Oracle® Solaris 11.3 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

Part No: E62734
2016 年 1 月

Part No: E62734

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWeb サイト(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc)を参照してください。

Oracle Supportへのアクセス

サポートをご契約のお客様には、My Oracle Supportを通して電子支援サービスを提供しています。詳細情報は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info) か、聴覚に障害のあるお客様は (http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs)を参照してください。

目次

このドキュメントの使用

- **概要** – 正当なファイルの保護、非表示のファイルアクセス権の表示、および不正なファイルの検出を行う方法について説明します。また、Oracle Solaris システム上で一定期間にわたってファイルの整合性を検証する方法についても説明します。
- **対象読者** - システム管理者。
- **必要な知識** - サイトのセキュリティー要件。

製品ドキュメントライブラリ

この製品および関連製品のドキュメントとリソースは http://www.oracle.com/pls/topic/lookup?ctx=E62101-01 で入手可能です。

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 ファイルアクセスの制御

この章では、Oracle Solaris でファイルを保護する方法について説明します。 また、システムに悪影響を与える可能性のあるアクセス権が設定されたファイルからシステムを保護する方法についても説明します。

この章で扱う内容は、次のとおりです。

UNIX アクセス権によるファイル保護

UNIX ファイルアクセス権と ACL を使用して、ファイルをセキュリティー保護できます。スティッキービットが設定されたファイルと、実行可能なファイルは、特殊なセキュリティー対策が必要です。

ファイルの監視と保護を行うコマンド

この表は、ファイルとディレクトリのモニタリングと保護を行うコマンドについて説明したものです。

表 1 ファイルとディレクトリの保護を行うコマンド

コマンド	説明	マニュアルページ
`ls`	ディレクトリ内のファイルとファイル情報を表示します。	ls(1)

コマンド	説明	マニュアルページ
chown	ファイルの所有権を変更します。	chown(1)
chgrp	ファイルのグループ所有権を変更します。	chgrp(1)
chmod	ファイルのアクセス権を変更します。記号モード (文字と記号) または絶対モード (8 進数) を使用して、ファイルのアクセス権を変更できます。	chmod(1)

ファイルとディレクトリの所有権

従来の UNIX ファイルアクセス権では、次に示す 3 つのユーザークラスに所有権を割り当てることができます。

- **ユーザー** – ファイルまたはディレクトリの所有者。通常はファイルを作成したユーザーです。ファイルの所有者は、そのファイルの読み取り、書き込み (ファイルを変更する)、または実行 (コマンドの場合) が行えるユーザーを決定できます。
- **グループ** – ユーザーグループのメンバー。
- **その他** – ファイル所有者ではなくグループメンバーでもないその他の全ユーザー。

ファイルアクセス権の割り当てや変更が行えるのは、通常、そのファイルの所有者だけです。さらに、root アカウントでもファイルの所有権を変更できます。システムのポリシーをオーバーライドするには、例2「自分のファイルの所有権を変更するためのユーザーの有効化」を参照してください。

ファイルは、7 つの形式のいずれかになります。各形式は、次のように記号で示されます。

- (マイナス記号)	テキストまたはプログラム
b	ブロック型特殊ファイル
c	文字型特殊ファイル
d	ディレクトリ
l	シンボリックリンク
s	ソケット
D	ドア
P	名前付きパイプ (FIFO)

UNIX ファイルアクセス権

次の表に、各ユーザークラスに与えることができる、ファイルまたはディレクトリのアクセス権を示します。

表 2 ファイルとディレクトリのアクセス権

記号	アクセス権	オブジェクト	説明
r	読み取り	ファイル	ファイルの内容を開いて読み込むことができます。
r	読み取り	ディレクトリ	ディレクトリ内のファイルを一覧表示できます。
w	書き込み	ファイル	ファイルの内容の変更またはファイルの削除を行えます。
w	書き込み	ディレクトリ	ディレクトリに対してファイルまたはリンクを追加できます。また、ディレクトリ内のファイルまたはリンクの削除も行えます。
x	実行	ファイル	ファイルがプログラムまたはシェルスクリプトの場合、そのファイルを実行できます。また、exec(2) システム呼び出しの 1 つを使用してそのプログラムを実行することもできます。
x	実行	ディレクトリ	ディレクトリ内のファイルを開いたり、実行したりできます。また、ディレクトリを作成し、その下にサブディレクトリを作成できます。
-	拒否	ファイルとディレクトリ	ファイルの読み込み、書き込み、または実行を行うことができません。

これらのファイルアクセス権は、通常のファイルと特殊ファイル (デバイス、ソケット、名前付きパイプ (FIFO) など) の両方に適用されます。

シンボリックリンクには、そのリンクが指すファイルのアクセス権が適用されます。

ディレクトリとそのサブディレクトリ内のファイルは、そのディレクトリに対するファイルアクセス権を制限することで保護できます。ただし、root 役割は、システム上のすべてのファイルとディレクトリにアクセスできることに注意してください。

setuid、setgid、およびスティッキービットを使用する特殊なファイルアクセス権

実行可能ファイルと公開ディレクトリには、 3 種類の特殊なアクセス権 (setuid、setgid、およびスティッキービット) を設定できます。これらのアクセス権を設定すると、その実行可能ファイルを実行するユーザーは、そのファイルの所有者 (またはグループ) の ID を持つことができます。

特殊なアクセス権はセキュリティー上の問題を引き起こすため、設定するときは十分な注意が必要です。たとえば、ユーザーは、ユーザー ID (UID) を `0` (`root` の UID) に設定するプログラムを実行することによって `root` 権限を取得できます。また、すべてのユーザーは、所有するファイルに対して特殊なアクセス権を設定できるため、これもセキュリティー上の問題の原因となります。

`root` 権限を取得するための `setuid` アクセス権や `setgid` アクセス権の承認されていない使用がないかどうか、システムをモニターするようにしてください。疑わしいアクセス権によって、管理プログラムの所有権が `root` や `bin` ではなく一般ユーザーに付与されていることが考えられます。この特殊なアクセス権を使用しているファイルをすべて検索し、リストする方法は、25 ページの「特殊なファイルアクセス権が設定されたファイルを見つける方法」を参照してください。

`setuid` アクセス権

`setuid` アクセス権を実行可能ファイルに設定すると、このファイルを実行するプロセスにはその実行可能ファイルを実行しているユーザーではなく、ファイルの所有者に基づいてアクセス権が与えられます。この特殊なアクセス権を使用すると、通常は所有者しか利用できないファイルやディレクトリにアクセスできます。

たとえば、`passwd` コマンドの `setuid` アクセス権によってユーザーはパスワードを変更できます。次に、`setuid` アクセス権のある `passwd` コマンドの例を示します。

```
-r-sr-sr-x   1 root     sys       56808 Jun 17 12:02 /usr/bin/passwd
```

この特殊なアクセス権にはセキュリティーリスクが存在します。この特殊なアクセス権は、プロセスの実行が終了したあとでも、高度な知識のあるユーザーは `setuid` プロセスによって与えられたアクセス権を維持する手段を見つけることができるため、セキュリティー上の危険が存在します。

注記 - プログラムから、予約済みの UID (0 から 100) で `setuid` アクセス権を使用しても、実効 UID が正しく設定されないことがあります。シェルスクリプトを使用するか、`setuid` アクセス権では予約済み UID を使用しないようにしてください。

`setgid` アクセス権

`setgid`アクセス権は `setuid` アクセス権に似ています。プロセスの実効グループ ID (GID) はファイルを所有するグループに変更され、ユーザーにはそのグループに与えられたアクセス権に基づくアクセス権が与えられます。`/usr/bin/mail` コマンドには、次のように `setgid` アクセス権が設定されています。

```
-r-x--s--x   1 root    mail     71212 Jun 17 12:01 /usr/bin/mail
```

setgid アクセス権がディレクトリに適用されると、このディレクトリ内で作成されたファイルは、ディレクトリを所有するグループに属します。生成するプロセスが所属するグループに含まれるわけではありません。ディレクトリに対する書き込み権および実行権を持つユーザーは、そのディレクトリにファイルを作成できます。ただし、作成したファイルはユーザーのグループではなくディレクトリを所有するグループに割り当てられます。

root 権限を取得するための setgid アクセス権の承認されていない使用がないかどうか、システムをモニターするようにしてください。疑わしいアクセス権によって、このようなプログラムへのグループアクセス権が、root や bin ではなく、意外なグループに与えられることがあります。このようなアクセス権を使用しているファイルをすべて検索し、一覧表示する方法は、25 ページの「特殊なファイルアクセス権が設定されたファイルを見つける方法」を参照してください。

スティッキービット

「スティッキービット」は、ディレクトリ内のファイルを保護するアクセス権ビットです。ディレクトリにスティッキービットが設定されている場合、ファイルを削除できるのは、ファイルの所有者、ディレクトリの所有者、または特権ユーザーのみです。特権ユーザーの例として root ユーザーが挙げられます。スティッキービットにより、ユーザーは /tmp などの公開ディレクトリからほかのユーザーのファイルを削除できなくなります。

```
drwxrwxrwt 7  root  sys   400 Sep  3 13:37 tmp
```

TMPFS ファイルシステム上でスワップファイルを作成するか公開ディレクトリを設定するときには、スティッキービットを手動で設定してください。手順については、例 5「絶対モードによる特殊なファイルアクセス権の設定」を参照してください。

umask のデフォルト値

ファイルやディレクトリを作成するときには、デフォルトのアクセス権が設定されます。このシステムデフォルトは変更が可能です。テキストファイルのアクセス権は 666 であり、すべてのユーザーに読み取り権と書き込み権を与えます。ディレクトリと実行可能ファイルのアクセス権は 777 で、すべてのユーザーに読み取り権、書き込み権、および実行権を与えます。一般に、ユーザーは .bashrc や .kshrc.user などの自分のシェル初期化ファイルでシステムデフォルトをオーバーライドします。管理者は、/etc/profile ファイルでデフォルトを設定することもできます。

umask コマンドによって割り当てられる値は、デフォルトから差し引かれます。この処理には、chmod コマンドでアクセス権を与えるのと同じ方法でアクセス権を拒否する効果があります。たとえば、chmod 022 コマンドはグループとその他のユーザーに

書き込み権を与えますが、umask 022 コマンドはグループとその他のユーザーの書き込み権を拒否します。

次の表に、umask の標準的な値とその値が実行可能ファイルに与える影響を示します。

表 3 各セキュリティーレベルの umask 設定

セキュリティーレベル	umask 設定	許可されないアクセス権
緩やか (744)	022	グループとその他のユーザーによる w
中程度 (751)	026	グループによる w、その他のユーザーによる rw
厳密 (740)	027	グループによる w、その他のユーザーによる rwx
厳しい (700)	077	グループとその他のユーザーによる rwx

umask 値の設定の詳細は、umask(1) のマニュアルページを参照してください。

ファイルアクセス権を設定するモード

chmod コマンドを使用すると、ファイルのアクセス権を変更できます。ファイルやディレクトリのアクセス権を変更するには、root またはファイルまたはディレクトリの所有者になる必要があります。

chmod コマンドを使用して、次のどちらかのモードでアクセス権を設定できます。

- **絶対モード** – 数値を使用してファイルアクセス権を表します。絶対モードを使用してアクセス権を変更するときは、3 つ 1 組のアクセス権を 8 進数で表します。絶対モードは、アクセス権の設定に一番多く使用されている方法です。
- **記号モード** – 文字と記号の組み合わせを使用して、アクセス権を追加したりアクセス権を削除したりします。

次の表に、絶対モードでファイルのアクセス権を設定するための 8 進数値を示します。これらの数字を 3 つ組み合わせて、所有者、グループ、その他のユーザーのファイルアクセス権をこの順に設定します。たとえば、値 644 は、所有者に対して読み取り権と書き込み権を設定し、グループとその他のユーザーに対しては読み取り専用権を設定します。

表 4 絶対モードによるファイルのアクセス権の設定

8 進数値	ファイルのアクセス権	設定されるアクセス権
0	---	なし

8 進数値	ファイルのアクセス権	設定されるアクセス権
1	--x	実行権のみ
2	-w-	書き込み権のみ
3	-wx	書き込み権と実行権
4	r--	読み取り権のみ
5	r-x	読み取り権と実行権
6	rw-	読み取り権と書き込み権
7	rwx	読み取り権、書き込み権、実行権

次の表に、記号モードでファイルアクセス権を設定するための記号を示します。記号では、アクセス権を設定または変更できる対象ユーザー、実行される操作、あるいは割り当てるまたは変更するアクセス権を指定できます。

表 5 記号モードによるファイルのアクセス権の設定

記号	機能	説明
u	<対象ユーザー>	ユーザー (所有者)
g	<対象ユーザー>	グループ
o	<対象ユーザー>	その他のユーザー
a	<対象ユーザー>	すべて
=	オペレータ	割り当て
+	オペレータ	追加
-	オペレータ	削除
r	アクセス権 *(*アクセス権ビット*)*	読み取り
w	アクセス権 *(*アクセス権ビット*)*	書き込み
x	アクセス権 *(*アクセス権ビット*)*	実行
l	アクセス権 *(*アクセス権ビット*)*	強制ロック、setgid ビットはオン、グループ実行ビットはオフ
s	アクセス権 *(*アクセス権ビット*)*	setuid または setgid ビットはオン
t	アクセス権 *(*アクセス権ビット*)*	スティッキービットはオン、その他の実行ビットはオン

機能列に <対象ユーザー> <操作> <アクセス権> の順で、ファイルまたはディレクトリのアクセス権を変更する記号を指定します。

who アクセス権を変更する対象となるユーザーを指定します。

operator 実行する操作を指定します。

permissions 変更するアクセス権を指定します。

絶対モードまたは記号モードで、ファイルに特殊なアクセス権を設定できます。しかし、ディレクトリに `setuid` アクセス権を設定する場合と、ディレクトリからこのアクセス権を削除する場合は、記号モードを使用する必要があります。絶対モードでは、3 つ 1 組のアクセス権の左端に新しい 8 進数値を追加して、特殊なアクセス権を設定します。例5「絶対モードによる特殊なファイルアクセス権の設定」を参照してください。次の表に、ファイルに特殊なアクセス権を設定する 8 進数値を示します。

表 6 絶対モードによる特殊なファイルアクセス権の設定

8 進数値	特殊なファイルアクセス権
1	スティッキービット
2	`setgid`
4	`setuid`

ファイル属性を使用して ZFS ファイルにセキュリティーを追加する

ZFS ファイルシステムでは、セキュリティー関連ファイルに特別な取り扱いのためのマークを付けることができます。ファイル属性は、ローカルファイル、NFS でマウントされたファイル、または CIFS でマウントされたファイルに影響を与えることができます。`chmod(1)` および `ls(1)` のマニュアルページでは、ファイル属性を設定および一覧表示する方法が説明されています。

セキュリティーに影響するファイル属性としては次のものがあります。

- `appendonly` 属性 – ファイルの末尾への追加は許可しますが、既存の内容の変更は防止します。この属性をログファイル上に設定すると、ログファイルエントリの変更を防ぐことができます。この属性を設定するにはプロセスに `PRIV_FILE_FLAG_SET` 特権が必要であり、削除するにはすべての特権が必要です。
- `immutable` 属性 – ファイルの内容の変更または削除を防止します。アクセス時間の更新を除く、ファイルメタデータの変更も防ぎます。ディレクトリ上に設定された場合、この属性はディレクトリとそのファイルの削除を防止します。この属性を設定するにはプロセスに `PRIV_FILE_FLAG_SET` 特権が必要であり、削除するにはすべての特権が必要です。

  例については、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』の「不変性を ZFS ファイルに適用する」を参照してください。
- `nounlink` 属性 – 重要なファイルまたはディレクトリの削除を防止します。ディレクトリ上に設定された場合、この属性はファイルの削除または名前の変更を防止します。この属性を使用すると、アプリケーションの重要なファイルが誤って削除さ

れないようにできます。この属性を設定するにはプロセスに PRIV_FILE_FLAG_SET 特権が必要であり、削除するにはすべての特権が必要です。

- sensitive 属性 – ファイルに PIN やパスワードなどの鍵情報が含まれていることを示します。機密ファイルは、監査レコードに書き込まれません。
- readonly 属性 – CIFS でマウントされたファイルの内容の変更を禁止します。ファイルの所有者はこの属性を設定またはクリアでき、write_attributes アクセス権を持つユーザーまたはグループはこの属性を設定またはクリアできます。

詳細は、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』の第 9 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。

アクセス制御リストによる UFS ファイルの保護

従来の UNIX ファイル保護機能は、ファイルの所有者、ファイルグループ、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。UFS ファイルシステムでは、アクセス制御リスト (ACL) により次のことが可能となり、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。

- ファイル所有者、グループ、その他のユーザー、特定のユーザーおよびグループにファイルアクセス権を定義します
- これらの各カテゴリにデフォルトのアクセス権を定義します

注記 - ZFS ファイルシステム内の ACL と NFSv4 ファイルに対する ACL については、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』の第 9 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。

たとえば、グループ内のすべてのユーザーがファイルを読み取れるようにする場合は、そのファイルにグループの読み取り権を設定すればすみます。ただし、グループ内の 1 人のユーザーだけがそのファイルに書き込めるようにしたい場合は、ACL を使用できます。

UFS ファイルシステムの ACL の詳細は、Oracle Solaris 10 リリース 用の*Solaris* のシステム管理*:* セキュリティーサービスを参照してください。

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

プログラムは、スタック上のデータの読み取りと書き込みを行います。通常、それらはコード用に特別に指定されたメモリーの読み取り専用部分から実行されます。ス

タック上のバッファーをオーバーフローさせる一部の攻撃では、新しいコードをそのスタックに挿入し、プログラムにそれを実行させようとします。スタックメモリーから実行権を削除すると、これらの攻撃が成功するのを防ぐことができます。ほとんどのプログラムは、実行可能スタックを使用せずに正しく機能できます。

プログラムは、スタック実行を明示的にマークまたは防止することができます。プログラム内の `mprotect()` 関数は、スタックを実行可能として明示的にマークします。詳細は、`mprotect(2)` のマニュアルページを参照してください。

スタックが悪意のあるプログラムによって使用されないようにする方法については、『Oracle Solaris 11.3 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』の「悪影響からのプロセスヒープと実行可能スタックの保護」を参照してください。

マウントされたファイルシステムにある実行可能ファイルによってシステムが侵害されないようにするには、`mount` コマンドに `nosetuid` 引数および `noexec` 引数を使用します。詳細は、`mount(1M)` のマニュアルページを参照してください。

ファイルの保護

次の手順では、UNIX アクセス権を持つファイルの保護、セキュリティーリスクのあるファイルの検出、およびそれらのファイルによるシステムの危殆化の防止を行います。

UNIX アクセス権によるファイルの保護

次のタスクマップは、ファイルアクセス権の一覧表示、ファイルアクセス権の変更、特殊なファイルアクセス権によるファイルの保護などのタスク操作について説明した箇所を示しています。

タスク	手順
ファイル情報を表示します。	19 ページの「ファイル情報を表示する方法」
ローカルファイルの所有権を変更します。	20 ページの「ファイルの所有者を変更する方法」 21 ページの「ファイルのグループ所有権を変更する方法」
ローカルファイルのアクセス権を変更します。	21 ページの「ファイルアクセス権を記号モードで変更する方法」 22 ページの「ファイルアクセス権を絶対モードで変更する方法」 23 ページの「特殊なファイルアクセス権を絶対モードで変更する方法」

▼ ファイル情報を表示する方法

ls コマンドを使用して、ディレクトリ内のすべてのファイルに関する情報を表示します。

- **次のコマンドを入力すると、現在のディレクトリ内のすべてのファイルの一覧が長形式で表示されます。**

```
% ls -la
```

-l　　ユーザー所有権、グループ所有権、ファイルのアクセス権などを長形式で表示します。

-a　　ドット (.) で始まる隠しファイルを含め、すべてのファイルを表示します。

ls コマンドのすべてのオプションについては、ls(1) のマニュアルページを参照してください。

例 1　ファイル情報の表示

次の例では、/sbin ディレクトリ内のファイルを部分的に表示しています。

```
% cd /sbin
% ls -l
total 4960
-r-xr-xr-x   1 root   bin      12756 Dec 19  2013 6to4relay
lrwxrwxrwx   1 root   root        10 Dec 19  2013 accept -> cupsaccept
-r-xr-xr-x   1 root   bin      38420 Dec 19  2013 acctadm
-r-xr-xr-x   2 root   sys      70512 Dec 19  2013 add_drv
-r-xr-xr-x   1 root   bin       3126 Dec 19  2013 addgnupghome
drwxr-xr-x   2 root   bin         37 Dec 19  2013 amd64
-r-xr-xr-x   1 root   bin       2264 Dec 19  2013 applygnupgdefaults
-r-xr-xr-x   1 root   bin        153 Dec 19  2013 archiveadm
-r-xr-xr-x   1 root   bin      12644 Dec 19  2013 arp
.
.
.
```

それぞれの行には、ファイルについての情報が次の順で表示されています。

- ファイルの形式 – たとえば、d。ファイル形式の一覧は、10 ページの「ファイルとディレクトリの所有権」を参照してください。
- アクセス権 – たとえば、r-xr-xr-x。詳細は、10 ページの「ファイルとディレクトリの所有権」を参照してください。
- ハードリンクの数 – たとえば、2。
- ファイルの所有者 – たとえば、root。
- ファイルのグループ – たとえば、bin。
- バイト数でのファイルサイズ – たとえば、12644。

- ファイルの作成日時、またはファイルが最後に変更された日時 - たとえば、`Dec 19 2013`。
- ファイルの名前 - たとえば、`arp`。

▼ ファイルの所有者を変更する方法

始める前に ファイルまたはディレクトリの所有者でない場合は、Object Access Management 権利プロファイルが割り当てられている必要があります。公開オブジェクトであるファイルを変更するには、`root` 役割が付与されている必要があります。

詳細は、『Oracle Solaris 11.3 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. **ローカルファイルのアクセス権を表示します。**

```
% ls -l example-file
-rw-r--r--   1 janedoe   staff   112640 May 24 10:49 example-file
```

2. **ファイルの所有者を変更します。**

```
# chown stacey example-file
```

3. **ファイルの所有者が変更されていることを確認します。**

```
# ls -l example-file
-rw-r--r--   1 stacey   staff   112640 May 26 08:50 example-file
```

NFS マウントしたファイルのアクセス権を変更するには、『Oracle Solaris 11.3 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。

例 2 自分のファイルの所有権を変更するためのユーザーの有効化

セキュリティー上の考慮事項 – `rstchown` 変数の設定をゼロに変更するにはそれ相応の理由が必要です。デフォルト設定では、容量の割り当て制限を回避するために、ユーザーは自分のファイルがほかのユーザーの所有になっているときはそれを一覧表示できません。

この例では、`rstchown` 変数の値は、`/etc/system` ファイル内でゼロに設定されます。この設定によりファイルの所有者は、`chown` コマンドを使用してファイルの所有権をほかのユーザーに変更できます。所有者は `chgrp` コマンドを使用し、ファイルのグループ所有権を所有者自身が属していないグループに設定することもできます。変更は、システムのリブート時に適用されます。

```
set rstchown = 0
```

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

▼ ファイルのグループ所有権を変更する方法

始める前に ファイルまたはディレクトリの所有者でない場合は、Object Access Management 権利が割り当てられている必要があります。公開オブジェクトであるファイルを変更するには、root 役割が付与されている必要があります。

詳細は、『Oracle Solaris 11.3 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. **ファイルのグループ所有権を変更します。**

```
% chgrp scifi example-file
```

グループの設定については、『Oracle Solaris 11.3 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」を参照してください。

2. **ファイルのグループ所有権が変更されていることを確認します。**

```
% ls -l example-file
-rw-r--r--   1 stacey   scifi   112640 June 20 08:55  example-file
```

例2「自分のファイルの所有権を変更するためのユーザーの有効化」も参照してください。

▼ ファイルアクセス権を記号モードで変更する方法

次の手順では、ユーザーはそのユーザーが所有するファイルのアクセス権を変更します。

1. **アクセス権を記号モードで変更します。**

```
% chmod who operator permissions filename
```

who	アクセス権を変更する対象となるユーザーを指定します。
operator	実行する操作を指定します。
permissions	変更するアクセス権を指定します。有効な記号のリストについては、表5を参照してください。
filename	ファイルまたはディレクトリを指定します。

2. **ファイルのアクセス権が変更されていることを確認します。**

```
% ls -l filename
```

注記 - ファイルまたはディレクトリの所有者でない場合は、Object Access Management 権利プロファイルが割り当てられている必要があります。公開オブジェクトであるファイルを変更するには、`root` 役割が付与されている必要があります。

例 3 アクセス権を記号モードで変更する

次の例では、所有者がその他のユーザーから読み取り権を削除します。

```
% chmod o-r example-file1
```

次の例では、所有者がユーザー、グループ、およびその他のユーザーに読み取り権と実行権を追加します。

```
% chmod a+rx example-file2
```

次の例では、所有者がグループのメンバーに読み取り権、書き込み権、および実行権を追加します。

```
% chmod g=rwx example-file3
```

▼ ファイルアクセス権を絶対モードで変更する方法

次の手順では、ユーザーはそのユーザーが所有するファイルのアクセス権を変更します。

1. **アクセス権を絶対モードで変更します。**

```
% chmod nnn filename
```

nnn — 所有者、グループ、その他のユーザーのアクセス権をこの順序で表す 8 進数値を指定します。有効な 8 進数値のリストについては、表4を参照してください。

filename — ファイルまたはディレクトリを指定します。

注記 - `chmod` コマンドを使用して、既存の ACL エントリを持つオブジェクトのファイルまたはディレクトリアクセス権を変更すると、それらの ACL エントリも変更される可能性があります。正確な変更は、`chmod` のアクセス権操作の変更と、ファイルシステムの `aclmode` および `aclinherit` プロパティー値に依存します。

詳細は、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』の第 9 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。

2. **ファイルのアクセス権が変更されていることを確認します。**

```
% ls -l filename
```

注記 - ファイルまたはディレクトリの所有者でない場合は、Object Access Management 権利プロファイルが割り当てられている必要があります。公開オブジェクトであるファイルを変更するには、`root` 役割が付与されている必要があります。

例 4 アクセス権を絶対モードで変更する

次の例では、管理者が公開されているディレクトリのアクセス権を 744 (読み取り/書き込み/実行、読み取り専用、読み取り専用) から 755 (読み取り/書き込み/実行、読み取り/実行、読み取り/実行) に変更します。

```
# ls -ld public_dir
drwxr--r--  1 jdoe   staff    6023 Aug  5 12:06 public_dir
# chmod 755 public_dir
# ls -ld public_dir
drwxr-xr-x  1 jdoe   staff    6023 Aug  5 12:06 public_dir
```

次の例では、所有者が実行可能シェルスクリプトのアクセス権を読み取り/書き込みから読み取り/書き込み/実行に変更します。

```
% ls -l my_script
-rw------- 1 jdoe   staff    6023 Aug  5 12:06 my_script
% chmod 700 my_script
% ls -l my_script
-rwx------ 1 jdoe   staff    6023 Aug  5 12:06 my_script
```

▼ 特殊なファイルアクセス権を絶対モードで変更する方法

始める前に ファイルまたはディレクトリの所有者でない場合は、Object Access Management 権利プロファイルが割り当てられている必要があります。公開オブジェクトであるファイルを変更するには、`root` 役割が付与されている必要があります。

詳細は、『Oracle Solaris 11.3 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. **特殊なアクセス権を絶対モードで変更します。**

```
% chmod nnnn filename
```

nnnn ファイルまたはディレクトリのアクセス権を変更する 8 進数値を指定します。一番左端の 8 進数値で、ファイルに特殊なアクセス権を設定します。特殊なアクセス権に有効な 8 進数値のリストについては、表6を参照してください。

filename　　ファイルまたはディレクトリを指定します。

注記 - chmod コマンドを使用して ACL エントリを持つファイルのグループアクセス権を変更する場合、グループアクセス権と ACL マスクの両方が新しいアクセス権に変更されます。新しい ACL マスクのアクセス権は、そのファイル上に ACL エントリを持つ追加ユーザーおよびグループのアクセス権を変更する場合があるので注意が必要です。ls -v コマンドを使用して、すべての ACL エントリに適切なアクセス権が設定されていることを確認してください。詳細は、ls(1) のマニュアルページを参照してください。

---

2. **ファイルのアクセス権が変更されていることを確認します。**

```
% ls -l filename
```

例 5　絶対モードによる特殊なファイルアクセス権の設定

次の例では、管理者が dbprog ファイルに setuid アクセス権を設定します。

```
# chmod 4555 dbprog
# ls -l dbprog
-r-sr-xr-x   1 db     staff        12095 May  6 09:29 dbprog
```

次の例では、管理者が dbprog2 ファイルに setgid アクセス権を設定します。

```
# chmod 2551 dbprog2
# ls -l dbprog2
-r-xr-s--x   1 db     staff       24576 May  6 09:30 dbprog2
```

次の例では、管理者が public_dir ディレクトリにスティッキービットのアクセス権を設定します。

```
# chmod 1777 public_dir
# ls -ld public_dir
drwxrwxrwt   2 jdoe   staff          512 May 15 15:27 public_dir
```

セキュリティーリスクのあるプログラムからの保護

次の手順では、システムでリスクのある実行可能ファイルを検出して、プログラムがプロセスヒープと実行可能スタックを悪用しないようにします。

- 25 ページの「特殊なファイルアクセス権が設定されたファイルを見つける方法」では、setuid ビットが設定されていても、root ユーザーによって所有されていないファイルを検出します。
- 『Oracle Solaris 11.3 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 』の「悪影響からのプロセスヒープと実行可能スタックの保護」では、悪意のあるソフトウェアの攻撃からプログラムを保護します。

▼ 特殊なファイルアクセス権が設定されたファイルを見つける方法

この手順では、プログラムでの setuid および setgid アクセス権が承認なしで使用される可能性を検出します。疑わしい実行可能ファイルによって、所有権が root または bin ではなく、通常のユーザーに与えられることがあります。

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.3 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. **find コマンドを使用して setuid アクセス権が設定されているファイルを検索します。**

   ```
   # find directory -user root -perm -4000 -exec ls -ldb {} \; >/tmp/filename
   ```

find *directory*	指定したディレクトリから始めて、マウントされているすべてのパスを検査します。ディレクトリとしてルート (/)、usr、opt などを指定できます。
-user root	root が所有するファイルを表示します。
-perm -4000	アクセス権が 4000 に設定されているファイルだけを表示します。
-exec ls -ldb	find コマンドの出力を ls -ldb 形式で表示します。ls(1) のマニュアルページを参照してください。
/tmp/*filename*	find コマンドの結果が書き込まれるファイルです。

   詳細は、find(1) のマニュアルページを参照してください。

2. **結果を /tmp/*filename* に出力します。**

   ```
   # more /tmp/filename
   ```

   背景知識については、12 ページの「setuid アクセス権」を参照してください。

例 6 setuid アクセス権が設定されているファイルを検索する

次の例の出力は、rar というグループのユーザーが /usr/bin/rlogin のコピーを作成し、そのアクセス権を root への setuid に設定したことを示しています。この結果、/usr/rar/bin/rlogin プログラムは root アクセス権で実行されます。

/usr/rar ディレクトリを調べ、/usr/rar/bin/rlogin コマンドを削除したあと、管理者は find コマンドの出力をアーカイブします。

```
# find /usr -user root -perm -4000 -exec ls -ldb {} \; > /var/tmp/ckprm
# cat /var/tmp/ckprm
-rwsr-xr-x  1 root   sys   32432  Jul 14 14:14  /usr/bin/atq
-rwsr-xr-x  1 root   sys   32664  Jul 14 14:14  /usr/bin/atrm
-rwsr-xr-x  1 root   bin   82836  Jul 14 14:14  /usr/bin/cdrw
-r-sr-xr-x  1 root   sys   41448  Jul 14 14:14  /usr/bin/chkey
-r-sr-xr-x  1 root   bin    7968  Jul 14 14:14  /usr/bin/mailq
-r-sr-sr-x  1 root   sys   45364  Jul 14 14:14  /usr/bin/passwd
-rwsr-xr-x  1 root   bin   37740  Jul 14 14:14  /usr/bin/pfedit
-r-sr-xr-x  1 root   bin   51472  Jul 14 14:14  /usr/bin/rcp
---s--x---  1 root   rar   41592  Jul 24 16:14  /usr/rar/bin/rlogin
-r-s--x--x  1 root   bin  213092  Jul 14 14:14  /usr/bin/sudo
-r-sr-xr-x  4 root   bin   24056  Jul 14 14:14  /usr/bin/uptime
-r-sr-xr-x  1 root   bin   79540  Jul 14 14:14  /usr/bin/xlock
# mv /var/tmp/ckprm /var/share/sysreports/ckprm
```

♦ ♦ ♦ 第 2 章

2 BART を使用したファイル整合性の検証

この章では、ファイル整合性ツール BART について説明します。BART は、一定期間にわたってシステム上のファイルの整合性を検証できるコマンド行ツールです。この章で扱う内容は、次のとおりです。

BART について

BART は、暗号化強度チェックサムとファイルシステムメタデータを使用して変更を判定する、ファイル整合性の走査および報告ツールです。BART は、壊れたファイルや異常なファイルを識別することによって、セキュリティー違反を検出したり、システム上のパフォーマンスの問題をトラブルシューティングしたりするのに役立ちます。BART を使用すると、配備されたシステム上にインストールされているファイルの相違が容易に、かつ確実に報告されるため、システムのネットワークの管理コストを削減できます。

BART を使用することで管理者は、既知のベースラインと比較し、ファイルレベルで見てどのような変化がシステムに起きたかを確認できます。BART を使用して、完全にインストールおよび構成されたシステムからベースラインまたは制御目録を作成します。作成したこのベースラインをあとでシステムのスナップショットと比較すれば、システムのインストール後にシステムで発生したファイルレベルの変化を示すレポートが生成されます。

BART の機能

BART では、強力で、かつ柔軟性の高い単純な構文が使用されます。このツールを使用すると、一定期間にわたって特定のシステム上のファイル変更を追跡できます。また、類似したシステム間のファイルの違いを追跡することもできます。このような比

較は、壊れたファイルや異常なファイル、またはソフトウェアが期限切れになったシステムを見つけるのに役立ちます。

BART には、ほかにも次のようなメリットや利用法があります。

- どのファイルをモニターするかを指定できます。たとえば、ローカルなカスタマイズをモニターできます。これにより、ソフトウェアを容易に、かつ効率的に再構成できるようになります。
- システム性能の問題をトラブルシューティングできます。

BART コンポーネント

BART は、2 つの主なファイルである目録 と比較ファイル (つまり、レポート) を作成します。オプションの規則ファイルを使用すると、目録やレポートをカスタマイズできます。

BART 目録

目録は、特定の時点でのシステムのファイルレベルのスナップショットです。目録にはファイルの属性に関する情報が含まれており、これには、いくつかの一意に識別する情報 (チェックサムなど) を含めることができます。`bart create` コマンドのオプションは、特定のファイルやディレクトリをターゲットにすることができます。29 ページの「BART 規則ファイル」で説明されているように、規則ファイルは、よりきめ細かなフィルタリングを提供できます。

注記 - デフォルトでは、BART は、ルート (/) ディレクトリの下にあるすべての ZFS ファイルシステムをカタログ化します。その他のファイルシステムタイプ (NFS ファイルシステムや TMPFS ファイルシステムなど)、およびマウントされた CD-ROM がカタログ化されます。

システムの目録は、Oracle Solaris の初期インストールの直後に作成できます。また、サイトのセキュリティーポリシーを満たすようにシステムを構成したあとに目録を作成することもできます。このタイプの制御目録によって、あとの比較のためのベースラインが提供されます。

ベースライン目録を使用すると、一定期間にわたって同じシステム上のファイル整合性を追跡できます。これはまた、ほかのシステムと比較するための基礎としても使用できます。たとえば、ネットワーク上のほかのシステムのスナップショットを作成したあと、これらの目録をベースライン目録と比較することができます。報告されたファイルの相違は、ほかのシステムをベースラインシステムと同期化するために何を行う必要があるかを示しています。

目録の形式については、40 ページの「BART 目録のファイル形式」を参照してください。目録を作成するには、`How to Create a Control Manifest`の説明に従って、31 ページの「制御目録を作成する方法」 コマンドを使用します。

BART レポート

BART レポートは、2 つの目録間のファイルごとの相違を一覧表示します。相違とは、両方の目録に対してカタログ化されている特定のファイルの任意の属性への変更のことです。また、ファイルエントリの追加または削除も相違と見なされます。

有効な比較のためには、2 つの目録が同じファイルシステムをターゲットにする必要があります。また、同じオプションと規則ファイルを使用して目録を作成および比較することも必要です。

レポートの形式については、43 ページの「BART レポート」を参照してください。レポートを作成するには、`How to Compare Manifests for the Same System Over Time`の説明に従って、34 ページの「一定期間内で同一システムの目録を比較する方法」 コマンドを使用します。

BART 規則ファイル

BART 規則ファイルは、追加または除外のために特定のファイルやファイル属性をフィルタリングしたり、ターゲットにしたりするために作成するファイルです。そのあと、BART 目録およびレポートを作成するときにこのファイルを使用します。目録を比較する場合、目録間の相違を報告するには規則ファイルが役立ちます。

注記 - 規則ファイルを使用して目録を作成する場合は、同じ規則ファイルを使用して比較目録を作成する必要があります。また、目録を比較するときにも、この規則ファイルを使用する必要があります。そうしないと、レポートによって、多くの無効な相違が一覧表示されます。

規則ファイルを使用してシステム上の特定のファイルやファイル属性をモニターする場合は、計画が必要です。規則ファイルを作成する前に、システム上のどのファイルおよびファイル属性をモニターするかを決定してください。

ユーザーエラーの結果として、規則ファイルには、構文エラーやその他のあいまいな情報が含まれている場合もあります。規則ファイルにエラーが含まれている場合は、これらのエラーも報告されます。

規則ファイルの形式については、41 ページの「BART 規則ファイルの書式」および `bart_rules(4)` のマニュアルページを参照してください。規則ファイルを作成する

には、39 ページの「規則ファイルを使用して BART レポートをカスタマイズする方法」を参照してください。

BART の使用について

bart コマンドは、目録を作成および比較するために使用されます。このコマンドは、どのユーザーでも実行できます。ただし、ユーザーがファイルのカタログ化やモニターを行うことができるのは、自分がアクセス権を持っているファイルだけです。そのため、ユーザーやほとんどの役割は自分のホームディレクトリ内のファイルを有効にカタログ化できますが、root アカウントは、システムファイルを含むすべてのファイルをカタログ化できます。

BART のセキュリティー上の考慮事項

BART 目録およびレポートは、だれでも読み取ることができます。BART の出力に機密情報が含まれている可能性がある場合は、出力を保護するための適切な対策を取ってください。たとえば、アクセス権が制限された出力ファイルを生成するオプションや、出力ファイルを保護されたディレクトリ内に配置するオプションを使用します。

BART の使用

タスク	説明	手順
BART 目録を作成します。	システムにインストールされているファイルごとに情報の一覧表示を生成します。	31 ページの「制御目録を作成する方法」
カスタム BART 目録を作成します。	システムにインストールされている特定のファイルに関する情報の一覧を生成します。	33 ページの「目録をカスタマイズする方法」
BART 目録を比較します。	一定期間における同一システムの変化を比較するレポートを生成します。 または、1 つまたは複数のシステムを制御システムと比較するレポートを生成します。	34 ページの「一定期間内で同一システムの目録を比較する方法」 36 ページの「異なるシステムからの目録の比較方法」
(オプション) BART レポートをカスタマイズします。	次に示す方法のいずれかでカスタム BART レポートを生成します。 ■ 属性を指定します ■ 規則ファイルを使用する	38 ページの「ファイル属性を指定して BART レポートをカスタマイズする方法」 39 ページの「規則ファイルを使用して BART レポートをカスタマイズする方法」

▼ 制御目録を作成する方法

この手順では、比較用のベースライン (制御) 目録の作成方法について説明します。このタイプの目録は、中央のイメージから多数のシステムをインストールするときに使用してください。または、インストールが同一であることを確認したい場合は、このタイプの目録を使用して比較を実行してください。制御目録の詳細は、28 ページの「BART 目録」を参照してください。形式の規則を理解するには、例7「BART 目録の形式の説明」を参照してください。

注記 - ネットワーク化されたファイルシステムをカタログ化しようとしないでください。BART を使用して、ネットワーク化されたファイルシステムをモニターすると、ほとんど価値のない目録を生成するために大量のリソースが消費されます。

始める前に `root` 役割になる必要があります。詳細は、『Oracle Solaris 11.3 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. **サイトのセキュリティー要件を満たすように Oracle Solaris システムをカスタマイズしたあと、制御目録を作成し、その出力をファイルにリダイレクトします。**

   `# bart create` *options* `>` *control-manifest*

   `-R` 目録の検査対象としてルートディレクトリを指定します。規則で指定されるパスはすべて、このディレクトリからの相対パスとなるように変換されます。目録で報告されるパスはすべて、このディレクトリからの相対パスとなります。

   `-I` カタログ化される個々のファイルの一覧 (コマンド行上で指定されるか、あるいは標準入力から読み取られる) を受け入れます。

   `-r` この目録の規則ファイルの名前です。- 引数を指定すると、規則ファイルが標準入力から読み取られます。

   `-n` ファイルリストに挙げられたすべての通常ファイルの署名を無効にします。このオプションは、パフォーマンスを上げる目的で使用できます。また、(システムログファイルの場合のように) ファイルリストの内容が変わる予定がある場合にも使用できます。

2. **目録の内容を確認します。**
   形式の説明については、例7「BART 目録の形式の説明」を参照してください。

3. **(オプション) 目録を保護します。**
   システム目録を保護するための 1 つの方法として、システム目録を `root` アカウントだけがアクセスできるディレクトリ内に配置します。

```
# mkdir /var/adm/log/bartlogs
# chmod 700 /var/adm/log/bartlogs
# mv control-manifest /var/adm/log/bartlogs
```

目録にわかりやすい名前をつけてください。たとえば、`mach1-120313` のように、目録が作成されたシステム名と日付を使用します。

例 7 BART 目録の形式の説明

この例では、出力例のあとに目録の形式の説明が表示されています。

```
# bart create
! Version 1.1
! HASH SHA256
! Saturday, September 07, 2013 (22:22:27)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/ D 1024 40755 user::rwx,group::r-x,mask:r-x,other:r-x
3ebc418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090 0 0
.
.
.
/zone D 512 40755 user::rwx group::r-x,mask:r-x,other:r-x 3f81e892
154de3e7bdfd0d57a074c9fae0896a9e2e04bebfe5e872d273b063319e57f334 0 0
.
.
.
```

各目録は、ヘッダーとファイルエントリで構成されています。各ファイルエントリは、ファイルタイプに応じて 1 行になります。たとえば、前の出力にある各ファイルエントリの場合、タイプ `F` はファイルを指定し、タイプ `D` はディレクトリを指定します。また、サイズ、内容、ユーザー ID、グループ ID、およびアクセス権も表示されます。特殊文字を正しく処理するため、出力内のファイルエントリはエンコードされたバージョンのファイル名でソートされます。この結果、すべてのエントリがファイル名をキーにして昇順に並べられます。標準でないファイル名 (改行文字やタブ文字が埋め込まれたファイル名など) の場合はすべて、ソート前に非標準文字が引用符で囲まれます。

`!` で始まる行には、目録についてのメタデータが示されてます。目録バージョン行には、目録の仕様バージョンが示されます。ハッシュ行には、使用されたハッシュメカニズムが示されます。チェックサムとして使用される `SHA256` ハッシュの詳細は、sha2(3EXT) のマニュアルページを参照してください。

日付行には、目録が作成された日付が日付形式で示されます。`date(1)` のマニュアルページを参照してください。一部の行は、目録比較ツールによって無視されます。無視される行には、メタデータ、空行、空白のみで構成された行、および `#` で始まるコメントが含まれます。

▼ 目録をカスタマイズする方法

目録は、次に示す方法のいずれかでカスタマイズできます。

- サブツリーを指定する

  個々のサブツリーの指定は、選択された重要なファイル (/etc ディレクトリ内のすべてのファイルなど) への変更をモニターするための効率的な方法です。
- ファイル名を指定する

  ファイル名の指定は、特に重要なファイル (データベースアプリケーションを構成および実行するファイルなど) をモニターするための効率的な方法です。
- 規則ファイルを使用する

  規則ファイルを使用して目録の作成と比較を行うと、複数のファイルまたはサブツリーの複数の属性を指定するという柔軟性が得られます。コマンド行からは、目録またはレポート内のすべてのファイルに適用されるグローバルな属性定義を指定できます。規則ファイルからは、グローバルには適用されない属性を指定できます。

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.3 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. **どのファイルのカタログ化とモニターを行うかを決定します。**

2. **次のオプションのいずれかを使用して、カスタムマニフェストを作成します。**

   - サブツリーを指定する。

     ```
     # bart create -R subtree
     ```
   - ファイル名 (1 つまたは複数) を指定する。

     ```
     # bart create -I filename...
     ```

     例:

     ```
     # bart create -I /etc/system /etc/passwd /etc/shadow
     ```
   - 規則ファイルを使用する。

     ```
     # bart create -r rules-file
     ```

3. **目録の内容を確認します。**

4. **(オプション) 目録を将来の使用のために保護されたディレクトリ内に保存します。**

   例については、31 ページの「制御目録を作成する方法」のステップ 3を参照してください。

ヒント - 規則ファイルを使用した場合は、目録とともに規則ファイルを保存します。有効な比較のためには、この規則ファイルを使用して比較を実行する必要があります。

▼ 一定期間内で同一システムの目録を比較する方法

一定期間にわたって目録を比較することによって、壊れたファイルや異常なファイルを見つけたり、セキュリティー違反を検出したり、システム上のパフォーマンスの問題をトラブルシューティングしたりすることができます。

始める前に `root` 役割になる必要があります。詳細は、『Oracle Solaris 11.3 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. **システム上でモニターするファイルの制御目録を作成します。**

   ```
   # bart create -R /etc > control-manifest
   ```

2. **(オプション) 目録を将来の使用のために保護されたディレクトリ内に保存します。**
   例については、31 ページの「制御目録を作成する方法」のステップ 3を参照してください。

3. **あとで、制御目録と同一の目録を準備します。**

   ```
   # bart create -R /etc > test-manifest
   ```

4. **2 番目の目録を保護します。**

   ```
   # mv test-manifest /var/adm/log/bartlogs
   ```

5. **2 つの目録を比較します。**
   同じコマンド行オプションと規則ファイルを使用して、それらを作成するために使用した目録を比較します。

   ```
   # bart compare options control-manifest test-manifest > bart-report
   ```

6. **BART レポートを調べ、異常がないかを確認します。**

例 8 一定期間にわたる同じシステムに対するファイル変更の追跡

この例は、一定期間にわたって `/etc` ディレクトリの変更を追跡する方法を示しています。このタイプの比較を使用すると、攻撃されたシステム上の重要なファイルを見つけることができます。

- 制御目録を作成します。

  ```
  # cd /var/adm/logs/manifests
  ```

```
# bart create -R /etc > system1.control.090713
! Version 1.1
! HASH SHA256
! Saturday, September 07, 2013 (11:11:17)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/.cpr_config F 2236 100644 owner@:read_data/write_data/append_data/read_xattr/wr
ite_xattr/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchr
onize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:all
ow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4e271c59 0 0 3ebc418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090
/.login F 1429 100644 owner@:read_data/write_data/append_data/read_xattr/write_x
attr/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchronize
:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow,ev
eryone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4bf9d6d7 0 3 ff6251a473a53de68ce8b4036d0f569838cff107caf1dd9fd04701c48f09242e
.
.
.
```

- あとで、同じコマンド行オプションを使用して、テスト目録を作成します。

```
# bart create -R /etc > system1.test.101013
Version 1.1
! HASH SHA256
! Monday, October 10, 2013 (10:10:17)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/.cpr_config F 2236 100644 owner@:read_data/write_data/append_data/read_xattr/wr
ite_xattr/read_attributes/write_attributes/read_acl/write_acl/write_owner/synchr
onize:allow,group@:read_data/read_xattr/read_attributes/read_acl/synchronize:all
ow,everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize:allow
4e271c59 0 0 3ebc418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090
.
```

.
.

- 目録を比較します。

```
# bart compare system1.control.090713 system1.test.101013
/security/audit_class
mtime  4f272f59
```

この出力は、`audit_class` ファイル上の変更時間が、制御目録が作成されたあとに変化したことを示しています。この変化が予期しないものである場合は、さらに調査を行うことができます。

▼ 異なるシステムからの目録の比較方法

異なるシステムの目録を比較することによって、システムが同様にインストールされているかどうか、または同期してアップグレードされているかどうかを判定できます。たとえば、システムを特定のセキュリティーターゲットにカスタマイズした場合は、この比較によって、そのセキュリティーターゲットを表す目録と、その他のシステムの目録の間の相違がすべて見つかります。

始める前に `root` 役割になる必要があります。詳細は、『Oracle Solaris 11.3 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. **制御目録を作成します。**

   ```
   # bart create options > control-manifest
   ```

   これらのオプションについては、`bart`(1M) のマニュアルページを参照してください。

2. **(オプション) 目録を将来の使用のために保護されたディレクトリ内に保存します。**
   例については、31 ページの「制御目録を作成する方法」のステップ 3を参照してください。

3. **テストシステム上で、同じ `bart` オプションを使用して目録を作成します。**

   ```
   # bart create options > test1-manifest
   ```

4. **(オプション) 目録を将来の使用のために保護されたディレクトリ内に保存します。**

5. **比較を実行するには、目録を中央の場所にコピーします。**
   例:

   ```
   # cp control-manifest /net/test-server/var/adm/logs/bartlogs
   ```

   テストシステムが NFS マウントしたシステムでない場合は、`sftp` または別の信頼できる手段を使用して、目録を中央の場所にコピーします。

6. **目録を比較し、その出力をファイルにリダイレクトします。**

```
# bart compare control-manifest test1-manifest > test1.report
```

7. **BART レポートを調べ、異常がないかを確認します。**

例 9 /usr/bin ディレクトリ内の疑わしいファイルの識別

この例では、2 つのシステム上の /usr/bin ディレクトリの内容を比較します。

- 制御目録を作成します。

```
# bart create -R /usr/bin > control-manifest.090713
! Version 1.1
! HASH SHA256
! Saturday, September 07, 2013 (11:11:17)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/2to3 F 105 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attribut
es/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:read
_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:re
ad_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4bf9d261 0
2 154de3e7bdfd0d57a074c9fae0896a9e2e04bebfe5e872d273b063319e57f334
/7z F 509220 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attribu
tes/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:rea
d_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:r
ead_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4dadc48a 0
2 3ecd418eb5be3729ffe7e54053be2d33ee884205502c81ae9689cd8cca5b0090
...
```

- 制御システムと比較する各システムで同一の目録を作成します。

```
# bart create -R /usr/bin > system2-manifest.101013
! Version 1.1
! HASH SHA256
! Monday, October 10, 2013 (10:10:22)
# Format:
#fname D size mode acl dirmtime uid gid
#fname P size mode acl mtime uid gid
#fname S size mode acl mtime uid gid
#fname F size mode acl mtime uid gid contents
#fname L size mode acl lnmtime uid gid dest
```

```
#fname B size mode acl mtime uid gid devnode
#fname C size mode acl mtime uid gid devnode
/2to3 F 105 100555 owner@:read_data/read_xattr/write_xattr/execute/read_attribut
es/write_attributes/read_acl/write_acl/write_owner/synchronize:allow,group@:read
_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow,everyone@:re
ad_data/read_xattr/execute/read_attributes/read_acl/synchronize:allow 4bf9d261 0
2 154de3e7bdfd0d57a074c9fae0896a9e2e04bebfe5e872d273b063319e57f334
...
```

- これらの目録を同じ場所にコピーします。

```
# cp control-manifest.090713 /net/system2.central/bart/manifests
```

- 目録を比較します。

```
# bart compare control-manifest.090713 system2.test.101013 > system2.report
/su:
gid  control:3  test:1
/ypcat:
mtime  control:3fd72511  test:3fd9eb23
```

この出力は、/usr/bin ディレクトリ内の su ファイルのグループ ID が、制御システムのグループ ID と同じでないことを示しています。この情報は、テストシステム上に別のバージョンのソフトウェアがインストールされたことを示している可能性があります。GID が変更されているため、だれかがファイルを改ざんしたことが可能性として考えられます。

▼ ファイル属性を指定して BART レポートをカスタマイズする方法

この手順は、既存の目録の出力を特定のファイル属性に関してフィルタリングするために役立ちます。

始める前に root 役割になる必要があります。詳細は、『Oracle Solaris 11.3 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. **どのファイル属性をチェックするかを決定します。**

2. **チェックされるファイル属性を含む 2 つの目録を比較します。**

   例:

   ```
   # bart compare -i lnmtime,mtime control-manifest.121513 \
     test-manifest.010514 > bart.report.010514
   ```

   各ファイル属性を区切るには、コマンド行構文でコンマを使用します。

3. **BART レポートを調べ、異常がないかを確認します。**

▼ 規則ファイルを使用して BART レポートをカスタマイズする方法

規則ファイルを使用すると、目的とする特定のファイルやファイル属性に関して BART 目録をカスタマイズできます。デフォルトの BART 目録に対して異なる規則ファイルを使用すると、同じ目録に対して異なる比較を実行できます。

始める前に `root` 役割になる必要があります。詳細は、『Oracle Solaris 11.3 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. **どのファイルおよびファイル属性をモニターするかを決定します。**
2. **適切な指示語を含む規則ファイルを作成します。**
3. **作成した規則ファイルを使用して制御目録を作成します。**

   ```
   # bart create -r myrules1-file > control-manifest
   ```

4. **(オプション) 目録を将来の使用のために保護されたディレクトリ内に保存します。**
   例については、31 ページの「制御目録を作成する方法」のステップ 3を参照してください。
5. **別のシステム上で同一の目録を作成します。これをあとで、またはその両方で行います。**

   ```
   # bart create -r myrules1-file > test-manifest
   ```

6. **同じ規則ファイルを使用して目録を比較します。**

   ```
   # bart compare -r myrules1-file control-manifest test-manifest > bart.report
   ```

7. **BART レポートを調べ、異常がないかを確認します。**

例 10 規則ファイルを使用した BART 目録および比較レポートのカスタマイズ

次の規則ファイルは、`bart create` コマンドに、`/usr/bin` ディレクトリ内のファイルのすべての属性を一覧表示するよう指示します。さらに、この規則ファイルは `bart compare` コマンドに、同じディレクトリ内のサイズと内容の変更のみを報告するよう指示します。

```
# Check size and content changes in the /usr/bin directory.
# This rules file only checks size and content changes.
```

```
# See rules file example.

IGNORE all
CHECK size contents
/usr/bin
```

- 作成した規則ファイルを使用して制御目録を作成します。

  ```
  # bart create -r usrbinrules.txt > usr_bin.control-manifest.121013
  ```

- `/usr/bin` ディレクトリへの変更をモニターする場合は常に、同一の目録を準備します。

  ```
  # bart create -r usrbinrules.txt > usr_bin.test-manifest.121113
  ```

- 同じ規則ファイルを使用して目録を比較します。

  ```
  # bart compare -r usrbinrules.txt usr_bin.control-manifest.121013 \
  usr_bin.test-manifest.121113
  ```

- `bart compare` コマンドの出力を調べます。

  ```
   /usr/bin/gunzip:  add
  /usr/bin/ypcat:
  delete
  ```

前の出力は、`/usr/bin/ypcat` ファイルが削除されたこと、および `/usr/bin/gunzip` ファイルが追加されたことを示しています。

BART 目録、規則ファイル、およびレポート

このセクションでは、BART が使用して作成するファイルの形式について説明します。

BART 目録のファイル形式

目録ファイルの各エントリは、ファイルタイプに応じて単一の行に示されます。各エントリは、ファイルの名前である *fname* で始まります。ファイル名に使用された特殊文字の解析によって起きる問題を防ぐため、ファイル名はエンコードされます。詳細は、41 ページの「BART 規則ファイルの書式」を参照してください。

後続のフィールドは、次のファイル属性を表します。

type
: ファイルの種類であり、次のような値となります。
  - B: ブロックデバイスノード
  - C: キャラクタデバイスノード

- D: ディレクトリ
- F: ファイル
- L: シンボリックリンク
- P: パイプ
- S: ソケット

size ファイルサイズ (バイト)。

mode ファイルのアクセス権を示す 8 進の数値。

acl ファイルの ACL 属性。ACL 属性を持つファイルの場合、`acltotext()` の出力が入ります。

uid このエントリの所有者の、数値で示したユーザー ID。

gid このエントリの所有者の、数値で示したグループ ID。

dirmtime ディレクトリに最後に変化が起きた時点。1970 年 1 月 1 日の 00:00:00 UTC から数えた秒数で示されます。

lnmtime リンクに最後に変化が起きた時点。1970 年 1 月 1 日の 00:00:00 UTC から数えた秒数で示されます。

mtime ファイルに最後に変化が起きた時点。1970 年 1 月 1 日の 00:00:00 UTC から数えた秒数で示されます。

contents ファイルのチェックサム値。この属性が指定されるのは通常ファイルのみです。コンテキストのチェックを無効にした場合や、チェックサムを計算できない場合、このフィールドの値は - になります。

dest シンボリックリンクのリンク先。

devnode デバイスノードの値。この属性が使用されるのは、キャラクタデバイスファイルとブロックデバイスファイルのみです。

詳細は、`bart_manifest(4)` のマニュアルページを参照してください。

BART 規則ファイルの書式

規則ファイルは、どのファイルを目録に含めるかや、どのファイル属性を目録またはレポートに含めるかを指定する行で構成されたテキストファイルです。#、空の行、空白を含む行は、ツールが無視します。

入力ファイルには、次に示す 3 種類の指示語が指定されます。

- オプションのパターンマッチング修飾子を持つ subtree 指示語
- `CHECK` 指示語
- `IGNORE` 指示語

例 11 規則ファイルの書式

```
<Global CHECK/IGNORE Directives>
<subtree1> [pattern1..]
<IGNORE/CHECK Directives for subtree1>

<subtree2> [pattern2..]
<subtree3> [pattern3..]
<subtree4> [pattern4..]
<IGNORE/CHECK Directives for subtree2, subtree3, subtree4>
```

注記 - すべての指示語が順番に読み取られます。あとの指示語で前の指示語をオーバーライドすることができます。

subtree 指示語は絶対パス名で始まり、そのあとに 0 個以上のパターンマッチング文が続く必要があります。

BART 規則ファイルの属性

`CHECK` 文と `IGNORE` 文は、どのファイル属性を追跡または無視するかを定義します。各目録を開始するメタデータによって、ファイルタイプごとの属性のキーワードが一覧表示されます。例7「BART 目録の形式の説明」を参照してください。

`all` キーワードは、すべてのファイル属性を示します。

BART 引用構文

BART が規則ファイルに使用する記述言語は、標準に準拠していないファイル名を表現する標準の UNIX 引用構文です。埋め込まれたタブ、スペース、改行、特殊文字は、ツールがファイル名を読み取ることができるようにそれらの 8 進形式にエンコードされます。この変動的な引用構文では、埋め込みのキャリッジリターンを含むファイル名などがコマンドパイプラインで正しく処理されません。規則記述言語を使用することで、シェル構文だけでは表現が難しい効率の悪い複雑なファイル名フィルタリング基準を表現できます。

詳細は、`bart_rules(4)` のマニュアルページを参照してください。

BART レポート

デフォルトモードでは、BART レポートは、変更されたディレクトリのタイムスタンプ (dirmtime) を除く、システム上にインストールされたすべてのファイルをチェックします。

```
CHECK all
IGNORE dirmtime
```

規則ファイルを指定すると、汎用指示語である CHECK all と IGNORE dirmtime がこの順で規則ファイルの先頭に自動的に付けられます。

BART の出力

次の終了ステータスが返されます。

0　　成功

1　　ファイル処理時の致命的でないエラー (アクセス権問題など)

>1　　致命的なエラー (無効なコマンド行オプションなど)

レポーティングメカニズムとして、詳細出力と、プログラムを考慮した出力の 2 種類を利用できます。

- 詳細出力はデフォルトの出力であり、地域対応化され、複数の行にわたって表示されます。詳細出力は国際化されたもので、ユーザーが内容を読み取ることができます。bart compare コマンドは、2 つのシステム目録を比較する時に、ファイル間の相違を示すリストを生成します。

  出力の構造は次のとおりです。

  filename attribute control:control-val test:test-val

  filename　　制御目録とテスト目録で相違があるファイルの名前。

  attribute　　比較される目録間で異なるファイル属性の名前。*test-val* の前に *control-val* が来ます。同じファイルの複数の属性に相違が見つかった場合、別々の行にそれぞれの相違が示されます。

  次に、/etc/passwd ファイルの属性の違いの例を示します。この出力から、size、mtime、および contents 属性に変化があったことがわかります。

  ```
  /etc/passwd:
  size control:74 test:81
  mtime control:3c165879 test:3c165979
  ```

```
contents control:daca28ae0de97afd7a6b91fde8d57afa
test:84b2b32c4165887355317207b48a6ec7
```

- `bart compare` コマンドの `-p` オプションを使用すると、プログラムを考慮した出力が生成されます。この出力は、プログラムでの操作に適しています。

  出力の構造は次のとおりです。

  filename attribute control-val test-val [*attribute control-val test-val*]*

filename	デフォルト形式における *filename* 属性に同じ
attribute control-val test-val	各ファイルの制御目録とテスト目録間で異なるファイル属性の説明

`bart` コマンドでサポートされる属性の一覧については、42 ページの「BART 規則ファイルの属性」を参照してください。

詳細は、`bart`(1M) のマニュアルページを参照してください。

ファイルセキュリティーの用語集

アクセス制御リスト (ACL) ファイルのアクセスまたは変更を行うアクセス権を持つユーザーまたはグループに関する情報が記載されているファイルに関連付けられたリスト。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。

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。通常のユーザーが特権アクションを実行できるようにするセキュリティーオーバーライドの集合。

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。

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。

セキュリティーポリシー ポリシーを参照してください。

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。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 一定の信頼レベルで管理タスクを実行できると判定されたユーザー。

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では、ホームディレクトリを暗号化する必要がある場合があります。

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。

索引

ま

や

T

U

Z