Oracle® Solaris 11.2 でのシステムおよび接続されたデバイスのセキュリティー保護

Part No: E53944-02
2014 年 9 月

ORACLE

目次

このドキュメントの使用法

『*Oracle® Solaris 11.2 でのシステムおよび接続されたデバイスのセキュリティー保護*』では、未承認アクセスから Oracle Solaris システムを保護およびモニターする方法について説明します。

- 「**概要**」– 未承認アクセスからシステムおよびデバイスをセキュリティー保護するさまざまな方法について説明します。
- 「**対象者**」– 企業ネットワーク上にセキュリティーを実装する責任のあるシステム管理者。
- 「**必要な知識**」– Oracle Solaris でサポートされているセキュリティーの概念および機能について熟知していること。

製品ドキュメントライブラリ

この製品の最新情報や既知の問題は、ドキュメントライブラリ (http://www.oracle.com/pls/topic/lookup?ctx=E56342) に含まれています。

Oracle サポートへのアクセス

Oracle のお客様は、My Oracle Support を通じて電子的なサポートを利用することができます。詳細は、http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info (聴覚に障害をお持ちの場合は http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs) を参照してください。

フィードバック

このドキュメントに関するフィードバックを http://www.oracle.com/goto/docfeedback からお聞かせください。

♦ ♦ ♦ 第 1 章

1 マシンセキュリティーの管理

マシンの情報をセキュアな状態に保持することは、システム管理の重要な責任です。この章では、マシンセキュリティー管理の概要を説明します。

システムおよびデバイスをセキュリティー保護する Oracle Solaris 11.2 の新機能

このセクションでは、既存のお客様のために、未承認アクセスからシステムおよびデバイスをセキュリティー保護する、このリリースの重要な新機能に関する情報に焦点を当てます。

- ブート検証がサポートされることによって、オペレーティングシステムのカーネルが保護されます。詳細は、33 ページの「ベリファイドブートの使用」を参照してください。
- Trusted Platform Modules (TPM) のサポート。詳細は、41 ページの「Oracle Solaris システムでの TPM の初期化」を参照してください。

コンピュータシステムへのアクセスを制御する

ワークスペースでは、サーバーに接続されたすべてのコンピュータを 1 つの大規模多重システムと見なすことができます。システム管理者は、この大規模なシステムのセキュリティー管理に責

任があります。システム管理者は、ネットワークの外部からの侵入を防ぐ必要があります。また、ネットワーク内部のコンピュータ上のデータの完全性を確保する必要もあります。

ファイルレベルにおいて、Oracle Solaris には標準セキュリティー機能が組み込まれており、ファイル、ディレクトリ、およびデバイスを保護するために使用できます。システムレベルとネットワークレベルでは、セキュリティーの内容はほぼ同じです。以降のセクションで説明されているように、セキュリティー防御の第一線はシステムへのアクセスを制御することです。

物理的なセキュリティーの管理

システムへのアクセスを制御するには、コンピュータ環境の物理的なセキュリティーを管理する必要があります。たとえば、システムにログインしたままこれを放置することは未承認アクセスを招く原因になります。侵入者がオペレーティングシステムやネットワークにアクセスしないとも限らないからです。コンピュータの周辺環境やコンピュータハードウェアは、不当なアクセスから物理的に保護される必要があります。

ハードウェア設定に対する未承認アクセスから SPARC システムを保護できます。`eeprom` コマンドを使って、パスワードがないと PROM にアクセスできないようにしてください。詳細は、59 ページの「SPARC ハードウェアへのアクセスにパスワードを必要にする方法」を参照してください。x86 ハードウェアを保護するには、ベンダーのドキュメントを参照してください。

ログイン制御の管理

パスワード割り当てとログイン制御によって、システムやネットワークへの未承認のログインを防止できます。パスワードはシンプルな認証メカニズムです。システム上のすべてのアカウントには、パスワードが必要です。アカウントにパスワードを設定しないと、ユーザー名を推測できる侵入者であれば誰でもネットワーク全体にアクセスできることになります。力ずくの野蛮な攻撃を許さないためには、強力なパスワードアルゴリズムが必要です。

ユーザーがシステムにログインすると、`login` コマンドはネームスイッチサービス `svc:/system/name-service/switch` 内の情報に従って、該当するネームサービスまたはディレクトリサービスデータベースを確認します。ネームサービスデータベースの値を変更するには、SMF コマンドを使用します。ネームサービスは、ログインに影響を与えるデータベースの場所を示します。

- `files` – ローカルシステムの `/etc` ファイルを指定します
- `ldap` – LDAP サーバーの LDAP ディレクトリサービスを指定します

- `nis` – NIS マスターサーバーの NIS データベースを指定します
- `dns` – ネットワーク上のドメインネームサービスを指定します。

ネームサービスについては、`nscd`(1M) のマニュアルページを参照してください。ネームサービスおよびディレクトリサービスについては、『Oracle Solaris 11.2 ディレクトリサービスとネームサービスでの作業: DNS と NIS 』および『Oracle Solaris 11.2 ディレクトリサービスとネームサービスでの作業: LDAP 』を参照してください。

`login` コマンドは、ユーザーによって指定されたユーザー名とパスワードを検証します。ユーザー名がパスワードデータベース内に存在しない場合、`login` コマンドはシステムへのアクセスを拒否します。あるいは、指定されたユーザー名に対するパスワードが正しくないと、`login` コマンドはシステムへのアクセスを拒否します。有効なユーザー名とそれに対応するパスワードが入力されれば、システムはシステムへのアクセスをユーザーに認可します。

PAM モジュールには、システムへのログインが正常に完了したあとのアプリケーションへのログインを効率化できます。詳細は、『Oracle Solaris 11.2 での Kerberos およびその他の認証サービスの管理 』の第 1 章「プラグイン可能認証モジュールの使用」を参照してください。

Oracle Solaris システムには、精巧な認証メカニズムと承認メカニズムが備わっています。ネットワークレベルでの認証メカニズムや承認メカニズムについては、29 ページの「リモートアクセスの認証と承認」を参照してください。

パスワード情報の管理

ユーザーはシステムにログインするときに、ユーザー名とパスワードの両方を入力する必要があります。ログイン名は公開されていますが、パスワードは秘密にしなければなりません。ユーザーは、自分のパスワードを他人に知られてはいけません。ユーザーは、自分のパスワードを慎重に選択し、頻繁に変更する必要があります。

パスワードは、最初にユーザーアカウントを設定するときに作成されます。ユーザーアカウントのセキュリティーを管理するために、パスワード有効期限を設定し、パスワードを定期的に強制変更することができます。また、ユーザーアカウントを無効にして、パスワードをロックすることもできます。パスワードの管理の詳細は、『Oracle Solaris 11.2 のユーザーアカウントとユーザー環境の管理 』の第 1 章「ユーザーアカウントとユーザー環境について」および `passwd`(1) のマニュアルページを参照してください。

ローカルパスワード

ネットワークでローカルファイルを使用してユーザーを認証している場合、パスワード情報はシステムの /etc/passwd ファイルと /etc/shadow ファイルに保持されます。ユーザー名などの情報は、/etc/passwd ファイルに保持されます。暗号化されたパスワード自体は、個別の*シャドウ*ファイル (/etc/shadow) に保持されます。このセキュリティー方式によって、暗号化されたパスワードにアクセスされることを防ぎます。/etc/passwd ファイルは、システムにログインできるすべてのユーザーが使用できますが、/etc/shadow ファイルを読み取ることができるのは root アカウントだけです。passwd コマンドを使用すると、ローカルシステム上のユーザーのパスワードを変更できます。

NIS パスワード

ネットワークで NIS を使用してユーザーを認証している場合、パスワード情報は NIS パスワードマップに保持されます。NIS では、パスワードの有効期間を指定できません。NIS パスワードマップに保持されているユーザーのパスワードを変更するには、コマンド passwd -r nis を使用します。

LDAP パスワード

Oracle Solaris の LDAP ネームサービスは、パスワード情報とシャドウ情報を LDAP ディレクトリツリーの ou=people コンテナに格納します。Oracle Solaris LDAP ネームサービスクライアントでユーザーのパスワードを変更するには、passwd -r ldap コマンドを使用します。LDAP ネームサービスは、パスワードを LDAP リポジトリに格納します。

パスワードポリシーは Oracle Directory Server Enterprise Edition で適用されます。具体的には、クライアントの pam_ldap モジュールは Oracle Directory Server Enterprise Edition で適用されているパスワードポリシー制御に従います。詳細は、『Oracle Solaris 11.2 ディレクトリサービスとネームサービスでの作業: LDAP 』の「LDAP ネームサービスのセキュリティーモデル」を参照してください。

パスワードの暗号化

パスワードの強力な暗号化は攻撃に対する最初の障壁になります。Oracle Solaris ソフトウェアには 6 つのパスワード暗号化アルゴリズムが用意されています。Blowfish および SHA アルゴリズムは、強力なパスワード暗号化を提供します。

注記 - FIPS 140 で承認されるには、SHA アルゴリズムを使用してください。詳細は、『Using a FIPS 140 Enabled System in Oracle Solaris 11.2 』の「passwd Command as a FIPS 140 Consumer」を参照してください。

パスワードアルゴリズムの識別子

サイトのアルゴリズムの構成は、`/etc/security/policy.conf` ファイルに指定します。`policy.conf` ファイルには、次の表に示す識別子でアルゴリズムを指定します。識別子とアルゴリズムのマッピングについては、`/etc/security/crypt.conf` ファイルを参照してください。

注記 - 可能な場合は、FIPS 承認アルゴリズムを使用してください。FIPS 承認アルゴリズムのリストについては、『Using a FIPS 140 Enabled System in Oracle Solaris 11.2 』の「FIPS 140 Algorithm Lists and Certificate References for Oracle Solaris Systems」を参照してください。

表 1-1 パスワードの暗号化アルゴリズム

識別子	説明	アルゴリズムのマニュアルページ
`1`	BSD システムや Linux システムの MD5 アルゴリズムと互換性のある MD5 アルゴリズム。	`crypt_bsdmd5`(5)
`2a`	BSD システムの Blowfish アルゴリズムと互換性のある Blowfish アルゴリズム。 **注記 -** FIPS 140 セキュリティーを向上させるには、`/etc/security/policy.conf` ファイル内の `CRYPT_ALGORITHMS_ALLOW=2a,5,6` エントリから Blowfish アルゴリズム (2a) を削除します。	`crypt_bsdbf`(5)
`md5`	BSD バージョンや Linux バージョンの MD5 よりも強力とされている Sun MD5 アルゴリズム。	`crypt_sunmd5`(5)
`5`	SHA256 アルゴリズム。SHA は、Secure Hash Algorithm (セキュアハッシュアルゴリズム) を表します。このアルゴリズムは、SHA-2 ファミリのメンバーです。SHA256 では 255 文字のパスワードがサポートされます。このアルゴリズムがデフォルトです (`(CRYPT_DEFAULT)`)。	`crypt_sha256`(5)
`6`	SHA512 アルゴリズム。	`crypt_sha512`(5)

識別子	説明	アルゴリズムのマニュアルページ
__unix__	非推奨。従来の UNIX 暗号化アルゴリズム。このアルゴリズムは、古いシステムに接続するときに使用できます。	crypt_unix(5)

注記 - そのユーザーの新しいパスワードを生成する際は、ユーザーの初期パスワードに使用されたアルゴリズムが引き続き使用されます (ユーザーの新しいパスワードを生成する前に、別のデフォルトアルゴリズムが選択された場合でも)。このメカニズムは、次の条件で適用されます。

- アルゴリズムがパスワード暗号化で使用することが許可されているアルゴリズムのリストに含まれている。
- 識別子が _unix_ 以外である。

パスワード暗号化のアルゴリズムを切り替える方法については、53 ページの「パスワード暗号化のデフォルトアルゴリズムを変更する」を参照してください。

policy.conf ファイルのアルゴリズム構成

policy.conf ファイルでは、デフォルトのアルゴリズムが次のように構成されています。

```
#
…
# crypt(3c) Algorithms Configuration
#
# CRYPT_ALGORITHMS_ALLOW specifies the algorithms that are allowed
to
# be used for new passwords.  This is enforced only in crypt_gensalt(3c).
#
CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6

# To deprecate use of the traditional unix algorithm, uncomment below
# and change CRYPT_DEFAULT= to another algorithm.  For example,
# CRYPT_DEFAULT=1 for BSD/Linux MD5.
#
#CRYPT_ALGORITHMS_DEPRECATE=__unix__

# The Oracle Solaris default is a SHA256 based algorithm.  To revert to
# the policy present in Solaris releases set CRYPT_DEFAULT=__unix__,
# which is not listed in crypt.conf(4) since it is internal to libc.
#
CRYPT_DEFAULT=5
…
```

CRYPT_DEFAULT の値を変更すると、新しいユーザーのパスワードは、新しい値に対応するアルゴリズムを使って暗号化されます。

既存のユーザーがパスワードを変更したときに新しいパスワードがどのアルゴリズムで暗号化されるかは、古いパスワードがどのように暗号化されているかによって異なります。たとえば、CRYPT_ALGORITHMS_ALLOW=1,2a,md5,5,6 かつ CRYPT_DEFAULT=6 であるとします。次の表は、パスワードの暗号化にどのアルゴリズムが使用されるかを示します。パスワードは「識別子 = アルゴリズム」で構成されます。

元のパスワード	変更後のパスワード	説明
1 = crypt_bsdmd5	同じアルゴリズムを使用します。	1 識別子は CRYPT_ALGORITHMS_ALLOW リストにあります。ユーザーのパスワードは引き続き crypt_bsdmd5 アルゴリズムで暗号化されます。
2a = crypt_bsdbf	同じアルゴリズムを使用します。	2a 識別子は CRYPT_ALGORITHMS_ALLOW リストにあります。このため、新しいパスワードは crypt_bsbdf アルゴリズムで暗号化されます。
md5 = crypt_md5	同じアルゴリズムを使用します。	md5 識別子は CRYPT_ALGORITHMS_ALLOW リストにあります。このため、新しいパスワードは crypt_md5 アルゴリズムで暗号化されます。
5 = crypt_sha256	同じアルゴリズムを使用します。	5 識別子は CRYPT_ALGORITHMS_ALLOW リストにあります。このため、新しいパスワードは引き続き crypt_sha256 アルゴリズムで暗号化されます。
6 = crypt_sha512	同じアルゴリズムを使用します。	6 識別子は CRYPT_DEFAULT の値です。このため、新しいパスワードは引き続き crypt_sha512 アルゴリズムで暗号化されます。
__unix__ = crypt_unix	crypt_sha512 アルゴリズムを使用します。	__unix__ 識別子は CRYPT_ALGORITHMS_ALLOW リストにありません。このため、crypt_unix アルゴリズムを使用することはできません。新しいパスワードは CRYPT_DEFAULT アルゴリズムで暗号化されます。

選択したアルゴリズムの構成の詳細については、policy.conf(4) のマニュアルページを参照してください。パスワード暗号化アルゴリズムを指定する場合は、53 ページの「パスワード暗号化のデフォルトアルゴリズムを変更する」を参照してください。

特殊なシステムアカウント

root アカウントは特殊な*システム*アカウントの 1 つです。これらのアカウントのうち、root アカウントにのみパスワードが割り当てられ、ログインできます。nuucp アカウントはファイル転送用にログインできます。他のシステムアカウントは、ファイルを保護したり、または root の完全な権限を使用せずに管理プロセスを実行したりします。

注意 - システムアカウントのパスワード設定は決して変更しないでください。Oracle Solaris からのシステムアカウントは、安全かつ確実な状態で配布されます。UID が 101 以下のシステムファイルは修正したり、作成したりしないでください。

次の表に、一部のシステムアカウントとその使用方法の一覧を示します。システムアカウントは特殊な機能を実行します。この一覧の各アカウントは、100 より小さい UID を持ちます。システムファイルの完全なリストを表示するには、`logins -s` コマンドを使用します。

表 1-2 選択されたシステムアカウントとその使用

システムアカウント	UID	用途
`root`	`0`	ほぼ無制限です。他の保護および許可をオーバーライドできます。`root` アカウントはシステム全体へのアクセス権を持ちます。`root` アカウントのパスワードは、非常に注意深く保護するようにしてください。`root` アカウントは、ほとんどの Oracle Solaris コマンドを所有しています。
`daemon`	`1`	バックグラウンド処理を制御します。
`bin`	`2`	一部の Oracle Solaris コマンドを所有します。
`sys`	`3`	多数のシステムファイルを所有します。
`adm`	`4`	一部のシステム管理ファイルを所有します。
`lp`	`71`	プリンタ用のオブジェクトデータファイルとスプールデータファイルを所有します。
`uucp`	`5`	UNIX 間のコピープログラム、UUCP 用のオブジェクトデータファイルとスプールデータファイルを所有します。
`nuucp`	`9`	システムにログインしてファイル転送を開始するためにリモートシステムで使用されます。

リモートログイン

侵入者にとって、リモートログインは魅力的な手段です。Oracle Solaris は、リモートログインをモニター、制限、および無効にする、いくつかのコマンドを提供します。手順については、表 3-1「ログインとパスワードの保護タスクマップ」を参照してください。

デフォルトでは、システムのマウスやキーボード、フレームバッファー、オーディオデバイスなど、ある種のシステムデバイスについては、リモートログインを通して制御したり読み取ったりすることはできません。詳細は、`logindevperm(4)` のマニュアルページを参照してください。

デバイスアクセスの制御

コンピュータシステムに接続された周辺機器は、セキュリティーリスクをもたらします。たとえば、マイクは会話をキャッチし、その会話をリモートシステムに送信します。CD-ROM の場合、その

情報を CD-ROM に残して、CD-ROM デバイスを次に使うユーザーが読み取れるようにすることができます。プリンタは、リモートサイトからもアクセスできます。システムの必須デバイス (たとえば、`bge0` などのネットワークインタフェース) もまた、セキュリティー問題を引き起こす可能性があります。

Oracle Solaris ソフトウェアには、デバイスへのアクセスを制御するための方法がいくつか用意されています。

- **デバイスポリシーを設定する** – 特定のデバイスにアクセスしているプロセスが特定の特権セットで実行されるように要求できます。それらの権限を持たないプロセスは、そのデバイスを使用できません。ブート時に、Oracle Solaris ソフトウェアはデバイスポリシーを構成します。サードパーティーのドライバは、そのインストール時にデバイスポリシーを構成できます。インストール後、管理者はデバイスポリシーをデバイスに追加できます。
- **デバイスを割り当て可能にする** – ユーザーがデバイスを使用する前に割り当てる必要があるように要求できます。割り当てによって、デバイスの使用が一度に 1 人のユーザーに制限されます。さらに、ユーザーがそのデバイスの使用を承認されていることを要求できます。
- **デバイスの使用を防ぐ** – コンピュータシステム上のどのユーザーも特定のデバイス (マイクなど) を使用できないように設定できます。たとえば、ある種のデバイスを使用できないようにする例としては、コンピュータキオスクが挙げられます。
- **デバイスを特定のゾーンに限定する** – デバイスの使用を非大域ゾーンに割り当てることができます。詳細は、『Oracle Solaris ゾーンの作成と使用 』の「非大域ゾーンでのデバイスの使用」を参照してください。デバイスおよびゾーンのより一般的な説明については、『Oracle Solaris ゾーンの紹介 』の「非大域ゾーンの /dev ファイルシステム」を参照してください。

デバイスポリシー

デバイスポリシーメカニズムを使用することで、デバイスを開こうとするプロセスに特定の権限を要求するように指定できます。デバイスポリシーによって保護されたデバイスをアクセスできるのは、デバイスポリシーで指定されている権限で稼働しているプロセスだけです。Oracle Solaris はデフォルトのデバイスポリシーを提供します。たとえば、`bge0` などのネットワークインタフェースでは、そのインタフェースにアクセスするプロセスが `net_rawaccess` 特権で実行されていることが必要です。この要件はカーネルで適用されます。特権の詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「プロセス権管理」を参照してください。

Oracle Solaris では、デバイスはファイルアクセス権とデバイスポリシーで保護されます。たとえば、/dev/ip ファイルのアクセス権は 666 です。しかし、このデバイスは適切な権限を持つプロセスによってしかオープンできません。

デバイスポリシーの構成は監査の対象とすることができます。デバイスポリシーの変更は、AUE_MODDEVPLCY 監査イベントによって記録されます。

デバイスポリシーの詳細は、次のページを参照してください。

- 表4-1「デバイスポリシーの構成タスクマップ」
- 75 ページの「デバイスポリシーコマンド」
- 『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「特権とデバイス」

デバイスの割り当て

デバイス割り当てメカニズムを使用すれば、CD-ROM などの周辺機器に対するアクセスを制限できます。デバイス割り当てが有効になっていない場合、周辺機器の保護はファイルアクセス権によってのみ行われます。たとえば、デフォルトでは周辺機器は次のように使用できます。

- CD-ROM ドライブまたはディスクの読み取りと書き込みは、任意のユーザーが行うことができます。
- すべてのユーザーがマイクを接続できます。
- すべてのユーザーが接続されたプリンタにアクセスできます。

デバイス割り当てを行うことで、承認されたユーザーにだけデバイスの使用を限定できます。デバイス割り当てによって、デバイスアクセスを完全に防ぐこともできます。デバイスを割り当てるユーザーは、そのユーザー自身が割り当てを解除するまでそのデバイスを独占的に使用できます。デバイスの割り当てが解除される際には、残っているすべてのデータがデバイスクリーンスクリプトによって消去されます。デバイスにスクリプトがない場合には、デバイスクリーンスクリプトを作成してそのデバイスから情報を一掃できます。この例は、83 ページの「新しいデバイスクリーンスクリプトの作成」を参照してください。

デバイス割り当てに関連した試み (デバイスの割り当て、デバイスの割り当て解除、割り当て可能なデバイスの一覧表示) は、監査の対象とすることができます。監査イベントは、other 監査クラスの一部です。

デバイス割り当てについての詳細は、次を参照してください。

- 表4-2「デバイス割り当ての管理タスクマップ」

- 76 ページの「デバイスの割り当て」
- 77 ページの「デバイス割り当てコマンド」

マシンリソースへのアクセス制御

一部のシステムリソースは、デフォルトで保護されています。さらに、システム管理者はシステムの動作状態を制御したり、モニターしたりすることができます。システム管理者は、だれがどのリソースを使用できるかを制限したり、リソースの使用状況を記録したり、だれがリソースを使用しているかをモニターしたりできます。システム管理者は、リソースの不適切な使用を最小限に抑えるようにシステムを設定することもできます。

アドレス空間レイアウトのランダム化

Oracle Solaris では、アドレス空間レイアウトのランダム化 (ASLR) を有効にするために、そのユーザーランドバイナリの多くにタグが付けられます。ASLR では、アドレス空間の主要な部分の開始アドレスがランダム化されます。このセキュリティー防御メカニズムにより、ソフトウェアの脆弱性を悪用しようとする ROP (Return Oriented Programming) 攻撃を失敗させることができます。

ゾーンは、そのプロセス用にこのランダム化されたレイアウトを継承します。ASLR の使用はすべてのバイナリに最適であるとは限らないため、ASLR の使用は、ゾーンのレベルとバイナリのレベルで構成できます。

ASLR の構成は次の 3 つです。

- **無効** – ASLR は、すべてのバイナリに対して無効です。
- **タグ付きバイナリ** – ASLR は、バイナリ内にコーディングされているタグによって制御されます。

  Oracle Solaris での ASLR のデフォルト値は、`tagged-binaries` です。ASLR を使用するために Oracle Solaris リリースの多くのバイナリにタグが付いています。
- **有効** – ASLR は、無効にするためのタグが明示的に付いているバイナリを除くすべてのバイナリに対して有効です。

`sxadm` コマンドは、ASLR を構成するために使用されます。このコマンドを実行するには、`root` 役割になる必要があります。例および情報については、`sxadm`(1M) のマニュアルページを参

照してください。開発者向けの情報については、『Oracle Solaris 11 セキュリティー開発者ガイド 』を参照してください。

スーパーユーザーアクセスの制限とモニタリング

システムでスーパーユーザーアクセスを行うには、root パスワードが必要です。デフォルトの構成では、ユーザーはリモートからシステムに root としてログインできません。リモートログインするときに、ユーザーは自分のユーザー名でログインしてから、su コマンドを使用して root になる必要があります。管理者は、必要に応じて su コマンドを使用中のユーザー (特にスーパーユーザーのアクセス権を取得しようとしているユーザー) をモニターできます。スーパーユーザーをモニタリングしたり、スーパーユーザーのアクセス権を制限したりする手順については、56 ページの「root アクセスのモニタリングと制限」を参照してください。

役割に基づくアクセス制御を構成してスーパーユーザーを置き換える

Oracle Solaris の機能である役割に基づくアクセス制御 (RBAC) は、スーパーユーザーの権限を管理役割に分散するように設計されています。スーパーユーザーすなわち root ユーザーは、システムのすべてのリソースにアクセスできますが、RBAC を使用すると、root の責任の多くを、個別の権限を持つ一連の役割に置き換えることができます。たとえば、ユーザーアカウントの作成を処理する 1 つの役割と、システムファイルの変更を処理する別の役割を設定できます。root アカウントを変更しない場合でも、このアカウントを役割として残し、その役割を割り当てないようにできます。この方法によって、システムへの root アクセスが事実上削除されます。

各役割を使用するには、既知のユーザーが自分のユーザー名とパスワードを使用してログインする必要があります。ログインしたユーザーは、特別な役割パスワードを入力してその役割を引き受けます。RBAC の詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「ユーザー権管理」を参照してください。

システムリソースの意図しない誤用の回避

システム管理者は、自分自身やユーザーによって意図しないエラーが引き起こされないように防止できます。

- たとえば、PATH 変数を正しく設定することによって、トロイの木馬の実行を防止できます。

- 制限されたシェルをユーザーに割り当てることもできます。システムのうち各人の作業に必要な部分だけをユーザーに提供するという方法でシェル機能を制限すると、ユーザーエラーを避けることができます。実際、慎重に設定すれば、作業を能率的に行う上で必要な部分以外にユーザーがアクセスできないように制限できます。
- そのユーザーがアクセスする必要がないファイルには、限定的なアクセス権を設定できます。

PATH 変数の設定

よく注意して、PATH 変数を正しく設定してください。そうしなければ、だれかが持ち込んだプログラムを誤って実行してしまい、セキュリティーが危険にさらされる可能性があります。データを壊したりシステムを損傷したりするおそれがあります。このようなプログラムは、「*トロイの木馬*」と呼ばれます。たとえば、公開ディレクトリの中に別の su プログラムが置かれていると、システム管理者が気づかずに実行してしまう可能性があります。このようなスクリプトは正規の su コマンドとまったく同じに見えます。このようなスクリプトは実行後に自らを削除してしまうため、トロイの木馬が実際に実行されたという証拠はほとんど残りません。

PATH 変数はログイン時に自動的に設定されます。このパスは、.bashrc や /etc/profile などの初期設定ファイルを通して設定されます。現在のディレクトリ (.) への検索パスを最後に指定すれば、トロイの木馬のようなタイプのプログラムを実行するのを防ぐことができます。root アカウントの PATH 変数には現在のディレクトリを一切含めないようにしてください。

ユーザーに制限付きシェルを割り当てる

標準シェルを使用すると、ユーザーはファイルを開く、コマンドを実行するなどの操作を行うことができます。制限付きシェルを使用すると、ディレクトリの変更やコマンドの実行などのユーザー能力を制限できます。制限付きシェルは、/usr/lib/rsh コマンドで呼び出されます。制限付きシェルは、リモートシェル /usr/sbin/rsh ではありません。

標準のシェルと異なる点は次のとおりです。

- ユーザーのアクセスはホームディレクトリ内に限定されるため、ユーザーは cd コマンドを使用してディレクトリを変更できません。したがって、システムファイルを閲覧することはできません。
- ユーザーは PATH 変数を変更できないため、システム管理者によって設定されたパスのコマンドしか使用できません。さらに、完全なパス名を使ってコマンドやスクリプトを実行することもできません。

- ユーザーは、> または >> を使用して出力をリダイレクトできません。

制限付きシェルでは、ユーザーが使用できるシステムファイルを制限できます。このシェルは、特定のタスクを実行するユーザーのために限られた環境を作成します。ただし、制限付きシェルは完全にセキュアなわけではなく、あくまでも経験の少ないユーザーが誤ってシステムファイルを損傷するのを防止することことが目的です。

制限付きシェルについては、`man -s1m rsh` コマンドを使用して `rsh`(1M) のマニュアルページを参照してください。

ファイル内のデータへのアクセス制限

Oracle Solaris はマルチユーザー環境なので、ファイルシステムのセキュリティーは、システムのもっとも基本的なセキュリティーリスクです。ファイルの保護には、従来の UNIX のファイル保護と、より確実なアクセス制御リスト (ACL) との両方が使用できます。

あるユーザーには一部のファイルの読み取りを許可したり、別のユーザーには一部のファイルを変更または削除するアクセス権を許可したりできます。一方、あるデータを、どのユーザーからも読み取られないよう設定することもできます。『Oracle Solaris 11.2 でのファイルのセキュリティー保護とファイル整合性の検証 』の第 1 章「ファイルアクセスの制御」では、ファイルアクセス権の設定方法について説明されています。

setuid 実行可能ファイルの制限

実行可能ファイルがセキュリティーリスクとなる場合があります。いくつかの実行可能プログラムは引き続き、正しく機能するには `root` として実行する必要があります。これらの `setuid` プログラムは、ユーザー ID が `0` に設定された状態で実行されます。このようなプログラムはだれが実行したとしても `root` ID で実行されます。`root` ID で動作するプログラムは、プログラムがセキュリティーを念頭に置いて作成されていない限り、セキュリティーの問題をはらんでいます。

Oracle Solaris が `setuid` ビットを `root` に設定して提供する実行可能プログラムを除き、`setuid` プログラムの使用を禁止することをお勧めします。`setuid` プログラムの使用を禁止できない場合は、その使用を制限する必要があります。しっかりした管理を行うためには `setuid` プログラムの数を少なくする必要があります。

詳細は、『Oracle Solaris 11.2 でのファイルのセキュリティー保護とファイル整合性の検証 』の「実行可能ファイルを原因とするセキュリティーへの悪影響を防止する」を参照してください。

手順については、『Oracle Solaris 11.2 でのファイルのセキュリティー保護とファイル整合性の検証 』の「セキュリティーリスクのあるプログラムからの保護」を参照してください。

デフォルトでのセキュリティー強化 (Secure By Default) 構成の使用

デフォルトでは、Oracle Solaris がインストールされると、一連の多数のネットワークサービスが無効になります。この構成は「デフォルトでのセキュリティー強化 (Secure By Default)」(SBD) と呼ばれます。SBD により、ネットワークリクエストを受け入れるネットワークサービスは `sshd` デーモンだけになります。ほかのネットワークサービスはすべて無効になるか、ローカル要求だけを処理するようになります。`ftp` などの個々のネットワークサービスを有効にするには、Oracle Solaris のサービス管理機能 (SMF) を使用します。詳細は、`netservices`(1M) および `smf`(5) のマニュアルページを参照してください。

リソース管理機能の使用

Oracle Solaris ソフトウェアには、精巧なリソース管理機能があります。これらの機能を使用することで、サーバー統合環境内のアプリケーションによるリソース利用の割り当て、スケジュール、モニター、上限設定などを行うことができます。リソース制御フレームワークにより、プロセスが使用するシステムリソースを制限できます。このような制約を行うことで、システムリソースを混乱させようとするスクリプトによるサービス拒否攻撃を防ぎやすくなります。

これらのリソース管理機能により、特定のプロジェクトに対してリソースを指定できます。また、使用できるリソースを動的に調整することもできます。詳細は、『Oracle Solaris 11.2 でのリソースの管理 』を参照してください。

Oracle Solaris ゾーンの使用

Oracle Solaris ゾーンは、単一の Oracle Solaris OS インスタンス内に存在するほかのシステムからプロセスが分離されるアプリケーション実行環境です。この分離を行うことで、1 つのゾーン内で稼働しているプロセスがほかのゾーンで稼働しているプロセスをモニタリングしたりそれらのプロセスに影響を及ぼしたりすることが防止されます。これは、スーパーユーザー権限によって稼働しているプロセスでも同様です。

Oracle Solaris ゾーンは、単一のサーバー上にアプリケーションを複数配置する環境に適しています。詳細は、『Oracle Solaris ゾーンの紹介 』を参照してください。

マシンリソースの使用状況のモニタリング

システム管理者は、システムの動作をモニターする必要があります。次の点を含め、マシンのあらゆる側面に注意する必要があります。

- 通常の負荷はどの程度か
- 誰がシステムへのアクセス権を持っているか
- 各ユーザーはいつシステムにアクセスするか
- システムでは通常どのようなプログラムを実行するか

このような情報を把握していれば、ツールを使用してシステムの使用状況を監査し、各ユーザーのアクティビティーをモニターできます。セキュリティー侵害と思われる場合は、モニタリング作業が特に役立ちます。監査サービスの詳細は、『Oracle Solaris 11.2 での監査の管理 』の第 1 章「Oracle Solaris での監査について」を参照してください。

ファイルの整合性のモニタリング

システム管理者は、管理対象のシステムにインストールされたファイルが予想外の方法で変更されないことを保証する必要があります。大規模インストールでは、各システム上のソフトウェアスタックの比較や報告を行うツールを使用すればシステムの追跡、記録が行えます。基本監査報告機能 (BART) を使用すると、一定期間にわたって1 つ以上のシステムをファイルレベルでチェックし、システムを包括的に検証できます。一定期間にわたってすべてのシステムまたは 1 つのシステムにおける BART目録の変化を調べることで、システムの整合性を検証できます。BART には、目録作成機能、目録比較機能、レポート生成規則などが用意されています。詳細は、『Oracle Solaris 11.2 でのファイルのセキュリティー保護とファイル整合性の検証 』の第 2 章「BART を使用したファイル整合性の検証」を参照してください。

ファイルアクセスの制御

Oracle Solaris は、システムにログインしているすべてのユーザーが、ほかのユーザーに属しているファイルを読み取ることができるマルチユーザー環境です。さらに、適切なアクセス権をもっているユーザーは、ほかのユーザーに属しているファイルを使用できます。詳細は、『Oracle

Solaris 11.2 でのファイルのセキュリティー保護とファイル整合性の検証 』の第 1 章「ファイルアクセスの制御」を参照してください。ファイルに適切なアクセス権を設定する手順については、『Oracle Solaris 11.2 でのファイルのセキュリティー保護とファイル整合性の検証 』の「ファイルの保護」を参照してください。

ディスク上のファイルの暗号化

ほかのユーザーがアクセスできないようにすることによって、ファイルを安全に保つことができます。たとえば、600 のアクセス権を持つファイルは、その所有者と root アカウントを除き、読み取ることができません。アクセス権 700 の付いたディレクトリも同様です。ただし、ほかのだれかがユーザーパスワードや root パスワードを推測して発見すると、そのファイルにアクセスできます。さらに、アクセス不能なはずのファイルも、システムファイルのバックアップをオフラインメディアにとるたびに、バックアップテープ上に保存されます。保護を強化するために、ディスク上の暗号化または暗号化フレームワークのコマンドを使用できます。

ZFS ファイルシステムの詳細は、『Oracle Solaris 11.2 での ZFS ファイルシステムの管理 』の「ZFS ファイルシステムの暗号化」を参照してください。

暗号化フレームワークは、digest、mac、および encrypt コマンドを提供します。通常のユーザーは、これらのコマンドを使用してファイルやディレクトリを保護することができます。詳細は、『Oracle Solaris 11.2 での暗号化と証明書の管理 』の第 1 章「暗号化フレームワーク」を参照してください。

アクセス制御リストの使用

ACL (「アクル」と読む) では、ファイルアクセス権の制御をより強化できます。ACL は、従来の UNIX ファイル保護機能では不十分な場合に追加で使用します。従来の UNIX ファイル保護機能は、所有者、グループ、その他のユーザーという 3 つのユーザークラスに読み取り権、書き込み権、実行権を提供します。ACL では、ファイルセキュリティーを管理するレベルがさらに詳細になります。

ACL を使用すると、次に示すような、きめ細かいファイルアクセス権を定義できます。

- 所有者のファイルアクセス権
- 所有者のグループのファイルアクセス権
- 所有者のグループに属していないユーザーのファイルアクセス権

- 特定ユーザーのファイルアクセス権
- 特定グループのファイルアクセス権
- 以上のカテゴリそれぞれのデフォルトアクセス権

アクセス制御リスト (ACL) を使用して ZFS ファイルを保護する方法については、『Oracle Solaris 11.2 での ZFS ファイルシステムの管理 』の第 7 章「ACL および属性を使用した Oracle Solaris ZFS ファイルの保護」を参照してください。レガシーファイルシステムでの ACL の使用については、『Oracle Solaris 11.2 でのファイルのセキュリティー保護とファイル整合性の検証 』の「アクセス制御リストによる UFS ファイルの保護」を参照してください。

マシン間でのファイルの共有

ネットワークファイルサーバーは、どのファイルを共有できるかを制御できます。また、共有ファイルにアクセスできるクライアント、およびそれらのクライアントに許可するアクセス権の種類も制御します。ファイルサーバーは、すべてのクライアントまたは特定のクライアントに、読み取り権と書き込み権、または読み取り専用アクセス権を与えることができます。アクセス制御は、`share` コマンドでリソースを利用可能にするときに指定します。

ZFS ファイルシステムの NFS 共有を作成すると、共有を削除するまでファイルシステムは永続的に共有されます。システムをリブートすると、SMF は共有を自動的に管理します。詳細は、『Oracle Solaris 11.2 での ZFS ファイルシステムの管理 』の「Oracle Solaris ZFS ファイルシステムと従来のファイルシステムの相違点」を参照してください。

共有ファイルへの `root` アクセスの制限

通常、スーパーユーザーは、ネットワーク上で共有されるファイルシステムには `root` としてアクセスできません。NFS システムは、要求者のユーザーをユーザー ID `60001` を持つユーザー `nobody` に変更することによって、マウントされているファイルシステムへの `root` アクセスを防止します。ユーザー `nobody` のアクセス権は、公共ユーザーに与えられているアクセス権と同じです。つまり、ユーザー `nobody` のアクセス権は資格をもたないユーザーのものと同じです。たとえば、ファイルの実行権しか公共に許可していなければ、ユーザー `nobody` はそのファイルを実行することしかできません。

NFS サーバーは、共有ファイルシステムへの `root` アクセスをホスト単位で与えることができます。これらの特権を付与するには、`share` コマンドの `root=`*hostname* オプションを使用しま

す。このオプションは慎重に使用してください。NFS でのセキュリティーオプションについては、『Oracle Solaris 11.2 でのネットワークファイルシステムの管理 』の第 5 章「ネットワークファイルシステムを管理するためのコマンド」を参照してください。

ネットワークアクセスの制御

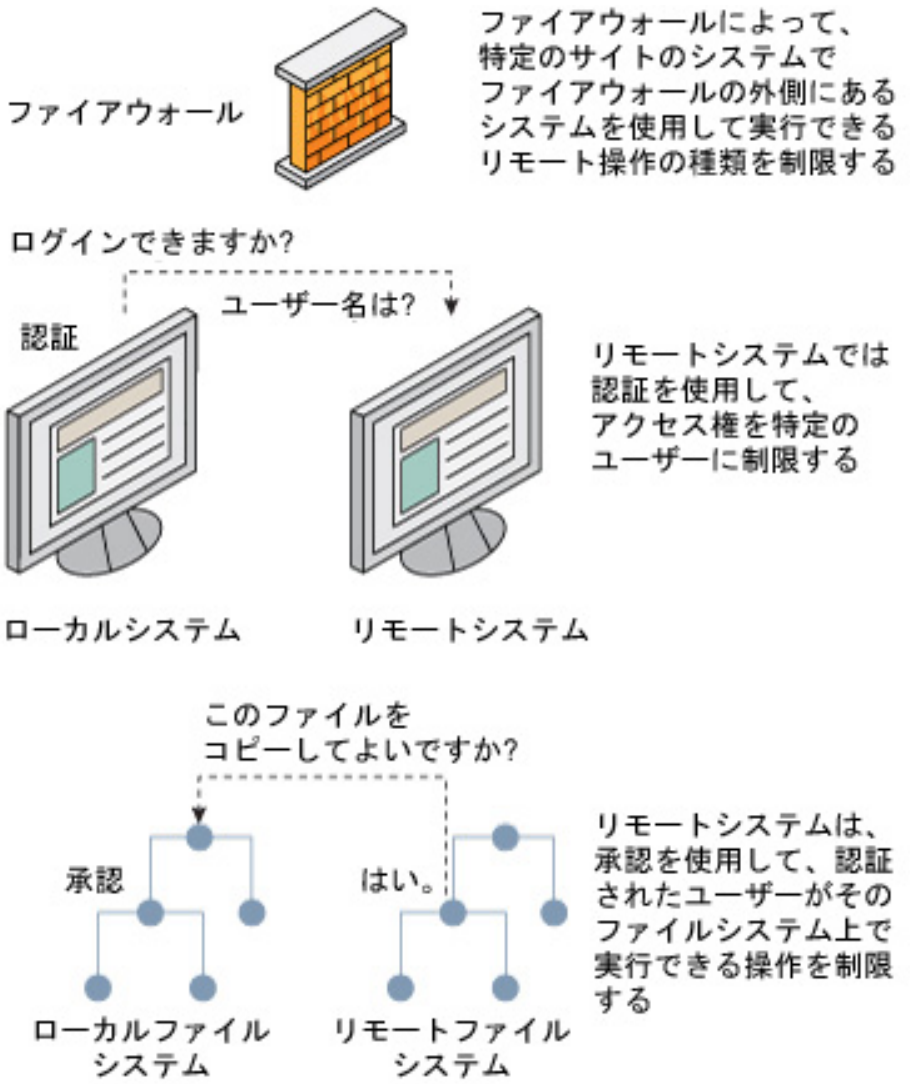
多くの場合、コンピュータは、接続されたコンピュータ間で情報を交換できるコンピュータネットワークの一部になっています。さらに、ネットワークに接続されたコンピュータは、ネットワーク上のほかのコンピュータにあるデータなどのリソースにアクセスできます。コンピュータをネットワーク化するとコンピューティング環境の処理能力と性能が向上しますが、ネットワークのコンピュータセキュリティーが複雑になります。

たとえば、コンピュータのネットワーク内では、個々のシステムは情報を共有できます。未承認アクセスがセキュリティーリスクとなります。多くの人々がネットワークにアクセスするので、(特にユーザーエラーを通して) 未承認アクセスが発生する可能性も大きくなります。また、パスワードの不適切な扱いも未承認アクセスの原因となります。

ネットワークセキュリティーメカニズム

一般にネットワークセキュリティーは、リモートシステムからの操作の制限またはブロックに基づきます。次の図は、リモート操作に適用できるセキュリティー制限を示します。

図 1-1 リモート操作のセキュリティー制限



リモートアクセスの認証と承認

*認証*は、ユーザーがリモートシステムへのアクセスを試みる際にアクセスを制御する方法です。認証は、システムレベルでもネットワークレベルでも設定できます。ユーザーがリモートシステムにアクセスすると、*「承認」*という方法でそのユーザーが実行できる操作が制限されます。次の表は、認証と承認を提供するサービスを示したものです。

表 1-3 リモートアクセスのための認証サービス

サービス	説明	詳細情報
IPsec	IPsec は、ホストに基づく認証および認可に基づく認証と、ネットワークトラフィックの暗号化を行います。	『Oracle Solaris 11.2 でのネットワークのセキュリティー保護 』の第 6 章「IP セキュリティーアーキテクチャーについて」
Kerberos	Kerberos は、システムにログインしているユーザーの認証と承認を暗号化を通して行います。	例については、『Oracle Solaris 11.2 での Kerberos およびその他の認証サービスの管理 』の「Kerberos サービスの動作」を参照してください。
LDAP	LDAP ディレクトリサービスは、認証と承認の両方をネットワークレベルで提供できます。	『Oracle Solaris 11.2 ディレクトリサービスとネームサービスでの作業: DNS と NIS 』
リモートログインコマンド	リモートログインコマンドを使用すると、ユーザーはネットワーク経由でリモートシステムにログインし、そのリソースを使用できます。リモートログインコマンドには `rlogin`、`rcp`、`ftp` などがあります。信頼されるホストの場合、認証は自動です。それ以外の場合は、自分自身を認証するように求められます。	『Oracle Solaris 11.2 でのリモートシステムの管理 』の第 3 章「リモートシステムへのアクセス」
SASL	簡易認証セキュリティー層 (SASL) は、ネットワークプロトコルに認証サービスとセキュリティーサービス (オプション) を提供するフレームワークです。プラグインによって、適切な認証プロトコルを選択できます。	『Oracle Solaris 11.2 での Kerberos およびその他の認証サービスの管理 』の「SASL について」
Secure RPC	Secure RPC を使用すると、リモートマシン上で要求を出したユーザーの認証が行われ、ネットワーク環境のセキュリティーが高まります。Secure RPC には、UNIX、DES、または Kerberos 認証メカニズムのいずれかを使用できます。	『Oracle Solaris 11.2 での Kerberos およびその他の認証サービスの管理 』の「Secure RPC について」
	Secure RPC を使用すると、NFS 環境にセキュリティーを追加できます。Secure RPC を備えた NFS 環境を Secure NFS と呼びます。	『Oracle Solaris 11.2 での Kerberos およびその他の認証サービスの管理 』の「NFS サービスと Secure RPC」

サービス	説明	詳細情報
Secure Shell	Secure Shell は、セキュアでないネットワークを経由したネットワークトラフィックを暗号化します。Secure Shell は、パスワード、公開鍵、またはこの両方の使用による認証を提供します。	『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の「Secure Shell (概要)」

Secure RPC に匹敵する機能として、Oracle Solaris の「*特権ポート*」メカニズムがあります。特権ポートには、1024 未満のポート番号が割り当てられます。クライアントシステムは、クライアントの資格を認証したあと、特権ポートを使用してサーバーへの接続を設定します。次に、サーバーは接続のポート番号を検査してクライアントの資格を検証します。

Oracle Solaris ソフトウェアを使用していないクライアントは、特権ポートを使用して通信できないことがあります。クライアントが特権ポートを使って通信できない場合は、次のようなエラーメッセージが表示されます。

```
“Weak Authentication
NFS request from unprivileged port”
```

ファイアウォールシステム

ファイアウォールシステムを設定すると、ネットワーク内のリソースを外部のアクセスから保護できます。「ファイアウォールシステム」は、内部ネットワークと外部ネットワークの間のバリアとして機能するセキュリティー保護ホストです。内部ネットワークは、ほかのネットワークを「信頼できる状態でない」ものとして扱います。 内部ネットワークと、インターネットなどの外部ネットワークとの間に、このような設定を必ず行うようにしてください。

ファイアウォールはゲートウェイとしても機能しますし、バリアとしても機能します。ゲートウェイとしては、ネットワーク間でデータを通過させます。バリアーとしては、データのネットワークとの間の自由な通過をブロックします。内部ネットワーク上のユーザーがリモートネットワーク上のホストにアクセスするには、ファイアウォールシステムにログインする必要があります。また、外部ネットワーク上のユーザーは、内部ネットワーク上のホストにアクセスする前に、まずファイアウォールシステムにログインしなければなりません。

ファイアウォールは、一部の内部ネットワーク間でも有効です。たとえば、パケットの転送をアドレスまたはプロトコルで制限するには、ファイアウォールまたはセキュアなゲートウェイコンピュータを設定できます。これにより、メールを転送するためのパケットを許可するが、ftp コマンドのパケットは許可しないようにできます。

さらに、内部ネットワークから送信されるすべての電子メールは、まずファイアウォールシステムに送信されます。ファイアウォールは、このメールを外部ネットワーク上のホストに転送します。ファイアウォールシステムは、すべての着信電子メールを受信して内部ネットワーク上のホストに配信するという役割も果たします。

注意 - ファイアウォール上で厳格かつ強固に適用されたセキュリティーを維持している場合でも、ネットワーク上のその他のホストでセキュリティーを緩くすれば、ファイアウォールシステムを突破できる侵入者は、内部ネットワーク上のその他のすべてのホストへのアクセスを取得できる可能性があります。

ファイアウォールシステムには、信頼されるホストを配置しないでください。「*信頼されるホスト*」とは、ユーザーがログインするときに、パスワードを入力する必要がないホストのことです。ファイアウォールシステムでは、ファイルシステムを共有しないでください。また、ほかのサーバーのファイルシステムをマウントしないでください。

Oracle Solaris の IPsec および IP フィルタ機能は、ファイアウォール保護を提供できます。ネットワークトラフィックの保護の詳細は、『Oracle Solaris 11.2 でのネットワークのセキュリティー保護 』を参照してください。

暗号化システムとファイアウォールシステム

ネットワーク外部からの未承認ユーザーは、宛先に到達する前にパケットを捕捉し、元の経路にパケットを戻す前に任意のデータを内容に挿入することで、パケット内のデータを破損させたり、破棄したりできます。この方法は、「*パケットスマッシング*」と呼ばれます。

ローカルエリアネットワーク上では、パケットはサーバーを含むすべてのシステムに同時に到達するので、パケットスマッシングは不可能です。ただし、ゲートウェイ上ではパケットスマッシングが可能なため、ネットワーク上のすべてのゲートウェイを保護する必要があります。

もっとも危険なのは、データの完全性に影響するような攻撃です。このような攻撃を受けると、パケットの内容が変更されたり、ユーザーが偽装されたりします。

その他の攻撃でも盗聴が伴う可能性がありますが、データの整合性が損なわれたり、ユーザーが偽装されたりすることはありません。盗聴者は、会話を記録して、あとで再生します。盗聴攻撃によってデータの完全性が損なわれることはありませんが、プライバシが侵害されます。ネットワーク上でやりとりされるデータを暗号化すると、重要な情報のプライバシを保護できます。

- セキュリティー保護されていないネットワーク経由のリモート操作を暗号化する方法については、『Oracle Solaris 11.2 での Secure Shell アクセスの管理 』の第 1 章「Secure Shell の使用 (タスク)」を参照してください。
- ネットワーク内のデータを暗号化および認証する方法については、『Oracle Solaris 11.2 での Kerberos およびその他の認証サービスの管理 』の第 2 章「Kerberos サービスについて」を参照してください。
- IP データグラムを暗号化する方法については、『Oracle Solaris 11.2 でのネットワークのセキュリティー保護 』の第 6 章「IP セキュリティーアーキテクチャーについて」を参照してください。

セキュリティー問題の報告

会社で重大なセキュリティー違反が発生した疑いがある場合は、Computer Emergency Response Team/Coordination Center (CERT/CC) に連絡してください。CERT/CC は、Defense Advanced Research Projects Agency (DARPA) の資金提供を受けたプロジェクトで、カーネギメロン大学の Software Engineering Institute にあります。CERT/CC はセキュリティー問題の解決を支援できます。また、特定のニーズに合った他の Computer Emergency Response Team を紹介することもできます。最新の連絡先情報については、CERT/CC (http://www.cert.org/contact_cert/) Web サイトを参照してください。

♦ ♦ ♦ 第 2 章

2 Oracle Solaris システムの整合性の保護

未承認のカーネルモジュール、トロイの木馬アプリケーション、およびシステムにロードされるその他の脅威から Oracle Solaris システムを保護できます。この章では、このような脅威からの保護を提供し、システム全体の整合性を保持する Oracle Solaris のセキュリティー機能について説明します。この章の内容は次のとおりです。

ベリファイドブートの使用

Oracle Solaris のベリファイドブートによって、システムのブートプロセスがセキュリティー保護されます。この機能は、次のような脅威からシステムを保護します。

- カーネルモジュールの破損
- 正当なカーネルモジュールになりすました悪意のあるプログラム (トロイの木馬ウイルス、スパイウェア、ルートキットなど) の挿入または置換
- 未承認のサードパーティーカーネルモジュールのインストール

悪質のあるプログラムは、サードパーティーに情報を渡したり、Oracle Solaris の動作を変更したりする可能性があります。一般に、サードパーティーモジュールに悪意がなくても、サイトの変更を制御するポリシーに違反している可能性があります。したがって、このようなモジュールが承認なしでインストールされることからシステムを保護する必要もあります。

ベリファイドブートと ELF 署名

Oracle Solaris では、ブート検証は `elfsign` の署名または鍵を使用して実行されます。Oracle Solaris カーネルモジュールは、工場でこれらの鍵を使用して署名されます。ファイル形式から、これらのモジュールは ELF オブジェクトとも呼ばれます。署名は、オブジェクトファイルで選択した ELF レコードの SHA-1 または SHA-256 チェックサムを使用して作成されます。SHA-1 または SHA-256 チェックサムは、RSA-2048 の非公開鍵と公開鍵のペアを使用して署名されます。公開鍵は `/etc/certs` で配布されていますが、非公開鍵は配布されていません。

すべての鍵は、システムの**ブート前環境**に格納されています。これは、Oracle Solaris をブートする前に実行されるソフトウェアまたはファームウェアです。このファームウェアは、`platform/.../unix` をロードおよびブートします。

ブート前環境は、システムのカテゴリごとに異なります。カテゴリごとにサポートされているブート前環境は、次のとおりです。

- レガシー SPARC システムおよび x86 システム - これらのシステムには、ファイルシステム外部に格納する機能が備わっていないため、ブート検証の構成設定はファイルシステム自体に格納されます。具体的には、構成情報は `/etc/system` に格納されます。鍵は、ルートファイルシステムおよびブートアーカイブの `/etc/certs/*SE` に格納されます。
- Oracle Integrated Lights Out Manager (ILOM) のベリファイドブートがサポートされている SPARC システム - 鍵および構成設定は Oracle ILOM に格納されます。

  Oracle ILOM はオペレーティングシステムのファイルシステム外部にあるため、ベリファイドブートの構成は、オペレーティングシステムのユーザー (管理者 (root) 特権を持つユーザーを含む) による改ざんから保護されます。したがって、このシステムカテゴリでは、ベリファイドブートがよりセキュアです。

  ベリファイドブートの構成が承認なしで変更されることを回避するには、Oracle ILOM へのアクセスがセキュアであることを確認する必要があります。Oracle ILOM のセキュリティー保護の詳細は、http://www.oracle.com/goto/ILOM/docs にあるドキュメントを参照してください。
- SPARC M5 シリーズ、SPARC M6 シリーズ、および SPARC T5 シリーズ - 構成設定はシステムの Oracle ILOM に格納されます。SPARC ファームウェアが構成情報を Oracle Solaris に送信します。

システムブート時の検証シーケンス

ベリファイドブートによって、Oracle Solaris カーネルモジュールの `elfsign` 署名の検証が自動化されます。管理者はベリファイドブートを使用することで、システムのリセットからブートプロセスの完了までのブートプロセスに、検証可能な信頼チェーンを作成できます。

システムのブート中に、ブートプロセスで開始されたコードの各ブロックで、次にロードする必要があるブロックが検証されます。検証およびロードのシーケンスは、最後のカーネルモジュールがロードされるまで続行されます。

あとでシステムで電源の再投入が実行されるときに、新しい検証シーケンスが開始されます。管理者は、検証に失敗したときに適切なアクションが行われるように、ベリファイドブートを構成することもできます。

SPARC での Oracle Solaris のブートフローを検討します。

`Firmware -> Bootblock -> /platform/.../unix -> genunix ->` *other kernel modules*

SPARC ファームウェアは工場でインストールされます。`fwupdate` ユーティリティーを使用することで、ファームウェアのデジタル署名を更新することもできます。このファームウェアは、初期の Oracle Solaris モジュールである Oracle Solaris の `/platform/.../unix` モジュールを検証してから、ロードします。同様に、モジュールの一部である Oracle Solaris カーネルの実行時ローダー `krtld` は、汎用の UNIX (`genunix`) モジュールおよび後続のモジュールを検証し、ロードします。

ベリファイドブートのポリシー

ベリファイドブートは、次の 2 つのポリシーで管理されます。

- UNIX および `genunix` モジュールの検証は、ブートポリシーで規定されます。ブートプロセス時に、これらのモジュールが最初にロードされます。
- `genunix` のあとにロードする必要があるその他のカーネルモジュールの検証は、モジュールポリシーで規定されます。

レガシー SPARC システムおよび x86 システムでは、ポリシーは `/etc/system` ファイルの `boot_policy` および `module_policy` 変数で定義されます。Oracle ILOM のベリファイドブートがサポートされている SPARC システムでは、Oracle ILOM の `boot_policy` および

module_policy プロパティーは /HOST*x*/verified_boot にあります。ここで、*x* は物理ドメイン (PDomain) 番号です。
変数またはプロパティーは両方とも、次の値のいずれかを使用して構成できます。

- none - ブート検証が実行されません。デフォルトでは、boot_policy と module_policy が両方とも構成されていないため、ベリファイドブートは無効になっています。
- warning - モジュールがロードされる前に、各カーネルモジュールの elfsign 署名が検証されます。モジュールの検証に失敗した場合でも、モジュールはロードされます。不一致は、システムコンソールまたはシステムログ (使用可能な場合) に記録されます。デフォルトのログは /var/adm/messages です。
- enforce - モジュールがロードされる前に、各カーネルモジュールの elfsign 署名が検証されます。モジュールの検証に失敗した場合は、モジュールがロードされません。不一致は、システムコンソールまたはシステムログ (使用可能な場合) に記録されます。デフォルトのログは /var/adm/messages です。

ポリシーを構成することに加えて、システムで elfsign X.509 公開鍵証明書を指定することもできます。モジュールと同様に、変数を使用するか、Oracle ILOM プロパティーを定義することで証明書を指定します。

ベリファイドブートがサポートされている Oracle ILOM が組み込まれたシステムでは、事前にインストールされたベリファイドブートの証明書ファイル /etc/certs/ORCLS11SE が Oracle ILOM の一部として提供されています。レガシー SPARC システムおよび x86 システムでは、証明書は Oracle Solaris の /etc/certs/ORCLS11SE ファイルとして使用できます。

証明書には、ELF オブジェクトの elfsign 署名を検証する際に使用される RSA 公開鍵が含まれています。ただし、企業で提供された証明書をインストールして /etc/certs/ORCLS11SE を置き換えることもできます。すべての証明書は個別の PDomain にロードされ、管理されます。

ベリファイドブートの有効化

デフォルトでは、システムでベリファイドブートが無効になっています。この機能を有効にする手順は、システムによって異なります。この機能を有効にするには、使用中のシステムに対応する手順 (このセクションで示す) を使用します。

▼ SPARC: Oracle ILOM のベリファイドブートがサポートされている SPARC システムでベリファイドブートを有効にする方法

Oracle ILOM のベリファイドブートがサポートされている SPARC システムでは、ベリファイドブートのプロパティーは `/HOSTx/verified_boot` にあります。ここで、*x* は物理ドメイン (PDomain) 番号 (`HOST0` や `HOST1`など) です。

注記 - 一部の SPARC システムには、1 つの物理ドメイン `/HOST` のみが存在しますが、その他の SPARC システムには複数の物理ドメインが存在します。この手順では、複数の物理ドメインを持つシステムが使用されていて、物理ドメインが `/HOSTx` と呼ばれると仮定します。システムに固有のセキュリティー機能については、システムのセキュリティーマニュアルを参照してください。

---

1. **(オプション) ベリファイドブートがシステムでサポートされているかどうかを確認します。**

```
# show /HOSTx/verified_boot
show: Invalid target /HOST/verified_boot
```

`fwupdate` を使用すると、システムの Oracle ILOM ファームウェアを更新できます。

2. **管理者として、Oracle ILOM ユーザーインタフェースにログインします。**

```
% ssh root@ilom
```

ここで、*ilom* には、Oracle ILOM サービスプロセッサの IP アドレス、またはシャーシモニタリングモジュールの IP アドレスを指定できます。

3. **ベリファイドブートのプロパティーを構成します。**

```
--> set /HOSTx/verified_boot/boot_policy=warning
--> set /HOSTx/verified_boot/module_policy=warning
```

注記 - プロパティーごとに「`warning`」または「`enforce`」を指定します。プロパティーには異なる構成を指定できます。これらのポリシーの構成については、35 ページの「ベリファイドブートのポリシー」を参照してください。

ブートポリシーが `enforce` を使用して構成されている場合に、UNIX または `genunix` モジュールで不一致が検出されると、システムがブートしません。その代わりに、システムは OpenBoot PROM (OBP) に戻ります。

---

4. **システムで提供されている証明書の代わりに使用する証明書を指定します。**

```
--> load /HOSTx/verified_boot/cert -source ftp-location
```

ここで、*ftp-location* は、証明書が格納される FTP サーバーとファイル名を表します。*ftp-location* は、URL 形式 (`ftp://`*server/filename*) で指定する必要があります。

5. **(オプション) ベリファイドブートの構成を表示します。**

```
--> show /HOSTx/verified_boot
/HOST0
Properties:
boot_policy = warning
module_policy = warning
cert = ftp://server/filename
```

▼ レガシー SPARC システムまたは x86 システムでベリファイドブートを有効にする方法

この手順は、システムのローカルファイルシステム外部にブート検証の構成を格納する方法がシステムに備わっていない場合に使用します。

このタイプのシステムでブート検証を有効にする際は、次のセキュリティーの考慮事項に注意してください。

- 構成情報はローカルファイルシステムに格納されるため、アクセス可能です。
- 任意の特権ユーザーが構成を変更できます。
- ポリシーの設定を変更でき、ブート検証自体を無効にできます。
- 任意の `elfsign` 署名者がオブジェクトモジュールを署名できる可能性のある追加の鍵を追加できます。

1. **`/etc/system` ファイルを編集します。**

   a. **`boot_policy` および `module_policy` 変数を追加し、構成します。**

   たとえば、`/etc/system` では、次のように入力します (太字で表示)。

```
* Verified Boot settings: 1=none (default), 2=warning, 3=enforce
set boot_policy=2
set module_policy=2
```

変数ごとに行う構成に対応する番号を指定します。変数には異なる構成を指定できます。これらのポリシーの構成については、35 ページの「ベリファイドブートのポリシー」を参照してください。

ブートポリシーが `enforce` を使用して構成されている場合に、UNIX または `genunix` モジュールで不一致が検出されると、システムがブートしません。その代わりに、システムは OpenBoot PROM (OBP) に戻ります。

b. **`verified_boot_certs` 変数に、1 つ以上の `elfsign` X.509 鍵証明書を指定します。**

```
set verified_boot_certs="/etc/certs/THIRDPARTYSE"
```

ここで、*THIRDPARTY* はユーザーが指定した証明書ファイルの名前です。

2. **ブートアーカイブ内の `/etc/system` ファイルを更新します。**

```
# bootadm update-archive
```

3. **(オプション) ベリファイドブートの構成を表示します。**

a. **アーカイブをマウントします。**

- SPARC システムの場合:

```
# mount -r -F hsfs /platform/sun4v/boot_archive /mnt
```

- x86 システムの場合:

```
# mount -r -F hsfs /platform/x86-type/boot_archive /mnt
```

ここで、*x86-type* は `i86pc` または `amd64` です。

b. **ベリファイドブートの構成および `elfsign` 鍵を表示します。**

```
# gzcat /mnt/etc/system | egrep 'verified|policy'
# ls -l /etc/certs
```

▼ Oracle ILOM のベリファイドブートがサポートされているシステムで証明書を管理する方法

この手順では、システムのベリファイドブート証明書を管理する方法について説明します。

1. **事前にインストールされている証明書をユーザーが指定した証明書に置き換えるには、次のように入力します。**

   --> **load /HOST***x*/**verified_boot/cert -source ftp://***server/filename*

2. **現在の証明書のコピーをユーザーが指定したソースに保存するには、次のように入力します。**

   --> **dump /HOST***x*/**verified_boot/cert -dest ftp://***server/filename*

3. **ユーザーがインストールした証明書を削除し、システムに事前にインストールされている証明書に戻すには、次のように入力します。**

   --> **reset /HOST***x*/**verified_boot/cert**

▼ elfsign 署名を手動で検証する方法

ベリファイドブートは、ブートプロセスの整合性を確認するための迅速かつ効率的な方法を提供する自動メカニズムです。ただし、カーネルモジュールの署名を手動で検証することもできます。

- **次のように、elfsign コマンド構文を使用します。**

  $ **elfsign verify -v** *kernel_module*

  例:

```
$ elfsign verify -v /kernel/misc/sparcv9/cardbus
elfsign: verification of /kernel/misc/sparcv9/cardbus passed.
format: rsa_sha1.
signer: O=Oracle Corporation, OU=Corporate Object Signing, \
               OU=Solaris Signed Execution, CN=Solaris 11
```

Trusted Platform Module について

Trusted Platform Module (TPM) は、システムに固有の暗号化済み構成情報が格納されるデバイスおよび実装を表します。この情報は、システムブート時にプロセスを測定するメトリックとして機能します。Oracle Solaris では、暗号化鍵をセキュアに格納するために TPM が使用されます。

Oracle Solaris では、次のコンポーネントに TPM が実装されています。

- TPM デバイスドライバは TPM デバイスと通信します。

- Trusted Computing Group (TCG) Software Stack (TSS) は、`tcsd` デーモンを使用した TPM デバイスとの通信チャネルとして機能します。
- PKCS #11 ライブラリには、TPM を使用して鍵を生成し、機密操作を実行するハードウェアトークンまたはプロバイダが実装されています。プロバイダでは、TPM デバイス内部でのみ使用可能な鍵で暗号化することで、すべてのプライベートデータオブジェクトが保護されます。PKCS #11 ライブラリは、RSA Security Inc. PKCS #11 Cryptographic Token Interface (Cryptoki) 標準に準拠しています。
- ブートプロセスの検証で TPM 関連の側面を管理するために、`tpmadm` コマンドが使用されます。

  詳細は、`tpmadm`(1M) のマニュアルページを参照してください。

プラットフォーム所有者は、特権操作を承認する際に使用される所有者のパスワードを設定することで、TPM を初期化する必要があります。プラットフォーム所有者は TPM 所有者とも呼ばれます。従来のスーパーユーザーと異なる点は、次の 2 つです。

- TPM 機能にアクセスするために、プロセス特権は必要ありません。呼び出し元プロセスの特権レベルに関係なく、特権操作では所有者のパスワードを把握しておくことが必要です。
- TPM 所有者は、TPM 鍵で保護されたデータのアクセス制御をオーバーライドできません。所有者は TPM を再初期化することで、データを効率的に破棄できます。ただし、所有者は、その他のユーザーが所有する TPM 鍵で暗号化されたデータにはアクセスできません。

このガイドで説明したその他の方法とともに Trusted Platform Module を使用すると、ユーザーまたはアプリケーションによる未承認アクセスからシステムがセキュリティー保護されます。

Oracle Solaris システムでの TPM の初期化

このセクションでは、Oracle Solaris システムで TPM を初期化する手順について説明します。SPARC システムと x86 システムとでは、手順が異なります。ただし、TPM を初期化するための特定の前提条件は、両方のプラットフォームで共通です。

- システムに TPM デバイス `/dev/tpm` をインストールする必要があります。
- TPM では TCG Trusted Platform Module 仕様バージョン 1.2 (別名 ISO/IEC 11889-1:2009) が使用されている必要があります。http://www.trustedcomputinggroup.org/resources/tpm_main_specification で公開されている仕様を参照してください。

- 次の Oracle Solaris TPM パッケージがインストールされている必要があります。
  - Trusted Platform Module ドライバ (`driver/crypto/TPM`)
  - TrouSerS TCG ソフトウェア (`library/security/trousers`)

  これらのパッケージをインストールするには、次のコマンドを使用します。

```
# pkg install driver/crypto/tpm
# pkg install library/security/trousers
```

▼ TPM デバイスがオペレーティングシステムで認識されているかどうかを確認する方法

この手順を使用して、インストールされている TPM デバイスが Oracle Solaris で認識されているかどうかを確認します。この手順は、SPARC システムと x86 システムの両方に適用されます。

- **端末ウィンドウで、次のコマンドを発行します。**

```
# prtconf -v |grep tpm
```

TPM デバイスが認識されている場合は、コマンドで次のような出力が生成されます。

```
# prtconf -v |grep tpm
tpm, instance #0
dev_path=/pci@0,0/isa@lf/tpm@0,fed40000:tpm
dev_link=/dev/tpm
```

出力が生成されない場合は、デバイスが無効になっている可能性があります。デバイスを有効にする方法については、システムのプラットフォームに応じて、43 ページの「Oracle ILOM インタフェースを使用して TPM を初期化する方法」または44 ページの「BIOS を使用して TPM を初期化する方法」を参照してください。

注記 - 代わりに `ls` コマンドを使用しても、同じ情報を取得できます。ただし、この出力に含まれる情報は、`prtconf` 構文で提供される情報よりも少ないです。

```
# ls -l /dev/tpm
lrwxrwxrwx  1 root root  44 May 22 2012 /dev/tpm ->
../devices/pci@0,0/isa@lf/tpm@0,fed40000:tpm
```

▼ SPARC: Oracle ILOM インタフェースを使用して TPM を初期化する方法

SPARC システムで TPM を初期化するには、システムの Oracle ILOM と Oracle Solaris の両方のインタフェースを使用します。

1. **Oracle ILOM プロンプトで、システムの電源をオフにします。**

```
-> stop -f/SYS
```

2. **TPM をアクティブにします。**

SPARC システムに応じて、次のコマンドセットのいずれかを使用して TPM をアクティブにします。

- SPARC M5 シリーズサーバーまたは SPARC T5 シリーズサーバーでは、次のコマンドを使用します。

```
-> set /HOST/tpm mode=activated
```

- SPARC M5-32 シリーズサーバーでは、次のコマンドを使用します。

```
-> set /HOST#/tpm mode=activated
```

ここで、# はインスタンス番号 (たとえば、HOST0/tpm) です。

- SPARC T4 サーバーでは、次のコマンドを使用します。

```
-> set /HOST/tpm enable=true activate=true
-> show /HOST/tpm
```

3. **Oracle Solaris プロンプトで、TPM を初期化します。**

TPM を初期化すると TPM 所有者となり、所有者パスワード (所有者 PIN とも呼ばれる) を割り当てる必要があります。

```
 # tpmadm init
TPM Owner PIN:
Confirm TPM Owner PIN
```

4. **TPM のステータスを確認します。**

```
 # tpmadm status
TPM Version: 1.2 (ATML Rev: 13.9, SpecLevel: 2, ErrataRev: 1)
TPM resources
Contexts: 16/16 available
```

```
Sessions: 2/3 available
Auth Sessions: 2/3 available
Loaded Keys: 18/21 available
Platform Configuration Registers (24)
PCR 0: E1 EE 40 D8 66 28 A9 08 B6 22 8E AF DC 3C BC 23 71 15 49 31
PCR 1: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 2: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 3: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 4: AF 98 77 B8 72 82 94 7D BE 09 25 10 2E 60 F9 60 80 1E E6 7C
PCR 5: E1 AA 8C DF 53 A4 23 BF DB 2F 4F 0F F2 90 A5 45 21 D8 BF 27
PCR 6: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 7: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
PCR 8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 9: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 11: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 13: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 15: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 17: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 18: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 19: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 20: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 21: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 22: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 23: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

5. **(オプション) TPM 暗号化プロバイダを有効にします。**

注記 - TPM 暗号化プロバイダは、Oracle Solaris よりも低速です。この手順は、TPM で暗号化操作を実行する場合にのみ実行してください。

```
# cryptoadm install provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
# cryptoadm list -mv provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
```

▼ x86: BIOS を使用して TPM を初期化する方法

x86 システムでは、Oracle Solaris を使用してサービスを初期化する前に、システムの BIOS で次の手順を実行します。

1. **Oracle Solaris プロンプトで、システムをリブートします。**

```
# reboot -p
```

2. **システムのブート中に F2 キーを押して、BIOS メニューにアクセスします。**

3. **BIOS メニューオプションを使用して、TPM を構成します。**

   a. **「Advanced」 -> 「Trusted Computing」に移動します。**

   b. **次のメニュー項目の値を指定することで、TPM を設定します。**

   ```
   TCG/TPM Support [Yes]
   Execute TPM Command [Enabled]
   ```

   c. **Esc キーを押して、BIOS メニューを終了します。**

   d. **「Save Changes and Exit」を選択します。**

   e. **ブートプロセスを続行するには、「Ok」をクリックします。**

4. **ブートプロセスが完了したら、`tcsd` デーモンを有効にします。**

   ```
   # svcadm enable -s svc:/application/security/tcsd
   ```

5. **TPM を初期化します。**

   TPM を初期化すると TPM 所有者となり、所有者パスワードを割り当てる必要があります。

   ```
   # tpmadm init
   TPM Owner PIN:
   Confirm TPM Owner PIN
   ```

6. **TPM のステータスを確認します。**

   ```
    # tpmadm status
   TPM Version: 1.2 (ATML Rev: 13.9, SpecLevel: 2, ErrataRev: 1)
   TPM resources
   Contexts: 16/16 available
   Sessions: 2/3 available
   Auth Sessions: 2/3 available
   Loaded Keys: 18/21 available
   Platform Configuration Registers (24)
   PCR 0: E1 EE 40 D8 66 28 A9 08 B6 22 8E AF DC 3C BC 23 71 15 49 31
   PCR 1: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 2: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 3: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 4: AF 98 77 B8 72 82 94 7D BE 09 25 10 2E 60 F9 60 80 1E E6 7C
   PCR 5: E1 AA 8C DF 53 A4 23 BF DB 2F 4F 0F F2 90 A5 45 21 D8 BF 27
   PCR 6: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 7: 5B 93 BB A0 A6 64 A7 10 52 59 4A 70 95 B2 07 75 77 03 45 0B
   PCR 8: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 9: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   PCR 11: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
   ```

```
PCR 12: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 13: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 14: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 15: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 16: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
PCR 17: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 18: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 19: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 20: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 21: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 22: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
PCR 23: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

7. **(オプション) TPM 暗号化プロバイダを有効にします。**

注記 - TPM 暗号化プロバイダは、Oracle Solaris よりも低速です。したがって、この手順は、TPM で暗号化操作を実行する場合にのみ実行してください。

```
# cryptoadm install provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
# cryptoadm list -mv provider='/usr/lib/security/$ISA/pkcs11_tpm.so'
```

▼ セキュアなキーストアとして TPM を使用するために PKCS #11 コンシューマを有効にする方法

始める前に この手順を実行するには、システムに TPM をインストールし、有効にする必要があります。`tcsd` デーモンも動作していることを確認します。

1. **TPM デバイスがインストールされていることを確認します。**

```
# ls -alF /dev/tpm
lrwxrwxrwx 1 root 39 Dec 27 2011 /dev/tpm -> ../devices/pci@0,0/isa@1/tpm@1,1670:tpm
```

2. **`tcsd` デーモンを有効にします。**

```
# svcadm enable tcsd
```

3. **個人用の TPM で保護されたトークンの格納領域を初期化します。**

```
$ pktool inittoken currlabel=TPM
```

注記 - この手順は、個々のユーザーが実行する必要があります。

4. **セキュリティー責任者のトークン PIN を設定します。**

```
$ pktool setpin token=tmp/TPM so
```

5. **ユーザーの PIN を設定します。**

```
$ pktool setpin token=tmp/TPM
```

6. **トークンを初期化するときに使用したトークン名を指定することで、TPM デバイスを使用する鍵と証明書を生成します。**

```
$ pktool gencert token=tpm/TPM -i
$ pktool list token=tpm/TPM
```

既存のアプリケーションで `libpkcs11` の暗号化フレームワークがすでに使用されている場合は、アプリケーションでセッション用の TPM トークンデバイスを選択することで、それらの操作で TPM トークンを使用できます。

例 2-1 TPM を使用するための PKCS #11 コンシューマの有効化

この例では、最初に TPM トークンに新しい名前が割り当てられます。その後、トークン上のすべての後続アクションで、この新しい名前が参照されます。

```
$ pktool inittoken currlable=TPM newlabel=JohnDoeTPM
$ pktool setpin token=tmp/JohnDoeTPM so
$ pktool gencert token=tpm/JohnDoeTPM -i
$ pktool list token=tpm/JohnDoeTPM
```

TPM のトラブルシューティング

このセクションで説明するコマンドを使用して、正常な TPM の使用を可能にするさまざまな動作コンポーネントをモニターし、TPM の問題のトラブルシューティングを行います。

- `tcsd` デーモンが動作していることを確認するには:

```
# svcs tcsd
STATE      STIME      FMRI
online     Nov_07     svc:/application/security/tcsd:default
```

- TPM デバイスがインストールされていることを確認するには:

```
# ls -alF /dev/tpm
lrwxrwxrwx 1 root 39 Dec 27 2011 /dev/tpm -> ../devices/pci@0,0/isa@1/tpm@1,1670:tpm
```

- TSS ソフトウェアパッケージがインストールされていることを確認するには:

```
# pkg info trousers
```

```
Name: library/security/trousers
Summary: TrouSerS TCG software to access a TPM device
Description: The TrouSerS library provides a software stack from the
Trusted Computer Group (TCG) that accesses a Trusted Platform Module
(TPM) hardware device.
Category: System/Security
State: Installed
Publisher: solaris
Version: 0.3.6
Build Release: 5.11
Branch: 0.175.1.0.0.24.0
Packaging Date: September 4, 2012 05:28:21 PM
Size: 3.65 MB
FMRI: pkg://solaris/library/security/
trousers@0.3.6,5.11-0.175.1.0.0.24.0:20120904T1728212
```

- 以前に TPM が再インストールされたあとの要件として、TPM をクリアするには:
  - Oracle Solaris プロンプトで:

    ```
    # tpmadm clear owner
    ```

  - Oracle ILOM プロンプトで:

    ```
    -> stop /SYS
    -> set /HOST/tpm forceclear=true
    -> start /SYS
    ```

♦ ♦ ♦ 第 3 章

3 システムアクセスの制御

この章では、Oracle Solaris システムにアクセスできるユーザーを制御する方法について説明します。

この章の内容は次のとおりです。

システムセキュリティーの概要については、第1章「マシンセキュリティーの管理」を参照してください。

ログインとパスワードのセキュリティー

システムへのアクセスを保護するために、リモートログインを制限したり、ユーザーにパスワードを持つように要求したり、root アカウントに複雑なパスワードを設定するように要求したりできます。ユーザーアクセスを管理するために、ユーザーにセキュリティーメッセージを表示したり、失敗したアクセス試行をモニターしたり、ログインを一時的に無効にしたりできます。

次のタスクマップは、ユーザーログインをモニターする手順と、ユーザーログインを無効にする手順を示しています。

表 3-1 ログインとパスワードの保護タスクマップ

タスク	説明	手順
ログイン時に、ユーザーにサイトセキュリティーを通知します。	ログイン画面に、サイトセキュリティー情報を含むテキストメッセージを表示します。	『Oracle Solaris 11 セキュリティーガイドライン 』の「バナーファイルにセキュリティーメッセージを配置する方法」

タスク	説明	手順
		『Oracle Solaris 11 セキュリティーガイドライン 』の「セキュリティーメッセージをデスクトップログイン画面に配置する方法」
ユーザーのログインステータスを表示します。	ユーザーのログインアカウントについての広範な情報 (フルネーム、パスワードの有効期限など) を一覧表示します。	50 ページの「ユーザーのログインステータスを表示する方法」
パスワードを所有していないユーザーを発見します。	パスワードを必要としないアカウントを持つユーザーだけを検出します。	51 ページの「パスワードを持たないユーザーを表示する方法」
ログインを一時的に無効にします。	システムシャットダウンや定常的な保守の中でマシンへのユーザーログインを拒否します。	52 ページの「ユーザーのログインを一時的に無効にする方法」

▼ ユーザーのログインステータスを表示する方法

始める前に `logins` コマンドを使用するには、User Management または User Security 権利プロファイルが割り当てられている管理者になる必要があります。デフォルトでは、`root` 役割がこの承認を持っています。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **`logins` コマンドを使用してユーザーのログインステータスを表示します。**

  `# logins -x -l` *username*

  -x　　ログインステータス情報の拡張セットを表示します。

  -l *username*　　指定するユーザーのログインステータスを表示します。変数 *username* はユーザーのログイン名です。複数のログイン名はコンマで区切ります。

  `logins` コマンドは、適切なパスワードデータベースを使ってユーザーのログインステータスを表示します。このデータベースは、ローカルの `/etc/passwd` ファイルか、ネームサービスのパスワードデータベースです。詳細は、`logins`(1M) のマニュアルページを参照してください。

例 3-1　ユーザーのログインステータスを表示する

次の例では、ユーザー `jdoe` のログインステータスが表示されます。

```
# logins -x -l jdoe
jdoe       500     staff          10   Jaylee Jaye Doe
```

```
/home/jdoe
/bin/bash
PS 010103 10 7 -1
```

jdoe	ユーザーのログイン名を示します。
500	ユーザー ID (UID) を示します。
staff	ユーザーのプライマリグループを示します。
10	グループ ID (GID) を示します。
Jaylee Jaye Doe	コメントを示します。
/home/jdoe	ユーザーのホームディレクトリを示します。
/bin/bash	ログインシェルを示します。
PS 010170 10 7 -1	次のパスワード有効期限情報を示します。 ■ パスワードの最終変更日 ■ 次に変更するまでに必要な日数 ■ 変更しないで使用できる日数 ■ 警告期間

▼ パスワードを持たないユーザーを表示する方法

始める前に logins コマンドを使用するには、User Management または User Security 権利プロファイルが割り当てられている管理者になる必要があります。デフォルトでは、root 役割がこの承認を持っています。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **logins コマンドを使用して、パスワードを持っていないユーザーをすべて表示します。**

```
# logins -p
```

-p オプションを指定すると、パスワードを持たないユーザーが一覧表示されます。logins コマンドは、system/name-service/switch サービスの password プロパティーで分散ネームサービスが指定されていないかぎり、ローカルシステムの passwd データベースを使用します。

例 3-2 パスワードを持たないアカウントの表示

次の例では、ユーザー pmorph と役割 roletop はパスワードを持っていません。

```
# logins -p
pmorph          501     other           1       Polly Morph
roletop         211     admin           1       Role Top
#
```

▼ ユーザーのログインを一時的に無効にする方法

システムシャットダウンや定常的な保守の際にユーザーのログインを一時的に無効にします。

注記 - この手順によって、すべてのユーザーが影響を受けるわけではありません。この手順で作成された `/etc/nologin` ファイルが存在していても、次のユーザーは引き続きシステムにログインできます。

- スーパーユーザー
- `root` 役割が割り当てられているユーザー
- `solaris.system.maintenance` 承認が割り当てられているユーザー

詳細は、nologin(4) のマニュアルページを参照してください。

始める前に `solaris.admin.edit/etc/nologin` 承認が割り当てられている管理者になる必要があります。デフォルトでは、`root` 役割がこの承認を持っています。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **テキストエディタで、`/etc/nologin` ファイルを作成します。**

   ```
   # pfedit /etc/nologin
   ```

   `solaris.admin.edit/etc/nologin` 承認の使用の例については、例3-3「ユーザーログインを無効にする」を参照してください。

2. **システムの利用に関するメッセージを入力します。**

3. **ファイルを閉じて、保存します。**

例 3-3 ユーザーログインを無効にする

この例では、ユーザーが、システム使用不可の通知の書き込みを承認されます。

```
% pfedit /etc/nologin
***No logins permitted.***
```

```
***The system will be unavailable until 12 noon.***
```

パスワード暗号化のデフォルトアルゴリズムを変更する

パスワードを暗号化するために別のアルゴリズムを使用するには、/etc/security/policy.conf ファイルを編集します。デフォルトでは、ユーザーパスワードは crypt_sha256 アルゴリズムで暗号化されます。アルゴリズムは、ファイルの CRYPT_DEFAULT パラメータに割り当てられた識別子 5 で表されます。別のアルゴリズムに切り替えるには、別の識別子を割り当てます。パスワード暗号化アルゴリズムと対応する識別子のリストについては、表1-1「パスワードの暗号化アルゴリズム」を参照してください。

注記 - 可能な場合は常に、FIPS 承認アルゴリズムを使用してください。FIPS 承認アルゴリズムおよび非承認アルゴリズムのリストについては、『Using a FIPS 140 Enabled System in Oracle Solaris 11.2 』の「FIPS 140 Algorithm Lists and Certificate References for Oracle Solaris Systems」を参照してください。

新しいアルゴリズムは新しいユーザーのパスワード暗号化にのみ適用されます。既存のユーザーの場合、以前のアルゴリズムが CRYPT_ALGORITHMS_ALLOW パラメータに定義されたままで、unix 以外であれば、それが引き続き機能します。この場合に暗号化の実装状態を確認する方法については、14 ページの「policy.conf ファイルのアルゴリズム構成」を確認してください。新しいパスワード暗号化アルゴリズムに既存のユーザーを追加するには、CRYPT_ALGORITHMS_ALLOW パラメータから以前のアルゴリズムを削除してください。

選択したアルゴリズムの構成の詳細については、policy.conf(4) のマニュアルページを参照してください。

▼ パスワード暗号化のアルゴリズムを指定する方法

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **/etc/security/polic.conf ファイルで、選択した暗号化アルゴリズムを表す識別子を CRYPT_DEFAULT 変数の値として指定します。**
2. **(オプション) 選択についての説明をファイルにコメントします。**

例:

```
# cat  /etc/security/policy.conf
…
# Sets the SHA256 (5) algorithm as default.
# SHA256 supports 255-character passwords.
# Passwords previously encrypted with MD5 (1) will be encrypted
# with SHA256 (5) when users change their passwords.
#CRYPT_DEFAULT=1
CRYPT_DEFAULT=5
```

この例では、CRYPT_DEFAULT の新しい値が 5 (SHA256、SHA256 アルゴリズム) になっています。SHA は、Secure Hash Algorithm (セキュアハッシュアルゴリズム) を表します。このアルゴリズムは、SHA-2 ファミリのメンバーです。SHA256 では 255 文字のパスワードがサポートされます。

3. **(オプション) CRYPT_ALGORITHM_ALLOWED から以前のアルゴリズムを削除して、新しいアルゴリズムを既存のユーザーに適用させます。**

   たとえば、SHA256 アルゴリズムが既存のユーザーにも確実に適用されるようにするには、CRYPT_ALGORITHM_ALLOWED から MD5 を示す以前の識別子 1 を除外するようにしてください。

   注記 - さらに、FIPS 140 セキュリティーを向上させるには、Blowfish アルゴリズム (2a) をエントリから除外します。

   ```
   CRYPT_ALGORITHMS_ALLOW=5,6
   ```

例 3-4 異機種システム混在環境でパスワードの暗号化アルゴリズムを制約する

この例では、BSD および Linux システムが含まれるネットワーク上の管理者は、すべてのシステムで使用できるようにパスワードを構成します。SHA512 暗号化は一部のネットワークアプリケーションで処理できないため、管理者はその識別子を許容されるアルゴリズムのリストに含めません。管理者は、CRYPT_DEFAULT 変数の値として SHA256 アルゴリズム 5 を保持しています。CRYPT_ALGORITHMS_ALLOW 変数には、BSD および Linux システムと互換性のある MD5 識別子と、BSD システムと互換性のある Blowfish 識別子が含まれています。5 は CRYPT_DEFAULT アルゴリズムであるため、CRYPT_ALGORITHMS_ALLOW リストに載せる必要はありません。しかし、保守のために、管理者は 5 を CRYPT_ALGORITHMS_ALLOW リストに入れ、使われていない識別子を CRYPT_ALGORITHMS_DEPRECATE リストに入れます。

```
CRYPT_ALGORITHMS_ALLOW=1,2a,5
#CRYPT_ALGORITHMS_DEPRECATE=__unix__,md5,6
CRYPT_DEFAULT=5
```

▼ NIS ドメイン用の新しいパスワードアルゴリズムを指定する方法

NIS ドメインのユーザーがパスワードを変更すると、NIS クライアントは、/etc/security/policy.conf ファイルにある自身のローカルアルゴリズム構成を調べ、NIS クライアントシステムでパスワードを暗号化します。

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **パスワード暗号化アルゴリズムを NIS クライアント上の /etc/security/policy.conf ファイルに指定します。**
2. **変更された /etc/security/policy.conf ファイルを NIS ドメインのすべてのクライアントシステムにコピーします。**
3. **混乱をできるだけ少なくするために、変更された /etc/security/policy.conf ファイルを NIS ルートサーバーとスレーブサーバーにコピーします。**

▼ LDAP ドメイン用の新しいパスワードアルゴリズムを指定する方法

適切に構成された LDAP クライアントでは、新しいパスワードアルゴリズムを使用できます。LDAP クライアントは NIS クライアントと同じように動作します。

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **パスワード暗号化アルゴリズムを LDAP クライアント上の /etc/security/policy.conf ファイルに指定します。**
2. **変更された policy.conf ファイルを LDAP ドメインのすべてのクライアントシステムにコピーします。**
3. **クライアントの /etc/pam.conf ファイルが pam_ldap モジュールを使用していないことを確認します。**

pam_ldap.so.1 を含むエントリの前にコメント記号 (#) があることを確認します。また、pam_authtok_store.so.1 モジュールには server_policy オプションを使用しないでください。

ローカルアルゴリズム構成に基づくパスワードの暗号化は、クライアントの pam.conf ファイルの PAM エントリに従って行われます。パスワードの認証もこの PAM エントリによって行われます。

LDAP ドメインのユーザーがパスワードを変更すると、LDAP クライアントは、/etc/security/policy.conf ファイルにある自身のローカルアルゴリズム構成を調べ、LDAP クライアントシステムでパスワードを暗号化します。続いてクライアントは、{crypt} タグ付きの暗号化パスワードをサーバーに送信します。このタグは、パスワードが暗号化済みであることをサーバーに知らせます。パスワードはそのままの形でサーバーに格納されます。認証時に、クライアントはこのパスワードをサーバーから取り出します。クライアントは、このパスワードと、入力されたユーザーのパスワードからクライアントが暗号化したばかりのパスワードとを比較します。

注記 - LDAP サーバーでパスワードポリシー制御を使用するには、pam.conf ファイルの pam_authtok_store エントリに server_policy オプションを指定します。パスワードはそのあと、LDAP サーバー上で暗号化されます。手順については、『Oracle Solaris 11.2 ディレクトリサービスとネームサービスでの作業: LDAP 』の第 4 章「Oracle Directory Server Enterprise Edition への LDAP クライアントの設定」を参照してください。

---

root アクセスのモニタリングと制限

デフォルトでは、root 役割は初期ユーザーに割り当てられ、ローカルシステムに直接ログインしたり、Oracle Solaris システムにリモートログインしたりすることはできません。

▼ だれが su コマンドを使用しているかをモニターする方法

sulog ファイルには、ユーザーから root に切り替えるために使用される su の試行だけでなく、ユーザー切替え (su) コマンドのすべての使用が記録されます。

このファイルへの su ログの記録は、デフォルトで、/etc/default/su ファイルの次のエントリで有効になっています。

```
SULOG=/var/adm/sulog
```

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **/var/adm/sulog ファイルの内容を定期的にモニタリングします。**

```
# more /var/adm/sulog
SU 12/20 16:26 + pts/0 stacey-root
SU 12/21 10:59 + pts/0 stacey-root
SU 01/12 11:11 + pts/0 root-rimmer
SU 01/12 14:56 + pts/0 jdoe-root
SU 01/12 14:57 + pts/0 jdoe-root
```

ここには、次のような情報が表示されます。

- コマンドが入力された日時。
- 試行に成功した場合。プラス記号 (+) は成功を示し、マイナス記号 (-) は失敗を示します。
- コマンドが実行されたポート。
- ユーザー名と切り替えたユーザー ID。

注意事項 ??? を含むエントリは、su コマンドの制御端末を識別できないことを示しています。通常、デスクトップが表示される前の su コマンドのシステム呼び出しには、??? が含まれます。たとえば、SU 10/10 08:08 + ??? root-root です。ユーザーがデスクトップセッションを開始すると、ttynam コマンドは、次のように制御端末の値を sulog に返します。SU 10/10 10:10 + pts/3 jdoe-root。

次のようなエントリは、su コマンドがコマンド行で呼び出されなかったことを示している場合があります。SU 10/10 10:20 + ??? root-oracle。Trusted Extensions のユーザーが GUI を使用して oracle 役割に切り替えた可能性があります。

▼ root ログインを制限およびモニターする方法

この方法では、ローカルシステムにアクセスしようとする root をただちに検出できます。

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **/etc/default/login ファイルの CONSOLE エントリを確認します。**

```
CONSOLE=/dev/console
```

デフォルトのコンソールデバイスは /dev/console に設定されています。このように設定されていると、root はコンソールにログインできます。root はリモートログインを行うことはできません。

2. **root がリモートログインできないことを検証します。**

   リモートシステムから、root としてログインを試みます。

```
mach2 % ssh -l root mach1
Password:      <Type root password of mach1>
Password:
Password:
Permission denied (gssapi-keyex,gssapi-with-mic,publickey,keyboard-interactive).
```

   デフォルト構成では、root は役割であり、役割はログインできません。また、デフォルトの構成では、ssh プロトコルによって root ユーザーのログインが阻止されます。

3. **root になろうとする試みをモニターします。**

   デフォルトでは、root になろうとする試みが SYSLOG ユーティリティーによってコンソールに表示されます。

   a. **デスクトップに端末コンソールを開きます。**

   b. **別のウィンドウで、su コマンドを使用して root になります。**

```
% su -
Password:      <Type root password>
#
```

   端末コンソールにメッセージが表示されます。

```
Sep 7 13:22:57 mach1 su: 'su root' succeeded for jdoe on /dev/pts/6
```

例 3-5 root アクセスの試行のログ記録

この例では、root の試行は SYSLOG によってログに記録されていません。そのため、管理者は、/etc/default/su ファイル内の #CONSOLE=/dev/console エントリからコメントを削除することによって、これらの試行をログに記録します。

```
# CONSOLE determines whether attempts to su to root should be logged
# to the named device
#
CONSOLE=/dev/console
```

ユーザーが root になろうとすると、この試行が端末コンソールに出力されます。

```
SU 09/07 16:38 + pts/8 jdoe-root
```

注意事項 `/etc/default/login` ファイルにデフォルトの `CONSOLE` エントリが含まれている場合にリモートシステムから `root` になるには、ユーザーはまず、自分のユーザー名を使用してログインする必要があります。自分のユーザー名を使用してログインしたあと、ユーザーは `su` コマンドを使用して `root` になることができます。

コンソールに `Last login: Wed Sep 7 15:13:11 2011 from mach2` のようなエントリが表示された場合、システムは、リモート `root` ログインを許可するように構成されています。リモート `root` アクセスを防止するには、`/etc/default/login` ファイル内の `#CONSOLE=/dev/console` エントリを `CONSOLE=/dev/console` に変更します。`ssh` プロトコルをデフォルトに戻す方法については、`sshd_config`(4) のマニュアルページを参照してください。

システムハードウェアアクセスの制御

物理的なシステムは、ハードウェア設定にアクセスする際にパスワードを要求することで保護できます。また、ユーザーがアボートシーケンスを使ってウィンドウ表示システムから離れるのを防ぐことでシステムを保護することもできます。

BIOS を保護するには、ベンダーのドキュメントを参照してください。

▼ SPARC ハードウェアへのアクセスにパスワードを必要にする方法

始める前に Device Security、Maintenance and Repair、または System Administrator 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **端末ウィンドウで、PROM セキュリティーモードを有効にします。**

```
# eeprom security-mode=command

Changing PROM password:
New password:      <Type password>
Retype new password:      <Retype password>
```

値として command か full を選択します。詳細は、eeprom(1M) のマニュアルページを参照してください。

前述のコマンドを入力する際に PROM パスワードを要求されない場合は、システムがすでに PROM パスワードを持っています。

2. **(オプション) PROM パスワードを変更します。**

注意 - PROM パスワードを忘れないでください。このパスワードがないと、ハードウェアは使用できません。

```
# eeprom security-password=       Press Return
Changing PROM password:
New password:       <Type password>
Retype new password:       <Retype password>
```

新しい PROM セキュリティーモードとパスワードはただちに有効になりますが、それが認識できるのは、ほとんどの場合、次回のブート時です。

▼ システムのアボートシーケンスを無効にする方法

注記 - 一部のサーバーシステムにはキースイッチがあります。このキースイッチを安全な位置に設定すると、ソフトウェアキーボードのアボート設定がオーバーライドされます。そのため、次の手順で行なった変更が実装されないことがあります。

始める前に solaris.admin.edit/etc/default/kbd 承認が割り当てられている管理者になる必要があります。デフォルトでは、root 役割がこの承認を持っています。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **KEYBOARD_ABORT の値を disable に変更します。**

/etc/default/kbd ファイルの enable 行をコメントにします。次に disable 行を追加します。

```
# cat /etc/default/kbd
…
# KEYBOARD_ABORT affects the default behavior of the keyboard abort
# sequence, see kbd(1) for details.  The default value is "enable".
# The optional value is "disable".  Any other value is ignored.
…
#KEYBOARD_ABORT=enable
KEYBOARD_ABORT=disable
```

2. **キーボードのデフォルトを更新します。**

```
# kbd -i
```

♦ ♦ ♦ 第 4 章

4 デバイスアクセスの制御

この章では、デバイスを保護するための作業について説明するとともに、参考となるセクションを示します。この章の内容は次のとおりです。

デバイスの保護についての概要は、16 ページの「デバイスアクセスの制御」を参照してください。

デバイスポリシーの構成

デバイスポリシーは、システムに不可欠なデバイスに対するアクセスの制限または防止を行うものです。このポリシーはカーネルで適用されます。

次のタスクマップは、デバイスポリシーに関連するデバイス構成作業の参照先を示しています。

表 4-1 デバイスポリシーの構成タスクマップ

タスク	説明	手順
システム上のデバイスのデバイスポリシーを表示します。	デバイスとそれらのデバイスポリシーの一覧を表示します。	64 ページの「デバイスポリシーを表示する方法」
デバイスポリシーの変更を監査します。	デバイスポリシーの変更を監査トレール内に記録します。	64 ページの「デバイスポリシーの変更を監査する方法」
`/dev/arp` にアクセスします。	Oracle Solaris IP MIB-II 情報を取得します。	65 ページの「`/dev/*` デバイスから IP MIB-II 情報を取得する方法」

▼ デバイスポリシーを表示する方法

- **システム上のすべてのデバイスのデバイスポリシーを表示します。**

```
% getdevpolicy | more
DEFAULT
read_priv_set=none
write_priv_set=none
ip:*
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
…
```

例 4-1 特定のデバイスのデバイスポリシーを表示する

この例では、3 つのデバイスのデバイスポリシーが表示されています。

```
% getdevpolicy /dev/allkmem /dev/ipsecesp /dev/bge
/dev/allkmem
read_priv_set=all
write_priv_set=all
/dev/ipsecesp
read_priv_set=sys_net_config
write_priv_set=sys_net_config
/dev/bge
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
```

▼ デバイスポリシーの変更を監査する方法

デフォルトでは、as 監査クラスに、AUE_MODDEVPLCY 監査イベントが含まれます。

始める前に Audit Configuration 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **AUE_MODDEVPLCY 監査イベントを含む監査クラスをあらかじめ選択します。**

```
# auditconfig -getflags
current-flags
# auditconfig -setflags current-flags,as
```

詳細な手順については、『Oracle Solaris 11.2 での監査の管理 』の「監査クラスを事前選択する方法」を参照してください。

▼ /dev/* デバイスから IP MIB-II 情報を取得する方法

Oracle Solaris IP MIB-II 情報を取得するアプリケーションは、/dev/ip ではなく /dev/arp を開く必要があります。

1. **/dev/ip および /dev/arp のデバイスポリシーを決定します。**

```
% getdevpolicy /dev/ip /dev/arp
/dev/ip
read_priv_set=net_rawaccess
write_priv_set=net_rawaccess
/dev/arp
read_priv_set=none
write_priv_set=none
```

/dev/ip の読み取りおよび書き込みには、net_rawaccess 特権が必要であることに注意してください。/dev/arp は特権を必要としません。

2. **/dev/arp を開き、tcp モジュールと udp モジュールをプッシュします。**
特権は不要です。この方法は、/dev/ip を開いて arp、tcp、および udp モジュールをプッシュするのと同じです。現在、/dev/ip を開くには特権が必要なため、/dev/arp メソッドを推奨します。

デバイス割り当ての管理

デバイス割り当ては一般に、デバイスセキュリティーの追加の層が必要なサイトで実装されます。通常、割り当て可能なデバイスにアクセスするユーザーには承認が必要です。

次のタスクマップは、デバイス割り当ての有効化、構成、およびトラブルシューティングを行うための手順を示しています。デフォルトではデバイス割り当ては有効になっていません。デバイス割り当てを有効にしたあとで、デバイスを割り当てるための手順について、71 ページの「デバイスの割り当て」を参照してください。

表 4-2 デバイス割り当ての管理タスクマップ

タスク	説明	手順
デバイスを割り当て可能にします。	デバイスを一度に 1 人のユーザーに割り当てられるようにします。	66 ページの「デバイス割り当てを有効にする方法」
デバイス割り当てを無効にします。	すべてのデバイスの割り当て制限を解除します。	

タスク	説明	手順
ユーザーによるデバイス割り当てを承認します。	デバイス割り当ての承認をユーザーに与えます。	67 ページの「ユーザーによるデバイス割り当てを承認する方法」
システム上の割り当て可能なデバイスを表示します。	割り当てが可能なデバイスと、そのデバイスの状態を一覧表示します。	68 ページの「デバイスの割り当て情報を表示する方法」
デバイスを強制的に割り当てます。	デバイスを、ただちに必要とするユーザーに割り当てます。	68 ページの「デバイスを強制的に割り当てる方法」
デバイスの割り当てを強制的に解除します。	現在ユーザーに割り当てられているデバイスの割り当てを解除します。	69 ページの「デバイスの割り当てを強制的に解除する方法」
デバイスの割り当てプロパティーを変更します。	デバイスを割り当てるための要件を変更します。	69 ページの「割り当て可能デバイスの変更方法」
デバイス割り当てを監査します。	デバイス割り当てを監査トレールに記録します	70 ページの「デバイス割り当てを監査する方法」
デバイスクリーンスクリプトを作成します。	物理デバイスからデータを一掃します。	83 ページの「新しいデバイスクリーンスクリプトの作成」

▼ デバイス割り当てを有効にする方法

始める前に Device Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **デバイス割り当てサービスを有効にし、このサービスが有効になっていることを確認します。**

```
# svcadm enable svc:/system/device/allocate
# svcs -x allocate
svc:/system/device/allocate:default (device allocation)
State: online since September 10, 2011 01:10:11 PM PDT
See: allocate(1)
See: deallocate(1)
See: list_devices(1)
See: device_allocate(1M)
See: mkdevalloc(1M)
See: mkdevmaps(1M)
See: dminfo(1M)
See: device_maps(4)
See: /var/svc/log/system-device-allocate:default.log
Impact: None.
```

2. **デバイス割り当てサービスを無効にするには、`disable` サブコマンドを使用します。**

```
# svcadm disable device/allocate
```

▼ ユーザーによるデバイス割り当てを承認する方法

始める前に User Security 権利プロファイルが割り当てられている管理者になる必要があります。権利プロファイルには、`solaris.auth.delegate` 承認が含まれている必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **適切な承認とコマンドが入った権利プロファイルを作成します。**

   一般には、`solaris.device.allocate` 承認を含む権利プロファイルを作成します。『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「権利プロファイルを作成する方法」の指示に従います。権利プロファイルに、次に示すような適切なプロパティーを指定します。

   - 権利プロファイル名: `Device Allocation`
   - 付与される承認: `solaris.device.allocate`
   - 特権を持つコマンド: `sys_mount` 特権を持つ `mount`、および `sys_mount` 特権を持つ `umount`

2. **(オプション) 権利プロファイルの役割を作成します。**

   『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「ユーザーへの権利の割り当て」の指示に従います。次に示す役割プロパティーを参考にしてください。

   - 役割名: `devicealloc`
   - 役割の完全名: `Device Allocator`
   - 役割の説明: `Allocates and mounts allocated devices`
   - 権利プロファイル: `Device Allocation`

     この権利プロファイルは、この役割に含まれているプロファイルのリストの先頭に存在する必要があります。

3. **権利プロファイルを承認されたユーザーまたは承認された役割に割り当てます。**

4. **これらのユーザーにデバイス割り当ての方法を教えます。**

   リムーバブルメディアの割り当て例は、71 ページの「デバイスを割り当てる方法」を参照してください。

▼ デバイスの割り当て情報を表示する方法

始める前に 66 ページの「デバイス割り当てを有効にする方法」を完了します。

Device Security 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **システム上の割り当て可能デバイスについての情報を表示します。**

  ```
  # list_devices device-name
  ```

  device-name は次のいずれかです。

  - `audio`[*n*] – マイクとスピーカー。
  - `rmdisk`[*n*] – リムーバブルメディアデバイス (USB など)。
  - `sr`[*n*] – CD-ROM ドライブ。
  - `st`[*n*] – テープドライブ。

注意事項 `list_devices` コマンドが次のようなエラーメッセージを返す場合は、デバイス割り当てが有効になっていないか、情報を取得するために必要なアクセス権がありません。

```
list_devices: No device maps file entry for specified device.
```

コマンドを実行するには、デバイス割り当てを有効にし、`solaris.device.revoke` 承認のある役割になります。

▼ デバイスを強制的に割り当てる方法

強制的な割り当ては、誰かがデバイスの割り当て解除を忘れた場合や、デバイスをただちに使用する必要がある場合などに行います。

始める前に `solaris.device.revoke` 承認が割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **自分の役割に適切な承認が含まれているかどうかを確認します。**

   ```
   $ auths
   ```

```
solaris.device.allocate solaris.device.revoke
```

2. **デバイスを必要としているユーザーにデバイスを強制的に割り当てます。**

   この例では、USB フラッシュドライブがユーザー jdoe に強制的に割り当てられます。

```
$ allocate -U jdoe
```

▼ デバイスの割り当てを強制的に解除する方法

ユーザーが割り当てたデバイスは、プロセスの終了時やそのユーザーのログアウトの際に自動的に割り当てが解除されることはありません。強制的な割り当て解除は、ユーザーがデバイスの割り当てを解除することを忘れた場合に行います。

始める前に solaris.device.revoke 承認が割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **自分の役割に適切な承認が含まれているかどうかを確認します。**

```
$ auths
solaris.device.allocate solaris.device.revoke
```

2. **デバイスの割り当てを強制的に解除します。**

   この例では、別のユーザーが割り当てられるように、プリンタが強制的に割り当て解除されます。

```
$ deallocate -f /dev/lp/printer-1
```

▼ 割り当て可能デバイスの変更方法

始める前に この作業を行うには、デバイス割り当てが有効になっていなければなりません。デバイス割り当てを有効にするには、66 ページの「デバイス割り当てを有効にする方法」を参照してください。root 役割になる必要があります。

- **device_allocate ファイルでデバイスエントリの 5 番目のフィールドを変更して、承認が必要であるかどうかを指定したり、solaris.device.allocate 承認を指定したりします。**

```
audio;audio;reserved;reserved;solaris.device.allocate;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;solaris.device.allocate;/etc/security/lib/fd_clean
```

```
sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
```

solaris.device.allocate は、デバイスの使用に solaris.device.allocate 承認が必要であることを示します。

例 4-2 任意のユーザーによるデバイス割り当てを許可する

次の例では、システム上の任意のユーザーが任意のデバイスを割り当てることができます。device_allocate ファイルの各デバイスエントリ内にある 5 番目のフィールドは、単価記号 (@) に変更されました。

```
# pfedit /etc/security/device_allocate
audio;audio;reserved;reserved;@;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;@;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;@;/etc/security/lib/sr_clean
…
```

例 4-3 一部の周辺機器の使用を防止する

次の例では、オーディオデバイスの使用が禁止されています。device_allocate ファイルのオーディオデバイスエントリにある 5 番目のフィールドは、アスタリスク (*) に変更されました。

```
# pfedit /etc/security/device_allocate
audio;audio;reserved;reserved;*;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;solaris device.allocate;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;solaris device.allocate;/etc/security/lib/sr_clean
…
```

例 4-4 すべての周辺機器の使用を防止する

次の例では、使用できる周辺機器はありません。device_allocate ファイルの各デバイスエントリにある 5 番目のフィールドは、アスタリスク (*) に変更されました。

```
# pfedit /etc/security/device_allocate
audio;audio;reserved;reserved;*;/etc/security/lib/audio_clean
fd0;fd;reserved;reserved;*;/etc/security/lib/fd_clean
sr0;sr;reserved;reserved;*;/etc/security/lib/sr_clean
…
```

▼ デバイス割り当てを監査する方法

デフォルトでは、デバイス割り当てコマンドは、監査クラス other の状態です。

始める前に Audit Configuration 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **ot 監査クラスを事前選択します。**

```
$ auditconfig -getflags
current-flags
$ auditconfig -setflags current-flags,ot
```

詳細な手順については、『Oracle Solaris 11.2 での監査の管理 』の「監査クラスを事前選択する方法」を参照してください。

デバイスの割り当て

デバイス割り当ては、一度に 1 人のユーザーだけが使用できるようにデバイスを予約 (確保) する作業です。マウントポイントが必要なデバイスはマウントする必要があります。次の手順でユーザーに、デバイスを割り当てる方法を示します。

▼ デバイスを割り当てる方法

始める前に 66 ページの「デバイス割り当てを有効にする方法」の説明に従って、デバイス割り当てが有効になっている必要があります。承認が必要な場合は、そのユーザーは承認を得ていなければなりません。

1. **デバイスを割り当てます。**
   デバイス名でデバイスを指定します。

```
% allocate device-name
```

2. **コマンドを繰り返して、デバイスが割り当てられていることを確認します。**

```
% allocate device-name
allocate. Device already allocated.
```

例 4-5 マイクを割り当てる

この例では、ユーザー `jdoe` がマイク `audio0` を割り当てます。

```
% whoami
```

```
jdoe
% allocate audio0
```

例 4-6 プリンタを割り当てる

この例では、ユーザーがプリンタを割り当てます。このユーザーが `printer-1` の割り当てを解除するか、このプリンタが強制的にほかのユーザーに割り当てられるまで、ほかのユーザーはこのプリンタを使用できません。

```
% allocate /dev/lp/printer-1
```

強制的な割り当て解除の例については、69 ページの「デバイスの割り当てを強制的に解除する方法」を参照してください。

例 4-7 USB フラッシュドライブを割り当てる

この例では、ユーザーが USB フラッシュドライブ `rmdisk1` を割り当てます。

```
% allocate rmdisk1
```

注意事項 `allocate` コマンドがデバイスを割り当てることができない場合は、コンソールウィンドウにエラーメッセージが表示されます。割り当てのエラーメッセージについては、`allocate`(1) のマニュアルページを参照してください。

▼ 割り当て済みデバイスをマウントする方法

適切な特権が付与されている場合、デバイスは自動的にマウントします。デバイスがマウントに失敗した場合は、この手順に従います。

始める前に デバイスをすでに割り当てている必要があります。67 ページの「ユーザーによるデバイス割り当てを承認する方法」の説明に従って、デバイスをマウントするために必要な特権が割り当てられています。

1. **デバイスの割り当てまたはマウントが行える役割になります。**

```
% su - role-name
Password:       <Type role-name password>
$
```

2. **この役割のホームディレクトリにマウントポイントを作成し、このマウントポイントを保護します。**

この手順を実行する必要があるのは、マウントポイントがはじめて必要になったときだけです。

```
$ mkdir mount-point ; chmod 700 mount-point
```

3. **割り当てが可能なデバイスを一覧表示します。**

```
$ list_devices -l
List of allocatable devices
```

4. **デバイスを割り当てます。**

デバイス名でデバイスを指定します。

```
$ allocate device-name
```

5. **デバイスをマウントします。**

```
$ mount -o ro -F filesystem-type device-path mount-point
```

`-o ro`	デバイスは読み取り専用としてマウントされることを示します。デバイスを書き込み可能にするには、`-o rw` を使用します。
`-F` *filesystem-type*	デバイスのファイルシステムフォーマットを示します。一般に、CD-ROM は HSFS ファイルシステムでフォーマットされています。フロッピーディスクは、一般に PCFS ファイルシステムでフォーマットされています。
device-path	デバイスへのパスを示します。`list_devices -l` コマンドの出力には、*device-path* が含まれます。
mount-point	ステップ 2 で作成したマウントポイントを示します。

例 4-8 CD-ROM ドライブを割り当てる

この例では、ユーザーは CD-ROM ドライブ `sr0` の割り当てとマウントが行える役割を引き受けます。このドライブは、HSFS ファイルシステムでフォーマットされています。

```
% roles
devicealloc
% su - devicealloc
Password:      <Type devicealloc password>
$ mkdir /home/devicealloc/mymnt
$ chmod 700 /home/devicealloc/mymnt
$ list_devices -l
...
device: sr0 type: sr files: /dev/sr0 /dev/rsr0 /dev/dsk/c0t2d0s0 ...
...
$ allocate sr0
$ mount -o ro -F hsfs /dev/sr0 /home/devicealloc/mymnt
$ cd /home/devicealloc/mymnt ; ls
```

List of the contents of CD-ROM

注意事項 `mount` コマンドがデバイスをマウントできない場合は、「`mount: insufficient privileges`」というエラーメッセージが表示されます。次の点を確認してください。

- `mount` コマンドをプロファイルシェルで実行していることを確認します。役割を引き受けた場合は、その役割にプロファイルシェルがあります。`mount` コマンドでプロファイルを割り当てられたユーザーの場合、プロファイルシェルを作成する必要があります。使用可能なプロファイルシェルのリストについては、`pfexec`(1) のマニュアルページを参照してください。
- 指定されたマウントポイントを所有していることを確認します。このマウントポイントに対する読み取り、書き込み、および実行のアクセス権が必要です。

以上の要件を満たしているにもかかわらず割り当て済みデバイスをマウントできないという場合は、管理者に問い合わせてください。まず、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「権利割り当てをトラブルシューティングする方法」を参照してください。

▼ デバイスの割り当てを解除する方法

割り当てを解除すると、ほかのユーザーもユーザーの使用後にそのデバイスを割り当てて使用できるようになります。

始める前に デバイスをすでに割り当てていなければなりません。詳細は、71 ページの「デバイスを割り当てる方法」を参照してください。

1. **デバイスがマウントされている場合は、デバイスのマウントを解除します。**

```
$ cd $HOME
$ umount mount-point
```

2. **デバイスの割り当てを解除します。**

```
$ deallocate device-name
```

例 4-9 マイクの割り当てを解除する

この例では、ユーザー `jdoe` がマイク `audio` の割り当てを解除します。

```
% whoami
jdoe
% deallocate audio0
```

例 4-10 CD-ROM ドライブの割り当てを解除する

この例では、Device Allocator 役割が CD-ROM ドライブの割り当てを解除します。次のメッセージが表示されたあとで、CD-ROM が取り出されます。

```
$ whoami
devicealloc
$ cd /home/devicealloc
$ umount /home/devicealloc/mymnt
$ ls /home/devicealloc/mymnt
$
$ deallocate sr0
/dev/sr0:      326o
/dev/rsr0:     326o
…
sr_clean: Media in sr0 is ready.  Please, label and store safely.
```

デバイス保護リファレンス

Oracle Solaris でのデバイスは、カーネルのデバイスポリシーによって保護されます。周辺機器は、デバイス割り当てによって保護できます。デバイス割り当ては、ユーザーレベルで任意に有効化と適用が行われます。

デバイスポリシーコマンド

デバイス管理コマンドは、ローカルファイルのデバイスポリシーを管理するコマンドです。デバイスポリシーは特権要件を含むことができます。Device Management および Device Security 権利プロファイルが割り当てられているユーザーはデバイスを管理できます。

次の表は、デバイス管理コマンドを示しています。

表 4-3 デバイス管理コマンド

コマンド	目的
add_drv(1M)	稼働中のシステムに新しいデバイスドライバを追加します。新しいデバイスにデバイスポリシーを追加するオプションを含みます。一般に、このコマンドはデバイスドライバのインストール中にスクリプト内で呼び出されます。
devfsadm(1M)	稼働しているシステム上のデバイスとデバイスドライバを管理します。また、デバイスポリシーの読み込みも行います。

コマンド	目的
	devfsadm コマンドは、ディスクデバイス、テープデバイス、ポートデバイス、オーディオデバイス、および擬似デバイスに対する /dev リンクのクリーンアップにも使用できます。名前付きドライバのデバイスの再構成も行えます。
getdevpolicy(1M)	1 つ以上のデバイスに関連付けられたポリシーを表示します。このコマンドはどのユーザーでも実行できます。
rem_drv(1M)	デバイスまたはデバイスドライバを削除します。
update_drv(1M)	既存のデバイスドライバの属性を更新します。デバイスのデバイスポリシーを更新するオプションを含みます。一般に、このコマンドはデバイスドライバのインストール中にスクリプト内で呼び出されます。

デバイスの割り当て

デバイス割り当てによって、データの消失、コンピュータウイルス、セキュリティー侵害などからサイトを保護できます。デバイスポリシーと違い、デバイス割り当ては任意です。デバイス割り当ては、割り当て可能デバイスへのアクセスを制限するのに承認を使用します。

デバイス割り当てのコンポーネント

デバイス割り当てメカニズムのコンポーネントは、次のとおりです。

- svc:/system/device/allocate サービス。詳細は、smf(5) のマニュアルページおよびデバイス割り当てコマンドのマニュアルページを参照してください。
- allocate、deallocate、dminfo、list_devices コマンド。詳細は、77 ページの「デバイス割り当てコマンド」を参照してください。
- Device Management および Device Security 権利プロファイル。詳細は、77 ページの「デバイス割り当て権利プロファイル」を参照してください。
- 各割り当て可能デバイスのデバイスクリーンスクリプト。

これらのコマンドとスクリプトは、次のローカルファイルを使用してデバイス割り当てを実装します。

- /etc/security/device_allocate ファイル。詳細は、device_allocate(4) のマニュアルページを参照してください。
- /etc/security/device_maps ファイル。詳細は、device_maps(4) のマニュアルページを参照してください。

- ロックファイル。割り当て可能デバイスごとに /etc/security/dev ディレクトリに配置します。
- 各割り当て可能デバイスに関連付けられたロックファイルの変更後の属性。

デバイス割り当てサービス

svc:/system/device/allocate サービスは、デバイス割り当てを制御します。このサービスはデフォルトで無効になっています。

デバイス割り当て権利プロファイル

デバイスおよびデバイス割り当てを管理するには、Device Management および Device Security 権利プロファイルが必要です。
これらの権利プロファイルには、次の承認が含まれています。

- solaris.device.allocate – デバイスを割り当てるために必要
- solaris.device.cdrw – CD-ROM の読み取りと書き込みを行うために必要
- solaris.device.config – デバイスの属性を構成するために必要
- solaris.device.mount.alloptions.fixed – 固定デバイスのマウント時にマウントオプションを指定するために必要
- solaris.device.mount.alloptions.removable – リムーバブルデバイスのマウント時にマウントオプションを指定するために必要
- solaris.device.mount.fixed – 固定デバイスをマウントするために必要
- solaris.device.mount.removable – リムーバブルデバイスをマウントするために必要
- solaris.device.revoke – デバイスを取り消すか、または再利用するために必要

デバイス割り当てコマンド

大文字のオプションが指定された allocate、deallocate、および list_devices コマンドは管理コマンドです。それ以外ではこれらのコマンドはユーザーコマンドです。次の表は、デバイス割り当てコマンドを示しています。

表 4-4 デバイス割り当てコマンド

コマンドのマニュアルページ	目的
allocate(1)	1 人のユーザーだけが使用できるように割り当て可能デバイスを予約します。 デフォルトでは、ユーザーがデバイスを割り当てるには solaris.device.allocate 承認が必要です。ユーザー承認を必要としないように、device_allocate ファイルを変更することもできます。そのように変更した場合、システム上のどのユーザーでもデバイスの使用割り当てを要求できます。
deallocate(1)	デバイスから割り当て予約を削除します。
dminfo(1M)	デバイスタイプ、デバイス名、またはフルパス名を指定して、割り当て可能デバイスを検索します。
list_devices(1)	割り当て可能なデバイスのステータスを表示します。 device_maps ファイルにリストされたデバイスに関連付けられている、デバイス特殊ファイルを列挙します。 -U オプションがあると、割り当て可能なデバイス、または指定されたユーザー ID に割り当てられているデバイスを一覧表示します。このオプションを使用すると、別のユーザーに割り当てることができるデバイスまたは割り当て済みのデバイスを確認できます。このコマンドを実行するには、ユーザーまたは役割に solaris.device.revoke 承認がなければなりません。

割り当てコマンドの承認

デフォルトでは、ユーザーが割り当て可能デバイスを予約するには solaris.device.allocate 承認を必要とします。solaris.device.allocate 承認を含める権利プロファイルを作成する方法については、67 ページの「ユーザーによるデバイス割り当てを承認する方法」を参照してください。

管理者がデバイスの割り当て状態を変更するには solaris.device.revoke 承認が必要です。たとえば、allocate および list_devices コマンドの -U オプションや、deallocate コマンドの -F オプションは、solaris.device.revoke 承認が必要です。

詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「承認を必要とする特別なコマンド」を参照してください。

割り当てエラー状態

deallocate コマンドが割り当ての解除に失敗する場合、または allocate コマンドが割り当てに失敗する場合は、デバイスは*「割り当てエラー状態」*になります。割り当て可能デバイスが割り当てエラー状態となった場合、そのデバイスの割り当てを強制的に解除する必要があります。割り当てエラー状態を処理できるのは、Device Management 権利プロファイルまたは Device Security 権利プロファイルを持つユーザーまたは役割だけです。

F オプションを指定した -deallocate コマンドは、割り当て解除を強制します。あるいは、allocate -U を実行してデバイスを特定のユーザーに割り当てることもできます。いったんデバイスが割り当てられると、発生したエラーメッセージを調査できます。デバイスに関する問題が解決されたあとで、そのデバイスの割り当てを強制的に解除できます。

device_maps ファイル

デバイス割り当てを設定すると、デバイスマップが作成されます。/etc/security/device_maps ファイルには、割り当て可能な各デバイスに関連付けられたデバイス名、デバイスタイプ、およびデバイス特殊ファイルが含まれています。

直観的にはわかりにくい各デバイスのために、device_maps ファイルはデバイス特殊ファイルのマッピングを定義します。このファイルによって、プログラムはどのデバイス特殊ファイルがどのデバイスに割り当てられているかを検出できます。たとえば、dminfo コマンドを使用すると、デバイス名、デバイスの種類およびデバイス特殊ファイルを取得して、割り当て可能なデバイスを設定するときに指定できます。dminfo コマンドは、device_maps ファイルを使用してデバイス割り当て情報を報告します。

各デバイスは、次の形式の 1 行のエントリで表されます。

device-name:*device-type*:*device-list*

例 4-11 device_maps エントリの例

次の例は、device_maps ファイルのエントリを示したものです。

```
audio0:\
audio:\
/dev/audio /dev/audioctl /dev/dsp /dev/dsp0 /dev/mixer0 /dev/sound/0
/dev/sound/0ctl /dev/sound/audio810\:0mixer /dev/sound/audio810\:0dsp
```

```
/dev/sound/audio810\:0 /dev/sound/audio810\:0ctl
```

device_maps ファイル内の行は、バックスラッシュ (\) で終了することにより、エントリを次の行に続けることができます。コメントも挿入できます。ポンド記号 (#) を付けると、1 つ前の行末にバックスラッシュのない改行まで、それに続くすべてのテキストはコメントになります。どのフィールドでも先行ブランクと後続ブランクを使用できます。フィールドの定義は次のとおりです。

device-name
: デバイスの名前を指定します。現在のデバイス名の一覧を表示する方法については、68 ページの「デバイスの割り当て情報を表示する方法」を参照してください。

device-type
: 汎用デバイスタイプを指定します。汎用名は、st、fd、rmdisk、audio などの、デバイスのクラスの名前です。*device-type* では、関連するデバイスが論理的にグループ化されます。

device-list
: 物理デバイスに関連付けられたデバイス特殊ファイルを一覧表示します。*device-list* には、特定のデバイスにアクセスできるすべての特殊ファイルが含まれている必要があります。リストが不完全な場合は、悪意を持ったユーザーでも個人情報を入手または変更できます。*device-list* フィールドには、/dev ディレクトリに入っているデバイスファイルを指定します。

device_allocate ファイル

デバイスを割り当て可能から割り当て不可能に変更したり、新しいデバイスを追加したりするために、/etc/security/device_allocate ファイルを変更できます。

device_allocate ファイル内のエントリは、デバイスが割り当て可能であると特に記載していないかぎり、そのデバイスが割り当て可能であることを示しません。

device_allocate ファイルでは、各デバイスは次の形式の 1 行のエントリで表されます。

device-name;*device-type*;reserved;reserved;*auths*;*device-exec*

次の例は、device_allocate ファイルのサンプルを示しています。

```
st0;st;;;;/etc/security/lib/st_clean
fd0;fd;;;;/etc/security/lib/fd_clean
sr0;sr;;;;/etc/security/lib/sr_clean
audio;audio;;;*;/etc/security/lib/audio_clean
```

audio デバイスエントリの 5 番目のフィールドにあるアスタリスク (*) に注意してください。

device_allocate ファイル内の行は、バックスラッシュ (\) で終了することにより、エントリを次の行に続けることができます。コメントも挿入できます。ポンド記号 (#) を付けると、1 つ前の行末にバックスラッシュのない改行まで、それに続くすべてのテキストはコメントになります。どのフィールドでも先行ブランクと後続ブランクを使用できます。フィールドの定義は次のとおりです。

device-name
: デバイスの名前を指定します。現在のデバイス名の一覧を表示する方法については、68 ページの「デバイスの割り当て情報を表示する方法」を参照してください。

device-type
: 汎用デバイスタイプを指定します。汎用名は、st、fd、sr などのデバイスクラス名です。*device-type* では、関連するデバイスが論理的にグループ化されます。デバイスを割り当て可能にするときは、device_maps ファイルの *device-type* フィールドからデバイス名を取得します。

reserved
: Oracle では、reserved で示される 2 つのフィールドを将来の使用に予約しています。

auths
: デバイスが割り当て可能であるかどうかを指定します。このフィールドにアスタリスク (*) が入っている場合は、デバイスが割り当て不可能であることを示します。承認を示す文字列が入っている場合や、空の場合は、デバイスが割り当て可能であることを示します。たとえば、*auths* フィールド内の文字列 solaris.device.allocate は、そのデバイスを割り当てるには solaris.device.allocate 承認が必要であることを示します。このフィールドに単価記号 (@) が入っている場合は、どのユーザーでもそのデバイスを割り当てることができることを示します。

device-exec
: 割り当てプロセス中にクリーンアップやオブジェクト再利用防止などの特殊処理のために呼び出されるスクリプトのパス名を指定します。*device-exec* スクリプトは、デバイスに対して deallocate コマンドを実行するたびに実行されます。

たとえば、sr0 デバイスについての次のエントリは、CD-ROM ドライブが solaris.device.allocate 承認を得たユーザーによって割り当て可能であることを示します。

```
sr0;sr;reserved;reserved;solaris.device.allocate;/etc/security/lib/sr_clean
```

デフォルトのデバイスとそれらの定義された特性を受け入れることを決定できます。新しいデバイスをインストールしたあとで、エントリを変更できます。使用前に割り当てが必要なデバイスはすべて、そのデバイスのシステムの device_allocate ファイルと device_maps ファイルで定義する必要があります。現在は、カートリッジテープドライブ、フロッピーディスクドライブ、CD-ROM ドライブ、リムーバブルメディアデバイス、およびオーディオチップが、割り当て可能とみなされます。これらのデバイスタイプには、デバイスクリーンスクリプトが用意されています。

注記 - Xylogics および Archive テープドライブもまた、SCSI デバイスのために提供されている `st_clean` スクリプトを使用します。端末、グラフィックスタブレット、その他の割り当て可能なデバイスなどのほかのデバイスについては、ユーザー独自のデバイスクリーンスクリプトを作成する必要があります。このスクリプトは、そのデバイスタイプのオブジェクト再利用の要件を満たしている必要があります。

デバイスクリーンスクリプト

デバイス割り当てによって、セキュリティー監査者が*オブジェクト再利用*の要件と呼ぶものの一部が満たされます。デバイスクリーンスクリプトは、使用可能なすべてのデータを、再利用の前に物理デバイスから消去するというセキュリティー要件に対応します。データのクリアは、そのデバイスが別のユーザーによって割り当て可能になる前に実行されます。デフォルトでは、カートリッジテープドライブ、フロッピーディスクドライブ、CD-ROM ドライブ、オーディオデバイスには、Oracle Solaris で提供されるデバイスクリーンスクリプトが必要です。このセクションでは、デバイスクリーンスクリプトが実行する処理について説明します。

テープ用のデバイスクリーンスクリプト

`st_clean` デバイスクリーンスクリプトでは、3 つのテープデバイスがサポートされます。

- SCSI ¼ インチテープ
- アーカイブ ¼ インチテープ
- オープンリール ½ インチテープ

`st_clean` スクリプトは、`mt` コマンドの `rewoffl` オプションを使用してデバイスをクリーンアップします。詳細は、`mt(1)` のマニュアルページを参照してください。このスクリプトは、システムブート中に実行されると、デバイスを照会し、デバイスがオンラインであるかどうかを確認します。デバイスがオンラインの場合、スクリプトは、そのデバイスにメディアが挿入されているかどうかを調べます。¼ インチのテープデバイスにメディアが挿入されていた場合、このデバイスは割り当てエラー状態になります。この場合、管理者はそのデバイスを手動でクリーンアップする必要があります。

通常のシステム操作中に、`deallocate` コマンドを対話型モードで実行すると、メディアを取り出すように求めるプロンプトが表示されます。割り当て解除は、デバイスからメディアが取り出されるまで見送られます。

フロッピーディスクドライブと CD-ROM ドライブ用のデバイスクリーンスクリプト

フロッピーディスクドライブと CD-ROM ドライブ用として、次のデバイスクリーンスクリプトが提供されています。

- **fd_clean** スクリプト – フロッピーディスクドライブ用デバイスクリーンスクリプト。
- **sr_clean** スクリプト – CD-ROM ドライブ用デバイスクリーンスクリプト。

これらのスクリプトは、eject コマンドを使用してドライブからメディアを取り出します。eject コマンドが失敗すると、デバイスは割り当てエラー状態になります。詳細は、eject(1) のマニュアルページを参照してください。

オーディオ用のデバイスクリーンスクリプト

オーディオデバイスは、audio_clean スクリプトを使用してクリーンアップします。スクリプトは、AUDIO_GETINFO ioctl システムコールを実行してデバイスを読み取ります。AUDIO_SETINFO ioctl システムコールを実行してデバイス構成をデフォルトにリセットします。

新しいデバイスクリーンスクリプトの作成

システムに新しく割り当て可能デバイスを追加する場合は、独自のデバイスクリーンスクリプトを作成する必要があります。deallocate コマンドは、デバイスクリーンスクリプトにパラメータを渡します。次に示すように、パラメータはデバイス名を含む文字列です。詳細は、device_allocate(4) のマニュアルページを参照してください。

clean-script `-[I|i|f|S]` *device-name*

デバイスクリーンスクリプトは、成功時には「0」を、失敗時には「0」より大きな値を返す必要があります。オプション -I、-f、および -S は、スクリプトの実行モードを決定します。

-I	システムをブートするときにだけ指定します。すべての出力は、システムコンソールに送られます。失敗した場合や、メディアを強制的に取り出せない場合は、デバイスを割り当てエラー状態にします。
-i	出力が抑止される点を除き、-I オプションと同じです。
-f	強制的なクリーンアップ用。このオプションは対話型であり、ユーザーがプロンプトに応答するものとみなします。このオプションが付いたスクリプト

は、クリーンアップの一部に失敗した場合に、クリーンアップ全体を完了しようとします。

-S 標準クリーンアップ。このオプションは対話型であり、ユーザーがプロンプトに応答するものとみなします。

♦ ♦ ♦ 第 5 章

5 ウイルススキャンサービス

この章では、ウイルス対策ソフトウェアの使用についての情報を提供します。この章の内容は次のとおりです。

ウイルススキャンについて

データは、各種*スキャンエンジン*を使用するスキャンサービス vscan によってウイルスから保護されます。スキャンエンジンとは、外部ホストに常駐するサードパーティーのアプリケーションであり、ファイルで既知のウイルスを調べます。ファイルがウイルススキャンの候補となるのは、そのファイルシステムが vscan サービスをサポートし、そのサービスが有効になっていて、ファイルのタイプが対象外になっていない場合です。そして、ファイルが最新のウイルス定義でまだスキャンされていない場合、またはファイルが最後にスキャンされた以後に変更されている場合、ファイルのオープンおよびクローズ操作中にウイルススキャンが実行されます。

vscan サービスは、複数のスキャンエンジンを使用するように構成できます。vscan サービスで 2 つ以上のスキャンエンジンを使用することがベストプラクティスです。ウイルススキャンの要求は、使用できるすべてのスキャンエンジンに配信されます。

vscanadm show コマンドは、システム上に構成されているスキャンエンジンを一覧表示します。

```
# vscanadm show
max-size=1GB max-size-action=allow
types=+*
no scan engines configured
```

vscan サービスについて

リアルタイムスキャン方法の利点は、ファイルが使用される*前に*最新のウイルス定義でスキャンされることです。この方法を使用することで、ウイルスがデータを危険にさらす前にそれらを検出できます。

ユーザーがクライアントからファイルを開くと、ウイルススキャンプロセスが次のように動作します。

1. vscan サービスは、そのファイルが最新のウイルス定義ですでにスキャンされているかどうか、およびそのファイルが最後にスキャンされた以後に変更されたかどうかに基づいて、ファイルのスキャンが必要かどうかを判断します。

   スキャンが必要ない場合は、プロセスが終了し、ユーザーはファイルへのアクセスが許可されます。
2. スキャンが必要である場合は、ファイルがスキャンエンジンに転送されます。

   転送が正常に完了すると、エンジンは現在のウイルス定義を使用してスキャンして、ファイルが感染しているかどうかを判断します。

   転送に失敗した場合は、次のようにプロセスが続行します。

   - ファイルスキャンを実行できる次のスキャンエンジンに、ファイルが転送されます。
   - 代替のエンジンが存在しない場合や使用できない場合は、ウイルススキャンが失敗したとみなされ、ファイルへのアクセスが拒否される可能性があります。
3. ウイルスが検出されない場合は、ファイルがスキャンスタンプでタグ付けされ、クライアントはそのファイルへのアクセスを許可されます。

   ウイルスが検出された場合、そのファイルには隔離されたことを示すマークが付けられます。隔離されたファイルの読み取り、実行、または名前の変更はできませんが、削除はできます。システムログには、隔離されたファイルの名前とウイルスの名前が記録され、さらに監査が有効になっていた場合は、同じ情報が含まれた監査レコードが作成されます。

vscan サービスの使用

ファイルのウイルススキャンは、次の要件が満たされたときに使用できます。

- 1 つ以上のスキャンエンジンがインストールされ、構成されている。
- ファイルが、ウイルススキャンをサポートしているファイルシステム上に存在する。
- そのファイルシステムでウイルススキャンが有効になっている。

- vscan サービスが有効になっている。
- vscan サービスが、指定されたファイルタイプのファイルをスキャンするように構成されている。

次の表は、vscan サービスを設定するために行うタスクを示しています。

タスク	説明	手順
スキャンエンジンをインストールします。	Oracle Solaris でサポートされているサードパーティー製品を 1 つ以上インストールし、構成します。	製品のドキュメントを参照してください。
ファイルシステムでウイルススキャンを使用できるようにします。	ZFS ファイルシステムでのウイルススキャンを有効にします。デフォルトでは、スキャンは無効になっています。	87 ページの「ファイルシステムでウイルススキャンを有効にする方法」
vscan サービスを有効にします。	スキャンサービスを開始します。	88 ページの「vscan サービスを有効にする方法」
スキャンエンジンを vscan サービスに追加します。	特定のスキャンエンジンを vscan サービスに組み込みます。	88 ページの「スキャンエンジンを追加する方法」
vscan サービスを構成します。	vscan プロパティーを表示および変更します。	89 ページの「vscan プロパティーを表示する方法」 89 ページの「スキャンするファイルのサイズを制限する方法」
特定のファイルタイプ向けに vscan サービスを構成します。	スキャンに組み込んだり、スキャンから除外したりするファイルタイプを指定します。	90 ページの「ウイルススキャンからファイルを除外する方法」

▼ ファイルシステムでウイルススキャンを有効にする方法

ファイルのウイルススキャンを可能にするには、ファイルシステムコマンドを使用します。たとえば、ZFS ファイルシステムをウイルススキャンに組み込むには、zfs(1M) コマンドを使用します。

ZFS ファイルシステムでは、一部の管理タスクを特定のユーザーに委託できます。委託管理の詳細は、『Oracle Solaris 11.2 での ZFS ファイルシステムの管理 』の第 8 章「Oracle Solaris ZFS 委任管理」を参照してください。

始める前に ZFS File System Management または ZFS Storage Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユー

ザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **ZFS ファイルシステムでのウイルススキャンを有効にします。**

  ```
  # zfs set vscan=on zfs-file-system
  ```

  たとえば、ZFS ファイルシステムが *path*/pool/volumes/vol1 である場合は、次のコマンドを入力します。

  ```
  # zfs set vscan=on path/pool/volumes/vol1
  ```

▼ vscan サービスを有効にする方法

始める前に VSCAN Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **ウイルススキャンサービスを有効にします。**

  ```
  # svcadm enable vscan
  ```

▼ スキャンエンジンを追加する方法

始める前に VSCAN Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **デフォルトのプロパティーを使用してスキャンエンジンを vscan サービスに追加するには、次のように入力します。**

  ```
  # vscanadm add-engine engineID
  ```

  詳細は、vscanadm(1M) のマニュアルページを参照してください。

▼ vscan プロパティーを表示する方法

始める前に VSCAN Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

- **すべてのスキャンエンジンまたは特定のスキャンエンジンについて、vscan サービスのプロパティーを表示します。**

  - **特定のスキャンエンジンのプロパティーを表示するには、次を入力します。**

    ```
    # vscanadm get-engine engineID
    ```

  - **すべてのスキャンエンジンのプロパティーを表示するには、次を入力します。**

    ```
    # vscanadm get-engine
    ```

  - **vscan サービスのいずれかのプロパティーを表示するには、次のように入力します。**

    ```
    # vscanadm get -p property
    ```

    ここで、*property* は vscanadm(1M) コマンドのマニュアルページに記載されているパラメータの 1 つです。

    たとえば、スキャンできるファイルの最大サイズを表示する場合は次を入力します。

    ```
    # vscanadm get max-size
    ```

▼ スキャンするファイルのサイズを制限する方法

多くのスキャンエンジンではスキャンできるファイルのサイズが制限されているため、vscan サービスの max-size プロパティーをスキャンエンジンの最大許容サイズ以下の値に設定する必要があります。その際、最大サイズよりも大きく、そのためにスキャンされないファイルをアクセス可能にするかどうかを定義します。

始める前に VSCAN Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **現在のプロパティーを表示します。**

```
# vscanadm show
```

2. **ウイルススキャンの最大サイズを設定します。**

   たとえば、128 メガバイトの制限を設定するには、次のように入力します。

   ```
   # vscanadm set -p max-size=128M
   ```

3. **そのサイズのせいでスキャンされないファイルへのアクセスが拒否されるように指定します。**

   ```
   # vscanadm set -p max-size-action=deny
   ```

   詳細は、vscanadm(1M) のマニュアルページを参照してください。

▼ ウイルススキャンからファイルを除外する方法

ウイルス対策保護を有効にした場合、特定のタイプのすべてのファイルがウイルススキャンから除外されるように指定できます。vscan サービスはシステムのパフォーマンスに影響を与えるため、特定のファイルタイプをウイルススキャンの対象とすることで、システムリソースを節約できます。

始める前に VSCAN Management 権利プロファイルが割り当てられている管理者になる必要があります。詳細は、『Oracle Solaris 11.2 でのユーザーとプロセスのセキュリティー保護 』の「割り当てられている管理権利の使用」を参照してください。

1. **ウイルススキャンに含まれているすべてのファイルタイプの一覧を表示します。**

   ```
   # vscanadm get -p types
   ```

2. **ウイルススキャンの対象となるファイルのタイプを指定します。**

   例:

   - **特定のファイルタイプ (たとえば、JPEG タイプ) をウイルススキャンから除外するには、次のように入力します。**

     ```
     # vscanadm set -p types=-jpg,+*
     ```

   - **特定のファイルタイプ (たとえば、実行可能ファイル) をウイルススキャンに含めるには、次のように入力します。**

     ```
     # vscanadm set -p types=+exe,-*
     ```

詳細は、vscanadm(1M) のマニュアルページを参照してください。

セキュリティー用語集

アクセス制御リスト (ACL) アクセス制御リスト (ACL) を使用すると、従来の UNIX ファイル保護よりもきめ細かな方法でファイルセキュリティーを確立できます。たとえば、特定のファイルにグループ読み取り権を設定し、そのグループ内の 1 人のメンバーだけにそのファイルへの書き込み権を与えることが可能です。

アプリケーションサーバー ネットワークアプリケーションサーバーを参照してください。

アルゴリズム 暗号化アルゴリズム。これは、入力を暗号化 (ハッシング) する既成の再帰的な計算手続きです。

暗号化アルゴリズム アルゴリズムを参照してください。

暗号化フレームワークにおけるポリシー Oracle Solaris の暗号化フレームワーク機能では、ポリシーは既存の暗号化メカニズムの無効化です。無効に設定されたメカニズムは使用できなくなります。暗号化フレームワークにおけるポリシーにより、プロバイダ (DES など) からの特定のメカニズム (`CKM_DES_CBC`など) を使用できなくなることがあります。

インスタンス 主体名の 2 番目の部分。インスタンスは、主体の主ノード指定します。サービス主体の場合、インスタンスは必ず指定する必要があります。`host/central.example.com` にあるように、インスタンスはホストの完全修飾ドメイン名です。ユーザー主体の場合、インスタンスは省略することができます。ただし、`jdoe` と `jdoe/admin` は、一意の主体です。プライマリ、主体名、サービス主体、ユーザー主体も参照してください。

オーセンティケータ オーセンティケータは、KDC にチケットを要求するときおよびサーバーにサービスを要求するときに、クライアントから渡されます。オーセンティケータには、クライアントとサーバーだけが知っているセッション鍵を使用して生成された情報が含まれます。オーセンティケータは、最新の識別として検査され、そのトランザクションが安全であることを示します。これをチケットとともに使用すると、ユーザー主体を認証できます。オーセンティケータには、ユーザーの主体名、ユーザーのホストの IP アドレス、タイムスタンプが含まれます。チケットとは異なり、オーセンティケータは一度しか使用できません。通常、サービスへのアクセスが要求されたときに使用されます。オーセンティケータは、そのクライアントとそのサーバーのセッション鍵を使用して暗号化されます。

鍵 1. 一般には、次に示す 2 種類の主要鍵のどちらか一方です。

- *対称鍵* – 復号化鍵とまったく同じ暗号化鍵。対称鍵はファイルの暗号化に使用されます。

- *非対称鍵*または*公開鍵* – Diffie-Hellman や RSA などの公開鍵アルゴリズムで使用される鍵。公開鍵には、1 人のユーザーしか知らない非公開鍵、サーバーまたは一般リソースによって使用される公開鍵、およびこれらの 2 つを組み合わせた公開鍵と非公開鍵のペアがあります。非公開鍵は、「秘密鍵」とも呼ばれます。公開鍵は、「共有鍵」や「共通鍵」とも呼ばれます。

2. キータブファイルのエントリ (主体名)。キータブファイルも参照してください。
3. Kerberos では暗号化鍵であり、次の 3 種類があります。

- 「非公開鍵」– 主体と KDC によって共有される暗号化鍵。システムの外部に配布されます。非公開鍵も参照してください。
- 「サービス鍵」– 非公開鍵と同じ目的で使用されますが、この鍵はサーバーとサービスによって使用されます。サービス鍵も参照してください。
- 「セッション鍵」 – 一時的な暗号化鍵。2 つの主体の間で使用され、その有効期限は 1 つのログインセッションの期間に制限されます。セッション鍵も参照してください。

仮想プライベートネットワーク (VPN) 暗号化とトンネルを使用して、セキュアな通信を提供するネットワーク。公開ネットワークを通してユーザーを接続します。

関係 `kdc.conf` または `krb5.conf` ファイルに定義される構成変数または関係の 1 つ。

監査トレール すべてのホストから収集した一連の監査ファイル。

監査ファイル バイナリ形式の監査ログ。監査ファイルは、監査ファイルシステム内に個別に格納されます。

監査ポリシー どの監査イベントが記録されるかを決定する設定であり、大域の設定とユーザーごとの設定があります。大域の設定は監査サービスに適用され、一般にどのオプション情報を監査トレールに含めるかを決定します。2 つの設定 `cnt` と `ahlt` は、監査キューがいっぱいになった時点でのシステムの処理を決定します。たとえば、各監査レコードにシーケンス番号を含めるように監査ポリシーを設定できます。

キーストア キーストアは、アプリケーションによる取得のために、パスワード、パスフレーズ、証明書、およびその他の認証オブジェクトを保持します。キーストアはテクノロジ固有にすることも、複数のアプリケーションで使用される場所にすることもできます。

キータブファイル 1 つまたは複数の鍵 (主体) が含まれるキーテーブル。ホストまたはサービスとキータブファイルとの関係は、ユーザーとパスワードの関係と似ています。

基本セット ログイン時にユーザーのプロセスに割り当てられる一連の特権。変更されていないシステムの場合、各ユーザーの初期の継承可能セットはログイン時の基本セットと同じです。

機密性 プライバシを参照してください。

強化 ホストが本来抱えるセキュリティー上の脆弱性を解決するためにオペレーティングシステムのデフォルト構成を変更すること。

許可されたセット プロセスによって使用できる一連の特権。

クライアント 狭義では、`rlogin` を使用するアプリケーションなど、ユーザーの代わりにネットワークサービスを使用するプロセスを指します。サーバー自身が他のサーバーやサービスのクライアントになる場合もあります。

広義では、a) Kerberos 資格を受け取り、b) サーバーから提供されたサービスを利用するホストを指します。

広義では、サービスを使用する主体を指します。

クライアント主体 (RPCSEC_GSS API) RPCSEC_GSS で保護されたネットワークサービスを使用するクライアント (ユーザーまたはアプリケーション)。クライアント主体名は、`rpc_gss_principal_t` 構造体の形式で格納されます。

クロックスキュー Kerberos 認証システムに参加しているすべてのホスト上の内部システムクロックに許容できる最大時間。参加しているホスト間でクロックスキューを超過すると、要求が拒否されます。クロックスキューは、`krb5.conf` ファイルに指定できます。

継承可能セット プロセスが `exec` の呼び出しを通して継承できる一連の特権。

権利 すべての機能を持つスーパーユーザーの代替アカウント。ユーザー権利の管理およびプロセス権利の管理で、組織はスーパーユーザーの特権を分割して、ユーザーまたは役割に割り当てることができます。Oracle Solaris の権利は、カーネル特権、承認、または特定の UID や GID としてプロセスを実行する機能として実装されています。権利は権利プロファイルおよび役割で収集できます。

権利プロファイル プロファイルとも呼ばれます。役割またはユーザーに割り当てることができるセキュリティーオーバーライドの集合。権利プロファイルには、承認、特権、セキュリティー属性が割り当てられたコマンド、および補足プロファイルと呼ばれるその他の権利プロファイルを含めることができます。

権利ポリシー コマンドに関連付けられるセキュリティーポリシー。現在、Oracle Solaris で有効なポリシーは `solaris` です。`solaris` ポリシーでは、特権と拡張特権ポリシー、承認、および `setuid` セキュリティー属性が認識されます。

公開オブジェクト `root` ユーザーによって所有され、すべてのユーザーが読み取ることのできるファイル。たとえば、`/etc` ディレクトリ内のファイルです。

公開鍵技術のポリシー 鍵管理フレームワーク (KMF) におけるポリシーは、証明書の使用を管理します。KMF ポリシーデータベースを使えば、KMF ライブラリによって管理される鍵や証明書の使用に、制約を設けることができます。

公開鍵の暗号化 暗号化スキームの 1 つ。各ユーザーが 1 つの公開鍵と 1 つの非公開鍵を所有します。公開鍵の暗号化では、送信者は受信者の公開鍵を使用してメッセージを暗号化し、受信者は非公開鍵を使用してそれを復号化します。Kerberos サービスは非公開鍵システムです。非公開鍵の暗号化も参照してください。

更新可能チケット 有効期限の長いチケットは、セキュリティーを低下させることがあるため、「更新可能」チケットに指定することができます。更新可能チケットには 2 つの有効期限があります。a) チケットの現在のインスタンスの有効期限と、b) 任意のチケットの最長有効期限です。クライアントがチケットの使用を継続するときは、最初の有効期限が切れる前にチケットの有効期限を更新します。たとえば、すべてのチケットの最長有効期限が 10 時間のときに、あるチケットが 1 時間だけ有効だとします。このチケットを保持するクライアントが 1 時間を超えて使用する場合は、チケットの有効期限を更新する必要があります。チケットが最長有効期限に達すると、チケットの有効期限が自動的に切れ、それ以上更新できなくなります。

コンシューマ Oracle Solaris の暗号化フレームワーク機能では、コンシューマはプロバイダが提供する暗号化サービスのユーザー。コンシューマになりえるものとして、アプリケーション、エンドユーザー、カーネル処理などが挙げられます。Kerberos、IKE、IPsec などはコンシューマの例です。プロバイダの例は、プロバイダを参照してください。

サーバー ネットワーククライアントにリソースを提供する主体。たとえば、システム `central.example.com` に `ssh` で接続する場合、そのシステムが `ssh` サービスを提供するサーバーになります。サービス主体も参照してください。

サーバー主体 (RPCSEC_GSS API) サービスを提供する主体。サーバー主体は、*service*`@`*host* という書式で ASCII 文字列として格納されます。クライアント主体も参照してください。

サービス 1. ネットワーククライアントに提供されるリソース。多くの場合、複数のサーバーから提供されます。たとえば、マシン `central.example.com` に `rlogin` で接続する場合、そのマシンが `rlogin` サービスを提供するサーバーになります。

2. 認証以外の保護レベルを提供するセキュリティーサービス (整合性またはプライバシ)。整合性とプライバシも参照してください。

サービス鍵 サービス主体と KDC によって共有される暗号化鍵。システムの外部に配布されます。鍵も参照してください。

サービス主体 1 つまたは複数のサービスに Kerberos 認証を提供する主体。サービス主体では、プライマリ名はサービス名 (`ftp` など) で、インスタンスはサービスを提供するシステムの完全指定ホスト名になります。ホスト主体、ユーザー主体も参照してください。

最小化 サーバーを稼働させる上で必要な最小限のオペレーティングシステムをインストールすること。サーバーの処理に直接影関係がないソフトウェアはすべて、インストールされないか、あるいはインストール後削除されます。

最少特権 指定されたプロセスにスーパーユーザー権限のサブセットのみを提供するセキュリティーモデル。最小特権モデルでは、通常のユーザーに、ファイルシステムのマウントやファイルの所有権

の変更などの個人の管理タスクを実行できる十分な特権を割り当てます。これに対して、プロセスは、スーパーユーザーの完全な権限 (つまり、すべての特権) ではなく、タスクを完了するために必要な特権のみで実行されます。バッファーオーバーフローなどのプログラミングエラーによる損害を、保護されたシステムファイルの読み取りまたは書き込みやマシンの停止などの重要な機能にはアクセスできない root 以外のユーザーに封じ込めることができます。

最少特権の原則 最少特権を参照してください。

再認証 コンピュータ操作を実行するためにパスワードを指定する際の要件。通常、sudo 操作では再認証が必要です。認証済み権利プロファイルには、再認証が必要なコマンドを含めることができます。認証済み権利プロファイルを参照してください。

シード 乱数生成のスターター (元になる値)。この値から生成が開始されます。このスターターがランダムソースから生じる場合、このシードは「ランダムシード」と呼ばれます。

資格 チケットと照合セッション鍵を含む情報パッケージ。主体の識別情報を認証するときに使用します。チケットとセッション鍵も参照してください。

資格キャッシュ KDC から受信した資格を含むストレージ領域。通常はファイルです。

主体 1. ネットワーク通信に参加する、一意の名前を持つ「クライアントまたはユーザー」あるいは「サーバーまたはサービス」のインスタンス。Kerberos トランザクションでは、主体 (サービス主体とユーザー主体) 間、または主体と KDC の間で対話が行われます。つまり、主体とは、Kerberos がチケットを割り当てることができる一意のエンティティーのことです。主体名、サービス主体、ユーザー主体も参照してください。

2. (RPCSEC_GSS API) クライアント主体、サーバー主体を参照してください。

主体名 1. 主体の名前。書式は、*primary/instance@REALM*。インスタンス、プライマリ、レルムも参照してください。

2.(RPCSEC_GSS API) クライアント主体、サーバー主体を参照してください。

承認 1. Kerberos では、主体がサービスを使用できるかどうか、主体がアクセスできるオブジェクト、各オブジェクトに許可するアクセスの種類を決定するプロセス。

2. ユーザー権利の管理で、役割またはユーザーに割り当てる (権利プロファイルに埋め込む) ことができる一連の操作 (そうしない場合、セキュリティーポリシーによって拒否される) を実行するための権利。承認はカーネルではなく、ユーザーアプリケーションレベルで適用されます。

初期チケット 直接発行されるチケット (既存のチケット許可チケットは使用されない)。パスワードを変更するアプリケーションなどの一部のサービスでは、クライアントが非公開鍵を知っていることを確認するために、「初期」と指定されたチケットを要求することができます。初期チケットを使用した検査は、クライアントが最近認証されたことを証明するときに重要になります。チケット許可チケットの場合は、取得してから時間が経過していることがあります。

信頼できるユーザー ある程度の信頼レベルで管理タスクを実行できるように決定されたユーザー。一般に、管理者は最初に信頼できるユーザーのログインを作成してから、ユーザーの信頼および能力レベルに合致した管理者権利を割り当てます。その後、これらのユーザーはシステムの構成および保守を支援します。*特権ユーザー*とも呼ばれます。

スーパーユーザーモデル コンピュータシステムにおける典型的な UNIX セキュリティーモデル。スーパーユーザーモデルでは、管理者は絶対的なシステム制御権を持ちます。一般に、マシン管理のために 1 人のユーザーがスーパーユーザー (`root`) になり、すべての管理作業を行える状態となります。

スキャンエンジン 既知のウイルスがないかどうかファイルを検査する、外部ホスト上に存在するサードパーティーのアプリケーション。

スレーブ KDC マスター KDC のコピー。マスター KDC のほとんどの機能を実行できます。各レルムには通常、いくつかのスレーブ KDC (と 1 つのマスター KDC) を配置します。KDC、マスター KDCも参照してください。

制限セット プロセスとその子プロセスでどの特権が利用できるかを示す上限。

整合性 ユーザー認証に加えて、暗号チェックサムを使用して、転送されたデータの有効性を提供するセキュリティーサービス。認証、プライバシも参照してください。

責務分離 最少特権の概念の一部。責務分離により、1 人のユーザーが、トランザクションを完了するためのすべての操作を実行または承認することが回避されます。たとえば、RBAC では、セキュリティーオーバーライドの割り当てからログインユーザーの作成を分離できます。1 つの役割がユーザーを作成します。個別の役割により、権利プロファイル、役割、特権などのセキュリティー属性を既存のユーザーに割り当てることができます。

セキュリティーサービス サービスを参照してください。

セキュリティー属性 セキュリティーポリシーをオーバーライドし、スーパーユーザー以外のユーザーによって実行されても成功する管理コマンドを有効にします。スーパーユーザーモデルでは、`setuid root` プログラムと `setgid` プログラムがセキュリティー属性です。これらの属性がコマンドで指定されると、そのコマンドがどのようなユーザーによって実行されているかにかかわらず、コマンドは正常に処理されます。特権モデルでは、セキュリティー属性として `setuid root` プログラムがカーネル特権およびその他の権利によって置き換えられます。特権モデルは、スーパーユーザーモデルと互換性があります。このため、特権モデルは `setuid` プログラムと `setgid` プログラムをセキュリティー属性として認識します。

セキュリティーフレーバ フレーバを参照してください。

セキュリティーポリシー ポリシーを参照してください。

セキュリティーメカニズム メカニズムを参照してください。

セッション鍵 認証サービスまたはチケット認可サービスによって生成される鍵。セッション鍵は、クライアントとサービス間のトランザクションのセキュリティーを保護するために生成されます。セッション鍵の有効期限は、単一のログインセッションに制限されます。鍵も参照してください。

ソフトウェアプロバイダ Oracle Solaris の暗号化フレームワーク機能では、暗号化サービスを提供するカーネルソフトウェアモジュールまたは PKCS #11 ライブラリ。プロバイダも参照してください。

ダイジェスト メッセージダイジェストを参照してください。

単一システムイメージ 単一システムイメージは、同じネームサービスを使用する一連の検査対象システムを記述するために、Oracle Solaris 監査で使用されます。これらのシステムは監査レコードを中央の監査サーバーに送信しますが、その監査サーバー上では、それらのレコードがまるで 1 つのシステムからやってきたかのように、レコードの比較を行えます。

遅延チケット 遅延チケットは、作成されても指定された時点まで有効になりません。このようなチケットは、夜遅く実行するバッチジョブなどのために効果的です。そのチケットは盗まれても、バッチジョブが実行されるまで使用できないためです。遅延チケットは、無効チケットとして発行され、a) 開始時間を過ぎて、b) クライアントが KDC による検査を要求したときに有効になります。遅延チケットは通常、チケット認可チケットの有効期限まで有効です。ただし、その遅延チケットが「更新可能」と指定されている場合、その有効期限は通常、チケット認可チケットの有効期限に設定されます。無効チケット、更新可能チケットも参照してください。

チケット ユーザーの識別情報をサーバーやサービスに安全に渡すために使用される情報パケット。チケットは、単一クライアントと特定サーバー上の特定サービスだけに有効です。チケットには、サービスの主体名、ユーザーの主体名、ユーザーのホストの IP アドレス、タイムスタンプ、チケットの有効期限を定義する値などが含まれます。チケットは、クライアントとサービスによって使用されるランダムセッション鍵を使用して作成されます。チケットは、作成されてから有効期限まで再使用できます。チケットは、最新のオーセンティケータとともに提示されなければ、クライアントの認証に使用することができません。オーセンティケータ、資格、サービス、セッション鍵も参照してください。

チケットファイル 資格キャッシュを参照してください。

デバイスの割り当て ユーザーレベルでのデバイス保護。デバイス割り当ては、一度に 1 人のユーザーだけが使用できるようにデバイスを設定する作業です。デバイスデータは、デバイスが再使用される前に消去されます。誰にデバイス割り当てを許可するかは、承認を使用して制限できます。

デバイスポリシー カーネルレベルでのデバイス保護。デバイスポリシーは、2 つの特権セットとしてデバイスに実装されます。この 1 つはデバイスに対する読み取り権を制御し、もう 1 つはデバイスに対する書き込み権を制御します。ポリシーも参照してください。

転送可能チケット チケットの 1 つ。クライアントがリモートホスト上のチケットを要求するときに使用できます。このチケットを使用すれば、リモートホスト上で完全な認証プロセスを実行する必要がありません。たとえば、ユーザー `david` がユーザー `jennifer` のマシンで転送可能チケットを取得した場合、`david` は自分のマシンにログインできます (新しいチケットを取得する必要はない、自分自身を認証できる)。プロキシ可能チケットも参照してください。

同期監査イベント 監査イベントの大半を占めます。これらのイベントは、システムのプロセスに関連付けられています。失敗したログインなど、あるプロセスに関連付けられた、ユーザーに起因しないイベントは、同期イベントです。

特権 1. 一般に、コンピュータシステム上で通常のユーザーの能力を超える操作を実行する能力または機能。スーパーユーザー特権は、スーパーユーザーに付与されているすべての権利です。特権ユーザーまたは特権アプリケーションは、追加の権利が付与されているユーザーまたはアプリケーションです。

2. Oracle Solaris システムにおいてプロセスに対する個々の権利。特権を使用すると、`root` を使用するよりもきめ細かなプロセス制御が可能です。特権の定義と適用はカーネルで行われます。特権は、*プロセス特権*や*カーネル特権*とも呼ばれます。特権の詳細は、`privileges`(5) のマニュアルページを参照してください。

特権エスカレーション 権利 (デフォルトをオーバーライドして許可する権利を含む) を割り当てられたリソース範囲の外部のリソースへのアクセス権を取得すること。その結果、プロセスは未承認の操作を実行できます。

特権セット 一連の特権。各プロセスには、プロセスが特定の特権を使用できるかどうかを判断する 4 セットの特権があります。詳細は、制限セット、有効セット、許可されたセット、および 継承可能セットを参照してください。

基本セットも、ユーザーのログインプロセスに割り当てられる特権セットです。

特権付きアプリケーション システム制御をオーバーライドできるアプリケーション。このようなアプリケーションは、セキュリティー属性 (特定の UID、GID、承認、特権など) をチェックします。

特権モデル コンピュータシステムにおいてスーパーユーザーモデルより厳密なセキュリティーモデル。特権モデルでは、プロセスの実行に特権が必要です。システムの管理は、管理者が各自のプロセスで与えられている特権に基づいて複数の個別部分に分割できます。特権は、管理者のログインプロセスに割り当てることも、特定のコマンドだけで有効なように割り当てることも可能です。

特権ユーザー コンピュータシステム上で通常ユーザーの権利を超えた権利が割り当てられているユーザー。信頼できるユーザーも参照してください。

特権を認識する 自身のコードでの特権の使用を有効および無効にするプログラム、スクリプト、およびコマンド。本稼動環境では、たとえば、プログラムのユーザーに、その特権をプログラムに追加する権利プロファイルの使用を要求することによって、有効になった特権をプロセスに提供する必要があります。特権の詳細は、`privileges`(5) のマニュアルページを参照してください。

認証 特定の主体の識別情報を検証するプロセス。

認証済み権利プロファイル 権利プロファイル の 1 つ。割り当てられたユーザーまたは役割は、プロファイルから操作を実行する前に、パスワードを入力する必要があります。この動作は、`sudo` の動作に似ています。パスワードが有効である時間の長さは構成可能です。

ネームサービススコープ 特定の役割が操作を許可されている適用範囲。つまり、NIS LDAP などの指定されたネームサービスからサービスを受ける個々のホストまたはすべてのホスト。

ネットワークアプリケーションサーバー ネットワークアプリケーションを提供するサーバー (ftp など)。レルムは、複数のネットワークアプリケーションサーバーで構成されます。

ネットワークポリシー ネットワークトラフィックを保護するためにネットワークユーティリティーで行われる設定。ネットワークセキュリティーについては、『Oracle Solaris 11.2 でのネットワークのセキュリティー保護 』を参照してください。

ハードウェアプロバイダ Oracle Solaris の暗号化フレームワーク機能では、デバイスドライバとそのハードウェアアクセラレータを指します。ハードウェアプロバイダを使用すると、コンピュータシステムから負荷の高い暗号化処理を解放され、その分 CPU リソースをほかの用途に充てることができます。プロバイダも参照してください。

パスフレーズ 非公開鍵がパスフレーズユーザーによって作成されたことを検証するために使用されるフレーズ。望ましいパスフレーズは、10 - 30 文字の長さで英数字が混在しており、単純な文や名前を避けたものです。通信の暗号化と復号化を行う非公開鍵の使用を認証するため、パスフレーズの入力を求めるメッセージが表示されます。

パスワードポリシー パスワードの生成に使用できる暗号化アルゴリズム。パスワードをどれぐらいの頻度で変更すべきか、パスワードの試行を何回まで認めるかといったセキュリティー上の考慮事項など、パスワードに関連した一般的な事柄を指すこともあります。セキュリティーポリシーにはパスワードが必要です。パスワードポリシーでは、AES アルゴリズムを使用してパスワードを暗号化することを要求したり、パスワードの強度に関連したそれ以上の要件を設定したりすることもできます。

非公開鍵 各ユーザー (主体) に与えられ、主体のユーザーと KDC だけが知っている鍵。ユーザー主体の場合、鍵はユーザーのパスワードに基づいています。鍵も参照してください。

非公開鍵の暗号化 非公開鍵の暗号化では、送信者と受信者は同じ暗号化鍵を使用します。公開鍵の暗号化も参照してください。

非同期監査イベント 非同期イベントは、システムイベントの内の少数です。これらのイベントは、プロセスに関連付けられていないため、ブロックした後に起動できるプロセスはありません。たとえば、システムの初期ブートや PROM の開始および終了のイベントは、非同期イベントです。

秘密鍵 非公開鍵を参照してください。

プライバシ セキュリティーサービスの 1 つ。送信データを送信前に暗号化します。プライバシには、データの整合性とユーザー認証も含まれます。認証、整合性、サービスも参照してください。

プライマリ 主体名の最初の部分。インスタンス、主体名、レルムも参照してください。

フレーバ 従来は、「セキュリティーフレーバ」と「認証フレーバ」は、認証のタイプ (AUTH_UNIX、AUTH_DES、AUTH_KERB) を指すフレーバとして、同じ意味を持っていました。RPCSEC_GSS もセキュリティーフレーバですが、これは認証に加えて、整合性とプライバシのサービスも提供します。

プロキシ可能チケット クライアントに代わってクライアント操作を行うためにサービスによって使用されるチケット。このことを、サービスがクライアントのプロキシとして動作するといいます。サービスは、チケットを使用して、クライアントの識別情報を所有できます。このサービスは、プロキシ可能チケットを使

用して、ほかのサービスへのサービスチケットを取得できますが、チケット認可チケットは取得できません。転送可能チケットと異なり、プロキシ可能チケットは単一の操作に対してだけ有効です。転送可能チケットも参照してください。

プロバイダ Oracle Solaris の暗号化フレームワーク機能では、コンシューマに提供される暗号化サービス。プロバイダには、PKCS #11 ライブラリ、カーネル暗号化モジュール、ハードウェアアクセラレータなどがあります。プロバイダは暗号化フレームワークに結合 (プラグイン) されるため、*プラグイン*とも呼ばれます。コンシューマの例は、コンシューマを参照してください。

プロファイルシェル 権利の管理で、役割 (またはユーザー) がコマンド行から、その役割の権利プロファイルに割り当てられた任意の特権付きアプリケーションを実行できるようにするシェル。プロファイルシェルのバージョンは、システム上で使用可能なシェルのバージョン (`bash` の `pfbash` バージョンなど) に対応します。

ホスト ネットワークを通じてアクセス可能なシステム。

ホスト主体 サービス主体のインスタンスの 1 つ (プライマリ名は `host`)。さまざまなネットワークサービス (`ftp`、`rcp`、`rlogin` など) を提供するために設定します。`host/central.example.com@EXAMPLE.COM` はホスト主体の例です。サーバー主体も参照してください。

ポリシー 一般には、意思やアクションに影響を与えたり、これらを決定したりする計画や手続き。コンピュータシステムでは、多くの場合セキュリティーポリシーを指します。実際のサイトのセキュリティーポリシーは、処理される情報の重要度や未承認アクセスから情報を保護する手段を定義する規則セットです。たとえば、セキュリティーポリシーが、システムの監査、使用するデバイスの割り当て、6 週ごとのパスワード変更を要求する場合があります。

Oracle Solaris OS の特定の領域におけるポリシーの実装については、監査ポリシー、暗号化フレームワークにおけるポリシー、デバイスポリシー、Kerberos ポリシー、パスワードポリシー、および 権利ポリシーを参照してください。

マスター KDC 各レルムのメイン KDC。Kerberos 管理サーバー `kadmind` と、認証とチケット認可デーモン `krb5kdc` で構成されます。レルムごとに、1 つ以上のマスター KDC を割り当てる必要があります。また、クライアントに認証サービスを提供する複製 (スレーブ) KDC を任意の数だけ割り当てることができます。

無効チケット まだ使用可能になっていない遅延チケット。無効チケットは、有効になるまでアプリケーションサーバーから拒否されます。これを有効にするには、開始時期が過ぎたあと、TGS 要求で `VALIDATE` フラグをオンにしてクライアントがこのチケットを KDC に提示する必要があります。遅延チケットも参照してください。

メカニズム 1. データの認証や機密性を実現するための暗号化技術を指定するソフトウェアパッケージ。たとえば、Kerberos V5、Diffie-Hellman 公開鍵など。

2. Oracle Solaris の暗号化フレームワーク機能では、特定の目的のためのアルゴリズムの実装。たとえば、認証に適用される DES メカニズム (CKM_DES_MAC など) は、暗号化に適用されるメカニズム (CKM_DES_CBC_PAD) とは別です。

メッセージダイジェスト メッセージダイジェストは、メッセージから計算されるハッシュ値です。ハッシュ値によってメッセージはほぼ一意に識別されます。ダイジェストは、ファイルの整合性を検証するのに便利です。

メッセージ認証コード (MAC) データの整合性を保証し、データの出所を明らかにするコード。MAC は盗聴行為には対応できません。

役割 特権付きアプリケーションを実行するための特別な ID。割り当てられたユーザーだけが引き受けられます。

有効セット プロセスにおいて現在有効である一連の特権。

ユーザー主体 特定のユーザーに属する主体。ユーザー主体のプライマリ名はユーザー名であり、その省略可能なインスタンスは対応する資格の使用目的を説明するために使われる名前です (`jdoe`、`jdoe/admin` など)。「ユーザーインスタンス」とも呼ばれます。サービス主体も参照してください。

ユーザーに起因しない監査イベント 開始した人を特定できない監査イベント。AUE_BOOT イベントなど。

レルム 1. 1 つの Kerberos データベースといくつかの鍵配布センター (KDC) を配置した論理ネットワーク。

2. 主体名の 3 番目の部分。主体名が `jdoe/admin@CORP.EXAMPLE.COM` の場合、レルムは `CORP.EXAMPLE.COM` です。主体名も参照してください。

admin 主体 *username*`/admin` という形式 (`jdoe/admin` など) の名前を持つユーザー主体。通常のユーザー主体より多くの特権 (ポリシーの変更など) を持つことができます。主体名とユーザー主体も参照してください。

AES Advanced Encryption Standard。対称 128 ビットブロックのデータ暗号技術。2000 年の 10 月、米国政府は暗号化標準としてこのアルゴリズムの Rijndael 方式を採用しました。ユーザー主体の暗号化に代わる米国政府の標準として、AES が採用されています。

Blowfish 32 ビットから 448 ビットまでの可変長鍵の対称ブロックの暗号化アルゴリズム。その作成者である Bruce Schneier 氏は、鍵を頻繁に変更しないアプリケーションに効果的であると述べています。

DES Data Encryption Standard。1975 年に開発され、1981 年に ANSI X.3.92 として ANSI で標準化された対称鍵の暗号化方式。DES では 56 ビットの鍵を使用します。

Diffie-Hellman プロトコル 公開鍵暗号化としても知られています。1976 年に Diffie 氏と Hellman 氏が開発した非対称暗号鍵協定プロトコルです。このプロトコルを使用すると、セキュアでない伝達手段で、事前の秘密情報がなくても 2 人のユーザーが秘密鍵を交換できます。Diffie-Hellman は Kerberos で使用されます。

DSA デジタル署名アルゴリズム。512 ビットから 4096 ビットまでの可変長鍵の公開鍵アルゴリズム。米国政府標準である DSS は最大 1024 ビットです。DSA は入力に SHA1 を使用します。

ECDSA Elliptic Curve Digital Signature Algorithm。楕円曲線数学に基づく公開鍵アルゴリズム。ECDSA 鍵サイズは、同じ長さの署名の生成に必要な DSA 公開鍵のサイズより大幅に小さくなります。

FQDN 完全指定形式のドメイン名。`central.example.com` など (単なる `denver` は FQDN ではない)。

GSS-API Generic Security Service Application Programming Interface の略。さまざまなモジュールセキュリティーサービス (Kerberos サービスなど) をサポートするネットワーク層。GSS-API は、セキュリティー認証、整合性、およびプライバシサービスを提供します。認証、整合性、プライバシも参照してください。

KDC 鍵配布センター (Key Distribution Center)。次の 3 つの Kerberos V5 要素で構成されるマシン。

- 主体と鍵データベース
- 認証サービス
- チケット許可サービス

レルムごとに、1 つのマスター KDC と、1 つ以上のスレーブ KDC を配置する必要があります。

Kerberos 認証サービス、Kerberos サービスが使用するプロトコル、または Kerberos サービスの実装に使用されるコード。

Oracle Solaris の Kerberos は、Kerberos V5 認証にほぼ準拠して実装されています。

「Kerberos」と「Kerberos V5」は技術的には異なりますが、Kerberos のドキュメントでは多くの場合、同じ意味で使用されます。

Kerberos (または Cerberus) は、ギリシャ神話において、ハデスの門を警護する 3 つの頭を持つどう猛な番犬のことです。

Kerberos ポリシー Kerberos サービスでのパスワードの使用方法を管理する一連の規則。ポリシーは、主体のアクセスやチケットのパラメータ (有効期限など) を制限できます。

kvno 鍵バージョン番号。特定の鍵に対して、生成順に付けられたシーケンス番号。もっとも大きい kvno が、最新の鍵を示します。

MAC 1. メッセージ認証コード (MAC)を参照してください。

2. 「ラベル付け」とも呼ばれます。政府のセキュリティー用語規定では、MAC は「Mandatory Access Control」の略です。「Top Secret」や「Confidential」というラベルは MAC の例です。MAC と対称をなすものに DAC (Discretionary Access Control) があります。UNIX アクセス権は DAC の 1 例です。

3. ハードウェアにおいては、LAN における一意のシステムアドレス。システムが Ethernet 上に存在する場合は、Ethernet アドレスが MAC に相当します。

MD5 デジタル署名などのメッセージ認証に使用する繰り返し暗号化のハッシュ関数。1991 年に Rivest 氏によって開発されました。その使用は非推奨です。

NTP Network Time Protocol (NTP)。デラウェア大学で開発されたソフトウェア。ネットワーク環境で、正確な時間またはネットワーククロックの同期化を管理します。NTP を使用して、Kerberos 環境のクロックスキューを管理できます。「クロックスキュー」も参照してください。

PAM プラグイン可能認証モジュール (Pluggable Authentication Module)。複数の認証メカニズムを使用できるフレームワーク。認証メカニズムを使用するサービスはコンパイルし直す必要がありません。PAM は、ログイン時に Kerberos セッションを初期化できます。

QOP 保護の品質。整合性サービスまたはプライバシサービスで使用する暗号化アルゴリズムを選択するときに使用されるパラメータの 1 つ。

RBAC Oracle Solaris のユーザー権利管理機能である、役割に基づくアクセス制御。権利を参照してください。

RBAC ポリシー 権利ポリシーを参照してください。

RSA デジタル署名と公開鍵暗号化システムを取得するための方法。その開発者である Rivest 氏、Shamir 氏、Adleman 氏によって 1978 年に最初に公開されました。

SEAM Solaris システム上の Kerberos の初期バージョンに対応する製品名。この製品は、マサチューセッツ工科大学 (MIT) で開発された Kerberos V5 テクノロジに基づいています。SEAM は、現在 Kerberos サービスと呼ばれています。引き続き、MIT バージョンとはわずかに異なります。

Secure Shell セキュリティー保護されていないネットワークを通して、セキュアなリモートログインおよびその他のセキュアなネットワークサービスを使用するための特別なプロトコル。

SHA1 セキュアなハッシュアルゴリズム。メッセージ要約を作成するために 2^{64} 文字以下の長さを入力するときに操作します。SHA1 アルゴリズムは DSA に入力されます。

stash ファイル stash ファイルには、KDC のマスター鍵を暗号化したコピーが含まれます。サーバーがリブートされると、このマスター鍵を使用して KDC が自動的に認証されてから、`kadmind` プロセスと `krb5kdc` プロセスがブートされます。stash ファイルにはマスター鍵が入っているため、このファイルやこのファイルのバックアップは安全な場所に保管する必要があります。暗号が破られると、この鍵を使用して KDC データベースのアクセスや変更が可能になります。

TGS チケット許可サービス。KDC のコンポーネントの 1 つ。チケットを発行します。

TGT チケット認可チケット。KDC によって発行されるチケット。クライアントは、このチケットを使用して、ほかのサービスのチケットを要求することができます。

索引

か

さ

た

ま

や

ら

わ

A

B

C

D

E

F

G

I

K

L

M

N

O

P

R

S

T

U

V