AIX バージョン 7.2

セキュリティー

AIX バージョン 7.2

セキュリティー

お願い

本書および本書で紹介する製品をご使用になる前に、 559 ページの『特記事項』に記載されている情報をお読みください。

本書は AIX バージョン 7.2 および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： AIX Version 7.2
Security

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

目次

本書について

このトピック集はファイル、システム、およびネットワーク・セキュリティーについての完全情報をシステム管理者に提供します。 このトピック集には、システム強化、アクセス権の変更、認証方式の設定、およびセキュリティー評価の共通基準 (Common Criteria Security Evaluation) フィーチャーの構成などのタスクの実行方法に関する情報が含まれています。 このトピック集は、オペレーティング・システムに付属のドキュメンテーション CD でも利用できます。

強調表示

本書では、以下の強調表示規則を使用します。

太字	名前がシステムによって事前定義されているコマンド、サブルーチン、キーワード、ファイル、構造体、ディレクトリー、およびその他の項目を示します。 さらに、ユーザーが選択するボタン、ラベル、およびアイコンなどのグラフィカル・オブジェクトも示します。
イタリック	実際の名前または値をユーザーが指定する必要があるパラメーターを示します。
`Monospace` (モノスペース)	具体的なデータ値の例、表示される可能性があるテキストの例、プログラマーとして作成する可能性があるプログラム・コードの一部の例、システムからのメッセージ、またはユーザーが実際に入力する必要がある情報を示します。

AIX での大/小文字の区別

AIX® オペレーティング・システムでは、すべて大文字小文字の区別をします。これは、英大文字と小文字を区別するということです。 例えば、**ls** コマンドを使用するとファイルをリストできます。 `LS` と入力すると、システムはそのコマンドが `is not found` (見つからない) と応答します。 同様に、**FILEA**、**FiLea**、および **filea** は、同じディレクトリーにある場合でも、3 つの異なるファイル名です。 予期しない処理が実行されないように、常に正しい大/小文字を使用するようにしてください。

ISO 9000

当製品の開発および製造には、ISO 9000 登録品質システムが使用されました。

セキュリティー

AIX オペレーティング・システムでは、システムの強化、アクセス権の変更、認証方式の設定、およびセキュリティー評価共通基準 (Common Criteria Security Evaluation) フィーチャーの構成などのタスクを実行することができます。 このトピック集は、オペレーティング・システムに付属のドキュメンテーション CD でも利用できます。

関連情報:

- カーネギー・メロン大学の Computer Emergency Response Team (CERT)
- Forum of Incident Response and Security Teams (FIRST)
- Center for Education and Research in Information Assurance and Security (CERIAS)

セキュリティーに関する新着情報

セキュリティーのトピック集について新しい情報または大幅に変更された情報を説明します。

新規情報または変更情報の参照方法

この PDF ファイルでは、左マージンに新規および変更情報を示すリビジョン・バー (|) が表示される場合があります。

2017 年 1 月

このトピック集に対する更新の要約を以下に示します。

- 156 ページの『監査イベント』のトピックで、監査イベントに関する情報が追加されました。
- 226 ページの『OpenSSH イメージ』のトピックで、OpenSSH イメージに関する情報が追加されました。

基本オペレーティング・システムの保護

基本オペレーティング・システムの保護では、ネットワークの接続性にかかわらず、システムを保護する方法について説明します。

以下のセクションでは、セキュリティー・オプションをオンにしてシステムをインストールする方法、および非特権ユーザーがシステムにアクセスできないよう AIX を保護する方法を説明します。

セキュア・システムのインストールと構成

AIX のセキュア・インストールと構成には、いくつかの要因が関連します。

トラステッド・コンピューティング・ベース

システム管理者は、特定プログラムをどの程度信頼できるかを判別する必要があります。 この判別には、あるプログラムに特権を与えてインストールするにはどれだけの信頼が必要かを判断する上での、システムの情報リソースの価値を考慮する作業も含まれます。

トラステッド・コンピューティング・ベース (TCB) は、システム全体の機密保護ポリシーの実施を受け持つシステムの部分です。 TCB をインストールして使用すると、トラステッド通信パスへのユーザー・アクセスを定義することができます。この通信パスはユーザーと TCB 間のセキュアな通信を可能にします。 TCB の機能は、オペレーティング・システムがインストールされている場合に限り、使用可能にすることができます。 既にインストール済みのマシンに TCB をインストールするには、保存インストールを実行する必要があります。 TCB を使用可能にすると、トラステッド・シェル、トラステッド・プロセス、およびセキュア・アテンション・キー (SAK) にアクセスできます。

TCB の検査**:**

トラステッド・コンピューティング・ベース (TCB) ファイルが正しく保護されていなかったり、構成ファイルに危険な値があると、オペレーティング・システムのセキュリティーが危うくなります。

tcbck コマンドは、トラステッド・コンピューティング・ベースのセキュリティー状態を監査します。 **tcbck** コマンドは、この情報を監査するために、`/etc/security/sysck.cfg` ファイルを読み取ります。このファイルには、すべての TCB ファイル、構成ファイル、およびトラステッド・コマンドの説明が含まれています。

/etc/security/sysck.cfg ファイルはオフラインでないため、ハッカーにより変更される可能性があります。 TCB を更新するたびに、オフラインの読み取り専用コピーを必ず作成してください。 さらに、検査を行う前に、このファイルをアーカイブ・メディアからディスクにコピーしてください。

sysck.cfg ファイルの構造**:**

tcbck コマンドは、検査するファイルを決定するために、**/etc/security/sysck.cfg** ファイルを読み取ります。 システム上の各トラステッド・プログラムは、**/etc/security/sysck.cfg** ファイル内のスタンザで記述されています。

各スタンザには以下の属性があります。

属性	説明
acl	このファイルのアクセス制御リストを表すテキスト文字列。 これは、**aclget** コマンドの出力と同じフォーマットでなければなりません。 これが実際のファイル ACL (アクセス制御リスト) と一致しない場合、**sysck** コマンドは **aclput** コマンドを使用してこの値を適用します。 **注:** SUID、SGID、および SVTX 属性は、モードに指定された属性がある場合はそれらと一致しなければなりません。
class	ファイルのグループの名前。 この属性により、**tcbck** コマンドへの単一の引数を指定して、同じクラス名の複数のファイルを検査することができます。 複数のクラスも、各クラスをコンマで区切ることによって指定することができます。
group	ファイル・グループのグループ ID または名前。 これがファイル・グループと一致しない場合、**tcbck** コマンドがファイルのグループ ID をこの値に設定します。
links	このファイルにリンクされたパス名の、コンマで区切ったリスト。 このリスト内のパス名のいずれかがこのファイルにリンクされていない場合、**tcbck** コマンドがリンクを作成します。 *tree* パラメーターを指定しない場合は、**tcbck** コマンドは、余分のリンクがあるというメッセージを出しますが、それらの名前を判別しません。 *tree* パラメーターを使用した場合は、**tcbck** コマンドはこのファイルにリンクされている追加のパス名もすべて表示します。
mode	コンマで区切られた値のリスト。 暗黙的値は SUID、SGID、SVTX、および TCB です。 ファイル許可は、最後の値でなければならず、8 進値として、あるいは 9 文字の文字列として指定することができます。 例えば、755 または rwxr-xr-x は、いずれも有効なファイル許可です。 これが実際のファイル・モードと一致しない場合、**tcbck** コマンドが適切な値を適用します。
owner	ファイル所有者のユーザー ID または名前。 これがファイル所有者と一致しない場合、**tcbck** コマンドがファイルの所有者 ID をこの値に設定します。

属性	説明
program	コンマで区切られた値のリスト。 最初の値は、検査プログラムのパス名です。 それ以外の値は、プログラムが実行されるときにそのプログラムに引数として渡されます。 **注:** 最初の引数は、*-y*、*-n*、*-p*、または *-t* のいずれかになります。どのフラグに **tcbck** コマンドが使用されたかによって決まります。
source	検査の前に、このソース・ファイルがコピーされるそのコピー元のファイル名。 その値がブランクであり、これが正規ファイル、ディレクトリー、あるいは名前付きパイプである場合、このファイルの空の新規バージョンがまだない場合には、それが作成されます。 デバイス・ファイルの場合、新規のスペシャル・ファイルが同じタイプのデバイス用に作成されます。
symlinks	このファイルにシンボリックにリンクされたパス名の、コンマで区切られたリスト。 このリスト内のパス名のいずれかがこのファイルへのシンボリック・リンクでない場合、**tcbck** コマンドがシンボリック・リンクを作成します。 *tree* 引数を使用した場合は、**tcbck** コマンドはこのファイルへのシンボリック・リンク先である追加のパス名もすべて表示します。

/etc/security/sysck.cfg ファイル内のスタンザが属性を指定していない場合、対応する検査は実行されません。

tcbck コマンドの使用**:**

tcbck コマンドは、セキュリティー関連ファイルが正しくインストールされており、明らかにシステム・セキュリティーを侵害するファイルがファイルシステム・ツリーに含まれていないことを確認し、さらにトラステッド・ファイルの更新、追加、または削除を行うために使用されます。

tcbck コマンドは、通常、次のタスクのために使用します。

- セキュリティー関連ファイルのインストールを確実に行うため。
- ファイルシステム・ツリーに、明らかにシステム・セキュリティーに違反しているファイルが含まれないようにするため。
- トラステッド・ファイルの更新、追加、削除を行うため。

tcbck コマンドは、以下の方法で使用することができます。

- 通常の使用
  - システムの初期化時に非対話式に使用する
  - **cron** コマンドと併用する
- 対話式の使用
  - 個々のファイルおよびファイルのクラスを確認する
- パラノイア的使用
  - `sysck.cfg` ファイルをオフラインで保管し、定期的に復元してマシンを確認する

暗号化の面では安全ではありませんが、TCB はチェックサムに **sum** コマンドを使用します。 別の checksum コマンドを使用して TCB データベースを手動でセットアップすることができます。例えば、**md5sum** コマンドを使用できます。このコマンドは *AIX Toolbox for Linux Applications CD* に付属の textutil RPM Package Manager パッケージで出荷されます。

トラステッド・ファイルの検査*:*

tcbck コマンドは、tcbck データベース内のすべてのファイルの検査と修正、およびすべてのエラーの修正とログ作成のために使用します。

tcbck データベース内のすべてのファイルを検査し、すべてのエラーを修正して報告するには、次のように入力します。

```
tcbck -y ALL
```

この場合、**tcbck** コマンドは、**/etc/security/sysck.cfg** ファイルに記述されている tcbck データベース内の各ファイルのインストールを検査します。

システムの初期化中にこの検査を自動的に実行し、エラー内容のログを作成するには、上記のコマンド・ストリングを **/etc/rc** コマンドに追加します。

ファイルシステム・ツリーの検査*:*

システムの保全性が損なわれている可能性があると思われる場合は、そのつど **tcbck** コマンドを実行してファイルシステム・ツリーを検査します。

ファイルシステム・ツリーを検査するには、次のように入力します。

```
tcbck -t tree
```

tcbck コマンドで *tree* 値を使用すると、システムのすべてのファイルについて、インストールが正しく行われているかどうかが検査されます (時間が長くかかることがあります)。 **tcbck** コマンドがシステム・セキュリティーに危険を及ぼす可能性のあるファイルを発見した場合は、その疑いのあるファイルを更新して、問題の属性を除去することができます。 さらに、ファイルシステム内の他のすべてのファイルについて、以下の検査が行われます。

- ファイル所有者が root であって、そのファイルに SetUID ビットが設定されていると、その SetUID ビットはクリアされます。
- ファイル・グループが管理グループであり、ファイルが実行可能であって、そのファイルに SetGID ビットが設定されている場合、その SetGID ビットはクリアされます。
- ファイルに **tcb** 属性が設定されている場合、この属性はクリアされます。
- ファイルがデバイス (文字またはブロック・スペシャル・ファイル) である場合、このファイルは除去されます。
- ファイルが **/etc/security/sysck.cfg** ファイルに記述されているパス名への追加リンクである場合、そのリンクは除去されます。
- ファイルが **/etc/security/sysck.cfg** ファイルに記述されているパス名への追加シンボリック・リンクである場合、そのシンボリック・リンクは除去されます。

  注**: tcbck** コマンドを実行する前に、すべてのデバイス・エントリーを **/etc/security/sysck.cfg** ファイルに追加しておく必要があります。これを行わないと、システムが使用できなくなります。 **/etc/security/sysck.cfg** ファイルにトラステッド・デバイスを追加するには、**-l** フラグを使用します。

重要**:** **tcbck -y tree** コマンド・オプションは実行しないでください。 このオプションを実行すると、TCB に正しくリストされていないデバイスが削除されて使用不可になるため、システムが使用できなくなる可能性があります。

トラステッド・プログラムの追加*:*

tcbck コマンドは、特定のプログラムを /etc/security/sysck.cfg ファイルに追加するために使用します。

特定のプログラムを /etc/security/sysck.cfg ファイルに追加するには、次のように入力します。

```
tcbck -a PathName [Attribute=Value]
```

値がファイルの現在の状態から推定されない属性だけを、コマンド・ラインに指定する必要があります。属性名はすべて **/etc/security/sysck.cfg** ファイルに入れられます。

例えば、次のコマンドでは、**/usr/bin/setgroups** という新しい SetUID ルート・プログラムが、**/usr/bin/getgroups** というリンクを持つものとして登録されます。

```
tcbck -a /usr/bin/setgroups links=/usr/bin/getgroups
```

管理ユーザーとして jfh と jsl を追加し、管理グループとして developers を追加して、**/usr/bin/abc** ファイルのセキュリティー監査時に検査対象とするには、次のように入力します。

```
tcbck -a /usr/bin/abc setuids=jfh,jsl setgids=developers
```

プログラムをインストールした後に、**/etc/security/sysck.cfg** ファイルに登録が必要な新規ファイルがどれであるかわからない場合があります。 これらのファイルは、次のコマンドで検出して、追加することができます。

```
tcbck -t tree
```

このコマンド・ストリングは、**/etc/security/sysck.cfg** ファイルに登録する必要のあるファイルの名前を表示します。

トラステッド・プログラムの削除*:*

/etc/security/sysck.cfg ファイルに記述されているファイルをシステムから除去する場合は、そのファイルの記述も **/etc/security/sysck.cfg** ファイルから除去する必要があります。

例えば、**/etc/cvid** プログラムを削除したとき、次のコマンド・ストリングを実行するとエラー・メッセージが表示されます。

```
tcbck -t ALL
```

結果のエラー・メッセージは次のとおりです。

```
3001-020 The file /etc/cvid was not found.
```

これはこのプログラムの記述が **/etc/security/sysck.cfg** ファイルに残っているためです。このプログラムの記述を除去するには、次のコマンドを入力します。

```
tcbck -d /etc/cvid
```

追加のトラステッド・オプションの構成*:*

トラステッド・コンピューティング・ベース (TCB) の追加オプションを構成することができます。

端末装置へのアクセス制限*:*

端末アクセスを制限するようにオペレーティング・システムを構成することができます。

getty コマンドおよび **shell** コマンドを使用すると、端末装置の所有者とモードが変更されて、非トラステッド・プログラムによる端末アクセスが防止されます。 このオペレーティング・システムでは、排他的な端末アクセスを構成する方法を提供しています。

セキュア・アテンション・キーの使用*:*

トラステッド通信パスは、セキュア・アテンション・キー (SAK) 予約済みキー・シーケンス (Ctrl-X、Ctrl-R の順) を押すことによって確立されます。

注**:** SAK は注意して使用してください。このキーにより、端末装置にアクセスしようとするすべてのプロセスと、その端末装置へのすべてのリンク (例えば、/dev/console は /dev/tty0 にリンクできます) が停止されます。

トラステッド通信パスの確立は、以下の条件のもとで行われます。

- システムにログインしているとき。

  SAK を押した後。
  - 新規にログイン・スクリーンが表示された場合、セキュア・パスができています。
  - トラステッド・シェルのプロンプトが表示された場合、その初期ログイン・スクリーンは、パスワードを盗もうとしていた無許可のプログラムであった、ということになります。 **who** コマンドを使用して、現在だれがこの端末装置を使用しているかを突き止めてから、ログオフしてください。
- 入力したコマンドによってトラステッド・プログラムを実行させたい場合。この例として、以下の場合があります。
  - root ユーザーとしての実行。 root ユーザーとして実行するのは、必ずトラステッド通信パスを確立した後にしてください。 このようにすれば、非トラステッド・プログラムが root ユーザー権限を使用して実行することが起こりません。
  - **su** コマンド、**passwd** コマンド、および **newgrp** コマンドの実行。これらのコマンドを実行するのは、必ずトラステッド通信パスを確立した後にしてください。

セキュア・アテンション・キーの構成*:*

トラステッド通信パスを作成するように、セキュア・アテンション・キーを構成します。

各端末装置はそれぞれ独立して構成することができるので、その端末装置でセキュア・アテンション・キー (SAK) を押すことでトラステッド通信パスが作成されます。これは、/etc/security/login.cfg ファイルの **sak_enabled** 属性で指定します。この属性の値が True の場合、SAK が使用可能になります。

通信にポートを使用する場合 (例えば、**uucp** コマンドによって)、使用される特定のポートには、**/etc/security/login.cfg** ファイル内のスタンザに次の行があります。

```
sak_enabled = false
```

この行がある場合 (あるいはそのスタンザにエントリーがない場合)、その端末装置の SAK は使用不可になります。

端末装置で SAK を使用可能にするには、次の行をその端末のスタンザに追加します。

```
sak_enabled = true
```

Trusted Execution

Trusted Execution (TE) とは、システムの保全性を検証する場合、および拡張セキュリティー・ポリシーを実装する場合に使用される機能の集合のことです (これらのポリシーは完全なシステムのトラスト・レベルを強化する目的で共用できます)。

悪意あるユーザーがシステムを損ねる通常のやり方では、システムへのアクセス権限を入手してから、Trojans をインストールして、そこに根付かせるか、またはセキュリティーに関する重要なファイルを改ざんします。この結果、システムはぜい弱にされて悪用されます。 Trusted Execution のもとになるフィーチャーのセットの真の狙いは、上述のような活動を妨げることですが、最悪のケースとしてシステムに何かの誤動作が起こった場合でも、それを特定できます。 Trusted Execution が提供する機能を使用して、システム管理者は、実行を許可される実行可能モジュールの実際のセットを決めるか、またはロードすること

を許可されるカーネル・エクステンションのセットを決めることができます。さらに、システムのセキュリティー状態を監査したり、変更済みのファイルを識別する場合にも、この機能を使用できます。その結果、システムのトラステッド・レベルは高められ、悪意あるユーザーがシステムを損傷させることは、より困難になります。 TE のもとでのフィーチャー・セットは、以下のようにグループ分けすることができます。

- Trusted Signature Database の管理
- Trusted Signature Database の保全性の監査
- セキュリティー・ポリシーの構成
- Trusted Execution Path および Trusted Library Path

注**:** TCB 機能は既に AIX オペレーティング・システムに存在しています。 TE は TCB 機能の一部とオーバーラップして、さらに強力で拡張されたメカニズムであり、システムの保全性をより良好に制御するための拡張セキュリティー・ポリシーを提供します。 トラステッド・コンピューティング・ベースの使用可能な状態が続いていれば、 Trusted Execution により、システム保全性を検査および保護するための新しい、より拡張された概念が導入されます。

Trusted Signature Database (TSD) 管理**:**

承認コンピューティング・ベース (TCB) のデータベースに類似したデータベースが存在し、このデータベースを使用して、システム上にあるトラステッド・ファイルの重要なセキュリティー・パラメーターを保管します。 このデータベースは Trusted Signature Database (TSD) と呼ばれ、`/etc/security/tsd/tsd.dat` にあります。

トラステッド・ファイル はシステムのセキュリティー全体像から見て重要なファイルであり、これを軽視すると、システム全体のセキュリティーを危険にさらすことになります。 通常、この説明に該当するファイルは、以下のとおりです。

- カーネル(オペレーティング・システム)
- すべての setuid ルート・プログラム
- すべての setgid ルート・プログラム
- root ユーザーまたはシステム・グループのメンバーによって単独に実行されるプログラム
- トラステッド通信パスにある間は管理者によって実行しなければならないプログラム (例えば、**ls** コマンド)
- システム操作を制御する構成ファイル
- カーネルまたはシステム構成ファイルを変更できる特権またはアクセス権限で実行されるプログラム

トラステッド・ファイルには、理想的には、Trusted Signature Database (TSD) に保管されている関連スタンザ・ファイルまたはファイル定義が必要です。 ファイルには、**trustchk** コマンドを使用して、ファイルの定義を TSD に追加することにより、「トラステッド」とマークを付けることができます。 **trustchk** コマンドは TSD に対して、エントリーの追加、削除、またはリスト表示することができます。

Trusted Signature Database:

Trusted Signature Database は、システム上にあるトラステッド・ファイルの重要なセキュリティー・パラメーターを保管する場合に使用されるデータベースです。 このデータベースは `/etc/security/tsd/tsd.dat` ディレクトリーにあります。

トラステッド・ファイルには、理想的には、Trusted Signature Database (TSD) に保管されている関連スタンザ・ファイルまたはファイル定義が必要です。 それぞれのトラステッド・ファイルには、固有の暗号ハッシュとデジタル署名が関連付けられています。 トラステッド・ファイルのデフォルト・セットの暗号

ハッシュは SHA-256 アルゴリズムを使用して生成され、デジタル署名は RSA を使用して AIX ビルド環境によって生成されます。そして、AIX インストール・ファイルセットの一部としてパッケージ化されます。 これらのハッシュ値および署名は、個別の AIX インストール・イメージの一部として出荷され、以下に示すサンプル・スタンザ・フォーマットで宛先マシンの Trusted Software Database (`/etc/security/tsd/tsd.dat`) に保管されます。

```
/usr/bin/ps:
           owner          = bin
           group          = system
           mode           = 555
           type           = FILE
           hardlinks      = /usr/sbin/ps
           symlinks       =
           size           = 1024
           cert_tag       = bbe21b795c550ab243
           signature      =
f7167eb9ba3b63478793c635fc991c7e9663365b2c238411d24c2a8a
           hash_value     = c550ab2436792256b4846a8d0dc448fc45
           minslabel      = SLSL
           maxslabel      = SLSL
           intlabel       = SHTL
           accessauths    = aix.mls.pdir, aix.mls.config
           innateprivs    = PV_LEF
           proxyprivs     = PV_DAC
           authprivs      =
aix.security.cmds:PV_DAC,aix.ras.audit:PV_AU_ADMIN
           secflags       = FSF_EPS
           t_accessauths  =
           t_innateprivs  =
           t_proxyprivs   =
           t_authprivs    =
           t_secflags     =
```

owner
ファイルの所有者。 この値はファイルが TSD に追加されるときに **trustchk** コマンドを使用して計算されます。

group ファイルのグループ。 この値は **trustchk** コマンドを使用して計算されます。

mode コンマで区切られた値のリスト。 暗黙的値は **SUID** (SUID セット・ビット)、**SGID** (SGID セット・ビット)、**SVTX** (SVTX セット・ビット)、および **TCB** (トラステッド・コンピューティング・ベース) です。 ファイル許可は、最後の値でなければならず、8 進値として指定することができます。 例えば、**uid** が設定され、許可ビットで **rwxr-xr-x** を指定するファイルの場合のモード値は **SUID,755** です。 値は **trustchk** コマンドを使用して計算されます。

type ファイルのタイプ。 この値は **trustchk** コマンドを使用して計算されます。可能な値は **FILE**、**DIRECTORY**、**MPX_DEV**、**CHAR_DEV**、**BLK_DEV**、および **FIFO** です。

hardlinks
ファイルへのハードリンクのリスト。 この値は **trustchk** コマンドを使用して計算できません。したがって、ファイルをデータベースへ追加するときに、ユーザーがこの値を追加する必要があります。

symlinks
ファイルへのシンボリック・リンク先のリスト。 この値は **trustchk** コマンドを使用して計算できません。 したがって、ファイルをデータベースへ追加するときに、ユーザーがこの値を追加する必要があります。

size ファイルのサイズを定義します。 **VOLATILE** 値はファイルが変更される頻度を表します。

cert_tag
このフィールドにより、ファイルのデジタル署名がファイルの署名を検証する際に使用できる関連証明書にマップされます。このフィールドには証明書 ID が保管され、ファイルを TSD へ追加するときに **trustchk** コマンドを使用して計算されます。 証明書は `/etc/security/certificates` ディレクトリーに保管されます。

signature
ファイルのデジタル署名。 **VOLATILE** 値はファイルが変更される頻度を表します。 このフィールドは **trustchk** コマンドを使用して計算されます。

hash_value
ファイルの暗号ハッシュ。 **VOLATILE** 値はファイルが変更される頻度を表します。 このフィールドは **trustchk** コマンドを使用して計算されます。

minslabel
オブジェクトの最小機密ラベルを定義します。

maxslabel
オブジェクトの最大機密ラベルを定義します (Trusted AIX システムで有効)。 この属性は通常ファイルおよび FIFO には適用されません。

intlabel
オブジェクトの保全性ラベルを定義します (Trusted AIX システムで有効)。

accessauths
オブジェクトでのアクセス許可を定義します (Trusted AIX システムで有効)。

innateprivs
ファイルの固有の特権を定義します。

proxyprivs
ファイルのプロキシー特権を定義します。

authprivs
権限を付与されてから、ユーザーに対して割り当てられる特権を定義します。

secflags
オブジェクトに関連付けるファイルのセキュリティー・フラグを定義します。

t_accessauth
Multi-Level Security (MLS) 特定のアクセス許可を付与された追加の Trusted AIX を定義します (Trusted AIX システムで有効)。

t_innateprivs
ファイルに対して MLS 特定の固有特権を付与された追加の Trusted AIX を定義します (Trusted AIX システムで有効)。

t_proxyprivs
ファイルに対して、MLS 特定のプロキシー特権を付与された追加の Trusted AIX を定義します (Trusted AIX システムで有効)。

t_authprivs
権限を付与されてから、ユーザーに対して割り当てられた MLS 特定の特権を持つ追加の Trusted AIX を定義します (Trusted AIX システムで有効)。

t_secflags
オブジェクトと関連付けられた MLS 特定のファイル・セキュリティー・フラグの付いた追加の Trusted AIX を定義します (Trusted AIX システムで有効)。

TSD に新規エントリーを追加するときに、そのエントリーへのシンボリック・リンクまたはハード・リンクをトラステッド・ファイルに持たせる場合、これらのリンクは、コマンド・ラインから **trustchk** コマンドに **symlinks** 属性および **hardlinks** 属性を使用して TSD へ追加できます。追加されるファイルが頻繁に変更される可能性がある場合は、コマンド・ラインで VOLATILE キーワードを使用します。 そして、**trustchk** コマンドは TSD への追加のファイル定義を生成するときに、 **hash_value** フィールドおよび **signature** フィールドは計算されません。 このファイルの保全性検査では、**hash_value** フィールドおよび **signature** フィールドは無視されます。

標準のファイル定義を TSD へ追加するには、秘密鍵 (ASN.1/DER フォーマット) を用意しておく必要があります。 **-s** フラグおよびデジタル証明書を使用します。このデジタル証明書は **-v** フラグを使用するときに対応する公開鍵が付いています。 秘密鍵はファイルの署名を生成および破棄する場合に使用されます。 この鍵を安全に保管することは、ユーザーの義務として行います。 証明書は、ユーザーが保全性検査を要求するときはいつでも署名を検査できるように、 /etc/security/certificates ファイルの証明書ストアに保管されます。 ディレクトリーおよび装置ファイルのような非標準ファイルには、署名計算を行うことは不可能であり、そのようなファイルを TSD へ追加するときに、秘密鍵および証明書を用意することは必須ではありません。

事前に計算されたファイル定義については、**-f** オプションを使用して TSD に追加されているファイルから提供することも可能です。 このケースでは、**trustchk** コマンドは、いずれの値も計算しません。そして、どのような検査もなく定義を TSD に保管します。この場合、ファイル定義の健全性については、ユーザーの責任です。

ライブラリー検証のサポート

ライブラリー検証をサポートするために、/etc/security/tsd/lib/ ディレクトリーに tsd.dat ファイルが追加されます。 データベースの名前は /etc/security/tsd/lib/lib.tsd.dat です。このデータベースは、対応するトラステッド・ライブラリーの .o ファイルのスタンザを含むライブラリー専用です。ライブラリーのすべての .o ファイルのスタンザは、次の例に示すフォーマットになります。

ライブラリー libc.a では、strcmp.o ファイルが .o ファイル・タイプの 1 つである場合、/etc/security/tsd/lib/lib.tsd.dat 内の strcmp.o ファイルのスタンザは、次の例のようになります。

```
/usr/lib/libc.a/strcmp.o:
   Type = OBJ
   Size = 2345
   Hash value
   Signature =
   Cert_tag =
```

このデータベースには、.o ファイルの **type**、**size hash**、**cert tag**、および **signature** に対応するエントリーがあります。 /etc/security/tsd/tsd.dat ファイル内で、ライブラリーのハッシュが対応するスタンザに合わせて更新されます。 これらの属性値はビルドの実行中に動的に生成され、インストール中にそれらの値が /etc/security/tsd/lib/lib.tsd.dat データベースに移動されます。

/etc/security/tsd/tsd.dat ファイルでは、ライブラリーのスタンザは **type** 属性 LIB を反映するように変更され、**size** および **signature** 属性は空です。 現在、**dynamica** 属性 **size**、**hash**、**signature** の値は **VOLATILE** 値として維持されています。 したがって、システムのブート中はライブラリー検証はスキップされます。 AIX 6.1.0 リリースから、トラステッド・ライブラリー・スタンザの **size**、**hash**、および **signature** はライブラリーの .o ファイルを使用して計算されるようになりました。 インストール中、tsd.dat データベースには計算された値を反映するように取り込みが行われ、トラステッド・ライブラリー

の、対応する .o ファイルのスタンザが /etc/security/tsd/lib/lib.tsd.dat データベースに保管されます。

リモート *TE* データベース・アクセス*:*

集中管理された Trusted Signature Database (TSD) ポリシーおよび Trusted Execution (TE) ポリシーは、LDAP に保管することで、ご使用のシステム環境に実装することができます。

TSD ポリシーおよび TE ポリシーを制御するデータベースは、各システムとは独立して保管されます。AIX の集中管理された TSD ポリシーおよび TE ポリシーは LDAP に保管されるため、中央で管理することができます。 集中管理された TSD ポリシーおよび TE ポリシーを使用することで、LDAP のポリシーがマスター・コピーであるか、またクライアントが再インストール、更新、あるいはセキュリティーが侵害された場合にはいつでも、そのポリシーでクライアントを更新できるかを検査することができます。 集中管理された TE ポリシーでは、別々に各クライアントを更新する必要はなく、1 つの場所で TE ポリシーを実行できます。 集中管理された TSD ポリシーは集中管理されていない TDS ポリシーよりもはるかに管理が容易です。

ローカルの TSD ポリシー・データおよび TE ポリシー・データの LDAP へのエクスポート、TSD ポリシー・データおよび TE ポリシー・データを LDAP で使用するためのクライアントの構成、TSD ポリシー・データおよび TE ポリシー・データのルックアップの制御、およびクライアント・システムからの LDAP データの管理には、AIX ユーティリティーを使用できます。 以下のセクションでは、これらの機能に関する詳細情報を提供します。

TSD ポリシー・データおよび ***TE*** ポリシー・データの ***LDAP*** へのエクスポート*:*

TSD ポリシーおよび TE ポリシーを管理するために、LDAP を中央リポジトリーとして使用するには、LDAP サーバーにポリシー・データを追加する必要があります。

LDAP クライアントが LDAP サーバーでポリシー・データを使用できるように、 LDAP サーバーに LDAP 用の TSD ポリシーと TE ポリシーのスキーマをインストールしておく必要があります。 LDAP 用の TSD ポリシーと TE ポリシーのスキーマは、AIX システムで **/etc/security/ldap/sec.ldif** ファイルから入手可能です。 LDAP サーバー用のスキーマは、**ldapmodify** コマンドを使用して、このファイルで更新される必要があります。

LDAP サーバーの TE データベースのバージョンを識別して、LDAP クライアントがその特定バージョンを認識できるようにするには、**/etc/nscontrol.conf** ファイルに **databasename** 属性を設定する必要があります。 **databasename** 属性は値として任意の名前を付け、その名前は ldif 形式を生成する際に **tetoldif** コマンドで使用されます。

tetoldif コマンドを使用して、ローカルの TSD ポリシー・ファイルと TE ポリシー・ファイルのデータを読み取り、LDAP で使用できる形式でそのポリシーを出力してください。 **tetoldif** コマンドを使用して生成された出力結果を ldif 形式でファイルに保存し、このデータを使用して **ldapadd** コマンドで LDAP サーバーに追加します。 LDAP 用の TSD ポリシー・データおよび TE ポリシー・データを生成するには、**tetoldif** コマンドで、以下のローカル・システムのデータベースを使用します。

- /etc/security/tsd/tsd.dat
- /etc/security/tsd/tepolicies.dat

TSD ポリシーおよび ***TE*** ポリシー用の ***LDAP*** クライアント構成*:*

LDAP に保管された TSD ポリシー・データおよび TE ポリシー・データを使用するには、LDAP クライアントとしてシステムを構成する必要があります。

システムを LDAP クライアントとして構成する場合は、 AIX **/usr/sbin/mksecldap** コマンドを使用します。 **mksecldap** コマンドは指定された LDAP サーバーを動的に検索して、 TSD ポリシー・データおよび TE ポリシー・データの場所を判別し、結果を **/etc/security/ldap/ldap.cfg** ファイルに保存します。

mksecldap コマンドを使用してシステムを LDAP クライアントとして正常に構成した後に、さらに、TSD ポリシー・データおよび TE ポリシー・データのルックアップ・ドメインとして LDAP を使用可能にするように、システムをさらに構成する必要があります。これを行うには、**/etc/nscontrol.conf** ファイルの「secorder」を構成します。

いったん、システムが LDAP クライアントとして、また TSD ポリシー・データおよび TE ポリシー・データのルックアップ・ドメインとして構成されると、**/usr/sbin/secldapclntd** クライアント・デーモンは、**trustchk** コマンドが LDAP クライアントで実行されるときは必ず、LDAP サーバーから TSD ポリシー・データおよび TE ポリシー・データを取り込みます。

trustchk コマンドによる ***LDAP*** の使用可能化**:**

LDAP TSD ポリシー と TE ポリシーのデータベースを使用可能にするために、すべての TSD ポリシーと TE ポリシーのデータベース管理コマンドを使用可能にします。

–R フラグを指定して **trustchk** コマンドを使用して、LDAP データベースの初期セットアップを実行します。 初期セットアップには、TSD ポリシー、TE ポリシー、ベース DN の追加、およびローカル・データベース **/etc/security/tsd/ldap/tsd.dat** ファイルおよび **/etc/security/tsd/ldap/tepolicies.dat** ファイルの作成が含まれます。

LDAP オプションを使用して、**–R** フラグ指定で **trustchk** コマンドを実行する場合、その操作は LDAP サーバーのデータに基づきます。 ファイル・オプションを使用して、**–R** フラグ指定で **trustchk** コマンドを実行する場合、その操作はローカル・データベースのデータに基づきます。 **–R** フラグのデフォルトでは、ファイル・オプションを使用することになります。

関連情報:

mksecldap コマンド

trustchk コマンド

Trusted Signature Database の保全性の監査**:**

trustchk コマンドを使用して、Trusted Signature Database (TSD) におけるファイル定義の保全状態を実際のファイルと対比して監査することができます。

trustchk コマンドが異常を識別した場合には、自動的に修正されるか、または修正を試みる前にユーザーにプロンプトが出されます。 サイズ、シグニチャー、cert_tag または hash_value の不一致のような異常が生じた場合は、修正は不可能です。 このような場合は、**trustchk** コマンドによってファイルはアクセス不能になり、使用不能になって何らかの損傷が生じます。

さまざまな不一致の属性に対して、次の修正処置が講じられます。

owner
ファイルの所有者は TSD の中の値にリセットされます。

group ファイルのグループは TSD の中の値にリセットされます。

mode ファイルのモード・ビットは TSD の中の値にリセットされます。

hardlinks
リンクが他のファイルを指す場合は、このファイルへのリンクに変更されます。 リンクが存在しない場合は、このファイルへの新しいリンクが作成されます。

symlinks
hardlinks の場合と同様。

type ファイルはアクセス不能になります。

size **VOLATILE** ファイルの場合を除き、ファイルはアクセス不能になります。

cert_tag
ファイルはアクセス不能になります。

signature
VOLATILE ファイルの場合を除き、ファイルはアクセス不能になります。

hash_value
VOLATILE ファイルの場合を除き、ファイルはアクセス不能になります。

minslabel
Trusted AIX システムでは、最小機密ラベルが TSD の中の値にリセットされます。

maxslabel
Trusted AIX システムでは、最大機密ラベルが TSD の中の値にリセットされます。

intlabel
Trusted AIX システムでは、保全性ラベルが TSD の中の値にリセットされます。

accessauths
アクセス許可は TSD の中の値にリセットされます。 Trusted AIX では、**t_accessauths** 値が **accessauths** 属性の一部とみなされます。

innateprivs
固有の特権は TSD の中の値にリセットされます。 Trusted AIX では、**t_innateprivs** 値が **innateprivs** 属性の一部とみなされます。

inheritprivs
継承特権は TSD の中の値にリセットされます。 Trusted AIX では、**t_inheritprivs** 値が継承属性の一部とみなされます。

authprivs
許可特権は TSD の中の値にリセットされます。 Trusted AIX では、**t_authprivs** 値が **authprivs** 属性の一部とみなされます。

aecflags
セキュリティー・フラグは TSD の中の値にリセットされます。 Trusted AIX では、**t_secglags** 値が **secflags** 属性の一部とみなされます。

-F オプションを使用して、代替データベースと対比することによってファイル定義の妥当性を検査することもできます。 システム管理者は、TSD を同じシステムに保管することを避けて、いくつかの代替ロケーションにデータベースをバックアップしなければなりません。 このファイル保全性は、**-F** オプションを使用して TSD のバックアップ・バージョンと突き合わせるために生成することができます。

セキュリティー・ポリシーの構成:

Trusted Execution (TE) 機能により、実行時のファイル保全性検査メカニズムが提供されます。 このメカニズムを使用して、ファイルへのすべてのアクセス要求の前にトラステッド・ファイルの保全性を検査し、その保全性検査をパスしたトラステッド・ファイルにのみシステムへのアクセスを効率的に許可するように、システムを構成することができます。

ファイルが (その定義を Trusted Signature Database に追加することにより) トラステッドとしてマーク付けされると、TE 機能によって、すべてのアクセスに関してファイルの保全性をモニターさせることができます。 TE は継続的にシステムをモニターすることができ、実行時 (例えば、ロード時) にシステム上に存在するすべてのトラステッド・ファイルの (悪意のあるユーザーまたは不正アプリケーションによる) 改ざんを検出することができます。 ファイルの改ざんが見つかった場合は、TE は、当該ファイルの実行またはアクセスの不許可あるいはエラーのロギングのような事前構成ポリシーに基づいた修正アクションを行うことができます。 ファイルが開いているかまたは実行中である場合、および Trusted Signature Database (TSD) にファイルのエントリーがある場合は、TE は次のアクションを実行します。

- バイナリーをロードする前に、ファイルのロードを受け持つコンポーネント (システム・ローダー) が Trusted Execution サブシステムを呼び出し、SHA-256 アルゴリズム (構成可能) を使用してハッシュ値を計算します。
- 実行時に計算されたこのハッシュ値を、TSD に保管されている値と突き合わせます。
- この値が一致した場合は、ファイルのオープンまたは実行が許可されます。
- この値が一致しない場合は、バイナリーが改ざんされたか、または何らかの損傷を受けています。 実行するアクションを決定するのはユーザーです。 TE メカニズムでは、ハッシュ値が一致しない場合に実行するアクションについて、ユーザー自身のポリシーを構成できるようになっています。
- 構成したポリシーに基づいて、適切なアクションが実行されます。

以下のポリシーを構成することができます。

CHKEXEC
: トラステッド実行可能ファイルのみのハッシュ値を、ファイルが実行用にメモリーにロードされる前に検査します。

CHKSHLIBS
: トラステッド共有ライブラリーのみのハッシュ値を、ライブラリーが実行用にメモリーにロードされる前に検査します。

CHKSCRIPTS
: トラステッド・シェル・スクリプトのみのハッシュ値を、スクリプトが実行用にメモリーにロードされる前に検査します。

CHKKERNEXT
: カーネル・エクステンションのみのハッシュ値を、カーネル・エクステンションが実行用にメモリーにロードされる前に検査します。

STOP_UNTRUSTD
: 承認されていないファイルのロードを停止します。 TSD に属するファイルのみがロードされます。 このポリシーは、上記のいずれかの CHK* ポリシーとの組み合わせでのみ、機能します。 例えば、**CHKEXEC=ON** と **STOP_UNTRUSTD=ON** を組み合わせた場合は、TSD に属さない実行可能なバイナリーはすべて、ブロックされて実行されません。

STOP_ON_CHKFAIL
: ハッシュ値の検査が失敗したトラステッド・ファイルのロードを停止します。 このポリシーは、CHK* ポリシーと組み合わせた場合にも機能します。 例えば、**CHKSHLIBS=ON** と

STOP_ON_CHKFAIL=ON を組み合わせた場合は、TSD に属さない共有ライブラリーはすべてブロックされ、メモリーにロードされて使用されることはありません。

TSD_LOCK
編集に使用できないように TSD をロックします。

TSD_FILES_LOCK
トラステッド・ファイルをロックします。 これにより、書き込みモードでトラステッド・ファイルを開くことができなくなります。

TE Trusted Execution 機能を使用可能/使用不可にします。 この機能が使用可能になっている場合のみ、上記のポリシーが有効になります。

次の表では、さまざまな CHK* ポリシーおよび STOP* ポリシーが使用可能になっている場合の両者間の相互作用について説明しています。

ポリシー	**STOP_UNTRUSTD**	**STOP_ON_CHKFAIL**
CHKEXEC	TSD に属さない実行可能ファイルのロードを停止します。	ハッシュ値が TSD 値と一致しない実行可能ファイルのロードを停止します。
CHKSHLIBS	TSD に属さない共有ライブラリーのロードを停止します。	ハッシュ値が TSD 値と一致しない共有ライブラリーのロードを停止します。
CHKSCRIPTS	TSD に属さないシェル・スクリプトのロードを停止します。	ハッシュ値が TSD 値と一致しないシェル・スクリプトのロードを停止します。
CHKKERNEXT	TSD に属さないカーネル・エクステンションのロードを停止します。	ハッシュ値が TSD 値と一致しないカーネル・エクステンションのロードを停止します。

注**:** ポリシーを有効にするために TE がオンにされるまでは、ポリシーをいつでも使用可能または使用不可にすることができます。 ポリシーが有効にされると、そのポリシーを使用不可にしても、次のブート・サイクルまで効力をもちません。 すべての情報メッセージは、**syslog** に記録されます。

関連情報:

TE_verify_reg カーネル・サービス

TE_verify_unreg カーネル・サービス

Trusted Execution Path および ***Trusted Library Path:***

Trusted Execution Path (TEP) には、トラステッド実行可能モジュールが配置されているディレクトリー・リストを定義します。 いったん TEP 検証が使用可能になれば、システム・ローダーにより実行可能になるのは、指定パスにあるバイナリーだけです。 Trusted Library Path (TLP) は、システムのトラステッド・ライブラリーのあるディレクトリーを定義する場合に使用されることを除いて、同じ機能をもっています。

いったん TLP 検証が使用可能になれば、システム・ローダーによりバイナリーへのリンクが可能になるのは、このパスで指定されたライブラリーだけです。 **trustchk** コマンドは、**trustchk** コマンドの TEP および TLP コマンド・ライン属性を使用して、両方のパス・リストをコロン区切りで設定する場合と同様に、TEP または TLP を使用可能にしたり、使用不可にしたりする場合に使用します。

トラステッド・シェルおよびセキュア・アテンション・キー*:*

トラステッド・シェルおよびセキュア・アテンション・キー (SAK) は、トラステッド・コンピューティング・ベース (TCB) と同じように実行しますが、 Trusted Execution が TCB に代わってシステムで使用可能になっている場合、トラステッド・シェルは Trusted Signature Database のみに属しているファイルを実行することが異なります。

TCB および SAK の詳細情報については、『トラステッド・コンピューティング・ベース』、『セキュア・アテンション・キーの使用』、および『セキュア・アテンション・キーの構成』を参照してください。

トラステッド実行 ***(TE)*** ポリシー・データベース**:**

トラステッド実行 (TE) ポリシーは **/etc/security/tsd/tepolicies.dat** ファイルに保管されます。 TE ポリシーのパスは TLP ディレクトリーおよび TEP ディレクトリーとともにリストされます。

セキュリティー・プロファイルの評価確認レベル 4+ と Labeled AIX Security および評価確認レベル 4+

システム管理者は、基本オペレーティング・システム (BOS) のインストール時に、 Base AIX Security (BAS) および評価保証レベル 4+ (EAL4+) オプションまたは Labeled AIX Security (LAS) および評価確認レベル 4+ (EAL4+) を指定してシステムをインストールできます。これらのオプションを備えたシステムには、BOS インストール時にインストールされるソフトウェアに対して制限があります。さらに、ネットワーク・アクセスも制限されます。

注**:** AIX バージョン 7.1 での評価は、現在も継続して行われています。 最新の情報については、「AIX バージョン 7.1 リリース情報」を参照してください。

セキュリティー・プロファイルの概要**:**

セキュリティー・プロファイルは、ネットワーク環境における汎用オペレーティング・システムのセキュリティー要件の仕様を定める製品です。 このプロファイルは、評価対象 (TOE) セキュリティー機能とその環境のセキュリティー目標を達成するために必要な要件を設定します。

セキュリティー・プロファイルには基本パッケージといくつかの拡張パッケージが含まれています。セキュリティー・プロファイル基本パッケージのサポートに関連する製品は、識別および認証、任意アクセス制御 (DAC)、監査、暗号サービス、セキュリティー・メカニズムの管理、およびトラステッド・チャネル通信です。 セキュリティー・プロファイルには追加のオプション・パッケージが含まれており、ラベル付きセキュリティー、保全性検査、拡張監査、汎用暗号化、拡張管理、拡張された識別と認証、トラステッド・ブート、および仮想化に対応しています。

前提事項

- TOE のための使用環境:

  このセクションの前提事項はすべて、特別な断りがない限り、Base AIX Security (BAS モード) および Labeled AIX Security (LAS モード) を示しています。 仮想入出力サーバー (VIOS) に関連するすべての前提事項には、「VIOS のみ」と明記されています。 VIOS の前提事項は、AIX オペレーティング・システムおよび Trusted AIX の前提事項と共通していません。
- 物理的事項:

  IT 環境は、TOE によって保護されている IT 資産の価値に相応する適切な物理的セキュリティーを TOE に提供します。

  注**:** VIOS のみ: 稼働環境は、TOE によって保護されている IT 資産の価値に相応する適切な物理的セキュリティーを TOE に提供します。
- 管理:
  - TOE セキュリティー機能は、1 人以上の有能な個人によって管理されます。 システム管理担当者は、注意深く、故意に怠慢を行うことのない友好的な人物であり、ガイダンス資料に記載された指示を順守します。

- 許可ユーザーは TOE が管理する一部の情報にアクセスでき、協力的な態度で行動することが期待されています。
- ユーザーは、セキュアな IT 環境内の一部のタスクまたはタスク・グループを遂行するために、十分な訓練を受け、信頼されています。 ユーザーは、そのユーザー・データを完ぺきに制御できる必要があります。
- VIOS のみ: TOE セキュリティー機能は、1 人以上の有能な個人によって管理されます。 システム管理担当者は、注意深く、故意に怠慢を行うことのない友好的な人物であり、ガイダンス資料に記載された指示を順守します。
- VIOS のみ: 許可ユーザーは TOE が管理する情報の少なくとも一部にアクセスするために必要な権限を保有し、協力的な態度で行動することが期待されています。
- VIOS のみ: ユーザーは、セキュアな稼働環境内の一部のタスクまたはタスク・グループを遂行するために、十分な訓練を受け、信頼されています。ユーザーは、そのユーザー・データを完ぺきに制御できる必要があります。

- 手順:
  - ユーザーまたは基礎のプラットフォームが意図的あるいは偶発的に引き起こした TOE のセキュリティー実施/セキュリティー関連ファイルの変更または破損は、管理ユーザーが検出する必要があります。
  - ターゲット・セキュリティー機能 (TSF) によって信頼されたリモート・トラステッド IT システムは、すべて、TOE に TSF データまたは TSF サービスを提供するため、またはセキュリティー・ポリシー決定事項の適用にあたって TSF をサポートするために、同じ管理制御のもとにあること、および TOE のセキュリティー・ポリシーに準拠するセキュリティー・ポリシー制約のもとで稼働することが想定されています。
  - TSF によって信頼されたリモート・トラステッド IT システムは、すべて、TOE に TSF データまたは TSF サービスを提供するため、またはセキュリティー・ポリシー決定事項の適用にあたって TSF をサポートするために、TSF によって使用されている機能 (その機能に定義された前提事項に整合しているもの) を正しく実装することが想定されています。
  - 以下の情報の保全性が確保されています。
    - すべての TSF コード (保全性検査メカニズムの開始前にロードおよび実行される保全性検査機能を含む)
    - すべての TSF データ (保全性検査メカニズムの開始前にロードおよび実行される TSF コードが使用する保全性検査を実行するための TSF データを含む)
  - VIOS のみ: ユーザーまたは基礎のプラットフォームが意図的あるいは偶発的に引き起こした TOE のセキュリティー実施/セキュリティー関連ファイルの変更または破損は、管理ユーザーが検出する必要があります。
- 接続: リモート・トラステッド IT システムとの間の接続、および TSF 自体によって保護されていない TSF の物理的に分離された各部品間の接続はすべて、TOE 環境内で物理的または論理的に保護されるため、伝送されるデータの保全性と機密性が確保され、通信エンドポイントの認証性も確保されます。

ソフトウェアの入手

ソフトウェアを取得するには、以下のステップを実行します。

1. 製品をダウンロードします。
2. 左側のペインにある「Entitled software support (ライセンス済みソフトウェア・サポート)」メニューの「Help (ヘルプ)」をクリックします。 共通基準評価構成では、製品およびすべての更新を物理メディアまたは Download Director から取得することが求められます。

製品のインストールについては、BAS/EAL4+ システムのインストールを参照してください。

BAS/EAL4+ システムのインストール**:**

RBAC は、このオプションが選択された場合、自動的に使用可能になります。

BOS のインストール時に BAS/EAL4+ オプションを設定するには、次のようにします。

1. 「Installation and Settings (インストールおよび設定)」画面で 「**More Options (**他のオプション**)**」を選択します。
2. 「More Options (他のオプション)」の下で、BAS/EAL4+ オプションに「**Yes (**はい**)**」を選択し、WPAR を使用している場合は TCB オプションに「**No (**いいえ**)**」を選択します。 カスタマイズ済みの `bosinst.data` ファイルを使用してプロンプトなしのインストールを実行する場合、TCB オプションを「**Yes (**はい**)**」に設定できます。

BAS のインストールのリモート root ログインを使用不可にします。リモート root ログインを使用不可にするには、インストール後に次のコマンドを実行します。

```
/usr/bin/chuser rlogin=false subgroups=SUADMIN root
```

管理ユーザーを **SUADMIN** グループに追加して、それらのユーザーが root になるための **su** を実行できるようにします。

「**Enable BAS and EAL4+ Technology (BAS** および **EAL4+** テクノロジーを使用可能にする**)**」オプションは、以下の条件でのみ使用可能です。

- インストール方式が新規および完全上書きインストールに設定されている。
- 英語が選択されている。
- 64 ビット・カーネルが使用可能にされている。
- 拡張ジャーナル・ファイルシステム (JFS2) が使用可能にされている。

「**Enable BAS and EAL4+ Technology (BAS** および **EAL4+** テクノロジーを使用可能にする**)**」オプションを「Yes (はい)」に設定すると、「**Trusted Computing Base (**トラステッド・コンピューティング・ベース**)**」オプションも「Yes (はい)」に設定され、有効な「**Desktop (**デスクトップ**)**」選択項目は「NONE (なし)」または「CDE」のみになります。

カスタマイズ済みの `bosinst.data` ファイルを使用してプロンプトなしのインストールを実行する場合、**INSTALL_TYPE** フィールドは `CC_EVAL` に設定する必要があり、以下のフィールドも次に示すとおりに設定する必要があります。

```
control_flow:
 CONSOLE = ???
 PROMPT = yes
 INSTALL_TYPE = CC_EVAL
 INSTALL_METHOD = overwrite
 TCB = yes
 DESKTOP = NONE or CDE
 ENABLE_64BIT_KERNEL = yes
 CREATE_JFS2_FS = yes
 ALL_DEVICES_KERNELS = no
 FIREFOX_BUNDLE = no
 HTTP_SERVER_BUNDLE = no
 KERBEROS_5_BUNDLE = no
 SERVER_BUNDLE = no
 ALT_DISK_INSTALL_BUNDLE = no
```

```
locale:
 CULTURAL_CONVENTION = en_US or C
 MESSAGES = en_US or C
```

RBAC について詳しくは、『ロール・ベース・アクセス制御 (RBAC)』を参照してください。

BAS/EAL4+ のネットワーク・インストール管理環境**:**

BAS/EAL4+ テクノロジー・クライアントのインストールは、ネットワーク・インストール管理 (NIM) 環境を使用して実行します。

NIM マスターは、適切な BAS/EAL4+レベルの AIX 7.1 をインストールするために必要なリソースを提供するように構成されています。 よって、NIM クライアントは、NIM マスターにあるリソースを使用してインストールできます。 **bosinst_data** リソース内の以下のフィールドを設定することで、プロンプトなしの NIM クライアントのインストールを行うことができます。

```
control_flow:
 CONSOLE = ???
 PROMPT = no
 INSTALL_TYPE = CC_EVAL
 INSTALL_METHOD = overwrite
 TCB = yes
 DESKTOP = NONE or CDE
 ENABLE_64BIT_KERNEL = yes
 CREATE_JFS2_FS = yes
 ALL_DEVICES_KERNELS = no
 FIREFOX_BUNDLE = no
 HTTP_SERVER_BUNDLE = no
 KERBEROS_5_BUNDLE = no
 SERVER_BUNDLE = no
 ALT_DISK_INSTALL_BUNDLE = no

locale:
 CULTURAL_CONVENTION = en_US or C
 MESSAGES = en_US or C
```

NIM マスターは、BAS/EAL4+ システムとして構成することはできません。また、他の BAS/EAL4+ システムと同じネットワークに接続することはできません。 NIM マスターからインストールを開始する場合、「**SMIT** インストール後、**NIM** クライアントを残す」メニュー・オプションを「No」に設定する必要があります。 NIM クライアントを BAS/EAL4+ システムとしてインストールした後に、その NIM クライアントを NIM マスターのネットワークから除去する必要があります。これを行わないと、NIM マスターを使用して追加ソフトウェアのインストールおよび更新を行うことはできません。

この状態の例として、2 つのネットワーク環境がある場合があります。最初のネットワークは NIM マスターと非 BAS/EAL4+ システムで構成され、2 番目のネットワークは BAS/EAL4+システムのみで構成されているとします。 この場合、NIM クライアントで NIM インストールを実行します。 インストールが完了したら、新しくインストールされた BAS/EAL4+ システムを NIM マスターのネットワークから切断し、評価済みネットワークに接続します。

もう 1 つの例は、1 つのネットワークから構成される場合です。 他のシステムが評価済み構成で稼働している場合、NIM マスターはネットワークに接続されません。また、NIM インストール中は BAS/EAL4+ システムはネットワークに接続されません。

BAS/EAL4+ソフトウェア・バンドル**:**

BAS/EAL4+ オプションが選択されている場合、/usr/sys/inst.data/sys_bundles/CC_EVAL.BOS.autoi インストール・バンドルの中身がインストールされます。

BAS/EAL4+ オプションが選択されていれば、オプションで、グラフィックスのソフトウェア・バンドルおよび文書サービスのソフトウェア・バンドルのインストールを選択できます。 **BAS/EAL4+** オプションを指定して「**Graphics Software (**グラフィックス・ソフトウェア**)**」オプションを選択すると、`/usr/sys/inst.data/sys_bundles/CC_EVAL.Graphics.bnd` ソフトウェア・バンドルの内容がインストールされます。 **BAS/EAL4+** オプションを指定して「Documentation Services Software (文書サービス・ソフトウェア)」オプションを選択すると、`/usr/sys/inst.data/sys_bundles/CC_EVAL.DocServices.bnd` ソフトウェア・バンドルの内容がインストールされます。

ライセンス・プログラム・プロダクト (LPP) がインストールされた後で、システムは、BAS/EAL4+ 要件に準拠するようにデフォルトの構成を変更します。 デフォルトの構成に対して以下の変更が加えられます。

- **/etc/pse.conf** ファイルから `/dev/echo` が除去される。
- ストリーム・デバイスをインスタンス化する。
- root だけが取り外し可能メディアにアクセスできる。
- **inetd.conf** ファイルから CC 以外の項目を除去する。
- さまざまなファイル・アクセス権を変更する。
- **sysck.cfg** ファイルにシンボリック・リンクを登録する。
- **sysck.cfg** ファイルにデバイスを登録する。
- デフォルトのユーザーおよびポート属性を設定する。
- doc_search アプリケーションをブラウザーでの使用のために構成する。
- **inittab** ファイルから httpdlite を除去する。
- **inittab** ファイルから writesrv を除去する。
- **inittab** ファイルから mkatmpvc を除去する。
- **inittab** ファイルから atmsvcd を除去する。
- **/etc/rc.tcpip** ファイルの snmpd を使用不可にする。
- **/etc/rc.tcpip** ファイルの hostmibd を使用不可にする。
- **/etc/rc.tcpip** ファイルの snmpmibd を使用不可にする。
- **/etc/rc.tcpip** ファイルの aixmibd を使用不可にする。
- **/etc/rc.tcpip** ファイルの muxatmd を使用不可にする。
- NFS ポート (2049) は特権のあるポートである。
- 欠落したイベントを **/etc/security/audit/events** ファイルに追加する。
- ループバック・インターフェースが実行していることを確認する。
- **/dev/console** の同義語を作成する。
- デフォルトの X サーバー接続アクセス権を施行する。
- **/var/docsearch** ディレクトリーを、すべてのファイルが一般読み取り可能になるように変更する。
- オブジェクト・データ・マネージャー (ODM) スタンザを追加して、コンソール・アクセス権を設定する。
- BSD スタイルの ptys でのアクセス権を 000 に設定する。
- **.netrc** ファイルを使用不可にする。
- パッチ・ディレクトリー処理を追加する。

グラフィカル・ユーザー・インターフェース:

BAS/EAL4+ 準拠のシステムには、グラフィカル・ユーザー・インターフェースとして X Windows System が搭載されています。

X Windows は、**aixterm** コマンドを使用した複数端末セッションを表示するメカニズムだけでなく、クロック、電卓、およびその他のグラフィカル・アプリケーションなどのグラフィカル・クライアントを表示するメカニズムも提供します。 X Windows System は、ユーザーがホストのコンソールでログインした後、初期コマンド・ラインから **xinit** コマンドを使用して開始されます。

X Windows セッションを開始するには、次のように入力します。

```
xinit
```

このコマンドは、ローカル・アクセス・メカニズムが呼び出し側のみに使用可能になった状態で X Windows サーバーを始動します。 UID が root に設定された X Windows クライアントは、アクセス制限で root オーバーライドを使用して UNIX ドメイン・ソケットを介して X Windows サーバーにアクセスできます。 UID が他のユーザーに設定されているか、他のユーザーによって開始された X Windows クライアントは、X Windows サーバーにアクセスできません。 この制限により、ホストの他のユーザーが X Windows サーバーに無許可アクセスを獲得できないようにします。

LAS/EAL4+ システムのインストール:

RBAC は、このオプションが選択された場合、自動的に使用可能になります。

BOS のインストール時に LAS/EAL4+ オプションを設定するには、次のようにします。

インストールのオプションは、「Installation and Settings (インストールおよび設定)」ウィンドウで、 3 をタイプ入力して「**Security Model (**セキュリティー・モデル**)**」を変更し、 4 をタイプ入力して「**More Options (**追加オプション**)**」フィールドを表示します。これらのオプションはインストール・タイプ (上書き、保存、またはマイグレーション) およびセキュリティー・オプションによって変わります。 LAS の場合、インストール方式は新規または完全な上書きです。 「**LAS/EAL4+** 構成のインストール」を選択します。

RBAC について詳しくは、『ロール・ベース・アクセス制御 (RBAC)』を参照してください。

LAS/EAL4+ 構成インストール **(Trusted AIX** でのみ使用可能**):**

「**LAS/EAL4+** 構成インストール」オプションにより、 Trusted AIX は LAS/EAL4+ 構成モードでインストールされます。 LAS/EAL4+ 構成モードは、Trusted AIX インストールに比べて、さらに厳しく制約されたセキュリティー用として提供されます。

カスタマイズ済みの `bosinst.data` ファイルを使用してプロンプトなしのインストールを実行する場合、**INSTALL_TYPE** フィールドはブランクにし、**TRUSTED_AIX** フィールドは `yes` に設定して、以下のフィールドは次に示すとおりに設定する必要があります。

```
control_flow:
 CONSOLE = ???
 PROMPT = yes
 INSTALL_TYPE =
 TRUSTED_AIX = yes
 INSTALL_METHOD = overwrite
 TCB = yes
 DESKTOP = NONE
 ENABLE_64BIT_KERNEL = yes
 CREATE_JFS2_FS = yes
```

```
 ALL_DEVICES_KERNELS = no
 FIREFOX_BUNDLE = no
 HTTP_SERVER_BUNDLE = no
 KERBEROS_5_BUNDLE = no
 SERVER_BUNDLE = no
 ALT_DISK_INSTALL_BUNDLE = no

locale:
 CULTURAL_CONVENTION = en_US or C
 MESSAGES = en_US or C
```

Trusted AIX の詳細については、「Trusted AIX」を参照してください。

LAS/EAL4+ のネットワーク・インストール管理環境**:**

LAS/EAL4+ テクノロジー・クライアントのインストールは、ネットワーク・インストール管理 (NIM) 環境を使用して実行します。

NIM マスターは、適切な LAS/EAL4+レベルの AIX 7.1 をインストールするために必要なリソースを提供するように構成されています。 よって、NIM クライアントは、NIM マスターにあるリソースを使用してインストールできます。 bosinst_data リソース内の以下のフィールドを設定することで、プロンプトなしの NIM クライアントのインストールを行うことができます。

```
control_flow:
 CONSOLE = ???
 PROMPT = no
 INSTALL_TYPE =
 TRUSTED_AIX = yes
 INSTALL_METHOD = overwrite
 TCB = yes
 DESKTOP = NONE
 ENABLE_64BIT_KERNEL = yes
 CREATE_JFS2_FS = yes
 ALL_DEVICES_KERNELS = no
 FIREFOX_BUNDLE = no
 HTTP_SERVER_BUNDLE = no
 KERBEROS_5_BUNDLE = no
 SERVER_BUNDLE = no
 ALT_DISK_INSTALL_BUNDLE = no

locale:
 CULTURAL_CONVENTION = en_US or C
 MESSAGES = en_US or C
```

NIM マスターは、LAS/EAL4+ システムとして構成することはできません。また、他の LAS/EAL4+ システムと同じネットワークに接続することはできません。 NIM マスターからインストールを開始する場合、「**SMIT** インストール後、**NIM** クライアントを残す」メニュー・オプションを「No」に設定する必要があります。 NIM クライアントを LAS/EAL4+ システムとしてインストールした後に、その NIM クライアントを NIM マスターのネットワークから除去する必要があります。これを行わないと、NIM マスターを使用して追加ソフトウェアのインストールおよび更新を行うことはできません。

この状態の例として、2 つのネットワーク環境がある場合があります。最初のネットワークは NIM マスターと非 LAS/EAL4+ システムで構成され、2 番目のネットワークは LAS/EAL4+システムのみで構成されているとします。 この場合、NIM クライアントで NIM インストールを実行します。 インストールが完了したら、新しくインストールされた LAS/EAL4+ システムを NIM マスターのネットワークから切断し、評価済みネットワークに接続します。

もう 1 つの例は、1 つのネットワークから構成される場合です。 他のシステムが評価済み構成で稼働している場合、NIM マスターはネットワークに接続されません。また、NIM インストール中は LAS/EAL4+ システムはネットワークに接続されません。

BAS/EAL4+ および **LAS/EAL4+** システムの物理的環境**:**

BAS/EAL4+ および LAS/EAL4+ システムには、実行される環境について特定の要件があります。

それらの要件は以下のとおりです。

- 物理アクセスは、許可された管理者だけがシステム・コンソールを使用できるように制限される必要がある。
- サービス・プロセッサーはモデムに接続されない。
- 端末装置への物理アクセスは許可されたユーザーに制限される。
- 物理ネットワークが盗聴やスプーフィング・プログラム (トロイの木馬とも呼ばれる) に対して機密保護機能を持つ。 無保護回線を介して通信を行う場合、追加のセキュリティー手段 (暗号化など) が必要とされます。
- AIX 7.1 BAS/EAL4+ または LAS/EAL4+ システムとは異なる他のシステム、または同じ管理制御下にない他のシステムとの通信は許可されない。
- 他の BAS/EAL4+ および LAS/EAL4+ システムと通信するときには、IPv4 のみが使用される。 IPv6 は評価済み構成に含まれますが、IPv4 によってもサポートされる IPv6 の機能のみが含まれます。
- システム時刻の変更がユーザーに許可されてはならない。
- LPAR 環境のシステムは PHB を共用できない。

BAS/EAL4+ および **LAS/EAL4+** システム組織上の環境**:**

BAS/EAL4+ および LAS/EAL4+ システムについて、次のような手続き上および組織上の要件が満たされている必要があります。

以下の要件が満たされている必要があります。

- 管理者は信頼できる人で、よく訓練されていなければならない。
- システム上の情報を使って作業することを許可されたユーザーだけが、システム上でユーザー ID を付与される。
- ユーザーは高品質なパスワードを使用する必要がある (可能な限りランダムで、特定のユーザーまたは組織に関係していない)。 パスワード・ルールの設定の詳細については、 72 ページの『パスワード』を参照してください。
- ユーザーは他人に自分のパスワードを開示してはならない。
- 管理者はセキュリティー・クリティカルなシステムを管理するための十分な知識を持っていなければならない。
- 管理者はシステム文書で提供されるガイダンスに従って作業する必要がある。
- 管理者は自分のパーソナル ID を使ってログインし、**su** コマンドを使用して管理用のスーパーユーザー・モードに切り替える必要がある。
- 管理者によってシステム・ユーザー用に生成されたパスワードは、ユーザーに安全に送信される必要がある。
- システムの担当者は、システムの安全な操作に必要な手順を確立し、実装する必要がある。
- 管理者は、セキュリティー・クリティカルなシステム・リソースへのアクセスが、許可ビットおよび ACL の適切な設定によって保護されていることを確認する必要がある。

- 物理ネットワークは、システムが保持している最も機密性の高いデータを搬送することを、組織によって承認される必要がある。
- 保守手順にはシステムの定期診断が含まれていなければならない。
- 管理者は、システム障害後の安全な操作およびリカバリーを確保する手順を用意している必要がある。
- *LIBPATH* 環境変数は変更してはならない。なぜなら、これはトラステッド・プロセスが非トラステッド・ライブラリーをロードすることになるからです。
- 盗聴およびトレース・ソフトウェア (tcpdump、trace) を、運用システムで使用してはならない。
- 匿名プロトコル (HTTP など) は公開情報 (例えば、オンライン文書) 用にのみ使用される。
- TCP ベースの NFS のみが使用できる。
- 取り外し可能メディアへのアクセスはユーザーに許可されていない。 装置ファイルは適切な許可ビットまたは ACL によって保護されます。
- 管理者は、動的パーティショニングを利用して、リソースの割り当て/割り当て解除を行ってはならない。 パーティション構成は、すべてのパーティションが実行中でないときにのみ行うことができます。

BAS/EAL4+ および **LAS/EAL4+** システムの稼働環境**:**

BAS/EAL4+および LAS/EAL4+システムについて、稼働上の要件および手順が満たされている必要があります。

以下の要件および手続きが満たされている必要があります。
- ハードウェア管理コンソール (HMC) を使用している場合、HMC は物理的に制御された環境に配置されている。
- 許可された担当者のみが稼働環境および HMC にアクセスできる。
- HMC を使用している場合、HMC は以下の作業にのみ使用できる。
  - パーティションの初期構成。 構成処理中、パーティションをアクティブにすることはできません。
  - 「停止中」のパーティションの再始動
- 構成されたシステムの稼働中 HMC を使用してはならない。
- システムの「コール・ホーム」機能を使用不可にしておかなければならない。
- システムへのリモート・モデム・アクセスを使用不可にしておかなければならない。
- AIX を LPAR 使用可能環境で稼働させる場合、管理者は、LPAR の文書で LPAR の EAL4+ 操作に関する要件を確認する必要がある。
- LPAR に対して、サービス権限機能を使用不可にしなくてはならない。

BAS/EAL4+ システムの構成**:**

Base AIX Security (BAS) および評価保証レベル 4+ (EAL4+) システムを構成することができます。

system、**sys**、**adm**、**uucp**、**mail**、**security**、**cron**、**printq**、**audit**、および **shutdown** グループは、管理グループと考えられます。 これらのグループには、トラステッド・ユーザーのみを追加します。

管理*:*

管理者は自分のパーソナル・ユーザー・アカウントを使ってログインし、**su** コマンドを使用してシステムの管理を担当する root ユーザーになる必要があります。

root アカウントのパスワードを推測されない効果的な方法は、許可された管理者だけが root アカウント上の **su** コマンドを使用できるようにすることです。 これを確実にするには、以下のようにします。

1. **/etc/security/user** ファイルの **root** スタンザに、以下のように項目を追加します。

```
root:
  admin = true
  .
  .
  .
  sugroups = SUADMIN
```

2. 許可された管理者のユーザー ID だけを含む **/etc/group** ファイルのグループを以下のように定義します。

```
system:!:0:root,paul
staff:!:1:invscout,julie
bin:!:2:root,bin
.
.
.
SUADMIN:!:13:paul
```

管理者は以下の手順も守らなければなりません。

- 分散システムを構成するハードウェア、ソフトウェア、およびファームウェアのコンポーネントが確実に安全な方法で分散、インストール、および構成されるための手順を確立し、実装する。
- システムが構成される際に、管理者だけが新規の信頼ソフトウェアをシステムに導入できるようにする。
- ユーザーがシリアル・ログイン・デバイス (例えば、IBM® 3151 端末装置) をログオフする前に、必ず画面を消去するための手順を実装する。

ユーザーおよびポートの構成*:*

ユーザーおよびポートに関する AIX 構成オプションは、評価の要件を満たすように設定する必要があります。 実際の要件としては、パスワードを正確に推測するという、メトリック品質を満たすメカニズムを TSF が提供するということです。 パスワードの存続時間中にアタッカーが達成できるパスワードを正確に推測する確率は、2^-20 より小さい値でなければなりません。

次の例に示す **/etc/security/user** ファイルでは、**/usr/share/dict/words** ディクショナリー・リストが使用されています。 **/usr/share/dict/words** ファイルは **bos.data** ファイルセットに含まれています。 **bos.data** ファイルセットは、**/etc/security/user** ファイルを構成する前にインストールしておく必要があります。 **/etc/security/user** ファイルの推奨値は次のとおりです。

```
default:
  admin = false
  login = true
  su = true
  daemon = true
  rlogin = true
  sugroups = ALL
  admgroups =
  ttys = ALL
  auth1 = SYSTEM
  auth2 = NONE
  tpath = nosak
  umask = 077
  expires = 0
  SYSTEM = "compat"
  logintimes =
  pwdwarntime = 5
  account_locked = false
  loginretries = 3
  histexpire = 52
  histsize = 20
```

```
  minage = 0
  maxage = 8
  maxexpired = 1
  minalpha = 2
  minother = 2
  minlen = 8
  mindiff = 4
  maxrepeats = 2
  dictionlist = /usr/share/dict/words
  pwdchecks =
  dce_export = false

root:
  rlogin = false
  login = false
```

/etc/security/user ファイル内のデフォルトの設定値は、単一ユーザーの特定の設定値で上書きしてはなりません。

注**:** root スタンザに `login = false` を設定すると、直接 root ログインができなくなります。 root アカウントの **su** 特権を持つユーザー・アカウントだけが、root アカウントとしてログインできることになります。 システムに対してサービス妨害攻撃が開始され、間違ったパスワードがユーザー・アカウントに送信される場合、システムはすべてのユーザー・アカウントをロックします。 この攻撃により、すべてのユーザー (管理ユーザーを含む) がシステムにログインできなくなる可能性があります。 ユーザーのアカウントがロックされると、そのユーザーは、システム管理者が **/etc/security/lastlog** ファイルにあるそのユーザーの `unsuccessful_login_count` 属性を `loginretries` ユーザー属性の値より小さい値に再設定するまで、ログインできません。すべての管理アカウントがロックされた場合は、システムを保守モードでリブートしてから、**chsec** コマンドを実行する必要があります。 **chsec** コマンドの使用方法についての詳細は、59 ページの『ユーザー・アカウント制御』を参照してください。

/etc/security/login.cfg ファイルの推奨値は次のとおりです。

```
default:
  sak_enabled = false
  logintimes =
  logindisable = 4
  logininterval = 60
  loginreenable = 30
  logindelay = 5
```

setuid/setgid プログラムのリスト**:**

BAS の使用が可能な AIX システムに対して作成されるトラステッド・アプリケーションのリスト。

root またはトラステッド・グループが所有するすべての非トラステッド・プログラムでは、**suid/sgid** ビットがオフになります。 BAS のインストール後、**suid** で root が所有するか、あるいは **sgid** でこれらのトラステッド・グループの 1 つが所有する、システム上のプログラムは、**system**、**sys**、**adm**、**uucp**、**mail**、**security**、**cron**、**printq**、**audit**、および **shutdown** のみです。 これらのグループにはトラステッド・ユーザーのみを追加します。

トラステッド・アプリケーションのリストは、次のカテゴリーの少なくとも 1 つに分類されるすべてのアプリケーションを考慮することによって作成されます。

- 対応するアプリケーションの SUID ルート・ビットが使用可能になっている。
- トラステッド・グループの 1 つに対する SGID ビットが使用可能になっている。
- 管理者のガイダンスの資料に従っていずれかのトラステッド・データベースにアクセスするアプリケーション。

注**: ipcs** コマンドの **setuid** ビットは、システム管理者が除去する必要があります。 システム管理者は、**chmod u-s /usr/bin/ipcs** および **chmod u-s /usr/bin/ipcs64** コマンドを実行する必要があります。

監査ファイル・システムの変更*:*

RBAC は、このオプションが選択された場合、自動的に使用可能になります。

/audit ファイル・システムは jfs ファイル・システムです。 jfs2 ファイル・システムへの変更が必要です。 さらに、BAS システムでは追加のコマンドを実行する必要があります。 このファイル・システムに変更を行うには、以下のステップを実行します。

1. BAS システムのファイル・システムを変更するには、次のコマンドを入力します。

   ```
   audit shutdown
   lsvg -l rootvg
   ```

   LAS システムの場合は、ステップ 3 に進みます。
2. TYPE フィールドに疑問符 (?) 記号が含まれる場合、次のコマンドを入力します。

   ```
   synclvodm -v rootvg
   ```
3. 次のコマンドを入力して、jfs ファイル・システムを削除し、jfs2 ファイル・システムを作成します。

   ```
   umount/audit
   rmfs /audit
   crfs -v jfs2 -m /audit -g rootvg -A yes -p rw -a size=100M
   ```

Trusted Signature Database (TSD) の更新*:*

このセクションでは、TSD を更新する手順を説明します。

BAS/LAS 構成でシステム・モード・ビットを変更すると、TSD の整合性エラーが発生します。

システムのリブート時は、「すべて無視 **(Ignore All)**」オプションを選択します。

TSD を更新するには、次のコマンドを入力します。

```
trustchk -u ALL mode
```

LAS システムの使用*:*

このセクションでは、LAS システムの使用ガイドラインを示します。

システムを isso としてインストールしたら、次のコマンドを入力して自動リブート・オプションを **false** に設定します。

```
chdev -l sys0 -a autorestart=false
```

TSD が引き続き intlabel エラーを生成する場合は、次のコマンドを入力し、**PV_ROOT** 特権がある isso を使用してエラーを削除してください。

```
cp /etc/security/tsd/tsd.dat /etc/security/tsd/tsd.dat.org
trustchk -q /usr/sbin/format /usr/sbin/fdformat /usr/sbin/mount /usr/sbin/unmount ¥
/usr/sbin/umount /usr/sbin/tsm /usr/sbin/getty /usr/sbin/login /usr/sbin/mkvg ¥
/usr/sbin/extendvg /usr/bin/w /usr/bin/uptime >/tmp/list.dat
grep -p SLTL /tmp/list.dat |sed 's/SLTL/SHTL/' >/tmp/new.dat
trustchk -w -a -f /tmp/new.dat
trustchk -y ALL
```

監査に関連するエラー・メッセージがコンソールに表示される場合、isso 特権を使用し、次のコマンドを入力して監査システムを再始動してください。

```
# audit shutdown
# audit start
```

ログイン試行に 3 回失敗すると、isso/so ログインはネットワークによってブロックされます。 ただし、管理者は引き続きローカル・コンソールでこれらのアカウントにアクセスできます。

cron/at によって実行されるコマンドの出力は、ユーザーのメール・スプールに転送されません。

ラベル範囲のある World-writeable ディレクトリー (/tmp など) は区画に分割されません。 ラベル間で情報が流れる可能性を防ぐために、管理者は初期構成後、直ちにこれらのディレクトリーを区画に分割する必要があります。

ネットワーク・インターフェース*:*

このセクションでは、ネットワーク・インターフェースの使用手順について説明します。

Trusted AIX では、デフォルトのネットワーク・インターフェースのラベル範囲は minSL=impl_lo から maxSL=ts_all です。 LAS/EAL4+ システムにはラベル範囲はありません。 LAS/EAL4+ インストール・オプションが選択されると、デフォルト・ルールは自動的に impl_lo に変更されます。 デフォルト・ルールを isso として変更するには、**netrule** コマンドを使用します。

次に例を示します。

```
/usr/sbin/netrule i+u default +impl_lo +impl_lo +impl_lo
```

WPAR の更新*:*

このセクションでは、AIX のワークロード区画 (WPAR) を EAL4+ に準拠させるための手順を説明します。

BAS システムで WPAR を作成し、その WPAR で次のコマンドを実行してそれが EAL4+ 準拠となるようにします。

```
/usr/lib/security/CC_EVALify.sh
```

LAS システムで初めて clogin を実行すると、(CC_EVALify.sh を含む) firstboot スクリプトが実行されます。

firstboot スクリプトが実行されると、clogin がログインのために TSM を呼び出すときに clogin は通常より長く実行されます。 ただし、WPAR は引き続き構成モードであるため、ログインは拒否されます。 別の clogin を試みる前に、WPAR が構成を完了するまで約 10 分間待つ必要があります。 新しく作成した WPAR システムでは、評価要件を満たすようにデフォルトのユーザー・オプションを設定する必要があります。これには以下が含まれます。

- root (BAS モード)
- isso/sa/so (LAS モード)

root および isso ユーザーにはパスワードがないか、ぜい弱なパスワードしか必要ではありません。 非トラステッド・ユーザーにグローバル環境またはそれぞれの WPAR へのアクセスを許可する前に、パスワードを更新する必要があります。

評価のパスワード要件として、パスワードを正確に推測されてしまう確率は最悪 1,000,000 分の 1 でなければなりません。また、1 分間に繰り返し試行してパスワードを正確に推測されてしまう確率は最悪 100,000 分の 1 でなければなりません。 この要件に準拠するために、/etc/security/user ファイルのユーザー・パラメーターを以下に変更します。

```
default:
 maxage       = 8
 maxexpired   = 1
 minother     = 2
 minlen       = 8
 maxrepeats   = 2
 loginretries = 3
 histexpire   = 52
 histsize     = 20
```

EFS の更新*:*

このセクションでは、暗号ファイルシステムとして評価された EFS のセキュリティー属性を設定する手順を説明します。

この評価には、root への全アクセス権限に対する Root Guard モードの側面は含まれていません。 EFS を使用可能にする時点で、次のコマンドを実行して、**efsmgr** コマンドと **egskeymgr** コマンドのセキュリティー属性を設定してください。

```
setsecattr -c accessauths=ALLOW_ALL
innateprivs=PV_DEV_QUERY,PV_DEV_CONFIG,PV_AU_ADD,PV_DAC_R,PV_DAC_W,PV_DAC_X /usr/sbin/efsmgr

setsecattr -c accessauths=ALLOW_ALL
innateprivs=PV_DEV_QUERY,PV_DEV_CONFIG,PV_AU_ADD,PV_DAC_R,PV_DAC_W,PV_DAC_X /usr/sbin/efskeymgr

setkst -t cmd
```

ハード・ディスクの消去*:*

AIX では、AIX 診断パッケージの「メディアのフォーマット」サービスを使用して hdisk を消去することができます。 診断パッケージについては、「*Diagnostic Information for Multiple Bus Systems*」ブック、およびご使用のハードウェアのユーザーズ・ガイドに詳細な説明があります。

ハード・ディスクを消去するには、次のコマンドを実行します。

```
diag -T "format"
```

このコマンドは、メニュー主導型インターフェースでの「メディアのフォーマット」サービスを開始します。 プロンプトが出されたら、ご使用の端末装置を選択します。

リソース選択リストが表示されます。 このリストから消去したい hdisk デバイスを選択し、画面の指示に従って変更をコミットしてください。

選択をコミットした後、メニューから「ディスクの消去」を選択します。 その後、選択の確認を求めるプロンプトが表示されます。 「はい」を選択します。

すると、「ドライブからデータを読み取る」か「ドライブにパターンを書き込む」かをたずねるプロンプトが表示されます。 「ドライブにパターンを書き込む」を選択します。

この後、ディスク消去オプションを変更することができます。使用したいオプションを指定した後、「変更のコミット」を選択します。 ディスクが消去されます。

注**:** この処理は完了までに長い時間がかかる場合があります。

リソース制限*:*

/etc/security/limits ファイルでリソース制限を設定する際には、制限がシステム上でのプロセスの必要に適合していることを確認してください。

特に、stack サイズは unlimited に設定しないでください。stack を無制限に設定すると、実行中のプロセスの他のセグメントが上書きされることがあります。stack_hard サイズも制限する必要があります。

監査サブシステム*:*

監査サブシステムの保護を助けるためのいくつかの手順があります。

- 監査サブシステムを、ユーザーの関係するすべてのセキュリティー・アクティビティーを記録するように構成します。 監査に必要なファイル・スペースが使用可能になっており、ファイルシステム・スペースの他の消費者によって損なわれないようにするには、監査データ専用のファイルシステムをセットアップしてください。
- 監査レコード (監査証跡、bin ファイル、および **/audit** に保管されているその他のすべてのデータ) を非 root ユーザーから保護します。
- BAS/EAL4+ システムの場合、監査サブシステムが使用される際に **bin** モード監査を設定する必要があります。 監査サブシステムのセットアップの詳細については、 163 ページの『監査のセットアップ』を参照してください。
- システム内の使用可能なディスク・スペースの少なくとも 20 % が、監査証跡専用でなければなりません。
- 監査が使用可能である場合、**/etc/security/audit/config** ファイル内の start スタンザの *binmode* パラメーターを panic に設定する必要があります。 bin スタンザの *freespace* パラメーターは、少なくとも監査証跡のストレージ専用のディスク・スペースの 25% に等しい値に構成する必要があります。 *bytethreshold* および *binsize* パラメーターは、いずれも 65,536 バイトに設定してください。
- 監査レコードをシステムから永続ストレージに保存用にコピーします。

分散システムにおける非共用ファイル*:*

/etc/security ディレクトリーにある以下のファイルは分散システムで共有されず、ホスト指定のままになります。

/etc/security/failedlogin
ホストごとの失敗したログインに関するログ・ファイル

/etc/security/lastlog
このホスト上で最後に成功したおよび失敗したログインに関するユーザーごとの情報

/etc/security/login.cfg
トラステッド・パス、ログイン・シェル、およびその他のログイン関連情報に関する、ホスト固有のログイン特性

/etc/security/portlog
このホスト上でロックされているポートに関するポートごとの情報

共有ファイルの自動生成済みバックアップ・ファイルも非共有です。 バックアップ・ファイルはオリジナル・ファイルと同じ名前ですが、小文字の o が前に付けられます。

ユーザー・ベースおよびポート・ベースのネットワーク・アクセス制御のための ***DACinet*** フィーチャーの使用*:*

DACinet フィーチャーを使用して、TCP ポートへのユーザーのアクセスを制限することができます。

DACinet の詳細については、 235 ページの『インターネット・ポート用の任意アクセス制御によるユーザー・ベースの TCP ポート・アクセス制御』を参照してください。 例えば、DACinet を使用して、ポート TCP/25 インバウンドへのアクセスを DACinet フィーチャーを使用できる root のみに制限すると、BAS/EAL4+ 準拠ホストの root ユーザーだけがこのポートにアクセスできます。 この状況では、通常のユーザーが telnet を使用して被害者側のポート TCP/25 に接続し、電子メール・スプーフィングを行うという可能性が制限されます。

ブート時に TCP 接続用の ACL を活動化するには、**/etc/inittab** から **/etc/rc.dacinet** スクリプトを実行します。 これにより、**/etc/security/acl** ファイル内の定義が読みとられ、ACL がカーネルにロードされます。 ACL によって保護されないポートは、**/etc/services** ファイルと同一のフォーマットを使用している **/etc/security/services** ファイルにリストする必要があります。

接続されたすべてのシステムのサブネットを 10.1.1.0/24 と想定すると、**/etc/security/acl** ファイル内の X (TCP/6000) について、アクセスを root ユーザーのみに制限する ACL 項目は以下のようになります。

```
6000    10.1.1.0/24 u:root
```

BAS/EAL4+ 準拠システムへの他のソフトウェアのインストール*:*

管理者は、BAS/EAL4+ 準拠システムに他のソフトウェアをインストールすることができます。 root ユーザーまたは root ユーザー特権を持つユーザーがこの追加ソフトウェアを実行しない限り、BAS/EAL4+ 準拠は無効になりません。 一般的な例として、通常のユーザーによってのみ実行され、SUID コンポーネントを持たないオフィス・アプリケーションがあります。

また、インストールされたソフトウェアを root ユーザー特権で実行すると、BAS/EAL4+ 準拠が無効になります。 これは、例えば旧式の JFS 用のドライバーをインストールしてはならないことを意味します。それらはカーネル・モードで実行するからです。 `/etc/security/privcmds` を通じて 1 つ以上の特権が付与されたアプリケーションはいずれも許容されません。 root として実行される追加デーモン (例えば、SNMP デーモン) も、BAS/EAL4+ 準拠を無効にします。 BAS/EAL4+ 使用可能システムは (通常は) 更新できません。

BAS/EAL4+ 準拠システムは、評価済み構成 (特に商用環境) で使用されることはめったにありません。一般に、実動システムが評価済みシステムに基づくようにするには追加サービスが必要とされます。しかし、それは評価済みシステムの仕様に完全に準拠しているわけではありません。

NSF v4 アクセス制御リストおよび内容ポリシー*:*

NFS v4 アクセス制御リスト (ACL) には、**Type**、**Mask**、および **Flags** の各フィールドが含まれます。

以下は、これらのフィールドについての説明です。

- 「**Type (**タイプ**)**」フィールドには、次のいずれかの値を指定できます。
  - `ALLOW` – 「**Who (**対象者**)**」フィールドで指定されたサブジェクトに対して、「**Mask (**マスク**)**」フィールドで指定された許可を付与します。
  - `DENY` – 「**Who (**対象者**)**」フィールドで指定されたサブジェクトに対して、「**Mask (**マスク**)**」フィールドで指定された許可を拒否します。
- 「Mask (マスク)」フィールドには、以下の詳細な許可値のうち 1 つ以上の値が入ります。

- READ_DATA / LIST_DIRECTORY – 非ディレクトリー・オブジェクトからデータを読み取るか、またはディレクトリー内のオブジェクトをリストします。
- WRITE_DATA / ADD_FILE – 非ディレクトリー・オブジェクトにデータを書き込むか、または非ディレクトリー・オブジェクトをディレクトリーに追加します。
- APPEND_DATA / ADD_SUBDIRECTORY – 非ディレクトリー・オブジェクトにデータを追加するか、またはサブディレクトリーをディレクトリーに追加します。
- READ_NAMED_ATTRS – オブジェクトの指定された属性を読み取ります。
- WRITE_NAMED_ATTRS – オブジェクトの指定された属性を書き込みます。
- EXECUTE – ファイルを実行するか、またはディレクトリーのトラバース/検索を行います。
- DELETE_CHILD – ディレクトリー内のファイルまたはディレクトリーを削除します。
- READ_ATTRIBUTES – ファイルの基本 (非 ACL) 属性を読み取ります。
- WRITE_ATTRIBUTES – ファイルまたはディレクトリーに関連付けられている時間を変更します。
- DELETE – ファイルまたはディレクトリーを削除します。
- READ_ACL – ACL を読み取ります。
- WRITE_ACL – ACL を書き込みます。
- WRITE_OWNER – 所有者およびグループを変更します。
- SYNCHRONIZE – アクセスを同期化します (他の NFS v4 クライアントとの互換性保持のために存在するが、実装された機能はない)。

- 「**Flags (フラグ)**」 フィールド – このフィールドは、ディレクトリー ACL の継承機能を定義し、「**Who (対象者)**」フィールドにグループが含まれるかどうかを示します。 このフィールドには、何も指定しないか、または以下の 1 つ以上のフラグを指定します。
  - **FILE_INHERIT** – このディレクトリーで、新たに作成された非ディレクトリー・オブジェクトがこのエントリーを継承することを指定します。
  - **DIRECTORY_INHERIT** – このディレクトリーで、新たに作成されたサブディレクトリーがこのエントリーを継承することを指定します。
  - **NO_PROPAGATE_INHERIT** – このディレクトリーで、新たに作成されたサブディレクトリーがこのエントリーを継承することを指定しますが、これらのサブディレクトリーはこのエントリーをこれらのサブディレクトリーの新たに作成されたサブディレクトリーには渡しません。
  - **INHERIT_ONLY** – このエントリーがこのディレクトリーに適用されず、新しく作成されたこのエントリーを継承するオブジェクトにのみ適用されることを指定します。
  - **IDENTIFIER_GROUP** – 「**Who (対象者)**」フィールドがグループを表すことを指定します。このエントリーを指定しない場合は、「**Who (対象者)**」フィールドはユーザーまたは「**Who (対象者)**」特殊値を表します。
- 「**Who (対象者)**」 フィールド – このフィールドには、次のいずれかの値が含まれます。
  - User (ユーザー) – このエントリーが適用されるユーザーを指定します。
  - Group (グループ) – このエントリーが適用されるグループを指定します。
  - Special (特殊) – この属性は次のいずれかの値です。
    - OWNER@ – このエントリーがオブジェクトの所有者に適用されることを指定します。
    - GROUP@ – このエントリーがオブジェクトを所有するグループに適用されることを指定します。
    - EVERYONE@ – このエントリーが所有者およびグループを含むシステムのすべてのユーザーに適用されることを指定します。

ACL が空の場合には、有効な UID の 0 を指定したサブジェクトのみが当該オブジェクトにアクセスできます。 オブジェクトの所有者は、ACL に何かが含まれているか否かに関わらず、暗黙的に次のマスク値をもつことになります。

- READ_ACL
- WRITE_ACL
- READ_ATTRIBUTES
- WRITE_ATTRIBUTES

APPEND_DATA 値は WRITE_DATA として実行されます。事実上、WRITE_DATA 値と APPEND_DATA 値の間には機能上の違いはありません。 両方の値の設定または設定解除は、そろえて行う必要があります。

オブジェクトの所有権は、WRITE_OWNER 値を使用して変更することができます。所有者またはグループが変更されると、**setuid** ビットがオフになります。 継承フラグはディレクトリーの ACL でのみ意味をもち、継承フラグが設定された後にディレクトリーで作成されるオブジェクトにのみ適用されます (例えば、既存のオブジェクトは親ディレクトリーの ACL への継承の変更に影響されません)。 NFS v4 ACL のエントリーは順序に従います。 要求されたアクセスを許可するかどうか決めるために、各エントリーが順番に処理されます。次の値をもつエントリーのみが考慮されます。

- 有効 UID と一致している **Who** フィールド
- エントリーまたは有効 GID に指定されているユーザー
- サブジェクトのエントリーに指定されているグループ

リクエスターのアクセスの全ビットが許可されるまで、各エントリーが処理されます。 アクセス・タイプは、エントリーによって許可された後は、後続のエントリーの処理において考慮されなくなります。該当のマスク値へのリクエスターのアクセスが必要で不確定な場合に DENY エントリーが検出された場合は、その要求は拒否されます。 評価が ACL の最後まで達した場合にも、その要求は拒否されます。

サポートされる最大 ACL サイズは 64 KB です。 ACL の中の各エントリーは可変長で、64 KB はエントリーに関する唯一の制限となります。

***WRITE OWNER* 値:**

NFS v4 ポリシーにより、オブジェクトの属性の読み取りおよび書き込みを行うことができるユーザーを制御することができます。

有効な UID 0 をもつサブジェクトは、常に NFS v4 ポリシーを指定変更することができます。 オブジェクト所有者は、ACL マスクの READ_ATTRIBUTES、WRITE_ATTRIBUTES、READ_NAMED_ATTRS、および WRITE_NAME_ATTRS 属性を使用して、オブジェクトの属性の読み取りおよび書き込みを他のユーザーに対して許可することができます。所有者は、ACL マスクの READ_ACL および WRITE_ACL 値を使用して、ACL の読み取りおよび書き込みを行うことができるユーザーを制御することができます。 オブジェクト所有者は、常に READ_ATTRIBUTES、WRITE_ATTRIBUTES、READ_ACL、および WRITE_ACL のアクセスが可能です。 オブジェクト所有者は、WRITE_OWNER 属性を使用して、オブジェクトの所有者およびグループを変更することを、他のユーザーに許可することもできます。オブジェクト所有者はオブジェクトの所有者またはグループをデフォルトで変更することはできませんが、オブジェクト所有者は所有者自身を指定する ACL に WRITE_OWNER エントリーを追加することができるか、またはオブジェクトが OWNER@ の **Who** 値によって WRITE_OWNER エントリーを指定する ACL 項目を継承することができます。 所有者またはグループが変更されると、**setuid** ビットがオフになります。

上記のルールには、次のようないくつかの例外があります。

- オブジェクトが UID 0 によって所有されている場合は、UID 0 のみが所有者を変更できますが、グループは引き続き WRITE_OWNER 属性をもつサブジェクトによって変更することができます。
- オブジェクトがサブジェクトに対して WRITE_OWNER 属性をもつと仮定すると、Technology Level 5300-05 より前の AIX 5.3 のバージョンでは、オブジェクトが非 UID 0 所有者をもつ場合は、所有者は別の非 UID 0 ユーザーにのみ変更することができます。 AIX (5300-05 適用) 以降では、オブジェクトが非 UID 0 所有者をもつ場合は、所有者は所有者の変更を試みるサブジェクトの EUID にのみ変更することができます。
- グループはサブジェクトの並行グループ・セット内のどのグループにも変更することができます。この例外として、GID 0 または GID 7 (システムまたはセキュリティー) には、この 2 つのグループがサブジェクトの並行グループ・セットにある場合でも、決して変更することはできません。

サポートされる ***LDAP*** ベースおよびファイル・ベースの管理データベース*:*

評価に関して、NFS 管理データベースはサポートされません。 DCE および NIS などの認証方式はサポートされません。

評価がサポートされるのは、以下の項目のみです。

- ファイル・ベース認証 (デフォルト)
- UNIX スタイルの LDAP ベース認証 (LDAP サーバー IBM Tivoli® Directory Server v 6.0 の使用)

ファイル・ベース認証の詳細情報については、『ユーザー認証』を参照してください。

LDAP 認証*:*

LDAP ベースの I&A は、「UNIX タイプ」の認証モードで構成されます。 このモードでは、管理データ (ユーザー名、ID、およびパスワード) は、データへのアクセスが LDAP 管理者に制限される LDAP 内に保管されます。

ユーザーがシステムへログインするときに、システムは SSL 接続を経由して LDAP 管理者アカウントの使用により LDAP サーバーにバインドし、ユーザーに必要なデータを (パスワードを含めて) LDAP から取り出して、 LDAP から取得したデータにより認証を行います。 システムは LDAP サーバー上の管理データベースを保守します。 その他のホストは、前に説明した同じメカニズムに従って管理データを LDAP サーバーからインポートします。システムは指定の LDAP サーバーですべての管理変更を行い、整合性のある管理データベースを維持します。 任意のコンピューター上のユーザー ID は、他のすべてのコンピューターでも同じ個人を表します。さらに、パスワード構成、名前と UID のマッピング、およびその他のデータは、分散システムのすべてのホストで同じです。

LDAP 認証セットアップの詳細情報については、『Light Directory Access Protocol』を参照してください。 LDAP で SSL をセットアップする場合の詳細情報については、『LDAP サーバーでの SSL のセットアップ』、および『LDAP クライアントでの SSL のセットアップ』を参照してください。

LDAP サーバー*:*

mksecldap -s コマンドは、 AIX システムをセキュリティー認証およびデータ管理のための LDAP サーバーとしてセットアップします。

以下のタスクを実行します。

- **-S** オプションを指定して RFC2307AIX スキーマを使用します。
- SSL (Secure Sockets Layer) を使用するサーバーの設定には、**-k** オプションを使用します。この処理では、**GSKit V8** ファイルセットおよび **idsldap.clt_max_crypto32bit63.rte** ファイルセット (32 ビッ

ト・システムの場合) または **idsldap.clt_max_crypto64bit63.rte** ファイルセット (64 ビット・システムの場合) をインストールする必要があります。ディレクトリー・サーバー用の鍵ペアを生成する場合は、ikeyman ユーティリティーを使用します。

評価の要件を満たすために、LDAP ユーザー・オプションを設定する必要があります。 RFC2370AIX スキーマにはユーザー属性を定義します。 BAS/EAL4+ システム構成の説明と同じ値を使用します。 Tivoli Directory Server 管理者は自分のパスワードの定期的な変更を強制されません (例えば、管理パスワードに対する **MaxAge** 値がありません)。 このために、LDAP 管理パスワードは AIX ユーザー (**MaxAge** = 8 (週)) と同じくらい頻繁に変更する必要があります。

Tivoli Directory Server 6.3 では、認証障害の処理はディレクトリー管理者または管理グループのメンバーに適用されません。 パスワードの構成ルールも管理アカウントに適用されません。 Tivoli Directory Server 6.3 が使用される場合は、必ずこれらのルールに従ってください。

管理者がユーザー管理のために共通の LDAP データベース・バックエンドを使用しない場合でも、管理者は、ユーザー資格情報が収められているデータベースが 1 つのネットワーク内の異なる TCP オフロード・エンジン (TOE) システム部分の間で確実に整合性が維持されるようにする必要があります。以下に例を示します。

- `/etc/group`
- `/etc/passwd`
- `/etc/security/.ids`
- `/etc/security/.profile`
- `/etc/security/environ`
- `/etc/security/group`
- `/etc/security/limits`
- `/etc/security/passwd`
- `/etc/security/user`

関連情報:

パッケージ、ファイルセット、および前提条件に関する IBM Tivoli Directory Server 情報

LDAP* クライアント*:

mksecldap -c コマンドは、セキュリティー認証およびデータ管理のための LDAP クライアントとして AIX システムをセットアップします。

以下のタスクを実行します。

- **-A** オプション指定の **mksecldap -c** コマンドを使用して、 **authType** の **unix_auth** を指定します。
- SSL を使用するクライアントの設定には **-k** オプション指定の **mksecldap-c** コマンドを使用します。クライアント SSL 鍵を指定するには、**GSKit** ファイルセットおよび **ldap.max_crypto_client** ファイルセットをインストールする必要があります。ディレクトリー・サーバー用の鍵ペアを生成する場合は、**gsk7ikm** ユーティリティーを使用します。

NFS v4* クライアント/サーバーおよび *Kerberos:

NFS v4 クライアント/サーバー環境には、認証データおよび NFS v4 クライアント/サーバー間にトラステッド・チャネルを確立する場合に Kerberos を維持するための LDAP が組み込まれていなければなりま

せん。 評価済み構成では、Kerberos 用に NAS v1.4 とユーザー・データベース用に IBM Tivoli Directory Server v6.0 (LDAP サーバー) がサポートされます。

NAS v1.4 (Kerberos バージョン 5 サーバー) は、ユーザー・データベース用に LDAP を使用して構成する必要があります。 Kerberos サーバーによって以前に認可された Kerberos チケットは、その有効期限が切れるまでは有効です。

Kerberos 認証を使用している場合、ユーザーによって開始されたリモート・プロシージャー・コールで使用される資格情報は、ユーザーによって保持された現行の Kerberos チケットに関連付けされます。そして、この資格情報は実際には影響がないか、またはプロセスの有効な UID ではありません。 **setuid** プログラムの実行中に Kerberos 認証を使用して NFS リモート・ファイルシステムをアクセスするときに、サーバーで見られる UID は Kerberos ID に基づくものであり、実行中の **setuid** プログラムを所有する UID に基づくものではありません。

評価済み構成には RPCSEC-GSS セキュリティーを使用するための NFS のセットアップが含まれます。詳細情報については、『ネットワーク・ファイルシステム』、『NFS サーバーの構成』、および『NFS クライアントの構成』を参照してください。 サーバーをセットアップするときに、Kerberos 認証を選択しサーバーの拡張セキュリティーを有効にします。これを有効にするには、SMIT で **chnfs** コマンドを使用します。 **chnfs** コマンドに RPCSEC_GSS セキュリティーを有効にするオプションがあります。クライアントをセットアップする場合は、以下の手順に従って NFS クライアントの構成で Kerberos を使用します。セキュリティーのための DES3 暗号化を指定して Kerberos データ・サーバーをセットアップするための手順については、『RPCSEC-GSS のためのネットワークのセットアップ』を参照してください。 評価済み構成では des3 暗号化のみサポートされます。

パスワード・ルール*:*

評価済み構成では、LDAP を使用する Kerberos サーバーをデータベースとして使用する場合のパスワード・ルールには、以下の値を指定する必要があります。

パスワード・ルールの詳細情報については、「*IBM Network Authentication Service Version 1.4 for AIX, Linux*」および「*Solaris Administrator's and User's Guide*」 の『Chapter 9. Managing Network Authentication Service passwords』を参照してください。

値のリストを以下に示します。

mindiff
4

maxrepeats
2

minalpha
2

minother
2

minlen
8

minage
0

histsize
10

DES3 暗号化タイプのみを使用して、AIX NFS v4 クライアントおよび AIX NFS v4 サーバーを明示的に安全に通信させるには、 DES3 暗号化タイプ (例えば、des3-cbc-sha1 など) 指定の「NFS/ホスト名」サーバー・プリンシパルを作成します。同時に、keytab ファイルにも (**kadmin** インターフェースを使用して) 対応するエントリーを作成し、 NFS v4 クライアント・マシン上にある /etc/krb5/krb5.conf ファイルの **default_tgs_enctypes** セクションの先頭のエントリーとして DES3 (例えば、 **des3-cbc-sha1** など) を作成します。

仮想入出力サーバー*:*

仮想入出力サーバー (VIOS) は別に分離された LPAR パーティションにあり、マッピングを介して LPAR パーティションおよび SCSI ベースの論理ボリュームと物理ボリュームのために機能を果たす VIOS SCSI デバイス・ドライバー間における、基本的な任意アクセス制御を提供します。

1 つの LPAR パーティションは (VIOS SCSI デバイス・ドライバーを経由して) 0 または 1 つ以上の論理および物理ボリュームにマップすることができますが、1 つのボリュームは 1 つの LPAR パーティションにのみマップすることができます。 LPAR パーティションに関するこのマッピング制限は、LPAR パーティションに割り当てられるボリュームのみに対するものです。 また、VIOS は仮想ネットワークを共有する LPAR パーティションのグループのために機能を果たす VIOS イーサネットのデバイス・ドライバーへの VIOS イーサネット・アダプターのデバイス・ドライバーのマッピングを制御します。評価済み構成では、LPAR パーティションのグループのために機能を果たすイーサネット・デバイス・ドライバーへのイーサネット・アダプターのデバイス・ドライバーのマッピングは、1 対 1 のみ許可されます。 1 対 1 マッピングは管理者によって構成され、デバイス・ドライバーによって実施されます。また、イーサネット・パケットには、評価済み構成で VLAN タグを付けないようにしてください。 このメカニズムは、特定のイーサネット・パケットを観察する LPAR パーティションを制限する場合に使用できます。

特権を持たないユーザーからのアクセスに対して、VIOS インターフェースを保護する必要があります。評価の要件を満たすために、VIOS ユーザー・オプションを設定する必要があります。 実際の要件としては、秘密鍵が所定の品質メトリック (すなわち、秘密鍵の存続時間中にアタッカーが秘密鍵を取得できる確率は 2^-20 より小さい値でなければならない) を満たしていることを確認するメカニズムを TSF が提供しなければならないということです。 /etc/security/user ディレクトリーに入っているユーザーに対して、次のパラメーターを変更する必要があります。

maxage
8

maxexpired
1

minother
2

minlen
8

maxrepeats
2

loginretries
3

histexpire
52

histsize
20

デフォルトを変更するには、以下のコマンドを使用します。

```
type oem_setup_env

chsec -f /etc/security/user -s default  -a maxage=8 -a maxexpired=1 -a minother=2
  -a minlen=8 -a maxrepeats=2 -a loginretries=3 -a histexpire=52 -a histsize=20
```

基本の管理者 (**padmin**) が新規ユーザーを作成する場合、そのユーザーに対してユーザー属性を明示的に指定しなければなりません。 例えば、名前が *davis* のユーザーを作成するには、 **padmin** は以下のコマンドを使用します。

```
mkuser  maxage=8 maxexpired=1 minother=2 minlen=8 maxrepeats=2 loginretries=3
  histexpire=52 histsize=20 davis
```

padmin は、以下のようにデーモンを停止してからリブートすることも必要です。

- **writesrv** および **ctrmc** を /etc/inittab ファイルから除去する場合:

  ```
  sshd:   stopsrc -s sshd
  ```

- ブート時にデーモンを始動しないようにするには、/etc/rc.d/rc2.d/Ksshd ファイルと /etc/rc.d/rc2.d/Ssshd ファイルを除去します。 リブート後に RSCT デーモンを停止する場合:

  ```
  stopsrc -g rsct_rm stopsrc -g rsct
  ```

すべてのユーザーは、そのロールに関係なく、管理ユーザーと見なされることになります。

システム管理者は基本の管理者 (**padmin**) に限定される以下のリストにあるコマンドを除く、すべてのコマンドを実行することができます。

- **chdate**
- **chuser**
- **cleargcl**
- **de_access**
- **diagmenu**
- **invscout**
- **loginmsg**
- **lsfailedlogin**
- **lsgcl**
- **mirrorios**
- **mkuser**
- **motd**
- **oem_platform_level**
- **oem_setup_env**
- **redefvg**
- **rmuser**
- **shutdown**
- **unmirrorios**

ログイン制御

システムのインストール後、セキュリティー上の理由により、ログイン画面のデフォルト値を変更することができます。

ハッカーとなり得る人物は、デフォルトの AIX ログイン画面から、ホスト名やオペレーティング・システム・バージョンなど、貴重な情報を手に入れることができます。 このような情報は、攻撃者が試行する方法を決める上で役立つことがあります。セキュリティー上の理由により、システムのインストール後できる限り早くログイン画面のデフォルトを変更する必要があります。

KDE および GNOME デスクトップにも同様のセキュリティー上の問題がいくつかあります。 KDE および GNOME について詳しくは、「インストールおよび移行」を参照してください。

ユーザー、グループ、およびパスワードについては、 53 ページの『ユーザー、グループ、およびパスワード』を参照してください。

ログイン制御のセットアップ**:**

`/etc/security/login.cfg` ファイルにログイン制御をセットアップすることができます。

パスワード推定によるシステムのアタックを困難にするために、`/etc/security/login.cfg` ファイルに次のようにログイン制御を設定します。

表 *1.* ログイン制御のための属性と推奨値

属性	**PtYs** に適用 **(**ネットワーク**)**	**TTYs** に適用	推奨値	コメント
sak_enabled	Y	Y	false	セキュア・アテンション・キーはほとんど必要ありません。 5 ページの『セキュア・アテンション・キーの使用』を参照してください。
logintimes	N	Y		許可されているログイン回数を指定します。
logindisable	N	Y	4	ログインの試行が 4 回連続して失敗すると、この端末装置でのログインを使用不可にします。
logininterval	N	Y	60	無効と指定されている試行が 60 秒以内に行われると、端末装置は使用不可にされます。
loginreenable	N	Y	30	端末装置が 30 分後に自動的に使用不可にされたら、それを再度使用可能にします。
logindelay	Y	Y	5	ログイン・プロンプトが表示される間隔の秒数。 これは、失敗した試行回数の倍数です。例えば、5 が初期値の場合は 5、10、15、20 秒になります。

これらのポート制限は、ネットワーク・ログインで使用される疑似端末で作動するのではなく、主に、接続された直列伝送端末で作動します。 このファイルでは、明示的端末を指定することができます。例えば、以下のようにします。

```
/dev/tty0:
        logintimes = 0600-2200
        logindisable = 5
        logininterval = 80
        loginreenable = 20
```

ログイン画面のウェルカム・メッセージの変更**:**

ログイン画面に特定の情報が表示されないようにするには、/etc/security/login.cfg ファイルの *herald* パラメーターを編集します。

デフォルトの *herald* には、ユーザーのログイン・プロンプトを表示するウェルカム・メッセージが含まれています。 このパラメーターを変更するには、**chsec** コマンドを使用するか、このファイルを直接編集してください。

次の例では、**chsec** コマンドを使用してデフォルトの *herald* パラメーターを変更しています。

```
# chsec -f /etc/security/login.cfg -s default
-a herald="Unauthorized use of this system is prohibited.¥n¥nlogin:"
```

chsec コマンドについての詳細は、「*Commands Reference, Volume 1*」を参照してください。

ファイルを直接編集するには、**/etc/security/login.cfg** ファイルを開き、次のように *herald* パラメーターを更新します。

```
default:
herald ="Unauthorized use of this system is prohibited¥n¥nlogin:"
 sak_enable = false
 logintimes =
 logindisable = 0
 logininterval = 0
 loginreenable = 0
 logindelay = 0
```

注**:** システムをさらに安全に保護するには、*logindisable* 変数および *logindelay* 変数を `0` より大きい数 (`# > 0`) に設定します。

CDE のログイン画面の変更**:**

このセキュリティー問題は、Common Desktop Environment (CDE) ユーザーにも影響します。 CDE ログイン画面にも、デフォルトで、ホスト名およびオペレーティング・システムのバージョンが表示されます。 この情報が表示されないようにするには、**/usr/dt/config/$LANG/Xresources** ファイルを編集します。ここで、**$LANG** はマシンにインストールされているローカル言語を表します。

ここで説明する例では、**$LANG** が **C** に設定されているとします。このファイルを **/etc/dt/config/C/Xresources** ディレクトリーにコピーします。次に、**/usr/dt/config/C/Xresources** ファイルを開いて、ホスト名およびオペレーティング・システムのバージョンが含まれているウェルカム・メッセージを除去するようにファイルを編集します。

CDE セキュリティー問題についての詳細は、 45 ページの『X11 および CDE 関連事項の管理』を参照してください。

ユーザー名の表示の使用不可化とパスワード・プロンプトの変更**:**

セキュアな環境では、ログイン・ユーザー名の表示を隠したり、デフォルトとは異なるカスタム・パスワード・プロンプトを提供したりする必要があります。

ログインおよびパスワード・プロンプトのデフォルト・メッセージの動作を以下に示します。

```
login: foo
foo's Password:
```

プロンプトおよびシステム・エラー・メッセージでユーザー名の表示を使用不可にするには、**/etc/security/login.cfg** ファイルの *usernameecho* パラメーターを編集します。 *usernameecho* のデフォルト値は true です。この値ではユーザー名が表示されます。 このパラメーターを変更するには、**chsec** コマンドを使用するか、ファイルを直接編集します。

次の例では、**chsec** コマンドを使用してデフォルトの *usernameecho* パラメーターを false に変更しています。

```
# chsec -f /etc/security/login.cfg -s default -a usernameecho=false
```

chsec コマンドについての詳細は、「*Commands Reference, Volume 1*」を参照してください。

ファイルを直接編集するには、**/etc/security/login.cfg** ファイルを開き、次のように *usernameecho* パラメーターを追加または変更します。

```
default:
 usernamecho = false
```

usernameecho パラメーターを false に設定すると、ログイン・プロンプトにユーザー名が表示されなくなります。その代わりに、次のように、システム・プロンプトとエラー・メッセージでユーザー名が「*」文字によってマスクされます。

```
login:
***'s Password:
```

パスワード・プロンプトは、**/etc/security/login.cfg** ファイルに *pwdprompt* パラメーターを設定して、個別にカスタム文字列に変更することができます。 デフォルト値は文字列 "*user*'s Password: " です。この *user* は認証中のユーザー名で置き換えられます。

このパラメーターを変更するには、**chsec** コマンドを使用するか、ファイルを直接編集します。

次の例では、**chsec** コマンドを使用してデフォルトの *pwdprompt* パラメーターを "Password: " に変更しています。

```
# chsec -f /etc/security/login.cfg -s default -a pwdprompt="Password: "
```

ファイルを直接編集するには、**/etc/security/login.cfg** ファイルを開き、次のように *pwdprompt* パラメーターを追加または変更します。

```
default:
 pwdprompt = "Password: "
```

pwdprompt パラメーターを "Password: " に設定すると、ログイン・プロンプトだけでなく、システム・パスワード・プロンプトを使用するその他のアプリケーションでも、指定したプロンプトが表示されるようになります。 カスタム・プロンプトが構成された場合のログインのプロンプトの動作は、次のようになります。

```
login: foo
Password:
```

システム・デフォルト・ログイン・パラメーターのセットアップ**:**

/etc/security/login.cfg ファイルを編集して、システム・デフォルトのログイン・パラメーターをセットアップします。

多くのログイン・パラメーター (例えば、新規ユーザー用にセットアップするもの: ログイン再試行回数、ログインの再使用可能化、ログイン間隔など) の基本デフォルトをセットアップするためには、`/etc/security/login.cfg` ファイルを編集します。

無人端末装置の保護**:**

lock コマンドおよび **xlock** コマンドを使用して、端末装置を保護します。

端末装置がログインしたままで無人状態になっている場合、すべてのシステムはぜい弱です。 最も深刻な問題は、root 権限で使用可能になっている端末装置をシステム・マネージャーが不在のままにしている場合に起こります。 一般に、ユーザーは端末装置から離れる場合にはログアウトしなければなりません。 システム端末を無保護のまま放置しておくと、セキュリティー上の危険に直面する可能性があります。 端末装置をロックするには、**lock** コマンドを使用します。インターフェースが AIXwindows の場合は、**xlock** コマンドを使用します。

自動ログオフの使用可能化**:**

自動ログオフを使用可能にして、侵入者がシステムのセキュリティーを脅かすことを防ぎます。

別のセキュリティー上の問題は、ユーザーが自分のアカウントを長時間不在のままにしておくことが原因で生じます。 こうした状況では、侵入者はユーザーの端末装置の制御をとり、システムのセキュリティーを危険にさらす可能性があります。

このような潜在的なセキュリティー上の危険を防ぐために、システム上で自動ログオフを使用可能にすることができます。 それには、環境変数 TMOUT および TIMEOUT を非アクティブ時間の秒数に設定します。 この非アクティブ時間が経過すると、次の例のように自動的にログオフされます。

```
TMOUT=600; TIMEOUT=600; export TMOUT TIMEOUT
```

上記の例では 600 という数は秒数を表し、10 分間に相当します。 この方式はシェル・アプリケーションからのみ機能します。 変数は、次のように読み取り専用にすることで誤って上書きされないように保護できます。

```
readonly TMOUT TIMEOUT
```

環境変数 TMOUT および TIMEOUT は、ユーザーの .profile ファイル内か、または `/etc/security/.profile` ファイル内で設定します。これにより、ユーザーの作成時にユーザーの .profile ファイルにそのファイルを追加できます。

スタック実行使用不可保護

コンピューター・システムを安全に保持することは、オンデマンド・ビジネスの重要な局面です。 今日の高度にネットワーク化された環境では、さまざまなソースからのアタックを撃退することが極度に困難になっています。

コンピューター・システムが高度な技術によるアタックの犠牲になって、その結果、企業や政府機関の毎日の業務が混乱に陥る可能性が増大しています。 アタックに対する絶対に確実な保護を提供できるセキュリティー手段は存在しないといっても、複数のセキュリティー・メカニズムを配置して、セキュリティー・アタックを挫折させる必要があります。 このセクションでは、バッファー・オーバーフローに基づく実行のために発生するアタックを阻止するために AIX で使用されているセキュリティー・メカニズムについて説明します。

セキュリティー・ブリーチ (抜け穴) はさまざまな形式で発生しますが、最も一般的な方法の 1 つは、システム提供の管理ツールをモニターしてバッファー・オーバーフローを探し出し、それを利用するという方法です。 バッファー・オーバーフロー・アタックが発生するのは、データが正しく検証されていないために (コマンド・ライン、環境変数、ディスク、または端末入出力など) 内部プログラム・バッファーが上書きされる場合です。 バッファー・オーバーフローを通して、実行中のプロセスにアタック・コードが挿入され、その実行中のプロセスの実行パスが変更されます。 戻りアドレスが上書きされて、挿入されたコードの位置にリダイレクトされます。 ブリーチの共通の原因として、境界検査が正しくないかまったく行われていないこと、またはデータ・ソースの妥当性に関する想定が間違っていることがあります。 例えば、

データ・オブジェクトに 1 KB のデータを保持できる十分な大きさがあっても、プログラムが入力の境界を検査しないために、そのデータ・オブジェクトに 1 KB を超えるデータをコピーさせる場合に、バッファー・オーバーフローが起こることがあります。

侵入者のゴールは、通常のユーザーに root 権限を提供するコマンドまたはツール (両方の場合もある) にアタックをかけることです。 すべての特権が使用可能になった状態でプログラムの制御が獲得され、バッファーのオーバーフローが許可されます。 典型的なアタックでは、root の所有する UID のセットまたはシェルの実行を引き起こすプログラムがフォーカスとされ、システムに対する root ベースのシェル・アクセスを獲得しようとします。

これらのアタックを防止するには、バッファー・オーバーフローを通して侵入するアタック・コードの実行をブロックします。 プロセスのメモリーのうち、一般に実行が行われない領域 (スタックおよびヒープ・メモリー領域) での実行を使用不可にします。

SED のバッファー・オーバーフロー保護メカニズム**:**

AIX では、スタック実行使用不可 (SED) メカニズムにより、スタックおよびプロセスの選択データ領域でコードの実行を使用不可にすることができます。

侵害性のプログラムを実行不可にして終了させると、アタッカーは、バッファー・オーバーフロー・アタックを通して root ユーザー特権を取得できなくなります。 この機能は、バッファー・オーバーフローを停止するものではありませんが、オーバーフローしたバッファーでのアタックの実行を使用不可に設定して、保護を提供します。

POWER4 プロセッサー・ファミリー以降、メモリーについてページ・レベルの実行の使用可能化/使用不可化機構を使用することができます。 AIX SED メカニズムでは、この基本的なハードウェア・サポートを使用して、選択されたメモリー領域での非実行フィーチャーを実装します。 このフィーチャーが使用可能になると、オペレーティング・システムは実行可能プログラムの実行中に各種のファイルを検査してフラグを付けます。 続いて、オペレーティング・システムのメモリー・マネージャーおよびプロセス・マネージャーに、作成中のプロセスについて SED が使用可能になったというアラートを出します。 選択されたメモリー領域は非実行のマークが付けられます。これらのマークされた領域でなんらかのコードが実行されると、ハードウェアは例外フラグを立て、オペレーティング・システムは対応するプロセスを停止します。例外およびアプリケーションの終了の詳細は、AIX エラー・ログ・イベントを通してキャプチャーされます。

SED は、主として **sedmgr** コマンドによって実装されます。 **sedmgr** コマンドにより、システム全体の SED モードの操作を制御することができ、また実行可能ファイルをベースとして SED フラグを設定することができます。

SED のモードとモニター**:**

AIX のスタック実行使用不可 (SED) メカニズムは、システム共通モード・フラグによって実装され、また実行可能ファイル・ベースの個別のヘッダー・フラグによっても実装されます。

システム共通フラグはシステム全体の SED の操作を制御しますが、ファイル・レベルのフラグは SED でファイルをどのように扱うべきかを示します。バッファー・オーバーフロー保護 (BOP) メカニズムは、4 つのシステム共通操作モードを提供します。

off SED メカニズムはオフであり、どのプロセスも SED 保護対象としてマークされません。

select ファイルの選択セットだけが SED 保護使用可能になり、モニター対象になります。ファイルの選

択セットは、実行可能プログラムのバイナリー・ヘッダーの中の SED 関連フラグを検討して選択されます。 実行可能プログラム・ヘッダーは、SED 関連フラグを使用可能にして、**select** モードに含むことを要求します。

setidfiles
保護メカニズムを要求しているファイルだけでなく、重要な **setuid** および **setgid** システム・ファイルのすべてについて、SED を使用可能にすることができます。 このモードでは、オペレーティング・システムは、**request** SED フラグが設定されているファイルに SED を提供するだけでなく、次のような特性を持つ実行可能ファイルについても SED を使用可能にします (ただし、ファイル・ヘッダーに *exempt* とマークされたファイルを除きます)。

- root が所有する SETUID ファイル
- 1 次グループが **system** または **security** である SETGID ファイル

all SED モードの免除を要求するファイルを除いて、システムにロードされるすべての実行可能プログラムが SED 保護の対象となります。 免除関連フラグは、実行可能プログラム・ヘッダーの一部です。

AIX の SED フィーチャーは、例外が発生したときにプログラムを停止する代わりにモニターする機能も提供します。 このシステム共通制御を使用すると、システム管理者は、SED を実動システムにデプロイする前にシステム環境をモニターして、障害や問題点がないか確認することができます。

sedmgr コマンドは、例外発生時にプロセスを停止する代わりに、ファイルをモニターするための SED の使用可能化を許可するオプションを提供します。 システム管理者は、実行可能プログラムで正当なスタック実行が進行しているかどうかを評価することができます。 この設定は、`-c` オプションを使用するシステム共通モードのセットと結合して機能します。 **monitor** モードがオンになると、システムは、SED 関連例外が発生してもプロセスの操作の継続を許可します。オペレーティング・システムは、プロセスを停止する代わりに、AIX エラー・ログに例外を記録します。 SED モニターがオフの場合は、オペレーティング・システムは違反のあるプロセスを停止し、SED ファシリティーごとに例外フラグを立てます。

SED モードのシステム共通フラグに変更を行った場合は、変更が有効になるようにシステムを再始動する必要があります。 これらの型のイベントは、すべて監査対象です。

実行可能ファイル用の **SED** フラグ**:**

AIX では、**sedmgr** コマンドを使用して SED メカニズムから実行可能ファイルにフラグを立てることができます。

2 つの新しい SED 関連フラグをサポートするようにリンカーが拡張され、実行可能ファイルのヘッダーで `select` オプションと `exempt` オプションが使用可能になりました。 実行可能ファイルは、**select** フラグにより、`select` モードのシステム共通 SED 操作時に SED 保護を要求して SED 保護の一部となることができます。また、**exempt** フラグにより、SED メカニズムからの免除を要求することができます。これらの実行可能ファイルは、どのプロセス・メモリー領域でも実行を使用不可にすることはできません。

免除フラグを使用すると、システム管理者は、SED メカニズムをモニターし、状態を評価することができます。 システム管理者は、アプリケーションの必要に応じてスタックおよびデータ領域での実行を使用可能にすることができますが、関連したリスクを理解している必要があります。

次の表は、システム共通設定およびファイル設定がどのように SED モードの操作に影響するかを示しています。

表 2. *SED* モードに影響するシステム共通設定およびファイル設定

システム **SED** モード	実行可能ファイルの **SED** フラグ			**Setuid root** ファイルまたは **setgid** システム/セキュリティー・ファイル
	request	**exempt**	**system**	
off	–	–	–	–
select	使用可能	–	–	–
setgidfiles	使用可能	–	–	使用可能
all	使用可能	–	使用可能	使用可能

SED の問題点と考慮事項:

デフォルトでは、AIX SED は **select** モードで出荷されます。 多数の **setuid** プログラムおよび **setgid** プログラムが、デフォルトで SED について **select** 使用可能になっており、保護モードで稼働します。

より古いバイナリー・ファイルが、スタック・ヒープ領域における非実行フィーチャーを処理できない場合、SED を使用可能にすると、それらのバイナリー・ファイルが中断される可能性があります。 これらのアプリケーションはスタック・データ領域で実行する必要があります。 システム管理者は、状態を評価し、**bopmgr** コマンドを使用してファイルに免除のフラグを立てることができます。 AIX Java™ 1.3.1 および AIX Java 1.4.2 は、Just-In-Time (JIT) コンパイラーを持っており、Java アプリケーションの実行中にネイティブ・オブジェクト・コードを生成して実行します (Java 仮想マシンが、アプリケーションの実行プロファイルを基に、どのコードをコンパイルするか決定します)。 このオブジェクト・コードは、JIT により割り当てられたデータ・バッファーに保管されます。 したがって、AIX が SED の **ALL** モードで稼働するように構成されている場合、システム管理者は Java バイナリー・ファイルの免除フラグを設定する必要があります。

実行可能ファイルの SED 関連フラグが変更された場合、それらのフラグはそのファイルの将来のロードおよび実行にのみ適用されます。 この変更は、このファイルに基づいて現在稼働中のプロセスには適用されません。 SED ファシリティーは、システム共通設定でもファイル・レベルの設定でも、32 ビットと 64 ビットの両方の実行可能プログラムの制御とモニターを行います。 SED ファシリティーは、AIX オペレーティング・システムが 64 ビット・カーネルで使用されている場合にのみ使用できます。

関連情報

sedmgr コマンド

AIX エラー・ロギング機能

X11 および CDE 関連事項の管理

X11 X サーバーおよび Common Desktop Environment (CDE) に関連したセキュリティー上の潜在的なぜい弱性があります。

/etc/rc.dt ファイルの除去:

高水準のセキュリティーを必要とするシステムの **/etc/rc.dt** ファイルを除去します。

CDE インターフェースはユーザーにとって便利ですが、それに関連したセキュリティー上の問題があります。 このため、高水準のセキュリティーを必要とするサーバーでは CDE を実行しないでください。 最善の解決策は CDE (dt) ファイルセットをインストールしないことです。 これらのファイルセットをシステムにインストールしてある場合は、アンインストールを検討してください。特に、**/etc/rc.dt** スクリプトは CDE を開始するので、アンインストールするのが適切です。

CDE についての詳細は、「オペレーティング・システムおよびデバイスの管理」を参照してください。

リモート **X** サーバーの無許可モニターの防止**:**

X11 サーバーに関連した重要なセキュリティー問題は、リモート・サーバーの無許可のサイレント・モニターです。

xwd コマンドおよび **xwud** コマンドを使用すると、キー・ストロークをキャプチャーする機能があるため、X サーバーのアクティビティーをモニターすることができます。したがって、パスワードや他の機密データを漏えいしてしまう可能性があります。 この問題を解決するには、これらの実行可能ファイルが構成上必要でない場合は除去するか、あるいは前述のコマンドへのアクセスが root のみになるように変更してください。

xwd コマンドおよび **xwud** コマンドは、**X11.apps.clients** ファイルセットの中にあります。

xwd コマンドおよび **xwud** コマンドを保存する必要がない場合は、OpenSSH または MIT Magic Cookies の使用を検討してください。 これらのサード・パーティーのアプリケーションを使用すれば、**xwd** コマンドおよび **xwud** コマンドの実行によって生じる危険を避けることができます。

OpenSSH および MIT Magic Cookies の詳細については、各アプリケーションの資料をそれぞれ参照してください。

アクセス制御の使用可能化と使用不可化**:**

X サーバーでは、リモート・ホストが **xhost +** コマンドを使用してシステムに接続することを許可します。

ホスト名を指定する際には必ず **xhost +** コマンドを使用してください。このコマンドで X サーバーのアクセス制御が使用不可になるからです。 これにより特定ホストへのアクセスの付与が許可されるため、X サーバーへの潜在的なアタックに対するモニターが容易になります。 特定のホストへのアクセスを認可するには、次のように **xhost** コマンドを実行します。

```
# xhost + hostname
```

ホスト名を指定しない場合、アクセスはすべてのホストに対して認可されます。

xhost コマンドについての詳細は、「コマンド・リファレンス」を参照してください。

xhost コマンドを実行するためのユーザー・アクセス権の使用不可化**:**

xhost コマンドの無許可実行を防止するために、**chmod** コマンドを使用することができます。

xhost コマンドが適切に使用されるようにするもう 1 つの方法は、このコマンドの実行を root ユーザー権限のみに制限することです。 そのためには、**chmod** コマンドを使用して、次のように **/usr/bin/X11/xhost** のアクセス権を 744 に変更します。

```
chmod 744/usr/bin/X11/xhost
```

setuid/setgid プログラムのリスト

AIX システム上にはさまざまな setuid/setgid プログラムがあります。 通常のユーザーには使用可能とする必要のないコマンドに関するこれらの特権を除去することができます。

以下のプログラムは、AIX の通常のインストールに組み込まれます。CC 構成の AIX システムでは、このリストは除去され、少数のプログラムが組み込まれます。

- /opt/IBMinvscout/bin/invscoutClient_VPD_Survey
- /opt/IBMinvscout/bin/invscoutClient_PartitionID
- /usr/lpp/diagnostics/bin/diagsetrto
- /usr/lpp/diagnostics/bin/Dctrl
- /usr/lpp/diagnostics/bin/diagela
- /usr/lpp/diagnostics/bin/diagela_exec
- /usr/lpp/diagnostics/bin/diagrpt
- /usr/lpp/diagnostics/bin/diagrto
- /usr/lpp/diagnostics/bin/diaggetrto
- /usr/lpp/diagnostics/bin/update_manage_flash
- /usr/lpp/diagnostics/bin/utape
- /usr/lpp/diagnostics/bin/uspchrp
- /usr/lpp/diagnostics/bin/update_flash
- /usr/lpp/diagnostics/bin/uesensor
- /usr/lpp/diagnostics/bin/usysident
- /usr/lpp/diagnostics/bin/usysfault
- /usr/lpp/X11/bin/xlock
- /usr/lpp/X11/bin/aixterm
- /usr/lpp/X11/bin/xterm
- /usr/lpp/X11/bin/msmitpasswd
- /usr/lib/boot/tftp
- /usr/lib/lpd/digest
- /usr/lib/lpd/rembak
- /usr/lib/lpd/pio/etc/piodmgrsu
- /usr/lib/lpd/pio/etc/piomkpq
- /usr/lib/lpd/pio/etc/pioout
- /usr/lib/mh/slocal
- /usr/lib/perf/libperfstat_updt_dictionary
- /usr/lib/sa/sadc
- /usr/lib/semutil
- /usr/lib/trcload
- /usr/sbin/allocp
- /usr/sbin/audit
- /usr/sbin/auditbin
- /usr/sbin/auditcat
- /usr/sbin/auditconv
- /usr/sbin/auditmerge
- /usr/sbin/auditpr
- /usr/sbin/auditselect
- /usr/sbin/auditstream

- /usr/sbin/backbyinode
- /usr/sbin/cfgmgr
- /usr/sbin/chcod
- /usr/sbin/chcons
- /usr/sbin/chdev
- /usr/sbin/chpath
- /usr/sbin/chtcb
- /usr/sbin/cron
- /usr/sbin/acct/accton
- /usr/sbin/arp64
- /usr/sbin/arp
- /usr/sbin/devinstall
- /usr/sbin/diag_exec
- /usr/sbin/entstat
- /usr/sbin/entstat.ethchan
- /usr/sbin/entstat.scent
- /usr/sbin/diskusg
- /usr/sbin/exec_shutdown
- /usr/sbin/fdformat
- /usr/sbin/format
- /usr/sbin/fuser
- /usr/sbin/fuser64
- /usr/sbin/getlvcb
- /usr/sbin/getlvname
- /usr/sbin/getvgname
- /usr/sbin/grpck
- /usr/sbin/getty
- /usr/sbin/extendvg
- /usr/sbin/fastboot
- /usr/sbin/frcactrl64
- /usr/sbin/frcactrl
- /usr/sbin/inetd
- /usr/sbin/invscout
- /usr/sbin/invscoutd
- /usr/sbin/ipl_varyon
- /usr/sbin/keyenvoy
- /usr/sbin/krlogind
- /usr/sbin/krshd
- /usr/sbin/lchangelv
- /usr/sbin/lchangepv

- /usr/sbin/lchangevg
- /usr/sbin/lchlvcopy
- /usr/sbin/lcreatelv
- /usr/sbin/ldeletelv
- /usr/sbin/ldeletepv
- /usr/sbin/lextendlv
- /usr/sbin/lmigratelv
- /usr/sbin/lmigratepp
- /usr/sbin/lparsetres
- /usr/sbin/lpd
- /usr/sbin/lquerylv
- /usr/sbin/lquerypv
- /usr/sbin/lqueryvg
- /usr/sbin/lqueryvgs
- /usr/sbin/lreducelv
- /usr/sbin/lresynclp
- /usr/sbin/lresynclv
- /usr/sbin/lsaudit
- /usr/sbin/lscfg
- /usr/sbin/lscons
- /usr/sbin/lslv
- /usr/sbin/lspath
- /usr/sbin/lspv
- /usr/sbin/lsresource
- /usr/sbin/lsrset
- /usr/sbin/lsslot
- /usr/sbin/lsuser
- /usr/sbin/lsvg
- /usr/sbin/lsvgfs
- /usr/sbin/login
- /usr/sbin/lvaryoffvg
- /usr/sbin/lvaryonvg
- /usr/sbin/lvgenmajor
- /usr/sbin/lvgenminor
- /usr/sbin/lvrelmajor
- /usr/sbin/lvrelminor
- /usr/sbin/lsmcode
- /usr/sbin/mailq
- /usr/sbin/mkdev
- /usr/sbin/mklvcopy

- /usr/sbin/mknod
- /usr/sbin/mkpasswd
- /usr/sbin/mkpath
- /usr/sbin/mkvg
- /usr/sbin/mount
- /usr/sbin/netstat64
- /usr/sbin/mtrace
- /usr/sbin/ndp
- /usr/sbin/newaliases
- /usr/sbin/named9
- /usr/sbin/named8
- /usr/sbin/netstat
- /usr/sbin/nfsstat
- /usr/sbin/pdelay
- /usr/sbin/pdisable
- /usr/sbin/penable
- /usr/sbin/perf/diag_tool/getschedparms
- /usr/sbin/perf/diag_tool/getvmparms
- /usr/sbin/phold
- /usr/sbin/portmir
- /usr/sbin/pshare
- /usr/sbin/pstart
- /usr/sbin/putlvcb
- /usr/sbin/putlvodm
- /usr/sbin/qdaemon
- /usr/sbin/quota
- /usr/sbin/reboot
- /usr/sbin/redefinevg
- /usr/sbin/repquota
- /usr/sbin/restbyinode
- /usr/sbin/rmdev
- /usr/sbin/ping
- /usr/sbin/rmgroup
- /usr/sbin/rmpath
- /usr/sbin/rmrole
- /usr/sbin/rmuser
- /opt/rsct/bin/ctstrtcasd
- /usr/sbin/srcd
- /usr/sbin/srcmstr
- /usr/sbin/rmsock64

- /usr/sbin/sendmail_ssl
- /usr/sbin/sendmail_nonssl
- /usr/sbin/rmsock
- /usr/sbin/sliplogin
- /usr/sbin/sendmail
- /usr/sbin/rwhod
- /usr/sbin/route
- /usr/sbin/snappd
- /usr/sbin/swap
- /usr/sbin/swapoff
- /usr/sbin/swapon
- /usr/sbin/swcons
- /usr/sbin/switch.prt
- /usr/sbin/synclvodm
- /usr/sbin/tsm
- /usr/sbin/umount
- /usr/sbin/umountall
- /usr/sbin/unmount
- /usr/sbin/varyonvg
- /usr/sbin/watch
- /usr/sbin/talkd
- /usr/sbin/timedc
- /usr/sbin/uucpd
- /usr/bin/bellmail
- /usr/bin/at
- /usr/bin/capture
- /usr/bin/chcore
- /usr/bin/acctras
- /usr/bin/acctctl
- /usr/bin/chgroup
- /usr/bin/chkey
- /usr/bin/chque
- /usr/bin/chquedev
- /usr/bin/chrole
- /usr/bin/chsec
- /usr/bin/chuser
- /usr/bin/confsrc
- /usr/bin/crontab
- /usr/bin/enq
- /usr/bin/filemon

- /usr/bin/errpt
- /usr/bin/fileplace
- /usr/bin/fileplacej2
- /usr/bin/fileplacej2_64
- /usr/bin/ftp
- /usr/bin/getconf
- /usr/bin/ipcs
- /usr/bin/ipcs64
- /usr/bin/iostat
- /usr/bin/logout
- /usr/bin/lscore
- /usr/bin/lssec
- /usr/bin/mesg
- /usr/bin/mkgroup
- /usr/bin/mkque
- /usr/bin/mkquedev
- /usr/bin/mkrole
- /usr/bin/mkuser
- /usr/bin/netpmon
- /usr/bin/newgrp
- /usr/bin/pagdel
- /usr/bin/paginit
- /usr/bin/paglist
- /usr/bin/passwd
- /usr/bin/pwck
- /usr/bin/pwdadm
- /usr/bin/pwdck
- /usr/bin/rm_mlcache_file
- /usr/bin/rdist
- /usr/bin/remsh
- /usr/bin/rlogin
- /usr/bin/rexec
- /usr/bin/rcp
- /usr/bin/rmque
- /usr/bin/rmquedev
- /usr/bin/rsh
- /usr/bin/ruptime
- /usr/bin/rwho
- /usr/bin/script
- /usr/bin/setgroups

- /usr/bin/setsenv
- /usr/bin/shell
- /usr/bin/su
- /usr/bin/sysck
- /usr/bin/tcbck
- /usr/bin/sysck_r
- /usr/bin/telnet
- /usr/bin/tftp
- /usr/bin/traceroute
- /usr/bin/tn
- /usr/bin/tn3270
- /usr/bin/usrck
- /usr/bin/utftp
- /usr/bin/vmstat
- /usr/bin/vmstat64
- /usr/bin/yppasswd
- /sbin/helpers/jfs2/backbyinode
- /sbin/helpers/jfs2/diskusg
- /sbin/helpers/jfs2/restbyinode

ユーザー、グループ、およびパスワード

AIX ユーザーおよびグループを管理することができます。

ログイン時の自動ホーム・ディレクトリーの作成

AIX オペレーティング・システムはユーザーのログイン時にホーム・ディレクトリーを自動的に作成します。

このフィーチャーは、ローカル・システムにホーム・ディレクトリーを持たない場合があるリモート側に定義されたユーザー (例えば、LDAP サーバーに定義されたユーザー) にとって便利です。 AIX オペレーティング・システムはユーザーのログイン時にホーム・ディレクトリーを自動的に作成するための 2 つのメカニズム (標準 AIX メカニズムと PAM メカニズム) を提供します。 これらのメカニズムは同時に使用可能にすることができます。

AIX メカニズム

AIX メカニズムは **getty**、**login**、**rlogin**、**rsh**、**telnet**、および **tsm** の各コマンドによるログインを対象としています。 AIX メカニズムは、pam_aix モジュールを使用して STD_AUTH 認証および PAM_AUTH 認証をサポートします。 **/etc/security/login.cfg** ファイルで AIX メカニズムを使用可能にするには、usw スタンザの mkhomeatlogin 属性を `true` に設定します (このファイルの追加情報については、`/etc/security/login.cfg` ファイルを参照してください)。 「ログイン時自動ホーム・ディレクトリー作成 (automatic-home-directory-creation-at-login)」フィーチャーを使用可能または使用不可にする場合は、**chsec** コマンドを使用します。 例えば、このフィーチャーを使用可能にするには、次のコマンドを実行します。

```
# chsec -f /etc/security/login.cfg -s usw -a mkhomeatlogin=true
```

使用可能に設定されると、認証が成功した後にログイン・プロセスでユーザーのホーム・ディレクトリーが検査されます。 ユーザーのホーム・ディレクトリーが存在していない場合は、そのディレクトリーが作成されます。

注: **mkhomeatlogin** 属性は、AIX バージョン 6.1 (6100-02 テクノロジー・レベル適用) 以降でのみサポートされます。

PAM メカニズム
AIX は PAM メカニズム用のホーム・ディレクトリーを作成するために pam_mkuserhome モジュールも提供します。 pam_mkuserhome モジュールは、ログイン・サービスのための他のセッション・モジュールと併せて使用することができます。この PAM モジュールをサービス用として使用可能にするには、そのサービスに 1 つのエントリーを追加する必要があります。 例えば、PAM を使用して **telnet** コマンドからホーム・ディレクトリーの作成を使用可能にするには、次のエントリーを **/etc/pam.cfg** ファイルに追加します。

```
telnet session optional pam_mkuserhome
```

アカウント ID

各ユーザー・アカウントには、そのアカウントを一意的に識別する数字 ID があります。 AIX オペレーティング・システムはアカウント ID に応じて権限を付与します。

同じ ID を持つアカウントは事実上同じアカウントです。このことを理解するのが重要です。 ユーザーおよびグループを作成する場合、AIX **mkuser** コマンドおよび **mkgroup** コマンドは、必ず、ターゲット・レジストリーを調べて、作成されるアカウントが既存のアカウントとの ID の衝突がないことを確認します。

また、アカウント作成時に **dist_uniqid** システム属性を使用してすべてのユーザー・レジストリー (グループ・レジストリー) を検査するように、システムを構成することができます。 **/etc/security/login.cfg** ファイル内の usw スタンザの **dist_uniqid** 属性は、**chsec** コマンドを使用して管理することができます。すべてのレジストリーに対して常に ID の衝突を検査するようにシステムを構成するには、次のコマンドを実行します。

```
# chsec -f /etc/security/login.cfg -s usw -a dist_uniqid=always
```

dist_uniqid 属性には、次の 3 つの有効な値があります。

never この値の場合、非ターゲット・レジストリーに対する ID 衝突の検査は行われません (デフォルト)。

always
この値の場合、他のすべてのレジストリーに対して ID 衝突の検査が行われます。ターゲット・レジストリーと他のいずれかのレジストリー間に衝突が検出された場合、**mkuser** (**mkgroup**) コマンドは、どのレジストリーでも使用されていない固有 ID を選出します。 これが失敗するのは、コマンド・ラインから ID 値を指定した場合だけです (例えば、いずれかのレジストリーで ID 234 を持っているユーザーが既に存在するときに `mkuser id=234 foo` と指定した場合です)。

uniqbyname
この値の場合、他のすべてのレジストリーに対して ID 衝突の検査が行われます。レジストリー間の衝突が許可されるのは、作成されるアカウントが `mkuser id=123 foo` タイプのコマンドに対して既存アカウントと同じ名前を持つ場合のみです。 ID をコマンド・ラインから指定しない場合は、新しいアカウントが別のレジストリー内にあって同じ名前を持つ既存アカウントと同じ ID 値を持つことはありません。 例えば、ID 234 を持つ *acct1* がローカル・アカウントであるとします。 LDAP アカウント *acct1* を作成する場合、`mkuser -R LDAP acct1` は LDAP アカウントに

235 の ID を選出することが可能です。 その結果、ID 234 を持つ *acct1* がローカルに作成され、ID 235 を持つ *acct1* が LDAP に作成されます。

注: ターゲット・レジストリー内の ID 衝突の検出は、**dist_uniqid** 属性にかかわらず常に実行されます。

uniqbyname 値は 2 つのレジストリーに対して適切に機能します。 2 つより多いレジストリーがあり、2 つのレジストリー間に ID 衝突が既に存在しているときに、衝突している ID 値を使用して 3 番目のレジストリーに新しいアカウントを作成した場合、**mkuser** (**mkgroup**) の動作は明確に定められていません。 新しいアカウントの作成の成否はレジストリーが検査される順序に依存します。

例えば、ローカル、LDAP、および DCE という 3 つのレジストリーで構成されているシステムがあるとします。 *acct1* アカウントが LDAP に存在し、*acct2* アカウントが DCE に存在していて、両方とも ID 234 を持っているとします。 システム管理者が `mkuser -R files id=234 acct1` (`mkgroup -R files id=234 acct1`) コマンドを実行して **uniqbyname** 値を持つローカル・アカウントを作成すると、**mkuser** (**mkgroup**) コマンドは最初に LDAP レジストリーを検査し、ID 234 が LDAP アカウント *acct1* に使用されていることを検出します。 作成されるアカウントが同じアカウント名を持っているため、**mkuser** (**mkgroup**) コマンドは ID 234 のローカル・アカウント *acct1* を正常に作成します。 DCE レジストリーが最初に検査される場合は、**mkuser** (**mkgroup**) コマンドは ID 234 が DCE アカウント *acct2* に使用されていることを検出し、ローカル・アカウント *acct1* の作成は失敗します。 ID 衝突の検査により、ローカル・レジストリーと複数のリモート・レジストリー間、または複数のリモート・レジストリー相互間で ID の一意性が強制されます。 リモート・レジストリーに新しく作成されたアカウントと、同じリモート・レジストリーを使用する他のシステムにおける既存ローカル・ユーザーのアカウントの間では、ID の一意性が得られる保証はありません。 **mkuser** (**mkgroup**) コマンドは、コマンドの実行時にリモート・レジストリーに到達できない場合、そのリモート・レジストリーを迂回します。

root アカウント

root アカウントは、システム上のすべてのプログラム、ファイル、およびリソースに対して事実上無制限のアクセスを持っています。

root アカウントは **/etc/passwd** ファイル内で `0` のユーザー ID (UID) を持つ特殊なユーザーで、一般に *root* というユーザー名を与えられます。 root アカウントが特殊であるのはこのユーザー名のためではなく、`0` という UID 値のためです。つまり、`0` という UID を持つユーザーは、だれでも root ユーザーと同じ特権を持ちます。 また、root アカウントは、常に、ローカル・セキュリティー・ファイルによって認証されます。

root アカウントには、必ず、パスワードを指定しておく必要があります。そのパスワードの共有はできません。システムをインストールした直後に、root アカウントにパスワードを指定してください。 システム管理者だけが root パスワードを知っている必要があります。 システム管理者は、root ユーザーとしての操作だけを行い、 root 権限を必要とするシステム管理機能を実行します。 その他の操作は、通常のユーザー・アカウントに戻してください。

重要: root ユーザーとして、繰り返し操作を行うと、システムが損傷する可能性があります。それは、root アカウントがシステム内の数多くの安全機能をオーバーライドするからです。

直接 **root** ログインの使用不可化:

潜在的なハッカーが使う一般的なアタック方法は、root のパスワードを手に入れることです。

この種のアタックを避けるには、root ID への直接アクセスを使用不可にしておき、システム管理者に **su -** コマンドを使用して root 特権を入手するように要求することができます。直接の root アクセスを制限すると、アタック・ポイントとしての root ユーザーを除去できるだけでなく、root アクセスを入手した

ユーザーと、それらのユーザーのアクションの時刻をモニターすることができます。 これは、**/var/adm/sulog** ファイルを表示して実行できます。別の選択肢としては、システム監査を使用可能にすることがあります。これにより、この種のアクティビティーが報告されます。

root ユーザーのリモート・ログイン・アクセスを使用不可にするには、**/etc/security/user** ファイルを編集します。 root のエントリーの rlogin 値として False を指定します。

リモート root ログインを使用不可にするには、その前にシステム管理者が非 root ユーザー ID でログインできないような状況を調べ、それに応じた準備をします。 例えば、ユーザーのホーム・ファイルシステムが満杯の場合、ユーザーはログインできなくなります。 リモート root ログインが使用不可になっていて、かつ、**su -** コマンドを使用して root に変更できるはずのユーザーのホーム・ファイルシステムが満杯の場合、root はシステムの制御を獲得することができません。 この問題を回避するために、システム管理者は、平均的なユーザーのファイルシステムよりも大きいホーム・ファイルシステムを自分用に作成することができます。

ユーザー・アカウント

ユーザー・アカウントに対するいくつかのセキュリティー管理用タスクがあります。

推奨されるユーザー属性**:**

ユーザー管理は、ユーザーとグループの作成、およびそれらの属性の定義から構成されます。

ユーザーの主要な属性は、ユーザーの認証方法です。ユーザーは、システム上の 1 次エージェントです。アクセス権限、環境、認証方法のほかに、ユーザーのアカウントにアクセスする方法、時期、および場所がその属性によって制御されます。

グループは、保護リソースに対してアクセス許可を共用することができるユーザーの集合です。 グループは、ID が指定され、メンバーと管理者で構成されます。 通常は、グループの作成者が第 1 管理者です。

それぞれのユーザー・アカウントごとに、パスワード属性やログイン属性なども含めて、数多くの属性を設定することができます。 構成可能な属性のリストについては、 84 ページの『ディスク・クォータ・システムの概要』を参照してください。 次の属性が推奨されます。

- それぞれのユーザーごとに、他のどのユーザーとも共有されないユーザー ID を指定する必要があります。 セキュリティー安全機能と責任能力ツールは、すべて、それぞれのユーザーに固有の ID が指定されている場合にのみ作動します。
- システム上のユーザーに分かりやすいユーザー名を指定します。大部分の電子メール・システムでは、着信メールにラベルを付けるのにユーザー ID を使用するので、実際の名前が最適です。
- ユーザーの追加、変更、および削除には、SMIT インターフェースを使用します。これらの作業をコマンド・ラインから行うことは可能ですが、SMIT インターフェースは、小さなエラーを減少させる上で役に立ちます。
- ユーザーがシステムにログインできるようになるまで、そのユーザー・アカウントに初期パスワードを指定しないようにします。 パスワード・フィールドが `/etc/passwd` ファイルに `*` (アスタリスク) として定義されている場合、アカウント情報は保持されますが、だれもそのアカウントにログインできません。
- 正しく機能するためにシステムで必要とするシステム定義のユーザー ID は、変更しないようにします。 システム定義のユーザー ID は、**/etc/passwd** ファイルにリストされています。
- 一般に、どのユーザー ID についても、*admin* パラメーターを `true` に設定しないようにします。 `/etc/security/user` ファイルに `admin=true` が設定されているユーザーの属性は、root ユーザーだけが変更できます。

オペレーティング・システムは、/etc/passwd ファイルと /etc/system/group ファイルに収められている次のような標準ユーザー属性をサポートします。

認証情報
　パスワードを指定する

資格情報
　ユーザー ID、プリンシパル・グループ、および補足グループ ID を指定する

環境　ホームまたはシェル環境を指定する

ユーザー名とグループ名の長さの制限**:**

ユーザー名とグループ名の長さの制限を構成して取得できます。

ユーザー名とグループ名の長さの制限パラメーターのデフォルト値は 9 文字です。 AIX 5.3 以降では、ユーザー名とグループ名の長さの制限を 9 文字から 256 文字に増やすことができます。 ユーザー名とグループ名の長さの制限パラメーターには終了 NULL 文字があるため、実際の有効な名前の長さは 8 文字から 255 文字までです。

ユーザー名とグループ名の長さの制限を指定するには、sys0 デバイスに対する **v_max_logname** システム構成パラメーターを使用します。 カーネルまたは ODM データベースから **v_max_logname** パラメーター値を変更または取得できます。 カーネルのパラメーター値は、システムが実行中に使用する値です。 ODM データベースのパラメーター値は、システムが次回の再起動後に使用する値です。

注**:** ユーザー名とグループ名の長さの制限を増やした後にこれを減らした場合は、予期しない動作が起こる可能性があります。 大きな制限で作成したユーザー名とグループ名がシステムに引き続き存在する可能性があります。

ODM データベースからユーザーおよびグループ名の長さ制限の取得**:**

v_max_logname パラメーターを取り出す場合は、コマンドまたはサブルーチンを使用します。

ODM データベースの **v_max_logname** パラメーターを取り出す場合は、 **lsattr** コマンドを使用します。 **lsattr** コマンドは **v_max_logname** パラメーターを max_logname 属性として表示します。

詳細情報については、「*Commands Reference, Volume 3*」の『**lsattr** コマンド』を参照してください。

以下の例は **lsattr** コマンドを使用して、 max_logname 属性を取り出す方法を示しています。

```
$ lsattr -El sys0
SW_dist_intr    false           Enable SW distribution of interrupts             True
autorestart     true            Automatically REBOOT system after a crash        True
boottype        disk            N/A                                              False
capacity_inc    1.00            Processor capacity increment                     False
capped          true            Partition is capped                              False
conslogin       enable          System Console Login                             False
cpuguard        enable          CPU Guard                                        True
dedicated       true            Partition is dedicated                           False
ent_capacity    4.00            Entitled processor capacity                      False
frequency       93750000        System Bus Frequency                             False
fullcore        false           Enable full CORE dump                            True
fwversion       IBM,SPH01316    Firmware version and revision levels             False
iostat          false           Continuously maintain DISK I/O history           True
keylock         normal          State of system keylock at boot time             False
max_capacity    4.00            Maximum potential processor capacity             False
max_logname     20              Maximum login name length at boot time           True
maxbuf          20              Maximum number of pages in block I/O BUFFER CACHE True
maxmbuf         0               Maximum Kbytes of real memory allowed for MBUFS  True
```

```
maxpout         0               HIGH water mark for pending write I/Os per file   True
maxuproc        128             Maximum number of PROCESSES allowed per user      True
min_capacity    1.00            Minimum potential processor capacity              False
minpout         0               LOW water mark for pending write I/Os per file    True
modelname       IBM,7044-270    Machine name                                      False
ncargs          6               ARG/ENV list size in 4K byte blocks               True
pre430core      false           Use pre-430 style CORE dump                       True
pre520tune      disable         Pre-520 tuning compatibility mode                 True
realmem         3145728         Amount of usable physical memory in Kbytes        False
rtasversion     1               Open Firmware RTAS version                        False
sec_flags       0               Security Flags                                    True
sed_config      select          Stack Execution Disable (SED) Mode                True
systemid        IBM,0110B5F5F   Hardware system identifier                        False
variable_weight 0               Variable processor capacity weight                False
$
```

カーネルからユーザーおよびグループ名の長さ制限の取得*:*

カーネルから **v_max_logname** パラメーターを取り出す場合は、コマンドおよびサブルーチンを使用します。

getconf コマンドの使用

LOGIN_NAME_MAX パラメーターを指定した **getconf** コマンドを使用して、カーネル内のユーザーおよびグループ名の長さ制限を取り出すことができます。 **getconf** コマンド出力には終了ヌル文字が含まれます。

以下の例は **getconf** コマンドを使用して、カーネルからユーザーおよびグループ名の長さ制限を取り出す方法を示しています。

```
$ getconf LOGIN_NAME_MAX
20
$
```

sysconf サブルーチンの使用

_SC_LOGIN_NAME_MAX パラメーターを指定した sysconf サブルーチンで、カーネル内のユーザーおよびグループ名の長さ制限を取り出すことができます。

以下の例は sysconf サブルーチンを使用して、カーネルからユーザーおよびグループ名の長さ制限を取り出す方法を示しています。

```
#include <unistd.h>
main()
{
       long len;

       len = sysconf(_SC_LOGIN_NAME_MAX);

       printf("The name length limit is %d¥n", len);
}
```

sys_parm サブルーチンの使用

SYSP_V_MAX_LOGNAME パラメーターを指定した sys_parm サブルーチンを使用して、カーネル内の現在のユーザー名の長さ制限を取り出すことができます。

以下の例は sys_parm サブルーチンを使用して、カーネルからユーザー名の長さ制限を取り出す方法を示しています。

```
#include <sys/types.h>
#include <sys/var.h>
#include <errno.h>
main()
{
   int rc;
   struct vario myvar;

   rc = sys_parm (SYSP_GET, SYSP_V_MAX_LOGNAME, &myvar);

   if (!rc)
      printf("Max_login_name = %d¥n", myvar.v.v_max_logname.value);
   else
      printf("sys_parm() failed rc = %d, errno = %d¥n", rc, errno);
}
```

ODM データベースでのユーザー・グループおよび名前の長さ制限の変更**:**

カーネル内でのユーザーおよびグループ名の長さ限界値は、システム・ブート段階でのみ構成できます。ODM データベースの値は **chdev** コマンドを使用して変更できます。 変更は次のシステム再起動により有効になります。

chdev コマンドを使用して、ODM データベースの **v_max_logname** パラメーターを変更する例を以下に示します。

```
$ chdev -l sys0 -a max_logname=30
sys0 changed
$
```

ユーザー・アカウント制御**:**

ユーザー・アカウントには変更できる属性があります。

それぞれのユーザー・アカウントごとに、1 組の属性が関連付けられます。 これらの属性は、**mkuser** コマンドを使用してユーザーを作成するときに、デフォルト値から作成されます。 これらの属性は、**chuser** コマンドを使用して変更することができます。 以下は、ログインを制御し、パスワードの品質には関連しないユーザー属性です。

account_locked
: アカウントを明示的にロックする必要がある場合、この属性を True に設定できます。デフォルトは False です。

admin
: True に設定すると、このユーザーはパスワードを変更できません。 変更できるのは管理者だけです。

admgroups
: このユーザーが管理権限を持つグループをリストします。 これらのグループに関しては、ユーザーはメンバーを追加したり削除したりできます。

auth1 ユーザー・アクセス権限を付与するのに使用される認証方式。通常は SYSTEM に設定されます。その場合、より新しいメソッドが使用されます。

注**: auth1** 属性は推奨されない属性であり、使用すべきではありません。

auth2 **auth1** で指定されたメソッドによってユーザーが認証された後に実行されるメソッド。 システムへのアクセスをブロックすることはできません。通常は NONE に設定されます。

注**: auth2** 属性は推奨されない属性であり、使用すべきではありません。

daemon
このブール・パラメーターは、ユーザーが **startsrc** コマンドを使用してデーモンまたはサブシステムを開始できるようにするかどうかを指定します。また、cron および at 機能の使用も制限します。

login このユーザーがログインできるかどうかを指定します。 ログインが成功すると、**unsuccessful_login_count** 属性が 0 の値 (**loginsuccess** サブルーチンから) にリセットされます。

logintimes
ユーザーがいつログインできるかを制限します。 例えば、ユーザーによるシステムへのアクセスを正規の勤務時間内だけに制限します。

registry
ユーザー・レジストリーを指定します。 システムに、ユーザー情報として、NIS、LDAP、または Kerberos のような代替レジストリーについて通知するために使用できます。

rlogin 指定されたユーザーが **rlogin** コマンドまたは **telnet** コマンドを使用してログインできるかどうかを指定します。このログイン属性は、リモート・ログインを制御するだけです。個別のリモート・コマンドを実行する機能の制御については、rcmds を参照してください。

su 他のユーザーが **su** コマンドを使用してこの ID に切り替えることができるかどうかを指定します。

sugroups
このユーザー ID を切り替えることができるグループを指定します。

ttys ある一定のアカウントを物理的に保護されている領域に制限します。

expires
学習者アカウントやゲスト・アカウントを管理します。これを使用して、アカウントを一時的にオフにすることも可能です。

loginretries
ユーザー ID がシステムによってロックされる前に、連続して失敗するログインの試行の最大数を指定します。 失敗した試行は **/etc/security/lastlog** ファイルに記録されます。

umask
ユーザーの初期 **umask** を指定します。

rcmds 指定されたユーザーが、**rsh** コマンドまたは **rexec** コマンドを使用して個別コマンドを実行できるかどうかを指定します。値 `allow` は、**rsh** コマンドまたは **rexec** コマンドを使用してリモート側でコマンドを実行できることを示します。値 `deny` は、リモート側でコマンドを実行できないことを示します。値 `hostlogincontrol` は、実行するリモート・コマンドが **hostallowedlogin** 属性および **hostsdeniedlogin** 属性によって制御されることを示します。リモート・ログインの制御については、 rlogin 属性を参照してください。

hostallowedlogin
ユーザーのログインを許可するホストを指定します。 この属性は、ユーザー属性が複数のホストによって共有されるネットワーク環境で使用されることを目的としています。

hostsdeniedlogin
ユーザーのログインを許可しないホストを指定します。 この属性は、ユーザー属性が複数のホストによって共有されるネットワーク環境で使用されることを目的としています。

maxulogs
ユーザー当たりのログインの最大回数を指定します。 ユーザーがログインの最大許容回数に達すると、ログインは拒否されます。

ユーザー属性の完全セットは、/etc/security/user、/etc/security/limits、/etc/security/audit/config、および /etc/security/lastlog ファイルに定義されます。 **mkuser** コマンドを使用するユーザー作成のデフォルトは、 **/usr/lib/security/mkuser.default** ファイルで指定されます。 **mkuser.default** ファイルには、監査クラスのほかに、**/etc/security/user** ファイルおよび **/etc/security/limits** ファイルの default スタンザの一般デフォルトをオーバーライドするオプションのみを指定する必要があります。これらのうちのいくつかの属性がユーザーのログイン方法を制御しますが、これらの属性は、指定された条件のもとで、ユーザー・アカウントを自動的にロックする (今後のログインを防止する) よう構成することができます。

ログイン試行が何回も失敗したためにユーザー・アカウントがシステムによってロックされると、システム管理者が **/etc/security/lastlog** ファイル内のそのユーザーの **unsuccessful_login_count** 属性をログイン再試行回数より小さい値に再設定するまで、そのユーザーはログインできません。 これは **chsec** コマンドを次のように使用して行うことができます。

```
chsec -f /etc/security/lastlog -s username -a
unsuccessful_login_count=0
```

デフォルトを変更するには、**chsec** コマンドを使用して、/etc/security/user ファイルまたは /etc/security/limits ファイルなど、該当するセキュリティー・ファイルのデフォルトのスタンザを編集します。 多くのデフォルトは、標準の動作をするように定義されています。 新規ユーザーが作成されるたびに設定される属性を明示的に指定するには、**/usr/lib/security/mkuser.default** の *user* エントリーを変更します。

拡張ユーザー・パスワード属性については、 72 ページの『パスワード』を参照してください。

ユーザー属性の影響を受ける、ログインに関連したコマンド

以下の表は、ログインを制御する属性と影響を受けるコマンドをリストしたものです。

ユーザー属性	コマンド
account_locked	**rexec**、**rsh**、**rcp**、**ssh**、**scp**、**rlogin**、**telnet**、**ftp**、**login**
login	コンソールからのログインにのみ影響します。 **login** 属性の値は、リモート・ログイン・コマンド、リモート・シェル・コマンド、またはリモート・コピー・コマンド (**rexec**、**rsh**、 **rcp**、**ssh**、**scp**、**rlogin**、**telnet**、および **ftp**) には影響しません。
logintimes	**rexec**、**rsh**、**rcp**、**ssh**、**scp**、**rlogin**、**telnet**、**ftp**、**login**
rlogin	リモート・ログイン・コマンド、特定のリモート・シェル・コマンド、および特定のリモート・コピー・コマンド (**ssh**、**scp**、**rlogin**、および **telnet**) にのみ影響します。
loginretries	**rexec**、**rsh**、**rcp**、**ssh**、**scp**、**rlogin**、**telnet**、**ftp**、**login**
/etc/nologin	**rexec**、**rsh**、**rcp**、**ssh**、**scp**、**rlogin**、**telnet**、**ftp**、**login**
rcmds=deny	**rexec**、**rsh**、**rcp**、**ssh**、**scp**
rcmds=hostlogincontrol および **hostsdeniedlogin=<target_hosts>**	**rexec**、**rsh**、**rcp**、**ssh**、**scp**、**rlogin**、**telnet**、**ftp**、**login**
ttys = !REXEC、**!RSH**	**rexec**、**rsh**、**rcp**、**ssh**、**scp**、**rlogin**、**telnet**、**ftp**、**login**
ttys = !REXEC、**!RSH**、**/dev/pts**	**rexec**、**rsh**
ttys = !REXEC、**!RSH**、**ALL**	**rexec**、**rsh**
expires	**rexec**、**rsh**、**rcp**、**ssh**、**scp**、**rlogin**、**telnet**、**ftp**、**login**

注**: rsh** は、リモート・コマンドの実行のみを不許可にします。 リモート・ログインは許可されます。

関連情報:

loginsuccess サブルーチン

rexec コマンド

rsh コマンド

startsrc コマンド

su コマンド

ログイン・ユーザー **ID:**

オペレーティング・システムは、ログイン・ユーザー ID によってユーザーを識別します。

ログイン・ユーザー ID により、システムはそのソースに対するすべてのユーザー処置をトレースできます。 ユーザーがシステムにログインした後、初期ユーザー・プログラムが実行される前に、システムは、ユーザー・データベースにあるユーザー ID に、プロセスのログイン ID を設定します。 ログイン・セッション中のすべての後続のプロセスは、この ID によりタグ付けされます。 これらのタグは、ログイン・ユーザー ID により実行されたすべてのアクティビティーの証跡を提供します。 ユーザーはセッションの途中で、実効ユーザー ID、実ユーザー ID、実効グループ ID、実グループ ID、および補足グループ ID を再設定できますが、ログイン・ユーザー ID を変更することはできません。

アクセス制御リストによるユーザー・セキュリティーの強化**:**

システムで、適切なレベルのセキュリティーを達成するには、ユーザー・アカウントを管理するために一貫したセキュリティー・ポリシーを開発する必要があります。 最も良く使用されるセキュリティー・メカニズムは、アクセス制御リスト (ACL) です。

ACL およびセキュリティー・ポリシーの開発に関する詳細は、 134 ページの『アクセス制御リスト』を参照してください。

PATH 環境変数**:**

PATH 環境変数は、重要なセキュリティー管理の要素です。 この変数では、コマンドを見つけるために検索するディレクトリーを指定します。

デフォルトのシステム共通 **PATH** 値は **/etc/profile** ファイルに指定され、各ユーザーは、通常、そのユーザーの **$HOME/.profile** ファイルに **PATH** 値を持っています。 **.profile** ファイルにある **PATH** 値は、システム共通 **PATH** 値をオーバーライドするか、あるいは、エクストラ・ディレクトリーを追加します。

PATH 環境変数に無許可変更を行うと、システム上のユーザーが他のユーザー (root ユーザーを含む) を「スプーフ」できるようになります。 スプーフィング・プログラム (トロイの木馬 プログラムとも呼ばれる) は、システム・コマンドを置換してから、そのコマンドで扱われる情報 (ユーザー・パスワードなど) を取り込みます。

例えば、あるユーザーが、コマンドの実行時にシステムが **/tmp** ディレクトリーを最初に検索するように **PATH** 値を変更したとします。 その上で、そのユーザーは `/tmp` ディレクトリーに **su** という名前のプログラムを入れておきます。このプログラムは **su** コマンドと同じように root パスワードを求めます。次に **/tmp/su** プログラムは root パスワードをかのユーザーにメールし、本物の **su** コマンドを呼び出しておいてから終了します。 このシナリオでは、**su** コマンドを使用した root ユーザーは常にその root パスワードを漏らすことになり、しかもそのことに気が付きません。

システム管理者およびユーザーに **PATH** 環境変数の問題が起きないようにするには、次のコマンドを実行します。

- 疑わしい場合は、絶対パス名を指定します。 絶対パス名が指定されると、**PATH** 環境変数は無視されます。
- root ユーザー用に指定する **PATH** 値には現行ディレクトリー (**.** (ピリオド) で指定する) を入れないようにします。 現行ディレクトリーを **/etc/profile** 内に指定しないようにします。
- root ユーザーは、専用の **.profile** ファイルに自分用の **PATH** 指定を入れておく必要があります。 **/etc/profile** 内の指定では、通常、すべてのユーザー向けの最小標準がリストされますが、root ユーザーが必要とするディレクトリーはデフォルトより多い場合も少ない場合もあります。
- 他のユーザーに対して、システム管理者に相談せずに自分の **.profile** ファイルを変更しないように警告します。 これを怠ると、信用していたユーザーが、予期せぬアクセスを許すような変更を行う可能性があります。 ユーザーの **.profile** ファイルには 740 に設定されたアクセス権を与える必要があります。
- システム管理者は、ユーザー・セッションから root 権限を獲得するために **su** コマンドを使用すべきではありません。それは、**.profile** ファイル内に指定されているそのユーザーの **PATH** 値が有効であるためです。ユーザーは自分自身の **.profile** ファイルを設定できます。 システム管理者は、そのユーザーのマシンに root ユーザーとしてログインするか、できれば自分の ID でログインして、次のコマンドを使用してください。

  ```
  /usr/bin/su - root
  ```

  これによって、そのセッションでは、root の環境が確実に使用されるようになります。 システム管理者は、別のユーザーのセッションで root として操作する場合、そのセッションでは、一貫して絶対パス名を指定する必要があります。
- 入力フィールド・セパレーター (**IFS**) 環境変数が **/etc/profile** ファイル内で変更されないように保護されます。 **.profile** ファイルの **IFS** 環境変数を使用して **PATH** 値を変更することが可能です。

secldapclntd デーモンの使用**:**

secldapclntd デーモンは LDAP サーバーへの接続を動的に管理します。

secldapclntd デーモンは開始されるときに **/etc/security/ldap/ldap.cfg** ファイルに定義されたサーバーに接続されます (LDAP サーバー当たり 1 接続)。 その後、**secldapclntd** デーモンは、LDAP 接続のために LDAP の処理要求が制限されていることが分かると、このデーモンは、自動的に、現在の LDAP サーバーに別の接続を確立することになります。 この処理は事前定義された接続の最大数に達するまで続きます。 接続の最大数に達すると、新規の接続は追加されません。

secldapclntd デーモンは、現在の LDAP サーバーへのすべての接続を定期的に検査します。 最初の接続以外のいずれの接続も事前定義の期間だけアイドル状態になれば、デーモンはその接続を閉じます。

/etc/security/ldap/ldap.cfg ファイルの connectionsperserver 変数は接続の最大数として使用されます。 しかし、connectionsperserver 変数が numberofthread 変数より大きい場合、**secldapclntd** デーモンは connectionsperserver 値を numberofthread 値に設定します。 connectionsperserver 変数の有効値は 1 から 100 の範囲です。 デフォルト値は 10 です (`connectionsperserver: 10`)。

/etc/security/ldap/ldap.cfg ファイルの connectionmissratio 変数は、新規の LDAP 接続数を設定するための基準です。 connectionmissratio 変数は、最初の試行中に LDAP 接続の獲得に失敗した (処理ミス) 操作の割合です。 失敗した試行回数が connectionmissratio 変数より大きい場合、**secldapclntd** デーモンは新規に LDAP 接続を確立して LDAP 照会を (connectionsperserver 変数に定義済みの接続数を超えない範囲で) 強化します。 connectionmissratio 変数の有効値は 10 から 90 の範囲です。 デフォルト値は 50 です (`connectionmissratio: 50`)。

/etc/security/ldap/ldap.cfg ファイルの connectiontimeout 変数は、接続が **secldapclntd** デーモンによってクローズされるまで、アイドルのままで残ることができる期間の指定として使用されます。connectiontimeout 変数に対する有効値は 5 秒以上です (上限はありません)。デフォルト値は 300 秒です (connectiontimeout: 300)。

セキュア・ユーザー・アカウントでの匿名 FTP のセットアップ

セキュア・ユーザー・アカウントで匿名 FTP をセットアップすることができます。

このシナリオでは、コマンド・ライン・インターフェースおよびスクリプトを使用して、セキュア・ユーザー・アカウントで匿名 FTP をセットアップします。

1. 以下のコマンドを入力して、システムに bos.net.tcp.client ファイルセットがインストールされているかどうかを検査します。

   ```
   lslpp -L | grep bos.net.tcp.client
   ```

   コマンド出力が何もなかった場合、ファイルセットはインストールされていません。 インストール方法については、「インストールおよび移行」を参照してください。
2. root 権限によって、/usr/samples/tcpip ディレクトリーに移動します。次に例を示します。

   ```
   cd /usr/samples/tcpip
   ```
3. アカウントをセットアップするには、次のスクリプトを実行します。

   ```
   ./anon.ftp
   ```
4. 「Are you sure you want to modify /home/ftp?」というプロンプトが出されたら、yes と入力します。 次のような出力が表示されます。

   ```
   Added user anonymous.
   Made /home/ftp/bin directory.
   Made /home/ftp/etc directory.
   Made /home/ftp/pub directory.
   Made /home/ftp/lib directory.
   Made /home/ftp/dev/null entry.
   Made /home/ftp/usr/lpp/msg/en_US directory.
   ```
5. /home/ftp ディレクトリーに移動します。次に例を示します。

   ```
   cd /home/ftp
   ```
6. 次のように入力して、home サブディレクトリーを作成します。

   ```
   mkdir home
   ```
7. 次のように入力して、/home/ftp/home ディレクトリーのアクセス権を drwxr-xr-x に変更します。

   ```
   chmod 755 home
   ```
8. 次のように入力して、/home/ftp/etc ディレクトリーに移動します。

   ```
   cd /home/ftp/etc
   ```
9. 次のように入力して、objrepos サブディレクトリーを作成します。

   ```
   mkdir objrepos
   ```
10. 次のように入力して、/home/ftp/etc/objrepos ディレクトリーのアクセス権を drwxrwxr-x に変更します。

    ```
    chmod 775 objrepos
    ```
11. 次のように入力して、/home/ftp/etc/objrepos ディレクトリーの所有者とグループを root ユーザーとシステム・グループに変更します。

    ```
    chown root:system objrepos
    ```
12. 次のように入力して、security サブディレクトリーを作成します。

```
mkdir security
```

13. 次のように入力して、/home/ftp/etc/security ディレクトリーのアクセス権を drwxr-x--- に変更します。

```
chmod 750 security
```

14. 次のように入力して、/home/ftp/etc/security ディレクトリーの所有者とグループを、root ユーザーとセキュリティー・グループに変更します。

```
chown root:security security
```

15. 次のように入力して、/home/ftp/etc/security ディレクトリーに移動します。

```
cd security
```

16. 次の SMIT 高速パスを入力して、ユーザーを追加します。

```
smit mkuser
```

このシナリオでは、test という名前のユーザーを追加しています。

17. SMIT フィールドに以下の値を入力します。

```
User NAME                                          [test]
ADMINISTRATIVE USER?                                true
Primary GROUP                                      [staff]
Group SET                                          [staff]
Another user can SU TO USER?                        true
HOME directory                                     [/home/test]
```

変更を入力したら、Enter を押してユーザーを作成します。 SMIT プロセスが完了したら、SMIT を終了します。

18. 次のコマンドで、このユーザーのパスワードを作成します。

```
passwd test
```

プロンプトが出されたら、希望するパスワードを入力します。 確認のために、新規パスワードをもう一度入力しなければなりません。

19. 次のように入力して、/home/ftp/etc ディレクトリーに移動します。

```
cd /home/ftp/etc
```

20. 次のコマンドを使って、/etc/passwd ファイルを /home/ftp/etc/passwd ファイルにコピーします。

```
cp /etc/passwd /home/ftp/etc/passwd
```

21. 好きなエディターを使って、/home/ftp/etc/passwd ファイルを編集します。次に例を示します。

```
vi passwd
```

22. コピーした内容から、root、ftp、およびテスト・ユーザー以外のすべての行を除去します。 編集後、内容は次のようになります。

```
root:!:0:0::/:/bin/ksh
ftp:*:226:1::/home/ftp:/usr/bin/ksh
test:!:228:1::/home/test:/usr/bin/ksh
```

23. 変更を保存し、エディターを終了します。

24. 次のように入力して、/home/ftp/etc/passwd ファイルのアクセス権を -rw-r--r-- に変更します。

```
chmod 644 passwd
```

25. 次のように入力して、/home/ftp/etc/passwd ファイルの所有者とグループを、root ユーザーとセキュリティー・グループに変更します。

```
chown root:security passwd
```

26. 次のコマンドを使って、/etc/security/passwd ファイルの内容を /home/ftp/etc/security/passwd ファイルにコピーします。

    ```
    cp /etc/security/passwd /home/ftp/etc/security/passwd
    ```

27. 好きなエディターを使って、/home/ftp/etc/security/passwd ファイルを編集します。次に例を示します。

    ```
    vi ./security/passwd
    ```

28. コピーした内容から、テスト・ユーザーに関するスタンザ以外のすべてのスタンザを除去します。
29. テスト・ユーザー・スタンザから flags = ADMCHG 行を除去します。 編集後、内容は次のようになります。

    ```
    test:
            password = 2HaAYgpDZX3Tw
            lastupdate = 990633278
    ```

30. 変更を保存し、エディターを終了します。
31. 次のように入力して、/home/ftp/etc/security/passwd ファイルのアクセス権を -rw------- に変更します。

    ```
    chmod 600 ./security/passwd
    ```

32. 次のように入力して、/home/ftp/etc/security/passwd ファイルの所有者とグループを、root ユーザーとセキュリティー・グループに変更します。

    ```
    chown root:security ./security/passwd
    ```

33. 好きなエディターを使用して、/home/ftp/etc/group ファイルを作成および編集します。次に例を示します。

    ```
    vi group
    ```

34. 次の行をファイルに追加します。

    ```
    system:*:0:
    staff:*:1:test
    ```

35. 変更を保存し、エディターを終了します。
36. 次のように入力して、/home/ftp/etc/group ファイルのアクセス権を -rw-r--r-- に変更します。

    ```
    chmod 644 group
    ```

37. 次のように入力して、/home/ftp/etc/group ファイルの所有者とグループを、root ユーザーとセキュリティー・グループに変更します。

    ```
    chown root:security group
    ```

38. 好きなエディターを使用して、/home/ftp/etc/security/group ファイルを作成および編集します。次に例を示します。

    ```
    vi ./security/group
    ```

39. 次の行をファイルに追加します。

    ```
    system:
     admin = true
    staff
     admin = false
    ```

40. 変更を保存し、エディターを終了します。 これを行うには、以下のステップを実行します。
    a. 次のように入力して、/etc/security/user ファイルを /home/ftp/etc/security ディレクトリーにコピーします。

       ```
       cp /etc/security/user /home/ftp/etc/security
       cd /home/ftp/etc/
       ```

b. エディターを使用し、次のように入力して、コピーした内容から、test ユーザーに関するスタンザ以外のすべてのスタンザを除去します。

```
vi ./security/user
```

c. 変更を保存し、エディターを終了します。

41. 次のように入力して、/home/ftp/etc/security/group ファイルのアクセス権を -rw-r----- に変更します。

```
chmod 640 ./security/group
```

42. 次のように入力して、/home/ftp/etc/security/group ファイルの所有者とグループを、root ユーザーとセキュリティー・グループに変更します。

```
chown root:security ./security/group
```

43. 次のコマンドを使って、適切な内容を /home/ftp/etc/objrepos ディレクトリーにコピーします。

```
cp /etc/objrepos/CuAt ./objrepos
cp /etc/objrepos/CuAt.vc ./objrepos
cp /etc/objrepos/CuDep ./objrepos
cp /etc/objrepos/CuDv ./objrepos
cp /etc/objrepos/CuDvDr ./objrepos
cp /etc/objrepos/CuVPD ./objrepos
cp /etc/objrepos/Pd* ./objrepos
```

44. 次のように入力して、/home/ftp/home ディレクトリーに移動します。

```
cd ../home
```

45. 次のように入力して、ユーザー用に新しいホーム・ディレクトリーを作成します。

```
mkdir test
```

これが新しい ftp ユーザーのホーム・ディレクトリーになります。

46. 次のように入力して、/home/ftp/home/test ディレクトリーの所有者とグループを test ユーザーとスタッフ・グループに変更します。

```
chown test:staff test
```

47. 次のように入力して、/home/ftp/home/test ファイルのアクセス権を -rwx------ に変更します。

```
chmod 700 test
```

48. 次のように入力して、テスト・ユーザーのリモート・ログインとコンソール・ログインを使用不可にします。

```
chuser login=false rlogin=false test
```

この時点で、ftp サブログインがマシンに設定されました。以下の手順で、これをテストできます。

1. ftp を使用して、test ユーザーを作成したホストに接続します。次に例を示します。

```
ftp MyHost
```

2. anonymous としてログインします。 パスワードを求めるプロンプトが出されたら、Enter を押します。
3. 次のコマンドを使って、新規に作成された test ユーザーに切り替えます。

```
user test
```

パスワードを求めるプロンプトが出されたら、ステップ 18 (65 ページ) で作成したパスワードを使用します。

4. **pwd** コマンドを使用して、ユーザーのホーム・ディレクトリーが存在するかどうかを検査します。次に例を示します。

```
ftp> pwd
     /home/test
```

この出力では、/home/test が ftp サブディレクトリーとして示されています。 ホスト上の絶対パス名は、実際には /home/ftp/home/test です。

注:

- ftp サブユーザーに対してのみユーザーを切り替えることができます。 例えば、test は ftp サブユーザーです。
- スクリプト anon.users.ftp を使用して ftp anonymous ユーザーを作成すると、スクリプトの *username* を置き換えて、任意の名前をユーザーに割り当てることができます。
- anonymous ユーザーの場合、サーバーがユーザー・アカウントのホーム・ディレクトリーで **chroot** コマンドを実行するので、*fileftpaccess.ctl* などの構成に関連したすべてのファイルは、それぞれの匿名ユーザーの ~/etc/ などのホーム・ディレクトリー内になければなりません。/etc/ftpaccess.ctl ファイルでの「書き込み専用」、「読み取り専用」、および「読み取り/書き込み」の制約事項として、chrooted パスに関連するパスがある必要があります。

詳細情報の参照先:

- セキュリティーの『TCP/IP Security』
- コマンド・リファレンスの『**ftp** コマンド』

システム特殊ユーザー・アカウント

AIX には、root アカウントおよびシステム・アカウントがすべてのオペレーティング・システム・ファイルおよびファイルシステムを所有するのを防止するための、システム特殊ユーザー・アカウントのデフォルト・セットがあります。

重要: システム特殊ユーザー・アカウントを除去する際には注意が必要です。 **/etc/security/passwd** ファイルの対応する行の最初にアスタリスク (*) を挿入すると、特定のアカウントを使用不可にすることができます。しかし、root ユーザー・アカウントを使用不可にしないよう気を付けてください。 システム特殊ユーザー・アカウントを除去したり、root アカウントを使用不可にしたりすると、オペレーティング・システムが機能しなくなります。

以下のアカウントがオペレーティング・システムで事前定義されています。

adm adm ユーザー・アカウントは、次の基本的なシステム機能を持っています。

- 診断機能。そのためのツールが **/usr/sbin/perf/diag_tool** ディレクトリーに保管されています。
- アカウンティング機能。そのためのツールが以下のディレクトリーに保管されています。
  - **/usr/sbin/acct**
  - **/usr/lib/acct**
  - **/var/adm**
  - **/var/adm/acct/fiscal**
  - **/var/adm/acct/nite**
  - **/var/adm/acct/sum**

bin bin ユーザー・アカウントは、通常、ほとんどのユーザー・コマンドの実行可能ファイルを所有します。 このアカウントの主要な目的は、すべてが root および sys ユーザー・アカウントだけに所有されないよう、重要なシステム・ディレクトリーおよびファイルの所有権を分散するのを援助することです。

daemon

daemon ユーザー・アカウントは、システム・サーバー・プロセス、およびそれに関連したファイルを所有し、実行するためだけに存在します。 このアカウントは、そのようなプロセスが適切なファイル・アクセス権を使って実行することを保証します。

nobody

nobody ユーザー・アカウントは、ネットワーク・ファイルシステム (NFS) により、リモート印刷を可能にするために使用されます。 このアカウントは、プログラムが root ユーザーへの root アクセスを一時的に許可するためのものです。 例えば、セキュア RPC またはセキュア NFS を使用可能にする前に、マスター NIS サーバー上の **/etc/public** 鍵を検査して、公開鍵および秘密鍵が割り当てられていないユーザーを見つけます。 root ユーザーとして、次のように入力すると、未割り当てのユーザーごとにデータベースにエントリーを作成できます。

```
newkey -u username
```

あるいは、nobody ユーザー・アカウント用にデータベースにエントリーを作成すると、どのユーザーでも、root としてログインせずに **chkey** プログラムを実行して、独自のエントリーをデータベース中に作成することができます。

root root ユーザー・アカウント UID 0。これを使ってシステム保守タスク、およびシステム上の問題のトラブルシューティングを実行できます。

sys sys ユーザーは、分散ファイル・サービス (DFS) キャッシュ用のデフォルトのマウント・ポイントを所有します。このマウント・ポイントは、クライアントに DFS をインストールまたは構成するより前に存在している必要があります。 **/usr/sys** ディレクトリーにもインストール・イメージを保管できます。

system

system グループは、システム管理者用のシステム定義グループです。 system グループのユーザーには、root 権限がなくても一部のシステム保守タスクを実行できる特権があります。

不要なデフォルト・ユーザー・アカウントの除去**:**

オペレーティング・システムのインストール中に、デフォルト・ユーザー ID およびグループ ID がいくつも作成されます。 システム上で実行されているアプリケーション、およびネットワーク内でのネットワークの位置によっては、これらのユーザーおよびグループ ID のいくつかがセキュリティー上の弱点になり、悪用の対象になる可能性があります。

次の表には、除去できる可能性のある最も一般的なデフォルト・ユーザー ID がリストされています。

表 *3.* 除去できる可能性のある一般的なデフォルト・ユーザー *ID*

ユーザー **ID**	説明
uucp、nuucp	uucp プロトコルが使用する隠しファイルの所有者。 uucp ユーザー・アカウントは、Unix-to-Unix Copy Program 用に使用されます。このプログラムは 1 群のコマンド、プログラム、およびファイルの集まりで、大部分の AIX システムに存在します。このプログラムにより、ユーザーは専用回線または電話回線を通して別の AIX システムと通信できます。
lpd	印刷サブシステムが使用するファイルの所有者
guest	アカウントへのアクセスがないユーザーへのアクセスを許可します。

次の表は、必要ないかもしれない一般的なグループ ID をリストしています。

表 4. 必要ないかもしれない一般的なグループ *ID*

グループ **ID**	説明
uucp	uucp および nuucp ユーザーの所属先のグループ
printq	lpd ユーザーの所属先のグループ

システムを分析し、本当に必要ない ID を判別します。 必要ないユーザーおよびグループ ID がまだあるかもしれません。 システムを実動させる前に、使用可能な ID の徹底的な評価を行ってください。

注**:** プリンター・ファイルセットへの依存のために `printq` グループを除去する代わりに、`/etc/inittab` エントリー内で `lp` ユーザー ID、**piobe** コマンド、および `qdaemon` プログラムを無効にして、セキュリティー・リスクを最小化します。これによりユーザーは、**print** コマンドを実行できなくなります。

セキュリティー・コンポーネントによって作成されるアカウント**:**

LDAP や OpenSSH などのセキュリティー・コンポーネントがインストールまたは構成されると、ユーザーおよびグループ・アカウントが作成されます。

作成されるユーザーおよびグループ・アカウントには以下が含まれます。

- インターネット・プロトコル **(IP)** のセキュリティー**:** IP セキュリティーは、インストール中にユーザー *ipsec* およびグループ *ipsec* を追加します。 これらの ID は、鍵管理サービスによって使用されます。 `/usr/lpp/group.id.keymgt` にあるグループ ID は、インストール前にカスタマイズすることはできませんので注意してください。
- **Kerberos** および **Public Key Infrastructure (PKI):** これらのコンポーネントは、新規ユーザーまたはグループ・アカウントを作成しません。
- **LDAP:** LDAP クライアントまたはサーバーがインストールされると、*ldap* ユーザー・アカウントが作成されます。 *ldap* のユーザー ID は修正されません。 LDAP サーバーがインストールされると、自動的に DB2® データベースがインストールされます。 DB2 インストールにより、グループ・アカウント *dbsysadm* が作成されます。 *dbsysadm* のデフォルト・グループ ID は 400 です。 LDAP サーバーの構成中、**mksecldap** コマンドは *ldapdb2* ユーザー・アカウントを作成します。
- **OpenSSH:** OpenSSH のインストール中に、ユーザー *sshd* とグループ *sshd* がシステムに追加されます。 対応するユーザーおよびグループ ID は変更しないでください。 SSH の特権分離フィーチャーに、ID が必要です。

ドメインなしのグループ

ドメインなしのグループ機能を使用すると、あるドメインで定義されているユーザーを、別のドメインで定義されているグループに割り当てることができます。この機能は、LDAP (Lightweight Database Access Protocol) およびローカルのドメインのみをサポートします。

LDAP Authentication Load Module (LDAP モジュール) を使用して、LDAP サーバー上にユーザーとグループを作成することができます。また、Local Authentication Load Module (ローカル・モジュール) を使用して、ローカル・システム上にユーザーとグループを作成することもできます。**domainlessgroups** 機能が有効に設定されていない場合、LDAP またはローカル・システムに作成されるユーザーおよびユーザー・グループを、作成されたロード・ドメインの外部のグループに割り当てることはできません。例えば、LDAP ドメインに作成されるユーザーを、ローカル・ドメインに関連付けられたグループに割り当てることはできません。

この制限を回避して LDAP グループとローカル・グループの両方にユーザーを割り当てるには、**domainlessgroups** システム・プロパティーを有効に設定します。**domainlessgroups** プロパティー

は、/etc/secvars.cfg ファイルで定義されます。これは LDAP モジュールおよびローカル・モジュールに対してのみサポートされます。このプロパティーに指定可能な値は以下のとおりです。

false (デフォルト値**)**
グループ属性は、LDAP モジュールおよびローカル・モジュールからマージされません。

true グループ属性は、LDAP モジュールおよびローカル・モジュールからマージされます。例えば、LDAP ユーザーをローカル・グループに割り当てることができます。

domainlessgroups プロパティーの値を表示するには、次のコマンドを実行します。

```
lssec -f /etc/secvars.cfg -s groups -a domainlessgroups
```

domainlessgroups プロパティーを true に設定するには、次のコマンドを実行します。

```
chsec -f /etc/secvars.cfg -s groups -a domainlessgroups=true
```

次の表は、**domainlessgroups** プロパティーの設定に応じて、ユーザーおよびグループのコマンドの結果がどのように異なるかを示しています。

表 *5.* **domainlessgroups** プロパティーによって影響を受ける、選択されたコマンドの結果

コマンド	**domainlessgroups** プロパティーを **true** に設定した場合の結果
`chgroup -R ldap|files`	指定されたドメインでグループを更新します。ユーザーを LDAP グループまたはローカル・グループに追加できます。
`chuser -R ldap|files`	指定されたドメインでユーザーの設定を変更します。他のドメインに定義されたグループが指定された場合、そのグループもユーザー情報を使用して更新されます。
`login` *username* or `su`	ユーザー・レジストリーからユーザー属性がリトリーブされます (グループ ID 属性を除く)。グループ ID のユーザー属性は、LDAP ドメインおよびローカル・ドメインの両方からマージされます。
`lsgroup -R ldap|files`	指定されたドメインのグループ属性をすべてリストします。指定されたグループが指定されたドメインに見つからない場合、このコマンドは失敗します。
`lsuser -R ldap|files`	ユーザーが定義されるドメインおよび別のドメインのすべてのグループから情報がマージされた後、ユーザーの属性をリストします。ユーザーの 1 次グループが、ユーザーが定義されるドメインに定義されていない場合、そのグループは別のドメインから解決されます。
`mkgroup -R ldap|files`	指定されたドメインでグループを作成します。グループを作成した後、ユーザー (LDAP またはローカル) をそのドメインのグループ・データベースのグループに割り当ててください。ユーザーは LDAP グループまたはローカル・グループに追加できます。
`mkuser -R ldap|files`	指定されたドメインにユーザーを作成します。他のドメインに定義されたグループが指定された場合、そのグループもユーザー情報を使用して更新されます。
`rmgroup -R ldap|files`	指定されたグループを指定されたドメインから削除します。そのグループが任意のドメインに定義されている任意のユーザーに対して 1 次グループとして割り当てられている場合、コマンドは失敗します。
`rmuser -R ldap|files`	指定されたユーザーを指定されたドメインから削除します。また、他のドメインに定義された任意のグループからユーザーを削除し、そのユーザーをメンバーとします。

関連概念:

170 ページの『LDAP 認証ロード・モジュール』
セキュリティー・サブシステムの LDAP 活用は、LDAP 認証ロード・モジュールとして実装されます。これは、NIS、DCE、KRB5 などのその他のロード・モジュールと概念的に似ています。 ロード・モジュールは **/usr/lib/security/methods.cfg** ファイルで定義されます。

関連情報:

chgroup コマンド

chuser コマンド

login コマンド

lsgroup コマンド

lsuser コマンド

mkgroup コマンド

mkuser コマンド

rmgroup コマンド

rmuser コマンド

su コマンド

パスワード

パスワードが推測されることは、システムに対するアタックの最も一般的なものの 1 つです。 したがって、パスワード制限ポリシーの制御とモニターは不可欠です。

AIX は、次のような値を確立するなど、パスワード・ポリシーをより強固にするための仕組みを提供します。

- パスワードの変更前後に経過する可能性のある週の最小数および最大数
- パスワードの最小の長さ
- パスワード選択時に使用できる英字の最小数

良いパスワードの設定:

良いパスワードは、システムへの無許可侵入に対する防御として有効な最前線になります。

パスワードは次のような場合に有効です。

- 英大文字および小文字の両方が混ざっている
- 英字、数字、または句読文字の組み合わせ。また、`~!@#$%^&*()-_=+[]{}|¥;:'",.<>?/<space>` などの特殊文字も使用できます。
- どこにも書き留めない
- `/etc/security/passwd` ファイルを使用する場合、長さは最低 7 文字から最大 PW_PASSLEN 文字 (LDAP などのレジストリーを使用する認証インプリメンテーションでは、この最大長を超えるパスワードを使用できます)
- 辞書にあるような実際の単語でない
- *qwerty* のようなキーボード上の文字のパターンでない
- 実際の単語や、逆につづった既知のパターンでない
- 自分自身、家族、または友人などの個人情報が含まれていない
- 以前のパスワードと同じパターンを使用しない
- 近くにいる人がパスワードを判別できないよう、比較的早くタイプできる

これらのメカニズムに加え、推測される可能性のある標準 UNIX ワードを含めないようにパスワードを制限することによって、さらに厳密なルールを強制することができます。 このフィーチャーでは、dictionlist を使用します。これには、まず **bos.data** および **bos.txt** ファイルセットをインストールしておくことが必要です。

以前に定義された dictionlist を実装するには、**/etc/security/users** ファイルの次の行を編集します。

```
dictionlist = /usr/share/dict/words
```

/usr/share/dict/words ファイルは dictionlist を使用して、標準 UNIX ワードがパスワードとして使用されないようにします。

/etc/passwd ファイルの使用**:**

/etc/passwd ファイルは、従来から、システムにアクセス権のあるすべての登録済みユーザーを追跡するのに使用されます。

/etc/passwd は、コロンで区切られたファイルで、次の情報を含みます。

- ユーザー名
- 暗号化パスワード
- ユーザー ID 番号 (UID)
- ユーザーのグループ ID 番号 (GID)
- ユーザーのフルネーム (GECOS)
- ユーザーのホーム・ディレクトリー
- ログイン・シェル

次に示すものは **/etc/passwd** ファイルの一例です。

```
root:!:0:0::/:/usr/bin/ksh
daemon:!:1:1::/etc:
bin:!:2:2::/bin:
sys:!:3:3::/usr/sys:
adm:!:4:4::/var/adm:
uucp:!:5:5::/usr/lib/uucp:
guest:!:100:100::/home/guest:
nobody:!:4294967294:4294967294::/:
lpd:!:9:4294967294::/:
lp:*:11:11::/var/spool/lp:/bin/false
invscout:*:200:1::/var/adm/invscout:/usr/bin/ksh
nuucp:*:6:5:uucp login user:/var/spool/uucppublic:/usr/sbin/uucp/uucico
paul:!:201:1::/home/paul:/usr/bin/ksh
jdoe:*:202:1:John Doe:/home/jdoe:/usr/bin/ksh
```

AIX は、暗号化されたパスワードを UNIX システムと同様に /etc/password ファイルに保管するのではなく、デフォルトでは /etc/security/password [1] ファイルに保管します。このファイルは、root ユーザーのみが読むことができます。 **/etc/passwd** にファイルされたパスワードは、AIX では、パスワードが存在するかどうか、あるいはアカウントがブロックされているかどうかを示すために使用されます。

/etc/passwd ファイルは root ユーザーが所有し、すべてのユーザーが読むことができます。ただし、root ユーザーのみが書き込みアクセス権 (-rw-r--r-- として示される) を持ちます。ユーザー ID にパスワードがある場合、パスワード・フィールドには ! (感嘆符) が入ります。 ユーザー ID にパスワードがない場合、パスワード・フィールドには * (アスタリスク) が入ります。 暗号化されたパスワードは

1. /etc/security/password

/etc/security/passwd ファイルに保管されます。次の例では、上記の **/etc/passwd** ファイルのエントリーに基づいた **/etc/security/passwd** ファイルの最後の 4 つのエントリーが含まれています。

```
guest:
        password = *

nobody:
        password = *

lpd:
        password = *

paul:
        password = eacVScDKri4s6
        lastupdate = 1026394230
        flags = ADMCHG
```

ユーザー ID jdoe には、**/etc/security/passwd** ファイルにエントリーがありません。これは、**/etc/passwd** ファイルでパスワードを設定していないためです。

/etc/passwd ファイルの整合性は、**pwdck** コマンドを使用して検査することができます。 **pwdck** コマンドは、すべてのユーザーまたは指定されたユーザーの定義を検査することによって、ユーザー・データベース・ファイル中のパスワード情報の正確さを検証します。

/etc/passwd ファイルの使用とネットワーク環境**:**

従来のネットワーク環境では、ユーザーはシステムへのアクセス権を得るために、各システム上にアカウントを持っていなければなりませんでした。

これは、通常、ユーザーが各システム上の **/etc/passwd** ファイルそれぞれにエントリーを持っているという意味でした。 しかし、分散環境では、すべてのシステムが確実に同じ **/etc/passwd** ファイルを持つための、簡単な方法がありません。 この問題を解決するため、Network Information System (NIS) を含むいくつかの方法で、/etc/passwd ファイルの情報をネットワーク全体で使用することが可能となりました。

ユーザー名およびパスワードを隠す**:**

セキュリティーのレベルを高めるため、ユーザー ID およびパスワードが、システム内で見えないようにしてください。

.netrc ファイルにユーザー ID およびパスワードが入っています。このファイルは暗号化またはエンコードによって保護されていないので、その内容はプレーン・テキストとしてはっきり表示されます。 これらのファイルを見つけるには、次のコマンドを実行します。

```
# find `awk -F: '{print $6}' /etc/passwd` -name .netrc -ls
```

これらのファイルを見つけたら、削除してください。 パスワードを保存するより有効な方法は、Kerberos を設定することです。 Kerberos の詳細については、 320 ページの『Kerberos』 を参照してください。

推奨パスワード・オプションの設定**:**

適切なパスワード管理は、ユーザーの教育という方法でのみ実現できます。 ある種のセキュリティーを強化するために、オペレーティング・システムには構成可能なパスワード制約事項が設けられています。 これらの制約事項を使って管理者は、ユーザーによるパスワードの選択に制約を加え、また、パスワードの定期的な変更を強制することができます。

パスワード・オプションおよび拡張ユーザー属性は、/etc/security/user ファイル、すなわちユーザーの属性スタンザを含む ASCII ファイルにあります。 これらの制約事項は、新規パスワードがユーザーに定

義される時点で必ず適用されます。 すべてのパスワード制約事項は、ユーザーごとに定義されます。 **/etc/security/user** ファイルのデフォルト・スタンザ内にこれらの制約事項を保持することで、すべてのユーザーに対して同じ制約事項が適用されます。 パスワード・セキュリティーを維持管理するには、すべてのパスワードを同じように保護する必要があります。

管理者はパスワード制限を拡張することもできます。 **/etc/security/user** ファイルの **pwdchecks** 属性を使用すれば、管理者は、新規のサブルーチン (メソッド とも呼ばれる) をパスワード制限コードに追加することができます。 このように、オペレーティング・システムで、ローカル・サイトのポリシーを追加して実施することが可能になります。 詳しくは、 79 ページの『パスワード制限の拡張』を参照してください。

パスワードの制約事項は、注意して適用してください。 過度に制限しようとすると、例えば、パスワード・スペースを制限してパスワードを推測しやすくしたり、覚えにくいパスワードの選択をユーザーに強制したためにどこかに書き留めないと覚えられなかったりすると、パスワード・セキュリティーが危険にさらされる可能性があります。 基本的には、パスワード・セキュリティーはユーザー次第であると言えます。 簡易なパスワード制限は、適切な指針および、現在のパスワードが固有であるかの随時監査と結合することにより、最良のポリシーとなります。

次の表は、**/etc/security/user** ファイル中のユーザー・パスワードに関連する、いくつかのセキュリティー属性に推奨される値をリストしています。

表 *6.* ユーザー・パスワードに推奨されるセキュリティー属性値

属性	説明	推奨値	デフォルト値	最大値
dictionlist	パスワードに標準 UNIX ワードが含まれていないことを確認します。	**/usr/share/dict/words**	適用不可	適用不可
histexpire	パスワードが再利用できるまでの週数	26	0	260*
histsize	許可されるパスワードの反復回数	20	0	50
maxage	パスワードが変更される必要が生じるまでの最大週数	8	0	52
maxexpired	有効期限が切れたパスワードをユーザーが変更できる *maxage* を超えた最大週数 (Root を除く)	2	-1	52
maxrepeats	パスワードで反復可能な文字の最大数	2	8	8
minage	パスワードが変更可能になるまでの最小週数。 これは、偶発的に暗号漏えいして最近変更されたパスワードを、管理者が常に容易にリセットできない限り、非ゼロ値に設定するべきではありません。	0	0	52
minalpha	パスワードに必要な英字の最小数	2	0	PW_PASSLEN**

表 6. ユーザー・パスワードに推奨されるセキュリティー属性値 (続き)

属性	説明	推奨値	デフォルト値	最大値
mindiff	パスワードに入れる必要のある固有文字の最小数	4	0	PW_PASSLEN**
minlen	パスワードの最小の長さ	6 (root ユーザーの場合 8)	0	PW_PASSLEN**
minother	パスワードで必要な非英字の最小数	2	0	PW_PASSLEN**
pwdwarntime	システムが、パスワード変更が必要であるという警告を出すまでの日数	5	適用不可	適用不可
pwdchecks	このエントリーは、パスワードの品質を検査するカスタム・コードを使って **passwd** コマンドを増強するのに使用できます。	詳しくは、 79 ページの『パスワード制限の拡張』を参照してください。	適用不可	適用不可

* 最大 50 個のパスワードが保存されます

** PW_PASSLEN は `userpw.h` ファイルに定義されています。

テキスト処理をシステムにインストールしている場合、管理者は、**/usr/share/dict/words** ファイルを **dictionlist** ディクショナリー・ファイルとして使用することができます。 このような場合、管理者は、**minother** 属性を `0` に設定することができます。ディクショナリー・ファイルの大部分のワードには、**minother** 属性カテゴリーに入る文字が使用されないため、**minother** 属性を 1 またはそれ以上の値に設定することで、このディクショナリー・ファイル内の大部分のワードを必要とすることがなくなります。

システム上のパスワードの最小の長さは、**minlen** 属性の値、または **minalpha** 属性の値に **minother** 属性の値を加えた値のいずれか大きい方によって設定されます。

パスワードの最大長は、**PW_PASSLEN** 属性で指定されている文字数です。保管されたパスワード値を生成するときに使用された文字数は、システムで使用されたパスワード・アルゴリズムによって異なります。パスワードのアルゴリズムは `/etc/security/pwdalg.cfg` ファイルに定義され、使用するデフォルトのパスワード・アルゴリズムは `/etc/security/login.cfg` ファイルの **pwd_algorithm** 属性から構成できます。**minalpha** 属性の値と **minother** 属性の値を加えた値が、**PW_PASSLEN** 属性より大きくてはなりません。 **minalpha** 属性の値を **minother** 属性の値に加えた値が **PW_PASSLEN** 属性より大きい場合には、**minother** 属性の値は **PW_PASSLEN** 属性から **minalpha** 属性の値を差し引いた値まで減少します。

histexpire 属性と **histsize** 属性の両方の値を設定すると、システムは、両方の条件を満たすのに必要な数のパスワードを、ユーザーあたり最大 50 パスワードというシステム限度までの範囲で保存します。 null パスワードは保存されません。

/etc/security/user ファイルを編集して、ユーザー・パスワードの管理に使用するデフォルトを組み込むことができます。 別の方法として、**chuser** コマンドを使用して、属性値を変更することができます。

このファイルで使用できる他のコマンドは、**mkuser**、**lsuser**、および **rmuser** コマンドです。 **mkuser** コマンドは、**/etc/security/user** ファイル中の各新規ユーザーごとにエントリーを作成し、その属性を **/usr/lib/security/mkuser.default** ファイルで定義される属性で初期化します。属性およびその値を表示するには、**lsuser** コマンドを使用します。 ユーザーの除去には、**rmuser** コマンドを使用します。

8 文字を超えるパスワードおよび Loadable Password Algorithm:

近年のコンピューター・ハードウェアの進歩により、旧来の UNIX パスワードの暗号化は、強引なパスワード推測攻撃にはぜい弱になっています。 暗号化としてはぜい弱なアルゴリズムが原因で、強力なパスワードであってもリカバリーが必要になる場合があります。 AIX は、セキュア・パスワードのハッシュ・メカニズムを提供する Loadable Password Algorithm (LPA) をサポートします。

従来のパスワード ***crypt*** 機能*:*

標準の AIX 認証メカニズムでは、**crypt** という片方向のハッシュ機能を使用してユーザーを認証します。 **crypt** 機能は、修正 DES アルゴリズムです。 これは指定されたパスワードと Salt で固定データ・アレイの片方向の暗号化を実行します。

crypt 機能はパスワード文字列の最初の 8 文字のみを使用し、ユーザーのパスワードは 8 文字に切り捨てられます。 パスワードの文字が 8 文字より少ない場合は、右側にゼロ・ビットが埋め込まれます。 各文字から 7 ビットを使用して、56 ビット DES 鍵が得られます。

Salt は文字セット「**A-Z**」、「**a-z**」、「**0-9**」、「**.** (ピリオド)」、「**/**」から選択された 2 文字の文字列です (Salt の 12 ビットは DES アルゴリズムを摂動するために使用されます)。 Salt はハッシュ・アルゴリズムを変更するために使用され、同一の平文パスワードで 4,096 の可能なパスワード暗号化を生成できるようになります。 これは DES アルゴリズムに対する修正 (ビット i が Salt で設定されている場合に、DES E-Box 出力でビット i と i+24 をスワッピングする) により実現され、DES 暗号化ハードウェアをパスワードの推測に使用できないようにします。

64 ビットの全ビットがゼロのブロックが DES 鍵で 25 回暗号化されます。 最終出力は、暗号化された 64 ビット値と連結した 12 ビットの Salt です。 得られた 76 ビット値は、13 個の印刷可能 ASCII 文字に base64 の形式で再コード化されます。

パスワード・ハッシュ・アルゴリズム*:*

MD5 などのハッシュ・アルゴリズムは、**crypt** 機能よりも中断が困難です。これにより、強引なパスワード解読攻撃に対して強力なメカニズムが提供されます。パスワード全体はハッシュの生成に使用されるので、パスワード・ハッシュ・アルゴリズムがパスワードの暗号化に使用されるときは、パスワードの長さに制限はありません。

ロード可能パスワードのアルゴリズム*:*

AIX 6.1 以降には、新規パスワードの暗号化アルゴリズムを容易にデプロイできる Loadable Password Algorithm (LPA) メカニズムが実装されています。

それぞれのパスワードの暗号化アルゴリズムは LPA ロード・モジュールで実装され、このアルゴリズムが必要になるランタイム時にロードされます。 サポートされる LPA およびその属性は `/etc/security/pwdalg.cfg` システム構成ファイルに定義されます。

管理者はパスワードを暗号化するときに特定の LPA を使用する、システム全体のパスワード暗号化メカニズムをセットアップすることができます。 システム全体のパスワード・メカニズムが変更されても、以前に選択されたパスワード暗号化メカニズム (**crypt** 機能など) によって暗号化されるパスワードも、引き続いてサポートされます。

9 文字以上のパスワードのサポート*:*

AIX 6.1 以降で実装された LPA のすべては、 8 文字を超えるパスワードをサポートします。 パスワードの長さの制限は、さまざまな LPA により異なります。 サポートされるパスワードの最大長は 255 文字です。

LPA 構成ファイル*:*

LPA 構成ファイルは /etc/security/pwdalg.cfg です。これはサポートされる LPA の属性を定義するスタンザ・ファイルです。

以下の LPA 属性が構成ファイルに定義されます。

- LPA モジュールへのパス
- ランタイム時に LPA モジュールに渡されるオプショナル・フラグ

構成ファイルに定義された LPA 属性は **getconfattr** および **setconfattr** インターフェースを使用してアクセスできます。

以下のスタンザは /etc/security/pwdalg.cfg に **ssha256** という名前の LPA を定義する例です。

```
ssha256:
    lpa_module = /usr/lib/security/ssha
    lpa_options = algorithm=sha256
```

システム・パスワード・アルゴリズム*:*

システム管理者は、LPA をパスワード・ハッシュ・アルゴリズムとして選択することにより、システム全体のパスワード・アルゴリズムを設定できます。 活動状態にすることができるシステム・パスワード・アルゴリズムは一時点で 1 つのみです。 システム・パスワード・アルゴリズムは /etc/security/login.cfg ファイルの **usw** スタンザの **pwd_algorithm** システム属性によって定義されます。

/etc/security/login.cfg ファイルの **pwd_algorithm** 属性に対する有効値は、 /etc/security/pwdalg.cfg ファイルに定義される LPA スタンザ名です。 **pwd_algorithm** 属性に対する別の有効値は、従来の **crypt** 暗号化と呼ばれる **crypt** です。 構成ファイルから **pwd_algorithm** 属性を省略すると、デフォルト値として **crypt** が使用されます。

/etc/security/login.cfg 例の以下の例では、システム全体のパスワードの暗号化アルゴリズムとして **ssha256** LPA を使用しています。

```
 ... ...
usw:
   shells = /bin/sh,/bin/bsh,/bin/csh,/bin/ksh,/bin/tsh,/bin/ksh93
   maxlogins = 32767
   logintimeout = 60
   maxroles = 8
   auth_type = STD_AUTH
   pwd_algorithm = ssha256
 ... ...
```

システム・パスワード・アルゴリズムは、新規に作成されたパスワードと変更されたパスワードに対してのみ有効になります。 移行後は、それ以降のすべての新規パスワードまたはパスワードの変更には、システム・パスワード・アルゴリズムが使用されます。システム・パスワード・アルゴリズムが選択される前に存在していたパスワードは、それが標準 **crypt** 機能またはその他のサポートされる LPA モジュールのいずれかで生成されたものであっても、なお、システムで機能します。 したがって、異なる LPA によって生成された混合パスワードはシステムで共存することが可能です。

システム・パスワード・アルゴリズムのセットアップ*:*

システム管理者は **chsec** コマンドを使用して、システム・パスワード・アルゴリズムをセットアップすること、または **vi** などのエディターを使用して、/etc/security/login.cfg ファイルの **pwd_algorithm** 属性を変更することができます。

システム・パスワード・アルゴリズムを設定する場合、 **chsec** コマンドは指定 LPA の定義を自動的に検査するため、 **chsec** コマンドの使用をお勧めします。

chsec コマンドの使用

smd5 LPA をシステム全体にわたるパスワードの暗号化モジュールとして設定するには、以下のコマンドを実行します。

```
chsec -f /etc/security/login.cfg -s usw -a pwd_algorithm=smd5
```

chsec コマンドを使用して、**pwd_algorithm** 属性を変更するときに、 **chsec** コマンドは /etc/security/pwdalg.cfg ファイルをチェックして、指定された LPA を検査します。 このチェックに失敗すると、**chsec** コマンドは失敗します。

エディターの使用

エディターを使用して手動で /etc/security/login.cfg ファイルの **pwd_algorithm** 属性値を変更する場合は、指定値が /etc/security/pwdalg.cfg ファイルに定義されているスタンザ名であることを確認します。

パスワード制限の拡張*:*

パスワード・プログラムでパスワードを受け入れたり拒否するために使用するルール (パスワード構成の制約事項) は、システム管理者がサイト固有の制約事項を提供するために拡張することができます。

制約事項は、メソッドを追加して拡張します。これらのサブルーチンは、パスワードの変更時に呼び出されます。 **/etc/security/user** ファイルにある **pwdchecks** 属性は、呼び出されるメソッドを指定します。

AIX Version 6.1 Technical Reference からは、**pwdrestrict_method** の説明が含まれています。これは、指定されたパスワード制限メソッドが準拠しなければならないサブルーチン・インターフェースです。パスワード構成制限を正しく拡張するには、システム管理者は、パスワード制限メソッドの作成時にこのインターフェースをプログラミングする必要があります。 パスワード構成制限を拡張するときは、注意が必要です。これらの拡張は、**login** コマンド、**passwd** コマンド、**su** コマンド、および他のプログラムに直接作用します。 システムのセキュリティーは、不正あるいは不完全なコードによって容易に損傷を被る可能性があります。

ユーザー認証

ユーザーの ID を確立するために識別と認証が使用されます。

各ユーザーは、システムにログインする必要があります。 ユーザーは、アカウントのユーザー名と、さらにそのアカウントにパスワードが設定されていれば、そのパスワードも入力します (セキュア・システムの場合は、すべてのアカウントにパスワードが必要です。パスワードがない場合、そのアカウントは無効になります)。 パスワードが正しければ、ユーザーは、そのアカウントにログインされます。つまり、ユーザーは、そのアカウントのアクセス権とその他の特権を入手します。 /etc/passwd ファイルと /etc/security/passwd ファイルに、ユーザー・パスワードが保守されています。

デフォルトでは、ユーザーはファイル・レジストリーで定義されます。 これは、ユーザー・アカウントとグループの情報はフラットな ASCII ファイルに保管されるということです。 プラグイン・ロード・モジュールの導入により、ユーザーを他のレジストリーでも定義できるようになりました。 例えば、ユーザー管理用に LDAP プラグイン・モジュールを使用する場合、ユーザー定義は LDAP リポジトリーに保管されます。 この場合、**/etc/security/user** ファイルにはユーザーのエントリーはありません (ただし、ユーザー属性 **SYSTEM** および **registry** の場合には例外があります)。複合ロード・モジュール (すなわち、認証部分とデータベース部分から成るロード・モジュール) をユーザー管理に使用する場合、半分のデータベース部分は AIX ユーザー・アカウント情報の管理方法を決定し、もう半分の認証部分は認証とパスワード関連の管理について記述します。認証部分は、特定のロード・モジュール・インターフェース (newuser、getentry、putentry など) を実装することにより、認証固有のユーザー・アカウント管理属性についても記述する場合があります。

認証方式は、/etc/security/user ファイルで定義されている **SYSTEM** 属性と registry 属性によって制御されます。 システム管理者は、/etc/security/login.cfg ファイルに authcontroldomain 属性を定義することで、**SYSTEM** 属性と registry 属性が強制的に authcontroldomain から取得されるように指定できます。例えば、authcontroldomain=LDAP と指定すると、システムは強制的にユーザーの **SYSTEM** および registry を LDAP から探して、ユーザーに使用された認証方式を判別します。 ローカルで定義されたユーザーの場合は例外があり、その場合は、authcontroldomain 設定が無視され、**SYSTEM** と registry は常に /etc/security/user ファイルから取得されます。

authcontroldomain 属性の許容トークンは、/usr/lib/security/methods.cfg ファイルのファイルまたはスタンザ名です。

SYSTEM 属性の値は、文法によって定義されます。この文法を使用することにより、システム管理者は 1 つ以上のメソッドを結合して、システムに対して特定のユーザーを認証します。 既知のメソッド・トークンは、compat、DCE、files、および NONE です。

システム・デフォルトは compat です。 デフォルトの SYSTEM=compat は、認証用にローカル・データベースを使用することをシステムに通知し、レゾリューションがない場合は、ネットワーク情報サービス (NIS) データベースの使用が試みられます。 files トークンは、認証時にローカル・ファイルのみを使用することを指定します。一方、SYSTEM=DCE を指定すると、DCE 認証フローが使用されます。

NONE トークンは、メソッドの認証をオフにします。すべての認証をオフにするには、NONE トークンがユーザーのスタンザの SYSTEM 行と auth1 行に置かれていなければなりません。

複数のメソッドを指定して、それらを論理コンストラクター AND および OR を使用して結合できます。 例えば、SYSTEM=DCE OR compat は、DCE またはローカル認証 (crypt()) を示された順序で行い、いずれかが正常に実行されれば、ユーザーがログインできることを示します。

同様の方法で、システム管理者は **SYSTEM** 属性に認証ロード・モジュール名を使用できます。 例えば、**SYSTEM** 属性が SYSTEM=KRB5files または compat に設定されている場合、AIX ホストは最初に認証用の Kerberos フローを試み、それに失敗すると、標準の AIX 認証を試みます。

SYSTEM および **registry** 属性は常に、**/etc/security/user** ファイルのローカル・ファイルシステムに保管されます。 AIX ユーザーが LDAP で定義され、**SYSTEM** と **registry** 属性がそれぞれに設定されている場合、そのユーザーは **/etc/security/user** ファイルにエントリーを持ちます。

ユーザーの **SYSTEM** および **registry** 属性は、**chuser** コマンドを使用して変更できます。

SYSTEM 属性の許容トークンは、**/usr/lib/security/methods.cfg** ファイル内に定義しておくことができます。

注: root ユーザーは、常に、ローカル・システム・セキュリティー・ファイルによって認証されます。 root ユーザーの **SYSTEM** 属性項目は、**/etc/security/user** ファイル内で特別に SYSTEM=compat に設定されます。

この代替の認証方法は、/etc/security/user にある **SYSTEM** 属性によってシステムに組み込まれています。例えば、分散コンピューティング環境 (DCE) ではパスワード認証が必要ですが、この環境では、それらのパスワードを **/etc/passwd** コマンドおよび **/etc/security/passwd** コマンドで使用される暗号化モデルとは異なる方法で検証します。 DCE を使って認証を行うユーザーは、**/etc/security/user** 内のスタンザを SYSTEM=DCE に設定することができます。

他の **SYSTEM** 属性値には、**compat**、**files**、および **NONE** があります。 compat トークンは、ネーム・レゾリューション (および後続の認証) がローカル・データベースに従っている場合に使用されますが、レゾリューションがない場合は、ネットワーク情報サービス (NIS) データベースが試行されます。 files トークンを指定すると、認証時にローカル・ファイルだけが使用されます。 最後に、NONE トークンはメソッドの認証をオフにします。 すべての認証をオフにするには、NONE トークンがユーザーのスタンザの **SYSTEM** 行と **auth1** 行に置かれていなければなりません。

このほかに、**SYSTEM** 属性の許容トークンを **/usr/lib/security/methods.cfg** 内に定義しておくことができます。

注: root ユーザーは、常に、ローカル・システム・セキュリティー・ファイルによって認証されます。 root ユーザーの **SYSTEM** 属性項目は、**/etc/security/user** 内で特別に SYSTEM = "compat" に設定されます。

パスワードの保護について詳しくは、「オペレーティング・システムおよびデバイスの管理」を参照してください。

ログイン・ユーザー ID

ユーザーに対して記録されるすべての監査イベントは、この ID のラベルが付けられ、監査レコードの生成時に検査することができます。 ログイン・ユーザー ID について詳しくは、「オペレーティング・システムおよびデバイスの管理」を参照してください。

認証ロード・モジュールによってサポートされるユーザーおよびグループ属性

一連のユーザー関連属性およびグループ関連属性は、AIX で識別と認証を達成するために使用されます。

以下の表では、これらのユーザー属性とグループ属性のほとんどを 1 つのリストとして表示するほかに、さまざまなロード・モジュールからのこれらの属性に対するサポートを示します。 表の各行は属性に対応しており、各列はロード・モジュールを表しています。 ロード・モジュールがサポートする属性は、ロード・モジュール列の「はい」で示されています。

注: PKI と Kerberos は認証専用のモジュールなので、データベース・モデル (LOCAL または LDAP など) と組み合わせる必要があります。 それらは、LOCAL または LDAP が提供する属性以外の、特定の追加 (拡張) 属性をサポートしています。 LOCAL や LDAP を使用して他の属性で機能的に達成できる場合もありますが、マーキングはこれらのモジュールの拡張属性に対してのみ示されています。

表 7. ユーザー属性と認証ロード・モジュールのサポート

ユーザー属性	Local	NIS	LDAP	PKI	Kerberos
account_locked	はい	いいえ	はい	いいえ	いいえ
admgroups	はい	いいえ	はい	いいえ	いいえ
admin	はい	いいえ	はい	いいえ	いいえ
auditclasses	はい	いいえ	はい	いいえ	いいえ
auth_cert	いいえ	いいえ	いいえ	はい	いいえ
auth_domain	はい	いいえ	はい	いいえ	いいえ
auth_name	はい	いいえ	はい	いいえ	いいえ
auth1 **注: auth1** 属性は推奨されない属性であり、使用すべきではありません。	はい	いいえ	はい	いいえ	いいえ
auth2 **注: auth2** 属性は推奨されない属性であり、使用すべきではありません。	はい	いいえ	はい	いいえ	いいえ
capabilities	はい	いいえ	はい	いいえ	いいえ
core	はい	いいえ	はい	いいえ	いいえ
core_compress	はい	いいえ	いいえ	いいえ	いいえ
core_hard	はい	いいえ	はい	いいえ	いいえ
core_naming	はい	いいえ	いいえ	いいえ	いいえ
core_path	はい	いいえ	いいえ	いいえ	いいえ
core_pathname	はい	いいえ	いいえ	いいえ	いいえ
cpu	はい	いいえ	はい	いいえ	いいえ
daemon	はい	いいえ	はい	いいえ	いいえ
data	はい	いいえ	はい	いいえ	いいえ
data_hard	はい	いいえ	はい	いいえ	いいえ
dce_export	はい	いいえ	はい	いいえ	いいえ
dictionlist	はい	いいえ	はい	いいえ	いいえ
expires	はい	いいえ	はい	いいえ	はい
flags	はい	いいえ	はい	いいえ	はい
fsize	はい	いいえ	はい	いいえ	いいえ
fsize_hard	はい	いいえ	はい	いいえ	いいえ
funcmode	はい	いいえ	はい	いいえ	いいえ
gecos	はい	はい	はい	いいえ	いいえ
groups	はい	はい	はい	いいえ	いいえ
groupsids	はい	はい	はい	いいえ	いいえ
histexpire	はい	いいえ	はい	いいえ	いいえ
home	はい	はい	はい	いいえ	いいえ
host_last_login	はい	いいえ	はい	いいえ	いいえ
host_last_unsuccessful_login	はい	はい	はい	いいえ	いいえ
hostsallowedlogin	はい	いいえ	はい	いいえ	いいえ
hostsdeniedlogin	はい	いいえ	はい	いいえ	いいえ
id	はい	はい	はい	いいえ	いいえ
krb5_attributes	いいえ	いいえ	いいえ	いいえ	はい
krb5_kvno	いいえ	いいえ	いいえ	いいえ	はい
krb5_last_pwd_change	いいえ	いいえ	いいえ	いいえ	はい
krb5_max_renewable_life	いいえ	いいえ	いいえ	いいえ	はい
krb5_mknvo	いいえ	いいえ	いいえ	いいえ	はい

表 7. ユーザー属性と認証ロード・モジュールのサポート (続き)

ユーザー属性	Local	NIS	LDAP	PKI	Kerberos
krb5_mod_date	いいえ	いいえ	いいえ	いいえ	はい
krb5_mod_name	いいえ	いいえ	いいえ	いいえ	はい
krb5_names	いいえ	いいえ	いいえ	いいえ	はい
krb5_principal	いいえ	いいえ	いいえ	いいえ	はい
krb5_principal_name	いいえ	いいえ	いいえ	いいえ	はい
krb5_realm	いいえ	いいえ	いいえ	いいえ	はい
lastupdate	はい	はい	はい	いいえ	いいえ
login	はい	いいえ	はい	いいえ	いいえ
loginretries	はい	いいえ	はい	いいえ	いいえ
logintimes	はい	いいえ	はい	いいえ	いいえ
maxage	はい	はい	はい	いいえ	はい
maxexpired	はい	はい	はい	いいえ	いいえ
maxrepeats	はい	いいえ	はい	いいえ	いいえ
maxulogs	はい	いいえ	はい	いいえ	いいえ
minage	はい	はい	はい	いいえ	いいえ
minalpha	はい	いいえ	はい	いいえ	いいえ
mindiff	はい	いいえ	はい	いいえ	いいえ
mindigit	はい	いいえ	はい	いいえ	いいえ
minlen	はい	いいえ	はい	いいえ	いいえ
minloweralpha	はい	いいえ	はい	いいえ	いいえ
minother	はい	いいえ	はい	いいえ	いいえ
minspecialchar	はい	いいえ	はい	いいえ	いいえ
minupperalpha	はい	いいえ	はい	いいえ	いいえ
nofiles	はい	いいえ	はい	いいえ	いいえ
nofiles_hard	はい	いいえ	はい	いいえ	いいえ
password	はい	はい	はい	いいえ	いいえ
pgid	はい	はい	いいえ	いいえ	いいえ
pgrp	はい	はい	はい	いいえ	いいえ
projects	はい	いいえ	はい	いいえ	いいえ
pwdchecks	はい	いいえ	はい	いいえ	いいえ
pwdwarntime	はい	いいえ	はい	いいえ	いいえ
rcmds	はい	いいえ	はい	いいえ	いいえ
registry	はい	いいえ	いいえ	いいえ	いいえ
rlogin	はい	いいえ	はい	いいえ	いいえ
roles	はい	いいえ	はい	いいえ	いいえ
rss	はい	いいえ	はい	いいえ	いいえ
rss_hard	はい	いいえ	はい	いいえ	いいえ
screens	はい	いいえ	はい	いいえ	いいえ
shell	はい	はい	はい	いいえ	いいえ
spassword	はい	はい	はい	いいえ	いいえ
stack	はい	いいえ	はい	いいえ	いいえ
stack_hard	はい	いいえ	はい	いいえ	いいえ
su	はい	いいえ	はい	いいえ	いいえ
sugroups	はい	いいえ	はい	いいえ	いいえ
sysenv	はい	いいえ	はい	いいえ	いいえ

表 7. ユーザー属性と認証ロード・モジュールのサポート (続き)

ユーザー属性	Local	NIS	LDAP	PKI	Kerberos
SYSTEM	はい	いいえ	いいえ	いいえ	いいえ
time_last_login	はい	いいえ	はい	いいえ	いいえ
time_last_unsuccessful_login	はい	いいえ	はい	いいえ	いいえ
tpath	はい	いいえ	はい	いいえ	いいえ
tty_last_login	はい	いいえ	はい	いいえ	いいえ
tty_last_unsuccessful_login	はい	いいえ	はい	いいえ	いいえ
ttys	はい	いいえ	はい	いいえ	いいえ
umask	はい	いいえ	はい	いいえ	いいえ
unsuccessful_login_count	はい	いいえ	はい	いいえ	いいえ
unsuccessful_login_times	はい	いいえ	はい	いいえ	いいえ
usrenv	はい	いいえ	はい	いいえ	いいえ

表 8. グループ属性と認証ロード・モジュールのサポート

ユーザー属性	Local	NIS	LDAP	PKI	Kerberos
admin	はい	いいえ	はい	いいえ	いいえ
adms	はい	いいえ	はい	いいえ	いいえ
dce_export	はい	いいえ	はい	いいえ	いいえ
id	はい	はい	はい	いいえ	いいえ
primary	はい	いいえ	はい	いいえ	いいえ
projects	はい	いいえ	はい	いいえ	いいえ
screens	はい	いいえ	はい	いいえ	いいえ
users	はい	はい	はい	いいえ	いいえ

ディスク・クォータ・システムの概要

ディスク・クォータ・システムを使用すれば、システム管理者は、ユーザーやグループに割り当てることができるファイルとデータ・ブロックの数を制御することができます。

ディスク・クォータ・システムの概念:

バークレー・ディスク・クォータ・システムをベースにしたディスク・クォータ・システムは、ディスク・スペースの使用を制御するための効率的な方法を提供します。 クォータ・システムは、個々のユーザーやグループに対して定義することが可能で、それぞれのジャーナル・ファイルシステム (JFS および JFS2) ごとに維持管理されます。

ディスク・クォータ・システムは、JFS ファイルシステムについては **edquota** コマンドで、JFS2 ファイルシステムについては **j2edlimit** コマンドでそれぞれ変更できる、以下のパラメーターに基づいて制限を設定します。

- ユーザーやグループのソフト制限
- ユーザーやグループのハード制限
- クォータ猶予期間

ソフト制限 は、正常操作時にユーザーまたはグループが使用することを許可される 1 KB のディスク・ブロックまたはファイルの数を定義します。ハード制限 は、設定されたディスク・クォータ以下の範囲内でユーザーが累積できるディスク・ブロックまたはファイルの最大量を定義します。 クォータ猶予期間 を指定すると、ユーザーは、短期間 (デフォルト値は 1 週間) だけソフト制限を超えることが可能になりま

す。 ユーザーが、指定された時間内に使用量をソフト制限以下に減らすことができなかった場合、システムは、そのソフト制限を許容最大割り当てとして解釈し、それ以上のストレージをそのユーザーに割り当てることはしません。 ユーザーは、使用量をソフト制限以下になるようにファイルを除去することで、この条件をリセットできます。

ディスク・クォータ・システムは、クォータを指定して使用可能にしたファイルシステムのルート・ディレクトリーに常駐する **quota.user** ファイルと **quota.group** ファイル内のユーザー・クォータとグループ・クォータをトラッキングします。 これらのファイルは、**quotacheck** コマンドと **edquota** コマンドによって作成され、クォータ・コマンドによって読み取ることができます。

オーバー・クォータ条件からのリカバリー**:**

ファイルシステムの使用量を減らすことにより、オーバー・クォータ条件からリカバリーすることができます。

クォータ制限を超えた場合にファイルシステムの使用量を減らすには、次の方法を使用することができます。

- ファイルシステムがその限度に達する原因となった現行プロセスを停止し、余分なファイルを除去して限度をクォータ未満にしてから、失敗したプログラムを再試行します。
- vi などのエディターの実行中は、シェル・エスケープ・シーケンスを使用して、ファイル・スペースを検査し、余分なファイルを除去してから、編集済みファイルを失わずに戻します。 C または Korn シェルを使用している場合は、これに代わる方法として、Ctrl-Z キー・シーケンスを使用してそのエディターを中断し、ファイルシステム・コマンドを出してから、**fg** (フォアグラウンド) コマンドで戻すことができます。
- クォータの限度に達していないファイルシステムにファイルを一時的に書き込み、余分なファイルを削除してから、そのファイルを正しいファイルシステムに戻します。

ディスク・クォータ・システムのセットアップ**:**

通常、ホーム・ディレクトリーとファイルが収められているファイルシステムだけが、ディスク・クォータを必要とします。

次の条件に該当する場合は、ディスク・クォータ・システムの実装を考慮してください。

- システムのディスク・スペースが限られている。
- より強力なファイルシステム・セキュリティーを必要としている。
- 多くの大学で見られるように、ディスク使用量レベルが高い。

これらの条件がご使用の環境に該当しない場合は、ディスク・クォータ・システムを実装して、ディスク使用量の限度を設けなくても差し支えありません。

ディスク・クォータ・システムは、ジャーナル・ファイルシステムだけで使用します。

注**: /tmp** ファイルシステムにはディスク・クォータを設定しないでください。

ディスク・クォータ・システムを設定するには、下記の手順を使用してください。

1. root 権限でログインします。
2. クォータを必要とするファイルシステムを決定します。

   注**:** 多くのエディターおよびシステム・ユーティリティーが **/tmp** ファイルシステムに一時ファイルを作成するため、このファイルシステムはクォータの対象外にしてください。

3. **chfs** コマンドを使用して、**/etc/filesystems** ファイルに **userquota** と **groupquota** の各クォータ構成属性を組み込みます。下記の例では、**chfs** コマンドを使って、**/home** ファイルシステムでのユーザー・クォータを使用可能にします。

   ```
   chfs -a "quota = userquota" /home
   ```

   /home ファイルシステムでユーザーとグループのクォータの両方を使用可能にするためには、次のように入力します。

   ```
   chfs -a "quota = userquota,groupquota" /home
   ```

   /etc/filesystems ファイルで、対応するエントリーが次のように表示されます。

   ```
   /home:
   dev         = /dev/hd1
   vfs         = jfs
   log         = /dev/hd8
   mount       = true
   check       = true
   quota       = userquota,groupquota
   options     = rw
   ```

4. オプションとして、代替ディスク・クォータ・ファイル名を指定します。 **quota.user** および **quota.group** ファイル名が、クォータが使用できるファイルシステムのルート・ディレクトリーに存在するデフォルト・ファイル名です。 **/etc/filesystems** ファイルの **userquota** 属性および **groupquota** 属性にこれらのクォータ・ファイルの代替名またはディレクトリーを指定できます。

   下記の例では、**chfs** コマンドを使って、**/home** ファイルシステムのユーザー・クォータとグループ・クォータを設定し、クォータ・ファイル名に **myquota.user** と **myquota.group** を指定しています。

   ```
   chfs -a "userquota = /home/myquota.user" -a "groupquota = /home
           /myquota.group" /home
   ```

   /etc/filesystems ファイルで、対応するエントリーが次のように表示されます。

   ```
   /home:
   dev         = /dev/hd1
   vfs         = jfs
   log         = /dev/hd8
   mount       = true
   check       = true
   quota       = userquota,groupquota
   userquota   = /home/myquota.user
   groupquota  = /home/myquota.group
   options     = rw
   ```

5. まだマウントされていない場合、指定されたファイルシステムをマウントしてください。
6. 各ユーザーまたはグループに対して必要なクォータ制限を設定します。 **edquota** コマンドを使用して、許容ディスク・スペースとファイルの最大数について、ユーザーまたはグループごとに、ソフト制限およびハード制限を設定します。

   以下のエントリーの例は、*davec* ユーザーのクォータ制限を示しています。

   ```
   Quotas for user davec:
   /home: blocks in use: 30, limits (soft = 100, hard = 150)
           inodes in use: 73, limits (soft = 200, hard = 250)
   ```

   このユーザーは、最大 100 KB のディスク・スペースのうち 30 KB を使用しています。 最大 200 のファイルのうち、*davec* は既に 73 のファイルを作成しています。 このユーザーは、一時ストレージに割り当てることができる 50 KB のディスク・スペースのバッファーと、 50 のファイルを持っています。

複数のユーザーに対してディスク・クォータを設定する場合には、**edquota** コマンドに **-p** フラグを使用して、あるユーザーのクォータを別のユーザー用に複製します。

davec というユーザーに対して設定したクォータを、*nanc* というユーザー用に複製するためには、以下のように入力します。

```
edquota -p davec nanc
```

7. **quotaon** コマンドを使用して、クォータ・システムを使用可能にします。 **quotaon** コマンドは、指定したファイルシステム、または **-a** フラグが指定されている場合には、クォータ対象のすべてのファイルシステム (**/etc/filesystems** ファイルに指示される) に対するクォータを使用可能にします。
8. **quotacheck** コマンドを使用して、実際のディスク使用状況に対するクォータ・ファイルの整合性を検査します。

   注**:** ファイルシステムでクォータを初めて使用可能にしたときと、システムのリブート後には、必ずこの検査を行うようにしてください。 **quotacheck** コマンドの実行には、同じサイズの JFS2 ファイルシステムよりも JFS ファイルシステムの方が時間がかかります。 クォータがリブート前ずっと使用可能になっている場合は、リブート時にファイルシステムに対して **quotacheck** コマンドを実行する必要はありません。

   システム始動中にこの検査を使用可能にしてクォータをオンにするためには、**/etc/rc** ファイルの最後に下記の行を追加します。

```
echo " Enabling filesystem quotas "
/usr/sbin/quotacheck -a
/usr/sbin/quotaon -a
```

許可されたグループの数

AIX 7.1 では、「許可されたグループの数」の値を構成し、取得できます。 この値は、ユーザーがメンバーになれるグループの数を定義します。

「許可されたグループの数」のデフォルト値は 128 です。 128 から 2048 の範囲で調整できます。 「許可されたグループの数」は、sys0 デバイスの v_ngroups_allowed システム構成パラメーターで指定されています。 カーネルまたは ODM データベースにある v_ngroups_allowed パラメーター値を変更または取得できます。 カーネル内のパラメーター値は、システムによって実行中に使用されます。 ODM データベース内のパラメーター値は、システムが再始動された後で有効になります。

ODM データベースからの「許可されたグループの数」の値の取得**:** v_ngroups_allowed パラメーターを取得するには、コマンドまたはサブルーチンを使用する必要があります。 ODM データベース内の v_ngroups_allowed パラメーターを取得するには、**lsattr** コマンドを使用する必要があります。

lsattr コマンドは v_ngroups_allowed パラメーターを ngroups_allowed 属性として表示します。 以下の例では、**lsattr** コマンドを使用して ngroups_allowed 属性を取得する方法を示します。

```
$lsattr -El sys0
SW_dist_intr    false           Enable SW distribution of interrupts            True
autorestart     true            Automatically REBOOT system after a crash       True
boottype        disk            N/A                                             False
capacity_inc    1.00            Processor capacity increment                    False
capped          true            Partition is capped                             False
conslogin       enable          System Console Login                            False
cpuguard        enable          CPU Guard                                       True
dedicated       true            Partition is dedicated                          False
ent_capacity    4.00            Entitled processor capacity                     False
frequency       93750000        System Bus Frequency                            False
fullcore        false           Enable full CORE dump                           True
fwversion       IBM,SPH01316    Firmware version and revision levels            False
```

```
iostat          false           Continuously maintain DISK I/O history          True
keylock         normal          State of system keylock at boot time            False
max_capacity    4.00            Maximum potential processor capacity            False
max_logname     20              Maximum login name length at boot time          True
maxbuf          20              Maximum number of pages in block I/O BUFFER CACHE True
maxmbuf         0               Maximum Kbytes of real memory allowed for MBUFS True
maxpout         0               HIGH water mark for pending write I/Os per file True
maxuproc        128             Maximum number of PROCESSES allowed per user    True
min_capacity    1.00            Minimum potential processor capacity            False
minpout         0               LOW water mark for pending write I/Os per file  True
modelname       IBM,7044-270    Machine name                                    False
ncargs          6               ARG/ENV list size in 4K byte blocks             True
pre430core      false           Use pre-430 style CORE dump                     True
pre520tune      disable         Pre-520 tuning compatibility mode               True
realmem         3145728         Amount of usable physical memory in Kbytes      False
rtasversion     1               Open Firmware RTAS version                      False
sec_flags       0               Security Flags                                  True
sed_config      select          Stack Execution Disable (SED) Mode              True
systemid        IBM,0110B5F5F   Hardware system identifier                      False
variable_weight 0               Variable processor capacity weight              False
ngroups_allowed 128             Number of Groups Allowed at boot time           True
$
```

カーネルからの「許可されたグループの数」の取得: カーネルから v_ngroups_allowed パラメーターを取得するには、sys_param サブルーチンを使用する必要があります。

```
#include<sys/types.h>
#include<sys/var.h>
#include<errno.h>
main()
{
 int rc;
 struct vario myvar;

 rc = sys_parm (SYSP_GET, SYSP_V_NGROUPS_ALLOWED, &myvar);

 if (!rc)
 printf("Number of Groups Allowed = %d¥n",
 myvar.v.v_ngroups_allowed.value);
 else
 printf("sys_parm() failed rc = %d, errno = %d¥n", rc, errno);
 }
```

ODM データベース内の「許可されたグループの数」の変更: システム・ブート・フェーズ時に、カーネル内の「許可されたグループの数」の値を構成する必要があります。 **chdev** コマンドを使用して ODM データベース内の値を変更してください。 この変更内容は、システムが再始動されたときに有効になります。

chdev コマンドを使用して ODM データベース内の **v_ngroups_allowed** パラメーターを変更するには、次のように入力します。

```
$ chdev -l sys0 -a ngroups_allowed=2048
sys0 changed
$
```

ロール・ベースのアクセス制御

システム管理は日常的な操作の重要な側面であり、セキュリティーはほとんどのシステム管理機能に固有の部分です。また、操作環境の保護に加えて、毎日のシステム稼働状況を緊密にモニターすることが必要です。

多くの環境では、異なるユーザーがそれぞれ異なるシステム管理の職務を遂行するよう求められています。これらの職務の分離を維持し、単一のシステム管理ユーザーが誤って、または故意に、システム・セキュリ

ティーを迂回できないようにする必要があります。 従来の UNIX システム管理ではこれらの目的を実現できませんが、ロール・ベースのアクセス制御 (RBAC) では可能です。

従来の UNIX 管理の制限

RBAC では、従来の UNIX システムの管理上の問題をいくつか解決しています。 問題には次のようなものがあります。

ルート管理アカウント

従来は、AIX および他の UNIX オペレーティング・システムには、ルート (通常は UID の 0 で指定されます) という名前の定義された単一のシステム管理者アカウントがあり、それによってシステム上のすべての特権システム管理タスクを実行することができます。 すべてのシステム管理タスクを単一のユーザーに依存することは、責務の分離に関して問題があります。 単一の管理アカウントは特定の環境で受け入れ可能であり、多くの環境では複数の管理者が必要となり、各管理者はさまざまなシステム管理タスクを受け持ちます。

管理責務をシステムの複数のユーザーで共有するために、従来は root ユーザーのパスワードを共有するか、または root ユーザーと同じ UID をもつ別のユーザーを作成するかのどちらかでした。 この方法でシステム管理責務を共有する場合、それぞれの管理者が完全なシステム制御権をもち、管理者が実行できる操作を制限する方法がないため、セキュリティー上の問題があります。 root ユーザーは最大の特権をもつユーザーであるため、許可されていない操作を実行することができ、そのような活動の監査を消去してこのような管理アクションを追跡できないようにします。

SUID による特権の拡大

UNIX オペレーティング・システムでのアクセス制御は、従来はアクセスを決定するプロセスに関連付けられている UID を使用することによって実行していました。 ただし、ルート UID の 0 によって、従来は許可検査が迂回できていました。 このため、root ユーザーとして実行しているプロセスは、すべてのアクセス検査に合格し、すべての操作を実行することができます。 これは、**setuid** アプリケーションの UNIX 概念に関するセキュリティー上の問題です。

setuid 概念により、コマンドを別の ID ひいてはコマンドを呼び出したユーザーの下で実行することができます。 これは、通常のユーザーが特権タスクを実行しなければならないときに必要なことです。 この例として、AIX **passwd** コマンドがあります。通常のユーザーは、ユーザー・パスワードを保管するファイルにアクセスできないので、ユーザーのパスワードを変更し、**passwd** コマンドを root ユーザーに対して **setuid** にするために追加の特権が必要です。 通常のユーザーが **passwd** コマンドを実行すると、オペレーティング・システムには、root ユーザーがファイルにアクセスしていてそのアクセスが許可されているように見えます。

この概念により確かに希望する機能が提供されますが、一方で固有のリスクを伴います。 **setuid** プログラムがルート・コンテキストで有効に実行されているため、アタッカーがプログラムの終了前にプログラムを支配することに成功した場合には、アタッカーがルートの全権限をもち、すべてのオペレーティング・システムのアクセス検査を迂回してすべての操作を実行することができます。 より良い解決方法は、root ユーザー特権の一部をプログラムに割り当てて、 91 ページの『最小特権の原則』に従い脅威を軽減することのみです。

RBAC のエレメント

RBAC により、システム管理に対するロールの作成、およびロール・システム・ユーザー・セット全体にわたる管理用タスクの代行が可能になります。 AIX では、RBAC は、通常、root ユーザーに予約されている管理機能が正規のシステム・ユーザーに割り当てられるメカニズムを提供します。

これを行うために、RBAC を使用して組織内にジョブ機能 (ロール) を定義し、それを特定のユーザーに割り当てます。 基本的には、RBAC はフレームワークであり、ロールを使用することによってシステム管理用として許可されます。 通常、ロールには環境に関する 1 つ以上の管理点をうまく取り扱うためのスコープについて定義されます。 ロールをユーザーに割り当てることにより、許可セットまたは特権セットを効果的にユーザーに与え、ユーザーの能力を高めます。 例えば、一方の管理ロールはファイルシステムの管理で、もう一方のロールはユーザー・アカウントの作成を使用可能にすることです。

RBAC 管理は従来の UNIX 管理と比較して、以下の利点があります。

- アカウント・アクセスを共有しなくても、複数のユーザーがシステム管理を行うことができる。
- 各管理者には必要以上の権限を付与する必要がないため、細かい管理によりセキュリティーを分離して管理できる。
- 最小の特権セキュリティー・モデルの実行を可能にする。 ユーザーおよびアプリケーションは必要な特権が必要になった場合のみ認可されます。したがって、システム・アタッカーから受けるインパクトを軽減できます。
- システム管理およびアクセス制御に関して、一貫して会社全体にわたるセキュリティー・ポリシーの実装と実施を可能にする。
- ロール定義を一度作成すると、ユーザーがジョブ機能を変更するときに、必要に応じてユーザーに割り当てたり、除去したりすることができる。

RBAC フレームワークは、以下の 3 つの中核概念を中心にしています。

- 権限
- ロール
- 特権

同時に、これらの概念により RBAC システムが最小の特権方針を実施することを可能にします。

権限:

権限はセキュリティー関連機能またはコマンドと関連付けられたテキスト・ストリングです。 権限 (Authorizations) とは、特権アクションを実行するための権利 (rights)、および異なるレベルの機能を別クラスのユーザーに提供するための権利 (rights) について、認可するメカニズムをユーザーに提供することです。

権限によって管理されるコマンドを実行するときに、呼び出し側のユーザーに必要な権限がある場合のみ、アクセスは認可されます。 権限は 1 つ以上のコマンドへのアクセス権限のロックを解除することができるキーとも言うことができます。 権限はユーザーには直接割り当てられません。 ユーザーは権限の集合であるロールを割り当てられます。

ロール:

ロールによりシステムの管理機能をグループ化してセットにまとめることが可能になります。 権限はキーであるという類似性を使用して、ロールは複数の権限を保持できるキー・リングと考えることができます。権限については、ロールに直接割り当てるか、またはサブロールを介して間接的に割り当てることが可能です。 分かりやすく言えば、サブロールは指定ロールの権限の継承元である別のロールです。

ロールはそれ自体でユーザーに追加の能力を認可しませんが、その代りに、権限の集合メカニズムおよび権限を別のユーザーに割り当てるための機能としてサービス提供します。 ロールの定義およびロールのユーザーへの割り当てを行うことにより、ユーザーが実行できるシステム管理タスクを決定します。ロールの定義後、ロール管理者は 1 ユーザーまたは複数のユーザーにロールを割り当てることにより、ロールに相当

する特権操作を管理することが可能になります。 また、ユーザーには複数のロールを割り当てることができます。 いったん、ロールがユーザーに割り当てられると、ユーザーはロールに割り当てられた権限を使用して、システムに関する管理コマンドへのアクセスのロックを解除することができます。

組織のポリシーと手続きに従って、ロールをユーザーに割り振る方法について決定します。 多くの権限をロールに割り当て過ぎたり、ロールを多くのユーザーに割り当て過ぎないようにしてください。 たいていのロールは管理担当のメンバーだけに割り当てるようにしてください。 これまで、root の能力は従来からトラステッド・ユーザーだけに付与されているように、ロールはトラステッド・ユーザーだけに割り当てるようにします。 ロールは正当なニーズのあるユーザーだけに必要な期間だけ認可します。 この手法により、権限のないユーザーが権限を獲得したり、または不正使用する機会を減らします。

特権**:**

特権はプロセスが特定のシステム制限をバイパスすることを許可するプロセス属性です。

特権メカニズムは、非トラステッド・アプリケーションに許可されていない機能を、トラステッド・アプリケーションに提供します。 例えば、特権はセキュリティー制約をオーバーライドする場合、特定のシステム・リソース (メモリーやディスク・スペースなど) の使用の拡大を許可する場合、およびプロセスのパフォーマンスや優先順位を調整する場合に使用できます。 特権とは、プロセスがシステムにおける特定のセキュリティー制約を乗り越えることを可能にする能力という概念です。

権限およびロールは、特権操作をアクセスするためのユーザーの能力を構成するユーザー・レベルのツールです。 一方、特権はカーネルで使用される制限メカニズムであり、プロセスが特別なアクションを実行することを許可されるかどうかの判別に使用されます。

特権はプロセスに関連付けられ、通常は、特権コマンドの呼び出しを経由して獲得されます。 これらの関連付けられた特権のために、プロセスは関連した特権操作を実行する資格を有することになります。 例えば、ユーザーがコマンドを実行する権限を持っているロールを使用すると、コマンドの実行時に特権セットがプロセスに割り当てられます。

最小特権の原則*:*

オペレーティング・システムでは、いくつかの操作に特権が与えられ、これらの操作を実行する許可は許可ユーザーに制限されています。 これらの特権操作には、通常、システムのリブート、ファイルシステムの追加と変更、ユーザーの追加と削除、およびシステムの日付と時刻の変更などのタスクが含まれます。

従来の UNIX システムでは、プロセスまたはユーザーは、通常モードまたは特権モード (スーパーユーザーまたは root とも呼ばれる) にすることができます。 通常のユーザーは特権操作を実行することができませんが、ルートとして実行しているプロセスでは、すべてのコマンドおよびすべてのシステム操作を実行することができます。 従来の UNIX システムは、非常に大まかな全か無かという特権の概念があり、過度の特権をもった管理者というセキュリティー上の脅威に直面していました。

単一の特権モードがシステムへのすべてのアクセスを許可する従来の UNIX の方式は、高度に保護されたシステムの要件に合わせるには粗すぎます。 保護されるように設計されたシステムでは、各プロセスに、タスクの実行に必要な特権の最も制限されたセットを必要とします。 特権には、特定の特権を要求するプロセスのみが該当の特権を許可されるようにする必要があるという利点があります。 この特権の制限は、最小特権のプリンシパルとして知られており、不注意な、または悪意のある管理者およびオペレーターによるシステムへの損傷を制限するために役立ちます。

例えば、パスワードの変更には、一般的には通常のユーザーがアクセス不能なファイルにアクセスするための特定の特権が必要となります。 ユーザーが常にこれらの特権をもっている場合には、セキュリティーの

観点からは望ましくない他のアクションも実行できます。 そのため、必須特権は **passwd** コマンドにのみ許可され、すべてのユーザーに許可されるわけではありません。

RBAC 環境では、ユーザー自身には初めから付与されている特権は何もありません。 ユーザーは単に特定のコマンドを実行することができ、それによって特権が許可されます。 代わりにユーザーが直接、特権を許可されている場合は、任意の時点でどんな方法でもその特権を使用することができます。 個別のコマンドに対する特権を制限することにより、特権が適用されるコンテキストを抑制することができます。 このことにより、トラステッド・アプリケーションがアタッカーによって不当に使用された場合にアタッカーが得るのは全特権をもつルートの全能力ではなく、限定された特権セットであるため、セキュリティーがより高まります。

トラステッド・アプリケーションに特権を与える前に、注意深く検査しなければなりません。 さらに、特権はアプリケーションに必要な時と場所で許可する必要があります。 トラステッド・アプリケーションは他のすべてのプログラムと同様ですが、1 つだけ異なるのは、トラステッド・アプリケーションは非トラステッド・アプリケーションには拒否されているアクションを実行することができるということです。

AIX RBAC

AIX では、AIX 6.1 より前は、制限付きの RBAC インプリメンテーションが提供されていました。

AIX 6.1 から、システム管理タスクを分割して非常に精細なメカニズムの新しい RBAC インプリメンテーションが提供されています。 この 2 つの RBAC インプリメンテーションは機能的に大きく異なるため、以下の用語が使用されています。

レガシー **RBAC** モード
: AIX 6.1 より前のバージョンに適用される AIX ロールの旧来の動作

拡張 **RBAC** モード
: AIX 6.1 で導入された新規インプリメンテーション

両方の操作モードがサポートされます。 ただし、拡張 RBAC モードが新しくインストールされた AIX 6.1 システムのデフォルトです。 以下のセクションでは、2 つのモードとその相違に関する簡潔な検討、および望ましい RBAC モードで運用するためのシステムを構成するために必要な情報を提供します。

レガシー RBAC モード:

AIX では、AIX 6.1 より前は、非 root ユーザーに特定のシステム管理タスクを許可する制限付き RBAC 機能が提供されていました。

この RBAC インプリメンテーションでは、指定された管理コマンドが非 root ユーザーによって呼び出される時に、コマンドのコードにより、ユーザーに必要な許可をもつロールを割り当てるかどうかが決定されます。 一致が検出された場合には、コマンドの実行が継続されます。 一致が検出されなかった場合は、コマンドは失敗し、エラーが出されます。許可された呼び出し側が操作を完了させるために必要な特権を得るように、許可によって制御されているコマンドを root ユーザーに対する **setuid** とする必要がしばしば生じます。

RBAC インプリメンテーションでは、管理コマンドへのアクセスを決定するために使用できる、事前定義されているがユーザーが拡張できる許可セットも導入しています。 さらに、ロールの作成、ロールへの許可の割り当て、およびユーザーへのロールの割り当てを行うための管理コマンドおよびインターフェースの枠組みも提供されています。

このインプリメンテーションによってシステム管理の責任を部分的に分割することができますが、そのためには次の制約があります。

1. 枠組みとして、コマンドおよびアプリケーションを RBAC 対応に変更する必要があります。
2. 事前定義許可は細分化できず、許可を作成するメカニズムは堅固なものではありません。
3. コマンドを実行するために所定の許可が指定されたロールをもつのと同様に、特定のグループのメンバーシップが、多くの場合、必要とされます。
4. 義務の分離の実施は困難です。 ユーザーが複数のロールに割り当てられている場合は、単一のロールの下で振る舞う方法がありません。 ユーザーには常に、すべてのロールに関するすべての許可があります。
5. 最小特権プリンシパルは、オペレーティング・システムでは採用されません。コマンドは通常、root ユーザーに対して SUID でなければなりません。

レガシー RBAC モードは互換性を保持するためにサポートされますが、拡張 RBAC モードはデフォルトの RBAC モードです。 拡張 RBAC モードは AIX 上で優先されます。

拡張 **RBAC** モード**:**

AIX 6.1 では、RBAC がより強力に実現されています。特定の操作に管理特権が必要なアプリケーションには、拡張 AIX RBAC インフラストラクチャーを用いた新しい統合オプションが加えられています。

これらの統合オプションは、細分化された特権と権限の使用、およびシステム上のすべてのコマンドを特権コマンドとして構成できるようにすることに重点を置いています。 拡張 RBAC モードの機能は、AIX 6.1 以降の AIX のすべてのインストールにおいて、デフォルトでインストールおよび使用可能にされます。

拡張 RBAC モードでは、下記の RBAC データベースを介して、権限、ロール、特権コマンド、デバイス、およびファイルの構成可能なセットが提供されます。 拡張 RBAC によって、データベースはローカル・ファイルシステムに常駐するかまたは LDAP を使用してリモート管理することができます。

- 権限データベース
- ロール・データベース
- 特権コマンド・データベース
- 特権デバイス・データベース
- 特権ファイル・データベース

拡張 RBAC モードでは、権限の階層を作成できるようにする権限の新しい命名規則が取り入れられています。 AIX にはシステム定義の権限の細分セットがあり、管理者は必要に応じて自由に追加のユーザー定義の権限を作成することができます。

ロールの性質が拡張されたことにより、負荷機能を分離することができるようになりました。拡張 RBAC では、ロール・セッションの概念が導入されています。 ロール・セッションは、1 つ以上の関連ロールを含んだプロセスです。 ユーザーは割り当てられたすべてのロールについてロール・セッションを作成することができるため、単一のロールまたはいくつかの選択されたロールを一度に活動化することができます。デフォルトでは、新しいシステム・プロセスにはいかなる関連ロールもありません。 ロールがさらに拡張されたことによって、アタッカーがユーザー・セッションを乗っ取ろうとしても、アタッカーはユーザーのロールの活動化を認証する必要があるため、アタッカーから保護するロールを活動化する前にユーザーが認証しなければならない要件をサポートすることができます。

特権コマンド・データベースの導入により、最小の特権原則が実装されます。 システム特権の細分度が増した結果、明示特権をコマンドに与えることができ、かつコマンドの実行を権限によって管理することができます。 これにより、コマンド自体に対するコード変更を必要とせずに、コマンド実行の許可検査を実施

する機能が提供されます。 特権コマンド・データベースを使用すると、必要な特権を割り当てるだけの機能は可能になるので、SUID および SGID アプリケーションは必要なくなります。

特権デバイス・データベースにより、特権によって管理されているデバイスにアクセスすることができます。一方、特権ファイル・データベースにより、非特権ユーザーは制限付きファイルに権限に基づいてアクセスすることができます。 これらのデータベースでは、システム管理タスクの細分度が増大して、従来は非特権のユーザーにこれらのタスクを割り当てることができます。

RBAC データベース内の情報は、収集および検査されてから、カーネルのセキュリティー・テーブル (KST) として指定されているカーネルの領域に送信されます。 KST 内のデータの状態によってシステムのセキュリティー・ポリシーが決まります。これは重要なことなので注意してください。 ユーザー・レベルの RBAC データベース内で変更されるエントリーは、この情報が **setkst** コマンドによって KST に送信されるまでセキュリティーの決定には使用されません。

RBAC モードの構成**:**

RBAC モードはカーネル内でシステム全体にわたる構成変数によって制御されます。 この変数は拡張 RBAC モードを使用可能または使用不可のいずれかにすることを指定します。

拡張 RBAC モードは、AIX 6.1 以降ではデフォルトで使用可能になります。 拡張 RBAC モードを使用不可にしてレガシー RBAC モードに戻すには、**sys0** デバイスで **chdev** コマンドを実行して、**enhanced_RBAC** 属性の値を **false** と指定します。 **enhanced_RBAC** 属性への変更を有効にするには、システムをリブートする必要があります。 拡張 RBAC モードを使用可能にするには、**enhanced_RBAC** 属性を **true** に設定してください。 プログラムで、**sys_parm()** システム・コールを使用してモードを設定または照会することもできます。

現行 RBAC モードを検索するには、システムで以下のコマンドを実行します。

```
lsattr -E -l sys0 -a enhanced_RBAC
```

以下のコマンドを実行してからシステムをリブートし、拡張 RBAC モードを使用不可にすることができます。

```
chdev -l sys0 -a enhanced_RBAC=false
```

WPAR 環境では、グローバル・システムからのみ RBAC モードを構成することが可能であり、システム上のすべての WPAR と同様にグローバル・システムに対して一様に影響を及ぼします。

レガシー **RBAC** モードと拡張 **RBAC** モードの比較**:**

既存のインターフェースおよび新規インターフェースは、システム構成を検査して新規コードで実行するか、または従来の動作のままで実行するように変更されました。

レガシー RBAC モードでは、コマンド自体のコード内で検査される権限のみ実施されます。 カーネルのセキュリティー・テーブル (KST) はコマンドの実行または許可検査には何も影響を与えません。 ユーザーが権限をもっているかどうかについて判別する場合は、すべてのユーザーの権限を検索して一致しているかどうかを検査するというレガシー RBAC モードの動作に従います。 **swrole** コマンドおよび **default_roles** および **auth_mode** 属性などの新規フィーチャーは、レガシー RBAC モードでは使用不可です。 ただし、新しい特権、権限、および権限用の管理コマンドは、レガシー RBAC モードでサポートされます。

下表はレガシー RBAC モードと拡張 RBAC モードとの相違に関するリストです。

表 9. レガシー *RBAC* モードと拡張 *RBAC* モードとの相違

フィーチャー	レガシー **RBAC**	拡張 **RBAC**
ロールの活動化	ユーザー・ロールのすべてが常に活動状態になる	デフォルトで、**swrole** コマンドを経由して明示的にロールを担うまで、ロールは非活動状態である
default_roles 属性	使用不可	サポートされる
swrole コマンド	使用不可	サポートされる
ロール管理コマンド	サポートされる	サポートされる
権限管理コマンド	サポートされる	サポートされる
権限階層	各権限が独立している。 階層機能はない。	権限階層の概念はサポートされており、権限を他の権限の親にすることができる
許可検査	コマンド自体が権限について検査する場合のみ実施される	特権コマンド・データベースおよび/またはコマンド自体によって実施される
細分特権	サポートされる	サポートされる
pvi コマンド	使用不可	サポートされる
カーネルのセキュリティー・テーブル	使用不可	サポートされる
RBAC データベースのロケーション	ローカル・ファイル	ローカル・ファイルまたは LDAP

拡張 RBAC の使用

システム管理者は RBAC を効果的に使用するために、以下の領域について精通している必要があります。

RBAC 権限:

権限はロール・ベース・アクセス制御 (RBAC) には重要は部分です。 オペレーティング・システムは権限文字列を使用して、特権操作を実行する前に資格が有ることを判別します。 関連検査については、明示的にコード内から行うか、保護されている特権実行可能プログラムを実行しているときにローダーに実行させることができます。

権限文字列の命名によって特権操作が意味する制御の内容を示します。 権限に対する AIX 命名規則は階層構造がサポートされており権限のテキスト名で表されます。 AIX 権限文字列はドット表記のフォーマットを使用して、権限階層を記述します。例えば、新規ファイルシステムを作成するための権限は **aix.fs.manage.create** です。この権限はロールに組み込まれ、このロールを割り当てられるユーザーは AIX ファイルシステムを作成できます。親の権限 **aix.fs.manage** がロールに組み込まれると、このロールを割り当てられるユーザーは、ファイルシステムの作成と同様に、他のファイルシステム管理を実行できます。

AIX RBAC はシステム提供の権限 (システム定義の権限) とインストール後に作成される権限 (ユーザー定義の権限) を区別します。

システム定義権限*:*

AIX は事前定義で変更できない権限のセットを提供します。 これらはシステム定義権限と呼ばれています。 これらの許可は各種の特権 AIX 操作に関連付けられます。関連付けは特権コマンド・データベースに指定します。

システム定義権限階層の最上位は **aix** 権限です。この権限はその他の全システム定義権限の親になります。この権限をロールに認可するということは、あらゆるシステム定義権限をそのロールに認可することになります。 AIX システム定義権限の完全セットおよび各権限の要旨を表示するには、以下のコマンドを実行します。

```
lsauth –f –a description ALL_SYS
```

上のコマンドの出力では、システム定義権限はマルチレベル階層で表示されます。 例えば、**aix** 権限には数個の直接の子があります。 その子は、それぞれ別の階層の親です。 **aix.fs** 権限には複数の子の権限が含まれます。さらに **aix.fs.manage** も含まれますが、これにも、また **aix.fs.manage.change** および **aix.fs.manage.create** などの複数の権限が含まれます。

ユーザー定義権限*:*

システム定義権限に加えて、AIX RBAC では、システム管理者が独自のカスタム権限を権限データベース (/etc/security/authorizations) に定義することができます。 このような権限は、ユーザー定義権限と呼ばれています。

システム管理者は、ユーザー定義権限の追加、変更、または削除を行うことができます。 例えば、システム管理者は、ユーザー定義権限を作成することによって一部のユーザーが特権コマンドを実行できるようにし、この権限をコマンドに関連付けて、ユーザーに割り当てられているロールに権限を認可することができます。

ユーザー定義権限は、システム定義権限と同じ階層概念をサポートします。 ただし、AIX のユーザー定義権限の命名については、次の制限があります。

- ユーザー定義権限は、新しい最上位の親より下位に定義されなければなりません。 すなわち、ユーザー定義権限はシステム定義権限 **(aix)** の子になることはできません。
- 権限名には最大 63 文字の印刷可能文字を指定することができます。
- 権限の親階層には最大 8 のレベルを指定することができます。
- 権限にはいくつでも直接の子を指定できますが、直接の親は 1 つだけ指定できます。 2 つの独立した権限には、同じ直接の子を指定することはできません。

この階層ではエレメントに複数の直接の親を指定することができないので、既存のシステム定義権限の親であるユーザー定義権限を作成することはできません。 そのため、**aix.custom** という名前の権限を作成しようとすると失敗し、**custom.aix** という名前の権限の作成は新しい権限をもたらし、**aix** システム定義権限の親としては機能しません。

複数のソフトウェア・コンポーネントにわたって権限名間の競合を避けるために、ユーザー定義権限の作成時には次の構文を使用することをお勧めします。

vendor_name.product_name.function.function1.function2...

vendor_name
ソフトウェア・モジュールのベンダーの名前を示します。

product_name
RBAC によって管理されるプロダクトの高水準プロダクト名。

function, function1, function2 ...
これらの文字列は、RBAC によって管理される機能を表します。 これらの文字列は、これらの機能の編成方法を示す階層表記も提供します。

例えば、**ibm.db2.manage** は、IBM DB2 データベースの組の管理面を表している可能性があります。 前述のように、*vendor_name* 文字列の **aix** は AIX 用に予約されていて、ユーザー定義権限に使用することはできません。

システム管理者がユーザー定義権限のリスト、作成、変更、および削除に使用できるいくつかの権限管理コマンドがあります。 ユーザー定義権限は、**mkauth** コマンドを使用して作成、**chauth** コマンドを使用し

て変更、**rmauth** コマンドを使用して削除、**lsauth** コマンドを使用して表示することができます。ユーザー定義のすべてのシステム権限およびそれぞれの要旨を表示するには、次のコマンドを実行してください。

```
lsauth –f –a description ALL_USR
```

ユーザー定義権限を作成する前に、次の問題について検討してください。

- 新しいユーザー定義権限を作成するより既存のシステム定義権限を使用する方が適切ではありませんか。
- 新しい権限は既存のユーザー定義の権限階層よりも下位に属するか、または新しい階層の最初の権限ですか。
- これが新しい階層である場合、構造は何ですか。
- 権限のテキスト記述は何ですか。
- 権限記述の言語変換は必要ですか。
- 権限の作成時に特定の権限 ID を指定する何らかの理由がありますか。 権限 ID の生成には **mkauth** コマンドを使用することをお勧めします。

これらの問題を検討した上で、次の手順により権限を作成してください。

1. 言語変換が必要な場合は、メッセージ・カタログに記述を作成または追加する。
2. 親権限がまだ存在しない場合は、**mkauth** コマンドを使用して階層内のすべての親権限を作成する。
3. **mkauth** コマンドを使用して所望の権限を作成する。 特定の値が必要な場合は、このコマンドを使用して **id** 属性を指定してください。

レガシー権限の移行*:*

AIX バージョン 6.1 より前の場合、オペレーティング・システムは、オペレーティング・システムにより認知された制限付き、事前定義の権限セットを所有していました。 これらの権限はシステムのいずれのファイルに定義されなくても、すぐにロールに割り当てることができました。 これらのレガシー権限を新しい AIX バージョン 6.1 以降の RBAC フレームワーク内でサポートするために、これらのレガシー権限がユーザー定義の権限として定義され、デフォルトで権限データベースに提供されます。

AIX オペレーティング・システムは新規権限の命名規則に移行し、AIX オペレーティング・システムでの従来の権限名に対する検査は変更されて、新規の対応する権限を検査することが追加され、いずれかの権限がプロセスに存在していればアクセスが許可されます。下表は既存の事前定義の権限および対応する新規システム定義の権限のリストです。

既存の **AIX** 権限	対応する新規権限
Backup	aix.fs.manage.backup
Diagnostics	aix.system.config.diag
DiskQuotaAdmin	aix.fs.manage.quota
GroupAdmin	aix.security.group
ListAuditClasses	aix.security.audit.list
PasswdAdmin	aix.security.passwd
PasswdManage	aix.security.passwd.normal
UserAdmin	aix.security.user
UserAudit	aix.security.user.change
RoleAdmin	aix.security.role
Restore	aix.fs.manage.restore

RBAC ロール:

ロールとは、権限をシステム管理タスクのセットにまとめて、ユーザーに割り当てるために使用されるメカニズムのことです。 AIX のロールは、基本的には権限の集合のためのコンテナーです。

AIX は権限をロールへ直接割り当てること、または権限をサブロールを介して直接割り当てることをサポートします。 サブロールは、あるロールに対してそのロールの **rolelist** 属性に指定できます。 指定されたサブロールが持っているロールを構成する場合は、サブロールにある権限のすべてをそのロールへ有効に割り当てます。

ロールをユーザーへ割り当てることにより、ユーザーはロールのアクセスを許可され、そのロールに含まれる権限を使用できるようになります。 システム管理者はロールを複数のユーザーに割り当てることが可能であり、複数のロールを 1 ユーザーに割り当てることも可能です。 複数のロールを割り当てられたユーザーは、システム管理機能を実行する必要がある場合は、複数のロール (最大 8 ロール) を活動化することができます。

AIX は事前定義されたシステム管理用のロール・セットを提供します。 ただし、お客様が自分自身のカスタム・ロールを作成すること、あるいは既存の事前定義ロールを変更することをお勧めします。 AIX ロールをリスト、作成、および除去することができる、いくつかのロール管理コマンドが使用可能です。 **mkrole** コマンドでロールの作成、 **chrole** コマンドでロールの変更、 **rmrole** コマンドでロールの除去、 **lsrole** コマンドでロールの表示を行うことができます。

新規の AIX ロールを作成する場合は、以下の課題について考慮してください。

- ロールの名前?
- ロール名はテキスト・ストリングであり、ロールの機能を推察できる名前にすることをお勧めします。 ロール名には最大 63 印刷可能文字を使用できます。
- ロールに必要な権限は何か? 権限をロールに直接割り当てるか、またはサブロールを介して間接的に割り当てるかを検討します。
- ロールを活動化するときに、ユーザーの認証を必要とするべきか?

ロールの活動化:

拡張 RBAC を備えた AIX バージョン 6.1 以降のデフォルトでは、ユーザーがシステムに認証を行う際に、そのユーザーのセッションにはロールまたは権限が関連付けられません。ロールをセッションに関連付けるためには、ユーザーは個別認証コマンド (**swrole** コマンド) を呼び出して、1 つ以上のロールに切り替えなければなりません。

ユーザーが活動化できるのは、以前ユーザーに割り当てられていたロールのみです。 デフォルトでは、ユーザーはロール・セッションに入る時またはロールのセッションへの追加時にそれ自身として認証する必要があります。ロールをオプションで指定して **auth_mode** ロール属性を伴う認証を必要としないようにすることができます。

新しいロール・セッションへの切り替えにより、前のセッションからロールを継承せずに新しいシェル (セッション) が作成されます。 これは、ロール用の新しいプロセス・シェルの作成およびプロセスへの新しいロール ID (RID) の割り当てによって完了します。 新しいセッションの作成は、プロセスのロール ID のみが変更されて UID または GID のような特性でない場合を除き、**su** コマンドを使用するのと同じです。 **swrole** コマンドにより、ユーザーは単一のロールまたは複数のロールから成るロール・セッションを作成することができます。 ユーザーが現行のロール・セッションから新規ロール・セッションに切り替

えることを防止する制限はありません。 新しいセッションは新しいプロセスであるため、新しいセッションは前のセッションからいかなるロールも継承しません。 前のセッションを復元するためには、ユーザーは現行のロール・セッションを終了しなければなりません。 セッションで想定されたロール (活動ロール・セット) は、セッション中に **rolelist** コマンドを実行することによってリストすることができます。管理者は、**rolelist** コマンドを使用して、所定のシステム・プロセスの活動ロール・セットをリストすることもできます。

ユーザーは、新しい **default_roles** ユーザー属性をもつデフォルトのロール・セットをオプションで割り当てられることがあります。 この属性は、任意のユーザーのために作成されるプロセスが常に所定のロール・セット (例えば **cron** コマンド)に関連付けられる必要がある状態を対象としています。 cron 機能はバックグラウンドで実行し、定義されたユーザーとしてコマンドを実行します。 実行されるコマンドのいくつかに許可が必要になる可能性があります。 これには、**cron** コマンドには後でロールを獲得するメカニズムがないため、ロールのセットを常にユーザー ID のために活動化するよう指定する機能が必要です。**default_roles** 属性は、8 つまでのロール名または特殊値 **ALL** を組み込むように設定することができます。 **default_roles=ALL** の設定により、ユーザーのすべてのロールがセッションに割り当てられます。ユーザーが 9 以上のロールを割り当てられていた場合には、最初の 8 つのロールのみがセッションで使用可能です。

セッション当たりの最大ロール数*:*

拡張 RBAC では、システム管理者は、提供されたロール・セッションで活動状態にすることができる、システム全体をベースにしたロールの最大数について構成できます。 デフォルトで、ユーザーはセッション当たり 8 個までのロールを活動状態にすることができます。

ある環境では、ユーザーが一時点で単一のロールだけを活動化することができるように、任務をより細かく分割することが必要になる場合があります。 このような環境では、セッション当たりの最大許容ロール数を制限するために、 `/etc/security/login.cfg` ファイル内の **usw** スタンザの **maxroles** 属性を変更することができます。セッション当たりの最大許容ロール数を指定するには、 **maxroles** 属性を 1 から 8 までの範囲の値に設定します。

セッション当たりのロール数の制限について現在値を表示するには、以下のコマンドを実行します。

```
lssec –f /etc/security/login.cfg –s usw –a maxroles
```

ユーザーがシステムを変更して、一時点で単一のロールのみ活動状態にするには、以下のコマンドを実行します。

```
chsec –f /etc/security/login.cfg –s usw –a maxroles=1
```

maxroles 属性値の変更は、作成される新規ロール・セッションのいずれに対しても即時に有効になりますので、システムのリブートは必要ありません。値を変更する前に存在していたロール・セッションは、変更による影響を受けません。 セッション当たりの最大ロール数の施行は、セッションの開始時に行われます。

事前定義ロール*:*

ロールの事前定義セットは、新しい AIX バージョン 6.1 以降のインストールのローカル・ロール・データベース (`/etc/security/roles`) に定義されます。このロール・セットは標準的な管理責務をグループ分けすることを目的にしています。

このロール・セットは管理任務を分割するための推奨方法としてサービス提供されます。 ロール管理者はこれらのロールを変更したり除去したりすることが可能であり、また、必要に応じて環境に合わせて新しいロールを作成することも可能です。 以下のリストは、提供されるロールおよび各ロールの能力の要旨です。

ロール名	ロールの説明
auditadm	監査管理者。 auditadm ロールには、システム全体、シングルユーザー、単一ロールの属性を含めて、システムの監査およびロギングのポリシーを構成する責務があります。 このロールは、監査証跡の表示に対するアクセス権限を持っています。
fsadm	ファイルシステム管理者。 fsadm ロールは、ファイル・システムを作成し、それらをシステム上でユーザーが使用できるようにします。 以下は fsadm ロールの業務の一部です。 • マウント・ポリシーの指定 • ポリシーの共有 • クォータの割り当て • 圧縮レベルの決定 • ファイルシステムのフォーマットの設定 • バックアップおよびリストア・アクティビティーの実行
isso	情報システムのセキュリティー担当者。 ISSO の業務はロールの作成と割り当てです。したがって、システムでは最も強力なロールです。 以下は ISSO の業務の一部です。 • セキュリティー・ポリシーの設定および維持 • ユーザーのパスワードの設定 • ネットワーク構成 • デバイス管理
pkgadm	ソフトウェア・パッケージ管理者。 pkgadm ロールはシステム上にインストールされているソフトウェアに責任を持ち、システム・ソフトウェアをインストール、アップデート、削除するデフォルトのアクセス権を持っています。
sa	システム管理者。 SA ロールは日常の管理のための機能を提供します。以下のような業務があります。 • ユーザー管理 (パスワード設定を除く) • ファイルシステム管理 • ソフトウェアのインストールおよび更新 • ネットワーク・デーモン管理 • デバイス割り振り

ロール名	ロールの説明
secadm	セキュリティー管理者。 secadm ロールは、システム上のセキュリティー設定を保守します。 secadm は、グループ内のメンバーシップ、ロール、権限、クリアランスなどの属性をユーザーに割り当て、それぞれのロールでまだ指定されていないロールを割り当てます。 secadm ロールはまた、RBAC 設定、アクセス制御リスト、所有権、メンバーシップを含むセキュリティー属性も、システム・オブジェクトに割り当てます。 以下は secadm ロールの業務の一部です。 • 新規ユーザー・アカウントへのパスワードの割り当て • ロックされたアカウントのアンロック
so	システム・オペレーター。 SO ロールは毎日の運用のための機能を提供します。以下のような業務があります。 • システムのシャットダウンおよびリブート • ファイルシステムのバックアップ、復元および割り当て • システム・エラー・ログ作成、トレースおよび統計 • ワークロード管理
svcadm	サービス管理者。 svcadm ロールは、システム・サービスを使用可能化、構成、および使用不可化します。 このロールは、IP アドレス、経路、ホスト名、およびファイアウォール・ポリシーなどのネットワーキング属性の構成を許可します。
sysop	システム・オペレーター。 sysop ロールは、システム診断の実行および日常的なシステム保守の実行を含むアクセス権を使用して、システム全体を保守します。 以下は sysop が責任を持つタスクの一部です。 • ログ・ファイルと印刷キューの消去 • システムの停止と再始動
useradm	ユーザー管理者。 useradm ロールは、パスワードを管理せずに、ユーザー保守に関連する高水準のタスクに責任を持ちます。 useradm は、デフォルトのセキュリティー設定による定義に従って、ユーザー・アカウントを作成、変更、および削除します。 このロールはまた、デフォルトのセキュリティー設定で追加のロールおよびグループも作成します。

ロールの移行*:*

AIX バージョン 6.1 より前の AIX システムが移行インストールにより AIX 拡張 RBAC レベルへ更新されている場合、 `/etc/security/roles` ファイルの移行に際して、現行のロール能力が維持されていれば、新規機能のファイルの更新が試みられます。

ファイルの役割定義が保存されて、固有ロール ID を組み込む変更だけで、このロールは新規フレームワークで適切に機能するようになります。事前定義の権限として知られていない `/etc/security/roles` ファイルに定義されている権限は、いずれもユーザー定義の権限と見なされます。移行では、これらの権限名はローカル `/etc/security/authorizations` 権限データベースのエントリーとして追加されます。 旧ロール

定義の移行に加えて、新規の定義済みロールがこのファイルに追加されます。 移行後は、権限とロールが環境のニーズに合せて定義されていることを、システム管理者が検証する必要があります。

RBAC 特権**:**

拡張 RBAC フレームワークは、システム特権への依存度が高く、非特権ユーザーが特権タスクを実行することを許可します。 特権とは、プロセスがシステム・コールで機能を補強することを認可するために使用されるメカニズムのことです。

特権の概念は、定義と多くの検査がカーネルで行われるため、基本的にはカーネル・レベルの構造です。しかし、ユーザー・レベルのインターフェースが提供され、特権をコマンド、デバイス、およびプロセスに割り当てる処理が行われます。

特権と権限の違いに注意することが大切です。特権と権限はいずれも、システム・セキュリティー・ポリシーに対する特定の許容可能な例外を制御するために使用されます。 特権と権限の違いを定義すると、特権とは特定のプロセスに関連付けられることであり、一方、権限とはロールを介してユーザーに関連付けられることです。 権限にはロールとそのロールを持つユーザーが共に備わっており、実行されているプログラムには依存しません。 特権にはプログラムが備わっており、システム・セキュリティー・ポリシーを精細に調整するメカニズムを提供します。 これらの関連付けられた特権のために、プロセスは関連した特権操作を実行する資格を有することになります。

特権は特権操作全体のアクセス制御を実施するビット・マスクの個別ビットで指定して、 AIX カーネルに定義されます。 AIX では 100 より多い特権が提供され、特権操作の非常に精細な制御を提供しています。 システム・コールでのアクセスを決める場合に、カーネルはプロセスが必要な関連特権ビットを設定されているかどうかを判別して、要求を認可または否認します。

特権は特権コマンド・データベースを介してコマンド呼び出しに割り当てられ、特権がデバイスの制御アクセスに使用されるのは、特権デバイス・データベースを介して行われます。

特権の命名および階層*:*

AIX 特権については、システム管理者が作成、変更または削除することはできません。

使用可能な特権リストおよび特権の要旨は、以下のコマンドを実行してシステムに表示できます。

```
lspriv -v
```

AIX で提供される特権は AIX 特権にリストされます。 すべての AIX 特権は **PV_** で始まる特権ビットのテキスト表記になっています。 **PV_** 接頭部の後の命名規則は特権と特権の間の階層関係で示されます。例えば、監査特権 **PV_AU_** は特権 **PV_AU_ADD**、**PV_AU_ADMIN**、**PV_AU_READ**、**PV_AU_WRITE**、および **PV_AU_PROC** の親です。特権について検査するときに、システムは最初にプロセスが必要な最も低位の特権を持っているかどうかを判別し、続いて、より強力な特権の存在について、階層を上げて検査を進めます。 **PV_ROOT** 特権は **PV_SU_** を除く、すべての特権の親を表す特殊な特権です。 **PV_ROOT** 特権を割り当てられたプロセスは、 **PV_SU_** を除くシステム上のあらゆる特権を割り当てられたかのように振る舞います。

プロセス特権セット*:*

複数の特権セットがカーネルで定義され、それによって特権操作をさまざまな方法で制御することができます。 複数の特権セットにより、オペレーティング・システムで動的特権制御を実行することができ、さらにアプリケーションで最小特権プリンシパルを管理することができます。

特権は、次の特権セットを用いてプロセスに関連付けられます。

Limiting Privilege Set (LPS)
指定されたプロセスに使用される特権に強力な制限を定義します。 システムでのいかなる特権の拡大も、この値を超えてプロセスの特権を上げることはできません。 つまり、定義されているどのシステム・インターフェースを使用しても、プロセスはこの値より多い特権を獲得することはできません。 言い換えれば、プロセスはどの時点においてもこれらの特権に限定されます。 このことは、残りの特権セットは常に LPS のサブセットになるということも意味します。 LPS を拡張することはできませんが、すべてのプロセスには LPS を削減する権利が与えられます。 ただし、いったん LPS を削減すると、元の値に拡張して戻すことはできません。 LPS の値を下げることにより、プロセスでは関連する特権に関する境界を制限することができます。 例えば、カスタム・ユーザー提供のプログラムを実行する直前に、プロセスにより LPS が削減されることがあります。 デフォルトでは、システムで使用可能なすべての特権が、プロセスのために LPS に設定されます。

Maximum Privilege Set (MPS)
プロセスが使用を許可されている特権の全セットです。 MPS には LPS にあるどの特権も含めることができますが、LPS より多く含めることはできません。 MPS は、多くの理由でプロセスの存続期間内に変更することができます。 その理由とは次のようなものです。

- 現在のプロセスが別の特権コマンドを実行してから、関連の追加特権を獲得する場合。
- プロセスに適切な特権がある場合には、MPS を動的な方法で方針に基づいて拡張することができます。

Effective Privilege Set (EPS)
プロセスのために現在活動状態である特権のリストです。 EPS は常にプロセスの MPS のサブセットであり、カーネルで使用されて特権操作に関するアクセス検査を実行します。 EPS はプロセスによって操作することができ、MPS と同じにすることができますが、MPS を超えることはできません。 EPS の動的操作は、最小特権プリンシパルを実施するプロセスによって実行することができます。例えば、ユーザー・スペース・コードにより、監査関連のシステム・コールまたはカーネル・コールを行う前に **priv_raise** API を使用して、EPS 中の監査特権ビットを潜在的に上げることができます。 特権は、監査呼び出しが戻された時に **priv_lower** API を使用して下げることができます。

Inheritable Privilege Set (IPS)
親プロセスから子プロセスの MPS および EPS に渡される特権です。 IPS には LPS にあるどの特権も含めることができますが、LPS より多く含めることはできません。 IPS は、次の方法でプロセスに設定することができます。

- プロセスに適切な特権がある場合には、**setppriv** システム・コールを使用して IPS を方針に基づいて拡張することができます。
- 特権コマンドの実行時に、そのコマンドに関連付けられている **inheritprivs** 属性に指定されている特権は、IPS に割り当てられます。

Used Privilege Set (UPS)
プロセスの存続中にアクセス検査に使用された特権を示します。 UPS を使用して、プロセスで必要とされる特権を判別することができます。 指定された特権がプロセスにあるかどうかをカーネルが検査する時に、成功した検査が特権の UPS に保管されます。

Workload Partition Privilege Set (WPS)
システム WPAR では、グローバル WPAR で許可されているすべての特権操作を許可しないように制限することができます。 システム WPAR で許可されている特権操作は、WPS を用いて制御することができます。 グローバル・ルートは、WPS を使用して、制限された特権セットを WPAR に割り当てることができます。 WPS は、`/etc/wpar/secattrs` 構成ファイルで、または

/usr/sbin/startwpar コマンドを使用して WPAR の開始時に指定することができます。 WPAR で実行中のすべてのプロセスには、それぞれの WPS と同等の LPS があります。

システム管理者は、管理コマンドを使用して、プロセスのさまざまな特権セットをリストおよび変更することができます。 **lssecattr** コマンドを使用して、LPS、MPS、EPS、IPS、および UPS をリストすることができます。 **setsecattr** コマンドを使用して、LPS、MPS、EPS、および IPS を変更することができます。 UPS は読み取り専用属性であるため、**setsecattr** コマンドによって変更することはできません。

特権コマンド・データベース:

許可、ロール、および特権によって、細分化されたセキュリティー管理を実行することができます。 ただし、さまざまなシステム操作による RBAC の活用によって RBAC セキュリティー・ポリシーを適用することができます。

従来はいくつかの AIX コマンドが直接、許可を検査するのと同時に、実行可能コード自体を検査の実行のために変更する必要がありました。 拡張 RBAC モードでは、システムの実行可能ファイルへの変更を必要とせずに特権コマンド・データベースを使用して、許可検査を実行し関連する特権を付与するためのフレームワークが提供されます。

特権コマンド・データベースでは、ユーザーが通常は実行できないコマンドを実行するために必要な、またはタスクを実行する適切な特権を得るために必要な、コマンドへのアクセスおよび権限が許可されています。 このデータベースでは、特定のコマンドの許可情報とともに、許可検査が成功した場合にプロセスに対して許可される特権を保存しています。 このデータベースは、ローカルに保管されている場合は、`/etc/security/privcmds` ファイル内に存在し、情報のスタンザが command-versus-security 属性の形式で入っています。次に、このデータベースのいくつかのキー属性を挙げます (すべての属性の詳細については、`/etc/security/privcmds` ファイルを参照してください)。

accessauths
コマンドの実行を保護するアクセス許可のリスト。リストされている許可のいずれか 1 つをもつユーザーは、コマンドを実行することができ、さらにコマンドに含まれている特権操作の一部または全部を行うことができます。

innateprivs
固有の特権は、呼び出し側がアクセス許可検査に成功した場合にプロセスに割り当てられる特権です。

authprivs
許可されている特権は、ユーザーに関連する許可がある場合にプロセスに割り当てられる追加の特権です。 この属性では、コマンドのより細分化された制御が可能になり、制限された一連のユーザーが追加の特権操作を実行することができます。

inheritprivs
継承可能な特権は、プロセスから子プロセスに渡される特権です。

secflags
セキュリティー・フラグのリスト。 FSF_EPS は、コマンドの実行時に最大特権セット (MPS) をロードして有効特権セット (EPS) にするためのフラグです

拡張 RBAC モード・システム上のユーザーがコマンドの実行を試みると、そのコマンドは最初に特権コマンド・データベースで検査されます。 そのコマンドが特権コマンド・データベースに存在している場合は、ユーザーのセッションおよびコマンドの **accessauths** 属性の値に関連付けられている許可に対して検査が行われます。 リストされている許可の 1 つがセッションにある場合には、ユーザーは、コマンドに対する DAC 実行検査の結果に関わらずコマンドを実行することができます。 呼び出し時は、コマンド・プ

ロセスには、最大特権セット (MPS) に割り当てられた **innateprivs** 属性の中にリストされている特権があります。 追加の許可検査は、**authprivs** 属性にリストされている「許可 ? 特権」のペアによって実行されます。 リストされている許可の 1 つがセッションにある場合には、関連する特権もコマンド・プロセスの MPS に追加されます。 **secflags** 属性で設定されている **FSF_EPS** 値をもつ特権コマンド・データベース内のコマンド・エントリーは、コマンドの呼び出し時に MPS 中のすべての特権を有効特権セット (EPS) に割り当てます。

コマンドは、特権コマンド・データベース内にあるときは特権コマンドとして認識されます。 特権コマンド・データベースにリストされていない setuid プログラムは技術的には特権コマンドのままですが、RBAC の性質を記述する時に特権コマンドとして参照されることはありません。 コマンドに特権コマンド・データベース中のエントリーがない場合には、そのコマンドは特権コマンドではなく、そのコマンドへのアクセスは DAC およびコマンド自体によって実行されます。 さらに、コマンドが特権コマンド・データベースにリストされている場合で、ユーザーのセッションにコマンドの呼び出しを許可する権限がない場合は、システムは DAC アクセス検査に戻り、その検査に問題がなければコマンドの実行を許可します。

いくつかの管理コマンドが特権コマンド・データベースの操作および照会のために作成されています。 特権ファイル・データベースの中のエントリーは、 **setsecattr** コマンドによってリストすること、 **lssecattr** コマンドによって表示すること、および **rmsecattr** コマンドによって除去することが可能です。

コマンドに必要な権限の判別*:*

多くのシステム管理アプリケーションでは、適切に実行するための権限が必要です。 特権コマンド・データベースに事前定義されたコマンド・セットが提供されますが、システム管理者はそれぞれの環境に固有のエントリーを追加する必要が生じる場合があります。 特権コマンド・データベースにより、データベースにエントリーを追加することができます。 このコマンドにアクセスできるようにするには、適切な権限が **accessauths** 属性にリストされなければなりません。

AIX オペレーティング・システムでは拡張 RBAC フレームワークを使用して、次の 2 とおりの方法で権限の使用と検査ができます。

- Access Auths (Access Authorization): 特権コマンド・データベースで指定される属性で、権限名のコンマ区切りリストが含まれています。 現在のセッションのユーザーが、リストにある権限のいずれかをもっていれば、コマンドを実行できます。 これは、保護された特権実行可能プログラムの実行時に、システム・ローダーによって検査されています。
- Check Auths (checkauths()): checkauths() API を使用して、特定の権限または権限のリストをプログラムで検査できます。 指定された権限は現在のセッション内のロールに存在する権限に対して検査されます。 この検査の結果に基づいて、プログラムによっては特権操作を実行します。

コマンドを特権コマンド・データベースに追加する前に、コマンドを確実に実行できるように権限セットを判別する必要があります。プログラムまたはアプリケーションによっては、追加の権限検査を内部で実行します。 カスタム・ロールを作成する際に、プロセスで使用する割り当て可能な権限リストを判別する必要があります。

次の基本方針に従って、コマンドの必須権限を判別します。

1. **PV_ROOT** 特権を呼び出し側シェルに割り当てるか、または *aix* 権限を持つロールを引き受ける。

   重要**:** グローバル WPAR で、**PV_ROOT** 特権を呼び出し側シェル・プロセスの有効な最大特権セットに割り当てる必要があります。システム WPAR 内では、この特権もプロセスの継承特権セットに追加する必要があります。
2. コマンドを実行する。

3. プロセスで使用する権限を記録する。
4. *Access Auths* の下にレポートされた権限を、特権コマンド・データベースにあるこのコマンドの **accessauths** 属性に保管する。 *Check Auths* の下にレポートされた権限は、システムでロールを作成時に使用可能です。

PV_ROOT 特権がシェルに割り当てられるか、または *aix* 権限を持つロールを引き受けるため、また、この 2 つの方法は非常に強力であるため、上記の手順は制御された環境で実行する必要があります。 さらに、このコマンドの実行はシステムに何らかの影響を与え、それが他のユーザーに影響する可能性もあります。 実際のところ、この手順は手探り法による手順になる可能性があります。 権限の全セットを取得するために、コマンドをさまざまなフラグとオプションを指定して繰り返し実行したり、長時間にわたるアプリケーションのために長時間実行する必要が生じる場合もあります。 プロセスの必須権限セットは、適切な権限をもつ管理者が実行できる次の手順のいずれかを使用して容易に集めることができます。

traceauth
実行するコマンドの引数を指定します。 **traceauth** コマンドは、その実行するコマンドを実行し、プロセスのライフタイムの間使用される両方のタイプの権限を記録します。 その実行するコマンドが終了すると、**traceauth** コマンドにより、**stdout** で使用された権限が表示されます。

lssecattr
コマンドが長時間プロセスである場合には、**lssecattr** コマンドを使用して、プロセスに使用された権限を表示することができます。 システムで権限のトレースをできるようにするには、次のコマンドを実行します。

setrunmode –c; setsecconf –o traceauth=enableプロセスに使用された権限を表示するには、次のように **lssecattr** コマンドを実行して、モニターされているプロセスの PID を置換します。

lssecattr –p –A *PID*

必須権限を確定したら、 108 ページの『特権コマンド・データベースへのコマンドの追加』の手順を実行してコマンドを特権コマンド・データベースに追加してください。 権限があるユーザーがコマンドを実行して、適切に実行されるかどうか検査しなければなりません。

コマンドに必要な特権の判別*:*

多くのアプリケーションでは、コマンドを正しく実行するために特定の特権が必要となります。 定義済みコマンドのセットが特権コマンド・データベースに提供され、システム管理者はそれぞれのアプリケーションまたは環境に固有のエントリーを追加する必要が生じる場合があります。特権コマンド・データベースを使用すると、コマンドおよびそのコマンドに関連付けられている特権に対応して、エントリーを追加することができます。

コマンドを特権コマンド・データベースに追加する前に、コマンドの実行が可能な限り安全であることを確実にするために、必須特権の最小セットを判別しなければなりません。 適切な実行に必要とされる以上に特権を許可することは、最小特権の原則に違反します。 そのため、システムに特権コマンドを追加する際に、最小必須特権を判別することは重要なステップです。

次の基本的な方針に従って、コマンドの最小必須特権を判別します。

1. 情報システムのセキュリティー担当者 (ISSO) または ISSO ロールを持つユーザーは、特権データベースに割り当てるコマンドを実行して、システム管理者に **PV_ROOT** 特権を割り当てることができます。 **PV_ROOT** 特権の呼び出し側シェルへの割り当ては、setsecattr コマンドを使用して行います。 次に例を示します。

   setsecattr -p eprivs=PV_ROOT mprivs=PV_ROOT $$

2. コマンドを実行して特権セットを収集します。
3. プロセスで使用する特権セットを記録します。
4. コマンドの **innateprivs** 属性の中の必要な特権を特権コマンド・データベースに保管します。

上記の手順は、**PV_ROOT** 特権がシェルに割り当てられて **PV_ROOT** 特権が非常に強力になるため、制御された環境で実行する必要があります。 さらに、コマンドの実行は他のユーザーに影響するおそれがある何らかのシステム影響を及ぼす可能性があります。 実際のところ、この手順は手探り法による手順になる可能性があります。 特権の全セットを獲得するために、コマンドをさまざまなフラグとオプションを指定して繰り返し実行したり、長時間にわたるアプリケーションのために長時間実行する必要が生じる場合もあります。 プロセスの必須特権セットは、適切な権限をもつ管理者が実行できる次の手順のいずれかを使用して容易に集めることができます。

tracepriv
: 実行するコマンドの引数を使用します。 **tracepriv** コマンドは、コマンドを実行し、プロセスのライフタイムの間使用される特権を記録します。 コマンドが終了すると、**tracepriv** コマンドにより、**stdout** で使用された特権が表示されます。

lssecattr
: コマンドが長時間プロセスである場合には、**lssecattr** コマンドを使用して、プロセスに使用された特権セットを表示することができます。 プロセスに使用された特権セットを表示するには、次のコマンドを実行して、モニターされているプロセスの PID を置換します。

  lssecattr –p –a uprivs *PID*

最小必須特権が確認された後で、 108 ページの『特権コマンド・データベースへのコマンドの追加』の手順を実行してコマンドを特権コマンド・データベースに追加してください。 権限があるユーザーがコマンドを実行して、適切に実行されるかどうか検査しなければなりません。

特権のエスカレーション*:*

新規プロセスが **fork** システム・コールによって作成される場合、**fork** は親プロセス (**fork** システム・コールを呼び出したプロセス) と同じ特権をプロセスに認可します。 プロセスが実行可能ファイルで **exec** システム・コールを行うときに、 **exec** は **exec** が現在所有する特権および実行可能ファイルによって所有されている特権に基づいて、実行可能ファイルの特権を再計算します。

エスカレートされた特権は、以下のように計算されます。

1. 初めに、旧 (親) プロセスが所有する継承可能特権と実行可能ファイルが所有する固有の特権セットの union (ビット位置 OR 演算) が計算されます。
2. ユーザーに適切な権限がある場合は、前のステップの結果と権限がある特権の union (ビット位置 OR) が計算されます。
3. 制限付き特権が存在する場合は、前のステップの結果と制限付き特権の論理積が計算されます。 制限付き特権がある場合は、その特権は **exec** システム・コール全体に継承されます。
4. その union の結果の特権セットが新規プロセス用の最大特権セットになります。
5. 継承された特権が実行可能ファイルに存在する場合は、その特権は新規プロセスの継承可能特権セットに割り当てられます。 存在しない場合は、旧 (親) プロセスによって所有されている継承可能特権セットが、新規プロセスの継承可能特権セットに持ち越されます。

実行可能ファイルに、このファイルの **FSF_EPS** ファイル・セキュリティー・フラグ・セットがある場合は、新規プロセス用の有効な特権のセットはその最大特権セットと同じです。 そうでなければ、新規プロセス用の有効な特権のセットは、旧 (親) プロセスによって所有されている継承可能特権セットと同じです。

特権コマンド・データベースへのコマンドの追加:

特権コマンド・データベースにコマンドを追加する前に、適切な許可および特権が割り当てられているかどうかを注意深く検討しなければなりません。

コマンドに有効な属性の詳細については、『/etc/security/privcmds』を参照してください。下記の質問は、コマンドに必要なエントリーを判別する参考にご使用ください。

1. コマンドを実行するために許可制御アクセスが必要ですか。

   YES 許可が存在しない場合には、**mkauth** コマンドを使用して作成してください。 許可を **accessauths** 属性に指定してください。

   NO すべてのユーザーにコマンドの実行を許可する必要がある場合は、**ALLOW_ALL** 許可を **accessauths** 属性に指定してください。

2. コマンドの所有者またはグループが適切な許可をもっていない場合でも、コマンドの実行を許可する必要がありますか。

   YES **ALLOW_OWNER** または **ALLOW_GROUP** 許可を **accessauths** 属性の許可のリストに追加してください。

3. コマンドの実行時に特権の明示セットが必要ですか。

   YES **tracepriv** コマンドをもつ root ユーザーとしてさまざまなオプションを用いてコマンドを実行し、**innateprivs** 属性に必要な特権を判別してください。

4. 特定の許可をもつユーザーに追加の特権を許可する必要がありますか。

   YES **authprivs** 属性に追加の「許可 - 特権」ペアを指定してください。

5. コマンドが SUID または SGID プログラムのように機能する必要がありますか。

   YES 必要に応じて EUID または EGID を指定してください。

6. コマンドに割り当てられた特権を子プロセスに渡す必要がありますか?

   YES **inheritprivs** 属性に特権を指定してください。

7. コマンドの呼び出し時にコマンドの有効な特権セットが最大特権セットと同じである必要がありますか。

   YES secflags 属性に **FSF_EPS** フラグを指定してください。

   NO **secflags** 属性を指定しないでください。 コマンド・コードが取り上げられ、**FSF_EPS** フラグが指定されないときに必要な、より下位の特権が使用されます。

8. 対象のコマンドは特別な実ユーザー ID 0 で実行する必要がありますか。

   YES RUID 属性を指定してください。

9. 実行対象のコマンドは極めて重要であり、規制する必要があり、そのコマンドを実行可能にする前に複数の担当者の存在が必要となりますか。

   YES **authroles** 属性を指定し、ロール・リストを使用して値を割り当ててください。 コマンドを実行するには、各ロールのユーザーが認証されている必要があります。

上記の質問に答えた後、適切なパラメーターを指定した **setsecattr** コマンドを実行して、コマンドをデータベースに追加してください。このコマンドが既存のコマンドであり、SUID または SGID コマンドである場合は、ファイルから **SUID** および **SGID** ビットを除去して最小特権モデルが実行されるように考慮する必要があります。

特権デバイス・データベース:

特権デバイス・データベースには、デバイスからの読み取りまたはデバイスへの書き込みを許可する特権のリストが保管されます。 このデータベースは、管理者が従来のアクセス制御によって管理できること以上にアクセスを制御するためのメカニズムを提供します。

このデータベースはローカル側に保管されると、 `/etc/security/privdevs` ファイルに収容されます。このデータベースには、以下の属性の読み取りまたは書き込み操作のために指定されたデバイスのアクセスで必要な特権が保管されます。

readprivs
デバイスからの読み取りを許可する特権をリストする

writeprivs
デバイスへの書き込みを許可する特権をリストする

特権デバイスが読み取りモードでオープンを要求されると、オープンは **readprivs** 属性で指定された特権のうちの 1 つが、プロセスの有効特権セット (EPS) に存在している場合のみ許可されます。 同様に、デバイスが書き込みモード用に開かれる場合は、**writeprivs** 属性で指定された特権が EPS に存在していなければなりません。

デバイスを特権デバイス・データベースへ追加するプロセスは、通常は一般的な操作ではありません。 **lssecattr** および **setsecattr** コマンドはデータベースを取り扱うために使用できますが、そのデータベースにエントリーを追加またはデータベースのエントリーの変更を行う場合は、かなりの調査が必要になります。 デバイスの読み取りおよび書き込み許可は特権によって制御されるため、デバイスをアクセスするために必要なコマンドおよびアプリケーションについては徹底的な調査を行う必要があり、適切な特権が指定されていることを確認しなければなりません。

特権ファイル・データベース:

従来の UNIX システムの多くのシステム構成ファイルは、root ユーザーによって所有され、他のユーザーが直接に変更することはできません。 RBAC を使用して、ロールを活動化してファイルの変更に必要な特権を獲得するコマンドを実行することにより、これらのシステム構成ファイルを変更することができます。

ファイルの変更を行うことができるコマンド・インターフェースをもたない AIX 構成ファイルもいくつかあります。このような場合には、適切な許可を持つ管理者に、別の方法ではアクセスできないファイルの編集および保存を直接行うことを許可するツールが必要になります。

特権ファイル・データベースでは、システム構成ファイルへのアクセスを決定する権限を使用する方式が提供されています。 このデータベースは、ローカルに保管されている場合は、`/etc/security/privfiles` ファイル内に入っています。 このデータベースにより、構成ファイルはこれらのファイルを表示または変更するために必要な許可にマップされます。 構成ファイルへのアクセスは、このデータベースで次の属性を指定して制御されます。

readauths
ファイルからの読み取りを行うことのできる許可のリスト

writeauths
ファイルへの書き込みを行うことのできる許可のリスト (読み取り許可はこの場合に暗黙指定されます)

特権ファイル・データベースの中のエントリーは、**lssecattr** コマンドによってリストすることができ、**setsecattr** コマンドによって作成または変更することができます。 特権ファイル・データベースで定義されているファイルは、 **/usr/bin/pvi** コマンドを使用して権限があるユーザーがアクセスすることができます。 **pvi** コマンドは、**/usr/bin/tvi** コマンドに基づく **vi** エディターの制限付き特権バージョンです。 **pvi** コマンドにより **tvi** コマンドと同じすべてのセキュリティー上の予防措置 (例えば、**–r** または **-t** フラグなし、シェル・エスケープなし、ユーザー定義マクロなし) が実施され、次の制約事項も加えられます。

- システムは拡張 RBAC モードでなければならない。
- 特権ファイル・データベースで定義されているファイルのみをオープンすることができる。
- 一度に 1 つのファイルのみオープンすることができる。
- 異なるファイル名への書き込みを行うと、コマンド・ラインに指定されているファイル名が使用不可になる。
- `/etc/security/privfiles` ファイルは、**pvi** コマンドを使用して編集することはできません。
- リンクをオープンする試みは失敗します。 通常ファイルのみ編集することができます。

許可検査がファイルをオープンする前に実行されます。 許可が一致した場合には、プロセスの特権セットに **PV_DAC_R** または **PV_DAC_W** (ファイルのオープンが読み取りまたは書き込みのいずれのためかによって決まります) が入れられるようになります。 許可が一致しない場合には、エラー・メッセージが表示され、ユーザーは **pvi** コマンドを使用したファイルへのアクセスを拒否されます。

カーネルのセキュリティー・テーブル:

権限に含まれる情報には、ロール、特権コマンドがあり、特権デバイス・データベースは、データがカーネルのセキュリティー・テーブル (KST) として指定されたカーネルの領域にロードされるまで、セキュリティーの考慮事項の対象としては使用されません。 拡張 RBAC モードでは、権限および特権検査がカーネルで行われます。したがって、カーネルでデータベースを使用するために、データベースをカーネルに送信する必要があります。

KST は以下のサブテーブルで構成されています。

- Kernel Authorization Table (KAT)
- Kernel Role Table (KRT)
- Kernel Command Table (KCT)
- Kernel Device Table (KDT)

すべてのテーブルまたは選択したテーブルは **setkst** コマンドを使用して、ユーザー・スペースからカーネルへ送信できます。 KRT および KCT は KAT に従属するもので、KAT が更新のために選択されると、KRT および KCT もテーブルを同期させるために更新して検査されます。 KST へ更新を追加するために好んで使われる方式は、ユーザー・レベルで必要なデータベースのすべてを作成または変更することです (**mkauth**、 **chauth**、 **mkrole**、および **setsecattr** などのコマンドを使用します)。そして、**setkst** コマンドを使用してテーブルをカーネルへ送信します。 いったん、テーブルがカーネルにロードされると、 **lskst** コマンドを使用して各テーブルの内容を表示できます。

KST に収められたテーブルは完全なテーブルとして、常に送信されます。 言い換えれば、KST に関しては、個別エントリーの変更は許可されなくて、テーブル全体を置き換える必要があります。 **setkst** コマン

ドは、テーブルをカーネルへ送信する前にテーブル自体とテーブル間の関係を検証します。 また、ブート・プロセスの初期の段階でデータベースを KST へ確実に送信するために、 **setkst** コマンドは `inittab` ファイルにも配置されます。

ある理由のためにテーブルが作成されないか、またはカーネルへロードできなくて、テーブルが事前にロードされなかった場合、システムは権限またはロールがないかのように作動します。 このシナリオでは不一致が検出されると、コマンド、API、および権限とロール検査のためのシステム・コールは「障害」を返します。 この状態でのシステム操作は、権限を実施するコマンドでユーザーがコード・セクションをアクセスできないこと以外は、レガシー RBAC モードと非常に類似しています。

root ユーザーを使用不可にする**:**

拡張 RBAC モードでは、システムを構成して、root ユーザーが関連する特殊な権限をもたないようにし、システムによって通常のユーザーとして扱われるようにすることができます。

従来、root ユーザーの ID の値が 0 の場合はオペレーティング・システムによって特権 ID として扱われ、強制的なセキュリティー検査を迂回することができました。 root ユーザーを使用不可にすることで、ユーザー ID が 0 の場合にセキュリティー検査を迂回できるようにするオペレーティング・システム内の検査が効果的に除去され、代わりにセキュリティー検査に合格する特権を得るプロセスが必要となります。 root ユーザーを使用不可にすることで、システム上に単一の全権限をもつユーザー ID がなくなるので、アタッカーによる損害が最小限にされます。 root ユーザーを使用不可にした後、システム管理は、特権ロールが割り当てられているユーザーによって実行されなければなりません。

ルート権限は、**/usr/sbin/setsecconf** コマンドを用いて使用不可にすることができます。 root ユーザーの権限を使用不可にするためには、次のコマンドを実行してからシステムをリブートしてください。

```
setsecconf –o root=disable
```

このコマンドを実行した後では、root ユーザー・アカウントは、リモートまたはローカルのログインでも su コマンドでもアクセスすることができません。 ただし、root ユーザー・アカウントはファイルシステム上のファイルの所有者のままであるため、アカウントが獲得される場合には、ユーザーは特権ファイルにアクセスすることができます。

ルートが使用不可にされたシステムでは、ルートによって所有されるプロセスには、もはや特殊な権限も特権もありません。 特権コマンド・データベースに追加されていないルートによって所有される setuid アプリケーションがシステムにある場合には、このことを考慮する必要があります。 これらの setuid アプリケーションは、プロセスが特権操作を実行できないので、ルート使用不可の環境では失敗すると考えられます。 ルート使用不可のシステムでは、特権操作を実行する必要のあるコマンドはすべて、特権コマンド・データベースに追加されて、適切な特権を割り当てられる必要があります。 そのため、システムとシステム上で使用されるアプリケーションについては、root ユーザーの権限を使用不可にする前に注意深く分析しなければなりません。

リモート **RBAC** データベース・サポート**:**

エンタープライズ環境では、すべてのシステムにわたって共通セキュリティー・ポリシーをこの環境に実装して、システムを強化できることが望まれます。ポリシーを制御するデータベースがそれぞれのシステムに個別に格納されていると、指定されたシステム管理者にはセキュリティー・ポリシーの管理が負担になります。 AIX の拡張 RBAC モードを使用することにより、 RBAC データベースを LDAP に保管することが可能になり、環境内のすべてのシステムのセキュリティー・ポリシーを集中して管理できます。

すべての RBAC 関連データベースを LDAP に保管するサポートが AIX に追加されました。 以下のデータベースは該当する RBAC データベースです。

- 権限データベース
- ロール・データベース
- 特権コマンド・データベース
- 特権デバイス・データベース
- 特権ファイル・データベース

注**:** LDAP に保管された権限データベースには、ユーザー定義権限のみ収めることができます。 システム定義の権限を LDAP に保管することはできません。したがって、それぞれのクライアント・システムのローカル側に残されます。

AIX はユーティリティーを提供していますので、ローカル RBAC データを LDAP にエクスポートすること、LDAP で RBAC データを使用してクライアントを構成すること、 RBAC データのルックアップを制御すること、およびクライアント・システムからの LDAP データを管理することが簡単に行うことができます。 以下のセクションで、拡張 RBAC により提供される LDAP フィーチャーに関する情報を提供します。

RBAC データの ***LDAP*** へのエクスポート**:**

RBAC データベース・リポジトリーとして LDAP を使用するための初期準備には、 LDAP サーバーに RBAC データを追加する必要があります。

LDAP クライアントが LDAP サーバーで RBAC データを使用できるようにするために、 LDAP サーバーには、インストールされている LDAP に対して RBAC スキーマが必要です。 LDAP の RBAC スキーマは、AIX システムに関して `/etc/security/ldap/sec.ldif` ファイルから選択可能です。 LDAP サーバーのスキーマは、**ldapmodify** コマンドを使用して、このファイルを更新する必要があります。

ローカル RBAC データベースのデータを読み取り、それを LDAP のフォーマットで出力するには、**/usr/sbin/rbactoldif** ファイルを使用します。 **rbactoldif** コマンドを使用して生成された出力結果をファイルに保存し、このデータを使用して **ldapadd** コマンドで LDAP サーバーに追加します。 LDAP 用の RBAC を生成するには、ローカル・システムで **rbactoldif** コマンドで以下のデータベースを使用します。

- `/etc/security/authorizations`
- `/etc/security/privcmds`
- `/etc/security/privdevs`
- `/etc/security/privfiles`
- `/etc/security/roles`

RBAC データを LDAP に保管するための場所については、いくつかの考慮事項があります。 LDAP 内の RBAC データはユーザーおよびグループ・データと同じ親 DN の下に置くことをお勧めします。 次に、必要に応じて、選択されたセキュリティー・ポリシーに合わせて、そのデータの ACL を調整する必要があります。

RBAC 用の ***LDAP*** クライアント構成**:**

LDAP に保管された RBAC データを使用するには、LDAP クライアントとしてシステムを構成する必要があります。

システムを LDAP クライアントとして構成する場合は、 AIX **/usr/sbin/mksecldap** コマンドを使用します。 **mksecldap** コマンドは指定された LDAP サーバーを動的に検索して、権限、ロール、特権コマンド、デバイス、およびファイル・データの場所を判別し、結果を /etc/security/ldap/ldap.cfg ファイルに保存します。

mksecldap コマンドを使用してシステムを LDAP クライアントとして正常に構成した後に、さらに、RBAC データのルックアップ・ドメインとして LDAP を使用可能にするために、システムを構成する必要があります。 /etc/nscontrol.conf ファイルを変更して、LDAP に保管されるデータベースの **secorder** 属性に LDAP を組み込まなければなりません。

いったん、システムが LDAP クライアントおよび RBAC データのルックアップ・ドメインの両方として構成されると、 **/usr/sbin/secldapclntd** クライアント・デーモンは LDAP から RBAC データを定期的に検索して、 **setkst** コマンドでカーネルのセキュリティー・テーブル (KST) へデータを送信します。 LDAP からの RBAC データを /etc/security/ldap/ldap.cfg ファイルの **rbacinterval** 属性により検索するために、デーモンによって使用される時間枠をユーザーが構成します。この属性のデフォルト値は 3600 であり、これは 1 時間ごとに 1 回の LDAP からの RBAC データの検索および KST の更新を指定するものです。 KST は管理者が **setkst** コマンドを実行するときに手動で更新することもできます。

ネーム・サービス制御ファイル*:*

RBAC データは、厳密にローカル・ファイルにのみ存在するか、厳密に LDAP にのみ存在するか、または /etc/nscontrol.conf ネーム・サービス制御ファイルの中の指定されたデータベースを構成することによってローカル・ファイルと LDAP にマージすることができます。

権限、ロール、特権コマンド、デバイス、およびファイル・データベースの検索順序は、/etc/nscontrol.conf ファイルで個々に指定されます。データベースの検索順序は、コンマで区切られたドメイン・リストである **secorder** 属性とともにファイルで指定されます。 以下に、権限データベースの構成例を示します。

```
authorizations:
        secorder = LDAP,files
```

この例では、権限が LDAP 中で検出されなかった場合は、権限に基づく照会ではまず LDAP を検索し次にローカル・ファイルを検索することを示しています。 システムで使用可能な権限の収集は、LDAP によって提供される権限とローカル・ファイルで提供される権限の組み合わせです。この組み合わせは単なる 2 つのドメインからの値の組み合わせではなく、値の共用体です。 上記の構成では、すべての LDAP 権限が含まれていて、ローカル・ファイルからの固有の権限のみが結果に追加されます。

変更および削除は、リストされた最初のドメインで試みられ、エンティティーが最初のドメインで見つからない場合、後続のドメインでのみ試みられます。 この場合は、LDAP が最初に試みられ、LDAP で権限が見つからない場合にのみローカル・ファイルが試みられます。 新しいエントリーは常に、**secorder** 属性にリストされている最初のドメインで作成されます。 上記の例では、新しい権限の作成は、LDAP データベースで行われます。

/etc/nscontrol.conf ファイルの中のデータベースにエントリーがない場合またはファイルが存在しない場合には、データベースの照会および変更が、ローカル・ファイル・データベースでのみ実行されます。 このファイルの中のデータベースの構成は、**chsec** コマンドによって設定することができ、**lssec** コマンドを使用してリストすることができます。 権限データが最初に LDAP から、次にローカル・ファイルから取り出されるように構成するには、次のコマンドを実行します。

```
chsec –f /etc/nscontrol.conf –s authorizations –a secorder=LDAP,files
```

/etc/nscontrol.conf ファイルの中の構成は、ライブラリーとコマンド・ライン・インターフェースの両方を制御します。 アプリケーションは、**getsecorder** インターフェースを使用してデータベースの **secorder** 属性の現在値を検索することができます。 **secorder** 属性の値は、**setsecorder** インターフェースによってプロセスに合わせて指定変更することができます。

LDAP に対する ***RBAC*** コマンドの使用可能化*:*

/etc/nscontrol.conf ファイルの構成を使用して、提供されたデータベースに定義されたドメイン (複数可) 内のエンティティーの照会、変更、作成、または除去を行うには、 RBAC データベースの管理コマンドを使用可能にします。

デフォルトで、ドメインはデータベースの **secorder** 属性の定義に従って処理されますが、コマンド・ラインで **–R** オプションを使用することにより、このデフォルトを指定変更できます。 コマンドに **–R** オプションを指定すると、指定されたドメインで操作が強制実行され、/etc/nscontrol.conf ファイル内の構成が指定変更されます。 リモート・ドメイン・サポートとして、以下の RBAC データベース管理コマンドを使用可能にします。

- **mkauth**、**chauth**、**lsauth**、および **rmauth**
- **mkrole**, **chrole**、**lsrole**、および **rmrole**
- **setsecattr**、**lssecattr**、および **rmsecattr**

さらに、/etc/nscontrol.conf ファイルに含まれる構成を使用にするには、 **setkst** コマンドを使用可能にします。 **setkst** コマンドは、このファイルに定義されている指定のデータベースのエントリーについて、マージされたコピーを取り出し、その結果データをカーネルのセキュリティー・テーブルにロードします。

クロスドメイン割り当て*:*

RBAC データが 2 つのドメイン (ローカル・ファイルおよび LDAP) に分割される環境を設計する場合は、エンティティーのクロスドメイン割り当ての問題について考慮する必要があります。 クロスドメイン割り当ての例として、LDAP 定義ロールのローカル・ユーザーへの割り当て、またはローカル定義ロールの LDAP ユーザーへの割り当てがあります。

リモート・エンティティー (LDAP ロール) のローカル・エンティティー (ローカル・ユーザー) への割り当ては、この環境では他のシステムに影響がないため、大した関心事ではありません。 しかし、ローカル・エンティティー (ローカル・ロール) をリモート・エンティティー (LDAP ユーザー) に割り当てる場合は、注意深く行う必要があります。 リモート・エンティティー (LDAP ユーザー) は複数のクライアントで可視の状態ですが、リモート・エンティティーに割り当てられるローカル・エンティティー (ローカル・ロール) が定義されていること、または各クライアント・システムに同じ定義があることは保証されていません。 例えば、ロールは各クライアントにローカル側で定義されますが、関連付けられた権限が異なっている場合があります。 したがって、このローカル・ロールを割り当てられるリモート・ユーザーは、これらのクライアントのそれぞれに異なる権限を持つことになり、このためにセキュリティーで望ましくない結果になることがあります。

ローカル・エンティティーを LDAP エンティティーに割り当てることにより、可能性のあるセキュリティー問題の発生を防ぐには、それぞれのクライアントによるエントリーの変更を防ぐ RBAC データベースへのアクセス制御を LDAP サーバーに実装することをお勧めします。 特権アカウントにより LDAP サーバーへ接続するクライアントのみ、LDAP RBAC エンティティーの変更を許可するようにしてください。 その他のクライアントについては、LDAP RBAC データベースには読み取りアクセスのみ可能にします。

拡張 **RBAC** でのサイズ制**:**

下表は RBAC 関連エレメントに対する各種制限のリストです。

表 *10. RBAC* 関連エレメントに対する各種制限

説明	最大サイズ
ロール名	63 印刷可能文字
セッション当たりの最大ロール数	8
最大権限名サイズ	63 印刷可能文字
権限階層の最大レベル数	9
コマンド当たりの最大アクセス権限数	8
コマンド当たりの最大許可特権数	8

拡張 **RBAC** の管理**:**

このセクションでは、RBAC の管理のためのコマンド・ライン使用による共通のシナリオについて説明します。 ここで、機能の主要な観点について例示します。 RBAC 管理のために SMIT インターフェースも提供されます。 RBAC SMIT メニューへの高速パスは `smit rbac` です。

ユーザー定義の権限の作成*:*

コマンドの実行を制御することができるユーザー定義の権限を作成できます。

ユーザー定義の権限を作成する場合は、mkauth コマンドを使用します。権限データベースへの変更は **setkst** コマンドを使用して、その変更をカーネルへダウンロードした後に有効になります。

- ユーザー定義の権限を作成しは、以下のコマンドを実行します。

  mkauth *auth_name*

ロールの作成作成変更*:*

ロールの作成には **mkrole** コマンドを使用します。

ロールの作成には **mkrole** コマンドを使用します。 ロール・データベースへの変更は、**setkst** コマンドを使用して、その変更をカーネルへダウンロードした後に有効になります。 ロールの変更には **chrole** コマンドを使用します。

- ロールを作成するには、以下のコマンドを実行します。

  mkrole dflt_msg="*My Role***"** *role_name*
- ロールを作成して既存のロールから権限を継承するには、以下のコマンドを実行します。

  mkrole rolelist=*child_role1***,***child_role2 role_name*
- ロール定義を変更するには、以下のコマンドを実行します。

  chrole rolelist=*child_role3 role_name*

権限のロールへの割り当て*:*

権限をロールへ割り当てる場合は、**mkrole** コマンドまたは **chrole** コマンドを使用します。

- **auth_name1** および **auth_name2** 権限を **role_name** ロールに割り当てるには、 **mkrole** コマンドを実行します。

```
mkrole authorizations=auth_name1,auth_name2 role_name
```

- **auth_name1** および **auth_name2** 権限を **role_name** に割り当てるには、 **chrole** コマンドを実行します。

  ```
  chrole authorizations=auth_name1,auth_name2 role_name
  ```

ロールに対する認証モードの設定*:*

ロールの活動化については、ロールの **auth_mode** 属性により制御できます。

auth_mode 属性に有効な値は、以下のとおりです。

NONE
認証は不要

INVOKER
呼び出し側はパスワードが必要。 これはデフォルトです。

ユーザーは指定されたロールを担うときに、自分のものとして認証を強制実行するには、以下のコマンドを入力します。

chrole auth_mod=INVOKER *role_name*

ユーザーへのロールの割り当て*:*

ロールをユーザーに割り当てる場合は、**chuser** コマンドを使用します。

role_name1 ロールおよび **role_name2** ロールをユーザー **user_name** に割り当てるには、以下のコマンドを実行します。

```
chuser roles=role_name1,role_name2 user_name
```

ロールの活動化*:*

デフォルトで特権コマンドを実行するために、ユーザーはセッションのロールを活動化する必要があります。

- **role_name1** ロールおよび **role_name2** ロールを活動化するには、以下のコマンドを実行します。

  ```
  swrole role_name1,role_name2
  ```

- ユーザーに割り当てられる一部のロールは、デフォルトのロールとして分類されます。 これらのロールはユーザーがログインするときに自動的に活動化されます。 そして、これらのロールはログイン・セッションが続いている間、常に、活動状態になっています。 ユーザーのデフォルトのロールとして role_name1 を割り当てる場合は、以下のコマンドを実行します。

  ```
  chuser roles=role_name1,role_name2 default_roles=role_name1 user_name
  ```

活動状態ロール・セットのリスト作成*:*

セッションの有効な活動状態ロール・セットに関する情報を表示情報する場合は、 **-e** オプションを指定した **rolelist** コマンドを使用します。

- セッションの有効な活動状態ロール・セットを表示するには、以下のコマンドを実行します。

  ```
  rolelist -e
  ```

ユーザー・ロールのリスト作成*:*

rolelist コマンドは、ユーザーの現行のロールまたは割り当て済みのロールについて、ロール情報と権限情報を提供します。

デフォルトで、**rolelist** コマンドはユーザーに割り当てられたロール・リストを表示します。 これは、基本的には `lsuser -a roles user1` コマンドによって表示される情報と同じですが、ロールのテキスト記述が用意されていれば、その記述も含まれます。

- 割り当てられたロールおよび関連する権限をリストするには、以下のコマンドを実行します。

  ```
  rolelist -a
  ```

セッション・ロールの監査*:*

ログイン・セッションで活動状態になっているロールは、UID および GID などのその他の属性と一緒に監査されます。 これらのロールは **auditpr** コマンドでリストすることができます。

監査証跡からロールを表示するには、以下のコマンドを実行します。

```
auditpr -h eli -i /audit/trail
```

実行中プロセスへの特権の割り当て*:*

実行中プロセスの特権を変更する場合は、**setsecattr** コマンドを使用します。

- プロセスに関連付けられた有効特権セットを更新するには、以下のコマンドを実行します。

  setsecattr –p eprivs=*privileges pid*
- 特権をプロセスの有効特権セットに追加する前に、この特権が最大特権セットに既に存在していることを確認する必要があります。 最大特権セットを変更するには、以下のコマンドを実行します。

  setsecattr –p mprivs=*privileges pid*

WPAR 特権の管理*:*

それぞれの WPAR は、その能力を決定する特権セットと関連付けられます。 これは WPAR 特権セット (WPS) と呼ばれます。

指定された WPAR 内で実行するプロセスは WPS で使用可能なプロセスの特権のみ使用できます。

- グローバル WPAR から WPS を変更するには、以下のコマンドを実行します。

  chwpar –S privs+=*privileges wpar_name*

コマンドに必要な特権の判別*:*

一部のコマンドには特権操作を実行するための特殊な特権が必要です。 特権はセキュリティーに関する制限をバイパスするためにカーネルで使用されます。

コマンドが正常に実行するために必要な特権を判別するコマンドのプロファイルを作成する場合は、**tracepriv** コマンドを使用します。 **tracepriv** コマンドは、実行するときに別のコマンドによって使用される特権を記録します。 このコマンドは特権を使用する試みが成功するように **PV_ROOT** 特権で実行する必要があります。 コマンドが完了すると、使用された特権セットは `stdout` に送信されます。

- 提供されたコマンドでプロファイルを作成するには、以下のコマンドを実行します。

  tracepriv –ef *command_name*

コマンドを制御する権限の使用*:*

コマンドの実行を制御する場合に、いくつかの権限を使用できます。

setsecattr コマンドは権限をコマンドに関連付ける場合に使用します。 **setsecattr** コマンドはスタンザを特権コマンド・データベース (/etc/security/privcmds) へ追加します。このデータベースの変更は **setkst** コマンドを使用してカーネルへダウンロードする必要があります。

- 権限をコマンドに関連付けるには、以下のコマンドを実行します。

  setsecattr –c accessauths=*auth_names* **innateprivs=***privileges* **proxyprivs=***privileges* **authprivs=***auth_name***=***privileges command_name*

デバイスの制御アクセス*:*

RBAC はデバイスの制御アクセスを推進するメカニズムを提供します。システム管理者は読み取りモードまたは書き込みモードでデバイスのオープンを要求する特権を指定できます。

例えば、DVD ライターへの書き込みアクセスは **PV_DEV_CONFIG** 特権により制御され、この特権を持つプロセスのみ DVD を作成できます。

- デバイス・データベースへデバイスを追加するには、以下のコマンドを実行します。

  setsecattr –d readprivs=*privileges* **writeprivs=***privileges device_name*

RBAC カーネルのセキュリティー・テーブルの更新*:*

setkst コマンドはセキュリティー・データベースを読み取り、そのデータベースの情報をカーネルのセキュリティー・テーブル (KST) にロードします。

デフォルトで、セキュリティー・データベースのすべてが KST に送信されます。 また、特定のデータベースを **-t** オプションで指定できます。しかし、権限データベースだけを KST へ送信するように指定しても、ロールおよび特権コマンドは特権データベースに従属しているために、KST のロールおよび特権コマンドも更新されます。

- すべての最新 RBAC データベースをカーネルへ送信するには、以下のコマンドを実行します。

  ```
  setkst
  ```

拡張 ***RBAC*** モード・スイッチの使用*:*

システム全体にわたって構成スイッチは、拡張 RBAC 機能を使用不可にして、レガシー RBAC 動作に戻すことを可能にするために提供されます。

システム管理者は **sys0** デバイスで **chdev** コマンドを実行して、 **enhanced_RBAC** 属性に値 **false** を指定し、システムをリブートすることにより、拡張 RBAC モードを使用不可にすることができます。 モードの切り替えについては、**enhanced_RBAC** 属性を **true** に設定して、システムをリブートすることにより、拡張 RBAC モードに戻すことができます。

- レガシー RBAC モードに戻すには、以下のコマンドを実行します。

  ```
  chdev -l sys0 -a enhanced_RBAC=false
  ```

- **enhanced_RBAC** 属性の値をリストするには、以下のコマンドを実行します。

  ```
  lsattr -E -l sys0 -a enhanced_RBAC
  ```

WPAR 環境では、グローバル・システムからのみ RBAC モードを構成することが可能であり、すべての WPAR と同様にグローバル・システムに対して影響を及ぼします。

注: 拡張 RBAC モードを使用不可にすると、システム、特に WPAR のセキュリティー限界を引き下げることになります。

RBAC 関連コマンド

以下の表では、RBAC フレームワークを管理および使用するために AIX オペレーティング・システムで提供されている RBAC 関連コマンドをリストします。

コマンド	説明
chauth	ユーザー定義の権限属性を変更する
chrole	ロール属性を変更する
ckauth	権限について現在のプロセスを検査する
lsauth	ユーザー定義およびシステム定義の権限属性を表示する
lskst	カーネルのセキュリティー・テーブルのエントリーをリストする
lspriv	システムで使用可能な特権を表示する
lsrole	ロール属性を表示する
lssecattr	コマンド、デバイス、プロセス、またはファイルのセキュリティー属性を表示する
mkauth	新規ユーザー定義の権限を作成する
mkrole	新規ロールを作成する
pvi	特権ファイル・エディター
rbacqry	アプリケーションで RBAC を使用可能にする
rbactoldif	LDAP 互換形式で RBAC ユーザー・レベル・データベースを出力する
rmauth	ユーザー定義の権限を除去する
rmrole	ロールを除去する
rmsecattr	コマンド、デバイス、またはファイルのセキュリティー属性の定義を除去する
rolelist	ユーザーまたはプロセスのロール情報を表示する
setkst	RBAC ユーザー・レベル・データベースにあるエントリーをカーネルのセキュリティー・テーブルへ送信する
setsecattr	コマンド、デバイス、プロセス、またはファイルのセキュリティー属性を設定する
setsecconf	カーネル・セキュリティー・フラグを変更する
swrole	新規ロール・セッションを作成する
tracepriv	正常に実行するためにコマンドで必要になる特権をトレースする

RBAC 関連ファイル

下表はデータベース情報を構成および保管する目的で AIX が提供する RBAC 関連ファイルのリストです。

ファイル	説明
`/etc/nscontrol.conf`	特定のセキュリティー・データベース用のネーム・サービス制御ファイル

ファイル	説明
/etc/security/authorizations	ユーザー定義の権限データベース
/etc/security/privcmds	特権コマンド・データベース
/etc/security/privfiles	特権ファイル・データベース
/etc/security/privdevs	特権デバイス・データベース
/etc/security/roles	ロール・データベース

アプリケーション内での拡張 RBAC の使用

多くのアプリケーションは、拡張 RBAC 環境で何も変更しなくても正常に実行します。 アプリケーションのアクセス権限を定義し、特権を関連付けて、アプリケーションを特権コマンド・データベースに割り当てるだけで十分に目的を果たすことが可能です。

ただし、アプリケーションは RBAC インターフェースを呼び出して拡張 RBAC を使用し、細かいレベルでアプリケーションの実行を制御することができるため、アプリケーションをより安全にすることができます。 拡張 RBAC を組み込んで効果のあるアプリケーションとして、以下のものを挙げることができます。

- root ユーザーまたは特定グループのメンバーのいずれかに使用を制限するアプリケーション。 通常、これらのアプリケーションは実効ユーザーの ID またはグループのメンバーシップを検査しますので、これを権限の検査に代えるように変更できます。
- 特権のないユーザーがコマンド呼び出しで特権を獲得できるように、 **setuid** または **setgid** モード・ビットを使用するアプリケーション。これらのアプリケーションは、通常、特権をまとめて使用することで、より安全になり、より少ない特権を使用してアプリケーションのタスクを完遂できます。

許可検査**:**

特権操作が実行可能かどうかを判別するために呼び出し側ユーザーのユーザー ID またはグループ ID を現在使用するアプリケーションは、代わりに許可を検査するように変更する必要があります。

例えば、ファイルシステム構成タスクを実行し、root ユーザー (UID = 0) がいくつかの特権操作を現在実行できるようにしているアプリケーションについて検討します。

```
if (getuid() == 0) {
   /* allow privileged operation to continue */
}
```

このアプリケーションで代わりに特定の許可 (**aix.fs.config**) をユーザーに与えて特権操作を行うことができるにするためには、**checkauths** API を使用して許可検査を実行するようにコードを変更することができます。

```
if (checkauths("aix.fs.config", CHECK_ALL)) {
  /* allow privileged operation to continue */
 }
```

checkauths API は、レガシー RBAC モードと拡張 RBAC モードの両方で使用可能であり、呼び出し側プロセスに指定された許可がある場合には **0** 成功コードを戻します。 **checkauths** API はまた、root ユーザーの権限が使用可能か使用不可か、ならびに root ユーザーに必要に応じて許可検査を迂回させるかさせないかを判別します。 AIX バージョン 6.1 以前は、**MatchAllAuths**、**MatchAnyAuths**、**MatchAllAuthsList**、および **MatchAnyAuthsList** API が通常、許可検査に使用されました。 AIX バージョン 6.1 以降で提供されるアプリケーションでは、レガシー RBAC モードと拡張 RBAC モードのサポートならびにルートの使用不可設定のために、代わりに **checkauths** API を使用する必要があります。

上記の例のように、**getuid**、**getgid**、または特定のタスクを実行する特定のユーザーにのみ許可される同様の機能を呼び出すアプリケーションは、代わりに **checkauths** API を使用して許可検査を実行するように変更することができます。 検査されるユーザー ID またはグループ ID が root ユーザーのものでない場合には、最初に **sys_parm** システム・コールを使用して、拡張 RBAC が使用可能か否かを照会することができます。 拡張 RBAC が使用可能でない場合には、コードは既に使用可能になっている検査を実行することができます。 拡張 RBAC が使用可能である場合には、コードは関係のあるシステムまたはユーザー定義の許可を検査することができます。

特権のまとめ**:**

アプリケーションは権限の検査のために、いったん変更されると、操作の際に細分化された「特権の囲い込み」を利用するために、さらに変更することができます。

アプリケーションでは **priv_raise** API を使用して操作を行うために必要な特権を引き上げたり、**priv_lower** API を使用して特権を引き下げたりすることができます。特権の引き上げは特権を持つ操作が試みられる直前に行われ、特権の引き下げは操作の完了後に行われることは、「特権の囲い込み (privileged bracketing)」と呼ばれ、アプリケーションが特権を使用する場合に好んで使われる方法です。特権を引き上げる場合、特権は特権コマンド・データベース内のアプリケーションの最大特権セットの状態で使用可能になっている必要があります。 特権を引き上げる場合、特権はプロセスの有効特権セット (EPS) に配置されます。 特権を引き下げると、特権は EPS から除去されます。 以下のコーディング・サンプルでは、特権が **auditproc** API の前後で一括されていることを表します。

```
priv_raise(PV_AU_ADMIN, -1); /* raise privilege when needed */
auditproc();     /* call auditing system call */
priv_lower(PV_AU_ADMIN, -1); /* lower privilege */
```

RBAC 認識アプリケーション**:**

もともとは、AIX およびルート使用可能拡張 RBAC システムでは、特権コマンド・データベースにないルートまたはルート所有の **setuid** プログラム (UID=0 指定) には、常に、カーネルですべての特権を認可されます。 したがって、カーネルでの特権チェックの戻りは、要求された特権がプロセスの有効特権セット (EPS) にない場合でも、必ず「成功」です。

このような動作は既存の **setuid** アプリケーションをサポートするために、現在でも必要ですが、 **setuid** プログラムがルートの能力のすべてを所有しているということで、セキュリティー上のリスクがあります。

ルート使用可能拡張 RBAC システム上のプロセスに適切な「特権の囲い込み (privilege bracketing)」を許可するために、プロセス構造体に新ビットが導入されました。 このビットが設定されると、プロセスは RBAC 認識プロセスになり、有効な UID 0 でも特別な特権は提供されません。 このビットはプログラムで **proc_rbac_op** システム・コールを使用して設定します。 特権コマンド・データベースにリストされていない **setuid** プログラムでは、いずれもこの機能を使用して、使用可能な特権を削減することにより、セキュリティーのぜい弱性を低減させることができます。 特権コマンド・データベースに定義されるプログラムは、自動的に RBAC 認識プロセスとしてマークを付けられ、リストされた特権をデータベースに割り当てるだけであることに注意してください。

以下のコーディングは、アプリケーションがそれ自体に RBAC 認識プロセスとしてマークを付けて、適切な「特権の囲い込み (privilege bracketing)」を行う方法を示す実例です。

```
#include <userpriv.h
#include <sys/priv.h>

privg_t effpriv;

int rbac_flags = SEC_RBACAWARE;
```

```
/* Mark the process as RBAC-aware. */
proc_rbac_op(-1, PROC_RBAC_SET, &rbac_flags);

/* Set the effective privilege set as empty. */
priv_clrall(effpriv);
setppriv(-1, &effpriv, NULL, NULL, NULL);

/* Raise privilege when required. */
priv_raise(PV_AU_ADMIN, -1);
auditproc();

/* Lower privilege when no longer needed. */
priv_lower(PV_AU_ADMIN, -1);
```

RBAC API:

システムで使用可能な RBAC 関連 API を下の表にリストします。 詳細情報については個別の API を参照してください。

API	説明
checkauths	権限リスト内のパス済みの権限を現在のプロセスに関連した権限と比較する
GetUserAuths	現在のプロセスに割り当てられた権限セットを取り出す
MatchAllAuths、MatchAllAuthsList、MatchAnyAuths、MatchAnyAuthsList	権限を比較する。 checkauths API はこれらの API に優先する
getauthattr、putauthattr	権限データベースに定義された権限を照会または変更する
getauthattrs	権限データベースから複数の権限属性を取り出す
putauthattrs	権限データベースの複数の権限属性を更新する
getcmdattr、putcmdattr	特権コマンド・データベースのコマンド・セキュリティー情報を照会または変更する
getcmdattrs	特権コマンド・データベースから複数のコマンド属性を取り出す
putcmdattrs	特権コマンド・データベースの複数のコマンド属性を更新する
getdevattr、putdevattr	特権デバイス・データベースのデバイス・セキュリティー情報を照会または変更する
getdevattrs	特権デバイス・データベースから複数のデバイス属性を取り出す
putdevattrs	特権デバイス・データベースの複数のデバイス属性を更新する
getpfileattr、putpfileattr	特権ファイル・データベースのファイル・セキュリティー情報を照会または変更する
getpfileattrs	特権ファイル・データベースから複数のファイル属性を取り出す
putpfileattrs	特権ファイル・データベースの複数のファイル属性を更新する
getroleattr、putroleattr	ロール・データベースに定義されたロールを照会または変更する
getroleattrs	ロール・データベースから複数のロール属性を取り出す
putroleattrs	ロール・データベースの複数のロール属性を更新する

API	説明
getsecorder	特定のセキュリティー・データベースの場合のドメインの順序付けを取り出す
setsecorder	特定のセキュリティー・データベースの場合のドメインの順序付けを設定する

AIX 特権

AIX で使用可能な特権を下の表にリストします。 各特権およびその関連システム・コールの説明が付けられています。 一部の特権は階層が形成されていて、この場合、1 つの特権で別の特権と関連付けられているすべての権限を認可できます。

特権について検査するときに、システムは最初にプロセスが必要な最も低位の特権を持っているかどうかを判別し、続いて、より強力な特権の存在について、階層を上げて検査を進めます。例えば、**PV_AU_** 特権を持っているプロセスは自動的に **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権も所有し、**PV_ROOT** 特権を持っているプロセスは **PV_SU_** 特権の場合を除いて、以下にリストする特権のすべてを自動的に所有します。

特権	説明	システム・コール参照
PV_ROOT	PV_SU_ (およびこれより上位にある特権) を除いて、以下にリストするすべての特権に相当するプロセスを認可する	
PV_AU_ADD	プロセスが監査レコードを記録/追加することを許可する	auditlog
PV_AU_ADMIN	プロセスが監査システムを構成および照会することを許可する	audit、 auditbin、 auditevents、 auditobj
PV_AU_PROC	プロセスがプロセスの監査状態を取得および設定することを許可する	auditproc
PV_AU_READ	プロセスが Trusted AIX 内の監査ファイルとしてマークを付けられたファイルを読み取ることを許可する	
PV_AU_WRITE	プロセスが Trusted AIX 内の監査ファイルとしてマークを付けられたファイルの書き込みまたは削除を行うこと、あるいは監査ファイルとしてファイルにマークを付けることを許可する	
PV_AU_	上位のすべての監査特権 (PV_AU_*) を結合したものと同等	
PV_AZ_ADMIN	プロセスがカーネルのセキュリティー・テーブルを変更することを許可する	sec_setkst
PV_AZ_READ	プロセスがカーネルのセキュリティー・テーブルを取得することを許可する	sec_getkat、 sec_getkpct、 sec_getkpdt、 sec_getkrt など
PV_AZ_ROOT	exec() (継承の目的のために使用される) での許可検査をプロセスにパスさせる	

特権	説明	システム・コール参照
PV_AZ_CHECK	すべての許可検査をプロセスにパスさせる	sec_checkauth
PV_DAC_R	プロセスが DAC 読み取り制限をオーバーライドすることを許可する	access、 creat、 accessx、 open、 read、 faccessx、 mkdir、 getea、 rename、 statx、 _sched_getparam、 _sched_getscheduler、 statea、 listea
PV_DAC_W	プロセスが DAC 書き込み制限をオーバーライドすることを許可する	上位のほとんど、 および setea、 write、 symlink、 _setpri、 _sched_setparam、 _sched_setscheduler、 fsetea、 rmdir、 removeea
PV_DAC_X	プロセスが DAC 実行制限をオーバーライドすることを許可する	上位のほとんど、および execve、 symlink、rmdir、chdir、fchdir、 ra_execve
PV_DAC_O	プロセスが DAC 所有権制限をオーバーライドすることを許可する	chmod、 utimes、 setacl、 revoke、 mprotect
PV_DAC_UID	プロセスがそのユーザー ID を変更することを許可する	setuid、seteuid、setuidx、setreuid、 ptrace64
PV_DAC_GID	プロセスがそのグループ ID を新規に設定または変更することを許可する	setgid、 setgidx、 setgroups、 ptrace64
PV_DAC_RID	プロセスがそのロール ID を新規に設定または変更することを許可する	setroles、getroles
PV_DAC_	上位のすべての DAC 特権 (PV_DAC_*) を結合したものと同等	
PV_FS_MOUNT	プロセスがファイルシステムをマウントおよびアンマウントすることを許可する	vmount、umount
PV_FS_MKNOD	プロセスが不特定タイプのファイルを作成すること、または mknod システム・コールを実行することを許可する	mknod
PV_FS_CHOWN	プロセスがファイルの所有権を変更することを許可する	chown、 chownx、 fchownx、 lchown
PV_FS_QUOTA	プロセスがディスク割り当て量の関連操作を管理することを許可する	quotactl
PV_FS_LINKDIR	プロセスがディレクトリーへのハード・リンクを作成することを許可する	link、unlink、remove
PV_FS_CNTL	ファイルシステムの拡張および縮小を除いて、プロセスが各種制御操作を実行することを許可する	fscntl
PV_FS_RESIZE	プロセスがファイルシステムでの操作の拡張および縮小を実行することを許可する	fscntl
PV_FS_CHROOT	プロセスがそのルート・ディレクトリーを変更することを許可する	chroot

特権	説明	システム・コール参照
PV_FS_PDMODE	プロセスが分割タイプのディレクトリーを作成または設定することを許可する	pdmkdir
PV_FS_	上位のすべてのファイルシステム特権 (PV_FS_*) を結合したものと同等	
PV_PROC_PRIV	プロセスがプロセスと関連付けられた特権セットを変更または表示することを許可する	setppriv、getppriv
PV_PROC_PRIO	プロセス/スレッドが優先順位、ポリシーおよびその他のスケジューリング・パラメーターを変更することを許可する	_prio_requeue、 _setpri、 _setpriority、 _getpri、 _sched_setparam、 _sched_setscheduler、 _thread_setsched、 thread_boostceiling、 thread_setmystate、 thread_setstate
PV_PROC_CORE	プロセスがコアをダンプすることを許可する	gencore
PV_PROC_RAC	プロセスがユーザー当たりの制限より多いプロセスを作成することを許可する	appsetrlimit、 setrlimit64、 mlock、 mlockall、 munlock、 munlockall、 plock、 upfget、 upfput、 restart、 brk、 sbrk
PV_PROC_RSET	リソース・セット (rset) をプロセスまたはスレッドに接続することを許可する	bindprocessor、 ra_attachrset、 ra_detachrset、 rs_registername、 rs_setnameattr、 rs_discardname、 rs_setpartition、 rs_getassociativity、 kra_mmapv
PV_PROC_ENV	プロセスがユーザー情報をユーザー構造体に設定することを許可する	ue_proc_register、 ue_proc_unregister、 usrinfo
PV_PROC_CKPT	プロセスが別のプロセスのチェックポイントを取るかまたは再始動することを許可する	setcrid、restart
PV_PROC_CRED	プロセスが資格情報属性を設定することを許可する	__pag_setvalue、 __pag_setvalue64、 __pag_genpagvalue
PV_PROC_SIG	プロセスが関連付けられていないプロセスへシグナルを送信することを許可する	_sigqueue、 kill、 signohup、 gencore、 thread_post、 thread_post_many
PV_PROC_TIMER	プロセスが精度の高いタイマーを送信および使用することを許可する	appresabs、 appresinc、 absinterval、 incinterval、 _poll、 _select、 _timer_settime
PV_PROC_RTCLK	プロセスが CPU 時間クロックをアクセスすることを許可する	_clock_getres、 _clock_gettime、 _clock_settime、 _clock_getcpuclockid
PV_PROC_VARS	プロセスがプロセスのチューナブル・パラメーターを取得および更新することを許可する	smttune

特権	説明	システム・コール参照
PV_PROC_PDMODE	プロセスが分割ディレクトリーの REAL モードを変更することを許可する	setppdmode
PV_PROC_	上位のすべてのプロセス特権 (PV_PROC_*) を結合したものと同等	
PV_TCB	プロセスがカーネル・トラステッド・ライブラリー・パスを変更することを許可する	chpriv、fchpriv
PV_TP	プロセスはトラステッド・パス・プロセスであり、トラステッド・パス・プロセスへの制限付きアクションを許可する。 (注: 旧 AIX BYPASS_TPATH 特権と同じ)	
PV_WPAR_CKPT	プロセスが WPAR でチェックポイント・リスタート操作を実行することを許可する	smcr_proc_info、 smcr_exec_info、 smcr_mapinfo、 smcr_net_oper、 smcr_procattr、 aio_suspend_io、 aio_resume_io
PV_KER_ACCT	プロセスがアカウンティング・サブシステムに関する制限付き操作を実行することを許可する	acct、 _acctctl、 projctl
PV_KER_DR	プロセスが動的再構成操作を起動することを許可する	_dr_register、 _dr_notify、 _dr_unregister、 dr_reconfig
PV_KER_TIME	プロセスがシステム・クロックおよびシステム時刻を変更することを許可する	adjtime、 appsettimer、 _clock_settime
PV_KER_RAC	プロセスが共有メモリー・セグメント用にラージ (ページング不可) ページを使用することを許可する	shmctl、vmgetinfo
PV_KER_WLM	プロセスが WLM 構成を初期化および変更することを許可する	_wlm_set、 _wlm_tune、 _wlm_assign
PV_KER_EWLM	プロセスが eWLM 環境を初期化または照会することを許可する	
PV_KER_VARS	プロセスがカーネル・ランタイムのチューナブル・パラメーターをテストまたは設定することを許可する	sys_parm、 getkerninfo、 __pag_setname、 sysconfig、 kunload64
PV_KER_REBOOT	プロセスがシステムをシャットダウンすることを許可する	reboot
PV_KER_RAS	プロセスが RAS レコード、エラー・ロギング、トレース、ダンプ機能について、構成または書き込むことを許可する	mtrace_set、mtrace_ctl
PV_KER_LVM	プロセスが LVM サブシステムを構成することを許可する	
PV_KER_NFS	プロセスが NFS サブシステムを構成することを許可する	

特権	説明	システム・コール参照
PV_KER_VMM	プロセスがカーネル内でスワップ・パラメーターおよびその他の VMM チューナブル・パラメーターを変更することを許可する	swapoff、 _swapon_ext、 vmgetinfo
PV_KER_WPAR	プロセスがワークロード・パーティションを構成することを許可する	brand、 corral_config、 corral_delete、 corral_modify、 wpar_mkdevexport、 wpar_rmdevexport、 wpar_lsdevexport
PV_KER_CONF	プロセスが各種のシステム構成操作を実行することを許可する	sethostname、 sethostid、 unameu、 setdomainname
PV_KER_EXTCONF	プロセスがカーネル・エクステンション内で各種の構成タスク (カーネル・エクステンション・サービス用) を実行することを許可する	
PV_KER_IPC	プロセスが IPC メッセージ・キュー・バッファーの値を大きくすること、および範囲指定された shmget の接続を許可する	msgctl、 shm_open、 shmget、 ra_shmget、 ra_shmgetv、 shmctl
PV_KER_IPC_R	プロセスが IPC メッセージ・キュー、セマフォー・セット、または共有メモリー・セグメントを読み取ることを許可する。	msgctl、 __msgrcv、 _mq_open、 semctl、 shmat、 shm_open、 __semop、 shmctl、 __semtimedop、 sem_post、 _sem_wait、 __msgrcv、 __msgxrcv
PV_KER_IPC_W	プロセスが IPC メッセージ・キュー、セマフォー・セット、または共有メモリー・セグメントに書き込むことを許可する。	_mq_open、 shmat、 _sem_open、 semctl、 shm_open、 shmctl、 mq_unlink、 sem_unlink、 shm_unlink、 msgctl、 __msgsnd
PV_KER_IPC_O	プロセスがすべての IPC オブジェクトで DAC 所有権をオーバーライドすることを許可する	msgctl、semctl、shmctl、fchmod、 fchown
PV_KER_SECCONFIG	プロセスがカーネル・セキュリティー・フラグを設定することを許可する	sec_setsecconf、 sec_setrunmode、 sec_setsyslab、 sec_getsyslab
PV_KER_PATCH	プロセスがカーネル・エクステンションをパッチすることを許可する	
PV_KER_	上位のすべてのカーネル特権 (PV_KER_*) を結合したものと同等	
PV_DEV_CONFIG	プロセスがカーネル・エクステンションおよびシステムのデバイスを構成することを許可する	sysconfig
PV_DEV_LOAD	プロセスがカーネル・エクステンションおよびシステムのデバイスをロードおよびアンロードすることを許可する	sysconfig
PV_DEV_QUERY	プロセスがカーネル・モジュールを照会することを許可する	sysconfig

特権	説明	システム・コール参照
PV_SU_ROOT	プロセスに標準 AIX スーパーユーザーと関連付けられたすべての特権を認可する	
PV_SU_EMUL	UID が 0 の場合、プロセスに標準 AIX スーパーユーザーと関連付けられたすべての特権を認可する	
PV_SU_UID	getuid システム・コールでは 0 を戻すようにする	getuidx
PV_SU_	上位のすべてのスーパーユーザー特権 (PV_SU_*) を結合したものと同等	
PV_NET_CNTL	プロセスがネットワーク・テーブルを変更することを許可する	socket、 bind、 listen、 _naccept、 econnect、 ioctl、 rmsock、 setsockopt
PV_NET_PORT	プロセスが特権付きポートにバインドすることを許可する	bind
PV_NET_RAWSOCK	プロセスがネットワーク層への直接アクセス権限を持つことを許可する	socket、 _send、 _sendto、 sendmsg、 _nsendmsg
PV_NET_CONFIG	プロセスがネットワーク・パラメーターを構成することを許可する	
PV_NET_	上位のすべてのネットワーキング特権 (PV_NET_*) を結合したものと同等	

以下の特定にリストされた特権は Trusted AIX に固有です。

Trusted AIX 特権	説明	システム・コール参照
PV_LAB_CL	プロセスがプロセスの認可の対象になるサブジェクト SCL を変更することを許可する	
PV_LAB_CLTL	プロセスがプロセスの認可の対象になるサブジェクト TCL を変更することを許可する	
PV_LAB_LEF	プロセスがラベル・エンコード・ファイルを読み取ることを許可する	
PV_LAB_SLDG	プロセスがプロセスの認可の対象になる SL をダウングレードすることを許可する	
PV_LAB_SLDG_STR	プロセスがプロセスの認可の対象になるパケットの SL をダウングレードすることを許可する	
PV_LAB_SL_FILE	プロセスがプロセスの認可の対象になるオブジェクト SL を変更することを許可する	
PV_LAB_SL_PROC	プロセスがプロセスの認可の対象になるサブジェクト SL を変更することを許可する	

Trusted AIX 特権	説明	システム・コール参照
PV_LAB_SL_SELF	プロセスがプロセスの認可の対象になるプロセス自体の SL を変更することを許可する	
PV_LAB_SLUG	プロセスがプロセスの認可の対象になる SL をアップグレードすることを許可する	
PV_LAB_SLUG_STR	プロセスがプロセスの認可の対象になるパケットの SL をアップグレードすることを許可する	
PV_LAB_TL	プロセスがサブジェクトおよびオブジェクト TL を変更することを許可する	
PV_LAB_	上位のすべてのラベル特権 (PV_LAB_*) を結合したものと同等	
PV_MAC_CL	プロセスが機密性認可の制限をバイパスすることを許可する	
PV_MAC_R_PROC	ターゲット・プロセスのラベルがプロセスの認可への対処の範囲内にあれば、プロセスに関する情報を読み取るときに、プロセスが MAC 読み取り制限をバイパスすることを許可する	
PV_MAC_W_PROC	ターゲット・プロセスのラベルがプロセスの認可への対処の範囲内にあれば、プロセスへシグナルを送信するときに、プロセスが MAC 書き込み制限をバイパスすることを許可する	
PV_MAC_R	プロセスが MAC 読み取り制限をバイパスすることを許可する	
PV_MAC_R_CL	オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 読み取り制限をバイパスすることを許可する	
PV_MAC_R_STR	メッセージのラベルがプロセスの認可の範囲内にあれば、 STREAM からメッセージを読み取るときに、プロセスが MAC 読み取り制限をバイパスすることを許可する	
PV_MAC_W	プロセスが MAC 書き込み制限をバイパスすることを許可する	
PV_MAC_W_CL	オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 書き込み制限をバイパスすることを許可する	

Trusted AIX 特権	説明	システム・コール参照
PV_MAC_W_DN	プロセス・ラベルがオブジェクトのラベルより上位にあり、オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 書き込み制限をバイパスすることを許可する	
PV_MAC_W_UP	プロセス・ラベルがオブジェクトのラベルより下位にあり、オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 書き込み制限をバイパスすることを許可する	
PV_MAC_OVRRD	MAC の対象外であるとフラグを立てられたファイルに対して、MAC 制限をバイパスする	
PV_MAC_	上位のすべての MAC 特権 (PV_MAC_*) を結合したものと同等	
PV_MIC	プロセスが保全性制限をバイパスすることを許可する	
PV_MIC_CL	プロセスが保全性認可の制限をバイパスすることを許可する	

ドメイン RBAC

AIX 6.1 で導入された役割ベースのアクセス制御 (RBAC) は、スーパーユーザー root のさまざまな機能を複数のロールに分割するメカニズムを提供します。ロールは、システム上の他のユーザーに委任することができます。 RBAC は任務を委任する機能を提供し、システムのセキュリティーを向上させます。システム上のアクティビティーの監査とトラッキングがより容易になるからです。 RBAC は責務を別のユーザー (許可ユーザーと呼ばれます) に委任しますが、システムの特定のリソースへの許可ユーザーの管理権限を制限するメカニズムは提供しません。 例えば、ネットワーク管理権限を持つユーザーは、システム上のすべてのネットワーク・インターフェースを管理できます。 いくつかのインターフェースについては、許可ユーザーが変更できないように制限することはできません。

RBAC のドメイン機能は、許可ユーザーへのアクセスを制限するのに使用されます。 システムのユーザーとリソースは、ドメインと呼ばれるタグを付けることによって分類され、ユーザーのリソースへのアクセスは個別のアクセス規則で決定されます。

定義　以下の定義はアクセス規則に関連しています。

サブジェクト: サブジェクトは、オブジェクトへのアクセスを要求するエンティティーです。 サブジェクトの一例はプロセスです。

オブジェクト: オブジェクトは有用な情報を保持するエンティティーです。 オブジェクトの例として、ファイル、デバイス、およびネットワーク・ポートがあります。

ドメイン: ドメインは、エンティティーが属するカテゴリーとして定義されます。 エンティティーがドメインに属していると、そのエンティティーへのアクセス制御は以下のようにアクセス規則によって管理されます。

アクセス規則

- 1 つのサブジェクトが、1 つのオブジェクトが属するドメインをすべて持っているとき、そのサブジェクトはそのオブジェクトにアクセスできます。 これは、そのサブジェクトが属するドメインのリストが、オブジェクトのドメインのスーパー・セットであることを示しています。 これはデフォルトの動作です。
- 1 つのサブジェクトが 1 つのオブジェクトの少なくとも 1 つのドメインを持っているとき、そのサブジェクトはオブジェクトにアクセスできます。 つまり、そのサブジェクトとオブジェクトは共通のドメインを 1 つ持っています。 このときの挙動はオブジェクトのセキュリティー・フラグにより異なります。
- オブジェクトは、特定のドメインへのアクセスを拒否できます。 conflict set と呼ばれるドメインのセットをオブジェクトが定義しているときに、サブジェクトのドメインの 1 つが conflict set の一部である場合、そのオブジェクトはそのサブジェクトへのアクセスを拒否できます。

ドメイン・データベース

システムがサポートするドメインは、/etc/security/domains の下の構成ファイルに保管されている必要があります。 ファイル内のスタンザのフォーマットは、以下に示すとおりです。

```
domain-name:
id = <number>
dfltmsg = <Message>
msgcat = <Message catalog>
msgset = <Message set in catalog>
msgnum = <Message id in catalog>
```

このデータベースは、**mkdom** コマンドと **chdom** コマンドを使用して操作することができます。データベースを表示するには、**lsdom** コマンドを使用します。 エントリーを削除するには、**rmdom** コマンドを使用します。

データベース内のエントリーは、データベースが **setkst** コマンドを使用してカーネルにダウンロードされるまで、有効にはなりません。

最大で 1024 個のドメインがシステムでサポートされているため、ドメイン ID (ID 属性) として指定できる最大値は 1024 となります。

ドメイン割り当てオブジェクト

ドメインをオブジェクトに割り当てるには、それをドメイン割り当てオブジェクト・データベースに定義する必要があります。 システム上のすべてのエンティティーのドメインは、/etc/security/domobjs の下の構成ファイルに保管されています。ファイル内のスタンザのフォーマットは以下に示すとおりです。これは、ドメインをオブジェクトに割り当てる例です。

```
/dev/hrvg:
domains=HR,IT
conflictsets=payroll
objtype=device
secflags=FSF_DOM_ANY
```

domains: オブジェクトへのアクセスが許可されるドメインを指定します。 ドメインの例としては、IT、HR、および Payroll などが考えられます。

objtype: ドメインに割り当てられるオブジェクトのタイプを示します。 objtype の種類には、device、file、netint、および netport があります。

conflict sets: サブジェクトが、この属性にリストされているドメインのこのセットの中のいずれかに属している場合、オブジェクトへのアクセスが許可されないことを示しま。

secflags: このフラグはオブジェクトの特殊なプロパティーを指定します。 フラグは **FSF_DOM_ANY** または **FSF_DOM_ALL** に設定することができます。フラグが **FSF_DOM_ANY** に設定されると、domains 属性リストに指定されたドメインのいずれか 1 つがサブジェクトに含まれるときに、サブジェクトはオブジェクトにアクセスできます。 フラグが **FSF_DOM_ALL** に設定されると、オブジェクトにアクセスするには、サブジェクトがリスト内のすべてのドメインの条件を満たしている必要があります。値が指定されないと、デフォルト値の **FSF_DOM_ALL** が使用されます。 **secflag** は、オブジェクトの domains 属性の動作にのみ影響を与えます。

ドメインは、ファイルシステム内のファイルに割り当てることができます。 デフォルトでは、オブジェクトのすべてのドメインをプロセスのドメインのサブセットにして、プロセスがオブジェクトにアクセスできるようにする必要があります。

1. デバイス: すべてのデバイス (ファイルシステムを含む) をドメインに割り当てることができます。 デバイスの構成などの管理アクティビティー時にドメイン検査が行われます。

   ```
   /dev/hrvg:
   domains=HR,IT
   conflictsets=payroll
   objtype=device
   secflags=FSF_DOM_ANY
   ```

2. ネットワーク・インターフェース: ネットワーク・インターフェース (例: en0) がドメインに割り当てられると、インターフェースのシャットダウンなどの管理アクティビティーが、インターフェースにドメイン検査を受けるよう要求します。

   ```
   en0:
   domains=NETIF,ADMIN
   objtype=netint
   flags=FSF_DOM_ALL
   ```

3. ネットワーク・ポート: TCP ポートと UDP ポートをドメインに割り当てることができます。アプリケーションがポートにバインドしようとすると、ドメイン検査が実行されます。

   ```
   TCP_<port#>:
   domains=NETIF,ADMIN
   type=netport
   flags=FSF_DOM_ALL
   ```

4. プロセス: プロセスは、そのプロセスが実行される元となったユーザーのドメインを継承します。 ユーザーがログインすると、ユーザー・シェル・プロセスはそのユーザーのドメインを持つようになります。 ドメインが設定されると、プロセスのこのようなドメインは存続期間中保持されます。 プロセスのドメインは、いかなるユーザー・インターフェースでもシステム・コールでも変更することができません。 ドメインを設定できるプロセスはログイン・プロセスだけです。 プロセスは **conflict set** 属性も **secflags** 属性も持ちません。

現在の制約

次の項目は、現在のドメイン RBAC 機能の制約です。

- ドメイン構成ファイルは現在、ローカル・システムではサポートされていますが、Lightweight Directory Access Protocol (LDAP) サーバーではサポートされていません。
- RBAC ドメインは、AIX ワークロード・パーティション (WPAR) 内ではサポートされません。
- RBAC ドメインを一時ファイルに適用することはできません。

拡張 RBAC の要件

ドメイン RBAC は拡張 RBAC で作成されるため、拡張 RBAC がシステム上で使用可能にされ、有効になっている必要があります。

カーネルのセキュリティー・テーブル

ドメイン・データベースとドメイン・オブジェクト・データベースで定義されたドメインとドメイン割り当てオブジェクトは、**setkst** コマンドを使用してカーネルにダウンロードされた後で、有効になります。この 2 つのテーブルは、カーネル・ドメイン・テーブル (KDOMT) およびカーネル・ドメイン・オブジェクト・テーブル (KDOT) と呼ばれます。

カーネル・セキュリティー・テーブルと **setkst** について、詳しくは「AIX セキュリティー・ガイド」の役割ベースのアクセス制御 (RBAC) のトピックを参照してください。

ドメイン・コマンド

以下の表では、ドメイン RBAC フレームワークを管理して使用するために AIX オペレーティング・システムで提供されているドメイン RBAC 関連コマンドをリストします。

コマンド	説明
mkdom	新規ドメインを作成します。
lsdom	ドメイン属性を表示します。
rmdom	ドメインを削除します。
chdom	ドメイン属性を変更します。
setsecattr	ドメイン・オブジェクト・データベースのセキュリティー属性を設定します。
lssecattr	ドメイン・オブジェクト・データベースのセキュリティー属性を表示します。
rmsecattr	ドメイン・オブジェクト・データベースの定義を削除します。
setkst	ドメイン RBAC ユーザー・レベル・データベース内のエントリーをカーネル・セキュリティー・テーブルへ送信します。

ドメイン RBAC 関連ファイル

以下の表では、データベース情報を構成し、保管するために AIX オペレーティング・システムで提供されている RBAC 関連ファイルをリストしています。

ファイル	説明
/etc/security/domains	ドメイン・データベース
/etc/security/domobjs	ドメイン・オブジェクト・データベース

ドメインの使用

ドメインの定義: ドメインは、**mkdom** コマンドを使用してドメイン・データベースに定義されます。

```
mkdom id=24 HR
```

ドメインの割り当て: ドメインは、ユーザー、ファイル、デバイス、ネットワーク・ポート、およびインターフェースなどのエンティティーに割り当てることができます。 ユーザー以外のすべてのエンティティーは、conflict set とセキュリティー・フラグ (**secflags**) をサポートします。

ユーザー: ユーザーは、**chuser** コマンドと **chsec** コマンドを使用してドメインに割り当てられます。

構文:

```
chuser domains = <comma-separated list of domains> username
```

例:

```
chuser domains=INET john
```

ユーザーに割り当てられたドメインは、ログイン時に活動状態にされます。セッションが活動状態のときにドメインが変更された場合は、再度ログインして、新しいドメインが有効になるようにする必要があります。

オブジェクト: ドメインを経由したオブジェクトへのアクセスを制限するには、**setsecattr** コマンドを使用して、オブジェクトがドメイン・オブジェクト・データベースに定義される必要があります。

構文:

```
setsecattr -o domains=<comma-separated list of allowed domains>
conflictsets=<comma-separated list of restricted domains>
secflags=<FSF_DOM_ALL or FSF_DOM_ANY>
objtype=<file or device or netint or netport>
object-path
```

例:

```
setsecattr -o domains=INET,WEB conflictsets=DB secflags=FSF_DOM_ANY objtype=netint en0
```

アクセス制御リスト

通常、ACL はアクセス制御エントリー (ACE) と呼ばれる一連のエントリーで構成されます。 各 ACE は、オブジェクトとの関係においてユーザーのアクセス権を定義します。

アクセスが試みられると、オペレーティング・システムはオブジェクトに関連する ACL を使用して、ユーザーにそのアクセス権があるかどうかを確認します。 これらの ACL と関連のアクセス検査が、AIX でサポートされる任意アクセス制御 (DAC) メカニズムの中核を成します。

オペレーティング・システムは各種のシステム・オブジェクトをサポートしており、これらのオブジェクトにより、ユーザー・プロセスは情報の保管やコミュニケーションが可能となります。 最も重要なタイプのアクセス制御対象オブジェクトは、次のとおりです。

- ファイルおよびディレクトリー
- 名前付きパイプ
- メッセージ・キュー、共有メモリー・セグメント、およびセマフォーなどの IPC オブジェクト

これらのオブジェクトにおけるすべてのアクセス許可検査は、オブジェクトが最初にアクセスされたときのシステム・コール・レベルで行われます。 System V プロセス間通信 (SVIPC) オブジェクトはステートレスにアクセスされるため、すべてのアクセスで検査が実行されます。 ファイルシステム名を持つオブジェクトの場合、実オブジェクトの名前を解決できることが必要です。名前は、相対的 (プロセスの作業ディレクトリーに対して)、または絶対的に (プロセスの root ディレクトリーに対して) 解決されます。 すべてのネーム・レゾリューションは、これらのディレクトリーの 1 つを検索することによって開始されます。

任意アクセス制御メカニズムにより、情報リソースの有効アクセス制御が許可され、情報の機密性および整合性を別個に保護します。所有者が制御するアクセス制御メカニズムは、ユーザーがそれらを実行した場合と同じほどの有効性しかありません。 すべてのユーザーは、アクセス許可が付与および拒否される方法、および設定方法を理解しておく必要があります。

例えば、ファイルシステム・オブジェクト (ファイルまたはディレクトリー) に関連する ACL は、そのオブジェクトへのアクセスに関して、さまざまなユーザーに対してアクセス権を施行できます。 こうした ACL は、異なるユーザーに対し、読み取りや書き込みといったさまざまなレベルのアクセス権を施行できます。

通常、各オブジェクトには所有者が定義され、一部のケースでは、オブジェクトが 1 次グループに関連付けられることもあります。 特定のオブジェクトの所有者は、その任意のアクセス属性をコントロールします。 所有者の属性は、オブジェクト作成プロセスの実効ユーザー ID に設定されます。

以下のリストは、さまざまなタイプのオブジェクトの直接アクセス制御属性を示しています。

所有者
: System V プロセス間通信 (SVIPC) オブジェクトの場合には、作成者または所有者がオブジェクトの所有権を変更できます。 SVIPC オブジェクトには、所有者のすべての権限 (アクセス許可を含め) を持つ関連作成者があります。 この作成者は、root 権限によっても変更できません。

  SVIPC オブジェクトは、作成プロセスの実効グループ ID に初期化されます。 ファイルシステム・オブジェクトの場合には、直接アクセス制御属性は、オブジェクト作成プロセスの実効グループ ID、または親ディレクトリーのグループ ID (親ディレクトリーのグループ継承フラグにより決定される) のいずれかに初期化されます。

グループ
: オブジェクトの所有者はグループを変更できます。 新規グループは、作成プロセスの実効グループ ID か親ディレクトリーのグループ ID のいずれかでなければなりません。 (上記のように、SVIPC オブジェクトは、変更できない関連作成グループを持ち、オブジェクト・グループのアクセス許可を共用します。)

モード
: **chmod** コマンド (8 進数表記の数字モードの) は、基本アクセス権および属性を設定できます。このコマンドで呼び出される **chmod** サブルーチンは、拡張アクセス権を使用不可にします。ACL を持つファイルに対して **chmod** コマンドの数字モードを使用すると、拡張アクセス権が使用不可になります。 **chmod** コマンドのシンボリック・モードは、NSF4 ACL タイプの拡張 ACL を使用不可にしますが、AIXC タイプ ACL の拡張アクセス権を使用不可にしません。 数字およびシンボリック・モードに関する詳細は、**chmod** を参照してください。

ソケットやファイルシステム・オブジェクトなど、オペレーティング・システム内の多くのオブジェクトには、さまざまなサブジェクトに関する ACL が関連付けられています。 これらのオブジェクト・タイプの ACL の詳細は、1 つ 1 つ異なります。

従来、AIX ではファイルシステム・オブジェクトへのアクセスを制御するために、モード・ビットをサポートしてきました。 また、モード・ビットに関連する固有形式の ACL もサポートしてきました。 この ACL は基本モード・ビットで構成され、複数の ACE エントリーの定義も可能でした。これらの各 ACE エントリーが、モード・ビットに関連するユーザーまたはグループのアクセス権を定義しました。 このクラシック・タイプの ACL の動作は AIXC ACL タイプという名前で今後も継続してサポートされます。

ファイルシステム・オブジェクトでの ACL のサポートは、その基礎にある物理ファイルシステム (PFS) に依存しています。 この PFS は、ACL データを理解し、さまざまなユーザーのアクセスを保管、検索、

および施行できなければなりません。 複数のタイプの ACL をサポートする物理ファイルシステムとは対照的に、中には ACL をまったくサポートしない (基本モード・ビットしかサポートしない) 物理ファイルシステムも存在する可能性があります。 AIX でいくつかのファイルシステムが複数の ACL タイプをサポートするように拡張されています。JFS2 と GPFS™ は、NFS バージョン 4 プロトコル・ベースの ACL タイプもサポートする機能を持つことになります。 この ACL は、AIX における NFS4 ACL タイプと呼ばれています。この ACL タイプは、NFS バージョン 4 プロトコル仕様のほとんどの ACL 定義を順守しています。 また、このタイプは AIXC ACL タイプと比較してより精細なアクセス制御をサポートし、継承などの機能も提供します。

複数アクセス制御リスト・タイプのフレームワーク・サポート

AIX オペレーティング・システムではバージョン 5.3.0 以降、オペレーティング・システム内で複数のファイルシステム・オブジェクトに対して複数のアクセス制御リスト (ACL) タイプが存在するインフラストラクチャーをサポートするようになりました。

このインフラストラクチャーでは、オブジェクトに関連する ACL タイプに関係なく、統一されたメソッドで ACL を管理できます。 このフレームワークには、以下のコンポーネントが含まれます。

ACL 管理コマンド
: これらのコマンドは、**aclget**、**aclput**、**acledit**、**aclconvert**、**aclgetttypes** などです。これらのコマンドは、 ACL タイプ固有のモジュールを起動する、ライブラリー・インターフェースを呼び出します。

ACL ライブラリー・インターフェース
: ACL ライブラリー・インターフェースは、ACL をアクセスする必要のあるアプリケーションに対してフロントエンドとして働きます。

ACL タイプ固有の動的ロード可能な **ACL** モジュール
: AIX オペレーティング・システムは、AIX Classic ACL (**AIXC**) 用および NFS4 ACL (**nfs4**) 用の ACL タイプ固有のモジュールのセットを提供しています。

バイナリー互換性**:**

既存の JFS2 ファイルシステム (既存の AIX ACL の有無に関係なく) 上で実行されるアプリケーションについては互換性の問題は存在しません。

ただし、より厳格な ACL (NFS4 など) が関連付けられたファイルシステム・オブジェクトが検出された場合、ファイルへのアクセスが失敗する可能性があることを、アプリケーションが認識する場合もあります。 ファイルの存在の有無を確認する単純検査には、NFS4 ACL では読み取りレベルの許可が必要です。

AIX オペレーティング・システムでサポートされるアクセス制御リスト・タイプ

AIX オペレーティング・システムは、現在 AIXC タイプおよび NFS4 ACL タイプをサポートしています。

既に説明したように、AIX は、基礎となる物理ファイルシステムがサポートする他のあらゆる ACL タイプを追加できるインフラストラクチャーもサポートします。 JFS2 PFS は、ファイルシステムのインスタンスが拡張属性バージョン 2 機能によって作成された場合、NFS4 ACL をネイティブにサポートします。

AIXC アクセス制御リスト**:**

AIXC アクセス制御リスト・タイプは、5.3.0 より以前の AIX リリースでサポートされた ACL タイプの動作を示します。AIXC ACL には、基本アクセス権と拡張アクセス権が含まれます。

AIXC アクセス制御リスト (ACL) タイプは、5.3.0 より以前の AIX リリースでサポートされた ACL タイプの動作を示します。 AIXC ACL には、基本アクセス権と拡張アクセス権が含まれます。 JFS2 ファイルシステムで AIXC ACL について許される最大サイズは 4KB です。

AIXC ACL の基本アクセス権の設定

基本アクセス権は、ファイル所有者、ファイル・グループ、および他のユーザーに割り当てられた、従来のファイル・アクセス・モードです。 アクセス・モードは、読み取り (r)、書き込み (w)、および実行/検索 (x) です。

ACL では、基本アクセス権は、次のフォーマットで示されます。ただし、*Mode* パラメーターは rwx として表されます (未指定の各アクセス権はハイフン (-) に置換)。

```
base permissions:
   owner(name): Mode
   group(group): Mode
   others: Mode
```

AIXC ACL の属性の設定

以下の属性を AIXC ACL に追加することができます。

setuid (SUID)
: セット・ユーザー ID モード・ビット。 この属性は、プロセスの有効ユーザー ID および保存済みユーザー ID を、実行時のファイルの所有者 ID に設定します。

setgid (SGID)
: セット・グループ ID モード・ビット。 この属性は、プロセスの有効グループ ID および保存済みグループ ID を、実行時のファイルのグループ ID に設定します。

savetext (SVTX)
: ディレクトリーにおいて、指定されたディレクトリーで、ファイルの所有者だけがファイルにリンクしたり、リンクを解除したりできることを示します。

これらの属性は、次のフォーマットで追加されます。

```
attributes: SUID, SGID, SVTX
```

AIXC アクセス **ACL** の拡張アクセス権の設定

拡張アクセス権により、ファイルの所有者は、そのファイルに対するアクセスをさらに厳密に定義できます。 拡張アクセス権は、特定の個人、グループ、またはユーザーおよびグループの組み合わせに対して、アクセス・モードを許可、拒否、または指定することによって、基本ファイル・アクセス許可 (所有者、グループ、その他) を変更します。アクセス権は、キーワードの使用によって変更されます。

permit、**deny**、および **specify** キーワードは次のように定義されます。

permit
: ユーザーまたはグループに、ファイルへの指定されたアクセスを付与する

deny ユーザーまたはグループについて、ファイルへの指定されたアクセスの使用を制限する

specify
ユーザーまたはグループへのファイル・アクセスを正確に定義する

ユーザーが、**deny** または **specify** キーワードのどちらかによって特定のアクセスを拒否された場合、そのアクセス否認をオーバーライドできる他のエントリーはありません。

拡張アクセス権を有効にするには、ACL で **enabled** キーワードを指定することが必要です。 デフォルト値は **disabled** キーワードです。

ACL では、拡張アクセス権は次のフォーマットです。

```
extended permissions:
  enabled | disabled
    permit   Mode  UserInfo...
    deny     Mode  UserInfo...
    specify  Mode  UserInfo...
```

それぞれの **permit**、**deny**、または **specify** エントリーごとに、別個の行を使用してください。 *Mode* パラメーターは、**rwx** として表されます (未指定の各アクセス権をハイフン (-) に置換)。 *UserInfo* パラメーターは、u:UserName、または g:GroupName、またはコンマで区切られた u:UserName と g:GroupName の組み合わせとして表されます。

注: 1 つのプロセスには 1 つのユーザー ID しかないため、1 つのエントリーに複数のユーザー名が指定された場合、そのエントリーはアクセス制御の決定に使用できません。

AIXC ACL のテキスト表示

以下のスタンザは AIXC ACL のテキスト表示を示しています。

```
Attributes:  { SUID | SGID | SVTX }
Base Permissions:
   owner(name): Mode
   group(group): Mode
   others: Mode
Extended Permissions:
   enabled | disabled
     permit   Mode  UserInfo...
     deny     Mode  UserInfo...
     specify  Mode  UserInfo...
```

AIXC ACL のバイナリー・フォーマット

AIXC ACL のバイナリー・フォーマットは /usr/include/sys/acl.h に定義され、現行の AIX リリースで実装されています。

AIXC ACL の例

以下は AIXC ACL の 1 例です。

```
attributes: SUID
base permissions:
      owner(frank):  rw-
      group(system): r-x
      others: ---
extended permissions:
      enabled
        permit  rw-  u:dhs
        deny    r--  u:chas, g:system
        specify r--  u:john, g:gateway, g:mail
        permit  rw-  g:account, g:finance
```

以下に、ACL エントリーについて説明します。

- 最初の行は **setuid** ビットがオンにされたことを示します。
- 次の行は、基本アクセス権を紹介するもので、オプションです。
- 続く 3 行は、基本アクセス権を指定します。 括弧で囲まれた所有者およびグループ名は、情報を提供しているだけです。 これらの名前を変更しても、ファイル所有者、またはファイル・グループを変更することにはなりません。 **chown** コマンドおよび **chgrp** コマンドのみがこれらのファイル属性を変更できます。
- その次の行は、拡張アクセス権を紹介するもので、オプションです。
- 次の行は、続く拡張アクセス権が使用可能になっていることを示します。
- 最後の 4 行は、拡張エントリーです。 最初の拡張エントリーは、ユーザー *dhs* に、ファイル上で読み取り (r) および書き込み (w) 許可を付与します。
- 2 番目の拡張エントリーは、ユーザー *chas* が、*system* グループのメンバー所属である場合のみ、読み取り (r) アクセスを拒否します。
- 3 番目の拡張エントリーは、ユーザー *john* が *gateway* グループおよび *mail* グループ両方のメンバーである限り、読み取り (r) アクセスがあることを指定します。 ユーザー *john* が両方のグループのメンバー所属先でないと、この拡張アクセス権は適用されません。
- 最後の拡張エントリーは、*account* グループおよび *finance* グループ両方 のすべてのユーザーに、読み取り (r) および書き込み (w) 許可を付与します。

  注**:** 複数の拡張エントリーを、制御下のオブジェクトへのアクセスを要求しているプロセスに適用することができます。その際、制限エントリーは許容モードに優先します。

完全な構文については、「コマンド・リファレンス」の『**acledit** コマンド』を参照してください。

NFS4 アクセス制御リスト**:**

AIX では、さらに NFS4 アクセス制御リスト (ACL) タイプもサポートします。

NFS4 ACL タイプは、「ネットワーク・ファイルシステム *(NFS)* バージョン *4* プロトコル *RFC 3530*」で指定されたとおりに、アクセス・コントロールを実装します。 JFS2 ファイルシステムで NFS4 ACL について許される最大サイズは 64KB です。

NFS V4 クライアントのみが NFS V4 ACL をサポートします。 キャッシュ・ファイルとプロキシーの両方は NFS V4 ACL をサポートしません。

NFS4 ACL のテキスト表示

テキスト形式 NFS V4 ACL は、ACE (アクセス制御エントリー) のリストで、各 ACE が 1 行に示されます。 1 つの ACE は、以下の形式の 4 つのエレメントから構成されます。

```
IDENTITY   ACE_TYPE      ACE_MASK     ACE_FLAGS

where:
IDENTITY => Has format of 'IDENTITY_type:(IDENTITY_name or IDENTITY_ID or IDENTITY_who):'
         where:
         IDENTITY_type => One of the following Identity type:
                 u : user
                 g : group
                 s : special who string (IDENTITY_who must be a special who)
                         IDENTITY_name => user/group name
                         IDENTITY_ID   => user/group ID
                         IDENTITY_who  => special who string (e.g. OWNER@, GROUP@, EVERYONE@)
ACE_TYPE => One of the following ACE Type:
                a : allow
```

```
                d : deny
                l : alarm
                u : audit
ACE MASK => One or more of the following Mask value Key without separator:
                r : READ_DATA   or LIST_DIRECTORY
                w : WRITE_DATA      or ADD_FILE
                p : APPEND_DATA     or ADD_SUBDIRECTORY
                R : READ_NAMED_ATTRS
                W : WRITE_NAMED_ATTRS
                x : EXECUTE         or SEARCH_DIRECTORY
                D : DELETE_CHILD
                a : READ_ATTRIBUTES
                A : WRITE_ATTRIBUTES
                d : DELETE
                c : READ_ACL
                C : WRITE_ACL
                o : WRITE_OWNER
                s : SYNCHRONIZE
ACE_FLAGS (Optional) => One or more of the following Attribute Key without separater:
                fi : FILE_INHERIT
                di : DIRECTORY_INHERIT
                oi : INHERIT_ONLY
                ni : NO_PROPAGATE_INHERIT
                sf : SUCCESSFUL_ACCESS_ACE_FLAG
                ff : FAILED_ACCESS_ACE_FLAG
```

注**:** SYNCHRONIZE Ace_Mask 値キー s に関して、AIX はこの値キーに関して何のアクションも実行しません。 AIX オペレーティング・システムは s 値キーを保管および保持しますが、この値キーは AIX にとって何の意味も持ちません。

WRITE_OWNER Ace_Mask が Ace_Type allow に設定されると、各ユーザーはファイルの所有権を変更してユーザー自身専用にすることができます。

ファイルの削除は、2 つの ACE によって決まります。すなわち、削除するオブジェクトの DELETE エントリー、およびその親ディレクトリーの DELETE_CHILD エントリーです。 AIX オペレーティング・システムは、ユーザーに 2 つのモードの動作を提供します。セキュア ・モードでは、DELETE は AIXC ACL と同様に動作します。 互換 モードでは、DELETE は NFS4 ACL の他の主要なインプリメンテーションと同様に動作します。 互換モードをオンにするには、次のように **chdev** コマンドを使用します。

```
chdev -l sys0 -a nfs4_acl_compat=compatible
```

構成変更が行われる前に、**chdev** コマンドの実行後にシステムをリブートする必要があります。

ご使用のシステムを 2 つのモード間で交互に切り替える場合は、セキュア・モードで AIX オペレーティング・システムによって生成された NFS4 ACL は、システムが互換モードに変更されても、他のプラットフォームでは受け入れられない可能性があります。

```
Example:
        u:user1(aa@ibm.com):     a    rwp    fidi
        *s:(OWNER@):             d    x      dini          * This line is a comment
        g:staff(jj@jj.com):      a    rx
        s:(GROUP@):              a    rwpx   fioi
        u:2:                     d    r     di             * This line shows user bin (uid=2)
        g:7:                     a    ac    fi             * This line shows group security (gid=7)
        s:(EVERYONE@):           a    rca   ni
```

NFS4 ACL のバイナリー・フォーマット

NFS4 ACL のバイナリー・フォーマットは /usr/include/sys/acl.h に定義され、現行の AIX リリースで実装されています。

NFS4 ACL の例

以下の例は、ディレクトリー (例えば、**j2eav2/d0**) に適用された NFS4 ACL を示します。

```
s:(OWNER@):         a         rwpRWxDdo         difi         * 1st   ACE
s:(OWNER@):         d         D                 difi         * 2nd  ACE
s:(GROUP@):         d         x                 ni           * 3rd   ACE
s:(GROUP@):         a         rx                difi         * 4th   ACE
s:(EVERYONE@):      a         c                 difi         * 5th   ACE
s:(EVERYONE@):      d         C                 difi         * 6th   ACE
u:user1:            a         wp                oi           * 7th   ACE
g:grp1:             d         wp                             *  8th  ACE
u:101:              a         C                              * 9th   ACE
g:100:              d         c                              * 10th  ACE
```

以下に、ACL エントリーについて説明します。

- 最初の ACE は、所有者が **/j2eav2/d0** ならびに当該 ACL が適用された後に作成されたすべての成果について以下の特権を有することを示します。
  - READ_DATA (= LIST_DIRECTORY)
  - WRITE_DATA (=ADD_FILE)
  - APPEND_DATA (= ADD_SUBDIRECTORY)
  - READ_NAMED_ATTR
  - WRITE_NAMED_ATTR
  - EXECUTE (=SEARCH_DIRECTORY)
  - DELETE_CHILD
  - DELETE
  - WRITE_OWNER
- 2 番目の ACE は、所有者が DELETE_CHILD に対する特権を否認されたこと (**/j2eav2** の下で作成されたファイルおよびサブディレクトリーが削除される) を示しますが、所有者は、最初の ACE (これによって所有者は DELETE_CHILD 特権が与えられている) により、依然として、それらを削除することができることを示します。
- 3 番目の ACE は、当該オブジェクト (**/j2eav2/d0**) のグループの全メンバーが EXECUTE (=SEARCH_DIRECTORY) の特権を拒否されながら、所有者は、依然として、最初の ACE により当該特権を許可されていることを示します。 この ACE は、NO_PROPAGATE_INHERIT フラグが指定されているため、そのすべての成果に伝搬させることはできません。 この ACE は、ディレクトリー **/j2eav2/d0** およびその直後の子ファイルとサブディレクトリーにのみ適用されます。
- 4 番目の ACE は、オブジェクト (**/j2eav2/d0**) のグループの全メンバーが、**/j2eav2/d0** およびそのすべての成果に対して、READ_DATA (= LIST_DIRECTORY) および EXECUTE (=SEARCH_DIRECTORY) 特権を有していることを示します。 ただし、3 番目の ACE グループ・メンバー (所有者を除く) には、**/j2eav2/d0** ディレクトリーおよびその直後の子ファイルおよびサブディレクトリーに対する EXECUTE (=SEARCH_DIRECTORY) 特権が許されていません。
- 5 番目の ACE は、**/j2eav2/d0** ディレクトリーおよびこの ACE が適用された後に作成されたすべての成果に対する READ_ACL 特権が全員に許されていることを示します。
- 6 番目の ACE は、**/j2eav2/d0** ディレクトリーおよびその成果に対する WRITE_ACL 特権について全員が拒否されていることを示します。 所有者は、常に、 NFS4 ACL 付きのファイルおよびディレクトリーに対して WRITE_ACL 特権を有しています。
- 7 番目の ACE は、user1 が WRITE_DATA (=ADD_FILE) および APPEND_DATA (= ADD_SUBDIRECTORY) 特権を、**/j2eav2/d0** ディレクトリーのすべての成果については有しているが、**/j2eav2/d0** ディレクトリー自体については有していないことを示します。

- 8 番目の ACE は、grp1 の全メンバーが WRITE_DATA (=ADD_FILE) および APPEND_DATA (= ADD_SUBDIRECTORY) の特権を拒否されていることを示します。 この ACE は、最初の ACE により、所有者が仮に grp1 に属していても、所有者には適用されません。
- 9 番目の ACE は、**UID 101** のユーザーは WRITE_ACL 特権を有しているが、6 番目の ACE により、全員 (所有者を除く) が、WRITE_ACL 特権を有していないことを示します。
- 10 番目の ACE は、**GID 100** のグループの全メンバーが READ_ACL 特権を拒否されているが、5 番目の ACE により、この特権を有することを示します。

アクセス制御リストの管理

コマンドを使用して ACL の表示と設定ができます。

アプリケーション・プログラマーおよびその他のサブシステム開発者は、このセクションで述べる ACL ライブラリー・インターフェースならびに ACL 変換ルーチンを使用することができます。

ACL 管理コマンド

以下のコマンドを使用して、ファイルシステム・オブジェクトの ACL を扱うことができます。

aclget *FileObject* という名前のファイル・オブジェクトの ACL を標準出力に、読み取り可能なフォーマットで書き込むか、または同様のものを *outAclFile* という名前の出力ファイルに書き込みます。

aclput
FileObject の ACL を、標準出力、すなわち *inAclFile* を介して指定された入力を使用して、ファイルシステム上に設定します。

acledit
指定された *FileObject* の ACL を編集するために、エディターを開きます。

aclconvert
ACL のタイプを別のタイプに変換します。 変換がサポートされていない場合は、このコマンドは失敗します。

aclgettypes
ファイルシステム・パスでサポートされている ACL タイプを取得します。

ACL ライブラリー・インターフェース

ACL ライブラリー・インターフェースは、ACL をアクセスする必要のあるアプリケーションに対してフロントエンドとして働きます。 アプリケーション (前述の汎用 ACL 管理コマンドを含む) は、登録されていない ACL syscall を直接呼び出すことはしません。その代わりに、汎用 syscall およびタイプ固有のロード可能なモジュールを、ライブラリー・インターフェースを介してアクセスします。 これにより、カスタマーのアプリケーション・プログラマーはロード可能なモジュールの複雑な用法から保護され、将来の AIX リリースにおけるバックワード・バイナリー互換性の問題が低減されます。

以下のライブラリー・インターフェースは syscall を呼び出します。

aclx_fget および **aclx_get**
aclx_get および **aclx_fget** 機能は、ファイルシステム・オブジェクトのアクセス・コントロール情報を検索し、**acl** により指定されたメモリー領域にそれを書き込みます。 **acl** のサイズとタイプ情報は、***acl_sz** と ***acl_type** に保管されます。

aclx_fput および **aclx_put**
aclx_put および **aclx_fput** 機能は、入力ファイル・オブジェクトの **acl** に指定されたアクセス・

コントロール情報を保管します。これらの機能は、ACL タイプ変換は行いません。ACL タイプ変換を行うには、呼び出し側が **aclx_convert** 機能を明確に呼び出す必要があります。

aclx_gettypes
aclx_gettypes 機能は、特定のファイルシステムでサポートされている ACL タイプのリストを取得します。 1 つのファイルシステム・タイプが複数の ACL タイプを同時にサポートすることができます。 各ファイルシステム・オブジェクトは、そのファイルシステムによりサポートされる ACL タイプのリストに属する、固有の ACL タイプと関連しています。

aclx_gettypeinfo
aclx_gettypeinfo 機能は、パスで指定されたファイルシステム上の ACL タイプの特性と能力を取得します。 ACL 特性は、通常、それぞれの ACL タイプに固有なデータ構造タイプとなることに注意してください。 AIXC および NFS4 ACL に使用されるデータ構造は、別のドキュメントで述べます。

aclx_print および **aclx_printStr**
これらの 2 つの機能は、バイナリー・フォーマットの ACL をテキスト表示に変換します。 これらの機能は、**aclget** および **acledit** コマンドで呼び出されます。

aclx_scan および **aclx_scanStr**
これらの 2 つの機能は、ACL のテキスト表示をバイナリー・フォーマットに変換します。

aclx_convert
ACL のタイプを別のタイプに変換します。 この機能は、**cp**、**mv**、または **tar** のようなコマンドによる暗黙的な変換に使用します。

ACL 変換

ACL 変換を使用することにより、ACL のタイプを別のタイプに変換することができます。 複数の ACL タイプのサポートは、特定の物理ファイルシステムでどの ACL タイプがサポートされているかによって異なります。 すべてのファイルシステムが、すべての ACL タイプをサポートするとは限りません。 例えば、ファイルシステム 1 が AIXC ACL タイプのみをサポートし、ファイルシステム 2 は AIXC と NFS4 ACL タイプをサポートする場合があります。 2 つのファイルシステム間で AIXC ACL をコピーすることは可能ですが、ファイルシステム 2 からファイルシステム 1 へ NFS ACL をコピーする場合は、ACL 変換を使用する必要があります。 ACL 変換は、アクセス・コントロール情報をできる限り多く保存します。

注**:** 変換プロセスは近似であり、アクセス・コントロール情報が失われる可能性があります。 ACL 変換を計画する場合は、これを考慮に入れてください。

AIX オペレーティング・システムの ACL 変換は、以下のインフラストラクチャーを使用してサポートされます。

ライブラリー・ルーチン
これらのルーチンおよびユーザー・レベルの ACL フレームワークによって、ACL 変換による ACL のタイプから別のタイプへの変換が可能となります。

aclconvert コマンド
このコマンドは ACL を変換します。

aclput および **acledit** コマンド
これらのコマンドは ACL タイプを変更する場合に使用します。

cp および **mv** コマンド
これらのコマンドは、複数の ACL タイプを扱う場合に使用可能にされ、必要に応じて、内部 ACL 変換を行います。

backup コマンド
このコマンドは、レガシー・フォーマットのバックアップを行うように要求された場合、 ACL 情報を既知のタイプとフォームに変換します。 ACL をネイティブ・フォーマットで取得するには、**-U** オプションを指定します。詳しくは、『backup』を参照してください。

各 ACL タイプは固有であり、アクセス制御マスクの改良は、ACL のタイプによって大きく異なります。変換アルゴリズムは近似であり、ACL を手動で変換する場合と同等ではありません。 変換が正確でない場合があります。 例えば、 NFS4 ACL は正確に AIXC ACL に変換できません。その理由は、NFS4 ACL はアクセス・マスクを最大 16 提供しており、AIXC ACL タイプでサポートされていない継承フィーチャーを有しているからです。 アクセス・コントロール情報の紛失を懸念する場合は、 ACL 変換機能ならびにインターフェースを使用しないでください。

注**:** ACL 変換アルゴリズムは本質的にプロプラエタリーであり、変更される可能性があります。

S ビットとアクセス制御リスト

setuid および **setgid** プログラムを使用し、S ビットを ACL に適用することができます。

setuid および setgid プログラムの使用

アクセス権ビット・メカニズムにより、多くの状況でリソースに対する効果的なアクセス制御が可能になります。 しかし、より厳格なアクセス制御のために、オペレーティング・システムは **setuid** プログラムと **setgid** プログラムを提供します。

AIX オペレーティング・システムでは、ID を uids および gids としてのみ定義します。 ID を uids および gids で定義しない ACL タイプは、AIX ID モデルにマップされます。 例えば、NFS4 ACL タイプは、ユーザー ID を user@domain 形式の文字列として定義し、その文字列が数値の UID および GID にマップされます。

ほとんどのプログラムは、そのプログラムを呼び出したユーザーのユーザー・アクセス権とグループ・アクセス権で実行されます。 プログラムの所有者は、そのプログラムを **setuid** プログラムまたは **setgid** プログラムにすること、つまり許可フィールドに setuid ビットまたは setgid ビットが設定されたプログラムにすることによって、それらを呼び出すユーザーのアクセス権を関連付けることができます。 プロセスがそのプログラムを実行するとき、プロセスはそのプログラムの所有者のアクセス権を取得します。 **setuid** プログラムは、その所有者のアクセス権により実行され、**setgid** プログラムはそのグループのアクセス権を持ち、両ビットは許可メカニズムに従って設定することができます。

プロセスには追加のアクセス権が割り当てられますが、これらの権限はその権限を持つプログラムによってコントロールされます。 したがって、**setuid** プログラムと **setgid** プログラムは、アクセス権が間接的に与えられている、ユーザーがプログラミングしたアクセス制御に対応することができます。 このプログラムはユーザーのアクセス権を保護して、トラステッド・サブシステムとして動作します。

これらのプログラムの使用は非常に効果的ですが、注意して設計しないとセキュリティー上のリスクが生じます。 特に、プログラムが所有者のアクセス権を持っている間は、ユーザーに制御を戻すことがあってはなりません。そのような状態で制御を戻すと、ユーザーが所有者の権限を無制限に使用してしまう可能性があるからです。

注**:** セキュリティー上の理由で、オペレーティング・システムはシェル・スクリプト内で、**setuid** または **setgid** プログラム呼び出しをサポートしていません。

ACL への S ビットの適用

NFS4 などの ACL は S ビットを直接扱いません。 NFS4 ACL では、これらのビットをどのように ACL の一部として組み入れるかを指定していません。 AIX オペレーティング・システムではこの問題への対処方法として、S ビットがアクセス検査を実行するときに使用されるようにし、NFS4 ACL 関連のアクセス検査を S ビットで補完するようにしています。 AIX オペレーティング・システムで提供される **chmod** コマンドは、NFS4 などの ACL の場合に、ファイルシステム・オブジェクトで S ビットを設定またはリセットするために使用できます。

管理アクセス権

オペレーティング・システムは、システム管理のために特別のアクセス権を提供します。

システム特権はユーザー ID とグループ ID に基づいています。 実効ユーザー ID またはグループ ID が 0 のユーザーは、特権があるユーザーとして認識されます。

0 の実効ユーザー ID を持つプロセスは root ユーザー・プロセスと呼ばれ、以下を実行することができます。

- すべてのオブジェクトの読み取りと書き込み。
- すべてのシステム機能のコール。
- **setuid-root** プログラムの実行による特定のサブシステム制御操作の実行。

システム管理者は、**su** コマンド特権と **setuid-root** プログラム特権の 2 つのタイプの特権を使用してシステムを管理できます。 **su** コマンドにより、呼び出したすべてのプログラムを root ユーザー・プロセスとして機能させることができます。 **su** コマンドはシステムを柔軟に管理できますが、安全性はあまり高くありません。

あるプログラムを **setuid-root** プログラムにするということは、プログラムが setuid ビットをセットされた root ユーザー所有のプログラムであるということを意味します。 **setuid-root** プログラムは、通常のユーザーがセキュリティーを損なわずに実施できる管理機能を提供します。特権はユーザーに直接認可されるのではなく、プログラム内にカプセル化されます。 必要なすべての管理機能を **setuid-root** プログラムにカプセル化するのは困難な場合もありますが、より水準の高いセキュリティーをシステム管理者に提供します。

アクセス許可

ユーザーがアカウントにログインすると (**login** または **su** コマンドを使用)、そのアカウントに割り当てられたユーザー ID およびグループ ID がユーザーのプロセスに関連付けられます。 これらの ID は、プロセスのアクセス権を決定します。

0 のユーザー ID を持つプロセスは、*root* ユーザー・プロセス と呼ばれます。これらのプロセスは、一般的にすべてのアクセス権を許可されています。 しかし、root ユーザー・プロセスがプログラムに対する実行許可を要求する場合、実行許可が最低 1 人のユーザーに付与されている場合のみアクセスが付与されます。

AIXC ACL の場合のアクセス許可

情報リソースの所有者は、アクセス権を管理する責任があります。リソースは、オブジェクトのモードに含まれる許可ビット により保護されています。 許可ビットは、オブジェクトの所有者、オブジェクトのグル

ープ、および others デフォルト・クラスに付与されるアクセス許可を定義します。 オペレーティング・システムは、個別に付与できる、3 つの異なるアクセスのモード (読み取り、書き込み、および実行) をサポートします。

ファイル、ディレクトリー、名前付きパイプ、およびデバイス (スペシャル・ファイル) に関しては、アクセスは次のように許可されます。

- ACL の各アクセス制御エントリー (ACE) ごとに、 ID リストがプロセスの ID と比較されます。 一致するものがあると、プロセスはそのエントリーに定義される許可と制限を受け取ります。 許可と制限の両方の論理和集合が、ACL で一致する各エントリーごとに計算されます。 要求プロセスが ACL のどのエントリーにも一致しない場合、デフォルト・エントリーの許可と制限を受け取ります。
- 要求済みアクセス・モードが許可されており (許可の和集合に含まれている)、制限されている (制限の和集合に含まれている) のではない場合、アクセスが付与されます。 そうでない場合、アクセスは拒否されます。

ACL の ID リストは、リスト中のすべての ID が、要求中のプロセスの実効 ID に対応するタイプに一致する場合、プロセスに一致します。 USER 型の ID は、プロセスの実効ユーザー ID に等しい場合に一致し、 GROUP 型の ID は、プロセスの実効グループ ID または補足グループ ID の 1 つに等しい場合に一致します。 例えば、次のような ID リストを持つ ACE の場合を考えてみます。

```
USER:fred, GROUP:philosophers, GROUP:software_programmer
```

これは、実効ユーザー ID *fred* および以下のグループ・セットを持つプロセスに一致します。

```
philosophers, philanthropists, software_programmer, doc_design
```

しかし、実効ユーザー ID *fred* および以下のグループ・セットを持つプロセスには一致しません。

```
philosophers, iconoclasts, hardware_developer, graphic_design
```

次のような ID リストを持つ ACE が両方のプロセスに一致することに注意してください。

```
USER:fred, GROUP:philosophers
```

つまり、ACE 機能の ID リストは、付与される指定されたアクセス用に保持する必要のある、条件のセットです。

これらのオブジェクトにおけるすべてのアクセス許可検査は、オブジェクトが最初にアクセスされたときのシステム・コール・レベルで行われます。 System V プロセス間通信 (SVIPC) オブジェクトはステートレスにアクセスされるため、すべてのアクセスで検査が実行されます。 ファイルシステム名を持つオブジェクトの場合、実オブジェクトの名前を解決できることが必要です。 名前は、相対的 (プロセスの作業ディレクトリーに対して)、または絶対的に (プロセスの root ディレクトリーに対して) 解決されます。 すべてのネーム・レゾリューションは、これらのディレクトリーの 1 つを検索することによって開始されます。

任意アクセス制御メカニズムにより、情報リソースの有効アクセス制御が許可され、情報の機密性および整合性を別個に保護します。 所有者が制御するアクセス制御メカニズムは、ユーザーがそれらを実行した場合と同じほどの有効性しかありません。 すべてのユーザーは、アクセス許可が付与および拒否される方法、および設定方法を理解しておく必要があります。

NFS4 ACL の場合のアクセス許可

WRITE_ACL 特権を有するユーザーはアクセス権を制御することができます。 情報リソースの所有者は、常に WRITE_ACL 特権を有しています。 NFS4 ACL 付きのファイルおよびディレクトリーの場合、アクセスは以下のようにして許可されます。

- ACE のリストが順番どおりに処理され、リクエスターと一致する「who」(すなわち、ID) を持つ ACE のみが処理対象と見なされます。 リクエスターの資格情報は、special who EVERYONE@ の ACE を処理している間は、検査されません。
- リクエスターのアクセスの全ビットが許可されるまで、各 ACE が処理されます。 1 つのビットがいったん許可されると、後続の ACE の処理においてビットは考慮されません。
- リクエスターのアクセスに対応したいずれかのビットが拒否されると、アクセスが拒否され、残りの ACE は処理されません。
- リクエスターのアクセスの全ビットが許可されることなく、処理すべき ACE が残っていない場合は、アクセスが拒否されます。

要求されたアクセスが ACE により拒否され、しかもリクエスト側ユーザーがスーパーユーザーまたは root の場合は、アクセスは一般的には許可されます。 READ_ACL、WRITE_ACL、READ_ATTRIBUTES、および WRITE_ATTRIBUTES については、オブジェクト所有者は常に許可されることに注意してください。 アクセス許可のアルゴリズムの詳細については、 139 ページの『NFS4 アクセス制御リスト』を参照してください。

アクセス制御リストのトラブルシューティング

以下の情報は、アクセス制御リスト (ACL) のトラブルシューティングに使用することができます。

オブジェクト障害アプリケーションにおける NFS4 アクセス制御リスト

問題のトラブルシューティングを行う場合、オブジェクト (ファイルやディレクトリーなど) に NFS4 ACL を設定することにより、戻りコードやトレース機能を使用することができます。いずれの手法も、**aclput** コマンドおよび **acledit** コマンドを使用して、問題の原因を見つけます。

トラブルシューティングのための戻りコードの使用

戻りコードを表示するには、**aclput** コマンドを実行した後、echo $? コマンドを使用します。 以下に戻りコードとその意味を示します。

22 (EINVAL、/usr/include/sys/errno.h で定義済み**)**

以下がこのコードに対する考えられる原因です。

- 4 つのフィールドのいずれかのに無効のテキスト・フォーマットがある。
- 入力 NFS4 ACL のサイズが 64 KB を超えている。
- ACE マスクが w (WRITE_DATA) に設定されているが p (APPEND_DATA) には設定されていない ACE、または ACE マスクが p (APPEND_DATA) に設定されているが w (WRITE_DATA) には設定されていない ACE が、既に少なくとも 1 つ存在するファイルに ACL が適用されている。
- ACE マスクが w (WRITE_DATA) に設定されているが p (APPEND_DATA) には設定されていない ACE、または ACE マスクが p (APPEND_DATA) に設定されているが w (WRITE_DATA) には設定されていない ACE が、既に少なくとも 1 つ存在し、さらに ACE フラグ fi (FILE_INHERIT) が存在するディレクトリーに、ACL が適用されている。
- OWNER@ が special **who** (**Identity**) として設定されている ACE が少なくとも 1 つあり、また、ACE マスク c (READ_ACL)、C (WRITE_ACL)、a (READ_ATTRIBUTE)、および A (WRITE_ATTRIBUTE) のうち 1 つ以上が ACE タイプ d により拒否されている。

124 (ENOTSUP、/usr/include/sys/errno.h で定義済み**)**

以下がこのコードに対する考えられる原因です。

- special who が、ACE のうちの 1 つで、3 つの値 (OWNER@、GROUP@、または EVERYONE@) のいずれでもなかった可能性がある。

- ACE タイプ u (AUDIT) または l (ALARM) の ACE が少なくとも 1 つ存在する。

13 (EACCES、/usr/include/sys/errno.h で定義済み)

以下がこのコードに対する考えられる原因です。

- NFS4 ACE を含んだ入力ファイルを読み取る許可を得ていない。
- ターゲット・オブジェクトの親ディレクトリーについて x (EXECUTE) アクセス権を持っていないため、ターゲット・オブジェクトの親ディレクトリーを検索することは許されていない。
- ACL の書き込みまたは変更を行う許可を得ていない可能性がある。 オブジェクトが既に NFS4 ACL と関連している場合は、ACE マスクについて C (WRITE_ACL) 特権を有することを確認してください。

トラブルシューティングのためのトレース機能の使用

問題の原因を見つけるために、トレース・レポートを生成することもできます。以下のシナリオは、NFS4 ACL を適用して、問題の原因を追求するためのトレースの使用法を示します。 例えば、以下の NFS4 ACL 付きのファイル、**/j2v2/file1** があるとします。

```
s:(EVERYONE@):  a       acC
```

そして、以下の ACL が **input_acl_file** 入力ファイルに含まれているとします。

```
s:(EVERYONE@):  a       rwxacC
```

トレース機能を使用してトラブルシューティングするための以下のステップを踏んでください。

1. 以下のコマンドを使用して、トレース機能、**aclput** および **trcrpt** を実行します。

   ```
   $ trace -j  478 -o trc.raw
   $->!aclput -i input_acl_file -t NFS4 /j2v2/file1
   $ ->quit
   $ trcrpt trc.raw > trc.rpt
   ```

2. トレース報告書を分析します。 ファイルやディレクトリーに ACL が適用される場合、 ACL の書き込みまたは変更のアクセスが検査された後に、ACL が適用されます。 ファイルには以下のような行が含まれます。

   ```
   478 xxx  xxx  ACL ENGINE: chk_access entry: type=NFS4 obj_mode=33587200 size=68 ops=16384 uid=100

   478 xxx  xxx  ACL ENGINE: chk_access exit: type=NFS4 rc=0 ops=16384 priv=0 against=0
   478 xxx  xxx  ACL ENGINE: set_acl entry: type=NFS4 ctl_flg=2 obj_mode=33587200 mode=0 size=48

   478 xxx  xxx  ACL ENGINE: validate_acl: type=NFS4 rc=22 ace_cnt=1 acl_len=48 size=12
   478 xxx  xxx  ACL ENGINE: set_acl exit: type=NFS4 rc=22 obj_mode=33587200 size=68 cmd=536878912
   ```

   chk_access exit を含んだ 2 行目は、ACL を書き込むためのアクセスが許可された (rc = 0) ことを示します。 validate_acl を含んだ 4 行目と set_acl exit を含んだ 5 行目は、ACL が正しく適用されなかったことを示します (rc=22 は **EINVAL** を示します)。 validate_acl を含んだ 4 行目は、ACE の 1 行目には問題がなかったことを示します (ace_cnt=1)。最初の ACE (s:(EVERYONE@): a rwxacC)) を参照する場合、アクセス・マスクとしての **p** は存在しません。 ACL を適用する場合は、**w** に追加して、**p** が必要です。

トラブルシューティング・アクセス否認

ファイルシステム操作 (例えば、read または write) は、NFS4 ACL に関連したオブジェクトに対しては失敗する可能性があります。 通常、エラー・メッセージが表示されますが、そのメッセージにはアクセス問題を判別するための十分な情報が含まれていない可能性があります。 アクセス問題を検出するために、トレース機能を使用することができます。例えば、以下の NFS4 ACL 付きのファイル、**/j2v2/file2** があるとします。

```
s:(EVERYONE@):  a       rwpx
```

次のコマンドは、「アクセス権否認」エラーを報告します。

```
ls -l /j2v2/file2
```

この問題をトラブルシューティングするには、以下のステップを踏んでください。

1. 以下のコマンドを使用して、トレース **ls -l /j2v2/file2** および trcrpt を実行します。

   ```
   $ trace -j  478 -o trc.raw
   $->!ls -l /j2v2/file2
   $ ->quit
   $ trcrpt trc.raw > trc.rpt
   ```

2. トレース報告書を分析します。 ファイルには以下のような行が含まれます。

   ```
   478  xxx  xxx   ACL ENGINE: chk_access entry: type=NFS4 obj_mode=33587711 size=68 ops=1024 uid=100
   478  xxx  xxx   ACL ENGINE: nfs4_chk_access_self: type=NFS4 aceN=1 aceCnt=1 req=128 deny=0
   478  xxx  xxx   ACL ENGINE: nfs4_mask_privcheck: type=NFS4 deny=128 priv=128
   478  xxx  xxx   ACL ENGINE: chk_access exit: type=NFS4 rc=13 ops=1024 priv=0 against=0
   ```

   3 行目は、READ_ATTRIBUTES のみの access mask = 128 (**0x80**) についてアクセスが拒否されたことを示します (**/usr/include/sys/acl.h** ファイルを参照)。

監査の概要

監査サブシステムによってシステム管理者はセキュリティー関連情報を記録することができ、それらを分析して、システム・セキュリティー・ポリシーの潜在的違反および実際の違反を検出することができます。

監査サブシステム

監査サブシステムには、検出、収集、および処理の各機能があります。

- 『監査イベント検出』
- 150 ページの『イベント情報の収集』
- 150 ページの『監査証跡情報の処理』

システム管理者は、これらの機能をそれぞれ構成できます。

監査イベント検出

イベント検出は、カーネル (監視プログラム状態コード) とトラステッド・プログラム (ユーザー状態コード) のどちらの場合も、トラステッド・コンピューティング・ベース (TCB) 全体に分散されています。 監査可能イベントとは、システムで発生する一切のセキュリティー関連のイベントです。セキュリティー関連の発生イベントとは、システムのセキュリティー状態の変化、システム・アクセス制御ポリシーまたはアカウンタビリティー・セキュリティー・ポリシーの違反の試みや実際の違反、あるいはその両方です。 監査可能イベントを検出したプログラムおよびカーネル・モジュールは、そのイベントをシステム監査ロガーに報告する責任があります。なお、このロガーはカーネルの一部として実行され、サブルーチン (トラステッド・プログラムの監査の場合) でアクセスすることも、カーネル・プロシージャー・コール (監視プログラム状態の監査の場合) 内でアクセスすることもできます。 報告される情報には、監査可能イベントの名前、そのイベントの成功または失敗、セキュリティー監査に関係する追加のイベント固有の情報があれば、その情報が含まれています。

イベント検出構成は、イベント検出をオンまたはオフにすることと、どのイベントがどのユーザーに対して監査されるかを指定することから成っています。 イベント検出を活動化する場合は、**audit** コマンドを使用して監査サブシステムを使用可能または使用禁止にします。 **/etc/security/audit/config** ファイルには、監査サブシステムが処理するイベントおよびユーザーが入っています。

イベント情報の収集

情報収集には、選択された監査可能イベントをロギングすることが含まれます。 この機能はカーネル監査ロガーによって実行され、このロガーには、システム・コールおよび監査可能イベントを記録するカーネル内プロシージャー・コール・インターフェースが用意されています。

監査ロガーは、完全な監査レコードを作る責任があります。このレコードは、すべてのイベントに共通する情報 (イベントの名前、責任のあるユーザー、イベントの時間と戻り状況など) を収めている監査ヘッダー、およびイベントに固有の情報を収めている監査証跡からなっています。監査ロガーはカーネル監査証跡に各連続レコードを追加し、これは次の 2 モードのどちらでも (または両方) 書き出すことができます。

BIN モード
: 監査証跡は交互のファイルに書き込まれるので、安全性が得られ、長期に保管しておくことができます。

STREAM モード
: 監査証跡は循環バッファーに書き込まれ、これは監査疑似デバイスから同期的に読み取られます。STREAM モードによると、即時の応答が得られます。

情報収集はフロントエンド (イベント記録) でも、バックエンド (証跡処理) でも構成することができます。 イベント記録はユーザー単位で選択できます。 各ユーザーは定義された監査イベントのセットを持ち、これらのイベントはその発生時に監査証跡に記録されるものです。 バックエンドでは、これらのモードは個別に構成可能であるので、管理者は特定の環境に最も適したバックエンド処理を採用することができます。 さらに、BIN モード監査は、証跡に使用可能なファイルシステム・スペースが少なくなりすぎた場合に、アラートを生成するように構成できます。

監査証跡情報の処理

オペレーティング・システムには、カーネル監査証跡を処理するためのオプションがいくつか用意されています。 BIN モード証跡は、監査証跡があるときにそれをストレージに保存する前に、出力用に圧縮するか、フィルターにかけるか、あるいはフォーマットするか、あるいはこれらを任意に組み合わせることができます。 圧縮はハフマン・エンコード (Huffman encoding) によって行われます。フィルター操作は、標準照会言語 (SQL) に似た監査レコード選択 (**auditselect** コマンドを使用して) で行われ、これによって監査証跡を選択的に見ることも、選択的に保存しておくこともできます。 監査証跡レコードのフォーマットを行うと、監査証跡を調べること、定期的セキュリティー・レポートを生成すること、および、監査証跡の印刷を行うことができます。

STREAM モード監査証跡はリアルタイムでモニターできるので、即時脅威モニター機能が得られます。これらのオプションの構成は別々のプログラムによって処理され、これらのプログラムはデーモン・プロセスとして呼び出して BIN または STREAM モードの監査証跡をフィルターにかけることができますが、フィルター・プログラムの中には、当然のことながら、一方のモードに適したものと他方のモードに適したものとがあります。

監査サブシステム構成

監査サブシステムにはグローバル状態変数があり、この変数は監査サブシステムがオンかどうかを示しています。 さらに、各プロセスにはローカル状態変数があり、この変数はそのプロセスに関する情報を監査サブシステムに記録させるかどうかを示しています。

これらの変数のどちらの場合も、イベントがトラステッド・コンピューティング・ベース (TCB) のモジュールおよびプログラムによって検出されるかどうかは、その変数によって判断されます。 ある特定のプロ

セスで TCB 監査をオフにすると、そのプロセスは独自の監査を行い、システム責任能力ポリシーをバイパスしません。 トラステッド・プログラムに自身を監査させるようにすると、情報の収集が効率化し、効果的になります。

監査サブシステム情報の収集

情報収集は、イベント選択とカーネル監査証跡モードを処理します。 これはカーネル・ルーチンによって行われ、このルーチンは、監査可能イベントを検出した TCB コンポーネントによって情報をログに記録するために使用されるインターフェースと、監査ロギング・ルーチンを制御するために監査サブシステムによって使用される構成インターフェースとを持っています。

監査ログ

監査可能イベントは、ユーザー状態と監視プログラム状態の次のインターフェースを使用してログに記録されます。 TCB のユーザー状態部分は **auditlog** または **auditwrite** サブルーチンを使用し、他方、TCB の監視プログラム状態部分はカーネル・プロシージャー・コールのセットを使用します。

各レコードごとに、監査イベント・ロガーは、イベント固有の情報の先頭に監査ヘッダーを付けます。 このヘッダーはどのユーザーとプロセスのイベントが監査されるのかを示すと共に、イベントの時間を示しています。 イベントを検出したコードはイベントのタイプと戻りコードまたは状況、およびオプションとして、追加のイベント固有情報 (イベント証跡) を戻します。 イベント固有情報はオブジェクト名 (例えば、アクセスが拒否されたファイルまたは失敗したログイン試行で使用された tty)、サブルーチン・パラメーター、およびその他の修正情報からなっています。

イベントは番号ではなく、シンボルで定義されます。このようにすると、イベント登録方式を使用しなくても、名前が衝突する可能性が少なくなります。 サブルーチンは監査可能であり、拡張可能カーネル定義には固定のスイッチド・バーチャル・サーキット (SVC) 番号がないので、番号でイベントを記録するのは困難です。 番号のマッピングは、カーネル・インターフェースが拡張または再定義されるたびに、訂正し、ログに記録することが必要になります。

監査レコード・フォーマット

監査レコードは共通ヘッダーと、そのあとに続く、そのレコードの監査イベントに特有の監査証跡とからなっています。 ヘッダーの構造は **/usr/include/sys/audit.h** ファイルに定義されています。監査証跡に入っている情報のフォーマットは各ベース・イベントに特有のものであり、**/etc/security/audit/events** ファイルに示されます。

監査ヘッダー内の情報は、その正確性を確保するためにロギング・ルーチンによって収集されるのが一般であるのに対し、監査証跡内の情報はイベントを検出したコードによって与えられます。 監査ロガーは監査証跡の構造も、そのセマンティクスも知りません。 例えば、**login** コマンドは失敗したログインを見つけると、そのログインが行われた端末と共にその特定イベントを記録し、**auditlog** サブルーチンを使用してそのレコードを監査証跡に書き込みます。監査ロガー・カーネル・コンポーネントはサブジェクト特有の情報 (ユーザー ID、プロセス ID、時間) をヘッダーに記録し、これを他の情報に追加します。呼び出し元はイベント名と結果フィールドだけをヘッダーに入れます。

監査ロガー構成

監査ロガーは完全な監査レコードを作る責任があります。 ログに記録させたい監査イベントはユーザーが選択しなければなりません。

監査イベントの選択

監査イベントの選択には次のタイプがあります。

プロセスごとの監査
: プロセス・イベントを効率的に選択するために、システム管理者は監査クラスを定義できます。 監査クラスはシステム内のベース監査イベントのサブセットです。 監査クラスを使用すると、ベース監査イベントを都合よく論理的にグループ化することができます。

  システムにいる各ユーザーごとに、システム管理者は監査クラスのセットを定義します。そのユーザーのベース・イベントが記録できるかどうかはその監査クラスで決まります。 ユーザーによって実行される各プロセスには、その監査クラスのタグが付けられます。

オブジェクト単位の監査
: オペレーティング・システムはオブジェクト・アクセスの監査を名前別に行います。つまり、オブジェクト (通常、ファイル) 別に監査を行います。 オブジェクト監査を名前別に行うと、少数の関連オブジェクトを監査するためにすべてのオブジェクト・アクセスを対象にしないで済みます。 さらに、監査モードが指定できるので、指定したモードのアクセス (read/write/execute) だけが記録されます。

カーネル監査証跡モード

カーネル・ロギングを BIN または STREAM モードに設定すると、カーネル監査証跡をどこに書き込むかを定義することができます。 BIN モードを使用するときは、レコードがそこに追加される少なくとも 1 つのファイル・ディスクリプターをカーネル監査ロガーに与えておかなければなりません (監査始動前に)。

BIN モードを使用すると、監査レコードは交互のファイルに書き込まれます。 監査始動時に、カーネルに 2 つのファイル・ディスクリプターと望ましい最大ビン・サイズが渡されます。 カーネルは呼び出しプロセスを中断し、監査レコードを最初のファイル・ディスクリプターに書き込むことを始めます。 最初のビンのサイズが最大ビン・サイズまで達したとき、 2 番目のファイル・ディスクリプターが有効であれば、カーネルは 2 番目のビンに切り替わり、呼び出しプロセスを再活動化します。 カーネルは別の有効なファイル・ディスクリプターで再呼び出しされるまで 2 番目のビンに書き込むことを続けます。 その時点で 2 番目のビンがいっぱいになっていれば、カーネルは最初のビンに戻るように切り替わり、呼び出しプロセスに即時に戻ります。 そうでなければ、呼び出しプロセスは中断され、カーネルは 2 番目のビンがいっぱいになるまでレコードを書き込むことを続けます。 処理は監査がオフにされるまで以上のように続けられます。 以下の図は、監査 BIN モードのイラストです。

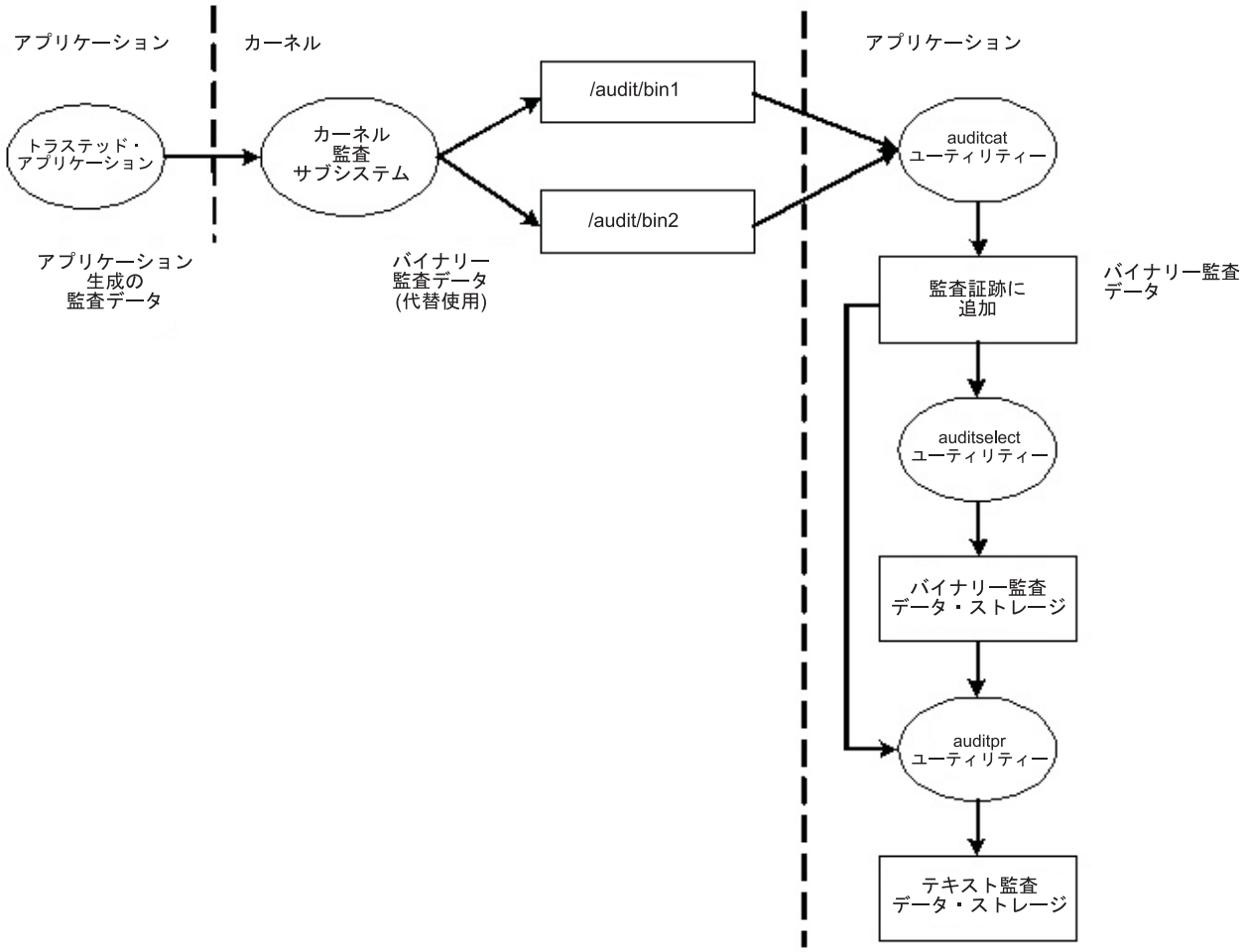


図 *1.* 監査 *BIN* モードのプロセス： 監査 BIN モードのプロセスを示すイラスト。

監査レコードの処理中に、監査サブシステムに常に書き込み先があるように、代替ビン・メカニズムが使用されます。 監査サブシステムが他のビンに切り替えると、最初のビンは空になり、その内容は **trace** ファイルに入れられます。 再びビンを切り替える時間になると、最初のビンが使用可能になります。 そして、データの保管および分析が、データ生成から切り離されます。 通常、カーネルがその時点で書き込んでいないビンからデータを読み取るため、**auditcat** プログラムが使用されます。 システムの監査証跡 (**auditcat** プログラムの出力) 用のスペースが決して使い果たされないようにするため、*freespace* パラメーターを **/etc/security/audit/config** ファイルで指定できます。システムのスペースがここで指定される 512 バイト・ブロックより少ないと、 syslog メッセージが生成されます。

監査が使用可能である場合、**/etc/security/audit/config** 中の `start` スタンザにある *binmode* パラメーターを `panic` に設定する必要があります。 `bin` スタンザの *freespace* パラメーターは、少なくとも監査証跡のストレージ専用のディスク・スペースの 25% に等しい値に構成する必要があります。 *bytethreshold* および *binsize* パラメーターは、それぞれ 65536 バイトに設定します。

STREAM モードでは、カーネルはレコードを循環バッファーに書き込みます。 カーネルがバッファーの終わりまで達したときは、先頭に戻るだけです。 プロセスは、`/dev/audit` と呼ばれる疑似デバイスを通して情報を読み取ります。 プロセスがこのデバイスをオープンすると、そのプロセス用にチャネルが作成されます。 オプションとして、そのチャネル上で読み取られるイベントは監査クラスのリストとして指定できます。以下の図は、監査 STREAM モードのイラストです。

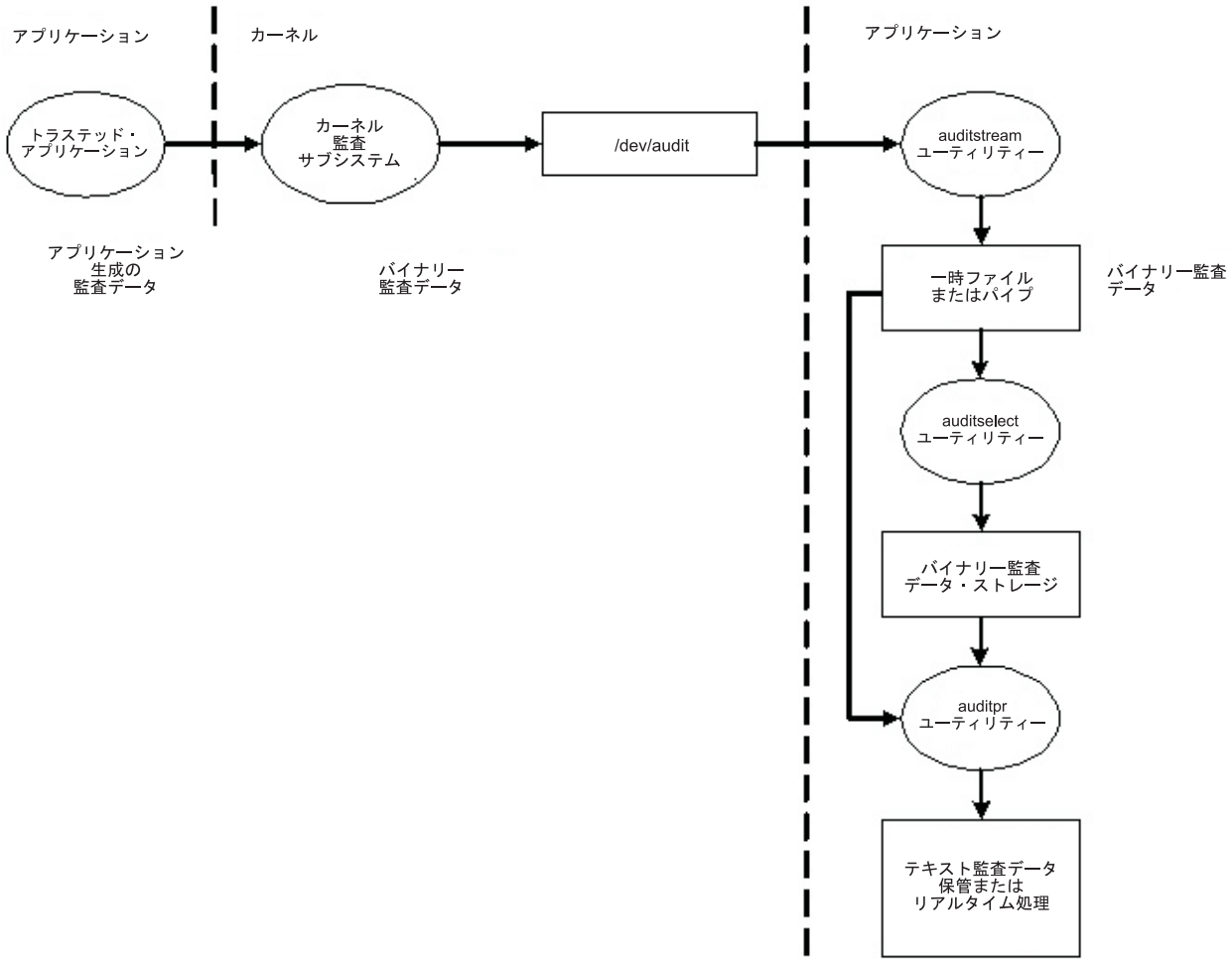


図 2. 監査 *STREAM* モードのプロセス： 監査 STREAM モードのプロセスを示すイラスト。

STREAM モードの主目的は、監査証跡をタイムリーに読み取れるようにすることであり、これは脅威モニターをリアルタイムで行う場合に望ましいモードです。 もう 1 つの使い方は即時に書き込まれる証跡を作成し、証跡をある種の書き込み可能メディアに保管すると起こるような、監査証跡をみだりに変更するといった事態を防止することです。

STREAM モードを使用するさらに別の方法は、リモート・システム上に監査情報を保管するプログラムに監査ストリームを書き込むことで、これによって、中央で近似時間処理が実行でき、同時に発信元ホストで悪用から監査情報を保護することにもなります。

監査レコードの処理

auditselect、**auditpr**、および **auditmerge** コマンドは、 BIN または STREAM モードの監査レコードの処理に使用可能です。 どちらのユーティリティーも、パイプで容易に使用できるようにフィルターとして操作し、特に STREAM モードの監査に役立ちます。

auditselect

ステートメントのような SQL を持つ特定の監査レコードのみを選択するのに使用できます。例えば、ユーザー *afx* が生成する **exec()** イベントだけを選択するには、次のように入力します。

```
auditselect -e "login==afx && event==PROC_Execute"
```

auditpr

バイナリー監査レコードを人間が読み取れる形式に変換します。 表示される情報量は、コマンド・ラインで指定されるフラグに依存します。 使用可能な情報すべてを入手するには、**auditpr** を次のように実行してください。

```
auditpr -v -hhelrtRpPTc
```

-v フラグが指定されると、カーネルがすべてのイベントに送達する標準監査情報に加え、イベントに特定の文字列 (**/etc/security/audit/events** ファイルを参照) である監査証跡が表示されます。

auditmerge

バイナリー監査証跡をマージするのに使用されます。 これは、結合する必要のある複数のシステムからの監査証跡がある場合に特に役立ちます。 **auditmerge** コマンドは、コマンド・ラインで証跡の名前を取り出し、マージされたバイナリー証跡を標準出力するので、それを読み取れるように、**auditpr** コマンドを使用する必要があります。 例えば、**auditmerge** および **auditptr** コマンドは次のように実行できます。

```
auditmerge trail.system1 trail.system2 | auditpr -v -hhelrRtpc
```

高速セキュリティー検査のための監査サブシステムの使用**:**

監査サブシステムをセットアップしないで単一の疑わしいプログラムをモニターするには、**watch** コマンドを使用できます。 このコマンドは、要求されたイベント、または指定したプログラムが生成するすべてのイベントを記録します。

例えば、**vi /etc/hosts** の実行時にすべての **FILE_Open** イベントを表示するには、次のように入力します。

```
watch -eFILE_Open -o /tmp/vi.watch vi /etc/hosts
```

/tmp/vi.watch ファイルは、該当のエディター・セッションのすべての **FILE_Open** イベントを表示します。

イベント選択

イベント選択は、不十分な詳細と過剰な詳細とのバランスを維持しなければなりません。

システムでの監査可能イベントのセットでは、実際に監査可能な発生イベントと、得られる監査の細分性を定義します。 監査可能イベントは前述したように、システムでのセキュリティー関連イベントを網羅していなければなりません。監査可能イベントを定義するとき使用する詳細度は、管理者が選択情報を理解することを困難にする不十分な詳細度と、過剰な情報収集が行われることになる極度の詳細度との間の平衡を取る必要があります。 イベントの定義は検出イベントの類似性を利用して行います。 ここでの説明の便宜上、検出イベント とは、監査可能イベントの任意の単一インスタンスのことです。例えば、あるイベントは各所で検出することができます。 基礎となる原理は、類似のセキュリティー属性をもつ検出イベントは同じ監査可能イベントとして選択されることです。 以下のリストはセキュリティー・ポリシー・イベントの種別を示しています。

- サブジェクト・イベント
  - プロセス作成
  - プロセス削除
  - サブジェクト・セキュリティー属性の設定: ユーザー ID、グループ ID
  - プロセス・グループ、制御端末
- オブジェクト・イベント
  - オブジェクト作成

  - オブジェクト削除
  - オブジェクトのオープン (オブジェクトとしてのプロセスを含む)
  - オブジェクトのクローズ (オブジェクトとしてのプロセスを含む)
  - オブジェクト・セキュリティー属性の設定: 所有者、グループ、ACL
- インポート/エクスポート・イベント
  - オブジェクトのインポートまたはエクスポート
- アカウンタビリティー・イベント
  - ユーザーの追加、パスワード・データベース内のユーザー属性の変更
  - グループの追加、グループ・データベース内のグループ属性の変更
  - ユーザー・ログイン
  - ユーザー・ログオフ
  - ユーザー認証情報の変更
  - トラステッド・パス端末の構成
  - 認証構成
  - 管理の監査: イベントと監査証跡の選択、スイッチ・オンまたはオフ、ユーザー監査クラスの定義
- 一般システム管理イベント
  - 特権の使用
  - ファイルシステム構成
  - デバイスの定義と構成
  - システム構成パラメーターの定義
  - 通常のシステム IPL およびシャットダウン
  - RAS 構成
  - 他のシステム構成
  - 監査サブシステムの開始
  - 監査サブシステムの停止
  - 監査サブシステムの照会
  - 監査サブシステムのリセット
- セキュリティー違反 (発生の可能性のある)
  - アクセス権の拒否
  - 特権の障害
  - 診断により検出される障害とシステム・エラー
  - TCB の変更の試み

監査イベント**:**

監査イベント とは、システムでのセキュリティー関連のすべての発生事象のことです。セキュリティー関連の発生イベントには、システムのセキュリティー状態の変化、システム・アクセス制御ポリシーまたはアカウンタビリティー・セキュリティー・ポリシー (あるいはその両方) の違反の試行や実際の違反などがあります。監査イベントを検出したプログラムおよびカーネル・モジュールは、それらのイベントをシステム監査ロガーに報告します。この監査ロガーはカーネルの一部として実行され、サブルーチンを使用して (トラステッド・プログラムの監査の場合) アクセスするか、またはカーネル・プロシージャー・コール内 (監

視プログラム状態の監査の場合) でアクセスすることができます。監査イベントに報告される情報には、イベントの名前、そのイベントの成功または失敗に関する情報、およびセキュリティー監査に関係するすべての追加のイベント固有情報が含まれます。

あるアクティビティーを監査するためには、どのコマンドまたはプロセスが監査イベントを開始するのかを指定し、そのイベントがシステムの **/etc/security/audit/events** ファイルにリストされていることを確かめなければなりません。 類似のイベントを監査クラスにまとめると、ユーザーへの監査イベントの割り当てが容易になります。 これらの監査クラスは `/etc/security/audit/config` ファイルの classes スタンザに定義されます。

以下の表では、AIX オペレーション・システムで発生する、一般的に使用されるいくつかの監査イベントをリストしています。

表 *11.* 監査イベント

ユーザーまたはシステム呼び出し	監査イベント	説明
fork	PROC_Create	プロセスが作成されたことを示します。
exit	PROC_Delete	呼び出しプロセスが終了したことを示します。
exec	PROC_Execute	新規プログラムを実行します。
setuidx	PROC_RealUID	プロセスのユーザー ID を設定します。
	PROC_AuditID	
	PROC_SetUserIDs	
setgidx	PROC_RealGID	プロセス・グループ ID を設定します。
accessx	FILE_Accessx	ファイルのアクセス可能性を判別します。
statacl	FILE_StatAcl	ファイルのアクセス制御情報を取得します。
revoke	FILE_Revoke	すべてのプロセスによるファイルへのアクセスを取り消します。
frevoke	FILE_Frevoke	他のプロセスによるファイルへのアクセスを取り消します。
usrinfo	PROC_Environ	ユーザー情報データの一部を変更します。
sigaction	PROC_SetSignal	このサブルーチンを発行したプロセスに特定のシグナルが送信されるときに実行するアクションを指定します。
setrlimit	PROC_Limits	最大システム・リソースの消費量を制御します。
nice	PROC_SetPri	`nice` 関数の使用を指定します。
setpri	PROC_Setpri	プロセスの固定優先度を設定します。
setpriv	PROC_Privilege	プロセスの 1 つ以上の特権ベクトルを変更します。
settimer	PROC_Settimer	指定された全システム・タイマーの現行値を設定します。
adjtime	PROC_Adjtime	システム・クロックを変更します。
ptrace	PROC_Debug	別のプロセスの実行をトレースします。
kill	PROC_Kill	シグナルを 1 プロセスまたはプロセスのグループに送信します。
setpgid	PROC_setpgid	プロセス・グループ ID を設定します。

表 11. 監査イベント (続き)

ユーザーまたはシステム呼び出し	監査イベント	説明
ld_loadmodule	PROC_Load	新規オブジェクト・モジュールをプロセス・アドレス・スペースにロードします。
	PROC_LoadError	オブジェクトのロードが失敗したことを示します。
setgroups	PROC_SetGroups	プロセス並行グループ・セットを変更します。
sysconfig	PROC_Sysconfig	カーネルまたはシステム構成に対するアクションを収集します。
audit	AUD_It	監査操作を開始および停止します。これは、監査状況も照会します。
auditbin	AUD_Bin_Def	`auditbin` システム呼び出しを変更します。
auditevents	AUD_Events	イベントを変更します。
auditobj	AUD_Objects	`auditobj` システム呼び出しを変更します。
auditproc	AUD_Proc	プロセスの監査状態の取得または設定を行います。
acct	ACCT_Disable	システム・アカウンティングを無効にします。
	ACCT_Enable	システム・アカウンティングを有効にします。
open および create	FILE_Open	**open** サブルーチンを呼び出します。
read	FILE_Read	ファイル記述子からデータを読み取ります。
write	FILE_Write	ファイル記述子にデータを書き込みます。
close	FILE_Close	オープン・ファイル記述子を閉じます。
link	FILE_Link	ファイル・システム・オブジェクトの新規ディレクトリー・エントリーを作成します。
unlink	FILE_Unlink	ファイル・システム・オブジェクトを除去します。
rename	FILE_Rename	ファイル・システム・オブジェクトの名前を変更します。
chown	FILE_Owner	ファイル所有権を変更します。
chmod	FILE_Mode	ファイル・モードを変更します。
fchmod	FILE_Fchmod	ファイル記述子のファイル許可を変更します。
fchown	FILE_Fchown	ファイル記述子の所有権を変更します。
truncate	FILE_Truncate	正規ファイルまたは共有メモリー・オブジェクトの長さを変更します。
symlink	FILE_Symlink	シンボリック・リンクを作成します。
pipe	FILE_Pipe	名前なしパイプを作成します。
mknod	FILE_Mknod	デバイスの特殊ファイルまたは先入れ先出し法 (FIFO) 特殊ファイルを作成します。
fcntl	FILE_Dupfd	ファイル記述子を複写します。
fscntl	FS_Extend	ファイル・システムを拡張します。
mount	FS_Mount	ファイル・システムを名前付きディレクトリーに接続します。

表 11. 監査イベント (続き)

ユーザーまたはシステム呼び出し	監査イベント	説明
umount	FS_Umount	マウントされたファイル・システムを切断します。
chacl	FILE_Acl	ファイルのアクセス制御リスト (ACL) を変更します。
fchacl	FILE_Facl	ファイル記述子の ACL を変更します。
chpriv	FILE_Privilege	ファイル・パス名の特権制御リスト (PCL) を設定します。
	FILE_Chpriv	PCL を変更します。
	FILE_Fchpriv	ファイル記述子の PCL を変更します。
chdir	FS_Chdir	現行作業ディレクトリーを変更します。
fchdir	FS_Fchdir	ファイル記述子を使用して現行作業ディレクトリーを変更します。
chroot	FS_Chroot	現行プロセスのルート・ディレクトリー (/) の意味を変更します。
rmdir	FS_Rmdir	ディレクトリー・オブジェクトを除去します。
mkdir	FS_Mkdir	ディレクトリーを作成します。
utimes	FILE_Utimes	**utimes** サブルーチンを呼び出します。
stat	FILE_Stat	**stat** サブルーチンを呼び出します。
msgget	MSG_Create	メッセージ・キューを作成します。
msgrcv	MSG_Read	メッセージをメッセージ・キューから受信します。
msgsnd	MSG_Write	メッセージをメッセージ・キューに送信します。
msgctl	MSG_Delete	メッセージ・キューを除去します。
	MSG_Owner	メッセージ・キューの所有権およびアクセス権を変更します。
	MSG_Mode	メッセージ・キューのアクセス権を照会します。
semget	SEM_Create	セマフォー・セットを作成します。
semop	SEM_Op	1 つ以上のセマフォーを増分または減分します。
semctl	SEM_Delete	セマフォー・セットを削除します。
	SEM_Owner	セマフォー・セットの所有権およびアクセス権を変更します。
	SEM_Mode	セマフォー・セットのアクセス権を照会します。
shmget	SHM_Create	新規共有メモリー・セグメントを作成します。
shmat	SHM_Open	**shmat** サブルーチンを、**Open** オプションを使用して呼び出します。
shmat	SHM_Detach	**shmat** サブルーチンを、**Detach** オプションを使用して呼び出します。
shmctl	SHM_Close	共有メモリー・セグメントを閉じます。
	SHM_Owner	共有メモリー・セグメントの所有権およびアクセス権を変更します。
	SHM_Mode	共有メモリー・セグメントのアクセス権を照会します。

表 11. 監査イベント (続き)

ユーザーまたはシステム呼び出し	監査イベント	説明
tcpip ユーザー・レベル	TCPIP_config	TCP/IP インターフェースへの変更をログに記録します。
	TCPIP_host_id	システム・ホスト名の変更の試行をログに記録します。
	TCPIP_route	ルーティング・テーブルへの変更をログに記録します。
	TCPIP_connect	**connect** サブルーチンを呼び出します。
	TCPIP_data_out	データを送信しました。
	TCPIP_data_in	データを受信しました。
	TCPIP_set_time	ネットワーク経由でのシステム時刻の変更の試行をログに記録します。
tcpip カーネル・レベル	TCP_ksocket	カーネル TCP/IP カーネル・サービスを呼び出します。
	TCP_ksocketpair	
	TCP_kclose	
	TCP_ksetopt	
	TCP_kbind	
	TCP_klisten	
	TCP_kconnect	
	TCP_kaccept	
	TCP_kshutdown	
	TCP_ksend	
	TCP_kreceive	
tsm	USER_Login	ユーザーをシステムにログインさせます。
	PORT_Locked	無効なログイン試行によりポートがロックされていることを示します。
	TERM_Logout	ユーザーをシステムからログアウトさせます。
rlogind または telnetd	USER_Exit	ユーザーがログアウトしたことを示します。
usrck	USER_Check	ユーザー定義の正確度を検証します。
	USRCK_Error	
logout	USER_Logout	ポート上のすべてのプロセスを停止します。
chsec	PORT_Change	ポート属性値の変更を示します。
chuser	USER_Change	ユーザー属性を変更します。
rmuser	USER_Remove	ユーザーを除去します。
mkuser	USER_Create	ユーザーを作成します。
setgroups	USER_SetGroups	現行プロセスの補足グループ ID を設定します。
setsenv	USER_SetEnv	環境変数を設定します。
su	USER_SU	セッションと関連付けられたユーザー ID を変更します。
grpck	GROUP_User	存在しないユーザーをグループから除去します。
	GROUP_Adms	存在しない管理ユーザーをグループから除去します。
chgroup	GROUP_Change	グループ属性を変更します。

表 11. 監査イベント (続き)

ユーザーまたはシステム呼び出し	監査イベント	説明
mkgroup	GROUP_Create	グループを作成します。
rmgroup	GROUP_Remove	グループを除去します。
passwd	PASSWORD_Change	ユーザー・パスワードを変更します。
pwdadm	PASSWORD_Flags	管理者パスワードを変更します。
pwdck	PASSWORD_Check	ローカル認証情報の正確度を検査します。
	PASWORD_Ckerr	
startsrc	SRC_Start	システム・リソース・コントローラーを開始します。
stopsrc	SRC_Stop	システム・リソース・コントローラーを停止します。
addssys	SRC_Addssys	SRCsubsys 定義をサブシステム・オブジェクト・クラスに追加します。
chssys	SRC_Chssys	サブシステム・オブジェクト・クラス内のサブシステム定義を変更します。
addserver	SRC_Addserver	サブサーバー定義をサブサーバー・オブジェクト・クラスに追加します。
chserver	SRC_Chserver	サブサーバー・オブジェクト・クラス内のサブサーバー定義を変更します。
rmsys	SRC_Delssys	サブシステム定義をサブシステム・オブジェクト・クラスから除去します。
rmserver	SRC_Delserver	サブサーバー定義を Subserver タイプ・オブジェクト・クラスから除去します。
enq	ENQUE_admin	ファイルをキューに入れます。
qdaemon	ENQUE_exec	キューに入れられたジョブをスケジュールします。
sendmail	SENDMAIL_Config	ローカル送達またはネットワーク送達のためにメールの経路指定を行います。
	SENDMAIL_ToFile	
at	AT_JobAdd	**at** コマンドを使用して実行をスケジュールしたコマンドを追加または除去します。
	At_JobRemove	
cron	CRON_JobRemove	**cron** コマンドを使用して実行をスケジュールしたコマンドを追加または除去します。
	CRON_JobAdd	
	CRON_Start	**cron** ジョブの開始を示します。
	CRON_Finish	**cron** ジョブの終了を示します。
nvload	NVRAM_Config	不揮発性ランダム・アクセス・メモリー (NVRAM) へのアクセスを指定します。
cfgmgr	DEV_Configure	デバイスを構成します。
chdev および mkdev	DEV_Change	デバイスでの変更を示します。
mkdev	DEV_Create	デバイスが作成されたことを示します。
	DEV_Start	デバイスが開始したことを示します。
installp	INSTALLP_Inst	互換性のあるインストール・パッケージ内の使用可能なソフトウェア・プロダクトをインストールします。
	INSTALLP_Exec	
rmdev	DEV_Stop	デバイスが停止したことを示します。
	DEV_Unconfigure	デバイスが構成されていないことを示します。
	DEV_Remove	デバイスが除去されたことを示します。

表 11. 監査イベント (続き)

ユーザーまたはシステム呼び出し	監査イベント	説明
lchangelv、lextendlv、および lreducelv	LVM_ChangeLV	論理ボリュームが変更されたことを示します。
lchangepv、ldeletepv、および linstallpv	LVM_ChangeVG	ボリューム・グループが変更されたことを示します。
lcreatelv	LVM_CreateLV	論理ボリュームがシステムに追加されていることを示します。
lcreatevg	LVM_CreateVG	ボリューム・グループがシステム内に作成されたことを示します。
ldeletepv	LVM_DeleteVG	ボリューム・グループがシステムから除去されたことを示します。
rmlv	LVM_DeleteLV	論理ボリュームがシステムから除去されたことを示します。
lvaryoffvg	LVM_VaryoffVG	ボリューム・グループを非アクティブにします。
lvaryonvg	LVM_VaryonVG	ボリューム・グループをアクティブにします。
論理ボリュームの操作	LVM_AddLV	論理ボリュームを既存のボリューム・グループに追加します。
	LVM_KDeleteLV	既存のボリューム・グループから論理ボリュームを除去します。
	LVM_ExtendLV	ボリューム・グループから割り振り解除されている物理区画を追加することによって、論理ボリュームのサイズを拡大します。
	LVM_ReduceLV	論理ボリュームのサイズを減らします。
	LVM_KChangeLV	既存の論理ボリュームを変更します。
	LVM_AvoidLV	論理ボリュームが特定の操作を実行することを許可しません。
物理ボリュームの操作	LVM_MissingPV	欠落している物理ボリュームを既存のボリューム・グループに追加します。
	LVM_AddPV	物理ボリュームを既存のボリューム・グループに追加します。
	LVM_AddMissPV	欠落している物理ボリュームを既存のボリューム・グループに追加します。
	LVM_DeletePV	既存のボリューム・グループから物理ボリュームを削除します。
	LVM_RemovePV	既存のボリューム・グループから物理ボリュームを除去します。
	LVM_AddVGSA	ボリューム・グループ状況領域 (VGSA) を既存の物理ボリュームに追加します。
	LVM_DeleteVGSA	VGSA を既存の物理ボリュームから除去します。
ボリューム・グループの操作	LVM_SetupVG	論理ボリュームを定義し、VGSA およびミラー書き込み整合性キャッシュ (MWCC) に関する情報を指定することで、ボリューム・グループをセットアップします。
	LVM_DefineVG	ボリューム・グループをカーネルに対して定義します。
	LVM_KDeleteVG	ボリューム・グループをカーネルから削除します。

表 11. 監査イベント (続き)

ユーザーまたはシステム呼び出し	監査イベント	説明
バックアップ操作とリストア操作	BACKUP_Export	バックアップ操作の進行状況を収集します。
	RESTORE_Import	リストア操作の進行状況を収集します。
shell	USER_Shell	ユーザーの tty 情報を収集します。
reboot	USER_Reboot	システム・リブートのイベントを収集します。
	PROC_Reboot	プロセス・リブートのイベントを収集します。 **reboot** サブルーチンは、システムを再始動するか、またはシステムで初期プログラム・ロード (IPL) 操作を繰り返します。

監査のセットアップ

この手順は、監査サブシステムのセットアップ方法を示したものです。特定の情報については、それらのステップで示されている構成ファイルを参照してください。

1. 監査するシステム・アクティビティー (イベント) を、**/etc/security/audit/events** ファイル内のリストから選択します。アプリケーションまたはカーネル・エクステンションに新しい監査イベントを追加した場合、新しいイベントを追加するためにファイルを編集する必要があります。
   - アプリケーション・プログラムに (**auditwrite** または **auditlog** サブルーチンを使用して) またはカーネル・エクステンションに (**audit_svcstart**、**audit_svcbcopy**、および **audit_svcfinis** カーネル・サービスを使用して) あるイベントを記録するコードを組み込んだ場合は、そのイベントをこのファイルに追加します。
   - 新しいすべての監査イベントに関するフォーマット指示が、**/etc/security/audit/events** ファイルに含められているか確認します。これらの指定により、監査レコードのフォーマット時に、**auditpr** コマンドで監査証跡を書き込むことができます。
2. 選択した監査イベントを、監査クラス という類似した項目のセットにグループ化します。 これらの監査クラスを、**/etc/security/audit/config** ファイル内の classes スタンザに定義します。
3. 以下に示すように、個人ユーザーに監査クラスを割り当て、監査しようとするファイル (オブジェクト) に監査イベントを割り当てます。
   - 個人ユーザーに監査クラスを割り当てるには、**/etc/security/audit/config** ファイルの users スタンザに 1 行追加します。ユーザーに監査クラスを割り当てるには、**chuser** コマンドを使用できます。
   - オブジェクト (データまたは実行可能ファイル) へ監査イベントを割り当てるには、そのファイルのスタンザを **/etc/security/audit/objects** ファイルに追加します。
   - **/usr/lib/security/mkuser.default** ファイルを編集して、新規ユーザーにデフォルトの監査クラスを指定することもできます。 このファイルには、新規ユーザー ID を生成する際に使用されるユーザー属性が保持されます。 例えば、すべての新規ユーザー ID に general 監査クラスを使用するには、以下のようにします。

```
user:
    auditclasses = general
    pgrp = staff
    groups = staff
    shell = /usr/bin/ksh
    home = /home/$USER
```

すべての監査イベントを取得するには、ALL クラスを指定します。 使用頻度が中程度のシステムでこれを行った場合でも、大量のデータが生成されます。 記録されるイベントの数を制限する方が、多くの場合、より実際的です。

4. **/etc/security/audit/config** ファイルで、 BIN 収集か STREAM 収集 (またはその両方) を使用して、必要なデータ収集のタイプを構成します。 監査データに個別のファイルシステムを使用することにより、監査データが、ファイル・スペースに関する他の情報と競合しないようにしてください。 これにより、十分のスペースが監査データ用に確保されます。 以下のように、データ収集のタイプを構成します。
   - BIN 収集を構成する場合:
     a. start スタンザで binmode = on に設定して、BIN モード収集を使用可能にします。
     b. ビンと証跡を構成するために binmode スタンザを編集し、BIN モードのバックエンド処理コマンドを含んでいるファイルのパスを指定します。バックエンド・コマンドのデフォルト・ファイルは **/etc/security/audit/bincmds** ファイルです。
     c. 監査ビンの大きさが必要を満たすのに十分かどうか確認し、それに応じて *freespace* パラメーターを設定して、ファイルシステムがいっぱいになったときにアラートが出されるようにします。
     d. 監査パイプで監査ビンを処理するシェル・コマンドを、**/etc/security/audit/bincmds** ファイルに入れます。
   - STREAM 収集を構成する場合:
     a. start スタンザで streammode = on に設定して、STREAM モード収集を使用可能にします。
     b. streammode スタンザを編集し、 streammode 処理コマンドを含んでいるファイルへのパスを指定するようにします。 この情報を収めているデフォルト・ファイルは、**/etc/security/audit/streamcmds** ファイルです。
     c. 監査パイプでストリーム・レコードを処理するシェル・コマンドを **/etc/security/audit/streamcmds** ファイルに入れます。
5. 構成ファイルに必要な変更を終えると、**audit start** コマンドを使用して監査サブシステムを使用できる状態になります。 これにより、値 1 で **AUD_It** イベントが生成されます。
6. **audit query** コマンドを使用して、どのイベントおよびオブジェクトが監査されているかを調べます。これにより、値 2 で **AUD_It** イベントが生成されます。
7. **audit shutdown** コマンド・オプションを使用して、監査サブシステムを再び非活動化します。 これにより、値 4 で **AUD_It** イベントが生成されます。

一般的な監査ログの生成**:**

以下は、一般的な監査ログ生成の例です。

この例では、システム管理者が監査サブシステムを使用して、大規模なマルチユーザー・サーバー・システムをモニターするという状況を想定しています。 IDS への直接的な統合は実行されず、すべての監査レコードを手動で検査して、不正がないかどうか調べます。 生成されるデータの量を管理可能なサイズにとどめるために、少数の不可欠な監査イベントだけが記録されます。

イベント検出と見なされる監査イベントを以下に示します。

FILE_Write
構成ファイルへのファイル書き込みに関して知る必要があるため、このイベントは **/etc** ツリー内のすべてのファイルで使用されます。

PROC_SetUserIDs
ユーザー ID のすべての変更

AUD_Bin_Def
監査ビン構成

USER_SU
su コマンド

PASSWORD_Change
passwd コマンド

AUD_Lost_Rec
レコードが失われた場合の通知

CRON_JobAdd
新規 cron ジョブ

AT_JobAdd
新規 at ジョブ

USER_Login
すべてのログイン

PORT_Locked
無効な試行が多すぎたことによる、端末でのすべてのロック

一般的な監査ログを生成する方法の例を、以下に示します。

1. 変更をモニターする重要なファイルのリストをセットアップします。例えば、**/etc** 内のすべてのファイルをモニターする場合、以下のように、**objects** ファイル内で **FILE_Write** イベント用にそれらを構成します。

   ```
   find /etc -type f | awk '{printf("%s:¥n¥tw = FILE_Write¥n¥n",$1)}' >> /etc/security/audit/objects
   ```

2. **auditcat** コマンドを使用して、BIN モードの監査をセットアップします。 **/etc/security/audit/bincmds** ファイルは次のようになります。

   ```
   /usr/sbin/auditcat -p -o $trail $bin
   ```

3. **/etc/security/audit/config** ファイルを編集し、必要なイベントのクラスを追加します。 既存のユーザーをすべてリストし、それらに custom クラスを指定します。

   ```
   start:
           binmode = on
           streammode = off

   bin:
           cmds = /etc/security/audit/bincmds
           trail = /audit/trail
           bin1 = /audit/bin1
           bin2 = /audit/bin2
           binsize = 100000
           freespace = 100000

   classes:
           custom = FILE_Write,PROC_SetUser,AUD_Bin_Def,AUD_Lost_Rec,USER_SU, ¥
                    PASSWORD_Change,CRON_JobAdd,AT_JobAdd,USER_Login,PORT_Locked

   users:
           root = custom
           afx = custom
           ...
   ```

4. 新しい ID で正しい監査呼び出しが自動的に関連付けられるように、以下のように、custom 監査クラスを **/usr/lib/security/mkuser.default** ファイルに追加します。

```
user:
    auditclasses = custom
    pgrp = staff
    groups = staff
    shell = /usr/bin/ksh
    home = /home/$USER
```

5. SMIT または **crfs** コマンドを使用して、**/audit** という名前の新しいファイルシステムを作成します。このファイルシステムには、2 つのビンと大規模な監査証跡を保持するだけの大きさが必要です。
6. **audit start** コマンド・オプションを実行し、**/audit** ファイルを検査します。 最初は、2 つのビン・ファイルと空の **trail** ファイルがあるはずです。そのシステムをしばらく使用した後で、以下のようにして読み取ることのできる、**trail** ファイル内の監査レコードができているはずです。

```
auditpr  -hhelpPRtTc -v | more
```

この例では、少数のイベントだけが使用されています。 すべてのイベントを参照するには、すべてのユーザーにクラス名 ALL を指定できます。 これにより、大量のデータが生成されます。 ユーザーの変更および特権の変更に関連するすべてのイベントを、custom クラスに追加することもできます。

重要なファイルへのファイル・アクセスのリアルタイム・モニター:

以下のステップを使用して、重要なファイルへのファイル・アクセスをリアルタイムでモニターすることができます。

次のステップを実行します。

1. 変更をモニターする重要なファイルのリストをセットアップします。例えば、**/etc** 内のすべてのファイルをモニターする場合、次のように **objects** ファイル内で **FILE_Write** イベント用にそれらを構成します。

```
find /etc -type f | awk '{printf("%s:¥n¥tw = FILE_Write¥n¥n",$1)}' >> /etc/security/audit/objects
```

2. すべてのファイル書き込みをリストするように、ストリーム監査をセットアップします。 (この例では、コンソールへのすべてのファイル書き込みがリストされますが、実稼働環境では、バックエンドでイベントを侵入検知システムに送信することもできます。) **/etc/security/audit/streamcmds** ファイルは、次のようになります。

```
/usr/sbin/auditstream | /usr/sbin/auditselect -e "event == FILE_Write" |
auditpr  -hhelpPRtTc -v > /dev/console &
```

3. 次のように、**/etc/security/audit/config** 内で STREAM モード監査をセットアップし、ファイル書き込みイベントのクラスを追加して、そのクラスで監査するすべてのユーザーを構成します。

```
start:
        binmode = off
        streammode = on

stream:
        cmds = /etc/security/audit/streamcmds

classes:
        filemon = FILE_write

users:
        root = filemon
        afx = filemon
        ...
```

4. この時点で **audit start** を実行します。 すべての **FILE_Write** イベントがコンソールに表示されます。

監査イベントの選択:

監査の目的は、システムのセキュリティーを危険にさらすおそれのあるアクティビティーを検出することです。

以下のアクティビティーが無許可ユーザーによって実行されたとき、システム・セキュリティーに違反したことになり、監査の候補となります。

- トラステッド・コンピューティング・ベースのアクティビティーに従事すること
- ユーザーを認証すること
- システムにアクセスすること
- システムの構成を変更すること
- 監査システムを回避すること
- システムを初期化すること
- プログラムをインストールすること
- アカウンティングを変更すること
- システムに出し入れするために情報を転送すること

検査システムには、監査対象となるイベントのデフォルト・セットはありません。必要に応じて、イベントまたはイベント・クラスを選択する必要があります。

あるアクティビティーを監査するためには、どのコマンドまたはプロセスが監査イベントを開始するのかを指定し、そのイベントがシステムの **/etc/security/audit/events** ファイルにリストされていることを確かめなければなりません。 次に、そのイベントを **/etc/security/audit/config** ファイル内の該当するクラスに追加するか、あるいは **/etc/security/audit/objects** ファイル内のオブジェクト・スタンザに追加しなければなりません。 監査イベントと証跡フォーマット指示のリストについては、システムの **/etc/security/audit/events** ファイルを見てください。監査イベント・フォーマットの書き方と使用法の説明については、『**auditpr** コマンド』を参照してください。

監査するイベントの選択を終えたら、類似のイベントを監査クラス別にまとめなければなりません。 次に、監査クラスがユーザーに割り当てられます。

監査クラスの選択

類似のイベントを監査クラスにまとめると、ユーザーへの監査イベントの割り当てが容易になります。 これらの監査クラスは **/etc/security/audit/config** ファイルの classes スタンザに定義されます。

監査クラスの代表例として、次のものがあります。

general
: システムの状態を変更し、ユーザーの認証を変更するイベント。システムのアクセス制御を回避する試みを監査します。

objects
: セキュリティー構成ファイルへの書き込みアクセス

kernel
: カーネル・クラス内のイベントは、カーネルの処理管理機能によって生成されます。

/etc/security/audit/config ファイル内のスタンザの一例を次に示します。

```
classes:
    general = USER_SU,PASSWORD_Change,FILE_Unlink,FILE_Link,FILE_Rename
    system = USER_Change,GROUP_Change,USER_Create,GROUP_Create
    init = USER_Login,USER_Logout
```

監査データ収集方式の選択

監査データがどのような使い方をされるかは、どのデータ収集方式を選択したかによって決まります。 大量のデータを長期間保管する必要がある場合は、BIN 収集を選択してください。 データをその収集と同時に処理する必要があるのであれば、 STREAM 収集を選択してください。 長期保管と即時処理の両方が必要なときは、両方の方式を選択してください。 これらの各メソッドについて、以下に説明します。

ビンの収集
: 大量の監査証跡を長期に保管しておくことができます。 監査レコードは一時的ビンの働きをするファイルに書き込まれます。 このファイルがいっぱいになると、監査サブシステムが他のビン・ファイルに書き込みを行う間に、データが **auditbin** デーモンによって処理され、レコードが監査証跡ファイルに書き込まれて保管されます。

ストリームの収集
: 監査データをその収集と同時に処理することができます。 監査レコードはカーネル内の循環バッファーに書き込まれ、**/dev/audit** を読み取ることによって取得されます。 監査レコードは、表示したり、用紙に印刷して監査証跡書類を作成したり、あるいは **auditcat** コマンドを使用してビン・レコードに変換したりすることができます。

ワークロード・パーティションの監査

WPAR 環境で使用可能な監査には 3 つのタイプ (グローバル、システム、およびグローバルからの監査) があります。

監査はグローバル WPAR、WPAR 内部、またはこの両方で使用可能にできます。システム WPAR およびグローバル WPAR のための監査構成は、非 WPAR 環境での構成に類似しています。 グローバル WPAR 監査は、システムおよびアプリケーション WPAR に対して開始できます。

注**:** アプリケーション WPAR の監査は WPAR の内部からは開始できませんが、グローバル WPAR 監査を使用して開始できます。

グローバル WPAR 監査は、グローバル・システムからのグローバル・システム管理者の WPAR 監査を支援します。 グローバル・システム管理者は、監査するクラスを WPAR ごとにグローバル `/etc/security/audit/config` ファイルに指定して、単一のロケーションからそれぞれの WPAR について監査するレベルを制御できます。

WPARS スタンザを `/etc/security/audit/config` ファイルに追加することで、グローバル・システム管理者は WPAR に対する監査すべきクラスのリストを提供できます。 例えば、

```
WPARS:
<wpar_name> = <auditclass>, ... <auditclass>
```

上の例では、<wpar_name> はシステムの WPAR 名にする必要があり、各監査クラス・パラメーターは該当するクラスのスタンザに定義されていなければなりません。

general、tcpip、および lvm の各クラスを指定して testwpar WPAR の監査を構成するには、次のスタンザを `/etc/security/audit/config` ファイルに追加します。

```
WPARS:
testwpar = general,tcpip,lvm
```

グローバル・システム管理者は、**audit** コマンドを使用して WPAR 名を次のように指定することで、WPAR 上で監査の開始と停止を行うことができます。

```
audit start -@ <wparname1> -@ <wparname2> ...
audit shutdown -@ <wparname1> -@ <wparname2> ...
```

監査するオブジェクトに絶対パスを指定することによって、グローバル環境から WPAR オブジェクトを監査できます。例えば、/wpars/wpar1/etc/security/passwd ファイルの監査イベントを定義するには、WPAR をホストしている AIX システムで、次のスタンザを /etc/security/audit/objects ファイルに追加します。

```
/wpars/wpar1/etc/security/passwd:
     r = "WPAR1_PASSWD_RD"
     w = "WPAR1_PASSWD_WR"
```

この先行スタンザは監査の開始 (-@ <wpar1>) 時に構文解析されて、wpar1 の /etc/security/passwd オブジェクトのオブジェクト監査を使用可能にします。 これらの属性は、/wpars/wpar1/etc/security/passwd ファイルが読み取られるたびに、WPAR1_PASSWD_RD 監査イベントを生成します。 また、これらの属性は、書き込みのためにファイルが開かれるたびに、WPAR1_PASSWD_WR 監査イベントも生成します。

注: グローバル環境からの WPAR 監査を使用可能にする前に、グローバル環境の監査を使用可能にしておく必要があります。

WPAR 名を表示するための監査レポートを生成するには、**auditpr** コマンドを使用できます。 例えば、

```
auditpr -v < /audit/trail
```

NFS 環境での監査

AIX 監査サブシステムは、マウント済みのファイルシステムの監査をサポートします。 クライアントのマウント済みファイルシステムの構成はローカル・ファイルシステムの構成と同様です。 監査可能なマウント済みオブジェクトの監査操作は、監査の概要で説明したように、ローカル・オブジェクトと同様です。 クライアントおよびマウント済みファイルシステムのサーバー上の監査動作は、このトピックの後半で説明します。

NFS クライアント上の監査

クライアントによりマウントされたファイルシステム上の監査可能オブジェクト上で実行されるすべての操作は、クライアントのログに記録されます。 これは、これらのオブジェクト上で NFS サーバーまたは他の NFS クライアントにより実行される操作がない場合か、またはクライアント上で絶対パス監査が有効になっている場合には妥当性があります。

詳しくは、**audit** コマンドのマニュアル・ページを参照してください。 絶対パス監査が有効になっておらず、サーバーまたは他のクライアントによりファイルが変更されると、一連の監査結果は予測不能になります。 このような動作を修正するには、クライアント上で監査を再開します。 1 つのファイルシステムが複数のクライアントにマウントされている場合には、イベントの正確なログを得るためにサーバー上で操作を監査するか、クライアント上で絶対パス監査を使用可能にすることをお勧めします。

注: 監査サブシステム構成では、マウント済み NFS ファイルシステムとしての監査ログ・ファイルシステムの使用はサポートされません。

NFS NFS サーバー上の監査

クライアントおよびサーバーの両方により実行される、マウント済みファイルシステム上のすべての操作は、NFS サーバーでログに記録されます。

サーバー側での制約

- NFS クライアント実行されたすべての操作がサーバーに送信されない場合、NFS キャッシングまたは本来備わっている NFS アーキテクチャーにより、その操作はサーバーにより監査されません。

  例えば**:** ファイルシステムのマウント後、ファイル上で実行される最初の読み取り操作のみがサーバーにより監査されます。 連続読み取り操作はサーバーにログされません。 これは、ファイル、リンク、およびディレクトリー上の読み取り操作に適用されます。
- クライアントによって実行される操作はサーバーに **nfsd** としてログに記録され、ユーザー名は **root** ユーザーになります。

例

File_System と名付けられたファイルシステムは、コマンド **mount server:/File_system /mnt** によりクライアントにマウントされます。 *File_System* ファイルシステム内の *A* と名付けられたファイルをサーバー上で監査する必要がある場合、`/File_system/A` は監査構成ファイル内で構成する必要があります。

A ファイルをクライアント上の *File_System* ファイルシステム内で監査すると決めた場合、`/mnt/A` はクライアント上で監査されるように構成する必要があります。

A ファイルがサーバーとクライアントの両方で監査されるように構成されている場合、*A*ファイル上でサーバーとクライアントの両方により実行される操作は、サーバー上で監査され、ログされます。また、クライアントにより実行される操作はクライアントにログされます。

A ファイル上でクライアントにより実行されたすべての操作は、操作名またはコマンド名の代わりに **nfsd** デーモンとしてサーバーにログされます。

Lightweight Directory Access Protocol (LDAP)

Lightweight Directory Access Protocol (LDAP) は、クライアント/サーバー・モデルにおいて、ローカルまたはリモートからディレクトリー (データベース) 内の情報をアクセスおよび更新するための標準的な方式を定義します。

プロトコルは、ディレクトリーの読み取り、ブラウズ、および検索のために最適化され、当初は X.500 ディレクトリー・アクセス・プロトコルに対する軽量のフロントエンドとして開発されました。 LDAP メソッドは、ホストのクラスターで使用され、セキュリティー認証を中央で行えるようにすると同時に、ユーザー情報およびグループ情報へのアクセスを可能にします。 この機能は、クラスター環境で、認証情報、ユーザー情報、グループ情報をクラスター間で共通にするために使われます。

LDAP 内のオブジェクトは、ディレクトリー情報ツリー (DIT) として知られる、階層構造で保管されます。 良好なディレクトリーは DIT の構造設計から開始されます。 DIT を注意深く設計してから、認証の手段として LDAP を実装する必要があります。

LDAP 認証ロード・モジュール

セキュリティー・サブシステムの LDAP 活用は、LDAP 認証ロード・モジュールとして実装されます。これは、NIS、DCE、KRB5 などのその他のロード・モジュールと概念的に似ています。 ロード・モジュールは **/usr/lib/security/methods.cfg** ファイルで定義されます。

LDAP ロード・モジュールは、LDAP プロトコルを介して、ユーザー認証および集中式ユーザーならびにグループ管理機能を提供します。 LDAP サーバー上で定義されたユーザーは、ローカルで定義されていなくても、LDAP クライアントにログインできるように構成することができます。

AIX LDAP ロード・モジュールは AIX オペレーティング・システムに完全に統合されました。 LDAP 認証ロード・モジュールがユーザー情報とグループ情報を与えるために使用可能になった後は、高水準 API、コマンド、システム管理ツールが通常の方法で働きます。 **-R** フラグは、ほとんどの高水準コマンドに導入されて、異なるロード・モジュール間で働きます。 例えば、クライアント・マシンから *joe* という名前の LDAP ユーザーを作成するには、以下のコマンドを使用します。

```
mkuser -R LDAP joe
```

注**:** LDAP インフラストラクチャーは 1 つのグループ内で無制限の数のユーザーをサポートできますが、1 つのグループに最大 25 000 人のユーザーが作成されており、そのグループに対してさまざまな操作がテストされました。 従来の POSIX インターフェースの一部には、そのグループの完全な情報を戻さないものがあります。 このような制限については、個々の API の資料を参照してください。

LDAP ベースの認証**:**

AIX 上の LDAP ベースの認証の部分としての各種エンティティーに関する制限があります。

LDAP インフラストラクチャーそのものは、データベースの内容に対してどのような制限も指定していません。 ただし、このセクションでは、テスト構成に基づいた制限に関する結果について説明します。 以下の制限は、AIX オペレーティング・システム上での LDAP 認証に関してテスト済みです。

ユーザーの合計数**:** 最大 500 000 人のユーザーが 1 つのシステムで作成され、数百のユーザーの同時認証がテストされました。

グループの合計数**:** 1 つのシステムで最大 500 のグループが作成され、テストされました。

グループ当たりのユーザーの最大数**:** 1 つのグループに最大 25 000 人のユーザーが作成され、そのグループに対してさまざまな操作がテストされました。

従来の POSIX インターフェースの一部には、そのグループの完全な情報を戻さないものがあります。 このような制限については、個々の API の資料を参照してください。 上記の値は実行されたテストに基づくものです。 これらの値は、必要なリソースが存在する場合に、より多くのユーザーやグループを含むシステムを構成できる可能性を妨げるものではありません。

IBM Tivoli Directory Server セキュリティー情報サーバーのセットアップ**:**

LDAP を使用して認証情報、ユーザー情報、グループ情報を提供する LDAP セキュリティー情報サーバーとしてシステムをセットアップするには、最初に LDAP サーバーとクライアントのパッケージをインストールする必要があります。

SSL (Secure Sockets Layer) が必要な場合、IBM Tivoli Directory Server バージョン 6.2 以前の **GSKitV7** パッケージ、または IBM Tivoli Directory Server バージョン 6.3 以降の GSKitV8 もインストールする必要があります。システム管理者は、GSKit キー管理コマンドを使用してキーを作成する必要があります。このコマンドは、GSKitV7 では **gsk7ikm** コマンド、GSKitV8 では **ikeyman** コマンドです。SSL を使用するサーバーの構成に関する詳細は、「Secure Communication with SSL」を参照してください。

サーバーを構成するには、**mksecldap** コマンドを実行します。**mksecldap** コマンドは、LDAP サーバーと *ldapdb2* という名前のバックエンド・データベースを設定し、その LDAP サーバーにローカル・ホスト

からのユーザー情報とグループ情報を設定して、LDAP サーバー管理者 DN (識別名) とパスワードを設定します。このコマンドは、オプションとして、クライアント/サーバー通信の SSL をセットアップすることができます。 **mksecldap** コマンドは、リブートするごとに LDAP サーバーが始動するように、/etc/inittab ファイルに項目を追加することもできます。

AIX ユーザーとグループは、以下のスキーマのいずれかを使用して LDAP サーバーに保管されます。

AIX スキーマ
: aixAccount および aixAccessGroup オブジェクト・クラスを含んでいます。 このスキーマは、AIX ユーザーおよびグループ用の属性の完全なセットを提供します。

RFC 2307 スキーマ
: posixAccount、shadowAccount、および posixGroup オブジェクト・クラスを含んでおり、さまざまなベンダーのディレクトリー製品で使用します。RFC 2307 スキーマは、AIX が使用する属性の小規模なサブセットだけを定義しています。

RFC2307AIX スキーマ
: posixAccount、shadowAccount、および posixGroup オブジェクト・クラスのほか、aixAuxAccount および aixAuxGroup オブジェクト・クラスが組み込まれています。 aixAuxAccount および aixAuxGroup オブジェクト・クラスは、AIX によって使用されるが、RFC 2307 スキーマによって定義されていない属性を提供します。

ユーザーおよびグループには RFC2307AIX スキーマ・タイプの使用を強くお勧めします。 RFC2037AIX スキーマ・タイプは、追加の AIX ユーザー管理機能をサポートするための特別な属性を持っており、RFC 2307 に完全に準拠しています。 RFC2307AIX スキーマ構成を持つ IBM Tivoli Directory Server サーバーは、AIX LDAP クライアントだけでなく、他の RFC 2307 準拠の UNIX および Linux LDAP クライアントもサポートします。

ユーザー情報とグループ情報は、すべて共通 AIX ツリー (サフィックス) の下に保管されます。 デフォルトのサフィックスは「cn=aixdata」です。 **mksecldap** コマンドは、**-d** フラグを介して、ユーザー提供のサフィックスを受け入れます。 ユーザー、グループ、ID、などのために作成するサブツリーの名前は、sectoldif.cfg 構成ファイルによって制御されます。 詳細については、sectoldif.cfg ファイルを参照してください。

この AIX ツリーは ACL (アクセス制御リスト) 保護になっています。 デフォルトの ACL は、**-a** コマンド・オプションで管理者として指定されたエンティティーにのみ、管理特権を認可します。 **-x** と **-X** コマンド・オプションを使用すれば、プロキシー ID に追加の特権を認可できます。 これらのオプションを使用すると、/etc/security/ldap/proxy.ldif.template ファイル内で定義した、プロキシー ID と構成アクセス特権が作成されます。プロキシー ID を作成すると、管理者 ID を使用せずに LDAP クライアントをサーバーにバインドすることが可能となり、LDAP サーバーに関するクライアント管理者特権が制限されることになります。

mksecldap コマンドは、他の目的、例えば、ユーザー ID ルックアップ情報のためにセットアップされる場合でも LDAP サーバー上で実行できます。この例では、**mksecldap** は AIX ツリーを追加し、そのツリーに、既存の LDAP サーバーに対する AIX セキュリティー情報を設定します。このツリーは、他の既存のツリーとは独立して ACL 保護されます。

注**:** **mksecldap** コマンドを実行し、サーバーを AIX セキュリティー情報サーバーに拡張する前に、既存の LDAP サーバーをバックアップする必要があります。

LDAP セキュリティー情報サーバーが正常にセットアップされた場合、 LDAP ユーザーおよびグループを管理し、LDAP ユーザーがこのサーバーにログオンできるように、同じホストをクライアントとしてセットアップすることができます。

LDAP セキュリティー情報サーバーのセットアップが正常に行われなかった場合は、**mksecldap** コマンドに **-U** フラグを指定して実行し、このセットアップを元に戻すことができます。 これによって、`ibmslapd.conf` (または `slapd.conf` または `slapd32.conf`) ファイルがセットアップ前の状態に戻されます。 正常にセットアップできなかった場合は、**mksecldap** コマンドをもう一度試行する前に、**-U** フラグを指定した **mksecldap** コマンドを実行してください。 そうでない場合、未処理のセットアップ情報が構成ファイルに残り、以降のセットアップの失敗の原因になることがあります。 安全な予防措置として、この元に戻すオプションはデータベースやその中のデータには何もしません。 **mksecldap** コマンドが実行される前に、データベースが存在していた可能性もあるからです。 **mksecldap** コマンドで作成されたデータベースがあるときは、それらは手動で削除してください。 また、**mksecldap** コマンドが以前から存在するデータベースにデータを加えた場合は、失敗したセットアップ試行をリカバリーするためにどのステップを行うべきか、決定してください。

関連概念:

SSL によるセキュア通信
LDAP クライアントとサーバー間で使用される認証タイプによって、パスワードが暗号化フォーマット (unix_auth) または平文 (ldap_auth) のいずれかになります。 ネットワーク あるいは (場合によっては) インターネットを介して暗号化されたパスワードを送信するときも、機密漏れ を防止するために、SSL (セキュア・ソケット層) を使用してください。 AIX では、ディレクトリー・サーバーとクライアント間のセキュア通信を提供できる、SSL 用のパッケージを用意しています。

関連情報:

mksecldap コマンド

LDAP クライアントのセットアップ**:**

クライアントが認証ならびにユーザー/グループ情報のために LDAP を使用できるようにセットアップするには、各クライアントが LDAP クライアント・パッケージをインストール済みであることを確認してください。 SSL (Secure Sockets Layer) が必要な場合は、GSKit をインストールし、キーを作成して、LDAP サーバーの SSL キー証明書をこのキーに加える必要があります。

LDAP サーバーのセットアップと同様に、クライアントのセットアップを **mksecldap** コマンドを使用して行うことができます。このクライアントが LDAP セキュリティー情報サーバーに連絡できるようにするには、セットアップ時にサーバー名を提供しておく必要があります。 このサーバーのバインド DN とバインド・パスワードも、クライアントがサーバー上の AIX ツリーへアクセスする時に必要です。 **mksecldap** コマンドは、サーバー・バインド DN、パスワード、サーバー名、サーバー上の AIX ツリー DN、SSL キー・パスとパスワード、およびその他の構成属性を **/etc/security/ldap/ldap.cfg** ファイルに保存します。

mksecldap コマンドは、BIND パスワードと SSL キー・パスワード (SSL が構成されている場合) を、暗号化された形式で `/etc/security/ldap/ldap.cfg` ファイルに保存します。暗号化されたパスワードはシステム固有であり、そのパスワードが生成されたシステム上で **secldapclntd** デーモンによってのみ実行することができます。 **secldapcIntd** デーモンは、`/etc/security/ldap/ldap.cfg` ファイルから、平文または暗号化されたパスワードを使用することができます。

クライアントのセットアップ時に、複数のサーバーを **mksecldap** コマンドに提供することができます。この場合は、クライアントは、提供されている順にサーバーに連絡し、クライアントが正常にバインドできた最初のサーバーとの接続を設定します。 クライアントとサーバーとの間に接続エラーが発生した場合

は、同じ論理を使用して、再接続要求が試行されます。 セキュリティー LDAP 活用モデルでは参照はサポートされていません。 複製サーバーが同期を保っていることは、重要です。

クライアントは、クライアント側のデーモン (**secldapcIntd**) を使用して LDAP セキュリティー情報サーバーと通信します。LDAP ロード・モジュールがクライアント上で使用可能になっていれば、高水準コマンドが、LDAP で定義されたユーザー用のライブラリー API を介してデーモンへ経路指定されます。 デーモンは要求された LDAP エントリーのキャッシュを保守します。 要求がキャッシュで満たされない場合は、デーモンは、サーバーに照会し、キャッシュを更新し、呼び出し側に情報を戻します。

別の微調整オプションとして、クライアントのセットアップ時の **mksecldap** コマンドに、デーモンの使用するスレッド数の設定、キャッシュ・エントリー・サイズ、キャッシュ満了タイムアウトなどを指定することができます。 これらのオプションは、経験豊かなユーザーのためのものです。 ほとんどの環境では、デフォルト値だけで十分です。

クライアント・セットアップの最終ステップで、**mksecldap** コマンドは、クライアント側のデーモンを開始し、**/etc/inittab** ファイルにエントリーを加えて、リブートごとにデーモンが開始するようにします。セットアップが正常に終了したかどうかは、**ls-secldapclntd** コマンドを介して **secldapcIntd** デーモン・プロセスを検査することにより確認できます。LDAP セキュリティー情報サーバーがセットアップされて実行されるとき、このセットアップが正常に終了していれば、このデーモンが実行されます。

サーバーは、クライアントより前にセットアップする必要があります。 クライアントのセットアップは、サーバー上にあるマイグレーション済みデータ依存します。 クライアントをセットアップするには、次のステップを実行します。

1. IBM Tivoli Directory Server クライアント・ファイルセットを AIX オペレーティング・システム上でインストールします。
   - IBM Tivoli Directory Server 5.2 上で、`ldap.client` ファイルセットをインストールします。
   - IBM Tivoli Directory Server 7.1 上で、`idsldap` ファイルセットをインストールします。
2. LDAP を構成するために、次のコマンドを実行します。

   ```
   # mksecldap -c -h server1.ibm.com -a cn=admindn -p adminpwd -d cn=basedn
   ```

   上記の値は環境に合った適切な値に置き換えてください。

関連情報:

mksecldap コマンド

secldapclntd コマンド

LDAP ネットグループ用のクライアント使用可能性**:**

NIS-LDAP の一部 (ネーム・レゾリューション方式) としてネットグループを使用することができます。

LDAP ネットグループ用にクライアントを使用可能にするには、以下のステップを実行します。

1. LDAP ベースのユーザー・グループ管理をインストールしてセットアップします。これについては『LDAP クライアントのセットアップ』(173 ページの『LDAP クライアントのセットアップ』) に詳しい説明があります。

   ネットグループ・セットアップが完了していない場合は、LDAP 定義済みユーザーはいずれもシステムによってリストされます。 例えば、*nguser* がネットグループ *mygroup* に属するユーザーであって、LDAP サーバーに既に定義されている場合、そのユーザーは `lsuser -R LDAP nguser` としてリストされます。

2. ネットグループ機能を使用可能にするには、**/usr/lib/security/methods.cfg** ファイルにある LDAP 用モジュール定義に、ネットグループ値のあるオプション属性を組み込む必要があります。 **/usr/lib/security/methods.cfg** ファイルを編集し、LDAP スタンザに行 `options = netgroup` を追加します。 これで、この LDAP ロード・モジュールにネットグループ対応のマークが付きます。 次に例を示します。

   ```
   LDAP:
         program = /usr/lib/security/LDAP
         program_64 =/usr/lib/security/LDAP64
         options = netgroup
   ```

   これで、`lsuser -R LDAP nguser`、`lsuser nguser`、または `lsuser -R LDAP -a ALL` のいずれのコマンドも、ユーザーをリストしなくなります。 この時点で、LDAP はこのクライアントのネットグループ専用データベースと見なされ、このクライアントへのアクセスが使用可能になっているネットグループはまだないと見なされます。

3. **/etc/passwd** ファイルを編集し、システムへのアクセスを必要とするネットグループの行を追加します。 例えば、*mygroup* が LDAP サーバー上のネットグループであって、必要なユーザーが含まれている場合、次の行を追加します。

   ```
   +@mygroup
   ```

4. **/etc/group** ファイルを編集し、`+:` 行を追加して、NIS によるグループのルックアップを使用可能にします。

   ```
   +:
   ```

   ここで `lsuser nguser` を実行すると、そのユーザーが戻されます。これは *nguser* がネットグループ *mygroup* に入っているためです。

   `lsuser -R LDAP nguser` コマンドではこのユーザーが見つかりませんが、`lsuser -R compat nguser` を実行すると、このユーザーは **compat** ユーザーと見なされるために見つかるようになります。

5. ネットグループ・ユーザーがシステムに対して認証されるためには、使用する方式が AIX 認証メカニズムに知られている必要があります。 **/etc/security/user** ファイル内の default スタンザに `SYSTEM = compat` が含まれている場合、**/etc/passwd** ファイルに追加されたそのネットグループ内のすべてのネットグループ・ユーザーの認証が可能になります。 また、もう 1 つのオプションとして、必要なユーザーの **/etc/security/user** ファイルに手動でスタンザを追加して、ユーザーを個別に構成する方法もあります。 *nguser* の場合のスタンザの例は次のようになります。

   ```
   nguser:
          SYSTEM = compat
          registry = compat
   ```

   これで、許可されたネットグループ内のネットグループ・ユーザーが、システムに対して認証されます。

   ネットグループ・フィーチャーを使用可能にすると、次のような条件もアクティブになります。

   - LDAP レジストリーのメンバーとして **/etc/security/user** ファイルに定義されているユーザー (`registry=LDAP` および `SYSTEM="LDAP"` が定義されている) は、LDAP ユーザーとして認証されません。 現在、これらのユーザーは **nis_ldap** ユーザーになっているため、ネイティブ NIS ネットグループのメンバーシップを必要とします。
   - レジストリー compat の意味は、ネットグループを使用するモジュールを組み込むように拡張されます。 例えば、LDAP モジュールがネットグループ使用可能になっている場合、compat には、ファイル、NIS、および LDAP レジストリーが含まれます。 これらのモジュールから検索されたユーザーは、compat というレジストリー値を持ちます。

関連情報

- NFS 用 exports ファイル資料
- TCP/IP 用 .rhosts ファイル・フォーマット資料
- TCP/IP 用 hosts.equiv ファイル・フォーマット資料

サポートされる **LDAP** サーバー**:**

AIX LDAP ベースのユーザーおよびグループ管理では、IBM Tivoli Directory Server、RFC 2307 対応スキーマ付きの IBM 以外のサーバー、および Microsoft Active Directory サーバーをサポートします。

IBM Tivoli Directory Server

AIX ユーザー/グループ管理は、IBM Tivoli Directory Server を使用して構成することを強くお勧めします。ユーザーおよびグループ管理のための IBM Tivoli Directory Server の設定の詳細については、『IBM Tivoli Directory Server セキュリティー情報サーバーの設定』を参照してください。

IBM 以外の **Directory Server**

AIX では、ユーザーおよびグループが RFC 2307 スキーマを使用して定義されているさまざまなディレクトリー・サーバーをサポートします。 そのようなサーバーに接続する LDAP クライアントとして構成された場合、AIX は RFC 2307 スキーマにより IBM Tivoli Directory Server と同じ方法でそれらのサーバーを使用します。これらのサーバーは、LDAP バージョン 3 プロトコルをサポートする必要があります。

RFC 2307 スキーマは AIX で使用できるユーザーおよびグループ属性の一部のみを定義するので、AIX がこのような LDAP サーバー (例えば、ユーザー・パスワード再設定実施、パスワード・ヒストリー、ユーザーごとのリソース制限、AIX `hostsallowedlogin` および `hostsdeniedlogin` 属性を使用した特定のシステムへのログイン制御、機能など) を使用するように構成されている場合は、一部の AIX ユーザーおよびグループ管理が機能しない可能性があります。

AIX では、非 RFC 2307 対応のディレクトリー・サーバーをサポートしません。 ただし、AIX は RFC 2307 対応でないサーバーでも作動するようにできます。この場合、ユーザーおよびグループはすべての必要な UNIX 属性を用いて定義されます。 AIX で必要とされるユーザーおよびグループ属性の最小セットは、RFC 2307 で定義されているセットです。 このようなディレクトリー・サーバーのサポートには、手動の構成が必要です。 AIX では、このためにスキーマ・マッピング・メカニズムを使用することができます。 スキーマ・ファイル・フォーマットおよびスキーマ・ファイル使用の詳細については、『LDAP Attribute Mapping File Format』を参照してください。

Microsoft Active Directory

AIX では、Microsoft Active Directory (AD) をユーザーおよびグループ管理のための LDAP サーバーとしてサポートします。 AD サーバーには、UNIX サポート・スキーマがインストールされていなければなりません。 AD の UNIX サポート・スキーマは、Microsoft Service For UNIX (SFU) パッケージから取り入れられます。 それぞれの SFU バージョンは、ユーザーおよびグループのスキーマ定義が先行バージョンとは若干異なります。 AIX では、SFU スキーマのバージョン 3.0 および 3.5 を用いた Windows 2000 および 2003 で実行する AD、および組み込み UNIX スキーマを用いた Windows 2003 R2 で実行する AD をサポートします。

UNIX システムと Windows システムの間のユーザーおよびグループ管理における相違によって、サーバーが AD である場合には、AIX コマンドのすべてが LDAP ユーザーに作用するわけではありません。作用しないコマンドには、**mkuser** および **mkgroup** があります。ほとんどのユーザーおよびグループ管

理コマンドは、AIX が AD との結び付けに使用する ID に指定されているアクセス権に応じて機能します。 このようなコマンドには、**lsuser**、**chuser**、**rmuser**、**lsgroup**、**chgroup**、**rmgroup**、**id**、**groups**、**passwd**、および **chpasswd** があります。

AIX では、Windows サーバーに対して 2 つのユーザー認証メカニズム、すなわち LDAP 認証および Kerberos 認証をサポートします。 どちらのメカニズムでも、AIX は、AIX 上の対応するユーザー・アカウントを必要とせずに、AD に対する LDAP プロトコルを介してユーザー識別をサポートします。

***LDAP* を使用して *Active Directory* で作業を行う *AIX* オペレーティング・システムの構成:**

AIX では、Microsoft Active Directory (AD) をユーザーおよびグループ管理のための LDAP サーバーとしてサポートします。 AD サーバーには UNIX サポート・スキーマがインストールされている必要があります。

管理者は **mksecldap** コマンドを使用して、IBM Tivoli Directory Server の場合と同じ方法で AD サーバー上で AIX を構成できます。**mksecldap** コマンドは処理を単純にするために、構成の詳細のすべてを非表示にします。 AD サーバーで AIX を構成するための **mksecldap** コマンドを実行する前に必要な条件は、以下のとおりです。

1. AD サーバーには UNIX サポート・スキーマがインストールされていなければならない。
2. AD サーバーには UNIX が使用可能になっているユーザーが含まれていなければならない。

UNIX スキーマをインストールして、 UNIX がサポートされている AD ユーザーを使用可能にするための詳細情報については、関連する Microsoft 文書を参照してください。

多くの場合、AD スキーマには同じ UNIX 属性に対して、複数の属性定義が含まれています (例えば、複数のユーザー・パスワードとグループ・メンバー定義があります)。 AIX はこの大多数をサポートしますが、使用する定義を選択するときに注意深く検討して計画する必要があります。 AIX システムと同じ AD を共有するその他の非 AIX には、矛盾を避けるために同じ定義の使用をお勧めします。

***Active Directory* のパスワード属性の選択:**

AIX では、**unix_auth** および **ldap_auth** の 2 つの認証メカニズムをサポートします。

unix_auth では、Microsoft Active Directory (AD) のパスワードは、暗号化形式にする必要があります。認証時に、暗号化されたパスワードは AD から取り出され、ユーザーが入力したパスワードの暗号化形式と比較されます。 両方のパスワードが一致すると、認証は正常に終了します。 **ldap_auth** モードでは、AIX はユーザーを、ユーザーの ID および指定されたパスワードによるサーバーへの LDAP バインド操作によって認証します。 このバインド操作が正常に終了した場合に、ユーザーは認証されます。 AD では、複数のユーザー・パスワード属性をサポートします。 異なる AIX 認証モードでは、異なる AD ユーザー・パスワード属性が必要になります。

unix_auth モード

次の AD パスワード属性は、**unix_auth** モードで使用することができます。

- **userPassword**
- **unixUserPassword**
- **msSFU30Password**

AIX におけるパスワードの管理は、AD の複数のパスワード属性によって困難になる可能性があります。どのパスワード管理属性が UNIX のクライアントによって使用されるべきかを知ることが難しい場合があります。 AIX LDAP 属性のマッピング機能により、ユーザーの必要に応じてパスワード管理をカスタマイズすることができます。

デフォルトでは、AIX は、Windows 2000 および 2003 で実行する AD には **msSFU30Password** 属性、Windows 2003 R2 では **userPassword** 属性を使用します。 異なるパスワードが使用された場合には、`/etc/security/ldap/sfu30user.map` ファイル (AD が Windows 2003 R2 で実行される場合には `/etc/security/ldap/sfur2user.map` ファイル) を変更する必要があります。 ワード **spassword** で始まる行を検索し、その行の 3 番目のフィールドを所望の AD パスワード属性名に変更してください。 詳細については、「LDAP Attribute Mapping File Format」を参照してください。 変更の後で、**mksecldap** コマンドを実行して AIX LDAP クライアントを構成します。 AIX LDAP クライアントが既に構成されている場合は、**restart-secldapclntd** コマンドを実行して **secldapclntd** デーモンを再始動し、変更を取り入れます。

unix_auth モードでは、Windows と UNIX でパスワードが同期しないことがあるために、両システム間でパスワードが異なる場合があります。 これは、パスワードを AIX から Windows に変更した時に、Windows では **uncodepwd** パスワード属性を使用しているために起こります。 AIX **passwd** コマンドを使用して UNIX パスワードを Windows パスワードと同じにするよう再設定できますが、 UNIX パスワードを AIX から変更するときに、AIX では Windows パスワードの自動変更はサポートされません。

ldap_auth モード

Active Directory には **unicodepwd** パスワード属性もあります。 このパスワード属性は、Windows システムによって、Windows のユーザーを認証するために使用されます。 **unicodePwd** パスワードは、AD へのバインド操作で使用しなければなりません。 **unix_auth** モードのもとで記述されたパスワードはいずれも、バインド操作では機能しません。 **ldap_auth** オプションがコマンド・ラインから指定されている場合は、**mksecldap** コマンドによって、マニュアル・ステップを必要としないクライアント構成時にパスワード属性が AD の **unicodePwd** 属性にマップされます。

AIX パスワードを **unicodePwd** 属性でマッピングすることにより、AD で定義されているユーザーは、同じパスワードを使用して Windows および AIX システムにログインすることができます。 AIX または Windows のどちらかから再設定されたパスワードは、AIX と Windows の両方のシステムに有効です。

Active Directory のグループ・メンバー属性の選択*:*

Microsoft's Service for UNIX は、**memberUid**、**msSFU30MemberUid**、および **msSFU30PosixMember** の各グループ・メンバー属性を定義します。

memberUid および **msSFU30MemeberUid** 属性はユーザー・アカウント名を受け入れ、**msSFU30PosixMember** は完全 DN のみを受け入れます。 例えば、AD で定義されているユーザー・アカウント *foo* (ラストネーム *bar* 付き) の場合は、次のようになります。

- **memberUid: foo**
- **msSFU30MemberUid: foo**
- **msSFU30PosixMember: CN=foo bar,CN=Users,DC=austin,DC=ibm,DC=com**

AIX オペレーティング・システムは、これらのすべての属性をサポートします。 どの属性を使用するかを決定するには、AD 管理者にお尋ねください。 デフォルトでは、**mksecldap** コマンドは、Windows 2000 および 2003 で稼働する AD に対して **msSFU30PosixMember** 属性、および Windows 2003 R2 で稼働する AD に対して **uidMember** 属性を使用するよう AIX オペレーティング・システムを構成します。

AD がユーザーを Windows からのグループに追加するときに該当の属性を選択するので、このような選択は AD の動作が原因です。ユーザーのビジネス戦略では、複数のプラットフォームをサポートするためにデフォルト以外のグループ・メンバー属性を使用する必要が生じる場合があります。

別のグループ・メンバー属性が必要な場合は、グループ・マッピング・ファイルを編集してマッピングを変更することができます。 AD 用のグループ・マッピング・ファイルは、Windows 2000 および 2003 で稼働する `/etc/security/ldap/sfu30group.map`、および Windows 2003 R2 用の `/etc/security/ldap/sfur2group.map` です。 ワード **users** で始まる行を見つけて、その 3 番目のフィールドをグループ・メンバー用の所望の属性名で置き換えます。 詳細については、『LDAP Attribute Mapping File Format』を参照してください。 この変更後に AIX LDAP を構成するための **mksecldap** コマンドを実行するか、または AIX クライアントが既に構成されている場合は、この変更を取り込むための **secldapclntd** デーモンを再始動するために **restart-secldapclntd** コマンドを実行します。

複数の組織単位*:*

使用する AD サーバーには複数の組織単位を定義することが可能であり、各組織単位にはユーザー・セットが含まれます。

たいていの Windows AD ユーザーは、 **cn=users,...** サブツリーに定義されますが、別の場所に定義することも可能です。 AIX 複数基本 DN フィーチャーは AD サーバーなどにも使用できます。 詳細情報については、『複数基本 DN サポート』を参照してください。

Windows サーバー用の ***Kerberos*** 認証*:*

LDAP 認証メカニズムに加えて、AIX オペレーティング・システムは、Windows サーバー用の Kerberos プロトコルを使用したユーザー認証もサポートします。

AIX オペレーティング・システムは、KRB5ALDAP 複合ロード・モジュールを作成して、Windows Active Directory の Windows KDC および LDAP 識別に関する Kerberos 認証をサポートします。 ユーザー識別情報は Microsoft Active Directory から取られるため、AIX オペレーティング・システム上の対応するユーザー・アカウントを作成する必要はありません。

LDAP ユーザー管理**:**

LDAP セキュリティー情報サーバー上のユーザーやグループの管理は、高水準コマンドを使用すれば、どの LDAP クライアントからでも行うことができます。

多くの場合、高水準コマンドには **-R** フラグが追加されますが、これによって、LDAP の他にも、DCE、NIS、KRB5 などの認証ロード・モジュールでも、ユーザーやグループを管理することができます。 **-R** フラグの使用に関する詳細は、ユーザーおよびグループを管理するコマンドをそれぞれ参照してください。

LDAP によってユーザーを認証できるようにするには、**chuser** コマンドを実行して、ユーザーの **SYSTEM** 属性値を LDAP に変更します。 定義済みの構文に従って **SYSTEM** 属性値を設定することにより、ユーザーを複数のロード・モジュール (例えば、compat と LDAP) で認証することが可能になります。 ユーザーの認証方式の設定については、 79 ページの『ユーザー認証』および **/etc/security/user** ファイルに定義されている **SYSTEM** 属性の構文を参照してください。

以下のいずれかの形式で、**-u** フラグを指定して **mksecldap** コマンドを実行すると、ユーザーはクライアント・セットアップ時に LDAP ユーザーになることができます。

1. 次のコマンドを実行します。

```
mksecldap -c -u user1,user2,...
```

ここで、*user1,user2,...* はユーザーのリストです。 このリストの中のユーザーは、ローカルに定義されていてもよいし、あるいはリモートの LDAP 定義ユーザーであってもかまいません。 **/etc/security/user** ファイル内の上記の各ユーザー・スタンザで、**SYSTEM** 属性が LDAP に設定されます。このようなユーザーは、LDAP によってのみ認証されます。 このリスト内のユーザーは、LDAP セキュリティー情報サーバーに存在していなければなりません。存在していないと、このホストからログインできません。 **SYSTEM** 属性を変更し、複数のメソッド (例えば、ローカルと LDAP) での認証を可能にするためには、**chuser** コマンドを実行してください。

2. 次のように実行します。

```
mksecldap -c -u ALL
```

このコマンドによって、ローカルに定義されているすべてのユーザーに関して、**/etc/security/user** ファイル内の各ユーザー・スタンザで、**SYSTEM** 属性が LDAP に設定されます。 このようなユーザーは、すべて LDAP によってのみ認証されます。ローカルに定義済みのユーザーは、LDAP セキュリティー情報サーバーに存在していなければなりません。存在していないと、このホストからログインできません。 LDAP サーバーには定義されているが、ローカルには定義されていないユーザーは、このホストからはログインできません。 リモートの LDAP 定義ユーザーがこのホストからログインできるようにするには、**chuser** コマンドを実行して、そのユーザーで **SYSTEM** 属性を LDAP に設定します。

別の方法として、ローカルに定義されていても、されていなくても、すべての LDAP ユーザーをローカル・ホストの LDAP で認証可能にするには、**/etc/security/user** ファイルの「デフォルト」スタンザがその値として「LDAP」を使用するように変更する方法もあります。 自分の **SYSTEM** 属性に値が定義されていないユーザーは、デフォルト・スタンザに定義されている値に従う必要があります。 例えば、default スタンザに "SYSTEM = "compat"" と指定されている場合、これを "SYSTEM = "compat OR LDAP"" に変更すると、AIX または LDAP のいずれかによるこれらのユーザーの認証が可能になります。 デフォルト・スタンザを "SYSTEM = "LDAP"" に変更すると、これらのユーザーは LDAP によってだけ認証可能になります。 **SYSTEM** 属性値が定義されているユーザーは、デフォルト・スタンザには影響されません。

複数の基本 ***DN*** のサポート*:*

AIX では、複数の基本 DN をサポートします。 エンティティーごとに最大 10 の基本 DN を /etc/security/ldap/ldap.cfg ファイルに指定することができます。

基本 DN は、/etc/security/ldap/ldap.cfg ファイル中の順序で優先順位付けがなされています。複数の基本 DN の場合の AIX コマンドによる操作は、基本 DN の優先順位に従って次の方法で行われます。

- 照会操作 (例えば、**lsuser** コマンドによる) は、基本 DN に対して、マッチング・アカウントが見つかるまで優先順位に従って行われます。また、すべての基本 DN を検索してもマッチング・アカウントが見つからなかった場合にはエラーが戻されます。 ALL の照会では、結果としてすべての基本 DN からすべてのアカウントが戻されます。
- 変更操作 (例えば、**chuser** コマンドによる) は、最初のマッチング・アカウントに対して行われます。
- 削除操作 (例えば、**rmuser** コマンドによる) は、最初のマッチング・アカウントに対して行われます。
- 作成操作 (例えば、**mkuser** コマンドによる) は、最初の基本 DN に対してのみ行われます。 AIX では、他の基本 DN に対するアカウントの作成をサポートしません。

矛盾が含まれていないアカウント・データベースを維持する責任は、ディレクトリー・サーバーの管理者が負います。 同じアカウントに複数の定義がある場合は、異なるサブツリーごとに、最初のアカウントのみ

が AIX 可視となります。 検索操作では、最初のマッチング・アカウントのみが戻されます。 同様に、変更または削除の操作は、最初のマッチング・アカウントに対してのみ行われます。

mksecldap コマンドは、LDAP クライアントの構成に使用される場合は、エンティティーごとに基本 DN を検出してそれを /etc/security/ldap/ldap.cfg ファイルに保存します。複数の基本 DN が LDAP サーバーでエンティティーに使用可能な場合は、**mksecldap** コマンドは、いずれか 1 つの基本 DN をランダムに使用します。 AIX に複数の基本 DN を処理させるには、**mksecldap** コマンドが正常に完了した後で /etc/security/ldap/ldap.cfg ファイルを編集する必要があります。適切な基本 DN の定義を検索して、必要な追加の基本 DN を追加します。 AIX ではエンティティーごとに最大 10 の基本 DN をサポートし、追加の基本 DN はすべて無視されます。

AIX では、基本 DN ごとにユーザー定義のフィルターおよび検索スコープもサポートします。 基本 DN には、その対等基本 DN とは異なる可能性がある独自のフィルターおよびスコープを指定することができます。 フィルターを使用して、AIX 可視のアカウントのセットを定義することができます。

フィルターを満たすアカウントのみが AIX 可視となります。

LDAP サーバーでの ***SSL*** のセットアップ***:***

LDAP サーバー上で SSL (Secure Sockets Layer) をセットアップするために、LDAP 暗号ファイルセットおよび **GSKit** ファイルセットをインストールし、サーバー暗号化サポートを使用可能にします。これらのファイルセットは AIX 拡張パック上にあります。

次に、以下のステップを実行して、IBM ディレクトリー・サーバー認証用の SSL サポートを使用可能にします。

1. IBM Tivoli Directory Server バージョン 6.2 の IBM Tivoli Directory Server **GSKit**、または IBM Tivoli Directory Server バージョン 6.3 の **GSKitv8** をインストールします (まだインストールしていない場合)。
2. 正しい GSKit キー管理ユーティリティーを使用して、IBM ディレクトリー・サーバー・プライベート・キーとサーバー証明書を生成します。IBM Tivoli Directory Server バージョン 6.2 では **gsk7ikm** ユーティリティーを、また、IBM Tivoli Directory Server version 6.3 以降では ikeyman ツールを使用してください。サーバーの証明書は、商業認証局 (CA)、例えば、VeriSign が署名することも、あるいは、GSKit キー管理ツールを用いて自己署名することもできます。認証局の公開証明書 (または自己署名証明書) は、クライアント・アプリケーションのキー・データベース・ファイルにも配布する必要があります。
3. サーバーのキー・データベース・ファイルおよび関連のパスワード・スタッシュ・ファイルをサーバー上に保管します。キー・データベースのデフォルト・パス、**/usr/ldap/etc** ディレクトリーは標準的なロケーションです。
4. 次のコマンドを実行してサーバーをセットアップします。ここで、**mykey.kdb** はキー・データベース、*keypwd* はキー・データベースのパスワードです。

   ```
   # mksecldap -s -a cn=admin -p pwd -S rfc2307aix -k /usr/ldap/etc/mykey.kdb -w keypwd
   ```

LDAP クライアントでの ***SSL*** のセットアップ***:***

LDAP クライアント上で SSL を使用するには、ldap.max_crypto_client および GSKit ファイルセットを AIX 拡張パックからインストールします。

次に、サーバーを SSL のために使用可能にした後、以下のステップを実行して、LDAP 用の SSL サポートを使用可能にする。

1. gsk7ikm を実行して、各クライアントにキー・データベースを生成する。

2. サーバー証明書を各クライアントにコピーする。 サーバー SSL が自己署名証明書を使用している場合は、最初に証明書をエクスポートする必要があります。
3. 各クライアント・システム上で gsk7ikm を実行し、サーバー証明書をキー・データベースにインポートする。
4. 各クライアントについて SSL を使用可能にする。

   ```
   # mksecldap -c -h servername -a adminDN -p pwd -k /usr/ldap/etc/mykey.kdb -p keypwd
   ```

   ここで、**/usr/ldap/etc/mykey.kdb** はキー・データベースへのフルパス、*keypwd* はキーのパスワードです。 コマンド・ラインからキー・パスワードを入力しないと、同じディレクトリーからのスタッシュ (隠しておいた)・パスワード・ファイルが使用されます。 このスタッシュ・ファイルの名前は、キー・データベースの名前に拡張子 **.sth** が付いたものと同じでなければなりません (例えば、**mykey.sth**)。

LDAP のホスト・アクセス制御**:**

AIX では、システムのユーザー・レベルでのホスト・アクセス (ログイン) 制御が提供されています。 管理者は **SYSTEM** 属性を LDAP に設定することにより、AIX システムにログインするように LDAP ユーザーを構成できます。

SYSTEM 属性は **/etc/security/user** ファイル内にあります。属性の値を設定するには、次のように **chuser** コマンドを使用することができます。

```
# chuser -R LDAP SYSTEM=LDAP registry=LDAP foo
```

注**:** このタイプの制御を使用する場合、デフォルトの **SYSTEM** 属性を LDAP に設定しないでください。そのようにすると、すべての LDAP ユーザーがシステムにログインできるようになります。

これで、ユーザー *foo* がこのシステムにログインできるように LDAP 属性が設定されます。 また、レジストリーが LDAP に設定されます。これにより、ログイン・プロセスで *foo* のログイン試行が LDAP に記録できるようになり、さらに、LDAP で行われるすべてのユーザー管理タスクが可能になります。

特定のユーザーがログインできるように、管理者は各クライアント・システムでそのようなセットアップを実行する必要があります。

AIX には、1 人の LDAP ユーザーが特定の LDAP クライアント・システムのみにログインするように制限するフィーチャーがあります。このフィーチャーにより、ホスト・アクセス制御の管理を集中化することができます。 管理者は、ユーザー・アカウントに 2 つのホスト・アクセス制御リスト (許可リストと拒否リスト) を指定できます。 これらの 2 つのユーザー属性は、ユーザー・アカウントとともに LDAP サーバーに保管されます。 ユーザーは許可リストで指定されているシステムまたはネットワークにアクセスできるのに対し、拒否リスト内のシステムまたはネットワークへのアクセスは拒否されます。 システムが許可リストと拒否リストの両方で指定されている場合、ユーザーはシステムへのアクセスを拒否されます。ユーザーがアクセス・リストを指定するには 2 とおりの方法があります。ユーザーの作成時に **mkuser** コマンドを使用する方法と、既存のユーザーに **chuser** コマンドを使用する方法です。 ユーザーに許可リストと拒否リストが両方ともない場合、後方互換性のために、デフォルトでユーザーはすべての LDAP クライアント・システムへのログインを許可されます。

ユーザーの許可リストおよび拒否リストの設定の例を、以下に示します。

```
# mkuser -R LDAP hostsallowedlogin=host1,host2 foo
```

これにより、ユーザー *foo* が作成され、ユーザー *foo* に、host1 および host2 へのログインだけが許可されます。

```
# mkuser -R LDAP hostsdeniedlogin=host2 foo
```

これにより、ユーザー *foo* が作成され、ユーザー *foo* に host2 以外のすべての LDAP クライアント・システムへのログインが許可されます。

```
# chuser -R LDAP hostsallowedlogin=192.9.200.1 foo
```

これにより、アドレス 192.9.200.1 のクライアント・システムにログインするための許可が与えられて、ユーザー *foo* が設定されます。

```
# chuser -R LDAP hostsallowedlogin=192.9.200/24 hostsdeniedlogin=192.9.200.1 foo
```

これにより、アドレス 192.9.200.1 のクライアント・システム以外の、192.9.200/24 サブネット内のすべてのクライアント・システムにログインするための許可が与えられて、ユーザー *foo* が設定されます。

詳しくは、『**chuser** コマンド』を参照してください。

SSL によるセキュア通信**:**

LDAP クライアントとサーバー間で使用される認証タイプによって、パスワードが暗号化フォーマット (unix_auth) または平文 (ldap_auth) のいずれかになります。 ネットワークあるいは (場合によっては) インターネットを介して暗号化されたパスワードを送信するときも、機密漏れを防止するために、SSL (セキュア・ソケット層) を使用してください。 AIX では、ディレクトリー・サーバーとクライアント間のセキュア通信を提供できる、SSL 用のパッケージを用意しています。

詳しくは、次の関連情報を参照してください。

- 181 ページの『LDAP サーバーでの SSL のセットアップ』
- 181 ページの『LDAP クライアントでの SSL のセットアップ』

LDAPA 認証専用モードの使用**:**

LDAP モジュールは、ユーザー認証とユーザー識別の両方をサポートする全機能を持つモジュールです。 LDAPA モジュールでは認証専用モードが提供されます。 LDAPA モジュールは LDAP モジュールに似ていますが、認証専用モードを使用する場合に指定できます。

認証専用モードでは、スタンドアロン・モジュールではなく複合モジュールを形成するために、別のデータベース・モジュールに LDAPA モジュールを結合する必要があります。 LDAPA モジュールはユーザー認証を行いますが、2 番目のモジュールでは識別を行います。 この結合されたモジュールは複合モジュールと呼ばれます。 この複合モジュールに対して、LDAP サーバーとデータベース・サーバーの両方でユーザーを定義する必要があります。

LDAPA モジュールの場合、グループ情報はデータベース・サーバーから提供されます。 例えば、LDAPA ファイルの場合では、グループ情報はローカルの /etc/group ファイルからのものです。一部の LDAP ユーザーが LDAP グループのみに属する場合、対応する LDAP グループをデータベース・サーバーに作成してから、LDAPA ファイル・モジュールを構成する必要があります。 この対応するグループを作成することによって、グループ設定がデータベース・サーバー上に存在しないために LDAPA ファイル・ユーザーがグループ設定を解決できないケースを避けることができます。

注**:** LDAPA モジュールはユーザーの作成および除去をサポートしません。 LDAPA ファイル・ユーザーを作成するには、システム管理者は LDAP モジュールを使用して LDAP ユーザーを作成し、次に同じユーザーをローカル側に作成する必要があります。次に、**chuser** コマンドを使用して、ユーザーの SYSTEM およびレジストリーを LDAPAfiles に設定することによって、そのユーザーを LDAPA ファイル・ユーザーにします。

LDAPA モジュールを使用して認証専用モードで LDAP を構成するには、-i <*databaseModule*> オプションを指定して **mksecldap** コマンドを使用します。このコマンドにより、options = authonly 設定を使用して LDAPA モジュールおよび <*databaseModule*> 複合ロード・モジュールが作成されます。

例えば、認証専用モードで LDAP を構成し、データベース・モジュール用のローカル・ファイルを作成するには、以下の例を参照してください。

```
mksecldap -c –h <ldap server> -a <binddn> -p <bind password> -i files
```

/usr/lib/security/methods.cfg ファイルは以下のコマンドによって更新されます。

```
LDAPA:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64
        options = authonly

LDAP:
        program = /usr/lib/security/LDAP
        program_64 =/usr/lib/security/LDAP64

LDAPAfiles:
        options = db=BUILTIN,auth=LDAPA
```

LDAPA スタンザでは、options = authonly 設定は、LDAPA モジュールを認証専用モードに設定することを示します。 LDAPAfiles スタンザは、複合ロード・モジュールを定義します。

LDAP モジュールは、RBAC のようなユーザー/グループ以外のデータを解決するために保持されます。LDAP モジュールは、LDAPA モジュールとは無関係にスタンドアロンの認証モジュールとしてまだ使用できます。

関連情報:

mksecldap コマンド

LDAPA がサポートする属性*:*

認証専用モードの LDAPA モジュールは、限られた数の AIX パスワード・ポリシー属性しかサポートしません。 残りの AIX 属性は、データベース・モジュールによって満たされます。

認証専用の LDAPA モジュールは、以下の属性をサポートします。

- maxage
- minage
- minlen
- lastupdate
- flags
- maxrepeats
- minalpha
- mindiff
- minother
- pwdwarntime
- pwdchecks
- histsize
- histexpire

- time_last_login
- time_last_unsuccessful_login
- tty_last_login
- tty_last_unsuccessful_login
- host_last_login
- host_last_unsuccessful_login
- unsuccessful_login_count
- account_locked
- loginretries
- logintimes

これらの属性はすべての LDAP サーバーでサポートされるわけではありません。 LDAP サーバーがリストされた属性のすべてサポートしていない場合に、サポートされている属性は、このリストとユーザー属性マッピング・ファイルの両方で共通の属性のみです。 マッピング・ファイルは `/etc/security/ldap` ディレクトリーにあります。

AIX スキーマ・サポートがない RFC2307 対応サーバーの場合、以下の AIX 属性がサポートされます。

- maxage
- minage
- lastupdate
- pwdwarntime
- lastupdate

Kerberos バインド**:**

バインド DN と BIND パスワードを使用した単純バインドの他に、**secldapclntd** デーモンは Kerberos V 資格情報を使用したバインドもサポートします。

バインド・プリンシパルの鍵は Keytab ファイルに保管されています。Kerberos バインドを使用するには、**secldapclntd** デーモンがこのキーを使用できることが必要です。 Kerberos バインドを使用可能にすると、**secldapclntd** デーモンは **/etc/security/ldap/ldap.cfg** クライアント構成ファイルに指定されたプリンシパル名と Keytab を使用して LDAP サーバーに対して Kerberos 認証を実行します。 Kerberos バインドを使用すると、**secldapclntd** デーモンは **/etc/security/ldap/ldap.cfg** ファイルに指定されたバインド DN と BIND パスワードを無視します。

Kerberos 認証が正常に実行されると、**secldapclntd** デーモンはバインド資格情報を **/etc/security/ldap/krb5cc_secldapclntd** ディレクトリーに保存します。保存された資格情報は、後で再バインドが行われるときに使用されます。 **secldapclntd** デーモンが LDAP サーバーへの再バインドを試みた時点で資格情報の時間が 1 時間を超えていると、**secldapclntd** デーモンは再初期化して、資格情報を更新します。

Kerberos バインドを使用するように LDAP クライアント・システムを構成するには、バインド DN と BIND パスワードを使用して **mksecldap** コマンドを実行して、クライアントを構成する必要があります。構成が正常に実行されたら、Kerberos 関連の属性に対応する正しい値を指定して **/etc/security/ldap/ldap.cfg** ファイルを編集します。 **secldapclntd** デーモンは、再始動したときに Kerberos バインドを使用します。 構成が正常に行われると、バインド DN と BIND パスワードは使用されなくなります。 これらは **/etc/security/ldap/ldap.cfg** ファイルから安全な方法で除去またはコメント化できます。

Kerberos プリンシパルの作成*:*

Kerberos バインドをサポートするには、鍵配布センター (KDC) に IDS サーバーとクライアントが使用するための 2 つ以上のプリンシパルを作成する必要があります。 最初のプリンシパルは LDAP サーバー・プリンシパルで、第 2 のプリンシパルはクライアント・システムがサーバーにバインドするときに使用するプリンシパルです。

各プリンシパル・キーは必ず Keytab ファイル内に置いて、サーバー・プロセスまたはクライアント・デーモン・プロセスを開始するために使用できるようにしてください。

次に示すものは、IBM ネットワーク認証サービスに基づいた例です。他のソースから Kerberos ソフトウェアをインストールした場合、実際のコマンドはここに示すものと異なる場合があります。

- KDC サーバーで root ユーザーとして kadmin ツールを開始します。

  ```
  #/usr/krb5/sbin/kadmin.local
  kadmin.local:
  ```

- 作成用に ldap/*serverhostname* プリンシパルを作成します。 *serverhostname* は LDAP サーバーを実行する完全修飾 DNS ホストです。

  ```
  kadmin.local: addprinc ldap/plankton.austin.ibm.com
  WARNING: no policy specified for "ldap/plankton.austin.ibm.com@ud3a.austin.ibm.com":
  Re-enter password for principal "ldap/plankton.austin.ibm.com@ud3a.austin.ibm.com":
  Principal "ldap/plankton.austin.ibm.com@ud3a.austin.ibm.com" created.
  kadmin.local:
  ```

- 作成したサーバー・プリンシパルの Keytab を作成します。 このキーはサーバー起動時に LDAP サーバーによって使用されます。 **slapd_krb5.keytab** という Keytab を作成するには、次のように入力します。

  ```
  kadmin.local: ktadd -k /etc/security/slapd_krb5.keytab ldap/plankton.austin.ibm.com
  Entry for principal ldap/plankton.austin.ibm.com with kvno 2,
  encryption type Triple DES cbc mode with HMAC/sha1 added to keytab
  WRFILE:/etc/security/slapd_krb5.keytab.
  Entry for principal ldap/plankton.austin.ibm.com with kvno 2,
  encryption type ArcFour with HMAC/md5 added to keytab WRFILE:/etc/security/slapd_krb5.keytab.
  Entry for principal ldap/plankton.austin.ibm.com with kvno 2,
  encryption type AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab
  WRFILE:/etc/security/slapd_krb5.keytab.
  Entry for principal ldap/plankton.austin.ibm.com with kvno 2,
  encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/security/slapd_krb5.keytab.
  kadmin.local:
  ```

- IDS 管理者用に ldapadmin という名前のプリンシパルを作成します。

  ```
  kadmin.local: addprinc ldapadmin
  WARNING: no policy specified for ldapadmin@ud3a.austin.ibm.com; defaulting to no policy.
  Note that policy may be overridden by ACL restrictions.
  Enter password for principal "ldapadmin@ud3a.austin.ibm.com":
  Re-enter password for principal "ldapadmin@ud3a.austin.ibm.com":
  Principal "ldapadmin@ud3a.austin.ibm.com" created.
  kadmin.local:
  ```

- バインド・プリンシパル **kdapadmin.keytab** 用の Keytab を作成します。このキーは **secldapclntd** クライアント・デーモンが使用できます。

  ```
  kadmin.local: ktadd -k /etc/security/ldapadmin.keytab ldapadmin
  Entry for principal ldapadmin with kvno 2, encryption type
  Triple DES cbc mode with HMCA/sha1 added to keytab WRFILE:/etc/security/ldapadmin.keytab.
  Entry for principal ldapadmin with kvno 2, encryption type
  ArcFour with HMAC/md5 added to keytab WRFILE:/etc/security/ldapadmin.keytab.
  Entry for principal ldapadmin with kvno 2, encryption type
  ```

```
AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/security/ldapadmin.keytab.
Entry for principal ldapadmin with kvno 2, encryption type
DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/security/ldapadmin.keytab.
kadmin.local
```

- クライアントが LDAP サーバーにバインドするための ldapproxy という名前のプリンシパルを作成します。

```
kadmin.local: addprinc ldapproxy
WARNING: no policy specified for ldapproxy @ud3a.austin.ibm.com; defaulting to no policy.
Note that policy may be overridden by ACL restriction
Enter password for principal "ldapproxy@ud3a.austin.ibm.com":
Re-enter password for principal "ldapproxy@ud3a.austin.ibm.com":
Principal "ldapproxy@ud3a.austin.ibm.com" created.
kadmin.local:
```

- バインド・プリンシパル **ldapproxy** 用の *ldapproxy.keytab* という Keytab を作成します。このキーは **secldapclntd** クライアント・デーモンが使用できます。

```
kadmin.local: ktadd -k /etc/security/ldapproxy.keytab ldapproxy
Entry for principal ldapproxy with kvno 2, encryption type
Triple DES cbc mode with HMAC/sh1 added to keytab WRFILE:/etc/security/ldapproxy.keytab.
Entry for principal ldapproxy with kvno 2, encryption type
ArcFour with HMAC/md5 added to keytab WRFILE:/etc/security/ldapproxy.keytab
Entry for principal ldapproxy with kvno 2, encryption type
AES-256 CTS mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/security/ldapproxy.keytab
Entry for principal ldapproxy with kvno 2,
encryption type DES cbc mode with RSA-MD5 added to keytab WRFILE:/etc/security/ldapproxy.keytab.
kadmin.local:
```

***IDS** サーバーの **Kerberos** バインドの使用可能化:*

以下の手順により、Kerberos バインド用に IDS サーバーを使用可能にします。

次の例は、IDS サーバーを Kerberos バインド用に構成する方法を示しています。

この例は IDS v5.1 を使用してテスト済みです。

1. **krb5.client** ファイルセットをインストールします。
2. **/etc/krb5/krb5.conf** ファイルが存在し、正しく構成済みであることを確認します。 このファイルを構成する必要がある場合は、**/usr/sbin/config.krb5** コマンドを実行します。

```
# config.krb5 -r ud3a.austin.ibm.com -d austin.ibm.com -c KDC -s alyssa.austin.ibm.com
Initializing configuration...
Creating /etc/krb5/krb5_cfg_type...
Creating /etc/krb5/krb5.conf...
The command completed successfully.
# cat /etc/krb5/krb5.conf
[libdefaults]
     default_realm = ud3a.austin.ibm.com
     default_keytab_name = FILE:/etc/krb5/krb5.keytab
     default_tkt_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts des-cbc-md5 des-cbc-crc
     defaut_tgs_enctypes = des3-cbc-shal1 arcfour-hmac aes256-cts des-cbc-md5 des-cbc-crc
[realms]
  ud3a.austin.ibm.com = {
     kdc = alyssa.austin.ibm.com:88
     admin_server = alyssa.austin.ibm.com:749
     default_domain = austin.ibm.com
  }

[domain_realm]
     .austin.ibm.com = ud3a.austin.ibm.com
     alyssa.austin.ibm.com = ud3a.austin.ibm.com
```

```
[logging]
     kdc = FILE:/var/krb5/log/krb5
     admin_server = FILE:/var/krb5/log/kadmin.log
     default = FILE:/var/krb5/log/krb5lib.log
```

3. ldap:/*serverhostname* プリンシパルの Keytab ファイルを取得し、**/usr/ldap/etc** ディレクトリーに置きます。例えば、**/usr/ldap/etc/slapd_krb5.keytab** とします。
4. サーバー・プロセスでこのファイルにアクセスできるようにアクセス権を設定します。

```
# chown ldap:ldap/usr/ldap/etc/slapd_krb5.keytab
#
```

5. IDS サーバーを Kerberos バインド用に使用可能にするには、**/etc/ibmslapd.conf** ファイルを編集し、次のエントリーを追加します。

```
dn: cn=Kerberos, cn-Configuration
cn: Kerberos
ibm-slapdKrbAdminDN: ldapadmin
ibm-slapdKrbEnable: true
ibm-slapdKrbIdentityMap: true
ibm-slapdKrbKeyTab: /usr/ldap/etc/slapd_krb5.keytab
ibm-slapdKrbRealm: ud3a.austin.ibm.com
objectclass: ibm-slapdKerberos
objectclass: ibm-slapdconfigEntry
objectclass: top
```

6. ldapproxy プリンシパルを cn-proxyuser,cn=aixdata という名前のバインド DN にマップします。
   a. バインド DN のエントリーが IDS サーバーに存在する場合、次のような内容で **ldapproxy.ldif** という名前のファイルを作成します。

```
dn: cn=proxyuser,cn=aixdata
changetype: modify
add: objectclass
objectclass: ibm-securityidentities
-
add:altsecurityidentities
altsecurityidentities: Kerberos:ldapproxy@ud3a.austin.ibm.com
```

   または

   b. バインド DN のエントリーがまだこのサーバーに追加されていない場合は、次のような内容で **proxyuser.ldif** という名前のファイルを作成します。

注: *proxyuserpwd* はご使用のパスワードに置き換える必要があります。

```
dn: cn=proxyuser,cn=mytest
cn: proxyuser
sn: proxyuser
userpassword: proxyuserpwd
objectclass: person
objectclass: top
objectclass: ibm-securityidentities
altsecurityidentities: Kerberos:ldapproxy@ud3a.austin.ibm.com
```

作成したバインド DN エントリーを、**ldapmodify** コマンドを使用して IDS サーバーに追加します。

```
# ldapmodify -D cn-admin -w adminPwd -f /tmp/proxyuser.ldif modifying entry cn=proxyuser,cn=mytest
#
```

7. IDS サーバーを再始動します。

AIX LDAP クライアントの ***Kerberos*** バインドの使用可能化**:**

LDAP サーバーへの初期バインドで、Kerberos を使用するように AIX LDAP クライアント・システムを構成することができます。

サーバー・ホストがそれ自身に対するクライアントになるには、IDS サーバーをこの方法で構成する必要があります。

この例は IDS v 5.1 を使用してテスト済みです。

1. **krb5.client** ファイルセットをインストールします。
2. **/etc/krb.conf** ファイルが存在し、正しく構成済みであることを確認します。 正しく構成されていない場合は、**/usr/sbin/config.krb5** コマンドを実行してこのファイルを構成してください。
3. バインド・プリンシパルの Keytab ファイルを取得し、**/etc/security/ldap** ディレクトリーに置きます。
4. アクセス権を 600 に設定します。
5. バインド DN と BIND パスワードを使用して **mksecldap** コマンドを実行し、クライアントを構成します。 AIX コマンドが LDAP ユーザーに対して機能することを確認してください。
6. **/etc/security/ldap/ldap.cfg** ファイルを編集して、Kerberos 関連の属性を設定します。 次の例では、バインド・プリンシパルは **ldapproxy** であり、Keytab ファイルは **ldapproxy.keytab** です。 IDS サーバーの管理者特権が必要な場合は、*ldapproxy* を *ldapadmin* に、*ldapproxy.keytab* を *ldapadmin.keytab* にそれぞれ置き換えてください。

   ```
   useKRB5:yes
   krbprincipal:ldapproxy
   krbkeypath:/etc/security/ldap/ldapproxy.keytab
   krbcmddir:/usr/krb5/bin/
   ```

   これで、**secldapclntd** デーモンが Kerberos バインドを使用するようになったため、バインド DN と BIND パスワードを **ldap.cfg** ファイルから除去またはコメント化することができます。
7. **secldapclntd** デーモンを再始動します。
8. これで、**/etc/security/ldap/ldap.cfg** ファイルを他のクライアント・システムに伝搬することができます。

LDAP セキュリティー情報サーバーの監査**:**

SecureWay Directory バージョン 3.2 (およびそれ以降) では、デフォルトのサーバー監査ロギング機能が提供されています。このデフォルトの監査プラグインは、一度使用可能になると、LDAP サーバーのアクティビティーをログ・ファイルに記録します。 このデフォルト監査プラグインについて詳しくは、*Packaging Guide for LPP Installation*の中の LDAP に関する箇所を参照してください。

AIX オペレーティング・システムで提供される LDAP セキュリティー情報サーバー監査機能は、*LDAP セキュリティー監査プラグイン* と呼ばれます。これは、SecureWay Directory のデフォルト監査サービスとは独立しているため、これらの監査サブシステムのいずれかまたは両方を使用可能にすることができます。 AIX 監査プラグインは、LDAP サーバーの AIX セキュリティー情報を更新または照会するイベントだけを記録します。 これは、AIX システム監査の枠組みの中で行われます。

/etc/security/audit/event ファイルには、LDAP に適合した次の監査イベントが入っています。

- `LDAP_Bind`
- `LDAP_Unbind`
- `LDAP_Add`
- `LDAP_Delete`
- `LDAP_Modify`
- `LDAP_Modifydn`

- LDAP_Search

ldapserver 監査クラス定義は、上記のすべてのイベントを含む **/etc/security/audit/config** ファイルにも作成されます。

LDAP セキュリティー情報サーバーを監査するには、**/etc/security/audit/config** ファイル内の各ユーザーのスタンザに次の行を追加します。

```
ldap = ldapserver
```

LDAP セキュリティー情報サーバーの監査プラグインは、AIX システム監査のフレーム内に実装されるので、AIX システム監査サブシステムの一部です。 **audit start** や **audit shutdown** などのシステム監査コマンドを使用して、LDAP セキュリティー情報サーバー監査を使用可能または使用不可にします。 すべての監査レコードは、システム監査証跡に加えられます。監査証跡は、**auditpr** コマンドによって検査することができます。 詳しくは、 149 ページの『監査の概要』を参照してください。

LDAP コマンド**:**

いくつかの LDAP コマンドがあります。

lsldap コマンド

lsldap コマンドは、構成された LDAP サーバーから命名サービス・エンティティーを表示するために使用することができます。 これらのエンティティーは、aliases、 automount、 bootparams、 ethers、 groups、 hosts、 netgroups、 networks、 passwd、 protocols、 rpc、および services です。

mksecldap コマンド

mksecldap コマンドは、IBM SecureWay Directory サーバーをセキュリティー認証およびデータ管理用にセットアップするために使用することができます。 このコマンドは、サーバーとすべてのクライアントで実行しなければなりません。

secldapclntd デーモン

secldapclntd デーモンは、LDAP ロード・モジュールから要求を受け取り、その要求を LDAP セキュリティー情報サーバーに転送し、結果をサーバーから LDAP ロード・モジュールに戻します。

LDAP 管理コマンド*:*

いくつかのコマンドは LDAP 管理に使用されます。

start-secldapclntd コマンド

start-secldapclntd コマンドは、**secldapclntd** デーモンがまだ稼働中でない場合に、このデーモンを始動します。

stop-secldapclntd コマンド

stop-secldapclntd コマンドは、実行中の **secldapclntd** デーモン・プロセスを終了します。

restart-secldapclntd コマンド

restart-secldapclntd スクリプトは、**secldapclntd** デーモンが稼働中の場合にいったん停止させてから再始動します。 **secldapclntd** デーモンが実行されていない場合は、単に始動するだけです。

ls-secldapclntd コマンド

ls-secldapclntd コマンドは、**secldapclntd** デーモンの状況をリストします。

flush-secldapclntd コマンド

flush-secldapclntd コマンドは、**secldapclntd** デーモン・プロセスのキャッシュを消去します。

sectoldif コマンド

sectoldif コマンドは、ローカルで定義されているユーザーおよびグループを読み取り、その結果を表示を **ldif** 形式で標準出力に出力します。

LDAP 属性のマッピング・ファイル・フォーマット*:*

これらのファイルは、AIX 属性名を LDAP 属性名へ変換するために、**/usr/lib/security/LDAP** モジュールおよび **secldapclntd** デーモンによって使用されます。

マッピング・ファイル内の各エントリーは、属性の変換を示しています。 エントリーには、スペースで区切られている、以下の 4 つのフィールドがあります。

```
AIX_Attribute_Name AIX_Attribute_Type LDAP_Attribute_Name LDAP_Value_Type
```

これらのフィールドについて、以下に説明します。

AIX_Attribute_Name
AIX 属性名を指定します。

AIX_Attribute_Type
AIX 属性タイプを指定します。 値は、SEC_CHAR、SEC_INT、SEC_LIST、および SEC_BOOL です。

LDAP_Attribute_Name
LDAP 属性名を指定します。

LDAP_Value_Type
LDAP 値のタイプを指定します。 値は、単一値の場合は **s**、複数値の場合は **m** です。

単一クライアント内の LDAP および KRB5LDAP

LDAP が、例えば KRB5LDAP などのように、複合モジュールの一部である場合、書き込み操作ではなく、読み取り操作のみが実行できます。 ただし、**/usr/lib/security/methods.cfg** ファイル内に以下の構成変更を行うと、例えば KRB5LDAP などの LDAP および複合ロード・モジュールは、以下のステップの実行により単一ファイルに入れることができます。

1. LDAP クライアントと KRB5LDAP クライアントを従来どおりに構成します。
2. **/usr/lib/security/methods.cfg** ファイルを以下のように編集します。

```
LXAP:   program = /usr/lib/security/LDAP program_64
        =/usr/lib/security/LDAP64

LDAP:   program = /usr/lib/security/LDAP program_64
        =/usr/lib/security/LDAP64

NIS:    program = /usr/lib/security/NIS program_64 =
        /usr/lib/security/NIS_64

DCE:    program = /usr/lib/security/DCE
```

```
KRB5:   program = /usr/lib/security/KRB5


KRB5LXAP: options = db=LXAP,auth=KRB5
```

3. デフォルト・スタンザの **/etc/security/user** ファイルを以下のように編集します。

   ```
   SYSTEM = "KRB5LXAP OR LDAP OR compat"
   ```

   LDAP ユーザーは、いつものように処理できます。 以下の例で KRB5LDAP ユーザーの処理を示します。

   ```
   mkuser -R KRB5LXAP <user_name>
   rmuser -R KRB5LXAP <user_name>
   lsuser -R KRB5LXAP <user_name>
   passwd -R KRB5LXAP <user_name>
   ```

EFS Encrypted File System

Encrypted Files System は、システムに存在する個別ユーザーが各自の鍵ストアを通じて、 J2 ファイルシステムにあるデータの暗号化を可能にします。

鍵は各ユーザーに関連付けられます。 これらの鍵は暗号化して保護された鍵ストアに保管され、正常なログインに際して、ユーザーの鍵はカーネルにロードされ、プロセス資格情報と関連付けられます。 その後、プロセスで EFS 保護されたファイルを開く必要がある場合には、これらの資格情報がテストされ、ファイル保護と一致する鍵が検出されれば、このプロセスはファイル鍵を暗号化解除でき、その結果ファイル内容を暗号化解除できます。 鍵管理に基づいたグループもサポートされます。

注**:** EFS はセキュリティー方針全体の一部です。 これは健全なコンピューター・セキュリティー・プラクティスおよび制御と連動するように設計されています。

Encrypted File System 使用可能度

Encrypted File System (EFS) の鍵管理、ファイル暗号化、およびファイル暗号化解除は、通常の操作ではユーザーに認識されません。

EFS は基本 AIX オペレーティング・システムの一部です。 EFS を使用可能にするには、root で (または RBAC **aix.security.efs** 権限を持つユーザー、詳細情報については、「EFS アクター」を参照)、 **efsenable** コマンドを使用して EFS を活動化し、EFS 環境を作成する必要があります。 これは 1 回限りのシステム使用可能化です。 EFS を使用可能にした後は、ユーザーがログインするときに、その鍵と鍵ストアがサイレント方式で作成されて、ユーザーのログイン・パスワードによって保護または暗号化されます。 次に、ユーザーの鍵は、EFS ファイルを暗号化または暗号化解除するときに、 J2 ファイルシステムによってサイレント方式で使用されます。 EFS ファイルはいずれも、その自体の固有ファイル鍵で保護され、このファイル鍵も、やはりファイル許可権に応じて、ファイル所有者またはグループの鍵で保護または暗号化されます。

デフォルトで、J2 ファイルシステムは EFS を使用可能にしていません。 J2 ファイルシステムの EFS が使用可能になっている場合、 J2 ファイルシステムは読み取りおよび書き込み要求に対して、カーネルでの暗号化および暗号化解除を透過的に管理します。 ユーザーおよびグループ管理のコマンドは (**mkgroup**、**chuser**、および **chgroup** など)、ユーザーの鍵ストアおよびグループの鍵ストアを透過的管理します。

以下の EFS コマンドは、ユーザーが自分の鍵およびファイル暗号化を管理できるようにする目的で提供されます。

efskeymgr
: 鍵を管理する

efsmgr

files/directories/file システムの暗号化を管理する

Encrypted File System アクター

EFS キーを管理および使用できるユーザーには、次の 3 つのタイプがあります。

ルートとしての全アクセス権限または制限付きアクセス権限**:**

鍵に対するルート・アクセス権限は、無制限または制限付きとすることができます。 いずれのモードであっても、ルートはユーザーに対する **su** だけで、ユーザーの暗号化ファイルまたは鍵ストアへのアクセス権限を取得することはできません。

あるモードでは、ルートはユーザーの鍵ストア・パスワードを再設定して、この鍵ストア内のユーザーの鍵にアクセスすることができます。 このモードでは、より柔軟にシステム管理を行うことができます。

別のモードでは、ルートはユーザーのログオン・パスワードを再設定することができ、ユーザーの鍵ストア・パスワードの再設定はできません。 ルートは、ユーザーの置換 (**su** コマンドを使用して) およびオープン鍵ストアの継承を行うことはできません。 ルートはユーザーとグループ、およびそれらに関連する鍵ストアの作成および削除を行うことができますが、これらの鍵ストア内の鍵へのアクセスはできません。 このモードでは、悪意のあるルートからのアタックに対してより高レベルの保護が可能になります。

鍵ストアの管理および使用のためのモードとして、Root Admin と Root Guard の 2 つのモードがあります。 EFS 管理鍵も提供されます。

EFS 管理鍵によって、Root Admin モードのすべての鍵ストアにアクセスしてパスワードを再設定することが可能になります。 この鍵は、**efs_admin** 特殊鍵ストアにあります。 **efs_admin** 特殊鍵ストアへのアクセスは、許可されたユーザー (インストール時の root ユーザーおよびセキュリティー・グループ、または RBAC **aix.security.efs** 許可) にのみ与えられます。

鍵ストアが Root Guard モードである場合は、この鍵ストアに含まれている鍵は、正しい鍵ストア・パスワードを用いずに取得することはできません。 このことは悪意のあるルートに対して強力なセキュリティーになりますが、問題もあります。すなわち、ユーザーがパスワードを忘れた場合には、鍵ストア内の鍵を解放せずにパスワードを再生成する方法がなく、結果としてユーザーはデータにアクセスすることができなくなります。 この鍵ストア・モードでは、いくつかの操作は直ちに処理することができず、保留操作としてスケジュールに入れられます。 このような保留操作は、ユーザー鍵ストア内のグループ・アクセス・キーの追加または抑止、あるいは秘密鍵の再生成のような場合に生成されます。 このような保留操作は、鍵ストアの所有者によって管理されます。

efs_admin 管理鍵**:**

efs_admin 鍵ストアには、root admin モード (デフォルト・モード) で、ユーザーまたはグループの鍵ストアを開くことができる特殊鍵が含まれています。

この特殊鍵ストアを開くためのパスワードは、EFS が活動化されている場合には root ユーザーおよびセキュリティー・グループの鍵ストアに保管されます。 このパスワードはその他のグループに指定することも、または **efskeymgr** コマンドで除去することもできます。この鍵は RBAC **aix.security.efs** 権限と連動して、ユーザーの EFS 管理を許可します (言い換えれば、root admin モードで鍵ストアをアクセスします)。

efs_admin RBAC の考慮事項

「Role Based Access Control (ロール・ベースのアクセス制御)」が使用可能にされたシステムでは、**efs_admin** コマンドは **aix.security.efs** 権限により保護されます。

ユーザー鍵ストア**:**

ユーザー鍵ストアにより、最も一般的な操作が自動的に管理されます。 **efskeymgr** コマンドは、保守タスクおよび高度な EFS 使用に対して使用されます。 ユーザーは **efsmgr** コマンドにより暗号化ファイルおよびディレクトリーを作成できます。鍵ストア管理はほとんどのユーザー管理コマンドに組み込まれています。 ユーザーがグループに追加されると、そのユーザーはグループの鍵ストアへのアクセス権限を自動的に所有することになります。

EFS アクセス権限をもつファイル所有者は **efsmgr** コマンドを使用して、他のユーザーおよびグループに EFS アクセス権限を認可します (ファイル所有者が UNIX の ACL を使用して制御する場合に類似しています)。ユーザーはオープン鍵ストアをもつ同じユーザー ID のもとで実行している別のプロセスに影響を及ぼさないで、自分のパスワードを変更できます。

Encrypted File System 鍵ストア

鍵ストアはパスワードで保護されます。 ユーザーはログイン・パスワードとは異なる代替の鍵ストア・パスワードを選択できます。 このケースでは、鍵ストアは開かれなくて、ユーザーの標準ログインのときに使用可能になります。 その代わりに、ユーザーは **efskey** コマンドを使用して鍵ストアを手動でロードして、鍵ストア・パスワードを提供する必要があります。

鍵ストアのフォーマットは **PKCS # 12** です。 鍵ストアは以下のファイルに保管されます。

ユーザー鍵ストア
/var/efs/users//keystore

グループ鍵ストア
/var/efs/groups//keystore

efsadmin 鍵ストア
/var/efs/efs_admin/keystore

ユーザーがログオン・パスワードおよび鍵ストア・パスワードを同じパスワードに設定すると、鍵ストアはログインのときに開かれて使用可能になります。

ユーザーは EFS **efskeymgr** コマンドを使用して、暗号化アルゴリズムのタイプと鍵の長さを選択できます。

鍵ストアへのアクセスはいずれの子プロセスにも継承されます。

グループ・ベースの鍵管理もサポートされます。 グループ鍵ストアがガード・モードの場合、グループ・メンバーだけがグループ鍵をメンバーの鍵ストアに追加したり、メンバーの鍵ストアから除去したりすることができます。 ユーザー鍵ストアには、グループの秘密鍵が収められているユーザーのグループ鍵ストアを開くためのユーザーの秘密鍵が収められており、パスワードも収められています。

注**:** ユーザーの鍵ストア・パスワードがログイン・パスワードと一致する場合のみ、 EFS 鍵ストアは標準 AIX ログインの一部として開かれます。 これはユーザーの鍵ストアを最初に作成するときに、デフォルトでセットアップされます。 標準 AIX ログイン以外のログイン方式 (例えば、ロード可能な認証モジュールおよびプラグ可能な認証モジュール) は、鍵ストアを自動的に開くことができません。

暗号化および継承

EFS は J2 のフィーチャーです。 ファイルシステムの **efs** オプションは **yes** に設定する必要があります (『**mkfs** コマンド』および『**chfs** コマンド』を参照してください)。

J2 EFS はユーザー・データを自動的に暗号化および暗号化解除します。 ただし、ユーザーに EFS 活動化ファイルへの読み取り権限を持っていて、正当な鍵を持っていない場合、ユーザーは通常の方法でファイルを読み取ることができません。言い換えれば、ユーザーが有効な鍵を持っていない場合は、データを暗号化解除することが不可能です。

すべての暗号機能は CLiC カーネル・サービスおよび CLiC ユーザー・ライブラリーの機能を利用しています。

デフォルトで、J2 ファイルシステムは EFS を使用可能にしていません。 J2 ファイルシステムは、ファイルシステム EFS 継承の活動化を可能にする前か、またはユーザー・データの EFS 暗号化の開始を可能にする前に、EFS を使用可能にする必要があります。 ファイルは暗号化ファイルとして、明示的に **efsmgr** コマンドで作成されるか、または EFS 継承を経由して暗黙的に作成されます。 EFS 継承を活動化することができるのは、ファイル・システム・レベル、ディレクトリー・レベル、またはその両方のレベルです。

ls コマンドは先頭に **e** の付いた暗号化ファイルのエントリーをリストします。

cp および **mv** コマンドは、「EFS から EFS」シナリオおよび「EFS から非 EFS」シナリオ全体にわたって、メタデータと暗号化データを継ぎ目なく処理することができます。

backup コマンド、**restore** コマンド、および **tar** コマンドと関連コマンドは、暗号化および暗号化解除に使用される EFS メタデータを含めて、暗号化データのバックアップおよび復元を行うことができます。

バックアップおよび復元

アーカイブ済み EFS ファイルに関連付けられている鍵ストアのアーカイブまたはバックアップを適切に管理することは重要なことです。 アーカイブ済み鍵ストアまたはバックアップ鍵ストアに関連付けられた鍵ストア・パスワードの管理および保守も行う必要があります。これらのタスクのいずれかに失敗すると、データ損失の原因となります。

EFS 暗号化ファイルのバックアップ時に、**–Z** オプションを指定した **backup** コマンドの使用により、ファイルの暗号化形式をファイルの暗号メタデータと共にバックアップすることができます。 ファイル・データとメタデータの両方とも強い暗号化によって保護されます。 この方法には、強い暗号化によってバックアップ・ファイルが保護されるというセキュリティー上の利点があります。 バックアップに使用されるファイルに関連付けられているファイル所有者およびグループの鍵ストアをバックアップする必要があります。 これらの鍵ストアは、次のファイルに入っています。

users keystores
/var/efs/users/*user_login*/*

group keystore
/var/efs/groups//keystore

efsadmin keystore
/var/efs/efs_admin/keystore

EFS バックアップ (**backup** コマンドに **–Z** オプションを指定して作成) を復元するには、 **restore** コマンドを使用します。 restore コマンドは暗号メタデータも必ず復元します。 復元処理の時には、ユーザーが個別の鍵ストアにおいてキーを変更していない場合は、バックアップ鍵ストアを復元する必要はありませ

ん。 ユーザーが鍵ストアをオープンするためにパスワードを変更するときに、鍵ストアの内部鍵は変更されません。 鍵ストアの内部鍵を変更する場合は、**efskeymgr** コマンドを使用します。

ユーザーの鍵ストアの内部鍵が同じままである場合には、ユーザーは現行の鍵ストアを使用して直ちに復元済みファイルのオープンおよび暗号化解除を行うことができます。ただし、ユーザーの鍵ストア内にある鍵が変更された場合は、バックアップ・ファイルに関連付けられてバックアップされた鍵ストアをオープンしなければなりません。 この鍵ストアは **efskeymgr –o** コマンドを使用してオープンすることができます。 **efskeymgr** コマンドにより、鍵ストアをオープンするためのパスワードに関するプロンプトが出されます。 このパスワードは、バックアップの時に鍵ストアに関連付けて使用されたものです。

例えば、ユーザー Bob の鍵ストアがパスワード **foo** (パスワード「foo」は保護パスワードではなく、簡素化のためにこの例で使用しているだけです) によって保護されていて、Bob の暗号化ファイルのバックアップが Bob の鍵ストアと一緒に 1 月に実行されたと仮定します。 この例では、Bob は **foo** を AIX のログイン・パスワードとしても使用しています。 2 月に、Bob はパスワードを **bar** に変更し、これによって **bar** に対する鍵ストア・アクセス・パスワードも変更されました。 3 月に Bob の EFS ファイルが復元され、Bob がそのファイルを現行の鍵ストアとパスワードを用いてオープンして表示することができたとすると、それは Bob が鍵ストアの内部鍵を変更しなかったためです。

ただし、Bob の鍵ストアの内部鍵を変更する必要があった場合には (**efskeymgr** コマンドを用いて)、デフォルトでは旧鍵ストアの内部鍵は廃棄されないで Bob の鍵ストアに当面残されます。 ユーザーがファイルをアクセスするときに、EFS は復元されたファイルが旧内部鍵を使用したことを自動的に認識し、さらに、EFS は廃棄予定の鍵を使用してファイルを暗号化解除します。 この同じインスタンスをアクセスする際に、EFS は新しい内部鍵を使用してファイルを変換します。 パフォーマンスについては、すべてが鍵ストアとファイルの暗号メタデータを介して処理され、ファイル・データを再暗号化する必要がありませんから、大きな影響はありません。

廃棄される予定の内部鍵を **efskeymgr** で除去する場合は、旧内部鍵を含む旧鍵ストアを復元して、この内部鍵で暗号化されたファイルと同時に使用する必要があります。

ここで、古いパスワードを安全に保守およびアーカイブするにはどうしたらよいかという疑問が生じます。パスワードをアーカイブする方法とツールがあります。 通常、このような方法では、すべての古いパスワードのリストを入れたファイルを作成し、このファイルを暗号化し、現行の鍵ストアで保護し、同様にその鍵ストアを現行のパスワードで保護するということを行います。 しかし、IT 環境およびセキュリティー・ポリシーは組織によって異なり、ご使用の環境に最も適したセキュリティー・ポリシーおよび実践を発展させるために、それぞれの組織に必要な特定のセキュリティー要件をよく考慮しなければなりません。

J2 EFS 内部メカニズム

それぞれの J2 EFS 活動ファイルは、特殊な拡張属性と関連付けられています。この属性には暗号権限を検証するために使用される EFS メタデータ、およびファイルの暗号化と暗号化解除を行うために使用される情報 (鍵、暗号アルゴリズムなど) が含まれます。

EA コンテンツは J2 には不透明です。 暗号権限 (アクセス制御) を判別するには、いずれの指定された EFS 活動ファイルにも、ユーザー資格情報と EFS メタデータの両方が必要です。

注**:** ファイルまたはデータが失われる恐れのある状態 (例えば、ファイルの EA の除去) では、特に注意を払う必要があります。

EFS 保護の継承

ディレクトリーが EFS 活動状態になると、新規に作成された直接の子は、手動でオーバーライドされない限り、自動的に EFS 活動状態になります。 親ディレクトリーの EFS 属性は、ファイルシステムの EFS 属性より優先されます。

ディレクトリーの継承の有効範囲は、1 レベルだけです。 親ディレクトリーが EFS 活動状態になっていれば、新規に作成された子も親の EFS 属性を継承します。 既存の子は、自分の現在の暗号化または非暗号化状態を維持します。 親が自分の EFS 属性を変更すると、論理継承チェーンは壊れます。そのような変更はそのディレクトリーの既存の子には伝搬されず、それらのディレクトリーには別途、適用する必要があります。

ワークロード・パーティションの考慮事項

Encrypted File System をワークロード・パーティション内で使用可能にするか、または使用する前に、まず、グローバル・システムで **efsenable** コマンドを使用して EFS を使用可能にする必要があります。この使用可能化の操作は 1 回のみ実行します。 そして、EFS が使用可能なファイルシステムを含むすべてのファイルシステムをグローバル・システムから作成する必要があります。

Encrypted File System のセットアップ

これは最初に行う作業です。

ステージを以下のように設定する必要があります。

1. **clic.rte** ファイルセットをインストールします。 このファイルセットには、EFS で必要になる暗号ライブラリーおよびカーネル・エクステンションが含まれます。 **clic.rte** ファイルセットは AIX 拡張パックにあります。
2. システム上の EFS を **efsenable** コマンドで使用可能にします (例えば、>efsenable –a)。 パスワードについてプロンプトが出されたら、root パスワードの使用をお勧めします。 ユーザー鍵ストアが自動的に作成されてから、**efsenable** コマンドの実行後にユーザーのログイン、また再ログインを行います。 一度、システムで **efsenable –a** を実行すると、システムは EFS が使用可能になり、再度 **efsenable** コマンドを実行する必要はありません。
3. **–a efs=yes** オプションを指定して、EFS が使用可能なファイルシステムを作成します。例えば、`crfs -v jfs2 -m /foo –A yes -a efs=yes -g rootvg -a size=20000`
4. ファイルシステムをマウントして、EFS が使用可能なファイルシステムで暗号継承をオンにします。これを行うには、**efsmgr** コマンドを実行します。ファイルシステム **/foo** が作成された上記の例を続行するには、`efsmgr –s –E /foo` コマンドを実行します。 これにより、このファイルシステム内で作成されて使用されるファイルは、すべて暗号化ファイルになります。

これ以降、オープン鍵ストアをもつユーザーまたはプロセスがこのファイルシステムでファイルを作成すると、そのファイルは暗号化されます。 ユーザーまたはプロセスがファイルを読み取ると、ファイルへのアクセス許可のあるユーザーに対しては、ファイルは自動的に暗号化解除されます。

詳細情報については、以下を参照してください。

- **chfs**、**chgroup**、**chuser**、**cp**、**efsenable**、 **efskeymgr**、**efsmgr**、**lsuser**、**ls**、**mkgroup**、 **mkuser**、および **mv** コマンド
- `/etc/security/group` ファイルおよび `/etc/security/user` ファイル

Encrypted File System 鍵ストアへのリモート・アクセス

エンタープライズ環境では、Encrypted File System (EFS) 鍵ストアを集中管理できます。 鍵ストアを制御するデータベースをシステムごとに別々に保管すると、鍵ストアの管理が困難になることがあります。 AIX Centralized EFS Keystore では、ユーザーおよびグループの鍵ストア・データベースを Lightweight Directory Access Protocol (LDAP) に保管できるので、EFS 鍵ストアを集中管理できます。

関連概念:

170 ページの『Lightweight Directory Access Protocol (LDAP)』
Lightweight Directory Access Protocol (LDAP) は、クライアント/サーバー・モデルにおいて、ローカルまたはリモートからディレクトリー (データベース) 内の情報をアクセスおよび更新するための標準的な方式を定義します。

Encrypted File System 鍵ストアへのリモート・アクセスの概要**:**

Encrypted File System (EFS) データベース、EFS コマンドの LDAP 使用可能化、および固有の鍵ストア・アクセスについて説明します。

すべての AIX EFS 鍵ストア・データベースを LDAP に保管できます。LDAP には以下の EFS データベースが入っています。

- ユーザー鍵ストア
- グループ鍵ストア
- 管理者鍵ストア
- Cookie

AIX オペレーティング・システムでは、以下の管理タスクの実行に役立つユーティリティーが提供されています。

- ローカル鍵ストア・データの LDAP サーバーへのエクスポート
- EFS 鍵ストア・データを LDAP で使用するためのクライアント構成
- EFS 鍵ストア・データへのアクセス制御
- クライアント・システムからの LDAP データの管理

LDAP 鍵ストア・データベースを使用するために、すべての EFS 鍵ストア・データベース管理コマンドが使用可能にされます。 `/etc/nscontrol.conf` ファイルにシステム全体の検索順序が指定されていない場合、鍵ストア操作はユーザーまたはグループの efs_keystore_access 属性によって決まります。 efs_keystore_access を LDAP に設定すると、EFS コマンドは LDAP 鍵ストア上で鍵ストア操作を実行します。

以下の表は、LDAP 用の EFS コマンドの変更の説明です

表 *12. LDAP* 用の *EFS* コマンドの使用可能化

コマンド	**LDAP** 情報
任意の EFS コマンド	LDAP に efs_keystore_access 属性を設定する場合、LDAP で鍵ストア操作を実行するためには、どのコマンドを使用する場合も特別なオプション -L *domain* を指定する必要はありません。
efskeymgr	LDAP で明示的に鍵ストア操作を実行できるように、-L load_module オプションを組み込んでいます。

表 12. *LDAP* 用の *EFS* コマンドの使用可能化 (続き)

コマンド	**LDAP** 情報
efsenable	LDAP で初期セットアップを実行して EFS 鍵ストアを収容できるように、-d Basedn オプションを組み込んでいます。初期セットアップでは、EFS 鍵ストア用の基本識別名 (DN) の追加、およびローカル・ディレクトリー構造 (/var/efs/) の作成が含まれています。
efskstoldif	以下のローカル・システムのデータベースから LDAP 用の EFS 鍵ストア・データを生成します。 • /var/efs/users/*username*/keystore • /var/efs/groups/*groupname*/keystore • /var/efs/efs_admin/keystore • すべての鍵ストアに対する Cookie (存在する場合)

すべての鍵ストア・エントリーは固有でなければなりません。 各鍵ストア・エントリーは、ユーザー名またはグループ名を含むエントリーの DN に直接対応しています。 システムは、ユーザー ID (uidNumber)、グループ ID (gidNumber)、および DN を照会します。 ユーザー名およびグループ名が対応する DN と一致すると、照会は成功します。 LDAP で EFS 鍵ストア・エントリーを作成または移行する前に、システム上のユーザー名とグループ名および ID が固有になるようにしてください。

関連タスク:

『Encrypted File System 鍵ストア・データの LDAP へのエクスポート』
Encrypted File System (EFS) 鍵ストア用に中央管理されたリポジトリーを LDAP として使用するには、LDAP サーバーに鍵ストア・データを追加する必要があります。

200 ページの『Encrypted File System 鍵ストア用の LDAP クライアントの構成』
LDAP に保管された Encrypted File System (EFS) 鍵ストア・データを使用するには、LDAP クライアントとしてシステムを構成する必要があります。

Encrypted File System 鍵ストア・データの **LDAP** へのエクスポート**:**

Encrypted File System (EFS) 鍵ストア用に中央管理されたリポジトリーを LDAP として使用するには、LDAP サーバーに鍵ストア・データを追加する必要があります。

LDAP で EFS 鍵ストア・エントリーを作成または移行する前に、システム上のユーザー名とグループ名および ID が固有になるようにしてください。

LDAP サーバーに EFS 鍵ストア・データを追加するには、以下のステップを実行します。

1. 次のようにして、LDAP 用の EFS 鍵ストア・スキーマを LDAP サーバーにインストールします。
   a. AIX システム上で `/etc/security/ldap/sec.ldif` ファイルから LDAP 用の EFS 鍵ストア・スキーマを取得します。
   b. **ldapmodify** コマンドを実行して、LDAP 用の EFS 鍵ストア・スキーマで、LDAP サーバーのスキーマを更新します。
2. **efskstoldif** コマンドを実行して、ローカルの EFS 鍵ストア・ファイルでデータを読み取り、そのデータを LDAP に対応した形式で出力します。 固有の鍵ストア・アクセスを維持するには、LDAP に存在する EFS 鍵ストア・データを、ユーザー・データおよびグループ・データとして同じ親識別名 (DN) の下に置くことを考慮してください。
3. データをファイルに保存します。
4. **ldapadd -b** コマンドを実行して、LDAP サーバーに鍵ストア・データを追加します。

関連概念:

198 ページの『Encrypted File System 鍵ストアへのリモート・アクセスの概要』
Encrypted File System (EFS) データベース、EFS コマンドの LDAP 使用可能化、および固有の鍵ストア・アクセスについて説明します。

Encrypted File System 鍵ストア用の **LDAP** クライアントの構成**:**

LDAP に保管された Encrypted File System (EFS) 鍵ストア・データを使用するには、LDAP クライアントとしてシステムを構成する必要があります。

EFS 鍵ストア用に LDAP クライアントを構成するには、以下のステップを実行します。

1. システムを LDAP クライアントとして構成する場合は、 **/usr/sbin/mkseccldap** コマンドを実行します。 **mkseccldap** コマンドは指定された LDAP サーバーを動的に検索して、 EFS 鍵ストア・データの場所を判別します。 次に、その結果を `/etc/security/ldap/ldap.cfg` ファイルに保存します。 **mkseccldap** コマンドは、ユーザー、グループ、管理者、および efscookies の鍵ストア・データの場所を判別します。
2. 以下のステップのいずれかを実行して、LDAP を EFS 鍵ストア・データのルックアップ・ドメインとして使用可能にします。
   - ユーザーまたはグループの **efs_keystore_access** 属性を **file** または **ldap** に設定します。
   - `/etc/nscontrol.conf` ファイルを使用して、鍵ストアの検索順序をシステム・レベルで定義します。 以下の表に例を示します。

表 *13. /etc/nscontrol.conf* ファイルの構成例

属性	説明	検索順序 **(secorder)**
efsusrkeystore	この検索順序はすべてのユーザーに共通です。	LDAP, files
efsgrpkeystore	この検索順序はすべてのグループに共通です。	files, LDAP
efsadmkeystore	この検索順序ではすべてのターゲット鍵ストアの管理者鍵ストアの場所を探索します。	LDAP, files

> 重要: `/etc/nscontrol.conf` ファイルで定義された構成は、ユーザーおよびグループの **efs_keystore_access** 属性に対して設定されたすべての値を指定変更します。ユーザーの **efs_adminks_access** 属性についても同じことが言えます。

あるシステムを LDAP クライアントとして構成し、LDAP を EFS 鍵ストア・データのルックアップ・ドメインとして使用可能にすると、LDAP 鍵ストア操作を実行するときは必ず、/usr/sbin/secldapclntd クライアント・デーモンが LDAP サーバーから EFS 鍵ストア・データを取得します。

関連概念:

198 ページの『Encrypted File System 鍵ストアへのリモート・アクセスの概要』
Encrypted File System (EFS) データベース、EFS コマンドの LDAP 使用可能化、および固有の鍵ストア・アクセスについて説明します。

Public Key Cryptography Standard #11

Public Key Cryptography Standard #11 (PKCS #11) サブシステムは、ハードウェア・デバイスのタイプにかかわらずアプリケーションからそのハードウェア・デバイス (トークン) にアクセスする方法を提供します。

このセクションの内容は、PKCS #11 標準のバージョン 2.20 に準拠しています。

PKCS #11 サブシステムは、以下のコンポーネントを使用します。

- API 共有オブジェクト (/usr/lib/pkcs11/ibm_pks11.so) は、PKCS #11 標準をサポートするデバイス・ドライバーへの汎用インターフェースとして提供されています。 このように階層化された設計により、新規の PKCS #11 デバイスが使用可能になったときには、既存のアプリケーションを再コンパイルしなくてもそれらのデバイスが有効になります。
- PKCS #11 デバイス・ドライバー。これは、Encrypted File System (EFS) や IP Security (IPSec) などの他のカーネル・コンポーネントに提供されている機能と類似した機能をアプリケーションに提供します。
- プラットフォームが暗号化コプロセッサー機能をサポートする場合、PKCS #11 デバイス・ドライバーは Advanced Encryption Standard (AES)、 Secure Hash Algorithm (SHA)、およびハッシュ・メッセージ認証コード (HMAC) の操作で使用可能なハードウェア・アクセラレーションを使用します。 パフォーマンスを向上させるために、ネットワーク・メモリー・アフィニティーを有効にすることができます。

関連情報:

AIX メモリー・アフィニティーのサポート

IBM 4758 モデル 2 暗号化コプロセッサー

IBM 4758 モデル 2 暗号化コプロセッサーは、セキュアなコンピューティング環境を提供します。

PKCS #11 サブシステムを構成する前に、サポートされているマイクロコードによってアダプターが適切に構成されていることを確認してください。

IBM 4960 Cryptographic Accelerator

IBM 4960 Cryptographic Accelerator は、暗号トランザクションをオフロードする手段を提供します。 PKCS #11 サブシステムを構成する前に、アダプターが適切に構成されていることを確認してください。

Public Key Cryptography Standard #11 サブシステムと併用するための **IBM 4758** モデル **2** 暗号化コプロセッサーの検査**:**

PKCS #11 サブシステムは、インストール中およびリブート時の PKCS #11 呼び出しをサポートできるアダプターを自動検出するように設計されています。 したがって、適切に構成されていない IBM 4758 モデル 2 暗号化コプロセッサーは、PKCS #11 インターフェースからアクセスできず、アダプターに送信される呼び出しは失敗します。

アダプターが正しくセットアップされているかどうかを検査するには、以下の手順を実行します。

1. 次のコマンドをタイプ入力して、アダプターのソフトウェアが適切にインストールされているかどうかを確認します。

   ```
   lsdev -Cc adapter | grep crypt
   ```

   IBM 4758 モデル 2 暗号化コプロセッサーが結果リストに含まれていない場合は、カードが正しく装着されているかどうか、およびサポート・ソフトウェアが正しくインストールされているかどうかをチェックしてください。
2. 次のコマンドをタイプ入力して、適切なファームウェアがカードにロードされているかどうかを確認します。

   ```
   csufclu /tmp/l ST device_number_minor
   ```

セグメント 3 イメージに PKCS #11 アプリケーションがロードされていることを検査します。 ロードされていない場合は、アダプター専用のドキュメンテーションを参照して、最新のマイクロコードとインストール手順を入手してください。

注: このユーティリティーを使用できない場合は、サポート・ソフトウェアがインストールされていません。

Public Key Cryptography Standards #11 サブシステムと併用するための **IBM 4960 Model 2 Cryptographic Accelerator** の検査:

PKCS #11 サブシステムは、インストール中およびリブート時の PKCS #11 呼び出しをサポートできるアダプターを自動検出するように設計されています。 したがって、適切に構成されていない IBM 4960 Cryptographic Accelerator は、PKCS #11 インターフェースからアクセスできず、アダプターに送信される呼び出しは失敗します。

次のコマンドを入力して、アダプターのソフトウェアが適切にインストールされているかどうかを確認します。

```
lsdev -Cc adapter | grep ica
```

IBM 4960 Cryptographic Accelerator が結果リストに含まれていない場合は、カードが正しく装着されており、サポートするデバイス・ドライバーが正しくインストールされていることを確認してください。

Public Key Cryptography Standard #11 サブシステムの構成

PKCS #11 サブシステムは、PKCS #11 をサポートしているデバイスを自動的に検出します。 ただし、複数のアプリケーションでこれらのデバイスを使用するためには、何らかの初期セットアップを実行する必要があります。

これらの手順は API を使用して (PKCS #11 アプリケーションを作成して)、または SMIT インターフェースを使用して実行できます。 PKCS #11 SMIT オプションには、メイン SMIT メニューの「**Manage the PKCS11 subsystem (PKCS11** サブシステムの管理**)**」、または **smit pkcs11** 高速パスのいずれかを使用してアクセスします。

トークンの初期化:

各アダプターまたは PKCS #11 トークンを問題なく使用できるようにするには、事前に初期化する必要があります。

この初期化手順では、トークンに固有ラベルを設定します。 このラベルを使用して、アプリケーションはトークンを一意的に識別できます。したがって、ラベルを重複して使用することはできません。 ただし、API はラベルが再使用されていないかどうかは検査しません。 この初期化は、PKCS #11 アプリケーションを使用して実行するか、またはシステム管理者が SMIT を使用して実行することができます。 トークンにセキュリティー担当者 PIN がある場合は、デフォルト値が 87654321 に設定されます。 PKCS #11 サブシステムのセキュリティーを確保するために、初期化後にこの値を変更する必要があります。

トークンを初期化するには、以下の手順を実行します。

1. `smit pkcs11` と入力して、トークン管理画面を表示します。
2. 「**Initialize a Token (**トークンの初期化**)**」を選択します。
3. サポートされているアダプターのリストから PKCS #11 アダプターを選択します。
4. Enter キーを押して、選択を確認します。

注**:** これで、このトークンの情報がすべて消去されます。

5. セキュリティー担当者 PIN (SO PIN) および固有のトークン・ラベルを入力します。

正しい PIN を入力した場合は、コマンド実行終了後に、アダプターが初期化または最初期化されます。

セキュリティー担当者 **PIN** の設定**:**

SO PIN をデフォルト値から変更するには、次のステップを実行します。

PIN をデフォルト値から変更するには、次のステップを実行します。

1. `smit pkcs11` と入力します。
2. 「**Set the Security Officer PIN (**セキュリティー担当者の **PIN** の設定**)**」を選択します。
3. PIN を設定する初期化済みアダプターを選択します。
4. 現行 PIN と新規 PIN を入力します。
5. 新規の PIN を検査します。

ユーザー **PIN** の初期化**:**

トークンを初期化した後に、アプリケーションからトークン・オブジェクトにアクセスできるようにするために、ユーザー PIN を設定する必要がある場合があります。

オブジェクトにアクセスする前にデバイスがユーザー・ログインを必要としているかについては、デバイス特定のドキュメンテーションを参照してください。

ユーザー PIN を初期化するには、以下の手順を実行します。

1. `smit pkcs11` と入力して、トークン管理画面を表示します。
2. 「**Initialize the User PIN (**ユーザー **PIN** の初期化**)**」を選択します。
3. サポートされているアダプターのリストから PKCS #11 アダプターを選択します。
4. SO PIN およびユーザー PIN を入力します。
5. ユーザー PIN を検査します。
6. 検査の終了後に、ユーザー PIN を変更する必要があります。

ユーザー ***PIN*** のリセット**:**

ユーザー PIN をリセットする場合は、SO PIN を使用して PIN 再初期化するか、または既存のユーザー PIN を使用して、ユーザー PIN を設定します。

PIN をリセットするには、次のようにします。

1. `smit pkcs11` と入力して、トークン管理画面を表示します。
2. 「**Set the User PIN (**ユーザー **PIN** の設定**)**」を選択します。
3. ユーザー PIN を設定する初期化済みアダプターを選択します。
4. 現在のユーザー PIN および新規の PIN を入力します。
5. 新規のユーザー PIN を検査します。

Public Key Cryptography Standard #11 の使用法

アプリケーションから PKCS #11 サブシステムを使用するには、サブシステムのスロット・マネージャー・デーモンが実行されていて、アプリケーションを API の共有オブジェクトにロードする必要があります。

スロット・マネージャーは、通常、**/etc/rc.pkcs11** スクリプトを呼び出す **inittab** によって、ブート時に開始されます。 このスクリプトは、スロット・マネージャー・デーモンを開始する前にシステムのアダプターを検査します。 したがって、ユーザーがシステムにログオンするまでは、スロット・マネージャー・デーモンを使用できません。 デーモンの開始後は、システム管理者が介入しなくても、サポートされているアダプターの数やアダプター・タイプの変更がサブシステム内で行われます。

API をロードするには、実行時にオブジェクト内でリンクするか、または据え置かれたシンボル解決を使用します。 例えば、アプリケーションは、以下の方法で PKCS #11 機能リストを取得することができます。

```
d CK_RV (*pf_init)();
void *d;
CK_FUNCTION_LIST *functs;

d = dlopen(e, RTLD_NOW);
if ( d == NULL ) {
   return FALSE;
}

pfoo = (CK_RV (*)())dlsym(d, "C_GetFunctionList");
if (pfoo == NULL) {
   return FALSE;
}

rc = pf_init(&functs);
```

Public Key Cryptography Standard #11 ツール

AIX オペレーティング・システム内部で暗号システムの管理に 2 つのツール、PKCS #11 Key Management ツールおよび PKCS #11 Administration ツールが使用可能です。 カーソル・ベースの GUI またはコマンド・ライン・インターフェースの使用により、これらのツールにアクセスできます。

注**:** AIX Cryptographic Framework ツールのユーザー補助には、バッチ処理機能の使用が必須です。 アクセシビリティーに対するバッチ処理機能の使用の詳細は、 206 ページの『バッチ処理』を参照してください。

PKCS #11 Key Management ツールは、AIX オペレーティング・システムの鍵、証明書、および PKCS #11 データの管理の集中型ツールです。 このツールにより管理されるオブジェクトは、例えば、暗号化アダプターの IBM ファミリー (例えば、IBM 4758、4960、および 4764) などの PKCS #11 プロバイダー、または AIX Cryptographic Framework のいずれかに保管されます。 PKCS #11 Key Management ツールの使用により種々の操作を実行できます。これらの操作には、PKCS #10 Certificate Signing Request (CSR) または自己署名証明書が含まれます。 更に、このツールを使用して、PKCS #11 トークン間の PKCS #11 オブジェクト・データの移送と同様に、PKCS #11 オブジェクト・データの探索、表示、削除、インポート、エクスポート、およびバックアップができます。 **p11km** コマンドの実行によりこのツールの GUI バージョンを開始できます。 このツールはすべての使用可能な PKCS #11 トークンをロードします。 これらのトークンの詳細は、矢印キーを使用してトークンのリストをスクロールアップまたはスクロールダウンすることにより表示できます。 トークンを選択するには、矢印キーを使用してトークンを強調表示し、Enter キーを押します。このツールのコマンド・ライン・バージョンは、次のコマンドの実行により開始できます。

```
p11km -b <batchfile>
```

PKCS #11 Administration ツールは、AIX PKCS #11 Cryptographic Framework の管理の集中型ツールです。 このツールにより、管理者またはセキュリティー担当者は、AIX Cryptographic Framework により制御されるトークンを管理することができます。 このツールを使用すると、PKCS #11 トークンの初期化、作成、破壊、スロットの管理、ユーザー・パスワードのリセット、オブジェクト削除の確認、オブジェ

クト・トラストの指定、パフォーマンスと一般管理用に AIX Cryptographic Framework の調整の実行などができます。 **p11admin** コマンドの実行により、このツールの GUI バージョンを開始できます。 このツールはすべての使用可能な PKCS #11 トークンをロードします。 これらのトークンの詳細は、矢印キーを使用してトークンのリストをスクロールアップまたはスクロールダウンすることにより表示できます。トークンを選択するには、矢印キーを使用してトークンを強調表示し、Enter キーを押します。 このツールのコマンド・ライン・バージョンは、次のコマンドの実行により開始できます。

```
p11admin -b <batchfile>
```

コマンド・プロファイル**:**

AIX 暗号フレームワーク・ツールは、OpenSSL ライブラリーを使用して、カスタム・プロファイルを作成するために使用される構成ファイルを構文解析します。これらのプロファイルを使用して **p11km** コマンドおよび **p11admin** コマンド用の、例えば GUI カラーなどのツール属性を設定できます。

206 ページの『バッチ処理』内に指定されているファイル・フォーマットの使用により、GUI をカスタマイズするため以下のプロファイル・ファイルを作成し、編集できます。

注**:** ご使用のプロファイル・ファイルの作成後、それの名前を付け、以下のようにユーザーのホーム・ディレクトリーに保管します。

```
$HOME/.p11km

$HOME/.p11admin
```

以下の GUI カラー属性がサポートされています。

```
action_name = "GUI_COLORS"
gui_fg_color = "<color name>" ## Foreground Color
gui_bg_color = "<color name>" ## Background Color
gui_vc_color = "<color name>" ## View Content Color
```

ここで、 *<color name>* は次のいずれかの値です。

- LIGHT GRAY
- WHITE
- BLACK
- DARK GRAY
- RED
- LIGHT RED
- YELLOW
- ORANGE または BROWN
- GREEN
- LIGHT GREEN
- BLUE
- LIGHT BLUE
- CYAN
- LIGHT CYAN
- MAGENTA
- LIGHT MAGENTA

例: p11km プロファイル ($HOME/.p11km)

```
[p11km_cmd]
gui_fg_color = "RED"
gui_bg_color = "BLACK"
gui_vc_color = "WHITE"
```

例: p11admin プロファイル ($HOME/.p11admin)

```
[p11admin_cmd]
gui_fg_color = "BLUE"
gui_bg_color = "LIGHT GRAY"
gui_vc_color = "BLACK"
```

バッチ処理**:**

コマンド・ラインからバッチ処理コマンドを実行して、PKCS #11 ツールの GUI バージョンで使用可能な同一タスクを実行できます。

PKCS #11 Key Management ツール (p11km) のコマンド書式は次のとおりです。

```
p11km -b <batchfile>
```

PKCS #11 Key Administration ツール (p11admin) のコマンド書式は次のとおりです。

```
p11admin -b <batchfile>
```

これらのツールはバッチ・ファイルを構文解析するために OpenSSL ライブラリーを使用するので、バッチ・ファイルのフォーマットは標準の OpenSSL 構成ファイルフォーマットに従います。 各セクションは別個のコマンドであり、属性値ペアは処理に必要な情報を提供します。各セクション・コマンドは上部から下部の順でバッチ処理されます。個々のバッチ・コマンドが失敗した場合、エラーが出力され、バッチ処理はその後のセクション・コマンドを処理しないで終了します。

以下に、OpenSSL 構成ファイル・フォーマットの例を示します。

```
[section1]
attribute1 = "value1"
attribute2 = "value2"
...
attributeN = "valueN"
[section2]
attribute1 = "value1"
attribute2 = "value2"
...
attributeN = "valueN"
...
...
[sectionN]
attribute1 = "value1"
attribute2 = "value2"
...
attributeN = "valueN"
```

PKCS #11 ツール・コマンド・セクションが OpenSSL 構成ファイル・セクションと確実に共存するようにするため、PKCS #11 セクションに以下のプレフィックスを使用します。

p11km ツール

```
p11km_cmd
```

p11admin ツール

```
p11admin_cmd
```

各 p11km_cmd セクションまたは p11admin_cmd セクションは、そのセクションに関連付けられた特定のコマンドを識別するストリング値を指定した、 1 つの action_name 属性のみを含む必要があります。

最も簡易な例は、追加のパラメーターを持たないコマンドを記述する 1 つのコマンド・セクションを含むファイルです。 システムで使用可能な PKCS #11 トークンをリストするバッチ・コマンドを実行するため、p11km ツールを使用する方法の例を、以下に示します。

```
[p11km_cmd_list_my_tokens]
action_name="LIST_TOKENS"
```

各バッチ・コマンドは、オプションのブール属性サポートします。

```
start_gui="<boolean>"
```

TRUE の値を指定したブール属性を含むバッチ・コマンドを実行した場合、そのコマンドの完了後、バッチ処理は終了し、GUI が始動します。

注: バッチ・ファイルにオプションの **start_gui** 属性が組み込まれたコマンドが含まれていた場合、その後にリストされているバッチ・コマンドは処理されません。

バッチ・コマンド:

バッチ・コマンドは PKCS #11 ツールにコマンド・ライン・アクセスを提供します。

PKCS #11 Key Management ツール (p11km) では、以下のバッチ・コマンドが使用可能です。

注: バッチ・コマンドを使用するには、以下のようにします。

1. 206 ページの『バッチ処理』で説明されているようにバッチ・ファイルを作成し、編集します。
2. 使用したいバッチ・コマンド用の属性を含む新規の p11km_cmd セクションを作成します。

List available PKCS #11 tokens (使用可能な **PKCS #11** トークンのリスト表示**)**

使用可能な PKCS #11 トークン用のレポートを生成し、トークン情報とスロット情報を表示します。

必須属性

```
action_name = "LIST_TOKENS"
```

オプションの属性

```
start_gui = "<boolean>"
```

<boolean> は、TRUE または FALSE のいずれかです。

例

```
[p11km_cmd_list_tokens]
action_name = "LIST_TOKENS"
```

List available PKCS#11 mechanisms (使用可能な **PKCS #11** メカニズムのリスト表示**)**

レポートの生成と (ドライバーとスロット属性値の指定により突き合わされた) 具体的な PKCS #11 トークンによりサポートされる使用可能な PKCS #11 メカニズムの表示を行います。

必須属性

```
action_name = "LIST_MECHANISMS"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

ここで、*<slot number>* は正整数値で、*<driver name>* は以下の値のうちの 1 つです。

値	説明
AIX	AIX OS Cryptographic Framework (AIX OS 暗号フレームワーク)
IBM_4758_4960	IBM 4758/4960 Cryptographic Hardware Adapter (IBM 4758/4960 暗号ハードウェア・アダプター)
IBM_4764	IBM 4764 Cryptographic Hardware Adapter (IBM 4764 暗号ハードウェア・アダプター)
Other	OTHER を指定した場合、**p11_driver_path** 属性も指定する必要があります。

オプションの属性

```
start_gui = "<boolean>"
```

補足属性

```
p11_driver_path = "<path to PKCS#11 driver>"
```

ここで、*<path to PKCS#11 driver>* はコマンドで使用される絶対 UNIX パス、および PKCS #11 ライブラリーのファイル名です。 この属性は、**p11_driver** 属性が OTHER に設定された場合にのみ指定できます。

例

```
[p11km_cmd_list_4764_slot_0_mechs]
action_name = "LIST_MECHANISMS"
p11_driver = "IBM_4764"
p11_slot = "0"
start_gui = "TRUE"
```

List available PKCS #11 objects (使用可能な **PKCS #11** オブジェクトのリスト表示**)**

レポートの生成と (ドライバーとスロット属性値の指定により突き合わされた) PKCS #11 トークンによりサポートされる使用可能な PKCS #11 オブジェクトの表示を行います。

必須属性

```
action_name = "LIST_OBJECTS"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

オプションの属性

```
p11_login = "<boolean>"
p11_label = "<string>"
p11_class = "<PKCS#11 Object Class>"
p11_private = "<boolean>"
p11_trusted = "<boolean>"
p11_sensitive = "<boolean>"
start_gui = "<boolean>"
```

ここで、*<PKCS#11 Object Class>* は、RSA の PKCS #11 仕様で定義されている以下の値のうちの 1 つです。

```
CKO_DATA
CKO_CERTIFICATE
CKO_PUBLIC_KEY
CKO_PRIVATE_KEY
CKO_SECRET_KEY
CKO_HW_FEATURE
CKO_DOMAIN_PARAMETERS
CKO_MECHANISM
CKO_VENDOR_DEFINED
```

例

```
[p11km_cmd_list_private_objs]
action_name = "LIST_OBJECTS"
p11_login = "TRUE"
p11_private = "TRUE"
p11_driver = "AIX"
p11_slot = "5"
```

Change PKCS #11 token user's PIN (PKCS #11 トークン・ユーザーの **PIN** の変更**):**
トークンにログインするときに使用される PKCS #11 トークン・ユーザーの PIN を変更します。

必須属性

```
action_name = "CHANGE_USER_PIN"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11km_cmd_change_my_pin]
action_name = "CHANGE_USER_PIN"
p11_slot = "1337"
p11_driver = "IBM_4764"
```

Delete PKCS #11 Objects (PKCS #11 オブジェクトの削除**)**
PKCS #11 オブジェクトを削除します。 オブジェクトは、**LIST_OBJECTS** コマンドの実行、および以下の属性を指定した同じテンプレートの使用に起因するオブジェクトの番号付きリストに基づいて削除されます。

```
p11_label = "<string>"
p11_class = "<PKCS#11 Object Class>"
p11_private = "<boolean>"
p11_trusted = "<boolean>"
p11_sensitive = "<boolean>"
p11_login = "<boolean>"
```

重要**:** バッチ処理間でトークン状態と一貫性が維持されないので、オブジェクトは気付かずに削除される場合があります。 あるオブジェクトが当初記載された時刻と削除された時刻の間に、同一のトークンに対して実行している他の (複数の) プロセスにより (複数の) オブジェクトが追加または削除された場合、オブジェクトのリストの順序は変化します。

必須属性

```
action_name = "DELETE_OBJECTS"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
p11_objects = "<CSV>"
```

ここで、<*CSV*> は、ワード ALL (すべてのトークン・オブジェクト)、または正整数のコンマ区切りリストで、この正整数は以下のオプションの属性の使用により番号付き順序で表示されるオブジェクトに対応します。

オプションの属性

```
p11_label = "<string>"
p11_class = "<PKCS#11 Object Class>"
p11_private = "<boolean>"
p11_trusted = "<boolean>"
p11_sensitive = "<boolean>"
p11_login = "<boolean>"
start_gui = "<boolean>"
```

例

```
[p11km_cmd_delete_seven_objects]
action_name = "DELETE_OBJECTS"
p11_slot = "0"
p11_driver = "AIX"
p11_objects = "1,5,10,11,12,27,33"
p11_login = "TRUE"
```

Move PKCS #11 objects: (PKCS #11 オブジェクトの移動**)**

PKCS #11 オブジェクトを移動します。 オブジェクトは、**LIST_OBJECTS** コマンドの実行、および同じテンプレートの使用に起因するオブジェクトの番号付きリストに基づいて移動されます。

重要**:** バッチ処理間でトークン状態と一貫性が維持されないので、オブジェクトは気付かずに移動される場合があります。 あるオブジェクトが当初記載された時刻と移動された時刻の間に、同一のトークンに対して実行している他の (複数の) プロセスにより (複数の) オブジェクトが追加または削除された場合、オブジェクトのリストの順序は変化します。

必須属性

```
action_name = "MOVE_OBJECTS"
########################################
###### Source Token Identification: ######
p11_driver = "<driver name>"
p11_slot = "<slot number>"
########################################
###### Target Token Identification: ######
p11_driver_target = "<driver name>"
p11_slot_target = "<slot number>"
########################################
###### Objects being moved to target: ######
p11_objects = "<CSV>"
```

オプションの属性

```
p11_label = "<string>"
p11_class = "<PKCS#11 Object Class>"
p11_private = "<boolean>"
p11_trusted = "<boolean>"
p11_sensitive = "<boolean>"
p11_login = "<boolean>"
start_gui = "<boolean>"
```

例

```
[p11km_cmd_move_three_objects]
action_name = "MOVE_OBJECTS"
p11_slot = "0"
p11_slot_target = "1"
p11_driver = "AIX"
p11_driver_target = "AIX"
p11_objects = "15,20,60"
p11_login = "FALSE"
```

Copy PKCS #11 objects (PKCS #11 オブジェクトのコピー**)**

PKCS #11 オブジェクトをコピーします。 オブジェクトは、**LIST_OBJECTS** コマンドの実行、および同じテンプレートの使用に起因するオブジェクトの番号付きリストに基づいてコピーされます。

重要**:** バッチ処理間でトークン状態と一貫性が維持されないので、オブジェクトは気付かずにコピーされる場合があります。 あるオブジェクトが当初記載された時刻とコピーされた時刻の間に、同一のトークンに対して実行している他の (複数の) プロセスにより (複数の) オブジェクトが追加または削除された場合、オブジェクトのリストの順序は変化します。

必須属性

```
action_name = "COPY_OBJECTS"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
p11_driver_target = "<driver name>"
p11_slot_target = "<slot number>"
p11_objects = "<CSV>"
```

オプションの属性

```
p11_label = "<string>"
p11_class = "<PKCS#11 Object Class>"
p11_private = "<boolean>"
p11_trusted = "<boolean>"
p11_sensitive = "<boolean>"
p11_login = "<boolean>"
start_gui = "<boolean>"
```

例

```
[p11km_cmd_copy_one_private_object]
action_name = "COPY_OBJECTS"
p11_slot = "0"
p11_slot_target = "1"
p11_driver = "AIX"
p11_driver_target = "AIX"
p11_objects = "3"
p11_login = "TRUE" ## REQUIRED FOR PRIVATE OBJECT MGT.
```

Export and backup PKCS #11 objects to a file (PKCS #11 オブジェクトをあるファイルへのエクスポートとバックアップ**)**

PKCS #11 オブジェクトをエクスポートし、バックアップをとる。 オブジェクトは、**LIST_OBJECTS** コマンドの実行、および同じテンプレートの使用に起因するオブジェクトの番号付きリストに基づいてエクスポートされます。

重要**:** バッチ処理間でトークン状態と一貫性が維持されないので、オブジェクトは気付かずにエクスポートされる場合があります。 あるオブジェクトが当初記載された時刻とエクスポートされた時刻の間に、同一のトークンに対して実行している他の (複数の) プロセスにより (複数の) オブジェクトが追加または削除された場合、オブジェクトのリストの順序は変化します。

必須属性

```
action_name = "EXPORT_OBJECTS"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
p11_object_file = "<file name>"
p11_objects = "<CSV>"
```

オプションの属性

```
p11_label = "<string>"
p11_class = "<PKCS#11 Object Class>"
p11_private = "<boolean>"
p11_trusted = "<boolean>"
p11_sensitive = "<boolean>"
p11_login = "<boolean>"
start_gui = "<boolean>"
```

例

```
[p11km_cmd_backup_objects]
action_name = "EXPORT_OBJECTS"
p11_slot = "0"
p11_driver = "AIX"
p11_objects = "ALL"
p11_login = "TRUE"
p11_object_file = "/home/user1/p11km.backup"
```

Import PKCS #11 objects from a file (ファイルからの **PKCS #11** オブジェクトのインポート**)**

PKCS #11 エクスポート・ファイルから作成された PKCS #11 オブジェクトをインポートします。

必須属性

```
action_name = "IMPORT_OBJECTS"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
p11_object_file = "<file name>"
```

オプションの属性

```
p11_login = "<boolean>" # REQUIRED TO IMPORT ANY PRIVATE OBJECTS
start_gui = "<boolean>"
```

例

```
[p11km_cmd_import_my_backed_up_objects]
action_name = "IMPORT_OBJECTS"
p11_slot = "0"
p11_driver = "AIX"
p11_login = "TRUE"
p11_object_file = "/home/user1/p11km.backup"
```

Create a self-signed certificate (自己署名証明書の作成**)**

PKCS #11 トークン上で自己署名の X.509 証明書および関連付けられた PKCS #11 オブジェクトを作成します。

必須属性

```
action_name = "CREATE_SSC"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
p11_login = "TRUE"
p11_ssc_label = "<string>"
p11_ssc_config = "<openssl configuration file>"
```

ここで、*<openssl configuration file>* は、自己署名証明書の作成で使用された値で記入された OpenSSL 構成ファイルの絶対 UNIX パスおよびファイル名です。

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11km_cmd_self_signed_certificate]
action_name = "CREATE_SSC"
p11_slot = "0"
p11_driver = "AIX"
p11_login = "TRUE"
p11_ssc_label = "Lab RADIUS Server"
p11_ssc_config = "/etc/radius/EAP-TLS/openssl.cnf"
```

Create a PKCS #10 certificate signing request (PKCS #10 証明書署名要求の作成**)**

PKCS #10 認証要求または証明書署名要求 (CSR) を作成します。

必須属性

```
action_name = "CREATE_CSR"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
p11_login = "TRUE"
p11_csr_label = "<string>"
p11_csr_file = "<path to CSR output file>"
p11_csr_type = "<DER or Base64>'
p11_csr_config = "<openssl configuration file>"
```

ここで、<*DER or Base64*> は、ASN.1 (DER) エンコード CSR 出力ファイルまたは Base64 エンコード出力ファイルのいずれかを生成し、<*path to CSR output file*> は CSR 出力への絶対 UNIX パスおよびファイル名を表します。

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11km_cmd_my_pkcs10_base64]
action_name = "CREATE_SSC"
p11_slot = "0"
p11_driver = "AIX"
p11_login = "TRUE"
p11_csr_label = "Lab RADIUS Server"
p11_csr_type = "Base64"
p11_csr_file = "/etc/radius/EAP-TLS/certreq.b64"
p11_csr_config = "/etc/radius/EAP-TLS/openssl.cnf"
```

PKCS #11 Administration ツール (p11admin) 内で以下のバッチ・コマンドが使用可能です。

注**:** バッチ・コマンドを使用するには、以下のようにします。

1. 206 ページの『バッチ処理』で説明されているようにバッチ・ファイルを作成し、編集します。
2. 使用したいバッチ・コマンド用の属性を含む新規の p11km_cmd セクションを作成します。

List available PKCS #11 tokens (使用可能な **PKCS #11** トークンのリスト表示**)**

使用可能な PKCS #11 トークン用のレポートを作成し、トークン情報とスロット情報を表示します。

必須属性

```
action_name = "ADM_LIST_TOKENS"
```

オプションの属性

```
start_gui = "<boolean>"
```

、ここで<*boolean*> は、TRUE または FALSE のいずれかです。

例

```
[p11admin_cmd_list_tokens]
action_name = "ADM_LIST_TOKENS"
```

List available PKCS#11 mechanisms (使用可能な **PKCS #11** メカニズムのリスト表示**)**

レポートの生成と (ドライバーとスロット属性値の指定により突き合わされた) PKCS #11 トークンによりサポートされる使用可能な PKCS #11 メカニズムの表示を行います。

必須属性

```
action_name = "ADM_LIST_MECHANISMS"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

ここで、<*slot number*> は正整数値で、<*driver name*> は以下の値のうちの 1 つです。

値	説明
AIX	AIX OS Cryptographic Framework (AIX OS 暗号フレームワーク)
IBM_4758_4960	IBM 4758/4960 Cryptographic Hardware Adapter (IBM 4758/4960 暗号ハードウェア・アダプター)
IBM_4764	IBM 4764 Cryptographic Hardware Adapter (IBM 4764 暗号ハードウェア・アダプター)
その他	OTHER を指定した場合、**p11_driver_path** 属性も指定する必要があります。

オプションの属性

```
start_gui = "<boolean>"
```

補足属性

```
p11_driver_path = "<path to PKCS#11 driver>"
```

ここで、*<path to PKCS#11 driver>* はコマンドで使用される絶対 UNIX パス、および PKCS #11 ライブラリーのファイル名です。 この属性は、**p11_driver** 属性が OTHER に設定された場合にのみ指定できます。

例

```
[p11admin_cmd_list_4764_slot_0_mechs]
action_name = "ADM_LIST_MECHANISMS"
p11_driver = "IBM_4764"
p11_slot = "0"
start_gui = "TRUE"
```

Display information for a PKCS #11 token (PKCS #11 トークンの情報の表示**)**

PKCS #11 トークンと PKCS #11 トークンのスロット情報を表示します。

必須属性

```
action_name = "ADM_SHOW_TOKEN_INFO"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11admin_cmd]
action_name = "ADM_SHOW_TOKEN_INFO"
p11_slot = "411"
p11_driver = "IBM_4764"
```

Initialize a PKCS #11 token (PKCS #11 トークンの初期化**):**

PKCS #11 トークンを初期化します。 初期化によりトークンはリセットされ、すべての保管された PKCS#11 オブジェクトとデータは消去され、次にトークンに新しい名前をつけることが許可されます。

重要**:** 初期化プロセス中にすべての PKCS #11 オブジェクトとデータが消去されるので、PKCS #11 トークンを初期化する前にオブジェクトとデータを必要としないことを確認してください。

必須属性

```
action_name = "ADM_INIT_TOKEN"
p11_driver = "<driver name>"
p11_slot = "<slot number>" ## SAME AS 'p11_init_slot'
p11_init_slot = "<slot number>" ## SAME AS 'p11_slot'
p11_init_label = "<string>" ## NEW TOKEN LABEL
```

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11admin_cmd]
action_name = "ADM_INIT_TOKEN"
p11_slot = "1"
p11_driver = "IBM_4764"
p11_init_slot = "1"
p11_init_label = "ABC Token"
```

View the clock for a PKCS #11 token (PKCS #11 トークンのクロックの表示**)**

トークンがクロックを持っている場合、PKCS #11 トークンのハードウェア・クロックを表示します。

必須属性

```
action_name = "ADM_CLOCK_VIEW"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11admin_cmd]
action_name = "ADM_CLOCK_VIEW"
p11_slot = "1"
p11_driver = "IBM_4764"
```

Set the clock for a PKCS #11 token (PKCS #11 トークンのクロックの設定**)**

トークンがクロックを持っている場合、PKCS #11 トークンのハードウェア・クロックを設定します。

必須属性

```
action_name = "ADM_CLOCK_SET"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
p11_clock_set = "<clock data>"
```

ここで、 *<clock data>* は、以下のフォーマットを持つ現在の UTC 日付および時刻です。 HH:MM:SS mm-dd-YYYY.

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11admin_cmd]
action_name = "ADM_CLOCK_SET"
p11_slot = "1"
p11_driver = "IBM_4764"
p11_clock_set = "23:59:59 12-31-1999"
```

Reset the PIN for a PKCS #11 token user (PKCS #11 トークン・ユーザーの **PIN** のリセット**)**

PKCS #11 トークン・ユーザーの PIN をリセットします。

必須属性

```
action_name = "ADM_RESET_USER_PIN"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11admin_cmd_change_so_pin]
action_name = "ADM_RESET_USER_PIN"
p11_driver = "AIX"
p11_slot = "0"
```

Change the PIN for PKCS #11 token security officer (PKCS #11 トークン・セキュリティー担当者の **PIN** の変更**)**

PKCS #11 トークン・セキュリティー担当者の PIN を変更します。 トークン管理が実行されるとき、この PIN が使用されます。

必須属性

```
action_name = "ADM_CHANGE_SO_PIN"
p11_driver = "<driver name>"
p11_slot = "<slot number>"
```

オプションの属性

```
start_gui = "<boolean>"
```

例

```
[p11admin_cmd_change_so_pin]
action_name = "ADM_CHANGE_SO_PIN"
p11_slot = "888"
p11_driver = "IBM_4764"
```

プラグ可能認証モジュール

プラグ可能認証モジュール (PAM) フレームワークにより、システム管理者は、プラグ可能モジュールの使用を通して、複数の認証メカニズムを既存のシステムに統合することができます。

PAM を使用できるアプリケーションは、既存のアプリケーションを変更せずに、新しいテクノロジーにプラグイン することができます。 この柔軟性により、管理者は以下のことが行えます。

- アプリケーション用にシステム上の任意の認証サービスを選択する
- 特定のサービスに対して複数の認証メカニズムを使用する
- 既存のアプリケーションを変更せずに、新しい認証サービス・モジュールを追加する
- 以前に入力した認証用のパスワードを複数のモジュールで使用する

PAM フレームワークは、ライブラリー、プラグ可能モジュール、および構成ファイルから成り立っています。 PAM ライブラリーは、PAM アプリケーション・プログラミング・インターフェース (API) を実装し、 PAM トランザクションを管理して、プラグ可能モジュール内で定義されている PAM サービス・プログラミング・インターフェース (SPI) を呼び出すためのサービスを提供します。 プラグ可能モジュールは、起動中のサービスと構成ファイル内のそのエントリーに基づいて、ライブラリーにより動的にロードされます。 プラグ可能モジュールのみならず、サービスに対して定義されている動作も、成功かどうかの決定要因になります。 スタッキング の概念により、複数の認証方式を通して認証を行うようにサービスを構成できます。 サポートされていれば、追加入力のプロンプトを出すのではなく、以前にサブミットされたパスワードを使用するようにモジュールを構成することもできます。

システム管理者は、/etc/security/login.cfg ファイルにある usw スタンザの **auth_type** 属性の変更を通して、PAM を使用するように AIX システムを構成することができます。 auth_type = PAM_AUTH を設定すると、従来の AIX 認証ルーチンを使用せずに、認証のために PAM API を直接呼び出すように PAM 対応コマンドを構成することができます。 この構成はランタイム決定であり、有効にするためにシステムをリブートする必要はありません。 **auth_type** 属性の詳細については、/etc/security/login.cfg ファイル参照で調べてください。以下のネイティブ AIX コマンドおよびアプリケーションは、**auth_type** 属性を認識するように変更され、PAM 認証に使用可能になっています。

- **login**
- **passwd**
- **su**
- **ftp**
- **telnet**
- **rlogin**
- **rexec**
- **rsh**
- **snappd**
- **imapd**
- **dtaction**
- **dtlogin**
- **dtsession**

以下の図は、PAM を使用するために構成されたシステム上の PAM 使用可能アプリケーション、 PAM ライブラリー、構成ファイル、および PAM モジュールの間の相互作用を示しています。 PAM 対応アプリケーションは、PAM ライブラリーの PAM API を起動します。 ライブラリーは、構成ファイル内のアプリケーション・エントリーに基づいて適切なモジュールを判別し、そのモジュールの中の PAM SPI を呼び出します。 アプリケーション内に実装された会話機能の使用を通して、 PAM モジュールとアプリケーション間で通信が行われます。 モジュールと構成ファイル内で定義されている動作によって成功か失敗かが決まり、それによって他のモジュールをロードする必要があるかどうかが決まります。 他のモジュールをロードする必要がある場合は、プロセスが続行します。そうでなければ、結果がアプリケーションに戻されます。

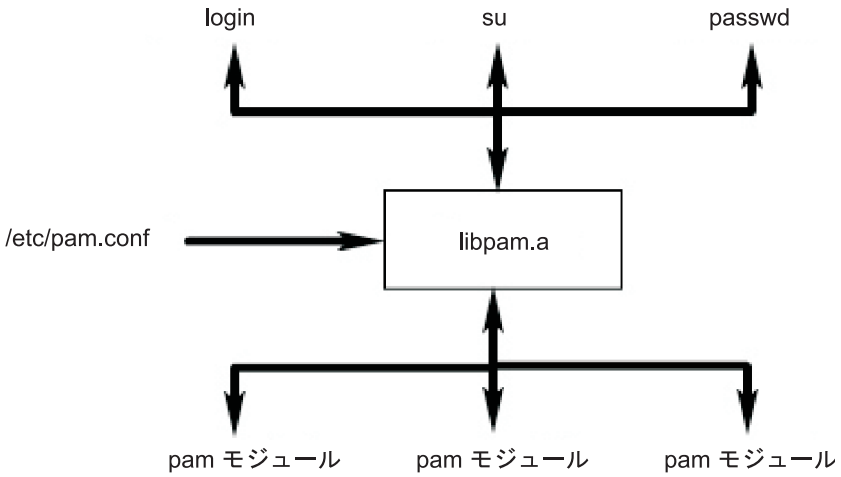


図 *3. PAM* フレームワークとエンティティー： この図は、PAM 対応コマンドが PAM ライブラリーを使用して適切な PAM モジュールにアクセスする方法を表しています。

PAM ライブラリー

PAM ライブラリー **/usr/lib/libpam.a** には、すべての PAM アプリケーションへの共通インターフェースとして機能し、かつモジュールのロードを制御する、PAM API が入っています。

モジュールは、**/etc/pam.conf** ファイルに定義されたスタッキング動作に基づいて、PAM ライブラリーによりロードされます。

以下の PAM API 関数は、PAM モジュールによって提供されている対応する PAM SPI を呼び出します。 例えば、pam_authenticate API は、PAM モジュール内の pam_sm_authenticate SPI を呼び出します。

- `pam_authenticate`
- `pam_setcred`
- `pam_acct_mgmt`
- `pam_open_session`
- `pam_close_session`
- `pam_chauthtok`

PAM ライブラリーには、アプリケーションが PAM モジュールを呼び出して情報を PAM モジュールに渡せるようにするいくつかのフレームワーク API も備えられています。以下の表は、AIX で実装されている PAM フレームワーク API とその機能を示したものです。

PAM フレームワーク **API**	機能
pam_start	PAM セッションを立ち上げます。
pam_end	PAM セッションを終了します。
pam_get_data	モジュールに固有なデータを検索します。
pam_set_data	モジュールに固有なデータを設定します。
pam_getenv	定義された PAM 環境変数の値を検索します。
pam_getenvlist	定義されたすべての PAM 環境変数ならびに値のリストを検索します。
pam_putenv	PAM 環境変数を設定します。
pam_get_item	共通の PAM 情報を検索します。
pam_set_item	共通の PAM 情報を設定します。
pam_get_user	ユーザー名を検索します。
pam_strerror	PAM 標準エラー・メッセージを取得します。

PAM モジュール

PAM モジュールにより、複数の認証メカニズムをシステムでまとめて、あるいは単独で使用できます。

特定の PAM モジュールは、 4 つのモジュール・タイプのうちの少なくとも 1 つを実装していなければなりません。モジュール・タイプと、そのモジュール・タイプに準拠するために必要な対応する PAM SPI について以下で説明します。

認証モジュール
: ユーザーを認証し、資格情報を設定、リフレッシュ、または破壊します。 これらのモジュールは、その認証と資格情報に基づいてユーザーを識別します。

  認証モジュール関数は以下のとおりです。

  - pam_sm_authenticate
  - pam_sm_setcred

アカウント管理モジュール
: ユーザー・アカウントと、認証モジュールによる識別の後のそれ以後のアクセスの妥当性を判別します。 これらのモジュールによって実行される検査には、通常、アカウントの有効期限とパスワード制約が含まれます。

  アカウント管理モジュール関数は以下のとおりです。

- pam_sm_acct_mgmt

セッション管理モジュール
: ユーザー・セッションを開始して終了します。さらに、セッション監査のサポートも提供されることがあります。

  セッション管理モジュール関数は以下のとおりです。

  - pam_sm_open_session
  - pam_sm_close_session

パスワード管理モジュール
: パスワードの変更と、関連属性の管理を行います。

  パスワード管理モジュール関数は以下のとおりです。

  - pam_sm_chauthtok

PAM 構成ファイル

/etc/pam.conf 構成ファイルは、各 PAM モジュール・タイプのサービス・エントリーから成り立ち、定義されているモジュール・パスを通してサービスをルーティングする役目を果たします。

ファイル内のエントリーは、次のようなスペース区切りのフィールドで構成されています。

service_name module_type control_flag module_path module_options

これらのフィールドについて、以下に説明します。

service_name
: サービスの名前を示します。 キーワード OTHER は、エントリー内で指定されていないアプリケーションに使用するデフォルトのモジュールを定義する場合に使用します。

module_type
: サービスのモジュール・タイプを示します。 有効なモジュール・タイプは、**auth**、**account**、**session**、または **password** です。 1 つのモジュールが、1 つ以上の複数のモジュール・タイプに対してサポートを提供します。

control_flag
: モジュールのスタッキング動作を示します。 サポートされている制御フラグは、required、requisite、sufficient、または optional です。

module_path
: サービスのためにロードするモジュールを指定します。 *module_path* の有効値は、モジュールの絶対パスとして指定でき、またモジュール名のみとしても指定できます。 モジュールの絶対パスを指定する場合、PAM ライブラリーは、32 ビット・サービスの場合はその *module_path* をロードに使用し、64 ビット・サービスの場合は 64 サブディレクトリーを使用します。 モジュールの絶対パスを指定しない場合、PAM ライブラリーは `/usr/lib/security` (32 ビット・サービスの場合) または `/usr/lib/security/64` (64 ビット・サービスの場合) をモジュール名の前に付加します。

module_options
: サービス・モジュールに渡すことができるオプションのスペース限定リストを指定します。 このフィールドの値は、*module_path* フィールドで定義されているモジュールがサポートするオプションに依存します。 このフィールドはオプションです。

誤った形式のエントリーや、**module_type** フィールドまたは **control_flag** フィールドの値が正しくないエントリーは、PAM ライブラリーでは無視されます。 行の開始が番号記号 (#) 文字で始まるエントリーはコメントを表すので、これも無視されます。

PAM は、一般に「スタッキング」と呼ばれる概念をサポートします。これにより、各サービスに複数のメカニズムを使用することが可能となります。 スタッキングは、1 つのサービスに、同一の **module_type** フィールドを持つエントリーを複数個作成することによって、構成ファイル内に実装されます。 モジュールは、所定のサービスのファイルにリストされている順序で呼び出され、最終的な結果は、各エントリーに対して指定されている **control_flag** フィールドによって決定されます。 **control_flag** フィールドの有効値と、対応するスタック内の動作は以下のとおりです。

control_flag フィールドの値	動作
required	結果が成功となるためには、スタック内のすべての必須 (required) モジュールが渡されなければなりません。 1 つ以上の必須モジュールが失敗した場合、スタック内のすべての必須モジュールが試行されますが、最初に失敗した必須モジュールからエラーが戻されます。
requisite	必須 (required) と似ていますが、必要 (requisite) モジュールが失敗した場合は、スタック内のそれ以降のモジュールは処理されず、必須または必要モジュールからの最初の障害コードを直ちに戻します。
sufficient	sufficient のフラグが立てられたモジュールが成功し、その前の必須 (required) または十分 (sufficient) モジュールが失敗していなければ、スタック内の残りのモジュールはすべて無視され、成功が戻されます。
optional	スタック内のモジュールがどれも必須 (required) ではなく、十分 (sufficient) モジュールが 1 つも成功していない場合は、サービスの少なくとも 1 つのオプション (optional) モジュールが成功していなければなりません。 スタック内の他のモジュールが成功している場合、オプション・モジュール内の失敗は無視されます。

次の **/etc/pam.conf** サブセットは、ログイン・サービス用の **auth** モジュール・タイプにおけるスタッキングの例です。

```
#
# PAM configuration file /etc/pam.conf
#

# Authentication Management
login   auth     required       /usr/lib/security/pam_ckfile    file=/etc/nologin
login   auth     required       /usr/lib/security/pam_aix
login   auth     optional       /usr/lib/security/pam_test      use_first_pass
OTHER   auth     required       /usr/lib/security/pam_prohibit
```

この例としての構成ファイルには、ログイン・サービス用として 3 つのエントリーが含まれています。 **pam_ckfile** と **pam_aix** が両方とも required として指定されているので、両方のモジュールが実行されます。結果全体が成功するためには、モジュールが両方とも正常に実行される必要があります。架空の **pam_test** モジュールに対応する 3 番目のエントリーはオプションであり、その成否はユーザーがログインできるかどうかに影響しません。 **pam_test** モジュールのオプション **use_first_pass** は、新規パスワードを求めるプロンプトを出す代わりに、前に入力されたパスワードの使用を要求します。

OTHER キーワードをサービス名として使用すると、構成ファイルに明示的に宣言されていない他のサービスのためのデフォルトを設定することが可能になります。 デフォルトを設定すると、特定のモジュール・タイプのすべてのケースが少なくとも 1 つのモジュールによってカバーされることになります。 この例の場合、ログイン以外のすべてのサービスは常に失敗します。これは **pam_prohibit** モジュールがすべての呼び出しに対して PAM 失敗を戻すためです。

pam_aix モジュール

pam_aix モジュールは、AIX セキュリティー・サービスに PAM 使用可能なアプリケーションを提供する PAM モジュールです。これはアプリケーションが存在している場所にある同等の AIX サービスを呼び出すことによって行われます。

呼び出されたサービスは、ユーザーの定義および **methods.cfg** ファイル内の対応するセットアップに基づいて、ロード可能認証モジュールまたは AIX 組み込み関数によって実行されます。 AIX サービスの実行中に生成されたエラー・コードは、対応する PAM エラー・コードにマップされます。

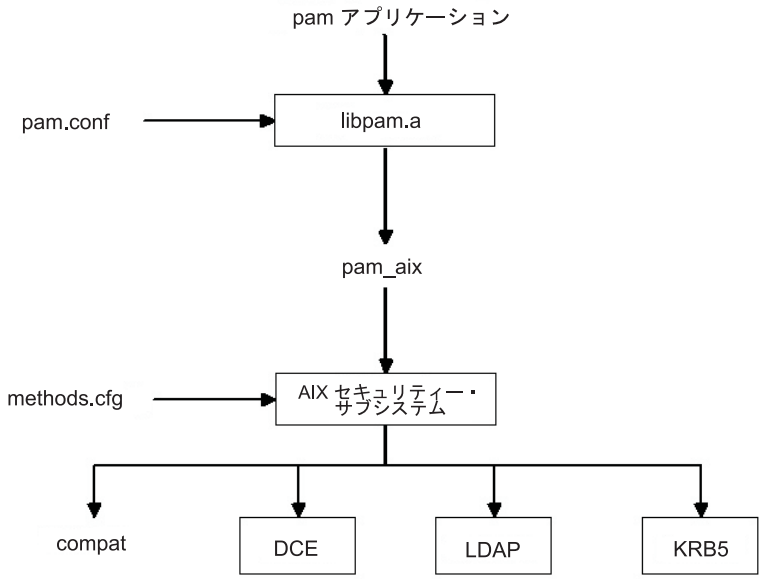


図 *4. PAM* アプリケーションから *AIX* セキュリティー・サブシステムへのパス

この図は、**/etc/pam.conf** ファイルが **pam_aix** モジュールを使用するように構成されている場合に、PAM アプリケーション API 呼び出しがたどるパスを表しています。 この図が示しているような統合により、いずれかのロード可能認証モジュール (DCE、LDAP、または KRB5) または AIX ファイル (compat) でのユーザー認証が可能になります。

pam_aix モジュールは、**/usr/lib/security** ディレクトリーにインストールされています。 **pam_aix** モジュールを統合するには、このモジュールを使用するように **/etc/pam.conf** ファイルを構成しておく必要があります。 スタッキングも使用可能ですが、次の **/etc/pam.conf** ファイルの例には示されていません。

```
#
# Authentication management
#
OTHER   auth     required       /usr/lib/security/pam_aix

#
# Account management
#
OTHER   account  required       /usr/lib/security/pam_aix

#
# Session management
#
OTHER   session  required       /usr/lib/security/pam_aix

#
```

```
# Password management
#
OTHER   password required       /usr/lib/security/pam_aix
```

pam_aix モジュールには、pam_sm_authenticate、pam_sm_chauthok、および pam_sm_acct_mgmt SPI 関数用のインプリメンテーションがあります。 pam_sm_setcred、pam_sm_open_session、および pam_sm_close_session SPI は、pam_aix モジュールにも実装されますが、これらの SPI 関数は PAM_SUCCESS 呼び出しを戻します。

PAM SPI 呼び出しから AIX セキュリティー・サブシステムへのおおよそのマッピングを次に示します。

```
 PAM SPI                    AIX
=========                  =====
pam_sm_authenticate   -->  authenticate
pam_sm_chauthtok      -->  passwdexpired, chpass
                           Note: passwdexpired is only checked if the
                           PAM_CHANGE_EXPIRED_AUTHTOK flag is passed in.
pam_sm_acct_mgmt      -->  loginrestrictions, passwdexpired
pam_sm_setcred        -->  No comparable mapping exists, PAM_SUCCESS returned
pam_sm_open_session   -->  No comparable mapping exists, PAM_SUCCESS returned
pam_sm_close_session  -->  No comparable mapping exists, PAM_SUCCESS returned
```

AIX セキュリティー・サブシステムに渡すことを意図されたデータは、モジュールを使用する前に pam_set_item 関数を使用するか、あるいは、データがまだ存在しない場合はデータ用の pam_aix モジュールを使用して、設定することができます。

PAM ロード可能認証モジュール

AIX セキュリティー・サービスが、既存のロード可能な AIX 認証モジュール・フレームワークを使用して PAM モジュールを呼び出すよう構成することができます。

/usr/lib/security/methods.cfg ファイルが正しくセットアップされると、PAM ロード・モジュールは AIX セキュリティー・サービス (passwd、login、など) を PAM ライブラリーに経路指定します。 PAM ライブラリーは **/etc/pam.conf** ファイルを検査して、使用する PAM モジュールを判別し、対応する PAM SPI の呼び出しを行います。 PAM からの戻り値は AIX エラー・コードにマップされ、呼び出し側プログラムに戻されます。

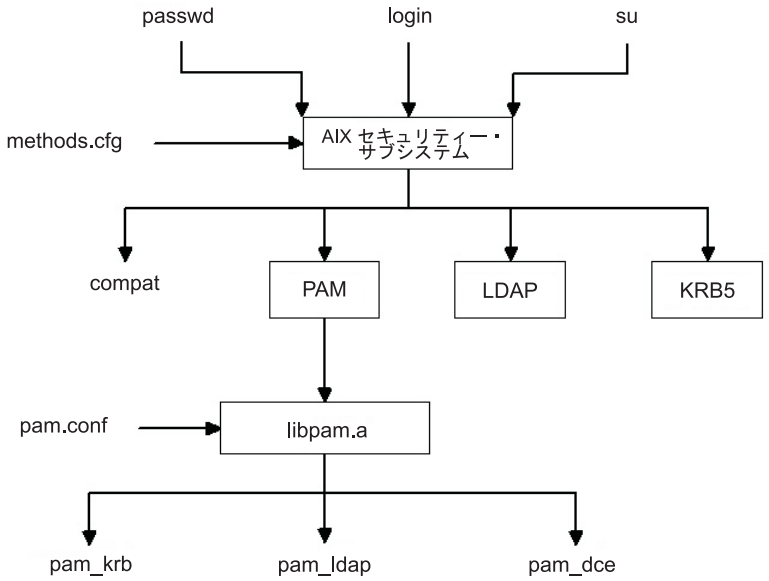


図 *5. AIX* セキュリティー・サービスから *PAM* モジュールへのパス

この図は、PAM が正しく構成されている場合に AIX セキュリティー・サービス呼び出しで取られるパスを表しています。 表示されている PAM モジュール (pam_krb、pam_ldap、および pam_dce) は、サード・パーティー・ソリューションの例としてリストしています。

PAM ロード・モジュールは **/usr/lib/security** ディレクトリーにインストールされる、認証専用モジュールです。 PAM モジュールはデータベースと結合されて、複合ロード・モジュールを形成する必要があります。 以下の例は、files というデータベースを伴う複合 PAM モジュールを形成するために、**methods.cfg** ファイルに追加できるスタンザを表しています。 **db** 属性の BUILTIN キーワードは、データベースを UNIX ファイルの集まりとして指定します。

```
PAM:
        program = /usr/lib/security/PAM

PAMfiles:
        options = auth=PAM,db=BUILTIN
```

ユーザーの作成と変更は、-R オプションを管理コマンドと共に使用して、ユーザーの作成時に *SYSTEM* 属性を設定することによって行います。 次に例を示します。

```
mkuser -R PAMfiles SYSTEM=PAMfiles registry=PAMfiles pamuser
```

こうすると、それ以後の AIX セキュリティー・サービス (login、passwd など) の呼び出しに対し、認証に PAM ロード・モジュールを使用するようにという指示が出されます。 この例では複合モジュールに **files** データベースを使用しましたが、インストールされていれば、LDAP などの他のデータベースを使用することもできます。 前述のとおりにユーザーを作成すると、次のような AIX セキュリティーから PAM API 呼び出しへのマッピングが行われます。

```
 AIX                    PAM API
=====                  =========
authenticate       -->  pam_authenticate
chpass             -->  pam_chauthtok
passwdexpired      -->  pam_acct_mgmt
passwdrestrictions -->  No comparable mapping exists, success returned
```

/etc/pam.conf ファイルをカスタマイズすれば、 PAM API 呼び出しが認証のために望ましい PAM モジュールに誘導されるようにすることができます。認証メカニズムをさらに洗練させるために、スタッキングを実装することができます。

PAM からユーザー・ダイアログに応じることはできないため、AIX セキュリティー・サービスによって求められたデータは、pam_set_item 関数を通して PAM に渡されます。 プロンプト入力はセキュリティー・サービスによって処理されるので、PAM モジュールとの統合のために作成される PAM モジュールは、pam_get_item 呼び出しですべてのデータを取得すべきであり、ユーザーにデータの入力を求めるプロンプトを出すべきではありません。

AIX セキュリティー・サービスが PAM に経路指定され、それに対応して PAM モジュールが操作を実行するために AIX セキュリティー・サービスの呼び出しを試行するという、生じうる構成エラーを発見するためのループ検出が準備されています。 このループ・イベントが検出されると、意図された操作は直ちに失敗します。

注**:** AIX セキュリティー・サービスから PAM モジュールへの PAM 統合を使用する場合は、pam_aix module を使用するように **/etc/pam.conf** ファイルを作成しない でください。ループ状態の原因になります。

PAM モジュールの追加

PAM モジュールを追加して、複数の認証メカニズムを使用可能にすることができます。

1. 32 ビット・バージョンのモジュールは **/usr/lib/security** ディレクトリーに入れ、64 ビット・バージョンのモジュールは **/usr/lib/security/64** ディレクトリーに入れます。
2. ファイル所有権を `root` に、アクセス権を `555` に設定します。 PAM ライブラリーは、root ユーザーが所有していないモジュールはロードしません。
3. **/etc/pam.conf** 構成ファイルを更新して、望ましいサービス名のエントリーにこのモジュールを含めます。
4. 影響を受けるサービスをテストして、その機能性を確認します。 ログイン・テストを実行するまでは、システムからログオフしないでください。

/etc/pam.conf ファイルの変更

/etc/pam.conf ファイルを変更する前に、いくつかのことを考慮してください。

/etc/pam.conf 構成ファイルを変更する際は、次の点を考慮してください。

- ファイルは、常に root ユーザーおよびグループ・セキュリティーにより所有されること。 ファイルに対するアクセス権は 644 にして、読み取りアクセスは誰でも付与し、変更アクセス権は root のみに限定すること。
- セキュリティーをより確実にするために、PAM 使用可能サービスをそれぞれ明示的に構成してから、OTHER サービス・キーワードに **pam_prohibit** モジュールを使用すること。
- 選択したモジュールに付随する資料を読み、どの制御フラグとオプションがサポートされていて、それらがどのような影響を及ぼすのかを判断すること。
- スタック・モジュール内の required、requisite、sufficient、および optional 制御フラグの動作に留意しながら、モジュールと制御フラグの順序付けを注意深く選択すること。

注**:** PAM 構成ファイルの構成に間違いがあると、システムにログインできなくなる可能性があります。その理由は、構成は root を含むすべてのユーザーに適用されるためです。 ファイルを変更した後は必

ず、システムからログアウトする前に、影響を受けるアプリケーションをテストしてください。 あるシステムにログインできない場合、そのシステムを保守モードでブートしてから **/etc/pam.conf** 構成ファイルを訂正すると、リカバリーできます。

PAM デバッグの使用可能化

プラグ可能認証モジュール (PAM) ライブラリーは、実行中にデバッグ情報を提供することができます。システムがデバッグ出力を収集できるようにすれば収集された情報を使用して PAM API 呼び出しをトラッキングし、現行の PAM セットアップ内の障害点を判別できます。

PAM デバッグ出力を使用可能にするには、以下のステップを実行します。

1. **touch** コマンドを使用して、pam_debug という名前の空のファイルを /etc/pam_debug ディレクトリー内に作成します (当該ファイルが存在していない場合)。PAM ライブラリーは /etc/pam_debug ファイルが存在するかどうかを検査し、このファイルが見つかると syslog 出力を使用可能にします。
2. 希望する優先順位レベルで auth syslog メッセージを記録するファイルが特定されるように、/etc/syslog.conf ファイルを編集します。例えば、PAM デバッグ・レベル・メッセージを /var/log/auth.log ファイルに送信するには、syslog.conf ファイルに次のテキストを新規行として追加します。

   ```
   *.debug /var/log/auth.log
   ```
3. ステップ 2 で参照される出力ファイル /var/log/auth.log がまだ存在しなければ、**touch** コマンドを使用して作成します。
4. syslogd デーモンを再始動して、構成の変更が認識されるようにするには、以下のステップを実行します。
   a. 次のコマンドを入力して、syslog デーモンを停止します。

      ```
      stopsrc -s syslogd
      ```
   b. 次のコマンドを入力して、syslog デーモンを開始します。

      ```
      startsrc -s syslogd
      ```

PAM アプリケーションを再始動すると、/etc/syslog.conf 構成ファイルに定義された出力ファイルにデバッグ・メッセージが収集されます。

OpenSSH と Kerberos バージョン 5 のサポート

Kerberos は、ネットワーク・ユーザーのための保護された認証の手段を提供する認証メカニズムです。Kerberos は、クライアントとサーバーの間で認証メッセージを暗号化することにより、ネットワークを介した平文パスワードの送信を防ぎます。 さらに、Kerberos は、管理トークンまたは資格情報の形で権限のシステムを提供します。

Kerberos を使用してユーザーを認証するには、ユーザーが **kinit** コマンドを実行して、KDC (鍵配布センター) と呼ばれる中央 Kerberos サーバーから初期資格情報を取得します。 KDC はユーザーを検証して、ユーザーに TGT (チケット許可チケット) と呼ばれる初期資格情報を渡します。 この後、ユーザーは Kerberos 対応 Telnet や OpenSSH などのサービスを利用して、リモート・ログイン・セッションを開始することができます。 Kerberos はユーザー資格情報を KDC から取得してユーザーを認証します。Kerberos は、この認証をユーザーとの対話なしで行いますので、ユーザーはログインするためのパスワードを入力する必要はありません。 IBM 版の Kerberos は、ネットワーク認証サービス (NAS) と呼ばれています。 NAS は AIX 拡張パック CD からインストールできます。 **krb5.client.rte** パッケージおよび **krb5.server.rte** パッケージから選択できます。 2003 年 7 月の OpenSSH 3.6 のリリースより、OpenSSH は、NAS バージョン 1.3 によって Kerberos 5 認証および権限をサポートします。

OpenSSH バージョン 3.8 以降は、NAS バージョン 1.4 を介して、Kerberos 5 認証ならびに権限をサポートします。 NAS (Kerberos) の旧バージョンからのマイグレーションは OpenSSH の更新よりも前に行う必要があります。 OpenSSH バージョン 3.8.x は、 NAS バージョン 1.4 以降とのみ連動します。

AIX では、オプションの方法として、Kerberos 認証機能のある OpenSSH が作成されています。 システムに Kerberos ライブラリーがインストールされていない場合、OpenSSH を実行すると Kerberos 認証がスキップされて、OpenSSH は次の構成済み認証方法 (AIX 認証など) を試行します。

Kerberos をインストールした後、Kerberos サーバーを構成する前に Kerberos の資料をお読みになることをお勧めします。 Kerberos のインストールおよび管理の方法についての詳細は、「*IBM Network Authentication Service Version 1.3 for AIX : Administrator's and User's Guide*」を参照してください。この資料は、`/usr/lpp/krb5/doc/html/lang/ADMINGD.htm` パスにあります。

関連情報:

OpenSSH

OpenSSH イメージ

OpenSSH イメージをインストールするには、次の手順を実行します。

1. AIX Web ダウンロード・パック・プログラム Web サイトにアクセスします。

   注: OpenSSH イメージは AIX 基本メディアの一部として出荷されますが、デフォルトではそのイメージはインストールされません。
2. 『追加情報』セクションで「ダウンロード」をクリックします。
3. 使用可能なパッケージにアクセスするための、お持ちの ID およびパスワードを使用してログインします。
4. 「**OpenSSH**」を選択して、「続行」をクリックします。
5. ご使用条件を受け入れて、パッケージをダウンロードします。
6. **uncompress** *packagename* コマンドを使用して、イメージ・パッケージを解凍します。次に例を示します。

   ```
   uncompress OpenSSH_6.0.0.6203.tar.Z
   ```
7. **tar -xvf** *packagename* コマンドを使用して、パッケージを untar します。次に例を示します。

   ```
   tar -xvf OpenSSH_6.0.0.6203.tar
   ```
8. **inutoc** コマンドを実行します。
9. **smitty install** コマンドを実行します。
10. 「**Install and Update Software (**ソフトウェアのインストールおよび更新**)**」を選択します。
11. 「**Update Installed Software to Latest Level (Update All) (**インストール済みソフトウェアを最新レベルに更新 **(**すべて更新**))**」を選択します。
12. 「**INPUT device / directory for software (**ソフトウェア用の入力デバイス**/**ディレクトリー**)**」用のフィールドにドット **(.)** を入力し、Enter を押します。
13. スクロールダウンして「**ACCEPT new license agreements (**新規ご使用条件に同意**)**」を表示し、**Tab** キーを押して、このフィールドを「**Yes (**はい**)**」に変更します。
14. Enter キーを 2 回押して、インストールを開始します。

OpenSSH イメージはプログラム一時修正 (PTF) でなく、基本レベルのイメージです。 インストールを行うと、前のバージョンのすべてのコードが新しいバージョンのイメージで上書きされます。

OpenSSH コンパイルの構成

以下の情報では、AIX 用に OpenSSH コードをコンパイルする方法について説明します。

OpenSSH を AIX バージョン 6.1 用に構成すると、次のような出力が得られます。

```
OpenSSH has been configured with the following options:
                     User binaries: /usr/bin
                   System binaries: /usr/sbin
               Configuration files: /etc/ssh
                   Askpass program: /usr/sbin/ssh-askpass
                      Manual pages: /usr/man
                          PID file: /etc/ssh
  Privilege separation chroot path: /var/empty
            sshd default user PATH: /usr/bin:/bin:/usr/sbin:/sbin:/usr/
                                    local/bin

                    Manpage format: man
                       PAM support: yes
                   OSF SIA support: no
                  KerberosV support: yes
                  Smartcard support: no
                    SELinux support: no
                      S/KEY support: no
              TCP Wrappers support: yes
              MD5 password support: no
                   libedit support: no
  Solaris process contract support: no
           Solaris project support: no
       IP address in $DISPLAY hack: no
           Translate v4 in v6 hack: no
                 BSD Auth support: no
              Random number source: OpenSSL internal ONLY

              Host: powerpc-ibm-aix6.1.0.0
          Compiler: cc
    Compiler flags: -bloadmap:file -qnostdinc -qnolm -qlist -qsource -qattr=full
    Preprocessor flags: -I/gsa/ausgsa/projects/o/openssh/freeware5/openssl-0.9.8r/
                include -I/gsa/ausgsa/projects/o/openssh/zlib -I/usr/include

          Linker flags: -L/gsa/ausgsa/projects/o/openssh/freeware5/
                    lib -L/gsa/ausgsa/projects/o/openssh/zlib -L/usr/include
                    -Wl,-blibpath:/usr/lib:/lib
             Libraries: -lcrypto -lz -lc  -lcrypt -lefs   -lwrap -lpam -ldl
```

注: AIX バージョン 6.1 および AIX バージョン 7.1 のコンパイル・オプションは類似しています。どちらのバージョンでもバイナリーが同じためです。

Kerberos サポートのある OpenSSH の使用

いくつかの初期セットアップでは、OpenSSH を Kerberos と併用する必要があります。

以下の手順には、Kerberos サポートが備わった OpenSSH を使用するのに必要な初期セットアップの情報が含まれています。

1. ご使用の OpenSSH クライアントおよびサーバーに、**/etc/krb5.conf** ファイルが存在していなければなりません。 このファイルは Kerberos に、使用する KDC、それぞれのチケットに与えるライフタイムなどの情報を提供します。 **krb5.conf** ファイルの例を次に示します。

   ```
   [libdefaults]
   ticket_lifetime = 600
   default_realm = OPENSSH.AUSTIN.XYZ.COM
   default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
   default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc

   [realms]
   OPENSSH.AUSTIN.xyz.COM = {
       kdc = kerberos.austin.xyz.com:88
   ```

```
    kdc = kerberos-1.austin.xyz.com:88
    kdc = kerberos-2.austin.xyz.com:88
    admin_server = kerberos.austin.xyz.com:749
    default_domain = austin.xyz.com
}

[domain_realm]
    .austin.xyz.com = OPENSSH.AUSTIN.XYZ.COM
    kdc.austin.xyz.com = OPENSSH.AUSTIN.XYZ.COM
```

2. また、各クライアント・マシンの **/etc/services** ファイルに Kerberos を追加する必要があります。

```
kerberos      88/udp    kdc    # Kerberos V5 KDC
kerberos      88/tcp    kdc    # Kerberos V5 KDC
kerberos-adm  749/tcp          # Kerberos 5 admin/changepw
kerberos-adm  749/udp          # Kerberos 5 admin/changepw
krb5_prop     754/tcp          # Kerberos slave
                               #  propagation
```

3. KDC がユーザー情報を保管するレジストリーとして LDAP を使用している場合は、 170 ページの『LDAP 認証ロード・モジュール』と Kerberos の資料をお読みください。 さらに、以下のことが行われていることを確認してください。
   - KDC が LDAP クライアントを実行している。 **secldapclntd** コマンドで LDAP クライアント・デーモンを始動できる。
   - LDAP サーバーが **slapd** LDAP サーバー・デーモンを実行している。
4. OpenSSH サーバーで、**/etc/ssh/sshd_config** ファイルを編集して以下の行を入れます。

```
KerberosAuthentication yes
KerberosTicketCleanup yes
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
UseDNS yes
```

   UseDNS が **Yes** に設定されている場合は、ssh サーバーは、ホストの逆ルックアップを実行して接続クライアントの名前を見つけます。 これは、ホスト・ベース認証を使用する場合、または最後に表示するログイン情報を IP アドレスではなくホスト名にする場合に必要です。

   注**:** 名前の逆ルックアップを実行すると DNS サーバーが到達不能になるため、一部の ssh セッションが停止します。 これが起こった場合は、UseDNS を `no` に設定して DNS ルックアップをスキップすることができます。 UseDNS が `/etc/ssh/sshd_config` ファイルで明示的に設定されていない場合、デフォルト値は `UseDNS yes` です。
5. SSH サーバーで、**startsrc -g ssh** コマンドを実行して ssh サーバー・デーモンを始動します。
6. SSH クライアント・マシンで、**kinit** コマンドを実行して初期資格情報 (TGT) を取得します。 TGT を受け取ったかどうかを確認するには、**klist** コマンドを実行します。 このコマンドは、そのユーザーに属するすべての資格情報を表示します。
7. **ssh** *username@servername* コマンドを実行して、サーバーに接続します。
8. Kerberos が、ユーザーを認証できるよう正しく構成されていると、パスワードの入力を促すプロンプトは表示されず、ユーザーは SSH サーバーに自動的にログインします。

ネットワークの保護

以下のセクションでは、IP セキュリティーのインストールと構成の方法、必要なネットワーク・サービスと不必要なネットワーク・サービスを識別する方法、ネットワーク・セキュリティーの監査およびモニターの方法について説明します。

TCP/IP セキュリティー

TCP/IP およびネットワーク・ファイルシステム (NFS) ソフトウェアのインストールが済んでいる場合は、ネットワークを介して通信するようにシステムを構成できます。

本書では、TCP/IP の基本的な概念ではなく、TCP/IP のセキュリティー関連の事項について説明します。TCP/IP のインストールおよび初期構成については、「ネットワークおよびコミュニケーションの管理」の『伝送制御プロトコル/インターネット・プロトコル』のセクションを参照してください。

システム管理者は、いくつかの理由から特定のセキュリティー・レベルを満たしていなければならない場合があります。 例えば、そのセキュリティー・レベルが会社の方針に関連する場合があります。 あるいは、システムが政府機関のシステムにアクセスする必要があり、したがって、ある特定のセキュリティー・レベルで通信する必要があるのかもしれません。 それらのセキュリティー規格は、ネットワーク、オペレーティング・システム、アプリケーション・ソフトウェア、あるいはシステムを管理しているユーザーが作成したプログラムにも適用される場合があります。

このセクションでは、TCP/IP により標準モードで、またセキュア・システムとして提供されるセキュリティー機能について説明し、さらにネットワーク環境に該当するセキュリティー上の考慮事項について説明します。

TCP/IP および NFS ソフトウェアをインストールした後、System Management Interface Tool (SMIT) **tcpip** 高速パスを使用して、システムを構成します。

dacinet コマンドについて詳しくは、「コマンド・リファレンス」を参照してください。

オペレーティング・システム固有のセキュリティー

ネットワーク・アクセス制御やネットワーク監査など、TCP/IP に使用できるセキュリティー機能の多くは、オペレーティング・システムを通して使用できるセキュリティー機能が基礎となっています。

以下のセクションで、TCP/IP セキュリティーについて概説します。

ネットワーク・アクセス制御**:**

ネットワーキングのためのセキュリティー・ポリシーは、オペレーティング・システムのセキュリティー・ポリシーを拡張したものであり、ユーザー認証、接続認証、およびデータ・セキュリティーで構成されます。

主要なコンポーネントは次のとおりです。

- ユーザー認証は、ユーザーがローカル・システムにログインする場合と同様にユーザー名とパスワードによって、リモート・ホストで提供されます。 **ftp**、**rexec**、**telnet** などのトラステッド TCP/IP コマンドは、オペレーティング・システムにおけるトラステッド・コマンドと同じ要件を持っており、同じ検査の対象となります。
- 接続認証は、リモート・ホストが、期待されるインターネット・プロトコル (IP) アドレスおよび名前を持つようにします。これによって、リモート・ホストは別のリモート・ホストを装うことができなくなります。
- データのインポートおよびエクスポートのセキュリティーは、同じセキュリティー・レベルおよび同じ権限レベルのネットワーク・インターフェース・アダプター間で、指定されたセキュリティー・レベルのデータの受け渡しを許可します。 例えば、最高機密データは、最上位のセキュリティー・レベルに設定されたアダプター間でしか受け渡すことができません。

ネットワーク監査**:**

TCP/IP には、監査サブシステムを使用してアプリケーション・プログラムを監査するネットワーク監査機能があります。

監査の目的は、システムのセキュリティーに影響を及ぼすアクションと、それらのアクションを行ったユーザーを記録することです。

以下のアプリケーション・イベントが監査の対象となります。

- ネットワークへのアクセス
- 接続
- データのエクスポート
- データのインポート

オブジェクトの作成と削除は、オペレーティング・システムによって監査されます。アプリケーション監査レコードは、カーネルによる冗長な監査を防止するために監査の延期と再開を行います。

トラステッド・パス、トラステッド・シェル、およびセキュア・アテンション・キー**:**

オペレーティング・システムは、無許可プログラムがユーザー端末からデータを読み取ることを防止するために、トラステッド・パス を提供します。 このパスは、パスワードの変更やシステムへのログインなど、システムとの安全な通信パスが必要なときに使用されます。

さらに、オペレーティング・システムは、トラステッド・シェル (**tsh**) を提供します。このシェルは、テストされて安全が確認されたトラステッド・プログラムだけを実行します。 TCP/IP はこれらの機能を両方ともサポートし、さらにセキュア・アテンション・キー (SAK) をサポートします。このキーはユーザーとシステムの間の安全な通信に必要な環境を確立します。 ローカル SAK は、 TCP/IP の使用中であればいつでも使用できます。 リモート SAK は、**telnet** コマンドを通して使用できます。

telnet でのローカル SAK は、他のオペレーティング・システムのアプリケーション・プログラムでこれが果たす機能と同じ機能を果たします。つまり、**telnet** プロセスと、その **telnet** が稼働中であった端末に関連付けられていた他のすべてのプロセスを終了させます。 しかし、telnet プログラムの内部では、トラステッド・パスを求める要求をリモート・システムへ送信するために、**telnet send sak** コマンド (**telnet** コマンド・モードであるときに) を使用できます。 また、**telnet set sak** コマンドを使用して、SAK 要求を開始する単一のキーを定義することもできます。

トラステッド・コンピューティング・ベースについての詳細は、 1 ページの『トラステッド・コンピューティング・ベース』を参照してください。

TCP/IP コマンドのセキュリティー

TCP/IP の一部のコマンドは、作動時に安全な環境を提供します。 そのコマンドは、**ftp**、**rexec**、および **telnet** です。

ftp 機能は、ファイル転送時のセキュリティーを提供します。 **rexec** コマンドは、外部ホスト上でコマンドを実行するために安全な環境を提供します。 **telnet** 機能は、外部ホストへログインするためのセキュリティーを提供します。

ftp、**rexec**、および **telnet** コマンドは、それぞれのコマンドの作動時にのみセキュリティーを提供します。 つまり、これらのコマンドは他のコマンド用に安全な環境をセットアップするわけではないということです。 他の操作についてシステムの安全を保護するためには、**securetcpip** コマンドを使用します。 こ

のコマンドを使用すると、システムの安全を保護するために、非トラステッド・デーモンや非トラステッド・アプリケーションを使用不可にしたり、 IP 層ネットワーク・プロトコルの安全を保護するオプションを使用したりすることができます。

ftp、**rexec**、**securetcpip**、**telnet** の各コマンドは、次の形態のシステム・セキュリティーとデータ・セキュリティーを提供します。

ftp **ftp** コマンドは、ファイル転送するための安全な環境を提供します。 外部ホストへの **ftp** コマンドを呼び出すと、ログイン ID の入力を求めるプロンプトが表示されます。デフォルトのログイン ID として、そのユーザーのローカル・ホスト上での現行ログイン ID が表示されます。 また、リモート・ホスト用のパスワードの入力を求めるプロンプトが表示されます。

自動ログイン・プロセスがそのローカル・ユーザーの **$HOME/.netrc** ファイルを検索し、外部ホストで使用するユーザーの ID とパスワードを見つけます。セキュリティーを確保するために、**$HOME/.netrc** ファイルについてのアクセス権は 600 (所有者による読み取りと書き込みのみ) に設定しておかなければなりません。 これを行わないと、自動ログインは失敗します。

注**:** .netrc ファイルを使用するには暗号化されていないファイルにパスワードを保管する必要があるため、システムを **securetcpip** コマンドで構成した場合には **ftp** コマンドの自動ログイン機能を利用することはできません。 この機能は、**/etc/security/config** ファイル内の tcpip スタンザから **ftp** コマンドを除去すると、再び使用可能にすることができます。

ftp コマンドでは、ファイル転送機能を使用するために 2 つの TCP/IP 接続が必要になります。 1 つはファイル転送プロトコル (FTP) 用で、もう 1 つはデータ転送用です。 プロトコル接続は 1 次接続であり、信頼できる通信ポート上で確立されるので安全です。 2 次接続は実際のデータ転送に必要で、ローカル・ホストとリモート・ホストはどちらも、この接続の相手側エンドが 1 次接続と同じホストを使用して確立されているかどうかを検査します。 1 次接続と 2 次接続が同じホストを使用して確立されていない場合、**ftp** コマンドはデータ接続が認証されなかったことを示すエラー・メッセージを表示した後、終了します。 この 2 次接続の検査により、第 3 のホストが別のホストあてのデータを代行受信できなくなります。

rexec **rexec** コマンドは、外部ホスト上でコマンドを実行するための安全な環境を提供します。 ユーザーは、ログイン ID とパスワードの両方の入力を求められます。

自動ログイン機能を使用すると、**rexec** コマンドはローカル・ユーザーの **$HOME/.netrc** ファイルを検索して、そのユーザーの外部ホスト用の ID とパスワードを見つけます。 セキュリティーを確保するために、**$HOME/.netrc** ファイルについてのアクセス権は 600 (所有者による読み取りと書き込みのみ) に設定しておかなければなりません。 これを行わないと、自動ログインは失敗します。

注**:** .netrc ファイルを使用するには暗号化されていないファイルにパスワードを保管する必要があるため、システムがセキュア・モードで作動している場合は **rexec** コマンドの自動ログイン機能を使用することはできません。 この機能は、**/etc/security/config** ファイルの tcpip スタンザからエントリーを除去すれば、再び使用可能にすることができます。

securetcpip

securetcpip コマンドは、TCP/IP セキュリティー機能を使用可能にします。 非トラステッド・コマンドへのアクセスは、このコマンドが入力された時点でシステムから除去されます。 次の各コマンドは、**securetcpip** コマンドを実行すると除去されます。

- **rlogin** および **rlogind**
- **rcp**、 **rsh**、および **rshd**
- **tftp** および **tftpd**

- **trpt**

securetcpip コマンドは、システムを標準レベルのセキュリティーから、さらに上位のセキュリティー・レベルに変換するために使用します。 一度システムが変換されたら、TCP/IP を再インストールする場合以外、**securetcpip** コマンドを再度入力する必要はありません。

telnet または **tn**

telnet (TELNET) コマンドは、外部ホストへログインするために安全な環境を提供します。 ユーザーは、ログイン ID とパスワードの両方の入力を求められます。 ユーザーの端末は、ホストへ直接接続した端末とまったく同様に扱われます。 つまり、端末へのアクセスは、許可ビットによって制御されます。 他のユーザー (グループと他のユーザー) は、その端末への読み取りアクセス権を持っていませんが、所有者から書き込み許可を得た場合はその端末へメッセージを書き込むことができます。 **telnet** コマンドを使用すると、 SAK を使用してリモート・システム上のトラステッド・シェルにアクセスすることもできます。 このキー・シーケンスは、ローカル・トラステッド・パスを起動するシーケンスとは異なり、**telnet** コマンドの内部で定義できます。

リモート・コマンドの実行アクセス:

/etc/hosts.equiv ファイルにリストされたホスト上のユーザーは、パスワードを入力せずにシステムで特定のコマンドを実行することができます。

次の表には、SMIT インターフェース、またはコマンド・ライン・インターフェースを使用して、リモート・ホストのリスト、追加、および除去を行う方法についての情報があります。

表 *14.* リモート・コマンドの実行アクセス・タスク

タスク	**SMIT** 高速パス	コマンドまたは ファイル
コマンド実行アクセス権があるリモート・ホストのリスト表示	**smit lshostsequiv**	view **/etc/hosts.equiv** ファイル
コマンド実行アクセスのためのリモート・ホストの追加	**smit mkhostsequiv**	edit **/etc/hosts.equiv** ファイル[注]
コマンド実行アクセスからのリモート・ホストの除去	**smit rmhostsequiv**	edit **/etc/hosts.equiv** ファイル[注]

注: これらのファイル手順の詳細については、「ファイル参照」の 『hosts.equiv File Format for TCP/IP』を参照してください。

ファイル転送プログラム・ユーザーの限定:

/etc/ftpusers ファイルにリストされたユーザーは、リモート FTP アクセスから保護されます。 例えば、ユーザー A がリモート・システムにログインし、さらにローカル・システム上のユーザー B のパスワードを知っているとします。ユーザー B が **/etc/ftpusers** ファイルにリストされている場合、ユーザー A がユーザー B のパスワードを知っていても、ユーザー A がユーザー B のアカウントとの間で FTP によるファイル転送を行うことはできません。

次の表には、SMIT、またはコマンド・ラインを使用して、限定されたユーザーのリスト、追加、および除去を行う方法についての情報があります。

リモート FTP ユーザー・タスク

タスク	SMIT 高速パス	コマンドまたは ファイル
限定された FTP ユーザーのリスト表示	**smit lsftpusers**	view **/etc/ftpusers** ファイル
限定されたユーザーの追加	**smit mkftpusers**	edit **/etc/ftpusers** ファイル[注]
限定されたユーザーの除去	**smit rmftpusers**	edit **/etc/ftpusers** ファイル[注]

注: これらのファイル手順についての詳細は、「ファイル参照」の『ftpusers File Format for TCP/IP』を参照してください。

トラステッド・プロセス

トラステッド・プログラムまたはトラステッド・プロセスは、特定のセキュリティー規格に合ったシェル・スクリプト、デーモン、プログラムのいずれかです。これらのセキュリティー規格は、いくつかのトラステッド・プログラムの認定もしている米国国防総省によって設定され、保守されます。

トラステッド・プログラムは、さまざまなレベルで保証されています。 セキュリティー・レベルには、A1、B1、B2、B3、C1、C2、D があり、レベル A1 が最上位のセキュリティー・レベルです。 各セキュリティー・レベルは、一定の要件を満たさなければなりません。例えば、C2 レベルのセキュリティーには、次の規格が組み込まれています。

プログラムの整合性
: 意図されたとおりに、そのプロセスが実行するようにします。

モジュール性
: プロセスの送信元コードが、他のモジュールから直接影響を受けたりアクセスされたりしない、いくつかのモジュールに分割されます。

最低特権の原則
: ユーザーが常に、認可された最低レベルの特権で操作することを規定しています。 つまり、ユーザーに一定のファイルを表示するだけのアクセス権しか与えなければ、誤ってそのユーザーにそのファイルを変更するアクセス権も与えてしまうことはありません。

オブジェクト再使用の制限
: 例えば、上書き用のフラグが付いていてもまだクリアされておらず、機密情報が入っている可能性があるメモリー・セクションを、ユーザーが偶然見つけてしまうことがないようにします。

TCP/IP には、いくつかのトラステッド・デーモンと多数の非トラステッド・デーモンが入っています。

以下は、トラステッド・デーモンの例です。

- **ftpd**
- **rexecd**
- **telnetd**

以下は、非トラステッド・デーモンの例です。

- **rshd**
- **rlogind**
- **tftpd**

トラステッド・システムであるためには、そのシステムがトラステッド・コンピューティング・ベースで動作しなければなりません。単一ホストの場合、そのマシンの安全が保護されていなければなりません。 ネットワークの場合は、ファイル・サーバー、ゲートウェイ、その他のホストの安全がすべて保護されていなければなりません。

ネットワーク・トラステッド・コンピューティング・ベース

ネットワーク・トラステッド・コンピューティング・ベース (NTCB) は、ネットワーク・セキュリティーを確保するためのハードウェアとソフトウェアで構成されています。 このセクションでは、 NTCB のコンポーネントを TCP/IP との関連で定義します。

ネットワークのハードウェア・セキュリティー機能は、TCP/IP に使用するネットワーク・アダプターによって提供されます。 それらのアダプターは、ローカル・システム宛てに送られたデータだけを受信することによって着信データを制御し、すべてのシステムに受信可能なデータをブロードキャストします。

NTCB のソフトウェア・コンポーネントは、トラステッドと見なされているプログラムだけで構成されています。 次の各表では、セキュア・システムの一部であるプログラムと、それに関連したファイルのリストをディレクトリーごとに示します。

/etc ディレクトリー

名前	所有者	グループ	モード	アクセス権
gated.conf	root	system	0664	rw-rw-r—
gateways	root	system	0664	rw-rw-r—
hosts	root	system	0664	rw-rw-r—
hosts.equiv	root	system	0664	rw-rw-r—
inetd.conf	root	system	0644	rw-r—r—
named.conf	root	system	0644	rw-r—r—
named.data	root	system	0664	rw-rw-r—
networks	root	system	0664	rw-rw-r—
protocols	root	system	0644	rw-r—r—
rc.tcpip	root	system	0774	rwxrwxr—
resolv.conf	root	system	0644	rw-rw-r—
services	root	system	0644	rw-r—r—
3270.keys	root	system	0664	rw-rw-r—
3270keys.rt	root	system	0664	rw-rw-r—

/usr/bin ディレクトリー

名前	所有者	グループ	モード	アクセス権
host	root	system	4555	r-sr-xr-x
hostid	bin	bin	0555	r-xr-xr-x
hostname	bin	bin	0555	r-xr-xr-x
finger	root	system	0755	rwxr-xr-x
ftp	root	system	4555	r-sr-xr-x
netstat	root	bin	4555	r-sr-xr-x
rexec	root	bin	4555	r-sr-xr-x
ruptime	root	system	4555	r-sr-xr-x
rwho	root	system	4555	r-sr-xr-x
talk	bin	bin	0555	r-xr-xr-x
telnet	root	system	4555	r-sr-xr-x

/usr/sbin ディレクトリー

名前	所有者	グループ	モード	アクセス権
arp	root	system	4555	r-sr-xr-x
fingerd	root	system	0554	r-xr-xr—
ftpd	root	system	4554	r-sr-xr—
gated	root	system	4554	r-sr-xr—
ifconfig	bin	bin	0555	r-xr-xr-x
inetd	root	system	4554	r-sr-xr—
named	root	system	4554	r-sr-x—
ping	root	system	4555	r-sr-xr-x
rexecd	root	system	4554	r-sr-xr—
route	root	system	4554	r-sr-xr—
routed	root	system	0554	r-xr-x—-
rwhod	root	system	4554	r-sr-xr—
securetcpip	root	system	0554	r-xr-xr—
setclock	root	system	4555	r-sr-xr-x
syslogd	root	system	0554	r-xr-xr—
talkd	root	system	4554	r-sr-xr—
telnetd	root	system	4554	r-sr-xr—

/usr/ucb ディレクトリー

名前	所有者	グループ	モード	アクセス権
tn	root	system	4555	r-sr-xr-x

/var/spool/rwho ディレクトリー

名前	所有者	グループ	モード	アクセス権
rwho (ディレクトリー)	root	system	0755	drwxr-xr-x

データ・セキュリティーと情報保護

TCP/IP のセキュリティー・フィーチャーでは、ネットワークを介して送信されるユーザー・データを暗号化しません。

ユーザーはパスワードやその他の機密情報の開示によって発生し得る通信中のリスクを識別し、そのリスクに基づいて適切な対応策を適用する必要があります。

TCP/IP セキュリティー・フィーチャーを米国国防総省 (DOD) 環境で使用するには、通信のセキュリティーに関する DOD 5200.5 および NCSD-11 を順守しなければならない場合があります。

インターネット・ポート用の任意アクセス制御によるユーザー・ベースの TCP ポート・アクセス制御

インターネット・ポート用の任意アクセス制御 (DACinet) は、AIX ホスト間での通信に関する、TCP ポートのユーザー・ベース・アクセス制御を提供します。

AIX は、追加の TCP ヘッダーを使用して、システム間でユーザーおよびグループ情報を転送することができます。 DACinet フィーチャーを使用することにより、宛先システム上の管理者は、宛先ポート、発信元のユーザー ID およびホストに基づいてアクセスを制御できます。

さらに、DACinet フィーチャーを使用することによって、管理者はローカル・ポートを root 専用に制限できます。 AIX などの UNIX システムは、1024 より下のポートを、root のみがオープンできる特権ポートとして扱います。 AIX では、1024 より上に root のみがオープンできる追加ポートを指定できます。したがって、ユーザーが既知のポートでサーバーを実行するのを防止することができます。

設定によって、非 DACinet システムが DACinet システムに接続できるかを任意に決められます。 DACinet フィーチャーの初期状態では、アクセスは拒否されます。 いったん DACinet を使用可能にした後で、DACinet を使用不可にする方法はありません。

dacinet コマンドは、ホスト名として指定されたアドレス、ドット 10 進数ホスト・アドレス、またはネットワーク・アドレスとそれに続くネットワーク・プレフィックスの長さを受け入れます。

次の例では、完全修飾ホスト名 *host.domain.org* で知られる単一のホストを指定しています。

```
host.domain.org
```

次の例では、IP アドレス 10.0.0.1 で知られる単一ホストを指定しています。

```
10.0.0.1
```

次の例では、最初の 24 ビット (ネットワーク・プレフィックスの長さ) を含んでいるネットワーク全体を、値 10.0.0.0 で指定しています。

```
10.0.0.0/24
```

このネットワークには、10.0.0.1 から 10.0.0.254 の間のすべての IP アドレスが含まれています。

TCP ベースのサービス用のアクセス制御**:**

DACinet は **/etc/rc.dacinet** 起動ファイルを使用します。また、使用する構成ファイルは、**/etc/security/priv**、**/etc/security/services**、および **/etc/security/acl** です。

/etc/security/services ファイルにリストされたポートは、ACL 検査を免除されているものと見なされます。 このファイルのフォーマットは **/etc/services** と同じです。このファイルを初期化する最も簡単な方法は、**/etc** から **/etc/security** にコピーしてから、ACL を適用するすべてのポートを削除する方法です。 ACL は 2 つの場所に保管されています。 現在アクティブになっている ACL はカーネルに保管されており、**dacinet aclls** を実行することによって読み取ることができます。 次のシステム・ブート時に **/etc/rc.tcpip** によって再活動化される ACL は、**/etc/security/acl** に保管されています。以下の形式が使用されます。

```
service host/prefix-length [user|group]
```

数値か **/etc/services** 内のリストのいずれかでサービスを指定できる場合、ホストは、ホスト名か、サブネット・マスクを指定したネットワーク・アドレスのいずれかで指定できます。ユーザーまたはグループは u: または g: プレフィックスを使用して指定されます。ユーザーまたはグループが指定されない場合、ACL は送信側ホストだけを考慮に入れます。 サービスにプレフィックス - を付けると、アクセスが明示的に使用不可になります。 最初の一致に従って、ACL は評価されます。 そのため、ユーザーのグループのアクセスを指定するものの、グループ・ルールの前にあるユーザーのためのルールを置くことにより、グループ内のこのユーザーのアクセスを明示的に拒否することができます。

/etc/services ファイルには、AIX でサポートされていないポート番号値を持つエントリーが 2 つ含まれています。システム管理者は **mkCCadmin** コマンドを実行する前に、そのファイルからこの 2 つのエントリーの行を除去しなければなりません。 **/etc/services** ファイルから以下の行を除去してください。

```
sco_printer     70000/tcp     sco_spooler    # For System V print IPC
sco_s5_port     70001/tcp     lpNet_s5_port  # For future use
```

***DACinet* の使用例:**

例えば、DACinet を使用して、ポート TCP/25 インバウンドへのアクセスを DACinet フィーチャーを使用している root だけに制限すると、他の AIX ホストからの root ユーザーだけがこのポートにアクセスできます。そのため、被害を受ける側でポート TCP/25 に対して Telnet を使用するだけで、通常のユーザーによって電子メールのスプーフが行われる可能性を制限することができます。

以下の例は、root 専用アクセス用に X プロトコル (X11) を構成する方法を示しています。 このサービスに ACL が適用されるようにするため、必ず **/etc/security/services** 内の X11 エントリーを除去してください。

接続されたすべてのシステムのサブネットを 10.1.1.0/24 と想定すると、**/etc/security/acl** 内の X (TCP/6000) について、アクセスを root ユーザーのみに制限する ACL 項目は以下のようになります。

```
      6000    10.1.1.0/24 u:root
```

Telnet サービスをグループ friends 内のユーザーに制限する場合 (そのサービスがどのシステムに由来するかは関係ない)、**/etc/security/services** から Telnet エントリーを除去した後で、以下の ACL 項目を使用してください。

```
telnet    0.0.0.0/0   g:friends
```

ユーザー fred 以外が Web サーバーにアクセスできるようにするには、以下のようにします。

```
-80    0.0.0.0/0 u:fred
80     0.0.0.0/0
```

ローカル・サービス実行用の特権ポート:

通常のユーザーが特定のポートでサーバーを実行できないようにするには、それらのポートを特権ポートとして指定します。

通常、どのユーザーでも 1024 より上の任意のポートをオープンすることができます。 例えば、ユーザーはポート 8080 (Web プロキシーを実行するために頻繁に使用される)、または 1080 (多くの場合、SOCKS サーバーが検出される) にサーバーを置くことができます。 **dacinet setpriv** コマンドを使用して、稼働中のシステムに特権ポートを追加することができます。 システム始動時に特権ポートとして指定されるポートは、**/etc/security/priv** ファイルにリストする必要があります。

このファイルにポートをリストするには、**/etc/services** で定義されているシンボル名を使用するか、またはポート番号を指定します。 次のエントリーでは、非 root ユーザーが通常のポートで SOCKS サーバーまたは Lotus Notes® サーバーを実行できなくなります。

```
1080
lotusnote
```

注: このフィーチャーは、ユーザーによるプログラムの実行を禁止するものではありません。 大抵の場合のサービスの実行が想定される既知のポートで、ユーザーがそれらのサービスを実行できなくなるだけです。

ネットワーク・サービス

オープン通信ポートを使用したネットワーク・サービスの識別および保護に関する情報が示されています。

ポート使用方法

以下の表は、AIX オペレーティング・システムでの既知のポート使用法に関する説明です。

注**:** このリストは、ソフトウェアの各種構成がインストールされた複数の AIX システムを検討して構築されました。

以下のリストには、AIX オペレーティング・システムに存在するすべてのソフトウェアのポート使用法が含まれていない可能性があります。

ポート/プロトコル	サービス名	別名
13/tcp	daytime	Daytime (RFC 867)
13/udp	daytime	Daytime (RFC 867)
21/tcp	ftp	ファイル転送 [Control]
21/udp	ftp	ファイル転送 [Control]
23/udp	telnet	Telnet
23/udp	telnet	Telnet
25/tcp	smtp	Simple Mail Transfer
25/udp	smtp	Simple Mail Transfer
37/tcp	time	Time
37/udp	time	Time
111/tcp	sunrpc	SUN Remote Procedure Call
111/udp	sunrpc	SUN Remote Procedure Call
161/tcp	snmp	SNMP
161/udp	snmp	SNMP
199/tcp	smux	SMUX
199/udp	smux	SMUX
512/tcp	exec	リモート・プロセス実行
513/tcp	login	Telnet リモート・ログイン
514/tcp	shell	cmd
514/udp	syslog	Syslog
518/tcp	ntalk	Talk
518/udp	ntalk	Talk
657/tcp	rmc	RMC
657/udp	rmc	RMC
1334/tcp	writesrv	writesrv
1334/udp	writesrv	writesrv
2279/tcp	xmquery	xmquery
2279/udp	xmquery	xmquery
32768/tcp	filenet-tms	FileNet® TMS
32768/udp	filenet-tms	FileNet TMS
32769/tcp	filenet-rpc	FileNet RPC
32769/udp	filenet-rpc	FileNet RPC
32770/tcp	filenet-nch	FileNet NCH
32770/udp	filenet-nch	FileNet NCH
32771/tcp	filenet-rmi	FileNet RMI
32771/udp	filenet-rmi	FileNet RMI
32772/tcp	filenet-pa	FileNet Process Analyzer
32772/udp	filenet-pa	FileNet Process Analyzer
32773/tcp	filenet-cm	FileNet Component Manager
32773/udp	filenet-cm	FileNet Component Manager
32774/tcp	filenet-re	FileNet Rules Engine

ポート/プロトコル	サービス名	別名
32774/udp	filenet-re FileNET Rules Engine	FileNet Rules Engine
32775/tcp	filenet-pch	Performance Clearinghouse
32775/udp	filenet-pch	Performance Clearinghouse
32776/tcp	filenet-peior	FileNet BPM IOR
32776/udp	filenet-peior	FileNet BPM IOR
32777/tcp	filenet-obrok	FileNet BPM CORBA
32777/udp	filenet-obrok	FileNet BPM CORBA

オープン通信ポートを使用するネットワーク・サービスの識別

サーバーでクライアント/サーバー・アプリケーションのオープン通信ポートを使用することにより、アプリケーションが着信クライアント要求を listen できます。

オープン・ポートは、起こり得るセキュリティー上の攻撃にぜい弱なため、オープン・ポートがあるアプリケーションを識別し、不必要にオープンしているポートは閉じてください。 これを実践することにより、インターネットへアクセスできる人であれば誰でも使用できるようになっているのはどのシステムかを理解できるので役立ちます。

どのポートがオープンしているかを判断するには、次のステップを実行します。

1. 次のように **netstat** コマンドを使用して、サービスを識別します。

   ```
   # netstat -af inet
   ```

   以下に、このコマンドの出力例を示します。 **netstat** コマンドの出力の最後の列は、各サービスの状態を示しています。 着信接続を待機しているサービスは、LISTEN 状態になっています。

これは、**netstat** コマンド実行時のコマンド出力の例です。

```
アクティブなインターネット接続 (サーバーを含む)
プロトコル      受信キュー          送信キュー  ローカル・アドレス      外部アドレス        (状態)
tcp4        0           0                   *.echo              *.*                 LISTEN
tcp4        0           0                   *.discard           *.*                 LISTEN
tcp4        0           0                   *.daytime           *.*                 LISTEN
tcp         0           0                   *.chargen           *.*                 LISTEN
tcp         0           0                   *.ftp               *.*                 LISTEN
tcp4        0           0                   *.telnet            *.*                 LISTEN
tcp4        0           0                   *.smtp              *.*                 LISTEN
tcp4        0           0                   *.time              *.*                 LISTEN
tcp4        0           0                   *.www               *.*                 LISTEN
tcp4        0           0                   *.sunrpc            *.*                 LISTEN
tcp         0           0                   *.smux              *.*                 LISTEN
tcp         0           0                   *.exec              *.*                 LISTEN
tcp         0           0                   *.login             *.*                 LISTEN
tcp4        0           0                   *.shell             *.*                 LISTEN
tcp4        0           0                   *.klogin            *.*                 LISTEN
```

これは、**netstat** コマンド実行時のコマンド出力の例です。

```
アクティブなインターネット接続 (サーバーを含む)
プロトコル      受信キュー              送信キュー  ローカル・アドレス              外部アドレス          (状態)
udp4        0           0                   *.kshell                    *.*                 LISTEN
udp4        0           0                   *.echo                      *.*
udp4        0           0                   *.discard                   *.*
udp4        0           0                   *.daytime                   *.*
udp4        0           0                   *.chargen                   *.*
udp4        0           0                   *.time                      *.*
udp4        0           0                   *.bootpc                    *.*
udp4        0           0                   *.sunrpc                    *.*
udp4        0           0                   255.255.255.255.ntp         *.*
udp4        0           0                   1.23.123.234.ntp            *.*
udp4        0           0                   localhost.domain.ntp        *.*
udp4        0           0                   name.domain..ntp            *.*
.....
```

2. **/etc/services** ファイルを開いて、Internet Assigned Numbers Authority (IANA) サービスを確認し、オペレーティング・システム内のポート番号にサービスをマップします。

   /etc/services ファイルの一部のサンプルを次に示します。

```
tcpmux  1/tcp  # TCP Port Service Multiplexer
tcpmux  1/tcp  # TCP Port Service Multiplexer
Compressnet  2/tcp  # Management Utility
Compressnet  2/udp  # Management Utility
Compressnet  3/tcp  # Compression Process
Compressnet  3/udp  Compression Process
Echo  7/tcp
Echo  7/udp
discard  9/tcp  sink null
discard  9/udp  sink null
..............
rfe  5002/tcp  # Radio Free Ethernet
rfe  5002/udp  # Radio Free Ethernet
rmonitor_secure  5145/tcp
rmonitor_secure  5145/udp
pad12sim  5236/tcp
pad12sim  5236/udp
sub-process  6111/tcp  # HP SoftBench Sub-Process Cntl.
sub-process  6111/udp  # HP SoftBench Sub-Process Cntl.
xdsxdm  6558/ucp
xdsxdm  6558/tcp
afs3-fileserver  7000/tcp  # File Server Itself
afs3-fileserver  7000/udp  # File Server Itself
af3-callback  7001/tcp  # Callbacks to Cache Managers
af3-callback  7001/udp  # Callbacks to Cache Managers
```

3. 実行中のサービスを除去して、不要なポートをクローズします。

注**:** ポート 657 は、ノード間の通信のために、Resource Monitoring and Control (RMC) によって使用されます。 このポートをブロックしたり、制限することはできません。

TCP および UDP ソケットの識別

LISTEN 状態の TCP ソケットと、データの到着を待機しているアイドル UDP ソケットを識別するには、**netstat -af** コマンドの変種である **lsof** コマンドを使用します。

例えば、LISTEN 状態の TCP ソケットおよび IDLE 状態の UDP ソケットを表示するには、次のように **lsof** コマンドを実行します。

```
# lsof -i | egrep "COMMAND|LISTEN|UDP"
```

生成される出力は、以下のようになります。

コマンド	PID	ユーザー	FD	タイプ	デバイス	サイズ/オフ	ノード	名前
dtlogin	2122	root	5u	IPv4	0x70053c00	0t0	UDP	*:xdmcp
dtlogin	2122	root	6u	IPv4	0x70054adc	0t0	TCP	*:32768(LISTEN)
syslogd	2730	root	4u	IPv4	0x70053600	0t0	UDP	*:syslog
X	2880	root	6u	IPv4	0x70054adc	0t0	TCP	*:32768(LISTEN)
X	2880	root	8u	IPv4	0x700546dc	0t0	TCP	*:6000(LISTEN)
dtlogin	3882	root	6u	IPv4	0x70054adc	0t0	TCP	*:32768(LISTEN)
dtgreet	4656	root	6u	IPv4	0x70054adc	0t0	TCP	*:32768(LISTEN)

プロセス ID を識別した後で、以下のコマンドを実行することにより、プログラムに関するより詳細な情報を入手できます。

```
" # ps -fp PID#"
```

その出力には、プログラムのマニュアル・ページへアクセスするのに使用できる、コマンド名へのパスが含まれています。

インターネット・プロトコルのセキュリティー

IP セキュリティーは、IP 層でのデータ・トラフィックを確保することにより、インターネット上および企業ネットワーク間セキュア通信を可能にします。

IP セキュリティーの概要

IP セキュリティーを使用すると、アプリケーションを変更することなく、個別のユーザーまたは組織がすべてのアプリケーションのトラフィックを保護することができます。 したがって、電子メールなどのデータまたはアプリケーション固有の企業データの送信を保護することができます。

IP セキュリティーとオペレーティング・システム:

オペレーティング・システムでは、Internet Engineering Task Force (IETF) によって開発された、オープンでスタンダードなセキュリティー・テクノロジーである IP セキュリティー (IPsec) が使用されます。

IPsec は、通信スタックの IP 層において、暗号に基づくすべてのデータの保護を提供します。 既存のアプリケーションを変更する必要はありません。 IPsec は、IP バージョン 4 と 6 の両方の環境のために IETF によって選択された、業界標準のネットワーク・セキュリティー・フレームワークです。

IPsec は、次のような暗号化手法を使用してデータ・トラフィックを保護します。

認証　ホストまたはエンドポイントの ID を確認するプロセス

保全性検査
: ネットワークでの転送中に、データに対して変更が行われないことを保証するプロセス

暗号化
: ネットワークでの転送中に、データおよび私用 IP アドレスを「隠す」ことによって、プライバシーを守るプロセス

認証アルゴリズムは、暗号ハッシュ関数を使用して、秘密鍵を使用するデータ・パケット (固定の IP ヘッダー・フィールドを含む) を処理し、固有のダイジェストを作成します。これにより、送信側の ID とデータ保全性が証明されます。 受信側で、データは同じ機能と鍵を使用して処理されます。 データが変更されているか、または送信側の鍵が無効である場合、データグラムは廃棄されます。

暗号化は、暗号アルゴリズムを使用し、暗号テキストと呼ばれる暗号化データを作成するための特定のアルゴリズムと鍵を使用して、データを変更し、ランダム化します。 暗号化によって、転送中のデータは読み取り不能になります。 データは、受信されると、同じアルゴリズムと鍵を使用して (対称的な暗号化アルゴリズムを使用して) リカバリーされます。 暗号化されたデータの保全性を確保するには、暗号化と認証の両方を行う必要があります。

IPsec の場合、上記の基本サービスは、 Encapsulating Security Payload (ESP) および Authentication Header (AH) を使用して実行されます。 ESP は、元の IP パケットを暗号化し、ESP ヘッダーを作成し、暗号テキストを ESP ペイロードに置くことによって、機密性を確保します。

機密が問題でないときは、認証と完全性の検査に AH のみを使用することができます。 AH を使用すれば、IP ヘッダーの静的フィールドおよびデータにハッシュ・アルゴリズムが適用され、鍵付きダイジェストが計算されます。 受信側は、その鍵を使用してダイジェストの計算および比較を行って、パケットが変更されていないこと、および送信側の ID が認証されていることを確認します。

IP セキュリティー・フィーチャー**:**

以下は、IP セキュリティーのフィーチャーです。

AIX オペレーティング・システムの IKE (Internet Key Exchange) では、次のフィーチャーが使用可能です。

- AES 128 ビット、192 ビット、および 256 ビット・アルゴリズムをサポートします。
- 10/100 Mbps イーサネット・アダプター II (PCI) でのハードウェア・アクセラレーション
- RFC 2402 を使用した AH サポートおよび RFC 2406 を使用した ESP サポート
- マニュアル・トンネルは、IKE による自動キー・リフレッシュ方式をサポートしていない他のシステムと相互運用できるように構成することができる。また、IP バージョン 6 トンネルを使用するように構成することもできます。
- ホストまたはゲートウェイ・トンネルのためのカプセル化のトンネル・モードとトランスポート・モード
- HMAC (Hashed Message Authentication Code) MD5 (メッセージ・ダイジェスト 5) および HMAC SHA (Secure Hash Algorithm)
- 暗号化アルゴリズムには、64 ビットの初期ベクトル (IV) を持つ 56 ビット DES (Data Encryption Standard) CBC (Cipher Block Chaining)、Triple-DES、DES CBC 4 (32 ビット IV)、および AES CBC が含まれます。
- 二重 IP スタックのサポート (IP バージョン 4 および IP バージョン 6)

- IP バージョン 4 および IP バージョン 6 のトラフィックは、カプセル化とフィルター操作を行うことができる。 IP スタックは個々に分かれているので、各スタックの IP セキュリティー機能は個別に構成できます。
- さまざまな IP 特性 (送信元および宛先 IP アドレス、インターフェース、プロトコル、ポート番号など) によるセキュア・トラフィックと非セキュア・トラフィックのフィルター操作
- ほとんどのトンネル・タイプに対する自動的なフィルター・ルールの作成および削除
- トンネルおよびフィルター・ルール定義時の宛先アドレスに対するホスト名の使用。 このホスト名は、自動的に IP アドレスに変換されます (DNS が使用可能である場合)。
- **syslog** への IP セキュリティー・イベントのロギング
- 問題判別のためのシステム・トレースおよび統計の使用
- ユーザー定義のデフォルトのアクションにより、ユーザーは、定義したトンネルに一致しないトラフィックを許可するかどうかを指定できる。

IKE (Internet Key Exchange) (AIX 6.1 TL 05 以降) では、次の追加フィーチャーが使用可能です。
- RFC 4301 を使用した IPSec サポート、RFC 4302 を使用した AH サポート、および RFC 4303 を使用した ESP サポート
- Cipher based Message Authentication Code (CMAC) AES XCBC の認証アルゴリズム
- AES 128 ビット、192 ビット、256 ビット GCM (16 ビット IV)、AES-128-GMAC、AES-192-GMAC、および AES-256-GMAC を含む暗号化アルゴリズム
- フィルター・ルールに対応したポート範囲のサポート
- 拡張シーケンス番号

***IKE (Internet Key Exchange)* フィーチャー:**

AIX の IKE (Internet Key Exchange) では、次のフィーチャーが使用可能です。

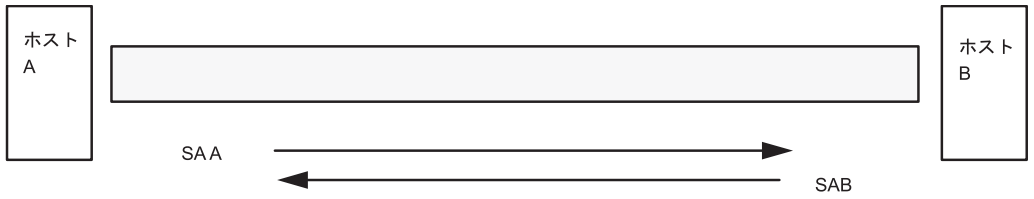
AIX 6.1 以降の IKE (Internet Key Exchange) では、次の追加フィーチャーが使用可能です。
- HMAC SHA2 256 ビット・ハッシュの AH サポート (TL 04 以降)
- GCM AES 128 ビット、192 ビット、256 ビット (16 ビット IV)、GMAC AES 128 ビット、192 ビット、256 ビットのアルゴリズムを使用する ESP 暗号化サポート。HMAC MD5 および HMAC SHA1 による ESP 認証のサポート (TL 04 以降)。
- IKEv1 (RFC2409) および IKEv2 (RFC4306) がサポートされます (TL 02 以降)。 IKEv1 は **isakmpd** デーモンによってサポートされ、 IKEv2 は **ikev2d** デーモンによってサポートされます (TL 02 以降)。 IKEv1 と IKEv2 トンネルの共存は可能です。
- 保全性アルゴリズム CMAC_AES_XCBC および HMAC_SHA2_256 のサポート (TL 04 以降)
- PRF アルゴリズム PRF_SHA2_256 のサポート (TL 04 以降)
- Diffie Hellman グループ 14、19 および 24 のサポート (TL 04 以降)

セキュリティー・アソシエーション:

セキュア通信は、セキュリティー・アソシエーションと呼ばれる概念に基づいて構築されます。セキュリティー・アソシエーションは、セキュリティー・パラメーターの特定のセットを 1 つのトラフィックのタイプに関連付けます。

IP セキュリティーによって保護されたデータを使用する場合は、方向とヘッダー・タイプ (AH または ESP) ごとに、別個のセキュリティー・アソシエーションが存在します。 セキュリティー・アソシエーションに含まれる情報には、パーティーの IP アドレス、セキュリティー・パラメーター索引 (SPI) と呼ば

れる固有の ID、認証や暗号化に選択されたアルゴリズム、認証キーと暗号キー、およびキー・ライフタイムなどがあります。 次の図は、ホスト A とホスト B の間のセキュリティー・アソシエーションを示しています。



SA = セキュリティー・アソシエーション。以下で構成される。

宛先アドレス
SPI
キー
暗号のアルゴリズムとフォーマット
認証アルゴリズム
キー・ライフタイム

図 *6.* ホスト *A* とホスト *B* 間のセキュア・トンネルの設定

この図は、ホスト A とホスト B の間で実行される仮想トンネルを示しています。セキュリティー・アソシエーション A は、ホスト A からホスト B へ向かう矢印です。セキュリティー・アソシエーション B は、ホスト B からホスト A へ向かう矢印です。セキュリティー・アソシエーションは、宛先アドレス、SPI、キー、暗号のアルゴリズムとフォーマット、認証アルゴリズム、およびキー・ライフタイムから構成されます。

鍵管理の目的は、IP トラフィックを保護するセキュリティー・アソシエーションのネゴシエーションと計算です。

トンネルおよび鍵管理**:**

トンネルは、2 つのホスト間にセキュア通信をセットアップするために必要なセキュリティー・アソシエーションのネゴシエーションおよび管理に使用します。

以下のタイプのトンネルがサポートされています。それぞれには、異なった鍵管理手法が使用されます。

- IKE トンネル (動的鍵交換、IETF 標準)
- マニュアル・トンネル (静的な固定鍵、IETF 標準)

IKE (Internet Key Exchange) トンネル・サポート**:**

IKE トンネルは、IETF によって作成された ISAKMP/Oakley (Internet Security Association and Key Management Protocol) 標準に基づいています。 このプロトコルを使用して、セキュリティー・パラメーターのネゴシエーションとリフレッシュが行われ、鍵が安全に交換されます。

次のタイプの認証がサポートされます。

- 事前共有鍵
- X.509v3 デジタル証明書署名
- AIX 6.1 TL 04 以降、IKEv2 では、デジタル証明書に基づいた X509v3 認証方式の一環として ECDSA-256 デジタル証明書署名がサポートされています。

ネゴシエーションは 2 フェーズのアプローチを使用します。 フェーズ 1 では、パーティー双方の認証が行われ、フェーズ 2 で通信を安全に行うために使用するアルゴリズムが指定されます。フェーズ 2 では、データ転送時に使用する IP セキュリティー・パラメーターについてのネゴシエーションと、セキュリティー・アソシエーションおよび鍵の作成と交換が行われます。

次の表は、IKE トンネルのサポートに AH および ESP セキュリティー・プロトコルで使用できる認証アルゴリズムを示しています。

表 *15. IKE* トンネル・サポートの認証アルゴリズム

アルゴリズム	**AH IP** バージョン **4 & 6**	**ESP IP** バージョン **4 & 6**
HMAC MD5	X	X
HMAC SHA1	X	X
DES CBC 8		X
Triple-DES CBC		X
AES CBC (128、192、256)		X
ESP null		X
AES-XCBC-MAC-96	X	X
AES GCM (128, 192, 256)		X
AES GMAC (128, 192, 256)	X	
ESP_ENCR_NULL_ AUTH_AES_GMAC		X

マニュアル・トンネル・サポート**:**

マニュアル・トンネルには後方互換性があり、 IKE キー管理プロトコルをサポートしていないマシンとの相互運用が可能です。 マニュアル・トンネルの欠点は、キー値が静的であることです。 暗号キーと認証キーがトンネルのライフタイム中同じであり、手動で更新する必要があります。

次の表は、マニュアル・トンネルのサポートに AH および ESP セキュリティー・プロトコルで使用できる認証アルゴリズムを示しています。

アルゴリズム	**AH IP** バージョン **4**	**AH IP** バージョン **6**	**ESP IP** バージョン **4**	**ESP IP** バージョン **6**
HMAC MD5	X	X	X	X
HMAC SHA1	X	X	X	X
AES CBC (128、192、256)			X	X
Triple-DES CBC			X	X
DES CBC 8			X	X
DES CBC 4			X	X

IKE トンネルではより効果的なセキュリティーが提供されるため、キー管理の方法としては IKE が推奨されます。

ネイティブ・フィルター操作機能**:**

フィルター操作 は、さまざまな特性に基づいて着信パケットおよび発信パケットを許可または拒否できる基本機能です。 これにより、ユーザーまたはシステム管理者はホストを構成して、そのホストと他のホストとの間のトラフィックを制御できます。

フィルター処理は、送信元および宛先アドレス、IP バージョン (4 または 6)、サブネット・マスク、プロトコル、ポート、経路指定特性、フラグメント化、インターフェース、トンネル定義など、さまざまなパケット属性について実行されます。

特定の種類のトラフィックを特定のトンネルに関連付けるために、フィルター・ルール と呼ばれるルールが使用されます。 マニュアル・トンネルの場合の基本構成では、ユーザーがホスト間トンネルを定義すると、そのホストからのトラフィックがすべてセキュア・トンネルを介して送信されるようにするためにフィルター・ルールが自動生成されます。 より特殊なタイプのトラフィック (例えばサブネット間のトラフィックなど) が必要な場合は、特定のトンネルを使用するトラフィックを厳密に制御できるように、これらのフィルター・ルールを編集または置換することができます。

IKE トンネルの場合も、トンネルがアクティブにされたときにフィルター・ルールが自動的に生成され、フィルター・テーブルに挿入されます。

同様に、トンネルを変更または削除すると、そのトンネルのフィルター・ルールは自動的に削除されます。これにより、IP セキュリティー構成が単純化され、人間の介入によるエラーが削減されます。 トンネル定義は、インポートとエクスポート・ユーティリティーを使用して、各マシンとファイアウォールに伝搬させて共有することができます。これは、多数のマシンを管理する際に役立ちます。

フィルター・ルールは、特定のタイプのトラフィックをトンネルに関連付けるために必要ですが、フィルターに掛けられるデータは、必ずしもトンネルを通る必要はありません。 このフィルター・ルールの性質により、このオペレーティング・システムでは、基本的なファイアウォール機能が提供されます。これは、真のファイアウォールによる保護を受けていないイントラネットまたはネットワーク内にあるマシンとの間でトラフィックの流れを制限しようとしている方々に役立ちます。 この場合、フィルター・ルールは一群のマシンを取り巻く第 2 の保護バリアとなります。

フィルター・ルールは、生成されると、テーブルに保管されて、カーネルにロードされます。 パケットが送信可能になるか、またはネットワークから受信可能になると、リスト内の上から下までフィルター・ルールが検査され、パケットを許可するか、拒否するか、またはトンネルを介して送信するべきかが決定されます。 一致するものが見つかるか、またはデフォルトのルールに到達するまで、ルールの基準がパケット特性と比較されます。

IP セキュリティー機能には、ユーザー定義のきわめて詳細な基準に基づいた、非セキュア・パケットに対するフィルター操作も実装されています。これは、IP セキュリティーの認証属性または暗号化属性を必要としないネットワーク間およびマシン間の IP トラフィックの制御を可能にする便利な機能です。

デジタル証明書サポート:

IP セキュリティーは、X.509 バージョン 3 デジタル証明書の使用をサポートします。

Key Manager ツールは、証明書要求の管理、キー・データベースの保守、およびその他の管理機能の実行を行います。

デジタル証明書については、「デジタル証明書の構成」に説明があります。 Key Manager とその機能については、「IBM Key Manager ツールの使用法」に説明があります。

仮想プライベート・ネットワークと IP セキュリティー:

仮想プライベート・ネットワーク (VPN) は、インターネットなどの公衆ネットワーク上のプライベート・イントラネットの安全性を拡張します。

VPN は、基本的には専用トンネルであるものを介して、リモートのユーザー、事業所、および提携会社/取引先の情報をインターネット経由で伝送します。 企業は、より高価な専用回線、長距離電話、およびフリーダイヤル電話番号を使用することなく、直通回線または近距離電話を使用してインターネット・サービス・プロバイダー (ISP) 経由でインターネットにアクセスすることができます。 IPsec は、IP バージョン 4 と 6 の両方の環境用に IETF によって選択された、業界標準のネットワーク・セキュリティー・フレームワークであるため、 VPN ソリューションでは IPsec セキュリティー規格が使用できます。これを使用する場合、既存のアプリケーションに対して変更を行う必要はありません。

AIX オペレーティング・システムにおける VPN 実装を計画するための資料として、「*A Comprehensive Guide to Virtual Private Networks, Volume III: Cross-Platform Key and Policy Management*」(ISBN SG24-5309-00) の Chapter 9 をお勧めします。この資料は、インターネット上からも入手可能です (http://www.redbooks.ibm.com/redbooks/SG245309.html)。

IP セキュリティー・フィーチャーのインストール

AIX の IP セキュリティー・フィーチャーは、個別にインストールしてロードすることができます。

インストールする必要があるのは、次のファイルセットです。

- `bos.net.ipsec.rte` (カーネル IP セキュリティー環境用のランタイム環境とコマンド)
- `bos.msg.`*LANG*`.net.ipsec` (*LANG* は指定する言語、例えば en_US)
- **bos.net.ipsec.keymgt**
- `clic.rte` (C 用の CryptoLite、 DES、triple DES および AES 暗号化用のファイルセット)

また、IKE デジタル署名をサポートするには、`gskit.rte` ファイルセット または `gskkm.rte` を拡張パックからインストールする必要があります。

ファイルセットをインストールした後、『IP セキュリティーのロード』の推奨手順または **mkdev** コマンドを使用して、IP バージョン 4 用と IP バージョン 6 用に IP セキュリティーを個別にロードすることができます。

IP セキュリティーのロード**:**

IP セキュリティーが開始されたら、SMIT を使用して、IP セキュリティー・モジュールを自動的にロードします。 さらに、SMIT を使用すると、カーネル拡張および IKE デーモンが確実に正しい順序でロードされます。

注**:** IP セキュリティーをロードすると、フィルター機能が使用可能になります。 ロードを行うときは、まず、適切なフィルター・ルールを確実に作成しておくことが重要です。 これを行わないと、外部通信がすべてブロックされてしまう恐れがあります。

ロードが正常に完了すると、**lsdev** コマンドは、IP セキュリティー・デバイスを `Available` と表示します。

```
lsdev -C -c ipsec

   ipsec_v4 Available IP Version 4 Security Extension
   ipsec_v6 Available IP Version 6 Security Extension
```

IP セキュリティー・カーネル拡張がロードされると、いつでもトンネルやフィルターを構成できるようになります。

IP セキュリティー構成の計画

IP セキュリティーを構成するには、まずトンネルおよびフィルターの構成を計画します。

すべてのトラフィックが使用する単純なトンネルを定義している場合、フィルター・ルールを自動的に生成できます。 より複雑なフィルター操作が必要な場合は、フィルター・ルールを個別に構成できます。

IP セキュリティーは、仮想プライベート・ネットワーク・プラグイン、または System Management Interface Tool (SMIT) を使用して構成することができます。 SMIT を使用する場合は、次の高速パスが使用可能です。

smit ips4_basic
IP バージョン 4 用の基本構成

smit ips6_basic
IP バージョン 6 用の基本構成

特定のサイトで IP セキュリティーを構成するときは、まず、どんな方式を使用するつもりかを決定しておく必要があります。例えば、トンネルとフィルターのどちら (あるいはその両方) を使用したいのか、どんなタイプのトンネルが最も必要に適しているのか、といった点を決定しておかなければなりません。 以下のセクションでは、これらの決定を下す前に理解しておく必要のある情報を提供します。

ハードウェア・アクセラレーション:

10/100 Mbps イーサネット PCI アダプター II (フィーチャー・コード 4962) は、規格に基づいた IP セキュリティーを備えており、AIX オペレーティング・システムの IP セキュリティー機能をオフロードするように設計されています。

AIX システムに 10/100 Mbps イーサネット PCI アダプター II がある場合、IP セキュリティー・スタックは、アダプターの次の機能を使用します。

- DES または Triple DES アルゴリズムを使用した暗号化および暗号化解除
- MD5 または SHA-1 アルゴリズムを使用した認証
- セキュリティー・アソシエーション情報の保管

アダプターの機能は、ソフトウェア・アルゴリズムの代わりに使用されます。 10/100 Mbps イーサネット・アダプター II (PCI) は、マニュアル・トンネルと IKE トンネルに使用可能です。

IP セキュリティーのハードウェア・アクセラレーション・フィーチャーは、5.1.0.25 以降のレベルの **bos.net.ipsec.rte** および **devices.pci.1410ff01.rte** ファイルセットで使用できます。

受信側 (インバウンド・トラフィック) では、ネットワーク・アダプターにオフロードできるセキュリティーの関連付けの数に、制限はありません。送信側 (アウトバウンド・トラフィック) では、サポートされている構成を使用するすべてのパケットがアダプターにオフロードされます。 ただし、トンネル構成の中には、一部アダプターにオフロードできないものもあります。

10/100 Mbps イーサネット PCI アダプター II では、以下のフィーチャーがサポートされます。

- ESP を使用した DES、3DES、または NULL 暗号化
- ESP または AH のいずれか一方のみを使用した HMAC-MD5 または HMAC-SHA-1 認証 (ESP と AH を両方使用する場合は、最初に ESP を実行する必要があります。 これは、IKE トンネルで使用する場合は常にそうです。ただし、マニュアル・トンネルで使用する場合は、任意の順番を選択できます。)
- トランスポートおよびトンネル・モード
- IPV4 パケットのオフロード

注**:** 10/100 Mbps イーサネット・アダプター II (PCI) では、 IP オプションでパケットをハンドルできません。

10/100 Mbps イーサネット・アダプター II (PCI) を IP セキュリティーに使用できるようにするためには、ネットワーク・インターフェースを切り離し、IPsec オフロード機能を有効にする必要があります。

ネットワーク・インターフェースを切り離すには、SMIT インターフェースを使用して次のステップを実行します。

IPsec オフロード機能を有効にするには、SMIT インターフェースを使用して以下を行います。

1. **root** ユーザーとしてログインします。
2. コマンド・ラインに `smitty eadap` と入力し、Enter を押します。
3. 「**Change / Show Characteristics of an Ethernet Adapter (**イーサネット・アダプターの特性の変更**/**表示**)**」オプションを選択し、Enter を押します。
4. 10/100 Mbps イーサネット PCI アダプター II を選択し、Enter を押します。
5. 「IPsec Offload (IPsec オフロード)」フィールドを「yes (はい)」に変更して、Enter を押します。

コマンド・ラインからネットワーク・インターフェースの切り離しを行う場合は、次のように入力します。

```
# ifconfig enX detach
```

コマンド・ラインから IPsec オフロードの属性を有効にする場合は、次のように入力します。

```
# chdev -l entX -a ipsec_offload=yes
```

IPsec オフロード属性が有効になったかどうかをコマンド・ラインから確認するには、次のように入力します。

```
# lsattr -El entX detach
```

コマンド・ラインから IPsec オフロード属性を無効にする場合は、次のように入力します。

```
# chdev -l entX -a ipsec_offload=no
```

enstat コマンドを使用して、トンネル構成で IPsec オフロード属性が利用されていることを確認します。IPsec オフロード属性が使用可能になっていれば、**enstat** コマンドにより IPsec パケットの送受信の統計がすべて表示されます。 例えば、イーサネット・インターフェースが **ent1** の場合、次のように入力します。

```
# entstat -d ent1
```

出力は次の例のようになります。

```
.
.
.
10/100 Mbps Ethernet PCI Adapter II (1410ff01) Specific Statistics:
---------------------------------------------
.
.
.
Transmit IPsec packets: 3
Transmit IPsec packets dropped: 0
Receive IPsec packets: 2
Receive IPsec packets dropped: 0
```

ネットワーク・チューナブル・パラメーター:

ご使用の構成内のトンネルの数に応じて、ソケットの最大バッファー・サイズを増やすことができます。

ご使用の環境で多数のトンネルが稼働しており、**sb_max** チューナブル・パラメーターがデフォルト値のままである場合、ネットワーク内の負荷が高いため、IKE デーモン・プロセスおよびトンネル・マネージャーのデーモン・プロセスが応答を停止する可能性があります。

sb_max チューナブル・パラメーターに以下の値を使用することが必要になる場合があります。

- 500 個のトンネルの場合は 10 MB
- 1000 個のトンネルの場合は 20 MB

関連情報:

sb_max チューナブル

トンネルとフィルター:

IP セキュリティーは、トンネル とフィルター という 2 つの別々の部分で構成されています。トンネルにとってフィルターは必須ですが、フィルターにとってトンネルは必須ではありません。

フィルター操作 は、ルール と呼ばれるさまざまな特性に基づいて、着信パケットおよび発信パケットを許可または拒否できる機能です。この機能を使用することにより、システム管理者は、ホストを構成してそのホストと他のホストとの間のトラフィックを制御できます。 フィルター処理は、送信元および宛先アドレス、IP バージョン (4 または 6)、サブネット・マスク、プロトコル、ポート、経路指定特性、フラグメント化、インターフェース、トンネル定義など、さまざまなパケット属性について実行されます。 このフィルター処理は、IP 層で実行されるので、アプリケーションを変更する必要はありません。

トンネル は、2 つのホスト間のセキュリティー・アソシエーションを定義します。 このセキュリティー・アソシエーションには、トンネルのエンドポイント間で共有される、特定のセキュリティー・パラメーターが含まれています。

次の図は、パケットがネットワーク・アダプターから IP スタックに入る様子を示します。 そこからフィルター・モジュールが呼び出されて、パケットを許可するか拒否するかが決定されます。 トンネル ID を指定した場合、既存のトンネル定義に対してパケットが検査されます。 トンネルからのカプセル解除が正常である場合は、パケットは上層プロトコルに渡されます。この機能は、発信パケットに対しては逆順で生じます。 トンネルはフィルター・ルールに基づいてパケットを特定のトンネルに関連付けますが、フィルター操作機能は、パケットをトンネルに渡さずに実行することができます。

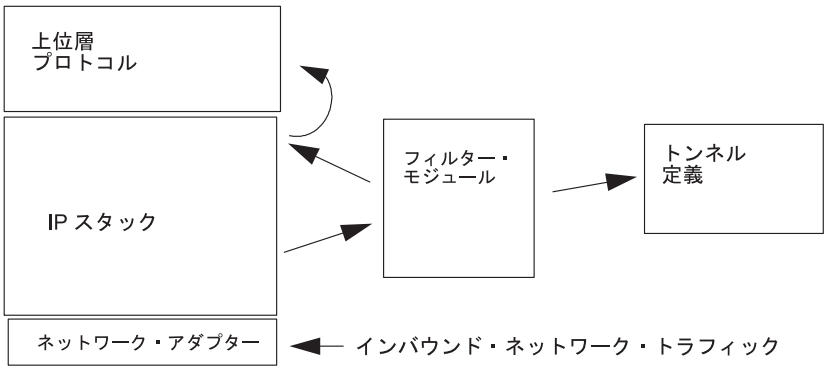


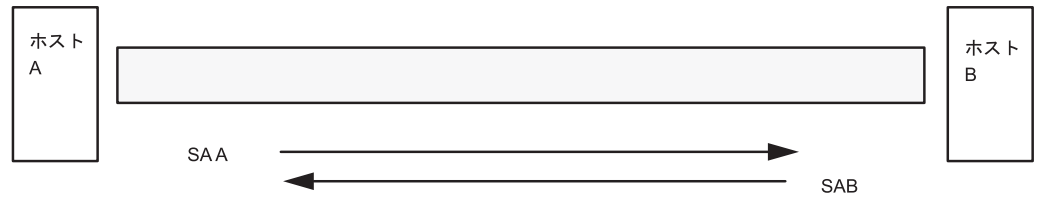
図 7. ネットワーク・パケットの経路指定

この図には、ネットワーク・パケットがたどる経路が示されています。ネットワークからのインバウンド・パケットは、ネットワーク・アダプターに入ります。 そこから IP スタックに入り、さらにフィルター・モジュールに送られます。 このフィルター・モジュールから、パケットはトンネル定義に送られるか、あるいは IP スタックに戻されて上層プロトコルに転送されます。

トンネルとセキュリティー・アソシエーション:

データを認証するか、またはデータを認証して暗号化する必要があるときは、常にトンネルを使用します。トンネルは、2 つのホスト間のセキュリティー・アソシエーションを指定して定義します。 このセキュリティー・アソシエーションは、暗号化と認証アルゴリズムおよびトンネルの特性のためのパラメーターを定義します。

次の図は、ホスト A とホスト B の間の仮想トンネルを示しています。



SA = セキュリティー・アソシエーション。以下で構成される。

宛先アドレス
SPI
キー
暗号のアルゴリズムとフォーマット
認証アルゴリズム
キー・ライフタイム

図 8. ホスト *A* とホスト *B* 間のセキュア・トンネルの設定

この図は、ホスト A とホスト B の間で実行される仮想トンネルを示しています。セキュリティー・アソシエーション A は、ホスト A からホスト B へ向かう矢印です。セキュリティー・アソシエーション B は、ホスト B からホスト A へ向かう矢印です。セキュリティー・アソシエーションは、宛先アドレス、SPI、鍵、暗号のアルゴリズムとフォーマット、認証アルゴリズム、およびキー・ライフタイムから構成されます。

セキュリティー・パラメーター索引 (SPI) と宛先アドレスは、固有のセキュリティー・アソシエーションを識別します。 これらのパラメーターは、トンネルを一意的に指定するのに必要です。 その他にも、暗号アルゴリズム、認証アルゴリズム、鍵、ライフタイムなどのパラメーターを指定したり、デフォルトのパラメーターを使用したりできます。

トンネルに関する考慮事項:

IP セキュリティーに使用するトンネルのタイプを決定する前に、いくつかの事柄を考慮する必要があります。

セキュリティー・ポリシーの構成は、トンネル・エンドポイントの定義とは別のプロセスであるため、IKE トンネルとマニュアル・トンネルは異なります。

IKE には、2 ステップのネゴシエーション・プロセスがあります。 ネゴシエーション・プロセスの各ステップはフェーズ と呼ばれ、フェーズごとに別々のセキュリティー・ポリシーを設定できます。

インターネット鍵のネゴシエーションが開始すると、そのネゴシエーション用のセキュア・チャネルがセットアップされます。 これを鍵管理 フェーズ、またはフェーズ *1* と呼びます。このフェーズでは、パーティーはそれぞれ事前共有鍵またはデジタル証明書を使用して、相手を認証し、ID 情報を渡します。このフェーズでは、二人のパーティーは、どのようにして安全に通信するかを決定するセキュリティー・アソシエーションをセットアップし、2 番目のフェーズでの通信に使用する保護を決定します。 このフェーズの結果が *IKE* または フェーズ *1* トンネルです。

2 番目のフェーズは、データ管理 フェーズ、またはフェーズ 2 と呼ばれ、IKE トンネルを使用して、実際にトラフィックを保護する AH および ESP のセキュリティー・アソシエーションを作成します。 2 番目のフェーズは、IP セキュリティー・トンネルを使用するデータも決定します。 例えば、以下を指定できます。

- サブネット・マスク
- アドレスの範囲
- プロトコルとポート番号の組み合わせ

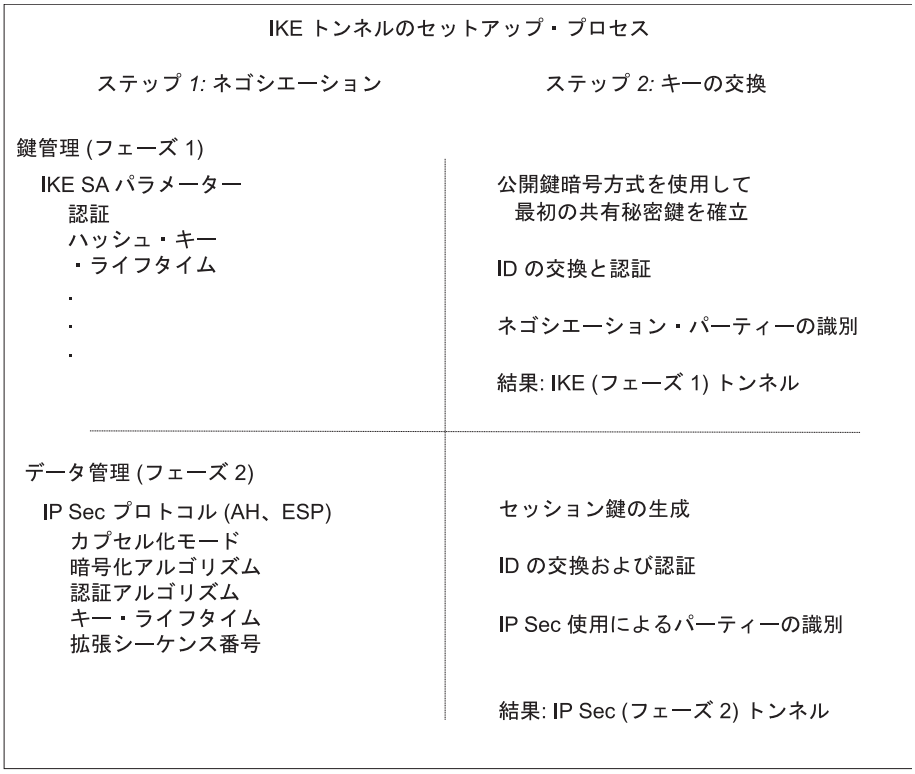


図 *9. IKE* トンネルのセットアップ・プロセス

この図には、IKE トンネルをセットアップするための 2 つのステップから成る 2 フェーズ・プロセスが示されています。

注**:** IKEv2 にも 2 つのフェーズがあります。 1 番目のフェーズは *IKE SA* フェーズまたはフェーズ *1*、2 番目のフェーズは *CHILD SA* フェーズまたは フェーズ *2* と呼ばれています。 両方のトンネルが IKEv1 で確立される方式とは違って、フェーズ 1 トンネルが IKEv2 で確立されるときに、フェーズ 2 トンネルが自動的に活動化されます。 IKEv2 トンネルの構成は IKEv1 トンネルの構成に類似しています。

通常は、鍵管理 (IKE) トンネルのエンドポイントは、データ管理 (IP セキュリティー) トンネルのエンドポイントと同じです。 IKE トンネルのエンドポイントは、ネゴシエーションを実行するマシンの ID です。 IP セキュリティー・トンネルのエンドポイントは、IP セキュリティー・トンネルを使用するトラフィックのタイプを示します。 2 つのトンネル間のすべてのトラフィックが同一のトンネルで保護されるような、単純なホスト間トンネルでは、フェーズ 1 およびフェーズ 2 トンネルのエンドポイントは同じです。 ネゴシエーションするパーティーが 2 つのゲートウェイであれば、IKE トンネルのエンドポイントは 2 つのゲートウェイであり、さらに IP セキュリティー・トンネルのエンドポイントは、マシンまたはサブネット (ゲートウェイの背後の)、またはトンネル・ユーザーのアドレスの範囲 (ゲートウェイの背後の) です。

鍵管理パラメーターおよびポリシー*:*

ユーザーは、IKE ネゴシエーション中に使用されるパラメーターを指定して、鍵管理ポリシーをカスタマイズできます。 例えば、事前共有鍵認証または署名モード認証用の、鍵管理ポリシーがあります。フェーズ 1 で、交換を実行するための、特定の鍵管理セキュリティー属性を決定する必要があります。

フェーズ 1 (鍵管理フェーズ) では、IKE トンネル構成の次のパラメーターを設定します。

鍵管理 (フェーズ 1) トンネル
この IKE トンネルの名前。 それぞれのトンネルごとに、ネゴシエーションのエンドポイントが指定されなければなりません。 これらは、IKE メッセージの送信および妥当性検査を計画する 2 つのマシンです。 トンネルの名前は、VPN Boston または VPN Acme のように、トンネルのエンドポイントを表すことができます。

ホスト ID タイプ
IKE 交換で使用される ID タイプ。 ID タイプおよび値は、適切な鍵検索が実行できるように、事前共有鍵の値と一致する必要があります。 事前共有鍵値の検索に別の ID が使用される場合は、ホスト *ID* は鍵の ID で、そのタイプ は KEY_ID です。 KEY_ID タイプは、単一のホストが複数の事前共有鍵値を持つ場合に便利です。

ホスト ID
IP アドレス、完全修飾ドメイン名 (FQDN)、または、完全修飾ドメイン名におけるユーザー (*user@FQDN*) として表されるホスト ID の値です。 例えば、jdoe@studentmail.ut.edu とします。

IP アドレス
リモート・ホストの IP アドレス。 この値は、ホスト ID のタイプが KEY_ID の場合、またはホスト ID のタイプが 1 つの IP アドレスとして解決できない場合に必要です。 例えば、ユーザー名がローカル・ネーム・サーバーで解決できない場合、リモート側の IP アドレスを入力する必要があります。

データ管理パラメーターおよびポリシー*:*

データ管理プロポーザル・パラメーターは、IKE トンネル構成のフェーズ 1 で設定されます。 これらのパラメーターは、マニュアル・トンネルで使用されるのと同じ IP セキュリティー・パラメーターで、トンネル内のデータ・トラフィックを保護するために使用される保護のタイプを記述しています。 同じフェーズ 1 トンネルの下で、複数のフェーズ 2 トンネルを開始することができます。

以下のエンドポイント ID タイプは、 IP セキュリティー・データ・トンネルを使用するデータのタイプを表しています。

ホスト、サブネット、または範囲
トンネルを通過するデータ・トラフィックが、特定のホスト用か、サブネット用か、またはアドレス範囲用であるかを記述します。

ホスト/サブネット **ID**
このトンネルにトラフィックを渡すローカル・システム、およびリモート・システムの、ホスト ID、またはサブネット ID を含んでいます。 フェーズ 2 ネゴシエーションで送信される ID、および、ネゴシエーションが正常に行われた場合に作成されるフィルター・ルールを決定します。

サブネット・マスク
サブネット内のすべての IP アドレスを記述します (例えば、ホスト 9.53.250.96 およびマスク 255.255.255.0)。

開始 **IP** アドレスの範囲
トンネルを使用するアドレス範囲の開始 IP アドレスを指定します (例えば、9.53.250.96 から 9.53.250.93 の範囲の 9.53.250.96)。

終了 **IP** アドレスの範囲
トンネルを使用するアドレス範囲の終了 IP アドレスを指定します (例えば、9.53.250.96 から 9.53.250.93 の範囲の 9.53.250.93)。

ポート
特定のポート番号を使用するデータを記述します (例えば、21 または 23)。

プロトコル
: 特定のプロトコルでトランスポートされるデータを記述します (例えば、TCP または UDP)。 フェーズ 2 ネゴシエーションで送信されるプロトコル、および、ネゴシエーションが正常に行われた場合に作成されるフィルター・ルールを決定します。 ローカル・エンドポイントのプロトコルは、リモート・エンドポイントのプロトコルと一致する必要があります。

エンド・ポート
: データ伝送用のエンド・ポートを記述します (例えば、100 または 500)。 デフォルトでは、65355 がエンド・ポートです。

制約事項: IKEv2 の場合、IPv4 または IPv6 のアドレスの範囲のみをトラフィック・セレクターとして使用してください。 エンド・ポートは、IKEv2 および AIX 6.1 TL 04 以降にのみ適用されます。

トンネル・タイプの選択:

マニュアル・トンネルを使用するか、IKE トンネルを使用するかの判断は、リモート・エンドのトンネル・サポートおよび必要な鍵管理タイプによって異なります。

IKE トンネルは、業界標準のセキュア鍵ネゴシエーションおよび鍵リフレッシュを備えているので、使用可能な場合は IKE トンネルを使用することをお勧めします。 このトンネルは、IETF ESP および AH ヘッダー・タイプを利用し、再生防止保護もサポートします。オプションで、署名モードを構成することにより、デジタル証明書を使用することができます。

リモート・エンドが、マニュアル・トンネルを必要とするいずれかのアルゴリズムを使用する場合は、マニュアル・トンネルを使用してください。 マニュアル・トンネルは、多数のホスト間の相互運用性を保証します。 キーは、静的で、変更しにくく、更新に手間取ることがあるので、安全ではありません。マニュアル・トンネルは、本オペレーティング・システムが稼働中のホストと、IP セキュリティーが稼働中で、暗号および認証アルゴリズムの共通セットを持つ、他のマシンとの間で使用できます。 ほとんどのベンダーが、DES を使用した Keyed MD5 または HMAC MD5 を提供しています。 これは、ほとんどの IP セキュリティーのインプリメンテーションで作動する基本サブセットです。

マニュアル・トンネルを設定する際の手順は、そのトンネルの最初のホストを設定するのか、最初のホストの設定と一致するパラメーターが必要な 2 番目のホストを設定するのかによって異なります。 最初のホストを設定する際は、キーを自動生成でき、そのアルゴリズムをデフォルトにすることができます。 2 番目のホストを設定するときは、可能であれば、リモート・エンドからトンネル情報をインポートしてください。

リモート・システムがファイアウォールに隠れているかどうかの判断も考慮すべき重要な点です。 リモート・システムがある場合、ファイアウォールの介入に関する情報を設定に指定しなければなりません。

IKE と DHCP または動的割り当てアドレスの併用:

オペレーティング・システムで IP セキュリティーを使用する一般的なシナリオの例として、リモート・システムがサーバーとの IKE セッションを開始するときに、その ID を特定の IP アドレスに結び付けることができない場合があります。

これは、ローカル・エリア・ネットワーク (LAN) 環境において、 IP セキュリティーを使用して LAN 上のサーバーに接続し、データを暗号化するような場合に生じます。 その他の一般的な使用としては、リモート・クライアントがサーバーにダイヤルするときに、リモート ID の識別に完全修飾ドメイン名 (FQDN) または電子メール・アドレス (user@FQDN) を使用している場合の例を挙げることができます。

鍵管理フェーズ (フェーズ 1) においては、非 IP アドレスを ID としてメインモードを使用する場合、RSA 署名がサポートされる唯一の認証モードです。 換言すれば、事前共有鍵認証を使用したい場合は、アグレッシブ・モード、または IP アドレスを ID として、メインモードを使用する必要があります。 事実、IPsec トンネルを確立したい DHCP クライアントの数が大きい場合、それぞれの DHCP クライアントに対して固有の、事前共有鍵を定義することは非現実的です。したがって、このシナリオでは RSA 署名認証を使用することをお勧めします。 トンネル定義において、すべての DHCP クライアントについてトンネルを 1 回だけ定義すれば済むように、リモート ID としてグループ ID を使用することもできます (トンネル定義サンプル・ファイル **/usr/samples/ipsec/group_aix_responder.xml** を参照)。グループ ID は AIX IPsec の固有のフィーチャーです。任意の IKE ID (単一 IP アドレスのような)、FQDN、ユーザー FQDN、ある範囲内の、または一連の IP アドレスなどを含めるために 1 つのグループ ID を定義し、その後、このグループ ID を、トンネル定義におけるフェーズ 1 またはフェーズ 2 リモート ID として使用することができます。

注**:** グループ ID を使用する場合は、トンネルはレスポンダー・ロール専用として定義する必要があります。 つまり、このトンネルを DHCP クライアント側から起動する必要があることを意味します。

IP セキュリティー・アソシエーションが作成され、 TCP または UDP トラフィックが暗号化されるデータ管理フェーズ (フェーズ 2) では、汎用データ管理トンネルを構成できます。 このため、フェーズ 1 で認証された要求はいずれも、IP アドレスがデータベースに明示的に構成されていないとき、定義済みデータ管理フェーズ用の汎用トンネルを使用します。 これにより、いずれのアドレスも汎用トンネルに一致させることができ、フェーズ 1 において厳密な公開鍵ベースのセキュリティー検証が正常に行われている限り、そのアドレスを使用することができます。

汎用データ管理トンネル定義のための ***XML*** の使用*:*

汎用データ管理トンネルは、**ikedb** が認識する XML フォーマットを使用して定義することが可能です。

IKE XML インターフェースおよび **ikedb** コマンドに関する詳細については、 258 ページの『IKE トンネル構成のためのコマンド・ライン・インターフェース』のセクションを参照してください。汎用データ管理トンネルは、DHCP で使用されます。 XML フォーマットでは、IPSecTunnel という名前のタグが使用されます。これは、フェーズ 2 トンネル という名前で呼ばれることもあります。 汎用データ管理トンネル は、実際のトンネルではなく、着信データ管理メッセージ (特定のキー管理トンネルの下) がキー管理トンネルに定義されたいずれのデータ管理トンネルとも一致しない場合に使用される、IPSecProtection です。 汎用データ管理トンネルは、応答側が AIX システムであるケースでしか使用されません。 汎用データ管理トンネル IPSecProtection の指定はオプションです。

汎用データ管理トンネルは、IKEProtection エレメントで定義されます。 この定義に使用される XML 属性は、*IKE_IPSecDefaultProtectionRef* と *IKE_IPSecDefaultAllowedTypes* の 2 つです。

まず、IPSecProtection の定義を行う必要があります。これは、一致する IPSecTunnels (データ管理トンネル) がない場合にデフォルトとして使用されるものです。 デフォルトとして使用する IPSecProtection には、`_defIPSprot_` で始まる IPSec_ProtectionName が必要です。

次に、このデフォルトの IPSecProtection を使用したい IKEProtection に進みます。 デフォルト IPSec_Protection の名前が含まれている **IKE_IPSecDefaultProtectionRef** 属性を指定してください。

この IKEProtection の **IKE_IPSecDefaultAllowedTypes** 属性にも、値を指定する必要があります。 これには、以下の 1 つ以上の値を使用できます (複数の値を指定する場合は、値をスペースで区切ってください)。

```
Local_IPV4_Address
Local_IPV6_Address
Local_IPV4_Subnet
Local_IPV6_Subnet
Local_IPV4_Address_Range
Local_IPV6_Address_Range
Remote_IPV4_Address
Remote_IPV6_Address
Remote_IPV4_Subnet
Remote_IPV6_Subnet
Remote_IPV4_Address_Range
Remote_IPV6_Address_Range
```

これらの値は、イニシエーターが指定した ID のタイプに対応しています。 IKE ネゴシエーションでは、実際の ID は無視されます。 指定された IPSecProtection は、**IKE_IPSecDefaultAllowedTypes** 属性にイニシエーターのローカル ID タイプと対応する Local_ で始まる文字列が含まれ、イニシエーターのリモート ID タイプと対応する Remote_ で始まる文字列が含まれる場合に使用されます。 つまり、すべての **IKE_IPSecDefaultAllowedTypes** 属性には、対応する IPSec_Protection が使用される順番で、少なくとも 1 つの Local_ と少なくとも 1 つの Remote_ 値が指定されていなければならないといえます。

汎用データ管理トンネルの例*:*

データ管理トンネルは、システムにメッセージを送信するために使用することができます。

起動側は、フェーズ 2 (データ管理) メッセージで次の情報を AIX システムに送信します。

```
local ID type:    IPV4_Address
local ID:         192.168.100.104

remote ID type:   IPV4_Subnet
remote ID:        10.10.10.2
remote netmask:   255.255.255.192
```

AIX システムには、これらの ID と一致するデータ管理トンネルがありません。 しかし、次の属性が定義された IPSecProtection があります。

```
IKE_IPSecDefaultProtectionRef="_defIPSprot_protection4"
IKE_IPSecDefaultAllowedTypes="Local_IPV4_Address
                              Remote_IPV4_Address
                              Remote_IPV4_Subnet
                              Remote_IPV4_Address_Range"
```

着信メッセージのローカル ID タイプ IPV4_Address が、許可されているタイプ Local_IPV4_Address の Local_ 値の 1 つと一致します。 また、メッセージのリモート ID IPV4_Subnet が値 Remote_IPV4_Subnet と一致します。 これにより、データ管理トンネルのネゴシエーションは、IPSecProtection として _defIPSprot_protection4 に進みます。

/usr/samples/ipsec/default_p2_policy.xml ファイルは、汎用 IPSecProtection を定義した完全な XML ファイルで、サンプルとして使用できます。

IKE (Internet Key Exchange) トンネルの構成

SMIT (System Management Interface Tool) またはコマンド・ラインを使用して IKE (Internet Key Exchange) トンネルを構成することができます。

IKE トンネル構成のための **SMIT** インターフェースの使用*:*

SMIT インターフェースを使用して IKE トンネルを構成し、基本的な IKE データベース関数を実行することができます。

SMIT は、基礎的な XML コマンド関数を使用して、IKE トンネル定義に対する追加、削除、および変更を行います。 また、IKE トンネルを簡単に構成する場合は、IKE SMIT を使用します。 IKE SMIT では、XML 構文のサンプルを使用して IKE トンネル定義を作成できます。 IKE SMIT のメニューには、IKE データベースのバックアップ、復元、および初期化を行うオプションも用意されています。

IPv4 IKE トンネルを構成するには、**smitty ike4** 高速パスを使用します。 IPv6 IKE トンネルを構成するには、**smitty ike6** 高速パスを使用します。 IKE データベース関数は、「Advanced IP Security Configuration (拡張 IP セキュリティー構成)」メニューから呼び出せます。

IKE トンネル構成のためのコマンド・ライン・インターフェース**:**

ikedb コマンドを使用すると、 IKE データベース内の情報の検索、更新、削除、インポート、およびエクスポートを XML インターフェース上から行えます。

ikedb コマンドでは、 IKE データベースへの書き込み (put) とデータベースからの読み取り (get) が有効です。 入出力に使用されるファイル・フォーマットは、XML (Extensible Markup Language) ファイルです。 XML ファイルのフォーマットは、DTD (Document Type Definition) で定義されます。 **ikedb** コマンドでは、put 時に XML ファイルの妥当性検査に使用される DTD を確認できます。 ただし、**-e** フラグを使用して DTD にエンティティーの宣言を追加できる場合、その DTD に対して行える変更操作は、この宣言の追加のみです。 入力 XML ファイル内の外部 DOCTYPE 宣言はすべて無視されます。また、内部 DOCTYPE 宣言もすべてエラーになる可能性があります。 DTD を使用して XML ファイルの構文解析を行う際のルールは、XML の規格で指定されています。 **/usr/samples/ipsec** ファイルは、一般的なトンネルのシナリオを定義する典型的な XML ファイルの例です。構文の詳細については、「コマンド・リファレンス」の『**ikedb** コマンド』の説明を参照してください。

IKE トンネルの開始、停止、およびモニターには、**ike** コマンドを使用できます。 また、この **ike** コマンドでは、IKE トンネルや IP セキュリティー・トンネルの活動化、除去、またはリストも行えます。構文の詳細については、「コマンド・リファレンス」の『**ike** コマンド』の説明を参照してください。

次の例は、**ike** コマンドと **ikedb** コマンド、そして IKE トンネルの状況を構成および検査する他のいくつかのコマンドの使用法を示しています。

1. トンネル・ネゴシエーションを開始する (トンネルの開始) か、または着信システムを応答側として機能させる (指定された役割に応じて) には、以下で示すように、**ike** コマンドにトンネル番号を指定して使用します。

   ```
   # ike cmd=activate numlist=1
   ```

   以下の例で示すように、リモート ID または IP アドレスを使用することもできます。

   ```
   # ike cmd=activate remid=9.3.97.256
   # ike cmd=activate ipaddr=9.3.97.100, 9.3.97.256
   ```

   このコマンドの終了には数秒かかる場合があるため、コマンドは、ネゴシエーションの開始後に戻ります。
2. トンネルの状況を表示する場合は、**ike** コマンドを次のように使用します。

   ```
   # ike cmd=list
   ```

   出力は、以下のようなものになります。

   ```
   Phase 1 Tunnel ID       [1]
   Phase 2 Tunnel ID       [1]
   ```

   出力は、フェーズ 1 およびフェーズ 2 のトンネルが現在アクティブになっていることを示しています。

3. トンネルの冗長リストを表示するには、**ike** コマンドを次のように使用します。

```
# ike cmd=list verbose
```

出力は、以下のようなものになります。

```
Phase 1 Tunnel ID       1
Local ID Type:          Fully_Qualified_Domain_Name
Local ID:               bee.austin.ibm.com
Remote ID Type:         Fully_Qualified_Domain_Name
Remote ID:              ipsec.austin.ibm.com
Mode:                   Aggressive
Security Policy:        BOTH_AGGR_3DES_MD5
Role:                   Initiator
Encryption Alg:         3DES-CBC
Auth Alg:               Preshared Key
Hash Alg:               MD5
Key Lifetime:           28800 Seconds
Key Lifesize:           0 Kbytes
Key Rem Lifetime:       28737 Seconds
Key Rem Lifesize:       0 Kbytes
Key Refresh Overlap:    5%
Tunnel Lifetime:        2592000 Seconds
Tunnel Lifesize:        0 Kbytes
Tun Rem Lifetime:       2591937 Seconds
Status:                 Active

Phase 2 Tunnel ID       1
Local ID Type:          IPv4_Address
Local ID:               10.10.10.1
Local Subnet Mask:      N/A
Local Port:             any
Local Protocol:         all
Remote ID Type:         IPv4_Address
Remote ID:              10.10.10.4
Remote Subnet Mask:     N/A
Remote Port:            any
Remote Portocol:        all
Mode:                   Oakley_quick
Security Policy:        ESP_3DES_MD5_SHA_TUNNEL_NO_PFS
Role:                   Initiator
Encryption Alg:         ESP_3DES
AH Transform:           N/A
Auth Alg:               HMAC-MD5
PFS:                    No
SA Lifetime:            600 Seconds
SA Lifesize:            0 Kbytes
SA Rem Lifetime:        562 Seconds
SA Rem Lifesize:        0 Kbytes
Key Refresh Overlap:    15%
Tunnel Lifetime:        2592000 Seconds
Tunnel Lifesize:        0 Kbytes
Tun Rem Lifetime:       2591962 Seconds
Assoc P1 Tunnel:        0
Encap Mode:             ESP_tunnel
Status:                 Active
```

4. 新しくアクティブにした IKE トンネルの動的フィルター・テーブルにあるフィルター・ルールを表示するには、**lsfilt** コマンドを次のように使用します。

```
# lsfilt -d
```

出力は次の例のようになります。

```
1 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no udp eq 4001 eq 4001 both both no all
  packets 0 all
2 *** Dynamic filter placement rule *** no
0 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 yes all any 0 any 0 both both no all
```

```
  packets 0 all

*** Dynamic table ***

0 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no udp eq 500 eq 500 local both no all
  packets 0
0 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no ah any 0 any 0 both inbound no all
  packets 0
0 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no esp any 0 any 0 both inbound no all
  packets 0
1 permit 10.10.10.1 255.255.255.255 10.10.10.4 255.255.255.255 no all any 0 any
  0 both outbound yes all packets 1
1 permit 10.10.10.4 255.255.255.255 10.10.10.1 255.255.255.255 no all any 0 any
  0 both inbound yes all packets 1
```

この例のマシンは、1 つの IKE トンネル以外にトンネルはなにも持っていません。 動的フィルター配置ルール (この静的テーブルの出力例の 2 番目のルール) をユーザーが移動して、他のすべてのユーザー定義のルールとの相対的配置を制御することができます。 動的テーブル内のルールは、トンネルがネゴシエーションされ、対応するルールがフィルター・テーブルに挿入される際に、自動的に作成されます。 これらのルールは、表示できますが、編集はできません。

5. 動的フィルター・ルールのロギングをオンにするには、次の例で示すように、 2 番目のルールに対するロギング・オプションを Yes に設定し、**chfilt** コマンドを使用します。

   ```
   # chfilt -v 4 -n 2 -l y
   ```

   IKE トラフィックのロギングに関する詳細は、 287 ページの『ロギング機能』を参照してください。
6. トンネルを活動停止するときは、**ike** コマンドを次のように使用します。

   ```
   # ike cmd=remove numlist=1
   ```
7. トンネル定義を表示するには、**ikedb** コマンドを次のように使用します。

   ```
   # ikedb -g
   ```
8. ピア・マシンで生成された XML ファイルから IKE データベースに定義を書き込み、データベース内に存在する同名の既存オブジェクトを上書きする場合は、**ikedb** コマンドを次のように使用します。

   ```
   # ikedb -pFs peer_tunnel_conf.xml
   ```

   `peer_tunnel_conf.xml` は、ピア・マシンで生成された XML ファイルです。
9. *tunnel_sys1_and_sys2* という名前のフェーズ 1 トンネルとそれに従属するすべてのフェーズ 2 トンネルの定義を、それぞれのプロポーザルや保護と合わせて取り出すには、**ikedb** コマンドを次のように使用します。

   ```
   # ikedb -gr -t IKETunnel -n tunnel_sys1_and_sys2
   ```
10. データベースからすべての事前共用キーを削除する場合は、**ikedb** コマンドを次のように使用します。

    ```
    # ikedb -d -t IKEPresharedKey
    ```

IKE トンネル・グループのサポートに関する一般情報は、 261 ページの『グループのサポート』を参照してください。コマンド・ラインからのグループの定義には、**ikedb** コマンドを使用できます。

AIX IKE と Linux Affinity:

Linux 構成ファイルを使用して AIX IKE トンネルを構成することが可能です。

Linux 構成ファイルを使用して AIX IKE トンネルを構成するには、**-c** フラグ (変換オプション) を指定して **ikedb** コマンドを使用します。このようにすると、`/etc/ipsec.conf` および `/etc/ipsec.secrets`

Linux 構成ファイルを IKE トンネル定義として使用することができます。 この **ikedb** コマンドは、Linux 構成ファイルを構文解析し、 XML ファイルを作成し、さらにオプションで XML トンネル定義を IKE データベースに追加します。 このコマンドの使用後、**ikedb -g** コマンドを使用してトンネル定義を表示できます。

グループのサポート**:**

IP セキュリティーは 1 つのトンネル定義内への IKE ID のグループ化をサポートしています。これにより、複数の ID を単一のセキュリティー・ポリシーに関連付けられるようになり、別々のトンネル定義を作成する必要がなくなりました。

グループ化が特に有用なのは、複数のリモート・ホストへの接続をセットアップする場合です。なぜなら、複数のトンネル定義をセットアップしたり管理したりしなくても済むからです。また、セキュリティー・ポリシーに変更を加える必要がある場合にも、複数のトンネル定義を変更する必要がありません。

グループは、そのグループ名をトンネル定義内で使用する前に、定義する必要があります。グループのサイズは 1 KB までに制限されています。 ネゴシエーションの開始側では、データ管理トンネル定義でのみ、グループをリモート ID として使用できます。ネゴシエーションの応答側では、キー管理およびデータ管理トンネル定義で、グループをリモート ID として使用できます。

グループは、グループ名と、IKE ID および ID タイプのリストから構成されます。 ID は、同じタイプにすることもできますし、以下のタイプを混合させても構いません。

- IPv4 アドレス
- IPv6 アドレス
- FQDN
- user@FQDN
- X500 DN タイプ

セキュリティー・アソシエーションのネゴシエーション中に、グループ内の ID が線形に検索されて、最初に一致するものが求められます。

コマンド・ラインからグループを定義する場合は、 258 ページの『IKE トンネル構成のためのコマンド・ライン・インターフェース』を参照してください。

IKE トンネル構成のシナリオ**:**

以下のシナリオでは、トンネルのセットアップ時にしばしば直面する状況のタイプを説明します。 シナリオとしては、事業所、ビジネス・パートナー、およびリモート・アクセスのケースがあります。

- 事業所のケースでは、お客様が 2 つのトラステッド・ネットワークを持っており、それらを相互に接続したい、すなわち 1 つの場所のエンジニアリング・グループと別の場所のエンジニアリング・グループを接続したいと考えています。 この例では、相互に接続されたゲートウェイが、同一のトンネルを使用しています。すべてのトラフィックはこれらのゲートウェイ間をパスしていきます。 トンネルのどちらのエンドのトラフィックもカプセル解除され、会社のイントラネット内を自由に通過します。

  IKE ネゴシエーションの最初のフェーズで、 IKE セキュリティー・アソシエーションが 2 つのゲートウェイ間で作成されます。 IP セキュリティー・トンネル内をパスするトラフィックは、2 つのサブネット間のトラフィックであり、サブネット ID がフェーズ 2 ネゴシエーションで使用されます。 そのトンネルに対するセキュリティー・ポリシーおよびトンネル・パラメーターを入力すると、トンネル番号が作成されます。 トンネルを開始するには、**ike** コマンドを使用します。

- ビジネス・パートナーのケースでは、ネットワークは信頼できず、ネットワーク管理者がセキュリティー・ゲートウェイの背後の少数のホストへのアクセスを制限したい場合があります。 この場合は、ホスト間のトンネルは、2 つの特定のホストで使用するために、IP セキュリティーで保護されたトラフィックを転送します。 フェーズ 2 トンネルのプロトコルは、AH または ESP です。 このホスト間トンネルは、ゲートウェイ間トンネル内で保護されます。
- リモート・アクセスのケースでは、トンネルはオンデマンドでセットアップされ、高水準のセキュリティーが適用されます。 IP アドレスは分かりやすくないので、完全修飾ドメイン名、または *user@* 完全修飾ドメイン名を推奨します。 オプションとして、キーをホスト ID へ関連付けるために KEYID を使用することができます。

デジタル証明書と鍵マネージャーの概念

デジタル証明書は、ID を公開鍵に結合します。これによって、暗号化転送の送信側または受信側を検証することができます。

IP セキュリティーは、デジタル証明書を使用して公開鍵暗号 (非対称暗号方式 とも呼ばれる) を使用します。これは、そのユーザーだけが知っている秘密鍵を使用してデータを暗号化し、さらに、指定された公開鍵と秘密鍵のペアで関連付けられている公開 (共有) 鍵を使用して、暗号化解除します。 鍵ペア は、ユーザーの暗号化方式に対する鍵として機能する、長いデータ文字列です。

公開鍵暗号においては、公開鍵は、ユーザーが通信しようとする相手に対して与えられます。 送信側は、すべてのセキュア通信に、それらの通信の割り当てられた鍵ペアに対応する秘密鍵を使用して、デジタル署名を行います。 受信側は、公開鍵を使用して、送信側の署名を検証します。 メッセージが、公開鍵を使用して正常に暗号化解除されると、受信側は、送信側が認証されたことを検証することができます。

公開鍵暗号方式では、信頼できるサード・パーティー (認証局 (*CA*) と呼ばれる) に頼って、信頼できるデジタル証明書を発行します。 受信側は、トラステッドとして認定できる発行組織または認証局を指定します。 証明書が特定の期間発行され、有効期限が切れると、置き換える必要があります。

AIX には、デジタル証明書を管理する Key Manager ツールがあります。以下のセクションでは、証明書自体に関係する概念について説明します。

デジタル証明書のフォーマット**:**

デジタル証明書には、証明書の所有者の ID および認証局に関する情報が含まれています。 以下の図は、デジタル証明書のイラストです。

図 *10.* デジタル証明書の内容

この図には、デジタル証明書の 4 つのエンティティーが示されています。それらは、上から、所有者の識別名、所有者の公開鍵、発行者 (CA) の識別名、および発行者の署名です。

次のリストは、さらにそのデジタル証明書の内容について説明しています。

所有者の識別名

所有者の一般名とディレクトリー・ツリー内のコンテキスト (位置) の組み合わせです。 例えば、次の図のような単純なディレクトリー・ツリーでは、Prasad が所有者の一般名で、コンテキストは country=US、organization=ABC、lower organization=SERV です。したがって、識別名は次のようになります。

```
/C=US/O=ABC/OU=SERV/CN=prasad.austin.ibm.com
```

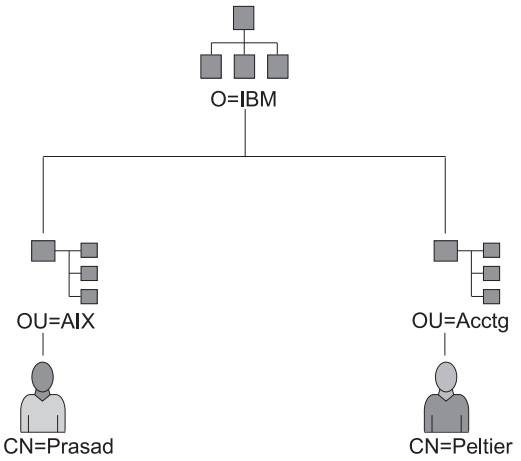


図 *11.* ディレクトリー・ツリーから派生した識別名の例

この図はディレクトリー・ツリーです。トップレベルには O=ABC が置かれ、そこから枝分かれして 2 番目のレベルではエンティティーが 2 つになります。 レベル 2 には OU=AIX と

OU=Acctg が含まれ、それぞれ別の枝に置かれています。それぞれの枝には、最終レベル上の単一エンティティーに向かってのびている枝があります。 最終レベルにはそれぞれ、CN=Prasad と CN=Peltier が含まれています。

所有者の公開鍵
: データの暗号化解除のために受信側で使用されます。

サブジェクト代替名
: IP アドレス、電子メール・アドレス、完全修飾ドメイン名などの ID。

発行日付
: デジタル証明書が発行された日付。

有効期限
: デジタル証明書が期限切れになる日付。

発行者の識別名
: 認証局の識別名。

発行者のデジタル署名
: 証明書の妥当性検査に使用されるデジタル署名。

デジタル証明書のセキュリティー上の考慮事項:

デジタル証明書だけでは、ID を証明することはできません。

デジタル証明書は、所有者のデジタル署名を検査するために必要な公開鍵を提供することにより、ユーザーがデジタル証明書の所有者の ID を確認できるようにするだけです。 ユーザーは、ユーザーの公開鍵を他のユーザーに送信しても問題ありません。これは、ユーザー・データの暗号が、鍵ペアのもう一方である、そのユーザーの秘密鍵がないと暗号化解除することができないためです。 したがって、所有者は、デジタル証明書の公開鍵に属する秘密鍵を保護する必要があります。 デジタル証明書の所有者のすべての通信は、秘密鍵が知られてしまうと、暗号解読されてしまいます。 秘密鍵なしでは、デジタル証明書を悪用することはできません。

認証局とトラスト階層:

デジタル証明書は、その証明書を発行した認証局 (CA) と同等に信頼できる唯一のものです。

この信頼の一部として、証明書が発行された際のポリシーが理解されている必要があります。 それぞれの組織、またはユーザーは、信頼して受け入れることができる認証局を決定する必要があります。

Key Manager ツールを使用すると、組織も自己署名証明書を作成することができます。この証明書は、テストの際、あるいはユーザーやマシンの数が少ない環境において有用です。

セキュリティー・サービスのユーザーとして、どんなデジタル証明書でも取得し、妥当性検査できるように、その公開鍵を知っている必要があります。 また、単にデジタル証明書を受信しただけでは、その認証性が保証されたことにはなりません。 その証明書の認証性を検査するには、ユーザーは、そのデジタル証明書を発行した認証局の公開鍵が必要です。まだ CA の公開鍵の保証済みコピーを持っていない場合には、その CA の公開鍵を取得するためのデジタル証明書がさらに必要な場合があります。

証明書取り消しリスト:

デジタル証明書は、その証明書の有効期間中を通して使用されるものと想定されます。 しかし、必要であれば、証明書はその有効期限の前に、無効にすることができます。

証明書を無効にする必要が生じるのは、例えば、社員が会社を退職した場合、または証明書の秘密鍵が漏えいした場合などです。 証明書を無効にするには、その状況を、適切な認証局 (CA) に通知する必要があります。 CA がその証明書を取り消すと、無効になった証明書のシリアル番号が証明書取り消しリスト (CRL) に追加されます。

CRL は定期的に発行される、署名付きデータ構造で、パブリック・リポジトリーで入手可能です。 CRL は、HTTP または LDAP サーバーから検索することができます。 各 CRL には、現在時刻のタイム・スタンプと nextUpdate のタイム・スタンプが含まれています。 このリスト内の失効したすべての証明書は、その証明書のシリアル番号で識別されます。

IKE トンネルを構成し、ユーザーの認証方式としてデジタル証明書を使用する場合には、ユーザーは、「RSA Signature with CRL Checking (CRL 検査付きの RSA 署名)」を選択して、その証明書が取り消されていないことを確認することができます。 CRL 検査が使用可能になっている場合には、このリストはネゴシエーション・プロセス中に検索され、検査されて、キー管理トンネルが作成されます。

> 注: この IP セキュリティーのフィーチャーを使用するには、システムが、SOCKS サーバー (HTTP サーバー用、バージョン 4)、LDAP サーバー、またはその両方を使用するように構成されている必要があります。 CRL を取得するためにどの SOCKS サーバーまたは LDAP サーバーを使用するかが分かっている場合は、それらを `/etc/isakmpd.conf` ファイルに追加することができます。

インターネット・アプリケーションでのデジタル証明書の使用:

公開鍵暗号システムを使用するインターネット・アプリケーションは、公開鍵を取得するために、デジタル証明書を使用する必要があります。

公開鍵暗号を使用するアプリケーションは多数あります。以下のアプリケーションもその 1 つです。

仮想プライベート・ネットワーク **(VPN)**
: VPN (仮想プライベート・ネットワーク)、(セキュア・トンネル とも呼ばれる) をファイアウォールなどのシステム間に設定することができ、これによって、保護されていない通信リンクを介してセキュア・ネットワーク間での保護接続を行うことができます。 これらのネットワーク向けのすべてのトラフィックは、関与するシステム間で暗号化されます。

  トンネリングで使用されるプロトコルは、IP セキュリティーおよび IKE 標準に準拠します。これによって、リモート・クライアント (例えば、在宅勤務者) とセキュア・ホストまたはネットワーク間での、セキュアされ、暗号化された接続が可能になります。

Secure Sockets Layer (SSL)
: SSL は、通信のプライバシーおよび保全性を提供するプロトコルです。 SSL は、Web サーバーと Web ブラウザー間の接続を保護するために Web サーバーによって使用され、 LDAP (Lightweight Directory Access Protocol) クライアントと LDAP サーバー間の接続を保護するために LDAP によって使用され、また、クライアント・システムとホスト・システム間の接続のために Host-on-Demand V.2 によって使用されます。 SSL は、キー交換、サーバー認証、さらにオプションとして、クライアント認証のためにデジタル証明書を使用します。

Secure Electronic Mail
: 電子メールを保護するための PEM または S/MIME などの規格を使用して、多くの電子メール・システムは、デジタル署名およびキー交換のために、デジタル証明書を使用して、メール・メッセージの暗号化と暗号化解除を行います。

デジタル証明書と認証要求:

デジタル証明書を要求するには、認証要求 を作成して、CA に送信する必要があります。

署名付きデジタル証明書には、所有者の識別名、所有者の公開鍵、CA の識別名、および CA の署名のフィールドが入っています。自己署名デジタル証明書には、その所有者の識別名、公開鍵、および署名が入っています。

認証要求には、要求側の識別名、公開鍵、および署名のフィールドがあります。CA は、デジタル証明書内の公開鍵を使用して、要求側の署名を検査して、以下の事項を確認します。

- 認証要求が、要求側と CA との間の転送中に変更されていないこと。
- 要求側が、認証要求に含まれている公開鍵に対応する秘密鍵を所有していること。

さらに CA には、要求側の ID を、あるレベルまで検査する責任があります。この検査に対する要件は、所有者の ID に関する、非常に簡単な検査から、完全検証まで、広範囲にわたります。

Key Manager ツール:

Key Manager ツールは、デジタル証明書を管理します。このツールは、拡張パックの **gskkm.rte** ファイルセットにあります。

デジタル証明書および署名サポートを設定するには、最低限、タスク 1、2、3、4、6、および 7 を行う必要があります。 その上で IKE トンネルを作成し、RSA 署名を認証方式として使用するトンネルにポリシーを関連付けます。

キー・データベースは、certmgr コマンドを使用して Key Manager ツールを開くことによってコマンド・ラインから作成および構成できます。

このセクションでは、Key Manager を使用して以下のタスクを行う方法を説明します。

キー・データベースの作成*:*

キー・データベースを使用することにより、VPN エンドポイントは有効なデジタル証明書を使用して接続を行うことができます。 キー・データベース (*.kdb) フォーマットは、IP セキュリティー VPN で使用されます。

Key Manager には、以下のタイプの CA デジタル証明書が用意されています。

- RSA Secure Server 認証局
- Thawte パーソナル・プレミアム認証局
- Thawte パーソナル・フリー・メール認証局
- Thawte パーソナル基本認証局
- Thawte パーソナル・サーバー認証局
- Thawte サーバー認証局
- Verisign クラス 1 パブリック・プライマリー認証局
- Verisign クラス 2 パブリック・プライマリー認証局
- Verisign クラス 3 パブリック・プライマリー認証局
- Verisign クラス 4 パブリック・プライマリー認証局

これらの署名デジタル証明書を使用することにより、クライアントはこれらの署名者からの有効なデジタル証明書を持つサーバーに接続することができます。キー・データベースは、作成されたままの形で、いずれかの署名者の有効なデジタル証明書を持つサーバーに接続できます。

このリストにない署名デジタル証明書を使用する場合は、その証明書を CA に対して要求し、自らのキー・データベースにそれを追加しなければなりません。 『CA ルート・デジタル証明書の追加』を参照してください。

certmgr コマンドを使用してキー・データベースを作成する場合は、次の手順を使用します。

1. 次のように入力して、Key Manager ツールを開始します。

   ```
   # certmgr
   ```

2. 「Key Database File (キー・データベース・ファイル)」リストから、「**New (新規)**」を選択します。
3. 「**Key database type (**キー・データベース・タイプ**)**」フィールドのデフォルト値「CMS key database file (CMS キー・データベース・ファイル)」を選択します。
4. 「**File Name (**ファイル名**)**」フィールドに以下のファイル名を入力します。

   ```
   ikekey.kdb
   ```

5. 「**Location (**ロケーション**)**」フィールドにデータベースのロケーションを入力します。

   ```
   /etc/security
   ```

   注**:** キー・データベースは、名前を **ikekey.kbd** にし、**/etc/security** ディレクトリーに置く必要があります。 そのようにしないと、IP セキュリティーは正常に機能しません。
6. 「**OK**」をクリックします。 「**Password Prompt (**パスワード・プロンプト**)**」画面が表示されます。
7. 「**Password (**パスワード**)**」フィールドにパスワードを入力し、「**Confirm Password (**パスワードの確認**)**」フィールドにもう一度入力します。
8. パスワードの有効日数を変更する場合は、「**Set expiration time? (**有効期限の設定**)**」フィールドに希望する日数を入力します。 このフィールドのデフォルト値は、60 日です。パスワードの有効期限を設けない場合は、「**Set expiration time? (**有効期限の設定**)**」フィールドをクリアします。
9. 暗号化されたパスワードを stash ファイルに保存するには、「**Stash the password to a file? (**パスワードをファイルへ **stash** する**)**」フィールドを選択し、「Yes」と入力します。

   注**:** IP セキュリティーにおいてデジタル証明書を使用できるようにするには、パスワードを stash ファイルに保管する必要があります。
10. 「**OK**」をクリックします。 確認画面が表示され、キー・データベースを作成したことが確認されます。
11. もう一度「**OK**」をクリックし、「IBM Key Management」画面に戻ります。他のタスクを実行するか、ツールを終了することができます。

CA ルート・デジタル証明書の追加**:**

CA にルート・デジタル証明書を要求し、受け取ったら、これを自分のデータベースに追加することができます。

ほとんどのルート・デジタル証明書は、以下のような *.arm の形式をとります。

```
cert.arm
```

データベースに CA ルート・デジタル証明書を追加するには、次の手順を使用します。

1. Key Manager ツールを開始していなければ、次のように入力してツールを開始します。

   ```
   # certmgr
   ```

2. メインスクリーンの「Key Database File (キー・データベース・ファイル)」リストから、「**Open (**開く**)**」を選択します。

3. CA ルート・デジタル証明書を追加するキー・データベース・ファイルを強調表示して、「**Open (**開く**)**」をクリックします。
4. パスワードを入力して「**OK**」をクリックします。パスワードが受け入れられると「IBM Key Management」画面に戻ります。 選択したキー・データベース・ファイルがタイトル・バーに表示され、このファイルが開いており作業する準備ができていることを示します。
5. 「**Personal/Signer Certificates (**個人/署名者証明書**)**」リストから、「**Signer Certificates (**署名者証明書**)**」を選択します。
6. 「**Add (**追加**)**」をクリックします。
7. 「**Data type (**データ・タイプ**)**」リストから、例えば次のようなデータ・タイプを選択します。

   ```
   Base64-encoded ASCII data
   ```
8. CA ルート・デジタル証明書の証明書ファイル名およびロケーションを入力するか、「**Browse (**ブラウズ**)**」をクリックして名前とロケーションを選択します。
9. 「**OK**」をクリックします。
10. CA ルート・デジタル証明書のラベル (例えば、「Test CA Root Certificate」) を入力して、「**OK**」をクリックします。 「**Key Management**」画面に戻ります。 「**Signer Certificates (**署名者証明書**)**」フィールドには、追加した CA ルート・デジタル証明書のラベルが示されます。 他のタスクを実行するか、ツールを終了できます。

トラスト設定の確立*:*

インストールされた CA 証明書は、デフォルトでトラステッド状態に設定されます。 トラステッド設定は、必要に応じて変更することができます。

トラスト設定を変更するには、次のようにします。

1. Key Manager ツールを開始していなければ、次のように入力してツールを開始します。

   ```
   # certmgr
   ```
2. メインスクリーンの「**Key Database File (**キー・データベース・ファイル**)**」リストから、「**Open (**開く**)**」を選択します。
3. デフォルトのデジタル証明書を変更するキー・データベース・ファイルを強調表示して「**Open (**開く**)**」をクリックします。
4. パスワードを入力して「**OK**」をクリックします。パスワードが受け入れられた後、「**IBM Key Management**」画面に戻ります。 選択したキー・データベース・ファイルがタイトル・バーに表示され、このファイルが開いていることを示します。
5. 「**Personal/Signer Certificates (**個人/署名者証明書**)**」リストから、「**Signer Certificates (**署名者証明書**)**」を選択します。
6. 変更する証明書を強調表示して「**View/Edit (**表示/編集**)**」をクリックするか、項目をダブルクリックします。 その証明書項目について、「**Key Information (**キー情報**)**」画面が表示されます。
7. この証明書をトラステッド・ルート証明書にするには、「**Set the certificate as a trusted root (**トラステッド・ルート証明書として設定**)** 」の隣のチェックボックスを選択して「**OK**」をクリックします。 証明書がトラステッドでなければ、チェックボックスをクリアして「**OK**」をクリックします。
8. 「**Signer Certificates (**署名者証明書**)**」画面の「**OK**」をクリックします。「**IBM Key Management**」画面に戻ります。 他のタスクを実行するか、ツールを終了することができます。

CA ルート・デジタル証明書の削除*:*

署名デジタル証明書リストのいずれかの CA のサポートが必要でなくなったときは、その CA のルート・デジタル証明書を削除する必要があります。

注**:** CA ルート・デジタル証明書を削除する前に、後で CA ルートを再作成する場合に備えてバックアップ・コピーを作成してください。

データベースから CA ルート・デジタル証明書を削除するには、次の手順を使用します。

1. Key Manager ツールを開始していなければ、次のように入力してツールを開始します。
   ```
   # certmgr
   ```
2. メインスクリーンの「**Key Database File (**キー・データベース・ファイル**)**」リストから、「**Open (**開く**)**」を選択します。
3. CA ルート・デジタル証明書を削除するキー・データベース・ファイルを強調表示して「**Open (**開く**)**」をクリックします。
4. パスワードを入力して「**OK**」をクリックします。パスワードが受け入れられると「**Key Management**」画面に戻ります。 選択したキー・データベース・ファイルがタイトル・バーに表示され、このファイルが開いており編集する準備ができていることが示されます。
5. 「**Personal/Signer Certificates (**個人**/**署名者証明書**)**」リストから、「**Signer Certificates (**署名者証明書**)**」を選択します。
6. 削除する証明書を強調表示して、「**Delete (**削除**)**」をクリックします。 「確認」画面が表示されます。
7. 「**Yes (**はい**)**」をクリックします。 「**IBM Key Management**」画面に戻ります。 CA ルート・デジタル証明書のラベルが「**Signer Certificates (**署名者証明書**)**」フィールドに表示されなくなります。他のタスクを実行するか、ツールを終了することができます。

デジタル証明書の要求*:*

デジタル証明書を獲得するには、Key Manager を使用して要求を生成し、これを CA に提出します。 生成する要求ファイルは、PKCS#10 フォーマットです。 CA は依頼者の ID を検査して、デジタル証明書を送信します。

デジタル証明書を要求するときは、次の手順に従います。

1. Key Manager ツールを開始していなければ、次のように入力してツールを開始します。
   ```
   # certmgr
   ```
2. メインスクリーンの「**Key Database File (**キー・データベース・ファイル**)**」リストから、「**Open (**開く**)**」を選択します。
3. 要求の生成元の **/etc/security/ikekey.kdb** キー・データベース・ファイルを強調表示して「**Open (**開く**)**」をクリックします。
4. パスワードを入力して「**OK**」をクリックします。パスワードが受け入れられた後、「**IBM Key Management**」画面に戻ります。 選択したキー・データベース・ファイルがタイトル・バーに表示され、このファイルが開いており編集する準備ができていることが示されます。
5. 「作成 **(Create)**」 > 「新規証明書要求 **(New Certificate Request)**」をクリックします。
6. 「**New (**新規**)**」をクリックします。
7. 続いて表示される画面で、自己署名デジタル証明書のキー・ラベルを入力します。例えば、次のようにします。

```
keytest
```

8. 「common name (共通名)」(デフォルトはホスト名) および「organization (組織)」を入力して、「country (国名)」を選択します。 その他のフィールドについては、デフォルト値を使用するか、または新しい値を選択します。
9. サブジェクト代替名を定義します。 サブジェクト代替に関連付けられているオプション・フィールドは、電子メール・アドレス、IP アドレス、および DNS 名です。 IP アドレスのトンネル・タイプについては、 IKE トンネルで構成したものと同じ IP アドレスを IP アドレス・フィールドに入力します。 *user@FQDN* のトンネル ID タイプの場合は、電子メール・アドレス・フィールドに入力します。 FQDN のトンネル ID タイプについては、DNS 名フィールドに完全修飾ドメイン名を入力します (例、*hostname.companyname*.com)。
10. 画面の下部に、ファイルの名前を入力します。例えば、次のようにします。

    ```
    certreq.arm
    ```

11. 「**OK**」をクリックします。 確認画面が表示され、新規デジタル証明書の要求を作成したことが確認されます。
12. 「**OK**」をクリックします。 「**IBM Key Management**」画面に戻ります。 「**Personal Certificate Requests (**個人証明書要求**)**」フィールドには、作成した新規デジタル証明書要求のキー・ラベル (PKCS#10) が表示されます。
13. ファイルを CA に送信して、新規デジタル証明書を要求します。他のタスクを実行するか、ツールを終了することができます。

新規デジタル証明書の追加 *(*受信*):*

CA から新規のデジタル証明書を受信した場合、これを要求の生成元であるキー・データベースに追加する必要があります。

新しいデジタル証明書を追加 (受信) するときは、次の手順に従います。

1. Key Manager ツールを開始していなければ、次のように入力してツールを開始します。

    ```
    # certmgr
    ```

2. メインスクリーンの「**Key Database File (**キー・データベース・ファイル**)**」リストから、「**Open (**開く**)**」を選択します。
3. 証明書要求の生成元のキー・データベース・ファイルを強調表示して「**Open (**開く**)**」をクリックします。
4. パスワードを入力して「**OK**」をクリックします。パスワードが受け入れられた後、「IBM Key Management」画面に戻ります。 選択したキー・データベース・ファイルがタイトル・バーに表示され、このファイルが開いており編集する準備ができていることが示されます。
5. 「**Personal/Signer Certificates (**個人/署名者証明書**)**」リストから、「**Personal Certificate (**署名者証明書**)**」を選択します。
6. 「**Receive (**受信**)**」をクリックして、新規に受信したデジタル証明書をデータベースに追加します。
7. 「**Data type (**データ・タイプ**)**」リストから、新規デジタル証明書のデータ・タイプを選択します。デフォルトは、**Base64-encoded ASCII data** です。
8. 新規デジタル証明書の証明書ファイル名およびロケーションを入力するか、または「**Browse (**ブラウズ**)**」をクリックして名前とロケーションを選択します。
9. 「**OK**」をクリックします。
10. 新規のデジタル証明書の記述ラベルを入力します。例えば、次のように行います。

    ```
    VPN Branch Certificate
    ```

11. 「**OK**」をクリックします。 「**IBM Key Management**」画面に戻ります。 「**Personal Certificates (**署名者証明書**)**」フィールドには、追加した新規デジタル証明書のラベルが表示されます。 他のタスクを実行するか、ツールを終了することができます。 証明書のロード中にエラーがあった場合は、証明書ファイルが -----BEGIN CERTIFICATE----- テキストで始まり、-----END CERTIFICATE----- テキストで終わっていることを確認してください。

    次に例を示します。

```
-----BEGIN CERTIFICATE-----
ajdkfjaldfwwwwwwwwwwadafdw
kajf;kdsajkflasasfkjafdaff
akdjf;ldasjkf;safdfdasfdas
kaj;fdljk98dafdas43adfadfa
-----END CERTIFICATE-----
```

    テキストがこれと異なる場合は、先頭と終了が適切になるように証明書ファイルを編集してください。

デジタル証明書の削除*:*

デジタル証明書の削除が必要になることがときどきあります。

注**:** デジタル証明書を削除する前に、後で再作成する場合に備えてバックアップ・コピーを作成してください。

データベースからデジタル証明書を削除するには、次の手順を使用します。

1. Key Manager ツールを開始していなければ、次のように入力してツールを開始します。

```
# certmgr
```

2. メインスクリーンの「**Key Database File (**キー・データベース・ファイル**)**」リストから、「**Open (**開く**)**」を選択します。
3. デジタル証明書を削除するキー・データベース・ファイルを強調表示して、「**Open (**開く**)**」をクリックします。
4. パスワードを入力して「**OK**」をクリックします。パスワードが受け入れられた後、「**IBM Key Management**」画面に戻ります。 選択したキー・データベース・ファイルがタイトル・バーに表示され、このファイルが開いており編集する準備ができていることが示されます。
5. 「**Personal/Signer Certificates (**個人**/**署名者証明書**)**」リストから、「**Personal Certificate (**署名者証明書**)**」を選択します。
6. 削除するデジタル証明書を強調表示して、「**Delete (**削除**)**」をクリックします。 「確認」画面が表示されます。
7. 「**Yes (**はい**)**」をクリックします。 「**IBM Key Management**」画面に戻ります。 削除したデジタル証明書のラベルが「**Personal Certificates (**個人証明書**)**」フィールドに表示されなくなります。 他のタスクを実行するか、ツールを終了することができます。

データベース・パスワードの変更*:*

データベース・パスワードの削除が必要になることがときどきあります。

キー・データベースを変更する場合は、次の手順を使用します。

1. Key Manager ツールを開始していなければ、次のように入力してツールを開始します。

```
# certmgr
```

2. メインスクリーンの「**Key Database File (**キー・データベース・ファイル**)**」リストから、「**Change Password (**パスワードの変更**)**」を選択します。
3. 「**Password (**パスワード**)**」フィールドに新規パスワードを入力し、「**Confirm Password (**パスワードの確認**)**」フィールドにもう一度入力します。
4. パスワードの有効日数を変更する場合は、「**Set expiration time? (**有効期限の設定**)**」フィールドに希望する日数を入力します。 このフィールドのデフォルト値は、60 日です。パスワードの有効期限を設けない場合は、「**Set expiration time? (**有効期限の設定**)**」フィールドをクリアします。
5. 暗号化されたパスワードを stash ファイルに保存するには、「**Stash the password to a file? (**パスワードをファイルへ **stash** する**)**」フィールドを選択し、「Yes」と入力します。

   注**:** IP セキュリティーにおいてデジタル証明書を使用できるようにするには、パスワードを stash ファイルに保管する必要があります。
6. 「**OK**」をクリックします。 ステータス・バーのメッセージにより、要求が正常に行われたことが示されます。
7. もう一度「**OK**」をクリックし、「**IBM Key Management**」画面に戻ります。他のタスクを実行するか、ツールを終了することができます。

デジタル証明書を使用する ***IKE*** トンネルの作成**:**

デジタル証明書を使用する IKE トンネルを作成するには、IKE トンネル変換ポリシー・ファイルに、認証モードとして RSA 署名を指定する必要があります。

RSA 署名を指定した XML ポリシー・ファイルの例を以下に示します。

```
<!-- define the policy for IKE tunnel -->
<IKEProtection
 IKE ProtectionName="ike_3des_sha">
 <IKETTransform
  IKE AuthenticationMethod="RSA_signatures"
  IKE Encryption="3DES-CBC"
  IKE Hash="SHA"
  IKE DHGroup="1"/>
</IKEProtection>
```

IP セキュリティーでは、以下の IKE トンネルのホスト ID タイプがサポートされています。

- IP アドレス
- 完全修飾ドメイン名 (FQDN)
- *user@FQDN*
- X.500 識別名
- キー ID

IKE トンネルが RSA 署名モードを使用する場合、通常、X.500 識別名が IKE トンネル定義で使用されます。例えば、トンネルのローカル・ホストおよびリモート・ホストが **/C=US/O=ABC/OU=SERV/CN=localname.austin.ibm.com** および **/C=US/O=ABC/OU=SERV/CN=remotename.austin.ibm.com** として識別される場合、XML ファイルの IKE トンネル定義は次のサンプル・コンテンツのように示されます。

```
<IKETunnel>
 IKE TunnelName="Key_Tunnel"
 IKE ProtectionRef="ike_3des_sha">
<IKELocalIdentity>
 <ASN1_DN Value="/C=US/O=ABC/OU=SERV/CN=localname.austin.ibm.com">
 </ASN1_DN>
```

```
</IKELocalIdentity>
<IKERemoteIdentity>
 <ASN1_DN Value="/C=US/O=ABC/OU=SERV/CN=remotename.austin.ibm.com">
 </ASN1_DN>
</IKERemoteIdentity>
</IKETunnel>
```

必要な認証を認証局 (CA) から取得するには、Key Manager ツールを使用して認証要求を生成します。例えば、認証で **/C=US/O=ABC/OU=SERV/CN=name.austin.ibm.com** をサブジェクト識別名として使用する場合、デジタル証明書要求を作成する際に以下の値を Key Manager ツールに入力する必要があります。

共通名
: *name.austin.ibm.com*

組織 ABC

組織単位
: SERV

国名 US

入力される X.500 識別名は、通常、システム管理者または LDAP 管理者によって設定された名前です。組織単位の値はオプションです。

IP セキュリティーは、デジタル証明書のサブジェクト代替名として、他の識別タイプの入力もサポートします。例えば、代替ホスト ID として IP アドレス 10.10.10.1 を使用する場合、デジタル証明書要求に以下の値を入力する必要があります。

共通名
: *name.austin.ibm.com*

組織 ABC

組織単位
: SERV

国名 US

サブジェクト代替 **IP** アドレス・フィールド
: 10.10.10.1

この情報を用いてデジタル証明書要求を作成すると、 CA はこれを使用して個人デジタル証明書を作成します。

個人デジタル証明書を要求する際、CA は以下の情報を必要とします。

- X.509 証明書を要求している。
- 署名のフォーマットが RSA 暗号化による MD5 である。
- サブジェクト代替名を指定しているかどうか。 代替名タイプは、次のリストで提供されます。
  - IP アドレス
  - 完全修飾ドメイン名 (FQDN)
  - *user@FQDN*

  以下のサブジェクト代替名情報は、証明書要求ファイルに含まれます。
- 計画されたキー使用 (デジタル署名ビットを選択する必要があります)。

- Key Manager デジタル証明書要求ファイル (PKCS#10 フォーマット)。

証明書要求を作成するための、Key Manager ツールの使用方法を説明した特別の手順については、 269 ページの『デジタル証明書の要求』を参照してください。

IKE トンネルをアクティブにする前に、 CA から受信した個人デジタル証明書を Key Manager データベース `ikekey.kdb` に追加する必要があります。 詳しくは、 270 ページの『新規デジタル証明書の追加 (受信)』 を参照してください。

IP セキュリティーでは、以下のタイプの個人デジタル証明書がサポートされています。

サブジェクト **DN**
: サブジェクト識別名は、以下のフォーマットおよび順序とする必要があります。

  `/C=US/O=ABC/OU=SERV/CN=`*`name`*`.austin.ibm.com`

  Key Manager ツールでは、**OU** 値は 1 つに限られます。

サブジェクト **DN** およびサブジェクト代替名を **IP** アドレスとする
: 以下のように、サブジェクト識別名およびサブジェクト代替名を IP アドレスとして指定することができます。

  `/C=US/O=ABC/OU=SERV/CN=name.austin.ibm.com` および `10.10.10.1`

サブジェクト **DN** およびサブジェクト代替名を **FQDN** とする
: 以下のように、サブジェクト識別名およびサブジェクト代替名を完全修飾ドメイン名として指定することができます。

  `/C=US/O=ABC/OU=SERV/CN=name.austin.ibm.com` および `bell.austin.ibm.com`

サブジェクト **DN** およびサブジェクト代替名を *user@FQDN* とする
: 以下のように、サブジェクト識別名およびサブジェクト代替名をユーザー・アドレス (*user_ID@fully_qualified_domain_name*) として指定することができます。

  `/C=US/O=ABC/OU=SERV/CN=name.austin.ibm.com` および `name@austin.ibm.com`

サブジェクト **DN** および複数のサブジェクト代替名
: 以下のように、サブジェクト識別名を複数のサブジェクト代替名に関連付けることができます。

  `/C=US/O=ABC/OU=SERV/CN=name.austin.ibm.com` および `bell.austin.ibm.com`、`10.10.10.1`、および `user@name.austin.ibm.com`

ネットワーク・アドレス変換

IP セキュリティーでは、アドレスがネットワーク・アドレス変換 (NAT) の対象とされるデバイスを使用することができます。

NAT は、インターネット接続共有のためのファイアウォール・テクノロジーの一部として広く使用されており、さらにルーターおよびエッジ・デバイスの標準機構ともなっています。 IP セキュリティー・プロトコルは、リモート IP アドレスをベースとした、リモート・エンドポイントおよびそれらのポリシーの識別に依存します。 ルーターやファイアウォールなどの中間デバイスが専用アドレスを共用アドレスに変換するとき、IP セキュリティーにおける必要な認証プロセスが失敗することがあります。その理由は、IP パケットのアドレスが、認証ダイジェストが計算された後に変更されているからです。 新規の IP セキュリティー NAT サポートにより、 ネットワーク・アドレス変換を行うノードの背後で構成されているデバイスは、IP セキュリティー・トンネルを確立することができます。 IP セキュリティー・コードは、リモート・アドレスが変換されている場合、それを検出することができます。 NAT サポート付きの、新規の IP セキュリティー・インプリメンテーションを使用すれば、VPN クライアントは、使用可能にされている

NAT とのインターネット接続を介して、自宅または出先からオフィスへ接続することが可能となります。

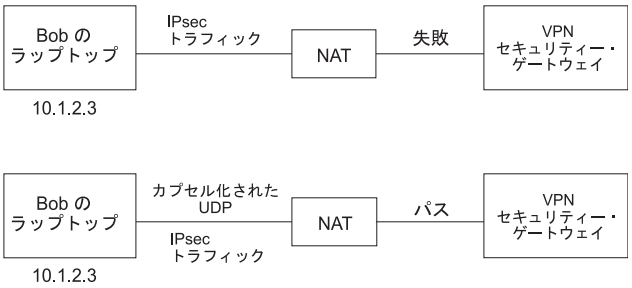


図 *12. NAT* 使用可能 *IP* セキュリティー

このダイアグラムは、NAT 使用可能の IP セキュリティー・インプリメンテーション (UDP カプセル化トラフィック) と NAT 使用不可のインプリメンテーションとの差を示します。

NAT と連動する IP セキュリティーの構成:

IP セキュリティーにおいて NAT を使用するには、**/etc/isakmpd.conf** ファイルに *ENABLE_IPSEC_NAT_TRAVERSAL* 変数を設定する必要があります。 この変数が設定されると、ポート 4500 上でトラフィックを送受信するためにフィルター・ルールが追加されます。

次の例は、*ENABLE_IPSEC_NAT_TRAVERSAL* 変数が設定されたときのフィルター・ルールを示しています。

```
Dynamic rule 2:
Rule action         : permit
Source Address      : 0.0.0.0 (any)
Source Mask         : 0.0.0.0 (any)
Destination Address : 0.0.0.0 (any)
Destination Mask    : 0.0.0.0 (any)
Source Routing      : no
Protocol            : udp
Source Port         : 0 (any)
Destination Port    : 4500
Scope               : local
Direction           : inbound
Fragment control    : all packets
Tunnel ID number    : 0

Dynamic rule 3:
Rule action         : permit
Source Address      : 0.0.0.0 (any)
Source Mask         : 0.0.0.0 (any)
Destination Address : 0.0.0.0 (any)
Destination Mask    : 0.0.0.0 (any)
Source Routing      : no
Protocol            : udp
Source Port         : 4500
Destination Port    : 0 (any)
Scope               : local
Direction           : outbound
Fragment control    : all packets
Tunnel ID number    : 0
```

ENABLE_IPSEC_NAT_TRAVERSAL 変数を設定すると、フィルター・テーブルにさらにいくつかのフィルター・ルールが追加されます。 特殊な IPSEC NAT メッセージは UDP カプセル化を使用し、このトラ

フィックが流れるようにするためにフィルター・ルールを追加する必要があります。 さらに、フェーズ 1 においては署名モードが必要です。 証明書の中で IP アドレスが ID として使用される場合は、専用 IP アドレスを含める必要があります。

IP セキュリティーでは、元の IP アドレスと NAT アドレスのマッピングを保守するために、NAT キープアライブ・メッセージを送信する必要もあります。 インターバルは、**/etc/isakmpd.conf** ファイル内の *NAT_KEEPALIVE_INTERVAL* 変数で指定します。 この変数は、NAT キープアライブ・パケットを送信する頻度を秒数で指定します。 *NAT_KEEPALIVE_INTERVAL* の値を指定しなかった場合は、デフォルト値の 20 秒が使用されます。

NAT 交換の使用に関する制限**:**

NAT デバイス背後のエンドポイントは、ESP プロトコルを使用して自らのトラフィックを保護する必要があります。

ESP は IP セキュリティー用に選択された優位ヘッダーであり、ほとんどのカスタマー・アプリケーションで使用できます。 ESP にはユーザー・データのハッシュは含まれますが、IP ヘッダーのハッシュは含まれません。 AH ヘッダーの整合性チェックは、IP ソース・アドレスと宛先アドレスをキー付きメッセージ整合性チェックに組み込みます。 アドレス・フィールドに変更を加える、NAT またはリバース NAT デバイスは、メッセージ整合性チェックを無効にします。 したがって、トンネルのフェーズ 2 ポリシーにおいて AH プロトコルのみを定義し、しかもフェーズ 1 交換で NAT が検出された場合は、`NO_PROPOSAL_CHOSEN` という通知ペイロードが送信されます。

さらに、NAT を使用する接続においては、元の IP アドレスがパケット内でカプセル化されるように、トンネル・モードを選択する必要があります。 トランスポート・モードと NAT 付きアドレスは互換性がありません。 NAT が検出されており、かつフェーズ 2 においてトランスポート・モードのみが提案されている場合は、`NO_PROPOSAL_CHOSEN` という通知ペイロードが送信されます。

トンネル・モード競合の回避**:**

リモート・ピアがゲートウェイでオーバーラップするエントリーについて交渉することがあります。このオーバーラップによりトンネル・モード競合が発生します。

以下の図はトンネル・モード競合を示しています。

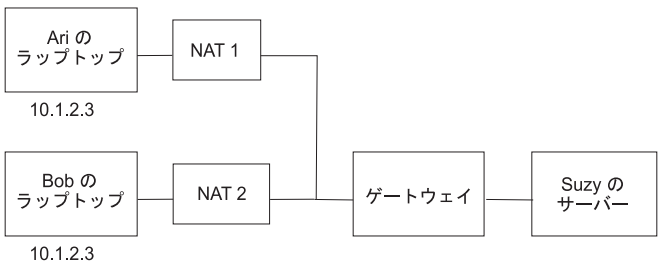


図 *13.* トンネル・モード競合

ゲートウェイには、10.1.2.3 IP アドレス用の可能なセキュリティー・アソシエーション (SA) が 2 つあります。 これらの重複したリモート・アドレスにより、サーバーから来たパケットの送信先について混乱が生じます。 Suzy のサーバーと Ari のラップトップの間にトンネルが構成された場合、 IP アドレスが使用され、Suzy は同じ IP アドレスを使用して Bob の間にトンネルを構成することはできません。 トンネル・モード競合を回避するには、同じ IP アドレスを使用してトンネルを定義してはなりません。 リモー

ト・アドレスはリモート・ユーザーの制御下にないため、完全修飾ドメイン名などのリモート・ホストや、完全修飾ドメイン名のユーザーを識別するためには、別の ID タイプを使用する必要があります。

マニュアル・トンネルの構成

デバイスが自動キー入力方式をサポートしていない場合には、IP セキュリティーのマニュアル・トンネルを構成することができます。

マニュアル・トンネルおよびフィルター**:**

トンネルの設定のプロセスは、1 つの終端のトンネルを定義し、もう 1 つの終端にその定義をインポートし、そのトンネルとフィルター・ルールを両端で活動化します。 これで、トンネルは使用可能になります。

マニュアル・トンネルをセットアップするために、別途にフィルター・ルールを構成する必要はありません。 2 つのホスト間のすべてのトラフィックがそのトンネルを通過する場合は、必要なフィルター・ルールが自動的に生成されます。

トンネルに関する情報は、明示的に提供されない場合、両端で一致している必要があります。 例えば、送信元に対して指定された暗号化および認証アルゴリズムは、宛先の値が指定されていないと、その宛先に対して使用されます。

最初のホスト上のマニュアル・トンネルの作成*:*

トンネルは、 SMIT `ips4_basic` 高速パス (IP バージョン 4 の場合)、または SMIT `ips6_basic` 高速パス (IP バージョン 6 の場合) を使用して構成できます。あるいは、以下の手順を使用して手動でトンネルを作成できます。

以下は、マニュアル・トンネルを作成するために使用される **gentun** コマンドの例です。

```
gentun -v 4 -t manual -s 5.5.5.19 -d 5.5.5.8 ¥
   -a HMAC_MD5 -e DES_CBC_8 -N 23567
```

lstun -v 4 コマンドを使用すると、この例で作成したマニュアル・トンネルの特性をリストできます。 出力は次の例のようになります。

```
Tunnel ID           : 1
IP Version          : IP Version 4
Source              : 5.5.5.19
Destination         : 5.5.5.8
Policy              : auth/encr
Tunnel Mode         : Tunnel
Send AH Algo        : HMAC_MD5
Send ESP Algo       : DES_CBC_8
Receive AH Algo     : HMAC_MD5
Receive ESP Algo    : DES_CBC_8
Source AH SPI       : 300
Source ESP SPI      : 300
Dest AH SPI         : 23576
Dest ESP SPI        : 23576
Tunnel Life Time    : 480
Status              : Inactive
Target              : -
Target Mask         : -
Replay              : No
New Header          : Yes
Snd ENC-MAC Algo    : -
Rcv ENC-MAC Algo    : -
```

トンネルを活動化するには、次のコードを入力します。

```
mktun -v 4 -t1
```

トンネルに関連付けられているフィルター・ルールが自動的に生成されます。

フィルター・ルールを表示させるには、**lsfilt -v 4** コマンドを使用します。出力は次の例のようになります。

```
Rule 4:
Rule action           : permit
Source Address        : 5.5.5.19
Source Mask           : 255.255.255.255
Destination Address   : 5.5.5.8
Destination Mask      : 255.255.255.255
Source Routing        : yes
Protocol              : all
Source Port           : any 0
Destination Port      : any 0
Scope                 : both
Direction             : outbound
Logging control       : no
Fragment control      : all packets
Tunnel ID number      : 1
Interface             : all
Auto-Generated        : yes

Rule 5:
Rule action           : permit
Source Address        : 5.5.5.8
Source Mask           : 255.255.255.255
Destination Address   : 5.5.5.19
Destination Mask      : 255.255.255.255
Source Routing        : yes
Protocol              : all
Source Port           : any 0
Destination Port      : any 0
Scope                 : both
Direction             : inbound
Logging control       : no
Fragment control      : all packets
Tunnel ID number      : 1
Interface             : all
Auto-Generated        : yes
```

デフォルト・フィルター・ルールの場合も含め、フィルター・ルールを活動化するときは、**mktun -v 4 -t 1** コマンドを使用します。

他端をセットアップするには (これが、このオペレーティング・システムを使用するもう 1 つのマシンである場合)、トンネルの定義をホスト A にエクスポートし、次にホスト B にインポートすることができます。

次のコマンドでは、トンネル定義を **ipsec_tun_manu.exp** という名前のファイルにエクスポートし、**-f** フラグで指示されたディレクトリーの **ipsec_fltr_rule.exp** ファイルに、関連するフィルター・ルールをエクスポートします。

```
exptun -v 4 -t 1 -f /tmp
```

2 番目のホスト上のマニュアル・トンネルの作成*:*

トンネルのマッチングする終端を作成するには、エクスポート・ファイルをコピーし、リモート・マシンにインポートします。

次のコマンドを使用して。トンネルのマッチングする終端を作成します。

```
imptun -v 4 -t 1 -f /tmp
```

オプションは以下のとおりです。

1　　インポートされるトンネル

/tmp　　インポート・ファイルが常駐するディレクトリー

トンネル番号は、システムによって生成されます。 **gentun** コマンドの出力から、または **lstun** コマンドを使用してトンネルをリストし、そのトンネル番号を入手して、インポートする正しいトンネル番号を判別することができます。 インポート・ファイルにトンネルが 1 つだけある場合、またはすべてのトンネルをインポートする場合、**-t** オプションは不要です。

リモート・マシンで、このオペレーティング・システムが稼働していない場合、エクスポート・ファイルは、トンネルの他の終端のアルゴリズム、キー、およびセキュリティー・パラメーター索引 (SPI) の値を設定するためのリファレンスとして使用できます。

ファイアウォール製品からのエクスポート・ファイルをインポートして、トンネルを作成できます。 これを行うには、ファイルをインポートするときに、次のようにして **-n** オプションを使用します。

```
imptun -v 4 -f /tmp -n
```

フィルターの除去**:**

フィルターを完全に除去して IP セキュリティーを停止するには、**rmdev** コマンドを使用します。

デフォルト・フィルター・ルールは、**mkfilt -d** コマンドを使用してフィルター処理がオフになってもアクティブなままとなります。 このコマンドでは、すべてのフィルター・ルールを使用停止または除去し、デフォルト・ルールの保護を残しながら、新規ルールをロードすることができます。 デフォルトのフィルター・ルールは *DENY* です。 **mkfilt -d** コマンドを使用してフィルター処理を非活動化すると、**lsfilt** コマンドからのレポートに、フィルター処理がオフにされているが、内外いずれでも、パケットが許可されないことが示されます。 完全に IP セキュリティーを停止したい場合は、**rmdev** コマンドを使用してください。

IP セキュリティー・フィルターの構成

フィルターは、ほとんど自動的に生成されたフィルター・ルールを使用して、簡単にセットアップすることができます。また、IP パケットの属性を基にした、非常に特殊なフィルター機能を定義して、カスタマイズすることができます。

フィルター・テーブルの各行は、ルール と呼ばれます。 ルールの集合が、マシンへの入出力が受け入れられるパケットと、それらのパケットが送信される方法を決定します。 着信パケットでのフィルター・ルールとの突き合わせは、送信元アドレスおよび SPI 値を、フィルター・テーブル内にリストされた値と比較して行われます。 したがって、このペアは固有でなければなりません。 フィルター・ルールでは、送信元と宛先のアドレスおよびマスク、プロトコル、ポート番号、方向、フラグメント・コントロール、送信元経路指定、トンネル、インターフェース・タイプなど、通信に関係する多くの局面を制御できます。

フィルター・ルールには、次のようなタイプがあります。

- 静的フィルター・ルールは、トラフィックの一般フィルター操作やマニュアル・トンネルとの関連付けに使用されるフィルター・テーブルで作成されます。 このルールは、追加、削除、変更、および移動が可能です。特定のルールを識別するために、オプションの記述テキスト・フィールドを追加することもできます。

- 自動生成フィルター・ルールとユーザー指定フィルター・ルール (自動生成 フィルター・ルールとも呼ばれる) は、IKE トンネルの使用のために作成された、特定のルールのセットです。 静的フィルター・ルールおよび動的フィルター・ルールはどちらも、データ管理トンネル情報およびデータ管理トンネル・ネゴシエーションを基に作成されます。
- 事前定義フィルター・ルールは、変更、移動、または削除ができない汎用フィルター・ルールです。これには、`all traffic` ルール、`ah` ルール、および `esp` ルールなどがあります。これらのルールは、すべてトラフィックに関係しています。

genfilt コマンドの指示フラグ (**-w**) は、指定されたルールを、入力パケットの処理中に使用するのか、出力パケットの処理中に使用するかを指定するために使用されます。 このフラグに **both** の値が使用された場合は、このルールは入力処理中と出力処理中の両方に使用されることを指定します。 AIX IPsec では、フィルター処理がオンになっていると、少なくとも 1 つのルールがすべてのネットワーク・パケットの運命を決定します (着信か発信かに関係なく)。 着信パケット (あるいは発信パケット) の処理中にのみルールを使用したい場合は、**genfilt** コマンドの **-w** スイッチを使用して、そうするように選択することができます。 例えば、パケットがホスト A からホスト B に送信される場合、発信 IP パケットは *A* の送信元アドレスと、*B* の宛先アドレスを持ちます。ホスト A では、このパケットは IPsec フィルターにより、アウトバウンド処理中に処理され、ホスト B ではインバウンド処理中に処理されます。ホスト A とホスト B の間にゲートウェイ G があると想定します。ゲートウェイ G では、この同じパケット (すべての不変のフィールドが同じ値を持っている) は、インバウンド処理で一度、そしてアウトバウンド処理で一度の計 2 回処理されます (**ipforwarding** オプションが設定されている場合)。 ゲートウェイ G を通ってホスト A からホスト B に送られるパケットの場合、次の許可ルールが必要です。

- ホスト A – **src addr** は `A`、**dest addr** は `B`、方向はアウトバウンドに設定
- ホスト B – **src addr** は `A`、**dest addr** は `B`、方向はインバウンドに設定

しかし、ゲートウェイ G では、2 つのルールが必要になります。

1. **src addr** は `A`、**dest addr** は `B`、方向はアウトバウンドに設定
2. **src addr** は `A`、**dest addr** は `B`、方向はインバウンドに設定

上記のルールは、次によって置き換えることができます: **src addr** は `A`、**dest addr** は `B`、方向は両方に設定。 したがって、方向の **both** の値は、**ipforwarding** オプションが `no` に設定されているゲートウェイで一般に使用されます。上記の構成は、ゲートウェイ G を通ってホスト A からホスト B に送られるパケット専用です。パケットを逆方向で送信したい (ゲートウェイ G を通ってホスト B からホスト A へ) 場合は、そのための別のルールが必要です。

注**:** 方向 **both** は、関連したルールが着信と発信パケットの両方に使用されることを暗黙指定しています。しかし、発信元と宛先アドレスが逆になる場合は、そのルールが適用されることを意味しません。 例えば、*A* が送信元アドレス、*B* が宛先アドレス、方向が **both** に設定されたルールをサーバー A が持っている場合、*B* が送信元アドレス、*A* が宛先アドレスの着信パケット *A* はこのルールと一致しません。一般に **both** オプションは、パケットを転送するゲートウェイで使用されます。

これらのフィルター・ルールに関連するものに、サブネット・マスク (グループ ID が 1 つのフィルター・ルールに関連付けられている) および host-firewall-host 構成オプションがあります。 以下のセクションでは、種々のフィルター・ルールのタイプと、それらのタイプに関連するフィーチャーについて解説します。

AIX の IP フィルター:

IPFilter はネットワーク・アドレス変換 (NAT) またはファイアウォール・サービスの提供に使用できるソフトウェア・パッケージです。

IPFilter バージョン 4.1.13 オープン・ソース・ソフトウェアは AIX に移植されたもので、 IP Filter Web サイト (http://coombs.anu.edu.au/~avalon/) に提示されたライセンスと整合性が取れています。 IPFilter ソフトウェアは AIX 拡張パックに組み込まれて出荷されています。installp パッケージ、ipfl には、マニュアル・ページおよびライセンスが含まれています。

AIX オペレーティング・システムでは、IPFilter 製品はカーネル・エクステンションとして /usr/lib/drivers/ipf にロードされます。 **ipf**、**ipfs**、**ipfstat**、**ipmon**、および **ipnat** バイナリーもこのパッケージに付けて出荷されます。

パッケージのインストール後に、以下のコマンドを実行してカーネル・エクステンションをロードします。

```
/usr/lib/methods/cfg_ipf -l
```

カーネル・エクステンションをアンロードする場合は、以下のコマンドを実行します。

```
/usr/lib/methods/cfg_ipf -u
```

パケットの転送が必要な場合は、ipforwarding (ネットワーク・オプション) を使用可能にすることを覚えておいてください。 IPFilter とマニュアル・ページおよび FAQ の詳細情報については、 IPFilter Web サイト (http://coombs.anu.edu.au/~avalon/) で確認してください。

静的フィルター・ルール**:**

各静的フィルター・ルールには、スペースで区切られたフィールドがあります。

次のリストは、静的フィルター・ルールの各フィールドの名前を示しています (括弧内は、ルール 1 の場合の各フィールドの例です)。

- Rule_number (1)
- Action (permit)
- Source_addr (0.0.0.0)
- Source_mask (0.0.0.0)
- Dest_addr (0.0.0.0)
- Dest_mask (0.0.0.0)
- Source_routing (no)
- Protocol (udp)
- Src_prt_operator (eq)
- Src_prt_value (4001)
- Dst_prt_operator (eq)
- Dst_prt_value (4001)
- Scope (both)
- Direction (both)
- Logging (no)
- Fragment (all packets)
- Tunnel (0)
- Interface (all).

静的フィルター・ルールの例

```
1 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no udp eq 4001 eq 4001 both both no all
   packets 0 all

2 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no ah any 0 any 0 both both no all packets
   0 all

3 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no esp any 0 any 0 both both no all packets
   0 all

4 permit 10.0.0.1 255.255.255.255 10.0.0.2 255.255.255.255 no all any 0 any 0 both
   outbound no all packets 1 all outbound traffic

5 permit 10.0.0.2 255.255.255.255 10.0.0.1 255.255.255.255 no all any 0 any 0 both
   inbound no all packets 1 all


6 permit 10.0.0.1 255.255.255.255 10.0.0.3 255.255.255.255 no tcp lt 1024 eq 514 local
   outbound yes all packets 2 all


7 permit 10.0.0.3 255.255.255.255 10.0.0.1 255.255.255.255 no tcp/ack eq 514 lt 1024
   local inbound yes all packets 2 all


8 permit 10.0.0.1 255.255.255.255 10.0.0.3 255.255.255.255 no tcp/ack lt 1024 lt 1024
   local outbound yes all packets 2 all


9 permit 10.0.0.3 255.255.255.255 10.0.0.1 255.255.255.255 no tcp lt 1024 lt 1024 local
   inbound yes all packets 2 all


10 permit 10.0.0.1 255.255.255.255 10.0.0.4 255.255.255.255 no icmp any 0 any 0 local
   outbound yes all packets 3 all


11 permit 10.0.0.4 255.255.255.255 10.0.0.1 255.255.255.255 no icmp any 0 any 0 local
   inbound yes all packets 3 all


12 permit 10.0.0.1 255.255.255.255 10.0.0.5 255.255.255.255 no tcp gt 1023 eq 21 local
   outbound yes all packets 4 all


13 permit 10.0.0.5 255.255.255.255 10.0.0.1 255.255.255.255 no tcp/ack eq 21 gt 1023 local
   inbound yes all packets 4 all


14 permit 10.0.0.5 255.255.255.255 10.0.0.1 255.255.255.255 no tcp eq 20 gt 1023 local
   inbound yes all packets 4 all


15 permit 10.0.0.1 255.255.255.255 10.0.0.5 255.255.255.255 no tcp/ack gt 1023 eq 20 local
   outbound yes all packets 4 all


16 permit 10.0.0.1 255.255.255.255 10.0.0.5 255.255.255.255 no tcp gt 1023 gt 1023 local
   outbound yes all packets 4 all


17 permit 10.0.0.5 255.255.255.255 10.0.0.1 255.255.255.255 no tcp/ack gt 1023 gt 1023 local
   inbound yes all packets 4 all


18 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0 no all any 0 any 0 both both yes all
   packets
```

上の例の各ルールを説明します。

ルール **1**
Session Key デーモン用のルールです。 このルールは、IP バージョン 4 のフィルター・テーブルだけに表示されます。 このルールは、セッション・キーのリフレッシュ用のパケットを制御するために、ポート番号 4001 を使用します。 ルール 1 は、ポート番号が特定の目的のためにどのように使用されるかを示す例です。

注**:** ロギングを目的とする場合以外、このフィルター・ルールは変更しないでください。

ルール **2** およびルール **3**
これらのルールによって、認証ヘッダー (AH) およびカプセル化セキュリティー・ペイロード (ESP) ヘッダーの処理が可能となります。

注**:** ロギングを目的とする場合以外、ルール 2 および 3 は変更しないでください。

ルール **4** およびルール **5**
アドレス 10.0.0.1 およびアドレス 10.0.0.2 の間のトラフィックをトンネル 1 を通してフィルターに掛ける、自動生成ルールの集合。ルール 4 はアウトバウンド・トラフィック用で、ルール 5 はインバウンド・トラフィック用です。

注**:** ルール 4 には、アウトバウンド・トラフィック のユーザー定義記述があります。

ルール **6** からルール **9**
アドレス 10.0.0.1 およびアドレス 10.0.0.3 間のアウトバウンド rsh、 rcp、rdump、rrestore、および rdist サービスをトンネル 2 を通してフィルターに掛ける、ユーザー定義ルールの集合。この例では、ロギングは Yes に設定され、管理者がトラフィックのタイプをモニターできます。

ルール **10** およびルール **11**
アドレス 10.0.0.1 およびアドレス 10.0.0.4 間のすべてのインバウンドおよびアウトバウンド icmp サービスのすべてのタイプをトンネル 3 を通してフィルターに掛ける、ユーザー定義ルールの集合。

ルール **12** からルール **17**
10.0.0.1 および 10.0.0.5 間のアウトバウンド・ファイル転送プロトコル (FTP) サービスをトンネル 4 を通してフィルターに掛ける、ユーザー定義フィルター・ルール。

ルール **18**
自動生成ルールは、常にテーブルの最後に配置されます。 この例では、他のフィルター・ルールと突き合わないすべてのパケットを許可します。 他のフィルター・ルールと突き合わないすべてのトラフィックを拒否するように設定することもできます。

ルールは、それぞれ別々に (**lsfilt** を使用して) 表示して、各フィールドとその値をリストすることができます。 次に例を示します。

```
Rule 1:
Rule action          : permit
Source Address       : 0.0.0.0
Source Mask          : 0.0.0.0
Destination Address  : 0.0.0.0
Destination Mask     : 0.0.0.0
Source Routing       : yes
Protocol             : udp
Source Port          : eq  4001
Destination Port     : eq  4001
Scope                : both
Direction            : both
Logging control      : no
```

```
Fragment control     : all packets
Tunnel ID number     : 0
Interface            : all
Auto-Generated       : yes
```

以下のリストには、フィルター・ルールで指定可能なすべてのパラメーターが含まれています。

-v IP バージョン: 4 または 6

-a アクション:

- **d** 拒否
- **p** 許可

-s 送信元アドレス。 IP アドレスまたはホスト名のどちらでも可。

-m 送信元サブネット・マスク

-d 宛先アドレス。 IP アドレスまたはホスト名のどちらでも可。

-M 宛先サブネット・マスク

-g 送信元経路指定制御: `y` または `n`

-c プロトコル。 値は、`udp`、`icmp`、`tcp`、`tcp/ack`、`ospf`、`pip`、`esp`、`ah`、および `all` のいずれかです。

-o 送信元ポートの操作または ICMP タイプの操作

-p 送信元ポートの値または ICMP タイプの値

-0 宛先ポートの操作、または ICMP コードの操作

-P 宛先ポートの値、または ICMP コード値

-r 経路指定 :

- **r** 転送パケット。
- **l** ローカルの受信/発信パケット。
- **b** 両方。

-l ログ制御

- **y** ログに組み込みます。
- **n** ログに組み込みません。

-f フラグメント化。

- **y** フラグメント・ヘッダー、フラグメント、および非フラグメントに適用します。
- **o** フラグメントおよびフラグメント・ヘッダーのみに適用します。
- **n** 非フラグメントのみに適用します。
- **h** 非フラグメントおよびフラグメント・ヘッダーのみに適用します。

-t トンネル ID

-i `tr0` または `en0` などのインターフェース

詳しくは、『**genfilt** コマンド』および『**chfilt** コマンド』の説明を参照してください。

自動生成およびユーザー指定のフィルター・ルール:

特定のルールが、IP セキュリティー・フィルターおよびトンネル・コードを使用するために自動生成されます。

自動生成されるルールには、以下のルール・セットが含まれています。

- IKE の IP バージョン 4 キーをリフレッシュするセッション・キー・デーモンのためのルール
- AH および ESP パケットの処理のためのルール

フィルター・ルールは、トンネルを定義する際にも自動的に生成されます。マニュアル・トンネルの場合、自動生成ルールが、送信元アドレス、宛先アドレス、マスク値、およびトンネル ID を指定します。 これらのアドレス間のすべてのトラフィックは、トンネルを経由して流れます。

IKE トンネルに関しては、自動生成されたフィルター・ルールが、IKE ネゴシエーション中にプロトコルおよびポート番号を決定します。 IKE フィルター・ルールは、別のテーブルに保持されます。このテーブルは、静的フィルター・ルールの後、自動生成ルールの前に検索されます。 IKE フィルター・ルールは、静的フィルター・テーブル内のデフォルトの位置に挿入されます。ただしこれらのルールは、ユーザーが削除できます。

自動生成ルールは、トンネル経由のすべてのトラフィックを許可します。ユーザー定義ルールは、特定のタイプのトラフィックに制約を設定することができます。 IP セキュリティーは、パケットに適用されるルールの中で最初に検出されるルールを使用するため、これらのユーザー定義ルールは自動生成ルールの前に置いてください。以下は、ICMP 操作に基づいてトラフィックをフィルターに掛ける、ユーザー定義フィルター・ルールの例です。

```
1 permit 10.0.0.1 255.255.255.255 10.0.0.4 255.255.255.255 no icmp any 8 any 0
   local outbound no all packets 3 all
2 permit 10.0.0.4 255.255.255.255 10.0.0.1 255.255.255.255 no icmp any 0 any 0 local
   inbound no all packets 3 all
3 permit 10.0.0.4 255.255.255.255 10.0.0.1 255.255.255.255 no icmp any 8 any 0 local
   inbound no all packets 3 all
4 permit 10.0.0.1 255.255.255.255 10.0.0.4 255.255.255.255 no icmp any 0 any 0 local
   outbound no all packets 3 all
```

単一トンネルの構成を単純化するために、フィルター・ルールは、トンネルの定義時に自動生成されます。この機能は、**gentun** に **-g** フラグを指定することによって抑制できます。サンプル・フィルター・ファイルは、種々の TCP/IP サービスに対してフィルター・ルールを生成する **genfilt** コマンドを使って、**/usr/samples/ipsec/filter.sample** 内から見つけることができます。

事前定義フィルター・ルール:

いくつかの事前定義フィルター・ルールは、特定のイベントが発生したときに自動生成されます。

`ipsec_v4` または `ipsec_v6` デバイスがロードされると、事前定義ルールがフィルター・テーブルに挿入され、起動されます。 デフォルトでは、この事前定義ルールはすべてのパケットを許可しますが、これはユーザーによって構成可能なものであり、すべてのパケットを拒否するようにも設定できます。

注: リモートで構成する場合には、構成が完了する前に拒否ルールを使用可能にすることがないようにしてください。そのようにすると、セッションがマシンからロックアウトされてしまいます。 このような事態は、デフォルトのアクションを許可に設定するか、 IP セキュリティーを活動化する前にリモート・マシンにトンネルを構成しておくことによって防げます。

IP バージョン 4 および IP バージョン 6 の両方のフィルター・テーブルに事前定義ルールがあります。どちらのルールも、すべてを拒否するように個別に変更することが可能です。これによって、追加のフィル

ター・ルールで特別にトラフィックが定義されていない限り、トラフィックが通らないようにすることができます。 事前定義ルールを変更するための他の唯一のオプションは、**-l** オプションを指定した **chfilt** であり、これによってそのルールにマッチするパケットをログに記録することが可能になります。

IKE トンネルをサポートするために、動的フィルター・ルールが IP バージョン 4 フィルター・テーブルに入っています。 この位置は、動的フィルター・ルールがフィルター・テーブルに挿入される位置です。この位置は、ユーザーがフィルター・テーブルのその位置を上下に移動することで、制御することができます。 トンネル・マネージャー・デーモンと **isakmpd** デーモンが初期化されて、 IKE トンネルがネゴシエーション可能になると、ルールが自動的に動的フィルター・テーブルに作成され、 AH および ESP パケットと同様に、IKE メッセージを処理できるようになります。

サブネット・マスク**:**

サブネット・マスクは、フィルター・ルールと関連付けられた ID の集合をグループ化するために使用されます。 このマスク値は、フィルター・ルールの ID と AND 演算され、そのパケットで指定された ID と比較されます。

例えば、以下に示すように、10 進の IP アドレスと完全に一致しなければならないと指定された送信元 IP アドレス `10.10.10.4`、およびサブネット・マスク `255.255.255.255` のフィルター・ルールです。

	2 進	**10** 進
送信元 IP アドレス	`1010.1010.1010.0100`	`10.10.10.4`
サブネット・マスク	`11111111.11111111.11111111.11111111`	`255.255.255.255`

`10.10.10.x` サブネットは `11111111.11111111.11111111.0` または `255.255.255.0` として指定されます。着信アドレスには、そのアドレスに適用されたサブネット・マスクがあり、その組み合わせがフィルター・ルール内の ID と比較されます。 例えば、`10.10.10.100` のアドレスは、サブネット・マスクを適用すると `10.10.10.0` になり、これはフィルター・ルールと一致します。

`255.255.255.240` のサブネット・マスクでは、そのアドレスの最後の 4 ビットについては、どんな値でも許されます。

ホスト**-**ファイアウォール**-**ホスト構成**:**

トンネル用の、ホスト-ファイアウォール-ホスト構成オプションを使用すると、ユーザーのホストおよびファイアウォールの間にトンネルを 1 つ作成でき、ユーザーのホストと、ファイアウォールの後ろのホストの間の通信を正しく行うために必要なフィルター・ルールを、自動的に生成できます。

自動生成されたフィルター・ルールは、指定されたトンネルを経由する 2 つの非ファイアウォール・ホスト間のすべてのルールを許可します。 デフォルト・ルール (ユーザー・データグラム・プロトコル (UDP)、認証ヘッダー (AH)、およびカプセル化セキュリティー・ペイロード (ESP) ヘッダー用) では、事前にホストとファイアウォール間の通信を処理できるようにしておく必要があります。 ファイアウォールは、セットアップを完了するために正しく構成されていなければなりません。 ユーザーが作成したトンネルからのエクスポート・ファイルを使用して、ファイアウォールが必要とする SPI 値およびキーを入力する必要があります。

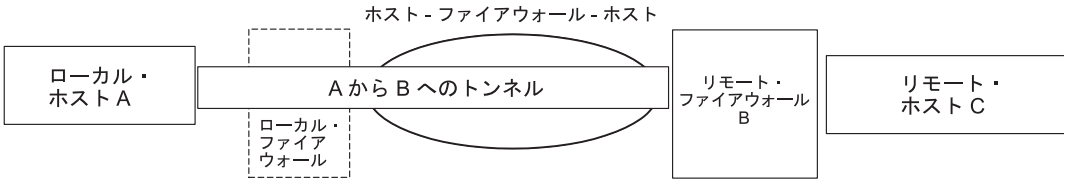


図 *14.* ホスト-ファイアウォール-ホスト

図は、ホスト-ファイアウォール-ホスト構成を示しています。ホスト A は、ローカル・ファイアウォールを通ってインターネットへ抜けるトンネルを持っています。 これは、さらにリモート・ファイアウォール B に達し、その先のリモート・ホスト C へ達します。

ロギング機能

ホストが互いに通信しあうとき、転送されたパケットはシステム・ログ・デーモンの **syslogd** に記録できます。 また、IP セキュリティーに関するその他の重要なメッセージも表示されます。

システム管理者は、トラフィックの分析とデバッグ支援のためにこのログ情報をモニターできます。 以下はロギング機能をセットアップするためのステップです。

1. **/etc/syslog.conf** ファイルを編集して、次のエントリーを追加します。

   ```
   local4.debug var/adm/ipsec.log
   ```

   local4 機能を使用して、トラフィック・イベントおよび IP セキュリティー・イベントを記録します。標準のオペレーティング・システム優先順位が適用されます。 debug の優先順位は、 IP セキュリティー・トンネルとフィルターを通過したトラフィックが安定し、正しく移動したことを示すまで、維持し続けてください。

   注**:** フィルター・イベントのロギングにより、 IP セキュリティー・ホストでの有効なアクティビティーが作成されますが、大量のストレージを消費する恐れがあります。
2. **/etc/syslog.conf file** を保存します。
3. ログ・ファイルに指定したディレクトリーに移動して、名前が同じ空ファイルを作成します。 上記の場合は、**/var/adm** ディレクトリーに移動し、次のコマンドを発行します。

   ```
   touch ipsec.log
   ```
4. **syslogd** サブシステムに **refresh** コマンドを発行します。

   ```
   refresh -s syslogd
   ```
5. IKE トンネルを使用している場合は、**/etc/isakmpd.conf** ファイルが必要な **isakmpd** ロギング・レベルを指定していることを確認します。 (IKE ロギングについての詳細は、 292 ページの『インターネット・プロトコルのセキュリティーの問題診断』を参照してください。)
6. ホスト用のフィルター・ルールの作成時に、特定のルールに一致するパケットを記録する場合は、そのルールの *-l* パラメーターを **Y** (Yes) に設定します (**genfilt** または **chfilt** コマンドを使用)。
7. パケット・ロギングをオンにして、コマンドで **ipsec_logd** デーモンを始動します。

   ```
   mkfilt -g start
   ```

   次のコマンドを実行すれば、パケット・ロギングを中止できます。

   ```
   mkfilt -g stop
   ```

以下のサンプル・ログ・ファイルには、トラフィック・エントリーとその他の IP セキュリティー・ログ・エントリーが含まれています。

```
1. Aug 27 08:08:40 host1 : Filter logging daemon ipsec_logd (level 2.20)
   initialized at 08:08:40 on 08/27/97A
2. Aug 27 08:08:46 host1 : mkfilt: Status of packet logging set to Start
   at 08:08:46 on 08/27/97
3. Aug 27 08:08:47 host1 : mktun: Manual tunnel 2 for IPv4, 9.3.97.244, 9.3.97.130
   activated.
4. Aug 27 08:08:47 host1 : mkfilt: #:1 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
   udp eq  4001 eq  4001  both both l=n f=y t=0 e= a=
5. Aug 27 08:08:47 host1 : mkfilt: #:2 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
   ah any 0 any 0  both both l=n f=y t=0 e= a=
6. Aug 27 08:08:47 host1 : mkfilt: #:3 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
   esp any 0 any 0  both both l=n f=y t=0 e= a=
7. Aug 27 08:08:47 host1 : mkfilt: #:4 permit 10.0.0.1 255.255.255.255 10.0.0.2
   255.255.255.255 icmp any 0 any 0  local outbound l=y f=y t=1 e= a=
8. Aug 27 08:08:47 host1 : mkfilt: #:4 permit 10.0.0.2 255.255.255.255 10.0.0.1
   255.255.255.255 icmp any 0 any 0  local inbound l=y f=y t=1 e= a=
9. Aug 27 08:08:47 host1 : mkfilt: #:6 permit 0.0.0.0 0.0.0.0 0.0.0.0 0.0.0.0
    all any 0 any 0  both both l=y f=y t=0 e= a=
10. Aug 27 08:08:47 host1 : mkfilt: Filter support (level 1.00) initialized at
    08:08:47 on 08/27/97
11. Aug 27 08:08:48 host1 : #:6 R:p  o:10.0.0.1 s:10.0.0.1 d:10.0.0.20 p:udp
    sp:3327 dp:53 r:l a:n f:n T:0 e:n l:67
12. Aug 27 08:08:48 host1 : #:6 R:p  i:10.0.0.1 s:10.0.0.20 d:10.0.0.1 p:udp
    sp:53 dp:3327 r:l a:n f:n T:0 e:n l:133
13. Aug 27 08:08:48 host1 : #:6 R:p  i:10.0.0.1 s:10.0.0.15 d:10.0.0.1 p:tcp
    sp:4649 dp:23 r:l a:n f:n T:0 e:n l:43
14. Aug 27 08:08:48 host1 : #:6 R:p  o:10.0.0.1 s:10.0.0.1 d:10.0.0.15 p:tcp
    sp:23 dp:4649 r:l a:n f:n T:0 e:n l:41
15. Aug 27 08:08:48 host1 : #:6 R:p  i:10.0.0.1 s:10.0.0.15 d:10.0.0.1 p:tcp
    sp:4649 dp:23 r:l a:n f:n T:0 e:n l:40
16. Aug 27 08:08:51 host1 : #:4 R:p  o:10.0.0.1 s:10.0.0.1 d:10.0.0.2 p:icmp
    t:8 c:0 r:l a:n f:n T:1 e:n l:84
17. Aug 27 08:08:51 host1 : #:5 R:p  i:10.0.0.1 s:10.0.0.2 d:10.0.0.1 p:icmp
    t:0 c:0 r:l a:n f:n T:1 e:n l:84
18. Aug 27 08:08:52 host1 : #:4 R:p  o:10.0.0.1 s:10.0.0.1 d:10.0.0.2 p:icmp
    t:8 c:0 r:l a:n f:n T:1 e:n l:84
19. Aug 27 08:08:52 host1 : #:5 R:p  i:10.0.0.1 s:10.0.0.2 d:10.0.0.1 p:icmp
    t:0 c:0 r:l a:n f:n T:1 e:n l:84
20. Aug 27 08:32:27 host1 : Filter logging daemon terminating at 08:32:27 on
    08/27/97l
```

以下の項では、ログ・エントリーについて説明します。

1 フィルター・ロギング・デーモンが始動しました。

2 フィルター・パケット・ロギングが **mkfilt -g start** コマンドでオンに設定されました。

3 トンネルの始動。トンネル ID、送信元アドレス、宛先アドレス、およびタイム・スタンプを示しています。

4-9 フィルターが起動しています。 ロード済みのすべてのフィルター・ルールを示しています。

10 フィルターの起動を示すメッセージ。

11-12 これらのエントリーは、ホスト用の DNS ルックアップを示しています。

13-15 これらのエントリーは、一部の Telnet 接続を示しています (その他のエントリーは、スペースの都合でこの例から除去しました)。

16-19 これらのエントリーは、2 つの PING を示しています。

20 フィルター・ロギング・デーモンが停止します。

以下の例では、フェーズ 1 トンネルとフェーズ 2 トンネルについて折衝する 2 つのホストを、開始する側のホストの視点から示しています。 (**isakmpd** ロギング・レベルが **isakmp_events** として指定されています。)

```
1. Dec  6 14:34:42 host1 Tunnel Manager: 0: TM is processing a
    Connection_request_msg
 2. Dec  6 14:34:42 host1 Tunnel Manager: 1: Creating new P1 tunnel object (tid)
 3. Dec  6 14:34:42 host1 isakmpd: 192.168.100.103 >>> 192.168.100.104 ( SA PROPOSAL
    TRANSFORM  )
 4. Dec  6 14:34:42 host1 isakmpd: ::ffff:192.168.100.103 <<< 192.168.100.104 ( SA
    PROPOSAL TRANSFORM  )
 5. Dec  6 14:34:42 host1 isakmpd: Phase I SA Negotiated
 6. Dec  6 14:34:42 host1 isakmpd: 192.168.100.103 >>> 192.168.100.104 ( KE NONCE  )
 7. Dec  6 14:34:42 host1 isakmpd: ::ffff:192.168.100.103 <<< 192.168.100.104 ( KE
    NONCE  )
 8. Dec  6 14:34:42 host1 isakmpd: Encrypting the following msg to send: ( ID HASH
     )
 9. Dec  6 14:34:42 host1 isakmpd: 192.168.100.103 >>> 192.168.100.104 ( Encrypted
    Payloads )
10. Dec  6 14:34:42 host1 isakmpd: ::ffff:192.168.100.103 <<< 192.168.100.104 (
    Encrypted Payloads )
11. Dec  6 14:34:42 host1 Tunnel Manager: 1: TM is processing a P1_sa_created_msg
    (tid)
12. Dec  6 14:34:42 host1 Tunnel Manager: 1:   Received good P1 SA, updating P1
    tunnel (tid)
13. Dec  6 14:34:42 host1 Tunnel Manager: 0: Checking to see if any P2 tunnels need
    to start
14. Dec  6 14:34:42 host1 isakmpd: Decrypted the following received msg: ( ID HASH
     )
15. Dec  6 14:34:42 host1 isakmpd:  Phase I Done !!!
16. Dec  6 14:34:42 host1 isakmpd: Phase I negotiation authenticated
17. Dec  6 14:34:44 host1 Tunnel Manager: 0: TM is processing a
    Connection_request_msg
18. Dec  6 14:34:44 host1 Tunnel Manager: 0: Received a connection object for an
    active P1 tunnel
19. Dec  6 14:34:44 host1 Tunnel Manager: 1: Created blank P2 tunnel (tid)
20. Dec  6 14:34:44 host1 Tunnel Manager: 0: Checking to see if any P2 tunnels need
    to start
21. Dec  6 14:34:44 host1 Tunnel Manager: 1: Starting negotiations for P2 (P2 tid)
22. Dec  6 14:34:45 host1 isakmpd: Encrypting the following msg to send: ( HASH SA
    PROPOSAL TRANSFORM NONCE ID ID  )
23. Dec  6 14:34:45 host1 isakmpd: 192.168.100.103 >>> 192.168.100.104 ( Encrypted
    Payloads )
24. Dec  6 14:34:45 host1 isakmpd: ::ffff:192.168.100.103 <<< 192.168.100.104 (
    Encrypted Payloads )
25. Dec  6 14:34:45 host1 isakmpd: Decrypted the following received msg: ( HASH SA
    PROPOSAL TRANSFORM NONCE ID ID  )
26. Dec  6 14:34:45 host1 isakmpd: Encrypting the following msg to send: ( HASH  )
27. Dec  6 14:34:45 host1 isakmpd: 192.168.100.103 >>> 192.168.100.104 ( Encrypted
    Payloads )
28. Dec  6 14:34:45 host1 isakmpd: Phase II SA Negotiated
29. Dec  6 14:34:45 host1 isakmpd: PhaseII negotiation complete.
30. Dec  6 14:34:45 host1 Tunnel Manager: 0: TM is processing a P2_sa_created_msg
31. Dec  6 14:34:45 host1 Tunnel Manager: 1: received p2_sa_created for an existing
    tunnel as initiator (tid)
32. Dec  6 14:34:45 host1 Tunnel Manager: 1: Filter::AddFilterRules: Created filter
    rules for tunnel
33. Dec  6 14:34:45 host1 Tunnel Manager: 0: TM is processing a List_tunnels_msg
```

以下の項では、ログ・エントリーについて説明します。

1-2 **ike cmd=activate phase=1** コマンドが接続を開始します。

3-10 **isakmpd** デーモンがフェーズ 1 トンネルについてネゴシエーションします。

11-12 Tunnel Manager が有効なフェーズ 1 セキュリティー・アソシエーションを応答側から受信します。

13 Tunnel Manager が **ike cmd=activate** に、これ以上の作業を示すフェーズ 2 の値があるかどうかを検査します。 そのような値はありません。

14-16 **isakmpd** デーモンがフェーズ 1 ネゴシエーションを終了します。

17-21 **ike cmd=activate phase=2** コマンドがフェーズ 2 トンネルを開始します。

22-29 **isakmpd** デーモンがフェーズ 2 トンネルについてネゴシエーションします。

30-31 Tunnel Manager が有効なフェーズ 2 セキュリティー・アソシエーションを応答側から受信します。

32 Tunnel Manager が動的フィルター・ルールを書き込みます。

33 **ike cmd=list** コマンドが IKE トンネルを表示します。

フィールド・エントリーのラベル:

ログ・エントリーのフィールドは、DASD スペース容量を節約するために、次のように省略されます。

フィールド	意味
#	このパケットを記録させるルール番号。
R	ルール・タイプ **p** 許可 **d** 拒否
i/o	フィルター・サポート・コードによってパケットを代行受信されたとき、それが移動していた方向。 そのパケットと関連したアダプターの IP アドレスを識別します。 • インバウンド (i) パケットの場合、これはパケットが着いたアダプターです。 • アウトバウンド (o) パケットの場合、これは IP 層がパケットの送信を操作しなければならないと判断したアダプターです。
s	パケットの送信側の IP アドレスを指定します (IP ヘッダーから抽出される)。
d	パケットの予定受信側の IP アドレスを指定します (IP ヘッダーから抽出される)。
p	パケットのデータ部にメッセージを作成するために使用された高水準プロトコルを指定します。 使用される可能性があるのは、次のような数値または名前です。 `udp`、`icmp`、`tcp`、`tcp/ack`、`ospf`、`pip`、`esp`、`ah`、または `all`。
sp/t	パケットの送信側と関連したプロトコル・ポート番号を指定します (TCP/UDP ヘッダーから抽出される)。 プロトコルが ICMP または OSPF である場合、このフィールドは **t** と置き換えられ、IP タイプを示します。
dp/c	パケットの予定受信側と関連したプロトコル・ポート番号を指定します (TCP/UDP ヘッダーから抽出される)。 プロトコルが ICMP である場合、このフィールドは **c** と置き換えられ、IP コードを示します。
-	情報が使用不可であることを指定します。
r	パケットにローカル関係があるかどうかを示します。 **f** 転送パケット **l** ローカル・パケット **o** 発信 **b** 両方
l	特定のパケットの長さをバイト単位で指定します。
f	そのパケットがフラグメントであるかどうかを識別します。
T	トンネル ID を示します。
i	パケットがオンになったインターフェースを指定します。

インターネット鍵交換のロギング:

isakmpd デーモンを使用して、インターネット鍵交換イベントの SYSLOG 機能へのロギングを使用可能にすることができます。

isakmpd デーモンの場合、**ike cmd=log** コマンドを使用してロギングを使用可能にします。 **log_level** パラメーターを使用して、/etc/isakmpd.conf 構成ファイルにロギング・レベルを設定することができます。 ログに記録する必要がある情報量に応じて、*none*、*errors*、*isakmp_events*、または *information* にレベルを設定することができます。

例えば、プロトコル情報と実装情報を記録する必要がある場合は、次のパラメーターを設定します。

```
log_level=INFORMATION
```

isakmpd デーモンは、プロポーザルの送信またはプロポーザルの評価という 2 つのプロセスの 1 つを開始します。 プロポーザルが受け入れられると、セキュリティー・アソシエーションが作成され、トンネルがセットアップされます。 プロポーザルが受け入れられないか、またはネゴシエーションが完了する前に接続がタイムアウトになると、**isakmpd** デーモンはエラーを表示します。 **tmd** の結果の SYSLOG 機能のエントリーは、ネゴシエーションが成功したかどうかを示します。 無効な証明書による失敗は、SYSLOG 機能に記録されます。 ネゴシエーションが失敗した原因を正確に判別するためには、**/etc/syslog.conf** に指定されているログ・ファイルのデータを調べてください。

SYSLOG 機能は、ログの各行に、日時、マシン、およびプログラムを示すプレフィックスが追加されます。 次の例では、googly がマシン名として使用され、isakmpd がプログラム名として使用されています。

```
Nov 20 09:53:50 googly isakmpd: ISAKMP_MSG_HEADER
Nov 20 09:53:50 googly isakmpd:  Icookie : 0xef06a77488f25315, Rcookie :0x0000000000000000
Nov 20 09:53:51 googly isakmpd:  Next Payload : 1(SA), Maj Ver : 1, Min Ver : 0
Nov 20 09:53:51 googly isakmpd:  Xchg Type : 2 (ID protected), Flag= 0, Encr : No,COMMIT : No
Nov 20 09:53:51 googly isakmpd:  Msg ID  : 0x00000000
```

より明確にするには、**grep** コマンドを使用してログ内の関連する行 (すべての **isakmpd** ロギングなど) を抽出し、**cut** コマンドを使用して各行からプレフィックスを除去します。

/etc/isakmpd.conf ファイル:

/etc/isakmpd.conf ファイルに、**isakmpd** デーモンのオプションを構成することができます。

以下のオプションは、/etc/isakmpd.conf ファイルで使用可能です。

ログ構成
: ログに記録する必要がある情報の量を決定します。 続いてレベルを設定します。 IKE デーモンは、このオプションを使用して、ロギングのレベルを指定します。

  構文: `none | error | isakmp_events | information`

  レベルの意味は次のとおりです。

  none ロギングなし。 これがデフォルトです。

  error プロトコル・エラーまたはアプリケーション・プログラミング・インターフェース (API) エラーをログに記録します。

  isakmp_events
  : IKE プロトコル・イベントまたはエラーをログに記録します。 問題のデバッグにはこのレベルを使用します。

  information
  : プロトコル情報および実装情報をログに記録します。

認識されない **IP** アドレス・ネゴシエーション
: このオプションは YES または NO に設定できます。このオプションを YES に設定する場合は、ローカル IKE データベースにフェーズ 1 トンネルの両方のエンドポイントの IP アドレスが含ま

れていなければなりません。 ホストが着信メインモード・トンネルを受け入れるためには、このオプションを YES に設定する必要があります。 IP アドレスは、1 次 ID であってもよく、また、他の ID タイプに関連付けられているオプションの IP アドレスであってもかまいません。

着信メインモード接続を受け入れるためには、このオプションを NO に設定します。このオプションを NO に設定すると、IKE データベースがフェーズ 1 エンドポイントの IP アドレスを指定していない場合にもホストが接続を受け入れることになります。 しかし、ホストが接続を受け入れるためには、証明書ベースの認証を使用する必要があります。 このようにすれば、IP アドレスを動的に割り当てられるホストが、相手側マシンへのメインモード・トンネルを開始できます。

このパラメーターを指定しない場合、デフォルトは NO です。

構文: MAIN_MODE_REQUIRES_IP= YES | NO

SOCKS4 サーバーの構成

SOCKS4_PORTNUM オプションはオプションです。 指定しない場合は、デフォルトの SOCKS サーバー・ポート値 1080 が使用されます。 このポート値は、SOCKS サーバーが HTTP サーバーと通信するときに使用されます。

構文: *mnemonic = value*
ここで、*mneumonic* および *value* は次の値にすることができます。

SOCKS4_SERVER= サーバー名を指定

SOCKS4_PORTNUM= SOCKS サーバー・ポート番号を指定

SOCKS4_USERID= ユーザー ID

LDAP サーバーの構成

構文: *mnemonic = value*
ここで、*mnemonic* および *value* は次の値にすることができます。

LDAP_SERVER= LDAP サーバー名を指定

LDAP_VERSION= LDAP サーバーのバージョン (2 または 3 が可能)

LDAP_SERVERPORT= LDAP サーバーのポート番号

LDAP_SEARCHTIME= クライアント検索のタイムアウト値

CRL フェッチ順序

このオプションは、HTTP サーバーまたは LDAP サーバーの両方が構成されている場合に、どちらを先に照会するかを定義します。 CRL_FETCH_ORDER オプションはオプションです。 デフォルトのフェッチ順序は HTTP が先、次に LDAP です。ただし、HTTP サーバーと LDAP サーバーの両方が構成されているかどうかによります。

構文: CRL_FETCH_ORDER= *protocol#, protocol#*
ここで、*protocol#* は HTTP または LDAP です。

IKEv1 および **IKEv2** ポート仕様

この文字列で **isakmpd** デーモン (IKEv1) および **ikev2d** デーモン (IKEv2) により使用されるポートを指定します。 **iked** デーモン (IKE メッセージ・ブローカー・デーモン) はこのエントリーを調べ、それぞれのポート上で **isakmpd** デーモンおよび **ikev2d** デーモンを起動します。

構文: v1=port-natport,v2=port-natport

インターネット・プロトコルのセキュリティーの問題診断

以下は、問題を検出したときにその解決を助けるヒントです。

IPSec を最初に構成するときにロギングをセットアップします。 ログは、フィルターとトンネルで行われていることを判断するのに非常に役立ちます。 (ログ情報の詳細については、 287 ページの『ロギング機能』を参照してください。)

どの IP セキュリティー・デーモンが実行中であるかを判別するには、次のコマンドを入力します。

```
ps -ef
```

tmd、**iked**、**isakmpd**、**ikev2d**、および **cpsd** デーモンが IP セキュリティーに関連しています。

注**:** IKEv1 および IKEv2 の両方が構成済みの場合は、**iked** デーモンが実行します。それ以外の場合は、**iskmpd** デーモンまたは **ikev2d** デーモンが実行します。 この構成は **/etc/isakmpd.conf** ファイルにあります。

マニュアル・トンネル・エラーのトラブルシューティング**:**

以下では、起こりうるいくつかのトンネル・エラーと、その解決策について説明します。

エラー	発生する可能性のある問題と解決策
mktun コマンドを出すと、次のエラーが発生します。 「insert_tun_man4(): write failed : The requested resource is busy.」	問題: 起動要求したトンネルが既に起動しているか、または SPI 値が一致していません。 修正方法: **rmtun** コマンドを出して活動停止させてから、 mktun コマンドを出して起動させます。 失敗したトンネル用の SPI 値が他のアクティブ・トンネルと一致しているかどうかを検査します。 各トンネルには、固有の SPI 値が必要です。

mktun コマンドを出すと、次のエラーが発生します。
 	
「Device ipsec_v4 is in Defined status.」
 	
「Tunnel activation for IP Version 4 not performed.」
	問題: IP セキュリティー・デバイスを使用することができません。
	
	修正方法: 次のコマンドを入力します。
	
	`mkdev -l ipsec -t 4`
	
	IP バージョン 6 のトンネル起動に対して同じエラーが生じる場合は、**-t** オプションを 6 に変更する必要があります。 そのデバイスを使用できるようにしてください。 IP セキュリティー・デバイスの状況を検査するには、次のコマンドを出します。
	
	`lsdev -Cc ipsec`

gentun コマンドを出すと、次のエラーが発生します。 「Invalid Source IP address」	問題: 送信元アドレスに有効な IP アドレスを入力していません。 修正方法: IP バージョン 4 のトンネルの場合、ローカル・マシンに使用可能な IP バージョン 4 のアドレスを入力したかどうかを検査してください。 トンネルの生成時には送信元にホスト名は使用できません。宛先にのみホスト名を使用できます。 IP バージョン 6 のトンネルの場合、使用可能な IP バージョン 6 のアドレスを入力したかどうかを検査してください。 `netstat -in` と入力したのに IP バージョン 6 のアドレスがない場合は、 (MAC アドレスを使用して) リンク・ローカル自動生成アドレス用に **/usr/sbin/autoconf6** (インターフェース) を実行するか、または **ifconfig** コマンドを使用して手動でアドレスを割り当てます。
gentun コマンドを出すと、次のエラーが発生します。 「Invalid Source IP address」	問題: 送信元アドレスに有効な IP アドレスを入力していません。 修正方法: IP バージョン 4 のトンネルの場合、ローカル・マシンに使用可能な IP バージョン 4 のアドレスを入力したかどうかを検査してください。 トンネルの生成時には送信元にホスト名は使用できず、宛先にのみホスト名を使用できます。 IP バージョン 6 のトンネルの場合、使用可能な IP バージョン 6 のアドレスを入力したかどうかを検査してください。 `netstat -in` と入力したのに IP バージョン 6 のアドレスがない場合は、 (MAC アドレスを使用して) リンク・ローカル自動生成アドレス用に **/usr/sbin/autoconf6** (インターフェース) を実行するか、または **ifconfig** を使用して手動でアドレスを割り当てます。

エラー	発生する可能性のある問題と解決策
mktun コマンドを出すと、次のエラーが発生します。 「insert_tun_man4(): write failed : A system call received a parameter that is not valid.」	問題: 無効な ESP と AH の組み合わせが生成されたか、または必要なときに新規のヘッダー書式を使用せずにトンネルを生成しました。 修正方法: 問題の特定のトンネルがどの認証アルゴリズムを使用しているかどうかを検査します。 HMAC_MD5 および HMAC_SHA アルゴリズムには新規のヘッダー書式が必要であることに注意してください。 この新規のヘッダー書式は、 SMIT 高速パスの **ips4_basic** または **chtun** コマンドの *-z* パラメーターを使用することによって変更できます。 DES_CBC_4 は新規のヘッダー書式では使用することができません。
IP セキュリティーを使用すると、次のエラーが発生します。 「The installed bos.crypto is back level and must be updated.」	問題: **bos.net.ipsec.*** ファイルが新しいバージョンに更新されていますが、対応する **bos.crypto.*** ファイルが更新されていません。 修正方法: **bos.crypto.*** ファイルを、更新済みの **bos.net.ipsec.*** ファイルに対応するバージョンに更新します。

IKE (Internet Key Exchange) トンネル・エラーのトラブルシューティング**:**

以下のセクションでは、IKE (Internet Key Exchange) トンネルの使用時に発生するエラーについて説明します。

IKE (Internet Key Exchange) トンネル・プロセス・フロー*:*

このセクションでは、IKE (Internet Key Exchange) トンネルのプロセス・フローについて説明します。

IKE トンネルは、**ike** コマンドと以下のデーモンとの通信によってセットアップされます。

tmd Tunnel Manager デーモン

iked IKE ブローカー・デーモン (IKEv1 と IKEv2 デーモンの両方がシステム両方に構成済みの場合のみアクティブ)

isakmpd
IKEv1 デーモン

ikev2d
IKEv2 デーモン

cpsd 証明書プロキシー・デーモン

IKE トンネルを正しくセットアップするには、**tmd** および **isakmpd** デーモンが実行中でなければなりません。 IP セキュリティーがリブート時に始動するように設定されている場合、これらのデーモンも自動的に始動します。 デーモンが自動的に始動しない場合は、次のコマンドを入力して始動する必要があります。

```
startsrc -g ike
```

Tunnel Manager は、トンネルを始動する要求を **isakmpd** コマンドに出します。 トンネルが既に存在しているか、無効である (例えば、無効なリモート・アドレスを持っている) 場合は、エラーが報告されます。 ネゴシエーションが開始されると、それが完了するまでに、ネットワークの待ち時間によっては、しばらく時間がかかる場合があります。 **ike cmd=list** コマンドはトンネルの状態をリストするので、ネゴシエーションが成功したかどうかを判別することができます。 また、Tunnel Manager はイベントを debug、event、および information のレベルで **syslog** に記録するので、ネゴシエーションの進行をモニターするために使用することができます。

一連の処理は、次のとおりです。

1. トンネルを開始するには、**ike** コマンドを使用します。
2. **tmd** デーモンが **isakmpd** デーモンにキー管理のための接続要求を出します (フェーズ 1)。
3. **isakmpd** デーモンが SA created またはエラー・メッセージで応答します。
4. **tmd** デーモンが **isakmpd** デーモンにデータ管理トンネルのための接続要求を出します (フェーズ 2)。
5. **isakmpd** デーモンが SA created またはエラー・メッセージで応答します。
6. トンネル・パラメーターがカーネル・トンネル・キャッシュに挿入されます。
7. フィルター・ルールがカーネル動的フィルター・テーブルに追加されます。

マシンが応答側として機能している場合、**isakmpd** デーモンが Tunnel Manager **tmd** デーモンに対して、トンネルのネゴシエーションが正常に行われ、新規トンネルがカーネルに挿入されたことを通知します。 このような場合、**tmd** デーモンが接続要求を出さないまま、プロセスはステップ 3 から開始し、ステップ 7 まで続きます。

Parse Payload ロギング機能*:*

2 つのエンドポイント間のセキュリティー・アソシエーション (SA) は、 IKE メッセージを交換することにより確立されます。 Parse Payload 機能は、人間が読むことのできる形式のメッセージを解析します。

Parse Payload ロギングは、**/etc/isakmpd.conf** ファイルを編集することにより使用可能にすることができます。 **/etc/isakmpd.conf** ファイル内のログ・エントリーは、次のようになります。

```
information
```

Parse Payload が記録する IKE ペイロードのタイプは、 IKE メッセージの内容によって異なります。 例には、SA ペイロード、キー交換ペイロード、証明書要求ペイロード、および署名ペイロードが含まれています。 ISAKMP_MSG_HEADER の後に 5 つのペイロードが続く Parse Payload ログの例を以下に示します。

```
ISAKMP_MSG_HEADER
        Icookie : 0x9e539a6fd4540990, Rcookie : 0x0000000000000000
        Next Payload : 1(SA), Maj Ver : 1, Min Ver : 0
        Xchg Type : 4 (Aggressive), Flag= 0, Encr : No,COMMIT : No
        Msg ID  : 0x00000000
        len     : 0x10e(270)
SA Payload:
        Next Payload : 4(Key Exchange), Payload len : 0x34(52)
        DOI          : 0x1(INTERNET)
        bitmask      : 1(SIT_IDENTITY_ONLY
   Proposal Payload:
        Next Payload : 0(NONE), Payload len : 0x28(40)
        Proposal # : 0x1(1), Protocol-ID : 1(ISAKMP)
        SPI size : 0x0(0), # of Trans : 0x1(1)
   Transform Payload:
        Next Payload : 0(NONE), Payload len : 0x20(32)
        Trans # : 0x1(1), Trans.ID : 1(KEY_IKE)
        Attr : 1(Encr.Alg     ), len=0x2(2)
        Value=0x1(1),(DES-cbc)
        Attr : 2(Hash Alg     ), len=0x2(2)
        Value=0x1(1),(MD5)
        Attr : 3(Auth Method  ), len=0x2(2)
        Value=0x3(3),(RSA Signature)
        Attr : 4(Group Desc   ), len=0x2(2)
        Value=0x1(1),(default 768-bit MODP group)
        Attr : 11(Life Type    ), len=0x2(2)
        Value=0x1(1),(seconds)
        Attr : 12(Life Duration), len=0x2(2)
        Value=0x7080(28800)
```

```
Key Payload:
        Next Payload : 10(Nonce), Payload len : 0x64(100)

        Key Data :
        33 17 68 10 91 1f ea da 38 a0 22 2d 84 a3 5d 5d
        a0 e1 1f 42 c2 10 aa 8d 9d 14 0f 58 3e c4 ec a3
        9f 13 62 aa 27 d8 e5 52 8d 5c c3 cf d5 45 1a 79
        8a 59 97 1f 3b 1c 08 3e 2a 55 9b 3c 50 cc 82 2c
        d9 8b 39 d1 cb 39 c2 a4 05 8d 2d a1 98 74 7d 95
        ab d3 5a 39 7d 67 5b a6 2e 37 d3 07 e6 98 1a 6b

Nonce Payload:
        Next Payload : 5(ID), Payload len : 0xc(12)

        Nonce Data:
        6d 21 73 1d dc 60 49 93
ID Payload:
        Next Payload : 7(Cert.Req), Payload len : 0x49(73)
        ID type      : 9(DER_DN), Protocol : 0, Port = 0x0(0)
Certificate Request Payload:
        Next Payload : 0(NONE), Payload len : 0x5(5)
        Certificate Encoding Type: 4(X.509 Certificate - Signature)
```

各ペイロード内で、**Next Payload** フィールドは現在のペイロードに続くペイロードをポイントします。現在のペイロードが IKE メッセージ内の最後のものであれば、**Next Payload** フィールドの値はゼロ (None) になります。

例中のペイロードには、それぞれ進行中のネゴシエーションに関する情報が含まれています。 例えば、SA ペイロードは Proposal (提案) および Transform (変換) ペイロードを持ち、それらは、起動側が応答側に提示する暗号化アルゴリズム、認証モード、ハッシュ・アルゴリズム、SA ライフ・タイプ、および SA 存続期間を示しています。

また SA ペイロードは、 1 つ以上の Proposal ペイロードおよび 1 つ以上の Transform ペイロードから構成されています。 Proposal ペイロードの **Next Payload** フィールドは、これが唯一の Proposal ペイロードであれば値 0、 Proposal ペイロードがもうひとつ続く場合には、値 2 をとります。同様に、Transform ペイロードの **Next Payload** フィールドは、これが唯一の Transform ペイロードであれば値 0、 Transform ペイロードがもうひとつ続く場合には、値 3 をとります。次のようになります。

```
ISAKMP_MSG_HEADER
        Icookie : 0xa764fab442b463c6, Rcookie : 0x0000000000000000
        Next Payload : 1(SA), Maj Ver : 1, Min Ver : 0
        Xchg Type : 2 (ID protected), Flag= 0, Encr : No,COMMIT : No
        Msg ID  : 0x00000000
        len     : 0x70(112)
SA Payload:
        Next Payload : 0(NONE), Payload len : 0x54(84)
        DOI          : 0x1(INTERNET)
        bitmask      : 1(SIT_IDENTITY_ONLY
   Proposal Payload:
        Next Payload : 0(NONE), Payload len : 0x48(72)
        Proposal # : 0x1(1), Protocol-ID : 1(ISAKMP)
        SPI size : 0x0(0), # of Trans : 0x2(2)
   Transform Payload:
        Next Payload : 3(Transform), Payload len : 0x20(32)
        Trans # : 0x1(1), Trans.ID : 1(KEY_IKE)
        Attr : 1(Encr.Alg     ), len=0x2(2)
        Value=0x5(5),(3DES-cbc)
        Attr : 2(Hash Alg     ), len=0x2(2)
        Value=0x1(1),(MD5)
        Attr : 3(Auth Method  ), len=0x2(2)
        Value=0x1(1),(Pre-shared Key)
        Attr : 4(Group Desc   ), len=0x2(2)
        Value=0x1(1),(default 768-bit MODP group)
```

```
        Attr : 11(Life Type    ), len=0x2(2)
        Value=0x1(1),(seconds)
        Attr : 12(Life Duration), len=0x2(2)
        Value=0x7080(28800)
   Transform Payload:
        Next Payload : 0(NONE), Payload len : 0x20(32)
        Trans # : 0x2(2), Trans.ID : 1(KEY_IKE)
        Attr : 1(Encr.Alg     ), len=0x2(2)
        Value=0x1(1),(DES-cbc)
        Attr : 2(Hash Alg     ), len=0x2(2)
        Value=0x1(1),(MD5)
        Attr : 3(Auth Method  ), len=0x2(2)
        Value=0x1(1),(Pre-shared Key)
        Attr : 4(Group Desc   ), len=0x2(2)
        Value=0x1(1),(default 768-bit MODP group)
        Attr : 11(Life Type    ), len=0x2(2)
        Value=0x1(1),(seconds)
        Attr : 12(Life Duration), len=0x2(2)
        Value=0x7080(28800)
```

Parse Payload ログの IKE メッセージ・ヘッダーには、交換タイプ (メインモードまたはアグレッシブ・モード)、メッセージ全体の長さ、メッセージ ID などが含まれています。

証明書要求ペイロードは、応答側に対して証明書を要求します。 応答側は、別のメッセージで証明書を送信します。以下の例では、SA ネゴシエーションの一部としてピアに送られた、証明書ペイロードと署名ペイロードを示しています。 証明書データおよび署名データは、16 進形式で印刷されています。

```
ISAKMP_MSG_HEADER
        Icookie : 0x9e539a6fd4540990, Rcookie : 0xc7e0a8d937a8f13e
        Next Payload : 6(Certificate), Maj Ver : 1, Min Ver : 0
        Xchg Type : 4 (Aggressive), Flag= 0, Encr : No,COMMIT : No
        Msg ID  : 0x00000000
        len     : 0x2cd(717)
Certificate Payload:

        Next Payload : 9(Signature), Payload len : 0x22d(557)
        Certificate Encoding Type: 4(X.509 Certificate - Signature)
        Certificate: (len 0x227(551) in bytes
        82 02 24 30 82 01 8d a0 03 02 01 02 02 05 05 8e
        fb 3e ce 30 0d 06 09 2a 86 48 86 f7 0d 01 01 04
        05 00 30 5c 31 0b 30 09 06 03 55 04 06 13 02 46
        49 31 24 30 22 06 03 55 04 0a 13 1b 53 53 48 20
        43 6f 6d 6d 75 6e 69 63 61 74 69 6f 6e 73 20 53
        65 63 75 72 69 74 79 31 11 30 0f 06 03 55 04 0b
        13 08 57 65 62 20 74 65 73 74 31 14 30 12 06 03
        55 04 03 13 0b 54 65 73 74 20 52 53 41 20 43 41
        30 1e 17 0d 39 39 30 39 32 31 30 30 30 30 30 30
        5a 17 0d 39 39 31 30 32 31 32 33 35 39 35 39 5a
        30 3f 31 0b 30 09 06 03 55 04 06 13 02 55 53 31
        10 30 0e 06 03 55 04 0a 13 07 49 42 4d 2f 41 49
        58 31 1e 30 1c 06 03 55 04 03 13 15 62 61 72 6e
        65 79 2e 61 75 73 74 69 6e 2e 69 62 6d 2e 63 6f
        6d 30 81 9f 30 0d 06 09 2a 86 48 86 f7 0d 01 01
        01 05 00 03 81 8d 00 30 81 89 02 81 81 00 b2 ef
        48 16 86 04 7e ed ba 4c 14 d7 83 cb 18 40 0a 3f
        55 e9 ad 8f 0f be c5 b6 6d 19 ec de 9b f5 01 a6
        b9 dd 64 52 34 ad 3d cd 0d 8e 82 6a 85 a3 a8 1c
        37 e4 00 59 ce aa 62 24 b5 a2 ea 8d 82 a3 0c 6f
        b4 07 ad 8a 02 3b 19 92 51 88 fb 2c 44 29 da 72
        41 ef 35 72 79 d3 e9 67 02 b2 71 fa 1b 78 13 be
        f3 05 6d 10 4a c7 d5 fc fe f4 c0 b8 b8 fb 23 70
        a6 4e 16 5f d4 b1 9e 21 18 82 64 6d 17 3b 02 03
        01 00 01 a3 0f 30 0d 30 0b 06 03 55 1d 0f 04 04
        03 02 07 80 30 0d 06 09 2a 86 48 86 f7 0d 01 01
        04 05 00 03 81 81 00 75 a4 ee 9c 3a 18 f2 de 5d
        67 d4 1c e4 04 b4 e5 b8 5e 9f 56 e4 ea f0 76 4a
```

```
        d0 e4 ee 20 42 3f 20 19 d4 25 57 25 70 0a ea 41
        81 3b 0b 50 79 b5 fd 1e b6 0f bc 2f 3f 73 7d dd
        90 d4 08 17 85 d6 da e7 c5 a4 d6 9a 2e 8a e8 51
        7e 59 68 21 55 4c 96 4d 5a 70 7a 50 c1 68 b0 cf
        5f 1f 85 d0 12 a4 c2 d3 97 bf a5 42 59 37 be fe
        9e 75 23 84 19 14 28 ae c4 c0 63 22 89 47 b1 b6
        f4 c7 5d 79 9d ca d0
Signature Payload:
        Next Payload : 0(NONE), Payload len : 0x84(132)

        Signature: len 0x80(128) in bytes
        9d 1b 0d 90 be aa dc 43 95 ba 65 09 b9 00 6d 67
        b4 ca a2 85 0f 15 9e 3e 8d 5f e1 f0 43 98 69 d8
        5c b6 9c e2 a5 64 f4 ef 0b 31 c3 cb 48 7c d8 30
        e3 a2 87 f4 7c 9d 20 49 b2 39 00 fa 8e bf d9 b0
        7d b4 8c 4e 19 3a b8 70 90 88 2c cf 89 69 5d 07
        f0 5a 81 58 2e 15 40 37 b7 c8 d6 8c 5c e2 50 c3
        4d 19 7e e0 e7 c7 c2 93 42 89 46 6b 5f f8 8b 7d
        5b cb 07 ea 36 e5 82 9d 70 79 9a fe bd 6c 86 36
```

デジタル証明書と署名モードの問題*:*

以下は、デジタル証明書と署名モードについて発生する可能性がある問題に対する、解決策です。

エラー	発生する可能性のある問題と解決策
エラー: **cpsd** (証明書プロキシー・サーバー・デーモン) が始動しません。ログ・ファイルに以下のようなエントリーが示されます。 「Sep 21 6:02:00 ripple CPS[19950]: Init():Lo adCaCerts() failed, rc =-12」	問題: 証明書データベースが開いていないか、検出されませんでした。 修正方法: Key Manager 証明書データベースが **/etc/security** 内に存在することを確認します。 ファイル **ikekey.crl**、**ikekey.kdb**、**ikekey.rdb**、**ikekey.sth** がデータベースを構成しています。 **ikekey.sth** ファイルのみが不足する場合は、 Key Manager データベースの作成時に stash password オプションが選択されていませんでした。 IP セキュリティーにおいてデジタル証明書を使用可能にするには、パスワードを隠しておく必要があります。 (詳しくは、『鍵データベースの作成』を参照してください。)
エラー: 証明書を受け取る際に Key Manager が次のエラーを示します。 「Invalid Base64-encoded data was found」	問題: 証明書ファイル内で、不要なデータが検出されました。あるいは、データが欠落しているか破壊されています。 修正方法:「DER」エンコードの証明書は、以下の文字列に囲まれていなければなりません。 BEGIN および END CERTIFICATE 文字列以外の文字が前後にあってはいけません。 -----BEGIN CERTIFICATE----- MIICMTCCAZqgAwIBAgIFFKZtANowDQYJKoZIhvcNAQEFBQAwXDELMAkGA1UEBhMC RkkxJDAiBgNVBAoTG1NTSCBDb21tdW5pY2F0aW9ucyBTZWN1cml0eTERMA8GA1UE CxMIV2ViIHRlc3QxFDASBgNVBAMTC1Rlc3QgUlNBIENBMB4XDTk5MDkyMTAwMDAw MFoXDTk5MTAyMTIzNTk1OVowOzELMAkGA1UEBhMCVVMxDDAKBgNVBAoTA0lCTTEe MBwGA1UEAxMVcmlwcGxlLmF1c3Rpbi5pYm0uY29tMIGfMA0GCSqGSIb3DQEBAQUA A4GNADCBiQKBgQC5EZqo6n7tZrpAL6X4L7mf4yXQSm+m/NsJLhp6afbFpPvXgYWC wq4pvOtvxgum+FHrE0gysNjbKkE4Y6ixC9PGGAKHnhM3vrmvFjnl1G6KtyEz58Lz BWW39QS6NJ1LqqP1nT+y3+Xzvfv8Eonqzno8mglCWMX09SguLmWoU1PcZQIDAQAB oyAwHjALBgNVHQ8EBAMCBaAwDwYDVR0RBAgwBocECQNhhzANBgkqhkiG9w0BAQUF AOBgQA6bgp4Zay34/fyAlyCkNNAYJRrN3Vc4NHN7IGjUziN6jK5UyB5zL37FERW hT9ArPLzK7yEZs+MDNvB0bosyGWEDYPZr7EZHhYcoBP4/cd0V5rBFmA8Y2gUthPi Ioxpi4+KZGHYyLqTrm+8Is/DVJaQmCGRPynHK35xjT6WuQtiYg== -----END CERTIFICATE----- 以下のオプションが、この問題の診断と解決に役立ちます。 • データが欠落しているか破壊されている場合、証明書を再作成します。 • ASN.1 パーサー (インターネット WWW より入手可) を使用して、証明書を正しく解析し、証明書が有効であるかを検査します。

エラー	発生する可能性のある問題と解決策
エラー: 個人証明書を受け取る際に Key Manager が次のエラーを示します。 「No request key was found for the certificate」	問題: 受け取る個人証明書に対する個人証明書要求が存在しません。 修正方法: 個人証明書要求を再作成して、新規に証明書を要求します。
エラー: IKE ネゴシエーションが失敗し、ログ・ファイルに以下のようなエントリーが示されます。 「inet_cert_service:: channelOpen(): clientInitIPC() :error,rc =2 (No such file or directory)」	問題: **cpsd** が実行されていないか、停止しました。 修正方法: IP セキュリティーを開始します。これにより適切なデーモンが始動します。
エラー: IKE ネゴシエーションが失敗し、ログ・ファイルに以下のようなエントリーが示されます。 「CertRepo::GetCertObj: DN Does Not Match: ("/C=US/O=IBM/ CN=ripple.austin.ibm.com")」	問題: IKE トンネルを定義する際に入力した X.500 識別名 (DN) が、個人証明書内の X.500 DN に一致しません。 修正方法: IKE トンネル定義が、証明書内の識別名に一致するよう変更します。

トレース機能**:**

トレースは、カーネル・イベントのトレース用のデバッグ機能です。トレースは、カーネル・フィルターおよびトンネル・コード内で発生するイベントまたはエラーに関するより詳細な情報を入手するために使用することができます。

「Advanced IP Security Configuration (拡張 IP セキュリティー構成)」メニューを介して、 SMIT IP セキュリティー・トレース機能を使用できます。 このトレース機能が取り込む情報には、エラー情報、フィルター、フィルター情報、トンネル、トンネル情報、カプセル化/カプセル解除、カプセル化情報、暗号、および暗号情報が含まれています。 設計によっては、エラー・トレース・フックが最も重要な情報を提供します。 情報トレース・フックは重大な情報を生成しますが、システム・パフォーマンスに影響を与えることもあります。 このトレースは問題の手掛かりを提供し、サービス技術員に問題を説明するときにも必要です。

トレースを使用可能にするには、IPSec デバイスを構成し、各 IPSec サブコンポーネントのトレース・レベルを 7 (有用なカーネル・トレース・データを生成する) に設定します。 IPSec デバイスを構成しないと、コンポーネント・トレース制御コマンドによって IPSec 関連エントリーがリストされません。 IPSec トレースを開始するには、SMIT 高速パス **smit ips4_start** (IP バージョン 4) または **smit ips6_start** (IP バージョン 6) を使用します。

注**:** IPSec コンポーネント・トレースが正しく設定されていないと、取り込まれたトレースは空になります。

カーネル・トレース・データを取り込むには、次のステップを実行します。

1. 現在のトレース・レベル設定を表示するために、すべてのコンポーネントを照会します。

   ```
   # ctctrl -q
   ```

2. IPSec コンポーネントおよびサブコンポーネントを確認します。 デフォルトのトレース・レベル 3 では、コンポーネントは以下のように初期表示されます。コンポーネントの初期のデフォルト・トレース・レベルを表示するには、次のように入力します。

```
# ctctrl -q -c ipsec -r
```

コンポーネント名	別名の有無	メモリー・トレース/レベル	システム・トラック/レベル	バッファー・サイズ/割り振り済み
ipsec	NO	ON/3	ON/3	40960/YES
.capsulate	NO	ON/3	ON/3	10240/YES
.filter	NO	ON/3	ON/3	10240/YES
.tunnel	NO	ON/3	ON/3	10240/YES

3. IPSec およびサブコンポーネントのトレース・レベルを 7 (カーネル・トレースをサポートする) に上げます。次のように入力してください。

   ```
   # ctctrl systracelevel=7 -c ipsec -r
   ```

4. IPSec とそのサブコンポーネントのトレース・レベルが変更されたことを確認するための照会を行います。次のように入力してください。

   ```
   # ctctrl -q -c ipsec -r
   ```

コンポーネント名	別名の有無	メモリー・トレース/レベル	システム・トラック/レベル	バッファー・サイズ/割り振り済み
ipsec	NO	ON/3	ON/7	40960/YES
.capsulate	NO	ON/3	ON/7	10240/YES
.filter	NO	ON/3	ON/7	10240/YES
.tunnel	NO	ON/3	ON/7	10240/YES

トレース機能にアクセスする場合、 SMIT 高速パス **smit ips4_tracing** (IP バージョン 4) か **smit ips6_tracing** (IP バージョン 6) を使用してください。 **smit ips4_tracing**、**smit ips6_tracing**、またはコマンド・ライン・トレース機能によって行われたカーネル・トレースにより、有効な IPSec トレース・データが生成されます。

ipsecstat コマンド:

ipsecstat コマンドを使用すると、IP セキュリティー・デバイスの状況、IP セキュリティー暗号アルゴリズム、および IP セキュリティー・パケットの統計情報をリストすることができます。

ipsecstat コマンドを実行すると、次のようなサンプル・レポートが生成されます。このサンプル・レポートは、IP セキュリティー・デバイスが使用可能状態にあること、 3 つの認証アルゴリズムがインストールされていること、3 つの暗号化アルゴリズムがインストールされていること、およびパケット・アクティビティーの現行レポートがあることを示しています。 この情報は、IP セキュリティー・トラフィックをトラブルシューティングする場合、どこに問題があるのかを判別するのに役立ちます。

```
IP Security Devices:
ipsec_v4 Available
ipsec_v6 Available

Authentication Algorithm:
HMAC_MD5 -- Hashed MAC MD5 Authentication Module
HMAC_SHA -- Hashed MAC SHA Hash Authentication Module
KEYED_MD5 -- Keyed MD5 Hash Authentication Module

Encryption Algorithm:
CDMF -- CDMF Encryption Module
DES_CBC_4 -- DES CBC 4 Encryption Module
DES_CBC_8 -- DES CBC 8 Encryption Module
3DES_CBC -- Triple DES CBC Encryption Module
```

```
IP Security Statistics -
Total incoming packets:  1106
Incoming AH packets:326
Incoming ESP packets:  326
Srcrte packets allowed:  0
Total outgoing packets:844
Outgoing AH packets:527
Outgoing ESP packets:  527
Total incoming packets dropped:  12
  Filter denies on input:  12
  AH did not compute: 0
  ESP did not compute:0
  AH replay violation:0
  ESP replay violation:  0
Total outgoing packets dropped:0
  Filter denies on input:0
Tunnel cache entries added: 7
Tunnel cache entries expired:  0
Tunnel cache entries deleted:  6
```

注: 現在は全世界で DES が使用可能となっているため、CDMF を使用する必要はありません。CDMF を使用するトンネルは、DES または Triple DES を使用するように再構成してください。

IP セキュリティー・リファレンス

IP セキュリティー用のコマンドおよびメソッドがあります。 IKE トンネル、フィルター、および事前共有鍵をマイグレーションすることもできます。

コマンドのリスト:

以下の表では、コマンドをリストします。

コマンド	目的
ike cmd=activate	IKE (Internet Key Exchange) ネゴシエーションを開始します
ike cmd=remove	IKE トンネルを非活動状態にします
ike cmd=list	IKE トンネルをリストします
ikedb	IKE トンネル・データベースへのインターフェースを提供します
gentun	トンネル定義を作成します
mktun	トンネル定義を活動化します
chtun	トンネル定義を変更します
rmtun	トンネル定義を除去します
lstun	トンネル定義をリストします
exptun	トンネル定義をエクスポートします
imptun	トンネル定義をインポートします
genfilt	フィルター定義を作成します
mkfilt	フィルター定義を活動化します
mvfilt	フィルター・ルールを移動します
chfilt	フィルター定義を変更します
rmfilt	フィルター定義を除去します
lsfilt	フィルター定義をリストします
expfilt	フィルター定義をエクスポートします
impfilt	フィルター定義をインポートします
ipsec_convert	IP セキュリティーの状況をリストします
ipsecstat	IP セキュリティーの状況をリストします
ipsectrcbuf	IP セキュリティー・トレース・バッファーの内容をリストします
unloadipsec	暗号モジュールをアンロードします

メソッドのリスト:

以下に、メソッドのリストを示します。

defipsec
IP バージョン 4 または IP バージョン 6 用の IP セキュリティーのインスタンスを定義します

cfgipsec
ipsec_v4 または **ipsec_v6** を構成およびロードします

ucfgipsec
ipsec_v4 または **ipsec_v6** を構成解除します

IP セキュリティーのマイグレーション**:**

AIX オペレーティング・システムの旧バージョンから IKE トンネル、フィルターおよび事前共有鍵をマイグレーションすることができます。

IKE トンネルのマイグレーション*:*

トンネルをマイグレーションするには、以下のステップを実行します。

1. **bos.net.ipsec.keymgt.pre_rm.sh** スクリプトを実行します。このスクリプトを実行すると、次のファイルが **/tmp** ディレクトリーに作成されます。
   a. **p2proposal.bos.net.ipsec.keymgt**
   b. **p1proposal.bos.net.ipsec.keymgt**
   c. **p1policy.bos.net.ipsec.keymgt**
   d. **p2policy.bos.net.ipsec.keymgt**
   e. **p1tunnel.bos.net.ipsec.keymgt**
   f. **p2tunnel.bos.net.ipsec.keymgt**

   重要**:** このスクリプトは 1 回限り実行します。 データベースを更新し、このスクリプトを再度実行すると、すべてのファイルが失われ、それらを検索できなくなります。 トンネルをマイグレーションする前に、 304 ページの『bos.net.ipsec.keymgt.pre_rm.sh スクリプト』のスクリプトを読んでください。
2. スクリプトにより作成されたファイルおよび **/tmp/lpplevel** ファイルを外部メディア (CD あるいはフロッピー・ディスク) に保存します。

事前共有鍵のマイグレーション*:*

事前共有鍵のフォーマットを更新するには、以下のステップを実行します。

IKE トンネル事前共有鍵データベースもマイグレーション時に破壊されます。 事前共有鍵フォーマットを更新するには、マイグレーション済みのシステム上で、以下のステップを実行してください。

1. 次のコマンドを実行して、**ikedb -g** コマンドの出力を保存します。
   ```
   ikedb -g > out.keys
   ```
2. **out.keys** ファイルを編集して、事前共有鍵フォーマットとして FORMAT=ASCII を FORMAT=HEX に置き換える。
3. 以下のコマンドを実行して、XML ファイルを入力する。
   ```
   ikedb -pF out.keys
   ```

フィルターのマイグレーション*:*

フィルターをマイグレーションするには、以下のステップを実行します。

1. 以下のステップを実行して、SMIT を用いて、フィルター・ルール・ファイルを **/tmp** ディレクトリーにエクスポートする。
   a. **smitty ipsec4** コマンドを実行する。
   b. 「拡張 IP セキュリティー構成」–> 「IP セキュリティー・フィルター・ルールの構成」–> 「IP セキュリティー・フィルター・ルールのエクスポート」の順に選択する。
   c. ディレクトリー名に /tmp を入力する。
   d. 「Filter Rules (フィルター・ルール)」オプションの下で F4 を押し、リストから「**all (**すべて**)**」を選択する。
   e. 「Enter」を押して、外部メディアに **/tmp/ipsec_fltr_rule.exp** ファイルのフィルター・ルールを保存する。

   マイグレーションするすべてのシステムについて、前のバージョンの AIX オペレーティング・システムからこのプロセスを実行する。
2. スクリプトにより作成された 6 つのトンネル・ファイル、**/tmp/lpplevel** ファイル、および **/tmp/ipsec_fltr_rule.exp** ファイルを、マイグレーション済みのシステム上の **/tmp** ディレクトリーにコピーする。
3. **bos.net.ipsec.keymgt.post_i.sh** スクリプトを実行し、トンネル構成をデータベースに再移植する。
4. **ikedb -g** コマンドを実行し、トンネルがデータベースに存在することを検証する。

   注**:** データベース内でトンネル情報が見つからない場合は、スクリプトを再度実行します。ただし、**/tmp** ディレクトリー内のすべての ***.loaded** ファイルの名前を元の名前に名前変更してください。

マイグレーション済みのシステム上において、フィルター・データベースはマイグレーション後に破壊されます。 マイグレーション済みシステムで **lsfilt** コマンドを実行すると、以下のエラーが発生します。

```
Cannot get ipv4 default filter rule
```

フィルター・データベースを更新するには、以下のステップを実行します。

1. /etc/security ディレクトリー内の ipsec_filter ファイルと ipsec_filter.vc ファイルを、新規にマイグレーションされたシステム上の無破壊ファイルと取り替えます。 これらのファイルがない場合は、IBM サービス担当へ要求してください。
2. 以下のステップを実行して、SMIT を用いて、フィルター・ルール・ファイルを **/tmp** ディレクトリーへインポートする。
   a. **smitty ipsec4** コマンドを実行する。
   b. 「拡張 IP セキュリティー構成」–> 「IP セキュリティー・フィルター・ルールの構成」–> 「IP セキュリティー・フィルター・ルールのインポート」の順に選択する。
   c. ディレクトリー名に /tmp を入力する。
   d. 「**Filter Rules (**フィルター・ルール**)**」オプションの下で **F4** を押し、リストから「**all (**すべて**)**」を選択する。
   e. Enter を押して、「filter rules (フィルター・ルール)」を再作成する。 「filter rules (フィルター・ルール)」は、 SMIT を介して、あるいは **lsfilt** コマンドを使用して、リストすることができます。

bos.net.ipsec.keymgt.pre_rm.sh スクリプト*:*

bos.net.ipsec.keymgt.pre_rm.sh スクリプトは、トンネル・データベースのコンテンツを AIX オペレーティング・システムが稼働中のシステム上に保存します。

```
#!/usr/bin/ksh
keymgt_installed=`lslpp -Lqc bos.net.ipsec.keymgt 2>/dev/null | awk -F: '{print $6}' | head -1`

if [ ! "$keymgt_installed" ]
then
  exit 0
fi


#  Copy the database to a save directory in case changes fail
if [ -d /etc/ipsec/inet/DB ]
then
  cp -R /etc/ipsec/inet/DB /etc/ipsec/inet/DB.sav || exit $?
fi

#  Remember the level you are migrating from
VRM=$(LANG=C lslpp -Lqc bos.net.ipsec.keymgt 2>/dev/null | awk -F: '{print $3}' | ¥
awk -F. '{print $1"."$2"."$3}')
VR=${VRM%.*}
echo $VRM > /tmp/lpplevel

IKEDB=$(which ikedb) || IKEDB=/usr/sbin/ikedb

XMLFILE=/tmp/full_ike_database.bos.net.ipsec.keymgt
PSKXMLFILE=/tmp/psk_ike_database.bos.net.ipsec.keymgt

#  See if ikedb exists.
if [ -f $IKEDB ]
then

  # If either of the ikedb calls below fails, that's OK.  Just remove the
  # resulting file (which may contain garbage) and continue.  The post_i
  # script will simply not import the file if it doesn't exist, which will
  # mean part or all of the IKE database is lost, but this is preferable
  # to exiting the script with an error code, which causes the entire
  # migration to fail.

  $IKEDB -g > $XMLFILE
  if [ $? -ne 0 ]
  then
    rm -f $XMLFILE  || exit $?
  fi

  if [[ $VR = "5.1" ]]; then
    # This is a special case.  The 5.1 version of ikedb is the only
    # one that does not include preshared keys in the full database
    # output.  So we have to retrieve those separately.
    $IKEDB -g -t IKEPresharedKey > $PSKXMLFILE
    if [ $? -ne 0 ]
    then
     rm -f $PSKXMLFILE  || exit $?
    fi
  fi

#  Make sure ikegui command is installed
elif [ -f /usr/sbin/ikegui ]
then

  #  Get database information and save to /tmp
  /usr/sbin/ikegui 0 1 0 0 > /tmp/p1proposal.bos.net.ipsec.keymgt 2>/dev/null
  RC=$?
  if [[ $RC -ne 0 ]]
```

```
  then
    rm -f /tmp/p1proposal.bos.net.ipsec.keymgt || exit $?
  fi

  /usr/sbin/ikegui 0 1 1 0 > /tmp/p1policy.bos.net.ipsec.keymgt 2>/dev/null
  RC=$?
  if [[ $RC -ne 0 ]]
  then
    rm -f /tmp/p1policy.bos.net.ipsec.keymgt || exit $?
  fi

  /usr/sbin/ikegui 0 2 0 0 > /tmp/p2proposal.bos.net.ipsec.keymgt 2>/dev/null
  RC=$?
  if [[ $RC -ne 0 ]]
  then
    rm -f /tmp/p2proposal.bos.net.ipsec.keymgt || exit $?
  fi

  /usr/sbin/ikegui 0 2 1 0 > /tmp/p2policy.bos.net.ipsec.keymgt 2>/dev/null
  RC=$?
  if [[ $RC -ne 0 ]]
  then
    rm -f /tmp/p2policy.bos.net.ipsec.keymgt || exit $?
  fi

  /usr/sbin/ikegui 0 1 2 0 > /tmp/p1tunnel.bos.net.ipsec.keymgt 2>/dev/null
  RC=$?
  if [[ $RC -ne 0 ]]
  then
    rm -f /tmp/p1tunnel.bos.net.ipsec.keymgt || exit $?
  fi

  /usr/sbin/ikegui 0 2 2 0 > /tmp/p2tunnel.bos.net.ipsec.keymgt 2>/dev/null
  RC=$?
  if [[ $RC -ne 0 ]]
  then
    rm -f /tmp/p2tunnel.bos.net.ipsec.keymgt || exit $?
  fi

fi
```

bos.net.ipsec.keymgt.post_i.sh スクリプト*:*

bos.net.ipsec.keymgt.post_i.sh スクリプトは、トンネル・データベースの内容を、AIX オペレーティング・システムが稼働中のマイグレーション済みシステムにロードします。

```
#!/usr/bin/ksh

function PrintDot {
    echo "echo ¥c"
    echo "¥".¥c"
    echo "¥¥¥c¥c"
    echo "¥"¥c"
    echo
}

function P1PropRestore {
    while :
    do
        read NAME
        read MODE
        if [[ $? = 0 ]]; then
            echo "ikegui 1 1 0 $NAME $MODE ¥c"
            MORE=1
            while [[ $MORE = 1 ]];
            do
```

```
                read AUTH
                read HASH
                read ENCRYPT
                read GROUP
                read TIME
                read SIZE
                read MORE
                echo "$AUTH $HASH $ENCRYPT $GROUP $TIME $SIZE $MORE ¥c"
            done
            echo " > /dev/null 2>&1"
            PrintDot
        else
            return 0
        fi
    done
}

function P2PropRestore {
    while :
    do
        read NAME
        FIRST=yes
        MORE=1
        while [[ $MORE = 1 ]];
        do
            read PROT
            if [[ $? = 0 ]]; then
                read AH_AUTH
                read ESP_ENCR
                read ESP_AUTH
                read ENCAP
                read TIME
                read SIZE
                read MORE
                if [[ $FIRST = "yes" ]]; then
                    echo "ikegui 1 2 0 $NAME $MODE ¥c"
                fi
                echo "$PROT $AH_AUTH $ESP_ENCR $ESP_AUTH $ENCAP $TIME $SIZE $MORE ¥c"
                FIRST=no
            else
                return 0
            fi
        done
        echo " > /dev/null 2>&1"
        PrintDot
    done
}

function P1PolRestore {
    while :
    do
        read NAME
        read ROLE
        if [[ $? = 0 ]]; then
            read TIME
            read SIZE
            read OVERLAP
            read TTIME
            read TSIZE
            read MIN
            read MAX
            read PROPOSAL
            echo "ikegui 1 1 1 $NAME $ROLE $OVERLAP $TTIME $TSIZE $MIN $MAX 1 0 0 $PROPOSAL > ¥
/dev/null 2>&1"
            PrintDot
        else
            return 0
```

```
        fi
    done
}

function P2PolRestore {
    while :
    do
        read NAME
        read ROLE
        if [[ $? = 0 ]]; then
            read IPFS
            read RPFS
            read TIME
            read SIZE
            read OVERLAP
            read TTIME
            read TSIZE
            read MIN
            read MAX
            echo "ikegui 1 2 1 $NAME $ROLE $IPFS $RPFS $OVERLAP $TTIME $TSIZE $MIN $MAX 1 0 0 ¥c"
            MORE=1
            while [[ $MORE = 1 ]];
            do
                read PROPOSAL
                read MORE
                echo "$PROPOSAL $MORE ¥c"
                FIRST=no
            done
        else
            return 0
        fi
        echo " > /dev/null 2>&1"
        PrintDot
    done
}

function P1TunRestore {
    while :
    do
        read TUNID
        read NAME
        if [[ $? = 0 ]]; then
            read LID_TYPE
            read LID
            if [[ $LPPLEVEL = "4.3.3" ]]; then
                read LIP
            fi
            read RID_TYPE
            read RID
            read RIP
            read POLICY
            read KEY
            read AUTOSTART
            echo "ikegui 1 1 2 0 $NAME $LID_TYPE ¥"$LID¥" $LIP $RID_TYPE ¥"$RID¥" ¥
$RIP $POLICY $KEY $AUTOSTART > /dev/null 2>&1"
            PrintDot
        else
            return 0
        fi
    done
}

function P2TunRestore {
    while :
    do
        read TUNID
        read NAME
```

```
        if [[ $? = 0 ]]; then
            read P1TUN
            read LTYPE
            read LID
            read LMASK
            read LPROT
            read LPORT
            read RTYPE
            read RID
            read RMASK
            read RPROT
            read RPORT
            read POLICY
            read AUTOSTART
            echo "ikegui 1 2 2 0 $NAME $P1TUN $LTYPE $LID $LMASK $LPROT $LPORT $RTYPE
              ¥$RID $RMASK $RPROT $RPORT $POLICY $AUTOSTART > /dev/null 2>&1"
            PrintDot
        else
            return 0
        fi
    done
}

function allRestoreWithIkedb {

    ERRORS=/tmp/ikedb_msgs.bos.net.ipsec.keymgt
    echo > $ERRORS
    $IKEDB -p $XMLFILE 2>> $ERRORS
    if [ -f $PSKXMLFILE ]
    then
        $IKEDB -p $PSKXMLFILE 2>> $ERRORS
    fi

}

P1PROPFILE=/tmp/p1proposal.bos.net.ipsec.keymgt
P2PROPFILE=/tmp/p2proposal.bos.net.ipsec.keymgt
P1POLFILE=/tmp/p1policy.bos.net.ipsec.keymgt
P2POLFILE=/tmp/p2policy.bos.net.ipsec.keymgt
P1TUNFILE=/tmp/p1tunnel.bos.net.ipsec.keymgt
P2TUNFILE=/tmp/p2tunnel.bos.net.ipsec.keymgt
XMLFILE=/tmp/full_ike_database.bos.net.ipsec.keymgt
PSKXMLFILE=/tmp/psk_ike_database.bos.net.ipsec.keymgt
CMD_FILE=/tmp/commands
IKEDB=$(which ikedb) || IKEDB=/usr/sbin/ikedb

echo "building ISAKMP database ¥n"
$IKEDB -x || exit $?

if [ -f $XMLFILE ]; then
    echo "¥nRestoring database entries¥c"
    allRestoreWithIkedb
    echo "¥ndone¥n"

elif [ -f /tmp/*.bos.net.ipsec.keymgt ]; then
    echo "¥nRestoring database entries¥c"

    LPPLEVEL=`cat /tmp/lpplevel`

    echo > $CMD_FILE
    touch $P1PROPFILE; P1PropRestore < $P1PROPFILE >> $CMD_FILE
    touch $P2PROPFILE; P2PropRestore < $P2PROPFILE >> $CMD_FILE
    touch $P1POLFILE; P1PolRestore < $P1POLFILE >> $CMD_FILE
    touch $P2POLFILE; P2PolRestore < $P2POLFILE >> $CMD_FILE
    touch $P1TUNFILE; P1TunRestore < $P1TUNFILE >> $CMD_FILE
    touch $P2TUNFILE; P2TunRestore < $P2TUNFILE >> $CMD_FILE
```

```
    mv $P1PROPFILE ${P1PROPFILE}.loaded
    mv $P2PROPFILE ${P2PROPFILE}.loaded
    mv $P1POLFILE ${P1POLFILE}.loaded
    mv $P2POLFILE ${P2POLFILE}.loaded
    mv $P1TUNFILE ${P1TUNFILE}.loaded
    mv $P2TUNFILE ${P2TUNFILE}.loaded

    ksh $CMD_FILE

    echo "done¥n"
fi
```

ネットワーク・ファイルシステムのセキュリティー

ネットワーク・ファイルシステム (NFS) は広く使用可能なテクノロジーで、ネットワーク上の各種のホスト間でデータを共有できるようにします。

NFS は、DES に加えて、Kerberos 5 認証の使用もサポートするようになりました。Kerberos 5 セキュリティーは、RPCSEC_GSS と呼ばれるプロトコル・メカニズムに基づいています。

標準の UNIX 認証システムに加えて、NFS は個々のメッセージを基準にしてネットワーク内のユーザーおよびマシンを認証する手段も提供します。 この追加の認証システムは、データ暗号化規格 (DES) の暗号化および公開鍵暗号方式を使用しています。

NFS は、DES に加えて、Kerberos 5 認証の使用もサポートするようになりました。Kerberos 5 セキュリティーは、RPCSEC_GSS と呼ばれるプロトコル・メカニズムに基づいています。 NFS での Kerberos 認証の管理および使用の方法については、「*NFS* 管理ガイド」を参照してください。

ネットワーク・ファイルシステム保護の一般ガイドライン

ネットワーク・ファイルシステム (NFS) の保護の助けとなるいくつかのガイドラインがあります。

- 最新のソフトウェア・パッチがインストールされていることを確認する。 セキュリティー問題に関連したパッチは特に重要です。所与のインフラストラクチャーの全ソフトウェアを保守する必要があります。 例えば、オペレーティング・システムにパッチをインストールしていながら、Web サーバーへのパッチのインストールを怠ると、Web サーバーも更新していれば回避できたはずの、ご使用の環境に接続する手段を、アタッカーに与えることになります。 最新の使用可能なセキュリティー情報について IBM System p Security Alerts にサブスクライブするには、Web アドレスの http://www14.software.ibm.com/webapp/set2/subscriptions/pqvcmjd にアクセスしてください。
- 必要最低限の特権を備えたファイルシステムをエクスポートできるように、 NFS サーバーを構成する。 ユーザーが必要とするのがファイルシステムからの読み取りのみであれば、ファイルシステムへの書き込みができないようにする必要があります。こうすることにより、重要なデータの上書き、構成ファイルの変更、あるいはエクスポート・ファイルシステムへの悪質な実行可能コードの書き込みの試みを軽減することができます。 SMIT を使用するか、または **/etc/exports** ファイルを直接編集することによって、特権を指定してください。
- ファイルシステムにアクセスする必要があるユーザーに対して明示的にエクスポートできるように、NFS サーバーを構成する。 NFS の大半のインプリメンテーションでは、どの NFS クライアントに特定のファイルシステムへのアクセスを許すかを指定できるようになっています。 これにより、無許可ユーザーによるファイルシステムへのアクセスの試みを軽減できます。 特に、ファイルシステムを NFS サーバー自身へエクスポートするような NFS サーバーを構成しないでください。
- エクスポート・ファイルシステムは自らの区画内に留めてください。 アタッカーは、エクスポート・ファイルシステムが満杯になるまで当該ファイルシステムへ書き込みを行い、システム低下を発生させる可能性があります。 これにより、ファイルシステムは、それを必要とする他のアプリケーションやユーザーが使用できなくなる場合があります。

- NFS クライアントに、root ユーザー資格情報または不明のユーザー資格情報を使ってファイルシステムにアクセスさせないようにする。 ほとんどの NFS インプリメンテーションは、特権のあるユーザーまたは不明のユーザーからの要求を、特権のないユーザーにマップするように構成できます。 これにより、アタッカーがファイルにアクセスしようとしたり、特権のあるユーザーとしてファイル操作を行うシナリオを回避できます。
- NFS クライアントに、エクスポート・ファイルシステム上で suid および sgid プログラムを実行させない。 これによって、NFS クライアントが特権を使って悪質なコードを実行することを防止できます。 もしも、特権のある所有者またはグループが所有する実行可能モジュールをアタッカーが作成できるとすると、重大な損害を NFS サーバーに与える可能性があります。 これは **mknfsmnt -y** コマンド・オプションを指定すれば可能です。
- セキュア NFS を使用する。 セキュア NFS は、DES 暗号化機能を用いて RPC トランザクションに関与するホストを認証します。 RPC は、NFS がホスト間で要求を伝達する場合に使用するプロトコルです。 セキュア NFS アタッカーが RPC 要求内のタイム・スタンプを暗号化して RPC 要求をスプーフする企てを軽減します。 受信側がタイム・スタンプを正常に暗号化解除し、それが正しいことを確認すると、RPC 要求がトラステッド・ホストから送信されたものであることの確認となります。
- NFS が不要なときは、オフにしておく。 これによって、侵入者にとって使用可能となりうるアタック・ベクトルの数が減少します。

NFS は、Triple-DES および Single-DES に加えて、Kerberos 5 認証を伴う AES 暗号化タイプの使用もサポートするようになりました。AES 暗号化タイプを使用するための Kerberos 5 の構成方法については、「NFS System Management」ガイドを参照してください。

関連概念:

309 ページの『ネットワーク・ファイルシステムのセキュリティー』

関連情報:

NFS 構成のチェックリスト

システム起動時の NFS デーモンの始動

NFS サーバーの構成

NFS クライアントの構成

識別マッピング

NFS ファイルシステムのエクスポート

RPCSEC-GSS 用のネットワークのセットアップ

NFS ファイルシステムのアンエクスポート

エクスポートされたファイルシステムの変更

エクスポートされたファイルシステムへの root ユーザー・アクセス

NFS ファイルシステムの明示的なマウント

自動マウント・サブシステム

定義済み NFS マウントの確立

定義済み NFS マウントの除去

NFS 用ファイルのエクスポート

mknfsmnt コマンド

ネットワーク・ファイルシステムの認証

NFS は、DES アルゴリズムを使用しますが、その目的はそれぞれ異なります。 NFS は、NFS のサーバーとクライアントとの間で送信されるリモート・プロシージャー・コール (RPC) メッセージのタイム・スタンプを暗号化するために DES を使用します。 この暗号化されたタイム・スタンプは、送信側を認証するトークンとまったく同様にマシンを認証します。

NFS では、NFS のクライアントとサーバーとの間で交換されるすべての RPC メッセージを認証することができるために、各ファイルシステムごとに追加の、オプション・セキュリティー・レベルを提供されます。 デフォルトでは、ファイルシステムは標準的な UNIX 認証を行ってエクスポートされます。ファイルシステムをエクスポートするときに secure オプションを指定すると、この余分のセキュリティー・レベルを利用することができます。

セキュア・ネットワーク・ファイルシステムの公開鍵暗号方式**:**

ユーザーの公開鍵と秘密鍵は、両方とも **publickey.byname** マップ内に保管され、ネット名で索引化されます。

秘密鍵はユーザーのログイン・パスワードと一緒に DES で暗号化されます。 **keylogin** コマンドは、暗号化された秘密鍵を使用し、それをログイン・パスワードで復号してから、セキュア・ローカル鍵サーバーに渡して、将来の RPC トランザクション用に保存します。 **yppasswd** コマンドはログイン・パスワードを変更するだけでなく、公開鍵と秘密鍵を自動的に生成するため、ユーザーには自分の公開鍵も秘密鍵も分かりません。

keyserv デーモンは、各 NIS マシン上で実行される RPC サービスです。 NIS 内では、**keyserv** は次の公開鍵サブルーチンを実行します。

- **key_setsecret** サブルーチン
- **key_encryptsession** サブルーチン
- **key_decryptsession** サブルーチン

key_setsecret サブルーチンは、ユーザーの秘密鍵 (SK_A) を将来の利用のために保管するように鍵サーバーに指示します。このサブルーチンは、通常 **keylogin** コマンドによって呼び出されます。 クライアント・プログラムは **key_encryptsession** サブルーチンをコールして、暗号化された会話鍵を生成し、最初の RPC トランザクション内でサーバーに渡します。 鍵サーバーはサーバーの公開鍵を検索し、それを (以前に **key_setsecret** サブルーチンによってセットアップされた) クライアントの秘密鍵と結合して共通鍵を生成します。 サーバーは **key_decryptsession** サブルーチンをコールして、鍵サーバーに会話鍵を復号するように求めます。

これらのサブルーチン・コールでは、呼び出し側の名前が暗黙に指定されており、それを何らかの方法で認証する必要があります。 鍵サーバーがこのために DES 認証を使用することはできません。なぜなら、そうするとデッドロックが発生する可能性があるからです。 鍵サーバーは秘密鍵をユーザー ID (UID) 別に保管し、ローカル・ルート・プロセスにのみ要求を認可することによって、この問題を解決します。 クライアント・プロセスは、クライアント側から要求を出す root ユーザー所有の **setuid** サブルーチンを実行し、クライアントの実 UID を鍵サーバーに知らせます。

ネットワーク・ファイルシステムの認証要件**:**

セキュア NFS の認証は、送信側が現在時刻を暗号化でき、受信側がそれを復号して自分のクロックに照会して検査できることを基礎としています。

このプロセスには、次の要件があります。

- 両者が現在時刻について合意する必要がある。
- 受信側と送信側で同じ DES 暗号鍵を使用する必要がある。

現在時刻に関する合意*:*

ネットワークで時刻の同期化機能を使用している場合は、timed デーモンによりクライアントとサーバーのクロックの同期が維持されます。 時間の同期化機能を使用していない場合は、クライアントはサーバーのクロックに基づいて正しいタイム・スタンプを計算します。

そのため、クライアントは RPC セッションを開始する前にサーバーの時刻を判別し、クライアントのクロックとサーバーのクロックとの時間差を計算します。 その結果に基づいて、クライアントはそのタイム・スタンプを調整します。 RPC セッションの間にクライアントとサーバーのクロックが同期しなくなり、そのずれがサーバーがクライアントの要求を拒否し始めるポイントにまで達すると、クライアントはサーバー時刻を再決定します。

同じ ***DES*** 鍵の使用*:*

クライアントとサーバーは公開鍵暗号を使用して、同じ DES 暗号化鍵を計算します。

任意のクライアント A とサーバー B について、共通鍵 と呼ばれる鍵は A と B 以外は推測できません。 クライアントは次の式を計算して、共通鍵を求めます。

$$K_{AB} = PK_B{}^{SK}A$$

ここで、*K* は共通鍵、*PK* は公開鍵、*SK* は秘密鍵を表し、これらの鍵はいずれも 128 ビットの数です。 サーバーは次の式を計算して、同じ共通鍵を求めます。

$$K_{AB} = PK_A{}^{SK}B$$

この共通鍵を計算するにはどちらか一方の秘密鍵を知っていなければならないため、これを計算できるのはサーバーとクライアントのみです。 共通鍵は 128 ビットで、DES は 56 ビット鍵を使用するため、クライアントとサーバーは共通鍵から 56 ビットを抽出して DES 鍵を形成します。

ネットワーク・ファイルシステムの認証プロセス*:*

クライアントは、サーバーと対話しようとするとき、タイム・スタンプの暗号化に使用する鍵を無作為に生成します。 この鍵を会話鍵 (*CK*) と呼びます。

クライアントは共通 DES 鍵 (『認証要件』を参照) を使用して会話鍵を暗号化し、最初の RPC トランザクションでサーバーに送信します。このプロセスを以下の図に示します。

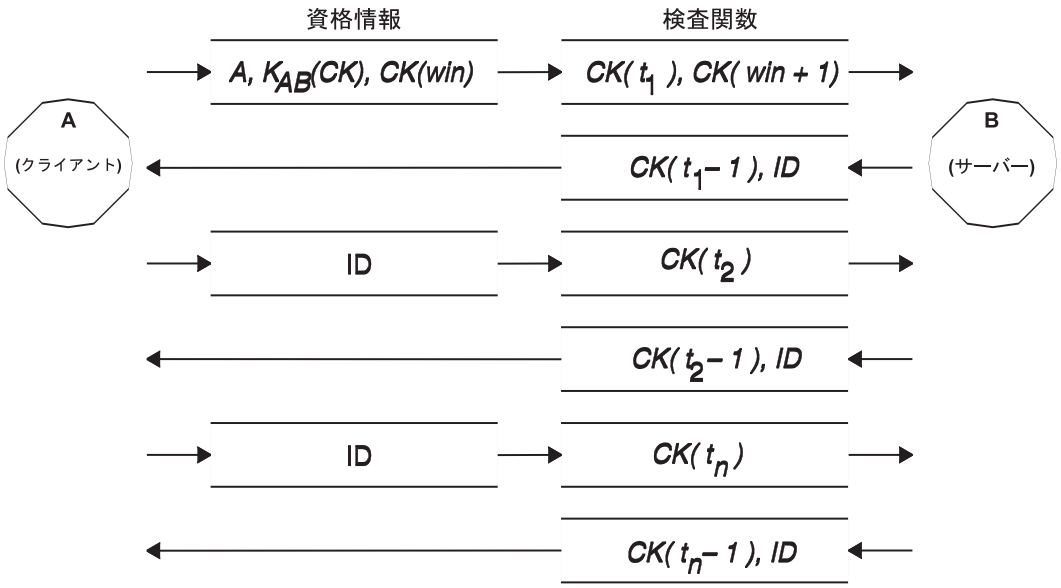


図 *15.* 認証プロセス： この図には、認証プロセスが示されています。

この図は、クライアント A がサーバー B に接続していることを示しています。*K*(*CK*) という表記は、*CK* が DES 共通鍵 *K* で暗号化されることを表します。最初の要求では、クライアントの RPC 資格情報にクライアント名 (*A*)、会話鍵 (*CK*)、および *CK* で暗号化された *win* (ウィンドウ) と呼ばれる変数が入っています。 (デフォルト・ウィンドウ・サイズは 30 分です。) 最初の要求内のクライアントの検査関数には、暗号化されたタイム・スタンプと、指定されたウィンドウの暗号化された検査関数 *win* + 1 が入っています。ウィンドウ検査関数では、正しい資格情報の推測が困難になるため、セキュリティーが強化されます。

クライアントを認証したあとで、サーバーは次の項目を資格情報テーブルに保管します。

- クライアント名 *A*
- 会話鍵 *CK*
- ウィンドウ
- タイム・スタンプ

サーバーは、1 つ前のタイム・スタンプより時系列的にあとになるタイム・スタンプのみを受け入れるため、再生されたトランザクションは確実に拒否されます。サーバーは、資格情報テーブルへの索引 ID と、*CK* で暗号化されたクライアント・タイム・スタンプ・マイナス 1 を、検査関数でクライアントに戻します。 クライアントが送信したタイム・スタンプを知っているのはサーバーのみなので、クライアントはこのような検査関数を送信できたのはサーバーのみであることを認識します。 タイム・スタンプから 1 を減算するのは、クライアント検査関数として再使用できないように無効にするためです。 最初の RPC トランザクションのあとで、クライアントはその ID と暗号化されたタイム・スタンプのみをサーバーに送信し、サーバーは *CK* によって暗号化されたクライアント・タイム・スタンプ・マイナス 1 を返送します。

DES 認証用ネットワーク・エンティティーの命名

DES 認証ではネット名を使用して命名を行います。

ネット名 とは、認証用の印刷可能文字列です。公開鍵と秘密鍵は、ユーザー名別ではなくネット名別に保管されます。 ネット名は、**netid.byname** NIS マップにより、ローカル UID およびグループ・アクセス・リストにマップされます。

ユーザー名は各ドメイン内で固有です。 ネット名は、オペレーティング・システムとユーザー ID を NIS およびインターネットのドメイン名に連結することによって割り当てられます。 ドメインの命名規則でよく使われるのは、ローカル・ドメイン名にインターネット・ドメイン名 (com、edu、gov、mil) を追加することです。

ネットワーク名は、ユーザーのみでなくマシンにも割り当てられます。 マシンのネット名はユーザーのネット名の場合と同様に形成されます。 例えば、`eng.xyz.com` ドメイン内の `hal` という名前のマシンのネット名は `unix.hal@eng.xyz.com` となります。ネットワークを介したホーム・ディレクトリーへの全アクセス権限を必要とするディスクレス・マシンの場合は、マシンを正しく認証することが重要です。

あらゆるリモート・ドメインからのユーザーを認証するために、それらのユーザー用のエントリーを 2 つの NIS データベース内に作成します。 一方は公開鍵と秘密鍵に関するエントリーであり、もう一方はローカル UID とグループ・アクセス・リストのマッピングに関するエントリーです。 これにより、リモート・ドメイン内のユーザーは、 NFS やリモート・ログインなど、すべてのローカル・ネットワーク・サービスにアクセスできます。

/etc/publickey ファイル

`/etc/publickey` ファイルには、NIS が publickey マップを作成するために使用する名前と公開鍵が入っています。

publickey マップはセキュア・ネットワーキングに使用されます。 このファイル内の各エントリーは、ネットワーク・ユーザー名 (ユーザー名またはホスト名のどちらかを指す)、それに続くユーザーの公開鍵 (16 進表記)、コロン 1 個、および暗号化されたユーザーの秘密鍵 (同じく 16 進表記) から構成されます。 デフォルトでは、**/etc/publickey** ファイル内のユーザーはユーザー nobody のみです。

/etc/publickey ファイルには暗号鍵が含まれているため、テキスト・エディターを使用してこのファイルを変更しないでください。 `/etc/publickey` ファイルを変更するには、**chkey** または **newkey** コマンドを使用します。

公開鍵システムのブートに関する考慮事項

電源障害の発生後にマシンを再始動すると、保管されていた秘密鍵はすべて失われ、どのプロセスも NFS ファイルシステムのマウントなどのセキュア・ネットワーク・サービスにアクセスできなくなります。 ただし、root ユーザーの秘密鍵を復号するパスワードを入力できれば、ルート・プロセスの継続が可能です。 この問題の解決方法は、root ユーザーの秘密鍵を復号して、鍵サーバーが読むことができるファイルに保管しておくことです。

すべての **setuid** サブルーチン呼び出しが正しく動作するわけではありません。 例えば、所有者 `A` がある **setuid** サブルーチンを呼び出したとします。この所有者 `A` がマシンの開始以後まだそのマシンにログインしていない場合は、このサブルーチンはどのセキュア・ネットワーク・サービスにも `A` としてアクセスすることはできません。しかし、ほとんどの **setuid** サブルーチン呼び出しの所有者は root ユーザーです。 root ユーザーの秘密鍵は起動時に必ず保管されます。

セキュア・ネットワーク・ファイルシステムのパフォーマンスの考慮

セキュア NFS がシステム・パフォーマンスに影響を与えるいくつかの方法があります。

- クライアントとサーバーの両方が共通鍵を計算する必要があります。 共通鍵の計算所要時間は約 1 秒です。 クライアントとサーバーの両方がこの操作を実行する必要があるため、初期 RPC 接続の確立には結果的に約 2 秒かかります。 初期 RPC 接続後は、鍵サーバーが以前の計算結果をキャッシュするため、共通鍵を毎回計算し直す必要はありません。
- 各 RPC トランザクションには次の DES 暗号化操作が必要です。

1. クライアントが要求タイム・スタンプを暗号化する。
2. サーバーがそれを復号する。
3. サーバーが応答タイム・スタンプを暗号化する。
4. クライアントがそれを復号する。

システム・パフォーマンスはセキュア NFS によって低下する可能性があるために、セキュリティー強化の利点とシステム・パフォーマンスの要件を比較考慮して、それぞれどの程度に重視するのかを検討してください。

セキュア・ネットワーク・ファイルシステム・チェックリスト

このチェックリストを使用して、セキュア NFS が正しく機能することを確認してください。

- クライアント上で **-secure** オプションを指定してファイルシステムをマウントするときは、サーバー名を **/etc/hosts** ファイル内のサーバー・ホスト名と一致させる必要があります。ホスト名の解決にネーム・サーバーを使用している場合は、ネーム・サーバーから戻されるホスト情報が **/etc/hosts** ファイル内のエントリーと一致することを認証してください。マシンのネット名は **/etc/hosts** ファイルの 1 次エントリーを基にしており、**publickey** マップ内の鍵はネット名でアクセスされるため、これらの名前が一致しないと結果的に認証エラーとなります。
- セキュア・エクスポートおよびマウントと、非セキュア・エクスポートおよびマウントとを併用しないでください。 併用すると、ファイル・アクセスが正しく判別されないことがあります。 例えば、クライアント・マシンで **-secure** オプションを指定せずにセキュア・ファイルシステムをマウントするか、**-secure** オプションを指定して非セキュア・システムをマウントすると、ユーザーはユーザー自身としてではなく `nobody` としてアクセス権があることになります。 この条件は、NIS にとって未知のユーザーがセキュア・ファイルシステム上でファイルの作成または変更を試みた場合にも発生します。
- NIS では **chkey** コマンドおよび **newkey** コマンドを使用するたびに、必ず新しいマップを伝搬する必要があるため、これらのコマンドはネットワークの負荷が小さいときにのみ使用してください。
- **/etc/keystore** ファイルまたは **/etc/.rootkey** ファイルを削除しないでください。マシンの再インストール、移動、またはアップグレードを行う場合は、**/etc/keystore** ファイルと **/etc/.rootkey** ファイルを保存してください。
- パスワードの変更には、**yppasswd** コマンドを使用し、**passwd** コマンドは使用しないように、ユーザーに指示してください。 これにより、パスワードと秘密鍵の同期が維持されます。
- **login** コマンドは **keyserv** デーモンの publickey マップから鍵を取り出さないので、ユーザーは **keylogin** コマンドを実行する必要があります。 各ユーザーの `profile` ファイルに **keylogin** コマンドを記述すると、ログイン時にこのコマンドを自動的に実行できます。 ただし、**keylogin** コマンドを使用する場合はユーザーが自分のパスワードを再入力する必要があります。
- コマンド **newkey -h** または **chkey** のいずれかを使用して各ホスト上で root ユーザー用の鍵を生成する場合は、**keylogin** コマンドを実行してそれらの新しい鍵を **keyserv** デーモンに渡す必要があります。 鍵は **/etc/.rootkey** ファイルに保管され、**keyserv** デーモンが始動するたびにこのファイルが読み取られます。
- NIS のマスター・サーバー上で **yppasswdd** デーモンと **ypupdated** デーモンが動作しているかどうかを定期的に検査してください。 この 2 つのデーモンは publickey マップを保守するのに必要です。
- セキュア NFS を使用するすべてのマシン上で **keyserv** デーモンが動作しているかどうかを定期的に検査してください。

セキュア・ネットワーク・ファイルシステムの構成

NIS のマスター・サーバーとスレーブ・サーバーでセキュア NFS を構成するには、次の手順を実行します。

1. NIS マスター・サーバー上で、次のように **newkey** コマンドを使用して、ユーザーごとに NIS /etc/publickey ファイルにエントリーを作成します。
   - 通常のユーザーの場合は、次のように入力します。

     ```
     smit newkey
     ```

     または

     ```
     newkey -u username
     ```

     ホスト・マシン上の root ユーザーの場合は、次のように入力します。

     ```
     newkey -h hostname
     ```
   - 別の方法として、**chkey** コマンドまたは **newkey** コマンドを使用して、ユーザー自身の公開鍵を設定することもできます。
2. NIS publickey マップを作成します。これに対応する NIS **publickey.byname** マップは、NIS サーバー上にのみ存在します。
3. **/etc/rc.nfs** ファイル内の次のスタンザをアンコメントします。

   ```
   #if [ -x /usr/sbin/keyserv ]; then
   #  startsrc -s keyserv
   #fi
   #if [ -x /usr/lib/netsvc/yp/rpc.ypupdated -a -d /etc/yp/`domainname` ]; then
   #  startsrc -s ypupdated
   #fi
   #DIR=/etc/passwd
   #if [ -x /usr/lib/netsvc/yp/rpc.yppasswdd -a -f $DIR/passwd ]; then
   #  startsrc -s yppasswdd
   #fi
   ```
4. **startsrc** コマンドを使用して、**keyserv**、**ypupdated**、および **yppasswdd** デーモンを始動します。

NIS クライアント上にセキュア NFS を構成するには、**startsrc** コマンドを使用して **keyserv** デーモンを始動します。

セキュア・ネットワーク・ファイルシステムを使用したファイルシステムのエクスポート

セキュア NFS をエクスポートするには、次の手順のいずれかを使用します。

- SMIT を使用してセキュア NFS ファイルシステムをエクスポートするには、次のステップを実行します。
  1. **lssrc -g nfs** コマンドを実行して、NFS が既に稼働中であることを確認します。 出力は nfsd および rpc.mountd デーモンがアクティブであることを示します。
  2. publickey マップが存在していること、および keyserv デーモンが稼働中であることを確認します。 詳しくは、 315 ページの『セキュア・ネットワーク・ファイルシステムの構成』を参照してください。
  3. **smit mknfsexp** 高速パスを実行します。
  4. 「PATHNAME of directory to export (エクスポート対象のディレクトリーの PATHNAME)」、「MODE to export directory (ディレクトリーをエクスポートするモード)」、および「EXPORT directory now, system restart or both (すぐにディレクトリーを EXPORT するか、システムを再起動する、あるいはその両方)」の各フィールドに、該当する値を指定します。「Use SECURE option (セキュア・オプションの使用)」フィールドに「yes (はい)」を指定します。
  5. 他の任意のオプション特性を指定するか、またはデフォルト値を受け入れます。
  6. SMIT を終了します。 **/etc/exports** ファイルが存在しない場合は、作成されます。

7. エクスポートしたいそれぞれのディレクトリーごとにステップ 3 から 6 を繰り返します。

- テキスト・エディターを使用してセキュア NFS ファイルシステムをエクスポートするには、次のステップを実行します。
  1. 任意のテキスト・エディターを使用して **/etc/exports** ファイルを開きます。
  2. ディレクトリーの絶対パス名を使用して、エクスポートしたいディレクトリーごとに 1 つのエントリーを作成します。 エクスポートしたいディレクトリーを、それぞれ左端から表示します。 ディレクトリーの中にはエクスポート済みの他のディレクトリーを入れないでください。 /etc/exports ファイルのエントリーの完全な構文、および secure オプションを指定する方法については、/etc/exports ファイルの資料を参照してください。
  3. **/etc/exports** ファイルを保存して閉じます。
  4. NFS が現在稼働中の場合は、次のように入力します。

     ```
     /usr/sbin/exportfs -a
     ```

     exportfs コマンドに **-a** オプションを指定して実行すると、/etc/exports ファイル内のすべての情報がカーネルに送信されます。
- NFS ファイルシステムを一時的に (つまり **/etc/exports** ファイルを変更せずに) エクスポートするには、次のように入力します。

  ```
  exportfs -i -o secure /dirname
  ```

  ここで、dirname はエクスポートするファイルシステムの名前です。 **exportfs -i** コマンドを実行すると、指定したディレクトリーは **/etc/exports** ファイル内で検査されず、すべてのオプションがコマンド・ラインから直接取り出されます。

セキュア・ネットワーク・ファイルシステムを使用したファイルシステムのマウント

セキュア NFS ディレクトリーを明示的にマウントすることができます。

セキュア NFS ディレクトリーを明示的にマウントするには、次のステップを実行します。

1. 次のようにコマンドを実行して、NFS サーバーがディレクトリーをエクスポートしたことを確認します。

   ```
   showmount -e ServerName
   ```

   ここで、ServerName は NFS サーバーの名前です。 このコマンドにより、NFS サーバーから現在エクスポートされているディレクトリーの名前が表示されます。 マウントしたいディレクトリーが表示されていない場合は、サーバーからそのディレクトリーをエクスポートします。
2. **mkdir** コマンドを使用して、ローカル・マウント・ポイントを確立します。 NFS がマウントを正常に完了するためには、NFS マウントのマウント・ポイント (プレースホルダー) として機能するディレクトリーが存在する必要があります。 このディレクトリーは空でなければなりません。 このマウント・ポイントは他のディレクトリーとまったく同じように作成でき、特殊な属性は不要です。
3. publickey マップが存在していること、および keyserv デーモンが稼働中であることを確認します。 詳しくは、 315 ページの『セキュア・ネットワーク・ファイルシステムの構成』を参照してください。
4. 次のように入力します。

   ```
   mount -o secure ServerName:/remote/directory /local/directory
   ```

   ここで、ServerName は NFS サーバーの名前、/remote/directory はマウントしたい NFS サーバー上のディレクトリー、/local/directory は NFS クライアント上のマウント・ポイントです。

注: セキュア NFS をマウントできるのは root ユーザーのみです。

エンタープライズ識別マッピング

今日のネットワーク環境は、システムおよびアプリケーションの複合的なグループで構成されているので、複数のユーザー・レジストリーを管理する必要があります。複数のユーザー・レジストリーを扱うことは、ユーザー、管理者、およびアプリケーション開発者に影響を与える大規模な管理問題にすぐに発展します。エンタープライズ識別マッピング (EIM) を使用することにより、管理者およびアプリケーション開発者はこの問題に取り組むことができます。

このセクションでは、これらの問題を説明し、現在の業界アプローチについて概説し、EIM アプローチについて解説します。

複数のユーザー・レジストリーの管理

多数の管理者は、さまざまなユーザー・レジストリーを介してそれぞれ固有の方法でユーザーを管理する、異なるシステムおよびサーバーを含むネットワークを管理しています。

これらの複合ネットワークでは、管理者は、複数のシステムにまたがるそれぞれのユーザーの ID およびパスワードを管理する責任があります。 さらに、管理者は、多くの場合、これらの ID およびパスワードを同期化しなければなりません。 複数の ID およびパスワードを記憶したり、それらの同期を保つことは、ユーザーに負荷がかかります。 この環境におけるユーザーおよび管理者のオーバーヘッドには費用がかかるので、管理者は、多くの場合、企業の管理よりも、失敗したログイン試行のトラブルシューティングや、忘れられたパスワードのリセットに貴重な時間を費やします。

複数のユーザー・レジストリーの管理の問題は、複数層または異種のアプリケーションを提供したいアプリケーション開発者にも影響を与えます。 カスタマーは、それぞれのシステムが独自のユーザー・レジストリーを処理している、多数の異なるタイプのシステムにまたがって重要な業務データを持っています。 その結果、開発者は、それらのアプリケーション用の、プロプラエタリー・ユーザー・レジストリーとそれに関連したセキュリティー・セマンティクスを作成しなければなりません。 これは、アプリケーション開発者に関する問題を解決しますが、ユーザーおよび管理者のオーバーヘッドは増えます。

エンタープライズ識別マッピングへの現在のアプローチ

複数のユーザー・レジストリーの管理に関する問題を解決するための現在のいくつかの業界アプローチを使用することができますが、どのソリューションも未完成のものです。 例えば、LDAP (Lightweight Directory Access Protocol) は、分散ユーザー・レジストリーのソリューションを提供しています。 ただし、LDAP などのソリューションを使用するには、管理者は、さらに別のユーザー・レジストリーおよびセキュリティー・セマンティクスを管理するか、またはそれらのレジストリーを使用するために構築されている既存のアプリケーションを置き換える必要があります。

このタイプのソリューションを使用すると、管理者は、個々のリソースの複数のセキュリティー・メカニズムを管理しなければなりません。それによって、管理オーバーヘッドが増え、機密漏れの可能性も増えます。複数のメカニズムが単一のリソースをサポートするとき、 1 つのメカニズムを介して権限を変更したり、 1 つ以上のその他のメカニズムの権限を変更し忘れる可能性がより高くなります。例えば、ユーザーが 1 つのインターフェースを介して適切にアクセスを拒否されますが、 1 つ以上のその他のインターフェースを介してアクセスを許可されるときに、機密漏れが生じる可能性があります。

この作業を完了した後、管理者は問題が完全に解決されていないことが分かるでしょう。 一般的に各企業は、このタイプのソリューションを実際的なものとするために、現行のユーザー・レジストリーとそれに関連したセキュリティー・セマンティクスを作成することに、あまりにも多額の費用を投資してきました。別

のユーザー・レジストリーとそれに関連したセキュリティー・セマンティクスを作成すれば、アプリケーション・プロバイダーに関する問題は解決されますが、ユーザーまたは管理者に関する問題は解決されません。

別のソリューションは、シングル・サインオン・アプローチを使用することです。 管理者がすべてのユーザー ID およびパスワードを含むファイルを管理できるようにする、いくつかの製品が使用可能です。 ただし、このアプローチには、いくつかの欠点があります。

- このアプローチは、ユーザーが直面する問題のうちの 1 つしか処理しません。 ユーザーは 1 つの ID およびパスワードを提供することにより複数のシステムにサインオンすることができますが、ユーザーは、その他のシステム上のパスワードを持つ必要があったり、これらのパスワードを管理する必要があります。
- このアプローチでは、平文または復号可能なパスワードがこれらのファイル内に保管されているので、機密漏れを起こすことにより新しい問題が発生します。 平文ファイル内、または管理者を含め誰かによって容易にアクセス可能なファイル内に、パスワードを保管しないでください。
- このアプローチでは、異種の、複数層のアプリケーションを提供するサード・パーティー・アプリケーション開発者に関する問題は解決されません。 それらのアプリケーション開発者は、独自のアプリケーション用のプロプラエタリー・ユーザー・レジストリーを提供しなければなりません。

これらの欠点があるにもかかわらず、一部の企業は、複数ユーザー・レジストリー問題を軽減するので、これらのソリューションを使用しています。

エンタープライズ識別マッピングの使用法

EIM アーキテクチャーは、エンタープライズ内の個人またはエンティティー (ファイル・サーバーや印刷サーバーなど) と、エンタープライズ内の個人またはエンティティーを表す多数の ID との間の関係について記述します。 さらに、EIM は、アプリケーションがこれらの関係に関して質問できるようにする API のセットを提供します。

例えば、1 つのユーザー・レジストリー内に個人のユーザー ID が指定されると、その同じ個人を表す別のユーザー・レジストリー内の ID を判別することができます。 ユーザーが 1 つの ID で認証され、その ID を別のユーザー・レジストリー内の適切な ID にマップすることができる場合、ユーザーは、再び認証用の資格情報を提供する必要はありません。 ユーザーは、別のユーザー・レジストリー内でのそのユーザーを表す ID のみを知っている必要があります。 したがって、EIM は、エンタープライズ用の、一般化された ID マッピング機能を提供します。

異なるレジストリー内のユーザーの ID 間でマップする機能には多数の利点があります。 まず、アプリケーションは、認証用の 1 つのレジストリーを使用している一方で、与信用のまったく異なるレジストリーを使用するという柔軟性を持つことができます。 例えば、管理者は、SAP リソースにアクセスするために SAP ID をマップすることができます。

ID マッピングでは、管理者は次のステップを実行する必要があります。

1. エンタープライズ内の人物またはエンティティーを表す EIM ID を作成します。
2. エンタープライズ内の既存のユーザー・レジストリーについて記述する EIM レジストリー定義を作成します。
3. 作成された EIM ID に、それらのレジストリー内のユーザー ID 間の関係を定義します。

既存のレジストリーへのコード変更は必要ありません。 ユーザー・レジストリー内のすべての ID のマッピングは必要ありません。 EIM は 1 対多のマッピングを可能にします (つまり、単一のユーザーを、単一ユーザー・レジストリー内の複数の ID にマッピングすることができます)。 また、EIM は多対 1 のマ

ッピングを可能にします (つまり、単一 ID を共有する複数のユーザーを単一ユーザー・レジストリーにマッピングすることができます。これはサポートされていますが、セキュリティー上、勧められていません)。 管理者は、EIM 内のいずれかのタイプのユーザー・レジストリーを表示することができます。

EIM では、既存のデータを新規リポジトリーにコピーしたり、両方のコピーの同期を保つ必要がありません。 EIM が導入する唯一の新規データは、関係についての情報です。 管理者は、LDAP ディレクトリー内でこのデータを管理します。これにより、1 つの場所でデータを管理して、情報が使用される場所であればどこでもレプリカを作成するといった柔軟性を提供します。

Kerberos

Kerberos は、物理的に不安定なネットワーク上において、プリンシパルの識別を検証する手段を提供するネットワーク認証サービスです。 Kerberos は、ネットワーク・トラフィックがデータのキャプチャー、テスト、および置換に対してぜい弱であるという想定の下で、相互認証、データの保全性、およびプライバシーを提供します。

Kerberos プリンシパルは Kerberos 認証サービスを使用する固有の ID です。 Kerberos はホストのオペレーティング・システムによる認証に頼らないで、基礎をホスト・アドレス上でのトラストに置くか、またはネットワーク上のすべてのホストの物理的セキュリティーを要求することで、ID を検査します。

Kerberos チケットはユーザーを識別する資格情報です。 チケットには、チケット許可チケットとサービス・チケットの 2 つのタイプがあります。 チケット許可チケットは初期 ID 要求のためのものです。 ホスト・システムにロギングするときには、ユーザーはパスワードやトークンのような身元を確認するものが必要となります。 チケット許可チケットを取得すると、それを使用して特定のサービスに対するサービス・チケットを要求できます。 この 2 チケット方式は、Kerberos のトラステッド・サード・パーティーと呼ばれます。 チケット許可チケットは、Kerberos サーバーに対しユーザーを認証し、また、サービス・チケットはサービスに対しユーザーが信頼できることを伝えます。

Kerberos のトラステッド・サード・パーティーまたは仲介は、鍵配布センター (KDC) と呼ばれます。 KDC はすべての Kerberos チケットをクライアントに発行します。

セキュア・リモート・コマンドの概要

以下にセキュア・リモート・コマンドに関する詳細情報を提供します。

注**:**

1. 分散コンピューティング環境 (DCE) バージョン 2.2 より、DCE セキュリティー・サーバーで Kerberos バージョン 5 チケットを戻すことが可能となりました。
2. すべてのセキュア・リモート・コマンド (rcmds) は、Expansion Pack DVD から入手できる IBM ネットワーク認証サービス (NAS) で提供される Kerberos バージョン 5 ライブラリーを使用します。 `krb5.client.rte` ファイルセットをインストールする必要があります。これも Expansion Pack DVD から入手できます。
3. DVD メディアを使用して AIX オペレーティング・システムをマイグレーションする場合で、Kerberos が既にインストールされていれば、Expansion Pack DVD から `krb5.client.rte` をインストールするようにインストール・スクリプトでプロンプトが出されます。
4. NIM リソースを使用して AIX オペレーティング・システムをマイグレーションする場合で、Kerberos が既にインストールされていれば、lpp_source ディレクトリーに `krb5` を追加してください。

セキュア・リモート・コマンド (rcmds) は、**rlogin**、**rcp**、**rsh**、**telnet**、および **ftp**です。これらのコマンドは、まとめて「標準 AIX 認証方式」と呼ばれています。追加の方式は Kerberos で提供されます。

Kerberos バージョン 5 認証方式を使用する場合、クライアントは DCE セキュリティー・サーバーまたは Kerberos サーバーから Kerberos バージョン 5 チケットを取得します。 チケットは、ユーザーの現行 DCE の一部であるか、接続しようとしている TCP/IP サーバーに対して暗号化されたローカル資格情報です。 TCP/IP サーバー上のデーモンがこのチケットを暗号化解除します。 このアクションで TCP/IP サーバーはユーザーを確実に識別できます。 チケットで記述されている DCE またはローカル・プリンシパルが、オペレーティング・システムのユーザー・アカウントへアクセスを許可されていれば、接続は継続します。 secure rcmds は Kerberos クライアントとサーバーを Kerberos バージョン 5 と DCE の両方でサポートします。

クライアントの認証に加え、Kerberos バージョン 5 は、現行ユーザーの資格情報を TCP/IP サーバーに転送します。 資格情報に転送可能のマークが付くと、クライアントは資格情報を Kerberos チケット許可チケット (TGT) としてサーバーに送信します。 TCP/IP サーバー側では、ユーザーが DCE セキュリティー・サーバーと通信している場合、デーモンが **k5dcecreds** コマンドを使用してチケット許可チケットを完全 DCE 資格情報にアップグレードします。

ftp コマンドは、他の secure rcmds とは異なる認証方式を使用します。 つまり、GSSAPI セキュリティー・メカニズムを使用して、**ftp** コマンドと **ftpd** デーモン間で認証を受け渡します。 **clear**、**safe**、および **private** サブコマンドを使用して、ftp クライアントはデータ暗号化をサポートします。

オペレーティング・システムのクライアント/サーバー間では、**ftp** コマンドにより、暗号化されたデータ接続のためのマルチ・バイト転送が可能となります。 標準では、暗号化されたデータ接続に対しては単一バイト転送のみが定義されます。サード・パーティーのマシンに接続してデータ暗号化を使用する場合は、**ftp** コマンドは単一バイト転送の制限に従います。

システム構成**:**

すべての secure rcmds に対して、システム・レベル構成メカニズムにより、システムごとに許可可能な認証方式が決定されます。 構成により着信接続と発信接続の両方が制御されます。

認証構成は、**libauthm.a** ライブラリーと、**lsauthent** コマンドおよび **chauthent** コマンドで構成されます。これらのコマンドは、**get_auth_methods** および **set_auth_methods** ライブラリー・ルーチンへのコマンド・ライン・アクセスを提供します。

この認証方式は、ネットワーク上のユーザーの認証にどの方式を使用するかを定義します。 システムは以下のような認証方式をサポートします。

- Kerberos バージョン 5 は DCE の基本であり、最も一般的な方式です。
- Kerberos バージョン 4 は、rlogin、rsh、および rcp secure rcmds によってのみ使用されます。 これは、SP システムでのみ旧バージョンとの互換性をサポートするために提供されます。 Kerberos バージョン 4 チケットは、DCE 資格情報にはアップグレードされません。

複数の認証方式が構成され、最初の方式が接続に失敗すると、クライアントは次の構成された認証方式を使用して認証を試みます。

認証方式は、どの順序でも構成は可能です。 唯一の例外として、標準 AIX は最後の認証方式として構成する必要があります。これはフォールバック・オプションがないためです。 標準 AIX が構成済み認証方式でない場合、パスワード認証は試みられず、このメソッドを使用した接続の試みはすべて拒否されます。

システムの構成は認証方式なしで行うことも可能です。この場合、システムは secure rcmds コマンドを使用したシステムとの接続をすべて拒否します。 また、Kerberos バージョン 4 は **rlogin**、**rsh**、および **rcp** コマンドのみをサポートするので、Kerberos バージョン 4 のみを使用するように構成したシステムでは、telnet または FTP を使用した接続はできません。

Kerberos バージョン **5** のユーザー検証**:**

Kerberos バージョン 5 認証方式は、ユーザーの検証に使用することができます。

Kerberos バージョン 5 認証方式を使用する場合、TCP/IP クライアントは TCP/IP サーバーに対し暗号化されたサービス・チケットを取得します。 サーバーがチケットを暗号化解除する場合は、サーバーはユーザーを (DCE またはローカル・プリンシパルで) 識別する secure 方式を持っています。 ただし、サーバーは、この DCE またはローカル・プリンシパルがローカル・アカウントへのアクセスを許可されるかどうかについて判別する必要があります。 DCE またはローカル・プリンシパルからローカル・オペレーティング・システム・アカウントへのマッピングは、共有ライブラリー `libvaliduser.a` によって処理されます。このライブラリーには、kvalid_user と呼ばれる単一のサブルーチンがあります。 異なったマッピング方式を優先したい場合、システム管理者は **libvaliduser.a** ライブラリーに代替方式を提供する必要があります。

DCE の構成**:**

secure rcmds を使用するには、接続可能なネットワーク・インターフェースごとに 2 つの DCE プリンシパルが存在している必要があります。

2 つの DCE プリンシパルは次のとおりです。

```
host/FullInterfaceName
ftp/FullInterfaceName
```

ここで、*FullInterfaceName* はインターフェース名およびドメイン名です。

ローカル構成**:**

secure rcmds を使用するには、ネットワークが接続可能なすべてのネットワーク・インターフェースについて 2 つのローカル・プリンシパルが存在している必要があります。

2 つのローカル・プリンシパルは次のとおりです。

```
host/FullInterfaceName@Realmname
ftp/FullInterfaceName@Realmname
```

ここで、*FullInterfaceName* はインターフェース名およびドメイン名であり、 *RealmName* はローカル Kerberos バージョン 5 レルムの名前です。

以下の資料の関連情報を参照してください。

- 「*Technical Reference: Communications, Volume 2*」の『get_auth_method サブルーチン』および『set_auth_method サブルーチン』
- 「*Commands Reference, Volume 1*」の『chauthent コマンド』
- 「*Commands Reference, Volume 3*」の『lsauthent コマンド』

ネットワーク認証サービスまたは非 **AIX** サービスを使用した **AIX** オペレーティング・システムへの認証

AIX 6.1 より前の場合、KRB5 ロード・モジュールはネットワーク認証サービス (NAS) 環境に対する Kerberos 認証を処理し、KRB5A ロード・モジュールは非 AIX システム環境に対する Kerberos 認証を処理していました。 AIX 6.1 以降、KRB5 ロード・モジュールはネットワーク認証サービス (NAS) 環境と非 AIX システム環境の両方の Kerberos 認証を行います。 `etc/security/methods.cfg` ファイル内の **is_kadmind_compat** 属性で、KRB5 環境か KRB5A 環境かを指定します。 AIX 7.1 以降、KRB5A ロー

ド・モジュールは使用不可になっています。 したがって、KRB5 環境か KRB5A 環境かを指定するには、`etc/security/methods.cfg` ファイル内の **is_kadmind_compat** 属性を使用する必要があります。

Kerberos クライアントが NAS に対して認証するように構成されている場合、 KRB5 ロード・モジュールは Kerberos 認証および Kerberos プリンシパル管理を実施します。 このモジュールは、システム管理者が AIX ユーザー管理コマンドを使用して、 Kerberos プリンシパル管理を行うことを可能にします。 プリンシパル管理を使用するには、Kerberos サーバーにより kadmin 管理プロトコルがサポートされていなければなりません。このサポートは **kadmind** デーモン (AIX オペレーティング・システム上で実行される Kerberos サーバー) を使用することで、NAS により提供されます。

注**:** Kerberos クライアントを構成するときに、認証が NAS に対するものであることを指定する必要があります。そのように指定しないと、クライアントは非 AIX サービスに対して認証するように構成され、プリンシパル管理は使用不可になります。

非 AIX システムに対して Kerberos プリンシパルを使用する場合、Kerberos は非 AIX システムに保管され、 kadmin Kerberos データベース・インターフェースの使用による AIX オペレーティング・システムからの管理はできません。 このケースでは、Kerberos プリンシパルの管理ツールを使用して、プリンシパル管理を単独で実行する必要があります。 これらのツールは Kerberos 製品のパーツにするか、または OS (例えば、Windows 2000) に組み込まれている場合があります。 非 AIX システムに対して Kerberos を使用する本来の目的は、Windows 2000 Active Directory サーバーに対する認証を提供することでした (Windows 2000 Active Directory サーバーでは、Active Directory のアカウント管理ツールと API を使用して Kerberos プリンシパル管理が行われます)。しかし、非 AIX システムに対する Kerberos の使用は、Kerberos 管理インターフェースがサポートされない、対応するその他の KDC でも可能です。

IBM NAS を使用した **Kerberos** 統合ログイン用システムのインストールと構成**:**

ネットワーク認証サービス (NAS) の IBM Kerberos インプリメンテーションは、拡張パックに同梱されて出荷されます。

Kerberos バージョン 5 サーバー・パッケージをインストールするには、次のコマンドを実行して `krb5.server.rte` ファイルセットをインストールします。

```
installp –aqXYgd . krb5.server
```

Kerberos サーバーとして構成済みのマシンが Kerberos クライアントとしても使用される場合は、Kerberos KRB5 パッケージを全部インストールします。

また、DCE には Kerberos ユーティリティーと同じ名前の Kerberos クライアント・ユーティリティーのセットがあります。 DCE と Kerberos コマンド間 (つまり、**klist** コマンド、**kinit** コマンド、および **kdestroy** コマンド間) でネームスペースが衝突するのを避けるためには、これらの Kerberos コマンドを **/usr/krb5/bin** ディレクトリーおよび **/usr/krb5/sbin** ディレクトリーにインストールします。

Kerberos コマンドを実行するには、Kerberos ディレクトリーを PATH 定義に追加しない場合は、次のように完全修飾コマンドのパス名を指定する必要があります。

```
export PATH=$PATH:/usr/krb5/sbin:/usr/krb5/bin
```

注**:** Java14 SDK は **kinit** コマンドもインストールします。この場合、PATH 環境変数には、このコマンドを他の **kinit** コマンドの前に指定したほうがよいでしょう。 Java14 **kinit** プログラムに代えて、ネットワーク認証サービス・コマンドが必要になる場合は、 Java14 **kinit** プログラムを PATH 定義の別の場所に移動します。

ネットワーク認証サービスの資料は、 krb5.doc.*lang*.pdf|html パッケージに入っています。ここで、*lang* は、サポートされる言語を表します。

AIX オペレーティング・システムには、複合ロード・モジュールを形成するために使用可能な 2 つのデータベース・モジュール (LDAP と BUILTIN) があります。 LDAP モジュールは LDAP レジストリー (ディレクトリー) に保管された情報をアクセスする場合に使用され、 BUILTIN モジュールはファイルのレジストリー (ローカル・ファイルシステム) に保管された情報をアクセスする場合に使用されます。 作成される複合ロード・モジュールには、通常、KRB5files または KRB5LDAP という名前が付けられます。 これらの名前は、KRB5 が認証およびローカル・ファイルか、または LDAP 用として使用されることを示しています。

また、ネットワーク認証サービスは、Kerberos 情報をローカル・ファイルシステム (Kerberos レガシー・データベース) か、または LDAP のいずれかへの保管もサポートします。可能な構成としては、次の 4 つの構成があります。

- Kerberos サーバーの情報が Kerberos レガシー・データベースに保管された KRB5files
- Kerberos サーバーの情報が Kerberos LDAP データベースに保管された KRB5files
- Kerberos サーバーの情報が Kerberos レガシー・データベースに保管された KRB5LDAP
- Kerberos サーバーの情報が Kerberos LDAP データベースに保管された KRB5LDAP

LDAP が Kerberos プリンシパルまたは AIX ユーザーおよびグループ情報を保管するためのストレージ・メカニズムである場合は、LDAP を構成してから Kerberos 構成コマンドを起動します。 LDAP の構成後に **mkkrb5srv** コマンドを使用して Kerberos サーバーを構成します。

レガシー・データベース・ストレージを使用したネットワーク認証サービス・サーバーの構成*:*

ネットワーク認証サービス KDC および管理サーバーはレガシー Kerberos データベースを指定してセットアップでき、ネットワーク認証サービス・サーバーは **mkkrb5srv** コマンドを使用して構成できます。

mkkrb5srv コマンドの使用についての詳細は、「**mkkrb5srv** コマンド」を参照してください。

注**:** 同じ物理システムに DCE と Kerberos サーバー・ソフトウェアの両方をインストールしないでください。 そうしなければならない場合は、DCE クライアント/サーバーまたは Kerberos クライアント/サーバーのデフォルトの操作可能インターネット・ポート番号を変更する必要があります。 どちらのケースでも、そのような変更は、ご使用の環境内に存在する DCE と Kerberos の配置によってはインターオペラビリティーに影響があります。 DCE と Kerberos の共存については、ネットワーク認証サービスの資料を参照してください。

Kerberos バージョン 5 は、ホストのクロックが指定した KDC の最大クロック・スキュー内に無いすべてのホストからのチケット要求をリジェクトするよう設定されます。 最大クロック・スキューのデフォルト値は、300 秒 (5 分) です。 Kerberos では、サーバー/クライアント間で時刻の同期化の一定の形式が構成されることが要求されます。 時刻の同期化には、**xntpd** デーモンまたは **timed** デーモンを使用することをお勧めします。 **timed** デーモンを使用するには、次のようにします。

1. 次のように、**timed** デーモンを開始して、KDC サーバーをタイム・サーバーとしてセットアップします。

   ```
   timed -M
   ```

2. 次のように、各 Kerberos クライアント上で **timed** デーモンを開始します。

   ```
   timed -t
   ```

3. Kerberos KDC および kadmin サーバーを構成するには、**mkkrb5srv** コマンドを実行します。 例えば、MYREALM レルム、sundial サーバー、および xyz.com ドメイン用に Kerberos を構成するには、次のコマンドを実行します。

   ```
   mkkrb5srv -r MYREALM -s sundial.xyz.com -d xyz.com -a admin/admin
   ```

   kadmind コマンドおよび **krb5kdc** コマンドが /etc/inittab ファイルから開始されるまで数分間待ちます。

ネットワーク認証サービスは **/var** ファイルシステムのスペースを使用して情報を保管します。 この情報には認証済みユーザーのデータベース、ログ、および資格情報キャッシュ・ファイルが含まれます。 これらのファイルのサイズは時間が経つにつれて増加します。 この情報を保持するために、フリー・スペースを定期的にモニターして、 **/var** ファイルシステムに十分なフリー・スペースがあることを確認してください。

代表的な **mkkrb5srv** コマンドの例を以下に示します。

```
mkkrb5srv –r Realm_Name –s KDC_Server –d Domain_Name –a Admin_Name
```

表 16 の変数値は、レガシー・データベースを使用してネットワーク認証サービス・サーバーを構成する方法を示す使用例です。

表 *16.* **mkkrb5srv** コマンドの変数名

変数名	変数値
レルム名	MYREALM
KDC サーバー	kdcsrv.austin.ibm.com
ドメイン名	austin.ibm.com
管理者名	admin/admin

既存の Kerberos サーバー構成がある場合は、**mkkrb5srv –U** または **unconfig.krb5** コマンドを使用して、これを除去できます。

重要**:** 既存の Kerberos サーバー構成を保持する必要がある場合は、以下のステップを実行しないでください。

以下の手順は、レガシー・データベースを使用してネットワーク認証サービス・サーバーを構成する方法を示す例です。

1. 次のコマンドを入力します。

   ```
   mkkrb5srv -r MYREALM -s kdcsrv.austin.ibm.com -d austin.ibm.com -a admin/admin
   ```

   このコマンドの入力後に、マスター・データベース・パスワードのプロンプトが出されます。

   ネットワーク認証サービスでは、KDC と管理サーバーが別々のホストにある構成をサポートしないため、 KDC と管理サーバーの両方には、ローカル・ホストが使用されます。 次のエラー・メッセージが表示された場合は、無視してください。 `The -s option is not supported.`
2. プロンプトに従って、マスター・データベース・パスワードを入力します。
3. プロンプトに従って、管理プリンシパル・パスワードを入力します。

   管理プリンシパル・パスワードを入力し、**mkkrb5srv** コマンドを使用して **kadmind** デーモンおよび **krb5kdc** デーモンを /etc/inittab ファイル・パスから開始させます。 このプロセスは数分かかるかもしれません。

4. 次のコマンドを実行して、/etc/inittab ファイルのエントリーを検査します。

```
lsitab krb5kdc
lsitab kadm
```

5. 次のコマンドを入力して、KDC および kadmind サーバーが始動されたことを検査します。

```
ps -ef | grep -v grep | grep krb5
```

mkkrb5srv コマンドを使用して、Kerberos レルム (MYREALM) 用のマスター KDC および kadmind 管理サーバーを作成します。 このコマンドは構成ファイルの作成、基本データベースの初期化、および KDC サーバーと kadmind サーバーの始動も行います。

mkkrb5srv コマンドを実行すると、次のアクションが実行されます。

1. **/etc/krb5/krb5.conf** ファイルが作成される。 レルム名、Kerberos 管理サーバー、およびドメイン名の値が、コマンド・ラインで指定されたとおりに設定される。 **/etc/krb5/krb5.conf** ファイルにも、default_keytab_name、kdc、および admin_server ログ・ファイルのパスが設定される。
2. **/var/krb5/krb5kdc/kdc.conf** ファイルが作成される。 /var/krb5/krb5kdc/kdc.conf ファイルに、*kdc_ports*、*kadmin_port*、*max_life*、*max_renewable_life*、*master_key_type*、および *supported_enctypes* 変数の値が設定される。 さらに、このファイルで、*database_name*、*admin_keytab*、*acl_file*、*dict_file*、および *key_stash_file* 変数のパスも設定される。
3. **/var/krb5/krb5kdc/kadm5.acl** ファイルが作成される。 admin、root、および host プリンシパルのアクセス制御がセットアップされる。
4. データベースと、1 つの admin プリンシパルが設定される。 Kerberos のマスター・キーと名前を要求されるので、Kerberos 管理プリンシパル ID のパスワードを設定する。 災害復旧対策として、マスター・キー、管理プリンシパル ID、パスワードを安全に保管することが重要です。

詳しくは、 330 ページの『実行のサンプル』および 329 ページの『エラー・メッセージとリカバリー・アクション』を参照してください。

LDAP ストレージを使用する ***Kerberos*** サーバーの構成***:***

ネットワーク認証サービスの kadmin と Kerberos 統合ログインのための KDC サーバーは、**mkkrb5srv** コマンドを使用してセットアップすることができます。

表 17 の変数値は、 **mkkrb5srv** コマンドを使用して、ネットワーク認証サービス・サーバーの構成要素を LDAP ストレージで構成する方法を示す使用例です。

表 *17.* **mkrb5srv** コマンドの変数名

変数名	変数値
Realm_Name	MYREALM
KDC_Server	kdcsrv.austin.ibm.com
Domain_Name	austin.ibm.com
Admin_Name	admin/admin
LDAP サーバー	kdcsrv.austin.ibm.com
LDAP 管理者名	cn=root
LDAP 管理者パスワード	secret

以下の手順は、**mkkrb5srv** コマンドを使用して、ネットワーク認証サービス・サーバーの構成要素を LDAP ストレージで構成する方法を示す例です。

1. 次のコマンドを実行します。

```
mkkrb5srv -r MYREALM -s kdcsrv.austin.ibm.com -d austin.ibm.com¥
-a admin/admin -l kdcsrv.austin.ibm.com -u cn=root -p secret
```

2. KDC および kadmind サーバーが次のコマンドの実行によって始動されたことを検査します。

```
ps -ef | grep -v grep | grep krb5
```

LDAP の出力を指定して **mkkrb5srv** コマンド実行した結果は、レガシー・データベース構成を指定してコマンドを実行した結果と類似したものになります。 ただし、LDAP を使用する場合は、ローカル・ファイルシステムにデータベースが作成されません。 代わりに、LDAP に関する情報を保持するための `.kdc_ldap_data` ファイルが `/var/krb5/krb5kdc` ファイルに作成されます。

使用法に関する追加情報については、「**mkkrb5srv** コマンド」を参照してください。

Kerberos 統合ログインの構成*:*

Kerberos インストールの完了後は、ユーザー認証の 1 次手段として Kerberos を使用するようにシステムを構成する必要があります。

ユーザー認証の 1 次手段として Kerberos を使用するようにシステムを構成するには、次のパラメーターを使用して **mkkrb5clnt** コマンドを実行します。

```
mkkrb5clnt -c KDC -r realm -a admin -s server -d domain -A -i database -K -T
```

表 18 の変数値は、 AIX ユーザー/グループ・リポジトリーとしてローカル・ファイルシステムを使用して、 Kerberos 統合ログインのためのシステムを構成する方法を示す使用例です。

表 *18.* **mkkrb5clnt** コマンドの変数名

変数名	変数値
レルム名	MYREALM
KDC サーバー	kdcsrv.austin.ibm.com
ドメイン名	austin.ibm.com
管理サーバー	kdcsrv.austin.ibm.com
管理者名	admin/admin
AIX ユーザー/グループ・データベース	files

次のコマンドは AIX ユーザー/グループ・リポジトリーとしてローカル・ファイルシステムを使用して、Kerberos 統合ログインのためのシステムを構成する方法を示す例です。

次のようなコマンドを実行します。

```
mkkrb5clnt -r MYREALM -c kdcsrv.austin.ibm.com -s kdcsrv.austin.ibm.com¥
-a admin/admin -d austin.ibm.com -A -i files -K -T
```

前の例の結果、次のアクションが行われます。

1. このコマンドは `/etc/krb5/krb5.conf` ファイルを作成します。レルム名、Kerberos 管理サーバー、およびドメイン名の値はコマンド・ラインで指定されたものが設定されます。 `default_keytab_name`、`kdc`、および `kadmin` ログ・ファイルのパスも更新されます。
2. **-i** フラグは完全な統合ログインを構成します。 入力するデータベースは、AIX ユーザー識別情報が保管されている場所です。 これは、Kerberos プリンシパルのストレージとは異なります。 Kerberos プリンシパルが保管されているストレージは、Kerberos の構成時に設定されます。
3. **-K** フラグはデフォルトの認証方式として Kerberos を構成する。 これにより、ユーザーはログイン時に Kerberos での認証が可能となります。

4. **-A** フラグにより Kerberos データベースにエントリーが追加され、root が Kerberos の管理ユーザーとなる。
5. **-T** フラグによりサーバー管理のチケット許可チケットが取得される。

注**:** IBM ネットワーク認証サービス (NAS) に対する認証用の Kerberos クライアント環境を構成するには、**mkkrb5clnt** コマンドに **-D** オプションを使用しないでください。**mkkrb5clnt** コマンドに **-D** オプションを指定しなければ、**is_kadmind_compat**属性は /usr/lib/security/methods.cfg ファイルに組み込まれず、Kerberos クライアント環境は IBM NAS に対する認証用に構成されます。

構成は /etc/krb5/krb5.conf ファイルを検査して検証します。 以下に示すのは、クライアント・マシン上の /etc/krb5/krb5.conf ファイルの例です。

```
[libdefaults]
     default_realm = MYREALM
     default_keytab_name = FILE:/etc/krb5/krb5.keytab
     default_tkt_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts des-cbc-md5 des-cbc-crc
     default_tgs_enctypes = des3-cbc-sha1 arcfour-hmac aes256-cts des-cbc-md5 des-cbc-crc
[realms]
     MYREALM = {
        kdc = kdcsrv.austin.ibm.com:88
        admin_server = kdcsrv.austin.ibm.com:749
        default_domain = austin.ibm.com
     }
[domain_realm]
    .austin.ibm.com = MYREALM
    kdcsrv.austin.ibm.com = MYREALM
[logging]
     kdc = FILE:/var/krb5/log/krb5kdc.log
     admin_server = FILE:/var/krb5/log/kadmin.log
     default = FILE:/var/krb5/log/krb5lib.log
```

注**:** Kerberos プリンシパル・ストレージに LDAP を使用する場合、krb5.conf ファイルには [realms] stanza: の下に以下の行を追加します。

```
vdb_plugin_lib = /usr/lib/libkrb5ldplug.a
```

システムが KDC とは別の DNS ドメイン内にインストールされる場合は、次のような追加のアクションの実行が必要です。

1. **/etc/krb5/krb5.conf** ファイルを編集して、[domain realm] の後に別のドメインを追加する。
2. その別のドメインをレルムにマップする。

例えば、abc.xyz.com ドメインにあるクライアントを自分の MYREALM レルムに含めたい場合は、以下のように /etc/krb5/krb5.conf ファイルを変更します。

```
[domain realm]
     .austin.ibm.com = MYREALM
     .raleigh.ibm.com = MYREALM
```

ネットワーク認証サービスの構成が終了するときに、オペレーティング・システムへのログイン・プロセスは変更されないでそのまま残ります。 正常なログイン後は、Kerberos チケット許可チケットがユーザーの実行プロセスに関連付けられます。 ユーザーの $KRB5CCNAME 環境変数は、そのチケット許可チケットを指します。 ログインが正常に行われたこと、およびユーザーがチケット許可チケットを所有していることを検査するには、 **klist** コマンドを使用します。

注**: mkkrb5clnt** コマンドを実行すると、次のスタンザが methods.cfg ファイルに追加されます。

```
KRB5:
        program = /usr/lib/security/KRB5
        program_64 = /usr/lib/security/KRB5_64
```

```
        options = is_kadmind_compat=yes

KRB5files:
        options = db=BUILTIN,auth=KRB5
```

追加情報の参照先:

- **mkkrb5clnt** コマンドについては、**mkkrb5clnt** コマンドを参照してください。
- methods.cfg ファイルについては、methods.cfg ファイルを参照してください。

エラー・メッセージとリカバリー・アクション*:*

mkkrb5srv コマンドの使用中に起こるエラーには、次のものがあります。

- **krb5.conf**、**kdc.conf**、または **kadm5.acl** ファイルが既に存在していると、**mkkrb5srv** コマンドは値を変更しません。 「ファイルは既に存在しています」というメッセージを受け取ります。 構成値は、**krb5.conf**、**kdc.conf**、または **kadm5.acl** ファイルを編集して、いずれも変更することができます。
- 何らかの入力を間違え、データベースが作成されなかった場合は、作成した構成ファイルを除去し、コマンドを再実行します。
- データベースと構成値に不整合がある場合は、**/var/krb5/krb5kdc/*** ディレクトリーからそのデータベースを除去してから、コマンドを再実行します。
- **kadmind** デーモンと **krb5kdc** デーモンがマシン上で開始済みであることを確認します。 **ps** コマンドを使用して、デーモンが実行中であることを確認します。 デーモンが開始されていなければログ・ファイルをチェックします。

mkkrb5clnt コマンドの使用中に起こるエラーには、次のものがあります。

- **krb5.conf** の値の間違いは、**/etc/krb5/krb5.conf** ファイルを編集して修正できます。
- **-i** フラグの値の間違いは、**/usr/lib/security/methods.cfg** ファイルを編集して修正できます。

非 ***KRB5*** 認証時の ***kadmind*** デーモンへの依存関係の除去*:* kadmind デーモンが使用不可のときに、シングル・サインオン (SSO) などの非 KRB5 認証メカニズムを使用していると、KRB5 ロード・モジュールが原因で遅延が生じます。 この依存関係は、**methods.cfg** ファイルに kadmind_timeout パラメーターを設定することで解除されます。

指定可能な値は kadmind_timeout=<seconds> であり、ここで seconds は 0 より大きい値でなくてはなりません。

KRB5 ロード・モジュールが、ダウンしている kadmind サーバーに接続しようとすると、伝送制御プロトコル (TCP) のタイムアウトが生じます。 kadmind_timeout パラメーターは初期 TCP タイムアウトの後のさらなる遅延を防ぎます。 kadmind_timeout パラメーターは KRB5 ロード・モジュールが初期 TCP タイムアウトした後に kadmind 接続をもう 1 回試行する時間枠を指定します。 kadmind サーバーが実行されている場合、デフォルトの動作は引き続き有効です。

デフォルトでは kadmind_timeout は使用不可です。 kadmind_timeout パラメーターを使用可能にするには、methods.cfg ファイルを次のように変更します。

```
KRB5:
        program = /usr/lib/security/KRB5
        options = kadmind_timeout=300
KRB5files:
        options = db=BUILTIN,auth=KRB5
```

作成されるファイル:

mkkrb5srv コマンドは次のファイルを作成します。

- **/etc/krb5/krb5.conf**
- **/var/krb5/krb5kdc/kadm5.acl**
- **/var/krb5/krb5kdc/kdc.conf**

mkkrb5clnt コマンドは次のファイルを作成します。

- **/etc/krb5/krb5.conf**

mkkrb5clnt -i *files* オプションは、**/usr/lib/security/methods.cfg** ファイルに次のスタンザを追加します。

```
KRB5:
  program =
  options =
KRB5files:
  options =
```

実行のサンプル:

このセクションでは、サンプル実行からの例を提供します。

次の示すものは **mkkrb5srv** コマンドの一例です。

```
# mkkrb5srv -r MYREALM -s sundial.xyz.com -d xyz.com -a admin/admin
```

次のような出力が表示されます。

```
  Fileset                      Level  State      Description
  ----------------------------------------------------------------------------
Path: /usr/lib/objrepos
  krb5.server.rte            1.3.0.0  COMMITTED  Network Authentication Service
                                                 Server

Path: /etc/objrepos
  krb5.server.rte            1.3.0.0  COMMITTED  Network Authentication Service
                                                 Server

The -s option is not supported.
The administration server will be the local host.
Initializing configuration...
Creating /etc/krb5/krb5.conf...
Creating /var/krb5/krb5kdc/kdc.conf...
Creating database files...
Initializing database '/var/krb5/krb5kdc/principal' for realm 'MYREALM'
master key name 'K/M@MYREALM'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter database Master Password:
Re-enter database Master Password to verify:
WARNING: no policy specified for admin/admin@MYREALM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Enter password for principal "admin/admin@MYREALM":
Re-enter password for principal "admin/admin@MYREALM":
Principal "admin/admin@MYREALM" created.
Creating keytable...
Creating /var/krb5/krb5kdc/kadm5.acl...
Starting krb5kdc...
krb5kdc was started successfully.
```

```
Starting kadmind...
kadmind was started successfully.
The command completed successfully.
Restarting kadmind and krb5kdc
```

mkkrb5clnt コマンドの例を以下に示します。

```
mkkrb5clnt -r MYREALM -c sundial.xyz.com -s sundial.xyz.com ¥
           -a admin/admin -d xyz.com -i files -K -T -A
```

次のような出力が表示されます。

```
Initializing configuration...
Creating /etc/krb5/krb5.conf...
The command completed successfully.
Password for admin/admin@MYREALM:
Configuring fully integrated login
Authenticating as principal admin/admin with existing credentials.
WARNING: no policy specified for host/diana.xyz.com@MYREALM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Principal "host/diana.xyz.com@MYREALM" created.

Administration credentials NOT DESTROYED.
Authenticating as principal admin/admin with existing credentials.

Administration credentials NOT DESTROYED.
Authenticating as principal admin/admin with existing credentials.
Principal "kadmin/admin@MYREALM" modified.

Administration credentials NOT DESTROYED.
Configuring Kerberos as the default authentication scheme
Making root a Kerberos administrator
Authenticating as principal admin/admin with existing credentials.
WARNING: no policy specified for root/diana.xyz.com@MYREALM;
  defaulting to no policy. Note that policy may be overridden by
  ACL restrictions.
Enter password for principal "root/diana.xyz.com@MYREALM":
Re-enter password for principal "root/diana.xyz.com@MYREALM":
Principal "root/diana.xyz.com@MYREALM" created.

Administration credentials NOT DESTROYED.
Cleaning administrator credentials and exiting.
```

認証時の **kadmind** デーモンへの依存関係の除去:

kadmind デーモンが使用不可の場合、KRB5 ロード・モジュールは認証に失敗することがあります。 この依存関係は methods.cfgファイルに *kadmind* パラメーターを設定して解除できます。

可能な値は、**kadmind** ルックアップを使用不可にする場合は kadmind=no または kadmind=false、**kadmind** ルックアップを使用可能にする場合は kadmind=yes または kadmind=true です (デフォルト値は yes です)。 このオプションを no に設定すると、認証時に **kadmind** デーモンに接続されません。 したがって、システムがプロンプトを出したときに、ユーザーが正しいパスワードを入力すれば、 **kadmind** デーモンの状況に関係なく、ユーザーはシステムにログインできます。 ただし、このデーモンが使用不可の場合 (例えば、このデーモンがダウンしているときやマシンにアクセスできないとき) は、**mkuser**、**chuser**、または **rmuser** などの AIX ユーザー管理コマンドは Kerberos 統合ユーザーの管理用に機能しません。

kadmind パラメーターのデフォルト値は yes です。 これは、認証時に **kadmind** ルックアップが実行されるということです。 デフォルトの場合、このデーモンが使用不可のときは認証にかかる時間が長くなる可能性があります。

認証時に **kadmind** デーモンの検査を使用不可にするには、次のようにして **methods.cfg** ファイルのスタンザを変更します。

```
KRB5:
        program = /usr/lib/security/KRB5
        options = kadmind=no

KRB5files:
        options = db=BUILTIN,auth=KRB5
```

kadmind デーモンが使用不可の場合、root ユーザーはユーザー・パスワードを変更できません。 パスワードを忘れたりした状態では、**kadmind** デーモンを使用可能にする必要があります。また、ユーザーがログイン・プロンプトに Kerberos プリンシパル名を入力することにした場合、そのプリンシパル名の基本名は、ユーザー名の長さに関する AIX の制限に従って切り捨てられます。 この切り捨てられた名前は、AIX ユーザー識別情報の検索 (ホーム・ディレクトリー値の検索など) のために使用されます。

kadmind デーモンが使用不可 (デーモンがダウンまたは到達不能) の場合、**mkuser** コマンドは次のようなエラーを出します。

```
3004-694 Error adding "krb5user": You do not have permission.
```

kadmind パラメーターが no に設定されているか、または **kadmind** デーモンがアクセス可能になっていない場合、システムは Kerberos データベースにプリンシパルが存在していることを検証することができません。この場合、システムは Kerberos 関連属性を検索しません。 この状態では未完了または不正確な結果になります。例えば、**lsuser** コマンドで全照会を行っても、いずれのユーザーも結果に表示されなくなる可能性があります。

さらに、**chuser** コマンドは AIX 関連属性のみを管理し、 Kerberos 関連属性については管理しません。**rmuser** コマンドによって Kerberos プリンシパルは削除されません。そして、Kerberos 認証済みユーザーに関する **passwd** コマンドは失敗します。

kadmind デーモンが常駐するネットワークがアクセス可能でない場合、応答時間は遅延されます。methods.cfg ファイルで kadmind オプションを no に設定すると、マシンがアクセス可能でない場合の認証時の遅延がなくなります。

kadmind デーモンがダウンしている場合、パスワードが期限切れになっているユーザーはログインできず、またそのパスワードの変更もできません。

kadmind=no と設定していても **kadmind** デーモンが実行されている場合は、 **login**、**su**、**passwd**、**mkuser**、**chuser**、および **rmuser** の各コマンドを実行できます。

ネットワーク認証サービスに対する **Kerberos:** トラブルシューティング情報**:**

AIX オペレーティング・システムで稼働する Kerberos サーバーを使用している Kerberos クライアントについてのトラブルシューティング情報を提供します。

LDAP モジュールはエラーおよびデバッグ情報を syslog サブシステムに書き込みます。

IBM ネットワーク認証サービスは、固有のログ・ファイルを使用して KDC および **kadmind** デーモンに対して行われた要求をログに記録します。 ログ・ファイルは krb5.conf ファイルの [logging] スタンザに指定されます。これらのファイルのデフォルトのロケーションは、/var/krb5/log/krb5kdc.log ファイルと /var/krb5/log/kadmin.log ファイルです。

問題が IBM Tivoli Directory Server に関連している場合は、IBM Tivoli Directory Server によって生成されたログ・ファイルを確認してください。 デフォルトでは、ログ・ファイルは /var/ldap/ibmslapd.log ファイルと /var/ldap/db2cli.log ファイルです。

- ***AIX Kerberos*** 認証済みユーザーを作成するには、どのようにしますか**?**

  root ユーザーは、管理用タスクを実行するために必要な特権を認可するための Kerberos 資格情報を取得する必要があります。 管理用タスクは KDC サーバーの kdcsrv.austin.ibm.com で実行されます。

  次のコマンドを実行して、AIX ユーザー・アカウント (foo) および Kerberos プリンシパル (foo@MYREALM) を Kerberos データベースに作成します。

  ```
  kinit root/kdcsrv.austin.ibm.com
  mkuser –R KRB5files SYSTEM=KRB5files registry=KRB5files foo
  ```

  また、これらのコマンドは KRB5files ファイルに対しても、ユーザーを認証します。

  mkseclдap コマンドを使用して LDAP を構成すれば、次のコマンドを入力して、AIX Kerberos 認証済みユーザーを作成できます。

  ```
  mkuser –R KRB5LDAP SYSTEM=KRB5LDAP registry=KRB5LDAP foo
  ```

- ***Kerberos*** 認証済みユーザーを除去するには、どのようにしますか**?**

  Kerberos 認証済みユーザーを除去するには、次のコマンドを実行します。

  ```
  rmuser –R KRB5files foo
  ```

  mkseclдap コマンドを使用して LDAP を構成すれば、次のコマンドを入力して、Kerberos 認証済みユーザーを除去できます。

  ```
  rmuser –R KRB5LDAP foo
  ```

- ***Kerberos*** 認証済みユーザーのパスワードを変更するには、どのようにしますか**?**

  Kerberos 認証済みユーザーのパスワードを変更するには、次のコマンドを入力します。

  ```
  passwd –R KRB5files foo
  ```

- ***AIX Kerberos*** 拡張属性について説明してください。

  Kerberos プリンシパル情報は、AIX **lsuser** および **chuser** コマンドで AIX 拡張属性を使用して取り扱われます。表示できるのは、GET アクセス・モードを持っている属性だけです。 SET アクセス・モードを持っている属性には、特権ユーザー (AIX オペレーティング・システムでは root) が値を割り当てることができます。 AIX Kerberos 認証済みユーザーは自分の Kerberos 拡張属性、および他の、許可された AIX 属性 (id、pgrp、groups、gecos、home、shell など) を表示できます。

  表 19 は、 AIX Kerberos 拡張属性およびそのアクセス・モードのリストです。

表 *19. AIX Kerberos* 拡張属性およびアクセス・モード

拡張属性名	説明	アクセス・モード
krb5_principal_name	AIX ユーザー名に関連付けられたプリンシパル名	GET
krb5_principal	krb5_principal_name 属性と同じ	GET
krb5_realm	プリンシパルが所属している Kerberos レルム名	GET
krb5_last_pwd_change	プリンシパルのパスワードの最終変更日時	GET
krb5_attributes	KDC により使用される属性のセット	GET/SET
krb5_mod_name	プリンシパルの最終変更者の名前	GET

表 19. AIX Kerberos 拡張属性およびアクセス・モード (続き)

拡張属性名	説明	アクセス・モード
krb5_mod_date	プリンシパルの最終変更日時	GET
krb5_kvno	プリンシパルの現行キー (パスワード) のバージョン	GET/SET
krb5_mkvno	データベース・マスター・キーのバージョン番号。 これは他のインプリメンテーションとの互換性のために提供されます。 このフィールドは 0 です。	GET
krb5_max_renewable_life	プリンシパル用に発行されるチケットの最大更新存続期間	GET/SET
krb5_names	名前:ホスト名ペアのリスト。 このフィールドは将来の使用のために用意されています。 この属性を変更しないでください。	GET/SET

krb5_attributes 拡張属性は、KDC で使用するために使用可能な Kerberos プリンシパル属性のセットです。 特権ユーザーは **chuser** コマンドを使用して Kerberos 属性を変更できます。

```
chuser –R KRB5files krb5_attributes=+requires_preauth krb5user
```

フラグを設定する場合は、フラグの前に (+) を付加します。 フラグをリセットする場合は、フラグの前に (-) を付加します。 例えば、

+attribute_name はフラグを設定します。

-attribute_name はフラグをリセットします。

注**:** ユーザーを作成すると、requires_hwauth、needchange、password_changing_service、および support_desmd5 属性を除く、すべての属性が設定されます。

以下の項目は krb5_attributes 拡張属性の属性です。

allow_postdated
設定すると、プリンシパル用に先日付チケットが発行されます。

allow_forwardable
設定すると、プリンシパル用に転送可能チケットが発行されます。

allow_tgs_req
設定すると、チケット許可チケットを使用してプリンシパル用のサービス・チケットが発行されます。

allow_renewable
設定すると、プリンシパル用に更新可能チケットが発行されます。

allow_proxiable
設定すると、プリンシパル用にプロキシー可能チケットが発行されます。

allow_dup_skey
設定すると、プリンシパル用に user-to-user 認証が使用可能になります。

allow_tix
設定すると、プリンシパル用にチケットが発行されます。

requires_preauth
設定すると、チケットが発行される前に、ソフトウェア事前認証が要求されます。

requires_hwauth
設定すると、チケットがプリンシパル用に発行される前に、ソフトウェアによるハードウェア事前認証が要求されます。

needchange
設定すると、チケットが発行される前に、プリンシパル用のキー (パスワード) を変更する必要があります。

注: needchange フラグが設定されると、ユーザーには次のログイン試行のときに、パスワードを変更するためのプロンプトが出されます。 この場合、ユーザーは (Kerberos 使用を) 認証されますが、チケット許可チケットは持っていません。 チケット許可チケットを取得するには、ユーザーは **kinit** コマンドを起動する必要があります。 needchange フラグは、ネットワーク認証サービス・モジュールを使用している Kerberos のみに適用されます。

allow_svr
設定すると、プリンシパル用にサービス・チケットが発行されます。

password_changing_service
設定すると、プリンシパルはサービスを変更するためのパスワード用の特殊プリンシパルです。

support_desmd5
設定すると、KDC は RSA MD5 チェックサム・アルゴリズムを使用するチケットを発行する場合があります。

注: この属性を設定すると、相互運用性の問題が発生する可能性があります。

- ***AIX Kerberos*** 拡張属性をリストするには、どのようにしますか**?**

  AIX Kerberos 拡張属性をリストするには、次のコマンドを入力します。

  ```
  lsuser –R KRB5files foo
  ```

  –a オプションを使用すると、特定の拡張属性もリストすることができます。例えば、

  ```
  lsuser -R KRB5files -f -a krb5_principal krb5_principal_name krb5_realm
  ```

- ***AIX Kerberos*** 拡張属性を変更するには、どのようにしますか**?**

  特権ユーザーのみ、SET アクセス・モードを指定されている krb5_kvno、krb5_max_renewable_life、krb5_attributes and krb5_names の各拡張属性を変更できます。

  – foo に発行されたチケットの最大更新可能存続期間を 5 日に変更するには、次のコマンドを入力します。

    ```
    chuser -R KRB5files krb5_max_renewable_life=432000 foo
    ```

  – foo に関連付けられたプリンシパルのキー (パスワード) バージョン番号を変更するには、次のコマンドを入力します。

    ```
    chuser -R KRB5files krb5_kvno=4 foo
    ```

  – 333 ページの表 19 にリストされた Kerberos プリンシパル属性をすべて設定するには、次のコマンドを入力します。

    ```
    chuser -R KRB5files krb5_attributes=+allow_postdated,+allow_forwardable,¥
    +allow_tgs_req,+allow_renewble,+allow_proxiable,+allow_dup_skey,+allow_tix,¥
    +requires_preauth,+requires_hwath,+needchange,+allow_svr,¥
    +password_changing_service,+support_desmd5 foo

    lsuser -R KRB5files -a krb5_attributes foo
    ```

– 333 ページの表 19 にリストされた Kerberos プリンシパル属性をすべてリセットするには、次のコマンドを入力します。

```
chuser -R KRB5files krb5_attributes=-allow_postdated,-allow_forwardable,¥
-allow_tgs_req,-allow_renewable,-allow_proxiable,-allow_dup_skey,¥
-allow_tix,-requires_preauth,-requires_hwauth,-needchange,-allow_svr,¥
-password_changing_service,-support_desmd5 foo

lsuser -R KRB5files -a krb5_attributes foo
```

– krb5_names を変更して、AIX ユーザー名/ホスト名ペアを追加するには、次のコマンドを入力します。

```
lsuser -R KRB5files -a krb5_names foo

chuser -R KRB5files krb5_names=bar:greenjeans.austin.ibm.com foo

lsuser -R KRB5files -a krb5_names foo
```

- ***KRB5files*** に定義されているユーザーのすべてをリストするには、どのようにしますか**?**

  Kerberos 認証済みユーザーのすべてをリストするには、次のコマンドを実行します。

  ```
  lsuser -R KRB5files -a registry ALL
  ```

- ***AIX*** ユーザーを ***Kerberos*** 認証済みユーザーに変換するには、どのようにしますか**?**

  mkseckrb5 コマンドを使用して、AIX ユーザーを Kerberos 認証済みユーザーに変換します。 **mkseckrb5** コマンドは、非管理ユーザー (201 より大きいユーザー ID を持つユーザー) を Kerberos 認証済みユーザーに変換します。 **mkseckrb5** コマンドを起動すると、ネットワーク認証サービスの管理プリンシパル名とパスワードのためのプロンプトが出されます。 ランダム化オプションを使用しない場合は、変換する各ユーザーのパスワードについてもプロンプトが出されます。

  注**: mkseckrb5** コマンドはローカル・ユーザーのみを変換します。 LDAP などのリモート・ドメインのユーザーについては、このコマンドを使用して変換できません。

  次の例は AIX ユーザーを Kerberos 認証済みユーザーに変換するときに、ランダム化オプションを使用しない 例です。

  1. 次のコマンドを入力します。

     ```
     mkseckrb5 foo
     ```

  2. Kerberos のユーザーでログインするには、その前にユーザーの SYSTEM およびレジストリー属性を次のように設定します。

     ```
     chuser -R KRB5files SYSTEM=KRB5files registry=KRB5files foo
     ```

  次の例は AIX ユーザーを Kerberos 認証済みユーザーに変換するときに、ランダム化オプションを使用する例です。

  1. 次のコマンドを入力します。

     ```
     mkseckrb5 -r user1
     ```

  2. 変換の完了後は、ユーザーの SYSTEM、レジストリー属性、およびパスワードを次のように設定します。

     ```
     chuser -R KRB5files SYSTEM=KRB5files registry=KRB5files user1

     passwd -R KRB5files user1
     ```

- ***Kerberos*** プリンシパルのパスワードを変更するには、どのようにしますか**?**

  root ユーザーは次の **passwd** コマンドを入力して Kerberos プリンシパルのパスワードを設定できます。

```
passwd -R KRB5files foo
```

passwd コマンドを入力すると、次のメッセージが表示されます。

```
Changing password for "foo"
foo's Old password:
foo's New password:
Enter the new password again:
```

root ユーザーとして **passwd** コマンドを入力すると、古いパスワードは無視されます。 古いパスワードについては、methods.cfg ファイルに rootpwdrequired オプションを使用して、プロンプトを使用不可に設定できます。古いパスワードのプロンプトを使用不可に設定するには、次のように /usr/lib/security/methods.cfg ファイルを編集します。

```
KRB5files:
   options = db=BUILTIN,auth=KRB5,rootrequiresopw=false
```

- ***needchange*** 属性が設定されている場合の正常なログイン後に、チケット許可チケットを取得するには、どのようにしますか**?**

  needchange フラグが設定されている場合の正常なログイン後に、チケット許可チケットを取得するには、**kinit** コマンドを起動します。 このサブジェクトの詳細情報については、「**needhange** 属性」を参照してください。

- 私のパスワードが ***AIX*** オペレーティング・システムで承認されないのは、なぜですか **?**

  ご使用のパスワードが承認されていない場合は、次のことを行ってください。

  – KDC および kadmind サーバーが稼働中であることを確認します。
  – パスワードが AIX オペレーティング・システムとネットワーク認証サービスの両方の要件に合っているか確認します。

- パスワード・ルールを変更するには、どのようにしますか**?**

  AIX オペレーティング・システム上で password-policy 属性を変更して、パスワード・ルールを変えることができます。 Kerberos データベース上でパスワード・ポリシーを変更する場合は、ネットワーク認証サーバーの kadmin ツールを使用できます。

- 標準 ***AIX*** 認証のみ使用することで、 ***Kerberos*** 認証済みユーザーを認証済みにすることができますか**?**

  Kerberos 認証済みユーザー (foo) は、AIX **crypt()** 認証を次のように使用して、認証済みにすることができます。

  1. **passwd** コマンドを使用して、ユーザー foo (/etc/security/passwd) の AIX パスワードを設定します。
  2. テスト目的で、異なるパスワードを選択します。 例えば、

     ```
     passwd -R files foo
     ```

  3. ユーザーの SYSTEM 属性を、次のように変更します。

     ```
     chuser -R KRB5files SYSTEM=compat foo
     ```

     SYSTEM 属性変更すると、認証方式は Kerberos から **crypt()** に変わります。

     注**:** この例では、ユーザーはローカル認証を使用してログインするため、AUTHSTATE 値は compat であり、チケット許可チケットは発行されません。 バックアップ手段として **crypt()** 認証を使用したい場合は、ステップ 4 に進んでください。

  4. バックアップ手段として **crypt()** 認証を使用するには、次のように SYSTEM 属性を変更します。

     ```
     chuser -R KRB5files SYSTEM="KRB5files or compat" foo
     ```

- クライアント ***kadmind*** のポートを変更するには、どのようにしますか**?**

  NAS を使用している Kerberos 認証済みシステムで Kerberos プリンシパル管理を実行するには、**kadmind** デーモンを使用します。 次の例はクライアント **kadmind** のポートを変更する方法の説明です。 この例では、**kadmind** デーモンは kdcsrv.austin.ibm.com サーバー上で実行し、ポート 812 を使用します。

  1. **config.krb5** コマンドを使用して、クライアントを構成します。

     ```
     config.krb5 –C –r MYREALM –c kdcsrv.austin.ibm.com –s ¥
     kdcsrv.austin.ibm.com –d austin.ibm.com
     ```

  2. krb5.conf ファイルを編集して、ポート番号を変更します。

     ```
     admin_server = kdcsrv.austin.ibm.com:812
     ```

- ***Kerberos*** 資格情報を除去するには、どのようにしますか**?**

  ユーザーがログインするたびに、前の Kerberos 資格情報は上書きされます。しかし、ユーザーがログアウトするときに、これらの資格情報は除去されません。これらの資格情報を除去するには、次のように NAS **kdestroy** コマンドを入力します。

  ```
  /usr/krb5/bin/kdestroy
  ```

- ***KDC*** 上で ***ticket-life*** 時間を変更するには、どのようにしますか**?**

  KDC 上で ticket-life 時間を変更するには、次のようにします。

  1. kdc.conf ファイルの max_life 属性を変更します。例えば、

     ```
     max_life = 8h 0m 0s
     ```

  2. **krb5kdc** および **kadmind** デーモンを停止して、開始します。
  3. krbtgt/MYREALM および kadmin/admin プリンシパルの max_life 値をステップ 1 で入力した値に変更します。 次に例を示します。

     ```
     kadmin.local
     kadmin.local: modify_principal -maxlife "8 hours" krbtgt/MYREALM
     ```

- ***kadmind*** デーモンが使用不可の場合は、どのようになりますか**?**

  kadmind デーモンが使用不可の場合は、認証にかかる時間が長くなるか、または失敗する可能性があります。 kadmind デーモンが配置されているネットワークの部分がアクセス可能でないか、または kadmind サーバーをホストしているシステムがダウンしている場合、認証は失敗する場合があります。システムがアクセス不可の場合、methods.cfg ファイルで kadmind オプションが no に設定してあると、認証時の遅れは解消されます。

  kadmind デーモンがダウンしている場合、パスワードが期限切れになっているユーザーはログインできません。 kadmind デーモンが使用不可で (デーモンがダウンしているかまたは到達不能である場合)、ユーザーが **mkuser** コマンドを入力すると、次のようなエラーが表示されます。

  ```
  3004-694 Error adding "krb5user": You do not have permission
  ```

  さらに、**chuser** および **lsuser** コマンドでは AIX 関連属性のみが管理され、Kerberos 関連属性は管理されません。 **rmuser** コマンドは Kerberos プリンシパルを削除しません。したがって、**passwd** コマンドは Kerberos 認証済みユーザーの場合に失敗します。

  kadmind デーモンが使用不可の場合、root ユーザーはユーザー・パスワードを変更できません。 例えば、パスワードを忘れたりした状態では、kadmind デーモンを使用可能にする必要があります。また、ユーザーがログイン・プロンプトに Kerberos プリンシパル名を入力することを選択した場合、そのプ

リンシパル名の基本名は (AIX ユーザー名の長さ制限に従って) 切り捨てられます。 この切り捨てられた名前が AIX ユーザー識別のための情報検索に使用されます (例えば、ホーム・ディレクトリー値の検索)。

- ***LDAP*** の ***AIX*** ユーザー/グループ管理を使用して ***Kerberos*** 統合ログインを行えるよう ***AIX*** オペレーティング・システムを構成するには、どのようにするのですか**?**

  LDAP を使用して AIX ユーザーやグループの情報を保管することについて計画している場合は、**mkkrb5srv** および **mkkrb5clnt** コマンドを実行する前に、**mksecldap** コマンドを使用して LDAP サーバーおよびクライアントを構成します。 Kerberos サーバーを構成するには、**mkkrb5srv** コマンドを使用します。 Kerberos クライアントを構成するには、–i LDAP オプションを指定した **mkkrb5clnt** コマンドを使用します。 例えば、

  ```
  mkkrb5clnt -r MYREALM -c kdcsrv.ustin.ibm.com¥
   –s kdcsrv.austin.ibm.com -a admin/admin -d austin.ibm.com -A -i LDAP -K –T
  ```

- 正常なログイン後に ***Kerberos*** 対応のリモート・コマンドを使用するには、どのようにするのですか**?**

  AIX ユーザーが Kerberos を使用してシステムに認証されると、 Kerberos 対応のリモート・コマンドに対してチケット許可チケットを使用できます。

  次の例では、**mkkrb5srv** コマンドを使用して、NAS サーバーが kdcsrv.austin.ibm.com に構成されます。 また、このシステムは **mkkrb5clnt** コマンドを使用して、Kerberos ベースのログインも構成されます。 2 番目のシステムの tx3d.austin.ibm.com は **mkkrb5clnt** コマンドを使用することで、クライアントとして構成されます。

  1. ホスト・プリンシパル host/tx3d.austin.ibm.com のキーを tx3d システムにある /etc/krb5/krb5.keytab ファイルに保存します。
  2. クライアント・マシンを構成するときに **mkkrb5clnt** が使用されているため、これらのキーは /var/krb5/security/keytab/tx3d.austin.ibm.com.keytab ファイルに抽出されます。次のように指定して、このファイルを /etc/krb5/krb5.keytab ファイルにリンクします。

     ```
     ln -s /var/krb5/security/keytab/tx3d.austin.ibm.com.keytab /etc/krb5/krb5.keytab
     ```

  3. 非 AIX Kerberos サーバーを使用して tx3d.austin.ibm.com システムを構成すると、明示的にホスト・プリンシパルが作成され、キーが抽出されます。 例えば、

     ```
     kadmin -p admin/admin

     kadmin: addprinc -randkey host/tx3d.austin.ibm.com
     kadmin: ktadd -k /etc/krb5/krb5.keytab host/tx3d.austin.ibm.com
     kadmin:
     ```

     kadmin ツールは tx3d.austin.ibm.com システムから起動されるため、キーは tx3d.austin.ibm.com システム上の /etc/krb5/krb5.keytab ファイルに抽出されます。 また、Kerberos 管理サーバーをホスティングするマシン (例えば、kdcsrv) 上で、このステップを実行できます。キーを Keytab ファイルに抽出してから、このファイルを転送して tx3d 上の /etc/krb5/krb5.keytab ファイルとマージします。

  4. リモート・コマンドを使用可能にして、Kerberos バージョン 5 認証を tx3d.austin.ibm.com システムで使用します。

     ```
     lsauthent
     Standard Aix
     chauthent -k5 -std
     lsauthent
     Kerberos 5
     Standard Aix
     ```

5. リモート・コマンドを使用可能にして、Kerberos バージョン 5 認証を kdcsrv.austin.ibm.com システムで使用します。

```
chauthent -k5 -std
lsauthent
Kerberos 5
Standard Aix
```

6. Kerberos 認証ユーザー (foo) を kdcsrv に作成して、パスワードを設定します。

```
mkuser -R KRB5files SYSTEM=KRB5files registry=KRB5files foo
passwd -R KRB5files foo
```

7. ユーザー foo を tx3d に作成します。

```
mkuser -R files foo
```

8. Kerberos 認証を使用して kdcsrv.austin.ibm.com システムに Telnet でログインします。
9. チケット許可チケットが発行されたことを確認するために、**klist** コマンドを入力します。

```
/usr/krb5/bin/klist
```

以下は、Kerberos 対応のリモート・コマンドです。

注**:** 以下の例のコマンドを実行する前に、 `.klogin`、`.rhost` または `hosts.equiv` ファイルを除去します。

– リモート tx3d.austin.ibm.com ホスト・システム上で **rsh** コマンドにより **date** コマンドを入力します。

```
rsh tx3d date
```

– **rlogin** コマンドを使用して、リモート tx3d.austin.ibm.com システムにログインします。

```
hostname
kdcsrv.austin.ibm.com
rlogin tx3d -l foo
*******************************
* Welcome to AIX Version 6.1! *
*******************************
hostname
tx3d.austin.ibm.com
id
uid=234(foo) gid=1(staff)
```

– **rcp** コマンドを使用して、ファイルをリモート tx3d.austin.ibm.com システムに転送します。

```
rsh tx3d "ls -l /home/foo"
total 0
echo "Testing Kerberize-d rcp" >> xfile
rcp xfile tx3d:/home/foo
rsh tx3d "ls -l /home/foo"
total 0
-rw-r--r-- 1 foo staff 0 Apr 28 14:30 xfile
rsh tx3d "more /home/foo/xfile"
Testing Kerberize-d rcp
```

– Kerberos 資格情報を指定して、リモート tx3d.austin.ibm.com システムに Telnet でログインします。

```
telnet tx3d
Trying...
Connected to tx3d.austin.ibm.com.
Escape character is '^]'.
[ Kerberos V5 accepts you as "foo@MYREALM" ]
```

– tx3d.austin.ibm.com システムに Telnet でログインし、プロンプトが出されたらホスト名と ID を入力します。

```
hostname
tx3d.austin.ibm.com
id
uid=234(foo) gid=1(staff)
```

– Kerberos 対応の **ftp** コマンドを使用できるようにするには、 **kadmin** コマンドを (tx3d.austin.ibm.com から) 使用し、 FTP サービス・プリンシパルの ftp/tx3d.austin.ibm.com を作成し、それを /etc/krb5/krb5.keytab ファイルに抽出する必要があります。

```
kadmin: addprinc -randkey ftp/tx3d.austin.ibm.com@MYREALM
kadmin: ktadd -k /etc/krb5/krb5.keytab ftp/tx3d.austin.ibm.com@MYREALM
kadmin:
```

以下は、Kerberos 資格情報を指定して tx3d.austin.ibm.com リモート・システムへ FTP でファイル転送する方法の例です。

```
ftp tx3d
Name (tx3d:foo): foo
232 GSSAPI user foo@MYREALM is authorized as foo
230-Last login: Thu May 19 17:58:57 CDT 2005 on ftp from kdcsrv.austin.ibm.com
230 User foo logged in.
ftp> ftp> ls -la
```

非 **AIX** システムでの **Kerberos** サーバーに対する **Kerberos** クライアントの構成**:**

非 AIX システム (Windows Active Directory、Solaris、および HP) で稼働する Kerberos サーバーに対して、AIX Kerberos クライアントを構成できます。

Windows Server Kerberos サービスに対する ***Kerberos*** の構成*:*

Windows Server Kerberos サービスに対する Kerberos の構成には、いくつかの方式があります。

KRB5 の Kerberos 認証専用モジュールは複合ロード・モジュールの認証の一部に使用できます。 ユーザーは構成でロード・モジュールに対する Kerberos 環境を指定します。 KRB5 ロード・モジュールは、Windows 2000 または Windows 2003 Server Kerberos サービスに対する認証のための代わりの方式として、 Kerberos を使用可能にします。 AIX BUILTIN 疑似ロード・モジュールにより、セキュリティー・ライブラリー関数へのアクセスが可能になります。 BUILTIN ロード・モジュールは認証専用ロード・モジュールと結合することで、複合ロード・モジュールのデータベース・パーツを提供することが可能です。また、このロード・モジュールは、レガシー・ユーザーおよびグループ・ストレージとファイルシステムへのアクセスも可能にします。 LDAP ロード・モジュールは、複合ロード・モジュールのデータベース・パーツとしても使用できます。

AIX システム上の NAS に対する他の Kerberos 環境と違って、この環境は Kerberos プリンシパル管理を提供しません。 KRB5 ロード・モジュールは、Kerberos プリンシパルが非 AIX システムに保管される環境で使用可能で、**kadmin** Kerberos データベース・インターフェースの使用による AIX オペレーティング・システムからの管理はできません。 Kerberos プリンシパル管理は Kerberos プリンシパル管理ツールを使用して単独で実行されます。 これらのツールはソフトウェア・ベンダーが開発した Kerberos 製品に含まれているか、または Windows 2000 などの OS に組み込まれている場合があります。

Windows Server 2000 Kerberos サービスの構成*:*

Windows Server 2000 Kerberos サービスと NAS クライアントは、 Kerberos プロトコル・レベル (RFC1510) で相互運用できます。 Windows Server 2000 は **kadmin** インターフェースをサポートしないため、 AIX クライアントの構成では **mkkrb5clnt** コマンドに –D フラグを指定します。 Windows システムでプリンシパルを管理する場合は、 Windows ツールを使用します。

Windows Server 2000 Kerberos サービスに対して Kerberos ベース認証のための AIX クライアントを構成するには、以下の手順を使用します。

1. Windows Server 2000 をセットアップします。 Microsoft Active Directory Server の構成については、 Microsoft の文書を参照してください。
2. NAS クライアントが AIX クライアントにインストールされていない場合は、 AIX 拡張パックから `krb5.client.rte` ファイルセットをインストールします。
3. 以下の構成情報を指定した **mkkrb5clnt** コマンドを使用して、 AIX Kerberos クライアントを構成します。

   レルム
   : Windows Active Directory のドメイン名

   ドメイン
   : Active Directory サーバーをホスティングするマシンのドメイン名

   KDC
   : Windows サーバーのホスト名

   サーバー
   : Windows サーバーのホスト名

   mkkrb5clnt コマンドの例を以下に示します。

   ```
   mkkrb5clnt -r MYREALM -d austin.ibm.com -c w2k.austin.ibm.com -s w2k.austin.ibm.com -D
   ```

   mkkrb5clnt コマンドに **-D** オプションを指定すると、`/etc/methods.cfg` ファイルに **is_kadmind_compat=no** オプションが作成され、非 AIX システムに対する認証用の Kerberos クライアント環境が構成されます。 IBM ネットワーク認証サービス (NAS) に対する認証用の Kerberos クライアント環境を構成するには、**mkkrb5clnt** コマンドに **-D** オプションを使用しないでください。

   注: **mkkrb5clnt** コマンドを実行すると、次のスタンザが `methods.cfg` ファイルに追加されます。

   ```
   KRB5:
           program = /usr/lib/security/KRB5
           program_64 = /usr/lib/security/KRB5_64
           options = authonly,is_kadmind_compat=no

   KRB5files:
           options = db=BUILTIN,auth=KRB5
   ```

   詳細情報の参照先:

   - **mkkrb5clnt** コマンドおよび許容フラグについては、**mkkrb5clnt** コマンドを参照してください。
   - `methods.cfg` ファイルについては、`methods.cfg` ファイルを参照してください。
4. Windows は DES-CBC-MD5 および DES-CBC-CRC 暗号化タイプをサポートしているため、`krb5.conf` ファイル情報を以下の内容に類似したものに変更します。

   ```
   [libdefaults]
       default_realm = MYREALM
       default_keytab_name = FILE:/etc/krb5/krb5.keytab
       default_tkt_enctypes = des-cbc-md5 des-cbc-crc
       default_tgs_enctypes = des-cbc-md5 des-cbc-crc
   ```

5. ホスト・プリンシパルを作成します。

   Windows アカウント名は NAS プリンシパル名のように複数パーツに分かれていないため、完全修飾ホスト名 (`host/<fully_qualified_host_name>`) を使用してアカウントを直接作成できません。 その代りに、「service-principal-name (サービス-プリンシパル-名前)」マッピングからプリンシパル・

インスタンスが作成されます。 このケースでは、ホスト・プリンシパル用のアカウントが作成され、「principal-name (プリンシパル-名前)」マッピングが追加されます。

Active Directory サーバー上で Active Directory 管理ツールを使用して、次のように tx3d.austin.ibm.com AIX クライアント用の新規ユーザー・アカウントを作成します。

a. 「User (ユーザー)」フォルダーを選択します。
b. 右クリックして、「New (新規)」を選択します。
c. 「User (ユーザー)」を選択します。
d. 「First name (ファーストネーム)」フィールドに tx3d を入力して、 「次へ」をクリックします。
e. パスワードを作成して、「次へ」をクリックします。
f. 「完了」をクリックして、ホスト・プリンシパルを作成します。

6. Windows Server 2000 マシン上で、コマンド・ラインから **Ktpass** コマンドを入力して、tx3d.keytab ファイルを作成し、次のように AIX ホスト・アカウントをセットアップします。

```
Ktpass -princ host/tx3d.austin.ibm.com@MYREALM -mapuser tx3d -pass password -out tx3d.keytab
```

7. tx3d.keytab ファイルを AIX ホスト・システムへコピーします。
8. tx3d.keytab ファイルを AIX システムの /etc/krb5/krb5.keytab ファイルへ次のようにマージします。

```
ktutil
rkt tx3d.keytab
wkt /etc/krb5/krb5.keytab
q
```

9. Active Directory のユーザー管理ツールを使用して、 Windows ドメイン・アカウントを作成します。
10. Windows ドメイン・アカウント用の AIX アカウントを作成して Kerberos 認証を使用するには、次のコマンドを実行します。

```
mkuser registry=KRB5files SYSTEM=KRB5files foo
```

11. AIX システムにログインして構成を検査するには、 **telnet** コマンドを実行します。

Windows Active Directory に対して KRB5 を使用するための Kerberos 統合ログイン・セッションの例を以下に示します。

```
telnet tx3d

Trying...
Connected to tx3d.austin.ibm.com.
Escape character is '^]'.

telnet (tx3d.austin.ibm.com)
login: foo
foo's Password:
*******************************************************************************
* Welcome to AIX Version 6.1! *
*******************************************************************************
echo $AUTHSTATE
KRB5files

/usr/krb5/bin/klist
Ticket cache: FILE:/var/krb5/security/creds/krb5cc_foo@AUSTIN.IBM.COM_203
Default principal: foo@AUSTIN.IBM.COM

Valid starting Expires Service principal
04/29/05 14:37:28 04/30/05 00:39:22 krbtgt/AUSTIN.IBM.COM@AUSTIN.IBM.COM
```

```
        Renew until 04/30/05 14:37:28

    04/29/05 14:39:22 04/30/05 00:39:22 host/tx3d.austin.ibm.com@AUSTIN.IBM.COM
```

Windows Server 2003 Kerberos サービスの構成*:*

Windows Server 2003 Kerberos サービスに対して Kerberos クライアントを構成できます。

Windows Server 2003 Kerberos サービスに対して AIX クライアントを構成するには、 341 ページの『Windows Server 2000 Kerberos サービスの構成』に記載されている手順を使用します。

注**:** NAS **kpasswd** クライアント・ユーティリティーでは、 Windows Server 2003 Kerberos サービスに関する Kerberos プリンシパルのパスワードを変更できません。 したがって、Kerberos を使用している AIX システムへ正常にログインした後では、ユーザーは Windows Server 2003 のパスワードを変更できません。

Sun Solaris および ***HP-UX Kerberos*** ドメイン・コントローラーに対する ***Kerberos*** の構成*:*

Sun Solaris および HP-UX Kerberos ドメイン・コントローラーに対して Kerberos クライアントを構成できます。

AIX システム上の NAS に対する Kerberos 環境と違って、この環境では Kerberos プリンシパル管理は提供されません。 KRB5 ロード・モジュールは、Kerberos プリンシパルが非 AIX システムに保管される環境で使用可能で、**kadmin** Kerberos データベース・インターフェースの使用による AIX オペレーティング・システムからの管理はできません。 Kerberos プリンシパル管理は Kerberos プリンシパル管理ツールを使用して単独で実行されます。 これらのツールはソフトウェア・ベンダーが開発した Kerberos 製品に含まれているか、または OS に組み込まれている場合があります。

Sun Solaris の構成*:*

Sun Solaris に対して Kerberos クライアントを構成できます。

Sun Enterprise Authentication Mechanism (SEAM) と AIX NAS クライアントは、 Kerberos プロトコル・レベル (RFC1510) で相互運用されます。 Solaris **kadmind** デーモンのインターフェースは AIX NAS クライアントの **kadmin** インターフェースと互換性がないため、 AIX クライアントを構成するときに **mkkrb5clnt** コマンドに –D フラグを指定します。 Solaris システム上でプリンシパル管理を行う場合は、Solaris ツールを使用します。 パスワード変更のためのプロトコルが SEAM Kerberos サーバーと AIX NAS サーバーとで異なるため、プリンシパルのパスワード変更は構成を失敗させる原因になります。

Solaris は以下の例のように使用します。

SEAM に対する Kerberos ベース認証のために AIX クライアントを構成するには、以下の手順を使用します。

1. SUN 文書を確認して SEAM を構成します。
2. NAS クライアントが AIX クライアントにインストールされていない場合は、 AIX 拡張パックから `krb5.client.rte` ファイルセットをインストールします。
3. AIX Kerberos クライアントを構成するには、以下の構成情報を指定した **mkkrb5clnt** コマンドを使用します。

   レルム
   : Solaris Kerberos レルム名: AUSTIN.IBM.COM

ドメイン
: Kerberos servers サーバーをホスティングするマシンのドメイン名: Austin.ibm.com

KDC
: KDC をホスティングする Solaris システムのホスト名: sunsys.austin.ibm.com

サーバー
: **kadmin** デーモンをホスティングする Solaris システムのホスト名 (通常は KDC と同じ): sunsys.austin.ibm.com

注**:** Solaris と AIX NAS クライアントの **kadmin** インターフェースは異なるため、 NAS クライアントではサーバー名を使用しないで、**mkkrb5clnt** コマンドに –D フラグを使用する必要があります。

mkkrb5clnt コマンドの例を以下に示します。

```
mkkrb5clnt -r AUSTIN.IBM.COM -d austin.ibm.com¥
 -c sunsys.austin.ibm.com -s sunsys.austin.ibm.com -D
```

mkkrb5clnt コマンドに **-D** オプションを指定すると、/etc/security/methods.cfg ファイルに **is_kadmind_compat=no** オプションが作成され、非 AIX システムに対する認証用の Kerberos クライアント環境が構成されます。 IBM ネットワーク認証サービス (NAS) に対する認証用の Kerberos クライアント環境を構成するには、**mkkrb5clnt** コマンドに **-D** オプションを使用しないでください。

注**: mkkrb5clnt** コマンドを実行すると、次のスタンザが methods.cfg ファイルに追加されます。

```
KRB5:
        program = /usr/lib/security/KRB5
        program_64 = /usr/lib/security/KRB5_64
        options = authonly,is_kadmind_compat=no

KRB5files:
        options = db=BUILTIN,auth=KRB5
```

詳細情報の参照先:

- **mkkrb5clnt** コマンドおよび許容フラグについては、**mkkrb5clnt** コマンドを参照してください。
- methods.cfg ファイルについては、methods.cfg ファイルを参照してください。

4. 以下のように、Solaris **kadmin** ツールを使用して、host/tx3d.austin.ibm.com@MYREALM ホスト・プリンシパルを作成し、それをファイルに保存します。

```
kadmin: add_principal -randkey host/tx3d.austin.ibm.com
Principal "host/tx3d.austin.ibm.com@AUSTIN.IBM.COM" created.

kadmin:ktadd -k /tmp/tx3d.keytab host/tx3d.austin.ibm.com
Entry for principal host/tx3d.austin.ibm.com with kvno 3,
    encryption type DES-CBC-CRC added to keytab WRFILE:/tmp/tx3d.keytab.

kadmin: quit
```

5. tx3d.keytab ファイルを AIX ホスト・システムへコピーします。
6. tx3d.keytab ファイルを AIX システムの /etc/krb5/krb5.keytab ファイルへ次のようにマージします。

```
ktutil
rkt tx3d.keytab
l
slot KVNO Principal
wkt /etc/krb5/krb5.keytab
q
```

7. Kerberos プリンシパルを作成するには、Solaris **kadmin** ツールを使用します。

```
add_principal sunuser
```

8. Solaris Kerberos プリンシパル用の AIX アカウントを作成して Kerberos 認証を使用するには、次のコマンドを実行します。

```
mkuser registry=KRB5files SYSTEM=KRB5files sunuser
```

9. **telnet** コマンドを使用し、 sunuser のユーザー名とパスワードを指定して AIX システムにログインし、構成を検査します。

   Solaris KDC に対して KRB5 を使用するための Kerberos 統合ログイン・セッションの例を以下に示します。

```
telnet tx3d

echo $AUTHSTATE
KRB5files

echo $KRB5CCNAME
FILE:/var/krb5/security/creds/krb5cc_sunuser@AUSTIN.IBM.COM_207

View credentials:
/usr/krb5/bin/klist
```

HP-UX の構成**:**

HP-UX に対して Kerberos クライアントを構成できます。

HP-UX 11i に対する認証のステップは 344 ページの『Sun Solaris の構成』の手順に類似しています。 HP-UX KDC と AIX NAS クライアントは、 Kerberos プロトコル・レベル (RFC1510) で相互運用されます。 パスワード変更のプロトコルも互換性があります。 HP-UX **kadmind** デーモンのインターフェースは AIX NAS クライアントの **kadmin** インターフェースと互換性がないため、 AIX クライアントを構成するときに **mkkrb5clnt** コマンドに –D フラグを指定する必要があります。

以下の手順を使用して、HP-UX 11i Kerberos バージョン 2.1 に対する Kerberos ベース認証のための AIX クライアントを構成します。

1. HP 文書を確認して HP-UX 11i Kerberos バージョン 2.1 を構成します。
2. NAS クライアントが AIX クライアントにインストールされていない場合は、 AIX 拡張パックから `krb5.client.rte` ファイルセットをインストールします。
3. 以下の構成情報を指定した **mkkrb5clnt** コマンドを使用して、 AIX Kerberos クライアントを構成します。

   レルム
   : HP Kerberos レルム名: HPSYS.AUSTIN.IBM.COM

   ドメイン
   : HP-UX Kerberos サーバーをホスティングするマシンのドメイン名: austin.ibm.com

   KDC
   : KDC をホスティングする HP-UX システムのホスト名 : hpsys.austin.ibm.com

   サーバー
   : HP-UX サーバーのホスト名: hpsys.austin.ibm.com

   注**:** HP-UX と AIX NAS クライアントの **kadmin** インターフェースは異なるため、 NAS クライアントではサーバー名を使用しないで、**mkkrb5clnt** コマンドに –D フラグを使用する必要があります。

   mkkrb5clnt コマンドの例を以下に示します。

```
mkkrb5clnt -r AUSTIN.IBM.COM -d austin.ibm.com¥
 -c hpsys.austin.ibm.com -s hpsys.austin.ibm.com -D
```

mkkrb5clnt コマンドに **-D** オプションを指定すると、/etc/security/methods.cfg ファイルに **is_kadmind_compat=no** オプションが作成され、非 AIX システムに対する認証用の Kerberos クライアント環境が構成されます。 IBM ネットワーク認証サービス (NAS) に対する認証用の Kerberos クライアント環境を構成するには、**mkkrb5clnt** コマンドに **-D** オプションを使用しないでください。

注**: mkkrb5clnt** コマンドを実行すると、次のスタンザが methods.cfg ファイルに追加されます。

```
KRB5:
        program = /usr/lib/security/KRB5
        program_64 = /usr/lib/security/KRB5_64
        options = authonly,is_kadmind_compat=no

KRB5files:
        options = db=BUILTIN,auth=KRB5
```

詳細情報の参照先:

- **mkkrb5clnt** コマンドおよび許容フラグについては、**mkkrb5clnt** コマンドを参照してください。
- methods.cfg ファイルについては、methods.cfg ファイルを参照してください。

4. krb5.conf ファイルを変更して、暗号化タイプを HP-UX Kerberos セットアップ (**krbsetup**) のときに使用される値と一致させます。 DES-CRC 値を使用する場合は、次のように AIX クライアント上で krb5.conf ファイルの [libdefaults] スタンザを編集します。

   ```
   default_tkt_enctypes = des-cbc-crc

   default_tgs_enctypes = des-cbc-crc
   ```

5. HP-UX **kadmin_ui** ツールを使用して、host/tx3d.austin.ibm.com ホスト・プリンシパルを作成します。
6. キーを抽出して、それをファイルに保存します。 「Principal Information (プリンシパル情報)」ウィンドウの「**編集**」メニューから 「Extract Service Key (サービス・キーの抽出)」を選択して、キーを抽出します。
7. tx3d.keytab ファイルを AIX ホスト・システムへコピーします。
8. tx3d.keytab ファイルを AIX システムの /etc/krb5/krb5.keytab ファイルへ次のようにマージします。

   ```
   ktutil
   rkt tx3d.keytab
   l
   slot KVNO Principal
   wkt /etc/krb5/krb5.keytab
   q
   ```

9. HP-UX **kadmin_ui** ツールを使用して hpuser の Kerberos プリンシパルを作成し、「Edit/Attribute」タブをクリックして pw_require フラグをクリアします。
10. 次のように、HP-UX 上で Kerberos プリンシパル用の AIX アカウントを作成します。

    ```
    mkuser registry=KRB5files SYSTEM=KRB5files hpuser
    ```

11. **telnet** コマンドを使用し、hpuser のユーザー名とパスワードを指定して AIX システムにログインし、構成を検査します。

    HP-UX に対して KRB5 を使用するための Kerberos 統合ログイン・セッションの例を以下に示します。

```
telnet tx3d

echo $AUTHSTATE
KRB5files

View credentials:
/usr/krb5/bin/klist
```

12. **passwd** コマンドを使用して、パスワードを変更します。

    注**:** パスワードの変更の際は HP-UX パスワード・ポリシーを順守します。 パスワード・ポリシーを設定する方法を決める場合は、HP-UX の文書を参照してください。

非 ***AIX*** システムに対する ***Kerberos:*** 質問とトラブルシューティング情報***:***

非 AIX システムで稼働する Kerberos サーバーを使用している Kerberos クライアントについての質問に回答します。

注**:** 以下の例では、 Microsoft Active Directory Server が使用されます。 しかし、ここに挙げた例は Solaris および HP システムにも適用できます。

トラブルシューティングの最初のステップとして、サーバーとデーモンのすべてが稼働していることを確認してください。

非 AIX システムに対する Kerberos は、エラーおよびデバッグに関する情報を記録するために syslog サブシステムを使用します。 syslog のロギングについて詳しくは、「**syslogd** デーモン」を参照してください。

- ***AIX*** ユーザーを作成するには、どのようにしますか**?**

  次のコマンドを実行して、AIX ユーザー・アカウント (foo) を作成します。

  ```
  mkuser registry=KRB5files SYSTEM=KRB5files foo
  ```

  mkuser コマンドは AIX にユーザーを作成します。 また、Windows Server Active Directory に AIX アカウント用のユーザーのアカウントも作成する必要があります。 Windows Server Active Directory にユーザー・アカウントを作成すると、暗黙的にプリンシパルが作成されます。
- ***Kerberos*** 認証済みユーザーを除去するには、どのようにしますか**?**

  Kerberos 認証済みユーザーを除去するには、次のコマンドを実行します。

  ```
  rmuser –R KRB5files foo
  ```

  rmuser コマンドは AIX からユーザーを除去します。 また、Windows Server Active Directory から Windows Server のユーザー管理ツールを使用して、ユーザーを除去する必要があります。
- ***Kerberos*** 認証済みユーザーのパスワードを変更するには、どのようにしますか**?**

  Kerberos 認証済みユーザーのパスワードを変更するには、次のコマンドを実行します。

  ```
  passwd –R KRB5files foo
  ```

  KDC が **kpasswd** コマンドをサポートしている場合、 **passwd** コマンドは Kerberos サーバー上の Kerberos プリンシパル・パスワード foo@MYREALM を変更します。
- ユーザーに対してクライアントの期限切れパスワードの変更を許可するには、どのようにしますか**?**

  ユーザーに対してクライアントの期限切れパスワードの変更を許可するには、allow_expired_pwd=yes オプションを `methods.cfg` ファイルに追加します。このオプションが `yes` に設定されていると、パス

ワードが期限切れのユーザーは期限切れパスワードを変更するようにプロンプトが出されます。 このオプションが no に設定されているか、または 存在していない場合は、ユーザーを認証できません。

```
KRB5:
   program = /usr/lib/security/KRB5
   options = authonly,allow_expired_pwd=yes
```

- ***AIX*** ユーザーを ***Kerberos*** 認証済みユーザーに変換するには、どのようにしますか**?**

  AIX ユーザーを Kerberos 認証済みユーザーに変換するには、以下の手順を実行します。

  1. 次のコマンドを実行して、ユーザーが Windows Server Active Directory にアカウントを持っていることを検査します。

     ```
     chuser registry=KRB5files SYSTEM=KRB5files foo
     ```

  2. ユーザーが Active Directory にアカウントを持っていなければ、 Active Directory にアカウントを作成し、**chuser** コマンドを使用して SYSTEM およびレジストリー属性を設定します。 Active Directory のアカウントは AIX ユーザー名と同じユーザー名でないことがあります。 AIX ユーザー名に異なる名前が使用されている場合は、auth_name 属性を使用して、それを Active Directory の名前にマップします。

     ```
     chuser registry=KRB5files SYSTEM=KRB5files auth_name=Christopher chris
     ```

- パスワードを忘れたときは、どのようにしますか**?**

  パスワードを忘れたときは、Active Directory 管理者がパスワードを変更する必要があります。 AIX root ユーザーは Active Directory Kerberos プリンシパルのパスワードを設定できません。

- ***auth_name*** 属性と ***auth_domain*** 属性の目的は何ですか**?**

  注**:** これらの属性はオプションです。 AIX システムが 8 文字を超える長さのユーザー名をサポートしていれば、 auth_name 属性が必要でない場合があります。

  auth_name 属性および auth_domain 属性は、KDC 上で AIX ユーザー名を Kerberos プリンシパル名にマップします。 例えば、AIX ユーザーの chris が auth_name=christopher 属性および auth_domain=SOMEREALM 属性を持っている場合、Kerberos プリンシパル名は christopher@SOMEREALM です。 auth_domain 属性を使用することで、要求はデフォルトのレルム名の代わりに SOMEREALM レルム名に送信されます。 これで、ユーザーの chris は MYREALM レルムの代わりに SOMEREALM レルムに対する認証が可能になります。 この例では、SOMEREALM レルム名を含めるために krb5.conf ファイルを変更することも必要になります。

- 標準 ***AIX*** 認証を使用すれば、***Kerberos*** 認証済みユーザーを認証できますか**?**

  はい、以下の手順を実行して、標準 AIX 認証で Kerberos 認証済みユーザーを認証できます。

  1. **passwd** コマンドを使用して、AIX パスワード (/etc/security/passwd) を設定します。

     ```
     passwd -R files foo
     ```

  2. ユーザーのレジストリーおよび SYSTEM 属性を変更します。

     ```
     chuser -R KRB5files registry=files SYSTEM=compat foo
     ```

     このコマンドは認証を Kerberos から compat (crypt サブルーチンを使用する) に変更します。 次回のログインではユーザーの foo が試行され、ローカル・パスワードは /etc/security/passwd ファイルから使用されます。

     また、Kerberos 認証が失敗したときにローカル認証を許可するために、次のように SYSTEM 属性を変更して、 crypt 認証をバックアップ手段として使用することもできます。

     ```
     chuser -R KRB5files SYSTEM="KRB5files or compat" foo
     ```

- ***Windows Server 2000 Kerberos*** サービスを使用しているとき、 ***AIX*** に ***Kerberos*** サーバーをセットアップする必要がありますか**?**

  いいえ、Kerberos サーバー (KDC) を AIX クライアントに構成する必要はありません。ユーザーは Active Directory KDC に対して認証しているからです。何か別の目的で、AIX ネットワーク認証サービス KDC を Kerberos サーバーとして使用する計画を立てている場合は、Kerberos サーバーを構成する必要があります。
- ***AIX*** がパスワードを受け入れないときは、どうすればいいですか**?**

  AIX がパスワードを受け入れないときは、以下の手順を実行します。
  - クライアントは Windows 2000 Active Directory Server と通信していることを確認します。
  - パスワードが AIX と Windows Server 2000 Active Directory の両方の要求と一致していることを確認します。 AIX におけるパスワード・ポリシー・ルールの変更については、「表示ポリシーの変更」を参照してください。

  注**:** ユーザーは Windows Server 2003 Kerberos サービスのパスワードを変更できません。
- システムにログインできない場合は、どうすればいいですか**?**

  システムにログインできない場合は、次の手順を実行します。
  - Windows システムでは、次の手順を実行して KDC が実行していることを確認します。
    1. 「コントロール パネル」で、「管理ツール」アイコンを選択します。
    2. 「サービス」アイコンを選択します。
    3. 「Kerberos 鍵配布センター」が「開始」状態であることを確認します。
  - AIX システムで、/etc/krb5/krb5.conf ファイルが正しい KDC を指していること、およびそれが有効なパラメーターであることを確認します。
  - AIX システムで、client-keytab ファイルにホスト・キーが含まれていることを確認します。 例えば、デフォルトの Keytab ファイルが /etc/krb5/krb5.keytab であれば、以下のように実行します。

    ```
    ktutil
    rkt /etc/krb5/krb5.keytab
    l
    ```
  - Keytab ファイルにある **kvno** コマンドの出力結果が **Ktpass** コマンドの出力結果と一致していることを確認します。
  - auth_name 属性および auth_domain 属性が設定されている場合、それらは Active Directory KDC 上の有効なプリンシパル名を参照していることを確認します。
  - Kerberos ログイン用に SYSTEM 属性が設定されていることを確認します。
  - パスワードが期限切れでないことを確認します。
- チケット許可チケット検査を使用不可にするには、どのようにしますか**?**

  チケット許可チケット検査は、次のように KRB5 スタンザの下で /usr/lib/security/methods.cfg ファイルに 1 つのオプションを指定すると使用不可にできます。

```
KRB5:
   program = /usr/lib/security/KRB5
   options = tgt_verify=no
KRB5files:
   options = db=BUILTIN,auth=KRB5
```

tgt_verify オプションに設定可能な値は、チケット許可チケット検査を使用不可にする場合は no または false、使用可能にする場合は yes または true です。 デフォルトでは、チケット許可チケット検査は使用可能になっています。 tgt_verify オプションを no に設定すると、チケット許可チケット検査は使用不可になり、ホスト・プリンシパル・キーを転送する必要はありません。この変更では、認証目的での keytab ファイルの必要性がなくなるに過ぎません。 その他の Kerberos 対応アプリケーションでは、ホスト・プリンシパルおよびサービス・プリンシパルには Keytab ファイルが必要になる場合もあります。

- ホスト名が解決されなくて、しかも、完全修飾ホスト名が失敗したためにログインできない場合は、どうすればいいですか**?**

  チケット許可チケット検査では、host/<host_name> プリンシパルが KDC に作成されている必要があります。 このホスト名は認証が実行されるクライアントの完全修飾名です。 クライアント・システムは、ホスト・プリンシパル名の host/<host_name> を使用してサービス・チケットを要求します。一部の構成では、クライアント・マシンは完全修飾ホスト名を取得できず、代わりにショート・ネームを取得します。 そのような場合は不一致が起こり、チケット許可チケット検査は失敗して、ログインは失敗します。 例えば、/etc/hosts がショート・ネームのみを保持し、 /etc/netsvc.conf ファイルに hosts=local,bind が指定されている場合、ネーム・レゾリューションはショート・ネームを返します。

  ネーム・レゾリューション問題を訂正するには、次のうちのどれかを実行します。

  – 完全修飾ホスト名が返されるように、/etc/netsvc.conf ファイルのネーム・レゾリューションの順序を変更します。 netsvc.conf ファイルには、ホスト名と別名を解決するための順次配列を指定します。

    次の例では、リゾルバーは BIND サービスを使用してホスト名を解決しています。 BIND サービスが失敗した場合、リゾルバーは /etc/hosts ファイルを使用します。両方の方式が失敗すると、リゾルバーは nis を使用します。

    ```
    hosts=bind,local,nis
    ```

    検索順序で使用される最初の方式が local でなければならない場合は、 /etc/hosts ファイルのショート・ネーム (myhost) を完全修飾ホスト名 (myhost.austin.ibm.com) に変更します。

  – チケット許可チケット検査が要求されない場合は、「チケット許可チケット検査を使用不可にするには、どのようにしますか?」のチケット許可チケット検査を使用不可にする手順に従ってください。

- *AIX* 以外の *Kerberos* サーバーで *Kerberos* ユーザー・パスワードの有効期限が切れたとき、*passwdexpired* サブルーチンが *0* を戻すのはなぜですか*?*

  passwdexpired サブルーチンが 0 を戻すのは、AIX 以外の Kerberos サーバーからは、**kadmin** インターフェースの互換性がないか使用できないために、パスワードの有効期限情報を直接取得できないからです。

  methods.cfg ファイル内で **allow_expired_pwd** フラグを使用すると、AIX は Kerberos 認証インターフェースを使用してパスワードの有効期限情報を取得できます。 パスワードの有効期限情報の実際の状況は、ログイン時に取得されるか、**authenticate** サブルーチンと **passwdexpired** サブルーチンを呼び出すことによって取得されます。

Kerberos モジュール

Kerberos モジュールは、NFS クライアントならびにサーバー・コードにより使用される、カーネル・エクステンションです。 これを使用すると、NFS クライアントおよびサーバー・コードは、**gss** デーモンを呼び出すことなく、Kerberos のメッセージ整合性およびプライバシー機能を処理できるようになります。

Kerberos モジュールは **gss** デーモンによってロードされます。 使用されるメソッドはネットワーク認証サービスのバージョン 1.2 をベースにしており、一方、ネットワーク認証サービスは MIT Kerberos をベースにしています。

Kerberos モジュールのロケーションは **/usr/lib/drivers/krb5.ext** です。

関連情報は **gss** デーモンを参照してください。

関連情報:

AIX 向け IBM ネットワーク認証サービスおよび関連テクノロジーに関する IBM developerWorks リソース

Remote Authentication Dial In User Service サーバー

IBM の Remote Authentication Dial-In User Service (RADIUS) は、認証、許可、およびアカウンティングを行うために設計されたネットワーク・アクセス・プロトコルです。これはポート・ベースのプロトコルで、ネットワーク・アクセス・サーバー (NAS) と認証およびアカウンティング・サーバーの間の通信を定義します。

NAS は RADIUS のクライアントとして動作します。このクライアントと RADIUS サーバーの間のトランザクションは、共有秘密鍵 を使用して認証されます。この共有秘密鍵はネットワークで送信されません。 クライアントと RADIUS サーバー間のユーザー・パスワードはすべて暗号化されます。

クライアントは指定の RADIUS サーバーにユーザー情報を渡し、戻された応答に基づいて行動する責任を負っています。 RADIUS サーバーは、ユーザー接続要求を受信してユーザーを認証し、さらにクライアントがユーザーにサービスを配信するために必要なすべての構成情報を戻す責任を負っています。 RADIUS サーバーは、拡張プロキシー情報が構成されている場合、他の RADIUS サーバーのプロキシー・クライアントとして機能することができます。 RADIUS は、トランスポート・プロトコルとしてユーザー・データグラム・プロトコル (**UDP**) を使用します。

RADIUS 認証および許可プロトコルは、IETF RFC 2865 標準に基づいています。 このサーバーは、RFC 2866 に定義されているアカウンティング・プロトコルも提供します。 このほかにサポートされている標準は、RFC 2284 (EAP)、RFC 2869 の一部、RFC 2882 のパスワード有効期限メッセージ、MD5-Challenge、および TLS です。 これらの RFC について詳しくは、以下のリンクを参照してください。

IETF RFC 2865
http://www.ietf.org/rfc/rfc2865.txt

RFC 2866
http://www.ietf.org/rfc/rfc2866.txt

RFC 2284
http://www.ietf.org/rfc/rfc2284.txt

RFC 2869
http://www.ietf.org/rfc/rfc2869.txt

RFC 2882
http://www.ietf.org/rfc/rfc2882.txt

http://www.ietf.org でもこれらの RFC 標準をすべて表示することができます。

RADIUS サーバーのインストール

RADIUS サーバーは、**installp** コマンドまたは SMIT を使用してインストールできます。 RADIUS ソフトウェアは AIX の基本メディアに収められており、イメージ名は radius.base と bos.msg.<lang>.rte です。

ユーザー名とパスワードを保管するための情報データベースとして LDAP ディレクトリーを使用する予定がある場合は、ldap.server をインストールする必要があります。 **installp** ソフトウェアは、RADIUS サーバーがインストールされているシステムごとにインストールする必要があります。

EAP-TLS 認証の使用を計画する場合は (例えば、無線ネットワーク上でデジタル証明書を認証する場合)、OpenSSL 0.9.7 以降もインストールする必要があり、`/etc/radius/radiusd.conf` 構成ファイル内の libssl.a ライブラリーには絶対パスを指定します。

RADIUS デーモンは **radiusctl** コマンドを使用して開始できます。開始時に、以下の各項目には実行中の radiusd プロセスが複数あります。

- 許可
- アカウンティング
- 他のデーモンのモニター

リブートすると、RADIUS が EAP-TLS 用に構成済みでなければ、これらのデーモンは実行レベル 2 で自動的に開始されます。

このルーチンを変更するには、**/etc/rc.d/rc2.d/Sradiusd** ファイルを変更します。

注**:** RADIUS が EAP-TLS を使用してデジタル証明書を認証するように構成されている場合、自動的に開始するようにデーモンを構成できません。それは、管理者が証明書パスフレーズを入力する必要があり、**radiusctl** コマンドを使用して RADIUS の開始および再始動を手動で行う必要があるからです。

RADIUS の停止と再始動

RADIUS サーバーの `/etc/radius/radiusd.conf` 構成ファイル、あるいはデフォルトの許可ファイル `/etc/radius/authorization/default.policy` または `/etc/radius/authorization/default.auth` に変更を加えるときはいつでも **radiusd** デーモンを停止し、再始動する必要があります。この操作は SMIT またはコマンド・ラインから行うことができます。

RADIUS サーバーの始動、再始動、および停止には、次のコマンドを使用します。

```
radiusctl start
radiusctl restart
radiusctl stop
```

RADIUS の停止と開始が必要な理由は、デーモンが、上記の構成ファイルに含まれるすべてのデフォルト属性のメモリー・テーブルを作成する必要があることです。 共有メモリーはローカル・ユーザーごとに使用されますが、ローカル・ユーザー・テーブルは、パフォーマンス上の理由でデーモンの初期化時にしか作成されません。

オンデマンド・フィーチャー**:**

必要な場合は、複数の RADIUS 認証およびアカウンティング・サーバー・デーモンを開始できます。

各サーバーはそれぞれ別個のポートで listen します。 **radiusd.conf** ファイルは、認証の場合はデフォルト・ポート番号 1812、アカウンティングの場合は 1813 を設定した状態で出荷されます。 これらは IANA が割り当てたポート番号です。 **radiusd.conf** を更新すると、これらのポート番号を、必要に応じて

他のポート (複数) と一緒に使用できます。必ず既存のサービスに割り当てられていないポート番号を使用してください。 **radiusd.conf** ファイルの 「**Authentication_Ports**」 および **Accounting_Ports** フィールドに複数のポート番号を入力すると、ポートごとに **radiusd** デーモンが開始されます。 このデーモンはそれぞれのポート番号で listen します。

RADIUS 構成ファイル

RADIUS デーモンはいくつかの構成ファイルを使用します。 これらのファイルのサンプル・バージョンが RADIUS パッケージに入れて提供されています。

すべての構成ファイルは、root ユーザーと security グループによって所有されます。ディクショナリー・ファイルを除き、すべての構成ファイルは、SMIT (System Management Interface Tool) を使用して編集することができます。 構成ファイルへの変更を有効にするには、サーバーを再始動する必要があります。

radiusd.conf ファイル**:**

radiusd.conf ファイルには、RADIUS の構成パラメーターが含まれています。

デフォルトでは、RADIUS は、/etc/radius ディレクトリーで radiusd.conf ファイルを検索します。 構成ファイルのエントリーは、ファイルに示されているフォーマットでなければなりません。 RADIUS は有効なキーワードと値しか受け入れないため、有効なキーワードまたは値が使用されていない場合はデフォルトを使用します。 RADIUS デーモンを起動するときは、SYSLOG 出力を検査して、構成パラメーターのエラーがないか確認します。 すべての構成パラメーターのエラーがサーバーを停止させるとは限りません。

このファイルは認証およびアカウンティング・サーバーの動作に影響を与えるため、適切に読み取り保護および書き込み保護する必要があります。また、このファイル内に機密データが存在する可能性もあります。

重要**: radiusd.conf** ファイルを編集する場合は、エントリーの順序を変更しないでください。 SMIT パネルはその順序に依存しています。

以下に示すのは、radiusd.conf ファイルの例です。

```
#------------------------------------------------------------------#
#              CONFIGURATION FILE                                  #
#                                                                  #
#  By default RADIUS will search for radiusd.conf in the           #
#  /etc/radius directory.                                          #
#                                                                  #
#  Configuration file entries need to be in the below              #
#  formats.  RADIUS will accept only valid "Keyword : value(s)",   #
#  and will use defaults, if "Keyword : value(s)" are not          #
#  present or are in error.                                        #
#                                                                  #
#  It is important to check the syslog output when launching       #
#  the radius daemons to check for configuration parameter         #
#  errors. Once again, not all configuration errors will lead to   #
#  the server stopping.                                            #
#                                                                  #
#  Lastly, this file should be appropriately read/write protected, #
#  because it will affect the behavior of authentication and       #
#  accounting, and confidential or secretive material may          #
#  exist in this file.                                             #
#                                                                  #
#  IF YOU ARE EDITING THIS FILE, DO NOT CHANGE THE ORDER OF THE    #
#  ENTRIES IN THIS FILE. SMIT PANELS DEPEND ON THE ORDER.          #
#                                                                  #
#                                                                  #
```

```
#-------------------------------------------------------------------#

#-------------------------------------------------------------------#
#             Global Configuration                                  #
#                                                                   #
#   RADIUSdirectory  :  This is the base directory for the RADIUS   #
#                       daemon. The daemon will search this         #
#                       directory for further configuration files.  #
#                                                                   #
#   Database_location : This is the value of where the              #
#                       authentication (user ids & passwords)       #
#                       will be stored and retrieved.               #
#                       Valid values: Local, LDAP, UNIX             #
#                       UNIX  - User defined in AIX system          #
#                       Local - Local AVL Database using raddbm     #
#                       LDAP  - Central Database                    #
#                                                                   #
#   Local_Database   :  This indicates the name of the local        #
#                       database file to be used.                   #
#                       This field must be completed if the         #
#                       Database location is Local.                 #
#                                                                   #
#   Debug_Level      :  This pair sets the debug level at which     #
#                       the RADIUS server will run.  Appropriate    #
#                       values are 0,3 or 9. The default is 3.      #
#                       Output is directed to location specified    #
#                       by *.debug stanza in /etc/syslog.conf       #
#                                                                   #
#                       Each level increases the amount of messages#
#                       sent to syslog. For example "9" includes    #
#                       the new messages provided by "9" as well    #
#                       as all messages generated by level 0 and 3.#
#                                                                   #
#                       0 : provides the minimal output to the      #
#                           syslogd log.  It sends start up         #
#                           and end messages for each RADIUS        #
#                           process.  It also logs error            #
#                           conditions.                             #
#                                                                   #
#                       3 : includes general ACCESS ACCEPT, REJECT  #
#                           and DISCARD messages for each packet.   #
#                           This level provides a general audit     #
#                           trail for authentication.               #
#                                                                   #
#                       9 : Maximum amount of log data. Specific    #
#                           values of attributes while a            #
#                           transaction is passing thru             #
#                           processing and more.                    #
#                           [NOT advised under normal operations]   #
#                                                                   #
#-------------------------------------------------------------------#
RADIUSdirectory   : /etc/radius
Database_location : UNIX
Local_Database    : dbdata.bin
Debug_Level       : 3
#-------------------------------------------------------------------#
#                Accounting Configuration                           #
#                                                                   #
#  Local_Accounting :  When this flag is set to ON or TRUE a file   #
#                      will contain a record of ACCOUNTING START    #
#                      and STOP packets received from the Network   #
#                      Access Server(NAS).  The default log file    #
#                      is:                                          #
#                      /var/radius/data/accounting                  #
#                                                                   #
#  Local_accounting_loc  : /var/radius/data/accounting              #
#                        path and file name of the local            #
```

```
#                        accounting data file. Used only if Local_ #
#                        Accounting=ON. If the default is          #
#                        changed, then the path and file need to   #
#                        to be created (with proper permissions)   #
#                        by the admin.                             #
#                                                                  #
#------------------------------------------------------------------#
Local_Accounting     : ON
Local_Accounting_loc : /var/radius/data/accounting
#------------------------------------------------------------------#
#     Reply Message Attributes                                     #
#                                                                  #
#     Accept_Reply-Message : Sent when the RADIUS server           #
#                            replies with an Access-Accept packet  #
#                                                                  #
#     Reject_Reply-Message : Sent when the RADIUS server           #
#                            replies with an Access-Reject packet  #
#                                                                  #
#     Challenge_Reply-Message   : Sent when the RADIUS server      #
#                                 replies with an Access-Challenge #
#                                 packet                           #
#------------------------------------------------------------------#
Accept_Reply-Message :
Reject_Reply-Message :
Challenge_Reply-Message :
Password_Expired_Reply-Message :
#------------------------------------------------------------------#
#    Support Renewal of Expired Password                           #
#                                                                  #
#    Allow_Password_Renewal: YES or NO                             #
#                            Setting this attribute to YES allows  #
#                            users to update their expired password#
#                            via the RADIUS protocol. This requires#
#                            the hardware support of               #
#                            Access-Password-Request packets.      #
#------------------------------------------------------------------#
Allow_Password_Renewal  : NO
#------------------------------------------------------------------#
#    Require Message Authenticator in Access-Request               #
#                                                                  #
#    Require_Message_Authenticator: YES or NO                      #
#                                   Setting this attribute to YES  #
#                                   checks message authenticator   #
#                                   in Access-Request packet.If not#
#                                   present, it will discard the   #
#                                   packet.                        #
#------------------------------------------------------------------#
Require_Message_Authenticator : NO
#------------------------------------------------------------------#
#       Servers ( Authentication and Accounting )                  #
#                                                                  #
#   Authentication_Ports : This field indicates on which port(s)   #
#                          the authentication server(s) will listen#
#                          on.  If the field is blank an           #
#                          authentication daemon will not be       #
#                          started.                                #
#                          The value field may contain more than   #
#                          one value.  Each value is REQUIRED to   #
#                          be separated by a comma ','.            #
#                                                                  #
#                          The value field must contain a numeric  #
#                          value, like "6666".  In this case a     #
#                          server daemon will listen on "6666".    #
#                                                                  #
#   Accounting_Ports     : The same as authentication_Ports.  See  #
#                          above definitions.                      #
#                                                                  #
```

```
# [NOTE] There is no check for port conflicts.  If a server is     #
#        currently running on the specified port the deamon will   #
#        error and not run.  Be sure to check the syslog output    #
#        insure that all servers have started without incident.    #
#                                                                  #
#                                                                  #
# [Example]                                                        #
#   Authentication_Ports  : 1812,6666 (No Space between commas)    #
#                                                                  #
#   In the above example a sever will be start for each port       #
#   specified.  In the case                                        #
#                                                                  #
#           6666 : port 6666                                       #
#                                                                  #
#------------------------------------------------------------------#
Authentication_Ports  : 1812
Accounting_Ports      : 1813
#------------------------------------------------------------------#
#       LDAP Directory User Information                            #
#                                                                  #
#   Required if RADIUS is to connect to a LDAP Version 3 Directory #
#   and the Database_location field=LDAP                           #
#                                                                  #
#   LDAP_User     : User ID which has admin permission to connect  #
#                   to the remote (LDAP) database. This is the     #
#                   the LDAP administrator's DN.                   #
#                                                                  #
#   LDAP_User_Pwd : Password associated with the above User Id     #
#                   which is required to authenticate to the LDAP  #
#                   directory.                                     #
#                                                                  #
#------------------------------------------------------------------#
LDAP_User        : cn=root
LDAP_User_Pwd    :
#------------------------------------------------------------------#
#       LDAP Directory Information                                 #
#                                                                  #
#   If the Database_location field is set to "LDAP" then the       #
#   following fields need to be completed.                         #
#                                                                  #
#   LDAP_Server_name     : This field specifies the fully qualified#
#                          host name where the LDAP Version 3      #
#                          Server is located.                      #
#   LDAP_Server_Port     : The TCP port number for the LDAP server #
#                          The standard LDAP port is 389.          #
#   LDP_Base_DN          : The distinguished name for search start #
#   LDAP_Timeout         : # seconds to wait for a response from   #
#                          the LDAP server                         #
#   LDAP_Hoplimit        : maximum number of referrals to follow   #
#                          in a sequence                           #
#   LDAP_Sizelimit       : size limit (in entries) for search      #
#   LDAP_Debug_level     : 0=OFF 1=Trace ON                        #
#                                                                  #
#------------------------------------------------------------------#
LDAP_Server_name          :
LDAP_Server_port          :  389
LDAP_Base_DN              :  cn=aixradius
LDAP_Timeout              :  10
LDAP_Hoplimit             :  0
LDAP_Sizelimit            :  0
LDAP_Debug_level          :  0
#------------------------------------------------------------------#
#       PROXY RADIUS Information                                   #
#                                                                  #
#                                                                  #
#   Proxy_Allow               :  ON or OFF. If ON, then the server #
#                                can proxy packets to realms it    #
```

```
#                                knows of and the following        #
#                                fields must also be configured.   #
#   Proxy_Use_Table           :  ON or OFF. If ON, then the server #
#                                can use table for faster          #
#                                processing of duplicate requests  #
#                                Can be used without proxy ON, but #
#                                it is required to be ON if        #
#                                Proxy_Use_Table is set to ON.     #
#   Proxy_Realm_name          :  This field specifies the realm    #
#                                this server services.             #
#   Proxy_Prefix_delim        :  A list of separators for parsing  #
#                                realm names added as a prefix to  #
#                                the username.  This list must be  #
#                                mutually exclusive to the Suffix  #
#                                delimiters.                       #
#   Proxy_Suffix_delim        :  A list of separators for parsing  #
#                                realm names added as a suffix to  #
#                                the username.  This list must be  #
#                                mutually exclusive to the Prefix  #
#                                delimiters.                       #
#   Proxy_Remove_Hops         :  YES or NO.  If YES then the       #
#                                will remove its realm name, the   #
#                                realm names of any previous hops  #
#                                and the realm name of the next    #
#                                server the packet will proxy to.  #
#                                                                  #
#   Proxy_Retry_count         :  The number of times to attempt    #
#                                to send the request packet.       #
#                                                                  #
#   Proxy_Time_Out            :  The number of seconds to wait     #
#                                in between send attempts.         #
#                                                                  #
#------------------------------------------------------------------#
Proxy_Allow               :   OFF
Proxy_Use_Table           :   OFF
Proxy_Realm_name          :
Proxy_Prefix_delim        :   $/
Proxy_Suffix_delim        :   @.
Proxy_Remove_Hops         :   NO
Proxy_Retry_count         :   2
Proxy_Time_Out            :   30
#------------------------------------------------------------------#
#    Local Operating System Authentication Configuration           #
#                                                                  #
#  UNIX_Check_Login_Restrictions  : ON or OFF. If ON, during       #
#                                   local operating system authen- #
#                                   tication, a call to            #
#                                   loginrestrictions() will be    #
#                                   made to verify the user has    #
#                                   no local login restrictions.   #
#                                                                  #
#------------------------------------------------------------------#
UNIX_Check_Login_Restrictions : OFF
#------------------------------------------------------------------#
#    Global IP Pooling Flag                                        #
#                                                                  #
#  Enable_IP_Pool : ON or OFF. If ON, then RADIUS Server will do   #
#                   IP address assignment from a pool of addresses #
#                   defined to the RADIUS server.                  #
#                                                                  #
#------------------------------------------------------------------#
Enable_IP_Pool            :   OFF
#------------------------------------------------------------------#
# Send Accept MA: ON or OFF. Some NAS's dislike it if Message      #
#                 Authenticators (MA's) are present in an ACCEPT   #
#                 message. Use this option to disable sending MA   #
#                 when sending an ACCEPT.                          #
```

```
#                                                                     #
# NOTE: Sometimes these same NAS's do not like custom ACCEPT        #
# messages either.                                                    #
#                                                                     #
#---------------------------------------------------------------------#
Send_Accept_MA : ON
#---------------------------------------------------------------------#
#                                                                     #
# Maximum_Threads : The number of threads that will get             #
#                   spawned to handle authentication                #
#                   requests. If nothing is specified               #
#                   RADIUS defaults to 10.                          #
#                                                                     #
#---------------------------------------------------------------------#
Maximum_Threads : 99
#---------------------------------------------------------------------#
#                                                                     #
# EAP_Conversation_Timeout : The number of seconds to wait          #
#                            before a conversation becomes          #
#                            stale and gets deleted.                #
#                                                                     #
# NOTE: This prevents Denial-of-Service (DoS) attacks on the        #
#       RADIUS Authentication Server. You may need to increase      #
#       the value of this timeout if your network has high          #
#       latency.                                                    #
#                                                                     #
#---------------------------------------------------------------------#
EAP_Conversation_Timeout : 30
#---------------------------------------------------------------------#
# Global EAP-TLS (eap-tls) Configuration Settings:                  #
#                                                                     #
# Examples:                                                           #
#                                                                     #
# Enable_EAP-TLS : ON or OFF. If ON, then the server                #
#                  can use OpenSSL to authenticate users            #
#                  using EAP-TLS. These users must first            #
#                  have an EAP authentication type of 13            #
#                  (or EAP-TLS). This setting is found in           #
#                  smitty, using: 'smitty rad_conf_users'           #
#                                                                     #
# NOTE: The following attributes below are completely ignored       #
#       if the above 'Enable_EAP' attribute is not 'ON'.            #
#                                                                     #
# OpenSSL_Library : /opt/freeware/lib/libssl.a(libssl.so.0.9.7)     #
# OpenSSL_Ciphers : ALL:!ADH:RC4+RSA:+SSLv2:@STRENGTH               #
# RootCA_Dir          : /etc/radius/tls                             #
# RootCA_File         : /etc/radius/tls/cacert.pem                  #
# Server_Cert_File    : /etc/radius/tls/cert-srv.pem                #
# Server_PrivKey_File : /etc/radius/tls/cert-srv.pem                #
# Server_CRL_File     : /etc/radius/tls/crl.pem                     #
#                                                                     #
# NOTE: Server_Cert_File and Server_PrivKey_File can be the         #
#       same file if the file is of the following format (but       #
#       in any order):                                              #
#                                                                     #
#       -----BEGIN RSA PRIVATE KEY-----                             #
#       Proc-Type: 4,ENCRYPTED                                      #
#       <rsa private key data here>                                 #
#       -----END RSA PRIVATE KEY-----                               #
#       -----BEGIN CERTIFICATE-----                                 #
#       <certificate data here>                                     #
#       -----END CERTIFICATE-----                                   #
#                                                                     #
#---------------------------------------------------------------------#
Enable_EAP-TLS      : ON
OpenSSL_Library     : /opt/freeware/lib/libssl.a(libssl.so.0.9.7)
OpenSSL_Ciphers     : ALL:!ADH:RC4+RSA:+SSLv2:@STRENGTH
```

```
RootCA_Dir          : /etc/radius/tls
RootCA_File         : /etc/radius/tls/radiusdcacert.pem
Server_Cert_File    : /etc/radius/tls/cert-srv.pem
Server_PrivKey_File : /etc/radius/tls/cert-srv.pem
Server_CRL_File     :
```

各ユーザーの EAP 認証方式は、SMIT を使用して設定できます。 各ユーザーの EAP 方式を設定するには、以下のステップを実行してください。

```
Radius サーバー
     -> ユーザーの構成
          -> ローカル・データベース
             LDAP ディレクトリー
                  -> ユーザーの追加
                     ユーザーの特性の変更/表示
                          ->
                          ログイン・ユーザー ID [ ]
                          EAP タイプ [0 2 4]
                          パスワードの最大経過日数
```

EAP タイプが選択されると、以下の選択が使用可能になります。

0 なし

2 MD5 - Challenge

4 TLS

選択された EAP 方式は、認証を実行するために `radiusd.conf` ファイルに設定されている認証方式のシーケンスと比較されます。

/etc/radius/clients ファイル:

`clients` ファイルには、RADIUS サーバーへの要求を作成できるクライアントのリストが含まれます。

通常、クライアント、NAS、または AP ごとに、クライアント IP アドレスと共に、RADIUS サーバーとクライアント間の共有秘密鍵を入力する必要があり、またオプションで IP プーリング用の *poolname* を入力します。

このファイルは、次の形式のエントリーで構成されています。

```
<Client IP Address>    <Shared Secret>    <Pool Name>
```

サンプルのエントリー・リストは次のように表示されます。

```
10.10.10.1      mysecret1      floor6
10.10.10.2      mysecret2      floor5
```

共有秘密鍵は、クライアント・ハードウェアと RADIUS サーバーの両方で構成されている文字列です。 共有秘密鍵の最大長は 258 バイトで、大文字小文字の区別があります。 共有秘密鍵は、RADIUS パケットで送信されることはなく、ネットワークを通して送信されることもありません。 システム管理者は、両方のサイド (クライアントと RADIUS サーバー) で厳格に秘密が保たれるように構成する必要があります。 共有秘密鍵は、ユーザー・パスワードの暗号化に使用され、また、メッセージ認証属性を使用するメッセージ整合性の検査にも使用することができます。

各クライアントの共有秘密鍵は **/etc/radius/clients** ファイル内で固有でなければなりません。良いパスワードの場合と同様に、大文字/小文字、数字、記号を混用するのが最良です。 共有秘密鍵は、機密保護のために、少なくとも 16 文字の長さにします。 **/etc/radius/clients** ファイルは SMIT を使用して変更できます。 辞書アタックを防ぐために、共有秘密鍵は頻繁に変更してください。

poolname は、動的変換時にグローバル IP アドレスの割り当て元となるプールの名前です。 システム管理者は、RADIUS サーバーをセットアップするときに *poolname* を作成します。 *poolname* は、SMIT パネルを使用して、「**Configure Proxy Rules (**プロキシー規則の構成**)**」 > 「**IP Pool (IP** プール**)**」 > 「**Create an IP Pool (IP** プールの作成**)**」から追加します。 この名前は、サーバー側の IP プーリング時に使用されます。

/etc/radius/dictionary ファイル**:**

dictionary ファイルには、RADIUS プロトコルによって定義され、AIX RADIUS サーバーによってサポートされる属性の記述が含まれます。

このファイルは、パケット・データを検証および作成するときに RADIUS デーモンが使用します。 ベンダー固有属性もここに追加してください。 dictionary ファイルは、どのエディターでも変更できます。 SMIT のインターフェースはありません。

以下は、サンプルの dictionary ファイルの一部です。

```
#########################################################################
#                                                                       #
#   This file contains dictionary translations for parsing              #
#   requests and generating responses.  All transactions are            #
#   composed of Attribute/Value Pairs.  The value of each attribute     #
#   is specified as one of 4 data types.  Valid data types are:         #
#                                                                       #
#   string - 0-253 octets                                               #
#   ipaddr - 4 octets in network byte order                             #
#   integer - 32 bit value in big endian order (high byte first)        #
#   date - 32 bit value in big endian order - seconds since             #
#                                   00:00:00 GMT,  Jan.  1,  1970       #
#                                                                       #
#   Enumerated values are stored in the user file with dictionary       #
#   VALUE translations for easy administration.                         #
#                                                                       #
#   Example:                                                            #
#                                                                       #
#   ATTRIBUTE         VALUE                                             #
#   ---------------   -----                                             #
#   Framed-Protocol = PPP                                               #
#   7               = 1     (integer encoding)                          #
#                                                                       #
#########################################################################
ATTRIBUTE       User-Name                       1       string
ATTRIBUTE       User-Password                   2       string
ATTRIBUTE       CHAP-Password                   3       string
ATTRIBUTE       NAS-IP-Address                  4       ipaddr
ATTRIBUTE       NAS-Port                        5       integer
ATTRIBUTE       Service-Type                    6       integer
ATTRIBUTE       Framed-Protocol                 7       integer
ATTRIBUTE       Framed-IP-Address               8       ipaddr
ATTRIBUTE       Framed-IP-Netmask               9       ipaddr
ATTRIBUTE       Framed-Routing                  10      integer
ATTRIBUTE       Filter-Id                       11      string
.
.
.
```

注**:** default.policy ファイルまたは default.auth ファイル内で (または特定の user_id.policy または user_id.auth ファイルに対して) 定義された属性は、ローカル AIX ディクショナリー構成ファイルで定義済みの有効な RADIUS 属性でなければなりません。 ディクショナリー内に属性が見つからない場合、**radiusd** デーモンはロードされず、エラー・メッセージが記録されます。

注**:** システムのディクショナリー、default.policy ファイル、および default.auth ファイルを変更した場合は、**stopsrc** コマンドと **startsrc** コマンドを実行するか、SMIT を使用して RADIUS デーモンを再始動する必要があります。

/etc/radius/proxy ファイル**:**

/etc/radius/proxy ファイルは、プロキシー・フィーチャーをサポートする構成ファイルです。 このファイルは、プロキシー・サーバーがパケットを転送できる既知のレルムをマップします。

/etc/radius/proxy ファイルは、そのレルムおよび 2 つのサーバー間の共有秘密鍵に関してパケットを処理するサーバーの IP アドレスを使用します。

このファイルには、SMIT を使用して変更することのできる以下のフィールドが含まれています。

- レルム名
- ネクスト・ホップの **IP** アドレス
- 共有秘密鍵

以下に、/etc/radius/proxy ファイルの例を示します。

注**:**

共有秘密鍵は長さは 16 文字にしてください。 同じ共有秘密鍵を RADIUS サーバーのネクスト・ホップでも構成する必要があります。

```
# @(#)91 1.3  src/rad/usr/sbin/config_files/proxy, radconfig, radius530 1/23/04 13:11:14
#####################################################################
#                                                                   #
#       This file contains a list of proxy realms which are         #
#       authorized to send/receive proxy requests/responses to/from #
#       this RADIUS server and their Shared secret used in encryption.#
#                                                                   #
#       The first field is the name of the realm of the remote RADIUS #
#       Server.                                                     #
#                                                                   #
#       The second field is a valid IP address for the remote RADIUS  #
#       Server.                                                     #
#                                                                   #
#       The third column is the shared secret associated with this  #
#       realm.                                                      #
#                                                                   #
#       NOTE: This file contains sensative security information and #
#             precautions should be taken to secure access to this  #
#             file.                                                 #
#                                                                   #
#####################################################################
# REALM NAME              REALM IP              SHARED SECRET
#----------------       ------------------     ----------------------
#  myRealm                10.10.10.10             sharedsec
```

認証

従来の認証では名前と固定パスワードが使用され、一般に、ユーザーが最初にマシンにログインするとき、またはサービスを要求するときに認証が行われていました。 RADIUS は認証データベースに依存して、ユーザー ID、パスワード、およびその他の情報を保管しています。

ユーザー認証を行う場合、このサーバーはローカル・データベース、UNIX パスワード、または LDAP を使用できます。 データベースのロケーションは、セットアップ時にサーバーの **/etc/radius/radiusd.conf**

ファイルで構成するか、SMIT でこのファイルを更新して構成します。 RADIUS 構成ファイルについて詳しくは、 354 ページの『RADIUS 構成ファイル』を参照してください。

ユーザー・データベース**:**

RADIUS ソフトウェアは、さまざまなデータベースを使用してユーザー情報を保管することができます。

ローカル、UNIX、または LDAP データベースを使用してユーザー情報を保管することができます。

UNIX:

UNIX 認証オプションを使用すると、RADIUS でローカル・システム認証方式を使用してユーザーを認証できます。

ローカル UNIX 認証を使用するには、**radiusd.conf** ファイルの **database_location** フィールドを編集するか、SMIT の「Database Location (データベースのロケーション)」フィールドで「UNIX」を選択します。この認証方式は、UNIX **authenticate()** アプリケーション・プログラミング・インターフェース (API) を呼び出して、ユーザー ID とパスワードを認証します。 パスワードは、UNIX が使用するデータ・ファイル (**/etc/passwords** など) と同じファイルに保存されます。ユーザー ID とパスワードは、**mkuser** コマンドまたは SMIT を使用して作成されます。

UNIX データベースを使用するには、次のように「**Database Location**」フィールドで「UNIX」を選択します。

```
    Configure Server

RADIUS Directory             /etc/radius
*Database Location           [UNIX]
Local AVL Database File Name [dbdata.bin]
Local Accounting             [ON]

Debug Level                  [3]
.
.
.
```

ローカル*:*

radiusd.conf ファイルの「**database_location**」フィールドまたは SMIT の「Database Location (データベースのロケーション)」エントリーに「Local (ローカル)」という語が含まれている場合、RADIUS サーバーはすべてのユーザー ID とパスワードのロケーションとして **/etc/radius/dbdata.bin** を使用します。

ローカル・ユーザー・データベースは、ユーザー ID とパスワードの情報を含むフラット・ファイルです。 パスワードはハッシュ・フォーマットで保存されます。 ハッシュは、メモリー・スペース内のデータに直接アクセスするための高速アドレッシング技法です。 ユーザー・パスワードの追加、削除、または変更を行うには、**raddbm** コマンドを実行するか、SMIT を使用します。 **radiusd** デーモンを開始すると、このデーモンは **radiusd.conf** ファイルを読み取り、ユーザー ID とパスワードをメモリーにロードします。

注**:** ユーザー ID の最大長は 253 文字で、パスワードの最大長は 128 文字です。

ローカル・ユーザー・データベースを使用するには、次のように、「**Database Location**」フィールドで「Local」を選択します。

```
    Configure Server

RADIUS Directory             /etc/radius
*Database Location           [Local]
Local AVL Database File Name [dbdata.bin]
Local Accounting             [ON]

Debug Level                  [3]
.
.
.
```

LDAP:

RADIUS は LDAP バージョン 3 を使用してリモート・ユーザー・データを保管できます。

RADIUS は LDAP バージョン 3 の API 呼び出しを使用して、ユーザー・データにリモート側にアクセスします。 LDAP バージョン 3 のアクセスは、**/etc/radiusd.conf** ファイルの **database_location** フィールドが `LDAP` に設定され、サーバー名、LDAP 管理者ユーザー ID、および LDAP 管理者パスワードが構成されている場合に行われます。

AIX は、IBM Tivoli Directory Server でサポートされパッケージされている LDAP バージョン 3 クライアント・ライブラリーを使用します。 LDAP はスケーラブルなプロトコルです。LDAP を使用する利点は、ユーザーと処理中のデータの場所を中央のロケーションで探索できるため、RADIUS サーバーの管理がしやすくなることです。 コマンド・ライン・ユーティリティー **ldapsearch** を使用して、すべての RADIUS データを表示することができます。

また、LDAP を RADIUS 用に使用するには、事前に LDAP の構成と管理を行っておく必要があります。

RADIUS サーバーには、オブジェクト・クラスや属性などの RADIUS スキーマをディレクトリーに追加するための LDAP `ldif` ファイルがありますが、LDAP のセットアップと構成は行う必要があります。

RADIUS LDAP オブジェクトを使用するための RADIUS 専用のサフィックスが別個に作成されます。 このサフィックスは、cn=aixradius という名前のコンテナーであり、 365 ページの『RADIUS LDAP サーバーの構成』で説明されているように 2 つのオブジェクト・クラスを含みます。 RADIUS 提供の `ldif` ファイルを適用すると、このサフィックスと RADIUS スキーマが作成されます。

LDAP を認証データベースとして使用する場合、以下のフィーチャーを利用できます。

1. すべての RADIUS サーバーから表示およびアクセスできるユーザー・データベース
2. アクティブ・ユーザーのリスト
3. ユーザー ID ごとに最大数のログインが可能なフィーチャー
4. ユーザーごとに構成可能な **EAP** タイプ
5. パスワード有効期限日付

LDAP データベースを使用するには、次のように「**Database Location (**データベースのロケーション**)**」フィールドで「`LDAP`」を選択します。

```
    Configure Server

RADIUS Directory             /etc/radius
*Database Location           [LDAP]
Local AVL Database File Name [dbdata.bin]
Local Accounting             [ON]

Debug Level                  [3]
.
.
.
```

関連情報:

IBM Directory Server

***RADIUS LDAP* サーバーの構成:**

LDAP ユーザー認証を構成する場合は、必ず LDAP サーバー・スキーマを更新する必要があります。 LDAP システム管理者は、LDAP RADIUS ユーザーを定義する前に、AIX RADIUS 定義属性とオブジェクト・クラスを追加する必要があります。

LDAP サーバーには必ずサフィックスを追加します。 RADIUS のサフィックス名は `cn=aixradius` です。 サフィックスとは、ディレクトリー階層の最上位のエントリーを識別する識別名です。

サフィックスを追加するときは、LDAP ディレクトリーに空のコンテナーがあります。コンテナーとは、ネームスペースを区画化するときに使用できる空のエントリーです。 コンテナーはファイルシステム・ディレクトリーに似ています。ファイルシステム・ディレクトリーはその下にディレクトリー・エントリーを置くことができます。これで、SMIT を使用してユーザー・プロファイル情報を LDAP ディレクトリーに追加できます。 LDAP 管理者の ID とパスワードは **/etc/radius/radiusd.conf** ファイルに保管され、RADIUS サーバー上で SMIT を使用して構成できます。

LDAP ディレクトリー・エントリーに保管された情報を編成するために、スキーマがオブジェクト・クラスを定義します。 オブジェクト・クラスは、必要属性とオプション属性のセットで構成されます。 属性は**タイプ = 値**のペアの形式になっており、タイプは固有のオブジェクト ID (OID) によって定義され、値には構文が定義されています。 LDAP ディレクトリー内のすべてのエントリーは、それぞれ 1 つのオブジェクトの 1 つのインスタンスです。

注**:** オブジェクト・クラスは単独では、ディレクトリー情報ツリーまたはネームスペースを定義しません。この定義が行われるのは、エントリーが作成され、オブジェクト・クラスの特定のインスタンスに固有の識別名が割り当てられた場合に限ります。 例えば、コンテナー・オブジェクト・クラスに固有の DN が割り当てられた場合、そのクラスは、オブジェクト・クラス組織単位のインスタンスである他の 2 つのエントリーと関連付けることができます。 この結果は、ツリー状の構造またはネームスペースとなります。

オブジェクト・クラスは RADIUS サーバーに固有のものであり、**ldif** ファイルから適用されます。 属性の一部は既存の LDAP スキーマ属性ですが、RADIUS に固有の属性もあります。 新しい RADIUS オブジェクト・クラスは、構造的であり、抽象的です。

セキュリティー上の目的で、LDAP サーバーへのバインドでは単純バインド、または SASL API 呼び出しの **ldap_bind_s** が使用されます。この呼び出しには DN および認証方式としての CRAM-MD5 が組み込まれ、さらに LDAP 管理者パスワードが組み込まれます。 これは、ネットワークを通じてパスワードそのものを伝送するのではなく、メッセージ・ダイジェストを送信します。 CRAM-MD5 は、どちらの側 (クライアントまたはサーバー) にも特別な構成を必要としないセキュリティー・メカニズムです。

注**:** オブジェクト・クラス内のすべての属性は単一値です。

RADIUS LDAP ネームスペース**:**

RADIUS LDAP ネームスペースは、その階層の最上位に cn=aixradius コンテナーを持っています。 cn=aixradius の下には、2 つの組織単位 (OU) があります。 これらの OU は、エントリーを固有にするためのコンテナーです。

次の図は、RADIUS LDAP スキーマを示しています。 この図では、コンテナーと組織単位がすべて円で表され、線または分岐で接続されています。 中央にある aixradius コンテナーは下に向かって 2 つの組織単位、ibm-radiususer と ibm-radiusactiveusers に分岐しています。 ibm-radiususer コンテナーの下には、(暗黙の) userid、password、および maxLogin というコンテナーがあります。 ibmradiusactiveusers コンテナーの下には、(暗黙の) userid +、login number、login status、および session_id コンテナーがあります。 aixradius コンテナーの上には、aixsecurity コンテナーがあり、root コンテナーが一番上です。

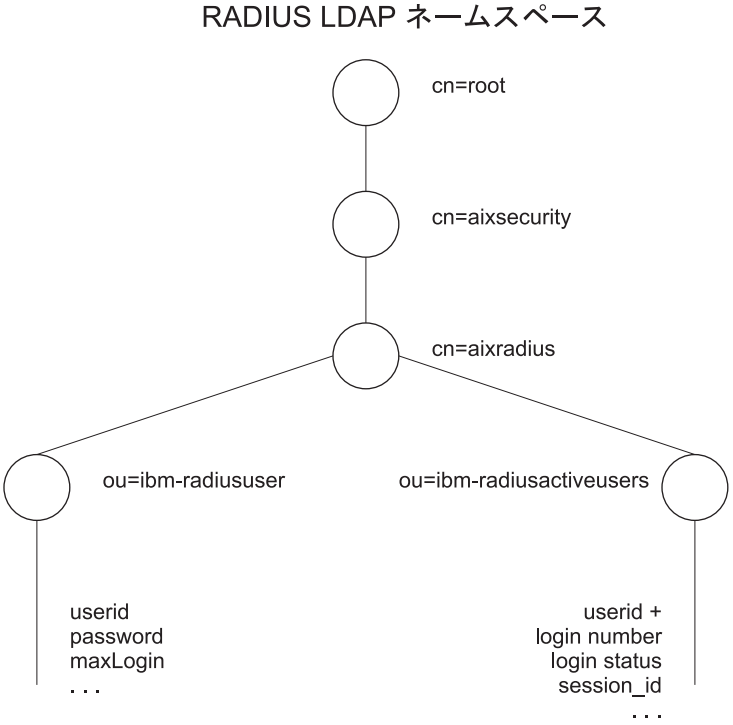


図 *16. RADIUS LDAP* ネームスペース

LDAP ネームスペース・スキーマ・ファイル**:**

LDAP スキーマ・ファイルは、LDAP ネームスペースのためのオブジェクト・クラスと RADIUS 特定の属性を定義します。

以下の LDAP スキーマ・ファイルは、/etc/radius/ldap ディレクトリーにあります。

IBM.V3.radiusbase.schema.ldif

このファイルは、RADIUS サーバーの最上位のオブジェクト・クラスを定義します (cn=aixradius)。さらにこのファイルは、cn=aixradius オブジェクト・クラスの下に以下のような分岐も作成します。

```
ou=ibm-radiususer
ou=ibm-radiusactiveusers
```

次のコマンドを使用すると、必要な情報を追加できます。

```
ldapadd -D ldap_admin_id -w password -i /etc/radius/ldap/IBM.V3.radiusbase.schema.ldif
```

このコマンドは LDAP サーバー・システムで実行するか、**-h** (ホスト・システム名) オプションを使用してリモート側で実行することができます。

IBM.V3.radius.schema.ldif

このファイルは、RADIUS 特定の属性およびオブジェクト・クラスを定義します。

次のコマンドを入力すると、新規の RADIUS 属性とオブジェクト・クラスを追加できます。

```
ldapmodify -D ldap_admin_id -w password -i /etc/radius/ldap/IBM.V3.radius.schema.ldif
```

また、SMIT を使用してデータベースのロケーションとして LDAP と指定し、さらに LDAP サーバー名と管理者パスワードを入力することも必要です。 これを行った後は、SMIT を使用して RADIUS LDAP ユーザーをディレクトリーに追加することができます。

ユーザー・プロファイル・オブジェクト・クラス*:*

LDAP ユーザー・プロファイルをシステムに入力しておかないと、RADIUS サーバーはシステムに対してユーザーを認証できません。 プロファイルにはユーザー ID とパスワードが含まれます。

ユーザー・プロファイル・オブジェクトはネットワークへのアクセスを持つ特定の個人に関するデータを提供し、認証情報を含みます。 **ibm-radiusUserInstance** オブジェクト・クラスは、デーモンからの LDAP API 呼び出しと同期的にアクセスされます。 DN の先頭にある固有フィールドはユーザー ID です。 **MaxLoginCount** フィールドは、LDAP ユーザーがログインできる回数を制限します。

アクティブ・ログイン・リスト・オブジェクト・クラス*:*

LDAP アクティブ・ログイン・リストは、現在ログインしているユーザーに関する情報を含むデータを表します。

ユーザーごとに複数のレコードがあり、開始レコードは `login_number = 1` で、最大数 `MaxLoginCount` は 5 です。セッション ID は RADIUS `start_accounting` メッセージから取られます。 **ibm-radiusUserInstance** オブジェクトが作成されると部分的に完了したレコードが作成されます。 これは、RADIUS アカウンティング・パケットの受信前は、ほとんどのフィールドが空であることを意味します。 RADIUS `start_accounting` メッセージが受信されると、**ibm-radiusactiveusers** オブジェクトはユーザーが現在ログインしていることを指定するように更新され、固有のセッション情報が正しいログイン番号に書き込まれます。 `stop_accounting` メッセージが受信されると、アクティブ・ログイン・リスト・レコード内の情報はクリアされます。 アクティブ・ログイン・レコードは、ユーザーが現在ログオフしていることを反映するように更新されます。 アカウンティングの開始と停止の各メッセージのセッション番号は、同じ固有番号です。 このオブジェクト・クラスは、LDAP API 呼び出しで同期的にアクセスされます。

パスワード認証プロトコル*:*

パスワード認証プロトコル (**PAP**) は、クライアントとサーバーの両方で構成できる 1 つの値の MD5 ハッシュ・アルゴリズムを使用してユーザーのパスワードをコード化することにより、セキュリティーを提供します。

その仕組みは、次のとおりです。

1. ユーザー・パスワードを含むパケットでは、認証フィールドに要求オーセンティケーターと呼ばれる 16 オクテットの乱数が含まれます。
2. 要求オーセンティケーターとクライアントの共有秘密鍵は MD5 ハッシュの中に書き込まれます。 その結果が 16 オクテットのハッシュとなります。
3. ユーザー提供のパスワードは、16 オクテットになるまでヌルで埋め込まれます。
4. ステップ 2 のハッシュが、埋め込まれたパスワードを使用して XOR (排他 OR) 演算されます。 これが、パケットの中で *user_password* 属性として送信されるデータです。
5. RADIUS サーバーはステップ 2 のものと同じハッシュを計算します。
6. このハッシュはステップ 4 のパケット・データを使用して XOR 演算され、それによってパスワードがリカバリーされます。

チャレンジ・ハンドシェーク認証プロトコル**:**

RADIUS は、パスワード保護用に PPP の **CHAP** の使用もサポートします。

CHAP を使用すると、ユーザーのパスワードはネットワークで送信されません。 その代わりに、パスワードの MD5 ハッシュが送信され、RADIUS サーバーが、保管されたパスワードなどのユーザーの情報からそのハッシュを再構築し、それをクライアントが送信した値と比較します。

拡張可能認証プロトコル**:**

拡張可能認証プロトコル (**EAP**) は、複数の認証方式をサポートするために設計されたプロトコルです。

EAP は、クライアントと認証サーバー間の認証通信の構造を指定し、認証データの内容は定義しません。この内容は、認証に使用される特定の **EAP** 方式によって定義されます。 一般的な **EAP** 方式には次のものがあります。

- MD5-challenge
- ワンタイム・パスワード
- 汎用トークン・カード
- Transport Layer Security (TLS)

RADIUS は **EAP** を利用して、RADIUS サーバーとそのクライアント間で **EAP** データを転送するときに使用される RADIUS 属性を指定します。 これで、RADIUS サーバーは、この **EAP** データを、さまざまな **EAP** 認証方式を実装するバックエンド・サーバーに直接送信できるようになります。

AIX RADIUS サーバーは MD5-challenge EAP 方式のみをサポートします。

ユーザー認証に使用される EAP 方式は、ローカル・データベースまたは LDAP 内のユーザーのエントリーに値を設定することにより、ユーザー・レベルで設定できます。

デフォルトでは、各ユーザーの EAP はオフになっています。

許可

RADIUS では、許可ポリシー・ファイルの `default.auth` と `default.policy` に定義された、ユーザーごとの許可属性を使用することができます。

許可属性とは、RFC に指定され、/etc/radius/dictionary ファイルに定義された有効な RADIUS プロトコル属性です。 許可はオプションであり、ハードウェア NAS またはアクセス・ポイントの構成方法によって異なります。 許可属性が必要な場合は、それを構成する必要があります。 許可は、認証が正常に完了した後にしか実行されません。

ポリシーとは構成可能なユーザー属性と値のペアのことで、これによってユーザーがネットワークにアクセスする方法を制御できます。 ポリシーは、RADIUS サーバーにグローバルになるように定義したり、あるいはユーザーに特定になるように定義することができます。

/etc/radius/authorization/default.auth と default.policy の 2 つの許可構成ファイルが出荷されています。 **default.policy** ファイルは、着信した access request パケットとの突き合わせに使用されます。 このファイルに含まれる属性と値のペアは、当初はブランクであり、希望の設定に構成する必要があります。 認証後、このポリシーは、access accept パケットまたは access reject パケットがクライアントに戻されたかどうか確認します。

各ユーザーは、*user_id*.policy ファイルも持つことができます。 ユーザーがその特定のユーザー ID 用に固有のポリシー・ファイルを作成した場合、そのファイルの属性が最初に検査されます。 *user_id*.policy ファイル内の属性と値のペアが完全に一致しない場合は、default.policy ファイルが検査されます。 access request パケットの属性ペアがどちらのファイルでも一致しない場合、access reject パケットが送信されます。 どちらか一方のファイルで一致が見つかった場合は、access accept パケットがクライアントに送信されます。 これによって、事実上 2 つのレベルのポリシーが確立されます。

default.auth ファイルは、ポリシーが検査された後にクライアントに戻す属性と値のペアのリストとして使用されます。 **default.auth** ファイルにも属性と値のペアが含まれますが、これも当初はブランクなので、希望の設定に構成する必要があります。 希望の許可属性設定を構成するには、**default.auth** ファイルを編集するか、SMIT を使用する必要があります。値を含む属性は、いずれも access accept パケットに入れて NAS に自動的に戻されます。

また、固有のユーザー名に基づき、.auth 拡張子を付けたファイル (***user_id*.auth** など) を作成して、ユーザー固有の戻り許可属性を定義することもできます。このカスタム・ファイルは /etc/radius/authorization ディレクトリーに常駐させる必要があります。 各ユーザー・ファイルを作成および編集するための SMIT パネルがあります。

各ユーザーの許可属性は、default.auth ファイルまたは global.auth ファイルにあるデフォルトの許可属性と共に access-accept パケットに入れて送り返されます。

default.auth ファイルと *user_id*.auth ファイルで値が共通の場合は、ユーザーの値がデフォルト値をオーバーライドします。 これにより、すべてのユーザーに対していくつかのグローバル許可属性 (サービスまたはリソース) を指定でき、さらにユーザーごとに、より具体的なレベルの許可も指定できます。

注: 許可属性をユーザー固有の許可属性と結合する場合、組み合わせによる何らかの別の動作を望まない限り、default.auth ファイルを使用する代わりに、global.auth ファイルを使用します。

6100-02 テクノロジー・レベルを適用済みの AIX バージョン 6.1 以降では、 RADIUS は global.auth 許可ファイルをサポートします。 このファイルは、ユーザー固有の許可属性 (user_id.auth ファイルに定義されている) とグローバル許可属性のセットを結合するという、元の意図に取って代わって強化するものです。

default.auth ファイルと違って、 user_id.auth ファイルはユーザー固有の許可ファイルで検出された類似属性でオーバーライドされ、これにより、それらの属性との結合でユーザーの許可をより柔軟に維持できるようにします。

属性が default.auth ファイルと user_id.auth ファイルで共通の場合は、デフォルト値はユーザーの値でオーバーライドされます。 これにより、すべてのユーザーは一部のデフォルトの許可属性 (サービスまたはリソース) に対して、デフォルト値をオーバーライドすることが可能になり、さらにユーザーごとに、より具体的なレベルを許可することも可能です。

global.auth ファイルの属性についても同じことが言えます。ただし、これらの属性は user_id.auth 属性でオーバーライドされません。その代りに、2 つのファイルの属性が結合されます。 これはベンダー固有の属性 (VSA) が指定されている場合に役に立ちます。

許可プロセスは、次のとおりです。

1. デーモンの起動時に、/etc/radius/authorization/default.policy ファイル、および default.auth ファイルからのデフォルト・ポリシーと許可リストがメモリーに読み取られます。
2. ユーザー ID とパスワードを認証します。
3. 着信パケットの属性と値のペアを検査します。
   a. カスタム ***user_id*.auth** ファイルを検査します。
   b. 一致が見つからない場合、**default.policy** ファイルを検査します。
   c. 一致が見つからない場合は、access reject パケットが送信されます。
4. ユーザーの許可属性がある場合は、それを適用します。
   a. **/etc/radius/authorization/*user_id*.auth** ファイルと **default.auth** ファイルを読み取り、2 つのエントリーを比較します。
   b. デフォルト・エントリーの上の、ユーザーのファイルにあるエントリーを使用します。
   c. 得られた属性と global.auth ファイルにある属性を結合します。
5. 許可属性を access accept パケットで戻します。

アカウンティング

RADIUS アカウンティング・サーバーは、クライアントからアカウンティング要求を受信し、要求を正常に受信してアカウンティング・データを書き込んだことを示す応答をそのクライアントに戻す責任を負っています。

radiusd.conf ファイルでローカル・アカウンティングを使用可能にすることができます。

RADIUS アカウンティングを使用するようにクライアントを構成すると、サービス配信の開始時に、そのクライアントは ACCOUNTING_START パケットを生成します。このパケットには、配信するサービスのタイプと配信先のユーザーが記述されています。 クライアントはこのパケットを RADIUS アカウンティング・サーバーに送信し、このサーバーはパケットを受信した旨の確認通知を戻します。サービス配信の終了時には、クライアントは ACCOUNTING_STOP パケットを生成します。このパケットには、配信されたサービスのタイプと、さらにオプションで、経過時間、入力と出力のオクテット数、入力と出力のパケット数などの統計が記述されています。 RADIUS アカウンティング・サーバーは ACCOUNTING_STOP パケットを受信すると、アカウンティング・クライアントにパケットを受信した旨の確認通知を戻します。

ACCOUNTING_REQUEST は、START、STOP のいずれの場合も、ネットワークを通じて RADIUS アカウンティング・サーバーに発信されます。 確認通知を受信するまで、クライアントが ACCOUNTING_REQUEST パケットの送信の試みを継続することをお勧めします。 また、1 次サーバーがダウンしたり、プロキシー構成を使用して到達不可の場合、クライアントは代替サーバー (複数の場合もあり) に要求を転送することもできます。 プロキシー・サービスについて詳しくは、 372 ページの『プロキシー・サービス』を参照してください。

アカウンティング・データは、標準 RADIUS フォーマットの *attribute=value* で、ローカル /etc/var/radius/data/accounting ファイルに書き込まれます。書き込まれるデータはパケット内のアカウンティング・データで、タイム・スタンプ付きです。 RADIUS アカウンティング・サーバーは、アカウンティング・パケットを正常に記録できない場合、クライアントに **Accounting_Response** 確認通知を送信せず、**syslog** ファイルにエラー情報を記録します。

/var/radius/data/accounting ファイル**:**

/var/radius/data/accounting は、クライアントが ACCOUNTING START および ACCOUNTING STOP パケットで送信する内容をキャプチャーします。

/var/radius/data/accounting ファイルは、最初にインストールされた時点では空になっています。 データは、クライアントが ACCOUNTING START パケットと ACCOUNTING STOP パケットで送信する内容に基づいてファイルに書き込まれます。

以下は、AIX RADIUS サーバーが /var/radius/data/accounting ファイルに書き込むデータのタイプのサンプルです。 情報は、ご使用システムのセットアップ方法によって異なります。

注**:**

- **/var** ファイルシステムが、すべてのアカウンティング・データを処理できる十分な大きさであることを確認してください。
- このファイルのデータ解析に、サード・パーティーの Perl スクリプトを使用できます。 アカウント・データからレポートを生成するスクリプトの例が、http://www.pgregg.com/projects/radiusreport にあります。
- アカウンティング・パケットをプロキシーすることもできます。

```
Thu May 27 14:43:19 2004
 NAS-IP-Address = 10.10.10.1
 NAS-Port = 1
 NAS-Port-Type = Async
 User-Name = "rod"
 Acct-Status-Type = Start
 Acct-Authentic = RADIUS
 Service-Type = Framed-User
 Acct-Session-Id = "0000000C"
 Framed-Protocol = PPP
 Acct-Delay-Time = 0
 Timestamp = 1085686999

Thu May 27 14:45:19 2004
 NAS-IP-Address = 10.10.10.1
 NAS-Port = 1    <-- rod was physically connected to port #1 on the hardware
 NAS-Port-Type = Async
 User-Name = "rod"
 Acct-Status-Type = Stop
 Acct-Authentic = RADIUS
 Service-Type = Framed-User
 Acct-Session-Id = "0000000C"   <-- note the session id's are the same so can match up start with stops
 Framed-Protocol = PPP
 Framed-IP-Address = 10.10.10.2   <-- IP address of user rod
 Acct-Terminate-Cause = User-Request  <-- user cancelled the session
 Acct-Input-Octets = 4016
 Acct-Output-Octets = 142
 Acct-Input-Packets = 35
 Acct-Output-Packets = 7
 Acct-Session-Time = 120  <--- seconds
 Acct-Delay-Time = 0
 Timestamp = 1085687119  <--- note "rod" was only logged on for 120 seconds (2 minutes)
```

プロキシー・サービス

プロキシー・サービスを使用すると、RADIUS サーバーが NAS から別の RADIUS サーバーに要求を転送し、さらに NAS に応答メッセージを戻すことができます。 プロキシー・サービスは、レルム名に基づきます。

RADIUS サーバーは、同時にプロキシー・サーバーとバックエンド・サーバーの両方として動作できます。 このメカニズムは、アカウンティング・パケットと認証パケットの両方に適用できます。 プロキシーは、デフォルトでは **radiusd.conf** ファイルで使用不可になっています。

レルム**:**

レルムとは、通常は User-Name 属性に含まれる値の前または後に挿入される ID のことで、RADIUS サーバーはこれを使用して、認証とアカウンティングのプロセスの開始時に接続するサーバーを識別できます。

以下の例は、RADIUS でのレルムの使用法を示しています。

ユーザー *Joe* は、サクラメントにある会社 *XYZ* に雇用されています。 この地域のレルムは *SAC* です。しかし、*Joe* は現在、出張でニューヨーク市にいます。 ニューヨーク市のレルムは *NYC* です。 *Joe* が *NYC* レルムにダイヤルインしたときに渡されるユーザー名は *SAC/Joe* です。 これで、*NYC* RADIUS レルム・サーバーに対し、このパケットは *SAC* レルム・ユーザーの認証とアカウンティングを行うサーバーに転送する必要があることが通知されます。

レルム ***user-name*** 属性**:**

認証およびアカウンティング・パケットがレルムを通ってどのように経路指定されるかは、**User-Name** 属性に基づいています。 この属性は、認証またはアカウンティングを実行する最終サーバーに到着するためにパケットが通るレルムの順序を定義します。

パケットは、**User-Name** 属性の中でレルムが相互に文字列として並べられることによって経路指定されます。 **User-Name** 属性に実際に挿入されるレルムが最終的にパケットのパスを決定しますが、この決定は、RADIUS のレイアウトを配置する管理者に委ねられます。 レルム・ホップの名前は、**User-Name** 属性の前でも後でも挿入することができます。 複数のレルムを区切るときに最も一般的に使用される文字は、**User-Name** 属性の前に付けるプレフィックス・デリニエーターとしてはスラッシュ (/)、そして **User-Name** 属性の後に付けるサフィックス・デリニエーターとしてはアンパーサンド (&) です。 デリニエーターは **radiusd.conf** ファイルで構成されます。 **User-Name** 属性は、左から右へ解析されます。

プレフィックス・メソッドのみを使用する **User-Name** 属性の例を、次に示します。

```
USA/TEXAS/AUSTIN/joe
```

サフィックス・メソッドのみを使用する **User-Name** 属性の例を、次に示します。

```
joe@USA@TEXAS@AUSTIN
```

プレフィックスとサフィックスの両方のメソッドを使用することも可能です。 パケットが通過するレルム・ホップを指定すると、ホップの順序は左から右に解析され、すべてのプレフィックス・ホップが処理されてからサフィックス・ホップが処理される、ということを覚えておくことが重要です。 ユーザーの認証またはアカウンティング・データの書き込みは、1 つのノードで行う必要があります。

次の例では両方のメソッドが使用されていますが、前の例と同じ結果が生成されています。

```
USA/joe@TEXAS@AUSTIN
```

プロキシー・サービスの構成:

RADIUS プロキシー構成情報は、/etc/radius ディレクトリーの proxy ファイルの中にあります。

初期 **proxy** ファイルには、エントリーの例が含まれています。 proxy ファイルには、**Realm Name**、**Next Hop IP address**、および **Shared Secret** の 3 つのフィールドがあります。

プロキシー規則を構成するには、次のいずれかを選択してください。

```
    Configure Proxy Rules

List all Proxy
Add a Proxy
Change / Show Characteristics of a Proxy
Remove a Proxy
```

「**List all Proxy (**すべてのプロキシーをリスト**)**」オプションを選択すると、**/etc/radius/proxy** ファイルを読むことができます。3 つのフィールドが列形式で表示されます。 以下は各列の見出しです。

```
realm_name   next_hop_address  shared_secret
```

「**Add a Proxy (**プロキシーの追加**)**」を選択して、次の画面を表示します。 パネルから情報が検索され、**/etc/radius/proxy** ファイルの下部にデータが追加されます。

プロキシー・チェーンの各ホップは、RADIUS サーバー間で共有秘密鍵を使用します。共有秘密鍵は **/etc/radius/proxy_file** に入っています。共有秘密鍵は、チェーン内のプロキシー・ホップごとに固有でなければなりません。

共有秘密鍵の作成についての詳細は、 360 ページの『/etc/radius/clients ファイル』を参照してください。

プロキシーを追加するには、次に示すフィールドから選択します。

```
            Add a Proxy
*Realm Name                              []  (max 64 chars)
*Next Hop IP address (dotted decimal)    [xx.xx.xx.xx]
*Shared Secret                           []  (minimum 6, maximum 256 chars)
```

「**Change/Show (**変更**/**表示**)**」オプションを選択すると、レルム名のリストが表示されます。 このリストはポップアップ画面に表示され、そこからレルム名を選択する必要があります。

「**Remove a Proxy (**プロキシーの除去**)**」オプションを選択すると、レルム名のリストが表示されます。このリストはポップアップ画面に表示され、ユーザーがそこからレルム名を選択する必要があります。 名前を選択すると、確認のポップアップ画面が表示されてからレルムが除去されます。

次の例は、**radiusd.conf** ファイルのプロキシー構成情報セクションです。

```
#-------------------------------------------------------------------#
#       PROXY RADIUS Information                                    #
#                                                                   #
#                                                                   #
#   Proxy_Allow               :  ON or OFF. If ON, then the server  #
#                                  can proxy packets to realms it   #
#                                  knows of and the following       #
#                                  fields must also be configured.  #
#   Proxy_Use_Table           :  ON or OFF. If ON, then the server  #
#                                  can use table for faster         #
#                                  processing of duplicate requests #
```

```
#                                Can be used without proxy ON, but #
#                                it is required to be ON if        #
#                                Proxy_Use_Table is set to ON.     #
#   Proxy_Realm_name          :  This field specifies the realm    #
#                                this server services.             #
#   Proxy_Prefix_delim        :  A list of separators for parsing  #
#                                realm names added as a prefix to  #
#                                the username.  This list must be  #
#                                mutually exclusive to the Suffix  #
#                                delimiters.                       #
#   Proxy_Suffix_delim        :  A list of separators for parsing  #
#                                realm names added as a suffix to  #
#                                the username.  This list must be  #
#                                mutually exclusive to the Prefix  #
#                                delimiters.                       #
#   Proxy_Remove_Hops         :  YES or NO.  If YES then the       #
#                                will remove its realm name, the   #
#                                realm names of any previous hops  #
#                                and the realm name of the next    #
#                                server the packet will proxy to.  #
#                                                                  #
#   Proxy_Retry_count         :  The number of times to attempt    #
#                                to send the request packet.       #
#                                                                  #
#   Proxy_Time_Out            :  The number of seconds to wait     #
#                                in between send attempts.         #
#                                                                  #
#------------------------------------------------------------------#
Proxy_Allow               :   OFF
Proxy_Use_Table           :   OFF
Proxy_Realm_name          :
Proxy_Prefix_delim        :   $/
Proxy_Suffix_delim        :   @.
Proxy_Remove_Hops         :   NO
Proxy_Retry_count         :   2
Proxy_Time_Out            :   3
```

RADIUS サーバーの構成**:**

RADIUS サーバー・デーモンはいくつかの構成ファイルを使用します。サーバー構成情報は **/etc/radius/radiusd.conf** ファイルに保存されます。サーバー構成ファイルのパッケージが、デフォルト値が設定された状態で出荷されます。

注**:** 以下は、RADIUS の「Configure Server (サーバーの構成)」SMIT パネルの例です。

```
    サーバーの構成

RADIUS Directory            /etc/radius
*Database Location           [UNIX]
Local AVL Database File Name [dbdata.bin]
Local Accounting             [ON]
Local Accounting Directory   []

Debug Level                  [3]
Accept Reply-Message         []
Reject Reply-Message         []
Challenge Reply-Message      []
Password Expired Reply Message []
Support Renewal of Expired Passwords [NO]
Require Message Authenticator  [NO]

*Authentication Port Number  [1812]
*Accounting Port Number      [1813]

LDAP Server Name             []
LDAP Server Port Number      [389]
LDAP Server Admin Distinguished Name     []
LDAP Server Admin Password         []
LDAP Base Distinguished Name       [cn=aixradius]
LDAP Size Limit                    [0]
LDAP Hop Limit                     [0]
LDAP wait time limit               [10]
LDAP debug level                   [ 0]

Proxy Allowed                      [OFF]
Proxy Use table                    [OFF]
Proxy Realm Name                   []
Proxy Prefix Delimiters            [$/]
Proxy Suffix Delimiters            [@.]
     NOTE: prefix & suffix are mutually exclusive
Proxy Remove Hops                  [NO]
Proxy Retry Count                  [2]
Proxy Timeout                      [30]
UNIX Check Login Restrictions      [OFF]
Enable IP Pool                     [ON]
Authentication Method Sequence     [TLS, MD5]
OpenSSL Configuration File         []
```

ロギング・ユーティリティー

RADIUS サーバーは SYSLOG を使用して、活動状況とエラー情報をログに記録します。

ログ情報には、以下の 3 つのレベルがあります。

0 問題またはエラー、およびデーモンの開始のみがログに記録されます。

3 access_accept、access_reject*、discard、および error メッセージの監査証跡が記録されます。

注**:** discard メッセージは、着信パケットが無効で、応答パケットが生成されない場合に記録されます。

9 レベル 0 および 3 のロギング情報を含め、さらに詳細な情報が記録されます。 デバッグする場合は、レベル 9 のロギングのみを実行します。

ロギングのデフォルト・レベルはレベル 3 です。レベル 3 のロギングは、RADIUS サーバーの監査レベルを改善するときに使用します。 サーバーのロギングのレベルに応じて、該当するログに保管された活動状況を使用して、疑わしい活動状況パターンがあるか検査できます。 セキュリティー違反があった場合、SYSLOG 出力を使用して、いつ、どのようにその違反が発生したかを判別でき、さらに受け入れたアクセス数も判別できる場合もあります。 この情報は、将来の問題を防止するため、さらに優れたセキュリティー手段の開発に利用できます。

関連情報:

IBM Directory Server

syslogd デーモンを使用するための **RADIUS** の構成**:**

SYSLOG を使用してアクティビティーおよびエラー情報を表示するには、syslogd デーモンを使用可能にする必要があります。

syslogd デーモンを使用可能にするには、以下のステップを完了してください。

1. /etc/syslog.conf ファイルを編集して、次のエントリーを追加します: local4.debug var/adm/ipsec.log。 *local4* 機能を使用して、トラフィック・イベントおよび IP セキュリティー・イベントを記録します。 標準のオペレーティング・システム優先順位が適用されます。 *debug* の優先順位は、 IP セキュリティー・トンネルとフィルターを通過したトラフィックが安定し、正しく移動したことを示すまで、維持し続けてください。

   注**:** フィルター・イベントのロギングにより、 IP セキュリティー・ホストでの有効なアクティビティーが作成されますが、大量のストレージを消費する恐れがあります。
2. **/etc/syslog.conf file** を保存します。
3. ログ・ファイルに指定したディレクトリーに移動して、名前が同じ空ファイルを作成します。 上記の場合、/var/adm ディレクトリーに移動し、次のように **touch** コマンドを実行します。

   ```
   touch ipsec.log
   ```
4. 以下のように、syslogd サブシステムに **refresh** コマンドを発行します。

   ```
   refresh -s syslogd
   ```

SYSLOG 出力設定の構成**:**

SYSLOG 出力にどのくらいのデバッグ情報を含めたいかに応じて、Debug_Level 0、3、または 9 を radiusd.conf ファイルに設定することができます。

デフォルト設定は 3 です。radiusd.conf ファイルのデバッグ・セクションは、次のようになります。

```
#.
#.
#.
#   Debug_Level      :  This pair sets the debug level at which    #
#                       the RADIUS server will run.  Appropriate   #
#                       values are 0,3 or 9. The default is 3.     #
#                       Output is directed to location specified   #
#                       by *.debug stanza in /etc/syslog.conf      #
#                                                                  #
#                       Each level increases the amount of messages#
#                       sent to syslog. For example "9" includes   #
#                       the new messages provided by "9" as well   #
#                       as all messages generated by level 0 and 3.#
#                                                                  #
#                       0 : provides the minimal output to the     #
#                           syslogd log.  It sends start up        #
#                           and end messages for each RADIUS       #
#                           process.  It also logs error           #
#                           conditions.                            #
#                                                                  #
#                       3 : includes general ACCESS ACCEPT, REJECT #
#                           and DISCARD messages for each packet.  #
#                           This level provides a general audit    #
#                           trail for authentication.              #
#                                                                  #
```

```
#                       9 : Maximum amount of log data. Specific   #
#                           values of attributes while a           #
#                           transaction is passing thru            #
#                           processing and more.                   #
#                           [NOT advised under normal operations]  #
#                                                                  #
#------------------------------------------------------------------#
```

以下の例は、さまざまなデバッグ・レベルの出力例を示しています。

デバッグ・レベル **3** のアカウンティング・パケット

```
Aug 18 10:23:57 server1 syslog: [0]:Monitor process [389288] has started
Aug 18 10:23:57 server1 radiusd[389288]: [0]:Local database (AVL) built.
Aug 18 10:23:57 server1 radiusd[389288]: [0]:Authentication process started : Pid= 549082 Port = 1812
Aug 18 10:23:57 server1 radiusd[389288]: [0]:Accounting process started : Pid= 643188 Port = 1813
Aug 18 10:23:57 server1 radiusd[643188]: [0]:Socket created [15]
Aug 18 10:23:57 server1 radiusd[643188]: [0]:Bound Accounting socket [15]
Aug 18 10:23:57 server1 radiusd[549082]: [0]:Socket created [15]
Aug 18 10:23:57 server1 radiusd[549082]: [0]:Bound Authentication socket [15]
Aug 18 10:24:07 server1 radiusd[643188]: [1]:*** Start Process_Packet() ***
Aug 18 10:24:07 server1 radiusd[643188]: [1]:Code 4, ID = 96, Port = 41639 Host = 10.10.10.10
Aug 18 10:24:07 server1 radiusd[643188]: [1]:ACCOUNTING-START - sending Accounting Ack to User [ user_id1 ]
Aug 18 10:24:07 server1 radiusd[643188]: [1]:Sending Accounting Ack of id 96 to 10.10.10.10 (client1.ibm.com)
Aug 18 10:24:07 server1 radiusd[643188]: [1]:send_acct_reply() Outgoing Packet:
Aug 18 10:24:07 server1 radiusd[643188]: [1]: Code = 5, Id = 96, Length = 20
Aug 18 10:24:07 server1 radiusd[643188]: [1]:*** Leave Process_Packet() ***
Aug 18 10:24:13 server1 radiusd[643188]: [2]:*** Start Process_Packet() ***
Aug 18 10:24:13 server1 radiusd[643188]: [2]:Code 4, ID = 97, Port = 41639 Host = 10.10.10.10
Aug 18 10:24:13 server1 radiusd[643188]: [2]:ACCOUNTING-STOP - sending Accounting Ack to User [ user_id1 ]
Aug 18 10:24:14 server1 radiusd[643188]: [2]:Sending Accounting Ack of id 97 to 10.10.10.10 (client1.ibm.com)
Aug 18 10:24:14 server1 radiusd[643188]: [2]:send_acct_reply() Outgoing Packet:
Aug 18 10:24:14 server1 radiusd[643188]: [2]: Code = 5, Id = 97, Length = 20
Aug 18 10:24:14 server1 radiusd[643188]: [2]:*** Leave Process_Packet() **
```

レベル **9** のアカウンティング・パケット

```
Aug 18 10:21:18 server1 syslog: [0]:Monitor process [643170] has started
Aug 18 10:21:18 server1 radiusd[643170]: [0]:Local database (AVL) built.
Aug 18 10:21:18 server1 radiusd[643170]: [0]:Authentication process started : Pid= 389284 Port = 1812
Aug 18 10:21:18 server1 radiusd[643170]: [0]:Accounting process started : Pid= 549078 Port = 1813
Aug 18 10:22:03 server1 radiusd[643170]: [0]:PID = [389284] dead
Aug 18 10:22:03 server1 radiusd[643170]: [0]:PID = [549078] dead
Aug 18 10:22:03 server1 radiusd[643170]: [0]:All child processes stopped. radiusd parent stopping
Aug 18 10:22:09 server1 syslog: [0]:Monitor process [1081472] has started
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Local database (AVL) built.
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Inside client_init()
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Number of client entries read: 1
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Inside read_authorize_policy routine for file:
  /etc/radius/authorization/default.policy.
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Inside read_authorize_file routine for file:
  /etc/radius/authorization/default.policy.
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:read_authorize_file() routine complete.
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Inside read_authorize_file routine for file:
  /etc/radius/authorization/default.auth.
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:read_authorize_file() routine complete.
Aug 18 10:22:09 server1 radiusd[549080]: [0]:connect_to_LDAP_server:Database Location (where the data
  resides)=LDAP.
Aug 18 10:22:09 server1 radiusd[549080]: [0]:connect_to_LDAP_server:LDAP Server name= server1.austin.ibm.com.
Aug 18 10:22:09 server1 radiusd[549080]: [0]:connect_to_LDAP_server:LDAP Server port= 389.
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Authentication process started : Pid= 549080 Port = 1812
Aug 18 10:22:09 server1 radiusd[389286]: [0]:connect_to_LDAP_server:Database Location (where the data
 resides)=LDAP.
Aug 18 10:22:09 server1 radiusd[389286]: [0]:connect_to_LDAP_server:LDAP Server name= server1.austin.ibm.com.
Aug 18 10:22:09 server1 radiusd[389286]: [0]:connect_to_LDAP_server:LDAP Server port= 389.
Aug 18 10:22:09 server1 radiusd[1081472]: [0]:Accounting process started : Pid= 389286 Port = 1813
Aug 18 10:22:10 server1 radiusd[549080]: [0]:Socket created [15]
Aug 18 10:22:10 server1 radiusd[549080]: [0]:Bound Authentication socket [15]
Aug 18 10:22:10 server1 radiusd[389286]: [0]:Socket created [15]
Aug 18 10:22:10 server1 radiusd[389286]: [0]:Bound Accounting socket [15]
```

```
Aug 18 10:22:15 server1 radiusd[389286]: [1]:*** Start Process_Packet() ***
Aug 18 10:22:15 server1 radiusd[389286]: [1]:Incoming Packet:
Aug 18 10:22:15 server1 radiusd[389286]: [1]:  Code = 4, Id = 94, Length = 80
Aug 18 10:22:15 server1 radiusd[389286]: [1]:  Authenticator = 0xC5DBDDFE6EFFFDBD6AE64CA35947DD0F
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =  40, Length =   6, Value = 0x00000001
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =   1, Length =   8, Value = 0x67656E747931
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =   4, Length =   6, Value = 0x00000000
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =   8, Length =   6, Value = 0x0A0A0A01
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =  44, Length =   8, Value = 0x303030303062
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =  30, Length =  10, Value = 0x3132332D34353638
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =  31, Length =  10, Value = 0x3435362D31323335
Aug 18 10:22:15 server1 radiusd[389286]: [1]:    Type =  85, Length =   6, Value = 0x00000259
Aug 18 10:22:15 server1 radiusd[389286]: [1]:Starting parse_packet()
Aug 18 10:22:15 server1 radiusd[389286]: [1]:Code 4, ID = 94, Port = 41639 Host = 10.10.10.10
Aug 18 10:22:15 server1 radiusd[389286]: [1]:Acct-Status-Type = Sta
```

レベル **0** の認証パケット

```
Aug 18 10:06:11 server1 syslog: [0]:Monitor process [1081460] has started
Aug 18 10:06:11 server1 radiusd[1081460]: [0]:Local database (AVL) built.
Aug 18 10:06:11 server1 radiusd[1081460]: [0]:Authentication process started : Pid= 549076 Port = 1812
Aug 18 10:06:11 server1 radiusd[1081460]: [0]:Accounting process started : Pid= 389282 Port = 18
```

レベル **3** の認証パケット

```
Aug 18 10:01:32 server2 radiusd[389276]: [3]:*** Start Process_Packet() ***
Aug 18 10:01:32 server2 radiusd[389276]: [3]:Code 1, ID = 72, Port = 41638 Host = 10.10.10.10
Aug 18 10:01:32 server2 radiusd[389276]: [3]:authenticate_password_PAP: Passwords do not match, user is rejected
Aug 18 10:01:32 server2 radiusd[389276]: [3]:Authentication failed for user [user_id1] using IP [10.10.10.10]
Aug 18 10:01:32 server2 radiusd[389276]: [3]:ACCESS-REJECT - sending reject for id 72 to 10.10.10.10
  (client1.ibm.com)
Aug 18 10:01:32 server2 radiusd[389276]: [3]:send_reject() Outgoing Packet:
Aug 18 10:01:32 server2 radiusd[389276]: [3]: Code = 3, Id = 72, Length = 30
Aug 18 10:01:32 server2 radiusd[389276]: [3]:*** Leave Process_Packet() ***
Aug 18 10:01:53 server2 radiusd[389276]: [4]:*** Start Process_Packet() ***
Aug 18 10:01:53 server2 radiusd[389276]: [4]:Code 1, ID = 74, Port = 41638 Host = 10.10.10.10
Aug 18 10:01:53 server2 radiusd[389276]: [4]:authenticate_password_PAP: Passwords Match, user is authenticated
Aug 18 10:01:53 server2 radiusd[389276]: [4]:Authentication successful for user [user_id1] using IP [10.10.10.10]
Aug 18 10:01:53 server2 radiusd[389276]: [4]:Authorization successful for user [user_id1] using IP [10.10.10.10]
Aug 18 10:01:53 server2 radiusd[389276]: [4]:ACCESS-ACCEPT - sending accept for id 74 to 10.10.10.10
  (client1.ibm.com)
Aug 18 10:01:53 server2 radiusd[389276]: [4]:send_accept() Outgoing Packet:
Aug 18 10:01:53 server2 radiusd[389276]: [4]: Code = 2, Id = 74, Length = 31
Aug 18 10:01:53 server2 radiusd[389276]: [4]:*** Leave Process_Packet() **
```

レベル **9** の認証パケット

```
Aug 18 10:03:56 server1 radiusd[389278]: [1]:*** Start Process_Packet() ***
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Incoming Packet:
Aug 18 10:03:56 server1 radiusd[389278]: [1]:  Code = 1, Id = 77, Length = 58
Aug 18 10:03:56 server1 radiusd[389278]: [1]:  Authenticator = 0xE6CB0F9C22BB4E799854E734104FB2D5
Aug 18 10:03:56 server1 radiusd[389278]: [1]:    Type =   1, Length =   8, Value = 0x67656E747931
Aug 18 10:03:56 server1 radiusd[389278]: [1]:    Type =   4, Length =   6, Value = 0x00000000
Aug 18 10:03:56 server1 radiusd[389278]: [1]:    Type =   2, Length =  18, Value = 0x**********
**********************
Aug 18 10:03:56 server1 radiusd[389278]: [1]:    Type =   7, Length =   6, Value = 0x00000001
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Starting parse_packet()
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Code 1, ID = 77, Port = 41638 Host = 10.10.10.10
Aug 18 10:03:56 server1 radiusd[389278]: [1]:User-Name = "user_id1"
Aug 18 10:03:56 server1 radiusd[389278]: [1]:NAS-IP-Address = 10.10.10.10
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Framed-Protocol = PPP
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Leaving parse_packet()
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Verifying Message-Authenticator
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Message-Authenticator successfully verified
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside proxy_request_needed() function
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Proxy is not turned on
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Username = [user_id1]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Client IP = [10.10.10.10]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside parse_for_login( user_id1 )
Aug 18 10:03:56 server1 radiusd[389278]: [1]:User_id remaining after prefix removal = [user_id1]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:User_id remaining after suffix removal = [user_id1]
```

```
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside rad_authenticate() function
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Authentication request received for [client1.austin.ibm.com]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Calling get_ldap_user() to get LDAP user data
Aug 18 10:03:56 server1 radiusd[389278]: [1]:get_ldap_user:LDAP user id: user_id1.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:get_ldap_user:LDAP max_login_cnt:2.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:get_ldap_user:LDAP EAP_type: 4.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:get_ldap_user:LDAP passwordexpiredweeks: 9.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:get_ldap_active_sessions:number of free entries= 2.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:get_ldap_active_session:dn retrieved=
radiusuniqueidentifier=user_id11,ou=radiusActiveUsers,cn=aixradius.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside get_client_secret routine for ip:10.10.10.10
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Found NAS-IP = [10.10.10.10]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Found shared secret.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:authenticate_password_PAP: Passwords Match, user is authenticated
Aug 18 10:03:56 server1 radiusd[389278]: [1]:is_ldap_pw:password for user has NOT expired
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Authentication successful for user [user_id1] using IP [10.10.10.10]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside rad_authorize() routine.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside read_authorize_policy routine for file:
  /etc/radius/authorization/user_id1.policy.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside read_authorize_file routine for file:
  /etc/radius/authorization/user_id1.policy.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Did not open /etc/radius/authorization/user_id1.policy file.
  File may not be found.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Error reading policy file: /etc/radius/authorization/user_id1.policy.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:rad_authorize:default policy list and userpolicy list were empty.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:In create_def_copy() routine.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Successfully made a copy of the master authorization list.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside read_authorize_file routine for file:
  /etc/radius/authorization/user_id1.auth.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Did not open /etc/radius/authorization/user_id1.auth file.
  File may not be found.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:rad_authorize:copy authorization list and user list were empty.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Authorization successful for user [user_id1] using IP [10.10.10.10]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:ACCESS-ACCEPT - sending accept for id 77 to 10.10.10.10
  (client1.austin.ibm.com)
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside proxy_response_needed() function
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Proxy is not turned on
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Inside get_client_secret routine for ip:10.10.10.10
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Found NAS-IP = [10.10.10.10]
Aug 18 10:03:56 server1 radiusd[389278]: [1]:Found shared secret.
Aug 18 10:03:56 server1 radiusd[389278]: [1]:send_accept() Outgoing Packet:
Aug 18 10:03:56 server1 radiusd[389278]: [1]: Code = 2, Id = 77, Length = 31
Aug 18 10:03:56 server1 radiusd[389278]: [1]:send_accept() Outgoing Packet:
Aug 18 10:03:56 server1 radiusd[389278]: [1]:  Code = 2, Id = 77, Length = 31
Aug 18 10:03:56 server1 radiusd[389278]: [1]:  Authenticator = 0xCCB2B645BBEE86F5E4FC5BE24E904B2A
Aug 18 10:03:56 server1 radiusd[389278]: [1]:    Type =  18, Length =  11, Value = 0x476F6F646E65737321
Aug 18 10:03:56 server1 radiusd[389278]: [1]:*** Leave Process_Packet() ***
Aug 18 10:04:18 server1 radiusd[389278]: [2]:*** Start Process_Packet() ***
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Incoming Packet:
Aug 18 10:04:18 server1 radiusd[389278]: [2]:  Code = 1, Id = 79, Length = 58
Aug 18 10:04:18 server1 radiusd[389278]: [2]:  Authenticator = 0x774298A2B6DD90D7C33B3C10C4787D41
Aug 18 10:04:18 server1 radiusd[389278]: [2]:    Type =   1, Length =   8, Value = 0x67656E747931
Aug 18 10:04:18 server1 radiusd[389278]: [2]:    Type =   4, Length =   6, Value = 0x00000000
Aug 18 10:04:18 server1 radiusd[389278]: [2]:    Type =   2, Length =  18, Value = 0x*******
*************************
Aug 18 10:04:18 server1 radiusd[389278]: [2]:    Type =   7, Length =   6, Value = 0x00000001
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Starting parse_packet()
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Code 1, ID = 79, Port = 41638 Host = 10.10.10.10
Aug 18 10:04:18 server1 radiusd[389278]: [2]:User-Name = "user_id1"
Aug 18 10:04:18 server1 radiusd[389278]: [2]:NAS-IP-Address = 10.10.10.10
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Framed-Protocol = PPP
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Leaving parse_packet()
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Verifying Message-Authenticator
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Message-Authenticator successfully verified
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Inside proxy_request_needed() function
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Proxy is not turned on
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Username = [user_id1]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Client IP = [10.10.10.10]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Inside parse_for_login( user_id1 )
Aug 18 10:04:18 server1 radiusd[389278]: [2]:User_id remaining after prefix removal = [user_id1]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:User_id remaining after suffix removal = [user_id1]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Inside rad_authenticate() function
```

```
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Authentication request received for [client1.austin.ibm.com]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Calling get_ldap_user() to get LDAP user data
Aug 18 10:04:18 server1 radiusd[389278]: [2]:get_ldap_user:LDAP user id: user_id1.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:get_ldap_user:LDAP max_login_cnt:2.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:get_ldap_user:LDAP EAP_type: 4.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:get_ldap_user:LDAP passwordexpiredweeks: 9.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:get_ldap_active_sessions:number of free entries= 2.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:get_ldap_active_session:dn retrieved=
  radiusuniqueidentifier=user_id11, ou=radiusActiveUsers, cn=aixradius.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Inside get_client_secret routine for ip:10.10.10.10
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Found NAS-IP = [10.10.10.10]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Found shared secret.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:authenticate_password_PAP: Passwords do not match, user is rejected
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Authentication failed for user [user_id1] using IP [10.10.10.10]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:ACCESS-REJECT - sending reject for id 79 to 10.10.10.10
 (client1.austin.ibm.com)
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Inside proxy_response_needed() function
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Proxy is not turned on
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Inside get_client_secret routine for ip:10.10.10.10
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Found NAS-IP = [10.10.10.10]
Aug 18 10:04:18 server1 radiusd[389278]: [2]:Found shared secret.
Aug 18 10:04:18 server1 radiusd[389278]: [2]:send_reject() Outgoing Packet:
Aug 18 10:04:18 server1 radiusd[389278]: [2]: Code = 3, Id = 79, Length = 30
Aug 18 10:04:18 server1 radiusd[389278]: [2]:send_reject() Outgoing Packet:
Aug 18 10:04:18 server1 radiusd[389278]: [2]:  Code = 3, Id = 79, Length = 30
Aug 18 10:04:18 server1 radiusd[389278]: [2]:  Authenticator = 0x05D4865C6EBEFC1A9300D2DC66F3DBE9
Aug 18 10:04:18 server1 radiusd[389278]: [2]:    Type =  18, Length =  10, Value = 0x4261646E65737321
Aug 18 10:04:18 server1 radiusd[389278]: [2]:*** Leave Process_Packet() **
```

パスワード有効期限

パスワード有効期限により、RADIUS クライアントは、あるユーザーのパスワードが期限切れになったときに通知され、RADIUS プロトコルを通してそのユーザーのパスワードを更新することができます。

パスワード有効期限では、4 つの追加パケット・タイプと新規属性がサポートされます。 新しいパケット・タイプは AIX ディクショナリーに入れた状態で出荷されます。パスワード有効期限フィーチャーは必ずオンにしてください。

RADIUS がインストールされたすべてのシステムで、期限切れパスワードの RADIUS による更新を可能にすることはお勧めできません。 **radiusd.conf** ファイルのエントリーの 1 つに、期限切れパスワードの RADIUS による変更を許可または不許可に設定できるオプションが用意されています。 このオプションのデフォルトは不許可です。 `Password_Expired_Reply_Message` ユーザー応答メッセージを追加することができ、これは password-expired パケットに入れて戻されます。パスワード属性は新旧両方とも、PAP メソッドで暗号化および暗号化解除する必要があります。

ベンダー固有属性

ベンダー固有属性 (VSA) は、リモート・アクセス・サーバーのベンダー (通常はハードウェアのベンダー) が自社のサーバーでの RADIUS の働き方をカスタマイズするために定義します。

ベンダー固有属性は、ユーザーに複数のタイプのアクセス権を与える場合に必要です。 VSA は、RADIUS 定義の属性と組み合わせて使用できます。

VSA はオプションですが、NAS ハードウェアが正しく機能するために追加属性の構成を必要とする場合は、VSA をディクショナリー・ファイルに追加しなければなりません。

VSA はさらに許可を設定する場合にも使用できます。 **User-Name** と **Password** のほかに、VSA を許可に使用することができます。 サーバー側では、ユーザー許可ポリシー・ファイルに、特定のユーザーの Access-Request パケットで検査する属性のリストが含まれます。 このパケットに、ユーザー・ファイルにリストされた属性が含まれていない場合、NAS に access_reject が送り返されます。 VSA は、***user_id*.policy** ファイルで「attribute=value (属性=値)」のペアのリストとして使用することもできます。

以下は、ディクショナリーから引用した VSA セクションの例です。

```
########################################################################
#                                                                      #
#   This section contains examples of dictionary translations for      #
#   parsing vendor specific attributes (vsa). The example below is for #
#   "Cisco." Before defining an Attribute/Value pair for a    #
#   vendor a "VENDOR" definition is needed.                            #
#                                                                      #
#   Example:                                                           #
#                                                                      #
#   VENDOR          Cisco           9                                  #
#                                                                      #
#   VENDOR:  This specifies that the Attributes after this entry are   #
#            specific to Cisco.                                        #
#   Cisco :  Denotes the Vendor name                                   #
#   9     :  Vendor Id defined in the "Assigned Numbers" RFC           #
#                                                                      #
########################################################################

#VENDOR          Cisco           9

#ATTRIBUTE       Cisco-AVPair              1       string
#ATTRIBUTE       Cisco-NAS-Port            2       string
#ATTRIBUTE       Cisco-Disconnect-Cause    195     integer
#
#----------------Cisco-Disconnect-Cause--------------------------------#
#
#VALUE           Cisco-Disconnect-Cause    Unknown                 2
#VALUE           Cisco-Disconnect-Cause    CLID-Authentication-Failure 4
#VALUE           Cisco-Disconnect-Cause    No-Carrier              10
#VALUE           Cisco-Disconnect-Cause    Lost-Carrier            11
#VALUE           Cisco-Disconnect-Cause    No-Detected-Result-Codes    12
#VALUE           Cisco-Disconnect-Cause    User-Ends-Session       20
#VALUE           Cisco-Disconnect-Cause    Idle-Timeout            21
#VALUE           Cisco-Disconnect-Cause    Exit-Telnet-Session     22
#VALUE           Cisco-Disconnect-Cause    No-Remote-IP-Addr       23
```

RADIUS 応答メッセージ・サポート

応答メッセージは、**radiusd.conf** ファイル内で作成され構成されるテキストです。

これは、NAS または AP がユーザーに文字列として戻すためのメッセージです。 これには、成功、障害、またはチャレンジの 3 種類のメッセージがあります。 これらは読み取り可能テキスト・フィールドで、その内容はインプリメンテーションごとに異なり、サーバー構成時に構成されます。 これらの属性のデフォルトは「テキストなし」です。 構成できる属性は、なし、1 つ、2 つ、3 つ、あるいは全部です。

RADIUS は以下の応答メッセージ属性をサポートします。

- 受け入れた応答メッセージ
- 拒否した応答メッセージ
- CHAP 応答メッセージ
- パスワード有効期限切れの応答メッセージ

これらの属性は **radiusd.conf** 構成ファイルに追加され、デーモン開始時にグローバル構成構造内に読み取られます。 これらの値は、SMIT RADIUS パネルを使用して「**Configure Server (**サーバーの構成**)**」オプションの一部として設定します。 各文字列の最大文字数は 256 バイトです。

この機能は次のように実装されます。

1. **radiusd** デーモンを開始すると、このデーモンは **radiusd.conf** ファイルを読み取り、応答メッセージ属性を設定します。

2. access request パケットが受信されると、ユーザーの認証が行われます。
3. 認証の応答が access accept の場合、「Accept Reply-Message (受け入れた応答メッセージ)」のテキストがあるかどうか検査されます。 このテキストが存在する場合、その文字列が access accept パケットで戻されます。
4. 認証が拒否された場合、「Reject Reply-Message (拒否した応答メッセージ)」のテキストがあるかどうか検査され、そのテキストがパケットで戻されます。
5. 認証がチャレンジされる場合、CHAP 応答メッセージ属性が検査され、Access-Challenge パケットの一部として送信されます。

RADIUS サーバーの IP プール構成

RADIUS サーバーでは、IP アドレス・プールから IP アドレスを動的に割り当てることができます。

IP アドレス割り当ては許可プロセスの一部であり、認証の後に実行されます。 システム管理者はユーザーごとに固有の IP を割り当てる必要があります。 ユーザーに IP アドレスを動的に割り当てるために、RADIUS サーバーは次の 3 つのオプションを提供します。

- フレーム・プール属性
- ベンダー固有属性の使用
- RADIUS サーバー側の IP プーリング

フレーム・プール属性

IP プール *poolname* は Network Access Server (NAS) で定義する必要があります。 RADIUS サーバーが Access-Accept パックで **Framed-Pool** 属性 (タイプ 88 属性) を送信するためには、NAS が RFC2869 準拠でなければなりません。 システム管理者は NAS を構成し、ユーザーの許可属性を更新する必要があります。そのために、RADIUS サーバーのグローバル **default.auth** ファイルまたは **user.auth** ファイルのいずれかに **Framed-Pool** 属性を組み込みます。 RADIUS サーバーのディクショナリー・ファイルには次の属性が含まれています。

```
ATTRIBUTE    Framed-Pool    88       string
```

NAS が複数のアドレス・プールを使用できない場合、NAS はこの属性を無視します。 NAS 上のアドレス・プールには IP アドレスのリストが含まれます。 NAS は、指定されたプールで定義されている IP アドレスの 1 つを選出してユーザーに割り当てます。

ベンダー固有属性

独立系ソフトウェア・ベンダー (ISV) の製品の中には、**Framed-Pool** 属性を使用できなくても IP アドレス・プールを定義できるものがあります。 RADIUS サーバーでは、ベンダー固有属性 (VSA) モデルを使用して、そのようなアドレス・プールを使用することができます。 例えば、Cisco NAS は、Cisco-AVPair と呼ばれる属性を提供しています。 RADIUS サーバーのディクショナリー・ファイルには次の属性が含まれています。

```
VENDOR     Cisco           9
ATTRIBUTE  Cisco-AVPair    1       string
```

NAS が Access-Request パケットを送信する際、`Cisco-AVPair= "ip:addr-pool=`*poolname*`"`としてこの属性が組み込まれます。ここで、*poolname* は、NAS で定義されているアドレス・プールの名前です。 要求の認証と許可が行われた後、RADIUS サーバーは Access-Accept パケットで属性を戻します。 次に、NAS は、定義済みのプールを使用して、ユーザーに IP アドレスを割り当てます。システム管理者は NAS を構成し、ユーザーの許可属性を更新する必要があります。そのために、RADIUS サーバーのグローバル **default.auth** ファイルまたは **user.auth** ファイルのいずれかに VSA 属性を組み込みます。

RADIUS サーバー側の IP プーリング

IP アドレスのプールから IP アドレスを生成するように RADIUS サーバーを構成することができます。その IP アドレスは Access-Accept パケットの Framed-IP-Address 属性で戻されます。

システム管理者は SMIT インターフェースを使用して IP アドレスのプールを定義することができます。アドレスは **/etc/radius/ippool_def** ファイルに維持されます。 *poolnames* は **etc/radius/clients** ファイルで定義されます。 システム管理者は NAS-Port 番号も構成する必要があります。 RADIUS サーバー・デーモンは **etc/radius/clients** ファイルおよび **/etc/radius/ippool_def** ファイルの情報を使用して、データ・ファイルを作成します。 デーモンが開始されると、システム管理者は、RADIUS サーバーが停止されるまで *poolnames* または IP アドレスの範囲を変更または追加することはできません。 RADIUS サーバー・デーモンは、開始されると、構成ファイル (**/etc/radius/radius.conf**) を読み取り、IP アプリケーションが使用可能 (`Enable_IP_Pooling=YES`) の場合はグローバル IP フラグ (`IP_pool_flag`) をオンに設定します。次に、デーモンは **poolname.data** ファイルが存在するかどうかを検査します。 存在する場合はそのファイルを読み取り、ファイルの情報を共有メモリーに保持します。 次に、クライアントから受信した要求に基づいて、このファイルと共有メモリーを更新します。 このファイルが存在しない場合は、デーモンは **etc/radius/clients** ファイルおよび **/etc/radius/ippool_def** ファイルの情報を使用して、新しいファイルを作成します。 **poolname.data** ファイルの最大サイズ限界は 256 MB (AIX セグメント・サイズ限界) です。 **poolname.data** ファイルが 256 MB を超える場合、RADIUS サーバーはエラー・メッセージをログに記録して終了します。

デーモンは **/etc/radius/ippool_def** から IP プール詳細を取得し、プール名ごとに IP アドレスのテーブルを共有メモリーに維持します。 このテーブルには、NAS-IP-address、NAS-port、および IN USE フラグのエントリーがあります。 デーモンは NAS-IP NAS-port をキーとするハッシュ・テーブルを維持します。 複数のユーザーから要求が受信されると、UDP は要求をキューに入れ、デーモンはその要求から NAS-IP および NAS-port データを取得します。 デーモンはそれらの情報を使用して、*poolname* が該当 NAS に定義されているかどうかを検査します。この場合、**etc/radius/clients** ファイルから読み取られた情報が検査されます。

デーモンはプールから未使用アドレスの取得を試みます。 未使用アドレスがある場合、そのアドレスは NAS-IP フラグおよび NAS-port フラグにより「使用中」のマークを付けられ、RADIUS サーバーに戻されます。 この IP アドレスはデーモンにより **Framed-IP-Address** 属性を与えられ、受け入れパケットで NAS に戻されます。 **poolname.data** ファイルも更新されて、共有メモリー内の情報と同期されます。

プールが存在しない場合、またはプールが存在していても未使用アドレスが残っていない場合は、RADIUS サーバーにエラーが戻されます。 ログ・ファイルにエラー「`Could not allocate IP address`」が記録され、RADIUS サーバーにより Access-Reject パケットが NAS に送信されます。

エラー・コードは次のとおりです。

- NOT_POOLED – **nas_ip** に定義されているプールはない。
- POOL_EXHAUSTED – **nas_ip** にプールが定義されているが、そのプール内のアドレスは現在すべて使用中である。

既に IP アドレスが割り当てられている NAS と NAS-port の組み合わせから認証要求が出されると、デーモンは、IN USE フラグのマークをオフにし、テーブルから NAS-IP-address エントリーおよび NAS-port エントリーを消去して、前の割り当てをプールに戻します。次に、そのプールから新しい IP アドレスを割り当てます。

RADIUS サーバーが NAS から Accounting-Stop パケットを受信したときも、IP アドレスがプールに戻されます。 Accounting-Stop パケットには NAS-IP-address エントリーと NAS-port エントリーが含まれていなければなりません。 デーモンは、以下の場合に **ippool_mem** ファイルにアクセスします。

- 新しい IP アドレスを取得するための要求が出された場合。 IN USE フラグを true に設定します。
- Accounting-Stop パケットが受信された場合。 IN USE フラグを false に設定して IP アドレスを解放します。

いずれの場合も、共有メモリー・システム・コールにより、共有メモリー内のデータと **poolname.data** ファイル内のデータが同期されます。 システム管理者は、RADIUS サーバー構成ファイル (**radiusd.conf**) の *Enable_IP_Pooling* パラメーターを使用して、IP 割り当てを `ON` または `OFF` に切り替えることができます。 これは、システム管理者がグローバル **default.auth** ファイルまたは **user.auth** ファイルのいずれかで IP アドレスを割り当てている場合に便利です。割り当て済みの IP アドレスを使用するには、システム管理者は `Enable_IP_Pool = NO` を設定する必要があります。

SMIT を使用して作成された **/etc/radius/ippool_def** ファイルの一例:

プール名	開始範囲	終了範囲
Floor5	192.165.1.1	192.165.1.125
Floor6	192.165.1.200	192.165.1.253

SMIT を使用して作成された **/etc/radiusclients** ファイルの一例:

NAS-IP	共有秘密鍵	プール名
1.2.3.4	Secret1	Floor5
1.2.3.5	Secret2	Floor6
1.2.3.6	Secret3	Floor5
1.2.3.7	Secret4	

上の例で、NAS-IP-Address 1.2.3.7 のプール名はブランクです。 この場合、この NAS に関しては (グローバル `IP_pool_flag = True` であっても) IP プーリングは行われません。 Access-Request パケットが受信されると、RADIUS サーバーは認証および許可を実行します。正常に実行された場合は、要求で定義されている IP アドレス、あるいはグローバル **default.auth** ファイルまたは **user.auth** ファイルにある IP アドレスが Access-Accept パケットで送信されます。 この場合、NAS-Port 属性は不要です。

IP プーリングが `True` の場合は、静的 IP アドレスも、グローバル **default.auth** ファイルまたは **user.auth** ファイルの一部として、あるいは Access-Request パケットの一部として、システム管理者により定義されています。 RADIUS サーバーは、この IP アドレスを、該当 NAS 用の定義済みプール名から割り当てられた IP アドレスに置き換えます。 プール内のすべての IP アドレスが使用中の場合は、サーバーはエラー (pool is full) をログに記録し、Access-Reject パケットを送信します。サーバーは **auth** ファイルに定義されている静的 IP アドレスをすべて無視します。

NAS-IP から Access-Request パケットが受信されたときに IP プーリングが `True` であって、有効なプール名が NAS 用に定義されており、しかも NAS-Port が定義されていない場合は、サーバーは Access-Reject パケットを送信します。

デーモンによって作成された **Floor5.data** ファイルの一例を次に示します。

IP アドレス	NAS-IP	NAS-Port	In Use
192.165.1.1	1.2.3.4	2	1
192.165.1.2	1.2.3.4	3	0
.....	.....	....	....
192.165.1.124	1.2.3.6	1	1
192.165.1.125	1.2.3.6	6	1

デーモンによって作成された **Floor6.data** ファイルの一例を次に示します。

IP アドレス	NAS-IP	NAS-Port	In Use
192.165.200	1.2.3.4	1	1
192.165.201	1.2.3.4	4	1
.....	.....	....	....
192.165.1.252	1.2.3.4	5	0
192.165.1.253	1.2.3.4	6	1

指定された NAS についてすべての割り当て済み IP アドレスの解放が必要になると (例えば NAS が停止されると)、すべてのプールからすべての IP アドレスを解放して ***poolname*.data** ファイルを初期化する必要が生じることがあります。システム管理者は、SMIT で次のメニューを使用してこれを行うことができます。

- Clear IP Pool for a Client (クライアントの IP プールを消去する)
- Clear entire IP Pool (IP プール全体を消去する)

IP プール用 SMIT パネル

「Client Configuration (クライアント構成)」の「**Add a Client (**クライアントの追加**)**」では、オプションとして「**Pool Name (**プール名**)**」を入力することができます。この名前には最大 64 文字を使用できます。 「**Pool Name (**プール名**)**」をブランクにすると、IP プーリングは実行されず、RADIUS サーバーは、**Framed-IP-Address** 許可属性を使用してシステム管理者が定義した IP アドレスを割り当てます。

「**IP Pool (IP** プール**)**」を選択すると、以下のオプションが表示されます。

- List all IP Pools (IP プールをすべてリスト)
- Create an IP Pool (IP プールを作成)
- Change/Show Characteristics of an IP Pool (IP プールの特性の変更/表示)
- Delete an IP Pool (IP プールを削除)
- Clear IP Pool for a Client (クライアントの IP プールを消去する)
- Clear entire IP Pool (IP プール全体を消去する)

「**List all IP Pools (IP** プールをすべてリスト**)**」: このオプションは、「**Pool Name (**プール名**)**」、「**Start Range IP address (**開始範囲 **IP** アドレス**)**」、および「**Stop Range IP address (**停止範囲 **IP** アドレス**)**」をリストするために使用します。

「**Create an IP Pool (IP** プールを作成**)**」: このオプションは、プール名、開始範囲、および終了範囲を追加するために使用します。このデータは、**ippool_def** ファイルの下部に追加されます。プール名が重複しないように、また IP アドレスの範囲が排反しないように、検査が行われます。 このアクションは、RADIUS サーバー・デーモンが実行中でないときにのみ実行できます。

「**Change/Show Characteristics of an IP Pool (IP** プールの特性の変更/表示**)**」: このオプションは、プール名のリストをポップアップ・パネルで表示します。 このパネルから特定プール名を選択する必要があります。 プール名を選択すると、選択された名前のパネルが表示されます。 Enter を押すと、**ippool_def** ファイル内のそのプール名に関するデータが更新されます。 このアクションは、RADIUS サーバー・デーモンが実行中でないときにのみ実行できます。

「**Delete an IP Pool (IP** プールを削除**)**」: このオプションを選択すると、選択できるプール名のリストが表示されます。 プール名を選択すると、「**Are You Sure (**よろしいですか**?)**」というポップアップ・パネルが表示され、選択されたプール名を削除する前に確認することができます。 **rmippool** スクリプトが呼び出されて、選択されたプール名が **ippool_def** ファイルから削除されます。このアクションは、RADIUS サーバー・デーモンが実行中でないときにのみ実行できます。

「**Clear IP Pool for a Client (**クライアントの **IP** プールを消去する**)**」: このオプションを選択すると、該当 NAS に属する IP アドレスの **IN-USE** エントリーに `0` のマークが付けられます。これはこの NAS の IP アドレスがすべて使用可能になったことを意味します。 このアクションは、RADIUS サーバー・デーモンが実行中でないときにのみ実行できます。

「**Clear Entire IP Pool (IP** プール全体を消去**)**」: このオプションを選択すると、「**Are You Sure (**よろしいですか**?)**」というポップアップ・パネルが表示され、**ippool_mem** ファイル全体を削除する前に確認することができます。このアクションは、RADIUS サーバー・デーモンが実行中でないときにのみ実行できます。

RADIUS SMIT パネル

SMIT を使用して RADIUS サーバーを構成する場合、アスタリスク (*) でマークされたフィールドは必要フィールドです。

SMIT 高速パスは、次のとおりです。

```
smitty radius
```

RADIUS メインメニューは、次のとおりです。

```
       RADIUS Server

Configure Server
Configure Clients
Configure Users
Configure Proxy Rules
Advanced Server Configuration
Start RADIUS Server daemons
Stop RADIUS Server daemons
```

次の画面取りは、RADIUS の「Configure Server (サーバーの構成)」SMIT パネルのサンプルです。

```
Configure Server
  RADIUS Directory                          /etc/radius
* Database Location                         [Local]                  +
  Local AVL Database File Name              [dbdata.bin]
  Debug Level                               [9]                      +#
  Local Accounting                          [ON]                     +
  Local Accounting Directory                [/var/radius/data/accou>
  Accept Reply-Message                      []
  Reject Reply-Message                      []
  Challenge Reply-Mesage                    []
  Password Expired Reply-Message            []
  Support Renewal of Expired Password       [NO]                     +
  Require Message Authenticator             [NO]                     +
* Authentication Port Number                [1812]
* Accounting Port Number                    [1813]
  LDAP Server Name                          []
  LDAP Server Port Number                   [389]                     #
  LDAP Server Admin Distinguished Name      [cn=root]
  LDAP Server Admin Password                []
  LDAP Base Distinguished Name              [cn=aixradius]
  LDAP Size Limit                           [0]                       #
  LDAP Hop Limit                            [0]                       #
  LDAP wait time limit                      [10]                      #
  LDAP debug level                          [0]                      +#
  Proxy Allowed                             [OFF]                    +
  Proxy Use Table                           [OFF]                    +
  Proxy Realm Name                          []
  Proxy Prefix Delimiters                   [$/]
  Proxy Suffix Delimiters                   [@.]
  Proxy Remove Hops                         [NO]                     +
  Proxy Retry Count                         [2]                       #
  Proxy Timeout                             [30]                      #
  UNIX Check Login Restrictions             [OFF]                    +
  Enable IP Pool                            [OFF]                    +
  Send Message Authenticator for ACCEPT     [ON]                     +
  Maximum RADIUS Server Threads             [15]                      #
  EAP Conversation Timeout (Seconds)        [30]                      #
  Enable EAP-TLS                            [ON]                     +
  Required Options for EAP-TLS
       Path to OpenSSL Library              [/opt/freeware/lib/libs>
       OpenSSL Cipher List                  [ALL:!ADH:RC4+RSA:+SSLv>
       Root CA Directory (Full Path)        [/etc/radius/tls]
       Root CA Certificate (Full Path)      [/etc/radius/tls/radius>
       RADIUS Server Certificate (Full Path) [/etc/radius/tls/cert-s>
       RADIUS Server Private Key (Full Path) [/etc/radius/tls/cert-s>
       RADIUS Server CRL (Full Path)        []
```

F1 キーを押すと、すべてのフィールドとメニュー・オプションに関する詳細な SMIT ヘルプ情報が表示されます。

乱数発生ルーチン

乱数は、RADIUS パケットのオーセンティケーター・フィールドを生成する場合に必要です。

可能な限り最高性能の乱数発生ルーチンを提供することが重要です。これは、侵入者が RADIUS サーバーをだまして、予想される要求に応答するように仕掛け、その応答を利用して、それ以降の access-request に対してその RADIUS サーバーになりすまそうとする可能性があるためです。 AIX RADIUS サーバーは **/dev/urandom** カーネル・エクステンションを使用して、疑似乱数を生成します。 このカーネル・エクステンションは、疑似デバイス・ドライバーを利用して、ハードウェア・ソースからエントロピーのサンプルを収集します。 このデバイスは、NIST テストに合格して適正なランダム性が証明されたものです。

グローバリゼーションへの対応

RADIUS の **raddbm** コマンドおよび SMIT パネルはグローバリゼーションに対応しており、いずれも標準の AIX グローバリゼーション API 呼び出しを使用してこの機能を提供します。

関連情報

コマンド: **installp**、**mkuser** および **raddbm**

AIX 侵入防止

AIX 侵入防止は、不適切なデータや無許可のデータ、またはシステムに有害と判断されるその他のデータを検出します。

次のセクションでは、AIX で提供される各種の侵入検出機能について説明します。

関連情報

コマンド: **chfilt**、**ckfilt**、**expfilt**、**genfilt**、**impfilt**、**lsfilt**、**mkfilt**、**mvfilt**、**rmfilt**

侵入検出

「侵入検出」とは、システムへの無許可アクセスの試みをインターセプトおよび拒否するために、システム・イベントをモニターおよび分析するアクションのことです。 AIX では、無許可アクセスや無許可アクセス試行の検出は、特定のアクションを監視し、それらのアクションにフィルター・ルールを適用することによって行われます。

注**:** 侵入検出を使用可能にするには、ホスト・システムに **bos.net.ipsec** ファイルセットをインストールする必要があります。 この検出テクノロジーは、既存の AIX インターネット・プロトコル・セキュリティー (IPsec) フィーチャーに基づいて構築されています。

パターン・マッチング・フィルター・ルール**:**

パターン・マッチングとは、IPsec フィルター・ルールを使用してネットワーキング・パケットをフィルター処理することです。 フィルター・パターンは、テキスト文字列の場合、16 進数文字列の場合、または複数のパターンを含むファイルの場合があります。 パターンのフィルター・ルールが設定され、ネットワーク・パケット本体の中にそのパターンが検出されると、事前定義されたフィルター・ルールのアクションが実行されます。

パターン・マッチング・フィルター・ルールは、インバウンド・ネットワーク・パケットのみに適用されます。フィルター・ルール・テーブルにフィルター・ルールを追加するには、**genfilt** コマンドを使用します。 このコマンドによって生成されたフィルター・ルールは、マニュアル・フィルター・ルールと呼ばれます。 フィルター・ルールを活動化または非活動化するには、**mkfilt** コマンドを使用します。 **mkfilt** コマンドは、フィルター・ロギング機能を制御するためにも使用することができます。

パターン・ファイルには、テキスト・パターンまたは 16 進数パターンのリストを 1 行に 1 つずつ入れることができます。パターン・マッチング・フィルター・ルールは、ウィルス、バッファーのオーバーフロー、そしてその他のネットワーク・セキュリティー・アタックから保護するために使用できます。

パターン・マッチング・フィルター・ルールは、その使用範囲が広すぎたり、パターンの数が多すぎると、システム・パフォーマンスに悪影響を及ぼす場合があります。 このルールの適用の範囲はできるだけ狭くしておくのが最良です。 例えば、既知のウィルス・パターンを **sendmail** に適用する場合は、フィルター・ルールに **sendmail** SMTP 宛先ポート 25 を指定します。 こうすると、他のトラフィックはすべて通過するため、パターン・マッチングによるパフォーマンス低下は避けられます。

genfilt コマンドは、ClamAV の一部のバージョンで使用されるパターン・フォーマットを認識し、理解します。

関連情報:

genfilt コマンド

mkfilt コマンド

 ClamAV Web サイト

パターンのタイプ:

パターンの基本タイプには、テキスト、16 進数、およびファイルの 3 つがあります。 パターン・マッチング・フィルター・ルールは、着信パケットのみに適用されます。

テキスト・パターン

テキスト・フィルター・パターンは、次のような ASCII 文字列です。

```
GET /../../../../../../../../
```

16 進数パターン

16 進数パターンは、次のようになります。

```
0x33c0b805e0cd16b807e0cd1650558becc7460200f05d0733ffb8c800b9fffff3abb00150
e670e47132c0e67158fec03c8075f033c033c9b002fa99cd26fb4183f90575f5c3
```

注: 16 進数パターンは、先頭の 0x によってテキスト・パターンと見分けられます。

テキスト・パターンが入っているファイル

ファイルには、テキスト・パターンまたは 16 進数パターンのリストを 1 行に 1 つずつ入れることができます。 サンプルのパターン・ファイルは、http://www.clamav.net にあります。

shun ポートおよび **shun** ホストのフィルター・ルール:

shun フィルター・ルールを設定すると、リモート・ホスト、またはリモート・ホストとポートのペアがローカル・マシンへアクセスしないよう作用することができます。

shun フィルター・ルールは、ルールの指定基準が満たされた場合にリモート・ホストまたはリモート・ホストとポートのペアからのローカル・マシンへのアクセスを拒否する、実効ルールを動的に作成します。

一般にアタックの前にはポート・スキャンが行われるため、shun ポートのフィルター・ルールは、このアタックの動作を検出して侵入を防止するという点で特に有効です。

例えば、ローカル・ホストがサーバー・ポート 37 (すなわち、タイム・サーバー) を使用しない場合、リモート・ホストはポート・スキャンを実行していない限り、ポート 37 にアクセスするはずがありません。shun ポートのフィルター・ルールをポート 37 に適用します。この場合、リモート・ホストがそのポートにアクセスしようとすると、shun フィルター・ルールが実効ルールを作成して、shun ルールの「**expiration time (**満了時刻**)**」フィールドに指定された時間の間、そのホストのそれ以降のアクセスがブロックされます。

shun ルールの「**expiration time (**満了時刻**)**」フィールドを 0 に設定すると、動的に作成された実効 shun ルールは満了しなくなります。

注:

1. shun ポートのフィルター・ルールによって指定された満了時刻は、動的に作成された実効ルールのみに適用されます。

2. 動的に作成された実効ルールは、**lsfilt -a** コマンドでのみ表示することができます。

shun ホスト・フィルター・ルール

shun ホストのフィルター・ルールの基準が満たされると、動的に作成された実効ルールによって、指定された満了時刻までの間、そのリモート・ホストからのすべてのネットワーク・トラフィックがブロックまたは shun (回避) されます。

shun ポート・フィルター・ルール

shun ポートのフィルター・ルールの基準が満たされると、動的に作成された実効ルールによって、満了時刻が過ぎるまで、このリモート・ホストの特定のポートからのネットワーク・トラフィックのみがブロックまたは shun (回避) されます。

ステートフル・フィルター・ルール**:**

ステートフル・フィルターは、ソース・アドレスと宛先アドレス、ポート番号、および状況などの情報を検査します。 その後に、IF、ELSE、または ENDIF フィルター・ルールをこれらのヘッダー・フラグに適用することにより、ステートフル・システムが、個々のパケットとそのヘッダー情報のコンテキストではなく、セッション全体のコンテキストでフィルター処理をするかどうかを判断できます。

ステートフル・インスペクションでは、着信および発信の通信パケットが検査されます。ステートフル・フィルター・ルールを **mkfilt -u** コマンドで活動化すると、IF ルールが満たされるまで、常に ELSE ブロック内のルールが検査されます。 IF ルールまたは条件が満たされた後は、フィルター・ルールが **mkfilt -u** コマンドによって再活動化されるまで、IF ブロック内のルールが使用されます。

ckfilt コマンドは、ステートフル・フィルター・ルールの構文を検査して、次の図のようにルールを表示します。

```
%ckfilt -v4
Beginning of IPv4 filter rules.
Rule 2
IF Rule 3
    IF Rule 4
        Rule 5
    ELSE Rule 6
        Rule 7
    ENDIF Rule 8
ELSE Rule 9
    Rule 10
ENDIF Rule 11
Rule 0
```

時刻指定ルール**:**

時刻指定ルールは、フィルター・ルールが **mkfilt -v [4|6] -u** コマンドで発効した後にそのフィルター・ルールが適用される時間を秒数で指定します。

満了時刻は **genfilt -e** コマンドで指定します。詳しくは、『**mkfilt** コマンド』および『**genfilt** コマンド』を参照してください。

注**:** タイマーは IF、ELSE、または ENDIF ルールには効果を及ぼしません。 shun ホストまたは shun ポートのルールで満了時刻が指定された場合、その時刻はその shun ルールによって開始された実効ルールのみに適用されます。 shun ルールには満了時刻はありません。

SMIT からのフィルター・ルールへのアクセス

SMIT からルールを構成することができます。

SMIT からフィルター・ルールを構成するには、次のステップを実行します。

1. コマンド・ラインから、次のコマンドを入力します。`smitty ipsec4`
2. 「**Advanced IP Security Configuration (**拡張 **IP** セキュリティー構成**)**」を選択します。
3. 「**Configure IP Security Filter Rules (IP** セキュリティー・フィルター・ルールの構成**)**」 を選択します。
4. 「**Add an IP Security Filter Rule (IP** セキュリティー・フィルター・ルールの追加**)**」を選択します。

```
                        Add an IP Security Filter Rule

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                 [Entry Fields]
* Rule Action                                        [permit]                +
* IP Source Address                                  []
* IP Source Mask                                     []
  IP Destination Address                             []
  IP Destination Mask                                []
* Apply to Source Routing? (PERMIT/inbound only)     [yes]                   +
* Protocol                                           [all]                   +
* Source Port / ICMP Type Operation                  [any]                   +
* Source Port Number / ICMP Type                     [0]                      #
* Destination Port / ICMP Code Operation             [any]                   +
* Destination Port Number / ICMP Type                [0]                      #
* Routing                                            [both]                  +
* Direction                                          [both]                  +
* Log Control                                        [no]                    +
* Fragmentation Control                              [0]                     +
* Interface                                          []                      +
  Expiration Time (sec)                              []                       #
  Pattern Type                                       [none]                  +
  Pattern / Pattern File                             []
  Description                                        []


Where "Pattern Type" may be one of the following
  x   none                                                                   x#
  x   pattern                                                                x
  x   file                                                                   x
  x   Anti-Virus patterns
```

「action (アクション)」フィールドの選択項目は、
`permit`、`deny`、`shun_host`、`shun_port`、`if`、`else`、`endif` です。

パターン・ファイルを指定する場合は、フィルター・ルールを **mkfilt-a** コマンドで活動化したときに読み取り可能でなければなりません。 フィルター・ルールは /etc/security/ipsec_filter データベースに保管されます。

AIX Security Expert

AIX Security Expert は、すべてのセキュリティー設定 (TCP、NET、IPSEC、システム、および監査) のためのセンターを提供します。

AIX Security Expert は、システム・セキュリティーの強化ツールです。 これは **bos.aixpert** ファイルセットの一部です。 AIX Security Expert は、高レベル・セキュリティー、中レベル・セキュリティー、低

レベル・セキュリティーのための単純なメニュー設定のほか、300 を超えるセキュリティー構成設定を組み込みつつ、上級管理者にはそれぞれのセキュリティー要素の制御を提供する AIX 標準設定セキュリティーを提供しています。 AIX Security Expert を使用することにより、セキュリティー強化についての大量の資料を読んで、個別にそれぞれのセキュリティー要素を実装する必要がなく、適切なレベルのセキュリティーを実装することができます。

AIX Security Expert を使用すると、セキュリティー構成のスナップショットを取ることができます。 このスナップショットは、他のシステム上に同じセキュリティー構成をセットアップするのに使用できます。これにより、時間を節約し、全社的な環境ですべてのシステムが適切なセキュリティー構成を持つようにすることができます。

AIX Security Expert は SMIT から実行できます。または **aixpert** コマンドを使用できます。

AIX Security Expert 設定

大きく分けて、以下のようなセキュリティー設定が使用可能です。

高レベル・セキュリティー
: 高レベルのセキュリティー

中レベル・セキュリティー
: 中レベルのセキュリティー

低レベル・セキュリティー
: 低レベルのセキュリティー

拡張セキュリティー
: カスタムのユーザー指定のセキュリティー

AIX 標準設定
: オリジナルのシステム・デフォルトのセキュリティー

セキュリティーを元に戻す
: 一部の AIX Security Expert 構成設定は元に戻すことができます。

セキュリティーの確認
: 現行のセキュリティー設定の詳細なレポートを提供します。

AIX Security Expert セキュリティー強化

セキュリティー強化は、セキュリティーを引き締めたり、より高いレベルのセキュリティーを実装することにより、システムのあらゆる要素を保護します。

セキュリティー強化は、すべてのセキュリティー構成の決定および設定が妥当で適切なものであることを保証するのを助けるものです。 AIX システムのセキュリティーを強化するために、多くのセキュリティー構成設定の変更が必要になる場合があります。

AIX Security Expert では、効果的な共通セキュリティー構成設定を集中管理するためのメニューを提供しています。これらの設定は、適切に保護された UNIX システムに関する広範囲にわたる調査を基にしています。 広範な安全保護環境を対象としたデフォルト・セキュリティー設定 (高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティー) が提供されており、上級管理者は個別にそれぞれのセキュリティー構成設定を設定することができます。

システムを構成する際、設定するセキュリティー・レベルが高過ぎると、必要なサービスが拒否されることがあります。 例えば、**telnet** と **rlogin** は、ログイン・パスワードが暗号化されずにネットワーク上で送

信されるため、高レベル・セキュリティーでは使用不可になります。 システムを構成する場合にセキュリティー・レベルが低すぎると、システムはセキュリティーの脅威に対してぜい弱になる可能性があります。各企業には各社固有のセキュリティー要件があるので、事前定義の高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティーの構成設定は、特定の企業のセキュリティー要件に完全一致するものというよりは、セキュリティー構成を行うための開始点として使用するのに最適です。

AIX Security Expert の使用による実践的なアプローチとしては、デプロイされる実稼働環境と同様にテスト・システムを (実際のテスト環境に) 設定することです。 インストールには業務アプリケーションが必要であり、GUI を使用して AIX Security Expert を実行します。 AIX Security Expert はトラステッド状態で、この稼働システムを分析します。 選択したセキュリティー・オプションに応じて、 AIX Security Expert は、他のセキュリティー設定と同時に、スキャン保護が使用可能にされるポートを使用可能にするか、監査をオンにするか、業務アプリケーションまたはその他のサービスで使用されていないネットワーク・ポートをブロックします。 これらのセキュリティー構成が配置される場所で再テストを実施することにより、システムの実稼働環境へのデプロイの準備が整います。 また、このシステムのセキュリティー・ポリシーまたは構成を定義する AIX Security Expert XMLfile を使用すると、企業システムに類似したシステムにまったく同じ構成を実装することができます。

セキュリティー強化について詳しくは、「NIST Special Publication 800-70, NIST Security Configurations Checklist Program for IT Products」を参照してください。

デフォルトでの保護

デフォルトでの保護 (Secure By Default (SbD)) は、最小限のソフトウェア・セットを保護構成でインストールするという概念です。

AIX Secure by Default (SbD) のインストール・オプションでは、ぜい弱なコマンドおよびファイルを取り除くための TCP クライアントおよびサーバーのライト・バージョンをインストールします。 **bos.net.tcp.client** および **bos.net.tcp.server** ファイルセットは SbD インストールの一部であり、 **telnet** および **ftp** のような平文フォーマットでネットワーク上のパスワードの伝送が可能なアプリケーションの場合を除いて、これらのファイルセットにはすべてのコマンドとファイルが含まれます。 さらに、**rsh**、**rcp**、および **sendmail** など、使用されている可能性のあるアプリケーションは、SbD ファイルセットから除外されます。

SbD のインストールの自動化された最後のプロセスは、AIX Security Expert 高水準セキュリティー構成の設定の組み込みです。 これを行うには、`/etc/firstboot` スクリプトの「`/usr/sbin/aixpert -f /etc/security/aixpert/core/SbD.xml -p 2>/etc/security/aixpert/log/firstboot.log`」を使用して **aixpert** コマンドを実行します。

bos.net.tcp.client および **bos.net.tcp.server** ファイルセットを再度インストールすることにより、ODM 変数の *SbD_STATE* を sbd_disable に変更し、 AIX Security Expert を使用してシステムをデフォルトのセキュリティー・レベルにすることにより、マシンを SbD モード以外のモードにすることができます。

移行インストールまたは保存インストールを使用して、SbD インストール済みシステムを実現することはできません。 SbD は個別インストール・メニュー・パスです。

注**:** SbD モードのシステムを Service Pack で更新すると、更新後、更新されたシステムは SbD モードではありません。

SbD インストール・オプションを使用せずに安全にシステムを構成することは可能です。 例えば、AIX Security Expert の高レベル、中レベル、または低レベルのセキュリティー・オプションを、通常のインストールで構成することができます。

SbD インストール済みシステムと AIX Security Expert 高レベル・セキュリティー構成による通常のインストールとの違いは、**telnet** コマンドの検査によって最も明確に示されます。 どちらの場合も **telnet** コマンドは使用不可です。 SbD インストールでは、**telnet** バイナリーまたはアプリケーションはシステム上にインストールされることは決してありません。

SbD インストールが使用される場合、以下のサービスがインストール時にシステムにインストールされないか、使用不可にされます。これらのサービスの一部がシステムにインストールされないことにより、これらのコマンドをシステムからアクセスまたは実行することができません。 これらのコマンドおよびプログラムが必要な場合には、SbD インストール・オプションを使用しないでください。さらに、スクリプト、リモート・プログラム、または従属ファイルセットのいずれかでこれらのコマンドおよびプログラムが必要な場合には、 SbD インストール・オプションを使用しないでください。

サービス	プログラム	引数
bootps	/usr/sbin/bootpd	bootpd/etc/bootp
comsat	/usr/sbin/comsat	comsat
exec	/usr/sbin/rexecd	rexecd
finger	/usr/sbin/fingerd	fingerd
ftp	/usr/sbin/ftpd	ftpd
instsrv	/u/netinst/bin/instsrv	instsrv -r /tmp/netinstalllog /u/netinst/scripts
login	/usr/sbin/rlogind	rlogind
netstat	/usr/bin/netstat	netstat -f inet
ntalk	/usr/sbin/talkd	talkd
pcnfsd	/usr/sbin/rpc.pcnfsd	pcnfsd
rexd	/usr/sbin/rpc.rexd	rexd
rquotad	/usr/sbin/rpc.rquotad	rquotad
rstatd	/usr/sbin/rpc.rstatd	rstatd
rusersd	/usr/lib/netsvc/rusers/rpc.rusersd	rusersd
rwalld	/usr/lib/netsvc/rwall/rpc.rwalld	rwalld
shell	/usr/sbin/rshd	rshd
sprayd	/usr/lib/netsvc/spray/rpc.sprayd	sprayd
systat	/usr/bin/ps	ps -ef
talk	/usr/sbin/talkd	talkd
telnet	/usr/sbin/telnetd	telnetd -a
tftp	/usr/sbin/tftpd	tftpd -n
uucp	/usr/sbin/uucpd	uucpd

IBM Systems Director Console for AIX に、HealthMetrics ポートレットなどの一部の機能も存在します。これらは、SbD モードで AIX オペレーティング・システムを実行する場合は使用不可です。これらの機能を使用可能にするには、その機能の実行に必要なファイルセットをインストールします。

LDAP を使用したセキュリティー・ポリシーの配布

LDAP を使用して、AIX Security Expert XML 構成ファイルを配布することができます。 AIX Security Expert を使用すると、1 つのシステムから別のシステムにセキュリティー構成をコピーすることができま

す。 これにより、類似のシステムが同じセキュリティー構成を保持することができます。 このような整合性があるため、セキュリティー上のぜい弱性を低減させることができます。

お勧めする実行方法は、AIX Security Expert を使用して単一システムを構成し、共同のセキュリティー・ポリシーおよびシステムが働く環境に従ってセキュリティー・レベルを設定することです。 この構成は /etc/security/aixpert/core/appliedaixpert.xml ファイルに取り込まれます。続いて、このファイルは構成済みおよび信頼できる LDAP サーバーに移動することができます。 この LDAP サーバーと接続している他のシステムは、**aixpertldap** コマンドを使用してこの XML 構成ファイルを自動的に検出します。

既存の LDAP サーバーは、いずれも aixpert スキーマに従って更新され、接続されている各クライアントに aixpert 構成 XML ファイルを配布することができます。 LDAP サーバーに更新済みの aixpert スキーマがない場合は、コマンド ldapmodify -c -D <bindDN> -w <bindPwd> -i /etc/security/ldap/sec.ldif で aixpert スキーマを更新します。 LDAP サーバーがこの aixpert スキーマで更新されると、クライアントは **aixpertldap** コマンドの -u オプションを使用して、クライアントの XML 構成ファイルを LDAP に配置することができます。これらの構成ファイルは手動で更新する必要があります。

注: この機能は信頼できるモデルの LDAP が適切な場所に配置されているかどうかに依存しています。 LDAP への書き込み特権を持っているユーザーは、別のマシンのユーザーによってアップロードされたデータを変更することができます。同様に、LDAP クライアントにセキュリティー上のぜい弱性があると、このぜい弱性を悪用して、クライアントに関連付けられた AIX Security Expert XML 構成ファイルが読みとられることにより、他の LDAP クライアントのセキュリティー状況が読み取られ、その内容を知られてしまいます。

例えば、appliedaixpert.xml ファイルは、**BranchOfficeSecurityProfile** という名前で LDAP サーバー上に保存することができます。 あるいは、異なる方法で構成された appliedaixpert.xml ファイルは、**InternetDirectAttachedSystemsProfile** という名前で保存されることがあります。 LDAP 接続の他のシステムは AIX Security Expert を用いて構成されるため、これらのセキュリティー・プロファイルは自動的にメニュー・オプションとして表示されます。 これにより、システム管理者は、共同のセキュリティー・ポリシーのガイドラインの範囲内で環境に最適のセキュリティー・プロファイルを選択することができます。

次に、AIX Security Expert はシステムを保護するために使用されます。 システムに実装されたセキュリティー構成設定の完全なリストが /etc/security/aixpert/core/appliedaixpert.xml ファイルに取り込まれます。このファイルには、このシステムのためのセキュリティー・ポリシーが含まれています。 セキュリティー・ポリシーは AIX Security Expert 検査セキュリティー・オプションが使用されるときに比較されます。 また、このセキュリティー・ポリシーは、ご使用の IT 環境全体のシステム・セキュリティーの整合性を保持するために、他のシステムにコピーして適用することができます。 セキュリティー・ポリシーを他のシステムにコピーするには、2 つの方法 (手動による方法と LDAP を使用する方法) があります。

AIX Security Expert セキュリティー・ポリシーのコピー

1 つのシステムから別のシステムにセキュリティー・ポリシーをコピーする場合に、 AIX Security Expert を使用します。

1 つのシステムで AIX Security Expert を実行し、それと同じセキュリティー・ポリシーを別のシステムに適用することができます。 例えば、Bob は AIX Security Expert を彼の 6 台の AIX システムに適用したいと考えています。 彼は、高、中、低、拡張、または AIX 標準設定セキュリティーを使用して、1 つのシステム (Alpha) にセキュリティー設定を適用します。 彼は、使用環境内で互換性の問題がないかこのシステムをテストします。 これらの設定に問題がなければ、同じ設定を別の AIX システムに名前で適

用することができます。 /etc/security/aixpert/core/appliedaixpert.xml ファイルを Alpha から別のシステムにコピーすることにより、この設定をシステム Alpha から、同じセキュリティー設定を適用しようとするシステムにコピーします。

注: このファイルは他のシステム上にある同じディレクトリーおよびファイル名にコピーしないでください。 **aixpert** コマンドがセキュリティー・ポリシーを実装するときに /etc/security/aixpert/core/appliedaixpert.xml を上書きしてしまうからです。

代わりに、Alpha のセキュリティー・ポリシーを /etc/security/aixpert/custom/ ディレクトリーにコピーします。これにより、他のシステムから AIX Security Expertシステム管理 GUI を介するか、または直接 **aixpert** コマンドを使用して、Alpha のセキュリティー・ポリシーを表示および適用することができます。

例えば、Alpha の appliedaixpert.xml セキュリティー・ポリシーが /etc/security/aixpert/custom/*AlphaPolicy* として他のシステムに配置されていた場合、 aixpert -f /etc/security/aixpert/custom/*AlphaPolicy* コマンドは即時にこのセキュリティー・ポリシーを適用し、このシステムはマシン Alpha と同じセキュリティー構成を持つことになります。 さらに、Alpha のセキュリティー・ポリシーがこのディレクトリーにある場合、このポリシーは可視の状態であり、「Aix Security Expert」->「Overview and Tasks (概要およびタスク)」-> 「Customized Options (カスタマイズ・オプション)」->「AlphaPolicy」のパスを経由して、他のシステム・コンソールから、このポリシーを適用できます。

ユーザー定義の AIX Security Expert XML ルールを使用したカスタマイズ可能セキュリティー・ポリシー

XML ファイルを使用して固有のセキュリティー・ポリシーを構成することができます。

AIX Security Expert は、これらの XML ファイルを動的に認識します。作成されたカスタム XMLsecurity ポリシーはいずれも、記述ファイルを付けて /etc/security/aixpert/custom/ ディレクトリーに置かれます。 したがって、コンソール・グラフィカル・インターフェースを経由して AIX Security Expert をアクセスするときに、 aixpert DTD 内のグラフィカル XML フィーチャーのリッチ・セットをよく理解することができます。

DTD は、次のとおりです。

```
<?xml version='1.0'?>

<!--START-->

<!ELEMENT AIXPertSecurityHardening (AIXPertEntry+)>

<!-- AIXPertEntry should contain only one instance of the following elements. -->

<!ELEMENT AIXPertEntry (AIXPertRuleType,
  AIXPertDescription, AIXPertPrereqList, AIXPertCommand,
  AIXPertArgs,AIXPertGroup)>

<!-- AIXPertEntry's name should be unique. -->

<!ATTLIST AIXPertEntry
 name ID #REQUIRED
 function CDATA ""
>

<!ELEMENT AIXPertRuleType EMPTY>
<!ATTLIST AIXPertRuleType    type (LLS|MLS|HLS|DLS|SCBPS|Prereq) "DLS">
<!ELEMENT AIXPertDescription (#PCDATA)>
```

```
<!ELEMENT AIXPertPrereqList (#PCDATA)>
<!ELEMENT AIXPertCommand (#PCDATA)>
<!ELEMENT AIXPertArgs (#PCDATA)*>
<!ELEMENT AIXPertGroup (#PCDATA)*>
```

AIXPertEntry 名は XMLfile 内で固有の名前です。 この名前は、「Aix Security Expert」->「Overview and Tasks (概説およびタスク)」-> 「Customized Options (カスタマイズ・オプション)」->「<xml file=""></xml>」のパスを経由して、システム・コンソールから、このファイルを表示するときに、選択可能な図形ボタンの名前になります。

<!ELEMENT AIXPertRuleType EMPTY>
この XML ファイルはカスタムとして指定する必要があります。

<!ATTLIST AIXPertRuleType type (LLS|MLS|HLS|DLS|SCBPS|Prereq|Custom) "DLS"
この XML ファイルはカスタムとして指定する必要があります。

<!ELEMENT AIXPertDescription (#PCDATA)>
上記グラフィカル・インターフェースを経由して表示される場合、記述テキストは、マウスでこのボタンをポイントしたときにポップアップ・ウィンドウに表示されます。

<!ELEMENT AIXPertPrereqList (#PCDATA)>
前提条件ルールをこのルールとして選択することは可能です。 前提条件ルールでは、aixpert がこのルールを実装する前に 0 を返さなければなりません。 この XML ファイルはグラフィカル・インターフェースを介して表示され、このルールは前提条件ルールが十分でない場合は、グレーに塗りつぶされます。 前提条件ルールを作成している場合、AIXPertRuleType は「Prereq」でなければなりません。

前提条件ルールの AIXPertDescription フィールドには、前提条件ルールを満たすために行うべきことを記述する必要があります。 「カスタム」ルールが、前提条件ルールのうちの 1 つが満たされていないため、グレーに塗りつぶされる場合、ユーザーには前提条件の記述用ポップアップ・ウィンドウが表示されます。このウィンドウでユーザーが前提条件を訂正するための必須項目を明らかにします。

<!ELEMENT AIXPertCommand (#PCDATA)>
このエレメントは絶対パスと aixpert がこのセキュリティー・ルールを実行するコマンドであり、例えば、/usr/bin/ls のように指定します。

<!ELEMENT AIXPertArgs (#PCDATA)*>
このエレメントには、上のコマンドの引数 (例えば、-1) を含める必要があります。

<!ELEMENT AIXPertGroup (#PCDATA)*>
aixpert ルールがグラフィカル・インターフェースから表示されるときに、そのセットをグループ分けすることは可能です。 例えば、ルールの共通セットには、すべて、「Network Security (ネットワーク・セキュリティー)」の AIXPertGroup 名を指定することが可能です。

ぜい弱なパスワードに対する厳しい検査

この AIX 機能は、パスワードの変更時にぜい弱なパスワードを検査します。 このオプションを AIX Security Expert とともに選択した場合は、ユーザーがパスワードを選択または変更するときに、この追加のパスワード検査が行われます。この検査によって、英語の辞書に記載されているワード、および最近の米国勢調査に基づく 1000 の最も一般的な米国のファーストネームが使用されないようにします。

AIX Security Expert によってサポートされる COBIT 制御目標

AIX Security Expert は、AIX のデフォルトおよび拡張セキュリティー設定 (高レベル、中レベル、および低レベル) に加えて、SOB-COBIT ベスト・プラクティス・セキュリティー・レベルをサポートします。

米国連邦議会は、企業により開示された財務情報の正確性と信頼性を改善することで投資家を保護する目的で、「Sarbanes-Oxley Act of 2002 (2002 年の Sarbanes-Oxley 法令)」を制定しました。 COBIT の制御目標フィーチャーは、この法律に準拠するために、システム管理者が IT システムの構成、保守、および監査を行う場合に役立ちます。 SOX 構成アシストは aixpert コマンド・ラインからアクセスされます。この機能は Sarbanes-Oxley 法令の SOX セクション 404 について支援しますが、 AIX Security Expert の SOX 構成アシストは、内部統制に関する SOX セクション 404 の COBIT 最良事例に則ったセキュリティー設定を自動的に実装します。さらに、AIX Security Expert は、システムが現在この方法で構成されているかどうかを監査員に報告するための SOX 監査フィーチャーを提供します。 このフィーチャーは、システム構成の自動化に関して、IT SOX 準拠に対する支援および監査プロセスの自動化に対する支援を可能にします。

SOX はどのようにして IT がセクション 404 を遵守しなければならないかについてガイダンスを提示していないため、 IT 業界では www.isaca.org/ に詳述されている既存の統制内容に集中して取り組んでいます。 具体的に言えば、IT の統制内容は「Control Objectives for Information and related Technology (COBIT)」で取り上げられているものです。

AIX Security Expert は、以下の制御目標をサポートします。

- パスワード・ポリシーの実施
- 違反およびセキュリティー活動状況のレポート
- ソフトウェアに関する悪意のある阻止、検出、および修正、ならびに無許可ソフトウェア
- 公衆網を用いたファイアウォール・アーキテクチャーおよび接続

AIX Security Expert は、各制御目標のもとで指定された属性のすべてをサポートするわけではありません。サポートされる属性およびそれぞれの制御目標は、以下の表に要約されています。

パスワード・ポリシーの実施

説明	セキュリティー設定
パスワードの最大経過日数	maxage=13
パスワード・ヒストリーの適用	histsize=20
パスワードの最小経過日数	minage=1
パスワードの最小の長さ	minlen=8
少なくとも 6 文字を含む	Minalpha=6
旧パスワードとの類似性	mindiff=4
パスワード有効期限の警告日数	pwdwarntime=14

セキュリティー違反および活動状況のレポート

説明	セキュリティー設定	注釈
監査使用可能	yes	
直接 root ログインなし	yes	
特権拡大のための監査を使用可能にする	yes	AIXpert は USER_SU 監査イベントを活用します。 このイベントがオンにされていることを確認してください。

ソフトウェアに関する悪意のある検出および修正

AIX Security Expert は、AIX トラステッド・ソフトウェア実行機能を利用して、ソフトウェアがだれにも改ざんされないようにします。 **trustchk** コマンドは、トラステッド・ソフトウェア・データベースに登録されているオブジェクトの整合性を検査します。

ファイアウォール設定

AIX Security Expert は、IPSec をオンにし、ポート・スキャンを避けるためにフィルター・ルールを使用可能にします。 回避されるポートを次の表に示します。

サービス	説明
Tcp/11、udp/11	Systat
Tcp/13、udp/13	日中
(RFC 867) Tcp/19、udp/19	文字生成プログラム
Tcp/25	Simple Mail Transfer (SMTP)
Tcp/43、udp/43	Who Is (ニックネーム)
Tcp/63、udp/63	Whois++
Tcp/67、udp/67	ブートストラップ・プロトコル・サーバー (bootps)
Tcp/68、udp/68	ブートストラップ・プロトコル・クライアント (bootpc)
Tcp/69、udp/69	小規模ファイル転送
(tftp) Tcp/79、udp/79	Finger
Tcp/87	Private Terminal Link
Tcp/110	Post Office Protocol – バージョン 3 (POP3)
Udp/111	SUN Remote Procedure Call
Tcp/113	認証サービス (auth)
Udp/123	Network Time Protocol
Udp/161	SNMP
Udp/162	SNMPTRAP
Tcp/194	Internet Relay Chat Protocol
Tcp/443	TLS/SSL による HTTP プロトコル
Tcp/511	PassGo
Tcp/514	Cmd (シェル)
Tcp/520	拡張ファイル・ネーム・サーバー (efs)
Tcp/540	Uucpd (uucp)
Tcp/546	DHCPv6 クライアント
Tcp/547	DHCPv6 サーバー
Tcp/555	Dsf
tcp/559	TEEDTAP
tcp/593	HTTP RPC Ep Map
udp/635	RLS Dbase
tcp/666	Mdqs
tcp/777	Multiling HTTP
tcp/901	SNMPNSMERES

サービス	説明
tcp/902	IDEAFARM-CHAT
tcp/903	IDEAFARM-CATCH
tcp/1024	予約済み

AIX Security Expert を使用した COBIT 制御目標の適用

SCBPS レベルをシステムに適用する場合は、**aixpert –l s** コマンドを使用します。 このための監査ログは、**AIXpert_apply** イベントをオンにして生成することができます。 すべての障害 (前提条件障害または適用障害のいずれか) は、**stderr** および監査サブシステム (使用可能な場合) に報告されます。

SOX-COBIT 準拠の検査、監査、および事前監査機能

システムの SOX-COBIT 準拠を検査する場合は、 **aixpert –c –l s** コマンドを使用します。 AIX Security Expert は、サポートされる制御目標準拠のみを検査します。 この検査中に検出された違反はすべて報告されます。 デフォルトでは、違反はすべて **stderr** に送信されます。

SOX-COBIT 準拠監査レポートを生成する場合も、同じコマンド (**aixpert –c –l s**) を使用します。監査レポートを生成するには、監査サブシステムをセットアップして使用可能にします。 **AIXpert_check** 監査イベントがオンにされていることを確認してください。 監査サブシステムをセットアップした後で、**aixpert –c –l s** コマンドを再実行します。 このコマンドは、失敗したすべての制御目標の監査ログを生成します。 監査ログの **Status** フィールドには、**failed** というマークが付けられます。 このログには、失敗の理由も含まれます。このログは、**auditpr** コマンドの **-v** オプションを使用して表示することができます。

–p オプションを **aixpert –c –l s** コマンドに追加すると、監査レポートに成功した制御目標も含められます。 これらのログ・エントリーの状況フィールドには **Ok** が入れられます。

aixpert -c -l s -p コマンドを使用して、詳細な SOX-COBIT 準拠監査レポートを生成することができます。

–p オプションの指定の有無にかかわらず、要約レコードが生成されます。 要約レコードには、処理されたルールの数、失敗したルール (検出された非準拠のインスタンス) の数、およびシステムが検査されるセキュリティー・レベル (この場合は SCBPS となる) に関する情報が含まれます。

AIX Security Expert パスワード・ポリシー・ルールのグループ

AIX Security Expert は、パスワード・ポリシーに対する特定のルールを提供します。

強力なパスワード・ポリシーは、システム・セキュリティーを達成するための基盤となるものです。 パスワード・ポリシーは、パスワードが推測しにくく (パスワードに英数字、数字、および特殊文字が適切に混合されている)、定期的に有効期限切れがあり、有効期限切れ後の再利用を不可にします。 以下の表は、それぞれのセキュリティー設定のパスワード・ポリシー・ルールをリストしたものです。

表 20. *AIX Security Expert* パスワード・ポリシー・ルール

アクション・ボタン名	定義	**AIX Security Expert** が設定する値	元に戻す
最小文字数	/etc/security/user の **mindiff** 属性に適切な値を設定します。これは、新規パスワードで必要とされる、旧パスワードで使用されていない文字の最小文字数を指定します。	高レベル・セキュリティー 4 中レベル・セキュリティー 3 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい
パスワードの最小経過期間	/etc/security/user の **minage** 属性に適切な値を設定します。これは、パスワードの変更が可能になるまでの、週の最小数を指定します。	高レベル・セキュリティー 1 中レベル・セキュリティー 4 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい
パスワードの最大経過期間	/etc/security/user の **maxage** 属性に適切な値を設定します。これは、パスワードの変更が可能になるまでの週の最大数を指定します。	高レベル・セキュリティー 13 中レベル・セキュリティー 13 低レベル・セキュリティー 52 **AIX** 標準設定 制限なし	はい
パスワードの最小文字数	/etc/security/user の **minlen** 属性に適切な値を設定します。これは、パスワードの最小文字数を指定します。	高レベル・セキュリティー 8 中レベル・セキュリティー 8 低レベル・セキュリティー 8 **AIX** 標準設定 制限なし	はい
英字の最小文字数	/etc/security/user の **minalpha** 属性に適切な値を設定します。これは、パスワードの英字の最小文字数を指定します。	高レベル・セキュリティー 2 中レベル・セキュリティー 2 低レベル・セキュリティー 2 **AIX** 標準設定 制限なし	はい

表 20. *AIX Security Expert* パスワード・ポリシー・ルール (続き)

アクション・ボタン名	定義	**AIX Security Expert** が設定する値	元に戻す
パスワードのリセット時間	/etc/security/user の **histexpire** 属性に適切な値を設定します。これは、パスワードをリセットできるまでの週の最大数を指定します。	高レベル・セキュリティー 13 中レベル・セキュリティー 13 低レベル・セキュリティー 26 **AIX** 標準設定 制限なし	はい
パスワードに **1** つの文字を使用できる最大回数	/etc/security/user ファイルの **maxrepeats** 属性に適切な値を設定します。これは、パスワードに 1 つの文字を使用することのできる最大回数を指定します。	高レベル・セキュリティー 2 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 8	はい
パスワードの再利用時間	/etc/security/user の **histsize** 属性に適切な値を設定します。これは、ユーザーが再使用することのできない旧パスワードの数を指定します。	高レベル・セキュリティー 20 中レベル・セキュリティー 4 低レベル・セキュリティー 4 **AIX** 標準設定 制限なし	はい
有効期限後のパスワードの変更時間	/etc/security/user の **maxexpired** 属性に適切な値を設定します。これは、期限切れパスワードをユーザーが変更できる、**maxage** 後の、週の最大数を指定します。	高レベル・セキュリティー 2 中レベル・セキュリティー 4 低レベル・セキュリティー 8 **AIX** 標準設定 -1	はい
英字以外の文字の最小数	/etc/security/user の **minother** 属性に適切な値を設定します。これは、パスワードの英字以外の文字の最小数を指定します。	高レベル・セキュリティー 2 中レベル・セキュリティー 2 低レベル・セキュリティー 2 **AIX** 標準設定 制限なし	はい

表 20. *AIX Security Expert* パスワード・ポリシー・ルール (続き)

アクション・ボタン名	定義	**AIX Security Expert** が設定する値	元に戻す
パスワード有効期限の警告時間	/etc/security/user の **pwdwarntime** 属性に適切な値を設定します。これは、パスワード変更が必要であるという警告をシステムが発行するまでの日数を指定します。	高レベル・セキュリティー 5 中レベル・セキュリティー 14 低レベル・セキュリティー 5 **AIX** 標準設定 制限なし	はい

AIX Security Expert ユーザー・グループ・システムおよびパスワード定義のグループ

AIX Security Expert は、ユーザー、グループ、およびパスワード定義に対して特定のアクションを実行します。

表 21. *AIX Security Expert* ユーザー・グループ・システムとパスワード定義

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
グループ定義の確認	グループ定義の正確さを検査します。レポート・エラーを修正するには、次のコマンドを実行します。 `% grpck -y ALL`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 無効	いいえ
TCB 更新	**tcbck** コマンドを使用して、TCB を検査および更新します。 以下のコマンドを実行します。 `% tcbck -y ALL` **注:** ご使用のシステムで TCB が必須である場合、TCB が使用可能でないときこのルールは失敗します。 前提条件のルール (prereqtcb) も警告を出し、失敗します。 システムのインストール時に、前提条件の TCB を選択する必要があります。	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 はい	いいえ

ファイル定義の確認	**sysck** コマンドを使用して、/etc/objrepos/inventory のファイル・ベースを確認および修正します。
	`% sysck -i -f ¥`
	`/etc/security/sysck.cfg.rte`
		高レベル・セキュリティー
		はい
		中レベル・セキュリティー
		はい
		低レベル・セキュリティー
		はい
		AIX 標準設定
		無効
			いいえ

表 21. *AIX Security Expert* ユーザー・グループ・システムとパスワード定義 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
パスワード定義の確認	パスワード定義の正確さを検査します。レポート・エラーを修正するには、次のコマンドを実行します。 `% pwdck -y ALL`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 無効	いいえ
ユーザー定義の確認	ユーザー定義の正確さを検査します。 レポート・エラーを修正するには、次のコマンドを実行します。 `% usrck -y ALL`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 無効	いいえ

AIX Security Expert ログイン・ポリシー推奨のグループ

AIX Security Expert は、ログイン・ポリシーに対して特定の設定を提供します。

注**:** root が実行するセキュリティーに関連したアクティビティーの責任能力をより確実にするために、ユーザーは root としてログインするのではなく、まず通常のユーザー ID を使用してログインしてから、**su** *command* を使用して、root としてコマンドを実行することを推奨します。 その後システムは、複数のユーザーが root パスワードを認識して使用しているときに、root アカウントを使用して実行されるアクティビティーに異なるユーザーを関連付けることができます。

表 22. *AIX Security Expert* ログイン・ポリシー推奨

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ログイン失敗の間隔	適切な値を `/etc/security/login.cfg` の **logininterval** 属性に設定します。これはポートへのログイン試行失敗の時間間隔 (秒) を指定するもので、この時間間隔を過ぎるとポートは使用不可になります。 例えば、**logininterval** が 60 に設定され、**logindisable** が 4 に設定されている場合、1 分以内にログイン試行が 4 回失敗すると、アカウントは使用不可になります。	高レベル・セキュリティー 300 中レベル・セキュリティー 60 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい

表 22. *AIX Security Expert* ログイン・ポリシー推奨 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
アカウントのロックまでのログイン試行回数	/etc/security/user の **loginretries** 属性に適切な値を設定します。これは、アカウントが使用不可になるまでの、アカウントごとの連続ログイン試行回数を指定します。 root で設定しないでください。	高レベル・セキュリティー 3 中レベル・セキュリティー 4 低レベル・セキュリティー 5 **AIX** 標準設定 制限なし	はい
リモート **root** ログイン	/etc/security/user の **rlogin** 属性の値を変更します。これは、システムで root アカウントのリモート・ログインを許可するかどうかを指定します。	高レベル・セキュリティー False 中レベル・セキュリティー False 低レベル・セキュリティー 無効 **AIX** 標準設定 True	はい
ロック後のログインの再使用可能化	/etc/security/login.cfg の **loginreenable** 属性に適切な値を設定します。これは、ポートが **logindisable** によって使用不可にされ、その後アンロックされるまでの時間間隔 (秒) を指定します。	高レベル・セキュリティー 360 中レベル・セキュリティー 30 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい
ログイン試行の失敗後のログインの使用不可化	適切な値を /etc/security/login.cfg の **logindisable** 属性に設定します。これは、ポートがロックされるまでのポートでのログイン試行の失敗の回数を指定します。	高レベル・セキュリティー 10 中レベル・セキュリティー 10 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい
ログイン・タイムアウト	/etc/security/login.cfg の **logintimeout** 属性に適切な値を設定します。これは、パスワード入力に許可される時間間隔を指定します。	高レベル・セキュリティー 30 中レベル・セキュリティー 60 低レベル・セキュリティー 60 **AIX** 標準設定 60	はい

表 22. *AIX Security Expert* ログイン・ポリシー推奨 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ログイン失敗の間の遅延	/etc/security/login.cfg の **logindelay** 属性に適切な値を設定します。これは、ログイン失敗の間の遅延 (秒) を指定します。 ログインが失敗するたびに、追加の遅延期間が追加されます。 例えば、**logindelay** が 5 に設定されている場合、端末装置は最初のログイン失敗後、次の要求まで 5 分間待ちます。 2 度目のログインが失敗した後、端末装置は 10 秒間 (2*5) 待ち、3 度目のログインが失敗した後、端末装置は 15 秒間 (3*5) 待ちます。	高レベル・セキュリティー 10 中レベル・セキュリティー 4 低レベル・セキュリティー 5 **AIX** 標準設定 制限なし	はい
ローカル・ログイン	/etc/security/user の **login** 属性の値を変更します。これは、システム上で root アカウントに対してコンソール・ログインが許可されるかどうかを指定します。	高レベル・セキュリティー False 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 True	はい

AIX Security Expert 監査ポリシー推奨のグループ

AIX Security Expert では、特定の監査ポリシー設定を提供しています。

他のセキュリティー設定と同様に、bin 監査でも、高、中、または低レベル・セキュリティーの監査ルールを適用する前に、分析 (前提条件) ルールを満たしている必要があります。 bin 監査では、以下の分析ルールを満たしている必要があります。

1. 監査する前提条件ルールは、監査が現在実行中でないことを確認する必要があります。 監査が既に実行中の場合は、監査は以前に構成されているので、AIX Security Expert は既存の監査構成およびプロシージャーを変更することはできません。
2. 自動的にオンに変更されるボリューム・グループに少なくとも 100 メガバイトのフリー・スペースがあるか、現在 100 メガバイト以上のサイズの /audit ファイルシステムが存在していなければなりません。

上記の前提条件が満たされ、かつ、監査オプションが AIX Security Expert の中から選択されている場合、AIX Security Expert は、以下の方法でシステムに構成されて監査を使用可能にします。 AIX Security Expert の「**binaudit** の使用可能化」アクション・ボタンは、監査ポリシーを設定します。 システム上で監査を使用可能にする必要があります。

1. 監査の開始前に、/audit JFS ファイルシステムを作成してマウントする必要があります。 ファイルシステムは、少なくとも 100 メガバイトのサイズを持っていなければなりません。
2. 監査は bin モードで実行する必要があります。 /etc/security/audit/config ファイルは、次のように構成する必要があります。

```
start:
            binmode = on
            streammode = off
bin:
            trail = /audit/trail
            bin1 = /audit/bin1
```

```
            bin2 = /audit/bin2
            binsize = 10240
            cmds
= /etc/security/audit/bincmds
      .
      .
      etc
```

3. 高、中、および低レベル・セキュリティーでは、root および通常ユーザーに対する監査エントリーを追加します。
4. 高、中、および低レベル・セキュリティーでは、監査はリブート時に使用可能になる必要があります。
5. 作成される新規ユーザーの場合、高、中、および低レベル・セキュリティーに対して監査が使用可能になっている必要があります。 これは、auditclasses エントリーを /usr/lib/security/mkuser.default ファイルのユーザー・スタンザに追加することにより実行できます。
6. /audit ファイルシステムを満杯にしないようにするために、**cronjob** を追加する必要があります。

監査を元に戻すルールは、監査をシャットダウンして、リブート時にその使用可能性を除去する必要があります。

以下の表は、AIX Security Expert が「**binaudit** の使用可能化」に対して設定する値をリストしたものです。

表 23. 「*binaudit* の使用可能化」に対して *AIX Security Expert* が設定する値

高レベル・セキュリティー	中レベル・セキュリティー	低レベル・セキュリティー	**AIX** 標準設定
root および通常ユーザーに対して、次の監査エントリーを追加します。 root: General Src Mail Cron Tcpip Ipsec Lvm User: General Src Cron Tcpip 新規に作成されたユーザーの監査を使用可能にするために、/usr/lib/security/ mkuser.default ファイルのユーザー・スタンザに、以下のエントリーを追加します。 auditclasses=general,SRC,¥ cron,tcpip	root および通常ユーザーに対して、次の監査エントリーを追加します。 root: General Src Tcpip User: General Tcpip 新規に作成されたユーザーの監査を使用可能にするために、 /usr/lib/security/ mkuser.default ファイルのユーザー・スタンザに、以下のエントリーを追加します。 auditclasses=general, tcpip	root および通常ユーザーに対して、次の監査エントリーを追加します。 root: General Tcpip User: General 新規に作成されたユーザーの監査を使用可能にするために、 /usr/lib/security/ mkuser.default ファイルのユーザー・スタンザに、以下のエントリーを追加します。 auditclasses=general	/etc/security/audit/config ファイルには、以下のエントリーが含まれています。 default=login AUDIT (監査) クラス・ログインは、以下のように定義されます。 login = USER_SU, USER_Login, USER_Logout, TERM_Logout, USER_Exit **注:** 標準設定機能は監査を使用不可にします。

表 23. 「*binaudit* の使用可能化」に対して *AIX Security Expert* が設定する値 (続き)

高レベル・セキュリティー	中レベル・セキュリティー	低レベル・セキュリティー	**AIX** 標準設定
root および通常ユーザーに対して、次の監査エントリーを追加します。 **root:** general src mail cron tcpip ipsec lvm aixpert *User:* general src cron tcpip 新規に作成されたユーザーの監査を使用可能にするために、/usr/lib/security/ mkuser.default ファイルのユーザー・スタンザに、以下のエントリーを追加します。 `auditclasses=general,SRC, cron,tcpip`	root および通常ユーザーに対して、次の監査エントリーを追加します。 **root:** general src tcpip aixpert *User:* general tcpip 新規に作成されたユーザーの監査を使用可能にするために、 /usr/lib/security/ mkuser.default ファイルのユーザー・スタンザに、以下のエントリーを追加します。 `auditclasses=general, tcpip`	root および通常ユーザーに対して、次の監査エントリーを追加します。 **root:** general tcpip aixpert *User:* general 新規に作成されたユーザーの監査を使用可能にするために、 /usr/lib/security/ mkuser.default ファイルのユーザー・スタンザに、以下のエントリーを追加します。 `auditclasses=general`	はい

cronjob を毎時実行して、/audit のサイズを確認する必要があります。 Audit Freespace Equation が true の場合は、Audit Trail Copy Actions を実行しなければなりません。 Audit Freespace Equation は、/audit ファイルシステムが満杯にならないようにするために定義されているものです。/audit ファイル・システムが満杯になると、Audit Trail Copy Actions が実行されます (監査を使用不可にし、/audit/trail のバックアップを /audit/trailOneLevelBack に取ってから、監査を再度使用可能にします)。

AIX Security Expert /etc/inittab エントリーのグループ

AIX Security Expert は、システムのブート時に開始されないように、/etc/inittab の中の特定のエントリーをコメント化します。

表 24. *AIX Security Expert /etc/inittab* エントリー

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
qdaemon の使用不可化 / **qdaemon** の使用可能化	/etc/inittab ファイルにある次のエントリーをコメント化またはアンコメントします。 `qdaemon:2:wait:/usr/bin/startsrc –sqdaemon`	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい

表 24. *AIX Security Expert /etc/inittab* エントリー (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
lpd デーモンの使用不可化 / **lpd** デーモンの使用可能化	/etc/inittab ファイルにある次のエントリーをコメント化またはアンコメントします。 lpd:2:once:/usr/bin/startsrc -s lpd	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
CDE の使用不可化 / CDE の使用可能化	システムで LFT が構成されていない場合は、/etc/inittab ファイルにある次のエントリーをコメント化またはアンコメントします。 dt:2:wait:/etc/rc.dt	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
piobe デーモンの使用不可化 / **piobe** デーモンの使用可能化	/etc/inittab ファイルにある次のエントリーをコメント化またはアンコメントします。 piobe:2:wait:/usr/lib/lpd/pio/etc/pioinit >/dev/null 2>&1	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい

AIX Security Expert /etc/rc.tcpip 設定グループ

AIX Security Expert は、システムのブート時に開始されないように、/etc/rc.tcpip の中の特定のエントリーをコメント化します。

以下の表は、システムのブート時に開始されないように /etc/rc.tcpip の中でコメント化されるエントリーをリストしたものです。

表 25. *AIX Security Expert /etc/rc.tcpip* 設定

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
メール・クライアントの使用不可化 / メール・クライアントの使用可能化	/etc/rc.tcpip から次のエントリーをコメント化、またはアンコメントします。 start /usr/lib/sendmail "$src_running"	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい

表 25. *AIX Security Expert /etc/rc.tcpip* 設定 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
経路指定デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/routed "$src_running" -q	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
mrouted デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/mrouted "$src_running"	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
timed デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/timed	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 はい	はい
rwhod デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/rwhod "$src_running"	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
印刷デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/lpd "$src_running"	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

表 25. *AIX Security Expert /etc/rc.tcpip* 設定 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
SNMP デーモンの使用不可化/SNMP デーモンの使用可能化	/etc/rc.tcpip から次のエントリーをコメント化、またはアンコメントします。 `start /usr/sbin/snmpd "$src_running"`	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー SNMP デーモンの使用不可化 **AIX** 標準設定 アンコメント	はい
DHCP エージェントの停止	/etc/rc.tcpip から次のエントリーをコメント化します。 `start /usr/sbin/dhcprd "$src_running"`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
DHCP サーバーの停止	/etc/rc.tcpip から次のエントリーをコメント化します。 `start /usr/sbin/dhcpsd "$src_running"`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
autoconf6 の停止	/etc/rc.tcpip から次のエントリーをコメント化します。 `start /usr/sbin/autoconf6 ""`	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
DNS デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 `start /usr/sbin/named "$src_running"`	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

表 25. AIX Security Expert /etc/rc.tcpip 設定 (続き)

アクション・ボタン名	説明	AIX Security Expert が設定する値	元に戻す
gated デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/gated "$src_running"	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 はい	はい
DHCP クライアントの停止	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/dhcpd "$src_running"	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
DPID2 デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/dpid2 "$src_running"	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
NTP デーモンの使用不可化	/etc/rc.tcpip から次のエントリーをコメント化します。 start /usr/sbin/xntpd "$src_running"	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

AIX Security Expert /etc/inetd.conf 設定グループ

AIX Security Expert は、/etc/inetd.conf の中の特定のエントリーをコメント化します。

AIX のデフォルト・インストールでは、システムのセキュリティーを危うくする可能性のある多くのネットワーク・サービスが使用可能になっています。 AIX Security Expert は、不要で、安全でないサービスの各エントリーを /etc/inetd.conf ファイルからコメント化することにより、それらのサービスを使用不可にします。 AIX 標準設定では、これらのエントリーはアンコメントされています。 以下の表は、/etc/inetd.conf でコメント化またはアンコメントされるエントリーをリストしたものです。

表 26. *AIX Security Expert /etc/inetd.conf* 設定

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
/etc/inetd.conf での **sprayd** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `sprayd sunrpc_udp udp wait root ¥ /usr/lib/netsvc/`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での UDP chargen サービスの使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `chargen dgram udp wait root internal`	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
telnet の使用不可化 / telnet の使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `telnet stream tcp6 nowait root ¥ /usr/sbin/telnetd telnetd`	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での UDP Echo サービスの使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `echo dgram udp wait root internal`	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **tftp** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `tftp dgram udp6 SRC nobody ¥ /usr/sbin/tftpd tftpd -n`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

表 26. *AIX Security Expert /etc/inetd.conf* 設定 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
krshd デーモンの使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 kshell stream tcp nowait root ¥ /usr/sbin/krshd krshd	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **rusersd** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 rusersd sunrpc_udp udp wait root ¥ /usr/lib/netsvc/	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **rexecd** の使用不可化 / /etc/inetd.conf での **rexecd** の使用可能化	/etc/inetd.conf から次のエントリーをコメント化します。 exec stream tcp6 nowait root ¥ /usr/sbin/rexecd rexecd	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
POP3D の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 pop3 stream tcp nowait root ¥ /usr/sbin/pop3d pop3d	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **pcnfsd** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 pcnfsd sunrpc_udp udp wait root ¥ /usr/sbin/rpc.pcnfsd pcnfsd	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

表 26. *AIX Security Expert /etc/inetd.conf* 設定 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
/etc/inetd.conf での **bootpd** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 bootps dgram udp wait root ¥ /usr/sbin/bootpd	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **rwalld** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 rwalld sunrpc_udp udp wait root ¥ /usr/lib/netsvc/	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での UDP discard サービスの使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 discard dgram udp wait root ¥ internal	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での TCP daytime サービスの使用不可化 / /etc/inetd.conf での TCP daytime サービスの使用可能化	/etc/inetd.conf から次のエントリーをコメント化またはアンコメントします。 daytime stream tcp nowait root ¥ internal	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での **netstat** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 netstat stream tcp nowait nobody ¥ /usr/bin/netstat	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

表 26. AIX Security Expert /etc/inetd.conf 設定 (続き)

アクション・ボタン名	説明	AIX Security Expert が設定する値	元に戻す
rshd デーモンの使用不可化 / **rshd** デーモンの使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `shell stream tcp6 nowait root ¥ /usr/sbin/rshd rshd rshd`	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー コメント **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での **cmsd** サービスの使用不可化 / /etc/inetd.conf での **cmsd** サービスの使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `cmsd sunrpc_udp udp wait root ¥ /usr/dt/bin/rpc.cms cmsd`	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での **ttdbserver** サービスの使用不可化 / /etc/inetd.conf での **ttdbserver** サービスの使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `ttdbserver sunrpc_tcp tcp wait ¥ root /usr/dt/bin/`	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での **uucpd** の使用不可化 / /etc/inetd.conf での **uucpd** の使用可能化	/etc/inetd.conf から次のエントリーをコメント化またはアンコメントします。 `uucp stream tcp nowait root ¥ /usr/sbin/uucpd uucpd`	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での UDP time サービスの使用不可化 / /etc/inetd.conf での UDP time サービスの使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `time dgram udp wait root internal`	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい

表 26. *AIX Security Expert /etc/inetd.conf* 設定 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
/etc/inetd.conf での TCP time サービスの使用不可化 / /etc/inetd.conf での TCP time サービスの使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `time stream tcp nowait root ¥ internal`	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での **rexd** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `rexd sunrpc_tcp tcp wait root ¥ /usr/sbin/tpc.rexd.rexd rexd`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 はい	はい
/etc/inetd.conf での TCP chargen サービスの使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `chargen stream tcp nowait root ¥ internal`	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **rlogin** の使用不可化 / /etc/inetd.conf での **rlogin** の使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `login stream tcp6 nowait root ¥ /usr/sbin/rlogind rlogind`	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
/etc/inetd.conf での talk の使用不可化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `talk dgram udp wait root ¥ /usr/sbin/talkd talkd`	高レベル・セキュリティー コメント 中レベル・セキュリティー コメント 低レベル・セキュリティー コメント **AIX** 標準設定 アンコメント	はい

表 26. AIX Security Expert /etc/inetd.conf 設定 (続き)

アクション・ボタン名	説明	AIX Security Expert が設定する値	元に戻す
/etc/inetd.conf での **fingerd** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 finger stream tcp nowait nobody ¥ /usr/sbin/fingerd fingerd	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
FTP の使用不可化 / FTP の使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 ftp stream tcp6 nowait root ¥ /usr/sbin/ftpd ftpd	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい
IMAPD の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 imap2 stream tcp nowait root ¥ /usr/sbin/imapd imapd	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **comsat** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 comsat dgram udp wait root ¥ /usr/sbin/comsat comsat	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での **rquotad** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 rquotad sunrpc_udp udp wait root ¥ /usr/sbin/rpc.rquotad	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 はい	はい

表 26. *AIX Security Expert /etc/inetd.conf* 設定 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
/etc/inetd.conf での UDP daytime サービスの使用不可化 / /etc/inetd.conf での UDP daytime サービスの使用可能化	/etc/inetd.conf から次のエントリーをコメント化、またはアンコメントします。 `daytime dgram udp wait root internal`	高レベル・セキュリティー コメント 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 アンコメント	はい

/etc/inetd.conf での **krlogind** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。
	`klogin stream tcp nowait root ¥`
	`/usr/sbin/krlogind krlogind`
		高レベル・セキュリティー
		はい
		中レベル・セキュリティー
		無効
		低レベル・セキュリティー
		無効
		AIX 標準設定
		はい
			はい

/etc/inetd.conf での TCP discard サービスの使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。
	`discard stream tcp nowait root ¥`
	`internal`
		高レベル・セキュリティー
		はい
		中レベル・セキュリティー
		無効
		低レベル・セキュリティー
		無効
		AIX 標準設定
		はい
			はい

/etc/inetd.conf での TCP echo サービスの使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `echo stream tcp nowait root internal`	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

/etc/inetd.conf での **sysstat** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。
	`systat stream tcp nowait nodby ¥`
	`/usr/bin/ps ps -ef`
		高レベル・セキュリティー
		はい
		中レベル・セキュリティー
		はい
		低レベル・セキュリティー
		無効
		AIX 標準設定
		はい
			はい

表 26. *AIX Security Expert* /etc/inetd.conf 設定 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
/etc/inetd.conf での **rstatd** の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `rstatd sunrpc_udp udp wait root ¥ /usr/sbin/rpc.rstatd rstatd`	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/inetd.conf での dtspc の使用不可化	/etc/inetd.conf から次のエントリーをコメント化します。 `dtspc stream tcp nowait root ¥ /usr/dt/bin/dtspcd`	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

AIX Security Expert コマンドの SUID の使用不可化のグループ

以下のコマンドは、デフォルトで SUID ビットを設定された状態でインストールされます。 高、中、および低セキュリティーでは、このビットは設定解除されます。 AIX 標準設定では、これらのコマンドに対して SUID ビットが復元されます。

表 27. *AIX Security Expert* コマンドの *SUID* の使用不可化

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
hls_filepermgr	ファイル許可マネージャー: 高位のオプションを指定した **fpm** コマンドを実行して、特権コマンドから setuid、setgid を除去します。	高レベル・セキュリティー	はい
mls_filepermgr	ファイル許可マネージャー: 中位のオプションを指定した **fpm** コマンドを実行して、特権コマンドから setuid、setgid を除去します。	中レベル・セキュリティー	はい
lls_filepermgr	ファイル許可マネージャー: 低位のオプションを指定した **fpm** コマンドを実行して、特権コマンドから setuid、setgid を除去します。	低レベル・セキュリティー	はい

AIX Security Expert リモート・サービスの使用不可化のグループ

AIX Security Expert は、高レベル・セキュリティーおよび中レベル・セキュリティーに対して非セキュア・コマンドを使用不可にします。

以下のコマンドおよびデーモンは、セキュリティーの抜け穴を見つけるためによく利用されます。 高レベル・セキュリティーと中レベル・セキュリティーでは、これらの非セキュア・コマンドは実行許可を拒否され、デーモンは使用不可にされます。 低レベル・セキュリティーでは、これらのコマンドおよびデーモンには影響はありません。 AIX 標準設定では、これらのコマンドとデーモンは使用可能になります。

- **rcp**

- **rlogin**
- **rsh**
- **tftp**
- **rlogind**
- **rshd**
- **tftpd**

表 *28. AIX Security Expert* リモート・サービスの使用不可化

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
非セキュア・デーモンの使用可能化	TCB が使用可能な場合は、**rlogind**、**rshd**、および **tftpd** デーモンの実行許可が設定され、これらのデーモンのモード・ビット変更によって sysck データベースが更新されます。 TCB が使用可能になっていない場合は、**rlogind**、**rshd**、および **tftpd** デーモンの実行許可が設定されます。	高レベル・セキュリティー 無効 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	はい
非セキュア・コマンドの使用不可化	1. TCB が使用可能な場合は、**rcp**、**rlogin**、**rsh** コマンド、および **tftp** の実行許可が除去され、**sysck** データベースがこれらのコマンドのモード・ビット変更で更新されます。 TCB が使用可能になっていない場合は、**rcp**、**rlogin**、および **rsh** コマンドの実行許可が除去されます。 2. **rcp**、**rlogin**、**rsh**、**tftp**、および **uftp** コマンドの現行インスタンスを停止します (但し、これらのコマンドの 1 つが AIX Security Expert の親プロセスである場合を除きます)。 3. **tcpip:** スタンザを `/etc/security/config` に追加して、**ftp** および **rexec** での **.netrc** の使用を制限します。	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	はい
非セキュア・コマンドの使用可能化	1. TCB が使用可能な場合は、**rcp**、**rlogin**、**rsh**、および **tftp** コマンドの実行許可を設定し、**sysck** データベースをこれらのコマンドのモード・ビット変更で更新します。 TCB が使用可能になっていない場合は、**rcp**、**rlogin**、および **rsh** コマンドに対する実行許可が設定されます。 2. `/etc/security/config` ファイルを除去します。	高レベル・セキュリティー 無効 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

表 28. *AIX Security Expert* リモート・サービスの使用不可化 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
非セキュア・デーモンの使用不可化	1. TCB が使用可能な場合、**rlogind**、**rshd**、および **tftpd** デーモンの実行許可を除去し、**sysck** データベースをこれらのデーモンのモード・ビット変更で更新します。 TCB が使用可能になっていない場合は、**rlogind**、**rshd**、および **tftpd** デーモンの実行許可が除去されます。 2. **rlogind**、**rshd**、および **tftpd** デーモンの現行インスタンスを停止します (ただし、これらのデーモンの 1 つが AIX Security Expert の親プロセスの場合を除きます)。	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	はい
NFS デーモンの停止	• すべての NFS マウントを除去します。 • NFS を使用不可にします。 • `/etc/inittab` から NFS 起動スクリプトを除去します。	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	はい
NFS デーモンの使用可能化	• `/etc/exports` にリストされているすべてのエントリーをエクスポートします。 • エントリーを /etc/inittab に追加して、システム再起動時に `/etc/rc.nfs` を実行します。 • 即時に `/etc/rc.nfs` を実行します。	高レベル・セキュリティー 無効 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

AIX Security Expert 認証を必要としないアクセスの除去グループ

AIX では、ネットワークへのログインにユーザー認証を必要としないサービスはほとんどサポートしていません。

`/etc/hosts.equiv` ファイルおよびすべてのローカル **$HOME**`/.rhosts` ファイルは、パスワードなしでローカル・ホスト上でリモート・コマンドを実行できるホストおよびユーザー・アカウントを定義しています。明示的にこの機能が必要でないかぎり、これらのファイルは消去してください。

表 29. *AIX Security Expert* 認証を必要としないアクセスの除去

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
rhosts および netrc サービスの除去	.rhosts および .netrc ファイルにはユーザー名とパスワードが非暗号化テキスト形式で保管されているため、悪用される可能性があります。	高レベル・セキュリティー .rhosts および .netrc ファイルを、root を含めたすべてのユーザーのホーム・ディレクトリーから除去する。 中レベル・セキュリティー .rhosts および .netrc ファイルを、root を含めたすべてのユーザーのホーム・ディレクトリーから除去する。 低レベル・セキュリティー .rhosts および .netrc ファイルを root のホーム・ディレクトリーから除去する。 **AIX** 標準設定 .rhosts および .netrc ファイルを、root を含めたすべてのユーザーのホーム・ディレクトリーから除去する。	はい
/etc/hosts.equiv ファイルからのエントリーの除去	/etc/hosts.equiv ファイルは、ローカル・ユーザーの **$HOME**/.rhosts ファイルと共に、外部ホスト上のどのユーザーがリモート側からローカル・ホスト上でのコマンドの実行を許可されているかを定義しています。 外部ホスト上のある人物がユーザー名とホスト名の詳細情報を入手した場合、その人物は認証なしにローカル・ホスト上でリモート・コマンドを実行することができます。	高レベル・セキュリティー /etc/hosts.equiv からすべてのエントリーを除去する。 中レベル・セキュリティー /etc/hosts.equiv からすべてのエントリーを除去する。 低レベル・セキュリティー /etc/hosts.equiv からすべてのエントリーを除去する。 **AIX** 標準設定 /etc/hosts.equiv からすべてのエントリーを除去する。	はい

AIX Security Expert ネットワーク・オプションの調整のグループ

ネットワーク・オプションを適切な値に調整することは、セキュリティーの大きな部分を占めます。 ネットワーク属性を 0 に設定するとオプションが使用不可になり、ネットワーク属性を 1 に設定するとオプションが使用可能になります。

以下の表は、高、中、および低レベル・セキュリティーに対するネットワーク属性の設定をリストしたものです。 表では、特定のネットワーク・オプションの提案値がどのようにネットワークのセキュリティーの確保に役立つかについての説明も提供しています。

表 30. *AIX Security Expert* ネットワーク・セキュリティーのためのネットワーク・オプションの調整

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ネットワーク・オプション ipsrcrouteforward	システムが発信元経路指定されたパケットを転送するかどうかを指定します。 ipsrcrouteforward を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 0 低レベル・セキュリティー 無効 **AIX** 標準設定 1	はい
ネットワーク・オプション ipignoreredirects	受信したリダイレクトを処理するかどうかを指定します。	高レベル・セキュリティー 1 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション clean_partial_conns	同期文字 (SYN) アタックを回避するかどうかを指定します。	高レベル・セキュリティー 1 中レベル・セキュリティー 1 低レベル・セキュリティー 1 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション ipsrcrouterecv	システムが発信元経路指定されたパケットを受け入れるかどうかを指定します。 ipsrcrouterecv を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション ipforwarding	カーネルがパケットを転送する必要があるかどうかを指定します。 ipforwarding を使用不可にすると、リダイレクトされたパケットがリモート・ネットワークに到達しないようにすることができます。	高レベル・セキュリティー 0 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい

表 30. *AIX Security Expert* ネットワーク・セキュリティーのためのネットワーク・オプションの調整 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ネットワーク・オプション ipsendredirects	カーネルがリダイレクト・シグナルを送信する必要があるかどうかを指定します。 ipsendredirects を使用不可にすると、リダイレクトされたパケットがリモート・ネットワークに到達しないようにすることができます。	高レベル・セキュリティー 0 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 1	はい
ネットワーク・オプション ip6srcrouteforward	システムが発信元経路指定された IPv6 パケットを転送するかどうかを指定します。 ip6srcrouteforward を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 1	はい
ネットワーク・オプション directed_broadcast	ゲートウェイへのダイレクテッド・ブロードキャストを許可するかどうかを指定します。 directed_broadcast を使用不可にすると、ダイレクトされたパケットがリモート・ネットワークに到達しないようにすることができます。	高レベル・セキュリティー 0 中レベル・セキュリティー 0 低レベル・セキュリティー 0 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション tcp_pmtu_discover	TCP アプリケーション用のパス MTU ディスカバリーを使用可能または使用不可にします。 tcp_pmtu_discover を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 0 低レベル・セキュリティー 0 **AIX** 標準設定 1	はい
ネットワーク・オプション bcastping	ブロードキャスト・アドレスに送信される ICMP エコー・パケットへの応答を許可します。 bcastping を使用不可にすると、smurf アタックを防止することができます。	高レベル・セキュリティー 0 中レベル・セキュリティー 0 低レベル・セキュリティー 0 **AIX** 標準設定 制限なし	はい

表 30. *AIX Security Expert* ネットワーク・セキュリティーのためのネットワーク・オプションの調整 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ネットワーク・オプション icmpaddressmask	システムが ICMP アドレス・マスク要求に応答するかどうかを指定します。 icmpaddressmask を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 0 低レベル・セキュリティー 0 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション udp_pmtu_discover	UDP アプリケーションの Maximum Transfer Unit (MTU) ディスカバリーを使用可能または使用不可にします。 udp_pmtu_discover を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 0 低レベル・セキュリティー 0 **AIX** 標準設定 1	はい
ネットワーク・オプション ipsrcroutesend	アプリケーションが発信元経路指定されたパケットを送信できるかどうかを指定します。 ipsrcroutesend を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 1	はい
ネットワーク・オプション nonlocsrcroute	厳密に発信元経路指定されたパケットをローカル・ネットワークの外のホストに発信できるかどうかをインターネット・プロトコルに指定します。 nonlocsrcroute を使用不可にすると、発信元経路指定アタックによるアクセスを防止できます。	高レベル・セキュリティー 0 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション tcp_tcpsecure	ぜい弱性から TCP 接続を保護する。 値: • 0 = 無保護 • 1 = 確立された接続への偽の SYN の送信 • 2 = 確立された接続への偽の RST の送信 • 3 = 確立された TCP 接続へのデータの投入 • 5-7 = 上記のぜい弱性の組み合わせ	高レベル・セキュリティー 7 中レベル・セキュリティー 7 低レベル・セキュリティー 5 **AIX** 標準設定 制限なし	はい

表 30. *AIX Security Expert* ネットワーク・セキュリティーのためのネットワーク・オプションの調整 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ネットワーク・オプション sockthresh	ネットワーク・メモリーの使用限度を指定します。 いずれの新規ソケット接続も sockthresh チューナブル値を超えることはできません。 ソケットに割り当てできるネットワーク・メモリーの最大量を指定します。	高レベル・セキュリティー 60 中レベル・セキュリティー 70 低レベル・セキュリティー 85 **AIX** 標準設定 制限なし	はい

以下のネットワーク・オプションは、ネットワーク・セキュリティーよりむしろネットワーク・パフォーマンスに関連したものです。

表 31. *AIX Security Expert* ネットワーク・パフォーマンスのためのネットワーク・オプションの調整

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ネットワーク・オプション rfc1323	rfc1323 チューナブルは、TCP ウィンドウのスケーリング・オプションを使用可能にします。	高レベル・セキュリティー 1 中レベル・セキュリティー 1 低レベル・セキュリティー 1 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション tcp_sendspace	tcp_sendspace チューナブルは、送信側アプリケーションが送信呼び出しでブロックされるまでに、どのくらいのデータをカーネルでバッファーに入れられるかを指定します。	高レベル・セキュリティー 262144 中レベル・セキュリティー 262144 低レベル・セキュリティー 262144 **AIX** 標準設定 16384	はい
ネットワーク・オプション tcp_mssdflt	リモート・ネットワークとの通信に使用されるデフォルトの最大セグメント・サイズ。	高レベル・セキュリティー 1448 中レベル・セキュリティー 1448 低レベル・セキュリティー 1448 **AIX** 標準設定 1460	はい

表 31. *AIX Security Expert* ネットワーク・パフォーマンスのためのネットワーク・オプションの調整 (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ネットワーク・オプション extendednetstats	ネットワーク・メモリー・サービスに対するもっと広範な統計を使用可能にします。	高レベル・セキュリティー 1 中レベル・セキュリティー 1 低レベル・セキュリティー 1 **AIX** 標準設定 制限なし	はい
ネットワーク・オプション tcp_recvspace	tcp_recvspace チューナブルは、受信側システムがカーネルで受信側ソケット・キューに何バイトのデータをバッファーに入れられるかを指定します。	高レベル・セキュリティー 262144 中レベル・セキュリティー 262144 低レベル・セキュリティー 262144 **AIX** 標準設定 16384	はい
ネットワーク・オプション sb_max	sb_max チューナブルは、個々のソケットのキューに入れられるソケット・バッファー数の上限を設定します。これは、送信側のソケットまたは受信側のソケットのキューに入れられるバッファーが消費するバッファー・スペース量を制御します。	高レベル・セキュリティー 1048576 中レベル・セキュリティー 1048576 低レベル・セキュリティー 1048576 **AIX** 標準設定 1048576	はい

AIX Security Expert IPsec フィルター・ルールのグループ

AIX Security Expert では、以下の IPsec フィルターを提供しています。

表 32. *AIX Security Expert IPsec* フィルター・ルール

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
5 分間のホストの回避	既知のぜい弱性を持つ、ホスト上のいくつかの tcp および udp ポートに向けられたパケットを 5 分間回避またはブロックします。 ホストは、これらのポートを宛先としたパケットを 5 分間受信しません。	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	はい

表 32. *AIX Security Expert IPsec* フィルター・ルール (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
ポート・スキャンからのホストの保護	ポート・スキャンから保護します。 ポート・スキャンを行うすべてのリモート・ホストが 5 分間回避またはブロックされます。 このリモート・ホストからのすべてのパケットは、5 分間受信されません。	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	はい

AIX Security Expert 各種グループ

AIX Security Expert では、高、中、および低セキュリティーに対する各種セキュリティー設定を提供しています。

表 33. *AIX Security Expert* 各種グループ

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
パス root からのドットの除去	**$HOME**/.profile、**$HOME**/.kshrc、**$HOME**/.cshrc、および **$HOME**/.login ファイルで、PATH 環境変数にドット (.) がないかどうかをチェックし、ある場合は除去します。 **注:** ドットの除去は、ファイルのエントリーが PATH 環境変数で始まり、かつドットを含む場合に限られます。 PATH 環境変数が他の変数を含んでいるか、スクリプトから呼び出されるプログラムから返された値に設定される場合、ファイルは変更されません。変更されないパスの例を次に示します。ここで、*pathprog* はパス・ストリングを返すプログラムです。 `PATH="$(pathprog)"` このパスでは、変数 *pathprog* のコンテンツが解決される前に、ドットがパスから除去されるので、返されたパスに存在するドットは除去されません。	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 はい	はい
システム・アクセスの制限	**cron** ジョブの実行を許可されているユーザーが root のみであることを確認します。	高レベル・セキュリティー root を cron.allow ファイルで唯一のユーザーにして、cron.deny ファイルを除去する。 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 cron.allow ファイルを除去して、cron.deny ファイルのすべてのエントリーを削除する。	はい

表 33. *AIX Security Expert* 各種グループ (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
/etc/environment からのドットの除去	/etc/environment ファイルの **PATH** 環境変数からドット (.) を除去します。	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 はい	はい
非 root パスからのドットの除去	すべての非 root ユーザーの **$HOME**/.profile、**$HOME**/.kshrc、**$HOME**/.cshrc、および **$HOME**/.login ファイルにある **PATH** 環境変数からドット (.) を除去します。 **注:** ドットの除去は、ファイルのエントリーが PATH 環境変数で始まり、かつドットを含む場合に限られます。PATH 環境変数が他の変数を含んでいるか、スクリプトから呼び出されるプログラムから返された値に設定される場合、ファイルは変更されません。変更されないパスの例を次に示します。ここで、*pathprog* はパス・ストリングを返すプログラムです。 PATH="$(*pathprog*)" このパスでは、変数 *pathprog* のコンテンツが解決される前に、ドットがパスから除去されるので、返されたパスに存在するドットは除去されません。	高レベル・セキュリティー はい 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	いいえ
/etc/ftpusers ファイルへの root ユーザーの追加	root ユーザー名を /etc/ftpusers ファイルに追加して、リモート root **ftp** を使用不可にします。	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい
/etc/ftpusers ファイルからの root ユーザーの除去	/etc/ftpusers から root エントリーを除去して、リモート root **ftp** を使用可能にします。	高レベル・セキュリティー 無効 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 はい	はい

表 33. *AIX Security Expert* 各種グループ (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す

ログイン herald の設定	/etc/security/login.cfg を調べて、herald 値が指定されていないことを確認します。 デフォルト herald が使用されている場合は、その herald を変更する必要があります。 herald は、システムのロケールが **en_US** か別の英語のロケールの場合にのみ変更することができます。この基準を満たしている場合は、/etc/security/login.cfg ファイルのデフォルト・スタンザにある herald 属性の値は、次のように設定されます。
	`Unauthorized use of this ¥`
	`system is prohibited.¥nlogin:`
	注: セキュリティー設定は、新規セッションでのみ有効になります。 セキュリティー設定は、構成が設定されていたセッションでは有効になりません。
		高レベル・セキュリティー
		herald="Unauthorized use of this system is prohibited.¥nlogin:"
		中レベル・セキュリティー
		herald="Unauthorized use of this system is prohibited.¥nlogin:"
		低レベル・セキュリティー
		herald="Unauthorized use of this system is prohibited.¥nlogin:"
		AIX 標準設定
		herald=
			はい

ゲスト・アカウントの除去	高、中、および低セキュリティーでは、ゲスト・アカウントを除去するほか、マシン上のゲストのデータも除去します。 AIX 標準設定では、システム上にゲスト・アカウントが作成されます。 **注:** AIX Security Expert はそのようなユーザーの対話式タスクを処理するよう設計されていないため、システム管理者はこのアカウントに対して明示的にパスワードを設定する必要があります。	高レベル・セキュリティー ゲスト・アカウントおよびデータの除去 中レベル・セキュリティー ゲスト・アカウントおよびデータの除去 低レベル・セキュリティー ゲスト・アカウントおよびデータの除去 **AIX** 標準設定 マシン上にゲスト・アカウントを追加	はい
Crontab アクセス権	root の **crontab** ジョブが root によって所有されており、書き込み可能になっていることを確認します。	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー はい **AIX** 標準設定 無効	はい
X サーバー・アクセスの使用可能化	X サーバーへのアクセスの認証を義務付けます。	高レベル・セキュリティー 認証が必要 中レベル・セキュリティー 認証が必要 低レベル・セキュリティー 無効 **AIX** 標準設定 不要	いいえ

表 33. *AIX Security Expert* 各種グループ (続き)

アクション・ボタン名	説明	**AIX Security Expert** が設定する値	元に戻す
オブジェクトの作成許可	/etc/security/user の **umask** 属性に適切な値を設定します。これによりデフォルトのオブジェクト作成許可が指定されます。	高レベル・セキュリティー 077 中レベル・セキュリティー 027 低レベル・セキュリティー 無効 **AIX** 標準設定 022	はい
コア・ファイル・サイズの設定	/etc/security/limits の **core** 属性に適切な値を設定します。これにより、root のコア・ファイル・サイズが指定されます。 **注:** セキュリティー設定は、新規セッションでのみ有効になります。 セキュリティー設定は、構成が設定されていたセッションでは有効になりません。	高レベル・セキュリティー 0 中レベル・セキュリティー 0 低レベル・セキュリティー 0 **AIX** 標準設定 2097151	はい
SED フィーチャーの使用可能化	「スタック実行使用不可 (SED)」フィーチャーを使用可能にして、指定されたファイルについて **sedmgr** コマンドを実行します。 **注:** ルールを有効にするには、システム・リブートが必要です。	高レベル・セキュリティー setidfiles 中レベル・セキュリティー 無効 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	
ルート・パスワード保全性検査	ルート・パスワードがぜい弱でないことを確認します。ルートの dictionlist 属性は /etc/security/aixpert/dictionary/English に設定されます。これにより、設定されているルート・パスワードがぜい弱でないことを **passwd** コマンドで確認できます。	高レベル・セキュリティー はい 中レベル・セキュリティー はい 低レベル・セキュリティー 無効 **AIX** 標準設定 無効	はい

AIX Security Expert セキュリティーを元に戻す

いくつかの AIX Security Expert セキュリティー設定およびルールは、元に戻すことができます。

以下に示す AIX Security Expert セキュリティー設定およびルールは、元に戻すことはできません。

- 高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティーのパスワード定義の確認。
- 高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティーのユーザー定義の確認。

- 高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティーのグループ定義の確認。
- 高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティーの TCB 更新。
- 高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティーの X サーバー・アクセスの使用可能化。
- 高レベル・セキュリティーと AIX 標準設定の root 以外のパスからのドットの除去。
- 高レベル・セキュリティー、中レベル・セキュリティー、および低レベル・セキュリティーのゲスト・アカウントの除去。

AIX Security Expert セキュリティーの確認

AIX Security Expert は、現行システムおよびネットワーク・セキュリティー設定のレポートを生成することができます。

AIX Security Expert (aixpert コマンド) を使用してシステムを構成した後は、「セキュリティーの確認」オプションを使用して各種構成設定のレポートを作成することができます。 これらの設定のいずれかが AIX Security Expert の制御の外で変更されると、AIX Security Expert のセキュリティーの確認オプションがその変更を /etc/security/aixpert/check_report.txt ファイルのログに記録します。

例えば、低レベル・セキュリティーを適用すると、**talkd** デーモンが /etc/inetd.conf で使用不可になります。 後で **talkd** デーモンが使用可能にされ、セキュリティーの確認が実行されると、この情報は次のように check_report.txt ファイルに記録されます。

```
coninetdconf.ksh: Service talk using protocol udp should be disabled, however it is enabled now.
```

適用されたセキュリティー設定が変更されていなければ、check_report.txt ファイルは空になります。

「セキュリティーの確認」オプションは定期的に実行し、結果レポートを検討して、AIX Security Expert セキュリティー設定の適用以降に設定が変更されているか確認します。 「セキュリティーの確認」オプションは、ソフトウェアのインストールや更新など、大きなシステム変更の一部としても実行します。

関連情報:

aixpert コマンド

AIX Security Expert ファイル

AIX Security Expert は、いくつかのファイルを作成して使用します。

/etc/security/aixpert/core/aixpertall.xml
すべての可能なセキュリティー設定の XML リストが含まれます。

/etc/security/aixpert/core/appliedaixpert.xml
適用されたセキュリティー設定の XML リストが含まれます。

/etc/security/aixpert/core/secaixpert.xml
AIX Security Expert GUI で処理された場合、選択されたセキュリティー設定の XML リストが含まれます。

/etc/security/aixpert/log/aixpert.log
適用されたセキュリティー設定のトレース・ログが含まれます。 AIX Security Expert は syslog を使用しません。AIX Security Expert は、/etc/security/aixpert/log/aixpert.log に直接書き込みを行います。

注: AIX Security Expert XML およびログ・ファイルは、以下の許可によって作成されます。

/etc/security/aixpert/

```
drwx------
```

/etc/security/aixpert/core/

```
drwx------
```

/etc/security/aixpert/core/aixpertall.xml

```
r--------
```

/etc/security/aixpert/core/appliedaixpert.xml

/etc/security/aixpert/core/secaixpert.xml

/etc/security/aixpert/log

```
drwx------
```

/etc/security/aixpert/log/aixpert.log

```
-rw-------
```

/etc/security/aixpert/core/secundoaixpert.xml

```
rw-------
```

/etc/security/aixpert/check_report.txt

```
rw-------
```

AIX Security Expert 高レベル・セキュリティーのシナリオ

これは、AIX Security Expert の高レベル・セキュリティー用のシナリオです。

セキュリティー・レベルの AIX Security Expert ビューの一部は、National Institute of Standards and Technology の資料の「*Security Configuration Checklists Progarm for IT Pruducts - Guidance for CheckLists Users and Developers*」(NIS Web サイト: http://www.nist.gov/index.html で資料名を検索) からの引用です。 ただし、高、中、低レベルのセキュリティーは、人によって異なる意味を持ちます。 使用するシステムの稼働環境を理解することが大切です。 選択したセキュリティー・レベルが高過ぎると、自分のコンピューターから自分自身をロックアウトしてしまう可能性があります。選択したセキュリティー・レベルが低すぎると、コンピューターがサイバー・アタックに対してぜい弱になる可能性があります。

ここに示す例は、高レベル・セキュリティーを必要とする環境です。 Bob は彼のシステムをインターネット・サービス・プロバイダーと同じ場所に配置します。 このシステムは直接インターネットに接続されて、HTTP サーバーとして稼働し、重要なユーザー・データを含み、その管理は Bob がリモート側で行う必要があります。 このシステムは、ISP を使用してオンラインに接続される前に、孤立したローカル・ネットワークでセットアップとテストを行う必要があります。

この環境にとって的確なセキュリティー・レベルは高レベル・セキュリティーですが、Bob はシステムへのリモート・アクセスを必要とします。 高レベル・セキュリティーでは、**telnet**、**rlogin**、**ftp** のほか、ネットワーク上でパスワードが暗号化されずに伝送されるその他の共通接続が許可されません。 これらのパスワードは、インターネット上の何者かに容易にスヌープされてしまう可能性があります。 Bob には、リモート側で安全にログインする方法 (例えば、**openssh** など) が必要です。 Bob は、AIX Security Expert の詳細資料を読み、高レベル・セキュリティーで禁止される可能性のある事柄の中に、彼の環境に固有なものがあるかどうかを確認することができます。 そのようなものがある場合は、高レベル・セキュリティーの詳細情報のパネルが表示されたときに、それを選択解除することができます。 また Bob は、HTTP サーバー、または彼のシステムで提供することを予定しているその他のサービスを構成して始動する必要があります。

こうすると、Bob が高レベル・セキュリティーを選択したときに、AIX Security Expert は実行中のサービスが必要なことを認識して、それらのポートへのアクセスをブロックしなくなります。 他のすべてのポートへのアクセスはぜい弱性につながる可能性があるため、高レベル・セキュリティーではこれらのポートはブロックされます。 この構成のテスト後は、Bob のマシンはインターネット上ですぐに使用することができます。

AIX Security Expert 中レベル・セキュリティーのシナリオ

ここに示すのは、AIX Security Expert の中レベル・セキュリティー用のシナリオです。

Alice は、会社のファイアウォールの背後に配置された、社内ネットワークに接続されるシステムのセキュリティー強化を必要としています。このネットワークは保護されており、きちんと管理されています。 このシステムは、**telnet** および **ftp** システムのアクセスを必要とする多くのユーザーによって使用されます。 Alice は共通のセキュリティー設定 (ポート・スキャン保護やパスワード有効期限など) を配備したいと考えているのですが、このシステムはまたほとんどのリモート・アクセス方式を受け入れるものでなければなりません。 このシナリオの場合、Alice のシステムに最も適したセキュリティー設定は、中レベル・セキュリティーです。

AIX Security Expert 低レベル・セキュリティーのシナリオ

ここに示すのは、AIX Security Expert の低レベル・セキュリティー用のシナリオです。

Bruce はある期間システムの管理を担当しています。 システムは孤立した機密保護機能のあるローカル・ネットワークに常駐しています。 このシステムは、多種多様な人およびサービスに使用されています。 彼はこのシステムのセキュリティー・レベルを最小限から上げたいと考えていますが、このシステムへのどのような形のアクセスも中断することはできません。 Bruce のマシンに適したセキュリティー・レベルは、低レベル・セキュリティーです。

セキュリティー・チェックリスト

以下は、新たにインストールしたシステムまたは既存のシステムに対して実行されるセキュリティー・アクションのチェックリストです。

このリストは、完全なセキュリティー・チェックリストではありませんが、ご使用の環境のセキュリティー・チェックリストを構築するための基礎として使用できます。

- 新規システムをインストールするときは、セキュア基本メディアから AIX をインストールします。インストール時に、以下の手順を実行します。
  - サーバー上に CDE、GNOME、または KDE などのデスクトップ・ソフトウェアをインストールしないでください。
  - 必要なセキュリティーのフィックス、および推奨されている保守および技術レベルのフィックスをインストールします。 最新のサービス会報、セキュリティー勧告、およびフィックス情報については、IBM System p eServer™ Support Fixes Web サイト (http://www.ibm.com/support/fixcentral) を参照してください。
  - 初期インストールの後にシステムをバックアップし、安全な場所にシステム・バックアップを保管します。
- 制限されたファイルおよびディレクトリーのためのアクセス制御リストを確立します。
- 不必要なユーザー・アカウントおよびシステム・アカウント (daemon、bin、sys、adm、lp、uucp など) を使用不可にします。 アカウントの削除は、ユーザー ID およびユーザー名など、システム・バックアップ上のデータと関連がある可能性があるアカウント情報を削除してしまうため、推奨されていま

せん。ユーザーが以前に削除されたユーザー ID を使って作成され、システム・バックアップがシステム上に復元される場合、新規ユーザーは、復元されたシステムに予期しないアクセスを行うことがあります。

- **/etc/inetd.conf**、**/etc/inittab**、**/etc/rc.nfs**、および **/etc/rc.tcpip** ファイルを定期的に検討し、不要なデーモンおよびサービスをすべて除去します。
- 以下のファイルの許可条件が正しく設定されていることを確認します。

```
-rw-rw-r-- root     system  /etc/filesystems
-rw-rw-r-- root     system  /etc/hosts
-rw------- root     system  /etc/inittab
-rw-r--r-- root     system  /etc/vfs
-rw-r--r-- root     system  /etc/security/failedlogin
-rw-rw---- root     audit   /etc/security/audit/hosts
```

- root アカウントはリモートからはログインできないようにします。 root アカウントは、システム・コンソールからのみログインできるようにする必要があります。
- システム監査を使用可能にします。 詳しくは、 149 ページの『監査の概要』を参照してください。
- ログイン制御ポリシーを使用可能にします。 詳しくは、 39 ページの『ログイン制御』を参照してください。
- **xhost** コマンドを実行するためのユーザー・アクセス権を使用不可にします。詳しくは、 45 ページの『X11 および CDE 関連事項の管理』を参照してください。
- **PATH** 環境変数への無許可変更を防止します。 詳しくは、 62 ページの『PATH 環境変数』を参照してください。
- telnet、rlogin、および rsh を使用不可にします。 詳しくは、 229 ページの『TCP/IP セキュリティー』を参照してください。
- ユーザー・アカウント制御を確立します。 詳しくは、 59 ページの『ユーザー・アカウント制御』を参照してください。
- 厳密なパスワード・ポリシーを施行します。 詳しくは、 72 ページの『パスワード』を参照してください。
- ユーザー・アカウントのディスク・クォータを確立します。 詳しくは、 85 ページの『オーバー・クォータ条件からのリカバリー』を参照してください。
- 管理アカウントが **su** コマンドを使用できるようにします。 **/var/adm/sulog** ファイル内の **su** コマンドのログをモニターします。
- X Window システムを使用するとき、画面ロックを使用可能にします。
- **cron** および **at** コマンドへのアクセスを、アクセスが必要なアカウントのみに制限します。
- **ls** コマンドの別名を使用して、隠しファイルおよびファイル名の隠し文字を表示します。
- **rm** コマンドの別名を使用して、ファイルをシステムから不用意に削除することを防ぎます。
- 不必要なネットワーク・サービスを使用不可にします。 詳しくは、 237 ページの『ネットワーク・サービス』を参照してください。
- 頻繁にシステム・バックアップを実行し、バックアップの保全性を確認します。
- セキュリティー関連の電子メール配布先リストに加入します。

一般的 AIX システム・サービス

以下の表では、AIX 内の一般的なシステム・サービスをリストします。 この表は、ご使用のシステムを保護するための開始点を認識するために使用します。

システムの保護を開始する前に、オリジナルの構成ファイルをすべてバックアップしてください。特に、次のファイルが重要です。

- **/etc/inetd.conf**
- **/etc/inittab**
- **/etc/rc.nfs**
- **/etc/rc.tcpip**

サービス	デーモン	開始点	機能	コメント
inetd/bootps	inetd	/etc/inetd.conf	ディスクレス・クライアントへの bootp サービス	• ネットワーク・インストール管理 (NIM) およびシステムのリモート・ブートのために必要 • tftp と並行して動作する • ほとんどの場合、使用不可にする
inetd/chargen	inetd	/etc/inetd.conf	文字生成プログラム (テストのみ)	• TCP および UDP サービスとして使用可能 • サービス妨害攻撃の機会を与える • ネットワークをテストしているのでない限り、使用不可にする
inetd/cmsd	inetd	/etc/inetd.conf	カレンダー・サービス (CDE で使用される)	• root として実行するので、セキュリティーが問題 • CDE でこのサービスが必要でない限り、使用不可にする • バック・ルーム・データベース・サーバー上では使用不可にする
inetd/comsat	inetd	/etc/inetd.conf	電子メールの着信を通知する	• root として実行するので、セキュリティーが問題 • めったに必要とされない • 使用不可にする
inetd/daytime	inetd	/etc/inetd.conf	旧式のタイム・サービス (テストのみ)	• root として実行する • TCP および UDP サービスとして使用可能 • サービス妨害 PING 攻撃の機会を与える • サービスは旧式であり、テストのみに使用される • 使用不可にする
inetd/discard	inetd	/etc/inetd.conf	/dev/null サービス (テストのみ)	• TCP および UDP サービスとして使用可能 • サービス妨害攻撃で使用される • サービスは旧式であり、テストのみに使用される • 使用不可にする

サービス	デーモン	開始点	機能	コメント
inetd/dtspc	inetd	/etc/inetd.conf	CDE サブプロセス制御	• このサービスは、デーモンのホスト上でのプロセス開始を要求しているCDE クライアントへの応答として、 **inetd** デーモンによって自動的に開始される。 これにより、サービスが攻撃されやすくなる。 • CDE を使用していないバック・ルーム・サーバー上では使用不可にする • CDE は、このサービスがないと機能できない • 絶対に必要というのでない限り、使用不可にする
inetd/echo	inetd	etc/inetd.conf	エコー・サービス (テストのみ)	• UDP および TCP サービスとして使用可能 • サービス妨害または Smurf 攻撃で使用される可能性がある • 誰か他の人にエコーして、ファイアウォールをすり抜けたり、またはデータストームを開始するために使用される • 使用不可にする
inetd/exec	inetd	/etc/inetd.conf	リモート実行サービス	• root ユーザーとして実行する • ユーザー ID およびパスワードを入力する必要がある。それらは無保護のまま渡される。 • このサービスは、スヌープされる可能性が高い • 使用不可にする
inetd/finger	inetd	/etc/inetd.conf	ユーザーの finger 検査	• root ユーザーとして実行する • システムおよびユーザーに関する情報を提供する • 使用不可にする
inetd/ftp	inetd	/etc/inetd.conf	ファイル転送プロトコル	• root ユーザーとして実行する • ユーザー ID およびパスワードは無保護のまま転送されるので、それらをスヌープすることができる • このサービスを使用不可にして、パブリック・ドメイン・セキュア・シェルの組を使用する
inetd/imap2	inetd	/etc/inetd.conf	インターネット・メール・アクセス・プロトコル	• このサーバーの最新バージョンを使用しているか確認する • メール・サーバーを実行している場合にのみ必要。 その他の場合には、使用不可にする。 • ユーザー ID およびパスワードは無保護のまま渡される

サービス	デーモン	開始点	機能	コメント
inetd/klogin	inetd	/etc/inetd.conf	Kerberos ログイン	• 自分のサイトが Kerberos 認証を使用する場合には、使用可能にする
inetd/kshell	inetd	/etc/inetd.conf	Kerberos シェル	• 自分のサイトが Kerberos 認証を使用する場合には、使用可能にする
inetd/login	inetd	/etc/inetd.conf	rlogin サービス	• IP スプーフィング、DNS スプーフィングされやすい • ユーザー ID およびパスワードを含むデータは無保護のまま渡される • root ユーザーとして実行する • このサービスの代わりに、セキュア・シェルを使用する
inetd/netstat	inetd	/etc/inetd.conf	現行のネットワーク状況の報告	• システム上で実行している場合、ネットワーク情報がハッカーに提供される可能性がある • 使用不可にする
inetd/ntalk	inetd	/etc/inetd.conf	ユーザーは相互に通話することができる	• root ユーザーとして実行する • 実動サーバー上またはバック・ルーム・サーバー上では必須でない • 絶対に必要というのでない限り、使用不可にする
inetd/pcnfsd	inetd	/etc/inetd.conf	PC NFS ファイル・サービス	• 現在使用中でない場合には、サービスを使用不可にする • このサービスと類似のサービスが必要な場合には、 Microsoft の SMB 仕様のリリースより前の pcnfsd デーモンとして Samba を考慮する
inetd/pop3	inetd	/etc/linetd.conf	Post Office Protocol	• ユーザー ID およびパスワードは無保護のまま送信される • ご使用のシステムがメール・サーバーであり、 POP3 のみをサポートするアプリケーションを使用しているクライアントを持っている場合にのみ必要 • クライアントが IMAP を使用する場合には、それで代用するか、または POP3 サービスを使用する。このサービスは、Secure Socket Layer (SSL) トンネルを持っている。 • メール・サーバーを実行していたり、 POP サービスを必要とするクライアントを持っていたりする場合を除き、使用不可にする
inetd/rexd	inetd	/etc/inetd.conf	リモート実行	• root ユーザーとして実行する • **on** コマンドと同等 • サービスを使用不可にする • その代わりに、**rsh** および **rshd** を使用する

サービス	デーモン	開始点	機能	コメント
inetd/quotad	inetd	/etc/inetd.conf	ファイル・クォータのレポート (NFS クライアント用)	• NFS ファイル・サービスを実行している場合にのみ必要 • このサービスは、**quota** コマンドに応答する必要があるのでない限り、使用不可にする • このサービスを使用する必要がある場合には、このサービス用のすべてのパッチおよびフィックスを最新のものに保つ
inetd/rstatd	inetd	/etc/inetd.conf	カーネル統計サーバー	• システムをモニターする必要がある場合には、SNMP を使用して、このサービスを使用不可にする • **rup** コマンドを使用するためには必要
inetd/rusersd	inetd	/etc/inetd.conf	ログインされたユーザーについての情報	• これは必須サービスではない。 使用不可にする • root ユーザーとして実行する • システム上の現行ユーザーのリストを提供する。rusers と同等。
inetd/rwalld	inetd	/etc/inetd.conf	すべてのユーザーへの書き込み	• root ユーザーとして実行する • ご使用のシステムが対話式ユーザーを持っている場合には、このサービスを保持する必要がある • ご使用のシステムが実動サーバーまたはデータベース・サーバーである場合には、必要ない • 使用不可にする
inetd/shell	inetd	/etc/inetd.conf	rsh サービス	• 可能であれば、このサービスを使用不可にする。 その代わりにセキュア・シェルを使用する • このサービスを使用しなければならない場合には、 TCP ラッパーを使用して、スプーフィングを停止し、公開を制限する • **Xhier** ソフトウェア配布プログラムには必須
inetd/sprayd	inetd	/etc/inetd.conf	RPC スプレー・テスト	• root ユーザーとして実行する • NFS ネットワーク問題の診断に必要な場合がある • NFS を実行しない場合には、使用不可にする
inetd/systat	inetd	/etc/inted.conf	"ps -ef" 状況レポート	• リモート・サイトで、ご使用のシステム上の処理状況を表示できるようにする • デフォルトでは、このサービスは使用不可。 サービスが使用可能になっていないことを周期的に確認しなければならない。

サービス	デーモン	開始点	機能	コメント
inetd/talk	inetd	/etc/inetd.conf	ネット上の 2 人のユーザー間で分割画面を確立する	• 必須サービスではない • **talk** コマンドで使用される • ポート 517 で UDP サービスを提供する • UNIX ユーザー用の複数の対話式チャット・セッションが必要であるのでない限り、使用不可にする
inetd/ntalk	inetd	/etc/inetd.conf	"new talk" は、ネット上の 2 人のユーザー間で分割画面を確立する	• 必須サービスではない • **talk** コマンドで使用される • ポート 517 で UDP サービスを提供する • UNIX ユーザー用の複数の対話式チャット・セッションが必要であるのでない限り、使用不可にする
inetd/telnet	inetd	/etc/inetd.conf	Telnet サービス	• リモート・ログイン・セッションをサポートしているが、パスワードおよび ID は無保護のまま渡される • 可能であれば、このサービスを使用不可にし、その代わりにリモート・アクセス用のセキュア・シェルを使用する
inetd/tftp	inetd	/etc/inetd.conf	小規模ファイル転送	• ポート 69 で UDP サービスを提供する • root ユーザーとして実行するので、暗号漏えいの可能性がある • NIM によって使用される • NIM を使用しているか、またはディスクレス・ワークステーションをブートしなければならないのでない限り、使用不可にする
inetd/time	inetd	/etc/inetd.conf	旧式のタイム・サービス	• **rdate** コマンドで使用される **inetd** の内部機能 • TCP および UDP サービスとして使用可能 • ブート時にクロックを同期化するために使用されることがある • このサービスは旧式である。 その代わりに、ntpdate を使用する • このサービスを使用可能にし、ご使用のシステムをテスト (ブート/リブート) して、問題がなくなったら、このサービスを使用不可にする

サービス	デーモン	開始点	機能	コメント
inetd/ttdbserver	inetd	/etc/inetd.conf	tool-talk データベース・サーバー (CDE 用)	• **rpc.ttdbserverd** は root ユーザーとして実行するので、暗号漏えいの可能性がある • CDE の必須サービスとして述べられているが、CDE はそれがなくても作動できる • セキュリティーが重要なバック・ルーム・サーバーまたはシステム上で実行してはならない
inetd/uucp	inetd	/etc/inetd.conf	UUCP ネットワーク	• UUCP を使用するアプリケーションを持っているのでない限り、使用不可にする
inittab/dt	init	/etc/rc.dt script in the /etc/inittab	CDE 環境へのデスクトップ・ログイン	• コンソール上で X11 サーバーを開始する • 他の X11 ステーションが同じマシンにログインできるようにするために、 X11 Display Manager Control Protocol (xdcmp) をサポートする • サービスは、個人のワークステーション上でのみ使用する必要がある。バック・ルーム・システムのためにこのサービスを使用することは避ける。
inittab/dt_nogb	init	/etc/inittab	CDE 環境へのデスクトップ・ログイン (グラフィック・ブートでない)	• システムが完全に立ち上がるまでグラフィック表示されない • inittab/dt と同じ問題
inittab/httpdlite	init	/etc/inittab	**docsearch** コマンド用の Web サーバー	• docsearch エンジン用のデフォルトの Web サーバー • ご使用のマシンが文書サーバーであるのでない限り、使用不可にする
inittab/i4ls	init	/etc/inittab	ライセンス・マネージャー・サーバー	• 開発マシンの場合には、使用可能にする • 実動マシンの場合には、使用不可にする • ライセンス要件を持つバック・ルーム・データベース・マシンの場合には、使用可能にする • コンパイラー、データベース・ソフトウェア、またはその他のいずれかのライセンス交付を受けたプロダクト用のサポートを提供する
inittab/imqss	init	/etc/inittab	"docsearch" 用の検索エンジン	• docsearch エンジン用のデフォルトの Web サーバーのパーツ • ご使用のマシンが文書サーバーであるのでない限り、使用不可にする

サービス	デーモン	開始点	機能	コメント
inittab/lpd	init	/etc/inittab	BSD ライン・プリンター・インターフェース	• 他のシステムからの印刷ジョブを受け入れる • このサービスを使用不可にしても、印刷サーバーにジョブを送信できる • 印刷に影響を与えないことを確認した後に、このサービスを使用不可にする
inittab/nfs	init	/etc/inittab	ネットワーク・ファイルシステム (NFS)/ネット情報サービス (NIS)	• NFS および NIS サービスは、UDP 上または RPC 上のどちらに構築されたかに基づいている • 認証は最小である • バック・ルーム・マシンの場合には、このサービスを使用不可にする
inittab/piobe	init	/etc/inittab	プリンター入出力バックエンド (印刷用)	• **qdaemon** デーモンで実行依頼されるジョブのスケジューリング、スプーリング、および印刷を処理する • サーバーに印刷ジョブを送信しているためにご使用のシステムから印刷していない場合には、使用不可にする
inittab/qdaemon	init	/etc/inittab	キュー・デーモン (印刷用)	• **piobe** デーモンに印刷ジョブを実行依頼する • ご使用のシステムから印刷していない場合には、使用不可にする
inittab/uprintfd	init	/etc/inittab	カーネル・メッセージ	• 通常は必要ない • 使用不可にする
inittab/writesrv	init	/etc/inittab	tty への注釈の書き込み	• 対話式 UNIX ワークステーション・ユーザーによってのみ使用される • サーバー、バック・ルーム・データベース、および開発マシンの場合には、このサービスを使用不可にする • ワークステーションの場合には、このサービスを使用可能にする
inittab/xdm	init	/etc/inittab	従来型の X11 Display Management	• バック・ルーム実動サーバーまたはデータベース・サーバー上で実行しない • X11 Display Management が必要とされるのでない限り、開発システム上で実行しない • グラフィックスが必要である場合、ワークステーション上での実行を受け入れ可能にする

サービス	デーモン	開始点	機能	コメント
rc.nfs/automountd		/etc/rc.nfs	自動ファイルシステム	• NFS を使用する場合、ワークステーションでは、このサービスを使用可能にする • 開発サーバーまたはバック・ルーム・サーバーの場合には、自動マウント機能を使用しない
rc.nfs/biod		/etc/rc.nfs	ブロック入出力デーモン (NFS サーバーには必須)	• NFS サーバーのみに使用可能にする • NFS サーバーでない場合には、nfsd および rpc.mountd とともにこのサービスを使用不可にする
rc.nfs/keyserv		/etc/rc.nfs	セキュア RPC 鍵サーバー	• セキュア RPC に必要な鍵を管理する • NFS と NIS を使用していない 場合、このサービスを使用不可にする
rc.nfs/nfsd		/etc/rc.nfs	NFS サービス (NFS サーバーには必須)	• 認証が貧弱 • このサービス自体がスタック・フレームの破壊に加担する可能性がある • NFS ファイル・サーバー上では、使用可能にする • このサービスを使用不可にする場合には、**biod**、**nfsd**、および **rpc.mountd** も使用不可にする
rc.nfs/rpc.lockd		/etc/rc.nfs	NFS ファイル・ロック	• NFS を使用しない場合には、使用不可にする • ネットワークにまたがってファイル・ロックを使用していない場合には、このサービスを使用不可にする • **lockd** デーモンについては、「SANS Top Ten Security Threats」で言及されている
rc.nfs/rpc.mountd		/etc/rc.nfs	NFS ファイル・マウント (NFS サーバーには必須)	• 認証が貧弱 • このサービス自体がスタック・フレームの破壊に加担する可能性がある • NFS ファイル・サーバー上でのみ使用可能にする • このサービスを使用不可にする場合には、**biod** および **nfsd** も使用不可にする
rc.nfs/rpc.statd		/etc/rc.nfs	NFS ファイル・ロック (それらをリカバリーするため)	• NFS にまたがってファイル・ロックを実装する • NFS を使用しているのでない限り、使用不可にする
rc.nfs/rpc.yppasswdd		/etc/rc.nfs	NIS パスワード・デーモン (NIS マスター用)	• ローカル・パスワード・ファイルを取り扱うために使用される • 問題のマシンが NIS マスターであるときにのみ必須。その他のすべての場合には使用不可にする。

サービス	デーモン	開始点	機能	コメント
rc.nfs/ypupdated		/etc/rc.nfs	NIS 更新デーモン (NIS スレーブ用)	• NIS マスターからプッシュされる NIS データベース・マップを受信する • 問題のマシンがマスター NIS サーバーへの NIS スレーブであるときにのみ必須
rc.tcpip/autoconf6		/etc/rc.tcpip	IPv6 インターフェース	• IP バージョン 6 を実行しているのでない限り、使用不可にする
rc.tcpip/dhcpcd		/etc/rc.tcpip	動的ホスト構成プロトコル (クライアント)	• バック・ルーム・サーバーは DHCP に依存すべきでない。 このサービスを使用不可にする。 • ホストが DHCP を使用していない場合には、使用不可にする
rc.tcpip/dhcprd		/etc/rc.tcpip	動的ホスト構成プロトコル (リレー)	• DHCP ブロードキャストをグラブし、それらを別のネットワーク上のサーバーに送信する • ルーター上で見つかったサービスを複製する • DHCP を使用していないか、またはネットワーク間での情報の引き渡しに依存していない場合には、このサービスを使用不可にする
rc.tcpip/dhcpsd		/etc/rc.tcpip	動的ホスト構成プロトコル (サーバー)	• ブート時にクライアントからの DHCP 要求に応答する。 IP 名、番号、ネットマスク、ルーター、およびブロードキャスト・アドレスなどの、クライアント情報を提供する。 • DHCP を使用していない場合には、このサービスを使用不可にする • DHCP を使用していないホストとともに、実動サーバーおよびバック・ルーム・サーバー上で使用不可にする
rc.tcpip/dpid2		/etc/rc.tcpip	旧式の SNMP サービス	• SNMP を必要とするのでない限り、使用不可にする
rc.tcpip/gated		/etc.rc.tcpip	インターフェース間でのゲート経路指定	• ルーター機能をエミュレートする • このサービスを使用不可にし、その代わりに、RIP またはルーターを使用する
rc.tcpip/inetd		/etc/rc.tcpip	inetd サービス	• 完全に保護されているシステムでは、このサービスを使用不可にする必要があるが、多くの場合、実際的でない • このサービスを使用不可にすると、一部のメールおよび Web サーバーに必要なリモート・シェル・サービスが使用不可になる

サービス	デーモン	開始点	機能	コメント
rc.tcpip/mrouted		/etc/rc.tcpip	マルチキャスト経路指定	• ネットワーク・セグメント間でマルチキャスト・パケットを送信するためのルーター機能をエミュレートする • このサービスを使用不可にする。その代わりに、ルーターを使用する。
rc.tcpip/names		/etc/rc.tcpip	DNS ネーム・サーバー	• ご使用のマシンが DNS ネーム・サーバーである場合にのみ、このサービスを使用する • ワークステーション、開発マシン、および実動マシンの場合には使用不可にする
rc.tcpip/ndp-host		/etc/rc.tcpip	IPv6 ホスト	• IP バージョン 6 を使用するのでない限り、使用不可にする
rc.tcpip/ndp-router		/etc/rc.tcpip	IPv6 経路指定	• IP バージョン 6 を使用している場合以外は、これを使用不可にします。IP バージョン 6 の代わりにルーターの使用を考慮してください。
rc.tcpip/portmap		/etc/rc.tcpip	RPC サービス	• 必須サービス • RPC サーバーは、**portmap** デーモンを使って登録される。 RPC サービスを位置指定する必要のあるクライアントは、特定のサービスが位置付けられている場所をクライアントに通知するように **portmap** デーモンに依頼する • 残っている唯一のデーモンが **portmap** であるように、 RPC サービスを削減するように管理した場合にのみ、使用不可にする
rc.tcpip/routed		/etc/rc.tcpip	インターフェース間での RIP 経路指定	• ルーター機能をエミュレートする • ネットワーク間でパケット用のルーターを持っている場合には、使用不可にする
rc.tcpip/rwhod		/etc/rc.tcpip	リモート "who" デーモン	• データを収集し、同じネットワーク上のピア・サーバーにブロードキャストする • このサービスを使用不可にする。

サービス	デーモン	開始点	機能	コメント
rc.tcpip/sendmail		/etc/rc.tcpip	メール・サービス	• root ユーザーとして実行する • マシンがメール・サーバーとして使用されるのでない限り、このサービスを使用不可にする • 使用不可にする場合には、以下のいずれかを行う。 – キューを消去するために crontab 内にエントリーを入れる。 **/usr/lib/sendmail -q** コマンドを使用する。 – ご使用のサーバー用のメールが他の何らかのシステムに送達されるように、DNS サービスを構成する
rc.tcpip/snmpd		/etc/rc.tcpip	Simple Network Management Protocol (SNMP)	• SNMP ツールを介してシステムをモニターしない場合には、使用不可にする • SNMP は、クリティカル・サーバー上で必要
rc.tcpip/syslogd		/etc/rc.tcpip	イベントのシステム・ログ	• このサービスを使用不可にすることは推奨しない • サービス妨害攻撃されやすい • どんなシステムでも必要
rc.tcpip/timed		/etc/rc.tcpip	古い時刻デーモン	• このサービスを使用不可にし、その代わりに xntp を使用する
rc.tcpip/xntpd		/etc/rc.tcpip	新しい時刻デーモン	• システム上のクロックの同期を保つ • このサービスを使用不可にする • 他のシステムをタイム・サーバーとして構成し、その他のシステムが ntpdate を呼び出す cron ジョブでそれらに同期化できるようにする
dt login		/usr/dt/config/Xaccess	制限付きでない CDE	• X11 ステーションのグループに CDE ログインを提供していない場合には、 dtlogin をコンソールに制限することができる。
anonymous FTP service		user rmuser -p <username>	匿名 FTP	• 匿名 FTP 機能により、特定のユーザーへの FTP 使用をトレースできないようにする。 • 以下のように、そのユーザー・アカウントが存在する場合には、ユーザー ftp を除去する。**rmuser -p ftp** • ご使用のシステムに ftp できない者のリストを **/etc/ftpusers** ファイルに移植することによって、さらに良いセキュリティーが得られる。

サービス	デーモン	開始点	機能	コメント
anonymous FTP writes			匿名 FTP アップロード	• FTP に属しているべきファイルはない • FTP 匿名アップロードにより、ご使用のシステム上に入れられるコードが間違った動作をする可能性がある • **/etc/ftpusers** ファイルに、許可しないユーザーの名前を入れる • ご使用のシステムに FTP を介して匿名でアップロードすることを許可しないシステム生成ユーザーの例は、以下のとおり。 root、daemon、bin.sys、admin.uucp、guest、nobody、lpd、nuucp、ladp • 以下のように、**ftpusers** ファイルへの所有者権限およびグループ権限を変更する。 **chown root:system /etc/ftpusers** • 以下のように、**ftpusers** ファイルへの許可をさらに厳密な設定に変更する。 **chmod 644 /etc/ftpusers**。
ftp.restrict			システム・アカウントへの ftp	• 外部からのどのユーザーも、**ftpusers** ファイルを使用して root ファイルの置換が許可されるべきではない。
root.access		/etc/security/user	root アカウントへの rlogin/telnet	• **etc/security/user** ファイル内の rlogin オプションを false に設定する • root としてログインする人は、まず、自分の名前の下でログインしてから、 root に **su** する必要がある。これにより、監査証跡が提供される。
snmpd.readWrite		/etc/snmpd.conf	SNMP 読み取り/書き込みコミュニティー	• SNMP を使用していない 場合には、SNMP デーモンを使用不可にする。 • **/etc/snmpd.conf** ファイルでは、コミュニティー・プライベートおよびコミュニティー・システムを使用不可にする • ご使用のシステムをモニターしている IP アドレスに 'public' コミュニティーを制限する
syslog.conf			syslogd を構成する	• **/etc/syslog.conf** が構成されていない場合には、このデーモンを使用不可にする • システム・メッセージをログ記録するために **syslog.conf** を使用している場合には、このサービスを使用可能のままにする

ネットワーク・サービス・オプションの要約

高水準のシステム・セキュリティーを達成するために、ネットワーク・オプションの中には、0 で使用不可になり、1 で使用可能にできるものがあります。 次にリストするものは、**no** コマンドで使用できるその種のパラメーターです。

パラメーター	コマンド	目的
bcastping	/usr/sbin/no -o bcastping=0	ブロードキャスト・アドレスに対する ICMP エコー・パケットへの応答を許可する。 このパラメーターを使用不可にすると、Smurf アタックを防止できる。
clean_partial_conns	/usr/sbin/no -o clean_partial_conns=1	SYN (シーケンス番号を同期化する) アタックを回避するかどうかを指定する。
directed_broadcast	/usr/sbin/no -o directed_broadcast=0	ゲートウェイへのダイレクテッド・ブロードキャストを許可するかどうかを指定する。 0 に設定すると、ダイレクトされたパケットがリモート・ネットワークに到達しないようにすることができる。
icmpaddressmask	/usr/sbin/no -o icmpaddressmask=0	システムが ICMP アドレス・マスク要求に応答するかどうかを指定する。 このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。
ipforwarding	/usr/sbin/no -o ipforwarding=0	カーネルがパケットを転送する必要があるかどうかを指定する。このパラメーターを使用不可にすると、リダイレクトされたパケットがリモート・ネットワークに到達しないようにすることができる。
ipignoreredirects	/usr/sbin/no -o ipignoreredirects=1	受信されるリダイレクトを処理するかどうかを指定する。
ipsendredirects	/usr/sbin/no -o ipsendredirects=0	カーネルがリダイレクト・シグナルを送信する必要があるかどうかを指定する。このパラメーターを使用不可にすると、リダイレクトされたパケットがリモート・ネットワークに到達しないようにすることができる。
ip6srcrouteforward	/usr/sbin/no -o ip6srcrouteforward=0	システムが発信元経路指定された IPv6 パケットを転送するかどうかを指定する。 このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。
ipsrcrouteforward	/usr/sbin/no -o ipsrcrouteforward=0	システムが発信元経路指定されたパケットを転送するかどうかを指定する。 このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。
ipsrcrouterecv	/usr/sbin/no -o ipsrcrouterecv=0	システムが発信元経路指定されたパケットを受け入れるかどうかを指定する。このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。

パラメーター	コマンド	目的
ipsrcroutesend	/usr/sbin/no -o ipsrcroutesend=0	アプリケーションが発信元経路指定されたパケットを送信できるかどうかを指定する。 このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。
nonlocsroute	/usr/sbin/no -o nonlocsrcroute=0	厳密に発信元経路指定されたパケットがローカル・ネットワーク外のホストに発信されている可能性があることをインターネット・プロトコルに通知する。 このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。
tcp_icmpsecure	/usr/sbin/no -o tcp_icmpsecurer=1	ICMP (Internet Control Message Protocol) 発信元および PMTUD (Path MTU Discovery) アタックから TCP 接続を保護する。 ICMP メッセージのペイロードを検査し、TCP ヘッダーのシーケンス番号が受け入れ可能なシーケンス番号の範囲内にあるかどうか調べる。 値: 0=off (デフォルト)。1=on。
ip_nfrag	/usr/sbin/no -o ip_nfrag=200	IP 再アセンブリー・キューに一度に保持できる IP パケットのフラグメントの最大数を指定する (デフォルト値は 200 で、IP 再アセンブリー・キュー上の 1 つの IP パケットのフラグメントを 200 まで保持できる)。
tcp_pmtu_discover	/usr/sbin/no -o tcp_pmtu_discover=0	このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。
tcp_tcpsecure	/usr/sbin/no -o tcp_tcpsecure=7	ぜい弱性から TCP 接続を保護する。 値: 0=保護なし、1=確立された接続に偽 SYN を送信、 2=確立された接続に偽 RST を送信、3=確立された TCP 接続にデータを注入、 5–7=上記ぜい弱性の組み合わせ。
udp_pmtu_discover	/usr/sbin/no -o udp_pmtu_discover=0	TCP アプリケーション用のパス MTU ディスカバリーを使用可能または使用不可にする。このパラメーターを使用不可にすると、発信元経路指定アタックによるアクセスを防止できる。

ネットワーク・チューニング可能オプションについての詳細は、「パフォーマンス・マネージメント」を参照してください。

Trusted AIX

Trusted AIX は、AIX で Multi Level Security (MLS) 機能を使用可能にします。

注**:** MLS はラベル・ベース・セキュリティーとも呼ばれます。

通常の AIX と比べると、Trusted AIX ラベル・ベース・セキュリティーはシステム内のすべてのサブジェクトとオブジェクトのラベルを実装します。

注: Trusted AIX インストール・オプションは、ラベル付きセキュリティー AIX 環境を使用可能にします。 システムのアクセス制御は、Multi Level Security (MLS) 環境を提供するラベルに基づき、以下をサポートします。

- ラベル付きオブジェクト: ファイル、IPC オブジェクト、ネットワーク・パケット、およびその他のラベル付きオブジェクト
- ラベル付きプリンター
- Trusted Network: IPv4 と IPv6 の RIPSO および CIPSO に対するサポート

このモードのインストールを選択したら、通常の AIX の上書きインストールを実行しないで通常の AIX 環境に戻すことはできないことに注意してください。 このモードのインストールを選択する場合は、事前に Trusted AIX 環境に対する必要性を評価してください。 Trusted AIX の詳細については、入手可能な AIX の公開資料を参照してください。

標準 AIX は、情報管理者が基本的なレベルのシステムとネットワーク・セキュリティーを提供できる一連のセキュリティー機能を備えています。 AIX の主なセキュリティー機能は次のとおりです。

- ログインおよびパスワード制御されたシステムとネットワーク・アクセス
- ユーザー、グループ、およびワールドのファイル・アクセス権
- アクセス制御リスト (ACL)
- 監査サブシステム
- ロール・ベース・アクセス制御 (RBAC)

Trusted AIX は、これらの主な AIX オペレーティング・システムのセキュリティー機能を構築して、AIX セキュリティーをさらに強化してネットワーク・サブシステムに拡張します。

Trusted AIX は、AIX アプリケーション・プログラミング・インターフェース (API) と互換性があります。 AIX で実行されるアプリケーションは、Trusted AIX でも実行されます。 ただし、追加のセキュリティー制限により、MLS 認識ではないアプリケーションは、Trusted AIX 環境で動作するために特権が必要な場合があります。 このような状況でアプリケーションのプロファイルを作成するには、**tracepriv** コマンドを使用できます。

Trusted AIX は、追加のセキュリティー機能をサポートするために AIX API を拡張します。 これにより、お客様は独自の安全なアプリケーションをその AIX API と新しい Trusted AIX 拡張を使用して開発できます。

Trusted AIX は、AIX システムが複数のセキュリティー・レベルで情報を処理できるようにします。 これは拡張 B1 セキュリティーに対する米国国防総省 (DoD) TCSEC とヨーロッパ ITSEC 基準を満たすように設計されています。

標準 AIX セキュリティーについては、『基本オペレーティング・システムの保護』および『ネットワークの保護』を参照してください。

Trusted AIX の概要

Trusted AIX は、オペレーティング・システム内のラベル・ベースのセキュリティー機能を提供することにより、標準 AIX オペレーティング・システムのセキュリティーを強化するものです。

Trusted AIX ラベル・ベースの環境は、インストール時のオプションを選択してインストールすることができます。 Trusted AIX をインストールする場合、通常の AIX の上書きインストールを実行しないで、通常の AIX 環境に戻ることはできません。 インストールが完了すると、Trusted AIX 環境は AIX 環境

内に作成された WPAR をすべて含めて、 AIX システム全体に適用されます。ラベル・ベースのセキュリティー (Multi Level Security または MLS とも呼ばれます) は防衛および情報産業でよく使用されますが、一般の産業でも使用できます。 これは Trusted AIX で使用可能なラベルをカスタマイズすることにより実現できます。 Trusted AIX のフレッシュ・インストールにより、標準 MLS インプリメンテーションに従うラベルが提供されます。

Trusted AIX 環境は、通常の AIX と、いくつかの追加パッケージおよびファイルセットで構成されます。 さらに、カーネル・スイッチは、カーネルに Trusted AIX モードでの操作を強制します。 CD または DVD からブートすると、システムは通常の AIX 環境にブートされます。 インストール・メニューが表示されると、インストーラーで Trusted AIX オプションを選択することが可能になり、 MLS 関連ファイルのインストールが開始されます。 インストールが完了すると、インストールは最初のブート再組み立てを開始しなければなりません。 最初のブート・シーケンスでは、「Config Assistant (構成アシスタント)」により各種のユーザーのためのメニューが提供され、 ISSO、SA、および SO ユーザーがセットアップされます。次に、システムはブート操作を完了して、MLS が設定されます。

Trusted AIX は、機密保護に関する以下の 4 つの基本要素によりシステム・セキュリティーを強化します。

- 機密性
- 保全性
- 可用性
- アカウンタビリティー

AIX が提供するセキュリティー機能に加えて、Trusted AIX では次の機能が拡張されます。

機密ラベル **(SL)**
: すべてのプロセスおよびファイルは、それぞれのセキュリティー・レベルに応じてラベル付けされます。 各プロセスは、それぞれのプロセスのセキュリティー範囲内にあるオブジェクトにのみ、アクセスすることができます。

保全性ラベル **(TL)**
: すべてのプロセスおよびファイルは、それぞれの整合性レベルに応じてラベル付けされます。 各ファイルは、それぞれのファイルより整合性レベルの低いラベルを持つプロセスでは書き込むことはできません。 各プロセスは、それぞれのプロセスより整合性レベルの低いラベルを持つファイルから読み取ることはできません。

ファイル・セキュリティー・フラグ
: 個々のファイルに追加のフラグを付けて、セキュリティーに関連した操作を制御することができます。

カーネル・セキュリティー・フラグ
: システム全体で、さまざまなセキュリティー機能を使用可能または使用不可にすることができます。

特権　多くのコマンドおよびシステム・コールは、特定の特権があるプロセスにのみ使用可能です。

権限　各ユーザーには、固有の権限セットを付与することができます。 それぞれの権限により、ユーザーは特定のセキュリティー関連機能を実行することができます。 権限はロールを使用してユーザーに割り当てられます。

ロール
: ロール・ベース・アクセス制御機能は Trusted AIX の一部として、管理任務について選択的委任できる機能を非ルート・ユーザーに提供します。 この委任は、関係のある権限をロールに集め、そのロールを非ルート・ユーザーに割り当てることにより実現されます。

機密性

権限のないユーザー・グループに対する情報の開示を中心として生じる危険性については、機密性を高めることで対処します。

Trusted AIX は、すべてのデータ・リソースを保護するために、オブジェクト再使用およびアクセス制御のメカニズムを提供します。オペレーティング・システムでは、保護されたデータ・リソースにアクセスできるのは特別に許可されたユーザーのみであり、当該ユーザーは意図的な操作であっても、あるいは誤った操作であっても、許可されていないユーザーが保護リソースを使用できるようにすることはできません。

管理者は機密ファイルに関して、フロッピー・ディスクまたは他の取り外し可能メディアへの書き込み、保護されていないプリンターでの印刷、または無許可リモート・システムへのネットワークを介した転送を防ぐことができます。 このセキュリティー保護は、オペレーティング・システムによって実施され、悪意のあるユーザーまたは不正な処理によってバイパスされることはありません。

保全性

権限のないユーザー・グループによる情報の変更を中心として生じる危険性については、保全性を高めることで対処します。

Trusted AIX は、多くのセキュリティー・メカニズムを提供し、データがシステム上で生成されている場合でもネットワーク・リソースからインポートされている場合でも、トラステッド・コンピューティング・ベースおよび保護データの保全性を確保します。 さまざまなアクセス制御セキュリティー・メカニズムによって、許可された人しか情報を変更することはできません。 悪意のあるユーザーまたは不正なプロセスがシステム・リソースを占有または使用不可にするのを防ぐために、Trusted AIX はルート特権を除去します。 特殊な管理権限およびロールにより、ユーザーにルート特権を付与するのではなく、管理責務を分割することができます。

可用性

ホスト・マシンにおけるサービスのアクセス可能性を中心として生じる危険性については、可用性を高めることで対処します。 例えば、不正プログラムによってファイル・スペースが満たされていて新しいファイルを作成できない場合は、アクセスは引き続き可能ですが、可用性がありません。

Trusted AIX は、サービス妨害を行う可能性のある無許可のユーザーおよびプロセスによるアタックからシステムを保護します。 非特権プロセスでは、保護されたファイルおよびディレクトリーに対する読み取りまたは書き込みを行うことはできません。

アカウンタビリティー

システム上のアクションに関して実行される予測不能なプロセスを中心として生じる危険性については、アカウンタビリティーを高めることで対処します。 例えば、システム・ファイルを変更したユーザーまたはプロセスをトレースすることができない場合は、その後このようなアクションを停止させる方法を判断することができません。

今回のセキュリティー機能の強化によって、ユーザーにシステムへのアクセスを許可する前に、すべてのユーザーの識別および認証を確実に行うことができます。 監査サービスにより管理者は、一連の監査可能イベントおよびセキュリティー関連のすべてのシステム・イベントの監査証跡を得ることができます。

Trusted AIX のプロパティー

- Trusted AIX は AIX インストール・メニューを使用してインストールされます。 Trusted AIX のインストールの際に、追加のオプションを選択できます。

- Trusted AIX 環境では、通常の AIX の上書きインストールを実行しないで、通常の AIX 環境に戻ることはできません。
- ルートは Trusted AIX 環境でのログインからは使用不可です。
- Trusted AIX 環境では、作成された WPAR は、いずれもラベル付きセキュリティー環境でも作動します。
- Trusted AIX は MAC (必須アクセス管理) および MIC (必須保全性管理) の両方をサポートします。お客様は MAC と MIC のラベル・セットを別々に定義できます。
- ラベル・エンコード・ファイルは `/etc/security/enc` ディレクトリーに配置されておりラベルからバイナリーへの変換情報を取り込みます。 デフォルトのラベル・エンコード・ファイルは、「Compartmented Mode Workstations (CMW)」ラベル関連命名要件に従います。
- NIM インストールはクライアントから開始されたときにサポートされます。 ルートが MLS システムでのログインには使用不可であるために、サーバーから NIM インストールをプッシュすることはできません。
- JFS2 (J2) ファイルシステム (拡張属性バージョン 2) は、ラベルを AIX に保管するために使用可能になっています。その他のファイルシステム (J1 または NFS など) は、単一レベルのファイルシステム (マウント・ポイントに割り当てられたラベル) として、 Trusted AIX 環境にのみマウントできます。
- Trusted AIX の場合、X 環境は使用不可に設定されます。
- Trusted AIX はネットワーク・ベースのラベル・ベース通信用の CIPSO および RIPSO プロトコルをサポートします。これらのプロトコルは IPv4 と IPv6 の両方でサポートされます。
- 一部の AIX セキュリティー・メカニズムは、通常の AIX と Trusted AIX で共通です。 これらの共通セキュリティー・メカニズムのうちの 2 つは、ロール・ベース・アクセス制御 (RBAC) と保全性検査のための Trusted Execution です。
- ルートは Trusted AIX がインストールされている場合は使用不可であるため、インストーラーはインストール後の最初のブート用に ISSO、SA、および SO ユーザーのパスワードをセットアップする必要があります。 システムはこれらのパスワードが設定されるまで使用不可の状態になっています。
- AIX 6 セキュリティー機能の Redbooks® 資料には、 Trusted AIX のユース・ケースと実例が記載されています。

マルチレベル・セキュリティー

セキュア・システムの主な目的は、サイト・セキュリティー・ポリシーを実行して、責任能力と可用性を与えることです。

Trusted AIX セキュリティー・ポリシーは、許容システム・アクセスのタイプを判別するルールの定義済みセットを提供します。 この場合、ユーザーが自分のアクションに対して責任があり、オペレーティング・システムへの変更が防止されます。

Trusted AIX はアクセス制御と特定の認識すべき基準を使用して、ファイル、ディレクトリー、プロセス、およびデバイスへのアクセスを制御します。

Trusted AIX は、すべてのセキュリティー関連イベントの監査証跡を維持します。 この監査証跡により、個人の責任能力において、有効ユーザー ID および実ユーザー ID をプログラム (**su** コマンドなど) で変更することさえも可能になります。 また、Trusted AIX は、権限と最小特権 (ユーザーまたはプロセスが操作を実行できるようにする最も制限的な特権セットの付与) を持つ特定の個人に対して管理機能を制限します。

識別および認証

識別および認証 (I&A) セキュリティー・メカニズムは、システムへのアクセスのそれぞれの個別要求が、適切に識別されて認証されることを保証するという責任を持ちます。 識別にはユーザー名が必要であり、認証にはパスワードが必要です。

すべての Trusted AIX アカウントはパスワードで保護されます。 ISSO (情報システムのセキュリティー担当者) は、ユーザーがパスワードの長さと複雑性制約に従って、ユーザー自身のパスワードを選択できるようにシステムを構成します。 また、ISSO はユーザー単位を基本にして、最小および最大パスワード経過日数パラメーター (有効期限) についても、パスワード有効期限前に警告することも含めて、指定することができます。

識別および認証のセキュリティー・メカニズムには、すべてのユーザー名とユーザー ID が固有であることが必要です。 ログインでは、有効なパスワードを持たないアカウントは使用できません。 ISSO ロールを持つユーザーは、すべてのユーザーに対する初期パスワードを追加する必要があります。 各ユーザーには監査目的のために使用される追加の固有 ID が割り当てられます。

保管されるのはパスワードの暗号化フォームのみです。 プレーン・テキスト・フォームでは、パスワードはシステムに保管されません。 暗号化されたパスワードは、特権プロセスによるものを除くアクセスが保護されている、シャドー・パスワード・ファイルに保管されます。 詳細情報については、『**passwd** コマンド』を参照してください。

Trusted AIX システムは 2 つのタイプのアカウント (システム・アカウント、およびユーザー・アカウント) を認識します。 システム・アカウントは 128 未満のユーザー ID を持っています。 システム・アカウントは関連付けられたパスワードを持つことが可能ですが、システムへのログオンでは使用できません。

任意アクセス制御

任意アクセス制御 (DAC) は、ファイルまたはディレクトリーの所有者の管理下にあるセキュリティー・アスペクトです。

UNIX アクセス権

リソースに対する所有者権限をもつユーザーには、次のことが可能です。

- 他のユーザーに直接アクセスできるようにする
- 他のユーザーにコピーへのアクセスを認可する
- プログラムを元のリソースに (例えば、SUID プログラムを使用して) アクセスできるようにする

従来の UNIX 許可ビット・メソッド (owner/group/other および read/write/execute) は、この DAC 機能の例です。

許可ビットにより、ユーザーは、他のユーザーおよびグループに対してファイル内のデータへのアクセスを (知っておくべき基準に基づいて) 認可または否認することができます。このタイプのアクセスは、ユーザー ID およびユーザーが属するグループに基づいています。 すべてのファイルシステムのオブジェクトに、関連するアクセス権があり、所有者、グループ、およびワールドに対するアクセスが記述されています。

ファイルの所有者は、**chown** および **chgrp** コマンドを使用してファイルの所有権またはグループを変更することによって、他のユーザーにアクセス権を付与することもできます。

umask

ファイルが作成される時には、すべての許可ビットは最初、オンになります。 その後、ファイルの特定の許可ビットは、ログイン処理時に設定された umask プロセスによって除去されます。 デフォルトの umask は、ユーザーのシェルによって作成されたすべてのファイルおよびユーザーのシェルから実行されるすべてのコマンドに適用されます。

デフォルトでは、カーネル項目の umask 設定は 000 (すべてのユーザーがすべてのアクセス権を使用できる状態にする) です。 AIX では、カーネル umask は 022 (グループおよびワールドの書き込み許可ビットをオフにする) に設定されます。 ただし、ユーザーは必要に応じてこの設定を指定変更することができます。

注**:** umask を 022 よりアクセスが容易な設定に変更する際には、細心の注意が必要です。 ファイルおよびプロセスにおいてより多くのアクセス権を許可すればそれだけ、システム全体のセキュリティーは低下します。
デフォルトの umask 設定を指定変更するには、次の 2 とおりの方法があります。

- `.profile`、`.login`、または `.chsrc` ファイルの umask 値を変更することができます。 この変更は、ログイン・セッション時に作成されるすべてのファイルに影響を及ぼします。
- **umask** コマンドを使用して、個別のプロセスで umask レベルを設定することができます。 **umask** コマンドを実行した後、作成されるすべての新規ファイルは、次のいずれかのイベントが起こるまでは新しい umask 値によって影響を受けます。
  - **umask** コマンドを再実行する

    または
  - **umask** コマンドが発行されたシェルを終了する

引数を指定せずに **umask** コマンドを実行した場合は、**umask** コマンドはセッションの現行の umask 値を戻します。

プロファイルの中に umask 値を指定しないことにより、ログイン・セッションがカーネルの 022 umask 値を継承できるようにしてください。 022 よりアクセスが容易な umask 値を使用するには、細心の注意が必要です。

特定のファイルに追加のアクセス権が必要な場合は、これらのファイルが作成された後に、**chmod** コマンドを注意深く使用してアクセス権を設定してください。

アクセス制御リスト

標準の UNIX 許可ビットおよび umask 値に加えて、AIX はアクセス制御リスト (ACL) もサポートします。

UNIX 許可ビットでは、ファイル所有者、1 つのグループ、およびシステム上の全ユーザーに関するアクセスしか制御しません。 ACL により、ファイル所有者は追加の特定のユーザーおよびグループにアクセス権限を指定することができます。 許可ビットと同様に、ACL もファイルまたはディレクトリーなどの個別のシステム・オブジェクトに関連付けられます。

setuid および setgid 許可ビット

setuid および setgid 許可ビット (ユーザー ID の設定およびグループ ID の設定) により、プログラムを実行しているユーザーのユーザー ID またはグループ ID ではなくファイル所有者のユーザー ID またはグループ ID でプログラム・ファイルを実行することができます。これは、ファイルに関連付けられてい

る setuid および setgid ビットを設定することによって達成されます。 これにより、保護されたサブシステムの開発が許可され、ユーザーは特定のファイルを、そのファイルを所有することなくアクセスおよび実行することができます。

オブジェクトの作成時に setgid ビットが親ディレクトリーに設定された場合は、新しいオブジェクトは、オブジェクトの作成者のグループではなく、親ディレクトリーと同じグループをもつことになります。 ただし、setuid ビット・セットを使用してディレクトリーで作成されたオブジェクトは、ディレクトリーの所有者ではなくオブジェクトの作成者によって所有されます。 親ディレクトリーの setuid/setgid ビットは、サブディレクトリーの作成時にサブディレクトリーによって継承されます。

setuid および setgid 許可ビットには、セキュリティー・リスクを生じさせる可能性があります。 所有者としてルートとともに実行されるように設定されているプログラムには、基本的にシステムへのアクセスが制限されない可能性があります。 ただし、Trusted AIX システムでは、特権およびその他のアクセス制御を使用して、このセキュリティー・リスクをかなり軽減します。

ロール・ベース・アクセス制御エレメント

Trusted AIX は、ロール・ベース・アクセス制御 (RBAC) をサポートします。 RBAC と root/システム・スーパーユーザー固有のシステムが機能するオペレーティング・システム・メカニズムは、通常のユーザーが自分に割り当てられたロールを使用しても実行できます。

AIX RBAC の中心となるエレメントは、次のとおりです。

権限　これらの文字列は、直接名前で表し、名前で制御する特権操作を示します。 例えば、権限の文字列 `aix.network.manage` は、AIX オペレーティング・システムのネットワーク管理機能を定義します。

特権　特権はプロセスが特定のシステム制限と限度をバイパスできるようにするプロセスの属性です。 特権はプロセスに関連付けられ、通常、特権コマンドを実行することで獲得されます。

ロール
AIX RBAC のロール・エレメントにより、ユーザーは管理機能のセットをシステムで結合でき、通常のユーザーが管理するためにこれらの機能を割り当てることができます。 AIX のロールは、権限 (システム権限とカスタム権限の両方が可能) と他のロール (サブロール) の集合から構成されます。

ロール・ベース・アクセス制御の詳細については、RBAC を参照してください。

必須アクセス制御

必須アクセス制御 (MAC) は、オブジェクトの機密性およびユーザーの認可に基づいてオブジェクトへのアクセスを制限するシステム実施の方法です。 対照的に、任意アクセス制御は、システムによってではなく個別のファイル所有者によって実施されます。

MAC 用のラベルの使用

Trusted AIX では、ラベルのシステムを使用して MAC を実施します。 Trusted AIX システムでは、すべての名前付きオブジェクトには、そのオブジェクトの機密レベルを識別するための機密ラベル (SL) があります。 プロセスにも SL があります。 プロセスの SL は、プロセスがどのレベルの機密情報にアクセスできるかを示します。 通常、プロセスには、オブジェクトにアクセスするために、オブジェクトの機密レベルと等しいかそれ以上の機密レベルがなければなりません。 SL を使用して、ファイルを読み取り専用としてアクセス可能にするか、または通常のユーザーによるファイルへのアクセスを完全に防止することができます。

ファイル、IPC オブジェクト、ネットワーク接続、およびプロセスのようなすべてのシステム・オブジェクトに SL があります。 オブジェクトが作成される時に、SL はそのオブジェクトに自動的に入れられます。すべてのコア・ダンプはオブジェクトとみなされ、システムによって自動的にラベル付けされます。

Trusted AIX のインストール前に存在するオブジェクトは、これらのオブジェクトが Trusted AIX のインストール後にアクセスされるときにデフォルトの SYSTEM_LOW SL (SLSL) を受け取ります。 SL は、これらのオブジェクトに永続的に設定されるわけではありません。 SL を設定するためには、**settxattr** コマンドをオブジェクトに対して実行しなければなりません。 Trusted AIX のインストール後に作成されるオブジェクトの場合は、オブジェクトの SL は作成プロセスの SL に設定されます。

ユーザーおよびラベル

システムは各ユーザー・アカウントに、システムのデフォルトまたはユーザー固有の設定のいずれかによって有効な SL の範囲を割り当て、ユーザーはこの範囲内でのみ操作を行うことができます。 プロセスまたはユーザーは、プロセスまたはユーザーの現行の機密ラベルでファイルおよびディレクトリーの作成のみを行うことができ、システム強制の MAC の制限の対象となるファイルの読み取りおよび書き込みのみを行うことができます。

MAC の実施

必須アクセス制御は、プロセスがファイルシステム・オブジェクトのオープン、ファイルシステム・オブジェクトの属性の検索、プロセスへのシグナルの送信、STREAM を使用したデータの転送、またはネットワーク・インターフェースを使用したパケットの送受信を試みるときは常に実施されます。 すべてのファイルシステム・オブジェクトへのアクセスは、MAC と DAC の両方の基準を満たした場合にのみ可能です。 ユーザーがファイルへのアクセスを試みると、許可ビットまたは ACL のような DAC 制限が検査される前に MAC 制限が実施されます。

ファイルシステム・オブジェクトへのアクセスは、オブジェクトの SL だけでなく、オブジェクトが常駐するディレクトリーの SL によっても制限されます。 そのため、ファイルシステム・オブジェクトは、オブジェクト自体の SL とは異なる機密レベル (ディレクトリーの SL) で保護することができます。 ファイルシステム・オブジェクトは、1 つ以上のディレクトリーに配置されている複数の名前 (リンク) をもつことができます。 それぞれの名前 (リンク) はリンクが指すファイルと同じ SL で保護されますが、リンクはさまざまなレベルで保護されたディレクトリーに入っているので、さまざまなリンクの有効な保護が異なることがあります。

オブジェクトの名前は、オブジェクトが常駐するディレクトリーに保管されます。 そのため、そのディレクトリーにアクセスするすべてのプロセスで、ディレクトリーの中のすべてのオブジェクトの名前を表示することができます。ただし、適切なアクセスを行うプロセスのみが、いずれかのオブジェクトからの読み取りまたはオブジェクトへの書き込みを行うことができます。

SL のリスト作成および変更

システム上のオブジェクトおよびプロセスの SL は、**lstxattr** コマンドを用いて表示することができ、**settxattr** コマンドを用いて変更することができます。

適切な許可をもつユーザーおよび適切な特権をもつプロセスのみが、ファイルまたはプロセスの SL を変更することができます。

settxattr コマンドを用いて、ファイルシステム・オブジェクトの SL を低レベル SL に変更するためには、ユーザーは aix.mls.label.sl.downgrade 許可をもたなければなりません。 ファイルシステム・オブジェクトの SL をアップグレードするためには、ユーザーは aix.mls.label.sl.upgrade 許可をもたなけ

ればなりません。プロセスの SL を変更するためには、アップグレードには aix.mls.proc.sl.upgrade 許可を、ダウングレードには aix.mls.proc.sl.downgrade 許可をユーザーがもっている必要があります。

オープン・ファイル記述子に関する MAC

読み取り/書き込みおよび単純なファイル・アクセスの場合は、プロセスがファイルにアクセスするときに MAC 検査が行われます。いったんプロセスがファイルのファイル記述子をもつと、プロセスの SL がファイルの SL よりも低いレベルに変更されたとしても、プロセスはファイルの読み取りおよび書き込みを行うことができます。ただし、ファイル所有者、許可、ラベル、および特権の設定などの一部の操作では、プロセスがファイル記述子を獲得した後でアクセス検査を実行します。

つまり、MAC 検査および分割ディレクトリー・パスの解決は、プロセスがファイル記述子を使用してファイルにアクセスするときは実行されません。 ファイルまたはプロセス (あるいはその両方) の SL は変更されることがあり、アクセスは引き続き許可されます。

必須保全性制御

必須保全性制御 (MIC) は、オブジェクトの保全性およびユーザーの認可に基づいてオブジェクトへのアクセスおよびオブジェクトの変更を制限するシステム実施の方法です。 MAC はオブジェクトの機密性に関係があり、MIC はオブジェクトの信頼性に関係があります。

MIC 用のラベルの使用

Trusted AIX では、ラベルのシステムを使用して MIC を実施します。 Trusted AIX システムでは、すべての名前付きオブジェクトには、そのオブジェクトの保全性レベルを識別するための保全性ラベル (TL) があります。 プロセスにも TL があります。 プロセスの TL は、プロセスがどのレベルの保全性情報にアクセスできるかを示します。 TL が高位であればあるほど、オブジェクトまたはプロセスの信頼性が増します。

プロセスは、オブジェクトを変更するためには、少なくともオブジェクトと同等の信頼性がなければなりません。 したがって、プロセスには、オブジェクトの TL と同等かそれ以上の TL がなければなりません。そのため、保全性ラベルを使用して、ファイルへのアクセスを読み取り専用にすることができます。

さらに、プロセスは、プロセス自体より信頼性の低いオブジェクトからのデータを使用することはできません。このため、オブジェクトには、プロセスの TL と同等かそれ以上の TL がなければなりません。

ファイルおよびプロセスのようなすべてのシステム・オブジェクトに TL があります。 オブジェクトが作成される時に、TL はそのオブジェクトに自動的に入れられます。 すべてのコア・ダンプはオブジェクトとみなされ、システムによって自動的にラベル付けされます。

Trusted AIX のインストール前にシステム上に存在するオブジェクトは、これらのオブジェクトが Trusted AIX のインストール後にアクセスされるときにデフォルトの SYSTEM_LOW TL (SLTL) を受け取ります。 TL は、これらのオブジェクトに永続的に設定されるわけではありません。 TL を設定するためには、**settxattr** コマンドをこれらのオブジェクトに対して実行しなければなりません。 Trusted AIX のインストール後に作成されるオブジェクトの場合は、これらのオブジェクトの TL は、オブジェクトを作成したプロセスの保全性レベルに設定されます。

ユーザーおよびラベル

システムは各ユーザー・アカウントに、システムのデフォルトまたはユーザー固有の設定のいずれかによって有効な TL の範囲を割り当て、ユーザーはこの範囲内でのみ操作を行うことができます。 プロセスまたはユーザーは、プロセスまたはユーザーの現行の TL でファイルおよびディレクトリーの作成のみを行う

ことができ、システム強制の MIC の制限の対象となるファイルの読み取りおよび書き込みのみを行うことができます。

MIC の実施

必須保全性制御は、MAC が実施されるたびに実施されます。 さらに、MIC は、ファイルまたはディレクトリーが削除または名前変更されるときに実施されます。

TL の変更

オブジェクトおよびプロセスの TL は、**lstxattr** コマンドを用いて表示することができ、**settxattr** コマンドを用いて変更することができます。

適切な許可をもつユーザーおよび適切な特権をもつプロセスのみが、ファイルまたはプロセスの TL を変更することができます。 **settxattr** コマンドを用いて、ファイルシステム・オブジェクトの TL を低レベル TL に変更するためには、ユーザーは `aix.mls.label.tl.downgrade` 許可をもたなければなりません。 ファイルシステム・オブジェクトの TL をアップグレードするためには、ユーザーは `aix.mls.label.tl.upgrade` 許可をもたなければなりません。プロセスの TL を変更するためには、アップグレードには `aix.mls.proc.tl.upgrade` 許可を、ダウングレードには `aix.mls.proc.tl.downgrade` 許可をユーザーがもっている必要があります。

NOTL

ファイルシステム、ipc オブジェクト、またはプロセスに適用できる特殊な TL、`NOTL` があります。 オブジェクトまたはプロセスに `NOTL` TL がある場合は、MIC 検査はオブジェクトまたはプロセスに関しては行われません。 特権ユーザーのみが、TL を `NOTL` に設定でき、TL が現在 `NOTL` である場合は TL を変更することができます。

オープン・ファイル記述子に関する MIC

読み取り/書き込みおよび単純なファイル・アクセスの場合は、プロセスがファイルにアクセスするときに MIC 検査が行われます。いったんプロセスがファイルのファイル記述子をもつと、プロセスの TL がファイルの TL よりも低いレベルに変更されたとしても、プロセスはファイルの読み取りおよび書き込みを行うことができます。 ただし、ファイル所有者、許可、ラベル、および特権の設定などの一部の操作では、プロセスがファイル記述子を獲得した後でアクセス検査を実行します。 つまり、MIC 検査は、プロセスがファイル記述子を使用してファイルにアクセスするときは実行されません。 ファイルまたはプロセス (あるいはその両方) の TL は変更されることがあり、アクセスは引き続き許可されます。

ラベル

Trusted AIX システムでサブジェクトとオブジェクトのセキュリティー・レベルを表すために、ラベルが使用されます。 システムで使用されるラベルとラベル間の関係は、ISSO が定義します。

機密ラベル **(SL):**

各サブジェクトとオブジェクトに関連付けられた SL は、アクセス制御の Bell-LaPadula モデルに基づいた必須アクセス制御ポリシーを実行するために使用されます。

SL は次の 2 つの部分から構成されます。

- 階層機密区分
- 1 つ以上のコンパートメントの集合

それぞれのインストール場所では、システムのラベルの名前と関係を定義できます。 システム管理者は、サイト・ポリシーに応じてこれらの名前と関係をラベルのエンコード・ファイルに設定できます。

SL 機密区分**:**

機密区分は階層順序を持っており、機密性のレベルを表します。

例えば、Top Secret、Secret、および Unclassified がサイトで有効な機密区分である場合、Top Secret は Secret より機密性が高く、Secret は Unclassified より機密性が高くなります。 Trusted AIX は、最大 32,000 の階層の機密区分をサポートします。

SL コンパートメント**:**

コンパートメントは、トピックまたはワークグループを表します。 各コンパートメントには、NATO や CRYPTO などの名前があります。

コンパートメントは固有の組み込み順序はありませんが、ISSO はどのコンパートメントと機密区分を結合できるかについて制約を課すことができます。 Trusted AIX は、最大 1,024 個のコンパートメントをサポートします。

SL コンポーネント**:**

人間が理解できる形式で、SL はエレメントの文字列により表されます。最初のエレメントは機密区分を表し、その他のエレメントはコンパートメントを表します。 これらのエレメントはスペースで区切られます。

例えば、ファイルにブラジル経済に関する極秘情報が含まれている場合、ファイルの階層機密区分は「極秘」(TS) となり、そのコンパートメントは「ブラジル」(B) と「経済」(e) となります。人間が理解できる SL の形式は、TS B e または Top Secret Brazil economy となります。

SL 関係**:**

システム・ユーザーにとって、ラベル間の関係とラベルの使用法を理解することは重要です。

MAC ラベル間の関係には、次の 3 つのタイプがあります。

- 優位
- 同等
- 比較不可

優位

ある SL (L1) が別の SL (L2) に対して優位と言えるのは、次の条件が両方とも真である場合のみです。

- L1 の機密区分が L2 の機密区分と等しいかそれより上である。
- L1 のコンパートメントのセットが L2 のコンパートメントのセットを完全に含んでいる。

例えば、最高機密情報の SL L1 がコンパートメント A および B にあり (TS A B)、機密情報の別の SL L2 がコンパートメント A にはあって B にはない (S A) と仮定した場合には、TS A B は S A に対して優位となります。これは、機密区分 TS が機密区分 S に対して優位であり、L1 のコンパートメントのセットが L2 のコンパートメントのセットを完全に含んでいるためです。 この例では、L2 は L1 に対して優位ではありません。

表 34. SL 優位

L1		L2		優位
ラベル	コンパートメント	ラベル	コンパートメント	
TOP SECRET	A,B	SECRET	A	L1 > L2

同等

ある SL (L1) が別の SL (L2) と同等と言えるのは、次の条件が両方とも真である場合のみです。

- L1 の機密区分が L2 の機密区分と等しい。
- L1 のコンパートメントのセットが L2 のコンパートメントのセットと等しい。

2 つのラベルが同等である場合は、それぞれのラベルはもう 1 つのラベルに対して優位となります。 例えば、最高機密情報が入っているファイルの SL がコンパートメント A にあり (TS A)、最高機密情報が入っている別のファイルもコンパートメント A にある (TS A) と仮定した場合には、2 つの SL は同等であり、お互いに優位となります。

表 35. SL 同等

L1		L2		優位
ラベル	コンパートメント	ラベル	コンパートメント	
TOP SECRET	A	TOP SECRET	A	L1 = L2

比較不可

2 つの SL が結合不可です (L1 は L2 と同等ではなく、L1 は L2 より優位ではなく、L2 も L1 より優位ではありません)。 ある SL (L1) が別の SL (L2) に対して比較不可と言えるのは、次の条件が真である場合のみです。

- L1 のコンパートメントのセットが L2 のセットを完全には含んでおらず、L2 が L1 のセットを完全には含んでいない。 そのため、L1 と L2 は結合不可とみなされます。

例えば、ラベル L1 のファイルがコンパートメント A および B の最高機密情報を含んでいて (TS A B)、L2 がコンパートメント C の機密情報を含んだファイルのラベルである (C C) と仮定した場合には、L1 は L2 に対して比較不能です。

表 36. 比較不可の SL

L1		L2		優位
ラベル	コンパートメント	ラベル	コンパートメント	
TOP SECRET	A, B	CLASSIFIED	C	-

保全性ラベル **(TL):**

TL はシステム・オブジェクトまたはプロセスでのトラストのレベルを表します。 TL の構造は SL の構造と同じですが、ただし、TL に階層型分類だけでコンパートメントはありません。

プロセスの TL がオブジェクトの TL より優位にある場合のみ、そのプロセスを変更または削除できます。 プロセスの TL がオブジェクトの TL およびそのオブジェクトが存在するディレクトリーの TL の両方より優位にある場合のみ、そのプロセスを削除または名前変更できます。 オブジェクトの TL がプロセスの TL より優位にある場合のみ、そのプロセスはそのオブジェクトをアクセスできます。

オブジェクトまたはプロセスの TL を判別するには、**lstxattr** コマンドを使用します。オブジェクトまたはプロセスの TL を変更するには、**settxattr** コマンドを使用します。

サブジェクトとオブジェクトのラベル:

Trusted AIX では、プロセスはサブジェクトとして識別され、各プロセスには SL があります。

MAC 検査に使用される SL は、有効 SL (ESL) と呼ばれます。 ESL はプロセスの認可範囲内になければなりません。 認可範囲には、上限と下限があります。 上限は最大認可 (最大 CL) と呼ばれ、下限は最小認可 (最小 CL) と呼ばれます。 ESL、最大 CL、および最小 CL は、プロセスの資格情報構造に保管され、プロセスの作成時に割り当てられます。 最大 CL は最小 CL と ESL を支配し、ESL は最小 CL を支配しなければなりません。 プロセスの SL をリストして設定するには、**settxattr** と **lstxattr** コマンドを使用できます。

システム内にある各種のオブジェクトへのアクセスは制御する必要があります。 オブジェクトは、以下のいずれかです。

- プロセス
- ファイル (データ・ファイルまたはバイナリー)
- IPC オブジェクト、ネットワーク・パケットなど

MLS システム上のすべてのオブジェクトおよびサブジェクトはラベル付けされます。

ディレクトリー
: ディレクトリーは SL 範囲 (最小 SL と最大 SL) と関連付けられます。最大 SL 最小 SL と同等か、または上位でなければなりません。 ディレクトリー内のすべてのファイルは、この範囲内にあります。

ファイル
: 通常のファイルは 2 つの SL と関連付けられますが、その値は常に同じです。 したがって、効率的にこれらのファイルは SL を 1 つだけ持っています。 シンボリック・リンクは、これらの SL に対して異なる値を持つことが可能です。

特殊ファイル
: デバイス、tty、fifo などの特殊ファイルは最大および最小 SL と関連付けられます。 ディレクトリー、ファイル、および特殊ファイルは、プロセスが最小および最大 TL と関連付けられている保全性ラベル (TL) を 1 つだけ持っています。

プロセス
: すべてプロセスは、最大および最小保全性認可範囲と同様に、最大および最小機密認可範囲と関連付けられています。 これらの値はユーザーの認可値から継承されます。 プロセスが実行している際の機密および保全性レベルは、「有効機密レベル」および「有効保全性レベル」と呼ばれています。

ユーザーの認可ラベル:

ユーザーは最大および最小の機密認可ラベル (SCL) と最大および最小の保全性認可ラベル (TCL) を持っています。

最大および最小の機密認可ラベル

各ユーザーは最大機密認可ラベル (最大 SCL) を持っています。 ユーザーの有効 SL は、最大 SCL によって支配されなければなりません。 最大 SCL は、特定のユーザーが機密性の高いデータを表示するのを制限するために使用されます。 最小 SCL は、セキュリティー・レベルの高いユーザーがセキュリティー・レベルの低いユーザーにデータを送信するのを防止するために使用されます。

例えば、ユーザー A が PUBLIC_A の最大 SCL と最小 SCL を持っていて、ユーザー B が PUBLIC_B の最大 SCL と最小 SCL を持っているとします。最小 SCL がない場合、ユーザー A は IMPL_LO の有効 SL でログインして、ユーザー B が読み取れるファイルに書き込むことで、情報をユーザー B に対して送信できます。 ただし、最小 SCL がある場合は、ユーザー A は PUBLIC_A でログインする必要があり、PUBLIC_A でしかファイルを書き込めません。PUBLIC_A で書き込まれるファイルは、ユーザー B が読むことはできません。

最大および最小の保全性認可ラベル

各ユーザーは最大保全性認可ラベル (最大 TCL) も持っています。 ユーザーの有効 TL は、最大 TCL によって支配されなければなりません。 最大 TCL は、特定のユーザーが機密性の高いデータを表示するのを制限するために使用されます。 最小 TCL は、セキュリティー・レベルの高いユーザーがセキュリティー・レベルの低いユーザーにデータを送信するのを防止するために使用されます。

ファイルシステム・オブジェクト上のラベル:

すべてのファイルには特定のセキュリティー情報が含まれます。 新規ファイルが作成されるときに、このファイルを作成したプロセスとして同じ SL を持ちます。 ファイル内の情報の SL はファイルの SL を上下することにより、アップグレードまたはダウングレードできます。

ディレクトリーは作成されるときに、最小および最大 SL を割り当てられます。 作成に際して、両方とも作成プロセスの有効 SL と等しく設定され、基本的には単一レベルのディレクトリーが作成されます。 これらの SL を変更できるのは、適切な特権および権限を持つユーザーだけです。 新規オブジェクトを作成するプロセスの有効 SL がこのディレクトリーの SL の範囲内にある場合のみ、新規オブジェクトをこのディレクトリーに作成できます。

通常、ウィンドウが、ユーザーの有効 SL に等しい SL を持つ分離された子プロセスとして作成されます。 デバイス (例えば、ウィンドウに関連付けられた疑似端末装置) も、これに関連付けられた SL を持っています。 プロセス間通信で使用されるデバイスである名前付きパイプは、この名前付きパイプを作成したプロセスの有効 SL を継承します。 プロセス間通信の双方向データ・チャネルを提供するために使用されるデバイスであるストリームも、ストリームを作成したプロセスの有効 SL を継承します。

すべてのデバイスは最小 SL および最大 SL を持っています。 最大 SL は最小 SL より優位にあります。 デフォルトでは最小 SL と最大 SL は等しく設定されます。 プロセスの SL がデバイスまたはディレクトリーの最小 SL より優位にある場合、プロセスはそのようなデバイスのみ読み取りモードでアクセスできます。 プロセスの SL がデバイスまたはディレクトリーの最小および最大 SL によって定義される範囲内にある場合、プロセスはそのようなデバイスのみ書き込みモードでアクセスできます。

ファイル・セキュリティー・フラグ

オブジェクト に、プロセスがオブジェクトを処理する方法に影響を与えるファイル・セキュリティー・フラグ (FSF) のマークを付けることができます。 FSF および各 FSF の設定に必要な特権のリストについては、「ファイル・セキュリティー・フラグ」を参照してください。 プロセスはファイル・セキュリティー・フラグを持っていません。

ファイルの除去:

以下に該当する場合は、ファイルシステムからオブジェクトを除去できます。

- オブジェクトの除去を試行するプロセスは、ファイルを含むディレクトリー内のファイル名を表示できなければなりません。 すなわち、プロセスにはオブジェクトを除去する元のディレクトリーまでのパス

の各ディレクトリーに検索アクセスが必要であり、これらの各ディレクトリーを支配する有効 SL が必要です。 ファイル名を表示するには、**ls** コマンドを使用します。
- プロセスにはオブジェクトを除去する元のディレクトリーに対する書き込みアクセスが必要です。

印刷ファイル*:*

プリンター・サブシステムは、適切な機密ラベルを使用してすべての出力に自動的にラベル付けを行います。 各印刷ジョブに、セキュリティー関連ラベルとマーキングをすべて表示するバナー・ページとトレーラー・ページが自動的に提供されます。

ファイルのバックアップおよび復元*:*

backup コマンドを使用して AIX のディスクまたはテープに書き込む場合、SL がデータに含まれます。

backup コマンドまたは **restore** コマンドを使用して、磁気テープまたはディスクからラベルなしのデータをインポートまたはエクスポートするには、SO 権限が必要です。 ラベルなしデータが書き込まれると、そのデータに対してファイルの SYSTEM_LOW のデフォルト SL、およびディレクトリーの SYSTEM_LOW から SYSTEM_HIGH までの SL 範囲が割り当てられます。

IPC オブジェクトのラベル*:*

AIX IPC 機能はすべて中間オブジェクトの作成とアクセスに使用されます。

AIX では、異なる 3 つの IPC 機能が定義されます。
- メッセージ・キュー
- セマフォー
- 共有メモリー

これらはプロセス間通信のための中間オブジェクト (IPC オブジェクト) の作成とアクセスに使用されます。 各 IPC オブジェクトは、ファイルを保護する属性と同様の属性セットで保護されます。 これらの属性は次のとおりです。
- オブジェクト所有者のユーザー ID とグループ ID
- オブジェクト作成者のユーザー ID とグループ ID
- リソース・アクセス・モード。これはファイル・アクセス権ビットと類似しています。 各オブジェクトは、ワールド、グループ、オブジェクト所有者に対する読み取り、書き込み、および実行のアクセス権限を持っています。
- リソース使用率を追跡するシーケンス番号
- リソースを識別するキー

他のシステム・オブジェクトと同様、Trusted AIX はこれらの属性を追加のセキュリティー属性で拡張します。 Trusted AIX システムでは、すべての IPC オブジェクトに次の属性があります。
- 機密ラベル (SL)
- 保全性ラベル (TL)

settxattr コマンドを使用すると、IPC オブジェクトのすべてのセキュリティー属性を表示できます。 IPC オブジェクトの属性を読み取るには、そのオブジェクトに対する DAC READ と MAC READ アクセス権限が必要です。

***IPC* オブジェクトへのアクセス:**

IPC オブジェクトは、「Trusted AIX プログラミング」のトピックで解説しているいくつかのシステム・コールによって作成、削除、およびアクセスされます。通常のユーザーは、これらの操作を行いません。このトピックでは、IPC オブジェクトの作成、削除、およびアクセスについてのルールを概説します。

IPC オブジェクトにアクセスするには、プロセスが DAC、MIC、および MAC のアクセス検査に合格しなければなりません。

DAC アクセス検査は、オブジェクトおよびプロセスのユーザー ID とグループ ID のモード (所有者、グループ、またはワールド) に基づいています。 プロセス有効 UID がオブジェクト所有者 UID またはオブジェクト作成者 UID のいずれかと同じである場合は、プロセスは IPC オブジェクトに DAC 所有者としてアクセスできます。このことは、DAC グループ・アクセスにも適用されます。

MAC アクセスは、プロセスおよびオブジェクトの SL に基づいています。 MIC アクセスは、プロセスおよびオブジェクトの TL に基づいています。

IPC オブジェクトの内容に関するアクセス・ルールは、IPC オブジェクトの属性に関するものと同じです。 IPC オブジェクトの内容または属性を読み取るには、DAC READ、MIC READ、および MAC READ アクセス権限が必要です。 IPC オブジェクトに書き込みを行うには、DAC WRITE、MIC WRITE、および MAC WRITE アクセス権限が必要です。

IPC オブジェクト属性は、IPC オブジェクトの内容より厳重に制限されています。 そのため、IPC オブジェクト属性の変更には、より高い特権が必要です。 モードのような標準 AIX 属性を変更するには、プロセスにはオブジェクトに対する DAC OWNER および MAC WRITE アクセス権限が必要です。 IPC オブジェクトの SL を変更するには、プロセスに以下のすべてが含まれていなければなりません。

- PV_SL_PROC 特権
- DAC OWNER (ダウングレードのみ)
- DAC WRITE
- MAC WRITE
- PV_SL_UG 特権 (SL のアップグレードの場合)、または PV_SL_DG 特権 (SL のダウングレードの場合)
- PV_MAC_CL (既存の SL または新規 SL がプロセスの許可の外にある場合)
- MIC WRITE

IPC オブジェクトの TL を変更するには、プロセスに以下のすべてが含まれていなければなりません。

- PV_TL 特権
- DAC OWNER
- MAC WRITE
- MIC WRITE

さらに、メモリー内の共有メモリー・セグメントをロックまたはロック解除するためには、プロセスに PV_KER_IPC_O 特権がなければなりません。プロセスには、msgctl サブルーチンの中のメッセージ・キューの msg qbytes を変更するための PV_KER_IPC 特権も必要です。

関連概念:

504 ページの『Trusted AIX プログラミング』
システム・セキュリティーは、トラステッド・コンピューティング・ベース (TCB) のソフトウェア、ハードウェア、およびファームウェアに依存します。 これにはオペレーティング・システムのカーネル全体、すべてのデバイス・ドライバーと System V STREAMS モジュール、カーネル・エクステンション、およびすべてのトラステッド・プログラムが含まれます。 セキュリティー決定を行う際にこれらのプログラムで使用されるファイルもすべて TCB の一部として見なされます。

IPC オブジェクト作成および削除**:**

IPC オブジェクト作成には制限はありません。 プロセスが IPC オブジェクトを作成するときに、このオブジェクトはプロセス SL および TL を継承します。

IPC オブジェクトのアクセス・モードはオブジェクトを作成するシステム・コールで指定されなければなりません。

IPC オブジェクトを削除する場合、プロセスはこのオブジェクトへの `DAC OWNER`、 `MIC WRITE`、および `MAC WRITE` アクセス権限を持っていなければなりません。

Trusted Networking:

拡張セキュリティー・システムの拡張セキュリティー属性には、安全なネットワーキングの要件が必要です。 AIX Trusted Network は、U.S. DoD RFC1108 Revised Internet Protocol Security Option (RIPSO) や Commercial Internet Protocol Security Option (CIPSO) を含む、広く認識されているいくつかの安全なネットワーキングの標準をサポートします。

AIX には、IPv4 と IPv6 の両方に対する Trusted Network のサポートがあります。 他のトラステッド・システムと通信する場合、SL は CIPSO/RIPSO 標準に応じて IP オプションでカプセル化されます。 MAC 検査は、パケットで送受信される SL の IP 層で実行されます。 許可されるラベル範囲は、ネットワーク・ルールを使用して構成されます。 ネットワーク・ルールは、ホスト・ルールとインターフェース・ルールから構成されます。 AIX Trusted Network は、デフォルトのインターフェース・ルールのみをインストールします (構成済みインターフェースごとに 1 ルール)。より精細なフィルター処理を行えるようにホスト・ルールを構成することができます。 ホスト・ルールとインターフェース・ルールの両方を構成するには、`netrule` コマンドを使用できます。 `netrule` コマンドでサポートされる操作には、ルールの追加、削除、リスト、および照会があります。

Trusted Network サブシステムを初期化し、Trusted Network ルール・データベースを保守するには、`tninit` コマンドを使用できます。

root の無効化**:**

Trusted AIX システムでは、root ユーザー・アカウントが無効化されます。これにより、主に、すべての特権を持つ 1 ユーザーがシステムに及ぼす損害が最小限になります。

root ユーザーのすべてのタイプのシステム・ログインが無効になります。 root ユーザーのログインが許可されるのは、**su** コマンドを使用した場合のみです。 root が所有するプロセスには、特殊な特権は割り当てられません。 root 所有の setuid と非 setuid プログラムは、権限があるユーザーにより起動された場合には従来どおりに動作します。 権限がないユーザーの場合、DAC modebit または ACL が実行を許可する場合にプログラムが実行されますが、そのプログラムには特権が割り当てられません。このため、このプログラムは、権限がないユーザーにより実行された場合に特権操作を実行できない場合があります。したがって、アプリケーションで特権操作を実行する必要がある場合は、新規にインストールされたアプリケーションに適切な特権を割り当てる必要があります。

システム管理タスクを実行できるのは、情報システム・セキュリティー担当者 (ISSO)、システム管理者 (SA)、またはシステム担当者 (SO) のロールが割り当てられたユーザーです。 これらのロールを使用すると、ユーザーはシステム管理タスクを実行できます。

注: Trusted AIX のインストール時、root アカウントの **su** 属性が `false` に設定されます。 root アカウントから他の管理ユーザーまでのアクセスを許可するには、ISSO 権限ユーザーが **chuser** コマンドを使用してこの属性を `true` にリセットし、このアカウントにパスワードを割り当てる必要があります。

監査におけるラベルのサポート:

監査サブシステムの主な目的は、セキュリティー関連イベントのモニターと記録です。

監査サブシステムで提供される情報により、次のタイプの情報が記録されます。

- セキュリティー・ポリシーに違反する行為
- セキュリティー関連アクションの正常終了

監査サブシステムは、次の機能を備えています。

- 監査するイベントを判別する
- システムの実行中に監査をオンまたはオフにする
- 監査証跡ファイルを (情報の損失なしで) 途切れなく切り替える
- 監査情報を人間が理解できる形式に変換する
- 監査情報のサブセットを選択して処理する

監査サブシステムを設定する場合、ISSO は監査される内容、監査が実行される状況、監査を開始および停止する方法を理解しなければなりません。 監査の構成、開始および停止、管理、および検討の詳細については、『監査の概要』を参照してください。

監査サブシステムはその現在の状態を保持し、電源遮断、システム破壊、電源障害、またはその他の中断後にその状態で自動的に再開します。 監査サブシステムは、監査レコードを既存の監査ファイルに保管できない状況が発生した場合に、それ自体を自動的にシャットオフし、システムをシャットダウンし、監査ファイルを変更することができます。 ファイルシステムが指定レベルまで満杯になると、監査ファイルは自動的に切り替えられます。 ただし、壊滅的な電源障害が発生した場合は、少数の監査レコードが失われる可能性があります。

マルチレベル・ディレクトリーと分割ディレクトリー:

マルチレベル・ディレクトリーは、単一 SL ではなく SL 範囲が割り当てられた標準のディレクトリーです。 分割ディレクトリーは、単一のディレクトリーとしてユーザーに表示されます。 ただし、ユーザーに対して実際に表示されるファイルは、分割ディレクトリーの隠しサブディレクトリーにあります。

マルチレベル・ディレクトリー:

マルチレベル・ディレクトリーは、単一 SL ではなく SL 範囲が割り当てられた標準のディレクトリーです。

マルチレベル・ディレクトリーにあるファイル名を表示するには、プロセスがディレクトリーの最小 SL より高いセキュリティー・レベルで動作している必要があります。 実際のファイルを作成または削除するには、プロセスはマルチレベル・ディレクトリーの SL 範囲内で動作している必要があります。

マルチレベル・ディレクトリーの各ファイルにはその独自の SL があり、標準 MAC 制限により保護されます。 ただし、ディレクトリーへのアクセスのあるプロセスは、そのディレクトリー内のすべてのオブジェクトの名前を表示できます。 このため、プロセスはディレクトリーで MAC 読み取りおよび書き込みの能力がありますが、そのディレクトリー内のすべてのファイルの名前を表示できても、ディレクトリー内の一部のファイルには読み取りまたは書き込み、あるいはその両方を行うことができない場合があります。

分割ディレクトリー*:*

分割ディレクトリーは、単一のディレクトリーとしてユーザーに表示されます。ただし、ユーザーに対して実際に表示されるファイルは、分割ディレクトリーの隠しサブディレクトリーにあります。

マルチレベル・ディレクトリーはセキュリティー上のリスクをもたらします。 高セキュリティー・レベルで操作しているプロセスは、低セキュリティー・レベルでファイルを読み取ることができ、続いて同じ高セキュリティー・レベルでファイルを作成することができます。 MAC 機能により、低セキュリティー・プロセスによる新しいファイルの読み取りは防止されますが、低セキュリティー・プロセスによる新しいファイルの名前の表示は引き続き可能です。 高セキュリティー・プロセスにより、元の高セキュリティー・ファイルの内容に基づいて新しいファイル名が付けられた場合は、新しいファイル名を読み取ることによって、低セキュリティー・プロセスが高セキュリティー情報にアクセスできるようになります。

分割ディレクトリーが作成されて、プロセスがディレクトリーをアドレス指定すると、システムはアドレス指定プロセスと同じ SL をもつ隠しサブディレクトリーを作成します。 その後、プロセスがファイルを作成する場合は、そのファイルは実際は隠しサブディレクトリーの中に作成されます。 分割ディレクトリーにはこのような隠しサブディレクトリーがいくつかある可能性がありますが、分割ディレクトリーをアドレス指定するプロセスでは、アドレス指定プロセスと同じ SL をもつ隠しサブディレクトリー内のファイルのみが表示されます。 プロセスが分割サブディレクトリーの子ディレクトリーを作成すると、その子ディレクトリーは分割サブ-サブディレクトリーになります。

分割ディレクトリーは SL 範囲として SYSTEM_LOW から SYSTEM_HIGH を割り当てられます。このようにして、プロセスは分割ディレクトリーをアクセスできます。

aix.mls.pdir.mkdir 許可をもつユーザーは、**pdmkdir** コマンドを使用して分割ディレクトリーを作成することができます。 空の分割ディレクトリーは、**pdrmdir** コマンドを使用して除去することができます。 **pdset** コマンドを使用して、通常のディレクトリーを分割ディレクトリー・タイプに変更することができます。 分割ディレクトリーを通常のディレクトリーに変更するコマンドはありません。

分割ディレクトリー内では、1 つの分割ディレクトリーの中のファイルを、同じ分割ディレクトリーの中のより高位の SL を用いて、他のすべての既存の分割されたサブディレクトリーにリンクすることができます。 このことにより、分割ディレクトリーまたは同じ分割ディレクトリーの中のより高位の分割されたサブディレクトリーへのアクセスを行うすべてのプロセスによって、分割ディレクトリー内のファイルにアクセスすることができます。 このファイルのリンクには、**pdlink** コマンドを使用することができます。

分割ディレクトリーのアクセス・モード*:*

プロセスには作成時に 2 つのモード (実モードまたは仮想モード) のいずれかが割り当てられます。 このモードにより、プロセスが分割ディレクトリーを表示する方法が決まります。

実モード・プロセスは、分割ディレクトリーを標準マルチレベル・ディレクトリーとして処理します。 分割サブディレクトリーには、通常の DAC、MIC、および MAC 制限に従って標準ディレクトリーとしてアクセスできます。 実モード・プロセスは、DAC、MIC、および MAC 制限に従って、分割ディレクトリーに入り、すべてのサブディレクトリーを表示できます。

仮想モード・プロセスは、分割ディレクトリーには入りませんが、その代わりに、最大と最小 SL が両方ともプロセスの有効 SL と同じである分割サブディレクトリーにリダイレクトされます。

実モード・プロセスは、**pdmode** コマンド (`pdmode ls` など) を使用して仮想モードでコマンドを実行できます。 同様に、仮想モード・プロセスは、**pdmode** コマンド (`pdmode -r ls` など) を使用して実モードでコマンドを実行できます。 ただし、この場合は、`aix.mls.pdir.mode` 権限が必要です。この権限がある場合、`pdmode -r sh` を実行すると、仮想モードで実行しているシェルから実モードで実行しているシェルに切り替えることもできます。実モードで実行中に、仮想モードでプログラムを起動する場合は、権限は必要ありません。

ディレクトリー・タイプの表示および変更:

`secflags` 属性の一部としてディレクトリー・タイプを表示するには、**lstxattr** コマンドを使用できます。`FSF_PDIR` は分割ディレクトリーを示し、`FSF_PSDIR` は分割サブディレクトリーを示し、`FSF_PSSDIR` は分割サブサブディレクトリーを示します。通常のディレクトリー・タイプを分割ディレクトリー・タイプに変更するには、**pdset** コマンドを使用します。

Trusted AIX 管理

Trusted AIX システムの管理には、Trusted AIX に固有のいくつかの要因が関係します。

Trusted AIX インストール

Trusted AIX は、インストール・メニューの「**セキュリティー・モデル**」オプションを使用して、基本オペレーティング・システム・インストールのみで使用可能です。

Trusted AIX の移行はサポートされていません。保存インストールの場合、ファイルシステムは JFS2 にする必要があります。 プロンプトを出さないネットワーク・インストールで、デフォルト管理ユーザーに関連付けるパスワードについては、表 37 を参照してください。

表 37. デフォルト管理ユーザーのパスワード

ユーザー	パスワード
isso	isso
sa	sa
so	so

実行モード

システム構成と保守および日次操作に対して、構成モードと操作モードの 2 つの実行モードを使用できます。

システムはブートすると、最初に構成モードで稼働します。 初期設定が完了すると、実行モードは操作モードに変更されます。

構成モードは、システムの保守およびリカバリーに使用されます。 システムが単一ユーザー・モードでブートされると、システムは最小限に構成され、ネットワーキングは使用不可にされます。 構成モードは、システムの重大なセキュリティー関連部分の管理に使用されます。

操作モードは、標準のシステム操作モードです。 デフォルトの実行レベルに入るために必要なすべてのタスクが完了すると、システムはこのモードに変更されます。

システムの実行モードを表示するには **getrunmode** コマンドを使用し、システムの実行モードを変更するには **setrunmode** コマンドを使用します。

カーネル・セキュリティー・フラグ

カーネル・セキュリティー・フラグは、特定のセキュリティー・フィーチャー (例えば、ラベル検査の強制)、読み取り操作での保全性ラベルの検査、およびその他の目的を使用可能または使用不可にする場合に使用されます。

カーネルはセキュリティー検査を強制する前にカーネル・セキュリティー・フラグについて検査します。これらのフラグは Trusted AIX が使用可能な場合のみサポートされます。 ユーザー・スペースでは、これらのフラグは ODM データベースに保管されます。 システムの実行モードに応じて、カーネルは対応するカーネル・セキュリティー・フラグについて検査します。

表 *38.* カーネル・セキュリティー・フラグおよびデフォルト値

カーネル・セキュリティー・フラグ	使用可能の場合	使用不可の場合	操作モードのデフォルト	構成モードのデフォルト
tnet_enabled	トラステッド・ネットワーク機能は使用可能	トラステッド・ネットワーク機能は構成できないか、または使用できない	使用不可	使用不可
tl_write_enforced	書き込み、削除、および名前変更操作で MIC 強制操作を行う	構成は TL は書き込みチェックには使用されないように設定される	使用可能	使用可能
tl_read_enforced	読み取り操作で MIC 強制操作を行う	構成は TL は読み取りチェックには使用されないように設定される	使用不可	使用不可
sl_enforced	MAC 強制操作を行う	構成は SL はアクセス制御には使用されないように設定される	使用可能	使用不可
trustedlib_enabled	ファイルシステム・オブジェクトでの FSF_TLIB フラグは順守される	FSF_TLIB フラグは順守されない	使用不可	使用不可

カーネル・パラメーターの設定

Trusted AIX カーネルは、サイト・ポリシーに必要なセキュリティー制約を実行するように構成することができます。

getsecconf コマンドを使用して表示されたセキュリティー構成は、**setsecconf** コマンドを使用して変更できます。 構成可能なカーネル・パラメーターは次のとおりです。

- 機密ラベルの実行
- 保全性読み取りの実行
- 保全性書き込みの実行
- Trusted Network
- トラステッド・ライブラリー

これらのパラメーターを構成できるのは、システムが構成実行モードにある場合のみです。

/etc/security/enc/LabelEncodings ファイルのカスタマイズ

システムのラベルは `/etc/security/enc/LabelEncodings` ファイルで定義され、各サイトに合わせてカスタマイズすることができます。

ラベルのカスタマイズは、Trusted AIX のインストール後に行うことができます。

Trusted AIX システムには、システム上の他のすべての機密ラベルより下位に定義されている SYSTEM LOW SL (SLSL) および他のすべての機密ラベルより上位に定義されている SYSTEM HIGH SL (SHSL) があります。同様に、SYSTEM LOW TL (SLTL) はシステム上の他のすべての保全性ラベルより下位に定義され、SYSTEM HIGH TL (SHTL) は他のすべての保全性ラベルより上位に定義されています。 これらの定義は、/etc/security/enc/LabelEncodings ファイルに定義されているように最高位および最低位の SL および TL の値をとります。

Trusted AIX システムがブートされる時に、システム・ラベルが /etc/security/enc/LabelEncodings ファイルからカーネルにダウンロードされます。 **setsyslab** コマンドを使用してラベルをカーネルにダウンロードすることもできます。 カーネルに定義されているシステム・ラベルは、**getsyslab** コマンドを用いてリストすることができます。システムのリブートは、/etc/security/enc/LabelEncodings ファイルを変更した後に行うことをお勧めします。

コメントは、/etc/security/enc/LabelEncodings ファイルの中のキーワードを開始できる任意の位置に記入することができます。 コメントは、先頭の * から始まり、行末まで続きます。

/etc/security/enc/LabelEncodings ファイルには、バージョン情報および以下の必須セクションが含まれます。それぞれのセクションは、下記のいずれかのセクション・キーワードで始めて、その後にコロン (**:**) を付けてください。

- 種別
- 情報ラベル
- 機密ラベル
- 認可
- チャネル
- プリンター・バナー
- 認定範囲

/etc/security/enc/LabelEncodings ファイルは、VERSION エントリーから始まります。 このエントリーは文字列で、空白文字を入れることができます。

以下のキーワードはそれぞれセクションに記入することができます。 これらのキーワードはセミコロン (**;**) で終わります。

name=*name*
種別またはコンパートメントのフルネームを定義するキーワード。

sname=*name*
省略名を定義するキーワード。 オプションです。

aname=*name*
種別の代替キーワード。 オプションです。

value=*value*
種別またはコンパートメントの内部整数値を指定するキーワード。

compartments=*bit*
上記のワードがラベル中に存在する場合に 0 または 1 としなければならないコンパートメント・ビットを指定するキーワード。

ラベル・エンコード・フォーマットに対する Trusted AIX の機能拡張

ラベル・エンコード方式では、Defense Intelligence Agency Document DDS-2600-6216-93 に規定されているように、保全性ラベルをサポートしていません。

デフォルトでは、機密ラベルが保全性ラベルとして使用されます。 Trusted AIX では、機密ラベル・セクションとは異なることがあるオプションの保全性ラベル・セクションをサポートしています。 これにより、機密ラベルおよび保全性ラベルにさまざまな種別の名前および値を使用できる柔軟性が与えられます。例えば次のように、機密ラベルに SL の接頭部、保全性ラベルに TL の接頭部を付けることができます。

表 *39.* 機密ラベルの種別の名前および値

name	sname	value
name= SL IMPLEMENTATION LOW	sname= SL_IMPL_LO	value= 0
name= SL UNCLASSIFIED	sname= SL_U	value= 20
name= SL PUBLIC	sname= SL_PUB	value= 40
name= SL SENSITIVE	sname= SL_SEN	value= 60
name= SL RESTRICTED	sname= SL_RES	value= 80
name= SL CONFIDENTIAL	sname= SL_CON	value= 100
name= SL SECRET	sname= SL_SEC	value= 120
name= SL TOP SECRET	sname= SL_TS	value= 140

表 *40.* 保全性ラベルの種別の名前および値

name	sname	value
name= TL IMPLEMENTATION LOW	sname= TL_IMPL_LO	value= 0
name= TL UNCLASSIFIED	sname= TL_U	value= 20
name= TL PUBLIC	sname= TL_PUB	value= 40
name= TL SENSITIVE	sname= TL_SEN	value= 60
name= TL RESTRICTED	sname= TL_RES	value= 80
name= TL CONFIDENTIAL	sname= TL_CON	value= 100
name= TL SECRET	sname= TL_SEC	value= 120
name= TL TOP SECRET	sname= TL_TS	value= 140

次のルールが、保全性ラベル・セクションに適用されます。

- 「INTEGRITY LABELS」セクションは「NAME INFORMATION LABELS」セクションの後にだけ追加してください。 管理者がオプションの「NAME INFORMATION LABELS」セクションを定義していない場合は、「INTEGRITY LABELS」セクションは「ACCREDITATION RANGE」セクションの次に追加してください。
- ラベル・エンコード・ファイル内に存在できる「INTEGRITY LABELS」セクションは 1 つだけです。同じセクションがオブジェクトとサブジェクトの両方に適用されます。
- 新しい「INTEGRITY LABELS」セクションは任意指定のセクションです。 このセクションが存在しない場合は、必須の「CLASSIFICATIONS」セクションで指定されている種別を使用しなければなりません。
- 「INTEGRITY LABELS」セクションは「CLASSIFICATIONS」セクションに類似している場合があります。 さらに、**"name="**、**"sname="**、**"aname="**、および **"value="** のキーワードが入ります。「CLASSIFICATIONS」セクションの一部である **"initial compartments="** および **"initial markings="** のキーワードは、「INTEGRITY LABELS」セクションでは無効になります。

- **"value="** のデータ範囲は、「CLASSIFICATIONS」セクションのデータ範囲 (最小 0 から最大 32,000) と同じになります。

システムの始動

システム・セキュリティーは、システムの始動シーケンス時に自動的に起動します。 始動シーケンス時に表示されるセキュリティー・パラメーターがシステムに対して正しいことを検証する必要があります。

構成開始モード:

構成モードは、システムの保守およびリカバリーに使用されます。

システムが単一ユーザー・モードでブートされると、システムは最小限に構成され、ネットワーキングは使用不可にされます。

始動操作モード:

操作モードは、日次操作に対して使用されます。

通常、システムはマルチユーザー・モードに直接ブートされます。 ブート権限プログラムが有効なユーザー名とパスワードを受け取ると、システムは操作モードに入り、コンソール・ログイン認証画面が表示され、有効なユーザーがログインできるようになります。

機密ラベル、任意アクセス制御、必須アクセス制御、特権検査、識別と認証、権限などのセキュリティー・メカニズムは、関連するセキュリティー構成フラグで指定されたとおり、構成モードと操作モードの両方でアクティブになります。 詳しくは、『**getsecconf** コマンド』を参照してください。

期待されるシステム機能をすべて使用できるように、システムは操作モードでのみ操作することをお勧めします。

ブート・プロセス:

新規ブート・スクリプトが Trusted AIX システムの `/etc/inittab` ファイルに追加されました。 新規ブート・スクリプトは `rc.mls.boot`、`rc.mls.net`、および `rc.mls` であり、この順序で実行されます。

`rc.mls.boot` スクリプトで実行されるステップは、以下のとおりです。

1. 対話式の保全性検査の実行では、それぞれの違いをどのように取り扱うかについて、プロンプトを出してユーザーに情報を求めます (**trustchk** コマンドを使用)。
2. 構成モードのカーネル・セキュリティー・フラグを設定します (**setsecconf** コマンドを使用)
3. システム・ラベルを設定します (最小および最大機密ラベルと保全性ラベル)
4. 構成モードのカーネル・セキュリティー・フラグは画面に表示されます

`rc.mls.net` スクリプトで実行されるステップは、以下のとおりです。

1. Trusted AIX サブシステムを初期化します。
2. `/etc/security/rules.int` ファイルがある場合は、カーネルにルール・データベースをロードします。

`rc.mls` スクリプトで実行されるステップは、以下のとおりです。

1. Trusted AIX サブシステムを初期化します。
2. `/etc/security/rules.int` ファイルがある場合は、カーネルにルール・データベースをロードします。

注: ブート・スクリプトに何らかの変更を行うと、システム誤動作を起こすことがあります。

システム始動のカスタマイズ:

お勧めはできませんが、システム始動時のブート認証およびシステム保全性検査を使用不可にすることができます。

ブート認証およびシステム保全性検査が使用不可でない限り、オペレーターは、システムを始動するために物理的にシステム・コンソールの近くにいなければなりません。

BOOT 認証の使用不可化:

BOOT 認証は **rmitab bootauth** コマンドを実行するか、または SMIT メニューを使用して使用不可にすることができます。

システム保全性検査の使用不可化:

自動システム・ブート保全性検査を使用不可にするには、**rc.mls.boot** スクリプトから `trustchk` 行を除去します。

システムのシャットダウン

システムのシャットダウンは特権操作であり、`aix.system.boot.shutdown` 権限により保護されます。

システムをシャットダウンできるのは、`SO` ロールまたはこの権限がある他のロールを持つユーザーです。

トラステッド・リカバリー

システムは不明な状態で電源オフになることがあります。この原因は、電源異常、不慮の電源オフ、またはハードウェア障害によるものである可能性があります。Trusted AIX では、特別にリブート手順を実行しなくても、このような状況からリカバリーすることができます。

システムがリブートすると、システムがどのように電源オフになったかに関係なく、すべての保護メカニズムがアクティブになります。システム始動プロシージャー時、ユーザーがログオンする前に、すべてのファイルシステムの損傷が自動的に検査されます。始動スクリプトは **fsck** コマンドを実行して、権限がないユーザーが損傷したファイルや情報漏えいしたファイルにアクセスできないようにします。

trustchk コマンドは、ファイルまたはディレクトリーのセキュリティー属性にある不整合を報告し、ユーザーにこれらの属性の修復についてのプロンプトを対話式に出します。 ファイルシステムの保全性が危険にさらされた可能性がある場合は必ず **trustchk** コマンドを実行してください。 詳しくは、『**trustchk** コマンド』を参照してください。

ログイン

Trusted AIX ユーザーは、システムにログインするために、適切に割り当てられた機密認可と保全性認可を持っている必要があります。

ユーザーの認可は、`/etc/security/user` ファイルでユーザー属性として定義されます。 `minsl` と `maxsl` 属性は、ユーザーの機密認可を定義します。 `mintl` と `maxtl` 属性は、ユーザーの保全性認可を定義します。 `defsl` と `deftl` 属性は、ログイン時のユーザーの有効機密レベルと保全性レベルを定義します。

ユーザーの認可属性は、`chuser` と `chsec` コマンドで変更し、`lsuser` と `lssec` コマンドでリストできます。

ユーザーは自分のラベルをリストすることはできますが、それらを変更することはできません。 他のユーザーの認可レベルをリストするには、ユーザーは `aix.mls.clear.read` 権限を持っている必要があります。認可を変更するには、ユーザーは `aix.mls.clear.write` 権限を持っている必要があります。

ログインするには、次の支配ルールがすべて真でなければなりません。

- `minsl` 値が `defsl` 値によって支配されなければならない
- `defs` 値が `maxsl` 値によって支配されなければならない
- `mintl` 値が `deftl` 値によって支配されなければならない
- `deftl` 値が `maxtl` 値によって支配されなければならない

login コマンドの **–e** および **–t** オプションを使用したログインで、望ましい有効機密レベルと保全性レベルを指定できます。詳しくは、「**login** コマンド」を参照してください。

システムの認定範囲内にない機密レベルでログインするには、`aix.mls.label.outsideaccred` 権限を持っている必要があります。

Trusted AIX では、システム・ユーザー (128 より小さい uid のユーザー) のログインを許可しません。

ログインの失敗の原因

ログイン試行は、さまざまな理由で失敗する可能性があります。

以下のいずれかに該当する場合、ログイン試行は失敗します。

- 無効なログイン ID が入力された
- 無効なパスワードが入力された
- このアカウントに関してそれまでに失敗したログイン試行回数がシステム制限を越えたために、アカウントがロックされた
- ポートに対してそれまでに失敗したログイン試行回数がシステム制限を越えたために、ログイン・ポートがロックされた
- ログイン ID に有効な認可がない
- 指定されたラベル (ラベルが指定されていない場合は、ログイン ID のデフォルトの機密ラベルまたは保全性ラベル) が有効ではない、ログイン ID の認可内にない、ログイン・デバイスの認可内にない、またはシステムの認定範囲内にない
- ユーザーがログイン・シェル・プログラムのパス名に対する DAC アクセスを持っていないか、ユーザー・アカウントがログイン・シェル・プログラムに対する DAC 実行アクセスを持っていない
- ユーザーがログイン・シェル・プログラムのパス名に対する MAC または MIC 読み取りアクセスを持っていないか、ログイン・シェル・プログラムに対する MAC または MIC 読み取りアクセスを持っていない
- ログイン ID の uid が 128 より小さい

su コマンドを使用したユーザーの切り替え

Trusted AIX システムでは、**-** オプションを使用した **su** コマンドが呼び出される際、現行ユーザーの認可は新規ユーザーの認可レベルを支配しなければなりません。

機密ラベルと保全性ラベルに対して、次の条件を満たしている必要があります。

- 現行ユーザーの最大認可は、新規ユーザーの最大認可より上位でなければなりません。
- 新規ユーザーの最小認可は、現行ユーザーの最小認可より上位でなければなりません。
- 現行ユーザーの有効な認可は、新規ユーザーの最大認可により下位であり、新規ユーザーの最小認可より上位でなければなりません。

ユーザーのセキュリティー責任

ユーザーが認識、理解、および順守しなければならない特定の責任があります。 ユーザーはパスワードを秘密に保持し、ユーザー状況の変更を報告し、疑わしいセキュリティー違反を報告し、その他にもいろいろと行う必要があります。

パスワード

パスワードは記憶する必要があり、メディアには書き込まないでください。 パスワードが他人に取得されると、システム上の情報のセキュリティーが危険にさらされる可能性があります。

パスワード・セキュリティーに対する最大の脅威は、パスワードの漏えいです。 パスワードを得た可能性のあるユーザーによる不正攻撃からアカウントを保護する最も簡単な方法は、パスワードを定期的に変更することです。 各パスワードの存続期間の間に危険にさらされる可能性を少なくするために、パスワードを頻繁に変更する必要があります。 1 つのパスワードを長く使い続けるほど、危険にさらされる可能性が高くなります。

ユーザーが自分のパスワードを選択できる場合は、新規パスワードは長さを 6 文字以上にして、少なくとも 2 つの英字と 1 つの数字を含める必要があります。 パスワードにはユーザーの個人的または職業上の情報 (友人、ユーザーの名前、ペットの名前、仕事の名称など) を指定しないでください。また、辞書にあるような一般的な用語も使用しないでください。パスワードを推測する際には、辞書を 1 つ以上チェックしたり、ユーザーの名前、子供やペットの名前、誕生日などの大量の個人項目を調べることが多くあります。

ISSO はパスワードに有限の存続期間を指定できます。 パスワードの有効期限が切れている状態でユーザーがログインしようとすると、そのパスワードは変更する必要があり、パスワードを変更しない限りユーザーはログインできないことがユーザーに通知されます。 ユーザー・パスワードは、指定されたパスワードの存続期間よりも頻繁に変更することをお勧めします。 ユーザーのパスワードが漏えいした疑いがある場合は、直ちにパスワードを変更してください。

システムの無人放置

ユーザーがアクティブ・セッションにログインしている場合は、システムを無人で放置してはなりません。少しの間でもマシンから離れる際は、その前にシステムをログオフすることを強くお勧めします。

セキュア・システムの管理

セキュア・コンピューター・システムの管理には、セキュリティー・ポリシーの作成および実施ならびに通常のシステム・モニターが含まれます。

次のリストは、ご使用のサイトの保護機能管理ポリシーを開発するための開始点の役目を果たします。

- システムの認定範囲における最大セキュリティー・レベルは、システムが位置するサイトの最大セキュリティー・レベルより大きくてはいけません。
- システム・ハードウェアは安全な場所になければなりません。 最も安全な場所は、通常、屋内の 1 階以外の部屋です。
- システム・ハードウェアへの物理的アクセスの制限、監視、および文書化を行わなければなりません。
- システムのバックアップおよびアーカイブ・メディアは、システム・ハードウェアの設置場所とは離れた安全な場所に保管しなければなりません。 この場所への物理的アクセスは、システム・ハードウェアへのアクセスと同じ方法で制限する必要があります。
- 操作資料および管理文書は、それらの内容に関する知識を習得する必要がある方のみが利用できるようにしておかなければなりません。

- システムのリブート、電源障害、およびシャットダウンは記録しておく必要があります。 ファイルシステムの損傷は文書化し、影響を受けるすべてのファイルについてセキュリティー・ポリシー違反の可能性がないか分析しなければなりません。
- 新しいプログラムのインストールは、インポートであれ作成であれ、制限およびモニターがなされなければなりません。 新しいプログラムは、実行前に注意深く調査しテストする必要があります。
- すべてのシステム・ソフトウェアの異常なまたは予期せぬ動作については、文書化して報告し、動作の原因を判別しなければなりません。
- 可能な限り、少なくとも 2 人の人間がシステムを管理する必要があります。 1 人は `isso` ロールをもち、もう 1 人は `sa` ロールをもっている必要があります。
- `PV_ROOT` 特権は、使用すべきではありません。 システムを管理するには、ISSO、SA、または SO ユーザーが特権プログラムを実行することで十分なはずです。
- 監査情報はログに収集し、定期的に検討する必要があります。不規則なイベントまたは異常なイベントについては書き留めて、その原因を調べる必要があります。
- `isso`、`sa`、および `so` ロールによるログインの回数は最小限にしなければなりません。
- setuid および setgid プログラムの数は最小限にし、保護されたサブシステムでのみ使用する必要があります。
- 新しいプログラムに割り当てられている特権は、既存のプログラムに割り当てられている特権を検討して、最小限に抑える必要があります。
- ファイルおよびディレクトリーのセキュリティー属性は、**trustchk** コマンドによって定期的に確認しなければなりません。
- すべてのパスワードには、少なくとも 8 文字を指定しなければなりません。 このことは、ISSO ユーザーが定期的に確認するようにしてください。
- すべてのユーザーが有効なデフォルトのログイン・シェルをもたなければなりません。 このことは、SA ユーザーが定期的に確認するようにしてください。
- 通常のユーザーのユーザー ID は、システム ID であってはなりません。 このことは、SA ユーザーが定期的に確認するようにしてください。 システム ID は、128 より小さい uid をもつ ID です。

システム構成:

システムを適切に構成するために、ISSO および SA は特定のステップを実行する必要があります。 ISSO は主にセキュリティー管理に対して責任があり、SA は主に日次管理に対して責任があります。

ISSO は以下の作業を実行します。

- 割り当て可能デバイスに対するシステムの監査、アカウンティング、およびセキュリティーを含む、基本的なセキュリティー機能をインストールし、構成します。
- `/etc/rc.mls` ファイルと `/etc/rc.mls.boot` ファイルにあるシステム始動スクリプトを編集して、サイト・セキュリティー・ポリシーに適合させます。

  注: システム始動スクリプトに対する変更は評価構成の一部ではなく、システムの認定前に対応する必要があります。
- システム全体のログイン・パラメーターを構成します。
- システム全体のパスワード・パラメーターを構成します。
- TTY デバイスの SL 範囲を構成して、ユーザーが TTY ポートに指定された SL 範囲にログインできるようにします。 詳しくは、『**chsec** コマンド』を参照してください。

- テープ・ドライブとフロッピー・ディスク・ドライブのシステム・デバイス SL を構成します。 詳しくは、『**setsecattr** コマンド』を参照してください。
- システムのサイト構成可能セキュリティー機能を構成します。

  注**:** 構成可能セキュリティー機能に対する変更は評価構成の一部ではなく、システムの認定前に対応する必要があります。 デフォルトの構成設定値を変更すると、システムが安全性の低いモードで稼働する可能性があります。
- トラステッド・ブートとトラステッド・リカバリーのためにトラステッド・セキュリティー・データベースを構成します。 詳しくは、『**trustchk** コマンド』を参照してください。
- システムでユーザー・グループを構成します。

プリンターを構成するには、ISSO と SA が連携して作業します。 SA はシステムのプリンターを構成し、ISSO はプリンターの SL 範囲を構成します。

ネットワーク構成**:**

ISSO は主にネットワーク・セキュリティーに対して責任があり、SA は主に日次ネットワーク管理に対して責任があります。 ISSO と SA は、連携してネットワークを適切に構成します。

ネットワーク・セキュリティーは、Trusted AIX のインストール時にデフォルトの設定で構成されます。これは機密ラベルをネットワーク上の他の Trusted AIX ホストに受け渡すこともできます。 ISSO はシステムで提供される基本ネットワーク機能をインストールして構成します。 ISSO はネットワーク・テーブルを構成し、データベースを保存するために **tninit** コマンドを実行します。

ネットワーク・アクセス*:*

ネットワークを介して非 Trusted AIX システムに接続しているか、トラステッド・ネットワーキング機能を使用していない Trusted AIX システムに接続する場合は、一部のセキュリティー属性が非 Trusted AIX システムによって送信されない場合があります。この場合、Trusted AIX システムはデフォルトのセキュリティー・メカニズムを適用します。 デフォルトのセキュリティー・メカニズムは、システム管理者が設定します。

ユーザー・アカウントの構成**:**

システムでユーザー・アカウントを構成するには、ISSO と SA が連携して作業します。 ISSO は主にセキュリティー関連のユーザー属性の管理に対して責任があり、SA は主にその他のユーザー属性に対して責任があります。

ISSO はユーザーごとに以下の作業を実行します。
- 認可の構成。 詳しくは、『**chsec** コマンド』および『**chuser** コマンド』を参照してください。
- ロールと権限の構成
- ユーザー・グループの構成
- ホーム・ディレクトリーの認可レベルの設定。 詳しくは、『**settxattr** コマンド』を参照してください。
- パスワードの設定
- 監査マスクの設定

SA は以下の作業を実行します。
- ユーザー・アカウントの構成
- セキュリティー属性が必要な新規ユーザー・アカウントの ISSO への通知

ファイルシステム構成**:**

ほとんどのファイルシステムは Trusted AIX でサポートされますが、ファイルシステム・オブジェクト上で拡張属性に関連した Trusted AIX セキュリティーのサポートは EAv2 による JFS2 でのみ使用可能です。

EAv1 ファイルシステムによる JFS2 は、Trusted AIX システムにマウントされたときに、 EAv2 へ変換されます。 これらの JFS2 ファイルシステム上のファイルはセキュリティー属性を持っていません。 システムはデフォルトの SYSTEM_LOW 属性を使用して、これらのファイルをアクセスします。 セキュリティー属性については、**settxattr** コマンドを使用してファイルに設定できます。

ネットワーク環境では、あるシステムのディレクトリーを共有としてマークを付けることができます。これにより、そのディレクトリーは、ローカル・ディスク・パーティションにあるファイルシステムのルート・ディレクトリーであるかのように、ネットワーク上のその他のシステムでマウントしてアクセスできるようになります。

ファイルシステムはマルチレベル・ファイルシステム (MLFS) か、または単一レベル・ファイル・システム (SLFS) のいずれかです。 MLFS での各ファイル・オブジェクトは、それ自体のラベルを持っており、一方、SLFS のすべてのオブジェクトはマウント・ポイントとして同じラベルを持っています。 SLFS はマルチレベル・ディレクトリーおよび分割ディレクトリーをサポートしません。

ファイルシステム・アクセス*:*

プロセスがファイルシステム・オブジェクトへのアクセスを試みるときに、システムは各パス名コンポーネントへのアクセス権限を検査します。

プロセスがパス名の全ディレクトリーへの検索アクセス権限を持っていない場合、このプロセスはオブジェクトをアクセスできません。 相対パス名が使用される場合は、現行ディレクトリーへのアクセスについて、現行ディレクトリーがパス名の先頭にピリオド (.) を使用して、明示的に参照されているかどうかを検査されます。

Trusted Network 管理**:**

Trusted Network を管理する場合、多くの考慮事項があります。これには、構成と構成データベース、netrule 構文とルール規格、Trusted Network フラグ、および RIPSO/CIPSO オプションがあります。

デフォルト構成の警告*:*

AIX Trusted Network のネットワーキング機能は、考えられる限り望ましい構成を可能にするように注意深く設計されています。 したがって、AIX Trusted Network を理解しないで構成のデフォルト値を変更するには、危険があります。

マシンを不適切に構成することにより、自動的にダウングレード、アップグレード、またはセキュリティー情報も一緒に除去される可能性があります。 このために、AIX Trusted Network に精通されていない場合は、ネットワーキング・テーブルのデフォルト値を変更しないでください。

AIX Trusted Network 構成データベース*:*

ブート時のネットワーク構成は rules.host ファイルおよび rules.int ファイルに設定します。

デフォルトの Trusted AIX インストールの場合は、ホスト・ルールまたはルール・ファイルがありません。 新規または更新ルールをファイルに保存するには、**-u** フラグを指定した **netrule** コマンドを使用し

ます。 ファイルは **tninit** コマンドで取り扱うことができるバイナリー・データベースです。 **tninit** コマンドを使用する場合、ユーザーには aix.mls.network.init 権限が必要です。

***AIX Trusted Network** ルール・データベースの表示:*

AIX Trusted Network ルール・データベース・セットの内容は、 **tninit** コマンドに **disp** アクションを指定して表示できます。

.host および **.int** の拡張子を *filename* に付加して、ホスト・ルール・データベースのファイル名およびインターフェース・ルール・データベースのファイル名を生成するには、以下のコマンドを入力します。 両方のファイルの内容が人間に読み取り可能なフォームで標準出力ストリームへ送信されます。

```
tninit disp filename
```

ブート・デフォルト構成を表示するには、以下のコマンドを入力します。

```
tninit disp /etc/security/rules
```

***AIX Trusted Network** ルール・データベースのロード:*

tninit コマンドは、AIX Trusted Network ルール・データベースのセットを読み取り、それらをカーネルにロードしてアクティブ・セットにします。ホストとインターフェース認定テーブルのファイル名は、**tninit disp** アクションと同じ方式で指定されます。

オプションの **-m** フラグは、システムが既存のホスト・ルールを維持することを指定します。 **-m** フラグを指定しないと、新規アクティブ・セットがロードされる前にすべての既存ホスト・ルールが除去されます。 **-m** フラグを指定すると、既存と新規のホスト・ルール・セットが集約され、競合がある場合は、新規ルールで既存ルールが上書きされます。 インターフェース・ルールはすべて **-m** フラグの指定の有無に関係なく上書きされます。

次のコマンドは、既存ルール・セットを維持する一方で新規ルールをロードします。

```
tninit -m load /dir/dir/filename
```

このコマンドは、*filename* パラメーターで指定されたファイルを使用し、**.host** と **.int** 拡張子を追加してデータベースを構成する 2 つのファイルを作成します。

***AIX Trusted Network** ルール・データベースの保存:*

ルール・データベースのロードと保存には、同様のセマンティクスが使用されます。

指定されたファイル名には **.int** と **.host** が追加され、データベースの保管に使用される 2 つのファイルが作成されます。 **tninit** コマンドの保存アクションにより、カーネルで現在アクティブなすべてのルールが保存されます。

デフォルトのルール・セットを作成するには、**netrule** コマンドを使用して目的のサイト・セキュリティー・ポリシーに適合させるようにカーネル・ルールを調整し、**tninit** コマンドを実行します。 次のコマンドは、/etc/security/rules.int ファイルと /etc/security/rules.host ファイルを作成します。

```
tninit save /etc/security/rules
```

***AIX Trusted Network** カーネルの構成:*

aix.mls.network.config 権限があれば、**netrule** コマンドを使用して、サイトのセキュリティー・ポリシーに適合するようにカーネルの AIX Trusted Network ルール・セットを完全に構成することができます。

netrule コマンドは、カーネルでホストおよびネットワーク・インターフェース・ルールの両方を取り扱う場合に使用します。 詳細情報については、『**netrule** コマンド』を参照してください。

システムの各インターフェースには、関連付けられたルールがあります。インターフェース・ルールを削除すると、そのインターフェースはデフォルトの状態に戻ります。 インターフェースをもう 1 つ追加すると、新しいインターフェース・ルールで現在のルールを上書きします。 インターフェース・ルールに『default』 を指定して照会すると、デフォルト・インターフェース・ルールが表示されます。 例: `# netrule iq default`

netrule 構文*:*

netrule コマンドには、ホストとインターフェースの構文ルールがあります。

netrule コマンドは、ホストに対して使用される場合に次の構文ルールを持っています。

netrule h l [**i** | **o** | **io**]

netrule h q { **i** | **o** } *src_host_rule_specification dst_host_rule_rule_specification*

netrule h - [{ **i** | **o** } [**u**] [*src_host_rule_specification dst_host_rule_specification*]

netrule h + { **i** | **o** } [**u**] *src_host_rule_specification dst_host_rule_specification* [*flags*] [*RIPSO/CIPSO_options*] *security*

netrule コマンドは、インターフェースに対して使用される場合に次の構文ルールを持っています。

netrule i l

netrule i q *interface*

netrule i + [**u**] *interface* [*flags*] [*RIPSO/CIPSO_options*] *security*

最初のエレメント `h` または `i` は、ホストまたはネットワーク・インターフェースの操作を示します。

目的のアクションを次にリストします。 これらの 4 つの異なるアクションが可能です。

l すべてのルールをリストする

q 特定のルールを照会する

- ホスト・ルールを除去するか、インターフェース・ルールをそのデフォルト状態に戻す

+ ルールを追加または上書きする

ホスト・ルールの 3 番目のエレメントは、ルール・タイプを識別します。 ホスト・ルールの場合、着信ルールと発信ルールには違いがあります。 着信ルールはすべての着信パケットに適用され、発信ルールはすべての発信パケットに適用されます。`i` は着信ルールを示し、`o` は発信ルールを示します。適用される場合、`io` または「なし」は着信と発信の両方のルールを示します。 ホストまたはインターフェースのルールを追加または除去するときに最後のエレメント `u` を指定すると、ホストまたはインターフェースのルールが正常に追加または除去された後に、`/etc/security/rules.host` と `/etc/security/rules.int` のファイルが更新されます。

AIX Trusted Network のルール仕様*:*

インターフェース・ルールでは、ネットワーク・インターフェースの名前を入力する必要があります。 ホスト・ルールの柔軟性はかなり高いため、より複雑なルール仕様が必要です。

インターフェースを指定するには、ルールを適用するネットワーク・インターフェースの名前を入力してください。 ネットワーク・インターフェース名は `en0` のような名前です。 **ifconfig -a** コマンドを使用して、ネットワーク・インターフェース名を表示することができます。 名前のみを用いて、特定のインターフェースを指定しなければなりません。 ポート、プロトコル、またはサブネット・マスクを指定することはできません。

ホスト・ルールにはさらに複雑なルール仕様が必要です。 AIX Trusted Network システムでは、最も具体的な適用可能なルールが使用されます。 例えば、24 のマスクを指定したホスト・ルールをサブネット上のすべてのホストに適用するようにサイト・ポリシーを構成することができますが、ネット上の 1 つのホストには、より具体的なルールを適用することができ、このホストはより具体的なルールを使用します。 別のより具体的なルールを、このホスト上の 1 つの特定の TCP ポートに適用することもできます。 AIX Trusted Network 構成の柔軟性により、アプリケーションに必要ないかなるサイト・セキュリティー・ポリシーも実施することができます。 正確な構文は次のとおりです。

source_host [**/***mask*] [**=** *proto*] [**:***start_port_range* [**:***end_port_range*]]

destination_host [**/***mask*] [**=** *proto*] [**:***start_port_range* [**:***end_port_range*]]

source_host
ソース・ホストのホスト名、IPv4 アドレス、または IPv6 アドレス。

destination_host
宛先ホストのホスト名、IPv4 アドレス、または IPv6 アドレス。

mask サブネット・マスク。 数字は MSB のいくつのビットが関係あるかを示しています。 IPv4 アドレス/サブネット・ペアが *a.b.c.d/e* と書かれているときは、*e* は 0 から 32 の範囲の数字です。 この数字は、サブネット・マスクの先頭の 1 の数を示します。 例えば、IPv4 アドレスの場合は、`/24` はネットマスク `255.255.255.0` を示します。32 ビットで表示される場合は、`11111111.11111111.11111111.00000000` となります。これは 24 個の 1 の後に 8 個のゼロが続いています。

proto `/etc/protocols` ファイルに記録されているプロトコル番号または名前 (例えば、`=tcp`)。

start_port_range
ルールが適用される TCP ポートまたは UDP ポートのいずれか、あるいはルールがポートの範囲に適用される場合はその範囲の先頭。 これは、`/etc/services` ファイルに記録されている UDP または TCP サービスのポート番号または名前のいずれかとすることができます。

end_port_range
ポート範囲の上限。

AIX Trusted Network フラグの説明*:*

The AIX Trusted Network システムには 2 つのフラグ・クラスターがあります。 これらが指定されないと、デフォルト値が使用されます。

-d フラグおよび **-r** フラグは、以下のように使用されます。

-d *drop*

drop すべてのパケットから AIX Trusted Network をドロップするように構成する

r このインターフェースではすべてのパケットをドロップする

n このインターフェース (インターフェース・デフォルト) ではすべてのパケットを自動的にドロップしない

i インターフェース・デフォルトを使用する (ホスト・デフォルト、ホストのみ)

-f*rflag*:*tflag*

rflag 着信 (受信) パケットでのセキュリティー・オプション要求

r RIPSO のみ

c CIPSO のみ

e CIPSO または RIPSO のいずれか

n CIPSO でも RIPSO でもない (システム・デフォルト)

a 制限なし

i インターフェース/システム・デフォルトの使用 (デフォルト)

tflag 発信 (送信) パケットでのセキュリティー・オプション処理

r RIPSO をすべての発信パケット IP ヘッダーに入れる

c CIPSO をすべての発信パケット IP ヘッダーに入れる

i インターフェース・デフォルトを使用する (ホスト・デフォルト、ホストのみ)

***RIPSO*/*CIPSO* オプション:**

AIX Trusted Network サブシステムは、CIPSO と RIPSO パケットのラベル付けの構成オプションをサポートします。

-rpafs=*PAF_field* [**,** *PAF_field* ...]

IPSO パケットが受信されたときに受け入れられる各 *PAF_field* を指定します。 最大 256 フィールドまで指定できます。

-epaf=*PAF_field*

送信パケットで IPSO を使用してエラー・パケットが送信されたときにエラー応答に付加される *PAF_field* を指定します。

-tpaf=*PAF_field*

送信パケットで IPSO が使用される場合に発信パケットに適用される *PAF_field* を指定します。

PAF_field:**NONE** | *PAF* [**+** *PAF* ...]

PAF_field は *PAF* の集合です。 1 つの *PAF_field* には、5 つの個別の *PAF* を含めることができます。これらは **GENSER**、**SIOP-ESI**、**SCI**、**NSA**、および **DOE** です。 *PAF_field* は、これらの値を正符号 (**+**) で区切って組み合わせたものです。 例えば、**GENSER** と **SCI** の両方を含む *PAF_field* は、**GENSER+SCI** と表されます。 特殊な *PAF_field* **NONE** を使用できます。これは *PAF* を設定しないで *PAF_field* を指定します。

-DOI=*doi*

CIPSO パケットに対する解釈のドメインを指定します。 着信 CIPSO パケットにはこの **DOI** が必要であり、発信 CIPSO パケットにはこの **DOI** を使用してラベルが付きます。

-tags=*tag*[**,***tag* ...]

tag=**1** | **2** | **5**

CIPSO オプションによる送信に対して受け入れられるタグのセットを指定します。 これはコンマで区切った **1**、**2**、および **5** の組み合わせです。 例えば、**1,2** では **1** と **2** のタグが有効になります。

AIX Trusted Network ネットワーク・セキュリティー・ポリシー*:*

最小許容 SL、最大許容 SL、およびデフォルト SL を指定する必要があります。

パケット自体の SL に関する情報を持っていないすべてのパケットには、暗黙またはデフォルト SL が適用されます。 レベルは以下の構文で入力します。

+min +max +default

ラベル・エンコード・ファイルに従って、有効なラベルを使用することができます。 スペースが使用されるラベルを引用符で囲む必要はありません。

netrule 例*:*

netrule コマンドの例を次に示します。

セキュリティー・オプションを渡さず、すべてのパケットの通過を許可するように **en0** を構成するには、次のように入力します。

```
netrule i+ en0 +impl_lo +ts all +impl_lo
```

CONFIDENTIAL A から **TOP SECRET ALL** までの範囲内の CIPSO パケットのみを受け入れるようにホスト **185.0.0.62** を構成するには、次のように入力します。

```
netrule h+i 192.168.0.0 /24 185.0.0.62 -fc:c +confidential a +top secret all +confidential a
```

サブネットからすべての Telnet パケットを除去するには、次のように入力します。

```
netrule h+i 192.168.0.0 /24 =tcp :telnet 192.0.0.5 -dr +impl_lo +impl_lo +impl_lo
```

詳細および例については、『**netrule** コマンド』を参照してください。

ユーザー・アカウントの管理*:*

各ユーザーの識別と認証 (I&A) 情報は保護され、ユーザーを固有に識別し、システム内のユーザーのアクセス許可を検証するために使用されます。

ユーザー ID 情報には、ユーザーの名前、ログイン ID テキスト名、ユーザー ID、グループ ID、ホーム・ディレクトリー、パスワード、パスワード・エージング・パラメーター、シェル、認可、権限、および監査マスクがあります。 ユーザー関連情報はほぼすべて以下のファイルに保管されます。

/etc/passwd
: ユーザー名、ユーザー ID、1 次グループ割り当て、およびホーム・ディレクトリー

/etc/group
: 2 次グループ割り当ておよびホーム・ディレクトリー

/etc/security/passwd
: 暗号化形式のユーザー・パスワード

/etc/security/user
: ログイン制限、パスワード・パラメーター (最小文字数など)、umask など。

通常のユーザーは、/etc/security/passwd ファイルと /etc/security/user ファイルを読むことができません。 /etc/security/passwd ファイルは、オンの任意アクセス・ビットなしで、SYSTEM_HIGH の SL で保護されます。通常ユーザーが暗号化パスワードを読めないようにすると、暗号化パスワードを突き合わせる順次暗号化/比較ルーチンがなくなります。

権限があるユーザーはこれらのファイルを直接編集できますが、通常は **smit** コマンドを使用してユーザー・パラメーターを編集した方が便利です。 **smit** コマンドは System Management Interface Tool (SMIT) を起動し、ユーザー保守などのシステム管理タスクの選択メニューを表示します。

ユーザー *ID* およびグループ *ID:*

ユーザー ID のクラスには、システム ID と通常ユーザー ID の 2 つのクラスがあります。システム ID は、保護サブシステムの所有権と特殊なシステム管理機能用に予約されています。 通常ユーザー ID は、システムを対話式に使用する個人に割り当てられます。

各ユーザーは、システムでユーザーを識別するための固有のユーザー ID を持っています。ユーザーには 1 つ以上のグループ ID も割り当てることができます。グループ ID は同じグループ内のユーザーで共有され、必ずしも固有ではありません。 ID に使用される数値には範囲の制限があります。 この ID の範囲制限を以下の表に定義します。 これらの値は、十分な数のシステム ID、通常ユーザー ID、およびグループ ID を可能にするように定義されています。

システム・ユーザー **ID**
: 0 から 127

通常ユーザー **ID**
: 128 から MAXUID

通常グループ **ID**
: 0 から MAXUID-1

MAXUID 値は、/usr/include/sys/param.h ファイルに定義されます。

新規ユーザーにユーザー ID 値を割り当てる際は注意が必要です。 通常ユーザーに間違えて 128 より小さいユーザー ID 値が割り当てられた場合、そのユーザーはシステムにログインできません。

ユーザー ID 値は再利用しないでください。 ユーザーを削除したら、そのエントリーを /etc/passwd と /etc/security/passwd ファイルに残して、アカウントをロックすることをお勧めします。 これを行うには、**smit** コマンドを使用します。これにより、ユーザーはログインできなくなり、その ID は再利用されません。 ID を再利用しないことで、前のユーザーの除去されていないファイルに新規ユーザーがアクセスできなくなります。 その結果、監査証跡があいまいなく再構成されます。

/etc/passwd、/etc/security/passwd、および /etc/group ファイルは、**mkuser**、**chuser**、**rmuser**、**pwdadm**、および **passwd** コマンドを使用して管理できます。 これらのコマンドは、前述の予防措置とその他のシステム・セキュリティーの考慮事項をすべて実施します。 **mkuser** コマンドは、システムへの通常ユーザーの追加のみを行います。

注**:** 以下の基準を慎重に順守してください。

- 以前に使用されたユーザー ID を新規ユーザーに再割り当てしない
- 重複ユーザー ID を割り当てない
- システム ID を通常ユーザーに割り当てない
- MAXUID をユーザー ID またはグループ ID として割り当てない

パスワード*:*

パスワードはユーザーに関連付けられた文字ストリングです。パスワードを使用してセッションの開始時にユーザーを認証します。

パスワードはシャドー・ファイルに暗号形式で保管されます。非暗号形式のパスワードはシステムに保管されません。

注**:** ロール・ユーザーのパスワードは、システムのセキュリティーに対して非常に重要であり、常に保護しなければなりません。

パスワード・エージング*:*

パスワード・エージング基準が満たされていると、ユーザーはそのパスワードを変更できます。

パスワード・エージングでは、定義された期間、パスワードがシステムに存在した場合に、ユーザーはそのパスワードを変更する必要があります。 パスワード・エージングには、最小経過期間と最大経過期間があります。 パスワードは、この最小経過期間が経過する前に変更することはできません。 最大経過期間が過ぎた後は、パスワードを変更する必要があります。

パスワード・エージングのパラメーターは、`/etc/security/user` ファイルで設定できます。パスワード・エージングに関連したパラメーターは次のとおりです。

maxage
パスワードが有効である週の最大数

maxexpired
有効期限切れのパスワードをユーザーが変更できる maxage 後の週の最大数

minage
パスワードの変更から次の変更までの週の最小間隔

minlen
パスワードの最小の長さ

他のパラメーターを設定して、パスワードで使用できる文字を指定することができます。 パスワードのパラメーターの詳細については、『**passwd** コマンド』を参照してください。

シェル*:*

ワード・プロセッサーや表計算などのアプリケーションで作業している場合、通常、ユーザーはオペレーティング・システムと直接やりとりする必要はなく、アプリケーションがこのやりとりを管理します。 ただし、一部のユーザーは、アプリケーションのインターフェースを使用しないで、オペレーティング・システムと直接やりとりする必要があります。

オペレーティング・システムと直接やりとりする必要がある場合、ユーザーはシェル・プログラムを使用します。 シェル・プログラムを使用すると、ユーザーは AIX コマンドを入力し、ファイルとディレクトリーに直接アクセスして他の操作を実行できます。 すべてのユーザーが、`/etc/passwd` ファイルに指定されたデフォルトのシェル・プログラムを持っている必要があります。ユーザーのデフォルト・シェル・プログラム (`/bin/sh`、`/bin/csh`、`/bin/ksh` など) は、ユーザーがシェルを使用する必要があるときに **login** または **xterm** コマンドで実行されます。

ログイン有効 *SL* および *TL:*

ユーザーにはデフォルトのログイン SL と TL が割り当てられます。 デフォルトのログイン SL と TL は、正常なログイン後のユーザーのプロセスの有効 SL と有効 TL です。

ユーザーがデフォルトのログイン SL でログインしたくない場合は、**login** コマンドの **-e** オプションを使用してログイン時に別の SL を選択できます。ユーザーが提供する SL は、ユーザーの認可により支配され、システム認定範囲内になければなりません。 ユーザーが TL をログイン時に指定するには、**login** コマンドの **-t** オプションを使用できます。

デフォルトのログイン SL と TL は、各ユーザーのユーザー名と認可とともに、/etc/security/user ファイルで定義されます。 ユーザーの有効 SL は、/etc/security/login.cfg ファイルで指定された TTY SL 範囲内になければなりません。ユーザーの有効 SL は、TTY の最大 SL によって支配され、最小 SL を支配する必要があります。 ユーザーの有効 TL は、TTY の TL と同じでなければなりません。

認可*:*

ユーザーのプロセス・シェルにはログインで 6 つのラベルが割り当てられます。

MAC 検査で有効 SL がシステムによって使用されます。 最小 SL 認可および最大 SL 認可は有効 SL を制限します。すなわち、有効 SL は最大 SL 認可より優位にすることができないため、最小 SL より優位にする必要があります。 MIC 検査で有効 TL がシステムによって使用されます。 最小 TL 認可および最大 TL 認可は有効 TL を制限します。すなわち、有効 TL は最大 TL 認可より優位にすることができないため、最小 TL より優位にする必要があります。

ISSO 権限を付与されたユーザーはユーザーの SL 認可、TL 認可、デフォルト・ログイン SL、およびデフォルト・ログイン TL を変更できます。 これらの値は /etc/security/user ファイルに定義されます。

ユーザー情報の担当部門*:*

単一ユーザーはユーザーをシステムへ追加できません。 ユーザーは SA および ISSO 権限を付与されたユーザーの組み合わせアクションによってシステムへ追加されます。

SA 権限を付与されたユーザーは非セキュリティー関連のユーザー情報を追加することができます。このユーザー情報には、ユーザーの名前、ユーザー ID、グループ ID、ログイン ID テキスト名、シェル、およびホーム・ディレクトリーが含まれます。 ISSO 権限を付与されたユーザーはセキュリティー関連のユーザー情報を追加することができます。このユーザー情報には、ユーザーのパスワード、認可、監査マスク、およびロールが含まれます。 ユーザーを追加する二人の資格により、権限を持つ単一ユーザーがシステム全体の権限をその他の任意のユーザーに権限付与することを防ぐことができます。

拡張監査*:*

Trusted AIX は、監査サブシステムが拡張されて、追加のセキュリティー詳細情報を取り込むことが可能になりました。

新規の監査レコード・フィールド*:*

以下のフィールドが Trusted AIX の AIX 監査レコードのすべてに追加されました。これらの新規フィールドは、選択基準として **auditselect** コマンドで使用できます。

- 監査対象プロセスのロール
- 監査対象プロセスまたはオブジェクトの有効 TL
- 監査対象プロセスまたはオブジェクトの有効 SL

- 監査対象プロセスの有効特権

Trusted AIX は一部の監査証跡で、以下のセキュリティー属性も監査します。

- 監査対象プロセスまたはオブジェクトの TL
- 監査対象プロセスまたはオブジェクトの SL
- Trusted AIX 関連セキュリティー・フラグ

これらの新規セキュリティー属性は **auditpr -v** コマンドで表示できます。

監査範囲*:*

Trusted AIX には、管理者が監査対象プロセスまたはオブジェクトの TL および/または SL に基づく監査範囲のセットを指定することを許可する、メカニズムが組み込まれています。 TL または SL が監査範囲外であるすべてのオブジェクトおよびサブジェクトは無視されます。

プロセスおよびオブジェクトの監査範囲を設定するには、**war** スタンザを `/etc/security/audit/config` ファイルへ追加します。

```
war:
        obj_min_sl = "impl_lo a,b"
        obj_max_sl = "TS a,c"
        sub_min_sl = "impl_lo a,b"
        sub_max_sl = "TS a,c"
        obj_min_tl = impl_lo
        obj_max_tl = TS
        sub_min_tl = impl_lo
        sub_max_tl = TS
```

obj_min_sl および **obj_max_sl** はオブジェクトの SL 監査範囲の定義です。 **sub_min_sl** および **sub_max_sl** はサブジェクト (プロセス) の SL 監査範囲の定義です。 **obj_min_tl** および **obj_max_tl** はオブジェクトの TL 監査範囲の定義です。 **sub_min_tl** および **sub_max_tl** はサブジェクト (プロセス) の TL 監査範囲の定義です。

war スタンザは **audit start** コマンドによって読み取られ、監査サブシステムが始動する前にカーネルへアップロードされます。 **war** スタンザを省略した場合は、カーネルの現在の監査範囲が除去されます。カーネルに TL SL 監査がない場合、カーネルはいずれの TL または SL 監査範囲検査も行いません。

Trusted AIX カーネル・フラグ*:*

システムがシステムで Trusted AIX システムとして構成されるときに、グローバル・カーネル・フラグは **_system_configuration** 変数で使用可能に設定されます。システムが Trusted AIX システムとして構成されているかどうかを判別するために、 **__MLS_KERNEL()** マクロが提供されます。 このマクロはユーザー・スペース・アプリケーションまたはカーネル・ルーチンから呼び出すことができます。 **__MLS_KERNEL()** マクロからの戻り値が **1** の場合は、システムが Trusted AIX として構成済みであることを示します。 その他の戻り値の場合は、Trusted AIX システムとして構成されていないことを示します。

既存プログラムの更新*:*

一般に、既存の特権またはトラステッド・プログラムは、変更しないでトラステッド・システムで正しく機能します。

しかし、信頼レベルを強化すること、および/またはこれらのプログラムの上位互換性のために、いくらかの変更を行うことができます。 新規プログラムの作成のために、既存のプログラムを更新する多くの適用すべき推奨事項もあります。 以下の推奨事項は特に適用すべき項目です。

- プログラムが特権プロセスであるかどうかを判別するために (有効ユーザー ID が 0 であるかどうかを判別する)、 Direct Privilege Checking (直接の特権検査) に従って、このプログラムを変更する必要があります。
- ACL の可能性のある存在を反映するために、標準 UNIX システム許可ビット (モード・ビット) を取り扱うコードを変更する必要があります。
- 特権の使用について、setuid-to-root として実行するために使用するコードをテストする必要があり、かつ、適切な特権が割り当てられている必要があります。

バックアップおよび復元:

Trusted AIX システムにおけるデータのインポートおよびエクスポートは、トラステッド・バージョンの **backup** および **restore** コマンドを使用します。

backup および **restore** コマンドはラベルを取り扱えるように拡張されました。 これらの拡張機能はユーザーには認識されません。そして、ラベルの拡張機能だけではなく、これらのコマンドは標準の AIX **backup** および **restore** コマンドと同等の機能を持っています。拡張セキュリティー情報のバックアップまたは復元を使用不可にするには、**-O** フラグを使用します。

システムのインポート/エクスポートは特権と権限メカニズムの組み合わせによって保護されます。

cron の制約事項:

システムが構成モードの場合、**cron** コマンドは使用不可にされ、いずれのジョブも実行しません。 システムが操作モードの場合、**cron** コマンドは、ジョブが実行依頼された機密ラベルおよびユーザーのデフォルト保全性ラベルでジョブを実行します。

制約事項としては、ユーザーの最小認可および最大認可があります。 つい最近には、この認可はジョブが実行依頼された時刻または **cron** コマンドが最後に再始動された時刻のいずれかの設定から行われます。 **cron** コマンドを管理できるのは SA ユーザーだけです。

ファイルシステムのマウントおよびアクセス:

Trusted AIX は、EAv2 ファイルシステムを使用した JFS2 でラベル付け (SL および TL) をサポートします。 SA または SO は、必要に応じて、ラベル付け (CDFS または HSFS) をサポートしないファイルシステムをマウントできます。 この場合、マウントされたファイルシステム上のすべてのファイルは個別の SL、TL、または FSF を持っていませんが、その代わりにマウント・ポイントのセキュリティー属性を継承します。

Trusted AIX システム管理

Trusted AIX システムの適切な管理のためのガイドラインを理解して、システム・セキュリティーを確実にしなければなりません。

Trusted AIX システム管理は、アカウントが管理ロールに関連付けられた特定ユーザーが実施します。 これらのユーザーは情報システムのセキュリティー担当者 (ISSO)、システム管理者 (SA)、およびシステム担当者 (SO) と呼ばれ、これらの各ユーザーは、管理用タスクの特定サブセットを実行することができる権限を持っています。 これらのユーザーは、`isso`、`sa`、および `so` ロールを定義されたシステムにそれぞれ関連付けられます。 用語の ISSO、SA、および SO は、それぞれ `isso`、`sa`、および `so` ロールを持ってい

るユーザーの名称です。 一部の管理職務については、一人で行動する管理者は、これらの職務を遂行するだけの十分な権限を持っていないために、協業している 3 人のシステム管理者のうちの 2 人だけで達成できます。 例えば、新規ユーザーをシステムへ追加する場合は、SA が新規ユーザー・アカウントを追加することが可能であり、 ISSO だけがユーザーのパスワード、認可、および監査マスクを設定できます。 この労務分割は「二人ルール (two-man rule)」として知られています。

注: 二人ルールの有効性は管理ロールに割り当てられる権限に依存します。 必要以上に多くの権限を管理ロールに関連付けると、システムは内部攻撃の影響を受けやすくなります。 権限のロールへの関連付けに関する詳細情報については、『RBAC』を参照してください。

isso、sa、および so ロールを定義されたシステムは、デフォルトで以下の Trusted AIX 権限に関連付けられます。 システムをぜい弱にする可能性がある、これらの関連付けを変更する場合は、十分に注意して行ってください。

表 41. ロールおよび権限

isso	sa	so
		aix.mls.login
	aix.mls.printer	
aix.mls.network.config		
aix.mls.network.init		
aix.mls.network.config		
aix.mls.login		
aix.mls.pdir		
aix.mls.system.label		
aix.mls.tpath		
aix.mls.label		
aix.mls.system.config		
aix.mls.proc		
aix.mls.clear		
aix.mls.lef		
aix.mls.stat		
aix.mls.printer		

情報システム・セキュリティー担当者のシステム管理:

Trusted AIX システムは ISSO、SA、および SO ユーザーによって管理されます。

Trusted AIX インストールでは、 **isso**、**sa**、および **so** の 3 つのデフォルト・ユーザー・アカウントが作成されます (これらのアカウントは正規の AIX から Trusted AIX への移行のケースでは、まだ存在していません)。これらのユーザーはそれぞれ isso、sa および so と関連付けられます。

注: デフォルトのアカウントは Trusted AIX システムの初期設定および構成だけを目的にしています。 これらのロールをその他の通常のユーザーに割り当てることをお勧めします。 これらのロールがその他のユーザーに割り当てられると、デフォルトのユーザー・アカウントを除去できるようになります。 Trusted AIX インストールに関する詳細情報については、「インストールおよび移行」を参照してください。

ISSO 活動

情報システム・セキュリティー担当者 (ISSO) の基本的な職務はシステムのセキュリティー管理です。 ISSO 権限を持っているユーザーのみ、ISSO 活動を行うことができます。 これらの活動は、以下のとおりです。

- サイト・セキュリティー・ポリシーを計画、実装、および実施する。
- システム全体にわたるユーザーの整理、権限、特権、ログイン制御、およびパスワード・パラメーターのためのデフォルトを設定する。
- ユーザー・アカウントがシステム管理者によって作成されるときの、ユーザーに配置されるトラスト・レベルを反映するユーザー認証プロファイルをセットアップする。
- セキュリティー属性、SL、および TL をデバイス (端末装置、プリンター、取り外し可能ディスク・ドライブ、および磁気テープ・ドライブ) へ割り当てる
- セキュリティー・フラグ、ラベル、特権、および権限セットをファイルへ割り当てる
- システム障害のイベントでシステムをトラステッド状態にリカバリーする

監査システムの管理*:*

監査コマンドへのアクセスは **AUDITSYS** 権限を持つユーザーに制限されます。詳細情報については、『**audit**、**auditselect**、および **auditpr** コマンド』を参照してください。

下の例で以下に示す項目について説明します。

1. 監査証跡ファイルに使用されるファイルシステムの作成方法
2. 監査システムの始動方法
3. いくつかのレコードの生成方法
4. 各種のタイプのレコードを取り出すために監査証跡を解析する方法

FSADMIN 権限を持っているユーザーとして、以下のコマンドを実行します。

```
/usr/sbin/crfs -v jfs -g rootvg -m /audit -a size=32M -A yes
mount /audit
```

以下のエントリーを /etc/security/audit/config ファイルのユーザー・セクションに追加するには、**/tbin/auctlmod -e** コマンドを使用します。

```
username = ALL
```

username はシステムにログオン可能な実ユーザーの名前で置き換えます。

ISSO ユーザーとして、/tmp/top_secret という名前のファイルを作成し、ファイルの SL を **TS ALL** に変更します。

```
touch /tmp/top_secret
/usr/sbin/settxattr -f sl= "TS ALL" /tmp/top_secret
```

AUDITSYS 権限を持っているユーザーとして、以下のコマンドを実行します。

```
/usr/sbin/audit start
```

これで、監査システムのセットアップは完了して開始されました。 *username* で指定されたユーザーがシステムにログオンするときに、このユーザーのアクションは記録されます。

/etc/security/audit/config ファイルの *username* によって指定されたユーザー名でシステムにログオンし、以下のコマンドを実行します。

```
ls -l /tmp/top_secret
exit
```

AUDITSYS 権限を持っているユーザーとして、以下のコマンドを実行します。

```
audit shutdown
$ /usr/sbin/auditselect -e "mac_fail==WILDCARD" /audit/trail | ¥
/usr/sbin/auditpr -v -APSV > /tmp/audit_trail-mac_failure
```

/tmp/audit_trail-mac_failure ファイルにリダイレクトされた監査証跡をテストして、 **mac_fail** を検索します。 auditselect は以下のオプションを受け入れるように変更されています。

- **subj_sl**
- **obj_sl**
- **mac_fail**
- **mac_pass**
- **mic_fail**
- **mic_pass**
- **priv_fail**
- **priv_pass**
- **auth_pass**
- **fsf_fail**
- **fsf_pass**

これらのオプションは、すべてマッチング値として単語の「**WILDCARD**」を使用します。

オブジェクトとプロセス・ラベルの管理*:*

すべてのファイルシステム・オブジェクトとシステム・プロセスに関連したラベルがあります。

通常のファイル以外のすべてのファイルシステム・オブジェクトには、機密ラベルと保全性ラベルの範囲があります。 プロセスには、機密ラベルと保全性ラベルの両方の範囲があります。 この範囲に加えて、プロセスには有効 SL と有効 TL があります。 このラベルは、プロセスが実行されている現行の SL または TL を示します。 ラベルを表示するには、**lstxattr** コマンドを使用します。ファイルシステム・オブジェクトとプロセスのラベルを設定するには、**settxattr** コマンドを使用します。

ネットワーク・セキュリティーの管理*:*

AIX Trusted Network では、ISSO がいくつかのテーブルを定義する必要があります。 これらのテーブルは、/etc/security ディレクトリーに保管されます。 **tninit** コマンドは、バイナリー・バージョンを生成し、それをカーネルにロードするために使用されます。

ホストとネットワーク・インターフェースのルールにより、システムが着信と発信のネットワーク・パケットを処理する方法が決まります。 ホスト・ルールは特定のホストに適用されます。 ネットワーク・インターフェース・ルールは、ホストがネットワークに接続するインターフェースに適用されます。ホスト・ルールとインターフェース・ルール間に競合がある場合は、ホスト・ルールが優先されます。

ルールを追加、編集、照会するには、**netrule** コマンドを使用します。通常、これらのルールは、使用されるプロトコル、ルールが適用されるアドレスの範囲 (ホストとポートの両方)、パケットに割り当てられる SL に関連します。 詳細情報については、『**netrule** コマンド』を参照してください。

AIX Trusted Network サブシステムを初期化し、バイナリー・フォーマットでルールを保存し、テキスト・フォーマットでルールを表示するには、**tninit** コマンドを使用します。

セキュリティーの構成可能機能*:*

ブート・シーケンス時に、構成可能機能の設定が表示されます。

構成可能設定は ODM に保管されます。 これらの設定を表示するには **getsecconf** コマンドを使用し、ISSO ユーザーがこれらの設定を変更するには **setsecconf** コマンドを使用します。

ラベルの管理*:*

ISSO ユーザーは `/etc/security/enc/LabelEncodings` ファイルを変更して、ラベル・エンコードを追加、変更、または削除できます。 `/etc/security/enc/LabelEncodings` ファイルには、人間に読み取り可能な名前がシステム機密ラベルのバイナリー表記にマップされる方法を定義します。

注**:** 稼働システムで機密ラベル・エンコード・ファイルを変更する場合は、非常に注意深く行わないと、ラベルが無効になることがあります。 オブジェクトは単語または制約のある単語の組み合わせでラベルを付けることができるため、不注意による制約組み合わせ単語の変更、追加、または削除の結果、ラベルが無効になることがあります。

`/etc/security/enc/LabelEncodings` ファイルは **l_init** ライブラリー・ルーチンによってバイナリー・フォームに変換され、テーブルに保管されます。 これらのテーブルは SL、プリンター・バナー、および認可とその内部バイナリー・エンコードとの間での変換に使用されます。

Trusted AIX は MITRE Compartmented Mode Workstation Labeling ソフトウェアをラベル付けインプリメンテーションの基本として使用します。「Compartmented Mode Workstation Labeling の文書: Encodings Format, DDS-2600-6216-93 (MTR 10649 revision 1), September 1993」には、標準ラベルのエンコード・フォーマットについて解説されています。

標準ラベルのエンコード・フォーマットは保全性ラベルおよび機密ラベルを取り扱います。このラベルは `/etc/security/enc/LabelEncodings` ファイルの 「**Sensitivity Labels (**機密ラベル**)**」セクションに指定されているものと同じです。

Trusted AIX はオプションで「保全性ラベル」セクションをサポートし、保全性ラベルが機密ラベルと異なることを可能にします。

分割ディレクトリーの管理*:*

通常のユーザー・プロセスでは、分割ディレクトリーは、通常のディレクトリーと同様に表示され、機能します。 ただし、分割ディレクトリーでは、種々の SL を持つさまざまなプロセスに対して、同じディレクトリーのさまざまな内容が表示されます。

例えば、**SECRET** セキュリティー・ラベルで実行されるプロセスが分割ディレクトリーに **foo** というファイルを作成した場合、TOP SECRET セキュリティー・ラベルで実行される 2 番目のプロセスはそのディレクトリー内の **foo** ファイルを表示できず、そのファイルにアクセスできません。 また、2 番目のプロセスは、最初の **foo** ファイルを干渉せずに、独自の **foo** ファイルを作成できます。

これは隠しサブディレクトリーを使用して行われます。 プロセスが分割ディレクトリーにアクセスする際に使用する固有の SL ごとに、分割サブディレクトリーがあります。 プロセスが分割ディレクトリーにア

クセスすると、システムはそのプロセスを隠しサブディレクトリーに自動的にリダイレクトします。 前述の例では、2 つの **foo** ファイルは、ユーザーに対しては同じディレクトリー内に表示されますが、実際は異なるサブディレクトリー内にあります。

分割ディレクトリーの詳細については、 469 ページの『分割ディレクトリー』を参照してください。

分割ディレクトリーは、EAv2 のある JFS2 でサポートされます。

分割ディレクトリーの作成*:*

分割ディレクトリーが作成されるときのデフォルト SL 範囲は、低位レベル・システム SL から高位レベル・システム SL までです。 分割ディレクトリーをアクセスするときに、カーネルはラベル特定の子ディレクトリーを作成し (それが存在していない場合)、ユーザー・プロセスを子ディレクトリーにリダイレクトします。

pdmkdir コマンドを使用して、分割ディレクトリーを作成します。 **pdmkdir** コマンドには、DAC、MAC、および MIC の各制約事項をオーバーライドするための **aix.mls.pdir.create** 権限が必要です。 **pdrmdir** コマンドを使用して、空の分割ディレクトリーを除去します。

分割サブディレクトリーおよびサブ*-*サブディレクトリー

分割ディレクトリーのラベル特定の子ディレクトリーが分割サブディレクトリーです。 プロセスが分割サブディレクトリーの下に子ディレクトリーを作成すると (**mkdir** コマンドを使用)、その子ディレクトリーが分割サブ-サブディレクトリーです。

分割サブディレクトリーが作成されるときに、このサブディレクトリーはこの親の分割ディレクトリーのセキュリティー属性 (最小 SL および最大 SL を除く) を継承します。 最小および最大 SL は、分割サブディレクトリーを最初にアクセスする仮想モード・プロセスの有効 SL に設定されます。

Trusted AIX は、以下の 4 つの異なるタイプのディレクトリーを認識します。

- 通常のディレクトリー (dir)
- 分割ディレクトリー (pdir)
- 分割サブディレクトリー (psdir)
- 分割サブ-サブディレクトリー (pssdir)

仮想モードおよび実モード*:*

分割ディレクトリーには仮想モードと実モードの 2 つのアクセス・モードがあります。

仮想モードでは、分割ディレクトリーにアクセスするプロセスは、そのラベル固有の分割サブディレクトリーの内容しか表示できません。 分割ディレクトリーは、仮想モードで実行されるプロセスでは表示されません。分割ディレクトリーは、実モードで実行されるプロセスで表示されます。 実モードで実行されるプロセスは、分割ディレクトリーと分割サブディレクトリーのすべての実内容が表示できます。 実モード・プロセスの場合、システムはリダイレクトを実行しません。

デフォルトでは、プロセスは仮想モードで実行されます。 実モードは、ファイルシステムの管理専用です。 現行プロセスのシェルのモード以外のモードでコマンドを実行するか、別のモードのシェルに切り替えるには、**pdmode** コマンドを使用します。

実モードのユーザー・プロセスは分割ディレクトリーとサブディレクトリーを認識して操作できますが、このタイプのアクセスと操作は注意して実行する必要があります。 例えば、実モードのプロセスで通常のディレクトリーが作成されるか分割ディレクトリーに移動された場合、そのディレクトリーは仮想モードで実行されるプロセスでは表示されません。

分割ディレクトリーは仮想モードのプロセスでは通常のディレクトリーのように見えますが、その分割ディレクトリーにはいくつかの制限があります。

階層*:*

分割ディレクトリーおよびサブディレクトリーは階層になっています。

分割ディレクトリーおよびサブディレクトリーの階層は、以下のルールによって規定されます。

- ディレクトリーは以下のタイプのうちのどれかにする。
  - 正規のディレクトリー
  - 分割ディレクトリー
  - 分割サブディレクトリー
  - 分割サブ-サブディレクトリー
- ディレクトリーは、いかなるときでも複数のタイプにすることはできない
- 分割サブディレクトリーの親は分割ディレクトリーでなければならない
- 分割サブディレクトリーの子ディレクトリーは、それぞれ分割サブ-サブディレクトリーでなければならない
- 分割サブ-サブディレクトリーの親は分割サブディレクトリーでなければならない

上記のルールに違反すると、結果として、無効な分割ディレクトリー・ツリーになったり、行動が未定義のままの不整合のファイルシステムになったりします。

ファイルシステムのマウント*:*

分割ディレクトリーまたはサブディレクトリーをマウント・ポイントにすることはできますが、分割サブ-サブディレクトリーをマウント・ポイントにすることはできません。 同様に、マウントされているファイルシステムのルートを分割ディレクトリーまたはサブディレクトリーにすることはできますが、分割サブ-サブディレクトリーにすることはできません。

ディレクトリーの作成および削除*:*

仮想モード・プロセスの実行が分割サブ-サブディレクトリーで行われているとき、 **mkdir** コマンドは正規のディレクトリーを作成します。 同じプロセスが分割サブディレクトリーにあって、**mkdir** コマンドを実行する場合、分割サブ-サブディレクトリーは自動的に作成されます。 いずれかの空のディレクトリーは MAC、MIC、および DAC 制限に従って削除できます。

ディレクトリーの移動*:*

MAC、MIC、および DAC 制限がディレクトリーを移動するときに適用されます。

正規のディレクトリーはどこにでも移動できます。 この新規の親ディレクトリーが分割サブディレクトリーの場合、移動された正規のディレクトリーは分割サブ-サブディレクトリーになります。 そうでない場合は、なお正規のディレクトリーのままです。その新規の親が分割ディレクトリーであり、その名前が可能性のある分割サブディレクトリーの名前と調和しない場合、すぐに、その可能性のある分割サブディレクトリーへの仮想モード・プロセスのリダイレクトは失敗します。

分割ディレクトリーは別の正規のディレクトリーへ移動することが可能であり、移動後もそのまま分割ディレクトリーであり続けます。 ネストされた分割ディレクトリーは、さらに追加の利点がないために Trusted AIX ではサポートされません。

分割サブディレクトリーは分割ディレクトリーにのみ移動できますが、移動後も分割サブディレクトリーのままです。 分割サブディレクトリーを正規のディレクトリー、分割サブディレクトリー、または分割サブ-サブディレクトリーへ移動することは禁止されています。

分割サブ-サブディレクトリーはどこへも移動できます。 移動先の新規の親が正規のディレクトリー、分割ディレクトリー、または分割サブ-サブディレクトリーの場合、これは正規の親ディレクトリーになります。 そうでない場合は、なお分割サブ-サブディレクトリーのままです。

表 42. ディレクトリー移動の要約

移動元ディレクトリー・タイプ	移動先が正規のディレクトリー	移動先が分割ディレクトリー	移動先が分割サブディレクトリー	移動先が分割サブ-サブディレクトリー
正規	許可、 正規のディレクトリーのまま	許可[1]、 正規のディレクトリーのまま	許可[1]、 分割サブ-サブディレクトリーになる	許可、 正規のディレクトリーのまま
分割	許可、 分割ディレクトリーのまま	許可[1]、 分割ディレクトリーのまま	未許可	許可、 分割ディレクトリーのまま
分割サブディレクトリー	未許可	許可、 分割サブディレクトリーのまま	未許可	未許可
分割サブ-サブディレクトリー	許可、 正規のディレクトリーになる	許可、 正規のディレクトリーになる	許可、 サブ-サブディレクトリーのまま	許可、 正規のディレクトリーになる

[1] 名前が可能性のある (現在は存在していない) 分割サブディレクトリーの名前と調和しない場合、すぐに、その可能性のある分割サブディレクトリーへの仮想モード・プロセスのリダイレクトは失敗します。

ディレクトリー・タイプの変更*:*

正規のディレクトリーを分割ディレクトリー・タイプに変更する場合は、 **pdset** コマンドを使用します。 分割ディレクトリーを正規のディレクトリーに変更するコマンドはありません。

i ノード番号の置き換え*:*

分割サブディレクトリーがアクセスされて、 i ノード番号またはその親分割ディレクトリー (..) の i ノード番号が必要とされるときに、それぞれ、その親分割ディレクトリーの i ノード番号またはその親分割ディレクトリーの親の i ノード番号が返されます。 分割サブ-サブディレクトリーがアクセスされ、分割サブ-サブディレクトリー (..) の親の i ノード番号が必要とされるときに、その祖父母分割ディレクトリーの i ノード番号が返されます。

分割ディレクトリー・コマンド*:*

以下のコマンドは分割ディレクトリーに適用されます。

pdmkdir
分割ディレクトリーを作成する

pdrmdir
分割ディレクトリーおよびサブディレクトリーを除去する

pdlink
分割サブディレクトリー全体にわたるファイルをリンクする

pdset ディレクトリーを分割ディレクトリーに設定する

pdmode

現行ディレクトリーのアクセス・モードを返す

指定ディレクトリーのアクセス・モードでコマンドを実行する

分割ディレクトリーに変換されている正規のディレクトリーは正規のディレクトリーに変換して戻すことができます。

システム・セキュリティーの検討*:*

ISSO はシステムのセキュリティー状況の検討に対して責任があります。 システム・セキュリティーの検討は、インストールのすぐ後、またはシステム保全性が危険にさらされたときに行う必要があります。また、システム・セキュリティーの検討は定期的に行ってください。

`/etc/security/tsd/tsd.dat` ファイルに保管されるシステム保全性データベース・ディレクトリーには、重要なコマンドやシステム・デバイスなどのファイルシステム・オブジェクトのセキュリティー関連情報が含まれます。 新規デバイスが追加されたり、ファイルのセキュリティー情報が変更された場合は、このデータベースを更新する必要があります。 詳しくは、『**trustchk** コマンド』を参照してください。

trustchk コマンドは、ファイル、ディレクトリー、またはデバイスの現行のセキュリティー設定値をシステム保全性データベース内の対応するエントリーと比較し、セキュリティー属性の不整合を修復します。 **trustchk** コマンドを実行できるのは、ISSO 権限があるユーザーのみです。

TTY 管理*:*

TTY デバイスの最小 SL、最大 SL、および TL は、`/etc/login.cfg` ファイル内の ttys データベースに定義されます。 詳しくは、『**chsec** コマンド』を参照してください。

TTY ポートでログインしているユーザーの有効 SL は、このファイル内でこのポートに対して定義された範囲内になければなりません。 TTY ポートに対して NOTL 以外の TL が定義されている場合は、ユーザーの有効 TL は指定された TL と同じでなければなりません。

ユーザーの認可の管理*:*

ISSO、SA、および SO ユーザーを含む各ユーザーがシステムにログインするにはラベルが必要です。 ユーザーの認可は、`/etc/security/user` ファイルにユーザーのスタンザの一部として指定できます。 **minsl**、**maxsl**、**defsl**、**mintl**、**maxtl**、および **deftl** 属性は、ユーザーに対してそれぞれ最小 SL、最大 SL、デフォルト SL、最小 TL、最大 TL、およびデフォルト TL を指定します。 これらの属性がユーザーのスタンザに指定されると、ファイルのデフォルト・スタンザに指定された値がユーザーに割り当てられます。

セキュリティー認可データベースを変更できるのは、ISSO ユーザーのみです。 ユーザーの認可をリストするには **lsuser** コマンドと **lssec** コマンドを使用し、ユーザーの認可を変更するには **chuser** コマンドと **chsec** コマンドを使用できます。

デフォルトの SL 値は、最大 SL 値によって支配され、最小 SL を支配しなければなりません。 同様に、デフォルトの TL 値は、最大 TL 値によって支配され、最小 TL を支配しなければなりません。

注*:* ユーザーがシステムに正常にログインするには、前述の関係が真でなければなりません。

システム管理者のためのシステム管理**:**

SA ユーザーは、主に、セキュリティーに関連しないシステム管理の面で責任があります。

SA ユーザーの責任は次のとおりです。

- ユーザー・アカウントを追加、除去、維持する。
- システム・ソフトウェアとファイルシステムの内部保全性を確保するタスクを ISSO ユーザーと共有する。
- ファイルシステムを作成し、維持する。 これにはディスク・レイアウトの計画、ディスクの区画化とディスク区画サイズの変更、swap スペースとシステムおよびユーザー・ディレクトリーのスペースの割り当て、ファイルシステム使用率のモニター、不良ディスク・ブロックの検出と処理、ファイルとファイルシステムを移動、削除、アーカイブ、または圧縮することによるファイルシステム・スペースの管理などがあります。
- エラー・データを解析し、ファイルシステム、システム・メモリー、デバイスなどのシステム・コンポーネントをテストしてシステム問題を識別し、報告する。

ユーザー・アカウントの管理**:**

SA ユーザーはシステムへの新規ユーザーの追加に対して責任があります。 ISSO ユーザーは、新規ユーザーのログオンの許可およびシステムでのコマンドの実行に対して責任があります。

ユーザー・アカウントへの権限の追加については、『情報システム・セキュリティー担当者のシステム管理』を参照してください。

SA ユーザーがユーザーをシステムに追加したら、その新規ユーザーがシステムにアクセスできるように初期パスワードを設定するために ISSO ユーザーに通知しなければなりません。

ユーザーがシステムにアクセスしないことが決まったら、そのユーザーを直ちに除去する必要があります。ユーザーを除去できるのは、SA ユーザーのみです。 システムから除去されたユーザーのユーザー ID は再利用しないでください。ただし、この ID を元のユーザーに戻し、システムでそのユーザーを再インストールする場合を除きます。

ユーザー・アカウントの設定および変更については、『**mkuser**、**rmuser**、**chuser**、および **pwadm** コマンド』を参照してください。

プリンターの管理**:**

プリンターは適切にインストールされると、SA と SO ユーザーの結合アクションによりシステムに追加されます。 SO ユーザーはプリンターをシステムに追加し、SA ユーザーはプリンターの SL 範囲を設定します。 ISSO ユーザーはこれらの作業の両方を実行する権限を持っています。

プリンターがシステムに追加されるまで、プリンターの SL 範囲は設定できません。 プリンターを管理するには、**smit** コマンドを使用します。

注**:** PostScript と ASCII ファイルのラベル付き印刷は、PostScript プリンターでのみサポートされます。

プリンターへの MAC アクセスは、ファイルを印刷するプロセスの SL により決まります。 この SL は、バナー、ヘッダー/フッター、およびトレーラー・ページに表示されます。 **lp** コマンドを使用するプロセスには、印刷されるファイルへの MAC、MIC、および DAC アクセスが必要です。これらのアクセスがない場合、**lp** コマンドは印刷要求を生成しません。

プリンターがシステムから除去された場合は、プリンター・プロファイルをシステムから即時に削除する必要があります。 この操作を実行できるのは、SO 権限のあるユーザーのみです。

ファイルシステムの管理*:*

ファイルシステムはディレクトリー、データ・ファイル、実行可能ファイル、およびスペシャル・ファイルで構成されます。 ファイルシステムは、ハード・ディスクおよびフロッピー・ディスケットなどの各種大容量ストレージ・デバイスに保管できます。

SA ユーザーのみがファイルシステムの作成および保守を行うことができますが、ファイルシステムのマウントおよびアンマウントは SA ユーザーおよび SO ユーザーの両方が可能です。

fsck コマンドによるファイルシステムの検査*:*

ファイルシステムの内部保全性は **fsck** コマンドで定期的に検査する必要があります。 **fsck** コマンドはアンマウント済みのファイルシステムで実行しなければなりません。. **fsck** コマンドは SA ユーザーによってのみ実行できます。

デフォルトで、**fsck** コマンドは対話式に実行し、孤立ファイルまたはディレクトリーが検出されるときに実施する必要のあるアクションについて、ユーザーにプロンプトを出します。ユーザーには、ファイルを削除するか、またはファイルのリカバリーを試みるかの選択肢があります。 ユーザーがファイルのリカバリーを指定すると、**fsck** コマンドにより `/lost+found` ディレクトリーへのファイルの保管が試みられます。

fsck コマンドが完了してリカバリーされたファイルを `/lost+found` ディレクトリーに保管した後に、ISSO ユーザーはそのセキュリティー・レベルを決定するために、リカバリーされたファイルについて検討する必要があります。 通常のユーザーがリカバリー済みのファイルをアクセスすることを防ぐために、`/lost+found` ディレクトリーを **SYSTEM_HIGH** SL に割り当てることをお勧めします。

詳細情報については、『**fsck** コマンド』を参照してください。

システム担当者 **(SO)** のシステム管理**:**

S0 ユーザーは、主に、セキュリティーに関連するシステム管理の面で責任があります。

ファイルシステムの管理*:*

システム担当者 (SO) はファイルシステムの管理に対して責任があります。

サポートされるファイルシステム*:*

Trusted AIX は、すべてのディスク・ベース・ファイルシステムをサポートします。

Trusted AIX では、単一レベルのファイルシステムとして、JFS2 以外のすべてのファイルシステムがサポートされます。 これらのファイルシステムは、Trusted AIX システムでマウントされ、ラベルと他のセキュリティー属性を自動的に受け取り、Trusted AIX で実行されるセキュリティー・メカニズムに従います。単一レベルのファイルシステムにあるすべてのファイル・オブジェクトは、同じセキュリティー属性を持っています。 これらのセキュリティー属性は、マウント・ポイントから継承されます。

Trusted AIX では、マルチレベルのファイルシステムとして JFS2 が実装されます。 マルチレベルのファイルシステムにある各ファイル・オブジェクトは、独自のセキュリティー属性 (セキュリティー・ラベル) を持っています。 例えば、JFS2 ディレクトリーには、独立した最小 SL と最大 SL があります。

単一レベルのファイルシステムでは、マウント・ポイントの最小 SL と最大 SL は同等であり、そのマウント・ポイント下のすべてのディレクトリーとファイルもこれらの SL と同等でなければなりません。

ファイルシステムのマウントおよびアンマウント*:*

SO ユーザー (**aix.fs.manage.mount** 権限を持つ) は、ファイルシステムのマウントまたはアンマウントを行うことができます。 **mount** コマンドは、デバイスのスペシャル・ファイル名とマウント・ディレクトリー (オプション) を使用します。

マルチレベル JFS2 ファイルシステムがマウントされると、マウント・ディレクトリーにファイルシステムのルートのラベルが割り当てられます。 マルチレベル・ファイルシステムでは、各ファイルに独自の機密ラベルと保全性ラベルがあります。 ファイルが変更されると、それに応じてそのラベルが更新されます。

プリンターの管理*:*

SO ユーザーは **lpadmin** コマンドを使用することで、プリンターの追加と取り外し、プリンターの変更、およびプリンター・サブシステム全体のその他特定タイプの制御を行うことができます。 SA ユーザーは **lpadmin** コマンドを使用することで、プリンターの機密ラベル (SL) を追加または変更すること、およびコマンドの使用可能化と使用不可化により、プリンターを使用可能にしたり、使用不可にしたりすることができます。

プリンター・サブシステム*:*

プリンター・サブシステムは、プリンター操作に関連した多くのタスクを実行します。

プリンター・サブシステムのタスクは次のとおりです。

- プリンターとその属性の管理
- ユーザー印刷ジョブの受信、保管、およびスケジューリング
- 複数のプリンターに対する印刷ジョブのスケジューリング
- プリンターとインターフェースをとるプログラムの開始
- プリンターと印刷ジョブの状況の追跡
- 問題が発生した場合の問題の報告
- ユーザー印刷ジョブをプリンターの SL 範囲に含まれるものに制限
- アクセスを実行依頼済みユーザー印刷ジョブに制限
- アクセスをプリンター対応ファイルとディレクトリーに制限
- プリンター出力の適切なラベル付け

プリンターのセキュリティー機能*:*

Trusted AIX では、いくつかのセキュリティー機能を組み込むために、プリンター・サブシステムを変更します。

プリンター・サブシステムは、システム ID **lp** により所有される保護サブシステムです。 これにより、通常のユーザーは、ユーザー自身が実行依頼した印刷ジョブとプリンター・デバイスのスペシャル・ファイル以外のプリンター対応ファイルとディレクトリーにアクセスできません。

プリンター・サブシステムは、ユーザーの実行依頼済み印刷ジョブがプリンターの SL 範囲に含まれていることを検証します。 この検証が実行されるのは、ユーザーが **lp** コマンドで印刷ジョブを実行依頼した

とき、およびその実行依頼されたジョブが **lpsched** デーモンにより印刷される前です。ユーザーの印刷ジョブが拒否された場合に備えて、管理者はプリンター・サブシステムのセキュリティー検査を理解している必要があります。

バナー・ページは、すべての印刷ジョブに対して印刷されます。 バナー・ページには、人間が理解できる印刷ジョブの SL が含まれます。 バナー・ページは、すべての印刷ジョブの前後に表示されます。 ユーザーはバナーなしで印刷できますが、これは監査可能アクションです。 各ページのヘッダー・ラベルとフッター・ラベルが正しいこと、およびこれらがバナー・ページのラベルによって支配されていることを常に検証する必要があります。

注**:** ライン・プリンター管理者は、プリンターごとにラベル範囲を設定する必要があります。 プリンターに単一ラベルを割り当てるには、次のコマンドを実行します。

lpadmin -d *printer_name* **-J***label* **-L***label* これにより、指定された *label* のある情報だけがプリンターで印刷されます。

プリンター・コマンドの要約*:*

一部のプリンター・サブシステム・コマンドは、ユーザーが実行できます。 ただし、SO、SA または ISSO ユーザーしか実行できないプリンター・サブシステム・コマンドもあります。

ユーザーが実行できるプリンター・サブシステム・コマンドを次の表に示します。

lp ファイルをプリンターに送信します

lpstat プリンター・サブシステムの状況報告を行います

プリンター・サブシステム管理コマンドには SO 権限が必要です。ただし、SA または ISSO 権限を持つユーザーは **lpadmin** コマンドを実行してプリンターのラベル範囲を指定し、 **lpstat** コマンドを実行してプリンターとジョブ要求 SL を表示できます。 プリンター・サブシステムの管理コマンドを次の表に示します。

accept
プリンターでジョブを許可します

cancel
ファイルの印刷要求を取り消します

disable
プリンターを非活動化します

enable
プリンターを活動化します

lpadmin
プリンター構成を設定または変更します

lpfilter
プリンター・フィルターを設定または変更します

lpforms
プリンターの書式を設定または変更します

lpmove
印刷要求を移動します

lpsched
要求を印刷します

lpshut
印刷サービスを停止します

lpusers
プリンター優先順位を設定または変更します

reject　プリンターでジョブを阻止します

コマンド・ラインからのプリンター管理*:*

コマンド・ラインからプリンターを管理するには、**accept**、**enable**、**disable**、**lpstat**、および **lp** コマンドを使用できます。

accept コマンドを使用すると、ジョブをプリンターに送信できます。 プリンター *laser* が印刷ジョブを受け入れるようにするには、次のコマンドを実行します。

```
/usr/sbin/accept laser
```

laser で指定されたプリンターは、印刷ジョブ要求を受信できるようになります。ただし、プリンターが使用可能になっていないと、印刷ジョブは印刷されません。プリンターを使用可能にするには、enable コマンドを実行します。

```
/usr/bin/enable laser
```

enable コマンドと **disable** コマンドは管理コマンドであり、これらのコマンドを実行できるのは ISSO または SA 権限を持つユーザーのみです。

プリンターが適切に設定されたことを確認するには、次の **lpstat** コマンドを実行します。

```
lpstat -p laser -l
```

このコマンドは、プリンター *laser* の詳細形式の状況報告を表示します。 **-l** オプションを指定しないで **lpstat** コマンドを実行すると、要約形式の状況報告が表示されます。 ユーザーが SA または ISSO 権限を持ち、**-l** オプションが使用される場合は、プリンターの SL 範囲も報告されます。

印刷要求の状況を判別するには、**lpstat** コマンドを実行します。

```
lpstat -o
```

このコマンドは、すべての **lp** 印刷要求をリストします。 ユーザーが SA または ISSO 権限を持っている場合は、各要求の有効 SL と認可が報告されます。

filename を印刷するには、次の **lp** コマンドを実行します。

```
lp -d laser filename
```

それ以外の場合は、**lp** コマンドを実行するときに印刷ジョブの出力先を指定しなければなりません。

管理者がデフォルトの出力先プリンターを設定してある場合は、**-d** *destination_ptr* オプションは必要ありません。 例えば、プリンター laser のファイル filename を印刷するには、次の **lp** コマンドを入力します。

```
lp filename
```

システム・シャットダウンの管理*:*

SO ユーザーは、システムをリブートするか、またはシステムを完全に停止することでシステムをシャットダウンできます。

SO ユーザーは以下のコマンドを実行して、システムのリブートまたは停止を行うか、システムの初期状態を変更できます。

reboot
システムを自動的にリブートします

halt すべてのシステム操作を停止します

shutdown
すべてのシステム操作を停止します

init システムの初期状態を変更します

ファイルのバックアップおよび復元*:*

バックアップは、ハードウェア障害や誤ったファイルの削除によるデータ損失を防ぐのに有効です。 バックアップは定期的に実行し、完全バックアップの間に増分バックアップを行う必要があります。

backup コマンドと **restore** コマンドには、ファイル・バックアップの名前、ロケーション、タイプ、およびその他のオプションを指定するためのオプションがあります。 ファイルまたはブート可能テープで、ルート・ボリューム・グループの Trusted AIX インストール可能イメージを作成するには、**mksysb** コマンドを使用できます。 これらのコマンドを実行するには、**smit** コマンドを使用します。 ファイルシステムのバックアップは、適切にラベルを付けて、安全な場所に保管する必要があります。

Trusted AIX プログラミング

システム・セキュリティーは、トラステッド・コンピューティング・ベース (TCB) のソフトウェア、ハードウェア、およびファームウェアに依存します。 これにはオペレーティング・システムのカーネル全体、すべてのデバイス・ドライバーと System V STREAMS モジュール、カーネル・エクステンション、およびすべてのトラステッド・プログラムが含まれます。 セキュリティー決定を行う際にこれらのプログラムで使用されるファイルもすべて TCB の一部として見なされます。

トラステッド・ソフトウェアを作成する場合は、基本的なシステム・セキュリティーの原則と機能を十分に理解している必要があります。 UNIX ベース・システムにおけるほぼすべてのセキュリティー欠陥は、不完全に作成されたトラステッド・ソフトウェアによるものです。 ただし、Trusted AIX カーネル・セキュリティー検査を使用すると、拡張セキュリティー機能を使用するアプリケーションを作成できます。
Trusted AIX 用に作成されたアプリケーションは、さまざまなセキュリティー・レベルのファイルとプロセスに対して機密性を持つことができ、アプリケーションが使用するプロセスまたはファイルのレベルに応じて異なって動作することができます。 このようなアプリケーションは、マルチレベル認識 (MLS) アプリケーションとして知られています。

トラステッド・システム・プログラマーは、Trusted AIX セキュリティー機能に完全に精通している必要があり、新しい Trusted AIX システム・コールおよびセキュリティー関連コマンドとライブラリーをすべて理解していなければなりません。 この情報は、トラステッド・ソフトウェアを作成または変更するプログラマー用です。 これにはトラステッド・ソフトウェアの変更および作成に対するガイドライン、原則、および注意が含まれます。 この情報では一部のセキュリティー原則とメソッドに対する導入説明を示しており、トラステッド・システム・プログラマーは安全なシステムに関するその他の資料をお読みになることをお勧めします。

トラステッド・ソフトウェアのプリンシパル

トラステッド・ソフトウェアの作成と変更には、いくつかの重要なプリンシパルが関連しています。これには、信頼と特権、トラステッド・ソフトウェアの設計、最小特権、プログラミング規則、および TCB の保護が含まれます。

信頼および特権:

プロセスに対して適切に特権が与えられている場合に限り、そのプロセスは基本セキュリティー制限 (MAC、MIC、DAC、およびその他の制限付き操作) をバイパスできます。特権 (1 つ以上) を使用して実行されるプロセスは特権プロセスと呼ばれ、そのプロセスが実行されるプログラムは特権 (トラステッド) プログラムと呼ばれます。

特権という用語は、プロセスがセキュリティー関連の操作を実行できるようにする個々の属性を表します。Trusted AIX は、特定のセキュリティー操作を識別してグループ化し、個別の特権を各操作に関連付けます。 これにより、スーパーユーザー (または root) 特権が基本システムから除去されます。 特権はプロセスと実行可能ファイルに関連付けられます。

以下の状況では、プログラムはトラステッドでなければなりません。

- プログラムが特権プロセスとして実行されるように構成されるか、意図されている。これは特権プロセスで実行されることを目的としたすべてのプログラムに適用されます。
- プログラムがセキュリティーの決定において他のトラステッド・プログラムから依存されている。 例えば、機密データベースを変更するプログラムは、他のプログラムがセキュリティーの決定を行う場合にそのデータベース内のデータに依存する場合には、トラステッドでなければなりません。

非トラステッド・プログラムを特権プロセスとして実行しないようにすることは重要です。 非トラステッド・プログラムを特権プロセスとして実行しないようにする方法がいくつかあります。

- 特権プロセスに非トラステッド・プログラムの実行を通常は許可しない。例えば、特権シェルライク・プログラムを実行するユーザーに、特権シェルライク・プログラムで非トラステッド・プログラムを実行しないように注意します。
- 非トラステッド実行可能ファイルに対して固有の特権、継承された特権、または権限のある特権を許可しない。

デバイス・ドライバー、STREAMS モジュール、およびカーネル・エクステンションを含む、オペレーティング・システムのカーネルの部分はすべてトラステッドでなければなりません。 ファイルや物理デバイスなどのデータ・オブジェクトも、これらにセキュリティーの決定においてトラステッド・プログラムから依存される情報が含まれている場合はトラステッドと見なされます。

トラステッド・ソフトウェア設計:

トラステッド・ソフトウェアの作成プロセスは、重要なソフトウェア・コンポーネントの作成プロセスに類似しています。 トラステッド・ソフトウェアの作成では、入念に考えられて文書化された仕様、設計、実装、テスト、および構成制御サイクルに従う必要があります。

トラステッド・ソフトウェア設計の最も重要な面は、サブジェクトとオブジェクトの識別、および適切な抽象化レベルの正確なセキュリティー・アクションの定義です。 多くのセキュリティー・ポリシーは、サブジェクト、オブジェクト、およびアクションに対する制限です。 サブジェクトがオブジェクトの読み取り、変更、または作成のための許可を要求すると、セキュリティー・ポリシーはこれらの要求をモニターし、これらの要求を承認または拒否します。

サブジェクト

サブジェクトは、通常、ユーザー ID とグループ ID により表されます。 通常、この目的にはプロセスの有効ユーザー/グループ ID が使用されますが、実ユーザー/グループ ID を使用した方が適切な場合もあります。

オブジェクト

オブジェクトはアクセスが制御されるデータの集合です。 多くの場合、オブジェクトはファイルです。 トラステッド・プログラムで同じファイル内にある論理的に異なるオブジェクトに対するアクセスを制御するのは一般的ですが、通常はオブジェクトを 1 対 1 でファイルにマップする方が適切です。

サブジェクトがオブジェクトとして見なされることもあります。例えば、通常、プロセスはサブジェクトとして見なされます。 ただし、あるプロセスがもう 1 つのプロセスに影響を与えようとすると、通常、そのもう 1 つのプロセスはこの操作に対するオブジェクトとして見なされます。

要求

要求はトラステッド・モジュールがサブジェクトのために実行する一連のアクションです。 それぞれの要求は、要求の入力、可能な出力、および結果について、すべての副次作用を含めて明確に識別する必要があります。 すべての要求を正確に識別することは、セキュリティー・ポリシーの定義のための重要な準備です。

セキュリティー・ポリシー

セキュリティー・ポリシーには、指定のサブジェクトに代わって指定のオブジェクトに関する要求が実行されることを示す単純ステートメントがあります。 サブジェクト、オブジェクト、および要求は慎重に定義する必要があり、セキュリティー・ポリシーは簡潔で分かりやすくなければなりません。監査目的のために使用される要求サブジェクトとオブジェクトの ID を指定することは重要です。

最小特権:

最小特権のプリンシパルでは、ソフトウェア・モジュールに対して、目的のタスクを実行するのに必要な最低限の機能を与える必要があることを示しています。

最小特権には、トラステッド・プログラムがその機密機能を可能な限り少ないプログラム領域で使用できるように任意に制限するプリンシパルがあります。 最小特権は、ソフトウェア・エラーや予期しない副次作用による損害を削減するのに役立ちます。 すべてのトラステッド・ソフトウェアは、最小特権のプリンシパルに応じて設計する必要があります。

特権の割り当ておよび除去:

1 つのトラステッド・ソフトウェア手法として、プログラムが実行の初期段階で特権が必要になるすべての操作を実行し、次に、残りの操作期間は特権を解放する方法があります。これを「特権の囲い込み (privilege bracketing)」と言います。

特権の使用に関連する以下の考慮事項を覚えておいてください。

- 各ユーザーのプロセスには、プロセスの実行時に最大特権セットが割り当てられる。 この特権セットはいつでも減らすことはできますが、特権を持たないユーザーは増やすことができません。
- 特権操作を行う時に最大セットの特権を有効セットへ増やしたり、有効セットから減らしたりすることは実行プロセスの責任です。

- プロセス特権はプロセスが空でない固有の特権セットを所有している実行可能ファイルを実行するときに変更されます。 詳細情報については、『**exec** コマンド』を参照してください。
- また、プロセスは実行されるときに制限特権セットを提供されます。 適切な特権により、プロセスは最大セットにある特権数を制限セットの特権数まで増やすことができます。

一時的な *MAC* ラベルの変更*:*

プロセスがその通常の操作ラベルから MAC ラベルを変更する必要がある場合、そのラベルの変更の期間はできる限り短くなければなりません。 これを実現するには、ライブラリー・ルーチンを使用します。

これらのライブラリー・ルーチンの詳細については、 542 ページの『Trusted AIX システム・コール』を参照してください。

機密ファイルの一時的なオープン*:*

機密ファイルとは、システム・セキュリティーを脅かす可能性のある情報を含むシャドー・パスワード・ファイルなどのファイルです。 機密ファイルを読み取りまたは書き込み用にオープンする場合、必要な時間だけオープンしてください。

ファイル・ディスクリプターの **close-on-exec** 属性を設定するには、**fcntl** システム・コールを使用します。 これにより、権限のないプロセスは **exec** システム・コールを介してオープン・ファイル・ディスクリプターを継承できません。

機密操作の集中化*:*

機密操作は特権が必要となる操作です。機密操作を特権のないプロセスによって実行されると、システムのセキュリティーを危うくすることがあります。

機密操作は識別可能なモジュール (サブルーチンまたは分離されたプログラム) に限定してください。 大きいプログラムを別々のプログラムに分割することにより、その分割された一部のプログラムは、より少ない特権で済むか、または特権が不要になります。 これにより、システムのセキュリティーを偶然に危うくする可能性を減少させます。

有効ルート・ディレクトリーの使用*:*

プログラムを特定のディレクトリー・ツリー内に限定することができます。これを行うには、プログラムの有効ルート・ディレクトリーをツリーのベース・ディレクトリーに設定し (**chroot** システム・コールを使用)、プログラムの作業ディレクトリーをこの同じツリー内に設定します。 事実上、特権プロセスがアクセスできるファイルをツリー内のファイルに制限するため、これは最小特権メカニズムです。 これは親 (トラステッド) プロセスがトラステッドまたは非トラステッドの子プロセスを制限する場合に特に有効です。

ルート・ディレクトリーを変更すると新規ルート・ツリー外のファイルが保護されますが、潜在的なセキュリティー問題が生じます。 ルート・ディレクトリーを変更することで、これが慎重に行われていないと新規ルート・ツリーのセキュリティーが危険にさらされる可能性が出てきます。 これは新規ルート・ツリー内のランタイム・リンカーと共有オブジェクトが偽造である可能性がある場合に起こります。 この手順は注意深く慎重に使用してください。

保護サブシステムの使用*:*

保護サブシステムは、特殊サブシステムのための保全性の保護を備えています。サブシステムは、同じユーザー ID またはグループ ID、あるいはその両方により所有される、システムに特定の機能を実装するために使用されるプログラムまたはデータ・ファイル、あるいはその両方の集合です。

サブシステムには、setuid プログラムまたは setgid プログラムが組み込まれることがあります。 保護サブシステムは、システム・ユーザー ID のユーザー ID を持つサブシステムです。

システム・ユーザー ID は、127 以下の値のユーザー ID です。 ユーザーはシステム・ユーザー ID ではログインできません。 保護サブシステムを使用すると、特権プロセスの数が著しく削減されます。

最小アクセス・モード*:*

トラステッド・プログラム (実際はすべてのプログラム) は、確実に必要な読み取り/書き込みアクセス・モードのオブジェクトのみをオープンします。 すなわち、基本的に、読み取り用のオープンで十分なときに、書き込みと読み取り用のオブジェクトのオープンは行いません。 特に機密性の高い状況の場合、プロセスは書き込みが必要な特定のロケーションで書き込み専用にオープンします。

この手法はプログラムで他のプロセスを作成する場合に特に重要です。これは特権の引き渡しと他の一般機能 (機密ファイルへの接続のオープンなど) がトラステッド・ソフトウェア設計の重要な側面であるためです。 特権はすべての制限を無効にできます。 特権を持つコマンドを新規に作成する場合は、慎重に検討して設計を行う必要があります。

その他のトラステッド・プログラミング規則*:*

Trusted AIX はさらに多くのトラステッド・プログラミング規則を使用します。

冗長*:*

冗長はセキュリティー・システムに対して有効な手法です。 セキュリティーは絶対ではありませんが、ほとんどの場合、システムへの不正なアクセスを行おうとするパスに十分な数の障壁を設置することが重要です。

冗長セキュリティー検査の利点は、検査が一度失敗するか障害が起きても、別の検査で保護できることです。 冗長検査の欠点は、セキュリティー検査全体がシステムで分離または分散されることです。 このため、冗長検査は非常に有効ですが、慎重にその設計、文書化、および維持を行う必要があります。

カーネル検査の重複なし*:*

カーネルが実行できる検査をプロセスが実行することは、あまり推奨されません。 例えば、プロセスはファイルの MAC ラベルを読み取らず、必須のアクセス検査をそれ自体で実行します。 可能な限り、カーネルを検査することで検査を行います。

カーネルが検査を実行しなければならない主な理由が 2 つあります。

- カーネル操作は他のプロセスに対してアトミックである一方、プロセス検査は他のプロセスと事実上コンカレントとなることがあります。
- さらに重要なことに、使用されるアルゴリズムは、新しいバージョンのカーネルで変更される可能性があります。 エンド・ユーザーのソフトウェアの一部であるアルゴリズムに対するこのような変更を追跡することは困難です。

直接の特権検査*:*

プログラムが特権プロセスとして起動されたかどうかについて、そのプログラムで判別を試行 (例えば、有効または最大特権ベクトルの検査) しないようにしてください。 その代りに、プログラムは適切な特権が割り当てられているものとして起動されることを前提にする必要があります。

プログラムが特権プロセスでないと、特権システム・コールは失敗し、このプログラムは該当するアクションをとることになります。 プログラムに特権が付与されていない場合に、プログラム自体が特定の操作の実行を拒否することは、通常、有効なセキュリティー手段ではありません。 プログラムに特権が付与されている場合、検査する意味がありません。 プログラムに特権が付与されていない場合、このプログラムはその他の特権のないプログラムよりひどく危害を加えることはできません。

しかし、この検査は不注意による誤用を救済する手段として有効に使用できます。プログラムに特権を付与しようとして付与されないことを述べる重要なエラー・メッセージを出すことができます。

機密機能の伝搬*:*

機密機能は、非トラステッド・プログラムに提供された場合にシステムのセキュリティーを妥協して解決するトラステッド・プログラムの機能です。

特権プログラムがその特権または一般的な機能を、システム・コールの **fork** および **exec** ファミリーを使用して他のプログラムに伝搬する時には注意しなければなりません。 **exec** システム・コールは、あるプログラムから別のプログラムに特権を渡すので、最も重要です。 **fork** システム・コールは、新しいプロセスを作成しますが、新しいプロセスの特権は親の特権と同じです。 基本的な危険は、実行可能プログラム・ファイルがトラステッドでない可能性があるか、または非トラステッド・プログラムによって変更されている可能性があることです。 次の注意事項について検討する必要があります。

- トラステッド・プログラムでは、オブジェクト (主としてファイル) へのオープン接続を、ファイルが開かれるモードで子および子孫がファイルに適切にアクセスすることが確実でない限り、子プロセスに渡さないように注意しなければなりません。 モードが他に存在するものより制限されているオブジェクトに新しい接続を渡すのが、プロセスのために最良の方法です。
- 絶対ルート以外の有効ルート・ディレクトリーによって実行するトラステッド・プロセスでは、子プロセスが混乱しないようにしておかなければなりません。例えば、子プログラムがシャドー・パスワード・ファイルのようなトラステッド・ファイルをオープンする時に、子プログラムは、有効なルートが絶対ルートであるという前提のもとで絶対パス名を使用することができます。
- トラステッド・プログラムが、より制限されている umask を子に課す必要が生じる場合もあります。
- 多くのプロセス属性は、子プロセスによって継承されます。 トラステッド・プログラムで、子プロセスが非トラステッドであり、トラステッド・プロセスのラベルより上位でない MAC ラベルをもち、これらの属性が非トラステッドの祖先からトラステッド・プログラムによって継承されていることが知られている場合には、これらの属性は潜在的な隠れチャネルのソースとなることができます。
- **fork** および **exec** システム・コールに関する特権伝搬のルールに注意してください。 **fork** システム・コールが起こると、親プロセスの特権は子プロセスの特権になります。 特権は、**exec** システム・コールの間に変更されます。

特に機密度の高い状態では、トラステッド・プログラムによるトラステッド・ファイルへのアクセス制御を検討して、非トラステッド・プログラムによる変更からファイルを適切に保護することができます。 例えば、ファイルの所有者に許可されている最大限の DAC 書き込み許可を用いて、ファイルがルートに所有されるようにすることができます。

有効ルート環境*:*

トラステッド・プログラムは、正しい絶対パス名に依存しています。例えば、**login** プログラムは、正しいシャドー・パスワード・ファイルである `/etc/security/passwd` ファイルに依存しています。

このファイルにはデータ・ファイルだけでなく、トラステッド・プログラムの実行可能ファイルも収められています。 プログラムの有効ルート・ディレクトリーを直接変更するために、非トラステッド・プログラ

ムが **chroot** システム・コールを使用できない場合、 TCB により非トラステッド・プログラムが有効ルート下で実行できる状態になっている可能性があります。このように非トラステッド・プログラムが絶対パス名に依存するトラステッド・プログラムを実行できる場合は、セキュリティー問題が発生する可能性があります。

実 ***ID*** および有効 ***ID*** による認証**:**

トラステッド・プログラムにはプロセスに関連付けられた、いくつかのユーザーおよびグループ ID を使用することが必要になる場合があります。 これらの ID とそれらに該当する使用目的との違いを理解することが大切です。

実ユーザーおよびグループ **ID**

通常、実ユーザーおよびグループ ID はプロセスが作成されたログイン・セッションのログイン ID を表します。一部のケースでは、実 ID (特に実ユーザー ID) はセキュリティー判断に使用できます。 そのようなインスタンスでは許可検査を行います。 実ユーザー ID が ID 検査のフォームによるコマンドで使用されます。 これは、 **setuid-on-exec** または **setgid-on-exec** 制御ビットを、悪意ある使用または不注意による使用から保護する場合に、特に役立ちます。 ただし、実 ID の検査は標準の UNIX 手法からは逸脱しているため、必要なときだけに行うようにしてください。 UNIX システムで全体の原則は、アクセスおよびその他の関連セキュリティー検査には有効 ID が使用されることです。このように承認済みの方式から逸脱する場合、注意深い検討と文書化なしでは行わないようにしてください。

有効ユーザーおよびグループ **ID**

有効ユーザーおよびグループ ID はすべてのアクセス制御判断 (DAC および MAC) で使用する必要があります。 システム・ユーザーは 0 から 127 までの範囲のユーザー ID を持っています。 通常のユーザーの ID 値は 128 以上です。

トラステッド・コマンドの絶対パス名**:**

セキュリティー侵害行為の例として、偽のトラステッド・プログラムを作成し、そのプログラムの中に、管理者または通常のユーザーも使用するシェルのようなプログラムの検索パスを組み込もうとする場合があります。 例えば、既存または新規のユーザー・パスワードを取り込むために、 **passwd** コマンドの偽のコピーが使用される可能性があります。

このようなセキュリティー侵害を防ぐためには、管理上の実施項目として、現在の作業ディレクトリーを検索パスから除去することが適切です。 しかし、必ずしも強力に保護されていない他の検索パスがある可能性があり、通常のユーザーには現在の作業ディレクトリーをその検索パスに置くことを許可する必要があります。 有効な対策は、トラステッド・プログラムは必ず絶対パス名 (例えば、`/usr/bin/passwd`) で起動することです。トラステッド・プログラム自体は、その最初の呼び出し引数と名前を検査します。 そこで、適切な絶対パス名が使用されていなければ、トラステッド・プログラムは実行を拒否します。 さらに、トラステッド・プログラムは、絶対ルートとは異なる有効ルート・ディレクトリーを持っていないことも確認する必要があります。

注**:** このように、ユーザーが絶対パス名を使用するように訓練されている限りは有効です。 しかし、そうではなくて、ユーザーが不注意で相対パス名を使用して、偽のプログラムが起動されると、セキュリティー侵害行為を防ぐことはできません。

ディレクトリー・ツリーの構造化*:*

ディレクトリー・ツリーは重要なファイルの保護を強化するために、注意して構造化する必要があります。基本的なガイドラインは、ディレクトリーの検索アクセスは可能な限り制限する必要があるということです (例えば、すべての共用アクセス可能ファイルは、ファイルシステムのルートに近いディレクトリーに配置します)。

さらに、可能な限り絶対ルートに近いところに非常に重要なディレクトリーを配置するのはよい考えです。それにより保護する必要のある中間ディレクトリーの数を削減することになるからです。

読み取り専用ファイルシステム*:*

ディレクトリー・ツリー構造の最終地点は、稀にしか変更されないトラステッド・ファイルがそれ自体のファイルシステムに置かれ、読み取り専用としてマウントされる場所です。 これにより、事実上、その内容は通常のシステム操作では変更されません。 この手法は、トラスト・プログラムの膨大な実行可能ファイルに対してよく使用されます。

ファイルの変更が必要な場合は、さらに保護された状況 (単一ユーザー・モードやさらに保護された別のマシンなど) でファイルシステムを書き込み可能として再マウントできます。 このような更新の後は、正しい構成 (適切な DAC、MIC、MAC ラベルなど) を得るために、プログラムを使用してファイルシステムを走査することをお勧めします。

また、読み取り専用ファイルシステムでは、DAC、MIC、および MAC の情報は変更できません。 ファイルシステムを適切に構成すると、DAC 情報、MIC および MAC ラベルのいずれか (またはその両方) の変更を試みるセキュリティー侵入方式に対して保護することができます。

パスワードの処理*:*

標準システム・ユーティリティー以外のプログラムでは、通常、ユーザーに対するログイン・パスワードの照会は適切ではありません。 パスワードは非常に機密性の高い情報であり、その処理はできる限り少ない既存の信頼性の高いシステム・ユーティリティーに制限されなければなりません。

特定のトラステッド・サブシステムがそれ自体の固有パスワードを実装することが適切である場合があります。 ただし、システム強制メカニズムほど安全ではないため、このような秘密のパスワード方式に依存するのは危険な可能性があります。

トラステッド・コンピューティング・ベース *(TCB)* の保護*:*

TCB のエレメントを保持するファイルは、非トラステッド・プログラムによる変更、また場合によっては開示 (読み取り) から保護しなければなりません。

変更からの保護は重要であり、開示からの保護も重要になることがあります。 以下のファイルは保護する必要があります。

- セキュリティー決定を行う際にトラステッド・プログラムで使用されるデータを含むすべてのファイル (シャドー・パスワード・ファイルなど)
- トラステッド・プログラムのすべての実行可能ファイル
- TCB の部分へのアクセスを許可する疑似ファイル (`/dev/kmem` など)

注*:* システム初期設定ファイル (rc ファイル) は、TCB の一部として特に保護する必要があります。

変更からの保護*:*

権限のない変更からの保護は、主に DAC 情報を適切な値に設定することで行います。 通常、これらのファイルは、ファイルの所有者のみに許可された書き込みアクセスを持つシステム・ユーザー ID が所有します。

MIC はオブジェクトの保全性を保護することで変更から保護するように設計されています。 高い MIC ラベルをファイルに設定することで、低い MIC ラベルのプロセスがファイルを変更、削除、名前変更するのを防止します。 ファイルの不要な変更を防止するには、この方法が適しています。

権限のない変更から保護するために、MAC を使用することがあります。ただし、MAC は開示 (読み取り) からの保護のみを行うように設計されており、変更からの保護には適していません。 MAC 基本ポリシーでは、サブジェクトによる高いラベルのオブジェクトの変更を禁止していません。 直接ファイル書き込みの場合は許可されていませんが、特定のトラステッド・サブシステムではこれを許可している場合があります。 また、実行可能プログラム・ファイルなどの多くのトラステッド・ファイルは、一般にアクセスできるように低い MAC ラベルで保持されなければなりません。 このため、高い MAC ラベルをファイルに設定することが必ずしも適しているとは限りません。

ファイル・セキュリティー・フラグを使用しても、ファイルの変更から保護できます。 一部のファイル・セキュリティー・フラグは、特権サブジェクトによるオブジェクトの変更をも防止します。 **FSF_TLIB** ファイル・セキュリティー・フラグがファイルに設定されている場合、そのファイルを変更できるのはシステムが構成モードである場合のみです。この場合、**trustedlib_enabled** カーネル・セキュリティー・フラグがオンになっていることを前提とします。 ファイルに **FSF_TLIB** を設定するには、プロセスでその EPS 内に **PV_TCB** 特権が必要です。 もう 1 つの関連ファイル・セキュリティー・フラグは、**FSF_APPEND** フラグです。このフラグは、前に書き込まれたデータの変更を防止します。 **FSF_APPEND** フラグ・セットのあるファイルでは、データを追加することのみできます。 これはファイルにレコードを記録するアプリケーションに有効です。

これらのフラグは、通常、プログラム制御ではなく、インテグレーターがファイルに対して設定します。プログラマーは、これらのフラグとその機能を認識している必要があります。

開示からの保護*:*

DAC と MAC を使用して読み取りアクセスから TCB ファイルを保護できます。これらのファイルの MAC ラベルは、これらのファイルの情報の機密性を正確に反映しなければなりません。 例えば、特定のアルゴリズムが機密に分類される場合、そのアルゴリズムを使用するプログラムの実行可能ファイルの MAC ラベルは適切に設定する必要があります。

開示からデータを保護するために、MAC ラベルを人為的に高く (ファイル内のデータの実際の機密区分よりも高く) 設定することが可能です。 ただし、このような拡大した機密区分は慎重に使用しなければなりません。

多くの場合、ファイル自体を適切に保護するには、絶対ルートからのディレクトリー・チェーン全体を保護する必要があります。 これを保護しないと、悪意のあるプログラムによりディレクトリー・チェーンの部分がリンク解除され、ファイルの偽コピーでサブツリーが新規に作成される可能性があります。

例えば、トラステッド・ファイルが `/A/B/foo` に保管されているとします。 **foo** は変更から保護されている一方で、ディレクトリー **B** は保護されていません。 この場合、悪意のある非トラステッド・プログラムにより **B** にある **foo** へのリンクが除去され、古いファイル **foo** の偽コピーで新しいファイル **foo** が作成される可能性があります。 `/A/B/foo` を開くトラステッド・プログラムは偽ファイルを開き、その偽データを知らずに騙されて使用してしまいます。

トラステッド・プログラムは、TCB ファイルにアクセスするために正しいパス名に依存します。 このため、TCB ファイルのパス名で使用されるシンボリック・リンク・ファイルは、ファイルと同様に強力に保護する必要があります。

権限のない開示から保護するために、MIC を使用することがあります。ただし、MIC は主に変更 (書き込み) からの保護のみを対象としており、開示からの保護には適していません。

機密ラベルの操作*:*

種々の機密ラベルを持つサブジェクトまたはオブジェクトに関する状態のためのトラステッド・プログラムのガイドラインがあります。

機密ラベルの形式とラベル間の支配関係を理解している必要があります。 より高くなることは「支配する (上位にある)」ことを意味し、より低くなることは「支配される (下位にある)」ことを意味します。一方で、アップグレードすることはデータの機密区分を高いラベルに上げることを意味し、ダウングレードすることはデータの機密区分を低いラベルに下げることを意味します。

基本 ***MAC*** 制約*:*

基本必須アクセス制御には、非トラステッド・サブジェクトでは機密ラベル A でラベル付けされているデータを機密ラベル B でラベル付けすることは (B が A より上位にならない限り) できないという制約があります。

基本 MAC 制約はすべてのクラスのデータに及びます。 データの再ラベル付け (つまり、データ・コンテナーのラベルの変更) およびデータ・コンテナー間のラベル付けされたデータの移動についての制限もこれに含まれます。

システムのさまざまなレベル (システム・コール、システム・サービス・ユーティリティーなど) で、この基本的な制約はより具体的な一連のルールとなりますが、常に同じ基本的な考え方でデータのアップグレードまでは行うことができます。例えば、基本的な制約として、プロセスの機密ラベルがオブジェクトの機密ラベルより上位の場合は、プロセスは大きなクラスのいずれのオブジェクトも読み取り用にオープンすることができ、オブジェクトの機密ラベルがプロセスの機密ラベルより上位の場合は、書き込み用にオープンすることができます。

通常ファイルの場合、書き込み操作はより制限され、プロセスと同じ機密ラベルのファイルに対してのみ許可されます。 ディレクトリーおよびデバイスの場合は、サブジェクトの SL (機密ラベル) がオブジェクトの最小 SL より上位であり、さらにオブジェクトの最大 SL がサブジェクトの SL より上位である場合に、書き込み操作を行うことができます。 FIFO スペシャル・ファイル (名前付きパイプ) の場合は、読み取り操作は、隠れチャネルの理由のためプロセスと同じ機密ラベルで FIFO スペシャル・ファイルにも制限されています。

データをより高い機密ラベルに移行することができますが、これはオブジェクトや状況によっては必要ありません。 例えば、オペレーティング・システムは本来、非特権プロセスが自分より高いラベルのファイルを書き込み用にオープンすることを許可しませんが、基本 MAC 制約では許可されます。 このアップグレードを非トラステッド・サブジェクトに許可するかどうかは、設計と方針の問題です。このアップグレードが役立つ場合もあり、役立たない場合もあります。 例えば、機密ラベルの高いファイルへの直接書き込みを許可しないのは、プロセスがこれらのファイルを読み取れないからであり、読み取りなしでの書き込みは意味がなく、役立つという以前の問題だからです。 しかし、非トラステッド・サブジェクトの要求に応じてファイルのラベルを高くした単純トラステッド・ユーティリティーは、許容できる役に立つユーティリティーだと言えます。

システム・コール・レベルでは、非特権プロセスにのみ制限があります。つまり、特権プロセスはこの制約に束縛されないということです。 しかし、実際にはトラステッド・システムで実行されるすべてのサービスは非トラステッド・ユーザーに合わせて設計されており、そのため、ユーザー・サービスのレベルでは制約が優先されます。

基本 MAC 制約は、非トラステッド・プログラムがデータ転送の処理で使用するすべての手段に適用されます。 ただし、基本 MAC 制約は、多くの場合、2 つのコンポーネントに分けられます。 最初のコンポーネントは、データ転送 (またはラベル付け) を目的としたオペレーティング・システムの各機能にのみ対応します。 これらの機能には、例えばファイルの読み取りおよび書き込みやプロセス間データ通信が含まれます。 もう 1 つのコンポーネントは、前記に当てはまらない通信の手段に対応します。これは隠れチャネルと呼ばれます。 隠れチャネルに関して基本 MAC 制約を完全に実施することは、ほとんど不可能です。 このために、他の要因に対して妥当なトレードオフがある場合に限り、低データ転送速度 (例えば、0.1 ビット/秒) の隠れチャネルが存在できます。

基本 MAC 制約は、直接的で単純であり、マルチレベル・データの処理に関する詳細なガイドラインの数は比較的少ないです。

マルチレベル操作*:*

sec_setplab システム・コールにより、特権プロセスはそのプロセス・ラベルを任意に変更できます。

非特権プロセスのほぼすべての MAC と MIC 制約が既存のシステム・コール (基本オペレーティング・システムで定義されているもの) の特権プロセスにも実行されるため、マルチレベル操作を実行する必要がある特権プロセスは **sec_setplab** システム・コールに大きく依存しなければなりません。 ただし、トラステッド・プログラムは、次の方法のみで sec_setplab() のみを使用します。

- マルチレベル操作 (読み取り用のより高いラベルのファイルのオープンなど) を実行する **sec_setplab** システム・コールの使用はすべて、実行される実際の高レベル操作のセマンティクスを反映し、**sec_setplab** システム・コールの詳細な使用を非表示にするライブラリー・ルーチンを使用してのみ行います。
- 唯一の例外は、非常に単純なプロセス・ラベルの変更であり、これは広範囲なマルチレベル操作の一部ではありません。 このような単純な操作では、**sec_setplab** システム・コールを直接使用できます。

sec_setplab システム・コールに対するこのようなガイドラインには 2 つの理由があります。 まず、**sec_setplab** システム・コールなどの機密性が高く危険な可能性のある機能は、適切に設計されたモジュラー方法でのみ使用しなければなりません。 次に、トラステッド・システムの標準が進化すると、低レベルのシステム・コールがマルチレベル操作に対してさまざまなメカニズムをサポートする可能性があります。

高レベル操作をライブラリー・ルーチンにカプセル化すると、オペレーティング・システムの新バージョンに対して最適な上位互換性と適応性がもたらされ、UNIX システムのトラステッド・バージョン間で移植性が確実になります。

トラステッド・システムは、このようなルーチンの基本セットを備えています。 可能な限り、これらのルーチンを使用してください。 このルーチン・セットは、オペレーティング・システムの以降のバージョンで拡張されます。 トラステッド・システム・プログラマーは、このようなライブラリー・ルーチンを必要に応じて作成することもできます。

MAC と MIC 制約に対するもう 1 つの例外は、MAC または MIC 制限をバイパスするために 1 つ以上の使用可能 MAC または MIC 特権を使用することです。これらの特権の使用を許可する場合は注意が必要です。

System V* プロセス間通信 *(IPC):

プロセス間通信 (IPC) メカニズム (メッセージ・キュー、セマフォー、および共有メモリー) は、DAC、MIC、および MAC 制限に従います。 通常、System V IPC オブジェクトを作成し、使用するためのコマンドはありません。

AIX IPC 関連システム・コールは変更され、Trusted AIX に対してマルチレベル認識となりました。以下のシステム・コールが変更されました。

- **msgget**
- **msgsnd**
- **msgrcv**
- **msgctl**
- **semget**
- **semop**
- **semctl**
- **shmget**
- **shmctl**
- **shmat**
- **shmdt**

また、IPC オブジェクトの MAC 属性を操作するために特別に設計された以下のシステム・コールが、Trusted AIX に対して追加されました。

sec_getmsgsec
メッセージ・キューのセキュリティー属性を取得します

sec_getsemsec
セマフォーのセキュリティー属性を取得します

sec_getshmsec
共有メモリー・セグメントのセキュリティー属性を取得します

sec_setmsglab
メッセージ・キューのセキュリティー属性を設定します

sec_setsemlab
セマフォーのセキュリティー属性を設定します

sec_setshmlab
共有メモリー・セグメントのセキュリティー属性を設定します

IPC オブジェクトを操作するプロセスの特権要件については、『IPC オブジェクトへのアクセス』を参照してください。 IPC 属性を操作するために、**settxattr** コマンドを使用できます。

高位レベル・インプリメンテーションおよび高位レベル・システム ***MIC*** ラベルおよび ***MAC*** ラベル*:*

トラステッド・プロセスには、システム上でその他のすべてのラベルより優位にある MAC ラベルを判別することが、頻繁に必要になります。 使用可能な 2 種類の MAC ラベル (高位レベル・インプリメンテーション MAC ラベル、高位レベル・システム MAC ラベル) があります。

高位レベル・インプリメンテーション MAC ラベルは、Trusted AIX によってサポートされる最高位レベル MAC ラベルです。 このラベルは階層型の分類のようになっており、サイトには使用されていないカテゴリーが含まれています。 このラベルの生成は簡単ですが、ラベルの使用には注意が必要です。 このラベルではいずれのプロセスも作成されません。

高位レベル・システム MAC ラベルはサイトに使用されている最高位の MAC ラベルです。 これは管理者によって **LabelEncodings** ファイルに定義されます。

高位レベル・システム MAC ラベルの使用はそれほど効果的ではありませんが、特権プロセスであっても、管理者が **LabelEncodings** ファイルに適切なパラメーターを設定することにより、アクションを効果的に制約することができるため、使用されることを強くお勧めします。

MIC は類似の高位レベル・インプリメンテーションおよび高位レベル・システム・ラベルを持っています。

ユーザーおよびシステムのログイン範囲*:*

ユーザーに対してサービスを実行するトラステッド・プログラムは、これらの操作に関係する MIC ラベルと MAC ラベルを、ユーザーがログインできる値またはシステム全体で許可されたログイン・ラベル、あるいはその両方に制限しなければならない場合があります。

システムでユーザーに割り当てられる認可はユーザー・データベース・ファイル `/etc/security/user` にあり、**getuserattr** と **getuserattrs** ライブラリー・ルーチンを使用してアクセスされます。

Trusted AIX では、ユーザーはシステム認定範囲内にあり、ユーザーの最大認可により支配されて tjat がユーザーの最小認可を支配するラベルのシステムで操作できます。 ユーザーが種々のラベルで操作できるプログラムでは、必ず新規ラベルがそのユーザーに対して有効でなければなりません。

例えば、**upgrade** という名前のユーティリティーが、ユーザーの要求時にファイルの MAC ラベルを上げるように定義されたと仮定します。 基本的な MAC 制限では、**upgrade** は MAC ラベルがユーザーのもので支配されているファイルのみを受け入れるように要求しています。 さらに、新規ラベルをユーザーがログインできるものにして、各ユーザーとシステム全体の両方のラベル範囲の制限を組み込むことが賢明であると判断されます。ただし、基本的な MAC 制限からは必ずしも必要ではありません。 **upgrade** ユーティリティーは、**sl_cmp** と **accredrange** の両方のインターフェースをこの目的に使用します。

ディレクトリー・ツリー構造*:*

システム・コールは、非特権プロセスで作成されたディレクトリー・ツリーが非減少ラベル構造に従うように機能します。ここでは、ファイルのラベルがその親ディレクトリーのものと同等であるか、分割ディレクトリーの範囲内にあり、ディレクトリーのラベルはその親ディレクトリーのものを支配します (支配には同等も含まれることに注意してください)。 これは非トラステッド・プログラムでは通常の構造です。

ただし、特権プロセスはこの制限によって束縛されず、親ディレクトリーの MAC ラベル関係が任意であるディレクトリー・ツリーを作成できます。 MAC 検索アクセスはツリーのルート近くでは制限されるため、このような構成は有効です。 例えば、データ・オブジェクトの集合の MAC ラベルがオブジェクトの単一ラベルよりも高い集約保護を、エレメントよりも高いディレクトリーの MAC ラベルを設定することで実装できます。 非トラステッド・プロセスは、データの集約へのアクセスを得るために、ディレクトリーのラベルを支配しなければなりません。

減少ラベルを持つディレクトリー・ツリーを作成する場合は、慎重に行う必要があります。 非特権プロセスは、ファイルがその親のラベルを支配していないか同等でない場合は、そのファイルを書き込み用にオープンすることはできません。

分割ディレクトリーの操作*:*

分割ディレクトリーのインプリメンテーションの結果、異なる動作をするシステム・コールがいくつかあります。

以下のシステム・コールでは、分割ディレクトリーのインプリメンテーションの結果として、それぞれの動作が異なってきます。

- getdirents
- link
- mkdir
- mount
- rename
- rmdir
- stat
- lstat
- fstat

処理モード*:*

pdmode は指定のモードでコマンドを実行できます。 プロセスは **setppdmode** システム・コールを使用して、そのモードを実モードまたは仮想モードに設定できます。 **setppdmode** システム・コールには、**PV_PROC_PDMODE** 特権が必要です。 別のプロセスのモードを変更するプロセスのためのメカニズムはありません。

ディレクトリー・タイプ*:*

通常のディレクトリーを分割ディレクトリーに変更するには **pdset** コマンドを使用できますが、分割ディレクトリー (または分割サブディレクトリーか分割サブサブディレクトリー) を通常のディレクトリーに変更するコマンドはありません。

分割ディレクトリーを作成するには、**pdmkdir** システム・コールを使用することもできます。 **pdmkdir** システム・コールには、**PV_FS_PDMODE** 特権が必要です。

MIC および ***MAC*** ラベルに関する考慮事項*:*

MIC ラベルと MAC ラベル間の関係を判別するには、すべてのプログラムで **sl_cmp** 関数と **tl_cmp** 関数のみを使用する必要があります。

内部ラベルのフォーマットが以降のシステム・バージョンで変更される可能性があり、これらのライブラリー・ルーチンは新しいフォーマットを追跡するため、これは非常に重要です。同様に、MIC ラベルと MAC ラベルを操作する他のライブラリー・ルーチンは多くあり、可能な限りこれらを使用する必要があります。

setea、**lsetea**、および **fsetea** システム・コールは、ファイルの MIC ラベルまたは MAC ラベルを変更します。 **fsetea** システム・コールは、ファイル・ディスクリプターを受け入れます。

デバイス・ドライバー*:*

Trusted AIX システムのデバイス・ドライバーを作成する場合は、いくつかの原則とガイドラインがあります。ユーザーは、基本システムのデバイス・ドライバーの作成のためのメカニズム、およびこれらのメカニズムの使用に関する予防措置に精通している必要があります。

デバイス管理サブシステム*:*

AIX システムでのデバイスは抽象であり、デバイスのスペシャル・ファイルの参照によりアクセスされるすべてのデータ・オブジェクトをカバーするために使用されます。 あるケースでは、これらのデータ・オブジェクトは実際の物理デバイスを表し、あるケースでは (データ・ストレージ・オブジェクトがまったくない `/dev/null` などのケースを含めて)、まったく異ります。 後者のインスタンスは疑似デバイスとして頻繁に参照されます。

Trusted AIX システムは 2 つのデバイス・タイプを提供します。単一ラベルのデバイスとマルチレベルのデバイスです。 マルチレベルのデバイスはトラステッドで、一時点に複数の機密レベルでデータを処理します。 単一ラベルのデバイスは、通常、非トラステッドです。 通常、データ上のラベルは、マルチレベルのデバイスが取り扱う情報と、データが常に正しくラベル付けされる方法で関連付けられます。 単一ラベルのデバイスは、通常、外部のラベル付けに依存します。

ハード・ディスクは、マルチレベルのデバイスの一例です。 ハード・ディスクに保管されているデータのすべては、関連付けられた機密ラベルを持っています。 デバイスにアクセスするためにはセキュリティー上の認可を必要とするような環境に、物理的に設置されているプリンターは、単一ラベルのデバイスの一例です。 そのプリンターには、認可を受けたデータのみを送信できます。

デバイス・ドライバー開発の注意事項*:*

デバイス・ドライバーはオペレーティング・システムのカーネルの一部であり、そのためにデバイス・ドライバーのアクションは制限されません。 デバイス・ドライバーの作成または変更はカーネル自体の変更と同じくらい細心の注意を払う必要があります。 残念ながら、ユーザーには、デバイス・ドライバーの作成または変更が必要になる場合が少なくありません。この場合は、細心の注意を払って行ってください。

ドライバーが (時にはまったく悪気なく) システムのセキュリティーを妨害してしまう可能性もあり、デバイス・ドライバーの作成の際に注意すべきことを具体的にすべてを挙げることは不可能です。 したがって、機密保護機能のあるデバイス・ドライバーの作成には、デザイナーの判断と経験に多くのことが残されています。

デバイス・ドライバーはデバイス管理のみ実行し、それ以外は何も行いません。これに従って、基本的には、新規のシステム・コールをシステムに追加する目的で作成するデバイス・ドライバーは、 `/dev/kmem` のような疑似デバイス・ドライバーも含めて、新規のシステム・コールを考慮に入れて設計する必要があります。 このセクションでのガイドラインは、基本的には、正規のデバイス・マネージャーである、そのようなドライバーについて言及します。

ユーザーは新規デバイス・ドライバーの作成を始める前に、標準のデバイス・ドライバーについて学習してください。 デバイス・ドライバーの基本的なセキュリティー・アクションは、**open** および **ioctl** システム・コールの実行により、デバイス・ドライバーを組み込むことです。

デバイスのオープン*:*

多くのシステム・オブジェクトと同様に、デバイスのアクセスに関連付けられたセキュリティー検査の多くは、デバイスが **open** システム・コールによって開かれるときに実行されます。

カーネルは最初に基本操作のセットを実行し、次にオープン要求の処理をデバイス・ドライバーに渡します。 カーネルはデバイス・ドライバーに制御を渡す前に、以下のセキュリティー検査を行います。

- プロセスがデバイス・スペシャル・ファイルへの MAC アクセス権限を持っていない場合は、オープンを失敗にする
- プロセスがデバイスのスペシャル・ファイルへの MIC アクセス権限を持っていない場合は、オープンを失敗にする
- プロセスがデバイスのスペシャル・ファイルへの DAC アクセス権限を持っていない場合は、オープンを失敗にする

多くのデバイスでは、デバイスからの (**read** システム・コールによる) 読み取りは、 MAC ラベルが読み取りプロセスより優位にない別のプロセスによって検出できるやり方で、デバイスの状態を変更します。 これにより隠れチャネルが構成されます。 実際には、先入れ先出し (FIFO) のデバイスがこの問題の対象になります。 これらのケースでは、読み取り権限を同じ MAC ラベルでデバイスとして存在するプロセスへの読み取り権限を制限することが、一般的な方法です。 デバイス・ドライバー内で、この検査が行われます。

非標準デバイスの設計には、いくつかのルールまたはガイドラインがあります。 ユーザーは必須および任意アクセス制御の基本原則を理解して適用する必要があります。 幸いにも、多くのデバイス・ドライバーは正規のデバイスとして構成することが可能であり、非標準デバイスの特殊な機能をそれほど頻繁に扱う必要はありません。

デバイス・ドライバー・オープンの例*:*

以下は標準システム・デバイス・ドライバーを利用する非標準デバイス処理の例です。 これらの例は、そのようなデバイス・ドライバーの考えられる相違について説明することを目的としています。

/dev/null

`/dev/null` はデータ・コンテナーを持っていない疑似デバイスです。 `/dev/null` に書き込まれたデータは破棄され、読み取り要求に対して、常にファイルの終わり (EOF) が返されます。したがって、オープンで必要な MAC デバイス制限はありません。 互換性のために、必ずしも必須ということではありませんが、`/dev/null` デバイス・ファイルに DAC アクセス権限が必要です。

/dev/tty

プロセスが `/dev/tty` のオープンを実行するときに、デバイス・ドライバーは要求プロセスの制御端末である端末装置を実際にオープンしようとします。 このために、`/dev/tty` の代わりにプロセスの制御端末処理について、 MIC、MAC、および DAC アクセス権限を検査する必要があります。互換性のために、必ずしも必須ということではありませんが、`/dev/tty` に DAC アクセス権限が必要です。

ioctl の制限*:*

すべてのデバイス・ドライバー・インターフェース機能はトラステッドでなければならず、通常、**ioctl** インターフェースには特別な注意が必要です。

一般に、書き込み権限を持つプロセスのみが、書き込み権限を持たないその他のプロセスによって検出できるファイルの特性を変更できます。書き込み権限を持っているということは、書き込みのためのファイル・オープンの権限を持っているか、あるいはプロセスの MAC ラベルがデバイスのラベルに等しいということです。この制限は、より低位の MAC ラベルでプロセスによって検出可能なアクションを実行できるプロセスがないという、基本的な MAC 制限に起因します。

アクションの目的がユーザー・データの読み取り/書き込み操作であれば、上述の制限に従う必要があります。 したがって、制限に従わない場合は隠れチャネルと見なされ、処理能力を制限すること、および/または監査可能にすることが必要になります。

デバイスがトラステッド・デバイスとして構成されていない場合でも、一部のデバイス制御アクションには、特権プロセスに制限することが必要になることがあります。

その他の制限*:*

デバイス・ドライバーが特殊なセキュリティー検査を実施することが必要になるその他のケースは、それほど多くありません。

1 つの例として、デバイスでの読み取りの際に、読み取りプロセスの MAC ラベルが優位になることがない MAC ラベルを持つプロセスによって検出できるやり方で、デバイスの状態を変更する例があります。これはデバイス・ドライバー自体によって制限され、監査されることが必要になることがある、潜在的な隠れチャネルであることを示しています。

デバイス・ドライバー・プログラミングの要約*:*

以下のガイドラインはデバイス・ドライバーを実装する場合に考慮する必要があります。

注**:** 新規システム・コールが追加され、ストリーム・デバイスおよび FIFO デバイスの各読み取り/書き込みに対して、拡張セキュリティーがサポートされます。 2 つの新規ライブラリー API の eread() と ewrite() がこの拡張セキュリティー属性をサポートします。 MLS カーネルの場合は、セキュリティー・フラグの DEV_SEC_ERDWR がデバイスに設定されます。 同様に、FIFO の場合は、GNF_SEC_ERDWR がデバイスに設定されます。 これらのフラグは各読み取り/書き込みに関する追加のセキュリティー検査を使用可能にします。

一般の設計技法

デバイス・ドライバー内でのすべてのセキュリティー検査は、モジュール形式で作成して、簡単に識別できるようにする必要があります。

デバイス・ドライバー内での検査

必ず、デバイス・ドライバーから MIC、MAC、および DAC 検査を維持するほうが優れた方法です。 そのような検査を行わないデバイス・ドライバーは、非トラステッド・システムまたはその他のタイプのトラステッド・システムの移植を簡単に行うことができます。

正規のデバイス・ドライバーのインプリメンテーションでは、カーネルが MIC、MAC、および DAC 検査を実行し、ドライバーはそのほかに必要な特権検査を実行します。 非標準のデバイス・ドライバーのインプリメンテーションでは、すべての検査 (MIC、MAC、DAC、特権検査) はデバイス・ドライバー内で実行されます。正規または非標準デバイス・ドライバーのどちらを実装するかについての選択は、設計段階で判断する項目です。

DAC

DAC は、デバイスをアクセスするために使用されるファイルシステムのエントリー・ポイントに基づいて、各デバイスのスペシャル・ファイルに対して実施されます。

正しいインストールの検査

MAC 検査を行うデバイス・ドライバーはいずれも、デバイスが誤って定義された可能性に対して (妥当な範囲で) 確実に対処する必要があります。

特権アクセス

デバイス・ドライバーによって、特定のデバイス操作を特権プロセスに限定させることが適切かもしれません。 ただし、これらの状態に対して、いくつかの具体的な推奨事項があります。

refmon カーネル機能を使用して、必要な特権を持っているかどうかを判別することができます。

最小特権*:*

Trusted AIX は、最小特権の概念を導入しています。 最小特権は、以前に強力だった root ユーザーをより細分性の高い特権メカニズムに分離します。この特権の分離により、トラステッド・ソフトウェアでプログラミング・エラーや他の問題があった場合に、システム・セキュリティーへの損害が最小限になります。

特権操作*:*

各プロセスには、有効、最大、継承可能、および制限の 4 つの特権ベクトルが関連付けられます。

最大特権ベクトルは、各プロセスに対してアクティブとなる特権の上限を定義します。 有効特権ベクトルは、特権決定を行うために検査される特権を定義します。 有効特権セットは常に最大特権セットのサブセットであり、最大特権セットは常に制限特権セットのサブセットです。 制限特権セットは、プロセスがその最大、継承可能、および有効の特権セットに持っている可能性のある特権を定義します。 継承可能特権セットは、fork と exec 全体の子プロセスで継承される特権のセットを表します。

新規テキスト・イメージが実行されると、以下のアルゴリズムに基づいて特権の拡大が実行されます。 特殊な特権として、**PV_ROOT**、**PV_SU_**、**PV_SU_EMUL**、**PV_SU_ROOT**、**PV_AZ_ROOT**、および **PV_SU_UID** が挙げられます。

以下のアルゴリズムは、最小特権サブシステムに関する 2 つの重要な概念を示します。 1 つ目の概念は、特殊な特権 (**PV_ROOT**、**PV_SU_**、**PV_SU_EMUL**、**PV_SU_ROOT**、**PV_AZ_ROOT**、および **PV_SU_UID**) のみが新規プロセス・イメージの実行中に無条件に伝搬できるということです。 2 つ目の概念は、プロセスの有効特権ベクトルはファイルに **FSF_EPS** が設定されていない限り、すべての特権がクリアされるということです。 これにより、最小特権システムに対するブラケットなしでトラステッド・システム下で実行されるアプリケーションとの後方互換性が確保されます。

```
new_max_privs = old_inheritable_privs
new_max_privs = new_max_privs | file_innate_privs
IF (user was assigned some of authorizations in file PAS)
new_max_privs = new_max_privs | file_authorized_privs
new_max_privs = new_max_privs & old_limiting_privs
IF (old_max_privs contain one or more special privileges)
new_max_privs += same set of special privileges
IF (FSF_EPS is set for the executable)
new_eff_privs = new_max_privs
ELSE
new_eff_privs = old_inheritable_privs
IF (old_eff_privs contain one or more special privileges)
new_eff_privs += same set of special privileges
new_limiting_privs = old_limiting_privs
```

特権の割り当ておよび除去*:*

以下の標準システム・ライブラリー・ルーチンで、システムで特権を取り扱う方法について説明します。これらのルーチンはシステム上の特権プログラムでのみ有効です。

priv_raise
指定された特権リストを追加して (または増やして)、プロセスの有効特権ベクトルを変更する。特権リストはプロセスの最大特権ベクトルに存在していなければなりません。存在していないと、エラー通知が返されます。

priv_remove
指定された特権リストを除去して、プロセスの有効および最大特権ベクトルを変更する。 プロセスが有効または最大特権を除去できない場合は、エラー通知が返されます。

priv_lower
指定された特権リストを除去して (または減らして)、プロセスの有効特権ベクトルを変更する。プロセスが有効特権を減らすことができない場合は、エラー通知が返されます。

これらのルーチンは、それぞれ特権のコンマ区切りリストを受け入れますこのリストは **-1** (マイナスの 1、無効な特権番号) で終了します。これらの特権を必要とすることがあるコードの最小セクション辺りで特権を増減する手法は、「特権の囲い込み (privilege bracketing)」と呼ばれています。ソフトウェアの設計またはインプリメンテーションが悪くて引き起こされる、セキュリティー違反の可能性を削減するために、トラステッド・アプリケーションは「特権の囲い込み (privilege bracketing)」を使用します。

setppriv
プロセスの有効、最大、継承可能、および制限ベクトルが、特権セットの設定によって変更される。 パスされた特権セットが無効か、または許可されていない場合は、エラー通知が返されます。

権限*:*

権限はさまざまな特権セットを特定の権限のあるユーザーに与えます。

通常、コマンドまたはユーティリティーは実行の開始時に関連のある権限を検査し、その特権を設定します。このため、特定の権限のあるユーザーは、コマンドのプログラミング方法に応じて、実行されるコマンドごとに異なる特権セットを受け取ります。

コード自体から煩わしい特権の設定を取り除くために、AIX はバイナリー外部に権限セットと特権セットを備えています。 Privileged Authorization Set (PAS) と Authorized Privilege Set (APS) を使用して、コマンド自体ではなくシステムが権限に基づいた特権設定を実行します。

checkauths
権限リスト内のパス済みの権限を現在のプロセスに関連した権限と比較します。

許可検査の詳細については、 95 ページの『RBAC 権限』を参照してください。

監査*:*

Trusted AIX には、監査証跡の生成および情報を管理するためのコマンド・セットが組み込まれています。これはトラステッド・システム・プログラマーがこれらのプログラムに対して、変更または追加するために必要になるものではありません。

audit 監査デーモンを制御する

auditbin
監査証跡ファイルを制御する

auditselect
監査証跡ファイルから監査レコードをマージおよび選択する

auditpr
選択された監査イベントを人間に読み取り可能なフォームで表示する

監査がトラステッド・システム・プログラマーに関心のある基本領域は、トラステッド・プログラムによって生成される監査イベントにあります。 たいていのトラステッド・プログラムは、システム監査証跡へメッセージを発行することが必要になります。

監査する状態*:*

トラステッド・プログラムによって検出および監査する必要のある状態の判別について、的確なガイドラインがいくつかあります。 これは主として判断と監査の方針によるものです。 基本システムは状態を成功、障害、オブジェクト・アクセス、および潜在的な隠れチャネルに分割します。

成功*:*

基本的な使用方法の履歴を設定するには、成功した操作を監査することが重要です。

例えば、デバイス割り当てプログラムは、特定ユーザーがデバイスの割り当ておよび割り当て解除を行う時期を記録することが重要です。 これにより、プログラムでシステムを経由する情報の流れをトレースすることが可能になり、その後、デバイスが悪用されたかどうかを判別できます。 一方、一部の監査を行う考え方として、成功した操作はトラステッド・ソフトウェアによって合法的、かつ、適切であると判断されたという理由で、このような操作にはほとんど関心を示しません。

障害*:*

失敗した操作を監査することは、許可されていないサービスまたはデータへのアクセス権限を獲得しようとするユーザーの検出に役立ちます。 そのような失敗が頻繁に発生するということは、悪意のある人 (特別に優秀でもなければ) ということになります。

基本システムでは障害を 5 つのカテゴリーに分割します。
- 特権障害 (特権を持たないプロセスが特権プロセスに制限されたアクションを実行しようとする試み)
- MAC 障害 (アクションが MAC 制約事項に違反するという障害)
- MIC 障害 (アクションが MIC 制約事項に違反するという障害)
- DAC 障害 (アクションが DAC 制約事項に違反するという障害)
- その他の障害 (例えば、不正なパスワードでログインする試み)

オブジェクト・アクセス*:*

提供されたオブジェクト (例えば、シャドー・パスワード・ファイル) をアクセスするユーザーをモニターするためのオブジェクトを監査することは必要です。

潜在的な隠れチャネル*:*

隠れチャネルはプロセス間で異なる MAC ラベルでの情報の引き渡しに使用できるため、潜在的な隠れチャネルを監査することは重要です。 潜在的な隠れチャネルの使用ということは、これらのチャネルがこの目的のために使用されたということではなく、その目的で使用される可能性があるということです。

監査システムにより作成される各エントリーには、監査エントリーの理由が含まれます (成功、MAC 障害、MIC 障害、DAC 障害、特権障害、その他の障害、または潜在的な隠れチャネル)。 このエントリーにはシステム自体が作成する監査レコードおよびユーザー・プログラムが作成する監査レコードの両方が含まれます。

ユーザーがトラステッド・ユーザー (管理者) だったかどうかについて考慮することは有益ではありますが、トラステッド・ユーザーまたは非トラステッド・ユーザーが、より強力な監査を必要とするかどうかを決める絶対的な方式はありません。 例えば、トラステッドと想定されていて、このことで監査をそれほど必要としない場合でも、広範囲なアクションを対象にして、権限のない管理者のアクションを記録することは有益です。 正規のユーザーは損害を受けることは少ないので、その意味では監査をそれほど必要としませんが、これらのユーザーもそれほど信用できるとは言えませんので、より厳しい監査を行ったほうがよいでしょう。 システム管理者は、多くの場合、彼らのアクションに対する監査を増やし、セキュリティー・ブリーチ (抜け穴) のケースでは、自分たちに罪はないという態度を示します。

以下のイベントは監査可能にする必要があります。

- 成功した操作、特に情報の転送またはアクセス制御パラメーターの変更を行った操作
- セキュリティーの理由で失敗する操作
- 成功であったかどうかに関係なく、管理者による操作
- 潜在的な隠れチャネルの使用
- 特定オブジェクトをアクセスする操作
- 実際の監査証跡に対して後続の内容に影響のあるアクション

監査情報レベル*:*

高水準の監査情報は低レベルの監査情報より有効です。 トラステッド・プログラムは操作のハイレベル・ビューを維持して、優れた監査メッセージを生成します。

管理者が書き込みのためにセキュリティー・ファイルをオープンしたことだけを記録することは、ファイルに対して行われた実際のハイレベル操作を記録すること (例えば、管理者がファイルに新規のエントリーを作成したことを、新規のエントリーのためのキー情報も含めて記録すること) に比べて、それほど有用ではありません。監査情報は可能な限りハイレベルにすることを強くお勧めします。

取り込む情報は数個のイベントの情報を 1 つにまとめるよりも単一イベントごとのほうが優れています。複数のイベントで発生した監査を分割することの基本的な理由は、発生ごとに分割することにより、それぞれが選択可能になるからです。

監査クラスおよびイベント*:*

各トラステッド・プログラムは監査クラス、監査イベント・タイプ、およびこのプログラムが **auditlog** システム・コールを使用して、監査メッセージを発行するときに使用する理由を判別する必要があります。

各監査イベントは監査クラスに所属しています。 イベントをクラスに割り当てることにより、大多数のイベントを効率的に処理することができます。 監査クラス定義は `/etc/security/audit/config` ファイルに定義されます。

監査クラスはイベントの記録を使用可能および使用不可にする場合に使用されます。 2 つのイベントを分離して使用可能にすることが重要であれば、これらのイベントを同じ監査クラスにしないでください。 したがって、一般にイベントをクラスにグループ分けすることはよい方法です。 通常、各トラステッド・プログラムまたは関連トラステッド・プログラムのセットには、このプログラム自体の使用のために 1 個の監査クラス名 (またはまれなケースですが、数個の監査クラス名) が予約されています。

監査可能なシステム処置が監査イベントとして /etc/security/audit/events ファイルに定義されます。

隠れチャネル*:*

すべてのトラステッド・ソフトウェアは隠れチャネル方式に加わらないことを前提にしています。 さらに、このソフトウェアは、隠れチャネルを悪用する非トラステッド・ソフトウェアによって使用されないように設計する必要があります。 このセクションでは隠れチャネルを定義して、その検出と制限に関するガイドラインを提示します。

隠れチャネルの定義*:*

ラベル B がラベル A より上位である場合を除いて、ラベル A のプロセスでは、別のプロセスによって検出可能なアクションを実行することはできません。

この定義は、2 つの状態 (直接データ操作と付随操作) に分けることができます。 直接データ操作は、ファイルの読み取りおよび書き込みなどのユーザー・データの保管または通信の直接的な方法としてユーザーを対象としています。 これらの操作は、基本 MAC 制約を完全に順守したものでなければなりません。他のすべての操作は付随操作です。 付随操作を使用して基本 MAC 制約に反しデータを渡すことを、隠れチャネルと呼びます。

隠れチャネルを活用するには、送信側 (ラベル X) および受信側 (ラベル Y) と呼ばれる 2 つの非トラステッド・プロセスが必要です。 受信側の MAC ラベルは送信側の MAC ラベルより上位でないと想定されます (上位である場合には、送信側から受信側へのデータ・フローが正しいアップグレードとなります)。 このチャネルを活用するには、送信側と受信側の両方が、MAC に反してデータを伝送するために、承認済みのリソースの使用に関する特定の規則を使用します。

隠れチャネル利用についての唯一の基準は、受信側のラベルが送信側のラベルより上位でなく、送信側と受信側の両方が非トラステッドであるということです。 通常は送信側と受信側の両方が同じユーザーのために使用されます。 TCB 自体は基本 MAC 制約を守り、隠れチャネルの悪意のある使用によってこの制約に違反するいかなるコードもないことが想定されます。 (実際は、特権プロセスには、隠れチャネルを使用せずに MAC に違反するより効果的な方法が多くあります。) 非トラステッド・プロセスには、関係のあるトラステッド・プログラムを使用して隠れチャネルが活用される可能性があります。

通常は、隠れチャネルはシステムから除外する必要があります。 ただし、他のシステムのニーズ (例えば、パフォーマンス、信頼性、または互換性) が隠れチャネルなしでは受け入れ難いほど制約されている場合もあります。

処理能力のガイドライン*:*

基本システムでは、処理能力に基づいて隠れチャネルを制限する場合に、以下のガイドラインを使用します。

100 ビット/秒より大
: これらのチャネルが存在することを許可しません。

0.1 から **100** ビット/秒
: この範囲のチャネルは絶対に必要であるときに存在できますが、使用される場合は可能であればいつでも検出されて監査されます。

0.1 ビット/秒未満
: この範囲のチャネルは必要なところに存在できますが、その使用を検出する特別な必要性はありません。

すべての追加 TCB プログラムは上記と同じガイドラインに従うことを強くお勧めします。 さらに、10 ビット/秒のような比較的遅いチャネルでも 4,500 バイト/時間を伝送できるように考慮してください。 10 ビット/秒は違法にダウングレードになってしまう意味のあるデータ量です。 したがって、隠れチャネルの処理能力を可能な限り低く制限するための、あらゆる努力をする必要があります。

多くの隠れチャネルの処理能力は、通常、隠れチャネルを悪用する可能性のあるプロセス以外のプロセスの活動により低く抑えられます。しかし、すべてのシステムには活動が低い期間があるため、このことが隠れチャネルの処理能力を制限することに影響を与えませんが、それでも、このガイドラインをお勧めします。

隠れチャネルの検出*:*

隠れチャネルを検出することは、注意深い分析と設計には重要事項の 1 つです。 隠れチャネルを検出するために、いくつかの具体的なガイドラインがあります。

「条件モジュール (term module)」は、カーネル内であろうとプロセス内であろうと隠れチャネルの検出または制限を行う TCB コードのユニットを指します。隠れチャネルを検出することは、レベル B がレベル A より優位にない場合に、レベル A で非トラステッド・プロセス (送信側) が、レベル B で別のプロセス (受信側) によって検出可能なアクションを実行するモジュールを使用できるかどうかを判別するという、基本的な事柄です。

例えば、ファイルの MAC ラベルがユーザーの MAC ラベルより優位にない場合、共通の隠れチャネルとは、非トラステッド・ユーザーの利益になるように、トラステッド・プロセスによってファイルに書き込まれるデータのことです。

隠れチャネルを検出するために提案されている方法論は、それほど多くありません。 最も有名なものとして Shared Resource Matrix (SRM) があります。この手法の解説については、以下を参照してください。

- • Kemmerer, R.A. "Shared Resource Matrix Methodology: An Approach to Identifying Storage and Timing Channels," ACM Transactions on Computing Systems 1(3) 1983, 256-277.
- • Tsai, CR. "A Formal Method for the Identification of Covert Storage Channels in Source Code," Proceedings of the 1987 IEEE Symposium on Security and Privacy, 74-87.

監査による隠れチャネルの検出*:*

潜在的な隠れチャネルの使用を監査する能力があれば、この脅威に対して有効に対応することができます。しかし、監査することは有効であるけれども、監査イベントはそれほど多くはありません。 「監査を行わせるために付随して起こるイベントの使用」に対する「実際の悪用」の比率が低い場合、監査の使用は少なくなります。

隠れチャネルの制限*:*

隠れチャネルを制限するための最も優れた方法は、隠れチャネルを除去することです。

そのようにしないのであれば、 処理能力のガイドラインで検討したガイドラインに従って制限してください。 それに加えて、可能であり有効であれば必ず、チャネルが使用される可能性について監査する必要があります。

一般に、カーネルまたはデバイス・ドライバー・コードで隠れチャネルを制限することは困難です。カーネルおよびデバイス・ドライバー・コードは効率的に設計されていて、そのチャネルの処理能力は高くなるように設計されているからです。 トラステッド・プロセスでは、より容易に隠れチャネルを制限することができます。

注**:** 同じラベルの場合、または受信側が送信側より優位にある場合には、プロセスで隠れチャネルの使用を制限する理由がありません。 したがって、たいていの TCB モジュールは、これらのケースで制限を課すことなくシステム・パフォーマンスを高めることができます。

ラベル当たりの割り当て量*:*

多くの隠れチャネルでは、異なる MAC ラベルでのプロセス間で共有されるリソース・プールが使用されます。 これらのチャネルは、それぞれの MAC ラベルごとに分離した固定サイズのリソース・プールを作成することにより、効果的に制限することができます。その場合、プロセスができることは、その MAC ラベルのプールから使用するリソースを調節することだけです。

時間超過、未使用リソースは 1 つのプールから別のプールへ移して、動的要求に備えることができます。このリソース移行自体が隠れチャネルですが、処理能力をかなり低くするという制限を簡単に行うための 1 つの手法です。

時間遅延*:*

隠れチャネルを制限するための 1 つの手法として、隠れチャネルが存在するサービスが実行されるときの時間の一定量を TCB に確実にパスするやり方があります。 これは、パスされている情報量に基づいて計算できる指定時間の間は、モジュールを活動休止させるだけの簡単なものです。

ただし、適切に行わないと、時間遅延は隠れチャネルを悪用するプログラムによって妨げられることがよくあります。 例えば、悪用するプロセスは多くの送信側/受信側プロセスを作成することができます。 一方、TCB は時間遅延を使用して、簡単に、それぞれのセットを特定の処理能力に制限することが可能であり、全セットの総計が単一チャネルの処理能力になってしまいます。

特定の TCB サービスには、サービスを使用する可能性のあるすべてのプロセスに、ある方法で時間遅延を確実に適用する方法が優れています。

時間遅延は制限することには役立ちますが、これは悪意のあるプログラムによって比較的簡単に対抗策を取られてしまう傾向にあり、注意深く設計する必要があります。

データ制限*:*

隠れチャネルの処理能力については、時間を延ばすだけでなく、戻される情報量を削減して低減することができます。一連の操作の結果としてデータを戻すプログラムは、同じ時間枠でより少ない情報のパケットをただ戻すだけの場合がよくあります。

概算時間*:*

隠れチャネルを悪用するための多くの手法には、相対または絶対時間を測定するための正確な方法を持っている悪用プロセスが必要になります。 これらのチャネルは、プロセスで正確に時間を判別することが許可されていないために、時折制限されることがあります。

時間情報を戻す TCB サービスに確実に時間を概算させることは比較的容易ですが、時には、プロセスは時間経過の測定をこのサービス自体の命令時間もカウントする、といった事実とは異なる方法をとってしまいます。チャネルを制限することになる、このような手法は注意して使用する必要があります。

ノイズ作成者*:*

多くの隠れチャネルの処理能力は、通常、チャネルを悪用するプロセス以外の活動により時には低く抑えられます。 目的が、常時、特定レベルでの活動が確実に存在することであるトラステッド・プログラムを作成することは可能です (お勧めはしません)。時には、このプログラムを「ノイズ作成者 (noisemaker)」と呼んでいます。

ノイズ作成者の使用について概念的に訴えているだけでは、通常、ノイズ作成者がノイズをいつ作って、いつ作らないかを判断することは無理です。 したがって、これを隠れチャネルの制限のための手法としてはお勧めしません。

U-T-U チェーン*:*

非トラステッド・プロセス **U1** が特権を持つトラステッド・プロセス **T** を起動し、このトラステッド・プロセス **T** は、次に、**U1** とは異なるラベルで別の非トラステッド・プロセス **U2** を起動する状態になることがあります。 **U1** および **U2** は、その他のプロセスの子孫であるプロセスに基づいて、潜在的な隠れチャネルを持っている異なる MAC ラベルでの非トラステッド・プロセスを表します。 (実際には、T および U はトラステッドおよび/または非トラステッド・プロセスの順序にすることができます。) この状態を「U-T-U チェーン」と呼んでいます。

トラステッド・プロセスは、不許可の直接データ操作を除外することと、もちろん、隠れチャネルも除外することの両方を含む基本 MAC 原則に対して、 2 つの非トラステッド・プロセス間で情報がパスされないことを確実にする必要があります。 以下の項目について考慮してください。

- 読み取り/書き込みモードでファイルがオープンされている状態で **U2** をオープンできなかったときに、ファイル・ディスクリプターをオープンのままに残すことができない。
- **U2** のラベルが U1 より優位でない場合に、環境変数を消去する必要がある。
- **U2** のラベルが **U1** より優位でいない場合に、**U1** から **U2** にパススルーされる作業ディレクトリーは、隠れチャネル (多分、小規模) を構成できる。同様に、子プロセスによって自動的に継承されたプロセス・パラメーターの多くは、隠れチャネルを構成できる可能性がある。

U-T-U チェーンを適切に管理することは可能である (言い換えれば、隠れチャネルを十分に制約できる)。しかし、これを確実にすることは困難であり、通常は U-T-U チェーンを避ける必要がある。 ただし、U2 がトラステッドではない不安があること、トラステッドかもしれないが特権を持っていないことに注意してください。

隠れチャネルの例*:*

システム・プログラマーが作成するモジュールに存在することがある隠れチャネルの例を以下に示します。

印刷サービスの隠れチャネルの例*:*

以下は印刷サービスの隠れチャネルの例です。

トラステッド・ライン・プリンター・サービスは、実行依頼されたそれぞれのジョブに、要求プロセスの MAC ラベルで正しくタグを付けて、最終的には印刷するために使用される待機ジョブをそのラベルで維持します。 ジョブにはロング・ネームの使用が許可されます。

状況プログラムにより、ユーザーはキューに入れられたユーザーのすべてのジョブを、ユーザー割り当てジョブ名を含めて、ジョブのラベルに関係なく見ることができるようになります。 その後、同じユーザーに代わって操作するデータを含む名前のジョブを送信側プロセスが作成して、受信側にひそかに渡すことができるため、このプログラムは隠れチャネルとして使用できます。

注: 隠れ利用の唯一の基準は、受信側のラベルが送信側のラベルより上位にないこと、および送信側および受信側両方が非トラステッドであることです。 送信側および受信側両方は、一般に、同じユーザーに成り代わります。

このチャネルは、そのユーザーの現在の MAC ラベルが上位にあるジョブのみの表示をユーザーに許可してクローズされます。 これにより、受信側の MAC ラベルが送信側の MAC ラベルより強制的に上位に変更され、正当なアップグレードのみにチャネルが使用できるようになります。 当然のこととして、上位のジョブ (non-dominated job) が存在する場合、状況プログラムは「other jobs exist (その他のジョブが存在しています)」というメッセージをユーザーに出すことができます。 これはかなり小さいチャネルですが、存在理由は優れた操作性にあります。

注: 上位のジョブの検出は通常の操作ではおそらくまれなことですので、そのようなジョブの検出を監査することが有益な場合もあります。

これは、マルチレベルの名前付きデータ・オブジェクト (この場合は、待機印刷ジョブ) が、異なる MAC ラベルでプロセスによってアクセス可能な隠れチャネルの一般的な例です。 このチャネルは、オブジェクトの MAC ラベルを名前にも適用して効率的に除去されます。 名前以外の属性 (サイズなど) も、隠れ情報を持っている場合があります。

リソース・プールの例:

トラステッド・プログラムが非トラステッド・クライアントにサービスを行うときに、トラステッド・プログラムはリソースの特定タイプ (例えば、バッファー) を異なる MAC ラベル間で共有されているリソースのプールから割り当てます。

これを隠れチャネルとして使用する 1 つの方法として、送信側および受信側がすべてのリソースを 1 つだけ除いて全部を割り当てるように調整します。その場合はできる限り、さまざまな MAC ラベルか、またはさまざまなユーザー ID で実行しているその他のプログラムごとに割り当てます。 次に、送信側は残りの 1 つのリソースを割り当てるか、あるいは割り当てないかを生じさせます。そして、受信側もそのリソースの割り当てを試みて、これを検出します。

共有リソース・チャネルに対して、これはよく知られている例です。 この例は、上の説明のようにラベルごとのリソース・プールの割り当てに限定されるものです。 これも監査で検出できます。

データベースの例:

トラステッド・データベース・システムにより、ユーザー・プログラムでデータをマルチレベル・データベースに保管できるようになります。 直接アクセスは基本 MAC 制約により適切に制御されます。

ただし、エントリーをデータベースに作成するために必要な時間は、データベースの現在の合計サイズに大きく依存しています。 したがって、送信側はエントリーを作成したり、または除去することでデータベースのサイズに影響を及ぼすことになり、受信側はエントリーを作成するときに必要な時間を測るだけで、このサイズを検出することができます。データベース・アクセスがまったく効率的でない場合、このチャネルについては処理能力が低くなる可能性があります。

チャネルを制限する目的で保証済みの最小アクセス時間を課すことができます。 平均浪費時間を減らすことができるように、時間遅延を疑似乱数にすることができます。 しかし、それでも、これは時間遅延の方式であり、注意深く実装する必要があります。

多くの悪意のないデータベースの使用がある中で、チャネルの悪用を検出することは困難であり、すべてのアクセスについて監査するだけでは効率的とは考えられません。

プログラミングの例:

このセクションではトラステッド・プログラミングの例を提供します。

トラステッド・プログラムの特権検査の例:

これはトラステッド・プログラムが呼び出しプロセスが特定の特権を所有しているかどうかについて、検査するためのモジュラー・ルーチンです。

```
#include <sys/priv.h>
#include <sys/secattr.h>

int
priv_check (int priv)
{
  /* the process's security attributes */
  secattr_t secattr;

/* get the calling process's security attributes */
  if ( sec_getpsec(-1, &secattr;) != 0 )
  {
    return (-1);
/* error retrieving the process's cred structure */
  }

  /*
   * return whether or not specified priv is in the
   * calling process's maximum privilege set
   */
  return privbit_test(secattr.sc_maxpriv, priv);
}
```

有効な機密ラベルの変更例:

このプログラムは現在のプロセスの有効な機密ラベルを高位レベル・システムに変更します。

以下の特権がプログラムの固有の特権セットに必要です。

- **PV_LAB_LEF**
- **PV_LAB_SLUG**
- **PV_LAB_SL_SELF**

```
#include <stdio.h>
#include <mls/mls.h>
#include <unistd.h>
#include <sys/secattr.h>
#include <userpriv.h>
#include <sys/mac.h>
#include <sys/secconf.h>

#define SUCCESS 0
#define ERROR 1

int
main()
{
 sl_t sl_syshi; /* System high SL */
 secattr_t attr;
 char *clBuffer = NULL;

 /*
  * Get the system high and low SLs.
  */
 if ((sec_getsyslab(NULL, &sl_syshi, NULL, NULL)) != 0 ) {
```

```
 fprintf (stderr, "Call to sec_getsyslab failed.¥n");
 exit(ERROR);
}

/*
 * Initialize this process with initlabeldb() to access the
 * system default Label database.
 */
priv_raise(PV_LAB_LEF , -1);
if (initlabeldb(NULL) != 0) {
 fprintf(stderr, "Could not read the Label Encodings Database.¥n");
 exit(ERROR);
}
priv_remove(PV_LAB_LEF, -1);

/*
 * Get the process clearance range and effective SL.
 */
priv_raise(PV_LAB_SLUG, PV_LAB_SL_SELF, -1);
if (sec_getpsec(-1, &attr) != 0) {
 fprintf (stderr, "Problem getting Trusted AIX security attributes of program.¥n");
 exit(ERROR);
}

/* malloc for the maximum SL label length that can be formed for process */
if((clBuffer = (char *) malloc(maxlen_cl())) == NULL) {
 perror("malloc");
 exit(ERROR);
}
/* Convert the binary effective SL to human readable */
if (clbtohr(clBuffer, &attr.sc_sl, HR_LONG) != 0) {
 fprintf(stderr, "Unable to convert SL to human readable form.¥n");
 exit(ERROR);
}
printf("Program's intitial effective SL = %s.¥n",clBuffer);

/*
 * Set the process effective SL to system high.
 * The process may not have its maximum SL at system high,
 * so set it also to system high.
 */
attr.sc_sl = sl_syshi;
attr.sc_sl_cl_max = sl_syshi;

if (sec_setplab(-1, &attr.sc_sl, NULL, &attr.sc_sl_cl_max,
  NULL, NULL, NULL) != 0) {
 fprintf (stderr, "Problem setting the effective SL of program.¥n");
 exit(ERROR);
}

priv_lower(PV_LAB_SLUG, PV_LAB_SL_SELF, -1);

if (sec_getpsec(-1, &attr) != 0) {
 fprintf (stderr, "Problem getting Trusted AIX security attributes of program.¥n");
 exit(ERROR);
}

/* Convert the binary effective SL to human readable */
if (clbtohr(clBuffer, &attr.sc_sl, HR_LONG) != 0) {
 fprintf(stderr, "Unable to convert to SL to human readable form.¥n");
 exit(ERROR);
}
printf("Program's modified effective SL = %s.¥n",clBuffer);
return(SUCCESS);
}
```

機密ラベル分類の設定および機密ラベルの比較の例:

これは機密ラベル分類の設定と、それぞれの機密ラベルの比較にライブラリー・ルーチンを使用する例です。

PV_LAB_LEF 特権がプログラムのプロキシー特権セットおよび呼び出しプロセスの最大特権セットに必要です。

```
#include <stdio.h>
#include <mls/mls.h>
#include <userpriv.h>
#include <errno.h>

#define SUCCESS 0
#define ERROR 1
int
main (int argc, char **argv)
{
/* Sensitivity labels */
sl_t sl1, sl2;

/* strings to hold labels' names */
char *slBuffer1 = NULL;
char *slBuffer2 = NULL;

if (argc != 3) {
fprintf(stderr, "Usage: compare slabel1 slabel2¥n");
exit(ERROR);
}
/*
* Initialize this process with initlabeldb() to access the
* system default Label database.
*/
priv_raise(PV_LAB_LEF , -1);
if (initlabeldb(NULL) != 0) {
fprintf(stderr, "Could not read the Label Encodings Database.¥n");
exit(ERROR);
}
priv_remove(PV_LAB_LEF, -1);

/* Convert the passed SL to binary format */
if (slhrtob(&sl1, argv[1]) != 0) {
fprintf(stderr, "Unable to convert %s to binary form.¥n", argv[1]);
exit(ERROR);
}
if (slhrtob(&sl2, argv[2]) != 0) {
fprintf(stderr, "Unable to convert %s to binary form.¥n", argv[2]);
exit(ERROR);
}

/* malloc for the maximum SL label length that can be formed */
slBuffer1 = (char *) malloc(maxlen_sl());
slBuffer2 = (char *) malloc(maxlen_sl());

if ((slBuffer1 == NULL) || (slBuffer2 == NULL)) {
perror("malloc");
exit(ERROR);
}

/*
* Translate the label back to human readable (long) form.
* This is not a necessary step. It is shown as an example
* usage of slbtohr() API.
*/
if (slbtohr(slBuffer1, &sl1, HR_LONG) != 0) {
fprintf(stderr, "Unable to convert to binary human readable form.¥n");
```

```
exit(ERROR);
}

if (slbtohr(slBuffer2, &sl2, HR_LONG) != 0) {
fprintf(stderr,"Unable to convert to binary human readable form.¥n");
exit(ERROR);
}

/*
* Use sl_cmp() to compare the dominance of the two labels.
*/
if (sl_cmp(&sl1, &sl2) == LAB_SAME) {
printf("label (%s) equals label (%s).¥n",
slBuffer1, slBuffer2);
}
else if (sl_cmp(&sl1, &sl2) == LAB_DOM) {
printf("label (%s) dominates label (%s).¥n",
slBuffer1, slBuffer2);
}
else if (sl_cmp(&sl2, &sl1) == LAB_DOM) {
printf("label (%s) dominates label (%s).¥n",
slBuffer2, slBuffer1);
}
else {
printf("The two labels are disjoint.¥n");
}

return (SUCCESS);
}
```

監査情報の設定の例*:*

このプログラムは監査情報を取り出して設定します。

以下の特権がプログラムの固有の特権セットに必要です。

- **PV_AU_ADMIN**
- **PV_DAC_GID**

```
#include <sys/types.h>
#include <sys/priv.h>
#include <sys/audit.h>

char buf[1024];
int main(int argc, char *argv[])
{
 int rc, len, p;
 /* *Get process audit preseclection mask */
 priv_raise(PV_AU_ADMIN, -1);
 rc = auditproc(0, AUDIT_QEVENTS, buf, sizeof (buf));
 priv_lower(PV_AU_ADMIN, -1);
 if (rc)
  fprintf(stderr, "Failed to get audit info¥n");
        /* *Add the `kernel audit class to the preselection mask */
 p = 0;
 while ((len = strlen(&buf;[p])) > 0)
  p += len +1;
        strncat(&buf;[p], "kernel", (sizeof(buf)-p-1));
 p += strlen("kernel") + 2;
 buf[p] = 0;
 priv_raise(PV_AU_ADMIN, -1);
 rc = auditproc(0, AUDIT_EVENTS, buf, p);

 priv_lower(PV_AU_ADMIN, -1);
 if (rc)
  fprintf(stderr, "Failed to set audit info¥n");
```

```
 /* *Set the GID of the process to generate an audit record */
 priv_raise(PV_DAC_GID, -1);
 rc = setgid(129);
 priv_lower(PV_DAC_GID, -1);
 if (rc)
  fprintf(stderr, "Failed to setgid¥n");
 exit(0);
}
```

クライアントの例:

このプログラムは 2 つのメッセージをサーバーに送信します。 1 つは標準 **write** ルーチン、もう 1 つは **ewrite** ルーチンを使用しています。

セキュア・メッセージが SECRET で送信されます。 **write** 呼び出しを使用して送信される無保護メッセージは、 netrule を経由して構成可能なセキュリティー属性のデフォルト・セットを渡されることに注意してください。

以下の特権がプログラムの固有の特権セットに必要です。

- **PV_LAB_LEF**
- **PV_MAC_CL**
- **PV_LAB_SLUG_STR**

```
#include  <sys/mac.h>
#include  <sys/socket.h>
#include  <netinet/in.h>
#include  <sys/priv.h>
#include  <sys/secattr.h>
#include  <errno.h>
#include  <stdio.h>
#define SECURE 1
int
main(int  argc,  char  *argv[])

{
 int  sockfd;
 int  uid,  gid;
 char  buf[BUFSIZ];

 struct  sockaddr_in  serv_addr;

#ifdef  SECURE
 int  l_init_result  =  0;

 int  ewrite_result  =  0;

 sec_labels_t  seclab;

#endif  /*SECURE*/


 uid  =  getuid();

 gid  =  getgid();


 if  (  argc  !=  3  )

 {
  fprintf(stderr,  "Usage:%s:  ADDR  PORT¥n",  argv[0]);

  exit(1);
 }
```

```
#ifdef  SECURE
 /*
  * *  Gain  access  to  the  Label  Encodings  Database
  *
  *     */


 priv_raise(PV_LAB_LEF,-1);
 l_init_result  =  initlabeldb(NULL);
 if  (  priv_remove(PV_LAB_LEF,  -1)  !=  0  )
 {
  fprintf(stderr,  "Privilege  Failure¥n");
  exit(1);
 }
 if  (  l_init_result  !=  0  )
 {
  fprintf(stderr,  "Could  not  read  the  Label  Encodings  Database¥n");
  exit(0);
 }
#endif  /*SECURE*/
  /*
   * *  Fill  in  the  structure  "serv_addr"  with  the  address
   * of
   * *  the  server  that  we  want  to  connect  with.
   * */
 memset  ((char  *)  &serv_addr;,  '¥0',  sizeof(serv_addr));
 serv_addr.sin_family  =  AF_INET;
 serv_addr.sin_addr.s_addr  =  inet_addr(argv[1]);
 serv_addr.sin_port  =  htons(atoi(argv[2]));
 /*  Open  a  TCP  socket  (an  Internet  stream  socket).  */
 if  (  (sockfd  =  socket(AF_INET,  SOCK_STREAM,  0))  <  0)
 {
  perror("tcpclient:  ");
  fprintf(stderr,  "client:  Cant  open  stream  socket¥n");
  exit(0);
 }
 if  (  connect(sockfd,  (struct  sockaddr  *)  &serv_addr;,
    sizeof(serv_addr))  <  0  )
 {
  perror("tcpclient:  ");
  fprintf(stderr,  "client:  Cant  connect  to  server¥n");
  exit(0);
 }
 /*
  * *  Send  a  normal  write  to  the  server,  which  will  be
  * *  assigned  default  security  attributes
  * */
 strcpy(buf,  "This  has  the  default  security  attributes.¥n");
 if  (  write(sockfd,  buf,  strlen(buf)+1)  ==  -1  )
 {
  perror("tcpclient:  ");
  fprintf(stderr,  "write  error¥n");
 }
#ifdef  SECURE
  strcpy(buf,  "This  message  is  at  SECRET¥n");
 /*  Set up  the  SL  and  CLs  */
 slhrtob(&seclab.sl;, "SECRET");
 slhrtob(&seclab.sl_cl_min;,  "SECRET");
 slhrtob(&seclab.sl_cl_max;,  "SECRET  A  B");
 seclab.sl.sl_format  =  STDSL_FORMAT;
 seclab.sl_cl_min.sl_format  =  STDSL_FORMAT;
 seclab.sl_cl_max.sl_format  =  STDSL_FORMAT;
 /*  This  ewrite  call  needs  PV_MAC_CL  and  PV_LAB_SLUG_STR  */
 priv_raise(PV_MAC_CL,PV_LAB_SLUG_STR,-1);
 ewrite_result  =  ewrite(sockfd,  buf,strlen(buf)+1,  &seclab;);
 priv_lower(PV_MAC_CL,PV_LAB_SLUG_STR,-1);
```

```
 if  (ewrite_result  ==  -1)
 {
  perror("tcpclient  call");
  fprintf(stderr,  "ewrite  error¥n");
 }
 fflush(stderr);
#endif  /*SECURE*/
 fprintf(stderr,  "exiting  .....  ¥n");
 sleep(3);
 close(sockfd);
 exit(0);
}
```

サーバーの例*:*

このプログラムはサーバーとしての機能を果たし、**eread** ルーチンを使用して、ポートに送信されるメッセージを受信します。 メッセージを正常に受信したら、このプログラムはメッセージのセキュリティー属性を出力します。

以下の特権がプログラムの固有の特権セットに必要です (FSF_EPS secflags は割り当てない)。

- **PV_LAB_LEF**
- **PV_MAC_CL**
- **PV_MAC_R_STR**

```
#include  <sys/mac.h>
#include  <sys/socket.h>
#include  <sys/priv.h>
#include  <sys/secattr.h>
#include  <sys/stropts.h>
#include  <netinet/in.h>
#include  <errno.h>
#include  <stropts.h>
#include  <unistd.h>
#include  <stdio.h>
#include  <mls/mls.h>
#define  MAX_HR_LABEL_LEN 2048
#define  SECURE 1
int
main(int  argc,  char  *argv[])
{
 pid_t  childpid;
 uint  clilen;
 int  sockfd,  newsockfd;
 struct  sockaddr_in  cli_addr,  serv_addr;


#ifdef  SECURE
 int  l_init_result;
 char  label_str[MAX_HR_LABEL_LEN];
 sec_labels_t  seclab;
#endif  /*  SECURE  */
 if  (  argc  !=  2  )
 {
  fprintf(stderr,  "Usage:%s  PORT¥n",  argv[0]);
  exit(1);
 }
#ifdef  SECURE
 priv_raise(PV_LAB_LEF,  -1);
 l_init_result  =  initlabeldb(NULL);
 if  (priv_remove(PV_LAB_LEF,  -1)  !=  0)
 {
  fprintf(stderr,  "Privilege  Failure¥n");
  exit(1);
 }
```

```
 if  (l_init_result  !=  0)
 {
  fprintf(stderr,  "Could  not  read  the  Label  Encodings     Database¥n");
  exit(1);
 }
#endif  /*  SECURE  */
 /*  Open  a  TCP  socket  (an  Internet  stream  socket).  */
 if  (  (sockfd  =  socket(AF_INET,  SOCK_STREAM,  0))  <  0  )
 {
  perror("tcpserver:  ");
  fprintf(stderr,  "server:  Cant  open  stream  socket¥n");
  exit(1);
 }
 /*Bind  our  local  address  so  that  the  client  can  send  to  us*/
 memset((char  *)  &serv_addr;,  '¥0',  sizeof(serv_addr));
 serv_addr.sin_family  =  AF_INET;
 serv_addr.sin_addr.s_addr  =  htonl(INADDR_ANY);
 serv_addr.sin_port  =  htons(atoi(argv[1]));
 if  (  bind(sockfd,  (struct  sockaddr  *)  &  serv_addr,
    sizeof(serv_addr))  <  0  )
 {
  perror("tcpserver:  ");
  fprintf(stderr,  "server:  Cant  bind  local  address¥n");
  exit(0);
 }
 listen(sockfd,  5);
 for  (;;)
 {
  /*
   * *  Wait  for  a  connection  from  a  client  process.
   * */
  fprintf(stdout,  "Waiting  for  a  connection  from  a  client¥n");
  clilen  =  sizeof(cli_addr);
  newsockfd  =  eaccept(sockfd,  (struct  sockaddr  *)  &      cli_addr,
    &clilen;,  &seclab;);
  if  (  newsockfd  <  0  )
  {
   perror("tcpserver:  ");
   fprintf(stderr,  "server:  accept  error¥n");
  }
  /*  Print  SL  */
  if  (  slbtohr(label_str,  &seclab.sl;,  HR_SHORT)  !=  0  )
  {
   fprintf(stderr,"problem  converting  sl  to  string¥n");
  }
  else
  {
   fprintf(stdout,  "sl  =  %s.¥n",label_str);
  }
  /*  Print  MIN  CLEARANCE  */
  if  (  slbtohr(label_str,  &seclab.sl_cl_min;,  HR_SHORT)  !=  0  )
  {
   fprintf(stderr,"problem  converting  min  clearance  to  string¥n");
  }
  else
  {
   fprintf(stdout,  "sl_cl_min  =  %s.¥n",label_str);
  }

  /*  Print  MAX  CLEARANCE  */
  if  (  slbtohr(label_str,  &seclab.sl_cl_max;,  HR_SHORT)  !=  0  )
  {
   fprintf(stderr,"problem  converting  max  clearance  to  string¥n");
  }
  else
  {
```

```
   fprintf(stdout,  "sl_cl_max  =  %s.¥n",label_str);
  }
  if  (  (childpid  =  fork())  <  0  )
  {
   perror("tcpserver:  ");
   fprintf(stderr,  "server:  fork  error¥n");
   exit(0);
  }
  else  if  (  childpid  ==  0  )  /*  child  process  */
  {
   int  i,  j;
   char  buf[BUFSIZ];
#ifdef  SECURE
  sec_labels_t  e_seclab;
#endif  /*  SECURE  */
  close(sockfd);
  for  (;;)
  {
      int  ret,  flag;
      struct  strbuf  ctstr,  dtstr;
      char  ctbuf[2048],  dtbuf[2048];
      ctstr.maxlen=2048;
      ctstr.buf  =  ctbuf;
      dtstr.maxlen=2048;
      dtstr.buf  =  dtbuf;
#ifdef  SECURE
      fprintf(stdout,  "Calling  eread¥n");
      priv_raise(PV_MAC_CL,PV_MAC_R_STR,-1);
      ret  =  eread(newsockfd,  buf,  sizeof(buf),&e_seclab;);
      priv_lower(PV_MAC_CL,PV_MAC_R_STR,-1);
      if  (  ret  <  1  )
     {
         if  (  ret  ==  -1  )
   fprintf(stderr,  "eread  error¥n");
         else
   fprintf(stderr,  "eread  no  data¥n");
   close(newsockfd);
   exit(ret);
      }
      fprintf(stdout,  "¥n%s",  buf);
      fprintf(stdout,  "¥n");
      /*  Print  SL  */
      if  (  slbtohr(label_str,  &e_seclab.sl;,  HR_SHORT)  !=  0  )
      {
   fprintf(stderr,"problem  converting  sl  to  string¥n");
       }
      else
      {
   fprintf(stdout,  "sl  =  %s.¥n",label_str);
       }
       /*  Print  MIN  CLEARANCE  */
       if  (  slbtohr(label_str,&e_seclab.sl_cl_min;,HR_SHORT)!=  0)
      {
   fprintf(stderr,"problem  converting  min  CL  to string¥n");
       }
      else
      {
   fprintf(stdout,  "sl_cl_min  =  %s.¥n",label_str);
       }
       /*  Print  MAX  CLEARANCE  */
      if  (  slbtohr(label_str,&e_seclab.sl_cl_max;,HR_SHORT)  !=0)
      {
   fprintf(stderr,"problem  converting  max  CL  to  string¥n");
       }
       else
              {
   fprintf(stdout,  "sl_cl_max  =  %s.¥n",label_str);
```

```
        }
        fflush(stdout);
#else  /*  NOT  SECURE  */
        fprintf(stdout,  "Calling  read¥n");
        if  (read(newsockfd,  buf,  sizeof(buf))  <  1)
       {
   if  (ret  ==  -1)
      fprintf(stderr,  "read  error¥n");
   else
      fprintf(stderr,  "read  no  data¥n");
   close(newsockfd);
   exit(ret);
        }
        fprintf(stdout,  "%s¥n",  buf);
        fflush(stdout);
#endif  /*  NOT  SECURE  */
              }
 }
 /*  parent  process  */
 close(newsockfd);
        }
}
```

Trusted AIX ユーザーおよびポート・セキュリティー属性*:*

ユーザーおよびポートの認可属性を取り出して、ユーザーの認可属性とポート認可属性を比較する場合は、ユーザーおよびポートのセキュリティー属性が使用されます。

Trusted AIX の場合、以下の追加属性が **usersec.h** ファイルに定義されています。

S_MINSL
ユーザーの最小機密認可ラベル。 SEC_CHAR タイプ。

S_MAXSL
ユーザーの最大機密認可ラベル。 SEC_CHAR タイプ。

S_DEFSL
ユーザーのデフォルト機密ラベル。 SEC_CHAR タイプ。

S_MINTL
ユーザーの最小保全性認可ラベル。 SEC_CHAR タイプ。

S_MAXTL
ユーザーの最大保全性認可ラベル。 SEC_CHAR タイプ。

S_DEFTL
ユーザーのデフォルト保全性ラベル。 SEC_CHAR タイプ。

ポートには次の属性が有効です。

S_MINSL
ポートに割り当てられた最小機密ラベル。 SEC_CHAR タイプ。

S_MAXSL
ポートに割り当てられた最大機密ラベル。 SEC_CHAR タイプ。

S_TL ポートに割り当てられた保全性ラベル。 SEC_CHAR タイプ。

以下に、ユーザーが指定ポートにログインできるかどうかについて判別する例を示します。

```
#include <mls/mls.h>
#include <usersec.h>
#include <stdio.h>
```

```
#include <errno.h>

struct userlabels {
 sl_t minsl;
 sl_t maxsl;
 sl_t defsl;
 tl_t mintl;
 tl_t maxtl;
 tl_t deftl;
};

struct portlabels {
 sl_t minsl;
 sl_t maxsl;
 tl_t tl;
};

void getuserlabels(char * username, struct userlabels *usrlab);
void getportlabels (char * portname, struct portlabels *portlab);
void displayuseraccess (char * username, struct userlabels *usrlab,
     struct portlabels *portlab);

int
main (int argc, char **argv)
{

 struct userlabels usrlab;
 struct portlabels portlab;
 char *username = NULL;
 char *portname = NULL;


 if (argc != 3 ) {
  fprintf (stderr, "Usage: %s <username> <portname>¥n", argv[0]);
  exit(1);
 }
 username = argv[1];
 portname = argv[2];

 initlabeldb(NULL);
 getuserlabels(username, &usrlab;);
 getportlabels(portname, &portlab;);
 displayuseraccess(username , &usrlab;, &portlab;);
 endlabeldb();
}

void getuserlabels(char *username,  struct userlabels *userlab)
{

 dbattr_t attributes[6];
 memset (attributes, 0, sizeof(attributes));

 attributes[0].attr_name = S_MINSL;
 attributes[0].attr_type = SEC_CHAR;

 attributes[1].attr_name = S_MAXSL;
 attributes[1].attr_type = SEC_CHAR;

 attributes[2].attr_name = S_DEFSL;
 attributes[2].attr_type = SEC_CHAR;

 attributes[3].attr_name = S_MINTL;
 attributes[3].attr_type = SEC_CHAR;

 attributes[4].attr_name = S_MAXTL;
 attributes[4].attr_type = SEC_CHAR;
```

```
 attributes[5].attr_name = S_DEFTL;
 attributes[5].attr_type = SEC_CHAR;

 if (getuserattrs(username, attributes, 6)) {
  fprintf(stderr,
    "Error retriving attributes for user %s¥n", username);
  exit (1);
 }

 if (clhrtob (&(userlab->minsl), attributes[0].attr_char)) {
  fprintf(stderr, "minsl conversion error¥n");
  exit (1);
 }

 if (clhrtob(&(userlab->maxsl), attributes[1].attr_char)) {
  fprintf(stderr, "maxsl conversion error¥n");
  exit (1);
 }

 if (clhrtob(&(userlab->defsl), attributes[2].attr_char)) {
  fprintf(stderr, "defsl conversion error¥n");
  exit (1);
 }

 if (tlhrtob(&(userlab->mintl), attributes[3].attr_char)) {
  fprintf(stderr, "mintl conversion error¥n");
  exit (1);
 }

 if (tlhrtob(&(userlab->maxtl), attributes[4].attr_char)) {
  fprintf(stderr, "maxtl conversion error¥n");
  exit (1);
 }

 if (tlhrtob(&(userlab->deftl), attributes[5].attr_char)) {
  fprintf(stderr, "deftl conversion error¥n");
  exit (1);
 }

 printf("User %s has the following clearance values¥n", username);
 printf("minsl:%s¥n", attributes[0].attr_char);
 printf("maxsl:%s¥n", attributes[1].attr_char);
 printf("defsl:%s¥n", attributes[2].attr_char);
 printf("mintl:%s¥n", attributes[3].attr_char);
 printf("maxtl:%s¥n", attributes[4].attr_char);
 printf("deftl:%s¥n", attributes[5].attr_char);

 return;
}

void getportlabels(char *portname, struct portlabels *portlab)
{
 int rc =0;
 char *val = NULL;
 if ( (rc = getportattr(portname,S_MINSL,(char*)&val;, SEC_CHAR)) != 0 ) {
  perror ("Error retrieving port attributes");
  exit(1);
 }

 if (slhrtob(&(portlab->minsl), val)) {
  fprintf(stderr, "port minsl conversion error¥n");
  exit (1);
 }

 if ( (rc = getportattr(portname,S_MAXSL, (char*)&val;, SEC_CHAR)) != 0 ) {
  perror ("Error retrieving port attributes");
  exit(1);
```

```
  }

  if (slhrtob(&(portlab->maxsl), val)) {
   fprintf(stderr, "port maxsl conversion error¥n");
   exit (1);
  }

  if ( (rc = getportattr(portname,S_TL, (char*)&val;, SEC_CHAR)) != 0 ) {
   perror ("Error retrieving port attributes");
  }

  if (tlhrtob(&(portlab->tl), val)) {
   fprintf(stderr, "port tl conversion error¥n");
   exit (1);
  }

  return;
 }

 void displayuseraccess (char *username, struct userlabels *usrlab, struct portlabels *portlab)
 {
  CMP_RES_T cmpres;
  cmpres = sl_cmp(&(usrlab->defsl), &(portlab->minsl));
  if (cmpres != LAB_DOM && cmpres != LAB_SAME) {
   printf("Default SL of user does not dominate the minimum SL of tty ¥n");
   exit(1);
  }

  cmpres = sl_cmp(&(portlab->maxsl), &(usrlab->defsl));
  if (cmpres != LAB_DOM && cmpres != LAB_SAME) {
   printf("Default SL of user is not dominated by maximum SL of tty ¥n");
   exit(1);
  }

  cmpres = tl_cmp(&(portlab->tl), &(usrlab->deftl));
  if (cmpres != LAB_SAME) {
   printf("Default TL of user is not same as TL of tty ¥n");
   exit(1);
  }

  printf("The user can login on the specified port¥n");
  return;

}
```

***Trusted AIX* システム・コール:**

追加の Trusted AIX 機能を取り扱うためにシステム・コールが提供されます。

eaccept
ソケットによる接続を受け入れる

ebind セキュリティー属性を取り扱うために拡張されたバインド

econnect
セキュリティー属性を取り扱うために拡張されたソケットによる接続を開始する

eread ストリームから読み取り、メッセージ・セキュリティー属性を取り出す

ereadv
ストリームから読み取り、メッセージ・セキュリティー属性を取り出す

erecv セキュリティー属性を取り扱うために拡張された recv、recvfrom、recvmsg

erecvfrom
セキュリティー属性を取り扱うために拡張された recv、recvfrom、recvmsg

erecvmsg
セキュリティー属性を取り扱うために拡張された recv、recvfrom、recvmsg

esend セキュリティー属性を取り扱うために拡張された send、sendto、sendmsg

esendmsg
セキュリティー属性を取り扱うために拡張された send、sendto、sendmsg

esendto
セキュリティー属性を取り扱うために拡張された send、sendto、sendmsg

ewrite
ストリームに書き込み、メッセージ・セキュリティー属性を設定する

ewritev
ストリームに書き込み、メッセージ・セキュリティー属性を設定する

sec_getmsgsec
メッセージ・キューのセキュリティー属性を取得する

sec_getpsec
プロセスに関連付けられたセキュリティー情報を取得する

sec_getrunmode
カーネルの操作モードを取り出す

sec_getsecconf
現在のセキュリティー構成フラグを返す

sec_getsemsec
セマフォーのセキュリティー属性を取得する

sec_getshmsec
共有メモリー・セグメントのセキュリティー属性を取得する

sec_getsyslab
デフォルトのシステム機密ラベルを取得する

sec_gettlibbufsize
カーネル内のライブラリー・パス・エントリーを取り出す

sec_gettlibpath
カーネル内のライブラリー・パス・エントリーを取り出す

pdmkdir
分割ディレクトリーまたはサブディレクトリーを作成/設定/設定解除する

sec_setauditrange
システム・グローバル監査ラベルの範囲を設定する

sec_setplab
指定されたプロセスの有効機密ラベル、最小機密認可、最大機密認可、および保全性ラベルを設定する。

setppdmode
プロセスの分割ディレクトリー・モード (実または仮想) を設定する

setppriv
プロセスに関連付けられた特権セットを設定する

sec_setptlibmode
プロセスの TLIB モードを設定する

sec_setrunmode
カーネルの操作モードを設定する

sec_setsecconf
カーネル・セキュリティー構成フラグを設定する

sec_setsemlab
セマフォーのセキュリティー属性を設定する

sec_setshmlab
共有メモリー・セグメントのセキュリティー属性を設定する

sec_setsyslab
デフォルトのシステム機密、情報、および保全性ラベルを設定する

AIX C ライブラリー機能**:**

サブルーチンおよびマクロが追加の Trusted AIX 機能を取り扱うために提供されます。

accredrange
機密ラベルが認定の範囲内にあるかどうかを判別する。

clbtohr
提供されたバイナリー認可ラベルを人間に解読可能なフォーマットに変換する。

clhrtob
提供された人間に解読可能な認可ラベルをバイナリー・フォーマットに変換する。

getfsfbitindex、getfsfbitstring
ファイル・セキュリティー・フラグの文字列と索引を取得するルーチン

getmax_sl、getmax_tl
ラベル・エンコード・ファイルから最大機密および保全性ラベルを取り出す。

getmin_sl、getmin_tl
ラベル・エンコード・ファイルから最小機密および保全性ラベルを取り出す。

getsecconfig、setsecconfig
runmodes のカーネル・セキュリティー構成フラグを取り出して設定するルーチン

initlabeldb、endlabeldb
ラベル・データベースの初期化および終了ルーチン

maxlen_sl、maxlen_cl、maxlen_tl
初期設定されたラベル・エンコード・ファイルに基づいて、人間に解読可能ラベルの最大長を取り出す。

priv_isnull
特権が提供された特権セットに設定されているかどうかを判別する。

priv_lower
特権セット操作

priv_raise
特権セット操作

priv_remove
特権セット操作

priv_subset
特権セット操作

privbit_clr
指定された特権セットの指定特権を消去する。

priv_clrall
指定された特権セットのすべての特権を消去する。

priv_comb
最初の 2 つに指定された特権セットを結合して、結果を 3 番目に指定された特権セットに置く。

priv_copy
最初に指定された特権セットを 2 番目に指定された特権セットへコピーする。

priv_isnull
特権が提供された特権セットにないかどうかを判別する。

priv_mask
最初の 2 つに指定された特権セットの論理積を計算して、結果を 3 番目に指定された特権セットに置く。

priv_rem
2 番目に指定された特権セットを最初に指定された特権セットから除去して、結果を 3 番目に指定された特権セットに置く。

privbit_set
指定された特権セットに指定特権を設定する。

priv_setall
指定された特権セットのすべての特権を設定する。

priv_subset
最初に指定された特権セットが 2 番目に指定された特権セットのサブセットであるかどうかを判別する。

privbit_test
指定された特権が指定された特権セットに設定されていることをテストして確認する。

slbtohr、clbtohr、tlbtohr
バイナリー・ラベルから人間に解読可能ラベルへの変換ルーチン

slhrtob、clhrtob、tlhrtob
人間に解読可能ラベルからバイナリー・ラベルへの変換ルーチン

sl_clr、tl_clr
ラベルをリセットするルーチン

sl_cmp、tl_cmp
ラベル比較ルーチン

tl_cmp
保全性ラベルを比較する

Trusted AIX 特権

Trusted AIX では、以下の特権を使用できます。 各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

監査特権**:**

Trusted AIX では、以下の監査特権を使用できます。各特権の概要と記述およびその使用法を示します。一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_AU_
結合される他のすべての **PV_AU_** 特権と同等です

PV_AU_ADD
プロセスが監査レコードを記録/追加することを許可します

PV_AU_ADMIN
プロセスが監査システムを構成および照会することを許可します

PV_AU_PROC
プロセスがプロセスの監査状態を取得および設定することを許可します

PV_AU_READ
監査ファイルとしてマークを付けられたファイルをプロセスが読み取ることを許可します

PV_AU_WRITE
監査ファイルとしてマーク付けされたファイルの書き込みまたは削除、あるいはファイルへの監査ファイルとしてのマーク付けをプロセスが行うことを許可します

権限特権**:**

Trusted AIX では、以下の権限特権を使用できます。各特権の概要と記述およびその使用法を示します。一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_AZ_ADMIN
プロセスがカーネルのセキュリティー・テーブルを変更することを許可します

PV_AZ_READ
プロセスがカーネルのセキュリティー・テーブルを取得することを許可します

PV_AZ_ROOT
プロセスは **exec** システム・コール時に許可検査を受け渡します

PV_AZ_CHECK
プロセスがすべての許可検査を受け渡すことを許可します

DAC 特権:

Trusted AIX では、以下の DAC 特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_DAC_
結合される他のすべての PV_DAC_ 特権と同等です

PV_DAC_O
プロセスが DAC 所有権制限をオーバーライドすることを許可します

PV_DAC_R
プロセスが DAC 読み取り制限をオーバーライドすることを許可します

PV_DAC_W
プロセスが DAC 書き込み制限をオーバーライドすることを許可します

PV_DAC_X
プロセスが DAC 実行制限をオーバーライドすることを許可します

PV_DAC_UID
プロセスがそのユーザー ID (UID) を設定または変更することを許可します

PV_DAC_GID
プロセスがそのグループ ID (GID) を設定または変更することを許可します

PV_DAC_RID
プロセスがそのロール ID (RID) を設定または変更することを許可します

ファイルシステム特権**:**

Trusted AIX では、以下のファイルシステム特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_FS_
結合される他のすべての **PV_FS_** 特権と同等です

PV_FS_MKNOD
プロセスが **mknod** システム・コールを実行して任意のタイプのファイルを作成することを許可します

PV_FS_MOUNT
プロセスがファイルシステムをマウントおよびアンマウントすることを許可します

PV_FS_CHOWN
プロセスがファイルの所有権を変更することを許可します

PV_FS_QUOTA
プロセスがディスク・クォータに関連した情報を管理することを許可します

PV_FS_LINKDIR
プロセスがディレクトリーへのハード・リンクを作成することを許可します

PV_FS_RESIZE
プロセスがファイルシステムでの操作の拡張および縮小を実行することを許可します

PV_FS_CNTL
ファイルシステムの拡張および縮小を除いて、プロセスが各種制御操作を実行することを許可します

PV_FS_CHROOT
プロセスがそのルート・ディレクトリーを変更することを許可します

PV_FS_PDMODE
プロセスが分割タイプのディレクトリーを作成または設定することを許可します

プロセス特権**:**

Trusted AIX では、以下のプロセス特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_PROC_
結合される他のすべての **PV_PROC** 特権と同等です

PV_PROC_PRIO
プロセス/スレッドが優先順位、ポリシー、およびその他のスケジューリング・パラメーターを変更することを許可します

PV_PROC_CORE
プロセスがコアをダンプすることを許可します

PV_PROC_RAC
プロセスがユーザー当たりの制限より多いプロセスを作成することを許可します

PV_PROC_RSET
リソース・セット (**rset**) をプロセスまたはスレッドに接続することを許可します

PV_PROC_ENV
プロセスがユーザー情報をユーザー構造体に設定することを許可します

PV_PROC_CKPT
プロセスが別のプロセスのチェックポイントを取るかまたは再始動することを許可します

PV_PROC_CRED
プロセスがプロセスの資格情報属性を設定することを許可します

PV_PROC_SIG
プロセスが関連付けられていないプロセスへシグナルを送信することを許可します

PV_PROC_PRIV
プロセスがプロセスと関連付けられた特権セットを変更または表示することを許可します

PV_PROC_TIMER
プロセスが精度の高いタイマーを送信および使用することを許可します

PV_PROC_RTCLK
プロセスが CPU 時間クロックをアクセスすることを許可します

PV_PROC_VARS
プロセスがプロセスのチューナブル・パラメーターを取得および更新することを許可します

PV_PROC_PDMODE
プロセスが分割ディレクトリーの REAL モードを変更することを許可します

カーネル特権**:**

Trusted AIX では、以下のカーネル特権を使用できます。 各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_KER_
結合される他のすべての **PV_KER_** 特権と同等です

PV_KER_ACCT
プロセスがアカウンティング・サブシステムに関する制限付き操作を実行することを許可します

PV_KER_DR
プロセスが動的再構成操作を起動することを許可します

PV_KER_TIME
プロセスがシステム・クロックおよびシステム時刻を変更することを許可します

PV_KER_RAC
プロセスが共有メモリー・セグメント用にラージ (ページング不可) ページを使用することを許可します

PV_KER_WLM
プロセスが WLM 構成を初期化および変更することを許可します

PV_KER_EWLM
プロセスが eWLM 環境を初期化または照会することを許可します

PV_KER_VARS
プロセスがカーネル・ランタイムのチューナブル・パラメーターをテストまたは設定することを許可します

PV_KER_REBOOT
プロセスがシステムをシャットダウンすることを許可します

PV_KER_RAS
プロセスが RAS レコード、エラー・ロギング、トレース、およびダンプ機能を構成または書き込むことを許可します

PV_KER_LVM
プロセスが LVM サブシステムを構成することを許可します

PV_KER_NFS
プロセスが NFS サブシステムを構成することを許可します

PV_KER_VMM
プロセスがカーネル内でスワップ・パラメーターおよびその他の VMM チューナブル・パラメーターを変更することを許可します

PV_KER_WPAR
プロセスがワークロード・パーティションを構成することを許可します

PV_KER_CONF
プロセスが各種のシステム構成操作を実行することを許可します

PV_KER_EXTCONF
プロセスがカーネル・エクステンション内で各種の構成タスクを実行することを許可します

PV_KER_IPC
プロセスが IPC メッセージ・キュー・バッファーの値を大きくすること、および範囲指定された **shmget** システム・コールの接続を許可します

PV_KER_IPC_R
プロセスが IPC メッセージ・キュー、セマフォー・セット、または共有メモリー・セグメントを読み取ることを許可します

PV_KER_IPC_W
プロセスが IPC メッセージ・キュー、セマフォー・セット、または共有メモリー・セグメントに書き込むことを許可します

PV_KER_IPC_O
プロセスがすべての IPC オブジェクトで DAC 所有権をオーバーライドすることを許可します

PV_KER_SECCONFIG
プロセスがカーネル・セキュリティー・フラグを設定することを許可します

PV_KER_PATCH
プロセスがカーネル・エクステンションをパッチすることを許可します

ラベル特権**:**

Trusted AIX では、以下のラベル特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_LAB_
結合される他のすべてのラベル特権 (**PV_LAB_***) と同等です

PV_LAB_CL
プロセスの認可に応じて、プロセスがサブジェクト SCL を変更することを許可します

PV_LAB_CLTL
プロセスの認可に応じて、プロセスがサブジェクト TCL を変更することを許可します

PV_LAB_LEF
プロセスがラベル・データベースを読み取ることを許可します

PV_LAB_SLDG
プロセスの認可に応じて、プロセスが SL をダウングレードすることを許可します

PV_LAB_SLDG_STR
プロセスの認可に応じて、プロセスがパケットの SL をダウングレードすることを許可します

PV_LAB_SL_FILE
プロセスの認可に応じて、プロセスがオブジェクト SL を変更することを許可します

PV_LAB_SL_PROC
プロセスの認可に応じて、プロセスがサブジェクト SL を変更することを許可します

PV_LAB_SL_SELF
プロセスの認可に応じて、プロセスがそれ自体の SL を変更することを許可します

PV_LAB_SLUG
プロセスの認可に応じて、プロセスが SL をアップグレードすることを許可します

PV_LAB_SLUG_STR
プロセスの認可に応じて、プロセスがパケットの SL をアップグレードすることを許可します

PV_LAB_TL
プロセスがサブジェクトおよびオブジェクト TL を変更することを許可します

MAC 特権**:**

Trusted AIX では、以下の MAC 特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは

PV_AU_ADMIN、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_MAC_
結合される他のすべての MAC 特権 (**PV_MAC_***) と同等です

PV_MAC_CL
プロセスが機密性認可の制限をバイパスすることを許可します

PV_MAC_R_PROC
ターゲット・プロセスのラベルが活動中のプロセスの認可内にあれば、プロセスに関する情報を読み取るときに、プロセスが MAC 読み取り制限をバイパスすることを許可します

PV_MAC_W_PROC
ターゲット・プロセスのラベルが活動中のプロセスの認可内にあれば、プロセスにシグナルを送信するときに、プロセスは MAC 書き込み制限をバイパスすることを許可します

PV_MAC_R
プロセスが MAC 読み取り制限をバイパスすることを許可します

PV_MAC_R_CL
オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 読み取り制限をバイパスすることを許可します

PV_MAC_R_STR
メッセージのラベルがプロセスの認可の範囲内にあれば、 STREAM からメッセージを読み取るときに、プロセスが MAC 読み取り制限をバイパスすることを許可します

PV_MAC_W
プロセスが MAC 書き込み制限をバイパスすることを許可します

PV_MAC_W_CL
オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 書き込み制限をバイパスすることを許可します

PV_MAC_W_DN
プロセス・ラベルがオブジェクトのラベルより上位にあり、オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 書き込み制限をバイパスすることを許可します

PV_MAC_W_UP
プロセス・ラベルがオブジェクトのラベルより下位にあり、オブジェクトのラベルがプロセスの認可の範囲内であるときに、プロセスが MAC 書き込み制限をバイパスすることを許可します

PV_MAC_OVRRD
MAC の対象外であるとフラグを立てられたファイルに対して、MAC 制限をバイパスします

MIC 特権:

Trusted AIX では、以下の MIC 特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは

PV_AU_ADMIN、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_MIC
プロセスが保全性制限をバイパスすることを許可します

PV_MIC_CL
プロセスが保全性認可の制限をバイパスすることを許可します

ネットワーク特権**:**

Trusted AIX では、以下のネットワーク特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_NET_
結合される他のすべてのネットワーク特権 (**PV_NET_***) と同等です

PV_NET_CNTL
プロセスがネットワーク・テーブルを変更することを許可します

PV_NET_PORT
プロセスが制限ポートにバインドすることを許可します

PV_NET_RAWSOCK
プロセスがネットワーク層への直接アクセス権限を持つことを許可します

PV_NET_CONFIG
プロセスがネットワーク・パラメーターを構成することを許可します

スーパーユーザー特権**:**

Trusted AIX では、以下のスーパーユーザー特権を使用できます。各特権の概要と記述およびその使用法を示します。 一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_SU_
結合される他のすべてのスーパーユーザー特権 (**PV_SU_***) と同等です

PV_SU_ROOT
標準スーパーユーザーに関連付けられたすべての同等の特権をプロセスに付与します

PV_SU_EMUL
プロセス UID が 0 の場合に、標準スーパーユーザーに関連付けられたすべての同等の特権をプロセスに付与します

PV_SU_UID
getuid システム・コールでは 0 を戻すようにします

各種特権:

Trusted AIX では、以下の各種特権を使用できます。各特権の概要と記述およびその使用法を示します。一部の特権は階層を形成します。この場合、ある特権が別の特権に関連した権限をすべて付与できます。

特権を検査する際、システムはまずプロセスが必要最低特権を持っているかを判別するために検査し、次に階層を上に進んでより強力な特権があるかを検査します。 例えば、**PV_AU_** 特権のあるプロセスは **PV_AU_ADMIN**、**PV_AU_ADD**、**PV_AU_PROC**、**PV_AU_READ**、および **PV_AU_WRITE** 特権を自動的に保持し、**PV_ROOT** 特権のあるプロセスは下にある特権を **PV_SU_** 特権以外すべて自動的に保持します。

PV_ROOT
PV_SU_ (および **PV_SU_** が支配する特権) を除く、他のすべての同等の特権をプロセスに付与します

PV_TCB
プロセスがカーネルのトラステッド・ライブラリー・パスを変更することを許可します

PV_TP
プロセスがトラステッド・パス・プロセスであることを示し、トラステッド・パス・プロセスに制限されるアクションを許可します

PV_TP_SET
プロセスがカーネルのトラステッド・パス・フラグを設定またはクリアすることを許可します

PV_WPAR_CKPT
プロセスがワークロード・パーティションでチェックポイント・リスタート操作を実行することを許可します

PV_DEV_CONFIG
プロセスがシステムのカーネル・エクステンションとデバイスを構成することを許可します

PV_DEV_LOAD
プロセスがシステムのカーネル・エクステンションとシステムのデバイスをロードおよびアンロードすることを許可します

PV_DEV_QUERY
プロセスがカーネル・モジュールを照会することを許可します

Trusted AIX のトラブルシューティング

よくある質問への応答が、Trusted AIX のトラブルシューティングに役立つ場合があります。

どのように **Trusted AIX** にログインしたらよいですか。
Trusted AIX では、適切なロールを用いてインストール時に下記の 3 つの管理ユーザーを作成します。

これらのアカウントに対するパスワードは、Trusted AIX のインストール後に最初にシステムを起動するときに設定しなければなりません。 ネットワークから非プロンプト・モードでシステムをインストールした場合は、次のパスワードがデフォルトのアカウントに設定されます。

ユーザー	パスワード
isso	isso
sa	sa
so	so

どのようにして **su** を **root** に対して指定したらよいですか。

Trusted AIX のインストール時に、root の **su** 属性が **false (偽)** に設定されているので、どのユーザーもこのアカウントにアクセスできません。 このアカウントにアクセスするためには、デフォルトの管理ユーザーの isso および sa は、**chuser** コマンドを使用してルート・アカウントのこの属性を **true (真)** に変更する必要があります。

su がルートに使用可能でルート・アカウントのパスワードが設定されていない場合は、システム上のいずれのユーザーもルート・アカウントにアクセスすることができます。 これを避けるためには、**su** 属性をリセットする前にルート・アカウントのパスワードを設定することをお勧めします。

自分用の管理ユーザーを作成するべきですか、それともデフォルトの管理ユーザーを使用するべきですか。

デフォルトの管理ユーザーは、カスタマイズの目的でシステムを設定するためだけのものです。これらのアカウントはシステムのカスタマイズにのみ使用することを強くお勧めします。ただし、必ずしも必要というわけではありません。

適切なロールの isso、sa、および so を用いてユーザー自身の 3 つの管理ユーザーを作成して、これらのデフォルトのユーザーを削除または使用不可にしてください。

なぜかシステムにログインできません。

ルート (uid 0 のアカウント) または 128 未満の uid をもつアカウントでログインしようとする場合は、アクセスは拒否されます。 これらのアカウントはシステム・アカウントと呼ばれます。システム・アカウントにアクセスするには、非システム・アカウント・ユーザーとしてログインし、アカウントに **su** を指定する必要があります。

ログイン時に表示されるラベル・エンコード・ファイルに関するエラーはありますか。

ラベル・エンコード・ファイルが破壊された場合は、root ユーザーとしてシングル・ユーザー・モードに入らなければなりません。 ルート・アカウントは、シングル・ユーザー・モードでのみアクセス可能です。

labck コマンドを使用してラベル・エンコード・ファイル (/etc/security/enc/LabelEncodings) が適切かどうかを検査してください。 ファイルが適切でない場合は、ファイルを変更し、シングル・ユーザー・モードを終了する前に **labck** コマンドを使用して再検査を行います。

対話モード (**trustchk -t ALL**) で **trustchk** を実行して、システムの状態の妥当性検査を行ってください。

Trusted AIX ライブラリー **API** を使用する **Trusted Trusted AIX** でプログラムのコンパイルを行うことができない理由は何ですか。

開発ツールキットがデフォルトでインストールされていません。 インストール・メディアから bos.mls.adt ファイルセットをインストールする必要があります。

コマンドの特権に対して行った変更によって、コマンドが正常に働かなくなってしまった場合、どのように訂正したらよいですか。

当該コマンドに対して対話モード (**trustchk -t**) で **trustchk** を実行して、特権を修正してください。

なぜか **/etc/security/enc** ディレクトリーにアクセスできません。

/etc/security/enc ディレクトリーにアクセスするには、シェルで PV_LAB_LEF および PV_MAC_R 特権が必要です。 ご使用のシェルにこれらの特権を割り当ててください。

ブート時にどのようにして **trustchk** を使用不可にしますか。

/etc/rc.mls スクリプトの中の trustchk 行を除去するかまたはこの行に注釈を付けてください。

システムでブートのたびにブート確認プロンプトが出るのを防ぐには、どうしたらよいですか。

ご使用のシステムのブート確認が有効になっている可能性があります。 Trusted AIX サブメニューの SMIT メニューを使用して、このブート確認を無効にすることができます。

ファイルシステム・オブジェクトの **SL** を変更しようとしても変更されません。

次のいくつかの可能性があります。

/usr/sbin/settxattr は何らかのエラー・メッセージを戻しましたか。

戻しているのであれば、次のことを確認して詳細を調べてください。 次に例を示します。

/usr/sbin/settxattr を実行する権限はありますか。

権限がない場合は、ユーザーの特権および権限を確認してください。

構文は正しいですか。

構文については、**settxattr** マニュアル・ページを参照してください。

要求された **SL** またはその省略形が存在しますか。

"con a b" の要求は、デフォルトのラベル・エンコード・ファイル (/etc/security/enc/LabelEncodings) によってシステム上で機能しますが、"conf a b" の要求は、どちらも "confidential compartment A compartment B" の論理省略形に見えても機能しません。

複数語のラベルに引用符を使用する必要がありましたか。

settxattr -f sl=con <*filename*> は機能し、settxattr -f -a sl="con a b" <*filename*> も機能しますが、settxattr -a sl=con a b <*filename*> は機能しません。

settxattr は何らかのエラー・メッセージを戻しましたか。

エラー・メッセージが戻されていない場合は、ファイルシステム・オブジェクトがシンボリック・リンクである可能性があります。 変更しようとしているオブジェクトがシンボリック・リンクである場合は、リンク自体の SL を変更するのかまたはそのリンクが指しているオブジェクトを変更するのかをまず確認してください。 **settxattr** は、リンクに従う代わりにリンク自体のラベルを設定します。

サード・パーティー・アプリケーションをインストールしてそれをシステム上で正しく機能させるには、どうしたらよいですか。

サード・パーティー・アプリケーションをインストールしたのにそれが正しく機能しない場合は、特別な特権を必要とする特定の制限付きファイルまたはディレクトリーにアクセスしようとしている可能性があります。 アプリケーションでこれらの制限付きオブジェクトにアクセスする必要があるかどうかを評価した後、必要とされる特権を次のように決定してください。

- ご使用のシェルに PV_ROOT を割り当てる
- tracepriv -f -e <*third party command*> を実行する

こうするとアプリケーションに必要な特権がリストされます。 **setsecattr** コマンドを使用して、これらの特権を特権コマンド・データベースに追加してください。

なぜか特定のコマンドを実行できません。

ほとんどのコマンドは権限によって保護されているので、一部の特権コマンドの実行は、呼び出し

側のユーザーが対応する権限をもっている場合にのみ可能になります。 このことは、コマンドの実行に必要な権限が現在のセッションのために活動化されているロールの 1 つに存在しているかどうかを識別することによって確認することができます。

`rolelist -ae` でアクティブ権限を確認し、`lssecattr -c <command>` を使用してコマンドが必要としている権限を確認してください。

ラベルが適切に表示されないコマンドがあります。

このようなコマンドのほとんどは、ファイル `/etc/security/enc/LabelEncodings` に依存して、ラベルを人間が読み取れる形式に変換したり、その逆の変換を行います。 このファイルが破壊された場合または変更された場合は、これらのコマンドは目的どおりには機能しないことがあります。

ファイル・セキュリティー・フラグ

ファイル・セキュリティー・フラグはファイルがアクセスされる方法に影響を与えます。これらのフラグはファイル自体の拡張属性 (EA) の一部として保管されます。 ファイル・セキュリティー・フラグはヘッダー・ファイルに定義されます。

FSF_APPEND
ファイルは操作モードで追加のみ可能であり、変更することはできない。

FSF_AUDIT
ファイルは監査サブシステムの一部としてマークを付けられる。 これらのファイルの読み取りまたは書き込みを行うには、プロセスに **PV_AU_READ** または **PV_AU_WRITE** 特権をそれぞれ所有する必要があります。

FSF_MAC_EXMPT
PV_MAC_OVRRD 特権を持つ EPS は、オブジェクトのアクセスを試みるときに、MAC 制限を無視する。

FSF_PDIR
ディレクトリーは分割ディレクトリーである。

FSF_PSDIR
ディレクトリーは分割サブディレクトリーである。

FSF_PSSDIR
ディレクトリーは分割サブ-サブディレクトリーである。

FSF_TLIB
オブジェクトはトラステッド・ライブラリーの一部としてマークを付けられる。 マシンは構成モードで稼働するか、または **trustedlib_enabled** カーネル・セキュリティー・フラグを OFF にする必要があります。

FSF_TLIB_PROC
TLIB プロセスとしてマークを付けられたプロセスは、**TLIB** フラグ・セットを持つ oto ***.so** ライブラリーのみリンクできる。 システムは構成モードで稼働するか、または **trustedlib_enabled** カーネル・セキュリティー・フラグを OFF にする必要があります。

Trusted AIX コマンド

Trusted AIX システムを管理するために、セキュリティー関連コマンドが提供されています。

labck LabelEncodings ファイルを検証します

getsecconf
カーネル・セキュリティー・フラグを表示します

setsecconf
Trusted AIX カーネル・セキュリティー・フラグを変更します

getsyslab
カーネルの最大ラベルと最小ラベルを表示します

setsyslab
カーネルの最大ラベルと最小ラベルを設定します

getrunmode
システムの現行実行モードを表示します

setrunmode
システムの実行モードを切り替えます

pdlink
分割サブディレクトリーをまたがってファイルをリンクします

pdmkdir
分割ディレクトリーとサブディレクトリーを作成します

pdmode
現行の分割ディレクトリーのアクセス・モードを戻すか、指定された分割ディレクトリーのアクセス・モードでコマンドを実行します

pdrmdir
分割ディレクトリーと関連サブディレクトリーを除去します

pdset 分割ディレクトリーとサブディレクトリーを設定または設定解除します

bootauth
権限があるユーザーがシステムをブートしていることを検証します

chuser
ユーザーの認可属性を変更します

lsuser ユーザーの認可属性を表示します

chsec ユーザーの認可属性とポート・ラベルを変更します

lssec ユーザーの認可属性とポート・ラベルを表示します

trustchk
ファイルの属性を検査します

lstxattr
ファイル、プロセス、および IPC オブジェクトのラベルとセキュリティー・フラグ属性を表示します

settxattr
ファイル、プロセス、および IPC オブジェクトのラベルとセキュリティー・フラグ属性を変更します

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒*103-8510*
東京都中央区日本橋箱崎町*19*番*21*号
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態で提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。 IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

記載されている性能データとお客様事例は、例として示す目的でのみ提供されています。実際の結果は特定の構成や稼働条件によって異なります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。 IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確証できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願いします。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があり、単に目標を示しているものです。

表示されている IBM の価格は IBM が小売り価格として提示しているもので、現行価格であり、通知なしに変更されるものです。卸価格は、異なる場合があります。

本書はプランニング目的としてのみ記述されています。記述内容は製品が使用可能になる前に変更になる場合があります。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名前はすべて架空のものであり、類似する個人や企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほのめかしたり、保証することはできません。これらのサンプル・プログラムは特定物として現存するままの状態で提供されるものであり、いかなる保証も提供されません。 IBM は、お客様の当該サンプル・プログラムの使用から生ずるいかなる損害に対しても一切の責任を負いません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生した創作物には、次のように、著作権表示を入れていただく必要があります。

© (お客様の会社名) (西暦年).

このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。

© Copyright IBM Corp. _年を入れる_.

プライバシー・ポリシーに関する考慮事項

サービス・ソリューションとしてのソフトウェアも含めた IBM ソフトウェア製品 (「ソフトウェア・オファリング」) では、製品の使用に関する情報の収集、エンド・ユーザーの使用感の向上、エンド・ユーザーとの対話またはその他の目的のために、Cookie はじめさまざまなテクノロジーを使用することがあります。多くの場合、ソフトウェア・オファリングにより個人情報が収集されることはありません。 IBM の「ソフトウェア・オファリング」の一部には、個人情報を収集できる機能を持つものがあります。ご使用の「ソフトウェア・オファリング」が、これらの Cookie およびそれに類するテクノロジーを通じてお客様による個人情報の収集を可能にする場合、以下の具体的事項を確認ください。

この「ソフトウェア・オファリング」は、Cookie もしくはその他のテクノロジーを使用して個人情報を収集することはありません。

この「ソフトウェア・オファリング」が Cookie およびさまざまなテクノロジーを使用してエンド・ユーザーから個人を特定できる情報を収集する機能を提供する場合、お客様は、このような情報を収集するにあたって適用される法律、ガイドライン等を遵守する必要があります。これには、エンドユーザーへの通知や同意の要求も含まれますがそれらには限られません。

このような目的での Cookie などの各種テクノロジーの使用について詳しくは、『IBM オンラインでのプライバシー・ステートメントのハイライト』(http://www.ibm.com/privacy/jp/ja/)、『IBM オンラインでのプライバシー・ステートメント』(http://www.ibm.com/privacy/details/jp/ja/) の『クッキー、ウェブ・ビーコン、その他のテクノロジー』というタイトルのセクション、および『IBM Software Products and Software-as-a-Service Privacy Statement』(http://www.ibm.com/software/info/product-privacy) を参照してください。

商標

IBM、IBM ロゴおよび ibm.com は、世界の多くの国で登録された International Business Machines Corp. の商標です。他の製品名およびサービス名等は、それぞれ IBM または各社の商標である場合があります。現時点での IBM の商標リストについては、http://www.ibm.com/legal/copytrade.shtml をご覧ください。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

Java およびすべての Java 関連の商標およびロゴは、Oracle やその関連会社の米国およびその他の国における商標または登録商標です。

UNIX は、The Open Group の米国およびその他の国における登録商標です。

索引

日本語, 数字, 英字, 特殊文字の順に配列されています。なお, 濁音と半濁音は清音と同等に扱われています。

[ア行]

[カ行]

[サ行]

[タ行]

[ナ行]

［ハ行］

［マ行］

［ヤ行］

［ラ行］

A

B

C

D

E

F

I

K

L

M

N

O

P

R

S

T

V

W

X

[特殊文字]

Printed in Japan

日本アイ・ビー・エム株式会社
〒103-8510 東京都中央区日本橋箱崎町19-21