



## ***Security Features User's Guide***

***Revision 21.0***

***DOC10130-1LA***

# Security Features User's Guide

First Edition

Steve Calapai

Updated for Revision 22.0

by

Jerry Ornstein

This guide documents the software operation of the Prime Computer and its supporting systems and utilities as implemented at Master Disk Revision Level 22.0 (Rev. 22.0).

Prime Computer, Inc.  
Prime Park  
Natick, Massachusetts 01760

The information in this document is subject to change without notice and should not be construed as a commitment by Prime Computer, Inc. Prime Computer, Inc., assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

Copyright © 1988 by Prime Computer, Inc. All rights reserved.

PRIME, PRIME, PRIMOS, and the PRIME logo are registered trademarks of Prime Computer, Inc. DISCOVER, INFO/BASIC, INFORM, MIDAS, MIDASPLUS, PERFORM, Prime INFORMATION, PRIME/SNA, PRIMELINK, PRIMENET, PRIMEWAY, PRIMIX, PRISAM, PST 100, PT25, PT45, PT65, PT200, PW153, PW200, PW250, RINGNET, SIMPLE, 50 Series, 400, 750, 850, 2250, 2350, 2450, 2550, 2655, 2755, 4050, 4150, 4450, 6350, 6550, 9650, 9655, 9750, 9755, 9950, 9955, and 9955II are trademarks of Prime Computer, Inc.

#### PRINTING HISTORY

First Edition (DOC10130-11A) July 1987 for Revision 21.0  
Update 1 (UPD10130-11A) October 1988 for Revision 22.0

#### CREDITS

Editorial: Roberta King, Kathe Rhoades  
Illustration: Anna Spoerri  
Document Preparation: Kathy Normington  
Production: Paula Brown, Judy Gordon

## HOW TO ORDER TECHNICAL DOCUMENTS

To order copies of documents, or to obtain a catalog and price list:

### United States Customers

Call Prime Telemarketing,  
toll free, at 1-800-343-2533,  
Monday through Friday,  
8:30 a.m. to 5:00 p.m. (EST).

### International

Contact your local Prime  
subsidiary or distributor.

## CUSTOMER SUPPORT

Prime provides the following toll-free numbers for customers in the United States needing service:

1-800-322-2838 (within Massachusetts)	1-800-541-8888 (within Alaska)
1-800-343-2320 (within other states)	1-800-541-8888 (within Hawaii)

For other locations, contact your Prime representative.

## SURVEYS AND CORRESPONDENCE

Please comment on this manual using the Reader Response Form provided in the back of this book. Address any additional comments on this or other Prime documents to:

Technical Publications Department  
Prime Computer, Inc.  
500 Old Connecticut Path  
Framingham, MA 01701

# Contents

ABOUT THIS BOOK	vii
1 SECURITY FEATURES OVERVIEW	1-1
Types of Users	1-1
Types of System Security	1-4
Login Security	1-6
Data Security	1-10
Security Monitor	1-11
2 LOGIN SECURITY	2-1
Accessing the System	2-1
Changing Your Login Password	2-5
Completing a Work Session	2-8
Accessing Remote Systems	2-9
3 FILE SYSTEM SECURITY	3-1
What Are Access Control Lists (ACLs)?	3-2
Specific ACLs and Access Categories	3-6
Default Protection	3-10
Priority ACLs and Device ACLs	3-15
Who May Distribute Rights	3-15
Access Requirements for Essential PRIMOS Commands	3-17
4 USING ACLs	4-1
Commands for Using ACLs and Access Categories	4-1
Listing Access Rights	4-2
Controlling Access to Files and Directories	4-4
Setting Access Rights to Match the Rights on Other Objects	4-8
Reverting to Default Protection	4-9
Changing Access Rights	4-9
Tips on Setting Access Rights	4-11

## APPENDICES

A	SECURITY RELATED MESSAGES	A-1
B	GLOSSARY	B-1
	INDEX	X-1

# About This Book

The Security Features User's Guide provides information for new and experienced users about security features of the PRIMOS® operating system and Prime computers.

The purpose of this book is to discuss the security features available to you as a user of a Prime system and to explain the security mechanisms that your System Administrator may use to enforce security on your system.

## WHAT THIS BOOK CONTAINS

The Security Features User's Guide contains information users need to know about security on a Prime system. It is intended for those users who regularly access Prime systems that meet the requirements of the C2 level of system security as specified by the Department of Defense.

## ORGANIZATION OF THIS BOOK

The Security Features User's Guide consists of four chapters and two appendices.

- Chapter 1, SECURITY FEATURES OVERVIEW, presents an overview of security features available on a secure Prime system.
- Chapter 2, LOGIN SECURITY, tells you how to access either the local system or a remote system.

- Chapter 3, FILE SYSTEM SECURITY, explains Access Control Lists (ACLs), Prime's mechanism for protecting file system data from unauthorized users. The chapter contains a discussion of the access requirements for basic PRIMOS commands.
- Chapter 4, USING ACLs, shows how to use access control lists if allowed to by your System Administrator in order to protect your personal work from unauthorized users.
- Appendix A, SECURITY RELATED MESSAGES, lists and explains system messages related to security.
- Appendix B, GLOSSARY, provides a list of basic PRIMOS terms.

#### SECURITY FEATURES INTRODUCED AT REV. 22.0

Three new security features are listed below. They are introduced in Chapter 1 and explained in detail in Chapter 2.

- Count of failed logins
- Computer-generated passwords
- Forced expiration of passwords

#### OTHER USEFUL BOOKS FOR PRIME USERS

Other useful books for Prime users include

- PRIMOS User's Guide (DOC4130-51A) which introduces the new user to PRIMOS and to the Prime file system, compilers, and system facilities. The presentation is largely tutorial and centers on that small portion of the software that does most of the typical user's work. References to other Prime documents tell you where to find detailed information.
- New User's Guide to EDITOR and RUNOFF (FDR3104-101B) which explains how to use Prime's line-oriented text editor (ED) and text formatter (RUNOFF). This book has two updates: COR3104-001 and COR3104-002.
- PRIMOS Commands Reference Guide (DOC3108-71A) which contains information on the format and usage of all PRIMOS user commands.
- CPL User's Guide (DOC4302-31A) which explains the full usage of the Prime Command Procedure Language (CPL).



- Programmer's Guide to BIND and EPFs (DOC8691-11A) and its update package (UPD8691-11A) which explain how to use Prime's new linking utility, BIND, and how to program most effectively with Executable Program Formats (EPFs), the dynamic runfiles created by BIND.
- Subroutines Reference II: File System (DOC10081-11A) which explains subroutines that allow you to set, modify, list, and remove protection on file system objects from programs. This book has two updates: UPD10081-11A and UPD10081-12A.
- The reference guide(s) for the programming language(s) you will use on your Prime system.

For books on special topics, consult the Guide to Prime User Documents (DOC6138-6PA). This guide helps you make the best use of Prime technical documentation. It lists all books available through Rev. 21.0. Each listing includes a description of the book, its printing history, and its intended audience.

An online file, accessible by typing `HELP DOCUMENTS`, provides an up-to-date, cumulative list of manuals, updates, and programmer's companions.

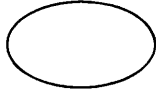
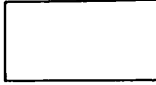
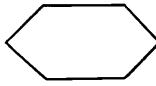

#### PRIME DOCUMENTATION CONVENTIONS

The following conventions are used in command formats, statement formats, and in examples throughout this document. Examples illustrate the uses of these commands and statements in typical applications.

<u>Convention</u>	<u>Explanation</u>	<u>Example</u>
UPPERCASE	In command formats, words in uppercase indicate the names of commands, options, statements, and keywords. Enter them in either uppercase or lowercase.	SLIST
lowercase	In command formats, words in lowercase indicate variables for which you must substitute a suitable value.	LOGIN user-id
Abbreviations in format statements	If an uppercase word in a command format has an abbreviation, either the abbreviation is underscored or the name and abbreviation are placed within braces.	LOGOUT { SET_QUOTA } SQ }

<u>Convention</u>	<u>Explanation</u>	<u>Example</u>
Brackets [ ]	Brackets enclose a list of one or more optional items. Choose none, one, or more of these items.	LD [ -BRIEF -SIZE ]
Braces { }	Braces enclose a list of items. Choose one and only one of these items.	CLOSE { filename ALL }
Braces within brackets [{ }]	Braces within brackets enclose a list of items. Choose either none or only one of these items; do not choose more than one.	BIND [ { pathname options } ]
Hyphen -	Wherever a hyphen appears as the first character of an option, it is a required part of that option.	SPOOL -LIST
<u>Underscore in examples</u>	In examples, user input is underscored but system prompts and output are not.	OK, <u>RESUME MY_PROG</u> This is the output of MY_PROG.CPL OK,
Angle brackets in messages < >	In messages, a word or words enclosed within angle brackets indicates a variable for which the program substitutes the appropriate value.	Disk <diskname>

#### ADDITIONAL CONVENTIONS USED IN THIS BOOK

<u>Convention</u>	<u>Explanation</u>	
Oval	Ovals are used in diagrams to represent files.	
Rectangle	Rectangles are used in diagrams to represent directories.	
Hexagon	Hexagons are used in diagrams to represent segment directories.	
Triangle	Triangles are used in diagrams to represent access categories.	

## COMMAND-LINE FORMAT

The PRIMOS command line has the following basic format:

```
command [argument] ... [option [argument]]...
```

A command is a PRIMOS-recognized word that signals a certain action. An option is a PRIMOS-recognized word that begins with a hyphen and further defines the action the command should take. An argument is a variable that is used with either a command or an option. An argument may set a value for an option, or it may specify an object on which the command or option acts.

One or more spaces must separate each command-line element (command, argument, or option) from the next.

Thus, if you use the command-line format

```
LOGIN [user-id] [-ON systemname]
```

you could create the following command:

```
LOGIN TERRY -ON SYSX
```

In this example

LOGIN	is the command
TERRY	is an argument for the command LOGIN
-ON	is an option for the command LOGIN
SYSX	is an argument for the option -ON

# 1

## Security Features Overview

A secure computer system requires a coherent security policy that is designed to prevent unauthorized entry to and use of computers. Access to information on a secure system is controlled on a need-to-know basis by the System Administrator or by other individuals with administrative responsibility.

The purpose of this chapter is to introduce you to the security features on secure Prime systems, and to describe briefly the security features that are available to you as a PRIMOS user. Specific topics include

- Types of users on Prime systems
- Types of system security
- Security auditing

### TYPES OF USERS

For purposes of security, users are categorized according to the type of system access they require.

A Prime system distinguishes among the following types of users:

- System Administrator
- Privileged users
- PRIMOS users
- Unauthorized users

In a secure system environment, each type of user has a well-defined role and clearly defined access privileges to the system. The following pages discuss these roles and privileges in more detail.

### System Administrator

The System Administrator is responsible for setting up, configuring, and maintaining the Prime system. As part of this responsibility, the administrator creates and maintains the User Profile Database, which identifies each user who is to have access to the system and the attributes individual users have once they access the system. Such attributes include membership in ACL groups and a definition of the user's command environment. The values for each of these attributes are set by the System Administrator.

Although the administrator may delegate some responsibility to a system operator or to a Project Administrator, the System Administrator is ultimately responsible for defining and enforcing system security.

The System Administrator is the one user of the system who requires access to all areas of the system. The administrator may override user-defined protection, may assign (but not decode) user login passwords, and may audit all system activity.

### Privileged Users

Privileged users are users whose jobs require them to have unlimited access to certain areas of the system, usually on a temporary basis. Privileged users include system operators, Project Administrators, subsystem administrators (such as a Batch Administrator) and Prime Customer Service Representatives.

System operators require special access rights that enable them to shut down disk partitions to perform backups, run system utilities, boot the system, install new software, archive data, and monitor system activity. Operators may also be requested by the System Administrator to maintain the User Profile Database.

Project Administrators require unlimited access to certain sections of the User Profile Database and to certain sections of the PRIMOS file system that relate to their projects. However, unlike the System Administrator, they do not have unlimited access to the entire system.

Subsystem administrators, such as a Batch Administrator, have rights and security responsibilities to a single subsystem.

Prime Customer Service Representatives typically require unlimited access to the entire system to diagnose hardware and software problems. Access is generally granted on a temporary basis.

### PRIMOS Users

PRIMOS users are those who use the software available on a Prime system. This category includes programmers, users of database management products, and others who need access to the system.

The type of system access a user has depends upon how the System Administrator defines the user in the User Profile Database. Typically, user profiles vary from user to user, and in some cases the System Administrator grants special privileges to a PRIMOS user. For example, some users may be allowed to start up spooler processes, whereas other users are not granted that privilege.

On a secure Prime system, users must use a password in order to gain access to the system. For maximum security, the System Administrator may also require users to enter their password on a separate line. With this method, the password does not echo (appear) on the screen.

### Unauthorized Users

A computer system must be secure from those who would attempt to gain access to the system for purposes of fraud or malicious destruction of data. These unauthorized users may either be employees of the same company or may be others using Public Data Networks to attempt to gain access to the system.

Authorized PRIMOS users can track unauthorized attempts to log in under a given user ID. When a user logs in successfully, PRIMOS displays a record of all unsuccessful login attempts under that user ID since the last successful login. This information is available only to the individual user at the successful login. The count of failed logins between successful logins is automatically reset to zero once the count has been displayed. Chapter 2 contains a more detailed discussion of this feature.

## TYPES OF SYSTEM SECURITY

There are two categories of system security: hardware security and software security. The next sections describe these categories.

### Hardware Security

Hardware security consists of security within the building or office as well as security for the computer equipment and storage media. As a user, you may be responsible for ensuring the security of your office, terminals, printers, or other computer-related equipment and media. Security issues that must be considered in any secure environment include

- Controlling access to your office or any room containing computer equipment or media
- Ensuring that sensitive documents, tapes, or disks are protected from being removed from the area
- Ensuring that data that is no longer needed is properly erased from computer media such as tapes

Security for the equipment (which includes terminals, printers, and modems) involves keeping track of its use outside the computer room and office. To ensure security for the equipment, use the following guidelines:

- Keep an up-to-date inventory of all equipment. Each item entry should include a brief description, serial number, and current location. You may want to keep a separate inventory of tapes, disks, and other forms of media.
- Label all portable equipment by using indelible ink or by engraving it.
- Set up procedures to control the movement of equipment. This is especially important if equipment is sometimes taken out of the building or off-site. Someone in authority should always know where any piece of equipment is at any given time.

### Software Security

Software security consists of security against illegal access to the system itself and security against illegal access to data after login.

Maintaining software security requires your System Administrator to do the following:

- Control access to the system itself, so that unauthorized users cannot log in and use the system (called login security, or system access)
- Control a user's access to files, directories, and data on assignable devices after login (called data security, data access, or file system security).

The PRIMOS operating system allows the System Administrator to implement login security through a User Profile Database and to implement data security through Access Control Lists (ACLs).

Some of the features of the PRIMOS ACL security system are as follows:

- Default protection can be supplied, both for the system and for individual files and directories.
- Default is closed (that is, no access).
- Access rights are set on a user-by-user basis. Thus, every user has a set of specially tailored access rights. No two users need have the same rights.
- Access to the system is controlled by a single person, the System Administrator.
- Access to data can be controlled by a single person, either by the owner (creator) or by an external administrator.
- After access controls are set on a file or directory, no password or other transferable information is required for a guest user to use the data.
- For secure systems, a password is a requirement for login. This password can be different for every user.
- Login passwords are recorded in the machine in an encrypted (scrambled) form, so that they cannot be read by humans or be easily decrypted.
- If required, passwords can still be set on directories on a system, but they are not allowed on systems that adhere to the C2 security criteria.

The following sections provide more detailed information on login and data security.



Note

In some previous versions of PRIMOS, data security was handled by passwording. Although it provided a measure of security, the password system for data security as implemented had some drawbacks. The Edit Profile utility and ACL system were developed to eliminate these drawbacks. Data security by passwording is not allowed on a system that adheres to the C2 security criteria.

LOGIN SECURITY

Login security in PRIMOS is provided by the User Profile system. The User Profile system consists of a database that your System Administrator builds and maintains.

The User Profile Database includes an entry for every authorized user. Each user entry is a set of tables mapping the user ID, the login password, and the project (or projects) to which this user ID is allowed access. When you, as a user, attempt to log in, PRIMOS consults this database and determines if you should be allowed access to the system. If you are allowed access, PRIMOS then checks this database for the system and project access rights your System Administrator has set for you.

User IDs

Your user ID identifies you to PRIMOS. Your login password is used by PRIMOS to confirm your user ID. Both your user ID and login passwords must be registered in the User Profile Database before you can attempt to access PRIMOS.

A system is more secure if every user has a unique user ID. However, your System Administrator may decide that a group of people may use the same user ID. People in such a group share the same system restrictions and privileges. Allowing several people to share one user ID decreases security, but this may be offset by the simplicity of providing an identical operating environment for many people in a single operation.

For optimum security, your System Administrator may require that your user ID not be your given name or initials. IDs that are given names or initials are less secure than IDs that are not as obviously associated with a specific user.

### Login Passwords

At login, you must supply a login password. In addition, your System Administrator may have established a minimum length for login passwords. For example, the System Administrator may require that your password be at least six characters long. If this is the case at your site, your password will always be at least six characters long. If you change your password, you must specify a password that is at least six characters in length.

A password can be entered in one of two ways:

- By typing "LOGIN user-id password". Typing the password on the same line as the LOGIN command means the password is echoed (that is, appears) on the screen.
- By typing only "LOGIN user-id". The password is omitted from the login line. In this case, PRIMOS prompts for the password. The password is not echoed, and is thus not displayed on the screen. In a secure system, this method is preferable to the echoing of passwords.

On a secure Prime system, your System Administrator may require you to enter your password at the "Password?" prompt only. This non-echoing method of entering passwords is more secure because another user cannot discover the password by looking at the screen while you're logging in.

### Note

If you log in at a half-duplex terminal, passwords are echoed, regardless of how the System Administrator sets up the system.

### Changing Passwords

You increase the security of the system environment when you change your password. Changing your password frequently but at randomly timed intervals makes it difficult for unauthorized personnel to break into the system.

There are three circumstances in which you would change your password.

- You may change your password at any time by using the `CHANGE_PASSWORD` command.
- You must change your password when it expires. When your password expires, PRIMOS prompts you for a new password when you attempt to log in.
- Your password should be changed the very first time you log in. If computer-generated passwords have been enabled, then PRIMOS gives you a new password the very first time you log in.

The `CHANGE_PASSWORD` command allows you to change your login password at any time. Typically, your System Administrator will encourage you to change your password immediately after you log in for the first time. Changing the password at first login ensures that only one person (the person who performed the change) knows the password.

If your System Administrator has enabled computer-generated passwords, then PRIMOS generates a new password for you the very first time you log in. The new password replaces the initial password issued to you by the administrator. All subsequent passwords issued to you are computer generated.

Passwords are stored by the system in an encrypted form. They cannot be called out and read by anyone, including the System Administrator. A password should not be written down or told to anyone. If it is, the security provided by password encryption is lost.

If you forget your password, the System Administrator cannot find out what your password is. The only remedy is for the System Administrator to assign you a new password.

The `CHANGE_PASSWORD` command, computer-generated passwords, and password expiration are documented in Chapter 2, LOGIN SECURITY.

### Project IDs

Every user must be registered in the User Profile Database as a member of at least one project. During a terminal session, you must be associated with a project ID. At login, the project ID may be supplied by you or by the PRIMOS internal login program.

The System Administrator may provide you with a system default project, particularly if you have no true project affiliation. If your system uses projects to provide special operating environments, you may be required to specify those project IDs at login. If your system does not use specific projects, all users become members of the system default project.

User IDs With Multiple Projects: A user ID may be a member of several projects, each of which gives a different set of access rights and restrictions. The maximum number of projects with which a user ID can be associated is the number of projects on the system. A system can have a maximum of 4096 projects, which means that potentially you could be a member of 4096 projects.

When you are a member of more than one project, you can specify a particular project on the command line by using the -PROJECT option of the LOGIN command. For example, suppose that user JOE is associated with two projects, ALPHA and OMEGA. The command line

```
LOGIN JOE -PROJECT ALPHA
```

logs in user JOE to the project ALPHA; the command line

```
LOGIN JOE -PROJECT OMEGA
```

logs in the same user to project OMEGA.

If you are a member of several projects and do not specify a project ID at login, you are assigned to the default project indicated in your user profile. If your user profile does not contain a default project, you are prompted for a project ID. Requiring that users provide project IDs at login adds another level of security to the system.

### Degrees of Login Security

As the preceding discussion suggests, your System Administrator may require a user ID, a long password, and a project ID to identify you to the system. These requirements provide the most security on a Prime system. If your system is considered a secure one, you probably will be required to meet these requirements before you can access the system. Chapter 2, LOGIN SECURITY, discusses the PRIMOS login procedure in detail.

External Login Programs: In addition to the PRIMOS login procedure, your System Administrator may require you to enter additional login information through an external login program. External login programs are created and implemented at your site by your System Administrator or other person at your site who is in charge of system security. These programs add another level of security to the login procedure by requiring you to enter additional passwords or other verification information before you are allowed to access the system. Check with your System Administrator to see if your site uses an external login program.

### Network Security (Remote Logins)

Many Prime installations contain two or more systems connected in a network. A network is a combination of communications hardware and PRIMOS software. Prime's network is called PRIMENET™.

To enhance security throughout a network, the Administrators for each system in the network will typically require that you obtain a remote ID on the system in the network to which you need access. Remote IDs are discussed in Chapter 2.

### DATA SECURITY

Data security is the control of a user's access to data after login. PRIMOS provides three methods to protect the information contained in files, directories, and on storage media:

- Access Control Lists (ACLs) for directories and files
- Priority ACLs for disk partitions
- Device ACLs for data on assignable devices

#### Access Control Lists (ACLs)

Access Control Lists (ACLs) are the cornerstone of the file system access control mechanism. If your System Administrator grants you Protect (P) access on your login directory, you can protect your files and directories using ACLs.

ACLs can be set on a directory or a file. In addition, the System Administrator may also set ACLs for assignable peripheral devices. When an ACL is set on a directory, all file system objects contained in the directory are given the same protection by default. This default protection can be overridden by setting a specific ACL on a lower level file or directory with the SET\_ACCESS command.

An ACL can provide access control both for individual users and for user groups. Both types of control can be combined in one ACL. An ACL can also use a special identifier (\$REST) to control access rights for all users who do not appear in the ACL by name or as group members.

If you are included in an ACL both as a member of an ACL group and as a named user, your rights as a named user override your group rights. Thus, you receive only those rights assigned by your user ID. You or your Administrator can use this control to either increase or decrease your rights or the rights of other named users.

If you are a member of more than one ACL group and the groups are in an ACL, you receive the sum (logical union) of all access rights for those groups.

By using the proper combination of ACLs, you can also prevent the unauthorized copying of licensed programs. The X (Execute) access right prevents local EPFs from being copied or read with a standard file system utility, but allows them to be executed.

Chapter 3, FILE SYSTEM SECURITY, and Chapter 4, USING ACLs, provide detailed information on what ACLs are and how to use them.

### Priority ACLs

Your System Administrator can set priority ACLs to govern access to any disk partition on the system. Priority ACLs override all other data security mechanisms in PRIMOS. They are generally intended for temporary use, such as when a system operator performs a backup.

### Device ACLs

On a Prime system, ACLs are extended to protect data not only in files and directories but also on assignable peripheral devices, such as magnetic tape drives and printers.

As a user, you will either have access rights to a device or not. If you attempt to assign a magnetic tape drive, for example, and receive a message stating that you have insufficient access rights, it is because your System Administrator has not specifically added you to the ACL for that device.

### SECURITY MONITOR

PRIMOS provides the System Administrator with an online auditing tool, called the Security Monitor, to keep track of who is logged in and which parts of the system they are accessing.

The monitor allows the System Administrator to record specified system events and to store that information on external storage media. The System Administrator can thus enforce accountability on users, detect break-in attempts, and detect and trace the history of security violations.

# 2

## Login Security

This chapter discusses the security involved when accessing the local or a remote system. Topics include

- Accessing the system
- Changing your login password
- Completing a work session
- Accessing remote systems

### ACCESSING THE SYSTEM

#### Obtaining a User ID

Before you can access the system, you must obtain a user ID and login password from your System Administrator. A user ID is a name that can have a maximum of 32 characters. The first character must be a letter. The remaining characters may be letters, digits, periods (.), underscores (\_), or dollar signs (\$). A login password is a string of as many as 16 characters. It may contain any ASCII character (listed in the Prime User's Guide) except CR, CONTROL-P, CONTROL-S, CONTROL-Q, and PRIMOS reserved characters (also listed in the Prime User's Guide).

## Logging In

Once you have your user ID and password, you can log in to the system. Logging in identifies you to the system and establishes the initial contact between you and the system. When you finish logging in, you are attached to your origin directory and your access rights on the system have been established.

To log in to the system, you first type the command

LOGIN

The system then prompts you for your user ID:

User id?

You must type in your user ID. The system now requests your password:

Password?

You must type in your login password. For security reasons, the password does not appear on the screen as you type it.

Some systems organize users into specific groups called projects. If your system uses the project structure, PRIMOS requests a project ID:

Project ID?

You must type in your project ID. If you are prompted for a project ID and you do not have a project ID, see your System Administrator.

On a secure Prime system, you must always provide the system with your user ID and password; whether you supply a project ID depends upon your installation. Once you have successfully provided all of the information requested, the login procedure is completed, and PRIMOS responds as in the following example:

OK, LOGIN

User id? FRED

Password? NIX (Password entered will not appear on the terminal)

Project id? RESEARCH

FRED (user 13) logged in Friday, 19 Jun 87 11:23:28.

Welcome to PRIMOS version 21.0.

Last login Thursday, 18 Jun 87 09:49:44.



The word NIX in this example is the login password. RESEARCH is the project ID. The number in parentheses (user 13) is the user number assigned by PRIMOS. The time is expressed in 24-hour format (hh:mm:ss).

If you misspell your user ID or password, you receive the message

Invalid user ID or password; please try again.

If you enter a project ID incorrectly, you receive the message

Invalid project ID; please try again.

If you repeat the login process without errors and still have trouble, ask your System Administrator for help. If the problem is that the system itself is fully loaded, a message such as "maximum number of users exceeded" may be displayed. In this case, try to log in again later, when some other user may have logged out.

Alternative Form of the LOGIN Command: You may also enter arguments and options on the same line as the LOGIN command (although for security reasons your System Administrator may disallow passwords on the login line). The format is

LOGIN [user-id [login-password]] [-PROJECT project-id]

The arguments and options are explained below.

<u>Argument/Option</u>	<u>Description</u>
user-id	Your user identification name.
login-password	Your login password. The password may be included on the login line if your system allows.
-PROJECT project-id	project-id is the name that associates you with a particular project.

## SECURITY FEATURES USER'S GUIDE

If you type your login password on the command line, you must first type your user ID there. For example,

OK, LOGIN FRED NIXOLOG -PROJECT RESEARCH

FRED (user 13) logged in Wednesday, 17 Jun 87 12:27:21.  
Welcome to PRIMOS version 21.0.  
Last login Wednesday, 17 Jun 87 11:23:28.  
OK,

### Counts of Failed Logins

PRIMOS counts the number of failed attempts to log in under your user ID. When you log in successfully, PRIMOS informs you of any unsuccessful attempts to log in that have occurred since you last logged in. Any number of unsuccessful attempts would indicate that unauthorized personnel may be attempting to break into the system. You should alert your System Administrator to what may be a very serious problem. Remember, the warning is displayed on your terminal only when you log in. When you log in the display is as follows:

OK, LOGIN FRED  
Password? (Password does not echo.)

FRED (user 13) logged in Wednesday, 17 Feb 88 10:42:46  
Welcome to PRIMOS version 22.0  
Copyright (C) 1988; Prime Computer, Inc.  
Last login Wednesday, 16 Feb 88 10:25:35

Warning! There were 3 failed attempts to login under this id since  
the last successful login.  
OK,

The counter is automatically set to zero at this time. If there have been no failed attempts to log in under your ID, the warning is not displayed.

### The Origin Directory

Once you have successfully logged in, PRIMOS connects you to a directory set by the System Administrator as your Initial Attach Point (IAP). This directory is your origin directory and is now also your current directory.

In order to use PRIMOS commands fully, your user ID needs to be included in an Access Control List (ACL) that grants you access rights

for file system objects in your origin directory. Full access rights for directories are Owner, Protect, Delete, Add, List, and Use (OPDALU). Full access rights for files and segment directories are Owner, Read, Write, and Execute (ORWX). You may or may not have full rights, depending upon how your user ID has been defined. If you are attached to your origin directory and are unable to execute commands, you may not have the necessary rights. See your System Administrator. ACLs are explained fully in Chapters 3 and 4.

#### CHANGING YOUR LOGIN PASSWORD

Once you have logged in, you may change your current login password with the `CHANGE_PASSWORD` (CPW) command. A login password may contain a maximum of 16 characters. A minimum character length for passwords may be set by your System Administrator. If this is the case, your login password must be at least as long as the minimum length. A login password may contain any ASCII character except PRIMOS reserved characters (listed in the PRIMOS User's Guide). The format for changing an existing login password is

```
{ CHANGE_PASSWORD } old-login-password
{ CPW }
```

or

```
CHANGE_PASSWORD -PROMPT
```

If you type your old login password on the command line, it will appear on the screen. To avoid having your old password appear on the screen, use the `-PROMPT` option. When you use the `-PROMPT` option instead of typing the old password on the command line, the system prompts you for your old password. For example,

```
OK, CPW -PROMPT
Old password? OLD LOGIN PASSWORD (Password does not echo.)
```

This procedure allows you to enter your old password without having it appear on the screen.

No matter which way you enter your old password, the system now requests the new login password with the prompt

```
New password?
```

## SECURITY FEATURES USER'S GUIDE

As usual, the new password does not appear on the terminal as you type it. The system requests the new login password a second time, for verification, with the prompt

Reenter new password for confirmation:

PRIMOS prints an appropriate error message if the old password is incorrect, if the two new passwords entered do not match, or if the format of the new password is invalid. In any of these cases, the old password is not changed.

The example below illustrates CHANGE\_PASSWORD:

```
OK, CPW NIXOLOG  
New password? STIXOLOG (Password does not echo.)  
Reenter new password for confirmation: STIXOLOG (Not echoed.)
```

### Password Expiration

Your System Administrator has the option of causing your login password to expire after a predetermined length of time. If your System Administrator uses this option, you are notified by PRIMOS when you log in that your current password has expired and that you must change it in order to access the system.

If the password expiration feature is in effect, the first time you log in after the expiration you see this display:

```
OK, LOGIN FRED  
Password: NIXOLOG (Password does not echo.)  
  
Your password has expired; please change it.  
New password: SIXODEN (Password does not echo.)  
Reenter new password for confirmation: SIXODEN (Not echoed.)  
Your new password has been confirmed.  
OK,
```

You have two opportunities to enter a new password. If you make an error in entering your new password a second time, the login is aborted.

### Computer-generated Passwords

Your System Administrator also has the option of having PRIMOS generate a password for you. If this feature is in effect, the first time you log in, PRIMOS determines whether you have been issued a computer-generated password. If not, PRIMOS generates one for you. For example,

OK, LOGIN FRED

Password: NIXOLOG (Password entered does not echo.)

Computer generated passwords are in effect.

Please ensure that you can view your new password in privacy.

Type RETURN to continue: <CR>

Your new password is HIXAKE

Reenter new password for confirmation: HIXAKE (Not echoed.)

Your new password has been confirmed.

OK,

If the computer-generated password feature is in effect when you use the CHANGE\_PASSWORD command, PRIMOS generates the new password:

OK, CPW NIXALOG

Computer generated passwords are in effect.

Please ensure that you can view your new password in privacy.

Type RETURN to continue: <CR>

Your new password is CADOMON.

Reenter new password for confirmation: CADOMON (Not echoed.)

Your new password has been confirmed.

OK,

### Password Expiration and Computer-Generated Passwords

If both the password expiration and computer-generated password features are in effect, PRIMOS generates a password for you when your current password has expired. An example follows.

OK, LOGIN FRED  
 Password: NIXOLOG (Password does not echo.)  
 Your password has expired.  
 Computer generated passwords are in effect.  
 Please ensure that you can view your new password in privacy.  
 Type RETURN to continue: <CR>  
  
 Your new password is CAWAVER  
 Reenter new password for confirmation: CAWAVER (Not echoed.)  
 Your new password has been confirmed.  
 OK,

### COMPLETING A WORK SESSION

When you finish a session at the terminal, give the LOGOUT command to terminate your access to PRIMOS. The format is

#### LOGOUT

PRIMOS acknowledges the LOGOUT command with the following message:

user-id (user number) logged out week day, date time.  
 Time used: xxh xxm connect, xxm xxs CPU, xxm xxs I/O  
 OK,

The various parts of this display have the following meanings:

<u>Term</u>	<u>Description</u>
number	The user number assigned at LOGIN
time	Time of logout, expressed in 24-hour format (hh:mm:ss)
xxh xxm connect	The amount of elapsed clock time between login and logout, in hours and minutes
xxm xxs CPU	Central Processing Unit time used, in minutes and seconds
xxm xxs I/O	The amount of input/output time used, in minutes and seconds

For example,

OK, LO

FRED (user 13) logged out Wednesday, 17 Jun 87 13:52:20.  
Time used: 01h 22m connect, 01m 14s CPU, 03m 35s I/O.  
OK,

It is good practice to log out after every session. Logging out closes all files and releases the PRIMOS process to another user. Logging out also protects your files and directories against unauthorized access by someone who comes across your unattended terminal while you are still logged in.

However, if you forget to log out, the system automatically logs out your terminal process after a time delay. This delay is set by the System Administrator. The default is 1000 minutes (16 hours, 40 minutes), but most System Administrators lower this value.

#### ACCESSING REMOTE SYSTEMS

Many Prime installations contain two or more systems connected in a network. A network is a combination of communications hardware and PRIMOS software. Prime's network is called PRIMENET. The PRIMENET facility is described in detail in the User's Guide to Prime Network Services.

In a network, the system to which your terminal is connected is called the local system, and all other systems are called remote systems.

Occasionally, in a Prime network, a user ID on any one system is typically recognized by the other (remote) system or systems. This recognition allows you to log in on the local system and give PRIMOS commands, such as ATTACH and COPY, that use a remote directory, just as if the remote directory were on the local system.

However, in a secure network environment, the System Administrator on a remote system may not allow that system to recognize the ID you normally use. If the remote system does not recognize your ID, you cannot access directories on that system. In this case, you must take the following steps in order to access the remote system:

1. Ask the System Administrator on the remote system to assign you a user ID on that system.
2. Log in to the local system.
3. Use the ADD\_REMOTE\_ID (ARID) command to add your remote ID to the remote system.

Once you have followed these three steps, any command you give that accesses the remote system (for example, a command that reads a file stored on the remote system) is carried out using your remote ID rather than your local ID.

The remote ID is valid for the duration of your login session or until you remove it with the `REMOVE_REMOTE_ID` (RRID) command. The `ADD_REMOTE_ID` and `REMOVE_REMOTE_ID` commands are discussed in detail in the following sections.

### Adding Remote IDs

You can define a remote ID with the `ADD_REMOTE_ID` (ARID) command. The format is

```

{ ADD_REMOTE_ID } remote-id [password] [-PROMPT] -ON systemname
{ ARID           } [-PROJECT project-id]
```

The arguments and options for this command are the same as those for the `LOGIN` command (explained earlier in this chapter), with one exception. With the `-PROMPT` option, the command sets the terminal to half-duplex, and prompts you for the password instead of forcing you to enter it on the command line.

remote-id is the user ID that is used for you by the remote system specified by systemname. You must supply any password or project-id required for access to the remote system. If the remote ID does not exist on the remote system or if any required password or project ID is missing or incorrect, a subsequent attempt to access the remote system will fail.

You may have remote IDs for as many as 16 different systems simultaneously, but you may have only one for each remote system. For example, if you add the remote ID JINKS on SYSA and then add the remote ID LYNX on SYSA, LYNX will replace JINKS. All remote IDs are removed when you log out.

For example, assume your user ID is JAMES on SYSP:

```
ADD_REMOTE_ID JAKE PASS -ON SYSK -PROJECT SALES
```

JAKE now exists with the password PASS as a remote ID for accessing SYSK for the duration of the login session or until it is removed with the `REMOVE_REMOTE_ID` command, described below. JAKE is associated with a project named SALES. Remember that JAKE must already have been defined by the System Administrator on SYSK as a valid user ID with the password PASS in order for the remote ID to be effective.



Because the remote ID is valid only for the duration of the login session, you may find it convenient to incorporate the ARID command into your login file (such as LOGIN.CPL), being sure to use the -PROMPT option. When you specify -PROMPT on the command line, ARID prompts for your password on the remote system. Thus, your login password on the remote system does not (and should not) appear in your login file.

### Examining Your Remote IDs

You can use the LIST\_REMOTE\_ID (LRID) command to examine the existing remote IDs you have established. The format is

```
{ LIST_REMOTE_ID } [-ON systemname]
  LRID
```

If you give the -ON option, only the remote ID for systemname is listed; if you omit the -ON option, all of your remote IDs are displayed. Passwords are never displayed. For example,

```
OK, LIST_REMOTE_ID
System  User ID                      Project ID
-----  -
SYSB    JIM
SYSK    JAKE                      SALES
SYSM    JODY
OK,
```

### Removing Remote IDs

Your remote ID list can contain a maximum of 16 remote IDs, that is, one ID per system. If your list has reached the 16-ID limit, you cannot add more remote IDs unless you remove at least one with the REMOVE\_REMOTE\_ID command. (To list existing remote IDs, use the LIST\_REMOTE\_ID command, explained above.)

The command format for REMOVE\_REMOTE\_ID is

```
{ REMOVE_REMOTE_ID } -ON systemname
  RRID
```

where systemname is the node name of the system whose ID is to be deleted. If systemname is not in the list, you will receive the error message "Not found".

# 3

## File System Security

To protect file system data from illegal access, PRIMOS provides a protection mechanism known as Access Control Lists (ACLs). Access control lists specify who may have access to a file system object and exactly what kind of rights each user has.

The ACL system is extremely flexible. Various access rights may be granted or denied to a single user or to a group of users. An ACL can protect a single object or a set of objects.

Protection may be very loose; for example, all users may be granted ALL access rights. On the other hand, protection may be very secure; no one except the user with the P (Protect), ALL, or O (Owner) access rights to a file system object may be allowed any access at all.

Moreover, if you do not wish to use the ACL system actively, you can establish one ACL (or have your System or Project Administrator establish one for you) that automatically protects all your work in a given directory and its subtree. This is called default protection.

This chapter introduces ACLs, and includes the following topics:

- What ACLs are and what format they have
- Types of access rights and types of users
- Types of ACLs (standard ACLs, priority ACLs, and device ACLs)
- What access categories are

- Types of protection (specific ACLs, access categories, and default protection)
- Who may distribute rights

Chapter 4 explains how to set and use ACLs if you are allowed to do so by your System Administrator.

#### WHAT ARE ACCESS CONTROL LISTS (ACLs)?

An Access Control List (ACL) is a list of users and the access privileges granted to each of those users. When an ACL is associated with a file, a directory, or a segment directory, it becomes the protective mechanism for that object.

A simple ACL could look like this:

```
BILL:    ALL
$REST:   NONE
```

In this example, user BILL has ALL rights to the object that this ACL protects. \$REST, a special designation indicating "everybody else," has no rights (NONE).

When any user gives a command concerning an ACL-protected file or directory, PRIMOS checks the user ID against the ACL associated with the file or directory. If the user has the appropriate access rights, the command is executed; if not, the command is not executed, and PRIMOS returns the message "Insufficient access rights" or "No Information".

Types of Access Rights

The access rights or privileges that ACLs grant to users are summarized in Table 3-1.

Table 3-1  
ACL Access Rights

Symbol	Right	Applies To	Meaning
X	Execute	Files	EPF file may be executed.
R	Read	Files	File may be read.
W	Write	Files	File may be modified.
U	Use	Directories and Devices	User may attach to directories or use a device.
L	List	Directories	Directory contents may be listed.
A	Add	Directories	Directory entries may be added.
D	Delete	Directories	Directory entries may be deleted.
P	Protect	Directories	Access rights may be changed.
O	Owner	Files and Directories	RWLOCK and access rights except P and ALL may be changed.
ALL		Files and Directories	All of the above rights.
NONE		Files, Devices, and Directories	No access allowed.

Definitions, uses, and combinations of access rights are discussed more fully at the end of Chapter 4 in the section TIPS ON SETTING ACCESS RIGHTS.

### Types of Users

Access rights are granted to users. The designation (or identifier) for users in the ACL may be any of three types:

- A user ID (for example, JONES).
- A group name. All group names begin with a period (for example, .COMMITTEE). A number of user IDs are associated with a group, and each of them has the access rights granted to the group name. For example .COMMITTEE might be composed of user IDs JANE, FRED, and BARBARA. An individual user may be a member of as many as 32 groups. Groups are established by the System or Project Administrator in the User Profile Data Base.
- The special identifier \$REST. \$REST specifies the rights granted to all users not otherwise identified in the ACL (that is, "everybody else").

### Types of ACLs

PRIMOS defines three types of ACLs:

- Standard ACLs, which may be set on individual files and directories by any user who has Protect (P) rights on the object's parent directory
- Priority ACLs, set by privileged users for an entire disk partition
- Device ACLs, set for devices by the System Administrator or a privileged user from the supervisor terminal

Standard ACLs may be set by individual users on their own files and directories if the System Administrator allows this. If your administrator allows you to set ACLs, you are responsible for protecting your files and directories from unauthorized use. Chapter 4 explains how to use ACL commands for protecting your files and directories.

Priority ACLs may be used by privileged users only. Priority ACLs are set and enforced on an entire disk partition, thus allowing either the System Administrator or another privileged user to override any standard ACLs set by a user.

Device ACLs are set by the System Administrator to protect data contained on media mounted on an assignable peripheral device, such as a tape drive.

What an ACL Looks Like

An ACL has a specific format. A colon separates the identifier (user ID, group name, or \$REST) from the rights associated with it. On the terminal, the pairs of identifiers and rights are displayed in columns.

For example, consider the following file ACL:

```
CAROL:      ALL
.COMMITTEE:  RW
$REST:      R
```

In this example, an ACL gives CAROL all access rights (Owner, Protect, Delete, Add, List, Use, Execute, Read, and Write). Everyone in the .COMMITTEE group has Read and Write access, and everyone else (that is, \$REST) has Read access only.

Overlapping Access Rights

A user who is in more than one ACL group is granted the sum (logical union) of all access modes for each of those groups. If a user has access rights from both a user ID and a group name, the user ID takes precedence, and the user is granted rights under the user ID only.

Consider the following example. The groups .ANIMALS and .RODENTS are composed of the following users:

.ANIMALS

```
SQUIRREL
ROBIN
MOUSE
DOG
```

.RODENTS

```
SQUIRREL
MOUSE
```

The directory FEEDER is protected by an ACL that contains the following entries:

```
DOG:      NONE
.ANIMALS:  DLU
.RODENTS:  ALU
$REST:    LU
```

ROBIN has Delete, List, and Use access to FEEDER as a member of the .ANIMALS group. SQUIRREL and MOUSE have Delete, Add, List, and Use access as the sum of rights granted to the .RODENTS and .ANIMALS groups. DOG has no access; although DOG is in the .ANIMALS group,

user ID rights (here NONE) take precedence over group rights. WORM, another user, has List and Use access, because he is covered by the \$REST identifier.

#### SPECIFIC ACLS AND ACCESS CATEGORIES

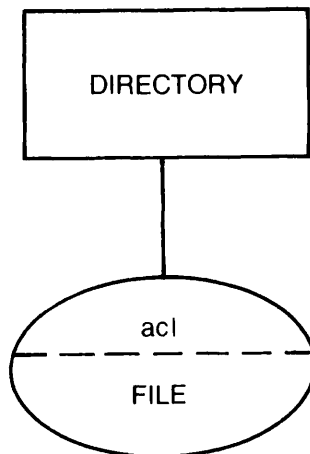
An ACL may exist in either of the following three forms:

- As an unnamed ACL, that is, an unnamed attribute of a file system object, called a specific ACL
- As a named ACL, that is, a named file system object, called an access category
- As a default ACL, that is, an ACL that is part of another object but also protects a subordinate file system object (discussed later in this chapter)

These definitions are explained fully below.

#### Specific ACLs (Unnamed ACLs)

A specific ACL exists as an attribute of a specific file system object and not as a named file system object in itself. The ACL does not appear in a directory listing. It is linked to the single object that it protects and has no separate existence of its own. Figure 3-1 illustrates a specific ACL protecting a file in a directory.

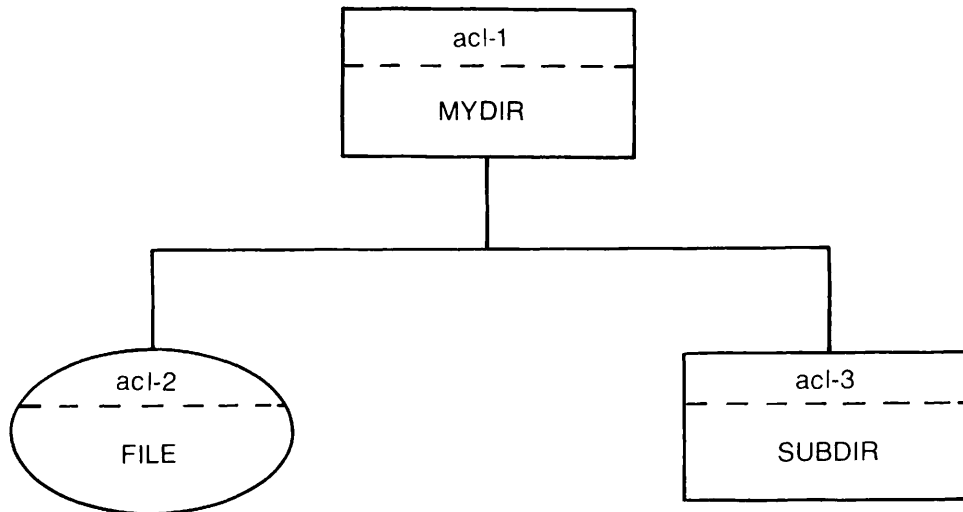


A Specific ACL Protecting a File  
Figure 3-1

By using the SET\_ACCESS command, you may set a specific ACL on a file, segment directory, or directory. By doing so, you may define precisely who may access the object and exactly what rights each user has. If you set a specific ACL on an object and later delete the object, the ACL (as an attribute of the object) is deleted along with it.

If you set an ACL on a directory, it automatically protects all objects within that directory (and within its subtree) that are not otherwise protected. This is explained in more detail in the discussion DEFAULT PROTECTION below.

Figure 3-2 illustrates three specific ACLs in a directory tree. acl-1 protects the directory MYDIR. acl-2 protects the file FILE. acl-3 protects the subdirectory SUBDIR. Each one has been set individually.



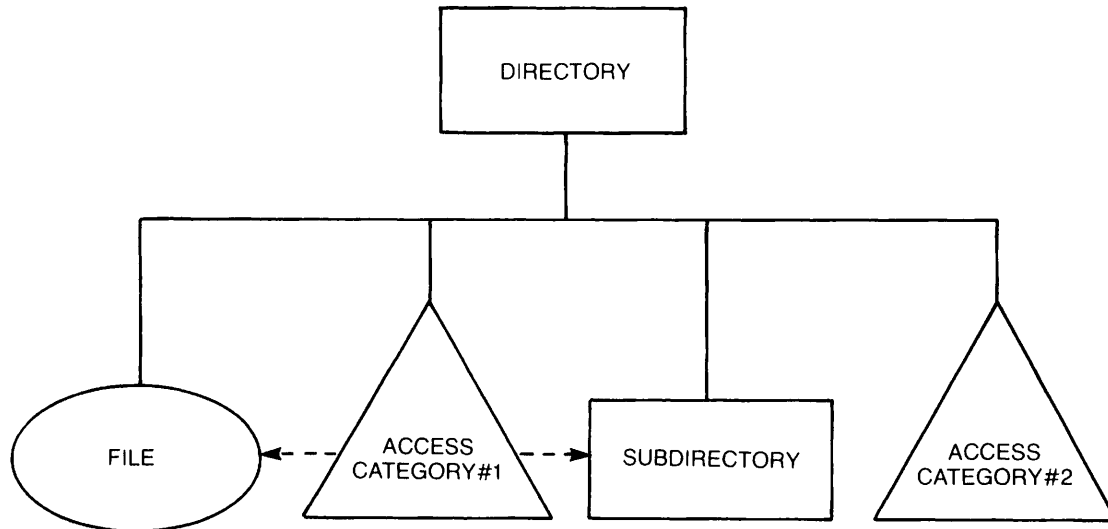
Specific ACLs Protecting Objects  
Figure 3-2

#### Access Categories (Named ACLs)

An access control list may also exist as a named file system object (such as "GUARD.ACAT") that resides within a directory at any level. In this form the list is called an access category. It differs from a specific ACL in that it exists as a separate, named file system object. Moreover, an access category may exist without being linked to any object. It may protect any number of files, directories, and segment directories; or, it may protect none. An access category must reside in the directory containing the object(s) it explicitly protects.



Figure 3-3 illustrates two access categories, a file, and a subdirectory in a directory. Access category #1 is protecting the file and the subdirectory. Access category #2 is not currently protecting anything.



A dotted arrow (<-->) points from an access category to an object it protects.

Access Categories in a Directory  
Figure 3-3

The following example illustrates how access categories appear in a directory listing, displayed with the LD command:

OK, LD

<DISK>JODY>WORK (ALL access)

18 records in this directory, 18 total records out of quota of 0.

3 Files.

FINANCES	MEMO	REPORT
----------	------	--------

2 Access Categories.

COVER.ACAT	MEMO.ACAT
------------	-----------

OK,

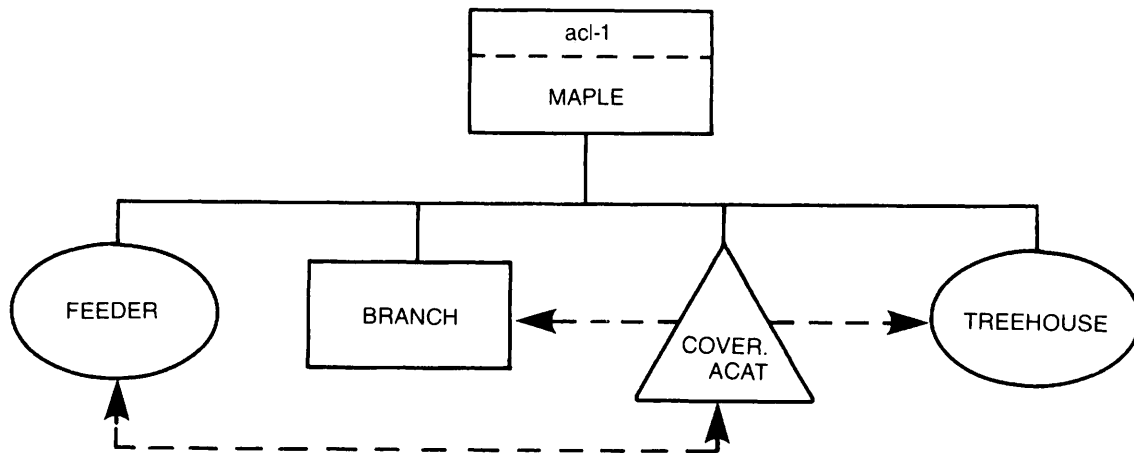
In this example, the directory WORK contains two access categories: COVER.ACAT and MEMO.ACAT. This directory listing gives no information about whether these access categories are currently protecting any object(s). You obtain this information with the -CATEGORY\_PROTECTED option to the LD command, discussed in the PRIMOS Commands Reference Guide.

If you have a group of objects (such as a group of programs) that you want to make accessible to a special group of users, you can create an access category (a named ACL) and link all the objects to the access category. The use of access categories provides a convenient alternative to setting the same specific ACL individually on each of the objects; the convenience is apparent if the list of users is long or needs frequent adjustment. By changing the access rights once, in the access category, you can change the protection on all the objects that the access category protects.

If you delete the objects linked to an access category, the access category itself continues to exist as a file system object.

#### Note

It is essential that you do not confuse a specific ACL, which exists without a name of its own as an attribute of a single object, with an access category, which has a name, exists independently, and may be linked to several objects. Figure 3-4 illustrates the difference.



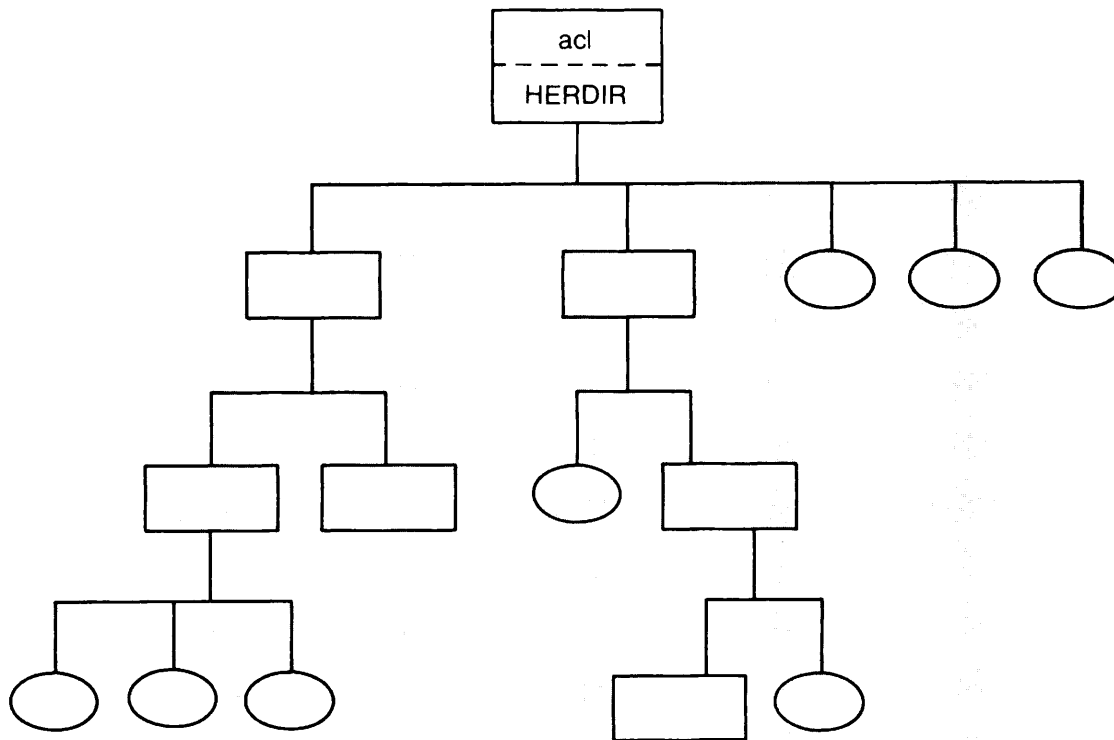
acl-1 is a specific ACL and an attribute of MAPLE;  
COVER.ACACAT is an access category that is linked to  
three objects: FEEDER, BRANCH, and TREEHOUSE.

Specific ACLs and Access Categories  
Figure 3-4

#### DEFAULT PROTECTION

You do not need to set a specific ACL or an access category on each file system object individually in order to protect it. Instead, you may use default protection, which provides protection automatically. If you set a specific ACL or an access category on a directory, this ACL automatically provides default protection for all objects lower in the directory tree, unless you explicitly specify otherwise. This default feature is useful if all your work should have the same protection. In this case, you can set an ACL only once, on the topmost directory you use.

Figure 3-5 illustrates a directory tree protected by default protection. The specific ACL protecting HERDIR has automatically become the default protection for each of the other objects in the tree.



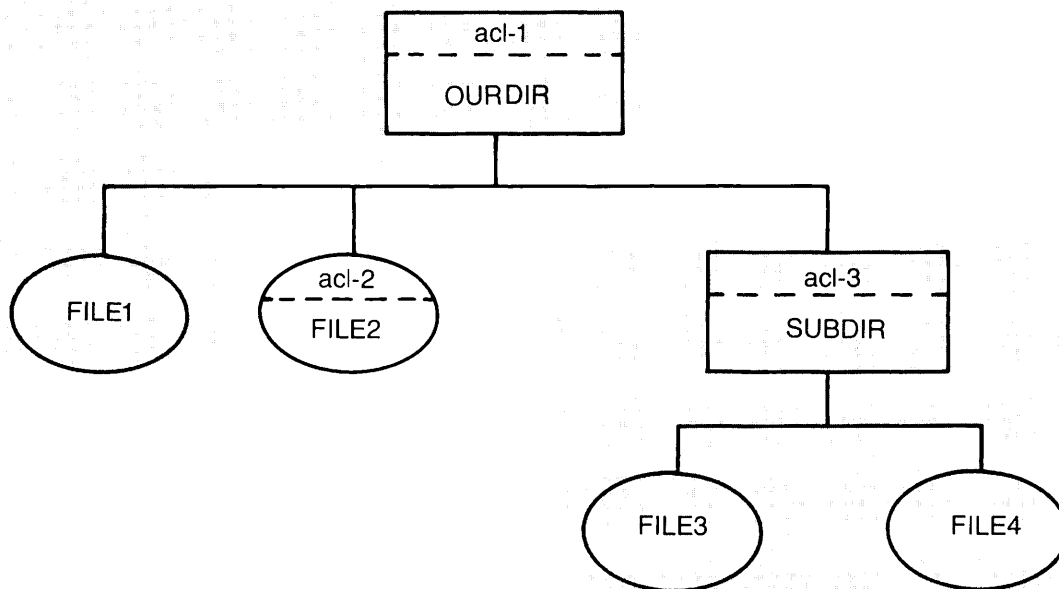
The shaded area includes all objects protected by default from the specific ACL protecting HERDIR.

A Directory Tree Under Default Protection  
Figure 3-5

Providing Default Protection From Specific ACLs

If you have Protect (P) or Owner (O) access, you may override the default protection by setting a specific ACL on any given object. This might be useful if you have an object (for example, a sensitive report) that you wish no one else to access, but you want other users to be able to read your other files. You can set a specific ACL on this single object so that only you can access it. At the same time you can allow \$REST (everybody else) Read (R) access to your other files through default protection.

Figure 3-6 illustrates the combined use of specific ACLs and default ACLs. OURDIR is protected by the specific ACL acl-1. FILE1 is also protected by acl-1, which serves as a default ACL. FILE2 is protected by specific ACL acl-2. SUBDIR is protected by specific ACL acl-3. FILE3 and FILE4 are also protected by acl-3, which provides them default protection.



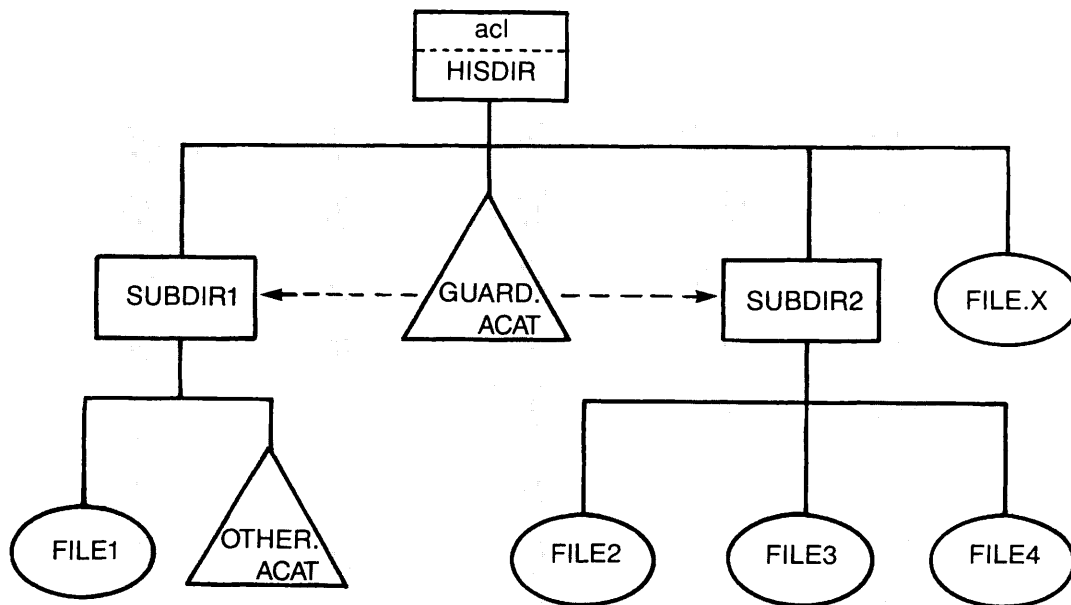
Shaded areas show spheres of protection provided by acl-1 and acl-3.

Default Protection and Specific ACLs  
Figure 3-6

Providing Default Protection From Access Categories

Access categories can also provide default protection. If you set an access category on a directory, this access category automatically becomes the default protection for all objects created lower in this branch of the tree, unless you explicitly specify otherwise.

Figure 3-7 illustrates the spheres of protection for two access categories. SUBDIR1 and SUBDIR2 are protected by GUARD.ACAT. FILE2, FILE3, and FILE4 are also protected by GUARD.ACAT through default protection, because GUARD.ACAT is linked to SUBDIR2. FILE1 is protected by OTHER.ACAT.



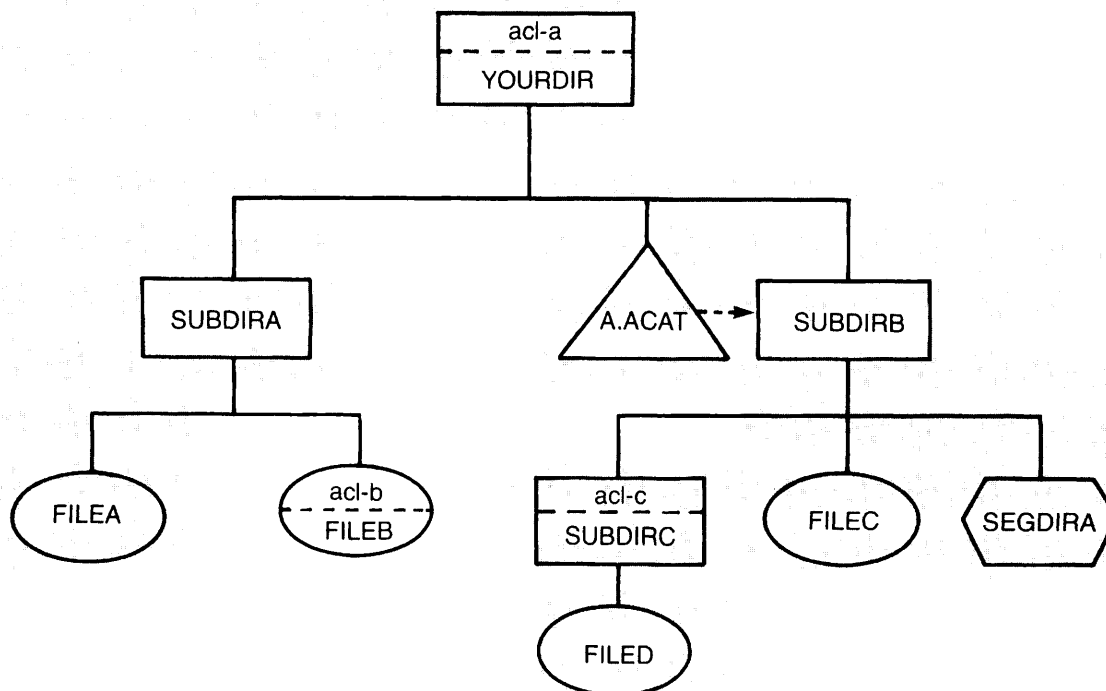
Dotted arrows (<-->) point from an access category to an object that it protects.

Shaded areas show the spheres of protection of the access categories.

Default Protection and Access Categories  
Figure 3-7

Combining Default Protection, Specific ACLs, and Access Categories

All three protection mechanisms -- specific ACLs, access categories, and default protection -- may exist in the same directory tree, as illustrated by Figure 3-8.



A dotted arrow (--->) points from the access category to the object that it protects.

Shaded areas show the spheres of protection for the specific ACLs and access category.

Default Protection, Specific ACLs,  
and Access Categories

Figure 3-8

In Figure 3-8, specific `acl-a` protects `YOURDIR` explicitly and, by default, `SUBDIR` and `FILEA`. Specific `acl-b` protects `FILEB` explicitly. Access category `A.ACAT` protects `SUBDIRB` explicitly and, by default, `SEGDIRA` and `FILEC`. Specific `acl-c` protects `SUBDIRC` explicitly and, by default, `FILED`.

PRIORITY ACLS and DEVICE ACLS

At times (for example, during system backups) the System Administrator may need to control all access to the system. For this reason, the system operator or System Administrator may override any user-defined ACL by creating a priority ACL. The priority ACL defines access for the entire disk. When a priority ACL is active on a disk, its contents are always displayed either by the LIST\_ACCESS command following the normal ACL or by the LIST\_PRIORITY\_ACCESS command. The LIST\_ACCESS and LIST\_PRIORITY\_ACCESS commands are further discussed in Chapter 4, USING ACLS.

Device ACLs are set by the System Administrator to protect data contained on media mounted on an assignable peripheral device, such as a tape drive. As a user you will either have U (Use) or NONE access to a device. If you attempt to assign a device and receive the message

Insufficient access rights

it is because the System Administrator has not included you in the ACL group that has the Use (U) access right.

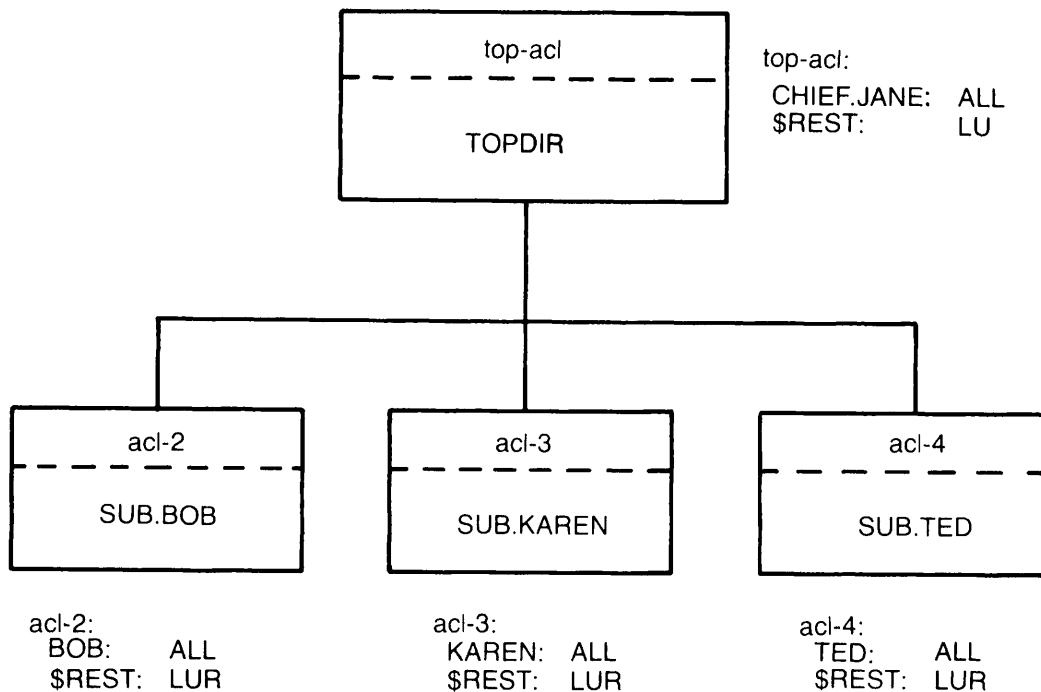
WHO MAY DISTRIBUTE RIGHTS

Access rights to your origin directory are usually first assigned to you by the System Administrator (or by a Project Administrator). If the System Administrator has not granted you Protect or Owner access rights, you will not be able to set or use ACLs.

If you are granted Protect access rights, you can change the access of the objects, that is, files and directories, for which you have the Protect access right. If you are granted Owner access rights, you can change the all access rights except the P (Protect) and ALL access rights of the objects for which you have the Owner access right. With either the Protect or Owner access right you can share rights with other users or designate partial rights to others anywhere in their directory tree.

For example, consider the directory tree in Figure 3-9.

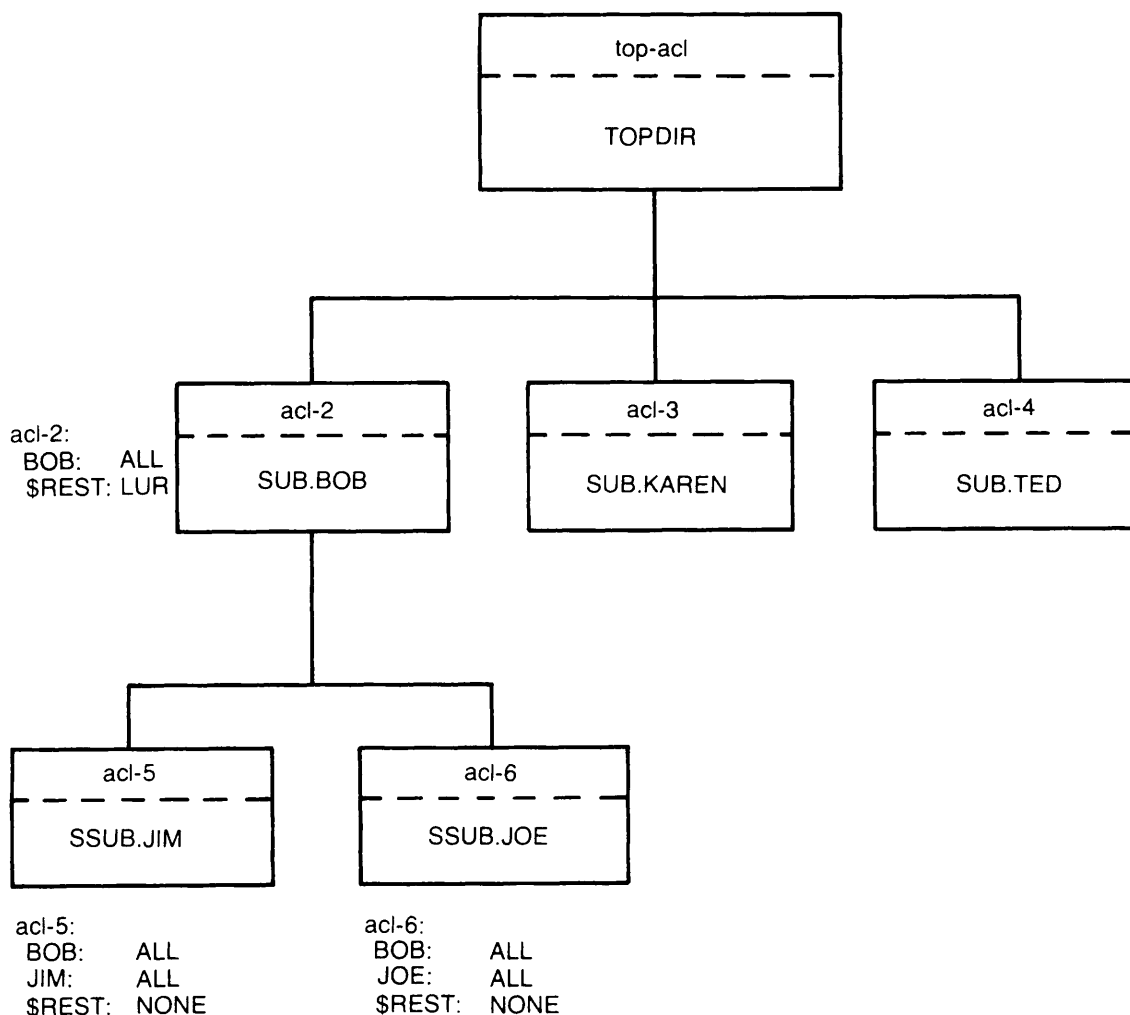




ACLs in a Directory Tree  
Figure 3-9

In Figure 3-9, CHIEF.JANE has ALL rights to TOPDIR. CHIEF.JANE has granted ALL rights in SUB.BOB, SUB.KAREN, and SUB.TED to BOB, KAREN, and TED, respectively. CHIEF.JANE herself (as part of the \$REST group) now has only List, Use, and Read access to SUB.BOB, SUB.KAREN, and SUB.TED. However, she could, if she wished, alter the specific ACLs on each of the subdirectories to grant herself ALL rights.

The control available to the users of the subdirectories is slightly different. BOB has ALL rights to his own directory (SUB.BOB) and can change rights to it. BOB has only List, Use, and Read rights to SUB.KAREN and SUB.TED, and may not change these rights. He may grant rights to subdirectories lower in his own branch of the tree, as illustrated by Figure 3-10.



More ACLs in a Directory Tree  
Figure 3-10

In Figure 3-10, BOB has granted ALL rights to JOE and to himself in SSUB.JOE and ALL rights to JIM and himself in SSUB.JIM. Everyone else (\$REST) has no rights (NONE).

If your System Administrator has given you the Protect (P) access right in your origin directory, you may set and change the access rights to your files and directories as needed. Chapter 4, USING ACLS, explains the commands used to set ACLs.

#### ACCESS REQUIREMENTS FOR ESSENTIAL PRIMOS COMMANDS

PRIMOS recognizes more than 100 commands. Some of these commands invoke subsystems that themselves respond to subcommands or extensive dialogs. However, most users can do most of their work by using only

about a dozen commands. This section introduces some of the essential commands needed by all users and explains the access rights you must have in order to use them. Table 3-2 summarizes these commands and the access rights required to execute them. For more information on these commands, refer to the Prime User's Guide.

Table 3-2  
Access Rights for Essential PRIMOS Commands

Command	Function	Required Access Rights
ATTACH	Connect to a different directory	U
CNAME	Change a file system object name	DA
COPY	Copy a file system object	RU (on object to copy from) AU (on object to copy to) D (to copy over an existing object)
CREATE	Create an empty subdirectory	A
DELETE	Remove unwanted file system objects	D
EDIT_ACCESS	Edit ACL for file system object	P or O
LIST_ACCESS	List ACLs for file system object	L
LD	List the contents of a directory	L
SET_DELETE	Protect a file system object from accidental deletion	D
SLIST	Examine contents of a file	R
SET_ACCESS	Set ACL on a file system object	P or O

# 4

## Using ACLs

If your System Administrator permits it, you will want to use ACLs to secure your files and directories against unauthorized use.

This chapter explains how to use the ACL system and includes

- Commands for using ACLs and access categories
- Tips on setting access rights effectively

### COMMANDS FOR USING ACLS AND ACCESS CATEGORIES

The next sections of this chapter explain the commands for using ACLs and access categories. These commands explain how to

- List access rights (LIST\_ACCESS)
- Control access to files and directories (SET\_ACCESS)
- Change access to files and directories (EDIT\_ACCESS)

## LISTING ACCESS RIGHTS

The LIST\_ACCESS (LAC) command allows you to list the access rights connected to any object. The format is

```
{ LIST_ACCESS } [objectname]
{ LAC }
```

objectname may be a pathname. If you do not specify an objectname on the command line, the system lists the access rights for the current directory. If the object is an access category, the system lists its contents (the ACL). In all other cases, the ACL that protects the object you specify is listed. Priority ACLs (explained later in this section) are always listed after other ACLs.

### Examples

To list the access rights for the current directory, enter the following:

```
OK, LIST_ACCESS
ACL protecting "<Current directory>":
    CAT:      NONE
    SQUIRREL: ALUR
    .BIRDS:   ALL
    $REST:    LUR
OK,
```

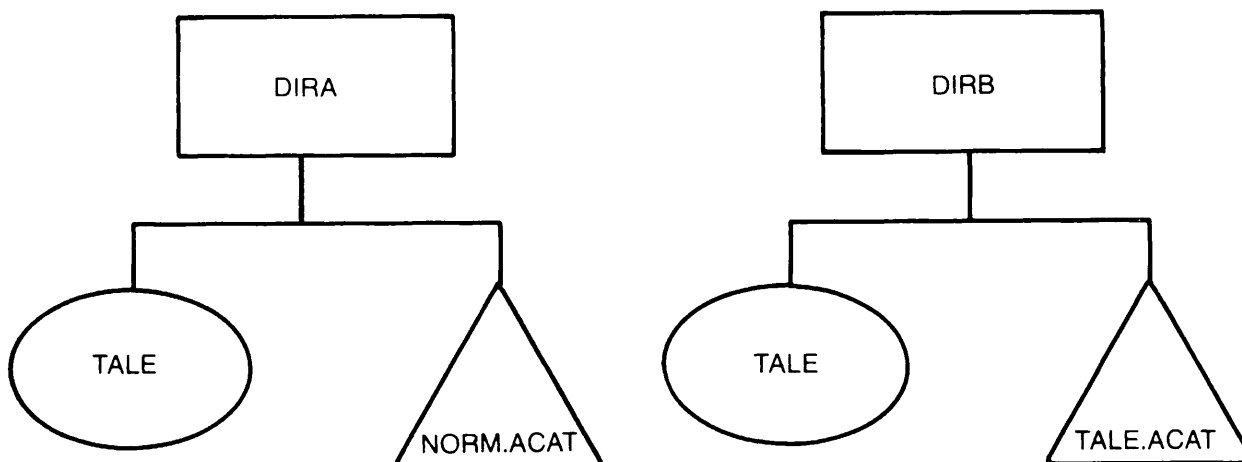
To list the access rights for a subdirectory, enter the following:

```
OK, LIST_ACCESS BEECH>BRANCH5>TWIG42
"BEECH>BRANCH5>TWIG42" protected by default ACL (from "<FOREST>BEECH"):
    ROBIN:    DALUR
    .BOYS:    NONE
    .LEAVES:  ALL
    $REST:    LUR
OK,
```

### Specifying the .ACAT Suffix

You do not need to supply the .ACAT suffix when listing an access category, unless there is an object in the directory with the same base

name as that of the category. For example, consider the two directories in Figure 4-1:



Listing Access Categories  
Figure 4-1

To list the contents of NORM.ACAT in DIRA, you may type either of the following:

```
LIST_ACCESS NORM
LIST_ACCESS NORM.ACAT
```

To list the contents of TALE.ACAT in DIRB, you must specify

```
LIST_ACCESS TALE.ACAT
```

(If you specify LIST\_ACCESS TALE, access rights for the file TALE are displayed.)

### Listing Priority ACLs

Priority ACLs are always displayed by the LIST\_ACCESS command following the normal ACL.

For example,

OK, LIST\_ACCESS

ACL protecting "<Current directory>":

JOHN: ALL

.GROUP: ALL

\$REST: LUR

Priority ACL in effect for "<Current directory>":

.ADMINISTRATORS: ALL

OK,

In addition, you may also examine the contents of a priority ACL on any disk using the LIST\_PRIORITY\_ACCESS command. The format is

```
{ LIST_PRIORITY_ACCESS } diskname
{ LPAC }
```

For example,

OK, LIST\_PRIORITY\_ACCESS FOREST

Priority ACL on partition "<FOREST>":

DEER: ALL

\$REST: NONE

OK,

If no priority ACL exists on the disk, you will receive the message

Priority ACL not found. <FOREST> (list\_priority\_access)  
ER!

#### CONTROLLING ACCESS TO FILES AND DIRECTORIES

The SET\_ACCESS command specifies the access rights to be associated with a file or directory and controls the creation of access categories.

To use the `SET_ACCESS` command, you must have Protect (P) or Owner (O) access under at least one of the following conditions, as appropriate:

- To the parent directory of any object specified
- To the directory specified
- In the existing access category specified

In all cases you must have List (L) and Use (U) access to the parent directory.

The syntax of the command has the following four forms:

```
SET_ACCESS pathname acl [-NO_QUERY]
SET_ACCESS pathname -CATEGORY category-name [-NO_QUERY]
SET_ACCESS pathname -LIKE reference
SET_ACCESS pathname [-NO_QUERY]
```

The operation of each of these forms is discussed next.

### Setting ACLs on Existing Files, Segment Directories, and Directories

To protect an existing file, segment directory, or directory with a specific access control list, give the command

$$\left\{ \begin{array}{l} \text{SET\_ACCESS} \\ \text{SAC} \end{array} \right\} \text{ pathname acl } \left\{ \begin{array}{l} \text{-NO\_QUERY} \\ \text{-NQ} \end{array} \right\}$$

pathname is the name of the file, segment directory, or directory to be protected. If pathname does not exist, `SET_ACCESS` creates it as an access category, as explained under Creating an Access Category below.

acl is the access control list that specifies access for the pathname. The ACL is composed of pairs of identifiers and rights. Each pair is connected by a colon in the form: identifier:rights. Do not put spaces before or after the colon. To grant multiple rights, type the letter symbols for the rights with no intervening spaces (for example, PDALU). You may type the letter symbols in any order (such as RLJU, WURALD), but, when listed, they will always appear on the terminal in the order: OPDALURWX.

The ACL may contain as many as 32 pairs, but may not in total be longer than 160 characters, including blanks. Multiple pairs are separated by spaces. The \$REST grouping, unless specified on the command line, is automatically given no rights (the designation NONE).



If a specific ACL (as opposed to the default ACL) already exists for this object, you are queried before the default ACL is replaced with the new specific ACL. You may suppress the query using the `-NO_QUERY` option.

For example, consider the following session:

```
OK, SET_ACCESS BLUEJAY JOHNNY:ALUR .DOGS:LUR
A specific ACL for "BLUEJAY" already exists.
Do you want to replace it? YES
OK,
```

The command and the YES answer now give to JOHNNY Add, List, Use, and Read rights to BLUEJAY. The users in the .DOGS group get List, Use, and Read rights. Everybody else (\$REST) automatically gets no rights. If BLUEJAY had not already been protected by a specific ACL, the query would not have appeared. The new specific ACL on BLUEJAY looks like this:

```
JOHNNY:      ALUR
.DOGS:       LUR
$REST:       NONE
```

### Using Access Categories

Creating an Access Category: To create an access category, use the `SET_ACCESS` command with the following format:

$$\left\{ \begin{array}{l} \text{SET\_ACCESS} \\ \text{SAC} \end{array} \right\} \text{ category-name acl } \left\{ \begin{array}{l} \text{-NO\_QUERY} \\ \text{-NQ} \end{array} \right\}$$

category-name is the name to be given to the new access category. The category name may be given as a pathname. If this name does not end in the suffix `.ACAT`, (for access category) the system automatically appends this suffix. acl specifies the pairs of identifiers and rights for the access category. Unless you specify the `-NO_QUERY` option, you are queried to double-check your intent.

For example, assume that you want to create the following access category:

```
OK, SET_ACCESS PROTECT ME:ALL .GROUP:LUR
"PROTECT.ACAT" does not exist. Create access category? YES
OK,
```

If you now use the LIST\_ACCESS PROTECT command, you see the following display:

```
Access category "PROTECT.ACAT":
ME:      ALL
.GROUP:  LUR
$REST:   NONE
```

#### Note

If the category-name you select for your new access category already exists in your directory as the name of a file, segment directory, or directory, then SET\_ACCESS assumes that you are trying to set a specific ACL on that object.

Replacing the Contents of an Access Category: If the access category already exists, the SET\_ACCESS command replaces the category's existing access list with the new access list specified on the command line. The format is identical to that for creating new access categories:

```
{ SET_ACCESS } category-name acl { -NO_QUERY }
{ SAC         } { -NQ }
```

You do not need to include the .ACAT suffix when you specify the access category name.

For example, assume that you want to change the contents of the access category PROTECT.ACAT created in the last example:

```
OK, SET_ACCESS PROTECT ME:ALL $REST:LUR
"PROTECT.ACAT" is an existing access category.
Do you want to replace it? YES
OK,
```

PROTECT.ACAT now contains the following ACL:

```
ME:      ALL
$REST:   LUR
```

Protecting Objects With Access Categories: To protect an existing file system object with an existing access category, use the SET\_ACCESS command with the following format:

```
SET_ACCESS objectname -CATEGORY category-name
```

objectname may be given as a pathname. The object and the access category must be in the same directory.

For example, assume you want to protect an object with the access category PROTECT.ACAT above:

```
SET_ACCESS MYFILE -CATEGORY PROTECT
```

Deleting an Access Category: To delete an access category, whether it is empty or not, use the DELETE command, as for any other file system object:

```
DELETE category-name
```

If a category currently protecting an object is deleted, the access for that object reverts to the default protection.

#### SETTING ACCESS RIGHTS TO MATCH THE RIGHTS ON OTHER OBJECTS

You can use the SET ACCESS command with the -LIKE option to make the access rights of one object match another object. The format is

```
SET_ACCESS objectname -LIKE reference
```

Both objectname and reference must be the names of existing file system objects and may be given as pathnames. If objectname is the name of an access category, the ACL it contains becomes identical to the ACL associated with the reference. If the objectname is a file, directory, or segment directory, a specific ACL is set on the objectname identical to the ACL associated with the reference.

For example, the file OUTLINE gives ALL access rights to MARY, LUR rights to .GROUP, and no rights to anyone else (\$REST). To set rights on a second file (REPORT) to be identical to rights on OUTLINE, give the command

```
SET_ACCESS REPORT -LIKE OUTLINE
```

The following access rights now exist on REPORT:

MARY:	ALL
.GROUP:	LUR
\$REST:	NONE

REVERTING TO DEFAULT PROTECTION

You may have an object protected by a specific ACL or by an access category and find that you wish to change its protection back to default access. To do this, use the SET\_ACCESS command in the following format:

```
SET_ACCESS objectname
```

objectname must be a file, segment directory, or directory. It may not be an MFD. The objectname may be expressed as a pathname.

For example, consider the following sequence:

```
OK, LIST_ACCESS MEMO
```

```
ACL protecting "MEMO":
    $REST:    DALURW
```

```
OK, SET_ACCESS MEMO
```

```
OK, LIST_ACCESS MEMO
```

```
"MEMO" protected by default ACL (from "<DISK>DIRNAME"):
```

```
    JEAN:     ALL
    $REST:    NONE
```

```
OK,
```

The original rights on the file MEMO (\$REST:DALURW) have been removed and the file has reverted to default access (JEAN:ALL and \$REST:NONE).

CHANGING ACCESS RIGHTS

The EDIT\_ACCESS command allows you to modify existing ACLs and access categories. Its format is

```
{ EDIT_ACCESS } objectname acl { -NO_QUERY }
{ EDAC        }
```

objectname may be any file system object and may be expressed as a pathname. The old access list is modified to reflect the new rights given in each identifier:rights pair specified in acl. If you use EDIT\_ACCESS with objects that are default-protected or category-protected, you are queried to determine whether or not you wish to create a new specific ACL. Use the -NO\_QUERY option to suppress this query.

If the identifier already exists in the ACL, its access is changed. A null access (for example, JOHN:) indicates that the identifier should be removed from the list. To edit an ACL, you must have either Protect (P) or Owner (O) access on its parent directory or Protect or Owner access in the ACL itself. This means that a user, for example, JACK, may change the ACL protecting his top-level directory (assuming that he has Protect or Owner access on it).

For example, consider the following sequence:

OK, LIST\_ACCESS REPORTS

"Reports" protected by default ACL (from "<DISK>DIRNAME"):

JACK: LUR  
STEVE: ALL  
\$REST: NONE

OK, EDIT\_ACCESS REPORTS JACK:DALURW JILL:LUR

"REPORTS" is default-protected. Create specific ACL? YES

OK, LIST\_ACCESS REPORTS

ACL protecting "REPORTS":

JACK: DALURW  
JILL: LUR  
STEVE: ALL  
\$REST: NONE

OK,

JACK has changed his original rights (LUR) to REPORTS to DALURW. JILL now has LUR access. The original rights of STEVE (ALL) and \$REST (NONE) remain unchanged.

#### Note

If you change the access rights on a directory to which you are attached and wish to check the change, you must reattach to the directory before the changes will take effect for you. If you change access rights on your origin directory and subsequently use the ORIGIN command, the rights will not be changed. For the ORIGIN command to reflect the new rights, you must log out and then log in again.

#### EDIT\_ACCESS and SET\_ACCESS

Both the SET\_ACCESS and EDIT\_ACCESS commands can be used to change existing access rights, but the two commands have different results.

SET\_ACCESS replaces the entire existing access list with the new list given on the command line.

EDIT\_ACCESS modifies the existing access list to merge the new list given on the command line with the old list. An identifier that appears in both new and old lists is given its new rights only.

Which Command to Use: If you are making only minor changes to a long access list, EDIT\_ACCESS is probably easier to use. If you are making substantial changes (especially many deletions and few additions or modifications), SET\_ACCESS is probably easier to use.

#### WARNING

To change the access of top-level directories, such as your origin directory, you should use EDIT\_ACCESS, not SET\_ACCESS. If you use SET\_ACCESS and fail to include yourself in the access list, you may no longer have any rights at all to your own directory. You can create the same problem with EDIT\_ACCESS, but to do so, you must explicitly remove yourself from the ACL. If this happens, see your System Administrator.

#### TIPS ON SETTING ACCESS RIGHTS

The following discussion will help you to select the most useful combination of access rights for various purposes.

#### More Information on Access Rights

Owner (O): The Owner (O) right allows you to set any other access right on files or directories except P or ALL. If the object is a file or a segment directory, you are also permitted to set the read/write lock.

Protect (P): Protect access applies to directories and allows you to set rights on any object in the directory. Protect is the most powerful of all access rights. If you have Protect access, you may change the ACL protecting the directory to allow others rights. Because List access is required in order to read the directory in the first place, and Use access is required to attach to it, Protect should never be given to other users without List and Use.

Delete (D): In the ACL system, Delete access is associated with individual directories, not with individual files. Because of this fact, you may mark important files as "delete-protected" with the SET\_DELETE command. If Delete access is available on a directory, any file system object immediately contained in that directory may be deleted. Both Delete and Add access (explained below) are required to change the names of files.

Add (A): Add access applies to directories. Objects can be created only in a directory to which the user has Add access. Because newly created files are opened with all file accesses available, granting Add access without Write access (explained below) allows users to create new files (for example, to copy a file) but not to modify them after they exist. Both Delete and Add access are required in order to change the names of files.

List (L): List access allows the user to list the contents of a directory.

Use (U): Use access allows a user to attach to a directory, use the directory name in a pathname, or use an assignable peripheral device. A user may thus pass through a directory but cannot (without List access) obtain any information about the directory. Use access is available when you want to give access to an object, but also want to keep all other directory information invisible. If, for any reason, an object cannot be accessed, the message "No information" is returned. Use access is required on all directories, including MFDs, in order for you to be able to attach to them, and consequently to search them for entries. Use access, on its own, is the most restrictive access right. However, because Use access controls the user's ability to attach to a directory, it should usually be granted along with other access rights to enable the other rights to work.

Read (R): Read access allows a user to read or execute files.

Write (W): Write access allows the user to create or modify files and to truncate files.

Execute (X): The Execute designation allows a user to execute a local EPF (on the user's local system) but not to read or copy it using standard file system utilities, for example, COPY or FUTIL. A user who has Read access automatically has X access. A user who has only X rights has no rights to remote EPFs.

ALL: The ALL designation grants all of the above rights to a user. Specifying OPDALURWX instead of ALL on the command line grants the same rights as ALL. If you specify OPDALURWX, the list of letters, instead of ALL, appears when you list the ACL.

NONE: The NONE designation denies all access. The \$REST group is automatically given NONE as access rights to protect your work from intrusion.

### Rights for Personal Directories

Normally, you are granted all rights (that is, OPDALURWX, or ALL) to the directories that you log in to and that you create. If your System Administrator does not permit you to change the ACLs to your personal directories, the Protect and Owner access rights are omitted, resulting in DALURWX rights. These rights allow you to access the directory to perform any PRIMOS operation except change the protection and the attributes of objects in the directory and its subtrees.

### Rights for Other Users in a Group or Project

When several people work together on a project, it is often useful to allow them to have access to each other's files, but not to alter or destroy them. In such cases the combination LUR is helpful. Users with LUR rights may examine files and directories and may execute programs, but they may not change the contents of files or directories.

### Rights for Outsiders

True outsiders are often denied rights altogether with the NONE designation. Sometimes, however, you may wish certain files to be accessible to anyone. In this case, granting UR access allows users to attach to a directory and read files whose names are known in advance. UR rights do not allow users to modify files, and do not allow them to list the contents of the directory.

### Sharing Information

Sometimes it is useful to allow users to share files. At the same time the file creators do not want their other files to be altered. In this situation, you can grant ALUR rights to other users. ALUR rights allow new files to be created and written to (because new files are always opened with RW rights), but do not allow existing files to be modified.

### ACL Subroutines

Several subroutines that manipulate ACLs are available. These subroutines allow you to access the ACL system from programs and to set, modify, list, and remove protection on objects just as you would from the terminal. For details on these subroutines, see the Subroutines Reference Guide, Vol II.





# APPENDICES

# A

## Security Related Messages

The following is an alphabetical list and explanation of messages associated with PRIMOS security features.

- ACCESS VIOLATION

You attempted to perform operations in segments to which you have no rights.

- ACCESS\_VIOLATION\$

The process you initiated has attempted to perform a CPU instruction that has violated the access control rules of the processor. No information is available to differentiate between a write violation, read violation, execute violation, and gate violation.

- ACL too big.

Either the ACL contains more than 32 entries, or the ACL exceeds space limitations.

- Already exists.

You attempted to create an object with the same name as an object that already exists.

- Bad remote password.

You attempted to log in to another system using an invalid password.

- Cannot access like reference.

The object you specified in the -LIKE option of SET\_ACCESS could not be accessed for some reason.

- Category protects MFD.

You attempted to delete an access category that protects the MFD. You must remove the MFD from the category before you can delete the category.

- The device is in use.

You attempted to assign a device that is currently assigned to another user.

- Device not assigned.

You attempted to perform I/O operations on a device before assigning that device.

- Device is not started.

You attempted to access a disk partition that is not physically or logically connected to the system. To access the disk, the system operator must first start it up.

- The directory is not empty.

You attempted to delete a directory that is not empty. (The directory still contains files.)

- Disk is write-protected.

You attempted to write to a disk that is write-protected.

- File is delete-protected.

You attempted to delete a file on which the delete-protect switch has been set to prevent deletion.

- Illegal access mode.

The access portion of an access pair contains an unknown access mnemonic.

- Illegal identifier.

The identifier portion of an access pair contains an illegal user ID or group ID.

- Illegal name.

The name you specified for a file or directory does not conform to PRIMOS file naming standards.

- Illegal remote reference.

You attempted to perform a network operation when either the network was not in operation or you were not part of the network.

- Illegal treename.

The string you specified for a treename contains incorrect syntax.

- Incorrect access control list format.

The ACL you specified in SET\_ACCESS or EDIT\_ACCESS was not in proper format. This message usually results if you omit a colon between the identifier and the access rights.

- Insufficient access rights.

You do not have the necessary access rights either to a file system object or to perform the action you desire.

- Maximum remote users exceeded.

No more users may access the network.

- No information.

You have tried to access a file system object and you could not. Some common reasons include insufficient access rights, a nonexistent object, and the wrong type of object. This message does not reveal whether the specified object exists.

- Not a file or a directory.

You attempted to protect an access category with an ACL. Only files, directories, and segment directories can be protected by an ACL.

- device-name NOT ASSIGNED

You attempted to access an I/O device that has not been assigned to you.

- Object is category-protected.

You attempted to use `EDIT_ACCESS` on an object that is currently protected by an access category.

- Object is default-protected.

You attempted to use `EDIT_ACCESS` on an object that is currently default-protected.

- Operation illegal on directory.

You tried to perform an illegal operation (such as editing) on a directory.

- Priority ACL not found.

No priority ACL exists for the partition you specified in a `LIST_PRIORITY_ACCESS` command.

- Remote system not up.

You attempted to access a remote system that is not running.

- Slave validation error.

Either you are trying to access a remote system that requires user validation and you did not issue an `ADD_REMOTE_ID` command (to establish a remote ID), or a previous `ADD_REMOTE_ID` command established a user ID, project ID, or password that was invalid on the remote system.

- System console command only.

The command you issued can be issued only from the supervisor terminal.

- Top-level directory not found or inaccessible.

Your attempt to attach to a top-level directory has failed either because the directory does not exist, because PRIMOS cannot reach the system where it does exist (if a remote system is down), or because you do not have the right (Use, or U, access) to attach to it. You remain attached to your current directory.

# B

## Glossary

The following is a glossary of concepts and conventions that may be useful to new users of Prime computers and the PRIMOS operating system.

### access category

A file system object that contains only an access control list. An access category is used to protect other file system objects.

### Access Control List (ACL)

A list of users and the access privileges granted to each user.

Access rights are

O	Owner
P	Protect
D	Delete
A	Add
L	List
U	Use
R	Read
W	Write
X	Execute
ALL	All of the above
NONE	No access at all

When an ACL is associated with a file system object, it protects that object by allowing access only to the users listed within it, and allowing those users only their listed rights. See Chapters 3 and 4 for more information on access control lists.

### ACL

See Access Control List.

### category name

The name of an access category. Category names follow the rules for objectnames. See also objectname.

### current directory

The directory to which you are currently attached.

### directory

A file system object that contains a list of objectnames, along with information on their characteristics and location. MFDs, origin directories, and subdirectories are all directories.

### directory name

The name of a directory. Directory names follow the rules for objectnames. See also objectname.

### device

A mechanical unit, such as a card reader, tape reader, or keyboard.

### file system object

An organized collection of data stored on a disk (or on a peripheral storage medium such as tape). Each object has an identifying label called an objectname. File system objects include files, directories, segment directories, and access categories.

### filename

The name of a file. Filenames follow the rules for objectnames. Directory names and a filename may be combined into a pathname. Most commands accept a pathname wherever a filename is required. See also objectname.

### identifier

The generic term for "user" listed in access control lists. Identifiers may be of three types: a user ID (for example, SMITH), a group name (for example, .CLUB), or the special identifier \$REST (that is, "everybody else").

### Initial Attach Point (IAP)

The directory to which you are attached when you log in. Also called your origin directory.

### MFD

The Master File Directory. A special directory that contains the names of the directories on a logical disk or partition. There is one MFD for each logical disk.



object

See file system object.

objectname

A sequence of 32 or fewer characters that names a file system object. Within any directory, each objectname is unique.

Objectnames may contain only the following characters:

A through Z  
0 through 9  
\_ # \$ - . \* & /

The first character of an objectname must not be numeric. On some devices an underscore (\_) prints as a back arrow (←).

In objectnames, suffixes indicate various types of files. (A period separates the suffix from the base name of the file.)

Objectname suffixes that are recognized by Prime software include

<u>Objectname With Suffix</u>	<u>Meaning</u>
filename.compiler-name	Source file
filename.LIST	Listing file
filename.BIN	Binary (object) file
filename.RUN	Dynamic V-mode runfile (executable)
segdirname.SEG	Static V-mode runfile (executable)
filename.SAVE	Static R-mode runfile (executable)
filename.CPL	CPL file
categoryname.ACAT	Access category
filename.CDML	COBOL Data Manipulation Language (CDML) preprocessor input file
filename.FDML	FORTTRAN Data Manipulation Language (FDML) preprocessor input file
filename.DPTCFG	DPTCFG input file
filename.EFASL	EMACS fastload file
filename.EM	EMACS extension file
filename.FBIN	FORMS object file
filename.FORM	FORMS file
filename.ERROR	Error output file (used by VPTCFG, CDML, and FDML)

Other suffixes, which are not recognized by Prime software but are used by Prime and are recommended to order and clarify your work, include

<u>Objectname With Suffix</u>	<u>Meaning</u>
filename.ABBREV	Abbreviation file
filename.COMI	Command input file
filename.COMO	Command output file
filename.CONFIG	Configuration file
filename.GVAR	Global-variable file
filename.MAP	Map file (created by BIND, SEG, or LOAD)
filename.PH	Phantom command file
filename.T	Temporary file
filename.RUNI	RUNOFF source (input) file
filename.RUNO	RUNOFF output file
filename.ERR	Error message text file
filename.HELP	Help text file
filename.INS.language	Insert and include files
filename.LOG	System activity log (used by FTS, VISTA, OAS, POWERPLUS)

#### open

Active state of a file unit. A command or program opens a file unit in order to read it or write to it.

#### option

A PRIMOS term, usually preceded by a hyphen and related to a command. An option signals an activity. For example, in the command line

SPOOL -LIST

-LIST is the option that tells the spooler to list at your terminal the files waiting to be printed.

#### origin directory

The directory to which you are attached when you log in. Also called your Initial Attach Point (IAP).

#### packname

See volume name.

#### parent directory

Any directory that contains a subdirectory. For example, if DIR.2 is a subdirectory of DIR.1, DIR.1 is the parent directory of DIR.2.

**pathname**

A multipart name that uniquely specifies a particular file system object within a file system tree. A pathname (also called a treename) gives a path from a disk volume, through a directory and subdirectories, to a particular object.

**phantom**

A process that runs independently of a terminal, under the control of a CPL program or a command file.

**process**

A particular program running in a particular address space.

**quota**

The maximum number of records (1 record = 1024 user data words) that the contents of a directory can occupy on a disk partition. A quota of 0 means that no record limit exists on the directory.

**reserved characters**

Characters that are reserved by PRIMOS for special uses. They may not be used in objectnames. The following are reserved characters:

= ; , ( ) ' [ ] ! { } ^ " ? : ~ | < > + ' % \  
space delete/rubout

**runfile**

The executable version of a program, consisting of the linked binary file, subroutines and library entries used by the program, common areas, initial settings, and so on. (Runfiles are created by BIND, SEG, or LOAD.)

**segment directory**

A special type of directory consisting entirely of numerical pointers to the first record of each file cataloged in it. Segment directories are usually referenced by programs.

**subdirectory**

A directory that is beneath another directory.

**suffixes, objectname**

See objectname conventions.

**system name**

The name of a computer system on a network. A system name is assigned when a local PRIMOS system is built or configured.

**treename**

See pathname.

volume

A self-sufficient unit of disk storage, including an MFD, a Disk Record Availability Table, and associated files and directories. A volume may occupy a complete disk pack or it may be a partition within a multihead disk pack.

volume name

A sequence of 6 or fewer characters labeling a volume. The name is assigned during formatting (by MAKE). The STATUS DISKS command uses this name in its DISK column to identify the disk. Also called a packname.



# INDEX

# Index

## A

### Access categories,

- as named ACLs, 3-7
- creating, 4-6
- deleting, 4-8
- protecting objects with, 4-7
- replacing the contents of, 4-7

### Access requirements for PRIMOS commands, 3-17

### Access rights,

- add, 4-12
- all, 4-12
- delete, 4-11
- execute, 4-12
- for PRIMOS commands, table, 3-18
- how to list, 4-2
- list, 4-12
- none, 4-12
- owner, 4-11
- protect, 4-11
- read, 4-12
- use, 4-12
- write, 4-12

### Accessing remote systems, 2-9

### Accessing the system, 2-1

### ACL security system, features of, 1-5

### ACL subroutines, 4-13

### ACLs,

- access rights (table of), 3-3
- default protection, 3-10
- default protection from access categories, 3-13
- default protection from specific ACLs, 3-12
- device, 3-4, 3-15
- device, explanation of, 1-11
- explanation of, 1-10
- for origin directory, 2-4
- how to modify, 4-9
- how to set on directories, 4-5
- how to set on files, 4-5
- how to set on segment directories, 4-5
- how to use, 4-1
- introduction to, 3-2
- overlapping access rights, 3-5
- priority, 3-4, 3-15
- priority, explanation of, 1-11
- priority, how to list, 4-3
- specific (unnamed ACLs), 3-6
- standard, 3-4
- types of, 3-4

ACLs (continued)

types of users for, 3-4  
what they look like, 3-5

ADD\_REMOTE\_ID (ARID) command,  
2-10

Adding remote IDs, 2-10

C

CHANGE\_PASSWORD (CPW) command,  
1-8, 2-5

Changing access rights, 4-9

Changing passwords, 1-7, 2-5  
example of, 2-6  
password expires, 1-7, 1-8

Combining default protection,  
specific ACLs, and access  
categories, 3-14

Commands,

ADD\_REMOTE\_ID (ARID), 2-10  
CHANGE\_PASSWORD (CPW), 1-8,  
2-5  
EDIT\_ACCESS (EDAC), 4-9  
LIST\_ACCESS (LAC), 4-2  
LIST\_PRIORITY\_ACCESS (LAC),  
4-4  
LIST\_REMOTE\_ID (LRID), 2-11  
PRIMOS, access requirements  
for, 3-17  
REMOVE\_REMOTE\_ID (RRID), 2-11  
SET\_ACCESS (SAC), 4-4

Commands for using ACLs and  
access categories, 4-1

Completing a work session, 2-8

Computer-generated passwords,  
1-8, 2-7  
and password expiration, 2-7  
example of, 2-7

Controlling access to files and  
directories, 4-4

Counts of failed logins, 2-4

Creating an access category, 4-6

D

Data security, 1-5, 1-10

Default protection for ACLs,  
3-10

Deleting an access category, 4-8

Device ACLs, 3-15  
explanation of, 1-11

Distribution of access rights,  
who controls, 3-15

E

EDIT\_ACCESS (EDAC) command, 4-9  
compared with SET\_ACCESS, 4-10

Examining remote IDs, 2-11

Expiration of password, 2-6

External login programs, 1-9

F

Failed logins, counts of, 2-4

File system security, 3-1

H

Hardware security, 1-4

L

-LIKE, option of SET\_ACCESS, 4-8

LIST\_ACCESS (LAC) command, 4-2

LIST\_PRIORITY\_ACCESS (LPAC)  
command, 4-4

LIST\_REMOTE\_ID (LRID) command,  
2-11

Listing access rights, 4-2

Listing priority ACLs, 4-3

Logging in, 2-2  
alternative form, 2-3

Logging out, 2-8

Login,  
external programs, 1-9  
remote, 1-9

Login passwords, 1-7, 2-2  
changing, 1-7, 2-5

Login security, 1-6  
degrees of, 1-9  
external login programs, 1-9  
overview, 2-1

Logins, counts of failed, 2-4

## M

Multiple project IDs, 1-8

## N

Network security, 1-9

## O

Origin directory, 2-4

Overlapping access rights, 3-5

## P

Password expiration, 2-6

Password expiration and  
computer-generated password,  
2-7

Passwords,  
changing, 1-7  
computer-generated, 2-7

PRIMOS commands, (See also  
Commands)  
access requirements for, 3-17  
table of access requirements,  
3-18

PRIMOS users, 1-3

Priority ACLs, 3-15  
explanation of, 1-11  
listing, 4-3

Privileged users, definition of,  
1-2

Project IDs, 1-8, 2-2

Protecting objects with access  
categories, 4-7

Providing default protection from  
access categories, 3-13

Providing default protection from  
specific ACLs, 3-12

## R

Remote IDs,  
adding, 2-10  
examining, 2-11  
removing, 2-11

Remote systems,  
access to, 2-9

REMOVE\_REMOTE\_ID (RRID) command,  
2-10

Removing remote IDs, 2-11

Replacing the contents of an  
access category, 4-7



Reverting to default protection,  
4-9

S

Security,  
    changing login passwords, 1-7  
    data, 1-10  
    file system, 3-1  
    hardware, 1-4  
    login, 1-6  
    network, 1-9  
    project IDs, 1-8  
    software, 1-4

Security features,  
    ACL security system, 1-5  
    data security, 1-5  
    user profile database, 1-5

Security Monitor, discussion of,  
1-11

SET\_ACCESS (SAC) command, 4-4  
    compared with EDIT\_ACCESS,  
    4-10

Setting access rights to match  
    the rights on other objects,  
4-8

Setting ACLs,  
    on directories, 4-5  
    on files, 4-5  
    on segment directories, 4-5

Software security, 1-4

Specific ACLs and access  
    categories, 3-6

System access, overview of, 1-1

System Administrator,  
    responsibilities of, 1-2

System security, types of, 1-4

U

Unauthorized users,  
    counts of failed logins, 1-3

User IDs, 1-6  
    obtaining, 2-1  
    with multiple projects, 1-8

Users,  
    IDs, 1-6  
    PRIMOS, 1-3  
    privileged, 1-2  
    System Administrator, 1-2  
    unauthorized, 1-3

W

What an ACL looks like, 3-5



# SURVEY

## READER RESPONSE FORM

DOC10130-1LA

Security Features User's Guide

Your feedback will help us continue to improve the quality, accuracy, and organization of our publications.

1. How do you rate this document for overall usefulness?

☐ *excellent*      ☐ *very good*      ☐ *good*      ☐ *fair*      ☐ *poor*

2. What features of this manual did you find most useful?

---

---

---

---

---

---

3. What faults or errors in this manual gave you problems?

---

---

---

---

---

---

4. How does this manual compare to equivalent manuals produced by other computer companies?

☐ *Much better*                      ☐ *Slightly better*                      ☐ *About the same*  
☐ *Much worse*                      ☐ *Slightly worse*                      ☐ *Can't judge*

5. Which other companies' manuals have you read?

---

---

Name: \_\_\_\_\_ Position: \_\_\_\_\_

Company: \_\_\_\_\_

Address: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ Postal Code: \_\_\_\_\_

First Class Permit #531 Natick, Massachusetts 01760

# BUSINESS REPLY MAIL

Postage will be paid by:



Attention: Technical Publications  
Bldg 10  
Prime Park, Natick, Ma. 01760



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

