

# Tru64 UNIX

---

## セキュリティ管理ガイド

Part Number: AA-RQ2YB-TE

**2002 年 11 月**

ソフトウェア・バージョン: Tru64 UNIX Version 5.1B 以上

本書では、Tru64 UNIX におけるセキュリティの概念と管理について説明します。

---

日本ヒューレット・パッカード株式会社

---

© 2002 日本ヒューレット・パッカード株式会社

CyberSafe は CyberSafe 社の登録商標です。ActiveTRUST および TrustBroker は CyberSafe 社の商標です。Microsoft® および Windows NT® は米国 Microsoft 社の米国ならびに他の国における登録商標です。The Open Group™ および UNIX® は、The Open Group の米国ならびに他の国における商標です。

SSH Secure Shell テクノロジーに含まれる SSH は、SSH Communications Security 社 (<http://www.ssh.com>) の登録商標です。

このドキュメントに記載されているその他の会社名および製品名は、各社の商標または登録商標です。

Confidential computer software. Valid license from Compaq 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.

Compaq shall not be liable for technical or editorial errors or omissions contained herein. The information in this document is provided "as is" without warranty of any kind and is subject to change without notice. The warranties for Compaq products are set forth in the express limited warranty statements accompanying such products. Nothing herein should be construed as constituting an additional warranty.

原典    Securiry Administration (AA-RH95E-TE)  
         Copyright ©2002 Hewlett-Packard Company

---

# 目次

## まえがき

## 1 識別，認証，および承認

1.1	認証の概要 .....	1-1
1.2	認証の実装 .....	1-4
1.2.1	パスワード認証 .....	1-4
1.2.2	ホスト・ベース認証 .....	1-5
1.2.3	公開鍵認証 .....	1-6
1.2.3.1	デジタル署名と証明書 .....	1-7
1.2.4	秘密鍵認証 .....	1-8
1.2.5	暗号化方式 .....	1-10
1.3	ローカル認証方式 .....	1-11
1.3.1	基本 (BSD) メカニズム .....	1-11
1.3.2	エンハンスド・セキュリティ・メカニズム .....	1-12
1.3.2.1	ログイン制御の拡張 .....	1-13
1.3.2.2	パスワードの拡張 .....	1-13
1.4	リモート認証方式 .....	1-14
1.4.1	NIS プロトコル .....	1-14
1.4.2	LDAP メカニズム .....	1-15
1.4.3	セキュア・シェル・アプリケーション .....	1-15
1.4.4	SSO/Kerberos メカニズム .....	1-16
1.4.5	Advanced Server for UNIX (ASU) .....	1-16
1.4.6	IPsec プロトコル .....	1-17
1.4.7	SSL，CDSA，および GSS の API .....	1-17
1.5	セキュリティ統合アーキテクチャの概要 .....	1-18

1.5.1	SIA とユーザ変更ルーチン .....	1-20
1.5.2	SIA の構成 .....	1-21
1.5.2.1	SIA メカニズムの初期化 .....	1-23
1.5.2.2	SIA メカニズムの追加 .....	1-24
1.5.2.3	SIA メカニズムの削除 .....	1-25
1.5.3	SIA のロギング .....	1-25
<b>2</b>	<b>システム・リソースの保護</b>	
2.1	アクセス制御の概要 .....	2-1
2.2	Tru64 UNIX のアクセス許可 .....	2-2
2.2.1	ファイルとディレクトリに対する Tru64 UNIX のアクセス許可の表示 .....	2-2
2.2.2	ファイルとディレクトリに対するアクセス許可の設定 (chmod) .....	2-4
2.2.2.1	文字と演算記号を使ってアクセス許可を指定する方法 .....	2-4
2.2.2.1.1	ファイルのアクセス許可の変更 .....	2-5
2.2.2.1.2	ディレクトリのアクセス許可の変更 .....	2-6
2.2.2.1.3	パターン・マッチング文字の使用 .....	2-7
2.2.2.1.4	絶対アクセス許可の設定 .....	2-7
2.2.2.2	8 進数を使ってアクセス許可を指定する方法 .....	2-8
2.2.3	省略時のアクセス許可の設定 .....	2-10
2.2.3.1	ユーザ・マスクの設定 .....	2-13
2.3	アクセス制御リスト (ACL) .....	2-14
2.3.1	ACL の有効化と無効化 .....	2-15
2.3.1.1	NFS 上での ACL の有効化 .....	2-16
2.3.2	ACL の構造 .....	2-17
2.3.3	ACL を使ったアクセス・チェック .....	2-19
2.3.4	ACL の継承 .....	2-21
2.3.4.1	ACL の継承の例 .....	2-23

2.3.5	ACL の管理 .....	2-25
2.3.5.1	dxsetacl インタフェースの使用 .....	2-25
2.3.5.2	getacl コマンドの使用 .....	2-26
2.3.5.3	setacl コマンドの使用 .....	2-26
2.3.6	コマンドやアプリケーションと ACL の相互作用 .....	2-27
2.3.6.1	pax コマンドと tar コマンド .....	2-28
2.3.6.2	アーカイブ・コマンド .....	2-29

### 3 システムの監査

3.1	監査の概要 .....	3-1
3.1.1	監査ファイル .....	3-3
3.1.2	監査ツール .....	3-5
3.1.2.1	コマンド行インタフェース .....	3-5
3.1.2.2	グラフィック・インタフェース .....	3-6
3.1.3	監査マスク .....	3-7
3.1.3.1	システム・コール .....	3-7
3.1.3.2	トラステッド・イベント .....	3-9
3.1.3.3	サイト定義イベント .....	3-11
3.1.3.4	イベント・エイリアス .....	3-12
3.1.4	監査レコード .....	3-13
3.1.4.1	監査レコードの追加エントリ .....	3-15
3.2	監査サブシステムの構成 .....	3-18
3.2.1	監査データ格納場所の集中化 .....	3-22
3.2.1.1	監査ハブへの監査データ格納場所の集中化の構成 ....	3-23
3.2.1.2	監査ハブでのリモート・システムの監査データ記憶域 の構成 .....	3-24
3.3	監査サブシステムの管理 .....	3-24
3.3.1	監査サブシステムの起動時の省略時設定の変更 .....	3-25
3.3.2	監査デーモンの起動，停止，一時停止 .....	3-25

3.3.3	監査ログのアーカイブ .....	3-25
3.3.4	監査データの回復 .....	3-26
3.4	監査イベントの管理 .....	3-26
3.4.1	監査マスクの表示 .....	3-27
3.4.2	システムで監査できるイベントの特定 .....	3-27
3.4.3	監査イベントの有効化 .....	3-28
3.4.4	監査イベントの無効化 .....	3-30
3.4.5	プロセスのトレース .....	3-30
3.4.5.1	トレース・プロセス・データの表示 .....	3-32
3.4.5.2	アクティブ・プロセスの監査 .....	3-33
3.4.5.3	追加のシステム・コール引数の動的な監査 .....	3-34
3.4.6	ファイル操作の監査 .....	3-35
3.5	監査レポートの生成および表示 .....	3-38
3.5.1	監査レコードのフィルタリング .....	3-40
3.5.2	簡略型監査レコードの表示 .....	3-40
3.5.3	監査イベント間の依存関係 .....	3-42
3.6	従来の UNIX のロギング・ツール .....	3-43
3.7	TruCluster での監査 .....	3-45
3.7.1	クラスタ・コマンドの例 .....	3-47
3.8	監査レポートに対する対処 .....	3-49

## A エンハンスト・セキュリティ

A.1	エンハンスト・セキュリティのインストール .....	A-1
A.2	エンハンスト・セキュリティの有効化 .....	A-2
A.2.1	エンハンスト・セキュリティを有効にする際の留意点 ....	A-3
A.2.1.1	NIS の使用 .....	A-3
A.2.1.2	セグメントの共用 .....	A-4
A.2.1.3	Execute Bit Set Only By Root .....	A-4
A.2.2	エンハンスト・セキュリティの構成 .....	A-5

A.2.2.1	有効期間 .....	A-6
A.2.2.2	最短変更時間 .....	A-6
A.2.2.3	変更制御 .....	A-6
A.2.2.4	ログインの最大試行回数 .....	A-6
A.2.2.5	ログイン操作の時間間隔 .....	A-7
A.2.2.6	ログインの時間間隔 .....	A-7
A.2.2.7	端末ごとのログイン・レコード .....	A-7
A.2.2.8	正常ログインのロギング .....	A-7
A.2.2.9	ログイン失敗のロギング .....	A-7
A.2.2.10	エンハnst・プロファイルの自動作成 .....	A-8
A.2.2.11	保証機能 .....	A-8
A.2.2.12	暗号化 .....	A-8
A.3	エンハnst・セキュリティ・データベース .....	A-8
A.3.1	エンハnst (保護) パスワード・データベース .....	A-9
A.3.2	システム省略時設定データベース .....	A-9
A.3.3	端末制御データベース .....	A-10
A.3.4	ファイル制御データベース .....	A-10
A.3.5	デバイス割り当てデータベース .....	A-11
A.4	エンハnst・セキュリティ・データベースの管理ユーティリティ .....	A-12
A.5	エンハnst・セキュリティおよびユーザの認証 .....	A-13
A.5.1	ユーザ・プロファイル .....	A-14
A.5.1.1	/etc/passwd 情報の回復 .....	A-15
A.5.2	エンハnst・セキュリティ認証データベースの完全性 チェック .....	A-15
A.5.3	ファイル制御データベースへのアプリケーションの追加 .	A-16
A.6	エンハnst・セキュリティと NIS .....	A-16
A.6.1	NIS アカウント用テンプレート .....	A-19
A.6.2	エンハnst・セキュリティ環境での NIS マスタ・サーバ の構成 .....	A-20

A.6.2.1	手作業:小規模ユーザ・アカウント・データベースの マップ .....	A-21
A.6.2.2	自動処理:大規模ユーザ・アカウント・データベースの マップ .....	A-21
A.6.3	エンハンスト・セキュリティ環境での NIS スレーブ・サー バの設定 .....	A-22
A.6.4	エンハンスト・セキュリティ環境での NIS クライアントの 設定 .....	A-23
A.6.5	ローカル・アカウントの NIS への移行 .....	A-23
A.6.6	NIS サポートの削除 .....	A-23
A.6.7	導入時の注意点 .....	A-24
A.6.8	NIS のトラブルシューティング .....	A-26
A.7	TruCluster でのエンハンスト・セキュリティ .....	A-27
A.7.1	TruCluster での基本セキュリティからエンハンスト・セ キュリティへのアップグレード .....	A-28
A.7.2	TruCluster でのエンハンスト・セキュリティのインストー ルおよび構成 .....	A-28
A.7.3	アクセス制御リスト .....	A-29
A.7.4	分散ログインと NIS .....	A-30
A.7.5	デーモン .....	A-31
A.8	デバイスの安全確保 .....	A-31
A.8.1	デバイス・セキュリティの特性 .....	A-32
A.8.1.1	dxdevices プログラムを使ったデバイスの変更, 追加お よび削除 .....	A-33
A.8.1.2	dxdevices プログラムを使った省略時の値の設定 .....	A-33
A.8.2	セキュリティ・データベースの更新 .....	A-33
A.9	エンハンスト・セキュリティのトラブルシューティング .....	A-34
A.9.1	ロック・ファイル .....	A-34
A.9.2	必須のファイルとファイル内容 .....	A-35
A.9.2.1	/tcb/files/auth.db データベース .....	A-36



A.9.2.2	/etc/auth/system/ttys.db ファイル .....	A-37
A.9.2.3	/etc/auth/system/default ファイル .....	A-37
A.9.2.4	/etc/auth/system/devassign ファイル .....	A-38
A.9.2.5	/etc/passwd ファイル .....	A-38
A.9.2.6	/etc/group ファイル .....	A-38
A.9.2.7	/sbin/rc[023] ファイル .....	A-38
A.9.2.8	/dev/console ファイル .....	A-38
A.9.2.9	/dev/pts/* および /dev/tty* ファイル .....	A-38
A.9.2.10	/sbin/sulogin ファイル .....	A-38
A.9.2.11	/sbin/sh ファイル .....	A-39
A.9.2.12	/vmunix ファイル .....	A-39
A.9.3	ログインまたはパスワード変更での問題 .....	A-39

## B セキュア・シェル

B.1	セキュア・シェルのサーバとクライアント .....	B-2
B.2	セキュア・シェルの概要 .....	B-2
B.3	セキュア・シェルのサーバとクライアントの構成 .....	B-3
B.3.1	サーバの構成 .....	B-4
B.3.2	クライアントの構成 .....	B-7
B.4	安全性の低いネットワーク・コマンドでセキュア・シェルを使用するための構成 .....	B-9
B.5	セキュア・シェルによるユーザ認証の構成 .....	B-10
B.5.1	パスワード認証の構成 .....	B-11
B.5.2	公開鍵認証の構成 .....	B-12
B.5.2.1	クライアント上の公開鍵認証の構成 .....	B-12
B.5.2.2	サーバ上の公開鍵認証の構成 .....	B-15
B.5.2.3	リモート・サーバへのアクセス .....	B-16
B.5.2.4	ユーザ・アクセスの制限 .....	B-16
B.5.2.5	パスフレーズの管理 .....	B-17

B.5.3	ホスト・ベース認証の構成 .....	B-18
B.6	セキュア・シェル・サーバの管理 .....	B-21
B.6.1	sshd2 デモンの起動, 停止, 再起動, およびリセット .	B-21
B.6.2	ユーザ・アクセスのホーム・ディレクトリへの限定 .....	B-22
B.6.3	公開ホスト鍵と秘密ホスト鍵の作成 .....	B-22
B.6.4	セキュア・シェル接続を使った TCP/IP ポートと X11 データのフォワーディング .....	B-23
B.6.4.1	TCP/IP ポート・フォワーディング .....	B-23
B.6.4.2	X11 フォワーディング .....	B-24
B.7	セキュア・シェル・コマンドの使用 .....	B-25
B.7.1	クライアントとサーバ間のファイル・コピー .....	B-25
B.7.1.1	scp2 コマンドの使用 .....	B-25
B.7.1.2	sftp2 コマンドの使用 .....	B-26
B.7.2	サーバへのログインとサーバ上でのコマンド実行 .....	B-26

## C Single Sign On

C.1	Kerberos サーバおよびクライアント .....	C-1
C.2	Kerberos 認証プロセス .....	C-2
C.3	SSO ソフトウェアのアップデート .....	C-3
C.4	SSO ソフトウェアのインストールおよび構成 .....	C-3
C.4.1	Windows 2000 システムでの SSO ソフトウェアのインストールおよび構成 .....	C-4
C.4.1.1	Active Directory スキーマの拡張 .....	C-4
C.4.1.2	MMC のアップデート .....	C-5
C.4.2	Tru64 UNIX システムでの SSO ソフトウェアのインストールおよび構成 .....	C-6
C.4.2.1	SSO ソフトウェアの構成 .....	C-6
C.4.2.2	TruCluster Server 環境での SSO ソフトウェアの構成 .....	C-9
C.4.2.3	Kerberos と他の SIA メカニズムの併用 (必要に応じて) .....	C-9

C.5	Tru64 UNIX の SSO の構成ファイル .....	C-10
C.5.1	krb.conf ファイル .....	C-10
C.5.2	krb.realms ファイル .....	C-12
C.5.3	v5srvtab ファイル .....	C-14
C.5.4	.k5login ファイル .....	C-15
C.5.5	ldapcd.conf ファイル .....	C-16
C.5.6	ldapusers.deny ファイル .....	C-18
C.6	アカウントおよびグループの作成 .....	C-19
C.6.1	ユーザ・アカウントの作成 .....	C-19
C.6.1.1	Tru64 UNIX の creacct コマンドを使用したユーザ・アカウントの作成 .....	C-19
C.6.1.2	MMC インタフェースを使用したユーザ・アカウントの作成 .....	C-20
C.6.2	プリンシパルのパスワードの設定 .....	C-24
C.6.3	コンピュータ・アカウントの作成 .....	C-24
C.6.4	グループの作成 .....	C-25
C.7	SSO ソフトウェアの管理 .....	C-27
C.7.1	チケットの要求 .....	C-28
C.7.2	チケットの表示 .....	C-28
C.7.3	信任状キャッシュの削除 .....	C-29
C.7.4	サービス鍵テーブルの管理 .....	C-29
C.8	SSO ソフトウェアのトラブルシューティング .....	C-31
C.8.1	SSO の構成パラメータの問題 .....	C-31
C.8.2	kinit コマンドの使用および Tru64 UNIX の初回チケットの取得に関する問題 .....	C-31
C.8.3	Tru64 UNIX へのパスワードの入力 .....	C-33
C.8.4	TruCluster における SSO の問題 .....	C-34

## D LDAP (Lightweight Directory Access Protocol)

D.1	LDAP の概要 .....	D-1
-----	----------------	-----

D.2	Tru64 UNIX LDAP クライアント・ソフトウェアのインストール .....	D-2
D.3	Tru64 UNIX LDAP クライアント・ソフトウェアの構成 .....	D-3
D.3.1	ldapcd.conf ファイルの更新 .....	D-3
D.3.2	LDAP ランタイム構成変数の設定 .....	D-6
D.4	LDAP クライアント・デーモンの管理 .....	D-7
D.5	アクセス制御の管理 .....	D-7
D.5.1	ldapusers.deny ファイル .....	D-7
D.5.2	ldapusers.allow ファイル .....	D-8
 <b>E C2 レベル・セキュリティの構成</b>		
E.1	セキュリティ・ポリシーの確立 .....	E-2
E.2	最小限の C2 構成 .....	E-5
E.3	初期構成 .....	E-6
E.3.1	一般的な構成 .....	E-6
E.3.2	secconfig を使用した、エンハンスド・パスワードと認証 .....	E-6
E.3.3	ライブラリ .....	E-7
E.3.4	アカウント・プロトタイプとアカウント・テンプレート .....	E-8
E.3.5	監査サブシステムの構成 .....	E-8
E.3.6	システムの保全性の確認 .....	E-9
E.3.7	ネットワーク・セキュリティの構成 .....	E-9
E.3.8	インストール後のセキュリティの構成 .....	E-11
E.3.8.1	リモート・アクセスの umask .....	E-11
E.3.8.2	デバイス .....	E-11
E.3.8.3	アカウント .....	E-11
E.3.8.4	root アクセス .....	E-12
E.3.9	ネットワークの構成 .....	E-13
E.4	物理的セキュリティ .....	E-14
E.5	アプリケーション .....	E-14
E.6	定期的に行うセキュリティ管理手順 .....	E-15

E.7	参照ドキュメントとチェック・ツール .....	E-19
用語集		
索引		
例		
1-1	省略時の /etc/sia/matrix.conf ファイル .....	1-23
1-2	/var/adm/sialog ファイルの例 .....	1-25
2-1	8 進数によるアクセス許可の設定 .....	2-10
2-2	ファイルの ACL の表示 .....	2-26
2-3	ファイルに対する ACL の設定 .....	2-26
3-1	アクティブ監査セッションの例 .....	3-33
B-1	sshd2_config ファイルの例 .....	B-4
B-2	ssh2_config ファイルの例 .....	B-7
B-3	公開鍵認証によるログイン時の画面表示 .....	B-16
C-1	krb.conf ファイルの例 .....	C-11
C-2	krb.realms ファイルの例 .....	C-14
C-3	.k5login ファイルの例 .....	C-15
C-4	ldapcd.conf ファイルの例 .....	C-16
C-5	/etc/ldapusers.deny ファイルの例 .....	C-18
D-1	ldapcd.conf ファイルの例 .....	D-4
D-2	省略時の ldapusers.deny ファイル .....	D-8
図		
1-1	セキュリティ統合アーキテクチャ .....	1-20
2-1	ファイルに対する Tru64 UNIX のアクセス許可フィールド ...	2-3
3-1	監査サブシステム .....	3-2
3-2	クラスタ内の監査データの流れ .....	3-46
A-1	NIS およびエンハンスド・セキュリティ・ファイル .....	A-18

C-1	「新しいオブジェクト — ユーザー」ウィンドウ 必要な情報	C-21
C-2	「新しいオブジェクト — ユーザー」ウィンドウ パスワード 情報 .....	C-22
C-3	Tru64 UNIX ユーザのプロパティ・ダイアログ・ボックス ...	C-23
C-4	グループのプロパティ・ダイアログ・ボックス .....	C-27

## 表

1-1	認証方式の比較 .....	1-4
1-2	セキュア・シェル・コマンド .....	1-16
2-1	Tru64 UNIX のアクセス許可コード .....	2-2
2-2	アクセス許可を示す 8 進数の組み合わせ .....	2-9
2-3	8 進数とアクセス許可フィールドの対応関係 .....	2-9
2-4	ユーザ・マスクのアクセス許可の組み合わせ .....	2-11
2-5	ACL エントリの例 .....	2-19
3-1	監査されない可能性があるシステム・コール .....	3-8
3-2	状態依存情報 .....	3-43
3-3	/var/adm にある従来の UNIX ログ・ファイル .....	3-44
A-1	エンハンスド・セキュリティ・データベース .....	A-8
A-2	NIS のトラブルシューティング .....	A-27
B-1	従来の安全性の低いネットワーク・コマンドとセキュア・シェ ル・コマンド .....	B-1
C-1	SSO の構成ファイル .....	C-10
C-2	キャッシュ・パラメータ .....	C-17

---

## まえがき

本書では、識別、認証および承認の方法、リソースの保護、および監査を含む Tru64 UNIX セキュリティの概念と管理方法について説明します。

### 本書の対象読者

本書は、Tru64 UNIX オペレーティング・システム・ソフトウェアで稼動しているシステムのセキュリティの構成および管理を担当するユーザを対象としています。

### 改訂情報

本書には、今回のバージョンで利用できる識別、認証、および承認の方法に関する情報が含まれています。

前回まで本書の「一般ユーザ編」に記載されていた情報は、『*Tru64 UNIX ユーザーズ・ガイド*』に記載されています。

前回まで本書の「プログラマ編」に記載されていた情報は、『*セキュリティ・プログラミング・ガイド*』に記載されています。

### 本書の構成

本書の各章の内容は以下のとおりです。

第 1 章	Tru64 UNIX によるユーザの認証で使用される、ローカルおよびリモートでの識別、認証、および承認の方法について説明します。
第 2 章	Tru64 UNIX を構成し、システム・リソースへのアクセスを管理する方法を説明します。
第 3 章	Tru64 UNIX を監査し、システムの動作を表示する方法を説明します。
付録 A	エンハンスド・セキュリティを Tru64 UNIX にインストールし、構成する方法を説明します。
付録 B	Tru64 UNIX でセキュア・シェル・ソフトウェアを構成する方法を説明します。

付録 C	Tru64 UNIX で Single Sign On (Kerberos) ソフトウェアを構成する方法を説明します。
付録 D	LDAP (Lightweight Directory Access Protocol) を使って、Tru64 UNIX をクライアントとして構成する方法を説明します。
付録 E	C2 レベル以上のセキュリティに適合するように Tru64 UNIX を構成する方法を説明します。

## 参考資料

それぞれの章の情報に関する追加情報については、以下のマニュアルを参照してください。

- 『システム管理ガイド』 マニュアルでは、Tru64 UNIX における一般的な管理作業を実施する方法について説明します。
- 『*Tru64 UNIX* ユーザーズ・ガイド』 では、Tru64 UNIX オペレーティング・システム・ソフトウェアを実行するシステムへのユーザによるアクセスおよび対話の方法について説明します。
- 『ネットワーク管理ガイド：接続編』 マニュアルでは、ネットワークの設定、構成、およびトラブルシューティングの方法について説明します。
- 『リリース・ノート』 には、マニュアルに記載されていない、セキュリティに関する重要な情報が含まれている場合があります。

## 表記法

本書では以下の表記法を使用します。

\	行の最後のバックスラッシュは、継続を表します。
#	番号記号は、Tru64 UNIX システムに root としてログインした場合のシステム・プロンプトを表します。
<b>net stop</b>	太字 (ボールド体) はユーザが入力する文字を示します。
>>>	コンソール・サブシステムのプロンプトです。
<i>file</i>	イタリック体 (斜体) は、変数値、プレースホルダ、および関数の引数名を示します。
[   ]	構文の定義では、大カッコはオプションの項目を示し、中カッコは必須項目を示します。大カッコまたは中カッコの中の項目を縦線で区切っている場合は、そこに併記されている項目の中から 1 つの項目を選択することを示します。
{   }	



...

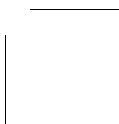
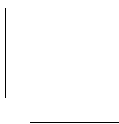
構文定義では、水平の反復記号は、前の項目を 1 回以上繰り返して使用できることを示します。

cat(1)

リファレンス・ページの参照には、該当するセクション番号をカッコ内に示します。たとえば、cat(1) は、cat コマンドについての情報が、リファレンス・ページのセクション 1 に記載されていることを示します。

[Ctrl/x]

この記号は、スラッシュの前に指定されているキーを押しがら、スラッシュの後のキーまたはマウス・ボタンを押すことを示します。例中では、このようなキーの組み合わせは、四角あるいは大カッコで囲まれて示されます (たとえば、[Ctrl/C])。



---

## 識別，認証，および承認

識別，認証，および承認は，システム・セキュリティに不可欠の要素です。システム・セキュリティは，セキュリティ・ポリシを作成することによって構築します。セキュリティ・ポリシによって，システム上のリソースにアクセスするユーザやアプリケーション（エンティティ）との間，およびローカル・ネットワークや拡張ネットワークの内部にある他のシステムとの間で，必要な信用レベルが決まります。

システムのリソースにアクセスするエンティティを完全に信用するには，まずそのエンティティを識別する必要があります。エンティティは識別されたら，システムに対して自身が本物であることを証明しなければなりません。最後に，エンティティを識別して認証した後，システムはそのエンティティによるアクセスが承認されるシステム上のリソースを決定する承認情報を適用します。

この章には，次の情報が含まれています。

- 認証の概要
- 認証の実装
- ローカル認証方式
- リモート認証方式
- セキュリティ統合アーキテクチャの概要

### 1.1 認証の概要

認証を成功させるには，ユーザやアプリケーション（エンティティ）がシステムに対して何らかの形で信任状を提示する必要があります。信任状として使えるのは，そのユーザだけが知っていること（ユーザ名とパスワード），そのユーザ自身の情報（バイオメトリックスや網膜スキャン），またはそのユーザだけが持っているもの（スマート・カードや証明書）です。

Tru64 UNIX オペレーティング・システム・ソフトウェアでは、従来からチャレンジ・レスポンス方式によるユーザ認証にローカル認証方式 (ユーザのログイン名とパスワード) が使われていました。この場合、システムはユーザが入力した名前とパスワードをローカルのユーザ・アカウント・データベース内のエントリと照合します。

ユーザ・アカウント・データベース内のユーザ・エントリは、管理者がユーザ・アカウントを作成したときに作成されます。一致するエントリがユーザ・アカウント・データベースに存在しない場合、そのユーザは認証されません。一致するエントリがユーザ・アカウント・データベースに存在し、パスワードが正しければ、そのユーザは認証されます。

コンピュータ環境が分散化、オープン化、および多様化するにしたがって、何らかのネットワーク接続を経由して別のシステムからシステムにアクセスするリモート・ユーザに対するチャレンジ・レスポンス方式による認証では、次のような問題が発生しました。

- 分散型の認証データベースがない

システムごとに独立したユーザ・アカウント・データベースが維持され、データベースの情報を異なるシステム間で共有できません。このため、ユーザは利用するリソースが存在するシステムごとにユーザ・アカウントを作成する必要があります。また、各システムは他のシステムの識別情報やユーザの識別情報の正当性を安全な信頼性の高い方法で判定できません。

- システム間で共通の暗号化方式がない

暗号化されたデータをシステム間でやり取りできません。ユーザ名やパスワードを含むデータが平文のままネットワーク経由で配布されます。

- データの完全性を判定できない

各システムは、データが送信中に傍受、参照、変更されたかどうかを簡単に検出できません。

- データの送信元が判定できない

各システムは、データの送信元を簡単に証明できません。

認証に関するこれらの問題を解決するため、リモート認証方式 (アプリケーション、メカニズム、プロトコル、およびアプリケーション・プログラミング・インタフェース (API)) が開発されました。この方式では、何らかのネッ

トワーク接続を経由して別のシステムからアクセスするユーザのために、次のようなりモート認証サービスが提供されます。

- 分散型の認証データベース

ユーザ認証情報は、プライマリ・サーバに保存され、維持されます。各システムはプライマリ・サーバにユーザ認証要求を送信します。システムごとに独自のユーザ・アカウント・データベースを維持する必要はありません。各システムでは、他のシステムの識別情報やユーザの識別情報の正当性を安全で信頼性の高い方法で判定する認証方式が使われます。

- システム間で共通の暗号化方式

各システムは、ユーザに意識させることなく暗号化されたデータをやり取りします。

- データの完全性を判定する機能

各システムは、データが転送中に傍受、参照、および変更されたかどうかを判定できます。

- データの送信元を判定する機能

各システムがデータの送信元を証明できるため、ユーザやエンティティはデータに関して特定の操作を実行した事実やデータの所有者に関する証拠を否定できません。

ローカル認証方式とリモート認証方式では、ローカル認証やリモート認証に必要な複数の信用レベルを持つセキュリティ・ドメインを作成するセキュリティ・ポリシを実装できます。

ローカル認証の場合は、Tru64 UNIX システムのリソースへのアクセスを要求するユーザの認証をそのシステムが担当します。リモート認証の場合は、Tru64 UNIX システムがリモート・サーバのクライアントになり、そのシステムのリソースへのアクセスを要求するユーザの認証処理をリモート・サーバに依頼します。表 1-1 は、各種のローカルおよびリモート認証方式について、認証の実装方法を比較した表です。

表 1-1: 認証方式の比較

認証方式	ローカル/リモート	実装方法
基本 (BSD) セキュリティ・メカニズム	ローカル	パスワード
エンハンスド・セキュリティ・メカニズム	ローカル	パスワード
r* コマンド (rsh, rcp, および rlogin)	ローカル	<ul style="list-style-type: none"> <li>パスワード</li> <li>ホスト・ベース</li> </ul>
セキュア・シェル・アプリケーション	リモート	<ul style="list-style-type: none"> <li>パスワード</li> <li>ホスト・ベース</li> <li>公開鍵</li> </ul>
NIS (Network Information Service) プロトコル	リモート	パスワード
LDAP (Lightweight Directory Access Protocol)	リモート	パスワード
Single Sign On (SSO)/Kerberos メカニズム	リモート	パスワードと秘密鍵
Advanced Server for UNIX (ASU)	リモート	パスワード
IPsec プロトコル	リモート	公開鍵
SSL (Secure Socket Layer) API	リモート	公開鍵
CDSA (Common Data Security Architecture) API	リモート	公開鍵
GSS API	リモート	パスワードと秘密鍵

## 1.2 認証の実装

この項では、認証の実装について説明します。

### 1.2.1 パスワード認証

パスワード認証では、ユーザがパスワードで保護されたユーザ・アカウントを持つ必要があります。このユーザ・アカウントは、ローカルの Tru64

UNIX パスワード認証メカニズムとリモートのパスワード認証方式のどちらかを使って認証できます。

ユーザが認証を必要とする操作 (システムへのログインなど) を実行すると、ユーザはパスワードの入力を求められます。システムはユーザが入力した名前とパスワードをユーザ・アカウント・データベース内のエントリと照合します。ユーザ・アカウント・データベース内のユーザ・エントリは、管理者がユーザ・アカウントを作成したときに作成されます。一致するエントリがユーザ・アカウント・データベースに存在しない場合、そのユーザは認証されません。一致するエントリがユーザ・アカウント・データベースに存在し、パスワードが正しければ、そのユーザは認証されます。

## 1.2.2 ホスト・ベース認証

ホスト・ベース認証は、ホスト名とユーザ名の識別に基づく UNIX の非対話型認証方式です。ホスト・ベース認証では、各ユーザが特定のホストからネットワーク・コマンド (`rcp`, `rlogin`, `rsh` の各コマンドや同等のセキュア・シェル・コマンドなど) を使用するユーザに対して、自分のユーザ・アカウントへのアクセスを許可できます。セキュア・シェル・コマンドの詳細については、1.4.3 項を参照してください。

ユーザがアクセスを許可するときは、自分のホーム・ディレクトリ内の `.rhosts` ファイルおよび `.shosts` ファイルにホスト名とユーザ名から成るエントリを追加します。ユーザが使用するファイルは、次の条件で決まります。

- ユーザが `rcp`, `rlogin`, `rsh` の各コマンドを使用する場合は、`.rhosts` ファイルを使用します。
- ユーザがセキュア・シェル・コマンドを使用するか、または `rcp`, `rlogin`, `rsh` の各コマンドでセキュア・シェル接続を使用するように構成した場合は、`.rhosts` ファイルおよび `.shosts` ファイルを使用します。これらのコマンドでセキュア・シェル接続を使用するように構成する方法の詳細については、B.4 節を参照してください。

`.shosts` ファイルの用途や形式は `.rhosts` ファイルと同じですが、このファイルを読み込むのはセキュア・シェル・サーバだけです。2 つのファイルが両方とも存在する場合、セキュア・シェル・サーバは最初に `.rhosts` ファイルを読み込み、次に `.shosts` ファイルを読み込みます。どちらかのファイルで特定の接続に対するアクセスが許可されてい

れば、もう一方のファイルでアクセスが許可されていなくてもセキュア・シェル接続が使われます。

セキュア・シェルのホスト・ベース認証では、他にも構成が必要な項目があります。セキュア・シェルのホスト・ベース認証の詳しい構成方法については、B.5.3 項を参照してください。

.rhosts ファイルや .shosts ファイルのエントリには、該当するユーザを認証するのに使うホストの完全修飾ドメイン名を入れます。たとえば、ホスト zeus 上にある \$HOME/peter/.shosts ファイルの次のエントリは、リモート・ホスト saturn.ne.corp.com のユーザ evan と justin に対して、zeus 上にある peter のユーザ・アカウントへのログインを許可します。

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

ユーザ evan と justin はパスワードを入力する必要がなく、次のコマンドを入力すれば zeus 上にある peter のユーザ・アカウントにログインします。

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

ホスト・ベース認証は、スクリプトや自動化されたプロセス (cron ジョブ、自動バックアップ、自動ファイル転送など)、および認証情報を入力するユーザがないその他の状況に最適です。非対話式のログインは、本質的に安全とは言えません。ユーザに対するチャレンジがない認証では、必ずある程度リスクを想定する必要があります。

.rhosts と .shosts のファイル形式は同じです。詳細については、.rhosts(4) を参照してください。

### 1.2.3 公開鍵認証

公開鍵認証は暗号化技術の一種で、2 つの通信プリンシパル (通常はクライアントとサーバ) が公開鍵と秘密鍵を使って相手のシステムやユーザを認証し、やり取りするデータを暗号化および復号します。一方の公開鍵または秘密鍵で暗号化したデータは、もう一方の対応する公開鍵または秘密鍵でしか復号できません。暗号化の詳細については、1.2.5 項を参照してください。

公開鍵は公開され、ユーザが通信先とするシステムの特定のディレクトリにコピーされます。秘密鍵は一切公開・配布されず、ユーザは秘密鍵に秘密のパスフレーズを割り当てます。公開鍵と秘密鍵には数学的な相互関係がありますが、一方の鍵からもう一方の鍵を算出することは不可能です。このため、公開鍵の情報から対応する秘密鍵が漏洩するおそれはありません。



公開鍵を使う場合は、システム (クライアント) が他のシステム (サーバ) にサービスを要求した時点で認証が開始されます。

1. クライアントがサーバの公開鍵のコピーを使ってメッセージを暗号化し、サーバにそのメッセージを送信します。
2. サーバがユーザにパスフレーズの入力进行、入力されたパスフレーズとサーバの秘密鍵を使ってメッセージを復号します。
3. サーバがクライアントの公開鍵のコピーを使って応答メッセージを暗号化し、クライアントにそのメッセージを送信します。
4. クライアントが自身の秘密鍵を使ってメッセージを復号します。

それぞれの秘密鍵に対応する公開鍵は 1 つしかないので、暗号化と復号が成功することによってプリンシパルが認証されます。

#### 1.2.3.1 デジタル署名と証明書

あるユーザが鍵のペアを生成して公開鍵を公開する場合、使われる識別情報が本物だという保証はありません。公開鍵の不正利用を防ぐには、デジタル署名およびデジタル証明書と呼ばれるデジタル文書を使用します。

デジタル署名は、文書を基にした小さなデータとプリンシパルの秘密鍵から生成されます。通常、デジタル署名はハッシュと秘密の署名アルゴリズムを使って作成され、秘密鍵で暗号化されます。ハッシュとは、テキストの文字列から生成される数値のことです。

元の電子メッセージのバイナリ・コードに対してハッシュ・アルゴリズムが実行されると、いわゆるメッセージ・ダイジェスト (元のメッセージに固有の 160 ビットから成る数字列) が生成されます。このメッセージ・ダイジェストに対して秘密の署名アルゴリズムが実行されます。この署名処理では、プリンシパルの秘密鍵が署名アルゴリズムに組み込まれます。この結果得られた数字列がデジタル署名です。プリンシパルは、署名を生成した側のプリンシパルの公開鍵を持っていれば、その署名を検証できます。

デジタル証明書は、個人、サーバ、会社、その他のプリンシパルを識別するための電子文書で、公開鍵によってそのプリンシパルと結び付けられています。運転免許、パスポート、その他の一般的に使われる個人 ID と同じように、デジタル証明書はプリンシパルの公認された身元証明を提供します。

デジタル証明書の発行、検証、および管理は、認証局 (CA) と呼ばれる信頼される第三者機関によって行われます。ただし、テスト目的の証明書は IPsec で提供されるツールを使って作成できます。他の身分証明と同じように、CA の信頼性は非常に重要です。CA が識別情報を検証する方法は、各 CA のポリシーによって異なります。一般に、CA はデジタル証明書を発行する前に、公開された検証手順を使って、デジタル証明書を要求するプリンシパルが申し立てどおりのプリンシパルであることを確認します。発行されたデジタル証明書は、CA のデータベースに保管されます。

デジタル証明書には、識別するプリンシパルの名前、そのプリンシパルの公開鍵、有効期限、証明書を発行した CA の名前、シリアル番号、およびその他の情報が入っています。また、最も重要な要素として発行元の CA のデジタル署名が含まれています。これにより、プリンシパルが不明でも CA が信頼されていれば、デジタル署名を含むデジタル証明書によってそのプリンシパルを識別することができます。

#### 1.2.4 秘密鍵認証

秘密鍵認証は暗号化技術の一種で、2 つの通信プリンシパル (通常はクライアントとサーバ) が同じ秘密鍵 (セッション鍵) を使ってシステムやユーザを認証し、やり取りするデータを暗号化および復号します。一方のセッション鍵で暗号化したデータは、もう一方の一致するセッション鍵でしか復号できません。暗号化の詳細については、1.2.5 項を参照してください。

信頼される KDC (Key Distribution Center) は、プリンシパル間の通信を仲介し、プリンシパル間専用のセッション鍵を作成します。秘密鍵を使う場合は、ユーザがログインした時点で認証が開始されます。

1. ユーザがユーザ名とパスワードをシステム (クライアント) に入力します。
2. クライアントが入力されたユーザ名を KDC に送信します。

KDC は、仲介する各ユーザのエントリを暗号化されたプリンシパル・データベースに保存しています。このエントリには、ユーザについての情報 (パスワードやマスター鍵など) が入っています。

3. KDC がプリンシパル・データベースから該当するユーザのエントリを検索します。一致するエントリがあれば、KDC は次のものを作成します。
  - クライアントと KDC が専用で使うセッション鍵。

- セッション鍵のコピー，ユーザの名前，および有効期限が入った TGT (チケット・グランティング・チケット)。KDC は KDC のマスター鍵を使ってこのチケットを暗号化します。
4. KDC がセッション鍵と TGT の入ったメッセージを作成します。このメッセージはユーザのマスター鍵で暗号化され，クライアントに送信されます。
  5. クライアントが一方方向ハッシュ関数を使ってユーザのパスワードをユーザのマスター鍵に変換します。次に，そのマスター鍵を使ってメッセージを復号し，セッション鍵と TGT を取り出します。

セッション鍵と TGT は，ユーザの信任状キャッシュ (クライアント上のファイル) に保存されます。クライアントはそれ以降の KDC との通信にこのセッション鍵と TGT を使用します。クライアントを含むどのプリンシパルも，TGT の内容を参照および変更できません。

次に示す手順では，2 つのプリンシパル (クライアント) が通信しようとしている場合の認証処理について説明します。

1. 要求する側のプリンシパルが次の内容を含むメッセージを KDC に送信します。
  - TGT
  - 通信相手となるプリンシパルの識別情報
  - 認証子 (プリンシパルと KDC が共有するセッション鍵を使って暗号化されたタイムスタンプ)
2. KDC がマスター鍵を使って TGT を復号します。前述のように，TGT にはユーザの名前とセッション鍵のコピーが入っています。次に，KDC がセッション鍵を使って認証子を復号します。

該当するプリンシパルと KDC だけが同じセッション鍵を持っているので，認証子が正常に復号されれば，KDC とプリンシパルは互いの識別情報を確認することができます。
3. KDC がそれぞれのプリンシパルのためにチケットのペアを作成します。各チケットには，相手側のプリンシパルの名前，タイムスタンプ，チケットの有効期間，およびプリンシパルどうしが専用で使う新しいセッション鍵が入っています。

4. KDC が要求する側のプリンシパル用のチケットから成るメッセージを作成します。この中には、要求される側のプリンシパルの鍵で暗号化した要求される側のプリンシパル用のチケットが入っています。

KDC は、要求する側のプリンシパルと共有するセッション鍵を使ってこのメッセージを暗号化し、要求する側のプリンシパルに送信します。

5. 要求する側のプリンシパルが KDC と共有しているセッション鍵を使ってメッセージを復号します。新しいセッション鍵と要求される側のプリンシパル用のチケットが取り出されます。
6. 要求する側のプリンシパルは新しいセッション鍵を使って認証子 (タイムスタンプ) を暗号化し、この認証子と要求される側のプリンシパル用のチケットが入ったメッセージを要求される側のプリンシパルに送信します。
7. 要求される側のプリンシパルが次の処理を実行します。
  - a. 自分のマスター鍵を使ってチケットを復号します。
  - b. チケットからセッション鍵を取り出します。
  - c. 取り出したセッション鍵を使って認証子を復号します。

ここで使われるセッション鍵は 2 つのプリンシパル専用なので、認証子の復号に成功すれば、認証子の暗号化と復号に同じセッション鍵が使われたことになり、プリンシパル間で互いの身元を確認することができます。

### 1.2.5 暗号化方式

暗号化とは、システムが暗号化アルゴリズム (暗号) を使ってデータを暗号化および復号することです。暗号によって、平文 (暗号化されていないテキスト) が暗号文 (暗号化されたテキスト) に変換されます。復号とは、同じ暗号を使って暗号文を平文に変換することです。

クライアントとサーバは同じ暗号をサポートする必要があります。また、サポートするプロトコルのバージョン、許容する暗号強度に関する企業方針、政府の輸出規制、データの機密性、暗号処理の速度などの要因に応じて、複数の暗号スイート (暗号セット) をサポートすることができます。管理者は、クライアントとサーバでサポートされる任意の暗号スイートを有効または無効にできます。

クライアントとサーバは、接続開始時の処理の一環として、互いの共通の最高強度の有効な暗号スイートを特定し、それをセッションに使用します。

ほぼすべての暗号で、鍵 (暗号化されていないテキストと何らかの方法で結び付けられた値) とアルゴリズム (鍵とテキストを結びつける公式) が使われます。暗号には、一般に次の 2 種類があります。

- メッセージを複数のセクション (64 ビット単位のテキストなど) に分割し、各セクションに鍵を結び付けるブロック暗号。現在の暗号は、ほとんどがブロック暗号です。
- ビットごとに 1 つの鍵を適用するストリーム暗号。

暗号アルゴリズムには、たとえば次のようなものがあります。

- DES (Data Encryption Standard)
- 3DES
- RSA (Ron Rivest , Adi Shamir , Len Adleman)
- Blowfish および Twofish

## 1.3 ローカル認証方式

ローカル認証とは、Tru64 UNIX システムがリソースへのアクセスを要求するユーザを何らかの認証方式によって認証することです。この項では、次のローカル認証方式について説明します。

- 基本 (BSD) メカニズム
- エンハンスド・セキュリティ・メカニズム

### 1.3.1 基本 (BSD) メカニズム

基本 (BSD) セキュリティは、Tru64 UNIX オペレーティング・システム・ソフトウェアが稼動するシステムで利用できる標準のセキュリティ・レベルです。

基本セキュリティでは、`/etc/passwd` ファイルに格納されているユーザ情報を使ってユーザを認証します。Tru64 UNIX のユーザ・アカウントを作成すると、特に指定しなければそのユーザのエントリがローカル・システムの `/etc/passwd` ファイルに追加されます。各エントリには、各ユーザに関する情報 (ユーザ名、ユーザ ID (UID)、パスワード、ログイン・ディレクトリ、

シェルなど)が入っています。/etc/passwd ファイルおよびユーザ・アカウントの作成の詳細については、『システム管理ガイド』を参照してください。

ユーザが Tru64 UNIX システムにログインするためにユーザ名とパスワードを入力すると、Tru64 UNIX サーバはユーザが入力した情報を etc/passwd ファイルの各エントリと照合します。一致するエントリが存在し、パスワードが正しければ、認証に成功し、ユーザはログインすることができます。一致するエントリが存在しないと、認証に失敗し、ユーザはログインできません。

/etc/passwd ファイルの編集や整合性チェックを実行するには、pwck コマンドまたは vipw コマンドを使用します。基本セキュリティを使用している場合、ユーザは passwd コマンドを使ってパスワードを変更します。

基本 (BSD) メカニズムでは、ローカル・システム上にある /etc/group ファイルによってユーザ承認情報が提供されます。/etc/passwd ファイルの編集や整合性チェックを実行するには、grpck コマンドまたは vipw コマンドを使用します。/etc/group ファイルおよびユーザ・グループの作成の詳細については、『システム管理ガイド』を参照してください。

### 1.3.2 エンハンスト・セキュリティ・メカニズム

Tru64 UNIX オペレーティング・システムには、エンハンスト・セキュリティ・メカニズムがオプションで用意されています。エンハンスト・セキュリティ・メカニズムは、BSD メカニズムをベースに構築されていますが、セキュリティ情報の一部がデータベースに移動され、侵入回避機能が追加され、アカウントやパスワードの制御が強化されています。エンハンスト・セキュリティのデータベースでは、BSD のファイルに比べてシステム・セキュリティの制御が強化されています。

エンハンスト・セキュリティ・メカニズムをインストールして構成したシステムのことをトラステッド・システムと呼びます。エンハンスト・セキュリティ機能をすべて有効にすると、『*Trusted Computer System Evaluation Criteria*』(TCSEC, いわゆる『*Orange Book*』)で定義された、C2 クラスの信頼性を満たすシステムを構成できます。また、そのシステムは『*Information Technology Security Evaluation Criteria*』(ITSEC)で定義された F-C2 機能クラスも満たしています。C2 クラスの信頼性のセキュリティを満たすシステムの構築方法については、付録 E を参照してください。

エンハンスト・セキュリティは、次の機能拡張を提供することで BSD セキュリティを拡張しています。

- ログイン制御の拡張
- パスワードの拡張

#### 1.3.2.1 ログイン制御の拡張

エンハンスド・セキュリティのログイン制御に関する機能を次に示します。

- 最後にログインに成功および失敗した端末の記録
- 最後にログインに成功した日時の記録
- 最後にログインに失敗した日時の記録
- 連続して失敗したログイン操作回数の記録
- 連続して指定回数以上アクセス操作に失敗したアカウントの自動ロックアウト
- ログイン失敗後の次のログイン操作までの遅延時間，およびログイン操作が完了しないままログイン操作の失敗が確定するまでの最大時間に関する端末ごとの設定
- 端末からの新しいアクセスがロックされるまでの最大連続失敗ログイン操作回数に関する端末ごとの設定
- 最後に成功したログイン操作と最後に失敗したログイン操作に関する情報をログイン時に表示する機能
- 端末経由のログインが発生するごとにログイン情報を記録する機能

#### 1.3.2.2 パスワードの拡張

エンハンスド・セキュリティでは，パスワードの制御に関して以下の機能が提供されます。

- パスワードの最大長の設定 (最大 80 文字)
- パスワードの有効期間の設定
- パスワードの最小長の設定
- システム生成パスワード (意味のない音節で構成される発音可能なパスワード，文字セット内のランダムな文字で構成される発音できないパスワード，または任意のアルファベット (ASCII の英字) で構成される発音できないパスワードなど)

- ユーザごとのパスワード生成フラグ (たとえば, ユーザにシステム生成パスワードの使用を強制できる)
- ユーザのパスワードを最後に変更した (そのユーザ以外の) 人の記録
- パスワードの使用履歴

## 1.4 リモート認証方式

リモート認証とは, Tru64 UNIX システムがリモート・サーバのクライアントになり, そのシステムのリソースへのアクセスを要求するユーザの認証処理をリモート・サーバに依頼することです。各システムが使用するリモート認証方式は, Tru64 UNIX システムとリモート・サーバで設定される共通のリモート認証方式およびやり取りされるデータの種類によって決まります。

- リモート・サーバが認証するユーザ名とパスワードのデータは, NIS や LDAP を使ってやり取りできます。
- `rcp`, `rlogin`, `rsh` の各ネットワーク・コマンドを使ってやり取りされるデータには, セキュア・シェルまたは Kerberos を使用できます。
- `ftp`, `telnet` の各ネットワーク・コマンドを使ってやり取りされるデータには, Kerberos を使用できます。
- `rcmd()` 関数を使用するアプリケーションによってやり取りされるデータには, セキュア・シェルを使用できます。
- IP トランスポートを使ってやり取りされるデータには, IPsec を使用できます。
- 分散アプリケーションによってやり取りされるデータには, SSL, CDSA, または GSS の各 API を使用できます。

以降の項では, 各種のリモート認証方式について説明します。

### 1.4.1 NIS プロトコル

NIS は, LAN (ローカル・エリア・ネットワーク) で情報を共有するための分散クライアント/サーバ型データ検索サービスです。

NIS サーバには, BSD セキュリティやエンハンスド・セキュリティのユーザ・アカウント・データベースの一部または全部を格納できます。ユーザが NIS クライアントとして構成されたシステムにログインするためにユーザ名とパスワードを入力すると, NIS クライアントはそのユーザ情報を NIS



サーバに送信して認証処理を依頼します。NIS サーバはユーザ情報をデータベース内のエントリと照合し、その結果を NIS クライアントに返信します。一致するエントリが存在し、パスワードが正しければ、認証に成功し、ユーザはログインすることができます。一致するエントリが存在しないと、認証に失敗し、ユーザはログインできません。

Tru64 UNIX システムは、NIS クライアントおよび NIS サーバとして構成できます。NIS の詳細については、『ネットワーク管理ガイド：サービス編』を参照してください。

### 1.4.2 LDAP メカニズム

LDAP (Lightweight Directory Access Protocol) は、TCP/IP 上で動作するインターネット標準の分散クライアント/サーバ型ディレクトリ・サービスのプロトコルです。

LDAP サーバは、ユーザの識別情報と承認情報を LDAP ディレクトリのエントリとして格納します。ユーザが LDAP クライアントとして構成されたシステムにログインするためにユーザ名とパスワードを入力すると、LDAP クライアントはそのユーザ情報を LDAP サーバに送信して認証処理を依頼します。LDAP サーバはユーザ情報を LDAP ディレクトリ内のエントリと照合し、その結果を LDAP クライアントに返信します。一致するエントリが存在し、パスワードが正しければ、認証に成功し、ユーザはログインすることができます。一致するエントリが存在しないと、認証に失敗し、ユーザはログインできません。

Tru64 UNIX システムは、LDAP クライアントおよび LDAP サーバとして構成できます。ユーザ認証のために Tru64 UNIX を LDAP クライアントとして構成する方法については、付録 D を参照してください。

### 1.4.3 セキュア・シェル・アプリケーション

セキュア・シェルは、一連のネットワーク・コマンドを提供するクライアント/サーバ型アプリケーションです。セキュア・シェル・コマンドを使ってデータをやり取りすることにより、リモート認証サービスが提供されます。セキュア・シェル・コマンドは、従来の安全性の低いネットワーク・コマンドに追加して、またはそれらに代えて使用できます (表 1-2)。

表 1-2: セキュア・シェル・コマンド

作業	従来のコマンド	セキュア・シェル・コマンド
リモート・システム上でのコマンド実行	rsh	ssh
リモート・システムへのログイン	rlogin または telnet	ssh
システム間のファイル転送	rcp または ftp	scp または sftp

必要に応じて、`rcp`、`rlogin`、`rsh` の各コマンドおよび `rcmd()` 関数の使用時に自動的にセキュア・シェルが使われるように構成することもできます。

特に指定がなければ、オペレーティング・システム・ソフトウェアを Tru64 UNIX Version 5.1B 以上にアップグレードしたとき、またはこれらのバージョンをインストールしたときにセキュア・シェル・ソフトウェアがインストールされます。セキュア・シェル・ソフトウェアの設定と使用の詳細については、付録 B を参照してください。

#### 1.4.4 SSO/Kerberos メカニズム

Single Sign On (SSO) ソフトウェアは、`ftp`、`rcp`、`rlogin`、`rsh`、`telnet` の各ネットワーク・コマンドを使ってデータをやり取りする場合に、Kerberos を使ったりリモート認証サービスを提供するオプションのクライアント/サーバ型ソフトウェアです。

SSO ソフトウェアのインストール、設定、および使用の詳細については、付録 C を参照してください。

#### 1.4.5 Advanced Server for UNIX (ASU)

Advanced Server for UNIX (ASU) ソフトウェアは、認証要求を Windows NT Server Version 4.0 に転送するように Tru64 UNIX オペレーティング・システム・ソフトウェアを構成するためのメカニズムを提供するオプションのクライアント/サーバ型ソフトウェアです。

Tru64 UNIX システムに代わって、Windows NT Server Version 4.0 がユーザ・アカウント情報を使ってユーザを認証します。この機能は、Windows NT Server Version 4.0 上にユーザ・アカウント情報があり、Tru64 UNIX システム上にユーザ・アカウント・データベースを作成したくない場合に便利です。

Windows NT Version 4.0 の認証に関する設定の詳細については、ASU の『*Advanced Server for UNIX* インストレーション/管理ガイド』を参照してください。

#### 1.4.6 IPsec プロトコル

IPsec プロトコルは、TCP/IP トランスポート・プロトコルを使ってやり取りされるデータに IP 層での認証サービスを提供するオプションのセキュリティ・フレームワークです。

IPsec を有効にして IP 層を保護し、使用する接続を構成することにより、ネットワークを保護することができます。アプリケーション、コマンド、ユーティリティは、変更せずにそのまま IPsec を使用できます。

IPsec の有効化と構成の詳細については、『ネットワーク管理ガイド』を参照してください。

#### 1.4.7 SSL , CDSA , および GSS の API

SSL (Secure Socket Layer) API , CDSA (Common Data Security Architecture) API , および GSS (Generic Security Service) API は、これらの API を使用する分散アプリケーションに対してリモート認証サービスを提供します。これらの API を使用するアプリケーションは、基盤となるセキュリティ・メカニズムを意識することなくリモート認証サービスを要求できます。これらのセキュリティ API は、広範囲のセキュリティ・メカニズムをサポートします。

- CDSA は、アプリケーションが特定のサービスに関して使用するリモート認証サービス・モジュールをソフトウェアやハードウェアのベンダが作成するアーキテクチャです。アプリケーションは、CSSM (Common Security Services Manager) API を使って CDSA 内のモジュールにリモート認証サービスを要求します。
- SSL API は、主に Web アプリケーション開発者が複数のソケット間でリモート認証サービスを提供するために使用します。
- GSS API は、ソケット方式のプロトコルに基づいてネットワーク内でアプリケーションがリモート認証サービスを提供できるようにします。これにより、クライアントは必要な認証メカニズムがサーバにあるかどうかを判定した後、そのメカニズムを使って認証を行います。

詳細については、`cdsa(3)`、`ssl(3)`、および『セキュリティ・プログラミング・ガイド』を参照してください。

## 1.5 セキュリティ統合アーキテクチャの概要

Tru64 UNIX を構成して、1 つまたは複数のセキュリティ方式を使用するようにできます。Tru64 UNIX では、セキュリティ方式の適用順序がセキュリティ統合アーキテクチャ (SIA) によって管理されています。各セキュリティ方式によってセキュリティ・メカニズムが提供され、SIA に追加されます。セキュリティ・メカニズムは、認証の方法を決定するための独自のルールと、エンティティを認証する前にそのメカニズムが達成すべき信頼性レベルを定義したものです。

SIA には 2 つのインタフェース・レイヤがあります。1 つはアプリケーション、コマンド、およびユーティリティを作るプログラマ向けのメカニズム非依存インタフェースで、もう 1 つはシステム・セキュリティのプロバイダ用のメカニズム依存インタフェースです。セキュリティを扱うアプリケーション、コマンド、ユーティリティは、セキュリティ・サービスを提供するために SIA のメカニズム非依存インタフェースを呼び出します。メカニズム非依存インタフェースは、構成された各 SIA メカニズム・レイヤの対応するルーチンを呼び出して、要求されたアクセスが許可されるかどうかを判定します。

システム・セキュリティのプロバイダは、あらかじめ定義された一連のエントリ・ポイントを持つ共有ライブラリとしてセキュリティ・メカニズムを提供します。これにより、セキュリティ・ニーズの変化に合わせて、アプリケーション、コマンド、ユーティリティに変更を加えずに新しい SIA メカニズムを追加できます。

Tru64 UNIX では、次のセキュリティ・メカニズムが提供されます。

- `libc.so` ライブラリの基本 (BSD) セキュリティ・メカニズム (標準)。
- `/usr/shlib/libsecurity.so` ライブラリのエンハンスド・セキュリティ・メカニズム。これは、OSFC2SEC サブセットに格納されているオプションのメカニズムです。
- `/usr/shlib/libisialdap.so` ライブラリの LDAP メカニズム。これは、OSFLDAPAUTH サブセットと OSFSSOW2K サブセットに格納されているオプションのメカニズムです。

- /usr/shlib/libcsfk5siad.so ライブラリの Kerberos ベースのメカニズム。これは、OSFSSOW2K サブセットに格納されているオプションのメカニズムです。

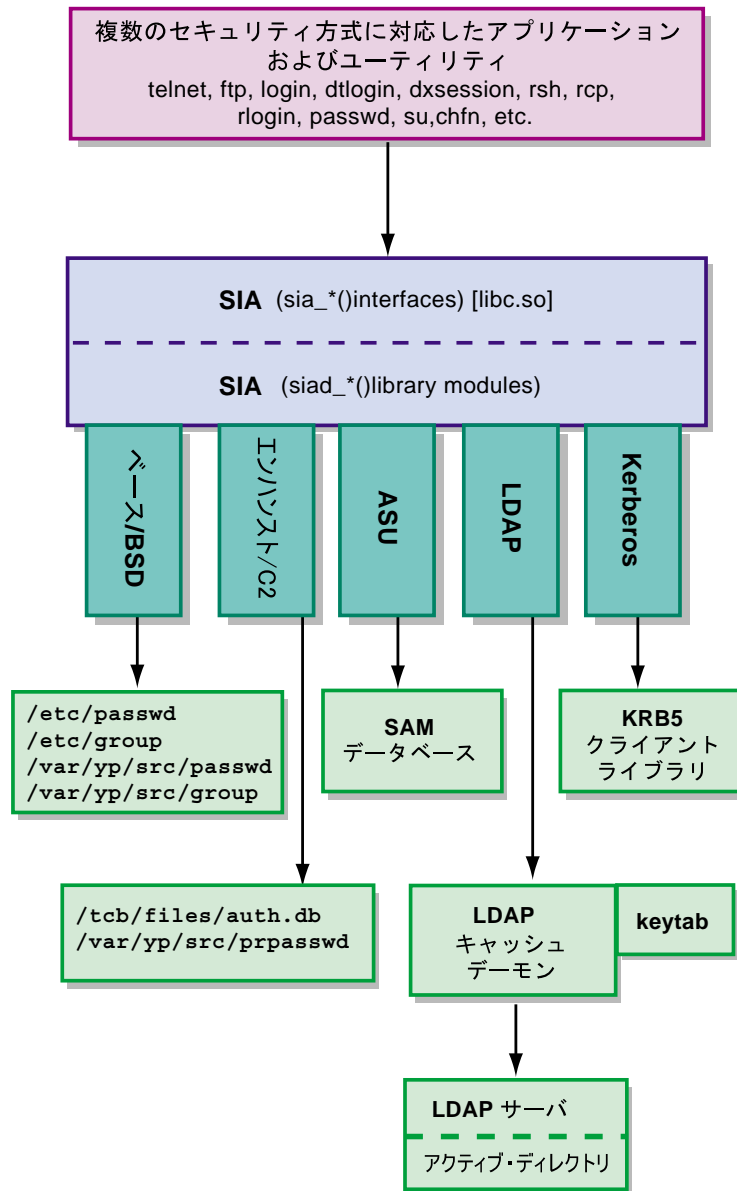
SIA メカニズムでは、次の基本領域を扱う一連の定義済みインタフェースを使ってセキュリティ・ポリシが実装されます。

- セッション制御
- 認証 (パスワードと承認) データの取得
- パスワード、finger 情報、シェル・データの変更

SIA メカニズムではすべてのセキュリティ機能を提供する必要はありませんが、データベースの変更機能を提供する場合はセッション機能も提供する必要があります。

SIA フレームワークにより、Tru64 UNIX システムや TruCluster Server 環境で複数のメカニズムが共存できる構造が提供されます。たとえば、ユーザが Kerberos パスワードを使ってログインしようとしたが、Kerberos サーバが稼動していないために認証に失敗した場合、同じログイン処理の中でローカルの /etc/passwd ファイルを使った省略時の BSD メカニズムによるユーザ認証を自動的に実行することができます。図 1-1 に、SIA のアーキテクチャを示します。

図 1-1: セキュリティ統合アーキテクチャ



ZK-0685U-AI

### 1.5.1 SIA とユーザ変更ルーチン

ユーザが変更ルーチンのコマンド (passwd, chfn, chsh などのコマンド) を入力すると、メカニズム依存ルーチンが呼び出され、実行しようとしてい

る変更がセキュリティ・メカニズムによってサポートされているかどうか、および変更の対象となるユーザがセキュリティ・メカニズムによって認識されているかどうか判定されます。

`chfn` コマンドや `chsh` を使って `finger` 情報やシェル情報を変更する場合、そのユーザが複数のセキュリティ・メカニズムによって認識されていると、変更を適用するセキュリティ・メカニズムをユーザに選択してもらうためのメニューが表示されます。

`passwd` コマンドを使ってパスワード情報を変更する場合は、メカニズム非依存インタフェースによって該当するユーザのパスワード変更をサポートするセキュリティ・メカニズムが識別されます。複数のセキュリティ・メカニズムが識別された場合は、パスワードの変更を適用するメカニズムをユーザに選択してもらうためのメニューが表示されます。1 つまたは全部を選択できます。

変更ルーチンの詳細については、『システム管理ガイド』を参照してください。

## 1.5.2 SIA の構成

SIA を設定すると、1 つまたは複数のセキュリティ・メカニズムを使用できるようになります。一般的なセキュリティ・メカニズムの組み合わせには、ローカル・システムのセキュリティのみを扱うローカル・メカニズムといくつかのシステムにまたがるセキュリティ事項を扱うリモート (分散) セキュリティ・メカニズムが含まれています。SIA の階層構造により、これら 2 つのメカニズムは共存できるようになっています。メカニズム依存ルーチンは、ユーザがセキュリティ・メカニズムによって認識されているかどうかに基づいて認証処理します。メカニズムの統合は、たとえばログインやセッションの処理に有効です。SIA の階層構造により、セッション処理時に複数のセキュリティ・メカニズム間でコンテキストが受け渡されます。このコンテキストには、収集された名前、パスフレーズ、およびセッション処理の現在の状態が含まれています。

各セキュリティ・メカニズムは、以前に実行されたメカニズムの認証プロセスを信頼することができます。これにより、認証の保証が可能になります。この場合、分散メカニズムがユーザの認証に成功すると、ローカル・メカニズムはその認証を信頼して受け入れ、セッション処理を継続することができます。また、SIA ではローカル・メカニズムが保証を受け入れない

ようにすることもできます。この場合、以前の認証結果に関係なく、ローカル・メカニズムで独自の認証処理が実行されます。その結果、ユーザに対してパスワードの入力要求が複数回行われることがあります。SIA ではメカニズムの順序は任意ですが、保証を受け入れるメカニズムは保証を受け入れないメカニズムの後に構成してください。Tru64 UNIX によって提供されるメカニズムは、保証を受け入れます。

/etc/sia/matrix.conf ファイルには、SIA メカニズムごとのエントリを作成する必要があります。SIA メカニズムが呼び出される順序は、matrix.conf ファイル内のエントリの並び順によって決まります。matrix.conf ファイル内の SIA メカニズムのエントリは、/usr/sbin/siacfg ユーティリティを使って管理します。

matrix.conf ファイルには、事前に定義されたメカニズム依存インタフェース `siad_*()` が 1 行ごとに指定されています。各行には、一意のメカニズム識別子 (または `mech_type`) とそのメカニズムの共有ライブラリへのパスで構成される属性が含まれています。次に matrix.conf ファイルのエントリの例を示します。

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

`siad_init` はルーチン名、`BSD` は一意の識別子、`libc.so` は `siad_init` ルーチンを持つ共有ライブラリ名です。`libc.so` ライブラリには省略時のメカニズムが含まれていて、このライブラリは既に開かれているので、このライブラリへのパスを指定する必要はありません。複数のメカニズムを構成する場合は、次のように 1 つのエントリに複数の属性を指定します。

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

この場合は、最初に `libsecurity.so` ライブラリ内の `siad_init` ルーチンが呼び出され、次に `libc.so` ライブラリのルーチンが呼び出されます。

`sia_*()` インタフェースは、この属性セットを使って各メカニズムを左から右へ順番に呼び出します。特定のカテゴリのインタフェースに対するサポートがメカニズムによって提供されない場合は、省略時の `BSD` メカニズムが使われます。

---

#### 注意

Tru64 UNIX Version 5.0 以前は、`siacfg` コマンドが存在しませんでした。また、`matrix.conf` ファイルを編集できないようにし、依存するメカニズムのプロバイダが



ら各メカニズムの属性を含む構成ファイルが提供されるようにするため、`/etc/sia/matrix.conf` ファイルは別のファイル (`/etc/sia/bsd_matrix.conf` ファイルや `/etc/sia/OSFC2_matrix.conf` ファイルなど) へのリンクになっていました。

---

Tru64 UNIX オペレーティング・システムには、`libc.so` ライブラリに含まれる BSD メカニズムのルーチンを使用する省略時の `matrix.conf` ファイルが用意されています。例 1-1 に省略時の `matrix.conf` ファイルを示します。

#### 例 1-1: 省略時の `/etc/sia/matrix.conf` ファイル

---

```
# 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)
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 メカニズムの初期化

SIA メカニズムは、システムのブート時に `/usr/sbin/siainit` ユーティリティによって初期化されます。`siainit` ユーティリティは、`/etc/sia/matrix.conf` ファイルを読み込み、`siad_init()` エントリ・

ポイントに設定された各メカニズムを呼び出してブート時の初期化処理を実行します。

`siad_init()` の呼び出しによってエラーが返されると、システムは強制的にシングル・ユーザ・モードに移行し、コンソールに "SIA Initialization Failure" というメッセージが表示されます。このため、`siad_init()` の呼び出しによってエラーが返るのは、セキュリティ上の問題が発生した場合、または root ユーザにログインが許可されない場合に限られます。`siad_init()` ルーチンの実行に成功すると、`/etc/sia/siainitgood` ファイルが (存在しない場合は) 作成されます。実行に失敗すると、`/etc/sia/siainitgood` ファイルは削除されます。

TruCluster 環境では、`siainitgood` ファイルが削除されると、問題が解決されるまで各ノードで実行されるプロセスが認証要求や承認要求の処理に失敗し始めます。引き続き SIA インタフェースを使用した場合は、構成されているメカニズムごとに `matrix.conf` のエントリ・ポイントを記述する内部構造を設定する前に、`siainitgood` ファイルの存在確認が行われます。

### 1.5.2.2 SIA メカニズムの追加

各 SIA メカニズムのプロバイダは、`siacfg` コマンドを使って、該当するメカニズムをサポートするように `matrix.conf` ファイルを更新します。

SIA にメカニズムを追加するときは、`siacfg` コマンドにメカニズムの一意の名前とメカニズムの共有ライブラリの場所を指定する必要があります。`siacfg` コマンドは指定されたライブラリを開き、すべての関数カテゴリを検索し、必要な関数 (ルーチン) がすべてライブラリにあることを確認します。必要なルーチンがすべて見つかったら、各ルーチンのエントリを `matrix.conf` ファイルに追加します。

SIA メカニズムを手作業で追加するときは、以下の手順に従います。

1. インストール手順に従ってメカニズムをインストールします。
2. `/etc/sia` ディレクトリに移動します。
3. 次のようにして SIA メカニズムを追加します。

```
# /sbin/siacfg -a SIA_mechanism_name library_path
```
4. 次のようにしてシステムをリブートします。

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

### 1.5.2.3 SIA メカニズムの削除

SIA メカニズムを削除するときは、以下の手順に従います。

1. 次のようにしてシステムをシングル・ユーザ・モードにします。

```
# /usr/sbin/shutdown now
```

2. 次のようにして SIA メカニズムを削除します。

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

siacfg コマンドは、*SIA\_mechanism\_name* に指定された名前を持つルーチン・エントリを `matrix.conf` ファイルから削除します。

3. 次のようにしてシステムをリブートします。

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

### 1.5.3 SIA のロギング

SIA は、呼び出し側のコマンドやユーティリティに返される成功または失敗の応答の受け渡しのみをサポートします。このため、どの SIA メカニズムでエラーが発生したかを判定するのがむずかしい場合があります。そのため、SIA には、次の種類のログ・エントリについてタイムスタンプ付きのエントリを `/var/adm/sialog` ファイルに生成する汎用のロギング機能が用意されています。

- EVENT。通常は SIA 内の処理に成功したことを表します。
- ERROR。通常は SIA 内の処理に失敗したことを表します。
- ALERT。SIA インタフェース内のセキュリティ構成やセキュリティ上のリスクを表します。

次のようにして `/var/adm/sialog` ファイルを作成する必要があります。

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

例 1-2 に `/var/adm/sialog` ファイルの例を示します。

#### 例 1-2: `/var/adm/sialog` ファイルの例

---

```
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
```

### 例 1-2: /var/adm/sialog ファイルの例 (続き)

---

```
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
```

---

---

## システム・リソースの保護

ユーザの識別と認証が終わると、Tru64 UNIX は任意アクセス制御 (DAC: Discretionary Access Control) 機能を使ってシステム・リソース (ディレクトリ、ファイル、デバイス、プロセスなど) へのアクセスを管理します。Tru64 UNIX では、Tru64 UNIX のアクセス許可とアクセス制御リスト (ACL) によって DAC を実現しています。

この章には、次の情報が含まれています。

- アクセス制御の概要
- Tru64 UNIX のアクセス許可
- アクセス制御リスト (ACL)

### 2.1 アクセス制御の概要

Tru64 UNIX オペレーティング・システム・ソフトウェアは、Tru64 UNIX のアクセス許可と ACL を使ってファイルやディレクトリへのアクセスを制御します。所有ユーザ、所有ユーザ・グループ、およびその他すべてのユーザに対してファイルやディレクトリの読み取り、書き込み、および実行アクセス許可を設定する場合は、Tru64 UNIX のアクセス許可と ACL を使用します。また、特定のユーザやグループに対してファイルやディレクトリの読み取り、書き込み、および実行アクセス許可を設定する場合は、ACL を使用します。

プロセスがファイルやディレクトリへのアクセスを要求すると、以下の手順で Tru64 UNIX のアクセス許可と ACL がチェックされます。

- 要求しているプロセスに root ユーザ特権がある場合は、ファイルやディレクトリへのアクセスが許可されます。ACL や Tru64 UNIX のアクセス許可はチェックされません。
- ACL が無効になっている場合、または該当するファイルやディレクトリに対応する ACL がない場合は、Tru64 UNIX のアクセス許可がチェックされます。

- ACL が有効になっていて、該当するファイルやディレクトリに対応する ACL がある場合は、サーバが要求されたリソースへのパスに含まれる各オブジェクトの ACL を収集して、アクセスの可否を決定します。アクセスを禁止する ACL が 1 つでもパスに含まれている場合は、アクセスが拒否されます。

## 2.2 Tru64 UNIX のアクセス許可

ファイルまたはディレクトリを作成すると、省略時の設定ではオペレーティング・システムまたは実行しているプログラムが Tru64 UNIX の読み取り、書き込み、および実行の各アクセス許可を次の 3 種類のユーザに割り当てます。

- u (ユーザ/所有者)
- g (グループ)
- o (他のすべてのユーザ/world と呼ばれる)

表 2-1 に、ファイルおよびディレクトリに適用される Tru64 UNIX のアクセス許可コードを示します。

表 2-1: Tru64 UNIX のアクセス許可コード

アクセス許可コード	ファイル	ディレクトリ
r (読み取り)	内容の表示やプリントができます。	内容を読むことはできるが検索はできません。通常 r と x は一緒に使用されます。
w (書き込み)	内容の変更や削除ができます。	エントリの追加や削除ができます。
x (実行)	ファイルをプログラムとして実行できます。	ディレクトリを検索できます。

### 2.2.1 ファイルとディレクトリに対する Tru64 UNIX のアクセス許可の表示

ファイルに対する Tru64 UNIX のアクセス許可を表示するには、次のコマンドを入力します。

```
$ ls -l filename
```

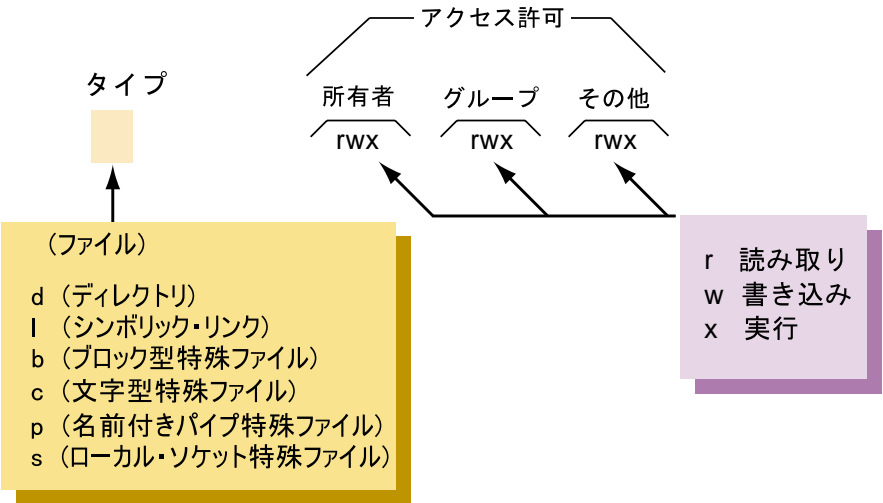
*filename* エントリは複数個指定できます。

カレント・ディレクトリ内のファイルとサブディレクトリに対する Tru64 UNIX のアクセス許可を表示するには、次のように `ls -l` コマンドを入力します。

```
$ 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 $
```

各行の先頭フィールドには、アクセス許可コードが表示されます。アクセス許可コードは、エントリのタイプとユーザ、グループ、および world に対するアクセス許可セットを示しています。先頭のダッシュ (-) は、エントリがファイルであることを示します。2 番目以降のダッシュは、該当するクラスユーザがその操作に関するアクセス許可を持っていないことを示します。  
図 2-1 は、アクセス許可フィールドについて説明した図です。

図 2-1: ファイルに対する Tru64 UNIX のアクセス許可フィールド



ZK-0536U-AI

注意

`ls` コマンドで表示したファイルの所有者、グループ、およびその他すべてのユーザに対するアクセス許可によってプロセスがファイルにアクセスできることが示されていても、そのファイ

ルの ACL がプロセスに対してアクセスを許可しない場合もあります。これは、そのプロセスがファイルのグループと同じ実効グループを持っている場合についても同じです。ACL の詳細については、2.3 節を参照してください。

## 2.2.2 ファイルとディレクトリに対するアクセス許可の設定 (chmod)

ユーザが自分のファイルやディレクトリに対するアクセス許可を設定または変更するときは、`chmod` (change mode) コマンドを使用します。ファイルやディレクトリのアクセス許可を設定または変更できるのは、そのファイルまたはディレクトリの所有者と `root` ユーザだけです。

通常、ユーザは自分自身にファイルの読み取り、書き込み、および実行のアクセスを許可します。自分のグループのメンバにはファイルの読み取りアクセスと、仕事の性質やグループの構成に応じて変更や実行のアクセスを許可します。その他すべてのユーザについては、ファイルへのあらゆる種類のアクセスを禁止します。

ファイルやディレクトリにどのようなアクセス許可を設定しても、`root` ユーザがいつでもそれらを変更できるという点を理解しておくことが重要です。たとえば、ユーザが `chmod` コマンドを使って自分だけが `report20` ファイルにアクセスできるように指定しても、依然として `root` ユーザはこのファイルにアクセスできます。

`chmod` コマンドによってアクセス許可セットを指定する方法には、次の 2 種類があります。

- 文字と演算記号を使ってアクセス許可を指定する方法
- 8 進数を使ってアクセス許可を指定する方法

### 2.2.2.1 文字と演算記号を使ってアクセス許可を指定する方法

文字と演算記号を使ってファイルやディレクトリのアクセス許可を変更できます。

以下は、文字と演算記号を使う場合の `chmod` コマンドの形式です。

**chmod** *userclass-operation-permission filename*

*userclass-operation-permission* エントリは、実際には次のように 3 つのコードを表しています。



- *userclass* には、次のいずれかの文字を指定します。
  - u ユーザ (所有者) を示します。
  - g グループを示します。
  - o 他のすべてのユーザ (所有者とグループを除く) を示します。
  - a すべてのユーザ (ユーザ、グループ、および他のすべてのユーザ) を示します。
- *operation* には、次のいずれかの記号を指定します。
  - + アクセス許可を追加します。
  - アクセス許可を削除します。
  - = それまでの設定に関係なくアクセス許可を割り当てます。
- *permission* には、次のいずれかの文字を指定します。
  - r。読み取りを示します。
  - s。ユーザまたはグループの ID を設定します。このアクセス許可は、ファイル実行中の実効ユーザ ID またはグループ ID をそのファイルの所有者またはグループ所有者の ID に設定します。このアクセス許可は、u または g の *userclass* エントリとともに設定して、通常は他のユーザがアクセスできないファイルに一時的または限定的なアクセスを許可する場合に使用します。ls -l コマンドを使ってアクセス許可を表示すると、ユーザまたはグループの実行許可の位置に s が表示され、設定したユーザ ID または設定したグループ ID のアクセス許可でファイルが実行されることが示されます。
  - w 書き込みを示します。
  - x 実行を示します。

*filename* エントリには、アクセス許可を変更するファイルの名前を指定します (複数指定も可能)。パターン・マッチング文字を使ってファイルを指定することもできます。詳細については、2.2.2.1.3 項を参照してください。

#### 2.2.2.1.1 ファイルのアクセス許可の変更

*file1* という名前のファイルに対するアクセス許可を変更するときは、以下の手順に従います。

1. *file1* に対するアクセス許可を表示します。

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

所有者 (larry) は読み取り/書き込みのアクセス許可を持ち、グループとその他のユーザは読み取りアクセス許可だけを持っています。

2. グループ (g) とその他のユーザ (o) の両方のアクセス許可を拡張するには、既存の読み取り (r) アクセスに加えて書き込みアクセス (+w) を file1 に許可します。

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

3. ファイルの新しいアクセス許可を表示します。

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

グループと他のすべてのシステム・ユーザが file1 に対して読み取りと書き込み (rw) のアクセス許可を持っています。

#### 2.2.2.1.2 ディレクトリのアクセス許可の変更

ディレクトリのアクセス許可の変更手順は、ファイルのアクセス許可を変更する場合と同じです。ただし、ディレクトリについての情報を表示するときは、ls -ld コマンドを入力します。

project という名前のディレクトリに対するアクセス許可を変更するときには、以下の手順に従います。

1. project に対するアクセス許可を表示します。

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

2. グループ (g) のアクセス許可を拡張するには、既存の読み取りと実行 (r-x) のアクセスに加えて書き込みアクセス (+w) を project に許可します。

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

3. 新しいディレクトリのアクセス許可を表示します。

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

グループが project ディレクトリに対して読み取り、書き込み、および実行 (rwx) のアクセス許可を持っています。

### 2.2.2.1.3 パターン・マッチング文字の使用

ディレクトリ内のすべてのエントリに対して同じ変更を行いたいときは、`chmod` コマンドでパターンマッチング文字のアスタリスク (\*) を使用できます。

カレント・ディレクトリ内のすべてのファイル (\*) に対する実行 (x) アクセスをグループ (g) に許可するには、次のコマンドを入力します。

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

カレント・ディレクトリ内のすべてのファイルに対する新しいグループのアクセス許可を表示するには、次のコマンドを入力します。

```
$ 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 絶対アクセス許可の設定

絶対アクセス許可 (=) の割り当ては、ファイル (複数も可) に対するすべてのアクセス許可をそれまでの設定状態に関係なくリセットします。

`file3` というファイルに対するすべてのアクセス許可をリセットするときには、以下の手順に従います。

1. ファイルのアクセス許可を表示します。

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

2. すべてのユーザ (a) に 3 つのアクセス許可 (rwx) をすべて割り当てます。

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

3. ファイルの新しいアクセス許可を表示します。

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

絶対アクセス許可の割り当ては、アクセス許可を削除する場合にも適用できます。

`file3` というファイルに対する実行アクセス許可を削除するときは、以下の手順に従います。

1. `file3` からすべてのグループ (a) に対する実行アクセス許可 (x) を削除します。

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

2. ファイルの新しいアクセス許可を表示します。

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

#### 2.2.2.2 8 進数を使ってアクセス許可を指定する方法

8 進数を使ってファイルやディレクトリのアクセス許可を変更できます。

以下は、8 進数を使う場合の `chmod` コマンドの形式です。

**`chmod`** *octalnumber filename*

*octalnumber* エントリには、所有者、グループ、その他のユーザに対するアクセス許可を指定する 3 桁の 8 進数を指定します。*filename* エントリには、アクセス許可を変更するファイルの名前 (またはスペースで区切ったファイル名のリスト) を指定します。パターン・マッチング文字を使ってファイルを指定することもできます。詳細については、2.2.2.1.3 項を参照してください。

8 進数は、次のようにアクセス許可の種類に対応しています。

```
4 = 読み取り (r)
2 = 書き込み (w)
1 = 実行 (x)
```

1 つのアクセス許可フィールドに対するアクセス許可をまとめて指定するには、次のように該当する 8 進数を合計します。

```
3 = -wx (2 + 1)
6 = rw- (4 + 2)
7 = rwx (4 + 2 + 1)
0 = --- (アクセス許可なし)
```

表 2-2 にアクセス許可を示す 8 進数の可能な組み合わせを示します。

表 2-2: アクセス許可を示す 8 進数の組み合わせ

8 進数	2 進数	アクセス許可	説明
0	000	なし	アクセスは一切許可されない
1	001	--x	実行
2	010	-w-	書き込み
3	011	-wx	書き込み/実行
4	100	r--	読み取り
5	101	r-x	読み取り/実行
6	110	rw-	読み取り/書き込み
7	111	rwX	読み取り/書き込み/実行

ファイルやディレクトリのアクセス許可コード全体は 3 桁の 8 進数で指定します。各桁の数値は、それぞれ owner、group、および others に対応します。表 2-3 に一般的なアクセス許可コードと各アクセス許可フィールドとの対応関係を示します。

表 2-3: 8 進数とアクセス許可フィールドの対応関係

8 進数	owner フィールド	group フィールド	others フィールド	完全なコード
777	rwX	rwX	rwX	rwXrwXrwX
755	rwX	r-x	r-x	rwXr-xr-x
700	rwX	---	---	rwX-----
666	rw-	rw-	rw-	rw-rw-rw-

例 2-1 では、ls -l コマンドで file3 というファイルのアクセス許可を表示しています。次に、chmod 754 コマンドによって、ユーザには 3 つのアクセス許可すべて (rwX) を、グループには読み取りと実行のアクセス許可 (rx) を、他のすべてのユーザには読み取りアクセス許可のみ (r) をそれぞれ割り当てています。

### 例 2-1: 8 進数によるアクセス許可の設定

```
$ 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 省略時のアクセス許可の設定

ファイルまたはディレクトリを作成すると、省略時のアクセス許可が設定されます。省略時のアクセス許可は、オペレーティング・システムか、そのファイルまたはディレクトリを作成したプログラムのどちらかによって設定されます。省略時のアクセス許可を設定することにより、ユーザはファイルやディレクトリを作成するたびにアクセス許可コードを明示的に指定する作業から解放されます。オペレーティング・システムは、省略時のアクセス許可値として実行可能ファイルに読み取り、書き込み、実行(777)を、それ以外のすべてのファイルに読み取りと書き込み(666)をそれぞれ割り当てます。

新規に作成するファイルおよびディレクトリに対する省略時のアクセス許可を変更するには、`umask` コマンドを使ってユーザ・マスクを設定します。ユーザ・マスクは、ファイルやディレクトリを作成したときの省略時のデフォルトのアクセス許可を決定する数値です。ファイルやディレクトリを作成すると、作成したプログラムが指定するアクセス許可の種類からユーザ・マスク値によって禁止された種類を差し引いた値がアクセス許可として設定されます。

以下は、`umask` コマンドの形式です。

**umask** *octal\_number*

*octal\_number* エントリには、省略時のアクセス許可(777 または 666)から差し引くアクセス許可を示す 3 桁の 8 進数を指定します。

ユーザ・マスクの設定は、8 進数を使ったアクセス許可の設定とほぼ同じです(2.2.2.2 項)。ファイルやディレクトリのアクセス許可コードを 3 桁の 8 進数で指定します。各桁は、アクセス許可の種類を示します。各桁(1 桁目, 2 桁目, 3 桁目)は、次の対応関係を持つ 3 ビットの値を表します。

- 1 桁目は、ファイルの `owner` に対応します。

- 2桁目は、ファイルの group に対応します。
- 3桁目は、others に対応します。

ユーザ・マスクを設定すると、実際には作成したプログラムの要求に関係なく許可しないアクセスの種類が指定されます。表 2-4 にユーザ・マスクのアクセス許可の組み合わせを示します。umask のアクセス許可値は、通常のアクセス許可コードに指定する値とは逆になっています。

表 2-4: ユーザ・マスクのアクセス許可の組み合わせ

各桁の値	許可されるアクセス	説明
0	rwx	読み取り/書き込み/実行
1	rw-	読み取り/書き込み
2	r-x	読み取り/実行
3	r-	読み取り
4	-wx	書き込み/実行
5	-w-	書き込み
6	-x	実行
7	---	アクセスは一切許可されない

たとえば、(実行可能ファイルに対して) 027 というユーザ・マスクを指定すると、次のような結果になります。

- owner には、ファイルを作成したプログラムによって要求されるアクセスがすべて許可されます。
- group には、書き込みアクセスが許可されません。
- others には、すべてのアクセスが許可されません。

省略時のユーザ・マスクは 022 です。これは、所有者にすべてのアクセスを許可し、ユーザ・グループと他のすべてのユーザには所有者のファイルへの書き込みを許可しないことを示します。

自分が所有するファイルやディレクトリに設定するユーザ・マスクの適切な値は、システム上の情報リソースがどの程度自由に共有されているかによって変わります。次のガイドラインを参考にしてください。

- 開放的なコンピュータ環境では、ユーザ・マスク値として 000 を指定し、ファイルやディレクトリへのアクセスに対して一切制限を設定しない方法が考えられます。これにより、プログラムがファイルを作成してアクセス許可コードを指定すると、プログラムが指定したアクセス許可コードに対してユーザ・マスクによる制限が一切かかりません。
- もう少し安全なコンピュータ環境では、ユーザ・マスクとして 066 を指定することにより、所有者には完全なアクセスを許可しながら、それ以外のユーザにはファイルの読み取りと書き込みを禁止する方法が考えられます。これにより、ファイルを作成すると、作成したプログラムが指定するアクセス許可の種類から所有者以外のユーザによる読み取りまたは書き込みアクセスを禁止するユーザ・マスクの制限値を差し引いた値がアクセス許可として設定されます。
- 安全が確保されたコンピュータ環境では、ユーザ・マスク値として 077 を指定し、所有者だけがファイルにアクセスできるようにする方法が考えられます。これにより、ファイルを作成すると、作成したプログラムが指定するアクセス許可の種類から所有者以外のユーザによる読み取り、書き込み、および実行を禁止するユーザ・マスクの制限値を差し引いた値がアクセス許可として設定されます。

ユーザ・マスクのしくみを理解するため、次のコマンドを入力した場合を考えます。

```
$ umask 037
```

このコマンドにより、実行可能ファイルに対して Tru64 UNIX のアクセス許可コード 740 (Tru64 UNIX の省略時のアクセス許可コード 777 から `umask` のコード 037 を差し引いた値) が設定され、次のような結果が得られます。

- 所有者には、すべてのアクセスが許可されます (7)。
- 所有者のグループのメンバには、読み取りアクセスだけが許可されます (4)。
- 他のユーザには、すべてのアクセスが許可されません (0)。

さらに、単純にファイルを作成した場合を考えます。省略時では、所有者にすべてのアクセスを許可し、他のすべてのユーザに読み取りと実行だけを許可する省略時のアクセス許可がテキスト・エディタによって常に割り当てられます。ここでは、既にユーザ・マスク 037 が設定されているので、ファイルのアクセス許可がさらに制限されます。この結果、所有者にはすべてのア



クセスが許可されますが、グループはファイルを実行できなくなり、他のすべてのユーザにはアクセスが一切許可されません。

#### 2.2.3.1 ユーザ・マスクの設定

ユーザ・マスクは、次のいずれかの方法で設定します。

- ユーザのログイン・スクリプトに `umask` コマンドを指定する方法。指定した値がユーザのログイン時に自動的に設定されるため、最も一般的で効率的なユーザ・マスクの指定方法です。
- ログイン・セッション中にシェル・プロンプトで `umask` コマンドを入力する方法。設定したユーザ・マスクの値は、そのログイン・セッションの間のみ有効です。

次の手順は、テキスト・エディタを使って作成するファイルのアクセス許可をユーザ・マスクによってどのように制限するかを詳しく示した例です。

1. ユーザ・マスクの現在の値を表示します。

```
$ umask
```

- ユーザ・マスクの値が `000` であれば、ファイル作成プログラムによって設定されるアクセス許可に制限が一切かかりません。手順 3 に進みます。
- ユーザ・マスクの値が設定されている場合は、それを書き留めます。手順 2 に進みます。

2. ユーザ・マスクの値を `000` に設定して、ファイル作成プログラムによって設定されるアクセス許可に制限が一切かからないようにします。ユーザ・マスクの値を元に戻す必要がある場合は、値をリセットする前に必ず現在の値を書き留めてください。

```
$ umask 000
```

3. ファイルを作成して保存し、テキスト・エディタを終了します。
4. 作成したファイルのアクセス許可を表示します。ここでは、すべてのユーザに読み取りおよび書き込みのアクセスが許可されるものとします。

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

5. ユーザ・マスクを `022` にリセットします。

```
$ umask 022
```

ユーザ・マスク 022 により、所有者にはすべてのアクセスを許可し、他のすべてのユーザには読み取りおよび実行のアクセスだけを許可するというアクセス許可制限が設定されます。

6. もう 1 つのファイルを作成して保存し、テキスト・エディタを終了します。

7. 作成したファイルのアクセス許可を表示します。

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

ユーザ・マスクの値 022 に従って、グループと他のすべてのユーザに対する書き込みアクセス許可が取り消されています。

8. ユーザ・マスクを元の値または別の値にリセットする (必要に応じて)。

## 2.3 アクセス制御リスト (ACL)

ACL は、ファイルおよびディレクトリに対する Tru64 UNIX のアクセス許可の機能拡張です。ACL を使用すると、ファイルやディレクトリのアクセス許可を特定のユーザやグループごとに設定することができます。ファイルやディレクトリに対する通常のアクセス許可とは別に、ACL にはアクセス許可が個別に指定されたユーザやグループのリストが含まれています。

1 つのファイルには、1 つの ACL (アクセス ACL) を対応付けることができます。1 つのディレクトリには、3 つの ACL (アクセス ACL、省略時アクセス ACL、および省略時ディレクトリ ACL) を対応付けることができます。アクセス ACL は、ファイルやディレクトリにアクセスできるユーザを決定するために使われます。各省略時 ACL は、ディレクトリ内に作成されるファイルやサブディレクトリに継承する ACL を決定するために使われます。

ACL はいつでも設定できますが、ACL のアクセス・チェックや継承は ACL を有効にしたときのみ行われます。システム上で ACL を無効にすると、ACL コマンドの実行時に警告メッセージが表示されます。また、ファイル・システムによっては ACL をサポートしないものがあります。ACL を設定できるのは、ACL をサポートするファイル・システム上のファイルとディレクトリだけです。

ACL は、ファイルやディレクトリのプロパティ・リストに保存された拡張ファイル属性です。システム上で ACL を無効にした場合も、プロパティ・リストをサポートするファイル・システム上の任意のファイルやディレクトリに ACL を対応付けることができます。ACL のアクセス・チェックや継承

は ACL を有効にしたときのみ行われます。詳細については、`acl(4)` および `proplist(4)` を参照してください。

次のファイル・システムは、プロパティ・リストをサポートしています。

- AdvFS (Advanced File System)  
AdvFS ファイル・システムでは、プロパティ・リストのエントリに 1560 バイトという上限があります。ACL はプロパティ・リストのエントリに保存されるため、これは 3 つのディレクトリ ACL の他に、62 の ACL エントリを指定できることを意味します。この上限を超えると、EINVAL エラーが返されます。
- クライアントとサーバの両方が Tru64 UNIX システムであるネットワーク・ファイル・システム (NFS)。NFS 上で ACL を有効にする方法の詳細については、2.3.1.1 項を参照してください。
- UNIX ファイル・システム (UFS)

UFS ACL の省略時の上限は 1548 バイトであり、これは AdvFS の 62 エントリ + 必須の 3 エントリという上限と同じです (AdvFS の上限にはヘッダも含まれています)。

UFS の ACL に対する上限は構成可能で、`sec` サブシステムの `ufs-sec-proplist-max-entry` という属性で定義されます。UFS のプロパティ・リストの要素サイズは構成可能で、`sec` サブシステムの `ufs-proplist-max-entry` という属性で定義されます。これらの属性は、`sysconfig` コマンドを使って動的に構成することができます。また、`sysconfigdb` コマンドを使って `sysconfigtab` ファイル内のこれらの属性を構成することもできます。

`ufs-proplist-max-entry` の値は、`ufs-sec-proplist-max-entry` 属性の値にプロパティ・リスト要素のヘッダを保持するスペースを足した値以上にする必要があります。sysconfig ユーティリティは、`ufs-proplist-max-entry` を自動調整してこの要件を達成します。`ufs-proplist-max-entry` の省略時の値は 8192 バイトです。詳細については、`sysconfig(8)`、`sysconfigdb(8)`、および `sysconfigtab(4)` を参照してください。

### 2.3.1 ACL の有効化と無効化

ACL サブシステムは基本システムの一部として出荷されますが、基本システムの構成要素は現行の動作に ACL を必要としないため、ACL に関するファイ

ルは組み込まれていません。ACL を使う場合や、レイヤード・プロダクトが ACL のサポートを必要とする場合は、ACL を有効にする必要があります。

- ACL の情報を表示するには、次のコマンドを入力します。

```
# sysconfig -q sec
```

- ACL を動的に有効にするには、次のコマンドを入力します。

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

システム起動時に ACL を有効にするには、`secconfig` ユーティリティを使用するか、または次の手順に従います。

1. 次のように、ACL モードを `enable` にするエントリを含む `stanza` ファイルを作成します。

```
sec:
    acl_mode = enable
```

2. `/etc/sysconfigtab` ファイルを更新します。

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

ACL を無効にするには、次のコマンドを入力します。

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

---

#### 注意

ACL を無効にすると、ACL でアクセスを禁止しているファイルやディレクトリに対してプロセスがアクセスできるようになります。このため、NFS を使ってファイルを共有するシステムでは、共通のセキュリティ・ドメインを持つことが特に重要です。

ACL を無効にしてもファイルやディレクトリに ACL を設定できますが、その場合はアクセス許可のチェックに ACL が使われず、新規に作成したファイルやディレクトリには省略時 ACL が継承されません。

---

#### 2.3.1.1 NFS 上での ACL の有効化

NFSv3 を使用していて、Tru64 UNIX NFS サーバで ACL が有効ならば、ACL のアクセス・チェックと NFS クライアントからの ACL の継承が適切

に行われます。ただし、NFS クライアントで ACL を設定できるようにするには、次の手順を実行する必要があります。

1. NFS サーバ上で、プロパティ・リストのデーモン (proplistd) を起動します。

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

2. NFS クライアント上で、次のいずれかの方法を使って proplist オプション付きでファイル・システムをマウントします。

- /etc/fstab ファイルのオプション・フィールドに proplist を追加します。

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

- mount コマンドに proplist オプションを追加します。

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

---

#### 注意

---

ACL を使う場合は、NFSv2 は使用できません。NFSv2 のプロトコル上の制限により、NFSv2 を使用すると ACL が必ずしも適用されない可能性があります。詳細については、`acl(4)` の `nfs-flatten-mode` を参照してください。

---

### 2.3.2 ACL の構造

ACL は、ACL エントリのリストで構成されます。各 ACL には、少なくとも次の 3 つのエントリがあります。

- 所有ユーザのエントリ
- 所有グループのエントリ
- その他すべてのユーザのエントリ

これらのエントリは、ファイルやディレクトリに対する Tru64 UNIX のアクセス許可に対応しています。コマンドによってこれらの ACL エントリを変更すると、Tru64 UNIX のアクセス許可も変更されます。

ACL の外部 (印字可能) 表現は、カンマ (,) または改行で区切られたエントリで構成されます。ACL エントリ内のフィールドはコロン (:) で区切られます。次の例は、典型的な ACL エントリです。

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

ACL エントリのキーワードと修飾子は以下のように定義されています。

user::	修飾子フィールドが空の user エントリは、ファイルを所有するユーザのアクセス許可を定義します。このエントリ (所有ユーザ・エントリと呼ばれる) は、UNIX のユーザに対するアクセス許可と常に同じです。ACL には、user:: エントリが 1 つだけ含まれていなければなりません。
user:	修飾子フィールドが空でない user エントリは、修飾子フィールドで指定されたユーザのアクセス許可を定義します。修飾子フィールドには、ユーザ名とユーザ識別情報 (UID) のどちらかが含まれていなければなりません。ACL には 0 または 1 個以上の user: エントリを含めることができます。
group::	修飾子フィールドが空の group エントリは、ファイルを所有するグループのメンバのアクセス許可を定義します。このエントリ (所有グループ・エントリと呼ばれる) は、UNIX のグループに対するアクセス許可と常に同じです。ACL には、group:: エントリが 1 つだけ含まれていなければなりません。
group:	修飾子フィールドが空でない group エントリは、修飾子フィールドで指定されたグループのメンバのアクセス許可を定義します。修飾子フィールドには、グループ名とグループ識別情報 (GID) のどちらかが含まれていなければなりません。ACL には 0 または 1 個以上の group: エントリを含めることができます。
other::	other エントリは、修飾子が空の場合のみ有効です。このエントリは、ACL の他のエントリのどれと

も一致しないユーザすべてのアクセス許可を定義します。このエントリは、UNIX の other に対するアクセス許可と常に同じです。ACL には、other:: エントリが 1 つだけ含まれていなければなりません。

アクセス許可フィールドの文字列は、ls コマンドで Tru64 UNIX のアクセス許可を表示したときの文字列と同じで、順序も同じです。r は読み取りアクセス、w は書き込みアクセス、x は実行または検索アクセスを示します。これらの文字の代わりにハイフン (-) が使われている場合は、そのアクセスが拒否されることを示します。

ACL の user、group、other の各エントリは、そのファイルまたはディレクトリに対する Tru64 UNIX のアクセス許可に対応しています。ACL が有効で、chmod コマンドを使用してファイルやディレクトリの Tru64 UNIX のアクセス許可を変更した場合、chmod は所有ユーザ・エントリ、所有グループ・エントリ、および other エントリ用のアクセス ACL を適切に変更します。

表 2-5 に一般的な ACL エントリを示します。

表 2-5: ACL エントリの例

エントリ	照合基準
group:acct:r--	acct グループのすべてのユーザに対し、読み取りを許可します。
user:joe:rw-	ユーザ joe に対し、読み取りと書き込みを許可します。
user::rwx	オブジェクトの所有ユーザ (ファイルの作成後に所有ユーザが変更された場合も含む) に対し、読み取り、書き込み、および実行を許可します。
group::r--	オブジェクトの所有グループ (ファイルの作成後に所有グループが変更された場合も含む) に対し、読み取りを許可します。
other::r--	所有ユーザ、所有グループ、および ACL エントリにリストされたユーザとグループを除くすべてのユーザおよびグループに対し、読み取りを許可します。

2.3.3 ACL を使ったアクセス・チェック

ファイルまたはディレクトリにアクセス ACL が設定されている場合は、そのファイルまたはディレクトリへのアクセスを許可する前に追加的なア

クセス・チェックが実行されます。具体的には、次のチェックが次の順序で実行されます。

1. プロセスにスーパーユーザ権限がある場合、ファイルやディレクトリへのアクセスが許可されます。アクセス ACL と Tru64 UNIX のアクセス許可はチェックされません。
2. ACL が無効になっている場合や、ACL は有効になっているがファイルやディレクトリに対応するアクセス ACL がない場合は、Tru64 UNIX のアクセス許可がチェックされます。
3. ファイルやディレクトリに対するアクセス ACL の追加エントリは、以下のようにチェックされます。
  - a. プロセスのユーザ ID (UID) がオブジェクトの所有者のものである場合、所有ユーザ (user::) のエントリに指定されたアクセスが許可されます。他の ACL エントリはチェックされません。これは、Tru64 UNIX の user に対するアクセス許可を使用するのと同じです。
  - b. プロセスの UID が user: エントリに指定された UID に一致するか、または user: エントリに指定されたユーザ名で解決できる場合、そのエントリに指定されたアクセスが許可されます。他の残りの ACL エントリはチェックされません。
  - c. プロセスのグループ ID (GID) がファイルの GID に一致するか、またはプロセスの補助グループのうち 1 つがファイルの GID に一致する場合、group:: エントリと (次のリスト項目で説明するように) 一致する他の group: エントリのアクセス許可を合わせたものがそのプロセスに対して許可されます。
  - d. プロセスの GID がいずれかの group: エントリの GID に一致するか、いずれかの group: エントリに指定されたグループ名で解決できる場合、またはプロセスのいずれかの補助グループの GID またはグループ名が ACL のいずれかの group: エントリに一致する場合は、一致するすべてのグループ・エントリの保護属性 (アクセス許可) を合わせたものがそのプロセスに対して許可されます。たとえば、グループ sales とグループ eng に属するユーザがいて、ファイルのアクセス ACL がグループ sales に読み取りアクセスを許可し、グループ eng に書き込みアクセスを許可する場合、そのユーザにはそのファイルに対する読み取りと書き込みのアクセスが許可されます。



- e. `other::` エントリで指定されたアクセスが許可されます。これは、Tru64 UNIX の `other` に対するアクセス許可を使用するのと同じです。

---

#### 注意

---

Tru64 UNIX のアクセス許可を持つファイルやディレクトリと 3 つの必須エントリ (`user::` , `group::` , および `other::`) だけを含むアクセス ACL を持つファイルやディレクトリは、区別できません。

---

### 2.3.4 ACL の継承

ファイルやディレクトリを作成すると、親ディレクトリから ACL が継承されます。親ディレクトリ内に作成したファイルやサブディレクトリに継承される ACL は、省略時 ACL によって次のように決定されます。

- 親ディレクトリに省略時 ACL がない場合
  - ディレクトリ内に作成される新しいファイルの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	なし

- ディレクトリ内に作成される新しいサブディレクトリの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	なし
省略時アクセス ACL	なし
省略時ディレクトリ ACL	なし

Tru64 UNIX のアクセス許可だけが使われます。

- 親ディレクトリに省略時アクセス ACL はあるが、省略時ディレクトリ ACL がない場合

- ディレクトリ内に作成される新しいファイルの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	親の省略時アクセス ACL

- ディレクトリ内に作成される新しいサブディレクトリの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	親の省略時アクセス ACL
省略時アクセス ACL	親の省略時アクセス ACL
省略時ディレクトリ ACL	なし

- 親ディレクトリに省略時アクセス ACL はないが、省略時ディレクトリ ACL がある場合

- ディレクトリ内に作成される新しいファイルの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	なし

- ディレクトリ内に作成される新しいサブディレクトリの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	親の省略時ディレクトリ ACL
省略時アクセス ACL	なし
省略時ディレクトリ ACL	親の省略時ディレクトリ ACL

- 親ディレクトリに省略時アクセス ACL と省略時ディレクトリ ACL の両方がある場合

- ディレクトリ内に作成される新しいファイルの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	親の省略時アクセス ACL

- ディレクトリ内に作成される新しいサブディレクトリの ACL は、次のように設定されます。

ACL の種類	ステータス
アクセス ACL	親の省略時ディレクトリ ACL
省略時アクセス ACL	親の省略時アクセス ACL
省略時ディレクトリ ACL	親の省略時ディレクトリ ACL

ディレクトリに省略時 ACL を設定しても、そのディレクトリ内の既存のファイルやサブディレクトリの ACL は変更されません。

#### 2.3.4.1 ACL の継承の例

ACL の継承の例を以下に示します。

- temp ディレクトリに省略時 ACL が無い状態で、次のコマンドを実行して temp ディレクトリに省略時アクセス ACL を設定したとします。

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

この場合、temp ディレクトリの中に作成されるすべてのファイルやディレクトリは、アクセス ACL として次の ACL を継承します。

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

- temp ディレクトリに省略時 ACL が無い状態で、次のコマンドを実行して temp ディレクトリに省略時ディレクトリ ACL を設定したとします。

```
% setacl -D -u user::rw-,group::r--,other::r--, \
user:jdoue:rxw temp
```

この場合、temp ディレクトリの中に作成されるすべてのディレクトリは、アクセス ACL および省略時ディレクトリ ACL として次の ACL を継承します。

```
#
# file: temp
# owner:smith
# group:system
#
user::rw-
user:jdoe:rw-
group::r--
other::r--
```

- temp ディレクトリに省略時 ACL がない状態で、次のコマンドを実行して temp ディレクトリに省略時アクセス ACL と省略時ディレクトリ ACL を設定したとします。

```
% setacl -D -u user::rw-,group::r--,other::r--, \
    user:jdoe:rw- temp

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

この場合、temp ディレクトリの中に作成されるすべてのディレクトリは、アクセス ACL および省略時ディレクトリ ACL として次の ACL を継承します。

```
#
# file: temp
# owner:smith
# group:system
#
user::rw-
user:jdoe:rw-
group::r--
other::r--
```

次の ACL は、省略時アクセス ACL として継承されます。

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

temp ディレクトリの中に作成されるすべてのファイルは、アクセス ACL として次の ACL を継承します。

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

### 2.3.5 ACL の管理

以下のコマンドを使って、ACL の表示や変更を行います。

dxsetacl	ファイルやディレクトリの ACL を作成、表示、および変更するためのグラフィック・インタフェースです。
getacl	ファイルやディレクトリの ACL を表示します。
setacl	ファイルやディレクトリの ACL を作成、変更、および削除します。

---

#### 注意

---

ファイルまたはディレクトリの ACL を作成、変更、削除するには、あらかじめそのファイルまたはディレクトリの所有者になる (またはスーパーユーザ特権を持つ) 必要があります。

---

#### 2.3.5.1 dxsetacl インタフェースの使用

dxsetacl グラフィカル・インタフェースを使って、ファイルやディレクトリの ACL を作成、表示、および変更することができます。dxsetacl インタフェースは、「アプリケーション・マネージャ」の「CDE デスクトップ・アプリケーション」の中にあります。また、コマンド行で次のコマンドを実行して開くこともできます。

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

詳細については、`dxsetacl(1)` と `dxsetacl` のオンライン・ヘルプを参照してください。

### 2.3.5.2 `getacl` コマンドの使用

ファイルやディレクトリの ACL を表示するには、`getacl` コマンドを使います。例 2-2 では、`getacl file.txt` コマンドによって `file.txt` というファイルの ACL を表示しています。

#### 例 2-2: ファイルの ACL の表示

---

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

---

`getacl` コマンドを使って、アクセス ACL が設定されていないファイルやディレクトリの ACL を表示すると、Tru64 UNIX のアクセス許可が ACL の形式で表示されます。

詳細については、`getacl(1)` を参照してください。

### 2.3.5.3 `setacl` コマンドの使用

`setacl` コマンドを使って、ファイルやディレクトリの ACL を作成、変更、または削除することができます。例 2-3 では、`getacl file.txt` コマンドによって `file.txt` というファイルの ACL を表示しています。続いて、`setacl` コマンドによって ACL を更新しています。

#### 例 2-3: ファイルに対する ACL の設定

---

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

### 例 2-3: ファイルに対する ACL の設定 (続き)

---

```
user:jdoue: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:jdoue:rw-
user:evan:rw-
group::rw-
other::r--
```

---

所有グループのエントリが更新され、書き込みアクセス許可が追加されています。evan というユーザ・エントリは既存のエントリと一致しないので、読み取りと書き込みのアクセス許可とともに追加されています。other エントリは変更されていません。

詳細については、`setacl(1)` を参照してください。

## 2.3.6 コマンドやアプリケーションと ACL の相互作用

ACL は、POSIX P1003.6 ドラフト 13 に基づく、POSIX および System V 互換の UNIX 拡張です。既存のすべてのコマンド、ユーティリティ、およびアプリケーション (特に POSIX 互換でないアプリケーション) が ACL を適切に使用または伝達するとは限りません。このため、コマンド、ユーティリティ、またはアプリケーションを使って ACL が設定されているファイル・システム・オブジェクト (ファイルまたはディレクトリ) をアクセスまたは操作した場合は、処理の完了後に ACL が削除または変更されていないことを確認する必要があります。

ファイルを変更するプログラムの多くは、以下の手順で処理を実行します。

- ファイルの新しいバージョンを一時的な名前で作成します。
- ファイルの既存のバージョンを削除します。

- 新しいバージョンの名前を一時的な名前から本当の名前に変更します。

変更しているファイルに ACL があっても、ファイルの一時的なバージョンの作成時にプログラムが ACL を複製しない場合、上記の手順ではファイルの ACL が削除されるか、または (もしあれば) 親ディレクトリの省略時アクセス ACL で置き換えられます。ACL を持つファイルに対してそのようなプログラムを使用した場合は、後で ACL を復元しなければなりません。また、この手順ではファイルからハード・リンクが削除されます。この方法でファイルを変更する一般的なコマンドには、以下のものがあります。

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

この問題を回避するには、元のファイルを一時ファイルへコピーし、一時ファイルに対して処理を行ってから、`-p` オプションを指定せずに `cp` を実行して、元のファイルにコピーし戻します。この手順によって元の ACL を維持できます。

ACL のあるファイルをコピーするときは、必ず `cp -p` コマンドを使って ACL やその他の拡張属性 (プロパティ・リスト) を正しくコピーしてください。

#### 2.3.6.1 `pax` コマンドと `tar` コマンド

`pax` コマンドと `tar` コマンドは、特に指定がなければアーカイブ作成時にファイルやディレクトリの ACL とその他の拡張属性をアーカイブします。ただし、`pax` コマンドや `tar` コマンドを使ってアーカイブからファイルおよびディレクトリを抽出するときは、特に指定されない限りアーカイブされたファイルやディレクトリの ACL がアーカイブから抽出されません。この場合、格納先のディレクトリに省略時 ACL が定義されていれば、アーカイブから抽出したファイルおよびディレクトリはその省略時 ACL を継承します (2.3.4 項)。

アーカイブから ACL やプロパティ・リストの情報を復元するには、アーカイブからファイルやディレクトリを抽出する際に `tar` コマンドには `-p` オプションを指定し、`pax` コマンドには `-p p` または `-p e` オプションを指定します。`pax` コマンドと `tar` コマンドは、ACL のユーザおよびグループの情報を UID および GID として保存します。したがって、`tar -p`、`pax -p p` または `pax -p e` コマンドを使って、アーカイブ元のシステムとユーザやグ

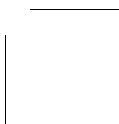
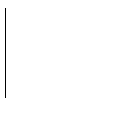


ループの情報を共有していないシステム上にアーカイブを復元すると、意図していないファイル・アクセスを許可する可能性があります。

現時点では、ACL および拡張属性 (プロパティ・リスト) についての正式な業界標準はありません。このため、`pax` コマンドと `tar` コマンドでプロパティ・リストと ACL をサポートする機能拡張は、Tru64 UNIX 独自のものです。他のベンダの `pax` および `tar` では、Tru64 UNIX 独自の機能拡張は無視されます。ただし、他のベンダのシステムとの相互運用性を確保するためには、複数のベンダに配布するアーカイブの作成時に `-v` オプションを指定して、ACL やその他の拡張属性がアーカイブされないようにしてください。

#### 2.3.6.2 アーカイブ・コマンド

アーカイブ・コマンドの `dump`、`rdump`、`restore`、`rrestore`、`vdump`、`rvdump`、`vrestore`、および `rvrestore` は、常に ACL を含むすべての拡張属性 (プロパティ・リスト) を保存および復元します。このため、省略時ディレクトリ ACL または省略時アクセス ACL を持つディレクトリにファイルを抽出すると、抽出されたファイルに対して予期しない ACL が作成されることがあります。ACL が有効になっているシステムでは、抽出の終了後に必ずすべての ACL を確認してください。



---

## システムの監査

Tru64 UNIX オペレーティング・システム・ソフトウェアには、監査サブシステムが含まれています。監査サブシステムは、各ファイルのオープン、ファイルの作成、ログイン、およびプリント・ジョブの送信など、システムで発生するイベントを記録します。各イベントは作成したユーザの監査 ID (AUID) が関連付けられるため、どのアクションも特定のユーザまでたどることができます。ユーザは、監査サブシステムと直接やり取りすることはありません。

この章では、以下について説明しています。

- 監査の概要
- 監査サブシステムの構成
- 監査サブシステムの管理
- 監査イベントの管理
- 監査レポートの生成および表示
- 従来の UNIX のログイン・ツール
- TruCluster での監査
- 監査レポートに対する対処

### 3.1 監査の概要

監査は、システム上で実行されたアクションを記録するための手段を提供します。ユーザのログイン時に、システムはユーザのプロセスと監査 ID (AUID) を対応付ける ID レコードを作成します。AUID は、どのようなプログラムを実行しているときでも、プロセスに関連付けられたままです。ユーザが実ユーザ ID または実効ユーザ ID を変更しても (su コマンドを使って、root または別のユーザになるなど)、システムは監査 ID に基づいて、どの認証ユーザが特定のアクションを行ったかを依然として認識できます。ただ

し、ユーザ ID の信頼性は、ユーザ ID の検証に使われる認証メカニズムに依存することを忘れないでください。

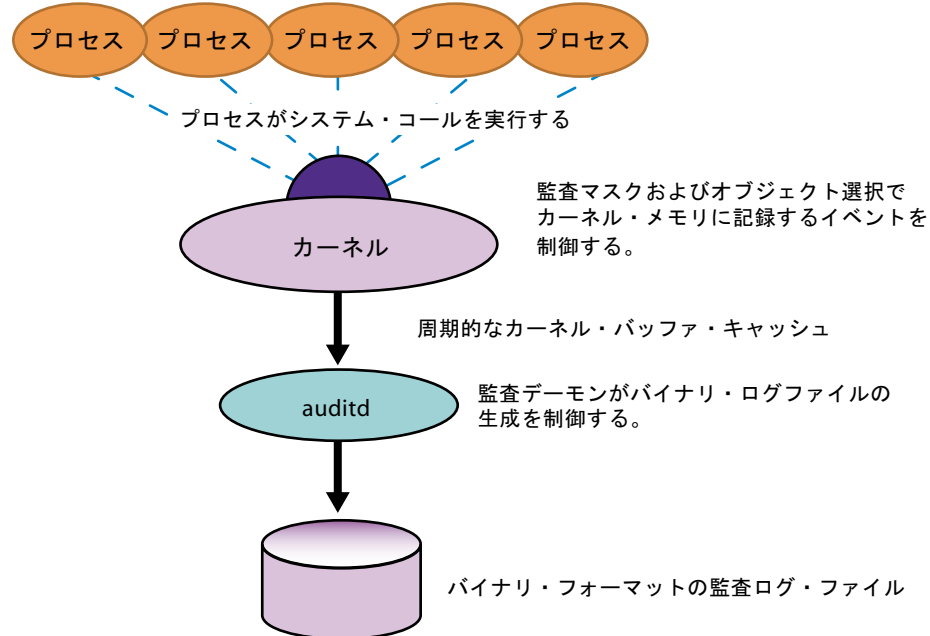
システムは、各ユーザの特性や承認された権限を記述する詳細な認証プロファイルを管理します (特定のユーザに対するログイン制限など)。

簡単に言うと、監査は次の内容から構成されます。

1. システムのイベントに伴って監査レコードが生成されます。この監査レコードには、イベント内容、発生日時、およびイベントの原因となったユーザの ID などの情報が含まれます。
2. 監査レコードは、他の監査レコードとともに監査ログ・ファイルと呼ばれるファイルに保存されます。
3. ユーティリティを使用して、監査ログ・ファイルに情報を表示し、その情報からレポートを作成します。

図 3-1 に監査サブシステムの概要を示します。

図 3-1: 監査サブシステム



ZK-1582U-AIJ

### 3-2 システムの監査

監査は、システム上のイベントを監視する強力なツールとして使用できます。監査を通じて、次のようなことが実現できます。

- ユーザがセキュリティに違反し侵害しないようにします。システムのイベントが監視されており、セキュリティ違反を特定の個人までたどれることをユーザが知ることで、セキュリティに違反する気持ちをなくさせます。
- 侵害行為やシステムの弱点を探っていると見られる動作を検出します。監査により、システム・セキュリティの侵害の試みの失敗が判明した場合、その後の試行で侵害に成功しないように対策を取ることができます。
- 侵入が発生した場合、損害を見積もり、システムを回復します。侵入後の監査証跡を詳しく分析すると、セキュリティ侵害の内容、およびシステムを元の状態に戻すために必要な手順を把握するのに役立ちます。また、今後同じような侵入が発生しないように対策を講じることもできます。
- アプリケーション・ソフトウェアの評価およびデバッグを行います。指定プロセスについてシステム・コールの終了状態および引数を監視する監査機能では、アプリケーション内部での動作状況を調べることができます。ソース・コードがなくても、アプリケーションがどのファイルにアクセスしようとしているか調べることもできます。

ユーザに対して、システム上で実行される監査の目的および性質を、一般的な言葉で説明することが大切です。ユーザのファイルおよびシステム・リソースへのアクセスを保護するツールとして、監査を肯定的に説明してください。これにより不平不満を緩和することができます。つまり、システムが常に監査されていることを公にすることで、ユーザが偵察されていると感じることが少なくなります。セキュリティ違反を犯す可能性のあるユーザには、動作が監視されていることを知ることで強力な抑止力となります。

### 3.1.1 監査ファイル

次のリストでは、監査サブシステムで使用するファイルについて説明します。  
`/sbin/init.d/audit`

監査サブシステムの起動およびシャットダウン・スクリプト。

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

省略時のログ・ファイル。Tru64 UNIX 監査サブシステムは指定された任意の場所の任意のファイルにイベントを記録できます。

`hostname` は監査ログを生成したシステムの名前です。

`nnn` は世代番号で、000 ~ 999 の番号です。

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

監査サブシステムからのステータス・メッセージ用の省略時のログ・ファイルです。

`/etc/rc.config.common`

監査サブシステムの現在のシステム省略時設定を含みます。

`/etc/sec/audit_events`

セキュリティに関係するシステム・イベントをすべて記載するリストを含みます。

このファイルは、システム上のどのイベントを監査するかを制御する、`auditmask` コマンドへの入力として使用されます。

`/etc/sec/site_events`

サイト固有の監査イベントを定義するファイル。これはアプリケーション固有の監査を監査サブシステムに統合するのに使用します。

`/etc/sec/event_aliases`

監査可能なイベント・セットを表す、エイリアス (別名) のリストを含むファイル。

`/etc/sec/auditmask_style`

プロファイル用の監査のスタイル・フラグを定義するファイル。

`/etc/sec/file_objects/*`

プロファイル用に監視するファイル名のリストを含むディレクトリ。

`/etc/sec/rc_audit_events`

監視する監査イベントのリストを含むファイル。このリストはシステムのスタートアップ時にロードされ、ファイルは、

`/etc/rc.config` または `/etc/rc.config.common` にある実行時構成変数 `AUDITMASK_FLAG` に指定されます。

`/etc/sec/fs_objects`

監査を選択 (または選択解除) するファイルのリスト。

`/etc/sec/auditd_loc`

現在の監査ログが使用不能または一杯になったときに、監査ログを保存する代替パスおよびホストのリスト。

`/etc/sec/auditd_clients`

ローカル・システムに保存するために、監査データを送信できるリモート・ホストのリスト。

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

クラスタ・システムでの監査に必要な CDSL。監査サブシステムに関連付けられるデバイスを指定します。

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

クラスタ・システムでの監査に必要な CDSL。監査サブシステムに関連付けられるデバイスを指定します。

### 3.1.2 監査ツール

次のインタフェースを使って、監査サブシステムを管理できます。

- コマンド行インタフェース
- ポイント&クリック操作やオンライン・ヘルプを備えたグラフィック・インタフェース。

#### 3.1.2.1 コマンド行インタフェース

次のコマンドは監査サブシステムのコマンドです。これらのコマンドを使うには、`root` ユーザである必要があります。コマンドの詳細については、関連するリファレンス・ページを参照してください。

`sysman auditconfig`

監査ログの格納場所、容量不足になったときに監査サブシステムが取る措置、システムのディスク領域を解放するために

	<p>ログを削除するまでに監査ログを取っておく期間 (指定によっては無期限) , ネットワークをまたいで監査を行うかどうかなどを含め , システムの監査環境を対話的に設定します。</p>
<code>auditmask</code>	<p>監査ログに含めるイベントを選択します。また監査ログに現在記録されているイベントのリストを表示します。</p> <p><code>auditmask</code> コマンドを使って行った監査サブシステムへの変更は一時的であり , システムをリブートするとリセットされます。</p>
<code>audgen(2)</code> と <code>audgen(8)</code>	<p>特権プログラムまたは選択したメッセージを含むスクリプトから監査イベントのレコードを生成します。</p>
<code>auditd</code>	<p>監査デーモンの起動 (監査の開始) , 監査ログ・ストレージの管理 , および監査デーモンの構成を行います。</p> <p><code>auditd</code> コマンドを使って行った監査サブシステムへの変更は一時的であり , システムをリブートするとリセットされます。</p>
<code>audit_tool</code>	<p>監査ログから情報を抜粋し , その情報を読みやすい形式で表示します。</p>
<code>audit_tool.ultrix</code>	<p>ULTRIX システム上で作成した監査ログから情報を抜粋し , その情報を読みやすい形式で表示します。</p>

### 3.1.2.2 グラフィック・インタフェース

監査サブシステムのグラフィック・インタフェースには , CDE のアプリケーション・マネージャを使って次のようにアクセスします。



「システム管理」 「日常管理」 「監査マネージャ」

グラフィック・インタフェースを使って、監査ログの管理パラメータを監査したり、設定したりするためのカーネルを構成することはできません。このようなカーネルを構成するには、`sysman auditconfig` コマンドを使う必要があります。

### 3.1.3 監査マスク

監査マスクは監査対象イベントを指定し、そのイベントが成功を示す場合、失敗を示す場合、またはその両方の場合を監査するのかを指定します。監査マスクには、次の 2 種類があります。

- システム監査マスク。これには、次のイベントが含まれます。
  - `/etc/sec/audit_events` ファイルで定義されたシステム・コール名
  - `audit.h` で定義されたトラステッド・イベント名
  - `/etc/sec/site_events` ファイルのサイト固有の名前
  - `/etc/sec/event_aliases` ファイルで定義されたエイリアス
- プロセス監査マスク。これは、特定のプロセスに適用されます。

特別なプロセス監査マスクにログイン・プロセス・マスクがあります。これは、エンハnst・セキュリティ・サブセットがインストールされ、アクティブになっている場合のみ使用できます。ログイン・プロセス・マスクはユーザ・ログイン・プロセスに対して設定されます。ユーザのログイン時、ログイン・プロセスにプロセス監査マスクが適用され、そのプロセスから派生するすべてのプロセスが同じ監査マスクを継承します。エンハnst・セキュリティの詳細については、付録 A を参照してください。

#### 3.1.3.1 システム・コール

セキュリティ関連のすべてのシステム・コールでは、監査データが生成されますが、セキュリティに関係しない一部のシステム・コールでは、監査データが生成されないことがあります。システム・コールで監査データが生成されない状況について、表 3-1 に示します。

表 3-1: 監査されない可能性があるシステム・コール

システム・コール	監査レコードが生成されない原因
<code>close()</code>	無効なファイル記述子が渡されたため、システム・コールの実行が失敗に終わりました。
<code>dup2()</code>	無効なファイル記述子が渡されたため、システム・コールの実行が失敗に終わりました。
<code>execv()</code> , <code>execve()</code> , <code>exec_with_loader()</code>	<code>namei()</code> 検索の失敗、スレッドの終了の失敗、ハンドラの呼び出しの強制終了による失敗。
<code>fcntl(*)</code>	無効なファイル記述子が渡されたため、システム・コールの実行が失敗に終わりました。
<code>ioctl(*)</code>	無効なファイル記述子が渡されたため、システム・コールの実行が失敗に終わりました。
<code>msfs_syscall()</code>	システム・コールのあらゆる失敗。
<code>priocntlse()</code>	無効なプロセスを指定した (ESRCH) か、このシステム・コールで他のプロセスが変更されていません。
<code>pro-plist_syscall()</code>	システム・コール引数の <code>copyin()</code> での失敗。
<code>reboot()</code>	正常リブートは監査されません。 <code>reboot()</code> システム・コールは、正常にリブートできたときには戻りません。
<code>security()</code>	<code>getluid</code> オプションについては監査レコードはありません。このオプションはセキュリティに関係がなく、監査を行っても、無用の監査レコードが大量に生成されます。
<code>swapctl()</code>	SC_ADD 形式のシステム・コールのみ監査されます。他の形式はセキュリティに関係しません。
<code>uadmin()</code>	障害の発生した A_REBOOT または A_SHUTDOWN のみ監査されます。その他の場合、システムがリブートされ、システム・コールは戻りません。

アスタリスク (\*) の印が付いているシステム・コールでは、通常、セキュリティ関連オプションの場合にのみ監査データが生成されます。ただし、`-e` または `-E` フラグを指定した `auditmask` でプロセスを実行すると、すべてのオプションで監査データが生成されます。

システム・コール監査の次の側面に注意してください。

- NOTE: `uid changed` というレコードがあることがあります。これは主に `SETUID` イベントで発生しますが、あるスレッドがプロセス(タスク)内のすべてのスレッドの `UID` を変更した場合にも記録されます。

- 監査レコードには、v ノード情報と操作対象オブジェクトのファイル・タイプが含まれます。このため、たとえばシンボリック・リンクに対して `chmod` コマンドを実行した場合、リンクで参照される実際のオブジェクトが記録されます。
- 設計上、一部のシステム・コールでは、障害が発生し、その障害がセキュリティ関連でなければ、障害に対する監査レコードが生成されません。このようなシステム・コールのリストについては、表 3-1 を参照してください。
- `ioctl` システム・コールでは、`TIOCSTI` 操作のみ監査されます。
- `fcntl` システム・コールでは、`F_DUPFD`、`F_SETTIMES` および `F_CNVT` のみ監査されます。
- プロセスの監査制御フラグが `USR` に設定された場合、`ioctl` および `fcntl` システム・コールすべてが監査されます。

### 3.1.3.2 トラストッド・イベント

トラストッド・イベントとは、セキュリティ保護メカニズムに関連するイベントのことです。このイベントは、システム・コールに直接対応するとは限りません。トラストッド・イベントの一覧を以下に示します。

<code>audit_daemon_exit</code>	監査デーモンが異常終了したことを示します。これは、監査デーモンの初期化中にメモリが不足した場合にのみ発生します。終了は新しい監査ログに記録され、指定した監査コンソールにメッセージが表示されます。
<code>audit_log_change</code>	監査デーモンが現在の監査ログをクローズし、新しいログに書き込みを開始した(たとえば <code>auditd -x</code> コマンドの結果として)ことを示します。ログの変更は現在の監査ログに記録され、指定した監査コンソールにメッセージが表示されます。
<code>audit_log_create</code>	現在のログ・ファイルが削除されたために、新しい監査ログが作成されたことを示します。新しいファイルには、削除された

ログ・ファイルより 1 大きい世代番号が与えられます。新規ログの作成は新しい監査ログの先頭に記録され、指定した監査コンソールにメッセージが表示されます。

`audit_log_overwrite`

`auditd` に `-o overwrite` オプションを指定したため、監査デーモンが現在の監査ログに上書きを開始したことを示します。上書きは、上書きが開始された監査ログの先頭に記録され、指定した監査コンソールにメッセージが表示されます。

`audit_reboot`

`auditd` に `-o halt` オプションを指定したため (ログがオーバフローした結果)、監査デーモンによりシステムのリブートが開始されたことを示します。リブートは現在の監査ログの末尾に記録され、リブートの前に、指定した監査コンソールにメッセージが表示されます。

`auditconfig`

`auditd` コマンドに `-o changeloc` オプションを指定したため、オーバフロー・アクションが変更されたことを示します。監査の設定変更は現在の監査ログに記録されます。

`audit_start`

監査デーモンが起動されたことを示します。

`audit_stop`

監査デーモンが正常に終了したことを示す (通常、`auditd` に `-k` オプションを指定)。シャットダウンは現在の監査ログの末尾に記録され、シャットダウン時に、指定した監査コンソールにメッセージが表示されます。

<code>audit_suspend</code>	<code>auditd</code> に <code>-o suspend</code> オプションを指定したため (ログがオーバフローした結果), 監査デーモンが監査を一時停止したことを示します。一時停止は現在の監査ログに記録され, 指定した監査コンソールにメッセージが表示されます。
<code>audit_xmit_fail</code>	監査デーモンが監査レコードをネットワーク上で送信し, 転送に失敗したことを示します。転送の失敗は, <code>/etc/sec/auditd_loc</code> ( <code>auditd -r</code> を使用) に次のパスとして指定したローカルのログ, または省略時のローカル・パス ( <code>/var/adm</code> ) に記録されます。
<code>audgen8</code>	<code>audgen8</code> コマンド ( <code>audgen()</code> ルーチンのコマンド行インタフェース) を使って監査レコードが生成されました。
<code>auth_event</code>	ユーザ認証およびユーザ・アカウントの管理に関するイベントが発生しました。トラステッド・イベント <code>auth_events</code> には, <code>passwd</code> , <code>su</code> , <code>rsh</code> , および <code>login</code> が含まれます。イベントは現在の監査ログに記録されます。
<code>login</code>	ユーザがシステムにログインしようとした。
<code>logout</code>	ユーザがシステムからログアウトしました。

### 3.1.3.3 サイト定義イベント

独自の監査イベントを定義することができます (これをサイト定義イベントと言います)。これは, アプリケーションの特定の動作に対してレコードを生成したい場合に役立ちます。

トラステッド・アプリケーション・ソフトウェアでは、サイト定義のイベントおよびサブイベントに対してデータを生成できます。このデータは、システムの監査データとともに監査ログに記録したり、アプリケーション固有のログに格納することができます。

サイト・イベントを定義するには、`/etc/sec/site_events` ファイルを作成し、システムのサイト・イベントにイベント名とイベント番号を付与する必要があります。`site_events` ファイルには、サイト・イベントごとに 1 つのエントリがあり、サブイベント番号を格納できます。

サイト・イベント番号に許される最小値は `MIN_SITE_EVENT` で、`<sys/audit.h>` に定義されています。通常、この値は 2048 です。省略時には最大 64 のサイト・イベントを定義できます。ただし、上限値を最大 1,048,576 まで増やすこともできます。

サイト・イベントの許容数の上限を変更するには、`/etc/sysconfigtab` ファイルにエントリを 1 つ追加した後、システムをリブートします。たとえば、最大 5000 個のサイト定義イベントを使用できるようにするには、`sysconfigdb` コマンドを使用して、次の行を `/etc/sysconfigtab` ファイルに追加した後、システムをリブートします。

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

`/etc/sec/site_events` ファイルを作成すると、アプリケーションでは `audgenl()` ライブラリ・ルーチンを使ってアプリケーション固有の監査データを生成できます。詳細については、`audgenl(3)` を参照してください。

#### 3.1.3.4 イベント・エイリアス

イベント・エイリアスにより、複数の監査イベントを、選択した単一の名前にグループ化できます。イベント・エイリアスには、システム・コール、トラステッド・イベント、または別のエイリアスを含めることができます。次の形式を使って、`/etc/sec/event_aliases` ファイルの省略時の `Tru64` UNIX イベント・エイリアスの後にイベント・エイリアスを定義します。

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

行の継続も可能です。詳細については、`/etc/sec/event_aliases` ファイルを参照してください。

### 3.1.4 監査レコード

監査レコードの出力には、少なくとも、次のデータが含まれます。唯一の例外は、ログインが完了しなかった場合、または監査 ID がプロセスに対して設定されなかった (特定のシステム・プロセスでは通常の状態) 場合、`audit_id` は現れません。

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

audit_id:
```

監査 ID (AUID)。AUID はログイン時にユーザに対応付けられ、ログイン・セッション中は変更されません。ユーザが起動したプロセスには、そのユーザに対応する AUID が付与されます。`su` コマンドを使用してユーザ ID を変更しても、AUID には影響がありません。

AUID と LUID (ログイン UID) は同義です。

`root` アクセス権を得た悪意を持ったプロセスでは、システム・コールを使用して自分の監査 ID を変更できることに注意してください。このような特権プロセスを監視するには、セキュリティ・システム・コールの監視を有効にします。監査 `uid` の変更は、`security()` システム・コールに関する監査レコード内で、最初の引数が `0x3` であることから識別できます。監査レコードには元の監査 ID と新しい監査 ID の両方が格納されています。元の監査 ID は結果コード (リターン値) として保存され、新しい監査 ID は引数として保存されています。

<code>ruid/euid:</code>	実ユーザ ID (RUID) と実効ユーザ ID (EUID)。
<code>pid:</code>	プロセス ID。
<code>ppid:</code>	親プロセス ID (PPID)。プロセスのリストを逆にたどって、そのプロセスが生成されたときのイベントと、それに対応する RUID および AUID を調べるのに役立ちます。
<code>cttydev:</code>	イベントが発生したデバイス。このレコードには主デバイス番号と副デバイス番号が含まれています。
<code>event:</code>	イベント名 (通常はシステム・コールの名前またはトラステッド・イベントの名前のいずれか)。
<code>result:または error:</code>	<p>イベントが成功の場合は、その結果。多くの場合、結果は 0 ですが、一部のシステム・コールでは別の値を返します。たとえば、<code>write()</code> は書き込まれたバイト数を返します。</p> <p>エラーの場合、システム・コールの監査レコードにはエラー・メッセージとエラー番号が返されます。たとえば、次のとおりです。</p> <pre>error:      Not owner (1)</pre> <p>トラステッド・イベントの監査レコードには、追加のエラー・メッセージがある場合があります。</p>
<code>ip address:</code>	イベントが発生したシステムの IP アドレス。



timestamp: イベントの日時。

#### 3.1.4.1 監査レコードの追加エントリ

システム・コールの監査レコード内の大部分のエントリは、そのシステム・コールへの引数です。次のリストでは、監査レコードに関連付けられるエントリのラベルを説明します。

ラベルはコンテキスト依存です。つまり、ラベルの意味は、そのラベルが現れる監査レコードの種類に依存します。たとえば、`mmap()` の監査レコードでは、`flag:` はマップ領域の属性を示します。しかし、`audcntl()` の監査レコードでは、`flag:` は HABITAT 要求とともに渡される数値です。

ユーザがコマンド行に入力したコマンドは、監査レポートでは `char param:` への引数として表示されます。たとえば、ユーザが `august_report` ファイルを `sept_report` という名前のファイルにコピーすると、監査レコードには次の内容が含まれます。

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

個々の監査レコードのコンテキストでは、エントリの意味は直感的に分かります。疑問点がある場合、監査対象のシステム・コールのリファレンス・ページを参照すると、レポートの理解に役立ちます。

監査レコードの各フィールドの名前と説明を以下のリストに示します。

address:                   メモリ・アドレス。たとえば、`mmap()` への引数です。

char param:               文字列。文字列はイベントへの引数またはイベント関連のその他の情報です。たとえば次のとおりです。

```
event:      open
char param: /etc/zoneinfo/localtime
```

cmd name:                このイベントを生成したプロセスの名前 (プロセスを開始した `exec` コマンドの `arg 0`)。

cntl flag:               制御フラグ。たとえば、`audcntl()` 要求へのフラグの 1 つです。

<code>descriptor:</code>	ファイル記述子。状態依存情報が使用できる場合は、実際のファイル名と記述子です。
<code>devname:</code>	デバイス名。
<code>directory:</code>	現在のディレクトリの名前。
<code>flag:</code>	システム・コールへの <code>flag</code> 引数。たとえば、 <code>mmap()</code> のコンテキストでは、マップ領域の属性を示します。 <code>audcntl()</code> のコンテキストの場合は、システム・コールの番号であり、HABITAT 要求の 1 つとともに渡されます。
<code>flags:</code>	フラグとしてシステム・コールに渡される引数。たとえば、 <code>open()</code> では <code>oflag</code> 引数として渡される値です。
<code>gid:</code>	グループ ID。
<code>home dir:</code>	ホーム・ディレクトリ。
<code>hostname:</code>	ホスト名。
<code>inode id:</code>	i ノード番号。 <code>inode dev</code> とともに、ファイル動作の監査レコードとして記録される記述子情報の一部です。
<code>inode dev:</code>	i ノードの主デバイス番号と副デバイス番号。
<code>int param:</code>	整数値。たとえば、 <code>setpgid()</code> では <code>process_id</code> 引数です。
<code>len:</code>	<code>mmap()</code> への引数。メモリ領域の長さです。
<code>login name:</code>	ログイン・ユーザ名

long param:	long 型の値。
mask:	マスク引数。たとえば, <code>audcntl()</code> では <code>SET_SYS_AMASK</code> 要求とともに渡される値です。
object mode:	オブジェクトの保護モード。たとえば, <code>open()</code> ではオープンするファイルのモードです。
operation:	<code>SET_SITEMASK</code> など, <code>audcntl()</code> への要求。
pgrp:	プロセス・グループ ID。たとえば, <code>setpgrp()</code> への <code>process_group_id</code> 引数です。
procname:	PID に対応するプロセス名。
prot:	<code>mmap()</code> への引数。メモリ領域の保護情報です。
req mode:	要求モード。たとえば, <code>open()</code> <code>O_CREAT</code> では新しいファイルの保護モードです。
request:	<code>security()</code> の呼び出しで要求されたセキュリティ・アクション。
shell:	ユーザのシェル・プログラム。通常, このレコード要素は <code>login</code> イベント・レコードに現れます。
username;	イベントに関連するユーザ名。  監査レポートの生成に, <code>audit_tool -w</code> オプションを使用するか監査マネージャの「UID/GID をローカルの名前に変換する」を選択すると, カッコ内にユーザ名が表示されます。このカッコは, <code>/etc/passwd</code> 内でユーザ名を検索するために, 監査縮小ツールで <code>getpw()</code> ルーチンを使用する必要があったことを示します。これは, ログイン時に RUID にユーザ名が対応付けられなかった場合 (たとえば, ログインが監査イベントに含まれていない

どの場合)に発生します。監査イベント間の依存関係については、3.5.3 項を参照してください。

## 3.2 監査サブシステムの構成

監査はカーネル内に組み込む必要があります。使用しているシステムのカーネルに監査サブシステムが構成されていない場合、監査構成手順のガイダンスに従うとカーネルを再構築して、監査サブシステムを含めることができます。

システム単位の監査に加え、エンハンスド・セキュリティ・サブセット OSFC2SEC<sub>nnn</sub> を含めることで、ユーザごとのイベント監査を指定することができます。監査用の `dxaudit` グラフィック・インタフェースには、ソフトウェア・サブセット OSFXC2SEC<sub>nnn</sub> が必要です。*nnn* は、Tru64 UNIX のバージョン番号です。現在のバージョン番号は、『リリース・ノート』を参照してください。これらのサブセットがインストールされているかどうかを調べるには、次のコマンドを実行します。

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

サブセットのインストールの詳細については、『インストレーション・ガイド』および `setld(8)` を参照してください。

監査サブシステムを構成する作業には次のフェーズがあります。

1. 監査用のカーネルを構成し、監査ログの管理パラメータを設定します。  
この部分では、監査ログの格納場所、容量不足になった場合の措置、システムのディスク領域を解放するためにログを削除するまでに監査ログを取っておく期間 (指定によっては無期限)、監査データの格納場所を集中化するかどうかなを選択します。

監査データの格納場所を集中化するかどうかは任意です。TCP/IP ネットワークで接続されたコンピュータがある場合、より高度のセキュリティ・レベルが設定されているリモート・ホスト (監査ハブ) のシステムに監査データを集中化する場合に行います。監査データの格納場所の集中化は、ローカル・システムの監査データを格納する方法に代わる選択肢です。3.2.1 項で説明されているように、構成プロセス時またはそれ以降に、監査データの格納場所の集中化を構成できます。

2. 監査対象イベントを選択します。

次の手順は、監査サブシステムを構成する方法を説明しています。

1. 監査サブシステムの構成のフェーズ 1 では、`sysman auditconfig` コマンドを入力して監査の構成を開始します。使用しているシステムのカーネルに監査サブシステムが構成されていない場合、`sysman auditconfig` のガイダンスに従うとカーネルを再構築できます。
    - a. 「ようこそ」画面で [はい] をクリックし、監査の構成を開始します。
    - b. 「情報」画面で [OK] をクリックします。
    - c. 監査ログの保存場所を指定します。次のいずれかの場所を指定できます。
      - 省略時の保存場所 (`/var/audit/auditlog.hostname.nnn`)
      - システム上のその他の保存場所。
      - 監査ハブにシステムの監査データを格納する場合、その監査ハブのホスト名を入力して、その後ろにコロンを付けます。「ログのパス名」画面で [次へ>] をクリックします。
    - d. 「ログ・ファイル・スペース不足時のアクション」画面で [次へ>] をクリックし、ログのスペースがなくなった場合の動作の省略時の動作を採用します (ファイル・スペースができるまで監査を停止する)。
    - e. 「ログ・ファイルの存在期間」画面で、月数で表される監査ログの省略時の期限を採用します。省略時の期限は無期限 (削除しない) で、削除を実行する時刻は 3 時です。
    - f. このシステムを監査ハブにして、他のネットワーク・システムが監査データを格納するようにしたい場合は、「リモート・クライアントがこのシステムに監査情報を記録できる」オプションを有効にした後、[リモート・クライアントの設定] ボタンをクリックします。「監査の設定：リモート・クライアント」ウィンドウが表示されます。このウィンドウに、監査データを記録する各クライアントの完全修飾子付きドメイン名を入力します。
- 「拡張監査オプション」画面の [フェーズ 1 の完了] をクリックして、監査コンソール・メッセージの省略時の格納先 (`syslog- /var/adm/syslog.dated/current/daemon.log`) を受け入れます。

- g. 「監査イベント情報」画面の「フェーズ 2 へ処理を進めますか？」という質問に対し、[はい]をクリックしてパート2に進みます。

2. 監査サブシステムの構成のフェーズ2では、次の表で説明されている6つのプロファイル・リストから選択する必要があります。監査プロファイルでは、監査サブシステムでの監査内容を定義するパラメータを統合し、プロファイル内でその情報をグループ化する手段を提供します。

プロファイル	説明
Desktop	シングルユーザ・システムに適した最小監査
NIS_server	NIS サーバとして使用するシステムに適した監査
Networked_system	ネットワーク上のシステムに適した監査
Server	ネットワーク・ベースのアプリケーション用のサーバとして使用するシステムに適した監査
Timesharing	複数の対話型ユーザをサポートするシステムに適した最小監査
Timesharing_extended_audit	複数の対話型ユーザをサポートするシステムに適した拡張監査

プロファイルは、次の3つのパートで構成されています。

監査スタイル情報	/etc/sec/auditmask_style 内の Profile: ラベルの下にあります。有効な監査スタイルの特性については、auditmask(8)の -c オプションを参照してください。
監視対象監査イベント	/etc/sec/event_aliases 内の Profile: ラベルの下にあります。
監視対象ファイル	このファイルの一覧は、 /etc/sec/file_objects/Profile ファイルにあります。監視を実行するには、オブジェクト選択フラグおよびオブジェクト選択解除フラグを適切に設定します。/etc/sec/auditmask_style 内の情報で、どのフラグが設定されているか分かります。このファイルはオプションであり、

/etc/sec/auditmask\_style 内で定義されている監査スタイルに obj\_sel と obj\_desel のどちらも含まれていない場合は、このファイルは不要です。

- a. 「監査イベントのカテゴリ選択」画面で、システム構成に最も近いプロファイルを選択し、[次へ>] をクリックします。

CTRL キーを使用すると、ネットワーク上のデスクトップ・システムに対して、Desktop と Networked\_system というように、複数のカテゴリを選択できます。複数のカテゴリを選択できますが、ニーズに合った最小限のプロファイルを使用するようにしてください。

監査するイベントの省略時のリストを採用することとして、「上級ユーザ向け監査イベントの修正/削除」画面で [次へ>] をクリックします。

- b. 一部のプロファイルにはこの手順が必要ですが、それ以外は次の手順に進んでください。監査するファイルの省略時のリストを採用することとして、「UNIX ファイル・システム・オブジェクト」画面で [次へ>] をクリックします。このメニュー項目は、すべてのプロファイルで表示されるわけではありません。
- c. 「拡張オプション」画面で [完了] をクリックし、プロファイルを選択した結果として設定された監査イベントのオプションを採用します。

3. 「監査設定の完了」画面で [了解] をクリックし、監査設定を完了します。

選択が完了すると、auditconfig は次の処理を実行します。

- /etc/rc.config.common ファイル内の AUDITMASK\_FLAG を変更します
- /etc/sec/rc\_audit\_events ファイルに書き込みを行います。
- /etc/sec/fs\_objects ファイルに書き込みを行います。

4. auditd -w コマンドを入力して、監査デーモンの構成をチェックします。省略時の指定を使用した場合、構成が次のように表示されます。

```
# auditd -w
```

```
Audit data and msgs:
```

```
-l) audit data destination
```

```
-c) audit console messages
```

```
= /var/audit/auditlog.hostname.001 [1]
```

```
= 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] 監査ログの名前。
- [2] 監査サブシステムのメッセージの格納場所。監査サブシステムの状態が変化すると、ここに記録されます。
- [3] ネットワークを介した監査は有効になっていません (リモート・クライアントはこのシステムに記録されません)。
- [4] 監査でオーバフロー状態として検出するファイル・システムの空きスペースの量 (パーセント)。この場合、10 パーセントです。  
監査ログが置かれているファイル・システムが 90 パーセント以上使用された状態になると、監査サブシステムはオーバフロー・アクションを実行します。
- [5] オーバフロー・アクションでは、格納スペースができるまで監査を一時停止します。

*alternate file systems* の部分には、`/etc/sec/auditd_loc` で指定したディレクトリ名が表示されます。

### 3.2.1 監査データ格納場所の集中化

TCP/IP ネットワークで接続された複数のコンピュータがある場合、ローカル・システムに監査データを格納する方法の代わりに、複数のシステムで監査デーモンを実行し、リモート・システム (監査ハブ) の監査デーモンに監査データを送信し、より高度なセキュリティ・レベルが設定されている単一のシステムに情報を格納することができます。

監査データの破損を防ぐために、リモート・システムの監査データは、監査ハブにある専用の監査ログ・ファイルに格納されます。

監査ツールを使用して、リモート・システムから監査データのログを取り出すと、監査ログの最初と最後のエントリは、完全なエントリにならずに途中で切れている可能性があります。これは、通信チャネルが正しく切断されなかったり、`auditd` デーモンがリモートでデータを受信しているときに口



グ・ファイルが強制的に切り替えられた場合に発生します。原因は、監査ハブへ送られるリモートの監査情報が、監査エントリごとではなく連続したストリームとして送られるためです。

エントリが途中で切れている場合は、監査ツールによって通知されます。監査ログからの他のレコードの取り出しには影響はありません。

監査データの格納場所の集中化には、監査ハブの構成と、この監査ハブに監査データを送信するリモート・システムの構成が必要です。監査サブシステムを構成する時点で、監査データの格納場所を集中化するように構成していない場合は、次の項で説明されているコマンドを使って構成します。

### 3.2.1.1 監査ハブへの監査データ格納場所の集中化の構成

監査データの格納場所を集中化するように監査ハブを構成するには、次の手順を実行します。

1. 監査情報の収集拠点となるホスト (監査ハブ) で、`/etc/sec/auditd_clients` ファイルを作成します。このファイルの各行に、監査ハブに監査データを送信するリモート・ホストの名前をそれぞれ入力します。
2. 次のコマンドを入力して監査ハブを使用可能にし、`/etc/sec/auditd_clients` ファイルに指定したリモート・ホストの監査デーモンからの監査データを受信できるようにします。

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

3. リモート監査デーモンのオプションを次のように設定します。

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

たとえば、ID 6 の監査デーモンが稼働しているリモート・ホストに対し、監査ログの格納場所を `/var/audit/NYC_Sys1` に設定するには、次のように入力します。

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

リモート・ホストの監査デーモン ID は整数です。監査デーモン ID を表示するには、次のように入力します。

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

このコマンドは、現在のオプションを表示します。少なくとも 1 つの子監査デーモンが存在しなければ、監査 ID は表示されません。1 つ以上の監査デーモンを実行している場合に、`-p` オプションが指定されないと、マスタ・デーモン (ローカル・システムの監査データを受信) が要求を処理します。

監査ツールを使用して、リモート・システムから監査データのログを受信すると、監査ログの最初と最後のエントリは、完全なエントリにならずに途中で切れている可能性があります。これは、通信チャネルが正しく切断されなかったり、`auditd` がリモートでデータを受信しているときにログ・ファイルが強制的に切り替えられた場合に発生します。原因は、監査ハブへ送られるリモートの監査情報が、監査エントリごとではなく連続したストリームとして送られるためです。

エントリが途中で切れている場合は、監査ツールによって通知されます。監査ログからの他のレコードの取り出しには影響はありません。

#### 3.2.1.2 監査ハブでのリモート・システムの監査データ記憶域の構成

それぞれのリモート・ホストで次のコマンドを入力し、監査ハブへ監査データを送信するようにします。

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

リモート・システムが監査ハブに接続できない場合、`auditd` の `-o` および `-r` フラグの指定に従って、リモート・ホストの監査デーモンは監査データをローカルに保存します。

### 3.3 監査サブシステムの管理

次の項で、以下を説明します。

- 監査サブシステム起動時の省略時の値を変更する
- 監査デーモンを起動、停止、一時停止する
- 監査ログをアーカイブする
- 監査データを回復する

### 3.3.1 監査サブシステムの起動時の省略時設定の変更

監査サブシステムに対するシステム起動時の省略時設定は、  
/etc/rc.config.common ファイル内の AUDITMASK\_FLAG 変数に保存されます。このフィールドを使用して、システムの起動時にコマンド行引数を auditmask に渡すことができます。AUDITMASK\_FLAG 変数により、監査スタイル・フラグが設定され、/etc/sec/rc\_audit\_events ファイルから監査用に選択されたイベントを auditmask が読み込むように設定されます。/etc/sec/fs\_objects ファイルには、監査サブシステムの構成時に、オブジェクトの選択および選択解除を監視するようにプロパティ・リストが変更されたファイルのリストが含まれます。

/etc/rc.config.common ファイルの AUDITMASK\_FLAG 変数は、rcmgr コマンド、sysman auditconfig コマンド、または CDE のダッシュボードから「監査マネージャ」を使用して変更できます。

/etc/sec/rc\_audit\_events ファイルは、テキスト・エディタ、sysman auditconfig コマンド、または CDE のダッシュボードから「監査マネージャ」を使って変更できます。

### 3.3.2 監査デーモンの起動、停止、一時停止

監査デーモンを起動するには、次のように入力します。

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

監査デーモンを停止するには、次のように入力します。

```
# auditd -dk
```

TruCluster Server 環境で監査デーモンを停止するには、次のコマンドを入力します。

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

監査デーモンを一時停止するには、次のように入力します。

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

### 3.3.3 監査ログのアーカイブ

監査ログ・データを収集した後、監査ログ・ファイルを体系的にアーカイブする手順が必要です。監査ログ・データを使用する多くのアプリケーションでは、数ヶ月前のデータを検索して分析する必要があります。

sysman auditconfig ユーティリティでは、root の cron ジョブを使用して、バイナリ監査ログを定期的に削除することができます。この機能は省略時の設定では無効になっています。この機能を有効にするときには、cron ジョブを実行する前に、監査ログ・ファイルのバックアップを定期的に取りようにする必要があります。

「ログ・ファイルの存在期間」ダイアログ・ボックスには、削除を実行する月単位の周期や削除の基準が表示されます。削除は指定した月の 1 日 (初日) に実行されます。時間は設定可能ですが、分は 0 分に設定されています。バイナリ監査ログは、バイナリ監査ログに含まれるすべての監査イベントが、削除実行日から「ログ・ファイルの存在期間」を差し引いた日付よりも古くなると削除されます。

たとえば、2 カ月目、4 カ月目、6 カ月目の 1 日午前 3 時に実行する cron ジョブを作成するには、次のように入力します。

```
"ログ・ファイルの存在期間" = "2 カ月間隔"  
"削除する時間 (0-23)" = "3"
```

バイナリ・ログ・ファイルは含まれているエントリがすべて 2 カ月以上古くなると、cron 実行時に削除されます。

### 3.3.4 監査データの回復

システムでパニック状態が発生した場合、crashdc ユーティリティを使って、パニック時にシステムに残されたすべての監査データを抽出し、crash ディレクトリ内の audit-data.*n* ファイルに格納します。

*n* はクラッシュ番号です。監査データがない場合、このファイルは作成されません。audit\_tool コマンドを使って、audit-data.*n* ファイルの内容を表示できます。

一部の監査レコードは、auditlog ファイルと audit-data.*n* ファイルの両方に記録されることがあります。audit-data.*n* ファイル内の最初の監査レコードは完全でない可能性があります。audit\_tool コマンドでは、これを破損レコードとして印を付けます。この場合、監査レコードは通常の auditlog ファイルに書き込まれています。

## 3.4 監査イベントの管理

次の項で、以下の作業の方法を説明します。

- 監査マスクを表示する

- システムで監査できるイベントを特定する
- 監査イベントを有効にする
- 監査イベントを無効にする
- プロセスをトレースする
- ファイル・アクセス操作を監査する

### 3.4.1 監査マスクの表示

現在の監査マスクを表示し、成功イベント、失敗イベントの一方または両方について監査されているかどうか確認するには、次のコマンドを使います。

- CDE のダッシュボードの `dxaudit` プログラム。  
「アプリケーション・マネージャ」 「日常管理」 「監査マネージャ」 [収集]
- `auditmask` コマンド。

```
# auditmask
```

`succeed` というキーワードを入力すると、成功イベントが監査の対象となります。`fail` というキーワードを入力すると、失敗イベントが監査の対象となります。両方のキーワードを入力すると、成功と失敗の両方のイベントが監査の対象となります。

### 3.4.2 システムで監査できるイベントの特定

次の手順を実行して、監査可能なすべてのイベントのリストを含む `all_auditable_events` ファイルを作成します。

1. 現在の監査マスクを `original_audit_mask` ファイルに保存します。

```
# auditmask > original_audit_mask
```

2. すべてのイベントを監査するように監査マスクを設定します。

```
# auditmask -f
```

`auditmask -f` コマンドを実行すると、システムの速度が低下することがあります。

3. `all_auditable_events` ファイルに監査マスクを保存します。

```
# auditmask > all_auditable_events
```

4. どのイベントも監査しないように監査マスクを設定します。

```
# auditmask -n
```

5. 監査マスクを元の状態 (イベント) に戻します。

```
# auditmask < original_audit_mask
```

### 3.4.3 監査イベントの有効化

監査サブシステムでは大量のデータを生成し収集できます。このため、監査サブシステムが生成するデータの上限を適切に設定し、監査ログの増加を監視する必要があります。この監視によって、監査ログでファイル・システムが一杯になり、サービス不能の状態にならないようにします。

監査サブシステムを構成し、プロファイルを選択する際に、監査対象のイベントを選択しますが、イベントの選択や選択解除を行えば、いつでも監査マスクを変更できます。

監査対象のイベントを選択するには、次のコマンドを使います。

- CDE のダッシュボードの `dxaudit` プログラム。  
「アプリケーション・マネージャ」 「日常管理」 「監査マネージャ」 [収集]
- 次の `auditmask` コマンド。

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

`event` 引数を `auditmask` コマンドを指定してシステム監査マスクを設定します。これは累積動作です。そのため、あるイベント・セットの監査をオンまたはオフにした後、最初のイベント・セットを変更しなくても、次のイベント・セットの監査をオンまたはオフにできます (これらのイベント・セットの共通集合は除く)。

`auditmask` コマンドに対するコマンド行の引数には、`:succeed:fail` オプション・フィールドがある 1 つ以上のイベントを含めることができます。このフィールドの `succeed` は、成功イベントを監査しないように指定する場合は 0、監査する場合は 1 です。`fail` は、失敗イベントを監査しないように指定する場合は 0、監査する場合は 1 です。

監査対象のイベントを選択するための `auditmask` コマンドの使用例のいくつかを次に示します。詳細については、`auditmask(8)` を参照してください。

- イベントをすべて選択するには、以下のコマンドを実行します。

```
# auditmask -f
```

このオプションでは、システムの速度が著しく低下します。

- 成功と失敗の両方のイベントを選択するには、次のコマンドのいずれかを実行します。

```
# auditmask eventname
# auditmask eventname:1:1
```

- 成功イベントのみを選択するには、次のコマンドを実行します。

```
# auditmask eventname:1:0
```

- 失敗イベントのみを選択するには、次のコマンドを実行します。

```
# auditmask eventname:0:1
```

- 指定したファイルのイベントを選択するには、次のコマンドを実行します。

```
# auditmask < filename
```

- 特定のユーザが生成したプロセスを選択するには、次のコマンドを実行します。

```
# auditmask -a AUID-of-user
```

- 特定のファイルについてのみファイル・アクセスを監査する (オブジェクト選択) には、次のコマンドを実行します。

```
# auditmask -x filename [:1|:0]
```

- システム・マスクまたはプロセス・マスクがイベントを指定した場合に、そのイベントを選択するには、次のコマンドを実行します。

```
# auditmask -c or
```

- システム・マスクおよびプロセス・マスクの両方がイベントを指定した場合にのみ、そのイベントを選択するには、次のコマンドを実行します。

```
# auditmask -c and
```

- プロセス監査マスクだけに基づいてイベントを選択するには、次のコマンドを実行します。

```
# auditmask -c usr
```

### 3.4.4 監査イベントの無効化

それぞれのイベントに対して監査を無効にするには、次のコマンドを実行します。

- すべてのイベントを無効にするには、次のコマンドを実行します。

```
# auditmask -n
```

- 特定のイベントを無効にするには、次のコマンドを実行します。

```
# auditmask eventname:0:0
```

- 特定のファイルに対するファイル・アクセス (オブジェクト選択) を無効にするには、次のコマンドを実行します。

```
# auditmask -y filename
```

### 3.4.5 プロセスのトレース

`auditmask` ユーティリティを使って、システム、単一のプロセス、あるいはユーザの監査 ID (AUID) に関連付けられているすべてのプロセスの監査特性を調整できます。監査レコードの生成は、システム監査マスク、プロセス監査マスク、およびプロセス `audcntl` フラグに基づきます。それぞれの監査マスクは、すべてのイベントのビットマップです。

プロセス `audcntl` フラグは、以下を指定します。

`or` システムまたはプロセス監査マスクのどちらかに指定イベントがあれば監査対象とします。省略時の `audcntl` フラグ設定は `or` です。

`and` システムおよびプロセス監査マスクの両方に指定イベントがあれば監査対象とします。

`off` プロセスに対して監査を行いません。

`usr` プロセス監査マスクに指定イベントがあれば監査対象とします。

システム・コールのセットやエイリアス名を指定することで、監査対象のイベントを指定できます。エイリアスは `/etc/sec/event_aliases` ファイルで定義されており、そこに追加もできます。このファイルを編集して、システ



ム・コールを追加または削除できます。オプションの拡張を使って、あらゆる成功イベントと失敗イベントを識別できます。たとえば次のとおりです。

`open:1:0`                      `open()` コールの成功を監査するよう指定します。

`open:0:1`                      `open()` コールの失敗を監査するよう指定します。

次の例は、さまざまな目的に即した `auditmask` コーティリティの使用方法を示します。これらの例では、プロセス管理マスクを変更します。特に指定しない限り、プロセス `audcntl` フラグは省略時の設定 `or` のままです

- コマンド引数を実行して、新しく作成されたプロセスのすべてを監査するには、次のコマンドを実行します。

```
# auditmask -E command args
```

- コマンド引数を実行し、新しく作成されたプロセスの失敗した `open()` システム・コールと成功した `ipc` イベント (`/etc/sec/event_aliases` で定義) を監査するには、次のコマンドを実行します。

```
# auditmask open:0:1 ipc:1:0 -e command args
```

- プロセス ID (PID) 999 について、`gettimeofday()`: 以外の、すべての (`-f`) イベントを監査します。

```
# auditmask -p 999 -f gettimeofday:0:0
```

- PID 999 について、(`-n`) イベントを監査しないようにします。

```
# auditmask -p 999 -n
```

- PID 999 について、`open()` および `exec()` イベントを監査します。

```
# auditmask -p 999 open exec
```

- PID 999 について、監査中のイベントに `exit()` を追加します。

```
# auditmask -p 999 exit
```

- PID 999 について、監査中のイベント・セットを取得します。

```
# auditmask -p 999
```

- PID 999 について、`audcntl` フラグを `usr` に設定します。

```
# auditmask -p 999 -c usr
```

現在実行中のプロセスをトレースする場合、`auditmask -c usr` コマンドを使用して、これらのシステム・コールに対するすべてのオプションをトレースします。

- AUID 1123 のユーザが所有する全プロセスに対し、すべての ipc イベントを監査します (AUID はユーザの初期 RUID と同じです)。

```
# auditmask -a 1123 ipc
```

- auditmask ヘルプ・オプションにアクセスするには、次のコマンドを実行します。

```
# auditmask -h
```

### 3.4.5.1 トレース・プロセス・データの表示

トレース・プロセスに関する情報を表示するには、次のコマンドを実行します。

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

auditd -d オプションは、カーネルのデータを監査ログに書き出します。

auditd -q オプションは、現在の監査ログの名前を提供します。

次のような情報が表示されます。

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 アクティブ・プロセスの監査

例 3-1 は , guest としてログインしたユーザによって起動されたプロセスを調べるために使用できるコマンドを示します。

#### 例 3-1: アクティブ監査セッションの例

```
# 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:

! Audcnt1 flag: または

# 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
-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 )
```

### 例 3-1: アクティブ監査セッションの例 (続き)

```
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
```

- ❶ ユーザ `guest` が実行しているプロセスを見つけ出し、プロセス ID と監査 ID も取得します。
- ❷ PID 23563 に対して、`auditmask` を `open` および `exec` に設定し、システム・マスクで OR 操作を実行します。`exec` は `execv`、`exec_with_loader` および `execve` のエイリアスであることに注意してください。
- ❸ 23563 プロセスの `auditmask` を取得します。
- ❹ 監査ログを 5 秒ごとにダンプ出力し、`auditd` の構成も表示します。
- ❺ 連続した (-f) 簡略型の (-B) 監査レポートを表示します。AUID を対応するユーザ名に変換します (-w)。  
監査ログの名前は、`auditd -w` コマンドの結果として取得されたことに注意してください。
- ❻ アスタリスク (\*) は `setuid` を伴う操作を示します。
- ❼ Ctrl/C で `audit_tool` プログラムを終了します (監査は継続します)。

詳細については、`auditmask(8)` を参照してください。

#### 3.4.5.3 追加のシステム・コール引数の動的な監査

追加のシステム・コール引数が必要な場合、`dbx` コマンドを使用して、任意のシステム・コールについて、記録する引数を動的に変更できます。システム・コールは、`/usr/include/sys` ディレクトリ内の関連する `.h` ファイルで定義されています。たとえば、`flock` はシステム・コール 131 (`/usr/include/sys/syscall.h` ファイルで定義) であり、ファイル記述子とオプションを引数として受け取ります。これらの引数を監査するには、次のように `dbx` コマンドを入力します。

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

aud\_parm 配列の最初のエントリはシステム・コールの最初の引数に対応し、2 番目のエントリはシステム・コールの 2 番目の引数に対応し、以降も同様に対応します。c エンコードはファイル記述子を記録することを示します。a エンコードは整数の引数を記録することを示します。エンコードのセットについては、<sys/audit.h> ファイルに説明があります。

### 3.4.6 ファイル操作の監査

オブジェクトの選択および選択解除は、監査ログの拡大を管理するための強力なツールです。

mount および reboot などのイベントは、システムの状態に影響を及ぼします。open および stat などのデータ・アクセス・イベントはファイルに作用します。reboot の動作は、通常、セキュリティに影響しますが、ファイルの open イベントは、必ずしもセキュリティに影響しません (サイトのセキュリティ・モデルに依存します)。

オブジェクトの選択および選択解除により、ファイルがデータ・アクセス操作の対象となったときに、どのファイルについて監査レコードを記録 (選択) し、またどのファイルについては記録しない (選択解除) かを選択できます。

データ・アクセス操作を次に示します。

```
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
```

オブジェクトの選択および選択解除は次のように機能します。

選択	オブジェクトの選択を行って、指定のファイル・セットのいずれかで実行された特定のデータ・アクセス操作を監査できます。auditmask -X コマンドまたは auditmask -x コマンドを使って、ファイル・プロパティ・リストでオブジェクトの選択フラグを設定することによって、これらのファイルが指定されます。同じ操作でも、フラグが設定されていないファイルで実行された場合は、監査データを作成しません。
----	---

たとえば、`/etc/passwd` ファイルと `/.rhosts` ファイルにフラグを設定し、`open` システム・コールを監査できます。これにより、`/etc/passwd` ファイルまたは `/.rhosts` ファイルの `open` は監査されますが、`/tmp/xxxx` ファイルの `open` は監査されません。

**選択解除** オブジェクトの選択解除を行って、指定のファイル・セットのいずれかを変更する特定のデータ・アクセス操作を監査できます。フラグが設定されていないファイルで実行された同じ操作は、ファイルが変更されたかどうかにかかわらずに監査データを作成します。`auditmask -Y` コマンドまたは `auditmask -y` コマンドを使って、ファイル・プロパティ・リストでオブジェクトの選択解除フラグを設定することによって、これらのファイルが指定されます。

`open` によるファイルの書き込みアクセスや切り捨てアクセスがファイル変更の例です。

監査の選択および選択解除では、`auditmask -c usr` コマンドで指定したプロセスに対する監査を削減するものではありません (監査制御フラグの値は `AUDIT_USR`)。

オブジェクトの選択および選択解除を使用するには、次の 3 ステップの手順を実行する必要があります。

1. オブジェクトの選択および選択解除を適用するファイルを選択します。複数のファイルに選択または選択解除を適用する場合、ファイル名 (完全なパス名) をリストしたファイルを作成します。1 行に 1 つファイル名を記述します。
2. 必要に応じて、監査スタイルを `Object Selection` または `Object Deselection` のいずれかに設定します。

監査マネージャから: 「収集」 「システムマスクの変更」

コマンド行から:

```
# auditmask -s obj_sel
```

または

```
# auditmask -s obj_desel
```

3. オブジェクトの選択および選択解除を次のように起動します。

オブジェクトの選択が 1 ファイルの場合:

```
# auditmask -x filename
```

オブジェクトの選択が *filelist* という名前のファイルにリストされている一連のファイルの場合:

```
# auditmask -X filelist
```

オブジェクトの選択解除が 1 ファイルの場合:

```
# auditmask -y filename
```

オブジェクトの選択解除が *filelist* という名前のファイルにリストされている一連のファイルの場合:

```
# auditmask -Y filelist
```

次の例に、監査対象ファイルの選択を有効にする方法を示します。

```
# 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
```

次の例に、一連のファイルを選択解除する方法を示します。

```
# 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
```

ファイル・リストに対するオブジェクトの選択フラグおよび選択解除フラグのステータスは、CDE のダッシュボードから次のように表示できます。

1. 「アプリケーション・マネージャ」 「日常管理」 「監査マネージャ」 [収集] [システムマスクの変更] [現在] [編集] [Object Selection/Deselection] を選択します。
2. 「ファイル」で、`/etc/sec/fs_objects` を指定します。
3. 「ファイル・タイプ」で、「ファイルの一覧」を指定します。
4. 「設定」はそのままにします。
5. [Query] をクリックします。

システムのオブジェクト選択およびオブジェクト選択解除の監査スタイルは相互に排他的ですが、異なるシステム上で同じオブジェクトに対して両方のモデルを同時に指定する可能性があります (NFS 経由)。NFS を介して属性を転送するには、`proplistd` デーモンを実行している必要があります。

### 3.5 監査レポートの生成および表示

省略時の設定では、次のような名前の付いた監査ログ・ファイルの `/var/audit` ディレクトリに監査レコードが保存されています。

`auditlog.hostname.nnn`

`hostname` はローカル・ホストの名前です。`nnn` は、自動生成される数値のサフィックスで、監査ログ・ファイル名に付加されます。数値のサフィックスは、000 ~ 999 までの連番です。

監査ログから、表示する監査レコードを含んだレポートを生成できます。次の機能を使用して、監査レポートを生成し、表示できます。

- 次のオプションを選択することによって、CDE のダッシュボードから「監査マネージャ」のグラフィック・インタフェースを使用できます。  
「システム管理」 「日常管理」 「監査マネージャ」 [レポート]  
[レポートの生成]

監査マネージャを使ってログを表示する場合、レポートに含めるイベント、時刻、AUID、および監査レコードのその他の属性などを指定した選択ファイルを作成します。



- `audit_tool` コマンド。`audit_tool` コマンドの使用例のいくつかを次に示します。詳細については、`audit_tool(8)` を参照してください。

---

#### 注意

---

監査サブシステムに新しいイベントが追加されているため、古いバージョンの Tru64 UNIX オペレーティング・システムで実行する `audit_tool` コマンドは、新しいイベントを認識できず、認識できないイベントについてエラー・メッセージを表示します。

---

- 一定期間に発生したすべての動作についてレポートを生成し、表示するには、次のコマンドを実行します。  

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

`start` 時間および `end` 時間は、`yyymmdd[hh[mm[ss]]]` という書式で指定します。時間 (hh)、分 (mm)、および秒 (ss) はオプションです。
- 特定のユーザによるすべてのファイルのオープンの失敗についてレポートを生成し、表示するには、次のコマンドを実行します。  

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

`login` を監査対象として事前選択していない場合、監査ログにはユーザ名が表示されません。この場合、UID (`audit_tool -u UID`) または AUID (`audit_tool -a AUID`) を指定することができます。
- 監査ログ内の各監査 ID (AUID) についてレポートを生成し、表示するには、次のコマンドを実行します。  

```
# audit_tool -R auditlog
```

省略時の設定では、レポート名は `report.AUID` となります。
- テキスト `string` がパラメータ・フィールドに含まれている、または特定の記述子に対応しているすべての監査レコードについてレポートを生成し、表示するには、次のコマンドを実行します。  

```
# audit_tool -s string auditlog
```
- 特定のプロセス ID に関連するすべての監査レコードのレポートを生成し、表示するには、次のコマンドを実行します。  

```
# audit_tool -p PID
```

- 特定のプロセス ID およびそのプロセスの子孫に関連するすべての監査レコードに対してレポートを生成し、表示するには、次のコマンドを実行します。

```
# audit_tool -p -PID
```

- ULTRIX システムで生成された監査ログに対してレポートを生成し、表示するには、次のコマンドを実行します。

```
# audit_tool.ultrix
```

### 3.5.1 監査レコードのフィルタリング

選択解除ファイルの内容を使用して、表示したくない監査レコードをフィルタ処理によって除外することができます。選択解除ファイルは、次の形式で入力された 1 行以上の文字列で構成されています。

[*hostname audit\_ID RUID event pathname flag*]

フィールド内のアスタリスク (\*) はワイルドカードであり、常に一致します。文字列の末尾にプラス記号 (+) を付けると、指定した文字列で始まるすべての文字列と一致します。*flag* では、*open* イベントに対して読み取りモード (r) または書き込みモード (w) を指定します。たとえば、パス名が */usr/lib/* で始まるオブジェクトについて、読み取りアクセスを行うすべての *open* 操作をフィルタ処理によって除外するには、ファイルに次の行を指定します。

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

選択解除ファイルで指定した行は、その他の選択オプションよりも優先されます。複数の選択解除ファイルを作成できますが、監査の縮小を実行するときには、選択解除ファイルは 1 つしか指定できません。

選択解除ファイルを使って、監査レポートを表示するには、次のコマンドを実行します。

```
# audit_tool -d deselection_file
```

### 3.5.2 簡略型監査レコードの表示

*audit\_tool -B* コマンドにより、簡略レコード形式で監査レポートが生成されます。

簡略型レポートの例は次のとおりです。

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
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)

```

簡略型レポートのカラム見出しは次のとおりです。

AUID:RUID:EUID	イベントに関連する監査 ID , 実ユーザ ID , および実効ユーザ ID。
PID	プロセス ID 番号。
RES/ (ERR)	RES は結果コードです。結果コードについては , 各システム・コールのリファレンス・ページを参照してください。  (ERR) は , エラーが発生した場合のエラー・コードです。エラー・コードの一覧と各コードの意味については , <code>errno(2)</code> を参照してください。
EVENT	最後の欄にはイベントと引数が表示されます。

以下の情報は , 簡略型レポートには表示されません。

- ユーザ名 (AUID:RUID:EUID の代わりにユーザ名を表示したい場合は , `audit_tool -wB` コマンドを使用します)
- PPID
- デバイス ID
- 現在のディレクトリ
- i ノード情報
- 記述子に対応する名前文字列
- IP アドレス
- タイムスタンプ

カスタマイズした簡易レポートを生成するには , `audit_tool(8)` の `-O` オプションを参照してください。

### 3.5.3 監査イベント間の依存関係

監査ログの一部の情報は、以前に監査したイベントに基づいています。たとえば、LOGIN イベントはログイン名と実 UID (RUID) を対応付けます。その結果、以降の (あるプロセスに対する) RUID はログイン名と対応付けられます。このようなデータは、状態依存情報と呼ばれます。省略時の設定では、audit\_tool コマンドは監査レコードの状態に関する情報を相互参照します。参照しないようにするには、-F オプションを実行します。

次の3つの監査レコードは、状態依存情報の例を示します。最初のレコードは、/etc/passwd の正常な open() を示し、3 という値を返しています。

```
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
```

次のレコードは、/etc/passwd ファイルに対する ftruncate() システム・コールの結果を状態依存情報を使って示します。状態依存データにより、このプロセスの記述子 3 にファイル名 /etc/passwd が対応付けられています。

```
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
```

状態依存データを管理していない場合、ftruncate() システム・コールが記述子 3 (vnode ID = 2323, dev = 8,1024) に対して発行されたことのみ表示されます。

```
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
```

表 3-2 に、状態依存情報およびそれを管理するのに必要な監査イベントをリストします。

表 3-2: 状態依存情報

必要な状態依存情報	監査するイベント
ログイン・ユーザ名	login , logout
プロセス名	execv , execve , exec_with_loader , exit
ファイル情報/ソケット情報	open , old_open , close , dup , dup2 , fcntl , bind , connect , accept , naccept , socket , proplist_syscall
現在のディレクトリ	chdir , chroot , fchdir
監査の状態	audit_suspend , audit_log_create , audit_log_overwrite , audit_shutdown , audit_xmit_fail

exit() システム・コールは、このシステム・コールによって終了するプロセスに対する状態依存情報が不要になることを audit\_tool に知らせます。これにより、audit\_tool の実行が速くなります。

状態依存データに関心がない場合、exit() を監査する必要はありません。また、audit\_tool プログラムに -F オプションを指定して、高速モードにします。

詳細については、audit\_tool(8) を参照してください。

### 3.6 従来の UNIX のロギング・ツール

セキュリティ関連の監査には監査サブシステムを使用します。従来の UNIX オペレーティング・システムのロギング・ツールが使用でき、次のイベント・カテゴリに対する監査機能が提供されています。

- ローカル・ログインおよびログアウト
- ファイル転送プロトコル (FTP) ログイン

- TCP/IP の外部ログインおよびログアウト (rlogin および telnet)
- DECnet の外部ログインおよびログアウト (dlogin および set host)
- ログイン失敗
- スーパーユーザへの切り替え失敗 (su コマンド)
- リブートおよびクラッシュ
- rsh および rcp のファイル転送要求
- DECnet のファイル転送要求

これらの各カテゴリの監査には、関連情報を保存するデータ・ファイル、および保存されたデータを参照する手段などが関係します。データを参照する手段として last や lastcomm などの特定コマンドを使用する場合があります。その他の場合は、ファイル内容を、more コマンドなどで直接表示します。

表 3-3 のように、アカウント・データは /var/adm ディレクトリのいくつかのファイルに保存されます。システムにあるログ・ファイルは、どのログイン機能およびアカウント機能を有効にしているかによって異なります。

**表 3-3: /var/adm にある従来の UNIX ログ・ファイル**

ファイル名	セキュリティ関連情報
wtmp	すべてのログイン、ログアウト、およびシャットダウンが記録されます。このログは last コマンドを使用して表示します。
syslog.dated/date/daemon.log	システム・デーモンによって生成されたメッセージ。
syslog.dated/date/kern.log	カーネルによって生成されたメッセージ (システム・クラッシュなど)。
syslog.dated/date/lpr.log	ライン・プリンタのスプール・システムによって生成されたメッセージ。
syslog.dated/date/mail.log	メール・システムによって生成されたメッセージ。
syslog.dated/date/user.log	ユーザ・プロセスによって生成されたメッセージ。

表 3-3: /var/adm にある従来の UNIX ログ・ファイル (続き)

ファイル名	セキュリティ関連情報
syslog.dated/date/syslog.log	DECnet ファイル転送の要求。
acct	ユーザ・コマンドを含む生のシステム・アカウント・データ。

これらのファイルの内容を保護してください。これらのファイルおよびディレクトリは、root アカウントが所有し、group や other では書き込み不可とする必要があります。

UNIX アカウントについては、『システム管理ガイド』を参照してください。

### 3.7 TruCluster での監査

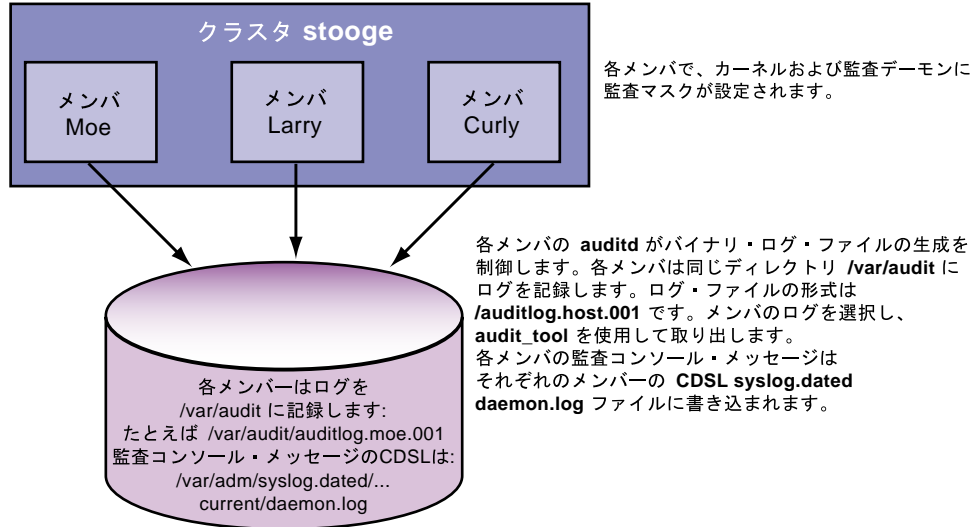
/usr/sbin/sysman auditconfig ユーティリティの [Custom Setup] メニュー上の「Audit Configuration」アイコンを使うと、監査オプションをクラスタ単位で構成できます。クラスタ作成時のカーネル構築では、監査のサポートが自動的に組み込まれます。

監査の構成は、2 段階の手順で行います。最初に、監査ログ・ファイルの名前および位置などの、監査デーモンのパラメータを指定します。次に、監査マスクと呼ばれる、監査対象のイベント一式を選択します。構成ユーティリティによって、各クラスタ・メンバで同じパラメータとイベントが使われるようになります。一元管理を維持するために、auditconfig ユーティリティは、この監査構成情報を /etc/rc.config.common ファイルに格納します。

監査を有効にするために新たにカーネルを構築する必要ありません。これは、クラスタ・カーネルの最初の構築で、DEC\_AUDIT カーネル・オプションと特殊ファイル /dev/audit がすべてのクラスタ・メンバに自動的に組み込まれるためです。

監査が有効のとき、監査デーモン (auditd) がクラスタの各メンバ上で実行されます。各監査デーモンは、イベント固有の監査ログ・エントリをプライベート監査ログ・ファイルに記録します。省略時のプライベート監査ログ・ファイルの名前は、/var/audit/auditlog.{membername}.nnn です。各監査ログ・エントリにはそのログが発生したメンバのホスト名が含まれているため、複数の監査ログ・ファイルのエントリを簡単にマージして表示することができます。図 3-2 は、監査データが収集される様子を示しています。

図 3-2: クラスタ内の監査データの流れ



ZK-1579U-AIJ

各監査デーモンは、そのエラーまたは状態のメッセージをローカルの  
/var/adm/syslog.dated/DATE/daemon.log ファイルか、共通の監査コ  
ンソール・ファイル(オプション)に書き込みます。

監査するイベント・式を監査マスクと呼びます。この監査マスクは、  
auditconfig ユーティリティを使って最初に生成されます。このユーティ  
リティでは、クラスタ単位で作成される共通監査マスクを指定します。監査  
デーモンの最初のスタートアップでは、/etc/rc.config.common ファイ  
ルの情報を使います。このため、各メンバで、同じ auditd コマンド行オ  
プションと、同じ監査マスクが使用されます。

実行中のクラスタでは、auditd -cluster オプションを使うと、すべての  
アクティブ・デーモンを auditd コマンドの対象とすることができます。た  
とえば auditd -cluster -w は、各メンバの状態を表示します。同じよう  
に auditmask -cluster オプションでは、すべてのアクティブ・メンバの  
監査マスクを変更または表示することができます。

1 つのクラスタ内の監査デーモンに異なる監査パラメータや異なる監査マ  
スクを指定することは可能ですが、混乱しやすいため避けてください。

audit\_tool ユーティリティでは、複数の監査ログ・ファイルをマージし  
て、時間経過順でソートされたイベントをクラスタ単位で表示することがで



きます。各イベントにホスト名情報が記録されるため、イベントの発生箇所を簡単に確認することができます。

### 3.7.1 クラスタ・コマンドの例

この項では、2つのメンバ(haydn と handel) からなるバージョン 5.1 A クラスタ内のメンバ・システムで実行する監査コマンドの例を示します。

各メンバの auditd 状態を表示するには、次のようにコマンドを実行します。

```
# 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
```

各メンバの auditd マスクの状態を表示するには、次のようにコマンドを実行します。

```
# 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
```

メンバが現在ログをとっているログ・ファイルの名前を表示するには、(メンバ hayden から) 次のようにコマンドを実行します。

```
# auditd -cluster -q

/var/audit/auditlog.haydn.001

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

すべてのクラスタ・メンバのカーネル auditd バッファを明示的にフラッシュして最新の監査記録を取得するには、次のようにコマンドを実行します。

```
# auditd -cluster -d
```

stdout に対する handel および haydn というメンバのログイン・イベントを表示するには、次のようにコマンドを実行します。

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

```
ruid/euid: 0/0
pid:525424 ppid:525423 cttydev: (6,1)
event:ログイン
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 15:54:180.19 1999 EDT
```

```
ruid/euid: 0/0
pid:525424 ppid:525423 cttydev: (6,1)
event:ログイン
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 15:54:260.44 1999 EDT
```

```
ruid/euid: 0/0
pid:1049810 ppid:1049809 cttydev: (6,2)
event:ログイン
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 15:54:370.74 1999 EDT

ruid/euid:   0/0
pid:1049810  ppid:1049809          cttydev: (6,2)
event:ログイン
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 監査レポートに対する対処

セキュリティ違反の兆候が見られる場合、監査の頻度を上げることを検討してください。さらに、違反の企てについての詳細情報を収集するため、監査するイベントのリストを調整することも必要です。たとえば、ファイル・システムに対する攻撃の場合、ファイルに対して成功または失敗したすべての `open`、`close`、`link`、`unlink`、`chdir`、`chmod`、`chown` およびその他のファイル関連アクセスのログを取ります。

監査トレールで、特定の承認されたユーザがセキュリティ違反を犯していると示された場合、セキュリティ違反の処理に関するサイトのセキュリティ規則に注意し、これらの規則に即した行動をとるようにします。さらに、次のような行動をとることも検討します。

- そのユーザと話し合い、セキュリティ管理の重要性と、全ユーザの協力が必要であることを説明します。
- そのユーザを専用のグループに入れ、そのユーザのシステムへのアクセスを制限します。
- 非常措置として、そのユーザのアカウントを削除し、そのユーザがシステムにアクセスできないようにします。これは一時的でも恒久的にも実行できます。

- 攻撃を加えたユーザの行いが変わったかどうかを確認するために、そのユーザの動作を監査します。監査情報が抽出するときは、そのユーザの監査 ID、UID、RUID およびユーザ名に関連する動作に注目してください。

ユーザ固有の監査は画面から次のように実行できます。「監査マネージャ」 [レポート] [レポートの生成] または、`audit_tool` のオプション `-a AUID`、`-u UID`、および `-U username` を使用します。

監査トレールにセキュリティ違反の兆候が見られるのに特定のユーザに焦点を絞れない場合は、外部から侵入されている可能性があります。このような場合の対応は、システムおよびもっと広範囲のユーザ・コミュニティに向けてする必要があります。この場合、次の手順を実行します。

- ユーザにパスワードの変更を依頼し、安全なパスワードを選択してもらうようにします。
- ユーザと会議を開き、システム・セキュリティの重要性について話し合います。
- 物理的なセキュリティを強化し、許可されたユーザだけがシステムに物理的にアクセスできるようにします。
- ファイル・システムのバックアップを取る頻度を高くします。これにより、侵入が発生し、システムのデータが失われるか変更されても、被害を最小限に抑えることができます。
- ネットワーク経由で攻撃されていると思われる場合は、ネットワーク関連動作の監査を強化します。

---

## エンハンスト・セキュリティ

この付録には、次の情報があります。

- エンハンスト・セキュリティのインストール
- エンハンスト・セキュリティの有効化
- エンハンスト・セキュリティ・データベース
- エンハンスト・セキュリティ・データベースの管理ユーティリティ
- エンハンスト・セキュリティおよびユーザの認証
- エンハンスト・セキュリティおよびNIS
- TruCluster でのエンハンスト・セキュリティ
- デバイスの安全確保
- エンハンスト・セキュリティのトラブルシューティング

### A.1 エンハンスト・セキュリティのインストール

エンハンスト・セキュリティをインストールするには、オプションのエンハンスト・セキュリティ・サブセット (OSFC2SECnnn および OSFXC2SECnnn) をインストールする必要があります。

エンハンスト・セキュリティ・サブセットをインストールする前に、万が一のために、ルート・ファイル・システムのバックアップ・コピーを作成してください。バックアップを実行するには、次のいずれかのコマンドを使用します (dump は UFS ファイル・システムでのみ使用可能)。

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

または

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

ご使用のシステム環境に合わせてテープ・デバイス名を変更してください。ファイル・システムのバックアップの詳細については、『システム管理ガイド』を参照してください。

次の方法で、エンハンスト・セキュリティ・サブセットをインストールできます。

- Tru64 UNIX (アドバンストまたはベーシック) オペレーティング・システム・ソフトウェアのフル・インストール時。フル・インストールでは、root アカウントでシステムが起動されます。アカウントを追加する前に `secconfig` スクリプトを実行してください。
- Tru64 UNIX の旧バージョンからオペレーティング・システム・ソフトウェアからシステムをアップデートする場合は、アップデート・インストール時。アップデート・インストール時、すべてのユーザ・アカウントとデータベースが保持されます。`secconfig` プログラムを実行すると、これらのデータがエンハンスト・セキュリティ形式に変換されます。

セキュリティ・サブセットをインストールすると、次のようなメッセージが表示されます。

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

パスワードの平凡性チェックを有効にする場合は、`OSFDCMTEXTnnn` サブセットもインストールする必要があります。サブセットのインストールの詳細については、『インストール・ガイド』を参照してください。

`nnn` は、Tru64 UNIX のバージョン番号です。現在のバージョン番号は、『リリース・ノート』を参照してください。

## A.2 エンハンスト・セキュリティの有効化

`secconfig` ユーティリティは対話型プログラムであり、システムのセキュリティ・レベル(「BASE」と「ENHANCED」)を有効にすることができます。システムがマルチユーザ・モードのときでも、このプログラムを実行することができます。ただし、選択したセキュリティ機能によっては、`secconfig` が完了するときにセキュリティ機能を変更する必要があります。その場合はシステムをリブートする必要があります。

エンハンスト・セキュリティを有効にするには、次の手順を実行します。

1. エンハンスト・セキュリティ・サブセット (`OSFC2SECnnn` および `OSFXC2SECnnn`) がインストールされていることを確認します。

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

*nnn* は、サブセットの現在のバージョン番号を表す数値です。現在のバージョン番号は、『リリース・ノート』を参照してください。

2. `secconfig` ユーティリティを使い、次の方法でエンハンスド・セキュリティを有効にします。
  - `/usr/sbin/sysman secconfig` コマンドを入力し、セキュリティ・レベルの入力が求められたら、ENHANCED セキュリティを選択します。
  - CDE を使い、「アプリケーション・マネージャ」「システム管理」「システム設定」「セキュリティ設定」を選択します。「セキュリティ設定」を選択すると、エンハンスド・セキュリティが有効になります。

エンハンスド・セキュリティを有効にする際の留意点については、A.2.1 項を参照してください。

3. システムをシングルユーザ・モードにしてリブートします (シャットダウン・メッセージにより、すぐにパスワードを変更する必要があることをユーザに伝える必要があります)。

保護パスワード・デーモン (`/usr/bin/prpasswd`) があるかどうかで、エンハンスド・セキュリティが有効になっているかどうか分かります。

## A.2.1 エンハンスド・セキュリティを有効にする際の留意点

以下の項では、エンハンスド・セキュリティを有効にする際の留意点について説明します。

### A.2.1.1 NIS の使用

システムが NIS (Network Information Service) で提供されるパスワード・データベースを使用する場合、NIS サーバのパスワード・データベースに登録されているユーザごとに、ローカルのエンハンスド認証プロファイルを作成するよう求められます。その後に変更した NIS パスワードは、データベースに伝達されません。ローカル・システムにあるエンハンスド・パス

ワードは期限切れとなり、ユーザは次のログイン時に新しいパスワードを入力する必要があります。

セキュリティ・レベルを基本セキュリティに戻すと、エンハンスド認証プロファイルのファイルはそのまま残されます。再度 ENHANCED セキュリティに戻すと、エンハンスド認証プロファイルのファイルがありそれにパスワードがある限り、エンハンスド・パスワードは更新されます。

エンハンスド・セキュリティと NIS の使用についての詳細は、A.6 節を参照してください。

#### A.2.1.2 セグメントの共用

シェアード・ライブラリではページ・テーブル共用メカニズムが使用されるため、通常のファイル・システムのアクセス許可では、承認されていない読み込みから保護することはできません。たとえば、ユーザ joe が次のシェアード・ライブラリを所有しているとします。

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

このシェアード・ライブラリをプログラム内で使用している場合、foo.so のテキスト部分は、ユーザ joe として実行していなくても、他の実行中のプロセスから見えてしまいます。ライブラリのテキスト部分のみこの方法で共用されます。データ・セグメントは共用されません。

すべてのセグメンテーションを無効にして承認されていない共用を回避するには、secconfig でセグメントの共用を無効にするか尋ねられたときに “yes” と応えてください。セグメントの共用がすでに無効のときは、secconfig スクリプトがそのことを報告します。

---

#### 注意

---

セグメントの共用を無効にすると、メモリの使用量が大きく増加します。

---

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

execute bit set only by root オプションを有効にすると、root ユーザのみがファイルの実行許可を設定できるようになります。

### A-4 エンハンスド・セキュリティ



## A.2.2 エンハンスト・セキュリティの構成

エンハンスト・セキュリティでは、ユーザ、端末、およびデバイスに適用するシステム省略時設定を指定できます。後述の項に、よく使用する省略時の設定とその構成方法について説明します。

システム省略時設定は、`/etc/auth/system/default` の省略時設定データベースに保存されます。このデータベースには、4 種類のフィールドがあります。

- 省略時設定データベースのみに存在するシステム単位のフィールドこれらのフィールドには `d_` という接頭辞が付きます。
- ユーザ省略時設定フィールド この値は、ユーザのプロファイルの対応するフィールドによって置き換えることができます。これらのフィールドには `u_` という接頭辞が付きます。
- 端末制御フィールド この値は、端末制御データベースの対応するフィールドによって置き換えることができます。これらのフィールドには `t_` という接頭辞が付きます。
- デバイス割り当てフィールド この値は、デバイス割り当てデータベース・ファイルの対応するフィールドによって置き換えることができます。これらのフィールドには `v_` という接頭辞が付きます。

次のインタフェースを使い、エンハンスト・セキュリティ関連のユーザ、端末、デバイス設定の省略時の値を変更します。

- `dxaccounts` GUI では、「Local Templates」 「Default」を選択して、ユーザ用の省略時設定フィールドを変更します。
- `dxdevices` の GUI では、デバイス用の省略時設定フィールドを変更します。
- `edauth` ユーティリティにより、省略時設定フィールドすべてに下位レベル・インタフェースが提供されます。

ファイル形式およびフィールド値については `authcap(4)` を、`edauth` の使用については `edauth(8)` を、各種フィールドの説明とそれぞれの値の意味については `default(4)`, `prpasswd(4)`, `ttys(4)`, および `devassign(4)` を参照してください。

#### A.2.2.1 有効期間

システム上でパスワードの有効期間の設定を行わない場合、default データベースで `u_exp` および `u_life` を 0 に設定します。パスワード長の制限を決定する省略時の方法はパスワードの有効期間に基づいているため、さらに `u_minchosen` および `u_maxchosen` をサイトにとって適切な値に設定します。

エントリの例を次に示します。

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

#### A.2.2.2 最短変更時間

`u_minchg` フィールドを次のように 0 にすると、最短の変更時間間隔を削除できます。

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

これにより、ユーザはパスワードを変更した後、すぐにまたパスワードを変更できるようになります。

#### A.2.2.3 変更制御

サイトのニーズに応じて、パスワード変更制御を構成することができます。default データベースに次のフィールドを入力すると、ユーザにパスワードの変更方法を選択させることができます。

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

(上記フィールドのうち、`u_pwdepth` は数値、それ以外は論理値となります。論理値フィールドは、指定されていれば真、その後に @ があれば偽となります。)

#### A.2.2.4 ログインの最大試行回数

侵入検出のために、連続的なログイン失敗の回数がカウントされ、ユーザ (`u_maxtries`) や端末 (`t_maxtries`) ごとに設定された最大回数と比較されます。最大回数を超えた場合、`u_unlock` や `t_unlock` で指定した時間の間、そのユーザ・アカウントまたは端末でのログインが無効となります。ユーザ・アカウントに対する侵入保護を無効にするには、`u_maxtries` を 0 にします。端末に対する侵入保護を無効にするには `t_maxtries` を 0 にします。ユーザに対する default データベースのエントリの例を次に示します。

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

#### A.2.2.5 ログイン操作の時間間隔

省略時の侵入保護時間 (86400 秒つまり 24 時間) がサイトに適していない場合, `u_unlock` フィールドをサイトに適した値 (`u_maxtries` 上限に達したときに, 直前のログイン失敗後に正常ログインが認識されるまでの秒数) に変更します。 `u_unlock` フィールドを 0 (`:u_unlock#0:`) に設定すると, ログイン試行間隔の時間は無限 (自動的に再度使用可能になることはない) に設定されます。 `t_maxtries` を使用すれば, 端末に対しても同じような動作を制御できます。

#### A.2.2.6 ログインの時間間隔

`default` データベースの `u_max_login_intvl` フィールドにより, システム全体のログイン間隔の最大許容時間を設定することができます。

端末に対するシステムの省略時設定のログイン・タイムアウトは, `default` データベースの `t_login_timeout` フィールドで変更できます。 `ttys` データベースの \* エントリでも設定することができます。 X windows では, このフィールドを 0 (無限) にする必要があります。

#### A.2.2.7 端末ごとのログイン・レコード

端末ごとの正常ログインおよびログイン失敗を記録しない場合, `default` データベースの `d_skip_ttys_updates` 論理フィールドを次のとおり設定します。

```
:d_skip_ttys_updates:\
```

この場合, 端末ごとの侵入保護が無効になる副作用があります。

#### A.2.2.8 正常ログインのロギング

厳密な C2 セキュリティでは, 正常ログインのロギングが必要になります。このロギングを無効にするには, `d_skip_success_login_log` 論理フィールドを次のように設定します。

```
:d_skip_success_login_log:\
```

#### A.2.2.9 ログイン失敗のロギング

通常, ユーザ・アカウントへのログイン失敗は, 記録されます。このロギングを無効にするには, `d_skip_fail_login_log` 論理フィールドを次のよ

うに設定します。これにより、システム全体での侵入検出および侵入保護も無効になります。

```
:d_skip_fail_login_log:\
```

**A.2.2.10 エンハンスト・プロファイルの自動作成**

d\_auto\_migrate\_users 論理フィールドを設定すると、ログイン時にエンハンスト・プロファイルがなければ自動的に作成することが可能になります。これにより、従来のユーザ・プロファイルの追加方法をそのまま使用できます。

**A.2.2.11 保証機能**

d\_accept\_alternate\_vouching フィールドを設定すると、エンハンスト・セキュリティと DCE を一緒に使用することができます。

**A.2.2.12 暗号化**

crypt() を使用してパスワード確認を行うプログラムをサポートするために、ユーザ・パスワードを /etc/passwd ファイルに残したまま、エンハンスト・プロファイルの他の機能を使用する場合、/usr/sbin/sysman secconfig ユーティリティを実行する前に default データベースに次のエントリを入力します。

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

これは、<prot.h> ファイルの AUTH\_CRYPT\_C1CRYPT 値に対応しています。

**A.3 エンハンスト・セキュリティ・データベース**

表 A-1 にエンハンスト・セキュリティ・データベースを示します。

表 A-1: エンハンスト・セキュリティ・データベース

データベース	保存場所	内容
保護パスワード	/tcb/files/auth.db /var/tcb/files/auth.db	ユーザ認証データベース
システム省略時設定	/etc/auth/system/default	データベース・フィールドの省略時の値
端末制御	/etc/auth/system/ttys.db	各端末のセキュリティ情報

表 A-1: エンハンスト・セキュリティ・データベース (続き)

データベース	保存場所	内容
ファイル制御	/etc/auth/system/files	システム・ファイルごとの保護属性
デバイス割り当て	/etc/auth/system/devasign	デバイスに固有の制御

### A.3.1 エンハンスト (保護) パスワード・データベース

保護パスワード・データベースには、システム上にアカウントを持つ各ユーザのエンハンスト認証プロファイルが格納されています。各プロファイルには、次のような情報が含まれます。

- ユーザ名とユーザ ID
- 暗号化されたパスワード
- ユーザの監査属性
- パスワード生成パラメータ
- 正常ログインおよびログイン失敗の日時および端末

エンハンスト (保護) パスワード・データベースは /tcb/files/auth.db ファイルに格納されています。

エンハンスト・パスワード・データベースの内容の詳細については、prpasswd(4) を参照してください。

### A.3.2 システム省略時設定データベース

システム省略時設定データベースには、データベース・フィールドの省略時の値が格納されています。エンハンスト (保護) パスワード・データベース、端末制御データベース、またはデバイス割り当てデータベースに管理者が明示的に値を設定していない場合に、これらの省略時の値が使用されます。

システム省略時設定データベースには、以下の情報が含まれます。

- 省略時のパスワード生成パラメータ
- ユーザごとに設定される、ログイン失敗の省略時の回数

- 直接接続されている端末ごとに設定される，許されるログイン失敗の省略時の回数
- 省略時のデバイス割り当てパラメータ

システム省略時設定データベースは `/etc/auth/system/default` に格納されています。詳細については，`default(4)` を参照してください。

### A.3.3 端末制御データベース

端末制御データベースには，システムに接続された各端末のログイン処理を制御するための情報が格納されています。システムは，端末からシステムへのアクセスを制御するための追加情報としてこのデータベースを使用します。管理者はサイトの物理的なニーズおよび管理上のニーズに合わせて，端末ごとに異なるログイン・ポリシーを設定することができます。

端末制御データベースの各エントリには，次のような情報が含まれます。

- 端末のデバイス名
- その端末から最後に正常ログインしたユーザ ID およびタイム・スタンプ
- その端末で最後にログインに失敗したユーザ ID およびタイム・スタンプ
- その端末からログインする際の，ログイン間の遅延
- その端末をロックするまでに許されるログイン失敗の回数

システムのインストール完了時には，端末制御データベースにシステム・コンソールのエントリが格納されています。システム設定時にこれらの初期値を変更します。ログインするためには，これも最初にインストールされる対応するエントリがデバイス割り当てデータベースに必要になります。

端末制御データベースは `/etc/auth/system/ttys.db` に格納されています。詳細については，`ttys(4)` を参照してください。

### A.3.4 ファイル制御データベース

ファイル制御データベースには，システム・ファイル（つまりエンハンスト・セキュリティの動作に必要なファイル）の保護属性の情報が格納されています。このデータベースはシステムの完全性を維持するのに役立ちます。システム・ファイルごとに 1 つのエントリが含まれます。

ファイル制御データベースの各エントリには，次のような情報が含まれます。

- ファイルの完全パス名
- ファイルの所有者およびグループ
- ファイルのモードおよびタイプ
- 承認可能な特権セットおよび承認された特権セット
- アクセス制御リスト

システムのインストールが完了したときには、ファイル制御データベースにセキュリティ関連の全システム・ファイルのエントリが格納されています。システム設定時にはこのデータベースを変更する必要はありませんが、まれにシステム動作時にこのデータベースのアップデートを行う必要があります。

/etc/auth/system/files ファイルの内容の詳細については、files(4) を参照してください。

### A.3.5 デバイス割り当てデータベース

デバイス割り当てデータベースには、ユーザとのデータ交換に使用するデバイスについての情報が格納されています。ログイン端末ごとに 1 つのエントリが、デバイス割り当てデータベース内に必要です。システムは、システム・デバイスで送受信されるデータのセキュリティ属性を制限するのにこのデータベースを使用します。

デバイス割り当てデータベースの各エントリには、デバイスを記述する情報と、デバイスのパス名を適切な物理デバイスに対応付けるための情報が含まれます。これは、いくつかの異なるパス名が同じ物理デバイスを参照することがあるため必要となります。たとえば、2 つのパス名が同じシリアル・ポートに対応することがあります。一方のパス名ではモデム制御を有効にし、もう一方のパス名ではモデム制御を無効にすることが考えられます。

デバイス割り当てデータベースの各エントリには、次のような情報が含まれます。

- デバイスのパス名
- 同じ物理デバイスに対応する他のパス名
- デバイスの種類

ログイン端末を参照するエントリには、対応するエントリが端末制御データベースに必要です。

デバイス割り当てデータベースは `/etc/auth/system/devassign` に格納されています。詳細については、`devassign(4)` を参照してください。

## A.4 エンハンスト・セキュリティ・データベースの管理ユーティリティ

Tru64 UNIX には、Berkeley データベース (Berkeley DB) のカスタム・バージョンが組み込まれており、重要なセキュリティ・ファイルに対する高性能なデータベース・サポートを提供します。DB 機能には、事前書き出しのロギングおよびチェックポイントを使用して変更内容を記録する、完全なトランザクション・サポートおよびデータベース・リカバリが含まれます。重大な障害が発生した場合、データベース・ファイルをリストアし、ログをロール・フォワードすることにより、セキュリティ・データベースは直前の一貫性のあるトランザクション状態に復元できます。

エンハンスト・セキュリティには、以下のデータベース管理ユーティリティが含まれており、`/usr/tcb/bin` ディレクトリに格納されています。

<code>db_archive</code>	アクティブなトランザクションで使用されなくなったエンハンスト・セキュリティ・データベース・ログ・ファイルを表示します。 <code>/var</code> の領域を回復するために、このファイルを安全にバックアップして、削除できます。
<code>db_checkpoint</code>	メモリの内容を書き出し、ログにチェックポイント・レコードを書き込んで、ディスクにログを書き出す。
<code>db_load</code>	ファイルまたは標準入力から読み込み、データベースにロードします。
<code>db_unload</code>	データベースをファイルにアンロードします。
<code>db_stat</code>	セキュリティ・データベースの統計情報を表示します。
<code>db_recover</code>	予期しない障害が発生した後、データベースを矛盾しない状態に復元します。



通常、セキュリティ・データベースはインストレーション・ユーティリティでのみロードまたはアンロードされます。データベースは、データベース管理作業を軽減するように設計されていますが、セキュリティ・データベースのログ・ファイルを追加すると、ログ・ファイルが拡大して /var の空き領域がなくなる可能性があります。このため、セキュリティ構成ユーティリティには、cron ジョブを作成して、アクティブなトランザクションで使用されなくなったログ・ファイルを定期的に削除するオプションがあります。

## A.5 エンハンスト・セキュリティおよびユーザの認証

エンハンスト・セキュリティを稼働させているシステムは、/etc/passwd ファイルとエンハンスト・セキュリティ・データベースを使ってユーザの認証を行います。エンハンスト・セキュリティ・データベースは以下のデータベースから構成されています。

- 保護パスワード・データベース (/tcb/files/auth.db および /var/tcb/files/auth.db)
- システム省略時設定データベース (/etc/auth/system/default)
- 端末制御データベース (/etc/auth/system/ttys.db)
- ファイル制御データベース (/etc/auth/system/files)
- デバイス割り当てデータベース (/etc/auth/system/devassign)

エンハンスト・セキュリティの認証データベースには、/etc/passwd に定義されているユーザ・アカウントごとに 1 つのエントリがあります。エンハンスト・セキュリティでは、暗号化されたパスワードを除き /etc/passwd ファイルは変更されていません。暗号化されたパスワードは /etc/passwd ファイルから auth.db データベースに移動しています。/etc/passwd ファイルの他のフィールド(シェルなど)はそのまま /etc/passwd ファイルにあり、通常の方法で使用されます。

エンハンスト・セキュリティの認証データベースでは、ユーザ名および UID によって各ユーザを一意に識別します。これらの情報は /etc/passwd ファイルのユーザのエントリと一致する必要があります。暗号化されたパスワード以外に、エントリにはエンハンスト・セキュリティでのみ使用する一連のフィールドおよび値が含まれます。これらのフィールドの説明については prpasswd(4) を、ファイル形式の説明については authcap(4) を参照してください。

ユーザ・アカウントはテンプレート・アカウントと関連付けることができます。このテンプレート・アカウントを使用すると、ユーザのグループに省略時の値を設定することができます。アカウントは最終的に、`/etc/auth/system/default` ファイルに保存されているシステム省略時設定テンプレートの値と関連付けられます。

エンハンスド・セキュリティを使用している場合、ユーザは `passwd` コマンドの使用を続行して自分のパスワードを変更します。

### A.5.1 ユーザ・プロフィール

エンハンスド・セキュリティの認証データベースにあるユーザのエントリは、ユーザ・プロフィールと呼ばれます。セキュリティ対応プログラムは、プロフィール内のフィールドおよび値を解釈します。ユーザ・プロフィールにはすべてのフィールドを含める必要はありません。フィールドがユーザ・プロフィールに指定されていない場合、システムはそのユーザに対応付けられたテンプレート・アカウントを参照し、最終的にシステム省略時設定テンプレートを参照して、フィールドの値を見つけます。

値は次のように取得されます。

- ユーザ・プロフィールにユーザ固有の値があれば、その値を使用します。
- ユーザ・プロフィールにテンプレート・アカウントへの参照があり、ユーザ固有の値が定義されていない場合、テンプレート・アカウントの値が使用されます。
- フィールドの値がユーザ・プロフィールとテンプレート・アカウントのいずれにも定義されず、システム省略時設定テンプレートにそのフィールド値が定義されている場合、システム省略時設定テンプレートの値が使用されます。
- 値がどこにも定義されていない場合、そのフィールドに対する静的なシステム省略時設定が使用されます。

システム省略時設定テンプレートの値は `/etc/auth/system/default` ファイルにあり、`dxaccounts` の「View Local Template」オプションまたは `edauth` ユーティリティを使って変更することができます。他のテンプレート・アカウントは `auth.db` データベースに保存されます。テンプレート・アカウントには、`/etc/passwd` に対応するエントリがないことに注意してください。

#### A.5.1.1 /etc/passwd 情報の回復

/etc/passwd ファイルが失われたとしてもエンハンスト・プロファイルがあれば、次のコマンド・シーケンスを使用して、失われたデータの一部を回復することができます。

```
# 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
```

これにより、次のようなエントリを含む psw.missing ファイルが作成されます。

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

1 次グループ情報、finger 情報、ホーム・ディレクトリ、およびログイン・シェルはエンハンスト・プロファイルに記録されません。これらのフィールドのデータは、別の方法で回復する必要があります。

#### A.5.2 エンハンスト・セキュリティ認証データベースの完全性チェック

/usr/tcb/bin/authchk コマンドを使って、エンハンスト・セキュリティ認証データベースの全体構成および内部の一貫性をチェックします。authchk コマンドは、各データベースのエントリの正当性をチェックし、他のデータベースの関連フィールドのチェックも行います。たとえば、ユーザの保護パスワード・データベース・エントリを /etc/passwd ファイルと照合します。

authchk コマンドは、データベース間の矛盾をリストしたレポートを生成します。プログラムの出力結果と実際のデータベースのエントリを比較し、矛盾点を直ちに修正してください。問題が発生するのは、通常、データベースの 1 つを手作業で更新し、関連するデータベースに必要な変更を行わなかった場合です。

authchk コマンド行では、次の引数を指定することができます。

- p      保護パスワード・データベースおよび /etc/passwd ファイルをチェックし、これらが完全で、矛盾していないことを確認します。また、保護パスワード・データベースに適切な値が設定されているかチェックします。
- t      端末制御データベースのフィールドに適切な値が設定されているかチェックします。

- f      ファイル制御データベースの構文および指定値の誤りをチェックします。このフラグを指定しない場合、エントリに不明な承認指定やユーザ名などが指定されていても無視されます。通常このような誤りは、“root”の代わりに“rooot”と入力したといった入力ミスが多いため、プログラムでは正しい値を推測しようとしています。
- v      冗長モード。
- a      -f, -p, -t, および -v の機能を実行します。実行中にプログラムの処理ステータスが表示されます。

詳細については、authck(8) を参照してください。

### A.5.3 ファイル制御データベースへのアプリケーションの追加

setld プログラム以外の方法でシステムにアプリケーションを追加する場合、アプリケーションの制御ファイルとデータベース・ファイルおよびプログラムについて、ファイル制御データベースのエントリを追加する必要があります。アプリケーション供給元に相談し、ファイルおよびプログラムの一覧、そして全ファイルに推奨される保護属性を入手することをお勧めします。

アプリケーションのファイルをファイル制御データベースに追加すると、アプリケーションのリソースの完全性を定期的にチェックすることができます。

ファイルの完全性チェックについての詳細は、fverify(8) を参照してください。

## A.6 エンハンスト・セキュリティと NIS

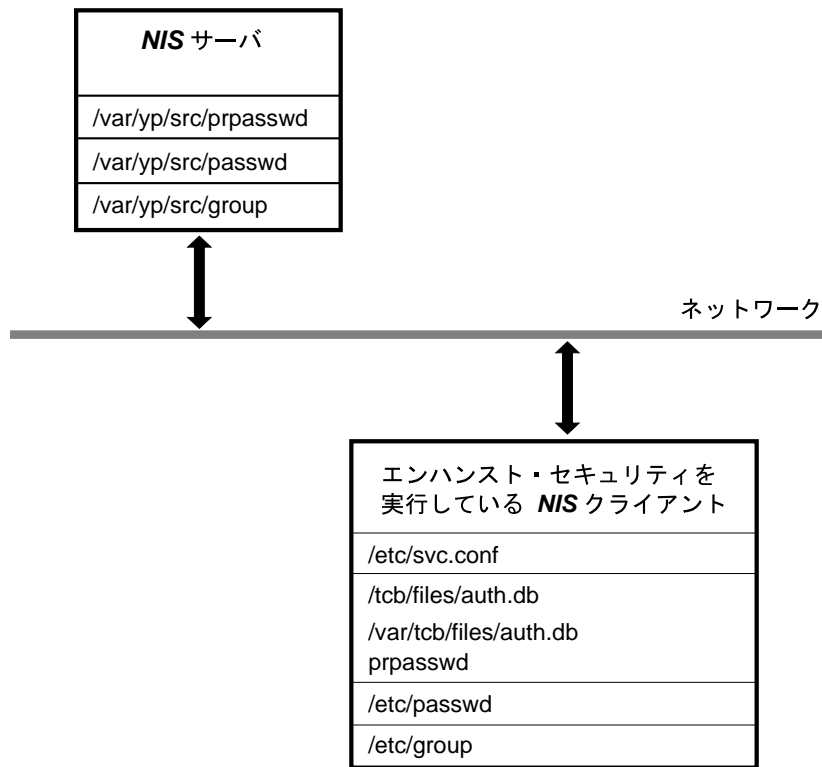
NIS (Network Information Service) を使用すると、エンハンスト (保護) パスワード・データベースの一部または全部を配布できます。同様に、BSD ユーザ・アカウント・データベースやグループ・データベースを配布することもできます。Tru64 UNIX の NIS マスタは、エンハンスト・セキュリティが稼働している Tru64 UNIX システムである NIS クライアントのほか、他メーカーのシステムを含め、BSD 認証を使う NIS クライアントにサービスを提供できます。

エンハンスド・セキュリティでNISを使用すると、以下のユーザ・アカウント・データベースを持つことになります。

- NIS マスタ・サーバ上：
  - /var/yp/src/passwd および /var/yp/src/group ファイルから生成され、ndbm または btree マップとして配布される NIS 配布型の基本ユーザ・アカウント・データベース。
  - /var/yp/src/prpasswd ファイルから生成され、btree マップとして配布される NIS 配布型のエンハンスド・セキュリティ・ユーザ・アカウント・データベース。
- NIS クライアント上：
  - /etc/passwd および /etc/group ファイルにあるローカルの基本ユーザ・アカウント・データベース
  - ローカルのエンハンスド・セキュリティ・ユーザ・アカウント・データベース

図 A-1 に、NIS マスタ・サーバおよびクライアント上のユーザ・アカウント・データベースを示します。

図 A-1: NIS およびエンハnst・セキュリティ・ファイル



ZK1087UAIJ

/etc/svc.conf ファイルの auth= エントリにより，ローカルのユーザ・アカウント・データベースおよび NIS エンハnst・セキュリティ・ユーザ・アカウント・データベースでユーザ・エントリを検索する順序を指定します。ローカルが先か，NIS (yp) が先かのいずれかになります。

/etc/passwd エントリでの + 符号による置き換え機能も同じように動作します。

#### 注意

NIS を使用した基本セキュリティ・システムをエンハnst・セキュリティ・システムにアップグレードする際，Create Entries for NIS Users という質問に yes と答えると，/usr/sbin/sysman secconfig ユーティリティは NIS ユーザ (/etc/passwd ファイ

ルの +username エントリ) に対して auth.db エントリのみを作成します。

---

エンハンスト・セキュリティ・ユーザ・アカウント・データベースには置き換え機能がありません。ユーザのプロファイルは、ローカル・データベースと NIS 配布データベースのいずれかにすべて含まれます。NIS アカウント用のテンプレートを定義し、NIS エンハンスト・セキュリティ・マップの一部として配布することはできますが、NIS はシステム省略時設定のテンプレート (/etc/auth/system/default) を配布しません。このテンプレートにより、ユーザのプロファイルに指定されていないフィールドに対して最終省略時設定が提供されます。したがって、エンハンスト・セキュリティでは、NIS クライアントはそれぞれの /etc/auth/system/default ファイルを使用して、ローカルのユーザ・プロファイルおよび NIS ユーザ・プロファイル両方の最終省略時設定を取得します。クライアントのシステム省略時設定ファイルに NIS マスタと異なる値が設定されている場合、予期しない動作をすることがあります。

passwd コマンドは、ユーザのローカル・エントリまたは NIS エンハンスト・セキュリティ・エントリにあるパスワードを変更します。yppasswd コマンドは、NIS 配布型の基本ユーザ・アカウント・データベースのフィールドを通常どおりに変更します。

NIS ユーザ・アカウントを変更するには、dxaccounts の「View NIS User」オプションを使用するか、または useradd、usermod および userdel ユーティリティに -x distributed=1 local=0 オプションを指定して実行します。

NIS の詳細については、『ネットワーク管理ガイド：サービス編』を参照してください。

### A.6.1 NIS アカウント用テンプレート

/var/yp/src/prpasswd ファイルは、NIS で配布されるエンハンスト・セキュリティ・ユーザ・アカウントのソースです。このファイルには、テンプレート・プロファイルと通常のユーザ・プロファイルを含めることができます。ローカルのユーザ・プロファイルと同様、NIS ユーザ・プロファイルにすべてのフィールドを含める必要はありません。NIS ユーザのプロファイルに指定されていないフィールドがある場合、システムはフィールドの値が見

つかるまで、そのユーザに対応する NIS テンプレート・アカウントを参照し、最終的にローカルのシステム省略時設定テンプレートを参照します。

値は次のように取得されます。

- ユーザ・プロファイルにユーザ固有の値があれば、その値を使用します。
- ユーザ・プロファイルにテンプレート・アカウントへの参照があり、ユーザ固有の値が定義されていない場合、テンプレート・アカウントの値が使用されます。
- フィールドの値がユーザ・プロファイルとテンプレート・アカウントのいずれにも定義されず、システム省略時設定テンプレートにそのフィールド値が定義されている場合、システム省略時設定テンプレートの値が使用されます。
- 値がどこにも定義されていない場合、そのフィールドに対する静的なシステム省略時設定が使用されます。

NIS テンプレート・アカウントを変更するには、`dxaccounts` の「View NIS Template」オプションまたは `edauth` ユーティリティを使用します。

システム省略時設定テンプレートの値は、NIS クライアントの `/etc/auth/system/default` ファイルに格納されています。NIS では、システム省略時設定テンプレートは配布されないことに注意してください。NIS クライアントは自分の `/etc/auth/system/default` ファイルを使用して、ローカルのユーザ・プロファイルと NIS ユーザ・プロファイルの両方の最終省略時設定を取得します。クライアントのシステム省略時設定ファイルに NIS マスタと異なる値が含まれる場合、予期しない動作をすることがあります。

## A.6.2 エンハンスト・セキュリティ環境での NIS マスタ・サーバの構成

エンハンスト・セキュリティ環境で NIS マスタを構成するには、次の手順を実行します。

1. マスタ・サーバ上で NIS が動作している場合は、次のように停止します。  

```
# /sbin/init.d/nis stop
```
2. エンハンスト・セキュリティ・サブセットがインストールされているか確認します。
3. システム省略時設定テンプレートを、以下のように変更します。



```
# edauth -dd default
```

次のフィールドを設定します。

```
d_skip_success_login_log:
d_skip_ttyupdates:
```

4. /var/yp/src/hosts , /var/yp/src/passwd ,  
/var/yp/src/group , および /var/yp/src/prpasswd  
ファイルを作成します。
5. sysman nis プログラムを実行します。
  - a. sysman nis プログラムによって最初にセキュリティ (ypbind の -s オプション) の入力が必要であれば、y を選択して ypbind -s を実行し、セキュア・ソケットを指定します。
  - b. sysman nis プログラムによって再度セキュリティ (ypbind の -S オプション) の入力が必要であれば、y を選択し、ドメイン名 1 つと承認されたスレーブ・サーバを 4 つまで指定します。
6. /etc/svc.conf ファイルに次のエントリがあることを確認します。

```
auth=local,yp
```
7. NIS を起動します。

```
# /sbin/init.d/nis start
```

#### A.6.2.1 手作業:小規模ユーザ・アカウント・データベースのマップ

エンハンスド・セキュリティを使ってクライアントをサポートする NIS マスタ・サーバでは、手作業での設定が最適です。dxaccounts プログラム、または代わりに adduser、addgroup、useradd、userdel および usermod コマンドを使用して、アカウント・マップを設定します。アカウントを設定する別の方法については、A.6.5 項を参照してください。

#### A.6.2.2 自動処理:大規模ユーザ・アカウント・データベースのマップ

既存の NIS 配布型の基本ユーザ・アカウント・データベースが大規模な場合、次のコマンドを入力して、NIS 配布型のエンハンスド (保護) パスワード・データベースを自動で作成できます。

```
# convuser -Mc
```

別の方法として、/var/yp/src/prpasswd ファイルを作成して次のコマンドを実行しても、マップを作成することができます。

```
# /usr/tcb/bin/edauth -Lg > /var/yp/src/prpasswd
# cd /var/yp; make prpasswd
```

### A.6.3 エンハnst・セキュリティ環境での NIS スレーブ・サーバの設定

NIS がスレーブ・サーバ上で稼働している場合、`/sbin/init.d/nis stop` コマンドを使って NIS を停止する必要があります。次の設定手順は、エンハnst・セキュリティを使用するクライアントをサポートする NIS スレーブ・サーバに固有のものです。

1. エンハnst・セキュリティ・サブセットがインストールされていることを確認します。
2. システム省略時設定テンプレートを、以下のように変更します。

```
# edauth -dd default
```

次のフィールドを設定します。

```
d_skip_success_login_log:
d_skip_ttys_updates:
```

3. `sysman nis` プログラムを実行します。
  - a. `sysman nis` プログラムによって最初にセキュリティ (`ypbind` の `-s` オプション) の入力 が求められたら、`y` を選択して `ypbind -s` を実行し、セキュア・ソケットを指定します。
  - b. `sysman nis` プログラムによって再度セキュリティ (`ypbind` の `-s` オプション) の入力 が求められたら、`y` を選択し、ドメイン名 1 つと承認されたスレーブ・サーバを 4 つまで指定します。
4. `/etc/svc.conf` ファイルに次のエントリがあることを確認します。

```
auth=local,yp
```
5. `/var/yp/ypxfr_1perday`、`/var/yp/ypxfr_1perhour`、`/var/yp/ypxfr_2perday` ファイルを編集し、それぞれに次の行を追加します。

```
ypxfr -a "$method" prpasswd
ypxfr -a "$method" prpasswd_nonsecure
```
6. NIS を起動します。

```
# /sbin/init.d/nis start
```

#### A.6.4 エンハンスト・セキュリティ環境での NIS クライアントの設定

NIS がスレーブ・サーバ上で稼働している場合、`/sbin/init.d/nis stop` コマンドを使って NIS を停止する必要があります。次の設定手順は、エンハンスト・パスワード・セキュリティを使用する NIS クライアントに固有のものです。

1. エンハンスト・セキュリティ・サブセットがインストールされていることを確認します。
2. 次のコマンドを使用して、システム省略時設定テンプレートを変更します。

```
# edauth -dd default
```

次のフィールドを設定します。

```
d_skip_success_login_log:
d_skip_ttys_updates:
```
3. `sysman nis` プログラムを実行します。
  - a. `sysman nis` プログラムによって最初にセキュリティ (`ypbind` の `-s` オプション) の入力が必要とされたら、`y` を選択して `ypbind -s` を実行し、セキュア・ソケットを指定します。
  - b. `sysman nis` プログラムによって再度セキュリティ (`ypbind` の `-s` オプション) の入力が必要とされたら、`y` を選択し、ドメイン名 1 つと承認されたスレーブ・サーバを 4 つまで指定します。
4. `/etc/svc.conf` ファイルを編集し、`auth` 用の `yp` エントリを含めます。エントリは、`auth=local,yp` となります。
5. `/sbin/init.d/nis start` コマンドを使って NIS を起動します。

#### A.6.5 ローカル・アカウントの NIS への移行

既存のローカル・アカウントを NIS へ移行するには、次のコマンドを実行します。

```
# edauth -Lg | edauth -NsC
```

#### A.6.6 NIS サポートの削除

NIS を削除する必要がある場合、NIS アカウントをローカル・データベースにコピーし、クライアント上で次のコマンドを実行して NIS を削除します。

```
# edauth -gN | edauth -sLC
# sysman nis
<メニューから Remove オプションを選択する>
```

クライアント上のエンハnst (保護) パスワード・データベースは、NIS データベースにあってローカル・データベースにないすべてのアカウントで更新されます。

## A.6.7 導入時の注意点

以下の情報は、エンハnst・セキュリティおよび NIS に固有です。

- エンハnst・セキュリティ環境で NIS を実行している場合にパスワードを変更するには、ローカルおよび NIS 配布型のエンハnst (保護) パスワード・データベース両方のエントリに対して、passwd コマンドを実行します。passwd コマンドは、svc.conf ファイル内の検索リスト (auth=local, yp エントリ) を使用し、エントリが NIS 配布型のエンハnst・パスワード・データベースにあるかどうかにかかわらず、エンハnst (保護) パスワード・データベースの中で最初に見つかった、指定されたユーザに対応するエントリにあるパスワードを更新します。
- エンハnst・パスワード・データベースの各エントリは、ローカルのエンハnst・パスワード・データベースまたは NIS 配布型のエンハnst・パスワード・データベースのいずれか 1 つのデータベースにのみ存在していなければなりません。エンハnst・パスワード・データベース情報の確認ルーチンおよび操作ルーチンは、見つかった最初のコピーに対して処理を実行します (svc.conf ファイルで定義)。NIS の yp ルーチンは、NIS 配布型のエンハnst・パスワード・データベースのみを対象に動作します。同じエントリが両方のデータベースにある場合、混乱が生じることがあります。このような状況が発生した場合、一方のエントリを削除してください。
- root アカウント情報は配布しないでください。クライアント・システム上でローカルの root アカウントを管理することで、NIS サーバがダウンしていても、root アカウントを使ってクライアント・システムにログインすることができます。
- 最後の正常ログインに関する情報を維持するには、ユーザがログインするたびにそのユーザのエンハnst・プロフィールを更新するように設定します。このため、NIS マスタでは、マップを再構築しスレーブへ送

信する必要があります。Tru64 UNIX Version 5.1A では、この更新をオプションとしているため、`d_skip_success_login_log` システム省略時設定フィールドの値を `true` に設定して、この更新を無効にできます。正常ログインのロギングを無効にした場合、NIS スレーブ・サーバを正しく構成すれば、NIS マスタ・サーバが常時使用可能でなくても正常にログインすることができます。

- スケーラビリティの改善には次のものがあります。
  - 単一のエントリを更新しても、`prpasswd` マップ全体が再構築されないことがあります。可能ならば、マップ・エントリは直接更新されます。

手作業による再構築後に `prpasswd` ファイルに対して手作業で変更を行わないでください。`rpc.yppasswdd` デーモンがエントリをキャッシュしているために、手作業による変更が置き換えられてしまう可能性があるからです。`prpasswd` ファイルを変更する必要がある場合は、NIS サーバを停止してから変更を行い、その後 NIS サーバを再起動します。
  - 正常ログインのロギングが有効の場合、正常ログインは NIS マップの配布が完了する前に完了します。NIS マスタが更新されたことを確認するまで待つだけです。ログイン失敗のロギングを有効にすると、失敗したログインは NIS マップからスレーブ・サーバへの配布が完了するまで待つて完了します。これはセキュリティの問題およびタイミングの問題に対処するために必要です。
- NIS マップのデータベース形式は構成可能です。`ndbm` の他に、`btree` または `hash` を選択することができます。NIS マップの保存に `ndbm` を使用する場合、保存できるアカウントのレコード数に制限があり、その数はアカウント名と UID の組合せに依存します。上限は通常約 30,000 エントリですが、アカウント名と UID の組合せによっては 10,000 エントリ以下となる場合もあります。`ndbm` にはこのような制約があるため、特にエンハンスド・セキュリティを使用する場合、データベース形式に `btree` を使用します。
- NIS サーバは、共通のデータベース形式で動作させるのが最適です。スレーブ・サーバにマスタとは別の形式を定義した場合 (`btree` ではなく `ndbm` など)、マップをスレーブ・サーバにプッシュする時間が大幅に増加します。これは、スレーブ・サーバが、マスタから単一の要素と

してデータベースを受信するのではなく、データベースを一度に 1 要素ずつ再構築しなければならないためです。

- NIS スレーブが NIS マスタ上の `ypservers` NIS マップにリストされていない場合、これらのスレーブに対応する NIS クライアントで性能上の問題が発生することがあります。この問題を解決するには、NIS マスタ上の `ypservers` NIS マップにすべての NIS スレーブを定義します。そして、スレーブ・サーバ上で次のコマンドを実行し、NIS マスタからユーザ・アカウント・データベースを取り込みます。

```
# /var/yp/ypxfr -d 'domainname' -h NISMASTER -c prpasswd
# /var/yp/ypxfr -d 'domainname' -h NISMASTER -c
prpasswd_nonsecure
```

この例では、`NISMASTER` を使用する NIS マスタ・サーバの名前に置き換えてください。このコマンドでは、これらのスレーブ・サーバに対するマップの最初のコピーが送信されます。

- ログインに失敗したログイン・プロセスでは、`prpasswd` マップで直前のログイン失敗の情報を確認する必要があります。これには、最新の `prpasswd` マップが必要です。このため、ログイン失敗が発生するたびに `prpasswd` マップに対して `yppush` 操作を実行する必要があります。このマップは (少なくとも) `rpc.yppasswdd` デーモンの通常運用時にプッシュする必要があります。このような構成では、`/var/yp/Makefile` の変数 `NOPUSH` を設定することはお勧めできません。
- `prpasswd` 情報の共有に NIS を使用できないサイトでは、NFS を使用して `/tcb/files` および `/var/tcb/files` ディレクトリを共有することができます。この場合、必要に応じて `-root=client1:client2:client3` または `-root=0` を用いて、関連ノードに root アクセス権付きでディレクトリをエクスポートする必要があります。 `exports(4)` を参照してください。データベースが破損しないように、NFS ロックを有効にする必要もあります。

## A.6.8 NIS のトラブルシューティング

表 A-2 では、一般的な NIS の問題および考えられる原因について説明します。

表 A-2: NIS のトラブルシューティング

問題	考えられる原因
ローカル・アカウントにはログインできるが、どの NIS アカウントにもログインできません。dxaccounts ユーティリティでは、アカウントが存在しロックされていないと表示されます。	1. /etc/svc.conf ファイルをチェックし、auth=local,yp 行があることを確認します。 2. /etc/passwd ファイルをチェックし、ファイルの最終行に "+:" があることを確認します。
ブート時に、スレーブ NIS サーバが更新済みの prpasswd マップを取得できません。	<pre>/var/yp/ypxfr_1perday , /var/yp/ypxfr_1perhour およ び /var/yp/ypxfr_2perday ファイル をチェックし、それぞれに次の行がある ことを確認します。  ypxfr -a "\$method" prpasswd ypxfr -a "\$method" prpasswd_nonsecure</pre>
dxaccounts プログラムの [表示] ポップアップ・メニューに、NIS ユーザ・アカウント・データベースのオプション ([NIS Users], [NIS Groups], [NIS Templates] など) が表示されません。	NIS が稼働していないか、または構成されていません。
<pre>/var/yp ディレクトリで make コマンド を実行すると、Map 'ypslaves' is empty for domain 'domainname' というメッセージが表示されます。</pre>	これは情報メッセージなので、対処は必要ありません。
<pre>/var/yp ディレクトリで make コマンドを実行すると、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! というメッセージが表示されま す。</pre>	ホスト・マップが存在しません。次のコマンドを実行します。 <pre># touch /var/yp/src/hosts # cd /var/yp # make</pre>

## A.7 TruCluster でのエンハンスト・セキュリティ

TruCluster は、単一のセキュリティ・ドメインです。特に指定しなければ、各メンバに対して、識別と認証、アクセス制御リスト (ACL)、および監査

が全く同じに構成され、ユーザおよびシステム管理者に対して一貫したインタフェースが提供されます。

1 つの認証ファイルがクラスタの全メンバ間で共用されるため、各ユーザ・アカウントはすべてのクラスタ・メンバで有効です。このためユーザは、どのメンバが接続を受け入れたかを意識することなく、クラスタにログインすることができます。同一構成の ACL チェックにより、認証とファイル・アクセス制御の一貫性が確保されます。ユーザは、どのメンバでも同じアクセス権を持ちます。クラスタ単位の監査設定により、クラスタの動作を統一的に取得できるようになります。

#### **A.7.1 TruCluster での基本セキュリティからエンハンスド・セキュリティへのアップグレード**

既存の TruCluster を基本セキュリティからエンハンスド・セキュリティへアップグレードするときには、クラスタ全体をリブートする必要があります。このアップグレードにより、ユーザ・アカウントが `/etc/passwd` ファイルから `auth.db` データベースへコピーされ、`/etc/passwd` ファイルのパスワードが削除され、新しいセキュリティ・ライブラリに切り替わります。`telnet` セッションのような、ユーザ名とパスワードを認証する新しいプロセスは、新しいライブラリを使って新しいデータベースにアクセスします。ただし、`dtlogin` セッションや CDE ウィンドウのロックのような既存のプロセスは、元のライブラリを続けて使います。このライブラリは、パスワードが削除された `/etc/passwd` ファイルにアクセスすることを想定しているため、既存のプロセスでのパスワード確認は必ず失敗します。特に、コンソール・ログイン・ウィンドウでこの問題が発生すると、`root` アカウントが無効になっているものと誤って解釈されてしまうことがあります。クラスタのリブートにより、この状態を防ぐことができます。

#### **A.7.2 TruCluster でのエンハンスド・セキュリティのインストールおよび構成**

TruCluster にエンハンスド・セキュリティを構成する理想的なタイミングは、TruCluster Server ライセンスおよびサブセットをロードしてクラスタを作成する前です。インストールの際に、エンハンスド・セキュリティのサブセット (`OSFC2SECnnn` および `OSFXC2SECnnn`) を選択する必要があります。



`nnn` は、Tru64 UNIX のバージョン番号です。現在のバージョン番号は、『リリース・ノート』を参照してください。

エンハンスド・セキュリティ・ソフトウェアは、次のように TruCluster のメンバにインストールされます。

- クラスタの最初のメンバに Tru64 UNIX をインストールする際には、TruCluster Server ソフトウェアをインストールする前にセキュリティ環境を完全に設定する必要があります。設定したセキュリティ構成は、他の構成データとともに他のクラスタ・メンバに、それぞれが起動するときに伝達されます。
- 最初のメンバ以外のクラスタ・メンバに Tru64 UNIX をインストールする場合、TruCluster ソフトウェアをインストールする際に、クラスタ内の既存のメンバからエンハンスド・セキュリティ環境が継承されます。
- クラスタ環境で稼働しているメンバでエンハンスド・セキュリティを有効にする場合は、1 つのメンバからエンハンスド・セキュリティを設定した後、クラスタ内の各マシンをリブートする必要があります。

クラスタでは、基本とエンハンスドの両方のセキュリティがサポートされています。基本セキュリティでは、標準 UNIX の `/etc/passwd` および `/etc/group` ファイルが使われます。基本セキュリティは、省略時の構成です。エンハンスド・セキュリティは、`secconfig` ユーティリティの [Custom Setup] メニュー上の「Security Configuration」アイコンを使って構成されます。

### A.7.3 アクセス制御リスト

`/usr/sbin/sysman secconfig` ユーティリティの [Custom Setup] メニュー上の「Security Configuration」アイコンには、チェック・ボックスがあります。このチェック・ボックスを使うと、基本セキュリティまたはエンハンスド・セキュリティのもとで、すべてのクラスタ・メンバ上のアクセス制御リストのサポートを有効または無効にすることができます。ACL サポートにより、アクセス・チェックおよび ACL 継承の状態 (有効または無効) が決まります。ACL の作成、変更、削除、および調査は、ACL のサポート状況に関係なく行うことができます。

NFS ファイル・システムには、ACL をサポートするために `proplistd` デーモンが必要です。

#### A.7.4 分散ログインと NIS

クラスタでは、共通の認証環境が利用でき、保護された、分散型の、高可用性のログインが可能です。クラスタは、NIS のマスタ、スレーブ、またはクライアントとしても機能させることができます (NIS 環境では、すべてのクラスタ・メンバの役割が同じでなければなりません)。

NIS マスタの場合、クラスタは、`/etc/passwd` の標準ユーザ・プロファイルと、エンハンスド・セキュリティ (保護パスワード・データベースで保守される) で提供されるエンハンスド・ユーザ・プロファイルの、両方の NIS 配布をサポートします。これらのエンハンスド・ユーザ・プロファイルは、`/etc/passwd` を `passwd` マップとして配布するのと同じ方法で、`prpasswd` マップとして NIS で配布することができます。

エンハンスド・セキュリティを稼働しているクラスタを NIS マスタとして設定するには、次の手順を実行します。

1. 指定したアカウントを、`passwd` を含め `/var/yp/src` にロードします。詳細については、『ネットワーク管理ガイド：サービス編』を参照してください。
2. `convuser -Mc` を使って、1 行ごとのエントリで `prpasswd` マップを設定します。詳細については、`convuser(8)` を参照してください。
3. 『ネットワーク管理ガイド：サービス編』で説明しているように、NIS を設定します。`nissetup` の実行中、メンバを NIS サーバの認証リストにバインドするために、`ypbind` のセキュリティ・オプション `-s` を選択します。そして、これらのサーバの 1 つとしてクラスタ別名を指定します。
4. Tru64 UNIX の『ネットワーク管理ガイド：サービス編』で説明しているように、`prpasswd` マップをサポートするようにマップ保守スクリプトを変更します。

---

#### 注意

---

1 つ以上のノードがエンハンスド・セキュリティを実行しているドメインで、Tru64 UNIX Version 5.1A 以上と DIGITAL UNIX バージョン 4.x システムが混在している場合は、Tru64 UNIX Version 5.1A 以上のシステムを NIS マスタにしなければなりません。

ん。エンハンスト・セキュリティを実行しているノードがない場合は、バージョン 4.x システムを NIS マスタにすることができます。

`dxaccounts` ユーティリティでは、NIS アカウントの追加とユーザのセキュリティ・オプションの変更を同時に行うことはできません。最初にアカウントを作成してから、ユーザのセキュリティ・オプションを変更しなければなりません。

ユーザの一次グループが `/var/yp/src/group` マップに定義されていないと、`useradd` コマンドは失敗します。

---

### A.7.5 デーモン

エンハンスト・セキュリティを有効にすると、新しいデーモン `prpasswd` が導入されます。そのデーモンの 2 つのインスタンス (親と子) が、各クラスタ・メンバ上で実行されます。親は主に、子の起動と再起動を担当します。子は、認証データベースへの変更の書き込みを担当します。クラスタでのロックの競合をなくすために、`/var` マウント・ポイントを提供しているクラスタ・メンバ上のデーモンだけが、すべてのクライアントに代わって書き込みを実際に行います。その他の `prpasswd` デーモンは、ホット・スタンバイ・モードになっています。マウント・ポイントを処理しているクラスタ・メンバのアクティブ・デーモンに障害が発生すると、別のメンバが両方の役割を引き受けます。

NIS マスタとしてエンハンスト・セキュリティで動作するクラスタでは、NIS `prpasswd` マップの `prpasswd` と同じように `rpc.yppasswd` デーモンが動作します。

## A.8 デバイスの安全確保

ワークステーションは、コンピュータ・ルームのように保護の容易な環境ではなく、普通のオフィスで使われるため、セキュリティ上の問題があります。

ワークステーションに近づけた誰かが、そのシステムのスーパーユーザになって、そこから別のシステムでのスーパーユーザになれる可能性があります。たとえば、システムをシングルユーザ・モードでブートするのも 1 つの方法です。

オフィスのドアに鍵がある場合、システムから離れるときはドアに鍵をかけてください。

また、テープ・カートリッジやフロッピー・ディスクのような取り外し可能メディアも、使用していないときには鍵のかかる場所に保管して保護しなければなりません。

ワークステーションの中には、コンソール・パスワードを設定できるものもあります。コンソール・パスワードを有効にしているときは、パスワードなしでは省略時のブートしか実行できません。コンソール・パスワードについての詳細は、ハードウェアおよびファームウェアのマニュアルを参照してください。

### A.8.1 デバイス・セキュリティの特性

セキュリティ特性の定義および設定にあたっては、次の点を考慮します。

- デバイス固有の情報の作成、管理。個々のデバイスのシステム省略時設定を置き換えて、必要に応じて追加の権利を許可したり、制限を追加したりできます。端末をロックして使用を防止することもできます。
- システムのセキュア構成に含まれるデバイスに対する省略時の制御パラメータの設定。端末のシステム省略時設定は次のとおりです。
  - ログイン失敗の最大回数は 10。
  - 出荷時には、ログイン・タイムアウトは設定されていません。つまり、暗黙に 0 となり、タイムアウトは無制限です。
  - ログイン失敗時の遅延時間は 2 秒。

セキュア・デバイスを作成または変更する前に、システムのハードウェアおよびソフトウェアのインストール中に必要となる、標準的なデバイス・インストール手順をすべて完了しておく必要があります。デバイス特殊ファイルが `/dev` ディレクトリに存在し、適切なアクセス許可が設定されている必要があります。端末の特殊ファイルは `root` が所有し、グループを `tty` にし、モードを `0620` にする必要があります。

`ls` コマンドを使用すると、インストールが完了していることを検証できます。一般的な例を次に示します。

```
# 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
```

dxdevices プログラムを使用して、システムを構成する全デバイスのセキュリティ特性を定義します。dxdevices プログラムは、デバイス割り当てデータベースおよび端末制御データベースを管理する手段を提供します。dxdevices プログラムを使って、端末や X ディスプレイにセキュリティ属性を割り当てることができます。

#### A.8.1.1 dxdevices プログラムを使ったデバイスの変更、追加および削除

「Devices」ダイアログ・ボックスを使用して、「Modify/Create」ダイアログ・ボックス、「Select devices」ダイアログ・ボックスの順に選択します。デバイスを追加または削除するには、まずデバイスを選択するか入力してから、[File] をクリックして必要な変更を行います。デバイスを変更するには、まずデバイスを選択してから、[Modify] をクリックして必要な変更を行います。詳細については、dxdevices のオンライン・ヘルプを参照してください。

#### A.8.1.2 dxdevices プログラムを使った省略時の値の設定

「Devices」ダイアログ・ボックスを使用して、「Defaults」ダイアログ・ボックスを選択します。必要に応じて、全端末のシステム省略時設定を設定します。「Modify Terminal」ダイアログ・ボックスで設定して、値を個別に置き換えない限り、端末ではこれらの省略時の値が使用されます。詳細については、dxdevices のオンライン・ヘルプを参照してください。

### A.8.2 セキュリティ・データベースの更新

デバイスの省略時設定またはデバイス固有のパラメータを割り当てると、システムは以下のセキュリティ・データベースを更新します。

- システム省略時設定データベース /etc/auth/system/default。すべてのシステム・デバイスに対する省略時の値 (省略時の制御パラメータなど) が含まれます。
- デバイス割り当てデータベース /etc/auth/system/devassign。システム・デバイスに対するデバイス固有の値が含まれます。
- 端末制御データベース /etc/auth/system/ttys.db。認証用のデバイス固有の値 (ログイン失敗の試行回数など) が含まれます。

セキュア構成の各デバイスに対応するエントリが、デバイス割り当てデータベースに必要です。このデータベースには、すべてのシステム・

デバイスのセキュリティ特性に関する情報が集約されています。これにはデバイスのパス名や種類なども含まれます。省略時の設定では、`/etc/auth/system/ttys.db` および `/etc/auth/system/devassign` データベースに、端末 (但し X ディスプレイは対象外) に対するワイルドカードのエントリがあります。

出荷時の X ディスプレイ・エントリには、サイトでシステム省略時設定のログイン・タイムアウトを変更する場合に備え、`:t_login_timeout#0:` エントリがあります。ワイルドカードを使った X ディスプレイ・エントリが必要な場合、次のように作成します。

```
# 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 エンハンスト・セキュリティのトラブルシューティング

この項では、システムで発生する問題と、問題の回避方法や修正方法について説明します。システムを正常に動作させるために必要な重要なファイルおよびプログラムを調査できるように、システムの起動にかかわる事柄について説明します。システムをシングルユーザ・モードにすると、安全にバックアップを行うことができます。システムでの重大なデータ損失を防ぐ方法は、これ以外にはありません。

以下の節で説明する問題が発生すると、システムがブートできなくなります。

### A.9.1 ロック・ファイル

システムのセキュリティ・データベースは、システムの正常動作に欠かせません。これらのデータベースではロック・ファイルを使用して、同期を取りながらセキュリティ関連データベースを書き換えます。プロセスは、データベース・エントリを書き換える前に、ロック・ファイルを自動的に作成します。ロック・ファイルがすでにある場合、プログラムでは、現在別のプロセスがデータベースを使用していると想定し、ロック・ファイルが削除されるまで待機します。ロック・ファイルが存在し続け、適正時間 (現在は 50 秒) 内に変更されない場合、ロック・ファイルを待っているプログラムはシステム・クラッシュまたはソフトウェア・エラーが発生したと見なし、ロック・ファイルを削除して新しいロック・ファイルを作成します。

システムでは、通常のファイル名に `:t` 拡張子を追加して、ロック・ファイルの名前とします。

システムのスタートアップ・スクリプトには、システムの起動時にすべてのロック・ファイルを削除する処理が含まれています。次のファイルには、関連するロック・ファイルがあり、そのファイルがあるためにシステムが正しく動作しないことがあります。

- `/dev/console`
- `/etc/auth/system/default`
- `/etc/auth/system/devassign`

### A.9.2 必須のファイルとファイル内容

システムを稼働させるには、次のファイルが必要です。

- `/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 /tcb/files/auth.db データベース

システムが動作を開始すると、セキュリティ・データベースでさまざまなパラメータが調査されます。いずれかのデータベースが壊れると、システムは正常にブートされません。可能であれば、スタートアップ・プログラムによってデータベースに問題があることが報告され、システムを修復できるように、システム・コンソールでシングルユーザ・シェルが起動されます。ただし、場合によっては、システムが起動しなくなり、『システム管理ガイド』に記載されているスタンドアロン手順でシステムを修復しなければならないことがあります。

root 用のエンハンスト (保護) パスワード・データベースのエントリは、/tcb/files/auth.db データベースに保存されます。root に対応するエントリが矛盾する場合、システムはシングルユーザ・モードになりますが、起動するシェルのセキュリティ・パラメータはすべて省略時の属性になります。

システムがシングルユーザ・モードの場合、次のコマンドを入力して、root に対するエンハンスト (保護) パスワード・データベースのエントリを作成できます。

```
# edauth root
```

次の例に、root に対する一般的なエンハンスト (保護) パスワード・データベースのエントリを示します。

```
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:
```

各フィールドの詳細については、prpasswd(4) を参照してください。システムをブートするには、次のフィールドが必須です。

<i>name</i>	root を含む必要があります。
<i>u_name</i>	root である必要があります。
<i>u_uid</i>	値は 0 とする必要があります。
<i>u_pwd</i>	暗号化されたパスワード。認証時、システムは入力されたパスワードを暗号化されたパスワードと照合



します。データベースのエントリを作成する際には、このフィールドを空にしても構いません。

*chkent*

すべてのデータベースと同様、エントリは *chkent* という単語で終わる必要があります。

このエントリにある他のフィールドは、情報用であるか、または予定外のアカウント・ロックから保護するために使用します。システムでは、シングルユーザ・モードに切り替わる際に、root アカウントをロックする可能性がある条件をすべて無効にします。

#### A.9.2.2 /etc/auth/system/tty.s.db ファイル

端末制御データベースには、システム・コンソールに対する有効なエントリがある必要があります。システム・コンソールのエントリは、*console* という単語で始め、その後にコロンを付ける必要があります。そして、*chkent* という単語で終らなければなりません。唯一必須のフィールドは *t\_devname* で、*console* という値を設定する必要があります。たとえば次のとおりです。

```
console:t_devname=console:chkent:
```

#### A.9.2.3 /etc/auth/system/default ファイル

システム省略時設定データベースは、最初のフィールドを *default* とし、*chkent* で終わる必要があります。このデータベースに関連する *:t* ロック・ファイルがあってはなりません。

一般的な例を次に示します。

```
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 /etc/auth/system/devassign ファイル

コンソールに対するエントリが矛盾している場合、アプリケーションは起動できません。最初のフィールドは `console` という単語にし、最後のフィールドは `chkent` という単語にする必要があります。 `v_type` フィールドには、 `terminal` を設定しなければなりません。

一般的な例を次に示します。

```
console:v_devs=/dev/console:v_type=terminal:\n:chkent:
```

#### A.9.2.5 /etc/passwd ファイル

`/etc/passwd` ファイルはパスワード・データベースです。このファイルが存在し、正しい形式でなければなりません。このファイルの暗号化されたパスワードは更新されません。

#### A.9.2.6 /etc/group ファイル

`/etc/group` ファイルはグループ・データベースです。このファイルが存在し、正しい形式でなければなりません。

#### A.9.2.7 /sbin/rc[023] ファイル

`/sbin/rc[023]` ファイルは、 `init` で実行レベルを切り替えるために使用されます。インストール後に、これらのファイルのコピーを保存しておいてください。

#### A.9.2.8 /dev/console ファイル

`/dev/console` ファイルは、システム・コンソールに対応するキャラクタ型デバイスです。このファイルは、システムのブートに必須です。

#### A.9.2.9 /dev/pts/\* および /dev/tty\* ファイル

`/dev/pts/*` および `/dev/tty*` ファイルは、プロセス間通信に使用する疑似端末デバイスです。

#### A.9.2.10 /sbin/sulogin ファイル

`/sbin/sulogin` 実行可能ファイルにより、シングルユーザ・モードでのアクセスを、 `root` のパスワードを知っているユーザに限定できます。

#### A.9.2.11 /sbin/sh ファイル

/sbin/sh 実行可能ファイルは、システムでシェルを起動し、シングルユーザ・モードへ移行するのに必要です。

#### A.9.2.12 /vmunix ファイル

/vmunix ファイルは、オペレーティング・システムの実行可能イメージです。ブート・ロード・ソフトウェアは、ブート時にオペレーティング・システムをメモリにロードし、制御をそれに移します。

### A.9.3 ログインまたはパスワード変更での問題

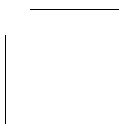
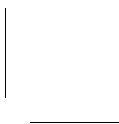
システムへのログインまたはパスワード変更の問題が発生した場合、`fverify` コマンドを使用して、セキュリティ・サブセットのファイルの属性をチェックしてください。たとえば、OSFC2SEC510 サブセットのファイルのファイル属性を確認するには、次のコマンドを入力します。

```
# cd /  
# /usr/sbin/fverify < /usr/.smdb./OSFC2SEC510.inv
```

ローカル・ユーザのプロファイル・ファイルのファイル属性は、`ls -l` および `authck -pf` コマンドを使用してチェックします。

ユーザから報告されたログイン上の問題が、保護プロファイルを更新できない、またはロックを取得できないといったもので、またアカウント管理を集中化している場合は、A.6 節を参照してください。

`dxaccounts` および `usermod` などのユーティリティでは、`/etc/.AM_is_running` というロック・ファイルを共用しています。このファイルが存在する場合、ユーティリティは警告を表示します。



# B

## セキュア・シェル

セキュア・シェル・ソフトウェアは、従来の安全性の低いネットワーク・コマンドに追加して、または置き換えて使用できる安全性の高い一連のネットワーク・コマンドを提供するクライアント/サーバ型ソフトウェアです。表 B-1 に、従来の安全性の低いネットワーク・コマンドと対応するセキュア・シェル・コマンドを示します。

表 B-1: 従来の安全性の低いネットワーク・コマンドとセキュア・シェル・コマンド

作業	従来の安全性の低いネットワーク・コマンド	セキュア・シェル・コマンド
リモート・システム上でのコマンド実行	rsh	ssh2
リモート・システムへのログイン	rlogin または telnet	ssh2
システム間のファイル転送	rcp または ftp	scp2 または sftp2

この付録には、次の情報が含まれています。

- セキュア・シェルのサーバとクライアント
- セキュア・シェルの概要
- セキュア・シェルのサーバとクライアントの構成
- 安全性の低いネットワーク・コマンドでセキュア・シェルを使用するための構成
- セキュア・シェルによるユーザ認証の構成
- セキュア・シェル・サーバの管理
- セキュア・シェル・コマンドの使用

## B.1 セキュア・シェルのサーバとクライアント

セキュア・シェル・サーバとは、セキュア・シェルのサーバ・ソフトウェアがインストールされ、セキュア・シェルの `sshd2` デーモンが起動されているシステムのことです。セキュア・シェル・ソフトウェアには、Tru64 UNIX オペレーティング・システム Version 5.1A 以上が稼動するシステム上で動作するセキュア・シェルのサーバ・ソフトウェアが含まれています。付録 B では、セキュア・シェル・サーバのことを以後「サーバ」と呼びます。

セキュア・シェル・クライアントとは、セキュア・シェルのクライアント・ソフトウェアがインストールされているシステムのことです。セキュア・シェル・ソフトウェアには、Tru64 UNIX オペレーティング・システム Version 5.1A 以上が稼動するシステム上で動作するセキュア・シェルのクライアント・ソフトウェアが含まれています。

セキュア・シェルのクライアント・ソフトウェアは、`scp2`、`sftp2`、および `ssh2` の各セキュア・シェル・コマンドと、セキュア・シェルのクライアント・ソフトウェアを管理するその他のセキュア・シェル・コマンドを提供します。付録 B では、セキュア・シェル・クライアントのことを以後「クライアント」と呼びます。

特に指定しない限り、Tru64 UNIX セキュア・シェルのサーバ・ソフトウェアとクライアント・ソフトウェアは Tru64 UNIX Version 5.1B 以上への更新またはインストールを実行し、`setld` の処理中に次のメッセージが表示されたときにインストールされます。

```
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 セキュア・シェルの概要

サーバが起動すると、`sshd2` デーモンが (特に指定しない限り) ポート 22 でクライアントによるソケット接続の開始を待ち受けます。クライアントが接続すると、`sshd2` デーモンが子プロセスを起動します。子プロセスはクラ

クライアントと公開ホスト鍵交換を開始します。公開ホスト鍵交換は、クライアントとサーバが公開ホスト鍵を交換して識別情報を相互認証する処理です。公開ホスト鍵は、セキュア・シェル・ソフトウェアのインストール時に `/etc/ssh2/hostkey.pub` としてサーバ上に作成されます。

クライアントが初めてサーバに接続すると、(特に指定しない限り) ユーザはサーバの公開ホスト鍵のコピーを受け取るように求められます。ユーザがサーバの公開ホスト鍵を受け取ると、クライアント上の該当するユーザの `hostkeys` ディレクトリに鍵がコピーされます。クライアントはそれ以降のサーバとの接続で、この公開ホスト鍵を使ってサーバを認証します。サーバの公開ホスト鍵をあらかじめクライアント上の該当するユーザの `hostkeys` ディレクトリに `key_port_servername.pub` としてコピーすることもできます。たとえば、サーバ名が `orange` の場合はサーバの鍵を `key_22_orange.pub` としてコピーします。

クライアントとサーバが互いを認証したら、子プロセスはユーザの認証を実行します。ユーザはサーバ上に有効なユーザ・アカウントとホーム・ディレクトリを持っていなければなりません。子プロセスがユーザの認証に失敗すると、接続は拒否されます。セキュア・シェルでは、次のいずれかのユーザ認証方式が使われます。

- パスワード
- 公開鍵
- ホスト・ベース (ホスト・ベース認証はクライアントとサーバの両方で UNIX オペレーティング・システム・ソフトウェアが稼動している場合のみ使用可能)

子プロセスがユーザの識別情報を認証すると、クライアントとの実際の接続が開始されます。接続には、コマンドの実行、暗号化されたデータ転送、および接続の終了が含まれます。接続が確立すると、クライアントとサーバ間のすべての認証と通信にセキュア・シェル接続が使われます。接続が終了すると、`sshd2` デーモンによって起動された子プロセスも終了します。

### B.3 セキュア・シェルのサーバとクライアントの構成

サーバとクライアントの通信方法を構成するには、サーバの `/etc/ssh2/sshd2_config` ファイルとクライアントの `/etc/ssh2/ssh2_config` ファイルのキーワードの値を設定します。各ユー

ザがクライアント上に自分の `$HOME/.ssh2/ssh2_config` ファイルを持つこともできます。`$HOME` は各ユーザのホームディレクトリの名前を表します。

`sshd2_config` ファイルと `ssh2_config` ファイルには、1 行ごとにキーワードと変数のペアが格納されています。各キーワードには省略時の値が設定されており、変更することができます。キーワードは大文字小文字を区別しません。空の行とナンバー記号 (#) で始まる行は、コメントとして無視されます。

### B.3.1 サーバの構成

`/etc/ssh2/sshd2_config` ファイル (または `sshd2 -f` コマンドで指定したファイル) には、`sshd2` デーモンが起動時に読み込む構成のキーワードと値が格納されています。`sshd2` デーモンを起動するコマンド行でキーワードと値を指定することで `sshd2_config` ファイルのキーワードの値の指定変更ができますが、この方法で設定した値は `sshd2` デーモンが再起動したときには元の `/etc/ssh2/sshd2_config` ファイルの値に戻されます。

`sshd2` デーモンを実行したまま `sshd2_config` ファイルを変更した場合は、`sshd2` デーモンをリセットして変更内容を反映する必要があります。この方法で変更した内容は、新しいクライアント接続だけに適用されます。`sshd2` のリセットの詳細については、B.6.1 項を参照してください。

例 B-1 に `/etc/ssh2/sshd2_config` ファイルの例を示します。`sshd2_config` ファイルの各キーワードの説明については、`sshd2_config(4)` を参照してください。

#### 例 B-1: `sshd2_config` ファイルの例

---

```
## sshd2_config
## SSH version Server Configuration File
##
## General

    VerboseMode    no
# QuietMode      yes
    AllowCshrcSourcingWithSubsystems no
    ForcePTYAllocation no
    SyslogFacility AUTH
# SyslogFacility LOCAL7

## Network
```

## B-4 セキュア・シェル



### 例 B-1: sshd2\_config ファイルの例 (続き)

---

```
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

HostKeyFile    hostkey
PublicHostKeyFile  hostkey.pub
RandomSeedFile  random_seed
IdentityFile    identification
AuthorizationFile  authorization
AllowAgentForwarding  yes

## Tunneling

AllowX11Forwarding  yes
AllowTcpForwarding  yes
# AllowTcpForwardingForUsers  sj1, cowboyneal@slashdot.org
```

### 例 B-1: sshd2\_config ファイルの例 (続き)

---

```
# 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
# PermitRootLogin              nopwd
# PermitRootLogin              yes

## SSH1 compatibility

# Ssh1Compatibility
# Sshd1Path

## Chrooted environment

# ChRootUsers                  ftp,guest
# ChRootGroups                 guest
```

### 例 B-1: sshd2\_config ファイルの例 (続き)

---

```
## subsystem definitions

subsystem-sftp                sftp-server
```

---

## B.3.2 クライアントの構成

/etc/ssh2/ssh2\_config ファイルには、クライアント・ソフトウェアが起動時に読み込む構成のキーワードと値が格納されています。各ユーザが自分の \$HOME/.ssh2/ssh2\_config ファイルを持つこともできます。\$HOME は各ユーザのホームディレクトリの名前を表します。最初に /etc/ssh2/ssh2\_config ファイルが読み込まれ、次に各ユーザのファイルが読み込まれます。EnforceSecureRutils キーワードを除き、最後に取得したキーワードの値が使われます。EnforceSecureRutils キーワードの詳細については、B.4 節を参照してください。

例 B-2 に /etc/ssh2/ssh2\_config ファイルの例を示します。ssh2\_config ファイルの各キーワードの説明については、ssh2\_config(4) を参照してください。

### 例 B-2: ssh2\_config ファイルの例

---

```
## 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

## General

    VerboseMode    no
# QuietMode      yes
# DontReadStdin   no
# BatchMode      yes
# Compression     yes
```

## 例 B-2: ssh2\_config ファイルの例 (続き)

---

```
# 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

Ssh1Compatibility yes
Ssh1AgentCompatibility none
# Ssh1AgentCompatibility traditional
# Ssh1AgentCompatibility ssh2
# Ssh1Path /usr/local/bin/ssh1

## Authentication
```

### 例 B-2: ssh2\_config ファイルの例 (続き)

---

```
## 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 安全性の低いネットワーク・コマンドでセキュア・シェルを使用するための構成

rsh, rlogin, rcp の各コマンドや rcmd() 関数を使用するアプリケーションでセキュア・シェル接続が自動的に使われるように構成するには, /etc/ssh2/ssh2\_config ファイルまたは各ユーザの \$HOME/.ssh2/ssh2\_config ファイルの EnforceSecureRutils キーワードを有効にします。特に指定しない限り, EnforceSecureRutils キーワードは無効になっています。/etc/ssh2/ssh2\_config ファイルで EnforceSecureRutils キーワードが有効になっている場合は, 自分の \$HOME/.ssh2/ssh2\_config ファイルで設定値を置き換えているユーザについてもすべてこのキーワードが有効になります。/etc/ssh2/ssh2\_config ファイルで EnforceSecureRutils キーワードが無効になっている場合は, 各ユーザが自分の \$HOME/.ssh2/ssh2\_config ファイルを使ってこのキーワードを有効にできます。

EnforceSecureRutils キーワードを有効にする場合の注意事項を以下に示します。

- sshd デーモンは子プロセス srcmd を実行および生成します。rshd と rlogind の各デーモンは実行されません。
- /etc/ssh2/sshd2\_config ファイルでは、EnforceSecureRutils キーワードとともに AllowTcpForwarding を有効にする必要があります (省略時の設定)。EnforceSecureRutils キーワードを有効にしたときは、AllowTcpForwarding キーワードを無効にしないでください。
- rsh コマンド、rcp コマンド、および rcmd() 関数を使用するアプリケーションでは、ユーザの認証にホスト・ベース認証のみを使用できます。ホスト・ベース認証の詳細については、B.5.3 項を参照してください。
- rlogin コマンドでは、ユーザの認証にホスト・ベース認証とパスワードによるユーザ認証を使用できます。ホスト・ベース認証の詳細については、B.5.3 項を参照してください。パスワード認証の詳細については、B.5.1 項を参照してください。

## B.5 セキュア・シェルによるユーザ認証の構成

クライアントとサーバは、同じ種類のユーザ認証を使用するように構成する必要があります。ユーザ認証については、次のいずれか、またはすべてを使用できます。

- パスワード
- 公開鍵
- ホスト・ベース (ホスト・ベース認証はクライアントとサーバの両方で UNIX オペレーティング・システム・ソフトウェアが稼働している場合のみ使用可能)

クライアントとサーバで使用するユーザ認証の種類を構成するには、サーバの /etc/ssh2/sshd2\_config ファイルとクライアントの /etc/ssh2/ssh2\_config ファイル (または各ユーザの \$HOME/.ssh2/ssh2\_config ファイル) に含まれる AllowedAuthentications キーワードに値を指定します。ユーザ認証方式は、AllowedAuthentications キーワードに指定した順序で使われます。たとえば、hostbased を先に指定すると、サーバは hostbased の認証を試

行してから次に指定された認証を試行します。このようにして、最初に成功した認証が採用されます。特に指定しない限り、サーバは公開鍵認証、パスワード認証の順に試行します。

### B.5.1 パスワード認証の構成

パスワード認証では、Tru64 UNIX のパスワード認証方式によって認証可能な、パスワードで保護されたユーザ・アカウントを各ユーザに用意する必要があります。Tru64 UNIX のパスワード認証方式には、次のようなものがあります。

- BSD
- エンハンスト・セキュリティ
- NIS (Network Information Service)
- LDAP (Lightweight Directory Access Protocol)
- Kerberos

パスワード認証を使用するには、`/etc/ssh2/sshd2_config` ファイルおよび `/etc/ssh2/ssh2_config` ファイルの `AllowedAuthentications` キーワードの値として、次のように `password` を設定します (省略時の設定)。

```
AllowedAuthentications    password
```

特に指定しない限り、ユーザによるパスワード入力は 1 回だけ許可されます。パスワード入力の試行回数は、`/etc/ssh2/sshd2_config` ファイルの `PasswordGuesses` キーワードの設定値で定義します。

`/etc/ssh2/sshd2_config` ファイルのキーワードの値を変更した場合は、次のコマンドを入力して `sshd2` デーモンをリセットする必要があります。

```
# /sbin/init.d/sshd reset
```

パスワード認証を使用すると、`stderr` にパスワード入力を求めるプロンプトが表示されます。このため、`stderr` がリダイレクトされていると、パスワードの入力待ちとなり、ログイン・コマンドが応答しなくなったように見えることがあります。

Tru64 UNIX のセキュア・シェル・クライアント上でユーザに対して表示されるパスワード入力プロンプトは、セキュア・シェル接続が使われているかどうかによって異なります。サーバ上で `sshd` デーモンが実行されている場合、ユーザに対するプロンプトは次のように表示されます。

`username's password:`

サーバ上で `sshd` デーモンが実行されていない場合、ユーザに対するプロンプトは次のように表示されます。

`Password:`

## B.5.2 公開鍵認証の構成

公開鍵認証では、ユーザが公開鍵と秘密鍵のペアを持っている必要があります。これらの鍵は、ユーザの認証およびクライアントとサーバ間でやり取りされるデータの暗号化と復号に使われます。公開鍵は公開され、ユーザの通信相手となるサーバに配布されます。秘密鍵はローカル・クライアントに隠蔽され、一切公開・配布されません。一方の公開鍵または秘密鍵で実行した操作は、もう一方の対応する公開鍵または秘密鍵でしか元に戻せません。

公開鍵認証には、クライアントの構成とサーバの構成が必要です。

### 注意

ユーザの秘密鍵および公開鍵は、サーバの秘密ホスト鍵 (`/etc/ssh2/hostkey`) および公開ホスト鍵 (`/etc/ssh2/hostkey.pub`) とは別のものです。サーバの秘密ホスト鍵と公開ホスト鍵はセキュア・シェル・ソフトウェアのインストール時に作成され、サーバを認証するときに使われます。ユーザの秘密鍵と公開鍵は各ユーザが作成し、そのユーザ自身を認証するときに使われます。

### B.5.2.1 クライアント上の公開鍵認証の構成

Tru64 UNIX のセキュア・シェル・クライアント上で公開鍵認証を構成するときは、以下の手順に従います。

1. `/etc/ssh2/ssh2_config` ファイルの `AllowedAuthentications` キーワードの値として、次のように `publickey` を設定する (省略時の設定)。

```
AllowedAuthentications    publickey,password
```

2. 各ユーザに対して、自分のユーザ・アカウントにログインし、`ssh-keygen` コマンドを入力して公開鍵と秘密鍵のペアを作成す



るように指示します。ユーザに対して次のような内容のプロンプトが表示されます。

```
$ ssh-keygen
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 1
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 2

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) 3
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
username-hostname.pub | 755B | 0.7 kB/s | TOC: 00:00:01 | 100%
Press return to upload to more hosts or Ctrl-D to exit.
^D
```

- 1 鍵のペアに割り当てるパスフレーズを入力します。
- 2 公開ホスト鍵認証を使用するリモート・ホストからクライアント上の自分のユーザ・アカウントにアクセスする場合は、**yes** を入力して `authorization` ファイルにホスト・エントリを追加します。ホスト・エントリによって、ユーザの名前とクライアント上の自分のユーザ・アカウントにアクセスするときに使うリモート・ホストの名前が特定されます。

**yes** を入力すると、認証時に使うユーザ名を入力するためのプロンプトが表示され、次にリモート・ホスト名を入力するためのプロンプトが表示されます。ホスト名は、完全修飾ドメイン名 (FQDN) を使って入力してください。たとえば、リモート・ホストの名前が `orange` で、そのホストの完全修飾ドメイン名が `color.art.com` であれば、`orange.color.art.com` と入力します。

`authorization` ファイルに追加されるホスト・エントリの形式は、次のとおりです。

```
Key username-hostname.pub
```

- 3 **yes** を入力して、クライアント上の自分のアカウントにアクセスするときに使うリモート・ホストのユーザ・アカウントに公開鍵をコピーします。**yes** を入力すると、公開鍵のコピー先となるリモート・ホストの名前とユーザ・アカウントを入力するためのプロンプトが表示されます。通常は、`authorization` ファイルに追加したホスト・エントリのユーザ名とリモート・ホスト名を入力します。ホスト名は、完全修飾ドメイン名 (FQDN) を使って入力してください。
- 4 ユーザの公開鍵がリモート・ホスト上の指定したユーザ・アカウントにコピーされます。ここでは、特に指定しない限りパスワード認証が利用可能な唯一の認証方式なので、ユーザはリモート・

ホスト上の指定したユーザ・アカウントのパスワードを入力する必要があります。

- ⑤ 公開鍵をリモート・ホストにコピーした結果を示すステータス・メッセージが表示されます。

ssh-keygen コマンドを実行すると、次のものが作成されます。

- クライアント上のユーザが使用する `$HOME/.ssh2` という名前のディレクトリ。`$HOME` はユーザのホームディレクトリの名前です。
- 以下の鍵のペア。
  - `$HOME/.ssh2/id_dsa_1024_a` ファイルには、ユーザの秘密鍵が格納されます。このファイルには、鍵の作成対象となったユーザだけがアクセスできるようにします。
  - `$HOME/.ssh2/id_dsa_1024_a.pub` ファイルと `$HOME/.ssh2/username-hostname.pub` ファイルには、ユーザの公開鍵が格納されます。`username-hostname.pub` ファイルは、ユーザが公開鍵認証を使用するサーバに接続する場合、そのサーバにコピーされるファイルです。
- `$HOME/.ssh2/identification` ファイルには、ユーザの秘密鍵ファイルの名前を特定する次のエントリが格納されます。

```
IdKey id_dsa_1024_a
```
- `$HOME/.ssh2/authorization` ファイルには、ユーザがローカル・ホスト上の自分のユーザ・アカウントにアクセスするときに使うリモート・ホストの公開鍵の名前が格納されます。

#### B.5.2.2 サーバ上の公開鍵認証の構成

サーバ上で公開鍵認証を構成するときは、以下の手順に従います。

1. `/etc/sshd2/sshd2_config` ファイルの `AllowedAuthentications` キーワードの値として、次のように `publickey` を設定します (省略時の設定)。

```
AllowedAuthentications publickey,password
```

`AllowedAuthentications` キーワードの値を変更した場合は、次のコマンドを入力して `sshd2` デーモンをリセットする必要があります。

```
# /sbin/init.d/sshd reset
```

2. クライアントとサーバが同じシステムでない場合は、サーバ上に `$HOME/.ssh2` ディレクトリを作成します。`$HOME` は公開鍵認証が構成されるユーザのホーム・ディレクトリの名前です。
3. 必要に応じて、ユーザの `$HOME/.ssh2` ディレクトリに `authorization` ファイルを作成します。`authorization` ファイルにクライアントのホスト・エントリを追加します。ホスト・エントリによって、ユーザがローカル・ホスト上の自分のユーザ・アカウントにアクセスするときに使うクライアントの公開鍵の名前が特定されます。

`authorization` ファイルに追加されるホスト・エントリの形式は、次のとおりです。

```
Key username-clienthostname.pub
```

4. 公開鍵認証の構成時にユーザが自分の公開鍵をクライアントからリモート・サーバにアップロードしなかった場合は、ユーザの公開鍵ファイル (`$HOME/username-hostname.pub`) をクライアントからサーバ上の該当ユーザの `$HOME/.ssh2` ディレクトリにコピーします。

### B.5.2.3 リモート・サーバへのアクセス

例 B-3 に、公開鍵認証を使用するリモート・サーバにログインしたときの画面表示例を示します。

#### 例 B-3: 公開鍵認証によるログイン時の画面表示

---

```
$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 ユーザ・アクセスの制限

ユーザが公開鍵認証を使用するサーバにログインしたときに、特定の UNIX コマンドしか実行できないように制限することができます。ユーザ・アクセスを制限するには、サーバ上の該当ユーザの `authorization` ファイルに含まれる `Key` エントリの下に実行を許可する UNIX コマンドを指定します。たとえば、次のエントリを指定した場合は、公開鍵認証によってユーザが認証され、ログイン・ディレクトリで `ls` コマンドが実行された後、ユーザはローカル・ホストのプロンプトに戻されます。

Key *username-hostname.pub*  
Command `ls`

#### B.5.2.5 パスフレーズの管理

パスフレーズは、各ユーザの Tru64 UNIX ユーザ・アカウントのパスワードではありません。パスフレーズは、ユーザが `ssh-pubkeymgr` コマンドで公開鍵と秘密鍵のペアを作成するときに入力する秘密のテキストです。クライアントとサーバがユーザについての情報をやり取りするときにパスフレーズが使われます。

ユーザが公開鍵認証を使用するサーバ上でセキュア・シェル・コマンドを入力すると、ユーザに対してパスフレーズの入力が必要されます。同じセッション中にユーザに対してパスフレーズの入力を繰り返し要求しないようにサーバを構成するには、セキュア・シェル・エージェントを実行してユーザの公開鍵をセキュア・シェル・エージェントにロードします。エージェントの実行中は、公開鍵の操作がすべてエージェントに転送されます。セキュア・シェル・エージェントは、ユーザがログアウトするか、またはセキュア・シェル・エージェント自体を停止したときに終了します。

セキュア・シェル・エージェントを実行するときは、以下の手順に従います。

1. サーバにログインします。
2. 次のようにして、セキュア・シェル・エージェントを起動します。

```
ssh-agent2 $SHELL
```

環境変数 `$SHELL` によって各ユーザのログイン・シェルが特定されます。

また、各ユーザのログイン・ファイル (`.login` ファイルなど) に `ssh-agent2 $SHELL` コマンドを追加すると、ユーザがサーバにログインした時点で自動的にセキュア・シェル・エージェントが起動します。

セキュア・シェル・エージェントが指定されたシェルを子プロセスとして呼び出し、シェル・プロンプトが表示されます。

3. 次のようにして、セキュア・シェル・エージェントに秘密鍵をロードします。

```
$ ssh-add2
```

`ssh-add2` コマンドを実行すると、ユーザに対してパスフレーズ入力のプロンプトが表示されます。

### B.5.3 ホスト・ベース認証の構成

ホスト・ベース認証は、パスワードやパスフレーズの識別ではなく、システムの識別に基づく認証方式です。ホスト・ベース認証は、クライアントとサーバの両方で Tru64 UNIX オペレーティング・システム・ソフトウェアが稼動している場合のみ使用できます。

ホスト・ベース認証では、次のものがが必要です。

- `/etc/hosts` ファイルにローカル・ホストおよび (該当する場合は) ローカルの TruCluster 別名の完全修飾ドメイン名。完全修飾ドメイン名の詳細については、『ネットワーク管理ガイド：接続編』および `hosts(4)` を参照してください。
- ユーザのホーム・ディレクトリに `.rhosts` または `.shosts` という名前のファイル。これらのファイルには、ユーザがローカル・ホストにアクセスするときに使う各リモート・ホストのホスト名と完全修飾ドメイン名が格納されます。`.shosts` ファイルを読み込むのはセキュア・シェル・サーバだけです。両方のファイルが存在する場合、セキュア・シェル・サーバは最初に `.rhosts` ファイルを読み込み、次に `.shosts` ファイルを読み込みます。どちらかのファイルで特定の接続に対するアクセスが許可されていれば、もう一方のファイルでアクセスが許可されていなくてもセキュア・シェル接続が使われます。
- ローカルホストとローカル・ホストの通信相手となるリモート・ホストの公開ホスト鍵ファイル。これらのファイルは、ローカル・ホスト上の `/etc/ssh2/knownhosts` ディレクトリに置く必要があります。

ホスト間通信の大部分は双方向であり、ホストはクライアントにもサーバにもなり得るので、ホスト・ベース認証を使って通信するすべてのホストに対して次の手順を実行します。

1. `/etc/ssh2/sshd2_config` ファイルに含まれる以下のキーワードに値を設定します。
  - `AllowedAuthentications` キーワードの値に `hostbased` (省略時の設定) が含まれていることを確認します。エントリが他にもある場合は、`hostbased` を最初のエントリとして設定します。たとえば、次のようにします。

```
AllowedAuthentications    hostbased,password
```

- `IgnoreRhosts` キーワードの値として `no` が設定されていることを確認します。たとえば、次のようにします。

```
IgnoreRhosts    no
```

`/etc/ssh2/sshd2_config` ファイルのキーワードの値を変更した場合は、次のコマンドを入力して `sshd2` デーモンをリセットします。

```
# /sbin/init.d/sshd reset
```

2. `/etc/ssh2/sshd2_config` ファイルまたは各ユーザの `$HOME/.ssh2/sshd2_config` ファイルに含まれる以下のキーワードに値を設定します。

- `AllowedAuthentications` キーワードの値に `hostbased` が含まれていることを確認します。エントリが他にもある場合は、`hostbased` を最初のエントリとして設定します。たとえば、次のようにします。

```
AllowedAuthentications    hostbased,password
```

- `DefaultDomain` キーワードの値として、ローカル・ホストの完全修飾ドメイン名を設定します。たとえば、ローカル・ホストの完全修飾ドメイン名が `color.art.com` の場合は、次のように入力します。

```
DefaultDomain    color.art.com
```

3. ホスト・ベース認証の使用時にローカル・ホストの公開ホスト鍵をローカル・ホストの `/etc/ssh2/knownhosts` ディレクトリから他のホストへコピーできるようにするには、次のように入力します。

```
# ln -sf /etc/ssh2/hostkey.pub \
/etc/ssh2/knownhosts/hostname.fqdn.ssh-dss.pub
```

`hostname` にはローカル・ホストの名前、`fqdn` には `/etc/hosts` ファイルで定義されているローカル・ホストの完全修飾ドメイン名を指定します。たとえば、ローカル・ホストの名前が `orange` で、そのホストの完全修飾ドメイン名が `color.art.com` であれば、`orange.color.art.com.ssh-dss.pub` と入力します。

TruCluster Server 環境では、クラスタ別名の公開ホスト鍵を `/etc/ssh2/knownhosts` ディレクトリに用意する必要があります。クラスタ別名の公開ホスト鍵を用意するには、次のコマンドを入力します。

```
# ln -sf /etc/ssh2/hostkey.pub \
/etc/ssh2/knownhosts/cluster_alias.fqdn.ssh-dss.pub
```

`cluster_alias` にはクラスタ別名の名前, `fqdn` には `/etc/hosts` ファイルで定義されているクラスタ別名の完全修飾ドメイン名を指定します。

TruCluster Server 環境の外にあるホストが TruCluster Server 環境の中にあるホストに接続するには, クラスタの外にあるホスト上の `/etc/ssh2/knownhosts` ディレクトリにクラスタ別名の公開ホスト鍵ファイル (`/etc/ssh2/hostkey.pub`) をコピーします。

4. ホスト・ベース認証を使用する各ユーザのホーム・ディレクトリに, `.rhosts` または `.shosts` というファイルを作成し, ローカル・ホストのエントリとローカル・ホストの通信相手となる各リモート・ホストのエントリを追加します。各エントリには, 完全修飾ドメイン名付きのホスト名を指定します。たとえば, ローカル・ホストの名前が `orange` で, そのホストの完全修飾ドメイン名が `color.art.com` であれば, 次のように入力します。

```
orange.color.art.com
```

ユーザは, 自分のホーム・ディレクトリにある `.rhosts` ファイルまたは `.shosts` ファイルの所有者になる必要があります。 `chown` コマンドを使ってユーザを `.rhosts` ファイルまたは `.shosts` ファイルの所有者に設定し, アクセス許可を 600 (所有者だけに読み取りと書き込みを許可) に設定します。

5. ローカル・ホストの通信相手となる各リモート・ホストの公開ホスト鍵ファイル (`/etc/ssh2/knownhosts/hostname.fqdn.ssh-dss.pub`) のコピーをローカル・ホスト上の `/etc/ssh2/knownhosts` ディレクトリに作成します。

公開ホスト鍵をコピーするには, `ssh-hostbased-setup` コマンドを使用します。 `ssh-hostbased-setup` コマンドは, 指定されたリモート・ホストまたは指定されたファイルに列挙されているリモート・ホストの公開ホスト鍵を確認し, 必要であればローカル・ホスト上の `/etc/ssh2/knownhosts` ディレクトリに鍵をコピーします。

`ssh-hostbased-setup` コマンドを使用するときは, 次のように入力します。

```
# ssh-hostbased-setup hostname | filename
```

<code>hostname</code>	ホスト・ベース認証を構成または確認するリモート・ホストの名前を指定します。
-----------------------	---------------------------------------



filename	ホスト・ベース認証を構成または確認するリモート・ホストの名前を含むファイルを指定します。通常は、 <code>.rhosts</code> ファイルまたは <code>.shost</code> ファイルを指定します。
----------	---

詳細については、`ssh-hostbased-setup(1)` を参照してください。

## B.6 セキュア・シェル・サーバの管理

この項では、次の各作業の実行方法について説明します。

- `sshd2` デーモンの起動、停止、再起動、およびリセット
- ユーザ・アクセスのホーム・ディレクトリへの限定
- 公開鍵と秘密鍵の作成
- セキュア・シェル接続を使った TCP/IP ポートと X11 データのフォワーディング

### B.6.1 `sshd2` デーモンの起動、停止、再起動、およびリセット

`sshd2` デーモンは、(特に指定しない限り) 起動時に `/etc/ssh2/sshd2_config` ファイルの構成情報を使用します。`sshd2` デーモンを起動するときは、コマンド行で構成オプションを指定できます。コマンド行の構成オプションは `/etc/ssh2/sshd2_config` ファイルの値よりも優先され、`sshd2` デーモンが再起動するまでの間有効です。構成情報を永続的に変更する場合は、`/etc/ssh2/sshd2_config` ファイルを編集します。`/etc/ssh2/sshd2_config` ファイルの詳細については、B.3.1 項を参照してください。

- `/etc/ssh2/sshd2_config` ファイルの構成情報を使って `sshd2` デーモンを起動するには、次のように入力します。

```
# /sbin/init.d/sshd start
```

- `/etc/ssh2/sshd2_config` ファイルのキーワードの値を変更して `sshd2` デーモンを起動するには、次のように入力します。

```
# /usr/sbin/sshd keyword value
```

- `sshd2` デーモンを停止するには、次のように入力します。

```
# /sbin/init.d/sshd stop
```

- `/etc/ssh2/sshd2_config` ファイルの構成情報を使って `sshd2` デーモンを再起動するには、次のように入力します。

```
# /sbin/init.d/sshd restart
```

- `sshd2` デーモンをリセットするには、次のように入力します。

```
# /sbin/init.d/sshd reset
```

## B.6.2 ユーザ・アクセスのホーム・ディレクトリへの限定

ユーザが `ssh2` コマンドまたは `sftp2` コマンドを入力したときのユーザ・アクセスをホーム・ディレクトリに限定するには、`ssh-chrootmgr` コマンドを使用します。

ユーザ・アクセスをホーム・ディレクトリに限定するには、以下の手順に従います。

1. 次のように、アクセスを制限するユーザまたはグループの名前を指定した `ssh-chrootmgr` コマンドを入力します。

```
# ssh-chrootmgr username username username
```

2. `/etc/ssh2/sshd2_config` ファイルを編集して、手順 1 で指定した制限ユーザ (または制限グループ) を `ChRootUsers` エントリ (または `ChRootGroups` エントリ) の値として追加します。

3. 次のようにして `sshd2` デーモンをリセットします。

```
# /sbin/init.d/sshd reset
```

4. `/etc/passwd` ファイルを編集して、`/bin/ssh-dummy-shell` を手順 1 で指定した制限ユーザおよび制限グループ内のユーザのシェルとして構成します。

詳細については、`ssh-chrootmgr(1)` および `ssh-dummy-shell(1)` を参照してください。

## B.6.3 公開ホスト鍵と秘密ホスト鍵の作成

公開ホスト鍵と秘密ホスト鍵は、セキュア・シェル・ソフトウェアのインストール時に作成されます。新しい公開ホスト鍵と秘密ホスト鍵を作成する必要があるのは、インストール時に作成された鍵を変更する場合だけです。

新しい公開ホスト鍵と秘密ホスト鍵を作成するには、以下の手順に従います。

1. 次のようにして `sshd2` デーモンを停止します。

```
# /sbin/init.d/sshd stop
```

2. 次のようにしてホスト鍵を作成します。

```
# ssh-keygen2 -P /etc/ssh2/hostkey
```

3. 次のようにして `sshd2` デーモンを起動します。

```
# /sbin/init.d/sshd start
```

---

#### 注意

---

サーバの古い公開ホスト鍵のコピーを持っているユーザが新しい公開ホスト鍵を使ってサーバに接続すると、警告メッセージが表示されます。ユーザは、クライアント上の自分の `hostkeys` ディレクトリからサーバの古い公開ホスト鍵を削除できます。次回クライアントがサーバに接続すると、ユーザはサーバの新しい公開ホスト鍵のコピーを受け入れるように求められます。

---

## B.6.4 セキュア・シェル接続を使った TCP/IP ポートと X11 データのフォワーディング

セキュア・シェル接続を使ってデータを転送するように TCP/IP ポートと X11 接続を構成できます。

### B.6.4.1 TCP/IP ポート・フォワーディング

フォワーディング (トンネリング) とは、そのままでは安全性の低い TCP データをセキュア・シェル接続を使って転送する方法です。たとえば、POP3、SMTP、HTTP などの接続でセキュア・シェル接続を使用する構成が可能です。

TCP/IP ポート・フォワーディングを構成するには、以下の手順に従います。

1. `/etc/ssh2/sshd2_config` ファイルの `AllowTcpForwarding` キーワードの値として `yes` を設定する (省略時の設定)。

`AllowTcpForwarding` キーワードの値を変更した場合は、次のコマンドを入力して `sshd2` デーモンをリセットする必要があります。

```
# /sbin/init.d/sshd reset
```

2. TCP/IP ポートのフォワーディングを構成します。TCP/IP ポート・フォワーディングには、次の 2 種類があります。

- ローカル TCP/IP ポート・フォワーディング (送信トンネルとも呼ばれる)

ローカル TCP/IP ポート・フォワーディングは、ローカルの TCP/IP ポートに出力されるデータを、指定したリモートの TCP/IP ポートに転送する機能です。ローカル・システム上の指定したポート (*local\_port*) に出力されるすべてのデータをリモート・システム上の指定したポート (*remote\_port*) に転送するには、次のように入力します。

```
# ssh2 -L local_port:remote:remote_port user@remote
```

- リモート TCP/IP ポート・フォワーディング (受信トンネルとも呼ばれる)

リモート TCP/IP ポート・フォワーディングは、リモートの TCP/IP ポートに出力されるデータを、指定したローカルの TCP/IP ポートに転送する機能です。リモート・システム上の指定したポート (*remote\_port*) に出力されるすべてのデータをローカル・システム上の指定したポート (*local\_port*) に転送するには、次のように入力します。

```
# ssh2 -R remote_port:local:local_port user@remote
```

#### B.6.4.2 X11 フォワーディング

X11 フォワーディングを有効にするには、以下の手順に従います。

1. `/etc/ssh2/sshd2_config` ファイルの `ForwardX11` キーワードの値として `yes` を設定する (省略時の設定)。

`ForwardX11` キーワードの値を変更した場合は、次のコマンドを入力して `ssh2` デーモンをリセットする必要があります。

```
# /sbin/init.d/sshd reset
```

2. 接続をテストするため、クライアントに接続して X11 コマンドを入力します。たとえば、次のように入力して X クロックのプログラムを起動します。

```
xclock &
```

X クロックのウィンドウが表示されれば、X11 フォワーディングが正常に動作しています。

---

注意

---

クライアントの `DISPLAY` 変数を設定しないでください。この変数を設定すると、暗号化が無効になります。セキュア・シェルによって転送される X 接続には、特別なローカル・ディスプレイ設定が使われます。

---

## B.7 セキュア・シェル・コマンドの使用

この項では、セキュア・シェル・コマンドを使った次の作業の実行方法について説明します。

- クライアントとサーバ間のファイル・コピー
- サーバへのログインとサーバ上でのコマンド実行

---

注意

---

クライアントが初めてサーバに接続すると、ユーザはサーバの公開ホスト鍵のコピーを受け入れるように求められます。ユーザが鍵を受け入れると、サーバの公開ホスト鍵のコピーがクライアント上の該当するユーザの `hostkeys` ディレクトリにコピーされます。クライアントはそれ以降のサーバとの接続で、この公開ホスト鍵を使ってサーバを認証します。

---

### B.7.1 クライアントとサーバ間のファイル・コピー

次のセキュア・シェル・コマンドを使ってファイルをコピーできます。

- `scp2` コマンド
- `sftp2` コマンド

#### B.7.1.1 `scp2` コマンドの使用

サーバとの間でファイルをコピーするには、クライアント上で `scp2` コマンドまたは `scp` コマンドを使用します。インストール処理中に、`scp` コマンド

実行可能ファイルから `scp2` コマンド実行可能ファイルへのシンボリック・リンクが作成されます。`scp2` コマンドは、通常のユーザ権限で実行されます。

- ローカル・システムからリモート・システムにファイルをコピーするには、次のように入力します。

```
scp2 /directory/file user@system:/directory/file
```

- リモート・システムからリモート・システムにファイルをコピーするには、次のように入力します。

```
scp2 user@system:/directory/file user@system:/directory/file
```

相対パスも使用できます。その場合は、ユーザのホーム・ディレクトリからの相対パスとして解釈されます。

`scp` コマンドの詳細については、`scp2(1)` を参照してください。

#### B.7.1.2 `sftp2` コマンドの使用

サーバとの間でファイルをコピーするには、クライアント上で `sftp2` コマンドまたは `sftp` コマンドを使用します。インストール処理中に、`sftp` コマンド実行可能ファイルから `sftp2` コマンド実行可能ファイルへのシンボリック・リンクが作成されます。

`sftp2` コマンドは `ftp` コマンドと同様の機能を持っていますが、`ftp` デモンや `ftp` クライアントを接続に使用しません。`sftp2` コマンドは、通常のユーザ権限で実行されます。

`sftp2` コマンドを使ってリモート・ホストに接続し、ファイルをコピーするには、次のように入力します。

```
sftp2 [options] hostname
```

`sftp2` コマンドに `scp2` の構文を使用することもできます。詳細については、B.7.1.1 項、`scp2(1)`、`sftp2(1)` を参照してください。

#### B.7.2 サーバへのログインとサーバ上でのコマンド実行

サーバにログインしてサーバ上でコマンドを安全に実行するには、クライアント上で `ssh2` コマンドまたは `ssh` コマンドを使用します。インストール処理中に、`ssh` コマンド実行可能ファイルから `ssh2` コマンド実行可能ファイルへのシンボリック・リンクが作成されます。

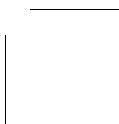
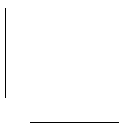
サーバ上でコマンドを実行するには、次のように入力します。

```
ssh2 [options] server_name [command]
```

ユーザがサーバへのログインに成功すると、sshd2 デーモンが次の処理を実行します。

1. そのユーザの権限で動作します。
2. 基本的な環境をセットアップします。
3. ユーザのホーム・ディレクトリに移動します。
4. ユーザのシェルを実行します。

ssh2 コマンドの詳細については、ssh2(1) を参照してください。





# C

## Single Sign On

Single Sign On (SSO) は、ftp、rcp、rlogin、rsh、および telnet ネットワーク・コマンドや Kerberos を使うアプリケーションを使用している場合に、Kerberos テクノロジを使って安全な接続を提供するオプションのクライアント/サーバ・ソフトウェアです。

この付録には、次の情報があります。

- Kerberos サーバおよびクライアント
- Kerberos 認証プロセス
- SSO ソフトウェアのアップデート
- SSO ソフトウェアのインストールおよび構成
- Tru64 UNIX の SSO 構成ファイル
- アカウントおよびグループの作成
- SSO ソフトウェアの管理
- SSO ソフトウェアのトラブルシューティング

### C.1 Kerberos サーバおよびクライアント

Kerberos サーバは、Kerberos サーバ・ソフトウェアがインストールされているシステムのことです。Kerberos サーバ・テクノロジーは、Tru64 UNIX オペレーティング・システム・ソフトウェアには備わっていません。Windows 2000 Server のような Kerberos サーバがイントラネットに存在しなければなりません。この付録では、Kerberos サーバをサーバと呼びます。

Kerberos クライアントは、Kerberos クライアント・ソフトウェアがインストールされているシステムのことです。Tru64 UNIX SSO ソフトウェアは、Tru64 UNIX Version 5.1A 以降のオペレーティング・システム・ソフトウェアを稼動しているシステムで実行する Kerberos クライアント・ソフトウェアを提供します。Kerberos クライアント・ソフトウェアは、ftp、rcp、

rlogin, rsh, telnet コマンドおよび rcmd 関数の Kerberos 版を提供します。この付録では、Kerberos クライアントをクライアントと呼びます。

## C.2 Kerberos 認証プロセス

Kerberos ネットワークは、レルムと呼ばれるセキュリティ・ドメインに分割されます。レルムにはそれぞれ一次サーバや二次サーバがあり、独自のセキュリティ・ポリシーを実現します。

管理者は、サーバにプリンシパル・データベースを作成し、管理します。このプリンシパル・データベースには、各プリンシパル(ユーザ、サービス、アプリケーション、およびホスト)に対するキー(エントリ)が含まれます。キーには、パスワードのような秘密の値など、プリンシパルに関する情報が含まれます。サーバはこのキー情報を使い、プリンシパルを認証します。プリンシパル・データベースの詳細については、Kerberos サーバのマニュアルを参照してください。

レルムは階層構造です。各レルムは複数の子レルムを持つことができます。また、各レルムは1つの親レルムを持つことができます。これにより、Kerberos を導入する組織では、組織内部のさまざまな情報クラスに対応するセキュリティ・レベルを設定したり、互いに直接の関係を持たないレルムの間で認証情報を共用できます。レルム内のプリンシパルの名前はすべて一意である必要があります。情報のセキュリティ・ポリシーは、レルム内のすべてのプリンシパルで共通です。

Tru64 UNIX SSO ソフトウェアをインストールすると、プリンシパルの認証要求を Kerberos サーバに送信する SIA (Security Integration Architecture) モジュール がインストールされます。

Tru64 UNIX クライアント上では、サービス鍵テーブルと呼ばれるファイルを作成し、管理できます。サービス鍵テーブルには、あらかじめクライアントとの接続を認証されたそれぞれのプリンシパル(ホスト)用のプリンシパル・データベースから抽出されたキーが含まれています。以降の要求に対して、クライアントはこのキー情報を再利用し、プリンシパルを再認証します。つまり、最初にサーバがプリンシパルを認証すると、そのプリンシパルのキーがサービス鍵テーブルにあれば、クライアントはプリンシパルを再認証できるのです。

## C.3 SSO ソフトウェアのアップデート

SSO Version 1.0 ソフトウェアをアップグレードするには、インストールされている SSO のサブセットをアンインストールし、新しく SSO ソフトウェアをインストールする必要があります。たとえば、SSO Version 1.0 ソフトウェア (W2KSSO100) をアンインストールするには、次のように入力します。

```
# /usr/sbin/setld -d W2KSSO100
```

## C.4 SSO ソフトウェアのインストールおよび構成

いくつかの SSO ソフトウェアを Tru64 UNIX システムと Windows 2000 Server システムにインストールする必要があります。

SSO ソフトウェアを Tru64 UNIX システムにインストールする前に、あらかじめ Windows 2000 Server システムにインストールしておく必要があります。

Tru64 UNIX オペレーティング・システムにインストールする SSO ソフトウェアは、Windows 2000 Active Directory に認証要求を送信する SIA (Security Integration Architecture) メカニズムをインストールします。

Windows 2000 Server にインストールする SSO ソフトウェアは、Active Directory を拡張し、Tru64 UNIX ユーザ・アカウントおよび、ユーザのログイン名、ユーザ ID (UID)、グループ ID (GID)、コメント、ホーム・ディレクトリへのパス、およびログイン・シェルなどのグループ属性を含めます。Tru64 UNIX ユーザ・アカウントおよびグループ属性の詳細については、『システム管理ガイド』を参照してください。

SSO ソフトウェアをインストールする前に、次の点に留意してください。

- Active Directory の名前は、大文字と小文字を使い分ける必要があります、小文字で入力されている必要があります。Active Directory 名を大文字に変える必要がある場合は、Windows のマニュアルを参照してください。
- SSO と分散型コンピューティング環境 (DCE) は、システム設定要件が異なるバージョンの Kerberos を実装していて、互換性がないため、DCE ソフトウェアと SSO は同じシステムにインストールできません。
- ASU (Advanced Server for UNIX) Version 5.1A 以上のソフトウェアと SSO Version 2.0 以上のソフトウェアを Tru64 UNIX システムで稼働させると、マシン・アカウント名の競合が発生します。この競合を解決す

るには、ホスト名を ASU サーバ名として使用しないよう ASU サーバを構成する必要があります (省略時の設定)。SSO ソフトウェアによって、Tru64 UNIX システムのホスト名と一致するマシン・アカウントが Active Directory に作成されます。省略時の設定どおりに、ASU サーバ名としてホスト名が ASU サーバに使われると、SSO ソフトウェアによって作成されたアカウントが指定変更され、SSO 機能が失敗します。ASU サーバ名としてホスト名が使われないように ASU サーバを設定する方法については、ASU 『*Advanced Server for UNIX* リリース・ノート』を参照してください。

#### **C.4.1 Windows 2000 システムでの SSO ソフトウェアのインストールおよび構成**

Windows 2000 のドメイン・コントローラに SSO ソフトウェアをインストールすると、SSO ソフトウェアは次の動作をします。

- Active Directory スキーマを拡張し、Tru64 UNIX ユーザ・アカウントおよびグループ属性を含めます。Active Directory は複製されるため、ドメインのすべての Windows 2000 Server は拡張された Active Directory のコピーを受け取ります。Active Directory については、Windows のマニュアルを参照してください。
- MMC (Microsoft Management Console) をアップデートして、ユーザおよびグループの Tru64 UNIX プロパティ・ページを含めます。

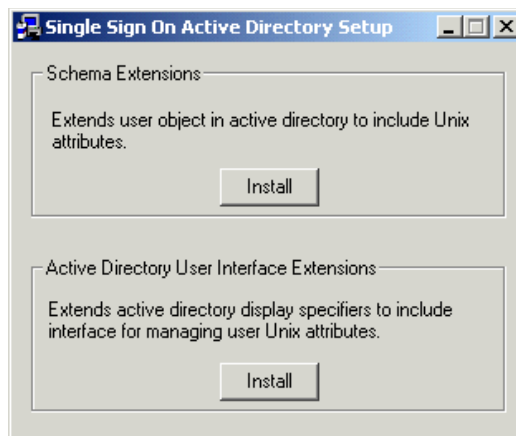
Tru64 UNIX 属性を含むユーザ・アカウントおよびグループを作成する Windows 2000 Server のそれぞれに SSO ソフトウェアをインストールする必要があります。SSO ソフトウェアはこのシステム上で MMC をアップデートして、ユーザおよびグループの Tru64 UNIX プロパティ・ページを含めます。

##### **C.4.1.1 Active Directory スキーマの拡張**

Windows 2000 のドメイン・コントローラで Active Directory スキーマを拡張するには、次の手順を実行します。

1. Windows 2000 のドメイン・コントローラの CD-ROM ドライブに Tru64 UNIX の基本 CD-ROM を挿入します。

2. Windows のエクスプローラ・ウィンドウで、CD-ROM ドライブのアイコンを開き、続いて Windows2000\_SSO フォルダ、kit/windows\_kit フォルダを開きます。
3. setup.exe ファイルをダブルクリックします。次のようなウィンドウが表示されます。



ZK-1674U-AI

4. 「Schema Extensions」ボックスで [Install] ボタンをクリックし、Active Directory を拡張して、Tru64 UNIX 属性が含まれるようにします。「Active Directory User Interface Extensions」ボックスで [Install] ボタンをクリックし、MMC をアップデートして Tru64 UNIX プロパティ・ページが含まれるようにします。

インストールに関する情報メッセージが、コマンド・プロンプト・ウィンドウに表示されます。このウィンドウは、インストールが正常終了すると自動的に閉じられます。

#### C.4.1.2 MMC のアップデート

Windows 2000 Server 上で MMC をアップデートし、Tru64 UNIX プロパティ・ページを含めるには、次の手順を実行します。

1. Windows 2000 Server の CD-ROM ドライブに Tru64 UNIX の基本 CD-ROM を挿入します。

2. Windows のエクスプローラ・ウィンドウで、CD-ROM ドライブのアイコンを開き、続いて Windows2000\_SSO フォルダ、kit/windows\_kit フォルダ、MMC\_Setup フォルダを開きます。
3. setup.exe ファイルをダブルクリックし、画面の説明に従います。

#### C.4.2 Tru64 UNIX システムでの SSO ソフトウェアのインストールおよび構成

SSO ソフトウェアは、基本の Tru64 UNIX オペレーティング・システム・ソフトウェアに含まれるオプションのサブセットです。

SSO サブセットは、OSFSSOW2Knnn、Single Sign On for Windows 2000 (Network-Server/Communications) という名前です。nnn は、Tru64 UNIX のバージョン番号です。現在のバージョン番号は、『リリース・ノート』を参照してください。

サブセットのインストールの詳細については、『インストレーション・ガイド』および setld(8) を参照してください。

Tru64 UNIX システムに SSO ソフトウェアをインストールした後、SSO ソフトウェアと Kerberos SIA メカニズムを構成します。任意で、TruCluster Server 環境で SSO ソフトウェアを構成することもできます。

##### C.4.2.1 SSO ソフトウェアの構成

SSO ソフトウェアを構成するには、/usr/sbin/w2ksetup スクリプトを実行する必要があります。/usr/sbin/w2ksetup スクリプトによって、次の必須情報の入力が必要です。

- Windows 2000 のドメイン・コントローラの管理特権を持つユーザ・アカウントの名前とパスワード。この管理者アカウントは、Administrators、DnsAdmins、Domain Admins、Domain Users、および Users グループのメンバである必要があります。
- Windows 2000 の完全ドメイン名。通常、ドメイン・ネーム・サービス (DNS) のネーム・サーバと同じドメイン名です。このドメイン名は、大文字と小文字を使い分ける必要があり、大文字で入力されている必要があります。

- w2khost.sso.corp.com など, Windows 2000 のドメイン・コントロールのホスト名。このドメイン名は, 大文字と小文字を使い分ける必要があり, 小文字で入力されている必要があります。

SSO ソフトウェアを構成するには, 次の手順を実行します。

1. Windows 2000 Server 用のエントリを /etc/hosts ファイルに追加します。詳細については, hosts(4) を参照してください。
2. 次のように入力して, SSO セットアップ・スクリプトを起動します。

```
# /usr/sbin/w2ksetup
```

SSO セットアップ・スクリプトを実行するように求められます。

3. SSO セットアップ・スクリプトを実行するかどうかを質問されたら「yes」を入力します。

```
Do you want to run the setup script now? [y/n]? y
```

次の情報が表示されます。

```
Running /usr/sbin/w2ksetup...  
/usr/sbin/w2ksetup[79]: 2598 Terminated
```

#### 注意

SSO セットアップ・スクリプトを実行するかどうかを質問されたときに「no」と入力すると, コマンド・プロンプトに戻ります。次のコマンドを入力して, SSO セットアップ・スクリプトの実行を再開できます。

```
# /usr/sbin/w2ksetup
```

Tru64 UNIX システムが認証に使用する Windows 2000 のドメイン名を入力するように求められます。

4. 次のように, [Enter] キーを押して, 省略時の名前を使用するか, ドメイン名を入力してから [Enter] キーを押す。

```
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]
```

Windows 2000 のホスト名コントローラの名前を入力するように求められます。

5. 次のように、[Enter] キーを押して、省略時の名前を使用するか、ホスト名を入力してから [Enter] キーを押す。

```
Enter the hostname of a Windows 2000 domain controller.  
Domain Controller: w2khost.sso.corp.com
```

---

#### 注意

---

Windows 2000 のドメイン・コントローラに接続できない場合は、`/etc/hosts` ファイルをチェックし、Windows 2000 のドメイン・コントローラの完全修飾ドメイン名のあるエントリがあるか確認してください。もし見つからなければ、ファイルにエントリを追加してください。また、クライアントとサーバ・システムの時刻が同じかどうかもチェックしてください。これらのシステム間に大きな時間差があると、接続に問題が発生する場合があります。

---

Windows 2000 のドメイン・コントローラに Tru64 UNIX システムのマシン・アカウントを作成するには、Windows 2000 KDC に管理特権のあるユーザ・アカウントの名前とパスワードを入力する必要があります。

次の情報が表示されます。

```
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. 次のように、Windows 2000 KDC の管理特権のあるアカウントのユーザ名とパスワードを入力します。

```
Enter Admin principal: user  
Password for user@SSO.CORP.COM:
```

次の情報が表示されます。

```
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.
```

Windows 2000 サービスを起動するように求められます。

7. すぐにサービスを起動する場合は、次の質問に対して「yes」と入力します。

```
Do you want to start the Windows 2000 services now? [y/n]?: y
```

次の情報が表示されます。

```
LDAP caching daemon (ldapcd) started
```

8. X サーバを再起動します。

```
# /sbin/init.d/xlogin restart
```

#### C.4.2.2 TruCluster Server 環境での SSO ソフトウェアの構成

TruCluster Server 環境で SSO ソフトウェアを構成するには、次の手順を実行します。

1. それぞれの TruCluster Server メンバで `/usr/sbin/w2ksetup` スクリプトを実行する (C.4.2.1 項)。
2. それぞれのクラスタ・エイリアスについて、`/usr/sbin/w2ksetup` スクリプトを最後に実行した TruCluster Server メンバで次のコマンドを実行します。

```
# creacct -h fully_qualified_cluster_alias_name -u
```

このコマンドによってクラスタ・エイリアスの完全修飾名が Active Directory に追加され、クラスタ・エイリアスの使用時に、SSO ログインが可能になります。

#### C.4.2.3 Kerberos と他の SIA メカニズムの併用 (必要に応じて)

Kerberos SIA メカニズムは、SIA メカニズムの構成ファイル (`/etc/sia/matrix.conf`) の最初のエントリである必要があります。SSO のインストール手順は、この制約に従います。ただし、他の SIA セキュリティ・メカニズムを追加する場合には、`/usr/sbin/siacfg` コマンドを

使って、Kerberos エントリを削除し、他のエントリを追加した後、Kerberos エントリを再度追加します。たとえば次のようにします。

1. 次のように入力して、Kerberos エントリを削除します。
- # siacfg -r Kerberos
2. 次のように入力して、新しいエントリを追加します。
- # siacfg -a sianew /location
3. 次のように入力して、Kerberos エントリを再度追加します。
- # siacfg -a -P -g sci Kerberos /usr/shlib/libcsfsiad.so

C.5 Tru64 UNIX の SSO の構成ファイル

表 C-1 に、Tru64 UNIX の SSO の構成ファイルを示します。

表 C-1: SSO の構成ファイル

ファイル	内容
krb.conf	ホスト・コンピュータの省略時のレルム。既知のレルムを、ホスト名やネットワークの場所に基づいて、一次および二次 Kerberos サーバに関連付けます。
krb.realms	サーバ名および対応するレルム名。
v5srvtab	Kerberos サーバのプリンシパル・データベースから抽出されるプリンシパル・キー。
.k5login	特定のユーザ・アカウントへのアクセスを承認されたプリンシパルのリスト。
ldapcd.conf	パフォーマンスを上げるためにチューニングできる SSO のキャッシュ・パラメータ。
ldapusers.deny	SSO がインストールされている時に、Tru64 UNIX のみで認証される各ローカル Tru64 UNIX ユーザのエントリ。

C.5.1 krb.conf ファイル

/krb5/krb.conf ファイルは、Kerberos のレルム情報が含まれたテキスト・ファイルです。この情報は、システムの省略時のレルムを定義し、既知のレルムを、ホスト名やネットワークの場所に基づいて、一次および二次 Kerberos サーバに関連付けます。

省略時の命名規則 (つまり, 順序規則または DNS の循環規則) を使って, Kerberos サーバの名前を構成できる場合は, `krb.conf` ファイルを構成したり, 管理したりする必要はありません。

`krb.conf` ファイルが見つからない場合, 空である場合, あるいは有効な省略時のレルムが含まれていない場合, Tru64 UNIX オペレーティング・システムによって, ホストのドメイン名が大文字に変換され, 省略時のレルム名として使用されます。Kerberos サーバの情報が, `krb.conf` ファイルで見つからないと, 省略時の命名規則が設定されていれば, Tru64 UNIX オペレーティング・システムはKerberos サーバを見つけようとします。

`krb.conf` ファイルでは, 各エントリは別々の行にある必要があり, フィールドは空白やタブで区切られ, コメントの行頭にはナンバー記号 (#) が付き (ナンバー記号の後ろから行末までの文字列は無視されます), 空白行や行頭, 行末の空白文字は無視されます。

Kerberos サーバを使用する順序を示すので, `krb.conf` ファイルでのエントリの順序は重要です。Kerberos サーバに接続しようとするアプリケーションは, エントリを順番に 1 つずつ読み込みます。他の Kerberos サーバが使用できない, またはネットワーク・タイムアウトが発生した場合 (たとえば, 認証処理中にクライアントと Kerberos サーバ間のネットワーク接続が中断された場合など), 二次 Kerberos サーバが使用されます。

例 C-1 に, `krb.conf` ファイルの例を示します。

#### 例 C-1: `krb.conf` ファイルの例

---

```
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]
```

---

例 C-1 のエントリは, 次の構成を作成します。

- ① 最初の行は, システムの省略時のレルム名を示します。ドメイン名と視覚的に区別するために, レルム名は大文字になっています。サイトで大文字表記での規則に従わない場合には, レルム名の大文字小文字

を正しく入力する必要があります。例では、システムの省略時のレルム名は `BIZ.COM` です。

- ② 2行目は、レルムの一次 Kerberos サーバのレルム名と完全修飾ドメイン名 (FQDN) を示します。例 C-1 では、一次 Kerberos サーバの FQDN は `shoe.biz.com` です。
- ③ 次の行は、レルムの二次 Kerberos サーバのレルム名と FQDN を示します。例 C-1 では、二次 Kerberos サーバの名前は `sneakers.biz.com` および `boot.biz.com` です。
- ④ 次の行は、レルム間認証が実行されるレルムの名前を示します。レルム間認証は、再認証を行わずに、レルムが他のレルムからの認証を受け入れることです。また、この行は、Kerberos サーバの FQDN に続いて `admin server` およびオプションの `tcp/port #` を指定します。

省略時の設定では、UDP が省略時の接続プロトコルになっているため、特に指定する必要はありません。Kerberos サーバが TCP を接続プロトコルとして使用する場合は、`tcp`、および TCP が使用する `port #` を指定する必要があります。ポートの値を指定するには、数値、または `tcp/88` や `tcp/kerberos5` など、`/etc/services` ファイルに記載されているサービス名を使います。

例 C-1 では、レルム名は `FOOTWEAR.BIZ.COM`、Kerberos サーバは `leather.footwear.biz.com` です。この Kerberos サーバは、省略時の UDP 接続プロトコルを使用します。

詳細については、`krb.conf(4)` を参照してください。

## C.5.2 `krb.realms` ファイル

`/krb5/krb.realms` ファイルは、Kerberos サーバ名と、それらに関連付けられている Kerberos レルム名が含まれたテキスト・ファイルです。セキュア・アプリケーションは、`krb.realms` ファイルを使って、チケットの要求先となるレルムを判別します。省略時の設定では、Tru64 UNIX オペレーティング・システムはサーバ名を大文字に変換し、レルム名として使用します。この場合、`krb.realms` ファイルを構成したり、管理したりする必要はありません。

ワイルドカードは、`krb.realms` ファイルの特殊文字で、1つのエントリで複数のサーバを1つのレルムに関連付けます。

使用できるワイルドカード文字は、次の 2 つです。

- 指定のドメイン・ルート名を含むサーバをすべて指定するには、ドメイン名の前にアスタリスク (\*) を指定します。たとえば、\*.biz.com は、footwear.exec.biz.com など、biz.com で終了するすべてのドメインのすべてのサーバを指定します。
- 任意の文字を指定するには、最初のフィールドのサーバ名またはドメイン名の前にクエスチョン・マーク (?) を使います。たとえば、???footwear.biz.com は、bigfootwear.biz.com など、biz.com ドメイン内で、「footwear」の前に 3 文字が入る名前の付いたあらゆるサーバを指定します。

セキュア・アプリケーションが krb.realms ファイルを検索するとき、一致するサーバ名があるかを検索した後、一致するドメイン名があるかを検索します。一致するものが見つからない場合、ワイルドカードを使って一致を検索します。該当する関連エントリがない、または krb.realms ファイルが存在しない場合、セキュア・アプリケーションはサーバのドメイン名を大文字に変換し、省略時の名前として使用します。

krb.realms ファイルでのエントリの順序は重要ではありません。それぞれのエントリは、別々の行にある必要があり、空白文字またはタブで区切られた次のような形式の 2 つのフィールドが必要です。

- 最初のフィールドには、サーバ名を指定します。ドメイン名を指定して、ドメインのすべてのサーバを 1 つのレルム名にマッピングできます。ドメイン名の前にピリオドを付ける必要があります。
- 次のフィールドには、対応するレルム名を指定します。命名規則により、ドメイン名と視覚的に区別するために、レルム名は大文字で指定します。サイトで大文字表記を使用しない場合には、レルム名の大文字小文字を正しく入力する必要があります。

krb.realms ファイルにコメントを作成するには、ナンバー記号 (#) を使います。ナンバー記号の後から行末までの文字列はすべて無視されます。また、空白行および行頭、行末の空白文字も無視されます。

例 C-2 は、krb.realms ファイルの例です。

## 例 C-2: krb.realms ファイルの例

---

```
footwear.biz.com SERIOUS.BIZ.COM [1]  
.admin.biz.com ADMIN.BIZ.COM [2]  
*.biz.com BIZ.COM [3]
```

---

例 C-2 のエントリは、次の関連付けを行います。

[1] footwear.biz.com サーバを SERIOUS.BIZ.COM レルムの関連付け。

[2] admin.biz.com ドメインにあるすべてのサーバと ADMIN.BIZ.COM レルムの対応付け。

ピリオドを前に追加すると、最初のフィールドがサーバ名ではなくドメイン名として認識されます。通常、レルム名はドメイン名を大文字にしたものなので、この行は特に必要ではありません。ただし、この例では、3 行目によって admin.biz.com ドメインのサーバを BIZ.COM レルムにマッピングされないようにするために、この行が必要です。

[3] ルート名 biz.com を名前に含むその他のドメインにあるすべてのサーバを BIZ.COM レルムに関連付け。たとえば、sales.biz.com のサーバと support.teams.biz.com ドメインが BIZ.COM レルムにマップされます。

詳細については、krb.realms(4) を参照してください。

### C.5.3 v5srvtab ファイル

/krb5/v5srvtab サービス鍵テーブル・ファイルは、Kerberos サーバのプリンシパル・データベースから抽出されたプリンシパル・キーを含むバイナリ・ファイルです。Tru64 UNIX などの Kerberos クライアントは、このサービス鍵テーブルのキー情報を使って、プリンシパルを再認証します。つまり、最初に Kerberos サーバがプリンシパルを認証すると、キーがサービス鍵テーブルにあれば、Kerberos クライアントはプリンシパルを再認証できるのです。

詳細については、v5srvtab(4) を参照してください。/krb5/v5srvtab サービス鍵テーブル・ファイルの管理については、C.7.4 項 を参照してください。

## C.5.4 .k5login ファイル

/krb5/.k5login ファイルは、特定のユーザ・アカウントへのアクセスが承認されたプリンシパルのリストが含まれた隠しテキスト・ファイルです。.k5login ファイルへの不正なアクセスは、ユーザ・アカウントの完全性を脅かします。

Kerberos のセキュア・デーモンを構成して、.k5login 承認を使うようにした場合、それぞれのユーザはホーム・ディレクトリに格納されている .k5login ファイルにプライベート承認リストを持っている必要があります。このリストによって、どのプリンシパルがユーザ・アカウントへアクセスできるか判別されます。

ユーザの .k5login ファイルを所有し、書き込み許可を持てるのは当該ユーザだけです。それ以外の場合、認証が失敗し、ユーザはアクセス権がないことを知らせるエラー・メッセージを受け取ります。

.k5login ファイルには、1 行ごとに 1 つのエントリがあります。エントリは、プリンシパル・データベースに登録されている通りのスペルで書かれたプリンシパルの完全名です。コメントを挿入したり、行末に空白を入れてはなりません。

例 C-3 は、通常、「jack@COMPANY.COM」としてログインしているが、プリンシパル「jack\_hill@COMPANY.COM」としてもアクセスしたい「Jack」というユーザに対して構成された .k5login ファイルの例です。さらに、「jill@HR.COMPANY.COM」というプリンシパルに対して、自身のアカウントへのアクセス許可を付与します。

### 例 C-3: .k5login ファイルの例

---

```
jack@COMPANY.COM
jack_hill@COMPANY.COM
jill@HR.COMPANY.COM
```

---

このアカウントにログインするには、Jack と Jill は次の情報を入力しなければなりません。

```
# rlogin hostname -l username
```

たとえば、Jack は次のコマンドを入力し、server1 というサーバにログインします。

```
# rlogin server1 -l jack
```

## C.5.5 ldapcd.conf ファイル

/etc/ldapcd.conf は、SSO のキャッシュ・パラメータが含まれたテキスト・ファイルです。ユーザが著しく長い時間 Tru64 UNIX システムの応答を待つような場合、テキスト・エディタを使って、Tru64 UNIX システムにある ldapcd.conf テキスト・ファイルのキャッシュ・パラメータの値を変更して Windows 2000 の SSO ソフトウェアをチューニングできます。

例 C-4 に、ldapcd.conf ファイルの例を示します。コメント行は、ナンバー記号 (#) で始まります。表 C-2 に、キャッシュ・パラメータの説明を示します。

### 例 C-4: ldapcd.conf ファイルの例

---

```
# 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
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"
```

---



表 C-2: キャッシュ・パラメータ

パラメータ	説明
connections	<p>キャッシュ・デーモンが Active Directory に対してオープンできる接続数。</p> <p>このエントリの値を増やすと、Active Directory に対してオープンできる接続も増加します。ただし、これにより、より多くのファイル記述子が使われ、Active Directory の負荷が増加します。</p> <p>通常、ワークステーション 1 台に対しては 4 つの接続、サーバ 1 台に対しては 15 の接続で十分です。</p> <p>省略時の設定は、4 つの接続です。</p>
pw_cachesize	<p>キャッシュに格納するユーザ・エントリ数の上限。</p> <p>最大ユーザ数の増減に従って、この値も増減させます。</p> <p>省略時の設定は、500 エントリです。</p>
pw_expirecache	<p>ユーザ・エントリをキャッシュする最長秒数。</p> <p>ユーザのエントリをキャッシュからすぐに取り出せるため、この値を増やすと Tru64 UNIX のパフォーマンスが向上します。</p> <p>前回使ってから時間が経っていないユーザ・アカウントを削除すると、このパラメータに指定された時間の間、ユーザのエントリがキャッシュに残ります。</p> <p>省略時の設定は、900 秒です。</p>
gr_cachesize	<p>キャッシュするグループ ID 数の上限。</p> <p>グループ ID をキャッシュからすぐに取り出せるため、この値を増やすと Tru64 UNIX のパフォーマンスが向上します。</p> <p>省略時の設定は、100 です。</p>
gr_expirecache	<p>グループ ID をキャッシュする最長秒数。</p> <p>省略時の設定は、900 秒です。</p>

#### 注意

/etc/ldapcd.conf ファイルのキャッシュ・パラメータの値を変更する場合、次のコマンドを入力し、キャッシュ・デーモンを再起動する必要があります。

```
# /sbin/init.d/ldapcd restart
```

---

詳細については、`ldapcd.conf(4)` を参照してください。

### C.5.6 `ldapusers.deny` ファイル

`/etc/ldapusers.deny` ファイルは、SSO がインストールされている時に、Tru64 UNIX によってのみ認証される各 Tru64 UNIX ユーザに対するエントリが含まれているテキスト・ファイルです。

1 行ごとにユーザ名を 1 つ入力します。また、このユーザ名は `/etc/passwd` ファイルのユーザ名と完全に一致しなければなりません。

コメントを作成するには、ナンバー記号 (#) を使います。ナンバー記号の後から行末までの文字列はすべて無視されます。また、空白行および行頭、行末の空白文字も無視されます。

例 C-5 に、`ldapusers.deny` ファイルの例を示します。

#### 例 C-5: `/etc/ldapusers.deny` ファイルの例

---

```
# 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
#           .
#           .
#           .
#           account_n
root
nobody
nobodyV
daemon
bin
uucp
uucpa
auth
cron
lp
tcb
adm
ris
wnn
pop
```

#### 例 C-5: /etc/ldapusers.deny ファイルの例 (続き)

---

```
imap
Peter
Dave
Evan
Martha
```

---

詳細については、`ldapusers.deny(4)` を参照してください。

## C.6 アカウントおよびグループの作成

Windows 2000 Server でユーザ・アカウント、コンピュータ・アカウント、およびグループを認証する場合は、これらのアカウントおよびグループを Windows 2000 Server に作成する必要があります。Tru64 UNIX システムに作成されたユーザ・アカウントおよびグループは、システムの Tru64 UNIX ユーザ・アカウントおよびグループ・ポリシに従って作成されるため、Windows 2000 サーバによる認証は行われません。

この項では、次の方法について説明します。

- ユーザ・アカウントの作成
- プリンシパルのパスワードの設定
- コンピュータ・アカウントの作成
- グループの作成

### C.6.1 ユーザ・アカウントの作成

Windows 2000 Server にユーザ・アカウントを作成するには、次の機能使います。

- Tru64 UNIX の `creacct` コマンド
- MMC (Microsoft Management Console) インタフェース

#### C.6.1.1 Tru64 UNIX の `creacct` コマンドを使用したユーザ・アカウントの作成

`creacct` コマンドを使用してユーザ・アカウントを作成するには、次のように入力します。

```
# creacct -a username
```

次の情報を入力するように求められます。

- Windows 2000 Server の管理者アカウントの名前とパスワード。
- ユーザ・アカウントに関するコメント。
- ユーザのホーム・ディレクトリとして使用される Tru64 UNIX のディレクトリ。

---

注意

---

有効なホーム・ディレクトリを指定しないと、ルート (/) ディレクトリが省略時の値として設定されます。ルート・ディレクトリの `.profile` ファイルまたは `.cshrc` ファイルでは、Tru64 UNIX コマンドが実行されるときに、`/sbin` ディレクトリを最初に調べるように環境パスが設定されています。その結果、ホーム・ディレクトリがルート・ディレクトリに設定されていると、`ls -l` コマンドは動作しません。

---

- ユーザ・アカウント用の Tru64 UNIX シェル。
- ユーザ・アカウントの Tru64 UNIX グループ ID (GID)。
- ユーザ・アカウントの Tru64 UNIX ユーザ ID (UID)。
- ユーザ・アカウントのパスワード。

詳細については、`creacct(1)` を参照してください。

#### C.6.1.2 MMC インタフェースを使用したユーザ・アカウントの作成

MMC インタフェースを使ってユーザ・アカウントを作成するには、次の手順を実行します。

1. 次のように、「Active Directory ユーザーとコンピュータ」ウィンドウを表示します。
  - a. 「スタート」をクリックします。
  - b. 「プログラム」から「管理ツール」、その後「Active Directory ユーザーとコンピュータ」を選択します。

「Active Directory ユーザーとコンピュータ」ウィンドウが表示されます。

- 必要であれば、「Users」フォルダを強調表示します。「操作」メニューで、「新規作成」の後に「ユーザー」を選択します。

「新しいオブジェクト — ユーザー」ウィンドウ (図 C-1) が表示されます。

図 C-1: 「新しいオブジェクト — ユーザー」ウィンドウ 必要な情報

新しいオブジェクト - ユーザー

作成先: ssonecorp.com/Users

姓(L): User

名(F): Peter    イニシャル(I):

フルネーム(A): User Peter

ユーザー ログオン名(U): User @ssonecorp.com

ユーザー ログオン名 (Windows 2000 以前)(W): SSO# User

< 戻る(B)    次へ(N) >    キャンセル

- 「新しいオブジェクト — ユーザー」ウィンドウに必要な情報を入力した後、[次へ] ボタンをクリックします。「名」と「イニシャル」フィールドは省略可能です。

次の「新しいオブジェクト — ユーザー」ウィンドウ (図 C-2) が表示されます。

図 C-2: 「新しいオブジェクト - ユーザー」ウィンドウ パスワード情報

新しいオブジェクト - ユーザー

作成先: sso.necorp.com/Users

パスワード(P): \*\*\*\*\*

パスワードの確認入力(C): \*\*\*\*\*

☒ ユーザーは次回ログオン時にパスワード変更が必要(M)

☐ ユーザーはパスワードを変更できない(S)

☐ パスワードを無期限にする(W)

☐ アカウントは無効(O)

< 戻る(B)    次へ(N) >    キャンセル

4. パスワードを入力し、パスワード・オプションを選択した後、[次へ] ボタンをクリックします。

確認画面が表示されます。

5. [次へ] ボタンをクリックして、ユーザ・アカウントを作成するか、[キャンセル] ボタンをクリックして、ユーザ・アカウントの作成を中止します。

作成を続けると、「Active Directory ユーザーとコンピュータ」ウィンドウが表示されます。

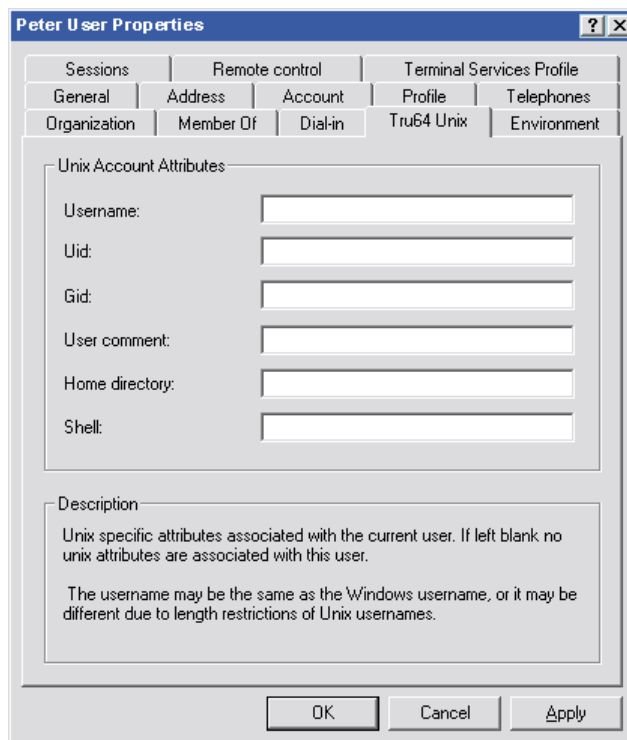
6. 作成したアカウント名をダブルクリックします。

作成したユーザのプロパティ・ダイアログ・ボックスが表示されます。

7. Tru64 UNIX タブをクリックします。

Tru64 UNIX ユーザのプロパティ・ダイアログ・ボックス (図 C-3) が表示されます。

☒ **C-3: Tru64 UNIX ユーザのプロパティ・ダイアログ・ボックス**



ZK-1677U-AI

8. Tru64 UNIX ユーザ・アカウント情報を入力します。Tru64 UNIX ユーザ・アカウントの制約が適用されます。たとえば、Tru64 UNIX ユーザ・アカウント名の文字数は、Tru64 UNIX オペレーティング・システムによって決められた最長文字数以上には設定できないなどの制約があります。Tru64 UNIX ユーザ・アカウントの制約の詳細については、『システム管理ガイド』を参照してください。

注意

有効なホーム・ディレクトリを指定しないと、ルート (/) ディレクトリが省略時の値として設定されます。ルート・ディレクトリの .profile ファイルまたは .cshrc ファイルでは、Tru64 UNIX コマンドが実行されるときに、/sbin ディレクトリを最初に調べるように環境パスが設定されています。そ

の結果、ホーム・ディレクトリがルート・ディレクトリに設定されていると、`ls -l` コマンドは動作しません。

属性の上にマウスを移動すると、ウィンドウの「Description」セクションに属性の説明が表示されます。

9. Tru64 UNIX ユーザ・アカウントの属性情報を入力し、[OK] ボタンをクリックします。

「Active Directory ユーザーとグループ」ウィンドウが表示されます。

## C.6.2 プリンシパルのパスワードの設定

プリンシパルのパスワードを設定するには、MMC (Microsoft Management Console) インタフェースか Tru64 UNIX の `creacct` コマンドが使えます。

`creacct` コマンドを使って、パスワードを設定するには、次のように入力します。

```
# creacct -s principal
```

Windows 2000 Server の管理者アカウントの名前とパスワードの後に、プリンシパルのパスワードを求められます。

`creacct` コマンドの詳細については、`creacct(1)` を参照してください。

MMC インタフェースを使用したパスワードの設定の詳細については、Windows 2000 のマニュアルを参照してください。

## C.6.3 コンピュータ・アカウントの作成

コンピュータ・アカウントを作成すると、Windows 2000 Server にアカウントが作成され、キー・サービス・テーブルに追加されます。コンピュータ・アカウントはすべて、Active Directory の「Computer」グループの下にある現在のドメインに追加されます。

コンピュータ・アカウントを作成するには、次の手順を実行します。

1. `creacct` コマンドを次のように入力してコンピュータの完全修飾名を指定します。

```
# creacct -h computer_name [-t keytable] [-u]
```



- 完全修飾名を指定しなかった場合は、`creacct` コマンドによって、コンピュータのローカル DNS 名に基づく名前が作成されます。
- `-t` オプションを指定しなかった場合は、省略時のサービス鍵テーブル・ファイル (`/krb5/v5srvtab`) にコンピュータ・アカウントが作成されます。
- `-u` オプションを指定して、新しいコンピュータ・アカウントのエントリが `/etc/ldapcd.conf` ファイルに追加されるようにします。

Windows 2000 Server で管理者特権を持つプリンシパルのユーザ名とパスワードを求められます。

2. ユーザ名とパスワードを入力します。

指定されたコンピュータが存在するかどうか、最初に Active Directory が検索されます。エントリが見つかった場合は、既存のエントリを置き換えるか、変更するように求められます。

3. 次のうちいずれかを選択します。
  - DNS コンピュータ名を変更します。
  - 既存のコンピュータ・アカウントを新しいコンピュータ・アカウントと置き換えます。既存のコンピュータ・アカウントを置き換える場合、DNS ホスト名を取得するために、`creacct` コマンドによって Active Directory が検索されます。その後、DNS ホスト名を変更するように求められます。既存の DNS ホスト名を再入力するか新しいホスト名を入力します。

詳細については、`creacct(1)` を参照してください。

## C.6.4 グループの作成

Windows 2000 Server にグループを作成するには、MMC インタフェースを使う必要があります。Tru64 UNIX および Windows 2000 のユーザがメンバとなるグループを作成できます。

グループを作成するには、次の手順を実行します。

1. 次のように、「Active Directory ユーザーとコンピュータ」ウィンドウを表示します。
  - a. 「スタート」をクリックします。

- b. 「プログラム」から「管理ツール」、その後「Active Directory ユーザーとコンピュータ」を選択します。

「Active Directory ユーザーとコンピュータ」ウィンドウが表示されます。

2. 必要であれば、「Users」フォルダを強調表示します。「操作」メニューで、「新規作成」の後に「グループ」を選択します。

「新しいオブジェクト — グループ」ウィンドウが表示されます。

3. グループ名を入力し、[OK] ボタンをクリックします。

「Active Directory ユーザーとコンピュータ」ウィンドウが表示されます。

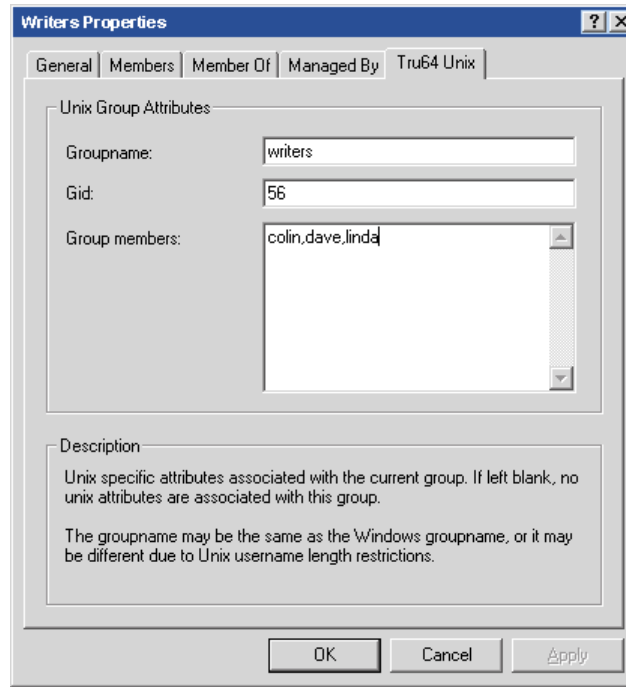
4. 作成したグループ名をダブルクリックします。

作成したグループのプロパティ・ダイアログ・ボックスが表示されます。

5. Tru64 UNIX タブをクリックします。

作成したグループの Tru64 UNIX プロパティ・ダイアログ・ボックス (図 C-4) が表示されます。

図 C-4: グループのプロパティ・ダイアログ・ボックス



ZK-1678U-AI

6. Tru64 UNIX グループ情報を入力します。  
属性の上にマウスを移動すると、ウィンドウの「Description」セクションに属性の説明が表示されます。
7. Tru64 UNIX グループの属性情報を入力し、[OK] ボタンをクリックします。  
「Active Directory ユーザーとグループ」ウィンドウが表示されます。

## C.7 SSO ソフトウェアの管理

この項では、次の方法について説明します。

- チケットの要求
- チケットの表示
- 信任状キャッシュの削除
- サービス鍵テーブルのエントリの管理

### C.7.1 チケットの要求

/sbin/kinit コマンドを使って、チケットを要求します。

- 初回チケットを要求するには、次のように入力します。

```
# kinit [-c cachename]
```

たとえば、省略時の信任状キャッシュからチケットを要求するには、次のように入力します。

```
# kinit
```

- 有効期限を定めたチケットを要求するには、次のように入力します。

```
# kinit -l nwndnhmns
```

たとえば、45 時間 30 分の有効期間のあるチケットを要求するには、次のように入力します。

```
# kinit -l 45h30m
```

- 先日付けチケットを要求するには、次のように入力します。

```
# kinit -d nwndnhmns principal domain
```

たとえば、「COMPANY.COM」というドメインの「mary/admin」というプリンシパルに対して、1 時間以内に開始する先日付けチケットを要求するには、次のように入力します。

```
# kinit -d 1h mary/admin@COMPANY.COM
```

有効期間および先日付けオプションの構文は、[nw] [nd] [nh] [nm] [ns] です。ただし、w = 週、d = 日、h = 時間、m = 分、s = 秒です。空白文字を使う場合は、引用符で囲む必要があります。たとえば、"1w 2d 3h 4m 5s" のように指定します。特に指定されなければ、省略時の有効期間および先日付けオプションは 1 時間単位です。

要求された先日付け期間がサーバのクロック・スキューの値 (通常、5 分) よりも短ければ、-d オプションが指定されていないかのように、チケットの開始時間が現在の時間に設定されてから、発行されます。

チケットの要求の詳細については、kinit(1) を参照してください。

### C.7.2 チケットの表示

/sbin/klist コマンドを使って、信任状キャッシュに格納されているチケットを表示します。

- 特定の信任状キャッシュに格納されているチケットをすべて表示するには、次のように入力します。

```
# klist -a [-c cachename]
```

たとえば、`/var/tmp/mycache` という信任状キャッシュにあるチケットをすべて表示するには、次のように入力します。

```
# klist -a -c /var/tmp/mycache
```

- 信任状キャッシュのすべてのチケットをフラグとアドレス付きで表示するには、次のように入力します。

```
# klist -a -f [-c cachename]
```

チケットの表示の詳細については、`klist(1)` を参照してください。

### C.7.3 信任状キャッシュの削除

特定の信任状キャッシュのチケットをすべて破棄し、信任状キャッシュを削除するには、次のように入力します。

```
# kdestroy [-c cachename]
```

たとえば、`/var/tmp/mycache` という信任状キャッシュにあるチケットをすべて破棄するには、次のように入力します。

```
# kdestroy -c /var/tmp/mycache
```

チケットの削除の詳細については、`kdestroy(1)` を参照してください。

### C.7.4 サービス鍵テーブルの管理

`/sbin/ktutil` コマンドを使って、サービス鍵テーブル・ファイルのエントリの表示や削除、サービス鍵テーブルの破棄、およびサービス鍵テーブルのマージを行います。

`/usr/sbin/creacct` コマンドを使って、Windows 2000 Server からキーを抽出し、サービス鍵テーブルに追加します。`/usr/sbin/creacct` コマンドを使用すると、Windows 2000 Server で管理者特権を持つプリンシパルのユーザ名とパスワードの入力を求められます。

省略時のサービス鍵テーブル・ファイルは、`/krb5/v5srvtab` です。`root` ユーザ・アカウントがこのファイルを所有します。

- サービス鍵テーブルのチケットをすべて表示するには、次のように入力します。

```
# ktutil [-t keytable]
```

たとえば、省略時のサービス鍵テーブルのチケットをすべて表示するには、次のように入力します。

```
# ktutil
```

- Windows 2000 Server から Tru64 UNIX ホストに対するキーを抽出し、サービス鍵テーブルに追加するには、次のように入力します。

```
# creacct [-t keytable] -x host/fully_qualified_host_name
```

たとえば、server1.company.com という Tru64 UNIX ホストに対するキーを抽出し、省略時のサービス鍵テーブルに追加するには、次のように入力します。

```
# creacct -x host/server1.company.com
```

- Windows 2000 Server からプリンシパルに対するキーを抽出し、サービス鍵テーブルに追加するには、次のように入力します。

```
# creacct [-t keytable] -x principal/fully_qualified_host_name
```

たとえば、w2kserverhost.company.com というシステムの user1 というユーザ・アカウントに対するキーを抽出し、省略時のサービス鍵テーブルに追加するには、次のように入力します。

```
# creacct -x user1/w2kserverhost.company.com
```

- 特定のプリンシパルに対するチケットをサービス鍵テーブルから削除するには、次のように入力します。

```
# ktutil -t keytable -d principal
```

たとえば、COMPANY.COM というドメインの mary/admin というプリンシパルに対するチケットを省略時のサービス鍵テーブルから削除するには、次のように入力します。

```
# ktutil -t WFILE:/krb5/v5srvtab -d mary/admin@COMPANY.COM
```

- サービス鍵テーブルを破棄するには、次のように入力します。

```
# ktutil -D -t keytable
```

たとえば、/krb5/mytable というサービス鍵テーブルを破棄するには、次のように入力します。

```
# ktutil -D -t WFILE:/krb5/mytable
```

- サービス鍵テーブルをマージするには、次のように入力します。

```
# ktutil -c from_keytable -t to_keytable
```

たとえば、`/krb5/srvtable` というサービス鍵テーブルのエントリをすべて省略時のサービス鍵テーブル `/krb5/v5srvtab` のエントリとマージするには、次のように入力します。

```
# ktutil -c /krb5/srvtable -t WFILE:/krb5/v5srvtab
```

サービス鍵テーブルのマージの詳細については、`ktutil(1)` および `creacct(1)` を参照してください。

## C.8 SSO ソフトウェアのトラブルシューティング

SSO ソフトウェアの相互依存関係は複数のオペレーティング・システムにまたがるため、C.4 節で説明されているとおりにセットアップ手順を実行することが大切です。この手順どおりに実行されないと、SSO ソフトウェアが正しく動作しません。その場合は、`invalid login` という応答メッセージが表示されます。SSO に関わる不具合の大半はセットアップに原因があるため、SSO ソフトウェアのセットアップを確認すればすぐに分かります。

### C.8.1 SSO の構成パラメータの問題

`/usr/sbin/creacct` コマンドまたは `/usr/sbin/w2ksetup` スクリプトを実行したときに、管理者アカウントが有効なものとして受け入れられなかった場合には、この管理者アカウントが Windows 2000 のドメイン・コントローラの `Administrators`、`DnsAdmins`、`Domain Admins`、`Domain Users`、`Users` グループのメンバであるか確認します。

Windows 2000 Server で Active Directory を作成する場合、名前は小文字で指定する必要があります。Kerberos レalm 名であるこの名前に対するその他の参照は、すべてのプラットフォームで大文字で入力される必要があります。

### C.8.2 kinit コマンドの使用および Tru64 UNIX の初回チケットの取得に関する問題

`kinit` コマンドを入力したときや、Tru64 UNIX システムにログインするときに、初回チケットの取得で問題が発生した場合には、次の手順を実行します。

1. 次のコマンドを入力して、`/etc/sia/matrix.conf` ファイルの内容を表示し、その中で LDAP と Kerberos に関連するエントリを探すことで、`matrix.conf` ファイルに LDAP と Kerberos のエントリがあるかどうかを確認します。

```
# more /etc/sia/matrix.conf
```

/etc/sia/matrix.conf ファイルに該当するエントリがない場合は、次のコマンドを入力します。

```
# 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. KDC のエントリが /etc/hosts ファイルにあるかどうかを確認します。たとえば、次のように入力します。

```
12.345.67.890 w2khost.mycompany.com w2khost
```

最初の /usr/sbin/w2ksetup ユーティリティを実行し、管理者の名前とパスワードを入力した後に次のエラー・メッセージが表示された場合は、通常この手順を実行すると問題が解決します。

```
ldap_gssapi_bind: Operations error
```

3. /krb5/krb.conf ファイルおよび /krb/krb.realms ファイルが正しく設定されているかどうかを確認します。

krb.conf ファイルの例を示します。

```
MYCOMPANY.COM
MYCOMPANY.COM w2khost.mycompany.com admin server
```

krb.realms ファイルの例を示します。

```
*.mycompany.com MYCOMPANY.COM
```

4. KDC とクライアントの時刻が同じかどうかを確認します。2 ~ 3 分の時差は許容範囲ですが、5 分以上の時差があってはなりません。
5. 次のコマンドを入力し、キーを抽出することによって v5srvtab ファイルが正しく動作しているかどうかを確認します。

```
# kinit -k
```

コマンドが失敗した場合は、/usr/sbin/w2ksetup スクリプトを再実行するか、/krb5/v5srvtab ファイルを削除してから、次のコマンドを入力します。

```
# /usr/sbin/creacct -h mymachinename -u
```



### C.8.3 Tru64 UNIX へのパスワードの入力

KDC からチケットを取得できても、その後の Tru64 UNIX システムへのログイン時にパスワードの入力を求め続けられる場合は、次の手順を実行します。

1. 次のコマンドを入力して、ターゲットのサービス・チケットを取得できることを確認します。

```
# kinit -S host/targetmachinename.mycompany.com
```

このコマンドが失敗した場合、アクセスしようとしているホストは Kerberos レalmに含まれていません。これを是正するには、次のコマンドを入力します。

```
# /usr/sbin/creacct -h targetmachinename.mycompany.com -u
```

2. /krb5/krb.conf ファイルの最初の行が、現在のホストとターゲットのホストと同じであることを確認します。
3. /etc/ldapusers.deny ファイルまたはユーザのホーム・ディレクトリの .k5login ファイルにログインを妨げるエントリがないことを確認します。
4. /etc/services および /etc/inetd.conf ファイルが正しく設定されており、次のエントリが含まれていることを確認します。

- /etc/services ファイル

```
kerberos 88/udp
kerberos 88/tcp
kerberos_master 749/udp
kerberos_adm 749/udp
kerberos_adm 749/tcp
```

- /etc/inetd.conf ファイル

```
kshell stream tcp nowait root /usr/sbin/rshd rshd -K
klogin stream tcp nowait root /usr/sbin/rlogind rlogind -K
```

---

#### 注意

- 接続を暗号化する場合は、inetd.conf ファイルの 2 行の行末に -x を追記します。
- inetd.conf ファイルを変更した場合には、次のコマンドを入力して、ネットワークを再起動する必要があります。

```
# rcinet restart
```

---

## C.8.4 TruCluster における SSO の問題

SSO ソフトウェアが TruCluster のクラスタ・メンバで正しく動作しない場合は、次の手順を実行します。

1. 次のエントリが、それぞれのクラスタ・メンバに対して定義され、  
/etc/ldapcd.conf ファイルに含まれていることを確認します。

```
directory:  
searchbase:  
machine_acctname:  
machine_dn:
```

対象クラスタ・メンバに対応するエントリがない場合は、そのクラスタ・メンバで /usr/sbin/w2ksetup スクリプトを再実行します。

2. 次のコマンドを入力し、それぞれのクラスタ・メンバに対応するエントリが表示されることを確認します。

```
# /usr/sbin/ktutil
```

それぞれのクラスタ・メンバのエントリが表示されない場合は、次のコマンドを入力します。

```
# /usr/sbin/creacct -h missingmembername -u
```

# D

## LDAP (Lightweight Directory Access Protocol)

LDAP (Lightweight Directory Access Protocol) は、TCP/IP 上で動作するインターネット標準の分散クライアント/サーバ型ディレクトリ・サービスのプロトコルです。LDAP サーバはディレクトリ内のエントリを管理し、ネットワーク上の LDAP クライアントが情報を利用できるようにします。LDAP サーバは、ユーザの識別や認証に必要なユーザ情報の集中リポジトリとして使用することができます。

この付録には、次の情報が含まれています。

- LDAP の概要
- Tru64 UNIX LDAP クライアント・ソフトウェアのインストール
- Tru64 UNIX LDAP クライアント・ソフトウェアの構成
- LDAP クライアント・デーモンの管理
- アクセス制御の管理

### D.1 LDAP の概要

LDAP サーバは NIS (Network Information Services) と似ていますが、NIS に比べて次のような利点があります。

- LDAP ディレクトリは、高いスケーラビリティを持っています。
- LDAP ディレクトリは動的に更新され、マップの再構築やネットワークへのマップのプッシュが不要なため、管理者の作業時間が短縮されます。また、変更がほとんど瞬時に反映されます。
- LDAP ディレクトリのデータベースは、ユーザ関連情報の管理を集中化するために使用できます。
- 属性変更の可否を属性レベルで制御できます。これにより、重要性の低い情報 (使用するログイン・シェルやメール転送アドレスなど) については、ユーザ自身による変更を許可できます。一方、機密性の高い情報

(UID, GID, ユーザのホーム・ディレクトリなど)の変更は、承認されたディレクトリ管理者だけに限定できます。

- 複数の LDAP サーバをセットアップして、ディレクトリ内のデータの可用性を高めることができます。レプリケーションと呼ばれる処理によって、すべての LDAP サーバにディレクトリの同一コピーがあることを保証できます。LDAP サーバは相互にバインドされ、ディレクトリに対する変更は標準の LDAP コマンドを使って伝達されます。

Tru64 UNIX システムは、LDAP クライアントおよび LDAP サーバとして構成できます。この項では、LDAP サーバが既に構成されていることを前提に、Tru64 UNIX を LDAP クライアントとして構成する方法を説明します。LDAP サーバの構成情報については、LDAP サーバのマニュアルを参照してください。

ユーザが LDAP クライアントとして構成された Tru64 UNIX システムにログインするためにユーザ名とパスワードを入力すると、LDAP クライアントはそのユーザ情報を LDAP サーバに送信して認証処理を依頼します。LDAP サーバは受信したユーザ情報を LDAP ディレクトリ内のエントリと照合します。一致するエントリが存在し、パスワードが正しい場合は、認証に成功し、ユーザはログインすることができます。一致するエントリが存在しない場合、またはパスワードが不正な場合は、認証に失敗し、ユーザはログインすることができません。認証処理の結果は、LDAP サーバから LDAP クライアントに返されます。

## D.2 Tru64 UNIX LDAP クライアント・ソフトウェアのインストール

LDAP クライアント・ソフトウェアをインストールするには、オプションの LDAP 認証 (ネットワーク・サーバ/通信) サブセット (OSFLDPAUTH $nnn$ ) をインストールする必要があります。このサブセットは、Tru64 UNIX 基本オペレーティング・システムを収録した CD-ROM に入っています。

$nnn$  は Tru64 UNIX のバージョン番号を表しています。最新のバージョン番号については、『リリース・ノート』を参照してください。サブセットのインストールの詳細については、『インストール・ガイド』を参照してください。

## D.3 Tru64 UNIX LDAP クライアント・ソフトウェアの構成

LDAP クライアント・ソフトウェアを構成する必要があります。LDAP クライアント・ソフトウェアを構成するには、次の作業を実行する必要があります。

- LDAP クライアント構成ファイル (`/etc/ldapcd.conf`) の更新。認証に使用する LDAP サーバに関する情報を指定します。省略時の `/etc/ldapcd.conf` ファイルは、LDAP クライアント・ソフトウェアのインストール時に作成されます。
- LDAP クライアント・ランタイム構成変数の設定。

### D.3.1 `ldapcd.conf` ファイルの更新

`/etc/ldapcd.conf` ファイル内には複数の属性があり、省略時の値を使うことも値を指定することもできますが、次の属性については値を指定する必要があります。

<code>directory</code>	ユーザ認証に使用する LDAP ディレクトリ・サーバのホスト名。
<code>searchbase</code>	ディレクトリ・サーバのデータベース内でユーザ情報が保存されているブランチのルート。
<code>port</code>	省略時のディレクトリ・サーバ・ポート。ディレクトリ・サーバのために使用しているポートと一致させる必要があります。
<code>machine_dn</code> および <code>machine_password</code>	<code>machine_dn</code> (識別名) と <code>machine_password</code> は、 <code>ldapcd</code> キャッシュ・デーモンがディレクトリ・サーバにバインドしてディレクトリからの情報検索および情報取得を実行するときに使用します。これらの値は、インストール時のディレクトリ・サーバの初期構成で設定されます。通常は、ディレクトリ・サーバの構成ファイル ( <code>sladpd.conf</code> ) に指定されたルート識別名とパスワードを使用します。

`/etc/ldapcd.conf` ファイルは、次のいずれかの方法で更新します。

- SysMan Menu のオプションを使用します。メニューを展開して「一般的なタスク」 - 「Setup LDAP Configuration」を選択します。このオプションを選択すると、「LDAP Configuration」というタイトルのウィンドウが表示され、ウィンドウ内に LDAP 構成属性のリストが表示されます。リストから属性を選択すると、現在の属性値の表示と新しい属性値を入力するための領域を含むダイアログ・ボックスが表示されます。

[OK] を選択すると属性値が更新され、「LDAP Configuration」ウィンドウが閉じて SysMan Menu に戻ります。

- テキスト・エディタを使って /etc/ldapcd.conf ファイルを変更します。

例 D-1 に ldapcd.conf 構成ファイルの例を示します。/etc/ldapcd.conf ファイルを変更するときは、SysMan Menu のオプションを使用することをお勧めします。ldapcd.conf ファイルの属性値を変更したときは、LDAP クライアント・デーモンを再起動する必要があります。LDAP クライアント・デーモンを再起動する方法については、D.4 節を参照してください。

#### 例 D-1: ldapcd.conf ファイルの例

---

```
#
# 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
#
pw_cachesize:    2000[6]
pw_expirecache: 120
gr_cachesize:    100
gr_expirecache: 600
:
machine_dn:      "cn=Directory Manager"[7]
machine_pass:    "password"

#
```

## 例 D-1: ldapcd.conf ファイルの例 (続き)

---

```
:  
  
# the objectClass name of a password entry  
pw_ouclass:      posixAccount 8  
  
# name mappings for password attribute fields  
pw_username:     uid 9  
pw_password:     userPassword 10  
pw_uid:          uidNumber  
pw_gid:          gidNumber  
pw_quota:        description  
pw_gecos:        geccos  
pw_homedir:      homedirectory  
pw_shell:        loginshell  
  
# the objectClass name of a group entry  
gr_ouclass:      posixGroup 11  
  
# name mappings for group attribute fields  
gr_ouclass:      unixGroup 12  
gr_name:         cn  
gr_password:     userPassword  
gr_gid:          gidNumber  
gr_members:      MemberUID
```

---

- 1** ユーザ認証に使用する LDAP ディレクトリ・サーバのホスト名。
- 2** ディレクトリ・サーバのデータベース内でユーザ情報が保存されているブランチのルート。
- 3** 省略時のディレクトリ・サーバ・ポート。ディレクトリ・サーバのために使用しているポートと一致させる必要があります。
- 4** ldapcd キャッシュ・デーモンが管理する、ディレクトリ・サーバに対して開かれた接続の最大数。
- 5** ldapcd キャッシュ・デーモンが管理するスレッドの最大数。各スレッドはローカル・プログラムに対する接続を 1 つだけ処理します。スレッドの数を増やすと LDAP キャッシュ・デーモンからの応答速度が向上しますが、必要なメモリ量が増えます。数多くの接続を必要とするサービ

ス (メール・サービスなど) を実行している場合は、(システムに十分なメモリがあれば) スレッドの最大数を 64 以上に設定してください。

- ⑥ `pw_cachesize` の値は、`passwd` エントリの最大キャッシュ数を示します。`pw_expirecache` の値は、`ldapcd` キャッシュ・デーモンが各 `passwd` エントリのキャッシュをチェックする最大期間を示します。`pw_expirecache` の値を超えると、`ldapcd` デーモンは要求された `passwd` エントリを検索するためにサーバに制御を戻します。  
  
`gr_cachesize` および `gr_expirecache` はそれぞれ `pw_cachesize` および `pw_expirecache` と同じ効果を持つ値で、`group` エントリに対して適用されます。
- ⑦ `machine_dn` の値は、`ldapcd` キャッシュ・デーモンがディレクトリ・サーバにバインドしてディレクトリからの情報検索および情報取得を実行するときに使用する識別名です。各システムに特定の DN を使用させることにより、どのコンピュータがどのような目的でディレクトリにアクセスしているかを特定することができます。また、ディレクトリへの読み取りアクセスや検索アクセスをコンピュータのアカウントに基づいて制御することもできます。
- ⑧ サーバ上の拡張スキーマに含まれる UNIX アカountの属性を定義するオブジェクト・クラスの名前。
- ⑨ LDAP 属性名 (右側) が、`getpwent` の呼び出しによって返される `passwd` 構造体のフィールド (左側) にマッピングされます。
- ⑩ `userPassword` 属性には、暗号化されたパスワードだけが格納されます。
- ⑪ サーバ上で定義された拡張スキーマに含まれる UNIX グループの属性を定義するオブジェクト・クラスの名前。
- ⑫ LDAP 属性名 (右側) が、`getgrent(3)` の呼び出しによって返される `group` 構造体のフィールド (左側) にマッピングされます。

### D.3.2 LDAP ランタイム構成変数の設定

`/etc/ldapcd.conf` ファイルを初めて更新した後は、LDAP クライアント・デーモンを起動する前に次のコマンドを入力して LDAP ランタイム構成変数を設定する必要があります。

```
# /usr/sbin/rcmgr set LDAPCD_CONF yes
```



このコマンドは 1 回だけ実行すれば済みます。/etc/ldapcd.conf ファイルを変更しても、このコマンドを再実行する必要はありません。

## D.4 LDAP クライアント・デーモンの管理

LDAP クライアント・デーモンを起動するには、次のコマンドを入力します。

```
# /sbin/init.d/ldapcd start
```

LDAP クライアント・デーモンを初めて起動すると、次の処理が実行されます。

- /etc/sia/matrix.conf ファイルを更新して、LDAP セキュリティ統合アーキテクチャ (SIA) メカニズムを追加します。
- /etc/inittab ファイルに次のエントリを追加して、システム起動時に LDAP クライアント・デーモンが自動的に起動されるようにします。

```
ldapcd:34:respawn:/usr/sbin/ldapcd -D > /dev/console 2>&1
```

LDAP クライアント・デーモンを停止するには、次のコマンドを入力します。

```
# /sbin/init.d/ldapcd stop
```

LDAP クライアント・デーモンを再起動するには、次のコマンドを入力します。

```
# /sbin/init.d/ldapcd restart
```

## D.5 アクセス制御の管理

特に指定しない限り、LDAP データベースで定義されたユーザはそのデータベースが LDAP 認証に使われるすべてのシステムにログインできます。ユーザ・アクセスを特定のシステムに限定する場合は、アクセス制御ファイルの /etc/ldapusers.deny および /etc/ldapusers.allow を使用します。

### D.5.1 ldapusers.deny ファイル

/etc/ldapusers.deny は、LDAP 認証では認証されない Tru64 UNIX ユーザの名前を入力するためのテキスト・ファイルです。ldapusers.deny ファイルに指定されたユーザは、そのシステムに構成された Tru64 UNIX のセキュリティ・メカニズムで認証され、LDAP 認証から除外されます。

省略時の /etc/ldapusers.deny ファイルは、LDAP クライアント・ソフトウェアのインストール時に作成されます。ユーザ名は、1 行に 1 つずつ入

力し、`/etc/passwd` ファイルのユーザ名と正確に一致させる必要があります。コメントを入力するときは、ナンバー記号(`#`)を使用します。ナンバー記号の直後から行末までの文字は無視されます。空白行および行の先頭や末尾の空白も無視されます。

例 D-2 に省略時の `/etc/ldapusers.deny` ファイルを示します。

#### 例 D-2: 省略時の `ldapusers.deny` ファイル

---

```
#
# ldapusers.deny - list of users who area not allowed to authenticate on
#                  this system via LDAP authentication (libsldap.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
```

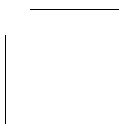
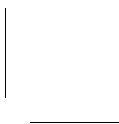
---

### D.5.2 `ldapusers.allow` ファイル

一部のユーザを除いてすべてのアクセスを拒否する場合は、`/etc/ldapusers.allow` ファイルを作成する必要があります。`/etc/ldapusers.allow` は、LDAP 認証でのみ認証される Tru64 UNIX ユーザの名前を入力するためのテキスト・ファイルです。`/etc/ldapusers.allow` ファイルがシステム上に存在する場合は、そのファイルに指定されたユーザのログインだけが LDAP 認証によって許可されます。`/etc/ldapusers.allow` が空の場合も有効であることに注意し

てください。このファイルが存在すること自体が強力なアクセス制御ルールになります。

ユーザ名は、1 行に 1 つずつ入力し、`/etc/passwd` ファイルのユーザ名と正確に一致させる必要があります。コメントを入力するときは、ナンバー記号 (#) を使用します。ナンバー記号の直後から行末までの文字は無視されます。空白行および行の先頭や末尾の空白も無視されます。



---

## C2 レベル・セキュリティの構成

この付録では、『*Trusted Computer System Evaluation Criteria*』（『*Orange Book*』とも呼ばれる）でも説明されているように、エンハンスド・セキュリティを使って、C2 レベル以上のセキュリティに見合うようにシステムを構成する方法について説明しています。『*Orange Book*』のオンライン版は、<http://nsi.org/Library/Compsec/orangebo.txt> で提供されています。

Tru64 UNIX は、『*Information Technology Security Evaluation Criteria*』（ITSEC）に定義されている、F-C2 機能クラスにも適合するように設計されています。

サイトのセキュリティ・ポリシに従い、C2 ネットワーク、および適切な物理的セキュリティを実施してシステムを使用すると、C2 レベルの環境を実現し、既存の Tru64 UNIX セキュリティ・メカニズムとの互換性を確保しながら、ユーザおよびシステムの情報の保護を強化できます。

Tru64 UNIX 製品の最新の評価および認定ステータスについては、当社の営業担当にお問い合わせください。

この付録には、次の情報があります。

- サイトのセキュリティ・ポリシの確立
- 最小限の C2 構成
- 初期構成
- 物理的セキュリティ
- アプリケーション
- 定期的に行うセキュリティ管理手順
- 参照ドキュメントとチェック・ツール

## E.1 セキュリティ・ポリシーの確立

セキュリティ・ポリシーは、組織内でコンピュータ環境を保守する方法や、機密情報を管理、保護、および配布する方法を定めた、規則および実施に関する規定です。セキュリティ・ポリシーには、次の情報が含まれる必要があります。

- セキュリティ・ポリシーを保守し変更する手順の文書化。
- 侵入またはその他のセキュリティ侵害が発生したときにシステム管理者がとる行動の確立。
- 次の事項を含む、サイトの監査ポリシーの決定。
  - 監査対象のユーザ操作。
  - ローカル定義の監査イベント。
  - 監査ログの格納場所。
  - 監査ログの見直し手順。
  - 認識できないアカウント名 (`login_uname`) でログインしようとしたときに監査が必要かどうか。この属性を使うと、プロンプトの表示のタイミングとパスワードの入力がずれた場合も監査ログに記録が残ります。
- サービス・アクセス・ポリシーの確立 (管理、パスワード)。
- システムの `umask` の決定 (022 がシステムの省略時の設定)。
- システムおよびサイトのパスワードなど、完全性を確認するスケジュールの確立。
- システムの境界の定義と、境界の間のインタフェース (たとえば、`telnet`, `ftp`) の定義。
- 磁気メディアのポリシーの確立 (特に、リムーバブル・メディアの場合)。
- システムにインストールするアプリケーション・ソフトウェアの決定とセキュリティ上の問題の確認。
- ユーザのパスワード・ポリシーの決定。次のような考慮事項があります。
  - 長いパスワードは解読されにくいですが、必然的に書き留めてしまいます。
  - ユーザが選択したパスワードを使用する場合、そのパスワードを知っているのは 1 人だけです。

- 機械生成のパスワードは解読されにくいですが、覚えるのも困難で、書き留めてしまう可能性があります。
- システムのログイン制御の決定。
- システムのスタートアップ、シャットダウン、アップグレードの手順の確立。
- バックアップおよび回復の手順の確立。
- システム管理者 (root アクセス) の決定、および管理者の役割の決定。
- 「アカウント・マネージャ」プログラムを使ってユーザ・アカウントを作成するための、1つ以上のセキュア・アカウント・プロトタイプ (ローカル・テンプレート) の確立。
- /usr/skel ディレクトリ内のスタートアップ・ファイルでのセキュア・アカウント・テンプレートの確立。
- システム上の各オブジェクトのアクセス制限の決定。
- システムのグループの確立。
- システム上の各ユーザのアクセス制限の決定。
- UID または GID を設定できる、システム上のすべてのプログラムへの参照の記録。
- システム上のファイル・システムのエクスポート制限の決定。
- システム上のサブジェクトおよびオブジェクトの変更管理手順の確立。
- 物理的アクセス変更の手順の確立。
- システムおよびコンソールの物理的アクセス要件の確立。
- ネットワーク構成要素の物理的アクセス要件の確立。
- ネットワーク・セキュリティ・ポリシの確立。
- システム上で実行されるリモート・アクセス・プログラム (ftp , telnet など) の決定。
- システムへのアクセス権を持つリモート・システムの決定。
- システムおよびサイトのコンソール・パスワード要件の決定。
- モデム・ポリシの確立 (認証、着信アクセスおよび発信アクセスの構成を考慮する必要があります)。

- 「ユーザ・セキュリティ・トレーニング」コースまたはサイトに関するマニュアルの作成。
- オペレーティング・システム，データベース，アプリケーション・メニューに対するユーザによるアクセス方法の文書化。
- サイトのセキュア・デバイスの決定。

システムを構成した後は，構成ファイルの変更は少なくし，計画的に変更します。システムの定期的なセキュリティ見直し時に，元の構成ファイルと現在のファイルの内容と許可を比較してください。次のファイルのリストを取得してセキュリティ・ポリシに添付することで，基本システムとネットワーク構成を文書化してください。

```

/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
/etc/remote
/etc/resolv.conf
/etc/rc.config
/etc/rc.site
/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/audit_loc

```

使用している場合

省略可能，/etc/rc.config と組み合わせて使用



```

/etc/securettys
/etc/svc.conf
/tcb/*
/usr/adm/messages
/var/spool/uucp/Permissions      UUCP を使用している場合
/var/spool/uucp/Systems          UUCP を使用している場合
/var/spool/uucp/remote.unknown   UUCP を使用している場合
/var/adm/cron/at.allow
/var/adm/cron/at.deny
/var/adm/cron/cron.allow
/var/adm/cron/cron.deny
/var/adm/crontab/                これらのディレクトリの任意のファイル
/var/tcb/*
/var/yp/src/*

```

## E.2 最小限の C2 構成

『*Orange Book*』に示されている最小限の C2 システムの要件に従った場合の Tru64 UNIX の構成は次のとおりです。

- サイトのセキュリティ・ポリシの要件が満たされるのは、『*Site Security Handbook (RFC 1244)*』で説明しているとおり、サイトのセキュリティ・ポリシを確立したときです。セキュリティ・ポリシは、文書化する必要があります。
- ユーザは自身のパスワードを変更することができ、パスワードは機械生成されていなければなりません(『*Green Book*』で推奨)。パスワードの構成についての詳細は、E.3.2 項を参照してください。
- ユーザの最後のログインを通知するという要件は、エンハンスド・セキュリティを構成したときに満たされます。
- 任意アクセス制御の要件は、システムに ACL (アクセス制御リスト) を構成することで満たされます。ACL の構成の詳細については、2.3 節を参照してください。小規模のシステム (32 ユーザ未満) では、`/usr/groups` を使用しないでください。
- オブジェクト再利用要件では、`xhost` エントリなしでワークステーションを構成する必要があります。
- 共有メモリの分離を有効にしなければなりません。`secconfig` でセグメント共有を無効にするか質問されたときに `yes` と答えると、分離を有効にできます。

- 監査サブシステムを構成し、使用できるようにする必要があります。監査サブシステムについては、第 3 章 を参照してください。
- トラストッド・コンピューティング・ベース (TCB) の完全性を確認する機能は、サイトのセキュリティ・ポリシーに従って、定期的に `fverify` コマンドおよび `authck` コマンドを実行することで実現できます。

## E.3 初期構成

Tru64 UNIX ソフトウェア・サブセット (オプションのエンハンスド・セキュリティとドキュメンテーション拡張サブセットを含む) をシステムにインストールした後に、ソフトウェアの構成を開始します。構成中に選択する項目のいくつかは、システムのセキュリティに影響を及ぼします。最大限の実用的なセキュリティがシステムに必要であると仮定して説明します。以降の項では、セキュリティに関して考慮すべき事柄を推奨構成について説明します。

### E.3.1 一般的な構成

一般的なシステム構成には次の事項が含まれます。

- ディレクトリ `/tmp`、`/var/tmp`、および `/var/spool` は、ルート (`/`) ディレクトリおよび `/usr` ディレクトリのあるファイル・システム以外に置いてください。
- JAVA を有効にして Netscape Navigator を実行しないでください。よく知られている安全なサイトに接続する場合だけ、JAVA を有効にしてください。
- できるだけシステムをインターネットに接続しないようにしてください。

### E.3.2 `secconfig` を使用した、エンハンスド・パスワードと認証

サイトのセキュリティ・ポリシーに合うように、エンハンスド・パスワード属性を選択します。詳細については A.2.2 項 を参照してください。

次のパスワード属性を使用します (省略時の設定は、`/etc/auth/system/default` ファイルに定義されています)。

- ユーザ選択または機械生成のどちらかのパスワードを選択し、次のように構成します。
  - ユーザ選択パスワードの場合 (`/etc/auth/system/default` ファイル内に `u_pickpw` フィールドあり)、長さの下限を 8 文字

(u\_minlen#8) に設定し，上限を 80 文字 (u\_maxlen#80) に設定します。

- 機械生成パスワードの場合 (/etc/auth/system/default ファイル内に u\_pickpw フィールドなし)，長さの下限を 0 文字 (u\_minlen#0) に設定し，上限を 10 文字 (u\_maxlen#10) に設定します。下限の値を 0 にすると，Tru64 UNIX は『*Green Book*』のアルゴリズムを使用して，パスワードを生成します。
- 空のパスワードを使えないようにします (u\_nullpw@)。
- パスワードの有効期間に 180 日を設定します (u\_exp#15724800)。
- アカウントの有効期間に 360 日を設定します (u\_life#31449600)。
- パスワード・ヒストリ・ファイルの深さに 9 を設定します (u\_pwdepth#9)。
- アカウントがロックされるまでにパスワード入力の再試行回数に 5 を設定します (u\_maxtries#5)。
- 新規アカウントがロックされるように設定します (u\_lock)。
- 端末がロックされるまでにログインを試すことができる回数の上限に 10 を設定します (t\_maxtries#10)。
- ログインの試行間の遅延に 2 秒を設定します (t\_logdelay#2)。
- 平凡性チェック (u\_restrict) およびサイトのパスワード制限 (u\_policy) を選択します。

アカウント・マネージャ (dxaccounts) または edauth プログラムを使って，省略時の設定を変更します。

### E.3.3 ライブラリ

システム上のライブラリが，攻撃に利用されることがあります。次の方法で，ライブラリを保護します。

- secconfig から入力を求められたときに yes と答え，セグメント共有を無効にします。
- アクセス許可が正しいこと (所有者以外の書き込み許可なし) と，リンクされたターゲット・ファイルも含め，シェアード・ライブラリ (/usr/shlib/\*.so) の所有者が root であることを確認します。この手順には，ls -lL を使います。

### E.3.4 アカウント・プロトタイプとアカウント・テンプレート

ユーザ・アカウントのスタートアップ・ファイルを作成するためのアカウント・テンプレートは、`/usr/skel/.login`、`/usr/skel/.cshrc`、および `/usr/skel/.profile` です。

アカウント・プロトタイプ(「ローカル・テンプレート」と呼ばれる)は、アカウント・マネージャ (dxaccounts) が提供します。プロトタイプでは、個々のユーザ・アカウントのパスワード有効期間およびログイン試行回数などの属性を設定できます。属性値がローカル・テンプレートの中に指定されていない場合は、`default` ファイルの値が使用されます。システム単位の省略時の属性値は、`/etc/auth/system/default` ファイルに格納されています。システムの省略時設定値は、`/usr/tcb/bin/edauth` コマンドを使って設定されます。

ユーザ・アカウントは、次の手順で構成します。

- 提供された省略時のテンプレートを使って、サイトのセキュリティ・ポリシーを反映したアカウント・テンプレートを作成します。
- `umask` を `/usr/skel/.login` ファイルに設定します (HP の推奨値は、`027` です)。
- 必要に応じて、ユーザに制限付きシェル (Rsh) を指定します。
- 各ユーザに、システム上の有効なエントリ・パス (ログイン・シェル) があることを確認します。ユーザの `/home/.profile` から、または `/etc/passwd` ファイル内のエントリからアプリケーションを起動するか、スタートアップ・プログラムを実行してユーザの開始点としてアプリケーションを起動することによって、ユーザを直接アプリケーションの中に入らせることができます。
- ユーザの `.profile` ファイルから呼び出されたメニュー・スクリプトによってユーザのアクセスを制限する場合、`Ctrl/C` およびその他のキーボード割り込みを無視するために、スクリプト・ファイルの先頭に `trap` コマンドを置く必要があります。

### E.3.5 監査サブシステムの構成

監査サブシステムのカーネル・オプションを構成するには、カーネルに監査サブシステムを取り込んで構築しておく必要があります。監査サブシステム

を構成するには、カーネルの構築後に `sysman auditconfig` ユーティリティを使います。監査は、次のように構成し、実行してください。

- 監査ログ (`/var/audit/auditlog.nnn`) には、省略時の場所を使います。オーバフローから保護するために、監査ログはルート (`/`) および `/usr` 以外のファイル・システムに置きます。
- `/etc/sec/auditd_loc` ファイルを編集して、監査ログ・データのオーバフローに備えて、監査ログの代替格納場所を指定します。
- `auditd` のメッセージをコンソール (`/dev/console`) に送ります。
- サイト・ポリシーの規定に従って、`trusted_events` を監査し、無効なアカウントでログインを試みたユーザの名前をログにとるように監査マスクを設定します。

監査デーモンをコマンド行から開始するには、次のコマンドを使います。

```
# /sbin/init.d/audit start
```

監査サブシステムについては、第 3 章 を参照してください。

### E.3.6 システムの保全性の確認

システムをリブートしてエンハンスド・セキュリティ・オプションを有効にしたら、`fverify` および `authck` プログラムを実行して、システムの保全性を確認します。

### E.3.7 ネットワーク・セキュリティの構成

コンピュータ環境を保護するためには、ネットワークを正しく構成することが大切です。ネットワーク構成の手助けとして、次のチェック・リストを利用してください。

- NIS (以前はイエロー・ページと呼ばれていた、Network Information Services) を使って root アカウント情報を配布しないでください。詳細については A.6 節 を参照してください。
- NIS を使う場合は、`ypserv(8)` で説明しているように、`/etc/yp/securenets` ファイルを使います。
- `ypbind` は、`-ypset` オプションや `-ypsetme` オプションを付けずに、`-s` フラグを付けて実行します (省略時の指定)。
- システムに `uucp` が構成されている場合は、次のようにします。

- uucp アカウントがパスワード制御されており、アクセスが必要なマシンのそれぞれに個別の uucp アカウントがあることを確認します。
- /var/spool/uucp/Permission ファイルに有効なエントリだけがあることを確認します。
- /var/spool/uucp/Systems ファイルに有効なエントリだけがあることを確認します。
- ファイル転送プロトコル (FTP) が保護されていることを確認し、可能な場合は、匿名 FTP アカウントを用意しないようにします。匿名 FTP を使う必要がある場合は、次のことを確認します。
  - FTP アカウントの保護パスワード・フィールドにアスタリスク (\*) があります。
  - FTP 用のホーム・ディレクトリ /usr/ftp が作成されています。/usr/ftp ディレクトリの下にサブディレクトリ /bin および /etc を作成します。
  - ホーム・ディレクトリ内には、ftp が所有しているものはありません。
  - 転送ファイルを置いたり、取り出したりするための公開サブディレクトリが、/usr/ftp ディレクトリ下に作成されています。ユーザ ftp は、この公開サブディレクトリに対して書き込みアクセス権だけを持つ。
  - パスワードなしの ftp アカウントだけからなる ~ftp/etc/passwd ファイルを作成します。
  - /etc/sia/bsd\_matrix.conf ファイルを ~ftp/etc/sia/matrix.conf にコピーします。
  - /sbin/ls を ~ftp/bin/ls にコピーします。
  - ftp アカウントのログイン・シェルは、/sbin/sink です。
- ワークステーションで DES クッキー・ベースの認証 (省略時の設定) を使用していることを確認します。詳細については、dtlogin(1) の XDM-AUTHORIZATION-1 の項を参照してください。
- /usr/bin/X11/xhost を実行したとき、何も報告されてはなりません。出力は次のようにならなければなりません。

```
# xhost
access control enabled, only authorized clients can connect
#
```

## E.3.8 インストール後のセキュリティの構成

システムをインストールして構成した後は、以降の項の作業を行います。

### E.3.8.1 リモート・アクセスの umask

サイトのセキュリティ・ポリシーに記述されているとおりに、ファイル `/etc/csh.login`、`/etc/profile`、および `/etc/init.d/inet` に `umask` エントリを追加します (`/etc/init.d/inet` ファイルは、アップデート・インストールの際に上書きされるので注意してください)。

### E.3.8.2 デバイス

`/usr/tcb/bin/dxdevices` を使って、サイトのセキュリティ・ポリシーを反映したセキュリティ属性を持つデバイスを作成します。

リモート・ログインのシェル・ファイルを次のように変更して、所有者だけが端末ポートを読み取れるようにします。

`/etc/profile` ファイルには、次の内容を追加します。

```
case "$TERM" in
none) ;;
*) /usr/bin/setacl -b '/usr/bin/tty' ;;
esac
```

`/etc/csh.login` ファイルには、次の内容を追加します。

```
if ($?TERM) then
  if ("$TERM" != "none") then
    /usr/bin/setacl -b '/usr/bin/tty'
  endif
endif
```

### E.3.8.3 アカウント

ユーザ・アカウントを次のように作成し、チェックします。

- アカウント・マネージャ (`/usr/bin/X11/dxaccounts`) を使用するか、以前のシステムから `/usr/users` の部分とその関連ファイルをリストアして、システムのユーザ・アカウントを作成します。

- ホーム・ディレクトリがオプション `noexec` , `nosuid` , および `nodev` を付けてマウントされていることを確認します。
- 次のようなコマンドを使って、CDE ユーザに対して自動一時停止機能が有効になっていることを確認します。

```
# grep extension.lockTimeout \
    ~/.dt/sessions/current/dt.resources
```

ステータスが 0 なら、自動一時停止機能は無効になっています。

- `/etc/passwd` および `/var/tcb/files/auth.db` データベースを参照して、ユーザのホーム・ディレクトリとパスワードが適切であることを確認します。

ユーザ・アカウントの作成については、『システム管理ガイド』を参照してください。

#### E.3.8.4 root アクセス

root アクセスは慎重に制御し、監視する必要があるため、次の条件が満たされていることを確認します。

- システムのインストール後、またはサポート・ベンダがマシンにアクセスした後は、すべてのパスワードを変更します。
- パスワード生成方法が漏れるのを防ぐために、ベンダにアクセスを許す前に root パスワードを変更します。
- シングルユーザ・パスワード機能が有効になっています。 `sulogin(8)` を参照してください。
- `su` コマンドを使って root になると、監査でログにとられます。
- システム上の `setuid 0` または `setgid 0` を実行するプログラムへのアクセスが制限されている (700 , 710 , または 711)。
- root または所有者だけが、`/var/spool/cron/crontabs` ファイルにアクセス可能です。
- root アクセスが特定のログイン・デバイスに制限されているか、ユーザが root アカウントにアクセスするには `su` コマンドを使わなければなりません。詳細については、`securettys(4)` のリファレンス・ページを参照してください。
- システム提供の UID のログインが、必要に応じて制限されている (`u_lock` フィールドの設定)。推奨される制限は、次の表のとおり。



UID	推奨するログイン状態
root	制限あり
daemon	不可
bin	不可
sys	不可
uucp	制限あり
nobody	不可
adm	制限あり
lp	不可

### E.3.9 ネットワークの構成

/etc/svc.conf ファイルを参照して、NIS 用に適切な構成が設定されていることを確認します。また、NIS を使う場合は、クライアント・マシンおよびサーバのファイル /etc/rc.config または /etc/rc.site の NIS\_DOMAIN 変数に、正しいドメイン名が定義されていることも確認します。

次の表のネットワーク・ファイルが保護されていることを確認します。

ファイル	コメント
/etc/exports	エントリの有効性を確認します。できる限り、 <code>-root=</code> オプションを使うのは避けま す。指定しているすべてのファイル・シス テムに対して、 <code>-access=&lt;hostname&gt;</code> およ び <code>-ro</code> オプションを使います。
/etc/hosts	
/etc/services	
/etc/protocols	
/etc/inetd.conf	
/etc/hosts.equiv	エントリがローカル・ホストであるこ とを確認します。
/etc/ethers	
~username/.rhosts および ~username/.shosts	これらのファイルを削除するか、 <code>-l</code> フラグを設 定して <code>rlogind</code> と <code>rshd</code> を実行します。

## E.4 物理的セキュリティ

サイトのセキュリティで重要なのが、環境内のすべての構成要素に関する物理的セキュリティです。以下のようにして物理的セキュリティをチェックします。

- システムとそのケーブルが、安全な環境にあることを確認します。
- すべてのネットワーク構成要素が物理的に保護されていることを確認します。この構成要素には、ファイル・サーバ、ブリッジ、ルータ、ハブ/コンセントレータ、ゲートウェイ、ターミナル・サーバ、およびモデムがあります。
- コンソールのプロンプトで次のコマンドを使って、サイトのポリシーに従ってブート・フラグが設定されていることを確認します。

```
>>> show
```

- システムがコンソール・パスワード機能をサポートしている場合は、この機能が使用されていることを確認します。コンソール・パスワードがサポートされているかどうかについては、ハードウェアのドキュメントで確認してください。
- コンソール・ターミナルのファンクション・キーに、ログインやパスワードの情報が設定されていないことを確認します。
- モデムに自動切断機能があることを確認します。また、モデムが安全な環境にあることも確認します。

## E.5 アプリケーション

システムで実行しているアプリケーション・ソフトウェアのセキュリティを確保するには、以下の条件を満たしている必要があります。

- `setuid` プログラムや `setgid` プログラムを制限しています。
- `root` だけが、制御ファイルおよび実行可能ファイルに書き込み可能です。
- ファイアウォール製品がインストールされている場合は、ファイアウォールのドキュメントを参照して適切に構成します。
- `screend` プログラムを実行している場合は、`screend(8)` で説明されているように構成します。

- トンネリング・ソフトウェアをインストールした場合は、そのドキュメントの記述に従って保護されています。

## E.6 定期的に行うセキュリティ管理手順

クラスごとの見直し作業の頻度は、サイトのセキュリティ・ポリシーによって決まります。定期的に、以下の作業を行います。

- システムとそのアプリケーションをバックアップします。
- 監査ログをチェックします。
- システム・アカウントのログをチェックします。
- `fverify` および `authck` プログラムを実行し、システムの保全性を確認します。`cops` および `tripwire` など、パブリック・ドメイン・プログラムにも、システムの保全性を確認するのに便利なものがあります。
- 必要で、承認されたプログラムだけがシステムにあることを確認します。
- コンパイラを利用できるのが、開発用のシステムだけであることを確認します。
- 次のコマンドを使って、システムに存在する `setuid` および `setgid` プログラムは、承認された、`root` 所有のものだけであることを確認します。

```
# find / \( -perm -4000 -o -perm -2000 \) -ls
```
- `/etc/exports` ファイルをチェックし、すべてのエントリが有効であることを確認します。
- ユーザ・アカウントを次のようにチェックします。
  - `/etc/passwd` ファイルをチェックし、すべてのアカウントがまだ有効であることを確認します。
  - 次のコマンドを実行して、対応する `/etc/passwd` のエントリのないエンハンスド・プロファイル (`prpasswd` エントリ) がないことを確認します。

```
# /usr/tcb/bin/convuser -dN
```
  - ホーム・ディレクトリのアクセス許可が、サイトのポリシーに従って設定されていることを確認します。
  - ユーザのホーム・ディレクトリ内のすべてのファイルを、そのユーザが所有していることを確認します。

- 各ユーザに、システム上の有効なエントリ・パス (ログイン・シェル) があることを確認します。
- ファイル `.rhosts` または `.netrc` 内のエントリが適切であることを確認します。`.exrc` ファイルおよび `.netrc` ファイルが、ユーザのホーム・ディレクトリだけにあることを確認します。
- `hosts.equiv` ファイルをチェックして、エントリが有効であることを確認します。
- 次のファイルのエントリが、システム・パラメータと競合せず、ファイルが 755 のアクセス許可で保護されていることを確認します。

```
.profile
.login
.cshrc
.kshrc
.logout
```

- 次のコマンドを使って、ユーザ・マスクがサイトのポリシーに合っていることを確認します。

```
# grep umask /usr/users/*/*.*
```

システムの省略時のマスクは 022 です。

- `/dev` ディレクトリを参照し、以下のことを確認します。
  - 特殊デバイスのアクセス許可が適切です。
  - `mem`, `kmem`, および `swap` などのデバイスへのアクセスが、適切に保護されている (440)。
  - 所有者だけが端末ポートを読み取り可能です。
  - ユーザが、ユーザの端末デバイスおよびプリンタ以外のデバイスを所有していません。
- モデム認証が、意図したとおりに機能していることを確認します。
- 次のコマンドを使って、異なる UID に対して同じユーザ名を使っていないことを、ローカルの `/etc/passwd` ファイルと NIS とを併せて、確認します。

```
# ( 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
```

- 次のコマンドを使って、同じ UID を持つユーザ名がないことを、ローカルの `/etc/passwd` ファイルと NIS とを合わせて、確認します

```
# ( 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
```

- 次の 2 つのコマンドを使って、すべてのアカウントにローカル・パスワードまたは NIS パスワードがあることを確認します。

```
# sort -t:-n /etc/passwd | awk -F:'$2 == "" print'
# /usr/tcb/bin/edauth -g | sed -f sed_file | egrep -v \
  ':u_pwd=[^:]|:u_istemplate:'
```

`sed_file` 内のコマンドは次のとおりです。

```
: top
/:\\$/ {
N
b top
}
s/:\\$/:/g
s/:[<tab><space>]*:/:/g
s/:[<tab><space>]*:/:/g
```

- 次のコマンドを使って、システム上のすべての隠しファイルを確認します。

```
# find / \( -name '*' !-name . ! !-name .. \) -print
```

- 次のコマンドを使って、`/dev` ディレクトリの外にはデバイス・ファイルが存在しないことを確認します。

```
# find / \( -type b -o -type c \) -print
```

- 次のスタートアップ・スクリプト内のエントリが適切で、ファイルが適切に保護されていることを確認します。

```
/sbin/inittab
/etc/init.d
/sbin/rc?.d      ? は実行レベル
```

- システム・クラッシュ発生時に `/var/adm/crash` に保存されたデータにアクセスできるユーザが、`root` および `adm` だけであることを確認します。
- パブリック・ドメインの `crack` プログラムのようなパスワード割り出しプログラムを使って、ユーザのパスワードを特定できないことを確認します。

- サイトの物理的セキュリティ・ポリシを確認します。
- 次のディレクトリのアクセス許可と所有者が、リストされているとおりに設定されていることを確認します。

ディレクトリ	アクセス許可	所有者	グループ
/	755	root	system
/bin	755	root	system
/dev	640	root または 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
/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

ディレクトリ	アクセス許可	所有者	グループ
/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 参照ドキュメントとチェック・ツール

次のドキュメントは、セキュアなコンピューティング環境を構築し、保守するのに役立ちます。

- 『*Site Security Handbook (RFC 1244)*』。このハンドブックは、インターネット特別技術調査委員会 (IETF) のセキュリティ分野とユーザ・サービス分野が共同で作業を行った、サイト・セキュリティ・ポリシ・ハンドブック・ワーキング・グループが作成したドキュメントです。このドキュメントのオンライン・コピーは、<http://www.net.ohio-state.edu/hypertext/rfc1244/toc.html> で提供されています。
- Tru64 UNIX 『インストレーション・ガイド』
- Tru64 UNIX 『インストレーション・ガイド — 上級ユーザ編』
- 『*Trusted Computer System Evaluation Criteria*』 (U.S. Department of Defense, National Computer Security Center, DoD 5200.28-STD, 1985 年 12 月)。このドキュメントは、そのカバーの色から、『*Orange Book*』として知られています。このドキュメントは、トラステッド・コンピュータ・システムの開発および評価のための、米国政府の決定的なガイドです。『*Orange Book*』のオンライン・コピーは、<http://nsi.org/Library/Compsec/orangebo.txt> で提供されています。
- 『*Password Management Guideline*』 (U.S. Department of Defense, (CSC-STD-002-85), 1985 年 4 月 12 日)。このドキュメントは、そのカバーの色から、『*Green Book*』として知られています。パス

ワード・ベースのユーザ認証メカニズムの設計，インプリメント，および使用についての推奨手順を示すことで，『*Orange Book*』をサポートしています。『*Green Book*』のオンライン・コピーは，<http://www.radium.ncsc.mil/tpep/library/rainbow/CSC-STD-002-85.html> で提供されています。

次のドキュメントは，セキュリティの概念および手順について理解するのに役立ちます。

- 『*Computer Security Basics*』 (O'Reilly and Associates, Inc.)。
- 『*Practical UNIX Security*』 (O'Reilly and Associates, Inc.)。
- 『*UNIX: Its Use, Control, and Audit*』 (249 Maitland Avenue, Altamonte Springs, Florida 32701-4201 の Institute of Internal Auditors Research Foundation にご相談ください)。

次のツールは，セキュア環境の維持に役立ちます。

- `crack` は，パブリック・ドメインのパスワード・チェック・プログラムです。
- `tripwire` は，UNIX システム用の保全性モニタです。`tripwire` ソフトウェアは，いくつかのチェックサム・ルーチンや署名ルーチンを使って，ファイルの変更を検出し，システムが維持する情報のうち選択されている項目を監視します。このプログラムは，ファイルおよびディレクトリのアクセス許可，リンク，サイズの変更も監視します。
- `COPS` は，Purdue University の Computer Oracle and Password System パッケージです。`COPS` は，既知の弱点がシステムにないかチェックし，システム管理者に警告します。場合によっては，`COPS` が自動的に問題を解決してくれます。
- `SATAN` は，ネットワークに関連するセキュリティ上のよくある問題をシステム管理者が発見するのに役立つツールです。`SATAN` は，問題を実際に試すことなく，問題点について報告します。見つかった各種の問題について，`SATAN` は，問題の説明，どのような影響があるか，および問題に対する解決策を示すチュートリアルを提供します。

次のスクリプトは，ユーザが作成できるツールの例です。このツールは，ログインおよびログアウトの情報を監査ログから抽出します。

```
#!/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}"
```

以下は上に示したロギング・スクリプトのための crontab エントリです。

```
0 9 * * * /var/adm/local/lreport
```

---

## 用語集

### アクセス ACL (Access ACL)

ACL の正式名で、オブジェクトへのアクセスの可否を判断するために参照されます。

### ACL (access control list : アクセス制御リスト)

従来の UNIX アクセス許可の拡張 (オプション) で、ユーザおよびグループごとに、読み取り、書き込み、および実行の許可を指定できます。

アクセス ACL (*Access ACL*)、省略時 ACL (*Default ACL*) も参照

### 管理者 (administrator)

本書では、システムのセキュリティに携わっているユーザに対して「管理者」という用語を使用します。

### 監査イベント (audit event)

監査サブシステムで監視して報告するイベント。イベントには、システム・イベント、アプリケーション・イベント、およびサイト定義イベントがあります。イベントは、システム上で実行するコマンド、システム・コール、ルーチン、またはプログラムで発生します。

### 監査 ID (AUID)

ログイン時に作成され、すべてのプロセスに引き継がれる ID。

### 監査 (auditing)

保護システム上でセキュリティに関係する動作を記録、調査、および確認することです。

### 認証 (authenticate)

システムの ID を証明するための証明書をエンティティがシステムに発行するプロセス。

### 認証方式 (authentication method)

エンティティが認証されるメカニズム。BSD、エンハンスド・セキュリティ、および Kerberos などが含まれます。

### **BASE (BSD) セキュリティ**

BSD UNIX システムで実行される従来のセキュリティ。BASE セキュリティは、ファイル `/etc/passwd` に存在するユーザ名およびパスワードに基づくユーザ認証からなります。BASE セキュリティは省略時の Tru64 UNIX セキュリティです。

### **BSD (Berkeley Software Distribution)**

カリフォルニア大学バークレー校のコンピュータ・システム・リサーチ・グループの UNIX ソフトウェア・リリース。Tru64 UNIX の機能の一部は BSD をベースとしています。

### **復号 (decryption)**

暗号化アルゴリズムを使って、暗号化されたテキストを平文 (復号されたテキスト) に変換する処理。

### **省略時 ACL (Default ACL)**

ディレクトリに関連付けられた ACL。2 種類の ACL (省略時アクセス ACL と省略時ディレクトリ ACL) で、ディレクトリ内に作成されるファイルおよびサブディレクトリに与えられる ACL が決まります。

### **デジタル証明書 (digital certificate)**

個人、サーバ、企業、またはその他のプリンシパルの ID の認証、およびそのプリンシパルと公開鍵を結び付けるために使われる電子的なドキュメント。

### **デジタル署名 (digital signature)**

メッセージの送り主またはドキュメントの署名者の ID を認証するために使われる電子的な署名。

### **任意アクセス制御 (DAC: discretionary access control)**

ディレクトリ、ファイル、デバイス、およびプロセスを含む、システム・リソースへのアクセスを管理します。Tru64 UNIX は、Tru64 UNIX アクセス権および ACL (Access Control List) 使用して、DAC を実現します。

### **暗号化 (encryption)**

暗号化アルゴリズムを使って、平文 (暗号化されていないテキスト) を暗号化されたテキストに変換する処理。

### **実効ユーザ ID (EUID: effective user ID)**

現在のユーザ ID ですが、ユーザ自身の ID とは限りません。たとえば、ログイン ID でログインしたユーザは、別の ID に変わる可能性があります。

ユーザが変更した ID は、ユーザが元のログイン ID へ切り替えるまで実効ユーザ ID となります。

#### エンハンスト・パスワード (**enhanced passwords**)

エンハンスト・セキュリティ・オプションで提供される拡張属性を持つパスワード。エンハンスト・パスワードは、ファイル `prpasswd` に格納され、エンハンスト・パスワード、保護パスワード、またはシャドウ・パスワードとも呼ばれます。

#### エンハンスト・セキュリティ (**Enhanced Security**)

BASE セキュリティで不足している点を補うオプションのセキュリティ機能。エンハンスト・セキュリティは、エンハンスト・パスワード・プロファイルから構成されています。

エンハンスト・パスワード (*enhanced passwords*) も参照

#### エンティティ (**entity**)

認証の対象となるユーザ、プログラム、システム。

#### ER (**external representation** : 外部表現)

ACL の POSIX 適合 ASCII 表現であり、ユーザへの表示に使用します。

IR (*internal representation* : 内部表現) も参照

#### 評価基準 (**evaluation criteria**)

トラステッド・コンピュータ・システム評価基準 (*TCSEC: Trusted Computer System Evaluation Criteria*)。Tru64 UNIX システムのエンハンスト・セキュリティ機能は、この基準に合うように設計されています。

#### ホスト・ベース認証 (**host-based authentication**)

ホスト名およびユーザ名での識別 (パスワードではなく) に基づく非対話式の UNIX 認証方式。

#### IPsec

TCP/IP 転送プロトコルを使って交換されるデータに対して、IP レイヤでリモート認証サービスを提供するプロトコル。

#### IR (**internal representation** : 内部表現)

ACL ライブラリ・ルーチンで使う ACL のバイナリ表現。

ER (*external representation* : 外部表現) も参照

## **Kerberos**

コンピュータ・ネットワークのサービスに対する要求を認証するセキュア・メソッド。

## **KDC (Key Distribution Center)**

プリンシパル間のやり取りを仲介し，秘密鍵認証の使用時に，専用のセッション・キーを作成します。

*ER (external representation* : 外部表現) も参照

## **LDAP (lightweight directory access protocol)**

TCP/IP で稼動するインターネット標準の分散型のクライアント/サーバ・ディレクトリ・サービス・プロトコル。

## **ログイン・スプーフ・プログラム (login spoofing program)**

パスワードを盗むために，login プログラムのように見せかけたプログラム。たとえば，スプーフ・プログラムでは，使われていない端末にログイン・プロンプトを出力し，ユーザからの入力を待ちます。

## **NIS (Network Information Service)**

ローカル・エリア・ネットワーク (LAN) の情報を共用するための分散型クライアント/サーバ・データ検索サービス。

## **オペレータ (operator)**

システムの日常的保守を担当している人。これには，バックアップ，ライン・プリンタの保守，およびその他の保守タスクなどがあります。

システム管理者 (*system administrator*) も参照

## **プリンシパル (principal)**

ユーザ，サービス，アプリケーション，またはホスト。

## **プロセス ID (PID:)**

実行中のプロセスに割り当てられた重複しない番号。

## **プロセス (process)**

オペレーティング・システムが制御する単位。プロセスは必ず1つのプログラムを実行しています。このプログラムは，現在のプログラムがシステム・コール `exec()` を呼び出したときに置き換わります。現在のプログラムが保護されている (trusted) とき，そのプロセスは保護されていると見なされます。プログラム (*program*) も参照

### プログラム (program)

プロセスによって実行されるように設計，コンパイル，インストールされた実行可能ファイル内のアルゴリズム一式。プログラムは，システムのセキュリティ・ポリシを満たしているときに保護されていると見なされます。

プロセス (*process*) も参照

### 公開鍵認証 (public key authentication)

接続中の 2 つのプリンシパル (通常，クライアントとサーバ) が，公開鍵や秘密鍵を使って互いとユーザを認証し，そこで交換されるデータを暗号化または復号する暗号化方法。

### PPID (parent process ID : 親プロセス ID)

親プロセス，つまり生成元プロセスの ID。

### root

スーパーユーザ (システム管理者) のログイン名。

### ルート・ディレクトリ (root directory)

UNIX システムのツリー状のファイル構造の最上位ディレクトリの名前。つまり，絶対パス名の先頭です。パス名の中のルート・ディレクトリは，先頭のスラッシュ (/) で表されます。ルート・ディレクトリ自身は単独のスラッシュで表します。

### ルート・ファイル・システム (root file system)

基本的なファイル・システムであり，その他のファイル・システムはすべてこの上にマウントされます。ルート・ファイル・システム内には，オペレーティング・システムのファイルがあり，これらのファイルで残りのシステムを実行します。

### 秘密鍵認証 (secret key authentication)

接続中の 2 つのプリンシパル (通常，クライアントとサーバ) が，セッション鍵と呼ばれる 1 つの秘密鍵を使ってシステムとユーザを認証し，そこで交換されるデータを暗号化または復号する暗号化方法。

### セキュア・シェル (Secure Shell)

コマンドの使用時に交換されるデータに対して，リモート認証サービスを提供するネットワーク・コマンド一式を備えたクライアント/サーバ・アプリケーション。

#### セキュリティ属性 (security attributes)

セキュリティを実現するためのトラステッド・コンピューティング・ベース (TCB) によって使用されるパラメータ。セキュリティ属性には、さまざまなユーザおよびグループ ID が含まれます。

#### SIA (Security Integration Architecture : セキュリティ統合アーキテクチャ)

セキュリティ統合アーキテクチャは、セキュリティ・メカニズムが使われる順序を管理します。

保証機能 (*vouching*) も参照

#### SSO (Single Sign On)

Kerberos を使って、ftp, rcp, rlogin, rsh, および telnet ネットワーク・コマンドの使用時に交換されるデータに対し、リモート認証サービスを提供するクライアント/サーバ・ソフトウェア。

#### サイト定義イベント (site-defined event)

アプリケーション・ソフトウェア (つまり、オペレーティング・システムではない) で作成される監査イベント。

#### スプーフ・プログラム (spoofing program)

ログイン・スプーフ・プログラム (*login spoofing program*) を参照

#### システム管理者 (system administrator)

システム管理者は、ファイル・システムの保守と修復、アカウントの作成、およびその他の管理作業を担当します。

#### TCB (trusted computing base : トラステッド・コンピューティング・ベース)

ハードウェア、ソフトウェア、およびファームウェア一式で、これらが一体となってシステムのセキュリティ・ポリシーを実現します。Tru64 UNIX の TCB には、システムのハードウェアおよびファームウェア、トラステッド Tru64 UNIX オペレーティング・システムと、セキュリティ・ポリシーを実現するトラステッド・コマンドおよびユーティリティが含まれます。トラステッド Tru64 UNIX システムと併せて提供されるオペレーティング・システムおよびその他のソフトウェアは、セキュリティの要件を満たしています。

#### 従来のセキュリティ (Traditional security)

*BASE (BSD)* セキュリティ を参照



### 平凡性チェック (triviality check)

簡単に推測できるパスワードを使わないようにするために、パスワードに対して行うチェック。平凡性チェックにより、辞書にある単語、ユーザ名、およびユーザ名を少し変えた単語をパスワードとして使うのを防ぐことができます。

### トロイの木馬 (Trojan horse)

ユーザによって起動されたときに、ユーザのデータを盗むか、ユーザのファイルを破壊するか、またはトロイの木馬を仕掛けた人が再度ユーザのアカウントにアクセスできるようにする仕組みを作成するプログラム。ウィルスとワームは、トロイの木馬の一種です。

ウィルス (*virus*)、ワーム (*worm*) も参照

### ウィルス (virus)

システム上の他のプログラムまたはファイルに潜入し、使用できる手段 (ディスク・ファイル、ネットワークなど) で別の同じようなコンピュータに自己複製を行うように設計されたコンピュータ・プログラム。ウィルスは、「感染」対象のプログラムまたはシステムに損害を与えたり破壊したりするように設計され、多くの場合、ウィルスのプログラムの誕生日のような、特定の日に発症するようにプログラムされています。

トロイの木馬 (*Trojan horse*)、ワーム (*worm*) も参照

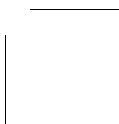
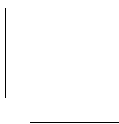
### 保証機能 (vouching)

セキュリティ・メカニズムが、先に実行されたセキュリティ・メカニズムの認証処理を信頼できるようにする技法。この機能は、セキュリティ統合アーキテクチャ (SIA) で実現されます。

### ワーム (worm)

システム上の他のプログラムまたはファイルに潜入し、使用できる手段 (ディスク・ファイル、ネットワークなど) で別の同じようなコンピュータに自己複製を行うように設計されたコンピュータ・プログラム。ワームは、損害を与えるようには設計されていませんが、本来の処理のために用意されているリソースを占有するので有害です。

トロイの木馬 (*Trojan horse*)、ウィルス (*virus*) も参照



## 数字および記号

- 2 進数
  - アクセス許可 ..... 2-9t
- 8 進数
  - アクセス許可の設定..... 2-8

## A

## ACL

- NFS 上での ..... 2-16
  - アーカイブ..... 2-28
  - 概要 ..... 2-14
  - 管理 ..... 2-25
  - カーネルのステータス ..... 2-15
  - 継承 ..... 2-21
  - 構造 ..... 2-17
  - コマンドとの相互作用 ..... 2-27
  - ステータスの確認..... 2-15
  - 表示 ..... 2-25
  - ファイルの ACL の表示 ..... 2-26
  - 変更 ..... 2-25
  - 無効化 ..... 2-15
  - 有効化 ..... 2-15
- ACL エントリ**
- 削除 ..... 2-26
  - 追加 ..... 2-26
  - 変更 ..... 2-26

**audgen8** トラストッド・イベン

- ト ..... 3-11
- audgen** コマンド ..... 3-6
- audit\_daemon\_exit** トラストッド・イベント ..... 3-9
- audit\_log\_change** トラストッド・イベント ..... 3-9
- audit\_log\_create** トラストッド・イベント ..... 3-9
- audit\_log\_overwrite** トラストッド・イベント ..... 3-10
- audit\_reboot** トラストッド・イベント ..... 3-10
- audit\_start** トラストッド・イベント ..... 3-10
- audit\_stop** トラストッド・イベント ..... 3-10
- audit\_subsystem**
  - イベント・エイリアス ..... 3-12
- audit\_suspend** トラストッド・イベント ..... 3-11
- audit\_tool.ultrix** コマンド ..... 3-6
- audit\_tool** コマンド ..... 3-6
- audit\_xmit\_fail** トラストッド・イベント ..... 3-11
- auditconfig** コマンド..... 3-5
- auditconfig** トラストッド・イベント ..... 3-10
- auditd** コマンド ..... 3-6

**auditmask** コマンド..... 3-6  
**AUID (監査 ID)** ..... 3-13  
**auth\_event** トラストッド・イベン  
ト..... 3-11  
**authck** プログラム ..... A-15

## B

**Berkeley** データベース ..... A-12

## C

**C2** 機能  
パスワード制御 ..... 1-13  
**chmod** コマンド ..... 2-4  
**creacct** コマンド  
構文 ..... C-29  
使用 ..... C-29

## D

**/dev/console** ファイル ..... A-38  
**/dev/pts/\*** ファイル ..... A-38  
**/dev/tty\*** ファイル ..... A-38

## E

**/etc/auth/system/default** ファイ  
ル..... A-37  
**/etc/auth/system/devassign** ファイ  
ル..... A-38  
**/etc/auth/system/ttys.db** ファイ  
ル..... A-37  
**/etc/group** ファイル ..... A-38  
**/etc/passwd** ファイル..... A-38

**/etc/sec/auditd\_clients** ファイ  
ル..... 3-22  
**/etc/sec/event\_aliases** ファイ  
ル..... 3-12  
**/etc/sec/site\_events** ファイル . 3-11

## F

**fverfy** コマンド ..... A-39

## K

**.k5login** ファイル  
エントリのフォーマット .... C-15  
例..... C-15  
**Kerberos**  
クライアント ..... C-1  
サーバ ..... C-1  
認証 ..... C-2  
レルム ..... C-2  
**krb.conf** ファイル  
エントリの順序 ..... C-11  
例..... C-11  
**krb.realms** ファイル  
エントリの順序 ..... C-13  
例..... C-13  
ワイルドカード ..... C-12

## L

**LDAP Module for System  
Authentication**  
アクセス制御 ..... D-7  
**ldapcd.conf** ファイル  
パラメータ..... C-16  
例..... C-16

**ldapusers.deny** ファイル  
    れい ..... C-18  
**login**  
    トラステッド・イベント ..... 3-11  
**logout** トラステッド・イベント 3-11  
**LUID (ログイン ID)** ..... 3-13

## N

**NIS**  
    置き換え ..... A-18  
    自動処理 ..... A-21  
    大規模データベース ..... A-21

## P

**passwd** ファイル ..... A-38  
**pts/\*** ファイル ..... A-38

## R

**rc[023]** ファイル ..... A-38  
**root** 認証プロファイル ..... A-36

## S

**/sbin/creacct** コマンド  
    ( **creacct** コマンド を参照 )  
**secconfig** コマンド ..... A-2  
**SIA**  
    SIA メカニズムの削除 ..... 1-25  
    SIA メカニズムの初期化 ..... 1-23  
    SIA メカニズムの追加 ..... 1-24  
    概要 ..... 1-19  
    構成 ..... 1-21

パスワードの変更 ..... 1-20  
ログイン ..... 1-25

### Single Sign On

Active Directory に関する考慮事  
    項 ..... C-3  
ASU に関する考慮事項 ..... C-3  
DCE に関する考慮事項 ..... C-3  
/etc/ldapcd.conf ファイル .... C-16  
/etc/ldapusers.deny ファイル C-18  
Kerberos ..... C-1  
/krb5/.k5login ファイル ..... C-15  
/krb5/krb.conf ファイル ..... C-10  
/krb5/krb.realms ファイル .. C-12  
/krb5/v5srvtab ファイル ..... C-14  
SIA ..... C-9  
Tru64 UNIX でのインストール C-6  
Windows 2000 でのインストー  
    ル ..... C-4  
アップグレード ..... C-3  
インストール ..... C-3  
インストールに関する考慮事項 C-3  
管理 ..... C-27  
グループの作成 ..... C-25  
構成ファイル ..... C-10  
コンピュータ・アカウントの作  
    成 ..... C-24  
サービス鍵テーブルの管理 .. C-29  
信任状キャッシュの削除 .... C-29  
チケットの表示 ..... C-28  
チケットの要求 ..... C-28  
パスワードの設定 ..... C-24  
ユーザ・アカウントの作成 .. C-19  
**sysman auditconfig** コマンド .. 3-5

## T

- /tcb/files/auth/r/root ファイル A-36
- Tru64 UNIX のアクセス許可
  - 概要 ..... 2-2
  - 表示 ..... 2-2
- tty\* ファイル ..... A-38

## U

- umask コマンド ..... 2-10

## X

- X サーバ
  - 再起動 ..... C-9
- X ディスプレイ ..... A-34

## あ

- アカウント
  - 追加 ..... A-2
- アカウント・ツール ..... 3-43
- アクセス許可
  - 2 進数 ..... 2-9t
  - 8 進数を使った設定 ..... 2-8
  - umask による指定 ..... 2-12
  - 組み合わせ ..... 2-9t
  - ファイルとディレクトリの設定 2-4
- アクセスの制限 (ファイル) ..... 2-11
- アップデート・インストレーション ..... A-2
- アプリケーション
  - ファイル制御データベースへの追加 ..... A-16
- アプリケーション固有の監査 ... 3-11

- アプリケーションの監査 ..... 3-11
- 暗号化 ..... 1-10
- 暗号化されたパスワード ..... A-36

## い

- インストール
  - エンハンスド・セキュリティ . A-5

## え

- エンハンスド・セキュリティ ... 1-12
  - NIS ..... A-16
  - NIS 移行 ..... A-23
  - NIS クライアントの設定 ... A-23
  - NIS スレーブ・サーバの設定 A-22
  - NIS の削除 ..... A-23
  - NIS のトラブルシューティング
    - グ ..... A-26
  - NIS マスタ・サーバの設定.. A-20
  - NIS ユーザ・アカウントのテンプレート ..... A-19
- TruCluster でのインストール A-28
- TruCluster での構成 ..... A-28
- TruCluster での認証 ..... A-31
- 暗号化 ..... A-8
- インストール ..... A-1
- 機能 ..... 1-12
- 最短時間の構成 ..... A-6
- 集中アカウント管理 ..... A-16
- 制御構成 ..... A-6
- 正常ログインのレコード ..... A-7
- セキュリティ ..... A-5
- プロファイルの移行 ..... A-8
- 保証機能 ..... A-8

有効期間の設定 .....	A-6
ユーザ・アカウントのテンプレート .....	A-14
ユーザの認証 .....	A-13
ログイン機能 .....	1-13
ログイン失敗のレコード .....	A-7
ログイン操作の時間間隔 .....	A-7
ログインの最大試行回数 .....	A-6
ログインの時間間隔.....	A-7
ログイン・レコードの構成....	A-7
エンハンスド・セキュリティ・データベース .....	A-8
エンハンスド・パスワード・データベース .....	A-9, A-36

## か

回復	
監査データ.....	3-26
監査.....	3-1
( 監査サブシステム も参照 )	
audgen コマンド.....	3-6
audit_tool.ultrix コマンド.....	3-6
audit_tool コマンド .....	3-6
auditconfig コマンド .....	3-5
auditd コマンド.....	3-6
auditmask コマンド .....	3-6
AUID (監査 ID) .....	3-13
CDE インタフェース.....	3-6
/etc/sec/auditd_clients ファイル .....	3-22
GUI.....	3-6
ID (AUID) .....	3-13
LUID (ログイン ID).....	3-13

sysman auditconfig コマンド..	3-5
TruCluster での.....	3-45
アカウント・ツール.....	3-43
アプリケーション固有の監査	3-11
一時停止.....	3-25
イベント	
サイト定義イベント (site-defined event) .....	3-11
事前選択 .....	3-35
状態依存情報.....	3-42
トラステッド・イベント....	3-9
イベントの選択 .....	3-24, 3-26
イベントの無効化.....	3-30
概要 .....	3-1
監査イベント間の依存関係...	3-42
監査ハブ.....	3-22
監査ホスト・ファイル .....	3-22
監査マスク.....	3-7
監査マネージャ .....	3-6
監査レコードの表示.....	3-38
監査レポートに対する対処...	3-49
起動 .....	3-25
クラッシュ回復 .....	3-26
グラフィック・インタフェース	3-6
グラフィック・インタフェースのアクセス.....	3-6
構成 .....	3-18
コマンド.....	3-5
コンソール・メッセージ .....	3-3
サイト定義イベント (site-defined event) .....	3-11
事前選択.....	3-35
実現上の注意 .....	3-7

自動監査コマンド .....	3-9
停止 .....	3-25
データ回復.....	3-26
データ増加の管理.....	3-35
データの管理 .....	3-35
トラステッド・イベント .....	3-9
ネットワーク監査ホスト・ファイ ル .....	3-22
ネットワークを介した監査...	3-22
ファイル.....	3-3
site_events ファイル.....	3-11
メッセージ.....	3-3
ユーザ監査マスク	
設定 .....	3-7
リモートでの監査 .....	3-22
レコード内容 .....	3-13
レコードの内容 .....	3-13
レポート, 簡略型.....	3-40
ロギング・ツール.....	3-43
ロギン監査マスク	
設定 .....	3-7
ログ・ファイル .....	3-3, 3-38
監査 ID (AUID) .....	3-1
監査イベント	
監査イベント・エイリアス...	3-12
監査イベントの管理.....	3-35
サイト定義.....	3-11
トラステッド監査イベント....	3-9
表示 .....	3-27
監査イベント・エイリアス .....	3-12
監査イベント間の依存関係 .....	3-42
監査イベントの事前選択 .....	3-35
監査可能なイベント .....	3-7
監査サブシステム	
構成 .....	3-18

監査マネージャのグラフィック・イン タフェース.....	3-6
監査レコード	
表示 .....	3-38
監査レコードの内容 .....	3-13
監査レポートに対する対処 .....	3-49
簡略型監査レポート .....	3-40

## き

擬似端末 .....	A-38
------------	------

## く

クラスタ	
NIS .....	A-30
アップグレード .....	A-28
分散ログイン .....	A-30
クラスタでの分散ログイン ....	A-30
クラッシュ回復	
監査データ.....	3-26
グラフィック・インタフェース	
監査サブシステム用.....	3-6
グループ	
データベース・ファイル ....	A-38
グループ・データベース .....	A-38

## こ

構成	
SIA .....	1-21
監査 .....	3-18
監査サブシステム.....	3-18
パスワード変更時間.....	A-6
パスワード変更制御.....	A-6
パスワード有効期間.....	A-6



コマンド	
chmod .....	2-4
umask .....	2-10
コンソール・ファイル.....	A-38
コンソール・メッセージ	
監査 .....	3-3

## さ

サイト定義イベント ( <b>site-defined event</b> ).....	3-11
削除	
絶対アクセス許可 .....	2-8
サービス鍵テーブル .....	C-2
チケットの削除 .....	C-30
破棄 .....	C-30
表示 .....	C-29
プリンシパルに対するエントリの抽出 .....	C-30
ホストに対するエントリの抽出 .....	C-30
マージ .....	C-30

## し

シェアード・ライブラリ .....	A-4
システム監査マスク .....	3-7
システム・コンソール.....	A-38
システム省略時設定データベース	
説明 .....	A-9
システム省略時設定データベース	
更新 .....	A-33
システムの起動 .....	A-34
従来のログイン .....	3-44

状態依存監査イベント.....	3-42
省略時設定データベース .....	A-9
省略時のアクセス許可	
設定 .....	2-10
省略時のユーザ・マスク	
( <b>umask</b> ).....	2-11
シングルユーザ・モード .....	A-37

## せ

セキュア・シェル	
scp2 コマンド .....	B-25
sftp2 コマンド .....	B-26
ssh2 コマンド.....	B-26
TCP/IP ポートのフォワーディング .....	B-23
X11 のフォワーディング ....	B-24
概要 .....	B-2
クライアント .....	B-2
公開鍵によるユーザ認証の構成 .....	B-12
公開ホスト鍵と秘密ホスト鍵の作成 .....	B-22
構成 .....	B-3
コマンド.....	1-16t, B-1t, B-25
サーバ .....	B-2
サーバの管理 .....	B-21
使用するコマンドの構成 .....	B-9
デーモンの起動 .....	B-21
デーモンの再起動.....	B-21
デーモンの停止 .....	B-21
デーモンのリセット.....	B-21
パスフレーズの管理.....	B-17

パスワードによるユーザ認証の構成 ..... B-11  
ホスト・ベースのユーザ認証の構成 ..... B-18  
ポートのフォワーディング.. B-23  
ユーザ・アクセスの制限 .... B-22  
ユーザ認証..... B-10  
ユーザ認証の構成..... B-10  
セキュア・シェル・クライアント  
構成 ..... B-7  
セキュア・シェル・サーバ  
構成 ..... B-4  
セキュリティ特性 ..... A-32  
セキュリティ・ポリシ..... E-2  
セグメントの共有 ..... A-4  
絶対アクセス許可  
削除 ..... 2-8  
設定 ..... 2-7  
設定  
8進数を使ったアクセス許可の 2-8  
絶対アクセス許可 ..... 2-7  
ファイルとディレクトリのアクセス  
許可 ..... 2-4

## た

端末制御データベース..... A-10,  
A-13, A-33

## ち

チケット  
先日付けによる要求..... C-28  
フラグ付きで表示..... C-29  
有効期間による要求..... C-28

## て

ディレクトリ  
アクセス許可の変更..... 2-6  
デバイス ..... A-32  
インストレーション..... A-32  
データベース ..... A-11, A-33  
デバイス割り当てデータベース A-13  
データ損失..... A-34  
データベース..... A-12  
エンハンスド・パスワード.. A-36  
グループ..... A-38  
ファイル制御 ..... A-16

## と

トラステッド・イベント ..... 3-9  
トラブルシューティング ..... A-34  
取り外し可能メディアの保護.. A-32

## に

認証  
advanced server for UNIX.... 1-16  
IPsec の概要 ..... 1-17  
Kerberos の概要 ..... 1-16  
概要 ..... 1-1  
公開鍵 ..... 1-6  
証明書 ..... 1-7  
セキュア・シェルの概要 ..... 1-15  
デジタル署名 ..... 1-7  
パスワード..... 1-4  
秘密鍵 ..... 1-8  
ホスト・ベース ..... 1-5  
認証構成  
パスワード変更時間..... A-6

パスワード変更制御..... A-6  
パスワード有効期間..... A-6  
ログインの最大試行回数 ..... A-6  
認証プロファイル ..... A-9, A-36

## ね

---

ネットワーク  
  監査ハブ..... 3-22  
  ネットワークを介した監査... 3-22

## は

---

パスワード  
  BSD ..... 1-11  
  LDAP..... 1-15, D-1  
  NIS ..... 1-14  
  エンハンスト・データベース . A-9  
  拡張 ..... 1-13  
  最大試行回数の構成..... A-6  
  データベース ..... A-38  
  変更 ..... 1-20  
パターン・マッチング  
  を使ったアクセス許可の変更 . 2-7  
バックアップ手順 ..... A-34

## ひ

---

必須のファイル ..... A-35  
表示  
  監査イベント ..... 3-27  
  監査レコード ..... 3-38

## ふ

---

ファイル  
  アクセスの制限 ..... 2-11  
  必須 ..... A-35  
ファイル制御データベース .... A-16  
  説明 ..... A-10  
  保存場所..... A-13  
ファイル属性..... A-39  
物理デバイス..... A-11  
プロセス監査マスク ..... 3-7  
ブート・ロード・ソフトウェア A-39

## へ

---

変更  
  ディレクトリのアクセス許可 . 2-6

## ほ

---

保護  
  リソース..... 2-1

## ゆ

---

ユーザ・アカウント  
  creacct コマンドを使用した作  
  成 ..... C-19  
  MMC インタフェースを使用した作  
  成 ..... C-20  
ユーザ監査マスク  
  設定 ..... 3-7  
ユーザの認証  
  エンハンスト・セキュリティ A-13  
ユーザ・マスク ..... 2-10

アクセス許可の組み合わせ.. 2-11t  
省略時の設定 ..... 2-11  
ログイン・スクリプト ..... 2-13

## り

---

リソース  
ACL..... 2-14  
Tru64 UNIX のアクセス許可.. 2-2  
保護 ..... 2-2, 2-14  
リモート監査..... 3-22

## ろ

---

ロギング・ツール ..... 3-43  
ログイン  
エンハンスト・セキュリティ 1-13

監査マスク, 設定..... 3-7  
最大試行回数の構成..... A-6  
ログイン ID (LUID)..... 3-13  
ログイン・スクリプト  
ユーザ・マスクの設定 ..... 2-13  
ログイン・タイムアウト ..... A-34  
ログ・ファイル ..... 3-3, 3-44  
ロック・ファイル ..... A-34

## わ

---

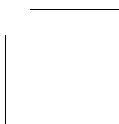
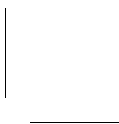
ワイルドカード  
を使ったアクセス許可の変更 . 2-7  
ワークステーション  
取り外し可能メディアの保護 A-32

## Tru64 UNIX ドキュメントの購入方法

Tru64 UNIX ドキュメントのご購入については、弊社担当営業または日本ヒューレット・パッカートの各営業所/代理店にお問い合わせください。

各ドキュメント・キットの注文番号は以下のとおりです。ドキュメント・キットに含まれるマニュアルの内容については『ドキュメント概要』を参照してください。

キット名	注文番号
Tru64 UNIX Documentation CD-ROM	QA-6ADAA-G8
Tru64 UNIX Documentation Kit	QA-6ADAA-GZ
End User Documentation Kit	QA-6ADAB-GZ
- Startup Documentation Kit	QA-6ADAC-GZ
- General User Documentation Kit	QA-6ADAD-GZ
- System and Network Management Documentation Kit	QA-6ADAE-GZ
Developer's Documentation Kit	QA-6ADAF-GZ
Reference Pages Documentation Kit	QA-6ADAG-GZ
TruCluster Server Documentation Kit	QA-6BRAA-GZ
Tru64 UNIX 日本語ドキュメント・キット	QA-6ADJB-GZ
スタートアップ・ドキュメント・キット	QA-6ADJC-GZ
一般ユーザ・ドキュメント・キット	QA-6ADJD-GZ
システム/ネットワーク管理ドキュメント・キット	QA-6ADJE-GZ
プログラミング・ドキュメント・キット	QA-6ADJF-GZ
CDE 翻訳ドキュメント・キット	QA-6ADJG-GZ
TruCluster Server 日本語ドキュメント・キット	QA-05SJA-GZ
Advanced Server for UNIX 日本語ドキュメント・キット	QA-5U2JA-GZ



# マニュアルに対するご意見

Tru64 UNIX  
セキュリティ管理ガイド  
AA-RQ2YB-TE

弊社のマニュアルに関して、ご意見、ご要望、または内容の不明確な部分など、お気づきの点がございましたら、下記にご記入の上、弊社社員にお渡しくださるようお願い申し上げます。

マニュアルの採点：

	大変良い	良い	普通	良くない
正確さ(説明どおりに動作するか)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
情報量(十分か)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
分かり易さ	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
マニュアルの構成	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
図(役立つか)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
例(役立つか)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
索引(項目の検索性)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ページ・レイアウト(情報の検索性)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

内容の不明確な部分がありましたら、以下にご記入ください：

ペー ジ


その他お気づきの点がございましたら、以下にご記入ください：


ご使用のソフトウェアのバージョン： \_\_\_\_\_

貴社名/部課名 \_\_\_\_\_

御名前 \_\_\_\_\_

記入日 \_\_\_\_\_

(注) 当用紙を受け取った弊社社員は、すみやかに下記にお送りください。

ビジネスクリティカルシステム統括本部 **BCS** 技術本部 **Alpha** ソフトウェア技術部