

IRIX Commercial Security Pak™ User's Guide

Document Number 007-3267-001

CONTRIBUTORS

Written by Jeffrey B. Zurschmeide

Production by Lorrie Williams

Engineering contributions by Nelson Bolyard, Paul Close, Ana Maria DeAlvarez, Ellen Desmond, Bill Kawakami, Sue Hui Liu, Gary Lowell, Alison Gabriel-Reilly, Sandra Romero, Casey Schaufler, and Aaron Schuman

St Peter's Basilica image courtesy of ENEL SpA and InfoByte SpA. Disk Thrower image courtesy of Xavier Berenguer, Animatica.

© 1997 Silicon Graphics, Inc.— All Rights Reserved

The contents of this document may not be copied or duplicated in any form, in whole or in part, without the prior written permission of Silicon Graphics, Inc.

The IRIX Commercial Security Pak includes documentation and software developed at the Massachusetts Institute of Technology, which includes this copyright information:

Copyright © 1990-1997 by the Massachusetts Institute of Technology.

Export of software employing encryption from the United States of America may require a specific license from the United States Government. It is the responsibility of any person or organization contemplating export to obtain such a license before exporting.

WITHIN THAT CONSTRAINT, permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of M.I.T. not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. M.I.T. makes no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

The IRIX Commercial Security Pak includes documentation and software developed at the University of California at Berkeley, which includes this copyright notice:

Copyright ©1983 Regents of the University of California. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. All advertising materials mentioning features or use of this software must display the following acknowledgement:

This product includes software developed by the University of California, Berkeley and its contributors.

4. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

Permission is granted to make and distribute verbatim copies of the kerberos portions of this manual provided the copyright notices and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided also that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure of the technical data contained in this document by the Government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013 and/or in similar or successor clauses in the FAR, or in the DOD or NASA FAR Supplement. Unpublished rights reserved under the Copyright Laws of the United States. Contractor / manufacturer is Silicon Graphics, Inc., 2011 N. Shoreline Blvd., Mountain View, CA 94043-1389.

Silicon Graphics, IRIS and the Silicon Graphics logo are registered trademarks and IRIX, Commercial Security Pak and IRIS InSight are trademarks of Silicon Graphics, Inc. Sun and RPC are registered trademarks and NFS is a trademark of Sun Microsystems, Inc. The X Window System is a trademark of Massachusetts Institute of Technology. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company, Ltd.

Contents

List of Tables vii

About This Guide ix

Who Should Read This Guide x

Accompanying Documentation x

What This Guide Contains xi

Conventions Used in This Guide xi

Additional Resources xii

The Silicon Graphics End User Documentation xii

Reference Pages xiii

Release Notes xiv

IRIX Help System xiv

The Silicon Graphics World Wide Web Site xiv

1. Introduction to the Commercial Security Pak 1

Commercial Security Pak Overview 1

Definition of a Trusted System 1

Reasons to Use The Commercial Security Pak 2

What You See and What You Don't 3

Commercial Security Pak Features 4

Identification and Authentication 4

Capabilities 5

Discretionary Access Control 5

System Audit Trail 6

Object Reuse Policy 6

2. Understanding System Access Control 7

Discretionary Access Control 7

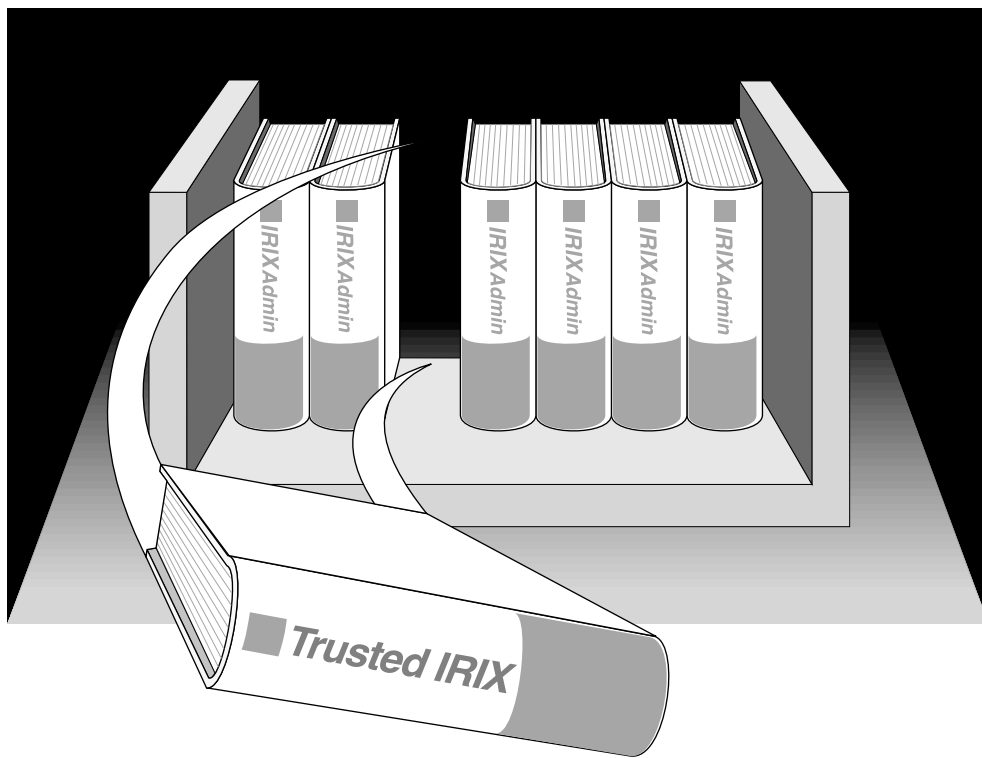
Using DAC 7

- Access Control Lists 12
 - Setting Directory Default ACLs With chacl 13
 - Text Form Representation of ACLs 13
- Capabilities 17
- Passwords Under the Commercial Security Pak 18
 - System-Generated Passwords 19
 - Password Aging 19
- 3. Understanding Auditing 21**
 - System Audit Trail 21
- 4. Using CSP-Kerberos 23**
 - Introduction to CSP-Kerberos 23
 - CSP-Kerberos Tickets 24
 - CSP-Kerberos Principals 24
 - CSP-Kerberos Tutorial 25
 - Setting Up to Use CSP-Kerberos 25
 - Ticket Management 25
 - CSP-Kerberos Password Management 29
 - CSP-Kerberos Applications 33
 - Overview of Kerberized Programs 33
- 5. Programming in a Trusted Environment 43**
 - Programming Guidelines 43
 - Commercial Security Pak System and Library Calls 43
- A. Computer Security Terms 47**
 - Index 75

List of Tables

Table i	Outline of Reference Page Organization	xiii
Table 5-1	Commercial Security Pak System and Library Calls	44

About This Guide



“About This Guide” includes brief descriptions of the contents of this guide and an explanation of typographical conventions used, and refers you to additional sources of information you might find helpful.

This guide explains how to use the Commercial Security Pak™ software with Silicon Graphics® workstations and servers. It provides descriptions of those user tasks that are specific to this software.

If you have a graphics workstation, you will need to be familiar with the user documentation of the standard IRIX™ operating system, on which this product is based. See the SGI_EndUser bookshelf in your IRIS InSight™ online documentation system.

Who Should Read This Guide

You should read this guide if you have never used a secure system before or if you are encountering the Commercial Security Pak for the first time.

The *Commercial Security Pak User's Guide* is written for "end users" on Commercial Security Pak systems. Frequently, people who would consider themselves end users find themselves performing advanced administrative tasks. For those individuals, the *Commercial Security Pak Administration Guide* has been prepared to help both the new and experienced administrator successfully perform all operations necessary to configure and maintain security on Commercial Security Pak systems.

Accompanying Documentation

You have the entire set of standard IRIX documentation online in addition to this release. The following documents are also included as part of this release of Commercial Security Pak:

Commercial Security Pak Administrator's Guide

This manual describes how to administer the security features at your site.

Commercial Security Pak Release Notes

This manual describes how to install the release and lists any known problems with the implementation.

What This Guide Contains

This guide contains the following chapters:

Chapter 1, "Introduction to the Commercial Security Pak"

Provides an overview of the Commercial Security Pak.

Chapter 2, "Understanding System Access Control"

Provides a comprehensive overview of the responsibilities of the user and the operating system features to be used.

Chapter 3, "Understanding Auditing"

Describes the auditing features and the user's responsibilities with respect to an audited environment.

Chapter 4, "Using CSP-Kerberos"

Describes the tasks and procedures necessary to successfully use the CSP-Kerberos authentication system.

Chapter 5, "Programming in a Trusted Environment"

Provides information on programming practices in a trusted environment.

Appendix A, "Computer Security Terms"

Provides a glossary of computer security terms and concepts used in these guides and elsewhere.

Conventions Used in This Guide

These type conventions and symbols are used in this guide:

Bold Literal command-line arguments (options/flags).

Italics Executable names, filenames, glossary entries, IRIX commands, manual/book titles, new terms, program variables, tools, utilities, variable command-line arguments, and variables to be supplied by the user in examples, code, and syntax statements

Fixed-width type

Error messages, prompts, and onscreen text

Bold fixed-width type

User input, including keyboard keys (printing and nonprinting); literals supplied by the user in examples, code, and syntax statements (*see also* <>)

ALL CAPS Environment variables, operator names, directives, defined constants, macros in C programs

"" (Double quotation marks) Onscreen menu items and references in text to document section titles

() (Parentheses) Following function names—surround function arguments or are empty if the function has no arguments; following IRIX commands—surround reference page (man page) section numbers

[] (Brackets) Surrounding optional syntax statement arguments

This guide uses the standard UNIX convention for referring to entries in IRIX documentation. The entry name is followed by the section number in parentheses. For example, *rcp*(1C) refers to the *rcp* online reference page.

Additional Resources

For easy reference, here is a list of the guides and resources provided with your system and the specific focus and scope of each. You can see the guides by invoking the InSight library on your desktop or through the system toolchest, or through the *iiv*(1) command from a shell window.

The Silicon Graphics End User Documentation

Your IRIS InSight documentation library contains a bookshelf titled *SGL_EndUser*. This bookshelf contains the end user documentation for your system. Some of these books include:

- *IRIS Essentials or Desktop User's Guide*
- *IRIS Glossary of Terms*
- *IRIS Utilities Guide*
- *Personal System Administration Guide*
- *Media Control Panels User's Guide*

These books have been written for standard IRIX. Where they differ from information in this book and in the *Commercial Security Pak Administration Guide*, the Commercial Security Pak books should be considered authoritative.

Reference Pages

The IRIX reference pages (called “man” or “manual” pages) provide concise reference information on the use of IRIX commands, subroutines, and other elements that make up the IRIX operating system. This collection of entries is one of the most important references for an administrator. Generally, each reference page covers one command, although some reference pages cover several closely related commands.

The IRIX reference pages are available online through the *man* command. To view a reference page, use the *man* command at the shell prompt. For example, to see the reference page for *diff*, enter:

```
man diff
```

It is a good practice to print those reference pages you consistently use for reference and those you are likely to need before major administrative operations and keep them in a notebook.

Each command, system file, or other system object is described on a separate page. The reference pages are divided into seven sections, as shown in Table i. When referring to reference pages, this document follows a standard UNIX® convention: the name of the command is followed by its section number in parentheses. For example, *cc*(1) refers to the *cc* reference page in Section 1.

Table i shows the reference page sections and the types of reference pages that they contain.

Table i Outline of Reference Page Organization

Type of Reference Page	Section Number
General Commands	(1)
System Calls and Error Numbers	(2)
Library Subroutines	(3)
File Formats	(4)

Table i Outline of Reference Page Organization

Type of Reference Page	Section Number
Miscellaneous	(5)
Demos and Games	(6)
Special Files	(7)

Release Notes

Release notes provide specific information about the current release. Exceptions to the documentation are found in this document. Release notes are available online through the *relnotes* command. Each optional product or application has its own set of release notes. The *grelnotes* command provides a graphical interface to the release notes of all products installed on your system.

IRIX Help System

Your IRIX system comes with a help system. This system provides help cards for commonly-asked questions about basic system setup and usage. The command to initiate a help session is *desktophelp*.

The Silicon Graphics World Wide Web Site

The Silicon Graphics World Wide Web (WWW) presence has been established to provide current information of interest to Silicon Graphics customers. The following URL addresses are accessible to most commercially available Web browsers on the Internet:

<http://www.sgi.com>

The Silicon Graphics general Web server, Silicon Surf

<http://www.mips.com>

The Silicon Graphics MIPS division server

<http://www.studio.sgi.com>

The Silicon Studio server

<http://www.ids.sgi.com>

The InterActive Digital Solutions server

<http://www.alias.com>

The Alias server

<http://www.sgi.com/Technology/TechPubs>

The Silicon Graphics Technical Publications Library

From these sites you can find all the Silicon Graphics Web-published information.

Introduction to the Commercial Security Pak

This *User's Guide* has been designed to introduce you to working with secure systems. In particular, it addresses your first interactions with Silicon Graphics Commercial Security Pak and gives you some pointers on how to maintain system integrity with regard to security. It also describes the various modifications and additions made to standard IRIX that make this system secure.

Commercial Security Pak Overview

This chapter provides a helping hand in learning how to use the system for day-to-day tasks. To this end, some explanation of security procedures and mechanisms must necessarily be provided.

Definition of a Trusted System

Operating systems that attempt to provide a secure environment for the development and storage of sensitive information are known as *trusted* systems. In an abstract sense, no system is ever perfectly secure from harm, so the term *trusted* rather than *secure* is more accurate. A trusted system can be thought of as any system that fits the following criteria:

- The system allows all users to do their ordinary and necessary work without difficulty.
- The system enforces the security policy deemed by the management to be appropriate to the site.

The first criterion is the most important. If users are unable to do their ordinary and necessary work, they either circumvent the security measures or they do not use the system at all. In either case, the trusted system is rendered useless. Many users are concerned that they cannot do their work in a trusted environment. A good site administration plan will structure a trusted system so that the user is relatively

unaffected by its functioning. Ideally, users should be able to perform all their tasks and never see the trusted features of the operating system.

The second criterion requires that the system have adequate security features to enforce the site security policy set forth by the management. The Commercial Security Pak offers a variety of security measures that are sufficient to satisfy most sites. These measures are:

Access Control Lists

An Access Control List allows the owner of a file or directory to make a specific list of users and user groups and the specific permissions each one is allowed to the file or directory.

Auditing The Audit subsystem allows your Site Security Officer to keep a precise log of all system activity.

Capabilities Capabilities allows your Site Security Officer to specify specific individuals to be given the capability to execute sensitive system files. This eliminates the risk of having a single all-powerful **root** account.

Discretionary Access Control

This is the standard IRIX system of file and directory permissions.

Identification and Authentication

The Commercial Security Pak has improved user identification and authentication facilities that ensure the integrity of system passwords and help to ensure that only authorized users are granted access to the system.

Reasons to Use The Commercial Security Pak

This security package is a significant improvement over conventional operating systems derived from the standard UNIX kernel. While trusted operating systems necessarily add some level of extra effort to user interactions, the system need not be hostile to the average, or even novice user.

This security package is designed to address the three fundamental issues of computer security: policy, accountability, and assurance. By fully addressing these areas, the system becomes a trustworthy base for secure development and business. Since the nature of a trusted system is already constrained, little must be trusted beyond the system itself. When you run your application programs on the system, you have a reasonable certainty that your applications are free from corruption and safe from intruders.

What You See and What You Don't

A number of features of the Commercial Security Pak are invisible to the user most of the time. These may occasionally affect your environment and so it is best to mention them here.

Access Control Lists (ACL)

ACLs make it much easier to specify who has access to your files. Some programs are unaware of ACLs; the implementation does not generally require that programs be aware of them. A file's ACL may be set when it is created, if the containing directory has a default ACL. See "Access Control Lists" in Chapter 2 for information on ACLs.

Auditing

Auditing is constantly in effect on most Commercial Security Pak systems. This is not a cause for user concern, as the purpose of the audit trail is to discern intentional violations of security by malicious intruders, not to spy on legitimate users. See Chapter 3, "System Audit Trail" for information on auditing.

Capabilities

It is possible to grant a user certain capabilities that have formerly been reserved for the superuser. It is also possible that userid 0 (**root**) need not have special privilege. See "Capabilities" in Chapter 2 for more information.

CSP-Kerberos

The CSP-Kerberos authentication system is provided and may be in use on your system. See Chapter 4, "Using CSP-Kerberos" for information on CSP-Kerberos.

Password Aging

The strictures on passwords in the Commercial Security Pak are not limited to the content of the password. The Administrator can set a minimum and a maximum amount of time for the use of a given password. It is quite possible that you would wish to log in and be unable to do so because your password had expired if you ignored the warnings to change it. See "Password Aging" in Chapter 2 for more information.

Password Generation

The Commercial Security Pak is equipped with a password generation program. When you attempt to change your password, this program presents you with several options for your new password. You must choose one of these passwords or choose not to change your password.

This feature can be defeated by your Administrator, in which case you are free to select your own password, subject to certain triviality tests. See "System-Generated Passwords" in Chapter 2 for more information.

Commercial Security Pak Features

The distinguishing difference between trusted and nontrusted systems is the security-enhanced feature set.

Every trusted system has a Trusted Computing Base (TCB). The TCB is the system hardware, the operating system program itself, and the commands, utilities, tools, and system files that are known to be secure. This set of hardware, files, and programs is the "trusted" part of a trusted system.

Within the TCB, there are *subjects* and *objects*. A subject is any active force on the system, such as a user's shell process, the audit daemon, or the operating system itself. An object is any passive resource on the system, such as a text file, a page of memory, or a piece of system hardware.

The Commercial Security Pak is fully configurable to your site's needs. You are free to select your own capabilities and Access Control Lists, and your own system of password protection, and CSP-Kerberos authentication for network services.

Identification and Authentication

The Identification and Authentication (I&A) mechanism controls user access to the system. In common terms, the I&A mechanism is the login procedure. This subsystem is always active if the system is running, and it is impossible to have any contact with the system without first logging in through the I&A system.

The improved I&A facilities of the Commercial Security Pak allow the administrator to be certain that the people on the system are authorized users and that private password integrity is maintained to the highest possible levels.

Passwords in the Commercial Security Pak

Under the Commercial Security Pak, encrypted passwords are stored separately from other user identification information. This separate location is hidden from normal user access, so the process of a systematic “dictionary encryption” hunt for a password is precluded. User clearance information is also stored in a hidden, or shadow, file. Under the Commercial Security Pak, the */etc/passwd* file does not contain the encrypted password; only the shadow password file contains that information.

In response to basic security requirements, passwords can be generated automatically for the users under the Commercial Security Pak. System administrators can configure the system to require this feature for every password change, or it can be an option for the user. The complexity, length, and character combinations required of passwords can also be configured. For example, it is possible to require users to mix control characters into their passwords. It is also possible to check and reject passwords that can be found in a dictionary, proper names, place names, and technical words associated with computers or the current project. System administrators can also require passwords to be changed on a regular basis.

Capabilities

Your Site Security Officer can require a user-specific capability requirement for access to system files. A capability requirement can be imposed on any file—without the corresponding capability endorsement, no user can access the file. The capability endorsements are made on a user-by-user basis. This allows the Site Security Officer to allow only certain users to access system files. The reason for this facility is to allow the Site Security Officer to designate certain users to perform system administration tasks, to fulfill the system administration roles without requiring a special login account for the role.

Discretionary Access Control

The Commercial Security Pak supports the POSIX P1003.1e Draft15 definition for Access Control Lists (ACLs). This draft standard provides for traditional file permission bits working in concert with the more versatile ACLs. DAC permissions are defined by the user who owns the file in question. For example, if a user has a personal file in his or her home directory, that user can set the DAC permissions to allow no other users on the system, except **root**, to view, copy, or edit that file. Default DAC permissions for newly created files are set via the *umask(1)* command.

Typically, DAC permissions are set to allow access on all but personal files.

Default DAC permissions for newly created files depend on the *umask* and on any default ACL entries found in the containing directory. Default DAC permissions for newly created sockets are specified with the *setpsoacl(2)* system call.

Access Control Lists

Access Control Lists allow users to specify on a user-by-user basis who may access their files and directories. The purpose of this feature is to provide a finer level of control than allowed through traditional discretionary access control.

System Audit Trail

The System Audit Trail provides a means for the system administrator to oversee each important event taking place on the system. The Audit Trail is useful for tracking changes in sensitive files and programs and for identifying inappropriate use of the system.

Object Reuse Policy

To preclude accidental disclosure of data, display memory and long-term data storage are subject to an object reuse policy and implementation. For example, all system memory is always automatically cleared before it is allocated to another program. Surrendered disk space is also cleaned prior to reallocation.

Understanding System Access Control

Access control allows administrators to set up policies and accounts that allow each user full access to the files and resources he or she needs, but not to other information and resources not immediately necessary to perform assigned tasks.

Discretionary Access Control

Discretionary access control is the name of the standard UNIX system of access permissions that allow the user to control access to files, directories, and other system resources. The owner of any file or other system object can control access to that object, even by those with equal or dominating clearances, by setting the DAC permissions. Additionally, Access Control Lists (ACLs) provide a finer granularity of control than provided by the traditional permission bits.

Using DAC

The Commercial Security Pak divides permissions into three categories, and users into three relative groups. The three categories of permissions are *read*, *write*, and *execute*. They are denoted as “r” for read, “w” for write, and “x” for execute in long listings of files. To get a long listing, type the following command at your system prompt in any directory:

```
ls -l
```

The command shows you more information about the files in the directory than an ordinary listing. Along with the permission information, the *ls -l* command lists the owners of the files, the size of the files, and the date they were last modified. Adding the **-D** command line option to *ls* displays the Access Control List for the file or directory as well.

Read permission allows you to look at the contents of a file. Write permission allows you to make changes to or remove a file. Execute permission allows you to run the file as a command from your shell prompt.

The three relative groups are the owner of the file, the owner's group, and every other user. If you get a long listing of a directory, you see that the permissions field looks something like this:

```
-rw-r--r--
```

Each character is significant in the permissions listing. A dash in any place means that no permission is granted, and the actions associated with that permission are denied. However, in the leftmost place, the contents of that space describes whether the file is an ordinary file, a directory, or special device file. If there is a dash in that place, the file in question is an ordinary file. If it is a directory, a "d" appears in that space. If the file is a block special device file, a "b" appears in the space, and if the file is a character special device file, a "c" appears there. For more complete information, consult the *ls(1)* reference page or the */usr/include/sys/stat.h* file.

Directory Permissions

Directories use the same permissions as files, but their meanings are slightly different. For example, read permission on a directory means that you can use the *ls* command to look at the contents of that directory. Write permission allows you to add, change, or remove files in that directory. (However, even though you may have write permission in that directory, you must also have write permission on the individual files to change or remove them, unless you own the directory.) Finally, execute permission on a directory allows you to use the *cd* command to change directories into that directory.

File Permissions

The first series of three places in the permissions field describes the permissions for the owner of the file. Here is an example of a long listing for a file:

```
-rwx-----+ 1 owner grp 6680 Apr 24 16:26 shell.script
```

The file is not a directory, so the first space is blank. The characters *rwx* indicate that the owner of the file, *owner*, has read, write, and execute permission on this file. The second series of three spaces describes permissions for the owner's group. In this case, the group is *grp*. If permissions for this file were slightly different, like this:

```
-rwxr-x---+ 1 owner grp 6680 Apr 24 16:26 shell.script
```

then any member of the group *grp* could read or execute the file but could not change it or remove it. All members of group *grp* can share a pool of files that are individually owned. Through careful use of group read and write permissions, you can create a set of source files that are owned by one person, but any group member can work on them.

The third series of spaces provides for all other users on the system and is called the public permissions.

The plus sign (+) at the end of the permission string indicates that an Access Control List is in effect for this file. Use the *ls -D* command to view the Access Control List for the file. Complete discussion of Access Control Lists is found in the section titled "Access Control Lists."

Individual groups can tailor their working set of files by using file permissions and Access Control Lists to share some files. A file that is set to be readable by any user on the system is called *publicly readable*.

Here is a long listing of a sample *Projects* directory:

```
total 410
drw-----+ 1 owner grp 48879 Mar 29 18:10 critical
-rw-r--r-- 1 owner grp 1063 Mar 29 18:10 meeting.notes
-rw-rw-rw- 1 owner grp 2780 Mar 29 18:10 new.deal
-rwxrwxrwx 1 owner grp 8169 Jun 7 13:41 new.items
-rw-rw-rw- 1 owner grp 4989 Mar 29 18:10 outside.response
-rw----- 1 owner grp 23885 Mar 29 18:10 project1
-rw-r----- 1 owner grp 3378 Jun 7 13:42 saved_mail
-rw-r--r-- 1 owner grp 2570 Mar 29 18:10 schedules
-rwxrwxr-x+ 1 owner grp 6680 Apr 24 16:26 shell.script
```

The files have varying permissions. Some are read/write only for the owner, some can be read only by members of the owner's group, and some can be read, changed, or removed by anybody. The shell script can be executed publicly, subject to its ACL, and the *critical* directory is also subject to an ACL.

Changing Permissions

You change the permissions on a file by means of the *chmod(1)* command. You can use *chmod* only to change files that you own. Generally, you use this command to protect files you want to keep secret or private, to protect private directories, and to grant permissions to files used by others. The command to restrict access to a file or directory to yourself only is:

```
chmod 600 filename
chmod 700 directoryname
```

Other permissions may be added by using the *chmod* command with the letter associated with the permission. For example, the command to add general write permission to a file is:

```
chmod +w filename
```

For more examples, see the *chmod(1)* reference page.

To set or change an Access Control List, use the *chacl*(1) command:

```
chacl acl_entry [ , acl_entry ] . . .
```

For more information on *chacl* and the ACL entry syntax, see the *chacl*(1) reference page and the section of this chapter titled “Text Form Representation of ACLs.”

Setting Permissions With *umask*

You can decide what default permissions your files have by means of the *umask* command. You place this command in your *.cshrc*, *.profile*, or *.login* file. The *umask*(1) reference page is also available for more information. By changing the setting of your *umask*, you can alter the default permissions on your files and directories to any available DAC permission.

A drawback to the *umask* command is that it makes every file you create receive the same permissions. For most purposes, you want the files you create to be accessible by the members of your group. For example, if an individual is suddenly called away and another person must take over that person’s portion of a project, the source files must be accessible by the new user. However, the personal files you keep in your home directory sometimes need to be private, and if you set your *umask* to allow group read and write privileges, any member of the group can access your personal files. But mechanisms are available to prevent this access. For example, you can create a directory of private files and alter the permissions on that directory with the *chmod* command to restrict all but your own access. Then it would not matter that the files were readable, because no other user would be allowed into the directory.

You can also use the *find* command to change all the files in your home directory to your chosen permission automatically at your convenience. You can set up your account so that this action happens every time you log out.

The *umask* command is an important part of DAC. It allows you to maintain security and still allow convenient access to your files. To set up your account to allow group read and write privileges and no other privileges, place this line in your *.cshrc* or *.profile* file:

```
umask 007
```

This makes every file you create have the following permissions:

```
-rw-rw----
```

With your *umask* set to 007, directories that you create have the following permissions:

```
drwxrwx---
```

In other words, you will have full use of the file or directory, and your group will have full use. No other user, except the Superuser (**root**), has access to your files.

Access Control Lists

Access Control Lists (ACLs) are a part of the Discretionary Access Control Features of your Commercial Security Pak system. An ACL works in the same way as standard file permissions, but it allows you to get a finer level of control over who may access the file or directory than standard permissions allow. ACLs allow you to specify file permissions on a user-by-user basis.

Every system file or directory has an Access Control List that governs its discretionary access. This ACL is referred to as the access ACL for the file or directory. In addition, a directory may have an associated ACL that governs the initial access for files and subdirectories created within that directory. This ACL is referred to as a default ACL. A user who wishes to gain access to the files in a directory must be on both ACLs and must be allowed by IRIX standard file permissions to successfully gain access. If you have not created an access ACL for a file, the default ACL serves both ACL functions. Note that the ACL on a file or directory also acts as an upper limit to the file permissions that can be set automatically with *umask*.

Hereafter in this section, directories are treated as files, and where the term file is used, it also applies to directories.

An ACL is stored in the same way that standard file permissions are stored—as an attribute of the file or directory. To view the ACL of a file, use the **-D** option to *ls(1)* as shown in this example:

```
ls -D /usr/people/ernie/testfile
```

This produces output similar to this:

```
testfile [user::rwx ,user:332:r-- ,user:ernie:rw-]
```

The above example shows full permissions for the owner in the first entry on the line, sets read permission for user ID 332 in the second entry, and sets read/write permission for

the user account ernie. The format of an ACL entry is discussed in the section titled “Text Form Representation of ACLs.”

To set or change an ACL, use the *chacl*(1) command:

```
chacl acl_entry [ ,acl_entry ] . . .
```

An ACL consists of a set of ACL entries. An ACL entry specifies the access permissions on the associated file for an individual user or a group of users. The order of internal storage of entries within an ACL does not affect the order of evaluation. In order to read an ACL from an object, a process must have read access to the file. In order to create or change an ACL , the process must own the file.

Setting Directory Default ACLs With *chacl*

To set a default ACL for the current directory and all its files and subdirectories, use the command:

```
chacl -d acl_entry [ ,acl_entry ] . . .
```

For information on the format of an ACL entry, see the section titled “Text Form Representation of ACLs.”

Text Form Representation of ACLs

This section defines the long and short text forms of ACLs. The long text form is defined first to give a complete specification with no exceptions. The short text form is defined afterwards because it is specified relative to the long text form.

Long Text Form for ACLs

The long text form is used for either input or output of ACLs and is defined as follows:

```
acl_entry [ ,acl_entry ] . . .
```

Though it is acceptable to place more than one entry on a physical line in a file, placing only one entry per line improves readability.

Each entry contains one ACL statement with three required colon-separated fields and an optional comment:

entry tag type : entry qualifier : discretionary access permissions # comment

Comments may be included with any entry. If a comment starts at the beginning of a line, then the entire line is interpreted as a comment. The first field must always contain the ACL entry tag type.

One of the following ACL entry tag type keywords must appear in the first field:

- user* A *user* ACL entry specifies the access granted to either the file owner or to a specified user account.
- group* A *group* ACL entry specifies the access granted to either the file owning user group or to a specified user group.
- other* An *other* ACL entry specifies the access granted to any process that does not match any user, group, or implementation-defined ACL entries.
- mask* A *mask* ACL entry specifies the maximum access which can be granted by any ACL entry except the user entry for the file owner and the *other* entry.

The second field contains the ACL entry qualifier (referred to in the remainder of this section as simply *qualifier*).

The following qualifiers are defined by default:

- uid* This qualifier specifies a user account name or a user ID number.
- gid* This qualifier specifies a user group name or a group ID number.
- empty* This qualifier specifies that no *uid* or *gid* information is to be applied to the ACL entry. The entry applies to the file owner only. An empty qualifier is represented by an empty string or by white space.

The third field contains the discretionary access permissions that are to apply to the user or group specified in the first field. The following symbolic discretionary access permissions are recognized in ACLs:

- Read access
- Write access
- Execute/search access
- No access

The discretionary access permissions field must contain exactly one each of the following letters in the following order. Any or all of these may be replaced by the no access character (a dash):

1. r
2. w
3. x

A user entry with an empty qualifier specifies the access granted to the file owner. A user entry with a *uid* qualifier specifies the access permissions granted to the user name matching the *uid* value. If the *uid* value does not match a user name, then the ACL entry specifies the access permissions granted to the user ID matching the *uid* value.

A group entry with an empty qualifier specifies the access granted to the default user group of the file owner. A group entry with a *gid* qualifier specifies the access permissions granted to the group name matching the *gid* value. If the *gid* value does not match a group name, then the ACL entry specifies the access permissions granted to the group ID matching the *gid* value. The *mask* and other entries contain an empty qualifier. A hashmark (#) starts a comment on an ACL entry. A comment may start at the beginning of a line, or after the required fields and after any custom-defined, colon-separated fields. The end of the line denotes the end of the comment.

If an ACL entry contains permissions that are not also contained in the *mask* entry, then the output text form for that entry must be displayed as described above followed by a hashmark (#), the string effective:, and the effective file access permissions for that ACL entry.

White space is permitted (but not required) in the entries as follows:

- at the start of the line
- immediately before and after a colon (:) separator
- immediately before the first hashmark (#) comment character
- at any point after the first hashmark (#) comment character.

Comments have no effect on the discretionary access check of the object with which they are associated.

Here is an example of a correct long text form ACL for a file:

```
user::rwx,user:332:r--,user:ernie:rw-
```

The above example sets full permissions for the owner (the first entry on the line) sets read permission for user ID 332 in the second entry, and sets read/write permission for the user account ernie.

Here are some examples with comments:

```
group:10:rw- # User Group 10 has read/write access
other::--- # No one else has any permission
mask::rw- # The maximum permission except for the owner is read/write
```

Short Text Form for ACLs

The short text form is used only for input of ACLs and is defined as follows:

```
acl_entry[ ,acl_entry ]...
```

Though it is acceptable to place more than one entry on a physical line in a file, placing only one entry per line improves readability.

Each line contains one ACL entry, as defined in “Long Text Form for ACLs,” with two exceptions. The ACL entry tag type keyword must appear in the first field in either its full unabbreviated form or its single letter abbreviated form.

The abbreviation for user is *u*, the abbreviation for group is *g*. The abbreviation for other is *o*, and the abbreviation for mask is *m*.

There are no exceptions for the second field in the short text form for ACLs.

The discretionary access permissions must appear in the third field in either absolute symbolic form or relative symbolic form. The relative symbolic form must be preceded by a plus sign (+) to indicate additional access or a caret (^) to indicate that access is to be removed. The relative symbolic string must be at least one character.

The symbolic string contains at most one each of the following letters in any order:

- r
- w
- x

For example, the short form should look very similar to the following:

```
u: :rwx # The file owner has complete access
u:332:+r # User Acct 332 has read access only
g:10:rw- # User Group 10 has read/write access
u:653:^w # User Acct 653 (who is in group 10) has read access only
o::--- # No one else has any permission
m::rw- # The maximum permission except for the owner is read/write
```

Capabilities

Capabilities are privileges assigned to specific accounts to allow those accounts to perform operations formerly reserved for the superuser (**root**). To maintain the principle of least privilege, the capabilities of the superuser account have been subdivided into various capabilities, which can be assigned to separate individual accounts. The corresponding capability is placed on sensitive executable files and programs on your system. The account capability and the executable capability must be compatible for the user to execute the program. For a more technical discussion, see the *capabilities(4)* reference page or Chapter 4 of the *Commercial Security Pak Administrator's Guide*.

The fundamental purpose of capabilities is to allow administrators to perform system administration from standard login accounts without requiring the use of superuser or other sorts of privileged accounts. A capability may be granted to any user account, and a corresponding capability attached to only those system objects that the owner of the account has a legitimate need to use. This follows the trusted systems principle of least privilege—using the lowest possible promotion of privileges necessary to get the job done. Capabilities implement least privilege both by limiting the number of users privileged to perform various tasks and by limiting the privilege to just that program, or section of code within a program, necessary to perform the proper action.

Passwords Under the Commercial Security Pak

This section describes the password mechanisms of the Commercial Security Pak. Passwords are the first line of defense of a trusted system. As a user, it is your responsibility to protect the privacy of your password at all times. Follow these rules when dealing with your password:

- Never give your password to another user, or allow another user to “borrow” your account.
- Never keep your password written down anywhere near your system.
- Always commit your password to memory. If you forget it, the Administrator can change it for you.

The Commercial Security Pak contains facilities to generate passwords for users and these facilities are configured to work by default. If your site has changed the configuration to allow you to select your own passwords, follow these rules when choosing your password:

- Never choose a password that could be guessed by someone who knows personal information about you. For example, if someone steals your wallet with the intent of finding out information about you, make certain that your password is not anything related to something someone might find in your personal information, such as variations on your name or the name of a friend or family member.
- Always use a random mix of printable characters, control characters, punctuation marks, and numerals when selecting a password.
- Each password must have at least six characters. However, only the first eight characters are significant.
- The password must contain at least two alphabet characters and one numeral character.
- The password must not be related to the user’s login name. Any reversing or circular shift of the characters in the login name is not allowed. For the purposes of this test, capital letters are assumed to be equivalent to their lowercase counterparts.
- The password must have at least three characters difference from the previous password. For the purposes of this test, capital letters are assumed to be equivalent to their lowercase counterparts.

System-Generated Passwords

The Commercial Security Pak supports mandatory password generation. The default generator presents the user with five selected passwords, and the user is free to accept or reject any of these. If the user does not accept any of the offered passwords, he or she may press the Enter key and the system presents a new set of password choices.

Password Aging

IRIX supports password aging. Password aging is defined as being able to set a minimum and maximum lifetime for passwords. Password aging is a very useful feature. By limiting the amount of time a password may be in use, you limit the amount of time a potential intruder has to crack your password. By enforcing a minimum lifetime, you prevent lazy users from simply changing their password briefly and then returning to their usual password immediately.

If a user does not change the password within the specified time period, the account is automatically locked. Any user can place the following line in their *.login* or */.profile* files to show notification when password expiration is imminent:

```
showpwage username
```

By default, *showpwage(1)* notifies the user only if the password is within seven days of expiration. This default can be changed with the *-d* flag. See the *showpwage(1)* reference page for a complete description of this command.

Generally, the only time that an account becomes locked is when the user is away for an extended period of time. But once locked, an account can be unlocked only by the superuser. The user should choose a new password the next time he or she logs in.

Understanding Auditing

This chapter describes the System Audit Trail for the user. There is no interface to allow users to alter or read the audit trail; it is accessible only to the audit administrator. This chapter explains what is happening within the audit system and how it applies to the ordinary user.

System Audit Trail

The System Audit Trail (SAT) is a subsystem that allows the site administrators to make a record of all system activity. The ongoing record of system activity shows general trends in system usage, and also violations of the security policy. The site administrators can monitor all system activity through the audit trail. There are many different types of activities that take place on a trusted computer system. There are login attempts, file manipulations, use of devices (such as printers and tape drives), and administrative activity. All of these activities can be logged and reviewed through the System Audit Trail.

It is vitally important to remember that the System Audit Trail does not exist to allow users to spy on one another, nor does it exist as a mechanism to entrap users. The Audit Trail exists as a means to locate intentional violations of security policy.

The Audit Trail is generated by additional code in the operating system kernel that notes specific important events, such as file creation, file changes, file removal, invocation of programs, and the login and logout events.

Audit information must be carefully gathered and protected so that actions affecting security can be traced to the responsible party. The Commercial Security Pak records the occurrences of security-relevant events in an audit log. For each event audited, the system records the date and time of the event, the initiating user, the type of event, the success or failure of the event, and the name and security classification of the files or programs used.

Most audit records are generated in the course of normal work. Even records with ominous sounding names, such as "sat_access_denied," happen in the course of ordinary activities. Your Auditor is not spying on your system activity, simply guarding against an outsider attempting to damage your work.

You do not need to take any action regarding the Audit Trail. It is maintained by the system and by the Auditor at your site. The Auditing process is completely transparent to the user. It is important to recognize that when working on a trusted system, your actions are audited. You should not, however, be apprehensive or fearful of the auditing process. Its function is to protect you from others who may try to use your identity for mischief.

Using CSP-Kerberos

CSP-Kerberos is based on the Kerberos V5 authentication system developed at MIT. Kerberos is named for the three-headed watchdog from Greek mythology, who guarded the entrance to the underworld.

Introduction to CSP-Kerberos

Under CSP-Kerberos, a client (generally either a user or a service) sends a request for a ticket to the Key Distribution Center (KDC). The KDC creates a ticket-granting ticket (TGT) for the client, encrypts it using the client password as the key, and sends the encrypted TGT back to the client. The client then attempts to decrypt the TGT, using its password. If the client successfully decrypts the TGT (if the client gave the correct password), it keeps the decrypted TGT, which indicates proof of the client's identity.

The TGT, which expires at a specified time, permits the client to obtain additional tickets, which give permission for specific services. The requesting and granting of these additional tickets is user-transparent.

Since CSP-Kerberos negotiates authenticated, and optionally encrypted, communications between two points anywhere on the Internet, it provides a layer of security that is not dependent on which side of a firewall either client is on. Since studies have shown that half of the computer security breaches in industry happen from inside firewalls, Silicon Graphics CSP-Kerberos plays a vital role in maintaining your network security.

The CSP-Kerberos package is designed to be easy to use. Most of the commands are nearly identical to UNIX network programs you are already used to. CSP-Kerberos is a single-sign-on system, which means that you have to type your password only once per session, and CSP-Kerberos does the authenticating and encrypting transparently.

CSP-Kerberos Tickets

Your CSP-Kerberos credentials, or “tickets,” are a set of electronic information that can be used to verify your identity. Your CSP-Kerberos tickets may be stored in a file, or they may exist only in memory.

The first ticket you obtain is a ticket-granting ticket, which permits you to obtain additional tickets. These additional tickets give you permission for specific services. The requesting and granting of these additional tickets happens transparently.

A good analogy for the ticket-granting ticket is a three-day ski pass that is good at four different resorts. You show the pass at whichever resort you decide to go to (until it expires), and you receive a lift ticket for that resort. Once you have the lift ticket, you can ski all you want at that resort. If you go to another resort the next day, you once again show your pass, and you get an additional lift ticket for the new resort. The difference is that the CSP-Kerberos programs notice that you have the weekend ski pass, and get the lift ticket for you, so you don't have to perform the transactions yourself.

CSP-Kerberos Principals

A CSP-Kerberos principal is a unique identity to which CSP-Kerberos can assign tickets. By convention, a principal is divided into three parts: the primary, the instance, and the realm. The format of a typical CSP-Kerberos V5 principal is `primary/instance@REALM`:

- The primary is the first part of the principal. In the case of a user, it's the same as your username. For a host, the primary is the word `host`.
- The instance is an optional string that qualifies the primary. The instance is separated from the primary by a slash (/). In the case of a user, the instance is usually null, but a user might also have an additional principal, with an instance called `admin`, which he or she used to administer a database. The principal `eugene@YOURSITE.COM` is completely separate from the principal `eugene/admin@YOURSITE.COM`, with a separate password and separate permissions. In the case of a host, the instance is the fully qualified hostname, for example, `laughter.yoursite.com`
- The realm is your CSP-Kerberos realm. In most cases, your CSP-Kerberos realm is your domain name, in uppercase letters. For example, the system `laughter.yoursite.com` is in the realm `YOURSITE.COM`.

CSP-Kerberos Tutorial

This tutorial is intended to familiarize you with the CSP-Kerberos client programs. In these examples, sample user names, such as *dave* and *jennifer*, sample hostnames, such as *laughter* and *rain*, and sample domain names, such as *yoursite.com* and *theirsite.com* are used. When you see one of these, substitute your username, hostname, or domain name accordingly.

Setting Up to Use CSP-Kerberos

Your system administrator has installed the CSP-Kerberos programs in whichever directory makes the most sense for your system. The directory */krb5* is used throughout this guide to refer to the top-level CSP-Kerberos directory. The directory */krb5/bin* is used to denote the location of the CSP-Kerberos user programs. Place the */krb5/bin* and */krb5/sbin* directories in your PATH environment variable. You will probably want to put it ahead of the directories */bin* and */usr/bin* so you automatically use the CSP-Kerberos network programs, rather than the standard IRIX versions, when you type the command names.

Ticket Management

On many systems, CSP-Kerberos is built into the login program, and you get tickets automatically when you log in. Other programs, such as *rsh*, *rcp*, *telnet*, and *rlogin*, can forward copies of your tickets to the remote host. Most of these programs also automatically destroy your tickets when they exit. However, Silicon Graphics recommends that you explicitly destroy your CSP-Kerberos tickets when you are through with them, just to be sure. One way to help ensure that this happens is to add the *kdestroy* command to your *.logout* file. Additionally, if you are going to be away from your system and are concerned about an intruder using your permissions, it is safest to either destroy all copies of your tickets or use a screensaver that locks the screen.

Obtaining Tickets With kinit

If your site is using the CSP-Kerberos *login* program, you get CSP-Kerberos tickets automatically when you log in. If your site uses another *login* program, you may need to explicitly obtain your CSP-Kerberos tickets, using the *kinit* program. Similarly, if your CSP-Kerberos tickets expire, use the *kinit* program to obtain new ones.

To use the *kinit* program, simply type *kinit* and then type your password at the prompt. For example, Jennifer (whose user name is *jennifer*) works for Yoursite, Inc. (a fictitious company with the domain name *yoursite.com* and the CSP-Kerberos realm *YOURSITE.COM*). She would enter the command:

```
kinit  
Password for jennifer@YOURSITE.COM: <Type Jennifer's password here>
```

If you type your password incorrectly, *kinit* returns the following error message, and you don't get CSP-Kerberos tickets:

```
kinit: Password incorrect
```

Notice that *kinit* assumes you want tickets for your own username in your default realm. Suppose Jennifer's friend David is visiting, and he wants to borrow a window to check his mail. David needs to get tickets for himself in his own realm, *THEIRSITE.COM*.

Note: Any alternate realm, such as *THEIRSITE.COM*, must be listed in your system's CSP-Kerberos configuration file, */etc/krb5.conf*. David would enter the command:

```
kinit dave@THEIRSITE.COM  
Password for dave@THEIRSITE.COM <Type Dave's password here>
```

David now has tickets which he can use to log onto his own system. Note that he typed his password locally on Jennifer's system, but it never went over the network. CSP-Kerberos on the local host performed the authentication to the KDC in the other realm.

If you want to forward your tickets to another host, you need to request forwardable tickets. You do this by specifying the **-f** option:

```
kinit -f  
Password for jennifer@YOURSITE.COM: <Type your password here>
```

Note: The *kinit* command does not tell you that it obtained forwardable tickets; you can verify this using the *klist* command

Normally, your tickets are good for your system's default ticket lifetime, which is ten hours on many systems. You can specify a different ticket lifetime with the **-l** option. Add the letter *s* to the value for seconds, *m* for minutes, *h* for hours, or *d* for days.

For example, to obtain forwardable tickets for *dave@THEIRSITE.COM* that are good for three hours, enter the command:

```
kinit -f -l 3h dave@THEIRSITE.COM
```

Password for dave@THEIRSITE.COM: <Type dave's password here>

You cannot mix units; specifying a lifetime of 3h30m results in an error. Note also that most systems specify a maximum ticket lifetime. If you request a longer ticket lifetime, it is automatically truncated to the maximum lifetime.

Viewing Your Tickets With `klist`

The `klist` command shows your tickets. When you first obtain tickets, you have only the ticket-granting ticket. The listing looks like this:

```
klist
Ticket cache: /tmp/krb5cc_ttypa
Default principal: jennifer@YOURSITE.COM
Valid starting    Expires          Service principal
01/07/97 19:49:21 01/08/97 05:49:19  krbtgt/YOURSITE.COM@YOURSITE.COM
```

The ticket cache is the location of your ticket file. In the above example, this file is named `/tmp/krb5cc_ttypa`. The default principal is your CSP-Kerberos principal.

The *valid starting* and *expires* fields describe the period of time during which the ticket is valid. The service principal describes each ticket. The ticket-granting ticket has the primary *krbtgt*, and the instance is the realm name.

Now, if Jennifer connects to the system `laughter.yoursite.com`, and types `klist` again, she gets the following result:

```
klist
Ticket cache: /tmp/krb5cc_ttypa
Default principal: jennifer@YOURSITE.COM
Valid starting    Expires          Service principal
01/07/97 19:49:21 01/08/97 05:49:19  krbtgt/YOURSITE.COM@YOURSITE.COM
01/07/97 20:22:30 01/08/97 05:49:19  host/laughter.yoursite.com@YOURSITE.COM
```

Here's what happens: When Jennifer uses `telnet` to connect to the host `laughter.yoursite.com`, the `telnet` program presents her ticket-granting ticket to the KDC and requests a host ticket for the host `laughter.yoursite.com`. The KDC sends the host ticket, which `telnet` then presents to the host `laughter.yoursite.com`, and she is allowed to log in without entering her password.

Suppose your CSP-Kerberos tickets allow you to log into a host in another domain, such as `rain.theirsite.com`, which is also in another CSP-Kerberos realm, `THEIRSITE.COM`. If

you *telnet* to this host, you receive a ticket-granting ticket for the realm THEIRSITE.COM, plus the new host ticket for rain.theirsite.com. The klist command now shows:

```
klist
Ticket cache: /tmp/krb5cc_ttypa
Default principal: jennifer@YOURSITE.COM
Valid starting      Expires            Service principal
06/07/96 19:49:21  06/08/96 05:49:19  krbtgt/YOURSITE.COM@YOURSITE.COM
06/07/96 20:22:30  06/08/96 05:49:19  host/laughter.yoursite.com@YOURSITE.COM
06/07/96 20:24:18  06/08/96 05:49:19  krbtgt/THEIRSITE.COM@YOURSITE.COM
06/07/96 20:24:18  06/08/96 05:49:19  host/rain.theirsite.com@YOURSITE.COM
```

You can use the **-f** option to view the flags that apply to your tickets. The flags are:

F	Forwardable
f	forwarded
P	Proxiable
p	proxy
D	postDateable
d	postdated
R	Renewable
I	Initial
i	invalid

Here is a sample listing. In this example, the user jennifer obtained her initial tickets (I), which are forwardable (F) and postdated (d) but not yet validated (i).

```
klist -f
Ticket cache: /tmp/krb5cc_320
Default principal: jennifer@YOURSITE.COM
Valid starting      Expires            Service principal
31 Jul 96 19:06:25  31 Jul 96 19:16:25  krbtgt/YOURSITE.COM@YOURSITE.COM
Flags: FdiI
```

In the following example, Dave's tickets were forwarded (f) to this host from another host. The tickets are reforwardable (F).

```
klist -f
Ticket cache: /tmp/krb5cc_p11795
Default principal: dave@THEIRSITE.COM
Valid starting      Expires            Service principal
07/31/96 11:52:29  07/31/96 21:11:23  krbtgt/THEIRSITE.COM@THEIRSITE.COM
                   Flags: Ff
07/31/96 12:03:48  07/31/96 21:11:23  host/rain.theirsite.com@THEIRSITE.COM
                   Flags: Ff
```

Destroying Your Tickets With `kdestroy`

Your CSP-Kerberos tickets are proof that you are indeed yourself, and tickets can be stolen. If this happens, the person who has them can masquerade as you until they expire. For this reason, you should destroy your CSP-Kerberos tickets when you are away from your computer.

Destroying your ticket is easy. Simply use the `kdestroy` command:

```
kdestroy
```

If `kdestroy` fails to destroy all your tickets, it beeps and displays an error message. For example, if `kdestroy` can't find any tickets to destroy, it displays the following message:

```
kdestroy
kdestroy: No credentials cache file found while destroying cache
Ticket cache NOT destroyed!
```

CSP-Kerberos Password Management

Your password is the only way CSP-Kerberos has of verifying your identity. If someone finds out your password, that person can masquerade as you—send e-mail that comes from you, read, edit, or delete your files, or log into other hosts as you—and no one can tell the difference. For this reason, it is important that you choose a good password, and keep it secret. If you need to give access to your account to someone else, you can do so through CSP-Kerberos. Never tell your password to anyone, including your system administrator, for any reason. Change your password frequently, particularly any time you think someone may have found out what it is.

Changing Your CSP-Kerberos Password

To change your CSP-Kerberos password, use the *kpasswd* command. It asks you for your old password (to prevent someone else from changing your password), and then prompts you for the new one twice. (The reason you have to type it twice is to make sure you have typed it correctly.) For example, user dave would do the following:

```
kpasswd  
Old password for dave: <Type your old password>  
New Password for dave: <Type your new password>  
Verifying, please re-enter  
New Password for dave: <Type the new password again>  
Password changed.
```

If Dave types an incorrect old password, he gets the following message:

```
kpasswd  
Old password for dave: <Type an incorrect old password>  
Incorrect old password.
```

If you make a mistake and don't type the new password the same way twice, *kpasswd* asks you to try again:

```
kpasswd  
Old password for dave: <Type the old password>  
New Password for dave: <Type the new password>  
Verifying, please re-enter  
New Password for dave: <Type a different new password>  
Mismatch - try again  
New Password for dave: <Type the new password>  
Verifying, please re-enter  
New Password for dave: <Type the same new password>  
Password changed.
```

Once you change your password, it takes some time for the change to propagate through the system. Depending on how your system is set up, this might be anywhere from a few minutes to an hour or more. If you need to get new CSP-Kerberos tickets shortly after changing your password, try the new password. If the new password doesn't work, try again using the old one.

CSP-Kerberos Password Advice

Your password can include almost any character you can type (except control keys and the Enter key). A good password is one you can remember, but that no one else can easily guess. Examples of bad passwords are words that can be found in a dictionary, any common or popular name, especially a famous person (or cartoon character), your name or user name in any form (such as forward, backward, repeated twice, and so on.), your spouse's, child's, or pet's name, your birth date, your social security number, and any sample password that appears in this (or any other) manual.

Silicon Graphics recommends that your password be at least six characters long, and contain UPPER and lower case letters, numbers, and/or punctuation marks. Some passwords that would be good if they weren't listed in this manual include:

- some initials, like "GykoR-66." for "Get your kicks on Route 66."
- an easy-to-pronounce nonsense word, like "slaRooBey" or "krang-its."
- a misspelled phrase, like "2HotPeetzas!" or "ItzAGurl!!!"

Note: Don't use any of the above passwords. They're meant to show you how to make up a good password. Passwords that appear in a manual are the first ones intruders try.

CSP-Kerberos allows your system administrators to automatically reject bad passwords, based on whatever criteria they choose; the Commercial Security Pak provides a facility to automatically generate good passwords. For example, if the user jennifer chose a bad password, CSP-Kerberos displays an error message like the following:

```
kpasswd
Old password for jennifer: <Type your old password here>
New Password for jennifer: <Type an insecure new password>
Verifying, please re-enter
New Password for jennifer: <Type it again>
ERROR: Insecure password not accepted. Please choose another.
kpasswd: Insecure password rejected while attempting to change password.
Please choose another password.
New Password for jennifer: <Type a good password here>
Verifying, please re-enter New Password for dave: <Type it again>
Password changed.
```

Your system administrators can choose the message that is displayed if you choose a bad password, so the message you see may be different from the above example.

Granting Access to Your Account

If you need to give someone access to log into your account, you can do so through CSP-Kerberos, without telling the person your password. Simply create a file called *.k5login* in your home directory. This file should contain the CSP-Kerberos principal of each person to whom you wish to give access. Each principal must be on a separate line. Here is a sample *.k5login* file:

```
jennifer@YOURSITE.COM  
dave@THEIRSITE.COM
```

This file allows the users *jennifer* and *dave* to use your user ID, provided that they had CSP-Kerberos tickets in their respective realms. If you log in to other hosts across a network, you need to include your own CSP-Kerberos principal in your *.k5login* file on each of these hosts.

Using a *.k5login* file is much safer than giving out your password, because:

- you can take access away any time simply by removing the principal from your *.k5login* file.
- although the user has full access to your account on one particular host (or set of hosts if your *.k5login* file is shared, such as over NFS™), that user does not inherit your network privileges.
- CSP-Kerberos keeps a log of who obtains tickets, so a system administrator could find out, if necessary, who was capable of using your user ID at a particular time.

One common application is to have a *.k5login* file in **root**'s home directory, giving **root** access to that system to the CSP-Kerberos principals listed. This allows system administrators to allow users to become **root** locally, or to log in remotely as **root**, without their having to give out the **root** password, and without anyone having to type the **root** password over the network.

CSP-Kerberos Applications

CSP-Kerberos is a single-sign-on system. This means that you have to type your password only once, and the CSP-Kerberos programs authenticate (and optionally encrypt) for you. The way this works is that CSP-Kerberos has been built into each of a suite of network programs. For example, when you use a CSP-Kerberos program to connect to a remote host, the program, the KDC, and the remote host perform a set of rapid negotiations. When these negotiations are completed, your program has proven your identity on your behalf to the remote host, and the remote host has granted you access, all in the space of a few seconds.

The CSP-Kerberos applications are versions of existing UNIX network programs with the CSP-Kerberos features added.

Overview of Kerberized Programs

The CSP-Kerberos network programs are those programs that connect to another host somewhere on the Internet. These programs include *rlogin*, *telnet*, *ftp*, *rsh*, *rcp*, and *ksu*. These programs have all of the original features of the corresponding standard *rlogin*, *telnet*, *ftp*, *rsh*, *rcp*, and *su* programs, plus additional features that use your CSP-Kerberos tickets for negotiating authentication and optional encryption with the remote host. In most cases, all you notice is that you no longer have to type your password, because CSP-Kerberos has already proven your identity.

The CSP-Kerberos network programs allow you the options of forwarding your tickets to the remote host (if you obtained forwardable tickets with the *kinit* program), and encrypting data transmitted between you and the remote host.

This section of the chapter assumes you are familiar with the standard versions of these programs, and highlights the CSP-Kerberos functions added in the CSP-Kerberos package.

telnet

The CSP-Kerberos *telnet* command works exactly like the standard *telnet* program, with the following CSP-Kerberos options added:

-f, -forward Forwards a copy of your tickets to the remote host.

- noforward** Turns off forwarding of tickets to the remote host. (This option overrides any forwarding specified in your system's configuration files.)
- F, -forwardable** Forwards a copy of your tickets to the remote host, and marks them re-forwardable from the remote host.
- noforwardable** Makes any forwarded tickets nonforwardable. (This option overrides any forwardability specified in your system's configuration files.)
- k realm** Requests tickets for the remote host in the specified realm, instead of determining the realm itself.
- K** Uses your tickets to authenticate to the remote host, but does not log you in.
- a** Attempts automatic login using your tickets. The *telnet* command assumes the same user name unless you explicitly specify another.
- x, -encrypt** Turns on encryption.
- noencrypt** Turns off encryption.

For example, if Dave wants to use the standard IRIX *telnet* to connect to the system *laughter.yoursite.com*, he enters the command:

```
telnet laughter.yoursite.com
Trying 128.0.0.5 ...
Connected to laughter.yoursite.com.
Escape character is '^]'.
NetBSD/i386 (laughter) (ttyp3)
login: dave
Password: <dave types his password here>
Last login: Fri Jun 21 17:13:11 from rain.theirsite.com
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
The Regents of the University of California. All rights reserved.

NetBSD 1.1: Tue May 21 00:31:42 EDT 1996

Welcome to NetBSD!
```

Note that the system *laughter.yoursite.com* asked for Dave's password. When he typed it, his password was sent over the network unencrypted. If an intruder were watching network traffic at the time, that intruder would know Dave's password.

If, on the other hand, jennifer wanted to use the CSP-Kerberos *telnet* to connect to the system `rain.theirsite.com`, she could forward a copy of her tickets, request an encrypted session, and log in as herself, as follows:

```
telnet -a -f -x rain.theirsite.com
Trying 128.0.0.5...
Connected to rain.theirsite.com.
Escape character is '^]'.
[ CSP-Kerberos V5 accepts you as "jennifer@theirsite.com" ]
[ CSP-Kerberos V5 accepted forwarded credentials ]
NetBSD 1.1: Tue May 21 00:31:42 EDT 1996
```

```
Welcome to NetBSD!
```

Note that Jennifer's system used CSP-Kerberos to authenticate her to `rain.theirsite.com`, and logged her in automatically as herself. She had an encrypted session, a copy of her tickets already waiting for her, and she never typed her password.

If you forward your CSP-Kerberos tickets, *telnet* automatically destroys them when it exits. The full set of options to CSP-Kerberos *telnet* are discussed in the *telnet* reference page.

rlogin

The CSP-Kerberos *rlogin* command works exactly like the standard *rlogin* program, with the following CSP-Kerberos options added:

- f, -forward** Forwards a copy of your tickets to the remote host.
- noforward** Turns off forwarding of tickets to the remote host. (This option overrides any forwarding specified in your system's configuration files.)
- F, -forwardable** Forwards a copy of your tickets to the remote host, and marks them forwardable from the remote host.
- noforwardable** Makes any forwarded tickets nonforwardable. (This option overrides any forwardability specified in your system's configuration files.)
- k realm** Requests tickets for the remote host in the specified realm, instead of determining the realm itself.

-x, -encrypt Turns on encryption.

-noencrypt Turns off encryption.

For example, if Dave wanted to use the standard *rlogin* to connect to the system *laughter.yoursite.com*, he would enter the command:

```
rlogin laughter.yoursite.com -l dave
Password: <dave types his password here>
Last login: Fri Jun 21 10:36:32 from :0.0
Copyright (c) 1980, 1983, 1986, 1988, 1990, 1991, 1993, 1994
The Regents of the University of California. All rights reserved.
```

```
NetBSD 1.1: Tue May 21 00:31:42 EDT 1996
```

```
Welcome to NetBSD!
```

Note: The system *laughter.yoursite.com* asks for Dave's password. When he types it, his password is sent over the network unencrypted. If an intruder were watching network traffic at the time, that intruder would know Dave's password. If, on the other hand, Jennifer wanted to use CSP-Kerberos *rlogin* to connect to the system *rain.theirsite.com*, she could forward a copy of her tickets, mark them as not forwardable from the remote host, and request an encrypted session as follows:

```
rlogin rain.theirsite.com -f -x
This rlogin session is using DES encryption for all data transmissions.
Last login: Thu Jun 20 16:20:50 from laughter
SunOS Release 4.1.4 (GENERIC) #2: Tue Nov 14 18:09:31 EST 1995
Not checking quotas. Try quota.real if you need them.
```

Notice that Jennifer's system uses CSP-Kerberos to authenticate her to *rain.theirsite.com*, and logs her in automatically as herself. She has an encrypted session, a copy of her tickets are waiting for her, and she never types her password.

If you forward your CSP-Kerberos tickets, *rlogin* automatically destroys them when it exits. The full set of options to CSP-Kerberos *rlogin* are discussed in the *rlogin* reference page.

FTP

The CSP-Kerberos FTP program works exactly like the standard FTP program, with the following CSP-Kerberos features added:

- k realm** Requests tickets for the remote host in the specified realm, instead of determining the realm itself.
- forward** Requests that your tickets be forwarded to the remote host. The `-forward` argument must be the last argument on the command line.
- protect level** (issued at the `ftp>` prompt) Sets the protection level. Clear is no protection; Safe ensures data integrity by verifying the checksum, and Private encrypts the data. Encryption also ensures data integrity.

For example, suppose Jennifer wants to get her RMAIL file from the directory `~jennifer/Mail`, on the host `laughter.yoursite.com`. She wants to encrypt the file transfer. The exchange looks like this:

```
ftp laughter.yoursite.com
Connected to laughter.yoursite.com.
220 laughter.yoursite.com FTP server (Version 5.60) ready.
334 Using authentication type GSSAPI; ADAT must follow
GSSAPI accepted as authentication type
GSSAPI authentication succeeded
Name (laughter.yoursite.com:jennifer): <Return>
232 GSSAPI user jennifer@YOURSITE.COM is authorized as jennifer
230 User jennifer logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> protect private
200 Protection level set to Private.
ftp> cd ~jennifer/MAIL
250 CWD command successful.
ftp> get RMAIL
227 Entering Passive Mode (128,0,0,5,16,49)
150 Opening BINARY mode data connection for RMAIL (361662 bytes).
226 Transfer complete.
361662 bytes received in 2.5 seconds (1.4e+02 Kbytes/s)
ftp> quit
```

The full set of options to CSP-Kerberos FTP are discussed in the *ftp* reference page.

rsh

The CSP-Kerberos *rsh* program works exactly like the standard UNIX *rsh* program, with the following CSP-Kerberos features added:

- f, -forward** Forwards a copy of your tickets to the remote host.
- noforward** Turns off forwarding of tickets to the remote host. (This option overrides any forwarding specified in your system's configuration files.)
- F, -forwardable**
 Forwards a copy of your tickets to the remote host, and marks them forwardable from the remote host.
- noforwardable**
 Makes any forwarded tickets nonforwardable. (This option overrides any forwardability specified in your system's configuration files.)
- k realm** Requests tickets for the remote host in the specified realm, instead of determining the realm itself.
- x, -encrypt** Turns on encryption.
- noencrypt** Turns off encryption.

For example, if your CSP-Kerberos tickets allowed you to run programs on the host *rain@theirsite.com* as root, you could run the *date* program as follows:

```
rsh rain.theirsite.com -l root -x date
This rsh session is using DES encryption for all data transmissions.
Fri Jun 21 17:06:12 EDT 1996
%
```

If you forward your CSP-Kerberos tickets, *rsh* automatically destroys them when it exits. The full set of options to CSP-Kerberos *rsh* are discussed in the *rsh* reference page.

rcp

The CSP-Kerberos *rcp* program works exactly like the standard *rcp* program, with the following CSP-Kerberos features added:

- k realm** Requests tickets for the remote host in the specified realm, instead of determining the realm itself.
- x, -encrypt** Turns on encryption.

For example, if you want to copy the file */etc/motd* from the host *laughter.yoursite.com* into the current directory, through an encrypted connection, you simply type:

```
rcp -x laughter.yoursite.com:/etc/motd .
```

The *rcp* program negotiates authentication and encryption transparently. The full set of options to CSP-Kerberos *rcp* are discussed in the *rcp* reference page.

ksu

The CSP-Kerberos *ksu* program replaces the standard IRIX *su* program. The *ksu* command first authenticates you to CSP-Kerberos. Depending on the configuration of your system, *ksu* may ask for your CSP-Kerberos password if authentication fails. Note that you should never type your password if you are remotely logged in using an unencrypted connection.

Once *ksu* has authenticated you, if your CSP-Kerberos principal appears in the target's *.k5login* file or in the target's *.k5users* file, it switches your user ID to the target user ID.

For example, Dave has put Jennifer's CSP-Kerberos principal in his *.k5login* file. If Jennifer uses *ksu* to become Dave, the exchange would look like the following example. (To differentiate between the two shells, Jennifer's prompt is represented as *jennifer%* and Dave's prompt is represented as *dave%*.)

```
jennifer% ksu dave
Account dave: authorization for jennifer@YOURSITE.COM successful
Changing uid to dave (3382)
dave%
```

Note that the new shell has a copy of Jennifer's tickets. The ticket filename contains Dave's UID with *.1* appended to it:

```
dave% klist
Ticket cache: /tmp/krb5cc_3382.1
Default principal: jennifer@YOURSITE.COM

Valid starting      Expires            Service principal
31 Jul 96 21:53:01  01 Aug 96 07:52:53  krbtgt/YOURSITE.COM@YOURSITE.COM
31 Jul 96 21:53:39  01 Aug 96 07:52:53  host/laughter.yoursite.com@YOURSITE.COM
dave%
```

If Jennifer had not appeared in Dave's *.k5login* file (and the system was configured to ask for a password), the exchange would have looked like this (assuming Dave has taken appropriate precautions in protecting his password):

```
jennifer% ksu dave
WARNING: Your password may be exposed if you enter it here and are
logged in remotely using an unsecure (non-encrypted) channel.
CSP-Kerberos password for dave@YOURSITE.COM: <jennifer types wrong password>
ksu: Password incorrect
Authentication failed.
jennifer%
```

Now, suppose Dave does not want to give Jennifer full access to his account, but wants to give her permission to list his files and use the *more* command to view them. He could create a *.k5users* file giving her permission to run only those specific commands.

The *.k5users* file is like the *.k5login* file, except that each principal is optionally followed by a list of commands. The *ksu* command lets those principals execute only the commands listed, using the *-e* option. Dave's *.k5users* file might look like this:

```
jennifer@YOURSITE.COM      /bin/ls /usr/bin/more
eugene@YOURSITE.COM      /bin/ls
eugene/admin@YOURSITE.COM *
dave@THEIRSITE.COM
```

The above *.k5users* file would let the account *jennifer* run only the commands */bin/ls* and */usr/bin/more*. It lets the account *eugene* run only the command */bin/ls* if he has regular tickets, but if he has tickets for his admin instance, *eugene/admin@YOURSITE.COM*, he can execute any command. The last line gives Dave in the realm *THEIRSITE.COM* permission to execute any command. (For example, having only a CSP-Kerberos principal on a line is equivalent to giving that principal permission to execute anything.) This is so that Dave can allow himself to execute commands when he logs in, using CSP-Kerberos, from a system in the realm *THEIRSITE.COM*.

Then, when Jennifer wants to list his home directory, she types:

```
jennifer% ksu dave -e ls ~dave
Authenticated jennifer@YOURSITE.COM
Account dave: authorization for jennifer@YOURSITE.COM for execution of
/bin/ls successful
Changing uid to dave (3382)
Mail          News          Personal      misc          bin
jennifer%
```


If Jennifer tries to give a different command to *ksu*, it prompts for a password as with the previous example.

Note that unless the *.k5users* file gives the target permission to run any command, the user must use *ksu* with the **-e** command option.

The *ksu* options you are most likely to use are:

- n** *principal* Specifies which CSP-Kerberos principal you want to use for *ksu*. (For example, the user eugene might want to use his admin instance.)
- c** Specifies the location of your CSP-Kerberos credentials cache (ticket file).
- C** Specifies the location you want the CSP-Kerberos credentials cache (ticket file) to be for the target user ID.
- k** Tells *ksu* not to destroy your CSP-Kerberos tickets when *ksu* is finished.
- f** Requests forwardable tickets. This is applicable only if *ksu* needs to obtain tickets.
- l** *lifetime* Sets the ticket lifetime. This is applicable only if *ksu* needs to obtain tickets.
- z** Tells *ksu* to copy your CSP-Kerberos tickets only if the UID you are switching to is the same as the CSP-Kerberos primary (either yours or the one specified by the **-n** option).
- Z** Tells *ksu* not to copy any CSP-Kerberos tickets to the new UID.
- e** *command* Tells *ksu* to execute command and then exit. See the description of the *.k5users* file above.
- a** *text* (At the end of the command line) Tells *ksu* to pass everything after **-a** to the target shell.

The full set of options to CSP-Kerberos *ksu* are discussed in the *ksu* reference page.

Programming in a Trusted Environment

This chapter gives guidelines on programming in a secure environment, and a list of new system and library calls available under the Commercial Security Pak.

Programming Guidelines

There are a number of guidelines that you, as a programmer in a secure environment, should follow:

- In order to simplify your work, do not duplicate the work done by the Identification & Authentication programs of the system or by CSP-Kerberos.
- Assure that all variables are in bounds.
- Reduce global variable usage wherever possible.
- Limit the functionality of each module to only one distinct task.
- Do not create a procedure that circumvents any of the programmatic flow.
- If overrides must be added, document them thoroughly in the code.
- By design and principle, minimize the use of privilege required or permitted by your programs.

Commercial Security Pak System and Library Calls

The following system and library calls are distributed with the Commercial Security Pak. CSP-Kerberos calls are not listed here, but are available using anonymous *ftp* from athena.mit.edu. Reference pages exist for each of these calls in reference page sections 2 and 3.

Table 5-1 below lists each call and its corresponding action.

Table 5-1 Commercial Security Pak System and Library Calls

System/Library Call	Action
<i>satctl(2)</i>	control the collection of audit data
<i>satread(2)</i>	read a block of audit record data
<i>satwrite(2)</i>	write a block of audit record data
<i>acl_copy_ext(3C)</i>	copy ACL from system to user space or from user to system space
<i>acl_delete_def_file(3C)</i>	delete the default ACL for a named directory
<i>acl_dup(3C)</i>	make a copy of an ACL
<i>acl_free(3C)</i>	free memory allocated by ACL interface calls
<i>acl_from_text(3C)</i>	convert a POSIX ACL string to a struct <i>acl</i> or a struct <i>acl</i> to a POSIX ACL string
<i>acl_get_fd(3C)</i>	get or set the ACL associated with an open file
<i>acl_get_file(3C)</i>	get or set the ACL for a pathname
<i>acl_size(3C)</i>	return the size of an ACL
<i>acl_valid(3C)</i>	validate an ACL
<i>cap_acquire(3C)</i>	make permitted set capabilities effective or remove effective capabilities
<i>cap_clear(3C)</i>	clear the fields of a capability
<i>cap_copy_ext(3C)</i>	copy capability from system to user space or from user to system space
<i>cap_dup(3C)</i>	make a copy of a capability
<i>cap_free(3C)</i>	free allocated capability

Table 5-1 (continued) Commercial Security Pak System and Library Calls

System/Library Call	Action
<i>cap_from_text(3C), cap_to_text, cap_value_to_text</i>	convert a POSIX capabilities string to internal form, convert capabilities to a POSIX capabilities string, or return the POSIX name for a capability value
<i>cap_get_fd(3C), cap_set_fd</i>	get or set the capabilities for an open file
<i>cap_get_file(3C), cap_set_file</i>	get or set the capabilities for a pathname
<i>cap_get_flag(3C), cap_set_flag</i>	get or set the value of a capability flag in a capability
<i>cap_get_proc(3C), cap_set_proc</i>	get or set process capabilities
<i>cap_init(3C)</i>	allocate a capability structure
<i>cap_size(3C)</i>	return the size of a capability
<i>getspwnam(3)</i>	get a user's name from the administrative database
<i>getuserinfolam(3), getuserinfo(3)</i>	get information about a user.
<i>ia_audit(3)</i>	create and write an audit record, using <i>satwrite()</i>
<i>libperfex(3C), start_counters, read_counters, print_counters,</i>	a procedural interface to R10000 counters
<i>sat_eventtostr(3), sat_strtoevent(3)</i>	convert an audit event index to or from an audit event string
<i>sat_intrp_pathname(3C)</i>	Portable interface to interpret <i>sat_pathname</i> structs
<i>sat_read_file_info(3C), sat_write_file_info, sat_free_file_info</i>	Portable interfaces to read audit file headers
<i>sat_read_header_info(3C), sat_free_header_info</i>	Portable interfaces to read audit record headers
<i>sgi_acl_strtoacl(3C), sgi_acl_acltostr</i>	convert an ACL string to a struct <i>acl</i> or convert a struct <i>acl</i> to an ACL string

Table 5-1 (continued) Commercial Security Pak System and Library Calls

System/Library Call	Action
<i>sgi_cap_cleared(3C)</i>	determine whether a user’s allowed capabilities are sufficient
<i>sgi_cap_set_from_text(3C)</i>	set all capabilities from a capabilities string
<i>sgi_cap_strtocap(3C), sgi_cap_captostr</i>	convert a capabilities string to a <i>cap_value_t</i> or convert a <i>cap_value_t</i> to a capabilities string
<i>sgi_getcapabilitybyname(3C)</i>	get the default and allowed capability sets for a named user

Computer Security Terms

The terms listed in this glossary are used throughout the trusted systems community. The terminology is sometimes confusing, so this list can reduce that confusion.

acceptance inspection

The final inspection to determine whether or not a facility or system meets the specified technical and performance standards. This inspection is held immediately after facility and software testing and is the basis for commissioning or accepting the information system.

access

A specific type of interaction between a subject and an object that results in the flow of information from one to the other.

access control

The process of limiting access to the resources of a system only to authorized programs, processes, or other systems (in a network).
Synonymous with controlled access and limited access.

access control list

A discretionary access control entity associated with an object, consisting of a list of entries where each entry is an identifier (a user or group of users) coupled with a set of access permissions for that user or group.

access control mechanism

Hardware or software features, operating procedures, management procedures, and various combinations of these designed to detect and prevent unauthorized access and to permit authorized access in an automated system.

access period

A segment of time, generally expressed on a daily or weekly basis, during which access rights prevail.

access port

A logical or physical identifier that a computer uses to distinguish different tty input/output data streams.

access type

The nature of an access right to a particular device, program, or file (for example, read, write, execute, append, modify, delete, or create).

accountability	The property that enables activities on a system to be traced to individuals who may then be held responsible for their actions.
add-on security	The retrofitting of protection mechanisms, implemented by hardware or software.
administrative security	The management constraints and supplemental controls established to provide an acceptable level of protection for data. Synonymous with procedural security.
administrator	In the trusted system, the Administrator is responsible for system administration tasks. The Administrator is responsible for file system maintenance and repair, account creation, and other miscellaneous administrative duties.
assurance	A measure of confidence that the security features and architecture of an operating system accurately mediate and enforce the security policy.
attack	The act of trying to bypass security controls on a system. An attack may be active, resulting in the alteration of data; or passive, resulting in the release of data. The fact that an attack is made does not necessarily mean that it will succeed. The degree of success depends on the vulnerability of the system or activity and the effectiveness of existing countermeasures.
audit trail	<p>A chronological record of system activities that is sufficient to enable the reconstruction, reviewing, and examination of the sequence of environments and activities surrounding or leading to an operation, a procedure, or an event in a transaction from its inception to final results.</p> <p>Alternatively, a set of records that collectively provide documentary evidence of processing used to aid in tracing from original transactions forward to related records and reports, and/or backwards from records and reports to their component source transactions.</p>
auditor	The Auditor is an administrator who maintains and examines the System Audit Trail. This person is responsible for maintaining and rearchiving the information, examining the records for abuse, and customizing the audit record gathering configuration.
authenticate	To verify the identity of a user, device, or other entity in a computer system, often as a prerequisite to allowing access to resources in a system.

	Alternately, to verify the integrity of data that has been stored, transmitted, or otherwise exposed to possible unauthorized modification.
	Alternately, to establish the validity of a claimed identity.
authentication	Verifying the claimed identity of a principal.
authenticator	The means used to confirm the identity or to verify the eligibility of a station, originator, or individual. Alternatively, a record containing information that can be shown to have been recently generated using the session key known only by the client and server.
authorization	The granting of access rights to a user, program, or process. Alternatively, the process of determining whether a client may use a service, which objects the client is allowed to access, and the type of access allowed for each.
availability of data	The state when data is in the place needed by the user, at the time the user needs it, and in the form needed by the user.
back door	trap door.
backup plan	contingency plan.
bandwidth	A characteristic of a communication channel that is the amount of information that can be passed through it in a given amount of time, usually expressed in bits per second.
benign environment	A nonhostile environment that may be protected from external hostile elements by physical, personnel, and procedural security countermeasures.
between-the-lines entry	Unauthorized access obtained by tapping the temporarily inactive tty of a legitimate user. See also piggyback.
browsing	The act of searching through storage to locate or acquire information without necessarily knowing of the existence or the format of the information being sought.

callback	A procedure for identifying a remote system. In a callback, the host system disconnects the caller and then dials the authorized telephone number of the remote system to reestablish the connection. Synonymous with dialback.
capability	A capability is an attribute of a process that determines whether or not a process has the appropriate privilege to perform a specific action where appropriate privilege is required. Each capability may have associated with it one or more flags. For processes, three flags are always associated with the capability, namely the effective, the permitted, and the inheritable flag. A file may have zero or more of these flags associated with it for a capability. Appropriate privilege is determined solely by a process having a specific capability's effective capability flag set.
certification	The technical evaluation of a system's security features that establishes the extent to which a particular computer system's design and implementation meet a set of specified security requirements.
channel	An information transfer path within a system. May also refer to the mechanism by which the path is effected.
ciphertext	The output of an encryption function. Encryption transforms plain text into ciphertext.
client	A process that makes use of a network service, on behalf of a user. Note that in some cases, a server may itself be a client of some other server (for example a print server may be a client of a file server).
closed security environment	An environment in which both of the following conditions hold true: (1) Application developers (including maintainers) have sufficient clearances and authorizations to provide an acceptable presumption that they have not introduced malicious logic. (2) Configuration control provides sufficient assurance that applications and the equipment are protected against the introduction of malicious logic prior to and during the operation of system applications.
communications security	Measures taken to deny unauthorized persons information derived from telecommunications of the U.S. Government concerning national security, and to ensure the authenticity of such telecommunications.

	Communications security includes cryptosecurity, transmission security, and physical security of communications security material and information.
compromise	A violation of the security policy of a system such that unauthorized disclosure of sensitive information may have occurred.
compromising emanations	Unintentional data-related or intelligence-bearing signals that, if intercepted and analyzed, disclose the information transmission received, handled, or otherwise processed by any information processing equipment.
computer abuse	The misuse, alteration, disruption or destruction of data processing resources. The key aspect is that it is intentional and improper.
computer cryptography	The use of a crypto-algorithm in a computer, microprocessor, or microcomputer to perform encryption or decryption in order to protect information or to authenticate users, sources, or information.
computer fraud	Computer-related crimes involving deliberate misrepresentation, alteration, or disclosure of data in order to obtain something of value (usually for monetary gain). A computer system must have been involved in the perpetration or cover-up of the act or series of acts. A computer system might have been involved through improper manipulation of input data; output or results; application programs; data files; computer operations; communications; or computer hardware, system software, or firmware.
COMSEC	communications security
concealment system	A method of achieving confidentiality in which sensitive information is hidden by embedding it in irrelevant data.
confidentiality	The concept of holding sensitive data in confidence, limited to an appropriate set of individuals or organizations.
configuration control	The process of controlling modifications to the system's hardware, firmware, software, and documentation that provides sufficient

- assurance that the system is protected against the introduction of improper modifications prior to, during, and after system implementation.
- configuration management
The management of security features and assurances through control of changes made to a system's hardware, software, firmware, documentation, test, test fixtures, and test documentation throughout the development and operational life of the system.
- confinement Preventing the leaking of sensitive data from a program.
- confinement channel
See covert channel.
- contamination The intermixing of data at different sensitivity and need-to-know levels. The lower level data is said to be contaminated by the higher level data; thus, the contaminating (higher level) data may not receive the required level of protection.
- contingency plan
A plan for emergency response, backup operations, and post-disaster recovery maintained by an activity as a part of its security program that will ensure the availability of critical resources and facilitate the continuity of operations in an emergency situation. Synonymous with disaster plan and emergency plan.
- control zone The space, expressed in feet of radius, surrounding equipment processing sensitive information, that is under sufficient physical and technical control to preclude an unauthorized entry or compromise.
- controlled access
See access control.
- controlled sharing
The condition that exists when access control is applied to all users and components of a system.
- cost-risk analysis
The assessment of the costs of providing data protection for a system versus the cost of losing or compromising the data.
- countermeasure
Any action, device, procedure, technique, or other measure that reduces the vulnerability of or threat to a system.

-
- covert channel** A communications channel that allows two cooperating processes to transfer information in a manner that violates the system's security policy. See also confinement channel.
- Alternately, a communication channel that allows a process to transfer information in a manner that violates the system's security policy. See also: covert storage channel, covert timing channel.
- covert storage channel**
A covert channel that involves the direct or indirect writing of a storage location by one process and the direct or indirect reading of the storage location by another process. Covert storage channels typically involve a finite resource (for example, sectors on a disk) that is shared by two subjects at different security levels.
- covert timing channel**
A covert channel in which one process signals information to another by modulating its own use of system resources (for example, CPU time) in such a way that this manipulation affects the real response time observed by the second process.
- crypto-algorithm**
A well-defined procedure or sequence of rules or steps used to produce a key stream or cipher text from plain text and vice versa.
- cryptography** The principles, means, and methods for rendering information unintelligible, and for restoring encrypted information to intelligible form.
- cryptosecurity** The security or protection resulting from the proper use of technically sound cryptosystems.
- DAC** Discretionary Access Control
- data** Information with a specific physical representation.
- data flow control**
See information flow control.
- data integrity** The property that data meets an a priori expectation of quality.
- Alternatively, the state that exists when computerized data is the same as that in the source documents and has not been exposed to accidental or malicious alteration or destruction.
- data security** The protection of data from unauthorized (accidental or intentional) modification, destruction, or disclosure.

- degauss To reduce magnetic flux density to zero by applying a reverse magnetizing field.
- degausser An electrical device that can generate a magnetic field for the purpose of degaussing magnetic storage media.
- Degausser Products List
A list of commercially produced degaussers that meet National Security Agency specifications. This list is included in the NSA Information Systems Security Products and Services Catalogue, and is available through the Government Printing Office.
- denial of service
Any action or series of actions that prevent any part of a system from functioning in accordance with its intended purpose. This includes any action that causes unauthorized destruction, modification, or delay of service. Synonymous with interdiction.
- dialback See callback.
- dial up The service whereby a computer can use the telephone to initiate and effect communication with a computer.
- disaster plan See contingency plan.
- Discretionary Access Control
A means of restricting access to objects based on the identity and need-to-know of the user, process, and/or groups to which they belong. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission (perhaps indirectly) on to any other subject.
- DoD Trusted Computer System Evaluation Criteria
A document published by the National Computer Security Center containing a uniform set of basic requirements and evaluation classes for assessing degrees of assurance in the effectiveness of hardware and software security controls built into systems. These criteria are intended for use in the design and evaluation of systems that will process and/or store sensitive or classified data. This document is Government Standard DoD 5200.28-STD and is frequently referred to as "The Criteria" or "The Orange Book."

domain	<p>The unique context (for example, access control parameters) in which a program is operating; in effect, the set of objects that a subject has the ability to access. See also process and subject.</p> <p>Alternatively, the set of objects that a subject has the ability to access.</p>
emanations	See compromising emanations.
embedded system	A system that performs or controls a function, either in whole or in part, as an integral element of a larger system or subsystem.
emergency plan	See contingency plan.
emission security	The protection resulting from all measures taken to deny unauthorized persons information of value that might be derived from interception and from an analysis of compromising emanations from systems.
end-to-end encryption	<p>The protection of information passed in a telecommunications system by cryptographic means, from point of origin to point of destination.</p> <p>Alternatively, protection of traffic in a communications network by encrypting it at the source and decrypting it at the destination so that all nodes it passes through remain ignorant of its actual content.</p>
entrapment	The deliberate planting of apparent flaws in a system for the purpose of detecting attempted penetrations.
environment	The aggregate of external procedures, conditions, and objects that affect the development, operation, and maintenance of a system.
EPL	Evaluated Products List
erasure	<p>A process by which a signal recorded on magnetic media is removed. Erasure is accomplished in two ways: (1) by alternating current erasure, by which the information is destroyed by applying an alternating high and low magnetic field to the media; or (2) by direct current erasure, by which the media are saturated by applying a unidirectional magnetic field.</p>
Evaluated Products List	A list of equipment, hardware, software, and/or firmware that have been evaluated against, and found to be technically compliant, at a particular level of trust, with the DoD TCSEC by the NCSC. The EPL is

included in the National Security Agency Information Systems Security Products and Services Catalogue, which is available through the Government Printing Office.

evaluation criteria

The U.S. Government specifies a set of criteria that trusted systems must meet to be evaluated successfully. A trusted system must offer a number of specific security features and must demonstrate that it can be maintained and distributed in a trusted fashion.

executive state One of several states in which a system may operate and the only one in which certain privileged instructions may be executed. Such instructions cannot be executed when the system is operating in other (for example, user) states.

exploitable channel

Any information channel that is usable or detectable by subjects external to the trusted computing base and whose purpose is to violate the security policy of the system. See also covert channel.

Alternatively, any channel that is usable or detectable by subjects external to the Trusted Computing Base.

failsafe Pertaining to the automatic protection of programs and/or processing systems to maintain safety when a hardware or software failure is detected in a system.

fail soft Pertaining to the selective termination of affected nonessential processing when a hardware or software failure is detected in a system.

failure access An unauthorized and usually inadvertent access to data resulting from a hardware or software failure in the system.

failure control The methodology used to detect and provide failsafe or fail soft recovery from hardware and software failures in a system.

fault A condition that causes a device or system component to fail to perform in a required manner.

fetch protection A system-provided restriction to prevent a program from accessing data in another user's segment of storage.

file protection The aggregate of all processes and procedures in a system designed to inhibit unauthorized access, contamination, or elimination of a file.

file security The means by which access to computer files is limited to authorized users only.

flaw	An error of commission, omission, or oversight in a system that allows protection mechanisms to be bypassed.
flaw hypothesis methodology	A systems analysis and penetration technique in which specifications and documentation for the system are analyzed and then flaws in the system are hypothesized. The list of hypothesized flaws is then prioritized on the basis of the estimated probability that a flaw exists and, assuming a flaw does exist, on the ease of exploiting it, and on the extent of control or compromise it would provide. The prioritized list is used to direct a penetration attack against the system.
formal access approval	Documented approval by a data owner to allow access to a particular category of information.
formal proof	A complete and convincing mathematical argument, presenting the full logical justification for each proof step, for the truth of a theorem or set of theorems. The formal verification process uses formal proofs to show the truth of certain properties of formal specification and for showing that computer programs satisfy their specifications.
formal verification	The process of using formal proofs to demonstrate the consistency between a formal specification of a system and a formal security policy model (design verification) or between the formal specification and its high-level program implementation (implementation verification).
front-end security filter	A security filter, which could be implemented in hardware or software, that is logically separated from the remainder of the system to protect the system's integrity. Alternatively, a process that is invoked to process data according to a specified security policy prior to releasing the data outside the processing environment or upon receiving data from an external source.
functional testing	The segment of security testing in which the advertised security features of the system are tested, under operational conditions, for correct operation.

- granularity** An expression of the relative size of a data object. For example, protection at the file level is considered coarse granularity, whereas protection at field level is considered to be of a finer granularity.
- Alternatively, the relative fineness or coarseness by which a mechanism can be adjusted. The phrase “the granularity of a single user” means the access control mechanism can be adjusted to include or exclude any single user.
- guard** A processor that provides a filter between two disparate systems operating at different security levels or between a user process and a database to filter out data that the user is not authorized to access.
- handshaking procedure** A dialog between two entities (for example, a user and a computer, a computer and another computer, or a program and another program) for the purpose of identifying and authenticating the entities to one another.
- host to front-end protocol** A set of conventions governing the format and control of data that is passed from a host to a front-end system.
- I&A** Identification and Authentication
- identification** The process that enables recognition of an entity by a system, generally by the use of unique machine-readable user names.
- Identification and Authentication** The process of determining (with some level of confidence) the true identity of a user. The identification process usually requires both a user name and a password. The authentication part of the process is the underlying logic that the *login* and *su* programs go through in validating this password and username.
- impersonating spoofing.**
- incomplete parameter checking** A system design flaw that results when all parameters have not been fully anticipated for accuracy and consistency, thus making the system vulnerable to penetration.
- individual accountability** The ability to associate positively the identity of a user with the time, method, and degree of access to a system.

information flow control

A procedure to ensure that information transfers within a system are not made from a higher security level object to an object of a lower security level. See also covert channel and data flow control.

Information Systems Security Products and Services Catalogue

A catalogue issued quarterly by the National Security Agency that incorporates the DPL, EPL, ETL, PPL, and other security product and service lists. This catalogue is available through the U.S. Government Printing Office, Washington, DC (202) 1202) 783-3238.

instance

The name often given to the second component of a principal identifier, or a particular principal from a group of related principals. In the latter usage, the instances are often created to partition permission for users. For example, a user might have a "normal" instance, and a "root" instance which exists to provide different privileges or to impose a naming convention on service key names. For an example of a particular service, the instance(s) identifies the host system(s) on which that service is provided and the principal identifier of the server.

integrity

In secure systems, the term "integrity" refers to the relative level of trust a user can place in using a system resource. A program obtained from a public-access bulletin board is of much lower integrity than one purchased from a reputable vendor. This program is in turn of much lower integrity than a program shipped as part of a trusted system.

Alternatively, sound, unimpaired or perfect condition.

interdiction

See denial of service.

internal security controls

Hardware, firmware, and software features within a system that restrict access to resources (hardware, software, and data) to authorized subjects only (persons, programs, or devices).

isolation

The containment of subjects and objects in a system in such a way that they are separated from one another, as well as from the protection controls of the operating system.

lattice

A partially ordered set for which every pair of elements has a greatest lower bound and a least upper bound.

- least privilege The principle that requires that each subject be granted the most restrictive set of privileges needed for the performance of authorized tasks. The application of this principle limits the damage that can result from accident, error, or unauthorized use.
- limited access access control.
- lock-and-key protection system
A protection system that involves matching a key or password with a specific access requirement.
- logic bomb A resident computer program that triggers the perpetration of an unauthorized act when particular states of the system are realized.
- login-spoofing program
This term refers to any program that represents itself as a login program in order to steal your password. For example, a spoofing program might print the IRIX login banner on an unattended system and wait for input from the user. The user dutifully types in the user name, and the program prompts for the password, turning off character echo. After storing away the user's password, the program reports that the password is incorrect and exits, which causes the real login program to be started on the system. The user then logs in, mistakenly assuming that he or she previously mistyped the name or password, and starts a session.
- loophole An error of omission or oversight in software or hardware that permits circumventing the system security policy.
- magnetic remanence
A measure of the magnetic flux density remaining after removal of the applied magnetic force. Refers to any data remaining on magnetic storage media after removal of the power.
- maintenance hook
Special instructions in software to allow easy maintenance and additional feature development. These are not clearly defined during access for design specification. Hooks frequently allow entry into the code at unusual points or without the usual checks, so they are a serious security risk if they are not removed prior to live implementation. Maintenance hooks are special types of trap doors. See also trap door, back door.
- malicious logic Hardware, software, or firmware that is intentionally included in a system for an unauthorized purpose. See also Trojan horse.

masquerading spoofing.

mimicking spoofing.

multiple access rights terminal

A system or port that may be used by more than one class of users, for example, users with different access rights to data.

multiuser mode of operation

A mode of operation designed for systems that process sensitive unclassified information in which users may not have a need-to-know for all information processed in the system. This mode is also for microcomputers processing sensitive unclassified information that cannot meet the requirements of the standalone mode of operation.

mutually suspicious

The state that exists between interacting processes (subsystems or programs) in which neither process can expect the other process to function securely with respect to some property.

National Computer Security Center

Originally named the DoD Computer Security Center, the NCSC is responsible for encouraging the widespread availability of trusted computer systems throughout the Federal Government.

National Security Decision Directive 145

Signed by President Reagan on 17 September 1984, this directive is entitled "National Policy on Telecommunications and Automated Information Systems Security." It provides initial objectives, policies, and an organizational structure to guide the conduct of national activities toward safeguarding systems that process, store, or communicate sensitive information; establishes a mechanism for policy development; and assigns implementation responsibilities.

National Telecommunications and Information Systems Security Advisory Memoranda Instructions

NTISS Advisory Memoranda and Instructions provide advice, assistance, or information of general interest on telecommunications and systems security to all applicable federal departments and agencies. NTISSAMs/NTISSIs are promulgated by the National Manager for Telecommunications and Automated Information Systems Security and are recommendatory.

- National Telecommunications and Information Systems Security Directives
NTISS Directives establish national-level decisions relating to NTISS policies, plans, programs, systems, or organizational delegations of authority. NTISSDs are promulgated by the Executive Agent of the Government for Telecommunications and Information Systems Security, or by the Chairman of the NTISSC when so delegated by the Executive Agent. NTISSDs are binding upon all federal departments and agencies.
- NCSC National Computer Security Center
- need-to-know The necessity for access to, knowledge of, or possession of specific information required to carry out official duties.
- network front end
A device that implements the necessary network protocols, including security-related protocols, to allow a computer system to be attached to a network.
- NSDD 145 See National Security Decision Directive 145.
- object A passive entity that contains or receives information. Access to an object potentially implies access to the information it contains. Examples of objects are records, blocks, pages, segments, files, directories, directory trees, and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, and network nodes.
- object reuse The reassignment and reuse of a storage medium (for example, page frame, disk sector, magnetic tape) that once contained one or more objects. To be securely reused and assigned to a new subject, storage media must contain no residual data (magnetic remanence) from the object(s) previously contained in the media.
- open security environment
An environment that includes those systems in which at least one of the following conditions holds true: (1) Application developers (including maintainers) do not have sufficient clearance or authorization to provide an acceptable presumption that they have not introduced malicious logic. (2) Configuration control does not provide sufficient assurance that applications are protected against the introduction of malicious logic prior to and during the operation of system applications.

Operations Security

An analytical process by which the U.S. Government and its supporting contractors can deny to potential adversaries information about capabilities and intentions by identifying, controlling, and protecting evidence of the planning and execution of sensitive activities and operations.

OPSEC Operations Security

Orange Book Alternate name for DoD Trusted Computer Systems Evaluation Criteria.

output Information that has been exported by a computer.

overt channel A path within a computer system or network that is designed for the authorized transfer of data. Compare covert channel.

overwrite procedure

A stimulation to change the state of a bit followed by a known pattern. See also magnetic remanence.

password A protected/private character string used to authenticate an identity.

penetration The successful act of bypassing the security mechanisms of a system.

penetration signature

The characteristics or identifying marks that may be produced by a penetration.

penetration study

A study to determine the feasibility and methods for defeating controls of a system.

penetration testing

The portion of security testing in which the evaluators attempt to circumvent the security features of a system. The evaluators may be assumed to use all system design and implementation documentation, which may include listings of system source code, manuals, and circuit diagrams. The evaluators work under the same constraints applied to ordinary users.

periods processing

The processing of various levels of sensitive information at distinctly different times. Under periods processing, the system must be purged of all information from one processing period before transitioning to the next when there are different users with differing authorizations.

- permissions A description of the type of authorized interactions a subject can have with an object. Examples include read, write, execute, add, modify, and delete.
- personnel security The procedures established to ensure that all personnel who have access to sensitive information have the required authority as well as appropriate clearances.
- physical security The application of physical barriers and control procedures as preventive measures or countermeasures against threats to resources and sensitive information.
- piggyback Gaining unauthorized access to a system via another user's legitimate connection. See also between-the-lines entry.
- plain text The input to an encryption function or the output of a decryption function. Decryption transforms ciphertext into plain text.
- Preferred Products List A list of commercially produced equipments that meet requirements prescribed by the National Security Agency. This list is included in the NSA Information Systems Security Products and Services Catalogue, issued quarterly and available through the Government Printing Office.
- principal A uniquely named client or server instance that participates in a network communication.

 Alternately, an individual listing in a Kerberos database.
- principal identifier The name used to uniquely identify each different principal.
- print suppression Eliminating the displaying of characters in order to preserve their secrecy. (For example, not displaying the characters of a password as it is keyed in.)
- privileged instructions A set of instructions (for example, interrupt handling or special computer instructions) to control features (such as storage protection features) that are generally executable only when the automated system is operating in the executive state.

procedural security	Synonymous with administrative security.
process	A program in execution. It is completely characterized by a single current execution point (represented by the machine state) and address space.
protection-critical portions of the TCB	Those portions of the TCB whose normal function is to deal with the control of access between subjects and objects. Their correct operation is essential to the protection of the data on the system.
protection philosophy	An informal description of the overall design of a system that delineates each of the protection mechanisms employed. A combination, appropriate to the evaluation class, of formal and informal techniques is used to show that the mechanisms are adequate to enforce the security policy.
protection ring	One of a hierarchy of privileged modes of a system that gives certain access rights to user programs and processes authorized to operate in a given mode.
protocols	A set of rules and formats, semantic and syntactic, that permit entities to exchange information.
pseudo-flaw	An apparent loophole deliberately implanted in an operating system program as a trap for intruders.
Public Law 100-235	Also known as the Computer Security Act of 1987, this law creates a means for establishing minimum acceptable security practices for improving the security and privacy of sensitive information in federal computer systems. This law assigns to the National Institute of Standards and Technology responsibility for developing standards and guidelines for federal computer systems processing unclassified data. The law also requires establishment of security plans by all operators of federal computer systems that contain sensitive information.
rainbow series	The informal name given to a set of books published by the NCSC that deal with computer security. The books are published with covers in different colors, hence the term "rainbow." The most used book in the rainbow series is the Orange book, the DoD Trusted Computer Systems Evaluation Criteria.

read	A fundamental operation that results only in the flow of information from an object to a subject.
read access	Permission to read information.
recovery procedures	The actions necessary to restore a system's computational capability and data files after a system failure.
reference monitor concept	An access-control concept that refers to an abstract system that mediates all accesses to objects by subjects.
reference validation mechanism	An implementation of the reference monitor concept. A security kernel is a type of reference validation mechanism.
reliability	The probability of a given system performing its mission adequately for a specified period of time under the expected operating conditions.
residual risk	The portion of risk that remains after security measures have been applied.
residue	Data left in storage after processing operations are complete, but before degaussing or rewriting has taken place.
resource encapsulation	The process of ensuring that a resource not be directly accessible by a subject, but that it be protected so that the reference monitor can properly mediate accesses to it.
restricted area	Any area to which access is subject to special restrictions or controls for reasons of security or safeguarding of property or material.
risk	The probability that a particular threat will exploit a particular vulnerability of the system.
risk analysis	The process of identifying security risks, determining their magnitude, and identifying areas needing safeguards. Risk analysis is a part of risk management. Synonymous with risk assessment.
risk assessment	See risk analysis.
risk index	The disparity between the minimum authorization of system users and the maximum sensitivity of data processed by a system.

risk management	The total process of identifying, controlling, and eliminating or minimizing uncertain events that may affect system resources. It includes risk analysis; cost benefit analysis; selection; implementation and test; security evaluation of safeguards; and overall security review.
safeguards	See security safeguards.
scavenging	Searching through object residue to acquire unauthorized data.
seal	To encipher a record containing several fields, in such a way that the fields cannot be individually replaced without either knowledge of the encryption key or leaving evidence of tampering.
secure configuration management	The set of procedures appropriate for controlling changes to a system's hardware and software structure for the purpose of ensuring that changes will not lead to violations of the system's security policy.
secure state	A condition in which no subject can access any object in an unauthorized manner.
secure subsystem	A subsystem that contains its own implementation of the reference monitor concept for those resources it controls. However, the secure subsystem must depend on other controls and the base operating system for the control of subjects and the more primitive system objects.
security critical mechanisms	Those security mechanisms whose correct operation is necessary to ensure that the security policy is enforced.
security evaluation	An evaluation done to assess the degree of trust that can be placed in systems for the secure handling of sensitive information. One type, a product evaluation, is an evaluation performed on the hardware and software features and assurances of a computer product from a perspective that excludes the application environment. The other type, a system evaluation, is done for the purpose of assessing a system's security safeguards with respect to a specific operational mission and is a major step in the certification and accreditation process.

- security fault analysis
A security analysis, usually performed on hardware at gate level, to determine the security properties of a device when a hardware fault is encountered.
- security features
The security-relevant functions, mechanisms, and characteristics of system hardware and software. Security features are a subset of system security safeguards.
- security filter
A trusted subsystem that enforces a security policy on the data that passes through it.
- security flaw
An error of commission or omission in a system that may allow protection mechanisms to be bypassed.
- security flow analysis
A security analysis performed on a formal system specification that locates potential flows of information within the system.
- security kernel
The hardware, firmware, and software elements of a trusted system that implement the reference monitor concept. The security kernel must mediate all accesses, be protected from modification, and be verifiable as correct.
- security measures
Elements of software, firmware, hardware, or procedures that are included in a system for the satisfaction of security specifications.
- security perimeter
The boundary where security controls are in effect to protect assets.
- security policy
The set of laws, rules, and practices that regulate how an organization manages, protects, and distributes sensitive information.
- security policy model
A formal presentation of the security policy enforced by the system. It must identify the set of rules and practices that regulate how a system manages, protects, and distributes sensitive information.
- security requirements
The types and levels of protection necessary for equipment, data, information, applications, and facilities to meet the security policy.

security requirements baseline	A description of minimum requirements necessary for a system to maintain an acceptable level of security.
security safeguards	The protective measures and controls that are prescribed to meet the security requirements specified for a system. Those safeguards may include but are not necessarily limited to: hardware and software security features, operating procedures, accountability procedures, access and distribution controls, management constraints, personnel security, and physical structures, areas, and devices. Also called safeguards.
security specifications	A detailed description of the safeguards required to protect a system.
security test and evaluation	An examination and analysis of the security safeguards of a system as they have been applied in an operational environment to determine the security posture of the system.
security testing	A process used to determine that the security features of a system are implemented as desired. This includes hands-on functional testing, penetration testing, and verification.
sensitive information	Any information, the loss, misuse, modification, or unauthorized access of which could affect the national interest or the conduct of federal programs, or the privacy to which individuals are entitled under Section 552a of Title 5, U.S. Code, but that has not been specifically authorized under criteria established by an Executive order or an act of Congress to be kept classified in the interest of national defense or foreign policy.
sensitivity	In secure systems, "sensitivity" is a measure of the risk associated with the disclosure of the data in question. A map of a foreign city (UNCLASSIFIED) is less sensitive than the map of a foreign military base (SECRET) which is in turn less sensitive than the name of the asset who provided the maps (TOP SECRET).
server	A particular principal that provides a resource to network clients.
service	A resource provided to network clients, often provided by more than one server (for example, remote file service).

session key	A temporary encryption key used between two principals, with a lifetime limited to the duration of a single communications “session.”
Site Security Officer	The person responsible for ensuring the security of a trusted system. The person who serves this administrative role at your site is your point of contact for all security-related questions.
software security	General-purpose executive, utility, or software development tools and applications programs or routines that protect data handled by a system.
software system test and evaluation process	A process that plans, develops, and documents the quantitative demonstration of the fulfillment of all baseline functional performance, operational, and interface requirements.
spoofing	An attempt to gain access to a system by posing as an authorized user. Synonymous with impersonating, masquerading, or mimicking.
SSO	Site Security Officer
stand alone shared system	A system that is physically and electrically isolated from all other systems, and is intended to be used by more than one person either simultaneously (for example, a system with multiple monitors) or serially, with data belonging to one user remaining available to the system while another user is using the system (for example, a personal computer with nonremovable storage media such as a hard disk).
stand alone single-user system	A system that is physically and electrically isolated from all other systems, and is intended to be used by one person at a time, with no data belonging to other users remaining in the system (for example, a personal computer with removable storage media such as a floppy disk).
state variable	A variable that represents either the state of the system or the state of some system resource.
storage object	An object that supports both read and write accesses.
STS	Subcommittee on Telecommunications Security of NTISSC

Subcommittee on Automated Information Systems Security	NSDD-145 authorizes and directs the establishment, under the NTISSC, of a permanent Subcommittee on Automated Information Systems Security. The SAISS is composed of one voting member from each organization represented on the NTISSC.
Subcommittee on Telecommunications Security	NSDD-145 authorizes and directs the establishment, under the NTISSC, of a permanent Subcommittee on Telecommunications Security. The STS is composed of one voting member from each organization represented on the NTISSC.
subject	An active entity, generally in the form of a person, process, or device, that causes information to flow among objects or changes the system state. Technically, a process/domain pair.
system integrity	The quality that a system has when it performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system.
Systems Security Steering Group	The senior government body established by NSDD-145 to provide top-level review and policy guidance for the telecommunications security and automated information systems security activities of the U.S. Government. This group is chaired by the Assistant to the President for National Security Affairs and consists of the Secretary of State, Secretary of Treasury, the Secretary of Defense, the Attorney General, the Director of the Office of Management and Budget, and the Director of Central Intelligence.
tampering	An unauthorized modification that alters the proper functioning of a piece of equipment or system in a manner that degrades the security or functionality it provides.
TCB	Trusted Computing Base
TCSEC	DoD Trusted Computer System Evaluation Criteria
technical attack	An attack that can be perpetrated by circumventing or nullifying hardware and software protection mechanisms, rather than by subverting system personnel or other users.

technical vulnerability	A hardware, firmware, communication, or software flaw that leaves a computer processing system open for potential exploitation, either externally or internally, thereby resulting in risk for the owner, user, or manager of the system.
threat	Any circumstance or event with the potential to cause harm to a system in the form of destruction, disclosure, modification of data, and/or denial of service.
threat agent	A method used to exploit a vulnerability in a system, operation, or facility.
threat analysis	The examination of all actions and events that might adversely affect a system or operation.
threat monitoring	The analysis, assessment, and review of audit trails and other data collected for the purpose of searching out system events that may constitute violations or attempted violations of system security.
time-dependent password	A password that is valid only at a certain time of day or during a specified interval of time.
top-level specification	A nonprocedural description of system behavior at the most abstract level; typically, a functional specification that omits all implementation details.
trap door	A hidden software or hardware mechanism that can be triggered to permit system protection mechanisms to be circumvented. It is usually activated in an innocent-appearing manner, for example, a special "random" key sequence at a monitor. Software developers often introduce trap doors in their code to enable them to reenter the system and perform certain functions. Synonymous with back door.
Trojan horse	A computer program with an apparently or actually useful function that contains additional (hidden) functions that surreptitiously exploit the legitimate authorizations of the invoking process to the detriment of security. For example, making a "blind copy" of a sensitive file for the creator of the Trojan horse.

trusted computer system

A system that employs sufficient hardware and software assurance measures to allow its use for simultaneous processing of a range of sensitive or classified information. A system is trusted when it is believed that it can enforce a particular security policy.

Trusted Computing Base

The totality of protection mechanisms within a computer system, including hardware, firmware, and software, the combination of which is responsible for enforcing a security policy. A TCB consists of one or more components that together enforce a unified security policy over a product or system. The ability of a TCB to enforce correctly a unified security policy depends solely on the mechanisms within the TCB and on the correct input by system administrative personnel of parameters (for example, a user's clearance level) related to the security policy.

Alternatively, this term refers to the set of hardware and software that together enforce the system's security policy. The TCB is made up of only those programs and hardware elements that are known to follow security policy and are considered to be secure.

trusted distribution

A trusted method for distributing the TCB hardware, software, and firmware components, both originals and updates, that provides methods for protecting the TCB from modification during distribution and for detection of any changes to the TCB that may occur.

trusted process A process whose incorrect or malicious execution is capable of violating the system security policy.

trusted software

The software portion of the TCB.

untrusted process

A process that has not been evaluated or examined for adherence to the security policy. It may include incorrect or malicious code that attempts to circumvent the security mechanisms.

user

Person or process accessing the system either by direct connections (such as through the system console), or indirect connections (such as to prepare input data or receive output that is not reviewed for content or classification by a responsible individual).

user ID

A unique symbol or character string that is used by a system to identify a specific user.

user profile	Patterns of a user's activity that can be used to detect changes in normal routines.
virus	A self-propagating Trojan horse, composed of a mission component, a trigger component, and a self-propagating component.
vulnerability	A weakness in system security procedures, system design, implementation, internal controls, and so on, that could be exploited to violate system security policy.
vulnerability analysis	The systematic examination of systems to determine the adequacy of security measures, identify security deficiencies, and provide data from which to predict the effectiveness of proposed security measures.
vulnerability assessment	A measurement of vulnerability that includes the susceptibility of a particular system to a specific attack and the opportunities available to a threat agent to mount that attack.
work factor	An estimate of the effort or time needed by a potential penetrator with specified expertise and resources to overcome a protective measure.
worm	A worm is a virus which has a very narrow purpose. A worm is designed to track down and eliminate specific data. Unlike a simple virus, which by its very nature is obviously present, a worm is designed to remain unnoticed in order that it may continue its task unchecked. Because it may follow a serpentine path in its hunt for particular data, it has earned the nickname "worm."
write	A fundamental operation that results only in the flow of information from a subject to an object.
write access	Permission to write to an object.

Index

A

access control, 7
Access Control Lists, 12
accountability, 2
ACL, 12
assurance, 2
audit trail, 6, 21

C

capabilities, 17
chacl, 13
changing permissions, 10
conventions, typographical, xi
CSP-Kerberos, 23
CSP-Kerberos access, allowing, 32
CSP-Kerberos FTP, 37
CSP-Kerberos ksu, 39
CSP-Kerberos password, changing, 30
CSP-Kerberos passwords, 29
CSP-Kerberos principal, 24
CSP-Kerberos realm, 24
CSP-Kerberos rlogin, 35
CSP-Kerberos rsh, 38
CSP-Kerberos telnet, 33
CSP-Kerberos ticket, 24

D

DAC, 5, 7
 changing permissions, 10
 directory permissions, 8
 Discretionary Access Control, 7
 file permissions, 9
 permissions, 7
 umask, 11
 using, 7
definition of a trusted system, 1
directory permissions, 8
Discretionary Access Control (see DAC), 5
documentation conventions, xii

F

file permissions, 9
FTP, 37

H

help
 reference, xiii

I

IRIX permissions (DAC), 7

K

k5login, 32
kdestroy, 29
Kerberos, 23
Key Distribution Center, 23
kinit, 25
klist, 27
kpasswd, 30
ksu, 39

L

least privilege, 17
locked account, 19
login accounts
 locked, 19

M

man command, xiii
man pages, xiii

N

network security, 23

O

object reuse, 6

P

password, 5
 aging, 19
 generation, 5, 19
 guidelines, 18
 locked accounts, 19
 selection, 18
password, changing, 30
password aging, 19
password generation, 19
passwords, 3, 29
permissions
 changing, 10
 directory, 8
 file, 9
 umask, 11
permissions (DAC), 7
principal, 24

R

realm, 24
rlogin, 35
rsh, 38

S

SAT, 21
 system audit trail, 21
security
 policy, 2
showpwage, 19
System Audit Trail (see SAT), 6

T

TCB, 4

telnet, 33

ticket, 23, 24, 25

trust

 definition, 1

Trusted Computing Base, 4

Trusted System, 1

typographical conventions, xi

U

umask, 11

Tell Us About This Manual

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

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

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

Please send the title and part number of the document with your comments. The part number for this document is 007-3267-001.

Thank you!

Three Ways to Reach Us

- To send your comments by **electronic mail**, use either of these addresses:
 - On the Internet: techpubs@sgi.com
 - For UUCP mail (through any backbone site): *[your_site]!sgi!techpubs*
- To **fax** your comments (or annotated copies of manual pages), use this fax number: 415-965-0964
- To send your comments by **traditional mail**, use this address:

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