

Tru64 UNIX

Security Administration

Part Number: AA-RH95E-TE

September 2002

Product Version: Tru64 UNIX Version 5.1B or higher

This manual describes Tru64 UNIX security concepts and administration.

© 2002 Hewlett-Packard Company

CyberSafe is a registered trademark, and ActiveTRUST and TrustBroker are trademarks of CyberSafe Corporation. Microsoft® and Windows NT® are trademarks of Microsoft Corporation in the U.S. and/or other countries. The Open Group™ and UNIX® are trademarks of The Open Group in the U.S. and/or other countries.

Contains SSH Secure Shell technology, SSH is a registered trademark of SSH Communications Security Corp. (<http://www.ssh.com>)

All other product names mentioned herein may be the trademarks of their respective companies.

Confidential computer software. Valid license from Compaq Computer Corporation, a wholly owned subsidiary of Hewlett-Packard Company, required for possession, use, or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

None of Compaq, HP, or any of their subsidiaries shall be liable for technical or editorial errors or omissions contained herein. The information is provided "as is" without warranty of any kind and is subject to change without notice. The warranties for HP or Compaq products are set forth in the express limited warranty statements accompanying such products. Nothing herein should be construed as constituting an additional warranty.

Contents

About This Manual

1 Identification, Authentication, and Authorization

1.1	Authentication Overview	1-1
1.2	Authentication Implementations	1-4
1.2.1	Password Authentication	1-4
1.2.2	Host-Based Authentication	1-4
1.2.3	Public Key Authentication	1-5
1.2.3.1	Digital Signatures and Certificates	1-6
1.2.4	Secret Key Authentication	1-7
1.2.5	Encryption Methods	1-9
1.3	Local Authentication Methods	1-9
1.3.1	Base (BSD) Mechanism	1-10
1.3.2	Enhanced Security Mechanism	1-10
1.3.2.1	Login Control Enhancements	1-11
1.3.2.2	Password Enhancements	1-11
1.4	Remote Authentication Methods	1-12
1.4.1	NIS Protocol	1-12
1.4.2	LDAP Mechanism	1-12
1.4.3	Secure Shell Application	1-13
1.4.4	SSO/Kerberos Mechanism	1-13
1.4.5	Advanced Server for UNIX (ASU)	1-14
1.4.6	IPsec Protocol	1-14
1.4.7	SSL, CDSA, and GSS APIs	1-14
1.5	Security Integration Architecture Overview	1-15
1.5.1	SIA and User Change Routines	1-17
1.5.2	Configuring SIA	1-18
1.5.2.1	SIA Mechanism Initialization	1-20
1.5.2.2	Adding an SIA Mechanism	1-20
1.5.2.3	Removing an SIA Mechanism	1-21
1.5.3	SIA Logging	1-21

2 Securing System Resources

2.1	Access Control Overview	2-1
2.2	Tru64 UNIX Permissions	2-1

2.2.1	Displaying Tru64 UNIX File and Directory Permissions .	2-2
2.2.2	Setting File and Directory Permissions (chmod)	2-3
2.2.2.1	Specifying Permissions with Letters and Operation Symbols	2-4
2.2.2.1.1	Changing File Permissions	2-5
2.2.2.1.2	Changing Directory Permissions	2-5
2.2.2.1.3	Using Pattern-Matching Characters	2-6
2.2.2.1.4	Setting Absolute Permissions	2-6
2.2.2.2	Specifying Permissions with Octal Numbers	2-7
2.2.3	Setting Default Permissions	2-8
2.2.3.1	Setting the User Mask	2-11
2.3	Access Control Lists (ACLs)	2-12
2.3.1	Enabling and Disabling ACLs	2-13
2.3.1.1	Enabling ACLs on NFS	2-14
2.3.2	ACL Structure	2-14
2.3.3	Access Checking with ACLs	2-16
2.3.4	ACL Inheritance	2-17
2.3.4.1	ACL Inheritance Examples	2-19
2.3.5	Managing ACLs	2-21
2.3.5.1	Using the dxsetacl Interface	2-21
2.3.5.2	Using the getacl Command	2-21
2.3.5.3	Using the setacl Command	2-22
2.3.6	ACL Interaction with Commands and Applications	2-23
2.3.6.1	The pax and tar Commands	2-23
2.3.6.2	Archiving Commands	2-24

3 Auditing the System

3.1	Auditing Overview	3-1
3.1.1	Audit Files	3-3
3.1.2	Audit Tools	3-5
3.1.2.1	Command-Line Interface	3-5
3.1.2.2	Graphical Interface	3-6
3.1.3	Audit Masks	3-6
3.1.3.1	System Calls	3-6
3.1.3.2	Trusted Events	3-8
3.1.3.3	Site-Defined Events	3-10
3.1.3.4	Event Alias	3-10
3.1.4	Audit Records	3-11
3.1.4.1	Additional Entries in Audit Records	3-12
3.2	Configuring the Audit Subsystem	3-15
3.2.1	Centralizing Audit Data Storage	3-19

3.2.1.1	Configuring Centralized Audit Data Storage on the Audit Hub	3-20
3.2.1.2	Configuring Remote System Audit Data Storage on an Audit Hub	3-21
3.3	Managing the Audit Subsystem	3-21
3.3.1	Changing the Audit Subsystem Startup Defaults	3-21
3.3.2	Starting, Stopping, and Suspending the Audit Daemon ...	3-21
3.3.3	Archiving Audit Logs	3-22
3.3.4	Recovering Audit Data	3-22
3.4	Managing Audit Events	3-23
3.4.1	Displaying the Audit Mask	3-23
3.4.2	Identifying Events that can be Audited on the System	3-23
3.4.3	Enabling Audit Events	3-24
3.4.4	Disabling Audit Events	3-25
3.4.5	Tracing a Process	3-26
3.4.5.1	Displaying Trace Process Data	3-27
3.4.5.2	Auditing Active Processes	3-28
3.4.5.3	Dynamically Auditing Additional System Call Arguments	3-29
3.4.6	Auditing File Operations	3-30
3.5	Generating and Displaying Audit Reports	3-32
3.5.1	Filtering Audit Records	3-34
3.5.2	Displaying Abbreviated Audit Records	3-34
3.5.3	Dependencies Among Audit Events	3-35
3.6	Traditional UNIX Logging Tools	3-37
3.7	Auditing in a TruCluster	3-38
3.7.1	Cluster Command Examples	3-40
3.8	Responding to Audit Reports	3-42

A Enhanced Security

A.1	Installing Enhanced Security	A-1
A.2	Enabling Enhanced Security	A-2
A.2.1	Enabling Enhanced Security Considerations	A-3
A.2.1.1	Using NIS	A-3
A.2.1.2	Segment Sharing	A-3
A.2.1.3	Execute Bit Set Only By Root	A-4
A.2.2	Configuring Enhanced Security	A-4
A.2.2.1	Aging	A-4
A.2.2.2	Minimum Change Time	A-5
A.2.2.3	Changing Controls	A-5
A.2.2.4	Maximum Login Attempts	A-5

A.2.2.5	Time Between Login Attempts	A-5
A.2.2.6	Time Between Logins	A-6
A.2.2.7	Per-Terminal Login Records	A-6
A.2.2.8	Successful Login Logging	A-6
A.2.2.9	Failed Login Logging	A-6
A.2.2.10	Automatic Enhanced Profile Creation	A-6
A.2.2.11	Vouching	A-6
A.2.2.12	Encryption	A-6
A.3	Enhanced Security Databases	A-7
A.3.1	Enhanced (Protected) Password Database	A-7
A.3.2	System Defaults Database	A-7
A.3.3	Terminal Control Database	A-8
A.3.4	File Control Database	A-9
A.3.5	Device Assignment Database	A-9
A.4	Enhanced Security Database Management Utilities	A-10
A.5	Enhanced Security and Authenticating Users	A-10
A.5.1	User Profiles	A-11
A.5.1.1	Recovery of /etc/passwd Information	A-12
A.5.2	Enhanced Security Authentication Database Integrity Checking	A-12
A.5.3	Adding Applications to the File Control Database	A-13
A.6	Enhanced Security and NIS	A-13
A.6.1	Templates for NIS Accounts	A-15
A.6.2	Configure a NIS Master with Enhanced Security	A-16
A.6.2.1	Manual Procedure: Maps for Small User Account Databases	A-17
A.6.2.2	Automated Procedure: Maps for Large User Account Databases	A-17
A.6.3	Setting Up a NIS Slave Server with Enhanced Security ..	A-17
A.6.4	Setting Up a NIS Client with Enhanced Security	A-18
A.6.5	Moving Local Accounts to NIS	A-18
A.6.6	Removing NIS Support	A-18
A.6.7	Implementation Notes	A-19
A.6.8	Troubleshooting NIS	A-21
A.7	Enhanced Security in a TruCluster	A-22
A.7.1	Upgrading from Base to Enhanced Security in a TruCluster	A-22
A.7.2	Installing and Configuring Enhanced Security in a TruCluster	A-22
A.7.3	Access Control Lists	A-23
A.7.4	Distributed Logins and NIS	A-23
A.7.5	Daemons	A-24
A.8	Securing Devices	A-25

A.8.1	Device Security Characteristics	A-25
A.8.1.1	Modifying, Adding, and Removing Devices with the dxdevices Program	A-26
A.8.1.2	Setting Default Values with the dxdevices Program ...	A-26
A.8.2	Updating Security Databases	A-26
A.9	Enhanced Security Troubleshooting	A-27
A.9.1	Lock Files	A-27
A.9.2	Required Files and File Contents	A-28
A.9.2.1	The /tcb/files/auth.db Database	A-28
A.9.2.2	The /etc/auth/system/ttys.db File	A-29
A.9.2.3	The /etc/auth/system/default File	A-29
A.9.2.4	The /etc/auth/system/devassign File	A-30
A.9.2.5	The /etc/passwd File	A-30
A.9.2.6	The /etc/group File	A-30
A.9.2.7	The /sbin/rc[023] Files	A-30
A.9.2.8	The /dev/console File	A-30
A.9.2.9	The /dev/pts/* and /dev/tty* Files	A-30
A.9.2.10	The /sbin/sulogin File	A-31
A.9.2.11	The /sbin/sh File	A-31
A.9.2.12	The /vmunix File	A-31
A.9.3	Problems Logging In or Changing Passwords	A-31

B Secure Shell

B.1	Secure Shell Servers and Clients	B-1
B.2	Secure Shell Overview	B-2
B.3	Configuring the Secure Shell Server and Client	B-3
B.3.1	Configuring the Server	B-3
B.3.2	Configuring the Client	B-6
B.4	Configuring Nonsecure Network Commands to Use Secure Shell	B-8
B.5	Configuring Secure Shell User Authentication	B-9
B.5.1	Configuring Password Authentication	B-10
B.5.2	Configuring Public Key Authentication	B-10
B.5.2.1	Configuring Public Key Authentication on the Client .	B-11
B.5.2.2	Configuring Public Key Authentication on the Server	B-14
B.5.2.3	Accessing a Remote Server	B-15
B.5.2.4	Restricting User Access	B-15
B.5.2.5	Managing Passphrases	B-15
B.5.3	Configuring Host-Based Authentication	B-16
B.6	Managing the Secure Shell Server	B-18
B.6.1	Starting, Stopping, Restarting, and Resetting the sshd2 Daemon	B-19

B.6.2	Restricting Users to Home Directories	B-19
B.6.3	Creating a Public and Private Host Key	B-20
B.6.4	Forwarding TCP/IP Ports and X11 Data Through a Secure Shell Connection	B-20
B.6.4.1	TCP/IP Port Forwarding	B-20
B.6.4.2	X11 Forwarding	B-21
B.7	Using the Secure Shell Commands	B-22
B.7.1	Copying Files Between Clients and Servers	B-22
B.7.1.1	Using the scp2 Command	B-22
B.7.1.2	Using the sftp2 Command	B-23
B.7.2	Logging In and Executing Commands on a Server	B-23

C Single Sign On

C.1	Kerberos Servers and Clients	C-1
C.2	Kerberos Authentication Process	C-2
C.3	Upgrading the SSO Software	C-2
C.4	Installing and Configuring the SSO Software	C-2
C.4.1	Installing and Configuring the SSO Software on the Windows 2000 System	C-3
C.4.1.1	Extending the Active Directory Schema	C-4
C.4.1.2	Updating the MMC	C-5
C.4.2	Installing and Configuring the SSO Software on the Tru64 UNIX System	C-5
C.4.2.1	Configuring the SSO Software	C-5
C.4.2.2	Configuring the SSO Software in a TruCluster Server Environment	C-8
C.4.2.3	Adding Other SIA Mechanisms with Kerberos (if required)	C-8
C.5	SSO Configuration Files on Tru64 UNIX	C-8
C.5.1	The krb.conf File	C-9
C.5.2	The krb.realms File	C-10
C.5.3	The v5srvtab File	C-12
C.5.4	The .k5login File	C-12
C.5.5	The ldapd.conf File	C-13
C.5.6	The ldapusers.deny File	C-15
C.6	Creating Accounts and Groups	C-16
C.6.1	Creating a User Account	C-16
C.6.1.1	Creating a User Account Using the Tru64 UNIX creacct Command	C-17
C.6.1.2	Creating a User Account Using the MMC Interface ...	C-17
C.6.2	Setting a Principal's Password	C-21
C.6.3	Creating a Computer Account	C-21

C.6.4	Creating a Group	C-22
C.7	Managing the SSO Software	C-23
C.7.1	Requesting Tickets	C-23
C.7.2	Displaying Tickets	C-24
C.7.3	Removing the Credential Cache	C-25
C.7.4	Managing the Service Key Table	C-25
C.8	Troubleshooting the SSO Software	C-26
C.8.1	SSO Configuration Problems	C-26
C.8.2	Problems Using the kinit Command or Obtaining an Initial Ticket on Tru64 UNIX	C-27
C.8.3	Password Prompting on Tru64 UNIX	C-28
C.8.4	Problems with SSO in a TruCluster	C-29

D Lightweight Directory Access Protocol

D.1	LDAP Overview	D-1
D.2	Installing the Tru64 UNIX LDAP Client Software	D-2
D.3	Configuring the Tru64 UNIX LDAP Client Software	D-2
D.3.1	Updating the ldapcd.conf File	D-2
D.3.2	Setting the LDAP Runtime Configuration Variable	D-5
D.4	Managing the LDAP Client Daemon	D-6
D.5	Managing Access Control	D-6
D.5.1	The ldapusers.deny File	D-6
D.5.2	The ldapusers.allow File	D-7

E C2 Level Security Configuration

E.1	Establishing a Security Policy	E-1
E.2	Minimum C2 Configuration	E-4
E.3	Initial Configuration	E-5
E.3.1	General Configuration	E-5
E.3.2	Enhanced Passwords and Authentication Using seconfig	E-6
E.3.3	Libraries	E-6
E.3.4	Account Prototypes and Templates	E-7
E.3.5	Configuring the Audit Subsystem	E-7
E.3.6	Verifying That Your Installation Is Secure	E-8
E.3.7	Configuring Network Security	E-8
E.3.8	Postinstallation Security Configuration	E-9
E.3.8.1	umask for Remote Access	E-9
E.3.8.2	Devices	E-9
E.3.8.3	Accounts	E-10
E.3.8.4	Root Access	E-10
E.3.9	Network Configuration	E-11

E.4	Physical Security	E-12
E.5	Applications	E-12
E.6	Periodic Security Administration Procedures	E-12
E.7	Reference Documents and Verification Tools	E-16

Glossary

Index

Examples

1-1	Default /etc/sia/matrix.conf File	1-19
1-2	Sample /var/adm/sialog File	1-22
2-1	Setting Octal Permissions	2-8
2-2	Displaying the ACL for a File	2-21
2-3	Setting the ACL on a File	2-22
3-1	Sample Active Auditing Session	3-28
B-1	Sample sshd2_config File	B-4
B-2	Sample ssh2_config File	B-6
B-3	Public Key Authentication Login Output	B-15
C-1	Sample krb.conf File	C-10
C-2	Sample krb.realms File	C-11
C-3	Sample .k5login File	C-13
C-4	Sample ldapcd.conf File	C-13
C-5	Sample /etc/ldapusers.deny File	C-15
D-1	Sample ldapcd.conf File	D-3
D-2	Default ldapusers.deny File	D-7

Figures

1-1	Security Integration Architecture	1-17
2-1	Tru64 UNIX File Permission Fields	2-3
3-1	The Audit Subsystem	3-2
3-2	Audit Data Flow in a Cluster	3-39
A-1	NIS and Enhanced Security Files	A-14
C-1	New Object — User Window: Required Information	C-18
C-2	New Object — User Window: Password Information	C-19
C-3	Tru64 UNIX User Properties Dialog Box	C-20
C-4	Group Properties Dialog Box	C-23

Tables

1-1	Comparing Authentication Methods	1-3
1-2	Secure Shell Commands	1-13
2-1	Tru64 UNIX Permission Codes	2-2
2-2	Octal Permission Combinations	2-7
2-3	How Octal Numbers Relate to Permission Fields	2-8
2-4	User Mask Permission Combinations	2-9
2-5	Example ACL Entries	2-16
3-1	System Calls Not Always Audited	3-7
3-2	State-Dependent Information	3-37
3-3	Traditional UNIX Log Files in /var/adm	3-38
A-1	Enhanced Security Databases	A-7
A-2	NIS Troubleshooting	A-21
B-1	Traditional Nonsecure Network Commands and Secure Shell Commands	B-1
C-1	SSO Configuration Files	C-8
C-2	Caching Parameters	C-14

About This Manual

This manual describes Tru64 UNIX security concepts and administration including identification, authentication, and authorization methods, securing resources, and auditing.

Audience

This manual is intended for anyone who is responsible for configuring and administering security on a system running the Tru64 UNIX operating system software.

New and Changed Features

This guide has been reorganized and includes information about identification, authentication, and authorization methods that are available in this release.

The information that was previously in the User's section of this guide is now located in the *Command and Shell User's Guide*.

The information that was previously in the Programmer's section of this guide is now located in *Security Programming*.

Organization

The manual is organized as follows:

<i>Chapter 1</i>	Describes local and remote identification, authentication, and authorization methods that Tru64 UNIX can use to authenticate users.
<i>Chapter 2</i>	Describes how to configure Tru64 UNIX to manage access to system resources.
<i>Chapter 3</i>	Describes how to audit Tru64 UNIX to view system activity.
<i>Appendix A</i>	Describes how to install and configure Enhanced Security on Tru64 UNIX.
<i>Appendix B</i>	Describes how to configure the Secure Shell software on Tru64 UNIX.
<i>Appendix C</i>	Describes how to configure the Single Sign On (Kerberos) software on Tru64 UNIX.

<i>Appendix D</i>	Describes how to configure Tru64 UNIX as a client using the Lightweight Directory Access Protocol (LDAP).
<i>Appendix E</i>	Describes how to configure Tru64 UNIX to meet or exceed a C2 level of security.

Related Documentation

The following documents provide important information that supplements the information in certain chapters:

- The *System Administration* manual describes how to perform common Tru64 UNIX administrative tasks.
- The *Command and Shell User's Guide* describes how users access and interact with a system running the Tru64 UNIX operating system software.
- The *Network Administration: Connections* manual describes how to set up, configure, and troubleshoot your network.
- The *Release Notes* might contain important undocumented information about security.

Reader's Comments

HP welcomes any comments and suggestions you have on this and other Tru64 UNIX manuals.

You can send your comments in the following ways:

- Fax: 603-884-0120 Attn: UBPG Publications, ZKO3-3/Y32
- Internet electronic mail: readers_comment@zk3.dec.com

A Reader's Comment form is located on your system in the following location:

```
/usr/doc/readers_comment.txt
```

Please include the following information along with your comments:

- The full title of the manual and the order number. (The order number appears on the title page of printed and PDF versions of a manual.)
- The section numbers and page numbers of the information on which you are commenting.
- The version of Tru64 UNIX that you are using.
- If known, the type of processor that is running the Tru64 UNIX software.

The Tru64 UNIX Publications group cannot respond to system problems or technical support inquiries. Please address technical questions to your local

system vendor or to the appropriate HP technical support office. Information provided with the software media explains how to send problem reports to HP.

Conventions

This manual uses the following typographical conventions:

<code>\</code>	A backslash at the end of a line in an example indicates continuation.
<code>#</code>	A number sign represents the system prompt when you are logged in to a Tru64 UNIX system using the root user account.
net stop	Bold courier type indicates user input.
<code>>>></code>	The console subsystem prompt is three right angle brackets.
<i>file</i>	Italic (slanted) type indicates variable values, placeholders, and function argument names.
[] { }	In syntax definitions, brackets indicate items that are optional and braces indicate items that are required. Vertical bars separating items inside brackets or braces indicate that you choose one item from among those listed.
. . .	In syntax definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
cat(1)	A cross-reference to a reference page includes the appropriate section number in parentheses. For example, cat(1) indicates that you can find information on the cat command in Section 1 of the reference pages.
[Ctrl/x]	This symbol indicates that you hold down the first named key while pressing the key or mouse button that follows the slash. In examples, this key combination is enclosed in a box (for example, [Ctrl/C]).

Identification, Authentication, and Authorization

Identification, authentication, and authorization are the building blocks for system security. System security is built by creating a security policy that determines the level of trust you want between your system and other systems within a local and remote network and users and applications (entities) that access resources on your system.

To fully trust an entity to access resources on your system, you must first identify that entity. Once the entity is identified, it must then prove to the system that it knows something to authenticate it to the system. Finally, after the entity is identified and authenticated, the system will enforce the authorization information that determines which resources the entity is authorized to access on the system.

This chapter contains the following information:

- Authentication overview
- Authentication implementations
- Local authentication methods
- Remote authentication methods
- Security Integration Architecture overview

1.1 Authentication Overview

A successful authentication requires an entity to present some form of credential to the system. A credential can be something you know (user name and password), something you are (biometric or retinal scan), or something you have (smart card or certificate).

Traditionally, the Tru64 UNIX operating system software used local authentication methods to authenticate users by a challenge and response method (user login name and password). The system would attempt to match the user-supplied name and password with an entry in the local user account database.

An entry for a user is created in the user account database when the administrator creates a user account. If there is no matching entry in the user account database, the user is not authenticated. If there is a matching

entry in the user account database and the password is correct, the user is authenticated.

As computing environments became more distributed, open, and diverse, the challenge and response method created the following authentication problems for remote users who accessed a system from another system over some type of network connection:

- Lack of distributed authentication database

Each system maintained its own independent user account database and typically could not share this information with heterogeneous systems. Therefore, users needed a user account on each heterogeneous system from which they accessed resources. In addition, systems could not securely and reliably determine the legitimacy of another system's identity or a user's identity.

- Lack of common encryption methods between systems

Systems could not exchange encrypted data. Data, including user names and passwords, were distributed across the network in clear text.

- Inability to determine data integrity

Systems could not easily detect whether or not data was intercepted, viewed, and modified in transit.

- Inability to determine data source

Systems could not easily prove the origin of data.

To address these authentication problems, remote authentication methods (applications, mechanisms, protocols, and application programming interfaces (APIs)) were developed to provide the following remote authentication services for users who access a system from another system over some type of network connection:

- Distributed authentication database

A primary server stores and maintains user authentication information. Systems direct user authentication requests to the server; they do not need to maintain their own independent user account database. Systems use an authentication method that securely and reliably determines the legitimacy of another system's identity and a user's identity.

- Common encryption methods between systems

Systems exchange encrypted data that is transparent to users.

- The ability to determine data integrity

Systems can detect whether or not data was intercepted, viewed, and modified in transit.

- The ability to determine data source

Systems can prove the origin of data, so that a user or entity cannot deny having performed a particular action related to data or proof of ownership.

Using local and remote authentication methods, you can implement a security policy that creates a security domain with varying levels of trust for local authentication and remote authentication.

With local authentication, the Tru64 UNIX system authenticates a user who is requesting access to its resources. With remote authentication, the Tru64 UNIX system is a client that relies on a remote server to authenticate a user who is requesting access to its resources. Table 1–1 compares the local and remote authentication methods and how they implement authentication.

Table 1–1: Comparing Authentication Methods

Authentication Method	Local or Remote	Implementation
Base (BSD) Security Mechanism	Local	Password
Enhanced Security Mechanism	Local	Password
r* commands (rsh, rcp, and rlogin)	Local	<ul style="list-style-type: none"> • Password • Host-based
Secure Shell application	Remote	<ul style="list-style-type: none"> • Password • Host-based • Public key
Network Information Service (NIS) Protocol	Remote	Password
Lightweight Directory Access Protocol (LDAP)	Remote	Password
Single Sign-on (SSO)/Kerberos Mechanism	Remote	Password and Secret key
Advanced Server for UNIX (ASU)	Remote	Password
IPsec Protocol	Remote	<ul style="list-style-type: none"> • Public key • Pre-shared key (similar to secret key)
Secure Socket Layer (SSL) APIs	Remote	Public key
Common Data Security Architecture (CDSA) APIs	Remote	Public key
GSS APIs	Remote	Password and secret key

1.2 Authentication Implementations

This section describes the authentication implementations.

1.2.1 Password Authentication

Password authentication requires that a user have a password-protected user account that can be authenticated by either a local Tru64 UNIX password authentication mechanism or by using a remote password authentication method.

When a user performs an action that requires authentication (for example, logging in to the system), the user is prompted for a password. The system attempts to match the user-supplied name and password with an entry in the user account database. An entry for a user is created in the user account database when the administrator creates a user account. If there is no matching entry in the user account database, the user is not authenticated. If there is a matching entry in the user account database and the password is correct, the user is authenticated.

1.2.2 Host-Based Authentication

Host-based authentication is a noninteractive UNIX authentication method that is based on host name and user name identification. Host-based authentication allows a user to grant access to their user account to users who use network commands from a specific host; for example, the `rcp`, `rlogin`, or `rsh` command or equivalent Secure Shell commands. See Section 1.4.3 for more information about Secure Shell commands.

Users grant access by including a host name and user name entry in their `.rhosts` file and/or `.shosts` file in their home directory. The file that a user uses depends on the following:

- A `.rhosts` file if the users are using the `rcp`, `rlogin`, or `rsh` command.
- A `.rhosts` file and/or `.shosts` file if the users are using Secure Shell commands or have configured the `rcp`, `rlogin`, and `rsh` commands to use a Secure Shell connection. See Section B.4 for more information on configuring these commands to use a Secure Shell connection.

The `.shosts` file serves the same purpose and has the same format as the `.rhosts` file, but it is read only by the Secure Shell server. If both files exist, the Secure Shell server reads the `.rhosts` file first, then the `.shosts` file. If either of these files allows access for a particular connection, a Secure Shell connection is used, even if the other file does not allow access.

Secure Shell host-based authentication requires additional configuration. See Section B.5.3 for more information about Secure Shell host-based authentication configuration.

An entry in a `.rhosts` or `.shosts` file includes the fully qualified domain name of the host from which a specified user will be authenticated. For example, the following entry in the `$HOME/peter/.shosts` file on host `zeus` allows the users `evan` and `justin` at remote host `saturn.ne.corp.com` to log in to `peter`'s user account on `zeus`:

```
saturn.ne.corp.com evan
saturn.ne.corp.com justin
```

The users `evan` and `justin` will not be prompted for a password and will be logged in to the `peter` user account on `zeus` by entering the following command:

```
$ rlogin -l peter zeus
```

Host-based authentication is best used in scripts and automated processes, such as `cron` jobs, automated backups, automated file transfers, and in other situations where a user will not be present to input authentication information. The nature of any noninteractive login is inherently insecure. Whenever authentication without user challenge is permitted, some level of risk must be assumed.

The `.rhosts` and `.shosts` files have the same format. See `.rhosts(4)` for more information.

1.2.3 Public Key Authentication

Public key authentication is a type of cryptography in which two communicating principals, usually a client and server, use a public and private key to authenticate each other and the user and to encrypt and decrypt the data that they exchange. What one public or private key encrypts, only the other matching public or private key can decrypt. See Section 1.2.5 for more information on encryption.

The public key is published and copied to a specific directory on systems with which the user needs to communicate. The private key is never published or distributed and is assigned a secret passphrase by the user. While the public and private keys are mathematically related to each other, it is impossible to calculate one key from the other; therefore, the private key cannot be compromised through knowledge of its associated public key.

When using public keys, authentication begins when a system (client) request services from another system (server):

1. The client uses its copy of the server's public key to encrypt the message, then sends the message to the server.
2. The server prompts the user for the passphrase and uses the passphrase and its private key to decrypt the message.
3. The server uses its copy of the client's public key to encrypt a response message, then sends the message to the client.
4. The client uses its private key to decrypt the message.

The successful encryption and decryption authenticates a principal, because there is only one public key for each private key.

1.2.3.1 Digital Signatures and Certificates

A user can generate a key pair and release a public key, using any identity with no guarantee that the identity is authentic. To avoid misuse of public keys, you can use digital documents called digital signatures and digital certificates.

A digital signature is generated from a piece of information based on the document and the principal's private key. The digital signature is typically created with a hash and a private signing algorithm, which is then encrypted with the private key. (A hash is a number generated from a string of the text.)

A hash algorithm is performed on the original electronic message's binary code, resulting in what is referred to as a message digest (a 160-bit string of digits unique to the original message). A private signing algorithm is then performed on this message digest. The principal's private key is incorporated into the signature algorithm during the signing process. The resultant string of digits is the digital signature. A principal can verify the signature if it has the public key of the principal that generated the signature.

A digital certificate, is an electronic document used to identify an individual, a server, a company, or some other principal and bind that principal with a public key. Like a driver's license, a passport, or other commonly used personal IDs, a digital certificate provides generally recognized proof of a principal's identity.

Digital certificates are issued, validated, and maintained by a trusted third party called a Certificate Authority (CA). (IPsec provides tools to create certificates for testing purposes.) Like any form of identification, the authenticity of the CA is essential. The methods the CA uses to validate an identity vary depending on the policies of a given CA. In general, before issuing a digital certificate, the CA must use its published verification procedures to ensure that the principal requesting a digital certificate is the

principal it claims to be. Once issued, the CA stores the digital certificate in a repository.

A digital certificate includes the name of the principal it identifies, that principal's public key, an expiration date, the name of the CA that issued the certificate, a serial number, and other information. Most importantly, a digital certificate includes the digital signature of the issuing CA. The CA's digital signature allows a digital certificate to be used when the CA is trusted, but the principal identified by the digital certificate is unknown.

1.2.4 Secret Key Authentication

Secret key authentication is a type of cryptography in which two communicating principals, usually a client or server, use the same secret key, called a session key, to authenticate a system and user and to encrypt and decrypt the data that they exchange. What one session key encrypts, only the other matching session key can decrypt. See Section 1.2.5 for more information on encryption.

A trusted Key Distribution Center (KDC) mediates communication between principals and creates a session key for their exclusive use. When using secret keys, authentication begins at user login:

1. A user enters a user name and password into a system (client).
2. The client sends the user name to the KDC.
The KDC stores an entry for each user that it mediates in an encrypted principal database. The entry contains information about the user, including a password and master key.
3. The KDC looks up the user's entry in the principal database. If there is a matching entry, the KDC creates:
 - A session key for exclusive use between the client and KDC.
 - A Ticket-Granting Ticket (TGT), which contains a copy of the session key, the user's name, and an expiration time. The KDC encrypts the ticket using its master key.
4. The KDC creates a message that contains the session key and the TGT. The message is encrypted with the user's master key, then sent to the client.
5. The client uses a one-way hash function to convert the user's password into the user's master key. The client uses the master key to decrypt the message and extract the session key and TGT.

The session key and TGT are stored in the user's credential cache, which is a file on the client. The client will use the session key and TGT

for subsequent KDC communications. No principal, including the client, can look at or modify the contents of the TGT.

The following steps describe the authentication process when two principals (clients) want to communicate:

1. The requesting principal sends a message to the KDC that contains:
 - The TGT.
 - The identity of the principal with which it wants to communicate.
 - An authenticator, which is a timestamp encrypted by using the session key that the principal and KDC share.
2. The KDC uses its master key to decrypt the TGT. Remember, the TGT contains the user's name and a copy of the session key. The KDC then uses the session key to decrypt the authenticator.

If the authenticator is successfully decrypted, the KDC and principal can confirm each other's identity, because only that principal and the KDC have the same session keys.

3. The KDC creates a pair of tickets for the principals. Each ticket contains the name of the other principal, a timestamp, a time range for which the ticket is valid, and a new session key for exclusive use by the principals.
4. The KDC creates a message that contains the requesting principal's ticket, which includes the requested principal's ticket encrypted with the requested principal's key.

The KDC encrypts the message by using the session key that it shares with the requesting principal and sends the message to the requesting principal.

5. The requesting principal decrypts the message using the session key that it shares with the KDC. A new session key and a ticket for the requested principal is revealed.
6. The requesting principal encrypts an authenticator (timestamp) using the new session key and sends to the requested principal a message that contains the authenticator and the requested principal's ticket.
7. The requested principal:
 - a. Decrypts the ticket by using its master key.
 - b. Extracts the session key from the ticket.
 - c. Uses the extracted session key to decrypt the authenticator.

If the authenticator is successfully decrypted, principals can confirm each other's identity, because the same session key was used to encrypt

and decrypt the authenticator and the session key is used exclusively between the two principals.

1.2.5 Encryption Methods

Encryption is when a system uses cryptographic algorithms, called ciphers, to encrypt and decrypt data. A cipher transforms plain text (unencrypted text) into cipher text (encrypted text). Decryption is when the same cipher is used to transform cipher text to plain text.

Clients and servers must support the same ciphers and can support different cipher suites, or sets of ciphers, depending on factors such as the version of the protocol that they support, company policies regarding acceptable encryption strength, government export restrictions, data sensitivity, and the speed of the cipher. Administrators can enable or disable any of the supported cipher suites for clients and servers.

As part of the initial connection process, a client and server will identify the strongest enabled cipher suites they have in common and use that one for the session.

Almost all ciphers use a key (a variable that is combined in some way with the unencrypted text) and an algorithm (a formula for combining the key with the text). There are two general types of ciphers:

- A block cipher that breaks a message up into sections (for example, 64 bits of text) and combines a key with each section. Most modern ciphers are block ciphers.
- A stream cipher that applies a key to each bit, one at a time.

Examples of cipher algorithms include:

- DES (Data Encryption Standard)
- 3DES
- RSA (Ron Rivest, Adi Shamir, and Len Adleman)
- Blowfish and twofish

1.3 Local Authentication Methods

Local authentication is when the Tru64 UNIX system uses an authentication method to authenticate a user who is requesting access to its resources. This section describes the local authentication methods:

- Base (BSD) Mechanism
- Enhanced Security Mechanism

1.3.1 Base (BSD) Mechanism

Base (BSD) Security is the default level of security that is available on systems running the Tru64 UNIX operating system software.

Base security uses the user information stored in the `/etc/passwd` file to authenticate users. By default, when you create a Tru64 UNIX user account, an entry for the user is added to the `/etc/passwd` file located on the local system. Each entry contains information about a user including user name, user ID (UID), password, log in directory, and shell. See *System Administration* for more information on the `/etc/passwd` file and creating user accounts.

When a user enters a user name and password to log in to the Tru64 UNIX system, the Tru64 UNIX server checks the user supplied information against the entries in the `etc/passwd` file. If there is a matching entry and the password is correct, the authentication succeeds and the user can log in. If there is no matching entry, authentication fails and the user cannot log in.

You can use the `pwck` command or the `vipw` command to edit and perform some consistency checks on the `/etc/passwd` file. Users use the `passwd` command to change a password when using base security.

The Base (BSD) mechanism provides user authorization information through the `/etc/group` file located on the local system. You can use the `grpck` command or the `vipw` command to edit and perform some consistency checks on the `/etc/passwd` file. See *System Administration* for more information on the `/etc/group` file and creating user groups.

1.3.2 Enhanced Security Mechanism

The Tru64 UNIX operating system provides an optional enhanced security mechanism. The enhanced security mechanism builds upon the BSD mechanism by moving some security information into a database, adding the capability to perform break-in evasion and providing more account and password controls. The enhanced security databases are under much stricter system security controls than the BSD files.

When the enhanced security mechanism is installed and configured, the system is referred to as a trusted system. Enabling all enhanced security features will result in a system that can be configured to meet the C2 class of trust, as defined by the *Trusted Computer System Evaluation Criteria* (TCSEC, also called the *Orange Book*). The system also meets the F-C2 functional class as defined in the *Information Technology Security Evaluation Criteria* (ITSEC). See Appendix E for information on how to configure your system to meet C2 class of trust security.

Enhanced security extends BSD security by providing:

- Login control enhancements
- Password enhancements

1.3.2.1 Login Control Enhancements

The following lists enhanced security features for login control:

- Recording the last terminal used for a successful and unsuccessful login
- Recording the time of the last successful login
- Recording the time of the last unsuccessful login attempt
- Recording the number of consecutive unsuccessful login attempts
- Automatic account lockout after a specified number of consecutive bad access attempts
- A per-terminal setting for the delay between consecutive login attempts, and the maximum amount of time each attempt is allowed to complete the login before being declared a failed attempt
- A per-terminal setting for the maximum consecutive failed login attempts before locking any new accesses from that terminal
- Display information about last successful and last unsuccessful login attempts at login time.
- Recording login information whenever a login occurs over the terminal.

1.3.2.2 Password Enhancements

Enhanced security provides the following features for password control:

- Configurable maximum password length, up to 80 characters
- Configurable password lifetimes
- Variable minimum password length
- System-generated passwords that take the form of a pronounceable password made up of meaningless syllables, an unpronounceable password made up of random characters from the character set, or an unpronounceable password made up of random letters from the alphabet. (All letters are from ASCII.)
- Per-user password generation flags, which include the ability to require a user to have a system-generated password
- Record of who (besides the user) last changed the user's password
- Password usage history

1.4 Remote Authentication Methods

Remote authentication is when the Tru64 UNIX system is a client that relies on a remote server to authenticate a user who is requesting access to its resources. The remote authentication method that systems use depends on the common remote authentication method that is configured on the Tru64 UNIX system and the remote server and the type of data that is being exchanged:

- User name and password data that is to be authenticated by a remote server can be exchanged by using NIS or LDAP.
- Data that is exchanged by using the `rcp`, `rlogin`, or `rsh` network commands can use Secure Shell or Kerberos.
- Data that is exchanged by using the `ftp` or `telnet` network commands can use Kerberos.
- Data that is exchanged by applications that use the `rcmd()` function can use Secure Shell.
- Data that is exchanged by using the IP transport can use IPsec.
- Data that is exchanged by distributed applications can use SSL, CDSA, or GSS APIs.

The following sections describe the remote authentication methods.

1.4.1 NIS Protocol

NIS is a distributed client/server data lookup service for sharing information on a local area network (LAN).

A NIS server can store all or part of the BSD or enhanced security user account database. When a user enters a user name and password to log in to a system that is configured as an NIS client, the NIS client sends the user information to the NIS server for authentication. The NIS server checks the user information with the entries in the database and returns the results to the NIS client. If there is a matching entry and the password is correct, the authentication succeeds and the user can log in. If there is no matching entry, the authentication fails and the user cannot log in.

The Tru64 UNIX system can be configured as a NIS client and/or server. See *Network Administration: Services* for more information about NIS.

1.4.2 LDAP Mechanism

The Lightweight Directory Access Protocol (LDAP) is an Internet standard distributed client/server directory service protocol that runs over TCP/IP.

An LDAP server can store user identification and authorization information as entries in an LDAP directory. When a user enters a user name and password to log in to a system that is configured as an LDAP client, the LDAP client sends the user information to the LDAP server for authentication. The LDAP server checks the user information with the entries in the LDAP directory and returns the results to the LDAP client. If there is a matching entry and the password is correct, the authentication succeeds and the user can log in. If there is no matching entry, the authentication fails and the user cannot log in.

The Tru64 UNIX system can be configured as an LDAP client and/or server. See Appendix D for information on configuring Tru64 UNIX as an LDAP client for user authentication.

1.4.3 Secure Shell Application

Secure Shell is a client/server application that provides a suite of network commands that provides remote authentication services for data that is exchanged when using a Secure Shell command. You can use the Secure Shell commands in addition to or in place of the traditional nonsecure network commands (Table 1–2).

Table 1–2: Secure Shell Commands

Task	Traditional Command	Secure Shell Command
Execute commands on a remote system	rsh	ssh
Log in to a remote system	rlogin or telnet	ssh
Transfer files between systems	rcp or ftp	scp or sftp

Optionally, you can configure the `rcp`, `rlogin`, and `rsh` commands and the `rcmd()` function to automatically use Secure Shell.

By default, the Secure Shell software is installed when you upgrade to or install the Tru64 UNIX Version 5.1B or higher operating system software. See Appendix B for more information on configuring and using the Secure Shell software.

1.4.4 SSO/Kerberos Mechanism

The Single Sign-On (SSO) software is optional client/server software that uses Kerberos to provide remote authentication services for data that is exchanged when using the `ftp`, `rcp`, `rlogin`, `rsh`, and `telnet` network commands.

See Appendix C for more information on installing, configuring, and using the SSO software.

1.4.5 Advanced Server for UNIX (ASU)

The Advanced Server for UNIX (ASU) software is optional client/server software that provides a mechanism that you can use to configure the Tru64 UNIX operating system software to direct authentication requests to a Windows NT Server Version 4.0.

The Windows NT Server Version 4.0 uses its user account information to authenticate users on behalf of the Tru64 UNIX system. This is useful if you have user account information stored on a Windows NT Server Version 4.0 and you do not want to create a user account database on the Tru64 UNIX system.

See *ASU Installation and Administration Guide* for more information on configuring Windows NT Version 4.0 authentication.

1.4.6 IPsec Protocol

The IPsec protocol is an optional security framework that provides remote authentication services at the IP layer for data that is exchanged by using the TCP/IP transport protocol.

By enabling IPsec to secure the IP layer and configuring desired connections, you secure the network. Applications, commands, and utilities can transparently use IPsec without modification.

See *Network Administration: Connections* for more information on enabling and configuring IPsec.

1.4.7 SSL, CDSA, and GSS APIs

The Secure Socket Layer (SSL) APIs, Common Data Security Architecture (CDSA) APIs, and Generic Security Service (GSS) APIs provide remote authentication services when a distributed application is written to use them. Applications use the APIs to request remote authentication services without requiring knowledge of the underlying security mechanisms. The security APIs support a wide range of security mechanisms.

- CDSA is an architecture in which software and hardware vendors create remote authentication service modules that applications can use for specific services. Applications use the Common Security Services Manager (CSSM) APIs to request authentication services from modules in the CDSA.
- The SSL APIs are mainly for Web application developers to provide remote authentication services between two or more sockets.
- The GSS APIs allow applications to provide remote authentication services across a network based on a socket-like protocol where the

client first determines if the server has the necessary authentication mechanism to authenticate, then uses that mechanism for the authentication.

See `cdsa(3)`, `ssl(3)`, and *Security Programming* for more information.

1.5 Security Integration Architecture Overview

Tru64 UNIX can be configured such that one or more security methods can be used. Tru64 UNIX uses the Security Integration Architecture (SIA) to manage the order in which security methods are used. Each method provides a security mechanism that is added to the SIA. The security mechanism defines its own rules to determine the manner of authentication and the level of trust the mechanism has to achieve before authenticating the entity.

SIA has two layers of interfaces, a mechanism-independent interface for application, command, and utility programmers and a mechanism-dependent interface for system security providers. Security sensitive applications, commands, and utilities call SIA mechanism-independent interfaces to provide security services. The mechanism-independent interfaces call corresponding routines in each configured SIA mechanism layer to determine whether or not the requested access is to be allowed.

A system security provider provides a security mechanism as a shared library that includes a set of predefined entry points. As security needs change, new SIA mechanisms can be added without requiring changes to applications, commands, and utilities.

The following security mechanisms are provided with Tru64 UNIX:

- Base (BSD) Security mechanism in the `libc.so` library (default).
- Enhanced Security mechanism in the `/usr/shlib/libsecurity.so` library. This is an optional mechanism located in the OSFC2SEC subset.
- LDAP mechanism in the `/usr/shlib/libisialdap.so` library. This is an optional mechanism located in the OSFLDAPAUTH and OSFSSOW2K subsets.
- Kerberos-based mechanism in the `/usr/shlib/libcsfk5siad.so` library. This is an optional mechanism located in the OSFSSOW2K subset.

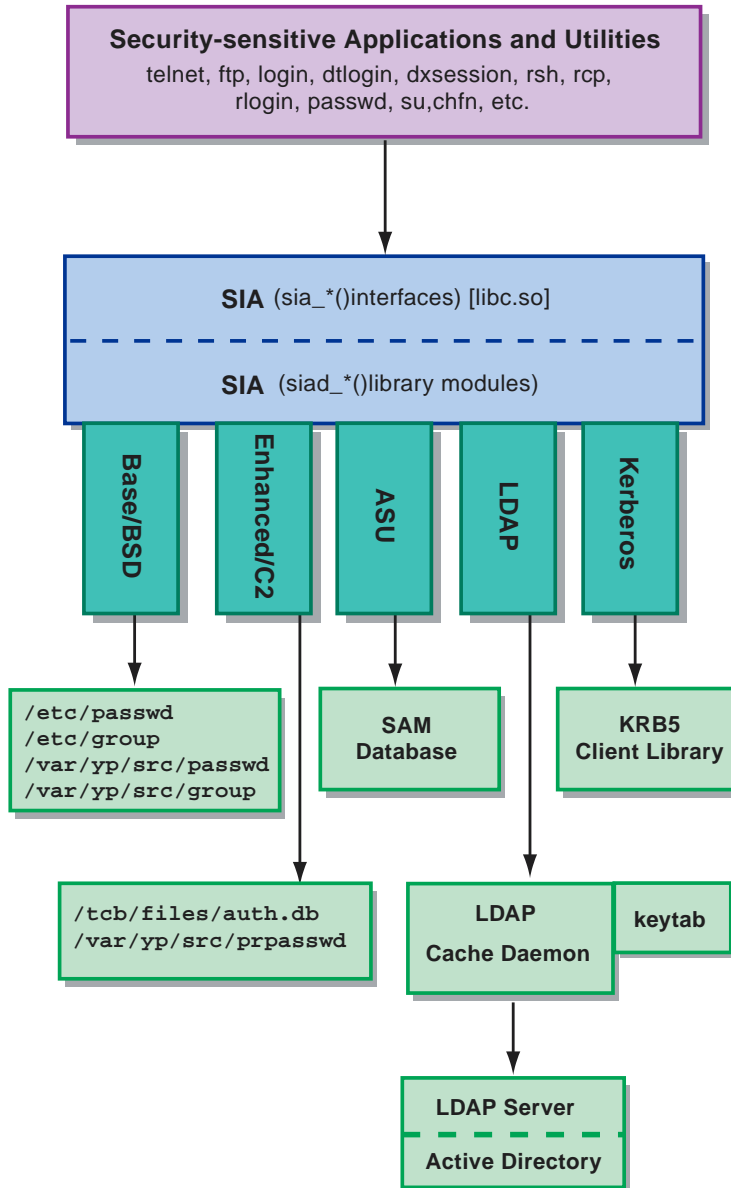
An SIA mechanism implements a security policy through a set of predefined interfaces that cover the following basic areas:

- Session control
- Retrieval of authentication (password and authorization) data
- Change of the password, finger, and shell data.

An SIA mechanism does not have to provide all the security functionality, but if it provides change functionality to its database, then it must also provide session functionality.

The SIA framework provides the structure through which different mechanisms can coexist on a Tru64 UNIX system or in a TruCluster Server environment. For example, if a user tries to log in with a Kerberos password and the authentication attempt fails because the Kerberos server is not available, then the login sequence could automatically attempt to use the local `/etc/passwd` file through the default BSD mechanism to authenticate the user. Figure 1-1 shows the SIA architecture.

Figure 1–1: Security Integration Architecture



ZK-0685U-AI

1.5.1 SIA and User Change Routines

When a user enters a command for a change routine (for example the `passwd`, `chfn`, and `chsh` commands), a call is made to a mechanism-dependent

routine to determine whether or not the mechanism supports the attempted change and if the user for which the change is being attempted is known to the mechanism.

When using the `chfn` command to change finger information and the `chsh` command to change shell information, if more than one mechanism knows the user, a menu is displayed from which the user selects the security mechanism to affect the change.

When using the `passwd` command to change password information, the mechanism-independent interface identifies which mechanisms support password change for any given user. If more than one mechanism is identified, a menu is displayed from which the user can select one or every mechanism for which to change the password.

See *System Administration* for more information about change routines.

1.5.2 Configuring SIA

SIA can be configured such that one or more security mechanisms can be used. A typical set of mechanisms might include a local mechanism, one only concerned with the local system security, and a remote (distributed) security mechanism, one is concerned with aspects of security which span several systems. The SIA layering allows these two mechanisms to coexist. The mechanism-dependent routines handle authentication based on whether a user is recognized by the mechanism. For example, mechanism integration is useful for login or session processing. The SIA layering passes context between the various mechanisms during the session processing. This context contains collected names, passphrases, and the current state of the session processing.

Mechanisms can trust the authentication process of a previously run mechanism thus allowing authentication vouching. In this case, if a user is successfully authenticated by the distributed mechanism, then the local mechanism can accept or trust that authentication and continue with the session processing. SIA also allows the local mechanism to not accept vouching. In this case, the local mechanism would be forced to do its own authentication processing regardless of previous authentication results. This could result in the user being prompted for the password several times. Although SIA allows any ordering of mechanisms, you should configure the mechanisms that accept vouching after those that do not. The mechanisms provided with Tru64 UNIX can accept authentication vouching.

Each SIA mechanism requires an entry in the `/etc/sia/matrix.conf` file. The order in which entries are listed in the `matrix.conf` file determines the order in which SIA mechanisms are called. Use the `/usr/sbin/siacfg` utility to manage SIA mechanisms entries in the `matrix.conf` file.

The `matrix.conf` file contains a line for each of the predefined mechanism-dependent `siad_*()` interfaces. Each line contains attributes of a unique mechanism identifier or `mech_type` and the path to the mechanism's shared library. The following is a sample entry in the `matrix.conf` file:

```
siad_init=(BSD,libc.so)
```

Where, `siad_init` is the routine, `BSD` is the unique identifier, and `libc.so` is the shared library with the `siad_init` routine. Because the `libc.so` library contains a default mechanism, no path to the library is required because it is already open. When more than one mechanism is configured, there will be multiple attributes per entry, for example:

```
siad_init=(OSFC2,/usr/shlib/libsecurity.so),(BSD,libc.so)
```

In this case, the `siad_init` routine in the `libsecurity.so` library is called first followed by the routine in the `libc.so` library. The `sia_*()` interfaces uses this set of attributes to call mechanisms in a left to right ordering. When a mechanism does not provide support for a particular category of interfaces, the default BSD mechanism will be used.

Note

Prior to Tru64 UNIX Version 5.0, the `siacfg` command did not exist and the `/etc/sia/matrix.conf` file was a link to another file, such as `/etc/sia/bsd_matrix.conf` file or `/etc/sia/OSFC2_matrix.conf` file to prevent the `matrix.conf` file from being edited and to ensure that dependent-mechanism providers supplied a configuration file that included the attributes for their own mechanism.

The Tru64 UNIX operating system provides a default `matrix.conf` file that will use the BSD mechanism routines contained in the `libc.so` library. Example 1-1 shows the default `matrix.conf` file.

Example 1-1: Default `/etc/sia/matrix.conf` File

```
# sia matrix configuration file (BSD only)
siad_init=(BSD,libc.so)
siad_chk_invoker=(BSD,libc.so)
siad_ses_init=(BSD,libc.so)
siad_ses_authent=(BSD,libc.so)
siad_ses_estab=(BSD,libc.so)
siad_ses_launch=(BSD,libc.so)
siad_ses_suauthent=(BSD,libc.so)
siad_ses_reauthent=(BSD,libc.so)
siad_chg_finger=(BSD,libc.so)
siad_chg_password=(BSD,libc.so)
```

Example 1–1: Default `/etc/sia/matrix.conf` File (cont.)

```
siad_chg_shell=(BSD,libc.so)
siad_getpwent=(BSD,libc.so)
siad_getpwuid=(BSD,libc.so)
siad_getpwnam=(BSD,libc.so)
siad_setpwent=(BSD,libc.so)
siad_endpwent=(BSD,libc.so)
siad_getgrent=(BSD,libc.so)
siad_getgrgid=(BSD,libc.so)
siad_getgrnam=(BSD,libc.so)
siad_setgrent=(BSD,libc.so)
siad_endgrent=(BSD,libc.so)
siad_ses_release=(BSD,libc.so)
siad_chk_user=(BSD,libc.so)
```

1.5.2.1 SIA Mechanism Initialization

The SIA mechanisms are initialized by the `/usr/sbin/siainit` utility when the system boots. The `siainit` utility reads the `/etc/sia/matrix.conf` file and calls each of the configured mechanisms at its `siad_init()` entry point to perform boot initialization.

An error response from any `siad_init()` call will force the system into single user mode and result in an `SIA Initialization Failure` message to be displayed on the console. Consequently, only problems that would cause a security risk or would not allow the root user to log in will warrant a failure response from the `siad_init()` call. Once each `siad_init()` routine is run successfully, the `/etc/sia/siainitgood` file is created, if it does not already exist. Upon failure, the `/etc/sia/siainitgood` file is removed.

In a TruCluster environment, if the `siainitgood` file is removed, processes executing on nodes might begin to have failures for authentication or authorization requests until the problem is resolved. Subsequent attempts to use the SIA interfaces will check to ensure that the `siainitgood` file exists prior to setting up the internal structures that describe the `matrix.conf` entry points for each configured mechanism.

1.5.2.2 Adding an SIA Mechanism

Each SIA mechanism provider should use the `siacfg` command to update the `matrix.conf` file to support their mechanism.

When adding a mechanism to the SIA, the `siacfg` command requires a unique name for the mechanism and the location of a mechanism's shared library. The `siacfg` command opens the library and looks for each of the

functional categories, ensuring that each of the required functions is in the library. If all are present, an entry for each routine is added to the `matrix.conf` file.

Follow these steps to manually add an SIA mechanism:

1. Install the mechanism according to its installation procedure.
2. Change to the `/etc/sia` directory.
3. Add the SIA mechanism; for example:

```
# /sbin/siacfg -a SIA_mechanism_name library_path
```
4. Reboot the system; for example:

```
# /usr/sbin/reboot
```

1.5.2.3 Removing an SIA Mechanism

Follow these steps to remove an SIA mechanism:

1. Bring the system to single-user mode; for example:

```
# /usr/sbin/shutdown now
```
2. Remove the SIA mechanism; for example:

```
# /sbin/siacfg -r SIA_mechanism_name
```

The `siacfg` command removes routine entries in the `matrix.conf` file with the specified `SIA_mechanism_name`.
3. Reboot the system; for example:

```
# /usr/sbin/reboot
```

1.5.3 SIA Logging

SIA supports only the passing of a success or failure response back to the calling command or utility; therefore, it can be difficult to determine which of the SIA mechanisms caused an error. Thus, SIA supports a general logging capability that produces time-stamped entries in the `/var/adm/sialog` file for the following types of log entries:

- `EVENT`, which is generally a success case within the SIA processing.
- `ERROR`, which is generally a failure within the SIA processing.
- `ALERT`, which represents security configuration or security risks within the SIA interfaces.

You must create the `/var/adm/sialog` file; for example:

```
# touch /var/adm/sialog
```

Example 1–2 shows a sample `/var/adm/sialog` file.

Example 1–2: Sample /var/adm/sialog File

```
SIA:EVENT Wed Feb 3 05:21:31 2002
Successful SIA initialization
SIA:EVENT Wed Feb 3 05:22:08 2002
Successful session authentication for terry on :0
SIA:EVENT Wed Feb 3 05:22:08 2002
Successful establishment of session
SIA:ERROR Wed Feb 3 05:22:47 2002
Failure to authenticate session for root on :0
SIA:ERROR Wed Feb 3 05:22:52 2002
Failure to authenticate session for root on :0
SIA:EVENT Wed Feb 3 05:22:59 2002
Successful session authentication for root on :0
SIA:EVENT Wed Feb 3 05:22:59 2002
Successful establishment of session
SIA:EVENT Wed Feb 3 05:23:00 2002
Successful launching of session
SIA:EVENT Wed Feb 3 05:24:40 2002
Successful authentication for su from root to terry
SIA:EVENT Wed Feb 3 05:25:46 2002
Successful password change for terry
```

Securing System Resources

After a user is identified and authenticated, Tru64 UNIX uses Discretionary Access Control (DAC) to manage access to system resources including directories, files, devices, and processes. Tru64 UNIX implements DAC through the use of Tru64 UNIX permissions and Access Control Lists (ACLs).

This chapter contains the following information:

- Access control overview
- Tru64 UNIX permissions
- Access Control Lists (ACLs)

2.1 Access Control Overview

The Tru64 UNIX operating system software uses Tru64 UNIX permission and ACLs to control access to files and directories. You use Tru64 UNIX permissions and ACLs to set read, write, and execute access permissions on files and directories for the owning user, the owning user's group, and all other users. In addition, you use ACLs to set read, write, and execute access permissions on files and directories for specific users and groups.

The following list describes how Tru64 UNIX permissions and ACLs are checked when a process requests access to a file or directory:

- If the requesting process has the root user privilege, access to the file or directory is granted. ACL and Tru64 UNIX permissions are not checked.
- If ACLs are not enabled or if there is no ACL associated with the file or directory, only Tru64 UNIX permission is checked.
- If ACLs are enabled and there are associated ACLs with the file or directory, the server collects the ACL of each object in the path to the requested resource to determine access. Access is denied if the ACL for any object in the path prohibits access.

2.2 Tru64 UNIX Permissions

When a file or directory is created, by default, the operating system or the program you are running assigns Tru64 UNIX read, write, and execute permissions for the three classes of users:

- u (user/owner)

- g (group)
- o (all others; also known as world)

Table 2–1 describes the codes for the Tru64 UNIX permissions as they apply to files and directories.

Table 2–1: Tru64 UNIX Permission Codes

Permission Code	File	Directory
r (read)	Contents can be viewed or printed.	Contents can be read, but not searched. Usually r and x are used together.
w (write)	Contents can be changed or deleted.	Entries can be added or removed.
x (execute)	File can be used as a program.	Directory can be searched.

2.2.1 Displaying Tru64 UNIX File and Directory Permissions

To display the Tru64 UNIX permissions for a file, enter:

```
$ ls -l filename
```

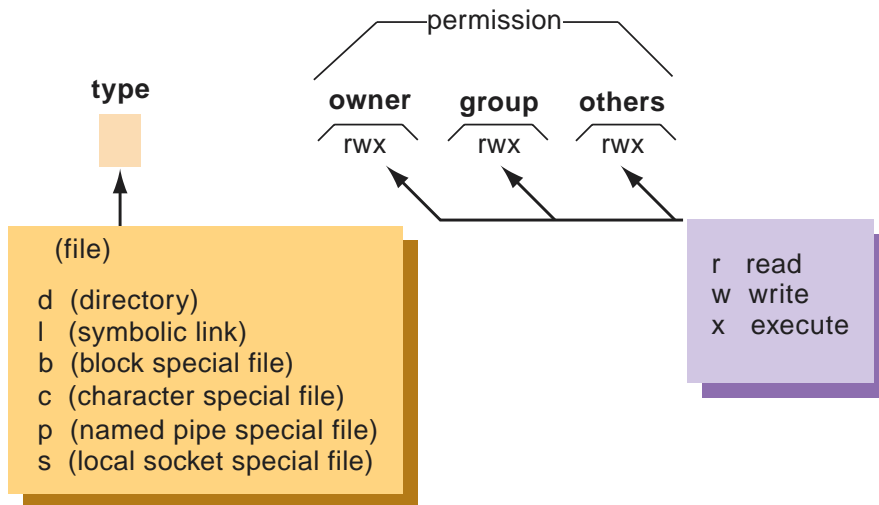
You can specify more than one *filename* entry.

To display the Tru64 UNIX permissions for the files and subdirectories in your current directory, enter the `ls -l` command:

```
$ ls -l
total 7
-rw-r--r-- 1 larry system 101 Jun 5 10:03 file1
-rw-r--r-- 1 larry system 171 Jun 5 10:03 file2
-rw-r--r-- 1 larry system 130 Jun 5 10:06 file3
drwxr-xr-x 2 larry system 32 Jun 5 10:07 project
-rw-r--r-- 1 larry system 0 Jun 5 11:03 record1
-rw-r--r-- 1 larry system 0 Jun 5 11:03 record6
drwxr-xr-x 2 larry system 32 Jun 5 10:31 reports $
```

The first field contains the permission code that identifies the type of entry followed by the permission set for user, group, and world. A dash (-) in the first place indicates that the entry is a file. A dash in subsequent positions indicates that the specified class of user does not have permission for that operation. Figure 2–1 describes the permission fields.

Figure 2–1: Tru64 UNIX File Permission Fields



ZK-0536U-AI

Note

Although a file's owner, group, and other permissions shown by the `ls` command specify that a process has access to a file, the file's ACL might not allow the process access. This can be true even if the process has the same effective group as the group for the file. See Section 2.3 for more information on ACLs.

2.2.2 Setting File and Directory Permissions (`chmod`)

Users use the `chmod` (change mode) command to set or change the permissions on their files and directories. Only the owner of a file or directory or the root user can set or change the permission on a file or directory.

Users usually permit themselves read, modify, and execute permission on a file; permit members of their group the read permission on a file and the modify or execute permission depending upon the nature of their work and the composition of the group; and prohibit all other users from having any access to a file.

It is important to realize that whatever permissions are set on a file and directory, the root user can always override them. For example, if a user uses the `chmod` command to specify that only the user can access the `report20` file, the root user can still access this file.

To specify the permissions set by the `chmod` command:

- Specify permissions with letters and operation symbols.
- Specify permissions with octal numbers.

2.2.2.1 Specifying Permissions with Letters and Operation Symbols

You can use letters and operation symbols to change file and directory permissions.

The following is the format of the `chmod` command when using letters and operation symbols:

chmod *userclass-operation-permission filename*

The *userclass-operation-permission* entry actually represents three codes as follows:

- The *userclass* is one of the following letters:
 - u for user (owner)
 - g for group
 - o for all others (besides owner and group)
 - a for all (user, group, and all others)
- The *operation* is one of the following symbols:
 - + to add permission
 - to remove permission
 - = to assign permission regardless of previous setting
- The *permission* is one of the following letters:
 - r for read
 - s to set user or group ID: This permission sets the effective user ID or group ID to that of the owner or group owner of file whenever the file is run. Use this permission setting with the u or g *userclass* entry to allow temporary or restricted access to files not normally accessible to other users. When you use the `ls -l` command to display permissions, an s appears in the user or group execute position to show that the file runs with set-user-ID or set-group-ID permission.
 - w for write
 - x for execute

The *filename* entry is the name of the file or files whose permissions you want to change. You can also use pattern-matching characters to specify files. See Section 2.2.2.1.3 for more information.

2.2.2.1.1 Changing File Permissions

Follow these steps to change permissions on a file called `file1`:

1. Display the permissions for the `file1` file:

```
$ ls -l file1
-rw-r--r-- 1 larry  system  101 Jun  5 10:03 file1
```

The owner (`larry`) has read/write permissions while the group and others have only read permissions.

2. Expand the permissions for both the group (`g`) and for others (`o`) by giving them write access (`+w`) to `file1` in addition to the read (`r`) access they already have:

```
$ chmod go+w file1
```

3. Display the new file permissions:

```
$ ls -l file1
-rw-rw-rw- 1 larry  system  101 Jun  5 10:03 file1
```

The group and all other system users have read and write (`rw`) permissions to the `file1` file.

2.2.2.1.2 Changing Directory Permissions

The procedure for changing directory permissions is the same as that for changing file permissions. However, to display information about a directory, enter the `ls -ld` command.

Follow these steps to change permissions on a directory called `project`:

1. Display the permissions for the `project` directory:

```
$ ls -ld project
drwxr-xr-x 2 larry  system  32 Jun  5 10:07 project
```

2. Expand the permissions for the group (`g`) by giving them write access (`+w`) to `project` in addition to the read and execute (`r-x`) access they already have:

```
$ chmod g+w project
```

3. Display the new directory permissions:

```
$ ls -ld project
drwxrwxr-x 2 larry  system  32 Jun  5 10:07 project
```

The group has read, write, and execute (`rwx`) permissions to the `project` directory.

2.2.2.1.3 Using Pattern-Matching Characters

If you want to make the same change to the permissions for all entries in a directory, you can use the pattern-matching character asterisk (*) with the `chmod` command.

To give execute (x) permission to the group (g) for all files (*) in the current directory, enter:

```
$ chmod g+x *
```

To display the new group permissions for all files in the current directory, enter:

```
$ ls -l
total 7
-rw-rwxrw- 1 larry  system  101 Jun 5 10:03 file1
-rw-r-xr-- 1 larry  system  171 Jun 5 10:03 file2
-rw-r-xr-- 1 larry  system  130 Jun 5 10:06 file3
drwxrwxr-x 2 larry  system   32 Jun 5 10:07 project
-rw-r-xr-- 1 larry  system    0 Jun 5 11:03 record1
-rw-r-xr-- 1 larry  system    0 Jun 5 11:03 record6
drwxr-xr-x 2 larry  system   32 Jun 5 10:31 reports
```

2.2.2.1.4 Setting Absolute Permissions

An absolute permission (=) assignment resets all permissions for a file or files, regardless of how the permissions were set previously.

Follow these steps to reset all the permissions for the `file3` file:

1. Display the file permissions:

```
$ ls -l file3
-rw-r-x-r-- 1 larry  system  130 Jun 5 10:06 file3
```

2. Assign all three permissions (rwx) to all users (a):

```
$ chmod a=rwx file3
```

3. Display the new file permissions:

```
$ ls -l file3
-rwxrwxrwx 1 larry  system  130 Jun 5 10:06 file3
```

You can also use an absolute assignment to remove permissions.

Follow these steps to remove the execute permission for the `file3` file:

1. Remove the execute permission (x) for all groups (a) from the `file3` file:

```
$ chmod a=rw file3
```

2. Display the new file permissions:

```
$ ls -l file3
-rw-rw-rw- 1 larry system 130 Jun 5 10:06 file3
$
```

2.2.2.2 Specifying Permissions with Octal Numbers

You can use octal numbers to change file and directory permissions.

The following is the format of the `chmod` command when using octal numbers:

chmod *octalnumber filename*

The *octalnumber* entry is a 3-digit octal number that specifies the permissions for owner, group, and others. The *filename* entry is the name of the file (or a list of file names separated by spaces) whose permissions you want to change. You can also use pattern-matching characters to specify files. See Section 2.2.2.1.3 for more information.

An octal number corresponds to each type of permission as follows:

```
4 = read (r)
2 = write (w)
1 = execute (x)
```

To specify a group of permissions for a permission field, add together the appropriate octal numbers; for example:

```
3 = -wx (2 + 1)
6 = rw- (4 + 2)
7 = rwx (4 + 2 + 1)
0 = --- (no permissions)
```

Table 2-2 lists the possible octal permission combinations.

Table 2-2: Octal Permission Combinations

Octal Number	Binary Number	Permissions	Description
0	000	None	No permissions granted
1	001	--x	Execute
2	010	-w-	Write
3	011	-wx	Write/execute
4	100	r--	Read
5	101	r-x	Read/execute

Table 2–2: Octal Permission Combinations (cont.)

Octal Number	Binary Number	Permissions	Description
6	110	rw-	Read/write
7	111	rwx	Read/write/execute

The entire permission code for a file or directory is specified with a 3-digit octal number, one digit each for owner, group and others. Table 2–3 shows some typical permission codes and how they relate to the permission fields.

Table 2–3: How Octal Numbers Relate to Permission Fields

Octal Number	Owner Field	Group Field	Others Field	Complete Code
777	rwx	rwx	rwx	rwxrwxrwx
755	rwx	r-x	r-x	rwxr-xr-x
700	rwx	---	---	rwx-----
666	rw-	rw-	rw-	rw-rw-rw-

In Example 2–1, the `ls -l file3` command displays the permissions for the `file3` file. Then the `chmod 754 file3` command gives all three permissions (`rwx`) to the user, read and execute (`rx`) to group, and read only (`r`) to all users.

Example 2–1: Setting Octal Permissions

```
$ ls -l file3
-rw-rw-rw- 1 larry system 130 Jun 5 10:06 file3
$ chmod 754 file3
$ ls -l file3
-rwxr-xr-- 1 larry system 130 Jun 5 10:06 file3
$
```

2.2.3 Setting Default Permissions

When a file or directory is created, default permissions are set for it. The default permissions are initially set either by the operating system or the program that created the file or directory. Setting default permissions relieves users of the task of specifying permission codes explicitly every time they create a file or directory. The operating system assigns the default permission values of read, write, and execute (`777`) for executable files and read and write (`666`) for all other files.

To change the default permissions on newly created files and directories, you set the user mask with the `umask` command. The user mask is a numeric value that determines the default permissions when a file or directory is

created. As a result, when you create a file or directory, its permissions are set to what the creating program specifies, minus what the user mask value forbids.

The following is the format of the `umask` command:

umask *octal_number*

The *octal_number* entry is a 3-digit octal number that specifies the permissions to be subtracted from the default permissions (777 or 666).

Setting the user mask is similar to setting the permission by using octal numbers (Section 2.2.2.2). The permission code for a file or directory is specified with a 3-digit octal number. Each digit represents a type of permission. The position of each digit (first, second or third) represents 3 bits that correspond to the following:

- The first digit is for the `owner` of the file.
- The second digit is for the `group` of the file.
- The third digit is for `others`.

When you set the user mask, you actually are specifying which permissions are not to be granted, regardless of the permissions requested by the file creating program. Table 2–4 lists the user mask permission combinations. The `umask` permission values are the inverse of those specified for regular permission codes.

Table 2–4: User Mask Permission Combinations

Column Head	Allowed Permissions	Description
0	<code>rxw</code>	Read/write/execute
1	<code>rw-</code>	Read/write
2	<code>r-x</code>	Read/execute
3	<code>r-</code>	Read
4	<code>-wx</code>	Write/execute
5	<code>-w-</code>	Write
6	<code>-x</code>	Execute
7	<code>---</code>	No permissions granted

For example, if you specify a user mask of 027 (and the file is executable):

- The `owner` is allowed all permissions requested by the program creating the file.
- The `group` is not allowed write permission.
- The `others` are not allowed any permissions.

The default user mask is 022, which gives the owner all the permissions and prevents members of the user's group or any other users from writing to the owner's files.

A good user mask value to set for your own files and directories depends upon how freely information resources are shared on your system. The following guidelines might be useful:

- In an open computing environment, you might specify 000 as a user mask value, which allows no restrictions on file or directory access. As a result, when a program creates a file and specifies permission codes for it, the user mask imposes no restrictions on what the creating program has specified.
- In a more secure computing environment, you might specify 066 as a user mask value, which allows you total access but prevents all others from being able to read or write to your files. As a result, when a file is created, its permissions are set to what the creating program specifies, minus the user mask restrictions that prevent read or write access for everyone but you.
- In a secure computing environment, you might specify 077 as a user mask value, which means that only you have access to your files. As a result, when a file is created, its permissions are set to what the creating program specifies, minus the user mask restrictions that prevent anyone else from reading, writing or executing your files.

To understand how a user mask works, assume that you have entered the following command:

```
$ umask 037
```

This command establishes a Tru64 UNIX permission code of 740 if the file is executable (the 777 Tru64 UNIX default permission code minus the 037 umask code) and produces the following results:

- You (the owner) are allowed all permissions (7).
- Members of your group are allowed read only permission (4).
- Others are not allowed any permissions (0).

Furthermore, assume that you have just created a file. By default, your text editor always assigns the following default permissions: owners are allowed all permissions and all others only read and execute permissions. However, because you have previously set a user mask of 037, it further restricts the file permissions. As a result, the owner still has all permissions, but the group cannot execute the file and all others have no permissions.

2.2.3.1 Setting the User Mask

You set the user mask in the following ways:

- Include the `umask` command in user login scripts. This is the most common and efficient way to specify a user mask, because the specified value is set automatically when the user logs in.
- Enter the `umask` command at the shell prompt during a login session. The user mask value you set is in effect for that login session only.

The following steps provide a more detailed example of how the user mask works in restricting permissions for files you create with a text editor:

1. Display the current value of your user mask:

```
$ umask
```

- If the user mask value is 000, there are no restrictions on the permissions established by file-creating programs. Go to step 3.
- If the user mask value is set, write it down. Go to step 2.

2. Set the user mask value to 000, so that there will be no restrictions on the permissions established by file-creating programs. Before resetting the user mask, make sure you have written down the current value in case you need to reset it.

```
$ umask 000
```

3. Create a file, save it and then exit your text editor.
4. Display the permissions of the file. We will assume that read and write permissions are granted for all users:

```
$ ls -l  
-rw-rw-rw- 1 user-name 15 Oct 27 14:42 yourfile
```

5. Reset the user mask to 022:

```
$ umask 022
```

A user mask of 022 establishes the following permission restrictions: owners are allowed all permissions and all others are allowed only read and execute permissions.

6. Create another file, save it and then exit your editor.
7. Display the permissions of the file:

```
$ ls -l  
-rw-r--r-- 1 user-name 15 Oct 27 14:45 yourfile2
```

The write permissions for the group and all others have been removed in accordance with the user mask value of 022.

8. Reset the user mask to its original value or to another value (if you choose).

2.3 Access Control Lists (ACLs)

ACLs are an extension of Tru64 UNIX permissions on files and directories. ACLs can be used to set permissions for specific users and groups on files and directories. In addition to the permissions normally associated with a file or directory, an ACL contains a list of users and groups with the specific permissions for each user or group.

A file can have only one ACL associated with it, an access ACL. A directory can have three ACLs associated with it: an access ACL, a default access ACL, and a default directory ACL. The access ACL is used to determine who has access to the file or directory. The default ACLs are used to determine what ACLs are inherited by files and directories created in the directory with the default ACLs.

You can set ACLs at any time, but access checking of ACLs and ACL inheritance take place only when ACLs are enabled. The ACL commands will display warning messages if ACLs are disabled on your system. Not all file systems support ACLs. You can set ACLs only on files and directories that are on file systems that support ACLs.

ACLs are extended file attributes stored in the file or directory's property list. ACLs can be associated with any file or directory on a file system that supports property lists, even if ACLs are disabled on the system. ACL access checking and ACL inheritance take place only when ACLs are enabled. See `acl(4)` and `proplist(4)` for more information.

The following file systems support property lists:

- Advanced file system (AdvFS)
On AdvFS file systems there is a hard limit of 1560 bytes for a property list entry. Because ACLs are stored in property list entries, this equates to 62 ACL entries in addition to the three directory ACL entries. The error `EINVAL` is returned if you exceed this limit.
- Network file system (NFS) where both client and server are Tru64 UNIX systems. See Section 2.3.1.1 for more information on enabling ACLs on NFS.
- UNIX file system (UFS)
The default value of the UFS ACL limit is 1548 bytes (the AdvFS limit includes the header), equivalent to the 62+3 required entry limit on AdvFS.
The UFS configurable limit on ACLs is defined by the `ufs-sec-proplist-max-entry` attribute in the `sec` subsystem.

The UFS configurable property list element size is defined by the `ufs-proplist-max-entry` in the `sec` subsystem. You can dynamically configure these attributes by using the `sysconfig` command or use the `sysconfigdb` command to configure these attributes in the `sysconfigtab` file.

The value of the `ufs-proplist-max-entry` attribute must be larger than the `ufs-sec-proplist-max-entry` attribute by at least enough space to hold a property list element header. The `sysconfig` utility automatically adjusts `ufs-proplist-max-entry` to achieve this. The default value of `ufs-proplist-max-entry` is 8192 bytes. See `sysconfig(8)`, `sysconfigdb(8)`, and `sysconfigtab(4)` for more information.

2.3.1 Enabling and Disabling ACLs

The ACL subsystem is shipped as part of the base system, but base system components do not require ACLs for current operations and no files are shipped with ACLs on them. If you want to use ACLs or if layered products need ACL support, then you must enable ACLs.

- To display ACL information, enter:

```
# sysconfig -q sec
```
- To dynamically enable ACLs, enter:

```
# sysconfig -r sec acl_mode=enable
```

You can enable ACLs when the system starts by using the `seccnfig` utility or by following these steps:

1. Create a stanza file containing the ACL mode enable entry; for example:

```
sec:  
    acl_mode = enable
```

2. Update the `/etc/sysconfigtab` file:

```
# sysconfigdb -m -f acl_mode.stanza sec
```

To disable ACLs, enter:

```
# sysconfig -r sec acl_mode=disable
```

Note

Disabling ACLs might allow processes to access files and directories to which ACLs disallow access. It is especially important that systems that share files using NFS have a common security domain.

If you disable ACLs, you can still set ACLs on files and directories, but the ACLs will not be used for permission checking and ACL inheritance of any default ACLs will not take place for newly created files and directories.

2.3.1.1 Enabling ACLs on NFS

If you are using NFSv3 and the Tru64 UNIX NFS server has ACLs enabled, ACL access checking and ACL inheritance from the NFS clients will be done properly; however, NFS clients will not be able to set ACLs unless the following steps are complete:

1. On the NFS server, start the property list (`proplistd`) daemon:

```
# /usr/sbin/proplistd 4
```

2. On the NFS client, mount the file system with the `proplist` option by using one of the following methods:

- Add `proplist` to the options field in the `/etc/fstab` file:

```
servername:/advfs /nfs_advfs nfs rw,proplist 0 0
```

- Add the `proplist` option to the mount command:

```
# mount -o proplist servername:/advfs /nfs_advfs
```

Note

You cannot use NFSv2 when using ACLs. Due to limitations in the NFSv2 protocols, ACLs might not always be honored when using NFSv2. For more information, see NFS Flatten Mode in the `acl(4)`.

2.3.2 ACL Structure

An ACL consists of a list of ACL entries. At a minimum there are three entries:

- One for the owning user
- One for the owning group
- One for others

These entries correspond to the Tru64 UNIX permissions for the file or directory. Any command that changes these ACL entries will also change the Tru64 UNIX permissions.

The external (printable) representation of an ACL consists of comma (,) or newline-separated entries. The fields in the ACL entries are separated by colons (:). The following example shows typical ACL entries:

```
user::rwx
user:peter:r-w
user:sam:r-x
group::rwx
other:---
```

The ACL entry keywords and qualifiers are defined as follows:

- | | |
|----------------------|---|
| <code>user::</code> | A <code>user</code> entry with a NULL qualifier field defines the permissions of the user who owns the file. This entry (called an owning-user entry) is always identical to the UNIX user permission. An ACL must contain just one <code>user::</code> entry. |
| <code>user:</code> | A <code>user</code> entry with a non-NULL qualifier field defines the permissions of the user specified by the qualifier field. The qualifier field must contain either a user name or a user identification (UID). An ACL can contain zero or more <code>user:</code> entries. |
| <code>group::</code> | A <code>group</code> entry with a NULL qualifier field defines the permissions of members of the group that owns the file. This entry (called an owning-group entry) is always identical to the UNIX group permission. An ACL must contain just one <code>group::</code> entry. |
| <code>group:</code> | A <code>group</code> entry with a non-NULL qualifier field defines the permissions of members of the group specified in the qualifier field. The qualifier field must contain either a group name or a group identification (GID). An ACL can contain zero or more <code>group:</code> entries. |
| <code>other::</code> | The <code>other</code> entry is valid only with a NULL qualifier. This entry defines the permission of all users that did not match any of the other entries in the ACL. This entry is always identical to the UNIX <code>other</code> permission. An ACL must contain just one <code>other::</code> entry. |

The characters in the permissions field are the same as the characters the `ls` command displays for the Tru64 UNIX permissions and are in the same order: `r` for read access, `w` for write access, and `x` for execute or search access. When a hyphen (`-`) character is used in place of one of the other characters, it indicates denial of that access.

ACL user, group, and other entries correspond to the Tru64 UNIX permission for the file or directory. If ACLs are enabled and you use the `chmod` command to change the Tru64 UNIX permission of a file or a directory, `chmod` also makes the appropriate changes to the access ACL for the owning user, the owning group, and the other entry.

Table 2-5 describes typical ACL entries.

Table 2-5: Example ACL Entries

Entry	Matching Criteria
<code>group:acct:r--</code>	Matches all users in group <code>acct</code> and grants read permission.
<code>user:joe:rw-</code>	Matches user <code>joe</code> and grants read and write permission.
<code>user::rwx</code>	Matches owning user of object, even if owning user changes after the file is created, and grants read, write, and execute permission.
<code>group::r--</code>	Matches owning group of object, even if owning group changes after the file is created, and grants read permission.
<code>other::r--</code>	Matches all users and all groups except the owning user and group and any other users and groups listed in ACL entries. Grants read permission.

2.3.3 Access Checking with ACLs

When there is an Access ACL on a file or directory, additional access checking takes place before someone is allowed access to that file or directory. The following checks are made in the following order:

1. If the process has the superuser authority, access to the file or directory is granted. The access ACL and the Tru64 UNIX permissions are not checked.
2. If ACLs are not enabled, or they are enabled and there is no access ACL associated with the file or directory, the Tru64 UNIX permissions are checked.

3. The additional entries in the access ACL for the file or directory are checked as follows:
 - a. If the user ID (UID) of the process is for the owner of the object, the permissions in the owning `user::` entry are granted. Any other ACL entries are not checked. This is identical to using the user Tru64 UNIX permission.
 - b. If the UID of the process matches a UID listed in a `user:` entry or resolves to a user name listed in a `user:` entry, the permissions in the entry are granted. Any remaining ACL entries are not checked.
 - c. If the group ID (GID) of the process matches the GID of the file, or if one of the supplementary groups of the process matches the GID of the file, the process is granted the union of the permissions of the `group::` entry and any matching `group:` entries as described in the next list item.
 - d. If the GID of the process matches the GID of any `group:` entries, or resolves to a group name listed in any `group:` entries or if the GID or group name of any of the supplementary groups of the process match any `group:` entries of the ACL, the process is granted the union of the protections of all matching group entries. For example, for a user belonging to group `sales` and group `eng`, if the access ACL on a file grants read access to group `sales` and write access to group `eng`, the user is granted read and write access to the file.
 - e. The permissions in the `other::` entry are granted. This is identical to using the Tru64 UNIX `other` permission.

Note

A file or directory with Tru64 UNIX permission and a file or directory with an access ACL containing only the three required entries (`user::`, `group::`, and `other::`) are indistinguishable.

2.3.4 ACL Inheritance

When a file or directory is created, it might inherit ACLs from its parent directory. When a new file or directory inherits a default ACL as its access ACL, the permissions of the new file or directory are the intersection of the permissions from the default ACL and the permissions requested by the command or program that is used to create the file or directory. The `umask` is not used. The default ACLs determine what ACLs are inherited by files and subdirectories created in a parent directory, as follows:

- If a parent directory has no default ACLs:
 - A new file created in that directory is given:

ACL Type	Status
Access ACL	None

- A new subdirectory created in that directory is given:

ACL Type	Status
Access ACL	None
Default access ACL	None
Default directory ACL	None

Only Tru64 UNIX permissions are used.

- If a parent directory has a default access ACL, but no default directory ACL:

- A new file created in that directory is given:

ACL Type	Status
Access ACL	Parent's default access ACL

- A new subdirectory created in that directory is given:

ACL Type	Status
Access ACL	Parent's default access ACL
Default access ACL	Parent's default access ACL
Default directory ACL	None

- If a parent directory has no default access ACL, but does have a default directory ACL:

- A new file created in that directory is given:

ACL Type	Status
Access ACL	None

- A new subdirectory created in that directory is given:

ACL Type	Status
Access ACL	Parent's default directory ACL
Default access ACL	None
Default directory ACL	Parent's default directory ACL

- If a parent directory has both a default access ACL and a default directory ACL:

- A new file created in that directory is given:

ACL Type	Status
Access ACL	Parent's default access ACL

- A new subdirectory created in that directory is given:

ACL Type	Status
Access ACL	Parent's default directory ACL
Default access ACL	Parent's default access ACL
Default directory ACL	Parent's default directory ACL

Setting the default ACLs on a directory does not modify the ACLs on files and subdirectories that already exist in the directory.

2.3.4.1 ACL Inheritance Examples

Examples of ACL inheritance follow:

- Assume that the directory `temp` contains no default ACLs, and the following command is entered to give the directory `temp` a default access ACL:

```
% setacl -d -u user::rwx,group::r-x,other::r-x,user:jdoue:rwx\
temp
```

Any file or directory that is created within the directory `temp` now inherits the following ACL as the access ACL:

```
#
# file: temp
# owner: smith
# group: system
#
user::rwx
user:jdoue:rwx
group::r-x
other::r-x
```

- Assume that the directory `temp` contains no default ACLs, and the following command is entered to give the directory `temp` a default directory ACL:

```
% setacl -D -u user::rwx,group::r-x,other::r-x,\
user:jdoue:rwx temp
```

Any directory that is created within the directory `temp` now inherits the following ACL as the access ACL, as well as its default directory ACL:

```
#
# file: temp
```

```
# owner: smith
# group: system
#
user::rwx
user:jdoh:rwx
group::r-x
other::r-x
```

- Assume that the directory `temp` contains no default ACLs, and the following commands are entered to give the directory `temp` a default access ACL and a default directory ACL:

```
% setacl -D -u user::rwx,group::r-x,other::r-x,\
  user:jdoh:rwx temp
```

```
% setacl -d -u user::rwx,group::r-x,other::r-x,\
  user:wilson:rwx temp
```

Any directory that is created within the directory `temp` now inherits the following ACL as the access ACL as well as the default directory ACL:

```
#
# file: temp
# owner: smith
# group: system
#
user::rwx
user:jdoh:rwx
group::r-x
other::r-x
```

The following ACL would be inherited as the default access ACL:

```
#
# file: temp
# owner: smith
# group: system
#
user::rwx
user:wilson:rwx
group::r-x
other::r-x
```

Any file created in the directory `temp` now inherits the following ACL as the access ACL:

```
#
# file: temp
# owner: smith
# group: system
#
user::rwx
user:wilson:rwx
group::r-x
```

```
other::r-x
```

2.3.5 Managing ACLs

The following commands display and modify ACLs:

<code>dxsetacl</code>	A graphical interface that creates, displays, and changes the ACL for files and directories.
<code>getacl</code>	Displays the ACL for files and directories.
<code>setacl</code>	Creates, changes, and removes the ACL for files and directories.

Note

You must be the owner of the file or directory (or have superuser authority) before you can create, change, or remove its ACL.

2.3.5.1 Using the `dxsetacl` Interface

You can use the `dxsetacl` graphical interface to create, display, and change the ACL for a file or directory. The `dxsetacl` interface is located in the CDE Desktop Applications under the Applications Manager. Alternatively, you can open it from the command line by entering the following command:

```
% /usr/bin/X11/dxsetacl &
```

See `dxsetacl(8X)` and the `dxsetacl` online help for more information.

2.3.5.2 Using the `getacl` Command

You can use the `getacl` command to display the ACL for a file or directory. In Example 2–2, the `getacl file.txt` command displays the ACL for a file called `file.txt`:

Example 2–2: Displaying the ACL for a File

```
$ getacl file.txt
#
# file: file.txt
# owner: peter
# group: system
#
user::rw-
user:jdoh:rw-
group:r--
```

Example 2–2: Displaying the ACL for a File (cont.)

```
other::r--
```

If you are using the `getacl` command to display the access ACL of a file or directory and that file or directory does not have an access ACL, the Tru64 UNIX permissions are shown in ACL format.

See `getacl(1)` for more information.

2.3.5.3 Using the `setacl` Command

You can use the `setacl` command to create, change, and remove the ACL for a file or directory. In Example 2–3, the `getacl file.txt` command displays the ACL for a file called `file.txt`. The `setacl` command updates the ACL:

Example 2–3: Setting the ACL on a File

```
$ getacl file.txt
#
# file: file.txt
# owner: peter
# group: system
#
user::rw-
user:jdoh:rw-
group::r--
other::r--
$ setacl -u group::rw-,user:evan:rw- file.txt
$ getacl file.txt
#
# file: file.txt
# owner: peter
# group: system
#
user::rw-
user:jdoh:rw-
user:evan:rw-
group::rw-
other::r--
```

The owning group entry is updated to include the write permission. The `evan` user entry does not match an existing entry and is added with read and write permission. The other entries remain unchanged.

See `setacl(1)` for more information.

2.3.6 ACL Interaction with Commands and Applications

ACLs are a POSIX and System V compatible extension to UNIX based on POSIX P1003.6 Draft 13. Not all existing commands, utilities, and applications properly use or propagate ACLs, especially applications that are not POSIX compliant. If you use any command, utility or application to access or manipulate a file system object (file or directory) that has an ACL, you must check the ACL after completion to make sure that the ACL has not been removed or changed.

Many programs that modify files use the following process:

- Create the new version of the file with a temporary name.
- Delete the existing version of the file.
- Rename the new version from the temporary name to the real name.

When the file being modified has an ACL and the program does not replicate the ACL when creating the temporary version of the file, the previous procedure will delete the file's ACL, or replace it with the default access ACL of the parent directory (if it has one). If you use such a program on a file with an ACL, you must restore the ACL afterwards. This procedure also causes any hard links to be removed from the file. Some common commands that use this method of modifying files are:

- `gzip`
- `compress`
- `emacs`

A solution is to copy the original file to a temporary file, do any processing on the temporary file, then use the `cp` command without the `-p` option to copy back. This procedure retains the original ACL.

Any time that you copy a file with an ACL, you should use the `cp -p` command to properly copy the ACL and any other extended attribute (property list).

2.3.6.1 The `pax` and `tar` Commands

Both the `pax` and `tar` command archive any ACLs and other extended attributes on archived files and directories by default when you create an archive. However, when you use the `pax` or `tar` commands to extract files and directories from an archive, any ACLs on the archived files and directories are not extracted from the archive by default. In this case, if the destination directory has default ACLs defined, the files and directories extracted from the archive inherit the default ACLs (Section 2.3.4).

To restore the ACLs and property list information from the archive, use the `-p` option with the `tar` command and the `-p p` or `-p e` options with the `pax` command when extracting files and directories from the archive. The `pax` and `tar` commands store the user and group information for ACLs as UIDs and GIDs. This means that if you use the `tar -p`, `pax -p p` or `pax -p e` commands to restore an archive on a system that does not share user and group information with the source system, you might be granting unintended access to files.

There currently are no formal industry standards for ACLs and extended attributes (property lists). Thus, the extensions to the `pax` and `tar` commands to support property lists and ACLs are specific to Tru64 UNIX. Other vendor's `pax` and `tar` implementations should simply ignore the Tru64 UNIX specific extensions. However, to ensure interoperability with other vendor's systems when archiving for multivendor distribution, use the `-v` option to prevent ACLs and any other extended attributes from being archived.

2.3.6.2 Archiving Commands

The archive commands `dump`, `rdump`, `restore`, `rrestore`, `vdump`, `rvdump`, `vrestore`, and `rvrestore` always save and restore all extended attributes (property lists) including ACLs. Attempting to extract files to a directory that has a default directory ACL or a default access ACL might cause unintended ACLs to be created on the extracted files. If ACLs are enabled on the system, make sure to check all ACLs after the extraction is complete.

Auditing the System

Provided with the Tru64 UNIX operating system software is the audit subsystem. The audit subsystem records events on the system, such as file opens, file creations, logins, and print jobs submitted. Each event is stamped with the audit ID (AUID) of the user who created it, which allows all actions to be traced directly to a user. Users have no direct interaction with the audit subsystem.

This chapter contains the following information:

- Auditing overview
- Configuring the audit subsystem
- Managing the audit subsystem
- Managing audit events
- Generating and displaying audit reports
- Traditional UNIX logging tools
- Auditing in a TruCluster
- Responding to audit reports

3.1 Auditing Overview

Auditing provides a means to record the actions that were performed on the system. When a user logs in, the system creates an identity record that associates an audit ID (AUID) with their processes; the AUID remains stamped on processes regardless of the program being run. Even if the user changes their real or effective user ID (for example, by using the `su` command to become root or another user), the system still knows which authenticated user caused a specific action based on the audit id. Be aware that the reliability of the user ID is only as good as the authentication mechanism that is used to verify the user's identity.

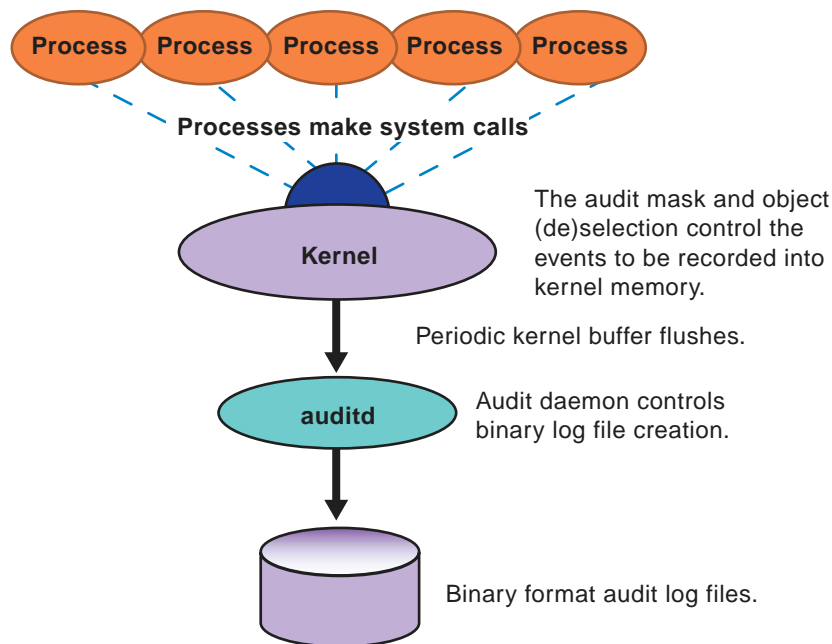
The system maintains an extensive authentication profile describing the characteristics and authorized capabilities of each user; for example, the particular login restrictions on the user.

In its simplest form, auditing consists of the following:

1. An event on the system results in the generation of an audit record. This audit record contains information about the event, such as what the event was, when it occurred, and the ID of the user who caused it.
2. The audit record is stored along with other audit records in a file called the audit log.
3. You use a utility to view information in the audit log file and to generate reports from that information.

Figure 3–1 provides an overview of the audit subsystem.

Figure 3–1: The Audit Subsystem



ZK-1582U-AI

Auditing provides you with a powerful tool for monitoring events on the system. Through auditing, you can accomplish the following:

- Discourage users from attempting to violate security. A user who knows that system events are monitored and that security violations can be tracked to the responsible individual might be dissuaded from attempting to violate security.
- Detect attempts at violations or activities indicative of probing a system for weak points. If an audit reveals failed attempts to violate system

security, you can take counter measures to lessen the likelihood of later attempts succeeding.

- Assess damage and restore the system if a break-in should occur. Careful analysis of an audit trail after a break-in can help you determine what occurred during the security violation and what steps are needed to return the system to its original state. It also allows you to take steps to prevent similar break-ins in the future.
- Evaluating and debugging application software. The capability of audit to monitor the completion status and arguments of system calls for a designated process provides the ability to assess what is going on inside an application. You can also determine which files the application is attempting to access even if you do not have the source code.

It is important that you inform users of the purpose and, in general terms, the nature of the auditing performed on the system. Present auditing in a positive light, as a tool to help protect the users' files and their access to system resources. This helps minimize any resentment; users who are openly told that their system is regularly audited are less likely to feel as though they are being spied upon. For those users who might be tempted to violate security, knowledge that activities are monitored can be a powerful deterrent.

3.1.1 Audit Files

The following lists the files used by the audit subsystem.

`/sbin/init.d/audit`

Startup and shutdown script for the audit subsystem.

`/var/audit/auditlog.hostname.nnn`

Default log file. The Tru64 UNIX audit subsystem can record events in any file at any location that you specify.

hostname is the name of the system that generated the audit log.

nnn is a generation number between 000 and 999.

`/var/adm/syslog.dated/current/daemon.log`

Default log file for status messages from the audit subsystem.

`/etc/rc.config.common`

Contains the current system defaults for the audit subsystem.

`/etc/sec/audit_events`

Contains a listing of all the system events that can have security relevance.

This file can be used as input to the `auditmask` command, which controls which events are audited on the system.

`/etc/sec/site_events`

A file defining site-specific audit events. It supports integrating application specific audits into the audit subsystem.

`/etc/sec/event_aliases`

A file containing a list of aliases that represent sets of events that can be audited.

`/etc/sec/auditmask_style`

A file defining audit style flags for profiles.

`/etc/sec/file_objects/*`

Directory containing lists of filenames to monitor for profiles.

`/etc/sec/rc_audit_events`

A file containing the list of audit events to monitor. The list is loaded at system startup, and the file is pointed to by the runtime configuration variable `AUDITMASK_FLAG` in `/etc/rc.config` or `/etc/rc.config.common`.

`/etc/sec/fs_objects`

A list of files that are subject to audit selection or deselection.

`/etc/sec/auditd_loc`

A list of alternate paths and hosts where audit logs can be stored if the current location becomes unavailable or full.

`/etc/sec/auditd_clients`

A list of the remote hosts that can send their audit data to be stored on the local system.

`/cluster/members/{memb}/dev/audit`

A CDSL required for audit on clustered systems. Points to a device associated with the audit subsystem.

```
/cluster/members/{memb}/dev/.audit/audS
```

A CDSL required for audit on clustered systems. Points to a device associated with the audit subsystem.

3.1.2 Audit Tools

You can manage the audit subsystem by using:

- A command-line interface
- A graphic interface that provides point-and-click operation and on-line help.

3.1.2.1 Command-Line Interface

The following commands are audit subsystem commands. You must be the root user to use these commands. For more information on a command, see its associated reference page.

<code>sysman auditconfig</code>	Interactively sets up the audit environment on your system including configuring the location for the audit logs, the action that the audit subsystem takes if file space is exhausted, how long the audit logs are held (forever if specified) on your system before being deleted to free disk space, and if you will be auditing across the network.
<code>auditmask</code>	Selects events for inclusion in the audit log or displays a list of events currently being recorded in the audit log. Changes made to the audit subsystem with the <code>auditmask</code> command are temporary and are reset on a system reboot.
<code>audgen</code>	Generates an audit event record from a privileged program or a script containing a message of your choice.
<code>auditd</code>	Activates the auditing daemon (turns on auditing), administers audit log storage, and configures the audit daemon.

Changes made to the audit subsystem with the `auditd` command are temporary and are reset on a system reboot.

`audit_tool`

Selectively extracts information from the audit log and presents it in a readable form.

3.1.2.2 Graphical Interface

You access the graphical interface to the audit subsystem by using the CDE Dashboard as follows:

System Applications→Daily Administration→Audit Manager

You can not use the graphical interface to configure the kernel for auditing and establishing audit log management parameters. You must use the `sysman auditconfig` command.

3.1.3 Audit Masks

Audit masks determine which events are audited and if those events are audited for successful occurrences, failed occurrences, or successful and failed occurrences. There are two kinds of audit masks as follows:

- The system audit mask, which includes the following events:
 - A system call name defined in the `/etc/sec/audit_events` file
 - A trusted event name defined in `audit.h`
 - A site-defined name in the `/etc/sec/site_events` file
 - An alias defined in the `/etc/sec/event_aliases` file
- The process audit mask, which applies to a specific process.

A special case of the process mask is the login process mask, which is only available when the Enhanced Security subset is installed and active. The login process mask is set for a user login process. When the user logs in, that audit mask is applied to the login process, and all offspring processes inherit that audit mask. See Appendix A for more information about Enhanced Security.

3.1.3.1 System Calls

All security-relevant system calls can generate audit data, but when there is no security relevance, some system calls do not generate audit data. The conditions under which a particular system call does not generate an audit data are described in Table 3-1.

Table 3–1: System Calls Not Always Audited

System Call	Cause for No Audit Record
<code>close()</code>	The system call failed because it was passed an invalid file descriptor.
<code>dup2()</code>	The system call failed because it was passed an invalid file descriptor.
<code>execv()</code> , <code>execve()</code> , <code>exec_with_loader()</code>	Failures triggered by failed <code>namei()</code> lookups, failures to terminate threads, aborts from handler callouts.
<code>fcntl()*</code>	The system call failed because it was passed an invalid file descriptor.
<code>ioctl()*</code>	The system call failed because it was passed an invalid file descriptor.
<code>msfs_syscall()</code>	Any system call failure.
<code>priocntlse()</code>	An invalid process was specified (ESRCH), or the call did not modify another process.
<code>proplist_syscall()</code>	Failures on <code>copyin()</code> of system call arguments.
<code>reboot()</code>	Successful reboots are not audited. The <code>reboot()</code> call does not return from a successful reboot.
<code>security()</code>	No audit record for the <code>getluid</code> option. This option has no security relevance, and if it were audited, many audit records of no use would result.
<code>swapctl()</code>	Only <code>SC_ADD</code> forms of the call are audited. Other forms have no security relevance.
<code>uadmin()</code>	Only a failed <code>A_REBOOT</code> or <code>A_SHUTDOWN</code> is audited. In other cases, the system is rebooted and the system call does not return.

The system calls marked with an asterisk (*) typically generate audit data only for security-relevant options. When executing processes from `auditmask` with the `-e` or `-E` flag, however, all options generate audit data.

Note the following aspects of system call auditing:

- Some records show `NOTE: uid changed`. This typically occurs in `SETUID` events, but may be seen anywhere when one thread changes the UID for all threads in a process (task).
- Audit records contain `vnode` information and the file type of the object of the operation. So, for example, if a `chmod` command is specified for a symbolic link, the actual object referenced by the link is described.
- By design, some system calls can fail and not generate an audit record for the failure if the failure is not security-relevant. See Table 3–1 for a list of the calls.

- Only TIOCSTI operations are audited for the `ioctl` system call.
- Only `F_DUPFD`, `F_SETTIMES`, and `F_CNVT` are audited for the `fcntl` system call.
- If the process audit control flag is set to `USR`, all `ioctl` and `fcntl` system calls are audited.

3.1.3.2 Trusted Events

A trusted event is an event that is associated with a security protection mechanism; it does not always correspond directly to a system call. A list of the trusted events follows:

<code>audit_daemon_exit</code>	Indicates that the audit daemon exited abnormally. This occurs only when there is insufficient memory available during initialization of the audit daemon. The exit is recorded in the new audit log, and a message is displayed on the designated audit console.
<code>audit_log_change</code>	Indicates that the audit daemon closed the current audit log and began writing a new log (for example, in response to the <code>auditd -x</code> command). The change in logs is recorded in the current audit log, and a message is displayed on the designated audit console.
<code>audit_log_create</code>	Indicates that a new audit log was created in response to the removal of the current log file. The new file has the generation number of the lost log file incremented by 1. The creation of the new log is recorded at the beginning of the new audit log, and a message is displayed on the designated audit console.
<code>audit_log_overwrite</code>	Indicates that the audit daemon began overwriting the current audit log as you specified with the <code>-o overwrite</code> option to <code>auditd</code> . The overwrite is recorded at the beginning of the newly overwritten audit log, and a message is displayed on the designated audit console.

audit_reboot	Indicates that the audit daemon initiated a system reboot (as a result of an overflow of the log) as you specified with the <code>-o halt</code> option to <code>auditd</code> . The reboot is recorded at the end of the current audit log, and a message is displayed on the designated audit console before the reboot occurs.
auditconfig	Indicates that the <code>-o changeloc</code> option to the <code>auditd</code> command was used to change the specified overflow action. The change in the audit setup is recorded in the current audit log.
audit_start	Indicates that the audit daemon has been started.
audit_stop	Indicates that the audit daemon was killed normally (typically, with the <code>-k</code> option to <code>auditd</code>). The shutdown is recorded at the end of the current audit log, and a message is displayed on the designated audit console when the shutdown occurs.
audit_suspend	Indicates that the audit daemon suspended auditing (as a result of an overflow of the log) as you specified with the <code>-o suspend</code> option to <code>auditd</code> . The suspension is recorded in the current audit log, and a message is displayed on the designated audit console.
audit_xmit_fail	Indicates that the audit daemon was sending audit records across a network and the transmission failed. The failure is recorded in the local log specified as the next path in <code>/etc/sec/auditd_loc</code> (with <code>auditd -r</code>) or the default local path (<code>/var/adm</code>).
audgen	A command-line interface to the <code>audgen</code> routine.

<code>auth_event</code>	An event associated with user-authentication and the management of user accounts occurred. Trusted <code>auth_events</code> include <code>passwd</code> , <code>su</code> , <code>rsh</code> , and <code>login</code> . The event is recorded in the current audit log.
<code>login</code>	A user attempted to log in to the system.
<code>logout</code>	A user logged out of the system.

3.1.3.3 Site-Defined Events

You can define your own audit events (referred to as site-defined events). This is useful if you want applications to generate records specific to their activities.

Trusted application software can generate data for the site-defined events and subevents. The data can be included in the audit logs with the system's audit data or stored in application-specific logs.

To define site events, you must create an `/etc/sec/site_events` file and add in it the event names and event numbers for the system's site events. The `site_events` file has one entry for each site event that can contain any number of subevents.

The lowest allowed site event number is `MIN_SITE_EVENT`, which is defined in `<sys/audit.h>`. Typically, the number is 2048. By default, up to 64 site events can be defined. However, this upper limit can be increased up to a maximum 1,048,576.

To change the upper limit for the allowed number of site events, add an entry in the `/etc/sysconfigtab` file, then reboot the system. For example, to allow up to 5000 site-defined events, use the `sysconfigdb` command to add the following lines to `/etc/sysconfigtab` file, then reboot the system:

```
sec:
    audit-site-events=5000
```

After the `/etc/sec/site_events` file is created, applications can use the `audgen1()` library routine to generate application-specific audit data. See `audgen1(3)` for more information.

3.1.3.4 Event Alias

An event alias groups multiple audit events under a single name that you select for auditing. An event alias can include a system call, trusted event, site event, or another alias. You use the following format to define your

event aliases in the `/etc/sec/event_aliases` file after the default Tru64 UNIX event aliases:

```
[alias: event[:success:failure] [event[:success:failure] ...]]
```

Continuation lines are allowed. See the `/etc/sec/event_aliases` file for more information.

3.1.4 Audit Records

The output of audit records include, at a minimum, the following data. The only exception is that if a login is not completed, `audit_id:` does not appear or if the audit id has not been set for the process (this is the normal state for some system processes).

```
audit_id:          ruid/euid:
pid:              ppid:                  cttydev:
event:
result:
ip address:
timestamp:
```

`audit_id:` The Audit ID (AUID). The AUID is associated with a user at login and should remain unchanged throughout the login session. Processes started by the user will have the AUID associated with them. The AUID is not affected by use of the `su` command to change user IDs.

AUID and LUID (login UID) are synonyms.

Note that a malicious process that gains root access can use system calls to change its audit ID. To monitor such a privileged process, enable monitoring of the security system calls. A change of the audit uid is identified in the audit record for the `security()` syscall by the first argument being `0x3`. Both the original and new audit IDs are available from the audit record. The original audit ID is saved as the result code (return value) and the new audit ID is the argument.

`ruid/euid:` The real user ID (RUID) and the effective user ID (EUID).

<code>pid:</code>	The process ID.
<code>ppid:</code>	The parent process ID (PPID). Useful for tracing back through a list of processes to the originating event and its associated RUID and AUID.
<code>cttydev:</code>	The device on which the event occurred. The record reports the major and minor numbers.
<code>event:</code>	The name of the event (typically, either the name of a system call or the name of a trusted event).
<code>result: or error:</code>	<p>If the event succeeds, the result. Often the result is 0. But some system calls return a different value. For example <code>write()</code> returns the number of bytes written.</p> <p>In the case of an error, the audit record for a system call returns the error message and number. For example,</p> <pre>error: Not owner (1)</pre> <p>Audit records for trusted events can have additional error messages.</p>
<code>ip address:</code>	The IP address of the system on which event occurred.
<code>timestamp:</code>	The date and time of the event.

3.1.4.1 Additional Entries in Audit Records

Most entries in an audit record for a system call are arguments to that system call. The following list describes many of the labels for entries that can be associated with an audit record.

Labels are context sensitive. That is, their meaning can depend on the type of audit record in which they appear. For example, in the audit record for `mmap()`, `flag:` indicates an attribute of the mapped region. But in an audit record for `audcntl()`, `flag:` is a number passed with a HABITAT request.

Commands entered on the command line by users appear as arguments to `char param:` in audit reports. For example, if a user copies the file `august_report` to a file named `sept_report`, the audit record includes the following:

```
event:  execve
char param:  /usr/bin/cp
char param:  cp august_report sept_report
```

In the context of a given audit record, interpreting the entries is a straightforward matter. If questions do arise, the reference page for the system call being audited will help clarify the report.

The following is a list of the audit record fields and the associated explanations of the fields:

<code>address:</code>	Memory address, typically an argument to <code>mmap()</code> .
<code>char param:</code>	A character string. The string can be the argument to an event or some other information relevant to the event. For example: <pre>event: open char param: /etc/zoneinfo/localtime</pre>
<code>cmd name:</code>	The name of the process which generated this event (arg 0 of the <code>exec</code> command that started the process).
<code>cntl flag:</code>	A control flag. For example, one of the flags to an <code>audcntl()</code> request.
<code>descriptor:</code>	A file descriptor. If state-dependent information is available, the actual file name and the descriptor.
<code>devname:</code>	A device name.
<code>directory:</code>	The name of the current directory.
<code>flag:</code>	A flag argument to a system call. For example, in the context of <code>mmap()</code> , it specifies the attributes of the mapped region. In the context of <code>audcntl()</code> , it is the number of a system call, and it is passed with one of the <code>HABITAT</code> requests.

flags:	Arguments passed to system calls as flags. For example, in <code>open()</code> , the value passed as the <code>oflag</code> argument.
gid:	The group ID.
home dir:	The home directory.
hostname:	A host name.
inode id:	An inode number. Along with <code>inode dev</code> , part of the descriptor information recorded for audit records involving activities with files.
inode dev:	The major and minor inode device numbers.
int param:	An integer value. For example, in <code>setpgid()</code> , the <code>process_id</code> argument.
len:	An argument to <code>mmap()</code> . The length of a region of memory.
login name:	The login user name.
long param:	A value of type <code>long</code> .
mask:	A mask argument. For example, in <code>audcntl()</code> , the value passed with a <code>SET_SYS_AMASK</code> request.
object mode:	The protection mode of an object. For example, in <code>open()</code> , the mode of the file being opened.
operation:	A request to <code>audcntl()</code> , such as <code>SET_SITEMASK</code> .
pgrp:	A process group ID. For example, the <code>process_group_id</code> argument to <code>setpgrp()</code> .
procname:	The process name associated with a PID.
prot:	An argument to <code>mmap()</code> . The protections on a region of memory.

req mode:	The request mode. For example, in an <code>open()</code> <code>O_CREAT</code> , the protection mode for the new file.
request:	The security action requested in a call to <code>security()</code> .
shell:	The user's shell program. Typically this record element appears in <code>login</code> event records.
username:	The user name associated with the event. If the <code>audit_tool -w</code> option or the Audit Manager Translate UID/GIDs to Local Names selection was used to generate the audit report, then the user name might appear in parentheses. Parentheses indicate that the audit reduction tool had to use the <code>getpw()</code> routine to look up the user name in <code>/etc/passwd</code> . This happens in cases where the user name was not associated with the RUID at login time (for example, logins were not included among the audited events). See Section 3.5.3 for a description of dependencies among audit events.

3.2 Configuring the Audit Subsystem

Auditing must be built into your kernel. If the audit subsystem is not configured into your system's kernel, the audit configuration procedure guides you through a kernel rebuild to include the audit subsystem.

In addition to system-wide auditing, you can specify auditing of events on a user-by-user basis by including the Enhanced Security subset `OSFC2SECnnn`. The `dxaudit` graphical interface for audit requires the software subset `OSFXC2SECnnn`. The `nnn` represents the Tru64 UNIX version number. See *Release Notes* for the current version number. Enter the following command to determine if these subsets have been installed:

```
# setld -i | grep C2SEC*
OSFC2SECnnn installed Enhanced Security (System Administration)
OSFXC2SECnnn installed Enhanced Security GUI
(System Administration)
```

See *Installation Guide* and `setld(8)` for information on installing subsets.

You configure the audit subsystem configuration in the following parts:

1. Configure the kernel for auditing and establishing audit log management parameters. In this part you select the location for the

audit logs, the action that audit takes if file space is exhausted, for how long the audit logs are held (forever if specified) on your system before being deleted to free disk space, and if you will be centralizing audit data storage.

Centralizing audit data storage is an option if you have computers linked in a TCP/IP network and you want to centrally store the audit data from those systems on a remote host (audit hub) that has a higher security level. Centralizing audit data storage is an alternative to storing audit data on a local system. You can configure centralizing audit data storage during the configuration process or later as described in Section 3.2.1.

2. Select the events to audit.

The following steps show you how to configure the audit subsystem:

1. In part one of configuring the audit subsystem, enter the `sysman auditconfig` command to start configuring audit. If the audit subsystem is not configured into your system's kernel, `sysman auditconfig` guides you through a kernel rebuild.
 - a. Click Yes on the Welcome screen to begin configuring audit.
 - b. Click OK on the Information screen.
 - c. Specify the audit log location. You can specify one of the following locations:
 - The default location (`/var/audit/auditlog.hostname.nnn`)
 - Another location on the system.
 - If this system will store its audit data on an audit hub, enter the host name followed by a colon (`hostname:`) of the audit hub.Click Next on the Log Pathname screen.
 - d. Accept the default for the action if log space is exhausted (Suspend auditing until space becomes available) by clicking Next on the Action On Log File Space Exhaustion screen.
 - e. Accept the default for the audit log lifespan in months (forever), and the hour of deletion (3) on the Log File Lifespan screen.
 - f. If you want this system to be the audit hub on which other network system store their audit data, enable the Remote clients can log audit information to the system option, then click on the Configure remote client button. A Configure remote client window is displayed in which you enter the fully qualified domain name for each client that will log audit data.

Accept the default for audit console message destination (`syslog-/var/adm/syslog.dated/current/daemon.log`) by clicking **Finish Part One** on the **Advanced Audit Options** screen.

- g. Proceed to part two by clicking **Yes** to the question **Do you wish to proceed to part two?** on the **Audit Event Information** screen.

2. In part two of configuring the audit subsystem, you must pick from a list of six profiles described in the following table. An audit profile provides a method to consolidate the parameters that define what is audited by the audit subsystem and to group that information under a profile.

Profile	Description
Desktop	Suggested minimal auditing for a single user system.
NIS_server	Suggested auditing for a system used as a NIS server.
Networked_system	Suggested auditing for a system on a network.
Server	Suggested auditing for a system that is used as a server for network-based applications.
Timesharing	Suggested minimal auditing for a system that is used to support multiple interactive users.
Timesharing_extended_audit	Extended auditing for a system that is used to support multiple interactive users.

A profile is made up of the following three parts:

Audit style information Located in `/etc/sec/auditmask_style` under the `Profile:` label. See `auditmask(8)` under the `-c` option for a description of the valid audit style characteristics.

Audit events to be monitored Located in `/etc/sec/event_aliases` under the `Profile:` label.

Files to be monitored The list of files is in the `/etc/sec/file_objects/Profile` file. Monitoring is performed by setting the appropriate object selection and deselection flags. Information in `/etc/sec/auditmask_style` determines which flags are set. This file is optional and does not need to exist if the audit style defined

in `/etc/sec/auditmask_style` does not include `obj_sel` or `obj_desel`.

- a. Select the profile closest to the configuration of your system and click Next on the Audit Event Category Selection screen.

Note that use of the CTRL key allows the selection of more than one category; for example, `Desktop` and `Networked_system` for a desktop system on a network. While multiple selections are possible, use the minimum profile that meets your needs.

Accept the default list of events to be audited by clicking Next on the Advanced User Audit Event Modification/Deletion screen.

- b. Some profiles include this step, otherwise skip to the next step. Accept the default list of files to be audited by clicking Next on the UNIX File System Objects screen. This menu item does not appear for all profiles.
- c. Accept the options for audit events that have been set as a result of the profile that were selected by clicking Finish on the Advanced Options screen.

3. Complete the audit set up by clicking OK on the Audit Configuration Complete screen.

When you have completed the selections, the `auditconfig` does the following:

- Modifies `AUDITMASK_FLAG` in the `/etc/rc.config.common` file.
- Writes the `/etc/sec/rc_audit_events` file.
- Writes the `/etc/sec/fs_objects` file.

4. Check the configuration of the audit daemon by entering the `auditd -w` command. If you accepted the system default, your configuration looks like the following:

```
# auditd -w

Audit data and msgs:
-l) audit data destination           = /var/audit/auditlog.hostname.001 [1]
-c) audit console messages          = syslog [2]

Network:
-s) network audit server status (toggle) = off [3]
-t) connection timeout value (sec)      = 4

Overflow control:
-f) % free space before overflow condition = 10 [4]
-o) action to take on overflow          = suspend audit [5]
```

[1] The name of the audit log.

- ❑ 2 The location of audit subsystem messages. Changes in the status of the audit subsystem are recorded here.
- ❑ 3 Auditing across the network is not enabled (remote clients may not log to this system).
- ❑ 4 Percent of remaining free space in the file system that will trigger an audit overflow condition. In this case, 10 percent.
If the file system containing the audit log becomes 90 percent full, the audit subsystem takes an overflow action.
- ❑ 5 The overflow action is to suspend auditing until storage space is available.
In place of *alternate file systems*, any directory names specified in `/etc/sec/auditd_loc` are displayed.

3.2.1 Centralizing Audit Data Storage

If you have computers linked in a TCP/IP network, you can run the audit daemon on multiple systems and have the audit data sent to the audit daemon on remote system (audit hub) to store the information on a single system with a higher security level as an alternative to storing the audit data on the local system.

To prevent corruption of audit data, the audit data from remote systems is stored in its own dedicated audit log file on the audit hub.

When you use the audit tool to retrieve data from these logs of audit data from remote systems, the first and last audit log entries may be fragments rather than complete entries. This can happen if the communications channel is not cleanly terminated or if the `auditd` daemon remotely receiving the data is forced to switch log files. This is because remote audit information is sent in a continuous stream to the audit hub, rather than as discrete audit entries.

The audit tool notifies you when it encounters a fragmented entry. This does not affect the retrieval of other records from the audit log.

Centralizing audit data storage requires configuration on the audit hub and on the remote systems that will direct audit data to the audit hub. If you did not configure for centralized audit data storage when you configure the audit subsystem, you can use the commands described in the following sections to do so.

3.2.1.1 Configuring Centralized Audit Data Storage on the Audit Hub

Follow these steps to configure the audit hub for centralized audit data storage:

1. On the host that is to be the central collecting point for audit information (the audit hub), create the file `/etc/sec/auditd_clients`. In this file, enter on a separate line the name of each remote host that will send audit data to the audit hub.
2. Enter the following command to enable the audit hub to receive audit data from audit daemons on remote hosts identified in the `/etc/sec/auditd_clients` file:

```
# /usr/sbin/auditd -s
```

3. Set the options for remote audit daemons as follows:

```
auditd [-p ID_of_daemon_serving_remote_host  
options_for_remote_daemon]
```

For example, to set the audit log location to `/var/audit/NYC_Sys1` for the remote host served by the audit daemon with ID 6, enter:

```
# /usr/sbin/auditd -p 6 -l /var/audit/NYC_Sys1
```

The IDs of audit daemons serving remote hosts are integers. To display the audit daemons IDs, enter:

```
# /usr/sbin/auditd -w
```

The previous command displays the current options. No IDs are displayed unless at least one child audit daemon exists. If the `-p` option is not specified when running with more than one audit daemon, the master daemon (accepting audit data for the local system) handles the request.

When you use the audit tool to retrieve data from these logs of audit data from remote systems, the first and last audit log entries may be fragments rather than complete entries. This can happen if the communications channel is not cleanly terminated or if the `auditd` remotely receiving the data is forced to switch log files. This is because remote audit information is fed in a continuous stream to the audit hub, rather than as discrete audit entries.

The audit tool notifies you when it encounters a fragmented entry. This does not affect the retrieval of other records from the audit log.

3.2.1.2 Configuring Remote System Audit Data Storage on an Audit Hub

On each remote system, enter the following command to direct the audit data to the audit hub:

```
# /usr/sbin/auditd -l audit_hub_name:
```

If the remote system cannot communicate with the audit hub, the audit daemon on the remote host stores the audit data locally, as specified with the `-o` and `-r` flags to `auditd`.

3.3 Managing the Audit Subsystem

The following sections describe how to:

- Change the audit subsystem startup default values
- Start, stop, and suspend the audit daemon
- Archive audit logs
- Recover audit data

3.3.1 Changing the Audit Subsystem Startup Defaults

The system startup defaults for the audit subsystem are stored in the `AUDITMASK_FLAG` variable in the `/etc/rc.config.common` file. This field is used to pass command-line arguments to `auditmask` at system startup. The `AUDITMASK_FLAG` variable sets the audit style flags and sets `auditmask` to read the events selected for auditing from the `/etc/sec/rc_audit_events` file. The `/etc/sec/fs_objects` file contains a list of the files that have property lists that were modified during the audit subsystem configuration to allow the file to be monitored with object selection and deselection.

You can modify the `AUDITMASK_FLAG` variable in the `/etc/rc.config.common` file by using the `rcmgr` command, the `sysman auditconfig` command, or the Audit Manager through the CDE Dashboard.

You can modify the `/etc/sec/rc_audit_events` file by using a text editor, the `sysman auditconfig` command, or the Audit Manager through the CDE Dashboard.

3.3.2 Starting, Stopping, and Suspending the Audit Daemon

To start the audit daemon, enter:

```
# auditd [-l log_pathname]
```

To stop the audit daemon, enter:

```
# auditd -dk
```

To stop the audit daemon in a TruCluster Server environment, enter the following commands:

```
# auditmask [-cluster] -n
# auditd [-cluster] -dk
```

To suspend the audit daemon, enter:

```
# auditd -o suspend | halt
```

3.3.3 Archiving Audit Logs

After the audit log data has been collected, it is important to have a structured archival procedure for the audit log files. Many of the applications that use the audit log data need to locate and analyze data that can be several months old.

The `sysman auditconfig` utility provides a method for periodically deleting the binary audit logs using a root `cron` job. This feature is disabled by default. If enabled, you should provide for periodic audit log file backup before each `cron` job run.

The Audit log lifespan in months dialog box reflects both the monthly cycle on which the deletion will execute and the criteria for the deletion. The deletion always occurs on the first of the month; the hour is adjustable, but the minute is set to zero. A binary audit log is deleted when all the audit events it contains are older than the deletion execution date minus the Audit log lifespan in months.

For example, to create a `cron` job that runs on the second, fourth, and sixth months on the first day of the month at 3:00 a.m., enter the following:

```
"Audit log lifespan in months" = "every second month"
"Hour of deletion(0-23)" = "3"
```

Binary log files that contain entries that are all at least two months old are deleted at `cron` execution.

3.3.4 Recovering Audit Data

In the event your system encounters a panic situation, the `crashdc` utility extracts any audit data left in the system at the time of the panic and places it in the crash directory, in the `audit-data.n` file

The `n` is the crash number. If no audit data was present, the file is not created. You can use the `audit_tool` command to display the contents of the `audit-data.n` file.

It is possible for some audit records to appear in both the `auditlog` file and in the `audit-data.n` file. It is also possible that the first audit record in the `audit-data.n` file may not be complete. The `audit_tool` command

marks this as a corrupted record. In this case, the audit record was written to the regular `auditlog` file.

3.4 Managing Audit Events

The following sections describe how to:

- Display the audit mask
- Identify events that can be audited on the system
- Enable audit events
- Disable audit events
- Trace a process
- Audit file access operations

3.4.1 Displaying the Audit Mask

To display the current audit mask and whether events are being audited under successful or failed occurrences or both, you can use:

- The `dxaudit` program found on the CDE Dashboard, as follows:
Application Manager→Daily Administration→Audit Manager→Collection
- The `auditmask` command:

```
# auditmask
```

If the keyword `succeed` is present, successful occurrences of that event will be audited; if the keyword `fail` is present, failed occurrences of that event will be audited; if both are present, successful and failed occurrences will be audited; if neither keyword is present, that event will not be audited.

3.4.2 Identifying Events that can be Audited on the System

Follow these steps to create the file `all_auditable_events` that contains a list of all possible auditable events:

1. Save the current audit mask to the file `original_audit_mask`:

```
# auditmask > original_audit_mask
```
2. Set the audit mask to all events:

```
# auditmask -f
```

The `auditmask -f` command may slow down your system.
3. Save the audit mask to the file `all_auditable_events`:

```
# auditmask > all_auditable_events
```

4. Set the audit mask to no events:
`# auditmask -n`
5. Restore the audit mask to the original events:
`# auditmask < original_audit_mask`

3.4.3 Enabling Audit Events

The audit subsystem is capable of generating and collecting large amounts of data. Therefore, it is important to place reasonable limits on the data being generated by the audit subsystem and to monitor the growth of the audit log. Monitoring ensures that audit logs do not grow to fill the file system and cause a denial of service situation.

You selected the events to audit when you configured the audit subsystem and chose a profile; however you can change the audit mask at any time by selecting and deselecting events.

To select events to audit, you can use:

- The `dxaudit` program found on the CDE Dashboard, as follows:
Application Manager→Daily Administration→Audit Manager→Collection
- The `auditmask` command as follows:

```
/usr/sbin/auditmask [flags] [event[:succeed:fail]] [-e,E  
file[args...]] [<event_list]
```

The `auditmask` command with event arguments sets the system audit mask. This is a cumulative operation, so it is possible to turn on or off audit for one set of events, then turn on or off audit for a second set of events without changing the first set of events (except for the intersection between the two sets).

Command line arguments to the `auditmask` command can include one or more events, each with an optional field `:succeed:fail`, where `succeed` is either 0 to specify no auditing of successful occurrences of the event or 1 to specify auditing of successful occurrences of the event; and `fail` is either 0 to specify no auditing of failed occurrences of the event or 1 to specify auditing of failed occurrences of the event.

The following are some examples of using the `auditmask` command to select events to audit. See `auditmask(8)` for more information.

- To select all events, enter:

```
# auditmask -f
```

This option causes a significant degradation in system performance.

- To select the successes and failures of an event, enter one of the following commands:


```
# auditmask eventname
# auditmask eventname:1:1
```
- To select only successes of an event, enter:


```
# auditmask eventname:1:0
```
- To select only failures of an event, enter:


```
# auditmask eventname:0:1
```
- To select events in a specified file, enter:


```
# auditmask < filename
```
- To select processes generated by a specific user, enter:


```
# auditmask -a AUID-of-user
```
- To select file activity only for certain files (Object Selection), enter:


```
# auditmask -x filename [:1|:0]
```
- To select an event if either the system mask or process mask specifies the event, enter:


```
# auditmask -c or
```
- To select an event only if both the system mask and process mask designate the event, enter:


```
# auditmask -c and
```
- To select an event based only on the process audit mask, enter:


```
# auditmask -c usr
```

3.4.4 Disabling Audit Events

To disable auditing for:

- All events, enter:


```
# auditmask -n
```
- A specific event, enter:


```
# auditmask eventname:0:0
```
- File activity for certain files (Object Deselection), enter:


```
# auditmask -y filename
```

3.4.5 Tracing a Process

The `auditmask` utility can be used to adjust the audit characteristics of the system, of a single process, or all processes associated with a user's audit ID (AUID). The generation of an audit record is a function of the system audit mask, the process audit mask, and the process `audcnt1` flag. Each audit mask is a bitmap for all the events.

The process `audcnt1` flag specifies:

- `or` Audit if the event is in either the system or the process audit mask. The default `audcnt1` flag setting is `or`.
- `and` Audit if the event is in both the system and the process audit mask.
- `off` No auditing done for this process.
- `usr` Audit if the event is in the process audit mask.

Specifying which events are audited can be done by naming a set of system calls and/or alias names. Aliases are defined in (and may be added to) the `/etc/sec/event_aliases` file. You can edit this file to add or delete system calls. An optional extension can be used to distinguish between successful occurrences and/or failed occurrences of any event. For example:

`open:1:0` Specifies successful occurrences of the `open()` call.

`open:0:1` Specifies failed occurrences of the `open()` call.

The following examples demonstrate the use of the `auditmask` utility for various purposes. These examples modify the process audit mask. Unless specified, the process `audcnt1` flag remains at its default setting of `or`.

- To execute command arguments and audit everything for the newly created process:

```
# auditmask -E command args
```
- To execute command arguments, audit failed `open()` system calls, and successful `ipc` events (defined in `/etc/sec/event_aliases`) for the newly created process:

```
# auditmask open:0:1 ipc:1:0 -e command args
```
- For process ID (PID) 999, audit all (`-f`) events except `gettimeofday()`:

```
# auditmask -p 999 -f gettimeofday:0:0
```
- For PID 999, audit no (`-n`) events:

- `# auditmask -p 999 -n`
- For PID 999, audit `open()` and `exec()` events:
 - `# auditmask -p 999 open exec`
- For PID 999, add `exit()` to the set of events being audited:
 - `# auditmask -p 999 exit`
- For PID 999, get the set of events being audited:
 - `# auditmask -p 999`
- For PID 999, set the `audcntl` flag to `usr`:
 - `# auditmask -p 999 -c usr`

When tracing a running process, the `auditmask -c usr` command traces all options for the system calls.

- For all processes owned by the user with AUID 1123, audit all `ipc` events (the AUID is the same as the user's initial RUID):
 - `# auditmask -a 1123 ipc`
- To access the `auditmask` help option:
 - `# auditmask -h`

3.4.5.1 Displaying Trace Process Data

To display information about the trace process, enter the following command:

```
# audit_tool `auditd -dq` -B
```

The `auditd -d` option flushes data from the kernel to the audit log. The `auditd -q` option provides the name of the current audit log.

Information similar to the following is displayed:

AUID:RUID:EUID	PID	RES/(ERR)	EVENT
1123:0:0	6691	0x14	audcntl (0x7 0x0 0x14 0x0)
1123:0:0	6691	0x0	execve (/sbin/date date)
1123:0:0	6691	0x2000	getpagesize ()
1123:0:0	6691	0x2000	getpagesize ()
1123:0:0	6691	0x0	getrlimit ()
1123:0:0	6691	0x3ffc0004000	mmap (-1 0x7 0x3ffc0004000 0x12 0x2000)
1123:0:0	6691	0x0	getrlimit ()
1123:0:0	6691	0x3ffc0006000	mmap (-1 0x7 0x3ffc0006000 0x12 0x4000)
1123:0:0	6691	0x0	getuid ()
1123:0:0	6691	0x1	getgid ()
1123:0:0	6691	0xa68	read (3)
1123:0:0	6691	0x120000000	mmap (3 0x5 0x120000000 0x102 0x4000)
1123:0:0	6691	0x140000000	mmap (3 0x7 0x140000000 0x2 0x2000)
1123:0:0	6691	0x4	open (/shlib/libc.so 0x0)
1123:0:0	6691	0xa68	read (/shlib/libc.so)
1123:0:0	6691	0x3ff80080000	mmap (/shlib/libc.so 0x5 0x3ff80080000 0x2 0x10e000)
1123:0:0	6691	0x3ffc0080000	mmap (/shlib/libc.so 0x7 0x3ffc0080000 0x2 0x10000)

```

1123:0:0      6691 0x3ffc0090000 mmap ( -1 0x7 0x3ffc0090000 0x12 0x9bf0 )
1123:0:0      6691 0x0      close ( 4 )
1123:0:0      6691 0x0      stat ( /shlib/libc.so )
1123:0:0      6691 0x0      set_program_attributes ( )
1123:0:0      6691 0x0      close ( 3 )
1123:0:0      6691 0x2000   getpagesize ( )
1123:0:0      6691 0x0      obreak ( 0x14000ea10 )
1123:0:0      6691 0x0      gettimeofday ( )
1123:0:0      6691 0x3      open ( /etc/zoneinfo/localtime 0x0 )
1123:0:0      6691 0x32e   read ( /etc/zoneinfo/localtime )
1123:0:0      6691 0x0      close ( 3 )
1123:0:0      6691 0x1d    write ( 1 )
1123:0:0      6691 0x0      close ( 0 )
1123:0:0      6691 0x0      close ( 1 )
1123:0:0      6691 0x0      close ( 2 )
1123:0:0      6691 0x0      exit ( )

```

3.4.5.2 Auditing Active Processes

Example 3–1 shows the commands that you can use to investigate a process started by a user logged in as `guest`.

Example 3–1: Sample Active Auditing Session

```

# ps -uguest -o user,pid,uid,comm 1
USER      PID    UID COMMAND
guest     23561  1123 csh
guest     23563  1123 ed

# auditmask -p 23563 open exec -c or 2
# auditmask -p 23563 3
! Audited system calls:
execv          succeed  fail
exec_with_loader  succeed  fail
open           succeed  fail
execve         succeed  fail

! Audited trusted events:

! Audcntl flag: or

# auditd -d 5s -w 4
Audit data and msgs:
-l) audit data destination    = /var/audit/auditlog.hostname.001
-c) audit console messages    = /var/audit/auditd_cons
-d) audit data dump frequency = 5s

Network:
-s) network audit server status (toggle) = off
-t) connection timeout value (sec)      = 4

Overflow control:
-f) % free space before overflow condition = 10

```

Example 3–1: Sample Active Auditing Session (cont.)

```
-o) action to take on overflow      = overwrite current auditlog

# audit_tool /var/audit/auditlog.hostname.001 -Bfw [5]
USERNAME      PID      RES/(ERR)    EVENT
-----
jdoe           23563    0x4          open ( /etc/motd 0x0 )
jdoe           23563    0x4          open ( /etc/passwd 0x0 )
jdoe           23563    0x4          open ( /etc/ftpusers 0x0 )
jdoe           23563    0x4          open ( /etc/hosts 0x0 )
jdoe           23583    0x0          execve ( /usr/bin/sh sh -c ps )
jdoe           23583    0x5          open ( /usr/shlib/libc.so 0x0 )
jdoe          * 23592    0x0          execve ( /sbin/ps ps gax ) [6]
jdoe           23599    0x0          execve ( /usr/bin/sh sh -c w )
jdoe           23599    0x5          open ( /usr/shlib/libc.so 0x0 )
jdoe          * 24253    0x0          execve ( /usr/ucb/w w )
jdoe           23563    0x4          open ( savethis 0x602 0640 )

[Ctrl/C] [7]
--interrupt:  exit (y/[n])?  y
```

- [1] Find out what process user guest is running and also get the process ID and audit ID.
- [2] For PID 23563, set the auditmask to open and exec, and perform an OR operation with the system mask. Note that exec is an alias for execv, exec_with_loader, and execve.
- [3] Get the auditmask for the 23563 process.
- [4] Dump to the audit log every 5 seconds and also show the auditd configuration.
- [5] Display a continuous (-f) abbreviated (-B) audit report. Resolve AUIDs to corresponding user names (-w).
Note that the name of the audit log was gotten from the results of the auditd -w command.
- [6] The asterisk (*) indicates an operation involving setuid.
- [7] Exit the audit_tool program with a Ctrl/C (auditing continues).

See auditmask(8) for more information.

3.4.5.3 Dynamically Auditing Additional System Call Arguments

If additional system call arguments are required, you can use the dbx command to dynamically change which arguments get recorded for any

system call. System calls are defined in their related .h file located in the /usr/include/sys directory. For example, flock is system call 131 (defined in the /usr/include/sys/syscall.h file), and takes as arguments a file descriptor and an option. To audit these arguments, enter the following dbx commands:

```
(dbx) a sysent[131].aud_param[0]='c'
99
(dbx) a sysent[131].aud_param[1]='a'
97
```

The first entry in the aud_parm array corresponds to the first system call argument, the second entry corresponds to the second system call argument, and so on. The c encoding indicates a file descriptor is recorded. The a encoding indicates an integer argument is recorded. The set of encodings is described in the <sys/audit.h> file.

3.4.6 Auditing File Operations

Object selection and deselection are powerful tools for managing the growth of audit logs.

Events such as mount and reboot affect system state. Data access events, such as open and stat, act on files. Although reboot attempts might always be security relevant, not all file open events are (depending on the site security model).

Object selection and deselection let you chose which files do (selection) or do not (deselection) result in audit records when those files are objects of data access operations.

Data access operations are:

```
read, open, close, link, lseek, access,
stat, lstat, dup, open, revoke, readlink,
fstat, pre_F64_stat, pre_F64_lstat,
dup2, pre_F64_fstat, readv, pread, getdirentries,
stat, lstat, fstat, _F64_readv
```

Object selection and deselection work as follows:

Selection Object selection allows you to audit selected data access operations performed on any one of a set of designated files. Files are designated by having the object selection flag set in the file property list by the commands `auditmask -X` command or `auditmask -x` command. The same operations performed on any unflagged file will not generate audit data. For example, you can flag the /etc/passwd file and the /.rhosts file and audit the open system call. This will cause an open on the /etc/passwd fuke or the /.rhosts file to be audited while an open on /tmp/xxxxx files would not be audited.

Deselection

Object deselection allows you to audit selected data access operations which modify any one of a set of designated files. The same operations performed on any unflagged file will generate audit data whether the file is modified or not. Files are designated by having the object deselection flag set in the file property list by the `auditmask -Y` command or the `auditmask -y` command.

File open's for write or truncate access, are examples of file modification.

Audit selection and deselection do not reduce auditing of processes for which the `auditmask -c usr` command was specified (the value of the audit control flag is `AUDIT_USR`).

Use of object selection/deselection requires three steps:

1. Decide which file or files you want to apply object selection or deselection to. If you want to apply selection or deselection to more than one file, you can create a file that is a list of file names (complete path names), one file name per line.
2. Set the audit style either to Object Selection or Object Deselection as appropriate:

From Audit Manager: Collection→Audit Style

From the command line:

```
# auditmask -s obj_sel
```

or

```
# auditmask -s obj_desel
```

3. Activate object selection and deselection as follows:

Object selection for one file:

```
# auditmask -x filename
```

Object selection for a series of files listed in a file named *filelist*:

```
# auditmask -X filelist
```

Object deselection for one file:

```
# auditmask -y filename
```

Object deselection for a series of files listed in a file named *filelist*:

```
# auditmask -Y filelist
```

The following are examples of how to enable the selection of a file for audit:

```
# auditmask -s obj_sel
# auditmask -q /etc/passwd
selection: off      deselection: off  -- /etc/passwd
# auditmask -x /etc/passwd
selection: off => on  -- /etc/passwd
# auditmask -q /etc/passwd
selection: on      deselection: off  -- /etc/passwd
```

The following example shows how to deselect a list of files:

```
# auditmask -s obj_desel
# cat desel_file
/etc/motd
/etc/fstab
/etc/passwd
# auditmask -Q desel_file
selection: off      deselection: off  -- /etc/motd
selection: off      deselection: off  -- /etc/fstab
selection: off      deselection: off  -- /etc/passwd
# auditmask -Y desel_file
deselection: off => on  -- /etc/motd
deselection: off => on  -- /etc/fstab
deselection: off => on  -- /etc/passwd
```

The status of the object selection and deselection flags for a file list can be displayed from the CDE dashboard as follows:

1. Select Application Manager→Daily Administration→Audit Manager→Modify System Mask→Current→Edit →Object Selection/Deselection.
2. Under File, select `/etc/sec/fs_objects`.
3. Under File, select List of Files.
4. Settings are not applicable.
5. Click on Query.

Although the system object selection and object deselection audit styles are mutually exclusive, it is possible for any one object to be subject to both models simultaneously on different systems (across NFS). The `proplistd` daemon needs to be running to transfer attributes across NFS.

3.5 Generating and Displaying Audit Reports

By default, audit records are located in the `/var/audit` directory in an audit log file called:

```
auditlog.hostname.nnn
```

The *hostname* is the name of the local host. The *nnn* is an automatically generated numeric suffix that is appended to audit log file names. A numeric suffix is in the sequential range from 000 to 999.

From the audit log, you generate a report that contains the audit records that you want to display. To generate and display an audit report, you can use:

- The Audit Manager graphical interface from the CDE Dashboard by selecting the following options:

Sysman Applications→Daily Administration→Audit Manager→Reports→Generate Reports

When displaying logs with the Audit Manager, you create a selection file that specifies such things as the events, times of day, AUIDs and other attributes of the audit records that you want included in the report.

- The `audit_tool` command. The following are some examples of using the `audit_tool` command. See `audit_tool(8)` for more information.

Note

Because new events have been added to the audit subsystem, the `audit_tool` command running earlier version of the Tru64 UNIX operation system will not recognize new events and will display an error message for unrecognizable events.

- To generate and display a report for all activities that occurred during a certain time period, enter:

```
# audit_tool -t start -T end auditlog
```

The *start* and *end* times are in the format `yymmdd[hh[mm[ss]]]`. Hours (hh), minutes (mm), and seconds (ss) are optional.

- To generate and display a report for all failed attempts by a specific user to open files, enter:

```
# audit_tool -U username -e open:0:1 auditlog
```

If `login` is not preselected for auditing, then no user names appear in the audit log. In this case, you can still select for UIDs (`audit_tool -u UID`) or AUIDs (`audit_tool -a AUID`).

- To generate and display a report for each audit ID (AUID) in the audit log, enter:

```
# audit_tool -R auditlog
```

By default, the reports will be named `report.AUID`.

- To generate and display a report of all audit records containing the text *string* in a parameter field or associated with a descriptor, enter:

- ```
audit_tool -s string auditlog
```
- To generate and display a report of all audit records associated with a specific process ID, enter:

```
audit_tool -p PID
```
  - To generate and display a report of all audit records associated with a specific process ID and all the descendants of that process, enter:

```
audit_tool -p -PID
```

### 3.5.1 Filtering Audit Records

You can use the contents of a deselection file to filter out audit records that you do not want to see. A deselection file consists of one or more lines in the following format:

```
[hostname audit_ID RUID event pathname flag]
```

An asterisk (\*) in a field is a wildcard, which always gives a match. A string ending with a plus sign (+) matches any string that starts with the designated string. The *flag* specifies read (r) or write (w) mode for open events. For example, to filter out all open operations for read access on objects whose pathname starts with /usr/lib/, specify the following line in the file:

```
* * * open /usr/lib/+ r
```

The lines that you specify in the deselection file take precedence over other selection options. You can create multiple deselection files, but you can specify only one deselection file each time you perform audit reduction.

To display an audit report using a deselection file, enter:

```
audit_tool -d deselection_file
```

### 3.5.2 Displaying Abbreviated Audit Records

The `audit_tool -B` command generates an audit report with an abbreviated record format.

An example of an abbreviated report:

| AUID:RUID:EUID | PID  | RES/(ERR) | EVENT                                                             |
|----------------|------|-----------|-------------------------------------------------------------------|
| -1:0:0         | 2056 | 0x0       | execve (/usr/sbin/rlogind rlogind )                               |
| -1:0:0         | 2057 | 0x0       | execve (/usr/bin/login login -p -h<br>alpha1.sales.dec.com guest) |
| 1234:0:0       | 2057 | 0x0       | login (guest)                                                     |
| 1234:1234:1234 | 2057 | 0x0       | execve (/bin/sh -sh)                                              |
| 1234:1234:1234 | 2058 | 0x0       | execve (/usr/bin/stty stty dec)                                   |

Column headings for abbreviated reports:



|                |                                                                                                                                                                                                                                                             |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AUID:RUID:EUID | The audit ID, real UID, and effective UID associated with the event.                                                                                                                                                                                        |
| PID            | The process ID number.                                                                                                                                                                                                                                      |
| RES/(ERR)      | RES is the result number. Refer to the reference page for the specific system call for information about the result number.<br><br>(ERR) is the error code, if an error occurred. For a list of error codes and their meanings, see <code>errno(2)</code> . |
| EVENT          | The event and arguments appear in the last column.                                                                                                                                                                                                          |

The following information does not appear in the abbreviated report:

- User name (If you want the user name instead of the AUID:RUID:EUID, use the `audit_tool -wB` command.)
- PPID
- Device ID
- Current directory
- Inode information
- Symbolic name referenced by descriptors
- IP address
- Time stamp

See the description of the `-O` option in `audit_tool(8)` to generate a customized brief report.

### 3.5.3 Dependencies Among Audit Events

Some information in the audit log is based on previously audited events. For example, the `LOGIN` event associates a login name with a real UID (RUID). Subsequent occurrences of that RUID (for a given process) can then be associated with a login name. Such data is called state-dependent information. By default, the `audit_tool` command cross references audit record state information; you enter the `-F` option to specify not to.

The following three audit records illustrate state-dependent information. The first record shows a successful `open()` of `/etc/passwd`, returning a value of 3:

```

audit_id: 1621 ruid/euid: 0/0 (username: root)
pid: 23213 ppid: 23203 cttydev: (6,1)
procname: state_data_test
event: open
char param: /etc/passwd
flags: 2 : rdwr
vnode id: 2323 vnode dev: (8,1024) [regular file]
object mode: 0644
result: 3 (0x3)
ip address: 16.153.127.241 (alpha1.sales.dec.com)
timestamp: Wed Nov 10 17:49:59.93 2000

```

The following record shows the result of an `ftruncate()` system call for the `/etc/passwd` file with state-dependent information. The state-dependent data currently associates the file name `/etc/passwd` with descriptor 3 for this process:

```

audit_id: 1621 ruid/euid: 0/0 (username: root)
pid: 23213 ppid: 23203 cttydev: (6,1)
procname: state_data_test
event: ftruncate
vnode id: 2323 vnode dev: (8,1024) [regular file]
object mode: 0644
descriptor: /etc/passwd (3)
result: 0
ip address: 16.153.127.241 (alpha1.sales.com)
timestamp: Wed Nov 10 17:49:59.96 2000

```

If state-dependent data is not being maintained, you would see only that the `ftruncate()` system call was against descriptor 3 (vnode ID = 2323, dev = 8,1024):

```

audit_id: 1621 ruid/euid: 0/0 (username: root)
pid: 23213 ppid: 23203 cttydev: (6,1)
event: ftruncate
vnode id: 2323 vnode dev: (8,1024) [regular file]
object mode: 0644
descriptor: 3
result: 0
ip address: 16.153.127.241 (alpha1.sales.com)
timestamp: Wed Nov 10 17:49:59.96 2000

```

Table 3-2 lists state-dependent information and the audit events required to maintain it.

**Table 3–2: State-Dependent Information**

| To get this state-dependent information | Audit these events                                                                                         |
|-----------------------------------------|------------------------------------------------------------------------------------------------------------|
| login user name                         | login, logout                                                                                              |
| process name                            | execv, execve,<br>exec_with_loader, exit                                                                   |
| file info/socket info                   | open, old_open, close, dup,<br>dup2, fcntl, bind, connect,<br>accept, naccept, socket,<br>proplist_syscall |
| current directory                       | chdir, chroot, fchdir                                                                                      |
| audit status                            | audit_suspend, audit_log_cre-<br>ate, audit_log_overwrite,<br>audit_shutdown, audit_xmit_fail              |

The `exit()` system call informs `audit_tool` that it no longer needs the state-dependent information for the exiting process. This allows `audit_tool` to run faster.

If you are not interested in state-dependent data, you do not need to audit `exit()`, and use the `-F` option to the `audit_tool` program for fast mode.

See `audit_tool(8)` for more information.

## 3.6 Traditional UNIX Logging Tools

For security-relevant auditing, use the audit subsystem. Traditional UNIX operating system logging tools are available and do provide some auditing capabilities for the following categories of events:

- Local login and logouts
- File Transfer Protocol (FTP) logins
- External logins and logouts for TCP/IP (`rlogin` and `telnet`)
- External logins and logouts for DECnet (`dlogin` and `set host`)
- Failed logins
- Failed attempts to become superuser (the `su` command)
- Reboots and crashes
- `rsh` and `rcp` file transfer requests
- DECnet file transfer requests

Auditing for each of these categories involves a data file, that stores the pertinent information, and a method for viewing the stored data. In some cases this method is a specific command, such as `last` or `lastcomm`. In

other cases the contents of the file are viewed directly, for example, with the `more` command.

The accounting data is stored in a number of files in the `/var/adm` directory as listed in Table 3-3. The presence of specific log files on your system depends on which logging and accounting features you have enabled.

**Table 3-3: Traditional UNIX Log Files in `/var/adm`**

| File Name                                 | Security-Relevant Information                                                                   |
|-------------------------------------------|-------------------------------------------------------------------------------------------------|
| <code>wtmp</code>                         | Records all logins, logouts, and shutdowns. Use the <code>last</code> command to view this log. |
| <code>syslog.dated/date/daemon.log</code> | Messages generated by system daemons.                                                           |
| <code>syslog.dated/date/kern.log</code>   | Messages generated by the kernel (for example, for system crashes).                             |
| <code>syslog.dated/date/lpr.log</code>    | Messages generated by the line printer spooling system.                                         |
| <code>syslog.dated/date/mail.log</code>   | Messages generated by the mail system.                                                          |
| <code>syslog.dated/date/user.log</code>   | Messages generated by user processes.                                                           |
| <code>syslog.dated/date/syslog.log</code> | Requests for DECnet file transfers.                                                             |
| <code>acct</code>                         | Raw system accounting data, including user commands.                                            |

Protect the contents of these files. The files and directories should be owned by the root user, and they should not be writeable by `group` or `other`.

See *System Administration* for information on UNIX accounting.

## 3.7 Auditing in a TruCluster

The Audit Configuration icon on the Custom Setup menu of the `/usr/sbin/sysman auditconfig` utility configures audit options clusterwide. The kernel build performed by cluster creation automatically includes support for audit.

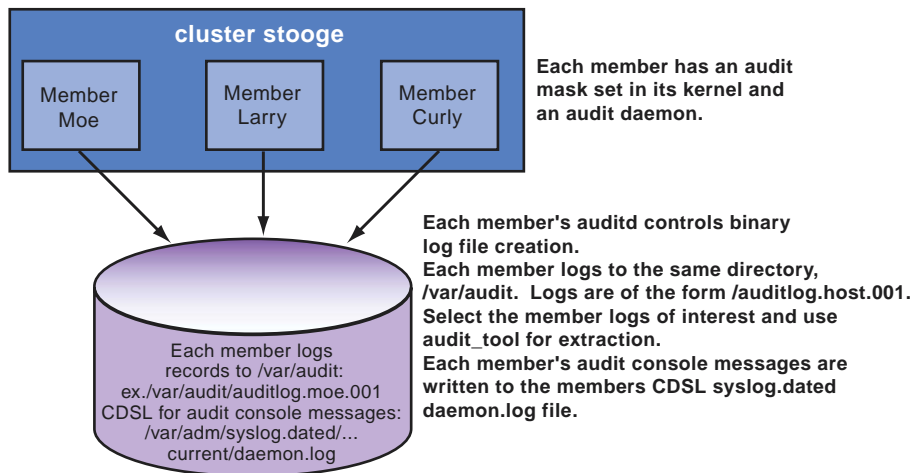
Audit configuration occurs in two steps. In the first, the parameters for the audit daemon, such as the name and location of the audit log file, are specified. In the second, a set of events to audit, called the audit mask, is selected. The configuration utility ensures that the same parameters and events are used on each cluster member. To maintain a single point of

administration, the `auditconfig` utility stores this audit configuration information in the `/etc/rc.config.common` file.

No additional kernel rebuild is required to enable auditing because the `DEC_AUDIT` kernel option and the `/dev/audit` special file are automatically included on all cluster members in the initial cluster kernel build.

When auditing is enabled, an audit daemon (`auditd`) runs on each member of the cluster. Each audit daemon records event-specific audit log entries in a private audit log file, named by default `/var/audit/auditlog.{membername}.nnn`. Each audit log entry includes the hostname of the member on which it occurred, facilitating a merged display of the entries from multiple audit log files. Figure 3–2 illustrates how audit data is gathered.

**Figure 3–2: Audit Data Flow in a Cluster**



ZK-1579U-AI

Each audit daemon writes its error or status messages to the local `/var/adm/syslog.dated/DATE/daemon.log` file, or optionally to a common audit console file.

The set of events to be audited, called the audit mask, is initially generated using the `auditconfig` utility. This utility specifies a common audit mask that is created clusterwide. Initial start up of the audit daemon uses information from the `/etc/rc.config.common` file. Thus, the same `auditd` command-line options and the same audit mask are used for each member.

On a running cluster, the `auditd` commands may be directed to every active daemon using the `auditd -cluster` option. For instance, `auditd -cluster -w` displays the status of each member. Similarly, the `auditmask -cluster` option can change or display the audit mask on every active member.

While it is possible to specify different audit daemon parameters or different audit masks for the audit daemons in a cluster, this can be confusing and is not recommended.

The `audit_tool` utility can merge audit log files to present a clusterwide view of events sorted by forward time progression. Because host name information is recorded with each event, event origin can easily be determined.

### 3.7.1 Cluster Command Examples

This section provides examples of audit commands as they would be executed on a member system in a two member (`haydn` and `handel`), Version 5.1A cluster.

To display each member's `auditd` status, enter:

```
auditd -cluster -w

Audit data and msgs:
-l) audit data destination = /var/audit/auditlog.handel.003
-c) audit console messages = syslog

Network:
-s) network audit server status (toggle) = off
-t) connection timeout value (sec) = 4

Overflow control:
-f) % free space before overflow condition = 10
-o) action to take on overflow = ignore

cluster member haydn.necorp.com auditd standard output:

Audit data and msgs:
-l) audit data destination = /var/audit/testauditlog.haydn.003
-c) audit console messages = syslog

Network:
-s) network audit server status (toggle) = off
-t) connection timeout value (sec) = 4

Overflow control:
-f) % free space before overflow condition = 10
-o) action to take on overflow = ignore
```

To display each member's audit mask status, enter:

```
auditmask -cluster

! Audited system calls:

! Audited trusted events:
```

```

login fail

! Audstyle flags: exec_argp exec_envp login_uname obj_sel

**** cluster member haydn.necorp.com standard output: ****
! Audited system calls:

! Audited trusted events:
login fail

! Audstyle flags: exec_argp exec_envp login_uname obj_sel

```

To display log file names the members are currently logging to, enter (from member hayden):

```

auditd -cluster -q

/var/audit/auditlog.haydn.001

cluster member handel.necorp.com auditd standard output:
/var/audit/auditlog.handel.001

```

To explicitly flush all the cluster member's kernel auditd buffers to get the latest audit records, enter:

```
auditd -cluster -d
```

To display the login event for both members called handel and haydn to stdout, enter:

```

audit_tool -e login auditlog.handel.001 auditlog.haydn.001

ruid/euid: 0/0
pid: 525424 ppid: 525423 ttydev: (6,1)
event: login
login name: root
devname: /dev/pts/1
.....
-- remote/secondary identification data --
hostname: mk.necorp.com
.....
char param: argv=login -h mk.necorp.com -p
char param: Failed authentication
error: 13
ip address: 10.0.0.1 (haydn-mc0)
timestamp: Mon May 3 15:54:18.19 1999 EDT

ruid/euid: 0/0
pid: 525424 ppid: 525423 ttydev: (6,1)
event: login
login name: (nil)
devname: /dev/pts/1
.....
-- remote/secondary identification data --
hostname: mk.necorp.com
.....
char param: argv=login -h mk.necorp.com -p
char param: Failed authentication
error: 13
ip address: 10.0.0.1 (haydn-mc0)
timestamp: Mon May 3 15:54:26.44 1999 EDT

```

```

ruid/euid: 0/0
pid: 1049810 ppid: 1049809 ttydev: (6,2)
event: login
login name: root
devname: /dev/pts/2
.....
-- remote/secondary identification data --
hostname: mk.necorp.com
.....
char param: argv=login -h mk.necorp.com -
char param: Failed authentication
error: 13
ip address: 10.0.0.2 (handel-mc0)
timestamp: Mon May 3 15:54:37.74 1999 EDT

```

```

ruid/euid: 0/0
pid: 1049810 ppid: 1049809 ttydev: (6,2)
event: login
login name: (nil)
devname: /dev/pts/2
.....
-- remote/secondary identification data --
hostname: mk.necorp.com
.....
char param: argv=login -h mk.necorp.com -p
char param: Failed authentication
error: 13
ip address: 10.0.0.2 (handel-mc0)
timestamp: Mon May 3 15:54:43.14 1999 EDT

```

```

4 records output
6 records processed

```

## 3.8 Responding to Audit Reports

Whenever you suspect an effort is being made to violate security, you should consider increasing the frequency of auditing. Additionally, you might want to tailor the list of events being audited to gather more specific information about the attempted violations. For example, if the attacks are against the file system, you might want to log all failed and successful file opens and closes, links, unlinks, chdirs, chmods, chowns, and other file-related activities.

When the audit trail appears to implicate a specific authorized user in attempts to violate security, be aware of your site security rules for handling security violation and make sure your actions comply with those rules. In addition, you can consider taking the following action:

- Talk with the user, reminding him or her of the importance of maintaining security and the need for all users to contribute to that effort.



- Restrict the user's access to the system by placing the user in a group of one.
- In extreme cases, deny the user system access by removing the user's account. This can be on a temporary or permanent basis.
- Audit the offending user's activities for indications that the user's behavior has changed. When you extract audit information, pay close attention to activities associated with the user's audit ID, UID, RUID, and user name.

User-specific audits can be done from the screen: Audit Manager → Reports → Generate Reports, or with the `audit_tool` options `-a AUID`, `-u UID`, and `-U username`.

If the audit trail indicates attempts to violate security but points to no specific user, it is possible that you are faced with intrusion by an outsider. Your responses must then be directed to the system and the larger user community. In this case you can take the following steps:

- Have users change their passwords and inform them about the selection of safe passwords.
- Hold meetings with users to discuss the importance of system security.
- Increase physical security to make sure that only authorized users can gain physical access to the system.
- Perform backups of the file system more frequently, to minimize the damage if a break-in should occur and data on the system is lost or altered.
- If attacks seem to be coming in over a network, increase the auditing of network-related activities.



# A

---

## Enhanced Security

This appendix contains the following information:

- Installing enhanced security
- Enabling enhanced security
- Enhanced security databases
- Enhanced security database management utilities
- Enhanced security and authenticating users
- Enhanced security and NIS
- Enhanced security in a TruCluster
- Securing Devices
- Enhanced security troubleshooting

### A.1 Installing Enhanced Security

To install enhanced security, you must install the optional enhanced security subsets (`OSFC2SECnnn` and `OSFXC2SECnnn`).

Before you the install enhanced security subsets, make a backup copy of the root file system as a precaution. The backup can be made by using one of the following commands (`dump` only works on UFS file systems):

```
dump -0uf /dev/rmt0h /
```

or

```
vdump -0Nuf /dev/rmt0h /
```

Substitute the appropriate tape device for your system. See *System Administration* for more information on backing up file systems.

You can install the enhanced security subsets in the following ways:

- During a full installation of the Tru64 UNIX (either advanced or basic) operating system software. A full installation brings up the system with only a root account. Run the `seconfig` script before adding accounts.
- During an update installation if you are updating your system from a previous version of the Tru64 UNIX operating system software. During an update installation, all user accounts and databases are preserved,

and running the `seconfig` program converts them to the enhanced security format.

After the security subsets are installed, you will see a message like the following:

```
Configuring "C2-Security " (OSFC2SECnnn)
Configuring "C2-Security GUI " (OSFXC2SECnnn)
```

If you plan to enable the password triviality checks, you must also install the `OSFDCMTEXTnnn` subset. See *Installation Guide* for more information on installing subsets.

The *nnn* represents the Tru64 UNIX version number. See *Release Notes* for the current version number.

## A.2 Enabling Enhanced Security

The `seconfig` utility is an interactive program that allows you to enable the security level (base and enhanced) on your system. You can run the program while the system is in multiuser mode. However, depending on the security features chosen, when `seconfig` is complete, you may need to change the security features, you must reboot your system.

Follow these steps to enable enhanced security:

1. Verify that the enhanced security subsets (`OSFC2SECnnn` and `OSFXC2SECnnn`) are installed.

```
/usr/sbin/setld -i | grep SEC
OSFC2SECnnn installed C2-Security (System Administration)
OSFXC2SECnnn installed C2-Security GUI (System Administration)
```

The *nnn* is a numeric value that represents the current version of the subset. See *Release Notes* for the current version number.

2. Use the `seconfig` utility to enable enhanced security by using one of the following methods:
  - Enter the `/usr/sbin/sysman seconfig` command and select ENHANCED security when prompted for a security level.
  - Use CDE and select Application Manager → System\_Management\_Uutilities → Configuration → Security. Enhanced security is enabled when you select Security.

See Section A.2.1 for enabling enhanced security considerations.

3. Bring down your system to single user and reboot (your shutdown message should inform users of the impending password changes).

The presence of the protected password daemon (`/usr/bin/prpasswd`) indicates that enhanced security is enabled.

## A.2.1 Enabling Enhanced Security Considerations

The following sections describe considerations when enabling enhanced security.

### A.2.1.1 Using NIS

If the system uses a password database that is served by NIS (Network Information Service), you are prompted to create a local enhanced authentication profile for each user in the NIS server password database. Subsequent changes in NIS passwords are not propagated to the database. The enhanced passwords now on the local system are expired and users must enter a new password the next time they log in.

If you change the security level back to base security, the enhanced authentication profile files are left in place. When you return to ENHANCED security, as long as there is an enhanced authentication profile file and it contains a password, the enhanced password is updated.

See Section A.6 for more information about using enhanced security and NIS.

### A.2.1.2 Segment Sharing

Because of the page table sharing mechanism used for shared libraries, the normal file system permissions are not adequate to protect against unauthorized reading. For example, user `joe` has the following shared library:

```
-rw----- 2 joe staff 100000 Sep 18 1997 /usr/shlib/foo.so
```

When this shared library is used in a program, the text part of `foo.so` may be visible to other running processes even though they are not running as user `joe`. Only the text part of the library, not the data segment, is shared in this way.

To disable all segmentation and avoid any unauthorized sharing, answer “yes” when `seccfg` asks if you want to disable segment sharing. The `seccfg` script reports when segment sharing is already disabled.

---

#### Note

---

Disabling segment sharing can cause excessive memory use.

---

### A.2.1.3 Execute Bit Set Only By Root

Enabling the `execute bit set only by root` option allows only the root user to set the execute permissions on files.

## A.2.2 Configuring Enhanced Security

Enhanced security provides the ability to specify system default values that apply to users, terminals, and devices. The following sections briefly describe some common defaults and how to configure them.

The system defaults are stored in the default database at `/etc/auth/system/default`. This database can contain four types of fields:

- System wide fields that exist only in the default database. These fields are prefixed with a `d_`.
- User default fields, whose values can be overridden by the corresponding fields in a user's profile. These fields are prefixed with a `u_`.
- Terminal control fields, whose values can be overridden by the corresponding fields in the terminal control database. These fields are prefixed with a `t_`.
- Device assignment fields, whose values can be overridden by the corresponding fields in the device assignment database file. These fields are prefixed with `v_`.

You use the following interfaces to change the default value of enhanced security related users, terminals, and devices settings:

- The `dxaccounts` GUI to modify the default fields for users by going to `Local Templates`→`Default`.
- The `dxdevices` GUI to modify the default fields for devices.
- The `edauth` utility provides a lower-level interface to all of the default fields.

See `authcap(4)` for a description of the file format and field values, `edauth(8)` for use of `edauth`, and `default(4)`, `prpasswd(4)`, `ttys(4)`, and `devassign(4)` for descriptions of the various fields and an interpretation of values.

### A.2.2.1 Aging

If you do not want password aging on your system, in the `default` database set `u_exp` and `u_life` to 0, and then (because of the way the default methods of determining length restrictions on passwords work based on the password lifetime) also set `u_minchosen` and `u_maxchosen` to appropriate values for the site.

An example entry could be as follows:

```
:u_exp#0:u_life#0:u_minchosen#5:u_maxchosen#32:\
```

### A.2.2.2 Minimum Change Time

You can remove the minimum change time interval by setting the `u_minchg` field to 0 as follows:

```
:u_minchg#0:\
```

This allows users to immediately change their password after a previous password change.

### A.2.2.3 Changing Controls

The password-changing controls can be configured to your site's needs. By putting the following fields in the `default` database, you allow users to select how their passwords are chosen:

```
:u_pickpw:u_genpwd:u_genchars:u_genletters:u_restrict:\
:u_policy:u_nullpw:u_pwdepth#0:\
```

(Of those, `u_pwdepth` is numeric and the rest are Boolean. A Boolean field is true if it is specified and false if it is followed by an @.)

### A.2.2.4 Maximum Login Attempts

In breakin detection, consecutive login failures are counted and compared to a maximum for a user (`u_maxtries`) or for a terminal (`t_maxtries`). If the maximum is exceeded, then logins to the user account or the terminal are disabled for a time period specified by `u_unlock` or `t_unlock`. To disable breakin evasion for user accounts, set `u_maxtries` to 0. To disable for terminals, set `t_maxtries` to 0. The `default` database entry for users would be as follows:

```
:u_maxtries#0:\
```

### A.2.2.5 Time Between Login Attempts

If the default evasion time (86400 seconds or 24 hours) is not appropriate for your site, change the `u_unlock` field to an appropriate value for your site (number of seconds before a success is recognized after the last failure, once the `u_maxtries` limit is reached). Setting the `u_unlock` field to 0 (`:u_unlock#0:`) sets the time between login attempts to infinity (no automatic reenabling occurs). The equivalent behavior for terminals is controlled by `t_maxtries`.

### A.2.2.6 Time Between Logins

You can set system wide maximum allowable time between logins in the `u_max_login_intvl` field of the default database.

The system default login timeout for terminals can be changed in the `t_login_timeout` field of the default database. It can also be set in the \* entry of the `ttys` database. This field should be 0 (infinite) for X displays.

### A.2.2.7 Per-Terminal Login Records

If you do not want to record per-terminal login successes and failures, set the `d_skip_ttys_updates` Boolean field in the default database as follows:

```
:d_skip_ttys_updates:\
```

This has the side effect of disabling any further per-terminal breakin evasion.

### A.2.2.8 Successful Login Logging

Strict C2 security requires the logging of successful logins. To disable this logging, set the `d_skip_success_login_log` Boolean field as follows:

```
:d_skip_success_login_log:\
```

### A.2.2.9 Failed Login Logging

Failed login attempts to user accounts are normally recorded. To disable this logging, which also disables breakin detection and evasion system wide, set the `d_skip_fail_login_log` Boolean field as follows:

```
:d_skip_fail_login_log:\
```

### A.2.2.10 Automatic Enhanced Profile Creation

Setting the `d_auto_migrate_users` Boolean field allows the creation of enhanced profiles at login time if they are missing, so that traditional methods of adding user profiles can be used without change.

### A.2.2.11 Vouching

You can set the `d_accept_alternate_vouching` field to allow enhanced security and DCE to work together.

### A.2.2.12 Encryption

If you want the user passwords to stay in the `/etc/passwd` file to support programs that use `crypt()` to do password validation, but still want to use other features of enhanced profiles, put the following entry in the default database before running the `/usr/sbin/sysman secconfig` utility:



```
:u_newcrypt#3:\
```

This corresponds to the AUTH\_CRYPT\_C1CRYPT value from the `<prot.h>` file.

## A.3 Enhanced Security Databases

Table A-1 identifies the enhanced security databases.

**Table A-1: Enhanced Security Databases**

| Database           | Location                                                               | Contents                                  |
|--------------------|------------------------------------------------------------------------|-------------------------------------------|
| Protected password | <code>/tcb/files/auth.db</code><br><code>/var/tcb/files/auth.db</code> | User authentication database              |
| System defaults    | <code>/etc/auth/system/default</code>                                  | Default values for database fields        |
| Terminal control   | <code>/etc/auth/system/ttys.db</code>                                  | Security information about each terminal  |
| File control       | <code>/etc/auth/system/files</code>                                    | Protection attributes of each system file |
| Device assignment  | <code>/etc/auth/system/devassign</code>                                | Device-specific controls                  |

### A.3.1 Enhanced (Protected) Password Database

The protected password database stores the enhanced authentication profile for each user who has an account on the system. Each profile contains information such as the following:

- User name and ID
- Encrypted password
- User's audit characteristics
- Password generation parameters
- Successful and unsuccessful login times and terminals

The enhanced (protected) password database is located in the file `/tcb/files/auth.db`.

See `prpasswd(4)` for more information on the contents of the enhanced password database.

### A.3.2 System Defaults Database

The system defaults database stores default values for database fields. These defaults are used when the administrator does not set explicit values

in the enhanced (protected) password database, terminal control database, or device assignment database.

The system defaults database contains information such as the following:

- Default password generation parameters
- Default number of unsuccessful login attempts allowed per user
- Default number of unsuccessful login attempts allowed per directly connected terminal
- Default device assignment parameters

More information on the contents of the system defaults database located in `/etc/auth/system/default` can be found in `default(4)`.

### A.3.3 Terminal Control Database

The terminal control database contains information that the administrator uses to control login activity at each terminal attached to the system. The system uses this database as an aid in controlling access to the system through terminals. The administrator can set different policies for logins at different terminals, depending upon the site's physical and administrative needs.

Each entry in the terminal control database contains information such as the following:

- Terminal device name
- User ID and time stamp of the last successful login attempt from this terminal
- User ID and time stamp of the last unsuccessful login attempt from this terminal
- Delay imposed between login attempts from this terminal
- Number of unsuccessful attempts that can be made before locking this terminal

When the system is installed, the terminal control database contains an entry for the system console. You modify these initial values during system setup. A corresponding entry, also initially installed, is required in the device assignment database before logins are allowed.

For more information about the contents of the terminal control database located in `/etc/auth/system/ttys.db`, see `ttys(4)`.

### A.3.4 File Control Database

The file control database contains information about the protection attributes of system files (that is, files important to the enhanced security operation). This database helps maintain system integrity. It contains one entry for each system file.

Each entry in the file control database contains the following information:

- Full pathname of the file
- File owner and group
- File mode and type
- Potential and granted privilege sets
- Access control list

When the system is installed, the file control database contains entries for all security relevant system files. You do not need to modify this database during system setup and rarely needs to update it during system operation.

See `files(4)` for more information about the contents of the `/etc/auth/system/files` file.

### A.3.5 Device Assignment Database

The device assignment database contains information about devices that are used to exchange data with users. Each login terminal must have an entry in the device assignment database. The system uses this database as an aid in restricting the security attributes of data that can be sent or received through the system's devices.

Each entry in the device assignment database contains information that describes a device and that relates the device pathname to the appropriate physical device. This is necessary because a number of distinct pathnames can refer to the same physical device. For example, two pathnames can refer to the same serial port – one with modem control enabled and the other with modem control disabled.

Each entry in the device assignment database contains information such as the following:

- Device pathname
- Other pathnames referencing the same physical device
- Device type

Entries referring to login terminals must have corresponding entries in the terminal control database.

The device assignment database is located in `/etc/auth/system/devassign`. See `devassign(4)` for more information.

## A.4 Enhanced Security Database Management Utilities

A customized version of the Berkeley Database (Berkeley DB) is embedded in Tru64 UNIX to provide high-performance database support for critical security files. The DB includes full transactional support and database recovery, using write-ahead logging and checkpointing to record changes. In the event of catastrophic failure, the security database can be restored to its last transaction-consistent state by restoring the database files and rolling the log forward.

The following database management utilities are included with Enhanced Security and located in the `/usr/tcb/bin` directory:

|                            |                                                                                                                                                                             |
|----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>db_archive</code>    | Displays the enhanced security database log files no longer involved in active transactions that can safely be backed up and deleted to regain space on <code>/var</code> . |
| <code>db_checkpoint</code> | Flushes memory, writes a checkpoint record to the log and flushes the log to disk.                                                                                          |
| <code>db_load</code>       | Reads from a file or standard input and loads into a database.                                                                                                              |
| <code>db_unload</code>     | Unloads the database into a file.                                                                                                                                           |
| <code>db_stat</code>       | Displays the security database statistics.                                                                                                                                  |
| <code>db_recover</code>    | Restores the database to a consistent state after an unexpected failure.                                                                                                    |

In general, the security database is loaded or unloaded only by installation utilities. While the database has been designed to minimize database administration tasks, the addition of security database log files does present the possibility of log files expanding to fill `/var`. Thus, the security configuration utility includes an option that creates a `cron` job to periodically delete log files no longer involved in active transactions.

## A.5 Enhanced Security and Authenticating Users

A system running enhanced security authenticates users by using the `/etc/passwd` file and an enhanced security authentication database, which is composed of the following subsidiary databases:

- Protected password database (`/tcb/files/auth.db` and `/var/tcb/files/auth.db`)
- System defaults database (`/etc/auth/system/default`)
- Terminal control database (`/etc/auth/system/ttys.db`)
- File control database (`/etc/auth/system/files`)
- Device assignment database (`/etc/auth/system/devassign`)

The enhanced security authentication database has an entry for each user account defined in `/etc/passwd`. Under enhanced security, the `/etc/passwd` file remains unchanged except for the encrypted password, which moves from the `/etc/passwd` file into `auth.db` database. The other fields in the `/etc/passwd` file (shell and so forth) remain in the `/etc/passwd` file and are used in a normal fashion.

The enhanced security authentication database uniquely identifies a user by user name and UID, which must match the user's entry in the `/etc/passwd` file. In addition to the encrypted password, an entry contains a set of fields and values used only by enhanced security. See `prpasswd(4)` for a description of these fields, and `authcap(4)` for a description of the file format.

A user account can be associated with a template account, which can be used to specify default values for a group of users. An account is always finally associated with the system default template values that are contained in the `/etc/auth/system/default` file.

Users continue to use the `passwd` command to change their password when using enhanced security.

### A.5.1 User Profiles

A user's entry in the enhanced security authentication database is called a user's profile. Security-aware programs interpret the fields and values in a profile. A user profile need not contain every possible field. If a field is not specified in a user's profile, the system looks in the template account associated with the user, and finally in the system default template, until it finds a value for the field.

Values are obtained as follows:

- If the user profile contains a user-specific value, that value is used.
- If the user profile contains a reference to a template account, and no user-specific value is defined, the value in the template account is used.
- If neither the user profile nor the template account defines a value for a field and the system default template defines a value for that field, the system default template value is used.

- If the value is defined nowhere else, a static system default value is used for the field.

The system default template values are located in the `/etc/auth/system/default` file and can be modified by using the `dxaccounts View Local Template` option, or with the `edauth` utility. Other template accounts are stored in the `auth.db` database. Note that template accounts have no corresponding entry in the `/etc/passwd` file.

### A.5.1.1 Recovery of `/etc/passwd` Information

If the `/etc/passwd` file is lost, but the enhanced profiles are still available, you can enter a command sequence like the following to recover some of the missing data:

```
bcheckrc
/tcb/bin/convuser -dn | /usr/bin/xargs /tcb/bin/edauth -g | \
 sed '/:u_id#/!d;s/.*:u_name=//;s/:u_id#/:*/;s/:u_.*$/:/' \
 > psw.missing
```

This creates a `psw.missing` file containing entries like the following:

```
root:*:0:
jdoe:*:1000:
```

Primary group, finger, home directory, and login shell information is not recorded in the enhanced profile. The data for those fields must be recovered by other means.

### A.5.2 Enhanced Security Authentication Database Integrity Checking

You use the `/usr/tcb/bin/authck` command to check the overall structure and the internal consistency of the enhanced security authentication database. The `authck` command checks for the correctness of entries within each database and also checks related fields in other databases. For example, it checks the protected password database entry for a user against the `/etc/passwd` file.

The `authck` command produces a report listing any discrepancies between the databases. Compare the output of the program with the actual database entries and rectify any differences immediately. Problems typically occur because someone has manually updated one of the databases without making the corresponding change to the related databases.

You can specify the following arguments on the `authck` command line:

- P Checks the protected password database and the `/etc/passwd` file to ensure that they are complete and that they agree with each other. It also checks the protected password database for reasonable values.

- t Checks the fields in the terminal control database for reasonable values.
- f Checks the file control database for syntax and value specification errors. Without this flag, entries with unknown authorizations, user names, and so on, are ignored. Typically these errors are typographical, such as “rooot” instead of “root,” and the program attempts to guess the right value.
- v Verbose mode.
- a Performs the functions of -f, -p, -t, and -v. Provides program activity status during operation.

See `authchk(8)` for more information.

### A.5.3 Adding Applications to the File Control Database

When you add applications to the system by a means other than the `setld` program, you should also add file control database entries for the application’s control and database files and programs. It is best to consult with the application supplier to get a file and program list, and suggested protection attributes for all files.

If you add the application’s files to the file control database, you gain the benefit of periodic integrity checking of that application’s resources.

See `fverify(8)` for more information on checking file integrity.

## A.6 Enhanced Security and NIS

The Network Information Service (NIS) can be used to distribute all or part of the enhanced (protected) password database as well as the BSD user account and group databases. A Tru64 UNIX NIS master can serve NIS clients that are Tru64 UNIX systems running enhanced security as well as any NIS clients using BSD authentication, including systems from other vendors.

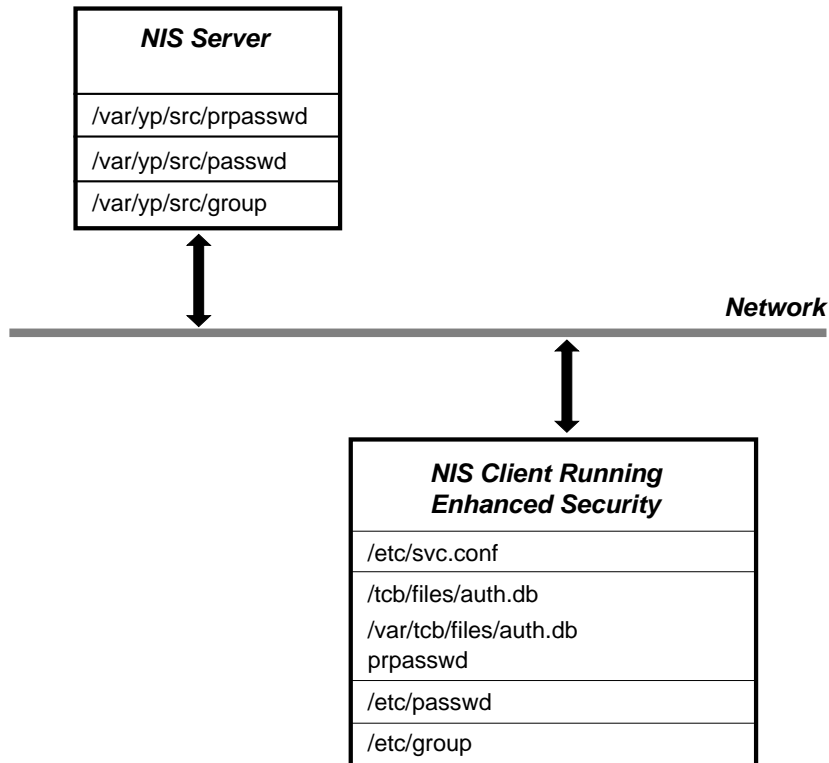
When you are running NIS over enhanced security you have the following user account databases:

- On the NIS master server:
  - The NIS-distributed base user account database generated from the `/var/yp/src/passwd` and `/var/yp/src/group` files and distributed as `ndbm` or `btree` maps.

- The NIS-distributed enhanced security user account database generated from the `/var/yp/src/prpasswd` file and distributed as `btree` maps.
- On the NIS client:
  - The local base user account database in the `/etc/passwd` and `/etc/group` files.
  - The local enhanced security user account database.

Figure A–1 shows the user account databases on the NIS master server and client.

**Figure A–1: NIS and Enhanced Security Files**



ZK-1087U-AI

The `auth=` entry in the `/etc/svc.conf` file indicates the order in which the local and NIS enhanced security user account databases are searched for user entries, either local first or NIS (yp) first.

The plus sign (+) override feature for `/etc/passwd` entries works as usual.



---

### Note

---

When upgrading from a base security system with NIS to an enhanced security system, the `/usr/sbin/sysman secconfig` utility only creates `auth.db` entries for NIS users (the `+username` entries in the `/etc/passwd` file) if you answer yes to the Create Entries for NIS Users question.

---

There is no override feature for the enhanced security user account database. A user's profile is contained completely in either the local database or in the NIS distributed data base. Although templates can be defined for NIS accounts and distributed as part of the NIS enhanced security maps, NIS does not distribute the system default template (`/etc/auth/system/default`). This template provides the final default values for fields not specified in a user's profile. Therefore, under enhanced security, a NIS client uses its own `/etc/auth/system/default` file to obtain final default values for both local and NIS user profiles. If the client system default file contains different values than that of the NIS master, unintended behavior can occur.

The `passwd` command changes the password in a user's local or NIS enhanced security entry. The `yppasswd` command changes the fields in the NIS-distributed base user account database as usual.

NIS user accounts can be modified using the `dxaccounts` View NIS User option, or by specifying the `-x distributed=1 local=0` options to the `useradd`, `usermod`, and `userdel` utilities.

See *Network Administration: Services* for more information about NIS.

## A.6.1 Templates for NIS Accounts

The `/var/yp/src/prpasswd` file is the source for enhanced security user accounts distributed by NIS. It can contain template profiles and normal user profiles. As with a local user profile, a NIS user profile need not contain every possible field. If a field is not specified in a NIS user's profile, the system looks in the NIS template account associated with the user, and finally in the local system default template, until it finds a value for the field.

Values are obtained as follows:

- If the user profile contains a user-specific value, that value is used.
- If the user profile contains a reference to a template account, and no user-specific value is defined, the value in the template account is used.
- If neither the user profile nor the template account defines a value for a field and the system default template defines a value for that field, the system default template value is used.

- If the value is defined nowhere else, a static system default is used for the field.

NIS template accounts are modified using the `dxaccounts` View NIS Template option, or with the `edauth` utility.

The system default template values are located in the `/etc/auth/system/default` file on the NIS client. Note that NIS does not distribute the system default template. A NIS client uses its own `/etc/auth/system/default` file to obtain final default values for both local and NIS user profiles. If the client system default file contains different values than that of the NIS master, unpredicted behavior can occur.

## A.6.2 Configure a NIS Master with Enhanced Security

Follow these steps to configure a NIS master with enhanced security:

1. If NIS is running on the master server, stop it:
 

```
/sbin/init.d/nis stop
```
2. Ensure that the enhanced security subsets are installed.
3. Modify the system default template:
 

```
edauth -dd default
```

 Set the following fields:
 

```
d_skip_success_login_log:
d_skip_ttyupdates:
```
4. Create `/var/yp/src/hosts`, `/var/yp/src/passwd`, `/var/yp/src/group`, and `/var/yp/src/prpasswd` files.
5. Run the `sysman nis` program.
  - a. When the `sysman nis` program first prompts for security (`-s` option to `ypbind`), choose `y` to run `ypbind -s`, which specifies a secure socket.
  - b. When the `sysman nis` program again prompts for security (`-S` option to `ypbind`), choose `y` and specify a domain name and up to four authorized slave servers.
6. Make sure that the `/etc/svc.conf` file has the following entry:
 

```
auth=local,yp.
```
7. Start NIS:
 

```
/sbin/init.d/nis start
```

### A.6.2.1 Manual Procedure: Maps for Small User Account Databases

For a NIS master server supporting clients using enhanced security, a manual procedure is best. Set up the account maps using the `dxaccounts` program or alternatively the `adduser`, `addgroup`, `useradd`, `userdel`, and `usermod` commands. See Section A.6.5 for another method of setting up accounts.

### A.6.2.2 Automated Procedure: Maps for Large User Account Databases

If you have a large existing NIS distributed base user accounts database, you can automate the creation of the NIS distributed enhanced (protected) password database by entering the following command:

```
convuser -Mc
```

Alternatively, you can create the map by creating a `/var/yp/src/prpasswd` file and then executing the following commands:

```
/usr/tcb/bin/edauth -Lg > /var/yp/src/prpasswd
cd /var/yp; make prpasswd
```

### A.6.3 Setting Up a NIS Slave Server with Enhanced Security

If NIS is running on a slave server, you must stop NIS using the `/sbin/init.d/nis stop` command. The following setup information is specific to a NIS slave server supporting clients using enhanced security:

1. Ensure that the enhanced security subsets are installed.
2. Modify the system default template:

```
edauth -dd default
```

Set the following fields:

```
d_skip_success_login_log:
d_skip_ttys_updates:
```

3. Run the `sysman nis` program.
  - a. When the `sysman nis` program first prompts for security (`-s` option to `ypbind`), choose `y` to run `ypbind -s`, which specifies a secure socket.
  - b. When the `sysman nis` program again prompts for security (`-S` option to `ypbind`), choose `y` and specify a domain name and up to four authorized slave servers.
4. Make sure that the `/etc/svc.conf` file has the following entry:

```
auth=local,yp.
```

5. Edit the `/var/yp/ypxfr_1perday`, `/var/yp/ypxfr_1perhour`, `/var/yp/ypxfr_2perday` files to add the following lines to each:

```
ypxfr -a "$method" prpasswd
ypxfr -a "$method" prpasswd_nonsecure
```

6. Start NIS:

```
/sbin/init.d/nis start
```

#### A.6.4 Setting Up a NIS Client with Enhanced Security

If NIS is running on a slave server, you must stop NIS using the `/sbin/init.d/nis stop` command. The following setup information is specific to a NIS client using enhanced password security:

1. Ensure that the enhanced security subsets are installed.
2. Modify the system default template using the following command:

```
edauth -dd default
```

Set the following fields:

```
d_skip_success_login_log:
d_skip_ttys_updates:
```

3. Run the `sysman nis` program.
  - a. When the `sysman nis` program first prompts for security (`-s` option to `ypbind`), choose `y` to run `ypbind -s`, which specifies a secure socket.
  - b. When the `sysman nis` program again prompts for security (`-S` option to `ypbind`), choose `y` and specify a domain name and up to four authorized slave servers.
4. Edit the `/etc/svc.conf` file to include a `yp` entry for `auth`. The entry should be as follows: `auth=local,yp`.
5. Start NIS using the `/sbin/init.d/nis start` command.

#### A.6.5 Moving Local Accounts to NIS

To move existing local accounts to NIS, enter:

```
edauth -Lg | edauth -NsC
```

#### A.6.6 Removing NIS Support

If you need to remove the NIS, copy the NIS accounts to the local database and then remove NIS using the following commands on the client:

```
edauth -gN | edauth -sLC
sysman nis
<select the Remove option from the menu>
```

The enhanced (protected) password database on the client is updated with any accounts from the NIS database that are not present in the local database.

## A.6.7 Implementation Notes

The following information is specific to enhanced security and NIS:

- To change your password when running NIS with enhanced security, use the `passwd` command for both local and distributed enhanced (protected) password database entries. The `passwd` command uses the search list in the `svc.conf` file (`auth=local,yp` entry) and updates the password in the first enhanced (protected) password database entry it finds for the specified user, even if that entry is in the NIS-distributed enhanced password database.
- It is very important that each enhanced password database entry exists in only one database, either the local enhanced password database or the NIS-distributed enhanced password database. The routines that check and manipulate the enhanced password database information work on the first copy found (as defined in the `svc.conf` file). NIS `yp` routines work on the NIS-distributed enhanced password database only. This can cause confusing results if you have the same entry in both places. If this happens, delete one of the entries.
- It is strongly recommended that you do not distribute `root` account information. Maintaining a local `root` account on a client system allows you to still log in on the client systems using the `root` account if your NIS server is down.
- To maintain last successful login information, you can configure an update to a user's enhanced profile each time that the user logs in. On a NIS master, this requires rebuilding the map and shipping it to the slaves. Tru64 UNIX Version 5.1B makes these updates optional and you can disable the updates by setting the value of the `d_skip_success_login_log` system default field to `true`. Disabling successful login logging means that the NIS master server does not always have to be available for logins to be successful if there is a properly configured NIS slave server.
- Scalability improvements include:
  - An update to a single entry does not always cause a rebuild of the entire `prpasswd` map. The map entries are updated directly if possible.

Manual modifications to the `prpasswd` file followed by manual rebuilds is strongly discouraged because the `rpc.yppasswdd` daemon might have cached an entry and the manual modification could be overwritten. If you feel the need to modify the `prpasswd` file, stop the NIS server, make the modifications, then restart the NIS server.

- If successful login logging is enabled, a successful login does not wait for the NIS map to be distributed before completing. It only waits to make sure that the NIS master has been updated. If unsuccessful login logging is enabled, unsuccessful login attempts still wait for the map to be distributed to the slave servers before completing. This is required for security and timing issues.
- The database format for NIS maps is configurable. You can choose `btree` or `hash` in addition to `ndbm`. When using `ndbm` for NIS map storage, there is a limit to the number of account records that can be stored, which depends on the mix of account names and UIDs. A typical limit is about 30,000 entries, but some mixes of account names and UIDs can result in a limitation of fewer than 10,000 entries. Because of this constraint in `ndbm`, use `btree` as your database format, especially when using enhanced security.
- NIS servers work best with a common database format. If a slave server has defined a different format than the master (`ndbm` instead of `btree`, for example), the time it takes to push any maps to that slave server is drastically increased because the slave server must rebuild its database one element at a time, instead of receiving the database from the master as a single entity.
- NIS slaves that are not listed in the `ypservers` NIS map on the NIS master can cause performance problems for NIS clients bound to those slaves. To solve this, define all NIS slaves in the `ypservers` NIS map on the NIS master. Then, on the slave server, execute the following commands to pull the user account databases from the NIS master:

```
/var/yp/ypxfr -d `domainname` -h NISMASTER -c prpasswd
/var/yp/ypxfr -d `domainname` -h NISMASTER -c
prpasswd_nonsecure
```

In the example, substitute the name of the local NIS master server for `NISMASTER`. This will transfer initial copies of those maps for those slave servers.

- A login process that encounters a login failure has to check the `prpasswd` map for the latest unsuccessful login information. This requires an up-to-date `prpasswd` map. Thus, the `yppush` operation for the `prpasswd` map must occur for each failed login; that map (at least) must be pushed during the normal operation of the `rpc.yppasswdd` daemon.

Setting the `/var/yp/Makefile` variable `NOPUSH` is not recommended for such configurations.

- Sites that cannot use NIS to share `prpasswd` information may want to use NFS to share the `/tcb/files` and `/var/tcb/files` directories. This requires exporting the directories with root access to the participating nodes with `-root=client1:client2:client3` or `-root=0`, as appropriate. See `exports(4)`. It also requires that NFS locking be enabled so that database corruption does not occur.

## A.6.8 Troubleshooting NIS

Table A–2 discusses some common NIS problems and possible reasons for those problems.

**Table A–2: NIS Troubleshooting**

Problem	Possible Reason
Successful login to a local account, but cannot log in to any of the NIS accounts. The <code>dxaccounts</code> utility displays that the account exists and is not locked.	<ol style="list-style-type: none"> <li>1. Check the <code>/etc/svc.conf</code> file and see if it contains the line <code>auth=local,yp</code>.</li> <li>2. Check the <code>/etc/passwd</code> file and see if there is a "+" as the last line of the file.</li> </ol>
Slave NIS server does not get the updated <code>prpasswd</code> maps on boot.	<p>Check the <code>/var/yp/ypxfr_1perday</code>, <code>/var/yp/ypxfr_1perhour</code>, and <code>/var/yp/ypxfr_2perday</code> files and verify that each contains the lines:</p> <pre>ypxfr -a "\$method" prpasswd ypxfr -a "\$method" prpasswd_nonsecure</pre>
The <code>dxaccounts</code> program View popup menu does not show any NIS User Account Database options (for example, NIS Users, NIS Groups, and NIS Templates).	NIS is not running or has not been configured.
When you issue the <code>make</code> command from <code>/var/yp</code> , you get the message <code>Map 'ypslaves' is empty for domain 'domainname'</code>	This is an informational message. No action is required.
When you issue the <code>make</code> command from <code>/var/yp</code> , you get the message <code>Map 'hosts.byname' is empty for domain 'domainname' cant bind to master for domainname hosts.byname no such map in server's domain will use slave copy!</code>	<p>The hosts map does not exist. Enter the following commands:</p> <pre># touch /var/yp/src/hosts # cd /var/yp # make</pre>

## A.7 Enhanced Security in a TruCluster

A TruCluster Server cluster is a single security domain. Identification and authentication, access control lists (ACLs), and auditing are configured identically on each member by default, presenting a coherent interface to the user and the system administrator.

Because a single copy of the authentication files is shared among all cluster members, each user account is valid on all cluster members and a user can log in to the cluster without concern for which member accepts the connection. Identically configured ACL checking means consistent authorization and file access control; a user has the same access rights from every member. Clusterwide audit settings ensure a uniform capture of cluster activity.

### A.7.1 Upgrading from Base to Enhanced Security in a TruCluster

Upgrading from base to enhanced security in an existing TruCluster requires a full cluster reboot. The upgrade copies user accounts from the `/etc/passwd` file into the `auth.db` databases, removes passwords from the `/etc/passwd` file, and switches to a new security library. A new process authenticating a user name and password, such as a `telnet` session, uses the new library and accesses the new databases. However, an existing process, such as a `dtlogin` session or a locked CDE window, continues to use the original library. Because this library expects to access the `/etc/passwd` file, from which passwords have been removed, an existing process consistently encounters password verification failures. In particular, the console login window encounters this problem, which can create the erroneous belief that the root account is disabled. The cluster reboot prevents this situation.

### A.7.2 Installing and Configuring Enhanced Security in a TruCluster

The best time to configure enhanced security for a TruCluster is before loading the TruCluster Server license and subsets and creating the cluster. You must select the enhanced security subsets (`OSFC2SECnnn` and `OSFXC2SECnnn`) during the installation.

The *nnn* represents the Tru64 UNIX version number. See *Release Notes* for the current version number.

The enhanced security software is installed on members in a TruCluster as follows:

- If you are installing Tru64 UNIX on the first member of cluster, you must completely set up your security environment before installing the TruCluster Server software. The security configuration, as well as other



configuration data, will be propagated to other cluster members as they start.

- If you are installing Tru64 UNIX on a member of a cluster other than the first member, the enhanced security environment will be inherited from the existing members in the cluster when the TruCluster software is installed.
- If you are enabling enhanced security on systems already running in a cluster environment, you must setup enhanced security from a single member, then reboot each member in the cluster.

Both base and enhanced security are supported in a cluster. Base security, which uses the standard UNIX `/etc/passwd` and `/etc/group` files, is the default configuration. Enhanced security is configured by using the Security Configuration icon on the Custom Setup menu of the `seconfig` utility.

### A.7.3 Access Control Lists

The Security Configuration icon on the Custom Setup menu of the `/usr/sbin/sysman seconfig` utility contains a checkbox that enables or disables access control list support on all cluster members, under either base or enhanced security. ACL support determines the state (enabled or disabled) of access checking and ACL inheritance. Note that ACLs can be created, modified, deleted and examined regardless of the state of ACL support.

The NFS file systems require the `proplistd` daemon to support ACLs.

### A.7.4 Distributed Logins and NIS

A cluster provides a common authentication environment to enable secure, distributed, highly available logins. The cluster can additionally function as a NIS master, slave, or client. (Note that all cluster members must play the same role in a NIS environment.)

As a NIS master, the cluster supports the NIS distribution of both standard user profiles from `/etc/passwd` and the enhanced user profiles available with enhanced security maintained in the protected password database. These enhanced user profiles can be distributed by NIS as the `prpasswd` map, in the same manner that `/etc/passwd` is distributed as the `passwd` map.

Follow these steps to set up a cluster running enhanced security as a NIS master:

1. Load `/var/yp/src`, including `passwd` with specified accounts. See *Network Administration: Services* for more information.

2. Set up the `prpasswd` map with one line per entry using `convuser -Mc`. See `convuser(8)` for more information.
3. Set up NIS as described in *Network Administration: Services*. When running `nissetup`, select the `-S` security option of `ypbind` to bind the member to an authorized list of NIS servers and specify the cluster alias as one of these servers.
4. Modify the map maintenance scripts to support `prpasswd` maps as discussed in the Tru64 UNIX *Network Administration: Services* guide.

---

#### Notes

---

In domains where one or more nodes are running enhanced security and that mix Tru64 UNIX Version 5.1B and higher and UNIX Version 4.x systems, a Tru64 UNIX Version 5.1B or higher system must be the NIS master. If none of the nodes are running enhanced security, a Version 4.x system can be the NIS master.

The `dxaccounts` utility does not allow you to add a NIS account and change the user's security options at the same time. You must first create the account and then change the user's security options.

The `useradd` command fails unless the user's primary group is defined in the `/var/yp/src/group` map.

---

### A.7.5 Daemons

Enabling enhanced security introduces a new daemon, `prpasswd`. Two instances of the daemon, a parent and a child, execute on each cluster member. The parent is primarily responsible for starting or restarting the child. The child is responsible for writing changes to the authentication database. To eliminate lock contention in a cluster, only the daemon on the cluster member serving the `/var` mount point actually performs writes for all clients. The other `prpasswd` daemons are in hot standby mode. If the cluster member serving the mount point and containing the active daemon fails, another member assumes both roles.

On a cluster acting as a NIS master with enhanced security, the `rpc.yppasswdd` daemon acts in the same fashion as the `prpasswd` for the NIS `prpasswd` map.

## A.8 Securing Devices

Workstations present security problems because they are typically found in ordinary offices, rather than the more easily protected environment of the computer room.

It is possible for someone who gains access to a workstation, to get superuser status on that system, and consequently on other systems. One method is to boot the system into single user mode.

If your office has a locking door, lock the door when you are away from your system.

You must also protect your removable media, such as tape cartridges and floppy disks by locking up all floppy disks and tape cartridges when they are not in use.

Some workstations allow a console password to be set. When a console password is in use, only a default boot can be done without a password. Check your hardware and firmware documentation for more information about console passwords.

### A.8.1 Device Security Characteristics

Defining and setting security characteristics include the following considerations:

- Creating and maintaining device-specific information. You can override system defaults for an individual device, where appropriate, to grant additional rights or to impose additional restrictions. You can also lock a terminal to prevent use.
- Setting default control parameters for the devices that are included in the system's secure configuration. The system defaults for terminals are as follows:
  - Maximum number of unsuccessful login attempts is 10.
  - Login timeout as shipped is unset, which implicitly defaults to 0 which is treated as infinite.
  - Delay between unsuccessful login attempts is 2 seconds.

Before you create or modify a secure device, all of the device installation procedures required during ordinary system hardware and software installation must be completed. The special files for devices must exist in the `/dev` directory and have the appropriate permissions. The special files for terminals must be owned by `root`, have the group set to `tty`, and have the mode set to `0620`.

You can verify that the installation has been completed with the `ls` command. The following example is typical:

```
ls -lg /dev/tty*

crw----- 1 root tty 0, 2 Aug 15 09:29 /dev/tty00

crw----- 1 root tty 0, 3 Aug 15 09:29 /dev/tty01
```

You define the security characteristics of all the devices that are part of the system by using the `dxdevices` program. The `dxdevices` program provides control over the device assignment database and the terminal control database. You can use the `dxdevices` program to assign security attributes to terminals and X displays.

#### A.8.1.1 Modifying, Adding, and Removing Devices with the `dxdevices` Program

Using the Devices dialog box, select the Modify/Create dialog box then the Select devices dialog box. To add or remove a device, first select or enter the device, then click on File to make the required changes. To modify a device, first select the device, then click on Modify to make the required changes. See the online help for `dxdevices` for more information.

#### A.8.1.2 Setting Default Values with the `dxdevices` Program

Using the Devices dialog box, select the Defaults dialog box. Set the system defaults for all of your terminals as required. A terminal uses these defaults unless specifically overridden by settings in the Modify Terminal dialog box. See the online help for `dxdevices` for more information.

### A.8.2 Updating Security Databases

When you assign device defaults or device-specific parameters, the system updates the following security databases:

- The system defaults database, `/etc/auth/system/default`, contains the default values (for example, default control parameters) for all system devices.
- The device assignment database, `/etc/auth/system/devassign`, contains device-specific values for system devices.
- The terminal control database, `/etc/auth/system/ttys.db`, contains device-specific values for authentication (for example, the number of failed login attempts).

Each device in your secure configuration must have an entry in the device assignment database. This database centralizes information about the security characteristics of all system devices. It includes the device pathname and type. By default a wildcard entry exists for

terminals (but not X displays) in the `/etc/auth/system/ttys.db` and `/etc/auth/system/devassign` databases.

The X display entries shipped on the system have `:t_login_timeout#0:` entries in them, in case a site changes its system default login timeout. If wildcard X display entries are needed, they can be created as follows:

```
echo \
 \'*\:*:t_devname=*\:*:t_login_timeout#0:t_xdisplay:chkent:\' \
 | /tcb/bin/edauth -s -dt

echo \'*\:*:v_type=xdisplay:chkent:\' | /tcb/bin/edauth -s -dv
```

## A.9 Enhanced Security Troubleshooting

This section describes problems that can occur on your system and gives guidance on how to avoid or correct from them. It provides you with insight on what is involved in the system startup, so you can examine critical files and programs required for correct system operation. Once the system is in single-user mode, there is no substitute for careful backup procedures. This is the only precaution that will avert serious data loss in your system.

The problems discussed in the following sections will prevent the system from booting.

### A.9.1 Lock Files

The system security databases are critical to correct system operation. These databases use a lock file to synchronize rewrites to security-relevant databases. Before a process rewrites a database entry, it automatically creates the lock file. If the lock file already exists, the program assumes that another process is currently using the database and waits for the lock file to be removed. If the lock file persists and is not modified within a reasonable time period (currently 50 seconds), the program waiting for the lock file removes it and creates a new one, assuming that there has been a system crash or software error.

The system names lock files by appending a `:t` extension to the normal file name.

The system's startup scripts include lines that remove all lock files at system startup. The following files have associated lock files that can prevent correct operation of the system:

- `/dev/console`
- `/etc/auth/system/default`
- `/etc/auth/system/devassign`

## A.9.2 Required Files and File Contents

The following files are required to run the system:

- /tcb/files/auth.db
- /etc/auth/system/ttys.db
- /etc/auth/system/default
- /etc/auth/system/devassign
- /etc/passwd
- /etc/group
- /sbin/rc[023]s
- /dev/console
- /dev/tty\*
- /dev/pts/\*
- /sbin/sulogin
- /sbin/sh
- /vmunix

### A.9.2.1 The /tcb/files/auth.db Database

When the system begins operation, it consults the security databases for various parameters. If any of the databases are corrupt, the system will not boot successfully. If possible, the startup programs report that there is a problem in the databases and start a single-user shell at the system console to allow you to repair the system. In some cases, however, the system will not boot and you must repair the system from standalone procedures described in the manual *System Administration*.

The enhanced (protected) password database entry for root is held in the /tcb/files/auth.db database. If the entry for root is inconsistent, the system enters single-user mode, but assumes default characteristics for all security parameters of the shell it starts.

When the system is in single-user mode, you can create an enhanced (protected) password database entry for root by entering the following command:

```
edauth root
```

The following example shows a typical enhanced (protected) password database entry for root:

```

root:u_name=root:u_id#0:\
 :u_pwd=encrypted_password:\

 :u_minchg#0:u_pickpw:u_nullpw:u_restrict@:\
 :u_maxtries#100:u_lock@:chkent:

```

For a complete explanation of all the fields, see `prpasswd(4)`. The following fields are required for the system to be able to boot:

<i>name</i>	Must contain <code>root</code> .
<i>u_name</i>	Must also be <code>root</code> .
<i>u_uid</i>	Must have a value of 0.
<i>u_pwd</i>	The encrypted version of the password. At authentication, the system checks the entered password against the encrypted version of the password. You can leave this field blank if you are creating the database entry.
<i>chkent</i>	As with all databases, the entry must end with the single word <code>chkent</code> .

The other fields in this entry are informational or are used to guard against unwanted account locking. The system overrides all conditions that can cause the root account to lock when changing to single-user mode.

### A.9.2.2 The `/etc/auth/system/ttys.db` File

The terminal control database must have a valid entry for the system console. The entry for the system console must begin with the word `console` followed by a colon. It must end with the single word `chkent`. The only required field is `t_devname`, which must be set to a value of `console`. For example:

```
console:t_devname=console:chkent:
```

### A.9.2.3 The `/etc/auth/system/default` File

The system default database must have an initial field `default` and must end with `chkent`. There must not be a `:t` lock file associated with this database.

The following example is typical:

```
default:\
 :d_name=default:\
```

```
:d_boot_authenticate@:\
:d_audit_enable@:\
:d_pw_expire_warning#3456000:\
:u_pwd=*\
:u_minchg#0:u_maxlen#20:u_exp#15724800:u_life#31449600:\
:u_pickpw:u_genpwd:u_restrict@:u_nullpw@:\
:u_genchars:u_genletters:u_maxtries#5:u_lock@:\
:t_logdelay#1:t_maxtries#5:t_lock@:t_login_timeout#60:\
:chkent:
```

#### A.9.2.4 The `/etc/auth/system/devassign` File

If the entry for the console is inconsistent, no application can be started. The field must start with the word `console` and end with the word `chkent`. The `v_type` field must be set to `terminal`.

The following example is typical:

```
console:v_devs=/dev/console:v_type=terminal:\
:chkent:
```

#### A.9.2.5 The `/etc/passwd` File

The `/etc/passwd` file is the password database. This file must be present and its format must be correct. No encrypted passwords are updated in this file.

#### A.9.2.6 The `/etc/group` File

The `/etc/group` file is the group database. This file must be present and its format must be correct.

#### A.9.2.7 The `/sbin/rc[023]` Files

The `/sbin/rc[023]` files are used by `init` to change between run levels. Save copies of these files after installation.

#### A.9.2.8 The `/dev/console` File

The `/dev/console` file designates the character device associated with the system console. This file must be present for the system to boot.

#### A.9.2.9 The `/dev/pts/*` and `/dev/tty*` Files

The `/dev/pts/*` and `/dev/tty*` files are pseudoterminal devices used for interprocess communication.



#### A.9.2.10 The /sbin/sulogin File

The `/sbin/sulogin` executable file allows restricting access in single-user mode to those users with the root password.

#### A.9.2.11 The /sbin/sh File

The `/sbin/sh` executable file must be present for the system to start a shell to transition to single-user mode.

#### A.9.2.12 The /vmunix File

The `/vmunix` file is the executable image of the operating system. The boot loading software loads the operating system into memory and transfers control to it at boot time.

### A.9.3 Problems Logging In or Changing Passwords

If users experience problems logging in to the system or changing their passwords, examine the file attributes for the files in the security subset using the `fverify` command. For example, to verify the file attributes for the files in the OSFC2SEC510 subset, enter the following commands:

```
cd /
/usr/sbin/fverify < /usr/.smdb./OSFC2SEC510.inv
```

The file attributes of the local user profile files are examined using the `ls -l` and `authck -pf` commands.

If a user complains of login troubles involving the inability to update the protected profile or to obtain a lock and you are running centralized account management, see Section A.6.

The utilities such as `dxaccounts` and `usermod` share a lock file called `/etc/.AM_is_running`. If the file is present, the utilities warn you.



# B

---

## Secure Shell

The Secure Shell software is client/server software that provides a suite of secure network commands that you can use in addition to or in place of traditional nonsecure network commands. Table B–1 describes the traditional nonsecure network commands and the equivalent Secure Shell commands.

**Table B–1: Traditional Nonsecure Network Commands and Secure Shell Commands**

Task	Traditional Nonsecure Network Command	Secure Shell Command
Execute commands on a remote system	rsh	ssh2
Log in to a remote system	rlogin or telnet	ssh2
Transfer files between systems	rcp or ftp	scp2 or sftp2

This appendix contains the following information:

- Secure Shell servers and clients
- Secure Shell overview
- Configuring the Secure Shell server and client
- Configuring nonsecure network commands to use Secure Shell
- Configuring Secure Shell user authentication
- Managing the Secure Shell server
- Using Secure Shell commands

### B.1 Secure Shell Servers and Clients

A Secure Shell server is a system on which the Secure Shell server software is installed and the Secure Shell `sshd2` daemon is started. The Secure Shell software includes the Secure Shell server software that runs on a system running the Tru64 UNIX Version 5.1A or higher operating system software. The remainder of this appendix refers to a Secure Shell server as server.

A Secure Shell client is a system on which the Secure Shell client software is installed. The Secure Shell software includes the Secure Shell client

software that runs on a system running the Tru64 UNIX Version 5.1A or higher operating system software.

The Secure Shell client software provides the Secure Shell `scp2`, `sftp2`, and `ssh2` commands and other Secure Shell commands to manage the Secure Shell client software. The remainder of this appendix refers to a Secure Shell client as client.

By default, the Tru64 UNIX Secure Shell server and client software are installed when you upgrade to or install the Tru64 UNIX Version 5.1B or higher software and the following message is displayed during the `setld` procedure:

```
Configuring "Secure Shell Base Components" (OSFSSHBASEnnn)
Creating ./etc/ssh2/ssh2_config
Creating ./etc/ssh2/sshd2_config
Creating ./etc/ssh2/ssh_dummy_shell.out
Generating 1024-bit dsa key pair
 Key generated.
1024-bit dsa hostkey
Private key saved to ./etc/ssh2/hostkey
Public key saved to ./etc/ssh2/hostkey.pub
Creating /.ssh2/ssh2_config
```

```
Installation of the Secure Shell Base Components (OSFSSHBASEnnn) subset is complete.
```

## B.2 Secure Shell Overview

When the server is started, the `sshd2` daemon listens on port 22 (by default) for a client to initiate a socket connection. When a client connects, the `sshd2` daemon starts a child process. The child process initiates a public host key exchange with the client. The public host key exchange is a process in which the client and server exchange their public host keys to authenticate their identity to each other. A public host key is created on the server as `/etc/ssh2/hostkey.pub` when you install the Secure Shell software.

The first time a client connects to a server, the user is (by default) prompted to accept a copy of the server's public host key. If the user accepts the key, the server's public host key is copied to the user's `hostkeys` directory on the client. The client uses this public host key to authenticate the server on subsequent connections. You can also copy the server's public host key in advance to the user's `hostkeys` directory on the client as `key_port_servername.pub`. For example, if the server name is `orange`, copy its key as `key_22_orange.pub`.

After the client and server authenticate each other, the child process attempts to authenticate the user. The user must have a valid user account and home directory on the server. If the child process fails to authenticate the user, it will refuse the connection. Secure Shell uses one of the following user authentication methods:

- Password
- Public key
- Host-based (Clients and servers can use host-based authentication only if both systems are running the UNIX operating system software.)

After the child process authenticates the user's identity, the actual connection with the client occurs. The connection includes command execution, encrypted data transfer, and termination of the connection. Once the connection is established, all authentication and communication between the client and server will use the Secure Shell connection. When the connection is terminated, the child process started by the `sshd2` daemon terminates.

## B.3 Configuring the Secure Shell Server and Client

You configure how servers and clients communicate by setting values to keywords in the `/etc/ssh2/sshd2_config` file for the server and in the `/etc/ssh2/ssh2_config` file for the client. Users can also have their own `$.HOME/.ssh2/ssh2_config` file on the client, where `$.HOME` is the name of the user's home directory.

The `sshd2_config` file and the `ssh2_config` file contain keyword-argument pairs, one per line. Keywords are assigned default values, which you can change. Keywords are case insensitive. Empty lines and lines starting with a number sign ( `#` ) are ignored as comments.

### B.3.1 Configuring the Server

The `/etc/ssh2/sshd2_config` file (or the file specified with the `sshd2 -f` command) contains configuration keywords and values that the `sshd2` daemon reads when it starts. You can override the values for a keyword in the `sshd2_config` file by entering the keyword and value on the command line when you start the `sshd2` daemon; however, values set this way are reset to the value in the `/etc/ssh2/sshd2_config` file when the `sshd2` daemon restarts.

If you modify the `sshd2_config` file while the `sshd2` daemon is running, you must reset the `sshd2` daemon to implement the change. Changes made this way apply only to new client connections. See Section B.6.1 for information on resetting the `sshd2` daemon.

Example B-1 is a sample `/etc/ssh2/sshd2_config` file. See `sshd2_config(4)` for a description of each keyword in the `sshd2_config` file.

### Example B-1: Sample sshd2\_config File

---

```
sshd2_config
SSH version Server Configuration File
##
General

VerboseMode no
QuietMode yes
AllowCshrcSourcingWithSubsystems no
ForcePTYAllocation no
SyslogFacility AUTH
SyslogFacility LOCAL7

Network

Port 22
ListenAddress 0.0.0.0
RequireReverseMapping no
MaxBroadcastsPerSecond 0
MaxBroadcastsPerSecond 1
NoDelay yes
KeepAlive yes
MaxConnections 50
MaxConnections 0
0 == number of connections not limited

Crypto

Ciphers AnyCipher
Ciphers AnyStd
Ciphers AnyStdCipher
Ciphers 3des
MACs AnyMAC
MACs AnyStd
MACs AnyStdMAC
RekeyIntervalSeconds 3600

User

PrintMotd yes
CheckMail yes
UserConfigDirectory "%D/.ssh2"
UserConfigDirectory "/etc/ssh2/auth/%U"
UserKnownHosts yes
LoginGraceTime 600
PermitEmptyPasswords no
StrictModes yes

User public key authentication
```

## Example B-1: Sample sshd2\_config File (cont.)

---

```
HostKeyFile hostkey
PublicHostKeyFile hostkey.pub
RandomSeedFile random_seed
IdentityFile identification
AuthorizationFile authorization
AllowAgentForwarding yes

Tunneling

AllowX11Forwarding yes
AllowTcpForwarding yes
AllowTcpForwardingForUsers sjl, cowboyneal@slashdot.org
DenyTcpForwardingForUsers "2[:isdigit:]*4, peelo"
AllowTcpForwardingForGroups privileged_tcp_forwarders
DenyTcpForwardingForGroups coming_from_outside

Authentication
Hostbased and PAM are not enabled by default.

BannerMessageFile /etc/ssh2/ssh_banner_message
BannerMessageFile /etc/issue.net
PasswordGuesses 1
AllowedAuthentications hostbased,publickey,password
AllowedAuthentications publickey,pam-1@ssh.com
AllowedAuthentications hostbased,publickey,password
RequiredAuthentications publickey,password
SshPAMClientPath ssh-pam-client

Host restrictions

AllowHosts localhost, foobar.com, friendly.org
DenyHosts evil.org, aol.com
AllowSHosts trusted.host.org
DenySHosts not.quite.trusted.org
IgnoreRhosts no
IgnoreRhosts no
IgnoreRootRHosts no
(the above, if not set, is defaulted to the value of IgnoreRHosts)

User restrictions

AllowUsers "sj*,s[:isdigit:]##,s(jl|amza)"
DenyUsers skuuppa,warezdude,31373
DenyUsers don@untrusted.org
AllowGroups staff,users
DenyGroups guest
```

### Example B-1: Sample sshd2\_config File (cont.)

---

```
PermitRootLogin nopwd
 PermitRootLogin yes

SSH1 compatibility

Ssh1Compatibility
Sshd1Path

Chrooted environment

ChRootUsers ftp,guest
ChRootGroups guest

subsystem definitions

 subsystem-sftp sftp-server
```

---

## B.3.2 Configuring the Client

The `/etc/ssh2/ssh2_config` file contains configuration keywords and values that the client software reads when it starts. Each user can also have their own `$HOME/.ssh2/ssh2_config` file, where `$HOME` is the name of the user's home directory. The `/etc/ssh2/ssh2_config` file is read first, then the user's version is read. The last obtained value for a keyword is used, except for the `EnforceSecureRutils` keyword. See Section B.4 for more information about the `EnforceSecureRutils` keyword.

Example B-2 is a sample `/etc/ssh2/ssh2_config` file. See `ssh2_config(4)` for a description of each keyword in the `ssh2_config` file.

### Example B-2: Sample ssh2\_config File

---

```
ssh2_config
SSH version Client Configuration File
##

The "*" is used for all hosts, but you can use other hosts as
well.
*:

Tru64 UNIX specific
Secure the r* utilities (no, yes)
 EnforceSecureRutils no
```



## Example B-2: Sample ssh2\_config File (cont.)

---

```
General

 VerboseMode no
QuietMode yes
DontReadStdin no
BatchMode yes
Compression yes
ForcePTYAllocation yes
GoBackground yes
EscapeChar ~
PasswordPrompt "%U%H's password: "
 PasswordPrompt "%U's password: "
 AuthenticationSuccessMsg yes

Network

 Port 22
 NoDelay no
 KeepAlive yes
SocksServer socks://login@socks.ssh.com:1080/11.12.0.0/
16,19.74.23.0/24

Crypto

 Ciphers AnyStdCipher
 MACs AnyMAC
 StrictHostKeyChecking ask
RekeyIntervalSeconds 3600

User public key authentication

 IdentityFile identification
 AuthorizationFile authorization
 RandomSeedFile random_seed

Tunneling

GatewayPorts yes
ForwardX11 yes
ForwardAgent yes

Tunnels that are set up upon logging in

LocalForward "110:pop3.ssh.com:110"
RemoteForward "3000:foobar:22"

SSH1 Compatibility
```

### Example B-2: Sample ssh2\_config File (cont.)

---

```
 Ssh1Compatibility yes
 Ssh1AgentCompatibility none
Ssh1AgentCompatibility traditional
Ssh1AgentCompatibility ssh2
Ssh1Path /usr/local/bin/ssh1

Authentication
Hostbased is not enabled by default.

AllowedAuthentications hostbased,publickey,password
 AllowedAuthentications publickey,password

For ssh-signer2 (only effective if set in the global configuration
file, usually /etc/ssh2/ssh2_config)

DefaultDomain foobar.com
SshSignerPath ssh-signer2

Examples of per host configurations

#alpha*:
Host alpha.oof.fi
User user
PasswordPrompt "%U:s password at %H: "
Ciphers idea

#foobar:
Host foo.bar
User foo_user
```

---

## B.4 Configuring Nonsecure Network Commands to Use Secure Shell

You can configure the `rsh`, `rlogin`, and `rcp` commands and applications that use the `rcmd()` function to automatically use a Secure Shell connection by enabling the `EnforceSecureRutils` keyword in the `/etc/ssh2/ssh2_config` file or in a user's `$HOME/.ssh2/ssh2_config` file. By default, the `EnforceSecureRutils` keyword is disabled. If the `EnforceSecureRutils` keyword is enabled in the `/etc/ssh2/ssh2_config` file, it is enabled for all users overriding the setting in the user's `$HOME/.ssh2/ssh2_config` file. If the `EnforceSecureRutils` keyword is disabled in the

`/etc/ssh2/ssh2_config` file, a user can enable it in their `$HOME/.ssh2/ssh2_config` file.

The following are considerations if you enable the `EnforceSecureRutils` keyword:

- The `sshd` daemon runs and spawns the `rcmd` child process; the `rshd` and `rlogind` daemons do not run.
- The `EnforceSecureRutils` keyword requires the `AllowTcpForwarding` keyword to be enabled in the `/etc/ssh2/sshd2_config` file (the default). If you enable the `EnforceSecureRutils` keyword, do not disable the `AllowTcpForwarding` keyword.
- The `rsh` and `rcp` commands and applications that use the `rcmd( )` function can use only host-based authentication to authenticate users. See Section B.5.3 for information on host-based authentication.
- The `rlogin` command can use host-based or password user authentication to authenticate users. See Section B.5.3 for information on host-based authentication. See Section B.5.1 for information on password authentication.

## B.5 Configuring Secure Shell User Authentication

You must configure the client and server to use the same type of user authentication, which can be any or all of the following:

- Password
- Public key
- Host-based (Clients and servers can use host-based authentication only if both systems are running the UNIX operating system software.)

You configure the type of user authentication that the client and server use by assigning values to the `AllowedAuthentications` keyword in the server's `/etc/ssh2/sshd2_config` file and in the client's `/etc/ssh2/ssh2_config` file or in the user's `$HOME/.ssh2/ssh2_config` file. The user authentication methods are used in the order in which they are listed for the `AllowedAuthentications` keyword. For example, if `hostbased` is listed first, the server will try `hostbased` authentication before trying the next listed authentication. The first successful authentication is the one used. By default, the server tries public key authentication, then password authentication.

## B.5.1 Configuring Password Authentication

Password authentication requires that a user have a password-protected user account that can be authenticated by a Tru64 UNIX password authentication method. Tru64 UNIX password authentication methods include:

- BSD
- Enhanced security
- Network Information Service (NIS)
- Lightweight Directory Access Protocol (LDAP)
- Kerberos

To use password authentication, set the value of the `AllowedAuthentications` keyword to include `password` (the default) in the `/etc/ssh2/sshd2_config` file and in the `/etc/ssh2/ssh2_config` file; for example:

```
AllowedAuthentications password
```

By default, users are allowed only one password attempt. The number of failed password attempts is defined by the value assigned to the `PasswordGuesses` keyword in the `/etc/ssh2/sshd2_config` file.

If you change the value of a keyword in the `/etc/ssh2/sshd2_config` file, you must enter the following command to reset the `sshd2` daemon:

```
/sbin/init.d/sshd reset
```

When using password authentication, the password prompt will be displayed on `stderr`. If `stderr` is redirected, it might appear that the login command has stopped responding while it waits for the password to be entered.

The way in which users are prompted for their password on a Tru64 UNIX Secure Shell client depends on whether or not a Secure Shell connection is used. If the server is running the `sshd` daemon, users are prompted as follows:

```
username's password:
```

If the server is not running the `sshd` daemon, users are prompted as follows:

```
Password:
```

## B.5.2 Configuring Public Key Authentication

Public key authentication requires that a user have a pair of keys. A public key and a private key. These keys are used to authenticate the user and to encrypt and decrypt the data exchanged between clients and servers. The

public key is published and distributed to the servers with which the user communicates. The private key is kept secret on the local client and never published or distributed. What one public or private key does, only the other associated public or private key can undo.

Public key authentication requires configuration on the client and on the server.

---

**Note**

---

A user's private key and public key are not the same as the server's private host key (`/etc/ssh2/hostkey`) and public host key (`/etc/ssh2/hostkey.pub`). The server's private and public host keys were created when you installed the Secure Shell software; they are used to authenticate the server. A user creates private and public keys; they are used to authenticate the user.

---

### B.5.2.1 Configuring Public Key Authentication on the Client

Follow these steps to configure public key authentication on a Tru64 UNIX Secure Shell client:

1. Set the value of the `AllowedAuthentications` keyword in the `/etc/ssh2/ssh2_config` file to include `publickey` (the default); for example:

```
AllowedAuthentications publickey,password
```

2. Instruct the user to log in to their user account and create a public and private key pair by entering the `ssh-pubkeymgr` command. The user will be prompted for the following information:

```
$ ssh-pubkeymgr
Setting host to hostname
Checking for publickey authentication to be enabled in the
client config..
Your client configuration is all set.

Checking for publickey authentication to be enabled in the
server config..
Your server configuration is all set.

Checking for existing user public keys..
Couldn't find your DSA keypair.. I'll generate you a new set..
Running ssh-keygen2... don't forget to give it a passphrase!

Generating 1024-bit dsa key pair
 2 00o.o0o.o0o.
Key generated.
```

```
1024-bit dsa, username@hostname.fqdn, date time -0500
Passphrase : secret passphrase
Again : secret passphrase
Private key saved to $HOME/.ssh2/id_dsa_1024_a
Public key saved to $HOME/.ssh2/id_dsa_1024_a.pub
If you are logging in from this computer, you need to have an
identification file that defines what private keys will be recognized
when you login. By default, this should be id_dsa_1024_a.
```

Creating your identity file..

Creating your authorization file..

Creating your local host public key..

The next section allows you to add hosts that you wish to login
from using public key authentication.

Do you want to add any hosts to your authorization file?
(Default: yes) **yes**

Type in user and hostname, press return after each one.

```
Add which user? username
Add which host? hostname
You added username at servername
as a trusted login.
Press return to continue or Ctrl-D to exit.
^D
All the new files are in your $HOME/.ssh2 directory.
```

Now that you have your public keypair generated, you can
copy your public key up to remote hosts so you can login to
them using public key authentication. You also need to add this key,
*username-hostname.pub*,
to the *~/.ssh2/authorization* file on the server.

```
Do you want to upload username-hostname key to a
remote host? (Default: yes)
Upload to which host? hostname.fqdn
Which user account? username
Now running scp2 to connect to hostname.fqdn..
Most likely you'll have to type a password :)
Host key not found from database.
Key fingerprint:
xolog-bivic-nomeb-behas-zanet-matuc-hedol-moliv-videl-melal-cixox
You can get a public key's fingerprint by running
% ssh-keygen -F publickey.pub on the keyfile.
Are you sure you want to continue connecting (yes/no)? yes
Host key saved to $HOME/.ssh2/hostkeys/key_22_hostname.fqdn.pub
```

```
host key for hostname.fqdn, accepted by username date time -0 500
username@hostname.fqdn's password: password 4
username-hostname.pub |755B|0.7 kB/s|TOC: 00:00:01|100% 5
```

Press return to upload to more hosts or Ctrl-D to exit.  
^D

- 1** Enter a passphrase assigned for the keypair.
- 2** Enter *yes* to add a host entry in the `authorization` file if you will access your user account on the client from a remote host that uses public host key authentication. A host entry identifies the user name and name of a remote host from which you will access your user account on the client.  
  
If you enter *yes*, you are prompted for the user name by which you want to be authenticated, then for the name of the remote host. Host names should be entered with their fully qualified domain name (fqdn). For example, if the remote host's name is `orange` and its fully qualified domain name is `color.art.com`, enter `orange.color.art.com`  
  
A host entry is added to the `authorization` file in the following format:  
  

```
Key username-hostname.pub
```
- 3** Enter *yes* to copy your public key to a user account on a remote host from which you will access your account on the client. If you enter *yes*, you are prompted for the name of the remote host and the user account to which your public key will be copied. This is usually the user name and the name of the remote host for which you added a host entry in the `authorization` file. Host names should be entered with their fully qualified domain name.
- 4** The user's public key will be copied to the specified user account on the remote host. Users must enter a password for the specified user account on the remote host because, by default, password authentication is the only authentication available at this time.
- 5** A status message is displayed that shows the results of copying the public key to the remote host.

The `ssh-pubkeymgr` command creates:

- A directory called `$HOME/.ssh2` for the user on the client. (`$HOME` is the name of the user's home directory.)
- The key pair:

- The `$HOME/.ssh2/id_dsa_1024_a` file contains the user’s private key. Only the user for which the key was created should have access to this file.
- The `$HOME/.ssh2/id_dsa_1024_a.pub` file and the `$HOME/.ssh2/username-hostname.pub` file contain the user’s public key. The `username-hostname.pub` file is the file that will be copied to servers that use public key authentication and to which the user will connect.
- The `$HOME/.ssh2/identification` file contains the following entry that identifies the name of the user’s private key file:
 

```
IdKey id_dsa_1024_a
```
- The `$HOME/.ssh2/authorization` file contains the names of public keys for remote hosts from which the user can access their user account on the local host.

### B.5.2.2 Configuring Public Key Authentication on the Server

Follow these steps to configure public key authentication on the server:

1. Set the value of the `AllowedAuthentications` keyword in the `/etc/ssh2/sshd2_config` file to include `publickey` (the default); for example:

```
AllowedAuthentications publickey,password
```

If you change the value of the `AllowedAuthentications` keyword, you must enter the following command to reset the `sshd2` daemon:

```
/sbin/init.d/sshd reset
```

2. If the client and server are not the same system, create a `$HOME/.ssh2` directory on the server. (`$HOME` is the name of the user’s home directory for which public key authentication is being configured.)
3. If necessary, create an authorization file in the user’s `$HOME/.ssh2` directory. Add to the authorization file a host entry for the client. A host entry identifies the name of a public key for a client from which the user can access their user account on the local host.

A host entry is added to the authorization file in the following format:

```
Key username-clienthostname.pub
```

4. If the user did not upload their public key from the client to the remote server when public key authentication was configured, copy from the client the user’s public key file (`$HOME/username-hostname.pub`) to the user’s `$HOME/.ssh2` directory on the server.



### B.5.2.3 Accessing a Remote Server

Example B-3 shows sample output when logging in to a remote server that is using public key authentication.

#### Example B-3: Public Key Authentication Login Output

---

```
$ssh server_name
Passphrase for key "/home/user/.ssh2/id_dsa_1024_a
with comment "1024-bit dsa, created by user@Local
date time +0200":
```

---

### B.5.2.4 Restricting User Access

You can restrict a user to execute only certain UNIX commands when a user logs in to a server that is using public key authentication. To restrict users, enter the allowable UNIX command(s) under the `Key` entry in the user's authorization file on the server. For example, the following entry would use public key authentication to authenticate the user, execute the `ls` command in the login directory, then return the user to the local host prompt:

```
Key username-hostname.pub
Command ls
```

### B.5.2.5 Managing Passphrases

The passphrase is not the user's Tru64 UNIX user account password. The passphrase is the secret text that the user entered with the `ssh-pubkeymgr` command to create a public and private key pair. The passphrase is used only by the client and server to exchange information about the user.

Users are prompted for the passphrase when they enter a Secure Shell command on a server that uses public key authentication. Users can configure the server so that it does not repeatedly prompt for a user's passphrase during a session by running the Secure Shell agent and loading their private keys into the Secure Shell agent. When the agent is running, all key-related operations are directed to the agent. The Secure Shell agent terminates when the user logs out or stops the Secure Shell agent.

Follow these steps to run the Secure Shell agent:

1. Log in to the server.
2. Start the Secure Shell agent:

```
ssh-agent2 $SHELL
```

The `$SHELL` environment variable identifies the user's login shell.

Alternatively, users can automatically start the Secure Shell agent when logging in to the server by adding the `ssh-agent2 $SHELL` command to their login file; for example, their `.login` file.

The Secure Shell agent invokes the specified shell as a child process, and the shell prompt appears.

3. Load the private keys into the Secure Shell agent:

```
$ ssh-add2
```

The `ssh-add2` command prompts the user for the passphrase.

### B.5.3 Configuring Host-Based Authentication

Host-based authentication is an authentication method that is based on system identification, not password or passphrase identification. Clients and servers can use only host-based authentication if both systems are running the Tru64 UNIX operating system software.

Host-based authentication requires:

- A fully qualified domain name in the `/etc/hosts` file for the local host and, if applicable, the local TruCluster alias. See *Network Administration: Connections* and `hosts(4)` for more information on fully qualified domain names.
- Users to have a file called `.rhosts` or `.shosts` in their home directory that includes a host name and fully qualified domain name entry for each remote host from which they will access the local host. The `.shosts` file is read only by the Secure Shell server. If both files exist, the Secure Shell server reads the `.rhosts` file first, then the `.shosts` file. If either of these files allows access for a particular connection, a Secure Shell connection is used, even if the other file forbids it.
- The public host key files for the local host and the remote hosts with which the local host communicates to be in the `/etc/ssh2/knownhosts` directory on the local host.

Because most communication between hosts is reciprocal and a host can be a client and a server, follow these steps on all hosts that will communicate by using host-based authentication:

1. Set the value of the following keywords in the `/etc/ssh2/sshd2_config` file:
  - Make sure that the value of the `AllowedAuthentications` keyword includes `hostbased` (the default). If there are other entries, set `hostbased` as the first entry. For example:

```
AllowedAuthentications hostbased,password
```

- Make sure that the value of the `IgnoreRhosts` keyword is set to `no`. For example:

```
IgnoreRhosts no
```

If you change the value of a keyword in the `/etc/ssh2/sshd2_config` file, enter the following command to reset the `sshd2` daemon:

```
/sbin/init.d/sshd reset
```

2. Set the value of the following keywords in the `/etc/ssh2/ssh2_config` file or in a user's `$HOME/.ssh2/ssh2_config` file:

- Make sure that the value of the `AllowedAuthentications` keyword includes `hostbased`. If there are other entries, set `hostbased` as the first entry. For example:

```
AllowedAuthentications hostbased,password
```

- Set the value of the `DefaultDomain` keyword to the fully qualified domain name for the local host. For example, if the fully qualified domain name for the local host is `color.art.com`, enter:

```
DefaultDomain color.art.com
```

3. To make local host's public host key available in its `/etc/ssh2/known-hosts` directory for other hosts to copy when using host-based authentication, enter:

```
ln -sf /etc/ssh2/hostkey.pub \
/etc/ssh2/knownhosts/hostname.fqdn.ssh-dss.pub
```

The *hostname* is the name of the local host and the *fqdn* is the fully qualified domain name for the local host as defined in the `/etc/hosts` file. For example, if the local host name is `orange` and its fully qualified domain name is `color.art.com`, enter `orange.color.art.com.ssh-dss.pub`.

In a TruCluster Server environment, the cluster alias public host key must be available in the `/etc/ssh2/knownhosts` directory. Enter the following command to make the cluster alias public host key available:

```
ln -sf /etc/ssh2/hostkey.pub \
/etc/ssh2/knownhosts/cluster_alias.fqdn.ssh-dss.pub
```

The *cluster\_alias* is the name of the cluster alias and the *fqdn* is the fully qualified domain name for the cluster alias as defined in the `/etc/hosts` file.

For a host outside of a TruCluster Server environment to connect to a host inside a TruCluster Server environment, copy the cluster alias public host key file (`/etc/ssh2/hostkey.pub`) to the `/etc/ssh2/knownhosts` directory on the host outside of the cluster.

4. For each user who will use host-based authentication, create a file called `.rhosts` or `.shosts` in each user's home directory and add an entry for the local host and an entry for each remote host with which the local host communicates. An entry includes the host name followed by its fully qualified domain name. For example, if the local host name is `orange` and its fully qualified domain name is `color.art.com`, enter:

```
orange.color.art.com
```

The user must be the owner of the `.rhosts` or `.shosts` file in their home directory. Use the `chown` command to set the user as the owner of the `.rhosts` or `.shosts` file and to set the permissions to 600 (read and write by owner only).

5. Make a copy of the public host key file (`/etc/ssh2/known-hosts/hostname.fqdn.ssh-dss.pub`) from each remote host with which the local host communicates is in the `/etc/ssh2/knownhosts` directory on the local host.

To copy public host keys, you use the `ssh-hostbased-setup` command. The `ssh-hostbased-setup` command verifies and, if necessary, copies the public host key from the specified remote host or from the remote hosts listed in the specified file to the `/etc/ssh2/knownhosts` directory on the local host.

To use the `ssh-hostbased-setup` command, enter:

```
ssh-hostbased-setup hostname | filename
```

`hostname` Specifies the name of the remote host for which host-based authentication is being configured or verified.

`filename` Specifies the file that contains the names of remote hosts for which host-based authentication is being configured or verified. The specified file is usually the `.rhosts` file or the `.shost` file.

See `ssh-hostbased-setup(1)` for more information.

## B.6 Managing the Secure Shell Server

This section describes how to:

- Start, stop, restart, and reset the `sshd2` daemon
- Restrict users to their home directories
- Create a public and private key

- Forward TCP/IP ports and X11 data through a Secure Shell connection

### B.6.1 Starting, Stopping, Restarting, and Resetting the sshd2 Daemon

When the sshd2 daemon starts, it uses the configuration information in the `/etc/ssh2/sshd2_config` file (the default). When you start the sshd2 daemon, you can specify configuration options on the command line. Command line configuration options override values in the `/etc/ssh2/sshd2_config` file and are effective only until the sshd2 daemon restarts. To permanently make a configuration change, edit the `/etc/ssh2/sshd2_config` file. See Section B.3.1 for more information on the `/etc/ssh2/sshd2_config` file.

- To start the sshd2 daemon using the configuration information in the `/etc/ssh2/sshd2_config` file, enter:  

```
/sbin/init.d/sshd start
```
- To start the sshd2 daemon and change a value for a keyword in the `/etc/ssh2/sshd2_config` file, enter:  

```
/usr/sbin/sshd keyword value
```
- To stop the sshd2 daemon, enter:  

```
/sbin/init.d/sshd stop
```
- To restart the sshd2 daemon using the configuration information in the `/etc/ssh2/sshd2_config` file, enter:  

```
/sbin/init.d/sshd restart
```
- To reset the sshd2 daemon, enter:  

```
/sbin/init.d/sshd reset
```

### B.6.2 Restricting Users to Home Directories

Use the `ssh-chrootmgr` command to restrict a user to a home directory when the user enters the `ssh2` command or the `sftp2` command.

Follow these steps to restrict a user to a home directory:

1. Enter the `ssh-chrootmgr` command with the name of the users or groups you want to restrict:  

```
ssh-chrootmgr username username username
```
2. Edit the `/etc/ssh2/sshd2_config` file and update the value of the `ChRootUsers` entry to include the restricted users (or the `ChRootGroups` entry to include the restricted groups) specified in step 1.

3. Reset the `sshd2` daemon:  

```
/sbin/init.d/sshd reset
```
4. Edit the `/etc/passwd` file and configure `/bin/ssh-dummy-shell` as the shell for the restricted users and users in the groups specified in step 1.

See `ssh-chrootmgr(1)` and `ssh-dummy-shell(1)` for more information.

### B.6.3 Creating a Public and Private Host Key

A public host key and private host key are created when you install the Secure Shell software. You need to create a new public host key and private host key only if you want to change them.

Follow these steps to create a new public host key and private host key:

1. Stop the `sshd2` daemon:  

```
/sbin/init.d/sshd stop
```
2. Create the host keys:  

```
ssh-keygen2 -P /etc/ssh2/hostkey
```
3. Start the `sshd2` daemon:  

```
/sbin/init.d/sshd start
```

---

**Note**

---

Users who have a copy of the server's old public host key will get a warning message when connecting to a server with a new public host key. Users can delete the server's old public host key file from their `hostkeys` directory on the client. The next time the client connects to the server, the user will be prompted to accept a copy of the server's new public host key.

---

### B.6.4 Forwarding TCP/IP Ports and X11 Data Through a Secure Shell Connection

You can configure TCP/IP ports and X11 connections to forward data by using a Secure Shell connection.

#### B.6.4.1 TCP/IP Port Forwarding

Forwarding, or tunneling, is a way to forward otherwise insecure TCP data through a Secure Shell connection. For example, you can configure POP3, SMTP, and HTTP connections to use a Secure Shell connection.

Follow these steps to configure TCP/IP port forwarding:

1. Set the value of the `AllowTcpForwarding` keyword to `yes` in the `/etc/ssh2/sshd2_config` file (the default).

If you change the value of the `AllowTcpForwarding` keyword, you must enter the following command to reset the `sshd2` daemon:

```
/sbin/init.d/sshd reset
```

2. Configure the TCP/IP port to forward. There are two kinds of TCP/IP port forwarding:

- Local TCP/IP port forwarding (also known as outgoing tunnels)

Local TCP/IP port forwarding forwards data going to a local TCP/IP port to a specified remote TCP/IP port. To forward all data going to the specified `local_port` on the local system to the specified `remote_port` on the remote system, enter:

```
ssh2 -L local_port:remote:remote_port user@remote
```

- Remote TCP/IP port forwarding (also known as incoming tunnels)

Remote TCP/IP port forwarding forwards data going to a remote TCP/IP port to a specified local TCP/IP port. To forward all data going to the specified `remote_port` on the remote system to the specified `local_port` on the local system, enter:

```
ssh2 -R remote_port:local:local_port user@remote
```

#### B.6.4.2 X11 Forwarding

Follow these steps to enable X11 forwarding:

1. Set the value of the `ForwardX11` keyword to `yes` in the `/etc/ssh2/sshd2_config` file (the default).

If you change the value of the `ForwardX11` keyword, you must enter the following command to reset the `sshd2` daemon:

```
/sbin/init.d/sshd reset
```

2. Test the connection by logging in to the client and entering an X11 command. For example, to start the X clock program, enter:

```
xclock &
```

If the X clock window is displayed, X11 forwarding is working.

---

**Note**

---

Do not set the `DISPLAY` variable on the client. Doing so will disable encryption. (X connections forwarded through Secure Shell use a special local display setting.)

---

## B.7 Using the Secure Shell Commands

This section describes how to use Secure Shell commands to:

- Copy files between clients and servers
- Log in and execute commands on a server

---

**Note**

---

When a client connects to a server for the first time, the user is prompted to accept a copy of the server's public host key. If the user accepts the key, a copy of the server's public host key is copied to the user's `hostkeys` directory on the client. The client uses this public host key to authenticate the server on subsequent connects.

---

### B.7.1 Copying Files Between Clients and Servers

You can use the following Secure Shell commands to copy files:

- The `scp2` command
- The `sftp2` command

#### B.7.1.1 Using the `scp2` Command

Use the `scp2` or `scp` command on a client to copy files to and from a server. The installation process creates a symbolic link from the `scp` command executable to the `scp2` command executable. The `scp2` command runs with normal user privileges.

- To copy files from a local system to a remote system, enter:

```
scp2 /directory/file user@system:/directory/file
```

- To copy files from a remote system to a remote system, enter:

```
scp2 user@system:/directory/file user@system:/directory/file
```

Relative paths can also be used; they are interpreted relative to the user's home directory.

See `scp2(1)` for more information about the `scp` command.



### B.7.1.2 Using the sftp2 Command

Use the `sftp2` command or the `sftp` command on a client to copy files to and from a server. The installation process creates a symbolic link from the `sftp` command executable to the `sftp2` command executable.

The `sftp2` command functions like the `ftp` command but does not use the `ftp` daemon or the `ftp` client for its connections. The `sftp2` command runs with normal user privileges.

To use the `sftp2` command to connect to a remote host to copy files, enter:

```
sftp2 [options] hostname
```

You can also use the `scp2` syntax with the `sftp2` command. See Section B.7.1.1, `scp2(1)`, `sftp2(1)` for more information.

### B.7.2 Logging In and Executing Commands on a Server

Use the `ssh2` command or the `ssh` command on the client to securely log in and execute commands on a server. The installation process creates a symbolic link from the `ssh` command executable to the `ssh2` command executable.

To execute commands on a server, enter:

```
ssh2 [options] server_name [command]
```

When a user successfully logs in to a server, the `sshd2` daemon:

1. Runs with the user's privileges.
2. Sets up a basic environment.
3. Changes to the user's home directory.
4. Runs the user's shell.

See `ssh2(1)` for more information about the `ssh2` command.



# C

---

## Single Sign On

Single Sign On (SSO) is an optional client/server software that uses Kerberos technology to provide a secure communication when using the `ftp`, `rcp`, `rlogin`, `rsh`, and `telnet` network commands and applications that use Kerberos.

This appendix contains the following information:

- Kerberos servers and clients
- Kerberos authentication process
- Upgrading the SSO software
- Installing and configuring the SSO software
- SSO configuration files on Tru64 UNIX
- Creating accounts and groups
- Managing the SSO software
- Troubleshooting the SSO software

### C.1 Kerberos Servers and Clients

A Kerberos server is a system on which the Kerberos server software is installed. Kerberos server technology is not provided with the Tru64 UNIX operating system software. A Kerberos server, for example a Windows 2000 Server, must exist in the intranet. The remainder of this appendix refers to a Kerberos server as server.

A Kerberos client is a system on which the Kerberos client software is installed. The Tru64 UNIX Single Sign On (SSO) software provides the Kerberos client software that runs on a system running the Tru64 UNIX Version 5.1A or higher operating system software. The Kerberos client software provides Kerberos versions of the `ftp`, `rcp`, `rlogin`, `rsh`, and `telnet` commands and the `rcmd` function. The remainder of this appendix refers to a Kerberos client as client.

## C.2 Kerberos Authentication Process

A Kerberos network is divided into security domains called realms. Each realm has a primary server, can have secondary servers, and implements its own security policy.

Administrators populate and maintain a principal database on the server. The principal database contains a key (entry) for each principal (user, service, application, and host) that contains information about a principal, including a secret value, such as a password. The server uses the key information to authenticate a principal. See your Kerberos server documentation for more information about the principal database.

Realms are hierarchical. Each realm can have child realms, and each realm can have a parent. This allows organizations implementing Kerberos to have different levels of security for different information classes within the organization and allows realms that have no direct contact to share authentication information. All principal names in a realm must be unique. The information security policy is the same for all principals within a realm.

When you install the Tru64 UNIX SSO software, a Security Integration Architecture (SIA) module is installed that directs principal authentication requests to a Kerberos server.

On Tru64 UNIX clients, you can populate and maintain a file called the service key table. The service key table contains a key that you extract from the principal database for each principal (host) that was previously authenticated to communicate with the client. The client reuses the key information to reauthenticate a principal on subsequent requests. That is, after the server initially authenticates a principal, a client can reauthenticate a principal if the principal's key is in its service key table.

## C.3 Upgrading the SSO Software

To upgrade the SSO Version 1.0 software, you must deinstall previously installed SSO subsets and install the new SSO software. For example, to deinstall the SSO Version 1.0 software (w2kss0100), enter:

```
/usr/sbin/setld -d w2kss0100
```

## C.4 Installing and Configuring the SSO Software

You must install some SSO software on the Tru64 UNIX system and some on the Windows 2000 Server system.

You must install the SSO software on the Windows 2000 Server system before installing on the Tru64 UNIX system.

The SSO software that you install on the Tru64 UNIX operating system installs a Security Integration Architecture (SIA) mechanism that directs authentication requests to the Windows 2000 Active Directory.

The SSO software that you install on the Windows 2000 Server extends the Active Directory to include Tru64 UNIX user account and group attributes such as user login name, User ID (UID), Group ID (GID), a comment, a path to a home directory, and a login shell. See *System Administration* for more information on Tru64 UNIX user account and group attributes.

Note the following considerations before you install the SSO software:

- The Active Directory name is case sensitive and must have been entered in lower-case letters. See your Windows documentation if you need to change the case of your Active Directory name.
- You cannot install the distributed computing environment (DCE) software and SSO on the same system, because SSO and DCE implement different versions of Kerberos with different system setup requirements that are not compatible with each other.
- If the Tru64 UNIX system will be running the Advanced Server for UNIX (ASU) Version 5.1A or higher software and the SSO Version 2.0 or higher software, there will be a machine account name conflict. To resolve the name conflict, you must configure the ASU server so that it does not use the host name as the ASU server name (the default). The SSO software creates a machine account in the Active Directory that matches the host name of the Tru64 UNIX system. If the ASU server also uses the host name as the ASU server name, which it does by default, it will overwrite the account created by the SSO software and will cause SSO functionality to fail.

See the ASU *Release Notes* for information on how to configure the ASU server so that it does not use the host name as the ASU server name.

#### **C.4.1 Installing and Configuring the SSO Software on the Windows 2000 System**

When you install the SSO software on a Windows 2000 domain controller, the SSO software:

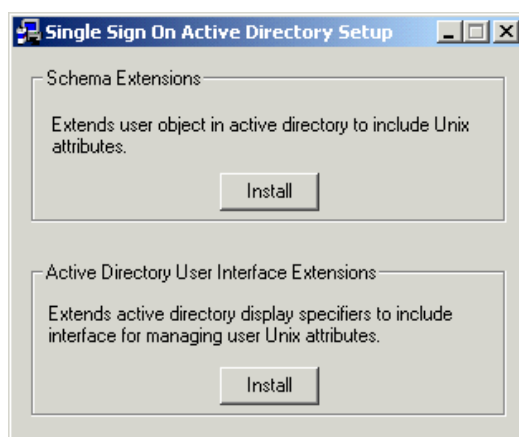
- Extends the Active Directory schema to include Tru64 UNIX user account and group attributes. Because the Active Directory is replicated, all Windows 2000 Servers in the domain will receive a copy of the extended Active Directory. See your Windows documentation for information about the Active Directory.
- Updates the Microsoft Management Console (MMC) to include a Tru64 UNIX property page for users and groups.

You must install SSO software on each Windows 2000 Server from which you want to create user accounts and groups that include Tru64 UNIX attributes. The SSO software updates the MMC on that system to include a Tru64 UNIX property page for users and groups.

#### C.4.1.1 Extending the Active Directory Schema

Follow these steps to extend the Active Directory schema on the Windows 2000 domain controller:

1. Insert the Tru64 UNIX base CD-ROM into the CD-ROM drive on the Windows 2000 domain controller.
2. In the Windows Explorer window, expand the icon associated with the CD-ROM drive, then expand the `Windows2000_SSO` folder, and then expand the `kit/windows_kit` folder.
3. Double-click on the `setup.exe` file. The following window is displayed:



ZK-1674U-AI

4. In the Schema Extensions box, click on the Install button to extend the Active Directory to include Tru64 UNIX attributes. In the Active Directory User Interface Extensions box, click on the Install button to update the MMC to include a Tru64 UNIX property page.

Informational messages about the installation are displayed in a command prompt window. This window automatically closes at the end of a successful installation.

### C.4.1.2 Updating the MMC

Follow these steps to update the MMC on a Windows 2000 Server to include the Tru64 UNIX property pages:

1. Insert the Tru64 UNIX base CD-ROM into the CD-ROM drive on the Windows 2000 Server.
2. In the Windows Explorer, expand the icon associated with the CD-ROM drive, expand the `Windows2000_SSO` folder, then expand the `kit/windows_kit` folder, and then the `MMC_Setup` folder.
3. Double-click on the `setup.exe` file and follow the instructions on the screen.

## C.4.2 Installing and Configuring the SSO Software on the Tru64 UNIX System

The SSO software is an optional subset that is included with the base Tru64 UNIX operating system software.

The SSO subset is called `OSFSSOW2Knnn`, Single Sign On for Windows 2000 (Network-Server/Communications). The *nnn* represents the Tru64 UNIX version number. See the *Release Notes* for the current version number.

See *Installation Guide* and `setld(8)` for information on installing subsets.

After you install the SSO software on the Tru64 UNIX system, you configure the SSO software and the Kerberos SIA mechanism. You can optionally configure the SSO software in a TruCluster Server environment

### C.4.2.1 Configuring the SSO Software

You must run the `/usr/sbin/w2ksetup` script to configure the SSO software. The `/usr/sbin/w2ksetup` script will prompt you for the following required information:

- The name and password of a user account with administrative privileges on the Windows 2000 domain controller. The administrative account must be a member of the Administrators, DnsAdmins, Domain Admins, Domain Users, and Users groups.
- The full Windows 2000 domain name, which is usually the same name as the domain name service (DNS) name server. This name is case sensitive and must be in uppercase letters.
- The host name of the Windows 2000 domain controller, such as `w2khost.sso.corp.com`. This name is case sensitive and must be in lowercase letters.

Follow these steps to configure the SSO software:

1. Add an entry in the `/etc/hosts` file for the Windows 2000 Server. See `hosts(4)` for more information.

2. Start the SSO setup script:

```
/usr/sbin/w2ksetup
```

You are prompted to run the SSO setup script.

3. Enter `yes` when asked if you want to run the SSO setup script:

```
Do you want to run the setup script now? [y/n]? y
```

The following information is displayed:

```
Running /usr/sbin/w2ksetup...
/usr/sbin/w2ksetup[79]: 2598 Terminated
```

---

**Note**

---

If you enter `no` when asked if you want to run the SSO setup script, you will return to the command prompt. You can run the SSO setup script later by entering the following command.

```
/usr/sbin/w2ksetup
```

---

You are prompted for the name of the Windows 2000 domain that the Tru64 UNIX system will use for authentication.

4. Press the `Enter` key to use the default name or enter a domain name and press the `Enter` key:

Enter the name of the Windows 2000 domain. This is in the form `domain.com` - Typically the Windows 2000 domain is the same as the DNS domain.

```
Domain: [SSO.CORP.COM]
```

You are prompted for the name of the Windows 2000 host name controller.

5. Press the `Enter` key to use the default name or enter a host name and press the `Enter` key:

```
Enter the hostname of a Windows 2000 domain controller.
Domain Controller: w2khost.sso.corp.com
```

---

**Note**

---

If you are unable to communicate with the Windows 2000 domain controller, check the `/etc/hosts` file for an entry



with the fully qualified domain name of the Windows 2000 domain controller. If it is missing, add it to the file. Also verify that system time on the client and server systems is the same, because a large time difference between the two systems can cause communication problems.

---

To create a machine account for the Tru64 UNIX system on the Windows 2000 domain controller, you must provide the name and password of a user account with administrative privileges on the Windows 2000 KDC.

The following information is displayed:

```
To create the machine account, you must be logged
in as root and have admin Kerberos credentials. For security
reasons Windows 2000 does not allow anyone to authenticate
through Kerberos using the Administrator account. Therefore
you must choose an account other than the Administrator
account that has admin privileges. The username of an account
must be in the Windows 2000 KDC
```

6. Enter the user name and password of an account in the Windows 2000 KDC with administrative privileges:

```
Enter Admin principal: user
Password for user@SSO.CORP.COM:
```

The following information is displayed:

```
Adding unixhost.sso.corp.com to directory...

Extracting host/unixhost.sso.corp.com key...

Updating /etc/ldapcd.conf...
```

```
The machine account has been set up.
You might need to set permissions in Windows 2000
to enable access to the UNIX information in the
user objects. Please see the Windows documentation
for further information.
```

You are prompted to start the Windows 2000 services.

7. Enter **yes** to the following question if you want to start services immediately:

```
Do you want to start the Windows 2000 services now? [y/n]?: y
```

The following information is displayed:

```
LDAP caching daemon (ldapcd) started
```

8. Restart the X server:

```
/sbin/init.d/xlogin restart
```

### C.4.2.2 Configuring the SSO Software in a TruCluster Server Environment

Follow these steps to configure the SSO software in a TruCluster Server environment:

1. Run the `/usr/sbin/w2ksetup` script on each TruCluster Server member (Section C.4.2.1).
2. For each cluster alias, enter the following command on the last TruCluster Server member on which you ran the `/usr/sbin/w2ksetup` script:

```
creacct -h fully_qualified_cluster_alias_name -u
```

This command adds the fully qualified cluster alias name to the Active Directory to allow SSO log in when using the cluster alias.

### C.4.2.3 Adding Other SIA Mechanisms with Kerberos (if required)

The Kerberos SIA mechanism must be the first entry in the SIA mechanism configuration file (`/etc/sia/matrix.conf`). The SSO installation procedure complies with this restriction. However, if you add other SIA security mechanisms, you must use the `/usr/sbin/siacfg` command to remove the Kerberos entry, add the other entries, then add back the Kerberos entry. For example:

1. Remove the Kerberos entry:

```
siacfg -r Kerberos
```
2. Add the new entries:

```
siacfg -a sianew /location
```
3. Add back the Kerberos entry:

```
siacfg -a -P -g sci Kerberos /usr/shlib/libcsfsiad.so
```

## C.5 SSO Configuration Files on Tru64 UNIX

Table C–1 identifies SSO configuration files on Tru64 UNIX.

**Table C–1: SSO Configuration Files**

File	Contains
<code>krb.conf</code>	The host computer's default realm. Associates known realms to their primary and secondary Kerberos servers by host name and network location.
<code>krb.realms</code>	Server names and their associated realm names.
<code>v5srvtab</code>	The principal keys extracted from the principal database on the Kerberos server.

**Table C–1: SSO Configuration Files (cont.)**

File	Contains
<code>.k5login</code>	A list of principals authorized to access a specific user account.
<code>ldapcd.conf</code>	SSO caching parameters that you can tune for performance.
<code>ldapusers.deny</code>	An entry for each local Tru64 UNIX user who is authenticated only by Tru64 UNIX when SSO is installed.

### C.5.1 The `krb.conf` File

The `/krb5/krb.conf` file is a text file that contains Kerberos realm information. The information defines the system's default realm and associates known realms to their primary and secondary Kerberos servers by host name and network location.

If you can configure the name of the Kerberos server by using default naming conventions (that is, the ordering convention or the DNS rotary convention), you do not need to configure and maintain a `krb.conf` file.

If the `krb.conf` file is not found, is empty, or does not list a valid default realm, the Tru64 UNIX operating system converts the host's domain name to uppercase letters and uses that as the default realm name. If the Kerberos server information is missing from the `krb.conf` file, the Tru64 UNIX operating system attempts to locate the Kerberos server when the default naming conventions are in place.

In the `krb.conf` file, each entry must be on a separate line, fields are separated by spaces or tabs, comments are preceded by a number sign (`#`) (characters after a number sign are ignored to the end of line), and blank lines and leading or trailing white space on a line are ignored.

The order of entries in the `krb.conf` file is important, because it identifies the order in which Kerberos servers are used. Applications read the entries one at a time in the listed order when attempting to connect to a Kerberos server. Secondary Kerberos servers are used when another Kerberos server is unavailable or a network timeout has occurred (for example, during the authentication sequence when the network connection between the client and a Kerberos server is interrupted).

Example C–1 shows a sample `krb.conf` file.

### Example C–1: Sample krb.conf File

---

```
BIZ.COM [1]
BIZ.COM shoe.biz.com admin server [2]
BIZ.COM sneakers.biz.com [3]
BIZ.COM boot.biz.com
FOOTWEAR.BIZ.COM leather.footwear.biz.com admin server [4]
```

---

The entries in Example C–1 create the following configuration:

- [1] The first line identifies the name of the system’s default realm. Realm names are in uppercase letters to distinguish them visually from domain names. You must type the correct case for the realm name if your site does not follow the uppercase convention. In the sample, `BIZ.COM` is the name of the system’s default realm.
- [2] The second line identifies the realm name and fully qualified domain name (FQDN) of the primary Kerberos server for the realm. In Example C–1, `shoe.biz.com` is the FQDN of the primary Kerberos server.
- [3] The next lines identify the realm name and the FQDN of the secondary Kerberos servers in the realm. In Example C–1, `sneakers.biz.com` and `boot.biz.com` are the names of secondary Kerberos servers.
- [4] The next line identifies the name of the realm where interrealm authentication is performed. Interrealm authentication is when a realm accepts authentication from other realms without reauthentication. The line also identifies the FQDN of the Kerberos server followed by `admin server`, and optionally a `tcp/port #`.

By default, UDP is the default communication protocol and does not need to be specified. If the Kerberos server uses TCP for its communication protocol, you must specify `tcp` and the `port #` that TCP uses. To specify a port value, use a numeric value or a service name listed in the `/etc/services` file, such as `tcp/88` or `tcp/kerberos5`.

In Example C–1, `FOOTWEAR.BIZ.COM` is the realm name and `leather.footwear.biz.com` is the Kerberos server. The server is using the default UDP communication protocol.

See `krb.conf(4)` for more information.

## C.5.2 The krb.realms File

The `/krb5/krb.realms` file is a text file that contains Kerberos server names and their associated Kerberos realm names. Secured applications use the `krb.realms` file to determine the realm from which to request a ticket.

By default, the Tru64 UNIX operating system converts the server's name to uppercase letters and uses that as the realm name. In this case, you do not need to configure and maintain a `krb.realms` file.

Wildcards are special characters in the `krb.realms` file that associate multiple servers to a single realm by using one entry.

There are two permitted wildcard characters:

- Use an asterisk (\*) before a domain name to specify all servers that have that domain root name. For example, `*.biz.com` specifies all servers in all domains ending in `biz.com`, such as `footwear.exec.biz.com`.
- Use a question mark (?) in the first field before a server or domain name to specify any letter. For example, `???footwear.biz.com` identifies any server in the `biz.com` domain that has a name with any three letters preceding `footwear`, such as `bigfootwear.biz.com`.

When secure applications search the `krb.realms` file, they check for a matching server name, then a matching domain name. If they do not find a match, they check for a wildcard match. If no associated entry applies or if the `krb.realms` file does not exist, the secure application converts the server's domain name to uppercase letters and uses that as the default.

The order of the entries in the `krb.realms` file is not important. Each entry must be on a separate line and requires two fields, separated by a space or a tab and formatted as follows:

- The first field specifies a server name. You can use a domain name to map each server in a domain to the same realm name. You must precede a domain name with a period.
- The second field specifies the associated realm name. By convention, realm names are in uppercase letters to distinguish them visually from domain names. You must type the correct case for the realm name if your site does not follow the uppercase convention.

To create comments in the `krb.realms` file, use the number sign (#). Any characters after a number sign are ignored to the end of the line. Blank lines and any leading or trailing white space on a line are also ignored.

Example C-2 shows a sample `krb.realms` file.

### Example C-2: Sample `krb.realms` File

---

```
footwear.biz.com SERIOUS.BIZ.COM [1]
.admin.biz.com ADMIN.BIZ.COM [2]
*.biz.com BIZ.COM [3]
```

---

The entries in Example C-2 create the following associations:

- ❶ Associates the server `footwear.biz.com` with the `SERIOUS.BIZ.COM` realm.
- ❷ Associates all servers in the `admin.biz.com` domain with the `ADMIN.BIZ.COM` realm.

Notice the addition of a preceding period identifies the first field as a domain name rather than a server name. Typically, this line is not required because the realm name is the uppercase letter equivalent of the domain name. However, in this example, it is required to prevent mapping the servers in the `admin.biz.com` domain to the `BIZ.COM` realm by the third line.

- ❸ Associates all other servers in other domains with the root name `biz.com` with the `BIZ.COM` realm. For example, servers in `sales.biz.com` and `support.teams.biz.com` domains are mapped to the `BIZ.COM` realm.

See `krb.realms(4)` for more information.

### C.5.3 The `v5srvtab` File

The `/krb5/v5srvtab` service key table file is a binary file that contains the principal keys extracted from the principal database on the Kerberos server. A Kerberos client, such as Tru64 UNIX, uses the key information in the service key table to reauthenticate a principal. That is, after the Kerberos server initially authenticates a principal, a Kerberos client can reauthenticate a principal if its key is in the service key table.

See `v5srvtab(4)` for more information. See Section C.7.4 for information on managing the `/krb5/v5srvtab` service key table file.

### C.5.4 The `.k5login` File

The `/krb5/.k5login` file is a hidden text file that contains a list of principals authorized to access a specific user account. Unauthorized access to the `.k5login` file jeopardizes the integrity of the user's account.

If you configure the Kerberos secured daemons to use `.k5login` authorization, each user must have a private authorization list in a `.k5login` file stored in the user's home directory. The list determines which principals are allowed to access the user's account.

Only the user can own and have write permissions to the user's `.k5login` file. Otherwise, authentication fails and users receive a permission denied error message.

The `.k5login` file has one entry per line. An entry is a full principal name spelled exactly as it appears in the principal database. Do not include comments or trailing spaces at the end of a line.

Example C-3 shows a sample `.k5login` file configured for a user named Jack who normally logs in as `jack@COMPANY.COM`, but wants to also log in as the principal `jack_hill@COMPANY.COM`. In addition, he wants to grant the principal `jill@HR.COMPANY.COM` access to his account.

#### Example C-3: Sample `.k5login` File

---

```
jack@COMPANY.COM
jack_hill@COMPANY.COM
jill@HR.COMPANY.COM
```

---

To log in to the account, Jack and Jill must enter the following information:

```
rlogin hostname -l username
```

For example, Jack would enter the following command to log in to a server called `server1`:

```
rlogin server1 -l jack
```

### C.5.5 The `ldapcd.conf` File

The `/etc/ldapcd.conf` is a text file that contains SSO caching parameters. If users are waiting an unusually long time for the Tru64 UNIX system to respond, you can tune the Windows 2000 SSO software by using a text editor to change the values of the caching parameters in the `ldapcd.conf` text file located on the Tru64 UNIX system.

Example C-4 shows an `ldapcd.conf` file. Comment lines begin with a pound sign (`#`). Table C-2 describes caching parameters.

#### Example C-4: Sample `ldapcd.conf` File

---

```
configuration file for ldapcd
#
format of the file is <id>: <value>
values that contain spaces, or : or # must be quoted
if an id listed but no value specified the id will
use the default value
#
max entries in cache, and number of seconds before entries
expire in the cache
#
connections: 4
pw_cachesize: 500
```

#### Example C-4: Sample ldapcd.conf File (cont.)

---

```
pw_expirecache: 900
gr_cachesize: 100
gr_expirecache: 900

usesasl: 1

directory: server1
searchbase: "cn=users,DC=SSO,DC=CORP,DC=COM"
machine_acctname: emerald.ne.corp.com
machine_dn: "cn=emerald,cn=computers,DC=SSO,DC=CORP,DC=COM"
```

---

**Table C-2: Caching Parameters**

Parameter	Description
connections	<p>The number of open connections that the caching daemon can make to the Active Directory.</p> <p>Increasing the value of this entry opens more connections to the Active Director; however, this consumes more file descriptors and increases the load on the Active Directory.</p> <p>Typically, 4 connections are adequate for a workstation, and 15 connections are adequate for a server.</p> <p>Default: 4 connections</p>
pw_cachesize	<p>The maximum number of user entries to store in cache.</p> <p>Increase or decrease this value as the maximum number of users increases or decreases.</p> <p>Default: 500 entries</p>
pw_expirecache	<p>The maximum number of seconds to cache a user entry.</p> <p>Increasing this value increases Tru64 UNIX performance, because a user's entry is readily available in the cache.</p> <p>If you delete a recently used user account, its entry remains in the cache for the amount of time specified by this parameter.</p> <p>Default: 900 seconds</p>



**Table C-2: Caching Parameters (cont.)**

Parameter	Description
gr_cachesize	The maximum number of group IDs to cache. Increasing this value increases Tru64 UNIX performance, because group IDs are readily available in the cache. Default: 100 group IDs
gr_expirecache	The maximum number of seconds to cache group IDs. Default: 900 seconds

**Note**

If you change the value of a cache parameter in the `/etc/ldapcd.conf` file, you must enter the following command to restart the caching daemon:

```
/sbin/init.d/ldapcd restart
```

See `ldapcd.conf(4)` for more information.

## C.5.6 The `ldapusers.deny` File

The `/etc/ldapusers.deny` file is a text file that contains an entry for each Tru64 UNIX user who is authenticated only by Tru64 UNIX when SSO is installed.

You must enter only one user name per line, and the user name must exactly match a user name in the `/etc/passwd` file.

To create comments, use the number sign (`#`). Any characters after a number sign are ignored to the end of the line. Blank lines and any leading or trailing white space on a line are also ignored.

Example C-5 shows a sample `ldapusers.deny` file.

### Example C-5: Sample `/etc/ldapusers.deny` File

```
The following file lists account names that are not allowed to use
the Windows 2000 authentication information when it is enabled.
Account names must match exactly the user account name in the
/etc/passwd file.
#
Syntax: account_1
.
.
.
```

### Example C-5: Sample /etc/ldapusers.deny File (cont.)

---

```
account_n
root
nobody
nobodyV
daemon
bin
uucp
uucpa
auth
cron
lp
tcb
adm
ris
wnn
pop
imap
Peter
Dave
Evan
Martha
```

---

See `ldapusers.deny(4)` for more information.

## C.6 Creating Accounts and Groups

You must create user accounts, computer accounts, and groups on the Windows 2000 Server if they are to be authenticated by the Windows 2000 Server. User accounts and groups created on the Tru64 UNIX system are created in accordance with the Tru64 UNIX user account and group policy on that system and are not authenticated by a Windows 2000 Server.

This section describes how to:

- Create a user account
- Set a principal's password
- Create a computer account
- Create a group

### C.6.1 Creating a User Account

To create user accounts on the Windows 2000 Server, you can use:

- The Tru64 UNIX `creacct` command

- Microsoft Management Console (MMC) Interface

### C.6.1.1 Creating a User Account Using the Tru64 UNIX `creacct` Command

To create a user account using the `creacct` command, enter:

```
creacct -a username
```

You are prompted for the following information:

- The name and password for an administrative account on the Windows 2000 Server.
- Comments for the user account.
- A Tru64 UNIX directory to be used as the user's home directory.

---

#### Note

---

If you do not specify a valid home directory, the default value is set to the `root (/)` directory. The `.profile` file or the `.cshrc` file in the `root` directory sets the environment path to first check the `/sbin` directory when Tru64 UNIX commands are executed. As a result, the `ls -l` command will not work when the home directory is set to the `root` directory.

---

- A Tru64 UNIX shell for the user account.
- A Tru64 UNIX group ID (GID) for the user account.
- A Tru64 UNIX user ID (UID) for the user account.
- A password for the user account.

See `creacct(1)` for more information.

### C.6.1.2 Creating a User Account Using the MMC Interface

Follow these steps to use the MMC interface to create a user account:

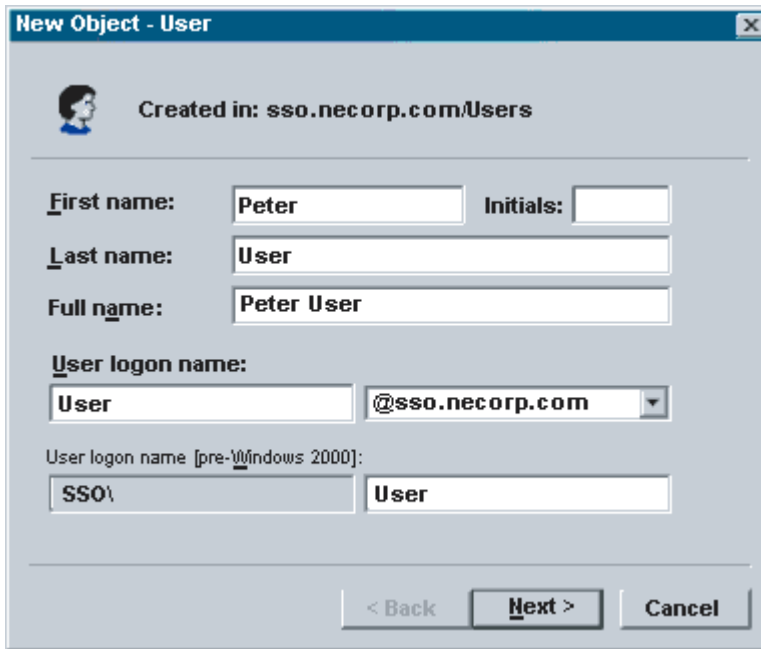
1. Display the Active Directory Users and Computers window:
  - a. Click on the Start button.
  - b. Choose Program, Administrative Tools, and then Active Directory Users and Computers.

The Active Directory Users and Computers window is displayed.

2. If necessary, highlight the Users folder. From the Action menu, choose New, and then User.

The New Object — User Window (Figure C–1) is displayed:

**Figure C-1: New Object — User Window: Required Information**



The screenshot shows a window titled "New Object - User" with a close button in the top right corner. Below the title bar, there is a small user icon and the text "Created in: sso.necorp.com/Users". The main area contains several input fields:

- First name:** A text box containing "Peter".
- Initials:** An empty text box.
- Last name:** A text box containing "User".
- Full name:** A text box containing "Peter User".
- User logon name:** A text box containing "User" and a dropdown menu showing "@sso.necorp.com".
- User logon name [pre-Windows 2000]:** A text box containing "SSO\" and another text box containing "User".

At the bottom of the window, there are three buttons: "< Back", "Next >", and "Cancel".

ZK-1675U-AI

3. Enter the required information in the New Object — User window, then click on the Next button. The First name and Initial fields are optional. A second New Object — User window (Figure C-2) is displayed:

Figure C-2: New Object — User Window: Password Information

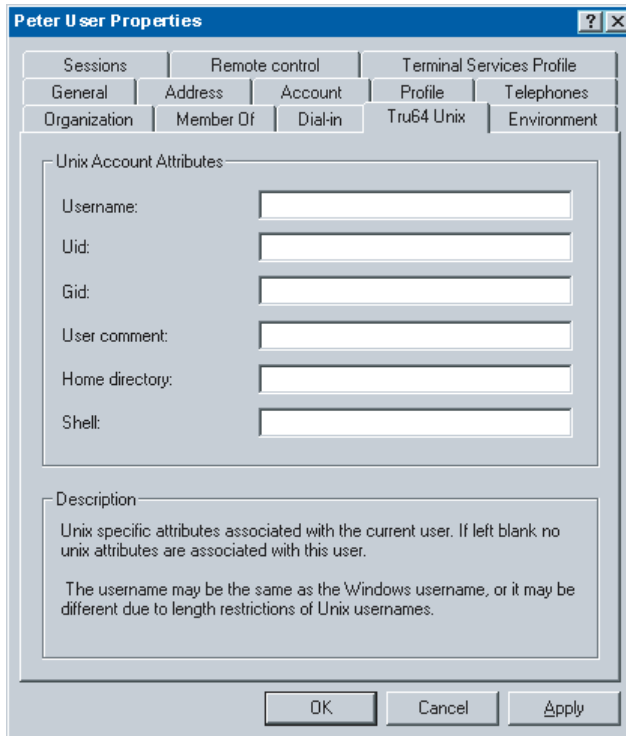


The screenshot shows a dialog box titled "New Object - User". At the top, there is a small user icon and the text "Created in: sso.compaq.com/Users". Below this, there are two text input fields: "Password:" and "Confirm password:", both containing six asterisks. Underneath the input fields are four checkboxes with the following labels: "User must change password at next logon" (checked), "User cannot change password", "Password never expires", and "Account is disabled". At the bottom of the dialog box, there are three buttons: "< Back", "Next >", and "Cancel".

ZK-1676U-AI

4. Enter a password and select password options, then click on the Next button.  
A confirmation window is displayed.
5. Click on the Continue button to create the user account, or click on the Cancel button to quit without creating the user account.  
If you choose to continue, the Active Directory Users and Computers window is displayed.
6. Double click on the name of the account that you just created.  
A properties dialog box for the user is displayed.
7. Click on the Tru64 UNIX tab.  
The Tru64 UNIX user properties dialog box (Figure C-3) is displayed.

**Figure C–3: Tru64 UNIX User Properties Dialog Box**



ZK-1677U-AI

8. Enter Tru64 UNIX user account information. Tru64 UNIX user account restrictions apply; for example, the length of the Tru64 UNIX user account name cannot exceed the maximum length determined by the Tru64 UNIX operating system software. See *System Administration* for more information on Tru64 UNIX user account restrictions.

---

**Note**

---

If you do not specify a valid home directory, the default value is set to the root (/) directory. The `.profile` file or the `.cshrc` file in the root directory sets the environment path to first check the `/sbin` directory when Tru64 UNIX commands are executed. As a result, the `ls -l` command will not work as expected when the home directory is set to the root directory.

---

When you move your mouse over an attribute, a description of that attribute is displayed in the Description section of the window.

9. Enter Tru64 UNIX user account attribute information and click on the OK button.

The Active Directory User and Group window is displayed.

## C.6.2 Setting a Principal's Password

To set a principal's password, you can use the Microsoft Management Console (MMC) interface or the Tru64 UNIX `creacct` command.

To use the `creacct` command to set a password, enter:

```
creacct -s principal
```

You are prompted for the name and password for an administrative account on the Windows 2000 Server, then for the principal's password.

See `creacct(1)` for more information about the `creacct` command.

See your Windows 2000 documentation for information on using the MMC interface to set passwords.

## C.6.3 Creating a Computer Account

Creating a computer account creates the account on the Windows 2000 Server and adds it to the key service table. All computer accounts are added to the current domain in the Active Directory under the Computers group.

Follow these steps to create a computer account:

1. Enter the `creacct` command specifying the fully qualified computer name:

```
creacct -h computer_name [-t keytable] [-u]
```

- If you do not specify a fully qualified name, the `creacct` command will construct one based on the local DNS name for the computer.
- If you do not specify the `-t` option, the computer account is created in the default service key table file (`/krb5/v5srvtab`).
- Specify the `-u` option so that the new computer account entry is added to the `/etc/ldapcd.conf` file.

You are prompted for the user name and password of a principal who has administrator privileges on the Windows 2000 Server.

2. Enter the user name and password.

The Active Directory is searched first for the specified computer. If an entry is found, you are prompted to replace or modify the existing entry.

3. Choose:

- Modify if you want to change the DNS computer name.

- Replace if you want to delete an existing computer account and replace it with a new one. When replacing an existing computer account, the `creacct` command searches the Active Directory for the host name to retrieve the DNS host name. It then prompts you to modify the DNS host name. You can reenter the existing DNS host name or enter a new one.

See `creacct(1)` for more information.

#### C.6.4 Creating a Group

You must use the MMC interface to create groups on the Windows 2000 Server. You can create a group of which Tru64 UNIX and Windows 2000 users are members.

Follow these steps to create a group:

1. Display the Active Users and Computers window:
  - a. Click on the Start button.
  - b. Choose Program, Administrative Tools, and then Active Directory Users and Computers.

The Active Directory Users and Computers window is displayed.

2. If necessary, highlight the Users folder. From the Action menu, choose New, and then Group.

The New Object — Group window is displayed.

3. Enter the name of the group and click on the OK button.

The Active Directory Users and Computers window is displayed.

4. Double click on the name of the group that you just created.

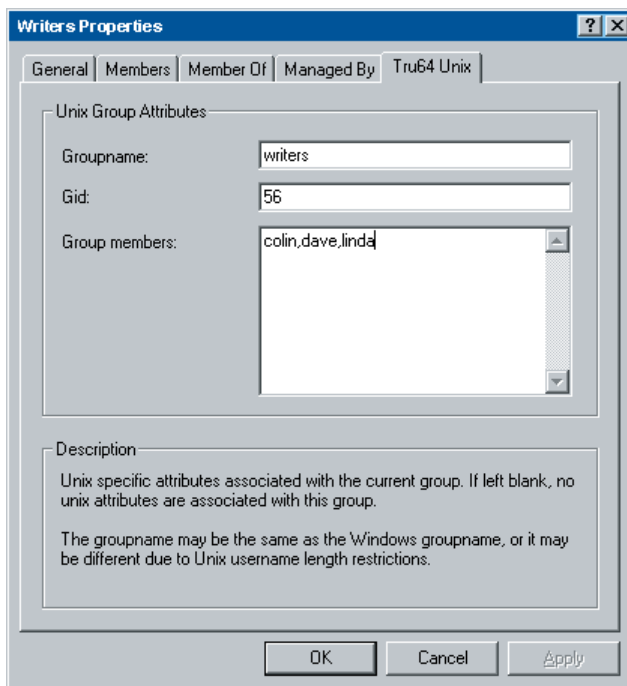
A properties dialog box for the group is displayed.

5. Click on the Tru64 UNIX tab.

The Tru64 UNIX properties dialog box for the group (Figure C–4) is displayed.



**Figure C-4: Group Properties Dialog Box**



ZK-1678U-AI

6. Enter Tru64 UNIX group information.

When you move your mouse over an attribute, a description of that attribute is displayed in the Description section of the window.

7. Enter Tru64 UNIX group attribute information and click on the OK button.

The Active Directory User and Group window is displayed.

## **C.7 Managing the SSO Software**

This section describes how to:

- Request tickets
- Display tickets
- Remove the credential cache
- Manage entries in the service key table

### **C.7.1 Requesting Tickets**

Use the `/sbin/kinit` command to request tickets.

- To request an initial ticket, enter:

```
kinit [-c cachename]
```

For example, to request a ticket from the default credential cache, enter:

```
kinit
```

- To request a ticket with a specific lifetime, enter:

```
kinit -l nwndnhmns
```

For example, to request a ticket with a lifetime of 45 hours and 30 minutes, enter:

```
kinit -l 45h30m
```

- To request a postdated ticket, enter:

```
kinit -d nwndnhmns principal domain
```

For example, to request a postdated ticket to start in one hour for a principal called mary/admin in a domain called COMPANY.COM, enter:

```
kinit -d 1h mary/admin@COMPANY.COM
```

The syntax of lifetime and postdate options is [*nw*][*nd*][*nh*][*nm*][*ns*], where w = weeks, d = days, h = hours, m = minutes, and s = seconds.

No spaces are allowed unless they are enclosed in quotation marks. For example, "1w 2d 3h 4m 5s". The default lifetime and postdate options are in hours unless otherwise specified.

If the requested postdate time period is less than the server's clock skew value (typically five minutes), the ticket's start time is set to the current time and it is issued as if the -d option had not been specified.

See `kinit(1)` for more information about requesting tickets.

## C.7.2 Displaying Tickets

Use the `/sbin/krlist` command to display the tickets stored in the credentials cache.

- To display all the tickets stored in a specific credential cache, enter:

```
klist -a [-c cachename]
```

For example, to display all the tickets in a credential cache called `/var/tmp/mycache`, enter:

```
klist -a -c /var/tmp/mycache
```

- To display all the tickets in the credential cache with their flags and addresses, enter:

```
klist -a -f [-c cachename]
```

See `klist(1)` for more information about displaying tickets.

### C.7.3 Removing the Credential Cache

To destroy all the tickets in a specific credential cache and remove the credential cache, enter:

```
kdestroy [-c cachename]
```

For example, to destroy all the tickets in a credential cache called `/var/tmp/mycache`, enter:

```
kdestroy -c /var/tmp/mycache
```

See `kdestroy(1)` for more information about removing tickets.

### C.7.4 Managing the Service Key Table

Use the `/sbin/ktutil` command to display and delete entries in the service key table file, destroy the service key table, and merge service key tables.

Use the `/usr/sbin/creacct` command to extract keys from the Windows 2000 Server and add them to the service key table. When using the `/usr/sbin/creacct` command, you are prompted for the user name and password of a principal who has administrative privileges on the Windows 2000 Server.

The default service key table file is `/krb5/v5srvtab`. This file is owned by the root user account.

- To display all the tickets in the service key table, enter:

```
ktutil [-t keytable]
```

For example, to display all the tickets in the default service key table, enter:

```
ktutil
```

- To extract a key for a Tru64 UNIX host from the Windows 2000 Server and add it to the service key table, enter:

```
creacct [-t keytable] -x host/fully_qualified_host_name
```

For example, to extract a key for a Tru64 UNIX host called `server1.company.com` and place it in the default service key table, enter:

```
creacct -x host/server1.company.com
```

- To extract a key for a principal from the Windows 2000 Server and add it to the service key table, enter:

```
creacct [-t keytable] -x principal/fully_qualified_host_name
```

For example, to extract a key for a user account called `user1` on a system called `w2kserverhost.company.com` and add it to the default service key table, enter:

- ```
# creacct -x user1/w2kserverhost.company.com
```
- To delete a ticket from the service key table for a specific principal, enter:

```
# ktutil -t keytable -d principal
```

For example, to delete a ticket in the default service key table for a principal called mary/admin in a domain called COMPANY.COM, enter:

```
# ktutil -t WFILE:/krb5/v5srvtab -d mary/admin@COMPANY.COM
```
 - To destroy a service key table, enter:

```
# ktutil -D -t keytable
```

For example, to destroy a service key table called /krb5/mytable, enter:

```
# ktutil -D -t WFILE:/krb5/mytable
```
 - To merge service key tables, enter:

```
# ktutil -c from_keytable -t to_keytable
```

For example, to merge all the entries in a service key table called /krb5/srvtable with the entries in the default service key table /krb5/v5srvtab, enter:

```
# ktutil -c /krb5/srvtable -t WFILE:/krb5/v5srvtab
```

See `ktutil(1)` and `creacct(1)` for more information about managing the service key table.

C.8 Troubleshooting the SSO Software

Because of the many interlocking dependencies that the SSO software has across multiple operating systems, it is critical that the setup procedures are followed exactly as described in Section C.4. If they are not, then the SSO software will not work properly, as generally signified with an `invalid login response message`. The majority of SSO problems are setup problems and can be found quickly by validating the setup of the SSO software.

C.8.1 SSO Configuration Problems

If the administrative account is not accepted as being valid when running the `/usr/sbin/creacct` command or the `/usr/sbin/w2ksetup` script, verify that the administrative account is a member of the Administrators, DnsAdmins, Domain Admins, Domain Users, and Users groups on the Windows 2000 domain controller.

When the Active Directory is created on the Windows 2000 server, its name must be specified in lower case. All other references to this name, which is the Kerberos Realm name, must be entered in upper case on all platforms.

C.8.2 Problems Using the kinit Command or Obtaining an Initial Ticket on Tru64 UNIX

Follow these steps if you are having trouble when entering the `kinit` command or receiving an initial ticket when logging in to the Tru64 UNIX system:

1. Verify that the `/etc/sia/matrix.conf` file has an entry for LDAP and Kerberos by entering the following command to display the contents of the `matrix.conf` file and looking for LDAP and Kerberos related entries in the display:

```
# more /etc/sia/matrix.conf
```

If the `/etc/sia/matrix.conf` file does not include the entries, enter the following commands:

```
# siacfg -r LDAP > /dev/null
# siacfg -r Kerberos > /dev/null
# siacfg -a -g pg -P LDAP /usr/shlib/libisialdap.so > /dev/null
# siacfg -a -P -g sci Kerberos /usr/shlib/libcsfk5siad.so
> /dev/null
```

2. Verify that there is an entry in the `/etc/hosts` file for the KDC; for example:

```
12.345.67.890 w2khost.mycompany.com w2khost
```

This step generally solves the problem when the initial `/usr/sbin/w2ksetup` utility is run and the following error message is displayed after the administrator's name and password are entered:

```
ldap_gssapi_bind: Operations error
```

3. Verify that the `/krb5/krb.conf` and `/krb/krb.realms` files are set up correctly.

Example `krb.conf` file:

```
MYCOMPANY.COM
MYCOMPANY.COM w2khost.mycompany.com admin server
```

Example `krb.realms` file:

```
*.mycompany.com MYCOMPANY.COM
```

4. Verify that the time on the KDC and the client are the same. A few minutes difference is acceptable, but should not exceed more than five (5) minutes.
5. Verify that the `v5srvtab` file is working correctly by entering the following command to extract the key:

```
# kinit -k
```

If the command fails, rerun the `/usr/sbin/w2ksetup` script or delete the `/krb5/v5srvtab` file, then enter the following command:

```
# /usr/sbin/creacct -h mymachinename -u
```

C.8.3 Password Prompting on Tru64 UNIX

Follow these steps if you are able to get a ticket from the `kdc`, but are still prompted for a password during subsequent logins on the Tru64 UNIX system:

1. Verify that a service ticket can be acquired for the target host by entering the following command:

```
# kinit -S host/targetmachinename.mycompany.com
```

If this command fails, the host that you are trying to access is not part of the Kerberos realm. To correct this, enter the following command:

```
# /usr/sbin/creacct -h targetmachinename.mycompany.com -u
```

2. Verify that the first line of the `/krb5/krb.conf` files are the same on the current host and the target host.
3. Verify that there are no entries in the `/etc/ldapusers.deny` file or in the `.k5login` file in the users home directory that prevents login.
4. Verify that the `/etc/services` and `/etc/inetd.conf` files are set up correctly and contain the following entries:

- In the `/etc/services` file:

```
kerberos 88/udp
kerberos 88/tcp
kerberos_master 749/udp
kerberos_adm 749/udp
kerberos_adm 749/tcp
```

- In the `/etc/inetd.conf` file:

```
kshell stream tcp nowait root /usr/sbin/rshd rshd -K
klogin stream tcp nowait root /usr/sbin/rlogind rlogind -K
```

Note

- Add `-x` to the end of the two lines in the `inetd.conf` file if you want encrypted communications.
- If you changed the `inetd.conf` file, you must enter the following command to restart the network:

```
# rcinet restart
```

C.8.4 Problems with SSO in a TruCluster

Follow these steps if the SSO software is not working correctly on cluster members in a TruCluster:

1. Verify that the `/etc/ldapcd.conf` file contains the following entries defined for each cluster member:

```
directory:  
searchbase:  
machine_acctname:  
machine_dn:
```

If these entries do not display for a cluster member, rerun the `/usr/sbin/w2ksetup` script on that cluster member.

2. Enter the following command and verify that an entry for each cluster member is displayed:

```
# /usr/sbin/ktutil
```

If an entry for each cluster member is not displayed, enter the following command:

```
# /usr/sbin/creacct -h missingmembername -u
```


D

Lightweight Directory Access Protocol

The Lightweight Directory Access Protocol (LDAP) is an Internet standard distributed client/server directory service protocol that runs over TCP/IP. An LDAP server manages entries in a directory, and makes the information available to LDAP clients across the network. An LDAP server can be used as a central repository of user information to identify and authenticate users.

This appendix contains the following information:

- LDAP overview
- Installing the Tru64 UNIX LDAP client software
- Configuring the Tru64 UNIX LDAP client software
- Managing the LDAP client daemon
- Managing access control

D.1 LDAP Overview

An LDAP server is similar to Network Information Services (NIS), but has the following advantages:

- An LDAP directory is highly scalable.
- LDAP directories are dynamically updated, saving administrators time because it is not necessary to rebuild maps and push them onto the network. Also, changes are available virtually immediately.
- An LDAP directory database can be used to centralize management of user related information.
- The ability to modify an attribute can be controlled at the attribute level. Users can be allowed to modify noncritical information (such as their preferred login shell or mail forwarding address) on their own. Modifications to more sensitive information (such as UID, GID, or a user's home directory) can be restricted to authorized directory managers only.
- You can set up multiple LDAP servers to make the data in the directory highly available. Through a process called replication, you can ensure that all LDAP servers have identical copies of the directory. The LDAP servers bind to one another and through standard LDAP commands, propagate changes to the directory.

The Tru64 UNIX system can be configured as an LDAP client and server. This section describes how to configure Tru64 UNIX as an LDAP client and requires that an LDAP server already be configured. See your LDAP server documentation for LDAP server configuration information.

When a user enters their user name and password to log in to Tru64 UNIX system that is configured as an LDAP client, the LDAP client sends the user information to the LDAP server for authentication. The LDAP server checks the user information with the entries in the LDAP directory. If there is a matching entry and the password is correct, the authentication succeeds and the user can login. If there is no matching entry or if the password is incorrect, the authentication fails and the user cannot login. The LDAP server returns the authentication results to the LDAP client.

D.2 Installing the Tru64 UNIX LDAP Client Software

To install the LDAP client software, you must install the optional LDAP Authentication (Network-Server/Communications) subset (`OSFLDPAUTH nnn`). This subset is located on the CD-ROM containing the Tru64 UNIX base operating system software.

The nnn represents the Tru64 UNIX version number. See *Release Notes* for the current version number. See *Installation Guide* for information on installing subsets.

D.3 Configuring the Tru64 UNIX LDAP Client Software

You must configure the LDAP client software. Configuring the LDAP client software requires:

- Updating the `/etc/ldapcd.conf` LDAP client configuration file to provide information about the LDAP server that it will use for authentication. A default `/etc/ldapcd.conf` file is provided when you install the LDAP client software.
- Setting the LDAP client runtime configuration variable.

D.3.1 Updating the `ldapcd.conf` File

In the `/etc/ldapcd.conf` file there are several attributes for which you can use the default value or provide a value; however, you must provide a value for the following attributes:

| | |
|-------------------------|---|
| <code>directory</code> | Host name of the LDAP directory server to be used for user authentication. |
| <code>searchbase</code> | The root of the branch in the directory server's database where user information is stored. |

| | |
|---------------------------------|--|
| port | The default directory server port; this must match the port you are using for the directory server. |
| machine_dn and machine_password | The machine_dn (distinguished name) and machine_password are what the ldapcd caching daemon uses to bind to the directory server to do searches and retrievals of information from the directory. These values are set when you initially configure the directory server during installation. Typically, you use the root distinguished name and password as specified in the directory server's configuration file (sladpd.conf). |

You update the `/etc/ldapcd.conf` file in the following ways:

- Use the SysMan Menu options. Expand the menu and select General Tasks - Setup LDAP Configuration. When you select this option, a window titled LDAP Configuration is displayed, containing a list of the LDAP configuration attributes. When you select an attribute from the list, a dialog box is displayed showing the current attribute value and providing an area for you to enter a new attribute value. Select OK to update the attribute values and exit the LDAP Configuration window and return to the SysMan Menu.
- Use a text editor to modify the `/etc/ldapcd.conf` file.

A sample `ldapcd.conf` configuration file is shown in Example D-1. HP recommends that you use the SysMan Menu options to modify the `/etc/ldapcd.conf` file. If you modify the value of an attribute in the `ldapcd.conf` file, you must restart the LDAP client daemon. See Section D.4 for information on restarting the LDAP client daemon.

Example D-1: Sample `ldapcd.conf` File

```
#
# directory server and port, active ldap connections cached
# by the daemon, max worker threads started
#
directory:      host.xyz.com 1
searchbase:    "o=XYZCompany" 2
port:          389 3
connections:   6 4
max_threads:   64 5

#
# max entries in cache, and number of seconds before entries
# expire in the cache
#
```

Example D-1: Sample `ldapcd.conf` File (cont.)

```
pw_cachesize: 20006
pw_expirecache: 120
gr_cachesize: 100
gr_expirecache: 600
:
:
machine_dn: "cn=Directory Manager"7
machine_pass: "password"

#
:

# the objectClass name of a password entry
pw_ogclass: posixAccount8

# name mappings for password attribute fields
pw_username: uid9
pw_password: userPassword10
pw_uid: uidNumber
pw_gid: gidNumber
pw_quota:
pw_comment: description
pw_gecos: gecos
pw_homedir: homedirectory
pw_shell: loginshell

# the objectClass name of a group entry
gr_ogclass: posixGroup11

# name mappings for group attribute fields
gr_ogclass: unixGroup12
gr_name: cn
gr_password: userPassword
gr_gid: gidNumber
gr_members: MemberUID
```

- ¹ Host name of the LDAP directory server to be used for user authentication.
- ² The root of the branch in the directory server's database where user information is stored.
- ³ The default directory server port; this must match the port you are using for the directory server.

- 4 Maximum number of open connections to the directory server maintained by the `ldapcd` caching daemon.
- 5 Maximum number of threads maintained by the `ldapcd` caching daemon. Each thread handles one connection to a local program. Allowing a higher number of threads may enable better response from the LDAP caching daemon, but requires more memory. If you are running a service that requires a large number of connections (for example, a mail service), set the maximum number of threads to 64 or greater (if your system has sufficient memory).
- 6 The value of `pw_cachesize` determines how many individual `passwd` entries are allowed to be cached. The value of `pw_expirecache` determines the maximum length of time that the `ldapcd` caching daemon will check the cache for an individual `passwd` entry. When the value of `pw_expirecache` is exceeded, the `ldapcd` daemon returns to the server to look for the requested `passwd` entry.

The values for `gr_cachesize` and `gr_expirecache` work similarly to `pw_cachesize` and `pw_expirecache`, but they work for group entries.
- 7 The value of `machine_dn` is the distinguished name by which the `ldapcd` caching daemon binds to the directory to do searches and retrievals of information from the directory. By requiring each system to use a particular DN, you can determine which machines are accessing the directory and for what purpose. Further, you can also control read and search access to the directory on a machine-account basis.
- 8 The name for the object class that defines the attributes for a UNIX account in the extended schema on your server.
- 9 LDAP attribute names (on the right) are mapped to fields (on the left) in the `passwd` structure returned by a call to `getpwent`.
- 10 Only the encrypted password is stored in the `userPassword` attribute.
- 11 The name for the object class that defines the attributes for a UNIX group in the extended schema defined on your server.
- 12 LDAP attribute names (on the right) are mapped to fields (on the left) in the `group` structure returned by a call to `getgrent(3)`.

D.3.2 Setting the LDAP Runtime Configuration Variable

After you initially update the `/etc/ldapcd.conf` file and before you start the LDAP client daemon, you must enter the following command to set the LDAP runtime configuration variable:

```
# /usr/sbin/rcmgr set LDAPCD_CONF yes
```

You only need to enter this command once. Changes to the `/etc/ldapcd.conf` file do not require that you reenter this command.

D.4 Managing the LDAP Client Daemon

Enter the following command to start the LDAP client daemon:

```
# /sbin/init.d/ldapcd start
```

The LDAP client daemon does the following when you start it for the first time:

- Updates the `/etc/sia/matrix.conf` file to include the LDAP Security Integration Architecture (SIA) mechanism.
- Adds the following entry to the `/etc/inittab` file to automatically start the LDAP client daemon when the system starts.

```
ldapcd:34:respawn:/usr/sbin/ldapcd -D > /dev/console 2>&1
```

Enter the following command to stop the LDAP client daemon:

```
# /sbin/init.d/ldapcd stop
```

Enter the following command to restart the LDAP client daemon:

```
# /sbin/init.d/ldapcd restart
```

D.5 Managing Access Control

By default, users defined in the LDAP database can log into every system that uses that database in conjunction with the LDAP Authentication. If you want to limit user access to specific systems, use the access control files `/etc/ldapusers.deny` and `/etc/ldapusers.allow`.

D.5.1 The `ldapusers.deny` File

The `/etc/ldapusers.deny` is a text file in which you enter the name of a Tru64 UNIX user who will not be authenticated by LDAP authentication. Users listed in the `ldapusers.deny` file are authenticated by the Tru64 UNIX security mechanisms configured on the system and are exempt from LDAP authentication.

A default `/etc/ldapusers.deny` file is provided when you install the LDAP client software. You must enter only one user name per line and the user name must exactly match a user name in the `/etc/passwd` file. To create comments, use the number sign (`#`). Any characters after a number sign are ignored to the end of the line. Blank lines and any leading or trailing white space on a line are also ignored.

Example D-2 shows the default `/etc/ldapusers.deny` file.

Example D–2: Default `ldapusers.deny` File

```
# ldapusers.deny - list of users who area not allowed to
# authenticate on this system via LDAP authentication
# (libsialdap.so & ldapcd)
#
# Account names must match exactly the user account name in the
# /etc/passwd file.
#
# Syntax: account_1
#         .
#         .
#         .
#         account_n
#
root
nobody
nobodyV
daemon
bin
uucp
uucpa
auth
cron
lp
tcb
adm
ris
wnn
pop
imap
ftp
anonymous
```

D.5.2 The `ldapusers.allow` File

If you want to disallow access to all but a few users, you must create the `/etc/ldapusers.allow` file. The `/etc/ldapusers.allow` is a text file in which you enter the name of a Tru64 UNIX user who will only be authenticated by LDAP authentication. If the `/etc/ldapusers.allow` file exists on a system, only users listed in that file are allowed to log in using LDAP authentication. Note that this is true even if `/etc/ldapusers.allow` is empty — its very existence invokes the stricter access control rules.

You must enter only one user name per line and the user name must exactly match a user name in the `/etc/passwd` file. To create comments, use the

number sign (#). Any characters after a number sign are ignored to the end of the line. Blank lines and any leading or trailing white space on a line are also ignored.

E

C2 Level Security Configuration

This appendix provides information on how to use enhanced security to configure your system to meet or exceed a C2 level of security as described in the *Trusted Computer System Evaluation Criteria* (also called the *Orange Book*). An on-line version of the *Orange Book* is available at <http://nsi.org/Library/Compsec/orangebo.txt>.

The system is also designed to meet the F-C2 functional class, as defined in the *Information Technology Security Evaluation Criteria* (ITSEC).

When a system is used in accordance with a site security policy, a C2 network, and the appropriate physical security, a C2 level environment can be achieved; expanding the protection of user and system information while maintaining full compatibility with existing Tru64 UNIX security mechanisms

Contact your sales representative for the latest evaluation and certification status of the Tru64 UNIX product.

This appendix contains the following information:

- Establishing a site security policy
- Minimum C2 configuration
- Initial configuration
- Physical security
- Applications
- Periodic security administration procedures
- Reference documents and verification tools

E.1 Establishing a Security Policy

A security policy is a statement of the rules and practices that regulate how an organization maintains its computing environment and how the organization manages, protects, and distributes sensitive information. A security policy should include:

- Documenting the process for maintaining and changing the security policy.

- Establishing the action taken by system administrators in the case of a break-in or other breach of security.
- Determining your site's audit policy including the following:
 - User activities you want to audit.
 - Locally defined audit events.
 - Location for audit logs.
 - Procedures for the review of audit logs.
 - Whether the auditing of login attempts to unrecognized account names (`login_uname`) is needed. Using this attribute can put passwords, entered out of sequence, with respect to the prompts, in the audit logs.
- Establishing a service access policy (supervision, passwords).
- Determining the `umask` for your system (022 is the system default).
- Establishing a schedule for verifying the integrity, including passwords, of your system and site.
- Defining the boundaries of the system and the interfaces (`telnet`, `ftp`, for example) between the boundaries.
- Establishing a magnetic media policy, especially for removable media.
- Determining which application software will be installed on your system and what its security implications are.
- Determining the password policy for your users. Some considerations follow:
 - Long passwords are hard to break, but inevitably they are written down.
 - If user-chosen passwords are used, only one person knows the password.
 - Machine-generated passwords are harder to break, but also harder to remember and will probably be written down.
- Determining the login controls for your system.
- Establishing a procedure for system startups, shutdowns and upgrades.
- Establishing a backup and recovery procedure.
- Determining who the system administrators (root access) are and exactly what their functions are to be.
- Establishing one or more secure account prototypes (Local Templates) for creating user accounts using the Account Manager program.

- Establishing a secure account template with startup files in the `/usr/skel` directory.
- Determining the access restrictions for each object on your system.
- Establishing the groups for your system.
- Determining the access restrictions for each user on your system.
- Recording for reference all the programs on your system that can set the UID or GID.
- Determining the export restrictions for file systems on your system.
- Establishing change control procedures for subjects and objects on your system.
- Establishing a procedure for physical access changes.
- Establishing the physical access requirements for your system and console.
- Establishing the physical access requirements for your network components.
- Establishing your network security policy.
- Determining which remote access programs (`ftp`, `telnet`, and such) will run on your system.
- Determining the remote systems that will have access to your system.
- Determining the console password requirements for your system and site.
- Establishing a modem policy. (Consider authentication, the configuration for dial-in and dial-out access.)
- Creating a “User Security Training” course or document for your site.
- Documenting how users will access the system: operating system, database, or application menu.
- Determining the secure devices for your site.

After your system is configured, the configuration files should change little and always in predictable ways. During periodic security reviews of your system, compare the base configuration files for content and permissions to the current files. Document the base system and network configuration by obtaining a listing of the following files and attaching them to the security policy:

```

/usr/skel/.profile
/usr/skel/.cshrc
/usr/skel/.login
/var/yp/<domain>/auto.master
/var/yp/<domain>/auto.home
/var/yp/<domain>/auto.###

```

```

/etc/auto.*
/etc/auth/*
/etc/dumpdates
/etc/ethers
/etc/exports
/etc/fstab
/etc/ftpusers
/etc/group
/etc/hosts
/etc/hosts.equiv
/etc/inetd.conf
/etc/motd
/etc/netgroups
/etc/passwd
/etc/profile
/etc/csh.login
/etc/logout          if used
/etc/remote
/etc/resolv.conf
/etc/rc.config
/etc/rc.site         optional, used with /etc/rc.config
/etc/screend.config
/etc/services
/etc/sec/site_events
/etc/sec/audit_events
/etc/sec/auditd_clients
/etc/sec/event_aliases
/etc/sec/auditd_cons
/etc/sec/auditd_loc
/etc/securettys
/etc/svc.conf
/tcb/*
/usr/adm/messages
/var/spool/uucp/Permissions      if UUCP is active
/var/spool/uucp/Systems          if UUCP is active
/var/spool/uucp/remote.unknown  if UUCP is active
/var/adm/cron/at.allow
/var/adm/cron/at.deny
/var/adm/cron/cron.allow
/var/adm/cron/cron.deny
/var/adm/crontab/               any files in these directories
/var/tcb/*
/var/yp/src/*

```

E.2 Minimum C2 Configuration

The *Orange Book's* requirements for a minimum C2 system is that the configuration for Tru64 UNIX is as follows:

- The requirement for a site security policy is met when you establish a security policy for your site as described in the *Site Security Handbook (RFC 1244)*. Your security policy should be in written form.
- Users should be able to change their own passwords and the passwords should be machine generated. (Recommended by the *Green Book*.) See Section E.3.2 for password configuration details.
- The requirement for users to be notified of their last login is met when enhanced security is configured.
- The discretionary access control requirement is met by configuring ACLs (access control lists) on your system. See Section 2.3 for more information on configuring ACLs. Do not use the `/usr/groups` approach for small systems (less than 32 users).
- The object reuse requirement mandates that workstations be configured with no `xhost` entries.
- Shared memory separation must be enabled. You do this by answering yes when `seccfg` asks if you want to disable segment sharing.
- The audit subsystem needs to be configured and available for use. See Chapter 3 for information on the audit subsystem.
- The ability to verify the integrity of the trusted computing base (TCB) is met by running the `fverify` and `authck` commands periodically as determined by your site's security policy.

E.3 Initial Configuration

After you have installed the Tru64 UNIX software subsets (including the optional enhanced security and documentation extension subsets) onto your system, you will start the software configuration. During the configuration, several of the selections you make will affect the security of your system. The assumption is that you need the maximum practical security configuration for your system. The following sections document the areas of concern for security and the recommended configuration.

E.3.1 General Configuration

General system configurations include:

- Ensure that the `/tmp`, `/var/tmp`, and `/var/spool` directories are on a file system other than that of the root (`/`) and `/usr` directories.
- Do not run Netscape Navigator with JAVA enabled. Only enable JAVA when you are connected to known secure sites.
- Avoid connecting systems to the Internet whenever possible.

E.3.2 Enhanced Passwords and Authentication Using `seccnfig`

Select the enhanced password attributes to match your site's security policy. See Section A.2.2 for details.

Use the following password attributes (defaults are defined in the `/etc/auth/system/default` file):

- Select either user-chosen or machine-generated passwords and configure as follows:
 - For user-chosen passwords (`u_pickpw` field in the `/etc/auth/system/default` file), set the minimum length to 8 characters (`u_minlen#8`) and the maximum length to 80 characters (`u_maxlen#80`).
 - For machine-generated passwords (no `u_pickpw` field in the `/etc/auth/system/default` file), set the minimum length to 0 characters (`u_minlen#0`) and the maximum length to 10 characters (`u_maxlen#10`). The value of 0 for minimum length causes Tru64 UNIX to use the *Green Book* algorithm to generate passwords.
- Ensure that null passwords cannot be used (`u_nullpw@`)
- Set the password expiration time to 180 days (`u_exp#15724800`)
- Set the account lifetime set to 360 days (`u_life#31449600`)
- Set the depth of the password history file to 9 (`u_pwdepth#9`)
- Set the number of tries to enter a password before locking the account to 5 (`u_maxtries#5`)
- Set new accounts to be locked (`u_lock`)
- Set the maximum number of login attempts before the terminal is locked to 10 (`t_maxtries#10`)
- Set the delay between attempted logins to 2 seconds (`t_logdelay#2`)
- Select triviality checks (`u_restrict`) and site password restrictions (`u_policy`)

Use the Account Manager (`dxaccounts`) or the `edauth` program to change the default settings.

E.3.3 Libraries

The libraries on your system can be used in an attack. Secure the libraries as follows:

- Disable segment sharing by answering `yes` when prompted by `seccnfig`.

- Verify that the permissions are correct (no write access except for the owner) and that the ownership is root on shared libraries (`/usr/shlib/*.so`), including any linked target files. Use the `ls -lL` command for this procedure.

E.3.4 Account Prototypes and Templates

The account templates used to create user account startup files are `/usr/skel/.login`, `/usr/skel/.cshrc` and `/usr/skel/.profile`.

Account prototypes (referred to as Local Templates) are provided by the Account Manager (`dxaccounts`). The prototypes let you set attributes like password expiration and login attempts for individual user accounts. If an attribute value is not specified in the local template, the value from the default file is used. The system-wide default attribute values are stored in the `/etc/auth/system/default` file. System default values are set with the `/usr/tcb/bin/edauth` command.

Configure user accounts as follows:

- Using the provided default templates, create account templates that reflect your site's security policy.
- Set the `umask` in the `/usr/skel/.login` file. (HP recommends a value of `027`.)
- Designate a restricted shell (`Rsh`) for users where appropriate.
- Verify that each user has a valid entry path (login shell) on the system. Users can be placed directly into an application by executing the application from the user's `/home/.profile` or from the entry in the `/etc/passwd` file or as a start point for the user with the execution of a startup program.
- If user access is restricted through menu scripts called from the user's `.profile` file, the scripts should have a `trap` command at the beginning of the file to ensure that `Ctrl/C` and other keyboard interrupts are ignored.

E.3.5 Configuring the Audit Subsystem

Before the audit subsystem kernel option can be configured, it needs to be included for the kernel build. Use the `sysman auditconfig` utility to configure the audit subsystem any time after the kernel build. Configure and run `audit` as follows:

- Use the default location for audit logs (`/var/audit/auditlog.nnn`). For overflow protection, put the audit logs on a file system other than `root (/)` and `/usr`.

- Establish an alternate location for audit logs to provide for an overflow of audit log data by editing the `/etc/sec/auditd_loc` file.
- Send `auditd` messages to the console (`/dev/console`).
- Set the audit mask to `audit trusted_events` and to log the name of a user (as described in your site policy) who attempts to log in to an invalid account.

If you are starting the audit daemon from the command line, use the following command:

```
# /sbin/init.d/audit start
```

See Chapter 3 for more information about the audit subsystem.

E.3.6 Verifying That Your Installation Is Secure

After you have rebooted the system to enable the enhanced security options, run the `fverify` and `authck` programs to verify the integrity of your system.

E.3.7 Configuring Network Security

Proper network configuration is a critical part of your secure computing environment. Use the following checklist as an aid to network configuration:

- Do not use NIS (Network Information Services, formerly called Yellow Pages) to distribute root account information. See Section A.6 for details.
- When using NIS, use the `/etc/yp/securenets` file, as described in `ypserv(8)`.
- Run `ypbind` with the `-S` flag and without the `-ypset` or `-ypsetme` options (default).
- If `uucp` is configured on your system, do the following:
 - Ensure that the `uucp` account is password controlled and that a separate `uucp` account is established for each machine that requires access.
 - Ensure that the `/var/spool/uucp/Permission` file has only valid entries.
 - Ensure that the `/var/spool/uucp/Systems` file has only valid entries.
- Ensure that the File Transfer Protocol (FTP) is secured and that, if possible, there are no anonymous FTP accounts. If you must use anonymous FTP, ensure the following:
 - The FTP account has an asterisk in the protected password field.
 - A `/usr/ftp` home directory is created for FTP. Create `/bin` and `/etc` subdirectories under the `/usr/ftp` directory.

- Nothing in the home directory is owned by ftp.
- A public subdirectory is created under the /usr/ftp directory for placement and retrieval of transferred files. User ftp should only have write access to the public subdirectory.
- Create an ~ftp/etc/passwd file with only the ftp account and no password.
- Copy the /etc/sia/bsd_matrix.conf file to ~ftp/etc/sia/matrix.conf.
- Copy /sbin/ls to ~ftp/bin/ls.
- The login shell for the ftp account should be /sbin/sink.
- Ensure that workstations are using DES-cookie based authentication (default). See the XDM-AUTHORIZATION-1 parts of dtlogin(1) for more information.
- When /usr/bin/X11/xhost is run, nothing should be reported. The output should look like the following:


```
# xhost
access control enabled, only authorized clients can connect
#
```

E.3.8 Postinstallation Security Configuration

After the system is installed and configured, perform the activities in the following sections.

E.3.8.1 umask for Remote Access

Add a umask entry as described in your site security policy to the /etc/csh.login, /etc/profile, and /etc/init.d/inet files. (Note that the /etc/init.d/inet file is overwritten during an update installation.)

E.3.8.2 Devices

Using /usr/tcb/bin/dxdevices, create the devices with the security attributes that reflect your site's security policy.

Ensure that terminal ports are readable only by the owner by modifying the remote login shell file as follows:

Add the following to the /etc/profile file:

```
case "$TERM" in
none) ;;
*) /usr/bin/setacl -b `/usr/bin/tty` ;;
esac
```

Add the following to the `/etc/csh.login` file:

```
if ($?TERM) then
  if ("$TERM" != "none") then
    /usr/bin/setacl -b '/usr/bin/tty'
  endif
endif
```

E.3.8.3 Accounts

Create and verify accounts as follows:

- Create the user accounts for your system using either the Account Manager (`/usr/bin/X11/dxaccounts`) or by restoring the `/usr/users` area and associated files from a previous system.
- Ensure that home directories are mounted with the `noexec`, `nosuid`, and `nodev` options.
- Ensure that CDE users have the autopause feature enabled by using a command similar to the following:

```
# grep extension.lockTimeout \  
~/dt/sessions/current/dt.resources
```

A 0 status indicates that the autopause feature is disabled.

- Review the `/etc/passwd` and `/var/tcb/files/auth.db` databases to verify that user home directories and passwords are appropriate.
- See *System Administration* for information on creating user accounts.

E.3.8.4 Root Access

Because root access must be carefully controlled and monitored, make sure the following conditions are met:

- That all passwords are changed after a system installation or after support vendors have had access to your machine.
- That the root password is changed before vendor access is granted to prevent exposure of your password generation methodology.
- That the single-user password feature is enabled. See `sulogin(8)`.
- That using the `su` command to become root is logged by audit.
- That access to the `setuid 0` or `setgid 0` programs on your system is restricted (700, 710, or 711)
- That the `/var/spool/cron/crontabs` files are accessible only by root or the owner.

- That root access is restricted to certain devices for login or that users must use the `su` command to access the root account. See `securettys(4)` for more information.
- The logins for the system-supplied UIDs are limited (setting the `u_lock` field) where appropriate. The following table provides the recommended restrictions:

| UID | Recommended login Status |
|--------|--------------------------|
| root | Restricted |
| daemon | Not allowed |
| bin | Not allowed |
| sys | Not allowed |
| uucp | Restricted |
| nobody | Not allowed |
| adm | Restricted |
| lp | Not allowed |

E.3.9 Network Configuration

Review the `/etc/svc.conf` file and ensure that a logical configuration has been set up for NIS. Also, if NIS is being used, verify that the client machines and the server have the correct domain name defined in the `NIS_DOMAIN` variable in the `/etc/rc.config` or `/etc/rc.site` file.

Ensure that the network files in the following table are protected:

| File | Comment |
|--|--|
| <code>/etc/exports</code> | Validate the entries. Avoid using the <code>-root=</code> option if possible. Use the <code>-access=<hostname></code> and <code>-ro</code> options on all specified file systems |
| <code>/etc/hosts</code> | |
| <code>/etc/services</code> | |
| <code>/etc/protocols</code> | |
| <code>/etc/inetd.conf</code> | |
| <code>/etc/hosts.equiv</code> | Validate that the entries are local hosts. |
| <code>/etc/ethers</code> | |
| <code>~username/.rhosts</code> and
<code>~username/.shosts</code> | Remove these files or run <code>rlogind</code> and <code>rshd</code> with the <code>-l</code> flag set. |

E.4 Physical Security

An important part of your site's security is the physical security of all the components in the environment. Check your physical security as follows:

- Verify that the system and its cabling are in a secure environment.
- Verify that all network components are physically secured. These include file servers, bridges, routers, hubs/concentrators, gateways, terminal servers, and modems.
- From the console prompt, ensure that the boot flags are set according to your site policy using the following command:

```
>>> show
```

- If your system supports the console password feature, ensure that it is being used. Consult your hardware documentation for information on console password support.
- Verify that a console terminal's function keys have not been programmed for login or password information.
- Ensure that modems have an automatic disconnect feature. Also make sure modems are in a secure environment.

E.5 Applications

To ensure the security of application software running on your system, make sure that the following conditions are met:

- Restrict any `setuid` or `setgid` programs.
- Ensure that any control files and executable files are writable only by the root account.
- If a firewall product is installed, see the firewall's documentation for the appropriate configuration information.
- If you are running the `screend` program, configure as described in `screend(8)`.
- If you have tunneling software installed, ensure that it is secure, as described in its documentation.

E.6 Periodic Security Administration Procedures

The frequency of the different classes of review activities is determined by your site's security policy. Perform the following activities on a regular schedule:

- Back up the system and its applications.
- Review the audit logs.

- Review the system accounting logs.
- Run the `fverify` and `authck` programs to verify the integrity of your system. Some public domain programs, such as `cops` and `tripwire`, are useful to help verify system integrity.
- Verify that your system has only necessary and authorized programs.
- Verify that compilers are available only on systems used for development.
- Verify that your system has only authorized, root-owned `setuid` and `setgid` programs using the following command:


```
# find / \( -perm -4000 -o -perm -2000 \) -ls
```
- Review the `/etc/exports` file to verify that all entries are valid.
- Check your user accounts as follows:
 - Review the `/etc/passwd` file to verify that all accounts are still valid.
 - Run the following command to ensure there are no enhanced profiles (`prpasswd` entries) without `/etc/passwd` entries:


```
# /usr/tcb/bin/convuser -dN
```
 - Verify that the home directory permissions are set according to your site policy.
 - Verify that all files in a user’s home directory are owned by that user.
 - Verify that each user has a valid entry path (login shell) on the system.
 - Verify that entries in a `.rhosts` or `.netrc` file are appropriate. Verify that that any `.exrc` and `.netrc` files are located only in user home directories.
 - Review the `hosts.equiv` file for valid entries.
 - Ensure that entries in the following files do not conflict with system parameters and that the files are protected by a permission of 755:


```
.profile
.login
.cshrc
.kshrc
.logout
```
 - Verify that user masks are set in accordance with your site policy using the following command:


```
# grep umask /usr/users/*/*.*
```

The system default mask is set to 022.
- Review the `/dev` directory and verify the following:

- That special devices have the proper permission.
- That access to devices such as `mem`, `kmem`, and `swap` are properly protected (440).
- That terminal ports are readable only by the owner.
- That users do not own any devices other than their terminal device and their printer.
- Verify that your modem authentication is functioning as intended.
- Use the following commands to verify that the same user name is not used for different UIDs, including between the local `/etc/passwd` file and NIS:

```
# ( ypcat passwd ; grep -v '^[+]' /etc/passwd ) | \
  sort -t: -k 1,1 -k 3,3n -u | \
  awk -F: '{if (n == $1) {print p; print}; \
                                         n=$1; p=$0}' | \
  more
```

- Use the following commands to verify that no user names use the same UID, including between the local `/etc/passwd` file and NIS:

```
# ( ypcat passwd ; grep -v '^[+]' /etc/passwd ) | \
  sort -t: -k 3,3n -k 1,1 -u | \
  awk -F: 'BEGIN {u=-1} {if (u == $3) \
                        {print p; print}; u=$3; p=$0}' | \
  more
```

- Use both of the following commands to verify that all accounts have local or NIS passwords:

```
# sort -t: -n /etc/passwd | awk -F: '$2 == "" print'
# /usr/tcb/bin/edauth -g | sed -f sed_file | egrep -v \
    ':u_pwd=[^:]|:u_istemplate:'
```

The commands in `sed_file` are as follows:

```
: top
/:\$ / {
N
b top
}
s/:\$ /:/g
s/:[<tab><space>]*:/:/g
s/:[<tab><space>]*:/:/g
```

- Use the following command to validate all the hidden files on your system:
- ```
find / \(-name '.*' ! -name . ! -name .. \) -print
```
- Use the following command to verify that no device files exist outside the `/dev` directory:

```
find / \(-type b -o -type c \) -print
```

- Ensure that entries in the following startup scripts are appropriate and that the files are properly protected:

```
/sbin/inittab
/etc/init.d
/sbin/rc?.d ? is the run level
```

- Ensure that the data saved in `/var/adm/crash` in the event of a system crash is accessible only to root and adm users.
- Using a password cracker program such as the public domain `crack` program, ensure that user passwords cannot be determined.
- Verify your site's physical security policy.
- Verify the permissions and ownership on the following directories are as listed:

Directory	Permission	Owner	Group
/	755	root	system
/bin	755	root	system
/dev	640	root or bin	system
/dev/null	666	root	system
/dev/ttys	666	root	system
/etc	755	root	system
/etc/rc.config	755	bin	bin
/etc/exports	644	root	system
/etc/passwd	644	root	system
/etc/resolv.conf	644	root	system
/etc/screend.config	755	root	system
/etc/sec	755	root	system
/home	555	root	system
/lib	755	root	system
/opt	755	root	system
/sbin	755	root	system
/sys	755	root	system
/tcb	755	root	system
/tmp	1777	root	system
/usr	755	root	system

Directory	Permission	Owner	Group
/usr/bin	755	root	system
/usr/lib	755	root	system
/usr/ucb	755	root	system
/usr/ucb	755	root	system
/var	755	root	system
/var/adm	755	root	system
/var/adm/crash	700	root	system
/var/adm/cron	755	root	system
/var/spool	755	root	system
/var/spool/cron	755	root	system
/var/spool/cron/atjobs	755	root	system
/var/spool/cron/crontabs	755	root	system
/var/spool/cron	755	root	system
/var/tcb	755	root	system
/var/tcb/audit	755	root	system
/var/tcb/bin	755	root	system
/var/tcb/files	755	root	system

## E.7 Reference Documents and Verification Tools

The following documents will help you create and maintain a secure computing environment:

- *Site Security Handbook (RFC 1244)*. This handbook is the product of the Site Security Policy Handbook Working Group, a combined effort of the Security Area and User Services Area of the Internet Engineering Task Force. An online copy of this document is available at <http://www.net.ohio-state.edu/hypertext/rfc1244/toc.html>
- *Tru64 UNIX Installation Guide*
- *Tru64 UNIX Installation Guide — Advanced Topics*
- *Trusted Computer System Evaluation Criteria*. U.S. Department of Defense, National Computer Security Center, DoD 5200.28-STD, December, 1985. This document, known as the *Orange Book* because of the color of its cover, is the U.S. Government's definitive guide to the development and evaluation of trusted computer systems. An online copy of the *Orange Book* is available at <http://nsi.org/Library/Compsec/orangebo.txt>



- *Password Management Guideline*. U.S. Department of Defense, (CSC-STD-002-85), April 12, 1985. This document, known as the *Green Book* because of the color of its cover, supports the *Orange Book* by presenting a set of recommended practices for the design, implementation, and use of password-based user authentication mechanisms. An online copy of the *Green Book* is available at <http://www.radium.ncsc.mil/tpep/library/rainbow/CSC-STD-002-85.html>

The following documents will help you understand security concepts and procedures:

- *Computer Security Basics* - O'Reilly and Associates, Inc.
- *Practical UNIX Security* - O'Reilly and Associates, Inc.
- *UNIX: Its Use, Control, and Audit* - Contact the Institute of Internal Auditors Research Foundation at 249 Maitland Avenue, Altamonte Springs, Florida 32701-4201.

The following tools can help you maintain a secure environment:

- `crack` — A public domain password-checking program available at .
- `tripwire` — An integrity-monitor for UNIX systems. The `tripwire` software uses several checksum and signature routines to detect changes to files and to monitor selected items of system-maintained information. The program also monitors for changes in permissions, links, and sizes of files and directories. .
- `COPS` — The Computer Oracle and Password System (`COPS`) package from Purdue University examines a system for a number of known weaknesses and alerts the system administrator to them; in some cases it can automatically correct these problems.
- `SATAN` — `SATAN` is a tool that helps system administrators recognize several common networking-related security problems. It reports the problems without actually exploiting them. For each type of problem found, `SATAN` offers a tutorial that explains the problem and what its impact could be and what can be done about the problem.

The following script is an example of a tool you can create to extract login and logout information from the audit logs:

```
#!/usr/bin/ksh -ph

Script to return summary of login/logout activities on the
system since the last time it was run.

export PATH=/usr/sbin:/usr/bin:/usr/ccs/bin:/sbin

where this script should run
```

```

Bdir=/var/adm/local
where to find audit log files
Adir=/var/audit

Ofile="{Bdir}/lasttime"
Nfile="{Bdir}/newtime"
Afile="{Bdir}/lastdata"
Tfile="{Bdir}/lastmsg"

Events="-e trusted_event"

umask 077

ensure the output format we need from date.
export LANG=C LC_ALL=C
export TZ=:UTC

if [! -f "${Ofile}"]
then
 print 700101000001 > "${Ofile}"
 touch -t 197001010000.01 "${Ofile}"
fi

date +%y%m%d%H%M%S > "${Nfile}"

curfile=$(auditd -q)
auditd -dx
sleep 20 # give time for compression of the old log
while [-f "$curfile" -a -f "$curfile".Z] || [-f "$curfile" \
 -a -f "$curfile".gz]
do
 sleep 2 # wait some more
done

: > "${Afile}"

for af in $(find "$Adir" -name "auditlog.*" -newer "${Ofile}" \
 -print | sort)
do
 audit_tool -b -t $(<"${Ofile}") -T $(<"${Nfile}") >> \
 "${Afile}" -o -Q $Events "${af}" 2>/dev/null

 # the suppressed errors are for the {un,}compressed messages
done

TZ=:localtime

if [-s "${Afile}"]
then
 audit_tool -B -Q "${Afile}" > "${Tfile}"

```

```
if [-s "${Tfile}"]
then
 Mail -s 'login/out audit summary' root < "${Tfile}"
fi
fi

mv -f "${Nfile}" "${Ofile}"
rm -f "${Afile}"
```

The following is the crontab entry for the above logging script:

```
0 9 * * * /var/adm/local/lreport
```



---

## Glossary

### **Access ACL**

The formal name of the ACL that is checked for access decisions on an object.

### **ACL (access control list)**

An optional extension of the traditional UNIX permission, which gives the user the ability to specify read, write, and execute permissions on a per user and per group basis.

See also *Access ACL*, *default ACLs*

### **administrator**

This document uses the term “administrator” in a generic sense to refer to any user involved in the security operation of the system.

### **audit event**

An event that is monitored and reported by the audit subsystem. Events include system events, application events, and site-definable events. An event can be any command, system call, routine, or program that runs on the system.

### **audit ID (AUID)**

An ID that is created at log in time and that is inherited across all processes.

### **auditing**

The recording, examining, and reviewing of security-related activities on a trusted system.

### **authenticate**

A process in which an entity presents credentials to a system to prove its identity.

### **authentication method**

A mechanism by which entities are authenticated. Examples include BSD, Enhanced Security, and Kerberos.

### **Base (BSD) security**

The traditional security that is delivered on BSD UNIX systems. Base security consists of user authentications based on user names and passwords located in the `/etc/passwd` file. Base security is the default Tru64 UNIX security.

**BSD (Berkeley Software Distribution)**

A UNIX software release of the Computer System Research Group of the University of California at Berkeley -- the basis for some features of Tru64 UNIX.

**decryption**

A process in which a cryptographic algorithm, called cipher, is used to transform cipher text (encrypted text) into plain text (decrypted text).

**default ACLs**

The ACLs associated with directories. These two types of ACLs (default access ACL and default directory ACL) determine what ACLs are given to files and subdirectories created in a directory.

**digital certificate**

An electronic document that can be used to authenticate the identity of an individual, a server, a company, or some other principal and bind that principal with a public key.

**digital signature**

An electronic signature that can be used to authenticate the identity of the sender of a message or the signer of a document.

**discretionary access control (DAC)**

Manages access to system resources including directories, files, devices, and processes. Tru64 UNIX implements DAC through the use of Tru64 UNIX permissions and Access Control Lists (ACLs).

**encryption**

A process in which a cryptographic algorithm, called cipher, is used to transform plain text (decrypted text) into cipher text (encrypted text).

**effective user ID (EUID)**

The current user ID, but not necessarily the user's ID. For example, a user logged in under a login ID may change to another user's ID. The ID to which the user changes becomes the effective user ID until the user switches back to the original login ID.

**enhanced passwords**

Passwords with the enhanced attributes made available by the enhanced security option. Enhanced passwords are stored in the `prpasswd` file and are sometimes referred to as extended, protected, or shadowed passwords.

**Enhanced Security**

The optional security feature that supplement base security. Enhanced security consists of enhanced password profiles.

See also *enhanced passwords*

**entity**

A user, program, or system that can be authenticated.

**ER (external representation)**

A POSIX-compliant ASCII representation of an ACL used for presentation to the user.

See also *IR (internal representation)*

**evaluation criteria**

The *Trusted Computer System Evaluation Criteria (TCSEC)*. The enhanced security features in the Tru64 UNIX system have been designed to meet this criteria.

**host-based authentication**

A noninteractive UNIX authentication method that is based on host name and user name identification (not passwords).

**IPsec**

A protocol that provides remote authentication services at the IP layer for data that is exchanged by using the TCP/IP transport protocol.

**IR (internal representation)**

A binary representation of an ACL used by the ACL library routines.

See also *ER (external representation)*

**Kerberos**

A secure method for authenticating a request for a service in a computer network.

**KDC (Key Distribution Center)**

Mediates communication between principals and creates a session key for their exclusive use when using secret keys authentication.

See also *ER (external representation)*

**lightweight directory access protocol (LDAP)**

an Internet standard distributed client/server directory service protocol that runs over TCP/IP.

**login spoofing program**

Any program that represents itself as a `login` program to steal a password. For example, a spoofing program might print the login banner on an unattended terminal and wait for input from the user.

**NIS (Network Information Service)**

A distributed client/server data lookup service for sharing information on a local area network (LAN).

**operator**

The person responsible for the day-to-day maintenance of a system, including backups, line printer maintenance, and other routine maintenance tasks.

See also *system administrator*

**principal**

A user, service, application, or host.

**process ID (PID)**

A unique number assigned to a process that is running.

**process**

A unit of control of the operating system. A process is always executing one program, which can change when the current program invokes the `exec()` system call. A process is considered trusted when its current program is trusted.

See also *program*

**program**

A set of algorithms designed, compiled, and installed in an executable file for eventual execution by a process. A program is considered trusted when it upholds the security policies of the system.

See also *process*

**public key authentication**

A type of cryptography in which two communicating principals, usually a client and server, use a public and private key to authenticate each other and the user and to encrypt and decrypt the data that they exchange.

**PPID (parent process ID)**

The process ID of the parent or spawning process.

**root**

The login name for the superuser (system administrator).

**root directory**

The name applied to the topmost directory in the UNIX system's tree-like file structure; hence, the beginning of an absolute pathname. The root directory is represented in pathnames by an initial slash (/); a reference to the root directory itself consists of a single slash.

**root file system**

The basic file system, onto which all other file systems can be mounted. The root file system contains the operating system files that get the rest of the system to run.



**secret key authentication**

A type of cryptography in which two communicating principals, usually a client or server, use the same secret key, called a session key, to authenticate a system and user and to encrypt and decrypt the data that they exchange.

**Secure Shell**

A client/server application that provides a suite of network commands that provides remote authentication services for data that is exchanged when using a Secure Shell command.

**security attributes**

The parameters used by the trusted computing base (TCB) to enforce security. Security attributes include the various user and group identities.

**SIA (Security Integration Architecture)**

The Security Integration Architecture manages the order in which security mechanisms are used.

See also *vouching*

**SSO (Single Sign On)**

Client/server software that uses Kerberos to provide remote authentication services for data that is exchanged when using the `ftp`, `rccp`, `rlogin`, `rsh`, and `telnet` network commands.

**site-defined events**

Audit events that are created by application software (that is, not the operating system).

**spoofing program**

See *login spoofing program*

**system administrator**

The system administrator is responsible for file system maintenance and repair, account creation, and other miscellaneous administrative duties.

**TCB (trusted computing base)**

The set of hardware, software, and firmware that together enforce the system's security policy. The Tru64 UNIX TCB includes the system hardware and firmware as delivered, the trusted Tru64 UNIX operating system, and the trusted commands and utilities that enforce the security policy. The operating system and other software distributed with the trusted Tru64 UNIX system satisfy security requirements.

**Traditional security**

See *Base (BSD) security*

**triviality checks**

Checks performed on passwords to prevent the use of easily guessed passwords. Triviality checks prevent the use of words found in the dictionary, user names, and variations of the user name as passwords.

**Trojan horse**

Any program that when invoked by a user steals the user's data, corrupts the user's files, or otherwise creates a mechanism whereby the Trojan horse planter can gain access to the user's account. Viruses and worms can be types of Trojan horses.

See also *virus, worm*

**virus**

A computer program designed to insinuate itself into other programs or files in a system and then to replicate itself through any available means (disk file, network, and so forth) into other similar computers, from which it can attack yet more systems. Viruses are designed with the object of damaging or destroying the "infected" programs or systems and are often programmed to become destructive at a specific time, such as the birthday of the virus's programmer.

See also *Trojan horse, worm*

**vouching**

A technique that allows a security mechanism to trust the authentication process of a previously run security mechanism. This feature is implemented by the Security Integration Architecture (SIA).

**worm**

A computer program designed to insinuate itself into other programs or files in a system and then to replicate itself through any available means (disk file, network, and so forth) into other similar computers, from which it can attack yet more systems. Worms are designed with no serious intent to do damage, but they are harmful because they occupy resources intended for legitimate use.

See also *Trojan horse, virus*

---

# Index

## A

**abbreviated audit reports**, 3–34

**absolute permissions**

removing, 2–6

setting, 2–6

**accounting tools**, 3–37

**accounts**

adding, A–1

**ACL**

archiving, 2–23

changing, 2–21

command interaction, 2–23

disabling, 2–13

enabling, 2–13

inheriting, 2–17

kernel status, 2–13

listing for files, 2–21

managing, 2–21

on NFS, 2–14

overview, 2–12

resource, 2–12

structure, 2–14

verifying status, 2–13

viewing, 2–21

**ACL entries**

adding, 2–22

modifying, 2–22

removing, 2–22

**aliases for audit events**, 3–10

**application-specific auditing**,  
3–10

**applications**

adding to the file control database,  
A–13

**audgen command**, 3–5

**audgen trusted event**, 3–9

**audit**, 3–1

( *See also* audit subsystem )

accessing the graphic interface, 3–6

accounting tools, 3–37

application-specific auditing, 3–10

audgen command, 3–5

audit hosts file, 3–19

audit hub, 3–19

Audit Manager, 3–6

audit mask, 3–6

audit\_tool command, 3–6

auditconfig command, 3–5

auditd command, 3–5

auditing remotely, 3–19

auditmask command, 3–5

AUID (audit ID), 3–11

CDE interface, 3–6

commands, 3–5

configuration, 3–15

console messages, 3–3

content of records, 3–11

crash recovery, 3–22

data recovery, 3–22

dependencies among audit events,  
3–35

disable events, 3–25

displaying audit records, 3–32

/etc/sec/auditd\_clients file, 3–19

events

preselection, 3–30

site-defined events, 3–10

state-dependent information,  
3–35

trusted events, 3–8

- files, 3-3
  - site\_events file, 3-10
- graphic interface, 3-6
- GUI, 3-6
- ID (AUID), 3-11
- implementation notes, 3-6
- log files, 3-3, 3-32
- logging tools, 3-37
- login audit mask
  - setting, 3-6
- LUID (login ID), 3-11
- managing data, 3-30
- managing growth of data, 3-30
- messages, 3-3
- network audit hosts file, 3-19
- networked auditing, 3-19
- overview, 3-1
- preselection, 3-30
- record content, 3-11
- reports, abbreviated, 3-34
- responding to audit reports, 3-42
- selecting events, 3-21, 3-23
- self-auditing commands, 3-8
- site-defined events, 3-10
- starting, 3-21
- stopping, 3-21
- suspending, 3-21
- sysman auditconfig command, 3-5
- trusted events, 3-8
- user audit mask
  - setting, 3-6
- audit events**
  - aliases for audit events, 3-10
  - displaying, 3-23
  - managing audit events, 3-30
  - site-defined, 3-10
  - trusted audit events, 3-8
- audit ID (AUID)**, 3-1, 3-11
- Audit Manager graphic interface**, 3-6
- audit records**
  - displaying, 3-32
- audit subsystem**
  - configuring, 3-16
  - audit\_daemon\_exit** trusted event, 3-8
  - audit\_log\_change** trusted event, 3-8
  - audit\_log\_create** trusted event, 3-8
  - audit\_log\_overwrite** trusted event, 3-8
  - audit\_reboot** trusted event, 3-9
  - audit\_start** trusted event, 3-9
  - audit\_stop** trusted event, 3-9
  - audit\_subsystem**
    - event aliases, 3-10
  - audit\_suspend** trusted event, 3-9
  - audit\_tool** command, 3-6
  - audit\_xmit\_fail** trusted event, 3-9
- auditable** events, 3-6
- auditconfig** command, 3-5
- auditconfig** trusted event, 3-9
- auditd** command, 3-5
- auditing**
  - in a TruCluster, 3-38
- auditing for applications**, 3-10
- auditmask** command, 3-5
- auth\_event** trusted event, 3-10
- authck** program, A-12
- authenticating users**
  - enhanced security, A-10
- authentication**
  - advanced server for UNIX, 1-14
  - certificates, 1-6
  - digital signatures, 1-6
  - host-based, 1-4
  - IPsec overview, 1-14
  - Kerberos overview, 1-13
  - overview, 1-1
  - passwords, 1-4
  - public key, 1-5
  - secret keys, 1-7
  - Secure Shell overview, 1-13
- authentication configuration**
  - maximum login attempts, A-5
  - password aging, A-4
  - password change time, A-5

password-changing controls, A-5  
**authentication profile**, A-7, A-28

## B

---

**backup procedures**, A-27  
**Berkeley database**, A-10  
**binary numbers**  
in permissions, 2-8t  
**boot loading software**, A-31

## C

---

**C2 features**  
password control, 1-11  
**changing**  
directory permissions, 2-5  
**chmod command**, 2-3  
**clusters**  
distributed logins, A-23  
NIS, A-23  
upgrades, A-22  
**commands**  
chmod, 2-3  
umask, 2-8  
**configuration**  
audit, 3-15  
password aging, A-4  
password change time, A-5  
password-changing controls, A-5  
**configuring**  
audit subsystem, 3-16  
SIA, 1-18  
**console file**, A-30  
**console messages**  
audit, 3-3  
**content of audit records**, 3-11  
**crash recovery**  
audit data, 3-22  
**creacct command**  
syntax, C-25  
using, C-25

## D

---

**data loss**, A-27  
**databases**, A-10  
enhanced password, A-28  
file control, A-13  
groups, A-30  
**default permissions**  
setting, 2-8  
**default user mask (umask)**, 2-10  
**defaults database**, A-7  
**dependencies among audit events**,  
3-35  
**/dev/console file**, A-30  
**/dev/pts/\* file**, A-30  
**/dev/tty\* file**, A-30  
**device assignment database**, A-11  
**devices**, A-25  
database, A-9  
databases, A-26  
installation, A-25  
**directories**  
changing permissions, 2-5  
**displaying**  
audit events, 3-23  
audit records, 3-32  
**distributed logins in a cluster**,  
A-23

## E

---

**encrypted password**, A-29  
**encryption**, 1-9  
**enhanced password database**,  
A-7, A-28  
**enhanced security**, 1-10  
aging configuration, A-4  
authenticating in a TruCluster,  
A-24  
authenticating users, A-10  
centralized account management,  
A-13

- configuring in a TruCluster, A-22
- controls configuration, A-5
- encryption, A-6
- failed login records, A-6
- features, 1-10
- installing, A-1
- installing in a TruCluster, A-22
- login features, 1-11
- login records configuration, A-6
- maximum login attempts, A-5
- minimum time configuration, A-5
- NIS, A-13
- NIS client setup, A-18
- NIS master server setup, A-16
- NIS migration, A-18
- NIS slave server setup, A-17
- profile migration, A-6
- removing NIS, A-18
- security, A-4
- successful login records, A-6
- templates for NIS user accounts, A-15
- templates for user accounts, A-11
- time between login attempts, A-5
- time between logins, A-6
- troubleshooting NIS, A-21
- vouching, A-6
- enhanced security databases**, A-7
- /etc/auth/system/default file**, A-29
- /etc/auth/system/devassign file**, A-30
- /etc/auth/system/ttydb file**, A-29
- /etc/group file**, A-30
- /etc/passwd file**, A-30
- /etc/sec/auditd\_clients file**, 3-19
- /etc/sec/event\_aliases file**, 3-10
- /etc/sec/site\_events file**, 3-10

## F

---

- file attributes**, A-31
- file control database**, A-13
  - description, A-9

- location, A-11
- files**
  - required, A-28
  - restricting access, 2-10
- fverify command**, A-31

## G

---

- graphic interface**
  - for audit subsystem, 3-6
- group database**, A-30
- groups**
  - database file, A-30

## I

---

- installing**
  - enhanced security, A-4

## K

---

- .k5login file**
  - entry format, C-13
  - sample, C-13
- Kerberos**
  - authentication, C-2
  - client, C-1
  - realms, C-2
  - server, C-1
- krb.conf file**
  - entry order, C-9
  - sample, C-9
- krb.realms file**
  - entry order, C-11
  - sample, C-11
  - wildcards, C-11

## L

---

- LDAP Module for System Authentication**
  - access control, D-6
- ldapcd.conf file**
  - parameters, C-13

- sample, C-13
- ldapusers.deny file**
  - sample, C-15
- lock file**, A-27
- log files**, 3-3, 3-38
- logging tools**, 3-37
- login**
  - audit mask, setting, 3-6
  - enhanced security, 1-11
  - login ID (LUID), 3-11
  - maximum tries configuration, A-5
  - trusted event, 3-10
- login script**
  - setting user mask, 2-11
- login timeouts**, A-27
- logout trusted event**, 3-10
- LUID (login ID)**, 3-11

## N

---

- network**
  - audit hub, 3-19
  - auditing across a network, 3-19
- NIS**
  - automated procedures, A-17
  - large databases, A-17
  - overrides, A-14

## O

---

- octal numbers**
  - in setting permissions, 2-7

## P

---

- passwd file**, A-30
- password**
  - BSD, 1-10
  - changing, 1-17
  - database, A-30
  - enhanced database, A-7
  - enhancements, 1-11

- LDAP, 1-12, D-1
  - maximum tries configuration, A-5
- NIS, 1-12
- pattern matching**
  - changing file permissions with, 2-6
- permissions**
  - binary numbers, 2-8t
  - combinations, 2-7t
  - setting file and directory, 2-3
  - setting with octal numbers, 2-7
  - specifying with umask, 2-10
- physical device**, A-9
- preselection of audit events**, 3-30
- process audit mask**, 3-6
- protecting removable media**, A-25
- pseudoterminal**, A-30
- pts/\* file**, A-30

## R

---

- rc[023] files**, A-30
- recovering**
  - audit data, 3-22
- remote auditing**, 3-19
- removing**
  - absolute permissions, 2-6
- required files**, A-28
- resource**
  - ACL, 2-12
- resources**
  - securing, 2-1
  - Tru64 UNIX permissions, 2-1
- responding to audit reports**, 3-42
- restricting file access**, 2-10
- root authentication profile**, A-28

## S

---

- /sbin/creacct command**
  - ( See creacct command )
- secconfig command**, A-2
- secure characteristics**, A-25

## **Secure Shell**

- client, B-1
- commands, 1-13t, B-1t, B-22
- configuring, B-3
- configuring commands to use, B-8
- configuring host-based user authentication, B-16
- configuring password user authentication, B-10
- configuring public key user authentication, B-10
- configuring user authentication, B-9
- creating public and private host keys, B-20
- forwarding ports, B-20
- forwarding TCP/IP port, B-20
- forwarding X11, B-21
- managing passphrases, B-15
- managing server, B-18
- overview, B-2
- resetting the daemon, B-19
- restarting the daemon, B-19
- restricting users, B-19
- scp2 command, B-22
- server, B-1
- sftp2 command, B-23
- ssh2 command, B-23
- starting the daemon, B-19
- stopping the daemon, B-19
- user authentication, B-9

### **Secure Shell client**

- configuring, B-6

### **Secure Shell server**

- configuring, B-3

### **securing**

- resources, 2-1

### **security policy, E-1**

### **segment sharing, A-3**

### **service key table, C-2**

- deleting a ticket, C-26
- destroying, C-26
- displaying, C-25

- extracting entries for a host, C-25
- extracting entries for a principal, C-25
- merging, C-26

### **setting**

- absolute permissions, 2-6
- file and directory permissions, 2-3
- permissions using octal numbers, 2-7

### **shared libraries, A-3**

### **SIA**

- adding SIA mechanism, 1-20
- changing password, 1-17
- configuring, 1-18
- logging, 1-21
- overview, 1-15
- removing SIA mechanism, 1-21
- SIA mechanism initialization, 1-20

### **Single Sign On**

- Active Directory considerations, C-3
- ASU considerations, C-3
- configuration files, C-8
- creating computer accounts, C-21
- creating groups, C-22
- creating user accounts, C-16
- DCE considerations, C-3
- displaying tickets, C-24
- /etc/ldapd.conf file, C-13
- /etc/ldapusers.deny file, C-15
- installation considerations, C-3
- installing, C-2
- installing on Tru64 UNIX, C-5
- installing on Windows 2000, C-3
- Kerberos, C-1
- /krb5/k5login file, C-12
- /krb5/krb.conf file, C-9
- /krb5/krb.realms file, C-10
- /krb5/v5srvtab file, C-12
- managing, C-23
- managing service key table, C-25
- removing the credential cache, C-25



- requesting tickets, C-23
- setting passwords, C-21
- SIA, C-8
- upgrading, C-2
- single-user mode**, A-29
- site-defined events**, 3-10
- state-dependent audit events**, 3-35
- sysman auditconfig command**, 3-5
- system audit mask**, 3-6
- system console**, A-29, A-30
- system defaults database**
  - description, A-7
  - updating, A-26
- system startup**, A-27

## T

---

- /tcb/files/auth/r/root file**, A-28
- terminal control database**, A-8, A-11, A-26
- ticket**
  - displaying with flags, C-24
  - requesting by lifetime, C-24
  - requesting by postdate, C-24
- traditional logging**, 3-38
- troubleshooting**, A-27
- Tru64 UNIX permissions**
  - displaying, 2-2
  - overview, 2-1

- trusted events**, 3-8
- tty\* file**, A-30

## U

---

- umask command**, 2-8
- update installation**, A-1
- user account**
  - creating with creacct command, C-17
  - creating with MMC Interface, C-17
- user audit mask**
  - setting, 3-6
- user mask**, 2-8
  - default, 2-10
  - login script, 2-11
  - permission combinations, 2-9t

## W

---

- wildcards**
  - changing file permissions with, 2-6
- workstation**
  - protecting removable media, A-25

## X

---

- X displays**, A-27
- X server**
  - restarting, C-7