷只包含来自同一归档集的文件。保留卷进程可以满足这一数据存储要求。

注 – -reserve 参数用于保留专供一个归档集使用的卷。站点使用保留卷时，可能导致频繁的卡盒载入和卸载操作。

-reserve 参数可为归档集保留卷。在设置 -reserve 参数，并将一个卷分配给某个归档集副本后，该卷的标识不会分配至其他任何归档集副本，即使某个正则表达式与之相匹配。

选择供某个归档集使用的卷后，系统会为该卷分配一个保留名称。此保留名称是联结归档集和卷的唯一标识。

-reserve 参数的格式如下所示：

```
-reserve keyword
```

此处指定的 *keyword* 取决于您所使用的格式。可用的格式包括归档集格式、所有者格式和文件系统格式，具体如下：

- 归档集格式。此格式使用 set **关键字**的方式是：-reserve set
- 所有者格式。此格式使用以下某一**关键字**：dir、user 或 group。代码示例 3-27 显示了这些指令的格式。

代码示例 3-27 -reserve 参数的所有者格式

```
-reserve dir
-reserve user
-reserve group
```

代码示例 3-27 中所示的三种所有者格式相互排斥。也就是说，只能对归档集和副本使用这三种所有者格式的其中一种。

- 文件系统格式。此格式使用 fs **关键字**的方式是：-reserve fs

在 archiver.cmd 文件中，您可为一种、两种或全部三种格式指定 -reserve 参数。在归档集参数定义中，这三种格式可以组合使用。

例如，代码示例 3-28 中显示了 archiver.cmd 文件的一个片断。以 arset.1 开头的行根据归档集、组和文件系统创建了一个保留名。

代码示例 3-28 指定了保留卷的 archiver.cmd 文件

```
params
arset.1 -reserve set -reserve group -reserve fs
endparams
```

保留卷的信息存储在库目录中。库目录中的行包括介质类型、 VSN、保留信息和保留日期及时间。保留信息包括归档集、路径名和文件系统三个部分，它们之间由双斜杠 (//) 分隔。

双斜杠**并不**表示路径名；它们仅仅是分隔符，用于分隔保留名称的三个组成部分。如代码示例 3-29 所示，在库目录中，描述保留卷的行以字符 #R 开头。

代码示例 3-29 显示保留卷的库目录

```
      6  00071  00071 lt     0xe8fe   12 9971464 1352412 0x6a000000 131072 0x
#       -il-o-b-----  05/24/00 13:50:02  12/31/69 18:00:00  07/13/01 14:03:00
#R lt 00071 arset0.3// 2001/03/19 18:27:31
     10 ST0001 NO_BAR_CODE lt     0x2741    9 9968052 8537448 0x68000000 1310
#       -il-o-------  05/07/00 15:30:29  12/31/69 18:00:00  04/13/01 13:46:54
#R lt ST0001 hgm1.1// 2001/03/20 17:53:06
     16 SLOT22 NO_BAR_CODE lt     0x76ba    6 9972252 9972252 0x68000000 1310
#       -il-o-------  06/06/00 16:03:05  12/31/69 18:00:00  07/12/01 11:02:05
#R lt SLOT22 arset0.2// 2001/03/02 12:11:25
```

请注意，为符合页宽，代码示例 3-29 中的行已作了删减。

一个或多个保留信息字段可以保留空白，这视 archiver.cmd 文件中定义的选项而定。所列的日期和时间是指创建保留信息的时间。保留行添加到每一个保留卷 （即在归档期间保留用于某个归档集的卷）的文件。

可以使用 samu(1M) 实用程序的 v 来显示保留信息，也可使用代码示例 3-30 中所示的 archiver(1M) 或 dump_cat(1M) 命令格式来进行显示。

代码示例 3-30 用于显示保留信息的命令

```
archiver -lv
dump_cat -V catalog_name
```

以下说明了每一种用于显示参数、关键字和分配至卷的保留名示例的格式。

- 归档集格式。如表 3-17 所示， `set` 关键字用于激活保留名称中的归档集部分。

表 3-17 归档集格式示例

指令和关键字	保留名称示例
`-reserve set`	`users.1//`
	`Data.1//`

例如，在代码示例 3-31 的 `archiver.cmd` 文件片段中，以归档集名 `allsets` 开头的行将按归档集为所有归档集设置保留卷。

代码示例 3-31 按归档集保留卷

```
params
allsets -reserve set
endparams
```

- 所有者格式。 `dir`、 `user` 和 `group` 关键字用于激活保留名称中的所有者部分。 `dir`、 `user` 和 `group` 关键字相互排斥，不能同时使用。 `dir` 关键字使用紧跟在归档集定义的路径参数后面的目录路径部分。`user` 和 `group` 是自我说明式的关键字。表 3-18 给出了相关示例。

表 3-18 所有者格式示例

指令和关键字	保留名称示例
`-reserve dir`	`proj.1/p105/`
	`proj.1/p104/`
`-reserve user`	`users.1/user5/`
	`users.1/user4/`
`-reserve group`	`data.1/engineering/`

注 – `-reserve` 参数用于保留专供一个归档集使用的卷。如果目录很多，但其中的文件较小且数量不多，则会造成很多较小的归档文件写入至每一个保留卷。由于数据文件对每个归档文件的 `tar`(1) 头文件来说相对较小，因此这些较小的分散文件会降低系统的性能。

- 文件系统格式。fs 关键字用于激活保留名称中的文件系统部分。表 3-19 给出了相关示例。

表 3-19 文件系统格式示例

指令和关键字	保留名称示例
-reserve fs	proj.1/p103/samfs1
	proj.1/p104/samfs1

第 97 页 “示例 4”显示了使用保留卷的完整归档示例。

归档程序将卷保留信息记录在库目录文件中。重新标记某个卷后，由于卷中的归档数据实际上已被清除，因此归档程序会自动取消保留该卷。

此外，您还可以使用 reserve(1M) 和 unreserve(1M) 命令分别保留及取消保留卷。有关这些命令的更多信息，请参见 reserve(1M) 和 unreserve(1M) 手册页。

设置归档优先级：-priority

Sun StorEdge SAM-FS 文件系统为文件的归档操作提供了一个可配置的优先级系统。每一个文件均分配有优先级。文件的优先级是通过文件的属性以及优先级乘数（可在 archiver.cmd 文件中为每一个归档集进行设置）计算出来的。文件属性包括联机 / 脱机、归档时限、创建副本的数量和大小。

默认情况下，归档程序不会对归档请求中的文件进行排序，并且所有属性乘数均为零。这将使归档程序按先发现先归档的顺序对文件进行归档。有关优先级的更多信息，请参见 archiver(1M) 和 archiver.cmd(4) 手册页。

可以通过设置优先级和排序方法，来控制文件的归档顺序。以下是设置优先级的示例：

- 选择 priority 排序方法，以按优先级的顺序对归档请求中的归档文件进行归档。
- 更改 archive_loaded 优先级，以减少介质载入次数。
- 更改 offline 优先级，以使联机文件的归档时间早于脱机文件。
- 更改 copy# 优先级，以按副本顺序创建归档副本。

表 3-20 中列出了归档优先级。

表 3-20 归档优先级

归档优先级	定义
`-priority age` *value*	归档时限属性乘数
`-priority archive_immediate` *value*	立即归档属性乘数
`-priority archive_overflow` *value*	多个归档卷属性乘数
`-priority archive_loaded` *value*	已载入归档卷属性乘数
`-priority copy1` *value*	副本 1 属性乘数
`-priority copy2` *value*	副本 2 属性乘数
`-priority copy3` *value*	副本 3 属性乘数
`-priority copy4` *value*	副本 4 属性乘数
`-priority copies` *value*	创建副本属性乘数
`-priority offline` *value*	文件脱机属性乘数
`-priority queuewait` *value*	队列等待属性乘数
`-priority rearchive` *value*	重新归档属性乘数
`-priority reqrelease` *value*	请求释放属性乘数
`-priority size` *value*	文件大小属性乘数
`-priority stage_loaded` *value*	已载入登台卷属性乘数
`-priority stage_overflow` *value*	多个登台卷属性乘数

其中的 *value*，用于指定一个以下范围之内的浮点数：

```
-3.400000000E+38 ≤ value ≤ 3.402823466E+38
```

调度归档：-startage、-startcount 和 -startsize

归档程序在扫描文件系统时，将识别要归档的文件。它将那些被识别为归档对象的文件放置在名为一个**归档请求**的列表中。在文件系统扫描结束后，系统会调度对归档请求中的文件进行归档。-startage、-startcount 和 -startsize 归档集参数控制归档的工作负荷，并确保及时地对文件进行归档。表 3-21 给出了这些参数的格式。

表 3-21 -startage、-startcount 和 -startsize 指令格式

指令	含义
-startage *time*	指定自扫描过程中标记第一个文件并将其列入归档请求，至开始归档所花费的**时间**。其中的 *time*，用于按第 63 页 “设置归档时限”中所述的格式指定时间。如果此参数未设置，可使用 interval 指令对其进行设置。
-startcount *count*	指定归档请求中可包含的文件数。当归档请求中文件数量达到 *count* 时，开始归档。应为 *count* 指定一个整数。默认情况下，不设置 *count*。
-startsize *size*	指定归档请求中要归档的所有文件的最小总大小（以字节为单元）。随着归档量逐渐增加，当文件的总大小达到 *size* 时，即开始归档。默认情况下，不设置 *size*。

examine=*method* 指令和 interval=*time* 指令，可与 -startage、-startcount 和 -startsize 指令交互。-startage、-startcount 和 -startsize 指令可优化归档时间，或者归档完成量。这些值可取代 examine=*method* 参数（如果已指定）。有关 examine 指令的更多信息，请参见第 49 页 “examine 指令：控制归档扫描”。有关 interval 指令的更多信息，请参见第 50 页 “interval 指令：指定归档时间间隔”。

可以在 archiver.cmd 文件中为每个归档副本，指定 -startage、-startcount 和 -startsize 指令。如果在其中指定了多个指令，则在满足第一个条件时，即开始归档操作。如果 -startage、-startcount 和 -startsize 这三者都未指定，则根据 examine=*method* 指令调度归档请求，具体如下：

- 如果 examine=noscan，则当第一个文件进入归档请求中后，归档程序将根据 interval=*time* 指令的参数值来调度归档请求。这属于连续归档法。默认情况下，examine=noscan。
- 如果 examine=scan | scaninodes | scandirs，则归档程序在文件系统扫描完成之后，开始调度归档请求。

archiver.cmd(4) 手册页中给出了如何使用这些指令的示例。

VSN 关联指令

archiver.cmd 文件的 VSN 关联部分用于为归档集分配卷。此部分以 vsns 指令开始，以 endvsns 指令结束。

还可以使用 File System Manager 软件配置 VSN 关联。有关更多信息，请参见 File System Manager 联机帮助。

可以通过以下格式的指令将一组卷分配给归档集：

```
archive_set_name.copy_num media_type vsn_expr ... [ -pool vsn_pool_name ... ]
```

表 3-22 VSN 关联指令的参数

参数	含义
archive_set_name	您的站点为归档集定义的名称。它必须是归档集分配指令中的第一个字段。归档集名通常暗示属于该归档集的文件的特征。指定归档集名时，只能使用 26 个英文字母、数字 (0-9) 和下划线字符 (_)，而不得使用其他特殊字符或空格。归档集名的第一个字符必须是字母。
copy_num	数字后面是一个或多个用于指定该副本归档特征的参数。归档副本指令以数字开头。该数字（1、 2、 3 或 4）是副本份数。
media_type	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
vsn_expr	正则表达式。请参见 regexp(5) 手册页。
-pool *vsn_pool_name*	已命名的 VSN 组。

一个关联至少需要三个字段：*archive_set_name* 和 *copy_number* 以及 *media_type*，而且至少还需要有一个卷。 *archive_set_name* 和 *copy_number* 通过一个句点 (.) 相连。

以下几个示例以不同的方法指定了相同的 VSN。

示例 1。代码示例 3-32 显示了两行 VSN 规范。

代码示例 3-32 VSN 规范 — 示例 1

```
vsns
set.1  lt  VSN001 VSN002 VSN003 VSN004 VSN005
set.1  lt  VSN006 VSN007 VSN008 VSN009 VSN010
endvsns
```

示例 2。代码示例 3-33 显示了 VSN 规范，它使用反斜杠字符 (\) 将上一行续接至下一行。

代码示例 3-33 VSN 规范 — 示例 2

```
vsns
set.1  lt  VSN001 VSN002 VSN003 VSN004 VSN005 \
 VSN006 VSN007 VSN008 VSN009 VSN010
endvsns
```

示例 3。代码示例 3-34 使用简单的正则表达式指定 VSN。

代码示例 3-34 VSN 规范 — 示例 3

```
vsns
set.1 lt VSN0[1-9] VSN10
endvsns
```

卷由一个或多个 *vsn_expression* 关键字指定，这些关键字即为 `regexp`(5) 手册页中所述的正则表达式。请注意，这些正则表达式所遵守的约定与通配符不同。除正则表达式之外，还可以指定从哪些 VSN 池中选择卷。 VSN 池通过 `-pool` *vsn_pool_name* 指令和 VSN 关联来表示。

归档程序需要使用卷对归档集中的文件进行归档时，将检查自动化库和手动载入的驱动器中选定介质类型的每一个卷，以确定它们是否符合 VSN 表达式的要求。归档程序将选择第一个符合表达式要求且包含足够空间以进行归档副本操作的卷。例如：

- 以下指令指定，将属于归档集 `ex_set` 的第 1 个副本中的文件复制到介质类型 `mo`，且将使用名称为 `optic20` 至 `optic39` 的 20 个卷中的任意卷：

```
ex_set.1 mo optic[2-3][0-9]
```

- 以下指令将属于归档集 `ex_set` 的第 2 个副本的文件复制到介质类型 `lt`，所使用的卷为以 TAPE 开头的任意卷：

```
ex_set.2  lt  ^TAPE
```

如果将 Sun StorEdge SAM-FS 环境配置为按归档集进行回收，则不要将 VSN 分配给多个归档集。

注 – 在设置 `archiver.cmd` 文件时，请确保将卷分配给用于归档元数据的归档集。每一个文件系统均有一个与其自身名称相同的归档集。有关保存元数据的更多信息，请参见 `samfsdump`(1M) 手册页或 《Sun StorEdge SAM-FS 故障排除指南》。

VSN 池指令

archiver.cmd 文件中，VSN 池部分以 vsnpools 指令开始，以 endvsnpools 指令结束或至 archiver.cmd 文件的结尾处结束。该部分命名了一组卷。

还可使用 File System Manager 软件配置 VSN 池。有关更多信息，请参见 File System Manager 联机帮助。

VSN **池**是一个已命名的卷组。 VSN 池非常适用于为某个归档集定义可用的卷，因为它可以提供一个有益于为归档集分配和保留卷的缓冲区。

可以使用 VSN 池定义多个单独的卷组，以供公司中的各部门、组中的各个用户、某些类数据和其他为方便而划分的组使用。系统会为 VSN 池分配名称、介质类型和一组卷。**暂用池**是指当 VSN 关联中的特定卷或另一个 VSN 池消耗殆尽时，系统临时使用的一组卷。有关 VSN 关联的更多信息，请参见第 78 页 “VSN 关联指令”。

如果某个卷被保留用于归档集，则该卷将不能再供它最初所属的 VSN 池使用。因此，已命名的 VSN 池中的卷数量随卷的使用情况而变化。可以按以下格式输入 archiver(1M) 命令来查看 VSN 池：

```
# archiver -lv | more
```

VSN 池定义至少需要三个字段：池名、介质类型以及至少一个 VSN，这些字段以空格分隔。语法格式如下：

```
vsn_pool_name  media_type  vsn_expression
```

表 3-23 VSN 池指令的参数

参数	含义
vsn_pool_name	指定 VSN 池
media_type	由 2 个字符表示的介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
vsn_expression	正则表达式。可以指定一个或多个 *vsn_expression* 参数。请参见 regcmp(3G) 手册页。

以下示例中使用了四个 VSN 池：users_pool、 data_pool、 proj_pool 和 scratch_pool。如果三个池中某一个池的卷消耗殆尽，则归档程序选择暂用池 VSN。代码示例 3-35 显示了一个使用四个 VSN 池的 archiver.cmd 文件。

代码示例 3-35 VSN 池示例

```
vsnpools
users_pool    mo ^MO[0-9][0-9]
data_pool     mo ^DA.*
```

代码示例 **3-35** VSN 池示例 （续）

```
scratch_pool mo ^SC[5-9][0-9]
proj_pool    mo ^PR.*
endvsnpools
vsns
users.1      mo    -pool users_pool   -pool scratch_pool
data.1       mo    -pool data_pool    -pool scratch_pool
proj.1       mo    -pool proj_pool    -pool scratch_pool
endvsns
```

关于磁盘归档

归档是指将文件从联机磁盘复制到归档介质的过程。通常，归档程序将归档副本写入至自动化库中磁光盘或磁带卡盒上的卷中；但对于磁盘归档，文件系统中的联机磁盘将用作归档介质。

应用磁盘归档功能，不仅可以将 Sun StorEdge SAM-FS 文件系统中的文件归档至同一计算机主机系统中的另一个文件系统，而且还可将源文件归档至其他 Sun Solaris 系统中的另一文件系统。如果在两个主机系统上应用磁盘归档功能，则其中一个系统为客户机，另一个为服务器。**客户机系统**是指作为源文件宿主的系统。**服务器系统**是指作为归档副本宿主的目标系统。

归档文件写入至的文件系统可以是任何一种 UNIX 文件系统，但它不一定必须是 Sun StorEdge SAM-FS 文件系统。如果磁盘归档副本写入至另一台主机，则该主机上必须至少安装了一个与 Sun StorEdge SAM-FS 兼容的文件系统。

归档程序对待归档至磁盘卷的文件的方式与对待归档至自动化库中卷的文件相同。仍然可以创建一份、两份、三份或四份归档副本。如果创建多份归档副本，则可以将其中一个归档副本写入至磁盘卷，而将其他归档副本写入至可移除介质卷。另外，如果您通常将文件归档至 Sun StorEdge SAM-FS 文件系统中的磁盘卷，则归档程序将根据该文件系统的 `archiver.cmd` 文件中的规则对归档文件副本自身进行归档。

以下概述了归档至联机磁盘与归档至可移除介质的相似点和不同点：

- 与写入至磁光盘或磁带的归档副本不同，写入至磁盘的归档副本不会记录在目录中。另外，磁盘卷中的归档文件不会出现在历史记录中。
- 如果您要将文件归档至可移除介质卷，则在安装文件系统后，无需更改 `archiver.cmd` 文件中的任何默认值便可开始归档。但是，如果您要将文件归档至磁盘卷，则在安装文件系统之前必须编辑 `archiver.cmd` 文件以定义磁盘归档集。
- 磁盘归档不能使用 `mcf`(4) 文件中的条目。您需要在 `archiver.cmd` 文件中指定磁盘归档集，并且需要在 `/etc/opt/SUNWsamfs/diskvols.conf` 文件中定义磁盘卷。后者是一个附加的配置文件，如果您仅将文件归档至可移除介质卷，则无需使用此文件。

diskvols.conf 文件必须在源文件所在的系统中进行创建。根据归档副本写入位置的不同，此文件还可能包含以下信息：

- 如果归档副本写入至同一主机系统中的文件系统，则 diskvols.conf 文件将定义 VSN 及其路径。
- 如果归档副本写入至另一个 Sun Solaris 系统，则 diskvols.conf 文件将包含该服务器系统的主机名。在此情况下，该服务器系统中也必须有一个 diskvols.conf 文件，以定义那些有权向该服务器系统写入数据的客户机。如果希望创建这种客户机 / 服务器关系，在启动第 83 页 “启用磁盘归档” 过程前，应确保充当服务器的主机上至少安装了一个 Sun StorEdge SAM-FS 文件系统。

配置原则

虽然磁盘归档卷的位置不受限制，但是，建议您不要将该卷设置在源文件所在的磁盘上。如果将客户机系统中的归档副本写入至服务器系统中的磁盘卷，则更加理想。我们建议您创建多个归档副本，并将它们写入至不同类型的归档介质。例如，您可以将第 1 个副本写入至磁盘卷，将第 2 个副本写入至磁带，而将第 3 个副本写入至磁光盘。

如果您将文件归档至服务器上的文件系统，则归档文件自身还会被归档至与目标服务器连接的库中的可移除介质卡盒。

磁盘归档指令

当归档至联机磁盘时，归档程序可识别 archiver.cmd 指令中的大多数。它所识别的指令是那些对归档集进行定义以及配置回收过程的指令。而它所忽略的指令是那些在磁盘归档环境中毫无意义的指令，因为这些指令专用于处理可移除介质卡盒。系统将着重识别以下用于磁盘归档集的指令：

- 第 64 页 “归档集副本参数” 中，除以下指令之外的所有回收指令：
  - -fillvsns
  - -ovflmin *min_size*
  - -reserve *method*
  - -tapenonstop
- 第 138 页 “步骤 2：编辑 archiver.cmd 文件” 中，除以下指令之外的所有指令：
  - -recycle_dataquantity *size*
  - -recycle_vsncount *count*

- vsns 和 endvsns 指令，以及 vsnpools 和 endvsnpools 指令。VSN 关联部分支持磁盘卷，并可使用 dk 介质类型定义磁盘卷。卷可以由一个或多个 VSN 表达式关键字 （正则表达式）指定。除 VSN 之外，还可以指定从哪些 VSN 池中选择卷。VSN 池是一个已命名的卷组。例如：

代码示例 3-36 vsns 和 vsnpools 指令的示例

```
vsnpools
data_pool dk disk0[0-5]
endvsnpools

vsns
arset0.1 dk disk10 disk1[2-5]
arset1.1 dk -pool data_pool
endvsns
```

- clients 和 endclients 指令。如果执行磁盘归档，以将原文件从客户机主机归档至服务器主机，则必须对服务器主机上的 diskvols.conf 文件进行配置。服务器系统上的 diskvols.conf 文件中必须包含客户机系统的名称。这些指令的格式如下所示：

代码示例 3-37 clients 和 endclients 指令的格式

```
clients
client_system1
client_system2
...
endclients
```

其中的 *client_system*，用于指定包含源文件的客户机系统的主机名。

- -recycle_minobs percent 回收程序指令。此选项用于设置回收程序对磁盘归档进行重新归档过程中的阈值。当磁盘上一个已归档的 tar 文件中无效文件的百分比达到此阈值时，回收程序会将该归档中的有效文件移动到新的 tar 文件中。一旦所有的有效文件移动完成，会将初始的 tar 文件标记为要从磁盘归档中删除的备选文件。回收可移除介质时，可忽略此选项。阈值默认设置为 50%。

有关磁盘归档指令的更多信息，请参见 archiver.cmd(4) 手册页。

▼ 启用磁盘归档

磁盘归档功能可随时启用。本节介绍的过程假定归档的前期准备工作已经就绪，现在只需在环境中添加磁盘归档功能。如果您是在初始安装过程中启用磁盘归档，请参见《Sun StorEdge SAM-FS 安装和升级指南》以了解相关信息。这时不要使用本过程，因为如果是在安装时添加磁盘归档功能，则本过程中有些步骤是不必要的。

注 – 在 4U4 之前的软件版本中，需要使用 archiver.cmd 文件中 `params` 部分的 `-disk_archive` 参数，才能启用磁盘归档。 4U4 软件版本已不再使用此参数，因此为了使归档功能在 4U4 版本中正常工作，必须对使用早期软件版本创建的 `archiver.cmd` 文件进行编辑。有关详细信息，请参见 `archiver.cmd`(4) 手册页。

1. **请确保在要写入磁盘归档副本的主机上，至少安装了一个 Sun StorEdge QFS 文件系统。**
2. **以超级用户的身份登录到要归档的文件所在的主机系统。**
3. **按照** 《Sun StorEdge SAM-FS 安装和升级指南》**中介绍的过程启用磁盘归档功能。**

   Sun StorEdge SAM-FS 的初始安装过程中包含了一个名为**启用磁盘归档**的步骤。该步骤可以拆分为两个过程。
4. **在包含要归档的文件的主机上，使用** `samd`**(1M)** `config` **命令传播配置文件的更改，然后重新启动系统。**

   例如：

```
# samd config
```

5. **以超级用户的身份登录到将要写入归档副本的主机系统。（可选）**

   仅当要将文件归档至另一台主机的磁盘上时，才需要执行此步骤。
6. **在将要写入归档副本的主机上，使用** `samd`**(1M)** `config` **命令传播配置文件的更改，并重新启动系统。（可选）**

   仅当要将文件归档至另一台主机的磁盘上时，才需要执行此步骤。

   例如：

```
# samd config
```

磁盘归档示例

示例 1

代码示例 3-38 显示了客户机系统 `pluto` 上的 `diskvols.conf` 文件。

代码示例 3-38 `pluto` 上的 `diskvols.conf` 文件

```
# This is file /etc/opt/SUNWsamfs/diskvols.conf on pluto
# VSN Name      [Host Name:]Path
#
```

代码示例 3-38 pluto 上的 diskvols.conf 文件 （续）

```
disk01                        /sam_arch1
disk02                        /sam_arch2/proj_1
disk03                   mars:/sam_arch3/proj_3
disk04                      /sam_arch4/proj_4
```

在上面的 diskvols.conf 文件中，名为 disk01、disk02 和 disk04 的 VSN 将写入至初始源文件所在的主机系统。 VSN disk03 将写入至服务器系统 mars 中的 VSN。

代码示例 3-39 显示了服务器系统 mars 上的 diskvols.conf 文件。

代码示例 3-39 mars 上的 diskvols.conf 文件

```
# This is file /etc/opt/SUNWsamfs/diskvols.conf on mars
#
clients
pluto
endclients
```

代码示例 3-40 显示了 pluto 上的 archiver.cmd 文件片断。

代码示例 3-40 pluto 上的 archiver.cmd 文件

```
vsns
arset1.2 dk disk01
arset2.2 dk disk02 disk04
arset3.2 dk disk03
endvsns
```

示例 2

在本示例中，文件 /sam1/testdir0/filea 位于归档集 arset0.1 中，归档程序将 /sam1/testdir0/filea 的内容复制到名为 /sam_arch1 的目标路径中。代码示例 3-41 显示了这个 diskvols.conf 文件。

代码示例 3-41 diskvols.conf 文件

```
# This is file /etc/opt/SUNWsamfs/diskvols.conf
#
# VSN Name    [Host Name:]Path
#
disk01                    /sam_arch1
disk02                    /sam_arch12/proj_1
```

代码示例 3-42 显示了 archiver.cmd 文件中与磁盘归档有关的指令行：

代码示例 3-42 archiver.cmd 文件中与磁盘归档有关的指令

```
.
vsns
arset0.1 dk disk01
endvsns
.
```

以下是对已归档至磁盘的 filea 文件运行 sls(1) 命令时的输出。在代码示例 3-43 中，请注意以下内容：

- dk 是磁盘归档介质的介质类型
- disk01 是 VSN
- f192 是磁盘归档 tar(1) 文件的路径

代码示例 3-43 sls(1M) 的输出

```
# sls -D /sam1/testdir0/filea
/sam1/testdir0/filea:
  mode: -rw-r-----  links:   1  owner: root      group: other
  length:    797904  admin id:      0  inode:     3134.49
  archdone;
  copy 1: ---- Dec 16 14:03        c0.1354 dk disk01 f192
  access:      Dec 19 10:29  modification: Dec 16 13:56
  changed:     Dec 16 13:56  attributes:   Dec 19 10:29
  creation:    Dec 16 13:56  residence:    Dec 19 10:32
```

示例 3

在本示例中，文件 /sam2/my_proj/fileb 位于客户机主机 snickers 的归档集 arset0.1 中，归档程序将该文件的内容复制到服务器主机 mars 的目标路径 /sam_arch1 中。

代码示例 3-44 显示了 snickers 上的 diskvols.conf 文件。

代码示例 3-44 snickers 上的 diskvols.conf 文件

```
# This is file /etc/opt/SUNWsamfs/diskvols.conf on snickers
#
# VSN Name    [Host Name:]Path
#
disk01         mars:/sam_arch1
```

代码示例 3-45 显示了 mars 上的 diskvols.conf 文件。

代码示例 3-45 mars 上的 diskvols.conf 文件

```
# This is file /etc/opt/SUNWsamfs/diskvols.conf on mars
#
clients
snickers
endclients
```

代码示例 3-46 显示了 archiver.cmd 文件中与本示例有关的指令。

代码示例 3-46 archiver.cmd 文件中与磁盘归档有关的指令

```
.
vsns
arset0.1 dk disk01
endvsns
.
```

设计归档操作

归档程序使用 archiver.cmd 文件进行自动存储管理操作。编写此文件之前，请复习一些可以改善 Sun StorEdge SAM-FS 文件系统和归档程序性能的通用原则，这是非常有益的，可确保您以最安全的方式存储数据。

每一个站点在计算应用、数据存储硬件和软件方面都是各不相同的。以下建议是 Sun Microsystems 根据多年的经验总结出来的。为您的站点编写 archiver.cmd 文件时，请确保通过考虑以下方面来反映站点的数据存储要求。

- 保存归档日志。归档日志可为数据恢复提供非常重要的信息，即便在 Sun StorEdge SAM-FS 软件无法使用时也是如此。我们建议您将这些日志保存在安全的地方，以防因发生灾难性故障而造成 Sun StorEdge SAM-FS 软件无法使用。
- 使用正则表达式指定卷。允许系统将文件存储在多个不同的卷上。通过正则表达式指定的卷范围可以使系统连续不断地运行。如果您为归档集副本指定特定的卷名，则会造成数据很快充满卷。因此，您不得不频繁地更换介质，从而导致不应有的工作流程问题出现。
- 根据文件的创建和修改频率以及是否需要保存所有修改副本，来设置您的归档时间间隔。请注意，归档时间间隔是指对文件系统执行扫描操作的时间间隔。如果将归档时间间隔设置得太短，则会使归档程序几乎不间断地执行扫描。
- 考虑您要使用的文件系统数量。与单个 Sun StorEdge SAM-FS 文件系统相比，多个 Sun StorEdge SAM-FS 文件系统通常可以提高归档程序的性能。归档程序为每一个文件系统运行单独的进程。多个文件系统的扫描时间要比单个文件系统少得多。

- 使用目录结构组织 Sun StorEdge SAM-FS 文件系统中的文件。出于性能考虑， Sun Microsystems 建议您不要将 10,000 个以上的文件放入同一个目录中。
- 请始终制作两份文件副本，并将它们存储至不同的卷上。如果将数据存储至同一种介质类型，则在介质出现物理问题时，您会面临数据丢失的风险。如果条件允许，请制作多份归档副本。
- 务必定期运行 `samfsdump`(1M) 来转储您的元数据。元数据 （包括目录结构和文件名等）存储在与文件系统同名的归档集中。在出现故障时，您可以使用此类信息来恢复文件系统。如果您不想执行此操作，可以通过将此归档集分配至不存在的 VSN，来防止归档此类数据。有关保存元数据的更多信息，请参见 《Sun StorEdge SAM-FS 故障排除指南》或 《Sun StorEdge SAM-FS 安装和升级指南》。

预备队列

归档程序和登台程序进程均可请求载入及卸载介质。如果请求的数量超过可用于介质载入的驱动器数量，则多余的请求会发送至预备队列。

预备队列中的归档和登台请求是指那些无法立即满足的请求。默认情况下，系统按先进先出 (First In First Out, FIFO) 的顺序执行预备请求。

您可以为各个预备请求分配不同的优先级。可以通过在预备命令文件中输入指令来改写 FIFO 默认值，此预备命令文件的位置为 `/etc/opt/SUNWsamfs/preview.cmd`。有关此文件以及设置归档和登台操作优先级的更多信息，请参见第 124 页 “排列预备请求的优先顺序”。

归档程序示例

表 3-24 显示了本节所有示例使用的目录结构。

表 3-24 目录结构示例

最高层目录	第 1 级子目录	第 2 级子目录	第 3 级子目录
/sam	/projs	/proj_1	/katie
/sam	/projs	/proj_1	/sara
/sam	/projs	/proj_1	/wendy
/sam	/projs	/proj_2	/joe
/sam	/projs	/proj_2	/katie
/sam	/users	/bob	
/sam	/users	/joe	

表 3-24 目录结构示例 （续）

最高层目录	第 1 级子目录	第 2 级子目录	第 3 级子目录
/sam	/users	/katie	
/sam	/users	/sara	
/sam	/users	/wendy	
/sam	/data		
/sam	/tmp		

示例 1

本示例介绍在无 archiver.cmd 文件可用时归档程序的操作。在本示例中， Sun StorEdge SAM-FS 环境包括一个文件系统、一个配有两个驱动器的光盘自动化库和六个卡盒。

代码示例 3-47 显示了 archiver(1M) -lv 命令的输出。它表明归档程序选择的默认介质类型为 mo。只有 mo 介质可用。

代码示例 3-47 archiver(1M) -lv 输出示例之一

```
# archiver -lv
Notify file: /etc/opt/SUNWsamfs/scripts/archiver.sh

Archive media:
media:lt archmax: 512.0M Volume overflow not selected
media:mo archmax:   4.8M Volume overflow not selected
```

代码示例 3-48 的输出表明归档程序使用了两个驱动器。它列出了 12 个卷及其存储容量和可用空间。

注 – archiver(1M) -lv 命令只能显示具有可用空间的 VSN。

代码示例 3-48 archiver(1M) -lv 输出示例之二

```
Archive libraries:
Device:hp30 drives_available:2 archive_drives:2
  Catalog:
  mo.optic00          capacity:  1.2G space: 939.7M  -il-o-------
  mo.optic01          capacity:  1.2G space: 934.2M  -il-o-------
  mo.optic02          capacity:  1.2G space: 781.7M  -il-o-------
  mo.optic03          capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic10          capacity:  1.2G space:  85.5M  -il-o-------
  mo.optic11          capacity:  1.2G space:  0      -il-o-------
```

代码示例 3-48 archiver(1M) -lv 输出示例之二 （续）

```
  mo.optic12          capacity:  1.2G space: 618.9k  -il-o-------
  mo.optic13          capacity:  1.2G space: 981.3M  -il-o-------
  mo.optic20          capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic21          capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic22          capacity:  1.2G space: 244.9k  -il-o-------
  mo.optic23          capacity:  1.2G space:   1.1G  -il-o-------
```

代码示例 3-49 的输出表明元数据和数据文件均包括在归档集 samfs 中。当文件的归档时间达到默认的四分钟 （240 秒）时，归档程序将开始创建文件的副本。

代码示例 3-49 archiver(1M) -lv 输出示例之三

```
Archive file selections:
Filesystem samfs  Logfile:
samfs Metadata
    copy:1  arch_age:240
samfs1  path:.
    copy:1  arch_age:240
```

代码示例 3-50 的输出表明归档集中的文件按指定的顺序归档至卷中。

代码示例 3-50 archiver(1M) -lv 输出示例之四

```
Archive sets:
allsets
samfs.1
 media: mo (by default)
 Volumes:
   optic00
   optic01
   optic02
   optic03
   optic10
   optic12
   optic13
   optic20
   optic21
   optic22
   optic23
 Total space available:   8.1G
```

示例 2

本示例说明如何将数据文件和元数据划分至两个不同的归档集。除了第 85 页 “示例 2”中所述的光盘自动化库之外，文件系统环境还配备了手动安装的 DLT 磁带机。较大的文件归档至磁带，而较小的文件归档至光盘卡盒。

代码示例 3-51 显示了archiver.cmd 文件的内容。

代码示例 3-51 archiver(1M) -lv 输出示例之一：显示 archiver.cmd 文件

```
# archiver -lv -c example2.cmd
Reading archiver command file "example2.cmd"
1: # Example 2 archiver command file
2: # Simple selections based on size
3:
4: logfile = /var/opt/SUNWsamfs/archiver/log
5: interval = 5m
6:
7: # File selections.
8: big . -minsize 500k
9: all .
10:    1 30s
11:
12: vsns
13: samfs.1 mo .*0[0-2]       # Metadata to optic00 - optic02
14: all.1 mo .*0[3-9] .*[1-2][0-9]  # All others for files
15: big.1 lt .*
16: endvsns
```

代码示例 3-52 显示了要使用的介质和驱动器，但没有显示 DLT 及其默认设置。

代码示例 3-52 archiver(1M) -lv 输出示例之二：显示介质和驱动器

```
Notify file: /etc/opt/SUNWsamfs/scripts/archiver.sh
Archive media:
media:lt archmax: 512.0M Volume overflow not selected
media:mo archmax:   4.8M Volume overflow not selected
Archive libraries:
Device:hp30 drives_available:0 archive_drives:0
  Catalog:
  mo.optic00        capacity:  1.2G space: 939.7M  -il-o-------
  mo.optic01        capacity:  1.2G space: 934.2M  -il-o-------
  mo.optic02        capacity:  1.2G space: 781.7M  -il-o-------
  mo.optic03        capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic04        capacity:  1.2G space: 983.2M  -il-o-------
  mo.optic10        capacity:  1.2G space:  85.5M  -il-o-------
  mo.optic11        capacity:  1.2G space:   0     -il-o-------
  mo.optic12        capacity:  1.2G space: 618.9k  -il-o-------
  mo.optic13        capacity:  1.2G space: 981.3M  -il-o-------
```

代码示例 3-52 archiver(1M) -lv 输出示例之二：显示介质和驱动器 （续）

```
  mo.optic20         capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic21         capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic22         capacity:  1.2G space: 244.9k  -il-o-------
  mo.optic23         capacity:  1.2G space:   1.1G  -il-o-------
Device:lt40 drives_available:0 archive_drives:0
  Catalog:
  lt.TAPE01          capacity:  9.5G space:  8.5G  -il-o-------
  lt.TAPE02          capacity:  9.5G space:  6.2G  -il-o-------
  lt.TAPE03          capacity:  9.5G space:  3.6G  -il-o-------
  lt.TAPE04          capacity:  9.5G space:  8.5G  -il-o-------
  lt.TAPE05          capacity:  9.5G space:  8.5G  -il-o-------
  lt.TAPE06          capacity:  9.5G space:  7.4G  -il-o-------
```

注 – archiver(1M) -lv 命令只能显示具有可用空间的 VSN。

代码示例 3-53 显示了文件系统的组织结构。大于 512000 字节 (500 KB) 的文件在四分钟后归档；其他所有文件在 30 秒后归档。

代码示例 3-53 archiver(1M) -lv 输出示例之三：显示文件系统组织结构

```
Archive file selections:
Filesystem samfs  Logfile: /var/opt/SUNWsamfs/archiver/log
samfs Metadata
    copy:1  arch_age:240
big  path:. minsize:502.0k
    copy:1  arch_age:240
all  path:.
    copy:1  arch_age:30
```

代码示例 3-54 的输出显示了归档集在可移除介质中的分割情况。

代码示例 3-54 archiver(1M) -lv 输出示例之四：显示归档集和可移除介质

```
Archive sets:
allsets
all.1
 media: mo
Volumes:
   optic03
   optic04
   optic10
   optic12
   optic13
   optic20
   optic21
   optic22
```

代码示例 3-54 archiver(1M) -lv 输出示例之四：显示归档集和可移除介质 （续）

```
    optic23
 Total space available:   6.3G
big.1
 media: lt
Volumes:
    TAPE01
    TAPE02
    TAPE03
    TAPE04
    TAPE05
    TAPE06
 Total space available:  42.8G
samfs.1
 media: mo
Volumes:
    optic00
    optic01
    optic02
 Total space available:   2.6G
```

示例 3

在本示例中，用户文件和项目数据文件归档至不同的介质。 data 目录中的文件按大小分别存储至光盘介质和磁带介质。分配至组 ID pict 的文件将被分配至另一个卷组。归档程序不对目录 tmp 和 users/bob 中的文件进行归档。归档程序每隔 15 分钟进行一次归档，并且保存归档记录。

代码示例 3-55 显示了这个示例。

代码示例 3-55 archiver(1M) -lv -c 命令的输出

```
# archiver -lv -c example3.cmd
Reading archiver command file "example3.cmd"
1: # Example 3 archiver command file
2: # Segregation of users and data
3:
4: interval = 30s
5: logfile = /var/opt/SUNWsamfs/archiver/log
6:
7: no_archive tmp
8:
9: fs = samfs
10: no_archive users/bob
11: prod_big data -minsize 50k
12:    1 1m 30d
13:    2 3m
```

代码示例 **3-55** archiver(1M) -lv -c 命令的输出 （续）

```
14: prod data
15:    1 1m
16: proj_1 projs/proj_1
17:    1 1m
18:    2 1m
19: joe . -user joe
20:    1 1m
21:    2 1m
22: pict . -group pict
23:    1 1m
24:    2 1m
25:
26: params
27: prod_big.1 -drives 2
28: prod_big.2 -drives 2
29: endparams
30:
31: vsns
32: samfs.1 mo optic0[0-1]$
33: joe.1 mo optic01$
34: pict.1 mo optic02$
35: pict.2 mo optic03$
36: proj_1.1 mo optic1[0-1]$
37: proj_1.2 mo optic1[2-3]$
38: prod.1 mo optic2.$
39: joe.2 lt 0[1-2]$
40: prod_big.1 lt 0[3-4]$
41: prod_big.2 lt 0[5-6]$
42: endvsns

Notify file: /etc/opt/SUNWsamfs/scripts/archiver.sh

Archive media:
media:lt archmax: 512.0M Volume overflow not selected
media:mo archmax:   4.8M Volume overflow not selected

Archive libraries:
Device:hp30 drives_available:0 archive_drives:0
  Catalog:
  mo.optic00        capacity:  1.2G space: 939.7M  -il-o-------
  mo.optic01        capacity:  1.2G space: 934.2M  -il-o-------
  mo.optic02        capacity:  1.2G space: 781.7M  -il-o-------
  mo.optic03        capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic04        capacity:  1.2G space: 983.2M  -il-o-------
  mo.optic10        capacity:  1.2G space:  85.5M  -il-o-------
  mo.optic11        capacity:  1.2G space:  0      -il-o-------
  mo.optic12        capacity:  1.2G space: 618.9k  -il-o-------
```

代码示例 **3-55** archiver(1M) -lv -c 命令的输出（续）

```
  mo.optic13         capacity:  1.2G space: 981.3M  -il-o-------
  mo.optic20         capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic21         capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic22         capacity:  1.2G space: 244.9k  -il-o-------
  mo.optic23         capacity:  1.2G space:   1.1G  -il-o-------

Device:lt40 drives_available:0 archive_drives:0
  Catalog:
  lt.TAPE01          capacity:  9.5G space:  8.5G  -il-o-------
  lt.TAPE02          capacity:  9.5G space:  6.2G  -il-o-------
  lt.TAPE03          capacity:  9.5G space:  3.6G  -il-o-------
  lt.TAPE04          capacity:  9.5G space:  8.5G  -il-o-------
  lt.TAPE05          capacity:  9.5G space:  8.5G  -il-o-------
  lt.TAPE06          capacity:  9.5G space:  7.4G  -il-o-------

Archive file selections:
Filesystem samfs  Logfile: /var/opt/SUNWsamfs/archiver/log
samfs Metadata
    copy:1  arch_age:240
no_archive Noarchive path:users/bob
prod_big  path:data minsize:50.2k
    copy:1  arch_age:60 unarch_age:2592000
    copy:2  arch_age:180
prod  path:data
    copy:1  arch_age:60
proj_1  path:projs/proj_1
    copy:1  arch_age:60
    copy:2  arch_age:60
joe  path:. uid:10006
    copy:1  arch_age:60
    copy:2  arch_age:60
pict  path:. gid:8005
    copy:1  arch_age:60
    copy:2  arch_age:60
no_archive Noarchive path:tmp
samfs  path:.
    copy:1  arch_age:240

Archive sets:
allsets

joe.1
 media: mo
 Volumes:
   optic01
 Total space available: 934.2M
```

代码示例 **3-55** archiver(1M) -lv -c 命令的输出（续）

```
joe.2
 media: lt
 Volumes:
   TAPE01
   TAPE02
 Total space available:  14.7G

pict.1
 media: mo
 Volumes:
   optic02
 Total space available: 781.7M

pict.2
 media: mo
 Volumes:
   optic03
 Total space available:   1.1G

prod.1
 media: mo
 Volumes:
   optic20
   optic21
   optic22
   optic23
 Total space available:   3.3G

prod_big.1
 media: lt drives:2
 Volumes:
   TAPE03
   TAPE04
 Total space available:  12.1G

prod_big.2
 media: lt drives:2
 Volumes:
   TAPE05
   TAPE06
 Total space available:  16.0G

proj_1.1
 media: mo
 Volumes:
   optic10
 Total space available:  85.5M
```

代码示例 3-55 archiver(1M) -lv -c 命令的输出 （续）

```
proj_1.2
 media: mo
 Volumes:
   optic12
   optic13
 Total space available: 981.9M

samfs.1
 media: mo
 Volumes:
   optic00
   optic01
 Total space available:   1.8G
```

示例 4

在本示例中，用户文件和项目数据文件归档至光盘介质。请注意，代码示例 3-56 没有使用表 3-24 中所示的目录结构。

本示例定义了四个 VSN 池；其中三个池分别用于用户、数据和项目，另一个是暂用池。当 proj_pool 中的介质用尽后，它使用 scratch_pool 来保留卷。本示例介绍如何依据归档集、所有者和文件系统等各个组成部分，来为每一个归档集保留卷。归档程序每隔 10 分钟进行一次归档，并且保存归档日志。

代码示例 3-56 显示了 archiver.cmd 文件及归档程序的输出。

代码示例 3-56 archiver.cmd 文件及归档程序的输出

```
Reading archiver command file "example4.cmd"
1: # Example 4 archiver command file
2: # Using 4 VSN pools
3:
4: interval = 30s
5: logfile = /var/opt/SUNWsamfs/archiver/log
6:
7: fs = samfs
8: users users
9:     1 10m
10:
11: data data
12:    1 10m
13:
14: proj projects
15:    1 10m
16:
```

代码示例 3-56 archiver.cmd 文件及归档程序的输出 （续）

```
Reading archiver command file "example4.cmd"
17: params
18: users.1 -reserve user
19: data.1 -reserve group
20: proj.1 -reserve dir -reserve fs
21: endparams
22:
23: vsnpools
24: users_pool mo optic0[1-3]$
25: data_pool mo optic1[0-1]$
26: proj_pool mo optic1[2-3]$
27: scratch_pool mo optic2.$
28: endvsnpools
29:
30: vsn
31: samfs.1 mo optic00
32: users.1 mo -pool users_pool -pool scratch_pool
33: data.1 mo -pool data_pool -pool scratch_pool
34: proj.1 mo -pool proj_pool -pool scratch_pool
35: endvsns

Notify file: /etc/opt/SUNWsamfs/scripts/archiver.sh

Archive media:
media:mo archmax:   4.8M Volume overflow not selected

Archive libraries:
Device:hp30 drives_available:0 archive_drives:0
  Catalog:
  mo.optic00        capacity:  1.2G space: 939.7M  -il-o-------
  mo.optic01        capacity:  1.2G space: 934.2M  -il-o-------
  mo.optic02        capacity:  1.2G space: 781.7M  -il-o-------
  mo.optic03        capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic04        capacity:  1.2G space: 983.2M  -il-o-------
  mo.optic10        capacity:  1.2G space:  85.5M  -il-o-------
  mo.optic11        capacity:  1.2G space:   0     -il-o-------
  mo.optic12        capacity:  1.2G space: 618.9k  -il-o-------
  mo.optic13        capacity:  1.2G space: 981.3M  -il-o-------
  mo.optic20        capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic21        capacity:  1.2G space:   1.1G  -il-o-------
  mo.optic22        capacity:  1.2G space: 244.9k  -il-o-------
  mo.optic23        capacity:  1.2G space:   1.1G  -il-o-------

Archive file selections:
Filesystem samfs  Logfile: /var/opt/SUNWsamfs/archiver/log
samfs Metadata
    copy:1  arch_age:240
```

代码示例 3-56 archiver.cmd 文件及归档程序的输出 （续）

```
Reading archiver command file "example4.cmd"
users  path:users
    copy:1  arch_age:600
data  path:data
    copy:1  arch_age:600
proj  path:projects
    copy:1  arch_age:600
samfs  path:.
    copy:1  arch_age:240


VSN pools:
data_pool media: mo Volumes:
   optic10
 Total space available:  85.5M

proj_pool media: mo Volumes:
   optic12
   optic13
 Total space available: 981.9M

scratch_pool media: mo Volumes:
   optic20
   optic21
   optic22
   optic23
 Total space available:   3.3G

users_pool media: mo Volumes:
   optic01
   optic02
   optic03
 Total space available:   2.7G

Archive sets:
allsets

data.1
  reserve:/group/
 media: mo
 Volumes:
   optic10
   optic20
   optic21
   optic22
   optic23
 Total space available:   3.4G
```

代码示例 3-56 archiver.cmd 文件及归档程序的输出 （续）

```
Reading archiver command file "example4.cmd"

proj.1
  reserve:/dir/fs
 media: mo
 Volumes:
   optic12
   optic13
   optic20
   optic21
   optic22
   optic23
 Total space available:   4.2G

samfs.1
 media: mo
 Volumes:
   optic00
 Total space available: 939.7M

users.1
  reserve:/user/
 media: mo
 Volumes:
   optic01
   optic02
   optic03
   optic20
   optic21
   optic22
   optic23
 Total space available:   6.0G
```

第 4 章

释放

释放是指释放程序通过识别已归档的文件并释放这些文件在磁盘高速缓存中的副本，从而使磁盘高速缓存空间可再利用的过程。这可以为其他从归档介质中创建或登台的文件腾出空间。释放程序只释放已归档的文件。释放文件后，系统会从磁盘高速缓存中清除该文件的所有数据。

当高速缓存的占用量达到站点指定的磁盘阈值时， Sun StorEdge SAM-FS 软件会自动调用释放进程。或者， `release`(1) 命令可以立即释放文件占用的磁盘空间或为文件设置释放参数。有关释放进程的更多信息，请参见 `sam-releaser`(1M) 手册页。

释放程序所包含的功能允许您指定哪些文件在归档之后立即释放、哪些文件不释放以及哪些文件可以部分释放。由于某些诸如 `filemgr`(1) 之类的实用程序只读取文件的起始部分，因此部分释放功能特别有用。采用部分释放功能时，文件的一部分保留在磁盘高速缓存中，而文件的剩余部分会被释放。读取仍保留在磁盘高速缓存中的文件的第一部分时，并不会导致系统从归档介质中将文件的剩余部分重新登台至磁盘高速缓存。本章即介绍上述功能及其他多种功能。

本章包括下列主题：

归档过程概述

当文件系统的利用率超过所配置的上限时，文件系统管理软件将启动释放程序。首先，释放程序读取 `releaser.cmd` 文件并收集用于控制释放进程的指令。其次，它扫描文件系统并收集每一个文件的有关信息。最后，在扫描整个文件系统后，释放程序开始按优先级顺序释放文件。

只要文件系统的利用率高于所配置的下限，释放程序就会继续释放文件。通常，释放程序会释放足够的空间，以使文件系统的利用率低于所配置的下限。如果释放程序没有发现任何需要释放的文件，将会退出。以后，当更多的文件可以释放时，释放程序即会运行。当高于上限时，文件系统会每隔一分钟启动一次释放程序。

上、下限可以使用 `high=`*percent* 和 `low=`*percent* 文件系统安装选项来设置。有关安装选项的更多信息，请参见 `mount_samfs`(1M) 手册页。

操作原理

文件系统可能包含成千上万个文件。由于只需释放几个大文件便有可能使文件系统的利用率降至下限，因此没有必要记录所有文件的释放优先级。但是，释放程序又必须检查每一个文件的优先级，否则就不能释放最恰当的备选文件。释放程序通过只确定前 10,000 个备选文件来解决这一问题。

确定前 10,000 个备选文件之后，如果随后的备选文件优先级不高于前 10,000 个备选文件的最低优先级，则释放程序会忽略随后的备选文件。

确定前 10,000 个备选文件的优先级之后，释放程序会选择释放具有最高优先级的文件。每释放一个文件，释放程序就会进行一次检查，确定文件系统的高速缓存利用率是否低于下限。如果是，则释放程序将停止释放文件。如果否，则释放程序将继续按优先级的顺序释放文件。

如果在释放全部 10,000 个备选文件之后，文件系统的利用率仍高于下限，则释放程序将重新确定 10,000 个新备选文件。

如果找不到任何合适的备选文件，则释放程序会退出。例如，在文件没有归档副本时，就会出现这种情况。如果出现这种情况，则 Sun StorEdge SAM-FS 软件将在退出释放程序的一分钟后再次启动释放程序。

定义

本节介绍本章中使用的术语。

时限

时限是指从发生指定事件开始到现在所经历的时间。文件的 inode 可以跟踪记录释放程序所使用的下列时间：

- 驻留更改时间
- 数据修改时间
- 数据访问时间

您可以使用带有 `-D` 选项的 `sls`(1) 命令来查看这些时间。每一种时间都有对应的时限。例如，如果当前时间是上午 10 点 15 分，则在上午 10 点 10 分所修改的文件的数据修改时限为 5 分钟。有关 `sls`(1) 命令的更多信息，请参见 `sls`(1) 手册页。

备选文件

备选文件即符合释放条件的文件。文件不能成为备选文件的原因包括：

- 该文件已脱机。
- 该文件尚未归档。
- `archiver.cmd` 命令文件为该文件指定了 `-norelease` 属性，并且尚未为该文件创建完所需的副本份数。
- 该文件标记为 “已损坏”。
- 该文件不是普通文件，而是目录文件、块文件、特殊字符文件或管道文件。
- 归档程序正在登台该文件以创建另一副本。登台之后，该文件便成为适合释放的文件。
- 该文件的时限为负数。这种情况通常会发生在未正确设置时钟的 NFS 客户机上。
- 文件被标记为不释放。这可使用 `release`(1) `-n` 命令来指定。
- 该文件在过去登台的时间小于最短驻留时间设置。有关更多信息，请参见第 111 页 “指定最短驻留时间：min_residence_age”。
- 已使用 `release`(1) 命令的 `-p` 选项将该文件标记为 “部分释放”，并且释放程序已部分释放该文件。
- 该文件太小。

优先级

优先级是一个表示备选文件级别的数值，该数值取决于用户提供的应用于该备选文件数值属性的权重。总优先级是以下两类优先级之和：时限优先级和大小优先级。

释放程序首先释放具有较大优先级数值的备选文件，然后释放具有较小优先级数值的备选文件。

权重

权重是一个数值，用于使优先级的计算倾向于包括您感兴趣的文件属性，并排除不感兴趣的文件属性。例如，如果将文件的大小权重设置为零，则在计算优先级时，不会考虑文件的大小属性。权重是介于 0.0 和 1.0 之间的浮点值。

部分释放

可以**部分释放**文件，方法是指定在磁盘高速缓存中保留文件的起始部分，而释放文件的剩余部分。在使用 `filemgr`(1) 等实用程序读取文件的起始部分时，部分释放功能非常有用。

关于部分释放和部分登台

释放和登台是两个互为补充的进程。文件在归档后，就可以从联机磁盘高速缓存中完全释放，站点也可以指定只在磁盘高速缓存中保留文件的起始部分 （即**存根**），而释放文件的其余部分。这种部分释放文件的功能可以使系统在不登台文件的情况下，立即访问文件存根中的数据。

系统管理员可以在安装文件系统时，指定部分释放的默认大小和保持联机的存根的最大值。系统管理员可以通过 `mount`(1M) 命令或 File System Manager 软件设置这些值。有关更多信息，请参见 File System Manager 联机帮助。`mount`(1M) 命令具有以下选项：

- 指定 `-o partial=`*n* 选项，来设置保持联机的文件存根的默认大小 (*n*)。`-o partial=`*n* 的设置值必须小于或等于 `-o maxpartial=`*n* 的设置值。最小设置值为 `-o partial=8` KB。默认设置为 `-o partial=16` KB。
- 指定 `-o maxpartial=`*n* 选项，来设置保持联机的文件存根大小的最大值 (*n*)。要限制可以保持联机的文件存根的大小，可使用 `-o maxpartial=`*n* 选项，并将其值指定为可以保持联机的最大存根大小。要禁用部分释放功能，可指定 `-o maxpartial=0`。

用户可以通过在 release(1) 命令中指定 -p 选项或在 sam_release(3) 库例程中指定 p 选项，来指定文件的默认存根大小。要为不同类型的文件或不同的应用程序指定不同大小的文件存根，用户可以在 release(1) 命令中指定 -s 选项，或在 sam_release(3) 库例程中指定 s 选项。-s 和 s 的值必须小于安装文件系统时使用 mount(1M) 命令的 -o maxpartial 选项指定的值。

系统管理员可以使用另一个安装选项，即 -o partial_stage=*n*，来确定在登台文件的剩余部分之前，应读取多少部分释放存根。也就是说，当读取的存根量大于 -o partial_stage=*n* 后，即开始登台文件。

默认情况下，系统将 -o partial_stage=*n* 选项设置为等于部分释放存根的大小。虽然用户可以配置该值，但应注意该值对文件登台的影响，具体如下：

- 如果将 -o partial_stage=*n* 选项设置为等于部分释放存根的大小，则系统的默认操作是直到应用程序到达部分释放存根的末尾时才允许登台文件。等待到达存根的末尾会推迟应用程序对文件剩余部分的访问。
- 如果将 -o partial_stage=*n* 选项设置为小于部分释放存根的值，则会出现以下情况。在应用程序超过 -o partial_stage=*n* 选项设置的阈值后，系统将登台文件的剩余部分。这可以加快应用程序对文件数据剩余部分的访问。

示例。假定您设置了下列选项：

- -o partial_stage=16（即 16 KB）
- -o partial=2097152（即 2 GB）
- -o maxpartial=2097152（即 2 GB）

所使用的程序为 filemgr(1)，它首先读取文件的前 8 KB。此时，系统不会登台文件。有一个视频点播程序读取同一个文件，并且当它读完文件的前 16 KB 时，系统开始登台文件。在安装并定位归档磁带后，该应用程序会继续读取 2 GB 的磁盘数据。当视频点播程序读完 2 GB 的文件数据后，它会在登台活动之后立即进行读取。由于在该应用程序读取部分文件数据时已经安装并定位了磁带，因此它不必等待。

有多个命令行选项可影响文件是否可以标记为部分释放。某些选项可由系统管理员启用，而另一些选项可由个别用户启用。以下几节介绍了不同类型的用户可以设置的释放特征。

系统管理员选项概述

系统管理员可以在安装文件系统时，更改部分释放的最大值和默认值。表 4-1 列出了可影响部分释放的 mount(1M) 选项。有关 mount(1) 命令的更多信息，请参见 mount_samfs(1M) 手册页。

表 4-1 影响部分释放的安装选项

mount(1M) 选项	作用
-o maxpartial=*n*	指定在文件标记为部分释放时，可以在联机磁盘高速缓存中保留的最大空间 (KB)。最大值为 2,097,152 KB，即 2 GB。最小值为 0，即不允许部分释放任何文件。 如果指定 -o maxpartial=0，则会禁用部分释放功能。已释放的文件会被完全释放，并且磁盘高速缓存中不会保留文件的任何部分。一旦安装文件系统，用户便不能再改写此选项指定的值。 默认情况下， *n* 变量设置为 16。此设置使得用户可以将文件标记为部分释放，且这个文件最多可在磁盘上保留 16 KB 的数据。
-o partial=*n*	当用户使用 release(1) 命令的 -p 选项将文件标记为部分释放时，该选项用于设置要在磁盘高速缓存中保留的默认空间 (KB)。*n* 变量的值至少为 8，但它也可以等于 -o maxpartial=*n* 选项的设置值。 由于某些应用程序不必访问整个文件便可完成其工作，因此该选项可用于确保应用程序能够从文件的起始部分获得所需的信息。同时，使用此选项还可阻止系统登台不必要的文件。 默认值为 -o partial=16。
-o partial_stage=*n*	指定在访问部分释放的文件时，从归档介质登台整个文件之前应读取 *n* 字节的文件数据。此选项的设置值通常小于 -o partial 的设置值。其中的 *n*，用于指定介于 0 和 -o maxpartial 参数之间的一个整数值。默认情况下，系统将它设置为 16 或是由 -o partial 选项指定的任何值。
-o stage_n_window=*n*	将一次可以登台的数据量指定为 *n*。其中的 *n*，用于指定一个介于 64 至 2,048,000 之间的整数。默认值为 256 KB。此选项仅适用于已设置 stage -n 属性的文件。

用户选项概述

系统管理员可以设置文件在释放后，可保留在磁盘高速缓存中的文件存根大小的最大值和默认值。此外，系统管理员还可以确定是否为特定的文件系统启用部分释放功能。

不过，通过使用 release(1) 命令和 sam_release(3) 库例程，用户可以设置其他释放属性以及指定要标记为部分释放的文件。表 4-2 中列出了可以指定部分释放属性的命令和库选项。有关 release(1) 命令的更多信息，请参见 release(1) 手册页。有关 sam_release(3) 库例程的更多信息，请参见 sam_release(3) 手册页。

表 4-2 用户释放选项

选项	作用
release(1) 命令和 -p 选项 或 sam_release(3) 库例程和 p 选项	-p 和 p 选项用于将指定的文件标记为部分释放。如果使用这些选项，则文件在释放后可以保留在联机磁盘高速缓存中的数据量，取决于在安装该文件所在的文件系统时为 -o partial=*n* 选项设置的值。这些选项不能用于指定保持联机的字节数。
release(1) 命令和 -s partial_size 选项 或 sam_release(3) 库例程和 s 选项	-s 和 s 选项用于将指定的文件标记为部分释放，并指定要在联机磁盘高速缓存中保留的文件数据量。-s 或 s 选项的变量用于指定要保持联机的数据量 (KB)。 用户指定的保持联机的文件数据量不能大于安装文件系统时指定的 -o maxpartial=*n* 值。如果用户指定的值大于文件系统的值，则系统会使用文件系统的值，而忽略用户指定的规范。

关于 releaser.cmd 文件

/etc/opt/SUNWsamfs/releaser.cmd 文件由指定站点特定释放操作的指令行组成。 releaser.cmd 文件可以包含用于设置释放优先级的指令、用于指定日志文件的指令以及用于执行其他操作的指令。

以下几节介绍了 releaser.cmd 文件中的指令：

- 第 108 页 “指定与时段和大小相关的释放优先级指令”
- 第 110 页 “指定用于单个文件系统的指令：fs”
- 第 111 页 “指定调试指令：no_release 和 display_all_candidates”
- 第 111 页 “指定最短驻留时间：min_residence_age”
- 第 111 页 “指定日志文件：logfile”
- 第 113 页 “阻止释放重新归档的文件：rearch_no_release”
- 第 113 页 “调整释放程序备选文件列表的大小：list_size”

有关这些指令的更多信息，请参见 `releaser.cmd`(4) 手册页。可使用 File System Manager 软件配置某些全局释放指令。有关更多信息，请参见 File System Manager 联机帮助。

指定与时段和大小相关的释放优先级指令

释放程序根据 `releaser.cmd` 文件中定义的指令所确定的优先级顺序来释放文件系统中的文件。在释放文件时，它既考虑文件的时限，又考虑文件的大小。默认情况下，站点首先释放最大且最旧的文件，而将最小且最新的文件保留在磁盘中。以下部分介绍了释放程序在确定文件系统中文件的释放优先级时如何考虑文件的时段和大小。

有关这些释放指令的更多信息，请参见 `releaser.cmd`(4) 手册页。

文件时限

释放程序在确定与时限相关的文件释放优先级要素时，将考虑下列可能的时限：

- 从最后一次访问文件到现在的时限
- 从最后一次修改文件到现在的时限
- 从文件在磁盘高速缓存中的驻留状态发生更改到现在的时限

在某些情况下，您可能希望文件的访问时限优先于修改时限。而在其他情况下，可能优先考虑由最近访问时间、修改时间和驻留状态更改时间得出的简单时限。

默认情况下，文件时段是指下列三个文件时段中最小的一个：

- 文件访问时限
- 文件修改时限
- 文件驻留时限

您可以使用指令指定在计算文件的释放优先级时要使用的加权时限优先级。

代码示例 4-1 显示了时限优先级指令的格式。

代码示例 4-1 时限优先级指令的格式

```
weight_age = float
weight_age_access = float
weight_age_modification = float
weight_age_residence = float
```

- `weight_age` 指令指定要为其分配加权因子的文件默认时限 （文件访问时限、修改时限和驻留时限三者中的较小者）。其中的 *float*，用于指定处于以下范围内的浮点数：$0.0 \leq float \leq 1.0$。默认情况下， *float* = `1.0`。

该指令不能与 weight_age_residence、 weight_age_modify 或 weight_age_access 指令结合使用。

- weight_age_residence、weight_age_modify 和 weight_age_access 指令指定由这些可能时限中的一个、二个或三个组合起来而确定的文件时限。其中的 *float*，用于指定处于以下范围内的浮点数：0.0 ≤ *float* ≤ 1.0。默认情况下， *float* = 1.0。

  这些指令不能与 weight_age 指令结合使用。

  如果使用 weight_age_residence、 weight_age_modify 和 weight_age_access 指令，则依据这三个时限的组合来计算文件的与时限相关的优先级。首先，收集每一个文件的可能时段数据。其次，将文件时限数据与 releaser.cmd 文件中指定的加权因子相乘。最后，将时限数据乘以每一个加权因子的结果相加 （如代码示例 4-2 所示），从而计算出与文件时限相关的优先级：

代码示例 4-2 优先级计算

```
  file access age * weight_age_access
+ file modification age * weight_age_modification
+ file residency age * weight_age_residence
_____________________________________________________
= age_related_priority
```

示例。代码示例 4-3 显示了 releaser.cmd 文件中的指令行，它们指定在计算文件的释放优先级时，只考虑文件的驻留时限 （而忽略修改时限和访问时限）。

代码示例 4-3 releaser.cmd 文件片断

```
weight_age_residence = 1.0
weight_age_modify = 0.0
weight_age_access = 0.0
```

计算文件的与时段相关的优先级之后，将其与文件的与大小相关的优先级相乘。下一节介绍了如何计算与文件大小相关的优先级。

文件大小

释放程序在确定与大小相关的文件释放优先级要素时将考虑文件的大小。文件的大小（在 4 KB 块中）乘以您为 weight_size 指令指定的权重，即可得出与大小相关的文件释放优先级要素。

weight_size 指令的格式如下所示：

```
weight_size = float
```

其中的 *float*，用于指定处于以下范围内的浮点数：0.0 ≤ *float* ≤ 1.0。默认情况下，*float* = 1.0。

示例。代码示例 4-4 显示了 `releaser.cmd` 文件，其中指定在计算文件的释放优先级时，忽略 `samfs1` 和 `samfs2` 文件系统中所有文件的大小。

代码示例 4-4 `releaser.cmd` 文件

```
# releaser.cmd file
logfile = /var/adm/default.releaser.log
weight_size = 0.0
#
fs = samfs1
weight_age = 1.0
logfile = /var/adm/samfs1.releaser.log
#
fs = samfs2
weight_age_modify = 0.3
weight_age_access = 0.03
weight_age_residence = 1.0
logfile = /var/adm/samfs2.releaser.log
```

指定用于单个文件系统的指令：`fs`

可以在 `releaser.cmd` 文件中使用 `fs = family_set_name` 指令来指定，`fs =` 之后的指令仅适用于指定的文件系统。此指令的格式如下：

```
fs = family_set_name
```

其中的 *family_set_name*，用于指定 `mcf` 文件中的系列集名称。

位于第一个 `fs =` 指令前面的指令是应用于所有文件的全局指令。`fs =` 指令后面的指令可以取代全局指令。本章所述的指令既可用作全局指令，也可用作专用于一个文件系统的指令。

`releaser.cmd`(4) 手册页中提供了 `fs =` 指令的示例。

指定调试指令：no_release 和 display_all_candidates

在调节或调试释放程序时， no_release 和 display_all_candidates 指令非常有用。这几个指令如下：

- no_release 指令可以防止文件从联机磁盘高速缓存中被删除。可以使用此指令来检查 releaser.cmd 文件中的指令，这不会真正地释放文件。此指令的格式如下：

```
no_release
```

- display_all_candidates 指令可以将所有释放备选文件的名称写入至日志文件。此指令的格式如下：

```
display_all_candidates
```

由于释放程序可将释放备选文件的名称写入至日志文件，而不是真正从文件系统中释放这些文件，因此这些指令对调试很有帮助。

指定最短驻留时间：min_residence_age

min_residence_age 指令可用于指定文件在成为释放备选文件之前，必须在文件系统中驻留的最短时间。此指令的格式如下：

```
min_residence_age = time
```

其中的 *time*，用于指定驻留时间，以秒为单位。默认时间为 600 秒，即 10 分钟。 *time* 设置的最大或最小值并没有具体规定。

指定日志文件：logfile

如果在 releaser.cmd 文件中指定了 logfile 指令，则释放程序会将其自身活动添加至指定的文件名，或新创建的文件名 （如果指定的文件名不存在）。此指令的格式如下：

```
logfile = filename
```

其中的 *filename*，用于指定日志文件的名称。

代码示例 4-5 是一个日志文件范例 （请注意，为适应页宽已将某些行换行）。

代码示例 4-5 释放程序日志文件示例

```
Releaser begins at Wed Apr 28 17:29:06 1999
inode pathname          /sam1/.inodes
low-water mark          24%
weight_size             1
weight_age              1
fs equipment ordinal    1
family-set name         samfs1
started by sam-amld?    yes
release files?          yes
display_all_candidates? no
---before scan---
blocks_now_free:    3481504
lwm_blocks:         3729362
---scanning---
10501 (R: Wed Apr 21 18:47:50 CDT 1999) 10001 min, 500 blks /sam1/testdir0/filevp
10500 (R: Wed Apr 21 18:48:10 CDT 1999) 10000 min, 500 blks /sam1/testdir0/filewq
...
---after scan---
blocks_now_free:      3730736
lwm_blocks:           3729362
archnodrop: 0
already_offline: 0
bad_inode_number: 0
damaged: 0
extension_inode: 0
negative_age: 0
nodrop: 1
not_regular: 9
number_in_list: 675
released_files: 202
too_new_residence_time: 0
too_small: 2
total_candidates: 675
total_inodes: 1376
wrong_inode_number: 0
zero_arch_status: 689
zero_inode_number: 0
zero_mode: 0
CPU time: 2 seconds.
Elapsed time: 10 seconds.
Releaser ends at Wed Apr 28 17:29:16 1999
```

`releaser`(1M) 手册页介绍了日志文件中包含的信息。随着释放程序的每一次运行，日志文件的大小会不断增加，因此，请确保减小日志文件的大小，或删除 `logfile` 关键字。

代码示例 4-6 显示了 ---after scan--- 行下，各统计项之间的数学关系：

代码示例 4-6 代码示例 4-5 中 ---after scan--- 行之后各统计项的数学关系

```
total_inodes = wrong_inode_number +
zero_inode_number +
zero_mode +
not_regular +
extension_inode +
zero_arch_status +
already_offline +
damaged +
nodrop +
archnodrop +
too_new_residence_time +
too_small +
negative_age +
total_candidates
   released_files = total_candidates
```

阻止释放重新归档的文件：rearch_no_release

默认情况下，系统会释放标记为重新归档的文件。如果在 releaser.cmd(4) 文件中指定 rearch_no_release 指令，则释放程序不会释放标记为重新归档的文件。此指令的格式如下：

```
rearch_no_release
```

调整释放程序备选文件列表的大小：list_size

可使用 list_size 指令来指定释放程序备选文件的数量。如果您注意到释放程序为了达到下限而释放所需数量的文件之前，在对多个文件系统执行扫描，那么可能需要考虑增大这个值以使之大于默认值 10,000。在包含许多小文件的文件系统中就可能存在这样的情况。您可以从释放程序日志文件中，获取有关释放程序的活动信息。此指令的格式如下：

```
list_size = number
```

其中的 number，用于指定范围如下的一个整数：10 ≤ *number* ≤ 2,147,483,648。

archiver.cmd 文件在释放过程中的角色

archiver.cmd 文件中的大多数指令会影响归档过程，但归档集分配指令可以使您指定应用于归档集中所有文件的释放属性。

归档集分配指令的格式如下：

```
archive_set_name  path  [search_criteria ...] [file_attributes]
```

表 4-3 显示了与释放有关的**文件属性**。

表 4-3 归档集分配**文件属性**

指令	作用
-release a	指定释放程序应在创建第一个归档副本后，释放归档集中的文件。如果您需要为每一个文件创建多份归档副本，请勿使用此选项。在此情况下，系统会登台第一份副本以创建第二份副本。
-release d	重置为默认设置。
-release n	指定不释放归档集中的文件。
-release p	指定释放程序应在归档文件后，部分释放归档集中的文件。

有关这些及其他 archiver.cmd 指令的更多信息，请参见第 31 页 “归档”。

规划释放程序的操作

您有必要为您的站点确定高速缓存中文件的特征。如果只需登台几千字节的小文件，载入磁带是不经济的，因此您可能更希望系统将小文件保存在高速缓存中。代码示例 4-7 显示了 releaser.cmd 文件中用于优先释放最大文件的指令。

代码示例 4-7 优先释放最大文件的指令

```
weight_size = 1.0
weight_age = 0.0
```

此外，您可能希望将最近修改过的文件保留在高速缓存中，因为您不久可能会重新修改它们。这可以避免因登台文件以进行修改而产生的开销。在这种情况下，可使用第二组时限权重。代码示例 4-8 显示了 `releaser.cmd` 文件中用于对文件进行加权的指令，以保证严格遵守从最早修改的文件到最新修改的文件的释放顺序。

代码示例 4-8 优先释放最早修改的文件的指令

```
weight_size = 0.0
weight_age_access = 0.0
weight_age_modify = 1.0
weight_age_residence = 0.0
```

不过，大多数情况并不采用这种简单直接的方式，如下面的示例所述。

示例 1。假定您希望首先释放最大的文件。文件系统中包括成百上千个大小相同的小文件和数个大文件。小文件的大小之和可能超过一个最大文件的大小。最终，释放程序将释放所有大文件。如果指定 `weight_age = 0.0`，则释放程序基本上按任意顺序释放小文件，因为它们的大小相同，并且具有相同的释放优先级。

此时，您可以设置 `weight_age = 0.01` 来进行进一步的加权。这样，如果两个文件大小相等，释放程序会首先释放较旧的文件。

示例 2。本示例介绍了一种指定如何首先释放最大文件的更好方法。

设置 `weight_size = 1.0` 和 `weight_age = 0.01`。

这两个指令通过优先选择较小的、不经常访问的文件（而不是较大的、经常访问的文件）作为释放备选文件，因而它们违背了首先释放最大文件的原则。通过使 `weight_age` 的值更小于 `weight_size` 的值，您可以将这一影响降低到您所需的程度。例如，根据上面的设置，已登台 100 分钟的 4 KB 文件与刚刚登台的 8 KB 文件具有相同的释放优先级。

此时，释放程序会选择释放其中任何一个文件。如果它选择 4 KB 的文件，则违背了首先释放最大文件的意向。为了降低这一影响，请将 `weight_age` 设置为更小的值，例如 0.001。如果 4 KB 文件登台的时间为 1,000 分钟，则它与刚刚登台的 8 KB 文件具有相同的优先级。

您可以使用 `no_release` 和 `display_all_candidates` 指令以及手动运行释放程序以获得按优先级顺序排列的备选文件列表，以便根据该表来调整优先级权重。

手动运行释放程序

有时，您可能需要手动运行释放程序。要进行此项操作，您需要知道文件系统的安装点以及释放程序试图达到的下限。

例如，要释放 /sam1 文件系统中的文件，直至其占用率达到 47%，可以超级用户身份登录，并键入以下命令：

```
# /opt/SUNWsamfs/sbin/sam-releaser /sam1 47 1.0
```

所有的命令行选项将取代 releaser.cmd 文件中指定的所有选项。当释放程序运行时，它将在屏幕上显示有关信息并写入至释放程序日志文件 （如果已在 releaser.cmd 文件中指定）。有关更多信息，请参见 sam-releaser(1M) 手册页。

第 5 章

登台

登台是将文件数据从近线或脱机存储设备复制回联机存储设备的过程。登台功能可以使您立即登台文件、不登台文件、指定部分登台以及指定其他登台操作。例如，不登台功能可供随机从大文件访问一小段记录的应用程序使用；启用该功能时，系统不需将文件登台为联机文件便可直接从归档介质中访问数据。

本章介绍 Sun StorEdge SAM-FS 文件登台功能。它包括下列主题：

关于 stager.cmd 文件

可以使用 stager.cmd 文件指定登台程序的操作。该文件的完整路径名为 /etc/opt/SUNWsamfs/stager.cmd。默认情况下，登台程序将执行下列操作：

- 登台程序尝试使用库中的所有驱动器来登台文件。
- 登台缓冲区的大小由介质类型决定，并且不锁定登台缓冲区。
- 不写日志文件。
- 一次最多可以激活 1000 个登台请求。

可以使用 stager.cmd 文件指定有关指令来改写这些默认操作。本节的后半部分将介绍登台程序的指令。有关登台程序指令的其他信息，请参见 stager.cmd(4) 手册页。

第 123 页 “stager.cmd 文件示例”显示了编写完毕的已设置所有可能指令的 stager.cmd 文件。

代码示例 5-1 显示了本章示例中所使用的 mcf 文件。

代码示例 5-1 本章示例中使用的 mcf 文件

```
#
# Sun StorEdge SAM-FS file system configuration example
#
# Equipment       Eq Eq Family Dev Additional
# Identifier      Or Tp Set    St  Parameters
# --------------- -- -- ------ --- ----------
samfs1            60 ms samfs1
/dev/dsk/c1t1d0s6 61 md samfs1 on
/dev/dsk/c2t1d0s6 62 md samfs1 on
/dev/dsk/c3t1d0s6 63 md samfs1 on
/dev/dsk/c4t1d0s6 64 md samfs1 on
/dev/dsk/c5t1d0s6 65 md samfs1 on
#
samfs2             2 ms samfs2
/dev/dsk/c1t1d0s0 15 md samfs2 on
/dev/dsk/c1t0d0s1 16 md samfs2 on
#
/dev/samst/c0t2d0 20 od -      on
/dev/samst/c1t2u0 30 rb dog  on /var/opt/SUNWsamfs/catalog/dogcat
/dev/samst/c1t5u0 31 od dog    on
/dev/samst/c1t6u0 32 od dog    on
/dev/rmt/0cbn     40 od -      on
/dev/samst/c1t3u1 50 rb bird on /var/opt/SUNWsamfs/catalog/birdcat
/dev/rmt/2cbn     51 tp bird   on
```

▼ 创建或修改 stager.cmd 文件并传播更改

1. **使用 vi(1) 或其他编辑器来编辑 stager.cmd 文件。**
   此文件的完整路径如下所示：

```
/etc/opt/SUNWsamfs/stager.cmd
```

有关可在这个文件中添加哪些指令的信息，请参见以下几节：

- 第 119 页 “指定驱动器数量”
- 第 119 页 “设置登台缓冲区大小”
- 第 120 页 “指定日志文件”
- 第 123 页 “指定登台请求的数量”

2. **保存并关闭 `stager.cmd` 文件。**

3. **使用带 `config` 选项的 `samd`(1M) 命令传播文件更改并重新启动系统。**

```
# samd config
```

指定驱动器数量

默认情况下，登台程序在登台文件时使用所有可用的驱动器。如果登台程序使所有驱动器处于繁忙状态，则会影响归档程序的活动。 `drives` 指令用于指定登台程序可用的驱动器数量。此指令的格式如下：

```
drives = library  count
```

表 5-1 *drives* 指令的参数

参数	含义
library	Sun StorEdge SAM-FS `mcf` 文件中定义的自动化库的系列集名。
count	所要使用的驱动器的最大数量。默认情况下，此数量与在 `mcf` 文件中为该库配置的驱动器数量相同。

例如，以下指令行表示只使用 `dog` 系列集库中的一个驱动器来登台文件：

```
drives = dog 1
```

有关 `mcf` 文件的更多信息，请参见 `mcf`(4) 手册页。

还可使用 File System Manager 软件来指定此指令。请参见 File System Manager 联机帮助，以了解更多信息。

设置登台缓冲区大小

默认情况下，登台程序在将要登台的文件从归档介质恢复至联机磁盘高速缓存之前，首先将此文件读入缓冲区中的内存。可以使用 `bufsize` 指令来设置非默认的缓冲区大小和锁定缓冲区 （可选）。这些操作可以改善系统性能。可以尝试不同的 *buffer_size* 值以确定最适合的缓冲区大小。此指令的格式如下：

```
bufsize = media  buffer_size  [ lock ]
```

表 **5-2** *bufsize* 指令的参数

参数	含义
media	指定 *mcf*(4) 手册页中列出的某归档介质类型。
buffer_size	指定一个 2 至 32 的值，默认值为 4。这个值乘以该介质类型的 *dev*_blksize 值，计算结果即为所使用的缓冲区大小。可以在 defaults.conf 文件中指定 *dev*_blksize 的值。为 *buffer_size* 指定的数值越大，所使用的内存就越多。有关此文件的更多信息，请参见 defaults.conf(4) 手册页。
lock	lock 参数指明登台程序在登台归档副本时是否使用锁定的缓冲区。如果指定 lock，归档程序将在复制操作期间在内存中的归档缓冲区上设置文件锁定。这可以避免由于为每一个 I/O 请求锁定和取消锁定缓冲区而造成的开销，从而减少占用系统 CPU 的时间。 仅在配有大量内存的大型系统上，才有必要指定 lock 参数。如果内存不足，则可能会导致内存用尽。 只有已为需要登台的文件启用直接 I/O 时， lock 参数才有效。默认情况下，不会指定 lock 参数，并且文件系统会在所有直接 I/O 缓冲区上设置锁定 （包括用于登台的缓冲区）。有关启用直接 I/O 的更多信息，请参见 setfa(1) 手册页、 sam_setfa(3) 库例程手册页，或 mount_samfs(1M) 手册页中介绍的 -O forcedirectio 选项。

例如，可以按以下方式在 stager.cmd 文件的指令行中指定该指令：

```
bufsize=od 8 lock
```

还可以使用 File System Manager 软件来指定此指令。请参见 File System Manager 联机帮助，以了解更多信息。

指定日志文件

可以要求 Sun StorEdge SAM-FS 软件收集文件登台事件的有关信息，并将这些信息写入日志文件。 logfile 指令用于指定登台程序可在其中写入记录信息的日志文件。此指令的格式如下：

```
logfile=filename [ event ]
```

其中的 *filename* 用于指定完整的路径名。

event 用于指定一个或多个登台事件。如果指定了多个 *event*，应该使用空格分隔每个 *event*。默认情况下启用以下事件：`finish cancel error`。下面列出了一些可能的 *event*：

表 5-3 *event* 参数的关键字

event	操作
`all`	记录所有登台事件。
`start`	记录文件开始登台的时间。
`finish`	记录文件结束登台的时间。默认启用。
`cancel`	记录操作员取消登台操作的时间。默认启用。
`error`	记录登台错误。默认启用。

指定了日志文件后，登台程序会将每一个登台文件的有关信息写入至日志文件中的一行或多行。该行中包括文件名、登台日期和时间以及 VSN 等信息。

以下指令行指定了文件 `/var/adm/stage.log`：

```
logfile=/var/adm/stage.log
```

代码示例 5-2 显示了登台程序日志文件的示例。

代码示例 5-2 登台程序日志文件示例

```
S 2003/12/16 14:06:27 dk disk01 e.76d 2557.1759 1743132 /sam1/testdir0/filebu 1
root other root 0
F 2003/12/16 14:06:27 dk disk01 e.76d 2557.1759 1743132 /sam1/testdir0/filebu 1
root other root 0
S 2003/12/16 14:06:27 dk disk02 4.a68 1218.1387 519464 /sam1/testdir1/fileaq 1
root other root 0
S 2003/12/16 14:06:43 dk disk01 13.ba5 3179.41 750880 /sam1/testdir0/filecl 1
root other root 0
F 2003/12/16 14:06:43 dk disk01 13.ba5 3179.41 750880 /sam1/testdir0/filecl 1
root other root 0
S 2003/12/16 14:06:59 dk disk01 17.167b 1155.1677 1354160 /sam1/testdir0/filedb
1 root other root 0
F 2003/12/16 14:06:59 dk disk01 17.167b 1155.1677 1354160 /sam1/testdir0/filedb
1 root other root 0
S 2003/12/16 14:06:59 dk disk02 f.f82 3501.115 1458848 /sam1/testdir1/filecb 1
root other root 0
S 2003/12/16 14:07:15 dk disk01 1f.473 1368.1419 636473 /sam1/testdir0/fileed 1
root other root 0
S 2003/12/16 14:07:15 dk disk02 16.f15 3362.45 1065457 /sam1/testdir1/filecz 1
root other root 0
```

代码示例 5-2 登台程序日志文件示例 （续）

```
S 2003/12/16 14:07:31 dk disk01 23.201d 3005.1381 556807 /sam1/testdir0/fileeq
1 root other root 0
S 2003/12/16 14:07:47 dk disk01 26.c4d 2831.1113 1428718 /sam1/testdir0/fileez
1 root other root 0
S 2003/12/16 14:07:47 dk disk02 1b.835 3736.59 1787855 /sam1/testdir1/filedp 1
root other root 0
```

如代码示例 5-2 所示，登台程序日志文件包含的信息行可拆分为九个字段。表 5-4 对登台程序日志文件字段的内容进行了介绍。

表 5-4 登台程序日志文件字段

字段	内容描述
1	登台活动。 `S` 表示开始登台。 `C` 表示取消登台。 `E` 表示登台出错。 `F` 表示完成登台。
2	登台操作发生的日期，格式为 *yyyy*/*mm*/*dd*。
3	登台操作发生的时间，格式为 *hh*:*mm*:*ss*。
4	归档介质类型。有关介质类型的信息，请参见 `mcf`(4) 手册页。
5	VSN。
6	归档文件 （`tar`(1) 文件）在介质上的起始物理位置和归档文件中的文件偏移量 （采用十六进制表示）。
7	Inode 编号和世代编号。世代编号是在索引编号被重新使用后生成的一个附加编号，它与索引编号一起用来标识使用的唯一性。
8	文件的大小。
9	文件的名称。
10	归档副本数。
11	文件的用户 ID。
12	文件的组 ID。
13	请求者的组 ID。
14	待登台文件所在驱动器的设备序号。

还可以使用 File System Manager 软件来指定此指令。有关更多信息，请参见 File System Manager 联机帮助。

指定登台请求的数量

可以使用 maxactive 指令来指定一次可以执行的登台请求的数量。此指令的格式如下：

```
maxactive=number
```

默认情况下， *number* 的值为 4000。所允许的最小值为 1。

例如，以下指令行指定队列中最多可以同时存在 500 个登台请求。

```
maxactive=500
```

stager.cmd 文件示例

代码示例 5-3 显示了 stager.cmd 文件的一个示例。

代码示例 5-3 stager.cmd 文件示例

```
# This is stager.cmd file /etc/opt/SUNWsamfs/stager.cmd
drives=dog 1
bufsize=od 8 lock
logfile=/var/adm/stage.log
maxactive=500
```

archiver.cmd 文件在登台过程中的角色

archiver.cmd 文件中的大多数指令会影响归档过程，但归档集分配指令可以使您指定应用于归档集中所有文件的登台属性。归档集分配指令的格式如下：

```
archive_set_name  path  [search_criteria ...] [file_attributes]
```

第 31 页 “归档”一章中详细地介绍了归档集分配指令及其参数。表 5-5 显示了归档集分配指令中可用作 *file_attributes* 的登台指令。

表 5-5 `archiver.cmd` 文件中的登台 *file_attributes*

指令	作用
`-stage a`	指定归档集中的文件应联合登台。
`-stage d`	重置为默认设置。
`-stage n`	指定不登台归档集中的文件。

有关这些指令及其他 `archiver.cmd` 指令的更多信息，请参见第 31 页 “归档”。

排列预备请求的优先顺序

归档程序和登台程序进程均可请求载入及卸载介质。如果请求的数量超过可用于介质载入的驱动器数量，则多余的请求会发送至预备队列。

预备队列中的归档和登台请求是指那些无法立即满足的请求。默认情况下，系统按先进先出 (First In First Out, FIFO) 的顺序执行预备请求。

预备队列中的条目数量取决于 `defaults.conf` 文件中的 `previews=` 指令。有关更改该指令值的信息，请参见 `defaults.conf`(4) 手册页。

您可以为各个预备请求分配不同的优先级。可以通过在预备命令文件中输入指令来改写 FIFO 默认值，此预备命令文件的位置如下：

```
/etc/opt/SUNWsamfs/preview.cmd
```

此文件根据请求是用于登台还是归档来安排预备请求。也可提高特定 VSN 的优先级。此外， `preview.cmd` 文件中的设置也可根据上限 (High Watermark, HWM) 或下限 (Low Watermark, LWM) 设置，来重新分配所有或特定文件系统的预备请求的优先级。

`sam-amld` 守护进程在启动时读取预备请求指令。这些指令必须是每一条独占一行。如果在 `sam-amld` 守护进程正在运行时更改此文件，那么必须重新启动 `sam-amld` 守护进程，以使更改生效。注释行以井字符 (#) 开头，并且延伸至行的末尾。有关此文件的更多信息，请参见 `preview.cmd`(4) 手册页。

`preview.cmd` 文件可以包含以下两种类型的指令：

- 全局指令，应用于所有文件系统。这些指令必须出现在第一行 `fs =` 前。

- 专用于文件系统的指令，它位于全局指令的后面。与 `archiver.cmd` 文件相似，`preview.cmd` 文件可以包含专用于单个文件系统的指令。在此文件中，专用于单个文件系统的指令必须出现在所有全局指令的后面。

  这些文件系统指令必须以 `fs = `*file_system_name* 指令开头。该指令用于指定文件系统，其后的所有指令均应用于该文件系统。文件中可以包含多个文件指令块。文件系统指令的应用范围到出现下一行 `fs =` 或到达文件末尾为止。

注 – 当多条指令影响一个文件系统时，专用于文件系统的指令将取代全局指令。

全局 VSN 和时限指令

VSN 和时限优先级指令均是全局指令。如果在 `preview.cmd` 文件中指定这些指令，则它们必须出现在任何专用于文件系统的指令的前面。也就是说，它们必须出现在所有 `fs =` 指令的前面。 VSN 优先级指令的格式如下：

```
vsn_priority = value
```

该指令是一个静态的优先级因子，它表示具有高优先级的 VSN 的总优先级将要增加的值。`vsn_priority` 的默认值为 1000.0。当 VSN 被安排为预备请求时，它们必须已被设置优先级标记，才能获得此增加值。可以使用带 `p` 选项的 `chmed`(1M) 命令，来设置优先级标记 （例如 `chmed +p lt.AAA123`）。只有提交的 VSN 请求尚不是预备请求时，对它们设置此优先级标记才会有效。时限优先级指令的格式如下：

```
age_priority = factor
```

该指令是一个静态的优先级因子，但它的整体影响是动态的。 `age_priority` 因子与请求成为预备请求的秒数 （即请求等待的时间）相乘，其结果将与请求的总优先级相加。请求的等待时间越长，时限因子就越大。设置该因子有助于确保较旧的请求，不会被具有其他较高优先级因子的较新请求无限期地取代。

如果该因子大于 1.0，则它可以增加时间因子在计算总优先级中的重要性。如果小于 1.0，则会降低时间因子的重要性。如果将该因子设置为 0.0，则在计算总优先级级时不考虑时间因子。

对于没有设置优先级标记的 VSN，系统将根据它在队列中等待的时间提高其优先级。这样，该 VSN 的优先级可能会高于以后进入队列且已设置优先级标记的 VSN。

全局或文件系统专用的界限指令

预备请求的界限指令既可以用作全局指令，也可用作文件系统的专用指令。界限指令用于确定预备请求的界限优先级 (wm_priority)。代码示例 5-4 表明 wm_priority 因子是多个设定值的和。

代码示例 5-4 wm_priority 的计算方法

```
lwm_priority +
lhwm_priority +
hlwm_priority +
hwm_priority
_____________________
= wm_priority
```

当 wm_priority 因子是正数时，计算出的总优先级所产生的影响是：提高归档请求的优先级而降低登台请求的优先级。不过， wm_priority 因子还可以是负数。在此情况下，将会降低归档请求的总优先级，这导致系统优先处理登台请求，然后再处理归档请求。如果将此因子设置为 0.0 （或根本不指定命令），则表示在文件系统遇到此情况时，系统不会对归档请求采取任何特殊的操作。有关此因子的更多信息，请参见第 128 页 “示例 1：强制执行登台请求”中的示例。

表 5-6 显示了四个界限优先级指令及其参数

表 5-6 界限优先级指令

优先级指令	参数
lwm_priority = *value*	其中的 *value*，用于指定当文件系统低于下限时，归档请求的 wm_priority 因子的更改数量。默认设置为 0.0。
lhwm_priority = *value*	其中的 *value*，用于指定当文件系统从超过下限上升至下限以上，但仍然低于上限时，归档请求的 wm_priority 因子的更改数量。这通常表示文件系统中的文件正在增加。默认设置为 0.0。
hlwm_priority = *value*	其中的 *value*，用于指定当文件系统从超过上限降至上限以下，但仍然低于下限时，归档请求的 wm_priority 因子的更改数量。这通常表示释放程序不能释放足够的磁盘空间，以使文件系统低于下限。默认设置为 0.0。
hwm_priority = *value*	其中的 *value*，用于指定当文件系统超过上限时，归档请求的 wm_priority 因子的更改数量。默认设置为 0.0。

总之，这四个界限设置用于创建包括百分比值 （表示文件系统的占用率）以及上限和下限设置级别在内的动态优先级因子。分配给预备请求的值取决于优先级因子是全局性的，文件系统专用的，还是未设置。

当文件系统的情况发生变化时，系统将根据以下条件重新计算与该文件系统关联的每一个 VSN 的优先级：相应的界限优先级设置；是否使用 chmed(1M) 命令的 p 选项设置了优先级。

界限优先级仅用于计算与归档有关的介质请求，而不能用于计算与登台有关的介质请求。

以下示例指令显示了如何在文件系统处于上下限位置时，小幅度地提高归档请求的优先级。代码示例 5-5 显示了一些设置，这些设置可用于启用释放程序以释放足够的磁盘空间，从而使文件系统低于下限。

代码示例 5-5 使文件系统低于下限的设置

```
lhwm_priority = -200.0
hlwm_priority = 100.0
```

计算预备请求的总优先级

预备请求的优先级数值由数个静态和动态因子共同决定。数值越大，优先级就越高。静态优先级因子在生成请求时进行设置。一旦请求生成并进入等待执行状态，静态优先级因子对总优先级的影响将不会发生变化。在请求等待执行期间，动态优先级因子可以提高或降低请求的总优先级。

预备请求的总优先级是所有优先级因子的总和，其计算方式如下：

```
total priority = vsn_priority + wm_priority + (age_priority *
time_in_sec_as_preview_request)
```

设置预备请求的优先级方案

除非绝对需要，否则请勿更改默认的预备请求 FIFO 方案。在下列条件下，可能需要更改默认的预备请求 FIFO 方案：

- 条件 1：确保首先处理登台请求，然后处理归档请求。
- 条件 2：确保归档请求在文件系统将要充满时获得最高优先级。
- 条件 3：将使用特定介质组的请求排在预备请求列表的顶部。

对于用户对数据访问要求极高、 VSN 驱动器数受限制或将文件归档作为后台功能的环境，可以使用 `preview.cmd` 文件来改变存储系统资源执行登台请求的方式。可以对 `preview.cmd` 文件的设置进行自定义，以支持上述所有方案，并影响已配置的 Sun StorEdge SAM-FS 环境。

由于该文件中的设置对数据并无影响，因此建议尝试并调整不同的指令设置，以便在权衡每一个预备请求的优先级时，能够在归档请求和登台请求之间找到适当的平衡点。

代码示例 5-6 显示了可以满足上述三个条件的 preview.cmd 文件。

代码示例 5-6 preview.cmd 文件示例

```
# condition 1
lwm_priority = -200.0
lhwm_priority = -200.0
hlwm_priority = -200.0
# condition 2
hwm_priority = 500.0
# condition 3
age_priority = 1.0
```

示例 1：强制执行登台请求

以下设置示例提供了一种可以确保登台请求先于归档请求处理的方法。本示例假定：

- 队列中已经有多个请求的等待时间达到了 100 秒。
- vsn_priority 的默认值为 1000。

表 5-7 显示了如何计算请求的总优先级。

表 5-7 请求优先级计算示例

优先级	计算
具有优先级的归档 VSN，下限：	1000 + (-200) + (1 x 100) = 900
具有优先级的登台 VSN，下限：	1000 + 0 + (1 x 100) = 1100
无优先级的登台 VSN，下限：	0 + 0 + (1 x 100) = 100

本示例表明在其他因子都相同时， wm_priority 为负值会使系统优先处理登台请求，然后再处理归档请求。

示例 2：强制执行归档请求

当在环境中权衡将文件登台回用户与将新文件归档至介质这二者的重要性时，最为关心的是超过上限的情形。在此情况下，如果没有足够的满足归档要求的文件来降低文件系统的占用率，则完成待定的归档请求是防止文件系统充满的次佳方法。

在此情况下，只需在 preview.cmd 文件中进行以下设置：

```
hwm_priority = 500.0
```

示例 3：根据介质确定请求的优先级

在面向项目的环境中，特定用户可能只处理占用特定 VSN 的文件组，并且其数据与其他用户的数据相互分开。在这种环境中，某些项目有时可能具有较高的优先级；因此，它们需要具有优先使用可用系统存储资源的权利。这时可以使用以下指令配置 preview.cmd 文件，以适当给予用户及其介质优先使用介质驱动器的权利：

```
hwm_priority = 5000.0
```

然后，对于每一个属于优先级用户群组的 VSN，输入以下信息：

```
# chmed +p lt.AAA123 ## or whatever VSN is used
```

以后，每一个要求访问 VSN AAA123 （或使用的任何 VSN）的请求均优先于预备队列中的其他待定安装请求。

将来，要降低用户介质的优先级，请为每一个 VSN 输入相反的命令：

```
# chmed -p lt.AAA123 ## or whatever media type is used
```

示例 4：确定复杂请求的优先级

假设存在两个具有以下要求的 Sun StorEdge SAM-FS 文件系统：

- 请求在队列中的等待时间不能太长 (age_priority)。
- 当文件系统低于下限时，优先处理登台请求。
- 当文件系统高于下限且低于上限时，无需区分归档请求或登台请求的优先级。代码示例 5-7 显示了受影响的指令。

代码示例 5-7 指令

```
lwm_priority = -200.0
lhwm_priority = 0.0
hlwm_priority = 0.0
```

在此情况下，其他指令保持不变。

当文件系统超过上限时，优先处理归档请求。

如果两个文件系统同时超过了上限，则应首先防止第二个文件系统 （例如 samfs2）充满。例如，当 samfs1 是一个用户工作文件系统，而 samfs2 是一个关键文件系统时，便会出现这种情况。

在任何情况下（无论出现何种情形），如果已设置 chmed(1M) 命令的 p 标记，则要求访问选定 VSN 组的请求将优先于预备请求队列中的其他请求。

代码示例 5-8 显示了可根据前面列出的要求确定请求优先级的 preview.cmd 文件。

代码示例 5-8 preview.cmd 文件

```
age_priority = 100.0
vsn_priority = 20000.0
lhwm_priority = -200.0
hlwm_priority = -200.0
fs = samfs1
hwm_priority = 1000.0
fs = samfs2
hwm_priority = 5000.0
```

第 6 章

回收

回收是指从归档卷中收回空间的过程。回收程序与归档程序配合工作，以收回由无用的归档副本占用的空间。当用户修改某个文件时，即可从系统中清除与该文件的旧版本相关联的归档副本。回收程序可以识别那些其中绝大部分是过期归档副本的归档卷，并将这些卷中的非过期副本移动到其他卷中。当某个给定的卷中只包含过期副本时，即可执行站点定义的操作。例如，可以重新标记此类卷以便立即重新使用此类卷，或将其中的数据导出至离站存储设备，从而单独保存文件更改的历史记录。由于回收过程只与用户的数据文件有关，因此用户不会觉察到回收过程。

本章包括下列主题：

回收过程概述

回收程序负责将过期归档副本占用的空间，保持在由站点指定参数定义的最低水平。在任何时候，给定归档卷的空间均由以下各项组成：

- **当前数据空间**，由当前有效的归档映像占用的空间。
- **过期数据空间**，由当前不再有效的归档映像占用的空间。
- **可用空间**，未被当前有效或过期的归档映像占用的空间。

卷的**容量**是指卷中可用于存储数据的总空间量。例如，对于一个已写入 3 GB 数据的 10 GB 磁带卷来说，它的容量为 10 GB，可用空间为 7 GB。

对于全新的归档介质或新标记的归档介质，其容量等于可用空间。当将数据归档至该介质时，可用空间会减少，而当前数据空间会增加。

当更改或删除文件系统中已归档的文件时，这些文件的归档映像会过期，并且其类别由当前数据类别变为过期数据类别。这些映像占用的物理空间并没有发生变化；只是文件系统中已没有指向该空间的文件。

这些过期的映像 （即过期数据）最终会占满全部可用空间。只有回收空间，才能删除这些映像并使它们占用的空间变为可用空间。回收程序的目标是将过期数据占用的空间转变为可用空间，而丝毫不损坏任何当前数据。

例如，只能在可移除介质卡盒 （如磁带）中添加数据，而不能在其中重新写入数据。重新使用卡盒的唯一方法是，从卡盒中移走所有当前数据，重新标记卡盒，然后从头开始使用卡盒。

通过输入 `sam-recycler(1M)` 命令来启动回收过程。回收过程既可手动执行，也可通过 `cron(1)` 作业来执行。表 6-1 介绍了回收方法。

表 6-1 回收方法与介质类型

回收方法	介质和说明
按自动化库	可移除介质卡盒。 当按库进行归档时，可以在 `recycler.cmd` 文件中指定回收指令。
按归档集	可移除介质卡盒和磁盘。 当按归档集进行归档时，请勿使用 `recycler.cmd` 文件。而应将所有回收指令放置在 `archiver.cmd` 文件中。

如表 6-1 所示，既可以按库进行回收，也可以按归档集进行回收。如果归档至磁盘，则只能按归档集进行回收。

回收程序和归档程序协同工作，具体如下：

1. 回收程序使用 `rearchive` 属性来标记卷中所有当前 （有效）的归档映像。
2. 如果是将文件归档至可移除介质，则回收程序使用 `recycle` 属性来标记所选择的归档卷。这可阻止归档程序再向此卷中写入归档映像。
3. 归档程序将所有已做标记的映像移至另一个卷。此操作过程称为**重新归档**。当归档程序将当前的归档映像从旧卷移至新卷后，旧卷中只包含可用空间和过期数据空间。如果是归档至可移除介质卡盒，则可以重新标记并重新使用该卡盒。如果是归档至磁盘，则回收程序将删除包含已过期归档映像的文件。

回收程序可以定期运行。每次启动该程序后，它会尽力完成所有工作。在归档程序可以重新归档文件之前，回收程序必须完成要重新归档的副本的标记工作。

有时，设置了 `rearchive` 属性的已过期的归档映像仍会保留在介质中。这可能会发生在下列情况下：

- 在回收程序标记已过期的归档映像后，归档程序没有运行。
- 在移动未过期的归档映像时，归档程序没有可使用的介质。
- 归档程序出现了其他各种异常。

在两次运行期间，回收程序将状态信息保留在库目录和 inode 中。在回收过程中，可以使用 sls(1) 命令及其 -D 选项来显示文件的相关信息。 sls(1) 命令的输出可以显示，是否已经安排对某个文件进行重新归档。

使用回收指令

recycler.cmd 文件可以接受的指令如以下几节所述：

- 第 133 页 “指定日志文件：logfile 指令”
- 第 133 页 “阻止回收：no_recycle 指令”
- 第 134 页 “指定回收整个自动化库：库指令”

指定日志文件：logfile 指令

logfile 指令用于指定回收程序日志文件。此指令的格式如下：

```
logfile = filename
```

其中的 *filename*，用于指定日志文件的路径。

以下是 logfile= 指令行的一个示例：

```
logfile=/var/adm/recycler.log
```

阻止回收：no_recycle 指令

no_recycle 指令可以阻止卷的回收。要指定 VSN，可以使用正则表达式以及一个或多个特定介质类型。此指令的格式如下：

```
no_recycle media_type VSN_regex [ VSN_regex ... ]
```

表 **6-2** no_recycle 指令的参数

参数	含义
media_type	指定 mcf(4) 手册页中列出的某一介质类型。
VSN_regexp	指定一个或多个由以空格分隔的正则表达式所描述的卷。有关正则表达式的格式信息，请参见 regexp(5) 手册页或第 57 页 “使用模式匹配的文件名 search_criteria：-name regex”。

指定 *media_type*，可以阻止回收存储在特定介质类型上的卷。可以通过指定一个或多个 *VSN_regexp* 规范，使用正则表达式来标识那些免于回收的特定卡盒。

例如，以下指令行可阻止回收程序回收那些 VSN 标识以 DLT 开头的磁带卷：

```
no_recycle lt DLT.*
```

指定回收整个自动化库：库指令

库指令可用于为那些与特定库关联的 VSN 指定各种不同的回收参数。此指令的格式如下：

```
library  parameter  [  parameter  ...  ]
```

其中的 *library*，用于指定库名称，该名称与在 mcf(4) 文件的系列集字段中指定的名称相同。

而其中的 *parameter*，用于指定一个或多个以空格分隔的 *parameter* 关键字 （见表 6-3）。

表 **6-3** 库指令中 *parameter* 的值

parameter	操作
-dataquantity *size*	限制回收程序可以安排重新归档的数据量，以免清除有用数据所在的卷。默认值为 1 GB。
-hwm *percent*	库利用率的上限。默认值为 95。
-ignore	阻止回收库中的卷。在测试 recycler.cmd 文件时，此指令十分有用。
-mail *email_address*	向指定的 *email_address* 发送回收电子邮件消息。默认情况下，系统不发送电子邮件。
-mingain *value*	最小 VSN 增益百分比。默认值为 50。
-vsncount *count*	限制可回收卷的数量。默认值为 1。

以以下的指令行为例：

```
gr47 -hwm 85 -ignore -mail root -mingain 40
```

它为库 gr47 指定了以下各项：

- 当库中卷的占用率达到 85% 时，应考虑对库执行回收操作。
- 最小增益百分比为 40%。
- 重新归档的数据量最多为 1 GB。这是一个默认值，因此未在 recycler.cmd 文件中指定。
- 只能回收一个卷。这也是一个默认设置。
- 将回收消息发送至 root 用户。

设计回收操作

配置回收程序之前，请注意以下事项：

- archiver.cmd 文件中的指令按归档集控制回收过程。recycler.cmd 文件中的指令按库控制回收过程。此外， recycler.cmd 文件还控制着回收程序的一般操作。有关回收程序指令的信息，请参见第 133 页 “使用回收指令”。
- 请勿回收包含可移除介质文件的卷。可移除介质文件可以使用 request(1) 命令来创建。回收程序不会保留由 request(1) 命令创建的可移除介质文件。包含可移除介质文件的卷不能清除干净。
- 当 Sun StorEdge SAM-FS 文件系统正在执行维护操作时，请勿运行回收程序。回收程序使用 .inodes 文件和 mcf 文件，来识别当前文件或过期文件以及与文件系统关联的设备。如果没有这些文件中的正确信息，回收程序可能会将当前归档数据视为过期数据而对其执行回收操作。
- 所有 Sun StorEdge SAM-FS 文件系统必须都已安装，方可运行回收程序。如果要对联机磁盘执行回收操作，则必须安装包含此磁盘卷的文件系统，并且主机系统必须可访问。

默认情况下，系统不会启用回收程序。必须通过输入 sam-recycler(1M) 命令来启动回收过程。启动回收程序后，第 134 页 “指定回收整个自动化库：库指令”中指定的默认回收程序设置将会生效。有关回收程序的更多信息，请参见 sam-recycler(1M) 手册页。

以下几节介绍回收程序的配置过程。此过程包括下列步骤：

- 第 136 页 “步骤 1：创建 recycler.cmd 文件”
- 第 138 页 “步骤 2：编辑 archiver.cmd 文件”
- 第 139 页 “步骤 3：运行回收程序”

- 第 141 页 “步骤 4：为回收程序创建 crontab 文件”
- 第 141 页 “步骤 5：删除 -recycle_ignore 和 ignore 参数”
- 第 141 页 “步骤 6：创建 recycler.cmd 文件”

如果要回收库中的卡盒，则配置过程中还需创建 recycler.cmd 文件以及 （可选）编辑 archiver.cmd 文件。如果将文件归档至磁盘，则只能按归档集进行归档，因此要启用这些磁盘卷的回收过程，还需要编辑 archiver.cmd 文件。以下过程介绍了如何使用 recycler.cmd 和 archiver.cmd 文件为任意归档介质配置回收程序。另外，还可以使用 File System Manager 软件配置回收过程。有关更多信息，请参见 File System Manager 联机帮助。如果您是通过 File System Manager 配置回收过程的，则您还需要完成以下的步骤 3、步骤 4 和步骤 6。

▼ 步骤 1：创建 recycler.cmd 文件

如果要回收库中卡盒上的归档副本，请执行本步骤。

如果要回收磁盘卷上的归档副本，则不必执行本步骤，因为回收过程由 archiver.cmd 文件中的指令控制。有关在 archiver.cmd 文件中配置回收过程的信息，请参见第 138 页 “步骤 2：编辑 archiver.cmd 文件”。

recycler.cmd 文件中包含了通用的回收指令，也可包含针对 Sun StorEdge SAM-FS 环境中每一个库的指令。有关回收指令的信息，请参见第 133 页 “使用回收指令”。

即使按归档集执行回收操作，也应在 recycler.cmd 文件中配置每一个库。这可以确保回收程序能够回收不属于归档集的 VSN （如有必要）。

典型的 recycler.cmd 文件中包含以下指令行：

- logfile= 指令行，用于指定回收程序日志文件。系统将回收消息和回收报告写入至该文件。
- 针对每一个含有待回收卷的库的一行或多行指令。该指令行必须包含要回收的库的系列集名 （来自 mcf 文件）。这使回收程序可以识别库。

由于仍在创建 recycler.cmd 中的指令行，并且尚未经过测试，因此应使用 ignore 关键字。本过程后面的步骤中会指导您删除 ignore 关键字。

要创建 recycler.cmd 文件，请执行以下步骤：

1. **成为超级用户**。
2. **使用 vi(1) 或其他编辑器打开文件 /etc/opt/SUNWsamfs/recycler.cmd**。
3. **添加本章所述的一行或多行指令来控制回收程序的活动**。
4. **保存并关闭该文件**。

recycler.cmd 文件示例

代码示例 6-1 显示了 recycler.cmd 文件的一个示例。

代码示例 6-1 recycler.cmd 文件示例

```
logfile = /usr/tmp/recycler.log
stk30  -hwm 51  -mingain 60  -ignore  -mail  root
```

以下部分介绍代码示例 6-1 中指定的参数。

-hwm 51 参数

通过指定上限，可使得当介质使用率低于此上限时，回收过程不能启动。此百分比是指库中已用空间与总容量的比率。例如，某个库含有 10 个 20 GB 的磁带，其中三个磁带的使用率为 100%，另外七个磁带的占用率均为 30%，则介质利用率为：

```
((3* 1.00 + 7 * 0.30) * 20G ) / ( 10 * 20G ) * 100%= 51%
```

请注意，该计算方法并不区分当前数据和过期数据，它只考虑介质的使用量。

在本示例中，如果空间利用率为 51% 或更小，回收程序不会自动选择自动化库的任何 VSN 进行回收。

注 – 可以使用下列命令设置回收标记，从而强制回收某 VSN：

```
# chmed +c lt.AAA123
```

设置 +c 回收标记后，归档程序不会再向该卷中写入任何归档映像。可以通过 samu(1M) 实用程序来查看 +c 标记。有关更多信息，请参见 chmed(1M) 和 samu(1M) 手册页。有关如何使用 samu(1M) 操作员实用程序的信息，请参见《Sun StorEdge QFS 配置和管理指南》。

-mingain 60 参数

minimum VSN gain percentage 用于设置通过回收卡盒所获得的空间增加量的下限。例如，在自动化库的某个卡盒中， 95% 为当前数据空间，另外 5% 是过期数据空间，因此通过回收该卡盒所获得的增益百分比仅为 5%。为获得此空间 （即 5%）而移动另外 95% 的空间，这可能是不值得的。将请最小增益百分比设置为 6% 或更大值，可以防止回收程序自动选择此示例 VSN 进行回收。

另一个示例是某个卡盒中具有 90% 的过期数据空间， 5% 的当前数据空间，以及 5% 的可用空间。如果回收此卡盒，则会获得 90% 的增益。

-ignore 参数

-ignore 参数用于防止回收程序回收某个特定的库，在配置回收程序时，应使用此关键字。

-mail 参数

-mail 关键字用于指定回收程序在回收指定的库后发送电子邮件。邮件消息的主题行如下所示：

```
Robot robot-name recycle
```

表 6-2 显示了消息正文范例。

代码示例 6-2 回收消息示例

```
I will recycle VSN vsn.
Cannot find any candidate VSN in this media changer.
Previously selected VSN vsn is not yet finished recycling.
Previously selected VSN vsn is now finished recycling. It will now
be post-recycled.
```

▼ 步骤 2：编辑 archiver.cmd 文件

如果按归档集执行回收，请执行本步骤。如果将文件归档至磁盘，则按归档集执行回收是唯一可行的方法，因此必须完成本步骤才能进行回收。

如果按库执行回收，请继续下一步骤。

- **要编辑 archiver.cmd 文件，请执行第 42 页“创建或修改 archiver.cmd 文件并传播更改”所述步骤。**

  archiver.cmd 文件中添加的用于启用按归档集回收的指令，必须位于 params 和 endparams 指令之间。表 6-4 显示了可使用的归档集回收指令。

表 6-4 归档集回收指令

指令	功能
-recycle_dataquantity *size*	限制回收程序可以安排重新归档的数据量，以免清除有用数据所在的卷。
-recycle_hwm *percent*	设置上限百分比。
-recycle_ignore	阻止回收归档集。

表 6-4 归档集回收指令 （续）

指令	功能
-recycle_mailaddr *mail_address*	将回收程序消息发送到 *mail_address*。
-recycle_mingain *percent*	将回收对象限制为那些按 *percent* 或更大百分比增加可用空间的 VSN。
-recycle_vsncount *count*	将可重新归档的卷的数量限制为 *count*。

有关上述指令的更多信息，请参见第 31 页 “归档”或 archiver.cmd(4) 手册页。

▼ 步骤 3：运行回收程序

1. **运行 sam-recycler(1M) 命令。**

   回收程序将读取 recycler.cmd 文件。

2. **检查标准输出日志、Sun StorEdge SAM-FS 日志和 /var/adm/messages，查看是否存在来自回收程序的错误消息。**

   如果有错误，请更正文件。

   代码示例 6-3 显示了回收可移除介质卡盒时的回收程序日志文件范例。

代码示例 6-3 可移除介质卡盒的回收程序日志文件示例

```
========== Recycler begins at Wed Dec 12 14:05:21 2001 ===========
Initial 2 catalogs:

0  Family: m160                  Path: /var/opt/SUNWsamfs/catalog/m160
   Vendor: ADIC                  Product: Scalar 100
   SLOT                  ty    capacity         space vsn
      0                  at        25.0G        25.0G CLN005
      1                  at        48.5G         6.1G 000003
      2                  at        48.5G        32.1G 000004
      3                  at        48.5G        35.1G 000005
      4                  at        48.5G        44.6G 000044
      5                  at        48.5G        45.1G 000002
      6                  at        48.5G        45.9G 000033
      7                  at        48.5G        48.5G 000001
   Total Capacity:  364.8G bytes, Total Space Available: 282.3G bytes
   Volume utilization 22%, high 95% VSN_min 50%
   Recycling is ignored on this robot.




1  Family: hy                    Path: /var/opt/SUNWsamfs/catalog/historian
```

代码示例 **6-3** 可移除介质卡盒的回收程序日志文件示例 （续）

```
    Vendor: Sun SAM-FS           Product: Historian
    SLOT                  ty    capacity         space vsn
       (no VSNs in this media changer)
    Total Capacity:  0    bytes, Total Space Available: 0    bytes
    Volume utilization 0%, high 95% VSN_min 50%
    Recycling is ignored on this robot.



8 VSNs:

                      ---Archives---   -----Percent-----   m160
 ----Status-----     Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN             0        0     0    87     13    m160:at:000003
no-data VSN             0        0     0    33     67    m160:at:000004
no-data VSN             0        0     0    27     73    m160:at:000005
no-data VSN             0        0     0     8     92    m160:at:000044
no-data VSN             0        0     0     7     93    m160:at:000002
no-data VSN             0        0     0     5     95    m160:at:000033
empty VSN               0        0     0     0    100    m160:at:CLN005
empty VSN               0        0     0     0    100    m160:at:000001



Recycler finished.

========== Recycler ends at Wed Dec 12 14:05:32 2001 ===========
```

代码示例 6-4 显示了回收磁盘归档文件时的回收程序日志文件范例。

代码示例 **6-4** 磁盘归档文件的回收程序日志文件示例

```
---Archives---   -----Percent-----
 ----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
new candidate          0        0     0    41     59  <none>:dk:disk01


677 files recycled from VSN disk01 (mars:/sam4/copy1)
0 directories recycled from VSN disk01 (mars:/sam4/copy1)
```

▼ 步骤 4：为回收程序创建 crontab 文件

如果系统运行正常，即可为超级用户创建 crontab 条目，以定期运行回收程序。您可能希望每隔 2 个小时运行一次回收程序，视您的站点条件而定。

- **创建 crontab 条目。**

  有关信息，请参见 cron(1M) 手册页。

  以下是超级用户的 crontab 文件中的条目示例，它确保 cron 守护进程在每个奇数小时内每隔 5 分钟运行一次回收程序：

```
5 1,3,5,7,9,11,13,15,17,19,21,23   * * * /opt/SUNWsamfs/sbin/sam-recycler
```

▼ 步骤 5：删除 -recycle_ignore 和 ignore 参数

1. **使用 vi(1) 或其他编辑器删除 archiver.cmd 文件中的 -recycle_ignore 参数。**
2. **使用 vi(1) 或其他编辑器删除 recycler.cmd 文件中的 ignore 参数。**

   现在，回收程序便真正开始执行回收过程。

▼ 步骤 6：创建 recycler.cmd 文件

如果您要回收可移除介质卡盒上的归档副本，请执行本步骤。如果只将文件归档至磁盘，则请勿执行本步骤。

在归档程序将 VSN 中的所有当前映像重新归档至另一个 VSN 后，回收程序将执行 recycler.sh 脚本。有关该脚本示例，请参见 recycler.sh(1M) 手册页。可以在 /opt/SUNWsamfs/examples/recycler.sh 中找到另一个示例，它显示了如何重新标记已回收的 VSN 并向超级用户发送电子邮件。

回收程序使用以下参数调用 /opt/SUNWsamfs/sbin/recycler.sh 脚本：

```
Media type: $1  VSN: $2  Slot: $3  Eq: $4
```

当回收程序确定已清除 VSN 中所有已知的有效归档副本后，它将调用 /opt/SUNWsamfs/sbin/recycler.sh 脚本。您应确定您的站点对清除已回收卡盒的要求。某些站点选择重新标记或重新使用卡盒；而其他站点选择从自动化库中取出卡盒以便将来用于访问历史文件。有关更多信息，请参见 recycler(1M) 和 recycler.sh(1M) 手册页。

第 7 章

使用 Sun SAM-Remote 软件

Sun SAM-Remote 客户机和 Sun SAM-Remote 服务器构成的客户机/服务器实施体系，可以使您在 Sun StorEdge SAM-FS 主机系统之间共享库和其他可移除介质设备。您可使用 Sun SAM-Remote 配置多个存储客户机，使它们能够从集中式磁带库或磁光盘库中归档和登台文件。例如，如果您的主机系统分布在一个跨越很大地域的网络中，则在一个城市中创建的文件可以归档至离此城市数英里远的库中的卡盒。

本章包括下列主题：

Sun SAM-Remote 软件概述

本概述包含下列主题：

特性

Sun SAM-Remote 软件具有以下优点：

- 允许在一台或多台 Sun SAM-Remote 客户机之间远程共享价值昂贵的可移除介质资源 （例如库）。
- 允许客户机向服务器转移数据。

- 允许多个 Sun StorEdge SAM-FS 服务器彼此互为主机。在 Sun SAM-Remote 环境中，服务器是在 `mcf` 文件中被配置成 `ss` 设备类型的主机系统。

您可以配置 Sun SAM-Remote 服务器和客户机，以便在两台或多台 Sun Solaris 主机系统之间提供多份归档副本。例如，您可以对两台运行 Sun StorEdge SAM-FS 软件的 Solaris 系统进行配置，使它们相互成为对方的 Sun SAM-Remote 服务器和 Sun SAM-Remote 客户机。这样配置的好处在于，您不仅可以为每一台服务器创建本地副本，而且还可以在另一台服务器上创建额外的数据归档副本。文件系统可在使用标准 NFS 的服务器之间共享。当无法访问本地库时， Sun SAM-Remote 软件会从归档副本中自动恢复文件数据。因此，使用这两台服务器的用户可以连续地访问各自的数据。即使在他们的主存储库不可用时，也是如此。

图 7-1 显示了一个配有两台 Sun SAM-Remote 主机系统服务器的环境。每台服务器均具有两台客户机。

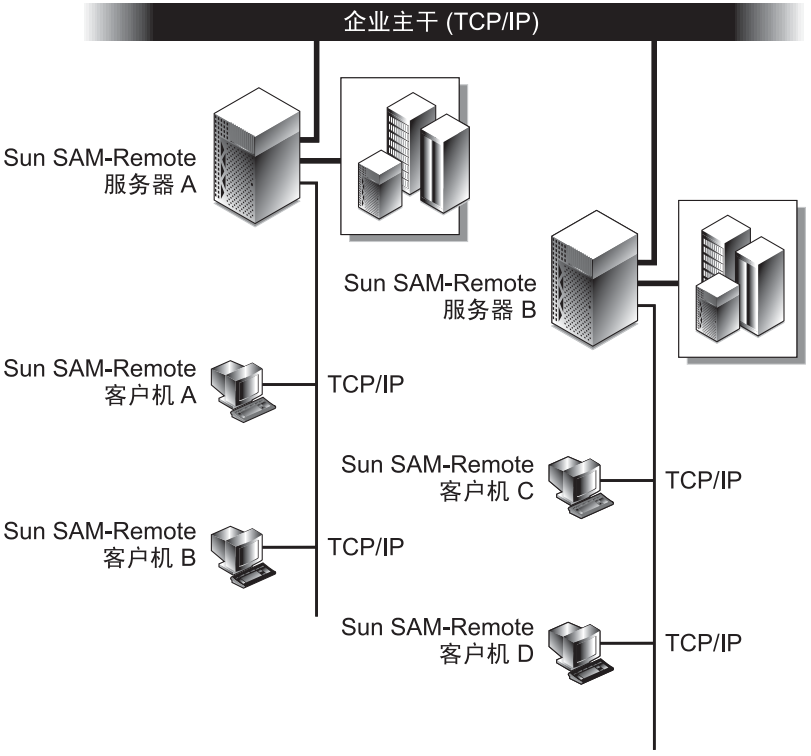

图 7-1 Sun SAM-Remote 服务器和客户机

要求

尝试配置 Sun SAM-Remote 环境之前，请确保您的环境安装了以下软件和硬件：

- SPARC 或 x64 系统应安装已获得使用许可的、可操作的 Sun StorEdge SAM-FS 4U0 版或更新版本的存储和归档管理软件包。
- 主机系统应安装相同修订版本的 Sun StorEdge SAM-FS 软件和相同的修补程序集合。如果某些主机系统需要升级，请参见 《Sun StorEdge SAM-FS 安装和升级指南》了解有关该主题的信息。
- 一个用作 Sun SAM-Remote 服务器的主机系统，其中至少应安装一个 SAM-QFS 文件系统。
- 客户机和安装 Sun StorEdge SAM-FS 软件的服务器之间应通过 TCP/IP 网络进行网络连接。

限制

存储及归档管理器对待远程库中磁带卡盒的方式与对待本地库中磁带卡盒的方式完全一样。不过， Sun SAM-Remote 软件存在以下限制：

- 您可以尝试使用 Sun SAM-Remote 回收介质，但前提是您已彻底测试您的环境。有关更多信息，请参见第 162 页 “使用 Sun SAM-Remote 软件回收”。
- Sun SAM-Remote 客户机上只有一个守护进程可以与 Sun SAM-Remote 服务器通信。
- Sun StorEdge SAM-FS 软件不能操作共享的 Sun StorEdge QFS 文件系统中的 Sun StorEdge QFS 客户机，因此 SAM-Remote 也不能进行这样的操作。当运行 Sun StorEdge SAM-FS 和 SAM-Remote 的服务器既是某些 Sun StorEdge QFS 文件系统的元数据服务器，又是其他 Sun StorEdge QFS 文件系统的客户机时，Sun StorEdge SAM-FS 和 SAM-Remote 只能操作那些将该服务器作为元数据服务器的文件系统。

技术概述

Sun SAM-Remote 客户机和 Sun SAM-Remote 服务器通过 TCP/IP 网络进行相互作用。各个 Sun SAM-Remote 客户机之间的网络可以是 Sun Solaris 操作环境支持的任何网络类型，如以太网、快速以太网或光纤通道等。

图 7-2 显示了 Sun SAM-Remote 客户机和 Sun SAM-Remote 服务器之间的相互作用。

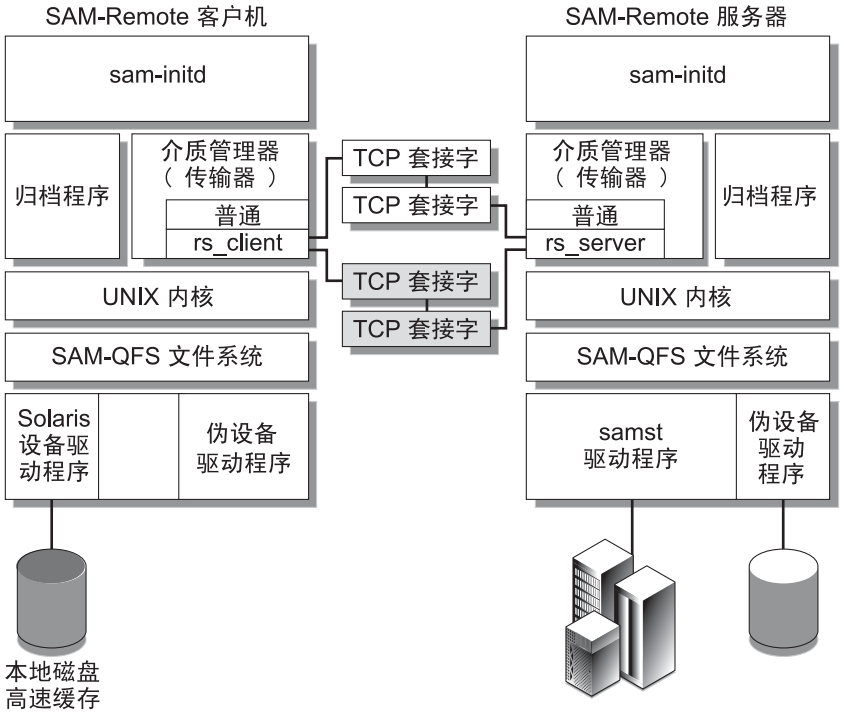

图 7-2 Sun SAM-Remote 服务器和客户机之间的交互作用

Sun SAM-Remote 服务器概述

Sun SAM-Remote 服务器不仅包括功能完备的 Sun StorEdge SAM-FS 存储管理主机，而且还包括用于定义各个客户机要共享的库的 Sun SAM-Remote 服务器守护进程。Sun SAM-Remote 服务器上必须至少配置一个 SAM-QFS 文件系统。

通过在服务器系统的 `/etc/opt/SUNWsamfs/mcf` 文件中添加 `ss` 设备类型行，您可将主机系统定义成为 Sun SAM-Remote 服务器。您必须为每台服务器提供唯一的系列集名称。每个守护进程最多可以配置 10 台客户机。要配置 10 台以上的客户机，请在 `mcf` 文件中为要配置的每 10 台客户机添加一条额外的远程服务器条目。有关服务器守护进程的更多信息，请参见 `sam-remote`(7) 手册页。

Sun SAM-Remote 客户机概述

Sun SAM-Remote 客户机是一个 Sun StorEdge SAM-FS 主机系统，该主机系统可用于建立包含多个伪设备的 Sun SAM-Remote 客户机守护进程。

通过在客户机系统的 `/etc/opt/SUNWsamfs/mcf` 文件中添加 `sc` 设备类型行，您可将主机系统定义成为 Sun SAM-Remote 客户机。有关客户机守护进程的更多信息，请参见 `sam-remote`(7) 手册页。

伪设备定义与 Sun SAM-Remote 服务器上的实际可移除介质设备相关的网络连接路径。伪设备的设备类型为 `rd`，即 *remote device*（**远程设备**）的缩写。您可以在 Sun SAM-Remote 客户机的 `/etc/opt/SUNWsamfs/mcf` 文件中定义伪设备。 Sun SAM-Remote 守护进程和伪设备均与一台特定的服务器相关联。

Sun SAM-Remote 守护进程支持客户机的伪设备，并且不限制每台客户机的伪设备数量。客户机可以使用的实际伪设备数是可配置的。在确定每台客户机可用的伪设备数目时，您可以将客户机和服务器之间同时发生的数据传输流的个数作为伪设备的数目。定义多个伪设备时，将会增加网络的总通信量。身为系统管理员，您可以根据需要确定系统实际所需的伪设备数。

Sun SAM-Remote 服务器和 Sun SAM-Remote 客户机之间的相互作用

Sun SAM-Remote 服务器守护进程 `sam-serverd` 在端口 1000 上监听客户机。您可以在 Sun Solaris `/etc/services` 目录中使用服务名 `rmtsam` 配置一个不同的端口。当 Sun SAM-Remote 客户机连接至 Sun SAM-Remote 服务器时， `sam-serverd` 守护进程将在另一端口上建立连接，然后使用已定义的端口，将此端口的编号传送给该客户机。套接字的大小将传递给客户机。套接字大小是可配置的，有关更多的详细信息，请参见第 148 页 “配置 Sun SAM-Remote 软件”。

库目录

Sun SAM-Remote 库目录是 Sun SAM-Remote 服务器上的目录子集。客户机目录将实时进行更新。分配给 Sun SAM-Remote 客户机目录的插槽仅受 Sun SAM-Remote 服务器的控制。

初始化期间，系统会生成一个客户机目录，并根据 Sun SAM-Remote 服务器目录文件中的信息，将其传递给 Sun SAM-Remote 客户机。主机和客户机之间建立连接后，可供客户机使用的介质将被标为 “可用”。如果客户机和服务器之间的连接中断，则客户机上的介质会被标为 “不可用”。您可以通过 `samu`(1M) `v` 显示屏幕查看介质是否可用。客户机上 `samu`(1M) `v` 显示屏幕所显示的信息是服务器上 `v` 显示屏幕所显示信息的一部分。通常，您应通过 Sun SAM-Remote 服务器上的 `samu`(1M) `v` 显示屏幕访问介质目录。有关 Sun SAM-Remote 服务器客户机文件的更多信息，请参见第 148 页 “配置 Sun SAM-Remote 软件”。有关使用 `samu`(1M) 操作员实用程序的信息，请参见《Sun StorEdge QFS 配置和管理指南》。

对目录所做的更改将根据需要在主机之间传递。例如，当服务器目录的更改内容涉及与某台客户机相关的介质类型时，则这些更改内容将会传送给该客户机，同时会更新该客户机目录。

归档

Sun SAM-Remote 归档处理过程与 Sun StorEdge SAM-FS 归档处理过程相同。 Sun SAM-Remote 客户机发出的安装请求将添加到服务器的安装请求表中。然后，客户机等待服务器发出一则表示介质已安装的消息。介质可用时，即会开始归档。

配置 Sun SAM-Remote 软件

本节介绍如何对 Sun SAM-Remote 服务器和客户机软件进行初始配置。包括以下几节内容：

- 第 148 页 “配置示例”
- 第 149 页 “配置软件”

配置示例

图 7-3 显示了本章过程所用的配置示例。本章中的示例说明了如何配置一个名为 `chicago` 的 Sun SAM-Remote 服务器。

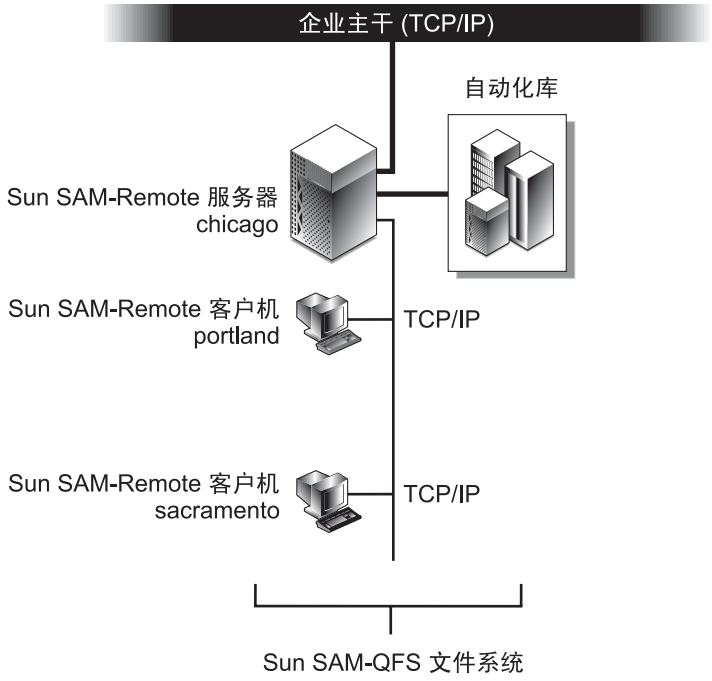

图 7-3 Sun SAM-Remote 配置示例

`portland` 和 `sacramento` 上的 Sun StorEdge SAM-FS 文件系统将 `chicago` 用作它们的 Sun SAM-Remote 服务器。

在本章的示例中， Sun StorEdge SAM-FS 文件系统将它们的某些归档副本写入由 `chicago` 控制的卡盒中。

配置软件

以下过程介绍了如何在 Sun SAM-Remote 服务器以及一台或多台 Sun SAM-Remote 客户机上配置 Sun SAM-Remote 软件。这些过程必须按所示的顺序执行，具体如下：

1. 第 150 页 “登录至潜在的服务器及客户机主机”
2. 第 150 页 “检验客户机和服务器配置”
3. 第 151 页 “编辑 `mcf` 文件”
4. 第 154 页 “定义 Sun SAM-Remote 客户机”
5. 第 154 页 “在服务器的 `mcf` 文件中定义 Sun SAM-Remote 服务器”
6. 第 155 页 “创建 Sun SAM-Remote 服务器配置文件”

7. 第 158 页 “启用归档”

在以下步骤中，您需要登录至主机系统，验证现有的软件版本，并根据需要升级软件。

▼ 登录至潜在的服务器及客户机主机

您必须以超级用户身份登录至所有潜在的服务器或客户机主机。

1. **以超级用户身份登录至 Sun SAM-Remote 服务器。**

   对于您要在其中安装 Sun Sun SAM-Remote 软件的服务器系统，您必须具有超级用户访问权限。

2. **以超级用户身份登录 Sun SAM-Remote 客户机。**

   对于您要在其中安装 Sun SAM-Remote 软件的所有客户机系统，您必须具有超级用户访问权限。

▼ 检验客户机和服务器配置

以下步骤用于确保您在要配置为 Sun SAM-Remote 环境一部分的系统中安装了所需的软件版本。

1. **在要配置为 Sun SAM-Remote 客户机或服务器的所有主机上，输入 `pkginfo`(1M) 命令及其 `-l` 选项。**

   您必须在要配置为 Sun SAM-Remote 环境一部分的所有客户机和服务器主机上，安装发行版本和修订版本相同的 Sun StorEdge SAM-FS 软件。例如：

代码示例 7-1 使用 `pkginfo`(1)

```
portland# pkginfo -l SUNWsamfs
   PKGINST:  SUNWsamfs
      NAME:  Sun SAM-FS and Sun SAM-QFS software Solaris 2.8
  CATEGORY:  system
      ARCH:  sparc
   VERSION:  4.0.5,REV=5.8.2003.01.12
    VENDOR:  Sun Microsystems, Inc.
    PSTAMP:  boomerang-20020712183351
  INSTDATE:  Jan 20 2003 07:30
   HOTLINE:  Please contact your local service provider
    STATUS:  completely installed
     FILES:     489 installed pathnames
                 12 shared pathnames
                  1 linked files
                 51 directories
                179 executables
```

代码示例 **7-1** 使用 pkginfo(1) （续）

```
               35813 blocks used (approx)

portland#
```

2. **检查 pkginfo(1) 命令的输出。**

   根据代码示例 7-1 中的输出示例，您会发现服务器运行的软件版本是 4U0.5，因此该服务器所在环境中的其他任何系统也必须运行 4U0.5 版本的软件。

   该示例假定已正确配置 Sun StorEdge SAM-FS 环境，并可正常操作。

3. **在要配置为 Sun SAM-Remote 客户机或服务器的所有主机上，输入 showrev(1M) 命令及其 -p 选项。**

   您必须在要配置为 Sun SAM-Remote 环境一部分的所有客户机和服务器主机上，安装相同的修补程序集。例如：

代码示例 **7-2** 使用 showrev(1M)

```
portland# showrev -p | grep SUNWsamfs
Patch: 113546-07 Obsoletes:  Requires:  Incompatibles:  Packages:
SUNWsamfs
portland#
```

4. **检查 showrev(1M) 命令的输出。**

   根据代码示例 7-2 中的输出示例，您会发现服务器运行的修补程序版本是 113546-07，因此该服务器所在环境中的其他任何系统也必须运行 113546-07 版本的修补程序。

5. **对于要在此环境中配置的每个系统，重复步骤 1、步骤 2、步骤 3 和步骤 4。**

6. **（可选）根据需要升级软件。**

   如果 pkginfo(1) 命令输出的信息表明 Sun SAM-Remote 环境中的所有系统均运行相同版本的软件和修补程序，则不必执行本步骤。

   如果要配置为 Sun SAM-Remote 环境一部分的某些系统运行旧版本的软件或修补程序，请将所有系统升级为最新的软件版本。以代码示例 7-1 为例，如果您在系统中运行的 Sun StorEdge SAM-FS 版本早于版本 4U0.5，则您必须将该系统至少升级至版本 4U0.5。

   有关升级软件的信息，请参见 《Sun StorEdge SAM-FS 安装和升级指南》。

▼ 编辑 mcf 文件

1. **在 Sun SAM-Remote 服务器上，停止 Sun StorEdge SAM-FS 功能。**

   a. **执行带有 idle *eq* 选项的 samcmd(1M) 命令，以将 Sun StorEdge SAM-FS 软件控制下的可移除介质驱动器置于空闲状态。**

例如：

```
# samcmd idle eq
```

参数	定义
eq	所访问的可移除介质驱动器在 mcf 文件中定义的设备序号。

对于环境中的每一个可移除介质驱动器，均运行 samcmd(1M) 命令。有关 samcmd(1M) 命令的更多信息，请参见 samcmd(1M) 手册页。

也可以使用 samu(1M) 操作员实用程序将驱动器置于空闲状态。有关如何使用 samu(1M) 操作员实用程序的信息，请参见 《Sun StorEdge QFS 配置和管理指南》。

注 – 在运行 samd stop 命令之前， Sun StorEdge SAM-FS 环境中的驱动器应该处于空闲状态。此命令的目的是使归档程序、登台程序和其他进程结束当前的任务。此外，它还可以卸载卡盒，并将它们放入各自的存储插槽内。

b. 输入 samd(1M) 命令及其 stop 选项，停止 sam-initd 守护进程及其子进程。

```
# samd stop
```

samd(1M) 命令安装在 /opt/SUNWsamfs/sbin 中。

2. **在客户机上，使用 vi(1) 或其他编辑器编辑现有的 Sun StorEdge SAM-FS /etc/opt/SUNWsamfs/mcf 文件。**

此步骤的目的是将该主机定义为 Sun SAM-Remote 客户机。代码示例 7-3 显示了客户机 portland 上已编辑完成的 mcf 文件。此 mcf 文件定义了文件系统，并显示 Sun SAM-Remote 客户机 portland 已定义成为 Sun SAM-Remote 服务器 chicago 的客户机。

代码示例 7-3 客户机 portland 上的 mcf 文件

```
# mcf file on portland
#
# Sun StorEdge QFS file system
#
# Equipment             Eq   Eq  Family    Dev  Additional
# Identifier            Ord  Ty  Set        St  Parameters
# ==========            ===  ==  ======     ==  ==========
samfs1                    1  ms  samfs1     on
/dev/dsk/c1t1d0s0        10  md  samfs1     on  /dev/rdsk/c1t1d0s0
/dev/dsk/c1t2d0s0        12  md  samfs1     on  /dev/rdsk/c1t2d0s0
```

代码示例 7-3 客户机 `portland` 上的 `mcf` 文件 （续）

```
#
# Define Sun SAM-Remote Client portland to Sun SAM-Remote server chicago
#
/etc/opt/SUNWsamfs/rmt200 200  sc chicagoss on /var/opt/SUNWsamfs/catalog/tcat
/dev/samrd/rd0            201  rd chicagoss on
/dev/samrd/rd1            202  rd chicagoss on
```

客户机上的 `mcf` 文件条目包括 Sun SAM-Remote 客户机的单行条目，以及您要配置的每个设备的伪设备条目。这些条目遵循 `mcf`(4) 联机资料中定义的语法。

第一组条目定义 Sun StorEdge QFS 文件系统。

第二组条目将 Sun SAM-Remote 客户机 `portland` 定义成为 Sun SAM-Remote 服务器 `chicago` 的客户机。第一行定义 Sun SAM-Remote 服务器自身。这些字段如下：

- "Equipment Identifier" 字段是客户机配置文件的路径名，稍后将在第 154 页“定义 Sun SAM-Remote 客户机”中创建。本示例中，配置文件的路径名为 `/etc/opt/SUNWsamfs/rmt200`。
- "Equipment Ordinal" 字段中包含一个介于 1 < *equipment_ordinal* < 65535 之间的唯一整数。本示例中服务器的设备序号为 200。
- "Equipment Type" 字段包含由两个字母组成的缩写 `sc`，表示 Sun SAM-Remote 客户机。
- "Family Set" 字段 `chicagoss` 与服务器的系列集名称相同。这是要在该特定服务器上使用的守护进程的系列集名称。对于每台客户机， Sun SAM-Remote 服务器均可拥有一个服务器守护进程。
- "Device State" 字段指定为 `on`。
- "Additional Parameters" 字段为可选字段。如上所示，可以在此字段中指定目录文件的路径。

此 `mcf` 文件中的最后两个条目定义了 Sun SAM-Remote 伪设备。伪设备定义 Sun SAM-Remote 服务器上实际设备的网络连接。包括以下条目：

- "Equipment Identifier" 字段是伪设备所用的 `/dev/samrd/rd*` 条目的路径名。这些条目在系统重新引导时创建。您可以定义不限数量的伪设备。
- "Equipment Type" 字段是伪设备的两字母缩写 `rd`。
- "Family Set" 字段 `chicagoss` 与客户机条目的系列集名称相同。

3. **（可选）在其他客户机上，使用 `vi`(1) 或其他编辑器编辑已有的 Sun StorEdge SAM-FS `/etc/opt/SUNWsamfs/mcf` 文件。**

   如果您还有其他客户机，则必须为其他每一个 Sun SAM-Remote 客户机完成此步骤。请执行步骤 2 中所述的相同过程。

在本章的示例中，您应为客户机 sacramento 完成相同的配置过程。编辑该主机系统的 mcf 文件，并将 portland 的 mcf 文件中的最后一组条目行复制到 sacramento 的 mcf 文件。这几行用于将 sacramento 主机系统定义成为 chicago 的 Sun SAM-Remote 客户机。

▼ 定义 Sun SAM-Remote 客户机

Sun SAM-Remote 客户机的配置文件中包含一个单行条目：Sun SAM-Remote 服务器的名称。如步骤 2 中第 151 页 “编辑 mcf 文件”中所示，此客户机配置文件的完整路径名在客户机的 mcf 文件中加以指定。

1. **在客户机上，使用 vi(1) 或其他编辑器打开 Sun SAM-Remote 客户机配置文件。**

   例如：

```
portland# vi /etc/opt/SUNWsamfs/rmt200
```

2. **在此文件中添加 Sun SAM-Remote 服务器的名称。**

   此步骤将生成一个单行文件。

   代码示例 7-4 显示了已为客户机 portland 编辑的配置文件。它指向名为 chicago 的 Sun SAM-Remote 服务器。

   代码示例 7-4 客户机配置文件

```
portland# cat /etc/opt/SUNWsamfs/rmt200
chicago
```

3. **对于每一个 Sun SAM-Remote 客户机，重复步骤 1 和步骤 2。**

   如果您有多台客户机，请在每一台客户机上创建客户机配置文件。

▼ 在服务器的 mcf 文件中定义 Sun SAM-Remote 服务器

此步骤用于在服务器的 mcf 文件中定义 Sun SAM-Remote 服务器。

- **在 Sun SAM-Remote 服务器上，使用 vi(1) 或其他编辑器编辑现有的 Sun StorEdge SAM-FS /etc/opt/SUNWsamfs/mcf 文件，以将系统定义成为 Sun SAM-Remote 服务器。**

  本步骤示例中，将编辑服务器 chicago 上的 mcf 文件。编辑后的 mcf 文件中定义了一个 Sun StorEdge QFS 文件系统，并将 chicago 定义为 Sun SAM-Remote 服务器。

代码示例 7-5 显示了 chicago 上的 mcf 文件。

代码示例 7-5 chicago 上的 mcf 文件

```
# mcf file on Sun SAM-Remote server chicago:
# Eq Identifier Eq Ord  Eq Typ Fam Set Dev St  Addl Params
#
samfs1                1   ms    samfs1  on
/dev/dsk/c2t6d0s0   11   md    samfs1  on  /dev/rdsk/c2t6d0s0
/dev/dsk/c2t6d0s1   12   md    samfs1  on  /dev/rdsk/c2t6d0s1
#
# define a tape library that client portland can use:
/dev/samst/c0t3u0  100   rb    rb100   on  /var/opt/SUNWsamfs/catalog/rb100.cat
/dev/rmt/0cbn      101   tp    rb100   on
/dev/rmt/1cbn      102   tp    rb100   on

# Define Sun SAM-Remote server chicago
#
/etc/opt/SUNWsamfs/rmt200 50  ss      chicagoss  on
```

这些条目遵循 mcf(4) 中定义的语法。在本示例文件中，它们的含义如下：

- "Equipment Identifier" 字段是您在以下的过程中配置的服务器配置文件的路径名。本示例中，该文件名为 /etc/opt/SUNWsamfs/rmt200。
- "Equipment Ordinal" 字段包含一个介于 1 ≤ *equipment_ordinal* ≤ 65535 之间的唯一整数。本示例中服务器的 equipment ordinal 为 50。
- "Equipment Type" 字段包含由两个字母组成的缩写 ss，表示 Sun SAM-Remote 服务器。
- "Family Set" 字段 chicagoss 与客户机 mcf 文件中所用的系列集名称相同。请注意， Sun SAM-Remote 服务器可以定义多个服务器守护进程。
- "Device State" 字段 （可选）在本示例中指定为 on。
- "Additional Parameters" 字段为可选字段。

注 – 您必须在该 Sun SAM-Remote 服务器的 mcf 文件中至少配置一个 Sun StorEdge SAM-FS 文件系统。

▼ 创建 Sun SAM-Remote 服务器配置文件

Sun SAM-Remote 服务器配置文件用于定义每一台客户机将要使用的磁盘缓冲区特性和介质。每个服务器守护进程可以配置十台客户机。如果您想支持更多台客户机，则必须配置其他 Sun SAM-Remote 服务器守护进程，如前面第 151 页 “编辑 mcf 文件”（步骤 2）中及第 154 页 “定义 Sun SAM-Remote 客户机”中所述。

1. **在服务器上，使用 vi(1) 或其他编辑器打开 Sun SAM-Remote 服务器配置文件。**

2. **编写服务器配置文件。**

代码示例 7-6 显示了服务器配置文件示例， `/etc/opt/SUNWsamfs/rmt200`。此文件位于 Sun SAM-Remote 服务器 `chicago` 上，其中定义了客户机 `portland` 和 `sacramento`。

代码示例 7-6 服务器配置文件 `rmt200`

```
#
# Sun SAM-Remote server config file /etc/opt/SUNWsamfs/rmt200
#
portland
     media
     100  at  (000031|000032|000034|000035|000037|000038)
     endmedia
#
sacramento
     media
     100 at  (000131|000132|000134|000135|000137|000138)
     endmedia
```

如代码示例 7-6 所示，服务器配置文件中包含每台客户机的多行条目。井符号 (#) 表示注释行。注释行右侧的所有内容均被忽略。

代码示例 7-7 显示了 Sun SAM-Remote 服务器配置文件的格式。

代码示例 7-7 服务器配置文件格式

```
client_name
      [ parameter1 ]
      media
            eq  media_type  regex
            [ eq  media_type  regex ]
            [. . .]
      endmedia
```

以下步骤介绍如何编写服务器配置文件。

a. 编写 *client_name* 字段。

client_name 定义每台客户机供 Sun SAM-Remote 守护进程调用的网络名称。*client_name* 中的第一个字符必须是该行中的首字符。*client_name* 可以是客户机的网络名称、 IP 地址或完整域名。

client_name 之后的 *parameter* （如果指定）和介质规范 （直至下一个客户机定义之前），均专用于该客户机。*parameter* 和 *media* 定义必须缩进一个空格或制表符。

b.（可选）编写 *parameter* 字段。

参数行采用 *keyword* = *value* 赋值对表示。您可以使用 *parameter* 字段指定网络块大小。net_block_size 参数指定此客户机的套接字所用的网络块大小 (KB)。此参数的格式如下：

```
net_blk_size=size
```

其中的 *size*，用于指定 4 ≤ *size* ≤ 64 范围内的一个整数。默认值为 4，即 4096 字节。

parameter 行必须缩进一个空格或制表符。

c. 编写 media 和 endmedia 关键字字段。

media 和 endmedia 关键字是服务器配置文件中必需的关键字。它们定义客户机可以使用的介质归档卷。这些介质关联按以下方式指定：

代码示例 7-8 服务器配置文件中的介质规范

```
media
      eq  media_type  (regex)
      [ eq  media_type  (regex) ]
      [. . .]
endmedia
```

media 和 endmedia 关键字界定了 Sun SAM-Remote 服务器配置文件的介质定义区域。*eq media_type regex* 行是介质定义行。*media* 定义必须缩进一个空格或制表符。*regex* 数据必须置于圆括号内。

介质类型规范包括以下几个要素：

参数	定义
eq	库的设备序数。 具有不同介质的网络连接库可以拥有多个 *eq media_type regex* 行，以便为每一种介质类型指定不同的 *eq media_type regex* 行。
media_type	由两个字符表示的特定介质类型。请注意，一般介质类型规范在 mcf 文件中有效，但不适用于 *media_type* 规范。规范必须适用于特定的介质类型 （例如 lt）。有关有效介质类型的信息，请参见 mcf(4) 手册页。 如果您的网络连接库拥有多个介质类型，请指定多个介质定义行。

参数	定义

regex	用于归档文件的卡盒的卷序列名 (Volume Serial Names, VSN)。每一个指定的 VSN 必须表示为扩展的正则表达式，并且 VSN 必须置于圆括号内。有关扩展正则表达式的信息，请参见 egrep(1) 手册页。
	您可以为每个 *media_type* 指定多个介质定义行，因此能够灵活地定义介质。例如，以下是有效的介质类型定义：
	`media`
	`100 lt (VSN1)`
	`100 lt (VSN2)`
	`endmedia`
	有关正则表达式的信息，请参见 regcomp(3C) 手册页。

注 – 请勿让多台客户机使用同一个物理介质卡盒。此外，如果 Sun SAM-Remote 服务器在 Sun SAM-Remote 环境之外拥有自身的文件系统，则不建议让客户机和服务器共用介质卡盒。

▼ 启用归档

以下步骤用于启用归档功能并完成配置过程。

1. **检验客户机上的 `archiver.cmd` 文件。**

   您可能需要执行以下任务，具体取决于您的配置：

   - 确保服务器配置文件中定义的 VSN 已分配给 `archiver.cmd` 文件中的正确归档集。
   - 如果以下指令所应用的归档集将归档至 Sun SAM-Remote 服务器连接的库，则应从 Sun SAM-Remote 客户机上的 `archiver.cmd` 文件中删除这些指令：
     - `-tapenonstop`
     - `-offline_copy direct`

2. **执行带有 `start` 选项的 `samd`(1M) 命令，启动服务器和客户机上的 Sun StorEdge SAM-FS 进程。**

   为确保系统读取服务器和客户机上的新配置文件，您必须启动或重新启动 Sun StorEdge SAM-FS 软件。

   在客户机和服务器上输入以下命令：

```
server# samd start
```

   有关启动及重新启动 Sun StorEdge SAM-FS 更多的完整指导，请参见《Sun StorEdge SAM-FS 安装和升级指南》。

3. **在服务器或客户机上激活 samu(1M)。**

   此步骤的目的是验证主机之间的连接情况。使用 samu(1M) 实用程序的 s 和 R 显示屏幕显示 Sun SAM-Remote 的连接状态。有关 samu(1M) 的更多信息，请参见 samu(1M) 手册页或《Sun StorEdge QFS 配置和管理指南》。

   代码示例 7-9 显示了 Sun SAM-Remote 客户机 portland 上的 samu(1M) 状态 s 显示屏幕。请注意，设备类型 sc 表示 Sun SAM-Remote 客户机。该行以下的消息表示已建立与服务器 chicago 的连接。

代码示例 7-9 客户机 samu(1M) s 显示屏幕

```
Device status                  samu   4.0.5 Wed May 02 14:44:44
License: License never expires.

ty     eq state   device_name                          fs status   pos
ms      1 on      samfs1                                1 m---------

md     10 on      /dev/dsk/c1t1d0s0                     1 ----------

md     12 on      /dev/dsk/c1t2d0s0                     1 ----------

s9     35 on      /dev/samst/c0t5u0                    35 m--------r
        move complete
lt     36 on      /dev/rmt/0cbn                        35 ---------p
        empty
lt     37 on      /dev/rmt/1cbn                        35 ---------p
        empty
lt     38 on      /dev/rmt/2cbn                        35 --l------r
        idle
lt     39 on      /dev/rmt/3cbn                        35 --l------r
        idle
sc    200 on      /etc/opt/SUNWsamfs/rmt200           200 ---------r
        server chicago connected
rd    201 on      /dev/samrd/rd0                      200 ---------r

rd    202 on      /dev/samrd/rd1                      200 ---------r

hy    203 on      historian                           203 ----------
```

代码示例 7-10 显示了 Sun SAM-Remote 服务器 chicago 上的 samu(1M) 状态 s 显示屏幕。请注意，设备类型 ss 表示 Sun SAM-Remote 服务器。此显示屏幕上的信息表明该系统是一个 Sun SAM-Remote 服务器。

代码示例 7-10 chicago 上的服务器 samu(1M) s 显示屏幕

```
Device status                    samu   4.0.5 Tue Apr 24 14:49:43
License: License never expires.

ty     eq state   device_name                          fs status   pos
ms      1 on      samfs1                                1 m---------

md     11 on      /dev/dsk/c2t6d0s0                     1 ----------

md     12 on      /dev/dsk/c2t6d0s1                     1 ----------

ss     50 on      /etc/opt/SUNWsamfs/rmt200            50 ---------r

sl    100 on      /dev/samst/c0t3u0                   100 m--------r

at    101 on      /dev/rmt/0cbn                       100 ---------p
        initializing
at    102 on      /dev/rmt/1cbn                       100 ---------p
        initializing
hy    103 on      historian                           103 ----------
```

代码示例 7-11 显示了 Sun SAM-Remote 服务器 chicago 上的 samu(1M) Sun SAM-Remote R 显示屏幕。

代码示例 7-11 chicago 上的服务器 samu(1M) R 显示屏幕

```
Remote server eq: 50             addr: 00001ca0 4.0.5 Wed May 02
14:55:37

message:

Client: portland
client index - 0
network block size - 4096
max file size - 0                        flags  - c0000000
min file size - 8
```

如果您具有多个 Sun SAM-Remote 客户机，则可以按 CONTROL-f 组合键来滚动查看客户机。

在代码示例 7-11 中，连接的客户机名为 portland。"client index" 字段表示此客户机的索引号为 0（为此服务器守护进程定义的客户机索引号可以是 0 到 9 之间的数字）。最大文件大小、最小文件大小及网络块大小的单位均为 “字节”。“标志”用于表示连接的状态，如下所示：

表 7-1 samu(1M) R 显示屏幕标志

标志	含义
0x00000000	没有连接。
0xc0000000	已建立连接。

4. 从服务器中运行 samu(1M) 实用程序，确保在客户机上可以使用目录。

您应该可以查看每一台客户机可用的 Sun SAM-Remote 目录，方法是使用 samu(1M) 实用程序的 v 显示屏幕来显示 VSN。在 samu(1M) 中，输入以下命令：

```
:v eq
```

eq 必须是 Sun SAM-Remote 客户机守护进程的设备序号 （与在 mcf 文件中定义的设备序号相同）。

代码示例 7-12 显示了服务器 chicago 上的 samu(1M) 显示屏幕。此显示屏幕是通过在服务器 chicago 上指定 :v 200 获得的。它显示了客户机 portland 可在服务器 chicago 上访问的卷。

代码示例 7-12 从 chicago 上查看时可用的卷

```
Robot VSN catalog by slot : eq 200 samu 4.0.5 Wed May 02 15:24:13
count 32
slot      access       time   count use flags          ty vsn

   1      2003/01/02   10:40      0   0% -il-o-b-R-U-   at 000032
   2      2003/01/02   11:41      0   0% -il-o-b-R---   at 000034
   3      2003/01/02   12:42    170  91% -il-o-b-----   at 000035
   4      2003/01/02   13:43     20   7% -il-o-b-----   at 000037
   5      2003/01/02   14:44      0   0% -il-o-b-----   at 000038
   6      2003/01/02   13:41      0   0% -il-o-b-----   at 000031
```

5. 在客户机上，输入 archiver(1M) 命令及其 -A 选项。

本步骤用于检验是否可将文件从客户机归档到服务器。您可以使用 archiver(1M) 命令及其 -A 选项进行检验。此选项将列出归档程序所要写入至的存储设备列表，其中包括服务器上的 VSN。有关此命令的信息，请参见 archiver(1M) 手册页。

如果未归档文件，请参见 《Sun StorEdge SAM-FS 故障排除指南》，获取有关排除归档程序故障的说明。

使用 Sun SAM-Remote 软件回收

本节介绍有关在 Sun SAM-Remote 环境中执行回收过程的信息。Sun Microsystems 建议您仅在出现本章所述的特殊情况时才有必要在 Sun SAM-Remote 环境中执行回收过程。请严格遵守本章所述的回收限制条件，否则可能导致数据丢失。 Sun StorEdge SAM-FS 软件中不存在这些强制性的限制条件。

由于回收过程包括释放卡盒上的空间以便储存更多的数据，因此如果回收过程没有正确配置，则回收程序可能会破坏归档卡盒上的重要数据。

注意 – 在 Sun SAM-Remote 环境中运行回收程序时，必须完全理解回收程序的每一个步骤。如果执行命令的顺序不正确，或在错误的系统上执行了回收命令，则可能导致无法挽回的数据丢失。确保在执行任何命令之前，已对该命令的操作进行了分析，如 `tplabel`(1M) 可能会删除 Sun SAM-Remote 客户机或 Sun SAM-Remote 服务器上的数据。

不要在 Sun SAM-Remote 服务器和 Sun SAM-Remote 客户机上重叠执行回收操作，这一点非常重要。这样的操作可能会导致意外的重新标记卡盒以及无法挽回的数据丢失。

不要回收包含可移除介质文件的卡盒。

在 Sun SAM-Remot 客户机和服务器环境中，客户机和服务器并不清楚对方的文件系统、数据文件以及 inode 文件。服务器和客户机各自必须专用一组特定的卡盒，而不能相互使用对方的卡盒。您可以通过在 Sun SAM-Remote 服务器的 `/etc/opt/SUNWsamfs/recycler.cmd` 文件中创建 `no_recycle` 列表，以防止意外回收 Sun SAM-Remote 客户机所用的 VSN。不过，在 `no_recycle` 列表中的卷上运行 `chmed`(1M) 命令的 `+c` 选项时，应多加小心。当您运行此命令为某个卷设置回收标志 (`+c`) 时，此操作将覆盖 `no_recycle` 列表，该列表位于 `/etc/opt/SUNWsamfs/recycler.cmd` 文件中。

请勿尝试在同一天既回收 Sun SAM-Remote 服务器上的卷，又回收 Sun SAM-Remote 客户机上的卷。

只有出现以下情况时，才可以在 Sun SAM-Remote 环境中执行回收过程：

- 系统中的每一个 VSN 由不同的客户机系统使用，或由服务器使用。任何 VSN 中的文件只能来自一个系统。

- Sun SAM-Remote 客户机只具有那些包含该客户机归档映像的 VSN 的目录条目。除这些 VSN 的目录条目之外，不具有其他任何 VSN 的目录条目。服务器配置文件中的介质定义行 （即 *eq media_type regex* 行）中的 *regex* 必须与客户机目录中指定的卷保持一致。此外，客户机目录中的 re *regex* 规范不能指定相同的卷。
- 系统按归档集执行归档过程。因此，当使用 Sun SAM-Remote 时，只能按归档集（而不是库）执行回收过程。

本章介绍了两种通过 Sun SAM-Remote 客户机和服务器启用回收过程的方法。方法如下：

- 第 163 页 “在 Sun SAM-Remote 环境中进行回收 — 方法 1”
- 第 186 页 “在 Sun SAM-Remote 环境中进行回收 — 方法 2”

在 Sun SAM-Remote 环境中进行回收 — 方法 1

本节介绍了一种在 Sun SAM-Remote 环境中启用回收过程的方法。本节采用的环境示例包括一台名为 sky 的服务器和一台名为 zeke 的客户机。此过程说明了如何配置 Sun SAM-Remote 环境以便在两种不同库的卡盒上创建文件的归档副本。归档副本 1 将写入 zeke 本机上的 StorageTek 库。归档副本 2 将远程写入与 sky 连接的 ADIC 库。以下几节介绍了这两个系统的相关文件。

注意 – 只有在严格执行本过程中的步骤，且您的配置经测试可以正确执行回收过程之后，才可以在 Sun SAM-Remote 环境中运行回收程序。

服务器 sky 的配置文件

服务器的 mcf 文件和服务器配置文件中必须具有 Sun SAM-Remote 的配置信息。以下代码示例显示了这些文件。

代码示例 7-13 显示了服务器 sky 上的 mcf 文件。

代码示例 7-13 服务器 sky 上的 mcf 文件

```
# This is the mcf file for the server (sky).
# The server parameters file (rmt1000) points
#    back to the correct automated library's equipment number
#    (70) for the ADIC Scalar 1000.
#
samfs1                100   ma   samfs1   on
/dev/dsk/c0t0d0s5     110   mm   samfs1   on   /dev/rdsk/c0t0d0s5
/dev/dsk/c3t2d0s3     120   mr   samfs1   on   /dev/rdsk/c3t2d0s3
/dev/dsk/c3t2d0s4     121   mr   samfs1   on   /dev/rdsk/c3t2d0s4
```

代码示例 7-13 服务器 sky 上的 mcf 文件 （续）

```
samfs2                139   ma   samfs2   on
/dev/dsk/c3t4d0s3     140   mm   samfs2   on   /dev/rdsk/c3t4d0s3
/dev/dsk/c3t4d0s4     141   mr   samfs2   on   /dev/rdsk/c3t4d0s4

# ADIC Scalar 1000
/dev/samst/c0t0u0 70 rb adic1 - /var/opt/SUNWsamfs/catalog/adic1
/dev/rmt/0bn          71    at   adic1    on
/dev/rmt/1bn          72    at   adic1    on
/dev/rmt/2bn          73    at   adic1    on
/dev/rmt/3bn          74    at   adic1    on
/dev/rmt/4bn          75    at   adic1    on
/dev/rmt/5bn          76    at   adic1    on
/dev/rmt/11bn         77    at   adic1    on
/dev/rmt/10bn         78    at   adic1    on
/dev/rmt/9bn          79    at   adic1    on
/dev/rmt/8bn          80    at   adic1    on
/dev/rmt/7bn          81    at   adic1    on
/dev/rmt/6bn          82    at   adic1    on

# Define Sun SAM-Remote server skyrs
/etc/opt/SUNWsamfs/rmt1000   1000   ss   skyrs   on
```

代码示例 7-14 显示了服务器 sky 上的服务器配置文件。

代码示例 7-14 服务器 sky 上的服务器配置文件

```
# Server configuration file /etc/opt/SUNWsamfs/rmt1000 on sky.
# The eq of the automated library MUST match the eq of the
#   automated library that you want to use in the mcf file.

zeke
   media
   70 at 00002[0-9]
   endmedia
```

客户机 zeke 的配置文件

客户机的 mcf 文件和客户机配置文件中必须具有 Sun SAM-Remote 的配置信息。以下代码示例显示了这些文件。

代码示例 7-15 显示了客户机 zeke 上的 mcf 文件。

代码示例 7-15 客户机 zeke 上的 mcf 文件

```
# mcf file for client (zeke)
#
samfs1                    10  ms  samfs1   on
/dev/dsk/c1t3d0s0         11  md  samfs1   on   /dev/rdsk/c1t3d0s0
/dev/dsk/c1t3d0s1         12  md  samfs1   on   /dev/rdsk/c1t3d0s1
/dev/dsk/c1t3d0s3         13  md  samfs1   on   /dev/rdsk/c1t3d0s3

# Define a StorageTek L20 with 1 drive and 20 slots (including cap)
/dev/samst/c0t2u0         50  rb  stk_l20  on /var/opt/SUNWsamfs/catalog/L20_cat
/dev/rmt/0hbn             51  lt  stk_l20  on

# Define zeke as a Sun SAM-Remote client using sky as the server
/etc/opt/SUNWsamfs/sky 200  sc  skyrs    on  /var/opt/SUNWsamfs/catalog/sky_cat
/dev/samrd/rd0         201  rd  skyrs    on
/dev/samrd/rd1         202  rd  skyrs    on
/dev/samrd/rd2         203  rd  skyrs    on
/dev/samrd/rd3         204  rd  skyrs    on
```

代码示例 7-16 显示了客户机 zeke 上的客户机配置文件。

代码示例 7-16 客户机 zeke 上的客户机配置文件

```
# cat /etc/opt/SUNWsamfs/sky
# File /etc/opt/SUNWsamfs/sky on Sun SAM-Remote client zeke:
sky
```

▼ 配置回收过程 — 方法 1

以下的步骤介绍了如何配置回收过程。它包括一个归档和回收的测试，以确保将来能够正确执行回收过程。由于测试周期各不相同，本过程可能要一天或两天才能完成，具体取决于文件归档和回收的频率。

注 – 请勿在服务器上使用 chmed(1M) 命令为客户机的 VSN 设置回收标志 (+c)。该操作会改写服务器上 /etc/opt/SUNWsamfs/recycler.cmd 文件中的 no_recycle 列表。

---

1. **参见第 131 页 “回收”中有关回收程序的章节。**

   在 Sun SAM-Remote 环境中运行回收程序时，您必须完全理解回收过程的每一步骤。如果您不熟悉回收过程，请抽出一段时间进行学习。

2. **确保 Sun SAM-Remote 客户机和服务器配置正确，且可以进行归档。**

   有关配置和验证 Sun SAM-Remote 环境的更多信息，请参见第 148 页 “配置 Sun SAM-Remote 软件”，该节介绍了有关配置 Sun SAM-Remote 客户机和服务器的更多信息。本过程包括用于确保归档的步骤。

3. **打开客户机系统上的 `archiver.cmd` 文件，添加回收指令。**

   本示例中，将按归档集（而不是库）来执行回收过程。因此，指定按归档集进行回收的指令必须出现在 `archiver.cmd` 文件中。

   代码示例 7-17 显示了客户机 `zeke` 上的 `archiver.cmd` 文件。为了与回收程序进行通信，此文件已经过编辑。

代码示例 7-17 客户机 `zeke` 上的 `archiver.cmd` 文件

```
# This is file /etc/opt/SUNWsamfs/archiver.cmd
#   on Sun SAM-Remote client zeke.
#
# wait

logfile = /var/opt/SUNWsamfs/archiver/archiver.log
trace = /var/opt/SUNWsamfs/trace/archiver all

interval = 1m

no_archive tmp
no_archive .

archmax = lt 2G
archmax = at 5G

drives = skyrs 4 # use up to four drives for remote archiving.

fs = samfs1
     1 4h
archiveset testdir0
     1 1m
     2 1m
defaultset .
     1 1m
     2 1m

params
```

代码示例 7-17 客户机 zeke 上的 archiver.cmd 文件 （续）

```
# Start with mingain high to reduce workload.
# If you need more recycling, reduce mingain.
# If too much recycling, increase High Water Mark.
archiveset.1 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
archiveset.1 -recycle_ignore
defaultset.1 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
defaultset.1 -recycle_ignore

# Remote directives.
# Use up to three drives per archive set.
# Load will split to two drives at 100m, to three drives at 150m.
archiveset.2 -drives 3 -drivemin 50m
defaultset.2 -drives 3 -drivemin 50m

# Remote directives.
# Start with mingain high to reduce workload.
# If you need more recycling, reduce mingain.
# If too much recycling, increase High Water Mark.
archiveset.2 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
archiveset.2 -recycle_ignore
defaultset.2 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
defaultset.2 -recycle_ignore
endparams

vsns
samfs1.1     lt 000173      # local copy.
archiveset.1 lt ^CEL        # local copy.
archiveset.2 at 00002[0-4]  # remote copy, sky ait-2
                            # tapes 20 through 24.
defaultset.1 lt ^CSM        # local copy.
defaultset.2 at 00002[5-9]  # remote copy, sky ait-2
                            # tapes 25 through 29.
endvsns
```

代码示例 7-17 中显示的指令用于执行以下操作：

- -recycle_hwm 指令用于设置归档集在库中的上限。当 VSN 的利用率超出此指令设置的百分比时，系统会开始回收归档集。
- -recycle_ignore 指令只是临时插入。此指令可在您配置并检测环境之前，防止进行回收过程。您可在以后的步骤中删除此指令。
- -recycle_mingain 指令设置为高，以限制需要重新获取空间的工作数量。也就是说，此指令设置为高以保证工作效率。
- -recycle_vsncount 1 指令用于防止回收过程造成系统崩溃。此指令指定回收程序每次清除一个 VSN 中的数据。清除第一个 VSN 中的数据之后，随后开始清除第二个。因此在任何时刻，队列中都有一个 VSN 被重新标记，一个 VSN 被清除。

4. **打开客户机上的 `recycler.cmd` 文件，指定用于接收回收日志输出的日志文件。**

   以下是客户机 `zeke` 上经过编辑的 `recycler.cmd` 文件，其中指定了回收程序日志文件：

代码示例 7-18 客户机 `zeke` 上的 `recycler.cmd` 文件

```
#
# This is the /etc/opt/SUNWsamfs/recycler.cmd file
# on client zeke.
#
logfile = /var/opt/SUNWsamfs/log/recycler
```

5. **检验服务器上的 `archiver.cmd` 文件是否指定按归档集执行回收过程。**

   使用 Sun SAM-Remote 时，必须指定按归档集（而不是库）执行回收过程。因此，指定按归档集进行回收的指令必须出现在 `archiver.cmd` 文件中。

   代码示例 7-19 显示了服务器 `sky` 上的 `archiver.cmd` 文件。此文件指定按归档集执行归档过程。

代码示例 7-19 服务器 `sky` 上的 `archiver.cmd` 文件

```
# This is the archiver.cmd for the server (sky).
#
# Number of drives: 10
# Number of Mounted Filesystems: 1
# Number of Tests per Filesystem: 1
# Number of Archive Copies per Test: 2

#wait
#trace = /var/opt/SUNWsamfs/trace/archiver all

logfile = /var/opt/SUNWsamfs/log/archiver
interval = 1m
no_archive .
archmax = at 5G
drives = adic1 6

fs = samfs1
     1 4h
testset testdir0
     1 1m
     2 1m
allsam1   .
     1 1m
```

代码示例 7-19 服务器 sky 上的 archiver.cmd 文件 （续）

```
      2 1m

params
allsam1.1 -drives 4 -drivemin 50m
allsam1.1 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
allsam1.1 -recycle_ignore
allsam1.2 -drives 4 -drivemin 50m
allsam1.2 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
allsam1.2 -recycle_ignore
testset.1 -drives 4 -drivemin 50m
testset.1 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
testset.1 -recycle_ignore
testset.2 -drives 4 -drivemin 50m
testset.2 -recycle_hwm 60 -recycle_mingain 90 -recycle_vsncount 1
testset.2 -recycle_ignore
endparams

vsns
samfs1.1 at 000000
allsam1.1 at 00000[1-5]    # vsns 1 through 5.
allsam1.2 at 00000[6-9]    # vsns 6 through 9.
testset.1 at 00001[0,4]    # vsns 10 and 14.
testset.2 at 00001[5,9]    # vsns 15 and 19.
endvsns
```

6. **编辑服务器上的** recycler.cmd **文件**。

   使用编辑器修改此文件，以指定以下各项：

   - 用于接收回收程序输出的回收程序日志文件。
   - 用于保护 Sun SAM-Remote 客户机所用 VSN 的 no_recycle 指令。根据配置要求，Sun SAM-Remote 客户机将其归档副本 2 写入至 Sun SAM-Remote 服务器库中的卡盒。您需要输入 no_recycle 指令，以防止 Sun SAM-Remote 服务器回收 Sun SAM-Remote 客户机用以归档的 VSN。

   以下是服务器 sky 上经过编辑的 recycler.cmd 文件，其中指定了回收程序日志文件：

代码示例 7-20 服务器 sky 上的 recycler.cmd 文件

```
#
# This is the /etc/opt/SUNWsamfs/recycler.cmd file
# on Sun SAM-Remote server sky.
#
logfile = /var/opt/SUNWsamfs/recycler/recycler.log
```

代码示例 **7-20** 服务器 sky 上的 recycler.cmd 文件 （续）

```
adic1 -ignore
no_recycle at 00002[0-9] # Prevents VSNs assigned to zeke from
                         # being recycled.
```

7. 在 Sun SAM-Remote 客户机上，使用 sam-recycler(1M) 命令测试回收程序。

在 Sun SAM-Remote 客户机系统上运行回收程序。此测试用于查看回收程序是否可以正确识别配置文件中指定的设备和 VSN。此项测试非常重要，这是因为：如果回收程序检测到它所运行于的系统在其目录 （包括历史记录目录）中列出的特定 VSN 上没有归档映像， recycler.sh 脚本可以调用要标记的卡盒。标记卡盒将会破坏卡盒上的所有数据。 Sun SAM-Remote 客户机和 Sun StorEdge SAM-FS 服务器之间不存在通信，无法告知对方是否具有归档副本。从本地的 Sun StorEdge SAM-FS 文件系统在本地提供所有的此类信息。

例如，您可以使用以下命令初次测试回收程序的操作：

```
zeke# sam-recycler -dvx
```

回收程序将会运行并将其活动记录至回收程序日志文件中。回收程序日志文件在 recycler.cmd 文件中定义。有关 sam-recycler(1M) 命令的更多信息，请参见 sam-recycler(1M) 手册页。

8. 检查回收程序日志文件。

您要查找的是以下消息：

```
Recycling is ignored on this archive set.
```

代码示例 7-21 显示了日志文件范例。

代码示例 **7-21** 客户机 zeke 上的回收程序日志文件

```
# recycler.log from client zeke.

========== Recycler begins at Mon Jun  4 09:49:41 2001 ===========
Initial 7 catalogs:

0  Family: stk_l20              Path: /var/opt/SUNWsamfs/catalog/L20_cat
   Vendor: STK                  Product: L20
   SLOT                 ty    capacity         space vsn
      0                 lt        33.0G        33.0G 000173
      1                 lt        32.8G        44.1M CEL170
      2                 lt        33.0G        33.0G CEL139
      4                 lt        32.8G        16.8G CFC504
```

代码示例 7-21 客户机 zeke 上的回收程序日志文件 （续）

```
      5                   lt        33.0G        33.0G CFC503
      6                   lt        32.9G         0    CSM689
      7                   lt        32.9G        19.6G CSM690
      8                   lt        33.0G        33.0G CSM691
      9                   lt        33.0G        33.0G CSM692
     10                   lt        10.0G        10.0G CLN018
     11                   lt        33.0G        33.0G 000766
   Total Capacity:  339.2G bytes, Total Space Available: 244.3G bytes
   Volume utilization 27%, high 95% VSN_min 50%
   Recycling is ignored on this robot.



1  Family: skyrs                Path: /var/opt/SUNWsamfs/catalog/sky_cat
   Vendor: (NULL)               Product: (NULL)
   SLOT                  ty    capacity         space vsn
      0                  at        48.5G        23.3G 000020
      1                  at        23.8G        23.8G 000021
      2                  at        48.5G        48.5G 000022
      3                  at        48.5G        48.5G 000023
      4                  at        48.5G        48.5G 000024
      5                  at        48.5G         2.6G 000025
      6                  at        48.5G       361.4k 000026
      7                  at        48.5G        48.5G 000027
      8                  at        48.5G        48.5G 000028
      9                  at        48.5G         0    000029
   Total Capacity:  460.8G bytes, Total Space Available: 292.5G bytes
   Volume utilization 36%, high 95% VSN_min 50%
   Recycling is ignored on this robot.



2  Family: hy                   Path: /var/opt/SUNWsamfs/catalog/historian
   Vendor: Sun SAM-FS               Product: Historian
   SLOT                  ty    capacity         space vsn
      (no VSNs in this media changer)
   Total Capacity:  0    bytes, Total Space Available: 0    bytes
   Volume utilization 0%, high 95% VSN_min 50%
   Recycling is ignored on this robot.



3  Family: defaultset.1         Path: /etc/opt/SUNWsamfs/archiver.cmd
   Vendor: Sun SAM-FS               Product: Archive set
```

代码示例 **7-21** 客户机 zeke 上的回收程序日志文件 （续）

```
    SLOT                  ty    capacity         space vsn
       0                  lt        33.0G        33.0G 000766
       1                  lt        33.0G        33.0G 000173
       2                  lt        32.9G         0    CSM689
       3                  lt        32.9G        19.6G CSM690
       4                  lt        33.0G        33.0G CSM691
       5                  lt        33.0G        33.0G CSM692
    Total Capacity:  197.6G bytes, Total Space Available: 151.5G bytes
    Volume utilization 23%, high 60% VSN_min 90%
    Recycling is ignored on this archive set.



 4  Family: defaultset.2         Path: /etc/opt/SUNWsamfs/archiver.cmd
    Vendor: Sun SAM-FS               Product: Archive set
    SLOT                  ty    capacity         space vsn
       0                  lt        32.9G         0    CSM689
       1                  at        48.5G        23.3G 000020
       2                  at        23.8G        23.8G 000021
       3                  at        48.5G         2.6G 000025
       4                  at        48.5G       361.4k 000026
       5                  at        48.5G        48.5G 000027
       6                  at        48.5G        48.5G 000028
       7                  at        48.5G         0    000029
    Total Capacity:  348.0G bytes, Total Space Available: 146.8G bytes
    Volume utilization 57%, high 60% VSN_min 90%
    Recycling is ignored on this archive set.



 5  Family: archiveset.1         Path: /etc/opt/SUNWsamfs/archiver.cmd
    Vendor: Sun SAM-FS               Product: Archive set
    SLOT                  ty    capacity         space vsn
       0                  lt        32.8G        44.1M CEL170
       1                  lt        32.8G        16.8G CFC504
       2                  lt        33.0G        33.0G CFC503
    Total Capacity:  98.6G bytes, Total Space Available: 49.8G bytes
    Volume utilization 49%, high 60% VSN_min 90%
    Recycling is ignored on this archive set.



 6  Family: archiveset.2         Path: /etc/opt/SUNWsamfs/archiver.cmd
    Vendor: Sun SAM-FS               Product: Archive set
```

代码示例 7-21 客户机 zeke 上的回收程序日志文件 （续）

```
   SLOT                  ty    capacity         space vsn
      0                  at        48.5G        23.3G 000020
      1                  at        23.8G        23.8G 000021
      2                  at        48.5G        48.5G 000022
      3                  at        48.5G        48.5G 000023
      4                  at        48.5G        48.5G 000024
   Total Capacity:  218.0G bytes, Total Space Available: 192.8G bytes
   Volume utilization 11%, high 60% VSN_min 90%
   Recycling is ignored on this archive set.



21 VSNs:

                     ---Archives---   -----Percent-----   defaultset.1
-----Status-----     Count    Bytes   Use Obsolete Free   Library:Type:VSN
in multiple sets       0      0        0   100      0 stk_l20:lt:CSM689
partially full       111      2.8G     8    31     61 stk_l20:lt:CSM690
empty VSN              0      0        0     0    100 stk_l20:lt:000173
empty VSN              0      0        0     0    100 stk_l20:lt:CSM691
empty VSN              0      0        0     0    100 stk_l20:lt:CSM692
empty VSN              0      0        0     0    100 stk_l20:lt:000766

                     ---Archives---   -----Percent-----   defaultset.2
-----Status-----     Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN            0      0        0   100      0   skyrs:at:000029
no-data VSN            0      0        0    99      1   skyrs:at:000026
partially full       111      2.8G     6    88      6   skyrs:at:000025
empty VSN              0      0        0     0    100   skyrs:at:000028
empty VSN              0      0        0     0    100   skyrs:at:000027

                     ---Archives---   -----Percent-----   archiveset.1
-----Status-----     Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN            0      0        0    99      1 stk_l20:lt:CEL170
partially full       677      2.3G     8    40     52 stk_l20:lt:CFC504
empty VSN              0      0        0     0    100 stk_l20:lt:CFC503

                     ---Archives---   -----Percent-----   archiveset.2
-----Status-----     Count    Bytes   Use Obsolete Free   Library:Type:VSN
in multiple sets       0      0        0    51     49   skyrs:at:000020
empty VSN              0      0        0     0    100   skyrs:at:000022
empty VSN              0      0        0     0    100   skyrs:at:000023
empty VSN              0      0        0     0    100   skyrs:at:000024
in multiple sets       0      0        0     0    100   skyrs:at:000021

                     ---Archives---   -----Percent-----   stk_l20
-----Status-----     Count    Bytes   Use Obsolete Free   Library:Type:VSN
```

代码示例 **7-21** 客户机 zeke 上的回收程序日志文件 （续）

```
empty VSN               0       0         0      0    100 stk_l20:lt:CLN018
partially full         13     80.3k       0      0    100 stk_l20:lt:CEL139



Recycler finished.

========== Recycler ends at Mon Jun  4 09:49:53 2001 ===========
```

9. **在 Sun SAM-Remote 服务器上输入 sam-recycler(1M) 命令，测试回收程序。**

   确保回收程序未回收 Sun SAM-Remote 服务器上任何为 Sun SAM-Remote 客户机保留的 VSN。

   例如：

```
zeke# sam-recycler -dvx
```

   上述命令将会运行回收程序，并将其活动写入至回收程序日志文件中。有关 sam-recycler(1M) 命令的更多信息，请参见 sam-recycler(1M) 手册页。

   代码示例 7-22 显示了回收程序日志文件范例。

代码示例 **7-22** 回收程序日志文件

```
# recycler.log file from server sky.

========== Recycler begins at Mon Jun  4 09:50:44 2001 ===========
Initial 6 catalogs:

0  Family: adic1             Path: /var/opt/SUNWsamfs/catalog/adic1
   Vendor: ADIC                  Product: Scalar 1000
   SLOT                  ty    capacity         space vsn
      0                  at         1.3G          1.2G 000001
      1                  at         1.3G          1.3G 000002
      2                  at         1.3G          1.3G 000004
      3                  at        48.5G          0    000010
      4                  at        48.5G          0    000011
      5                  at        48.5G         43.5G 000018
      6                  at        48.5G          0    000019
      7                  at        48.5G         23.3G 000020
      8                  at        23.8G         23.8G 000021
      9                  at        48.5G         48.5G 000022
     10                  at        48.5G         48.5G 000023
     11                  at        48.5G         48.5G 000024
     12                  at        48.5G          2.6G 000025
```

代码示例 **7-22** 回收程序日志文件 （续）

```
      13                    at        48.5G       361.4k 000026
      14                    at        48.5G        48.5G 000027
      15                    at        48.5G        48.5G 000028
      16                    at        48.5G          0    000029
      17                    at         1.3G          1.3G 000005
      18                    at        48.5G        48.5G 000016
      19                    at        23.8G        23.8G CLN001
      20                    at        23.8G        23.8G CLN002
      21                    at        23.8G        23.8G CLN004
      22                    at        23.8G        23.8G CLN003
      23                    at        48.5G       421.6M 000015
      24                    at         1.3G          1.3G 000000
      25                    at        48.5G          0    000013
      26                    at         1.3G          1.3G 000003
      27                    at        48.5G        43.6G 000007
      28                    at        48.5G        41.8G 000008
      29                    at        48.5G        46.9G 000006
      30                    at        48.5G        48.3G 000009
      31                    at        48.5G          0    000014
      32                    at        48.5G          0    000012
      33                    at        48.5G        40.1G 000017
   Total Capacity:  1.2T bytes, Total Space Available: 708.7G bytes
   Volume utilization 43%, high 95% VSN_min 50%
   Recycling is ignored on this robot.



1  Family: hy                    Path: /var/opt/SUNWsamfs/catalog/historian
   Vendor: Sun SAM-FS                Product: Historian
   SLOT                  ty    capacity         space vsn
      (no VSNs in this media changer)
   Total Capacity:  0    bytes, Total Space Available: 0    bytes
   Volume utilization 0%, high 95% VSN_min 50%
   Recycling is ignored on this robot.



2  Family: testset.1             Path: /etc/opt/SUNWsamfs/archiver.cmd
   Vendor: Sun SAM-FS                Product: Archive set
   SLOT                  ty    capacity         space vsn
      0                  at        48.5G          0    000010
      1                  at        48.5G          0    000014
   Total Capacity:  97.1G bytes, Total Space Available: 0    bytes
   Volume utilization 100%, high 60% VSN_min 90%: *** Needs recycling ***
   Recycling is ignored on this archive set.
```

代码示例 **7-22** 回收程序日志文件 （续）

```
3  Family: testset.2            Path: /etc/opt/SUNWsamfs/archiver.cmd
   Vendor: Sun SAM-FS               Product: Archive set
SLOT                  ty    capacity         space vsn
      0                  at        48.5G         0    000019
      1                  at        48.5G       421.6M 000015
   Total Capacity:  97.1G bytes, Total Space Available: 421.6M bytes
   Volume utilization 99%, high 60% VSN_min 90%: *** Needs recycling ***
   Recycling is ignored on this archive set.



4  Family: allsam1.1            Path: /etc/opt/SUNWsamfs/archiver.cmd
   Vendor: Sun SAM-FS               Product: Archive set
   SLOT                  ty    capacity         space vsn
      0                  at         1.3G         1.2G 000001
      1                  at         1.3G         1.3G 000002
      2                  at         1.3G         1.3G 000004
      3                  at         1.3G         1.3G 000005
      4                  at         1.3G         1.3G 000003
   Total Capacity:  6.5G bytes, Total Space Available: 6.3G bytes
   Volume utilization 3%, high 60% VSN_min 90%
   Recycling is ignored on this archive set.



5  Family: allsam1.2            Path: /etc/opt/SUNWsamfs/archiver.cmd
   Vendor: Sun SAM-FS               Product: Archive set
   SLOT                  ty    capacity         space vsn
      0                  at        48.5G        43.6G 000007
      1                  at        48.5G        41.8G 000008
      2                  at        48.5G        46.9G 000006
      3                  at        48.5G        48.3G 000009
   Total Capacity:  194.2G bytes, Total Space Available: 180.6G bytes
   Volume utilization 6%, high 60% VSN_min 90%
   Recycling is ignored on this archive set.



Need to select candidate for media changer testset.1 to free up 39.8G bytes.
Quantity of data to move limited to (no limit) bytes and 1 VSNs.
```

代码示例 **7-22** 回收程序日志文件 （续）

```
Checking 000010.  Need to free 39.8G, quantity limit: (no limit), VSN count: 1.
   VSN is in correct media changer... good.
   VSN is not already recycling... good.
   VSN has no request files... good.
   VSN has no 'archive -n' files...good.
   VSN was not specified as "no_recycle" in recycler.cmd file... good.
   VSN does not exceed VSN count limit... good.
   VSN does not exceed data quantity limit... good.
   VSN meets minimum gain requirement.
   Recycling is ignored on this media changer - VSN not marked for recycling.
Checking 000014.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN is in correct media changer... good.
   VSN is not already recycling... good.
   VSN has no request files... good.
   VSN has no 'archive -n' files...good.
   VSN was not specified as "no_recycle" in recycler.cmd file... good.
   VSN exceeds VSN count limit - skipped.
Checking 000019.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000015.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000001.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000003.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000004.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000005.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000002.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000008.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000007.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000006.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000009.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000011.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000029.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000013.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000012.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
```

代码示例 **7-22** 回收程序日志文件 （续）

```
Checking 000026.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000025.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000020.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000017.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000018.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking CLN003.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000021.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000022.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000027.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000028.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000023.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000024.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000016.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking CLN001.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking CLN002.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking CLN004.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000000.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
No candidate was found in this media changer.

Need to select candidate for media changer testset.2 to free up 38.8G bytes.
Quantity of data to move limited to (no limit) bytes and 1 VSNs.
Checking 000010.  Need to free 38.8G, quantity limit: (no limit), VSN count: 1.
   VSN not in correct media changer.
Checking 000014.  Need to free 38.8G, quantity limit: (no limit), VSN count: 1.
   VSN not in correct media changer.
Checking 000019.  Need to free 38.8G, quantity limit: (no limit), VSN count: 1.
   VSN is in correct media changer... good.
   VSN is not already recycling... good.
   VSN has no request files... good.
   VSN has no 'archive -n' files...good.
```

代码示例 **7-22** 回收程序日志文件 （续）

```
   VSN was not specified as "no_recycle" in recycler.cmd file... good.
   VSN does not exceed VSN count limit... good.
   VSN does not exceed data quantity limit... good.
   VSN meets minimum gain requirement.
   Recycling is ignored on this media changer - VSN not marked for recycling.
Checking 000015.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN is in correct media changer... good.
   VSN is not already recycling... good.
   VSN has no request files... good.
   VSN has no 'archive -n' files...good.
   VSN was not specified as "no_recycle" in recycler.cmd file... good.
   VSN exceeds VSN count limit - skipped.
Checking 000001.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000003.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000004.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000005.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000002.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000008.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000007.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000006.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000009.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000011.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000029.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000013.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000012.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000026.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000025.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000020.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000017.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000018.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
```

代码示例 **7-22** 回收程序日志文件 （续）

```
   VSN not in correct media changer.
Checking CLN003.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000021.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000022.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000027.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000028.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000023.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000024.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000016.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking CLN001.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking CLN002.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking CLN004.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
Checking 000000.  Need to free 0E, quantity limit: (no limit), VSN count: 0.
   VSN not in correct media changer.
No candidate was found in this media changer.
34 VSNs:

                    ---Archives---   -----Percent-----   testset.1
-----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN           0      0        0   100      0   adic1:at:000010
no-data VSN           0      0        0   100      0   adic1:at:000014

                    ---Archives---   -----Percent-----   testset.2
-----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN           0      0        0   100      0   adic1:at:000019
partially full      677      2.3G     5    93      2   adic1:at:000015

                    ---Archives---   -----Percent-----   allsam1.1
-----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
partially full       97    173.8M     1     9     90   adic1:at:000001
no-data VSN           0      0        0     2     98   adic1:at:000003
no-data VSN           0      0        0     2     98   adic1:at:000004
empty VSN             0      0        0     0    100   adic1:at:000005
empty VSN             0      0        0     0    100   adic1:at:000002

                    ---Archives---   -----Percent-----   allsam1.2
```

代码示例 7-22 回收程序日志文件 （续）

```
-----Status-----     Count     Bytes    Use Obsolete Free   Library:Type:VSN
no-data VSN            0         0      0      13     87   adic1:at:000008
partially full        98       1.6G     3       7     90   adic1:at:000007
no-data VSN            0         0      0       3     97   adic1:at:000006
empty VSN              0         0      0       0    100   adic1:at:000009

                    ---Archives---    -----Percent-----    adic1
-----Status-----     Count     Bytes    Use Obsolete Free   Library:Type:VSN
no-data VSN            0         0      0     100      0   adic1:at:000011
no_recycle VSN         0         0      0     100      0   adic1:at:000029
no-data VSN            0         0      0     100      0   adic1:at:000013
no-data VSN            0         0      0     100      0   adic1:at:000012
no_recycle VSN         0         0      0      99      1   adic1:at:000026
no_recycle VSN         0         0      0      94      6   adic1:at:000025
no_recycle VSN         0         0      0      51     49   adic1:at:000020
no-data VSN            0         0      0      17     83   adic1:at:000017
no-data VSN            0         0      0      10     90   adic1:at:000018
empty VSN              0         0      0       0    100   adic1:at:CLN003
no_recycle VSN         0         0      0       0    100   adic1:at:000021
no_recycle VSN         0         0      0       0    100   adic1:at:000022
no_recycle VSN         0         0      0       0    100   adic1:at:000027
no_recycle VSN         0         0      0       0    100   adic1:at:000028
no_recycle VSN         0         0      0       0    100   adic1:at:000023
no_recycle VSN         0         0      0       0    100   adic1:at:000024
empty VSN              0         0      0       0    100   adic1:at:000016
empty VSN              0         0      0       0    100   adic1:at:CLN001
empty VSN              0         0      0       0    100   adic1:at:CLN002
empty VSN              0         0      0       0    100   adic1:at:CLN004
partially full        12      88.3k     0       0    100   adic1:at:000000



Recycler finished.

========== Recycler ends at Mon Jun  4 09:51:05 2001 ===========
```

选择要回收的 VSN 时，请检查回收程序日志文件末尾显示柱状列表数据的部分。最左边的一栏以 `Status` 为标题。在上面的回收程序日志文件中，`Status` 栏中列出了数个具有 `no_recycle` 状态的 VSN。这些 VSN 是客户机使用的 VSN。

最适宜回收的 VSN 是那些在 `Count`、`Bytes` 和 `Use` 栏中具有 0 值的 VSN。列表中最后一个 VSN 的状态显示为 `partially full`。此 VSN，其 `Count`、 `Bytes` 和 `Use` 统计值分别为 12、 88.3k 和 0，并不是适于回收的 VSN。

10. **分析客户机和服务器的** `recycler.log` **文件**。

    本步骤说明如何选择适于回收的 VSN。

检查客户机的 recycler.log 文件。文件的末尾有一个 Status 栏。具有以下状态类型条目的 VSN 是适于回收的 VSN：

- no-data VSN。要回收 no-data VSN，请参见第 182 页 “回收 no-data VSN”。
- partially full。要回收 partially full VSN，请参见第 184 页 “回收 partially full VSN”。

▼ 回收 no-data VSN

no-data VSN 是最容易回收的 VSN。对于这些 VSN，其 Count、 Bytes 和 Use 字段均为 0（零）。

1. **检查客户机的 recycler.log 文件，确定是否有 no-data VSN。**

   根据本章中的示例，可以回收客户机 zeke 的 VSN 000029 和 000026，因为它们是 no-data VSN。这可以从代码示例 7-23 中看出，该表显示了客户机 zeke 上的 recycler.log 文件。

代码示例 7-23 客户机 zeke 上的 recycler.log 文件

```
# From the client zeke recycler.log file:
                    ---Archives---   -----Percent-----   defaultset.2
-----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN           0       0       0   100      0   skyrs:at:000029
no-data VSN           0       0       0    99      1   skyrs:at:000026
partially full      111     2.8G      6    88      6   skyrs:at:000025
empty VSN             0       0       0     0    100   skyrs:at:000028
empty VSN             0       0       0     0    100   skyrs:at:000027
```

2. **检查服务器的 recycler.log 文件，确定您在上一步骤中选择的 VSN 是否同样出现在服务器的回收程序日志文件中。**

   执行此操作的目的是确定来自服务器的有效数据是否归档在这些 VSN 上。

   代码示例 7-24 显示了服务器 recycler.log 文件中 no_recycle VSN 的数据。在上一个步骤中， VSN 000029 和 000026 已被选择进行回收，并且服务器的 recycler.log 文件中的数据与客户机的 recycler.log 文件中的数据相同。

代码示例 7-24 服务器 sky 上的 recycler.log 文件

```
# From the Server log file:

                    ---Archives---   -----Percent-----   adic1
-----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN           0       0       0   100      0   adic1:at:000011
no_recycle VSN        0       0       0   100      0   adic1:at:000029zeke
no-data VSN           0       0       0   100      0   adic1:at:000013
```

代码示例 **7-24** 服务器 sky 上的 recycler.log 文件 （续）

```
no-data VSN             0       0       0    100       0   adic1:at:000012
no_recycle VSN          0       0       0     99       1   adic1:at:000026
no_recycle VSN          0       0       0     94       6   adic1:at:000025
no_recycle VSN          0       0       0     51      49   adic1:at:000020
no-data VSN             0       0       0     17      83   adic1:at:000017
no-data VSN             0       0       0     10      90   adic1:at:000018
empty VSN               0       0       0      0     100   adic1:at:CLN003
no_recycle VSN          0       0       0      0     100   adic1:at:000021
no_recycle VSN          0       0       0      0     100   adic1:at:000022
no_recycle VSN          0       0       0      0     100   adic1:at:000027
no_recycle VSN          0       0       0      0     100   adic1:at:000028
no_recycle VSN          0       0       0      0     100   adic1:at:000023
no_recycle VSN          0       0       0      0     100   adic1:at:000024
empty VSN               0       0       0      0     100   adic1:at:000016
empty VSN               0       0       0      0     100   adic1:at:CLN001
empty VSN               0       0       0      0     100   adic1:at:CLN002
empty VSN               0       0       0      0     100   adic1:at:CLN004
partially full         12    88.3k      0      0     100   adic1:at:000000
```

3. （可选）使用 tplabel**(1M)** 或 odlabel**(1M)** 命令重新标记 **VSN**。

如果来自服务器的有效数据没有归档在选定的 VSN 上，则可以重新标记该 VSN。

注 – 此操作将会破坏 VSN 上的所有数据并收回 VSN 的空间。

例如，对于磁带 VSN 000029，可以输入以下命令：

```
server# tplabel -vsn 000029 -old 000029 at.000029
```

重新标记 VSN 000029 后，您将重新得到该 VSN 上 100% 的空间。

如果介质是磁光盘，则应使用 odlabel(1M) 命令。有关 odlabel(1M) 命令的更多信息，请参见 odlabel(1M) 手册页。

4. 调度回收计划。

在未启用 Sun SAM-Remote 软件的 Sun StorEdge SAM-FS 环境中，可以创建一个 cron(1) 作业以便自动执行回收过程。不过，一旦启用 Sun SAM-Remote 软件，请勿让回收程序自动运行。

注意 – 切记，不要在 Sun SAM-Remote 服务器和 Sun SAM-Remote 客户机上同时进行回收活动。您应该根据符合自身网站的需要，定期手动执行回收过程。虽然这种方式费时费力，不过，这是确保数据完好且避免错误重新标记卡盒的唯一方法。

▼ 回收 partially full VSN

您也可回收那些具有 partially full 状态的 VSN。

1. **检查客户机的 recycler.log 文件，确定是否有 partially full VSN。**

   根据本章中的示例，您可以回收客户机 zeke 的 VSN 000025，因为其状态为 partially full。这可以从代码示例 7-25 中看出，该表显示了客户机 zeke 上的 recycler.log 文件。

代码示例 7-25 客户机 zeke 上的 recycler.log 文件

```
# From the client zeke recycler.log file:
                    ---Archives---   -----Percent-----   defaultset.2
-----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN           0        0     0    100      0    skyrs:at:000029
no-data VSN           0        0     0     99      1    skyrs:at:000026
partially full      111      2.8G    6     88      6    skyrs:at:000025
empty VSN             0        0     0      0    100    skyrs:at:000028
empty VSN             0        0     0      0    100    skyrs:at:000027
```

表中显示， VSN 000025 的 6% 空间在使用中。这些是回收此 VSN 之前必须重新归档的有效归档映像。本过程中的以下步骤介绍如何确保这些有效归档映像重新归档至另一个 VSN。

2. **检查服务器的 recycler.log 文件，确保来自服务器的有效数据没有归档在该 VSN 上。**

   例如，在代码示例 7-26 中，查看上一步骤中被选择用于回收的 VSN 000025 的数据。服务器的 recycler.log 文件指出 VSN 000025 有 6% 的可用空间，这与客户机的 recycler.log 文件中报告的百分比相同。服务器并不知道客户机的归档映像，因此服务器无法报告所占用的百分比划分成 6% 的使用中归档映像和 88% 的过期映像。服务器将报告所有剩余的 94% 空间由过期的归档映像占用。

代码示例 7-26 服务器 sky 上的 recycler.log 文件

```
# From the Server log file:

                    ---Archives---   -----Percent-----   adic1
-----Status-----    Count    Bytes   Use Obsolete Free   Library:Type:VSN
no-data VSN           0        0     0    100      0    adic1:at:000011
no_recycle VSN        0        0     0    100      0    adic1:at:000029
no-data VSN           0        0     0    100      0    adic1:at:000013
no-data VSN           0        0     0    100      0    adic1:at:000012
no_recycle VSN        0        0     0     99      1    adic1:at:000026
no_recycle VSN        0        0     0     94      6    adic1:at:000025
no_recycle VSN        0        0     0     51     49    adic1:at:000020
no-data VSN           0        0     0     17     83    adic1:at:000017
no-data VSN           0        0     0     10     90    adic1:at:000018
```

代码示例 **7-26** 服务器 sky 上的 recycler.log 文件 （续）

```
empty VSN              0       0        0      0     100   adic1:at:CLN003
no_recycle VSN         0       0        0      0     100   adic1:at:000021
no_recycle VSN         0       0        0      0     100   adic1:at:000022
no_recycle VSN         0       0        0      0     100   adic1:at:000027
no_recycle VSN         0       0        0      0     100   adic1:at:000028
no_recycle VSN         0       0        0      0     100   adic1:at:000023
no_recycle VSN         0       0        0      0     100   adic1:at:000024
empty VSN              0       0        0      0     100   adic1:at:000016
empty VSN              0       0        0      0     100   adic1:at:CLN001
empty VSN              0       0        0      0     100   adic1:at:CLN002
empty VSN              0       0        0      0     100   adic1:at:CLN004
partially full        12    88.3k       0      0     100   adic1:at:000000
```

3. **在该 VSN 上运行 chmed(1M) 命令及其 +c 选项。**

   对于本过程中的示例，请输入以下的命令：

```
server# chmed +c at.000025
```

   该命令向回收程序表示您要重新归档此 VSN 上的有效文件。要重新归档的文件占用 6% 的空间 （参见客户机 recycler.log 文件中的 Use 栏）。有关 chmed(1M) 命令的更多信息，请参见 chmed(1M) 手册页。

4. **使用 sam-recycler(1M) 命令再次运行回收程序。**

   对于本过程中的示例，请输入以下的命令：

```
client# sam-recycler -dvx
```

   此命令用于标记要重新归档的每一个有效文件，并向归档程序表示每一个有效文件应重新归档至另一个 VSN。

5. **启动归档程序。**

   您可以让归档程序定期运行，也可以从客户机上的 samu(1M) 实用程序中输入 :arrun，来启动归档程序。有关 :arrun 命令的更多信息，请参见 samu(1M) 手册页。

6. **归档完成后，输入 sam-recycler(1M) 命令，在客户机上重新运行回收程序。**

   这可确保所有有效文件均已被重新归档。

   对于本过程中的示例，请输入以下的命令：

```
client# sam-recycler -dvx
```

7. **（可选）在服务器上使用 `tplabel`(1M) 或 `odlabel`(1M) 命令重新标记 VSN。**

   如果该 VSN 的 `Count`、`Bytes` 和 `Use` 字段均为 0（零），则可以从服务器中重新标记该 VSN。

   对于本过程中的示例，可使用以下的命令重新标记该磁带 VSN：

```
server# tplabel -vsn 000025 -old 000025 at.000025
```

   上面的命令将重新标记该 VSN 并破坏其中的所有数据。该 VSN 被重新标记后，您可以得到该 VSN 的 88% 空间。

   如果介质是磁光盘，则应使用 `odlabel`(1M) 命令。有关 `odlabel`(1M) 命令的更多信息，请参见 `odlabel`(1M) 手册页。

8. **安排回收计划。**

   在未启用 Sun SAM-Remote 软件的 Sun StorEdge SAM-FS 环境中，可以创建一个 `cron`(1) 作业以便自动执行回收过程。不过，一旦启用 Sun SAM-Remote 软件，请勿让回收程序自动运行。

注意 – 切记，不要在 Sun SAM-Remote 服务器和 Sun SAM-Remote 客户机上同时进行回收活动。您可根据符合自身网站的需要，定期手动执行回收过程。虽然这种方式费时费力，不过，这是确保数据完好且避免错误重新标记卡盒的唯一方法。

在 Sun SAM-Remote 环境中进行回收 — 方法 2

本节介绍使用 Sun SAM-remote 软件回收卷的另一种方法。

注意 – 只有在严格执行本过程中的步骤，且您的配置经测试可以正确执行回收过程之后，才可以在 Sun SAM-Remote 环境中运行回收程序。

▼ 配置回收过程 — 方法 2

1. **在 Sun SAM-Remote 客户机上输入 `sam-recycler`(1M) 命令，确定最适宜回收的卷。**

   例如：

```
client# sam-recycler -dvx
```

   您可以通过分析回收程序日志文件来确定最适宜的卷。

2. **在 Sun SAM-Remote 服务器上输入 `chmed`(1M) 命令，在选定的 VSN 上设置回收标志。**

   例如：

```
server# chmed +c at.00025
```

3. **在 Sun SAM-Remote 客户机上输入 `sam-recycler`(1M) 命令，回收在 Sun SAM-Remote 客户机上选定的 VSN。**

   例如：

```
client# sam-recycler -dvx
```

4. **等到所回收的 VSN 完全清除了归档映像。**

   客户机上的归档程序执行此项操作。

5. **在 Sun SAM-Remote 服务器上输入 `tplabel`(1M) 或 `odlabel`(1M) 命令，重新标记那些已完全清除归档映像的卷。**

6. **在 Sun SAM-Remote 服务器上，清除所有标志（例如 `R` 或 `c`），以免将卷用于在 Sun SAM-Remote 客户机上执行归档。**

   再次强调，请勿在 Sun SAM-Remote 服务器和 Sun SAM-Remote 客户机上同时执行回收活动。

第 8 章

高级主题

本章介绍超出基本系统管理和使用范围的高级主题。

它包括下列主题：

使用设备日志

设备日志工具可以提供设备专用的错误信息，您可以使用这些信息来分析某些类型的设备问题。它可帮助您确定自动化库、磁带机或光盘驱动器发生故障事件的顺序。请注意，设备日志工具并不能收集软介质错误，如可恢复的读取错误。

系统将设备日志消息分别写入至各个不同的日志文件。每一个自动化库、磁带和光盘驱动器设备均有一个日志文件，并且历史记录也有一个日志文件。日志文件位于 `/var/opt/SUNWsamfs/devlog` 目录下。每一个日志文件的名称分别与相应设备的设备序号相同。

示例。假设您的环境中具有 Sun StorEdge SAM-FS 文件系统和一个配有两个光盘驱动器的 Hewlett Packard 光盘库。

代码示例 8-1 显示了 `mcf` 文件。

代码示例 8-1 `mcf` 文件示例

```
/dev/samst/c1t5u0   40   hp   hp40   -   etc/opt/SUNWsamfs/hp40_cat
/dev/samst/c1t4u0   41   mo   hp40   -
/dev/samst/c1t6u0   42   mo   hp40   -
```

代码示例 8-2 显示了 /var/opt/SUNWsamfs/devlog 文件。

代码示例 8-2 devlog 文件

```
# pwd
/var/opt/SUNWsamfs/devlog
# ls
40      41      42      43
#
```

设备 43 是历史记录。

何时使用设备日志

设备日志可以方便地生成许多日志消息，特别适用于已打开所有设备的全部日志选项且具有大量设备活动的场合。设备日志设置最初设置为下面的默认值：

```
err retry syserr date
```

如果您怀疑 Sun StorEdge SAM-FS 环境中配置的某个设备出现问题，则应为该设备启用额外的日志事件。另外，如果您的服务供应商建议您这样做，则也应启用设备日志。在这些情况下，请将事件设置为 detail。在极为特殊的情况下，您的服务供应商可能建议您将某个设备的事件设置为 all。这可以添加额外的日志信息。不过，一般而言，在系统运行时设置过多的日志并无益处，甚至是不可行的。

当运行 samexplorer(1M) 命令时，系统会自动收集设备日志信息。这样，作为问题分析活动的一部分，文件系统服务人员可以复查任何可能的设备错误信息。

启用设备日志

您可以使用两种方法来启用设备日志。

对于第 1 种和第 2 种方法：

- *eq* 是设备在 mcf 文件中定义的设备序号，或对于所有设备，它为关键字 all。
- samset(1M) 手册页中列出了设备日志事件。另外，下面也列出了这些设备日志事件。请注意，设备日志消息仅以英文文本的格式提供。 *event* 是下表中的一种或多种事件类型：
  - all
  - date
  - default

- detail
- err
- event
- label
- mig
- module
- msg
- none
- retry
- stage
- stage_ck
- syserr
- tapealert
- time

可按以下两种方式中的一种启用设备日志。这些过程如下所示：

- 第 191 页 “使用 samset(1M) 命令启用设备日志”
- 第 191 页 “通过编辑 defaults.conf 文件启用设备日志”

▼ 使用 samset(1M) 命令启用设备日志

- **使用 samset(1M) 命令。**

  例如：

```
# samset devlog eq event
```

  其中的 *eq*，用于指定要为其记录消息的设备的设备序号。

  而其中的 *event*，用于指定一个或多个如第 190 页 “启用设备日志”中所述的事件。如果要指定多个事件，请用空格进行分隔。

  有关 samset(1M) 命令的更多信息，请参见 samset(1M) 手册页。

▼ 通过编辑 defaults.conf 文件启用设备日志

1. **成为超级用户。**
2. **使用 vi(1) 或其他编辑器打开文件 /etc/opt/SUNWsamfs/defaults.conf。**

3. **在 `defaults.conf` 文件中添加 `devlog` 指令。**

   添加以下指令：

   ```
   devlog eq event
   ```

   其中的 *eq*，用于指定要为其记录消息的设备的设备序号。

   而其中的 *event*，用于指定一个或多个如第 190 页 “启用设备日志”中所述的事件。如果要指定多个事件，请用空格进行分隔。

   Sun StorEdge SAM-FS 文件系统会在启动时，自动将每个可用设备的事件类型设置为 `default`。另外，还可以使用 `samset`(1M) 命令来确定每一个设备日志的当前设置。

4. **保存并关闭 `defaults.conf` 文件。**

5. **使用 `samd`(1M) `config` 命令应用 `defaults.conf` 文件中的更改。**

   ```
   # samd config
   ```

使用可移除介质文件

您可以使用 `request`(1) 命令手动创建、写入和读取那些不使用磁盘高速缓存来缓冲数据的文件。采用这种方式创建的文件称为**可移除介质文件**。

与典型的 Sun StorEdge SAM-FS 文件类似，可移除介质文件也具有权限、用户名、群组名和大小属性。但是，这些文件的数据并不保留在磁盘高速缓存中。因此，您可以创建文件大小大于磁盘高速缓存空间的文件，并将它们写入可移除介质卡盒。对于您在 `request`(1) 命令中指定的文件，系统会为它在 `.inodes` 文件中创建 inode 条目。用户并不需要知道文件在可移除介质中的起始点。（对于那些数据位于磁盘高速缓存中的文件来说，情形与此相同。）Sun StorEdge SAM-FS 文件系统会从 inode 条目中读取信息。多个可移除介质文件可以保留在同一个卷中。

如果可移除介质文件存储在多个卷中，则将该文件称为卷溢出文件。**卷溢出**功能允许系统在多个卡盒的多个卷中存储单个文件。卷溢出文件是一种可移除介质文件。如果您有非常大的文件，文件大小超出了所选介质的容量，则卷溢出功能将非常有用。

▼ 创建可移除介质或卷溢出文件

1. **使用 `tplabel`(1M) 或 `odlabel`(1M) 命令标记磁带或磁光盘卡盒。**
   有关这些命令的信息，请参见各自的手册页。

2. **使用 `request`(1M) 命令。**
   而且该命令至少应该使用以下选项：

```
request -m media_type -v vsn [vsn/vsn ...][-l vsn_file] input_file
```

表 8-1 `request`(1) 命令的参数

参数	含义
media_type	可移除介质卡盒的介质类型。有关有效 *media_type* 参数的信息，请参见 `mcf`(4) 手册页。
vsn	可移除介质卡盒的卷序列名。 如果指定了多个 *vsn*，则将创建卷溢出文件。您最多可以为卷溢出文件指定 256 个 *vsn*。应该使用正斜杠字符 (/) 来分隔 *vsn* 参数。 所指定的 *vsn* 不应该是 Sun StorEdge SAM-FS 环境中用于自动归档的卷。每次归档时，系统均会将下一个要归档的文件添加至当前数据的末尾，并将 EOF 标签移至数据的后面。
vsn_file	包含 *vsn* 列表的输入文件。如果您拥有许多 *vsn*，则在输入文件中指定 *vsn* 列表要比在命令行中便捷得多。
input_file	要写入至可移除介质卡盒的文件。该文件必须位于 Sun StorEdge SAM-FS 文件系统中。

示例 1。以下命令用于创建可移除介质文件。

```
# request -m lt -v aaa rem1
```

示例 2。以下命令用于在三个卷中创建卷溢出文件：

```
# request -m lt -v TAPE01/TAPE02/TAPE03 large.file
```

您必须依次读取和写入可移除介质文件。如果请求的卷位于 `mcf` 文件所定义的自动化库中，则 Sun StorEdge SAM-FS 文件系统会自动安装该卷。

如果卷中存在可移除介质文件，则回收程序不能对该卷进行回收。回收程序希望只有已归档的文件位于指定用于归档的特定卷中。另外，归档程序不会归档可移除介质文件。

NFS 不支持可移除介质文件。

请注意，使用 request(1) 命令将会忽略归档程序的一般功能。

有关说明如何创建可移除介质文件的示例，请参见 request(1) 手册页。

使用分段文件

Sun StorEdge SAM-FS 环境支持分段文件。将文件分成多个段可以提高磁带存储检索速度、改善存取性能以及增强大文件的可管理性。分段文件的大小可以大于物理磁盘高速缓存。使用分段文件，您可以只将文件的一部分时刻保留在磁盘高速缓存中。

可以使用 segment(1) 命令来指定分段大小。您所设置的分段大小不能小于当前文件的大小。

分段文件支持磁带拆分功能。将文件分段后，可将文件同时拆分至多个磁带设备中，这样大大缩短了存储各个文件段的时间。由于用户只需恢复所需的文件段 （而不是整个文件），因此提高了数据访问速度。

由于只有发生更改的文件部分才需要重新归档，因此分段还可以提高归档效率。文件的各个分段可以并行归档，并且分段文件可以并行登台。这提高了系统的归档和恢复性能。

用户可以为文件、目录或整个文件系统启用分段功能。分段文件支持其他所有 Sun StorEdge SAM-FS 功能。

以下几节说明了分段文件与非分段文件之间的差异。有关分段文件的更多信息，请参见 segment(1) 或 sam_segment(3) 手册页。

归档

对于分段文件，归档单位是文件段自身，而不是整个文件。所有归档属性和优先级均应用于单个文件段，而不应用于整个文件。

所归档的单位是文件段。您可以通过在 archiver.cmd 文件中为归档集指定 -drives 和 -drivemin 参数，来拆分段。

例如，假定文件系统中有一个大小为 100 MB 的分段文件，其段大小为 10 MB。如果 archiver.cmd 中使用 -drives 2 指令定义归档集，则该分段文件将并行归档至 2 个驱动器。段 1、3、5、7 和 9 归档在第一个驱动器中，而段 2、4、6、8 和 10 归档在第二个驱动器中。

归档程序只归档已发生更改的文件段 — 而不是整个文件。最多可以为每一个文件段创建四个归档副本。 Sun StorEdge SAM-FS 也支持对文件段使用卷溢出功能。

注 – 分段文件的索引不含用户数据。索引被视为元数据，由系统分配至文件系统归档集。

故障恢复

有关在出现故障时恢复分段文件的信息，请参见 《Sun StorEdge SAM-FS 故障排除指南》。

使用系统错误工具报告

系统错误工具 (System Error Facility, SEF) 报告系统用于收集自动化库中的磁带设备生成的日志检测数据，然后将这些数据写入至日志文件并转换成可读的格式。它包括以下项目：

- 日志文件，包含从磁带设备日志检测页收集的数据。
- `sefreport`(1M) 命令，以可读的格式将日志文件写入至 `stdout`。该日志文件可以作为用户分析脚本的输入项。

日志检测页因供应商而异。有关参数代码、控制位和参数值的含义，请参见每一个特定设备的供应商文档。

独立磁带机不支持 SEF。对于那些不支持 `tapealert`(1M) 功能的旧 SCSI-2 设备，SEF 报告功能非常有用。有关更多信息，请参见 `tapealert`(1M) 手册页。

▼ 启用 SEF 报告

1. **成为超级用户。**
2. **使用 `mkdir`(1) 命令创建 SEF 目录。**

   例如：

```
# mkdir /var/opt/SUNWsamfs/sef
```

3. **使用 touch(1) 命令创建日志文件。**

   在安装报告系统后，您随时可以通过创建 sefdata 日志文件来启用 SEF 报告功能。起初， SEF 日志文件必须为空。

```
# touch /var/opt/SUNWsamfs/sef/sefdata
```

   上面的命令示例表明所创建的 SEF 日志文件位于 /var/opt/SUNWsamfs/sef/sefdata 目录中。这是默认位置。

4. **使用 samd(1M) stop 和 samd(1M) start 初始化 SEF 报告功能。**

   例如：

```
# samd stop
# samd start
```

   生成的 SEF 数据将添加至日志文件的末尾。

   您可以对 SEF 报告功能进行配置，以便在其他位置记录和读取日志检测数据。有关从其他位置读取日志检测文件的更多信息，请参见 sefreport(1M) 手册页。

SEF 报告输出

使用 sefreport(1M) 命令之前，请确保 /opt/SUNWsamfs/sbin 位于您的命令路径中。 SEF 报告输出的内容由标题行和日志检测数据组成。

记录中每一页的日志检测数据将打印在标题行的后面。对于每一日志检测数据页，系统均会打印用于标识页码的行，随后是一行列标题。接下来打印数据，每行三列，各列的标题分别如下：param code、 control 和 param value。所有生成的数据均采用十六进制格式。

▼ 生成 SEF 输出

- **使用 sefreport(1M) 命令生成 SEF 输出。**

  下面是 sefreport(1M) 命令的最常用选项：

  - -d 选项。 -d 选项用于生成附加的设备信息。它将为每一条记录写入附加的包含设备序号和路径名的标题行。这便于用户搜索和查找与特定设备相关的 SEF 记录。
  - -v 或 -t 选项。

    -v 选项用于生成详细的信息。它将与设备序号、页码和 VSN 相关的信息添加至记录的每一行。这使用户可以只选择与特定设备或特定卷相关的行。

`-t` 用于生成包含文字说明的日志检测页。对于日志检测页输出的每一行，报告中都包含了一个额外的字符串，字符串的内容包括：设备序号、页码、 VSN 和参数编码说明。

不要在同一命令行中指定 `-t` 和 `-v` 选项。它们是互相排斥的。

例如，下面的 SEF 命令从默认位置读取 SEF 日志文件，写入每一个设备的设备编号和路径名，并生成输出：

```
# sefreport -d /var/opt/SUNWsamfs/sef/sefdata > sef.output
```

代码示例 8-3 显示了 `sef.output` 文件的内容。

代码示例 8-3 `sef.output` 的内容

```
Record no. 1
Mon Mar 26 11:17:48 2001  STK      9840            1.25 VSN 002981
   Eq no. 32   Dev name /dev/rmt/1cbn

   PAGE CODE 2
   param code  control   param value
       00h       74h     0x0
       01h       74h     0x0
       02h       74h     0x0
       03h       74h     0x0
       04h       74h     0x0
       05h       74h     0x40050
       06h       74h     0x0

   PAGE CODE 3
   param code  control   param value
       00h       74h     0x0
       01h       74h     0x0
       02h       74h     0x0
       03h       74h     0x0
       04h       74h     0x0
       05h       74h     0x140
       06h       74h     0x0

   PAGE CODE 6
   param code  control   param value
       00h       74h     0x0

Record no. 2
Mon Mar 26 11:30:06 2001  STK      9840            1.25 VSN 002999
```

代码示例 **8-3** sef.output 的内容 （续）

```
    Eq no. 31   Dev name /dev/rmt/0cbn

    PAGE CODE 2
    param code  control   param value
        00h       74h     0x0
        01h       74h     0x0
        02h       74h     0x0
        03h       74h     0x0
        04h       74h     0x0
        05h       74h     0x1400a0
        06h       74h     0x0

    PAGE CODE 3
    param code  control   param value
        00h       74h     0x0
        01h       74h     0x0
        02h       74h     0x0

        03h       74h     0x0
        04h       74h     0x0
        05h       74h     0x190
        06h       74h     0x0

    PAGE CODE 6
    param code  control   param value
        00h       74h     0x0

Record no. 3
Mon Mar 26 11:30:23 2001  STK      9840            1.25 VSN 002981
    Eq no. 32   Dev name /dev/rmt/1cbn

    PAGE CODE 2
    param code  control   param value
        00h       74h     0x0
        01h       74h     0x0
        02h       74h     0x0
        03h       74h     0x0
        04h       74h     0x0
        05h       74h     0x18400f0
        06h       74h     0x0


    PAGE CODE 3
```

代码示例 8-3 `sef.output` 的内容 （续）

```
    param code  control   param value
        00h       74h     0x0
        01h       74h     0x0
        02h       74h     0x0
        03h       74h     0x0
        04h       74h     0x0
        05h       74h     0x1e0
        06h       74h     0x0

    PAGE CODE 6
    param code  control   param value
        00h       74h     0x0
.
```

注 – 受本手册的内容所限，上面的输出已作了删节。

有关 SEF 日志文件的更多信息（包括内容和格式），请参见 `sefdata`(4) 手册页。有关可选 SEF 报告格式的更多信息，请参见 `sefreport`(1M) 手册页。

管理 SEF 日志文件

SEF 日志文件的管理方式，与任何其他 Sun StorEdge SAM-FS 日志文件相同。用户可以定期运行 `cron`(1) 作业，以将当前日志文件保存至另一位置、删除旧 SEF 文件、创建新 （空）的 SEF 文件或执行其他任务。

另外，用户还可使用 `log_rotate.sh`(1M) 实用程序来循环更新该日志文件。

有关用于管理 SEF 日志文件的工具的更多信息，请参见 `cron`(1) 或 `log_rotate.sh`(1M) 手册页。

SEF `sysevent` 功能

除了可以查看 SEF 日志文件以外，用户还可以使用 Solaris `sysevent` 来查看磁带机 SCSI 日志检测错误计数器的第 2 页和第 3 页，以便了解介质分析信息。默认情况下， SEF `sysevent` 特性是启用的，并且其默认的轮询间隔为每次卸载之前。SEF `sysevent` 的行为由 `defaults.conf` 和 `samset` 控制。

▼ 创建 SEF sysevent 处理器

1. **使用类似于以下的命令创建 /var/tmp/xx 文件：**

```
#!/bin/ksh
echo "$@" >> /var/tmp/xx.dat
exit 0
```

2. **使 /var/tmp/xx 文件成为可执行文件：**

```
# chmod a+rwx /var/tmp/xx
```

3. **键入以下命令，将 SEF sysevent 处理器添加到 syseventd(1M) 文件：**

```
# syseventadm add -vSUNW -pSUNWsamfs -cDevice -sSEF
/var/tmp/xx \"\$VENDOR\" \"\$PRODUCT\" \"\$USN\" \"\$REV\"
\$TOD \$EQ_ORD \"\$NAME\" \$INQ_TYPE \"\$MEDIA_TYPE\"
\"\$VSN\" \$LABEL_TIME \$LP2_PC0 \$LP2_PC1 \$LP2_PC2 \$LP2_PC3
\$LP2_PC4 \$LP2_PC5 \$LP2_PC6 \$LP3_PC0 \$LP3_PC1 \$LP3_PC2
\$LP3_PC3 \$LP3_PC4 \$LP3_PC5 \$LP3_PC6 \$WHERE \$sequence

# syseventadm restart
```

此命令将创建包含 SEF sysevent 处理器 (/var/tmp/xx) 的 /etc/sysevent/config/SUNW,SUNWsamfs,Device,sysevent.conf 文件，并将该事件处理器载入 syseventd 守护进程。在运行 syseventadm(1M) 命令时需要使用双引号，这是由于字符串可能为空，并且其中的数据包含位置信息。

注 – 由于 Solaris 8 中不存在 syseventadm(1M) 命令，因此您必须手动创建 /etc/sysevent/config/SUNW,SUNWsamfs,Device,sysevent.conf 文件及其内容，然后键入 pkill -HUP syseventd 重新启动 syseventd 守护进程。

---

4. **要加载 SEF sysevent 处理器，请执行 pkill -HUP syseventd 激活 /var/tmp/xx SEF sysevent 处理器。**

有关 SEF sysevent 使用方法的更多信息，请参见 sefsysevent(4) 手册页。

附录 A

具有供应商特定操作过程的自动化库的基本操作

Sun StorEdge SAM-FS 环境中可以包含来自众多不同厂商的自动化库。对于其中大多数库，您应该使用第 9 页 “使用自动化库和手动载入的驱动器”中所述的操作过程进行操作。但是，某些库需要执行供应商提供的特定操作过程，本章即介绍这些过程。

注 – Sun StorEdge SAM-FS 软件可与许多厂商的自动化库兼容。有关自动化库的型号、固件级别以及其他兼容性方面的信息，请与 Sun 的销售代表或授权服务提供商联系。

本章将介绍以下自动化库：

ADIC/Grau 自动化库

如果您使用的是 ADIC/Grau 自动化库，请使用本节所述的过程导入和导出卡盒。这些过程不同于第 9 页 “使用自动化库和手动载入的驱动器”中所述的过程。

由于从物理上向 ADIC/Grau 自动化库中添加和从中取出卡盒时，使用的是供应商提供的实用程序，因此 Sun StorEdge SAM-FS 接口 （`import`(1M)、 `samexport`(1M) 和 File System Manager）只能影响库目录。

注 – x64 硬件平台上的 Sun StorEdge SAM-FS 软件不支持 ADIC/Grau 网络连接库。

▼ 导入卡盒

要导入卡盒，请执行以下步骤。

1. **使用 ADIC/Grau 命令将卡盒从物理上移入自动化库。**
2. **使用 Sun StorEdge SAM-FS `import`(1M) 命令更新库目录。**
   此命令的使用格式如下：

```
import -v volser eq
```

表 A-1 `import`(1M) 命令的参数

参数	含义
volser	要添加的 *volser*。`grauaci` 接口在使用新条目更新库目录之前，将检验 ADIC/Grau 自动化库是否具有 *volser* 信息。
eq	所访问的设备在 `mcf` 文件中定义的设备序号。

▼ 导出卡盒

要导出卡盒，请执行以下步骤。

1. **使用 Sun StorEdge SAM-FS `samexport`(1M) 命令从库目录中删除该条目。**
   按以下某一种格式使用此命令：

```
samexport eq:slot
samexport media_type.vsn
```

表 A-2 `samexport`(1M) 命令的参数

参数	含义
eq	所访问的设备在 `mcf` 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。
media_type	介质类型。有关有效介质类型的列表，请参见 `mcf`(4) 手册页。
vsn	分配给卷的卷序列名。

导出每一个 VSN 后，samexport(1M) 命令将更新库目录，并将每一个 VSN 的库目录条目从库目录移至历史记录。

2. **使用 ADIC/Grau 命令将卡盒从物理上移出自动化库。**

Fujitsu LMF 自动化库

如果您使用的是 Fujitsu LMF 自动化库，请使用本节所述的过程导入和导出卡盒。这些过程不同于第 9 页 “使用自动化库和手动载入的驱动器”中所述的过程。

由于从物理上向 Fujitsu LMF 自动化库中添加或从中取出卡盒时，使用的是供应商提供的实用程序，因此 Sun StorEdge SAM-FS 接口 （import(1M)、 samexport(1M) 和 File System Manager）只能影响库目录。

注 – x64 硬件平台上的 Sun StorEdge SAM-FS 软件不支持 Fujitsu LMF 网络连接库。

▼ 导入卡盒

要导入卡盒，请执行以下步骤。

1. **使用 Fujitsu 命令将卡盒从物理上移入自动化库。**
2. **使用 Sun StorEdge SAM-FS** import**(1M) 命令更新库目录。**

   此命令的使用格式如下：

```
import -v volser eq
```

表 A-3 import(1M) 命令的参数

参数	含义
volser	要添加的 *volser*。 fujitsulmf 接口在使用新条目更新库目录之前，将检验 LMF 自动化库是否具有 *volser* 信息。
eq	所访问的设备在 mcf 文件中定义的设备序号。

▼ 导出卡盒

要导出卡盒，请执行以下步骤。

1. **使用 Sun StorEdge SAM-FS `samexport`(1M) 命令从库目录中删除条目。**
   按以下某一种格式使用此命令：

```
samexport eq:slot
samexport media_type.vsn
```

表 A-4 `samexport`(1M) 命令的参数

参数	含义
eq	所访问的设备在 `mcf` 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。
media_type	介质类型。有关有效介质类型的列表，请参见 `mcf`(4) 手册页。
vsn	分配给卷的卷序列名。

   导出每一个 VSN 后，`samexport`(1M) 命令将更新库目录，并将每一个 VSN 的库目录条目从 Sun StorEdge SAM-FS 库目录移至 Sun StorEdge SAM-FS 历史记录。

2. **使用 Fujitsu 命令将卡盒从物理上移出自动化库。**

IBM 3584 UltraScalable 磁带库

Sun StorEdge SAM-FS 环境可支持 IBM 3584 UltraScalable 磁带库。以下几节从多方面介绍此库的操作方法，这些操作不同于第 9 页 “使用自动化库和手动载入的驱动器”中所述的过程。

导入卡盒

启动 Sun StorEdge SAM-FS 软件时，邮箱中的卡盒不会自动导入。

清洁驱动器

要在 Sun StorEdge SAM-FS 环境中使用此库，应该禁用自动清洁，而启用手动清洁。此过程在《IBM 3584 UltraScalable Tape Library Planning and Operator Guide》（IBM 的出版物 GA32-0408-01）中进行了介绍。 `ibm3584`(7) 手册页中也对此进行了介绍。

分区

此库可以包含多个磁带机。如果您使用了多个驱动器，则可能需要将这一个物理库划分成两个、三个或四个逻辑库。如果您已将此库划分成两个或更多的逻辑库，则在将 IBM 3584 库添加到 Sun StorEdge SAM-FS 环境中之前，应确保这些逻辑库可以正常工作。

从已分区的库中导出卡盒时，只有从中导出卡盒的逻辑库才可以访问该抽屉插槽。如果您手动取出卡盒并将其重新插入，则任何逻辑分区均可访问该卡盒。

▼ 取出卡盒

以下步骤说明了在这种情况下所执行的操作：

1. **打开挡门。**
2. **取出卡盒。**
3. **关闭挡门。**
4. **等待挡门锁定，然后取消锁定。**
5. **打开挡门。**
6. **放回卡盒。**
7. **关闭挡门。**

有关在 Sun StorEdge SAM-FS 环境中将此库用作逻辑分区库的更多信息，请参见 IBM 的相关文档或 `ibm3584`(7) 手册页。

IBM 3494 库

Sun StorEdge SAM-FS 环境可支持 IBM 3494 库。以下几节从多方面介绍了此库的操作方法，这些操作不同于第 9 页 “使用自动化库和手动载入的驱动器”中所述的过程。

注 – x64 硬件平台上的 Sun StorEdge SAM-FS 软件不支持 IBM 3494 网络连接库。

▼ 导入卡盒

要导入卡盒，请执行以下步骤。

1. **将新的介质放入 I/O 插槽中。**
2. **关闭挡门。**

   自动化库将锁定挡门，并介质移至存储区域。一次只能导入 100 个卷。

   如果已将库配置为 `access=private`，则这是您需要执行的最后一个步骤。移动介质后这个库会通知守护进程，且介质会被添加到目录中。
3. **使用 `import`(1M) 命令将介质添加到目录中。（可选）**

   仅在将库配置为 `access=shared` 时才需执行此步骤。

   如果已将库配置为 `access=shared`，请执行 `import`(1M) 命令将介质添加到目录中。

▼ 导出卡盒

1. **使用 `export`(1M) 命令导出卡盒。**

   此命令会将介质移至 I/O 区域，并打开操作员面板上的输出模式指示灯。
2. **从物理上将介质从 I/O 区域中取出。**

Sony 8400 PetaSite 直接连接的自动化库

Sony 8400 PetaSite 系列自动化库不同于其他型号的 Sony 产品，因为它配有八插槽导入和导出邮箱 （插槽编号为 400 至 407）。因此，该系统可以方便快捷地执行导入和导出操作。此自动化库使用条码读取器。

由于邮箱插槽可用作存储插槽，因此 Sun StorEdge SAM-FS 库目录可以跟踪记录邮箱插槽。

注 – 本节所述的内容仅适用于 Sony 8400 PetaSite 直接连接的自动化库。这些信息既不适用于 Sony B9 和 B35 直接连接的自动化库，也不适用于第 209 页“Sony 网络连接的自动化库”。

▼ 导入磁带

要导入磁带，请执行以下步骤。

1. **按下自动化库前面板上的打开 / 关闭按钮，打开自动化库的挡门。**
2. **将卡盒装入邮箱插槽。**
3. **按下自动化库前面板上的打开 / 关闭按钮，手动关闭邮箱的挡门。**
   自动化库将在关闭挡门后检查邮箱插槽，以获取卡盒条码。如果条码有问题，该插槽的 `in`（进）和 `out`（出）指示灯均会闪烁。
4. **运行 `import`(1M) 命令，以使 Sun StorEdge SAM-FS 系统识别导入的卡盒。**
   此命令的使用格式如下：

```
import eq
```

其中的 *eq*，用于指定所访问的设备在 `mcf` 文件中定义的设备序号。

另外，还可使用 File System Manager 来执行此步骤。

导出磁带

导出磁带卡盒的步骤取决于您是否将邮箱插槽用作存储插槽。

▼ 在邮箱插槽未用作存储插槽时导出磁带

在邮箱插槽未用作存储插槽时，请使用以下步骤导出卡盒。

1. **运行 `move`(1M) 命令将卡盒移至邮箱插槽（插槽编号为 400 至 407）。**
   此命令的使用格式如下：

```
move source_slot destination_slot eq
```

表 A-5 `move`(1M) 命令的参数

参数	含义
source_slot	卡盒当前所在插槽的插槽编号。
destination_slot	卡盒将要移至的插槽编号。
eq	所访问的设备在 `mcf` 文件中定义的设备序号。

2. **按下自动化库前面板上的打开 / 关闭按钮。**

   此时，挡门会打开。

3. **从邮箱插槽中取出卡盒。**

4. **按下自动化库前面板上的打开 / 关闭按钮，手动关闭邮箱的挡门。**

5. **运行 `samexport`(1M) 命令，以使 Sun StorEdge SAM-FS 系统识别导出的卡盒。**

   此命令的使用格式如下：

   ```
   samexport eq
   ```

   其中的 *eq*，用于指定所访问的设备在 `mcf` 文件中定义的设备序号。

   另外，还可使用 File System Manager 来执行此步骤。

▼ 在邮箱插槽用作存储插槽时导出磁带

如果将邮箱插槽用作存储插槽，并且要导出的卡盒位于其中一个邮箱插槽，则请使用以下步骤导出卡盒。

1. **按下自动化库前面板上的打开 / 关闭按钮。**

   此时，挡门会打开。

2. **从邮箱插槽中取出卡盒。**

3. **按下自动化库前面板上的打开 / 关闭按钮，手动关闭邮箱的挡门。**

4. **运行 `samexport`(1M) 命令，以使 Sun StorEdge SAM-FS 系统识别导出的卡盒。**

   此命令的使用格式如下：

   ```
   samexport eq
   ```

   其中的 *eq*，用于指定所访问的设备在 `mcf` 文件中定义的设备序号。

   另外，还可使用 File System Manager 来执行此步骤。

▼ 如何将卡盒移至另一个插槽

要将卡盒移至另一个插槽，请执行以下步骤：

1. **确保来源插槽中具有卡盒，而目标插槽中没有卡盒。**

2. **运行 `move`(1M) 命令。**

此命令的使用格式如下：

```
move eq:source_slot destination_slot
```

表 A-6 move(1M) 命令的参数

参数	含义
eq	所访问的设备在 mcf 文件中定义的设备序号。
source_slot	卡盒当前所在插槽的插槽编号。
destination_slot	卡盒将要移至的插槽编号。

另外，还可使用 File System Manager 来执行此步骤。

Sony 网络连接的自动化库

如果您使用的是 Sony 网络连接的自动化库，请使用本节所述的过程导入和导出卡盒。这些过程不同于第 9 页 “使用自动化库和手动载入的驱动器”中所述的过程。

由于从物理上向 Sony 自动化库中添加和从中取出卡盒时，所使用的是供应商提供的实用程序，因此 Sun StorEdge SAM-FS 接口 （import(1M)、 samexport(1M) 和 File System Manager）只能影响库目录。

注 – x64 硬件平台上的 Sun StorEdge SAM-FS 软件不支持 Sony 网络连接库。

▼ 导入卡盒

要导入卡盒，请执行以下步骤。

1. **使用 Sony 命令将卡盒从物理上移入自动化库。**
2. **使用 import(1M) 命令更新库目录。**

   此命令的使用格式如下：

```
import -v [" ] volser [ " ] eq
```

表 **A-7** import(1M) 命令的参数

参数	含义
" "	引号。如果 *volser* 包含空格，则必须包含在引号内。
volser	要添加的 *volser*。PSC API 接口在使用新条目更新库目录之前，将检验 Sony 自动化库是否具有 volser 信息。如果卡盒从物理上并没有位于库中，则会在历史记录目录中添入一个条目。
eq	所访问的库在 mcf 文件中定义的设备序号。

▼ 导出卡盒

要导出卡盒，请执行以下步骤。

1. **使用 samexport(1M) 命令从库目录中删除条目。**

   按以下某一种格式使用此命令：

```
samexport eq:slot
samexport media_type.vsn
```

表 **A-8** samexport(1M) 命令的参数

参数	含义
eq	所访问的设备在 mcf 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。
media_type	介质类型。有关有效介质类型的列表，请参见 mcf(4) 手册页。
vsn	分配给卷的卷序列名。

   导出每一个 VSN 后，samexport(1M) 命令将更新库目录，并将每一个 VSN 的库目录条目从库目录移至历史记录。

2. **使用 Sony 命令将卡盒从物理上移出自动化库。**

StorageTek ACSLS 连接的自动化库

如果您使用的是 StorageTek ACSLS 连接的自动化库，请使用本节所述的过程导入和导出卡盒。这些过程不同于第 9 页“使用自动化库和手动载入的驱动器”中所述的过程。

邮箱是自动化库中的一个区域，它用于向自动化库中添加和从中取出卡盒。某些 StorageTek 自动化库一次只能导入和导出一个卡盒。 StorageTek 9714 和 StorageTek 9710 就属于受 Sun StorEdge SAM-FS 环境支持的带邮箱的 StorageTek 自动化库。 StorageTek 9730 使用邮箱插槽。在 StorageTek 文档中，通常将邮箱和邮箱插槽称为 *CAP*。

在向 ACSLS 连接的自动化库中导入和从中导出卡盒时，请记住以下几点：

- 导入卡盒时， Sun StorEdge SAM-FS 命令仅影响库目录。 `import`(1M) 命令并不会从物理上将卡盒插入自动化库中。您必须使用 ACSLS 命令才能从物理上导入卡盒。
- 导出卡盒时，除非在 `samexport`(1M) 命令中使用 `-f` 选项，否则 Sun StorEdge SAM-FS 命令仅影响库目录。使用 `-f` 选项后， Sun StorEdge SAM-FS 系统会将卷放入卡盒访问端口 (Cartridge Access Port, CAP) 中，并相应地更新目录。如果不使用 `-f` 选项，则仅更新目录而不会将卷放入 CAP 中，因此您仍必须使用 ACSLS 命令从物理上导出卡盒。

按照协议，应该由您来负责维护 ACSLS 清单和 Sun StorEdge SAM-FS 目录。

另外，还可使用 `samu`(1M) 或 File System Manager 来执行导入和导出过程。

▼ 导入磁带

- **要导入磁带卡盒，请使用 `import`(1M) 命令。**

  此命令的使用格式如下：

```
import -v vsn eq
```

表 A-9 `import`(1M) 命令的参数

参数	含义
vsn	分配给卷的卷序列名。
eq	所访问的设备在 `mcf` 文件中定义的设备序号。

`import`(1M) 命令将在库目录中添入一个新的 VSN。如果历史记录中包含此 VSN，则 Sun StorEdge SAM-FS 软件会将 VSN 信息从历史记录移至库目录。

▼ 使用邮箱导出磁带

您既可以按插槽导出磁带卡盒，也可以按 VSN 导出磁带卡盒。

- **要导出磁带卡盒，请使用 `samexport`(1M) 命令。**

  按以下某一种格式使用此命令：

```
samexport [-f] eq:slot
samexport [-f] media_type.vsn
```

表 A-10 `samexport`(1M) 命令的参数

参数	含义
`-f`	命令 Sun StorEdge SAM-FS 系统将卷放入卡盒访问端口 (Cartridge Access Port, CAP) 中，并相应地更新目录。
eq	所访问的设备在 `mcf` 文件中定义的设备序号。
slot	自动化库中存储插槽的编号，与库目录中标识的编号相同。
media_type	介质类型。有关有效介质类型的列表，请参见 `mcf`(4) 手册页。
vsn	分配给卷的卷序列名。

导出每一个 VSN 后，`samexport`(1M) 命令将更新库目录，并将每一个 VSN 的库目录条目从库目录移至历史记录。

词汇表

英文字母

DAU 磁盘分配单元 (Disk allocation unit, DAU)。联机存储的基本单位。也称作**块大小**。

FDDI 光纤分布式数据接口 (Fiber-distributed data interface, FDDI) 是一种局域网数据传输标准，最大传输距离在 200 km （124 英里）以内。 FDDI 协议基于令牌环协议。

FTP 文件传输协议 (File transfer protocol)。一种用于通过 TCP/IP 网络在两个主机之间传输文件的 Internet 协议。

LAN 局域网 (Local area network,LAN)。

LUN 逻辑单元编号 (Logical unit number)。

`mcf` 主配置文件 (master configuration file)。初始化期间读取的文件，用于定义文件系统环境中各个设备 （拓扑结构）之间的关系。

NFS 网络文件系统 (Network file system)。一种由 Sun 发布的文件系统，可对异构网络上的远程文件系统进行透明访问。

NIS Sun OS 4.0 （最低）网络信息服务 (Network Information Service)。一种分布式网络数据库，包含网络上系统和用户的相关重要信息。 NIS 数据库存储在主服务器和所有从属服务器上。

RAID 独立磁盘冗余阵列 (Redundant array of independent disks)。一种使用若干独立磁盘来可靠地存储文件的磁盘技术。该技术可在单个磁盘出现故障时防止数据丢失，并可提供容错磁盘环境以及比单个磁盘更高的吞吐量。

RPC 远程过程调用。 NFS 用于实现自定义网络数据服务器的底层数据交换机制。

SAM-QFS 一种组合了 Sun StorEdge SAM-FS 软件和 Sun StorEdge QFS 文件系统的配置。SAM-QFS 不仅为用户和管理员提供了高速的标准 UNIX 文件系统接口，而且还提供了若干存储及归档管理实用程序。 SAM-QFS 既可以使用 Sun StorEdge SAM-FS 命令集中的许多命令，也可以使用标准 UNIX 文件系统命令。

`samfsdump` 一个程序，为给定文件组创建控制结构转储文件并复制所有控制结构信息。该程序与 UNIX `tar`(1) 实用程序类似，但通常不复制文件数据。另请参见 *samfsrestore*。

`samfsrestore` 一个程序，用于从控制结构转储文件中恢复索引节点和目录信息。另请参见 *samfsdump*。

SCSI 小型计算机系统接口 (Small Computer System Interface)。一种电子通信技术规范，常用于像磁盘、磁带机和自动化库这样的外围设备。

Sun SAM-Remote 服务器 (Sun SAM-Remote server) 既是一台功能完备的 Sun StorEdge SAM-FS 存储管理服务器，又是一个 Sun SAM-Remote 服务器守护进程。它可定义 Sun SAM-Remote 客户机之间共享的库。

Sun SAM-Remote 客户机 (Sun SAM-Remote client) 一种具有客户机守护进程的 Sun StorEdge SAM-FS 系统。它包含许多伪设备，也有自己的库设备。客户机用以存储一个或多个归档副本的归档介质是由 Sun SAM-Remote 服务器决定的。

tar 磁带归档 (Tape archive)。一种用于归档映像的标准文件和数据记录格式。

TCP/IP 传输控制协议/Internet 协议 (Transmission Control Protocol/Internet Protocol)。 Internet 协议 (IP) 负责主机到主机的寻址、路由和数据包传递；传输控制协议 (TCP) 负责在各个应用点之间可靠地传递数据。

VSN 卷序列名 (Volume serial name)。如果是将数据归档至可移除介质卡盒， VSN 是写入卷标中的磁带和光盘的逻辑标识符。如果是将数据归档至磁盘高速缓存，VSN 是磁盘归档集的唯一名称。

WORM 单次写入多次读取 (write once read many)。介质的一种存储分类，即只能写入一次，但可多次读取。

A

安装点 (mount point) 文件系统所安装到的目录。

B

备份存储 (backup storage) 一组文件的快照，旨在防止意外丢失数据。备份不仅包括文件的属性，而且还包括关联的数据。

本地文件系统 (local file system) 安装在 Sun Cluster 系统的某一个节点上的文件系统。它对于其他节点来说，可用性不高。此外，本地文件系统也指安装在独立服务器上的文件系统。

C

超级块 (superblock) 文件系统中的一种数据结构，用于定义文件系统的基本参数。超级块将被写入存储系列集中的所有分区，并标识该系列集各个分区的成员。

传输器 (robot) 自动化库的一部分，用于在存储插槽和驱动器之间移动卡盒。也称作**传输设备** *(transport)*。

磁盘分配单元 (disk allocation unit) 请参见 *DAU*。

磁盘分散读写 (disk striping) 跨多个磁盘记录同一文件的过程。该方法可提高存取性能，进而增加整体存储能力。另请参见**分散读写** *(striping)*。

磁盘高速缓存 (disk cache) 文件系统软件的磁盘驻留部分，用于在联机磁盘高速缓存与归档介质之间创建并管理数据文件。单个磁盘分区或整个磁盘均可用作磁盘高速缓存。

磁盘缓冲区 (disk buffer) 在 Sun SAM-Remote 配置中，磁盘缓冲区是指服务器系统上用于将数据从客户机归档至服务器的缓冲区。

磁盘空间阈值 (disk space threshold) 由管理员定义的磁盘高速缓存的最大利用率或最小利用率。释放程序 (releaser) 可根据这些预定义的磁盘空间阈值来控制磁盘高速缓存的使用情况。

存储插槽 (storage slot) 自动化库中的位置，其中存储了卡盒 （如果卡盒并未在驱动器中使用）。如果是直接连接的库，则存储插槽中的内容将保存在自动化库的目录中。

存储系列集 (storage family set) 由一组磁盘组成，并由一个磁盘系列设备表示。

D

登台 (staging) 将近线文件或脱机文件从归档存储中恢复至联机存储的过程。

多读取器文件系统 (Multireader file system) 一种具备单写入、多读取特点的文件系统，允许您指定安装到多个主机上的文件系统。多个主机可读取该文件系统，但只有一个主机可向该文件系统写入数据。多个读取主机可通过 `mount`(1M) 命令的 `-o reader` 选项指定。一个写入主机可通过 `mount`(1M) 命令的 `-o writer` 选项指定。有关 `mount`(1M) 命令的更多信息，请参见 `mount_samfs`(1M) 手册页。

F

范围阵列 (extent array) 位于文件索引节点 (inode) 内的阵列，用于定义分配给该文件的每个数据块的磁盘位置。

分区 (partition) 设备的一部分或磁光盘卡盒的一面。

分散读写 (striping) 一种以交错方式将所有文件同时写入若干逻辑磁盘的数据存储方法。 SAM-QFS 文件系统提供两种类型的分散读写。即 “硬分散读写 (hard striping)”（使用分散读写组）和 “软分散读写 (soft striping)”（使用 `stripe=`*x* 安装参数）。硬分散读写在设置文件系统时启用，您需要在 `mcf`(4) 文件中定义分散读写组。软分散读写则通过 `stripe=`*x* 安装参数启用，您可针对各个文件系统或单个文件更改它。通过设置 `stripe=0` 可禁用它。如果文件系统由多个具相同数量元素的分散读写组组成，则可同时使用硬分散读写和软分散读写。另请参见**循环** *(round robin)*。

分散读写大小 (stripe size) 向分散读写的下一个设备写入数据前要分配的磁盘分配单元 (DAU) 数。如果使用 `stripe=0` 安装选项，文件系统将采用循环存取方式，而不是分散读写存取方式。

分散读写组 (striped group) 文件系统中的一组设备，在 `mcf` 文件中被定义为一个或多个 `g`*XXX* 设备。系统将分散读写组视作一个逻辑设备，并始终按照磁盘分配单元 (DAU) 的大小进行分散读写。

G

光纤通道 (Fibre Channel) ANSI 制定的标准，用于指定设备之间的高速串行通信。光纤通道常被用作 SCSI-3 中的一种总线结构。

归档程序 (archiver) 一种可自动控制文件到可移除卡盒的复制操作的软件程序。

归档存储 (archive storage) 归档介质上已创建的文件数据的副本。

归档介质 (archive media) 归档文件写入到的介质。归档介质可以是库中的可移除磁带或磁光盘 (magneto-optical) 卡盒。此外，归档介质也可以是另一系统上的某一安装点。

H

回收程序 (recycler) 一种 Sun StorEdge SAM-FS 实用程序，用于收回由过期归档副本占用的卡盒空间。

J

计时器 (timer) 一种限额软件，用于跟踪从用户达到软限制开始直到对该用户施加硬限制为止的时间段。

间接块 (indirect block) 包含一列存储块的磁盘块。文件系统最多可有三级间接块。第一级间接块包含一列用于数据存储的块。第二级间接块包含一列第一级间接块。第三级间接块包含一列第二级间接块。

介质 (media) 磁带或光盘卡盒。

介质回收 (media recycling) 对仅有很少归档文件的归档介质进行回收和再利用的过程。

近线存储 (nearline storage) 一种可移除介质存储。近线存储在访问之前需要自动安装。近线存储通常比联机存储便宜，但所需的访问时间相对长一些。

镜像写入 (mirror writing) 在互不相连的磁盘组上维护文件的两份副本的过程，可防止单个磁盘损坏所导致的数据丢失。

卷 (volume) 卡盒上用于共享数据的命名区域。一个卡盒可以有一个或多个卷。双面卡盒有两个卷，每一面为一个卷。

卷溢出 (volume overflow) 一种允许跨多个卷保存单个文件的功能。对于使用超大型文件 （超过了每个卡盒的容量）的站点，卷溢出功能非常有用。

K

卡盒 (cartridge) 一种包含了记录数据的介质的物理实体，如磁带或光盘。有时称作介质或卷。

可寻址存储 (addressable storage) 包括联机存储 (online)、近线存储 (nearline)、离站存储 (offsite) 和脱机存储 (offline) 等存储空间，用户可通过 Sun StorEdge QFS 或 Sun StorEdge SAM-FS 文件系统访问这些空间。

可移除介质文件 (removable media file) 一种特殊类型的用户文件，可直接从它所驻留的可移除介质卡盒 （如磁带或光盘卡盒）中访问。此外，该文件也用于写入归档和登台 (stage) 文件数据。

客户机-服务器 (client-server) 分布式系统中的交互模型。在该模型中，一个站点中的程序可向另一个站点上的程序发送请求并等待回应。发送请求的程序称作 “客户机 (client)”。提供响应的程序称作 “服务器 (server)”。

库 (library) 请参见**自动化库** *(automated library)*。

库目录 (library catalog) 请参见**目录** *(catalog)*。

块大小 (block size) 请参见 *DAU*。

块分配图 (block allocation map) 一种显示磁盘上每个可用存储块的位图。该位图可指出每个块的状态：是在使用中还是空闲。

宽限期 (grace period) 对于磁盘限额而言，宽限期是指达到软限制之后，系统允许用户继续创建文件并分配存储空间的时间。

L

离站存储 (offsite storage) 远离服务器的存储，用于灾难恢复。

联机存储 (online storage) 可即时访问的存储，如磁盘高速缓存。

连接 (connection) 建立在两个协议模块之间的通道，可提供稳定可靠的数据流传输服务。 TCP 连接就是一台计算机上的 TCP 模块到另一台计算机上的 TCP 模块的连接。

M

名称空间 (name space) 一组文件的元数据部分，用于标识文件、文件属性和存储位置。

目录 (catalog) 自动化库中的 VSN 记录。每个自动化库都有一个目录，而且一个站点有一个记录所有自动化库的历史记录。

目录 (directory) 一种指向文件系统中其他文件和目录的文件数据结构。

N

内核 (kernel) 用于提供基本系统功能的中央控制程序。 UNIX 内核可创建并管理各个进程，并提供不同功能以访问文件系统。此外， UNIX 内核还可提供常规安全性以及通信功能。

Q

驱动器 (drive) 一种向可移除介质卷传入数据或从中传出数据的机械装置。

全局指令 (global directive) 应用于所有文件系统的归档程序指令和释放程序指令。第一个 `fs =` 行之前显示的都是全局指令。

R

软限制 (soft limit) 对于磁盘限额而言，软限制是指用户可以暂时超过的文件系统资源 （块或索引节点）阈值限制。如果超过软限制，系统将启动一个计时器。当超过软限制达到一定时间，系统将无法再分配更多的系统资源，除非您将文件系统的使用率降至软限制水平以下。

S

设备日志 (device logging) 一项可配置功能，用于提供设备特定的错误信息，以供分析设备问题。

设备扫描程序 (device scanner) 该软件用于定期监视所有手动安装的可移除设备，并检测是否存在可供用户或其他进程请求的已安装卡盒。

设备系列集 (family device set) 请参见**系列集** *(family set)*。

审计 （全面） (audit (full)) 载入卡盒以验证其 VSN 的过程。对于磁光盘卡盒，其容量和空间信息将在确定后被输入到自动生成的库目录中。

释放程序 (releaser) 一种 Sun StorEdge SAM-FS 组件，用于标识归档文件并释放其磁盘高速缓存副本，从而增加磁盘高速缓存的可用空间。释放程序可根据阈值的上下限来自动调整联机磁盘存储量。

释放优先级 (release priority) 用于确定文件系统中的文件在归档后的释放先后顺序。释放优先级的计算方法是：将文件的各个属性值与该属性对应的权数相乘，然后将所有相乘结果取和。

数据设备 (data device) 在文件系统中，数据设备指存储文件数据的一个或一组设备。

索引节点 (inode) Index node （索引节点）的缩写。是文件系统用于描述文件的一种数据结构。一个索引节点可描述与文件相关联的所有属性 （除了名称）。这些属性包括：所有权、访问、权限、大小和磁盘系统上的文件位置。

索引节点文件 (inode file) 文件系统上的一种特殊文件 (`.inodes`)，包含了驻留在文件系统上的所有文件的索引节点结构。索引节点的大小是 512 字节；索引节点文件属于元数据文件。在文件系统中，元数据文件与文件数据分开存储。

T

脱机存储 (offline storage)
使用前需要操作员先将其载入。

W

网络连接自动化库 (network-attached automated library)
由不同供应商 （如 StorageTek、 ADIC/Grau、 IBM 或 Sony 等）生产的库，由供应商提供的软件包控制。 Sun StorEdge SAM-FS 文件系统通过使用自动化库的专用 Sun StorEdge SAM-FS 介质更换器守护进程，从而实现与供应商软件的连接。

伪设备 (pseudo device)
未关联任何硬件的软件子系统或驱动程序。

文件系统 (file system)
一种由文件和目录组成的多层结构集合。

文件系统专用指令 (file-system-specific directives)
位于 `archiver.cmd` 文件中的全局指令后的归档程序指令和释放程序指令。不同文件系统有不同的文件系统专用指令，但都以 `fs = `开头。文件系统专用指令的作用域一直到下一条 `fs = ` 指令行或文件结束标记。如果有多条指令作用于一个文件系统，则文件系统专用指令优先于全局指令。

X

系列集 (family set)
由一组独立物理设备 （如某个自动化库中的磁盘组或驱动器组）所代表的存储设备。另请参见**存储系列集** *(storage family set)*。

限额 (quota)
允许用户使用的系统资源量。

小型计算机系统接口 (Small Computer System Interface)
请参见 SCSI。

循环 (round robin) 一种按顺序将全部文件写入若干逻辑磁盘的数据存取方法。当将单个文件写入磁盘时，文件将整个写入第一个逻辑磁盘。然后，第二个文件将写入下一个逻辑磁盘，依此类推。每个文件的大小决定了 I/O 的大小。

另请参见**磁盘分散读写** *(disk striping)* 和**分散读写** *(striping)*。

Y

以太网 (Ethernet) 一种局域分组交换网络技术。以太网最初是针对同轴电缆设计的。但现在，它同样适用于屏蔽双绞线电缆。以太网是一种 10 MB/s 或 100 MB/s 的局域网。

硬限制 (hard limit) 对于磁盘限额而言，硬限制是文件系统资源、数据块或索引节点 (inode) 的最大限制，用户不能超过该限制。

预分配 (preallocation) 在磁盘高速缓存中预先保留一定数量的连续空间以备写入文件的过程。只能对大小为零的文件指定预分配。有关更多信息，请参见 `setfa`(1) 手册页。

元数据 (metadata) 与数据有关的数据。元数据是用于在磁盘上定位文件的确切数据位置的索引信息。元数据由以下各项的有关信息组成：文件、目录、访问控制列表、符号链接、可移除介质、分段文件和分段文件索引。

元数据设备 (metadata device) 用于存储文件系统元数据的设备，如固态磁盘或镜像设备等。在单独的设备上保存文件数据和元数据可以提高性能。在 `mcf`(4) 文件中，元数据设备被声明为 `ma` 文件系统中的 `mm` 设备。

远程过程调用 (remote procedure call) 请参见 *RPC*。

Z

直接 I/O (direct I/O) 一种针对大数据块对齐连续 I/O 的属性。`setfa`(1) 命令的 `-D` 选项即为直接 I/O 选项。该选项可为文件或目录设置直接 I/O 属性。如果应用于目录，直接 I/O 属性是可以继承的。

直接访问 (direct access) 一种文件属性 （永不登台），可指定近线 (nearline) 文件直接从归档介质上访问，而无需在磁盘高速缓存中接收。

直接连接库 (direct-attached library) 使用 SCSI 接口直接连接到服务器上的自动化库。通过 SCSI 连接的库直接由 Sun StorEdge SAM-FS 软件控制。

自动化库 (automated library) 一种自动控制设备，可在没有操作人员参与的情况下自动载入或卸载可移除介质卡盒。自动化库包含一个或多个驱动器，以及一种用于将卡盒移入或移出存储插槽和驱动器的传输机制。

租借 (lease) 授予客户机主机在指定时间段内对文件进行操作的权限。元数据服务器负责向每一台客户机主机发放租借。根据具体情况，可对租借进行续借以允许客户机主机继续操作文件。

索引

C

D

E

F

G

H

T

V

W

X

Y

Z