IRIX Commercial Security Pak™ Administrator's Guide

Document Number 007-3266-001

CONTRIBUTORS

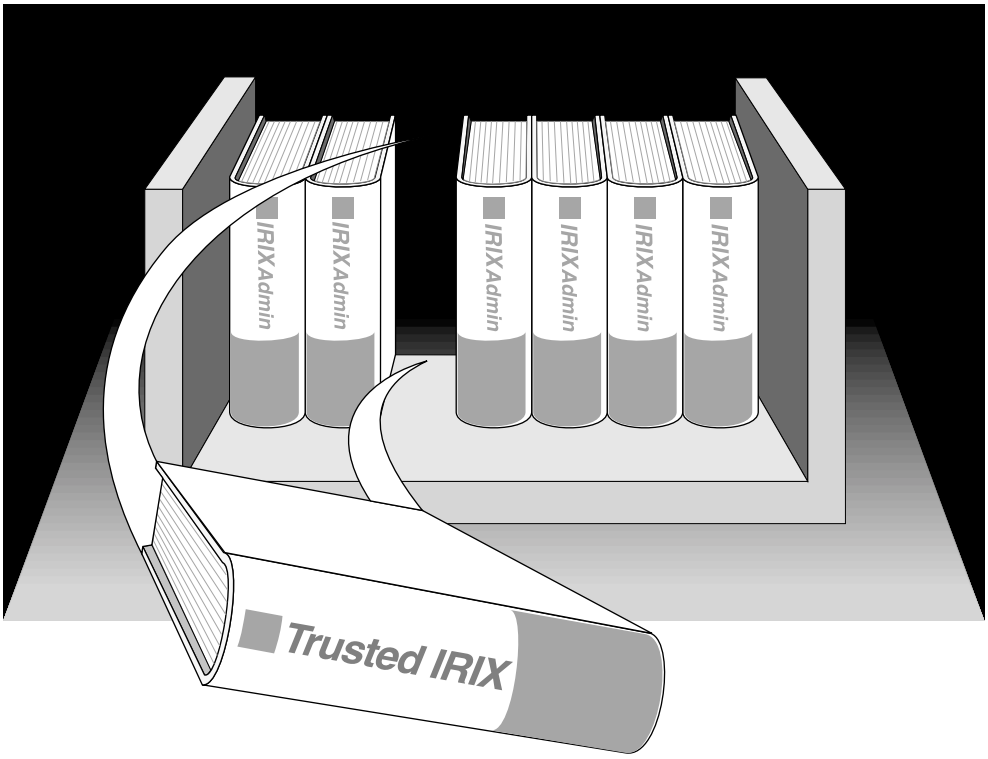
Written by Jeffrey B. Zurschmeide
Document Production by Lorrie Williams
Engineering Contributions by Ellen Desmond, Alison Gabriel-Reilly, Gary Lowell, and Casey Schaufler
St Peter's Basilica image courtesy of ENEL SpA and InfoByte SpA. Disk Thrower image courtesy of Xavier Berenguer, Animatica.

IRIX Commercial Security Pak™ Administrator's Guide
Document Number 007-3266-001

Contents

List of Tables

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 administer the special security features provided in the IRIX Commercial Security Pak™ with Silicon Graphics® workstations and servers.

If you have a graphics workstation, you may find it convenient to use the System Manager, which is described in the *Personal System Administration Guide*. That guide should be your first resource for administering graphics workstations. Regardless of whether you use the System Manager or the IRIX command-line interface, the results are the same. The System Manager does not create any new files on your system.

If you have a server, the *IRIX Admin* manual set is your primary guide to system administration, since without graphics you cannot use the System Manager.

This guide describes only those special features contained in the Commercial Security Pak software option.

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 features.

Chapter 2, "Planning Your System Security Policy"
: Provides a comprehensive discussion of the planning necessary to set up your security policies.

Chapter 3, "Administering Login Accounts"
: Provides information on the creation, maintenance, and removal of login accounts under your security policy.

Chapter 4, "Administering Access Control"
: Provides information on administering Discretionary Access Control (including Access Control Lists).

Chapter 5, "Administering the System Audit Trail"
: Provides information on the System Audit Trail.

Chapter 6, "Administering Identification and Authentication"
: Describes the Identification and Authentication procedures specific to the Commercial Security Pak.

Chapter 7, “Administering CSP-Kerberos”
: Describes the CSP-Kerberos package, distributed with the Commercial Security Pak.

Chapter 8, “System Data Files”
: Describes the system files added by the Commercial Security Pak.

Conventions Used in This Guide

These type conventions and symbols are used in this guide:

Bold	Literal command-line arguments (options/flags), nonalphabetic data types, operators, and subroutines.
Italics	Executable names, filenames, glossary entries (online, these show up as underlined), IRIX commands, manual/book titles, new terms, onscreen button names, program variables, tools, utilities, variable command-line arguments, variable coordinates, 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
[]	(Brackets) Surrounding optional syntax statement arguments

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

How to Use This Guide

This guide is written for administrators who are responsible for performing tasks beyond the reasonable scope of "end users" on IRIX systems that include the Commercial Security Pak. Frequently, people who would consider themselves end users find themselves performing advanced administrative tasks. This book has been prepared to help both the new and experienced administrator successfully perform all operations necessary to configure and maintain IRIX systems. It is hoped that people who considered themselves end users in the past will, by using this book, gain experience and confidence in successfully performing advanced system administration tasks.

Target Audience of This Guide

This guide is intended for administrators who are responsible for one or more systems running the IRIX operating system with Commercial Security Pak extensions, beyond the usual user responsibility for the user's home directory structure and immediate working directories. This guide and its companion administration guides have been written to provide directions for those who find themselves in the position of maintaining IRIX systems for themselves and others and who require more information about IRIX commands and system and network expertise.

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:

IRIX Admin Manual Set

This guide is an supplemental resource to the *IRIX Admin* manual set. This guide differs from the *IRIX Admin* documentation in certain areas and should be considered the authoritative guide for IRIX operating systems with the Commercial Security Pak extensions.

The *IRIX Admin* suite is intended for administrators: those responsible for servers, multiple systems, and file structures outside the user's home directory and immediate working directories. If you maintain systems for others or if you require more information about IRIX than is in the end-user manuals, these guides are for you. The

IRIX Admin guides are available through the IRIS InSight™ online viewing system. They are also available on the World Wide Web at http://www.sgi.com/Technology/TechPubs. The set comprises these volumes:

- *IRIX Admin: Software Installation and Licensing*—Explains how to install and license software that runs under IRIX, the Silicon Graphics implementation of the UNIX operating system. Contains instructions for performing miniroot and live installations using Inst, the command line interface to the IRIX installation utility. Identifies the licensing products that control access to restricted applications running under IRIX and refers readers to licensing product documentation.
- *IRIX Admin: System Configuration and Operation*—Lists good general system administration practices and describes system administration tasks, including configuring the operating system; managing user accounts, user processes, and disk resources; interacting with the system while in the PROM monitor; and tuning system performance.
- *IRIX Admin: Disks and Filesystems*—Describes how to add, maintain, and use disks and filesystems. Discusses how they work, their organization, and how to optimize their performance.
- *IRIX Admin: Networking and Mail*—Describes how to plan, set up, use, and maintain the networking and mail systems, including discussions of sendmail, UUCP, SLIP, and PPP.
- *IRIX Admin: Backup, Security, and Accounting*—Describes how to back up and restore files, how to protect your system's and network's security, and how to track system usage on a per-user basis.
- *IRIX Admin: Peripheral Devices*—Describes how to set up and maintain the software for peripheral devices such as terminals, modems, printers, and CD-ROM and tape drives. Also includes specifications for the associated cables for these devices.
- *IRIX Admin: Selected Reference Pages*—Provides concise reference page (manual page) information on the use of commands that may be needed while the system is down. Generally, each reference page covers one command, although some reference pages cover several closely related commands. Reference pages are available online through the *man*(1) command.

Reference Pages

The IRIX reference pages (often 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 reference pages that you use frequently 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 liststhe 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)
Miscellaneous	(5)
Demos and Games	(6)
Special Files	(7)

Release Notes

Release notes provide release-specific information about a product. Exceptions to the information in the administration guides are found in this document. Release notes are available online through the *relnotes* command. Each 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 system comes with an online 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*.

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.

Chapter 1

Introduction to the Commercial Security Pak

This guide introduces you to working with the security features in the Commercial Security Pak. This product has been designed to provide the usual security features necessary in a commercial environment. It provides features and assurance at the C2 level, according to the Department of Defense "Orange Book." The Orange Book is a common name for the 5200.28-STD Department of Defense Trusted Computer Systems Evaluation Criteria.

Commercial Security Pack Product 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 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 we use the term *trusted* rather than *secure*. 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 will circumvent the security measures or they will not use the system at all. In either case, the trusted system is rendered useless. Many users are concerned that they will not be able to 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. IRIX, with 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 IRIX 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 privilege to perform one or more of the tasks formerly reserved to the Superuser (**root**) account. This facility is used to allow users to perform system administration roles without the necessity of a specific login account for that role.

Discretionary Access Control
: This is the standard IRIX system of file and directory permissions.

Identification and Authentication
: The Commercial Security Pack 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 a Trusted System

The Commercial Security Pak 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 will be free from corruption and safe from intruders.

The most important security aspect of the system is a clear definition of the site security policy with respect to all the trusted system features listed above. To accomplish this, all system objects have been examined and altered to close potential security holes and determine a basic clearance level. This examination and revision process ensures the integrity and security of the system.

Another highly important security aspect is assurance. A secure system design must be inspected and approved by a competent agency. The Commercial Security Pak from Silicon Graphics is under evaluation for the C2 security rating from the National Computer Security Center (NCSC).

Reasons to Use the Commercial Security Pak

Many commercial sites are finding that in order to protect their investment in software and research from malicious intrusion both from inside and outside their organization, an extra measure of security is warranted. This security, though, should not impede the progress of their employees.

Ease of Use

As a modified version of an existing operating system, many of the underlying features of the Commercial Security Pak have withstood the test of time. Designing a system that promoted “ease of use” was a paramount consideration in the creation of IRIX. Silicon Graphics has a firm commitment to visual computing as evidenced by the graphical tools provided to you in the IRIX environment.

Greater User Friendliness

Part and parcel of our commitment to ease of use is by extension our commitment to “user-friendliness.” A consistent and logical framework underlies the design of Silicon Graphics visual desktop tools. This design permits even the novice user to move about the operating system with some confidence. The desktop provides a visual representation of the file system and allows you to navigate using the mouse alone.

Customer Support

You may contact Silicon Graphics customer support at 1-800-800-4SGI.

Commercial Security Pak Features

The distinguishing difference between trusted systems and nontrusted systems is the security-enhanced feature set. In the Commercial Security Pak, this feature set includes three main components: improved identification and authentication of users, auditing, and access control (ACLs and Capabilities).

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, or 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.

Identification and Authentication

The Identification and Authentication 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 Under 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.

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 executable files. A capability requirement can be imposed on any executable file — without the corresponding capability endorsement, no user can access the file. The capability endorsements are made separately for each user. This allows the Site Security Officer to allow only certain users to access system programs and to designate certain users to perform system administration tasks — to fulfill the system administration roles without requiring a special privileged login account (**root**) for the role.

Discretionary Access Control Permissions

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, save only **root**, to view, copy, or edit that file. Default DAC permissions for newly created files are set via the *umask*(1) command.

Thus, to gain access to a file owned by another user, the owner must have set the DAC permissions on the file to allow others to access it. Typically, DAC permissions would be 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 is allowed through traditional discretionary access control.

System Audit Trail

A foundation of system security is the 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.

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

The audit subsystem allows the administrator to create a dynamic record of the system's activity. This record allows the administrator to hold each user strictly accountable for his or her actions. The audit system is completely configurable at any time by the audit administrator.

Audit information must be carefully gathered and protected so that actions affecting security can be traced to the responsible party. IRIX 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.

The auditing process is transparent to the user. It is important to recognize that when working on a trusted system, your actions will be audited. You should not, however, be apprehensive or fearful of the auditing process. It is not used to spy on individual users or to trap you in any way. Its function is to protect you from others who may try to use your identity for mischief.

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 before being reallocated.

Chapter 2

Planning Your System Security Policy

Before you begin general use of your system, define your needs and examine and classify the various types of information and applications that will reside on your system. Although it is always possible to reconfigure your system resources and practices, your installation will benefit from planning. A "dry run" of your trusted system is often beneficial before classified data and users are allowed access.

This chapter covers the planning necessary to ensure a trouble-free and efficient trusted system implementation. If you do not plan your system, you are likely to forget to implement some necessary security policies or labels. While the system is fully configurable at any time, it is inconvenient to have to search through your system and fix existing files and user accounts. Time spent in planning will be much shorter than time spent in repairing and updating.

Planning Your Administrative Accounts

An important difference between the Commercial Security Pak and earlier IRIX product offerings is the capabilities mechanism. This mechanism provides the Site Security Officer with the ability to configure the system so that the Superuser powers are divided and distributed among several individuals.

This represents a significant departure from traditional UNIX system administration practice. Because you can configure the system to have no omnipotent Superuser, administrative habits may have to be changed to accommodate the stricter controls. Dividing the traditional Superuser into a set of administrative accounts does not make system administration simpler. Several people must be well trusted, though no one need be trusted absolutely with the integrity of the system.

One case where the complexity is evident is the creation of new user accounts. The System Administrator is responsible for maintaining the */etc/passwd* file, but does not maintain the */etc/capability* or */etc/shadow* files.

The administrative accounts are shown in Table 2-1.

Table 2-1 Commercial Security Pack Administrative Accounts

Account Name	Role	Capabilities
dbadmin	System Administrator	none
ops	Operator	none
sso	Site Security Officer	mac_read mac_write mac_relabel_open mac_upgrade mac_downgrade dac_read_search dac_write fowner setfpriv audit_write audit_control kill
auditor	Auditor	audit_write audit_control kill

It is not uncommon for one person to have responsibility for more than one of these accounts, although it is desirable to have one person for each role. These people should also hold normal user accounts for other work. Each administrative account holds responsibility for some aspect of the system.

System Administrator

The System Administrator uses an account called **dbadmin**. The System Administrator is responsible for day-to-day maintenance tasks and for installing new software for the system.

The System Administrator is also responsible for maintaining the functional state of the system. This task includes, but it not limited to, maintaining these configuration database files:

/etc/aliases	the configuration file for e-mail groups
/etc/exports	the configuration file for directories exported using NFS™
/etc/fstab	the configuration file for filesystems mounted both locally and via NFS
/etc/group	the configuration file for group name-to-group ID mapping
/etc/hosts	the configuration file for system-to-Internet address mapping
/etc/motd	the login message file
/etc/networks	the configuration file for network-to-Internet address mapping
/etc/passwd	the configuration file for user name-to-user ID mapping
/etc/sendmail.cf	the configuration file for the e-mail environment on this system
/etc/sys_id	the file that holds the name of this system

The System Administrator is also responsible to the users for necessary assistance, services, and maintenance.

Auditor

The person responsible for reviewing the System Audit Trail is called the Auditor and uses an account named **auditor**. The Auditor is allowed sufficient capability to manipulate the audit trail and to terminate any process. This account and role are properly part of the duties of the Site Security Officer, and can be safely entrusted to that individual.

The Auditor reviews and analyzes the audit logs. When the Auditor finds cause to believe that a security breach has occurred, he or she should notify the Site Security Officer for action. The Auditor may choose to further process the data into a form that is appropriate for analysis and long-term storage, using the reduction tool *sat_reduce*(1M) and the formatting tool *sat_interpret*(1M) .

The Auditor should be responsible for suggesting changes to the security policies and should take the initiative in enhancing the auditing level when that action is deemed necessary.

Operator

The Operator is responsible for day-to-day system state changes. This primarily includes making backups. The Operator does not determine or maintain system policies.

Site Security Officer

The Site Security Officer is responsible primarily for initiating, monitoring, and approving changes to the system. This individual should implement the security policies. For example, the Site Security Officer should monitor and maintain the password generation and change system and ensure that all users comply with the password policy.

The Site Security Officer also checks all new programs and files for security violations before they enter the TCB. It should be noted here that sites that wish to run the security-evaluated configuration must never add any new software to the TCB, although new software can be safely added if it is not made part of the TCB. The Site Security Officer must make a determination as to whether adding a program to the TCB is worth the risk involved with having a system that no longer matches the evaluated configuration.

Superuser

IRIX supports a least privilege mechanism through the implementation of POSIX *capabilities*, described in the section titled "Capabilities" in Chapter 4. The Superuser privileges have been broken out into a set of distinct capabilities, which can be granted and relinquished through a set of inheritance rules. The several versions of IRIX support three capability styles.

Traditional Superuser
: In the traditional Superuser style, privilege is defined by having an effective userid of 0 (**root)**. This style is best suited to friendly, cooperative, and open environments.

Augmented Superuser
: In the augmented Superuser style, a process with an effective user ID of 0 (**root**) is privileged. It is also possible to grant processes distinct capabilities, even if they do not have Superuser privilege. This style is best suited to open environments in which individuals require the ability to perform actions such as reserving system resources, rebooting

	the system, or setting the clock, but not some of the other powers normally associated with the Superuser, such as reading other people's mail.
No Superuser	In the no Superuser style, an effective user ID of 0 (**root**) has no privilege implications. Only the presence of capabilities provides privilege.

In all cases, **root** is the owner of most system administrative databases. System startup is done under this user ID as well, so even if you choose the no Superuser style, do not delete the **root** account.

How the Administrative Accounts Work Together

When all of your administrative accounts and roles are used properly, you have created a system to guard against failure of your security policy.

For example, to add a new user, the site manager or project manager requests the new user account from the Site Security Officer. When the request is approved, the **auditor**, **dbadmin**, **ops**, and **root** are notified so that the auditing can be changed if necessary, the user's account created, and the Superuser notified of the change in system usage. The process should happen in this manner:

1. The SSO approves the account request.
2. The System Administrator creates the account using the System Manager.
3. When the account is created, the System Administrator notifies the Site Security Officer of the new account. The SSO notifies the Auditor (if they are different people).
4. The Auditor sets up any ordinary or special auditing required by the Site Security Policy.
5. The Site Security Officer checks the work of the System Administrator and Auditor and then notifies the user that the account is open.

As another example, the same process applies when adding a new program to the TCB. The Site Security Officer reviews the program for security violations, the **auditor** arranges to audit the program's activity, **dbadmin** actually installs the approved program, and **root** alters the system configuration for the new program.

If a user is found in violation of the security policies or an intruder is discovered, a user or the **auditor** notifies the Site Security Officer, who first notifies **dbadmin** to lock the

account or security hole and then notifies **root** to repair the damage to the system. The **auditor** also increases the auditing of the suspect or violated area to increase security.

To remove a user or finish a project, the Site Security Officer determines the changes necessary to the system configuration and instructs the other administrative users to take the appropriate action to remove the user account and to make changes to the auditing process and the system configuration.

Creating Security Policies

You must establish an overall security policy for your site, called a Site Security Policy. A security policy is a series of official statements regarding the rules for the use of the system. The purpose of the policy is to create a clear code to ensure the safe use of the system. This policy should be clearly articulated in a written document available to all users. Each person involved with the secure system should know his or her own security guidelines and responsibilities. At some sites, the security policy itself may be classified.

In 1987, the U.S. Computer Security Act mandated that secure computing sites are responsible for the integrity and confidentiality of their resources and information. Computer systems with sensitive information have long been a favorite attack point for malicious and mischievous intruders. Because of the nature of computers, intruders can do much of their work without direct contact with a human being. Damage done to computers and the information they store can come in many forms. Sometimes the damage is severe, when irreplaceable information is lost or corrupted. Sometimes the damage is a case of theft, when a business competitor seeks advance knowledge of product development. Occasionally, users themselves damage the system, either accidentally or with malice.

A total security policy has a number of segments. Among these are the physical security policy, the procedural security policy, and the system security policy. Take care to ensure that your security policies are concordant and similar in approach to one another. If possible, for consistency, the same person or group should draft all security policies. Avoid complexity in your policies; it can cause users to become confused or to circumvent policy.

Physical Security Policy

A physical security policy is simply the security measures that protect the computer hardware from damage or unauthorized access. Damage in this sense can come from intruders or from aspects of the location, such as water damage or electrical power fluctuations. All physical components of your system, such as the central processing unit (CPU), any storage media, wiring, and remote terminals need to be governed by the physical security policy. Guidelines for effective physical security are as follows:

- Set the PROM password on your system. Instructions for this can be found in your *Owner's Guide* or in the *IRIX Admin: System Configuration and Operation* guide.
- Keep the system physically secure at all times, such as in a locked or guarded room.
- Restrict access to the room to those with immediate need to use the system.
- Request that all users clear the video screen upon finishing work.
- Maintain reasonable security against unauthorized and unrecorded entrance to the entire building or site where a trusted system is used. Such security can be in the form of keyed entry or other clear identification for authorized users.
- Shield and protect all wiring and cables, especially network connections and terminal lines. The lines should be physically covered and unavailable. In no case should any part of the wiring be connected to unsecure systems or any area outside the secure site.
- Keep physically secure all archive media and other data stored on magnetic media. Store it in a locked or guarded media library room. Segregate all media according to the classification of the information contained.
- Restrict access to the media library or libraries to the System Administrator.
- Remove, erase, or destroy obsolete data as soon as possible.
- Shred or otherwise destroy paper output when it is no longer needed.

The most secure software is of no use if your physical hardware is vulnerable. Therefore, when you are planning your system, take note of the location of the hardware. The computer itself should be located behind locked doors, and the number of people with access to the room should be strictly limited. Only users, System Administrators, and other responsible people should be allowed in the room at any time. This security should never be relaxed. The power source for the computer should always be protected so that the computer is not subject to power surges or momentary power outages. The location of the computer should have a limited number of windows or be totally enclosed.

Beyond this, any coaxial cable connections to other computers on the secure system should be within the same restricted area. If it is necessary to connect to another computer outside the restricted area, the coaxial cable should be routed in such a manner as to avoid convenient points where the cable may be tapped by an intruder.

Peripheral hardware you attach to your system must likewise be protected from intruders. If you use a storage media device such as a cartridge tape drive, you must store the cartridge tapes after use with as much attention, if not more, to security as any other portion of the system. If a cartridge tape falls into the hands of an intruder, it is a simple matter to extract all the information onto a different system. Printers must also be closely monitored, because once information has been printed, it is available to anyone who can read it. Information import and export devices are the weakest link in your trusted system.

System (PROM) Passwords

Silicon Graphics workstations and servers support system passwords. These are also often called PROM passwords, since the firmware that supports the passwording is stored in PROM chips on the system's CPU board. It is strongly recommended that all systems make use of PROM passwords.

These passwords are demanded of the user attempting to access any part of the system while the operating system is not running. For example, with most computers, if you press the hardware reset button, the computer offers you a chance to perform system maintenance before the system reboots. Or in some cases, if you insert an installation tape or floppy disk, the system boots from that media instead of booting the usual operating system. Once this has been accomplished, it is a trivial matter for the intruder to mount the disk or disks and gain access to all your files. Since the system password is required before the system does anything but boot the usual operating system, overall security is increased.

The methods for setting the PROM password differ from system to system. To set the system password, see your system's owner's guide, or the *IRIX Admin: System Configuration and Operation* guide. The owner's guide can also instruct you on procedures to follow to remove the password if you forget it.

Procedural Security Policy

The procedural policy is the segment of the security policy that dictates how the system is used. It should cover the responsibilities of each administrative user, such as the Site Security Officer (**sso**), Auditor (**auditor**), System Administrator (**dbadmin**), and the Superuser (**root**), and the responsibilities of ordinary users. Guidelines for procedural policy should list decisions concerning:

- Who may use the trusted system and what operations each person is allowed to perform. Keep the number of users as small as possible.
- What system information is available, and through what mechanisms it is available. For example, it is wise to limit the amount of sensitive data that may be printed on paper.
- How information is to be handled at all times.
- How to dispose of old information. Bear in mind that technology exists to read "erased" information from magnetic media. All physical copies of sensitive information should be destroyed rather than simply erased.
- How the system-provided security features are used. For example, you should decide how thorough the auditing of user activities must be, and how often the auditor reviews the audit logs. The responsibilities of the auditor are described later in this chapter.
- How often and at what level system backups are to occur.
- A procedure and schedule for storage, archiving, retrieval, and disposal of system data.
- A procedure for retiring user accounts for discontinued users.

Procedures used at your site for day-to-day activities can make or break your system security. If procedures are followed, your system security policy is much more likely to succeed than if procedures are lax.

System Security Policy

The system security policy is closely related to both the physical security policy and the procedural security policy. It draws on both the physical and procedural policies to create a total policy for the system. Some guidelines for an effective system security policy are given below:

- Uniquely identify all users. In no instance should a user be allowed to share an account or any identification with any other person.
- Authenticate each login attempt with a password. Each user account must have a unique password. This password is required for every login.
- Change each user's password frequently. Facilities exist within the Commercial Security Pak to generate sound passwords that are difficult for a potential intruder to guess or discover through systematic trial and error.
- Authorize each user to perform only those tasks and to view only the information that he or she needs to complete his or her work, and no more. By reducing the scope of each user account, you reduce the possibility of general damage by an intruder who gains access to a single account.
- Adequately audit each user during every login session. Determine the amount of audit recording necessary to ensure a reasonable knowledge of system activity at all times. Events such as login time, logout time, the creation, modification, and deletion of files, and the invocation of TCB programs should always be audited.
- Educate your users, System Administrators, and those who are present at your site but who are not users of the trusted system. Everyone, not only the managers and System Administrators, can take responsibility for security.
- Publish the security policies if possible, and make everyone at the site aware of security issues.
- Review the security policies regularly and make necessary changes. As your system works and changes, modifications to the policy may become necessary. A breach of security may necessitate tighter controls and changes to the policy. However, excessively tight controls may deny access and encourage misuse of the system.
- If possible, publish each change to the security policy clearly to all persons at the secure site.
- To ensure compliance, make certain that all levels of management understand and approve each policy.

- Assign a Site Security Officer who is the central figure responsible for security issues.
- Keep the security policy consistent with the goals and standards of your company.
- Do not make the security policies more rigid than necessary. A policy that is unrealistic is likely to be ignored or circumvented by unhappy users.

System security provides for the entire trusted system. The trusted system includes not only the hardware and software but each individual and the group as a whole. All of these combine to form a secure system for the safe development and storage of your sensitive information.

Planning for Users

During preinstallation planning, determine how many user accounts will be necessary for the people who will work on your system. You need to know how many users to expect and how much disk and memory space they require in order to make informed decisions regarding hardware resources.

Plan the capabilities required to access key system executable files, and which users can access those files. The users with these capabilities are those who have been selected to perform the system administration roles described in “Planning Your Administrative Accounts.” Plan Access Control Lists in working and home directories at this time as well. Information on Access Control Lists can be found in the section titled “Access Control Lists (ACLs)” in Chapter 4.

Examine your hardware at this time to determine how much disk space the users and information are likely to need. Planning adequate disk allocation at the beginning saves having to reinstall later if your system needs to grow. Include provisions for system audit requirements in system planning. The Audit Trail requires disk space. The specific amount required depends on the level of auditing configured and the amount of system activity. Audit Trail requirements are discussed in detail in Chapter 5, “Administering the System Audit Trail.”

Planning for Auditing

While you are planning your system, plan to use the System Audit Trail features effectively. Decide what kinds of events you wish to audit and how much information you wish to store. If disk space is limited, you may wish to restrict auditing to events such as file removal, attempts to access the system, and denial of service. You must also budget the time of the Auditor to review and reduce the audit log to evaluate the security status of your system.

Installation Notes

Installation of the Commercial Security Pak is similar to installing any software option. Installation instructions are found in the *IRIX Admin: Software Installation and Licensing* guide and in the *release notes* for this release.

Deactivating a System

When the time comes to deactivate a particular computer from your system, or perhaps deactivate your entire site, you should take certain steps to ensure that information is not inadvertently disclosed:

- All magnetic media not being maintained in a secure manner must be thoroughly erased or destroyed. Except for project records and results that must be maintained, destroy all backup media and other records that might be salvaged if they were merely discarded.
- All accounts on all systems must be thoroughly deactivated.
- All disks on all systems must be reformatted or destroyed.

Chapter 3

Administering Login Accounts

Two classes of login accounts are found under IRIX using the Commercial Security Pak: *user* accounts and *administrative* accounts. The administrative accounts are **sso** (the Site Security Officer), **ops** (the Operator), **root** (the Superuser), **dbadmin** (The System Administrator), and **auditor** (the Auditor). All other accounts are ordinary user accounts. This chapter discusses the appropriate use and management of user and administrative accounts.

User accounts are at once the first line of defense of a trusted system and potentially the weakest link in that system. Every user account can break system security if it is not managed well, and every user account can be used to enforce system security. The way your user accounts are managed is crucial to a successful secure system.

Users must have ready access to the files and resources they need to perform their work. If this access is not available or is inconvenient, users circumvent the security policies and create threats to system security. However, users should also not be allowed access to unnecessary files and resources, as this is a security threat in itself.

Guidelines for effective secure management of user accounts are explained in this chapter. Procedures for administering user accounts and user groups are also presented.

Creating User Accounts

The following sections give guidelines and instructions for creating user accounts.

Guidelines for User Accounts

Guidelines for user account administrators are listed below:

- Always use a different account name and user ID number for each user on your system. Each account should represent only one person, for accountability.
- Always create passwords for all accounts on your system.

- Never assign a login name that begins with a number. Some networks do not interpret these login names correctly.
- Always choose unique user identification names for your users. For example, the login name *steveb* is a better choice than *user001*. A login name and the other information associated with an account should always be readily associated with the person who owns that account. It is generally possible to find distinguishing personal characteristics to differentiate between two or more users with similar names.
- Include the user's full name and some personal identification, such as job title and phone number, in the comment field of the */etc/passwd* file. Be careful, however, not to include classified information in the */etc/passwd* file.
- Be certain that the user's environment is properly initialized for security. For example, in the *.profile* or *.cshrc* files, set the user's UMASK to 077. This initializes the default DAC permissions to allow the user to access only those files he or she creates.
- In the *.profile* or *.cshrc* files, set the PATH environment variable to include only those directories that the user is allowed to access. Also, in the PATH variable, make certain that the user's home directory is searched last, after the system directories, for commands. This guards against some forms of Trojan Horse attack. Do not include any temporary or public directories in the PATH, such as */tmp*.
- If possible, place a copy of the security policy in each account.
- When you remove a user account, first make a backup tape of all files in the home directory belonging to that account.
- When you remove a user account, assign new owners to any files on the system still owned by the removed user.

Creating User Accounts

This section gives directions on creating user accounts. Choose the user's login name, user ID number, and any administrative roles before beginning the process.

On a trusted system, shadow passwords (*/etc/shadow*) are always used (see *pwconv*(1).) When capabilities are installed, each user who is allowed to acquire capabilities on demand must have an entry in the */etc/capability* file. All of these databases, except */etc/passwd*, are protected from perusal by non-privileged users.

It is important to follow the procedures exactly as they are specified in this guide. These procedures often involve manipulating sensitive system access files. Failure to follow the exact procedures listed here could leave your system without the designed security protections.

Administrative Roles

When you are using the No Superuser capability style, you must assign at least one user each of the system roles. When you are using the augmented Superuser capability style you should assign users to the system roles. When you are using the traditional Superuser capability style you may assign users to the system roles. The Superuser is responsible for everything, and is all powerful within the bounds of the system. When you are using the No Superuser capability style, this level of privilege does not exist.

The Site Security Officer is responsible for the enforcement of the system security policy. The Site Security Officer has UID 0, comparable to **root**, and is allowed to:

- add entries to the user clearance and capability databases
- set capability sets on executable files
- shut down the system
- view and manipulate audit trail files
- enable and disable auditing of specific events
- kill any process
- restore backups

A permitted sub-role of the Site Security Officer is the Auditor. The Auditor is responsible for the audit trail only, and is allowed to:

- view and manipulate audit trail files
- enable and disable auditing of specific events
- kill any process

The System Administrator is responsible for all aspects of the system configuration except those related to the system security policy. For example, the System Administrator cannot add a user to the system without the help of the Site Security Officer, who is empowered to grant clearances. The System Administrator is allowed to:

- add entries to any system database except the user capability database
- manage the network configuration

The Operator is responsible for daily system maintenance activities. The Operator is allowed to

- create backups
- cancel print jobs

To summarize, the Site Security Officer sets security attributes, the Auditor watches what goes on, the System Administrator configures the system, and the Operator performs backups.

All non-security-related system administration should be done using the System Manager tools. These tools verify that the invoking user is allowed to perform the necessary role.

Creating Normal User Accounts

The System Administrator creates all new user accounts using the System Manager. Additionally, you should set a password for the new user as follows. Give these commands:

```
passwd username
passwd -f username
```

The first command creates a password for *username*. This password must be selected by the System Administrator and told to the new user. The second command forces the new user to change the password at the first login.

Removing a User

When a user has finished all use of a secure system, that user's account should be closed quickly. It is primarily the Site Security Officer's concern that unauthorized users not be allowed on the system, so the Site Security Officer needs to be informed at once when a user leaves or ceases to use the system. The Site Security Officer should replace the former user's encrypted password field (in */etc/shadow*) with the string "*INVALID*" and both capability lists in */etc/capability* with the string "*INVALID*." The entries in the files should not be removed. The Site Security Officer should also check for *crontabs*, *at* jobs, or print jobs the former user may have queued.

Once the user is removed, check all system files and change ownership of any files on the system that are owned by the defunct user account. If the user had access to other accounts, change the passwords on those accounts immediately. Also, remove the user's name from all groups on the system.

User Groups

On a trusted system, you typically have one or more confidential projects at any given time. Also typically, the users working on those projects need to share files and resources. To accommodate this need, you can create user groups. DAC provides a set of permissions for a file owner's group, as well as for the owner of the file and the whole user community.

IRIX provides for multiple concurrent groups. That is, a particular user can be a member of any number of groups, or even of all groups on your system. When you log in, your group ID is set to the group ID in your entry in the *passwd* file. To change to a different group, use the *newgroup*(1) command.

Group your users based on their common needs. Put all the users on a given project in the same group. All members of a group acquire the group ID in addition to their user ID when they log in. Using the DAC permissions, it is possible to give each member of a project team complete access to necessary files and exclude other users from confidential files.

Guidelines for User Groups

Suggested guidelines for user groups are:

- Place users working on the same project or who have similar needs in a group. Consider, for example, a group of data entry clerks. Users with similar needs may work on different projects, but they all need similar tools and resources.
- Add a group at the same time you add each new project to your system.
- Assign each group a unique and readily identifiable group name. For example, *motordev* is a better name than *group001*.
- Never begin a group name with a number, as this can be misinterpreted by the system.
- The file */etc/group* maintains a list of the valid groups and their members. It is possible to edit the */etc/passwd* file and change the ID number of a given group. No checking is done between these two files, and the System Administrator must make certain that all user IDs and group IDs given in these files are correct.
- Run the *pwck* program frequently to check your system for potential problems in the */etc/password* file.
- It is sometimes desirable to create a group containing only a single user who is performing specialized work.
- If you are creating new user accounts and a specialized group for those users, be sure to create the group entry before allowing any of the users to log in.

Adding a New Group

To add a group, the System Administrator logs in and creates the group using the System Manager.

Removing a Group

When a group has no more users, or a project group has finished all work, the group should be nullified. You should not, however, remove a group entirely, as the possibility exists that the same group name or ID number might be reused, creating a security hazard. To remove a group, use the System Manager to edit the */etc/group* file in the same way as to add a group, and remove all usernames from the entry for the defunct group. This way, the group is effectively removed, but the entry remains and so cannot be reused.

At your convenience, search through the system and find files that are owned by the defunct group and change their ownership to another group or remove them.

Performing the Superuser Role

The amount of administration that can be done by a single individual is significantly influenced by the capability style chosen for a system. As a rule, the more you care about protecting your information, the less you want to have a single individual with complete control of the system.

When you use the Traditional Superuser style, the only way for a process to have privilege is for it to have an effective user ID of 0, which corresponds to the user name **root** or **sso**. All privileged processes have universal privilege, and must be trusted to use only those which are appropriate to the task at hand. The *setuid* mechanism can be used to grant an unprivileged user privilege by setting the effective user ID.

In the Augmented Superuser style, a process can have privilege either because its effective user ID is **root** or because it has one or more capabilities. Processes with a user ID of **root** have universal privilege and must be trusted to use only those that are appropriate to the task at hand. Processes with a set of capabilities have limited privilege and need only be trusted to use those correctly. The *setuid* mechanism can be used in the traditional way, or a file capability set can be used to explicitly grant or deny privileges available to invocations of the program.

In the No Superuser style, a process can have privilege only when it has one or more capabilities. Processes with a user ID of **root** or **sso** have no special privilege whatsoever. Processes have limited privilege and need only be trusted to use those correctly. The *setuid* mechanism can be used in the traditional way, but the effective user ID of **root** is not considered sufficient to grant privilege. A file capability set can be used to explicitly grant or deny privileges available to invocations of the program.

The Superuser role is obviously dependent on which Superuser style you are using. In the Traditional style, the Superuser may, and in fact usually must, perform any system administration activity desired. In the Augmented style the Superuser may perform any system administration activity, but other processes performing other roles may exist as well. In the No Superuser style, the Superuser role is not used.

The **root** user ID owns most of the system administrative data files. The **root** user ID is associated with the System Administrator role. If you are using either of the Traditional or Augmented styles, the Superuser role can be thought of as a System Administrator role with extended powers. If you are using the No Superuser style, the Superuser role is replaced by the other, distinct roles.

Performing the Site Security Officer Role

The Site Security Officer (SSO) is responsible for maintaining the system security policy. This policy is defined by a set of system configuration database files:

/etc/capability	User capabilities.
/etc/shadow	Encrypted passwords.
Kerberos Files	All Kerberos-related files.

Operations Performed by the Site Security Officer

The **sso** performs running system maintenance that you define on your system. Most maintenance of this kind, such as killing defunct processes and resetting hardware, is performed as a service for users.

Performing the System Administrator Role

The System Administrator (**dbadmin**) is responsible for maintaining the functional state of the system. This task includes, but it not limited to, maintaining these files:

/etc/aliases	A configuration file for e-mail groups.
/etc/exports	A configuration file for directories exported through NFS.
/etc/fstab	A configuration file for filesystems mounted both locally and through NFS.
/etc/group	A configuration file for group name to group ID mapping.
/etc/hosts	A configuration file for system to Internet address mapping.
/etc/motd	The login message file.
/etc/networks	A configuration file for network to Internet address mapping.
/etc/passwd	A configuration file for user name to user ID mapping.
/etc/sendmail.cf	A configuration file for the e-mail environment on this system.
/etc/sys_id	A file that holds the name of this system.

Performing the Operator Role

The Operator is responsible for day-to-day system state changes. This includes making backups and cancelling print jobs. The operator does not determine or maintain system policies.

Performing the Auditor Role

The Auditor role is used to manage the System Audit Trail. At installation time, the Auditor is responsible for setting up the audit subsystem and seeing that it is activated on the system. Also, the nature and extent of auditing is controlled through the *sat_select*(1M) command.

The Auditor role includes the ability to modify system audit parameters and to *kill*(1) any process on the system in response to events that might appear in the audit trail. Through Discretionary Access Control, the Auditor is allowed to access files that no other user is allowed to change, but DAC also prevents the Auditor from accessing any user files not normally accessible.

Once the system is set up, the auditor is responsible for storing the audit files and periodically viewing and editing the files to produce a record of the system usage. A number of tools are available for use with the system audit trail, including these:

sat_select(1M)

sat_echo(1M)

sat_interpret(1M)

sat_reduce(1M)

sat_summarize(1M)

satctl(2)

satread(2)

satwrite(2)

sat_eventtostr(3)

Each tool is fully described in its own reference page.

Using these tools, the auditor creates records of all system activity. It is also the job of the auditor to review the audit logs periodically to check for security violations or notable activity. These are the types of activities to be flagged for observation:

- attempts at unauthorized entry
- system usage at unusual hours or from unusual locations
- attempts at access control violations
- unexpected use of privilege
- connections with systems outside of the local network
- any activity by particularly interesting subjects
- any accesses of particularly interesting objects
- modifications of system data files
- manipulation of the audit trail itself

Any notable instances of user behavior discovered by the auditor should be immediately brought to the attention of the Site Security Officer.

For a complete discussion of auditing under IRIX, please refer to the guide titled *IRIX Admin: Backup, Security, and Accounting*.

Chapter 4

Administering Access Control

Access control under the Commercial Security Pak has been described in general earlier in this guide. This chapter contains a detailed description of the Discretionary access control mechanisms. The Discretionary Access Control mechanisms are the standard system of file permissions, the use of Access Control Lists on files and directories, and the use of Capability attributes on system executable files.

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 added feature of Access Control Lists is implemented in the Commercial Security Pack. 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. Further, the user may set an Access Control List (ACL) for any file or directory. ACLs are discussed completely below in the section titled "Access Control Lists (ACLs)."

The Commercial Security Pak allows users to control access to their own files and change that access at will. The only user who can override those access decisions is the Superuser (**root**). Thus, DAC fills an otherwise unmet need for system security at the personal level.

DAC Permissions

IRIX divides permissions into three categories, and users into three relationships. The three relationships are the owner of the file, the owner's user group, and all users. If you view a long listing of a directory, you see that the permissions field for each file in the directory looks something like this:

```
-rwxrwxrwx
```

Note that the line of permissions has the string rwx repeated three times. The first instance of rwx applies to the file owner, the next instance applies to the group members, and the third applies to all other users on the system. The example above shows full permissions. A more restricted permission set might look like this:

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

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, enter:

```
ls -l
```

at your system prompt in any directory. Along with the permission information, the *ls -l* command lists the owners and the sizes of the files and the date they were last modified.

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.

Each character is separately significant in the permissions listing. Starting at the left, the first character is a dash. A dash in any other position 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 a file or a directory. If it is a directory, a "d" appears in that space. Other characters in this place indicate that the file is a pipe, a block or character special device file, or other type of file. See the *ls*(1) reference page.

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 after the leftmost place in the permissions field describe 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 leftmost 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 describe permissions for the owner's group. In this case, the group is *grp*. Suppose permissions for this file were slightly different, like this:

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

In that case, any member of the group *grp* could read or execute the file, but 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 doc 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. 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 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 in this directory have varying permissions. Some are restricted to 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 is executable by any user.

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 that need to be used by others. The command to restrict access to a file or directory to yourself only is this:

chmod 600 *filename*

chmod 700 *dirname*

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 this:

chmod +w *filename*

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

Setting Permissions With umask

You can decide what default permissions your files have by placing the *umask* command in your *.cshrc*, *.profile*, or *.login* file. There is a default *umask* setting for the entire system in the */etc/profile* and */etc/cshrc* files. By changing the setting of your *umask*, you can alter the default permissions on your files and directories to any available DAC permission. See the *umask*(1) reference page for more information.

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, you might want the personal files you keep in your home directory 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 no other user would be allowed into the directory.

You can also use the IRIX utilities 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 to any files or directories you indicate every time you log out.

For example, say you have three directories, called *personal*, *letters*, and *budget*. You can set up a *.logout* file in your home directory with commands to be executed each time you log out from the system. The following commands, placed in the *.logout* file will prevent access to the three example directories to anyone but you:

```
chmod 700 budget personal letters
chmod 600 budget/* personal/* letters/*
```

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 your account up to allow group read and write privileges and no other privileges, place this line in your *.cshrc* or *.profile* file:

```
umask 006
```

This makes every file you create have the following permissions:

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

With your *umask* set to 006, directories that you create have the following permissions:

```
drwxrwx---
```

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

Access Control Lists (ACLs)

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 file permissions to successfully gain access. If you have not created an access ACL for a file, the default ACL serves both ACL functions.

Hereafter in this section, directories are treated as files, and where the term file is used, consider that 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*

The command above produces output similar to this:

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

This example shows full permissions for the owner with the first entry on the line, sets read permission for user ID 332 with the second entry, and sets read/write permission for the user account ernie. The specific format of an ACL entry is discussed in the section titled "Long ACL Text Form."

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 separated by commas. 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. To read an ACL from an object, a process must have read access to the file.To create or change an ACL, the process must own the file.

ACLs have long and short text forms. The long text form is defined first in order 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 ACL Text Form

The long text form is used for either input or output of ACLs and is set up 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	Access granted to either the file owner or to a specified user account.
group	Access granted to either the file owning user group or to a specified user group.
other	Access granted to any process that does not match any user, group, or implementation-defined ACL entries.
mask	Maximum access that 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	User account name or a user ID number.
gid	User group name or a group ID number.
empty	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 discretionary access permissions field must contain exactly one each of the following characters in the following order:

r	Read access.
w	Write access.
x	Execute access.

Any or all of these may be replaced by the no-access dash(–).

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 *umask* and other entries contain an empty qualifier. A crosshatch (#) 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 *umask* entry, then the output text form for that entry must be displayed as described above followed by a crosshatch (#), 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 crosshatch (#) comment character
- at any point after the first crosshatch (#) 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 with the first entry on the line, sets read permission for user ID 332 with 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 ACL Text Form

The short text form is used by the *chacl*(1) command for input of ACLs, and is set up 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 command line contains one ACL entry, with the exception that 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 characters 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
```

Using ls -D and chacl

You can use the output from the *ls -D* command as the input to *chacl*. This is convenient for situations where you wish to duplicate a complex custom ACL onto a new file in a directory that does not use the complex ACL as the default.

Consider this example:

`ls -dD` *testdir*

The command given above produces the following output:

```
testdir [u::rwx,g::r-x,o::--x/u::rwx,g::r-x,o::---]
```

Create a new directory (it doesn't matter where) with this command:

`mkdir` *newdir*

Then use the following command to edit and copy the ACL (give this command all on one line):

```
chacl -b `ls -dD testdir | cut -d"[" -f2 |  cut -d"/" -f1` `ls -dD
testdir | cut -d"[" -f2 |  cut -d"/" -f2 | cut -d"]" -f1` newdir
```

The ACL from *testdir* will be replicated in *newdir*. Note that the *cut*(1) command is used within the above command line. For complete information on the correct use of *cut* in any command line, see the *cut*(1) reference page.

After giving the above command, an ACL listing of *newdir* shows that the ACL from *testdir* has been duplicated:

`ls -dD` *newdir*

```
newdir [u::rwx,g::r-x,o::--x/u::rwx,g::r-x,o::---]
```

Note that the cut and paste functions of the window manager can also be used to duplicate ACL entries from *ls -D* to *chacl*.

Capabilities

Capabilities are privileges assigned to specific accounts to allow those accounts to perform operations formerly reserved to the Superuser. 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.

The fundamental purpose of capabilities is to allow you 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.

It is usually inappropriate to grant capabilities to ordinary users of the system. Should you decide to do so, remember the principle of least privilege: A user should have only those capabilities for which a need can be demonstrated and no others.

Capabilities provide fine-grained control over the privileges of a process. A process can be granted specific capabilities to perform privileged system calls, but not be granted general override of the system's protection scheme as is the case with a setuid **root** program. The IRIX capability mechanism is designed to comply with Draft 15 of the POSIX P1003.1e Draft 15 specification.

The /etc/capability File

The file */etc/capability* is the database of capabilities for user accounts. Here is a sample */etc/capability* file:

```
root:all+eip:all+eip
auditor:CAP_AUDIT_WRITE,CAP_AUDIT_CONTROL,CAP_KILL+eip
ernie:all=:CAP_FOWNER,CAP_SETFCAP+eip
casey:all=:all+eip # We trust Casey.
jeff:all+eip CAP_NETWORK_MGT-eip:all+eip
fred:all=:all=
```

Each entry consists of up to three colon-separated fields, as follows:

```
username : default_capability : maximum_capability
```

- The username is the user's login name. This must be exactly the same as that found in the */etc/passwd* file.
- The default capability set is applied at login time to the user's shell process. A user may request additional capabilities at login time. If capabilities not present in this entry are requested at login time, the login attempt will fail.
- The maximum capability field describes all those capabilities that may be requested and received by the user's processes.

The default and maximum capability fields are of the following form:

```
capname,capname operator flags
```

The *capname* element(s) are taken from the list of capabilities supplied in the section titled "Capabilities in This Release."

The *operator* can be any one of the following:

+	Add this capability (or list of capabilities) to the following sets.
-	Delete this capability (or list of capabilities) to the following sets.
=	Revoke this capability (or list of capabilities) for the duration of this process for the following sets.

The *flags* that represent the capability sets are one or more of the following:

i	Inheritable set of capabilities. The inheritable set is the capabilities that can be passed to child processes.

e — Effective set of capabilities. The effective set is the capabilities currently active.

p — Permitted set of capabilities. The permitted set is the maximum set of capabilities for the process.

Each field contains a list of clauses. Each clause is a space-separated list of capabilities and an operator/set statement. All characters after # to the end of the entry line are interpreted as comments and are ignored. The clauses are interpreted sequentially, as read (left to right). This means that the last operation specified for a capability within an entry is the one that counts.

Now look at the sample */etc/capability* file again:

```
root:all+eip:all+eip
auditor:CAP_AUDIT_WRITE,CAP_AUDIT_CONTROL,CAP_KILL+eip:
ernie:all=:CAP_FOWNER,CAP_SETFCAP+eip
casey:all=:all+eip # We trust Casey.
jeff:all+eip CAP_NETWORK_MGT-eip:all+eip
fred:all=:all=
```

In this sample file, note the following:

- The **root** account has all capabilities added by default with all flags.
- The auditor account has only those capabilities necessary to manage the system audit trail, and the capability to kill processes.
- The ernie account has no default capabilities, but if necessary can acquire the capabilities to work on other people's files and set capability requirements for executable files.
- The casey account has no default capabilities, but can acquire full capabilities if necessary. There is also a comment to that effect.
- The jeff account has a default set of full capabilities, modified by a subsequent clause to delete the network management capability. However, Jeff can request a full capability set if needed.
- The fred account has no capabilities, nor can Fred request any.

Every running process has three capability sets: *effective*, *permitted*, and *inheritable*.

- The effective set is used in access control decisions for that process.
- The inheritable set is used in the calculation of new capability sets during *exec*(2) processing, when a user invokes an executable file.

- The permitted set is the maximum set of capabilities that the process may attain.

Each executable file has the same three capability sets as well. These sets influence the final effective capability set of the new process created when a user invokes the program:

- The new effective set is the intersection of the permitted set of the parent process and the executable file's effective set. That is, if the executable file's effective set of capabilities includes a capability that is within the permitted set of the calling process, but not within that process' effective set, the capability will be added to the child process' effective set.
- The new inheritable capability set is the intersection of the inheritable capabilities of the calling process and the inheritable capability set of the executable file. That is, only those capabilities that are inheritable by the executable file and are designated inheritable by the parent process will be inheritable in the new process.
- The new permitted capability set is the union of the executable file's permitted set and the intersection of the new inheritable set and the parent process' permitted set. That is, all permitted capabilities of both the file and the parent process are permitted so long as each capability is inheritable by both the parent process and the executable file.

The effective capability set of the parent process does not influence any of the new sets, and the executable file's inheritable set defines an upper bound on the capabilities available to the new process.

Capabilities in This Release

The following capabilities are shipped in this distribution:

ALL — Indicates all capabilities.

CAP_ACCT_MGT
: Privilege to issue accounting setup system calls such as *acct*(2).

CAP_AUDIT_CONTROL
: Privilege to manage the system audit trail such as the *sat_read*(2) and *sat_write*(2) system calls.

CAP_AUDIT_WRITE
: Privilege to write to the system audit trail such as the *sat_write*(2) system call.

CAP_CHOWN Privilege to change the owner of a file not owned by the process and with the system configured for _POSIX_CHOWN_RESTRICTED on changing file ownership.

CAP_CHROOT
Privilege to execute the *chroot*(2) system call.

CAP_DAC_EXECUTE
Privilege to execute a file when the permissions or Access Control List would prohibit it.

CAP_DAC_READ_SEARCH
Privilege to read a file or search a directory even though the permissions or Access Control List would prohibit it.

CAP_DAC_WRITE
Privilege to write a file or update a directory when permissions or Access Control Lists would have prohibited it.

CAP_DEVICE_MGT
Privilege to issue restricted device management calls and ioctl actions such as the following:

- XLV logical volume interface - Defines logical volumes and various parameters about them.
- *syssgi*(SGI_FS_INUMBERS) - Returns all the valid internal handles (inode numbers) on an XFS file system.
- *syssgi*(SGI_FS_BULKSTAT) - Returns file status (struct stat) “in bulk” for an entire file system.
- *fcntl*(F_FSSETDM) - Set the DMA parameters for a file.
- DMI interface - Used by tertiary storage management products.
- Set the CLOCAL flag on a port marked CD_MODEM using ioctl with TCSETA, TCSETAF or TCSETAW control parameters.
- Perform privileged operations on a disk using ioctl.
- Access to the hardware performance monitor using *syssgi*().
- Load, unload, register and unregister loadable device drivers, streams modules, and file systems (*mload*(4)).
- Revoke access to a device using *vhangup*(2).
- Control memory error handling using *syssgi*().

- Establish a user level interrupt handler (*uli*(3)).
- Get and set file system attributes.

CAP_FOWNER
: Privilege to operate on a file as if the process owned it. This capability overrides the requirement that the user ID associated with a process be equal to the file owner ID, except in the cases where the CAP_FSETID capability is applicable. In general, this capability, when effective, will permit a process to perform all the functions that any file owner would have for their files

CAP_FSETID
: Privilege to set the setuid or setgid bits of a file without being the owner. Also, the privilege to change the owner of a file with setuid or setgid bits set. This capability overrides the following restrictions:

  - That the effective user ID of the calling process shall match the file owner when setting the set-user-ID (S_ISUID) and set-group-ID (S_ISGID) bits on that file.
  - That the effective group ID or one of the supplementary group IDs of the calling process shall match the group ID of the file when setting the set-group-ID bit of that file.
  - That the set-user-ID and set-group-ID bits of the file mode shall be cleared upon successful return from chown.

CAP_KILL
: Privilege to send a *kill*(1M) signal to another process not owned by the sender. Also, privilege to use process synchronization calls (procblk) to a process.

CAP_MEMORY_MGT
: Privilege to issue restricted memory management calls, primarily memory locking. This capability overrides the restriction that a process may not manipulate the system memory management policies. The operations enabled by this capability include the following:

  - Lock or unlock a shared memory segment via the *shmctl*() interface.
  - Lock or unlock other segments of a process in memory (*mpin*(2), *plock*(2)).
  - Use of the *syssgi*(SGI_MINRSS) system call.
  - Retrieve the physical address of a page.

CAP_MKNOD
: This is an alias for CAP_DEVICE_MGT.

CAP_MOUNT_MGT
: Privilege to issue the *mount*(2) and *unmount*(2) calls.

CAP_NETWORK_MGT
: Privilege to issue restricted networking calls such as setting the network interface address, and network interface device management. This capability is required to change the system network configuration. The functions enabled by this capability include:

  - Downloading firmware to network device interfaces and starting them.
  - Setting the Media Access Control (MAC) address; for example, the Ethernet address of an interface.
  - Retrieving device management information from network devices.
  - Setting, controlling, and examining the FDDI SMT information.
  - Controlling the ARP mechanism.
  - Controlling the IP address(es), parameters, and flags of network interfaces.
  - Configuring the IP filter.
  - Using the private interface for *lockd*.
  - Using the private interfaces for the NFS service daemons.

CAP_NVRAM_MGT
: This is an alias for CAP_SYSINFO_MGT.

CAP_PRIV_PORT
: Privilege to open a socket on a privileged TCP port.

CAP_PROC_MGT
: Privilege to issue restricted process management calls. This capability is required to override the restrictions on changing the attributes of other processes and to perform privileged process operations. These include the following:

  - Tracing a setuid/setgid executable.
  - Setting resource limits larger than system or per process limits.
  - Use the kernel thread facilities.

- Update the real UID/GID within a share group (without the capability, the effective IDs are updated) when the UID or GID is changed.
- Set the per-process stack size in a share group using *prctl*().
- Force the process to be resident *prctl*.

CAP_QUOTA_MGT
: Privilege to issue restricted disk quota management calls.

CAP_SCHED_MGT
: Privilege to issue restricted scheduler calls such as the real time scheduler interfaces. This capability is required to manipulate the system process scheduler. The operations enabled by this capability include

  - changing the process priority to a high value
  - changing the process priority of another process
  - setting the process to have non-degrading priority
  - setting the real-time priority of a process
  - setting the time slice value for a process
  - controlling the association of processes to processors
  - setting the working set priority for a process
  - using the Frame Rate Scheduling features
  - altering the process resource limits
  - controlling the rate at which the buffer cache flush routine operates

CAP_SETFCAP
: This is an alias for CAP_SETFPRIV.

CAP_SETFPRIV
: Privilege to alter the capability set of a file.

CAP_SETGID
: Privilege to change the real, effective, and saved GID of the process. Also the privilege to change the process group ID.

CAP_SETPCAP
: This is an alias for CAP_SETPPRIV.

CAP_SETPPRIV
: Privilege to alter the capability set for a process.

CAP_SETUID Privilege to change the real, effective, and saved UID of the process.

CAP_SHUTDOWN
Privilege to shut the system down or reboot it. This capability is required to use the *uadmin*(2) system call, which can

- shut the system down
- reboot the system
- force remount of the root after automatic file system damage repair
- notify all processes to terminate gracefully
- power the system down (not supported on all systems)

CAP_STREAMS_MGT
Privilege to issue restricted STREAMS calls and operations.

CAP_SWAP_MGT
Privilege to issue the *swap*(2) call.

CAP_SYSINFO_MGT
Privilege to set system information such as hostname and the NVRAM values. This capability is required to manipulate the system identification information of the system. This includes the following:

- NVRAM contents on adapters such as the FDDI interface (typically addresses or names).
- Host ID, node name and domain name.
- Activate VM fault tracing.
- Control the treatment of UID 0:

  Conventional superuser, UID 0 has all privileges, capabilities are not used.

  Modified superuser, capabilities are used, but **root** doesn't require them. When **root** does an operation that would have needed capabilities, a record is kept.

  No superuser mode, UID 0 and the **root** account are not special.
- Change system tuning parameters.
- Invoke the internal kernel debugging support.
- Set the automatic power on time.

- Set the machine ID (serial number).

CAP_TIME_MGT

Privilege to set the system time. This capability is required to modify the system clock. This includes the following functions:

- Set the time trim adjustment (used for clock synchronization with external sources).
- Adjust the system clock.
- Set the system clock.
- Enable the fast clock.
- Control which processor will handle clock interrupts.

File Capabilities

Capabilities on a file are only meaningful for executable files on XFS format file systems. Capability requirements on files can be set by the Site Security Officer with the *chcap*(1M) command. The syntax is as follows:

chcap *CAP, CAP, CAP file*

For example, suppose you want to set capabilities to match those associated with the Auditor account:

```
auditor:CAP_AUDIT_WRITE,CAP_AUDIT_CONTROL,CAP_KILL+eip
```

Use this command:

chcap CAP_AUDIT_WRITE,CAP_AUDIT_CONTROL,CAP_KILL+eip *file*

To list the capability requirements of a file or directory, use this command:

```
ls -P
```

The **-P** flag stands for "Privilege." Note that you must have the appropriate capabilities to read the file in order to read the capabilities of the file.

Creating Custom Capabilities

You can create unique capabilities at your site. Simply add the capability tag you want (it must be unique) to your */etc/capability* file on the line for the user or users who are to have the capability, then use the *chcap*(1) command to add the capability to the files you desire.

Using attrinit(1) to Clean Up Capability Corruption

If you believe you have experienced corruption of some capability requirements on files or directories , you can use the *attrinit*(1) command to restore those capability requirements.

The */etc/irixcap* file is used with the *attrinit* command as follows:

Log in as **root** and change directories to the root (/) directory. Next, give this command:

```
attrinit -script=/etc/irixcap
```

Your capability integrity will be restored. The process may take a few moments.

Chapter 5

Administering the System Audit Trail

The System Audit Trail is a feature that allows the System Administrators to review a record of all system activity. The ongoing record shows general trends in system usage and also violations of your system use policy. For example, any unsuccessful attempts to use system resources can be recorded in the audit trail. If a user consistently attempts to access files owned by other users or attempts to guess passwords, this can be recorded in the audit trail. The System Administrators can monitor all system activity through the audit trail.

The System Audit Trail is described completely in the guide titled *IRIX Admin: Backup, Security, and Accounting*.

Special Audit Events in the Commercial Security Pak

In addition to those listed in the *IRIX Admin: Backup, Security, and Accounting* guide, the following audit events are available to the Commercial Security Pak:

sat_access_denied
: Access to the file or an element of the path was denied due to enforcement of file permissions.

sat_ae_audit
: Events from the SAT daemon.

sat_ae_lp
: Events from the print daemon.

sat_bsdipc_dac_change
: The ACL or UID of a socket was changed.

sat_bsdipc_dac_denied
: A packet could not be delivered to a socket because of DAC file permissions.

sat_bsd_if_setuid
: A change was made to an outgoing socket UID.

sat_proc_acct
: Extended process accounting information events.

sat_proc_own_ext_attr_write
: The calling process's Capability state was changed.

sat_session_acct
: Extended session accounting information

sat_sys_note
: System internal status data.

Auditing Unexpected Use of Privilege

The Commercial Security Pak has privileges implemented that are not present in standard IRIX. Because of this, the unexpected use of privilege is of special concern to the Auditor of a system with this optional software. Every interpreted audit record contains a line beginning with the keyword "Outcome." The field following this keyword can be equal to one of "Success," "Failure," or "Success due to privilege." The last case indicates that the user made a system call that would have failed except that Superuser privilege was invoked to assure its successful completion. This is not necessarily a security violation or an unexpected use of system privilege. It is perfectly normal to see these outcomes.

When an ordinary user runs a program that contains code that uses system privilege, "success due to privilege" outcomes are generated. A good example of this kind of program is *passwd*(1M). An ordinary user generates a record of this type simply by changing the password on his or her account. Records of this type are also generated when users use capabilities to edit system files.

One type of record to look for is an instance where the "SAT ID" or "Effective ID" field is different from the "User ID" field. This occurs when a user executes */bin/su* to gain SSO privileges or otherwise promotes the privilege level of a session. In most cases, this is not a security violation, since the privilege is necessary to successfully complete the */bin/su* command.

An instance of using Superuser privilege, though, is always worth examination in the audit trail. When you encounter an instance where a user has promoted his or her login session, you should check to see that the user is authorized to have the capability. If not, check whether the user indeed executed the */bin/su* command, or if he or she promoted the privilege of the session by some other means, such as a trojan horse *setuid* shell command.

Whenever a user promotes the privilege of his or her login session, the Auditor should also make a routine check of what actions the user took while the privilege was promoted.

Chapter 6

Administering Identification and Authentication

In order to ensure that the users on your system are the same people who have been entrusted to use it properly, Identification and Authentication improvements have been implemented. Some of these improvements are described in this chapter. The Kerberos authentication system has also been included in this distribution, and represents a significant improvement over conventional identification and authentication tools. Kerberos is described in Chapter 7, "Administering CSP-Kerberos."

In the unlikely event that an individual user breaks a security policy, that user must be held strictly accountable for his or her actions. Holding the owner of a user account responsible for the actions taken with that account is reasonable only if steps have been taken to assure that the individual using that account is in fact the individual assigned to the account. Trusted systems usually implement certain facilities to assure that users are adequately identified and that they authenticate themselves to the system with a password. To log in, the user must know

- a valid login name for the system
- the password associated with that login name

Because these items are all that is needed to gain access to your system, these pieces of information are created, closely guarded and maintained according to strict procedures outlined in this chapter. Of the two items of information, the most crucial is the account password. The login names are known to many people, or can easily be determined. It is possible to log in without specifying a label if the default label has been set, but the password is absolutely necessary. If a password is compromised or stolen, all information that is available to the user associated with the password is also compromised.

Administering Passwords

Passwords are the only mechanism available to authenticate your users. Once IRIX has accepted a user's password as valid, that user has access to all files available at his or her clearance for the duration of the login session.

The dangers involved when passwords are compromised cannot be overstated. An intruder can access all files available to the user at any time. Other features of IRIX make it likely that an intruder would be swiftly identified and locked out, but a tremendous amount of damage can take place in a short time if the accountability and Identification and Authentication procedures are not followed.

Many features taken for granted in standard IRIX are restricted in the Commercial Security Pak. In the area of user passwords, there are facilities to force the user to choose a system-generated password (which is random and difficult to guess) and the length of time that this password is valid can be specified (with both a minimum and maximum lifetime). The encrypted password is not visible to users. When the encrypted password is visible, a potential intruder may copy it and attempt to break the encryption.

If you choose to allow users to select their own passwords, a strict set of checks are performed on the passwords to disallow passwords without enough variation in the characters used. For example, all passwords must use a combination of letters, numerals, and control characters.

Password Aging

Password aging is defined as being able to set a minimum and maximum lifetime for passwords. IRIX supports this feature, and it is described in detail in the guide titled *IRIX Admin: System Configuration and Operation*.

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 interloper has to crack the password. By enforcing a minimum password 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 their password within the specified time period, the user is forced to change the account password when they next try to log in. Any user can place the following line in *.login* or *.profile* to notify them 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. But once locked, an account can be unlocked only by the Superuser or System Administrator. When the account is locked, the encrypted password is replaced by this string:

```
*LK*
```

To unlock the account, the System Administrator uses the *dbedit*(1) utility to remove the string from the password field for that account. Then, the Superuser should force the user to choose a new password by executing this command:

passwd -f *username*

Password aging is enforced for a particular user if his or her encrypted password in the */etc/shadow* file is followed by a comma and a non-null string of characters from the 64-character alphabet:

```
. / 0-9 A-Z a-z
```

The first character of the entry, *Maximum*, denotes the maximum number of weeks for which a password is valid. A user who attempts to log in after his or her password has expired is forced to change the password. The next character, *minimum*, denotes the minimum period in weeks that must pass before the password may be changed. If the second character is omitted, zero weeks is the default minimum. *M* and *m* have numerical values in the range 0-63, which correspond to the 64-character alphabet shown above (for example, / = 1 week; z = 63 weeks). If *minimum* = *Maximum* = 0 (derived from the string . or ..), the user is forced to change the password at the time of the next login (and the "age" disappears from the entry in the *shadow* file). If *minimum* > *Maximum* (signified, for example, by the string ./), only the Superuser can change the password. This is often done for accounts that are used for uucp logins. For example, the command

passwd -x1 -n10 nuucp

disallows the user from changing the password. For complete information on how to put an age limit on a user's password, consult the *passwd*(1) reference page.

Administering Password Generation

There are several options available to the security officer when deciding on password generation policy. IRIX comes equipped with a default password generation utility, */sbin/passwd_gen*, but also allows the individual site to install a custom password generating program. If you have a custom password generator you wish to use, you may replace */sbin/passwd_gen* with your own program, subject to the following constraints:

- The Site Security Officer is willing to accept the risk of using an unevaluated configuration.
- The new program must be placed in the */sbin* directory and renamed *passwd_gen*.
- The owner of the file must be **root**.
- The group of the file must be **sys**.
- The DAC permission of the file must be 555 (-r-xr-xr-x).
- Your system is no longer running the evaluated software configuration.

Additionally, any custom password generation program must generate a set of passwords on one line, separated by blank spaces.

To allow password selection, simply log in as the Superuser and rename the */sbin/passwd_gen* file to a different name.

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 to obtain a new set of passwords.

If you do not wish to implement password generation at your site, you may remove or rename the *passwd_gen* program. Once this is done, users are free to select their own passwords, subject to the following triviality checks:

- 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 numeric 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 are 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 different from the previous password. For the purposes of this test, capital letters are assumed to be equivalent to their lower case counterparts.

Note that if password generation is in effect, the generated passwords are not subject to these triviality checks since more stringent checks are applied internally.

Password Generator Algorithm

The IRIX password generator produces passwords of seven lowercase alphabet characters in length. The IRIX password generator attempts to produce pronounceable passwords, so it produces far fewer than the maximum possible number of passwords. The total password space (the total number of passwords that the generator can possibly produce) is 35,431,196 different passwords. This number is computed based on the size and number of phonemes available for combination into words and the method by which they're combined.

IRIX limits the total password guessing rate (for all accounts, on all tty and pty ports) to no more than one password per second. This guess rate is not user-configurable. Thus, a person who knew what parameters were used by the password generation program, who had unrestricted access to a IRIX system, and who was capable of attempting to log in once per second would be able to guess any password generated by that algorithm in, at most, 35,431,196 seconds, which is 410 days of uninterrupted guessing. In 41 days of uninterrupted guessing, the person would have a 10% chance of guessing any password. In 35 seconds, the person would have a "one in a million" chance of guessing a correct password for a given account.

Of course, it is extremely unlikely that someone attempting to break into an IRIX system would know the parameters used to generate passwords, or have unrestricted access to a well-maintained trusted system, so the rate of guessing would necessarily be much lower. If password aging is implemented, requiring users to change their passwords monthly, the chance of a potential intruder correctly guessing a password is negligible.

The password generator relies on a "pseudo-random" number generator, which is described in the *random*(3) reference page.

Password Issues

The password step can be eliminated from the login process if the user has no password set and the string

```
passwdreq=0
```

appears in the *login.options* file. This means that a user who does not have an initial password set does not have to create one or enter any password to log in. Obviously, this is a highly insecure practice, and you should not allow it on your system.

It is recommended that you have *passwdreq* set to 2 on your system at all times. The effect of setting *passwdreq* to 2 is described below. However, even if passwords are not required on a particular system, any user who maintains a password on his or her account must enter it at login time. Regardless of whether passwords are required, the system does not allow access to a passworded account without receiving the correct password.

If the string

```
passwdreq=1
```

appears in the *login.options* file, passwords are always required on the system and a user without a password is prompted to choose one immediately. This is the default behavior for the Commercial Security Pak.

If the *passwdreq* line reads

```
passwdreq = 2
```

then a user without a password already set is not allowed access and **sso** must create a password for the user before he or she can log in.

Assuming that the user enters the correct password, no other user input is required to complete the login process. If the password or any previous entry was incorrectly entered, the system responds with

```
Login incorrect. Try again.
```

and the login process is aborted.

If the account is new and has no password and passwords are required, the user sees this prompt:

```
Enter New Password:
```

At this time, the user is forced to enter a password before being allowed to complete the login process. The user is always prompted to re-enter the new password as an error check.

Login Failures

During the login process, login failures are counted and compared against the values set in the *login.options* file. The line

```
maxtries = number
```

indicates the number of unsuccessful attempts allowed per login port. The default value for this keyword is 5. If the user unsuccessfully attempts to log in 5 consecutive times on the same terminal, the system disallows logins on that terminal for the number of seconds specified in the *login.options* file by this entry:

```
disabletime = number
```

The default value for *disabletime* is 20 seconds. The **sso** account is exempt from this restriction, since it may be necessary for **sso** to log in quickly in an emergency.

If the keyword *syslog* is in the *login.options* file with either of the following settings, unsuccessful login attempts are placed in the system log with the date and time:

```
syslog = all syslog = fail
```

The login.options File

The *login.options* file allows you to set the options for all users on the system as shown in Table 6-1.

Table 6-1 Login Options

Option	Default Value	Meaning
maxtries	5	Maximum consecutive number of unsuccessful login attempts to any login name through the same port. A 0 in this space indicates “no limit.”
disabletime	20	The amount of time in seconds *login* waits before ending the session after *maxtries* unsuccessful attempts.
passwdreq	0	This field indicates whether passwords are required. If this field contains a 0, passwords are not required. If the field contains a 1, you may select a password when you log in if you do not have one. If the field contains a 2, you may not log in without a previously set password.
lastlog	1	This field indicates whether the user is to be notified about the last successful login attempt. A 1 in this field indicates that the user should be notified. If a 0 is present in this field, notification is suppressed.
syslog	all	This field directs the system to log all successful and failed login attempts to the system log. If the value in the field is *fail*, then only failed attempts are logged.

After your installation is complete, you may edit the *login.options* file to enforce your particular system security policies. Before you edit the file, be sure to make a backup copy of the original. If the file is removed, the default values in effect at installation time are used.

The /etc/shadow File

When the user logs in, the system encrypts the password and tests it against the encrypted password for the account listed in the */etc/shadow* file. The */etc/passwd* file is still in existence, though, and still contains all pertinent user information except the encrypted password. The *passwd* file also contains information about the minimum and maximum age of that user's password.

Chapter 7

Administering CSP-Kerberos

CSP-Kerberos™ is a set of programs and libraries that provide distributed user identification and authentication over an open network. Servers using CSP-Kerberos can be certain of the identities of clients making requests. Likewise, clients using CSP-Kerberos can be sure their requests are being serviced by authentic servers. Applications that use CSP-Kerberos distributed authentication are called *kerberized* applications.

This chapter contains the following major sections:

- "Introduction to CSP-Kerberos"

  This section briefly explains the nature of CSP-Kerberos and the software packages in this distribution.

- "Reasons To Use CSP-Kerberos"

  This section outlines some of the user interactions with CSP-Kerberos and provides more detail on how CSP-Kerberos performs its work.

- "Planning For CSP-Kerberos"

  This section details the planning you should do before you install and configure your CSP-Kerberos software.

- "Installing CSP-Kerberos"

  This section provides installation instructions for all classes of systems running the CSP-Kerberos software.

- "Configuring CSP-Kerberos"

  This section provides post-installation configuration advice, as well as some information about unusual configuration scenarios.

- "CSP-Kerberos Administrative Commands"

  This section details the administrative commands used to maintain and configure CSP-Kerberos.

- "CSP-Kerberos Libraries"

This section briefly lists the programming libraries distributed for CSP-Kerberos custom development.

- "Troubleshooting CSP-Kerberos"

  This section provides troubleshooting advice for CSP-Kerberos.

Introduction to CSP-Kerberos

This implementation of CSP-Kerberos is based on the Kerberos authentication system developed at the Massachusetts Institute of Technology. Under CSP-Kerberos, a client (generally either a user or a service) uses the *kinit*(1) utility to request services from the Key Distribution Center (KDC). The KDC creates a ticket-granting ticket (TGT) for the client, encrypts it using the client's 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 (that is, if the user 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 transparent to the user.

As with any software package that uses a centralized database, the installation procedure is somewhat involved, and requires forethought and planning. We have attempted to make this chapter as concise as possible, rather than making it an exhaustive description of the details of CSP-Kerberos.

It is assumed in this guide that readers are familiar with the user-level CSP-Kerberos mechanism of distributed authentication, as described in the *Commercial Security Pak User's Guide*.

The Silicon Graphics implementation of CSP-Kerberos software is based on Kerberos version 5 (Beta 6) from MIT, and can be categorized into the following general classes of files and programs:

- CSP-Kerberos KDC software

  CSP-Kerberos KDC software is provided in the *csp-kerberos.sw.kdc* package in your software distribution.

CSP-Kerberos KDC software executes on the host that provides and manages the secure CSP-Kerberos principal database. This package includes software commonly called the ticket-granting server (TGS), the key or CSP-Kerberos distribution center (KDC), the CSP-Kerberos Name Server (KNS), and the CSP-Kerberos Database Manager (KDBM).

This software also includes the utility commands (for example, *kdb5_create*, *kdb5_destroy*, *kadmin5* and *kdb5_edit*) that are used to manage the CSP-Kerberos principal database.

- CSP-Kerberos server software

  CSP-Kerberos server software is provided in the *csp-kerberos.sw.server* package in your distribution.

  CSP-Kerberos server software includes the daemon programs, such as *klogind*(8), *kshd*(8), *ftpd*(8), and *telnetd*(8), that provide CSP-Kerberos authenticated services to requesting clients.

- CSP-Kerberos client software

  CSP-Kerberos client-side software is distributed in the *csp-kerberos.sw.client* package in your software distribution,

  Kerberized client applications are client programs that obtain CSP-Kerberos authenticated services from kerberized servers. Examples of client programs are kerberized *rlogin*(1), *rsh*(1), *rcp*(1), *ksu*(1), *telnet*(1B), and *ftp*(1B).

  CSP-Kerberos utility commands run on hosts from which users establish CSP-Kerberos authenticated sessions. These commands manage user tickets and allow users to change their CSP-Kerberos passwords. Examples of utilities are *kinit*(1), *kpasswd*(1), *klist*(1), and *kdestroy*(1).

- CSP-Kerberos development libraries

  These programming libraries are distributed in the *csp-kerberos.sw.dev* package in your software distribution,

  CSP-Kerberos libraries are used to create and run kerberized applications. The library set includes libraries that provide assistance in obtaining and using tickets, encrypting and decrypting CSP-Kerberos data, managing CSP-Kerberos access control lists, and communicating with the CSP-Kerberos administration server.

  Note: Support for data stream encryption in kerberized applications is not available outside of the United States and Canada.

Reasons To Use CSP-Kerberos

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, you should not neglect the maintenance of security measures within your local area network or intranet.

User and CSP-Kerberos Interaction

Suppose that you walk up to a host intending to log in to it, and then *rlogin* to the system *laughter*. Here's what happens:

1. You log in to the workstation and use the *kinit* command to get a ticket-granting ticket. This command prompts you for your CSP-Kerberos password.

   The *kinit* command sends your request to the CSP-Kerberos master server system. The server software looks for your principal name's entry in the CSP-Kerberos database.

   If this entry exists, the CSP-Kerberos server creates and returns a ticket-granting ticket and the key that allows you to use it, encrypted by your password. If *kinit* can decrypt the CSP-Kerberos reply using the password you provide, it stores this ticket in a credentials cache on your local system for later use. The name of the credentials cache can be specified in the KRB5_CCNAME environment variable. If this variable is not set, the name of the file will be */tmp/krb5cc_<uid>*, where *<uid>* is your IRIX user ID number.

2. Now you use the *rlogin* client to access the system *laughter*, as shown in the following command:

   ```
   rlogin laughter
   ```

   The *rlogin* client checks your ticket file to see if you have a ticket for the host service for *laughter*. You don't, so *rlogin* uses the credential cache's ticket-granting ticket to make a request to the master server's ticket-granting service.

   This ticket-granting service receives the request for a ticket for host/laughter.yoursite.com, and looks in the master database for an entry for host/laughter.yoursite.com. If the entry exists, the ticket-granting service issues you a ticket for that service. That ticket is also cached in your credentials cache.

The *rlogin* client now sends that ticket to the *laughter klogind* service program. The service program checks the ticket by using its own service key. If the ticket is valid, it now knows your identity. If you are allowed to log in to *laughter* (because your username matches one in */etc/passwd*, or your CSP-Kerberos principal is in the appropriate *.k5login* file), *klogind* will let you log in.

How CSP-Kerberos Works

This section provides a simplified description of a general user's interaction with the CSP-Kerberos system. This interaction happens transparently, but CSP-Kerberos administrators might find a schematic description of the process useful. This description glosses over a lot of details; for more information, see "Kerberos: An Authentication Service for Open Network Systems," a paper presented at Winter USENIX 1988, in Dallas, Texas. This paper can be retrieved by FTP from *athena-dist.mit.edu*, in the location */pub/ATHENA/kerberos/doc/USENIX.ps*.

In an environment that provides network services, you use client programs to request services from server programs that are somewhere on the network. Suppose you have logged in to a workstation and you want to *rlogin* to a typical host. You use the local *rlogin* client program to contact the remote system's *rlogind* daemon.

Under CSP-Kerberos, the *klogind* daemon allows you to log in to a remote system if you can provide a CSP-Kerberos ticket that proves your identity. In addition to the ticket, you must also have possession of the corresponding ticket session key. The combination of a ticket and the ticket's session key is known as a credential.

Typically, a client program automatically obtains credentials identifying the person using the client program. The credentials are obtained from a CSP-Kerberos server that resides somewhere on the network. A CSP-Kerberos server maintains a database of user, server, and password information.

CSP-Kerberos gives you credentials only if you have an entry in the CSP-Kerberos server's CSP-Kerberos database. Your database entry includes your CSP-Kerberos principal (an identifying string, which is often just your username), and your CSP-Kerberos password. Every CSP-Kerberos user must have an entry in this database.

Each administrative domain has its own CSP-Kerberos database, which contains information about the users and services for that particular site or administrative domain. This administrative domain is the CSP-Kerberos realm.

Each CSP-Kerberos realm has at least one CSP-Kerberos server, where the master CSP-Kerberos database for that site or administrative domain is stored. A CSP-Kerberos realm may also have one or more slave servers, which have read-only copies of the CSP-Kerberos database that are periodically propagated from the master server. For more details on how this is done, see the “Installing CSP-Kerberos KDCs” section of this chapter.

The *kinit* command prompts for your password. If you enter it successfully, you obtain a ticket-granting ticket and a ticket session key, which gives you the right to use the ticket. This combination of the ticket and its associated key is known as your credentials. As illustrated below, client programs use your ticket-granting ticket credentials in order to obtain client-specific credentials as needed.

Your credentials are stored in a credentials cache, which is often just a file in */tmp*. The credentials cache is also called the ticket file. Note, however, that a credentials cache does not have to be stored in a file.

The master database also contains entries for all network services that require CSP-Kerberos authentication. Suppose that your site has a system, laughter@yoursite.com, that requires CSP-Kerberos authentication from anyone who wants to log in to it remotely. The host's CSP-Kerberos realm is ENGR.YOURSITE.COM.

This service must be registered in the CSP-Kerberos database, using the proper service name, which in this case is the principal:

```
host/laughter.yoursite.com@ENGR.YOURSITE.COM
```

The / character separates the CSP-Kerberos primary (in this case, host) from the instance (in this case, laughter.yoursite.com; the @ character separates the realm name (in this case, ENGR.YOURSITE.COM) from the rest of the principal. The primary, host, denotes the name or type of the service that is being offered: generic host-level access to the system. The instance, laughter.yoursite.com, names the specific system that is offering this service. There generally are many different systems, each offering one particular type of service, and the instance serves to give each one of these servers a different CSP-Kerberos principal.

Planning For CSP-Kerberos

Like most security precautions, some advance planning is necessary to make a successful CSP-Kerberos installation.

Because CSP-Kerberos is a complex system, customizable to your individual site needs, it is strongly recommended that you plan your CSP-Kerberos installation as outlined in the following subsections before attempting to install CSP-Kerberos on your network.

Planning CSP-Kerberos KDCs

You must have at least one KDC in your CSP-Kerberos realm. Regardless of the number of KDCs you choose to have, there must be one master KDC.

Slave KDCs provide an additional source of CSP-Kerberos ticket-granting services in the event of inaccessibility of the master KDC. The number of slave KDCs you need and the decision of where to place them, both physically and logically, depend on the specifics of your network.

All of the CSP-Kerberos authentication on your network requires that each client be able to contact a KDC. Therefore, you need to anticipate any likely reason a KDC might be unavailable and have a slave KDC to take up the slack.

Some considerations include the following:

- Have at least one slave KDC as a backup for when the master KDC is down, is being upgraded, or is otherwise unavailable.
- If your network is split such that a network outage is likely to cause some segment or segments of the network to become cut off or isolated, have a slave KDC accessible to each segment.
- If possible, have at least one slave KDC in a different building from the master, in case of power outages, fires, or other localized disasters.

Hostnames for the Master and Slave KDCs

Silicon Graphics recommends that your KDCs have a predefined set of hostnames, such as KDCSERVER for the master KDC and KDCSLAVE1, KDCSLAVE2, and so on for the slave KDCs. This way, if you need to swap a system, you need to change only a DNS entry, rather than the hostnames.

CSP-Kerberos Database Propagation

The CSP-Kerberos database resides on the master KDC, and must be propagated regularly (usually by a *cron* job) to the slave KDCs. In deciding how frequently the propagation should happen, you will need to balance the amount of time the propagation takes against the maximum reasonable amount of time a user should have to wait for a password change to take effect. Silicon Graphics recommends that this be no longer than an hour.

If the propagation time is longer than this maximum reasonable time (for example, you have a particularly large database, you have a lot of slaves, and/or you experience frequent network delays), you may wish to cut down on your propagation delay by performing the propagation in parallel. To do this, have the master KDC propagate the database to one set of slaves, and then have each of these slaves propagate the database to additional slaves.

Planning CSP-Kerberos Servers

On a CSP-Kerberos server, you must install the entire CSP-Kerberos software distribution. This includes support for kerberized servers such as *ftpd*, *klogind*, *telnetd*, and *rshd*.

If you need to install the CSP-Kerberos programs on a redundant server, you need to install CSP-Kerberos on that server and add that host to the CSP-Kerberos database. You also need to make sure the host's clock is within your maximum clock skew of the KDC. You must also generate a keytab (*v5srvtab*) file (using *kadmin5*) for that redundant server and install the keytab file on the redundant server. See "Installing CSP-Kerberos Servers" for more information.

Planning Redundant CSP-Kerberos Servers

Because the CSP-Kerberos server is the sole repository of information concerning the identities of network principals, you may want to consider installing one or more backup CSP-Kerberos servers. Backup servers allow continued use of the CSP-Kerberos authentication system if the primary CSP-Kerberos server becomes unavailable for any reason. This distribution of CSP-Kerberos includes software to propagate the CSP-Kerberos database from the primary CSP-Kerberos server to any CSP-Kerberos backup servers. This software exists as a client/server pair, *kprop* and *kpropd*.

Besides imparting the benefits of redundancy in case the primary CSP-Kerberos server is unavailable, backup servers let you configure clients and servers to query different CSP-Kerberos servers when users need to obtain tickets. Thus, additional CSP-Kerberos servers can potentially improve performance. In practice, however, the availability of multiple servers is not likely to improve performance except in very large networks. CSP-Kerberos is designed so that tickets are reusable; therefore, applications do not need to query CSP-Kerberos except when they need a new ticket. The average user might need to query CSP-Kerberos several times per day, but unless there are many users on the network, this load should not be difficult for one CSP-Kerberos server to handle.

However, if you are thinking of using backup servers, consider the trade-offs of installing redundant CSP-Kerberos servers by weighing the cost of running without CSP-Kerberos and the likelihood of primary CSP-Kerberos server failure against the extra expense and administration required to install and maintain multiple servers. Also, consider that having multiple servers means having multiple points of potential attack; all backup servers must be installed in a manner just as secure as the primary server. This includes both the physical installation, which should restrict access to the system, as well as the software installation, which must protect the security of the principal database.

Planning CSP-Kerberos Clients

On a CSP-Kerberos client, you need only install the *kerberos.sw.client* package of the CSP-Kerberos distribution, then make principal entries for each user on the client system on the KDC. This choice for a host provides support for authenticated outgoing CSP-kerberos sessions only. This allows users to obtain CSP-Kerberos tickets based on your system; these tickets are good for other hosts offering CSP-Kerberos authenticated services.

Planning CSP-Kerberos Realms

Although your CSP-Kerberos realm can be any ASCII string, convention is to make it the same as your domain name, in upper case letters. For example, hosts in the domain theirsite.com would be in the CSP-Kerberos realm THEIRSITE.COM.

If you need multiple CSP-Kerberos realms, Silicon Graphics recommends that you use descriptive names that end with your domain name, such as BOSTON.THEIRSITE.COM and SAN_FRANCISCO.THEIRSITE.COM.

Mapping Hostnames Onto CSP-Kerberos Realms

Mapping hostnames onto CSP-Kerberos realms is done through a set of rules in the *krb5.conf* configuration file. You can specify mappings for an entire domain or subdomain, and/or on a hostname-by-hostname basis. Since greater specificity takes precedence, you would do this by specifying the mappings for a given domain or subdomain and listing the exceptions. Here is an example *krb5.conf* file:

```
[libdefaults]
   default_realm = YOURSITE.COM

[realms]
   YOURSITE.COM   = {
      kdc = BIGKDC.YOURSITE.COM
      admin_server = SERVER.YOURSITE.COM
      default_domain = YOURSITE.COM
   }

[domain_realm]
   yoursite.com = YOURSITE.COM
   .yoursite.com = YOURSITE.COM

[login]
krb5_get_tickets = 1
krb4_get_tickets = 1
krb4_convert = 0

[logging]
   kdc = FILE:/var/log/krb5kdc.log
   admin_server = FILE:/var/log/kadmin.log
   default = FILE:/var/log/krb5lib.log
```

Planning New CSP-Kerberos Applications

You may want to allow users to develop their own authenticated clients and servers. This requires that they be allowed to link their programs with CSP-Kerberos libraries. It also requires cooperation with the CSP-Kerberos administrator, who must edit the primary CSP-Kerberos database of network principals to inform CSP-Kerberos about the new servers. See “CSP-Kerberos Libraries” for more information.

Planning For Synchronized Clocks

Because CSP-Kerberos depends on the use of encrypted time stamps that allow it to detect replay attacks (attempts to place packets back on the network) and expired tickets (tickets that are no longer current), it is critical that all relevant hosts (clients, servers, and CSP-Kerberos servers) keep their clocks synchronized within a few seconds of each other at all times.

If the relevant hosts on your network do not have synchronized clocks, it is very likely that CSP-Kerberos authentication will fail with “Time Out Of Bounds” or “Ticket Expired” errors.

With this in mind, you should initiate procedures that will keep the system clocks synchronized on the relevant hosts before trying to install and operate CSP-Kerberos. Most software that implements distributed time protocols needs some time to stabilize. This time is not merely to allow the administrator to perform the installation and configuration correctly; most distributed time software requires time to determine the characteristics of each local clock it is expected to synchronize. Sometimes this determination takes several days.

Your CSP-Kerberos installation will proceed more smoothly if you have a solid, trustworthy, distributed time mechanism already in place. For more information on keeping system clocks synchronized, see the section titled “Clock Skew Commands” in this chapter.

Planning Your CSP-Kerberos Administrators

Because CSP-Kerberos requires separate administration, the chief administrator may need to choose one or more trusted people to serve as CSP-Kerberos administrators. These people create and modify the CSP-Kerberos principal database residing on the CSP-Kerberos primary server, and they also monitor the log files written by CSP-Kerberos.

Planning for Third-Party Software

Your network probably consists of various types of hardware, running many different versions of software. Check with the vendors of your other hardware and software to determine whether they have implementations of this software available on their

platforms. If they do not, and if you want to use CSP-Kerberos on the unsupported platform, you must port MIT's Kerberos version 5 to that platform yourself.

Planning for Trusted Hosts

CSP-Kerberos can protect your host from certain types of break-ins, but it is possible to install CSP-Kerberos and still leave your host vulnerable to attack. This guide does not attempt to include an exhaustive list of countermeasures for every possible attack, but it is worth noting some of the larger holes and how to close them.

Silicon Graphics recommends that on a secure host, you disable the standard *ftp*, *login*, *telnet*, *shell*, and *exec* services in */etc/services*. We also recommend that secure hosts have an empty */etc/hosts.equiv* file and that there not be a *.rhosts* file in **root**'s home directory. You can grant CSP-Kerberos authenticated **root** access to specific CSP-Kerberos principals by placing those principals in the file *.k5login* in **root**'s home directory.

Installing CSP-Kerberos

This section gives installation procedures for each class of CSP-Kerberos systems:

- "Installing CSP-Kerberos KDCs"
- "Installing CSP-Kerberos Servers"
- "Installing CSP-Kerberos Clients"

Note: A system can be any or all of a client and a server and a KDC.

Before installing CSP-Kerberos, it is necessary to consider the following issues:

- The name of your CSP-Kerberos realm (or the name of each realm, if you need more than one). You can select any realm name you choose, but it is advisable to have this name track your IP domain names for ease of use. See "Planning CSP-Kerberos Realms" for more information.
- How you will map your hostnames onto CSP-Kerberos realms. Again, it is advisable to have this mapping match your IP domain names. See "Mapping Hostnames Onto CSP-Kerberos Realms" for more information.

- Which ports your KDC and *kadmin5* (database access) services will use. The defaults are strongly suggested for compatibility. See “Ports for the KDC and Admin Services” for more information.
- How many slave KDCs you need and where they should be located within your network topology. See “Planning CSP-Kerberos KDCs” for more information.
- The hostnames of your master and slave KDCs. See “Hostnames for the Master and Slave KDCs” for more information.
- How frequently you will propagate the database from the master KDC to the slave KDCs. See “CSP-Kerberos Database Propagation” for more information.

Installing CSP-Kerberos KDCs

The Key Distribution Centers (KDCs) issue CSP-Kerberos tickets. Each KDC contains a copy of the CSP-Kerberos database. The master KDC contains the master copy of the database, which it propagates to the slave KDCs at regular intervals. All database changes (such as password changes) are made on the master KDC.

Slave KDCs provide CSP-Kerberos ticket-granting services, but not database access. This allows clients to continue to obtain tickets when the master KDC is unavailable.

Silicon Graphics recommends that you install all of your KDCs to be able to function as either the master or one of the slaves. This will enable you to easily switch your master KDC with one of the slaves if necessary. This installation procedure is based on that recommendation. Note that this implies that security considerations are the same for all KDCs. Every slave KDC must be protected as thoroughly as your master KDC, since the same realm-wide information is present on each KDC.

Installing the Master KDC

This installation procedure will require you to go back and forth a couple of times between the master KDC and each of the slave KDCs. The first few steps must be done on the master KDC.

1. Edit the configuration files on the master KDC.

   Modify the configuration files, */etc/krb5.conf* and */krb5/lib/krb5kdc/kdc.conf* to reflect the correct information (such as the hostnames and realm name) for your realm. Silicon Graphics recommends that you keep *krb5.conf* in */etc*. The *krb5.conf* file may contain a pointer to *kdc.conf* You must change that pointer if you want to move *kdc.conf* to another location.

2. Create the database on the master KDC.

   Use the *kdb5_create* and *kdb5_stash* commands on the Master KDC to create the CSP-Kerberos database and the optional stash file. The stash file is a local copy of the master key that resides in encrypted form on the KDC's local disk. The stash file is used to authenticate the KDC to itself automatically before starting the *kadmind5* and *krb5kdc* daemons as part of the system's boot sequence. The stash file, like the keytab file, is a potential point of entry for a break-in, and if compromised, would allow unrestricted access to the CSP-Kerberos database. If you choose to install a stash file, it should be readable only by **root**, and should exist only on the KDC's local disk. The file should not be part of any backup of the system, unless access to the backup data is secured as tightly as access to the master password itself.

   If you elect not to create a stash file, you must start *krb5kdc* and *kadmind5* individually from a command line using the **-m** option, and you must provide the appropriate CSP-Kerberos master key for your KDC at start time. With a stash file, this process can be automated, as shown in the examples below.

   The decision to use a stash file or not comes down to convenience vs. better security.

   Note: The *kdb5_create* program prompts you for the master key for the CSP-Kerberos database. This key can be any string. A good key is one that you can remember, but no one else can guess. Examples of bad keys are words that can be found in a dictionary, any common or popular name, especially a famous person (or cartoon character), your user name in any form (for example, forward, backward, repeated twice, and so on), and any of the sample keys that appear in this manual. One example of a key which would be good if it did not appear in this manual is “SGIiys4CK5!”, which represents the sentence “Silicon Graphics Inc. is your source for CSP-Kerberos 5!” (It's the first letter of each word, substituting the numeral “4” for the word “for”, and includes the punctuation mark at the end.)

   The following is an example of how to create a CSP-Kerberos database and stash file on the master KDC, using the *kdb5_create* and *kdb5_stash* commands. Replace YOURSITE.COM with the name of your CSP-Kerberos realm.

```
/krb5/sbin/kdb5_create
Initializing database '/krb5/lib/krb5kdc/principal' for realm
'YOURSITE.COM',
master key name 'K/M@YOURSITE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:  <Type the master password>
Re-enter KDC database master key to verify: <Type it again>
```

This creates three files in the directory specified in your *kdc.conf* file (usually */krb5/lib/krb5kdc*, where the *kdc.conf* file itself resides): *principal.dir*, *principal.pag*, and *principal.ok*.

Next, create the stash file, *.k5<Realm_name>*. (The default directory is */krb5/lib/krb5kdc* and *<Realm_name>* is replaced with the actual name of the CSP-Kerberos realm that the stash file represents.) The following example shows the usage:

```
/krb5/sbin/kdb5_stash
Enter KDC database master key: <Type the same key again>
```

3. Add users to your database

   Add users to your CSP-Kerberos principals database using the *kadmin5* utility. A sample usage is shown below:

   ```
   kadmin5:  ank zursch
   Enter password: <Enter a password for this user>
   Re-enter password for verification: <Enter it again>
   kadmin5:  q
   ```

   The command *ank* within the utility stands for Add New Key. Users entered in the database may change their passwords (and should be encouraged to do so) with the */krb5/bin/kpasswd* command.

4. Add a principal entry to your CSP-Kerberos principals database for the *kadmind5* daemon. A sample usage is shown below:

   ```
   kadmin5:  ank kadmind5
   Enter password: <Enter a password for this utility>
   Re-enter password for verification: <Enter it again>
   kadmin5:  q
   ```

5. Start the CSP-Kerberos Daemons on the Master KDC

   At this point, you are ready to start the CSP-Kerberos daemons on the Master KDC. To do so, type the commands:

```
/krb5/sbin/krb5kdc
/krb5/sbin/kadmind5
```

Each daemon will fork and run in the background. Assuming you want these daemons to start up automatically at boot time, you can add them to the KDC's */etc/rc* or */etc/inittab* file. You need to have a stash file in order to do this.

Installing the Slave KDCs

You are now ready to start configuring the slave KDCs. Slave CSP-Kerberos KDCs allow clients to be able to get CSP-Kerberos tickets even when the master KDC is not available. Users will not be able to change their passwords — changes can only be made to the database on the Master KDC; however, users will be able to authenticate to servers, which is critically important in a distributed client-server environment. The current implementation of the client code does not provide load sharing in that the order of servers contacted is the same as those listed in the *krb5.conf* file.

In order to propagate the CSP-Kerberos database from the master KDC to the slaves, the *kprop* and *kpropd* client/server programs are used. Periodically, you should dump the CSP-Kerberos database out into an ASCII format, using the *kdb5_edit* program. Then run *kprop* to propagate the dumped database file to each slave server. This can be done conveniently with a *cron*(1) job.

On the slave KDC, the *kpropd* program is invoked through the *inetd* server. After *kprop* and *kpropd* have mutually authenticated with one another, and *kpropd* is satisfied with the identity of the master KDC, then the dumped ASCII database is transferred to the slave KDC in an encrypted fashion. After the database is transferred, *kpropd* will then run *kdb5_edit* with the appropriate arguments in order to undump the database into a usable form by the KDC slave.

To limit the possibility that your CSP-Kerberos database could be compromised, Silicon Graphics recommends that each KDC be a dedicated host, with limited access. If your KDC is also a server, or even just a client system, someone who obtained **root** access through a security hole in any of those areas could gain access to the CSP-Kerberos database.

Follow these steps to successfully propagate your Master KDC to the slave KDCs:

1. Each KDC needs a host principal in the master CSP-Kerberos KDC database. For example, if your master KDC is called zork.yoursite.com, and you have two slaves named zork1.yoursite.com and zork2.yoursite.com, you would use the *kadmin5* command as shown to add the following principals to the KDC:

   ```
   kadmin5:ark host/zork.yoursite.com
   kadmin5:ark host/zork1.yoursite.com
   kadmin5:ark host/zork2.yoursite.com
   ```

2. Extract the host *v5srvtab* file for each slave KDC. You must have a keytab on the slave server in order to decrypt tickets during the propagation process. Common sense security rules apply to copying keytab files over the network. You should encrypt the keytab files before any transfer. To extract the *v5srvtab* files, use the following commands:

   ```
   kadmin5:xst host
   kadmin5:xst zork1.yoursite.com host
   kadmin5:xst zork2.yoursite.com host
   ```

3. Next, copy the extracted keytab files to the */etc* directories on their respective hosts:

   ```
   rcp zork1.yoursite.com-new-srvtab zork1:/etc/v5srvtab
   rcp zork2.yoursite.com-new-srvtab zork2:/etc/v5srvtab
   ```

4. You must have an ACL file for *kpropd*. This ACL file must be named */krb5/lib/krb5kdc/kpropd.acl*. The ACL file must contain the host principals for each of the KDCs. And it must exist on both the master KDC and all slave KDCs. Continuing the example case, this ACL file would read as follows:

   ```
   host/zork.yoursite.com@YOURSITE.COM
   host/zork1.yoursite.com@YOURSITE.COM
   host/zork2.yoursite.com@YOURSITE.COM
   ```

5. The *inetd.conf* file on each involved system must contain this entry:

   ```
   krb5_prop stream tcp nowait root /krb5/sbin/kpropd  kpropd
   ```

   and of course you must restart *inetd* with the command:

   ```
   /etc/killall -HUP inetd
   ```

6. The */etc/services* file on each involved system must contain this entry:

   ```
   krb5_prop  754/tcp  # CSP-Kerberos slave propagation
   ```

7. Create the database file to be propagated with *kdb5_edit*:

   ```
   kdb5_edit: ddb /krb5/lib/krb5kdc/slave_datatrans
   kdb5_edit: exit
   ```

8. Propagate the file to each slave KDC. It is usually convenient to write a shell script to perform this task and to register the script as a *cron* job:

```
kprop -f /krb5/lib/krb5kdc/slave_datatrans zork1.yoursite.com
Database propagation to zork1.yoursite.com: SUCCEEDED

kprop -f /krb5/lib/krb5kdc/slave_datatrans zork2.yoursite.com
Database propagation to zork2.yoursite.com: SUCCEEDED
```

9. On each slave KDC, load the new database as shown:

```
kdb5_edit: lddb /krb5/lib/krb5kdc/from_master principal.dir
kdb5_edit: exit
```

10. On each slave KDC, create stash files and then start the *krb5kdc* daemon as shown:

```
/krb5/sbin/kdb5_stash
Enter KDC database master key: <Type the master key>

/krb5/sbin/krb5kdc
```

Installing CSP-Kerberos Servers

A server is a host that provides one or more services over the network. Servers can be "secure" or "insecure." A "secure" host is set up to require authentication from every client connecting to it. An "insecure" host will still provide CSP-Kerberos authentication, but will also allow unauthenticated clients to connect.

If you have CSP-Kerberos installed on all of your client systems, Silicon Graphics recommends that you make your hosts secure, to take advantage of the security that CSP-Kerberos authentication affords. However, if you have some clients that do not have CSP-Kerberos installed, you can run an insecure server, and still take advantage of CSP-Kerberos's authentication capability.

CSP-Kerberos Server Programs

Just as CSP-Kerberos provided its own CSP-Kerberos-enhanced versions of client network programs, CSP-Kerberos also provides CSP-Kerberos-enhanced versions of server network daemons. These are *ftpd*, *klogind*, *kshd*, and *telnetd* These programs are installed in the directory */krb5/sbin*. You may want to add this directory to **root**'s path.

CSP-Kerberos Server Configuration Files

For a secure server, Perform these steps:

1. Make the following changes to */etc/inetd.conf* on your new server:
   - Find and comment out any lines for the services *ftp*, *telnet*, *shell*, and *exec*.
   - Add the following lines to */etc/inetd.conf*:

   ```
   klogin  stream tcp nowait root /krb5/sbin/klogind klogind -k -c
   eklogin stream tcp nowait root /krb5/sbin/klogind klogind -k -c -e
   kshell  stream tcp nowait root /krb5/sbin/kshd kshd -k -c -A
   ftp     stream tcp nowait root /krb5/sbin/ftpd ftpd -a
   telnet  stream tcp nowait root /krb5/sbin/telnetd telnetd -a valid
   ```

   See the various reference pages for the above listed programs for information on their options and syntax.
2. Add a principal entry for your new server host in the KDC.
3. If you have any custom CSP-Kerberized applications being served, a principal entry must be created on the KDC for that application.
4. Extract a keytab (*v5srvtab*) file for your server from the KDC.
5. Place the *v5srvtab* file in the */etc* directory of your server.
6. Give the command:

   `killall -hup inetd`

   on your server to reset *inetd*.

For an insecure server, perform the following steps:

1. Make the following changes instead to */etc/inetd.conf*:
   - Find and comment out any lines for the services *ftp* and *telnet*.
   - Add the following lines to */etc/inetd.conf*:

   ```
   klogin  stream tcp nowait root /krb5/sbin/klogind klogind -k -c
   eklogin stream tcp nowait root /krb5/sbin/klogind klogind -k -c -e
   kshell  stream tcp nowait root /krb5/sbin/kshd kshd -k -c -A
   ftp     stream tcp nowait root /krb5/sbin/ftpd ftpd
   telnet  stream tcp nowait root /krb5/sbin/telnetd telnetd -a none
   ```

   See the various reference pages for the above listed programs for information on their options and syntax.

2. Add a principal entry for your new server in the KDC.
3. Extract a keytab (*v5srvtab*) file for your server from the KDC.
4. Place the *v5srvtab* file in the */etc* directory of your server.
5. Give the command:

   ```
   killall -hup inetd
   ```

   on your server to reset *inetd*.

Installing CSP-Kerberos Clients

Client system installation is much more straightforward than installation of the KDCs.

At the most basic level, you will install the *csp-kerberos.sw.client* software package from your distribution and register your users as principals in the KDC database.

Installing Client Programs

The CSP-Kerberized client programs are *rlogin*, *telnet*, *ftp*, *rcp*, *rsh* and *ksu*. All of these programs are in the directories */krb5/bin* and */krb5/sbin*.

You will probably want to have your users put */krb5/bin* and */krb5/sbin* ahead of */bin* and */usr/bin* in their paths, so they will by default get the CSP-Kerberos versions of *rlogin*, *telnet*, *ftp*, *rcp*, and *rsh*. If a user invokes the kerberized version of a program and attempts to access a system not using CSP-Kerberos, the kerberized version of the command will time out and then automatically use the non-kerberized version of the same program to access the remote host.

You will also need to educate your users to use the ticket management programs *kinit*, *krb524init*, *klist*, *kpasswd* and *kdestroy*, and to use the CSP-Kerberos programs *pfrom*, *ksu*, and *kpasswd* in place of their non-CSP-Kerberos counterparts *from, su* and *passwd*.

CSP-Kerberos Client Configuration Files

Each system running CSP-Kerberos must have a */etc/krb5.conf* and an */krb5/lib/krb5kdc/kdc.conf* file. All files have sample copies provided in your CSP-Kerberos distribution that are installed in the */krb5* directory.

Here is a sample *kdc.conf* file:

```
[kdcdefaults]
   kdc_ports = 750,88
[realms]
   YOURSITE.COM = {
      profile = /etc/krb5.conf
      database_name = /krb5/lib/krb5kdc/principal
      key_stash_file = /krb5/lib/krb5kdc/.k5.YOURSITE.COM
      kdc_ports = 750,88
      max_life = 10h 0m 0s
      max_renewable_life = 7d 0h 0m 0s
      master_key_type = des-cbc-crc
      supported_enctypes = des-cbc-crc:normal des:normal des:v4
des:norealm des:onlyrealm des:afs3
    }
```

Also, the following lines are added to your */etc/services* file:

```
# CSP-Kerberos (Commercial Security Pak Kerberos) services
kerberos        88/udp kdc  # CSP-Kerberos authentication--udp
kerberos        88/tcp kdc  # CSP-Kerberos authentication--tcp
kerberos-sec    750/udp     # CSP-Kerberos authentication--udp
kerberos-sec    750/tcp     # CSP-Kerberos authentication--tcp
kerberos_master 751/udp     # CSP-Kerberos authentication
kerberos_master 751/tcp     # CSP-Kerberos authentication
kerberos-adm    749/tcp     # CSP-Kerberos 5 admin/changepw
kerberos-adm    749/udp     # CSP-Kerberos 5 admin/changepw
kpop            1109/tcp    # Pop with CSP-Kerberos
kshell          544/tcp cmd # and remote shell
klogin          543/tcp     # CSP-Kerberos authenticated rlogin
eklogin         2105/tcp    # CSP-Kerberos encrypted rlogin
krb5_prop       754/tcp     # CSP-Kerberos slave propagation
krb524          4444/tcp    # CSP-Kerberos 5 to 4 ticket xlator
#CSP-Kerberos test
sample          906/tcp     # CSP-Kerberos sample app server
```

If your master KDC or any of your slave KDCs is running Kerberos V4, (or if you will be authenticating to any Kerberos V4 KDCs in another realm) you will need to switch the

port number for kerberos to 750 and create a kerberos-sec service (tcp and udp) on port 88, so the Kerberos V4 KDC(s) will continue to work properly. The file */etc/services.N* can replace your */etc/services* file directly in this case.

Configuring CSP-Kerberos

The following sections provide configuration advice for all types of systems in your CSP-Kerberos realm.

Configuring All Participating Systems

Usually the CSP-Kerberos configuration file is named */etc/krb5.conf* in IRIX applications. This file contains information about the local CSP-Kerberos configuration. A sample */etc/krb5.conf* file follows:

```
# Sample /etc/krb.conf file
YOURSITE.COM
yoursite.com        krb_kdc_1
yoursite.com        krb_kdc_2
yoursite.com        krb_kdc_1 admin server
```

Line 1 specifies the name of the local realm. In the example file, this is simply *YOURSITE.COM*. (You are free to name your realm whatever you choose; however, the realm name in the */etc/krb5.conf* file must match the realm name used to create the CSP-Kerberos database on the CSP-Kerberos KDC; realm names are case sensitive.) Lines 2 and 3 list the host names for two CSP-Kerberos KDCS, krb_kdc_1 and krb_kdc_2. These are the KDCs that the CSP-Kerberos software asks for tickets. The software searches the */etc/krb5.conf* file from the top and tries each listed KDC until it obtains a response. (This implies that load balancing could be achieved by having some hosts list krb_kdc_2 before krb_kdc_1 in their */etc/krb5.conf* files.)

The last line indicates the location at which the CSP-Kerberos administrative server process is running. It is recommended that only one administrative server process be configured, because no mechanism is in place to propagate changes to the CSP-Kerberos principal database from a slave back to the primary server. For example, if there were multiple administrative server processes running, and a password change were made on a server, the new password would not be propagated back to the master.

Configuring CSP-Kerberos Servers

The IRIX operating system supports the following kerberized servers: *telnetd*(8), *ftpd*(8), *klogind*(8), and *rshd*(8). The *telnetd* server provides enhancements to the existing protocols to support CSP-Kerberos authenticated sessions. If the client applications connecting to these servers support CSP-Kerberos, and if they choose to use CSP-Kerberos, they are supported. Non-kerberized client applications also are supported, however, and their functionality has not been changed.

The *klogind* server supports kerberized *rlogind* connections. The new *klogind* daemon is the same as the non-kerberized version, except that it performs CSP-Kerberos authentication on the client before executing */bin/login* with the **-f** (force login) option. The *klogind* daemon listens on port 543. If the client *klogin* wants to establish a connection that is encrypted in addition to being authenticated by CSP-Kerberos, the client can try to connect to port 2105 on your system, at which *eklogind* will be listening. This server uses the session key, established as part of the CSP-Kerberos authentication, to encrypt and decrypt the data stream between the host and your system. The *eklogind* daemon is actually the same binary file as *klogind*, but it is initiated with different options from *inetd*(8).

The new *rshd* server supports kerberized *rshd* connections. This server is analogous to *rshd*, except that it performs CSP-Kerberos authentication on the client. The *rshd* daemon listens for connections on port 544. The *rshd* daemon supports kerberized versions of both *rsh*(1) and *rcp*(1).

The *ftpd* server supports additional comments that allow a kerberized *ftp*(1) to send authentication commands, protect commands in the control channel, and transfer files on the data channel using cryptographic checksum, SAFE, or encrypted file transfer PRIVATE, as well as the current clear text transfers.

The network services configuration file, */etc/services*, should be modified to support running CSP-Kerberos utilities or kerberized clients and servers on your system. Specifically, the following lines should be added:

```
# /etc/services file
klogin            543/tcp        # kerberized rlogin
kshell            544/tcp        # kerberized rshd
kerberos          750/udp        # CSP-Kerberos server
kerberos_master   751/tcp        # CSP-Kerberos administrator
eklogin           2105/tcp       # CSP-Kerberos encrypted rlogin
```

Note: Support for data stream encryption in kerberized applications is not available outside of the United States and Canada; therefore, the *eklogin* line should not be added at sites outside of the United States and Canada.

You can advertise CSP-Kerberos network services over NIS if you are running this optional software on your network domain. Consult the NIS documentation for information on advertising network services in your NIS domain.

In addition, the */etc/inetd.conf* file must be modified to support the invocation of kerberized services. The following lines must be added to */etc/inetd.conf* to support kerberized services on your system:

```
# /etc/inetd.conf file
kshell   stream   tcp   nowait   root   /etc/kshd      kshd
klogin   stream   tcp   nowait   root   /etc/klogind   klogind
eklogin  stream   tcp   nowait   root   /etc/klogind   eklogind
```

The first line (*kshell*) enables the kerberized *krshd* (8) server. The second line (*klogin*(8)) line enables the kerberized *klogind*(8) server. The third line (*eklogin*) enables the kerberized and encrypted *klogind* server. Obviously, these lines should not be added if your site does not run these kerberized servers on your Silicon Graphics system.

You must add the */.k5login* file at sites that use the *ksu*(1) client application. This file contains a list of users who are allowed to become superuser. A sample */.k5login* file follows:

```
bstrand.root@yoursite.com
test.root@yoursite.com
```

Each line contains *username.root@ LOCALREALM*; *username* is the user name of the process. The *ksu* program tries to fetch a ticket-granting ticket by using *username. root@ LOCALREALM*.

The */etc/krb.conf* file must have permissions set to 600 and be owned by **root**.

Configuring Automatic Ticket Granting With Telnet

You can configure the *krb5.conf* file to allow users accessing your CSP-Kerberos services through the *telnet* program to obtain tickets without the use of the *kinit* command. Edit your */etc/krb5.conf* file and ensure that the `krb5_get_tickets` entry is set to 1 as shown here:

```
krb5_get_tickets = 1
```

If this value is set to 1, the user is prompted for his or her CSP-Kerberos password before the telnet connection is made. If this value is set to 0, the user is prompted for his or her IRIX password, and then must execute *kinit* to receive tickets.

The */etc/krb5.conf* file should have permissions set to 644 and be owned by **root**.

Adding New Kerberized Server Programs

To add a new kerberized server application to your system, you must add an entry to the */etc/inetd.conf* file if the server is to be started by *inetd*. You can access the */etc/inetd.conf* file through the menu system. You can also start the server by using your local *netstart*(8) script, or you can start the server manually.

Consider the following example for adding an entry to the CSP-Kerberos principal database: Assume that you are the CSP-Kerberos administrator. You are adding a server that requires the use of a ticket called mathsrv and your system hostname is localsys. Assume that no other server on your host requires a ticket named mathsrv. You must perform some further steps to enable this server to run.

You must go to the primary CSP-Kerberos server system and use *kdb_edit* to add a new entry to the principal database. The name of the principal should be mathsrv, and its instance should be localsys. Run the *kadmin5* program to obtain a random password for the principal.

Note: The CSP-Kerberos administrator cannot select this password. It must be a random password generated by the *kadmin5* program, which allows the administrator to edit the principal database. Because this password will never need to be typed in by any person, its content is unimportant.

After you have entered mathsrv.localsys into the CSP-Kerberos principal database, extract a new *v5srvtab* file from the database with the *kadmin5* command and move that file onto the localsys host. The MIT distribution of Kerberos version 4 provides a program call *ext_srvtab* to perform this task. When given the name localsys, the *ext_srvtab* program scans the database for all services registered with instances of localsys. Then it writes the names and passwords for these services to a file called *new-localsys-srvtab*. This file must be treated with caution, since it contains passwords to all registered services on your system.

Once the initial KDC database is created and loaded and the keytab file with an entry for *kadmind5* is extracted, you no longer need to use *kdb5_edit* to add new entries in the principals database. You can use the *kadmin5* command thereafter.

Transfer this file to localsys in whatever secure manner is appropriate. For example, encrypted and transferred using *ftp*, put onto a tape, and so on. Name this file */etc/v5srvtab* on localsys, set its permissions to 600, and ensure that it is owned by **root**. Double check the permissions to avoid a security breach.

Configuring CSP-Kerberos Clients

This distribution provides the following CSP-Kerberos client programs:

rcp(1) Copies files between systems that use CSP-Kerberos authentication when they connect to the remote host.

rlogin(1) Connects a terminal on the current local host system to a remote host that uses CSP-Kerberos authentication to determine authorization. Users who want to call the CSP-Kerberos authenticated remote login feature can use this command.

rsh(1) Uses CSP-Kerberos authentication to connect to the specified host, and then executes the specified command.

ksu(1) Uses CSP-Kerberos authentication to allow superuse of another account.

telnet(1B) Uses TELNET protocol to communicate with a remote host. Users who want to use CSP-Kerberos authenticated *telnet* can either use the *telnet* command with the **-a** option and *hostname* variable, or use the *telnet* command followed by passing the *open* command with the **-a** option and the *hostname* variable to *telnet*.

Users who want to run encrypted and CSP-Kerberos authenticated *telnet*(1B) sessions should escape into *telnet* and enter the following commands:

```
set autoencrypt (output)
set autodecrypt (input)
open -a hostname
```

If you do not set any of these authentication or encryption options, *telnet* will establish a unauthenticated session for the user. Before running CSP-Kerberos authenticated *telnet*, a user must acquire a service-granting ticket from CSP-Kerberos by running *kinit*(1) on your system.

	Unless a user uses an encrypted session when logging into your system, the CSP-Kerberos password will traverse the network from the local host to your system without being encrypted. Use *klogin* **-x** or the encrypted option of *telnet* to avoid this problem.
ftp(1B)	Uses the FTP protocol to authenticate clients to servers with the AUTH and ADAT commands. Allows transfer of files in clear text, cryptographic checksum, or encrypted form. All commands sent over the control channel are protected by base 64 encoding with optional encryption if the private command is selected. (The private command is not available outside of the United States and Canada.)

Users can also use the following commands:

- *kinit* (to obtain a ticket-granting-ticket).
- *klist* (to list CSP-Kerberos tickets).
- *kdestroy* (to destroy CSP-Kerberos tickets)
- *kpasswd* (to change CSP-Kerberos passwords).

Users should be encouraged to include the *kdestroy* command in their *.logout* files; this command helps ensure that their ticket files are not available to unauthorized users after they have logged off.

Adding New Kerberized Client Programs

If the servers were installed correctly, the only action you must take to add CSP-Kerberos clients is to move the program to the directory in which you want the client to reside. (If you want to use library routines such as *getservbyname* (see *getserv*(3)) to obtain the remote port number, you may need to add entries to the */etc/services* file, but these tasks are not specifically related to CSP-Kerberos.)

Adding New CSP-Kerberos Users

The procedure for adding new CSP-Kerberos users is similar to that for adding new CSP-Kerberos servers. Both involve editing the principal database on the primary CSP-Kerberos server. Unlike the server, the user needs a password that can be remembered. Thus, when the *kdb_edit* (or equivalent) program prompts the CSP-Kerberos administrator for a password, the CSP-Kerberos administrator must type in something that the user can remember and type in again. The CSP-Kerberos

administrator can have the user present, to type in his or her own password, or the CSP-Kerberos administrator can choose an initial password for the user and then have the user change it by using the *kpasswd* command. These are the same procedures used when system administrators create new accounts for users.

Configuring Your Firewall to Work With CSP-Kerberos

If you need offsite users to be able to get CSP-Kerberos tickets in your realm, they must be able to get to your KDC. This requires either that you have a slave KDC outside your firewall, or you configure your firewall to allow UDP requests into to at least one of your KDCs, on whichever port the KDC is running. (The default is port 88; other ports may be specified in the KDC's */krb5/lib/krb5kdc/kdc.conf* file.) Similarly, if you need off-site users to be able to change their passwords in your realm, they must be able to get to your CSP-Kerberos admin server. The default port for the admin server is 749.

If your onsite users inside your firewall need to get to KDCs in other realms, you must configure your firewall to allow outgoing TCP and UDP requests to port 88. Additionally, if they will need to get to any Kerberos V4 KDCs, you may also need to allow TCP and UDP requests to port 750. If your onsite users inside your firewall need to get to CSP-Kerberos admin servers in other realms, you will also need to allow outgoing TCP and UDP requests to port 749.

If any of your KDCs are outside your firewall, you must allow *kprop* requests to get through to the remote KDC. The *kprop* program uses the *krb5_prop* service on port 754 (tcp).

If you need your offsite users to have access to systems inside your firewall, you need to allow TCP connections from their offsite hosts on the appropriate ports for the programs they will be using. The following lines from */etc/services* show the default port numbers for the CSP-Kerberos programs:

```
# CSP-Kerberos (Commercial Security Pak Kerberos) services
kerberos        88/udp kdc  # CSP-Kerberos authentication--udp
kerberos        88/tcp kdc  # CSP-Kerberos authentication--tcp
kerberos-sec    750/udp     # CSP-Kerberos authentication--udp
kerberos-sec    750/tcp     # CSP-Kerberos authentication--tcp
kerberos_master 751/udp     # CSP-Kerberos authentication
kerberos_master 751/tcp     # CSP-Kerberos authentication
kerberos-adm    749/tcp     # CSP-Kerberos 5 admin/changepw
kerberos-adm    749/udp     # CSP-Kerberos 5 admin/changepw
kpop            1109/tcp    # Pop with CSP-Kerberos
```

```
kshell          544/tcp cmd # and remote shell
klogin          543/tcp     # CSP-Kerberos authenticated rlogin
eklogin         2105/tcp    # CSP-Kerberos encrypted rlogin
krb5_prop       754/tcp     # CSP-Kerberos slave propagation
krb524          4444/tcp    # CSP-Kerberos 5 to 4 ticket xlator
#CSP-Kerberos test
sample          906/tcp     # CSP-Kerberos sample app server
```

By default, CSP-Kerberos *telnet* and *ftp* use the same ports as the standard *telnet* and *ftp* programs, so if you already allow such connections through your firewall, the CSP-Kerberos versions will get through as well. If you do not already allow such connections through your firewall, but need your users to be able to use CSP-Kerberos connections, you can either allow *ftp* and *telnet* connections on the standard ports, or switch these programs to non-default port numbers and allow connections on those ports to get through.

CSP-Kerberos *klogin* uses the *klogin* service, which by default uses port 543. Encrypted CSP-Kerberos *klogin -x* uses the *eklogin* service, which by default uses port 2105.

CSP-Kerberos *rsh* uses the *kshell* service, which by default uses port 544. However, the server must be able to make a TCP connection from the *kshell* port to an arbitrary port on the client, so if your users are to be able to use *rsh* from outside your firewall, the server they connect to must be able to send outgoing packets to arbitrary port numbers. Similarly, if your users need to run *rsh* from inside your firewall to hosts outside your firewall, the outside server needs to be able to connect to an arbitrary port on the system inside your firewall. Because CSP-Kerberos *rcp* uses *rsh*, the same issues apply. If you need to use *rsh* (or *rcp*) through your firewall and are concerned with the security implications of allowing connections to arbitrary ports, Silicon Graphics suggests that you have rules that specifically name these applications and, if possible, list the allowed hosts.

A reasonably good cookbook for configuring firewalls is available by FTP from *ftp.livingston.com*, in the location */pub/firewall/firewall-1.1.ps.Z*. The book *UNIX System Security*, by David Curry, is also a good starting point.

Ports for the KDC and Admin Services

The default ports used by CSP-Kerberos are port 88 for the KDC and port 749 for the admin server. You can, however, choose to run on other ports, as long as they are specified in each host's */etc/services* and *krb5.conf* files, and the */krb5/lib/krb5kdc/kdc.conf* file

on each KDC. Because the *kadmin5* port was recently assigned, Silicon Graphics recommends that you specify it explicitly in your *krb5.conf* and *kdc.conf* files.

Configuring CSP-Kerberos DCE Interoperability

If you are configuring Silicon Graphics CSP-Kerberos into an existing installation using a Distributed Computing Environment KDC, you must make the following changes to your configuration for compatibility.

1. You must install the software package *csp-kerberos.sw.server-dce-interop* from your CSP-Kerberos software distribution on any servers that will be interoperating with a DCE installation.
2. The keytab file in Silicon Graphics CSP-Kerberos is */etc/v5srvtab*. However in DCE this file is found in */krb5/v5srvtab*. Create a link between these two locations with this command:

   ```
   ln /etc/v5srvtab /krb5/v5srvtab
   ```

3. The network service name must be aliased in the */etc/services* file. The CSP-Kerberos service is named `kerberos` in the Silicon Graphics implementation. DCE implementations expect to find the service called `kerberos5` on port 88.

   Thus, you must create an alias for the `kerberos` entry to also respond to the name `kerberos5`:

   Here is the standard DCE entry in */etc/services*:

   ```
   kerberos5  88/udp  kdc  # CSP-Kerberos 5/ DCE KDC
   ```

   The Silicon Graphics implementation entry is as follows:

   ```
   kerberos  88/udp  kdc  # CSP-Kerberos authentication--udp
   ```

   Add this alias entry to the */etc/services* file to allow interoperation:

   ```
   kerberos  88/udp  kerberos5 kdc  # CSP-Kerberos 5 / DCE KDC
   ```

4. You must make sure of the following settings in the */etc/krb5.conf* file:

   ```
   kdc_req_checksum_type = 2
   ```

   ```
   ccache_type = 1
   ```

   Setting the `kdc_req_checksum_type` entry to 2 forces the software to use CKSUMTYPE_RSA_MD4 for checksum service. This is the standard for DCE 1.1 and earlier compatibility. The Silicon Graphics implementation default is CKSUMTYPE_RSA_MD5.

Setting the `ccache_type` entry to 2 allows for current DCE compatibility. Set this value to 1 for DCE 1.0.3a and earlier systems and set it to 2 for DCE 1.1 systems.

An example */etc/krb5.conf* file looks similar to this:

```
[libdefaults]
   default_realm = laughter.rain.com
   kdc_req_checksum_type = 2
   ccache_type = 1
[realms]
   laughter.rain.com = {
          kdc = laughter
          admin_server = laughter.rain.com
          default_domain = rain.com
          }

[domain_realm]
   .rain.com = laughter.rain.com

[login]
   krb4_get_tickets = 0
   krb4_convert = 0

[logging]
   kdc = FILE:/var/log/krb5kdc.log
   admin_server = FILE:/var/log/kadmin.log
   default = FILE:/var/log/krb5lib.log
```

5. DCE Kerberos uses a specific form of credential that must be converted to be compatible with the Silicon Graphics implementation of CSP-Kerberos. To allow *klogind* to convert a Silicon Graphics ticket to a DCE credential, the program */krb5/sbin/k52dce* is included in your distribution. For complete information about this program, see the *k52dce* reference page.

CSP-Kerberos Administrative Commands

Your CSP-Kerberos database contains all of your realm's CSP-Kerberos users, called "principals," their passwords, and other administrative information about each principal. For the most part, you use the *kdb5* utilities to manipulate the CSP-Kerberos database as a whole, and the *kadmin5* program to make changes to the entries in the database. (One notable exception is that users use the *kpasswd* program to change their own passwords.) The *kadmin5* program has its own command-line interface, to which you type the database administration commands.

The *kadmin5* command provides for the maintenance of CSP-Kerberos principals and KADM5 policies. It exists as a CSP-Kerberos client, *kadmin5*, using CSP-Kerberos authentication to operate securely from anywhere on the network.

The *kdb5* tools provide a means to create, delete, edit, load, or dump a CSP-Kerberos database. They also include a command to stash a copy of the master database key in a file on a KDC, so that the KDC can authenticate itself to the *kadmind5* and *krb5kdc* daemons at boot time.

The kadmin5 Command and Its Options

You can invoke *kadmin5* with any of the following options:

-r *REALM* Use *REALM* as the default CSP-Kerberos realm for the database.

-k *keytab* Use the specified keytab file to decrypt the KDC response instead of prompting for a password on the TTY. In this case, the principal is host/hostname.

-p *principal* Use the CSP-Kerberos principal *principal* to authenticate to CSP-Kerberos.

If this option is not given, *kadmin5* will append *admin* to either the primary principal name, the environment variable USER, or to the username obtained from *getpwuid*, in order of preference.

-c *credentials cache*
Use *credentials_cache* as the credentials cache. The credentials cache should contain a service ticket for the kadmin5/admin service, which can be acquired with the *kinit* program. If this option is not specified, *kadmin5* requests a new service ticket from the KDC, and stores it in its own temporary credentials cache.

-l *Time* Use *Time* as the lifetime of an administrative ticket.

-d Specifies that the credentials cache is to be deleted after use.

-s Specifies that the credentials cache is to be saved after use.

-m Specifies that multiple operations will be permitted for only one entry of the administrative principal's password.

The kadmin5 Access Control List File

You must have an ACL file in */krb5/lib/krb5kdc* in order to run *kadmin5*. The ACL file specifies access control rules for CSP-Kerberos administrators. Note that this ACL file is in no way related to the ACL services provided in the IRIX portion of the Commercial Security Pak software distribution.

The ACL file controls which principals can or cannot perform which administrative functions on which principals. This file can contain comment lines, null lines or lines which contain ACL entries. Comment lines start with the hashmark sign (#) and continue until the end of the line. Lines containing ACL entries have the format:

```
principal operation-mask [operation-target]
```

Ordering is important. The first matching entry is the one which controls access for a particular principal on a particular host.

Here is a typical ACL file entry:

```
root/admin@YOURSITE.COM admci
```

The ACL file can either be specified on the command line for *kadmind5* as follows:

```
kadmind5 -a /krb5/lib/krb5kdc/<custom.acl>
```

or you can use the default */krb5/lib/krb5kdc/krb5_adm.acl*.

The ACL file can be created with any text editor, it is a plain ASCII text file.

For complete information on the syntax for ACL file entries, see the *kadmind5*(8) reference page.

The kadmin5 Date Format

Many of the *kadmin5* commands take a duration or time as an argument. The date can appear in a wide variety of formats, such as those shown below:

```
"15 minutes"
"7 days"
"1 month"
"2 hours"
"400000 seconds"
"next year"
"this Monday"
```

```
"next Monday"
yesterday
tomorrow
now
"second Monday"
fortnight
"3/31/92 10:00:07 PST"
"January 23, 1987 10:05pm"
"22:00 GMT"
```

Note that if the date specification contains spaces, you must enclose it in double quotes. Note also that you cannot use a number without a unit. (For example, “60 seconds” is correct, but “60” is incorrect.) All keywords are case insensitive. The following is a list of all of the allowable keywords:

Months	january, jan, february, feb, march, mar, april, apr, may, june, jun, july, jul, august, aug, september, sept, sep, october, oct, november, nov, december, dec
Days	sunday, sun, monday, mon, tuesday, tues, tue, wednesday, wednes, wed, thursday, thurs, thur, thu, friday, fri, saturday, sat
Units	year, month, fortnight, week, day, hour, minute, min, second, sec
Relative	tomorrow, yesterday, today, now, last, this, next, first, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth, ago
Time Zones	The kadmin5 command recognizes abbreviations for most of the world's time zones. A complete listing appears in Appendix A of this guide.
12-Hour Time Delimiters	am, pm

CSP-Kerberos Principal Commands

Each entry in the CSP-Kerberos database contains a CSP-Kerberos principal (a user) and the attributes and policies associated with that principal.

Retrieving Information About a Principal

To retrieve a listing of the attributes or policies associated with a principal, use the *kadmin5 show_principal* command, which requires the “inquire” administrative privilege. The syntax is

show_principal *principal*

The *show_principal* command has the alias *show.*

For example, suppose you wanted to view the attributes of the principals zursch/root@YOURSITE.COM and sysadm@YOURSITE.COM. You would type the following

```
kadmin5
kadmin5: show zursch/root
Principal: zursch/root@YOURSITE.COM
Key version: 3
Maximum life: 1 day 00:00:00
Maximum renewable life: 7 days 00:00:00
Master key version: 1
Expires: Mon Jan 18 22:14:07 EDT 2038
Password expires: Mon Sep 19 14:40:00 EDT 1996
Password last changed: Mon Jan 31 02:06:40 EDT 1996
Last modified: by root/admin@YOURSITE.COM on Wed Jul 13 18:27:08 EDT
1996
Attributes: DISALLOW_FORWARDABLE, DISALLOW_PROXIABLE, REQUIRES_HW_AUTH
Salt type: DEFAULT
```

Retrieving a List of Principals

To generate a listing of principals, use the *kadmin5 list_db* command, which requires the "list" privilege. The syntax is:

list_db *expression*

where *expression* is a shell-style "glob" regular expression that can contain the characters *, ?, [, and]. All policy names matching the expression are displayed. The *list_db* command has the alias *ldb*. For example:

```
kadmin5: ldb test*
test3@YOURSITE.COM
test2@YOURSITE.COM
test1@YOURSITE.COM
testuser@YOURSITE.COM
```

If no expression is provided, all principals are printed.

Adding or Modifying Principals

To add a principal to the database, use the *kadmin5 add_new_key* command. The syntax is

add_new_key [*options*] *principal*

To modify attributes of a principal, use the *kadmin5 modify_entry* command. The syntax is

modify_entry [*options*] *principal*

The *add_new_key* command has the alias *ank*. The *modify_entry* command has the alias *modent*.

The *add_new_key* and *modify_entry* commands take the following switches

-expiration *date*
Sets the expiration date of the principal to *date*.

-pwexpiration *date*
Sets the expiration date of the password to *date*.

-maxlife *maxlife*
Sets the maximum ticket life of the principal to *maxlife*.

-maxrenewlife *maxlife*
Sets the maximum renewable ticket life of the principal to *maxlife*.

If you simply want to use the default values, simply use the following process:

```
kadmin5: ank zursch
WARNING: no policy specified for "zursch@YOURSITE.COM";
defaulting to no policy.
Enter password for principal zursch  <Enter Password>
Re-enter password for principal zursch <Re-Enter Password>
Principal "zursch" created.
```

If, on the other hand, you want to set up an account that expires on January 1, 2000, you would type the following.

```
kadmin5: ank jennifer -expiration "1/1/2000 12:01am EST"
Enter password for principal jennifer: <Enter Password>
Re-enter password for principal jennifer: <Re-Enter Password>
Principal "jennifer@YOURSITE.COM" created.
```

If you need cross-realm authentication, add principals for the other realm's TGT to each realm. For example, if you need to do cross-realm authentication between the realms

YOURSITE.COM and THEIRSITE.COM, add the principals krbtgt/theirsite.com@YOURSITE.COM and krbtgt/yoursite.com@THEIRSITE.COM to both databases. Be sure the passwords and the key version numbers are the same in both databases.

Deleting Principals

To delete a principal, use the *kadmin5 delete_principal* command, which requires the "delete" administrative privilege. The syntax is

```
delete_entry principal
```

The *delete_entry* command has the alias *delent*. For example:

```
delent jennifer
Are you sure you want to delete the principal jennifer@YOURSITE.COM?
(yes/no): yes
Principal "jennifer@YOURSITE.COM" deleted.
```

Renaming Principals

To rename a principal, use the *kadmin5 rename_entry* command. The syntax is

```
rename_entry old_principal new_principal
```

The *rename_entry* command has the alias *renent*. For example:

```
renent test test0
Are you sure you want to rename the principal test@YOURSITE.COM to
test0@YOURSITE.COM"? (yes/no): yes
Principal "test@YOURSITE.COM" renamed to "test0YOURSITE.COM".
```

CSP-Kerberos Password Commands

The user's password is the fundamental level of security within CSP-Kerberos. All the standard rules regarding the use and selection of good passwords should be enforced under CSP-Kerberos. See the *IRIX Admin: Backup, Security, and Accounting* guide for more information on good password selection rules. Remember, though, that CSP-Kerberos passwords are in no way related to your system password used on IRIX, except that all passwords on any system should be well-chosen.

Changing Passwords

To change a principal's password, use the *change_pwd_key* command within *kadmin5*. The syntax is

change_pwd_key [*options*] *principal*

The *change_pwd_key* command has the alias *cpw*. For example:

```
cpw zursch
Enter password for principal zursch@YOURSITE.COM: <Enter Password>
Re-enter password for principal zursch@YOURSITE.COM <Re-Enter Password>
Password for zursch@YOURSITE.COM changed.
```

Note that *change_pwd_key* will not let you change the password to one that is in the principal's password history.

CSP-Kerberos Database Backup Commands

As with any file, it is possible that your CSP-Kerberos database could become corrupted. If this happens on one of the slave KDCs, you might never notice, since the next automatic propagation of the database would install a fresh copy. However, if it happens to the master KDC, the corrupted database would be propagated to all of the slaves during the next propagation. For this reason, Silicon Graphics recommends that you back up your CSP-Kerberos database regularly. Because the master KDC is continuously dumping the database to a file in order to propagate it to the slave KDCs, it is a simple matter to have a *cron* job periodically copy the dump file to a secure system elsewhere on your network. (Of course, it is important to make the host where these backups are stored as secure as your KDCs, and to encrypt its transmission across your network.) Then if your database becomes corrupted, you can load the most recent dump onto the master KDC.

Dumping a CSP-Kerberos Database to a File

To dump a CSP-Kerberos database into a file, use the *kdb5_edit dump_db* command on one of the KDCs. The syntax is

kdb5_edit dump_db [*filename* [*principals...*]]

For example:

```
kdb5_edit dump_db dumpfile
kadmin5/admin@YOURSITE.COM
krbtgt/yoursite.com@YOURSITE.COM
kadmin5/history@YOURSITE.COM
K/M@YOURSITE.COM
kadmin5/changepw@YOURSITE.COM
```

If you specify which principals to dump, you must use the full principal, as in the following example:

```
kdb5_edit dump -verbose dumpfile
K/M@YOURSITE.COM
kadmin5/admin@YOURSITE.COM
kadmin5/admin@YOURSITE.COM
K/M@YOURSITE.COM
```

Otherwise, the principals will not match those in the database and will not be dumped:

```
kdb5_edit dump dumpfile
K/M
kadmin5/admin
```

If you do not specify a dump file, *kdb5_edit* will dump the database to the standard output.

Backups of Secure Hosts

When you back up a secure host, you should exclude the host's keytab file from the backup. If someone obtained a copy of the keytab from a backup, that person could make any host masquerade as the host whose keytab was compromised. This could be particularly dangerous if the compromised keytab was from one of your KDCs. If the system has a disk crash and the keytab file is lost, it is easy to generate another keytab file. If you are unable to exclude particular files from backups, you should ensure that the backups are kept as secure as the host's **root** password.

Restoring a CSP-Kerberos Database From a Dump File

To restore a CSP-Kerberos database dump from a file, use the *kdb5_edit load_db* command on one of the KDCs. The syntax is

kdb5_edit load_db *dumpfilename dbname* [*admin_dbname*]

For example:

kdb5_edit load_db *dumpfile principal*

Creating and Destroying CSP-Kerberos Databases

If you need to create a new CSP-Kerberos database, use the *kdb5_create* command. The syntax is

kdb5_create

For example:

```
/krb5/sbin/kdb5_create
kdb5_create: No such file or directory while setting active database
to '/krb5/principal'
Initializing database '/krb5/lib/krb5kdc/principal' for
realm 'YOURSITE.COM',
master key name 'K/M@YOURSITE.COM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:  <Type the master key>
Re-enter KDC database master key to verify:  <Type it again>
```

If you need to destroy a CSP-Kerberos database, use the *kdb5_destroy* command. The syntax is:

kdb5_destroy

All traces of the database will be removed.

CSP-Kerberos Keytab Commands

A keytab is a host's copy of its own keylist, which is analogous to a user's password. A server that needs to authenticate itself to the KDC has to have a keytab that contains its own principal and key. Just as it is important for users to protect their passwords, it is equally important for servers to protect their keytabs. You should always store keytab files on local disk, and make them readable only by **root**, and you should never send a keytab file over a network in the clear. Ideally, you should run the *kadmin5* command to extract a keytab on the host on which the keytab is to reside.

The Keytab File

All CSP-Kerberos servers need a keytab file, called */etc/v5srvtab*. The keytab was called an *srvtab* in version 4 of Kerberos. The *v5srvtab* file has not been renamed to reflect the change in terminology (calling it a keytab file.) The keytab file is an encrypted, local, on-disk copy of the host's key. The keytab file, like the stash file, is a potential point of entry for a break in, and if compromised, would allow unrestricted access to its host. The keytab file must be readable only by **root**, and should exist only on the system's local disk. The file should not be part of any backup of the system, unless access to the backup data is secured as tightly as access to the system's **root** password itself.

In order to generate a keytab for a host, the host must have a principal in the CSP-Kerberos database. The procedure for adding hosts to the database is described fully in the section of this guide titled "CSP-Kerberos Principal Commands." The keytab is generated by running *kadmin5* and issuing the *ktadd* command.

For example, to generate a keytab file to allow the host trillium.yoursite.com to authenticate for the services host and sample, the administrator would issue the command (on trillium, within the *kadmin5* utility):

```
ktadd host/trillium.yoursite.com sample/trillium.yoursite.com
kadmin: Entry for principal host/trillium.yoursite.com@YOURSITE.COM with kvno 3,
encryption type DES-CBC-CRC added to keytab  WRFILE: /etc/v5srvtab.
kadmin: Entry for principal sample/trillium.yoursite.com@YOURSITE.COM with kvno
3, encryption type DES-CBC-CRC added to keytab  WRFILE: /etc/v5srvtab.
kadmin: quit
```

If you generate the keytab file on another host, you need to get a copy of the keytab file onto the destination host (trillium, in the example above) without sending it unencrypted over the network. If you have installed the CSP-Kerberos client programs, you can use encrypted *rcp*.

The ktutil Command

Keytab files are manipulated with the *ktutil* utility. This utility is similar in formation to the *kadmin5* and *kdb5* utilities. The available commands are:

clear_list, *clear* Clear the current keylist.

read_kt, *rkt* Read a krb5 keytab into the current keylist.

read_st, *rst* Read a krb4 srvtab into the current keylist.

write_kt, *wkt* Write the current keylist to a krb5 keytab.

write_st, *wst* Write the current keylist to a krb4 srvtab.

delete_entry, *delent*
Delete an entry from the current keylist.

list, *l* List the current keylist.

list_requests, *lr*, *?*
List available requests.

quit, *exit*, *q* Exit program.

Note that keylist files were called srvtab files in Kerberos version 4 and are called keytab files in version 5.

Clock Skew Commands

In order to prevent intruders from resetting their system clocks in order to continue to use expired tickets, CSP-Kerberos is set up to reject ticket requests from any host whose clock is not within the specified maximum clock skew of the KDC (as specified in the */krb5/lib/krb5kdc/kdc.conf* file). Similarly, hosts are configured to reject responses from any KDC whose clock is not within the specified maximum clock skew of the host (as specified in the *krb5.conf* file). The default value for maximum clock skew is 300 seconds (five minutes).

Silicon Graphics suggests that you add a line to client systems' */etc/rc* files to synchronize the system's clock to your KDC at boot time. On IRIX hosts, assuming you had a KDC called timeserver in your realm, this would be

```
gettime -s timeserver
```

If the host is not likely to be rebooted frequently, you may also want to set up a *cron* job that adjusts the time regularly.

Getting Correct DNS Information

Several aspects of CSP-Kerberos rely on name service. In order for CSP-Kerberos to provide its high level of security, it is less forgiving of name service problems than some other parts of your network. It is important that your Distributed Name Service (DNS) entries and your hosts have the correct information.

Each host's canonical name must be the fully qualified hostname (including the domain), and each host's IP address must reverse-resolve to the canonical name.

Other than the localhost entry, make all entries in each system's */etc/hosts* file in the following form:

```
IP address      fully-qualified hostname        aliases
```

Here is a sample */etc/hosts* file:

```
# this is a comment
127.0.0.1       localhost localhost@yoursite.com
127.150.83.5    laughter@yoursite.com trillium wake-robin
```

Finally, each host's keytab file must include a host/key pair for the host's canonical name. You can list the keys in a keytab file by entering the command *klist -K*. For example:

```
klist -K

Keytab name: /etc/v5srvtab
KVNO Principal
---- ---------------------------------------
1    host/laughter.yoursite.com@YOURSITE.COM
```

If you use the *telnet* command to reach the host with a fresh credentials cache (ticket file), and then use the *klist* command, the host's service principal should be host/fully-qualified-hostname@REALM_NAME

CSP-Kerberos Libraries

CSP-Kerberos libraries include the following:

libacl.a	Provides a (crude) mechanism to allow servers to manage authorization checking.
libcom_err.a	CSP-Kerberos error messages.
libcrypto.a	Cryptography, checksum, and message digest functions. In the international version, *libkadm.a*, this library contains functions used by the *kadmin* client and server
libdes425.a	Link-level compatibility library, lets you link V4 applications with V5 back-end code. Also includes encryption/checksum/md functions in the domestic version.
libgssapi.a	This library contains gssapi mechanism routines for CSP-Kerberos V5.
libgssapi_krb5.a	This library contains gssapi krb5 specific routines.
libkadm.a	Provides interfaces to the CSP-Kerberos database administration server.
libkdb5.a	Support functions for the KDC administration. No encryption/decryption functions are available in this library.
libknet.a	Provides an interface to some lower-level network management routines required by some CSP-Kerberos applications.
libkrb.a	Handles most of the CSP-Kerberos protocol. Provides the routines that perform CSP-Kerberos data encryption, which uses the Data Encryption Standard (DES) algorithms. **Note:** Support for data stream encryption in kerberized applications is not available outside of the United States and Canada.
libkrb4.a	This library contains krb4 compatibility functions.
libkrb524.a	This library contains krb5 to krb4 conversion functions.
libkrb5util.a	This library contains convenience sendauth/recvauth functions, with compatibility with V4 recvauth.
libpty.a	This library contains pty functions for bsd utilities.

All CSP-Kerberos libraries are installed in the */usr/lib* directory.

Several things should be noted about the CSP-Kerberos libraries. First, *libkadm.a* is used to build only *kpasswd*(1); *libkadm.a* is not designed to have user interfaces. Second, *libknet.a* is also used to build some CSP-Kerberos applications; like *libkadm.a*, it is not designed to be used by user-level applications. Finally, *libkrb.a*, as it appears in the default distribution, contains only routines that return without performing any encryption function. If CSP-Kerberos is built only from the source code on the default release media, it will not function properly.

The domestic (United States and Canada) release of CSP-Kerberos includes encryption software in *libcrypto.a*. The International release does not.

Troubleshooting CSP-Kerberos

Debugging CSP-Kerberos problems can be a complex task. The following mechanisms are designed to assist administrators in tracking down CSP-Kerberos problems:

CSP-Kerberos Log Files

Most implementations of the CSP-Kerberos server include tracing mechanisms that write messages to log files.The CSP-Kerberos server process writes messages to the */var/log/kadmin.log* file; the Kerberos KDC process writes to the */var/log/krb5kdc.log* file; and the default log file is */var/log/krb5lib.log* for other log entries. These files can contain useful information concerning CSP-Kerberos error conditions.

Logging is controlled by a section of the krb5.conf file, reproduced here:

```
[logging]
   kdc = FILE:/var/log/krb5kdc.log
   admin_server = FILE:/var/log/kadmin.log
   default = FILE:/var/log/krb5lib.log
```

These log file locations can be changed by altering this section of the */etc/krb5.conf* file.

Debug Flags

The CSP-Kerberos libraries are equipped with several compiled-in debug flags, which can be set from applications, and they tell the library to print relevant information to stderr. Currently, there are two flags: **krb_debug** and **krb_ap_req_debug**. The former is designed primarily for administrators who might be expected to have a more detailed understanding of the CSP-Kerberos libraries; the latter is designed primarily for application developers.

To use the debug flags from within a user's program, place the following code in the user's program:

```
extern int krb_debug;
extern int krb_ap_req_debug;
main()
{
/* variable declarations here...*/
    krb_debug = 1;
    krb_ap_req_debug = 1;
/* the rest of the code here */
}
```

This code tells the library that the user wants to see the diagnostic information. Some diagnostic information may be written to the */var/log/krb5lib.log* file on your system when this level of debugging is enabled.

Note: The */var/log* directory must exist for this log file to be created.

Error Messages

The following is a list of common error messages that may be encountered when installing and running CSP-Kerberos. Each message is followed by an explanation.

program : Could not get ticket for realm *realm* instance *instance*. No ticket file *tf_util*
: This message indicates that the user must run kinit(1) on the host to obtain a ticket-granting ticket.

krb_sendauth failure or krb_mk_req: get_ad_tkt failed
: These messages occur most often when the user requests kerberized service that is not provided on the specified host. They indicate that the remote host is not registered for the specified service.

program: Authentication failed
: This message indicates that the */etc/v5srvtab* file on the server probably has not been updated to reflect the CSP-Kerberos principal database.

Principal unknown (csp-kerberos)
: This message indicates that the */etc/krb.conf* file is not publicly readable, it does not contain a valid CSP-Kerberos server entry, or no valid CSP-Kerberos server could be contacted.

program : Could not get ticket for realm *realm*, instance *instance*: Principal unknown
: This message indicates that the principal in question has not been added to the CSP-Kerberos database.

Retry count exceeded (send_to_kdc)
: This message indicates that the client process could not communicate with the CSP-Kerberos server. The CSP-Kerberos server process may have died, or the communications path between the client process and the CSP-Kerberos server may be down.

kpasswd: Could not connect to server attempting to change password
: This message indicates that the client process (in this case, *kpasswd*) could not communicate with the CSP-Kerberos administrative server. The kadmind5 process may have died, or the communications path between the client process and the CSP-Kerberos server may be down.

Known CSP-Kerberos Weaknesses

This CSP-Kerberos implementation has several known weaknesses. Some of these weaknesses are specific to the IRIX operating system; however, most of them are problems of the CSP-Kerberos version 5 protocol or the CSP-Kerberos authentication paradigm. The weaknesses are as follows:

- Time synchronization is not secure.

  Because CSP-Kerberos uses encrypted time stamps as part of its replay detection scheme, CSP-Kerberos replay detection can be broken if the protocol used to synchronize the system clock can be broken. That is, a user who steals a ticket that was good for last Friday cannot try to use it directly, because CSP-Kerberos can detect that it has expired; however, if the user can get the system to think that it is last Friday, CSP-Kerberos will be unable to detect that the ticket has expired.

  This implementation of the network time protocol (NTP) does not provide any sort of authentication mechanism to ensure that the clock updates it receives are genuine; therefore, it could be tricked into setting its clock back.

  Newer versions of NTP support authenticated updates, but the authentication is clumsy and requires nontrivial efforts to be put into key management issues.

- Ticket information is stored in a file.

  CSP-Kerberos tickets are kept in files named */tmp/tkt_uid* on the IRIX file system. These files have permissions of 600 and are owned by the user; however, this mechanism is not entirely secure. For example, any user with **root** privileges can read and even modify the contents of the ticket file.

  Tickets must be stored on the client, either on disk or in memory. These locations are vulnerable to an attack by **root**.

  By itself, the ticket does not allow the malicious user to impersonate the legitimate user. All CSP-Kerberos servers insist on receiving an authenticator in addition to a valid ticket. The authenticator is a message encrypted by the session key stored in the ticket. Unfortunately, the session key is stored along with the ticket in the user's ticket file; therefore, anyone who can read the ticket can read the key and build an authenticator from it.

  The tickets are kept in */tmp* intentionally. It is inadvisable to allow ticket files to survive beyond the lifetime of the user session that created them, and even less advisable for them to survive across system reboots.

- Server key information is stored in a file.

  Like ticket information, CSP-Kerberos stores server password information in a file, */etc/v5srvtab*, which should be installed with permissions of 600 and should be owned by **root**. However, users gaining access to this file could use the information to create and install their own fake CSP-Kerberos servers. These servers would not be detectable by client processes, even those requesting two-way authentication.

- Automatic replay detection is not available.

  The MIT Kerberos implementers have not released the code for *krb_ck_repl;* the routine that checks for replay attempts. Users who want to check for replays must have their servers cache their own tickets and authenticators.

- CSP-Kerberos passwords are subject to offline, brute force attacks.

  A person can use *kinit*(1) to request CSP-Kerberos to send a packet encrypted with the key derived from any user's password. The cryptographer can then take this packet and attempt to break its encryption at his or her leisure. Although there is currently no well known way to break the DES encryption quickly, the cryptographer could always try a brute force sequential search. This is different from an online attack, in which someone tries to use brute force to break into a IRIX account by guessing passwords. The login sequence can be used to slow down the online attack, throwing the cryptographer off the system after some number of failed attempts. Offline attacks on CSP-Kerberos passwords are analogous to stealing the */etc/password* file, and then trying to determine passwords by comparing the password field in the entry to encrypted words from a dictionary. This sort of attack occurs offline, with no system regulation.

- For further reading on CSP-Kerberos security limitations, see “Limitations of the Kerberos Authentication System” by S. M. Bellovin and M. Merritt. This paper is available through anonymous FTP at the *athena-dist.mit.edu* site. It is found in the *pub/kerberos/doc/usenix.PS* file.

Chapter 8

System Data Files

IRIX relies on a number of administrative data files to provide crucial information for the system. It is the job of the System Administrator to keep these files correct and up to date. This chapter contains a list of the new system data files added as part of the Commercial Security Pak and their formats and functions.

The outline format used in this chapter for describing each administrative data file is as follows:

Pathname: The complete pathname of the file.

Description: A complete description of the purpose of the file.

Syntax: The syntax of a record or entry in the file.

DAC Permission:
The default Discretionary Access Control (DAC) file permissions associated with the file.

Home Directory Files

The following file is present in the home directory of each user.

~/.rhosts

Pathname: ~/.rhosts

Description: This file contains a list of hosts from which this user is allowed to initiate a remote session without additional authentication.

Syntax: *command hostname username*

DAC Permission:
-rw-r--r-- (644) root,sys

/var Directory Structure Files

/var/adm/OLDsulog

Pathname: /var/adm/OLDsulog

Description: This file is used for backups of the sulog file.

Syntax: Each entry in *OLDsulog* has the following form:

```
SU 09/09 10:21 + ttyq2 invoking user-new identity
```

DAC Permission:
-rw------- (600) admin,admin

/var/adm/sulog

Pathname: /var/adm/sulog

Description: This file contains a log of all uses of the su(1) command.

Syntax: Each entry in *sulog* has the following form:

```
SU 09/09 10:21 + ttyq2 invoking user-new identity
```

DAC Permission:
-rw------- (600) admin,admin

/dev directory Structure Files

The following files reside in the special */dev* directory structure. These device files control the physical hardware.

/dev/console

Pathname: /dev/console

Description: The console provides the operator interface to the system. The operating system and system utility programs display error messages on the system console.

The console is a logical terminal represented by a text window on the graphics monitor.

The evaluated configuration does not support the option of using a serial terminal.

The device special file */dev/console* represents the system console. */dev/console* is the slave side of pseudo-tty (see pty(7)).

Syntax: Special Device File

DAC Permission:
crw--w--w- (622) root,sys

dev/klog

Pathname: /dev/klog

Description: The */dev/klog* file is the kernel error logging interface. When this device is open, messages printed by the kernel, which normally appear only in the system console window, also are buffered by the *klog* driver. The messages obtained by reading from this driver are the text of the kernel error messages.

Normally, this device is opened and read by *syslogd*(1M), the system logging daemon.

Syntax: Special device file.

DAC Permission:
crw-r--r-- (644) root,sys

/dev/kmem

Pathname: /dev/kmem

Description: */dev/kmem* is a special file that is an image of the kernel virtual memory of the computer. It may be used, for example, to examine, and even to patch the system memory.

DAC Permission:
crw-r----- (640) root,sys

/dev/log

Pathname: /dev/log

Description: This file is a named pipe that is read by *syslogd*(1m) as a source of system log messages. If a program writes error messages to */dev/log*, *syslogd* receives the messages and places them in the system log.

Syntax Named pipe.

DAC Permission:
prw-rw-rw- (666) root,sys

/dev/ptc

Pathname: /dev/ptc

Description: This file is the master pseudo-terminal.

DAC Permission:
crw-rw-rw- (666) root,sys

/dev/tty

Pathname: /dev/tty

Description: This file is, in each process, a synonym for the control terminal associated with the process group of that process, if any.

DAC Permission:
crw-rw-rw- (666) root,sys

/etc Directory Files

/etc/TIMEZONE

Pathname: /etc/TIMEZONE

Description: This file contains the time zone (for example, EST), the hours of difference between the time zone and Greenwich time zone (for example, 5), and the alternative time zone (for example, EDT). All the information is in one line without any field separators.

Syntax: TZ=<*timezone*><*hours_from_GMT*><*daylight_timezone*>

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/capability

Pathname: /etc/capability

Description: This file specifies the system-file editing permissions for each account on your system. This file contains the following information for each account:

name	User's login name – contains no uppercase characters and must not be longer than eight characters.
capabilities	The various capabilities that the user is allowed.

Syntax: The following is a sample *capability* file:

```
root:all+eip:all+eip
sysadm:all=:all=
auditor:CAP_AUDIT_WRITE,CAP_AUDIT_CONTROL,CAP_KILL+eip
dbadmin:all=:all=
ernie:all=:CAP_FOWNER,CAP_SETFCAP+eip
casey:all=:all+eip
```

DAC Permission:
-rw-r--r-- (644) dbadmin,sys

/etc/cshrc

Pathname: /etc/cshrc

Description: This file is the prototype .cshrc.

Syntax: This file contains a sample of C-shell initialization commands. It is used as the default set of commands.

DAC Permission:
-rwxr-xr-x (755) root,sys

/etc/gettydefs

Pathname: /etc/gettydefs

Description: This file contains information used by *getty*(1M) to set up the speed and terminal settings for a serial line. This file supplies information on what the *login*(1) prompt should look like. It also supplies the speed to try next if the user indicates the current speed is not correct by typing a *break* character.

Syntax: *label# initial-flags # final-flags # login-prompt #next-label*

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/group

Pathname: /etc/group

Description: This file is the definition file for user groups on the system.

Syntax: *groupname*:*epasswd*:*GID*:[*user1*,*user2*]

DAC Permission:
-rw-r--r-- (644) dbadmin,sys

Dependencies: /etc/passwd

/etc/hosts

Pathname: /etc/hosts

Description: This file contains information regarding the known hosts on the network.

Syntax: *IP-address hostname alias*[*es*]

DAC Permission:
-rw-r--r-- (644) dbadmin,sys

/etc/hosts.equiv

Pathname: /etc/hosts.equiv

Description: This file contains a list of trusted hosts. When an *rlogin*(1C), *rcp*(1C), or *rsh*(1C) request from a listed host is made, and the initiator of the request is also listed in the */etc/passwd* file, no further validity checking is done as long as the login name and user ID number of the user on the remote host are identical to the listing in the local */etc/passwd* file. If these conditions are met, *rlogin* does not prompt for a password, and *rcp* and *rsh* complete successfully. So a remote user is "equivalenced" to a local user with the same user name and user ID number when the remote user's hostname is found in *hosts.equiv.*

Syntax: *hostname*

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/ioctl.syscon

Pathname: /etc/ioctl.syscon

Description: This file defines the state of the console device. When *init* comes up at boot time, and whenever it switches out of single-user state to normal run states, it sets the *ioctl*(2) states of the virtual console, */dev/console*, to those modes saved in the file */etc/ioctl.syscon*. This file is written by *init* whenever the single-user state is entered.

Syntax: `d26:1805:8bf:3b:0:3:1c:8:18:4:0:0:0:0:0:0`

DAC Permission:
-rw-r--r-- (644) root,sys

Referenced by: init

Modified by: init

/etc/inittab

Pathname: /etc/inittab

Description: This file supplies the script to *init*'s role as a general process dispatcher. The majority of *init*'s process dispatching activity involves creating instances of the terminal line process, */etc/getty*. Other processes typically dispatched by *init* are daemons and shells.

Syntax: id:rstate:action:process

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/motd

Pathname: /etc/motd

Description: This file is used for the "Message of the Day." The System Administrator can freely edit this file. The */etc/motd* file is displayed each time a user logs in.

Syntax: ASCII text file.

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/nologin

Pathname: /etc/nologin

Description: If the file is present, remote user logins via the network are not permitted.

Syntax: There is no syntax to this file. The existence of the file is all that is required.

DAC Permission:
-r--r--r-- (0444) root,sys

Dependencies: login

Referenced by: login

/etc/opasswd

Pathname: /etc/opasswd

Description: This file is a backup copy of */etc/passwd*.

Syntax: *username*:*e_passwd*[,*Mmww* | *lock_char*]:*UID*:*GID*:*GECOS*:$*HOME*:$*SHELL*

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/passwd

Pathname: /etc/passwd

Description: This file contains information about the user. Unlike standard IRIX, the encrypted password is not stored in this file. The encrypted password is kept in */etc/shadow*. The *passwd* file contains the following information for each user:

name	User's login name contains no uppercase characters and must not be greater than eight characters long.
unused	The field that is normally occupied by the password is unused.
numerical user ID	This is the user's ID in the system and it must be unique.

numerical group ID	This is the number of the primary group to which the user belongs.
user's real name	In some versions of UNIX, this field also contains the user's office location, extension, home phone, and so on.
initial working directory	The directory that the user is in at login. This is known as the "home" directory.
shell	The program to use as the command interpreter ("shell") when the user logs in. If the shell field is empty, the Bourne shell (*/bin/sh*) is assumed.

Syntax: *username*::*UID*:*GID*:*GECOS*: $*HOME*:$*SHELL*

DAC Permission:
-rw-r--r-- (644) dbadmin,sys

/etc/profile

Pathname: /etc/profile

Description: This file is the prototype shell environment command file for use with */bin/sh*. Commands in this file are executed when the shell starts up.

Syntax: ASCII text file.

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/rhost.conf

Pathname: /etc/rhost.conf

Description: This file is the configuration file for the remote login and remote shell programs. It specifies the parameters under which remote logins and shells are allowed on your system from systems that share your security policy and those that do not. Default capability sets and allowed login labels are specified here.

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/services

Pathname: /etc/services

Description: The */etc/services* file contains information regarding the known services available in the Internet.

Syntax: Example syntax:

```
smtp 25/tcp mail
```

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/shadow

Pathname: /etc/shadow

Description: This is the user password file. This file contains the following information for each user:

name	User's login name—contains no uppercase characters and must not be longer than eight characters.
password	Encrypted password and optional password aging information.

Syntax: The following is a sample *shadow* file:

```
root:kEXFeXFTPoxE
bill:6k/7KCFRPNVXg,z/
```

DAC Permission:
-rw-r--r-- (644) dbadmin,sys

/etc/syslog.conf

Pathname: /etc/syslog.conf

Description: This file directs the system log daemon (*syslogd*) to log messages in a given set of files. Each log message in a logfile is one line. For more information about this file, see the *syslogd*(1m) reference page.

Syntax: An example *syslog.conf* file:

```
kern.debug |/usr/adm/klogpp /usr/adm/SYSLOG
kern.debug |/usr/adm/klogpp /dev/console
daemon,auth,syslog,lpr.debug /usr/adm/SYSLOG
kern.err @ginger
```

```
*.emerg *
*.alert eric,beth
*.alert;auth.warning ralph
```

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/ttytype

Pathname: /etc/ttytype

Description: This file contains a list of the tty ports on the system, and for each port, the kind of terminal that is attached to it.

Syntax: Example:

```
iris-ansi console
iris-ansi systty
vt100 ttyd1
?h19 ttyd2
?h19 ttyd3
?v50am ttyd4
?v50am ttyd5
?v50am ttyd6
?v50am ttyd7
?v50am ttyd8
?v50am ttyd9
?v50am ttyd10
?v50am ttyd11
?v50am ttyd12
```

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/utmp

Pathname: /etc/utmp

Description: This file holds user information for such commands as *who*(1), *write*(1), and *login*(1). For more information about this file, see the reference page *utmp*(4).

Syntax: Example:

```
struct utmp {
char ut_user[8]; /*User login name*/
char ut_id[4]; /*/etc/inittab id usually line #)*/
```

```
char ut_line[12]; /* device name (console,lnxx)*/
short ut_pid; /*process id*/
short ut_type; /* type of entry */
struct exit_status {
~~~~short ~~~~e_termination;/*termination status*/
~~~~short ~~~~e_exit; /* Process exit status */
}ut_exit; /*exit status of a process marked */
/* as a DEAD_PROCESS.*/
time_t ut_time; /* time entry was made */
};
```

DAC Permission:

-rw-rw-r-- (664) adm,adm

/etc/wtmp

Pathname: /etc/wtmp

Description: This file contains one record per username with related information: inittab ID; device name connected to; process ID; type of entry (for example, a login process); exit status, and time the entry was made. For more information about this file, see the reference page *utmp*(4).

Syntax: Example:

```
struct utmp {
char ut_user[8]; /* User login name */
char ut_id[4]; /*/etc/inittab id usually line #*/
char ut_line[12]; /* device name (console,lnxx) */
short ut_pid; /* process id */
short ut_type; /* type of entry */
struct exit_status {
~~~~short ~~~~e_termination;/*termination status*/
~~~~short ~~~~e_exit; /* Process exit status */
} ut_exit; /* The exit status of a process marked as
DEAD_PROCESS. */
time_t ut_time; /* time entry was made */
};
```

DAC Permission:

-rw-rw-r-- (664) adm,adm

/etc/config Directory Files

All files in the config directory that lack suffixes contain only the words “on” or “off.” This indicates whether or not the named subsystem is activated at system startup time. Files with the suffix .options contain flags to the subsystem startup command.

/etc/config/acct

Pathname: /etc/config/acct

Description: This file contains either the word “on” or “off.” If it contains “on,” process accounting is turned on by default. If it contains the word “off,” process accounting is not run by default.

Syntax: The word “on” or “off.”

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/automount

Pathname: /etc/config/automount

Description: This file is used by the system to direct NFS to automatically mount or not mount network filesystems.

Syntax: The word “on” or “off.”

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/login.options

Pathname: /etc/config/login.options

Description: This file controls the default actions of the *login* program, such as the number of unsuccessful attempts to log in or the timeout period while waiting for a password. This file is described in the *login*(4) reference page.

Syntax: Example:

```
maxtries=5
disabletime=30
passwdreq
```

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/named

Pathname: /etc/config/named

Description: This file directs the system to spawn or not to spawn the *named*(1m) domain name server.

Syntax: The word “on” or “off.”

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/network

Pathname: /etc/config/network

Description: This file is used by the system to direct NFS to spawn the lock and status daemons or not to spawn them.

Syntax: The word “on” or “off.”

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/nfs

Pathname: /etc/config/nfs

Description: This file is used by the system to start the NFS daemons and mount the network filesystems.

Syntax: The word “on” or “off.”

DAC Permission:
-rw-r--r-- (644) root,sys

Referenced by: init

/etc/config/rwhod

Pathname: /etc/config/rwhod

Description: This file directs the system to spawn or not to spawn the *rwhod*(1m) server daemon.

Syntax: The word “on” or “off.”

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/satd.options

Pathname: /etc/config/satd.options

Description: This file contains saved *satd* options. A flag to *satd* fills this file with the current *satd* options.

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/sat_select.options

Pathname: /etc/config/sat_select.options

Description: This file contains saved options to *sat_select*. A flag to *sat_select* fills this file with the current *sat_select* options.

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/syslogd.options

Pathname: /etc/config/syslogd.options

Description: This file contains command line options for the *syslogd*(1m) program. *syslogd* reads and logs messages into a set of files.

Syntax: Optional site-specific flags belong in the options file. The available flags are these:

-f Specify an alternate configuration file.

-m Select the number of minutes between mark messages.

-d Turn on debugging.

-p Use the given name for the named pipe instead of */dev/log*.

DAC Permission:
-rw-r--r-- (644) root,sys

/etc/config/timed

Pathname: /etc/config/timed

Description: This file directs the system to spawn or not to spawn the *timed*(1m) clock controlling daemon.

Syntax: The word “on” or “off.”

DAC Permission:
-rw-r--r-- (644) root,sys

/usr Directory Structure Files

/usr/adm/lastlog/*username*

Pathname: /usr/adm/lastlog/*username*

Description: These files record information for use by the *login* program about your last login.

Syntax: A typical *lastlog* file might look like this:

```
^A(:4ujohnsmith.other.place.com
```

DAC Permission:
-rwxr-xr-x (755) root,sys

/usr/adm/oSYSLOG

Pathname: /usr/adm/oSYSLOG

Description: This file is a saved old version of the system log.

Syntax: A typical *oSYSLOG* has records of this form:

```
Sep 2 01:01:38 mymachine syslogd: restart
Sep 3 15:26:12 mymachine sendmail[15324]: AA15324:
from=, size=1027, class=0
Sep 3 17:14:02 mymachine sendmail[15424]: AA15424:
from=, size=1080, class=0
Sep 3 17:44:03 mymachine sendmail[15461]: AA15461:
from=, size=974, class=0
```

DAC Permission:
-rw-r--r-- (644) root,sys

/usr/adm/SYSLOG

Pathname: /usr/adm/SYSLOG

Description: This file contains a log of all events corresponding to those selected in the */etc/syslog.conf* file.

Syntax: A typical *SYSLOG* file looks like this:

```
Sep 2 01:01:39 mymachine syslogd: restart
Sep 3 09:58:35 mymachine sendmail[21326]: AA21326:
from=, size=2266, class=0
Sep 3 10:02:32 mymachine sendmail[21336]: AA21336:
from=, size=1605, class=0
Sep 3 10:07:15 mymachine sendmail[21342]: AA21342:
from=, size=2202, class=0
```

DAC Permission:
-rw-r--r-- (644) root,sys

/usr/lib/X11/xdm/Xresources

Pathname: /usr/lib/X11/xdm/Xresources

Description: This file contains default information about your X environment.

Syntax: The default *Xresources* file looks like this:

```
xlogin*login.translations: #override
<key> F1: set-session-argument(failsafe) finish-field()
<key> Return: set-session-argument() finish-field()
xlogin*borderWidth: 3
#ifdef COLOR
xlogin*greetColor: #f63
xlogin*failColor: red
xlogin*Foreground: black
xlogin*Background: #fdc
#else
xlogin*Foreground: black
xlogin*Background: white
#endif
```

DAC Permission:
-r--r--r-- (0444) root,sys

/usr/lib/X11/xdm/Xservers

Pathname: /usr/lib/X11/xdm/Xservers

Description: This file contains the list of displays to be managed.

Syntax: Most systems have only one display, numbered 0, so the file looks like this:

```
:0 Local local /usr/bin/X11/Xsgi :0
```

DAC Permission:
-r--r--r-- (0444) root,sys

/usr/spool/lp/pstatus

Pathname: /usr/spool/lp/pstatus

Description: Printer status information is stored in this file.

Syntax: Data file.

DAC Permission:
-rw-r--r-- (644) lp,sys

/usr/spool/lp/qstatus

Pathname: /usr/spool/lp/qstatus

Description: Print queue status information is stored in this file.

Syntax: Data file.

DAC Permission:
-rw-r--r-- (644) lp,sys

Appendix A

CSP-Kerberos Files and Error Messages

The definition files and error messages listed in this chapter are shipped by default with CSP-Kerberos. The same example realm and domain names used in the text have been used in these examples.

CSP-Kerberos Files

The krb5.conf File

Normally, you should install your *krb5.conf* file in the directory */etc*. However, note that you can override this default through the environment variable *KRB5_CONFIG*.

Here is an example of a generic *krb5.conf* file:

```
[libdefaults]
   ticket_lifetime = 600
   default_realm = yoursite.com
   default_tkt_enctypes = des-cbc-crc
   default_tgs_enctypes = des-cbc-crc

[realms]
   yoursite.com = {
       kdc = KDCSERVER.yoursite.com:88
       kdc = KDCSLAVE1.yoursite.com:88
       kdc = KDCSLAVE2.yoursite.com:88
       admin_server = KDCSERVER.yoursite.com:749
       default_domain = yoursite.com
       }
   }

[domain_realm]
   .yoursite.com = yoursite.com
    yoursite.com = yoursite.com

[logging]
   kdc = FILE:/dev/ttyp9
```

```
    admin_server = FILE:/dev/ttyp9
    default = FILE:/dev/ttyp9
```

Here is an example of a more extensive *krb5.conf* file, which includes a second CSP-Kerberos realm and authentication to MIT Kerberos V4 as well as V5 KDCs in the realm *yoursite.com*:

```
[libdefaults]
    ticket_lifetime = 600
    default_realm = yoursite.com
    default_tkt_enctypes = des-cbc-crc
    default_tgs_enctypes = des-cbc-crc
    krb4_srvtab = /etc/srvtab
    krb4_config = /krb4/lib/krb.conf
    krb4_realms = /krb4/lib/krb.realms

[realms]
    yoursite.com = {
        kdc = KDCSERVER.yoursite.com:88
        kdc = KDCSLAVE1.yoursite.com:88
        kdc = KDCSLAVE2.yoursite.com:88
        admin_server = KDCSERVER.yoursite.com:749
        default_domain = yoursite.com
        v4_instance_convert = {
        bleep = yoursite.com
         }
     }
    theirsite.com = {
        kdc = KDCSERVER.theirsite.com
        kdc = KDCSLAVE1.theirsite.com
        admin_server = KDCSERVER.theirsite.com
        }

[domain_realm]
    .yoursite.com = yoursite.com
    yoursite.com = yoursite.com
    .theirsite.com = theirsite.com
    theirsite.com = theirsite.com
```

For the KDCs, add a section onto the end of the *krb5.conf* file specifying the locations of the *kdc.conf* file, as in the following example:

```
[kdc]
    profile = ROOTDIR/etc/kdc.conf

[logging]
    admin_server = FILE:ROOTDIR/krb5/lib/krb5kdc/kadmind.log
```

```
   kdc = FILE:ROOTDIR/krb5/lib/krb5kdc/kdc.log
   default = CONSOLE
```

The kdc.conf File

Normally, you should install your *kdc.conf* file in the directory */krb5/lib/krb5kdc*. However, note that you can override this default by a pointer in the KDC's *krb5.conf* file, or through the environment variable *KRB5_KDC_PROFILE*.

Here's an example of a *kdc.conf* file:

```
[kdcdefaults]
   kdc_ports = 88,750

[realms]
   yoursite.com = {
      profile = /etc/krb5.conf
      database_name = krb5/lib/krb5kdc/principal
      admin_database_name = /krb5/lib/krb5kdc/principal.kadm5
      admin_database_lockfile = /krb5/lib/krb5kdc/principal.kadm5.lock
      admin_keytab = /krb5/lib/krb5kdc/kadm5.keytab
      acl_file = /krb5/lib/krb5kdc/kadm5.acl
      dict_file = /krb5/lib/krb5kdc/kadm5.dict
      key_stash_file = /krb5/lib/krb5kdc/.k5.yoursite.com
      kadmind_port = 749
      max_life = 10h 0m 0s
      max_renewable_life = 7d 0h 0m 0s
      master_key_type = des-cbc-crc
      supported_enctypes = des-cbc-crc:normal
      }
```

To add MIT Kerberos V4 support, change the supported_enctypes line to:

```
supported_enctypes = des-cbc-crc:normal des-cbc-crc:v4
```

CSP-Kerberos Error Messages

V5 Library Error Codes

This is the CSP-Kerberos v5 library error code list. Protocol error codes are as follows:

ERROR_TABLE_BASE_krb5 + the protocol error code number

Other error codes start at: ERROR_TABLE_BASE_krb5 + 128.

KRB5KDC_ERR_NONE
: No error

KRB5KDC_ERR_NAME_EXP
: Client's entry in database has expired

KRB5KDC_ERR_SERVICE_EXP
: Server's entry in database has expired

KRB5KDC_ERR_BAD_PVNO
: Requested protocol version not supported

KRB5KDC_ERR_C_OLD_MAST_KVNO
: Client's key is encrypted in an old master key

KRB5KDC_ERR_S_OLD_MAST_KVNO
: Server's key is encrypted in an old master key

KRB5KDC_ERR_C_PRINCIPAL_UNKNOWN
: Client not found in CSP-Kerberos database

KRB5KDC_ERR_S_PRINCIPAL_UNKNOWN
: Server not found in CSP-Kerberos database

KRB5KDC_ERR_PRINCIPAL_NOT_UNIQUE
: Principal has multiple entries in CSP-Kerberos database

KRB5KDC_ERR_NULL_KEY
: Client or server has a null key

KRB5KDC_ERR_CANNOT_POSTDATE
: Ticket is ineligible for postdating

KRB5KDC_ERR_NEVER_VALID
: Requested effective lifetime is negative or too short

KRB5KDC_ERR_POLICY
: KDC policy rejects request

KRB5KDC_ERR_BADOPTION
: KDC can't fulfill requested option

KRB5KDC_ERR_ETYPE_NOSUPP
: KDC has no support for encryption type

KRB5KDC_ERR_SUMTYPE_NOSUPP
: KDC has no support for checksum type

KRB5KDC_ERR_PADATA_TYPE_NOSUPP
: KDC has no support for padata type

KRB5KDC_ERR_TRTYPE_NOSUPP
: KDC has no support for transited type

KRB5KDC_ERR_CLIENT_REVOKED
: Clients credentials have been revoked

KRB5KDC_ERR_SERVICE_REVOKED
: Credentials for server have been revoked

KRB5KDC_ERR_TGT_REVOKED
: TGT has been revoked

KRB5KDC_ERR_CLIENT_NOTYET
: Client not yet valid - try again later

KRB5KDC_ERR_SERVICE_NOTYET
: Server not yet valid - try again later

KRB5KDC_ERR_KEY_EXP
: Password has expired

KRB5KDC_ERR_PREAUTH_FAILED
: Preauthentication failed

KRB5KDC_ERR_PREAUTH_REQUIRED
: Additional pre-authentication required

KRB5KDC_ERR_PREAUTH_REQUIRED
: Additional preauthentication required

KRB5KDC_ERR_SERVER_NOMATCH
: Requested server and ticket don't match

KRB5PLACEHOLD_27
: KRB5 error code 27

KRB5PLACEHOLD_28
: KRB5 error code 28

KRB5PLACEHOLD_29
: KRB5 error code 29

KRB5PLACEHOLD_30
: KRB5 error code 30

KRB5KRB_AP_ERR_BAD_INTEGRITY
: Decrypt integrity check failed

KRB5KRB_AP_ERR_TKT_EXPIRED
: Ticket expired

KRB5KRB_AP_ERR_TKT_NYV
: Ticket not yet valid

KRB5KRB_AP_ERR_REPEAT
: Request is a replay

KRB5KRB_AP_ERR_NOT_US
: The ticket isn't for us

KRB5KRB_AP_ERR_BADMATCH
: Ticket/authenticator don't match

KRB5KRB_AP_ERR_SKEW
: Clock skew too great

KRB5KRB_AP_ERR_BADADDR
: Incorrect net address

KRB5KRB_AP_ERR_BADVERSION
: Protocol version mismatch

KRB5KRB_AP_ERR_MSG_TYPE
: Invalid message type

KRB5KRB_AP_ERR_MODIFIED
: Message stream modified

KRB5KRB_AP_ERR_BADORDER
: Message out of order

KRB5KRB_AP_ERR_ILL_CR_TKT
: Illegal cross-realm ticket

KRB5KRB_AP_ERR_BADKEYVER
: Key version is not available

KRB5KRB_AP_ERR_NOKEY
: Service key not available

KRB5KRB_AP_ERR_MUT_FAIL
: Mutual authentication failed

KRB5KRB_AP_ERR_BADDIRECTION
: Incorrect message direction

KRB5KRB_AP_ERR_METHOD
: Alternative authentication method required

KRB5KRB_AP_ERR_BADSEQ
: Incorrect sequence number in message

KRB5KRB_AP_ERR_INAPP_CKSUM
: Inappropriate type of checksum in message

KRB5PLACEHOLD_51
: KRB5 error code 51

KRB5PLACEHOLD_52
: KRB5 error code 52

KRB5PLACEHOLD_53
: KRB5 error code 53

KRB5PLACEHOLD_54
: KRB5 error code 54

KRB5PLACEHOLD_55
: KRB5 error code 55

KRB5PLACEHOLD_56
: KRB5 error code 56

KRB5PLACEHOLD_57
: KRB5 error code 57

KRB5PLACEHOLD_58
: KRB5 error code 58

KRB5PLACEHOLD_59
: KRB5 error code 59

KRB5KRB_ERR_GENERIC
: Generic error

KRB5KRB_ERR_FIELD_TOOLONG
: Field is too long for this implementation

KRB5PLACEHOLD_62
: KRB5 error code 62

KRB5PLACEHOLD_63
: KRB5 error code 63

KRB5PLACEHOLD_64
: KRB5 error code 64

KRB5PLACEHOLD_65
: KRB5 error code 65

KRB5PLACEHOLD_66
: KRB5 error code 66

KRB5PLACEHOLD_67
: KRB5 error code 67

KRB5PLACEHOLD_68
: KRB5 error code 68

KRB5PLACEHOLD_69
: KRB5 error code 69

KRB5PLACEHOLD_70
: KRB5 error code 70

KRB5PLACEHOLD_71
: KRB5 error code 71

KRB5PLACEHOLD_72
: KRB5 error code 72

KRB5PLACEHOLD_73
: KRB5 error code 73

KRB5PLACEHOLD_74
: KRB5 error code 74

KRB5PLACEHOLD_75
: KRB5 error code 75

KRB5PLACEHOLD_76
: KRB5 error code 76

KRB5PLACEHOLD_77
: KRB5 error code 77

KRB5PLACEHOLD_78
: KRB5 error code 78

KRB5PLACEHOLD_79
: KRB5 error code 79

KRB5PLACEHOLD_80
: KRB5 error code 80

KRB5PLACEHOLD_81
: KRB5 error code 81

KRB5PLACEHOLD_82
: KRB5 error code 82

KRB5PLACEHOLD_83
: KRB5 error code 83

KRB5PLACEHOLD_84
: KRB5 error code 84

KRB5PLACEHOLD_85
: KRB5 error code 85

KRB5PLACEHOLD_86
: KRB5 error code 86

KRB5PLACEHOLD_87
: KRB5 error code 87

KRB5PLACEHOLD_88
: KRB5 error code 88

KRB5PLACEHOLD_89
: KRB5 error code 89

KRB5PLACEHOLD_90
: KRB5 error code 90

KRB5PLACEHOLD_91
: KRB5 error code 91

KRB5PLACEHOLD_92
: KRB5 error code 92

KRB5PLACEHOLD_93
: KRB5 error code 93

KRB5PLACEHOLD_94
: KRB5 error code 94

KRB5PLACEHOLD_95
: KRB5 error code 95

KRB5PLACEHOLD_96
: KRB5 error code 96

KRB5PLACEHOLD_97
: KRB5 error code 97

KRB5PLACEHOLD_98
: KRB5 error code 98

KRB5PLACEHOLD_99
: KRB5 error code 99

KRB5PLACEHOLD_100
: KRB5 error code 100

KRB5PLACEHOLD_101
: KRB5 error code 101

KRB5PLACEHOLD_102
: KRB5 error code 102

KRB5PLACEHOLD_103
: KRB5 error code 103

KRB5PLACEHOLD_104
: KRB5 error code 104

KRB5PLACEHOLD_105
: KRB5 error code 105

KRB5PLACEHOLD_106
: KRB5 error code 106

KRB5PLACEHOLD_107
: KRB5 error code 107

KRB5PLACEHOLD_108
: KRB5 error code 108

KRB5PLACEHOLD_109
: KRB5 error code 109

KRB5PLACEHOLD_110
: KRB5 error code 110

KRB5PLACEHOLD_111
: KRB5 error code 111

KRB5PLACEHOLD_112
: KRB5 error code 112

KRB5PLACEHOLD_113
: KRB5 error code 113

KRB5PLACEHOLD_114
: KRB5 error code 114

KRB5PLACEHOLD_115
: KRB5 error code 115

KRB5PLACEHOLD_116
: KRB5 error code 116

KRB5PLACEHOLD_117
: KRB5 error code 117

KRB5PLACEHOLD_118
: KRB5 error code 118

KRB5PLACEHOLD_119
: KRB5 error code 119

KRB5PLACEHOLD_120
: KRB5 error code 120

KRB5PLACEHOLD_121
: KRB5 error code 121

KRB5PLACEHOLD_122
: KRB5 error code 122

KRB5PLACEHOLD_123
: KRB5 error code 123

KRB5PLACEHOLD_124
: KRB5 error code 124

KRB5PLACEHOLD_125
: KRB5 error code 125

KRB5PLACEHOLD_126
: KRB5 error code 126

KRB5_ERR_RCSID
: $Id: admin.texinfo,v 1.7 1996/09/09 18:29:25 jcb Exp $

KRB5_LIBOS_BADLOCKFLAG
: Invalid flag for file lock mode

KRB5_LIBOS_CANTREADPWD
: Cannot read password

KRB5_LIBOS_BADPWDMATCH
: Password mismatch

KRB5_LIBOS_PWDINTR
: Password read interrupted

KRB5_PARSE_ILLCHAR
: Illegal character in component name

KRB5_PARSE_MALFORMED
: Malformed representation of principal

KRB5_CONFIG_CANTOPEN
: Can't open/find configuration file

KRB5_CONFIG_BADFORMAT
: Improper format of configuration file

KRB5_CONFIG_NOTENUFSPACE
: Insufficient space to return complete information

KRB5_BADMSGTYPE
: Invalid message type specified for encoding

KRB5_CC_BADNAME
: Credential cache name malformed

KRB5_CC_UNKNOWN_TYPE
: Unknown credential cache type

KRB5_CC_NOTFOUND
: Matching credential not found

KRB5_CC_END
: End of credential cache reached

KRB5_NO_TKT_SUPPLIED
: Request did not supply a ticket

KRB5KRB_AP_WRONG_PRINC
: Wrong principal in request

KRB5KRB_AP_ERR_TKT_INVALID
: Ticket has invalid flag set

KRB5_PRINC_NOMATCH
: Requested principal and ticket don't match

KRB5_KDCREP_MODIFIED
: KDC reply did not match expectations

KRB5_KDCREP_SKEW
: Clock skew too great in KDC reply

KRB5_IN_TKT_REALM_MISMATCH
: Client/server realm mismatch in initial ticket request

KRB5_PROG_ETYPE_NOSUPP
: Program lacks support for encryption type

KRB5_PROG_KEYTYPE_NOSUPP
: Program lacks support for key type

KRB5_WRONG_ETYPE
: Requested encryption type not used in message

KRB5_PROG_SUMTYPE_NOSUPP
: Program lacks support for checksum type

KRB5_REALM_UNKNOWN
: Cannot find KDC for requested realm

KRB5_SERVICE_UNKNOWN
: CSP-Kerberos service unknown

KRB5_KDC_UNREACH
: Cannot contact any KDC for requested realm

KRB5_NO_LOCALNAME
: No local name found for principal name

KRB5_MUTUAL_FAILED
: Mutual authentication failed

KRB5_RC_TYPE_EXISTS
: Replay cache type is already registered

KRB5_RC_MALLOC
: No more memory to allocate (in replay cache code)

KRB5_RC_TYPE_NOTFOUND
: Replay cache type is unknown

KRB5_RC_UNKNOWN
: Generic unknown RC error

KRB5_RC_REPLAY
: Message is a replay

KRB5_RC_IO
: Replay I/O operation failed

KRB5_RC_NOIO
: Replay cache type does not support non-volatile storage

KRB5_RC_PARSE
: Replay cache name parse/format error

KRB5_RC_IO_EOF
: End-of-file on replay cache I/O

KRB5_RC_IO_MALLOC
: No more memory to allocate (in replay cache I/O code)

KRB5_RC_IO_PERM
: Permission denied in replay cache code

KRB5_RC_IO_IO
: I/O error in replay cache i/o code

KRB5_RC_IO_UNKNOWN
: Generic unknown RC/IO error

KRB5_RC_IO_SPACE
: Insufficient system space to store replay information

KRB5_TRANS_CANTOPEN
: Can't open/find realm translation file

KRB5_TRANS_BADFORMAT
: Improper format of realm translation file

KRB5_LNAME_CANTOPEN
: Can't open/find lname translation database

KRB5_LNAME_NOTRANS
: No translation available for requested principal

KRB5_LNAME_BADFORMAT
: Improper format of translation database entry

KRB5_CRYPTO_INTERNAL
: Cryptosystem internal error

KRB5_KT_BADNAME
: Key table name malformed

KRB5_KT_UNKNOWN_TYPE
: Unknown Key table type

KRB5_KT_NOTFOUND
: Key table entry not found

KRB5_KT_END
: End of key table reached

KRB5_KT_NOWRITE
: Cannot write to specified key table

KRB5_KT_IOERR
: Error writing to key table

KRB5_NO_TKT_IN_RLM
: Cannot find ticket for requested realm

KRB5DES_BAD_KEYPAR
: DES key has bad parity

KRB5DES_WEAK_KEY
: DES key is a weak key

KRB5_BAD_ENCTYPE
: Bad encryption type

KRB5_BAD_KEYSIZE
: Key size is incompatible with encryption type

KRB5_BAD_MSIZE
: Message size is incompatible with encryption type

KRB5_CC_TYPE_EXISTS
: Credentials cache type is already registered.

KRB5_KT_TYPE_EXISTS
: Key table type is already registered.

KRB5_CC_IO
: Credentials cache I/O operation failed XXX

KRB5_FCC_PERM
: Credentials cache file permissions incorrect

KRB5_FCC_NOFILE
: No credentials cache file found

KRB5_FCC_INTERNAL
: Internal file credentials cache error

KRB5_CC_WRITE
: Error writing to credentials cache file

KRB5_CC_NOMEM
: No more memory to allocate (in credentials cache code)

KRB5_CC_FORMAT
: Bad format in credentials cache

KRB5_INVALID_FLAGS
: Invalid KDC option combination (library internal error) [for dual tgt library calls]

KRB5_NO_2ND_TKT
: Request missing second ticket [for dual tgt library calls]

KRB5_NOCREDS_SUPPLIED
No credentials supplied to library routine

KRB5_SENDAUTH_BADAUTHVERS
Bad sendauth version was sent

KRB5_SENDAUTH_BADAPPLVERS
Bad application version was sent (via sendauth)

KRB5_SENDAUTH_BADRESPONSE
Bad response (during sendauth exchange)

KRB5_SENDAUTH_REJECTED
Server rejected authentication (during sendauth exchange)

KRB5_PREAUTH_BAD_TYPE
Unsupported preauthentication type

KRB5_PREAUTH_NO_KEY
Required preauthentication key not supplied

KRB5_PREAUTH_FAILED
Generic preauthentication failure

KRB5_RCACHE_BADVNO
Unsupported replay cache format version number

KRB5_CCACHE_BADVNO
Unsupported credentials cache format version number

KRB5_KEYTAB_BADVNO
Unsupported key table format version number

KRB5_PROG_ATYPE_NOSUPP
Program lacks support for address type

KRB5_RC_REQUIRED
Message replay detection requires rcache parameter

KRB5_ERR_BAD_HOSTNAME
Hostname cannot be canonicalized

KRB5_ERR_HOST_REALM_UNKNOWN
Cannot determine realm for host

KRB5_SNAME_UNSUPP_NAMETYPE
Conversion to service principal undefined for name type

KRB5KRB_AP_ERR_V4_REPLY
: Initial Ticket response appears to be Version 4 error

KRB5_REALM_CANT_RESOLVE
: Cannot resolve KDC for requested realm

KRB5_TKT_NOT_FORWARDABLE
: Requesting ticket can't get forwardable tickets

KRB5_FWD_BAD_PRINCIPAL
: Bad principal name while trying to forward credentials

KRB5_GET_IN_TKT_LOOP
: Looping detected inside krb5_get_in_tkt

KRB5_CONFIG_NODEFREALM
: Configuration file does not specify default realm

KRB5_SAM_UNSUPPORTED
: Bad SAM flags in obtain_sam_padata

CSP-Kerberos V5 Database Library Error Codes

This is the CSP-Kerberos v5 database library error code table.

KRB5_KDB_RCSID
: $Id: admin.texinfo,v 1.7 1996/09/09 18:29:25 jcb Exp $

KRB5_KDB_INUSE
: Entry already exists in database

KRB5_KDB_UK_SERROR
: Database store error

KRB5_KDB_UK_RERROR
: Database read error

KRB5_KDB_UNAUTH
: Insufficient access to perform requested operation

KRB5_KDB_NOENTRY
: No such entry in the database

KRB5_KDB_ILL_WILDCARD
: Illegal use of wildcard

KRB5_KDB_DB_INUSE
Database is locked or in use--try again later

KRB5_KDB_DB_CHANGED
Database was modified during read

KRB5_KDB_TRUNCATED_RECORD
Database record is incomplete or corrupted

KRB5_KDB_RECURSIVELOCK
Attempt to lock database twice

KRB5_KDB_NOTLOCKED
Attempt to unlock database when not locked

KRB5_KDB_BADLOCKMODE
Invalid kdb lock mode

KRB5_KDB_DBNOTINITED
Database has not been initialized

KRB5_KDB_DBINITED
Database has already been initialized

KRB5_KDB_ILLDIRECTION
Bad direction for converting keys

KRB5_KDB_NOMASTERKEY
Cannot find master key record in database

KRB5_KDB_BADMASTERKEY
Master key does not match database

KRB5_KDB_INVALIDKEYSIZE
Key size in database is invalid

KRB5_KDB_CANTREAD_STORED
Cannot find/read stored master key

KRB5_KDB_BADSTORED_MKEY
Stored master key is corrupted

KRB5_KDB_CANTLOCK_DB
Insufficient access to lock database

KRB5_KDB_DB_CORRUPT
Database format error

KRB5_KDB_BAD_VERSION
: Unsupported version in database entry

KRB5_KDB_BAD_SALTTYPE
: Unsupported salt type

KRB5_KDB_BAD_ENCTYPE
: Unsupported encryption type

CSP-Kerberos V5 Magic Numbers Error Codes

This is the CSP-Kerberos v5 magic numbers error code table.

KV5M_NONE
: CSP-Kerberos V5 magic number table

KV5M_PRINCIPAL
: Bad magic number for krb5_principal structure

KV5M_DATA Bad magic number for krb5_data structure

KV5M_KEYBLOCK
: Bad magic number for krb5_keyblock structure

KV5M_CHECKSUM
: Bad magic number for krb5_checksum structure

KV5M_ENCRYPT_BLOCK
: Bad magic number for krb5_encrypt_block structure

KV5M_ENC_DATA
: Bad magic number for krb5_enc_data structure

KV5M_CRYPTOSYSTEM_ENTRY
: Bad magic number for krb5_cryptosystem_entry structure

KV5M_CRYPTOSYSTEM_ENTRY
: Bad magic number for krb5_cryptosystem_entry structure

KV5M_CS_TABLE_ENTRY
: Bad magic number for krb5_cs_table_entry structure

KV5M_CHECKSUM_ENTRY
: Bad magic number for krb5_checksum_entry structure

KV5M_CHECKSUM_ENTRY
: Bad magic number for krb5_checksum_entry structure

KV5M_AUTHDATA
: Bad magic number for krb5_authdata structure

KV5M_TRANSITED
: Bad magic number for krb5_transited structure

KV5M_ENC_TKT_PART
: Bad magic number for krb5_enc_tkt_part structure

KV5M_TICKET
: Bad magic number for krb5_ticket structure

KV5M_AUTHENTICATOR
: Bad magic number for krb5_authenticator structure

KV5M_TKT_AUTHENT
: Bad magic number for krb5_tkt_authent structure

KV5M_CREDS
: Bad magic number for krb5_creds structure

KV5M_LAST_REQ_ENTRY
: Bad magic number for krb5_last_req_entry structure

KV5M_PA_DATA
: Bad magic number for krb5_pa_data structure

KV5M_KDC_REQ
: Bad magic number for krb5_kdc_req structure

KV5M_ENC_KDC_REP_PART
: Bad magic number for krb5_enc_kdc_rep_part structure

KV5M_KDC_REP
: Bad magic number for krb5_kdc_rep structure

KV5M_ERROR
: Bad magic number for krb5_error structure

KV5M_AP_REQ
: Bad magic number for krb5_ap_req structure

KV5M_AP_REP
: Bad magic number for krb5_ap_rep structure

KV5M_AP_REP_ENC_PART
: Bad magic number for krb5_ap_rep_enc_part structure

KV5M_RESPONSE
: Bad magic number for krb5_response structure

KV5M_SAFE
: Bad magic number for krb5_safe structure

KV5M_PRIV
: Bad magic number for krb5_priv structure

KV5M_PRIV_ENC_PART
: Bad magic number for krb5_priv_enc_part structure

KV5M_CRED
: Bad magic number for krb5_cred structure

KV5M_CRED_INFO
: Bad magic number for krb5_cred_info structure

KV5M_CRED_ENC_PART
: Bad magic number for krb5_cred_enc_part structure

KV5M_PWD_DATA
: Bad magic number for krb5_pwd_data structure

KV5M_ADDRESS
: Bad magic number for krb5_address structure

KV5M_KEYTAB_ENTRY
: Bad magic number for krb5_keytab_entry structure

KV5M_CONTEXT
: Bad magic number for krb5_context structure

KV5M_OS_CONTEXT
: Bad magic number for krb5_os_context structure

KV5M_ALT_METHOD
: Bad magic number for krb5_alt_method structure

KV5M_ETYPE_INFO_ENTRY
: Bad magic number for krb5_etype_info_entry structure

KV5M_DB_CONTEXT
: Bad magic number for krb5_db_context structure

KV5M_AUTH_CONTEXT
: Bad magic number for krb5_auth_context structure

KV5M_KEYTAB
: Bad magic number for krb5_keytab structure

KV5M_RCACHE
: Bad magic number for krb5_rcache structure

KV5M_CCACHE
: Bad magic number for krb5_ccache structure

KV5M_PREAUTH_OPS
: Bad magic number for krb5_preauth_ops

KV5M_SAM_CHALLENGE
: Bad magic number for krb5_sam_challenge

KV5M_SAM_KEY
: Bad magic number for krb5_sam_key

KV5M_ENC_SAM_RESPONSE_ENC
: Bad magic number for krb5_enc_sam_response_enc

KV5M_SAM_RESPONSE
: Bad magic number for krb5_sam_response

KV5M_PREDICTED_SAM_RESPONSE
: Bad magic number for krb5_predicted_sam_response

KV5M_PASSWD_PHRASE_ELEMENT
: Bad magic number for passwd_phrase_element

ASN.1 Error Codes

ASN1_BAD_TIMEFORMAT
: ASN.1 failed call to system time library

ASN1_MISSING_FIELD
: ASN.1 structure is missing a required field

ASN1_MISPLACED_FIELD
: ASN.1 unexpected field number

ASN1_TYPE_MISMATCH
: ASN.1 type numbers are inconsistent

ASN1_OVERFLOW
: ASN.1 value too large

ASN1_OVERRUN
: ASN.1 encoding ended unexpectedly

ASN1_BAD_ID
: ASN.1 identifier doesn't match expected value

ASN1_BAD_LENGTH
: ASN.1 length doesn't match expected value

ASN1_BAD_FORMAT
: ASN.1 badly-formatted encoding

ASN1_PARSE_ERROR
: ASN.1 parse error

GSSAPI Error Codes

G_BAD_SERVICE_NAME
: No @ in SERVICE-NAME name string

G_BAD_STRING_UID
: STRING-UID-NAME contains nondigits

G_NOUSER UID does not resolve to username

G_VALIDATE_FAILED
: Validation error

G_BUFFER_ALLOC
: Couldn't allocate gss_buffer_t data

G_BAD_MSG_CTX
: Message context invalid

G_WRONG_SIZE
: Buffer is the wrong size

G_BAD_USAGE
: Credential usage type is unknown

G_UNKNOWN_QOP
: Unknown quality of protection specified

G_BAD_HOSTNAME
: Hostname in SERVICE-NAME string could not be canonicalized

KG_CCACHE_NOMATCH
: Principal in credential cache does not match desired name

KG_KEYTAB_NOMATCH
: No principal in keytab matches desired name

KG_TGT_MISSING
: Credential cache has no TGT

KG_NO_SUBKEY
: Authenticator has no subkey

KG_CONTEXT_ESTABLISHED
: Context is already fully established

KG_BAD_SIGN_TYPE
: Unknown signature type in token

KG_BAD_LENGTH
: Invalid field length in token

KG_CTX_INCOMPLETE
: Attempt to use incomplete security context

KG_CONTEXT
: Bad magic number for krb5_gss_ctx_id_t

KG_CRED
: Bad magic number for krb5_gss_cred_id_t

KG_ENC_DESC
: Bad magic number for krb5_gss_enc_desc

kadmin Time Zones

This is a complete listing of the time zones recognized by the *kadmin* command:

gmt	Greenwich Mean Time
ut, utc	Universal Time (Coordinated).
wet	Western European Time. (Same as GMT.)
bst	British Summer Time. (1 hour ahead of GMT.)
wat	West Africa Time. (1 hour behind GMT.)
at	Azores Time. (2 hours behind GMT.)
bst	Brazil Standard Time. (3 hours behind GMT.) Note that the acronym BST also stands for British Summer Time.

gst	Greenland Standard Time. (3 hours behind GMT.) Note that the acronym GST also stands for Guam Standard Time.
nft	Newfoundland Time. (3.5 hours behind GMT.)
nst	Newfoundland Standard Time. (3.5 hours behind GMT.)
ndt	Newfoundland Daylight Time. (2.5 hours behind GMT.)
ast	Atlantic Standard Time. (4 hours behind GMT.)
adt	Atlantic Daylight Time. (3 hours behind GMT.)
est	Eastern Standard Time. (5 hours behind GMT.)
edt	Eastern Daylight Time. (4 hours behind GMT.)
cst	Central Standard Time. (6 hours behind GMT.)
cdt	Central Daylight Time. (5 hours behind GMT.)
mst	Mountain Standard Time. (7 hours behind GMT.)
mdt	Mountain Daylight Time. (6 hours behind GMT.)
pst	Pacific Standard Time. (8 hours behind GMT.)
pdt	Pacific Daylight Time. (7 hours behind GMT.)
yst	Yukon Standard Time. (9 hours behind GMT.)
ydt	Yukon Daylight Time. (8 hours behind GMT.)
hst	Hawaii Standard Time. (10 hours behind GMT.)
hdt	Hawaii Daylight Time. (9 hours behind GMT.)
cat	Central Alaska Time. (10 hours behind GMT.)
ahst	Alaska-Hawaii Standard Time. (10 hours behind GMT.)
nt	Nome Time. (11 hours behind GMT.)
idlw	International Date Line West Time. (12 hours behind GMT.)
cet	Central European Time. (1 hour ahead of GMT.)
met	Middle European Time. (1 hour ahead of GMT.)
mewt	Middle European Winter Time. (1 hour ahead of GMT.)
mest	Middle European Summer Time. (2 hours ahead of GMT.)
swt	Swedish Winter Time. (1 hour ahead of GMT.)

sst	Swedish Summer Time. (1 hours ahead of GMT.)
fwt	French Winter Time. (1 hour ahead of GMT.)
fst	French Summer Time. (2 hours ahead of GMT.)
eet	Eastern Europe Time; Russia Zone 1. (2 hours ahead of GMT.)
bt	Baghdad Time; Russia Zone 2. (3 hours ahead of GMT.)
it	Iran Time. (3.5 hours ahead of GMT.)
zp4	Russia Zone 3. (4 hours ahead of GMT.)
zp5	Russia Zone 4. (5 hours ahead of GMT.)
ist	Indian Standard Time. (5.5 hours ahead of GMT.)
zp6	Russia Zone 5. (6 hours ahead of GMT.)
nst	North Sumatra Time. (6.5 hours ahead of GMT.) Note that the acronym NST is also used for Newfoundland Standard Time.
sst	South Sumatra Time; Russia Zone 6. (7 hours ahead of GMT.) Note that SST is also Swedish Summer Time.
wast	West Australian Standard Time. (7 hours ahead of GMT.)
wadt	West Australian Daylight Time. (8 hours ahead of GMT.)
jt	Java Time. (7.5 hours ahead of GMT.)
cct	China Coast Time; Russia Zone 7. (8 hours ahead of GMT.)
jst	Japan Standard time; Russia Zone 8. (9 hours ahead of GMT.)
kst	Korean Standard Time. (9 hours ahead of GMT.)
cast	Central Australian Standard Time. (9.5 hours ahead of GMT.)
cadt	Central Australian Daylight Time. (10.5 hours ahead of GMT.)
east	Eastern Australian Standard Time. (10 hours ahead of GMT.)
eadt	Eastern Australian Daylight Time. (11 hours ahead of GMT.)
gst	Guam Standard Time; Russia Zone 9. (10 hours ahead of GMT.)
kdt	Korean Daylight Time. (10 hours ahead of GMT.)
nzt	New Zealand Time. (12 hours ahead of GMT.)
nzst	New Zealand Standard Time. (12 hours ahead of GMT.)

nzdt	New Zealand Daylight Time. (13 hours ahead of GMT.)
idle	International Date Line East. (12 hours ahead of GMT.)

Index

D

E

F

G

H

I

K

L

M

T

U

V

Y

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 007-3266-001.

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