Vault Manager Guide

Optegra® Release 6

DOC40154-007

Parametric Technology Corporation

Parametric Technology Corporation, 140 Kendrick Street, Needham, MA 02494-2714
8 January 2001

Table of Contents

Preface

Introduction to Vault Management

Setting Up and Managing Vault

Giving Users Access to Vault

Creating and Modifying Authority Scheme Components

Setting Up and Modifying a Project

Setting Up Release/Revision Controls

Setting Up and Modifying User-defined File Attributes

Deleting Files and Log Entries

Message Identifiers

Examples of Revision Code Sequences

Corresponding Commands

Customizing the Interface

Glossary

Preface

The Vault Manager Guide contains information and instructions for determining how Vault is used at your site. It is for administrators responsible for implementing and maintaining Vault.

This document explains the product that allows you to perform the following types of work:

- Set up the Vault
- Add Users to the Vault
- Manage the Vault
- Correlate command line functionality with menu options

Related Documents

The following documents may be helpful as you use *Vault Manager Guide*:

- *Installing Oracle for Optegra Applications*
- *Using the License Manager*
- *Vault End User Guide*
- *Vault System Administrator Guide*
- *Vault Command Reference*

Book Conventions

The following table illustrates and explains conventions used in writing about Optegra applications.

Convention	Example	Explanation
EPD_HOME	`cd` $EPD_HOME`/install` (UNIX) `cd` %EPD_HOME%`\install` (Windows)	Represents the default path where the current version of the product is installed.
Menu selections	Vault > Check Out > Lock	Indicates a command that you can choose from a menu.
Command buttons and options	Mandatory check box, Add button, Description text box	Names selectable items from dialog boxes: options, buttons, toggles, text boxes, and switches.

User input and code	`Wheel_Assy_details`
	
	`-xvf /dev/rst0`
	
	`Enter command>` **`plot_config`**
		Enter the text in a text box or on a command line.
		
		Where system output and user input are mixed, user input is in bold.

System output	`CT_struct.aename`	Indicates system responses.
Parameter and variable names	`tar -cvf /dev/rst0` filename	Supply an appropriate substitute for each parameter or variable; for example, replace filename with an actual file name.
Commands and keywords	The `ciaddobj` command creates an instance of a binder.	Shows command syntax.
Text string	`"SRFGROUPA"` or `'SRFGROUPA'`	Shows text strings. Enclose text strings with single or double quotation marks.
Integer	n	Supply an integer for *n*.
Real number	x	Supply a real number for *x*.
#	`# mkdir /cdrom`	Indicates the root (superuser) prompt on command lines.
%	`%` `rlogin` remote_system_name `-l root`	Indicates the C shell prompt on command lines.
$	`$` `rlogin` remote_system_name `-l root`	Indicates the Bourne shell prompt on command lines.
>	`> copy` filename	Indicates the MS-DOS prompt on command lines.
Keystrokes	`Return` or `Control-g`	Indicates the keys to press on a keyboard.

Online User Documentation

Online documentation for each Optegra book is provided in HTML if the documentation CD-ROM is installed. You can view the online documentation from an HTML browser or from the HELP command.

You can also view the online documentation directly from the CD-ROM without installing it.

From an HTML Browser:

1. Navigate to the directory where the documents are installed. For example,

   $EPD_HOME`/data/html/htmldoc/` (UNIX)

   %EPD_HOME%`\data\html\htmldoc\` (Windows NT)
2. Click `mainmenu.html`. A list of available Optegra documentation appears.
3. Click the book title you want to view.

From the HELP Command:

To view the online documentation for your specific application, click HELP. (Consult the documentation specific to your application for more information.)

From the Documentation CD-ROM:

1. Mount the documentation CD-ROM.
2. Point your browser to:

   CDROM_mount_point`/htmldoc/mainmenu.html` (UNIX)

   CDROM_Drive:`\htmldoc\mainmenu.html` (Windows NT)

Printing Documentation

A PDF (Portable Document Format) file is included on the CD-ROM for each online book. See the first page of each online book for the document number referenced in the PDF file name. Check with your system administrator if you need more information.

You must have Acrobat Reader installed to view and print PDF files.

The default documentation directories are:

- $EPD_HOME`/data/html/pdf/`doc_number`.pdf` (UNIX)
- %EPD_HOME%`\data\html\pdf\`doc_number`.pdf` (Windows NT)

Resources and Services

For resources and services to help you with PTC (Parametric Technology Corporation) software products, see the *PTC Customer Service Guide*. It includes instructions for using the World Wide Web or fax transmissions for customer support.

Documentation Comments

PTC welcomes your suggestions and comments. You can send feedback in the following ways:

- Send comments electronically to `doc-webhelp@ptc.com`.
- Fill out and mail the PTC Documentation Survey located in the *PTC Customer Service Guide.*

Chapter 1

Introduction to Vault Management

This chapter provides an introductory description of Vault and an overview of the tasks performed by Vault managers.

- What Is Vault?
- What Does Vault Do?
- Overview of Vault Manager Tasks
- Prerequisites for Using this Book

Please note: See Appendix C, "Corresponding Commands" to correlate command line syntax with the graphical user interface. See the *Vault Command Reference* for elaboration on Vault commands.

What Is Vault?

Vault is a data management tool that helps automate the entire process of product development. With Vault, you can centralize control of design databases and ensure that there is no unauthorized access to product data. The purpose of Vault is to make it easier for you to manage the process of designing and manufacturing a product.

Vault provides concurrent control—only one user at a time can modify an Vault-controlled file. Vault includes facilities that back up, archive, restore, and recover data. Other facilities maintain information about users so only authorized users can access the database. Vault also includes a report generator (Interactive Query Facility).

It also provides a framework for organizing the work at your site into projects. You can define the stages of work within each project. You can customize authorities assigned to users. Users can be assigned to one or more projects with the same or different authorities for each project. You can define user lists so that Vault alerts specified users when it is time to review a file, part, or file set.

Vault is made up of:

- Vault Client
- Programming

Vault Client

Vault Client is the software interface between users and Vault. It operates on all platforms. Vault Client provides several user interfaces.

Navigator is available through the graphical user interface on UNIX systems. You use Navigator to view and modify a hierarchical tree representing an assembly or product structure stored in Vault.

Programming

Vault Programming allows you to write programs that call the Vault routines. These are the same routines that Vault calls when you execute the various Vault commands.

What Does Vault Do?

Vault performs the following functions:

- Centralizes storage for many kinds of files
- Controls data access in a distributed environment
- Limits access to the database to authorized users
- Ensures that only one person at a time can modify a file
- Provides easy-to-use tools for archiving, restoring, backing up, and recovering Vault files
- Allows designers and engineers to transfer individual files and parts as well as sets of files and parts to and from the Vault database
- Provides a method for relating a group of files and/or parts so that you can treat them as a single unit
- Includes a report generator
- Provides a framework for organizing the work at your site into projects
- Permits creation of an authority scheme for each project
- Allows each user to have a different authority for each job or project
- Differentiates among the following categories of files:
    - File revisions
    - Files belonging to different projects
    - In-work versus released files
    - Private files
    - Public files
- Maintains user lists that you define and informs designated users when to review a file that is ready for a status change
- Provides a mechanism for sending and reading messages
- Programming allows you to develop applications that can use Vault functions.

Overview of Vault Manager Tasks

Vault is designed so that you can install it and begin using it. Once in use, you can tailor aspects of Vault to better meet your needs.

Alternatively, you can spend some time planning your implementation of Vault. Then you can implement your plan before you begin using Vault.

A good way to decide which alternative is best for you, is to read this book. In addition, your service representative can provide valuable assistance.

The tasks you must perform as a Vault manager are summarized below. The order of the tasks depends on whether you decide to use and then tailor Vault or plan first and then implement your plan for Vault.

- Gain an understanding of each aspect of Vault. This includes
  - Types of users
  - Classification of data (public, project, or private)
  - Authority levels
  - How to divide work into projects
  - Revision levels
  - File sets and parts
  - User-defined input parameters, defaults, and file attributes
  - Release/revision control procedures
- Learn how to use Vault.
- Move existing data into Vault.
- Give users access to Vault and to the appropriate commands and data.
- Create authority schemes that provide needed security.
- Divide work into projects.
- Set up release/revision control procedures.
- Maintain the Vault database by archiving, restoring, and deleting data.
- Create reports about various aspects of Vault.

Prerequisites for Using this Book

Load the Vault software on your system, along with the relational database management system product (RDBMS) you are using.

This book assumes that you have a plan for your implementation of Vault before you perform the tasks described in this book. See *Vault System Administrator Guide* and your service representative for help in formulating a plan.

You must have the necessary authority to perform the function described. In most cases, this means having the command in your `public` command list. For project-related commands, the command must appear in your `project` command list.

Chapter 2

Setting Up and Managing Vault

This chapter provides information about users, data classifications, authority schemes, projects, and user-defined entities needed by Vault so it can meet the specific needs at your site. It presents the following topics:

- Introduction
- Vault Users
- Public, Project, and Private Files
- File Sets
- Command Lists
- Authority Schemes
- Projects
- Revision Codes for Vault
- Parts
- Attributes
- User-defined Input Parameters
- Setting Up Vault Defaults
- Sequence of Steps for Setting Up Vault
- Using the Distributed Vault
- Overview of Database Maintenance
- Introduction to the Vault Process Log
- Changing Your Password

Introduction

You can tailor Vault to reflect the product development processes at your site. Before you implement Vault, analyze how your company does business. When you clearly understand the sequence of events and who is responsible for doing what, you can apply those facts to designing a data management implementation to meet your needs.

This chapter describes the aspects of Vault that you must understand to set up and manage Vault.

Vault Users

Who will be using Vault? What part of the product will they be using? Vault users at your site will probably fit into these groups:

- System administrators
- Vault managers
- Project leaders
- CADDS 5i operators
- Designers, engineers, and other non-CADDS end users
- Vault operators
- Database administrators
- Reviewers
- Miscellaneous users

Each group of users has a different set of responsibilities; consequently, each needs to perform different types of Vault functions. Since Vault functions are performed by using one or more commands, each group of users needs authorization to use a different group of commands. A command list is a list of commands that a user can execute.

Each user must have a unique Vault User ID and password. Before users can execute any Vault commands, they must enter the ID and password to sign into Vault. When you assign a User ID and password you also assign a previously defined command list and other authority attributes. Vault uses the authorities associated with a user's ID to determine whether a user can execute a command or access a file.

System Administrators

System administrators are most concerned with using the Network Services Manager (NSM) to set up and maintain the network that Vault uses. Also, they are usually responsible for maintaining storage pools and performing backup and recovery functions.

Vault Administrator

There should be at least one Vault admisitrator at your site. You decide how much authority to give to the administrator. For complete access to all Vault functions, you can assign the administrator to each project and also assign the highest public and project authorities to the administrator. A Vault administrator can delegate some tasks to project leaders, operators, or other users. Typical responsibilities include

- Defining command lists
- Setting up the authority scheme for public and private files
- Assigning User IDs and passwords
- Assigning a set of authorities to each user
- Setting up user lists
- Defining projects and their authority schemes
- Assigning users to projects
- Archiving, deleting, recovering, and restoring Vault files
- Backing up Vault files

Please note: You can grant administrative privileges to Vault users who are not Vault administrators. Users with administrative privileges

- Do not need to enter file passwords for those files they have the authority to access
- Can reset, update, and replace files signed out to another user (execute the `reset`, `update`, and `replace` commands for files not signed out to them)
- Can change the status of a file (`chgfsc` command) without proceeding in a sequential order through the status codes.

These privileges apply to files they have the authority to access.

Project Leaders

You can assign a project leader to each project. The project leader can have many of the same responsibilities as the Vault administrator, but within the assigned project only. Project leaders can define the authority scheme for their projects and assign users to their projects. With good reason, they can also override some security measures for files assigned to their projects.

CADDS Operators

CADDS operators are designers and drafters who need to modify files stored in the database. They execute these CADDS commands with Vault modifiers:

- `GET PART`, `PUT PART`, and `UPDATE PART`
- `GET FILE` and `PUT FILE`
- `GET FILESET` and `PUT FILESET`

Designers, Engineers, and Other Non-CADDS End Users

Non-CADDS users need to execute Vault file access commands. They also use commands that modify file attributes and automate review procedures.

Operators

You might want to delegate routine maintenance tasks to an operator. An operator can perform the tasks below without access to any affected files.

- Complete (universal) and incremental backups of the database
- Deleting files marked for deletion
- Restoring files marked for restoration
- Archiving files marked for archiving

Database Administrators

The RDBMS administrator, someone familiar with the operation of the RDBMS that Vault uses, ensures that the RDBMS provides the resources needed by Vault.

Reviewers

Some users need to review files in the database. Reviewers determine which files are ready for a status change. Reviewers would primarily need authority to read files and register approval or rejection of the content.

Miscellaneous Users

If you store text files in the database, some users might want to access those files on the Vault host computer. These users would primarily use the file access and file maintenance commands. There might be some users who are notified about a review although they are not reviewers themselves.

Public, Project, and Private Files

Each file stored in the database is assigned one classification:

- `PUB` for public files
- `PRO` for project files
- `PRI` for private files

You designate the file classification when you store a file in the database for the first time. A file can have only one classification at a time. Once set, you can change a file classification if certain conditions are met.

Anyone with the proper authority can access a public file. Private files belong to the user who created them. Only the owner of a private file can read and modify it.

Project files belong to a particular project and are controlled according to the authority scheme for that project. A project is a set of files that can be defined and managed as a unit. While each project must have its own authority scheme, that scheme need not be unique. Projects are intended to be used for logical segments of activity at your site.

Guidelines

You can choose whether to use one, two, or three file classifications. An alternative is to have only project files. Every file in the database would be assigned to a project. This would eliminate the need for a public authority scheme. (If you choose this alternative, remember to delete the two public status codes that Vault comes with—`IW` and `RL`.)

Another alternative is to use all three classifications. When implementing this, you should clearly define which files are considered public and which files are private.

File Sets

A file set is a group of files that are logically related. There are Vault commands to define a file set name and associate files with that name. Files with different classifications can belong to the same file set.

Some commands that manipulate files in the database can operate on a file set as well as on individual files. Vault executes the command for all or none of the file set members.

This can save time when you want to perform the same operation on related files. It also minimizes the possibility of leaving out important information.

Creating a file set that includes a CADDS part and all related text files is one way to use file sets.

Command Lists

Whether or not you can execute a particular command depends on your assigned command list. When you give a new user an ID and password, you also assign the user a public command list. Users can execute only commands in their assigned list. For commands that operate on files, users can execute commands in their public command list only on data classified as `PUB` (public). To operate on project data, users need the appropriate commands in their project command list for the corresponding project.

Requirements

For users to successfully execute a command that manipulates a file, the command must be in the user's command list, and the user must have authority to access the file. A command in a user's assigned command list does not guarantee that the user has the authority to access the file that the command operates on. It only means that the user has permission to execute the command.

Guidelines

You can organize command lists to include a command list for manipulating public files, a command list for each project assignment, a single command list for all files, or any combination of these. You might plan a command list for each group of users. Or, if there are few users, you can create a personal command list for each user.

Authority Schemes

An authority scheme is a series of status levels, status codes, authority numbers and command or function lists assigned to users. It determines whether a user can access a particular file at a particular stage of work.

Public and private files are controlled by the public authority scheme.

Project files are controlled by the authority scheme for their project. Each project has its own authority scheme. If you choose to have only project files, you do not need a public authority scheme.

When you are ready to implement Vault at your site you will be defining status levels and assigning authority numbers to users.

Each status level reflects a stage of work that occurs at your site. Each file moves through this series of status levels until it is complete. Different users need different kinds of access to the file at various stages in its development.

To define a meaningful series of status levels, it is crucial that you have a clear understanding of

- How work flows in your company
- The function of each part of a status level

Chapter 4, “Creating and Modifying Authority Scheme Components” describes the parts of a status level in detail and describes the relationships among users, authority numbers, and status levels.

Projects

A project is a set of files that can be defined and managed as a unit. For example, an aircraft company might have one project devoted to building a jet and another project devoted to building a cargo plane.

Creating a Project

To create a project you define a project ID and a set of status levels associated with the project ID. You also assign files and users to that project.

These Vault commands allow you to define a project:

- `addp` defines a project ID and some attributes of the project
- `chgp` changes one or more attributes of an existing project
- `adds` defines a status level for a project
- `chgs` changes one or more attributes of an existing project status level

Authority Schemes for Projects

Once a project is defined in Vault, you can assign someone to the project as project leader. This person can set up status levels for the project and then assign the users and files to the project.

While each project must have its own set of status levels (distinguished by its project ID), more than one project can have the same set of status levels. You or the project leader can use the administrative copy (`admcopy`) command to copy a set of status levels from one project to another. Vault controls project files according to the project's authority scheme.

Users assign files to a project when they store them in the database for the first time. You can use the change file classification (`chgfcl`) command to change a file classification or move a file from one project to another.

Requirement

Users must be assigned to a project to access any files that belong to that project. When you assign a user to a project, you also assign

- A command list—Commands the user can execute within that project.
- Read authority number or read authority group name—Determines which files belonging to that project the user can read.
- Write authority number or write authority group name—Determines which files belonging to that project the user can modify.

User Authorities

Users' authorities can be different for each project they are assigned to.

Commands for Assigning Users to Projects

The following three commands allow you to assign users to projects:

- `addup` assigns a user to a project
- `chgup` changes one or more attributes of a user assigned to a project
- `remup` removes a user from a project

Revision Codes for Vault

The revision code of a Vault file distinguishes the design cycle of the file. A design cycle is the sequence of phases a file passes through until it is complete. During a design cycle, a file has a sequence of different status codes, but usually only one revision code.

Assigning a Revision Code to a File

When you store a file in the database for the first time you can specify a revision code. When you do not specify a revision code, Vault automatically assigns the initial revision code in the default revision code sequence.

The file keeps this revision code (unless you change it) for the remainder of the design cycle from in-work through release. When a file is approved for release, the in-work status code changes to a released status code. This copy of the file maintains the original revision code.

Modifying a Released File

A released version of a file cannot be modified. Thus a file is not released until it is finished and approved. (Finished implies here that a particular version of the file will no longer change.) If modifications are needed to a released file version, a new design cycle begins. Vault creates a file copy and assigns the next revision code. The file with the new revision code is a new file.

Users can modify new revisions of a file if they have the proper authority. When they execute a `get` command for a released file, the copy they receive has a revision code that is higher than the released version. Vault assigns the initial in-work status code in the associated set of status levels to the copy of the file with the new revision code. Multiple revisions of the same part can exist in the database and can be accessed.

Examples of Revision Code Sequences

Appendix B, “Examples of Revision Code Sequences” provides examples of revision code sequences and the consequences of changing the revision code sequence associated with a project.

Default Revision Codes

The Vault utility that loads revision codes (`ldedmrev`) allows you to load into Vault the exact revision codes you want to use at your site. During Vault software installation, you select a default revision code sequence or you use the local text editor to create a file containing your revision codes.

The name of the default revision code sequence established during installation is `DEFAULT`. You can change the content of the `DEFAULT` revision code sequence any time after installation by running the `ldedmrev` utility. See the installation guide for details.

Vault uses the default sequence for public and private files and for project files if you specify it when you add the project to Vault. To specify a revision code sequence different from the default sequence, execute the `chgp` (change project) command. To change the default sequence for public and private files, leave the Project ID blank when you execute the `chgp` command.

Multiple Revision Code Sequences

You can define multiple revision code sequences. For example, you might use one sequence for prototype parts and another sequence for production parts. The `addrs` (add revision sequence) and `delrs` (delete revision sequence) commands allow you to define and delete revision code sequences.

You can associate a revision code sequence with a project. More than one project can have the same revision sequence. When you do this, the associated revision code sequence becomes the default sequence for that project.

DEFAULT and NONE

Two reserved words indicate special revision sequences.

`DEFAULT` is the name of the default revision code sequence that is established during Vault installation.

`NONE` is the name of an empty revision code sequence. Assign `NONE` when no revision sequence is associated with the project. When files assigned to a project with no revision sequence are stored in the database, Vault does not assign a revision code.

Requirement

A file must have a revision code before Vault can change its status code from an in-work status code to a released status code.

Guideline

When implementing the review procedure for a file that is moving from an in-work status to a released status, instruct the last reviewer to make sure the file has a revision code. Alternatively, users can make sure a revision code has been assigned before initiating reviews.

Changing the Revision Code of a File

You can change the revision code of a file when it has an in-work status. The file must not be associated with a revision code sequence. The `chgfrev` (change file revision code) command allows you to do this.

You cannot change the revision code of a released file.

Parts

A Vault part is a collection of files known as a unit to an application such as CADDS or MEDUSA. An operation performed on a part is an all or nothing transaction. For example, changing the status code of a part changes the status code of all files that are members of the part. Executing the `GET` command for a part transfers all files belonging to that part. If any one of the files cannot be transferred, then no files are transferred.

Vault supports CADDS parts and custom (site-defined) parts.

CADDS Parts

A CADDS part is a special set of files which you can operate on. It is generated by the CADDS application.

Custom Parts

A custom part is a part whose member files are defined by you. You must use the Custom Part Facility to implement a rulebase that determines which files are included in your part.

How Vault Handles Parts

A part has a unique name, revision, an internally maintained version number, a part type, and several other hard-coded attributes. Vault stores this information in the part-directory table.

When you store files with Part as the selection scope, you specify the name of the part. Vault places a record in the part-to-file table for each file that belongs to a part. The record shows the name of the part that the file belongs to. When you issue a command that acts on a part, Vault looks at the part-to-file table to determine which files belong to the part.

In Vault, a file is either a member of a part or a standalone file (that is, not a member of a part). A Vault part file can belong to only one part. A part member file must have the same classification (public, project, or private), project ID, revision code, and status code as the part to which it belongs.

Most operations on part files are accomplished by operations on the entire part. Certain file operations in Vault are not allowed on a file that is a member of a part. In particular, you can issue the Vault commands `reqrvw` (Request Review),

`chgfsc` (Change File Status), `chgfcl` (Change File Classification), and `chfrev` (Change File Revision) only for a part or stand-alone file, not a part member file.

Flexible CADDS Part Definition

During Vault software installation, the standard list of the file types included in a CADDS part is loaded. You can customize this list at any time after installation.

If the standard definition of a CADDS part meets your requirements, you need not do anything. Perform this step only when you want to change the standard file types included in a CADDS part.

To change the file types included in a CADDS part, you edit the file that lists the current file types included in a part. The next table lists CADDS file types.

The `ldedmcpd` (Load EDM CADDS Part Definition) utility allows you to load your definition into Vault. Once loaded, all commands that access parts will use your definition of a CADDS part.

See the *Vault Server Installation Guide* for information about using the `ldedmcpd` utility.

Custom Part Facility

The Custom Part Facility allows you to define rules for which files are included in a part that is not a CADDS part. With this Facility, you can implement user-defined rulebases that allow Vault to recognize different groups of files as parts.

Table 2-1 Description of Custon Part Files

File Type	Description
0	Not defined
1	CADDS Catalogs
2	Object code
3	Text files
4	Configurations
5	Currently unassigned
6	PEP object code
7	Overlays (executable files)
8	Command tables
9	Loader data tables (symbol tables or libraries)
A	Accounting tables
B	Symbolic trace table file
C - E	Currently unassigned
F	Z80 binary files
20	CADDS part files
21	CADDS TVF files
22	CADDS figure files
23	CADDS-related part files
24	CADDS modifier or message tables
25 - 2F	CADDS-related part files
30 - 3F	APU files
40 - 44	CADDS user interface files
46 - 47	SIMS database files
48 - AF	Currently unassigned
B0	Batch files
B1 - B3	Mail files
B4 - BF	Currently unassigned
C0 - CF	Program overlay files
D0 - FE	Currently unassigned
FF	Work files

What Is a Rulebase?

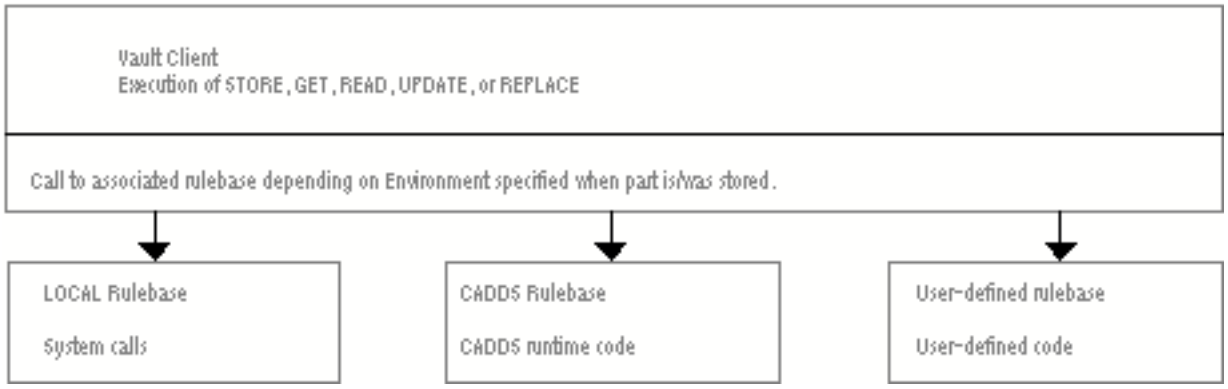
A rulebase is a set of rules that defines what files a part consists of for a specific application and how they should be handled for that application. You can implement a rulebase with an operating system command language such as the UNIX shell commands or compiled C source code.

With the Custom Part Facility, you can write your own rulebase. A user-defined rulebase can only support parts. Your rulebase cannot have knowledge of a MEDUSA catalog, for instance, or even an individual MEDUSA file unless it is a member of a MEDUSA part.

Vault has always included two rulebases—one for CADDS and one for local files. The `CADDS` and `LOCAL` rulebases are not limited to handling only parts.

When users execute the `STORE` command, the value specified for the Environment parameter determines which rulebase Vault uses. This is illustrated in the following figure.

Figure 2-1 Environment Parameter Determines Which Rulebase Vault Calls



The CADDS rulebase is the only rulebase that can handle CADDS parts. When you define your own rulebase, it handles only parts, but they are not considered to be CADDS parts.

User Interface

When you implement a user-defined rulebase, the only change that End Users see is that there is a new value they can specify for the Environment parameter when they execute the `store` command. You need to tell them what this value is and when they should use it.

When Is a Rulebase Applied?

The environment value you assign when you store a part always remains the same; once stored, you cannot change a part's environment value. Each subsequent execution of the `update`, `replace`, `get` or `read` command uses the rulebase associated with the environment that was specified when the part was stored.

Steps for Implementing a User-defined Rulebase

You can add a new value to the Local Rule list box on the Store In Vault dialog box in the Graphical User Interface:

1. Open the `Edmgui` resource file.
2. Type the following line:

   ```
   Edmgui*legalRulebaseValues: CADDS,LOCAL,myrulebase
   ```

   where myrulebase is the name of the rulebase you want to add to the Local Rule dialog box.

Please note: The syntax is strict:

   a. Match the rulebase values to the names in the `EDM.DEFAULTS` file exactly.
   b. Delimit values with commas, but no extra spaces.
   c. Include CADDS and LOCAL on this line, if you want them to remain in the Local Rule list.

Attributes

You can use attributes to hold information that keeps you informed about your files. You can use IQF to view the attributes for a given file or to view certain files/parts using their attributes as qualifiers. For example, you can list parts where the attribute name `clearance` has the value *`Secret`*.

For detailed information regarding user-defined attributes, refer to Chapter 7, “Setting Up and Modifying User-defined File Attributes”.

Planning Enough Space for Attributes

If your plan is to assign an average of one or two attributes to each file stored in Vault, then you need not be concerned about space allocation in the RDBMS. Vault already allocates enough space to accommodate your user-defined attributes.

If, however, your plan is to assign more than a few attributes to each file, you must consider whether the RDBMS is set up with enough space. You must answer such questions as, “How many attributes will be assigned to each file?” and “How often will the `store, update, and replace` commands be performed?”

There is a one-to-one correlation between an attribute assigned to a file and a row of information in the `ATTR_DATA` table (the RDBMS table that stores the attributes). Each time a file is stored, updated, or replaced, Vault creates a new internal version of the attribute information in the RDBMS.

You probably do not need to allocate additional space for attributes if you:

- Do not have more that one or two attributes assigned to each file
- Perform regular incremental backups and regularly delete old file versions
- Do not perform a `store, update, or replace` command more than once a day for each file

But if one or more of the reverse conditions are true at your site, then you must consider assigning additional space to the RDBMS. How much additional space depends on:

- How many files have attributes
- How many attributes are assigned to each file
- How often a file is stored, updated, or replaced
- How often you perform incremental backups
- How often you delete old file versions

The average row length for attribute data is 60 bytes. The row length for the attribute data index is 25 bytes. With a 250k extent for data space and a 156k extent for index space, the RDBMS can store 3500 average length rows. With a one megabyte extent for data space and a 622k extent for index space, the RDBMS can store 14,000 average length rows.

See the installation guide for information about assigning additional space for the RDBMS.

User-defined Input Parameters

Some Vault commands include user-defined input parameters. You can use these input parameters for anything useful. Vault does nothing with the information except save it.

For example, when you execute the `store` command to add a new file to the database, Vault prompts for several optional parameters including system type, user-defined type, and file description. If this information is useful, fill it in.

The following worksheet lists the user-defined input parameters for each command. The length indicates the maximum number of alphanumeric characters you can input. Describe the use in the last column.

Command(s)	Prompt	Keyword	Length	Description
`addfs`	File set description	fsdesc	30	_____
`addmfs`	Description	memdesc	30	_____
`addp/chgp`	Project name	projname	15	_____
	Contract number	contract	25	_____
	Unit of measurement	units	5	_____
	Description	projdesc	30	_____
`adds/chgs`	Status description	statdesc	30	_____
`addu/chgu`	User full name	fullname	25	_____
	User last name	lastname	15	_____
	User description	userdesc	30	_____
	User group	usergrp	2	_____
`addul`	User list description	listdesc	30	_____
`chgfa, store reserve`	System type User-defined type File description Part number Group technology code	systype usertype filedesc partnum gtcode	8 8 25 20 32	_____ _____ _____ _____ _____
`chgfsc, rsvp reqrvw`	Comments	comment	240	_____

Setting Up Vault Defaults

Vault provides defaults that you can customize.

- You can customize defaults for some Vault command parameters by listing your preferred default in the EDM.DEFAULTS file. You can find or create the EDM.DEFAULTS file in the $EDM_HOME/data directory. You can also have an individualized copy in your current working directory and/or home directory.
- You can determine the revision code sequence for public and private files. You can do this during Vault installation or at any time after installation. You use the ldedmrev (Load EDM Revisions) utility. See your installation guide.

Creating the EDM.DEFAULTS File

If you install a new version of Vault by running edminstall, then Vault creates the EDM.DEFAULTS file for you. Refreshing from a previous version of EDM or Optegra recreates the EDM.DEFAULTS file.

Location of Rulebase Executables

The CADDS variable specifies the path name of the CADDS rulebase. The LOCAL variable specifies the path name of the LOCAL rulebase.

When you install Vault, the system creates the EDM.DEFAULTS file and includes entries for CADDS and LOCAL. You should not change these entries.

If you define a new rulebase for use at your site, you must specify its name in the EDM.DEFAULTS file. The name of the rulebase is the value users can enter for the environment parameter when they execute the STORE command. Set the name of the rulebase equal to the path name of the rulebase executable.

If you define a new rulebase for use at your site, you must specify its name in the EDM.DEFAULTS file. The env and selscope parameters, which apply to specific rulebases, are set equal to the full path names of the rulebase executables. The full path name of the rulebase can serve as the environment or selection scope parameter for the STORE command.

The ENV variable is used directly by the STORE command. It is used indirectly in conjunction with the OBJECT-CLASS parameter by the following commands: get, read, replace, update. The following syntax in the EDM.DEFAULTS file sets the env parameter for each rulebase:

ENV(rulebase_name)=fully_qualified_name_of_executable

This syntax is the pattern for the following entries:

```
ENV(LOCAL)=$EDM_HOME
ENV(CADDS)=$EDM_HOME
```

Any user-written rulebase must have an ENV entry similar to the preceding ones.

The `selscope` parameter is used by the store command. The following syntax in the `EDM.DEFAULTS` file sets the allowed selection scopes for the store parameter for each rulebase:

```
selscope(rulebase_name)=selscope_values
```

This syntax is the pattern for the following entries:

```
selscope (CADDS)=F,P
selscope (LOCAL)=F,D
```

Any user-written rulebase must have a selscope entry similar to the preceding ones.

Case Sensitivity

The `EDMCASE` variable determines whether Vault converts lowercase selection name input to uppercase or accepts lowercase selection name input as is.

Set `EDMCASE` equal to Mixed to recognize and preserve lowercase input for selection names.

It is recommended that you set `EDMCASE` equal to Mixed so that the rulebases can set Vault names correctly, preserving lowercase input.

When you do not include `EDMCASE` in the `EDM.DEFAULTS` file, Vault converts all file, part, and file set selection names to uppercase.

`EDMCASE` has precedence over any case rules in the rulebase associated with the file or part.

Location of Command Audit Files, Menu Profile, and Other Files

The EDMTEMPDIR variable specifies a path name for the directory in which you want Vault to place your command audit files, Vault menu profile, and other files produced by Vault.

When you do not include EDMTEMPDIR in the EDM.DEFAULTS file, Vault places command audit files, your Vault menu profile, and any other files produced in your current working directory.

For example, if EDMTEMPDIR is set to /users/owright then the path name of the audit file resulting from the STORE is command is /users/owright/STORE.EDMAUDIT.

Tape Device Names

If the system defaults are not useful, you can use the EDM.DEFAULTS file to specify device names for the Vault logical tape units TAPE1, TAPE2, TAPE3, and TAPE4. For example,

```
TAPE1=/dev/rmt/5
```

Printers

Use the EDM.DEFAULTS file to specify the printer used by the Interactive Query Facility (IQF). The PRINTER statement has the following format.

```
PRINTER=command_for_printer
```

The command_for_printer value is an operating system-specific command, identical to a command you would enter at your OS prompt. For example, on a Solaris machine you could direct IQF output to a printer called lpeddie with the following statement in the EDM.DEFAULTS file.

```
PRINTER=lp -d lpeddie
```

There can be only one PRINTER statement in an EDM.DEFAULTS file. If a PRINTER statement does not exist, IQF sends printed output to the system's default printer.

OBJECT-CLASS Parameter

The OBJECT-CLASS parameter maps the part-type in the database to the rulebase with which the client software processes the part-type. The following part-types must be mapped: CADDS and LOCAL. The following lines show the mappings:

```
OBJECT-CLASS(CADDS)=CVOBJ
OBJECT-CLASS(LOCAL)=LOCAL
```

Environment Variables

If you can set in the `EDM.DEFAULTS` file, you can also set it with the operating system command that sets environment variables. To do this, prefix the EDM variable with `EDM_` and execute the OS command as you normally would. Vault treats any variables defined this way as if they were defined in the `EDM.DEFAULTS` file. For example, on a UNIX system you could enter:

```
setenv EDM_EDMTEMPDIR /user/snoopy/tempfiles
```

Example

The `EDM.DEFAULTS` file might look like the one below if a site-defined rulebase named JET existed.

```
VAULTID=local
ENV(LOCAL)=$EDM_HOME/bin/ddlclrb
ENV(CADDS)=$EDM_HOME/scripts/cvobj.csh
ENV(ADRAW)=$EDM_HOME/bin/cadds5if
SELSCOPE(CADDS)=F,P
SELSCOPE(LOCAL)=F,D
SELSCOPE(PS)=P
SELSCOPE(ADRAW)=P
OBJECT-CLASS(CADDS)=CADDS
OBJECT-CLASS(LOCAL)=LOCAL
OBJECT-CLASS(ADRAW)=CADDS
SELSCOPE=F
ENV=LOCAL
EDMCASE=MIXED
OBJECT_CLASS(JET)=JET
EDMTEMPDIR=/users/snoopy/tempfiles
TAPE2=/dev/rmt/1
ENV(JET)=$EDM_HOME/bin/ddjetrb
SELSCOPE(JET)=F,P
```

Multiple Copies of EDM.DEFAULTS File

Multiple copies of the EDM.DEFAULTS file can exist to provide two levels of variables—site-wide variables and individualized user variables.

When a Vault command is issued, the Vault first checks the EDM.DEFAULTS file in the $EDM_HOME/data directory. Vault then checks your current working directory. If there is no defaults file in your current working directory, the Vault checks your home directory.

For variables that are common to the EDM.DEFAULTS files in the $EDM_HOME/data directory and in your directory, Vault uses the values defined in your directory. Vault uses all other variables that are defined in either defaults file.

Use the EDM.DEFAULTS file in the $EDM_HOME/data directory to specify values for variables that you want your entire site to use. For example, you must define tape logicals in the EDM.DEFAULTS file that is in the $EDM_HOME/data directory. When you install a Vault, the system creates the EDM.DEFAULTS file and includes entries for CADDS and LOCAL. You should not change these entries.

In your current working directory or your home directory, use the EDM.DEFAULTS file to specify values for such things as selection scope, append option, and environment.

Command Parameter Values

The command input parameters ENV and SELSCOPE have default values. You can change one or more defaults in the EDM.DEFAULTS file. When you change the default value of a keyword, the Vault uses the new default with all commands that the keyword applies to.

If the system defaults are not useful for you, set a default by including a line in the EDM.DEFAULTS file in the following format.

```
keyword=value
```

For example, CLASS=PRO would make sense if you always work on project files.

The ENV parameter requires a value, set either explicitly on the command line or in the EDM.DEFAULTS file. In releases prior to Optegra 1.1, the value of ENV was LOCAL. You can retain that default value by placing the following line in the EDM.DEFAULTS file:

```
ENV=LOCAL
```

Any rulebase can be made the default with this syntax:

```
ENV=envvalue
```

The selscope parameter requires a value, set either explicitly on the command line or in the EDM.DEFAULTS file. In releases prior to Optegra 1.1, the value of selscope was F. You can retain that default value by placing the following line in the EDM.DEFAULTS file:

```
SELSCOPE=F
```

Any rulebase can be made the default with this syntax:

```
selscope=Selscope
```

The vaultid parameter specifies the vault which commands refer to. Its value should remain Local unless Distributed Vault is installed.

Planning User-defined Defaults for Command Parameters

The following worksheet lists the defaults you can change, along with their associated prompts, keywords, and possible values. Commands, the default applies to, are listed in the section “Keyword Defaults and Commands” on page 2-30. Make as many copies of the worksheet as you need. Enter the location of the defaults file, and User IDs or project IDs, if this information is pertinent.

Planning User-defined Command Parameter Defaults

Keyword	Alternate Values	Standard Default	New Default
output	F (File), B (Both Screen and File), M (Message only)	S (Screen)	_____
class	PRO (Project) PRI (Private)	PUB (Public)	_____
datecrit	B (Before date) S (Since date)	N (none)	_____
admin	Y (Yes)	N (No)	_____
env	CADDS	LOCAL	_____
dirname	n/a	LISTDIR. LISTING	_____
dirform	L (Long)	S (Short)	_____
loadtype	S (Store) N (Replace if newer)	R (Replace)	_____
ldirname	n/a	platform dependent	_____
clear	Y (Yes)	N (No)	_____
cycles	00-99	1	_____
memtype	P (part) S (file set) U (user-defined)	F (File)	_____
protgrp	0-FFFF	0000	_____
reada	0-99	00	_____
refcnt	0-99	0	_____
append	A (Append)	R (Replace)	_____
selscope	P (part) S (file set) A (all of a project)	F (File)	_____
signout	N (No)	Y (Yes)	_____
tapeunit	TAPE2 TAPE3 TAPE4	TAPE1	_____
termcnt	0-999	0	_____
unldtype	G (Get)	R (Read)	_____
ulmtype	A (All), O (One)	U (user)	_____
writea	0-99	00	_____

Keyword Defaults and Commands

Keyword	Commands
admin	addu, addup
append	archive, chgfa, chgfcl, chgfpw, chgfsc, copy, delete, get, listdir, load, marka, markd, markr, purge, read, readmsg, recsp, replace, reqrvw, reset, restore, scantape, signout, store, unload, unmark, update
class	load, reserve, store
clear	ubkup
cycles	ubkup
datecrit	get, load, read, scantape, signout, unload
dirform	listdir
dirname	listdir
env	store
ldirname	get, read
loadtype	load
memtype	addmfs, remmfs
output	archive, chgfa, chgcl, chgfpw, chgfsc, copy, delete, get,load, listdir, marka, markd, markr, purge, read, replace, reqrvw, reset, scantape, signout, store, unload,unmark, update
protgrp	addu
reada	addu, addup
refcnt	addusa
selscope	chgfa, chgfcl, chgfpw, chgfrev, chgfsc,copy, get, listdir, load, marka, markd, markr, purge, read, replace, reqrvw, reset, rsvp, scantape,signout, store, unload, unmark,update
signout	reserve
tapeunit	archive, ibkup, load, recsp, restore, scantape, ubkup, unload
termcnt	addusa
unldtype	unload
ulmtype	addmul, remmul
writea	addu, addup

Parameters for Which You Can Set Defaults in EDM.DEFAULT

Command	Parameters
addt	class admin protgrp reada writea
addusa	rejcnt termcnt
addmul	ulmtype
addmfs	memtype
archive	append output tapeunit tapenum tappend
addup	admin reada writea
chgfa	append output selscope
chgfcl	append output selscope
chgfpw	append output selscope
chgfrev	selscope
chgfsc	append output selscope
copy	append output selscope
delete	append output
get	append env ldirname object-class output selscope
ibkup	tapeunit tapenum tappend
listdir	append dirform dirname output selscope
load	append class datecrit loadtype output selscope tapeunit
marka	append output selscope
markd	append output selscope
markr	append output selscope
purge	append output selscope
read	append datecrit env ldirname object-class output selscope
replace	append env object-class output selscope
restore	append output tapeunit
readmsg	append
remmul	ulmtype
remmfs	memtype
recsf	tapeunit
recsp	tapeunit
reset	append output selscope
reserve	class signout

Command	Parameters
`rsvp`	selscope
`reqrvw`	append output selscope
`scantape`	append datecrit output selscope tapeunit
`signon`	dmserver
`signout`	append datecrit output selscope
`store`	append class env output selscope
`unmark`	append output selscope
`unload`	append datecrit output selscope tapeunit unldtype
`update`	append env object-class output selscope

Sequence of Steps for Setting Up Vault

After Vault software has been installed and you have made the necessary decisions about how to use Vault at your site, perform these tasks to set up Vault:

1. Establish authority scheme
   - Define a set of status levels for public files.
   - Create command lists for users.
   - Optionally, create authority groups.
2. Add users to Vault.
3. Add projects to Vault.
   - Optionally, create different command lists for project users.
   - Optionally, create different authority groups for project users.
4. Add a project administrator (or yourself) to each project.
   - Project administrators define status levels for projects.
   - Project administrators assign users to their projects.
   - Establish user lists.
5. Associate user lists with status codes for review purposes.
6. Establish user-defined attributes.
7. Load the files to be stored under Vault.

Establish Authority Scheme

We recommend that you establish your Vault authority scheme (by adding status levels and assigning authority numbers to users) before storing files in the database. The authority scheme determines who can access particular files. Use the `ADDS` (add status level) command described in "Creating a Status Level" on page 4-7 and `addu` (add user) command described in the sections "Parameters for Adding Users to Vault" on page 3-5 and "Adding Users to Vault" on page 3-8 to establish your authority scheme.

Each user must have a list of commands he or she is authorized to use for public files. Create this command list with the `addcl` (add command list) command described in Chapter 4, "Creating and Modifying Authority Scheme Components".

You must add a user to Vault with the `addu` command before that user can execute Vault commands. Once a user is defined within Vault, that user can begin executing Vault commands. However, until projects are defined and the user is added to a project(s), the user cannot use Vault in terms of projects.

Set Up Projects

If your site organizes work by projects, you must add each project to Vault with the `addp` (add a project) command (as described in Chapter 5, "Setting Up and Modifying a Project"). You must specify a project command list for each project user. This list can be the same or different from the public command list.

The next step is to add a user to each project with project administrator authority. You can add yourself if you do not want to designate anyone on the project as project administrator.

Each project can have a different authority scheme. The project administrator should define the status levels for a project before assigning the users to that project.

After users and projects have been established, the project administrator can add users to projects with the `addup` command. This person must have the `addup` command in his or her project command list.

Define File Attributes

To associate additional attributes, which you define, with your parts and files, you can use the Attribute Administration commands. User-defined attributes allow you to maintain an association between important information about your parts and files and the parts and files themselves. You can implement attributes at any time, but you should do so only with careful planning. Chapter 5, "Setting Up and Modifying a Project" provides instructions for establishing user-defined attributes.

Load Your Data

The final preparation step is to transfer files into Vault using the `store` or `load` (for CADDS data) commands.

Modifying User and Authority Information

Vault uses the commands described in Chapters 2, 3, and 4 to change user and authority information after your initial Vault scheme is established.

Commands that begin with `chg` (change) modify previously established information. For example, `chgp` (change project information) and `CHGUP` (change information about a user on a project).

Commands that begin with `rem` (remove) cancel a previously established association. For example, `remmul` (remove a user from a user list). Note that this command does not delete either the user or the user list.

Commands that begin with `del` delete previously established information. For example, `delp` (delete a project) and `delul` (delete a user list).

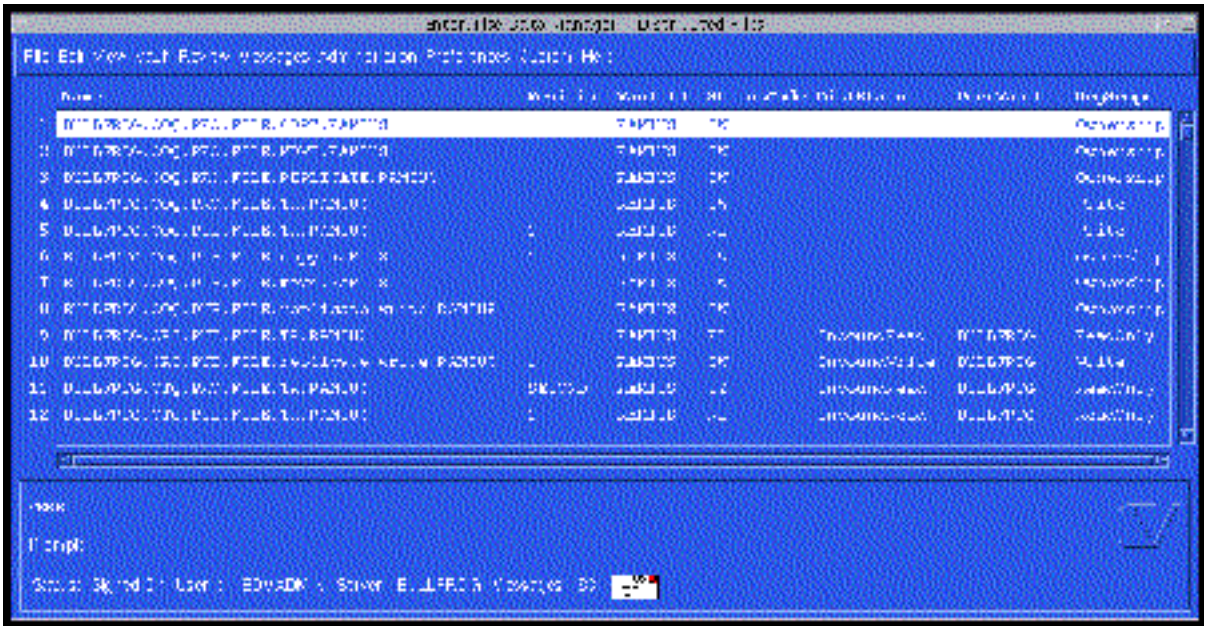
Using the Distributed Vault

The Distributed Vault allows access to objects across multiple servers (Vaults) connected in a distributed, cooperative arrangement. This transparency allows the user to access registered objects from any connected Vault without regard to location, as long as the user has adequate privileges. Just as an object must be stored in the standalone Vault before you can act on it, an object must first be registered for transparent access in the distributed environment.

Vault transparency helps the user locate and access the most suitable version of an object. The object's suitability depends on the user's authority on the Vault in which it is stored and the access defined when the object was registered.

The Distributed Vault browser only displays registered objects if the user has access to the owning Vault. The user must also know the object's name and type and satisfy the same conditions as a local user to access it. For example, the user may see an object in the browser with a Registration Scope indicating higher accessibility than privileges allow. In this case, the user can access the object at the assigned authority level, rather than the object's registration level.

Figure 2-2 The Enterprise Data Manager - Distributed Files Browser



Registering an Object

When you register an object, you make it available in the distributed view (that is, beyond the scope of the Vault which owns the object). You can only register an object that currently exists in the Vault.

At registration, you indicate the object's degree of availability:

- If an object is registered for READ-ONLY access, any user with adequate privileges can read it through the distributed view.
- If an object is registered for READ/WRITE access, any Vault user with adequate privilege can access it through the distributed view and perform actions which require write access, such as locking or updating the object.

Please note: An object registered with READ/WRITE access allows this privilege to all local Vault users.

If an object is not registered, only a user signed on to the Vault in which the object is stored can access the object.

To register an object in the distributed view, type the following on the command line:

```
ciregister selname=FILE1 selscope=f reglevel=R
```

where FILE1 indicates name of the object you are registering, `f` designates the object as a file, and `R` indicates READ-ONLY access to the object in the Distributed Vault.

You can use the `reglevel` of the `ciregister` command to indicate the object's accessibility. The Accessibility determines the object's access by other users. There are four possible registration levels, listed in the following table.

Registration Level	GUI Term	Definition
L	LOCAL	Removes the object from the Distributed Vault, making it available only to the local Vault.
R	READ	Allows READ-ONLY access.
W	WRITE	Allows READ/WRITE access.
O	OWNERSHIP	Allows WRITE access to the object, as well as export-move privileges. Export-move privileges permit an authorized user to move an object from one Vault to another in a distributed environment.

Access to the object (in this example, `FILE1`) depends on the user's access to the Vault on which the object is stored, as well as the registration level of the object.

In the example, a user with READ/WRITE access to objects in the Vault in which `FILE1` is stored has the same access to `FILE1` from the Distributed View. All other users are restricted to READ-ONLY access to `FILE1` in the Distributed View.

You can also register an object using the Graphical User Interface:

1. From the Vault > Files or Vault > Parts view, choose the object you want to register.
2. From the menu bar, choose Administration > Register. The Register dialog box appears.
3. Indicate the Accessibility you want to associate with the object: LOCAL, READ, WRITE, or OWNERSHIP.
4. Click OK.

A user in the Distributed Vault can then access `FILE1` transparently (without indicating the name of the Vault in which it is stored) by following this path from the menu bar:

View > Distributed > Files

Distributing an Object

Distributing an object copies it from the storing Vault to a target Vault using the Vault's Export/Import functionality. With proper permissions, you can distribute a file or part either by requesting immediate distribution or by indicating an event that automatically triggers distribution at a future time.

Before you can designate an object for distribution, you must sign on to the Vault on which it is stored. The object must be registered so you can then select it from the Distributed Files or Parts browser for distribution.

There are six distribution types. You can distribute an object upon request in any of these manners:

- Read Replica
- Write Replica
- Copy
- Move

- Update Original of a Write Replica
- Return Original of either a Read or Write Replica

However, you can only distribute an object when it is triggered by an associated event in one of two ways:

- Read Replica
- Update Original of a write replica

When you choose Distribute Now from the menu bar and indicate Replicate_Read, the Distribute Now dialog box appears.

When you provide all the necessary data, the Graphical User Interface issues an Export command to perform the export-side of this distribution. When control returns to the Graphical User Interface, the export has been accomplished.

As far as the user is concerned, the item is distributed when control returns and the object appears in the browser for Distributed Files or Parts.

The receiving Vaults must import the item, register the replica (noting the Vault from which the object originated), and update the object directory with distribution information.

Read Replica

A Read Replica is a READ-ONLY copy of a file or part distributed by the storing Vault to one or more target Vaults.

When the distribution concludes, the Distributed Vault browser displays an entry for the original object and an entry for of the target Vaults. The `reglevel` of the original remains either `R`, `W`, or `O`, but the `reglevel` of any replica is `R`.

This means that the replica is READ-ONLY, but the original remains in the state it was before the distribution.

Write Replica

A Write Replica is a READ/WRITE copy of a file or part created for distribution by the storing Vault to a single target Vault. The `reglevel` of the original object must be `W` or `O`.

When the distribution concludes, the Distributed Vault browser displays two entries for the object, each with a `reglevel`of `W`. Only the Write Replica can be checked out.

Copy

A Copy is a single distribution of a file or part to one or more target Vaults. The original object in the storing Vault must have a `reglevel` of `R`, `W`, or `O`.

When the distribution concludes, the target Vault receives a copy of the object but does not register it. Therefore, when the distribution concludes, the relationship between the original and the copy no longer exists. Since the copy is not registered, it does not appear in the distributed view browser.

Move

A Move is a single distribution of a duplicate of a file or part to a single target Vault. The original object in the storing Vault must have a `reglevel` of `O`.

When the distribution concludes, the target Vault receives the object, and the storing Vault unregisters it and marks it for deletion.

Update Original

An Update Original distribution updates the object in the storing Vault (the Vault in which the original object is stored) with the version of the item from the target Vault (the Vault in which the WRITE_REPLICA is stored).

When the distribution concludes, the target Vault exports a copy of the WRITE_REPLICA. The storing Vault accesses this copy and updates the original.

Return Original

A Return Original distribution can apply to either a READ_REPLICA or a WRITE_REPLICA.

For the return of a READ_REPLICA, the target Vault (the Vault in which the replica is stored) unregisters the replica and marks it for deletion during the distribution process. When control returns to the Graphical User Interface, the object no longer appears in the distributed view.

For the return of a WRITE_REPLICA, the target Vault (the Vault in which the replica is stored) unregisters the replica and marks it for deletion when the storing

Vault actually imports the item. When the object becomes unregistered then it disappears from the distributed view.

A Return Original distribution updates the object in the storing Vault (the Vault in which the original object is stored) with the version of the item from the target Vault (the Vault in which the replica is stored).

When the distribution concludes, the target Vault exports a copy of the replica. The storing Vault accesses this copy and updates the original. The replica no longer exists in the target Vault at the end of this process.

Distributing on Demand

You can distribute a registered object from the Graphical User Interface on demand with the following procedure:

1. Sign on to the Vault on which the object is stored.
2. Choose the view of the Vault which displays the object you want to distribute (either View > Vault > Distributed Files or View > Vault > Distributed Parts).
3. Choose an object from the browser.
4. From the menu bar, choose Administration > Distribute Now. The Distribute Now dialog box appears.
5. In the Vault List ID box, either type the name of the Vault to which you want to distribute the object, or click the editor button and choose from the options which appear on the Vault Lists browser.
6. Choose the distribution type you want from the Intent options list.
7. Click OK.

Distributing on Event

You can indicate automatic distribution of an object to occur in one of two ways:

- EXPORT_READ_REPLICA
- SYNC_REPLICA_SOURCE

You can designate any of the following events to trigger distribution:

- CHANGE_OBJECT_BINARY—the binary image of the object changes (any time a user updates or replaces the object in the Vault).
- CHANGE_OBJECT_METADATA—anything (except the `READ` command) affects the object.

- CHANGE_OBJECT_REVISION—the object's revision changes, either during the review process or if the user issues a `change_file_revision` command [also available on gui].
- CHANGE_OBJECT_STATUS—the object's status changes as the result of the review process, or as the result of the change-file-status command [also available on gui]

To do this, create a subscription to associate the distribution with the occurrence of an event on an object you indicate:

1. Sign on to the Vault on which the object is stored.
2. Choose the view of the Vault which displays the object you want to affect (either View > Vault > Files or View > Vault > Parts).
3. Choose an object from the browser.
4. From the menu bar, choose Administration > Distribute on Event. The Distribute On Event dialog box appears.
5. From the browser, choose the Subscription that you want to associate with the object you have chosen.
6. Click OK.

Either distribution option can occur multiple times on an object, if multiple events occur to trigger distribution. Each time the Vault distributes an object, it overwrites the previously distributed version.

Creating A Subscription

You can create a subscription to associate with an object which triggers distribution of a selected object when the action you indicate in the subscription occurs:

1. Sign on to the Vault on which the object is stored.
2. Choose the view of the Vault which displays the object you want to affect (either View > Vault > Files or View > Vault > Parts).
3. Choose an object from the browser.
4. From the menu bar, choose Administration > Distribute on Event. The Distribute On Event dialog box appears.
5. Click the Create editor button. The Subscribe for Distribution dialog box appears.
6. Fill out the boxes appropriately.
7. Click OK.

Overview of Database Maintenance

When the initial setup is complete, the following tasks must be performed to keep your system running efficiently.

- Backing up the database (backing up the entire database or creating a backup tape containing new files or files modified since the previous backup)
- Maintaining the size and availability of the database (adding more storage areas, moving currently unneeded files to tape, and deleting unneeded files)

Database Backup

There are two types of database backups that must be performed to ensure database security in the event of media failure or loss.

On a frequent and regular basis (for example, every night):

- Use the Vault incremental backup facility to create a tape version of all new files or files modified since the last incremental backup.
- Use the delete old file versions (`delov`) command to physically delete file versions that have been successfully backed up and are no longer needed.

On an occasional basis, back up the entire database for secure storage at an alternate location. Use the universal backup facility procedure provided with your operating system and compatible with Vault (see the operating system information for Vault).

Please note: To recover a damaged or defective storage area with the `recsp` (recover a storage pool) command, the latest universal backup tape created with the `ubkup` command is required.

See *Vault System Administrator Guide* for information about performing backups.

Database Size and Availability

To maintain the availability of the data in your system, perform these tasks on a regular basis: delete, purge, archive, restore, and recover files. You also need to add storage area as needed.

Clean Out Marked Files:

- Delete files. The `delete` command logically removes files from your system that have been previously marked to be deleted (with the `markd` command) and places them in a queue to be physically deleted during the next execution of the `delov` command.
- Purge files. Occasionally you may want to bypass the normal Vault deletion process and logically delete a file without marking it to be deleted. Use the `PURGE` command to do this.
- Delete Vault process log entries. The Vault process log file records most Vault actions in separate entries. This allows you to track transactions that occur in your Vault system.

After a period of time, the process log file becomes quite large. If this information accumulates indefinitely, it can eventually fill the allocated space in the relational database.

The `dellog` command deletes a specified cycle of Vault process log entries. Perform this operation on an occasional basis, as needed.

Archive Unneeded Files: Archive currently unneeded files. The `archive` command allows you to free up space on your system.

Restore and Recover Data:

- Restore files. Move previously archived files back to Vault.
- Recover a single file. Sometime a file has been logically deleted with the `delete` or `purge` commands. This is a file that can no longer be accessed from Vault but has not yet been physically deleted with the `delov` (delete old file versions) command. After a file has been physically deleted, you can recover it from the backup tape.
- Recover files from a damaged or defective storage pool area. Use the `recsp` command to restore the most current available file versions. (This is a very rare situation.)

Add Data Storage Area As Required:

- Add another data storage area. Use the `addsp` command when the current storage limit of your system is reached.

See *Vault System Administrator Guide* for information about storage pools.

Customizing Storage Pool Selection

Vault provides a default algorithm for selecting the storage pool in which to store a particular file. You can change this algorithm by changing the selection logic that Vault uses. Information about this appears in *Vault System Administrator Guide*.

Introduction to the Vault Process Log

Most Vault transactions are recorded in individual entries in the Vault process log. Log entries show the following information:

- Date and time of the transaction
- User ID of the person who issued the command
- Node the transaction was initiated from
- Operating system ID of the person who issued the command
- Name of the command issued
- Message code or identifier
- File name (when the transaction involves a Vault file)
- File revision (when the transaction involves a Vault file)

Vault can log various types of transactions. You determine which transactions are logged by specifying them in the `nsm.config` file.

Process log information is recorded in an Vault-controlled SQL table called `DM_AUDIT_LOG`. You can read the information in the log by running an Interactive Query Facility (IQF) query.

Transactions That Can Be Logged by Vault

The transactions that Vault records are

- Security violations—Code V

  For example,

```
CDM052V UNSUCCESSFUL SIGNON AFTER 3 ATTEMPTS.
```

- Information messages—Code I

  For example,

```
CDM017I SIGN OFF FROM EDM COMPLETED SUCCESSFULLY.
```

- System errors—Code F

  For example,

```
CMD105F ASMGR ERROR - INVALID LENGTH OF PLACEHOLDER
ARGUMENT.
```

- User errors—Code E

  For example,

```
CDM012E A User ID IS REQUIRED FOR THIS COMMAND. PLEASE ENTER
A User ID IN THE User ID FIELD.
```

- User warnings—Code W

  For example,

```
CMD083W EXIT FROM EDM SCREEN INTERFACE. WARNING! YOU ARE
STILL SIGNED ON TO EDM.
```

Determining Which Transactions Are Logged

The EDMLOG process as defined in the default nsm.config file causes all Vault messages, except for command triggering messages (Code T), to be written to the process log. It is the transaction severity codes specified in the definition of the Application Entity for the Log that determine which messages are logged. These severity codes appear next to the USER keyword, as shown below:

```
!  Application Entity for EDM Logging.
    AE(PDMLOG,pdmgrp,1)
    .  USER(LOG=IEWFV)
         OWNER(edm)
         PATH($EDM_HOME/bin/LOG.STARTUP)
         WORKDIR(/tmp)
         CLOSE
         SERVER
           CONCURRENCY(25,25)
         GRPCTL(1,1,2)
```

You decide which types of messages are logged. Edit the USER line in the nsm.config file if you wish to add a 'T' to the list of severity codes (iewfv) or if you wish to delete severity codes from the list.

If you wish to log all messages, you can edit the line to read (LOG=ALL). This is equivalent to (LOG=IEWFVT).

If you wish to log no messages, edit the line to read (LOG=Z) or (LOG=NONE).

Changing Your Password

If you are setting up Vault for the first time, you may be using a default Vault User ID and password that allows you to access Vault during the initial setup. Do not continue to use the default ID and password. If you have not done so already, change the password for this account to protect your system's security.

Using the Vault Administrator ID

There should be at least one Vault administrator ID within your Vault system. This ID has a public read/write authority level of 99 along with administrative privileges, which allows you to have system administrator authority for every command in your command list.

As Vault administrator, you should have two IDs—a Vault administrator ID and a general User ID. Use the general User ID when you do not need the authority of the administrator ID. This can prevent unintentional overrides of Vault authority checking.

Changing Your Vault User Password

To change your Vault user password, use the `chgupw` command. You must enter your current password. The current password is not shown as it is entered. You must enter a new password; you cannot leave the new Vault user password field blank. The new Vault user password is displayed until you press the processing key. Check for typing errors. Your new Vault user password will be in effect the next time you sign on to Vault.

Chapter 3

Giving Users Access to Vault

This chapter provides information and instructions for granting users the authority to use Vault. After you add a user to Vault, you can assign that user to a project. See Chapter 5, “Setting Up and Modifying a Project” for information about assigning users to projects.

- Overview of Giving Users Access to Vault
- Parameters for Adding Users to Vault
- Adding Users to Vault
- Changing User Attributes
- Parameters for Changing User Attributes
- Changing the Attributes of a User
- Deleting Users from Vault

Overview of Giving Users Access to Vault

To allow someone to use Vault, you must add that person to Vault with the `addu` command. When you execute the `addu` command, you assign to a user a unique

- Vault user ID
- Vault user password

In addition, each user requires a command list name and file access authority numbers or groups. The command list (previously added to Vault) lists the commands the user is authorized to execute. Assigning file access authority is discussed in the next section.

The `CHGU` and `DELU` commands allow you to change a user's attributes and delete a user.

Vault User IDs

Each Vault user ID must be unique and can include up to twelve letters and/or numbers. Vault users should not share common IDs as this will override Vault security and concurrence control.

Vault User Passwords

To protect their security, passwords should be unique for each user. When you add a user to Vault with `addu`, the password is displayed on the screen until you press the processing key. This allows you to verify your entry. During `addu` command execution, the password is encrypted and this encrypted value is stored in the database.

At our example site, the High Flying Airplane Company, each time a new user is added to Vault, the administrator assigns the same Vault user password as the Vault user ID. To protect their system security, Vault users are responsible for using the `chgupw` (change user password) command to immediately change their passwords.

Command List Names

All Vault users require a command list that specifies which Vault commands they can execute. The command list must be created before the user is added. When you execute the `addu` command, you assign a command list to the user.

Please note: When the `addu` command is in your command list, you can add a user with any level of authority, including authority that is greater than your

own. This means that when you assign a user relatively low authority levels but include the `addu` command in the assigned command list, the user can create another user ID with the highest authority levels and privileges.

Optional Identification Information

When executing the `addu` command, you can enter additional information as listed below. Vault stores these values but does not do anything with them. You can enter any information that is pertinent at your site.

- The user's full name (up to 25 alphanumeric characters, including blank spaces; for example, `WILBUR WRIGHT`).
- The user's last name (up to 15 alphanumeric characters).
- User description information (up to 30 alphanumeric characters, including blank spaces).
- User group code (one or two alphanumeric characters).

Common Login

The first time a user signs on to the Vault, the system prompts for identifying information:

- User ID
- Password
- Vault name

If Common Login is enabled (the default), the system stores this information in a common login repository in the user's home directory. This common login repository is called `$HOME/.optegra/optegra.ini`. The `.optegra` directory allows READ/WRITE access to `User` only, and no access for `Group` or `Other`.

The system creates this common login repository the first time the user invokes a Vault service. Each subsequent time the user signs on to the Vault with Common Login enabled, the system accesses this stored User ID and Password instead of prompting for it. If any ambiguity exists concerning which Vault to access, the system selects the first Vault in the user's local `pm.config` file.

The environment variable `$ANSPATH` contains the address of the Vault configuration file, `pm.config`. This file contains a list of Vaults in the format:

```
RESOURCE(VAULTID:::host:process_manager_domain
            :process_manager_AE:0,.....)
```

where VAULTID represents the name of a Vault which the user can access.

Reordering these entries in the `pm.config` file changes the default Vault.

Please note: If the user later signs on with a different User ID, the Common Login repository is updated to reflect these new values.

Enabling Common Login

Common Login is enabled by default. You can disable it with one of two options:

- For a single session
- Permanently

This is detailed in the *Vault End User Guide*.

Sequence of Steps for Adding Users

When you add a user to Vault, you assign public authorities to the user. Vault uses these authorities for public and private files. Follow the steps below to give users access to Vault.

1. Define the set of public status levels. Instructions are in Chapter 4, “Creating and Modifying Authority Scheme Components”.
2. Define the command list you plan to assign to the user. Instructions are in Chapter 4, “Creating and Modifying Authority Scheme Components”.
3. If you are using authority groups, define the authority group(s) you plan to assign to the user. Instructions are in Chapter 4, “Creating and Modifying Authority Scheme Components”.
4. Add the user to Vault with `addu` command. Instructions are in this chapter.

See Chapter 5, “Setting Up and Modifying a Project” to assign users to projects.

Parameters for Adding Users to Vault

The next table describes `addu` command parameters. Additional information about some parameters appears below.

Table 3-1 `addu` **Command Parameters**

Keyword	Parameter Description	Default
userid	Unique Vault user identifier. Enter 12 or fewer letters and/or numbers.	None
userpw	User password. Enter 8 or fewer letters and/or numbers.	None
fullname	User full name. Enter 25 or fewer alphanumeric characters, including blanks. Optional.	None
lastname	User last name. Enter 15 or fewer alphanumeric characters. Optional.	None
userdesc	User description field. Enter 30 or fewer alphanumeric characters, including blanks. Optional.	None
cmdlist	Existing command list name. Enter 8 or fewer letters and/or numbers.	None
usergrp	User group code. Enter 1 or 2 alphanumeric characters. Optional.	None
reada	Read authority number that sets a user's read authority limit. Enter a number between 0 and 99 *unless you enter a read authority group name.*	00
readag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a read authority number.*	None
writea	Write authority number that sets a user's write authority limit. Enter a number between 0 and 99 *unless you enter a write authority group name.*	00
writeag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a write authority number.*	None
admin	Indicates whether the user is given special administrative privileges. Enter: `Y` (yes) or `N` (no).	N
lanqpref	Specifies a language preference.	
areada	Associates a Read Authority Number. Enter a number between 0 and 99. A number entered here means a range from 0-<number>.	0
areadag	Associates a Read Authority Group Name. Enter a authority group name up to 8 alphanumeric characters unless you enter a read authority number.	
awritea	Associates a Write Authority Number. Enter a number between 0 and 99.A number entered here means a range from 0-<number>.	0

Table 3-1 `addu` **Command Parameters**

Keyword	Parameter Description	Default
awriteag	Associates a Write Authority Group Name. Enter authority group name up to 8 alphanumeric characters unless you enter a write attribute authority number. If both are entered the authority number is ignored.	
uompref	Specifies a preferred unit of measure system. Enter your preferred unit of measure system, for example, MKS/FPS	

File Access Information

To establish the file access authority for each user, enter information for the following parameters:

- Read authority number or read authority group
- Write authority number or write authority group
- Administrative privileges

Enter information for either the read authority number (the default value is 00) or read authority group. You must create an authority group with the `addag` command before you can assign it to a user.

Enter information for either the write authority number (the default value is 00) or write authority group. Again, you must create an authority group before you can assign it to a user.

Please note: If both authority numbers and authority groups are assigned to a user, Vault uses the authority groups and ignores the authority numbers. For example, if a user is assigned a read authority number of 50 and a read authority group that contains the number 60, the user can access only those files corresponding to 60.

Administrative Privileges

Enter a `Y` (yes) or `N` (no) for the administrative privileges parameter to indicate whether or not the user has administrative privileges in the public authority scheme. Users with administrative privileges

- Do not need to enter file passwords for those files they have the authority to access
- Can reset, update, and replace files signed out to another user (execute the `reset`, `update`, and `replace` commands for files not signed out to them)
- Can change the status of a file (`chgfsc` command) without proceeding in a sequential order through the status codes.

These privileges apply to files they have the authority to access.

Project Authority

Users do not have authority on any projects unless they are also added to each project with the `addup` command.

Adding Users to Vault

Use the `addu` command to assign a Vault user ID.

In the command line interface example which follows, a new Vault user, `Wilbur Wright`, is added to Vault. He signs on to Vault with the user ID of `WILBUR`. His password is shown on the screen as it is entered. He will be able to use the commands in the `genuser` command list with a read authority associated with the `GENERALR` authority group name and a write authority associated with the `GENERALW` authority group name. He is not assigned administrative privileges.

```
ciaddu
EDM> userid=Orville userpw=Orpw,
EDM> cmdlist=Genuser readag=Group1r,
EDM> writeag=Group1w
```

In this example, a new user Orville is added to Vault. His password (Orpw) is shown on the screen. He will be able to use the commands in the `genuser` command list with the read authority associated with the GROUP1R authority group name and the write authority associated with the GROUP1W authority group name. He is not assigned administrative privileges.

```
ciaddu
EDM> userid=Orville userpw=Orpw cmdlist=Genuser,
EDM> readag=Group1r writeag=Group1w,
EDM> fullname='Orville Wright' lastname=Wright,
EDM> userdesc='05/11/99' usergrp=AA protgrp=0000,
EDM> cmdlist=Genuser
```

In this example, a new user ORVILLE is added to Vault. Optional parameters define additional data about this user.

```
ciaddu
EDM> userid=Orville userpw=Orpw,
EDM> fullname='Orville Wright' lastname=Wright,
EDM> userdesc='Added to EDM 5/11/99' usergrp=Aa,
EDM> protgrp=0000 cmdlist=GENUSER reada=50,
EDM> readag=Group1r writeag=Group1w
```

In this example, a new user Orville is added to Vault. His password (Orpw) is shown on the screen. He will be able to use the commands in the genuser command list with the read authority associated with the Group1r authority group name and the write authority associated with the Group1w authority group name. He is not assigned administrative privileges.

Please note: Even though a read authority number (`reada=50`) is included in the command line, it will be ignored because a read authority group name is also specified.

Changing User Attributes

To change the attributes of a previously defined user, use the `chgu` command. With this command you can change any user attribute except the Vault user ID. To change a user ID, you must delete the ID and add the changed ID as a new user with the `addu` command.

If both single authorities and authority groups are assigned to a user, the authority groups override the single authorities. For example, if a user is assigned both a read authority number of 50 and a read authority group that contains authority number 60, the user can access only those files corresponding to 60.

Administrative Privileges

Enter a `Y` (yes) or `N` (no) for the administrative privileges parameter to indicate whether or not the user has administrative privileges in the public authority scheme. Users with administrative privileges

- Do not need to enter file passwords for those files they have the authority to access
- Can reset, update, and replace files signed out to another user (execute the `reset`, `update`, and `replace` commands for files not signed out to them)
- Can change the status of a file (`chgfsc` command) without proceeding in a sequential order through the status codes.

These privileges apply to files they have the authority to access.

Entering Data with the Command-line Format

If you are using the command-line format, enter changed or new data as follows:

- Enter a cross hatch (#) as the first character following the equals sign (=) to delete previous data and not replace it.

  For example, to delete the current user group code and not replace it, the keyword and value should look like this

  ```
  usergrp=#
  ```

- Enter the new or changed data following the equals sign (=). For example

  ```
  usergrp=33
  ```

Please note: If information for a parameter has not changed, you do not need to enter either the keyword associated with the parameter or the unchanged data.

Parameters for Changing User Attributes

The next table describes the `chgu` command parameters. Additional information appears below.

Vault User Passwords

To protect their security, passwords should be unique for each user. When you change a password with `chgu`, the password is displayed on the screen to allow you to verify your entry. During `chgu` command execution, the password is encrypted in the database.

Once a user is added to Vault, a password is not shown when it is entered in response to a prompt when using the full-screen format. This password is shown, however, when it is entered using the command-line format.

File Access Information

To change the file access authority for each user, enter information in the following parameter fields

- Read authority number or read authority group
- Write authority number or write authority group
- Administrative privileges

Enter information in either the read authority number or read authority group parameter field. If you enter an authority group name, add the name to Vault with the `addag` command before you specify it with `chgu`.

Enter information in either the write authority number or write authority group parameter field. If you enter an authority group name, add the name to Vault with the `addag` command before you specify it with `chgu`.

Command List Names

All Vault users require a command list that specifies which Vault commands they can execute. The command list must be created before it is specified with `chgu`.

Table 3-2 `chgu` **Command Parameters**

Keyword	Parameter Description	Default
userid	Existing Vault user identifier. Enter 12 or fewer letters and/or numbers.	None
userpw	User password. Enter 8 or fewer letters and/or numbers.	None
fullname	User full name. Enter 25 or fewer alphanumeric characters, including blanks. Optional.	None
lastname	User last name. Enter 15 or fewer alphanumeric characters. Optional.	None
userdesc	User description field. Enter 30 or fewer alphanumeric characters, including blanks. Optional.	None
cmdlist	Existing command list name. Enter 8 or fewer letters and/or numbers.	None
usergrp	User group code. Enter 1 or 2 alphanumeric characters. Optional.	None
reada	Read authority number that sets a user's read authority limit. Enter a number between 0 and 99 *unless you enter a read authority group name.*	None
readag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a read authority number.*	None
writea	Write authority number that sets a user's write authority limit. Enter a number between 0 and 99 *unless you enter a write authority group name.*	None
writeag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a write authority number.*	None
admin	Indicates whether the user is given special administrative privileges. Enter: `Y` (yes) or `N` (no).	None
areada	Associates a Read Authority Number. Enter a number between 0 and 99. A number entered here means a range from 0-<number>.	0
areadag	Associates a Read Authority Group Name. Enter a authority group name up to 8 alphanumeric characters unless you enter a read authority number.	
awritea	Associates a Write Authority Number. Enter a number between 0 and 99.A number entered here means a range from 0-<number>.	0

Changing the Attributes of a User

Use the `chgu` command to modify the attributes of an Vault user.

```
cichgu
EDM> userid=Orville usergrp=Bb
```

In this example, Vault user Orville has his user group code changed to BB.

```
cichgu
EDM> userid=Orville usergrp=#
```

In this example, Vault user Orville has his user group code removed (with no replacement).

Deleting Users from Vault

Use the `delu` command to cancel a user's authority to use Vault.

After the command is executed, no one can sign on to Vault using this user ID. A Vault user ID cannot be deleted if this ID

- Has any file signed out
- Has any user-defined tables opened
- Owns any private files
- Is on any user list
- Is a reviewer in any current review
- Has any review messages

Table 3-3 `delu` **Command Parameter**

Keyword	Parameter Description	Default
userid	Existing Vault user identifier. Enter 12 or fewer letters and/or numbers.	None

```
cidelu
EDM> userid=Orville
```

In this example, Vault user Orville is deleted.

Chapter 4

Creating and Modifying Authority Scheme Components

This chapter provides instructions for creating and modifying status levels, authority groups, and command lists. Vault does not include commands for handling status levels or authority groups, nor does it include the `admcopy` command.

- Overview of Authority Scheme Components
- Introduction to Status Levels
- Creating a Status Level
- Changing a Status Level
- Deleting Status Levels
- Introduction to Authority Groups
- Creating an Authority Group
- Changing an Authority Group
- Deleting an Authority Group
- Introduction to Command Lists
- Creating a Command List
- Changing a Command List
- Deleting a Command List
- Using Administrative Information

Overview of Authority Scheme Components

Your implementation of Vault can include:

- One public authority scheme and no project authority schemes
- One public authority scheme and one or more project authority schemes
- No public authority scheme and one or more project authority schemes

However you set up Vault, an authority scheme includes the same basic components.

Please note: If you have not done so already, see the *Vault System Administrator Guide* for complete information about authority schemes.

What Is an Authority Scheme?

An authority scheme determines what commands a user can perform and whether a user can access a particular file at a particular stage of work. Both public and project authority schemes includes these components:

- A series of status levels
- Authority numbers and/or authority groups assigned to users
- Command lists assigned to users

Public Authority Scheme

The public authority scheme consists of

- A series of status levels that have a blank project ID
- The authority numbers and authority groups you assign to users when you grant them access to Vault (add them to Vault with the `addu` command)
- The command lists you assign to users when you add them to Vault

Vault uses the public authority scheme to determine whether or not to allow access to public and private files. Public files are not owned by any project or user. Private files belong to the User ID that created them.

Project Authority Scheme

A project authority scheme consists of

- A series of status levels associated with the same project ID
- The authority numbers and authority groups you assign to users when you assign them to the project
- The command lists you assign to users when you add them to the project

Each project has its own authority scheme. The components that make up the authority scheme can be used by multiple projects and also by the public authority scheme. For example, you can copy a set of status levels to one or more projects. You can assign the same command lists and authority groups to users assigned to different projects.

Authority Groups and Command Lists

When you define an authority group or command list, you do not specify whether it is used with public and private files or with project files. It is only when you assign the authority group or command list to a user that you determine the classification of data it is used for.

When you add users to Vault, the command list and authority groups you assign become the ones Vault uses for public and private files.

When you add users to projects, the command list and authority groups you assign become the ones Vault uses for that project.

Sequence of Steps for Defining Authority Scheme Components

Follow the steps below to define authority scheme components.

1. Define a set of status levels.
2. If you are using authority groups, define authority groups.
3. Define command lists.

Whether you use a set of status levels for public files or for a project, the process of defining the set of status levels is the same. This chapter provides the instructions for defining a set of status levels. Chapter 5, “Setting Up and Modifying a Project” provides instructions for creating projects and assigning users to projects.

When you create a project, you define a project ID. To associate status levels with a project, first create the project, then follow the instructions in this chapter for creating status levels. Be sure to specify a previously defined project ID when you create the status levels. Chapter 5, “Setting Up and Modifying a Project” describes how to create a project and this chapter describes how to define status levels for the project.

Whether you use a command list or authority group for public or project data, the process of defining the command list or authority group is the same. This chapter provides the instructions for creating the authority groups and command lists that you can assign to users when you add them to Vault or when you assign them to projects.

Introduction to Status Levels

Each status level reflects a stage of work that occurs at your site. Each file moves through this series of status levels until it is complete. To add a status level to a public or project authority scheme, use the `adds` command. The `adds` command has six parameters: status code, project ID, sequence number, authority number, status partition code, and status description.

Default Status Levels

Vault includes two default public status levels. They are IW (initial in-work status level, sequence number 100, authority level 0) and RL (initial released level, sequence number 900, authority level 0). These status levels allow you to immediately store files in Vault. You can change or remove them with the `chgs` and `dels` commands.

Status Codes

Each status level must have a unique (within its authority scheme) alphanumeric status code. Status codes can, however, be shared among the public authority scheme and different projects. (You can only have one public authority scheme.) For example, at our sample site, the High Flying Airplane Company, there is a PR status code in the public authority scheme and in the `CARGO` project authority scheme. There cannot be two PR status codes within a single project authority scheme.

Project IDs

To add the status level to a public authority scheme, leave the Project ID parameter blank (this is the default). To add the status level to a project, specify a project ID. You must have previously defined the project with the `addp` command—see Chapter 5, “Setting Up and Modifying a Project” for instructions. The `ADDS` command must be in your command list for the project receiving the new status level.

Sequence Numbers

Each status level must be assigned a unique sequence number, indicating its position relative to the other status levels in the authority scheme. Assign the lowest sequence number in your scheme to the status level that comes earliest in the development cycle. Incrementing sequence numbers by at least 10 will allow easier insertion of new status levels. For example, add sequence numbers 10, 20, 30, and 40, rather than 1, 2, 3, and 4.

Authority Numbers

The authority number is the bridge that associates the status level of a work phase with an authority (both read and write access) level for users.

Status Partition Codes

Within each authority scheme, designate two status codes with a status partition code. These two codes partition (divide) the scheme into three phases: dormant, in-work, and released. Assign the `I` (In-work) status partition code to one status level. Assign the `R` (Released) code to a different status level.

The `I` status partition code divides the dormant phase and the initial in-work phase. The `R` code divides the in-work phase from the released phase. The sequence number associated with the `I` status partition code must be less than the number associated with the `R` status partition code.

Status Description

This is an optional parameter. Use it to clarify the purpose of the status level.

Creating a Status Level

Execute the `adds` command to add a status level to an authority scheme.

Table 4-1 `adds` **Command Parameters**

Keyword	Parameter Description	Default
statcd	Unique code that represents a phase of work in a Vault authority scheme. Enter 8 or fewer alphanumeric characters.	None
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers or leave blank for the public authority scheme.	Blank
seqnum	Number code that establishes order in this authority scheme. Enter a number between 0 and 9999.	None
authnum	Number code that associates a user's authority with a status code. Enter a number between 0 and 99.	None
statpc	Code to distinguish two special status levels within each authority scheme. These codes partition the scheme into *dormant*, *in-work,* and *released* status levels. Enter: `I` (for the initial in-work status) or `R` (for the initial released status). Leave this parameter blank for all other status levels. Note that within each authority scheme one status level must be assigned the I status partition code and one must be assigned the R status partition code. The status level associated with the R partition code must have a higher sequence number than the one associated with the I partition code.	None
statdesc	Thirty-character field (including blank spaces) to describe the phase of work of this status level (or any other information). Optional.	None

Enter the required information and any optional information.

The following status level can be added to a public authority scheme:

- Has a status code of ID (Initial Design)
- Has a blank project ID making it a public status level
- Has a sequence number of 10
- Has an authority number of 10
- Partitions dormant status levels and in-work status levels

Please note: Adding a status level to an authority scheme that is in use could affect the tracking of any file stored under that authority scheme.

```
ciadds
EDM> statcd=DR seqnum=20 authnum=30,
EDM> STATDESC=DRAFT
```

In this example, a status level is added to the public authority scheme. It has a status code of DR, its sequence number is 20, and its authority number is 30.

```
ciadds
EDM> statcd=Pr seqnum=30 authnum=40,
EDM> statdesc=PreliminaryVersion
```

In this example, the status level could not be added because the status description parameter field contains a blank space, but it is not enclosed in single quotation marks (’). Enclose a description that includes a blank space in single quotes as shown here.

```
ciadds
EDM> statcd=Pr seqnum=30 authnum=40,
EDM> statdesc='Preliminary Version'
```

Please note: To copy an existing authority scheme to a project, use the admcopy command, described at the end of this chapter.

Changing a Status Level

To change attributes of an existing public or project status level, use the `chgs` command. Use this command to change the status level

- Sequence number
- Authority number
- Status partition code
- Status description

The status level you are changing must already exist in the authority scheme.

You cannot change the status code of a status level. Delete the status level and add it again with the new status code.

To change status level information in a public authority scheme, leave the Project ID parameter blank (this is the default). To change project status level information, enter an existing project ID for the Project ID parameter.

Displaying Current Data

If you are using the full-screen format, display the current status level data by executing the `chgs` command and specifying only the status code and project ID parameters.

Changing a Sequence Number

If you are changing the sequence number of the status level, remember that each sequence number within the authority scheme must be unique.

Changing the Status Description

Use the status description parameter to provide comments about the status level. Your comment can be 30 or fewer alphanumeric characters (including blank spaces). If you are using the command-line format to enter this parameter, enclose the field with single quotation marks (') if your comment includes blank spaces.

Initial In-Work and Released Status Levels

To assign a status partition code to a different status level, first change the status level that is currently associated with the status partition code. Enter a pound sign (#) for the Status partition code parameter to delete the status partition code from the status level.

Immediately add or change another status level so that an initial in-work and an initial released status level always exist in the scheme. Vault displays a message when an initial in-work or initial released status partition code is removed, since it should be replaced immediately.

The `I` (for initial in-work) and the `R` (for initial released) status partition codes must be unique within the scheme to which they are being added. A status partition code cannot be changed to `I` or `R` if it already exists in the scheme. You cannot store or access files if the associated authority scheme does not include both the `I` and `R` status partition codes.

The sequence number associated with the `I` status partition code must be less than the sequence number associated with the `R` status partition code.

Entering Data with the Command-Line Format

If you are using the command-line format, enter changed or new data as follows:

- Enter a cross hatch (#) as the first character following the equal sign (=) to delete previous data and not replace it.
- For example, to delete the current status description and not replace it, the keyword and value should look like this

  ```
  statdesc=#
  ```

- Enter the new or changed data following the equals sign (=). For example

  ```
  authnum=30
  ```

Execute the `chgs` command to change a status level in an authority scheme.

Table 4-2 chgs Command Parameters

Keyword	Parameter Description	Default
statcd	Existing code that represents a phase of work in an Vault authority scheme. Enter 8 or fewer alphanumeric characters.	None
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers or leave blank for the public authority scheme.	Blank
seqnum	Number code that establishes order in this authority scheme. Enter a number between 0 and 9999.	None
authnum	Number code that associates a user's authority with a status code. Enter a number between 0 and 99.	None
statpc	Code to distinguish two special status levels within each authority scheme. These codes partition the scheme into *dormant*, *in-work,* and *released* status levels. Enter `I:` (for the initial in-work status) or `R` (for the initial released status). Leave blank for all other status levels. Within an authority scheme, assign I to one status level and R to another. The status level associated with the R must have a higher sequence number than the one associated with the I.	None
statdesc	Thirty-character (including blank spaces) field to describe the phase of work of this status level (or any other information). Optional.	None

Enter the required information and any optional information.

In the example below, the modified status level

- Has a status code of ID and a sequence number of 10.
- Now has an authority number of 20. (It was 10.)
- Divides dormant status levels and in-work status levels.

In the example below, a status level is changed. It has a status code of Dr, its new sequence number is 20, and its new authority number is 30.

```
cichgs
EDM> statcd=Dr SEQNUM=20 authnum=30
```

Deleting Status Levels

To delete an existing status level, use the `dels` command.

To delete a status level within a public authority scheme, leave the `Project ID` parameter blank.

To delete a status level within a project authority scheme, enter the name of an existing project ID for the `Project ID` parameter.

If you delete a status level with the `I` (initial in-work) or `R` (released) status partition code, be sure to add or change another status level immediately so that the initial in-work and released status levels are still defined. You cannot store or access files when the associated authority scheme does not have both the `I` and `R` status partition codes.

Please note: You cannot delete a status level if

- There are any active (not archived) Vault files or user-defined tables currently stored under that authority scheme with that status code.
- The status code is associated with a user list.

Table 4-3 `dels` **Command Parameters**

Keyword	Parameter Description	Default
statcd	Existing code that represents a phase of work in an Vault authority scheme. Enter 8 or fewer alphanumeric characters.	None
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers or leave blank for the public authority scheme.	Blank

```
cidels
EDM> statcd=Dr
```

In this example, the Dr status level is deleted from the public authority scheme.

Introduction to Authority Groups

An authority group is a list of authority numbers associated with a unique authority group name that determines what read or write access authority a user will have in Vault. The lowest authority number is 0 and the highest is 99. Each time you add a user to Vault (with `addu`) or to a project (with `addup`), you must specify a

- Read authority number or a read authority group name
- Write authority number or a write authority group name

For example, to give a user read authority for all files stored at a corresponding authority level of 0 through 50, specify a read authority number of 50.

You can create an authority group with the add authority group (`addag`) command and specify 0-50 as the authority numbers to accomplish the same thing. This authority group name can then be specified for the user as a read authority group name (or a write authority group name if you are specifying the user's write authority privileges).

In this case, the read authority group name provides the same file access as the read authority number (0-50). If, however, you want this person to read files only at authority levels 25-50, add an authority group name and specify 25-50 as the authority numbers associated with that name.

Authority Group Names versus Authority Group Numbers

When you add the user in this example to Vault or to a project, specify the authority group name in the Read authority group name parameter instead of a read authority number. In this way, the user can access files that have corresponding authority numbers of 25-50 and is denied access to files that have corresponding authority numbers of 0-24 and over 50.

In other words, an authority number allows access to any authority levels up to and including the number, 50 indicates 1 through 50. An authority group allows access to any authority level that appears in the group, 50 indicates only 50.

Use the administrative copy (`admcopy`) command to associate the authority numbers in an existing authority group with a new authority group name. `admcopy` is described later in this chapter.

Specifying Authority Numbers

When defining an authority group, specify authority numbers as

- Single numbers, separated by commas
- Ranges of numbers, separated by hyphens (-)

The second number of a range cannot be less than the first. There can be no blank spaces in your list. The maximum number of characters in the list is 300 (including hyphens and commas).

Please note: When using the command-line format, enclose the list of authority numbers with single quotation marks (').

Here is an example of correctly entered authority numbers:

5,10,20-40,50-80,99

Here are some incorrect examples (and why):

5 10 No blank spaces are allowed.

5,10-4 The second number of a range must be greater than the first.

5.6 Commas must separate the numbers.

Vault edits your authority number entries as follows:

- Duplicate entries are ignored.
- Entries not in ascending order are sorted and subsequently displayed in ascending order.
- Consecutive numbers are redisplayed in ranges.
- Individual entries that are part of a range are subsequently included in a range.

For example

80,80 becomes 80

80,70,60 becomes 60,70,80

80,81,82 becomes 80-82

Creating an Authority Group

Use the `addag` command to create an authority group.

Table 4-4 `addag` **Command Parameters**

Keyword	Parameter Description	Default
authgrp	Unique authority group name. Enter 8 or fewer letters and/or numbers.	None
authnums	List of authority numbers (and/or authority ranges). Enter up to 300 characters, including numbers, commas, and hyphens. Separate each individual number from the next by a comma (for example, 5,10,50). Separate each number within a range by a hyphen (for example, 10,30-40,50). The list *cannot* include blank spaces. Valid values are 0 through 99.	None

```
ciaddag
EDM> authgrp=Level1 authnums='0-20,30,35,90'
```

In this example, an authority group named `LEVEL1` is added to Vault. It includes authority levels 00-20, 30, 35, and 90.

Please note: You must start and end the list with a single quotation mark

Changing an Authority Group

To change the authority numbers of an existing authority group, execute the `chgag` command.

Specifying Authority Numbers

Specify authority numbers as

- Single numbers, separated by commas
- Ranges of numbers, separated by hyphens (-)

The second number of a range cannot be less than the first.

There can be no blank spaces in your list.

The maximum number of characters in the list is 300 (including hyphens and commas).

If you are using the command-line format to enter the authority numbers, enclose the list with single quotation marks (’).

Here is an example of some correctly entered authority numbers:

5,10,20-40,50-80,99

Here are examples of incorrectly entered authority numbers (and why):

5, 10, No blank spaces are allowed.

5.6 Commas must separate the numbers.

5,10-4, The second number of a range must be greater than the first.

How Vault Edits Your Authority Number Entries

Vault edits your authority number entries as follows:

- Duplicate entries are ignored.
- Entries not in ascending order are sorted and subsequently displayed in ascending order.

- Consecutive numbers are redisplayed in ranges.
- Individual entries that are part of a range are subsequently included in a range.

For example

80,80 becomes 80

80,70,60 becomes 60,70,80

80,81,82 becomes 80-82

80,80-82 becomes 80-82

Please note: The `chgag` command directly affects what files users can access.

Table 4-5 `chgag` **Command Parameters**

Keyword	Parameter Description	Default
authgrp	Existing authority group name. Enter 8 or fewer letters and/or numbers.	None
authnums	List of authority numbers (and/or authority ranges). Enter up to 300 characters, including numbers, commas, and hyphens. Separate each individual number from the next by a comma (for example, 5,10,50). Separate each number within a range by a hyphen (for example, 10,30-40,50). The list *cannot* include blank spaces. Valid values are 0 through 99.	None

```
cichgag
EDM> authgrp=Level1 authnums='0-20,30,35,99'
```

In this example, the Level1 authority group is changed.

```
cichgag
EDM> authgrp=Level1 authnums=0-20,30,35,99
```

In this example, the authority group could not be changed because the list was not enclosed with single quotation marks (').

Deleting an Authority Group

Use the `delag` command to delete an authority group

You cannot delete an authority group currently assigned to any public or project user.

Table 4-6 `delag` **Command Parameters**

Keyword	Parameter Description	Default
`authgrp`	Existing authority group name. Enter 8 or fewer letters and/or numbers.	None

```
cidelag
EDM> authgrp=Level1
```

In this example, the Level1 authority group name is deleted.

Introduction to Command Lists

A command list is a list of Vault commands. You must assign all Vault users a command list when you add them to Vault. The commands in your assigned command list are the only commands users can execute after they have successfully signed on to Vault.

For example, the `OPERATOR` command list at the High Flying Airplane Company includes the following commands:

```
ibkup
delov
```

If user CHARLES is assigned this public command list, he can only execute these two commands. When he uses menus, `ibkup` and `delov` are the only commands displayed on his Database Maintenance Menu. The Database Maintenance Menu is the only one listed on his Main Menu (in addition to the `signon` and `signoff` commands).

Please note: There are two Vault commands automatically assigned to users – `signon and signoff.`

Using the ALL Keyword for Easy Setup

There is a keyword—`allcmds`—that allows you to easily set up a command list with all Vault commands. When new commands are introduced, users with `allcmds` in their command list are automatically able to perform the new commands. Their command list need not be changed. (Vault arrives at your site with one Vault User ID and password associated with a command list containing the `all` keyword.)

Recommendations for Command Lists

To maintain rigorous control over your Vault system, we recommend that you create only one command list that includes the `all` keyword. Assign this list to a limited number of users.

Creating a Command List

Use the `addcl` command to create a command list. This command associates a command list name with a list of Vault commands.

Please note: When you use the addcl command, you must have '`EDMCASE=MIXED`' in the `EDM.DEFAULTS` file.

Using the Command-Line Format

In the example below, the command list name `JETCLA` is added. It will contain all Vault commands.

```
ciaddcl
EDM> cmdlist=jetcla allcmds=Y
```

In the next example, the command list name `jetclg` is added. `jetgen.commands` is a previously created local text file that contains the command names to be associated with this list.

```
ciaddcl
EDM> cmdlist=jetclg allcmds=N,
EDM> cmdfile=jetgen.commands
```

Creating Command Lists with the Command-Line Format

The procedure to create a command list with the command-line format depends on whether all or a subset of commands are to be included in the list. To create a command list with all commands, enter the command list name and specify `Y` (yes) for the `allcmds` keyword. For example

```
ciaddcl
EDM> cmdlist=all allcmds=Y
```

To include only a command subset, create a local text file that contains the command names to be included (one command name per line). Blank spaces in the file are ignored. Other data may be ignored or could cause the command to fail. There should be no blank lines. For example, a file called `COMMANDS.FILE` might contain the following two lines.

To execute the `ciaddcl` command, enter the unique command list name, `N` (no) for the `allcmds` keyword, and the name of the text file containing the commands to be included in the command list. For example

```
ciaddcl
EDM> cmdlist=operator allcmds=N,
EDM> cmdfile=COMMANDS.FILE
```

This command creates the `operator` command list with the `ibkup` and `delov` commands in it.

Because you must create a text file with the commands (unless you specify all commands) when you use the command-line format, use menus to create a list with many commands. This does not apply if the command list contains only a few commands (such as in the example above with only two commands).

Table 4-7 `addcl` **Command Parameters**

Keyword	Parameter Description	Default
cmdlist	Unique command list name. Enter 8 or fewer letters and/or numbers.	None
allcmds	Indicates whether all Vault commands are to be included in the command list. Enter: `Y` (yes) or `N` (no).	None
cmdfile	If the `Include all commands?` parameter is N (no), enter a local text file name, with 80 or fewer alphanumeric characters that contains command names to be included in the list.	None

Notes for Creating a Command List

In most cases, including the command name in a command list allows users assigned that command list to execute the command. Here are the exceptions:

- When a command list includes the `unload` command, it must also include the command that corresponds to the unload option parameter. To use the read unload option the `read` command must be in the command list. To use the get unload option the `get` command must be in the command list.
- When a command list includes the `admcopy` command, it must also include the command(s) that corresponds to the appropriate copy type parameter(s).

Table 4-8 Commands corresponding to the Copy Type Parameter(s)

To use copy type	You need this command
AG	`addag`
CL	`addcl`
PU	`addup`
SL	`adds`
UL	`addul and addmul`

Please note: While Vault does not require you to do so, there are some commands that it usually makes sense to combine with others within command lists. For example, if you include the `get` command, you will probably also want the `replace`, `reset`, and `update` commands in this list, too.

Use the `admcopy` command, described at the end of this chapter, to copy the commands in an existing command list to a new command list name.

Changing a Command List

Use the `chgcl` command to change the entries in a command list.

Using the Command-Line Format

In the example below, the `genuser` command list is changed. The command names in the `cmdfilea` file will be added to those already appearing in the `genuser` command list.

```
cichgcl
EDM> cmdlist=genuser chgtype=A cmdfile=cmdfilea
```

In the next example, the `genuser` command list is changed. The command names in the `cmdfiler` file will be removed from those already appearing in the `genuser` command list.

```
cichgcl
EDM> cmdlist=genuser chgtype=R cmdfile=cmdfiler
```

In the last example, the `admin` command list is changed. This list included the `all` command. Before executing the `cichgcl` command, create a file that includes the names of the commands you want to remove from the command list. In this example, the `admincoms` file contains the names of the commands to be removed.

```
cichgcl
EDM> cmdlist=admin chgtype=R cmdfile=admincoms
```

Notes for Changing a Command List

To remove or add commands in a previously defined command list, use the `chgcl` command. Modifications take effect immediately. Menu selections take effect the next time a user with the command list signs on to Vault.

Changing Entries with the Command-Line Format

Before using the command-line format to execute the `chgcl` command, create a local text file with the command names to be added or removed from this command list name. Create a separate file for each action (one file for the command names to be added and another file for those to be removed).

Include in the file only the command names to add or remove (one command name per line). Blank spaces are ignored. Blank lines are not allowed. Other data may be ignored or could cause the command to fail. Use the editing commands and file format for your operating system to create the file.

Specify the file that contains the command names to be added or removed with the `cmdfile` keyword.

Specify the type of change (adding or removing) with the `chgtype` keyword.

Please note: When you use the command-line format to execute the `chgcl` command, you can make only one kind of change at a time; that is, you cannot add and remove commands from a command list at the same time.

Changing Command Lists That Include All Commands

When using the command-line format to change a command list that includes the `all` entry, create a file containing the names of the commands to be removed. Execute the `cichgcl` command and enter `R` for `chgtype` and the name of the file that contains the names of the commands you are removing from the list.

When using the full-screen format to change a command list that includes the `all` entry, selectively remove commands from the command list displays.

The `chgcl` command allows you to add commands to a command list that includes the `allcmds` keyword. When you view the entry for the command list in the `COMMAND_LIST` table, you see `all` along with any other commands you added with the `chgcl` command. You can ignore commands listed with `allcmds`; you have access to all Vault commands.

Table 4-9 `chgcl` **Command Parameters**

Keyword	Parameter Description	Default
cmdlist	Existing command list name. Enter 8 or fewer letters and/or numbers.	None
chgtype	Indicates the type of action to be performed on the list. Enter: `A` (to add the commands) or `R` (to remove the commands).	None
cmdfile	Enter a local text file name, 80 or fewer alphanumeric characters, that contains the command names to be added or removed from the command list.	None

Deleting a Command List

Use the `delcl` command to delete a command list.

You cannot delete a command list currently assigned to any public or project user.

Table 4-10 `delcl` **Command Parameter**

Keyword	Parameter Description	Default
cmdlist	Existing command list name. Enter 8 or fewer letters and/or numbers.	None

Using the Command-Line Format

```
cidelcl
EDM> cmdlist=genuser
```

In this example, the `genuser` command list name is deleted.

Using Administrative Information

The copying procedures vary slightly, depending on the task you are performing. Use the `admcopy` command to copy administrative information including

- Authority groups
- Command lists
- Project user information
- Set of status levels for public files or projects
- User lists

Copying an Authority Group

To make a copy of an authority group

1. Enter `AG` for the copytype parameter.
2. Enter the original authority group name for the copyfrom parameter.
3. Enter the new authority group name for the copyto parameter.

You must have `addag` in your `PUBLIC` command list. The new authority group name must not already exist. The authority numbers in the original authority group are now associated with the new authority group name.

Copying a Command List

To make a copy of a command list

1. Enter `CL` for the copytype parameter.
2. Enter the original command list name for the copyfrom parameter.
3. Enter the new command list name for the copyto parameter.

You must have `addcl` in your `public` command list. The new command list name must not already exist. The commands in the original command list are now associated with the new command list name.

Copying Users Assigned to a Project

To copy the users on a project to a new project

1. Enter `PU` for the copytype parameter.
2. Enter the original Project ID for the copyfrom parameter.
3. Enter the new Project ID for the copyto parameter.

You must have `addup` in your `public` command list. The new Project ID must already exist in Vault. There cannot be any users already assigned to the new project. The project users assigned to the original Project ID are now assigned to the new Project ID.

Copying a Set of Status Levels

To make a copy of a set of status levels

1. Enter `SL` for the copytype parameter.
2. Enter the original Project ID or `PUBLIC` for the copyfrom parameter.
3. Enter the new Project ID or `PUBLIC` for the copyto parameter.

The new Project ID must already exist in Vault. You must have already been assigned to the new project. You must have the `ADDS` command in your command list for the new project. There cannot be any status levels already assigned to the new project. The status levels associated with the original Project ID (or the public scheme) are now associated with the specified Project ID (or the public scheme).

Copying a User List

To make a copy of a user list

1. Enter `UL` for the copytype parameter.
2. Enter the original user list name for the copyfrom parameter.
3. Enter the new user list name for the copyto parameter.

You must have `addul` and `addmul` in your `public` command list. The new user list name must not already exist in Vault. The ordered list of users associated with the original user list are now members of the new user list.

Table 4-11 `admcopy` **Command Parameters**

Keyword	Parameter Description	Default
copytype	Code to represent the copy type. Enter: `AG` (authority group) or `CL` (command list) or `PU` (project users) or `SL` (set of status levels) or `UL` (user list).	None
copyfrom	Copy source. Enter 24 or fewer letters and/or numbers.	None
copyto	Copy target. Enter 24 or fewer letters and/or numbers.	None

Using the Command-Line Format

```
ciadmcopy
EDM> copytype=Sa copyfrom=cargo,
EDM> copyto=Newcargo
```

In this example, the set of status levels belonging to the CARGO project is copied to the Newcargo project.

```
ciadmcopy
EDM> copytype=SA copyfrom=PUBLIC,
EDM> copyto=Newcargo
```

In this example, the set of status levels Vault uses for public files is copied to the Newcargo project.

Chapter 5

Setting Up and Modifying a Project

This chapter provides information and instructions for defining, modifying, and deleting revision code sequences, projects, and project users.

- How to Use Projects to Manage Data
- Planning Revision Code Sequences
- Creating a Revision Code Sequence
- Deleting a Revision Sequence
- Parameters for Creating a Project
- Creating a Project
- Parameters for Assigning Users to Projects
- Assigning Users to Projects
- Parameters for Modifying Project Attributes
- Changing Project Attributes
- Parameters for Changing Project User Authorities
- Changing a User's Project Authority
- Removing a User from a Project
- Deleting a Project

How to Use Projects to Manage Data

A project is a set of files that can be defined and managed as a unit. A project has its own authority scheme. Projects provide a way to divide the database logically into different segments to which users can be added and files can be stored. At our example site, the High Flying Airplane Company, there are two projects, JET and CARGO.

Project Authority Schemes

Each project has its own authority scheme that controls access to files stored under that project. This authority scheme can be the same or different from the public or any other project authority schemes. The project authority scheme includes the set of status levels assigned to the project and the authority numbers or groups and command lists assigned to users when you add them to the project.

Project Users

Only users assigned to a project can access the files for that project. Each user is assigned a set of authorities for the project. These authorities are only used in relation to that particular project's files. Project authorities consist of

- Project command list
- Project read authority
- Project write authority
- Project administrative authority

At the High Flying Airplane Company, SNOOPY, AMELIA, and ORVILLE are assigned to the CARGO project.

As Vault administrator, SNOOPY can use all Vault commands and can perform all project administrative functions. He has

- A read and write authority number of 99 (the highest value)
- A public and project command list with the `all` command (all Vault commands)
- Administrative privileges on the project (can reset or sign in another user's files or parts)

As project leader, AMELIA can access all CARGO project files and can modify those at or below the final review status level which is associated with the authority number 70.

She has:

- A write authority number of 70 and a read authority number of 99
- A project command list with the `all` command (all Vault commands)

Because she has the `all` command only in her CARGO project command list, she can add users (among other tasks) to the CARGO project (but not to Vault or to other projects).

ORVILLE, the other CARGO project member, has a set of authorities assigned to him that allow him to perform file access functions.

Command Lists and Authority Groups

When you define a command list or authority group, you do not specify whether it is for public or project files. You assign a command list and possibly an authority group to a user when adding that user to Vault or assigning that user to a project. After the command list or authority group is associated with a user, it is considered a project or public command list or authority group. A particular command list or authority group can be used for public files and also for one or more projects.

Please note: See the beginning of Chapter 4, "Creating and Modifying Authority Scheme Components" for a discussion of public and project authority schemes.

Sequence of Steps for Setting Up a Project

Here are the steps and commands used to set up a project:

1. Set up revision code sequences. Instructions for using the `addrs` (Add Revision Sequence) command are in this chapter.
2. Define the unique project name. Instructions for using the `addp` (Add Project) command are in this chapter.
3. Define the project command list(s). Instructions for using the `addcl` (Add Command List) command are in Chapter 4, "Creating and Modifying Authority Scheme Components".
4. Unless you are using only authority numbers, define the project authority groups. Instructions for using the `addag` (Add Authority Group) command are in Chapter 4, "Creating and Modifying Authority Scheme Components".
5. Add a project leader (or yourself) to the project. Instructions for using the `addup` (Add User to Project) command are in this chapter.

6. Define the project status levels—you must be assigned to the project and the `adds` (Add Status level) command must be in your command list for that project.

   Instructions for using the `adds` command are in Chapter 4, "Creating and Modifying Authority Scheme Components". When you define a status level for a project, specify a previously defined project ID. When you leave the project ID parameter blank, Vault associates the status level with the public authority scheme. Planning information for defining status levels is in the *Vault System Administrator Guide*.

7. Store existing project files.

8. Add other users to the project. Instructions for using the `addup` (Add User to Project) command are in this chapter.

Planning Revision Code Sequences

During Vault software installation, you load a revision code sequence that you define. DEFAULT is the reserved name of this sequence. Vault uses DEFAULT to assign revision codes to public and private files.

Vault does this by means of a dummy project with a project ID of blanks. You can change the revision code sequence for public and private files with the chgp (Change Project) command. Assign a different revision code sequence when you execute the command. Be sure to enter blanks in the project ID field.

Appendix B provides examples of revision code sequences and the effects of reclassifying files.

Assigning Revision Sequences to Projects

When you create a project, you must specify a revision code sequence for the project to use. You can assign

- The default revision code sequence by specifying DEFAULT.
- An empty revision code sequence by specifying NONE. The effect of this is as if the project did not have an assigned revision code sequence.
- A previously defined revision code sequence that has not yet been assigned to any other project.
- A previously defined revision code sequence that is already being used by one or more projects.

Reserved Words for Revision Sequences: The addrs and delrs commands allow you to create and delete revision code sequences. When you define a revision sequence, you assign a revision code sequence name. There are two reserved names for revision code sequences - DEFAULT and NONE.

DEFAULT is the name of the revision code sequence you loaded during software installation.

NONE is the name of the revision code sequence that is empty. It indicates that Vault does not assign revision codes to files belonging to projects that have NONE as the revision code sequence. You can, however, assign revision codes if you want to.

When You Do Not Want to Assign a Revision Sequence: When you add a project to Vault and you do not want the project to have a revision code sequence,

you enter NONE for the revision sequence name. When you store files in the database that are assigned to this project, Vault does not assign a revision code.

When you execute the chgfsc, reqrvw, or rsvp command, and the effect of that command is to change the file from an in-work status to a released status, the file must have a revision code. You must use the chgfrev command to assign a revision code. Then you can execute the chgfsc, reqrvw, or rsvp command. Assigning a revision code sequence to a project has no effect on files previously stored in that project.

Changing the Sequence Used for Public Files

If you want to change the revision code sequence loaded for public and private files, execute the chgp command. Leave the Project ID field blank and enter the name of the previously defined revision code sequence that you want to use.

Changing the DEFAULT Sequence

If you want to change the DEFAULT revision code sequence, you can reuse the utility that loads DEFAULT at any time after software installation. See your installation guide for instructions. You cannot delete the DEFAULT revision codes sequence with the delrs (delete revision code sequence) command.

When you modify the DEFAULT revision sequence, or change the sequence associated with a project, the revision codes of previously stored files are not affected until a version of the file needs a new revision code. Vault then assigns the next revision code in the new sequence. For example, if a file has the third revision code in the former sequence, Vault assigns the fourth revision code in the new sequence. Files stored in the Vault database for the first time receive the first revision code in the new sequence.

Creating a Revision Code Sequence

To create a revision code sequence, use the `addrs` command.

Table 5-1 `addrs` **Command Parameters**

Keyword	Parameter Description	Default
revname	Unique name you assign to the revision code sequence you are defining. Enter 8 or fewer letters and/or numbers.	None
revfile	Name of the local file that contains the list of codes assigned to the revision sequence you are creating. Enter 80 or fewer letters and/or numbers.	None

Using the Command-Line Format

Before you execute the `addrs` command with the command-line format, create a file with a local editor. In this file, list the revision codes that will be assigned to the revision sequence you are creating.

Use one line for each revision code. The order of the revision codes in the file is the order of the codes in the revision code sequence. Example:

```
ciaddrs
EDM> revname=Model revfile=revcodes.formodeL
```

Deleting a Revision Sequence

Execute the `delrs` command to delete a revision code sequence.

You cannot delete a revision sequence

- Associated with a project
- Used for public files
- Having the name `DEFAULT`

Table 5-2 `delrs` **Command Parameter**

Keyword	Parameter Description	Default
revname	Previously defined name of a revision code sequence. Enter 8 or fewer letters and/or numbers.	None

Using the Command-Line Format

```
cidelrs
EDM> revname=Model
```

Parameters for Creating a Project

To create a project, use the `addp` command. When you execute this command you define a unique project ID. You must also enter a revision code sequence name. All other project data is optional. If this information is useful at your site, specify a

- Project name
- Contract number
- Unit of measurement
- Project description

Vault does not use these parameters. You can enter any information you choose. For example, you could enter the expected date of completion in the contract number field. See the *Vault Manager Guide* for more information about entering information for user-defined parameters.

Please note: You must add a project ID with the `addp` command before you can perform any project related tasks, such as

- Adding the project's status levels
- Adding users to the project
- Storing files under the project authority scheme

Table 5-3 `addp` **Command Parameters**

Keyword	Parameter Description	Default
projid	Unique project identifier. Enter 8 or fewer letters and/or numbers.	None
revname	Previously defined name of a revision code sequence. Enter 8 or fewer letters and/or numbers. Enter: NONE for no revision sequence.	None
projname	Project name. Enter 15 or fewer alphanumeric characters. Optional.	None
contract	Contract number. Enter 25 or fewer alphanumeric characters. Optional.	None
units	Unit of measure code. Enter 5 or fewer alphanumeric characters, for example, FEET (feet), METER (meters), CM (centimeters). Optional.	None
projdesc	Project description. Enter 30 or fewer alphanumeric characters (including blanks). Optional.	None

Creating a Project

Execute the `addp` command to create a project.

Using the Command-Line Format

```
ciaddp
EDM> projid=Cargo revname=Default
```

In this example, the Cargo project is created with the Default revision sequence. In this example, none of the optional parameter information is specified.

```
ciaddp
EDM> projid=Cargo revname=Model,
EDM> projname=Cargo1 projdesc='Cargo plane project.',
EDM> contract=6789 units=Meter
```

In this example, the CARGO1 project is added to Vault. The other parameter fields describe various details of the project.

```
ciaddp
EDM> projid=Cargo2 revname=None,
EDM> projname=Cargo projdesc=Cargo Plane Project,
EDM> contract=6789 units=Meter
```

In this example, the project could not be added because the single quotation marks were omitted from the project description field. This field contains blank spaces; therefore, the quotation marks are required.

Parameters for Assigning Users to Projects

To assign users to a project, use the `addup` command. When you execute the `addup` command, you assign project authorities to each user:

- Project command list
- Project read authority number or read authority group
- Project write authority number or write authority group
- Project administrative privileges

If both authority numbers and authority groups are assigned to a user, the authority groups override the authority numbers.

Administrative Privileges for Users

Enter a `Y` (yes) or `N` (no) for the administrative privileges parameter to indicate whether or not the user has administrative privileges in the project authority scheme. Users with administrative privileges

- Can reset, update, and replace files signed out to another user (execute the `reset`, `update`, and `replace` commands for files not signed out to them)
- Can change the status of a file (`chgfsc` command) without proceeding in a sequential order through the status codes.

These privileges apply to files they have the authority to access.

Please note: Before you can assign a user to a project, you must perform the steps below. Instructions for steps 3, 4, and 5 are described in Chapter 4, "Creating and Modifying Authority Scheme Components".

1. Add the user to Vault (with `addu`). (See Chapter 3, "Giving Users Access to Vault".)
2. Add the project name to Vault (with `addp`).
3. Define the appropriate command list(s) (with `addcl`).
4. Define the project status levels (with `adds`).
5. Define the appropriate authority groups (with `addag`).

Project Command Lists

To add the first user to a project, you must have the `addup` command in your public command list.

After a person is assigned to a project, that person can add other users to the project if the `addup` command is in his or her project command list.

For example, at the High Flying Airplane Company, the system administrator has the `all` command in his public command list so he can add project users to any project. Each project leader has the `addup` command in his or her project command list so they can add users to their own projects as long as those users have already been added to Vault.

Please note: To copy the users on a project to a new project, use the `admcopy` command described in Chapter 4, "Creating and Modifying Authority Scheme Components".

Table 5-4 `addup` **Command Parameters**

Keyword	Parameter Description	Default
userid	Existing Vault User Identifier. Enter 12 or fewer letters and/or numbers.	None
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers.	None
cmdlist	Existing command list name. Enter 8 or fewer letters and/or numbers.	None
reada	Read authority number that sets a user's read authority limit. Enter a number between 0 and 99 *unless you enter a read authority group name.*	00
readag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a read authority number.*	None
writea	Write authority number that sets a user's write authority limit. Enter a number between 0 and 99 *unless you enter a write authority group name.*	00
writeag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a write authority number.*	None
admin	Indicates whether or not the user is given special administrative privileges for this project's files. Enter: `Y` (yes) or `N` (no).	N
areada	Associates a Read Authority Number. Enter a number between 0 and 99. A number entered here means a range from 0-<number>.	0
areadag	Associates a Read Authority Group Name. Enter an authority group name up to 8 alphanumeric characters *unless you enter a read authority number.*	
awritea	Associates a Write Authority Number. Enter a number between 0 and 99.A number entered here means a range from 0-<number>.	0
awriteag	Associates a Write Authority Group Name. Enter authority group name up to 8 alphanumeric characters *unless you enter a write attribute authority number.* If both are entered the authority number is ignored.	

Assigning Users to Projects

Execute the `addup` command to assign users to a project.

Using the Command-Line Format

```
ciaddup
EDM> userid=Beryl projid=CARGO cmdlist=projldr,
EDM> reada=90 writeag=Wauth1 admin=Y
```

In this example, user Beryl is added to the Cargo project. User Beryl:

- Can use the commands in the PROJLDR command list
- Has a read access authority limit of 90
- Has a write access authority group named WAUTH1
- Has administrative privileges for this project

```
ciaddup
EDM> userid=Erica projid=Cargo cmdlist=projldr,
EDM> reada=90 readag=Rauth1 writeag=Wauth1 admin=Y
```

In this example, user Erica is added to the Cargo project. Erica is assigned both a read authority number (90) and a read authority group (Rauth1).

The authority numbers associated with the Rauth1 authority group name determine Erica's read authority on the project, not the read authority number (90) specified on the command line. This is because when both authority numbers and authority groups are assigned to a user, Vault assigns both attributes but uses the authority groups for file access transactions and ignores the authority numbers. The authority groups override the authority numbers.

Parameters for Modifying Project Attributes

Use the `chgp` command to modify project attributes:

- Project name
- Revision code sequence
- Contract number
- Unit of measurement
- Project description

When you change the revision code sequence for a project, Vault does not change the revision codes already assigned to project files.

You can change revision codes of previously stored files with the `chgfrev` command. You can only do this, however, if the file does not have a released status and if the associated revision code sequence is NONE.

Please note: You cannot change the project ID with the `CHGP` command. To do this

1. Delete the project (`delp`)
2. Add it again with the changed ID (`addp`)

Entering Data with the Command-Line Format

If you are using the command-line format, enter changed or new data with keywords as follows:

- Enter a cross hatch (#) as the first character following the equals sign (=) to delete previous data and not replace it. For example, to delete the current contract number and not replace it, the keyword should look like this

  ```
  contract=#
  ```

- Enter the new or changed data following the equals sign (=). For example

  ```
  CONTRACT=333333
  ```

Table 5-5 `chgp` **Command Parameters**

Keyword	Parameter Description	Default
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers.	None
revname	Previously defined name of a revision code sequence. Enter 8 or fewer letters and/or numbers. Optional.	None
projname	Project name. Enter 15 or fewer alphanumeric characters. Optional.	None
contract	Contract number. Enter 25 or fewer alphanumeric characters. Optional.	None
units	Unit of measure code. Enter 5 or fewer alphanumeric characters, for example, FEET (feet), METER (meters), CM (centimeters). Optional.	None
projdesc	Project description. Enter 30 or fewer alphanumeric characters (including blanks). Optional.	None

Changing Project Attributes

Execute the `chgp` command to modify the attributes of a project.

Using the Command-Line Format

```
cichgp
EDM> projid=Cargo projname=X2,
EDM> contract=CT2334-02,
EDM> projdesc='Second Engine Design' UNITS=I
```

In this example, all of the description information for the Cargo project is changed.

```
cichgp
EDM> projid=Cargo projdesc=#
```

In this example, the current project description information is deleted by entering the pound sign character (#) as the first character following the equals sign (=) after the `projdesc` keyword. Other parameter information associated with this project ID remains unchanged.

Parameters for Changing Project User Authorities

The `chgup` command modifies the project authorities for a user assigned to a project. You can change the user's project

- Command list name. (The new name must have been previously defined with the `addcl` command.)
- Read and write authorities. (If you are changing the authorities to new authority groups, the group names must have been previously defined with the `addag` command.)
- Administrative privileges. Users with administrative privileges can reset, update, and replace files signed out to another user (execute the `reset`, `update`, and `replace` commands for files not signed out to them), and they can change the status of a file (`chgfsc` command) without proceeding in a sequential order through the status codes. These privileges apply to files they have the authority to access.

Please note: You cannot change a User ID with the `chgup` command. To do this you

1. Remove the user from the project (with `remup`).
2. Delete the user from Vault (with `delu`).
3. Add the new Vault User ID (with `addu`).
4. Assign the new Vault User ID to the project (with `addup`).

Entering Data

Enter changed or new data in response to the prompts or following the equals sign (=). To delete a read or write authority group name and not replace it with another name, enter a pound sign character following the equals sign (=).

You will need to do this when the user is currently assigned a read or write authority group name and must be changed to a read or write authority number.

When to Change Related Authority Schemes: For example, CHARLES is currently assigned a read authority group. To change his read authority from a group to a number, do not enter the number and leave the read authority group name information unchanged.

The same applies if you are changing a write authority from an authority group to a single authority number.

This is because if both authority numbers and authority groups are assigned to a user, Vault uses the authority groups and ignores the authority numbers. You must add the authority number and delete the authority group name.

If you are changing a user's authority from a number to an authority group name, you do not need to delete the authority number. In this case, the number is ignored.

Table 5-6 `chgup` **Command Parameters**

Keyword	Parameter Description	Default
userid	Existing Vault User IDentifier. Enter 12 or fewer letters and/or numbers.	None
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers.	None
cmdlist	Existing command list name. Enter 8 or fewer letters and/or numbers.	None
reada	Read authority number that sets a user's read authority limit. Enter a number between 0 and 99 *unless you enter a read authority group name.*	None
readag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a read authority number.*	None
writea	Write authority number that sets a user's write authority limit. Enter a number between 0 and 99 *unless you enter a write authority group name.*	None
writeag	Existing authority group name. Enter 8 or fewer letters and/or numbers *unless you enter a write authority number.*	None
admin	Indicates whether or not the user is given special administrative privileges for this project's files. Enter: `Y` (yes) or `N` (no).	None
areada	Associates a Read Authority Number. Enter a number between 0 and 99. A number entered here means a range from 0-<number>.	0
areadag	Associates a Read Authority Group Name. Enter a authority group name up to 8 alphanumeric characters *unless you enter a read authority number.*	
awritea	Associates a Write Authority Number. Enter a number between 0 and 99.A number entered here means a range from 0-<number>.	0
awriteag	Associates a Write Authority Group Name. Enter authority group name up to 8 alphanumeric characters *unless you enter a write attribute authority number.* If both are entered, the authority number is ignored.	

Changing a User's Project Authority

Execute the `chgup` command to modify the project authorities of a user assigned to a project.

Using the Command-Line Format

```
cichgup
EDM> userid=Beryl projid=Cargo writea=50,
EDM> writeag=#
```

In this example, BERYL is assigned a new write authority limit (50) for the CARGO project. Because she was previously assigned a write authority group name, the name must also be removed (using the pound sign (#) character).

```
cichgup
EDM> userid=Beryl projid=Cargo writeag=Cargog
```

In this example, Beryl is assigned a new write authority group (Cargog) for the CARGO project. If she also has a write authority number, Vault uses the authority group for file access transactions and ignores the authority number.

Removing a User from a Project

Execute the `remup` command to remove a user from one project or all projects to which the user is assigned.

Please note: Removing a user from a project (or all projects) does not delete the user from Vault. To do this, use the `DELU` command (described in Chapter 3, "Giving Users Access to Vault").

Removing a User from One Project

To remove a user from a single project, enter the Vault User ID and the project ID. You cannot remove a user from a specified project if he or she has a file associated with that project signed out or a user-defined table associated with the project opened.

Removing a User from All Projects

You cannot remove a user from all projects at once if he or she has any file or any user-defined table on any project signed out. To remove a user from all projects

1. Enter the Vault User ID.
2. Leave the Project ID field blank.
3. Enter: `Y` for the Remove user from all projects parameter.

Table 5-7 `remup` **Command Parameters**

Keyword	Parameter Description	Default
userid	Existing Vault User Identifier. Enter 12 or fewer letters and/or numbers.	None
allproj	Indicates whether you are removing the user from all projects. Enter: `Y` (yes) or `N` (no).	None
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers. (This parameter is ignored if you enter `Y` in the `Remove user from all projects` field.)	None

Using the Command-Line Format

```
ciremup
EDM> userid=Wilbur allproj=N projid=Jet
```

Deleting a Project

Execute the `delp` command to delete a project. This command deletes the project name and removes the project users from the project.

You cannot delete a project if there are any

- Active (not archived) files associated with the project
- Status levels associated with the project
- User-defined tables associated with the project

Table 5-8 `delp` **Command Parameter**

Keyword	Parameter Description	Default
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers.	None

Example of Using the Command-Line Format

```
cidelp
EDM> projid=Jet
```

In this example, the Jet project is deleted.

Chapter 6

Setting Up Release/Revision Controls

This chapter describes how to set up the review process for a project; including establishing status levels within a project and associating user lists and exit criteria to project status levels. The following topics are presented:

- Overview of Release/Revision Control
- Defining a User List Name
- Parameters for Adding Members to a User List
- Adding Members to a User List
- Parameters for Associating a User List and Status Code
- Associating a User List and Status Code
- Removing a User List/Status Code Association
- Removing Members from a User List
- Deleting a User List
- Establishing and Using Review Procedures

Overview of Release/Revision Control

The Release/Revision Control Facility manages the review process for files, parts, and file sets. It controls the status level of a file based on approval or rejection by reviewers. Use the commands described in this chapter to set up automatic review procedures.

When a stage of work is completed on a file, someone initiates a review with the `REQRVW` command. The members of the review team determine if the file should be changed to the next status code in its associated authority scheme.

When a user initiates a review, Vault sends a message to all members of the user lists associated with the file's current status code. The content of the message depends on the type of user list. After reviewing the file, reviewers issue the `RSVP` command to register their approval or rejection of the changes.

Sequence of Steps

The steps to set up automatic review procedures are

1. Define user list names. See "Defining a User List Name" on page 6-5.
2. Add members to user lists. See "Adding Members to a User List" on page 6-8.
3. Associate each user list with the appropriate status code. See "Associating a User List and Status Code" on page 6-11.

User Lists

A user list is a list of Vault User IDs and/or Vault user list names. The Release/Revision Control Facility employs user lists to send messages automatically to Vault users during the review cycle for a file. Message content depends on the type of user list.

There are three types of user lists. Each type functions differently. You specify user list type when you associate the list with a status code. The three types are:

- Notification list—Users on this list receive a message indicating that a file review has been initiated with the `reqrvw` command. When the review is complete, users receive a message indicating the results.
- Review list—Users on this list receive a review message in the order of their user list sequence numbers. Reviewers use the `rsvp` command to register their approval or rejection of the work. When the review is complete, reviewers who registered their approval or rejection receive a message indicating the results.
- Post-approval list—users on this list receive a message indicating the review results if the file status code change is approved.

Changing User Lists During a Review Cycle: You can modify review and notification lists while the list is in use by a review cycle. For a review list, however, the users and their sequence numbers at the time the review is initiated remain the same throughout that review cycle.

Approval Schemes

When you designate a user list as a review list (list type R) with the `addusa` command, you must specify which approval scheme to use with the list. There are three approval schemes that you can establish for a review.

- Unanimous—The file is approved for status code change if all reviewers approve the file changes.
- Majority—The file is approved for status code change if a majority of reviewers approve the file changes.
- Number of rejects—The file is approved for status code change unless one more than the specified number of rejections are registered.

If File Changes Are Approved: If the required number of reviewers approves the file, the file's status code is incremented and the file can be signed out for the next stage of work. Users on the post-approval list and the review initiator are informed of the results.

If File Changes Are Rejected: If the required number of reviewers does not approve the file, the file remains at the same stage of work and its status code is not changed. The file can be signed out again. The review initiator, users on the notification list, and users on the review list are notified that the work was rejected.

Automatic Status Code Changes

If a status code does not have a review list associated with it Vault automatically increments the file status to the next status when you execute the `reqrvw` command. Users on the associated post-approval and notification lists (if any) are notified of the automatic status change.

Multiple File Revisions

A review can occur on more than one file revision at the same time. All part files must be at the same status for the part to be reviewed. For a file set, all files with a blank revision definition in the file set are candidates for a status change. These files must be at the same status level in the file set to be reviewed.

Email Trigger

You can activate the email trigger so that when Vault sends messages as part of release/revision control procedures, Vault also sends system email messages. This allows users to get Vault messages without signing on to Vault. See *Vault System Administrator Guide* for information about activating the email trigger.

Defining a User List Name

To define a user list name, use the `addul` command.

Table 6-1 `addul` **Command Parameters**

Keyword	Parameter Description	Default
userlist	Unique user list name. Enter 24 or fewer letters and/or numbers.	None
listdesc	Description of the user list. Enter 30 or fewer characters including blanks. Optional.	None

Using the Command-Line Format

```
ciaddul
EDM> userlist=Cargoul
```

In this example, the Cargoul user list is defined.

```
ciaddul
EDM> userlist=jetul,
EDM> listdesc='Members Of The Jet Project'
```

Be sure to enclose the list description in single quotation marks when it includes one or more blank spaces.

Using the Graphical User Interface

See “Creating a User List” on page 6-17.

Parameters for Adding Members to a User List

To add users to a previously defined user list, execute the `addmul` command. A user list member can be an Vault User ID or a previously defined user list name.

Please note: User list members must have Vault User IDs already assigned to them (with the `addu` command) before they can be added to a user list.

Table 6-2 `addmul` **Command Parameters**

Keyword	Parameter Description	Default
userlist	Existing user list name. Enter 24 or fewer letters and/or numbers.	None
ulmname	Existing Vault User ID or previously defined user list name. Enter 24 or fewer letters and/or numbers.	None
seqnum	Establishes order in the user list. Enter a number, 0 through 9999.	0000
ulmtype	Code that indicates the type of member being added to the list. Enter: `U` for User ID, `A` for a user list in which all members must respond to a review, or `O` for a user list in which one member only can respond to a review.	U

Sequence Numbers

Sequence numbers establish the order in which the members receive a review message. The person(s) or user list with the lowest number receives the review message first, followed by the next lowest number, etc. Sequence numbers can be 0 through 9999. Sequence numbers do not affect notification or post-approval messages. If you are setting up a list for notification or post-approval messages, use the default value for sequence number (0000).

Member Names

The same Vault User ID or user list name can appear on a user list more than once. Each occurrence, however, must be associated with a different sequence number. For example, at our sample site, Beryl is on the Cargoul user list at sequence numbers 10 and 40.

The same Vault User ID or user list name can also be on more than one user list. For example, Amelia can be on the Cargoul user list and the `jetul` user list at the same or different sequence numbers.

Embedded User Lists

You can embed user lists for one level only. For example, if user list A is a member of user list B

- List A cannot include a user list name as one of its members.
- List B cannot be a member of another user list.

The sequence number assigned to an embedded user list becomes the sequence number of each member of the list when the top level user list is used as a review list.

For example, the Cargoul user list has two users, Amelia with a sequence number of 10, and Beryl with a sequence number of 40. Suppose you add the Cargoul user list with a sequence number of 100 to the `jetul` user list. When you initiate a review and the `jetul` user list is the review list, Amelia and Beryl both receive review messages at sequence number 100.

Member Types

When you add a member to a user list, you specify it to be one of three types.

- `U` if it is a User ID
- `A` if it is a user list name and all members of this embedded list must respond when the parent list is used as a review list
- `O` if it is a user list name and only one member of the embedded list must respond when the parent list is used as a review list

When you specify `O` as the member type, then only the member that responds receives a message when the review is complete.

Adding Members to a User List

Execute the `addmul` command to assign Vault User IDs and user list names to a user list.

Using the Command-Line Format

```
ciaddmul
EDM> userlist=Cargoul ulmname=Charles,
EDM> SEQNUM=0200
```

In this example, Charles is added to the Cargoul user list. His sequence number is 0200.

```
ciaddmul
EDM> userlist=jetul ulmname=qa seqnum=250,
EDM> ulmtype=A
```

In this example, the `qa` user list is added to the `jetul` user list. The sequence number is 250; only one member of the `qa` list can respond to a review request.

```
ciaddmul
EDM> userlist=cargonotificationlist,
EDM> ulmname=CHARLES
```

In this example, Charles is added to the cargonotificationlist user list. Because this list is to be used only for notification messages, the sequence number and member type defaults are used.

Parameters for Associating a User List and Status Code

To set up a release revision control scheme, use the `addusa` command. This command associates previously defined user lists with existing status codes. The next time a review is initiated for a file having the associated status code, Vault sends review and/or notification messages to members on the associated list(s). The status cannot change to the next status until the file is reviewed.

Designating the Function of the User List

When you execute the `addusa` command, you designate a user list as a notification, review, or post-approval list. You can associate one, two, or three lists with a status code—one for notification, one for review, and/or one for post-approval.

Establishing the Approval Scheme

For review lists, this command also establishes the approval scheme and termination count for the review. Approval can be

- Unanimous
- By majority
- If a specified number of rejections is not exceeded

Basing approval on the number of rejections is useful when a unanimous vote is too restrictive and a majority vote is too loose. For example, if there are 100 reviewers, 100 would have to vote yes for unanimous approval but only 51 would have to vote yes for majority approval. Setting the reject count to 10 would require 90 approvals.

Reject Count

When approval is determined by the number of rejections, you also specify a reject count. The reject count determines the maximum number of rejections that can be registered and still allow approval of the file changes. For example, if the reject count is two and 0, 1, or 2 rejections are registered, the review is approved. When the number of rejects is exceeded, approval is not possible, but the review can continue until either

- The termination count is reached
- All reviewers have responded

Termination Count

The termination count specifies the number of rejections that causes the review to terminate. Each review list has a termination count. If the termination count is reached, Vault ends the review.

When the termination count is one, Vault terminates the review after receiving one rejection. When the termination count is two, Vault terminates the review after receiving two rejections, and so on. The default termination count is zero. This causes the review to continue until all reviewers have registered their approval or rejection.

The number of rejections that are allowed for the review to continue is the termination count minus one.

You must be careful about the numbers you specify for termination count and reject count. If you specify three for the reject count and one for the termination count, then the review ends after one rejection when there is still the possibility of approval. On the other hand, if you specify one for the reject count and three for the termination count, then after one rejection the review continues even though approval is not possible.

Table 6-3 `addusa` **Command Parameters**

Keyword	Parameter Description	Default
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers or leave blank to associate the user list with a public status code.	Blank
statcd	Existing code that represents a phase of work in an Vault authority scheme. Enter 8 or fewer alphanumeric characters.	None
userlist	Existing user list name. Enter 24 or fewer letters and/or numbers.	None
listtype	Code that specifies how the user list functions. Enter: `R` (review) or `N` (notification) or `P` (post-approval).	None
termcnt	Number of rejections that causes the review to end. If list type is R, enter a number, 0 through 999.	0
apptype	Code that specifies the type of approval needed. If list type is R, enter: `U` (unanimous) or `M` (majority) or `N` (number of rejects).	None
rejcnt	Maximum number of rejections allowed for approval. When this number of rejections is surpassed, the review can continue but approval is not possible. If approval type is N, enter a number, 0 through 999.	0

Associating a User List and Status Code

Execute the `addusa` command to associate a user list with a status code.

Using the Command-Line Format

```
ciaddusa
EDM> projid=Jet statcd=re userlist=jetul,
EDM> listtype=N
```

In this example, a user list/status code association is established between the `jetul` list and the re status code for the Jet project files. Because the list type is N (notification), no other parameters are supplied.

```
ciaddusa
EDM> projid=Jet statcd=re userlist=jetul,
EDM> listtype=R termcnt=1 apptype=U
```

In this example, a user list/status code association is established between the `jetul` list and the `RE` status code for the Jet project files. For the status code change to take place, all reviewers must approve the change (`apptype=U`). After one rejection response is recorded, the review is terminated (`termcnt=1`).

```
ciaddusa
EDM> projid=Jet statcd=re userlist=jetul,
EDM> listtype=R apptype=N rejcnt=1 termcnt=1
```

In this example, a user list/status code association is established between the `jetul` list and the `RE` status code for the Jet project files. For the status code change to take place, all reviewers or all reviewers except one must approve the change (`rejcnt=1`). After one rejection response is recorded, the review is terminated (`termcnt=1`). Note that this may cause the review to end prematurely, as follows.

- If there are two reviewers on the review list and the first reviewer approves the file and the second reviewer rejects the file changes, the review terminates after the second response is received and the status change takes place.
- If there are two reviewers on the review list and the first reviewer rejects the file, the review terminates and the status change does not take place. Because the review terminated before the second reviewer entered a response, this person's response is unknown. If it would have been an approval response, the status change should have taken place.

```
ciaddusa
EDM> projid=Jet statcd=re userlist=jetul,
EDM> LISTTYPE=P
```

In this example, a user list/status code association is established between the `jetul` list and the re status code for the Jet project files. Because the list type is P (post-approval), no other parameters are supplied.

Removing a User List/Status Code Association

To remove the association between a status code and a user list, use the `remusa` command.

The next time a request for review occurs for a file at the specified status level, the list of users is no longer notified or sent review messages. If a review list association is removed, files are automatically moved to the next status level when a request for review is performed.

Please note: Removing a user list/status code association during a review cycle has no effect on that cycle.

Table 6-4 `remusa` **Command Parameters**

Keyword	Parameter Description	Default
userlist	Existing user list name. Enter 24 or fewer letters and/or numbers.	None
listtype	Code that specifies how the user list functions. Enter: `R` (review) or `N` (notification) or `P` (post-approval).	None
statcd	Existing code that represents a phase of work in an Vault authority scheme. Enter 8 or fewer alphanumeric characters.	None
projid	Existing project identifier. Enter 8 or fewer letters and/or numbers or leave blank to dissociate the user list from a public status code.	Blank

Using the Command-Line Format

```
ciaddusa
EDM> userlist=jetul listtype=N statcd=RE,
EDM> projid=Jet
```

In this example, a user list/status code association is removed between the `jetul` list and the RE status code for the Jet project files.

Removing Members from a User List

To remove a member from a user list, use the `remmul` command.

Removing all the members of a user list does not delete the user list name from Vault. The user list name remains in Vault and you can add members to it until you delete it with the `delul` (delete a user list name) command.

You can only remove one Vault User ID at a given sequence number or one user list name each time you execute the `remmul` command. For example, if a user is on a user list with four different sequence numbers, you must execute the `remmul` command four times to completely remove this user from the list.

If you want to remove all members from a user list and delete the user list name, execute the `delul` (delete user list) command. See "Deleting a User List" on page 6-15 for details.

If the user list is currently in use as a review list, removing a person during a review cycle has no effect on that cycle.

Table 6-5 `remmul` **Command Parameters**

Keyword	Parameter Description	Default
userlist	Existing user list name. Enter 24 or fewer letters and/or numbers.	None
ulmname	Existing user list member name. Enter 24 or fewer letters and/or numbers.	None
seqnum	Enter the number, 0 through 9999, that corresponds to this list member's existing sequence number.	None
ulmtype	Code that indicates the type of list member. Enter: `U` for User ID, `A` for a user list in which all members must respond to a review, or `O` for a user list in which one member only can respond to a review.	U

Using the Command-Line Format

```
ciremmul
EDM> userlist=jetul ulmname=qa,
EDM> seqnum=0250 ulmtype=A
```

In this example, the `qa` user list at sequence number 250 is removed from the `jetul` user list.

Deleting a User List

Execute the `delul` command to delete a user list. This has the effect of removing all members from the list as well as deleting the user list name.

If the user list is currently associated with a status code as a review list, notification list, or post-approval list, it cannot be deleted. Execute the `remusa` (remove a status code/user list association) command (described earlier in this chapter) and then execute the `delul` command.

Table 6-6 `delul` **Command Parameter**

Keyword	Parameter Description	Default
userlist	Existing user list name. Enter 24 or fewer letters and/or numbers.	None

Using the Command-Line Format

```
cidelul
EDM> userlist=cargoul
```

In this example, the `cargoul` user list is deleted.

Establishing and Using Review Procedures

The project or Vault administrator can establish review/release criteria. User members can initiate and respond to reviews.

Setting Up the Review Process for a Project

The Vault administrator or project leader can set up the review process for a project as follows:

1. From the menu bar, choose View > Admin > Projects. The Projects view appears.
2. From the browser, choose the project you want to review.
3. From the menu bar, choose Edit > Status Levels. The Status Levels dialog box appears.
4. Click Exit Criteria. The Status Level Exit Criteria dialog box appears.
5. From the List Type popup menu, select Review.
6. Click Create to create a user list from the users. The Add Status Level Criteria dialog box appears.
7. Fill in the boxes appropriately (The User List should already exist and have members. If it is not available, see the section "Creating a User List" on page 6-17.) You can click the Editor Buttons next to the Status Level and User List boxes to display a browser.
8. When you have set this dialog box up appropriately, click OK.
9. Vault will indicate that the user is added to the list.
10. When you have completed the list, click Cancel from each of the open dialog boxes to close them.

You can set up Vault to trigger email and Vault messages to initiate and track a review process. Base your event triggers to occur when a specified action occurs on a registered object. See "Email Trigger" on page 6-4.

You can also use this procedure to perform the following tasks by substituting the appropriate selection in the List Type box, described in Step 4:

- Notification
- Post Approval

Please note: The *Vault End User Guide* provides an overview of the release/revision cycle and instruction on how end users can participate in a review.

Creating a User List

Before you can request a review you must define a user list. The user list must contain the names of all Vault users to notify of the review request.

1. From the menu bar, choose View > Admin > User Lists.The User Lists view appears.
2. From the menu bar, choose Edit > Members. The Edit User List Member dialog box appears.
3. Choose the Member Type.
4. Fill in the other boxes appropriately.
5. Click OK.

Requesting a Review

You, or any user in the appropriate user list, can initiate a review cycle for the following objects using the Graphical User Interface. The storing Vault must be active.

- Files
- Parts
- File sets
- Binders

Please note: Your command list must contain the `reqrvw` keyword in order for you to initiate a review.

Initiate a review using the following procedure:

1. Select a view of the desired object type using one of the choices below.
   - View -> Vault -> Files or View -> Distributed -> Files
   - View -> Vault -> Parts or View -> Distributed -> Parts
   - View -> Vault -> Binders
   - View -> Vault -> Filesets
2. Select the item that you wish to have reviewed, for example a behavior specification file.
3. Initiate the review using Review -> Request. The Request a Review panel appears.
4. Enter the message you want to send to all review team members in the Request a Review panel.
5. Enter the review password if required.
6. Apply the panel to send the message and initiate the review.

Chapter 7

Setting Up and Modifying User-defined File Attributes

This chapter provides information and instructions for implementing user-defined file attributes.

Vault file attributes are built into the software. These include required entities such as revision code, classification, and status code, and also optional entities such as system type, file description, and part number. User-defined file attributes are defined by you. In this chapter, the term attributes refers only to user-defined attributes.

- Overview of Implementing User-defined Attributes
- Defining an Attribute
- Defining an Attribute Set
- Adding an Attribute to a Set
- Changing a Member of an Attribute Set
- Removing a Member from an Attribute Set
- Deleting an Attribute
- Deleting an Attribute Set
- Defining a Rule for When to Apply an Attribute Set
- Deleting a Rule
- Access and Security
- Unit of Measure

Overview of Implementing User-defined Attributes

You can use attributes to hold information that keeps you informed about your files. You can use IQF to view the attribute values for a given file or to view certain files/parts using their attributes as qualifiers. For example, you can list parts where the attribute name `clearance` has the value `Secret`.

What Is an Attribute?

An attribute is a piece of information related to a file or part. You can set up Vault so that when users execute the `store`, `replace`, or `update` commands, they include attribute values along with the information required for the file transfer. Vault maintains an association between an attribute and the file it relates to.

An attribute has a name, a type, and a value. You define the names and types of attributes. Users assign the values when they execute the `store`, `replace`, or `update` commands.

How Are Attributes Used?

For an attribute to be used with a file, that attribute must belong to a set of attributes. A set is a convenient way to group attributes. A set can contain single attributes along with other sets. You can create a set containing a hierarchy of sets as complex as you need it to be.

Having defined sets of attributes, you then define simple rules that determine when a particular set of attributes applies to a particular file. For example, you can set up a rule that instructs Vault to use the attribute set called `PROPERTIES` with all files belonging to the CARGO project.

When a user stores a new file in Vault, the user specifies the name of an attribute file. In the attribute file, the user specifies a value for each applicable attribute. Users can also include the attribute file when saving a file in Vault by updating or replacing it. In this way, attribute values can be changed.

Vault saves all attribute values whether or not the attribute file name is included when a Vault file is updated or replaced. Vault does this whether or not attributes are required. Vault prompts you for an attribute file name only when the data type or classification of an object has changed.

You should plan carefully before you implement attributes. You should be clear about what you are trying to accomplish with user-defined attributes before you define them and set up rules. When you delete a user-defined attribute, consider the implications of doing so.

Sequence of Steps for Implementing User-defined Attributes

Follow the steps below to implement user-defined attributes.

1. Define attribute names.
2. Define attribute set names.
3. Add members to attribute sets. A member can be an attribute or a set.
4. Define rules for when to apply a particular attribute set to a particular file.
5. Inform users of which attributes apply to which files. Be sure to tell them the attribute type for each attribute and what the defaults are. If you want to use a naming convention (not a requirement) for the name of the file that contains the attribute values, be sure to communicate that information as well.

After you have done this, users can create a file in which they specify values for the attributes. When they execute the `store`, `replace`, or `update` commands, they specify the name of the attribute file. If they do not include an attribute file when it is required to do so, or if the contents of the file are not correct, Vault does not perform the file transfer.

Brief Description of Commands for Implementing Attributes

Vault provides a set of commands that allow you to define attributes and sets, add attributes to sets, delete attributes, and create and delete rules for applying attribute sets. Execute these commands to implement user-defined attributes after you have set up your authority schemes, projects, and users.

The list below provides a brief description of the commands you use to implement user-defined attributes.

- `addattr` defines an attribute.
- `addaset` defines a set of attributes.
- `addmas` adds a member to a set of attributes.
- `chgmas` changes some parameters of a member of a set.
- `remmas` removes a member from an attribute set.

- `delattr` deletes an attribute.
- `delaset` deletes a set of attributes.
- `addrule` defines a rule for when to apply a set of attributes.
- `delrule` deletes a rule.

Defining an Attribute

Use the `addattr` command to define an attribute for use with files and parts stored in Vault.

Table 7-1 `addattr` **Command Parameters**

Keyword	Parameter Description	Default
attrname	Name of the attribute. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None
attrtype	Type of the attribute. Enter CHARACTER, NUMBER, or DATE, or as many as 24 characters, including only A-Z, 0-9, hyphen, and underscore, if you have defined your own attribute types.	None
attrdesc	Description of the attribute. Enter as many as 240 characters. When using the command-line format, if the field contains embedded blanks, enclose it in single quotation marks.	None

Using the Command-Line Format

Syntax:

`ciaddattr attrname=`attributeName `attrtype=`attributeType `attrdesc=`attributeDescription

Example:

```
ciaddattr
EDM> attrname=material attrtype=Character,
EDM> attrdesc='Material used to build the object.'
```

In this example, an attribute called Material is defined. The description has embedded blanks and so it is enclosed in single quotation marks.

Defining an Attribute Set

Use the `addaset` command to define the name of a set of attributes for use with files and parts stored in Vault. A set is a convenient way to group attributes. After you define the name of a set, you can use the `addmas` command to add previously defined attributes and sets to the set.

Table 7-2 `addaset` **Command Parameters**

Keyword	Parameter Description	Default
setname	Name of the attribute set. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None
setdesc	Description of the attribute set. Enter as many as 240 characters. When using the command-line format, and the field contains embedded blanks, enclose it in single quotation marks.	None

Using the Command-Line Format

Syntax:

`ciaddaset setname=`attributeSetName `setdesc=`SetDescription

Example:

```
ciaddaset
EDM> setname=Milestones
EDM> setdesc='Tasks that must be accomplished before release.'
```

In this example, an attribute set called Milestones is defined. The set description includes embedded blanks and is enclosed in single quotation marks.

Adding an Attribute to a Set

Use the `addmas` command to add a member to an attribute set. A member can be an attribute or an attribute set. You must define the attribute or attribute set before you can add it to an attribute set.

Table 7-3 `addmas` **Command Parameters**

Keyword	Parameter Description	Default
reqopt	Indicates whether applying the member is required or optional. Specify only for attributes being added. Vault discards this value if the member being added is a set. Enter **R** (Required) or **O** (Optional).	R
amemname	Name of a previously-defined attribute or attribute set. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None
amemtype	Type of the set member being added. Enter **A** for an Attribute or **S** for a Set.	A
setname	Name of the attribute set to which you are adding the attribute or attribute set. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None
defaval	Indicates the default value for the attribute being added. The default value must pass the rule defined for the attribute's data type. Vault uses exactly the data you enter, respecting upper and lowercase. Enter as many as 240 characters. Specify this value only when the member being added is an attribute. Vault discards this value if the member being added is a set. Specify NO-DEFAULT if you do not want a default value.	None

Building Attribute Sets

Vault associates an attribute with an object once for each distinct set the attribute belongs to. This allows you to build complex hierarchies of attribute sets within attribute sets. It does not matter if you add set A to set B and then add set B to set A. As Vault works through the sets, it passes by an attribute if it has already associated the object with that attribute. Since sets are merely a convenient way to group attributes, you can group them in any way that makes sense.

Using the Command-Line Format

Syntax:
`ciaddmas reqopt=`requiredOrOptional `amemname=`attributeOrSetName `amemtype=`memberType `setname=`SetName `defaval=`default

Example:

```
ciaddmas
EDM> amemname=Material amemtype=a,
EDM> setname=Properties reqopt=o Defaval=steel
```

In this example, an attribute called Material is added to the set called Properties.

Changing a Member of an Attribute Set

Use the `chgmas` command to change the required/optional status or the default value or attribute member of an attribute set. If you need to change the name of the attribute, you must delete the attribute, define a new one, and then add it to the attribute set.

Table 7-4 `chgmas` **Command Parameters**

Keyword	Parameter Description	Default
setname	Name of the attribute set that includes the member being changed. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None
amemname	Name of the attribute or attribute set you want to change. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None
amemtype	Type of set member being changed. Enter **A** for an Attribute or **S** for a Set.	A
reqopt	Indicates whether applying the member is required or optional. Specify only for attributes being changed. Vault discards this value if the member being changed is a set. Enter **R** (Required) or **O** (Optional).	R
defaval	Indicates the default value for the attribute being changed. The default value must pass the rule defined for the attribute's data type. Vault uses exactly the data you enter, respecting upper and lowercase. Enter as many as 240 characters. Specify this value only when the member being added is an attribute. Vault discards this value if the member being added is a set. Specify NO-DEFAULT if you do not want a default value.	None

Using the Command-Line Format

Syntax:

`cichgmas setname=`setName `amemname=`attributeOrSetName
`amemtype=`MemberType `reqopt=`RequiredOrOptional `defaval=`Default

Example:

```
cichgmas
EDM> amemname=Color amemtype=S,
EDM> setname=properties reqopt=O
```

In this example, an attribute set called Color is changed.

Removing a Member from an Attribute Set

Use the `remmas` command to remove a member from an attribute set. A member can be an attribute or a set name. When you remove a member, Vault no longer associates that member with the attribute set. The removed member continues to exist as an attribute or set on its own.

Table 7-5 `remmas` **Command Parameters**

Keyword	Parameter Description	Default
amemname	Name of the attribute or attribute set you want to remove. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None
amemtype	Type of set member being removed. Enter **A** for an Attribute or **S** for a Set.	A
setname	Name of the attribute set that includes the member being removed. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None

Using the Command-Line Format

Syntax:

`ciremmas amemname=`memberName `amemtype=`memberType `setname=`setName

Example:

```
ciremmas
EDM> amemname=Color amemtype=S,
EDM> setname=Properties
```

In this example, an attribute set called Color is removed from an attribute set called Properties.

Deleting an Attribute

Use the `delattr` command, with caution, to delete an attribute. After you execute the `delattr` command, Vault

1. Deletes the attribute
2. Removes the attribute from each set it belongs to
3. Deletes every occurrence of the attribute, including associated data

Please note: Be careful when you execute this command. It is recommended that you not delete attributes that have been used.

Table 7-6 `delattr` **Command Parameters**

Keyword	Parameter Description	Default
attrname	Name of the attribute you are deleting. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None

Using the Command-Line Format

Syntax:

```
cidelattr attrname=attributeName
```

Example:

```
cidelattr
EDM> attrname=Material
```

In this example, an attribute called Material is deleted.

Deleting an Attribute Set

Use the `delaset` to delete an attribute set. If the set is a member of another set, it is deleted from that set also. Any attributes or sets contained in the set that is deleted continue to exist but they are no longer associated in the set. Any rules that use the deleted set are also deleted.

Table 7-7 `delaset` **Command Parameters**

Keyword	Parameter Description	Default
SETNAME	Name of the attribute set you are deleting. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None

Using the Command-Line Format

Syntax:

`cidelaset setname=`setName

Example:

```
cidelaset
EDM> setname=Milestones
```

In this example, an attribute set called Milestones is deleted.

Defining a Rule for When to Apply an Attribute Set

Use the `addrule` command to define simple rules that Vault uses to determine when to use an attribute set. Use the Vault Rules Processor Language to define more complex rules or to define and modify additional simple rules.

Table 7-8 `addrule` **Command Parameters**

Keyword	Parameter Description	Default
rulename	Name of the rule you are defining. Enter as many as 24 characters—A-Z, 0-9, hyphen or underscore.	None
rulekey	Keyword that the rule operates on. Enter CLASS, PARTNUM, PROJID, STATUS, USER, or USERTYPE, or a user-defined keyword.	None
operator	Operator that the rule uses. Enteran equals sign (=) to indicate an exact match, angle brackets (<>) to indicate “not equal to”, or the keyword LIKE to indicate some matching characters. LIKE is a SQL wildcard operator.	=
testval	Exact value on which the rule uses the operator. This value must match what the keyword expects. Example: you can enter only PRO, PRI, or PUB when the command keyword is CLASS. Enter up to 240 letters, numbers, or special characters. When the operator is LIKE, you can include a percent sign (%) to represent one or more characters.	None
setname	Name of the attribute set that is applied when the rule is complied with. Enter as many as 24 characters including only A-Z, 0-9, hyphen, and underscore.	None

Using the Command-Line Format

Syntax:
`ciaddrule rulename=`ruleName `rulekey=`commandKeyword `operator=`operator `testval=`testValue `setname=`setName

Example:

```
ciaddrule
EDM> rulename=Release rulekey=Status,
EDM> operator='=' testval=Re setname=Color
```

In this example, the set called Color is applied when the status code is Re.

Deleting a Rule

Use the `delrule` command to delete rules that are used to determine when to apply attribute sets.

Table 7-9 `delrule` **Command Parameters**

Keyword	Parameter Description	Default
RULENAME	Name of the rule you are deleting. Enter as many as 24 characters— A-Z, 0-9, hyphen or underscore.	None

Using the Command-Line Format

Syntax:
`cidelrule rulename=`RuleName

Example:

```
cidelrule
EDM> rulename=Release
```

In this example, the rule named Release is deleted.

Access and Security

The Access and Security extension of User Defined attributes provides a method of associating an access authority level to individual user defined attributes. The access authority level operates identically to the same method which is used for assigning authority levels when adding a user.

In this case, the authority level of the attribute will prevail when a user attempts to access a file or part to which the attribute becomes associated. That is, if the user has insufficient access authority to an attribute, but has sufficient authority access to the attribute's associated file or part, then that attribute is not made available for that user. This feature applies to both storage and retrieval operations.

Unit of Measure

The Unit of Measure extension provides a method for associating a 'type', or 'unit' to an attribute's value. The unit of measure facility applies only to attributes defined as number, to the exclusion of date and/or character attributes. You can associate a unit of measure to preexisting number attributes, which allows you to migrate to this new feature without losing existing attributes data.

Any user-defined attribute, of the Number class, may be associated to a Unit of Measure. This is done by selecting an appropriate Unit of Measure Set. Three Unit of Measure sets are provided; length, temperature, and weight. Within each set you may then select an appropriate storage unit, e.g., feet for the length set. This will become the unit in which the attribute's value will be converted and stored into the database.

You may also select an appropriate presentation unit of measure. This may be the same as, or different from, the storage unit of measure, but must also be selected from the same Set. This unit will be used to determine the mode in which the attribute will be presented to the user, along with the necessary conversions, whenever the attribute is fetched from the database.

For each Unit of Measure Set, there are a suite of possible units from which to cho0se, representing both fps and mks measures systems.

Assigning the Unit of Measure

From the command line interface, you define the attribute and associate it with the unit of measure with two distinct commands: `addattr` and `chgattr`.

From the Graphical User Interface, follow this process:

1. From the menu bar, choose View > Admin > Attribute. The Attributes browser appears.
2. From the menu bar, choose File > New. The Define Attributes dialog box appears.
3. In the designated box, type an Attribute Name not yet associated with a Unit of Measure.
4. Click the Attribute Type Editor Button. The Attribute Types dialog box appears.
5. Choose an existing attribute type (Character, Date or Number).
6. Click OK.

a. If you enter number in the Attribute Types dialog box, further information displays at the bottom of this dialog box, under the heading Numeric Attribute Unit of Measure. In this instance, you can choose a domain from the Unit of Measure Set. The choices are:

   LENGTH

   TEMPERATURE

   WEIGHT

b. When you indicate the Unit of Measure Set, corresponding selections appear in the Storage Unit of Measure and Presentation Unit of Measure dialog boxes. Choose the unit of measure in which the Vault should store the converted value and the domain in which you intend to present the information.

c. In the Unit of Measure Precision box, indicate the number of decimal places to which you want the Vault to calculate and store the value.

7. Indicate the Authority Code of the attribute.

Please note: If you omit any specification of authority code, then the access and security profile of the attribute is identical to that of the file or part to which it will become associated.

You may, on the other hand, provide an alternative authority code. This would generally be done in order to make access to the attribute *more* restrictive, and, as stated earlier, the same rules which govern a user's access to files and parts, based upon the user's authority scheme, also govern that user's access to user-defined attributes.

8. Fill in the Description box on the Define Attribute dialog box.

9. Click OK.

Chapter 8

Deleting Files and Log Entries

This chapter provides information and instructions for deleting files, purging files, and deleting process log entries.

- Parameters for Deleting Files
- Deleting Files
- Parameters for Purging Files
- Purging Files
- Parameters for Deleting Process Log Entries
- Deleting Process Log Entries

Parameters for Deleting Files

To logically delete Vault files previously marked for deletion, use the `delete` command. (Mark files to be deleted with the `markd` command.)

Only files that are marked for deletion at the time the `delete` command is executed are deleted.

The deleted files will be physically removed from Vault after the `ibkup` (incremental backup) followed by the `delov` (delete old file versions) commands are executed. Vault does not physically delete files before they have been backed up. If the files were previously backed up, you can use the `delov` command.

Once deleted, you cannot access or recover a file with the `unmark` command. To recover files (one at a time), use the `recsf` (recover a single file) command.

Suppose you execute the `markd` command on a file followed by a `delete` command. This is the same as executing the `purge` command on that file, with one exception. A file marked for deletion can be unmarked (with the `unmark` command) to recover the file before you execute the `delete` command. A purged file can only be recovered using the `recsf` command.

Deleting File Set Members

Files that are marked to be deleted but are still members of a file set(s) cannot be deleted. These file set member names are listed on your screen. To prepare file set members for deletion

1. Remove them from the file set(s) with the `remmfs` command
2. Mark them to be deleted

You can also, mark all the files in a file set and then delete the file set.

Audit Information

Audit information created by the `delete` command is written in the `DELETE.EDMAUDIT` file.

Steps to Physically Remove a Vault File

You must perform the following steps to physically delete a file stored in Vault.

1. Remove the file from all file sets it is associated with (unless you are removing all the files in the file set).
2. Mark the file to be deleted with the `markd` command.
3. Logically delete the file with the `delete` command.
4. Perform an incremental backup with the `ibkup` command.
5. Physically delete the file with the `delov` command.

Table 8-1 `delete` **Command Parameters**

Keyword	Parameter Description	Default
output	Indicates where to send audit information. At the prompt enter: `S` (screen) or `F` (file) or `B` (both) or `M` (message only). When using the command-line format, you can only enter `F` or `M`.	S (Full- screen format) F (Command-line format)
append	Indicates whether to replace or append the audit file. If audit destination is F or B, enter: `R` (replace) or `A` (append). Ignored if audit destination is `S` or `M`.	R

Deleting Files

Execute the `delete` command to logically delete all files that have been marked for deletion with the `markd` command.

Using the Command-Line Format

```
cidelete
EDM> output=B append=A
```

In this example, the files previously marked for deletion with the `markd` command are deleted. Audit information will be copied to both the screen and the audit file. The new audit information will be appended to the existing audit file.

Parameters for Purging Files

To delete Vault files not previously marked for deletion (with the `markd` command), use the `purge` command.

This command is basically a one-step deletion procedure, in contrast to the two-step procedure of executing the `markd` (mark for deletion) command followed by the `delete` command.

When you execute the `purge` command, the file is physically erased from Vault the next time the `ibkup` (incremental backup) and `delov` (delete old file versions) commands are executed.

To `purge` a file, it

- Must be listed in the file directory
- Can be marked for delete with the `markd` command
- Cannot be a member of a file set
- Cannot be signed out
- Cannot be marked for archive with the `marka` command or archived
- Cannot be in review (`reqrvw` command executed for it)

Please note: Under normal circumstances, use the `markd` command to logically delete a file, followed by the `delete` command, instead of using the `purge` command. This allows the exact file to be recovered using the `unmark` command up until the `delete` command is executed.

PURGE Versus DELETE

Executing the `purge` command for a file is equivalent to executing the `markd` command on that file and then the `delete` command, except that with `markd` you can use the `unmark` command to recover a file marked for deletion.

You can only recover and access files deleted with the `purge` command one at a time with the `recsf` (recover a single file) command.

Audit Information

Audit information created by the `purge` command is written in the `purge.edmaudit` file.

Table 8-2 `purge` **Command Parameters**

Keyword	Parameter Description	Default
selscope	Indicates the entity you want purged. Enter: `F` (file) or `A` (all of a project) or `P` (part) or `U` (user's private files).	F
selname	Selection name. Enter 80 or fewer letters and/or numbers.	None
revision	Revision code. Enter 20 or fewer letters and/or numbers or leave blank for current revision.	Current revision
filepw	File password. If file(s) has password, enter 8 or fewer letters and/or numbers.	None
output	Indicates where to send audit information. At the prompt enter: `S` (screen) or `F` (file) or `B` (both) or `M` (message only). When using the command-line format, you can only enter `F` or `M`.	S (Full- screen format) F (Command-line format)
append	Indicates whether to replace or append the audit file. If audit destination is F or B, enter: `R` (replace) or `A` (append). Ignored if audit destination is S or M.	R

Purging Files

Execute the `purge` command to logically delete files that were not previously marked for deletion.

Using the Command-LIne Format

```
cipurge
EDM> selscope=P selname=cargo revision=Aa,
EDM> filepw=Projpw output=F append=A
```

In this example, the cargo part files are purged. The audit information is appended to the existing audit file.

Parameters for Deleting Process Log Entries

Delete entries from the Vault process log table with the `dellog` command. The process log records security violations, administrative actions, and system errors.

The `dellog` command deletes all the log entries dated on or after the From date and From time and before or on the To date and To time parameters.

The defaults for the fromdate and fromtime parameters are the earliest date and time of the entries in the log. The defaults for the todate and totime parameters are the current date and time. Thus, the default dates and times delete all of the entries.

Periodically use this command to delete old log entries and recover RDBMS storage space.

Table 8-3 `dellog` **Command Parameters**

Keyword	Parameter Description	Default
fromdate	USA or European standard date format. For USA format: month/day/year, for example, 05/01/93 for May 1, 1993. For European format: day.month.year, as in 01.05.93 for 1 May, 1993. Determines the starting date. Use the default value for the earliest log date entry.	Earliest date
fromtime	USA standard time format using a 24-hour clock. Enter the hour, followed by a colon (:), followed by the minute, followed by a colon (:), followed by the second (HH:MM:SS). For example, 09:05:22. Determines the starting time. Use the default value for the earliest log time entry. The range of allowable values is from 00:00:00 through 23:59:59.	Earliest time
todate	USA or European standard date format. For USA format: month/day/year, for example, 05/01/89 for May 1, 1989. For European format: day.month.year, as in 01.05.89 for 1 May, 1989. Determines the ending date. Use the default value for the current log date entry.	Current date
totime	USA standard time format using a 24-hour clock. Enter the hour, followed by a colon (:), followed by the minute, followed by a colon (:), followed by the second (HH:MM:SS). For example, 09:05:22. Determines the ending time. Use the default value for the current log time entry. The range of allowable values is from 00:00:00 through 23:59:59.	Current time

Deleting Process Log Entries

Execute the `dellog` command to delete Vault process log entries and recover RDBMS storage space.

Using the Command-LIne Format

```
cidellog
EDM> fromdate=01/01/97 fromtime=00:00:01,
EDM> todate=05/02/97 totime=23:59:59
```

In this example, all the log entries between January 1, 1997 and May 2, 1997 will be deleted.

Appendix A

Message Identifiers

This appendix lists Vault and IQF (Interactive Query Facility) command abbreviations. These abbreviations are primarily used as identifiers in messages.

- Vault Command Abbreviations
- IQF Command Abbreviations

Vault Command Abbreviations

This section lists command abbreviations.

Table A-1 List of Vault Command Abbreviations

Abbreviation	Command	Description
aag	addag	Add an authority group to Vault
aas	addaset	Add an attribute set to Vault
aat	addattr	Add an attribute to Vault
acl	addcl	Add a command list to Vault
adm	admcopy	Administrative copy
adp	addp	Add a project to Vault
ads	adds	Add a status level to an authority scheme
adt	addt	Add a user-defined table to the control of Vault
adu	addu	Add a user to Vault
afs	addfs	Add a file set
als	addusa	Add a user list/status code association
ama	addmas	Add a member to an attribute set
aml	addmul	Add members to a user list
ams	addmfs	Add a member to a file set
arc	archive	Move files marked for archiving from Vault to tape
arl	addrule	Add a classification rule to Vault
ars	addrs	Add a revision code sequence
asp	addsp	Add a storage pool to Vault
aul	addul	Add a user list name
aup	addup	Add a user to a project
cag	chgag	Change entries in an authority group
ccl	chgcl	Change entries in a command list
cfa	chgfa	Change file(s) attributes
cfc	chgfcl	Change file(s) classification
cfp	chgfpw	Change file(s) password
cfr	chgfrev	Change file(s) revision code
cfs	chgfsc	Change file(s) status code
clt	clst	Close a user-defined table
cma	chgmas	Change a member of an attribute set
cpa	chgp	Change the attributes of a project

Table A-1 List of Vault Command Abbreviations

Abbreviation	Command	Description
cps	chgsps	Change storage pool status
cpt	chgspt	Change storage pool type
cpw	chgupw	Change your Vault user password
cpy	copy	Copy file(s) to new Vault file(s)
csa	chgs	Change the attributes of a status level
ctl	chgctl	Change command trigger list
cua	chgu	Change a Vault user's attributes and/or authority
cup	chgup	Change a user's project authority
dag	delag	Delete an authority group from Vault
das	delaset	Delete an attribute set from Vault
dat	delattr	Delete an attribute from Vault
dcl	delcl	Delete a command list from Vault
del	delete	Delete files marked for deletion
dir	listdir	List file(s) on local or remote Vault node
dlg	dellog	Delete Vault audit log entries
dlp	delp	Delete a project from Vault
dls	dels	Delete a status level from an authority scheme
dlu	delu	Delete a user from Vault
dlv	delov	Delete old versions of files
drl	delrule	Delete a classification rule from Vault
drs	delrs	Delete a revision code sequence
dul	delul	Delete a user list
edm	edm	General Vault messages not related to any command
fsp	chgfsp	Change a file's storage pool
get	get	Sign out and copy file(s); modifications allowed
hlp	help	Vault help command
ibu	ibkup	Back up new and modified files
iqf	iqf	Interactive Query Facility
loa	load	Load file(s) from tape to Vault
mka	marka	Mark file(s) to be archived
mkd	markd	Mark file(s) to be deleted
mkr	markr	Mark file(s) to be restored
opt	opnt	Open a user-defined table

Table A-1 List of Vault Command Abbreviations

Abbreviation	Command	Description
pur	purge	Delete file(s) not previously marked for deletion
rea	read	Copy file(s); modifications not allowed
rep	replace	Save modified file(s); ability to modify ends
res	restore	Restore files from tape to Vault
rfs	remfs	Remove a file set
rls	remusa	Remove a user list/status code association
rma	remmas	Remove a member from an attribute set
rmg	readmsg	Read messages
rml	remmul	Remove a member from a user list
rms	remmfs	Remove a member from a file set
rmt	remt	Remove a user-defined table from Vault control
rsf	recsf	Recover a single file
rsp	recsp	Recover a storage pool
rst	reset	Cancel signing out of file(s); changes are lost
rsv	reserve	Define and sign out a Vault file name for future use
rup	remup	Remove a user from a project
rvp	rsvp	Respond to a review request
rvw	reqrvw	Request review of file(s)
scn	scantape	List the contents of a Vault tape
smg	sendmsg	Send a message
sof	signoff	Sign off and exit Vault session
son	signon	Sign on to Vault or another Vault User ID
sot	signout	Sign out file(s) to modify; file(s) not copied
str	store	Add a new file
ubu	ubkup	Back up entire Vault database
umk	unmark	Unmark previously marked files
unl	unload	Unload files to tape
upt	update	Save modified file(s); continue modifications

IQF Command Abbreviations

This section lists Interactive Query Facility command abbreviations.

Table A-2 List of IQF Command Abbreviations

Abbreviation	Command	Description
bak	backward	Scrolls a display towards the beginning
bot	bottom	Scrolls a display so the end is visible
brk	break	Inserts one or more empty lines into the display
cif	command interface	Calls the command interface
col	column	Scrolls a display left to the specified column
edt	edit	Accepts multiple IQF commands and then executes all
end	end	Ends display of a query result or help text
ext	exit	Terminates IQF
fil	file	Copies query result to a local file
fmt	format	Formats a display
fwd	forward	Scrolls a display towards the end
hlp	help	Displays on-line IQF help
lft	left	Scrolls the display to the left
lst	list	Displays the current query
prt	print	Sends the query result to a printer
rec	recall	Retrieves the last successful SELECT command
ret	retrieve	Retrieves the previous IQF command
run	run	Executes the current query
ryt	right	Scrolls the display to the right
sav	save	Copies the current query to a local file
sel	select	Specifies information to view
set	set	Assigns values to format parameters
sho	show	Displays the current format parameter settings
sta	start	Executes a current or stored query
tif	terminal interface	Calls the terminal interface
top	top	Scrolls a display towards the beginning
typ	type	Displays an IQF command file

Appendix B

Examples of Revision Code Sequences

This appendix provides examples of revision code sequences and shows how Vault uses them to assign revision codes to files.

- Data Used in Examples
- Storing Files Without Specifying a Revision Code
- Storing Files and Specifying a Revision Code
- Vault File Directory Examples
- Signing Out Files for Modification
- Changing File Classification
- Duplicate Revision Codes

Data Used in Examples

The revision code sequences used in the examples are:

- Sequence name: `none`

  This sequence name is a special, reserved name. The name indicates an empty revision code sequence. Vault does not assign a revision code when you store a file in a project that has a revision code sequence of none. When in-work files are ready to become released files, you must execute the `chgfrev` (Change file(s) revision code) command to assign revision codes to files belonging to a project with this revision code sequence.

- Sequence name: `default`

  This sequence is created by a Vault utility during Vault installation using a Company-supplied Vault sequence or a sequence you created. For the purposes of this discussion, the default sequence is the first 7 letters of the alphabet.

  `A, B, C, D, E, F, G`

- Sequence name: `revseqa`

  `A.A, B.B, C.C, D.D, E.E`

- Sequence name: `revseqb`

  `CODE01, CODE02, CODE03, CODE04, CODE05, CODE06, CODE07`

The table below shows four projects and their assigned revision code sequences. These assignments will be used in the examples.

Table B-1 Example of Projects and their assigned Revision Code Sequences

Project ID	Revision Sequence	Revision Sequence Description
PROJ01	default	Revision codes A through G
PROJ02	revseqa	Revision codes A.A through E.E
PROJ03	revseqb	Revision codes CODE01 through CODE07
PROJ04	none	No pre-established revision codes

Storing Files Without Specifying a Revision Code

When you store a file without specifying a revision code, Vault assigns the following revision codes to files stored under the following projects.

Table B-2 Vault Assigned Revision Codes

Project ID	Revision Code
PROJ01	A
PROJ02	A.A
PROJ03	CODE01
PROJ04	blank (none)

Note that because the revision code was not specified for the file belonging to `PROJ04`, and this project does not have a revision code sequence, no revision code is assigned. This also restricts the status under which you can store the file. You cannot store the file under a released status because a released file must have a revision code.

Storing Files and Specifying a Revision Code

When you store a file and specify a revision code, Vault assigns the following revision codes to files stored under the following projects:

Table B-3 User Specified Revision Code

Project ID	Revision Code
PROJ01	Revision code you entered (A, B, C, D, E, F, or G)
PROJ02	Revision code you entered (A.A, B.B, C.C, D.D, or E.E)
PROJ03	Revision code you entered (CODE01, CODE02, CODE03, CODE04, CODE05, CODE06, or CODE07)
PROJ04	Revision code you entered

Note that Vault verifies the entered revision code for PROJ01, PROJ02, and PROJ03 against the valid revision codes for each project. Vault does not verify the entered revision code for the file stored under PROJ04.

Vault File Directory Examples

The entries in the Vault File Directory shown in the table below will be used in later examples.

Table B-4 Vault File Directory Examples

File Name	Revision Code	Project ID	Status
VALVE	A	PROJ01	Released
VALVE	B	PROJ01	Released
VALVE	C	PROJ01	Released
PISTON	A.A	PROJ02	Released
PISTON	B.B	PROJ02	Released
PISTON	C.C	PROJ02	Released
CARBURETOR	CODE01	PROJ03	Released
CARBURETOR	CODE02	PROJ03	Released
CARBURETOR	CODE03	PROJ03	Released
CRANKSHAFT	17-AUG-98	PROJ04	Released
CRANKSHAFT	23-JUL-98,PROTOTYPE	PROJ04	Released
CRANKSHAFT	23-JUL-98,FINAL	PROJ04	Released

Signing Out Files for Modification

Using the information in the table in the previous section, Vault assigns the following revision codes to the files when you sign them out for modifications (execute the `get` command).

Table B-5 Revision Codes Assigned for Modified Files

Project ID	Revision Code
PROJ01	D
PROJ02	D.D
PROJ03	CODE04
PROJ04	Blank (none)

Within each project, the next revision code is selected from the currently associated revision code sequence assigned to the project. Vault does not assign a revision code to the file in `PROJ04` because this project has no pre-established revision code sequence. The following table shows the Vault File Directory with entries for the new files.

Table B-6 Vault File Directory

File Name	Revision Code	Project ID	Status
VALVE	A	PROJ01	Released
VALVE	B	PROJ01	Released
VALVE	C	PROJ01	Released
VALVE	D	PROJ01	In-work
PISTON	A.A	PROJ02	Released
PISTON	B.B	PROJ02	Released
PISTON	C.C	PROJ02	Released
PISTON	D.D	PROJ02	In-work
CARBURETOR	CODE01	PROJ03	Released
CARBURETOR	CODE02	PROJ03	Released
CARBURETOR	CODE03	PROJ03	Released
CARBURETOR	CODE04	PROJ03	In-work
CRANKSHAFT	17-AUG-98	PROJ04	Released
CRANKSHAFT	23-JUL-98,PROTOTYPE	PROJ04	Released
CRANKSHAFT	23-JUL-98,FINAL	PROJ04	Released
CRANKSHAFT	Blank (none)	PROJ04	In-work

You can change the status code of each of the in-work files to any status code in the set of status levels associated with the file's project. You can do so with the `chgfsc` (Change File(s) Status Code) command or as a result of the review process. You cannot, however, assign a released status code to the `CRANKSHAFT` file because it does not have a revision code. You must assign a revision code first. Then you can change the status to released.

Changing File Classification

When you change the classification of a file, it retains the revision code it had under its original classification. Vault assigns revision codes for the new classification only to new versions of the file.

Project to Project: Both Have Revision Code Sequences

When a file that belongs to one project is transferred to another project, it retains the revision code that it was assigned under the old (source) project. The revision code sequence of the new (target) project is used for future revisions of the file under the new project.

Vault does not change the revision codes for files created under one revision code sequence when those files are reclassified.

For example, all files in project `PROJ02` are reclassified as belonging to project `PROJ01`. The table below illustrates the entries in the original sample Vault File Directory.

Vault assigns revision code D to the next revision of file PISTON because this is the fourth revision code in the revision code sequence that is currently assigned to project PROJ01.

Table B-7 Examples of Vault-Assigned Revision Code

File Name	Revision Code	Project ID	Status
VALVE	A	PROJ01	Released
VALVE	B	PROJ01	Released
VALVE	C	PROJ01	Released
PISTON	A.A	PROJ01	Released
PISTON	B.B	PROJ01	Released
PISTON	C.C	PROJ01	Released
PISTON	D	PROJ01	In-work
CARBURETOR	CODE01	PROJ03	Released
CARBURETOR	CODE02	PROJ03	Released
CARBURETOR	CODE03	PROJ03	Released
CRANKSHAFT	17-AUG-98	PROJ04	Released
CRANKSHAFT	23-JUL-98,PROTOTYPE	PROJ04	Released
CRANKSHAFT	23-JUL-98,FINAL	PROJ04	Released

Project to Project: Not Having a Revision Code Sequence

You can reclassify files from a project having a revision code sequence to one having no revision code sequence and visa versa. Additionally, both the old (source) and new (target) projects could both be assigned the revision sequence NONE (no pre-established revision sequence).

For example, all files in project PROJ03 are reclassified as belonging to project PROJ04. The table below shows what the entries in the Vault File Directory would be.

Vault assigns a blank revision code to the next revision of file CARBURETOR because the project PROJ04 has no pre-established revision code sequence. Prior

to assigning a released status code, you must assign a revision code to the file with the CHGFREV command.

Table B-8 Revision Codes from Project to Project

File Name	Revision Code	Project ID	Status
VALVE	A	PROJ01	Released
VALVE	B	PROJ01	Released
VALVE	C	PROJ01	Released
PISTON	A.A	PROJ01	Released
PISTON	B.B	PROJ01	Released
PISTON	C.C	PROJ01	Released
PISTON	D	PROJ01	In-work
CARBURETOR	CODE01	PROJ04	Released
CARBURETOR	CODE02	PROJ04	Released
CARBURETOR	CODE03	PROJ04	Released
CARBURETOR	Blank (none)	PROJ04	In-work
CRANKSHAFT	17-AUG-98	PROJ04	Released
CRANKSHAFT	23-JUL-98,PROTOTYPE	PROJ04	Released
CRANKSHAFT	23-JUL-98,FINAL	PROJ04	Released

Duplicate Revision Codes

Since revision code sequences can have overlapping revision codes, it is important to be aware of the problems this can cause when transferring files among projects. For example, consider the following two revision code sequences and what happens when files are transferred and associated with new revision code sequences.

- Sequence name: `REVSEQX`

  `CODE01, CODE02, CODE03, CODE04`

- Sequence name: `REVSEQY`

  `CODEAA, CODE01, CODEBB, CODE02, CODECC, CODE03, CODEDD, CODE04`

A file in the Vault database belongs to a project that uses `REVSEQX`. The file is currently at its second revision and has a character revision code of `CODE02`.

You reclassify the file as belonging to a project that uses `REVSEQY`. The next revision code (its third) for this file will be `CODEBB`, as this is the third revision code in revision code sequence `REVSEQY`.

The file will not be able to acquire its fourth revision code, as the fourth revision code in revision code sequence `REVSEQY` is `CODE02`. This would cause a duplicate occurrence of the file at revision code `CODE02`. Vault returns an error until you change the fourth revision code in `REVSEQY`.

This situation can also occur using a revision code sequence of none. When it is time to assign a revision code to a file under this scheme, the revision code that is assigned to the file cannot be identical to any other revision code that any copy of the file has.

Appendix C

Corresponding Commands

This appendix contains a chart which shows Vault functionality accessible by using the command line and the Graphical User Interface.

You can use the chart to determine the path name for command-line functionality on the Graphical User Interface.

- Command Line and GUI Functionality

Please note: See the *Vault Command Reference* for elaboration on Vault commands.

Command Line and GUI Functionality

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
addag	View>Admin>Authority Groups	File>New
addaset	View>Admin>Attribute Sets	File>New
addattr	View>Admin>Attributes	File>New
addcl	View>Admin>Command Lists	File>New
addevals	View>Admin>Enumerations	Edit>Enumeration; Add
addfs	View>Vault>File Sets	File>New
addmas	View>Admin>Attribute Sets	Edit>Add Member
addmfs	View>Vault>File Sets	Edit>Members; Add
addmul	View>Admin>User Lists	Edit>Members; Add
addobj	View>Vault>Binders (View>Vault>Binders; File>Explode)	File>New
addp	View>Admin>Projects	File>New
addrel	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Edit>Binder; Add
addrs	View>Admin>Revision Sequences	File>New
addrule	View>Admin>Attribute Sets	Edit>Add Rule
adds	View>Admin>Projects	Edit>Status Levels; Add
addsub - Distributed File	View>Distributed>Files	Administration>Distribute On Event; Create
addsub - Distributed Part	View>Distributed>Parts	Administration>Distribute On Event; Create
addsub - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file)	Administration>Notify On Event; Create
addsub - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part)	Administration>Notify On Event; Create
addu	View>Admin>Users	File>New
addul	View>Admin>User Lists	File>New
addup	View>Admin>Projects	Edit>Users; Add
addusa	View>Admin>Projects	Edit>Status Levels; Exit Criteria; Create
addvlist	View>Admin>Vault Lists	File>New
addvlmem	View>Admin>Vault Lists	Edit>Members (Depends on Candidate and Member Lists)
admcopy - Authority Group	All views	Administration>Copy>Authority Group

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
admcopy - Command List	All views	Administration>Copy>Command List
admcopy - User List	All views	Administration>Copy>User List
associat - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>User Attributes
associat - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>User Attributes
chgag	View>Admin>Authority Groups	Edit>Authority Group
chgatrb	View>Admin>Type Definitions	Edit>Type Definition; “Attributes” Edit
chgatrbs - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Edit>Class Attributes
chgatrbs - Instance/Relationship	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Edit>Binder; Edit
chgattr - Numeric Attribute	View>Admin>Attributes	Edit>Unit of Measure
chgattr - All Attributes	View>Admin>Attributes	Edit>Protection
chgcl	View>Admin>Command Lists	Edit>Command List
chgctl	View>Admin>Command triggers	Edit>Command Trigger
chgdesc - Attribute Group	View>Admin>Type Definitions	Edit>Type Definition; “Attribute Groups” Edit Description
chgdesc - Binder	View>Vault>Binders	Edit>Description
chgdesc - enumeration	View>Admin>Enumerations	Edit>Description
chgdesc - /Instance/Relationship	View>Vault>Binders	Edit>Binder; Edit Description
chgdesc - Type Definition	View>Admin>Type Definitions	Edit>Description
chgfa - Catalog	View>Vault>Catalogs	Edit>File Attributes
chgfa - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>File Attributes
chgfa - File Set	View>Vault>File Sets	Edit>File Attributes
chgfa - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>File Attributes
chgfcl - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Edit>Classification
chgfcl - Catalog	View>Vault>Catalogs	Edit>Classification

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
`chgfcl` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>Classification
`chgfcl` - File Set	View>Vault>File Sets	Edit>Classification
`chgfcl` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>Classification
`chgfpw` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>Password
`chgfpw` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>Password
`chgfrev` - Binder	View>Vault>Binders (View>Vault>Binder; File>Explode; select a binder)	Edit>Revision
`chgfrev` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>Revision
`chgfrev` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>Revision
`chgfsc` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Edit>File Status Level
`chgfsc` - Catalog	View>Vault>Catalogs	Edit>File Status Level
`chgfsc` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>File Status Level
`CHGFSC` - File Set	View>Vault>File Sets	Edit>File Status Level
`chgfsc` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>File Status Level
`chgfsp`	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>Storage Pool
`chgmas`	View>Admin>Attribute Sets	Edit>Members
`chgp`	View>Admin>Projects	Edit>Project
`chgs`	View>Admin>Projects	Edit>Status Levels; Edit
`chgsps` - From Admi View	View>Admin>Storage Pools	Edit>Storage Pool Status
`chgsps` - From Files View	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>Storage Pool; “Storage Pool”...; Edit
`chgu`	View>Admin>Users	Edit>User
`chgup`	View>Admin>Projects	Edit>Users; Edit

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
chgupw	View>Admin>Users	Edit>Password
copy - Catalog	View>Vault>Catalogs	Vault>Duplicate
copy - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Vault>Duplicate
copy - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Vault>Duplicate
decl	View>Admin>Command Lists	Edit>Delete
defagrp	View>Admin>Type Definitions	Edit>Type Definition; “Attribute Groups” Create
defatrb	View>Admin>Type Definitions	Edit>Type Definition; “Attributes” Create
defenum	View>Admin>Enumerations	File>New
defotype	View>Admin>Type Definitions	File>New
delete	All views	Administration>Delete Marked Files
delevals	View>Admin>Enumerations	Edit>Enumeration; Delete
delenum	View>Admin>Enumerations	Edit>Delete
delag	View>Admin>Authority Groups	Edit>Delete
delagrp	View>Admin>Type Definitions	Edit>Type Definition; “Attribute Groups” Delete button
delaset	View>Admin>Attribute Sets	Edit>Delete
delatrb	View>Admin>Type Definitions	Edit>Type Definition; “Attributes” Delete
delattr	View>Admin>Attributes	Edit>Delete
dellog - All Before	View>Admin>Log Entries	Edit>Delete All Before
dellog - All Entries	View>Admin>Log Entries	Edit>Delete All Entries
dellog - All Since	View>Admin>Log Entries	Edit>Delete All Since
dellog - Within Range	View>Admin>Log Entries	Edit>Delete
delobj	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Edit>Delete
delotype	View>Admin>Type Definitions	Edit>Delete
delov	All views	Administration>Delete Old Versions
delp	View>Admin>Projects	Edit>Delete
delrel	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Edit>Binder; Delete
delrs	View>Admin>Revision Sequences	Edit>Delete
delrule	View>Admin>Attribute Sets	Edit>Delete Rule

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
`dels`	View>Admin>Projects	Edit>Status Levels; Remove
`DELSUB` - Distributed File	View>Distributed>Files	Administration>Distribute On Event; Delete
`delsub` - Distributed Part	View>Distributed>Parts	Administration>Distribute On Event; Delete
`delsub` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file)	Administration>Notify On Event; Delete
`delsub` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part)	Administration>Notify On Event; Delete
`delu`	View>Admin>Users	Edit>Delete
`delul`	View>Admin>User Lists	Edit>Delete
`export` - File	View>Distributed>Files	Administration>Distribute Now
`export` - Part	View>Distributed>Parts	Administration>Distribute Now
`get` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Vault>Check Out>Lock
`get` - Catalog	View>Vault>Catalogs	Vault>Check Out>Lock
`get` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Vault>Check Out>Lock
`get` - File Set	View>Vault>File Sets	Vault>Check Out>Lock
`get` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Vault>Check Out>Lock
`purge` - Catalog	View>Vault>Catalogs	Administration>Purge>Selected Object
`purge` -File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Administration>Purge>Selected Object
`purge` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Administration>Purge>Selected Object
`read` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Vault>Check Out>Do Not Lock
`read` - Catalog	View>Vault>Catalogs	Vault>Check Out>Do Not Lock
`read` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Vault>Check Out>Do Not Lock
`read` - File Set	View>Vault>File Sets	Vault>Check Out>Do Not Lock
`read - part`	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Vault>Check Out>Do Not Lock

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
`readattr` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>User Attributes
`readattr` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>User Attributes
`readmsg`	All views	Messages>Read
`register` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file)	Administration>Register
`register` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part)	Administration>Register
`remfs`	View>Vault>File Sets	Edit>Delete
`remmas`	View>Admin>Attribute Sets	Edit>Delete Member
`remmfs`	View>Vault>File Sets	Edit>Members; Remove
`remmul`	View>Admin>User Lists	Edit>Members; Remove
`remup`	View>Admin>Projects	Edit>Users; Remove
`remusa`	View>Admin>Projects	Edit>Status Levels; Exit Criteria; Remove
`remvlist`	View>Admin>Vault Lists	Edit>Delete
`remvlmem`	View>Admin>Vault Lists	Edit>Members (Depends on Candidate and Member Lists)
`replace` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Vault>Check In>Unlock
`replace` - Catalog	View>Vault>Catalogs	Vault>Check In>Unlock
`REPLACE` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Vault>Check In>Unlock
`replace` - File Set	View>Vault>File Sets	Vault>Check In>Unlock
`replace` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Vault>Check In>Unlock
`reqrvw` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Review>Request
`reqrvw` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Review>Request
`reqrvw` - File Set	View>Vault>File Sets	Review>Request
`reqrvw` - part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Review>Request
`reserve`	View>Local>Files	Vault>Reserve
`reset` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Vault>Unlock

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
`reset` - Catalog	View>Vault>Catalogs	Vault>Unlock
`reset` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Vault>Unlock
`reset` - File Set	View>Vault>File Sets	Vault>Unlock
`reset` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Vault>Unlock
`rsvp`- Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Review>Respond
`rsvp` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Review>Respond
`rsvp` - File Set	View>Vault>File Sets	Review>Respond
`rsvp` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Review>Respond
`sendmsg`	All views	Messages>Send
`signoff`	All views	File>sign Off
`signon`	All views	File>Sign On
`signout` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Vault>Check Out>Lock Only
`signout` - Catalog	View>Vault>Catalogs	Vault>Check Out>Lock Only
`signout` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Vault>Check Out>Lock Only
`signout` - File Set	View>Vault>File Sets	Vault>Check Out>Lock Only
`signout` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Vault>Check Out>Lock Only
`store` - Catalog	View>Local>Catalogs	Vault>Store
`store` -Directory	View>Local>Directories	Vault>Store
`store` - File	View>Local>Files	Vault>Store
`STORE` - Part	View>Local>Parts	Vault>Store
`unmark` - All Vault Files	All views	Administration>Unmark>All Vault Files
`unmark` - Selected Object	All views	Administration>Unmark>Selected Object
`update` - Binder	View>Vault>Binders (View>Vault>Binders; File>Explode; select a binder)	Vault>Check In>Keep Lock
`update` - Catalog	View>Vault>Catalogs	Vault>Check In>Keep Lock

Table C-1 Command Line and GUI Functionality

Command	View Menu Selection	Command Menu/Button Selection
`update` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Vault>Check In>Keep Lock
`update` - File Set	View>Vault>File Sets	Vault>Check In>Keep Lock
`update` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Vault>Check In>Keep Lock
`veruattr` - File	View>Vault>Files (View>Vault>Binders; File>Explode; select a file) (View>Distributed>Files)	Edit>User Attributes
`veruattr` - Part	View>Vault>Parts (View>Vault>Binders; File>Explode; select a part) (View>Distributed>Parts)	Edit>User Attributes

Appendix D

Customizing the Interface

Vault provides full customization of its graphical user interface to suit your work environment and application needs. This appendix provides guidelines for customization.

- Edmgui Resource File
- Preferences Menu
- Dialog Box Customization
- Font Customization

Edmgui Resource File

All interface settings are configured within a single resource file named `Edmgui`. You can modify this file to establish the Vault interface settings from two levels:

- User (local resource file)
- System administrator (system resource file)

To customize your interface, use the `Edmgui` resource file and the Preferences menu to make your modifications. You can change the interface in these ways:

- Reposition, add, delete, or rename menu bar items
- Change default values of text boxes
- Hide menu items and/or text boxes
- Change user-visible text

This resource file lists the default location, order, and name of menu items on the Vault menu. It also tells you which commands and keywords each menu item executes.

You also set CADDS-specific defaults in this file. If you turn off any CADDS defaults, these items do not appear on the interface.

User Settings

You can set personal interface settings for default values on the Preferences menu or the Design Rule Customization dialog box, shown later. Saved changes are written to the `$HOME/.Edmgui` file. The period (`.`) in `.Edmgui` distinguishes this file from the system resource file. Settings in `.Edmgui` take precedence over those in the system resource file, described in the next section, but not those in the override directory.

System Administrator Settings

The system administrator, or whoever manages the Vault at your site, can edit the interface resource file:

```
usr/apl/edm/data/app-defaults/C/Edmgui
```

Calling Other Programs from the Interface

With the Vault graphical user interface, you can call executables from the Custom menu for your own programs and software. By adding command lines to the `Edmgui` resource file, you can customize the interface.

For more information and examples on user-defined entries, check the heading Custom Entry List in `/edm/src/data/app-defaults/C/Edmgui`. The following sample script launches Vault Navigator from the Custom menu.

```
#!/bin/csh -f
#
# 15-Apr-95
#
# To autoload a local (local disk) ps file.
# NOTE: No command line options to perform autoload exist but it
# can be done thru environ variables. This is the reason for a
# wrapper script to launch navigator.
#
# The entry in the app-defaults/C/Edmgui resource file would look
# like the following:
# Edmgui.StartCN.textValue: Navigator...
# Edmgui.StartCN.source: launch_cn "@SELNAME"&
#
# Unless the user wants to block the Edmgui from continuing until
# the action defined in "source" is complete, the entry should
# always have an appended ampersand so the "system" call will
# return without waiting.
#
# CA_AUTO_LOAD = YES
# means autoload the ps file defined by CA_CONFIGURATION
#              = NO
# means don't automatic load any files
#
# setenv CA_AUTO_LOAD YES
#
# CA_CONFIGURATION is the selname of the file to open. @SELNAME
# argument of entry definition.
#
# setenv CA_CONFIGURATION $argv[1]
#
# CA_LOCATION = 1 means file is on the local disk
#             = 2 means file is in the vault
setenv CA_LOCATION 1
/usr/apl/edm/scripts/configuration_access
exit(0)
```

Preferences Menu

The Preferences menu lets you change interface defaults. These local settings and defaults apply only to the appearance and behavior of your own Vault user interface. Examples include colors, text fonts, and printer command settings. From the Preferences menu you can also replace the default query generated for each object type with your own queries.

To change the Vault interface for your personal use, follow these steps.

1. Choose Current Settings from the Preferences menu. (Your entries in the `Edmgui` resource file are displayed.) The Preferences dialog box appears.
2. Edit the text boxes for your desired settings and defaults.
3. Click OK.
4. Choose Preferences > Save Preferences to save your settings to the `.Edmgui` file. (See User Settings under Edmgui Resource File for further information.)
5. Exit `Edmgui` and restart in order for the values to be effective.

The Preferences menu items are as follows:

- Current Settings —Opens a dialog box that displays the current interface settings and defaults as currently defined in the `Edmgui` resource file.

Please note: To restore original settings, click the Reset button. OK or Apply implements changes only for the current session.

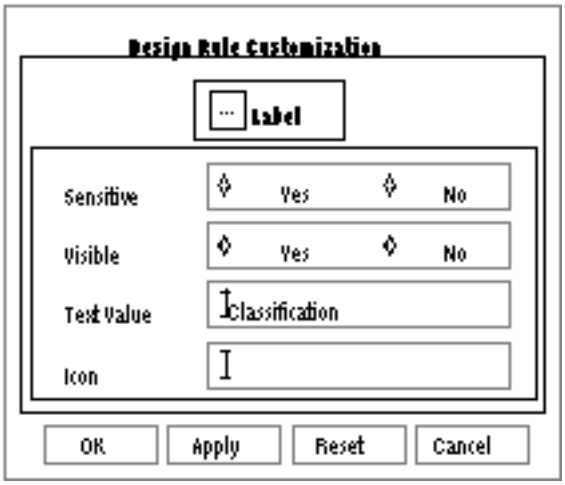
- Save Preferences — Permanently stores interface changes to your local version of the `Edmgui` file in `$HOME/.Edmgui`.
- Show Assist Line (and other Show items not shown)—Turns the display for the various items, such as the Assist line, on or off. A selected box indicates that the item is turned on, or active.

Other items from the Preference menu that you can choose to display include Show Status Line, Show Prompt Line, Show Logo Icon, Show Object Status, and Show Tool Palette.

Dialog Box Customization

The Design Rule Customization box allows you to change the text displayed in the dialog boxes. To change the text in a dialog box, place your cursor on the item you want to change. Press the Shift key and the right mouse button. The Design Rule Customization dialog box appears. Change the text to your preference. If you choose No for Sensitive, the item will be unavailable.

Figure D-1 Design Rule Customization Dialog Box



These changes are valid while you are in that session. To have them always valid, choose Preferences > Save Preferences to save your settings to the `.Edmgui` file. (See Edmgui Resource File > User Settingsfor further information.)

Font Customization

To customize the font, edit the font entries in your `.Edmgui` file or choose the font text boxes in the Preferences menu.

The syntax to specify a particular font is:

```
<fontname>, <size>, <style>
```

Where:

<fontname> is any valid font name available in the X-Server.

<size> is any valid size available for a particular fontname.

<style> can be bold, italic, bold-italic, medium (default). If style name doesn't match with any one of these, it defaults to medium.

In xls format these get mapped to:

```
bold : bold-r
italic : medium-o
bold-italic : bold-o
medium : medium-o
```

These get converted to a standard format specification for font as given below.

```
"-*-<fontname>-<style>-normal-*-*-<size>0-*-*-*-*-*-*"
```

Please note the addition of 0 to the size.

For example, in your `.Edmgui` resource file you specify a font for text field as follows:

```
Edmgui.textFieldFont:courier,14,bold
```

This gets mapped to:

```
"-*-courier-bold-r-normal-*-*-140-*-*-*-*-*-*"
```

Glossary

This glossary contains terms for end users, managers who oversee Vault operations, and system administrators.

action

A function to perform on behalf of a subscription. Examples are send message, export, register. Used by the Distributed Vault product.

Action Manager

The Vault service that processes actions requested by the Event Manager and by the Register function. Part of the Distributed Vault product.

Administrative server (DN)

The Vault service that oversees administrative functions that do not impact files or parts, such as adding a user or creating a file set. See *Vault server*.

aggregation

A dependent relationship between a parent binder and the files and binders in that binder. In an aggregation relationship, a change to one member of the collection of items in the binder affects the other members. You use aggregation when you want constraints and dependencies between members. All aggregated members have the same life cycle as their parent binder. A binder member may be associated to multiple binders but can only be aggregated to one binder. See *association*.

Application Entity (AE)

A Vault server process, such as the Administrative Server, Attribute Manager, Data Distribution Server, Data Management Facility, and Log Services Manager. Abbreviated as *AE*. See *Vault server*.

archived file

A file that has been moved to tape storage and is no longer available in the Vault storage pool. The metadata associated with archived files enables them to be restored for active use.

assembly

A group of parts, and their interelationships, that comprise a partial or complete product.

association

An independent relationship between a parent binder and the files and binders in that parent binder. All members can have the different life cycles. A binder member may be associated to more than one binder. You can use the association relationship to manage a group of objects or a large project with pieces managed by different groups. See *aggregation*.

attribute

Information about a Vault item, that is, a file or part (group of files). A Vault item has built-in attributes such as the file's physical location, classification, revision code, status level, and version. The system administrator can also define custom attributes. Attribute data for a file is stored in the RDBMS, whereas the file itself is stored in a storage pool.

attribute group

A collection of binder/relation attributes. Attribute groups are associated to binders and relations by a typedef.

Attribute Manager

The part of the Vault server that maintains the user-defined attribute data on parts and files.

attribute set

A collection of user-defined file and part attributes. Attribute sets are associated with files and parts through rules.

audit file

A local file that stores results generated by certain Vault commands. For example, when you mark files or read files from disk or tape that information is recorded in the audit file.

audit log

The record of all transactions for a Vault, both successful and failed.

authority group

A named set of authority numbers. Groups are created by a Vault administrator.

authority number

A two-digit integer (00-99) that controls access to files by users as part of an authority scheme. A user can access files with numbers less than or equal to their authority number. Authority numbers are assigned by an administrator (public files) or by a Project Administrator (project files). A given file is assigned an authority number through its status level.

authority scheme

A set of authority levels for each status level assigned to users. An authority scheme determines who can access a particular file at a particular point in its design cycle. Each user has a public authority scheme for public files and a project authority scheme for each project. A project may use a job type for an authority scheme.

binder

A user-defined collection of Vault files, parts, or other binders. Binders, unlike file sets, have ownership, revision, status level, and may have user-defined attributes. After you create the binder, you add objects by relating them to the binder. Binder members can belong to more than one binder. See also *relationship*.

catalog

A logical group of CADDS files, parts, and other CADDS catalogs. Within a local file system, a catalog is contained in a set of nested file directories. Once stored in Vault, catalog members are related by a common name prefix.

check-in process

This operation returns file(s) to the Vault. Like an update, it creates a new version of the file, if the file has been modified. Unlike an Update, checking in a file in to Vault unlocks the file for use by other Vault users. The Vault command name for check-in is `replace`.

check-out process

The retrieval of file(s) from the Vault for modification. Checking a file out of the Vault locks the file from modifications by other users. The vault command name for check-out is `get`.

classification

Classification is a required, built-in attribute of a Vault file or part. Each file and part in the Vault database is classified as public, private, or project.

client

The local application that requests a service from the Vault server.

command list

A set of Vault commands with a command list name. Each Vault user is assigned one command list per authority scheme.

command trigger

A user-written application program for customizing the behavior of a Vault command. A command trigger can call an external process before or after execution of the command.

Data Distribution Server (DDS)

The Vault service that oversees transfers of files between the server and clients. Abbreviated as *DDS*. See Vault server.

Data Management Facility (DMF)

Part of the Data Distribution Service that determines if a transfer of a file may be performed. Abbreviated as *DMF*. See Vault server.

design cycle

The sequence of phases that a file or part passes through until the revision is complete (released). In Vault, a phase is known as a status level. In the Vault, phases are grouped withina status level partitions. The public authority scheme and each project has its own design cycle.

directory

A group of unrelated files in the local file system. All files in a local directory can be stored (checked into the vault) as a group, though no association remains among them after they are stored.

distributed parts and files

Parts and files that have been copied to Vaults other than the one in which they were created. Possible states in the new vault include: replicated for read, replicated for write. Used by the Distributed Vault product.

Distributed Object Directory (DOD)

A repository that contains Vault locations of all registered files and parts. Part of the Distributed Vault product. A Vault in a distributed Vault environment consults the DOD to locate objects that you request.

Distributed Vault

Distributed Vault (DV) provides for a transparent and cooperative use of Vaults in a distributed environment. This arrangement allows you to access registered files and parts on any of the configured Vaults. You do not need to know the locations of these items in the Distributed Vault.

domain

A name which can be used to define all of the related client and server processes which constitute a single instance of an Optegra Vault environment. All Optegra Vault client and server processes execute within the context of a single domain. DOMAIN is synonymous with VAULT ID.

enumerated list

An exhaustive set of values to allow for a binder/relation attribute. Example: Color_list: Red, Green, Blue.

event

A data base change such as object status change, object revision change, object metadata change. The subscription command enables events to be monitored so that when they occur, actions can be performed automatically.

Event Manager

The Vault service that monitors events that have been logged by event triggers. Part of the Distributed Vault product.

event trigger

An Oracle procedure that watches for events that have subscriptions and logs them to an event queue database table. Part of the Distributed Vault product.

export services

Export services replicate registered files and parts from one vault (the exporter) to other vaults (the importers) in the Distributed Vault environment.

file

The smallest unit of named data. In the Vault, a file has associated metadata, or attributes, such as its owner, history and status. These attributes protect and manage the file. Files may be grouped into parts, file sets, or binders.

file set

A user-defined group of Vault files, parts, or other file sets. Members can belong to more than one file set. Unlike a part, a file set does not have attributes. Some Vault commands allow you to operate on all members of a file set at one time.

host

A host is a computer. It is represented by its name, called the host name. When configured with Optegra Vault, a host can be referred to as a Vault host.

import services

Import services replicate registered files and parts from an exporter Vault to importing Vaults in the Distributed Vault environment.

in-work

The designation for a file or part that is currently available for modification. Only the highest revision of a part or file may be inwork.

Internet Protocol (IP)

Part of TCP/IP. One part of the protocol is a computer addressing scheme. IP addresses use 32-bits to identify a node. They are usually represented in dotted decimal notation (for example, 102.52.94.97) with one part being your network ID and the other your system ID.

Interactive Query Facility (IQF)

A querying capability that enables you to extract and output information from the Vault RDBMS about files, projects, authority schemes, and so forth.

job type

A set of privileges in a project authority scheme created by a project administrator. It consists of a command list and an authority group. Each project has one or more job types. Each user assigned to a project has a single job type on that project.

life cycle

The set of all design cycles.

Log Services Manager

The Vault process that receives messages from each command and stores them optionally into the database audit log table.

mailing list

All the Vault user IDs on a user list. Users on the list are notified of a review and are informed of the results. Vault uses the user list to send action messages or notification messages for reviews. User lists are created by a Vault Project Administrator.

message

A Vault-maintained electronic message, usually associated with reviews, but also associated with the outcome of subscriptions.

metadata

Information about a Vault file such as its classification, location, revision, status level, version, and so forth. This data is stored in the RDBMS, whereas the data files themselves are stored in a storage pool.

network

A connection among two or more computers that enables them to directly transfer information. The Vault uses the TCP/IP network protocol.

Network Services Manager (NSM)

NSM performs system and network management functions for nodes that access the Vault. The Process manager is an example of an application under the control of NSM. See *Vault server*.

node

In NSM terms, a node represents the name of a computer upon which Vault client OR server processes are executed. Generally, a node is identical to a host, although the term 'host' also has connotations relative to internetwork addressing, whereas a node name has no such relationship.

notification message

A message sent by the Vault to inform you of the start of a review cycle.

Oracle

An relational database management system marketed by Oracle Corporation. Vault applications use Oracle7 for basic database management functions.

owner

The user or project to which a file or part belongs. According to the classification scheme, public files have no owner, whereas private files are owned by a Vault user and project files are owned by a project. Private files are accessible only by the user that owns them. Project files are accessible by users with authority on that project.

part

A group of related files from within an application, for example, CADDS. Part members are controlled together and usually share the same name prefix. A file can belong to only one part.

Each part has a classification, revision, status, and owner (public, project, or private) that it shares with its constituent files. Part types are defined with the environment parameter (GUI term is local rule) when a file is first stored in the Vault.

placeholder file

A reserved Vault file name. This name holds a place for a future file. The reserved file name contains only metadata and no corresponding data file.

Private (PRI)

The classification assigned to files and parts that are owned by a single user ID.

Process Manager

The program responsible for managing Vault process resources. It:

- Coordinates the Vault's startup and shutdown sequence

- Validates all incoming client AND server service requests.

- Allocates servers to clients, when requested, and starts additional servers, in response to these requests, if necessary and possible.

- Monitors all running client and server processes.

- The Process Manager program generally resides on the same machine as the Vault's server processes, although this is not a requirement.

Product structure

The diagram that represents all the relationships between the parts and subassemblies within an assembly.

Project (PRO)

The classification assigned to files and parts owned by a project. Access to project files is controlled by the project’s authority scheme.

Public (PUB)

The file classification assigned to Vault files and parts that are accessible to every user depending on their authority within the public authority scheme.

restore

To return a file/part to active use from archived status.

RDBMS

Relational Database Management System. Oracle and SQL/DS are RDBMS products used with Vault applications.

read authority

A level of permissions giving you the ability to read but not modify a file. The Vault option for reading copies a Vault file to your computer for viewing. This operation does not allow you to modify the file.

registration

A process by which you register files to the Distributed Object Directory for access on Distributed Vaults. A registered item can be accessed by users on other vaults in the Distributed Vault environment.

relationship

The connector between a parent binder, or container, and the members within it: files, parts, or other binders. Relationships can be an aggregation with a dependency on other items or an association independent of other items. See *association* and *aggregation*.

released

A designation for a file or part that is no longer being modified. Such objects are in a released status level partition. When you request to modify a released part or file, a new revision of the object is created at the initial in-work status level.

review list

A user list whose members are asked to review a portion of work.

revision sequence

A set of revision codes predefined for use in projects. Newly created files and parts are assigned the first revision code in their project's revision sequence. When a released object is checked out, a new object with the next revision code in the sequence is created.

Some projects do not have assigned revision sequences. The objects in such projects must be manually assigned a revision code before they are released.

revision code

An alphanumeric designation which, together with a name, uniquely identifies a part or file. A file or part may exist in the Vault under more than one revision code, but only the latest revision can be in-work. All other revisions are released. Parts and files in projects without an assigned revision sequence are created with a blank revision code which must be changed to a non-blank value before the object is released.

rulebase

A set of rules to define and handle a group of files as a single part within an application environment such as CADDS. User-defined rulebases are implemented within the Vault as an application or executable file.

status level

An alphanumeric code identifying the current phase of work on the Vault file or part. Status levels are a subdivision of the status partition levels: Pre-inwork, In-work, or Released. Each status code is associated to an authority number. The Vault compares the authority number of the file's status code with the authority number of a user to determine whether or not a given user can access a given file.

status level partition

A set of three design cycle phases to control whether you can modify a file: Dormant (optional), In-work (I), and Released (R). Status partitions are further divided into status levels. The In-work and Released partitions must have at least one assigned status level.

storage pool

The Vault term for the location of user files under its control. The Vault server requires at least two and preferably seven or more storage pools for each Vault installation.

Structured Query Language (SQL)

An ANSI standard for retrieving and manipulating data stored in a relational database management system.

subscription

A user-defined association between an object (part/file/project), an event on that object, and an action to take as a result of the event. Available with the Distributed Vault product.

typedef

A user-defined subclassification of the general category of objects known as binders. Before user-defined binders can be created, a typedef must be defined to create them under.

unit of measure

An optional attribute of a numeric part or file attribute. Unit of measure specifies the type of quantity represented by numbers, for example, 10 kg, 20 ft.

User-defined Table

An Oracle database table created outside of the Optegra vault that can be placed under vault access and security control.

user ID

An account set up for a Vault user.

user list

A list of Vault user IDs for use with a particular file or project. Vault uses the user list to notify users of a review and inform them of the results. User lists are created by the Vault Project Administrator.

tablespace

Logical storage units within the Vault for the relational database. The usable data, such as tables and indexes, is stored in tablespaces. Tablespaces are physically stored in one or more data files.

TCP/IP

Transmission Control Protocol/Internet Protocol. An Ethernet-based local area network communication standard.

Vault host

The the name of the computer on which the server processes of a Vault DOMAIN run. Ordinarily, the name of the DOMAIN and the name of the Vault host are the same, except that the DOMAIN name is the upper-cased value of the vault host.

Vault list

A Vault list is a list of DOMAINs, that is, a list of Vault IDs. Vault lists are used, by Distributed Vault, for registering and exporting objects. A Vault list determines which Vault IDs (DOMAINS) are to receive (Import) items from an exporting vault.

Vault server

Synonymous with Vault host.

version

A Vault object is uniquely identified by its name and revision. As files are checked in and out of the Vault, multiple copies of the same name and revision are created in the Vault. These multiple copies are distinguished internally by their version numbers. In general, only the latest version (version 0) of an object is accessible to users, but old versions are manipulated by Vault administrative tasks such as backup and recovery. The Vault uses a time stamp to identify old versions.

write authority

A level of permissions giving you the ability to modify a file.

Index

A

B

C

D

E

F

I

L

M

N

O

P

R

S

T

U

V

W