OpenVault™
Operator's and Administrator's Guide

Document Number 007-3211-004

CONTRIBUTORS

Written by Susan Thomas and Bill Tuthill
Production by Linda Rae Sande
Engineering contributions by Curtis Anderson, Murali Sathyanarayana, and Steve Philipson

OpenVault™ Operator's and Administrator's Guide
Document Number 007-3211-004

Contents

List of Figures and Examples

List of Tables

About This Guide

This *OpenVault Operator's and Administrator's Guide* describes how to administer and operate OpenVault. It also provides an introduction to tertiary storage management.

OpenVault is an emerging standard solution for tertiary storage management from Silicon Graphics. It provides a common interface for applications that manage removable media libraries, loadable drives, and OpenVault client applications.

This book does not describe how to develop OpenVault applications or device support—refer to the *OpenVault Applications Programmer's Guide* and the *OpenVault Infrastructure Programmer's Guide* for this information. Regular IRIX storage management utilities are described in the book *IRIX Admin: Backup, Security, and Accounting*.

Contents of This Guide

Chapter 1, "Introducing OpenVault," describes OpenVault architecture and operation.

Chapter 2, "Installing OpenVault" details server and client setup.

Chapter 3, "Cartridge Life Cycle" discusses the treament of media cartridges.

Chapter 4, "Administering OpenVault," presents procedures for system administrators.

Chapter 5, "Operating OpenVault," tells how to perform day-to-day operator tasks.

Chapter 6, "Reconfiguring OpenVault," talks about changing configurations.

Chapter 7, "Tertiary Storage Management," is a conceptual introduction to this topic.

Appendix A, "OpenVault Error Messages," discusses OpenVault error conditions.

Appendix B, "OpenVault Reference Pages," lists OpenVault administration commands.

Audience

This guide is intended for administrators who set up the OpenVault system and monitor its operation, and for operators who perform prescribed storage management tasks. To use the information in this guide, you should have the following experience:

- understanding UNIX system infrastructure including devices and networking
- writing UNIX shell and *perl* scripts
- using common text editors (for example, *emacs*, *jot*, *nedit*, or *vi*)
- using backup utilities such as *cpio*, *tar*, *xfsdump*, or IRIX NetWorker

Style Conventions

This guide uses the following stylistic conventions:

`screen font`	Indicates system output, such as responses to commands appearing on the screen. Code samples and screen displays also appear in this font.
`user input`	Indicates what you must enter at a command line, such as a command and optional arguments to commands.
variable	Indicates a substitutable string (for example, *logFile*) that you replace with a specific value (for example, myLog).
command	Designates command and utility names.
filename	Indicates directory pathnames and filenames.
...	Denotes omitted material or indicates that the preceding optional items may appear more than once in succession.

Product Support

Silicon Graphics, Inc., provides a comprehensive product support and maintenance program for its products. If you are in North America and would like support for your Silicon Graphics products, contact the Technical Assistance Center at 1-800-800-4SGI. If you are outside North America, contact the Silicon Graphics subsidiary or authorized distributor in your country.

Chapter 1

Introducing OpenVault

OpenVault is a storage library management facility that improves how applications can manage, store, and retrieve removable media (called cartridges). As an overseer of storage applications, and the libraries and drives that manage storage, OpenVault is aware of resources and allocates them accordingly, reducing bottlenecks and device mismanagement.

This chapter introduces OpenVault, and is divided into the following major sections:

What OpenVault Does

OpenVault is a package of mediation software that helps other applications access and manage removable media. This facility can support a wide range of removable media libraries, as well as a variety of drives associated with these libraries. OpenVault includes a database to track cartridges and storage devices. The modular design of OpenVault eases the task of adding support for new robotic libraries and drives.

The advantage of using OpenVault is that it can manage multiple applications and track the devices and removable media that each application uses. Additionally, because it adds a standards-based layer of software between the storage application and a library device, new libraries can be introduced without having to obtain an updated version of an application, and conversely, new applications can be added without having to update library or drive interface software.

OpenVault also has a command-line interface for performing administrator and operator tasks. These tasks include

- setting up and configuring applications and storage devices to run with OpenVault
- monitoring your storage management operations
- organizing your storage libraries for optimal operation

Chapter 4 and Chapter 5 describe how to perform administrator and operator tasks. Appendix B provides an overview of OpenVault administrative commands.

How OpenVault Fits In With Other Software

OpenVault, as middleware, operates between applications and devices, as shown in Figure 1-1. OpenVault uses a client/server model and is designed to work equally well in either a single-host computer system or in a networked environment supporting multiple computers.

OpenVault as Middleware

Software that mediates between operating systems and application programs is called *middleware*. Middleware creates a common language so that users can access data in a variety of formats, or devices from different vendors.

As middleware, OpenVault receives high-level requests from client applications, and translates these requests into a set of low-level robotic or device-controller commands to accomplish storage-related tasks. As overseer of storage management, OpenVault schedules competing storage requests from different applications for the available devices, establishes and allocates cartridge groups by application, and provides mapping from logical cartridge names (as seen by the application) to physical cartridge numbers (as used by robots).

OpenVault software, as middleware, provides two standard interfaces:

1. for storage-based applications that use cartridges
2. for devices (libraries and drives) that store and retrieve information on cartridges

Client-Server Model

OpenVault works as a client-server model. At the most basic level, both client and server can operate on the same host. More typical, though, is a distributed system where multiple clients located on a network send requests to a centrally-managed server, which mediates access to locally attached libraries and drives.

OpenVault Terms

This section describes some storage management terms. Terms not specific to OpenVault are followed by an asterisk (*).

AAPI	Administrative application programming interface. Used to make administrative requests of the system.
AAPI/R	AAPI response. The system response to administrative requests.
ADI	Abstract drive interface. The OpenVault server issues directives to the DCP in a language called ADI.
ADI/R	ADI response. The DCP replies to the OpenVault server in a language called ADI/R.
ALI	Abstract library interface. The OpenVault server issues directives to the LCP in a language called ALI.
ALI/R	ALI response. The LCP replies to the OpenVault server in a language called ALI/R.
barcode *	A horizontal strip of vertical bars of varying widths, which represent symbols referred to as “characters.” Barcodes are used as a cartridge label and are read by devices with read capability.
bay *	A storage area in a library where an array of cartridges reside.
cartridge *	A unit of removable media. May be a tape cartridge, a tape reel, an optical disc, a digital linear tape, a removable magnetic disk, or a videotape.
cartridge group	A set of cartridges organized by common characteristics. A cartridge group may be created to form a cartridge set for use by an application, or to organize cartridges by subject matter, users, and so forth.
catalog *	The listing of OpenVault components. The catalog tracks the status of cartridges, as well as authorized applications, drives, and libraries.

CAPI	Client application programming interface. Similar to AAPI, but used by applications to request service.
CAPI/R	CAPI response. The system response to application requests.
DCP	Drive control program. Required for each drive managed by OpenVault.
DLT *	Digital linear tape.
drive *	A device used to access the contents of cartridges.
drive group	A set of drives organized for a specific reason. Organization may be by type of drive (or drive capabilities) or for specific use by applications, user groups, and so forth.
LCP	Library control program. Required for each removable media library controlled by OpenVault.
library *	The collection of cartridges accessible by a storage device. A library can be automated (robot-assisted) or manual (worked by human operators).
OpenVault server	The set of processes that constitute the central mediation component that accepts client connections and fulfills access requests by forwarding them to appropriate library and drive control programs.
PCL	Physical cartridge label; usually a barcode, but may be human-readable.
shared secret *	A password that is common to two parties that share a secured transaction. For example, a client and server both share the password (secret) that is established at the setup of a client-server relationship.
slot *	A position in the library that can hold a cartridge. The slot may be free (unoccupied) or occupied.
slotmap *	A mapping of the free and occupied slots within a library.

OpenVault Definitions

server host	The host computer running the set of entities that manage OpenVault controlled drives, libraries, and media. These entities are responsible for managing and allocating resources.
client host	A host computer running one or more OpenVault controlled drives or libraries, or application(s) that request services from an OpenVault server. A client host can also be the OpenVault server host.

OpenVault system	The set of hosts (server and clients) comprise the OpenVault system.
OpenVault core	The set of entities that manage drives, libraries, and cartridges as well as service requests from OpenVault applications. The core does not include those processes that actually manipulate the devices on behalf of the OpenVault core (the LCPs and DCPs). The term “core” may also be used in various situations in place of the term OpenVault core.
OpenVault catalog	The central storage repository for an OpenVault system. This repository contains current knowledge about all libraries, drives, cartridges, and applications under OpenVault management.
OpenVault client	The set of processes that communicate with the OpenVault core, either to request services from the core (as would OpenVault applications), or to perform tasks on behalf of the core (an LCP or DCP).
OpenVault application	The set of processes that communicate with the OpenVault server using either CAPI or AAPI. An application might be doing backup or archive, or it might be simply monitoring OpenVault activities.
LCP	The OpenVault client process(es) that directly control a library under OpenVault management. The LCP interprets and services ALI requests on behalf of the OpenVault core.
DCP	The OpenVault client process(es) that directly control a library under OpenVault management. The DCP interprets and services ADI requests on behalf of the OpenVault core.

OpenVault Architecture

OpenVault is organized as a set of cooperating processes. The OpenVault server is a multithreaded process that accepts client connections and fulfills access requests by forwarding them to appropriate library and drive control programs. The OpenVault server maintains a catalog containing information about cartridges in the system, and descriptions of authorized applications, libraries, and drives.

Figure 1-1 shows the arrangement of OpenVault components.

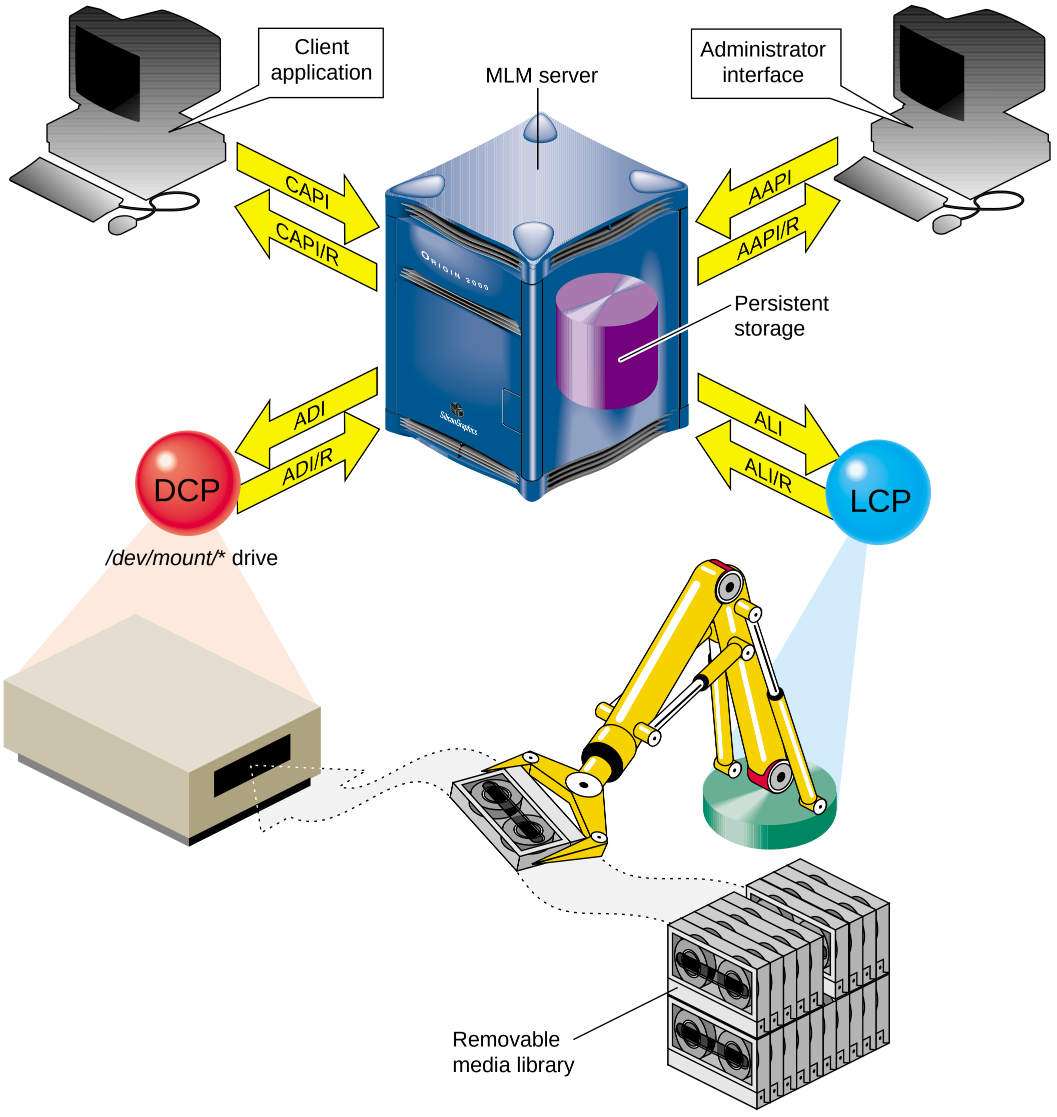

Figure 1-1 OpenVault Architecture

AAPI Programming Interface

AAPI (administrative API) is the language that administrative applications use to communicate with the OpenVault server. Commands and responses are text strings.

The command-response format is semi-asynchronous. After submitting each command, the application waits for the server to acknowledge receiving the command, but need not wait for results before sending the next command. AAPI communications libraries can also work synchronously if this makes implementation more convenient.

CAPI Programming Interface

CAPI (client application programming interface) is the language that client applications use to communicate with the OpenVault server. CAPI commands and responses are text strings. As with AAPI, the command-response format is semi-asynchronous, and access to the server is session-oriented. CAPI is a subset of AAPI.

The *OpenVault Application Programmer's Guide* tells how to program AAPI and CAPI.

OpenVault Server

The OpenVault server accepts requests from applications, and forwards commands to an LCP and DCP, which translate them into low-level robotic and drive control operations to serve that request. OpenVault also schedules competing requests from different applications, creates and enforces cartridge groups for specific application, and maps logical volume names (used by applications) to physical cartridge labels (used by libraries).

The OpenVault server manages cartridges, directing LCP and DCP to mount and unmount a cartridge. Often, cartridges store data. After requesting that a cartridge be mounted, the client application may read and write the media using POSIX standard I/O interfaces. Cartridges can also store audio-video streams for broadcast. In either case, OpenVault is not directly involved in I/O operations.

Client applications, libraries, and drives may be added to a live OpenVault server. The system administrator installs new programs on the appropriate hosts, and issues administrative commands on a live system to inform the OpenVault server that these new programs exist.

ALI/LCP Interface

A library control program (LCP) is a part of OpenVault that deals with low-level details of a removable media library and its configuration and control procedures. There is at least one LCP associated with each OpenVault-managed library.

The OpenVault server issues directives to the LCP in a language called ALI. The LCP replies to the OpenVault server in a language called ALI response (ALI/R).

ADI/DCP Interface

A drive control program (DCP) manages the configuration of drives, and performs the drive control tasks associated with CAPI mount and unmount requests. There is at least one DCP associated with each OpenVault-managed drive.

The OpenVault server issues directives to the DCP in a language called ADI. The DCP replies to the OpenVault server in a language called ADI response (ADI/R).

The *OpenVault Infrastructure Programmers's Guide* describes how to program ALI/LCP and ADI/DCP.

How OpenVault Operates

This section describes how LCP and DCP modules move from boot to operational state.

LCP Booting

When the LCP boots, it reads its configuration file and opens a connection with the OpenVault server. The server decides if the LCP should currently be in control of the library. If so, the OpenVault server tells the LCP that it controls the library. Once control is established, the LCP checks with its library to obtain additional information, such as:

- whether the library is actually of a type supported by this LCP
- whether barcodes or PCLs are supported
- list of cartridge form factors (for example, DLT)
- total number of slots in the library

- total number of occupied slots
- import/export port configuration
- the slotmap (barcode-to-slot-location mapping)
- other information that is relevant to the LCP

The LCP retrieves any stored state or configuration information from the OpenVault catalog (such as the error message log level). The LCP sends the OpenVault server its current slotmap and drive inventory so the catalog can be updated, if necessary. At this point, the library can accept mount and unmount commands from OpenVault.

DCP Booting

DCP booting is similar to LCP booting: the difference is that the LCP has an inventory list and the DCP has a capability list.

When the DCP boots, the DCP also reads its configuration file and opens a connection with the OpenVault server. The server adds the drive into its managed drive list and establishes whether the DCP has sole ownership of the drive or shares it with another DCP. If a drive is simultaneously connected to more than one host, the OpenVault server must decide which DCP (on which host) has control of the drive (the server can transfer control to another connected host on demand). Once the control is established, the DCP checks with its drive to obtain additional information, such as:

- whether the drive is actually of a type supported by this DCP
- the supported media formats (for example, EXABYTE-8mm-5GB)
- whether the listed access modes are supported
- whether a cartridge is loaded in the drive
- verify or acquire any other information that is relevant to the DCP

The DCP retrieves any stored state or configuration information from the OpenVault catalog (such as the error message log level). The DCP passes its capability list to the OpenVault server. The DCP sends the server its capability list so the OpenVault catalog can be updated, if necessary.

Note: OpenVault applications must run on the same host as the OpenVault DCP client attached to the drive(s) employed by that application. This may or may not be the same host as the OpenVault server and LCP client.

OpenVault Installation

Use the *inst* command (or comparable command) to load OpenVault software. You must install *OpenVault.sw.core* and *OpenVault.sw.libs* on the OpenVault server. For ease of administration, install *OpenVault.sw.admin* and *OpenVault.sw.user* on all OpenVault hosts. Install *OpenVault.*.libs* and the appropriate LCP software on each client system attached to an OpenVault-controlled library. Install *OpenVault.*.libs* and the appropriate DCP on each client system attached to an OpenVault-controlled drive.

For more detailed information and recommended procedures for OpenVault installation, see Chapter 2, "Installing OpenVault."

OpenVault Removal

To remove OpenVault software, or individual components, use the *versions* command; see versions(1M). The following command removes all OpenVault software:

```
# versions remove OpenVault.*.*
# rm -rf /usr/OpenVault
```

Cartridge Life Cycle

OpenVault stores and manipulates information on cartridges throughout their life cycles, and includes tools for the administrator to manage and monitor this information. For more information on this topic, see Chapter 3, "Cartridge Life Cycle."

The "life cycle" of a cartridge is the chain of states and events which affect a cartridge from the time that it first becomes part of a system until in ceases to be a part of that system. The major events in the life of a cartridge include:

- the physical and logical introduction of the cartridge into the system
- assignment of ownership (who or what application gets to use the cartridge)
- use of the cartridge by applications
- recycling of a cartridge when one owner no longer needs it
- disposal of the cartridge either by sending it to another system or removing it for disposal when the media reaches the end of its service life

Chapter 2

Installing OpenVault

This chapter describes how to install and set up OpenVault. There are two types of OpenVault hosts: the server, with the media library manager (MLM), and remote hosts, with an LCP or DCP, but without the MLM. Depending on which hosts have drives and libraries physically or logically connected, configurations fall into one of two categories:

- Local configuration—all OpenVault-managed drives, libraries, and applications reside on the OpenVault server host.
- Local-and-remote configuration—one or more libraries or drives (or both) reside on OpenVault hosts other than the OpenVault server host.

If you have already configured OpenVault and would like to change the configuration, see Chapter 4, "Administering OpenVault."

OpenVault Installation Requirements

OpenVault release 1.2 supports the following drives on IRIX versions 6.2 and later:

- DLT-7000 and DLT-2000
- IBM-3590 Magstar
- StorageTek STK-Timberline and STK-Redwood

OpenVault release 1.2 supports the following libraries on IRIX versions 6.2 and later:

- StorageTek STK-9710, STK-9714, and STK-9730
- StorageTek ACSLS controlled libraries.
- IBM-3494
- EMASS-Grau

This release of OpenVault supports barcoded tape media only. You must have at least one tape for each drive type; the norm is to load a tape in each available library slot.

Though not required, if you configure OpenVault using a graphics console, resizing and backward scrolling (in *xwsh* for example) can make setup easier.

Licensing

You must have a license installed to run the OpenVault server software. Licensing tools must be installed beforehand on the OpenVault server system. IRIX 6.2 users need to install the License Tools 2.1.1 or later package. IRIX 6.4 users require License Tools 3.0 or higher. IRIX 6.5 has the required License Tools. To check whether the License Tools are installed, enter the following command:

```
# versions -b license_eoe
```

To obtain a license, visit the Web site http://www.sgi.com/Products/license.html, or send e-mail to license@sgi.com, with a blank message, to obtain the license template. You may also use the FAX number, +1-650-390-0537, or this postal address:

Silicon Graphics, Inc.
Attn: License Administration MS 134
2011 North Shoreline Blvd.
Mountain View, CA 94043-1389.

The OpenVault license you received includes instructions on how to install it. Typically this involves copying text from the license sheet into the */var/flexlm/license.dat* file.

OpenVault Software Components

If you have not already installed OpenVault, you can install it using the *inst* command or the graphical Software Manager.

For more information about IRIX install programs, see the inst(1M) and swmgr(1M) reference pages. For information about last minute product changes, refer to the OpenVault release notes.

The default OpenVault product images include the following subsystems:

```
OpenVault.sw.core            OpenVault core servers
OpenVault.sw.admin           OpenVault administrative tools
OpenVault.sw.libs            OpenVault core runtime libraries
OpenVault.sw.user            OpenVault end-user tools
```

`OpenVault.man.manpages`	`OpenVault manual pages`
`OpenVault.man.relnotes`	`OpenVault release notes`
`OpenVault.dcp.libs`	`OpenVault run-time libraries for DCPs`
`OpenVault.dcp.DLT2000`	`OpenVault DCP for the DLT2000 drive`
`OpenVault.dcp.DLT-7000`	`OpenVault DCP for the DLT-7000 drive`
`OpenVault.dcp.EXB8505XL`	`OpenVault DCP for the EXB-8505 drive`
`OpenVault.dcp.IBM3590`	`OpenVault DCP for the IBM-3590 drive`
`OpenVault.dcp.STKredwood`	`OpenVault DCP for the STK-redwood drive`
`OpenVault.sw.config`	`OpenVault setup scripts`
`OpenVault.sw.core`	`OpenVault core servers`
`OpenVault.sw.admin`	`OpenVault administrative tools`
`OpenVault.sw.libs`	`OpenVault core runtime libraries`
`OpenVault.sw.user`	`OpenVault end-user tools`
`OpenVault.man.manpages`	`OpenVault manual pages`
`OpenVault.man.relnotes`	`OpenVault release notes`
`OpenVault.dcp.libs`	`OpenVault run-time libraries for DCPs`
`OpenVault.dcp.DLT2000`	`OpenVault DCP for DLT2000 drive`
`OpenVault.dcp.DLT-7000`	`OpenVault DCP for DLT-7000 drive`
`OpenVault.dcp.EXB8505XL`	`OpenVault DCP for EXB-8505 drive`
`OpenVault.dcp.IBM3590`	`OpenVault DCP for IBM-3590 drive`
`OpenVault.dcp.STKredwood`	`OpenVault DCP for STK-redwood drive`
`OpenVault.dcp.STKtimberline`	`OpenVault DCP for STK-timberline drive`
`OpenVault.dcp.pseudo`	`OpenVault DCP for the “pseudo” drive`
`OpenVault.dev.examples`	`OpenVault sample source for applications`
`OpenVault.dev.include`	`OpenVault app C/C++ include headers`
`OpenVault.docs.adminguide`	`OpenVault administrative guide`
`OpenVault.docs.architecture`	`OpenVault architecture specifications`
`OpenVault.docs.developer`	`OpenVault LCP/DCP/app developer guides`
`OpenVault.lcp.libs`	`OpenVault run-time libraries for LCPs`
`OpenVault.lcp.ATL2640`	`OpenVault LCP for ATL-2640 library`
`OpenVault.lcp.EMASS_Grau`	`OpenVault LCP for EMASS-Grau library`
`OpenVault.lcp.EXABYTE210`	`OpenVault LCP for EXB-210 library`
`OpenVault.lcp.IBM3494`	`OpenVault LCP for IBM-3494 library`
`OpenVault.lcp.STK9700`	`OpenVault LCP for STK-97XX library` [1]
`OpenVault.lcp.STKACSLS`	`OpenVault LCP for STK-ACSLS library` [2]
`OpenVault.lcp.pseudo`	`OpenVault LCP for the “pseudo” library`

1. This includes the STK-9710, STK-9714 and STK-9730 libraries.
2. Includes libraries that can be controlled via the StorageTek ACSLS interface.

Sample Configurations

Figure 2-1 shows an OpenVault local-only configuration.

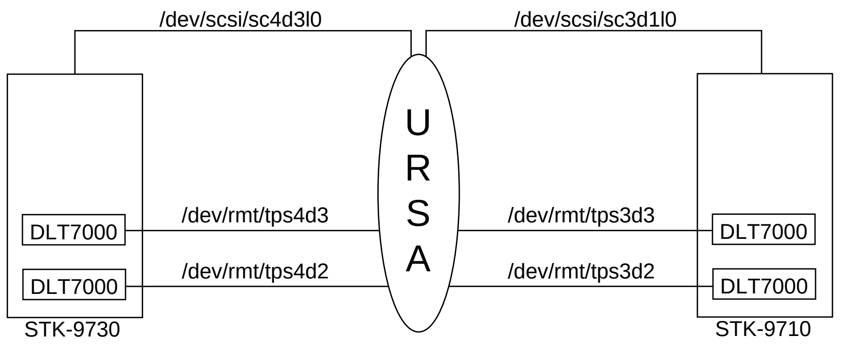

Figure 2-1 Local-Only OpenVault Configuration

In Figure 2-1, the OpenVault server host, ursa, controls two libraries: STK-9730 and STK-9710. The STK-9730 is connected at device address */dev/scsi/sc4d3l0*, while the STK-9710 is connected at device address */dev/scsi/sc3d1l0*.

The STK-9730 contains two drives:

- DLT-7000 connected at address */dev/rmt/tps4d2*, physically located as the bottom drive in the library.
- DLT-7000 connected at address */dev/rmt/tps4d3*, physically located as the top drive in the library.

The STK-9710 contains two drives:

- DLT-7000 connected at address */dev/rmt/tps3d2*, physically located as the bottom drive in the library.
- DLT-7000 connected at address */dev/rmt/tps3d3*, physically located as the top drive in the library.

Figure 2-2 shows an OpenVault local-and-remote configuration.

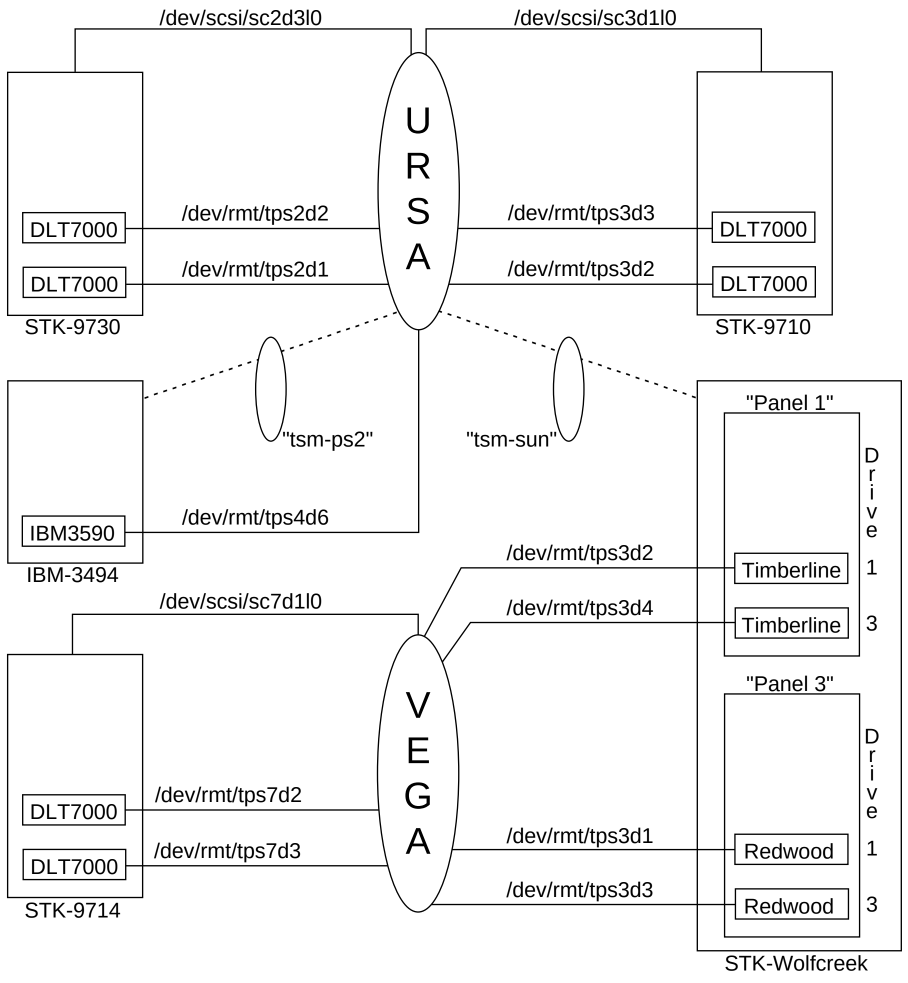

Figure 2-2 Local-and-Remote OpenVault Configuration

In Figure 2-2, ursa is the designated OpenVault server host, with four libraries:

- STK-9730 connected at device address */dev/scsi/sc2d3l0*
- STK-9710 connected at device address */dev/scsi/sc3d1l0*
- IBM-3494 connected via an IBM library server running on the host, tsm-ps2
- StorageTek WolfCreek silo connected via the ACSLS server host, tsm-sun

The STK-9730 on ursa has the following drives, both of which are connected to the host:

- DLT-7000 connected at address */dev/rmt/tps2d1*, physically located as the bottom drive in the library.
- DLT-7000 connected at address */dev/rmt/tps2d2*, physically located as the top drive.

The STK-9710 on ursa has the following drives, both of which are connected to the host:

- DLT-7000 connected at address */dev/rmt/tps3d2*, physically located as the bottom drive in the library.
- DLT-7000 connected at address */dev/rmt/tps3d3*, physically located as the top drive.

The IBM-3494 on host ursa has the following drive, connected to the host ursa:

- IBM-3590 connected at address */dev/rmt/tps4d6*, identified by the IBM Library Server as drive 0

The StorageTek ACSLS-controlled WolfCreek silo has the following drives, all of which are connected to vega, not ursa:

- STK-Redwood connected at address */dev/rmt/tps3d1*, located physically in LSM 2, Panel 3, as Drive 1
- STK-Redwood connected at address */dev/rmt/tps3d3*, located physically in LSM 2, Panel 3, as Drive 3
- STK-Timberline connected at address */dev/rmt/tps3d2*, located physically in LSM 2, Panel 1, as Drive 1
- STK-Timberline connected at address */dev/rmt/tps3d4*, located physically in LSM 2, Panel 1, as Drive 3

Host vega is an OpenVault client host with one library: STK-9714 connected at device address */dev/scsi/sc7d1l0*. The STK-9714 has the following drives, both of which are connected to vega:

- DLT-7000 connected at address */dev/rmt/tps7d3*, located physically as the bottom drive in the library.
- DLT-7000 connected at address */dev/rmt/tps7d2*, located physically as the top drive.

Notice that vega has two STK-Redwood drives and two STK-Timberline drives connected to it, but the library in which these drives reside is not connected to vega.

Configuration Roadmap

The OpenVault initial configuration tool (*setup*) automatically performs steps shown in Table 2-1. The details of each step are explained in the rest of this chapter.

Table 2-1 OpenVault Configuration Roadmap

1.	Prepare OpenVault hosts and devices. Install OpenVault license. Collect information and complete worksheets.	This step is required for all OpenVault hosts.
2.	Configure OpenVault core on the OpenVault server host.	This required step must be performed on the OpenVault server host.
3.	Configure drives attached to the OpenVault server host.	This step is required for drives connected to the OpenVault server host, and must be performed on the OpenVault server host.
4.	Configure libraries attached to the OpenVault server host.	This step is required for libraries connected to the OpenVault server host, and must be performed on the OpenVault server host.
5.	Enable drives and libraries attached to remote OpenVault client hosts.	If you have remote libraries and drives, perform this step on the server host.
6.	Enable applications to run on remote OpenVault client hosts	If you have applications running on remote hosts that request services from OpenVault, perform this step on the server host.
7.	Configure drives attached to OpenVault client hosts.	If you have drives attached to remote OpenVault client host(s), perform this step on the remote host(s).
8.	Configure libraries attached to OpenVault client hosts.	If you have libraries attached to remote OpenVault client host(s), perform this step on the remote host(s).
9.	Import media into OpenVault.	After configuring all libraries, perform this step on the OpenVault server host.
10.	Custom installation.	If desired.

Note: Some steps might not be relevant for your specific scenario.

Preparing OpenVault Devices and Hosts

This section explains how to prepare OpenVault-managed devices for configuration. All drives you configure must be housed in a library, because OpenVault cannot manage standalone drives (that is, drives not housed in a robotic library).

In this procedure, step 4 is required only if you are configuring a IBM-3494 library, and step 5 only if you are configuring a DAS library.

1. Cable all SCSI drives to appropriate hosts. Always ensure proper SCSI termination.
2. Cable all SCSI libraries to appropriate hosts.
3. For each host with an OpenVault-controlled drive attached to it, run the following commands as *root*:

   ```
   # killall mediad
   # chkconfig mediad off
   ```

   Note: The *mediad* daemon interferes with OpenVault operation, so it is essential that you prevent *mediad* from accessing OpenVault drives. Refer to the mediad(1M) man page for information on how to prevent *mediad* from accessing drives.
4. Prepare the IBM-3494 libraries that you are planning to configure:
   - Enable communication between the LCP host and the IBM-3494 server. The IBM-3494 library server runs on the PS/2 system located at the rear of the library. From the graphical administration interface for the library server available on the PS/2, inform the library server of the OpenVault host that will run the LCP for the IBM-3494. Refer to the IBM-3494 Library server documentation for details on how to do this.
   - On the OpenVault host that will run the LCP for the IBM-3494, prepare the */etc/ibmatl.conf* file. If this file does not exist, you must create it. Briefly, this file contains three fields:

     libraryname   *PS/2 hostname*   *LCPhostname*

     Here *libraryname* can be any name not already in this file. The *PS/2 hostname* is the hostname or IP address of the PS/2 system that is running IBM-3494 library server. The *LCPhostname* is the hostname of the machine that will run the LCP for the IBM-3494. Refer to the IBM-3494 documentation for details about the */etc/ibmatl.conf* file.

     Caution: It is critical that you remember the *LCPhostname*, which is also the OpenVault name for this library.

- As root, start *lmcpd* on the IBM-3494 LCP host,

  ```
  # cd /usr/OpenVault/clients/lcp/IBM-3494
  # ./lmcpd
  ```

5. Prepare the DAS PS/2 system for a EMASS-Grau library you plan to configure.

Configuration Worksheet

Use the following worksheets an aid to collecting information that is required for configuring an OpenVault system.

Fill out Table 2-2 with OpenVault server client host information.

Table 2-2 Host Worksheet

OpenVault server host	OpenVault client host(s), if any

Fill out Table 2-3 for drives in a SCSI library.

Table 2-3 Drive Worksheet (SCSI)

Drive device path	Hostname to which drive is connected	OpenVault name for drive	Library in which drive is housed	Drive's physical location (SCSI address)

Fill out Table 2-4 for drives in an IBM-3494 library.

Table 2-4 Drive Worksheet (IBM-3494)

Drive device path	Hostname to which drive is connected	OpenVault name for drive	Library in which drive is housed	Drive #

Fill out Table 2-5 for drives in an ACSLS library.

Table 2-5 Drive Worksheet (ACSLS)

Drive device path	Hostname to which drive is connected	OpenVault name for drive	Library in which drive is housed	LSM ID	Panel ID	Drive #

Fill out Table 2-6 for a SCSI library (use one worksheet per SCSI library).

Table 2-6 Library Worksheet (SCSI)

Library device path	Hostname to which library is attached	OpenVault name for library

Description of drive's physical location within library	OpenVault name for this drive

Fill out Table 2-7 for an IBM-3494 library (use one worksheet per IBM-3494 library).

Table 2-7 Library Worksheet (IBM-3494)

Hostname or IP address of PS/2 controlling the 3494	OpenVault name for library

Drive # as described by mtlib command	OpenVault name for this drive

Fill out Table 2-8 for an ACSLS library (use one worksheet per ACSLS library).

Table 2-8 Library Worksheet (ACSLS)

ACSLS server hostname	ACS ID	ACSAPI packet version	Hostname on which LCP will run	OpenVault name for library

Drive's physical location within library			OpenVault name for this drive
LSM ID	Panel ID	Drive #	

Completing the Worksheet

Follow the order outlined below to record information on the worksheet.

Acquiring Hostnames for OpenVault Hosts

You need to identify all hosts that form your OpenVault system. Identify only one host as the OpenVault server host.

In addition, you need to identify OpenVault client hosts, other than the OpenVault server host. These include any host on which you plan to configure a DCP, LCP, application, or OpenVault administrative commands. Wherever hostnames are required, use the exact output of the *hostname* command.

Generating Unique Names for Libraries

Generate unique names for all OpenVault-managed libraries. You may choose any naming scheme that suits your site, provided the names remain unique. If you prepared an IBM-3494 as described earlier, you must generate a library name, which must become the OpenVault name.

Collecting Information for Attached Drives

Determine which drives are attached to each host using the *hinv* command. From the example configuration in Figure 2-2, the following drives are on hosts ursa and vega:

```
ursa# hinv | grep Tape
Tape drive: unit 1 on SCSI controller 2: DLT
Tape drive: unit 2 on SCSI controller 2: DLT
Tape drive: unit 2 on SCSI controller 3: DLT
Tape drive: unit 3 on SCSI controller 3: DLT
Tape drive: unit 6 on SCSI controller 4: IBM Magstar 3590
```

```
vega# hinv | grep Tape
Tape drive: unit 1 on SCSI controller 3: STK SD3
Tape drive: unit 2 on SCSI controller 3: STK 9490
Tape drive: unit 3 on SCSI controller 3: STK SD3
Tape drive: unit 4 on SCSI controller 3: STK 9490
Tape drive: unit 2 on SCSI controller 7: DLT
Tape drive: unit 3 on SCSI controller 7: DLT
```

For each OpenVault-managed drive in an OpenVault library, follow these steps:

1. Determine to which OpenVault host the drive is physically cabled. Record this information in the drive worksheet. It may be necessary to follow SCSI cabling in order to determine this.
2. Determine the device using the *hinv* command on that host.

   It might be a good idea to insert a tape in the drive, unload all other drives that are on the same SCSI bus as this drive, and issue an *mt status* command.

   For example, to determine this information for one of the two STK-SD3 (STK-Redwood) drives connected to vega, insert a cartridge into drive in question, unload the other drives on SCSI controller 3, and issue *mt -f /dev/rmt/tps3d1 status* and *mt -f /dev/rmt/tps3d3 status* commands. By process of elimination, you arrive at the required information.

3. Record the name of the library in which the device is physically housed.
4. Generate a unique name for this drive and record this name.
5. Record the physical location of drives in the library. For example, in SCSI libraries it is typical to find drives ordered in a linear fashion, either vertically or horizontally as viewed from the front of the library.

   For SCSI libraries, a scheme that describes the location of the drive in relation to the topology of the other drives in the library might suffice. For example, “second drive from bottom” or “third drive from left” are descriptive terms. Typically StorageTek SCSI libraries order drives from the bottom, Exabyte SCSI libraries order drives from the top, while some SpectraLogic libraries order drives from left to right.
6. For drives in an ACSLS-controlled robot, a physical description is not appropriate. You need to query ACSLS for more information. A summary of how you can do this is shown below. Refer to the ACSLS administration guide for details.
   - Log in to the ACSLS server with login *acsss*.
   - Run the command *cmd_proc*.
   - Issue the command *query drive all*.
   - Record the LSM ID, the Panel ID, and the drive # for the relevant drive(s).

   ```
   ACSSA  query drive all
   08-28-98 04:49:16                           Drive Status
   Identifier   State           Status      Volume     Type
   0, 2, 1, 1 online         available              9490
   0, 2, 1, 3 online         available              9490
   0, 2, 3, 1 online         available              SD3
   0, 2, 3, 3 online         available              SD3
   ```

   The comma-separated tokens are the ACS ID, the LSM ID, the panel ID, and the drive number, respectively.

7. For drives in an IBM-3494 library, a physical description is also not appropriate. You need to query the IBM-3494 server for more information. A summary of how to do this is shown below. Refer to the IBM-3494 documentation for details.
   - Prepare the OpenVault host to run the LCP for the IBM-3494 as described in the section, "Collecting Information for IBM-3494 Libraries" on page 26. As *root*, enter these commands:

     ```
     # cd /usr/OpenVault/clients/lcp/IBM-3494
     # ./mtlib -l libraryname -D
     ```

     Here *libraryname* is the name that you generated earlier (step 4 on page 18) when you prepared the */etc/ibmatl.conf* file.

     With the configuration in Figure 2-2, the first output field of this *mtlib* command is a drive number; refer to IBM-3494 documentation concerning how to map drive numbers to physical locations:

     ```
     ursa 9# ./mtlib -l ibm3494 -D
     0, 00141700 003590B1A00
     ```
   - Record the drive number and physical location in the appropriate worksheets.
8. For drives in an EMASS-Grau library, determine locality information similarly.

Collecting Information for SCSI-Attached Libraries

Use this procedure to collect information for SCSI-attached libraries.

1. Record the device control path. On the OpenVault server, use the *hinv* command to determine connected SCSI libraries, and use the output of *hinv* with the *scsicontrol* command to determine the type of each attached SCSI library, as follows:

   ```
   # hinv | grep Juke | \
   awk '/Juke/ {printf "/dev/scsi/sc%dd%dl0\n", $7, $3}' | \
   xargs /usr/sbin/scsicontrol -i
   /dev/scsi/sc2d3l0:  Jukebox       STK     9730            1300
   /dev/scsi/sc3d1l0:  Jukebox       STK     9710            1805
   ```

   For each jukebox that *hinv* displays, issue a *scsicontrol* inquire command in the following form, where X is the controller number from the *hinv* output and Y is the unit number from the *hinv* output:

   ```
   # scicontrol -i /dev/scsi/scXdYl0
   ```
2. Repeat the previous step on every host with an OpenVault-controlled SCSI library.

3. Gather information about drives contained in libraries. For each SCSI library, enter these commands, where *LCPtype* is the LCP name, for example STK-9700:

   ```
   # cd /usr/OpenVault/clients/lcp/LCPtype
   # ./LCP* getlibinfo lcpcontrolpath | grep DRIVE
   ```

   Here *lcpcontrolpath* is the */dev/scsi* control path as described in step 1. In the sample local-only configuration on host ursa (shown in Figure 2-1), enter the following commands for the STK-9730 library:

   ```
   ursa# cd /usr/OpenVault/clients/lcp/STK-9700
   ursa# ./LCP* getlibinfo /dev/scsi/sc2d3l0 | grep DRIVE
   DRIVE  1030           -    - first_drive_from_BOTTOM
   DRIVE  1031           -    - second_drive_from_BOTTOM
   ```

   Use the output information to complete worksheets for SCSI-attached libraries.

Collecting Information for ACSLS Libraries

Use the procedure below to collect information for ACSLS libraries.

1. Record the hostname of the system running the ACSLS server.
2. Record the ACS ID of the ACSLS server that you plan to use. Refer to the documentation for the ACSLS server on how to obtain this ID.
3. Determine the packet version for ACSLS. The packet version is 1 less than the major version number of ACSLS running on the ACSLS server. For example, if you are running ACSLS version 5.1, use 4 as the packet version.
4. Record the drive information. The method for determining drive information is described in “Collecting Information for Attached Drives” on page 23.

Collecting Information for IBM-3494 Libraries

Use the procedure below to collect information for IBM-3494 libraries.

1. Record the hostname of the PS/2 system that is running the IBM-3494 server.
2. Record the drive information. The method for determining drive information is described in “Collecting Information for Attached Drives” on page 23.

Collecting Information for EMASS-Grau Libraries

For drives in an EMASS-Grau library, collect drive information similarly.

Planning Cartridge and Drive Groups

Chapter 3, "Cartridge Life Cycle," introduces the notion of cartridge and drive groups. The OpenVault *setup* script initially creates one cartridge group named *carts*, and one drive group named *drives*.

All media that you import by means of the initial configuration procedure are by default introduced into the *carts* cartridge group. If you would like to create more cartridge groups, to allow importing different media into different cartridge groups, you may do so by following the instructions provided in Chapter 6, "Reconfiguring OpenVault." At initial configuration time, be sure to answer no when asked if you want to "Import Media." The Import Media option is available as part of reconfiguration procedures—after creating the desired cartridge groups, you may import media.

As with Cartridge Groups, at initial configuration time all drives are introduced into the *drives* Drive Group. You may choose to add new drive groups and move drives from one drive group to another at a later time by following instructions provided in Chapter 6, "Reconfiguring OpenVault."

Note: When adding new cartridge and drive groups, remember to enable appropriate applications to use these cartridge groups and drive groups.

Selecting a Password

If your site requires security, it is recommended that you create an OpenVault password. The *setup* script asks you to enter an OpenVault password, for authentication of initially configured applications, drives, and libraries. If you want individual libraries, drives, or applications to use passwords other than the default password that you enter during initial setup, see the instructions in the Chapter 6, "Reconfiguring OpenVault."

Naming Libraries and Drives

The OpenVault setup procedure requires you to identify and provide OpenVault names for libraries and drives, each of which must have a unique OpenVault name. The *setup* script asks you to enter these names in one or more places. It is important that you enter all OpenVault drive and library names consistently and correctly. The configuration worksheets can help you do this.

If a library or drive is connected to the OpenVault **server** host, the OpenVault *setup* script requests the OpenVault name for that physical library or drive. The script offers a generated name as default, which you may override by entering a name you select.

If a library or drive is connected to a remote OpenVault **client** host, the OpenVault *setup* script requests an OpenVault name for that physical library/drive in these places:

- When you configure the OpenVault server host, you must enable each library/drive by providing its OpenVault name. At such time you have the choice of accepting a name generated by the *setup* script, or entering a name you select.
- When you actually configure the library/drive on the OpenVault client host, the script asks you for the OpenVault library/drive name that you entered when you enabled the library/drive on the OpenVault server host.

Be sure to enter the same OpenVault library or drive name in both places!

Note: It is possible that drives are connected to remote OpenVault clients, but housed in a library connected to the OpenVault server host. In this case, configuration of the library requires entry of all drive names—the *setup* script cannot offer defaults. If you wrote drive names for this library on your worksheet, then enter those existing names. If you did not write drive names on your worksheet, you must create them. Be sure to record drive names and enter them exactly later in the configuration process.

Regardless of where a drive is connected, at the time of configuring the library in which a drive is housed, you are asked to enter the OpenVault name of the drive. Be certain to enter same OpenVault drive name in all three places!

Configuring the OpenVault Server

This section describes the steps required to configure the OpenVault server, following the "Configuration Roadmap" on page 17. Depending on the specific configuration at your site, some questions might not apply.

You must configure the OpenVault server before any of its components. Follow these general steps:

1. Log in to the designated OpenVault server as *root*.
2. Ensure that the OpenVault license is installed in the */var/flexlm/license.dat* file.
3. Go to the OpenVault directory and execute the *setup* script:

```
# cd /usr/OpenVault
# ./setup
```

The *setup* script is the main OpenVault configuration tool. It sets up OpenVault based on installed hardware and software, and your input to various questions. If you do not see a choice you want, double check your installation to make sure the items are installed. Where possible, the *setup* script presents a default, indicated by "[*value*]:" at the end of line. You may accept this default by pressing the Enter key. Help is available at many prompts by entering a question mark (**?**).

Setting the OpenVault Server Port

The *setup* script starts by asking for the name of the OpenVault server:

```
What is the name of the OpenVault Server? [ursa]
```

Your answer informs *setup* whether it is executing on the OpenVault server or not. Press Enter if you are configuring the OpenVault server host. The script continues:

```
What port number is OpenVault using? [44444]
```

The default port number for OpenVault communications is 44444. Press Enter if this is acceptable. If anther application uses port number 44444, or if you prefer to select a different port, enter that port number. Make sure that no other application uses this port on the OpenVault server host.

Note: If you select another port number and you have remote OpenVault client host(s), then you must configure them with the same port number as on the server. Failure to do this prevents successful configuration of the client host. The script continues:

```
What default security key would you like to use? [none]
```

The OpenVault server, LCPs, DCPs, and applications are by default configured without a security key, implying no security. If you enter a security key, the OpenVault server uses it to authenticate new connections from a client (an LCP, DCP, or application). If you have already selected a security key (password), enter it now.

You may choose to configure OpenVault initially without security, which simplifies setup. Later, you can establish OpenVault security passwords by following steps in Chapter 6, “Reconfiguring OpenVault.”

Note: If you have remote OpenVault client hosts, you are prompted for the security key value at the time of configuring remote OpenVault client hosts. It is imperative that you enter the same value that you chose while configuring the OpenVault server. Failure to do so prevents successful configuration of the OpenVault client host.

Configuring OpenVault on the OpenVault Server

After you have configured the server port, the *setup* script prompts you with the following options:

```
OpenVault Configuration Menu
Automatic Configuration Option
    31 - Autoconfigure OpenVault on this host
        (setup OpenVault Server)
        (setup OpenVault Devices)
    q  - Exit.
Which operation would you like to do: [31]
```

Accept the default (31) to continue with autoconfiguration of the OpenVault server.

```
Would you like to set up an OpenVault server on this system? [Yes]
```

Use this option for the designated OpenVault server. If you selected the default option of setting up an OpenVault server, you see the following messages displayed, indicating successful configuration of the OpenVault server.

```
Waiting for OpenVault to initialize ...
The OpenVault server was successfully started.
Created Application: ov_umsh
created cartridge group: carts
Cartridge-group-application creation:
            Application: ov_umsh
            Group: carts
created drive group: drives
Drive-group-application creation:
            Application: ov_umsh
            Group: drives
The OV server ‘ursa’ is UP.
The OpenVault server was successfully set up on this system.
```

At this point the following items have been configured:

- The OpenVault core including its catalog and an *ovroot* process ready to service connections from OpenVault clients.
- A default cartridge group named `carts`
- A default drive group named `drives`
- The user mounting application *ov_umsh* has been enabled on the OpenVault server (this host); see ov_umsh(1M) for information.
- The *ov_umsh* application is eligible to use cartridges in the `carts` cartridge group, and drives in the `drives` drive group.

The *setup* script then asks you to configure locally attached drives and libraries.

```
Would you like to configure local LCPs and DCPs? [Yes]
```

It is recommended that you press Enter to proceed with configuration of locally attached drives and libraries at this time.

Configuring Locally Attached Drives on the OpenVault Server

The OpenVault *setup* script scans hardware to determine which SCSI tape drives are attached to this host. If any drives are present, each drive is presented for configuration.

```
Do you want to configure the dtype drive at addr as “tape1”? [Yes]
```

This prompt offers you the choice of configuring the drive of type *dtype* (for example, DLT-7000) at address *addr* (for example, */dev/rmt/tps2d1*).

The *setup* script proposes a name for the drive, “tape*N*” where *N* is 1 in this example. Consult the drive worksheets that you created in “Preparing OpenVault Devices and Hosts” on page 18. Find an entry with “Drive device pathname” equal to the *addr* shown, and “Hostname drive is connected to” equal to the current hostname.

If you find a matching *addr* entry and you wish to use the proposed name, press Enter. Answer “no” under the following conditions:

- If you do not find a matching entry, or do not want to configure this drive at all.
- If you wish to configure this drive at a later time.
- If you prefer a name other than “tape*N*,” look in the worksheet under the column “OpenVault name for drive” and use the name you wrote there.

If you answer “no” to the question, the *setup* script asks you to enter the name of your choice, or to completely bypass configuring this drive.

```
Enter the name for dtype drive at addr, or Enter to bypass this device []
```

To bypass configuration of this drive, press Enter.

If you wish to configure this drive, enter its name from the entry in the drive worksheet. Type the name exactly as it appears in your worksheet. The script proceeds to set up the drive, displaying steps as shown below. The configured DCP is started automatically.

```
Creating drive 9730-bottom-dlt in drive group drives
Created Drive: 9730-bottom-dlt
Updating core keys file for the new drive
Configuring DLT-7000 at /dev/rmt/tps2d1 to be “9730-bottom-dlt”
Starting the DCP for drive 9730-bottom-dlt
```

At this point the following items have been configured:

- The named drive entry has been added to the OpenVault catalog, and the drive has been added to the *drives* drive group.
- An authentication entry has been added to the */usr/OpenVault/core_keys* file.
- The drive's configuration file has been created (*/usr/OpenVault/dcp/dname/config*, where *dname* is the name chosen for this drive).

Repeat the process described in this section for every drive connected to the system. If you want to add or delete drives at a later time, you can configure or deconfigure them by following instructions in Chapter 6, "Reconfiguring OpenVault."

Configuring Locally Attached Libraries on the OpenVault Server

After drive configuration, the *setup* script examines the hardware for attached libraries, and presents each library for configuration.

Configuring SCSI-Attached Libraries

This section describes how to configure SCSI libraries.

```
Do you want to configure the ltype library at laddr as "lib1"? [Yes]
```

This prompt offers you the choice of configuring the drive of type *ltype* (for example, STK-9730) at address *laddr* (for example, */dev/scsi/sc2d3l0*).

The *setup* script proposes a name for the library, "lib*N*" where *N* is 1 in this example. Consult the SCSI library worksheets that you created in "Preparing OpenVault Devices and Hosts" on page 18. Find an entry with "Library device pathname" equal to the *laddr* shown, and "Hostname drive is connected to" equal to the current hostname.

If you find a matching *addr* entry and you wish to use the proposed name, press Enter. Answer "no" under the following conditions:

- If you do not find a matching entry, or do not want to configure this library at all.
- If you wish to configure this SCSI library at a later time.
- If you prefer a name other than "lib*N*," look in the worksheet under the column "OpenVault name for library" and use the name you wrote there.

If you answer "no" to the question, the *setup* script asks you to enter the name of your choice, or to completely bypass configuring this SCSI library.

```
Enter name for ltype library at laddr, or Enter to bypass this device []
```

To bypass configuration of this drive, press Enter. If you wish to configure this library, enter its name from the entry in the SCSI library worksheet. Type the name exactly as it appears in your worksheet.

If you chose to configure the SCSI library, the *setup* script also configures all drives contained in the library. The *setup* script queries the device to determine information about drives housed in this library. This device query might take a while, especially if the host has just been rebooted, if the device has just been power-cycled, or if the main door to the library has just been closed.

At this point you see output similar to the following, where *lname* is the name of the SCSI library that you chose, *laddr* is the SCSI library's device path, and *ltype* is the type of the SCSI library.

```
Creating library lname
Created Library: lname
Updating core keys file for the new library
Configuring ltype at laddr to be lname
```

After querying the information about contained drives, the setup script asks you to enter the OpenVault name for each drive contained in this library, one after the other, where *dlocation* is a text description of the drive's location, and *daddr* is the drive's "slot" address (the slot number that the SCSI robot uses to address the drive).

```
For the drive at location dlocation,
enter a drive name for the element address daddr:
```

You must enter the OpenVault name for the drive at the given physical location within the SCSI library. Refer to the SCSI library worksheet for this library to determine the OpenVault name of this drive. You need to match the description string *dlocation* with the column labeled "Description of drive's physical location within the library." Although your description may not exactly match the description provided by the *setup* script, choose the one that has the same meaning.

Tip: It is critical to the proper functioning of OpenVault that all drive names match up. If you have not already recorded a drive name on your worksheet, enter the correct name now. If you are running *setup* in a scrolling window, scroll up to where you configured this drive, and use the name that was entered there.

The *daddr* (the drive's slot address) is offered as a guide to the expert user. If you are familiar with the SCSI library's addressing scheme you may choose to use this as an unambiguous reference to the drive's physical location within the library.

If you do not want OpenVault to manage this drive, press Enter. The *setup* script bypasses configuring the drive into the library, so OpenVault never mounts into this drive.

Note: If this drive is connected to a remote OpenVault client system, enter the name from your worksheet to configure this drive. If you have not yet recorded a name for this drive, select a unique drive name to enter at the prompt, and record this name on the worksheet. You will be required to enter this name when you later configure drives on the remote OpenVault client system to which the drive is attached.

The script prompts you for OpenVault names for each drive housed in the SCSI library. After you enter the names (or bypass) you see output similar to the following:

```
OpenVault Server host name:     ursa
OpenVault Server port number:   44444
Library name:                   9710
LCP name:                       9710_lcp
Security key:                   none
LCP polling interval:           30
Library control path            /dev/scsi/sc3d1l0
Drives in the Library:          Name    Address
                           9710-bottom-dlt 1030
                           9710-top-dlt    1031
Create it now (Y|N)? []
```

The final line prompts you for confirmation. If you are satisfied with the SCSI library configuration summary shown above, enter **Y**. If you want to change one or more parameters, enter **N** and you are prompted again for the OpenVault drive names.

Note: The *setup* script does not allow you to change the name of the library at this time. However you may do so at a later time by following steps in Chapter 6, “Reconfiguring OpenVault.”

Upon confirmation, the *setup* script proceeds to configure the SCSI library:

```
LCP successfully created
Press enter to continue...
Starting the LCP for library 9710
```

At this point the following items have been configured:

- The named library entry has been added to the OpenVault catalog.
- An authentication entry has been added to the */usr/OpenVault/core_keys* file.
- The library's configuration file is created (*/usr/OpenVault/lcp/lname/config*, where *lname* is the name chosen for this library).

Configuring an IBM-3494 Library

If you have software for the IBM-3494 LCP installed on an OpenVault host, then the *setup* script offers the option of configuring this library, regardless of whether you plan to configure one on this host or not.

```
Do you want to configure the IBM-3494 library at - as "lib1"? [Yes]
```

This prompt offers you a choice of configuring this library with the name *lib1*. The *setup* script proposes a name for the library, "lib*N*" where *N* is 1 in this example. Consult the IBM-3494 library worksheets you created in "Preparing OpenVault Devices and Hosts" on page 18. If you have multiple IBM-3494 libraries to be configured, select one of them.

If you have not yet chosen a name, or can accept the default name, press Enter. (When the library is being configured on an OpenVault client, the *setup* script does not propose a default library name, but instead asks you to enter the name.)

If you do not want to configure this library, answer no. If you have chosen another name for this library (see worksheet "OpenVault name for library") and want to use that name, also answer no. If you answer no, the *setup* script allows you to enter a name, or completely bypass configuring this library (to bypass, press Enter).

```
Enter name for IBM-3494 library at -, or Enter to bypass this device []
```

If you wish to configure this library, enter its name from the IBM-3494 library worksheet. Type the name exactly as it appears in your worksheet. The *setup* script proceeds to configure the drives contained therein, and you see output similar to the following:

```
Creating library ibm3494
Created library ibm3494
Updating core keys file for he new library
Configuring IBM-3494 at - to be “ibm3494”
```

You are then prompted for the hostname or TCP/IP address for the PS/2 that is running the IBM Library Server for this library:

```
What is the hostname or TCP/IP address of the PS/2 inside the library?
```

The *setup* script queries the device to get information about drives housed in this library. This device query might take a while, especially if the host has just been rebooted, if the device has just been power-cycled, or if the main door to the library has just been closed. Soon you see output like this:

```
waiting for lmcpd to become active ...
```

After querying information about contained drives, the *setup* script asks you to enter the OpenVault name for each drives in this library, one after the other.

```
For the drive at location “-”
Enter a drive name for the element address daddr:  3590
```

Here *daddr* is the drive’s address as reported by the IBM-3494 Library server, usually a number starting from 0. Enter the OpenVault name for the drive at the given address. Refer to the IBM-3494 library worksheet to determine the OpenVault name for this drive. You must match drive address *daddr* with the column “Drive # number as described by the mtlib command.” See “Collecting Information for Attached Drives” on page 23 for details on how to run the *mtlib* command.

Tip: It is critical to the proper functioning of OpenVault that all drive names match up. If you have not already recorded a drive name on your worksheet, enter the correct name now. If you are running *setup* in a scrolling window, scroll up to where you configured this drive, and use the name that was entered there.

If you do not want OpenVault to manage this drive, press Enter. The script bypasses configuring the drive in the library, so OpenVault does not mount cartridges in this drive.

Note: If this drive is connected to a remote OpenVault client system, enter the name from your worksheet to configure this drive. If you have not yet recorded a name for this drive, select a unique drive name to enter at the prompt, and record this name on the worksheet. You will be required to enter this name when you later configure drives on the remote OpenVault client system to which the drive is attached.

The script prompts you for OpenVault names for each drive housed in the library. After you enter names of all drives (or bypass) you see output similar to the following:

```
OpenVault Server host name:     ursa
OpenVault Server port number:   44444
Library name:                   ibm3494
LCP name:                       ibm3494_lcp
TCP/IP Address of the Library:  tsm-ps2
Security key:                   none
LCP polling interval:           30
Drives in the Library:          Name    Address
                                3590    0
Create it now (Y|N)?  []
```

The final line prompts you for confirmation. If you are satisfied with the library and drive configuration shown, enter **Y**. If you want to change one or more parameters, enter **N** and you are prompted again for the OpenVault drive names. Upon confirmation, the *setup* script proceeds to configure the library:

```
LCP successfully created
Press enter to continue...
Starting the LCP for library ibm3494
```

Note: The *setup* script cannot change the name of the library at this time. However you may do so at a later time by following steps in Chapter 6, “Reconfiguring OpenVault.”

At this point the following items have been configured:

- The named library entry has been added to the OpenVault catalog.
- An authentication entry has been added to the */usr/OpenVault/core_keys* file.
- The library’s configuration file is created (*/usr/OpenVault/lcp/lname/config*, where *lname* is the name chosen for this library).

Configuring a StorageTek ACSLS library

If you have LCP software for the StorageTek ACSLS installed on an OpenVault host, the *setup* script offers the option of configuring this library, regardless of whether you plan to configure one on this host or not.

```
Do you want to configure the STK-ACSLS library at - as "lib1"? [Yes]
```

This prompt offers you a choice of configuring this library with the name *lib1*. The *setup* script proposes a name for the library, "lib*N*" where *N* is 1 in this example. Consult the library worksheets you created in "Preparing OpenVault Devices and Hosts" on page 18. If you have multiple STK-ACSLS libraries to configure on this host, select one of them.

If you have not yet chosen a name, or can accept the default name, press Enter. (When the library is being configured on an OpenVault client, the *setup* script does not propose a default library name, but instead asks you to enter the name.)

If you do not want to configure this library, answer no. If you have chosen another name for this library (see worksheet "OpenVault name for library") and want to use that name, also answer no.

If you answer no, the *setup* script allows you to enter a name, or completely bypass configuring this library (to bypass, press Enter).

```
Enter name for STK-ACSLS library at - or Enter to bypass this device []
```

If you wish to configure this library, enter its name from the entry in the STK-ACSLS library worksheet. Type the name exactly as it appears in your worksheet.

If you chose to configure the library, the *setup* script proceeds to configure the drives contained therein, and you see output similar to the following:

```
Creating library wolfcreek
Created library wolfcreek
Updating core keys file for he new library
Configuring STK-ACSLS at - to be "wolfcreek"
```

You are then prompted for the hostname of the ACSLS server for this library:

```
What is the hostname of the ACSLS server? []
```

After entering this hostname, enter the version number of the ACSLS interface. Identify the version of ACSLS installed on the ACSLS server, subtract 1 from this number, and enter that value. See "Collecting Information for ACSLS Libraries" on page 26 for details.

```
What is the version number of the ACSLS interface? [4]
```

Next enter the ID of the ACS that you are using for this LCP. Refer to the vendor's library documentation for information about obtaining the ACS ID.

```
What is the ACS ID of ACS? [0]
```

The *setup* script queries the device to determine information about drives housed in this library. This device query might take a while, especially if the host has just been rebooted, if the device has just been power-cycled, or if the main door to the library has just been closed. Soon you see output similar to the following:

```
ONC RPC: csi_init(): Initiation Started
ONC RPC: csi_init(): Initiation Completed
Acquiring drive information from library (may take a while) ...
ONC RPC: csi_init(): Termination Started
ONC RPC: csi_init(): Termination Completed
```

Upon querying the information about contained drives, the *setup* script asks you to enter the OpenVault name for each drives in this library, one after the other.

```
For the drive at location “LSMid 2, Panelid 1,Driveid1”
Enter a drive name for the element address 2,1,1: timberline1
```

The location string displayed is the LSM ID, Panel ID, and Drive number for the drive. See “Collecting Information for Attached Drives” on page 23 for steps to obtain information for drives in an ACSLS library. Consult the ACSLS library worksheet for this library and select the drive entry with the corresponding LSM ID, Panel ID, and Drive number. Enter the OpenVault name you selected.

Tip: It is critical to the proper functioning of OpenVault that all drive names match up. If you have not already recorded a drive name on your worksheet, enter the correct name now. If you are running *setup* in a scrolling window, scroll up to where you configured this drive, and use the name that was entered there.

If you do not want OpenVault to manage this drive, press Enter. The script bypasses configuring the drive in the library, so OpenVault does not mount cartridges in this drive.

Note: If this drive is connected to a remote OpenVault client system, enter the name from your worksheet to configure this drive. If you have not yet recorded a name for this drive, select a unique drive name to enter at the prompt, and modify your worksheet. You will be required to enter this name when you later configure drives on the remote OpenVault client system to which the drive is attached.

The script prompts you for OpenVault names for each drive housed in the library. After you enter names of all drives (or bypass) you see output similar to the following:

```
OpenVault Server host name:     ursa
OpenVault Server port number:   44444
Library name:                   wolfcreek
LCP name:                       wolfcreek_lcp
Security key:                   none
ACSLS Server Host name:         tsm-sun
ACSLS Server Version:           4
ACSLS ACS ID:                   0
LCP polling interval:           30
Drives in the Library:   Name        LSM     Panel    Drive
                         redwood1      2       3        1
                         redwood3      2       3        3
                         timberline1   2       1        1
                         timberline3   2       1        3
Create it now (Y|N)?  []
```

The final line prompts you for confirmation. If you are satisfied with the library and drive configuration shown, enter **Y**. If you want to change one or more parameters, enter **N** and you are prompted again for the OpenVault drive names.

Note: The *setup* script does not allow you to change the name of the library at this time. However you may do so at a later time by following steps in Chapter 6, "Reconfiguring OpenVault."

Upon confirmation, the *setup* script proceeds to configure the library:

```
LCP successfully created
Press enter to continue...
Starting the LCP for library stk-acsls
```

At this point the following items have been configured:

- The named library entry has been added to the OpenVault catalog.
- An authentication entry has been added to the */usr/OpenVault/core_keys* file.
- The library's configuration file is created (*/usr/OpenVault/lcp/lname/config*, where *lname* is the name chosen for this library).

Configuring an EMASS-Grau library

Configure an EMASS-Grau library in a way similar to configuring an IBM-3494 library.

Enabling Remote LCPs, DCPs, and Administration

This optional configuration step enables the following features:

- administration of the OpenVault server from remote OpenVault clients
- LCPs and DCPs that connect to the OpenVault server from OpenVault clients

This step enables remote connections. You must also run the *setup* script on the remote OpenVault client(s) to configure libraries and drives on client host(s).

```
Will remote OpenVault admin/LCP/DCP clients access this server? [No]
```

If you plan to enable one or more of these options, enter **`Yes`**; otherwise, accept the `No` default. You can add remote libraries, drives, administrative commands, or applications at a later time; see Chapter 6, "Reconfiguring OpenVault."

The *setup* script first prompts for the name of a remote OpenVault client host, which is system that meets one or more of the following tests:

- You plan to connect libraries and/or drives to that host.
- You plan to administer the OpenVault server from that host, by installing the OpenVault administrative subsystem.
- You plan to run applications on the host that request services from OpenVault.

To identify remote client hostnames, consult the Host worksheet that you generated in "Preparing OpenVault Devices and Hosts" on page 18. Enter the name of a remote OpenVault client host at the following prompt:

```
Enter a remote OpenVault client machine name, or Enter to continue: []
```

Enter the name of the remote client host. When you have entered all the hosts, press Enter to continue with configuration.

Enabling Remote Applications

After enabling remote clients, the *setup* script displays the following prompt. If you want to administer OpenVault from the named host, accept the default; otherwise enter **`No`**.

```
Will machine vega require administrative access to this server? [Yes]
```

Enabling Remote Libraries

After this, the *setup* script prompts for remote libraries. If libraries are connected to the named remote host, enter **`Yes`**; otherwise accept the default.

```
Are any libraries controlled by LCPs on vega? [No]
```

If you answered yes to the previous question, the *setup* script asks for OpenVault names of libraries on this host:

```
Do you want to configure a library on vega as “lib1”? [Yes]
```

The *setup* script presents a library name, which you may accept. If you have chosen library names in the configuration worksheet, enter those names. The script continues prompting for remote OpenVault library names. Press Enter when you have no more remote library names to enter.

Tip: Use the worksheets to record the names you select—you will be asked to enter them again when configuring the library or drives on a remote OpenVault client host.

Enabling Remote Drives

After enabling remote libraries, the *setup* script prompts for remote drives. If you plan to have drives connected to the named remote host, enter **`Yes`**; otherwise accept the default.

```
Are any drives controlled by DCPs on vega? [No]
```

If you answered yes to the previous question, the *setup* script asks for OpenVault names of drives on this host:

```
Do you want to configure a drive on vega as “drive1”? [Yes]
```

The *setup* script presents a drive name, which you may accept. If you have chosen drive names in the configuration worksheet, enter those names. The script continues prompting for remote OpenVault drive names. Press Enter when you have no more remote drive names to enter.

Importing Media—When?

After you configure locally attached drives and libraries, the script asks to import tapes that were discovered in the newly configured libraries. If you have libraries on remote OpenVault client hosts, wait until those libraries are configured, then invoke the import function as described in the section "Importing Media" on page 47. If you do not have any remote libraries, you may import media right away.

Configuring the OpenVault Clients

After configuring the OpenVault server, it is time to configure remote OpenVault clients, in any order. Consult the Hosts worksheet that you generated in "Preparing OpenVault Devices and Hosts" on page 18 for the list of OpenVault client hosts to configure.

To configure an OpenVault client host, follow these steps:

1. Log in to the remote host as *root*.
2. Change to the OpenVault directory, and start the *setup* script:

```
# cd /usr/OpenVault
# ./setup
```

The *setup* script outlines the general configuration strategy:

```
OpenVault Configuration
  The general strategy for setting up OpenVault is to
    1) configure the OpenVault server
    2) configure LCP/DCP's on the server machine
    3) configure server for local Applications
    4) if needed, configure server for remote LCP's, DCP's, and Apps
    5) if needed, install and configure LCP/DCP's on remote machines
    6) from the server, setup/import media for each library
```

The *setup* script then determines the name of the OpenVault server host, the OpenVault server port number, and the OpenVault security key that you chose while configuring the OpenVault server.

```
What is the name of the OpenVault Server? [vega]
```

Enter the name of the OpenVault server host. The default presented in this prompt is the hostname of the machine on which you are running *setup* script.

```
What is the port number OpenVault is using? [44444]
```

If you chose another port number when you configured the OpenVault server, enter that port number now; otherwise, accept the default.

```
What default security key would you like to use? [none]
```

Enter the security key that you chose when you configured the OpenVault server. If you did not select security at that time, accept the default.

Note: Specifying the exact values for the OpenVault server's hostname, port number, and security key is critical for proper functioning of all OpenVault components.

The *setup* script then provides the menu options shown below:

```
OpenVault Configuration Menu
  Configuration for Machines Running LCPs
    11 - Configure a new client LCP
  Configuration for Machines Running DCPs
    21 - Configure a new client DCP
  Automatic Configuration Option
    31 - Autoconfigure OpenVault on this host
  (setup OpenVault Devices)
    q  - Exit.
Which operation would you like to do: [31]
```

Tip: When first configuring an OpenVault client host, it is recommended that you select the Autoconfigure option (option 31).

Option 31 detects drives and libraries attached to this host. Selecting other options allows you to configure LCPs and DCPs, but those procedures are not automatic.

```
Would you like to configure local LCPs and DCPs? [Yes]
```

If you are ready to configure the libraries and drives on this host, accept the default. Configure all libraries and drives that you plan to attach to this host.

Configuring Attached Drives on OpenVault Client Hosts

If you have any drives attached to this host, the *setup* script prompts for each attached drive in succession. The set of dialogs presented are similar to the dialog when you configure drives on the OpenVault server. The main difference is that the script asks you to enter the OpenVault name for the drive, but does not offer a default name.

It is critical that you enter the OpenVault names for each drive exactly as you entered them (during server configuration when enabling remote drives). Consult your drive worksheet to make certain that the names match. A sample dialog is shown below:

```
Enter the name for the STK-redwood drive at /dev/rmt/tps3d1,
or Enter to bypass this device [] redwood1
Configuring STK-redwood at /dev/rmt/tps3d1 to be “redwood1”
                Starting the DCP for drive redwood1
```

At this point, the offered drive has been configured and the DCP for this drive is started. Use the *ov_stat* command on the server to show status of the newly configured drive.

Configuring Attached Libraries on OpenVault Client Hosts

The script scans the installed hardware and software to determine which libraries are available for configuration. Each detected library is offered for configuration. The set of dialogs presented are similar to the one when you configure libraries on the OpenVault server. The main difference is that the script asks you to enter the OpenVault name for the library, but does not offer a default name.

It is critical that you enter the OpenVault name for each library exactly as you entered it (during server configuration when enabling remote libraries). Consult your library worksheet to make certain that the names match. A sample dialog for a SCSI-attached library is shown below.

```
Enter the name for the STK-9714 library at /dev/scsi/sc7d1l0, or Enter
 to bypass this device [] 9714
Configuring STK-9700 at /dev/scsi/sc7d1l0 to be “9714”
For the drive at location “first drive from BOTTOM”,
 enter a drive name for the element address 1030: 9714-bottom-dlt
For the drive at location “second drive from BOTTOM”,
 enter a drive name for the element address 1031: 9714-top-dlt
LCP Creation Parameters Confirmation
OpenVault Server host name:     ursa
OpenVault Server port number:   44444
Library name:                   9714
LCP name:                       9714_lcp
Security key:                   none
LCP polling interval:           30
Library control path            /dev/scsi/sc7d1l0
Drives in the Library:          Name    Address
                                9714-bottom-dlt 1030
                                9714-top-dlt    1031
```

```
Create it now (Y|N)? [] Y
LCP successfully created
Press enter to continue...
Starting the LCP for library 9714
```

If you decide that you do not want to configure a library at this time, press Enter to bypass configuring this library.

The *setup* script prompts you for OpenVault drive names contained in the library. Type names exactly as you did when enabling those drives on the OpenVault server. Consult your library worksheet for details. You must match the location description string for each contained drive with the corresponding description string in the Library worksheet.

If you have drives in this library that are connected to another host, enter the exact OpenVault name for these drives, even if you have not yet configured the drives on that host. Consult your library worksheet for this library to get this information.

Configuring a non SCSI-attached library on an OpenVault client host follows the same procedure as configuring one on the OpenVault server host.

At this point the library is configured and the LCP is started. To verify current status of the LCP, run the *ov_stat* command on the OpenVault server.

Importing Media

After you have configured the libraries on the OpenVault server and all OpenVault clients, you need to import media to make it available for application use. Importing media is how the OpenVault server learns about each piece of media. Each tape that applications use must be imported before it can be made available for allocations and mounts. See Chapter 3, "Cartridge Life Cycle" for more information.

Caution: If you have media that contain data in a library, or if media are known to certain sensitive applications, take precautions so that other applications do not accidentally modify these media. Refer to the application's documentation for importing media known to that application only. **Failure to do so could lead to data loss.**

It is best to invoke the import media function of the *setup* script only after you have configured all libraries, including libraries on remote OpenVault client hosts.

The import function is available only on the OpenVault server host. To import media into a library, you must identify a cartridge group for each cartridge. The *setup* script imports all media found in a library into the same cartridge group. By default, media are imported into the default cartridge group called *carts*. If you would like to add more cartridge groups, do so now. To import media using multiple cartridge groups, see Chapter 6, "Reconfiguring OpenVault."

You must also identify a cartridge type for each cartridge; a list is shown in "Selecting Cartridge Types" on page 49. The *setup* script assumes that all media in a library is of the same cartridge type. If this is not true, skip automatic import of media and follow the procedures described in Chapter 3, "Cartridge Life Cycle."

The import function allows you to pre-allocate all media to an application. If you wish to import media that is pre-allocated to other applications, skip automatic import of media and follow the procedures described in Chapter 3, "Cartridge Life Cycle."

You may import media that is not pre-allocated. If you do, make sure that OpenVault applications know how to import media. For example, the media mounting application *ov_umsh* provided with OpenVault knows how to allocate media.

To use the import function, enter these commands as *root* on the OpenVault server host, and select the "Import Media" option 3, as shown below:

```
# cd /usr/OpenVault
# ./setup

OpenVault Configuration Menu
  Configuration for the OpenVault Server Machine
    1 - Manage Libraries, Drives and Applications
    2 - Manage Cartridge Groups And Drive Groups
    3 - Import Media
    4 - Enable remote applications
  Configuration for Machines Running LCPs
    11 - Configure a new client LCP
    12 - Modify an existing client LCP
    13 - Deconfigure an existing client LCP
  Configuration for Machines Running DCPs
    21 - Configure a new client DCP
    22 - Modify an existing client DCP
    23 - Deconfigure an existing client DCP

    q  - Exit.

Which operation would you like to do: 3
Would you like to import media ? [Yes]
```

The *setup* script checks for all configured libraries containing media to import, and shows each of the libraries individually.

```
Would you like import media from the library 9710 [Yes]
```

Accept the default if you are ready to import media in that library; otherwise, enter **`No`**. The script continues:

```
Would you like to add ALL cartridge to the SAME Cartridge group [Yes]
```

If you decide to import all cartridges in this library into the same cartridge group, accept the default. Otherwise, enter **`No`** to have the script skip importing media from this library.

Selecting Cartridge Types

Available cartridge types are presented (supported types are denoted with a star):

```
1.      D2-L
2.      D2-M
3.      D2-S
4.      DDS3
5.      DDS2
6.      DDS1
7.      STKredwood *
8.      STKtimberline *
9.      3590 *
10.     3490E *
11.     3490 *
12.     3480 *
13.     8mm-160m
14.     mammoth
15.     DLT-7000 *
16.     DLT2000 *
17.     DTF
99.     MIXED_MEDIA
Select the Cartridge type for the cartridges in this library:
```

Once you select the cartridge type, available cartridge groups are displayed. Enter the cartridge group of which all cartridges in this library should be members.

```
Cartridge Groups available are:
                      1. carts
Select the Cartridge group for the cartridges in this library:
```

Once you select a cartridge group, the script asks about pre-allocation. If you want to pre-allocate cartridges to an application, accept the default. If not, see "Not Pre-allocating Cartridges" on page 50.

```
Would you like to pre-allocate cartridges to specific application [Yes]
```

If you answer yes, you are prompted for an application name, *dmf* in this example:

```
Enter name of application for cartridge pre-allocation [ov_umsh] dmf
```

If the application name you enter does not exist, the *setup* script creates it for you. Once the import completes successfully, you see output similar to the following:

```
Created Application: dmf
Cartridge-group-application creation:
                     Application: dmf
                     Group: carts
                     Importing tapes with:
                         Cartridge Type = DLT2000
                         Cartridge Group = carts
                         Application = dmf
This may take a while...
Finished importing media from library, 9710
```

Not Pre-allocating Cartridges

If you do not wish to pre-allocate cartridges, enter **`No`** after the following prompt:

```
Would you like to pre-allocate cartridges to specific application [Yes]
```

Cartridges in this library are imported without being pre-allocated to any application. Once the import completes successfully, you see output similar to the following:

```
Importing tapes with:
          Cartridge Type = DLT-7000
          Cartridge Group = carts
This may take a while ...
Finished importing media from library, 9730
```

Custom Installation

Before doing a custom install, you must determine the types of all OpenVault-managed libraries and drives on all systems.

- For drives, see "Determining Attached SCSI Drives" on page 51.
- For SCSI libraries, see "Determining Attached SCSI Libraries" on page 52.
- For other libraries, see "Non SCSI-Attached Libraries" on page 52.

Remember to repeat the steps explained in this section for every host with an attached library or drive that is to be managed by OpenVault.

Determining Attached SCSI Drives

Use the *hinv* command to determine connected SCSI drives, and use the output of *hinv* with the *scsicontrol* command to determine the type of each attached SCSI drive:

```
# hinv | grep Tape | \
awk '/Tape/ {printf "/dev/scsi/sc%dd%dl0\n", $8, $4}' | \
xargs /usr/sbin/scsicontrol -i
/dev/scsi/sc2d1l0:  Tape         QUANTUM DLT-7000         1E46
/dev/scsi/sc2d2l0:  Tape         QUANTUM DLT-7000         1E46
/dev/scsi/sc3d2l0:  Tape         QUANTUM DLT-7000         1E46
/dev/scsi/sc3d3l0:  Tape         QUANTUM DLT-7000         1E46
/dev/scsi/sc4d6l0:  Tape         IBM     03590B1A        53F2
```

For each drive that *hinv* displays, issue a *scsicontrol* inquire command in the following form, where X is the controller number from the *hinv* output and Y is the unit number from the *hinv* output:

```
# scsicontrol -i /dev/scsi/scXdYl0
```

```
vendor          product         revision
----------------------------------------
QUANTUM         DLT-7000         1E46
QUANTUM         DLT-7000         1E46
QUANTUM         DLT-7000         1E46
QUANTUM         DLT-7000         1E46
IBM             03590B1A        53F2
```

Each line in the *scsicontrol* output shows the drive's control path, the drive vendor name, the drive product name, and its firmware revision number.

Determining Attached SCSI Libraries

Use the *hinv* command to determine connected SCSI libraries, and use the output of *hinv* with the *scsicontrol* command to determine the type of each attached SCSI library:

```
# hinv | grep Juke | \
awk ’/Juke/ {printf "/dev/scsi/sc%dd%dl0\n", $7, $3}’ | \
xargs /usr/sbin/scsicontrol -i
/dev/scsi/sc2d3l0:  Jukebox       STK     9730           1300
/dev/scsi/sc3d1l0:  Jukebox       STK     9710           1805
```

For each jukebox that *hinv* displays, issue a *scsicontrol* inquire command of the following form, where X is the controller number from the *hinv* output and Y is the unit number from the *hinv* output:

```
# scicontrol -i /dev/scsi/scXdYl0
```

```
vendor             product         revision
----------------------------------------
STK                9730            1300
STK                9710            1805
```

Each line in the *scsicontrol* output shows the library’s control path, the library vendor name, the library product name, and its firmware revision number.

Non SCSI-Attached Libraries

OpenVault supports several non-SCSI libraries. If you plan to manage this type of library, you must install the appropriate *OpenVault.lcp* subsystems.

For any library that you plan to manage with OpenVault, configure one and exactly one controlling LCP. Because the above-mentioned libraries are not SCSI-attached, you may designate any OpenVault host system to run the LCP. You must install the appropriate *OpenVault.lcp* subsystem on that chosen system. For simplification, it might be best to designate the OpenVault server as the LCP host for non-SCSI-attached libraries.

Note: If you unintentionally install the LCP for a non-SCSI-attached LCP on an OpenVault host, the OpenVault *setup* script asks if you would like to configure that LCP. Answer this question by entering **No**.

Other Guidelines for Custom Installation

This section offers guidelines for selecting OpenVault components to install.

- For OpenVault server hosts:
  - Required

```
OpenVault.sw.config         OpenVault setup scripts
OpenVault.sw.core           OpenVault core servers
OpenVault.sw.admin          OpenVault administrative tools
OpenVault.sw.libs           OpenVault core run-time libraries
```

  - Recommended

```
OpenVault.man.manpages      OpenVault manual pages
OpenVault.man.relnotes      OpenVault release notes
OpenVault.docs.adminguide   OpenVault Administrator’s Guide
```

  - Optional

```
OpenVault.sw.user           OpenVault end-user tools
OpenVault.dev.examples      OpenVault sample source for applications
OpenVault.dev.include       OpenVault app C/C++ include headers
OpenVault.docs.architecture OpenVault architecture specifications
OpenVault.docs.developer    OpenVault LCP/DCP/app developer guides
```

  - As needed

```
OpenVault.dcp.libs          OpenVault run-time libraries for DCPs
OpenVault.dcp.XXXX          Appropriate DCP susbsystem
OpenVault.lcp.libs          OpenVault run-time libraries for LCPs
OpenVault.lcp.XXXX          Appropriate LCP susbsystem
```

- For OpenVault client hosts:
  - Required

```
OpenVault.sw.config         OpenVault setup scripts
OpenVault.sw.libs           OpenVault core run-time libraries
```

  - Not recommended

```
OpenVault.sw.core           OpenVault core servers
```

  - Recommended

```
OpenVault.man.manpages      OpenVault manual pages
OpenVault.man.relnotes      OpenVault release notes
OpenVault.sw.admin          OpenVault administrative tools
OpenVault.docs.adminguide   OpenVault Administrator’s Guide
```

– Optional

```
OpenVault.sw.user           OpenVault end-user tools
OpenVault.dev.examples      OpenVault sample source for applications
OpenVault.dev.include       OpenVault app C/C++ include headers
OpenVault.docs.architecture OpenVault architecture specifications
OpenVault.docs.developer    OpenVault LCP/DCP/app developer guides
```

– As needed

```
OpenVault.dcp.libs          OpenVault run-time libraries for DCPs
OpenVault.dcp.XXXX          Appropriate DCP susbsystem
OpenVault.lcp.libs          OpenVault run-time libraries for LCPs
OpenVault.lcp.XXXX          Appropriate LCP susbsystem
```

The *OpenVault.dcp.libs* and *OpenVault.sw.libs* subsystems are required on each OpenVault host that contains an OpenVault DCP.

The *OpenVault.lcp.libs* and *OpenVault.sw.libs* subsystems are required on each OpenVault host that contains an OpenVault LCP.

The *OpenVault.sw.libs* subsystem is required on each OpenVault host that contains an OpenVault application, DCP, or LCP.

The installation tool enforces these requirements.

Chapter 3

Cartridge Life Cycle

This chapter describes the OpenVault cartridge life cycle and the administrative tools used to manage it. More information on the administrative tools is available in the reference page for each command.

The "life cycle" of a cartridge is the chain of states and events that affect a cartridge from the time that it first becomes part of a system until in ceases to be a part of that system. The major events in the life of a cartridge include:

- the physical and logical introduction of the cartridge into the system
- assignment of ownership (who or what application gets to use the cartridge)
- use of the cartridge by applications
- recycling of a cartridge when one owner no longer needs it
- disposal of the cartridge either by sending it to another system or removing it for disposal when the media reaches the end of its service life

Cartridge States

During the life cycle shown in Figure 3-1, these are the possible cartridge states:

unknown	If no information is available about a cartridge, then its state is "unknown." A new cartridge (one that OpenVault has not seen before) is in the unknown state. OpenVault needs to become aware of a cartridge before it can do anything with it.
defined	OpenVault needs to have certain information about a cartridge before it can be used. The majority of the information must come from a person—the OpenVault administrator—who knows what the cartridge is and can describe it to OpenVault. This includes the physical cartridge label (PCL, usually an optically readable label on the cartridge), the cartridge type, and information on the number of sides on the cartridge. All this information constitutes the cartridge definition.

available	In addition to knowing that a cartridge exists, OpenVault needs to know where information can be written on the cartridge, and whether an application has reserved that area. This information comprises the partition description for the cartridge. Once this information is entered the cartridge may be considered "available." That is, the cartridge may be assigned ("allocated") to an application that needs one.
allocated	For applications, OpenVault is a service for allocating and mounting volumes. Applications make a request to write data in a known place and manner. That place is a logical volume on a partition on a cartridge. The application asks the server to allocate a volume for it. The server looks for cartridges with available partitions that are not used by any application. When it finds one, the server modifies cartridge ownership information to show that the requesting application has control over that cartridge. It marks the partition as unavailable, and creates volume information to describe where data are stored on the cartridge. At this point the cartridge is considered "allocated" and the volume can be used as long as the application desires to do so.
deallocated	When an application is finished with cartridge, it can "deallocate" it. The server removes volume information for the cartridge, so the application can no longer request mounting of that volume. Information on data location is deleted and becomes unavailable. The application still owns the cartridge, however, giving the administrator some control over how to handle the cartridge thereafter. The "owned and deallocated" state can be used as a marker for a cartridge to be erased, destroyed (if the media is at the end of its useful life), or recycled without erasure. An administrator (or an administrative application) can mark the cartridge as available again without doing anything else to it. The cartridge becomes available for volume allocation, but only to the application that owns it (ownership remains as before). The cartridge must be recycled before other applications can use it.
recycled	An administrator (or an administrative application) may recycle deallocated cartridges. This entails marking the cartridge as available for allocation and removing ownership information. Once the cartridge is recycled, it is again in the "defined" and "available" state, but other information about the cartridge may be retained. This could include a count of the number of times that cartridge has been written so that its remaining life expectancy can be tracked.

purged — All information about a cartridge can be removed from the system, returning the cartridge to “unknown” state. This is typically done only when a cartridge has permanently left the OpenVault system, as when a cartridge has reached the end of its life, or when it is removed from the system for transfer elsewhere. There is no “purged” state as such; this term simply provides a way of referring to cartridges for which there once was descriptive data, all of which has been removed.

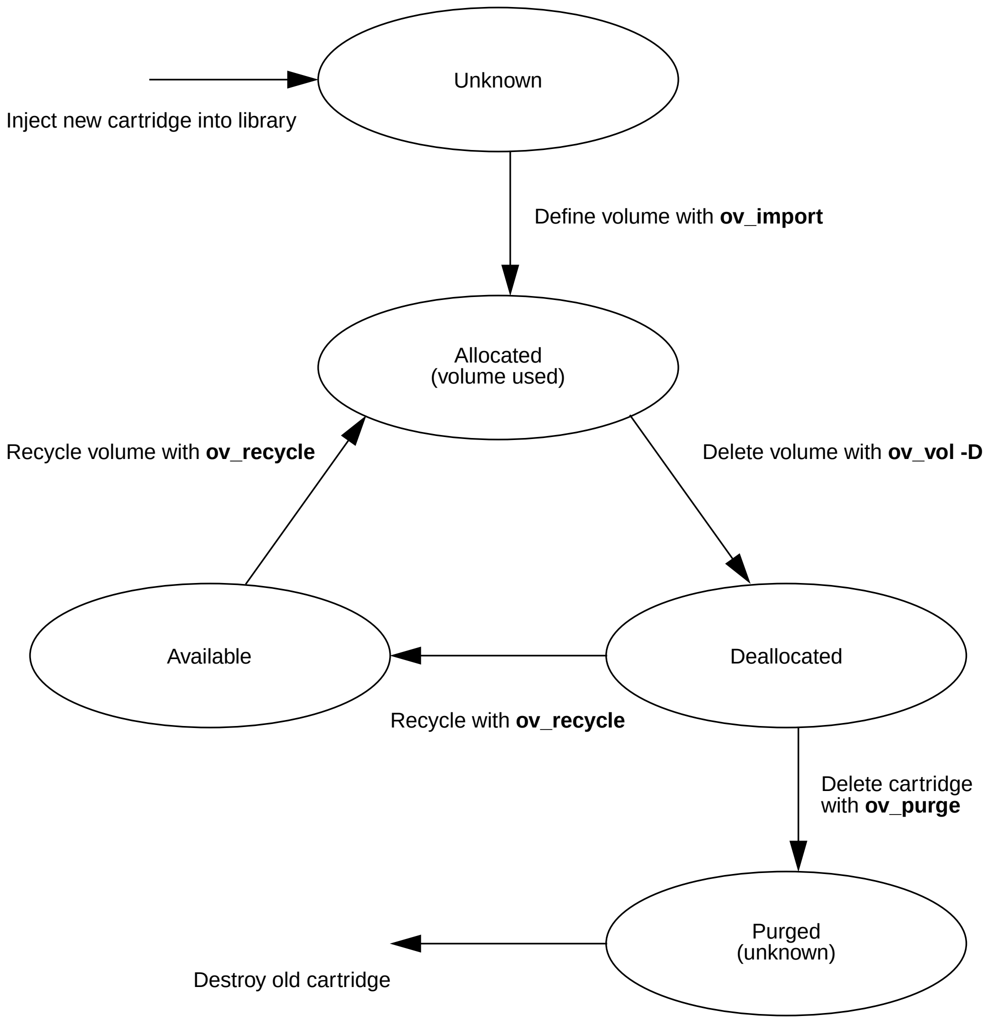
Unknown
Inject new cartridge into library
Define volume with **ov_import**
Allocated (volume used)
Recycle volume with **ov_recycle**
Delete volume with **ov_vol -D**
Available
Deallocated
Recycle with **ov_recycle**
Delete cartridge with **ov_purge**
Purged (unknown)
Destroy old cartridge

Figure 3-1 Cartridge Life Cycle (Simplified)

Managing Cartridges

Cartridges are managed in OpenVault through an administrative interface (AAPI). A set of command line tools are included that format requests to the OpenVault server and display the results on a standard terminal window.

Physical Entry Into a Library

Cartridges must be physically placed into a library before they can be used. There are two ways to accomplish this. One is to simply open the door to a library and place cartridges with barcodes in slots in the library. When the door is closed, the library control program (LCP) will command a scan of the slots in the library and report the contents of the library to the OpenVault server.

A second method is to use the inject/eject slot of the library if it is so equipped. Some libraries have an inject slot that is always open and ready to receive cartridges. For these libraries, one can simply insert a cartridge into the slot. The LCP will recognize the event and handle the cartridge appropriately.

Other libraries need to have their inject/eject slot opened under program control. In this case you may use the *ov_inject* command to instruct a library on how many cartridges to accept. A typical command would be:

```
ov_inject -l biglib -n 3
```

This tells OpenVault that you wish to insert three cartridges into the inject slot on a library named “biglib.”

Note that this feature has not been fully implemented for most libraries at the time of this writing. For now, the most reliable method of placing carts into a library is to take the library offline, open the door, and insert cartridges directly into storage slots.

Examining the Contents of a Library

It is often useful to see what a library believes to be contained within it. The command *ov_stat* performs this function (among others), displaying the contents of the slots in a library. A typical command would be:

```
ov_stat -L 9730 -s
```

The **-s** option specifies that a slotmap should be displayed, and the **-L** 9730 indicates that it should be for the library named “9730.” The output of this command looks like this:

```
# ov_stat -L 9730 -s
Library Name   Broken   Disabled   State   LCP SoftState  LCP HardState
9730           false    false      ready   ready          ready
Library: '9730'
Library    Slot Name      Slot Type  Occupied   PCL        Cart ID
9730       slot 0                    DLT        false
9730       slot 1                    DLT        false
9730       slot 10                   DLT        true       000199
9730       slot 11                   DLT        true       ABN695
9730       slot 12                   DLT        false
9730       slot 13                   DLT        true       000195
9730       slot 14                   DLT        false
9730       slot 15                   DLT        false
9730       slot 16                   DLT        true       000193
9730       slot 17                   DLT        false
9730       slot 18                   DLT        false
9730       slot 19                   DLT        false
9730       slot 2                    DLT        false
9730       slot 20                   DLT        false
9730       slot 21                   DLT        false
9730       slot 22                   DLT        true       000185
9730       slot 23                   DLT        false
9730       slot 24                   DLT        false
9730       slot 25                   DLT        false
9730       slot 26                   DLT        false
9730       slot 27                   DLT        false
9730       slot 3                    DLT        false
9730       slot 4                    DLT        false
9730       slot 5                    DLT        false
9730       slot 6                    DLT        false
9730       slot 7                    DLT        false
9730       slot 8                    DLT        true       000194
9730       slot 9                    DLT        true       000192

Library    Bay Name             Slot Type  Total Slots  Free Slots
9730       bay 1                DLT        7            0
```

This shows a library with 7 slots occupied, with the PCL listed for the cartridge in each occupied slot. The PCL info is needed to enter the data that OpenVault needs to manage cartridges. Note that at this point, the LCP has information about the PCLs of cartridges in this library, but the OpenVault server has no other information about the cartridges. Thus these cartridges are still “unknown” to OpenVault.

Creating Cartridge Data

The administrator (or an administrative application) must enter descriptive data about cartridges before they can be used. OpenVault provides two methods to enter that data: individual commands to enter data about the cartridge, partitions, and volumes on the cartridge (using *ov_cart*, *ov_part*, and *ov_vol*), and a tool that enters several of these data at once (*ov_import*). The individual commands will be discussed first as they illustrate the types of data needed. The *ov_import* command combines several steps of the individual commands and will be discussed later.

Cartridge data is entered with the *ov_cart* command. When outside of a library, a new cartridge (one with no data on it) could be entered as follows:

```
ov_cart -n ABN695 -T DLT7000 -g default
```

This creates an entry in the OpenVault catalog for a cartridge with a PCL of “ABN695.” The cartridge is a DLT 7000 cartridge, which is indicated to OpenVault with the **-T** (cartridge type) option and type description string of “DLT7000.” There is a unique description string for each type of cartridge known to the OpenVault system. Other examples include 8mm-160m, 3590, and STKtimberline. These strings tell OpenVault what kind of drive this tape uses. The **-g** option is used to specify a “cartridge group.” Cartridge groups are used to control which cartridges can be allocated by an application. In this case, the cartridge is being assigned to the group “default,” which would normally be used for cartridges which can be allocated by any application.

The *ov_cart* command generates output as follows:

```
New cart created with cartridge ID = ‘wIS2MTXrMmQAAlYR’
```

The “cartridge ID” is a unique string that identifies a cartridge, and is used to define other cartridge data. A PCL alone cannot be used as a unique identifier because a PCL might not be unique on an OpenVault system—two different types of cartridges could have the same PCL. OpenVault requires that on a given system a PCL must be unique for a single cartridge type (there cannot be two DLT7000 tapes with the same PCL).

If the cartridge was not a new cartridge but rather already belonged to an application and had data on it, one additional item of data would need to be entered. This would be the name of the application that owns the cartridge. For example, if the above cartridge were used by the *xfsdump* application, the command would be:

```
ov_cart -n ABN695 -T DLT7000 -g default -A xfsdump
```

This ensures that no other application can allocate this cartridge and overwrite its data.

When a cartridge is created, OpenVault also creates additional descriptive data about the cartridge based on its knowledge of that cartridge type. This is primarily information about sides on the cartridge, including how many sides the cartridge has, the name of each side, and creation time. Cartridge type information can be seen by running the *ov_carttype* command with no arguments. For the DLT700, this data follows:

```
Type Name  Media Type  Media Length  Slot Type  Number Sides  Side1Name Side2Name
DLT7000    DLT Compact IV   100      DLT        1             SideA
```

Most magnetic tapes have only one side. Optical and removable magnetic disk cartridges have two sides. At the time of this writing, OpenVault supports single sided media only.

Displaying Cartridge Data

Once data on a cartridge is entered into the system it can also be displayed. The *ov_lscarts* command is used for this purpose. Here are two examples of the use of this command. The first shows summary information on cartridges, and the second shows the PCLs of all cartridges known to the system.

```
# ov_lscarts
num carts num allocd num free
       15          7        8
# ov_lscarts '.*'
000185         000193         000197         ABN579         ABN695
000190         000194         000198         ABN580         ABN697
000192         000195         000199         ABN693
```

Without any arguments, *ov_lscarts* prints a summary of the number of cartridges in a system. This is because the number of cartridges in a system could be very large. If the default behavior was to show all cartridges, the screen could be completely filled with cartridge info. Thus *ov_lscarts* requires that a cartridge list be provided before it prints detailed information. The argument '.*' in the second example is a regular expression that matches all cartridge names. This is the standard way of displaying all known cartridges. Detailed information on cartridge(s) can be shown as follows:

```
# ov_lscarts -li ABN695
PCL      cart type   owner   state   cartID           part     volume
ABN695     DLT7000           ok      wIS2MTXrMmQAAlYR
```

In this example, the **-l** option specifies a long listing, and the **-i** specifies that the CartridgeID be printed also. Note that there is no partition or volume information, because we have not entered that data yet. Creating a partition is the next step.

Creating Partitions

A partition is a defined area on the media where data can be written. For OpenVault to be able to use a cartridge, at least one partition must be defined on it. Magnetic tape cartridges usually have only one partition. Optical cartridges often have one partition per side. Silicon Graphics tape driver software typically does not support multiple partitions per cartridge, so one is currently the limit for tapes on OpenVault. Partitions are created with the *ov_part* administrative command:

```
# ov_part -n wIS2MTXrMmQAAlYR -s SideA -p Part1 -b DLT7000
New partition created:
  cartridge ID = 'wIS2MTXrMmQAAlYR', partition Name = 'Part1'
```

The **-n** option indicates that this is a new partition on the cartridge with the CartridgeID 'wIS2MTXrMmQAAlYR'. Note that this is the same CartridgeID reported by *ov_cart* above when the cartridge data was entered for the cartridge with the PCL of ABN695. The **-s** option identifies the side of this cartridge on which to put the partition. Since there is only one side on a DLT7000 cartridge, there is no choice of side here—the side name of "SideA" from the cartridge type table (as shown by *ov_carttype* above) must be used.

Finally, a name must be provided for the partition. This follows the "-p" option and in this case is "Part 1." Any name may be specified here.

The *ov_part* command reports whether the partition creation was successful. If it was not, an error message is displayed.

At present there is no easy way for users to display partition information on a cartridge. The information is not needed unless you need to create a volume on the partition. If you need to do this, be sure to remember the name of the partitions you create. By convention, the standard name for the first partition is "Part 1" which must by typed in quotes on the command line. The quotes are needed so that the space between "Part" and "1" is not recognized as a delimiter between arguments on the command line.

Controlling Allocation Status

At this point the cartridge is fully defined and available. An application that requests a volume be allocated to it could get ownership of this cartridge. That is what we want to have happen for new cartridges that do not have data on them. However, if the cartridge already has data and we do not want it to be overwritten, we need to make sure that no application can allocate it.

The best time to do this is when the partition is created; otherwise an application might be able to allocate the cartridge before we can type a new command to prevent it. The allocatable status can be set with the "-a" option when we create the partition as follows:

```
# ov_part -n wIS2MTXrMmQAAlYR -s SideA -p Part1 -b DLT7000 -a false
```

This command is identical to the *ov_part* command above except for the addition of the **-a** option and its argument. A better way to define cartridges with pre-existing data is to use the *ov_import* command, described below.

Allocating Volumes

The above steps together fully define a cartridge—OpenVault now has cartridge, side, and partition information. All that is needed for an application to use the cartridge is for a volume to be created on it.

The administrative command to create a volume is *ov_vol*. The following command creates a volume named "VolOne" that will belong to an application named *ov_umsh* on the cartridge defined in the examples above:

```
# ov_vol -n VolOne -a ov_umsh -c wIS2MTXrMmQAAlYR -s SideA -p Part1
```

The **-n** option specifies that this is a new volume, with the following argument containing the name for the volume of "VolOne". The **-a** option specifies the owner application, in this case being *ov_umsh*. The following three option/argument pairs specify where the volume is to be created—on the cartridge with CartridgeID "wIS2MTXrMmQAAlYR", on its side "SideA" and partition "Part1". The output of this command is as follows:

```
New volume created:
  volume name = 'VolOne', application name = 'ov_umsh'
```

Applications may also create volumes by issuing an *allocate* request to OpenVault through the CAPI interface. OpenVault will look for an allocatable partition on a cartridge that is either not owned by any application or is owned by the requesting application. It will then create volume data for one such partition, mark the partition as not allocatable (because there is now a volume allocated on it), and set the cartridge ownership to the requesting application. See the *OpenVault Application Programmer's Guide* for further details.

Displaying Volume Data

Information on volumes can be displayed with the *ov_lsvols* command. This works similarly to the *ov_lscarts* command described above:

```
# ov_lsvols -l VolOne
volume     cartID            owner          partition
VolOne     wIS2MTXrMmQAAlYR  ov_umsh        Part1
```

As for *ov_lscarts*, issuing the command with no arguments causes summary information on volumes to be displayed, and a listing of all volumes requires a '.*' regular expression. The **-l** option specifies that a long listing be printed.

Deallocating Volumes

When an application has no further use for a volume. it may give that volume back to OpenVault for reuse. It does this by deallocating the volume, which deletes volume data. The application may do this through the CAPI interface with the *deallocate* command, or an administrator may perform the deallocation using the *ov_vol* command as follows:

```
ov_vol -D VolOne -a ov_umsh
```

The **-D** option indicates that the volume should be deleted. The **-a** option indicates which application owns the volume that is to be deleted. The application name is needed because while volume names must be unique within one application, they do not have to be unique across all applications on an OpenVault system. Specifying the application name indicates which specific volume with the given name is to be deleted.

Recycling Cartridges

OpenVault retains a fair amount of information about a cartridge after a volume on it is deallocated. The owner information (that is, which application owned the cartridge while there was a volume allocated on it) remains, as does the side and partition information. This allows administrative operations to be performed on these cartridges. A typical operation might be to erase the tape before it is released for use by other applications or users in order to protect sensitive data. Such operations are optional (and beyond the scope of this document); there is no requirement that they be performed. After such steps are performed, the volume can be "recycled," that is made available for allocation again.

The simplest method of recycling allows the owner of the cartridge to allocate another volume on it. To do this, you must tell OpenVault that this operation is allowable. This is done with the *ov_part* command as follows:

```
ov_part wIS2MTXrMmQAAlYR -s SideA -p Part1 -a true
```

The CartridgeID identifies the cartridge on which the partition resides. The **-s** and **-p** options indicate the side and partition name, as when the partition was created. The **-a** option allows setting of the allocatable status. Setting it to true allows the partition to be allocated, that is it allows a volume to be created on that partition.

Again, since the application that previously had a volume on the partition still owns the cartridge, only that application can allocate a new volume on that cartridge. Freeing up the cartridge so that any application can use it requires use of the *ov_recycle* command.

The *ov_recycle* command looks for cartridges that do not have any volumes on them and which are not available for allocation. The search can be restricted to cartridges belonging to a particular application, or to one or a list of cartridges. A typical *ov_recycle* command looks like this:

```
ov_recycle -A ov_umsh
```

The **-A** option indicates that all cartridges belonging to application *ov_umsh* that can be recycled should be recycled. Any such cartridges would have their owner information deleted, and their partitions would become allocatable again. This operation returns the cartridges to the defined and allocatable states described above, just as for a new cartridge with defined cartridge and partition information.

Simplified Entry of Media Information

A large amount of data can be entered at once, and data is shared between steps of the definition process so that the administrator does not have to read and re-enter intermediate results.

There is a simpler and faster method to define cartridge, partition, and volume data than the procedure outlined above of using *ov_cart*, *ov_part*, and *ov_vol*. The *ov_import* command may be used to perform these operations simultaneously. Data is shared between steps of the definition process so that the administrator does not have to read and re-enter intermediate results.

The *ov_import* command accepts a series of command line options that specify a cartridge group name and partition bit format. Then it reads lines from standard input (that is, from the keyboard, pipe, or input file) containing all the information needed to create complete cartridge and volume definitions. The format of the input lines is:

```
PCL CARTRIDGETYPE [VOLUMENAME APPNAME]
```

The volume name and application name are optional, but if used, both must be used together and in order. In this example, four cartridges are created with *ov_import*:

```
# ov_import -g default -b DLT7000
test001 DLT7000
test002 DLT7000
test003 DLT7000 vol1 ov_umsh
test004 DLT7000 vol2 ov_umsh
<EOF>
```

ov_import creates the cartridge data, a side and a partition on each cartridge with default names, and sets ownership and allocates a volume for the lines where volume name and owner were specified. The results can be seen with this *ov_lscarts* command:

```
# ov_lscarts -lig ’test.*’
PCL      cart type  owner     group     state   cartID              part    volume
test001  DLT7000              default   ok      wIS2MTXt0jIABa6O  PART 1
test002  DLT7000              default   ok      wIS2MTXt0jcACesb  PART 1
test003  DLT7000    ov_umsh   default   ok      wIS2MTXt0koADWwM  PART 1  vol1
test004  DLT7000    ov_umsh   default   ok      wIS2MTXt0lEAC3I8  PART 1  vol2
```

This shows that four cartridges now exist with “test” as the prefix of their PCL, that they all are DLT7000 cartridges, and that two of them have volumes and are owned by the application *ov_umsh*.

The *ov_import* command is most useful in defining cartridge information for a large number of cartridges at one time, such as when initially configuring an OpenVault system, or when adding a large number of cartridges to a library. In such cases, the *ov_stat* command can be used to get a list of PCLs of unknown cartridges in a library, and that list can then be edited into input for an *ov_import* command. This method minimizes the chances for human error in mistyping PCLs when creating cartridges, or entering incorrect CartridgeID strings for *ov_part* and *ov_vol* commands after a cartridge is created with *ov_cart*.

Removing Cartridges From Libraries

Administrators commonly remove cartridges for one of several reasons. These include disposal of cartridges after they have reached the end of their useful lives, moving cartridges to another library, or sending them to another site. As for the case of inserting cartridges in libraries, they can be removed from a library in two ways.

First, you can simply open the door of a library and remove cartridges. When you close the door, the LCP scans the library and reports changes to the OpenVault server.

The second method is to use the *ov_eject* command to tell the library which cartridges to move to the library's output port. A cartridge can be ejected by its PCL, CartridgeID, or position in a library, as follows:

```
ov_eject ABN695
ov_eject -C wIS2MTXrMmQAAlYR
ov_eject -l 9730 -b ‘bay 1’ -s ‘slot 11’
```

If a cartridge is unknown to OpenVault (that is, present in a library but OpenVault has no data on it) then it must be specified either by PCL or by library slot position. If the PCL is missing or unreadable, then only the library slot form can be used.

Purging Cartridge Data

When a cartridge has been removed from an OpenVault system and is never expected to return (such as in the case of a cartridge that has reached the end of its useful life), there may be no further need to retain data about that cartridge. The data can be deleted from the OpenVault system with the *ov_purge* command. This command deletes all cartridge, side, and partition information about a cartridge.

A cartridge cannot be purged if there are allocated volumes on it. This is a safety precaution to ensure that data on cartridges that are still in use are not lost. The *ov_purge* command also asks for confirmation before it attempts to delete data as an additional check to help prevent inadvertent loss of data. Here is an example of the use of this command:

```
# ov_purge -C wIS2MTXrMmQAAlYR
Are you sure you want to purge all information  for cartridge with
cartridge ID = wIS2MTXrMmQAAlYR (Y/N)? y
Deleted partition Part1
Deleted cartridge wIS2MTXrMmQAAlYR
```

After this command, no data remain in the OpenVault system about this cartridge, which returns to “unknown” state. If the cartridge is at the end of its useful life, the administrator should remove the cartridge from the library and destroy it, thus completing the life cycle for this cartridge.

Chapter 4

Administering OpenVault

This chapter describes how to set up and configure OpenVault, as well how to perform some basic management tasks. The sections in this chapter are:

- "OpenVault Configuration Files"
- "Administering OpenVault" on page 73
- "Monitoring OpenVault" on page 81

OpenVault Configuration Files

OpenVault uses configuration files to initiate a communication session between device components on OpenVault clients and server. Configuration files are automatically created by the OpenVault *setup* script, and contain information to bootstrap pieces of the system into full functionality. Once the various components have booted, the OpenVault server controls and manages clients and their components.

The following list defines terms used for OpenVault setup and administration:

port number	TCP port for OpenVault communication services, usually 44444.
throttling	After a certain number of clients are connected, the connection rate is reduced to minimize contention.
instance name	You assign an arbitrary name to each drive and library. The instance name distinguishes between DCPs or LCPs controlling the same drive or library from different hosts (hopefully not simultaneously).
security key	A password for authorizing client or application access to OpenVault.
default access path	Path used to access the drive via normal I/O channels, under */dev/rmt*.
passthrough driver	Path used to access a device for direct SCSI control, under */dev/scsi*.
drive handles	Directory for creating *attach* nodes for applications, often */tmp/mlm*.

OpenVault configuration files are described in detail by the following sections:

- “Server Configuration” on page 70
- “Clients Configuration” on page 70
- “Applications Configuration” on page 71
- “Setting Up Security” on page 71

Server Configuration

When first configuring OpenVault, login to the OpenVault server system as superuser and run *setup*, with its menu-based interface that helps guide you. For more information about setup procedures, see Chapter 2, “Installing OpenVault.”

The OpenVault server configuration is stored in the */usr/OpenVault/mlm/config* file.

Clients Configuration

OpenVault client software must run on all systems where OpenVault-managed drives and libraries are attached, in particular on systems not designated as the OpenVault server. These non-server systems are called OpenVault clients.

If you want to implement security, an authorization key must be created, as described in “Setting Up Security” on page 71. Client security keys are part of their library, drive, and application configuration files—they are not in a separate file by themselves.

Setting Up Drives

Each drive needs its own drive control program (DCP) and configuration description, located in the */usr/OpenVault/dcp/*/config* file. A drive-specific DCP should be installed using the standard OpenVault installation procedure.

For DCPs that are provided by Silicon Graphics, DCP installation produces an interactive *setup* script that requests the configuration information that is necessary to create the configuration file. For DCPs supplied by other venders, consult their documentation.

The *setup* script asks similar questions as for autoconfiguration of the OpenVault server, but limited to information needed for drive and DCP configuration.

Setting Up Libraries

Each library needs its own library control program (LCP) and configuration description, located in the */usr/OpenVault/lcp/*/config* file. A library-specific LCP should be installed using the standard OpenVault installation procedure.

For LCPs that are supplied by Silicon Graphics, LCP installation produces an interactive *setup* script that requests the configuration information that is necessary to create the configuration file. For LCPs supplied by other venders, consult their documentation.

The *setup* script asks similar questions as for autoconfiguration of the OpenVault server, but limited to information needed for library and LCP configuration.

Applications Configuration

The *setup* script asks you questions for configuring and storing the hostname of the OpenVault server used for the TCP interface. If you want to implement security, the application's OpenVault password must also be given. For details, see "Setting Up Application Security" on page 75.

Setting Up Security

An ASCII authorization key file must be created on each AAPI/CAPI client and server system, and the client must be configured to use Open Vault with each key file. The authorization key file uses a *shared secret*—the client and server share a secret password. The server has one key file (*/usr/OpenVault/var/core_keys*) containing the passwords of each of the clients it knows about (drives, libraries, and applications). The clients each have an authorization key file listing the servers it knows about.

Example 4-1 shows an example of a key file that can be used on either a client or a server.

Example 4-1 Client and Server Key Authorization File

```
#Host           Client         Instance     Language      Key
moon            System         OnlyInstance AAPI          a4sum
moon            Robbie         one          ALI           a2by4
moon            Herbie         alpha        ADI           gr8day
earth           networker      *            CAPI          un2
```

The meanings for each column in the key authorization file are described in Table 4-1.

Table 4-1 Key Authorization File Description

Column Title	Description
Host	IP hostname of the system that communicates with this system. For example, if this is a CAPI client, give the OpenVault server hostname. If this is the OpenVault server, give the IP hostname of the client running the application.
Client	Name of the application.
Instance	The instance name of the application. If more than one version is running, use asterisk (*) as a wildcard for multiple instances of the same application.
Language	The language used for the connection, either AAPI, CAPI, ADI, or ALI.
Key	A password used to secure the connection, or "none" for no security.

Any field in the key file can contain an asterisk "*" to specify all conditions. For example, if a server uses a * in the Host column, all OpenVault clients can connect with the server. If a client uses a * in the Host column, all OpenVault servers can connect with this client. If a client uses a * in the Key column, it indicates no password, and OpenVault runs with security disabled.

Note: Use asterisks carefully; too many asterisks in this file make the OpenVault system vulnerable to security violations. Also ensure that wildcard entries appear at the bottom of the file, after non-wildcard entries.

On the server host, the shared secret key is stored in */usr/OpenVault/var/core_keys*.

Setting Up Non-Robotic Libraries

Note: This is not supported in the OpenVault 1.x releases.

Non-robotic libraries are controlled and manipulated by a human operator rather than by a robotics interface. Organization of your cartridges is key to keeping your library accessible and usable. OpenVault can help by tracking the location of cartridges within a library. Assigning cartridges to a *bay* can further pinpoint cartridge location.

A bay is a storage location within the library. In the case of robotic libraries, bays can be a range of slots, a removable tray, or a storage silo. In the case of non-robotic libraries, a bay could be a bookshelf, a file drawer, or a storage closet. In these examples, a bookcase, file cabinet, or hallway is the library, partitioned into bays as you prefer.

Planning is important in determining how many bays and how many libraries are necessary for your storage needs. Be sure to plan adequate space to allow your library to grow and allow for further separation, based on detail (you may start with a “project,” but then add more bays for phase 1, phase 2, and so forth).

Another major difference with a manual library is the need for communications with the operator. OpenVault needs to send messages to the operator, via a terminal, to prompt for action, such as loading a cartridge or removing a cartridge. Be sure to put the terminal that the operator uses in a convenient location so that timely communication is possible.

Setting Up Offsite Libraries

Note: This is not supported in the OpenVault 1.*x* releases.

An offsite library is a special type of manual library, differing in that an offsite library has no drive. Storage bins (which can be referred to as bays) contain the cartridges. OpenVault tracks the location of offsite storage cartridges. If a cartridge is needed from an offsite storage area, it must manually be located and moved into another library (see “Moving Cartridges Within Libraries” on page 89). The chosen library must contain a drive that is compatible with the cartridge type.

Administering OpenVault

Once OpenVault has been set up, as described in “OpenVault Configuration Files” on page 69, you can start to configure it to meet your storage management needs. Some basic configuration tasks are:

- “Setting Logging Levels” on page 74
- “Registering Applications” on page 74
- “Unregistering Applications” on page 75
- “Setting Up Application Security” on page 75
- “Enabling Application Access to a Drive” on page 76

- “Managing Cartridge Groups” on page 76
- “Introducing Cartridges” on page 79

Tip: When using the OpenVault administrative commands, you can assign environment variable OVSERVER to name a default server. For example, if you add this line to your *.login* file, administrative commands use the OpenVault server *hostname*:

```
setenv OVSERVER hostname
```

If you want to use a different server, use the **-S** command-line option to specify another server (**-S** *newOVserver*). This overrides the OVSERVER environment variable.

Setting Logging Levels

You can configure the degree of system status reporting and message logging that you want to have. You can also specify a different name for the message log file. By default, system message logging sends all error messages to the file */usr/OpenVault/var/OVLOG.*

An example follows to set the “information” log level. See ov_msg(1M) for details.

- To set logging level to the information level:

  ```
  ov_msg -s -t core -m information
  ```

See “Accessing the OpenVault Message Log” on page 105 for message log information.

Registering Applications

Applications are client programs that can read data from or write data to (or both) removable media after OpenVault has found and mounted the desired media element. Reads and writes are performed using POSIX standard I/O facilities.

Applications can be added to (registered with) the OpenVault system at any time without the need to take OpenVault offline or perform a software upgrade. Registering an application introduces it to the OpenVault system so resources can be allocated for its use. See ov_app(1M) for a description of options.

Unless an application is registered, it cannot connect to the OpenVault server.

An authorization key, or password, can be added when registering an application to ensure secure transactions; see “Setting Up Application Security” on page 75.

To register an application without using security, follow these steps:

1. Check to make sure that the application is not already registered.

   ```
   ov_app -l '.*'
   ```

   A listing appears of applications that are registered with the default OpenVault server. To check on other hosts, use the **-H** *host1*, *host2* option. If you don’t see your application named, continue with step 2.

2. Register the application (*appName*) with the default server.

   ov_app -c *appName*

Unregistering Applications

Unregistering an application from the OpenVault system means the application can no longer connect to OpenVault.

- To unregister the application called *networker* (IRIX NetWorker):

  ```
  ov_app -d networker
  ```

Setting Up Application Security

Application security setup is similar to client security setup. Setting up an application with an authorization key ensures that, without the security key, no other application can connect to OpenVault masquerading as the originally authorized application.

As shown in the first and last lines of Example 4-1, the application name is used as a Client for the OpenVault server, which is named as the Host. The first line is an example that secures the OpenVault administrative tools and the last line is an example that secures the IRIX Networker application. The middle two lines appear only in the server keyfile and describe options for a library named “moon” and a drive named “moon”.

The key that is assigned in the key authorization file, also known as the password, must already exist in the configuration file for the application. See the section “Applications Configuration” on page 71 for details on setting up his file.

Enabling Application Access to a Drive

A drive is in exactly one drive group. Each application can be given access to one or more drive groups (all drives in a group). You can enable a drive for a specific application, allowing the application to run on the drive. This might be necessary if the application uses special features of the drive.

1. To enable a drive for the *tar* application, use the following command:

   ```
   ov_drive -c Herbie targroup false
   ```

   This command creates the drive named Herbie in the group *targroup*. The `false` argument indicates that the drive is not disabled for group Herbie.

2. Create the drive group *tardrive* and add the *tar* application with a prime precedence rating **-p 1** (lower number drives are used first).

   ```
   ov_drivegroup -c tardrive -A tar -p 1
   ```

   Now the *tar* command tends to use drives in this drive group. A **-p** *precedence* value organizes all accessible drive groups into a preference list for the application.

Drive groups allow a set of drives to be specified for a particular use, most likely to be available for a particular application. You can set up drive groups by repeating the *ov_drivegroup* **-c** command for each drive group you want, allocating applications to each. Adding a **-p** *precedence* flag when you specify the application allows you to make certain applications more or less likely to use drives in that drive group.

Managing Cartridge Groups

Cartridge groups allow sets of cartridges to be allocated for specific applications or purposes. You may want to do this, for example, based on the capacity of the cartridge or its content (for example, the existing data on the cartridge is related to other cartridges in the group). Creating cartridge groups helps you organize your media and fulfills the demands of the application for available storage.

Figure 4-1 shows an environment with several cartridge groups. Cartridges move in and out of groups (shown by the arrows) based upon demands made by the application or by their place in the media life cycle (see “Preventative Maintenance” on page 96). The cartridge groups in the figure are Unallocated, Void, and Degauss:

Unallocated	A set of unallocated cartridges that are available for allocation by any application. These cartridges have been introduced into the OpenVault system and are ready to be allocated to some application.

Void	A set of cartridges whose labels are not in the OpenVault library. This is a holding place until you move the cartridge to another group, such as Scratch, or find out why the OpenVault catalog entry for the cartridge is missing.
Degauss	A set of cartridges that are ready to be demagnetized or destroyed. Cartridges are demagnetized to ensure all data is destroyed. Some of these cartridges may need to be terminated if they are approaching the end of their media life span.
Full	IRIX NetWorker manages this set of cartridges. When a volume fills up, NetWorker marks it as full and holds it for a specified retention period.
Recyclable	IRIX NetWorker manages this set of cartridges, too. When the retention period expires and its cycle ends, a volume is marked for recycling.

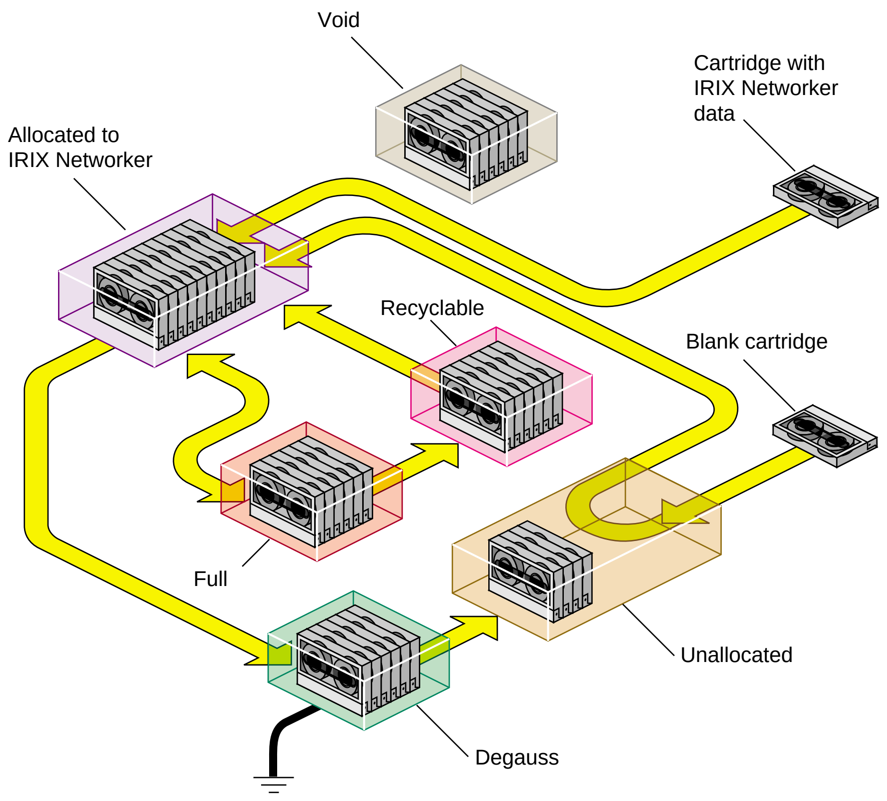
Void
Cartridge with IRIX Networker data
Allocated to IRIX Networker
Recyclable
Blank cartridge
Full
Unallocated
Degauss

Figure 4-1 Example Cartridge Group (Engineering)

Setting Up Cartridge Groups

Before setting up a cartridge group, plan where the group will reside and which drives or applications need to access it. Does the cartridge group require high throughput or high availability? Consider the network environment and any limitations it may impose when deciding where to locate the group. Consider the drives that will be servicing the group. What advantages or limitations do they impose? Also consider the growth potential for your storage requirements and choose a library that is adequate for its needs both now and for its projected future.

Use the following procedure to create a cartridge group.

1. Be sure you have registered the application that will manage this group. See "Registering Applications" on page 74 for details.
2. Determine whether to assign drives to the application. If you decide to do so, see "Enabling Application Access to a Drive" on page 76 for details.
3. Create the cartridge group and assign its access permission. For example, the following command creates the "NWbackup" group with high priority (**-p 1**) and permits only the IRIX Networker application access to its cartridges (**-q 1**).

   ```
   ov_cartgroup -c NWbackup -p 1 -A networker -q 1
   ```
4. Introduce cartridges into the cartridge group. Continue with the next section, "Introducing Cartridges" on page 79, for details.
5. If necessary, move cartridges between groups or within the library. Examples of when this might be necessary are when a cartridge
   - is no longer needed by an application—see "Removing Cartridges From Libraries" on page 89
   - can be moved back into the Unallocated group—see "Recycling Cartridges" on page 90
   - needs to be sent to the Degauss group for erasure—see "Destroying Cartridges" on page 90
   - needs to be moved within its own group—see "Moving Cartridges Within Libraries" on page 89

Introducing Cartridges

Cartridges must be introduced into the OpenVault system before they can be used by an application. When a cartridge is introduced, an entry in the OpenVault catalog is created for the cartridge and the cartridge is assigned a cartridge ID. The entry, or cartridge ID, tracks information such as the PCL or bar code, application ID (if it's associated with an application), number of reads and writes, form factor, capacity, and so forth.

Client applications may choose their own names for cartridges. Because OpenVault client applications operate in separate name spaces, different applications may use the same name for different cartridges. Moreover, cartridges used by one application are not visible to or accessible from another application, unless the system administrator permits specific cartridges to be moved from one application to another.

Cartridges can be introduced in two ways, as shown in Figure 4-1:

- cartridges without data (including data that can be overwritten).
- cartridges containing data that is needed by an application (partition needed)

To introduce a cartridge without data, follow these steps:

1. To facilitate the cartridge identification, create its PCL by attaching a bar code or a label to the outside of the cartridge. The PCL is tracked in the catalog entry for this cartridge and is used as the default identification for the cartridge in most commands.
2. Signal the library to open its inject port so you can insert the cartridge. You must specify the library name (**-l** *libraryName*).

   `ov_inject -l` *libaryName*
3. Insert the cartridge into the inject port.

4. Create a new (*ov_cart* **-n -T**) cartridge entry in the OpenVault catalog so the cartridge can be recognized and accessed. The **-n** *label* gives the PCL affixed to the cartridge. Also, assign the cartridge to a cartridge group (**-p** *cartridgeGroup*) so it can be used by an application needing a new cartridge:

   ov_cart -n *label* **-p** *cartridgeGroup* **-T** *cartridgeType*

   OpenVault checks its catalog to ensure the PCL is unique (new) and rejects the entry if that PCL already exists. Then *ov_cart* automatically creates sides on the cartridge, depending on the *cartridgeType* argument. The group name, form factor, media type, PCL and other identifying information are recorded in the OpenVault catalog for this cartridge.

   Note: This command (*ov_cart* **-n -T**) is a good way to enter each new cartridge into the OpenVault system. Bulk insertion of cartridges into OpenVault may also be done using the *ov_import* command.

5. Create a partition on side 1 of the cartridge.

   ov_part -n *cartridgeID* **-p** *partitionName* **-s SideA**

To introduce a cartridge that already contains data:

1. Follow the procedure for introducing cartridges without data, including step 5.
2. Create volumes for this cartridge. A volume allows the cartridge to be segmented so separate areas of the media can be used by an application:

   ov_vol -n *volumeName* **-a** *applicationName* **-c** *cartridgeID* **-p "PART 1" -s SideA**

   Use this command for each volume you are creating. OpenVault checks that the *applicationName* and *volumeName* combination is unique. Standard attributes are assigned to the volume and given default values if the values are not specified.

   Note: This *ov_import* command optionally allows you to automatically create volume records for cartridges with data, in addition to creating the cartridge, side, and partition information for OpenVault.

You have now successfully introduced a cartridge and associated it with an application. Repeat this procedure for each new cartridge you are adding to the OpenVault system. You can now perform operations on the cartridge, as described in Chapter 5, "Operating OpenVault."

Monitoring OpenVault

Monitoring OpenVault helps you spot potential problems so you can take appropriate action, and allows you to track the OpenVault system usage. If reconfiguration is necessary, it can be done before it becomes critical.

Probably most helpful in daily operations, monitoring can help you track the data you have on your media and where the media is located.

The following subsections elaborate on these tasks:

- “Checking Server Status and Configuration” on page 82
- “Checking Media Inventory” on page 82
- “Listing Cartridge Information” on page 83

OpenVault can be monitored on several levels and to varying degrees of detail. From the general to the specific, some of the areas that can be monitored are:

- OpenVault server status, including whether the server is up or down and by listing system error messages
- library status, including names, types, whether the library is up or down, and slotmaps (showing occupancy and reservations)
- drive status, including names, types, whether the drive is up or down, and cartridge occupancy
- libraries and drives without active LCPs and DCPs
- cartridge group status, including names of groups, number and names of cartridges in them
- names of registered applications
- cartridge status, including listing of all known cartridges, cartridge locations, and cartridge characteristics
- volume status, including cartridge ID and PCL, and the owning application
- system messages, including by library, drive, and type (warning, error, operator)

Check the OpenVault reference pages for a complete description of available commands and options (see Appendix B for a complete listing of reference pages).

Checking Server Status and Configuration

Checking the status of the OpenVault server allows you to view your system from a top-level viewpoint. Some things to check for are:

- Check that the default server is up:

  `ov_stat`

- Check that a specific server is up:

  `ov_stat -S` *serverName*

- Show the task queue for the default server:

  `ov_stat -q`

- Display all status items for the server, LCPs, DCPs, and applications:

  `ov_stat -a`

- Show configuration (**-c**) and drive (**-d**) mode information for a drive.

  `ov_stat -D` *driveName* **`-d -c`**

- Show library (**-l**), configuration (**-c**), and slot map (**-s**) information for a library:

  `ov_stat -L` *libName* **`-l -c -s`**

Checking Media Inventory

Knowing your media inventory is essential in a well-organized site. OpenVault helps you track the contents of your libraries and cartridge groups so you know the whereabouts of your cartridges, including those that are currently loaded into drives.

The following commands can be used to check the media inventory:

- Show location information for all cartridges, such as the library location, slot assignment, presence in slot, or if loaded in a drive, the name of that drive:

  `ov_lscarts -w '.*'`

- Show cartridges assigned to the IRIX Networker application, in long format:

  `ov_lscarts -A networker -l '.*'`

Listing Cartridge Information

Other information that is important to a smooth-running site is information regarding cartridge contents and usage, such as the number of reads, writes, mounts, and so forth.

The following can be used to check information about cartridges:

- Show a summary of all cartridge information on the default server:

  ```
  ov_lscarts '.*'
  ```

- Show the application and cartridge group associated with each cartridge:

  ```
  ov_lscarts -g -a '.*'
  ```

Chapter 5

Operating OpenVault

This chapter describes the OpenVault commands that are used to perform operator tasks. The tasks are divided into normal, everyday operations and those that are performed occasionally, mainly for organizational reasons. The sections in this chapter are:

- "Performing Daily Tasks" on page 85
- "Performing Occasional Tasks" on page 88

Performing Daily Tasks

This section describes the most common operator storage management tasks. Mainly, these tasks revolve around manipulating cartridges and managing devices (libraries and drives). The subsections described in this section are:

- "Manipulating Cartridges" on page 85
- "Managing Devices" on page 87

Manipulating Cartridges

The operator manipulates cartridges by performing the following operations:

- "Mounting Cartridges" on page 85
- "Checking the Task Request Queue" on page 86
- "Canceling a Pending Task Request" on page 86

Mounting Cartridges

A cartridge is mounted in a drive to allow a read or write operation to be performed by an application. When a cartridge is mounted, as performed with the default *ov_mount* command, it is placed into an available drive that is compatible with the cartridge's format. A device name is returned so the cartridge can be manipulated by an application.

Once the cartridge is mounted into the drive, a shell process is started that allows an application to manipulate the drive. After the application finishes its operation, the shell process exits, and the cartridge is automatically unmounted. By default, you specify a cartridge with its PCL (*ov_mount* **-C** *PCL*).

With the mount command options, you can specify:

- a cartridge using its cartridge ID:

  `ov_mount -C` *cartridgeID*

- a particular drive by its drive name, as recorded in the client configuration file (see "Setting Up Drives" on page 70):

  `ov_mount -d` *driveName* **`-C`** *PCL*

- a cartridge owned by an application:

  `ov_mount -C` *PCL* **`-A`** *application*

Note: Occasionally, you may have to use *ov_unmount* to free a cartridge that is still in a drive because the application that mounted it has died. The unmount command uses all the options as the mount command, as described in "Mounting Cartridges."

Checking the Task Request Queue

When an OpenVault task request is initiated (for example, by a client application or the OpenVault *ov_mount* command), a task request (with its own ID) is generated and placed in the OpenVault server task queue. To check the pending task requests, use:

```
ov_stat -q
```

Canceling a Pending Task Request

If you want to cancel a pending task request (for example, the desired cartridge is not available), use the cancel command (*ov_cancel taskID)*, specifying the task by its ID number (as obtained with the *ov_cancel* command).

With the *ov_cancel* command, you can also send an explanation to the task that originated the task request, explaining why the operation is being canceled:

`ov_cancel -r "`*reason for cancellation***`"`**

Tip: If the string contains spaces, enclose it in quotation marks.

Managing Devices

Sometimes the operator needs to step in and manually control a device to perform a task (such as taking a drive offline so it can be cleaned). This section describes some tasks the operator can perform to manage devices.

A device (a library or drive) is specified by its device name, which is the name used in the configuration file (see "Setting Up Drives" on page 70 and "Setting Up Libraries" on page 71).

The tasks described in this section are:

- "Disabling and Enabling Devices" on page 87
- "Cleaning a Drive" on page 88

Disabling and Enabling Devices

You can temporarily disable a device to take it offline for a period of time in case servicing is necessary, such as when a drive needs cleaning. Permanently disabling a device may be necessary if it is malfunctioning and repairs need to be performed.

When the device is disabled, either temporarily or permanently, all communication to it is stopped and the OpenVault system considers it unavailable.

The following commands disable and then enable the device:

- To temporarily disable a drive and a library:

  ```
  ov_drive -T name
  ov_library -T name
  ```

- To permanently disable a drive and a library:

  ```
  ov_drive -D name
  ov_library -D name
  ```

- To enable a drive and a library:

  ```
  ov_drive -E name
  ov_library -E name
  ```

Cleaning a Drive

Note: Cleaning facilities are not supported in OpenVault release 1.1.

OpenVault helps monitor a drive's cleaning schedule by tracking the dates a drive has been cleaned and the number of read/write errors occurring on the drive. The *ov_clean* command also helps find and load a cleaning cartridge, if one is available.

Drives are specified by the name recorded in the client configuration file (see "Setting Up Drives" on page 70)

- To display the cleaning information for a drive:

  `ov_clean -i -d` *driveName*

- To check whether a cleaning cartridge is available and which cartridge will be used if a cleaning cartridge is not specified:

  `ov_clean -n -d` *driveName*

- To perform the cleaning operation, using the cleaning cartridge OpenVault selects:

  `ov_clean -d` *driveName*

- To perform cleaning, using a particular cleaning cartridge, specified by PCL:

  `ov_clean -d` *driveName PCL*

Performing Occasional Tasks

This section describes the tasks that you probably perform occasionally, generally to provide organization for your cartridge storage and keep the OpenVault catalog synchronized with the movement of your cartridges and libraries. The OpenVault catalog tracks the location and status of all cartridges it is aware of.

The tasks described in this section are:

- "Removing Cartridges From Libraries" on page 89
- "Moving Cartridges Within Libraries" on page 89
- "Maintaining the Server Catalog" on page 90

Removing Cartridges From Libraries

When you want to remove a cartridge from a library (also known as ejecting), use the *ov_eject* command. The OpenVault catalog entry for that cartridge is not removed and all record of the cartridge is saved. The cartridge is either ejected (if the library can perform an eject) or an operator message is sent to the console, indicating to the operator that the cartridge can be physically removed. After a cartridge is ejected, it must be injected again (using *ov_inject*) before it can be used.

The following options can be used when ejecting a cartridge from a library:

- Eject a cartridge using its PCL (default):

  `ov_eject` *PCL*

- Eject a cartridge using its location (library, slot, and bay, if bays are present):

  `ov_eject -l` *libraryName* **`-s`** *slotNumber* **`-b`** *bayName*

Moving Cartridges Within Libraries

Note: Cartridge move is not supported in OpenVault release 1.1. To move cartridges within a library or between libraries, first eject them and then inject them into the library. See ov_eject (1M) and ov_inject(1M) for more information.

Occasionally you may want to reorganize cartridges in a library. For example, you may decide to group cartridges by application or to move them closer to their assigned drive.

Before moving any cartridges, it is helpful to generate a listing showing the contents of slots in the library. This should help you determine which slot addresses are immediately available for use and which cartridges may have to be moved to obtain their slot. The format of the slot address is specific to the LCP in use. To obtain a library slotmap:

`ov_stat -s -L` *library*

When using the *ov_move* command, you can specify cartridges by their PCL or by their location in the library.

- Move a cartridge using its PCL:

  `ov_move` *PCL destinationSlotAddress*

- Use the library location (including slot ID) instead of the PCL:

  `ov_move -L` *libraryName originalSlotAddress destinationSlotAddress*

Maintaining the Server Catalog

The OpenVault server catalog tracks the location and details of each cartridge known to the OpenVault system and the up or down status of the drives and libraries. Occasionally, you may need to step in and make some corrections to the catalog to update it. The tasks in this section describe these actions for the catalog:

- "Recycling Cartridges" on page 90
- "Destroying Cartridges" on page 90

Recycling Cartridges

When you want recycle a cartridge in OpenVault system for use by another application, use the *ov_recycle* command. All information relative to cartridge use (such as number of reads and writes) remains in the OpenVault catalog.

- Recycle the cartridge:

  `ov_recycle` *PCL*

- Recycle the cartridge, using its cartridge ID:

  `ov_recycle -C` *cartID*

- Recycle all cartridges owned by the given application:

  `ov_recycle -A` *application*

Destroying Cartridges

When you want to completely remove a cartridge from the OpenVault system, for example to destroy a cartridge when it is generating I/O errors at the end of its life cycle, use the *ov_purge* command, which erases the cartridge entry from the OpenVault catalog.

- Remove the cartridge entry:

  `ov_purge` *PCL*

- Remove the cartridge entry, using its cartridge ID:

  `ov_purge -C` *cartID*

- Suppress the interactive verification; immediately remove the cartridge entry:

  `ov_purge -f` *PCL*

Note: Use *ov_purge* sparingly, because important usage information is lost forever.

Chapter 6

Reconfiguring OpenVault

This chapter describes OpenVault methods for reconfiguration and performance tuning. The sections in this chapter include:

- Importing media into different cartridge groups
- Adding (configuring) or deleting (deconfiguring) drives
- Changing the drive group of a drive
- Changing the name of a library
- Adding remote drives, libraries, and applications at a later time
- Establishing OpenVault security after setup with "no security"
- Changing the OpenVault password for applications, libraries, and drives

Importing Media Into Cartridge Groups

To import media into different cartridge groups, use the *ov_import* command, perhaps automated with input scripts, to import cartridges into cartridge groups specified by the **-g** option. For example, to import four tapes into the DMF cartridge group, and two other tapes into the NetWorker cartridge group, run these commands:

```
# ov_import -g dmf -b DLT7000
test001 DLT7000 vol1 dmf
test002 DLT7000 vol2 dmf
test003 DLT7000 vol3 dmf
test004 DLT7000 vol4 dmf
Ctrl+D
```

```
# ov_import -g networker -b DLT7000
test0A DLT7000 volA networker
test0B DLT7000 volB networker
Ctrl+D
```

For more information, see "Simplified Entry of Media Information" on page 65.

Adding or Deleting Drives

When you add a drive to your system, OpenVault must recognize the drive in order to put it under management. Use the *ov_drive* command with the **-c** option to create an OpenVault record of (and DCP for) the drive. For example, to create *drive1* in the *xfsdump* drive group, enter this command:

```
# ov_drive -c drive1 xfsdump false
```

The *ov_drive* command takes three arguments after **-c**, specifying the drive's name, its drive group, and administrative disabling. The third flag controls whether the drive is temporarily disabled (`temp`), permanently disabled (`perm`), or not disabled (`false`).

To remove a drive from OpenVault management, use the *ov_drive* command with the **-d** (delete) option. For example, to delete *drive1*, enter this command:

```
# ov_drive -d drive1
```

Changing the Drive Group of a Drive

To change the drive group of a drive, you must first delete that drive (and its DCP), then create it anew in a different drive group.

To remove a drive from OpenVault management, use the *ov_drive* command with the **-d** (delete) option. For example, to delete *drive7*, enter this command:

```
# ov_drive -d drive7
```

If the new drive group does not exist, create it with the *ov_drivegroup* command. If that drive group already exists, skip the following step. It is likely that you should associate an application (DMF) with the drive group at this time:

```
# ov_drivegroup -c dmf -A DMF
```

To return the drive to OpenVault management, use the *ov_drive* command with the **-c** (create) option. For example, to create *drive7* in drive group DMF, not administratively disabled (`false`) until deleted, enter this command:

```
# ov_drive -c drive7 dmf false
```

Changing the Name of a Library

To change the name of a library, you must first delete that library (and its associated LCP), then create it anew with a different name. Before changing the name of a library, be sure to stop and restart the OpenVault server to avoid causing confusion:

```
# /usr/OpenVault/stop
# /usr/OpenVault/start
```

To remove a library from OpenVault management, use the *ov_library* command with **-d** (delete) option. For example, to delete *lib2*, enter this command:

```
# ov_library -d lib2
```

To return the library to OpenVault management, use the *ov_library* command with **-c** (create) option. For example, to rename this library *tapebot*, enter this command:

```
# ov_library -c tapebot false
```

The *ov_library* command takes two arguments after **-c**, specifying the library's name, and administrative disabling. The second argument controls whether the drive is temporarily disabled (`temp`), permanently disabled (`perm`), or not disabled (`false`).

Adding Remote OpenVault Components

To add remote OpenVault components, inform the server about them, then run the *setup* script on the remote client, as documented in Chapter 2, "Installing OpenVault."

Establishing OpenVault Security

When you initially configured OpenVault, you probably followed instructions in the documentation and created an installation without security. This implies that the *admin/keys* file, the *dcp/*/config* and *lcp/*/config* files, and the *var/core_keys* file, specify "none" as the security key.

Note: Perhaps this is obvious, but passwords for specific applications in *admin/keys*, for DCPs in *dcp/*/config*, or for LCPs in *lcp/*/config*, must be the same as the password given in the *var/core_keys* file for that component.

To establish security, become superuser and edit these files, substituting the password of your choice for the word "none" on lines reading key:

```
# cd /usr/OpenVault
# vi admin/keys dcp/*/config lcp/*/config var/core_keys
~
/none
```

Changing OpenVault Passwords

OpenVault authorization is aided by passwords specified in the *admin/keys* file, the *dcp/*/config* and *lcp/*/config* files, and *var/core_keys*. These passwords can all be the same, or they can be different from one component to the next. To change passwords, become superuser and edit these files, substituting the password of your choice for old password, either globally, or component by component:

```
# cd /usr/OpenVault
# vi admin/keys dcp/*/config lcp/*/config var/core_keys
~
/key:
```

Reconfiguring Server Operation

The */usr/OpenVault/mlm/config* file contains crucial OpenVault operational parameters, expected to remain fairly static, as shown in Table 6-1. Modify them with utmost care.

Table 6-1 OpenVault Server Parameters

Parameter	What it Controls
PORTNUM	TCP/IP port number on which OpenVault listens for connections
SEMAKEY	semaphore key that OpenVault uses for communication between components
MAXSTARTS	maximum number of connections before OpenVault rejects new ones
BOOTGRACE	after reboot, number of seconds OpenVault waits before ejecting drives
TASKRETRY	number of seconds OpenVault waits before reevaluating blocked mounts
MPRETRY	number of seconds OpenVault waits before retrying a failed mount

Chapter 7

Tertiary Storage Management

This chapter describes some strategies for optimizing storage management:

- "Tertiary Storage Devices" describes the mass storage options available today.
- "Connecting to a Host Computer" on page 98 talks about cabling tertiary storage devices to server machines.
- "Storage Management Applications" on page 101 discusses tertiary storage software for backup, archive, and hierarchical storage management.

Tertiary Storage Devices

This section discusses the hardware currently available for tertiary storage, also called nearline storage. The hardware used for secondary storage is usually magnetic disk, which offers the advantages of permanence, rapid random access, and decreasing cost. Laser-activated protein storage may eventually provide even higher capacity and lower power consumption than magnetic disk. Primary storage usually refers to chip-based electrical memory such as cache or random access memory (RAM).

Tape Drives

Tape drives, because of their rewritability and low cost per unit of data stored, are now the preferred method for backing up data to protect against data loss.

Tape cartridges cost from US$10 for a 2 GB DAT tape to almost US$100 for a 35 GB DLT tape. Cost per megabyte (now about 1¢ including amortized tape drive investment) has been declining as tape capacity increases while cartridge price remains about constant.

By comparison, storage on magnetic disk, which of course provides rapid and random access, now costs under 10¢ per megabyte and is declining more rapidly than tape cost.

The typical magnetic tape lasts about five years, and can be rewritten hundreds of times. Just before a tape fails, its soft error rates rise. OpenVault can transmit the soft error rate as reported by hardware. After a tape fails, there is no good way to recover the data stored on it. OpenVault can monitor the total number of reads and writes to a tape, so you can arrange transfer of data to new media before the tape fails.

Table 7-1 shows the characteristics of several popular tape drives now on the market.

Table 7-1 High Capacity Tape Drives

Drive	Native Capacity	Data Rate	Cartridge Load Time	Tape Size	Expected MTBF	Power Needed	Typical Price
DDS-3 (DAT)	12 GB	1.0 MB/sec	30 secs	4 mm	35,000 hours	6 w	$1500
DLT 4000	20 GB	1.5 MB/sec	45 secs	1/2 in.	80,000 hours	25 w	$3600
DLT 7000	35 GB	5.0 MB/sec	40 secs	1/2 in.	200,000 hours	37 w	$8500
Exabyte Mammoth	20 GB	3.0 MB/sec	20 secs	8 mm	200,000 hours	15 w	$4200
IBM Magstar 3570	5 GB	7.0 MB/sec	16 secs	1/2 in.	3 year uptime	40 w	$8500
IBM Magstar 3590	10 GB	9.0 MB/sec	16 secs	1/2 in.	3 year uptime	60 w	$2500 0
LTO Ultrium	100 GB	15 MB/sec	?	1/2 in.	?	?	$4000
Sony AIT	25 GB	3.0 MB/sec	7 secs	8 mm	200,000 hours	12 w	$5200

Preventative Maintenance

Tape drives should be kept in low-dust environments with moderate temperature and humidity. They should be cleaned when the cleaning light comes on, or in the absence of a cleaning indicator, at regular intervals as recommended by the manufacturer. If a tape drive has a cleaning indicator, it is best to clean the drive only when indicated, in order to reduce wear on the tape heads.

Using high quality media helps preserve tape heads and can reduce cleaning intervals. Always discard cleaning cartridges before they reach the end of tape.

Pay close attention to drive alerts and media faults and act quickly to resolve problems. Monitor your tape drives daily, and read the error logs. Tape drives usually give out subtle indicators before failing. More frequent error correction code (ECC) messages often indicate impending drive failure. Perform read-write confidence tests at regular intervals, because testing can identify failing hardware before data loss occurs.

Optical Drives

For software and data distribution, and for archival purposes, optical drives have now surpassed magnetic tape.

In moderate quantities, compact disc (CD) can be manufactured for under 50¢ a copy. Writable compact disc recordable (CDR) media now cost under $3 each. (Rewritable CDWR media now cost almost $20 each.) CDR burners have been declining in price to well under $500 today. Although the CD format is limited to about 650 MB, the digital versatile disc (DVD) format offers backward compatibility with CD, and much higher data capacity, variously quoted between 4 GB and 17 GB.

Although the cost per megabyte is much higher for CD than for magnetic tape, optical data can last virtually forever. The problem is that data must be staged before writing to disc, making the CDR inconvenient as a backup device—tapes can stop and start.

SCSI Media Changers

The SCSI 2 standard specified a range of commands for interfacing with removable media libraries, also called autochangers or jukeboxes. Such devices contain one or more tape or optical drives, and robotic mechanisms to exchange cartridges between storage slots and a drive. SCSI media changers can cost anywhere from under $4000 to much more, depending on the number of drives and slots configured in the device.

There are many manufacturers of SCSI media changers, and robotic designs vary, but most media changers include one or more of the tape drives listed in Table 7-1.

When media changers are configured with multiple drives, they can also have multiple SCSI interfaces, so that drives can operate concurrently. It is most useful to have two or more drives in a media changer, and as many storage slots as possible. Device control may be done over a serial line, or by means of the lowest numbered SCSI interface.

Silo Libraries

When data transfer speed and aggregate storage capacity are at a premium, datacenters often choose proprietary silo libraries from Emass/Grau, IBM, or STK. These silo devices contain high-speed robotic mechanisms and multiple tape drives, and operate under control of a dedicated computer interface, rather than under the SCSI standard. Each tape drive usually has a direct SCSI connection to a network server, however.

In OpenVault documentation, silo libraries and SCSI media changers are classified as removable media libraries.

Connecting to a Host Computer

This section offers guidelines for connecting drives and removable media libraries to host computers such as Silicon Graphics servers.

SCSI Connection Guidelines

A lot has been written about maximum SCSI cable length. This is not usually an issue because SCSI cables are available only in approved lengths. For single-ended SCSI, the maximum cable length is 6 meters, and 3 meters is the recommended maximum.

Differential SCSI improves signal integrity so data can be transmitted farther and faster than with single-ended connections. Differential technology doubles the number of signal wires, with each second wire carrying an inverted signal—because measuring the difference between two signals is more reliable than measuring a single binary signal. The recommended maximum cable length for differential SCSI is 25 meters.

In practice, any type of SCSI bus should be as short as possible with as few connections as required. The SCSI-2 standard allowed up to eight SCSI addresses, whereas SCSI-3 allows up to 16 addresses. On IRIX systems, address zero is reserved for the controller.

External devices must be terminated with an externally mounted SCSI port terminator on the rear of the drive. Terminators are not required on internally mounted drives, because internal termination is handled on the SCSI drive backplane.

Active terminators may be used to improve signal integrity on either single-ended or differential SCSI busses. Active terminators are usually battery powered and come with an LED to indicate that they are working. Some have an external power supply.

Sometimes you hear that you should cable high-speed devices closest to the SCSI controller, with low-speed devices further out. In practice, however, most SCSI busses are limited by the speed of the slowest communicating device, no matter what its position.

Table 7-2 compares the bandwidth of different SCSI types.

Table 7-2 SCSI Types and Speeds

SCSI Type	Data Word Size	Clock Speed	Bandwidth	Cable Length
narrow	8 bits	5 MHz	5 MB/sec	6 meters
wide	16 bits	5 MHz	10 MB/sec	6 meters
fast/narrow	8 bits	10 MHz	10 MB/sec	3 meters
fast/wide	16 bits	10 MHz	20 MB/sec	3 meters
Ultra fast/wide	16 bits	20 MHz	40 MB/sec	1.5 meter

SCSI-1, SCSI-2, and SCSI-3 refer to different protocols, with higher protocol numbers having additional commands and interfaces. People often mix up protocol with speed, bandwidth, and even connector type. In general, SCSI-1 is 5 MHz, SCSI-2 is 10 MHz, and SCSI-3 (or UltraSCSI) is 20 MHz. Three types of connectors are in wide use today:

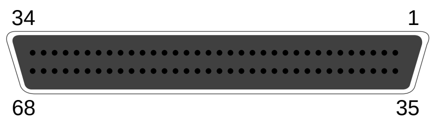

Figure 7-1 68-pin Wide SCSI-3 Connector

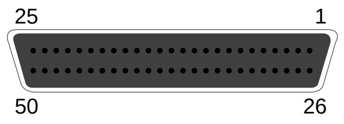

Figure 7-2 50-pin SCSI-2 Connector (Mini-micro)

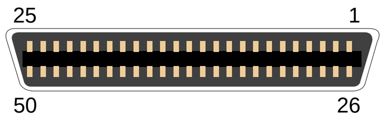

Figure 7-3 50-pin Centronics Parallel Connector

The first is the only type that can be used on fast and wide SCSI busses. The second and third types are functionally equivalent. The third type costs less than the second, but has a tendency to slow down the bus, and it should be used only with slow devices.

It is critical to put narrow devices at the end of a wide bus, with the wide bus terminated on the upper data lines and signals at the transition point. This results in fewer problems. Silicon Graphics sells 68-pin to 50-pin (both mini-micro and Centronics) SCSI cables that have termination built in to the connector at the wide end.

Sustained bandwidth is typically no more than 80% of the peak bandwidth. It depends on the quality of disk drives and communicating drives. Transfer rates decrease when you put too many devices on the same SCSI bus.

For maximum bandwidth, it is best to place two fast devices on different SCSI controller. For example, if you have three DLT 7000 drives intended for an Origin2000, attach each one to a separate SCSI bus on the XIO card. This way there will be little bus contention, and the Origin2000 server will be able to drive them all near their rated throughput.

Silicon Graphics Servers

The Silicon Graphics Origin2000 enterprise server can contain one to four node cards, each with one or two CPUs. The Origin2000 server is capable of transfer rates at 5 GBps sustained, and up to 6.24 GBps peak. In rack-mounted configurations, as many as 16 modules (64 nodes, 128 CPUs) can be connected with CrayLink. Each Origin2000 module can contain 12 XIO cards; a three-slot PCI board can be substituted for one of the XIO cards. Each Origin2000 module can hold five 3.5-inch UltraSCSI devices, usually disk, and one half-height 5.25-inch SCSI device. For mass storage, the following XIO card configurations are supported:

- four-port UltraSCSI (three differential, one single-ended or differential)
- UltraSCSI, 100BaseT Ethernet, two 460 Kbps serial ports

- three PCI slots, either 32-bit or 64-bit PCI cards supported
- two-port Fibre Channel (copper or fiber)

The Silicon Graphics Origin200 server tower includes two SCSI controllers with internal SCSI connectors. An Origin200 tower can accommodate six 3.5-inch UltraSCSI devices, usually disk, and two half-height 5.25-inch SCSI devices. Two towers can be connected to double capacity. For maximum flexibility, you choose how to configure external SCSI connectors. For example, an UltraSCSI adapter can be placed in one of three PCI slots. The Origin200 PCI bus is capable of delivering 267 MBps throughput. For faster aggregate bandwidth, the Origin200 server can be connected with CrayLink to an I/O tower offering the same choice of XIO cards as the Origin2000 server.

Storage Management Applications

This section gives an overview of software currently used for tertiary storage.

Scheduled Backup

The principal purpose of backup is to provide fall-back data in case of disaster. As a side benefit, backup allows users to recover files that they delete accidentally. In practice, the side benefit occurs more often, but is less important in the overall scheme of things.

In marketing literature, scheduled backup is often called “fully-automated lights-out” backup. This means that data backup occurs unattended, usually in the middle of the night when system load is low or nonexistent.

Suppose you have a server with 1 TB (terabyte, equal to 1000 GB) of data to safeguard. Dumping all the data to tape is called a “full” backup. Two DLT 7000 drives would take over 27 hours to copy this data, not including tape changing time (29 tapes are required). Fortunately not all data needs daily backup, because certain files seldom change.

The time-consuming and data-intensive nature of full backups led to the invention of incremental and differential backups. An incremental backup saves any data changed since the last backup. A differential backup saves any data changed since a full backup (level zero), or since a differential backup of lower level. Differential backups are also called “level” backups. Figure 7-4 shows a scheme using incremental backups during the week, and various levels of differential backups during weekends. Unlike incrementals, repeated level-one backups would go all the way back to a full backup.

Because incremental and differential backups save only files that change, less data is involved, reducing backup time and overall tape consumption. The downside is that more tapes are required to restore a filesystem, because full backups must be overlaid with increasing levels of differential backups, then with incremental backups.

Scheduled backup software should be able to include and exclude specific sets of files. Some filesystems can be recreated from software distributions, except for a limited set of files that should be saves. On UNIX systems, core dumps do not typically need saving.

The advent of robotic tape libraries has made the distasteful task of changing tapes for weekend backups all but a thing of the past. It has also reduced total backup time, because robotic libraries can change tapes as soon as they become full, rather than waiting for an operator to load a new tape.

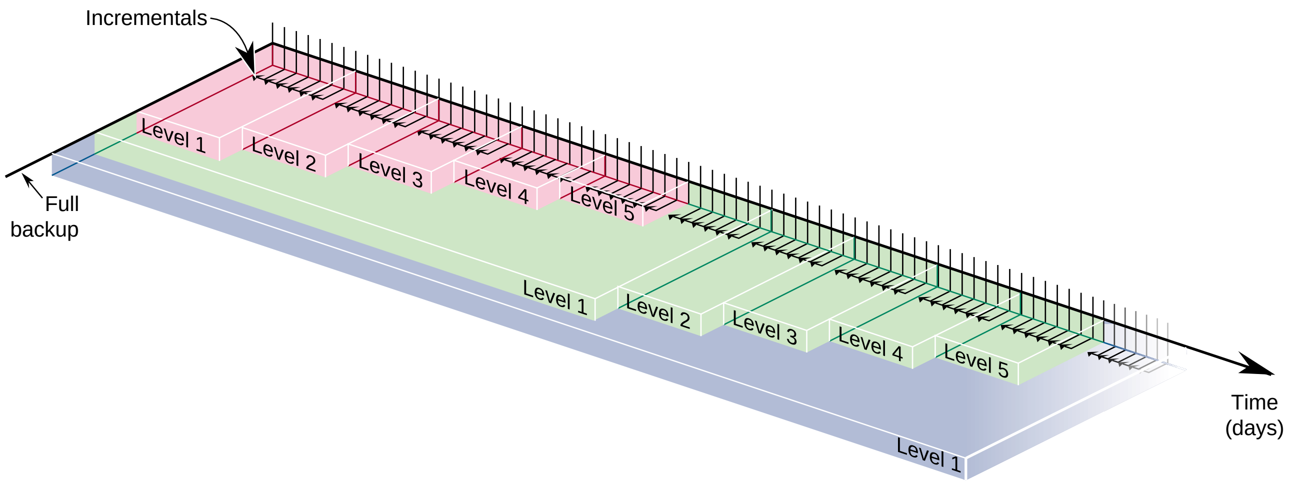

Figure 7-4 Incremental and Differential Backups

In most places, the backup window starts when the last employee goes home, and begins when the first employee arrives at work. (Backup is less reliable when files are changing.) Capacity planning should take into account the amount of data to be saved, the duration of the backup window, and the bandwidth of backup hardware.

Network backup has become critical now that productive work occurs on workstations and personal computers. Cost and performance considerations dictate that important data be kept on local disk. For convenience, and to ensure the integrity of backup data, workstations and personal computers can be backed up across the network. This is the usual client-server model—a server with huge tape capacity backs up a set of clients.

To reduce the workload for system administrators, backup software should provide convenient file retrieval for backup clients, including file search and version facilities. This brings up the security issue. Backup software must preserve normal system security, not allowing searching or recovery of other users' files. At the same time, administrators must be allowed to recover data for departed users, and to move data from one client to another when necessary.

Scheduled backup software should help you manage media, by including features for tape recycling, media aging, device cleaning, and perhaps bar code reading. Software should also be able to log backup failures due to power loss, hardware problems, and media bad spots, and notify administrators when intervention is required.

As stated above, the principal purpose of backup is to provide a cost-effective method for disaster recovery. For this to work well, you must create a disaster recovery plan and test all procedures to make sure they work correctly.

Storing backups offsite, in a fire vault, or both ways, is your only hope of recovery from fires and natural catastrophes. Clearly the frequency with which these are done equals the amount of work at risk: monthly offsites jeopardize a full month's data.

Supplemental Software

Database backup presents a challenge for backup software, because databases change internally, rather than on a per-file basis. Backing up a large database can take longer than the backup window allows. As a solution, database vendors often provide backup methods to save only changed data, and to roll in these changes if recovery is needed.

Taking a database offline for backup is simpler and faster, although many databases have 7-day, 24-hour uptime requirements. Saving from a live database is called "hot" backup.

Archiving involves taking a snapshot of data files as they reside on disk at a given time. The snapshot image is typically stored on removable media, such as tape or optical disc. Once the snapshot is safely stored, archived files may be deleted to conserve disk space. Whereas the goal of backup is to protect data against accidental loss or damage, the goals of archiving are to preserve data and to conserve online storage space.

Archiving is normally performed on data associated with specific projects, rather than on an entire system. Backup tapes are usually recycled or discarded, while archive media are intended to last a long time. For this reason, recordable CD is the ideal archive media, because it is more universal and permanent than tape.

Hierarchical Storage Management

HSM (hierarchical storage management) is a storage strategy that involves moving files from one medium to another, based on configurable a set of rules. One common rule is based on access rate—when a file becomes inactive, it get migrated. Storage hierarchy is usually governed by media cost and random access time. The goal of HSM is to conserve network storage resources, thereby providing users with a seemingly infinite storage capacity, at the lowest possible cost.

HSM was developed in the 1970s for use in mainframe applications, when disk storage was much more expensive than it is today, and tape storage was comparatively cheaper. According to one HSM manufacturer, between 60% and 80% of files on a typical system have not been accessed in 90 days, so HSM remains a viable strategy.

After migration, a stub file is left on disk as a link to the actual file on alternate media. When a user accesses a stub file, the HSM software locates the actual file on alternate media and restores the original data to disk. Most HSM systems are configured with three types of storage:

- online (hard disk drives)
- nearline (often magneto-optical jukeboxes for random access)
- farline (usually high capacity tape drives)

While hard disks have file access time in the millisecond range, optical jukeboxes have access times in the multiple second range. Tape libraries have file access times that vary widely depending on where data exists on tape, on the order of several minutes.

Most HSM software can be configured with a list of directories not to migrate. Also, the administrator can set high and low watermarks for migration time at each storage level, thereby controlling latency to suit user preferences.

Enterprise Storage Control

In large networks of heterogeneous systems, the management of scheduled backups can be a major chore. Several products are available to help deal with enterprise issues.

SNMP (simple network management protocol) has features to help manage backups in a network environment. Many network management products integrate SNMP support.

Alexandria, a high-performance backup product by Spectra Logic, can coordinate server and database backups across large networks, and includes facilities for cross-backup and storage node sharing.

NetWorker, a backup product for heterogeneous networks by Legato Systems, optionally includes GEMS (global enterprise management system) for managing storage nodes and enterprise backup scheduling.

OpenVault Error Messages

This appendix explains how to resolve error conditions, and includes a description of the possible error messages. The sections in this appendix are:

- "Error Conditions" on page 107
- "Accessing the OpenVault Message Log" on page 107
- "Error Messages" on page 107
- "OpenVault Processes and Files" on page 108

Most OpenVault errors are found in the log file */usr/OpenVault/var/OVLOG*. They can also be listed with the *ov_msg* **-l** option; see ov_msg(1M).

Error Conditions

Errors such as tape read and write errors are associated with an application error, whereas errors such as "`cannot find tape`" or "`cannot associate with driver`" are OpenVault errors.

Accessing the OpenVault Message Log

To view all the warning, error, and operator intervention messages that are contained in the message log on the default server, use the *ov_msg* command; see ov_msg(1M).

Error Messages

Refer to the OpenVault release notes for a listing of the error messages that apply to the OpenVault system.

OpenVault Processes and Files

Table A-1 shows some important OpenVault files.

Table A-1 OpenVault Configuration Files

OpenVault File	What it Controls
`var/corekeys`	File listing clients allowed OpenVault access, possibly with passwords.
`admin/keys`	File storing security passwords for OpenVault applications.
`var/OVLOG`	Repository of all events and errors that occur within OpenVault.
`mlm/config`	Configuration file for OpenVault server, MLM (media library manager).
`dcp/`*tapeN*`/config`	Configuration file for the drive named *tapeN*.
`lcp/`*libN*`/config`	Configuration file for the library named *libN*.
`dcp/`*tapeN*`/logfile`	Event and error log for the drive named *tapeN*.
`lcp/`*libN*`/logfile`	Event and error log for the library named *libN*.

Table A-2 shows some important OpenVault processes.

Table A-2 OpenVault Processes

OpenVault Process	What it Does
`ovroot`	If chkconfig(1) on, this process mediates and delegates service requests.
`MLM_catalog`	Grants services as requested, and manages the OpenVault catalog.
`MLM_capi`	Started by the server to manage each registered OpenVault application.
`MLM_aapi`	Started by the server to manage each OpenVault administrative application.
`MLM_dcp`	Started by the server to manage each attached drive and its related DCP.
`MLM_lcp`	Started by the server to manage each attached library and its related LCP.
`DCP`*tapeN*	The DCP (drive control program) that manages the drive named *tapeN*.
`LCP`*libN*	The LCP (library control program) that manages the library named *libN*.
`ssi | lmcpd`	Product-specific processes to manage ACSLS and IBM-3494, respectively.

Appendix B

OpenVault Reference Pages

This appendix provides a brief introduction to the reference pages for commands used to administer and operate OpenVault. Refer to the reference page itself for a complete description of options and usage information.

Table B-1 OpenVault Reference Pages

Name	**Use**	**Description**
ov_app(1M)	administrative	Adds, removes, or lists applications.
ov_cart(1M)	data entry	Allows entry or modification of cartridge information (PCL, cartridge group, attributes, and so forth).
ov_cartgroup(1M)	administrative	Adds or removes cartridge groups, and sets or modifies group priority and access permissions.
ov_carttype(1M)	status	Manages OpenVault cartridge types.
ov_dcstats(1M)	status	Displays drive and cartridge use statistics.
ov_drive(1M)	administrative	Adds, removes, or lists drives.
ov_drivegroup(1M)	administrative	Adds or removes drive groups, and sets or modifies group priority and access permissions.
ov_eject(1M)	operation	Ejects cartridges from library, by location or by cart ID.
ov_import(1M)	operation	Brings cartridges under OpenVault management.
ov_inject(1M)	operation	Injects cartridges into a library.—the library loading port opens so the operator can insert a cartridge.
ov_library(1M)	administrative	Adds, removes, or lists libraries.
ov_lscarts(1M)	status	Provides cartridge status, including cartridge type, state, location, group, association with application, partition and volume mapping, and so forth.
ov_lsvols(1M)	status	Provides volume status, including names, partitions, association with application, and containing cart ID.

Table B-1 (continued) OpenVault Reference Pages

Name	Use	Description
ov_mount(1M)	operation	Mounts a cartridge in preparation for read or write by an application. Options specify cart ID and a particular drive, media side, partition, volume, or application.
ov_msg(1M)	administrative	Administers the message system. Allows for listing and deleting messages, and setting levels or limits.
ov_part(1M)	data entry	Sets or modifies media partition information to prepare for volume setup. Information includes partition name, side, partition size, whether allocatable, and so forth.
ov_purge(1M)	operation	Removes the catalog entry for the cartridge. This should be used only when a cartridge is to be destroyed.
ov_recycle(1M)	operation	Makes unused cartridges available for reuse.
ov_shutdown(1M)	administrative	Shuts down and restarts OpenVault.
ov_slotttype(1M)	status	Manages OpenVault slot types.
ov_stat(1M)	status	Provides status about OpenVault servers and components, including up or down state, configuration information, slot maps of libraries, application registration status, task queues, missing DCP and LCP software, and so forth.
ov_system(1M)	administrative	Sets or retrieves information from the system table for system-level control.
ov_task(1M)	operation	Removes a task request from the server task queue.
ov_unmount(1M)	operation	Unmounts cartridge from a drive. Options specify cart ID and a drive, media side, partition, volume, or application.
ov_vol(1M)	data entry	Allows entry or modification of volume information, including volume name, associated application, attribute name/value pairs, and so forth.

Index

Tell Us About This Manual

As a user of Silicon Graphics products, you can help us to better understand your needs and to improve the quality of our documentation.

Any information that you provide will be useful. Here is a list of suggested topics:

- General impression of the document
- Omission of material that you expected to find
- Technical errors
- Relevance of the material to the job you had to do
- Quality of the printing and binding

Please send the title and part number of the document with your comments. The part number for this document is 007-3211-004.

Thank you!

Three Ways to Reach Us

- To send your comments by **electronic mail**, use either of these addresses:
  - On the Internet: techpubs@sgi.com
  - For UUCP mail (through any backbone site): *[your_site]*!sgi!techpubs
- To **fax** your comments (or annotated copies of manual pages), use this fax number: 650-932-0801
- To send your comments by **traditional mail**, use this address:

  Technical Publications
  Silicon Graphics, Inc.
  2011 North Shoreline Boulevard, M/S 535
  Mountain View, California 94043-1389