Oracle® Solaris 11.3 での監査の管理

Part No: E62759
2016 年 11 月

Part No: E62759

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法

『*Oracle® Solaris 11.3* での監査の管理』には、Oracle Solaris の監査機能について記載されています。

- 「**概要**」 – Oracle Solaris システムまたはシステムのネットワーク上で監査を管理する方法について説明します。
- 「**対象者**」 – 企業でセキュリティーを実装する必要があるシステム管理者。
- 「**必要な知識**」 - セキュリティーの概念および用語に熟知していること。

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 Oracle Solaris での監査について

Oracle Solaris の監査サブシステムは、システムがどのように使用されているかについてのレコードを保持しています。監査サービスには、監査データの分析を支援するツールが含まれています。

この章では、Oracle Solaris での監査機能について説明します。

計画の提案については、第2章「監査の計画」を参照してください。サイトで監査を構成する手順については、次の章を参照してください。

参照情報については、第7章「監査の参照情報」を参照してください。

監査とは

監査とは、システムリソースの使用状況に関するデータを収集することです。監査データは、セキュリティーに関連するシステムイベントの記録を提供します。このデータは、システムで発生する動作に対する責任の割り当てに使用できます。

監査を正常に機能させるには、まず識別と認証を行います。ログインのたびに、ユーザーがユーザー名を指定して PAM (Pluggable Authentication Module) 認証が成功すると、一意で変更不可能な監査ユーザー *ID* が生成されてそのユーザーに関連付けられるとともに、一意の監査セッション ID が生成されてそのユーザーのプロセスに関連付けられます。監査セッション ID は、そのログインセッション中に起動されるすべてのプロセスに継承されます。ユーザーが別のユーザーに切り替えても、すべてのユーザーアクションが同じ監査ユーザー ID で追跡されます。ID の切り替えについての詳細は、su(1M) のマニュアルページを参照してください。デフォルトでは、システムのブートやシャットダウンなどの特定のアクションは常に監査されます。

監査サービスを使用すると、次の操作を実行できます。

- システム上で発生するセキュリティーに関係するイベントのモニタリング
- ネットワーク全体の監査トレールへのイベントの記録
- 誤った使用または権限のない動作の検出
- アクセスパターンの確認と、ユーザーおよびオブジェクトのアクセス履歴の調査
- 保護メカニズムを迂回しようとする操作の検出
- ユーザーが ID を変更するときに発生する特権の拡大使用の検出

注記 - セキュリティーを維持するには、監査対象イベントにはパスワードなどの機密情報を含めないでください。詳細は、17 ページの「監査レコードと監査トークン」を参照してください。

監査の用語と概念

監査サービスでは、次の用語が使用されています。定義によっては、より詳細な説明への参照先も示します。

audit class	監査イベントのグループ。監査クラスは、監査対象のイベントのグループの選択方法を提供します。 詳細は、16 ページの「監査クラスおよび事前選択」、および audit_flags(5)、audit_class(4)、audit_event(4) の各マニュアルページを参照してください。
audit file system	バイナリ形式の監査ファイルのリポジトリ。 詳細は、19 ページの「監査ログ」および audit.log(4) のマニュアルページを参照してください。
audit event	監査可能なセキュリティー関連のシステムアクション。選択を容易にするために、イベントは監査クラスにグループ化されます。 詳細は、15 ページの「監査イベント」および audit_event(4) のマニュアルページを参照してください。

audit flag	コマンドまたはキーワードへの引数として指定される監査クラス。監査フラグは、プロセスで監査される監査クラスを事前選択します。 監査フラグの使用については、122 ページの「監査クラスの構文」を参照してください。`audit_flags(5)` のマニュアルページも参照してください。
audit plugin	キュー内の監査レコードを指定された場所に転送するモジュール。`audit_binfile` プラグインは、バイナリ監査ファイルを作成します。バイナリファイルは、監査ファイルシステム上に格納される監査トレールを構成します。`audit_remote` プラグインは、バイナリ監査レコードをリモートリポジトリに送信します。`audit_syslog` プラグインは、監査レコードを要約して、`syslog`(3C) を使用してシステムログに書き込みます。 詳細は、18 ページの「監査プラグインモジュール」、および `audit_binfile(5)`、`audit_remote(5)`、`audit_syslog(5)` の各モジュールのマニュアルページを参照してください。
audit policy	サイトで有効または無効を指定できる監査オプションのセット。次のようなポリシーを指定できます。 ■ 特定の種類の監査データを記録するかどうか ■ 監査の内容に含める情報の量 ■ 特定のタイプのファイルの処理方法 ■ 完全な監査キューの処理方法 詳細は、39 ページの「監査ポリシーについて」および `auditconfig(1M)` のマニュアルページを参照してください。
audit record	監査キュー内に収集された監査データ。1 つの監査レコードにつき 1 つの監査イベントが記述されます。各監査レコードは、一連の監査トークンから構成されます。 詳細は、17 ページの「監査レコードと監査トークン」および `audit.log(4)` のマニュアルページを参照してください。
audit token	監査レコードのフィールド。各監査トークンには、監査レコードの 1 つの属性 (ユーザー、グループ、プログラム、その他のオブジェクトなど) が記述されます。 詳細は、128 ページの「監査トークンの形式」および `audit.log(4)` のマニュアルページを参照してください。
audit trail	デフォルトのプラグイン `audit_binfile` を使用するすべての監査対象システムからの監査データを格納する 1 つ以上の監査ファイルの集合。

詳細は、127 ページの「監査トレール」を参照してください。

local auditing
: ローカルシステム上で生成された監査レコードの収集。レコードは、大域ゾーンまたは非大域ゾーン、あるいはその両方で生成される場合があります。

  詳細は、18 ページの「監査プラグインモジュール」を参照してください。

post-selection
: 事前選択された監査トレール内のどの監査イベントを検査するかの選択。デフォルトのアクティブなプラグイン `audit_binfile` によって監査トレールが作成されます。事後選択ツールである `auditreduce` コマンドは、監査トレールからレコードを選択します。

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

preselection
: どの監査クラスをモニターするかの最初の選択。事前選択された監査クラスの監査イベントが監査キュー内に収集されます。事前選択されていない監査クラスは監査されないため、それらのイベントはキューに入りません。

注記 - 事前選択された監査イベントのみが、`auditreduce` コマンドを使用したさらなる事後選択レビューに使用可能です。

詳細は、16 ページの「監査クラスおよび事前選択」、および `audit_flags(5)` と `auditconfig(1M)` のマニュアルページを参照してください。

public object
: `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` ディレクトリと `/usr/bin` ディレクトリの一部のファイルは公開オブジェクトです。公開オブジェクトの読み取り専用イベントは監査されません。たとえば、`file_read` (`fr`) 監査フラグが事前選択されている場合でも、公開オブジェクトの読み取り動作は監査されません。`public` 監査ポリシーオプションを変更すると、デフォルトをオーバーライドできます。

remote auditing
: アクティブな `audit_remote` プラグインが構成されている監査対象のシステムから監査レコードを受信して格納する監査リモートサーバー (ARS)。監査対象システムを ARS と区別するには、監査対象システムを「ローカルで監査されるシステム」と呼ぶことができます。

  詳細は、`auditconfig(1M)` のマニュアルページにある `-setremote` オプション、`ars(5)` のマニュアルページ、およ

び123 ページの「監査リモートサーバー」を参照してください。

監査イベント

監査イベントは、システム上の監査可能なアクションを表します。監査イベントは、/etc/security/audit_event ファイルに指定します。各監査イベントは、システムコールまたはユーザーコマンドに接続され、1 つ以上の監査クラスに割り当てられます。audit_event ファイルの形式については、audit_event(4) のマニュアルページを参照してください。

たとえば、AUE_EXECVE 監査イベントは、 execve() システムコールを監査します。コマンド auditrecord -e execve は、このエントリを表示します。

```
# auditrecord -e execve
execve
system call execve              See execve(2)
event ID    23                  AUE_EXECVE
class       ps,ex               (0x0000000080100000)
header
path
[attribute]                 omitted on error
[exec_arguments]            output if argv policy is set
[exec_environment]          output if arge policy is set
subject
[use_of_privilege]
return
```

監査クラス ps と監査クラス ex のどちらかを事前選択した場合は、すべての execve() システムコールが監査キュー内に記録されます。

監査は、ユーザーに起因するイベントとユーザーに起因しないイベントを処理します。監査ポリシーでは、次のように、イベントが同期イベントと非同期イベントに分割されます。

- **ユーザーに起因するイベント** – ユーザーによって起こるイベント。execve() システムコールはユーザーに起因させることができるため、その呼び出しはユーザーに起因するイベントと見なされます。ユーザーに起因するイベントはすべて、同期イベントです。
- **ユーザーに起因しないイベント** – カーネル割り込みレベルで、またはユーザーが認証される前に発生するイベント。na 監査クラスは、ユーザーに起因しない監査イベントを処理します。たとえば、システムのブートは、ユーザーに起因しないイベントです。ユーザーに起因しないイベントは、そのほとんどが非同期イベントです。ただし、失敗したログインなどの、プロセスが関連付けられている、ユーザーに起因しないイベントは同期イベントです。
- **同期イベント** – システムのプロセスに関連付けられたイベント。システムイベントの大半は同期イベントです。同期イベントをキューに入れることができない場合、キューに入れられるようになるまでプロセスはブロックされます。

- **非同期イベント** – プロセスに関連付けられていないイベント。そのため、ブロックしたあとに起動す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。

監査サービスによって定義される監査イベントに加えて、サードパーティーのアプリケーションも監査イベントを生成できます。32768 から 65535 までの監査イベント番号がサードパーティーのアプリケーションで使用できます。ベンダーは、イベント番号を予約し、監査インタフェースへのアクセスを取得するために Oracle Solaris の担当者に連絡する必要があります。

監査クラスおよび事前選択

各監査イベントは、1 つの監査クラスに属しています。監査クラスは、多数の監査イベントが入った便利な入れ物です。クラスを監査対象として事前選択した場合は、そのクラス内のすべてのイベントが監査キュー内に記録されます。たとえば、`ps` 監査クラスを事前選択すると、`execve()`、`fork()`、およびその他のシステムコールが記録されます。

システム上のイベントまたは特定のユーザーによって開始されるイベントに対して事前選択できます。

- **システム全体の事前選択** – `-auditconfig` コマンドの `-setflags` および `setnaflags` オプションを使用して、監査のためのシステム全体のデフォルトを指定します。

  注記 - `perzone` ポリシーが設定されている場合は、すべてのゾーンでデフォルトの監査クラスを指定できます。`perzone` 監査の場合は、システム全体ではなく、ゾーン規模がデフォルトです。

- **ユーザー固有の事前選択** – システム全体の監査デフォルトとともに含める個々のユーザーの監査に追加の監査フラグを指定します。`useradd`、`roleadd`、`usermod`、および `rolemod` コマンドは、`audit_flags` セキュリティー属性を `user_attr` データベース内に配置します。`profiles` コマンドは、権利プロファイルの監査フラグを `prof_attr` データベース内に配置します。

  監査事前選択マスクにより、ユーザーに対して監査されるイベントのクラスを決定します。ユーザーの事前選択マスクについては、126 ページの「プロセスの監査特性」を参照してください。使用されている構成済みの監査フラグについては、『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 「割り当てられた権利の検索順序」を参照してください。

監査クラスは、`/etc/security/audit_class` ファイルに定義されます。各エントリには、クラスの監査マスク、クラスの名前、およびクラスの記述名が含まれます。た

とえば、lo および ps クラス定義は、次のように audit_class ファイルに表示されます。

```
0x0000000000001000:lo:login or logout
0x0000000000100000:ps:process start/stop
```

監査クラスには、all と no の 2 つのグローバルクラスが含まれています。監査クラスについては、audit_class(4) のマニュアルページを参照してください。クラスの一覧については、/etc/security/audit_class ファイルを参照してください。

監査イベントのクラスへの割り当ては構成可能です。クラスからイベントを削除したり、クラスにイベントを追加したり、新しいクラスを作成して選択したイベントを含めることができます。手順については、61 ページの「監査イベントの所属先クラスの変更方法」を参照してください。クラスにマップされているイベントを表示するには、auditrecord -c *class* コマンドを使用します。

監査レコードと監査トークン

各監査レコードには、監査された 1 つのイベントの発生が記録されます。レコードには、動作を行なったユーザー、影響を受けたファイル、試みられた動作、その動作が発生した位置と時刻などの情報が含まれます。次の例は、header、subject、return の 3 つのトークンを含む login 監査レコードを示しています。

```
header,69,2,login - local,,example_system,2010-10-10 10:10:10.020 -07:00
subject,jdoe,jdoe,staff,jdoe,staff,1210,4076076536,69 2 example_system
return,success,0
```

監査イベントごとに保存される情報の種類は、一連の監査トークンによって定義されます。イベントの監査レコードが生成されるたびに、そのイベントに対して定義されたトークンの一部またはすべてが、そのレコードに書き込まれます。どのトークンが記録されるかは、イベントの性質によって決まります。前の例では、各行が監査トークンの名前で始まっています。監査トークンの内容は、トークン名のあとに続きます。header、subject、および return 監査トークンがまとまって、login - local 監査レコードを構成します。監査レコードを構成するトークンを表示するには、auditrecord -e *event* コマンドを使用します。

注記 - sensitive というシステム属性を持つファイルには、監査レコードに含まれている内容または内容の変更は入っていません。この属性によって、すべてのユーザーが特定のファイル内の機密情報 (パスワード、PIN、鍵など) にアクセスできなくなります。詳細は、pfedit(1M) のマニュアルページを参照してください。

---

praudit 出力例の各監査トークンの構造については、128 ページの「監査トークンの形式」を参照してください。監査トークンのバイナリストリームについては、audit.log(4) のマニュアルページを参照してください。

監査プラグインモジュール

監査プラグインモジュールは、監査レコードを監査キューからファイルまたはリポジトリに送信します。少なくとも 1 つのプラグインがアクティブであるか、ARS が構成されている必要があります。デフォルトでは、`audit_binfile` プラグインがアクティブです。プラグインは、`auditconfig -setplugin` *plugin-name* コマンドを使用して構成します。

監査サービスでは、次のプラグインが提供されます。

- `audit_binfile` プラグイン – バイナリ監査ファイルへの監査キューの配信を処理します。詳細は、`audit.log(4)` のマニュアルページを参照してください。
- `audit_remote` プラグイン – 監査キューから構成済みのリモートサーバーへのバイナリ監査レコードのセキュアな配信を処理します。`audit_remote` プラグインは、`libgss()` ライブラリを使用してサーバーを認証します。この転送は、プライバシと完全性のために保護されます。ARS については、123 ページの「監査リモートサーバー」を参照してください。
- `audit_syslog` プラグイン – 監査キューから `syslog` ログへの選択されたレコードの配信を処理します。

プラグインを構成する方法については、`auditconfig(1M)` のマニュアルページを参照してください。プラグイン構成の例については、79 ページの「監査ログの構成」のタスクを参照してください。プラグインについては、`audit_binfile(5)`、`audit_remote(5)`、`audit_syslog(5)` の各マニュアルページを参照してください。

監査リモートサーバー

監査リモートサーバー (ARS) は、`audit_remote` プラグインと対をなします。ARS は、サーバー構成に従って、プラグインにより送信されるデータを取得、処理、および格納できます。

ARS は、Oracle Solaris システムで無効な監査コンポーネントとして提供されます。リモート監査トレールの処理に使用するには、事前に ARS を構成する必要があります。ARS を構成するには:

- セキュアな監査データの転送に使用される、ベースとなるセキュリティーメカニズムを構成します。`audit_remote`(5) のマニュアルページを参照してください。
- `auditconfig -setremote` コマンドを使用して、監査リモートサブシステムを構成します。構成には、サーバー構成と接続グループ構成の両方が含まれています。接続グループは、同じローカルストレージパラメータを共有するシステムのセット

です。情報と例については、ars(5)のマニュアルページと auditconfig(1M) のマニュアルページを参照してください。

123 ページの「監査リモートサーバー」も参照してください。

監査ログ

監査レコードは監査ログ内に収集されます。監査サービスでは、監査レコードのための 3 つの出力モードが提供されます。

- 監査ファイルと呼ばれるログは、バイナリ形式で監査レコードを格納します。システムまたはサイトからの一連の監査ファイルによって、完全な監査レコードが提供されます。完全な監査レコードは監査トレールと呼ばれます。これらのログは audit_binfile プラグインまたは監査リモートサーバーで作成され、praudit および auditreduce 事後選択コマンドで確認できます。
- audit_remote プラグインは、監査レコードをリモートリポジトリにストリーム出力します。リポジトリは監査トレールを保持し、事後選択ツールを指定する役割を果たします。
- syslog ユーティリティーは、監査レコードの概要テキストを収集して格納します。syslog レコードは、完全なレコードではありません。次の例は、login 監査レコードの syslog エントリを示します。

```
Feb 5 11:54:57 example_system audit: [ID 702911 audit.notice] \
login - login ok  session 2870512630 by user as user:staff
```

注記 - syslog に代わりに、rsyslog パッケージをインストールして、リモート syslog 機能の rsyslog ユーティリティーを使用できます。Rsyslog は、フィルタリング、TCP、暗号化、精度の高いタイムスタンプ、出力制御などの機能をサポートするモジュラー設計を備えた、信頼性の高い拡張 syslog 実装です。

サイトでは、すべての書式の監査レコードを収集するように監査を構成できます。サイトにあるシステムを、バイナリモードをローカルに使用したり、バイナリファイルをリモートリポジトリに送信したり、syslog モードを使用したりするように構成できます。次の表では、バイナリ監査レコードを syslog 監査レコードと比較しています。

表 1 バイナリ、リモート、および syslog 監査レコードの比較

機能	バイナリレコードとリモートレコード	syslog レコード
プロトコル	バイナリ – ファイルシステムに書き込みます リモート – リモートリポジトリにストリーム出力します	リモートログ記録に UDP を使用します rsyslog では TCP を使用します

機能	バイナリレコードとリモートレコード	syslog レコード
データ型	バイナリ	テキスト
レコード長	無制限	監査レコードあたり最大 1024 文字
場所	バイナリ – システム上の zpool 内に格納されます リモート – リモートリポジトリ	syslog.conf ファイルで指定された場所に格納されます
構成方法	バイナリ – audit_binfile プラグインで p_dir 属性を設定します リモート – audit_remote プラグインで p_hosts 属性を設定し、そのプラグインをアクティブにします	audit_syslog プラグインをアクティブにし、syslog.conf ファイルを構成します
読み取り方法	バイナリ – 通常、バッチモードでは XML でのブラウザ出力 リモート – リポジトリによって手順が規定されます	リアルタイムに、または syslog 用に作成されたスクリプトによって検索されます 標準テキストの出力
完全性	完全であることと、正しい順序で表示されることが保証されます	完全であることが保証されていないサマリー
タイムスタンプ	協定世界時 (UTC)	監査されているシステム上の時間

プラグインと監査ログについての詳細は、次を参照してください。

- audit_binfile(5) のマニュアルページ
- audit_syslog(5) のマニュアルページ
- audit.log(4) のマニュアルページ
- ars(5)
- 83 ページの「監査トレールのための監査領域を割り当てる方法」
- 92 ページの「syslog 監査ログの構成方法」

バイナリレコードについて

バイナリ形式のレコードにより、強力なセキュリティーとカバレージが提供されます。バイナリ出力は、Common Criteria (http://www.commoncriteriaportal.org/) の監査要件などの、セキュリティー認定の要件を満たします。

audit_binfile プラグインは、レコードをスヌーピングから保護されたファイルシステムに書き込みます。単一のシステム上で、すべてのバイナリレコードが収集され、順番に表示されます。バイナリログ上の UTC タイムスタンプにより、1 つの監査トレール上のシステムがタイムゾーンをまたがって分散している場合の正確な比較が可能になります。praudit コマンドを使用すると、レコードを XML 形式でブラウザに表示できます。スクリプトを使用して、XML を解析することもできます。

audit_remote プラグインは、レコードをリモートリポジトリに書き込みます。このリポジトリは、ストレージと事後選択を処理します。

監査リモートサーバーではバイナリも生成されます。

syslog 監査レコードについて

これに対して、syslog または rsyslog レコードでは、利便性と柔軟性が向上する可能性があります。たとえば、さまざまなソースから syslog データを収集できます。また、syslog.conf ファイル内の audit.notice イベントをモニターする場合、syslog ユーティリティーは現在のタイムスタンプで監査レコードのサマリーをログに記録します。ワークステーション、ネットワークサーバー、ファイアウォール、ルーターなどのさまざまなソースからの syslog メッセージ用に開発された管理および分析ツールと同じツールを使用できます。レコードをリアルタイムで表示し、リモートシステム上に格納することができます。

syslog.conf を使用して監査レコードをリモートに格納すると、攻撃者による改変や削除からログデータが保護されます。ただし、syslog モードには、次のような欠点があることを考慮してください。

- レコードがネットワーク攻撃 (サービス拒否や、なりすましソースアドレスなど) を受けやすくなります。
- UDP プロトコルは、パケットを破棄したり、間違った順番でパケットを配信したりすることがあります。
- syslog エントリを 1024 文字に制限すると、ログ内の一部の監査レコードが切り捨てられる可能性があります。
- 単一のシステム上で一部の監査レコードが収集されず、順番に表示されない可能性があります。
- 各監査レコードには、ローカルシステムの日付と時間がスタンプされます。したがって、複数システムの監査トレールを構築する際に、タイムスタンプを信頼できません。

監査トレールの格納および管理

audit_binfile プラグインがアクティブな場合は、監査ファイルシステムにバイナリ形式の監査ファイルが保持されます。標準インストールでは /var/audit が使用されますが、追加のファイルシステムも使用できます。すべての監査ファイルシステムの内容は、監査トレールで構成されています。監査レコードは、これらのファイルシステム内に次の順序で格納されます。

- **プライマリ監査ファイルシステム** – /var/audit は、システムの監査ファイルのためのデフォルトのファイルシステムとして機能します

注記 - /var/audit は、実際には /var/share ファイルシステムへのシンボリックリンクであり、これによりルートプールの一部として共有アクセスが可能になります。ただし、Oracle Solaris では /var/audit はファイルシステムのように扱われます。

- **セカンダリ監査ファイルシステム** – システムの監査ファイルが管理者の判断で配置されるファイルシステム

これらのファイルシステムは、audit_binfile プラグインの p_dir 属性への引数として指定されます。各ファイルシステムは、このリスト内で前にあるファイルシステムがいっぱいになるまで使用されません。ファイルシステムエントリのリストを含む例については、80 ページの「監査ファイルのための ZFS ファイルシステムを作成する方法」を参照してください。

監査ファイルをデフォルトの監査ルートディレクトリに配置すると、監査証跡を確認する際に役立ちます。auditreduce コマンドは、監査ルートディレクトリを使用して監査トレール内のすべてのファイルを検索します。/var/audit は、デフォルトの監査ルートディレクトリとして機能します。

auditreduce コマンドでは、次のオプションを使用できます。

- auditreduce コマンドの -M オプションを使用すると、特定の物理マシンから監査ファイルを指定できます。
- -S オプションを使用すると、別の監査ファイルシステムを指定できます。

auditreduce コマンドの使用例については、103 ページの「監査トレールの監査ファイルをマージする方法」を参照してください。詳細は、auditreduce(1M) のマニュアルページを参照してください。

監査サービスでは、監査トレールのファイルを結合したり、フィルタしたりするためのコマンドが提供されます。auditreduce コマンドは、監査トレールの監査ファイルをマージします。また、ファイルをフィルタして特定のイベントを検出できます。praudit コマンドは、バイナリファイルを読み取ります。praudit コマンドのオプションにより、スクリプトやブラウザの表示に適した出力が得られます。

信頼性の高いタイムスタンプの保証

複数のシステムからの監査ログをマージする場合は、それらのシステム上の日付と時間が正確である必要があります。同様に、監査ログをリモートシステムに送信する場合も、記録システムとリポジトリシステムのクロックが正確である必要があります。Network Time Protocol (NTP) では、システムクロックが正確で、調整された状態に維持されます。詳細は、『Introduction to Oracle Solaris 11.2 Network Services』 の 第 3 章, 「Time-Related Services」および ntpd (1M) のマニュアルページを参照してください。

リモートリポジトリの管理

audit_remote プラグインが構成されたあとは、リモートリポジトリが監査レコードを受信します。ARS は、監査レコードの受信者を提供します。監査レコードは、保護された接続を経由して ARS にストリーム出力されるため、ローカルに格納される場合と同様に格納できます。ARS を構成するには、88 ページの「監査ファイルのためのリモートリポジトリを構成する方法」を参照してください。ARS については、123 ページの「監査リモートサーバー」および ars(5) のマニュアルページを参照してください。

監査とセキュリティーとの関連

監査は、システムの使用状況の疑わしい、または異常なパターンを明らかにすることによって、潜在的なセキュリティー違反の検出に役立ちます。監査ではまた、疑わしいアクションを追跡して特定のユーザーを突き止めるための手段も提供されるため、抑止力としても機能します。活動を監査されていることをユーザーが知っている場合、悪質な活動を試みる可能性は低くなると考えられます。

コンピュータシステム、特にネットワーク上のシステムを保護するためには、システムのプロセスまたはユーザーのプロセスが開始する前に活動を制御するメカニズムが必要です。セキュリティーの確保には、動作の経過をモニターするツールが必要となります。また、セキュリティーの確保には、動作終了後に動作内容を報告することも必要です。

ほとんどの監査アクティビティーでは現在のイベントがモニタリングされ、指定されたパラメータを満たすイベントが報告されるため、ユーザーがログインするか、またはシステムプロセスが開始される前に監査パラメータを設定します。これらのイベントが監査サービスでどのようにモニターおよび報告されるかについては、第2章「監査の計画」および 第3章「監査サービスの管理」で詳細に説明されています。

監査では、ハッカーによる不正な侵入を防止することはできません。ただし、監査サービスでは、たとえば、特定のユーザーが特定の日時に特定の動作を行なったことが報告されます。監査報告では、入力経路とユーザー名によってこのユーザーを特定できます。これらの情報は、すぐに端末に表示したり、ファイルに出力してあとで分析したりできます。このように、監査サービスの提供するデータから、次のことが判断できます。

- どのようにシステムセキュリティーが低下したか
- 必要なセキュリティーレベルを実現するために閉じることが必要なセキュリティーホールはどれか

監査の機能

監査では、指定したイベントが発生したときに監査レコードが生成されます。通常、次のイベントが発生すると監査レコードが生成されます。

- システムの起動とシャットダウン
- ログインとログアウト
- プロセスまたはスレッドの作成と破棄
- オブジェクトを開く、閉じる、削除する、または名前を変更する
- 権利の使用
- 識別動作と認証動作
- プロセスまたはユーザーによるアクセス権の変更
- パッケージのインストールなどの管理動作
- サイト固有のアプリケーション

次の 3 つが監査レコードの生成元になります。

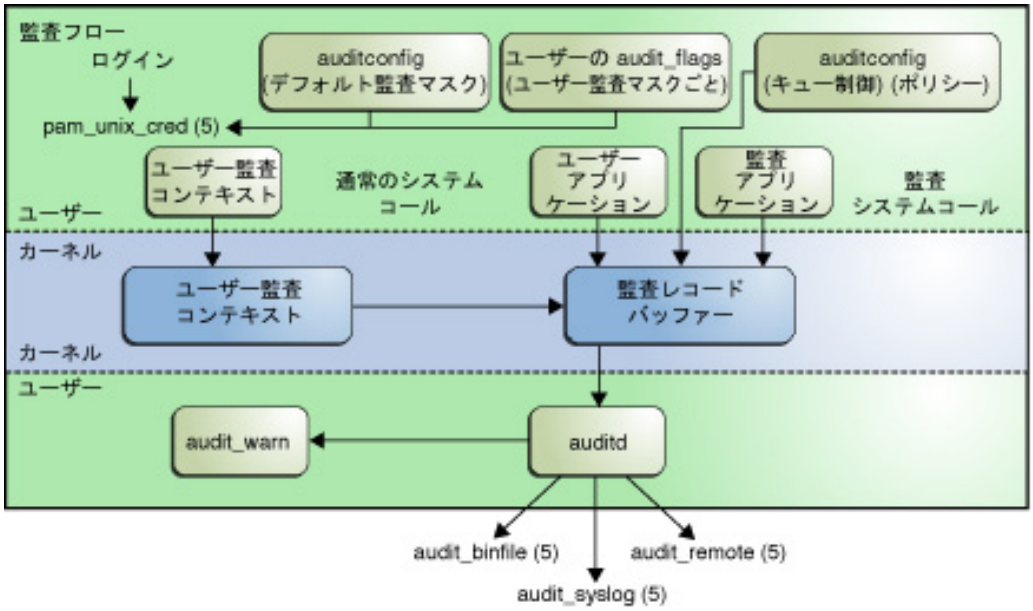
- アプリケーション
- 非同期監査イベントの結果
- プロセスのシステムコールの結果

関連するイベント情報が取得されたあと、これらの情報が監査レコードに書式設定されます。各監査レコードには、イベントを識別する情報、イベントの発生元、イベントの時間、およびその他の関連情報が格納されます。このレコードが次に監査キュー内に配置され、格納のためにアクティブなプラグインに送信されます。すべてのプラグインをアクティブにできますが、少なくとも 1 つのプラグインがアクティブであるか、リモート監査サーバーが構成されている必要があります。プラグインについては、24 ページの「監査の構成方法」および18 ページの「監査プラグインモジュール」で説明されています。

監査の構成方法

システム構成中に、どのクラスの監査レコードをモニターするかを事前選択します。各ユーザーに行う監査の程度は、細かく調整することもできます。次の図は、Oracle Solaris での監査のフローの詳細を示しています。

図 1 監査のフロー



監査データは、カーネルに収集されたあと、プラグインにより適切な場所に配布されます。

- `audit_binfile` プラグインは、バイナリ監査レコードを `/var/audit` 内に配置します。デフォルトでは、`audit_binfile` プラグインがアクティブです。事後選択ツールを使用すると、監査トレールの関心のある部分を検査できます。

  監査ファイルは、1 つ以上の ZFS プール内に格納できます。接続された監査ファイルの集合は、監査トレールと呼ばれます。
- `audit_remote` プラグインは、保護されたリンク全体にわたるバイナリ監査レコードを監査リモートサーバーに送信します。
- `audit_syslog` プラグインは、監査レコードの概要テキストを `syslog` ユーティリティーに送信します。

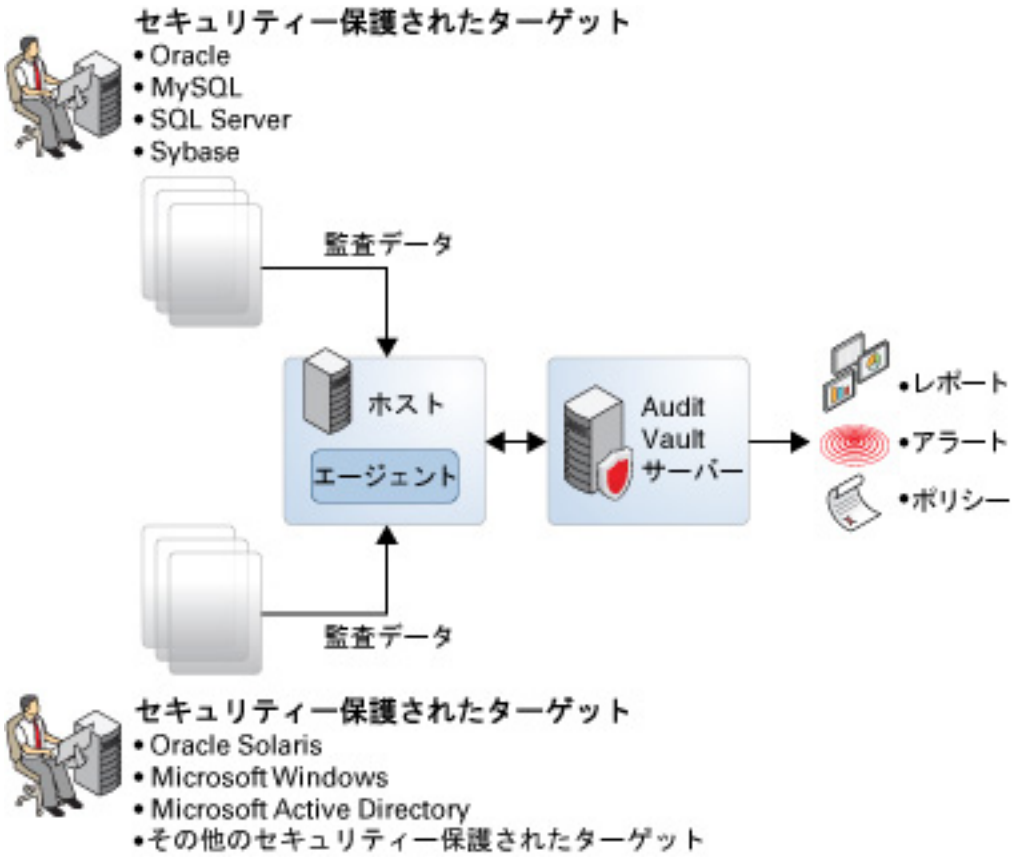
非大域ゾーンをインストールするシステムは、大域ゾーンからと同じようにすべてのゾーンを監査できます。これらのシステムは、非大域ゾーンのさまざまなレコードを収集するように構成することもできます。詳細は、28 ページの「Oracle Solaris Zones を使用したシステムの監査」を参照してください。

Oracle Audit Vault and Database Firewall を使用した監査レコードの格納および分析

Oracle Solaris システムの監査イベントは、Oracle Audit Vault and Database Firewall リリース 12.1.0.0 に組み込むことができます。Oracle Audit Vault and Database Firewall では、Oracle および Oracle 以外のデータベースからの監査データの統合およびモニタリングが自動化されます。その後、Oracle Audit Vault and Database Firewall を使用して、Oracle Solaris システム上で監査対象イベントを分析およびレポートできます。詳細は、Oracle Audit Vault and Database Firewall (`http://www.oracle.com/technetwork/products/audit-vault/overview/index.html`)を参照してください。

次の図は、指定したセキュリティー保護されたターゲットから Oracle Solaris 監査レコードを収集する Oracle Audit Vault and Database Firewall の動作を示しています。セキュリティー保護されたターゲットは、監査レコードまたはデータが格納される任意のシステムです。

図 2 Oracle Solaris と Audit Vault



ホストシステムは、Oracle Audit Vault and Database Firewall と通信する AV エージェントを実行するために指定されています。エージェントを使用すると、Oracle Audit Vault and Database Firewall がセキュリティー保護されたターゲットから監査データを受信して、処理できます。エージェントは、セキュリティー保護されたターゲット上で指定された監査トレールから監査レコードを読み取ります。これらの監査レコードは、ネイティブのバイナリ形式でエンコードされます。エージェントは、データを Oracle Audit Vault and Database Firewall で解析可能な形式に変換します。Oracle Audit Vault and Database Firewall はデータを受信し、必要に応じて、管理者およびセキュリティーマネージャー用のレポートを生成します。

エージェントは、個別のシステムの代わりに、セキュリティー保護されたターゲットにインストールできます。エージェントを使用した複数のシステムを構成して、Audit

Vault サーバーに接続することもできます。ただし、監査データを取得するには、セキュリティー保護されたターゲットを登録するときに、AV サーバーが通信する特定のシステムを指定します。

Oracle Solaris でセキュリティー保護されたターゲットと Oracle Solaris 以外でセキュリティー保護されたターゲットの両方からの監査レコードを受け入れるように Oracle Audit Vault and Database Firewall を構成するには、指定したホストシステム上にエージェントがインストールされ、アクティブになっていることを確認します。詳細は、Oracle Audit Vault and Database Firewall のドキュメント (`http://www.oracle.com/technetwork/database/database-technologies/audit-vault-and-database-firewall/documentation/index.html`) を参照してください。

Oracle Solaris Zones を使用したシステムの監査

ゾーンは、Oracle Solaris OS の単一インスタンス内に作成される仮想化オペレーティング環境です。監査サービスは、ゾーン内でのアクティビティーも含め、システム全体を監査します。非大域ゾーンがインストールされたシステムでは、大域ゾーンで単一の監査サービスを実行してすべてのゾーンを同様に監査できます。または、大域ゾーンの監査サービスを含め、非大域ゾーンごとに 1 つの監査サービスを実行できます。これらの監査サービスは別個に管理されます。

サイトは、次の場合に大域ゾーン内で単一の監査サービスを実行できます。

- 単一イメージの監査トレールを必要とするサイトである。
- 非大域ゾーンがアプリケーションコンテナとして使用されている。ゾーンが 1 つの管理ドメインの一部になっています。つまり、どの非大域ゾーンにおいても、ネームサービスファイルがカスタマイズされていません。

  システム上のすべてのゾーンが 1 つの管理ドメイン内に存在する場合は、`zonename` 監査ポリシーを使用して、異なるゾーン内に構成された監査イベントを区別できます。
- 管理者が低い監査オーバーヘッドを望んでいる。大域ゾーン管理者がすべてのゾーンを同様に監査します。また、大域ゾーンの監査デーモンがシステム上のすべてのゾーンを処理します。

サイトは、次の場合に非大域ゾーンごとに 1 つの監査サービスを実行できます。

- 単一イメージの監査トレールを必要としないサイトである。
- 非帯域ゾーンでネームサービスファイルがカスタマイズされている。これらの個々の管理ドメインは通常、ネットワークサーバーとして機能します。
- 個々のゾーン管理者が、自身が管理するゾーンの監査を制御する必要がある。ゾーンごとの監査では、ゾーン管理者は、自身が管理するゾーンの監査を有効にするか無効にするかを決定できます。

ゾーンごとの監査の利点は、監査トレールをゾーンごとにカスタマイズできることと、ゾーンの監査をゾーン単位で無効化できることです。これらの利点は、管理オーバーヘッドによって相殺される可能性があります。各ゾーン管理者は、監査を管理する必要があります。各ゾーンでは、独自の監査デーモンが実行され、独自の監査キューや監査ログが用意されます。これらの監査ログを管理する必要があります。

♦ ♦ ♦ 第 2 章

2 監査の計画

この章では、Oracle Solaris インストールに対する監査サービスのカスタマイズを計画する方法について説明します。

監査の概要については、第1章「Oracle Solaris での監査について」を参照してください。サイトで監査を構成する手順については、次の章を参照してください。

- 第3章「監査サービスの管理」
- 第4章「システム動作のモニタリング」
- 第5章「監査データの操作」
- 第6章「監査サービスに関する問題の分析および解決」

参照情報については、第7章「監査の参照情報」を参照してください。

監査の計画の概念

監査する動作の種類を適切に選択し、有用な監査情報を収集することが望まれます。監査対象者と監査対象も注意して計画する必要があります。デフォルトの `audit_binfile` プラグインを使用している場合は、監査ファイルが急速に拡大して使用可能な領域がいっぱいになることがあります。

単一システムの監査トレールの計画

注記 - 単一システムの監査トレールの実装は、`audit_binfile` プラグインにのみ適用されます。

システムが単一の管理ドメイン内にある場合、単一システムイメージの監査トレールを作成できます。

サイトに単一システムイメージの監査トレールを作成するには、次の要件に従います。

- すべてのシステムで同じネームサービスを使用します。

  監査レコードの正しい解釈のためには、`passwd`、`group`、および `hosts` ファイルの整合性がとれている必要があります。
- すべてのシステム上で監査サービスを同様に構成します。サービス設定の表示および変更については、`auditconfig(1M)` のマニュアルページを参照してください。
- すべてのシステムで同じ `audit_warn`、`audit_event`、および `audit_class` ファイルを使用します。

システム上の監査を有効にするための追加の考慮事項については、34 ページの「監査対象者と監査対象イベントの計画方法」を参照してください。

ゾーン内での監査の計画

システムに非大域ゾーンが含まれている場合は、大域ゾーンを監査する場合と同様にこれらのゾーンを監査できます。あるいは、非大域ゾーンごとの監査サービスを個別に構成したり、有効または無効にしたりできます。たとえば、大域ゾーンを監査せずに、非大域ゾーンのみを監査できます。

トレードオフについては、28 ページの「Oracle Solaris Zones を使用したシステムの監査」を参照してください。

ゾーン内に監査を実装する際は、次のオプションを使用できます。

すべてのゾーンに対応する 1 つの監査サービスの実装

すべてのゾーンを同様に監査すると、単一イメージの監査トレールを作成できます。単一イメージの監査トレールは、`audit_binfile` または `audit_remote` プラグインを使用しており、システム上のすべてのゾーンが 1 つの管理ドメインに含まれている場合に実行されます。その場合は、すべてのゾーン内のレコードが同一の設定で事前選択されるため、監査レコードを容易に比較できます。

この構成では、すべてのゾーンが 1 つのシステムの一部として扱われます。大域ゾーンはシステム上の唯一の監査サービスを実行し、ゾーンごとに監査レコードを収集します。`audit_class` および `audit_event` ファイルは大域ゾーンでのみカスタマイズし、そのあと、これらのファイルをすべての非大域ゾーンにコピーします。

すべてのゾーンに対応する単一の監査サービスを構成する際は、次のガイドラインを使用します。

- すべてのゾーンで同じネームサービスを使用します。

  注記 - ネームサービスファイルが非大域ゾーンでカスタマイズされ、perzone ポリシーが設定されていない場合は、使用可能なレコードの選択に監査ツールを注意して使用する必要があります。あるゾーン内のユーザー ID は、異なるゾーン内の同じ ID とは異なるユーザーを指している可能性があります。

- 監査レコードにゾーンの名前を含めることができるようにします。

  ゾーン名を監査レコードの一部として含めるには、大域ゾーンで zonename ポリシーを設定します。次に、auditreduce コマンドで、監査証跡からゾーンに基づいて監査イベントを選択できます。例については、auditreduce(1M) のマニュアルページを参照してください。

単一イメージ監査証跡を計画するには、34 ページの「監査対象者と監査対象イベントの計画方法」を参照してください。最初の手順から始めます。また、大域ゾーン管理者は、37 ページの「監査レコードのディスク容量を計画する方法」の説明に従ってストレージも確保する必要があります。

ゾーンごとに 1 つの監査サービスの実装

ゾーンごとに異なるネームサービスデータベースを使用している場合や、ゾーン管理者が自分のゾーン内の監査を制御する場合は、ゾーンごとの監査を構成することを選択します。

注記 - 非大域ゾーンを監査するには perzone ポリシーを設定する必要がありますが、大域ゾーンで監査サービスを有効にする必要はありません。非大域ゾーンの監査は大域ゾーンとは別に構成し、その監査サービスも大域ゾーンとは別に有効または無効にします。

- ゾーンごとの監査を構成する場合は、大域ゾーンで perzone 監査ポリシーを設定します。非大域ゾーンが最初にブートされる前にゾーンごとの監査が設定されている場合は、そのゾーンの最初のブート時に監査が開始されます。監査ポリシーを設定するには、75 ページの「ゾーンごとの監査を構成する方法」を参照してください。
- 各ゾーン管理者がゾーンの監査を構成します。

  非大域ゾーン管理者は、perzone と ahlt 以外のすべてのポリシーオプションを設定できます。
- 各ゾーン管理者はゾーン内で監査を有効化または無効化できます。

- 確認中に発生元のゾーンまで追跡できるレコードを生成するには、`zonename` 監査ポリシーを設定します。

監査の計画

次のタスクマップは、ディスク容量および記録対象のイベントを計画するために必要な主なタスクを示しています。

表 2 監査の計画タスクマップ

タスク	手順
監査対象者と監査対象イベントを決定します	34 ページの「監査対象者と監査対象イベントの計画方法」
監査トレールのストレージ容量を計画します	37 ページの「監査レコードのディスク容量を計画する方法」
リモートサーバーへの監査トレールの転送の計画	38 ページの「監査レコードのリモートストレージへのストリーム出力を準備する方法」

▼ 監査対象者と監査対象イベントの計画方法

始める前に 非大域ゾーンを実装している場合は、この手順を使用する前に、32 ページの「ゾーン内での監査の計画」を確認してください。

1. **監査ポリシーを決定します。**

   デフォルトでは、`cnt` ポリシーだけが有効になっています。

   使用可能なポリシーオプションの説明を表示するには、`auditconfig -lspolicy` コマンドを使用します。

   - ポリシーオプションの効果については、39 ページの「監査ポリシーについて」を参照してください。
   - `cnt` ポリシーの効果については、124 ページの「非同期イベントおよび同期イベントの監査ポリシー」を参照してください。
   - 監査ポリシーを設定するには、55 ページの「監査ポリシーを変更する方法」を参照してください。

2. **イベントからクラスへのマッピングを追加するかどうかを決定します。**

   ほぼすべての状況で、デフォルトのマッピングをそのまま使用できます。ただし、新しいクラスを追加する場合は、イベントからクラスへのマッピングに追加してください。

例は、61 ページの「監査イベントの所属先クラスの変更方法」を参照してください。

3. **事前選択する監査クラスを決定します。**

   監査クラスを追加したり、デフォルトのクラスを変更したりするには、ユーザーがシステムにログインする前に行うのが最適です。

   `auditconfig` コマンドの `-setflags` および `-setnaflags` オプションを使用して事前選択した監査クラスは、すべてのユーザーとプロセスに適用されます。成功、失敗、またはその両方に対してクラスを事前選択できます。

   監査クラスの一覧については、`/etc/security/audit_class` ファイルを参照してください。

4. **システム全体の事前選択に対するユーザーの変更を決定します。**

   一部のユーザーをシステムとは異なる方法で監査した方がよいと判断される場合は、個々のユーザーまたは権利プロファイルに対する `audit_flags` セキュリティー属性を変更できます。ユーザーの事前選択マスクは、監査フラグが明示的に設定されているユーザー、または明示的な監査フラグを含む権利プロファイルが割り当てられているユーザーに対して変更されます。

   手順については、51 ページの「ユーザーの監査特性を構成する方法」を参照してください。どの監査フラグの値が有効かについては、『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 「割り当てられた権利の検索順序」を参照してください。

5. **`audit_warn` 電子メールのエイリアスの管理方法を決定します。**

   `audit_warn` スクリプトは、監査システムによって管理上の注意が必要な状況が検出されたときは常に実行されます。デフォルトでは、`audit_warn` スクリプトは、`audit_warn` のエイリアスに電子メールを送信し、コンソールにメッセージを送信します。

   エイリアスを設定するには、59 ページの「`audit_warn` 電子メールエイリアスの構成方法」を参照してください。

6. **監査レコードをどの書式で、どこに収集するかを決定します。**

   次の 3 つの選択肢があります。

   - デフォルトでは、バイナリ監査レコードをローカルに格納します。デフォルトの格納ディレクトリは `/var/audit` です。`audit_binfile` プラグインをさらに構成するには、80 ページの「監査ファイルのための ZFS ファイルシステムを作成する方法」を参照してください。
   - `audit_remote` プラグインを使用して、バイナリ監査レコードを保護されたリモートリポジトリにストリーム出力します。レコードの受信者が存在する必要があります。要件については、23 ページの「リモートリポジトリの管理」を参照してくだ

さい。手順については、86 ページの「監査ファイルをリモートリポジトリに送信する方法」を参照してください。

- audit_syslog プラグインを使用して、監査レコードの概要を syslog に送信します。手順については、92 ページの「syslog 監査ログの構成方法」を参照してください。

  バイナリ形式と syslog 形式の比較については、19 ページの「監査ログ」を参照してください。

7. **縮小しているディスク容量について管理者にいつ警告するかを決定します。**

注記 - この手順は、audit_binfile プラグインにのみ適用されます。

監査ファイルシステム上のディスク容量が最小空き容量の割合 (または、弱い制限値) を下回ると、監査サービスは次に使用可能な監査ディレクトリに切り替えます。このサービスは次に、弱い制限値を超えたことを示す警告を送信します。

最小空き容量の割合を設定する方法を確認するには、例25「警告のための弱い制限値を設定する」を参照してください。

8. **すべての監査ディレクトリがいっぱいになったときの動作を決定します。**

注記 - この手順は、audit_binfile プラグインにのみ適用されます。

デフォルトの構成では、audit_binfile プラグインがアクティブであり、cnt ポリシーが設定されます。この構成では、カーネル監査キューがいっぱいになっても、システムは動作し続けます。システムは破棄された監査レコードを数えますが、イベントを記録しません。セキュリティーをより向上させるためには、cnt ポリシーを無効にして、ahlt ポリシーを有効にします。ahlt ポリシーは、非同期イベントが監査キューに配置できない場合、システムを停止します。

ただし、audit_binfile キューがいっぱいになったときに、別のアクティブなプラグインのキューがいっぱいでない場合、カーネルキューは、そのいっぱいでないプラグインにレコードを送信し続けます。audit_binfile キューがふたたびレコードを受け入れることができるようになると、監査サービスは、そのキューへのレコードの送信を再開します。

cnt および ahlt ポリシーオプションについては、124 ページの「非同期イベントおよび同期イベントの監査ポリシー」を参照してください。これらのポリシーオプションを構成する方法を確認するには、例10「ahlt 監査ポリシーオプションを設定する」を参照してください。

注記 - cnt または ahlt ポリシーは、少なくとも 1 つのプラグインのキューが監査レコードを受け入れている場合はトリガーされません。

監査レコード用のディスク容量の計画

audit_binfile プラグインは、監査トレールを作成します。監査トレールには専用のファイル領域が必要です。この領域は使用可能で、セキュリティー保護されている必要があります。システムは、初期のストレージとして /var/audit ファイルシステムを使用します。監査ファイルのための追加の監査ファイルシステムを構成できます。次の手順は、監査トレールのストレージを計画するときに解決する必要のある問題について説明しています。

▼ 監査レコードのディスク容量を計画する方法

始める前に 非大域ゾーンを実装している場合は、この手順を使用する前に、32 ページの「ゾーン内での監査の計画」を完了します。

この手順では、audit_binfile プラグインを使用することが前提となっています。

1. **サイトで必要な監査の程度を決定します。**

   サイトのセキュリティー上の必要性を考慮して、監査トレールに使用できるディスク容量を決定します。

   サイトのセキュリティーを確保しながら領域要件を削減する方法と、監査ストレージを設定する方法については、42 ページの「監査コストの制御」と43 ページの「効率的な監査」を参照してください。

   実際的な手順については、112 ページの「監査レコードの量が多い」、70 ページの「専用ファイルシステム上の監査ファイルを圧縮する方法」、および例34「監査ファイルを結合して削減する」を参照してください。

2. **どのシステムを監査するかを決定し、それらの監査ファイルシステムを構成します。**

   使用を予定しているすべてのファイルシステムの一覧を作成します。構成ガイドラインについては、21 ページの「監査トレールの格納および管理」および auditreduce(1M) のマニュアルページを参照してください。監査ファイルシステムを指定するには、83 ページの「監査トレールのための監査領域を割り当てる方法」を参照してください。

3. **すべてのシステム上のクロックの同期をとります。**

   詳細は、22 ページの「信頼性の高いタイムスタンプの保証」を参照してください。

監査レコードのリモートストレージへのストリーム出力の準備

audit_remote プラグインは、バイナリの監査トレールを、audit_binfile プラグインがローカル監査ファイルに書き込むのと同じ形式で ARS に送信します。audit_remote プラグインは、libgss ライブラリを使用して ARS を認証し、GSS-API メカニズムを使用してプライバシと完全性により転送を保護します。参照情報については、『Oracle Solaris 11.3 での Kerberos およびその他の認証サービスの管理』 の 「Kerberos サービスとは」および『Oracle Solaris 11.3 での Kerberos およびその他の認証サービスの管理』 の 「Kerberos ユーティリティー」を参照してください。

現在サポートされている GSS-API メカニズムは kerberosv5 だけです。詳細は、mech(4) のマニュアルページを参照してください。

▼ 監査レコードのリモートストレージへのストリーム出力を準備する方法

注記 - Kerberos レルムが構成されており、そのレルム内に識別された監査リモートサーバー (ARS) とすべての監査対象システムが存在する場合は、この手順をスキップできます。ARS と監査対象システムを構成する手順については、88 ページの「監査ファイルのためのリモートリポジトリを構成する方法」および86 ページの「監査ファイルをリモートリポジトリに送信する方法」で説明されています。

Kerberos レルムが構成されているかどうかを確認するには、次のコマンドを発行します。サンプル出力は、システム上に Kerberos がインストールされていないことを示しています。

```
# pkg info system/security/kerberos-5
pkg: info: no packages matching these patterns are installed on the system.
```

始める前に この手順では、audit_remote プラグインを使用することが前提となっています。

1. **マスター KDC (Key Distribution Center) パッケージをインストールします。**

   ARS として機能するシステムを使用するか、または近くのシステムを使用できます。ARS は、大量の認証トラフィックをマスター KDC に送信します。

   ```
   # pkg install pkg:/system/security/kerberos-5
   ```

   マスター KDC 上では、Kerberos kdcmgr および kadmin コマンドを使用してレルムを管理します。詳細は、kdcmgr(1M) および kadmin(1M) のマニュアルページを参照してください。

2. **ARS に監査レコードを送信するすべての監査対象システム上で、マスター KDC パッケージをインストールします。**

```
# pkg install pkg:/system/security/kerberos-5
```

このパッケージには kclient コマンドが含まれています。これらのシステム上では、kclient コマンドを実行して KDC と接続します。詳細は、kclient(1M) のマニュアルページを参照してください。

3. **KDC レルム内のクロックを同期化します。**
監査対象システムと ARS の間のクロックスキューが大きすぎる場合は、接続の試行が失敗します。接続が確立されると、127 ページの「バイナリ監査ファイルの命名規則」で説明されているように、ARS 上のローカル時間によって格納される監査ファイルの名前が決定されます。
クロックの詳細は、22 ページの「信頼性の高いタイムスタンプの保証」を参照してください。

監査ポリシーについて

監査ポリシーによって、ローカルシステムの監査レコードの特性が決定されます。これらのポリシーを設定するには、auditconfig コマンドを使用します。詳細は、auditconfig(1M) のマニュアルページを参照してください。

ストレージの要件やシステム処理への要求を最小限に抑えるために、デフォルトでは、ほとんどの監査ポリシーオプションが無効になっています。これらのオプションは監査サービスのプロパティーであり、システムブート時に有効になるポリシーを決定します。詳細は、auditconfig(1M) のマニュアルページを参照してください。

次の表を参照して、1 つまたは複数の監査ポリシーオプションを有効にしたときに発生する追加のオーバーヘッドを考慮しながら、サイトの要件を決定してください。

表 3 監査ポリシーオプションの働き

ポリシー名	説明	ポリシーに関する考慮事項
ahlt	非同期イベントにだけ適用されます。無効にすると、監査レコードが生成されないまま、イベントを完了できます。	システムの可用性の方がセキュリティーよりも重要な場合は、このオプションを無効にすることをお勧めします。
	有効になっている場合、このポリシーは、監査キューがいっぱいになるとシステムを停止します。監査キューの再配置、監査レコードの空き容量の確保、およびリブートは管理者の介入が必要です。このポリシーは、大域ゾーンでのみ有効にすることができます。ポリシーはすべてのゾーンに影響します。	セキュリティーを最優先にする環境では、このオプションを有効にすることをお勧めします。詳細は、124 ページの「非同期イベントおよび同期イベントの監査ポリシー」を参照してください。

ポリシー名	説明	ポリシーに関する考慮事項
arge	無効になっている場合、このポリシーは、実行されたプログラムの環境変数を execve 監査レコードから除外します。 有効になっている場合、このポリシーは、実行されたプログラムの環境変数を execve 監査レコードに追加します。監査レコードには、より詳細な情報が記録されます。	無効にすると、収集される情報が大幅に少なくなります。比較については、63 ページの「ユーザーによるすべてのコマンドを監査する方法」を参照してください。 少数のユーザーを監査する場合は、このオプションを有効にすることをお勧めします。このオプションはまた、ex 監査クラス内のプログラムで使用されている環境変数が不確かな場合にも役立ちます。
argv	無効になっている場合、このポリシーは、実行されたプログラムの引数を execve 監査レコードから除外します。 有効になっている場合、このポリシーは、実行されたプログラムの引数を execve 監査レコードに追加します。監査レコードには、より詳細な情報が記録されます。	無効にすると、収集される情報が大幅に少なくなります。比較については、63 ページの「ユーザーによるすべてのコマンドを監査する方法」を参照してください。 少数のユーザーを監査する場合は、このオプションを有効にすることをお勧めします。このオプションはまた、ex 監査クラス内の異常なプログラムが実行されていると考えられる理由がある場合にも役立ちます。
cnt	無効にすると、ユーザーまたはアプリケーションの実行がブロックされます。このブロックは、監査キューがいっぱいになったために、監査レコードを監査トレールに追加できない場合に実行されます。 有効にすると、監査レコードが生成されないまま、イベントを完了できます。生成されなかった監査レコードのカウントは行われます。	セキュリティーを最優先にする環境では、このオプションを無効にすることをお勧めします。 システムの可用性の方がセキュリティーよりも重要な場合は、このオプションを有効にすることをお勧めします。詳細は、124 ページの「非同期イベントおよび同期イベントの監査ポリシー」を参照してください。
group	無効にすると、グループの一覧が監査レコードに追加されません。 有効にすると、グループの一覧が特別なトークンとしてすべての監査レコードに追加されます。	通常は無効にしてもサイトのセキュリティー要件は満たします。 サブジェクトがどの補助グループに属しているかを監査する必要がある場合は、このオプションを有効にすることをお勧めします。
path	無効にすると、1 つのシステムコールで使用されたパスが、あっても 1 つだけ監査レコードに記録されます。 有効にすると、監査イベントで使用されたすべてのパスが、すべての監査レコードに記録されます。	無効にすると、監査レコードにパスが、あっても 1 つだけ記録されます。 有効にすると、1 つのシステムコールで使用された各ファイル名またはパスが、監査レコードに path トークンとして記録されます。
perzone	無効にすると、システムの単一の監査構成を保守します。大域ゾーンで 1 つの監査サービスが実行されます。zonename 監査トークンが事前選択された場合は、特定のゾーン内の	各ゾーンごとに監査ログ、キュー、およびデーモンを保守する理由が特にない場合は、無効にすると便利です。

ポリシー名	説明	ポリシーに関する考慮事項
	監査イベントを監査レコード内に配置できます。 有効になっている場合、このポリシーは、ゾーンごとに個別の監査構成、監査キュー、および監査ログを保持します。各ゾーンで監査サービスが実行されます。大域ゾーンでだけ有効にできます。	有効になったオプションは、zonename 監査トークンを含む監査レコードを検査するだけではシステムを効率的にモニターできない場合に役立ちます。
public	無効にすると、ファイルの読み取りが事前に選択されている場合に、公開オブジェクトの読み取り専用イベントが監査トレールに追加されなくなります。読み取り専用イベントを含む監査クラスとしては、fr、fa、および cl があります。 有効にすると、適切な監査クラスが事前に選択されている場合、公開オブジェクトの読み取り専用監査イベントのすべてが記録されます。	通常は無効にしてもサイトのセキュリティー要件は満たします。 このオプションを有効にするのはまれです。
seq	無効にすると、すべての監査レコードに順序番号が追加されません。 有効にすると、すべての監査レコードに順序番号が追加されます。順序番号は sequence トークンに格納されます。	監査が問題なく動作しているときは、無効にしても構いません。 cnt ポリシーが有効になっている場合は、このオプションを有効にすることをお勧めします。seq ポリシーを使用すると、データがいつ破棄されたかを判定できます。また、auditstat コマンドを使用すると、破棄されたレコードを表示することもできます。
trail	無効にすると、trailer トークンが監査レコードに追加されません。 有効にすると、trailer トークンがすべての監査レコードに追加されます。	無効にすると、作成される監査レコードが小さくなります。 有効にすると、各監査レコードの最後に trailer トークンが常に付加されます。trailer トークンは多くの場合、sequence トークンとともに使用されます。trailer トークンは、破損した監査トレールの回復に役立ちます。
zonename	無効にすると、zonename トークンが監査レコードに含まれません。 有効になっている場合、このポリシーでは、すべての監査レコード内に zonename トークンが含まれます。	無効になったオプションは、ゾーンごとに監査動作を追跡する必要がない場合に役立ちます。 有効になったオプションは、ゾーンに応じてレコードを事後選択することによって、ゾーン間で監査動作を分離したり、比較したりする場合に役立ちます。

監査コストの制御

監査処理によってシステムリソースが消費されるため、どの程度詳しく記録するかを制御する必要があります。監査の対象を決めるときには、監査に伴う次の 3 つのコストを考慮してください。

- 処理時間の増大に伴うコスト
- 監査データの分析に伴うコスト

デフォルトのプラグイン `audit_binfile` を使用している場合は、監査データのストレージコストも考慮する必要があります。

監査データの処理時間の増大に伴うコスト

処理時間の増大に伴うコストは、監査に関連する 3 つのコストの中ではもっとも重要性の低い問題です。通常、イメージ処理や複雑な計算処理などの計算集中型のタスクの実行中には、監査処理が発生しません。また、`audit_binfile` プラグインを使用している場合は、監査管理者が、事後選択のタスクを監査対象システムから監査データの分析専用のシステムに移動できます。最後に、カーネルイベントが事前選択されていないかぎり、監査サービスがシステム性能に測定可能な影響を与えることはありません。

監査データの分析に伴うコスト

分析に伴うコストは、収集される監査データの量にほぼ比例します。分析コストには、監査レコードをマージして検討するための時間が含まれます。

`audit_binfile` プラグインで収集されるレコードの場合は、レコードやそれをサポートしているネームサービスデータベースをアーカイブしたり、それらのレコードを安全な場所に保管したりするために必要な時間もコストに含まれます。サポートしているデータベースには、`groups`、`hosts`、および `passwd` が含まれています。

生成されるレコードの数が少ないほど、監査トレールの分析にかかる時間も短くなります。以降のセクション、43 ページの「監査データの格納に伴うコスト」および43 ページの「効率的な監査」では、効率的に監査する方法について説明します。効果的な監査では、収集するデータの量を削減しながら、サイトのセキュリティー目標を達成します。

監査データの格納に伴うコスト

audit_binfile プラグインを使用している場合は、ストレージコストが監査のもっとも大きなコストになります。監査データの量は次の要素によって左右されます。

- ユーザー数
- システム数
- 使用量
- 要求される追跡容易性と説明義務の程度

これらの要因はサイトごとに異なるため、監査データの格納用に前もって確保しておくディスク容量を決定できるような計算式はありません。次の情報を参考にしてください。

- 監査クラスを理解します。

  監査を構成する前に、クラスに含まれるイベントの種類を理解しておく必要があります。監査のイベントからクラスへのマッピングを変更すると、監査レコード収集を最適化できます。
- 監査クラスを慎重に事前選択し、生成されるレコードの量を減らします。

  完全な監査、つまり all クラスを使用した監査では、ディスク容量がすぐにいっぱいになります。プログラムのコンパイルといった単純なタスクによってさえ、巨大な監査ファイルが生成される可能性があります。中程度のサイズのプログラムでも、1 分も経たないうちに数千件の監査レコードが生成される可能性があります。

  たとえば、file_read 監査クラス fr を省くと、監査ボリュームを著しく削減できます。失敗した処理だけの監査を選択すると、監査ボリュームが減ることもあります。たとえば、失敗した file_read 処理 -fr のみを監査すると、すべての file_read イベントの監査よりも生成されるレコードを大幅に少なくすることができます。
- audit_binfile プラグインを使用している場合は、監査ファイルの効率的な管理も重要です。たとえば、監査ファイル専用の ZFS ファイルシステムを圧縮できます。
- サイトの監査方針を策定します。

  サイトで必要な追跡容易性の程度や、管理対象ユーザーの種類などの基準に基づいて、方針を策定します。

効率的な監査

次の方法により、組織のセキュリティー目標を達成する一方で、監査効率を高めることができます。

- できるだけ多くの監査クラスについて、システム全体のクラスではなく、ユーザーと役割のクラスのみを事前選択します。

- 一回にある一定の割合のユーザーのみをランダムに監査します。
- `audit_binfile` プラグインがアクティブな場合は、ファイルのフィルタリング、マージ、および圧縮によって、監査ファイルに対するディスクストレージの要件を軽減します。ファイルを保管する手順、リムーバブルメディアにファイルを転送する手順、およびファイルをオフラインで格納する手順を決定します。
- 監査データの異常な動作をリアルタイムでモニタリングします。
  - `audit_syslog` プラグイン – すでに開発している管理および分析ツールを、`syslog` ファイル内の監査レコードを処理するように拡張できます。
  - `audit_binfile` プラグイン – 特定の動作の監査トレールをモニターするための手順を設定できます。異常なイベントが検出された場合に、それに応じて特定のユーザーまたは特定のシステムの監査レベルを自動的に上げるようなスクリプトを作成します。

    たとえば、次のようにスクリプトを作成します。

    1. 監査対象システム上での監査ファイルの作成をモニターします。
    2. `tail` コマンドを使用して、監査ファイルを処理します。

       `tail -0f` コマンドから `praudit` コマンドに出力をパイプすることにより、レコードが生成されたときに監査レコードのストリームを生成できます。詳細は、`tail(1)` のマニュアルページを参照してください。
    3. このストリームを分析して異常なメッセージの種類やほかの兆候を調べ、または監査担当者に分析を配信します。

       また、このスクリプトを使用して、自動応答を発生させることもできます。
    4. 新しい `not_terminated` 監査ファイルが発生していないかどうか、監査ファイルシステムを常時モニターします。
    5. 監査ファイルに書き込めなくなったときに、未処理の `tail` プロセスを終了します。

♦ ♦ ♦ 第 3 章

3 監査サービスの管理

この章では、Oracle Solaris システム上での監査の構成や管理に役立つ手順について説明します。この章では、次のタスクについて説明します。

- 45 ページの「監査サービスのデフォルト構成」
- 48 ページの「監査サービスの構成」
- 63 ページの「監査対象のカスタマイズ」
- 72 ページの「ゾーンでの監査サービスの構成」
- 76 ページの「例: Oracle Solaris 監査の構成」

さらに、次の章では、その他の監査管理タスクについて説明します。

- 第4章「システム動作のモニタリング」
- 第5章「監査データの操作」
- 第6章「監査サービスに関する問題の分析および解決」

監査サービスの概要については、第1章「Oracle Solaris での監査について」を参照してください。計画の提案については、第2章「監査の計画」を参照してください。参照情報については、第7章「監査の参照情報」を参照してください。

監査サービスのデフォルト構成

監査サービスにはデフォルト構成があるため、Oracle Solaris をインストールした直後に、大域ゾーン上で動作可能になります。サービスを有効にしたり、使用可能になるように構成したりするために、追加のアクションは必要ありません。監査サービスのデフォルト構成を使用すると、次の操作が記録されます。

- ログインおよびログアウトの操作
- su コマンドの使用
- 画面ロックおよび画面ロック解除の操作

サービスのデフォルト構成はシステムのパフォーマンスに影響を与えないため、パフォーマンス上の理由でサービスを無効にする必要はありません。

適切な監査関連の権利 (Audit Review 権利プロファイルの権利など) を持っていれば、監査ログを確認できます。ログは `/var/audit` に格納されます。これらのファイルは、`praudit` および `auditreduce` コマンドを使用して表示します。詳細は、95 ページの「監査トレールデータの表示」を参照してください。

この章の以降のセクションでは、デフォルト構成ではユーザーのニーズを満たすのに不十分である場合に、監査サービスの構成をカスタマイズする手順について説明します。

監査サービスのデフォルトの表示

監査サービスは、次のパラメータによって規制されます。

- ユーザーに起因するイベントおよびユーザーに起因しないイベントのクラス
- 監査ポリシー
- 監査プラグイン
- キュー制御

監査サービスのデフォルトを表示するには、通常、`auditconfig -get*` サブコマンドを使用します。これらのサブコマンドは、アスタリスク (*) で表したパラメータ (`-getflags`、`-getpolicy`、`-getqctrl` など) の現在の構成を表示します。ユーザーに起因しないイベントのクラスに関する情報を表示するには、`auditconfig -getnaflags` サブコマンドを使用します。

`auditconfig` コマンドの詳細は、`auditconfig(1M)` のマニュアルページを参照してください。

注記 - 監査サービスの構成を表示するには、Audit Configuration または Audit Control 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 11.3 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。

次の例では、監査構成のデフォルト設定を表示するために使用する適切なコマンド構文を示します。

例 1 イベントのデフォルトクラスを表示する

この例では、ユーザーに起因するイベントおよびユーザーに起因しないイベントの事前選択されたクラスを表示するために、2 つのサブコマンドが使用されています。クラスにどのイベントが割り当てられているか、したがってどのイベントが記録されるかを表示するには、`auditrecord -c` *class* コマンドを実行します。

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)
```

lo は、login/logout 監査クラスのフラグです。マスク出力の形式は (*success*,*failure*) です。

```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

例 2 デフォルトの監査ポリシーを表示する

```
$ auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
```

アクティブなポリシーは現在のポリシーですが、このポリシーの値は監査サービスによって格納されていません。構成されたポリシーは監査サービスによって格納されるため、このポリシーは監査サービスを再開したときに復元されます。

例 3 デフォルトの監査プラグインを表示する

```
$ auditconfig -getplugin
Plugin: audit_binfile
Attributes: p_age=Oh;p_dir=/var/audit;p_fsize=4M;p_minfree=1;

Plugin: audit_syslog (inactive)
Attributes: p_flags=;

Plugin: audit_remote (inactive)
Attributes: p_hosts=;p_retries=3;p_timeout=5;
```

デフォルトでは、audit_binfile プラグインがアクティブです。

例 4 監査キュー制御を表示する

```
$ auditconfig -getqctrl
no configured audit queue hiwater mark
no configured audit queue lowater mark
no configured audit queue buffer size
no configured audit queue delay
active audit queue hiwater mark (records) = 100
active audit queue lowater mark (records) = 10
active audit queue buffer size (bytes) = 8192
active audit queue delay (ticks) = 20
```

アクティブなキュー制御は、カーネルによって現在使用されているキュー制御です。no configured の文字列は、システムがデフォルト値を使用していることを示します。

監査サービスの有効化および無効化

監査サービスは、デフォルトで有効になっています。perzone 監査ポリシーが設定されている場合、ゾーン管理者は必要に応じて、各非大域ゾーン内の監査サービスを有効にしたり、リフレッシュしたり、無効にしたりする必要があります。perzone 監査ポリシーが設定されていない場合は、大域ゾーンからすべての非大域ゾーンの監査サービスを有効にしたり、リフレッシュしたり、無効にしたりすると効率的です。

監査サービスを無効または有効にするには、Audit Control 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 11.3 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。

監査サービスを無効にするには、次のコマンドを使用します。

```
# audit -t
```

監査サービスを有効にするには、次のコマンドを使用します。

```
# audit -s
```

監査サービスが動作していることを確認するには、次のコマンドを使用します。

```
# auditconfig -getcond
audit condition = auditing
```

perzone 監査ポリシーが設定されている場合は、監査を有効にした非大域ゾーンで、この確認を実行する必要があります。

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

監査サービスの構成

ネットワーク上で監査を有効にする前に、サイトの監査要件を満たすようにデフォルト設定を変更できます。ベストプラクティスは、最初のユーザーがログインする前に、監査構成をできるだけカスタマイズすることです。

ゾーンを実装している場合は、大域ゾーンのすべてのゾーンを監査するか、または非大域ゾーンを個別に監査するかを選択できます。概要については、120 ページの「監査と Oracle Solaris Zones」を参照してください。計画については、32 ページの「ゾーン内での監査の計画」を参照してください。手順については、72 ページの「ゾーンでの監査サービスの構成」を参照してください。

監査サービスを構成するには、通常、auditconfig サブコマンドを使用します。これらのサブコマンドを使用して設定された構成は、システム全体に適用されます。

- auditconfig -get* は、アスタリスクで (*) 表したパラメータの現在の構成を表示します (46 ページの「監査サービスのデフォルトの表示」の例を参照)。
- auditconfig -set* は、アスタリスク (*) で表したパラメータ (-setflags、-setpolicy、-setqctrl など) に値を割り当てます。ユーザーに起因しないイベントのクラスを構成するには、auditconfig setnaflags サブコマンドを使用します。
- auditconfig -conf は、カーネル監査イベントからクラスへのマッピングを構成します。実行時クラスマッピングは、監査イベント - クラスデータベースファイル内のマッピングに一致するように変更されます。

システム全体ではなく、ユーザーまたはプロファイルに適用されるように、監査をカスタマイズすることもできます。ユーザーごとの監査クラスの事前選択は、audit_flags セキュリティー属性によって指定されます。126 ページの「プロセスの監査特性」で説明されているように、これらのユーザー固有の値と、システムに対して事前選択されたクラスによって、そのユーザーの監査マスクが決定されます。

システム単位にではなく、ユーザー単位にクラスを事前選択することによって、システムパフォーマンスへの監査の影響を軽減できる場合があります。また、特定のユーザーを、システムとは若干異なる方法で監査することもできます。

ユーザーまたはプロファイルに適用される監査を構成するには、次のコマンドを使用します。

- usrattr は、ユーザーに設定されている audit_flags 値を表示します。デフォルトでは、ユーザーはシステム全体の設定に関してのみ監査されます。
- usermod -K は、ユーザーに適用されるフラグを設定します。
- profile は、プロファイルに適用されるフラグを設定します。

userattr コマンドについては、userattr(1) のマニュアルページを参照してください。audit_flags キーワードについては、user_attr(4) のマニュアルページおよび audit_flags(5) を参照してください。

次のタスクマップは、監査を構成するための手順を示しています。すべてのタスクがオプションです。

表 4 監査サービスの構成タスクマップ

タスク	説明	手順
どのイベントが監査されるかを選択します。	システム全体の監査クラスを事前選択します。イベントがユーザーに起因する場合は、すべてのユーザーがこのイベントに関して監査されます。	50 ページの「監査クラスを事前選択する方法」

タスク	説明	手順
特定のユーザーに対してどのイベントが監査されるを選択します。	システム全体の監査クラスからのユーザー固有の違いを設定します。	51 ページの「ユーザーの監査特性を構成する方法」
監査ポリシーを指定します。	サイトに必要な追加の監査データを定義します。	55 ページの「監査ポリシーを変更する方法」
キュー制御を指定します。	デフォルトのバッファーサイズ、キュー内の監査レコード数、および監査レコードのバッファーへの書き込みの間隔を変更します。	58 ページの「監査キュー制御を変更する方法」
audit_warn 電子メールエイリアスを作成します。	監査サービスに注意が必要になったときにだれが電子メール警告を受信するかを定義します。	59 ページの「audit_warn 電子メールエイリアスの構成方法」
監査ログを構成します。	プラグインごとの監査レコードの場所を構成します。	79 ページの「監査ログの構成」
監査クラスを追加します。	重要なイベントを保持する新しい監査クラスを作成することによって、監査レコードの数を減らします。	60 ページの「監査クラスの追加方法」
イベントからクラスへのマッピングを変更します。	イベントからクラスへのマッピングを変更して、監査レコードの数を減らします。	61 ページの「監査イベントの所属先クラスの変更方法」

▼ 監査クラスを事前選択する方法

モニターするイベントを含む監査クラスを事前選択します。事前選択されたクラスに含まれていないイベントは記録されません。

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 11.3 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. **現在の事前選択されたクラスを確認します。**

```
# auditconfig -getflags
...

# auditconfig -getnaflags
,,,
```

出力の説明については、46 ページの「監査サービスのデフォルトの表示」を参照してください。

2. **ユーザーに起因するクラスを事前選択します。**

```
# auditconfig -setflags lo,ps,fw
user default audit flags = ps,lo,fw(0x101002,0x101002)
```

このコマンドは、login/logout、process start/stop、および file write クラス内のイベントの成功と失敗について監査します。

注記 - auditconfig -setflags コマンドでは現在の事前選択が置き換えられるため、事前選択するすべてのクラスを指定する必要があります。

---

3. **ユーザーに起因しないクラスを事前選択します。**

   na クラスには、ほかのイベントに加えて、PROM、ブート、およびユーザーに起因しないマウントが含まれています。

```
# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

   -setnaflags オプションの有効な引数は lo と na だけです。

注記 - auditconfig -setnaflags コマンドでは現在の事前選択が置き換えられるため、事前選択するすべてのクラスを指定する必要があります。

▼ ユーザーの監査特性を構成する方法

この手順を使用して設定したユーザー固有の監査特性は、システム用に事前選択されたクラスと組み合わされます。126 ページの「プロセスの監査特性」で説明されているように、この組み合わせによってユーザーの監査マスクが決定されます。

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. **(オプション) 既存のユーザー用に現在選択されている監査クラスを表示します。**

   a. **ユーザーの一覧を表示します。**

```
# who
adoe    pts/1        Oct 10 10:20 (:0.0)
adoe    pts/2        Oct 10 10:20 (:0.0)
jdoe    pts/5        Oct 12 12:20 (:0.0)
jdoe    pts/6        Oct 12 12:20 (:0.0)
...
```

   b. **各ユーザーの audit_flags 属性値を表示します。**

```
# userattr audit_flags adoe
# userattr audit_flags jdoe
```

2. **user_attr または in prof_attr データベース内の監査フラグを設定します。**

たとえば、ユーザーのサブセットの権利を定義する権利プロファイルを作成できます。その権利プロファイルが割り当てられているユーザーは同様に監査されます。

- **ユーザーの監査フラグを設定するには、`usermod` コマンドを使用します。**

```
# usermod -K audit_flags=fw:no jdoe
```

`audit_flags` キーワードの形式は *always-audit*:*never-audit* です。

always-audit このユーザーについて監査される監査クラスを一覧表示します。

never-audit 監査イベントがシステム全体で監査される場合でも、そのユーザーについて監査されることのない監査クラスを一覧表示します。

複数の監査クラスを指定するには、クラスをコンマで区切ります。詳細は、`audit_flags(5)` のマニュアルページを参照してください。

- **権利プロファイルの監査フラグを設定するには、`profiles` コマンドを使用します。**

```
# profiles -p "System Administrator"
profiles:System Administrator> set name="Audited System Administrator"
profiles:Audited System Administrator> set always_audit=fw,as
profiles:Audited System Administrator> end
profiles:Audited System Administrator> exit
```

ユーザーまたは役割に Audited System Administrator 権利プロファイルを割り当てた場合、そのユーザーまたは役割は、『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 「割り当てられた権利の検索順序」で説明されている検索順序に従ってこれらのフラグについて監査されます。

例 5　1 人のユーザーに関して監査するイベントを変更する

この例では、すべてのユーザーの監査事前選択マスクは次のとおりです。

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

管理者を除き、どのユーザーもログインしていません。

`AUE_PFEXEC` 監査イベントのシステムリソースへの影響を軽減するために、管理者は、このイベントをシステムレベルでは監査しません。代わりに、管理者は、ユーザー `jdoe` の `pf` クラスを事前選択します。`pf` クラスは、例15「新しい監査クラスを作成する」で作成されます。

```
# usermod -K audit_flags=pf:no jdoe
```

`userattr` コマンドによって、この追加が表示されます。

```
# userattr audit_flags jdoe
pf:no
```

ユーザー jdoe がログインしたとき、jdoe の監査事前選択マスクは、audit_flags 値とシステムのデフォルト値の組み合わせになります。289 は、jdoe のログインシェルの PID です。

```
# auditconfig -getpinfo 289
audit id = jdoe(1234)
process preselection mask = ss,pf,lo(0x0100000008011000,0x0100000008011000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

例 6　1 人のユーザーに関する監査事前選択の例外を変更する

この例では、すべてのユーザーの監査事前選択マスクは次のとおりです。

```
# auditconfig -getflags
active user default audit flags = ss,lo(0x11000,0x11000)
configured user default audit flags = ss,lo(0x11000,0x11000)
```

管理者を除き、どのユーザーもログインしていません。

管理者は、jdoe ユーザーに関する失敗した ss イベントを収集しないことを決定しました。

```
# usermod -K audit_flags=^-ss:no jdoe
```

userattr コマンドによって、この例外が表示されます。

```
# userattr audit_flags jdoe
^-ss:no
```

ユーザー jdoe がログインしたとき、jdoe の監査事前選択マスクは、audit_flags 値とシステムのデフォルト値の組み合わせになります。289 は、jdoe のログインシェルの PID です。

```
# auditconfig -getpinfo 289
audit id = jdoe(1234)
process preselection mask = +ss,lo(0x11000,0x1000)
terminal id (maj,min,host) = 242,511,example1(192.168.160.171)
audit session id = 103203403
```

例 7　選択されたユーザーを監査する (システム全体の監査はなし)

この例では、選択された 4 人のユーザーのシステム上でのログインと役割活動が監査されます。システムに対して監査クラスは事前選択されていません。

最初に、管理者は、システム全体のフラグをすべて削除します。

```
# auditconfig -setflags no
```

```
user default audit flags = no(0x0,0x0)
```

次に、管理者は、4 人のユーザーに対して 2 つの監査クラスを事前選択します。pf クラスは、例15「新しい監査クラスを作成する」で作成されます。

```
# usermod -K audit_flags=lo,pf:no jdoe
# usermod -K audit_flags=lo,pf:no kdoe
# usermod -K audit_flags=lo,pf:no pdoe
# usermod -K audit_flags=lo,pf:no zdoe
```

次に、管理者は、root 役割に対して pf クラスを事前選択します。

```
# userattr audit_flags root
# rolemod -K audit_flags=lo,pf:no root
# userattr audit_flags root
lo,pf:no
```

不当な侵入を記録し続けるために、管理者は、ユーザーに起因しないログインの監査を変更しません。

```
# auditconfig -getnaflags
active non-attributable audit flags = lo(0x1000,0x1000)
configured non-attributable audit flags = lo(0x1000,0x1000)
```

例 8　ユーザーの監査フラグを削除する

次の例では、管理者がユーザー固有の監査フラグをすべて削除します。現在ログインしているユーザーの既存のプロセスが引き続き監査されます。

管理者は、audit_flags キーワードに値を設定せずに usermod コマンドを実行します。

```
# usermod -K audit_flags= jdoe
# usermod -K audit_flags= kdoe
# usermod -K audit_flags= ldoe
```

次に、管理者はこの削除を確認します。

```
# userattr audit_flags jdoe
# userattr audit_flags kdoe
# userattr audit_flags ldoe
```

例 9　ユーザーのグループに対する権利プロファイルを作成する

管理者は、サイトにある管理上の権利プロファイルのすべてで pf クラスが明示的に監査されるようにしたいと考えています。割り当てを予定している権利プロファイルごとに、管理者は、監査フラグを含むサイト固有のバージョンを LDAP 内に作成します。

最初に、管理者は既存の権利プロファイルを複製し、次に、名前を変更して監査フラグを追加します。

```
# profiles -p "Network Wifi Management" -S ldap
profiles: Network Wifi Management> set name="Wifi Management"
profiles: Wifi Management> set desc="Audited wifi management"
profiles: Wifi Management> set audit_always=pf
profiles: Wifi Management> exit
```

使用を予定している権利プロファイルごとにこの手順を繰り返したあと、管理者は、Wifi Management プロファイル内の情報を一覧表示します。

```
# profiles -p "Wifi Management" -S ldap info
name=Wifi Management
desc=Audited wifi management
auths=solaris.network.wifi.config
help=RtNetWifiMngmnt.html
always_audit=pf
```

▼ 監査ポリシーを変更する方法

デフォルトの監査ポリシーを変更することにより、監査されるコマンドに関する詳細情報を記録したり、すべてのレコードにゾーン名を追加したり、サイトのほかのセキュリティー要件を満たしたりすることができます。

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 11.3 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. **現在の監査ポリシーを表示します。**

```
$ auditconfig -getpolicy
configured audit policies = argv,cnt
active audit policies = argv,cnt
```

出力の説明については、46 ページの「監査サービスのデフォルトの表示」を参照してください。

2. **使用可能なポリシーオプションを表示します。**

```
$ auditconfig -lspolicy
policy string    description:
ahlt             halt machine if it can not record an async event
all              all policies for the zone
arge             include exec environment args in audit recs
argv             include exec command line args in audit recs
cnt              when no more space, drop recs and keep a cnt
group            include supplementary groups in audit recs
none             no policies
path             allow multiple paths per event
perzone          use a separate queue and auditd per zone
public           audit public files
seq              include a sequence number in audit recs
trail            include trailer token in audit recs
windata_down     include downgraded window information in audit recs
```

```
windata_up       include upgraded window information in audit recs
zonename         include zonename token in audit recs
```

注記 - perzone と ahlt ポリシーオプションは、大域ゾーンでのみ設定できます。特定のポリシーオプションを使用した場合のトレードオフについては、39 ページの「監査ポリシーについて」を参照してください。

3. **選択した監査ポリシーオプションを有効または無効にします。**

```
# auditconfig [ -t ] -setpolicy [prefix]policy[,policy...]
```

-t　オプション。一時的な (アクティブな) ポリシーを作成します。一時的なポリシーは、デバッグまたはテストの目的で設定できます。

一時的なポリシーは、監査サービスがリフレッシュされるか、または auditconfig -setpolicy コマンドによってポリシーが変更されるか、システムがリブートされるまで有効です。

prefix　*prefix* の値が + のときは、ポリシーのリストが現在のポリシーに追加されます。*prefix* の値が - のときは、ポリシーのリストが現在のポリシーから削除されます。接頭辞を指定しない場合は、監査ポリシーがリセットされます。このオプションを使用すると、現在の監査ポリシーを保持できます。

policy　有効または無効にするポリシーを選択します。

例 10　ahlt 監査ポリシーオプションを設定する

この例では、厳格なサイトセキュリティーには ahlt ポリシーが必要です。

```
# auditconfig -setpolicy -cnt
# auditconfig -setpolicy +ahlt
```

ahlt ポリシーの前にあるプラス記号 (+) により、このポリシーが現在のポリシー設定に追加されます。プラス記号がない場合は、ahlt ポリシーによって現在の監査ポリシーがすべて置き換えられます。

例 11　一時的な監査ポリシーを設定する

この例では、ahlt 監査ポリシーが構成されます。デバッグのために、管理者は trail 監査ポリシーをアクティブなポリシーに (+trail) 一時的に (-t) 追加します。trail ポリシーは、破損した監査トレールの回復に役立ちます。

```
$ auditconfig -setpolicy ahlt
$ auditconfig -getpolicy
configured audit policies = ahlt
```

```
active audit policies = ahlt
$ auditconfig -t -setpolicy +trail
$ auditconfig -getpolicy
configured audit policies = ahlt
active audit policies = ahlt,trail
```

管理者は、デバッグが完了すると `trail` ポリシーを無効にします。

```
$ auditconfig -setpolicy -trail
$ auditconfig -getpolicy
configured audit policies = ahlt
active audit policies = ahlt
```

また、`audit -s` コマンドを実行して監査サービスをリフレッシュした場合も、この一時的なポリシーに加え、監査サービス内のほかの一時的な値がすべて削除されます。ほかの一時的な値の例については、58 ページの「監査キュー制御を変更する方法」を参照してください。

例 12 perzone 監査ポリシーを設定する

この例では、`perzone` 監査ポリシーが大域ゾーン内の既存のポリシーに追加されます。`perzone` ポリシー設定は永続的なプロパティーとして格納されるため、`perzone` ポリシーはセッション中と、監査サービスが再開されたときに有効になります。ゾーンの場合、ポリシーは、次のゾーンのブート時、または管理者が `zlogin` を実行してから `audit -s` コマンドを実行したときに使用可能になります。

```
$ auditconfig -getpolicy
configured audit policies = cnt
active audit policies = cnt
$ auditconfig -setpolicy +perzone
$ auditconfig -getpolicy
configured audit policies = perzone,cnt
active audit policies = perzone,cnt
```

例 13 外部監査の監査レコードを収集する

この例では、管理者が外部監査人の要件を満たす監査レコードを収集しています。管理者は、監査リモートサーバー (ARS) を使用して管理アクティビティーに関する情報を収集することに決定します。管理者は、ユーザーに起因できないアクション (ブートなど) も収集します。

管理者は、ARS を設定します。管理者は `cusa` クラスを監査することに加えて、監査の構成にポリシーを追加します。

```
# auditconfig -setflags cusa
user default audit flags = ex,xa,ua,as,ss,ap,lo,ft(0x80475080,0x80475080)
# auditconfig -setpolicy ahlt,argv,arge
# auditconfig -getpolicy
configured audit policies = ahlt,arge,argv
active audit policies = ahlt,arge,argv
# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

管理者が audit_remote プラグインを有効にして、監査サービスをリフレッシュすると、レコードが収集されます。

▼ 監査キュー制御を変更する方法

監査サービスは、監査キューパラメータのデフォルト値を提供します。auditconfig コマンドを使用すると、これらの値を検査、永続的に変更、および一時的に変更することができます。

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 11.3 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. **監査キュー制御の現在の値を表示します。**

```
$ auditconfig -getqctrl
...
```

出力の説明については、46 ページの「監査サービスのデフォルトの表示」を参照してください。

2. **選択された監査キュー制御を変更します。**

監査キュー制御の例と説明については、auditconfig(1M) のマニュアルページを参照してください。

- 一部またはすべての監査キュー制御を変更するには、-setqctrl オプションを使用します。

```
# auditconfig [ -t ] -setqctrl hiwater lowater bufsz interval
```

高水位 (hiwater) の値と低水位 (lowater) の値は、それぞれ、プロセスが一時停止される時点と再開される時点を示します。この時点は、未配信の監査レコード数の観点から測定されます。バッファーサイズ (bufsz) は、キューのバッファーサイズを表します。Interval は、監査出力の生成間の遅延 (ティック数で測定) を示します。

たとえば、ほかの制御を設定することなく *interval* 値を 10 に設定します。

```
# auditconfig -setqctrl 0 0 0 10
```

- 特定の監査キュー制御を変更するには、そのオプションを指定します。-setqdelay オプションは、**auditconfig -setqdelay 10** のように、-setqctrl 0 0 0 *interval* と同等です。

```
# auditconfig [ -t ] -setqhiwater value
# auditconfig [ -t ] -setqlowater value
# auditconfig [ -t ] -setqbufsz value
```

```
# auditconfig [ -t ] -setqdelay value
```

例 14 監査キュー制御をデフォルトにリセットする

管理者は、すべての監査キュー制御を設定したあと、リポジトリ内の *lowater* 値を元のデフォルトに変更します。

```
# auditconfig -setqctrl 200 5 10216 10
# auditconfig -setqctrl 200 0 10216 10
configured audit queue hiwater mark (records) = 200
no configured audit queue lowater mark
configured audit queue buffer size (bytes) = 10216
configured audit queue delay (ticks) = 10
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 5
active audit queue buffer size (bytes) = 10216
active audit queue delay (ticks) = 10
```

あとで、管理者は *lowater* 値を現在のセッションのデフォルトに設定します。

```
# auditconfig -setqlowater 10
# auditconfig -getqlowater
configured audit queue lowater mark (records) = 10
active audit queue lowater mark (records) = 10
```

▼ audit_warn 電子メールエイリアスの構成方法

/etc/security/audit_warn スクリプトは、注意を要する可能性のある監査インシデントを管理者に通知するためのメールを生成します。このスクリプトをカスタマイズしたり、root 以外のアカウントにメールを送信したりできます。

perzone ポリシーが設定されている場合、非大域ゾーン管理者は、非大域ゾーンで audit_warn 電子メールエイリアスを構成する必要があります。

始める前に solaris.admin.edit/etc/security/audit_warn 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、この承認を持つのは root 役割だけです。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。

- **audit_warn 電子メールエイリアスを構成します。**

  次のオプションのいずれかを選択します。

  - audit_warn スクリプトで、audit_warn 電子メールエイリアスをほかの電子メールアカウントに置き換えます。

    スクリプトの ADDRESS 行にある audit_warn 電子メールエイリアスを別のアドレスに変更します。

    ```
    #ADDRESS=audit_warn            # standard alias for audit alerts
    ADDRESS=audadmin               # role alias for audit alerts
    ```

注記 - 監査構成ファイルの変更の影響については、121 ページの「監査構成ファイルとパッケージ化」を参照してください。

- `audit_warn` 電子メールをほかの電子メールアカウントにリダイレクトします。

  `audit_warn` 電子メールエイリアスを適切なメールエイリアスファイルに追加します。別名をローカルの `/etc/mail/aliases` ファイルか、またはネームサービスの `mail_aliases` データベースに追加できます。`audit_warn` 電子メールエイリアスのメンバーとして `root` と `audadmin` の電子メールアカウントが追加された場合、`/etc/mail/aliases` エントリは次の例のようになります。

  ```
  audit_warn: root,audadmin
  ```

  次に、`newaliases` コマンドを実行して、`aliases` ファイルのためのランダムアクセスデータベースを再構築します。

  ```
  # newaliases
  /etc/mail/aliases: 14 aliases, longest 10 bytes, 156 bytes total
  ```

▼ 監査クラスの追加方法

独自の監査クラスを作成すると、サイトで監視する監査イベントだけをそのクラスに入れることができます。この方法により、収集されるレコード数を削減し、さらに監査トレール内のノイズも削減することができます。

1 つのシステム上でクラスを追加した場合は、監査されているすべてのシステムにその変更をコピーします。ベストプラクティスとして、最初のユーザーがログインする前に監査クラスを作成してください。

監査構成ファイルの変更の影響については、121 ページの「監査構成ファイルとパッケージ化」を参照してください。

ヒント - Oracle Solaris では、ファイルが含まれる独自のパッケージを作成したり、Oracle Solaris パッケージをサイトでカスタマイズされたファイルで置き換えたりすることができます。パッケージ内で `preserve` 属性を `true` に設定すると、`pkg` サブコマンド (verify、fix、revert など) は、そのパッケージを基準にして動作します。詳細については、`pkg`(1) および `pkg`(5) のマニュアルページを参照してください。

始める前に 一意のエントリのための空きビットを上位 8 ビットで選択します。顧客がどのビットを使用できるかを `/etc/security/audit_class` ファイルで確認します。

`solaris.admin.edit/etc/security/audit_class` 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、この承認を持つのは `root` 役割だけです。

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. **(オプション) `audit_class` ファイルのバックアップコピーを保存します。**

```
# cp /etc/security/audit_class /etc/security/audit_class.orig
```

2. **`audit_class` ファイルに新しいエントリを追加します。**

各エントリの書式は次のとおりです。

`0x`*64bitnumber*`:`*flag*`:`*description*

各フィールドについては、`audit_class(4)` のマニュアルページを参照してください。既存のクラスの一覧については、`/etc/security/audit_class` ファイルを参照してください。

例 15 新しい監査クラスを作成する

この例では、ある役割で実行される管理コマンドを保持するためのクラスを作成します。`audit_class` ファイルに追加されるエントリは次のとおりです。

```
0x0100000000000000:pf:profile command
```

このエントリによって、新しい `pf` 監査クラスが作成されます。例16「既存の監査イベントを新しいクラスにマッピングする」は、新しい監査クラスを生成する方法を示しています。

注意事項 `audit_class` ファイルをカスタマイズした場合は、ユーザーまたは権利プロファイルに直接割り当てられている監査フラグがすべて、新しい監査クラスと整合性がとれていることを確認してください。`audit_flags` 値が `audit_class` ファイルのサブセットでない場合は、エラーが発生します。

▼ 監査イベントの所属先クラスの変更方法

既存の監査クラスのサイズを減らしたり、独自のクラスにイベントを入れたりするために、監査イベントのクラスメンバーシップを変更できます。

注意 - `audit_event` ファイルではイベントをコメントにしないでください。このファイルは、`praudit` コマンドがバイナリ監査ファイルを読み取るときに使用します。また、このファイルに一覧表示されたイベントが、保管された監査ファイルに含まれることがあります。また、既存の監査クラスを削除しないでください。

1 つのシステム上で監査イベントからクラスへのマッピングを再構成した場合は、その変更を、監査されているすべてのシステムにコピーします。ベストプラクティスと

して、最初のユーザーがログインする前に、イベントからクラスへのマッピングを変更してください。

注記 - 監査構成ファイルの変更の影響については、121 ページの「監査構成ファイルとパッケージ化」を参照してください。

ヒント - Oracle Solaris では、ファイルが含まれる独自のパッケージを作成したり、Oracle Solaris パッケージをサイトでカスタマイズされたファイルで置き換えたりすることができます。パッケージ内で `preserve` 属性を `true` に設定すると、`pkg` サブコマンド (verify、fix、revert など) は、そのパッケージを基準にして動作します。詳細については、`pkg`(1) および `pkg`(5) 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/security/audit_event` 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、この承認を持つのは `root` 役割だけです。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. **(オプション) `audit_event` ファイルのバックアップコピーを保存します。**

```
# cp /etc/security/audit_event /etc/security/audit_event.orig
```

2. **特定のイベントがどのクラスに属するかを変更するには、イベントの *class-list* を変更します。**

   各エントリの書式は次のとおりです。

   number:*name*:*description*:*class-list*

number	監査イベント ID。
name	監査イベントの名前。
description	通常、監査レコードの作成を発生させるシステムコールまたは実行可能プログラム。
class-list	コンマ区切りの監査クラスの一覧。

3. **カーネルイベントをリフレッシュします。**

```
# auditconfig -conf
Configured 283 kernel events.
```

例 16 既存の監査イベントを新しいクラスにマッピングする

この例では、既存の監査イベントを例15「新しい監査クラスを作成する」で作成された新しいクラスにマップします。デフォルトでは、AUE_PFEXEC 監査イベントは複数の監査クラスにマッピングされます。新しいクラスを作成することによって、管理者

は、ほかのクラス内のイベントを監査することなく AUE_PFEXEC イベントを監査できます。

```
# grep pf /etc/security/audit_class
0x0100000000000000:pf:profile command
# grep AUE_PFEXEC /etc/security/audit_event
116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as
# pfedit /etc/security/audit_event
#116:AUE_PFEXEC:execve(2) with pfexec enabled:ps,ex,ua,as
116:AUE_PFEXEC:execve(2) with pfexec enabled:pf
# auditconfig -setflags lo,pf
user default audit flags = pf,lo(0x0100000000001000,0x0100000000001000)
```

監査対象のカスタマイズ

次のタスクマップでは、ユーザーのニーズに固有の監査を構成するための手順を示します。

表 5 監査のカスタマイズタスクマップ

タスク	説明	手順
システムでのユーザーの操作をすべて監査する。	すべてのコマンドについて 1 人または複数のユーザーを監査します。	63 ページの「ユーザーによるすべてのコマンドを監査する方法」
記録される監査イベントを変更して、その変更内容を既存のセッションに適用する。	ユーザーの事前選択マスクを更新します。	67 ページの「ログインしているユーザーの事前選択マスクを更新する方法」
変更内容を特定のファイルに格納する。	ファイルの変更を監査し、auditreduce コマンドを使用して特定のファイルを検索します。	66 ページの「特定のファイルに対する変更の監査レコードを検索する方法」
監査ファイルに使用するファイルシステムの容量を減らす。	ZFS 割り当て制限と圧縮を使用します。	70 ページの「専用ファイルシステム上の監査ファイルを圧縮する方法」
audit_event ファイルから監査イベントを削除する。	audit_event ファイルを正しく更新します。	68 ページの「特定のイベントの監査を回避する方法」

▼ ユーザーによるすべてのコマンドを監査する方法

サイトのセキュリティーポリシーの一環として、サイトによっては、root アカウントや管理役割によって実行されるすべてのコマンドの監査レコードが必要になります。サイトによっては、すべてのユーザーによるすべてのコマンドの監査レコードが必要な場合があります。さらに、サイトでコマンド引数や環境を記録することが必要になる場合もあります。

始める前に 監査クラスを事前選択し、監査ポリシーを設定するには、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。監査フラグをユーザー、役割、および権利プロファイルに割り当てるには、権利委託プロファイルに含まれている `solaris.audit.assign` 権限を持っている必要があります。ルート役割はすべての権限を持ちます。

1. **`lo` および `ex` クラスに関するユーザーレベルのイベント情報を表示します。**

   `ex` クラスは、`exec()` 関数および `execve()` 関数のすべての呼び出しを監査します。

   `lo` クラスは、ログイン、ログアウト、および画面ロックを監査します。次の出力は、`ex` および `lo` クラス内のすべてのイベントを一覧表示しています。

```
% auditconfig -lsevent | egrep " lo |,lo|lo,"
AUE_login                       6152 lo login - local
AUE_logout                      6153 lo logout
AUE_telnet                      6154 lo login - telnet
AUE_rlogin                      6155 lo login - rlogin
AUE_rshd                        6158 lo rsh access
AUE_su                          6159 lo su
AUE_rexecd                      6162 lo rexecd
AUE_passwd                      6163 lo passwd
AUE_rexd                        6164 lo rexd
AUE_ftpd                        6165 lo ftp access
AUE_ftpd_logout                 6171 lo ftp logout
AUE_ssh                         6172 lo login - ssh
AUE_role_login                  6173 lo role login
AUE_rad_login                   6174 lo connect to RAD
AUE_newgrp_login                6212 lo newgrp login
AUE_admin_authenticate          6213 lo admin login
AUE_screenlock                  6221 lo screenlock - lock
AUE_screenunlock                6222 lo screenlock - unlock
AUE_zlogin                      6227 lo login - zlogin
AUE_su_logout                   6228 lo su logout
AUE_role_logout                 6229 lo role logout
AUE_smbd_session                6244 lo smbd(1m) session setup
AUE_smbd_logoff                 6245 lo smbd(1m) session logoffAUE_sudo
    6650 lo,as,ua sudo(1M) execution


% auditconfig -lsevent | egrep " ex |,ex |ex,"
AUE_EXECVE                       23 ex,ps execve(2)
AUE_PFEXEC                       116, ex,ps,ua,as execve(2) with pfexec enabled
```

2. **管理者の `cusa` クラスを監査します。**

   - **管理役割に対してこれらのクラスを監査するには、それらの役割のセキュリティー属性を変更します。**

     次の例で、`root` は役割です。このサイトでは、`sysadm`、`auditadm`、および `netadm` の 3 つの役割が作成されています。すべての役割で、`cusa` クラスのイベントの成功と失敗が監査されます。

```
# rolemod -K audit_flags=cusa:no root

# rolemod -K audit_flags=cusa:no sysadm

# rolemod -K audit_flags=cusa:no auditadm
```

```
# rolemod -K audit_flags=cusa:no netadm
```

- **すべてのユーザーに対してこれらのクラスを監査するには、システム全体のフラグを設定します。**

  ```
  # auditconfig -setflags lo,ex
  ```

  出力は次のようになります。

  ```
  header,129,2,AUE_EXECVE,,mach1,2010-10-14 12:17:12.616 -07:00
  path,/usr/bin/ls
  attribute,100555,root,bin,21,320271,18446744073709551615
  subject,jdoe,root,root,root,root,2486,50036632,82 0 mach1
  return,success,0
  ```

3. **記録されるコマンドの使用に関する追加情報を指定します。**

   - **コマンドへの引数を記録するには、argv ポリシーを追加します。**

     ```
     # auditconfig -setpolicy +argv
     ```

     exec_args トークンは、コマンド引数を記録します。次の例では、表示のために行が折り返されています。

     ```
     header,151,2,AUE_EXECVE,,mach1,2010-10-14 12:26:17.373 -07:00
     path,/usr/bin/ls
     attribute,100555,root,bin,21,320271,18446744073709551615
     exec_args
     ,2,ls,/etc/security
     subject,jdoe,root,root,root,root,2494,50036632,82 0 mach1
     return,success,0
     ```

   - **コマンドが実行される環境を記録するには、arge ポリシーを追加します。**

     ```
     # auditconfig -setpolicy +arge
     ```

     exec_env トークンは、コマンド環境を記録します。次の例では、表示のために行が折り返されています。

     ```
     header,1460,2,AUE_EXECVE,,mach1,2010-10-14 12:29:39.679 -07:00
     path,/usr/bin/ls
     attribute,100555,root,bin,21,320271,18446744073709551615
     exec_args,2,ls,/etc/security
     exec_env
     ,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
     LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
     HOME=/home/jdoe,LOGNAME=jdoe,G_FILENAME_ENCODING=@locale,UTF-8,
     PRINTER=example-dbl,...,_=/usr/bin/ls
     subject,jdoe,root,root,root,root,2502,50036632,82 0 mach1
     return,success,0
     ```

▼ 特定のファイルに対する変更の監査レコードを検索する方法

目標が、/etc/passwd や /etc/default ディレクトリ内のファイルなどの、限られた数のファイルに対するファイル書き込みをログに記録することである場合は、auditreduce コマンドを使用してこれらのファイルを見つけることができます。

始める前に root 役割は、この手順内のすべてのタスクを実行できます。

管理権利が組織内に分散している場合は、次の点に注意してください。

- Audit Configuration 権利プロファイルを持つ管理者は、auditconfig コマンドを実行できます。
- Audit Review 権利プロファイルを持つ管理者は、auditreduce コマンドを実行できます。
- 監査フラグを割り当てることができるのは、root 役割だけです。

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. **ファイルの変更を監査するには、次の手順のいずれかを実行します。**

   - fw クラスを監査します。

     fw クラスをユーザーまたは役割の監査フラグに追加すると、このクラスをシステム全体の監査事前選択マスクに追加する場合に比べて、生成されるレコードが少なくなります。次の手順のいずれかを実行します。

     - fw クラスを特定の役割に追加します。

       ```
       # rolemod -K audit_flags=fw:no root
       # rolemod -K audit_flags=fw:no sysadm
       # rolemod -K audit_flags=fw:no auditadm
       # rolemod -K audit_flags=fw:no netadm
       ```

     - fw クラスをシステム全体のフラグに追加します。

       ```
       # auditconfig -getflags
       active user default audit flags = lo(0x1000,0x1000)
       configured user default audit flags = lo(0x1000,0x1000)

       # auditconfig -setflags lo,fw
       user default audit flags = lo,fw(0x1002,0x1002)
       ```

   - 成功したファイル書き込みを監査します。

     成功を監査すると、失敗と成功を監査する場合に比べて、生成されるレコードが少なくなります。次の手順のいずれかを実行します。

     - +fw フラグを特定の役割に追加します。

```
# rolemod -K audit_flags=+fw:no root
# rolemod -K audit_flags=+fw:no sysadm
# rolemod -K audit_flags=+fw:no auditadm
# rolemod -K audit_flags=+fw:no netadm
```

- `+fw` フラグをシステム全体のフラグに追加します。

```
# auditconfig -getflags
active user default audit flags = lo(0x1000,0x1000)
configured user default audit flags = lo(0x1000,0x1000)

# auditconfig -setflags lo,+fw
user default audit flags = lo,+fw(0x1002,0x1000)
```

2. **`auditreduce` コマンドを使用して、特定のファイルに関する監査レコードを取得します。**

```
# auditreduce -o file=/etc/passwd,/etc/default -O filechg
```

`auditreduce` コマンドは、`file` 引数のすべての出現を監査証跡で検索します。このコマンドにより、接尾辞 `filechg` を持つバイナリファイルが作成されます。このファイルには、必要なファイルのパスを含むすべてのレコードが含まれています。`-o file=` *pathname* オプションの構文については、`auditreduce(1M)` のマニュアルページを参照してください。

3. **`praudit` コマンドを使用して、`filechg` ファイルを読み取ります。**

```
# praudit *filechg
```

▼ ログインしているユーザーの事前選択マスクを更新する方法

この手順では、すでにログインしているユーザーを、システム全体の監査事前選択マスクの変更について監査する方法について説明します。通常、ログアウトしてから再度ログインするようにユーザーに指示することで、このタスクを完了できます。また、Process Management 権利プロファイルが割り当てられている役割で、`kill` コマンドを使用すると、アクティブなセッションを手動で終了できます。新しいセッションでは、新しい事前選択マスクが継承されます。

ただし、ユーザーセッションの終了が実用的でない場合もあります。代わりに、`auditconfig` コマンドを使用すれば、ログインしている各ユーザーの事前選択マスクを動的に変更できます。

次の手順では、次のコマンドを使用して、システム全体の監査事前選択マスクを `lo` から `lo,ex` に変更したことが前提となっています。

```
# auditconfig -setflags lo,ex
```

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。ユーザーセッションを終了するには、Process 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 11.3 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. **ログインしている通常のユーザーと各ユーザーのプロセス ID を一覧表示します。**

```
# who -a
jdoe  - vt/2         Jan 25 07:56  4:10   1597   (:0)
jdoe  + pts/1        Jan 25 10:10   .     1706   (:0.0)
...
jdoe  + pts/2        Jan 25 11:36  3:41   1706   (:0.0)
```

2. **あとで比較するために、各ユーザーの事前選択マスクを表示します。**

```
# auditconfig -getpinfo 1706
audit id = jdoe(1234)
process preselection mask = lo(0x1000,0x1000)
terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
audit session id = 103203403
```

3. **次の 1 つ以上のコマンドを実行することによって、適切な事前選択マスクを変更します。**

```
# auditconfig -setpmask 1706 lo,ex              /* このプロセス用 */

# auditconfig -setumask jdoe lo,ex              /* このユーザー用 */

# auditconfig -setsmask 103203403 lo,ex         /* このセッション用 */
```

4. **ユーザーの事前選択マスクが変更されたことを確認します。**
   たとえば、マスクを変更する前に存在していたプロセスを確認します。

```
# auditconfig -getpinfo 1706
audit id = jdoe(1234)
process preselection mask = ex,lo(0x40001000,0x40001000)
terminal id (maj,min,host) = 9426,65559,mach1(192.168.123.234)
audit session id = 103203403
```

▼ 特定のイベントの監査を回避する方法

メンテナンスのために、サイトでイベントが監査されないようにする場合があります。

注記 - Solaris カーネルイベントではこの手順を使用できません。ただし、この手順を使用して、2047 より大きいイベント番号を持つユーザーレベルの監査イベントを割り当てることができます。

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. **イベントのクラスを no クラスに変更します。**

   注記 - 監査構成ファイルの変更の影響については、121 ページの「監査構成ファイルとパッケージ化」を参照してください。

   たとえば、イベント 6214 と 6215 は、ユーザー管理の ua クラスに属します。

   ```
   ## audit_event file
   ...
   6214:AUE_kadmind_auth:authenticated kadmind request:ua
   6215:AUE_kadmind_unauth:unauthenticated kadmind req:ua

   ...
   ```

   これらのイベントを no クラスに変更します。

   ```
   ## audit_event file
   ...
   6214:AUE_kadmind_auth:authenticated kadmind request:no
   6215:AUE_kadmind_unauth:unauthenticated kadmind req:no

   ...
   ```

   ua クラスが現在監査中である場合でも、既存のセッションはイベント 6214 および 6215 を監査します。これらのイベントの監査を停止するには、67 ページの「ログインしているユーザーの事前選択マスクを更新する方法」の手順に従って、ユーザーの事前選択マスクを更新する必要があります。

   注意 - audit_event ファイルではイベントをコメントにしないでください。このファイルは、praudit コマンドがバイナリ監査ファイルを読み取るときに使用します。また、このファイルに一覧表示されたイベントが、保管された監査ファイルに含まれることがあります。

2. **カーネルイベントをリフレッシュします。**

   ```
   # auditconfig -conf
   Configured 283 kernel events.
   ```

▼ 専用ファイルシステム上の監査ファイルを圧縮する方法

監査ファイルは拡大する場合があります。例21「audit_binfile プラグインのためのファイルサイズを制限する」に示すように、ファイルのサイズに上限を設定できます。この手順では、圧縮を使用してサイズを削減します。

始める前に ZFS File System Management および 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。後者のプロファイルを使用すると、ストレージプールを作成できます。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. **ZFS ファイルシステムを監査ファイル専用にします。**

   手順については、80 ページの「監査ファイルのための ZFS ファイルシステムを作成する方法」を参照してください。

2. **次のオプションのいずれかを使用して ZFS ストレージプールを圧縮します。**

   どちらのオプションを使用した場合でも、監査ファイルシステムが圧縮されます。監査サービスがリフレッシュされると、圧縮率が表示されます。

   次の例では、ZFS プール `auditp/auditf` がデータセットです。

   - **デフォルトの圧縮アルゴリズムを使用します。**

     ```
     # zfs set compression=on auditp/auditf
     # audit -s
     # zfs get compressratio auditp/auditf
     NAME           PROPERTY       VALUE  SOURCE
     auditp/auditf  compressratio  4.54x  -
     ```

   - **より高い圧縮アルゴリズムを使用します。**

     ```
     # zfs set compression=gzip-9 auditp/auditf
     # zfs get compression auditp/auditf
     NAME           PROPERTY     VALUE     SOURCE
     auditp/auditf  compression  gzip-9    local
     ```

     `gzip-9` 圧縮アルゴリズムでは、デフォルトの圧縮アルゴリズムである `lzjb` に比べて、占有する容量が 1/3 少ないファイルが生成されます。詳細は、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』 の 第 7 章, 「Oracle Solaris ZFS ファイルシステムの管理」を参照してください。

3. **監査サービスをリフレッシュします。**

   ```
   # audit -s
   ```

4. **(オプション) 新しい圧縮の設定を確認します。**

たとえば、より圧縮率の高いアルゴリズムを使用した場合、情報は次のように表示されます。

```
# zfs get compressration auditp/auditf
NAME           PROPERTY       VALUE  SOURCE
auditp/auditf  compressratio  16.89x  -
```

▼ FTP および SFTP ファイル転送を監査する方法

FTP サービスは、ファイル転送のログを作成します。ssh プロトコルの下で実行される SFTP サービスは、ft 監査クラスを事前選択することによって監査できます。これらの両方のサービスへのログインを監査できます。

注記 - FTP サービスのコマンドとファイル転送をログに記録する方法については、proftpd(8) のマニュアルページを参照してください。

使用可能なログ作成オプションについては、ProFTPD Logging (http://www.proftpd.org/docs/howto/Logging.html) を参照してください。

- **SFTP と FTP のどちらを監査するのかに応じて、次の手順のいずれかを実行します。**
  - sftp のアクセスとファイル転送をログに記録するには、ft クラスを編集します。

    ft クラスには、次の SFTP トランザクションが含まれています。

    ```
    % auditrecord -c ft
    file transfer: chmod ...
    file transfer: chown ...
    file transfer: get ...
    file transfer: mkdir ...
    file transfer: put ...
    file transfer: remove ...
    file transfer: rename ...
    file transfer: rmdir ...
    file transfer: session start ...
    file transfer: session end ...
    file transfer: symlink ...
    file transfer: utimes
    ```
  - FTP サーバーへのアクセスを記録するには、lo クラスを監査します。

    次のサンプル出力に示すとおり、proftpd デーモンへのログインおよびログアウトで監査レコードが生成されます。

    ```
    % auditrecord -c lo | more
    ...
    ```

```
FTP server login
program     proftpd             See in.ftpd(1M)
event ID    6165                AUE_ftpd
class       lo                  (0x0000000000001000)
header
subject
[text]                      error message
return

FTP server logout
program     proftpd             See in.ftpd(1M)
event ID    6171                AUE_ftpd_logout
class       lo                  (0x0000000000001000)
header
subject
return
...
```

ゾーンでの監査サービスの構成

監査サービスは、ゾーン内での監査イベントも含め、システム全体を監査します。非大域ゾーンがインストールされているシステムでは、すべてのゾーンを同様に監査したり、ゾーンごとに監査を構成したりできます。詳細は、32 ページの「ゾーン内での監査の計画」を参照してください。

非大域ゾーンを、大域ゾーンを監査する場合とまったく同様に監査する場合は、非大域ゾーン管理者が監査レコードにアクセスできない可能性があります。また、大域ゾーン管理者は、非大域ゾーン内のユーザーの監査事前選択マスクを変更することもできます。

非大域ゾーンを個別に監査する場合は、監査レコードが非大域ゾーンのルートから非大域ゾーンと大域ゾーンに表示されます。

▼ すべてのゾーンの監査を同様に構成する方法

この手順に従えば、すべてのゾーンを同様に監査できます。この方法を使えば、必要とされるコンピュータオーバーヘッドと管理リソースが最小になります。

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. **大域ゾーンの監査を構成します。**

   48 ページの「監査サービスの構成」のタスクを行います。ただし、次の点は例外です。

   - `perzone` 監査ポリシーを有効にしないでください。
   - `zonename` ポリシーを設定します。このポリシーによって、ゾーンの名前がすべての監査レコードに追加されます。

     ```
     # auditconfig -setpolicy +zonename
     ```

2. **監査構成ファイルを変更した場合は、それらのファイルを大域ゾーンからすべての非大域ゾーンにコピーします。**

   `audit_class` または `audit_event` ファイルを変更した場合は、次の 2 つの方法のどちらかでコピーします。

   - これらのファイルをループバックマウントできます。
   - これらのファイルをコピーできます。

   非大域ゾーンが動作している必要があります。

   - **変更された `audit_class` および `audit_event` ファイルをループバックファイルシステム (`lofs`) としてマウントします。**

     a. **大域ゾーンから、非大域ゾーンを停止します。**

     ```
     # zoneadm -z non-global-zone halt
     ```

     b. **大域ゾーンで変更した監査構成ファイルごとに読み取り専用のループバックマウントを 1 つずつ作成します。**

     ```
     # zonecfg -z non-global-zone
     zone: add fs
     zone/fs: set special=/etc/security/audit-file
     zone/fs: set dir=/etc/security/audit-file
     zone/fs: set type=lofs
     zone/fs: add options [ro,nodevices,nosetuid]
     zone/fs: commit
     zone/fs: end
     zone: exit
     #
     ```

     c. **変更を有効にするには、非大域ゾーンをブートします。**

     ```
     # zoneadm -z non-global-zone boot
     ```

     あとで、大域ゾーンで監査構成ファイルを変更した場合は、ループバックマウントされたファイルを非大域ゾーンでリフレッシュするために各ゾーンをリブートします。

   - **ファイルをコピーします。**

a. 大域ゾーンから、各非大域ゾーン内の `/etc/security` ディレクトリを一覧表示します。

```
# ls /zone/zonename/root/etc/security/
```

b. 変更された `audit_class` および `audit_event` ファイルを各ゾーンの `/etc/security` ディレクトリにコピーします。

```
# cp /etc/security/audit-file /zone/zonename/root/etc/security/audit-file
```

あとで、大域ゾーンでこれらのファイルのいずれかを変更した場合は、変更されたファイルを非大域ゾーンにコピーする必要があります。

例 17 ゾーンで監査構成ファイルをループバックマウントとしてマウントする

この例では、システム管理者が `audit_class`、`audit_event`、および `audit_warn` ファイルを変更しました。

`audit_warn` ファイルは大域ゾーンでのみ読み取られるため、非大域ゾーンにマウントする必要はありません。

このシステム `machine1` 上で、管理者は 2 つの非大域ゾーン `machine1-webserver` と `machine1-appserver` を作成しました。管理者は、監査構成ファイルの変更を完了しました。管理者があとでこれらのファイルを変更した場合は、ループバックマウントを再読み込みするために、ゾーンをリブートする必要があります。

```
# zoneadm -z machine1-webserver halt
# zoneadm -z machine1-appserver halt
# zonecfg -z machine1-webserver
webserver: add fs
webserver/fs: set special=/etc/security/audit_class
webserver/fs: set dir=/etc/security/audit_class
webserver/fs: set type=lofs
webserver/fs: add options [ro,nodevices,nosetuid]
webserver/fs: commit
webserver/fs: end
webserver: add fs
webserver/fs: set special=/etc/security/audit_event
webserver/fs: set dir=/etc/security/audit_event
webserver/fs: set type=lofs
webserver/fs: add options [ro,nodevices,nosetuid]
webserver/fs: commit
webserver/fs: end
webserver: exit
#

# zonecfg -z machine1-appserver
appserver: add fs
appserver/fs: set special=/etc/security/audit_class
appserver/fs: set dir=/etc/security/audit_class
appserver/fs: set type=lofs
appserver/fs: add options [ro,nodevices,nosetuid]
appserver/fs: commit
appserver/fs: end
appserver: exit
```

非大域ゾーンがリブートされると、audit_class および audit_event ファイルは、これらのゾーンで読み取り専用になります。

▼ ゾーンごとの監査を構成する方法

この手順に従えば、個々のゾーン管理者が自身のゾーン内で監査サービスを制御できます。ポリシーオプションの完全な一覧については、auditconfig(1M) のマニュアルページを参照してください。

始める前に 監査を構成するには、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。監査サービスを有効にするには、Audit Control 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 11.3 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. **大域ゾーンで、監査を構成します。**

   a. **48 ページの「監査サービスの構成」のタスクを行います。**

   b. **perzone 監査ポリシーを追加してください。このコマンドについては、例 12「perzone 監査ポリシーを設定する」を参照してください。**

   注記 - 大域ゾーンで監査サービスを有効にする必要はありません。

2. **監査を予定している各非大域ゾーンで、監査ファイルを構成します。**

   a. **48 ページの「監査サービスの構成」のタスクを行います。**

   b. **システム全体の監査設定は構成しないでください。**
   特に、非大域ゾーンに perzone または ahlt ポリシーを追加しないでください。

3. **使用するゾーンで監査を有効にします。**

```
myzone# audit -s
```

例 18 非大域ゾーンで監査を無効にする

この例は、perzone 監査ポリシーが設定されている場合に機能します。noaudit ゾーンのゾーン管理者が、そのゾーンの監査を無効にします。

```
noauditzone # auditconfig -getcond
audit condition = auditing
```

```
noauditzone # audit -t
noauditzone # auditconfig -getcond
audit condition = noaudit
```

例: Oracle Solaris 監査の構成

このセクションでは、Oracle Solaris 監査を構成および実装する方法の一例を示します。特定のニーズおよび要件に従って、サービスのさまざまな属性を構成することから始めます。構成が完了すると、構成の設定を有効にするために監査サービスが起動されます。新しい要件を満たすたすように既存の監査構成を修正する必要があるたびに、この例と同じ順序に従ってアクションを実行してください。

1. 監査パラメータを構成します。
2. 監査サービスをリフレッシュします。
3. 新しい監査の構成を確認します。

- 最初に、管理者は一時的なポリシーを追加します。

```
# auditconfig -t -setpolicy +zonename
# auditconfig -getpolicy
configured audit policies = ahlt,arge,argv,perzone
active audit policies = ahlt,arge,argv,perzone,zonename
```

- 次に、管理者はキュー制御を指定します。

```
# auditconfig -setqctrl 200 20 0 0
# auditconfig -getqctrl
configured audit queue hiwater mark (records) = 200
configured audit queue lowater mark (records) = 20
configured audit queue buffer size (bytes) = 8192
configured audit queue delay (ticks) = 20
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 20
active audit queue buffer size (bytes) = 8192
active audit queue delay (ticks) = 20
```

- 次に、管理者はプラグインの属性を指定します。
  - `audit_binfile` プラグインの場合、管理者は `qsize` 値を削除します。

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/audit/sys1.1,/var/audit;
p_minfree=2;p_fsize=4G;
Queue size: 200
# auditconfig -setplugin audit_binfile  "" 0
# auditconfig -getplugin audit_binfile
```

```
Plugin: audit_binfile
Attributes: p_dir=/audit/sys1.1,/var/audit
p_minfree=2;p_fsize=4G;
```

- audit_syslog プラグインの場合、管理者は、成功したログインおよびログアウトイベントと失敗した実行可能ファイルが syslog に送信されるように指定します。このプラグインの qsize は 150 に設定されます。

```
# auditconfig -setplugin audit_syslog active p_flags=+lo,-ex 150
# auditconfig -getplugin audit_syslog
auditconfig -getplugin audit_syslog
Plugin: audit_syslog
Attributes: p_flags=+lo,-ex;
Queue size: 150
```

- 管理者は、audit_remote プラグインを構成したり、使用したりしません。

- 次に、管理者は監査サービスをリフレッシュし、構成を確認します。
  - 一時的な zonename ポリシーはもう設定されていません。

```
# audit -s
# auditconfig -getpolicy
configured audit policies = ahlt,arge,argv,perzone
active audit policies = ahlt,arge,argv,perzone
```

  - キュー制御は同じままです。

```
# auditconfig -getqctrl
configured audit queue hiwater mark (records) = 200
configured audit queue lowater mark (records) = 20
configured audit queue buffer size (bytes) = 8192
configured audit queue delay (ticks) = 20
active audit queue hiwater mark (records) = 200
active audit queue lowater mark (records) = 20
active audit queue buffer size (bytes) = 8192
active audit queue delay (ticks) = 20
```

  - audit_binfile プラグインには、指定されたキューサイズはありません。audit_syslog プラグインには、指定されたキューサイズがあります。

```
# auditconfig -getplugin
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;

Plugin: audit_syslog
Attributes: p_flags=+lo,-ex;
Queue size: 150
...
```

♦ ♦ ♦ 第 4 章

4 システム動作のモニタリング

この章では、システムでのアクティビティーのモニタリングを可能にする監査ログを構成する際に役立つ手順について説明します。さらに、次の章では、その他の監査管理タスクについて説明します。

- 第3章「監査サービスの管理」
- 第5章「監査データの操作」
- 第6章「監査サービスに関する問題の分析および解決」

監査サービスの概要については、第1章「Oracle Solaris での監査について」を参照してください。計画の提案については、第2章「監査の計画」を参照してください。参照情報については、第7章「監査の参照情報」を参照してください。

監査ログの構成

2 つの監査プラグイン `audit_binfile` と `audit_syslog`、は、ローカル監査ログを作成できます。次のタスクでは、このようなログを構成する方法について説明します。

監査ログの構成

次のタスクマップは、さまざまなプラグインの監査ログを構成するための手順を示しています。`audit_binfile` プラグインに対するログの構成はオプションです。その他のプラグインに対するログは、管理者が構成する必要があります。

表 6 監査ログの構成タスクマップ

タスク	説明	手順
`audit_binfile` プラグインのためのローカルストレージを追加する	監査ファイルのための追加のディスク容量を作成し、それをファイルアクセス権で保護する	80 ページの「監査ファイルのための ZFS ファイルシステムを作成する方法」

タスク	説明	手順
audit_binfile プラグインにストレージを割り当てる	バイナリ監査レコードのためのディレクトリを識別する	83 ページの「監査トレールのための監査領域を割り当てる方法」
監査レコードのリモートシステムへのストリーム出力を構成する	保護されたメカニズムを通して監査レコードをリモートリポジトリに送信できるようにする	86 ページの「監査ファイルをリモートリポジトリに送信する方法」
監査ファイルのためのリモートストレージを構成する	リモートシステム上で監査レコードを受信できるようにする	88 ページの「監査ファイルのためのリモートリポジトリを構成する方法」
audit_syslog プラグインのためのストレージを構成します。	テキスト形式の監査イベントを syslog にストリーム出力できるようにします。	92 ページの「syslog 監査ログの構成方法」

▼ 監査ファイルのための ZFS ファイルシステムを作成する方法

次の手順は、監査ファイルのための ZFS プールや、対応するファイルシステムとマウントポイントを作成する方法を示しています。デフォルトでは、`audit_binfile` プラグインの監査ファイルは `/var/audit` に保持されます。

始める前に ZFS File System Management および 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。後者のプロファイルを使用すると、ストレージプールを作成できます。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. **必要なディスク容量を決定します。**

   必要な監査の量によりディスク容量要件が決まります。

   注記 - デフォルトのクラスの事前選択によって、ログイン、ログアウト、役割の引き受けなどの、`lo` クラス内のイベントの記録されたインスタンスごとに約 80 バイトずつ拡大するファイルが `/var/audit` 内に作成されます。

2. **ミラー化された ZFS ストレージプールを作成します。**

   `zpool create` コマンドは、ZFS ファイルシステムのコンテナであるストレージプールを作成します。詳細は、『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』 の第 1 章, 「Oracle Solaris ZFS ファイルシステムの概要」を参照してください。

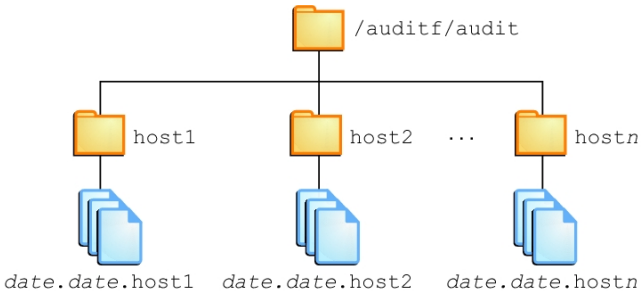
   ```
   # zpool create audit-pool mirror disk1 disk2
   ```

   たとえば、`c3t1d0` と `c3t2d0` の 2 つのディスクから `auditp` プールを作成し、それらをミラー化します。

```
# zpool create auditp mirror c3t1d0 c3t2d0
```

3. **監査ファイルのための ZFS ファイルシステムとマウントポイントを作成します。**

   ファイルシステムとマウントポイントは 1 つのコマンドで作成します。作成時に、ファイルシステムがマウントされます。たとえば、次の図は、ホスト名で格納された監査トレールのストレージを示しています。




   ```
   # zfs create -o mountpoint=/mountpoint  audit-pool/mountpoint
   ```

   たとえば、`auditf` ファイルシステムの `/audit` マウントポイントを作成します。

   ```
   # zfs create -o mountpoint=/audit auditp/auditf
   ```

4. **監査ファイルのための ZFS ファイルシステムを作成します。**

   ```
   # zfs create -p auditp/auditf/system
   ```

   たとえば、`sys1` システムのための暗号化されていない ZFS ファイルシステムを作成します。

   ```
   # zfs create -p auditp/auditf/sys1
   ```

5. **(オプション) 監査ファイルのための追加のファイルシステムを作成します。**

   追加のファイルシステムを作成する 1 つの理由は、監査のオーバーフローを回避するためです。ファイルシステムごとの ZFS 割り当て制限は、ステップ 7 に示すように設定できます。各割り当て制限に達すると、`audit_warn` 電子メールエイリアスによって通知されます。領域を解放するために、閉じられた監査ファイルをリモートサーバーに移動できます。

   ```
   # zfs create -p auditp/auditf/sys1.1

   # zfs create -p auditp/auditf/sys1.2
   ```

6. **プール内の監査ファイルを圧縮します。**

ZFS では通常、圧縮はファイルシステムのレベルで設定されます。ただし、このプール内のすべてのファイルシステムに監査ファイルが含まれているため、圧縮はプールのトップレベルのデータセットで設定されます。

```
# zfs set compression=on auditp
```

『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』 の 「ZFS の圧縮、複製解除、暗号化のプロパティー間の関連」も参照してください。

7. **割り当て制限を設定します。**

   親のファイルシステム、子孫のファイルシステム、またはその両方で割り当て制限を設定できます。親の監査ファイルシステム上で割り当て制限を設定した場合は、子孫のファイルシステム上の割り当て制限によって追加の制限が課せられます。

   a. **親の監査ファイルシステム上で割り当て制限を設定します。**

      次の例では、`auditp` プール内の両方のディスクが割り当て制限に達すると、`audit_warn` スクリプトによって監査管理者に通知されます。

      ```
      # zfs set quota=510G auditp/auditf
      ```

   b. **子孫の監査ファイルシステム上で割り当て制限を設定します。**

      次の例では、`auditp/auditf/`*system* ファイルシステムの割り当て制限に達すると、`audit_warn` スクリプトによって監査管理者に通知されます。

      ```
      # zfs set quota=170G auditp/auditf/sys1

      # zfs set quota=170G auditp/auditf/sys1.1

      # zfs set quota=165G auditp/auditf/sys1.2
      ```

8. **大規模なプールの場合は、監査ファイルのサイズを制限します。**

   デフォルトでは、監査ファイルはプールのサイズまで拡大できます。管理容易性のために、監査ファイルのサイズを制限します。例21「`audit_binfile` プラグインのためのファイルサイズを制限する」を参照してください。

例 19　監査アーカイブのための暗号化されたファイルシステムを作成する

サイトのセキュリティー要件に従うには、管理者は次の手順を実行します。

1. 必要に応じて、暗号化された監査ログを格納するための新しい ZFS プールを作成します。
2. 暗号化鍵を生成します。
3. 監査ログを格納するために暗号化を有効にして監査ファイルシステムを作成し、マウントポイントを設定します。
4. 暗号化されたディレクトリを使用するように監査を構成します。
5. 新しい構成の設定を適用するために、監査サービスをリフレッシュします。

```
# zpool create auditp mirror disk1 disk2


# pktool genkey keystore=file outkey=/filename keytype=aes keylen=256

# zfs create -o encryption=aes-256-ccm \
-o keysource=raw,file:///filename \
-o compression=on -o mountpoint=/audit auditp/auditf

# auditconfig -setplugin audit_binfile p_dir=/audit/

# audit -s
```

鍵が格納されているファイル (この例では *filename* など) をバックアップして、保護する必要があります。

管理者が `auditf` ファイルシステムの下に追加のファイルシステムを作成した場合は、これらの子孫のファイルシステムも暗号化されます。

例 20 `/var/audit` に割り当て制限を設定する

この例では、管理者は、デフォルトの監査ファイルシステム上で割り当て制限を設定します。この割り当て制限に達すると、`audit_warn` スクリプトによって監査管理者に警告が通知されます。

```
# zfs set quota=252G rpool/var/audit
```

▼ 監査トレールのための監査領域を割り当てる方法

この手順では、`audit_binfile` プラグインの属性を使用して、監査トレールに追加のディスク容量を割り当てます。

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 11.3 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. **`audit_binfile` プラグインの属性を決定します。**

   `audit_binfile(5)` のマニュアルページの `OBJECT ATTRIBUTES` のセクションを参照してください。

```
# man audit_binfile

...
OBJECT ATTRIBUTES
The p_dir attribute specifies where the audit files will be created.
The directories are listed in the order in which they are to be used.

The p_minfree attribute defines the percentage of free space that the
```

```
audit system requires before the audit daemon invokes the audit_warn
script.

The p_fsize attribute defines the maximum size that an audit
file can become before it is automatically closed and a new
audit file is opened. ... The format of the p_fsize value can
be specified as an exact value in bytes or in a human-readable
form with a suffix of  B,  K, M, G, T, P, E, Z (for bytes,
kilobytes, megabytes, gigabytes, terabytes,  petabytes, exabytes,
or zettabytes, respectively). Suffixes of KB, MB, GB, TB, PB, EB,
and ZB are also accepted.
```

2. **監査トレールにディレクトリを追加するには、`p_dir` 属性を指定します。**

   `/var/audit` はデフォルトのファイルシステムとして機能します。

   ```
   # auditconfig -setplugin audit_binfile p_dir=/audit/sys1.1,/var/audit
   ```

   前のコマンドは、`/audit/sys1.1` ファイルシステムを監査ファイルのプライマリディレクトリとして、またデフォルトの `/var/audit` をセカンダリディレクトリとして設定します。このシナリオでは、`/var/audit` は最後の手段のディレクトリとして機能します。この構成を成功させるには、`/audit/sys1.1` ファイルシステムが存在する必要があります。

   80 ページの「監査ファイルのための ZFS ファイルシステムを作成する方法」でも同様のファイルシステムが作成されています。

3. **監査サービスをリフレッシュします。**

   `auditconfig -setplugin` コマンドは、構成された値を設定します。この値は監査サービスのプロパティーであるため、このサービスがリフレッシュまたは再開されたときに復元されます。構成された値は、監査サービスがリフレッシュまたは再開されたときにアクティブになります。構成された値とアクティブな値については、auditconfig(1M) のマニュアルページを参照してください。

   ```
   # audit -s
   ```

例 21 `audit_binfile` プラグインのためのファイルサイズを制限する

次の例では、バイナリ監査ファイルのサイズが特定のサイズに設定されます。このサイズは M バイト単位で指定されます。

```
# auditconfig -setplugin audit_binfile p_fsize=4M

# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_age=0h;p_dir=/var/audit;p_minfree=1;p_fsize=4M;
```

デフォルトでは、監査ファイルは無制限に拡大できます。作成する監査ファイルを小さくするために、管理者は 4M バイトのファイルサイズ制限を指定します。このサイズ制限に達すると、監査サービスは新しいファイルを作成します。このファイルサイズ制限は、管理者が監査サービスをリフレッシュしたあとに有効になります。

```
# audit -s
```

例 22 ログローテーションの時間の指定

次の例では、監査ファイルに時間制限が設定されています。時間制限は時間、日付、週、または月の単位で指定されます。

```
# auditconfig -setplugin audit_binfile p_age=1w

# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_minfree=1;p_fsize=4M;p_age=1w;
Queue size: 200
```

デフォルトでは、監査ファイルに時間制限が設定されていません。外部操作でファイルのローテーションが発生するまで、ファイルは無期限に開いたままです。管理者がファイルの時間制限を 1 週間に設定しても、新しい監査ファイルは開いたままです。新しい時間制限を実装するには、管理者は監査サービスをリフレッシュします。

```
# audit -s
```

例 23 監査プラグインに対するいくつかの変更を指定する

次の例では、高いスループットと大規模な ZFS プールを備えたシステム上の管理者が、`audit_binfile` プラグインのキューサイズ、バイナリファイルのサイズ、および弱い制限値の警告を変更します。管理者は、監査ファイルを 4G バイトまで拡大できるようにし、ZFS プールの残りが 2% になったら警告を受信するようにし、許可されるキューサイズを 2 倍にします。デフォルトのキューサイズは、`active audit queue hiwater mark (records) = 100` にあるように、カーネル監査キューの高位境界値 `100` になります。また、監査ファイルの時間制限が 2 週間に設定されています。

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=2G;p_minfree=1;

# auditconfig -setplugin audit_binfile \
      "p_minfree=2;p_fsize=4G;p_age=2w" 200

# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;p_age=2w;
Queue size: 200
```

変更した指定は、管理者が監査サービスをリフレッシュしたあとに有効になります。

```
# audit -s
```

例 24 監査プラグインのキューサイズを削除する

次の例では、`audit_binfile` プラグインのキューサイ が削除されます。

```
# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
```

```
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
Queue size: 200

# auditconfig -setplugin audit_binfile "" 0

# auditconfig -getplugin audit_binfile
Plugin: audit_binfile
Attributes: p_dir=/var/audit;p_fsize=4G;p_minfree=2;
```

空の引用符 ("") によって、現在の属性値が保持されます。最後の `0` は、プラグインのキューサイズをデフォルトに設定します。

プラグインに対する `qsize` の指定の変更は、管理者が監査サービスをリフレッシュしたあとに有効になります。

```
# audit -s
```

例 25 警告のための弱い制限値を設定する

この例では、ファイルシステムの 2% がまだ使用可能なときに警告が発行されるように、すべての監査ファイルシステムの最小空き容量のレベルが設定されます。

```
# auditconfig -setplugin audit_binfile p_minfree=2
```

デフォルトの割合 (%) は `1` です。大規模な ZFS プールの場合は、適度に低い割合を選択します。たとえば、16T バイトプールの 10% は約 16G バイトであり、これにより、大量のディスク容量が残っているときに監査管理者に警告が通知されます。`2` の値を指定すると、約 2G バイトのディスク容量が残っているときに `audit_warn` メッセージが送信されます。

`audit_warn` 電子メールエイリアスが警告を受信します。エイリアスを設定するには、59 ページの「`audit_warn` 電子メールエイリアスの構成方法」を参照してください。

大規模なプールの場合、管理者はファイルサイズも 3G バイトに制限します。

```
# auditconfig -setplugin audit_binfile p_fsize=3G
```

プラグインに対する `p_minfree` と `p_fsize` の指定は、管理者が監査サービスをリフレッシュしたあとに有効になります。

```
# audit -s
```

▼ 監査ファイルをリモートリポジトリに送信する方法

この手順では、`audit_remote` プラグインの属性を使用して、監査トレールをリモート監査リポジトリに送信します。Oracle Solaris システム上にリモートリポジトリを構

成するには、88 ページの「監査ファイルのためのリモートリポジトリを構成する方法」を参照してください。

始める前に リモートリポジトリに監査ファイルの受信者が存在する必要があります。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 11.3 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. **`audit_remote` プラグインの属性を決定します。**
   `audit_remote(5)` のマニュアルページの `OBJECT ATTRIBUTES` のセクションを参照してください。

```
# man audit_remote

...
OBJECT ATTRIBUTES
The p_hosts attribute specifies the remote servers.
You can also specify the port number and the GSS-API
mechanism.

The p_retries attribute specifies the number of retries for
connecting and sending data. The default is 3.

The p_timeout attribute specifies the number of seconds
in which a connection times out.
```

   デフォルトポートは、`solaris_audit` IANA 割り当て済みポート `16162/tcp` です。デフォルトのメカニズムは `kerberos_v5` です。タイムアウトのデフォルトは 5 秒です。また、プラグインのキューサイズも指定できます。

2. **リモートの受信側のシステムを指定するには、`p_hosts` 属性を使用します。**
   この例では、受信側のシステムは別のポートを使用します。

```
# auditconfig -setplugin audit_remote \
    p_hosts=ars.example.com:16088:kerberos_v5
```

3. **変更するプラグインのその他の属性を指定します。**
   たとえば、次のコマンドは、すべてのオプション属性の値を指定します。

```
# auditconfig -setplugin audit_remote "p_retries=;p_timeout=3" 300
```

4. **これらの値を確認してから、プラグインをアクティブにします。**
   たとえば、次のコマンドは、プラグインの値を指定して確認します。

```
# auditconfig -getplugin audit_remote
Plugin: audit_remote (inactive)
Attributes: p_hosts=ars.example.com:16088:kerberos_v5;p_retries=5;p_timeout=3;
Queue size: 300

# auditconfig -setplugin audit_remote active
```

5. **監査サービスをリフレッシュします。**
   監査サービスは、リフレッシュ時に監査プラグインの変更を読み取ります。

```
# audit -s
```

例 26 監査キューのバッファーサイズのチューニング

この例では、監査キューが `audit_remote` プラグインの背後でいっぱいになっています。この監査対象システムは多数のクラスを監査するように構成されており、トラフィック量の多い、低速なネットワークを経由して転送しています。管理者は、プラグインのバッファーサイズを増やすことによって、レコードがキューから削除される前に監査キューを拡張可能にし、バッファーの制限を超えることがないようにます。

```
audsys1 # auditconfig -setplugin audit_remote "" 1000

audsys1 # audit -s
```

▼ 監査ファイルのためのリモートリポジトリを構成する方法

この手順では、1 つまたは複数の監査対象システムから監査レコードを受信して格納するためのリモートシステムである監査リモートサーバー (ARS) を構成します。次に、リモートサーバー上で監査デーモンをアクティブにします。

この構成は 2 つの部分から成ります。最初に、監査データをセキュアに転送するためのベースとなるセキュリティーメカニズム、つまり、KDC を構成します。2 番目に、監査対象システムと ARS の両方に監査サービスを構成します。この手順は、ARS と KDC が同じサーバー上に存在している、1 つの監査対象クライアントと 1 つの ARS を含むシナリオを示しています。より複雑なシナリオも同様に構成できます。最初の 4 つの手順では、KDC の構成について説明します。最後の手順では、監査サービスの構成について説明します。

始める前に 次を完了したことを確認します。

- `root` 役割になっています。
- 38 ページの「監査レコードのリモートストレージへのストリーム出力を準備する方法」の説明に従って、Kerberos パッケージをインストールしました。
- 86 ページの「監査ファイルをリモートリポジトリに送信する方法」の説明に従って、監査対象システムを構成した管理者とともに作業しています。

1. **サイトで KDC が構成されていない場合は、構成します。**

   監査対象システムと ARS の両方が使用できるシステム上の KDC、各システムのホスト主体、および `audit` サービス主体が必要です。次の例は、KDC 構成の方針を示しています。

   ```
   arstore # kdcmgr -a audr/admin -r EXAMPLE.COM create master
   ```

このコマンドは、管理主体 audr/admin を使用して EXAMPLE.COM レルム内にマスター KDC を作成し、そのマスター KDC を有効にして、Kerberos サービスを開始します。

2. **KDC が使用可能なことを確認します。**

   詳細は、kdcmgr(1M) のマニュアルページを参照してください。

```
# kdcmgr status

KDC Status Information
------------------------------------------
svc:/network/security/krb5kdc:default (Kerberos key distribution center)
State: online since Wed Feb 29 01:59:27 2012
See: man -M /usr/share/man -s 1M krb5kdc
See: /var/svc/log/network-security-krb5kdc:default.log
Impact: None.

KDC Master Status Information
------------------------------------------
svc:/network/security/kadmin:default (Kerberos administration daemon)
State: online since Wed Feb 29 01:59:28 2012
See: man -M /usr/share/man -s 1M kadmind
See: /var/svc/log/network-security-kadmin:default.log
Impact: None.

Transaction Log Information
------------------------------------------

Kerberos update log (/var/krb5/principal.ulog)
Update log dump :
Log version # : 1
Log state : Stable
Entry block size : 2048
Number of entries : 13
First serial # : 1
Last serial # : 13
First time stamp : Wed Feb 29 01:59:27 2012
Last time stamp : Mon Mar 5 19:29:28 2012


Kerberos Related File Information
------------------------------------------
(Displays any missing files)
```

3. **KDC キータブファイルに audit サービス主体を追加します。**

   KDC システム上で kadmin.local コマンドを入力することによって、この主体を追加できます。または、kadmin コマンドを使用し、パスワードを指定することによって、リモートから主体を追加できます。この例では、arstore システムが KDC を実行しています。

```
# kadmin -p audr/admin

kadmin: addprinc -randkey audit/arstore.example.com@EXAMPLE.COM

kadmin: ktadd audit/arstore.example.com@EXAMPLE.COM
```

4. **各監査対象システム上で、鍵を追加します。**

   受信者と送信者は鍵を持っている必要があります。

```
enigma # kclient

.. Enter the Kerberos realm:
EXAMPLE.COM

.. KDC hostname for the above realm:
arstore.example.com

.. Will this client need service keys ? [y/n]:
y
```

5. **ARS 上で監査サービスを構成します。**

   - **Kerberos レルム内の任意の監査対象システムから監査レコードを受け入れる接続グループを作成して、名前を付けます。**

     ```
     # auditconfig -setremote group create Bank_A
     ```

     Bank_A が接続グループです。hosts 属性が定義されていないため、このグループはすべての接続を受け入れます。つまり、これはワイルドカードグループです。audit_remote プラグインが正しく構成されているこの Kerberos レルム内の監査対象システムはすべて、この ARS に到達できます。

   - **このグループへの接続を制限するには、このリポジトリを使用できる監査対象システムを指定します。**

     ```
     # auditconfig -setremote group Bank_A "hosts=enigma.example.com"
     ```

     接続グループ Bank_A は現在、enigma システムからの接続のみを受け入れます。その他のどのホストからの接続も拒否されます。

   - **このグループ内の監査ファイルが大きくなりすぎないようにするには、最大サイズを設定します。**

     ```
     # auditconfig -setremote group Bank_A "binfile_fsize=4GB"

     # auditconfig -getremote
     Audit Remote Server
     Attributes: listen_address=;login_grace_time=30;max_startups=10;listen_port=0;
     Connection group: Bank_A (inactive)
     Attributes: binfile_dir=/var/audit;binfile_fsize=4GB;binfile_minfree=1;
     hosts=enigma.example.com;
     ```

6. **監査対象システム上で監査サービスを構成します。**

   ARS を指定するには、p_hosts 属性を使用します。

   ```
   enigma # auditconfig -setplugin audit_remote \
        active p_hosts=arstore.example.com

   enigma # auditconfig -getplugin audit_remote
   Plugin: audit_remote
   Attributes: p_retries=3;p_timeout=5;p_hosts=arstore.example.com;
   ```

7. **監査サービスをリフレッシュします。**

監査サービスは、リフレッシュ時に監査プラグインの変更を読み取ります。

```
# audit -s
```

KDC は現在、監査対象システム enigma と ARS の間の接続を管理しています。

例 27 監査レコードの同じ ARS 上の異なるファイルの場所へのストリーム出力

この例では、手順にある例を拡張します。管理者は、2 つの接続グループを作成することによって、ARS 上のホストごとに監査レコードを分離します。

audsys1 からの監査ファイルは、この ARS 上の Bank_A の接続グループにストリーム出力されます。

```
arstore # auditconfig -setremote group create Bank_A

arstore # auditconfig -setremote group active Bank_A "hosts=audsys1" \
   "hosts=audsys1;binfile_dir=/var/audit/audsys1;binfile_fsize=4M;"
```

audsys2 からの監査ファイルは、Bank_B の接続グループにストリーム出力されます。

```
arstore # auditconfig -setremote group create Bank_B

arstore # auditconfig -setremote group active Bank_B \
"hosts=audsys2;binfile_dir=/var/audit/audsys2;binfile_fsize=4M;"
```

保守を容易にするために、管理者はその他の属性値を同様に設定します。

```
arstore # auditconfig -getremote
Audit Remote Server
Attributes: listen_address=;login_grace_time=30;max_startups=10;listen_port=0;

Connection group: Bank_A
Attributes: binfile_dir=/var/audit/audsys1;binfile_fsize=4M;binfile_minfree=1;
hosts=audsys1

Connection group: Bank_B
Attributes: binfile_dir=/var/audit/audsys2;binfile_fsize=4M;binfile_minfree=1;
hosts=audsys2
```

例 28 KDC の別のシステム上への ARS の配置

この例では、管理者は ARS を KDC の別のシステム上に配置します。最初に、管理者はマスター KDC を作成して構成します。

```
kserv # kdcmgr -a audr/admin -r EXAMPLE.COM create master

kserv # kadmin.local -p audr/admin

kadmin: addprinc -randkey \
audit/arstore.example.com@EXAMPLE.COM

kadmin: ktadd -k /var/user/root/krb5.keytab.audit \
     audit/arstore.example.com@EXAMPLE.COM
```

/tmp/krb5.keytab.audit ファイルを ARS (arstore) にセキュアに転送したあと、管理者はこのファイルを正しい場所に移動します。

```
arstore # chown root:root krb5.keytab.audit

arstore # chmod 600 krb5.keytab.audit

arstore # mv krb5.keytab.audit /etc/krb5/krb5.keytab
```

このファイルを書き換えるのではなく、管理者には、ARS 上で ktutil コマンドを使用して KDC の krb5.keytab.audit ファイルを arstore の /etc/krb5/krb5.keytab ファイル内の既存の鍵とマージする方法もあります。

最後に、管理者は監査対象システム上で鍵を生成します。

```
enigma # kclient

.. Enter the Kerberos realm: EXAMPLE.COM

.. KDC hostname for the above realm: kserv.example.com

.. Will this client need service keys ? [y/n]: y
```

▼ syslog 監査ログの構成方法

監査サービスに、監査キュー内の監査レコードの一部またはすべてを syslog ユーティリティーにコピーするよう指示できます。バイナリ監査データと概要テキストの両方を記録した場合、バイナリデータでは完全な監査レコードが提供されるのに対して、概要ではリアルタイムで確認できるようにデータがフィルタリングされます。

始める前に audit_syslog プラグインを構成するには、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。syslog ユーティリティーを構成して auditlog ファイルを作成するには、root 役割になる必要があります。

1. **audit_syslog プラグインに送信される監査クラスを選択し、このプラグインをアクティブにします。**

   注記 - p_flags 監査クラスが、システムのデフォルトとして、あるいはユーザーまたは権利プロファイルの監査フラグで事前選択されている必要があります。事前選択されていないクラスについてレコードは収集されません。

   ```
   # auditconfig -setplugin audit_syslog \
       active p_flags=lo,+as,-ss
   ```

2. **syslog ユーティリティーを構成します。**

   a. **syslog.conf ファイルに audit.notice エントリを追加します。**

      エントリは、ログファイルの場所を含みます。

      ```
      # cat /etc/syslog.conf
      ```

```
…
audit.notice       /var/adm/auditlog
```

b. ログファイルを作成します。

```
# touch /var/adm/auditlog
```

c. ログファイルのアクセス権を 640 に設定します。

```
# chmod 640 /var/adm/auditlog
```

d. システムで実行中のシステムログサービスインスタンスを確認します。

```
# svcs system-log

STATE        STIME      FMRI
online       Nov_27     svc:/system/system-log:default
disabled     Nov 27     svc:/system/system-log:rsyslog
```

e. アクティブな syslog サービスインスタンスに関する構成情報をリフレッシュします。

```
# svcadm refresh system/system-log:default
```

3. 監査サービスをリフレッシュします。

リフレッシュ時に、監査サービスによって変更が監査プラグインに読み込まれます。

```
# audit -s
```

4. syslog ログファイルを定期的に保管します。

監査サービスでは、大量の出力が生成される可能性があります。ログの管理方法については、logadm(1M) のマニュアルページを参照してください。

例 29 syslog 出力の監査クラスを指定する

次の例では、syslog ユーティリティーによって、事前選択された監査クラスのサブセットが収集されます。pf クラスは、例15「新しい監査クラスを作成する」で作成されます。

```
# auditconfig -setnaflags lo,na

# auditconfig -setflags lo,ss

# usermod -K audit_flags=pf:no jdoe

# auditconfig -setplugin audit_syslog \
    active p_flags=lo,+na,-ss,+pf
```

auditconfig コマンドの引数はシステムに、すべてのログイン/ログアウト、ユーザーに起因しないイベント、およびシステム状態監査レコードの変更を収集するよう指示します。audit_syslog プラグインエントリは syslog ユーティリティーに、すべての

ログイン、ユーザーに起因しない成功したイベント、およびシステム状態の失敗した変更を収集するよう指示します。

jdoe ユーザーに関して、このバイナリユーティリティーは、pfexec コマンドへの成功した呼び出しと失敗した呼び出しを収集します。syslog ユーティリティーは、pfexec コマンドの成功した呼び出しを収集します。

例 30 syslog 監査レコードをリモートシステムに配置する

syslog.conf ファイル内の audit.notice エントリを変更して、リモートシステムを指定します。この例では、ローカルシステムの名前は sys1.1 です。リモートシステムは、remote1 です。

```
sys1.1 # cat /etc/syslog.conf

…
audit.notice       @remote1
```

remote1 システム上の syslog.conf ファイル内にある audit.notice エントリはログファイルを指定します。

```
remote1 # cat /etc/syslog.conf

…
audit.notice       /var/adm/auditlog
```

♦ ♦ ♦ 第 5 章

5 監査データの操作

この章では、さまざまなローカルシステムから生成される監査データを操作する際に役立つ手順について説明します。この章で扱う内容は、次のとおりです。

- 95 ページの「監査トレールデータの表示」
- 103 ページの「ローカルシステム上の監査レコードの管理」

さらに、次の章では、その他の監査管理タスクについて説明します。

- 第3章「監査サービスの管理」
- 第4章「システム動作のモニタリング」
- 第6章「監査サービスに関する問題の分析および解決」

監査サービスの概要については、第1章「Oracle Solaris での監査について」を参照してください。計画の提案については、第2章「監査の計画」を参照してください。参照情報については、第7章「監査の参照情報」を参照してください。

監査トレールデータの表示

デフォルトのプラグイン `audit_binfile` によって監査トレールが作成されます。このトレールには大量のデータが含まれる場合があります。次のセクションでは、このデータを操作する方法について説明します。

監査レコード定義の表示

監査レコード定義を表示するには、`auditrecord` コマンドを使用します。この定義によって、監査イベントの監査イベント番号、監査クラス、選択マスク、およびレコード書式が提供されます。

```
% auditrecord -options
```

コマンドで生成される画面出力は、使用するオプションによって異なります (次の部分的な一覧を参照)。

- -p オプションは、プログラムの監査レコード定義を表示します。
- -c オプションは、監査クラスの監査レコード定義を表示します。
- -a オプションを指定すると、すべての監査イベント定義が一覧表示されます。

表示された出力をファイルに出力することもできます。

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

例 31 プログラムの監査レコード定義の表示

この例では、login プログラムによって生成されたすべての監査レコードの定義が表示されます。ログインプログラムには、rlogin、telnet、newgrp、および Oracle Solaris の Secure Shell 機能が含まれています。

```
% auditrecord -p login
...
login: logout
program     various              See login(1)
event ID    6153                 AUE_logout
class       lo                  (0x0000000000001000)
...
newgrp
program     newgrp               See newgrp login
event ID    6212                 AUE_newgrp_login
class       lo                  (0x0000000000001000)
...
rlogin
program     /usr/sbin/login      See login(1) - rlogin
event ID    6155                 AUE_rlogin
class       lo                   (0x0000000000001000)
...
/usr/lib/ssh/sshd
program     /usr/lib/ssh/sshd    See login - ssh
event ID    6172                 AUE_ssh
class       lo                   (0x0000000000001000)
...
telnet login
program     /usr/sbin/login      See login(1) - telnet
event ID    6154                 AUE_telnet
class       lo                   (0x0000000000001000)
…
```

例 32 監査クラスの監査レコード定義の表示

この例では、例15「新しい監査クラスを作成する」で作成された pf クラス内のすべての監査レコードの定義が表示されます。

```
% auditrecord -c pf
pfexec
system call pfexec               See execve(2) with pfexec enabled
```

```
event ID    116                 AUE_PFEXEC
class       pf                  (0x0100000000000000)
header
path                    pathname of the executable
path                    pathname of working directory
[privileges]            privileges if the limit or inheritable set are changed
[privileges]            privileges if the limit or inheritable set are changed
[process]               process if ruid, euid, rgid or egid is changed
exec_arguments
[exec_environment]      output if arge policy is set
subject
[use_of_privilege]
return
```

use_of_privilege トークンは、特権が使用されている場合は常に記録されます。privileges トークンは、制限または継承可能セットが変更された場合に記録されます。process トークンは、ID が変更された場合に記録されます。これらのトークンをレコードに含めるために、ポリシーオプションは必要ありません。

例 33 監査レコード定義をファイルに出力する

この例では、すべての監査レコード定義を HTML 形式のファイルに出力するために、-h オプションが追加されています。この HTML ファイルをブラウザで表示する場合は、ブラウザの検索ツールを使用して特定の監査レコード定義を検索します。

```
% auditrecord -ah > audit.events.html
```

表示する監査イベントの選択

Audit Review 権利プロファイルが割り当てられている管理者として、auditreduce を使用すると、調査対象の監査レコードをフィルタリングできます。このコマンドは、入力ファイルを結合するときに、関連性の少ないレコードを除外できます。

auditreduce -*option argument* **[***optional-file***]**

ここで、*argument* はオプションで必要な特定の引数です。

次の一覧に、レコード選択オプションとそれに対応する引数の一部を示します。

-c　*argument* が監査クラス (ua など) である場合に、監査クラスを選択します。

-d　特定の日付のイベントをすべて選択します。*argument* の日付の形式は、*yyyymmdd* です。その他の日付オプション (-b や -a など) は、特定の日付の前と後のイベントを選択します。

-u　特定のユーザーのイベント属性をすべて選択します。このオプションには、ユーザー名を指定します。もう 1 つのユーザーオプ

ション -e は、実効ユーザー ID のイベント属性をすべて選択します。

`-g` 特定のグループのイベント属性をすべて選択します。このオプションには、グループ名を指定します。

`-m` 特定の監査イベントのインスタンスをすべて選択します。

`-o` オブジェクトタイプによって選択します。このオプションは、ファイル、グループ、ファイル所有者、FMRI、PID、およびその他のオブジェクトタイプによって選択するために使用します。

optional-file 監査ファイルの名前。

このコマンドでは、すべて大文字のファイル選択オプションも使用されます (次の例を参照)。オプションの完全な一覧については、`auditreduce(1M)` のマニュアルページを参照してください。

例 34 監査ファイルを結合して削減する

この例では、1 か月間のログインおよびログアウトのレコードのみが監査ファイルに保持されています。この例では、現在の日付が 9 月 27 日になっています。監査トレール全体が必要になった場合は、バックアップメディアから監査トレールを復元します。-O オプションを付けると、コマンドの出力が `lo.summary` という名前のファイルに書き込まれます。

```
# cd /var/audit/audit_summary
# auditreduce -O lo.summary -b 20100827 -c lo; compress *lo.summary
```

例 35 ユーザー監査レコードをサマリーファイルにコピーする

この例では、特定のユーザーの名前を含む監査トレール内のレコードがマージされます。-e オプションを指定すると実効ユーザーが検索されます。-u オプションを指定すると、ログインユーザーが検索されます。-O オプションを付けると、出力が `tamiko` ファイルに書き込まれます。

```
# cd /var/audit/audit_summary
# auditreduce -e tamiko -O tamiko
```

表示する情報をさらに絞り込むことができます。次の例では、次のようにフィルタリングされ、`tamikolo` というファイルに出力されます。

- ユーザーのログイン時間およびログアウト時間 (-c オプションで指定)。
- 2013 年 9 月 7 日 (-d オプションで指定)。日付の短い形式は、*yyyymmdd* です。
- ユーザー名 tamiko (-u オプションで指定)。

- 物理マシンの名前 (-M オプションで指定)。

```
# auditreduce -M tamiko -O tamikolo -d 20130907 -u tamiko -c lo
```

例 36 選択したレコードを単一のファイルにマージする

この例では、特定の日のログインおよびログアウトレコードが監査トレールから選択されます。これらのレコードは、ターゲットファイルにマージされます。ターゲットファイルは、監査ルートディレクトリを含むファイルシステム以外のファイルシステムに書き込まれます。

```
# auditreduce -c lo -d 20130827 -O /var/audit/audit_summary/logins

# ls /var/audit/audit_summary/*logins
/var/audit/audit_summary/20130827183936.20130827232326.logins
```

バイナリ監査ファイルの内容の表示

Audit Review 権利プロファイルが割り当てられている管理者として、praudit コマンドを使用すると、バイナリ形式の監査ファイルの内容を表示できます。

```
# praudit options
```

次の一覧に、オプションの一部を示します。これらのオプションのいずれかを -l オプションと組み合わせると、各レコードを 1 行で表示できます。

-s 監査レコードを短い形式 (1 行ごとに 1 トークン) で表示します。

-r 監査レコードを RAW 形式 (1 行ごとに 1 トークン) で表示します。

-x 監査レコードを XML 形式 (1 行ごとに 1 トークン) で表示します。このオプションは、さらに処理する場合に役立ちます。

auditreduce コマンドから praudit 出力をパイプして、auditreduce コマンドと praudit コマンドを同時に使用することもできます。

例 37 監査レコードを短い形式で表示する

この例では、auditreduce コマンドで抽出されたログインおよびログアウトのイベントが短い形式で表示されています。

```
# auditreduce -c lo | praudit -s

header,69,2,AUE_screenlock,,mach1,2010-10-14 08:02:56.348 -07:00
subject,jdoe,root,staff,jdoe,staff,856,50036632,82 0 mach1
return,success,0
```

```
sequence,1298
```

例 38 監査レコードを RAW 形式で表示する

この例では、auditreduce コマンドで抽出されたログインおよびログアウトのイベントが RAW 形式で表示されています。

```
# auditreduce -c lo | praudit -r

21,69,2,6222,0x0000,10.132.136.45,1287070091,698391050
36,26700,0,10,26700,10,856,50036632,82 0 10.132.136.45
39,0,0
47,1298
```

例 39 監査レコードを XML 形式に変換する

この例では、監査レコードを XML 形式に変換します。

```
# praudit -x 20100827183214.20100827215318.logins > 20100827.logins.xml
```

同様に、auditreduce コマンドでフィルタリングされた監査レコードを XML 形式で表示できます。

```
# auditreduce -c lo | praudit -x
<record version="2" event="screenlock - unlock" host="mach1"
iso8601="2010-10-14 08:28:11.698 -07:00">
<subject audit-uid="jdoe" uid="root" gid="staff" ruid="jdoe
rgid="staff" pid="856" sid="50036632" tid="82 0 mach1"/>
<return errval="success" retval="0"/>
<sequence seq-num="1298"/>
</record>
```

スクリプトを使えば、XML ファイルの内容を操作して目的の情報を抽出できます。

例 40 XML 形式の監査レコードをブラウザで読み取り可能にする

xsltproc ツールを使用して XML ファイルのレコードを再フォーマットすると、任意のブラウザで読み取り可能にすることができます。このツールを使用すると、スタイルシートの定義がファイルの内容に適用されます。再フォーマットされた内容を別のファイルに挿入するには、次のように入力します。

```
# auditreduce -c lo | praudit -x | xsltproc - > logins.html
```

ブラウザでは、logins.html の内容が次のような形式で表示されます。

```
                    Audit Trail Data

File: time: 2013-11-04 12:54:28.000 -08:00

Event: login - local
time: 2013-11-04 12:54:28.418 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jdoe uid: jdoe gid: staff ruid: jdoe rgid: staff
      pid: 1534 sid: 3583012893 tid: 0 0 host
RETURN errval: success retval: 0
```

```
Event: connect to RAD
time: 2013-11-04 12:54:52.029 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jdoe uid: jdoe gid: staff ruid: jdoe rgid: staff
      pid: 1835 sid: 3583012893 tid: 0 0 host
RETURN errval: success retval: 0

Event: role login
time: 2013-11-08 08:42:52.286 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jdoe uid: root gid: root ruid: root rgid: root
      pid: 4265 sid: 3583012893 tid: 0 0 host
RETURN errval: success retval: 0

Event: role logout
time: 2013-11-08 08:43:37.125 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jdoe uid: root gid: root ruid: root rgid: root
      pid: 4265 sid: 3583012893 tid: 0 0 host
RETURN errval: success retval: 0

Event: login - ssh
time: 2013-12-23 12:24:37.292 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jsmith uid: jsmith gid: staff ruid: jsmith rgid: staff
      pid: 2002 sid: 39351741 tid: 14632 202240 host.example.com
RETURN errval: success retval: 0

Event: role login
time: 2013-12-23 12:25:07.345 -08:00 vers: 2 mod: fe host: host
SUBJECT audit-uid: jsmith uid: root gid: root ruid: root rgid: root
      pid: 2023 sid: 39351741 tid: 14632 202240 host.example.com
RETURN errval: failure retval: Permission denied

Event: su
time: 2013-12-23 17:19:24.031 -08:00 vers: 2 mod: na host: host
RETURN errval: success retval: 0

Event: su logout
time: 2013-12-23 17:19:24.362 -08:00 vers: 2 mod: na host: host
RETURN errval: success retval: 0

Event: login - ssh
time: 2013-12-23 17:27:21.306 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jsmith uid: jsmith gid: staff ruid: jsmith rgid: staff
      pid: 2583 sid: 3401970889 tid: 13861 5632 host.example.com
RETURN errval: success retval: 0

Event: role login
time: 2013-12-23 17:27:28.361 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jsmith uid: root gid: root ruid: root rgid: root
      pid: 2593 sid: 3401970889 tid: 13861 5632 host.example.com
RETURN errval: success retval: 0

Event: role logout
time: 2013-12-23 17:30:39.029 -08:00 vers: 2 mod: host: host
SUBJECT audit-uid: jsmith uid: root gid: root ruid: root rgid: root
      pid: 2593 sid: 3401970889 tid: 13861 5632 host.example.com
RETURN errval: success retval: 0
```

そのほかのイベント

例 41 `pfedit` レコードのみを表示する

フィルタを使用すると、監査トレールから特定のレコードのみを抽出して、表示できます。この例では、`pfedit` コマンドを使用して取得されたレコードがフィルタリング

されています。サマリーファイルは 20130827183936.20130827232326.logins であるとします。pfedit コマンドを使用すると、AUE_admin_edit イベントが生成されます。したがって、pfedit レコードを抽出するには、次のコマンドを実行します。

```
auditreduce -m AUE_admin_edit 20130827183936.20130827232326.logins | praudit
```

例 42 監査トレール全体を印刷する

印刷コマンドへのパイプを使用すると、監査トレール全体の出力がプリンタに送られます。セキュリティー上の理由から、プリンタへのアクセスは制限されています。

```
# auditreduce | praudit | lp -d example.protected.printer
```

例 43 特定の監査ファイルを表示する

この例では、サマリーログインファイルを端末ウィンドウで調べます。

```
# cd /var/audit/audit_summary/logins

# praudit 20100827183936.20100827232326.logins | more
```

例 44 praudit 出力をスクリプトで処理する

praudit コマンドの出力は、必要に応じてテキストとして操作できます。たとえば、auditreduce コマンドでは選択できないレコードを選択することがあります。単純なシェルスクリプトを使用すると、praudit コマンドの出力を処理できます。次のサンプルスクリプトは、1 つの監査レコードを 1 行にまとめ、ユーザーが指定した文字列を検索したあと、監査ファイルを元の形式に戻します。

```
#!/bin/sh
#
## This script takes an argument of a user-specified string.
#  The sed command prefixes the header tokens with Control-A
#  The first tr command puts the audit tokens for one record
#  onto one line while preserving the line breaks as Control-A
#
praudit | sed -e '1,2d' -e '$s/^file.*$//' -e 's/^header/^aheader/' \\
| tr '\\012\\001' '\\002\\012' \\
| grep "$1" \\      ユーザーが指定した文字列を検索します

| tr '\\002' '\\012'      元の改行区切りを復元します
```

スクリプトの ^a は、^ と a という 2 つの文字ではなく、Control-A です。この接頭辞によって、header トークンが、テキストとして表示される header という文字列と区別されます。

次のようなメッセージは、praudit コマンドを使用するために必要な権限がないことを示しています。

```
praudit: Can't assign 20090408164827.20090408171614.sys1.1 to stdin.
```

プロファイルシェルで `praudit` コマンドを実行します。Audit Review 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 11.3 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。

ローカルシステム上の監査レコードの管理

次のタスクマップは、監査レコードの選択、分析、および管理の手順を示しています。

表 7 ローカルシステム上の監査レコードの管理タスクマップ

タスク	説明	手順
監査レコードをマージします。	複数の監査ディレクトリの監査ファイルを 1 つの監査トレールに結合します。	103 ページの「監査トレールの監査ファイルをマージする方法」
正確でない名前を付けられた監査ファイルを整理します。	監査サービスにより意図的でなく開いたままにされた監査ファイルに最終タイムスタンプを設定します。	105 ページの「`not_terminated` 監査ファイルを整理する方法」
監査トレールのオーバーフローを回避します。	監査ファイルシステムがいっぱいになることを防止します。	106 ページの「監査トレールのオーバーフローの防止」

▼ 監査トレールの監査ファイルをマージする方法

すべての監査ディレクトリの監査ファイルをマージすることによって、監査トレール全体の内容を分析できます。

注記 - 監査トレール内のタイムスタンプは協定世界時 (UTC) であるため、日付と時間を意味あるものにするには、現在のタイムゾーンに変換する必要があります。これらのファイルを `auditreduce` コマンドではなく、標準ファイルコマンドで操作する場合は、常にこの点に注意してください。

始める前に Audit Review 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 11.3 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. **マージされた監査ファイルを格納するためのファイルシステムを作成します。**

ディスク容量の制限に達する可能性を減らすために、このファイルシステムは、元のファイルを格納するために *How to Create ZFS File Systems for Audit Files*で作成したファイルシステムとは80 ページの「監査ファイルのための ZFS ファイルシステムを作成する方法」 内に存在するようにしてください。

2. **監査トレール内の監査レコードをマージします。**

   マージされた監査ファイルを格納するためのディレクトリに移動します。このディレクトリから、指定された接尾辞を含むファイルに監査レコードをマージします。ローカルシステム上の監査トレール内のすべてのディレクトリがマージされ、このディレクトリ内に配置されます。

   ```
   # cd audit-storage-directory
   # auditreduce -Uppercase-option -O suffix
   ```

   大文字オプションを `auditreduce` コマンドに指定すると、監査トレール内のファイルを操作できます。次の大文字オプションがあります。

`-A`	監査トレール内のすべてのファイルを選択します。
`-C`	完全ファイルだけを選択します。
`-M`	特定の接尾辞を持つファイルを選択します。接尾辞は物理マシン名、またはサマリーファイルに指定した接尾辞です。
`-O`	開始時間と終了時間を示す 14 文字のタイムスタンプおよび接尾辞 *suffix* が付いた監査ファイルを現在のディレクトリに作成します。
`-R` *pathname*	代替の監査ルートディレクトリである *pathname* 内の監査ファイルを読み取ることを指定します。
`-S` *server*	指定されたサーバーから監査ファイルを読み取ることを指定します。

   オプションの完全な一覧については、`auditreduce(1M)` のマニュアルページを参照してください。

例 45　サマリーファイルに監査ファイルをコピーする

次の例では、System Administrator 権利プロファイルが割り当てられた管理者が、監査トレールのすべてのファイルを別のファイルシステム上のマージされたファイルにコピーします。`/var/audit/storage` ファイルシステムは、監査のルートファイルシステムとして機能する `/var/audit` とは別のディスク上にあります。

```
$ cd /var/audit/storage
$ auditreduce -A -O All
$ ls /var/audit/storage/*All
20100827183214.20100827215318.All
```

次の例では、完全なファイルのみが、監査トレールから、マージされたファイルにコピーされます。完全パスは、-O オプションの値として指定されます。パスの最後のコンポーネント Complete は、接尾辞として使用されます。

```
$ auditreduce -C -O /var/audit/storage/Complete

$ ls /var/audit/storage/*Complete
20100827183214.20100827214217.Complete
```

次の例では、-D オプションを追加することによって、元の監査ファイルが削除されます。

```
$ auditreduce -C -O daily_sys1.1 -D sys1.1

$ ls *sys1.1
20100827183214.20100827214217.daily_sys1.1
```

▼ not_terminated 監査ファイルを整理する方法

異常なシステム中断が発生した場合、監査サービスは、監査ファイルがまだ開いている状態で終了します。または、ファイルシステムにアクセスできなくなり、システムが強制的に新しいファイルシステムに切り替えられます。この場合、その監査ファイルは監査レコードとして使用されなくなりますが、監査ファイルの終了タイムスタンプとして文字列 not_terminated が付いたままになります。auditreduce -O コマンドを使用して、ファイルに正しいタイムスタンプを付けます。

始める前に Audit Review 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 11.3 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. **監査ファイル上の not_terminated 文字列が付いたファイルを作成順に一覧表示します。**

   ```
   # ls -R1t audit-directory */* | grep not_terminated
   ```

   -R　サブディレクトリ内のファイルを一覧表示します。

   -t　最新のファイルからもっとも古いファイルの順で一覧表示します。

   -1　1 列でファイルを一覧表示します。

2. **古い not_terminated ファイルを整理します。**

   古いファイルの名前を auditreduce -O コマンドに指定します。

   ```
   # auditreduce -O system-name old-not-terminated-file
   ```

3. **古い not_terminated ファイルを削除します。**

```
# rm system-name old-not-terminated-file
```

例 46 閉じた not_terminated 監査ファイルを整理する

次の例では、not_terminated ファイルを検索し、名前を変更して、元のファイルを削除します。

```
ls -R1t */* | grep not_terminated
…/egret.1/20100908162220.not_terminated.egret
…/egret.1/20100827215359.not_terminated.egret

# cd */egret.1
# auditreduce -O egret 20100908162220.not_terminated.egret
# ls -1t
20100908162220.not_terminated.egret      現在の監査ファイル

20100827230920.20100830000909.egret      クリーンアップされた監査ファイル

20100827215359.not_terminated.egret      入力 (古い) 監査ファイル

# rm 20100827215359.not_terminated.egret
# ls -1t
20100908162220.not_terminated.egret      現在の監査ファイル

20100827230920.20100830000909.egret      クリーンアップされた監査ファイル
```

新しいファイルの開始タイムスタンプは、not_terminated ファイル内にある最初の監査イベントの時間を反映します。終了タイムスタンプは、そのファイル内にある最後の監査イベントの時間を反映します。

監査トレールのオーバーフローの防止

セキュリティーポリシーで、すべての監査データを保存することが必須である場合は、次の手順を確認して、監査レコードの損失を防止してください。

注記 - 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。

- audit_binfile プラグインで、最小空き容量サイズを設定します。

  p_minfree 属性を使用します。

  ディスク容量が最小空き容量サイズまでいっぱいになると、audit_warn 電子メールエイリアスによって警告が送信されます。例25「警告のための弱い制限値を設定する」を参照してください。
- 監査ファイルを定期的に保存するスケジュールを設定します。

  ファイルをオフラインのメディアにバックアップして、監査ファイルを保管します。これらのファイルを保存ファイルシステムに移動することもできます。

syslog ユーティリティーを使用してテキスト監査ログを収集している場合は、テキストログをアーカイブします。詳細は、logadm(1M) のマニュアルページを参照してください。

- スケジュールを設定して、アーカイブされた監査ファイルを監査ファイルシステムから削除します。
- 補助情報を保存し保管します。

  監査レコードの解釈に必要な情報を、監査トレールとともに格納します。少なくとも、passwd、group、および hosts を保存します。また、audit_event および audit_class もアーカイブすることがあります。
- 保管した監査ファイルの記録をとります。
- 保存したメディアを適切な方法で保管します。
- ZFS 圧縮を有効にすることによって、必要なファイルシステム容量を削減します。

  監査ファイル専用の ZFS ファイルシステムでは、圧縮によってファイルが大幅に縮小されます。例については、70 ページの「専用ファイルシステム上の監査ファイルを圧縮する方法」を参照してください。

  『Oracle Solaris 11.3 での ZFS ファイルシステムの管理』 の 「ZFS の圧縮、複製解除、暗号化のプロパティー間の関連」も参照してください。
- サマリーファイルを作成して、格納する監査データのボリュームを削減します。

  監査トレールからサマリーファイルを抽出するには、auditreduce コマンドのオプションを使用します。サマリーファイルには、指定された種類の監査イベントのレコードだけが含まれます。サマリーファイルを抽出するには、例34「監査ファイルを結合して削減する」および例36「選択したレコードを単一のファイルにマージする」を参照してください。

♦ ♦ ♦ 第 6 章

6 監査サービスに関する問題の分析および解決

この章では、監査関連の問題のトラブルシューティングを行う際に役立つ手順について説明します。さらに、次の章では、その他の監査管理タスクについて説明します。

- 第3章「監査サービスの管理」
- 第4章「システム動作のモニタリング」
- 第5章「監査データの操作」

監査サービスの概要については、第1章「Oracle Solaris での監査について」を参照してください。計画の提案については、第2章「監査の計画」を参照してください。参照情報については、第7章「監査の参照情報」を参照してください。

監査サービスのトラブルシューティング

このセクションでは、監査に関する問題のデバッグに役立つ監査のさまざまなエラーメッセージ、設定、およびその他のツールで提供される監査について説明します。

一般に、監査サービスでエラーを警告するために、さまざまな通知が送信されます。監査サービスに問題が存在すると判断した場合は、電子メールおよびログファイルを確認します。

- `audit_warn` エイリアスに送信された電子メールを読み取ります。

  `audit_warn` スクリプトは、`audit_warn` 電子メールエイリアスにアラートメッセージを送信します。正しく構成されたエイリアスが存在しない場合、メッセージは `root` アカウントに送信されます。
- 監査サービスのログファイルを確認します。

  `svcs -s auditd` コマンドの出力には、監査サービスが生成する監査ログのフルパスが一覧表示されます。
- システムログファイルを確認します。

  `audit_warn` スクリプトは、`/var/log/syslog` ファイルに `daemon.alert` メッセージを書き込みます。

  `/var/adm/messages` ファイルに情報が含まれている可能性があります。

問題を見つけて修正したあと、監査サービスを有効にするか、または再開します。

```
# audit -s
```

次のセクションでは、発生する可能性がある問題の事例および問題を解決する手順について説明します。

注記 - トラブルシューティングのタスクを実行する前に、適切な承認を持っていることを確認します。たとえば、監査を構成するには、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 11.3 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。

監査レコードが記録されていない

監査機能はデフォルトで有効になります。監査が無効になっていないにもかかわらず、監査レコードがアクティブなプラグインに送信されない場合は、このセクションで説明する次の要因のいずれか、または要因の組み合わせが原因である可能性があります。システムファイルを変更するには、`solaris.admin.edit/`*path-to-system-file* 承認が割り当てられている必要があります。デフォルトでは、root役割がこの承認を持っています。

監査サービスが実行されていない

監査が実行中かどうかを確認するには、次の方法のいずれかを使用します。

- 現在の監査の状況を確認します。

  次の出力は、監査が実行中でないことを示しています。

  ```
  # auditconfig -getcond
  audit condition = noaudit
  ```

  次の出力は、監査が実行中であることを示しています。

  ```
  # auditconfig -getcond
  audit condition = auditing
  ```

- 監査サービスが実行中であることを確認します。

  次の出力は、監査が実行中でないことを示しています。

  ```
  # svcs -x auditd

  svc:/system/auditd:default (Solaris audit daemon)
  State: disabled since Sun Oct 10 10:10:10 2010
  ```

```
Reason: Disabled by an administrator.
See: http://support.oracle.com/msg/SMF-8000-05
See: auditd(1M)
See: audit(1M)
See: auditconfig(1M)
See: audit_flags(5)
See: audit_binfile(5)
See: audit_syslog(5)
See: audit_remote(5)
See: /var/svc/log/system-auditd:default.log
Impact: This service is not running.
```

次の出力は、監査サービスが実行中であることを示しています。

```
# svcs auditd
STATE          STIME    FMRI
online         10:10:10 svc:/system/auditd:default
```

監査サービスが実行中でない場合、有効にします。手順については、48 ページの「監査サービスの有効化および無効化」を参照してください。

監査プラグインがアクティブになっていない

次のコマンドを使用して、プラグインがアクティブになっているかどうかを確認します。監査サービスが動作するには、少なくとも 1 つのプラグインがアクティブになっている必要があります。

```
# audit -v
audit: no active plugin found
```

どのプラグインもアクティブでない場合は、1 つのプラグインをアクティブにします。

```
# auditconfig -setplugin audit_binfile active
# audit -v
configuration ok
```

監査クラスが定義されていない

定義されていない監査クラスを使用しようとしている可能性があります。`pf` クラスの作成については、60 ページの「監査クラスの追加方法」を参照してください。

たとえば、次のフラグのリストには、Oracle Solaris ソフトウェアでは提供されなかった `pf` クラスが含まれています。

```
# auditconfig -getflags
active user default audit flags = pf,lo(0x0100000000000000,00x0100000000001000)
```

```
configured user default audit flags = pf,lo(0x0100000000000000,00x0100000000001000)
```

クラスを定義しない場合は、有効な値で `auditconfig -setflags` コマンドを実行して現在のフラグをリセットします。それ以外の場合は、クラスを定義する際に次の点を確認します。

- 監査クラスが `audit_class` ファイルで定義されている。

  ```
  # grep pf /etc/security/audit_class
  ```
  クラスが存在することを確認します

  ```
  0x0100000000000000:pf:profile
  ```

- マスクが一意である。一意でない場合は、マスクを置き換えてください。

  ```
  # grep 0x0100000000000000 /etc/security/audit_class
  ```
  マスクが一意であることを確認します

  ```
  0x0100000000000000:pf:profile
  ```

監査クラスにイベントが割り当てられていない

使用中のカスタマイズ済みクラスが定義されているが、そのクラスにイベントが割り当てられていない可能性があります。

カスタマイズ済みクラスにイベントが割り当てられているかどうかを確認するには、次の方法のいずれかを使用します。

```
# auditconfig -lsevent | egrep " pf|,pf|pf,"
AUE_PFEXEC      116 pf execve(2) with pfexec enabled

# auditrecord -c pf
```
pf クラスに割り当てられた監査イベントのリスト

イベントがクラスに割り当てられていない場合、適切なイベントをこのクラスに割り当てます。

監査レコードの量が多い

サイトでどのイベントを監査する必要があるかを決定したあと、推奨される次の方法を使用して、必要な情報のみを含む監査ファイルを作成します。ユーザー、役割、および権利プロファイルに監査フラグを割り当てるには、`root` 役割になる必要があります。

- 具体的には、監査証跡へのイベントと監査トークンの追加を回避します。次のポリシーによって、監査トレールのサイズが増加します。

arge	環境変数を execv 監査イベントに追加します。execv イベントの監査のコストは高くなることがありますが、監査レコードへの変数の追加のコストは高くありません。
argv	コマンドのパラメータを execv 監査イベントに追加します。監査レコードにコマンドのパラメータを追加する際のコストは、高くありません。
group	group トークンをオプションの newgroups トークンを含む監査イベントに追加します。
path	path トークンをオプションの path トークンを含む監査イベントに追加します。
public	ファイルイベントが監査されている場合は、公開オブジェクトに対して監査可能なイベントが発生するたびに、監査トレールにイベントを追加します。ファイルクラスには、fa、fc、fd、fm、fr、fw、cl などがあります。公開ファイルの定義については、12 ページの「監査の用語と概念」を参照してください。
seq	シーケンストークンをすべての監査イベントに追加します。
trail	トレーラトークンをすべての監査イベントに追加します。
windata_down	Trusted Extensions が構成されたシステム上で、ラベル付きウィンドウ内の情報がダウングレードされるとイベントを追加します。
windata_up	Trusted Extensions が構成されたシステム上で、ラベル付きウィンドウ内の情報がアップグレードされるとイベントを追加します。
zonename	ゾーン名をすべての監査イベントに追加します。大域ゾーンが唯一の構成されたゾーンである場合は、文字列 zone, global をすべての監査イベントに追加します。

次の監査レコードは、ls コマンドの使用を示しています。ex クラスが監査対象で、デフォルトのポリシーが使用されています。

```
header,129,2,AUE_EXECVE,,mach1,2010-10-14 11:39:22.480 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
subject,jdoe,root,root,root,root,2404,50036632,82 0 mach1
return,success,0
```

すべてのポリシーがオンの場合、同じレコードが次のようになります。

```
header,1578,2,AUE_EXECVE,,mach1,2010-10-14 11:45:46.658 -07:00
path,/usr/bin/ls
attribute,100555,root,bin,21,320271,18446744073709551615
exec_args,2,ls,/etc/security
exec_env,49,MANPATH=/usr/share/man,USER=jdoe,GDM_KEYBOARD_LAYOUT=us,EDITOR=gedit,
LANG=en_US.UTF-8,GDM_LANG=en_US.UTF-8,PS1=#,GDMSESSION=gnome,SESSIONTYPE=1,SHLVL=2,
HOME=/home/jdoe,LOGNAME=jdoe,G_FILENAME_ENCODING=@locale,UTF-8, PRINTER=example-dbl,
...
path,/lib/ld.so.1
attribute,100755,root,bin,21,393073,18446744073709551615
subject,jdoe,root,root,root,root,2424,50036632,82 0 mach1
group,root,other,bin,sys,adm,uucp,mail,tty,lp,nuucp,daemon
return,success,0
zone,global
sequence,197
trailer,1578
```

- `audit_syslog` プラグインを使用して、一部の監査イベントを `syslog` に送信します。

  これらの監査イベントを `audit_binfile` または `audit_remote` プラグインには送信しないでください。この方法は、`syslog` ログに送信する監査イベントのバイナリレコードを保持する必要がない場合にのみ有効です。
- 設定するシステム全体の監査フラグの数を減らし、個々のユーザーを監査します。

  システム全体で監査される監査クラスの数を減らすことによって、すべてのユーザーに対する監査の量を削減します。

  特定のユーザーや役割のイベントを監査するには、`roleadd`、`rolemod`、`useradd`、および `usermod` コマンドの `audit_flags` キーワードを使用します。例については、例29「syslog 出力の監査クラスを指定する」および `usermod(1M)` のマニュアルページを参照してください。

  特定の権利プロファイルのイベントを監査するには、`profiles` コマンドの `always_audit` および `never_audit` プロパティーを使用します。詳細は、`profiles(1)` のマニュアルページを参照してください。

注記 - ほかのセキュリティー属性と同様に、監査フラグは検索順序によって影響を受けます。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 「割り当てられた権利の検索順序」を参照してください。

- 独自のカスタマイズ監査クラスを作成します。

  使用しているシステムで監査クラスを作成できます。これらのクラスに、モニターが必要な監査イベントのみを指定します。手順については、60 ページの「監査クラスの追加方法」を参照してください。

注記 - 監査構成ファイルの変更の影響については、121 ページの「監査構成ファイルとパッケージ化」を参照してください。

バイナリ監査ファイルのサイズが無制限に増大する

Audit Review 権利プロファイルが割り当てられている管理者として、バイナリファイルのサイズを制限すると、アーカイブおよび検索を容易にすることができます。このセクションで説明するオプションのいずれかを使用すると、元のファイルから小さいバイナリファイルを作成することもできます。

- `p_fsize` 属性を使用して、個々のバイナリ監査ファイルのサイズを制限します。

  `p_fsize` 属性については、`audit_binfile(5)` のマニュアルページの「オブジェクト属性」のセクションを参照してください。

  例については、例21「`audit_binfile` プラグインのためのファイルサイズを制限する」を参照してください。
- `auditreduce` コマンドを使用してレコードを選択し、これらのレコードを、さらに分析するためにより小さなファイルに書き込みます。

  `auditreduce -`*lowercase* オプションは特定のレコードを検索します。

  `auditreduce -`*Uppercase* オプションは選択したレコードをファイルに書き込みます。詳細は、`auditreduce(1M)` のマニュアルページを参照してください。95 ページの「監査トレールデータの表示」も参照してください。

ほかのオペレーティングシステムからのログインが監査されていない

Oracle Solaris OS は、ソースには関係なく、すべてのログインを監査できます。ログインが監査されていない場合は、ユーザーに起因するイベントとユーザーに起因しないイベントの両方に `lo` クラスが設定されていない可能性があります。このクラスでは、ログイン、ログアウト、および画面ロックが監査されます。これらのクラスは、デフォルトで監査されます。

注記 - `ssh` ログインを監査するには、システムで Oracle Solaris から `ssh` デーモンが実行されている必要があります。このデーモンは、Oracle Solaris システム上の監査サービスに合わせて変更されています。詳細は、『Oracle Solaris 11.3 での Secure Shell アクセスの管理』 の 「Secure Shell の sunssh 実装」を参照してください。

例 47 ログインが監査されていることの確認

この例では、最初の 2 つのコマンドの出力は、ユーザーに起因するイベントとユーザーに起因しないイベントに lo クラスが設定されていないことを示しています。最後の 2 つのコマンドは、ログインイベントの監査が有効になるように lo クラスを設定します。

```
# auditconfig -getflags
active user default audit flags = as,st(0x20800,0x20800)
configured user default audit flags = as,st(0x20800,0x20800)

# auditconfig -getnaflags
active non-attributable audit flags = na(0x400,0x400)
configured non-attributable audit flags = na(0x400,0x400)

# auditconfig -setflags lo,as,st
user default audit flags = as,lo,st(0x21800,0x21800)

# auditconfig -setnaflags lo,na
non-attributable audit flags = lo,na(0x1400,0x1400)
```

♦ ♦ ♦ 第 7 章

7 監査の参照情報

この章では、監査の重要なコンポーネントについて説明します。この章の内容は次のとおりです。

監査の概要については、第1章「Oracle Solaris での監査について」を参照してください。計画の提案については、第2章「監査の計画」を参照してください。サイトで監査を構成する手順については、次の章を参照してください。

- 第3章「監査サービスの管理」
- 第4章「システム動作のモニタリング」
- 第5章「監査データの操作」
- 第6章「監査サービスに関する問題の分析および解決」

監査サービス

デフォルトでは、監査サービス `auditd` が有効になっています。サービスを有効化、リフレッシュ、または無効化する方法については、48 ページの「監査サービスの有効化および無効化」を参照してください。

顧客の構成がまだ存在しない場合は、次のデフォルトが設定されています。

- すべてのログインイベントが監査されます。

  成功したログイン試行と失敗したログイン試行の両方が監査されます。
- すべてのユーザーがログインおよびログアウトイベントについて、役割の引き受けや画面ロックも含め、監査されます。
- `audit_binfile` プラグインはアクティブです。`/var/audit` に監査レコードが格納され、監査ファイルのサイズには制限がなく、キューサイズは 100 レコードです。
- `cnt` ポリシーが設定されています。

  監査レコードによって使用可能なディスク容量がいっぱいになると、システムは、破棄された監査レコードの数を追跡します。使用可能なディスク容量の残りが 1% になると、警告が発行されます。
- 次の監査キュー制御が設定されています。
  - レコードのロックが発生する前の監査キュー内のレコードの最大数は 100
  - ブロックされた監査プロセスがブロック解除される前の監査キュー内のレコードの最小数は 10
  - 監査キューのバッファーサイズは 8192 バイト
  - 監査トレールへの監査レコードの書き込み間隔は 20 秒

デフォルトの設定を表示するには、46 ページの「監査サービスのデフォルトの表示」を参照してください。

監査サービスでは、一時的な (アクティブな) 値を設定できます。これらの値は、構成された (プロパティー) 値とは異なることがあります。

- 一時的な値は、監査サービスをリフレッシュまたは再開したときには復元されません。

  監査ポリシーと監査キューの制御は、一時的な値を受け入れます。監査フラグには、一時的な値は含まれません。
- 構成された値はサービスのプロパティー値として格納されるため、監査サービスをリフレッシュまたは再開したときに復元されます。

権利プロファイルは、だれが監査サービスを管理できるかを制御します。詳細は、120 ページの「監査を管理するための権利プロファイル」を参照してください。

デフォルトでは、すべてのゾーンが同様に監査されます。120 ページの「監査と Oracle Solaris Zones」を参照してください。

監査サービスのマニュアルページ

次の表に、監査サービスのための主要な管理マニュアルページをまとめています。

マニュアルページ	サマリー
audit(1M)	監査サービスのアクションを制御するコマンド audit -n は、audit_binfile プラグインの新しい監査ファイルを起動します。 audit -s は、監査を有効にしたり、リフレッシュしたりします。 audit -t は、監査を無効にします。 audit -v は、少なくとも 1 つのプラグインがアクティブであることを確認します。
audit_binfile(5)	デフォルトの監査プラグインであり、監査レコードをバイナリファイルに送信します。123 ページの「監査プラグイン」も参照してください。
audit_remote(5)	監査レコードをリモートの受信者に送信する監査プラグイン。
audit_syslog(5)	監査レコードの概要テキストを syslog ユーティリティーに送信する監査プラグイン。
audit_class(4)	監査クラスの定義を含むファイル。上位 8 ビットは、顧客が新しい監査クラスを作成するために使用できます。システムのアップグレードでこのファイルを変更することの効果の詳細は、60 ページの「監査クラスの追加方法」を参照してください。
audit_event(4)	監査イベントの定義を含み、それらのイベントを監査クラスにマップするファイル。このマッピングは変更できます。システムのアップグレードでこのファイルを変更することの効果の詳細は、61 ページの「監査イベントの所属先クラスの変更方法」を参照してください。
audit_flags(5)	監査クラスの事前選択の構文、失敗したイベントのみ、または成功したイベントのみを選択するための接頭辞、および既存の事前選択を変更するための接頭辞について説明しています。
audit.log(4)	バイナリ監査ファイルのネーミング、ファイルの内部構造、およびすべての監査トークンの構造について説明しています。
audit_warn(1M)	監査レコードの書き込み中に監査サービスで異常な状態が発生したときに電子メールエイリアスを通知するスクリプト。このスクリプトをサイトに合わせてカスタマイズすることで、手動による対処が必要な状況を警告したり、このような状況を自動的に処理する方法を指定したりできます。
auditconfig(1M)	監査構成パラメータを取得および設定するコマンド。 取得および設定できるパラメータの一覧を表示するには、オプションを付けずに、この auditconfig を発行します。
auditrecord(1M)	/etc/security/audit_event ファイル内の監査イベントの定義を表示するコマンド。サンプル出力については、95 ページの「監査レコード定義の表示」を参照してください。
auditreduce(1M)	バイナリ形式で格納される監査レコードを事後選択およびマージするコマンド。コマンドを実行すると、1 つまたは複数の入力監査ファイルから監査レコードがマージできます。レコードはバイナリ形式のままです。

マニュアルページ	サマリー
	大文字オプションは、ファイルの選択に影響を与えます。小文字オプションは、レコードの選択に影響を与えます。
auditstat(1M)	カーネル監査の統計情報を表示するコマンド。たとえば、このコマンドは、カーネル監査キュー内のレコードの数、破棄されたレコードの数、およびユーザープロセスがシステムコールの結果としてカーネル内に生成した監査レコードの数を表示できます。
praudit(1M)	標準入力からバイナリ形式の監査レコードを読み取り、それらのレコードを表示可能な書式で表示するコマンド。この入力は、auditreduce コマンドや、1 つの監査ファイルまたは監査ファイルのリストからパイプできます。また、現在の監査ファイルに tail -0f コマンドを使用して入力を生成することもできます。 サンプル出力については、99 ページの「バイナリ監査ファイルの内容の表示」を参照してください。
syslog.conf(4)	監査レコードのテキストサマリーを audit_syslog プラグインの syslog ユーティリティーに送信するように構成されたファイル。

監査を管理するための権利プロファイル

Oracle Solaris は、監査サービスを構成したり、サービスを有効または無効にしたり、監査トレールを分析したりするための権利プロファイルを提供します。監査構成ファイルを編集するには、root の特権が必要です。

- **Audit Configuration –** 管理者が監査サービスのパラメータを構成したり、auditconfig コマンドを実行したりできるようにします。
- **Audit Control -** 管理者が監査サービスを開始、リフレッシュ、および無効化したり、audit コマンドを実行してサービスを開始、リフレッシュ、または停止したりできるようにします。
- **Audit Review –** 管理者が監査レコードを分析できるようにします。この権利プロファイルは、praudit コマンドと auditreduce コマンドを使って監査レコードを読み取る権限を付与します。また、この管理者は auditstat コマンドも実行できます。
- **System Administrator –** Audit Review 権利プロファイルを含みます。System Administrator 権利プロファイルを持つ管理者は、監査レコードを分析できます。

監査サービスを処理するように役割を構成するには、『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 「役割の作成」を参照してください。

監査と Oracle Solaris Zones

非大域ゾーンは、大域ゾーンを監査する場合とまったく同様に監査することも、独自のフラグ、ストレージ、および監査ポリシーを設定することもできます。

すべてのゾーンが大域ゾーンから同様に監査されている場合は、audit_class および audit_event ファイルによって、大域ゾーン内およびすべての非大域ゾーン内での監査のためのクラスからイベントへのマッピングが提供されます。+zonename ポリシーオプションは、レコードをゾーン名で事後選択するために役立ちます。

ゾーンを個別に監査することもできます。大域ゾーンでポリシーオプション perzone が設定されている場合、各非大域ゾーンは独自の監査サービスを実行し、独自の監査キューを処理し、さらに監査レコードの内容と場所を指定します。非大域ゾーンは、ほとんどの監査ポリシーオプションを指定できます。システム全体に影響するポリシーを設定できないため、非大域ゾーンは ahlt または perzone ポリシーを指定できません。詳細は、28 ページの「Oracle Solaris Zones を使用したシステムの監査」および 32 ページの「ゾーン内での監査の計画」を参照してください。

ゾーンの詳細は、『Oracle Solaris ゾーンの紹介』を参照してください。

監査構成ファイルとパッケージ化

Oracle Solaris の監査構成ファイルは、パッケージ内で preserve=renamenew パッケージ属性でマークされています。この属性を使用すると、ファイルを変更でき、これらの変更はパッケージの更新やパッケージの修正が行われたあとも保持されます。preserve 値の効果については、pkg(5) のマニュアルページを参照してください。

これらの構成ファイルはまた、overlay=allow パッケージ属性でもマークされています。この属性を使用すると、これらのファイルを含む独自のパッケージを作成し、Oracle Solaris のファイルをそのパッケージのファイルに置き換えることができます。パッケージ内で overlay 属性を true に設定すると、pkg サブコマンド (verify、fix、revert など) は、そのパッケージに関する結果を返します。詳細については、pkg(1) および pkg(5) のマニュアルページを参照してください。

監査クラス

Oracle Solaris では、多数の監査イベントのための便利なコンテナとして監査クラスが定義されます。

監査クラスを再構成し、新しい監査クラスを作成できます。監査クラス名は、最長 8 文字です。クラスの説明は、72 文字に制限されています。数値と英数字以外の文字が使用できます。詳細は、audit_class(4) のマニュアルページおよび 60 ページの「監査クラスの追加方法」を参照してください。

注意 - `all` クラスを使用すると、大量のデータが生成され、ディスクがすぐにいっぱいになることがあります。`all` クラスは、特別な理由ですべての活動を監査する場合にだけ使用してください。

監査クラスの構文

監査クラス内のイベントは、成功、失敗、およびその両方について監査できます。

- 接頭辞を指定しなかったイベントのクラスは、成功した場合も失敗した場合も監査されます。
- プラス (+) 接頭辞が付いた場合、イベントのクラスは成功した場合のみが監査されます。
- マイナス (-) 接頭辞が付く場合、イベントのクラスは失敗した場合のみが監査されます。
- 現在の事前選択を変更するには、接頭辞または監査フラグの直前にキャレット (^) を追加します。例:
  - システムに対して `ot` が事前選択されており、ユーザーの事前選択が `^ot` である場合、そのユーザーは `other` クラス内のイベントに関して監査されません。
  - システムに対して `+ot` が事前選択されており、ユーザーの事前選択が `^+ot` である場合、そのユーザーは `other` クラス内の成功したイベントに関して監査されません。
  - システムに対して `-ot` が事前選択されており、ユーザーの事前選択が `^-ot` である場合、そのユーザーは `other` クラス内の失敗したイベントに関して監査されません。

監査クラスの事前選択の構文を確認するには、`audit_flags(5)` のマニュアルページを参照してください。

監査クラスとその接頭辞は、次のコマンドで指定できます。

- `auditconfig` コマンドのオプション `-setflags` および `-setnaflags` への引数として。
- `audit_syslog` プラグインの `p_flags` 属性の値として。この属性は、`auditconfig -setplugin audit_syslog active` コマンドのオプションとして指定します。
- `useradd`、`usermod`、`roleadd`、および `rolemod` コマンドの `-K audit_flags=` *always-audit-flags*`:`*never-audit-flags* オプションの値として。
- `profiles` コマンドの `-always_audit` および `-never_audit` プロパティーの値として。

監査プラグイン

監査プラグインでは、監査キューの監査レコードの処理方法を指定します。
監査プラグインは、`auditconfig -setplugin` コマンドへの引数として `audit_binfile`、`audit_remote`、および `audit_syslog` の名前で指定されます。これらのプラグインは、次の属性によってさらに詳細に指定できます。

- `audit_binfile` プラグイン

  `p_dir` 属性 – バイナリデータの送信先

  `p_minfree` 属性 – 管理者に警告が通知される前のディスク上に残っている最小の領域。

  `p_fsize` 属性 – 監査ファイルの最大サイズ。
- `audit_remote` プラグイン

  `p_hosts` 属性 – バイナリ監査データの送信先のリモートの認証された監査サーバー。

  `p_retries` 属性 – リモートの認証された監査サーバーにアクセスするために行う試行の数。

  `p_timeout` 属性 – リモートの認証された監査サーバーに到達するための試行の間隔の秒数。
- `audit_syslog` プラグイン

  `p_flags` 属性 – `syslog` に送信される監査レコードの概要テキストの選択
- すべてのプラグインについて、そのプラグインのためにキューに入れられる監査レコードの最大数 – `qsize` 属性

`audit_binfile(5)`、`audit_remote(5)`、`audit_syslog(5)`、`auditconfig(1M)` の各マニュアルページを参照してください。

監査リモートサーバー

監査リモートサーバー (ARS) は、監査対象システムから監査レコードをセキュアなリンク経由で受信し、それらのレコードを格納します。
この受信は、構成されている次のものに依存します。

- 特定の監査主体と GSS-API メカニズムを備えた Kerberos レルム
- 少なくとも 1 つの構成済みのアクティブな接続グループを含む ARS
- 接続グループ内の少なくとも 1 つの監査対象システム、および構成済みのアクティブな `audit_remote` プラグイン

接続グループは、ARS の group プロパティーで指定されます。ファイル管理の場合は、group で監査ファイルのサイズを制限したり、最小空き容量を指定したりすることができます。異なる接続グループを指定する主な理由は、例27「監査レコードの同じ ARS 上の異なるファイルの場所へのストリーム出力」に示すように、ARS 上の異なる保管場所を指定するためです。

ARS の詳細は、ars(5) のマニュアルページを参照してください。ARS の構成情報については、auditconfig(1M) のマニュアルページにある -setremote オプションを参照してください。

監査対象システムを構成するには、audit_remote(5) のマニュアルページ、および auditconfig(1M) のマニュアルページにある -setplugin オプションを参照してください。

監査ポリシー

監査ポリシーには、監査トレールにトークンまたは情報を追加するかどうかを指定します。

arge、argv、group、path、seq、trail、windata_down、windata_up、および zonename ポリシーは、監査レコードにトークンを追加します。windata_down および windata_up ポリシーは、Oracle Solaris の Trusted Extensions 機能によって使用されます。詳細は、『Trusted Extensions 構成と管理』 の 第 22 章, 「Trusted Extensions と監査」を参照してください。

その他のポリシーでは、トークンは追加されません。public ポリシーによって、公開ファイルの監査が制限されます。perzone ポリシーによって、非大域ゾーンのための個別の監査キューが確立されます。ahlt および cnt ポリシーによって、監査レコードを配信できないときの動作が決定されます。詳細は、124 ページの「非同期イベントおよび同期イベントの監査ポリシー」を参照してください。

各種の監査ポリシーオプションの効果は、39 ページの「監査ポリシーについて」で説明されています。監査ポリシーオプションについては、auditconfig(1M) のマニュアルページの -setpolicy オプションを参照してください。使用可能なポリシーオプションの一覧を表示するには、コマンド auditconfig -lspolicy を実行します。現在のポリシーを表示するには、コマンド auditconfig -getpolicy を実行します。

非同期イベントおよび同期イベントの監査ポリシー

ahlt ポリシーおよび cnt ポリシーは、監査キューがいっぱいで追加のイベントを受け入れられない場合の動作を管理します。

注記 - cnt または ahlt ポリシーは、少なくとも 1 つのプラグインのキューが監査レコードを受け入れることができる場合はトリガーされません。

cnt ポリシーと ahlt ポリシーは独立していますが、関連性があります。ポリシーの組み合わせには、それぞれ次のような効果があります。

- -ahlt +cnt は、出荷時のデフォルトのポリシーです。このデフォルトのポリシーでは、イベントが記録できない場合でも、監査対象イベントを処理できます。

  -ahlt ポリシーでは、非同期イベントの監査レコードがカーネル監査キューに配置できない場合、システムがイベントをカウントして処理を続行します。

  +cnt ポリシーでは、同期イベントがカーネル監査キューに到達しても配置できない場合、システムがイベントをカウントして処理を続行します。

  -ahlt +cnt の構成は通常、処理の続行により監査レコードが失われる可能性があっても処理を続行する必要のある場合に使用します。auditstat drop フィールドは、ゾーンで破棄された監査レコードの数を示します。
- +ahlt -cnt ポリシーでは、非同期イベントをカーネル監査キューに追加できない場合、処理が停止します。

  +ahlt ポリシーでは、非同期イベントの監査レコードがカーネル監査キューに配置できない場合、すべての処理が停止されます。システムはパニック状態になります。非同期イベントは、監査キューには入らず、呼び出しスタックのポインタから復元する必要があります。

  -cnt ポリシーでは、同期イベントがカーネル監査キューに配置できない場合、イベントを配信しようとするスレッドがブロックされます。スレッドは、監査領域が使用可能になるまでスリープキューに配置されます。カウントは保持されません。プログラムは、監査領域が使用可能になるまでハングアップしたように見えることがあります。

  +ahlt -cnt の構成は通常、システムの可用性より監査イベントの記録を優先する場合に使用します。auditstat wblk フィールドは、スレッドがブロックされた回数を示します。

  ただし、非同期イベントが発生した場合、システムがパニック状態になり停止します。監査イベントのカーネルキューは、保存したクラッシュダンプから手動で復元できます。非同期イベントは、監査キューには入らず、呼び出しスタックのポインタから復元する必要があります。
- -ahlt -cnt ポリシーでは、非同期イベントがカーネル監査キューに配置できない場合、イベントがカウントされ処理が続行します。同期イベントがカーネル監査キューに配置できない場合、イベントを配信しようとするスレッドがブロックされます。スレッドは、監査領域が使用可能になるまでスリープキューに配置されます。カウントは保持されません。プログラムは、監査領域が使用可能になるまでハングアップしたように見えることがあります。

  -ahlt -cnt の構成は通常、非同期監査レコードが失われる可能性より、すべての同期監査イベントの記録を優先する場合に使用します。auditstat wblk フィールドは、スレッドがブロックされた回数を示します。

- `+ahlt +cnt` ポリシーでは、非同期イベントがカーネル監査キューに配置できない場合、システムがパニック状態になります。同期イベントがカーネル監査キューに配置できない場合、システムがイベントをカウントして処理を続行します。

プロセスの監査特性

最初のログイン時に次の監査特性が設定されます。

- **プロセス事前選択マスク** – ユーザーの監査マスクが指定されている場合、システム全体の監査マスクとユーザー固有の監査マスクの組み合わせ。ユーザーがログインすると、ログインプロセスは、事前に選択されたクラスを結合し、そのユーザーのプロセスに対する「プロセス事前選択マスク」を確立します。プロセス事前選択マスクは、監査レコードを生成するイベントを指定します。

  さらに、`audit_flags(5)` のマニュアルページで説明されているように、事前選択では、成功したイベントのみの監査、失敗したイベントのみの監査、またはすべてのイベントの監査を指定できます。

  ユーザーのプロセス事前選択マスクを取得する方法は、次のアルゴリズムで表されます。

  `(system-wide default flags  +` *always-audit-classes*`) -` *never-audit-classes*

  `auditconfig -getflags` コマンドの結果から得られたシステム全体の監査クラスを、ユーザーの *always_audit* キーワードの `always-audit-classes` 値にあるクラスに加えます。次に、この合計から、ユーザーの *never-audit-classes* にあるクラスを引きます。`audit_flags(5)` のマニュアルページも参照してください。
- **監査ユーザー ID** – ユーザーがログインすると、プロセスは、変更不可能な監査ユーザー ID を取得します。この ID は、そのユーザーの初期プロセスで起動されたすべての子プロセスによって継承されます。監査ユーザー ID は、説明責任を実施するのに役立ちます。ユーザーが役割を引き受けたあとも、監査ユーザー ID は同じままになります。各監査レコード内に保存されている監査ユーザー ID を使用すると、常に元のログインユーザーまでアクションを追跡できます。
- **監査セッション ID** – 監査セッション ID は、ログイン時に割り当てられます。この ID は、すべての子プロセスによって継承されます。
- **端末 ID** – ローカルログインの場合、端末 ID は、ローカルシステムの IP アドレスと、そのあとに続く、ユーザーがログインした物理デバイスを識別するデバイス番号で構成されます。通常、ログインはコンソールから行われ、そのコンソールデバイスに対応する番号は `0,0` です。リモートログインの場合、端末 ID は、リモートシステムの IP アドレスと、そのあとに続くリモートポート番号およびローカルポート番号で構成されます。

監査トレール

監査トレールには、バイナリ監査ファイルが含まれています。監査トレールは、audit_binfile プラグインで作成されます。監査サービスが監査キュー内のレコードを収集してこのプラグインに送信すると、このプラグインがそれをディスクに書き込みます。

バイナリ監査ファイルの命名規則

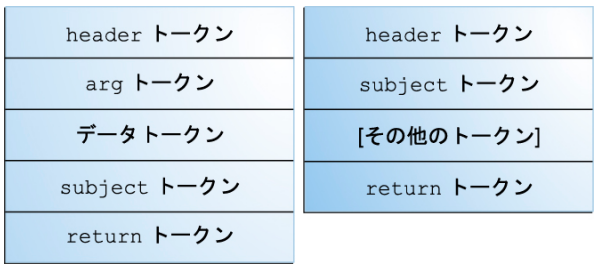
audit_binfile プラグインは、バイナリ監査ファイルを作成します。各バイナリ監査ファイルは、自己完結したレコードの集合です。ファイル名には、レコードが生成された時間の範囲と、それを生成したシステム名が含まれます。時間の範囲を示すタイムスタンプは、タイムゾーンをまたがっている場合でも正しい順序でソートされるようにするために、協定世界時 (UTC) で指定されます。

詳細は、audit.log(4) のマニュアルページを参照してください。開かれた監査ファイル名と閉じられた監査ファイル名の例については、105 ページの「not_terminated 監査ファイルを整理する方法」を参照してください。

監査レコードの構造

監査レコードは、一連の監査トークンです。監査トークンには、ユーザー ID、時間、日付などのイベント情報が入っています。監査レコードは、header トークンで始まり、オプションの trailer トークンで終わります。ほかの監査トークンには、監査イベントに関連する情報が入っています。次の図は、標準的なカーネル監査レコードと標準的なユーザーレベルの監査レコードを示しています。

図 3 標準的な監査レコードの構造



監査レコード分析

監査レコードの分析には、監査トレールからのレコードの事後選択が必要です。次の 2 つの方法のうちのいずれかを使用して、収集されたバイナリデータを解析できます。

- `praudit` コマンドを使用します。コマンドのオプションにより、さまざまなテキスト出力が提供されます。たとえば、`praudit` コマンドを使用すると、スクリプトとブラウザへの入力のための XML が得られます。`praudit` 出力には、バイナリデータの解析に役立つことだけが目的のフィールドは含まれません。`praudit` 出力の順序や形式は Oracle Solaris リリース間では保証されません。

  `praudit` 出力の例については、99 ページの「バイナリ監査ファイルの内容の表示」を参照してください。

  監査トークンごとの `praudit` 出力の例については、128 ページの「監査トークンの形式」にある個々のトークンを参照してください。
- バイナリデータのストリームを解析するためのプログラムを作成できます。このプログラムは、監査レコードの変種を考慮に入れる必要があります。たとえば、`ioctl()` システムコールは、「不正なファイル名」に対する監査レコードを作成します。このレコードには、「無効なファイル記述子」に対する `ioctl()` 監査レコードとは異なるトークンが含まれています。
  - 各監査トークン内のバイナリデータの順番については、`audit.log(4)` のマニュアルページを参照してください。
  - 目録の値については、`/usr/include/bsm/audit.h` ファイルを参照してください。
  - 監査レコード内のトークンの順序を表示するには、`auditrecord` コマンドを使用します。`auditrecord` コマンドの出力には、目録の値ごとに異なるトークンが含まれています。角括弧 (`[]`) は、監査トークンがオプションであることを表しています。詳細は、`auditrecord(1M)` のマニュアルページを参照してください。

監査トークンの形式

各監査トークンにはトークンの種類識別子と、そのあとにトークン固有のデータが続いています。次の表は、各トークンの名前と簡単な説明の一覧です。廃止されたトークンは、以前の Solaris リリースとの互換性のために維持されています。

表 8 監査のための監査トークン

トークン名	説明	詳細
`acl`	アクセス制御エントリ (ACE) とアクセス制御リスト (ACL) の情報	130 ページの「`acl` トークン」
`arbitrary`	書式情報と型情報が付いたデータ	`audit.log(4)` のマニュアルページ

トークン名	説明	詳細
argument	システムコールの引数値	130 ページの「argument トークン」
attribute	ファイル vnode の情報	131 ページの「attribute トークン」
cmd	コマンド引数と環境変数	131 ページの「cmd トークン」
exec_args	exec システムコールの引数	131 ページの「exec_args トークン」
exec_env	exec システムコールの環境変数	131 ページの「exec_env トークン」
exit	プログラム終了情報	audit.log(4) のマニュアルページ
file	監査ファイル情報	132 ページの「file トークン」
fmri	フレームワーク管理リソースインジケータ	132 ページの「fmri トークン」
group	プロセスグループ情報	132 ページの「group トークン」
header	監査レコードの始まりを示します	133 ページの「header トークン」
ip	IP ヘッダー情報	audit.log(4) のマニュアルページ
ip address	インターネットアドレス	133 ページの「ip address トークン」
ip port	インターネットポートアドレス	133 ページの「ip port トークン」
ipc	System V IPC 情報	134 ページの「ipc トークン」
IPC_perm	System V IPC オブジェクトアクセス情報	134 ページの「IPC_perm トークン」
opaque	構造化されていないデータ (形式が未指定)	audit.log(4) のマニュアルページ
path	パス情報	135 ページの「path トークン」
path_attr	アクセスパス情報	135 ページの「path_attr トークン」
privilege	特権設定情報	135 ページの「privilege トークン」
process	プロセス情報	135 ページの「process トークン」
return	システムコールのステータス	136 ページの「return トークン」
sequence	シーケンス番号	136 ページの「sequence トークン」
socket	ソケットの種類とアドレス	136 ページの「socket トークン」
subject	サブジェクト情報 (process と同じ形式)	137 ページの「subject トークン」
text	ASCII 文字列	137 ページの「text トークン」
trailer	監査レコードの終わりを示します	137 ページの「trailer トークン」
use of authorization	承認の使用	138 ページの「use of authorization トークン」
use of privilege	特権の使用	138 ページの「use of privilege トークン」
user	ユーザー ID とユーザー名	138 ページの「user トークン」
xclient	X クライアント ID	138 ページの「xclient トークン」
zonename	ゾーンの名前	139 ページの「zonename トークン」
Trusted Extensions トークン	label と X Window System の情報	『Trusted Extensions 構成と管理』 の「Trusted Extensions の監査のリファレンス」

次のトークンは廃止されました。

- liaison
- host
- tid

廃止されたトークンについては、そのトークンが含まれていたリリースの参照資料を参照してください。

監査レコードは常に、監査トレール内で監査レコードの始まりを示す header トークンで始まります。ユーザーの動作に起因するイベントの場合、subject と process トークンは、イベントを発生させたプロセスの値を参照します。ユーザーに起因しないイベントの場合、process トークンはシステムを参照します。

acl トークン

acl トークンでは、ZFS ファイルシステムのアクセス制御エントリ (ACE) に関する情報を記録する際と、レガシー UFS ファイルシステムのアクセス制御リスト (ACL) に関する情報を記録する際とで、別々の形式が使用されます。

acl トークンが UFS ファイルシステムに対して記録される場合、praudit コマンドでは、フィールドは次のように表示されます。

```
<acl type="1" value="root" mode="6"/>
```

acl トークンが ZFS データセットに対して記録される場合、praudit コマンドでは、フィールドは次のように表示されます。

```
<acl who="root" access_mask="default" flags="-i,-R" type="2"/>
```

argument トークン

argument トークンには、システムコールへの引数に関する情報、つまり、システムコールの引数の数、引数の値、および省略可能な説明が含まれています。このトークンを使用すると、監査レコード内で 32 ビット整数のシステムコール引数を指定できます。

praudit コマンドでは、argument トークンのフィールドは次のように表示されます。

```
<argument arg-num="2" value="0x5401" desc="cmd"/>
```

attribute トークン

attribute トークンには、ファイル vnode からの情報が含まれています。

attribute トークンには通常、path トークンが付いています。attribute トークンはパスの検索中に生成されます。パス検索エラーが発生すると、必要なファイル情報を取得するための vnode が利用できません。このため、attribute トークンは監査レコードの一部として組み込まれません。praudit コマンドでは、attribute トークンのフィールドは次のように表示されます。

```
<attribute mode="20620" uid="root" gid="tty" fsid="0" nodeid="9267" device="108233"/>
```

cmd トークン

cmd トークンは、コマンドに割り当てられた引数のリストおよび環境変数のリストを記録します。

praudit コマンドでは、cmd トークンのフィールドが表示されます。次に示す例は、切り詰められた cmd トークンです。行は、表示の都合上、折り返して記載されています。

```
<cmd><arge>WINDOWID=6823679</arge>
<arge>COLORTERM=gnome-terminal</arge>
<arge>...LANG=C</arge>...<arge>HOST=machine1</arge>
<arge>LPDEST=printer1</arge>...</cmd>
```

exec_args トークン

exec_args トークンは、exec() システムコールへの引数を記録します。

praudit コマンドでは、exec_args トークンのフィールドは次のように表示されます。

```
<exec_args><arg>/usr/bin/sh</arg><arg>/usr/bin/hostname</arg></exec_args>
```

注記 - exec_args トークンは、argv 監査ポリシーオプションが有効なときにだけ出力されます。

exec_env トークン

exec_env トークンは、exec() システムコールの現在の環境変数を記録します。

praudit コマンドでは、exec_env トークンのフィールドが表示されます。次の例では、表示の都合上、行が折り返して記載されています。

```
<exec_env><env>_=/usr/bin/hostname</env>
<env>LANG=C</env><env>PATH=/usr/bin</env>
<env>LOGNAME=jdoe</env><env>USER=jdoe</env>
<env>DISPLAY=:0</env><env>SHELL=/bin/csh</env>
<env>HOME=/home/jdoe</env><env>PWD=/home/jdoe</env><env>TZ=US/Pacific</env>
</exec_env>
```

注記 - exec_env トークンは、arge 監査ポリシーオプションが有効なときにだけ出力されます。

file トークン

file トークンは、古い監査ファイルが終了した時点で、新しい監査ファイルの開始と古い監査ファイルの終了をマークする特殊なトークンです。最初の file トークンは、監査証跡の前のファイルを特定します。最後の file トークンは、監査証跡の次のファイルを特定します。これらのトークンは、連続した監査ファイルを 1 つの監査トレールにまとめて「リンク」します。

praudit コマンドでは、file トークンのフィールドが表示されます。次の例では、表示の都合上、行が折り返して記載されています。

```
<file iso8601="2009-04-08 14:18:26.200 -07:00">
/var/audit/machine1/files/20090408211826.not_terminated.machine1</file>
```

fmri トークン

fmri トークンは、障害管理リソースインジケータ (FMRI) の使用を記録します。詳細は、fmri(5) のマニュアルページを参照してください。

praudit コマンドでは、fmri トークンの内容は次のように表示されます。

```
<fmri service_instance="svc:/system/cryptosvc"</fmri>
```

group トークン

group トークンは、プロセスの資格からグループエントリを記録します。group トークンは、group 監査ポリシーオプションがアクティブなときにだけ出力されます。

praudit コマンドでは、group トークンのフィールドは次のように表示されます。

```
<group><gid>staff</gid><gid>other</gid></group>
```

header トークン

header トークンは、監査レコードの開始を示すという意味で、特殊なトークンです。trailer トークンとの組み合わせでレコード内のほかのすべてのトークンを囲む特殊なトークンです。

まれに、header トークンに 1 つまたは複数のイベント修飾子が含まれている場合があります。

- fe は監査イベントが失敗したことを示す
- fp は特権の使用に失敗したことを示す
- na はユーザーに起因しないイベントを示す

  ```
  header,52,2,system booted,na,mach1,2011-10-10 10:10:20.564 -07:00
  ```

- rd はデータがオブジェクトから読み取られることを示す
- sp は特権の使用に成功したことを示す

  ```
  header,120,2,exit(2),sp,mach1,2011-10-10 10:10:10.853 -07:00
  ```

- wr はデータがオブジェクトに書き込まれることを示す

praudit コマンドでは、header トークンは次のように表示されます。

```
header,756,2,execve(2),,machine1,2010-10-10 12:11:10.209 -07:00
```

praudit コマンドでは、header トークンのフィールドは監査レコードの先頭に表示されます。次の例では、表示の都合上、行が折り返して記載されています。

```
<record version="2" event="execve(2)" host="machine1"
iso8601="2010-10-10 12:11:10.209 -07:00">
```

ip address トークン

ip address トークンには、インターネットプロトコルアドレス (IP アドレス) が含まれています。IP アドレスは、IPv4 または IPv6 の形式で表示できます。IPv4 アドレスは 4 バイトを使用します。IPv6 アドレスは、1 バイトを使って種類を記述し、さらに 16 バイトを使ってアドレスを記述します。

praudit コマンドでは、ip address トークンの内容は次のように表示されます。

```
<ip_address>machine1</ip_address>
```

ip port トークン

ip port トークンには、TCP または UDP ポートアドレスが含まれています。

praudit コマンドでは、ip port トークンは次のように表示されます。

```
ip port,0xf6d6
```

ipc トークン

ipc トークンには、呼び出し元が特定の IPC オブジェクトを識別するために使用する System V IPC メッセージハンドル、セマフォーハンドル、または共有メモリーハンドルが含まれています。

IPC オブジェクト識別子は、監査トークンの、コンテキストに依存しない性質に準拠していません。IPC オブジェクトを一意に識別するグローバルな「名前」はありません。代わりに、IPC オブジェクトはハンドルで識別されます。これらのハンドルは、IPC オブジェクトの動作中にのみアクティブです。しかし IPC オブジェクトの識別は問題となりません。System V の IPC メカニズムはあまり使用されず、すべてのメカニズムが同じ監査クラスを共有するからです。

次の表は、IPC オブジェクトの形式フィールドに指定できる値の一覧です。値は /usr/include/bsm/audit.h ファイル内で定義されます。

表 9 IPC オブジェクトの形式フィールドの値

名前	値	説明
AU_IPC_MSG	1	IPC メッセージオブジェクト
AU_IPC_SEM	2	IPC セマフォーオブジェクト
AU_IPC_SHM	3	IPC 共有メモリーオブジェクト

praudit コマンドでは、ipc トークンのフィールドは次のように表示されます。

```
<IPC ipc-type="shm" ipc-id="15"/>
```

IPC_perm トークン

IPC_perm トークンには、System V IPC アクセス権のコピーが含まれています。このトークンは、IPC 共有メモリーイベント、IPC セマフォーイベント、および IPC メッセージイベントによって生成される監査レコードに追加されます。

praudit コマンドでは、 IPC_perm トークンのフィールドが表示されます。次の例では、表示の都合上、行が折り返して記載されています。

```
<IPC_perm uid="jdoe" gid="staff" creator-uid="jdoe"
creator-gid="staff" mode="100600" seq="0" key="0x0"/>
```

値は、IPC オブジェクトに関連付けられた IPC_perm 構造から取り出されます。

path トークン

path トークンには、オブジェクトのアクセスパス情報が含まれています。

praudit コマンドでは、path トークンの内容は次のように表示されます。

```
<path>/export/home/srv/.xsession-errors</path>
```

path_attr トークン

path_attr トークンには、オブジェクトのアクセスパス情報が含まれています。アクセスパスは、path トークンオブジェクトの下の属性ファイルオブジェクトのシーケンスを指定します。openat() などのシステムコールは、属性ファイルにアクセスします。属性ファイルオブジェクトについての詳細は、fsattr(5) のマニュアルページを参照してください。

praudit コマンドでは、path_attr トークンは次のように表示されます。

```
path_attr,1,attr_file_name
```

privilege トークン

privilege トークンは、プロセス上での特権の使用を記録します。privilege トークンは、基本セットの特権に対して記録されません。特権が管理者の処理により基本セットから削除された場合、その特権の使用は記録されます。特権の詳細は、『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 「プロセス権管理」を参照してください。

praudit コマンドでは、privilege トークンのフィールドが表示されます。

```
<privilege set-type="Inheritable">ALL</privilege>
```

process トークン

process トークンには、シグナルの受信者など、プロセスに関連付けられたユーザーに関する情報が含まれています。

praudit コマンドでは、process トークンのフィールドが表示されます。次の例では、表示の都合上、行が折り返して記載されています。

```
<process audit-uid="-2" uid="root" gid="root" ruid="root"
rgid="root" pid="567" sid="0" tid="0 0 0.0.0.0"/>
```

return トークン

return トークンには、システムコールの戻りステータス (u_error) とプロセスの戻り値 (u_rval1) が含まれています。

return トークンは、必ずシステムコールに関してカーネルによって生成される監査レコードの一部として返されます。アプリケーションの監査中、このトークンは終了ステータスとその他の戻り値を返します。

praudit では、システムコールの return トークンは次のように表示されます。

```
return,failure: Operation now in progress,-1
```

praudit コマンドでは、return トークンのフィールドは次のように表示されます。

```
<return errval="failure: Operation now in progress" retval="-1/">
```

sequence トークン

sequence トークンには、シーケンス番号が含まれています。シーケンス番号は、監査レコードが監査トレールに組み込まれるたびに 1 ずつ増分します。sequence トークンは、seq 監査ポリシーオプションがアクティブなときにだけ出力されます。このトークンはデバッグに使用されます。

praudit コマンドでは、sequence トークンの内容は次のように表示されます。

```
<sequence seq-num="1292"/>
```

socket トークン

socket トークンには、インターネットソケットを記述する情報が含まれています。場合によっては、このトークンにリモートポートとリモート IP アドレスのみが含まれていることがあります。

praudit コマンドでは、socket トークンのインスタンスは次のように表示されます。

```
socket,0x0002,0x83b1,localhost
```

拡張されたトークンによって、ソケットの種類やローカルポートなどの情報が追加されます。

praudit コマンドでは、socket トークンのインスタンスは次のように表示されます。次の例では、表示の都合上、行が折り返して記載されています。

```
<socket sock_domain="0x0002" sock_type="0x0002" lport="0x83cf"
laddr="example1" fport="0x2383" faddr="server1.Subdomain.Domain.COM"/>
```

subject トークン

subject トークンは、ある操作を実行するユーザーまたは実行を試みるユーザーを記述します。形式は process トークンと同じです。

subjectトークンは、必ずシステムコールに関してカーネルによって生成される監査レコードの一部として返されます。praudit コマンドでは、subject トークンは次のように表示されます。

```
subject,jdoe,root,root,root,root,1631,1421584480,8243 65558 machine1
```

praudit コマンドでは、subject トークンのフィールドが表示されます。次の例では、表示の都合上、行が折り返して記載されています。

```
<subject audit-uid="jdoe" uid="root" gid="root" ruid="root"
rgid="root" pid="1631" sid="1421584480" tid="8243 65558 machine1"/>
```

text トークン

text トークンには、テキスト文字列が含まれています。

praudit コマンドでは、text トークンの内容は次のように表示されます。

```
<text>booting kernel</text>
```

trailer トークン

header と trailer の 2 つのトークンは、監査レコードの始端と終端を区別し、ほかのすべてのトークンを囲むという意味で、特殊なトークンです。header トークンは監査レコードを開始します。trailer トークンは監査レコードを終了します。trailer トークンはオプションのトークンであり、trail 監査ポリシーオプションが設定されているときにだけ、各レコードの最後のトークンとして追加されます。

トレーラが有効になっている監査レコードが生成された場合、auditreduce コマンドは、trailer がレコードヘッダーを正しくポイントしているかどうかを検証できます。また、trailer トークンを使用すると監査トレールを逆方向に検索できます。

praudit コマンドでは、trailer トークンが次のように表示されます。

```
trailer,136
```

use of authorization トークン

use of authorization トークンは、承認の使用を記録します。

praudit コマンドでは、use of authorization トークンは次のように表示されます。

```
use of authorization,solaris.role.delegate
```

use of privilege トークン

use of privilege トークンは、特権の使用を記録します。

praudit コマンドでは、use of privilege トークンのフィールドは次のように表示されます。

```
<use_of_privilege result="successful use of priv">proc_setid</use_of_privilege>
```

user トークン

user トークンは、ユーザー名とユーザー ID を記録します。このトークンは、ユーザー名が呼び出し側と異なる場合に存在します。

praudit コマンドでは、user トークンのフィールドは次のように表示されます。

```
<user uid="123456" username="tester1"/>
```

xclient トークン

xclient トークンには、X サーバーへのクライアント接続の数が含まれています。

praudit コマンドでは、xclient トークンの内容は次のように表示されます。

```
<X_client>15</X_client>
```

zonename トークン

zonename トークンは、監査イベントが発生したゾーンを記録します。文字列「global」は、大域ゾーンで発生した監査イベントを示します。

praudit コマンドでは、zonename トークンの内容は次のように表示されます。

```
<zone name="graphzone"/>
```

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。

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。

インスタンス ネットワーク通信の文脈では、インスタンスは主体名の 2 番目の部分です。インスタンスは、主体のプライマリを修飾します。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、およびユーザー主体も参照してください。

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、ネットワーク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、この鍵はネットワーク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。

監査トレール すべてのシステム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。システム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。

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システムプロセスを指します。ネットワークサーバー自体がほか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) ネットワークサーバーから提供されたサービスを利用するシステムを指します。

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システム上の内部システムクロックに許容できる最大時間。参加しているシステム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。

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。

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ネットワーク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。

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、複数のネットワークサーバーから提供されます。たとえば、システム `central.example.com` に `rlogin` で接続する場合、そのシステムが `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。ホスト主体とユーザー主体も参照してください。

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。セッション鍵も参照してください。

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、およびユーザー主体も参照してください。

2.(RPCSEC_GSS API) 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、およびレルムも参照してください。

2.(RPCSEC_GSS API) 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。

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とマスター KDCも参照してください。

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。認証とプライバシも参照してください。

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、クライアントシステムとサービス間のトランザクションをセキュリティー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。

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。

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。継承可能セット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。

認証 特定の主体の識別情報を検証するプロセス。

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。

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、および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、およびレルムも参照してください。

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。

監査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、クライアントシステムに認証サービスを提供する複製 (スレーブ) KDC を多数割り当てることができます。

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。

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。

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。

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、および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 要素で構成されるシステム。

- 主体と鍵データベース
- 認証サービス
- チケット認可サービス

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。

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。

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。

索引

さ

た

や

ら

わ

A

B

T

U

V

W

X

Z