CA Access Control

Windows エンドポイント管理ガイド

12.6.02

サードパーティに関する通知

CONTAINS IBM(R) 32-bit Runtime Environment for AIX(TM), Java(TM) 2 Technology Edition, Version 1.4 Modules

サンプル スクリプトおよびサンプル SDK コード

CA Access Control 製品に含まれているサンプル スクリプトおよびサンプル SDK コードは、情報提供のみを目的として現状有姿のまま提供されます。これらは特定の環境で調整が必要な場合があるため、テストや検証を実行せずに実稼働システムにデプロイしないでください。

CA Technologies では、これらのサンプルに対するサポートを提供していません。また、これらのスクリプトによって引き起こされるいかなるエラーにも責任を負わないものとします。

CA Technologies 製品リファレンス

このマニュアルが参照している CA Technologies の製品は以下のとおりです。

- CA Access Control Enterprise Edition
- CA Access Control
- CA Single Sign-On （CA SSO）
- CA Top Secret®
- CA ACF2™
- CA Audit
- CA Network and Systems Management （CA NSM、旧 Unicenter NSM and Unicenter TNG）
- CA Software Delivery （旧 Unicenter Software Delivery）
- CA Service Desk （旧 Unicenter Service Desk）
- User Activity Reporting （旧 CA Enterprise Log Manager）
- CA Identity Manager

ドキュメントの表記規則

CA Access Control のドキュメントには、以下の規則があります。

形式	意味
等幅フォント	コードまたはプログラムの出力
斜体	強調または新規用語
太字	表示されているとおりに入力する必要のある要素
スラッシュ（/）	UNIX および Windows のパスの記述で使用される、プラットフォームに依存しないディレクトリの区切り文字

また、本書では、コマンド構文およびユーザ入力の説明に（等幅フォントで）以下の特殊な規則を使用します。

形式	意味
斜体	ユーザが入力する必要のある情報
角かっこ（[]）で囲まれた文字列	オプションのオペランド
中かっこ（{}）で囲まれた文字列	必須のオペランドセット
パイプ（|）で区切られた選択項目	代替オペランド（1 つ選択）を区切ります。 たとえば、以下の例は「*ユーザ名*または*グループ名*のいずれか」を意味します。 {*username*|*groupname*}
…	前の項目または項目のグループが繰り返し可能なことを示します
下線	デフォルト値
スペースに続く、行末の円記号（¥）	本書では、コマンドの記述が 1 行に収まらない場合があります。 このような場合、行末の空白とそれに続く円記号（¥）は、そのコマンドが次の行に続くことを示します。 **注**: このような円記号はコピーしないでください。また、改行はコマンドに含めないようにしてください。 これらの文字は、実際のコマンド構文の一部ではありません。

例: コマンドの表記規則

以下のコードは、本書でのコマンド表記規則の使用方法を示しています。

```
ruler className [props({all|{propertyName1[,propertyName2]...})]
```

この例の内容

- 標準的な等幅フォントで表示されているコマンド名（ruler）は表示されているとおりに入力します。
- 斜体で表示されている *className* オプションは、クラス名（USER など）のプレースホルダです。

- 2 番目の角かっこで囲まれた部分を指定しなくても、コマンドは実行できます。この部分は、オプションのオペランドを示します。
- オプションのパラメータ（props）を使用する場合は、キーワード *all* を選択するか、またはカンマで区切られたプロパティ名を 1 つ以上指定します。

ファイル ロケーションに関する規則

CA Access Control のドキュメントには、ファイル ロケーションに関する以下の規則があります。

- *ACInstallDir* -- CA Access Control のデフォルトのインストール ディレクトリ。
  - Windows -- <インストール パス>
  - UNIX -- <インストール パス 2>
- *ACSharedDir* -- CA Access Control for UNIX で使用される、デフォルトのディレクトリ。
  - UNIX -- /opt/CA/AccessControlShared
- *ACServerInstallDir* -- CA Access Control エンタープライズ管理 のデフォルトのインストール ディレクトリ。
  - /opt/CA/AccessControlServer
- *DistServerInstallDir* -- デフォルトの配布サーバ インストール ディレクトリ。
  - /opt/CA/DistributionServer
- *JBoss_HOME* -- デフォルトの JBoss インストール ディレクトリ。
  - /opt/jboss-4.2.3.GA

CA への連絡先

テクニカル サポートの詳細については、弊社テクニカル サポートの Web サイト（http://www.ca.com/jp/support/）をご覧ください。

マニュアルの変更点

以下のドキュメントのアップデートは、本書の最新のリリース以降に行われたものです。

- エンドポイントの管理 - 以下の変更されたセクションを含む章が更新されました。
  - インストルメンテーションの仕組み - CA Access Control によるインストルメンテーションの実行方法について説明するトピックが追加されました。
- Policy Model の管理 - 以下の変更されたセクションを含む章が更新されました。
  - Policy Model データベース（管理者ガイド WIN）
  - パスワードの伝達と同期

目次

第 1 章: 概要

このセクションには、以下のトピックが含まれています。

本書の内容

本書では、オープン システムに統合的なセキュリティ ソリューションを提供する CA Access Control for Windows に採用されているさまざまな概念について説明します。 本書では、Windows エンドポイントの管理タスクと概念について説明します。

また、エンタープライズ管理機能、レポート機能、および拡張ポリシー管理機能を備えた CA Access Control Enterprise Edition についても説明します。

用語を簡潔に示すために、本書の全体を通してこの製品を CA Access Control と呼びます。

本書の対象読者

本書は、CA Access Control で保護される環境の実装およびメンテナンスを担当するセキュリティ管理者およびシステム管理者を対象にしています。

第 2 章: エンドポイントの管理

CA Access Control は、オペレーティング システムと動的に連携し、アクティブで統合的なセキュリティ ソリューションをオープン システムに提供するソフトウェアです。 ファイルのオープン、ユーザ ID の変更、ネットワーク サービスの取得など、セキュリティ保護が必要な操作をユーザが要求するたびに、CA Access Control は各イベントをリアルタイムでインターセプトし、その妥当性を検証してから、オペレーティング システム（OS）標準機能に制御を渡します。

このセクションには、以下のトピックが含まれています。

CA Access Control とは何か

CA Access Control は、ネイティブ プラットフォームのセキュリティ管理を行うための強力なツールを提供し、企業のセキュリティ要件に合わせて完全にカスタマイズできるセキュリティ ポリシーの実装を可能にします。 CA Access Control を使用すると、ネイティブのオペレーティング システムでは実現できない強力なセキュリティをユーザ、グループ、およびリソースに対して提供できます。また、組織全体のセキュリティを集中管理し、マルチプラットフォーム環境において Windows と UNIX のセキュリティ ポリシーを統合できます。

保護の対象

CA Access Control は、以下のエンティティを保護します。

- ［ファイル］

  特定のファイルにアクセスする権限があるか?

  CA Access Control は、ファイルへのユーザのアクセスを制限します。ユーザに対して、READ、WRITE、EXECUTE、DELETE、RENAME などのアクセス権限を 1 種類以上与えることができます。 アクセス権限は、個々のファイルに対して、または類似した名前を持つファイルの集合に対して指定できます。

- **端末**

  特定の端末を使用する権限があるか?

  このチェックは、ログイン プロセスで行われます。 CA Access Control データベースに個々の端末または端末グループを定義し、アクセス ルールにより、その端末または端末グループの使用を許可されているユーザまたはユーザ グループを指定できます。端末を保護することによって、強力な権限を持つユーザ アカウントのログインに未許可の端末が使用されることを確実に防止します。

- **ログイン時間**

  ユーザには、特定の曜日の特定の時間にログインする権限があるか?

  通常、エンド ユーザは平日の勤務時間帯にのみ端末を使用します。そのため、平日の曜日と時間帯によるログイン制限、および休日のアクセス制限を行うことによって、ハッカーやその他の無許可のアクセサから端末を保護できます。

- **TCP/IP**

  相手の端末には、ローカル コンピュータから TCP/IP サービスを受け取る権限があるか? 相手の端末には、ローカル コンピュータに TCP/IP サービスを供給する権限があるか? 相手の端末は、ローカル端末のすべてのユーザからサービスを受け取ることを許可されているか?

  オープン システムの長所は、コンピュータとネットワークの両方がオープンであるという点ですが、これは同時に短所でもあります。いったんコンピュータが外部に接続されると、故意または過失により、外部ユーザがシステムに侵入したり、そのユーザが行った行為が損害をもたらしたりする危険が発生します。CA Access Control には、「ファイアウォール」が用意されており、ローカルの端末やサーバが不特定の端末へサービスを提供することを防止します。

- **複数ログイン権限**

  ユーザは他の端末からログインできるか?

  *同時ログイン*とは、ユーザが複数の端末からシステムにログインできることを意味します。CA Access Control では、1 人のユーザが複数の端末から同時にログインすることを防止できます。 これにより、すでにログインしているユーザのアカウントで外部からの侵入者がログインすることを防止できます。

- **ユーザ定義エンティティ**

  標準エンティティ（TCP/IP サービスや端末など）および機能エンティティ（トランザクションの実行やデータベース内のレコードへのアクセスなどの*抽象*オブジェクト）の両方を定義して保護できます。

- **管理者権限**

  CA Access Control には、管理者権限をオペレータに委任する方法、および管理者権限自体を制限する方法が用意されています。

- **レジストリ キー**

  ユーザには、特定のレジストリ キーにアクセスする権限があるか?

  CA Access Control は、レジストリ キーへのユーザのアクセスを制限します。 ユーザに対して、READ、WRITE、DELETE などのアクセス権限を 1 種類以上与えることができます。 アクセス権限は、個々のレジストリ キーに対して、または類似した名前を持つレジストリ キーの集合に対して指定できます。

- **プログラム**

  特定のプログラムを信頼できるか? ユーザには、このプログラムを起動する権限があるか? ユーザは、プログラムを使用して、特定のリソースにアクセスできるか?

  セキュリティ管理者は、プログラムをテストして、これらのプログラムに、アクセス権の不正取得に利用される可能性があるセキュリティ ホールがないことを確認できます。 テストで安全とみなされたプログラムは、trusted プログラムとして定義されます。 CA Access Control の自己防衛機能モジュール（**Watchdog** ともいう）は、ある特定の時点で制御の対象になっているプログラムを認識し、そのプログラムが、trusted と分類された後に変更または移動されたかどうかをチェックします。 trusted プログラムが変更または移動された場合、その時点で trusted とはみなされなくなり、CA Access Control はプログラムの実行を許可しません。

さらに、CA Access Control では、以下のような作為的または偶発的な脅威に対して防御を行います。

- **強制終了**

  CA Access Control では、重要なサーバやサービス、またはデーモンを強制終了から保護できます。

- **パスワード攻撃**

  CA Access Control はさまざまなタイプのパスワード攻撃からパスワードを保護します。サイトのパスワード定義ポリシーを適用し、パスワードの盗用による侵入を検知します。

- **不適切なパスワード**

  CA Access Control のポリシーでは、十分な品質のパスワードを作成して使用することをユーザに強制するルールが定義されます。 CA Access Control では、ユーザが基準に合ったパスワードを作成して使用することを確実にするために、最長および最短のパスワード有効期限の設定、特定の語句の使用制限、文字の繰り返しの禁止、およびその他の制限事項の適用を行うことができます。パスワードを長期間継続して使用することは認められません。

- **アカウント管理**

  CA Access Control のポリシーによって、休止状態のアカウントの適切な処理が保証されます。

保護の方法

CA Access Control は、オペレーティング システムの初期化が終了するとただちに開始されます。 CA Access Control によって、保護の必要なシステム サービスにフックが設定されます。 このようにして、サービスが実行される前に CA Access Control に制御が渡されます。 CA Access Control によって、サービスの使用をユーザに許可するかどうかが決定されます。

たとえば、CA Access Control によって保護されているリソースにユーザがアクセスしようとするとします。 このアクセス要求によって、カーネルに対してリソースのオープンを指示するシステム コールが生成されます。そのシステム コールは CA Access Control によってインターセプトされ、アクセスを許可するかどうかが決定されます。 アクセスが許可された場合は、CA Access Control によって通常のシステム サービスに制御が渡されます。アクセスが許可されない場合は、システム コールをアクティブにしたプログラムに、permission-denied 標準エラー コードが返され、システム コールの処理が終了します。

これは、データベースに定義されたアクセス ルールとポリシーに基づいて決定されます。データベースには、アクセサとリソースという 2 種類のオブジェクトが定義されています。 *アクセサ*とは、ユーザおよびグループのことです。 *リソース*とは、ファイルやサービスなど、保護対象のオブジェクトのことです。 データベース内の各レコードには、アクセサまたはリソースが定義されています。

各オブジェクトはクラスに属します。クラスは、同じタイプのオブジェクトの集合です。 たとえば、TERMINAL は、CA Access Control によって保護されている端末（ワークステーション）であるオブジェクトを含むクラスです。

クラスのアクティブ化

CLASS ステータスに関する（そのクラスがアクティブか非アクティブかを示す）情報は、データベースに格納されています。 リソースに対するアクセス試行はすべて CA Access Control によってインターセプトされ、データベース内のステータスがチェックされます。 クラスがアクティブでない場合は、それ以上の権限チェックは行われずにアクセスが許可されます。

CA Access Control は、エンジンの起動時、およびユーザによる CLASS のアクティビティ ステータスの変更時に、アクティブ クラスの一覧を発行します。クラスがアクティブでない場合、リソースへのアクセスはインターセプトされないので、オーバーヘッドが軽減されます。

アクセサ エレメント

各ユーザは、*アクセサ エレメント*（ACEE）として表されます。ACEE は、データベースに格納されているユーザのレコードをメモリ内に反映したものです。CA Access Control は、ログイン プロセス時にアクセサ エレメントを作成します。 アクセサ エレメントは、ユーザのプロセスと関連付けられます。CA Access Control によって保護されているシステム サービスをプロセスが要求するたびに、またはプロセスがリソースにアクセスするために暗黙的な要求を発行するたびに、CA Access Control はそのリソースのレコードにアクセスします。 そして次に、以前に作成されたアクセサ エレメントの情報（ユーザのセキュリティ レベル、モード、グループなど）から、ユーザがリソースへのアクセスを許可されているかどうかを判断します。

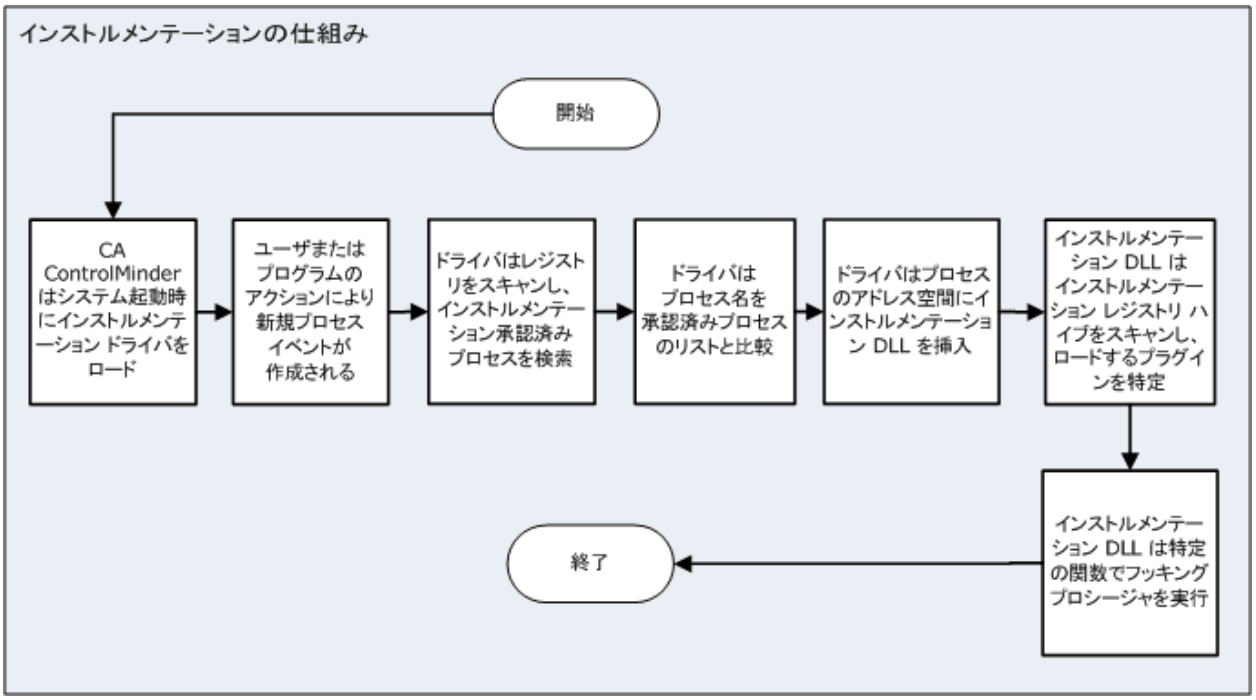
インストルメンテーションの仕組み

*インストルメンテーション*は、アプリケーションの実行フローの CA Access Control による監視、追跡、変更を実現する手法です。 インストルメンテーションにより、CA Access Control はシステム プロセスを監視し、アプリケーション アドレス空間で専用モジュールのインターセプトおよび実装を行うことができます。

インストルメンテーション プロセスには、*カーネル インストルメンテーション フェーズ*および*ユーザ モード インストルメンテーション フェーズ*という 2 つのフェーズがあります。

注: カーネル インターセプトおよびユーザ モード インターセプトの詳細については、「アカウントの保護」の章を参照してください。 インストルメンテーションの詳細については、「リファレンス ガイド」を参照してください。

以下の図に、インストルメンテーション プロセスを示します。



カーネル インストルメンテーション フェーズでは、CA Access Control により以下が実行されます。

1. CA Access Control はシステム起動時にインストルメンテーション ドライバ（cainstrm.sys）をロードします。
2. ユーザまたはプログラムのアクションの結果、新しいプロセス イベントが作成されます。
3. インストルメンテーション ドライバは一定の時間間隔でレジストリ ハイブをスキャンし、インストルメンテーション承認済みプロセスを探します。

   インストルメンテーションの ApplyonProcesses レジストリ キーを使用して、、インストルメンテーション承認済みプロセスのリストを指定します。インストルメンテーションのレジストリ キーの詳細については、「リファレンス ガイド」を参照してください。
4. CA Access Control は、新しいプロセス イベントを識別すると、承認済みプロセスのリストからプロセス名を検索します。プロセス名が見つかったら、ドライバはそのプロセスのアドレス空間にインストルメンテーション DLL を挿入します。

ユーザ モード インストルメンテーション フェーズでは、CA Access Control により以下が実行されます。

1. インストルメンテーション DLL は インストルメンテーションのレジストリ ハイブをスキャンしてプロセスのアドレス空間にロードするプラグインを特定し、以下のいずれかを実行します。
   - 見つかったすべてのプラグインを、プロセス メモリ アドレスにロードします。 手順 2 に進みます。
   - プラグインが見つからない場合、インストルメンテーション DLL は自身をアンロードします。
2. CA Access Control は Microsoft Detours ライブラリを使用して、各プラグインに含まれる特定の関数に基づいてフッキング プロシージャを実行します。

   Microsoft Detours は、Win32 関数のインストルメンテーション実行に使用するライブラリです。Microsoft Detours の詳細については、*Microsoft Detours の Web サイト* (http://www.microsoft.com/about/legal/en/us/intellectualproperty/iplicensing/programs/detours.aspx)を参照してください。

ネイティブ セキュリティの拡張

以下の CA Access Control の機能により、ネイティブ セキュリティが拡張されます。

スーパーユーザ アカウントの制限

通常、UNIX システムの root アカウントや Windows システムの Administrator アカウントなどオペレーティング システムを管理するユーザ（管理者）はシステム セットアップ時に自動的に作成される、事前定義されたアカウントです。 事前定義された各アカウントは、一連のシステム機能のセットを実行します。

root または Administrator のアカウントを持つユーザは、ユーザの作成、削除、および変更から、サーバのロック、環境設定の変更、およびシャットダウンまで、広範なタスクを実行できます。

これらのオペレーティング システムにおけるセキュリティ上の主なリスクの 1 つは、権限のないユーザがこれらのアカウントの持っている制御権を手に入れる可能性があることです。 このような事が発生した場合、システムは重大な危険にさらされることになります。

CA Access Control では、これらのアカウントに与える権限を制限して、これらのアカウントをメンバとして持つユーザ グループに属するユーザの権限を制限することができます。 これにより、オペレーティング システムの脆弱性をカバーします。

CA Access Control 管理者

CA Access Control のインストール時には、1 人以上の CA Access Control 管理者の名前を設定する必要があります。 CA Access Control 管理者には、ルール データベースのすべてまたは一部を変更する権限があります。 すべての権限を持つ管理者を最低 1 人は設定する必要があります。この管理者は、アクセス ルールを自由に変更または作成することができ、管理者のレベルを指定できます。

システムのユーザを定義した後、管理者以外のユーザに ADMIN 属性を割り当てることによって、管理者権限を割り当てることができます。

注: ADMIN 属性が割り当てられたユーザには、強力な権限が与えられます。このため、ADMIN ユーザの数は厳しく制限する必要があります。 また、1 人以上の CA Access Control 管理者の設定が終了した後に、スーパーユーザから ADMIN 属性を削除して、ネイティブのスーパーユーザの役割と CA Access Control 管理者 の役割を分離する方法もお勧めします。

CA Access Control では、常に最低 1 人のユーザがデータベースを管理する権限を持つ必要があるため、ADMIN 属性を持つ最後のユーザを削除することはできません。

CA Access Control 管理者がこのワークステーションから他のホストを管理する可能性がある場合は、そのホスト上のデータベースに、このワークステーションからの READ アクセス権と WRITE アクセス権の両方を管理者に与えるルールが定義されていることを確認してください。

サブ管理

CA Access Control には、サブ管理機能があります。 CA Access Control 管理者はこの機能を使用することで、一般ユーザに対して特定のクラスを管理できるようにする特定の権限を与えることができます。 このようなユーザをサブ管理者といいます。

たとえば、特定のユーザに対して、ユーザとグループを管理できる権限を与えることができます。

また、特定のクラスに対してだけではなく、そのクラスの指定されたレコードに対してアクセス権を許可することにより、より高いレベルのサブ管理を指定することもできます。

一般ユーザに与える管理者権限

CA Access Control では、管理者 グループのメンバでなくても管理タスクを実行できるように、必要な権限を一般ユーザ（管理者以外）に与えることができます。 このような細かい方法でタスクを委任できる（つまり、管理権限を付与できる）機能は、CA Access Control の最も重要な機能の 1 つです。

- SUDO クラスのレコードには、コマンド スクリプトが格納されています。ユーザは、付与された権限でそのスクリプトを実行できます。

- data プロパティの値はコマンド スクリプトです。この値は、省略可能なスクリプト パラメータ値を追加して変更することができます。
- SUDO クラスの各レコードは、あるユーザが別のユーザの権限を借用できるようにするためのコマンドを識別します。
- SUDO クラス レコードのキーは、SUDO レコードの名前です。 この名前は、ユーザが SUDO レコードでコマンドを実行する際に、コマンド名の代わりに使用されます。

ファイル保護の強化

CA Access Control では、論理ファイル名と絶対ファイル名の両方の形式がサポートされます。 たとえば、ファイル foo.txt が論理ドライブ D の ¥tmp ディレクトリに格納されており、論理名「D:」が物理ディスク 1、パーティション 0 に割り当てられている場合は、以下のように、論理ファイル名か絶対ファイル名のいずれかを使用して、CA Access Control データベースに対してファイルを定義します。

```
nr file D:¥tmp¥foo.txt
```

または

```
nr file ¥Device¥HardDisk1¥Partition1¥tmp¥foo.txt
```

注: 2 番目の形式を使用する場合は、ディスクの論理名が変更されても、ファイルは保護されたままになります。絶対ファイル名形式は、CA Access Control の汎用ファイル保護でもサポートされます。

CA Access Control では、サポートされている Windows オペレーティング システムで現在使用されているすべてのファイル システムが保護されます。最も一般的に使用されるファイル システムは、Windows File System（NTFS）と File Allocation Table（FAT）の 2 種類です。 CA Access Control では、CDFS（CD-ROM ファイル システム）もサポートしています。

CA Access Control により、File Allocation Table（FAT）に対する総合的なセキュリティ ソリューション、および NTFS や CDFS などその他のファイル システムに対する特別なセキュリティ レイヤが提供されます。

汎用ファイル保護

CA Access Control では、論理ファイル名形式と絶対ファイル名の両方がサポートされます。 絶対ファイル名形式は、CA Access Control の汎用ファイル保護でもサポートされます。

汎用ファイル保護により、指定したワイルドカード パターン（正規表現）に適合するすべてのファイルを保護できます。 指定したワイルドカード パターンに一致する名前のリソースが、指定した包括的なアクセス ルールによって保護されます。 CA Access Control では、ファイルを包括的に保護できます。

リソースが複数の包括的なアクセス ルールに一致する場合は、CA Access Control によって、ファイルに対して最も厳密に一致するルールが選択されます。

汎用ファイル保護の機能を使用すると、ほんのわずかなセキュリティ ルールを定義するだけで、保護の必要な多数のファイルを保護できます。

パスワード保護機能

Windows ネイティブ セキュリティにより、さまざまな方法でパスワードを保護し、パスワードの品質を強化できます。 Windows では以下の機能が提供されています。

- パスワードの最長有効期間の指定
- パスワードの最低文字数の指定
- ユーザのパスワード履歴を最大 24 件まで保存できます。
- ログインに繰り返し失敗した場合のアカウントのロックアウト
- パスワード変更前の Windows へのログオンの強制

CA Access Control では、同じルールが独自のメカニズムによって適用されます。 さらに、CA Access Control では、メインフレーム コンピュータとの双方向のパスワード同期機能が実装されています。

パスワード保護の強化

Windows ネイティブ セキュリティによって、非常に多くのユーザ パスワードに関する保護 (P. 28)が提供されます。 さらに、CA Access Control では、パスワード保護が大幅に拡張されているため、ハッカーによるパスワード盗用の可能性は極めて低くなりました。

CA Access Control を使用すると、より安全で確実なパスワードをユーザが選択するように、ルールを追加できます。 たとえば、最低限必要な英字、数字、特殊文字、小文字、または大文字の数を選択するようにユーザに要求できます。 また、置き換えられる旧パスワードと、ユーザが選択した新しいパスワードで、前者の文字列が後者の文字列に含まれないようにすることもできます。

Program Pathing

Program Pathing は、ファイルにアクセスするには特定のプログラムを介さなければならないことを要求する、ファイルに関連するアクセス ルールです。 Program Pathing により、機密ファイルのセキュリティを大幅に強化できます。 CA Access Control の Program Pathing を使用すると、システム内のファイルに対する保護を強化できます。

B1 セキュリティ レベル認証

CA Access Control には、セキュリティ レベル、セキュリティ カテゴリ、およびセキュリティ ラベルという「Orange Book」の B1 レベルの機能があります。

- データベースのアクセサとリソースには、セキュリティ レベルを割り当てることができます。 セキュリティ レベルは、1 から 255 までの整数です。アクセサのセキュリティ レベルが、リソースに割り当てられたセキュリティ レベル以上である場合にのみ、アクセサはリソースにアクセスできます。
- データベースのアクセサとリソースは、1 つ以上の *セキュリティ カテゴリ* に属することができます。 リソースに割り当てられているすべてのセキュリティ カテゴリにアクセサが属している場合のみ、そのアクセサはリソースにアクセスできます。

- セキュリティ ラベルは、特定のセキュリティ レベルを 0 個以上のセキュリティ カテゴリの集合に関連付けるための名前です。ユーザをセキュリティ ラベルに割り当てると、セキュリティ ラベルに関連付けられたセキュリティ レベルおよびセキュリティ カテゴリの両方がユーザに設定されます。

注: Orange Book の B1 レベルの機能の詳細については、「*実装ガイド*」を参照してください。

監査手順の設定

CA Access Control では、データベースに定義されている監査ルールに基づいて、アクセス拒否とアクセス許可のイベントに関する監査レコードが保存されます。 特定のイベントをログに記録するかどうかの決定は、以下のルールに基づいて行われます。

- すべてのアクセサおよびリソースに AUDIT プロパティがあり、このプロパティを設定すると、アクセスの成功または失敗、あるいはその両方のイベントをログに記録するかどうかを指定できる。さらに、アクセサの AUDIT プロパティでは、ログインの成功または失敗、あるいはその両方のイベントをログに記録するかどうかを指定できる。
- リソースまたはアクセサに AUDIT（ALL）属性が割り当てられている場合は、CA Access Control によって保護されているリソースに関するすべてのイベントが、アクセスが失敗したか成功したかにかかわらず、ログに記録される。
- CA Access Control によって保護されているリソースへのアクセスが成功し、ユーザまたはリソースに AUDIT （SUCCESS）が割り当てられている場合、イベントがログに記録される。
- CA Access Control によって保護されているリソースへのアクセスが失敗し、ユーザまたはリソースに AUDIT （FAIL）が割り当てられている場合、イベントがログに記録される。

システム監査担当者 (AUDITOR 属性が割り当てられているユーザ) のみが、ユーザおよびリソースに割り当てられた監査属性の変更などの監査タスクを実行できます。

リソースが警告モードの場合に、リソースのアクセス ルールに違反するアクセスが発生すると、警告モード監査レコードが生成されます。このレコードには、CA Access Control がリソースへのアクセスを許可したことが記述されます。

監査レコードによって、*監査ログ*（seos.audit）というファイルが構成されます。 監査ログの場所は、エラー ログの場所と同様にレジストリで指定されます。

監査ログ（およびエラー ログ）は、以下のレジストリ キーで指定されます。

```
HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥logmgr
```

監査ログはバイナリ ファイルであるため、編集または変更することはできません。ただし、CA Access Control エンドポイント管理 を使用することで、記録されたイベントを表示したり、時間制限やイベント タイプなどでイベントをフィルタ処理したりできます（また、seaudit ユーティリティを使用しても、同様のタスクを実行できます）。

後からイベントを調査できるように、古い監査ログおよびエラー ログをアーカイブ（バックアップ）することをお勧めします。

Unicenter TNG への監査イベントの送信

Unicenter TNG との統合は、インストール時に設定します。

監査データを Unicenter TNG に送信するか、または Unicenter TNG から CA Access Control を起動できるようにするか、あるいはその両方を行うかを選択できます。 この 2 つのオプションには関連性がありません。

最初のオプションを選択することにより、以下のサブキーにレジストリ値が設定されます。

```
HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥UCTNG
```

値 Integration を 1（yes）に設定すると、値 EvtManagerServer が Unicenter TNG ホストの名前を文字列で受け取ります。

Unicenter TNG に渡される監査イベントは、［Unicenter エンタープライズ管理］-［エンタープライズ マネージャ］-［Windows NT］-［イベント］ウィンドウのコンソール ログに表示されます。

監査イベント	表示色	重大度
成功	青	S
拒否	オレンジ色	F

監査イベント	表示色	重大度
失敗	オレンジ色	F
警告	青	W
CA Access Control の停止（監査終了）	青	I
CA Access Control の開始（監査開始）	青	I

2 番目のオプションを選択すると、Unicenter の［Worldview］メニューから CA Access Control を起動できます。CA Access Control を起動するには、［管理対象オブジェクト］ウィンドウで、TCP/IP ネットワークを表すアイコンをポイントして右クリックし、表示されたメニューから［CA Access Control］を選択します。

また、イベントに関する以下の情報も送信されます。

- 製品名（CA Access Control + バージョン番号）
- ユーザ名
- 端末名
- クラス名
- リソース名
- プロセス名
- イベントの時刻
- CA Access Control 監査形式の完全な監査メッセージ

［ユーザ名］、［端末名］、［クラス名］、［リソース名］、および［プロセス名］の各フィールドは、イベント タイプによっては送信されないこともあります。

コンポーネント

CA Access Control には、データベース（seosdb）、2 つのドライバ（seosdrv および drveng）、多数のサービス（Watchdog、Agent、Engine（seosd）、Policy Model、タスクの委任など）、およびグラフィカル ユーザ インターフェースが含まれます。

データベース

データベースには、以下の要素の定義が格納されます。

- 組織内のユーザおよびグループ
- 保護が必要なシステム リソース
- ユーザおよびグループによるシステム リソースへのアクセスを管理するルール

ドライバ

ドライバは、以下のタスクを実行することによって、CA Access Control のファイルとレジストリ キーをすべて保護します。

- ファイルを開く要求、レジストリ キーにアクセスする要求、プロセスを終了する要求、およびネットワーク アクティビティを実行する要求をインターセプトする
- これらの要求を CA Access Control Engine に渡し、Engine から要求の許可または拒否の決定を受け取る
- この決定をオペレーティング システムの元のシステム コールに転送する（オペレーティング システムは、ドライバから受け取った応答に基づいて処理を継続する）

サービス

Watchdog

Watchdog は、他の CA Access Control サービスが実行されていることを常時チェックします。 Watchdog は、他のサービスが停止していることを検出すると（ただし、停止することはほとんどありません）、ただちにそのサービスを再開します。

Agent

エージェントは以下のタスクを実行します。

- TCP/IP 上の専用アプリケーション プロトコルを介して CA Access Control クライアントと通信する
- CA Access Control ユーザのセキュリティを管理する

Engine

エンジンは以下のタスクを実行します。

- すべてのデータベース更新の管理を含むデータベースの管理を行う
- ドライバおよびエージェントから受け取ったアクセス要求を許可するかどうかを決定する
- Watchdog サービスが実行中かどうかをチェックし、実行停止を検出した場合は Watchdog サービスを再開する

Engine は、データベース アクセス要求を処理し、かつアクセス許可の決定を行うことによって、効率的なサービスを作成します。

Policy Model

何百、何千ものデータベースを個別に管理することは、現実的ではありません。 そのため、CA Access Control には、1 台のコンピュータから多数のコンピュータを管理できるコンポーネントである Policy Model サービスが用意されています。 Policy Model サービスの使用は任意ですが、このサービスを使用すると、大規模なサイトでの管理を大幅に簡略化できます。

Policy Model データベース（PMDB）は、この Policy Model サービスと共に使用します。 PMDB には、他の CA Access Control データベースと同様に、ユーザ、グループ、保護されているリソース、およびリソースへのアクセスを管理するルールが保存されています。 さらに、PMDB にはサブスクライバ端末のリストが含まれています。サブスクライバ端末は PMDB にリンクされた端末であるため、PMDB への変更はサブスクライバ データベースに自動的に送信されます。

ユーザは、組織に適用する基本的なセキュリティ ポリシーを作成し、必要なすべてのルールを単一のデータベース（Policy Model データベース）に実装できます。 サブスクライバには、Windows 端末と UNIX 端末の両方を含めることができるため、最小限の管理作業で一定のルールを保証できます。

PMDB は、システム管理者またはセキュリティ管理者が更新します。PMDB によってすべての更新内容が PMDB からサブスクライバにバッチ モードで伝達されるため、管理者は他の作業を行うことができます。

PMDB のサブスクライバには、別の PMDB とローカル データベースの 2 種類があります。また、この PMDB には、データベースの更新内容の伝達先となるサブスクライバの一覧が保存されています。 この機能によって、PMDB の階層を構築できます。ローカル データベースは、端末に定義されているユーザ、グループ、およびリソースを保護するために使用できます。

selang

コマンド ライン言語の selang を使用すると、CA Access Control のすべての機能を実行できます。 selang のコマンドを使用するには、コマンド プロンプト ウィンドウを開き、selang を起動します。selang はスクリプトでも使用できます。

selang とそのコマンドの詳細については、「*selang リファレンス ガイド*」を参照してください。

エンドポイント管理

CA Access Control では、2 つの方法で、企業内のリソースを管理し、リソースにアクセスするユーザを制御することができます。

- **selang** - CA Access Control コマンド言語。

  selang コマンド言語を使用すると、CA Access Control データベースに定義を作成することができます。 selang コマンド言語は、コマンド定義言語です。

  注: selang の使用法の詳細については、「*selang リファレンス ガイド*」を参照してください。

- **CA Access Control エンドポイント管理** - エンドポイント管理インタフェース。

  この Web ベースのインタフェースでは、中央の管理サーバからリモートのエンドポイントを管理することができます。

  注: CA Access Control エンドポイント管理のインストールの詳細については、「*実装ガイド*」を参照してください。

第 3 章: ユーザおよびグループの管理

このセクションには、以下のトピックが含まれています。

ユーザおよびグループ

CA Access Control では、アクションまたはアクセスのすべての試みが、要求を送信するユーザに代わって、実行されます。 したがって、システムのすべてのプロセスは、特定のユーザ名に関連付けられます。 CA Access Control のユーザは、ユーザ名によって識別されます。

ユーザとは、ログインできる人、またはバッチおよびデーモン プログラムの所有者すべてを指します。 CA Access Control では、アクセス試行のすべてがユーザによって実行されます。 CA Access Control は、CA Access Control データベースのユーザ情報とエンタープライズ ユーザ ストアのユーザ情報を使用できます。 ユーザ情報は、データベースの USER レコードまたは XUSER レコードのいずれかに格納されます。

注: *エンタープライズ ユーザ ストア*とは、ユーザやグループが格納されているオペレーティング システム内のストア（たとえば、UNIX システムの /etc/passwd や /etc/groups、Windows の Active Directory など）です。

*グループ*は、ユーザの集合です。 グループでは、グループ内のすべてのユーザに適用する共通のアクセスルールを定義します。 グループはネストする（他のグループに属する）こともできます。 CA Access Control は、CA Access Control データベースのグループ情報とエンタープライズ ユーザ ストアのグループ情報を使用できます。 通常は、ロール（database_administrators など）に基づいて、グループを作成し、そのグループにユーザを割り当てます。

ユーザ レコードは、重要なアクセサ レコードです。CA Access Control でグループを使用する主な目的は、一度にグループ内のすべてのユーザにアクセス権限を割り当てることです。 アクセス権限を個々のユーザに別々に割り当てるよりも一度に割り当てるほうが簡単で、エラーが発生する可能性も低くなります。

アクセサに関する情報の格納場所

CA Access Control が使用するユーザ情報とグループ情報は、CA Access Control データベースとホスト オペレーティング システムの両方に格納されます。 ホスト オペレーティング システム内の情報は、エンタープライズ ユーザ ストア、または単にエンタープライズ ストアと呼ばれます。 デフォルトでは、CA Access Control は、エンタープライズ ストアを使用しないように設定されています。 ただし、CA Access Control データベースに定義されているユーザまたはグループが見つからない場合は、エンタープライズ ストアに定義されているユーザおよびグループ メンバシップを検索して、その情報を使用するように、CA Access Control を設定することもできます。

注: CA Access Control は、エンタープライズ ストアの情報を使用しますが、エンタープライズ ストアに書き込みを行うのはネイティブ環境で selang コマンドが使用された場合のみです。

権限をチェックする際、CA Access Control は必ず自身のデータベースに定義されているアクセサをチェックしてから、エンタープライズ ストアを調べます。CA Access Control データベースに定義されているユーザと同じ名前のエンタープライズ ユーザがいる場合、CA Access Control はそのエンタープライズ ユーザを無視します。

CA Access Control によるユーザ レコードの検索方法

ユーザがログインすると、CA Access Control は、そのユーザに関連付けられたレコードを見つけるまで、以下の順序で検索を実施します。

1. CA Access Control は、自身のデータベースに定義されているユーザを検索します。
2. CA Access Control は、自身のキャッシュで、そのエンタープライズ ユーザを検索します。

   ネットワークが停止した場合は、オペレーティング システム（OS）により、ユーザは OS 内にキャッシュされた認証情報を使用してログインできます。 CA Access Control キャッシュの目的は、このような場合に CA Access Control がエンタープライズ ユーザのレコードも使用できるようにすることです。
3. CA Access Control は、オペレーティング システムを使用して、エンタープライズ ユーザ ストアで、そのエンタープライズ ユーザを検索します。
4. CA Access Control がデータベース内またはエンタープライズ ストア内でユーザに関連付けられたレコードを見つけられない場合、CA Access Control はユーザに _undefined USER レコード内の属性を割り当てます。

エンタープライズ ユーザ ストアとの統合

通常は、エンタープライズ ユーザ ストアに定義されているグループとユーザを使用するように CA Access Control を設定します。

デフォルトで、このように CA Access Control を設定しておけば、エンタープライズ ユーザまたはエンタープライズ グループを参照するアクセス ルールが作成されたときや、ユーザがオペレーティング システムにログインしたときに、ユーザまたはグループのレコードが事前に存在していなかった場合に、CA Access Control は自身のデータベースにそのユーザまたはグループのレコードを作成します。 これらのレコードには、XUSER クラス（エンタープライズ ユーザの場合）または XGROUP クラス（エンタープライズ グループの場合）が割り当てられます。 これらのクラスは、CA Access Control がアクセス ルールを適用する場合に必要とするプロパティを保持しています。 CA Access Control が必要に応じて作成するため、手動で管理する必要はありません。

CA Access Control がエンタープライズ ユーザ ストアから取得するエンタープライズ ユーザまたはエンタープライズ グループのプロパティは、名前と、グループ メンバシップ のプロパティのみです。

エンタープライズ ストアでアクセサを管理するためのガイドライン

エンタープライズ ユーザ ストアでアクセサを管理する場合は、以下のセクションに記載されているガイドラインを確認してください。

データベースに定義する必要があるユーザおよびグループ。

CA Access Control では、一部のユーザおよびグループを、エンタープライズ ユーザ ストアではなく、自身のデータベースに定義する必要があります。 これらのユーザおよびグループは、以下のとおりです。

- 事前定義済みユーザ (P. 47)
- 事前定義済みグループ (P. 48)
- CA Access Control 管理者
- プロファイル グループ
- 論理ユーザ

エンタープライズ ユーザの使用制限

CA Access Control は、エンタープライズ ユーザの使用に以下の制限を適用します。

- エンタープライズ ユーザの名前が、データベースに定義されるユーザと同じ場合、CA Access Control でそのエンタープライズ ユーザを作成、または参照することはできません。
- selang AC 環境を使用して、エンタープライズ ユーザを作成、削除、または変更することはできません。

- エンタープライズ ユーザを論理ユーザとして使用することはできません。
- デフォルトでは、ユーザがエンタープライズ ユーザ ストアに事前に定義されていない限り、CA Access Control でエンタープライズ ユーザを作成することはできません。ただし、UNIX システム上でこの動作を有効または無効にすることができます。

詳細情報:

UNIX 上で XUSER レコードを作成する前のエンタープライズ ストア チェックの有効化/無効化 (P. 43)

エンタープライズ グループの使用制限

CA Access Control は、エンタープライズ グループの使用に以下の制限を適用します。

- selang AC 環境内で、エンタープライズ グループを作成または削除することはできません。
- selang AC 環境内で、エンタープライズ グループのメンバシップを変更することはできません。
- エンタープライズ グループをプロファイル グループ (P. 50)として使用することはできません。

エンタープライズ ユーザおよびグループの有効化/無効化

CA Access Control はデフォルトではエンタープライズ ユーザ ストアに定義されているグループおよびユーザを使用できませんが、それができるように CA Access Control を有効化することができます。 CA Access Control の以前のバージョンとの互換性が必要な場合を除き、この機能を有効にしておくことをお勧めします。

CA Access Control がエンタープライズ ユーザとグループを使用できるようにするには、構成設定 osuser_enable を「yes」に設定します。 この動作を無効にするには、osuser_enabled の値を 「no」に設定します。

例: Windows 上でエンタープライズ ユーザとグループを有効にする

Windows 上でエンタープライズ ユーザとグループの使用を有効にするには、以下のレジストリ設定を指定します。

- キー： HKLM¥SOFTWARE¥ComputerAssociates¥AccessControl¥OS_user
- 名前： osuser_enabled
- タイプ： REG_DWORD
- 値： yes

例: UNIX 上でエンタープライズ ユーザとグループを有効にする

以下のコマンドを実行して、CA Access Control を停止してから、UNIX 上でエンタープライズ ユーザとグループの使用を有効にし、CA Access Control を再起動します。

```
secons -s
seini -s OS_User.osuser_enabled yes
seload
```

エンタープライズ ユーザのログイン時の XUSER レコードの作成の有効化/無効化

CA Access Control で、エンタープライズ ユーザの使用が有効になっている場合、デフォルトでは、ユーザがログインしたときにそのユーザのレコードが（XUSER クラスに）作成されます。 ただし、毎日同じ時刻に数千人のユーザがログインする場合など、このレコードを作成したくないこともあります。

ユーザがログインしたときに CA Access Control が XUSER レコードを作成しないようにするには、設定 create_user_in_db の値を 0（ゼロ）に変更します。 この動作を再び有効にするには、この値を 1 に設定します。

例: エンタープライズ ユーザが Windows にログインしたときの XUSER レコードの自動作成を無効にする

Windows 上で CA Access Control でのエンタープライズ ユーザ レコードの自動作成を無効にするには、以下のレジストリ設定を指定します。

- キー： HKLM¥Software¥ComputerAssociates¥AccessControl¥OS_user
- 名前： create_user_in_db
- タイプ： REG_DWORD
- 値： 0

例: エンタープライズ ユーザが UNIX にログインしたときの XUSER レコードの自動作成を無効にする

以下のコマンドを実行して、CA Access Control を停止してから、UNIX 上で XUSER レコードの自動作成を無効にし、CA Access Control を再起動します。

```
secons -s
seini -s OS_User.create_user_in_db 0
seload
```

UNIX 上で XUSER レコードを作成する前のエンタープライズ ストア チェックの有効化/無効化

ユーザがエンタープライズ ユーザ ストアに定義されていない場合は、CA Access Control でエンタープライズ ユーザを作成することができます。Windows では、ユーザが Windows のユーザ ストアに存在しない限り、CA Access Control でエンタープライズ ユーザを作成することはできません。UNIX のデフォルト動作は Windows とは逆です。 ただし、UNIX では、このデフォルト動作を有効または無効にすることができます。

チェックを無効にする（したがって、同等のエンタープライズ ユーザが存在しない場合に CA Access Control が XUSER レコードを作成できるようにする）には、verify_osuser の設定値を 0 に変更します。 チェックを適用するには、この値を 1 に設定します。

例: エンタープライズ ユーザ ストアをチェックせずに XUSER レコードの作成を有効にする

以下のコマンド セットを実行すると、CA Access Control は停止し、エンタープライズ ストアに同等のレコードがない XUSER レコードの作成が有効になり、CA Access Control の再起動が実行されます。

```
secons -s
seini -s OS_User.verify_osuser 0
seload
```

Windows での再利用エンタープライズ ストア アカウント

*再利用アカウント*とは、削除された後で（同じ名前を使用して）再作成されたエンタープライズ ストアのユーザまたはグループです。 これは、たとえば、ユーザ ストアからユーザを削除した後で（ユーザが退職した場合など）、その削除されたユーザと同じ名前の新規ユーザの新規アカウントを作成するときに発生します。

再利用アカウントは、セキュリティ ホールです。名前が同一の以前のアカウントに付与されていたアクセス許可と同じアクセス許可が新規のアクセサに必ずしも必要とは限らないためです。 この問題を解決するためには、CA Access Control の許可は SID に基づいています。 つまり、アクセス許可が付与されていた削除済みのアクセサと名前が同一の新規アクセサを作成しても、その新規アクセサに対しては、以前のアクセサに付与されていた古い許可は自動的には付与されません。

重要: 再利用アカウント アクセサは、古いアクセス許可を継承 *しません*。ただし、（SID ではなく）アクセサの名前を指定するデータベース アクセス ルールでは、これらのルールが適用されると思われることがあります。この問題は、secons -checkSID コマンドを使用して解決します。

Windows での再利用エンタープライズ アカウントの解決

関連付けられたデータベース ルールを持つエンタープライズ アカウント（ユーザまたはグループ）が再利用（つまり、削除されてから、同じ名前で作成）された場合、古いデータベース ルールが新規アカウントにも適用されると思うユーザもいるでしょう。 しかし、CA Access Control の許可は SID に基づいているので、これらのルールは適用されません。新規ユーザ/グループ用の新規ルールを作成する必要があります。 新規ルールを作成するには、事前に再利用アカウントを解決しておく必要があります。

再利用エンタープライズ アカウントを解決するには、コマンド プロンプトを開き、以下のコマンドを実行します。

```
secons -checkSID -users
secons -checkSID -groups
```

CA Access Control は、所有しているすべてのエンタープライズ ユーザ アカウント（XUSER レコード）を確認してから、すべてのグループ アカウント（XGROUP レコード）を確認し、エンタープライズ アカウントの SID とは異なる SID を持つアカウントを識別します。 CA Access Control で、命名規則 *SID* (*accountName*) に従って、これらのアカウントの名前を変更します。

これで、再利用アカウントの新規ルールを作成できます。

注: 再利用ユーザ アカウントは、ユーザがログインしたり、リソースにアクセスしようとしたときに、このように解決されます。 エンタープライズ アカウントを作成する場合は、secons -checkSID コマンドを、スケジュール タスクとして実行することをお勧めします。

例: 再利用グループ アカウント

ABCD 社のエンタープライズ ストアに、*interns* というグループがあります。このグループには、9 人のメンバが所属し、productA に取り組んでいます。管理者は、以下のように、このグループを CA Access Control に認識させ、グループのメンバがアクセスする必要があるファイルへのアクセス許可をこのグループに割り当てます。

```
nxg interns owner(msmith)
auth file c:¥products¥productA¥materials¥* xgid(interns) access(all)
auth file c:¥HR¥interns¥* xgid(interns) access(read)
```

interns が ABCD 社での就労期間を完了すると、エンタープライズ ストア管理者はこのグループを削除します。 3 か月後、6 人のメンバから成る新しい interns グループがエンタープライズ ストアに同じ名前で作成されます。CA Access Control データベース内の古いルールはまだ存在するので、新しい *interns* グループはこの古いルールの許可を継承するように思われます。しかし、これらのルールは古い interns グループに適用されるものなので、CA Access Control 管理者は新規グループ用の新規ルールを作成する必要があります。

このためには、管理者は、以下のように interns 再利用アカウントを識別して解決する必要があります。

```
secons -checkSID -groups interns
```

これにより、XGROUP リソースの名前と、このリソースへのアクセス ルール参照の名前が、「*SID* (*domain*¥interns)」に変更されます。 これで、管理者は、productB に取り組む新規 interns グループ用の新規ルールを作成できます。

```
nxg interns owner(msmith)
auth file c:¥products¥productB¥materials¥* xgid(interns) access(all)
auth file c:¥HR¥interns¥* xgid(interns) access(read)
```

注: secons ユーティリティの詳細については、「リファレンス ガイド」を参照してください。

データベース アクセサ

ユーザをどのように管理するかに関係なく、以下に説明するように、CA Access Control データベースに定義する必要があるアクセサがあります。

事前定義済みユーザ

CA Access Control は、以下のユーザを事前定義します。これらのユーザを削除することはできません。

+devcalc

（Windows）CA Access Control が偏差計算プロセス devcalc を実行するときのユーザ名。

_dms

拡張ポリシー管理サーバ コンポーネントのデータベース（DMS、DH リーダ、および DH ライタ）にインストールされている _dms ユーザは、policyfetcher および devcalc が DH および DMS と通信する場合に使用されます。

nobody

nobody ユーザは、実際のユーザに対応させることのできないユーザ レコードです。このレコードは、関連する許可をどのユーザにも付与しないルールを作成する場合に使用します。たとえば、*nobody* をリソースの所有者として設定し、どのユーザも、そのレコードの所有に関連する許可を取得しないようにすることができます。

+reportagent

CA Access Control がレポート エージェントを実行するときのユーザ名。

_seagent

_seagent は、CA Access Control が以下のような内部プロセスを実行するときのユーザ名です。

- PMDB プロセス、sepmdd
- （UNIX）偏差計算プロセス、devcalc
- ユーザおよびグループ レコード更新 の exit プロセス

_seagent ユーザには SERVER 属性が割り当てられています。

_sebuildla

（UNIX） _sebuildla ユーザは、CA Access Control デーモン seosd に対して lookaside データベースを作成するために CA Access Control が sebuildla ユーティリティを実行する際に使用するユーザ名です。

_seoswd

（UNIX） _seoswd は、データベースに trusted プログラムとして定義されているプログラムのファイル情報およびデジタル署名を監視する、seoswd Watchdog デーモンを実行するために使用されるユーザ名です。

_undefined

_undefined は、CA Access Control で定義されていないすべてのユーザを表します。 _undefined を使用して、未定義のユーザを ACL に含めることができます。

事前定義済みグループ

CA Access Control には、事前定義済みグループが用意されています。_interactive グループと _network グループを除き、これらの事前定義済みグループには、他のグループと同じようにユーザを追加できます。

_abspath

ログイン時に _abspath グループに属しているユーザは、プログラムを起動する場合に絶対パス名を使用する必要があります。

_interactive

ユーザは、アクセスの目的でのみ、_interactive グループのメンバになります。 ユーザは、アクセスしようとしているリソースと同じホストにログインしている場合、_interactive グループのメンバになります。 CA Access Control は、_interactive グループのメンバシップを動的かつ自動的に管理します。このメンバシップを変更することはできません。

_network

これは、_interactive の補完グループです。 ユーザは、アクセスの目的でのみ、_network グループのメンバになります。 ユーザは、リソースが属するホストとは別のホストにアクセスしようとする場合、_network グループのメンバになります。CA Access Control は、_network グループのメンバシップを動的かつ自動的に管理します。 このメンバシップを変更することはできません。

_restricted

_restricted グループのユーザに対しては、ファイルはすべて（Windows の場合はレジストリ キーも）CA Access Control によって保護されます。 ファイルまたは Windows のレジストリ キーで、アクセス ルールが明示的に定義されていない場合、アクセス許可は、そのクラス（FILE または REGKEY）の _default レコードが適用されます。

注: _restricted グループに属するユーザには、処理を実行するための十分な権限が付与されない可能性があります。 このため、ユーザを _restricted グループに追加する場合は、最初に警告モードの使用を検討してください。

_surrogate

ユーザが _surrogate グループのメンバを代理として使用する場合、CA Access Control は、その代理のアクションの監査証跡として元のユーザの名前が付けられた完全なトレースを書き込みます。

例: selang を使用して _restricted グループにユーザを追加する

以下の selang コマンドは、エンタープライズ ユーザ john_smith を _restricted グループに追加します。

```
joinx john_smith group(_restricted)
```

プロファイル グループ

プロファイル グループは、ユーザ プロパティのデフォルト値が収められている、CA Access Control データベースに定義されるグループです。 ユーザをプロファイル グループに割り当てた場合、そのユーザにすでに値が設定されていない限り、プロファイル グループはそのデフォルト値をユーザに提供します。

ユーザのプロファイル グループは、ユーザの作成時に指定できます。または、後でプロファイル グループにユーザを割り当てることもできます。

プロファイル グループを使用すると、管理者は、グループに割り当てる新規ユーザに対して、特定の権限が指定された標準設定を効率よく作成できます。 このセットアップでは、ユーザのホーム ディレクトリ、監査プロパティ、アクセス権限を定義する PMDB、およびプロファイル グループに関連付けられているユーザに影響を与えるさまざまなパスワード ルールなどを指定することができます。

CA Access Control がプロファイル グループを使用してユーザ プロパティを決定する方法

以下のプロセスでは、CA Access Control がプロファイル グループを使用してユーザ プロパティを指定する方法について説明します。

1. CA Access Control は、USER クラスまたは XUSER クラスのユーザのレコードにプロパティの値があるかどうかをチェックします。

   ユーザのレコードがプロパティの値を持っている場合は、CA Access Control はその値を使用します。

2. CA Access Control は、ユーザがプロファイル グループに割り当てられているかどうかをチェックします。

   ユーザがプロファイル グループに割り当てられている場合は、プロセスは続行します。ユーザがプロファイル グループに割り当てられていない場合、CA Access Control はデフォルトのプロパティ値をユーザに割り当てます。

3. CA Access Control は、プロファイル グループがそのプロパティの値を持っているかどうかをチェックします。

   プロファイル グループがプロパティの値を持っている場合、CA Access Control はその値をユーザに割り当てます。 プロファイル グループがプロパティの値を持っていない場合、CA Access Control はデフォルトのプロパティ値をユーザに割り当てます。

   注: ユーザまたはプロファイル グループの監査プロパティが設定されていない場合、グループの監査プロパティはユーザの監査プロパティに影響を与える場合があります。

詳細情報:

CA Access Control がユーザの監査モードを決定する方法 (P. 133)

アクセサ管理

CA Access Control エンドポイント管理 または selang を使用して、データベースまたはエンタープライズ ストアのユーザまたはグループのレコードを作成、変更、および削除することができます。

ユーザまたはグループの管理

特定のアクセサのプロパティを表示または変更する場合や、アクセサを削除する場合は、まずそのアクセサを見つける必要があります。

ユーザまたはグループの管理方法

1. CA Access Control エンドポイント管理 内で、以下の操作を実行します。
   a. ［ユーザ］をクリックします。
   b. ［ユーザ］*または*［グループ］サブタブのいずれかをクリックします。

   選択したサブタブに応じて、［ユーザ］ページまたは［グループ］ページが表示されます。
2. ［検索］セクションの以下のフィールドに入力します。

   ユーザ名/グループ名

   表示したいアクセサのマスクを定義します。 対象とするアクセサのフル ネームを入力するか、マスクを使用することができます。 たとえば、名前に「admin」を含むアクセサをリストするには、*admin* を使用します。

   すべてのアクセサをリストするには、アスタリスク（*）を、 1 文字を置換するには、疑問符（?）を使用します。

   ユーザ リポジトリ/グループ リポジトリ

   アクセサ リストの取得元のソースを指定します。 ソースとして、以下のいずれかを指定できます。

   - **内部アカウント** - CA Access Control データベースに定義されているアクセサ。
   - **エンタープライズ アカウント** - 特定のエンタープライズ ユーザ ストアに定義されているアクセサ。

AC アカウント/プロファイルのみを表示

以下のように、CA Access Control データベース内にレコードがあるアカウントのみをリストするかどうかを指定します。

- ［内部アカウント］を選択した場合は、CA Access Control データベース内に存在するアカウントのみをリストします（ネイティブ アカウントは含まれません）。
- ［エンタープライズ アカウント］を選択した場合は、CA Access Control エンタープライズ プロファイル（XUSER レコードまたは XGROUP レコード）を持つアカウントのみをリストします。

［Go］をクリックします。

選択したリポジトリに存在するアクセサのリストが表示されます。

3. 以下の*いずれか*の操作を行います。
   - ［表示］列で［🔍］をクリックして、アクセサのプロパティを表示します。
   - ［削除］列で［🗑］をクリックして、アクセサを削除します。
   - アクセサの名前をクリックして、アクセサのプロパティを変更します。
   - 削除するアクセサを選択し、［削除］をクリックします。
   - ［ユーザの作成］または［グループの作成］をクリックし、CA Access Control データベースに新規のユーザ レコードまたはグループ レコードを作成します。

例: リポジトリ内でのエンタープライズ ユーザの検索

以下の図は、ABC-DM1 エンタープライズ ユーザ ストアでの全ユーザの検索結果を示しています。

selang を使用したユーザ管理

エンタープライズ ユーザのレコードには、以下の selang コマンドを使用します。

- **newxusr** および **editxusr** - 新規のエンタープライズ ユーザ レコードを定義します。
- **chxusr** および **editxusr** - エンタープライズ ユーザの CA Access Control プロパティを変更します。
- **find xuser** - CA Access Control レコードを持つエンタープライズ ユーザをリストします。
- **rmxusr** - ユーザを削除します。
- **show xuser** - エンタープライズ ユーザの CA Access Control プロパティを表示します。

CA Access Control データベース ユーザ レコードには、以下の selang コマンドを使用します。

- **newusr** および **editusr** - 新規のユーザ レコードを定義します。
- **chusr** および **editusr** - ユーザのプロパティを変更します。
- **rmusr** - ユーザを削除します。
- **find user** - データベース ユーザをリストします。
- **show user** - ユーザのプロパティを表示します。

例: selang を使用してデータベースにユーザを定義する

以下の selang コマンドは、CA Access Control データベースに、セキュリティ レベルを 100 とする新規ユーザを定義します。

```
newusr internalUser level(100)
```

例: selang を使用してエンタープライズ ユーザのプロパティを変更する

以下の selang コマンドは、エンタープライズ ユーザ Terry に AUDITOR 属性を割り当てます。

```
chxusr Terry auditor
```

selang を使用したグループ管理

エンタープライズ グループの名前およびメンバシップを除く、任意のグループのすべてのプロパティを変更できます（名前とメンバシップの変更は、CA Access Control 内からは変更できません）。

グループ プロパティを変更したり、グループに関連付けるアクセス権を割り当てるには、CA Access Control エンドポイント管理 または以下の selang コマンドを使用できます。

- **join[-]** および **joinx[-]**

  内部グループのメンバシップを変更します。

  内部アクセサをグループに追加するには、join を使用します。エンタープライズ グループおよびユーザを内部グループに追加するには、joinx を使用します。 アクセサを内部グループから外すには、コマンドにマイナス（-）記号を付けます。

- **editgrp**、**newgrp**、**chgrp**

  内部グループのメンバシップ以外のプロパティを変更します。

- **editxgrp**、**newxgrp**、**chxgrp**

  エンタープライズ グループのメンバシップ以外のプロパティを変更します。

- **rmgrp**、**rmxgrp**

  内部グループ、 エンタープライズ グループを削除します。

例: selang を使用してデータベースにグループを定義する

以下の selang コマンドは、データベースに新規グループ「sales」を定義します。 グループのフル ネームは「Sales Department」です。

```
newgrp sales name('Sales Department')
```

例: selang を使用して、データベースに定義されているグループのプロパティを変更する

以下の selang コマンドによって、CA Access Control は、グループ AC_admins のメンバに対するすべてのイベントを監査します。

```
chgrp AC_admins audit(all)
```

例: selang を使用して、ACL にエンタープライズ グループを追加する

以下の selang コマンドは、myfile という ACL にエンタープライズ グループ mygroup を追加します。

```
Authorize FILE (myfile) xgid(mygroup)
```

例: selang を使用して、データベースに定義されているグループにエンタープライズ ユーザを追加する

以下の selang コマンドは、データベースに定義されているグループ AC_admins に、エンタープライズ ユーザ mydomain¥administrator を追加します。

```
joinx mydomain¥administrator group(AC_admins)
```

例: selang を使用して、データベースに定義されているグループにエンタープライズ グループを追加する

以下の selang コマンドは、_restricted グループにエンタープライズ グループ Guests を追加します。

```
joinx Guests group(_restricted)
```

第 4 章: リソースの管理

このセクションには、以下のトピックが含まれています。

リソース

リソースとは、アクセサがアクセスでき、アクセス ルールによって保護されるエンティティ、またはそのエンティティに対応する CA Access Control データベース レコードです。 リソースの例には、ファイル、プログラム、ホスト、端末などがあります。

CA Access Control でリソース レコードを作成する主な目的は、リソース レコードに対応するリソースのアクセス許可を定義することです。 リソースへのアクセスに必要なアクセス許可は、リソース レコードのアクセス制御リストに指定します。

リソース グループ

リソース グループは、その他のリソースから成るリストを含むリソースです。 リソース グループとは、CONTAINER、GFILE、GSUDO、GTERMINAL、または GHOST のいずれかのクラスのメンバです。

リソース グループはそれ自体がリソースであるため、そのメンバリソースに同じプロパティが割り当てられます。 したがって、リソース グループを使用するメリットは、管理の簡略化です。 リソース グループのプロパティを変更することで、すべてのメンバ リソースのプロパティを変更できます。

注: Windows では、リソースに対するユーザ認証をチェックする際に、CA Access Control により、リソース グループの所有者権限が考慮されます。これは、r12.0 で導入されました。 以前のリリースでは、認証プロセスではリソースの所有者のみが考慮されていました。

たとえば、none および no owner のデフォルト アクセスを備えた FILE リソースを定義します。 FILE リソースは指定された所有者を備えた GFILE リソースのメンバです。CA Access Control r12.0 以降では、指定されたグループ所有者にそのファイルの完全なアクセス権が与えられます。 以前のリリースでは、誰にもそのファイルのアクセス権が与えられていませんでした。

クラス

CA Access Control では、レコードに割り当てることのできるプロパティはレコードのクラスによって定義されます。 1 つのクラス内のすべてのレコードに、同じプロパティが割り当てられます。ただし、これらのプロパティの値は異なります。

クラスの例は、以下のとおりです。

- TERMINAL クラス。 tty1、tty などの端末のレコードが含まれます。
- FILE クラス。 ファイルのレコードが含まれます。
- PROGRAM クラス。 プログラムのレコードが含まれます。

各レコードには、レコード クラスに適したプロパティの値が保存されます。 たとえば、XUSER クラスのレコードにはエンタープライズ ユーザの勤務地や勤務時間などのプロパティが保存され、HOSTNET クラスのレコードにはネット サービスや IP アドレス データなどのプロパティが保存されます。

CA Access Control には、事前定義されたクラスが含まれています。 また、ユーザ定義クラスと呼ばれる新規クラスを定義することもできます。

クラスのデフォルト レコード

ほとんどのクラスには、デフォルト レコード（_default）を含めることができます。このレコードは、クラスのリソースのうち、データベースに対応するレコードが定義されていないリソースのアクセス タイプを指定します。

他のリソース レコードと同様に、_default レコードには、ACL および defaccess フィールドを含めることができます。_default レコードは、USER、GROUP、CATEGORY、SECLABEL、および SEOS を除くすべてのクラスに作成できます。

UACC クラス（廃止予定）

UACC クラスの使用はお勧めしません。 クラス内のレコードに対するデフォルト値を指定するには、_default レコードを使用してください。

CA Access Control の一部の旧バージョンでは、他のクラスの _default レコードに似たレコードに対して、UACC という別のクラスを使用していました。 UACC クラスの使用はお勧めしません。_default レコードを使用する場合、UACC クラスの対応するレコードはチェックされません。 今後のバージョンでは、UACC クラスはサポートされなくなる可能性があります。

たとえば、ユーザ Henderson がプロセス store_log の強制終了（kill）を試みたとします。 この場合、CA Access Control では、以下の順序で権限がチェックされます。 まず最初に、プロセス store_log がデータベースに定義されているかどうかがチェックされます。 CA Access Control は、データベースで PROCESS クラスの store_log というレコードを検索します。

- 該当するレコードが見つからない場合、このプロセスは CA Access Control に定義されていません。 この場合、CA Access Control は、PROCESS クラスの _default レコード、または UACC クラスの PROCESS レコードのいずれかを使用して、Henderson が store_log を強制終了（kill）できるかどうかを判断します。
  - ユーザ Henderson が _default レコードの ACL に定義されている場合は、ACL に指定された権限が適用されます。
  - Henderson が _default レコードの ACL に定義されていない場合は、_default レコードの defaccess プロパティに指定された権限が適用されます。 この権限は、_default の ACL に明示的に指定されていないすべてのユーザに適用されます。

- プロセス store_log がデータベースに定義されている場合は、ユーザ Henderson がデータベースでプロセス store_log の ACL に定義されているかどうかが問題になります。
  - ユーザ Henderson がプロセス store_log の ACL に定義されている場合は、ACL に指定された権限が適用されます。
  - ユーザ Henderson が ACL に定義されていない場合は、store_log リソースのデフォルト アクセス プロパティに指定された権限が適用されます。 この権限は、リソースのデフォルト アクセスといいます。

注: _default のデフォルト アクセス（defaccess）が NONE に設定されている場合、または、_default が未指定で UACC クラスの対応するリソースのデフォルトが NONE である場合は、クラスに定義されていないリソースにアクセスを試みたアクセサは、リソースへのアクセスを拒否されます。

_default（または UACC）のデフォルト アクセス権として最上位の権限（ALL、または場合によっては READ か EXECUTE）が設定されている場合、明示的に保護されていないリソースには、すべてのユーザがアクセスできます。

事前定義されたクラス

事前定義されたクラスは、以下のタイプに分類できます。

クラス タイプ	目的
アクセサ	ユーザ、グループなど、リソースにアクセスするオブジェクトを定義します。
定義	セキュリティ ラベルやセキュリティ カテゴリなど、セキュリティ エンティティを定義するオブジェクトを定義します。
インストール	CA Access Control の動作を制御するオブジェクトを定義します。
リソース	アクセス ルールによって保護されるオブジェクトを定義します。

以下の表は、事前定義クラスの一覧です。

クラス	クラス タイプ	説明
ADMIN	定義	ADMIN 属性を持たないユーザに管理責任を委任します。 これらのユーザにグローバル権限属性を付与し、管理者権限の適用範囲を制限します。

クラス	クラス タイプ	説明
AGENT	リソース	CA Access Control には適用されません。
AGENT_TYPE	リソース	CA Access Control には適用されません。
APPL	リソース	CA Access Control には適用されません。
AUTHHOST	アクセサ	CA Access Control には適用されません。
CALENDAR	リソース	時間制限が適用されるユーザ、グループ、およびリソースの Unicenter TNG カレンダ オブジェクトを定義します。
カテゴリ	定義	セキュリティ カテゴリを定義します。
CONNECT	リソース	外部接続を保護します。 このクラスのレコードは、どのユーザがどのインターネット ホストにアクセスできるかを定義します。 CONNECT クラスをアクティブにする前に、streams モジュールがアクティブであることを確認します。
CONTAINER	リソース	他のリソース クラスにあるオブジェクトのグループを定義します。これにより、複数の異なるオブジェクトのクラスに 1 つのルールを適用する際のアクセス ルールの定義が簡略化されます。
FILE	リソース	ファイル、ディレクトリ、またはファイル名マスクを保護します。
GAPPL	リソース	CA Access Control には適用されません。
GAUTHHOST	定義	CA Access Control には適用されません。
GFILE	リソース	このクラスの各レコードは、ファイルまたはディレクトリのグループを定義します。 グループを定義するには、ユーザをグループに追加する場合と同じ方法で、ファイルまたはディレクトリ（FILE クラスのリソース）を GFILE リソースに明示的に追加します。
GHOST	リソース	このクラスの各レコードは、ホストのグループを定義します。グループを定義するには、ユーザをグループに追加する場合と同じ方法で、ホスト（HOST クラスのリソース）を GHOST リソースに明示的に追加します。
GROUP	アクセサ	このクラスの各レコードは、内部グループを定義します。

クラス	クラス タイプ	説明
GSUDO	リソース	このクラスの各レコードは、あるユーザが実行しても、別のユーザが実行しているかのように見せかけることができるコマンドのグループを定義します。 sesudo コマンドはこのクラスを使用します。
GTERMINAL	リソース	このクラスの各レコードは、端末のグループを定義します。
HNODE	定義	HNODE クラスには、組織の CA Access Control ホストに関する情報が含まれます。 クラスの各レコードは、組織内のノードを表します。
HOLIDAY	定義	このクラスの各レコードは、ユーザのログインに特別な許可を必要とする期間を 1 つ以上定義します。
HOST	リソース	このクラスの各レコードは、ホストを定義します。 ホストは、ホスト名または IP アドレスによって識別されます。 オブジェクトには、ローカル ホストがこのホストからサービスを受信できるかどうかを決定するアクセス ルールが保存されます。 HOST クラスをアクティブにする前に、streams モジュールがアクティブであることを確認します。
HOSTNET	リソース	このクラスの各レコードは、IP アドレス マスクによって識別され、アクセス ルールを格納します。
HOSTNP	リソース	このクラスの各レコードは、ホストのグループを定義します。グループに属しているホストは、すべて同じ名前パターンになります。各 HOSTNP オブジェクトの名前にはワイルドカードが含まれています。
LOGINAPPL	定義	LOGINAPPL クラスの各レコードは、ログイン アプリケーションの定義、ログイン プログラムを使用してログインできるユーザの指定、およびログイン プログラムの使用方法の制御を行います。
MFTERMINAL	定義	MFTERMINAL クラスの各レコードは、メインフレーム CA Access Control 管理コンピュータを定義します。
POLICY	リソース	POLICY クラスの各レコードは、ポリシーのデプロイおよび削除に必要な情報を定義します。これらのレコードには、ポリシーをデプロイおよび削除するための selang コマンドのリストを含む RULESET オブジェクトへのリンクが含まれます。
PROCESS	リソース	このクラスの各レコードは、実行可能ファイルを定義します。

クラス	クラス タイプ	説明
PROGRAM	リソース	このクラスの各レコードは、条件付きアクセス ルールに従って使用できる trusted プログラムを定義します。 trusted プログラムとは、改ざんされないように Watchdog 機能で監視されている setuid または setgid プログラムのことです。
PWPOLICY	定義	PWPOLICY クラスの各レコードは、パスワード ポリシーを定義します。
RESOURCE_DESC	定義	CA Access Control には適用されません。
RESPONSE_TAB	定義	CA Access Control には適用されません。
RULESET	リソース	RULESET クラスの各レコードは、ポリシーを定義するルールのセットを表します。
SECFILE	定義	このクラスの各レコードは、変更されてはならないファイルを定義します。
SECLABEL	定義	このクラスの各レコードは、セキュリティ ラベルを定義します。
SEOS	インストール	このクラスのレコードはアクティブ クラスとパスワード ルールを指定します。
SPECIALPGM	インストール	SPECIALPGM クラスの各レコードは、Windows では、バックアップ機能、DCM 機能、PBF 機能、および PBN 機能を登録し、UNIX では、xdm 機能、バックアップ 機能、メール 機能、DCM 機能、PBF 機能、および PBN 機能を登録します。または、特別な権限保護を必要とするアプリケーションを論理ユーザ ID に関連付けます。 これにより、誰が実行しているかではなく何が実行されているかに従って、アクセス許可を効率的に設定できます。
SUDO	リソース	sesudo コマンドで使用されるこのクラスは、あるユーザ（一般ユーザなど）が実行しても、別のユーザ（root ユーザなど）が実行しているかのように見せかけることができるコマンドを定義します。
SURROGATE	リソース	このクラスの各レコードには、アクセサを代理として使用できるユーザを定義する、アクセサのアクセス ルールが含まれます。

クラス	クラス タイプ	説明
TCP	リソース	このクラスの各レコードは、メール、http、ftp などの TCP/IP サービスを定義します。
TERMINAL	リソース	このクラスの各レコードは、端末（ユーザがログインに使用できるデバイス）を定義します。
UACC	リソース	各リソース クラスのデフォルト アクセス ルールを定義します。
USER	アクセサ	このクラスの各レコードは、内部ユーザを定義します。
USER_ATTR	定義	CA Access Control には適用されません。
USER_DIR	リソース	CA Access Control には適用されません。
XGROUP	リソース	このクラスの各レコードは、CA Access Control に対するエンタープライズ ユーザを定義します。
XUSER	リソース	このクラスの各レコードは、CA Access Control に対してエンタープライズ グループを定義します。

注: CA Access Control データベース クラスの TCP および SURROGATE は、デフォルトではアクティブになっていません。

TCP クラスはアクティブだが、TCP レコードがなく、_default TCP リソースを変更していない旧リリースからアップグレードする場合、CA Access Control は、アップグレード中に、そのクラスを非アクティブにします。SURROGATE クラスについても、同様です。

以前のリリースで SURROGATE クラスをアクティブにして、SURROGATE レコードを定義、または SURROGATE レコードのいずれかの値をデフォルトから変更している場合、そのリリースからアップグレードすると、CA Access Control は、アップグレード後も SURROGATE クラスの設定を保持します。 クラスはアップグレード後もアクティブとなり、カーネル モードのインターセプトも引き続き有効化されます。

注: CA Access Control クラスの詳細については、「selang リファレンス ガイド」を参照してください。

ユーザ定義クラス

CA Access Control では、新しいクラスを定義し、そのクラスに適切なレコードを作成することによって抽象オブジェクトを保護できます。

例: データベース ビューのユーザ定義クラス

データベースを使用して独自のデータを格納および表示しているサイトがあるとします。

ユーザ定義クラス DATABASE_VIEWS を定義し、各データベース ビューをそのクラスのリソース メンバとして定義することができます。リソースに、そのデータベース ビューを作成する場合に必要なアクセス権限を定義する ACL を割り当てます。 ユーザがデータベース ビューを作成しようとしたときに、CA Access Control は、ユーザのアクセス権限をチェックし、ACL に基づいて作成を許可または拒否します。

ユーザ定義クラスのリソースでのワイルドカードの使用

ユーザ定義クラスのリソースの名前にワイルドカードを使用することで、複数の物理リソースに対応するリソース レコードを作成できます。ワイルドカードのパターンと一致する名前を持つ物理リソースはすべて、リソース レコードに関連付けられたアクセス権限によって保護されます。

使用できるワイルドカードは、以下のとおりです。

- * - 任意の複数文字に対応します。
- ? - 任意の 1 文字に対応します。

物理リソースの名前が複数のリソース レコード名と一致する場合、そのリソースには、ワイルドカードを除く、最も長い一致が使用されます。

CA Access Control では、リソース名として以下のワイルドカード パターンは使用できません。

- *
- /*
- /tmp/*
- /etc/*

ユーザ定義クラス - 例

銀行のサービスを提供しているシステムで、口座間での高額の送金を保護する場合を考えます。 このセキュリティを設定するには、以下の手順に従います。

1. 送金を表すレコードを格納するためのクラス（たとえば、TRANSFERS）を定義します。
2. 保護する必要がある金額レベルの送金ごとに、TRANSFERS クラスにレコードを定義します。

   たとえば、Upto.$1K、Upto.$1M、Upto.$10M、および Over.$10M という名前のレコードを定義します。

   送金を制御する必要があるその他のリソースを、TRANSFERS クラスのメンバとして定義します。
3. ユーザごとに、最大送金額の異なる実行権限を与えるには、TRANSFER クラスの各種レコードへのアクセスを許可または拒否します。
4. さらに、プログラムによる送金を処理するため、ユーザのアクセス許可をチェックしてから送金処理を許可するように、銀行の送金プログラムに CA Access Control API へのコールを挿入します。

Windows サービス保護

CA Access Control では、Windows サービスを保護することができます。 *Windows サービスは Windows のバックグラウンドで実行されるプログラムであり、UNIX におけるデーモンと同等の機能を果たします。*

CA Access Control の Windows サービス保護では、以下のいずれかをソースとするサービス アクセス イベントをインターセプトします。

- サービス管理および情報イベント

  CA Access Control は、サービス アクセスごとに services.exe プロセスをインターセプトします。 これには、サービスの起動や停止も含まれます。 たとえば、net start *service*、net stop *service* などはすべて保護されます。

  この場合のインターセプトされたイベントの監査は、保護対象サービスの名前を使用して行われます。

- サービス データベース管理イベント

  CA Access Control は、サービス制御管理データベースに対するレジストリ コールをインターセプトすることで、サービス状態のクエリや変更から保護します。 つまり、CA Access Control によって、保護対象サービスに関連付けられたレジストリ領域は自動的に保護されます。 実際には、CA Access Control は、サービス保護を定義するときに以下のレジストリ キーを保護します。

  HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥*service_name*
  HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥*service_name*¥*

  この場合にインターセプトされたイベントの監査は、完全なレジストリ パスを使用して行われます。

Windows サービスを保護する方法は、他のリソースを保護する場合と同じです。つまり、リソースをサービスに割り当て、アクセサをリソースのアクセス制御リストに追加します。 Windows サービスのリソース クラスは WINSERVICE です。 WINSERVICE リソースには、ACL と NACL の 2 つのアクセス制御リストがあります。 WINSERVICE のアクセス制御リスト内のエントリに有効なアクセス タイプは、以下のとおりです。

- Read
- 変更
- 先頭
- 停止

- 一時停止
- 再開

Windows サービス保護の有効化および無効化

CA Access Control の Windows サービス保護は有効または無効にすることができます。

Windows サービス保護を有効にするには、CA Access Control レジストリの Instrumentation¥PlugIns¥WinServiceplg セクション内の OperationMode を 1 に設定します。保護を無効にするには、OperationMode を 0 に設定します。

デフォルトでは、CA Access Control は、Windows サービスの保護を有効にします。

CA Access Control によって Windows サービスを保護するには、保護を有効にして、WINSERVICE クラスをアクティブにする必要があります。

Windows サービスの保護

Windows サービスを保護できるので、Windows の操作を保護することもできます。

Windows サービスを保護するには、以下の手順に従います。

1. Windows サービス保護が有効であること (P. 70) を確認します。
2. WINSERVICE クラスがアクティブであることを確認します (デフォルトでアクティブになっています) 。
3. 保護する Windows サービスと同じ名前で、CA Access Control に WINSERVICE レコードを作成します。

   注: Windows サービスの名前は、Windows サービスのプロパティ ダイアログの［全般］タブに表示されますが、そのタブ上の「表示名」と同一ではありません。
4. サービスに、アクセサとそのアクセス権限を割り当てます。

   これで、サービスは保護されます。

例: 印刷スプーラへのアクセスを制限する

Windows の印刷スプーラには、サービス名スプーラがあります。 以下の selang コマンドによって、WINSERVICE クラスがアクティブになり、スプーラへのデフォルト アクセスが「*読み取り*」に設定されます。

```
setoptions class+(WINSERVICE)
editres WINSERVICE(spooler) defacc(R)
```

Windows Server 2008 で IPv4 を使用しない Telnet 接続がセキュリティで保護されない

Windows Server 2008 では、IPv4 を使用しないと、CA Access Control で telnet 接続をセキュリティで保護できません。

Windows Server 2008 で、localhost の telnet 接続（localhost 間の telnet）を保護するには、/etc/HOSTS ファイルを以下のように変更します。

```
127.0.0.1       localhost
#       ::1                 localhost
127.0.0.1       <ドメイン サフィックスのないサーバ名>
```

お使いのコンピュータが IPv6 ドメイン内にある場合は、以下の行を追加します。

```
127.0.0.1    <ドメイン サフィックスのあるサーバ名>
```

保護対象 Windows サービスへのアクセスの試みの表示

CA Access Control は、Windows サービスを保護する場合、そのサービスに関連するアクセスの試みをインターセプトして、監査ログに記録します。このようなアクセスの試みは、サービスを管理するために（起動、停止など）services.exe プロセスを使用した結果である場合と、保護対象サービスのサービス データベース管理領域へのレジストリ アクセスの結果である場合があります。 services.exe プロセスを使用した結果によるアクセスでは、監査ログにサービス名しか記録されないのに対し、レジストリ アクセスの結果によるアクセスでは、完全なレジストリ パスが記録されます。Windows サービスに関連するすべてのアクセスの試みを表示するには、ワイルドカードを使用する必要があります。

保護対象 Windows サービスへのアクセス試行を表示するには、クラス WINSERVICE とリソース名 **myService** の監査レコードをフィルタ処理する監査フィルタを作成します。

CA Access Control では、定義した WINSERVICE リソースに対するすべての監査レコードが表示されます（アクセス試行が、レジストリを介するものか、サービス管理インターフェースを介するものかは関係ありません）。

例: 印刷スプーラ サービスへのすべてのアクセスの試みを表示する

この例では、以下のように、アクセス権を持たない印刷スプーラ サービスを CA Access Control に対して定義したとします。

```
er winservice spooler defaccess(none) owner(nobody)
```

以下のように seaudit ユーティリティを使用して、印刷スプーラ サービスへのすべてのアクセスの試みを一覧表示することができます。

```
seaudit -resource WINSERVICE *spooler* *
```

このコマンドにより、印刷スプーラ サービスに対するアクセス試行に関して記録された、クラス WINSERVICE のすべての監査レコードが一覧表示されます。 出力結果は以下のようになります。

```
seaudit - Audit log lister
3 Apr 2008 16:53:48 D WINSERVICE   bigHost1¥Administrator Read       69  2 Spooler
   c:¥WINDOWS¥system32¥services.exe bigHost1.comp.com
3 Apr 2008 16:53:48 D WINSERVICE   bigHost1¥Administrator Read       69  2 Spooler
   c:¥WINDOWS¥system32¥services.exe bigHost1.comp.com
3 Apr 2008 16:53:50 D WINSERVICE   bigHost1¥Administrator Read       69  2 Spooler
   c:¥WINDOWS¥system32¥services.exe bigHost1.comp.com
3 Apr 2008 16:53:50 D WINSERVICE   bigHost1¥Administrator Read       69  2 Spooler
```

```
   c:¥WINDOWS¥system32¥services.exe bigHost1.comp.com
3 Apr 2008 16:53:53 D WINSERVICE   bigHost1¥Administrator Read       69  2 Spooler
   c:¥WINDOWS¥system32¥services.exe bigHost1.comp.com
3 Apr 2008 16:53:53 D WINSERVICE   bigHost1¥Administrator Read       69  2 Spooler
   c:¥WINDOWS¥system32¥services.exe bigHost1.comp.com
03 Apr 2008 16:54:10 D WINSERVICE   bigHost1¥Administrator Read       69  2
HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥Spooler
   C:¥WINDOWS¥regedit.exe bigHost1.comp.com
03 Apr 2008 16:54:10 D WINSERVICE   bigHost1¥Administrator Read       69  2
HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥Spooler
   C:¥WINDOWS¥regedit.exe bigHost1.comp.com
03 Apr 2008 16:54:19 D WINSERVICE   bigHost1¥Administrator Read       69  2
HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥Spooler
   C:¥WINDOWS¥regedit.exe bigHost1.comp.com
03 Apr 2008 16:54:26 D WINSERVICE   bigHost1¥Administrator Read       69  2
HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥Spooler
   C:¥WINDOWS¥regedit.exe bigHost1.comp.com
03 Apr 2008 16:54:26 D WINSERVICE   bigHost1¥Administrator Modify     69  2
HKEY_LOCAL_MACHINE¥SYSTEM¥CurrentControlSet¥Services¥Spooler
   C:¥WINDOWS¥regedit.exe bigHost1.comp.com

Total records displayed 11
```

Windows レジストリ保護

CA Access Control では、Windows レジストリを保護することができます。

レジストリ キーを保護するには、クラス REGKEY のリソースをキーに割り当てます。 他のリソースと同じように、キーでアクセス権限を指定できます。

キーに対してアクセス権限を指定しても、キーのサブキーのアクセスには影響しません。ただし、サブキーの列挙（一覧表示）は例外で、キーの読み取りアクセス権が必要になります。

CA Access Control でサポートされるのは、Windows Server 2003 以降の Windows システム上の AC 環境における REGVAL リソースのみです。 これらのシステムでは、CA Access Control は、REGVAL クラスでレジストリ値を保護します。REGKEY アクセス権限は、キーの値へのアクセスに影響しません。

Windows Server 2003 より前のシステム上の AC 環境では、CA Access Control は REGVAL リソースをサポートしていません。REGKEY レコードに適用されるアクセス権限が、キーの値へのアクセスに影響します。

REGKEY レコードと REGVAL レコードの構造は同じです。 各レコードには、以下のアクセス制御リストが含まれています。

- ACL
- CALACL
- NACL
- PACL

REGVAL レコードと REGKEY レコードの両方において、以下の同じアクセスタイプが許可されます。

- READ
- WRITE
- DELETE
- NONE

注: CA Access Control のレジストリ保護では、ハイブのロードおよびアンロードのレジストリ操作は保護されません。 Windows Server 2008 以降のシステムでは、アクセサがアクセス権が NONE の保護されたレジストリ値にアクセスしようとした場合、CA Access Control は REG_NONE の値を返します。 REG_NONE の値は、値は存在するけれども値が指定されていないことを確認します。

Windows レジストリ エントリの保護

Windows レジストリ エントリを保護できるので、Windows 操作を追加保護することができます。

Windows レジストリ エントリを保護するには、以下の手順に従います。

1. REGKEY クラス レコードと REGVAL クラス レコードを使用する場合は、これらのレコードがアクティブであることを確認します（デフォルトでアクティブになっています）。
2. 保護するレジストリ キーまたは値の名前で、REGKEY レコードまたは REGVAL レコードを作成します。

   注: キーまたは値を指定する場合は、完全なレジストリ パス名を使用してください。 ワイルドカードを使用してキーにネストされているすべてのサブキーまたはサブキーの値を指定することができます。

   これで、レジストリ エントリは、CA Access Control がレコードに提供するデフォルトのアクセス権で保護されます。
3. （オプション）ユーザおよびグループと、そのアクセス権限を、REGKEY レコードまたは REGVAL レコードに含まれる適切なアクセス制御リストに割り当てます。

例: レジストリ キーに対するデフォルトのアクセス権を NONE に設定する

以下の selang コマンドは、レジストリ キーに対するデフォルトのアクセス権を NONE に設定します。

```
er REGKEY HKEY_LOCAL_MACHINE¥SOFTWARE¥Test¥Key1 defacc(NONE) owner(nobody)
```

この結果、key1 に対するデフォルトのアクセス権は以下のようになります。

Action	Windows Server 2003 以前のシステム	Windows Server 2003 以降のシステム	Windows Server 2008 以降のシステム
サブキーの列挙	拒否	拒否	拒否
キーのクエリ、変更、名前変更、または削除	拒否	拒否	拒否
キーへのハイブのロードおよびアンロード	拒否	拒否	拒否

Action	Windows Server 2003 以前のシステム	Windows Server 2003 以降のシステム	Windows Server 2008 以降のシステム
値の列挙	拒否	拒否	許可
値の読み取り、作成、名前変更、または削除	拒否	許可	許可
サブキーのサブキーを列挙	拒否	許可	許可
サブキーの作成	許可	許可	許可
サブキーのクエリ、変更、名前変更、または削除	許可	許可	許可
サブキーへのハイブのロードまたはアンロード	許可	許可	許可

例: レジストリ キーに対するデフォルトのアクセス権を READ に設定する

以下の selang コマンドは、レジストリ キーに対するデフォルトのアクセス権を READ に設定します。

```
er REGKEY HKEY_LOCAL_MACHINE¥SOFTWARE¥Test¥Key1 defacc(READ) owner(nobody)
```

この結果、key1 に対するデフォルトのアクセス権は以下のようになります。

Action	Windows Server 2003 以前のシステム	Windows Server 2003 以降	Windows Server 2008 以降
サブキーの列挙	許可	許可	許可
キーの読み取り	許可	許可	許可
キーの変更、名前変更、または削除	拒否	拒否	拒否
キーへのハイブのロードおよびアンロード	拒否	拒否	拒否
値の列挙	許可	許可	許可

Action	Windows Server 2003 以前のシステム	Windows Server 2003 以降	Windows Server 2008 以降
値の読み取り	許可	許可	許可
値の作成、名前変更、または削除	拒否	許可	許可
サブキーのサブキーを列挙	許可	許可	許可
サブキーの作成	許可	許可	許可
サブキーのクエリ、変更、名前変更、または削除	許可	許可	許可
サブキーへのハイブのロードまたはアンロード	許可	許可	許可
サブキーの値の列挙	許可	許可	許可
サブキーの値の作成	許可	許可	許可

例: レジストリ キーのワイルドカードに対するデフォルトのアクセス権を NONE に設定する

以下の selang コマンドは、レジストリ キー内のすべてのサブキーに対するデフォルトのアクセス権を NONE に設定します。

```
er REGKEY HKEY_LOCAL_MACHINE¥SOFTWARE¥Test¥Key1¥* defacc(NONE) owner(nobody)
```

ワイルドカード(*)は Key1 に適用されませんが、Key1 のすべてのサブキーに適用されます。これは、Key1 のすべてのサブキーに対してあらゆる形式のアクセスが拒否されるという意味です。また、親保護のルールにより、Key1 の名前変更または削除も拒否されます。

このコマンドは、Key1 の値へのアクセスを許可します。 Key1 のサブキーの値（たとえば、Key1¥subkey1¥ の値）に対するアクセスは、Windows システム間で異なります。

- Windows Server 2003 以降のシステムでは、このコマンドは、Key1 のサブキーの値を列挙するためのアクセスは拒否しますが、値を作成、名前変更、削除、および読み取るためのアクセスは許可します。
- Windows Server 2003 より前のシステムでは、このコマンドは、Key1 のサブキーの値に対するすべてのアクセスを拒否します。

例: レジストリ値に対するデフォルトのアクセス権を NONE に設定する

Windows Server 2003 以降のシステムでは、以下の selang コマンドで、アクセス権を NONE に設定して特定のレジストリ値を保護します。

```
er REGVAL HKEY_LOCAL_MACHINE¥SOFTWARE¥TestKey¥value1 defacc(NONE) owner(nobody)
```

注: Windows Server 2008 以降のシステムでは、アクセサがアクセス権が NONE の保護されたレジストリ値にアクセスしようとした場合、CA Access Control は REG_NONE の値を返します。 REG_NONE の値は、値は存在するけれども値が指定されていないことを確認します。

ファイル ストリームの保護

ストリームとは、バイト シーケンスのことです。 ファイル ストリームにはファイル データとファイルに関する追加情報が格納されています。 たとえば、キーワードやメタデータを格納するストリームを作成できます。

注: ファイル ストリームは、NTFS ファイル システムでのみ使用可能です。ファイル システムの詳細については、Microsoft Developer Network（MSDN）Library の Web サイトを参照してください。

FILE ルールを作成した場合、CA Access Control は、そのファイルのデフォルトのデータ ストリームを自動的に保護します。 たとえば、ファイル c:¥foo.txt を保護するルールでは、c:¥foo.txt::$DATA へのアクセス許可も制御されます。 ただし、CA Access Control では、デフォルト以外のデータ ストリームは自動的に保護されません。デフォルト以外のデータ ストリームについては、追加のファイル保護ルールを作成する必要があります。

ファイル ストリームを保護するには、以下の*いずれか*を実行します。

- 特定のストリームを保護するには、以下の形式でファイル ルールを作成します。

  drive:¥*path*¥*filename*.*ext*:*stream*

- 特定のストリーム内の特定のストリーム タイプを保護するには、以下の形式でファイル ルールを作成します。

  drive:¥*path*¥*filename*.*ext*:*stream*:*type*

- すべてのストリームを保護するには、以下の形式で包括的なファイル ルールを作成します。

  drive:¥*path*¥*filename*.*ext*:*

例: すべてのファイル ストリームを保護する

以下の selang コマンドは、ファイル c:¥foo.txt 内のすべてのストリームを保護する包括的なファイル ルールを作成します。

```
er file c:¥foo.txt:* owner(nobody) defaccess(none)
```

例: 特定のストリームを保護する

以下の selang コマンドは、ファイル c:¥foo.txt 内のストリーム *mystream* を保護するファイル ルールを作成します。

```
er file c:¥foo.txt:mystream owner(nobody) defaccess(none)
```

内部ファイルの保護

インストール中に、CA Access Control により、2 つのタイプの内部ファイルを保護するルールが書き込まれます。

- 内部ルール -- 設定ファイル、ログ ファイル、およびデータベース ファイルを保護します。

  内部ルールは削除できません。

- デフォルト ルール -- 通信の暗号化および認証に使用するルート証明書およびサーバ証明書などの機密ファイルを保護します。

  デフォルト ルールはインストール後に削除できます。

内部ファイル ルール

内部ファイル ルールにより、設定ファイル、ログ ファイル、およびデータベース ファイルが保護されます。 内部ファイル ルールは、selang に表示されず、削除できません。 しかし、FILE ルールを記述して、内部ファイル ルールを置き換えることができます。 これらの FILE ルールを削除すると、CA Access Control では内部ファイル ルールが復帰します。

データベース ファイルを除いて、CA Access Control により内部ファイル ルールで保護されるファイルには、以下のアクセス権限があります。

- CA Access Control の内部プロセスへのフル アクセス
- その他のすべてのアクセサに関する読み取りアクセスと実行アクセス（関連する場合）

CA Access Control により内部ファイル ルールで保護されるデータベース ファイルには、以下のアクセス権限があります。

- CA Access Control の内部プロセスにはデータベースに対するフル アクセス権限があります。
- NT AUTHORITY¥System ユーザにはデータベースに対する読み取りアクセス権限があります。
- 他のすべてのアクセサにはデータベースに対するアクセス権限がありません。

  注: 他のすべてのアクセサ用のデフォルト アクセス権限は r12.5 SP3 で変更されました。 以前のリリースでは、他のすべてのアクセサはデフォルトでデータベース ファイルに対して読み取りアクセス権を持っていました。

CA Access Control では、内部ファイル ルールで以下のファイルが保護されます。 表の 2 番目の列には、ファイルの場所を示すレジストリ サブキーおよびエントリが一覧表示されます (該当する場合)。CA Access Control では、以下のレジストリ キーで、レジストリ エントリが作成されします。

```
HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl
```

注: 一部のファイルの場所は内部的に定義され、対応するレジストリ エントリがありません。 これらのファイルの場所を設定することはできません。

ファイル	レジストリ サブキーとエントリ	デフォルトのファイルの場所
seosdrv.sys	-	%SystemRoot%¥system32¥drivers¥seosdrv.sys
cainstrm.sys	-	%SystemRoot%¥system32¥drivers¥cainstrm.sys
drveng.sys	-	%SystemRoot%¥system32¥drivers¥drveng.sys
pwdchange.dll	-	%SystemRoot%¥system32¥pwdchange.dll
SUSRAUTH.dll	-	%SystemRoot%¥system32¥SUSRAUTH.dll
eACSubAuth.dll	-	%SystemRoot%¥system32¥eACSubAuth.dll
eACPasswordFltr.dll	-	%SystemRoot%¥system32¥eACPasswordFltr.dll
すべてのデータベース ファイル	SeOSD¥dbdir	*ACInstallDir*¥Data¥seosdb

ファイル	レジストリ サブキーとエントリ	デフォルトのファイルの場所
すべてのヘルプ ファイル	lang¥help_path	*ACInstallDir*¥Data¥help
すべてのバイナリ	-	*ACInstallDir*¥bin
seosd.trace	SeOSD¥trace_file	*ACInstallDir*¥log
seos.audit	logmgr¥audit_log	*ACInstallDir*¥log
seos.audit.bak	logmgr¥audit_back	*ACInstallDir*¥log
seos.error	logmgr¥error_log	*ACInstallDir*¥log
seos.error.bak	logmgr¥error_back	*ACInstallDir*¥log
seos.msg	message¥filename	*ACInstallDir*¥Data
stop.ini	STOP¥STOPIniFileName	*ACInstallDir*¥Data
stopsignature.dat	STOP¥STOPSignatureFileName	*ACInstallDir*¥Data
response.ini	SeOSD¥ResponseFile	*ACInstallDir*¥Data
audit.cfg	logmgr¥AuditFiltersFile	*ACInstallDir*¥Data

注: 設定の詳細については、「*リファレンス ガイド*」を参照してください。

デフォルト ファイル ルール

CA Access Control では、機密ファイルを保護するために、インストール中にデフォルト ファイル ルールが作成されます。デフォルト ファイル ルールは、selang に表示され、削除できます。

以下の表では、CA Access Control によりデフォルト ファイル ルールで保護される機密ファイルと、そのアクセス権限および許可されているアクセサが一覧表示されています。

この表では、*PMDBDir* は Policy Model データベース（PMDB）があるディレクトリであり、*pmd_name* は各 Policy Model の名前です。デフォルトでは、*PMDBDir* は *ACInstallDir*¥Data にあります。 *PMDBDir* の場所は、以下のレジストリ エントリに定義されています。

```
HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥Pmd¥_Pmd_directory_
```

ファイル	デフォルト アクセス	許可されているアクセサ
ACInstallDir¥data¥crypto¥crypto.dat	なし	sechkey
ACInstallDir¥data¥crypto¥def_root.pem*	なし	sechkey
ACInstallDir¥data¥crypto¥sub.key	なし	sechkey
ACInstallDir¥data¥crypto¥sub.pem	なし	sechkey
ACInstallDir¥log¥policyfetcher.log	Read	+policyfetcher
PMDBDir¥*pmd_name*	Read、Chdir	-
PMDBDir¥*pmd_name*¥*	Read、Execute	-

第 5 章: 許可の管理

このセクションには、以下のトピックが含まれています。

アクセス権限

CA Access Control の主な目的は、アクセス権限（アクセス権とも呼ばれます）を割り当て、適用することです。

アクセス権限には、常に以下のコンポーネントがあります。

- アクセスの適用先のリソース（ファイル、ホスト、端末など）。
- アクセスのタイプ（読み取り、書き込み、削除、ログイン、実行など）。
- アクセサ（ユーザまたはグループのいずれか）。

以下の 1 つ以上に当てはまる場合、ユーザに対してリソースにアクセスする権限が割り当てられます。

- ユーザがリソースの ACL によって許可されている。
- ユーザが、アクセス権限が割り当てられたグループのメンバ。
- ユーザが、アクセス権限が割り当てられたプログラムを実行してアクセス。 たとえば、ユーザには、SPECIALPGM クラス内のプログラムを実行する権限、または SUDO クラス内のコマンドを実行する権限が割り当てられている。

注: クラス別のアクセス権限の詳細については、「selang リファレンス ガイド」を参照してください。

アクセス権限の設定 - 例

例: 内部ユーザへ読み取りアクセス権限を付与する

以下の selang コマンドは、端末 tty30 の ACL に内部ユーザ internal_user を追加し、端末への読み取りアクセス権限を付与します。

```
authorize TERMINAL tty30 access(READ) uid(internal_user)
```

例: エンタープライズ ユーザへ読み取りアクセス権限を付与する

以下の selang コマンドは、端末 tty30 の ACL にエンタープライズ ユーザ Terry を追加し、端末への読み取りアクセス権限を付与します。

```
authorize TERMINAL tty30 access(READ) xuid(Terry)
```

例: リソースに対するエンタープライズ ユーザのアクセス権限を変更する

以下の selang コマンドは、端末 tty30 への Terry のアクセスを none に設定し、Terry のアクセスを拒否します。

```
authorize TERMINAL tty30 access(NONE) xuid(Terry)
```

例: エンタープライズ ユーザのアクセス権限をリソースから削除する

以下の selang コマンドは、端末 tty30 の ACL から Terry を削除します。

```
authorize- TERMINAL tty30 xuid(Terry) access-
```

これで、Terry には、端末へのデフォルトのアクセス権が割り当てられます。

例: エンタープライズ ユーザにサブ管理者アクセスを付与する

以下の selang コマンドは、エンタープライズ ユーザ Terry を、ユーザとファイルを管理する権限を持つサブ管理者として設定します。

```
authorize ADMIN USER xuid(Terry)
authorize ADMIN FILE xuid(Terry)
```

アクセス制御リスト

リソースに対するアクセス権限は、アクセス制御リストに指定されます。各リソース レコードには、少なくとも 2 つのアクセス制御リストが割り当てられます。

ACL

リソースへのアクセスが許可されるアクセサと、そのアクセサが許可されるアクセスのタイプを指定します。

NACL

リソースへのアクセスが拒否されるアクセサと、そのアクセサが拒否されるアクセスのタイプを指定します。

アクセス権限は、ユーザがローカルでログインするかどうかなど、アクセスに関する状況によっても異なります。

条件付きアクセス制御リスト

条件付きアクセス制御リスト（CACL）は、ACL の拡張機能です。 アクセサがリソースへのアクセスを試みたときに、リソースの ACL と NACL にそのユーザのアクセス権限が定義されていない場合、CA Access Control は条件付きアクセス制御リストを確認します。

条件付きアクセス制御リストでは、アクセスが特定の方法による（たとえば、指定されたプログラムの使用による）場合のリソースへのアクセスを指定します。

たとえば、条件付きアクセス制御リストを使用して、Program Pathing ルールを定義できます。

CA Access Control では、以下の条件付きアクセス制御リストを使用することができます。

- プログラム アクセス制御リスト（PACL）
- TCP クラス アクセス制御リスト
- CALENDAR クラス アクセス制御リスト

条件付きアクセス制御リストのエントリを定義するには、selang authorize コマンドの via オプションを使用します。

他のアクセス制御リストと同様に、条件付きアクセス制御リストの各エントリでは、リソースへのアクセスが許可されるアクセサと、許可されるアクセスのタイプを指定します。 さらに、条件付きアクセス制御リストのエントリでは、権限を割り当てる条件も指定します。 PACL の条件とは、アクセサがアクセスをするために実行する必要があるプログラムの名前です。

例: PACL の使用

エンタープライズ ユーザ sysadm1 がプログラム secured_su を実行することによってスーパーユーザになれるようにするには、以下の selang コマンドを使用して、条件付きアクセス ルールを指定します。

```
authorize SURROGATE user.root xuid(sysadm1) via(pgm(secured_su))
```

defaccess - デフォルト アクセス フィールド

リソースのレコードには、デフォルト アクセス フィールド defaccess を含めることができます。 defaccess フィールドの値には、リソース アクセス制御リストのいずれでもカバーされないアクセサに許可するアクセス権限を指定します。

リソースに対するアクセス権限を決定する方法

アクセサがリソースへのアクセスを試みると、CA Access Control は、結果が得られるまで、事前定義された順序で 1 つ以上のチェックを実行することでアクセス権限をチェックします。 チェックによってアクセスの結果（アクセスの拒否または許可）が得られると、CA Access Control はそれ以上チェックを実行せず、代わりに結果を返します。

これらのチェックを実行する順序は重要です。リソースごとに、CA Access Control はデフォルトでは以下の順序でアクセス レコードをチェックします。

1. リソースの時刻ベースの制限
2. リソースの所有権（所有者はアクセスが許可される）
3. B1 チェック

4. リソースの NACL
5. リソースの ACL
6. リソースの PACL
7. リソースの defaccess フィールド

最後の 2 つのチェックの順序は、accpacl オプションの設定によって決まります。 リソース PACL の使用を無効にするには、selang コマンドの setoptions setpacl- を使用します。

1 つのアクセス制御リストに、同じユーザに影響する複数のエントリが含まれていることがあります。 たとえば、ユーザを明示的に指定するエントリと、そのユーザが属する各グループに対するエントリが含まれることがあります。 CA Access Control は、各レベルで有効なすべてのエントリをチェックしてから、次のレベルに進みます。 各レベルで競合するルールを解決する方法の詳細については、「ユーザのアクセス権限とグループのアクセス権限との相互作用 (P. 90)」を参照してください。

例: ファイルのアクセス許可の結果

以下の表は、アクセサ user1 がリソース ファイル 1 の読み取りを試みることを前提としています。

以下の表では、CA Access Control は accpacl オプションのデフォルトの設定に従って PACL を使用します。

user1 に対する NACL 内のエントリ	user1 に対する ACL 内のエントリ	user1 に対する PACL 内のエントリ	defaccess 内のエントリ	結果的に付与されるアクセス許可
Read	*(任意)*	*(任意)*	*(任意)*	読み取り拒否
(未定義)	なし	*(任意)*	*(任意)*	読み取り拒否
(未定義)	Read	*(任意)*	*(任意)*	読み取り許可
(未定義)	*(未定義)*	via pgm securereader	*(任意)*	securereader プログラムの実行によって読み取り許可
(未定義)	*(未定義)*	*(未定義)*	Read	読み取り許可

エントリが *(未定義)* と表示されている場合、これは、user1 に対するエントリがアクセス制御リストに存在しないことを意味します。

エントリが *(任意)* と表示されている場合、これは、CA Access Control によるチェックが行われず、アクセス制御リスト内のエントリは関係ないことを意味します。

CA Access Control は、左から右にチェックします。 すべての行で、アクセスが定義されているセルの右側に位置するセルの値は、*(任意)* になることに注意してください。 逆に、アクセスが定義されているセルの左側にあるセルの値はすべて *(未定義)* になります。

ユーザのアクセス権限とグループのアクセス権限との相互作用

ユーザ、およびユーザが属するグループに対して、アクセス権限を明示的に許可または拒否することができます。 場合によってはこれらのアクセス権限が競合することがあります。以下の例では、ユーザが 2 つのグループ（Group 1 と Group 2）のメンバであるときに競合するアクセス権限が同じリソースに割り当てられた場合、どのような結果になるかを示します。

累積グループ権限 (P. 91) オプションが設定されていることを前提とします（デフォルトの設定）。

ユーザのアクセス権限	Group 1 のアクセス権限	Group 2 のアクセス権限	最終的なアクセス権限
拒否されたアクセス	*(任意)*	*(任意)*	拒否されたアクセス
アクセス許可	*(任意)*	*(任意)*	アクセス許可
(未定義)	アクセス許可	*(未定義)*	アクセス許可
(未定義)	*(未定義)*	アクセス許可	アクセス許可
(未定義)	アクセス許可	アクセス許可	アクセス許可
(未定義)	拒否されたアクセス	*(任意)*	拒否されたアクセス
(未定義)	*(任意)*	拒否されたアクセス	拒否されたアクセス

エントリが (未定義) と表示されている場合、これは、ユーザまたはグループに対するエントリが定義されていないことを意味します。

エントリが (任意) と表示されている場合、これは、CA Access Control によるチェックが行われず、アクセス権限は関係ないことを意味します。

累積グループ権限 (ACCGRR)

累積グループ権限オプション (ACCGRR) では、CA Access Control がリソースの ACL をチェックする方法を制御します。 ACCGRR が有効な場合、CA Access Control は、ACL で、ユーザが属するすべてのグループで許可されている権限をチェックします。ACCGRR が無効な場合、CA Access Control は、ACL で適用可能なエントリのいずれかに値 none が含まれているかどうかをチェックします。none が含まれている場合、アクセスは拒否されます。none が含まれていない場合、CA Access Control は、ACL 内の最初の適用可能なグループ エントリを除くすべてのグループ エントリを無視します。このオプションはデフォルトで有効です。

ACCGRR オプションを有効にするには、以下の selang コマンドを使用できます。

```
setoptions accgrr
```

ACCGRR オプションを無効にするには、以下の selang コマンドを使用できます。

```
setoptions accgrr-
```

セキュリティ レベル、セキュリティ カテゴリ、およびセキュリティ ラベル

セキュリティ レベルとセキュリティ カテゴリは、リソースへのアクセスを制限する追加の方法を提供して、アクセス制御リストを補完します。

セキュリティ ラベルは、セキュリティ レベルとセキュリティ カテゴリを 1 つにまとめて、管理を簡易化する手段です。

セキュリティ レベル

セキュリティ レベルは、ユーザおよびリソースに割り当てることのできる 0 から 255 までの整数です。リソースのアクセス制御リストでユーザにアクセス権限が付与されていても、アクセサのセキュリティ レベルがリソースのセキュリティ レベルより低い場合、そのアクセサはそのリソースにアクセスできません。リソースのセキュリティ レベルがゼロの場合、そのリソースに対してセキュリティ レベルのチェックは実行されません。

セキュリティ レベルがゼロのアクセサは、セキュリティ レベルがゼロ以外のリソースにアクセスできません。

セキュリティ カテゴリ

セキュリティ カテゴリは、CATEGORY クラスにあるレコードの名前です。セキュリティ カテゴリは、アクセサとリソースに割り当てることができます。 リソースに割り当てられているすべてのセキュリティ カテゴリにアクセサが割り当てられている場合のみ、そのアクセサはリソースにアクセスできます。

セキュリティ ラベル

セキュリティ ラベルは、SECLABEL クラスにあるレコードの名前です。 セキュリティ ラベルによって、セキュリティ ラベルと複数のセキュリティ カテゴリを 1 つにまとめることができます。 セキュリティ ラベルをアクセサまたはリソースに割り当てると、そのセキュリティ ラベルに関連付けられたセキュリティ レベルとセキュリティ カテゴリの組み合わせが、アクセサまたはリソースに設定されます。 セキュリティ ラベルは、アクセサまたはリソースに設定された特定のセキュリティ レベルおよびセキュリティ カテゴリよりも優先されます。

例: セキュリティ ラベル High_Security の使用

High_Security は、セキュリティ レベル 255 と、セキュリティ カテゴリ MANAGEMENT および CONFIDENTIAL を含むセキュリティ ラベルであるとします。

ユーザ user1 をセキュリティ ラベル High_Security に割り当てた場合、user1 には、セキュリティ レベル 255 と、セキュリティ カテゴリ MANAGEMENT および CONFIDENTIAL が設定されます。

第 6 章: アカウントの保護

このセクションには、以下のトピックが含まれています。

別のユーザとしての実行の保護

CA Access Control で SURROGATE クラスを有効にすると、別のユーザとしての実行の保護を有効にします。 別のユーザとしての実行の保護では、特定のルールで変更が許可されている場合にのみ、あるユーザまたはグループが SID （セキュリティ識別子）を別の SID に変更できるように指定できます。 この機能を使用すると、ユーザに権限がない場合、ユーザが別のユーザの識別子に変更できないようにします。

注: セキュリティ識別子とは、オペレーティング システムに対してユーザまたはグループを識別する数値です。

たとえば、どのユーザも管理者として実行できないように CA Access Control ルールを定義するとします。 ユーザ Tom がいくつかのタスクを管理者として実行するプログラムを実行します。 この場合、Tom は管理者として実行する権限を持たないため、CA Access Control はこのプログラムの実行を許可しません。

別のユーザとしての実行の保護は、以下の 2 つのモードで実行できます。

- ユーザ モード インターセプト
- カーネル モード インターセプト

ユーザ モード インターセプト

ユーザ モード インターセプトを有効にすると、CA Access Control は Windows の RunAs ユーティリティから開始される、別のユーザとしての実行要求のみをインターセプトします。 ユーザ モード インターセプトは、サポートされるすべての Windows バージョンで使用可能です。

注: 別のユーザとしての実行の保護を有効にする（SURROGATE クラスを有効にする）場合、ユーザ モード インターセプトはデフォルトで有効になります。

ユーザ モード インターセプトには、以下のメリットがあります。

- CA Access Control は、別のユーザとしての実行要求を行ったユーザを識別します。

  RunAs ユーティリティを含む多くの Windows アプリケーションでは、NT AUTHORITY¥SYSTEM ユーザが要求を実行したユーザの代理となり、別のユーザとしての実行要求を行います。 ユーザ モード インターセプトでは、要求を行う NT AUTHORITY¥SYSTEM ユーザではなく、ユーティリティを実行したユーザを識別します。 たとえば、Tom が管理者として実行するために RunAs を実行すると、NT AUTHORITY¥SYSTEM ユーザが別のユーザとしての実行要求を行います。次に、CA Access Control は要求を行っているユーザが Tom であることを識別します。

- ユーザが RunAs ユーティリティを実行する場合にのみ、CA Access Control は別のユーザとしての実行要求をインターセプトします。

  これによって、パフォーマンスに及ぼす影響を最小限に抑えます。

ユーザ モード インターセプトのデメリットは、CA Access Control がすべての Windows プロセスから発生する、別のユーザとしての実行要求をすべてインターセプトするとは限らないという点です。

カーネル モード インターセプト

カーネル モード インターセプトを有効にすると、CA Access Control はすべての Windows プロセスから、別のユーザとしての実行要求をすべてインターセプトします。 カーネル モード インターセプトは、サポートされるすべての Windows バージョンで使用可能というわけではありません。

注: カーネル モード インターセプトが使用できない Windows バージョンの詳細については、「*リリース ノート*」を参照してください。

カーネル モード インターセプトのメリットは、Windows コンピュータで実行される別のユーザとしての実行要求をすべて保護できるという点です。

カーネル モード インターセプトには、以下のデメリットがあります。

- NT AUTHORITY¥SYSTEM ユーザが、要求を実行したユーザの代理となって別のユーザとしての実行要求を行うと、CA Access Control は実際に要求を行ったユーザを識別できません。

  たとえば、RunAs、FTP および telnet 要求は、すべて NT AUTHORITY¥SYSTEM ユーザによって実行されます。 Tom が管理者として実行するために RunAs を実行すると、NT AUTHORITY¥SYSTEM ユーザが別のユーザとしての実行要求を行います。次に、CA Access Control は要求を行っているユーザが NT AUTHORITY¥SYSTEM であると識別します。

- CA Access Control は、OS が通常操作の一部として行うすべての要求をインターセプトします。そのため、パフォーマンスに影響を及ぼす場合があります。

  CA Access Control は別のユーザとしての実行要求をキャッシュしますが、認証エンジンではその要求に関連した多くのイベントを認証する必要があります。

CA Access Control が別のユーザとしての実行要求に応答する方法

SURROGATE クラスの各レコードは、特定のユーザを別のユーザとしての実行から保護するための制限を定義します。 CA Access Control では、別のユーザとしての実行要求を権限のあるユーザのみがアクセスできる抽象オブジェクトとして扱います。SURROGATE クラスのレコードは、代理 (別のユーザとしての実行) の保護が適用される各ユーザまたはグループを表します。

あるユーザまたはグループが、別のユーザまたはグループとして実行することを要求した場合、CA Access Control は以下を実行します。

1. 要求を実行したユーザまたはグループの SURROGATE レコードのアクセス権限を確認します。SURROGATE レコードによっては、以下の*いずれか*が発生します。

   - 要求を実行したユーザまたはグループの SURROGATE レコードが、別のユーザとして実行することを許可または拒否します。

     CA Access Control は、別のユーザとしての実行要求を許可または拒否する際に、SURROGATE レコードのアクセス権限を使用します。

   - ユーザまたはグループには、SURROGATE レコードはありません。

     プロセスを手順 2 に進めます。

2. ユーザまたはグループのデフォルト SURROGATE レコードのアクセス権限を以下のように確認します。

   - 要求がユーザから実行された場合、CA Access Control はそのユーザに USER._default SURROGATE レコードに定義されているアクセスタイプを付与します。
   - 要求がグループから実行された場合、CA Access Control はそのグループに GROUP._default SURROGATE レコードに定義されているアクセス タイプを付与します。

   注: USER._default、GROUP._default、および _default SURROGATEUSER のデフォルト アクセス権限は読み取り権限です。 この場合、ユーザまたはグループの SURROGATE レコード で別のユーザとしての実行要求が拒否されていない限り、CA Access Control は別のユーザまたはグループとしての実行要求をすべて許可することを意味します。 この動作を変更するには、USER._default および GROUP._default レコードのアクセス権限を変更してください。 また、_default SURROGATE レコードのアクセス権限を変更することによって、ユーザとグループに同じデフォルト設定を適用することもできます。

別のユーザとしての実行の有効化

別のユーザとしての実行を使用すると、特定のユーザおよびグループに対して、別のユーザとしての実行要求を許可または拒否するルールを設定できます。

別のユーザとしての実行を有効化する方法

1. （オプション）カーネル モード インターセプトを有効にする手順を以下に示します。
   a. CA Access Control を停止します。
   b. 以下のレジストリ エントリの値を 1 に変更します。

      ```
      HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥
      SeOSD¥SurrogateInterceptionMode
      ```

   c. CA Access Control を再起動します。

   注: ユーザ モード インターセプトはデフォルトで有効になっています。
2. selang コマンド プロンプト ウィンドウを開きます。
3. SURROGATE クラスを有効にします。

   ```
   setoptions class+(SURROGATE)
   ```
4. CA Access Control に実装する SURROGATE レコードの selang ルールを定義します。
5. （カーネル モード インターセプトのみ）実際のユーザの代理として別のユーザとしての実行要求を行う、SYSTEM ユーザのルールを定義します。

   ```
   auth SURROGATE USER.Administrator uid("NT AUTHORITY¥SYSTEM") acc(R)
   ```

   Windows では、多くのユーティリティおよびサービス（例：RunAS など）を実行した元のユーザを、ユーティリティを実行したユーザではなく、ユーザ「NT AUTHORITY¥SYSTEM」として識別します。 このユーティリティを実行したユーザが、別のユーザとして実行することを許可するように SYSTEM ユーザのルールを定義する必要があります。

例: 別のユーザとしての実行要求を許可する

以下の selang ルールでは、データベース内のレコードで別のユーザとしての実行を明示的に拒否しない限り、どのユーザでも別のユーザとして実行することができます。

```
editres SURROGATE _default defaccess(READ)
```

例: 特定のユーザに対して、別のユーザとしての実行を拒否する

以下の selang ルールでは、データベース内のレコードで別のユーザとしての実行を明示的に許可しない限り、どのユーザも別のユーザとして実行することはできません。

```
newres SURROGATE USER.Administrator defaccess(NONE)
```

例: グループに対して、別のユーザとしての実行を許可する

以下の例では、Administrators グループに属するメンバが Administrator として実行することを許可します。

```
authorize SURROGATE USER.Administrator gid("Administrators")
```

Surrogate DO 機能のセットアップ

多くの場合、オペレータ、プロダクション担当者、およびエンドユーザは、スーパーユーザのみが実行できるタスクを実行する必要があります。

これまでの方法では、これらのタスクを実行する必要があるすべてのユーザに、スーパーユーザのパスワードを知らせていました。これはサイトのセキュリティを脅かすことにつながります。 このため、安全な代替策としてパスワードの公開を禁止すると、システム管理者はユーザからの正当な要求によってさまざまなルーチン タスクを実行しなければならず、システム管理者の負荷が大きくなります。

Surrogate DO（sesudo）ユーティリティは、このジレンマを解消します。このユーティリティは、SUDO クラスに定義されているアクションの実行をユーザに許可します。SUDO クラスの各レコードにはスクリプトが保存されていて、スクリプトを実行できるユーザとグループが指定されています。それらのユーザやグループに、目的に応じて必要な許可が与えられます。

たとえば、ユーザがシステム ユーザであるかのように、「印刷スプーラ」サービスを起動する SUDO リソースを定義するには、以下の selang コマンドを入力します。

```
newres SUDO StartSpooler data("net start spooler")
```

この newres コマンドによって、一部のユーザだけが実行のシステム権限を使用できる保護されたアクションとして、StartSpooler が定義されます。

重要: data プロパティには、完全な絶対パス名を使用してください。 相対パス名を使用すると、保護されていないディレクトリに仕掛けられたトロイの木馬プログラムが、誤って実行される可能性があるからです。

さらに、authorize コマンドを使用して、StartSpooler アクションを実行する権限をユーザに与えることもできます。 たとえば、ユーザ *operator1* に「印刷スプーラ」サービスの起動を許可するには、以下のコマンドを入力します。

```
authorize SUDO StartSpooler uid(operator1)
```

また、authorize コマンドを使用して、保護されたアクションの実行をユーザに対して明示的に禁止することもできます。たとえば、ユーザ *operator2* に「印刷スプーラ」サービスの起動を許可しないようにするには、以下のコマンドを入力します。

```
authorize SUDO StartSpooler uid(operator2) access(None)
```

sesudo ユーティリティを実行すると、保護されたアクションが実行されます。 たとえば、ユーザ *operator1* が「印刷スプーラ」サービスを起動するには、以下のコマンドを入力します。

```
sesudo -do StartSpooler
```

この sesudo ユーティリティは、最初に SUDO アクションの実行権限がユーザにあるかどうかをチェックし、そのユーザにリソースの権限がある場合は、そのリソースに定義されているコマンド スクリプトを実行します。この例に示した sesudo は、StartSpooler アクションの実行権限が operator1 にあるかどうかをチェックした後に、「net start spooler」コマンドをシステム権限で起動します。

注: sesudo ユーティリティの詳細については、「*リファレンス ガイド*」を参照してください。

SUDO レコードの定義（タスクの委任）

SUDO クラスのレコードには、コマンド スクリプトが格納されています。ユーザは、借用した権限でそのスクリプトを実行できます。 権限を利用できるかどうかは、スクリプトを実行する sesudo コマンドと SUDO レコードの両方で厳密に制御されます。

注: 対話型の Windows アプリケーション用の SUDO レコードを作成する場合、SUDO レコード用の対話型のフラグを設定する必要があります。 対話型のフラグを設定しない場合、アプリケーションはバックグラウンドで実行されるため、ユーザは操作できません。 詳細については、「*トラブルシューティング ガイド*」を参照してください。

SUDO レコードでは、comment プロパティを特別な目的に使用します。通常、このような comment プロパティを data プロパティといいます。

comment プロパティの値は、コマンド スクリプトです。禁止（prohibited）または許可（permitted）するスクリプト パラメータ値が必要に応じて追加される場合もあります。 comment プロパティ値全体は一重引用符で囲む必要があります。トロイの木馬の侵入を防ぐために、実行可能ファイルは完全パス名で参照する必要があります。

comment プロパティの形式は、以下のとおりです。

```
comment('cmd[;[prohibited-values][;permitted-values]]')
```

prohibited 値および permitted 値のリストは省略できるため、comment プロパティの値は以下のように簡略化することもできます。

```
newres SUDO NET comment('net use')
```

このコマンドに指定されている簡略化された値は、sesudoNET コマンドで「net use」コマンドを実行することを表します。 特定のスクリプト パラメータ値が禁止されていないため、すべての値が許可されます。

ワイルドカードと強力な変数を使用すると、prohibited パラメータおよび permitted パラメータを柔軟に指定できるようになります。 使用できるワイルドカードは、Windows の標準的なワイルドカードです。 禁止するパラメータおよび許可するパラメータには、以下の変数を指定することもできます。

変数	説明
$A	英字
$G	既存の CA Access Control グループ名
$H	（UNIX のみ）ユーザのホーム ディレクトリで始まるパラメータ
$N	数値
$O	sesudo を実行するユーザの CA Access Control での名前
$U	既存の CA Access Control ユーザ名
$e	空のエントリ。 ルールに対してパラメータが指定されていない SUDO コマンドを指定する場合に使用します。
$f	既存のファイル名
$g	既存の Windows グループ名
$h	既存のホスト名
$r	Windows 読み取りアクセス権がある既存のファイル
$u	既存の Windows ユーザ名
$w	Windows 書き込みアクセス権がある既存のファイル
$x	Windows 実行アクセス権がある既存のファイル

prohibited パラメータ値のリストをスクリプトに追加する場合は、以下のようにします。

- スクリプトと prohibited パラメータの値をセミコロンで区切り、全体を一重引用符で囲みます。 たとえば、ユーザに対して -start の使用は禁止するが、それ以外のすべてのパラメータの使用は許可する場合、以下のコマンドを入力します。

  ```
  newres SUDO scriptname comment('cmd;-start')
  ```

  ここで、*cmd* はユーザのスクリプトを表します。

  また、パラメータ値を許可せず、すべてのパラメータをデフォルトに設定する場合は、SUDO レコードを以下のように定義します。

  ```
  newres SUDO scriptname comment('cmd;*')
  ```

- 1 つのスクリプト パラメータに対して複数の prohibited 値を指定する場合は、スペース文字を区切り記号として使用します。たとえば、ユーザに対して -start および -stop の使用は禁止するが、それ以外のすべてのパラメータの使用は許可する場合は、以下のコマンドを入力します。

  ```
  newres SUDO scriptname comment('cmd;-start -stop')
  ```

- 複数のスクリプト パラメータに対して prohibited 値を指定する場合は、パイプ (|) を区切り記号として使用して、それぞれの prohibited 値セットの間を区切ります。 たとえば、スクリプトの最初のパラメータで -start および -stop を使用することを禁止し、2 番目のパラメータで既存の Windows ユーザ名（前出の変数の表を参照）を使用することを禁止する場合は、以下のコマンドを入力します。

  ```
  newres SUDO scriptname comment('cmd;-start -stop | $u')
  ```

  指定したパラメータよりスクリプトのパラメータが多い場合は、指定した最後の prohibited パラメータのセットが、残りすべてのパラメータに適用されます。

permitted パラメータ値のリストをスクリプトに追加する場合は、以下の操作を行います。

- sesudo ユーティリティがパラメータ値について以下の項目をチェックします。
  - 対応する *prohibited* 値のいずれとも一致しないこと。
  - 対応する少なくとも 1 つの *permitted* 値と一致すること。

  つまり、prohibited リストにあるパラメータ値は、permitted リストにも指定されていても、permitted にはなりません。
- *permitted* 値のリストと *prohibited* 値のリストをセミコロンで区切り、全体を一重引用符で囲みます。 prohibited 値のリストを指定しない場合でも、セミコロンは必要です。セミコロンがないと、permitted 値として指定した値が、prohibited 値として処理されます。 たとえば、スクリプトのパラメータ値として値 NAME のみを許可する場合は、以下のコマンドを入力します。

  ```
  newres SUDO scriptname comment('cmd;;NAME')
  ```
- 他のリストの指定も同様に行います。
  - 1 つのスクリプト パラメータに対して複数の permitted 値を指定する場合は、スペース文字を区切り記号として使用します。
  - 複数のスクリプト パラメータに permitted 値を指定する場合は、パイプ（|）を区切り記号として使用して、それぞれの permitted の値セットの間を区切ります。

  たとえば、2 つのパラメータがあるとします。最初のパラメータには Windows のユーザ名にしてはならない数字を指定し、2 番目のパラメータには Windows のグループ名にしてはならない英字を指定する必要がある場合は、以下のコマンドを入力します。

  ```
  newres SUDO scriptname comment('cmd;$u | $g ;$N | $A')
  ```

  スクリプトのパラメータが指定したパラメータより多い場合は、指定した最後の permitted パラメータのセットが、残りすべてのパラメータに適用されます。

したがって、comment プロパティ全体の形式は、スクリプト、パラメータごとの prohibited 値、パラメータごとの permitted 値の順になります。

```
comment('cmd; ¥
param1_prohib1 param1_prohib2 ... param1_prohibN | &yen;
param2_prohib1 param2_prohib2 ... param2_prohibN | &yen;
 ...
paramN_prohib1 paramN_prohib2 ... paramN_prohibN ; &yen;
```

```
param1_permit1 param1_permit2 ... param1_permitN | &yen;
param2_permit1 param2_permit2 ... param2_permitN |
 ...
paramN_permit1 paramN_permit2 ... paramN_permitN')
```

sesudo ユーティリティでは、ユーザが入力した各パラメータを以下の方法でチェックします。

1. パラメータ N と permitted パラメータ N が一致するかどうかを確認します （pemitted パラメータ N が存在しない場合、最後の permitted パラメータが使用されます）。
2. パラメータ N と prohibited パラメータ N が一致するかどうかを確認します（prohibited パラメータ N が存在しない場合、最後の prohibited パラメータが使用されます）。

すべてのパラメータが permitted パラメータと一致し、prohibited パラメータと一致するパラメータが存在しない場合、sesudo はコマンドを実行します。

例: ユーザに net send の実行を許可するタスクの委任をセットアップする

以下の手順では、ユーザ Takashi に net send コマンドの実行を許可して、net start コマンドの実行を許可しない方法を示します。

1. CA Access Control エンドポイント管理の［ユーザ］タブをクリックし、［権限および委任］サブタブをクリックします。

   ［権限および委任］メニュー オプションが左側に表示されます。
2. ［タスク委任］をクリックします。

   ［タスク委任］ページが表示されます。
3. ［タスクの作成］をクリックします。

   ［タスクの作成］ページが表示されます。
4. 以下のようにダイアログのフィールドに入力します。

フィールド	値
名前	NET
データ	net;start;send *
所有者	nobody
デフォルト アクセス	なし（オプションの選択なし）

フィールド	値
許可されたアクセサ	ユーザ: Takashi 許可 : 実行

［保存］をクリックします。

新しいタスクの委任（SUDO）レコードが作成されます。

5. タスクの委任ルールを確認します。
   a. Takashi でログインします。
   b. コマンドプロンプトを開き、以下のコマンドを実行します。

      ```
      sesudo -do NET start
      ```

      以下のメッセージが表示されます。

      ```
      sesudo: 'start' をパラメータ番号 1 として使用することは許可されていません。
      ```

      注: *net start* は prohibited 値として定義されたので、実行されません。
   c. 以下の値を実行します。

      ```
      sesudo -do NET send comp message
      ```

      このコマンドは実行されます。

例: 対話式アプリケーションを使用して、権限を必要とする操作を実行する権限をユーザに付与する

以下の例で示すように、ユーザは任意のスナップイン MSC モジュールを使用して、高い権限を必要とする操作を実行できます。

1. CA Access Control エンドポイント管理の［ユーザ］タブをクリックし、［権限および委任］サブタブをクリックします。

   ［権限および委任］メニュー オプションが左側に表示されます。
2. ［タスク委任］をクリックします。

   ［タスク委任］ページが表示されます。
3. ［タスクの作成］をクリックします。

   ［タスクの作成］ページが表示されます。

4. 以下のようにダイアログのフィールドに入力します。

フィールド	値
名前	サービス
データ	c:¥winnt¥system32¥mmc.exe
所有者	nobody
オプション	対話式（オプションの選択あり）
デフォルト アクセス	なし（オプションの選択なし）
許可されたアクセサ	ユーザ： Tori 許可： 実行

［保存］をクリックします。

新しいタスクの委任（SUDO）レコードが作成されます。 この［対話式］オプションは、サービスが開始されている状態のときに、ログインしたすべてのユーザが使用できるデスクトップ ユーザ インターフェースを提供します。 このインターフェースは、サービスが LocalSystem アカウントとして実行されている場合にのみ使用可能です。

5. タスクの委任ルールを確認します。
   a. Tori でログインします。
   b. コマンド プロンプトを開き、以下のコマンドを実行します。

      ```
      sesudo -do services
      ```

   c. mmc.exe が起動します。

ユーザの非アクティブ状態のチェック

ユーザの非アクティブ状態をチェックする機能を使用して、不在または会社を退職したユーザのアカウントを使用した不正なアクセスからシステムを保護します。 非アクティブ状態の日とは、ユーザがログインしていない日を指します。 ユーザ アカウントが一時停止されて、ログインできなくなるまでの、非アクティブ状態の日数を指定できます。 一時停止したアカウントは、手動で再びアクティブにする必要があります。

注: 非アクティブ状態のチェックでは、パスワード変更はアクティビティとしてカウントされます。 ユーザのパスワードが変更された場合、非アクティブ状態を理由としてそのユーザのアカウントを一時停止することはできません。

非アクティブ日数は、USER クラスまたは GROUP クラスのレコードの inactive プロパティを使用して設定できます。 GROUP クラスのレコードでの設定は、そのグループがプロファイル グループであるユーザのみに適用されます。 また、SEOS クラスの INACT プロパティを使用して、システム全体のすべてのユーザに非アクティブ状態を設定することもできます。

selang では、以下のコマンドを使用して、非アクティブ状態をグローバルに指定します。

```
setoptions inactive (numdays)
```

非アクティブ日数をグループに設定するには、以下のコマンドを使用します（この設定は、そのグループに対するシステム全体の非アクティブ設定よりも優先されます）。

```
editgrp groupName inactive (numdays)
```

非アクティブ日数をユーザに設定するには、以下のコマンドを使用します（この設定は、そのユーザに対するグループおよびシステム全体の設定よりも優先されます）。

```
editusr userName inactive (numdays)
```

一時停止しているユーザ アカウントを再びアクティブにするには、以下のコマンドを使用します。

```
editusr userName resume
```

一時停止しているプロファイル グループを再びアクティブにするには、以下のコマンドを使用します。

```
editgrp userName resume
```

システム全体レベルで非アクティブ ログイン チェックを無効にするには、以下のコマンドを使用します。

```
setoptions inactive-
```

グループに対する非アクティブ ログイン チェックを無効にするには、以下のコマンドを使用します。

```
editgrp groupName inactive-
```

ユーザに対する非アクティブ ログイン チェックを無効にするには、以下のコマンドを使用します。

```
editusr userName inactive-
```

第 7 章: ユーザ パスワードの管理

このセクションには、以下のトピックが含まれています。

パスワードおよびロックアウト ポリシーの管理

パスワードは最も一般的な認証方法ですが、パスワード保護方法には 以下のようなよく知られた問題があります。例えば、簡単なパスワードは推測されやすい、長い間同じパスワード使用したり、同じパスワードを繰り返し使用すると解読されやすい、平文のパスワードをネットワークで送信すると盗まれる危険性があるなどです。

Windows には独自のパスワード ルールとポリシーがあり、それに準拠したパスワードをユーザが使用することで、このような問題のほとんどを回避できます。 CA Access Control に追加されたルールでは、より安全なパスワードをユーザが確実に選択することができます。

CA Access Control で指定できるルールは以下のとおりです。

- 新しいパスワードは以前に使用したものと一致してはいけない。 CA Access Control に格納される使用済みのパスワードの数は、パスワード ポリシーで指定されます。
- 新しいパスワード中にユーザ名を使用することはできない。
- 新しいパスワードは変更前のパスワードを含むことはできない。
- 新しいパスワードは変更前のパスワードと一致してはならない。 CA Access Control では、大文字と小文字は区別されません。
- 新しいパスワードには、パスワード ポリシーで指定されている、英数字、特殊文字、数字、小文字、および大文字を、それぞれ最低文字数以上使用しなければならない。

- 新しいパスワードで繰り返し使用される文字の数が、パスワード ポリシーで指定されている数を超えてはいけない。
- CA Access Control の辞書で使用が禁止されている単語を、新しいパスワードに使用することはできない。辞書は以下のレジストリ サブキーの Dictionary 値で指定されています。

  HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥passwd

パスワードごとに、最長有効期限を指定する必要があります。つまり、有効期限を過ぎたパスワードは失効し、ユーザが新しいパスワードを選択する必要があります。

- パスワードごとに、最短有効期限を指定する必要があります。 最短有効期限を指定すると、ユーザが短期間に何度もパスワードを変更することを防止できます。 パスワード変更が頻繁に行われると、パスワード履歴スタックがオーバーフローし、以前使用したパスワードが再使用される場合があります。

パスワード品質チェックの設定

パスワード品質チェックを設定するには、以下の手順に従います。

1. CA Access Control エンドポイント管理の［環境設定］タブをクリックします。

   ［環境設定］メニュー オプションが左側に表示されます。

2. ［その他］セクションのオプションで［クラスのアクティブ化］をクリックします。

   ［クラスのアクティブ化］ページが表示されます。

3. ［ユーザ識別コントロール］セクションで［PASSWORD］を選択して、［保存］をクリックします。

   これで、パスワード品質チェックがアクティブになります。

4. ［ポリシー］セクションのオプションで、［ユーザ パスワード ポリシー］をクリックします。

   ［ユーザ パスワード ポリシー］ページが表示されます。

5. パスワードのチェックに使用するルールを定義し、［保存］をクリックします。

   パスワードのチェックに定義したルールは、パスワードが変更されたときに適用されます。

6. （UNIX のみ）sepass ユーティリティを使用して、新しいパスワードを更新します。

   注: sepass ユーティリティの詳細については、「リファレンス ガイド」を参照してください。

例: パスワード チェック ルールを定義する

以下の selang コマンドは、パスワード品質チェックをアクティブにし、以下の最小文字数を適用するパスワード ルールを定義します。

- 英数字： 6 文字
- 小文字： 3 文字
- 数字： 2 文字

```
setoptions class+ (PASSWORD)
setoptions password(rules(alphanum("6") lowercase("3") numeric("2")))
```

注: setoptions コマンドの形式の詳細については、「リファレンス ガイド」を参照してください。

エラー メッセージの解決

Windows システム上でユーザのパスワードを設定している場合、以下のメッセージが表示されることがあります。

パスワードが必要な長さよりも短い。

このエラーは、パスワードがポリシー要件を満たしていないことを意味します。 このエラーの原因は、以下のいずれかです。

- パスワードが必要な長さよりも短いか、または長い。
- パスワードが最近使用されており、Windows NT Change History フィールドに存在する。

- パスワードに完全に一意の文字が含まれていない。
- パスワードが他のパスワード ポリシー要件（CA Access Control パスワード ポリシーで設定された要件など）を満たしていない。

このエラーを回避するには、該当するすべての要件を満たすパスワードを設定するようにしてください。

第 8 章: 監視と監査

このセクションには、以下のトピックが含まれています。

セキュリティ監査担当者

セキュリティ監査担当者およびシステム管理者の最も重要な仕事の 1 つは、システムでのアクティビティを監査または監視して、疑わしいアクティビティや不正なアクティビティを検出することです。 セキュリティで保護された環境において、セキュリティ監査は重要な役割を果たします。CA Access Control には、以下のセキュリティ監査機能があります。

- システムにアクセスしたユーザ、アクセスされたリソース、リソースにアクセスした方法（ファイルの読み取りなど）、およびその日時を提供する機能
- セキュリティ違反の試みがあったときは、その試みが失敗に終わった場合でも、適切なユーザに通知および警告する機能
- セキュリティ ルールに対して行われた変更の内容と、変更を行ったユーザを表示する機能
- アクセス ルールを適用する前に、ルールの影響をテストする機能

CA Access Control での監査は、実社会での監査をモデルにしています。つまり、セキュリティ監査担当者は、システム管理者およびセキュリティ管理者とは独立して任務を実行します。ただし、運用する環境に最も適したモデルがほかにある場合は、この実装を変更できます。

セキュリティ監査担当者は、AUDITOR 属性が割り当てられているユーザです。 セキュリティ監査担当者として定義されているユーザは、ユーザおよびリソースに割り当てられた監査ルールの変更などの監査タスクを実行できます。 また、このユーザには ADMIN 属性がなくても CA Access Control の監査ユーティリティを使用する権限が与えられています。

イベントのインターセプト

CA Access Control は、以下の 2 つの条件が満たされた場合にイベントをインターセプトします。

- 適切なクラスがアクティブな場合。
- このイベントを予期するルールがデータベースに存在する場合。

たとえば、c:¥data¥payroll に存在するファイルへのすべてのファイル アクセスを監査するには、以下の汎用的なルールを使用できます。

```
newres FILE c:¥data¥payroll¥*
```

また、FILE クラスがアクティブ（デフォルト）であることを確認する必要があります。

インターセプトされるイベントのタイプ

CA Access Control は、以下の 2 つのタイプのイベントをインターセプトします。

- インターセプト イベント

  インターセプト イベントからの情報はプロセスの一部としてキャッシュされ、将来監査イベントによって使用されます。
- イベント監査

インターセプト モード

CA Access Control は、インターセプト モードに基づいて、インターセプト、権限チェック、アクセス要求イベントの監査レコードのログ記録を行います。 CA Access Control には、以下の 3 つのインターセプト モードがあります。

- Full Enforcement モード
- Audit Only モード
- No Interception モード

注: 警告モード (118以下のページで定義参照：)はインターセプト モードではありません。このモードは、Full Enforcement モードでのみ機能し、実装時の短期的な使用を目的としています。

Audit Only モード

Audit Only モードは、アクセス ルールをチェックしたり適用したりせずに、インターセプトされたすべてのイベントを記録します。 このモードは、コンプライアンス要件または規則に関するデータを収集するために使用します。 Audit Only モードでは、CA Access Control はイベントを監査し、監査イベントを記録しますが、認証要求を処理せず、ルールも適用しません。 その結果、CA Access Control は、インターセプトするすべてのアクセス要求を許可します。 これは、すべてのイベントについて、記録される監査ログが *P* （許可）であることを意味します。

Audit Only モードには、以下の制限事項が適用されます。

- Unicenter には監査レコードは送信されません。

  Audit Only モードでは、すべてのイベントが許可されます（*P*）。 許可されたイベントは、Unicenter には送信されません。

- リソースおよびユーザの監査プロパティは、*考慮されません*。

  Audit Only モードでは、リソース固有の設定であれ、ユーザ固有の設定であれ、*インターセプトされたすべてのイベント*が記録されます。

Audit Only モードのセットアップ

Audit Only モードは、アクセス ルールをチェックしたり適用したりせずに、インターセプトされたすべてのイベントを記録します。 このモードは、コンプライアンス要件または規則に関するデータを収集するために使用します。

Audit Only モードをセットアップするには、SeOSD¥GeneralInterceptionMode CA Access Control レジストリ エントリを 1 に設定します。

重要: Audit Only モードを使用する場合は、監査ログを書き込むための十分なディスク領域があること、および監査ログのサイズ制限の設定が十分な大きさであることを確認してください。監査ログ バックアップ (P. 157) のオプションについても、考慮する必要があります。

警告モード

*警告モード*とは、リソースに適用できるプロパティであると同時に、クラスに適用できるオプションです。 警告モードがリソースまたはクラスに適用されている場合にアクセス ルールのアクセス違反が発生すると、CA Access Control は、リターン コード W を付けて監査ログにエントリを記録しますが、リソースへのアクセスは許可されます。 クラスが警告モードの場合は、そのクラス内のすべてのリソースが警告モードになります。

警告モードは、CA Access Control が Full Enforcement モードの場合のみ有効です。

注: Full Enforcement モードは、CA Access Control for UNIX がサポートする唯一のモードです。 CA Access Control for Windows では、Audit Only モードもサポートしています。

警告モードは、アクセス ポリシーを導入または変更する場合に使用できます。 警告モードを使用する場合は、ポリシーを有効にする前に、監査ログで対象となるポリシーの結果を事前に確認することができます。 監査ログを表示するには、seaudit コマンドを使用します。

クラスにプロパティ *warning* がある場合は、クラスを警告モードに設定できます。 リソース グループまたはクラスが警告モードの場合に、アクセス ルール違反が発生すると、CA Access Control は、アクセスを許可し、（リソース グループまたはクラスではなく）リソースを参照するエントリを監査ログに記録します。

リソースの警告モードの設定とクラスの警告モードの設定は独立しています。リソースを警告モードに設定した場合、そのリソースが属するクラスから警告モードを削除したとしても、そのリソースは警告モードのままとなります。

注: リソースまたはクラスを警告モードに設定できるのは、リソースまたはクラスにプロパティ *warning* がある場合だけです。必ずしもすべてのリソースまたはクラスにこのプロパティがあるわけではありません。

詳細情報:

Audit Only モード (P. 117)

リソースの警告モードの設定

リソースを警告モードに設定することで、アクセス ルールを適用することなく、アクセス ルールの効果を監視できます。

注: 個々のリソースを警告モードに設定するだけでなく、クラスを警告モードに設定 (P. 121)することもできます。

リソースを警告モードに設定するには、以下の手順に従います。

1. CA Access Control エンドポイント管理で、警告モードに設定するリソースを編集します。

   適切な［変更］ページが表示されます。

2. ［監査］タブをクリックします。

   リソースに対する［監査モード］ページが表示されます。

3. ［警告モード］を選択し、［保存］をクリックします。

   変更したリソースが警告モードになります。

注: 警告モードでは、アクセス ルール違反が発生した場合、アクセスは許可されますが、CA Access Control は必ず警告レコードを監査ログに記録します。このため、リソースの audit プロパティを設定する必要はありません。

sereport ユーティリティ（レポート番号 6）を使用すると、警告モードであるすべてのリソースが表示されます。

例: ファイルを警告モードに設定する

以下の selang の例では、ファイル c:¥myfile を警告モードに設定します。

```
chres FIlE c:¥myfile warning
```

例: ファイルから警告モードをクリアする

以下の selang の例では、ファイル c:¥myfile の警告モードを無効にします。

```
chres FIlE c:¥myfile warning-
```

myfile の警告モードは無効になるので、CA Access Control は myfile に対するアクセス ルールを適用します。

例: 端末を警告モードに設定する

以下の selang の例では、端末 myterminal を警告モードに設定します。

```
chres terminal myterminal warning
```

この場合、CA Access Control は権限のあるユーザによる端末 myterminal からのアクセスを許可しますが、その端末からのアクセスが通常拒否されるユーザについては監査レコードをログに記録します。

クラスを警告モードに設定する

個々のレコードを警告モードに設定するのではなく、クラス内のすべてのレコードを警告モードに設定することができます。 警告モードを使用することで、アクセス ルールを適用することなく、アクセス ルールの効果を監視できます。

クラスを警告モードに設定する方法

1. CA Access Control エンドポイント管理 内で、以下の操作を実行します。
   a. ［設定］をクリックします。
   b. ［クラスのアクティブ化］をクリックします。

   ［クラスのアクティブ化］ページが表示されます。
2. ［警告］モードに設定するクラスの［警告］列のチェック ボックスをオンにします。
3. ［Save］をクリックします。

   確認メッセージが表示され、CA Access Control オプションが正常に更新されたことが通知されます。

警告モードが指定されたリソースの確認

CA Access Control を実装している場合は、警告モードを一時的な手段として使用する必要があります。 ユーザが必要とするリソースへの必要なアクセス権を持っていることを確認したら、警告モードをオフにします。CA Access Control は関連するルールの適用を開始します。

警告モードであるリソースを確認するために、警告モードであるすべてのリソースを示すレポートを作成できます。

レポートを作成するには、以下のコマンドを入力します。

```
sereport -f pathname.html -r 6
```

CA Access Control によってレポートが作成されます。

注: sereport ユーティリティの詳細については、「リファレンス ガイド」を参照してください。

警告モードであるクラスの確認

CA Access Control を実装している場合は、警告モードを一時的な手段として使用する必要があります。 ユーザが必要とするリソースへの必要なアクセス権を持っていることを確認したら、警告モードをオフにします。CA Access Control は関連するルールの適用を開始します。

警告モードであるクラスを確認するために、CA Access Control でこのデータを表示することができます。

このデータを表示するには、以下の selang コマンドを入力します。

```
setoptions cwarnlist
```

警告モードが指定されたクラスを示す表が表示されます。

注: setoptions の詳細については、「*selang リファレンス ガイド*」を参照してください。

システム メンテナンスの実行方法

システムをアップグレードしたり、新しいアプリケーションをインストールするために、特定の時間にシステム メンテナンスを実行しなければならない場合があります。 システム メンテナンス中は、CA Access Control ルールを警告モードに設定する必要があります。 メンテナンスが必要なリソースへのユーザ アクセスに影響しないことが確認できたら、警告モードをオフにする必要があります。そうすると、CA Access Control は関連ルールの適用を開始します。

システム メンテナンスの実行時に警告モードを使用するには、以下の操作を行います。

1. メンテナンスを開始する前に、以下の selang ルールを使用して、該当するクラスを警告モードに設定します。

   ```
   setoptions class(NAME) flags(W)
   ```

2. メンテナンスを実行します。
3. メンテナンスの実行後、seretrust ユーティリティを実行します。

   seretrust ユーティリティは selang コマンドを生成します。このコマンドは、データベース内で定義されているプログラムおよび保護対象ファイルを再度 trusted 状態にする場合に必要となります。
4. selang コマンドを実行して、データベース内で定義されたプログラムを再度信頼します。
5. 以下の selang ルールを使用して、ポリシー適用を有効にするために、クラスから警告モードを削除します。

   ```
   setoptions class(NAME) flags-(W)
   ```

6. CA Access Control 監査ログ ファイルを確認します。

   監査ログには、メンテナンスによる影響を受けたリソースの警告が含まれています。

注: seretrust ユーティリティの詳細については、「*リファレンス ガイド*」を参照してください。

Access Control のアクティビティの監視

CA Access Control のトレースは、CA Access Control によって実行されるすべてのアクションを確認できるリアルタイム ログです。 トレース レコードは *ACInstallDir*¥log¥seosd.trace に蓄積されます（ここで、*ACInstallDir* は CA Access Control のインストール ディレクトリです）。

または、以下のレジストリ サブキーで *trace_file* 値として指定されたファイルに蓄積されます。

```
HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥SeOSD¥
```

トレース ファイルのレコードはフィルタ処理できますが、トレース機能は本来セキュリティ監査ではなくシステム監視を目的として設計されたメカニズムです。

デフォルトでは、CA Access Control の初期化時にのみトレース メッセージが生成されます。 CA Access Control の初期化が終わると、トレース メカニズムは停止し、トレース メッセージは生成されません。

トレース レコード フィルタ

CA Access Control では、以下の 2 つのタイプのトレース レコードを生成します。

- ユーザ トレース レコード - ユーザが完了したレコードを記録します。
  例： ユーザ 1 がファイル c:¥tmp¥tmp.exe にアクセスしました。
- 一般トレース レコード - システムが完了したアクションを記録します。
  例： Watchdog がプログラムを untrusted に設定しました。

トレース レコードは seos.trace ファイルに書き込まれ、trcfilter.ini ファイルを使用してフィルタできます。

ユーザをトレース可能に設定した場合、そのユーザに関するトレース レコードが書き込まれるたびに、対応する監査レコードが seos.audit ファイルに書き込まれます。 監査レコードは audit.cfg ファイルによってフィルタされます。

注: トレース イベントによって生成された監査レコードはキャッシュされず、Full Enformacement フローが常に適用されます。

以下の selang コマンドで、ユーザをトレース可能に設定します。

```
editusr userName audit(trace)
```

トレース レコードまたは監査レコードを表示するには、seaudit ユーティリティを使用します。

トレース レコードのフィルタ処理

トレース フィルタ ファイルを使用すると、特定の種類のアクティビティがトレース ファイルに書き込まれないように指定できます。 トレース フィルタ ファイルは、*trace_filter* 値を使用して、以下のレジストリ キーで指定します。

```
HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥SeOSD
```

デフォルト値は *ACInstallDir*¥log¥trcfilter.ini です（ここで、*ACInstallDir* は CA Access Control のインストール ディレクトリです）。

重要: CA Access Control のインストール時に、*seosd.trace* という 1 行が書き込まれたトレース フィルタ ファイルが作成されます。このレコードは、絶対に削除しないでください。

トレース フィルタ ファイル内の各行は、トレースする必要がないアクセスまたはアクティビティを表します。 たとえば、Microsoft Word へのユーザ アクセスをトレース対象外にするには、トレース フィルタ ファイルに以下の行を追加します。

```
*winword.exe*
```

CA Access Control の監査対象

セキュリティ監査では、CA Access Control は、データベースに定義されている監査ルールと、監査の適用モードに従って、インターセプトされたイベントの監査レコードを保持します。 監査ログのレコードは、これらの監査ルールに従って蓄積されます。

完全監査では、以下のすべてのインターセプト イベントの監査レコードが提供されます。

- ファイル アクセス（FILE クラス）
- プログラム実行（PROGRAM クラス）
- レジストリ アクセス（REGKEY および REGVAL クラス）
- 代理実行コントロール（SURROGATE クラス）
- ネットワーク コントロール（CONNECT、TCP、HOST、GHOST、HOSTNET および HOSTNP クラス）

- ログイン（TERMINAL クラス）

  注: インターセプトされたログイン イベントはキャッシュされません。これらのイベントは、インターセプト イベントの監査プロセスに従って処理されます。

- サービス保護（WINSERVICE クラス）
- パスワード確認失敗（PASSWORD クラス）
- プロセス終了（PROCESS クラス）

イベントをログに記録するかどうかは、CA Access Control のインターセプト モードによって決まります。

ログイン インターセプトの制限

Windows のログイン インターセプトは、CA Access Control のサブ認証方式でのみサポートされています。

カーネルを介してログイン インターセプトを設定することはできません。結果として、以下の点を考慮する必要があります。

- サブ認証コンポーネントはドメイン コントローラ（DC）レベルで動作するので、ユーザのログイン イベントを認証（および CA Access Control のサブ認証モジュールをトリガ）する DC は OS に応じて異なります。Windows ドメイン環境では、CA Access Control は DC ごとにインストールする必要があります。
- Windows ドメイン環境で動作する場合、CA Access Control のログイン ポリシー（TERMINAL ルール）を DC 上に配置する必要があります。ターゲット サーバ上に配置する必要はありません。

  たとえば、Windows ドメインに参加しているが DC ではないファイル サーバで、ドメイン ユーザのログイン イベントを保護または監査する必要がある場合、CA Access Control のログイン ポリシーは、ターゲット ファイル サーバ上ではなく、DC 上で定義する必要があります。 これは、ドメイン ユーザが共有ファイル ディレクトリにアクセスしたときに、ファイル サーバ上ではなく、DC 上でログイン認証が発生することに起因します。

- 複数の DC が存在する場合、CA Access Control のログイン認証はいずれかの DC 上で処理されます。この結果、CA Access Control のログイン ポリシーをすべての DC 間で同期することをお勧めしています。

  具体的な実装方法としては、Policy Model メカニズム（すべての DC が PMDB のサブスクライバに該当）を使用する方法と、すべての DC をホストグループに追加し、拡張ポリシー管理に基づいて共通のポリシーをデプロイする方法が挙げられます。

- ログイン イベントに対応するユーザ プロパティは、実行時（イベント認証中）に更新される場合があります。 該当するプロパティについては、同期外れが発生します。これは、ログイン認証がいずれか 1 つの DC でのみ実行されることに起因します。 上記に該当するプロパティは *Gracelogins*、*Last accessed*、および *Last access time* です。

  つまり、例を挙げると、CA Access Control のサブ認証はすべての DC ではなく、いずれか 1 つの DC でのみ実行されるので、ユーザ プロパティ *Last access time* の値は DC 間で異なることになります。

- ローカル ユーザ（ドメイン ユーザ以外）のログイン イベントを適用するには、ローカル ユーザのアクセス先のローカル コンピュータに CA Access Control をインストールする必要があります。 これは、ローカル コンピュータがドメイン コンピュータとして使用されるためです（ドメインがローカル コンピュータ）。

- リモート デスクトップ プロトコル（RDP）/ターミナル サービス ログイン イベントは、以前の CA Access Control バージョンと同様にターゲット サーバに対して適用されます。ただし、RDP ログイン イベントの場合、CA Access Control のログイン ポリシーはターゲット サーバ上で定義する必要があります。

Full Enforcement モードでの CA Access Control 監査の対象

Full Enforcement モード（通常操作）では、CA Access Control は、以下のようにイベントをログに記録します。

- インターセプトされたリソースの警告モードが *オフ* の場合、CA Access Control は、リソースまたはユーザの *audit* プロパティに基づいてルールを適用し、イベントをログに記録します。

audit プロパティ	ログに記録されるイベント
ALL	すべて

audit プロパティ	ログに記録されるイベント
SUCCESS	許可されたアクセス
FAIL	拒否されたアクセス

- インターセプトされたリソースの警告モードがオンの場合、アクセス ルールに違反するアクセス要求があると、レコードが監査ログに書き込まれます（ルールが適用されていると、要求は失敗します）。 この監査レコードには、警告モードが有効なため違反が許可されたことが記述されます。

  このモードでは、ルールは適用されません。

Audit Only モードでの CA Access Control 監査の対象

Audit Only モードでは、CA Access Control は認証の要求を処理せず、ルールも適用しません。 アクセサのすべてのインターセプト ログイン イベントおよび CA Access Control によって保護されているリソースのすべてのインターセプト イベントは、アクセスが失敗したか成功したかに関係なく、ログに記録されます。

CA Access Control が監査ログに書き込むイベントを変更する方法

CA Access Control が監査ログに書き込むイベントは、以下の 2 つの方法で変更することができます。

- リソースまたはアクセサの AUDIT プロパティを使用して CA Access Control が監査ログに書き込む監査イベントを定義します。

  注: GROUP または XGROUP の AUDIT プロパティを使用して、グループのすべてのメンバに監査プロパティを設定することができます。 ただし、ユーザの監査モードが USER レコード、XUSER レコード、またはプロファイル グループに定義されている場合は、AUDIT プロパティを使用してグループ メンバに監査モードを設定することはできません。

- 監査構成ファイル、audit.cfg を使用して CA Access Control が監査ログに送信するイベントをフィルタします。 audit.cfg ファイルを使用して監査ログにイベントを追加することはできません。

監査レコードの数を減らすために、ログ ファイルに書き込まれる連続した監査イベントを制御することもできます。 このカスタマイズの基準は、連続して一致する監査イベント（つまり、同じプロセス ID、スレッド ID、ルール ID、およびアクセス マスクを持つリソースへのアクセス）間の時間間隔です。 この時間間隔（秒単位）を設定するには、AuditRefreshPeriod レジストリ エントリの値を設定します。 デフォルトでは、AuditRefreshPeriod は 0（ゼロ）に設定されます。つまり、すべてのイベントがログ ファイルに記録されます。

監査ルールの設定

CA Access Control では、セキュリティ監査のために、データベースに定義されている監査ルールに基づいて、アクセス拒否およびアクセス許可のイベントに関する監査レコードが保存されます。

すべてのアクセサおよびリソースに AUDIT プロパティがあり、このプロパティでは以下の 1 つ以上の値を設定できます。

FAIL

アクセサによるリソースへの失敗したアクセスをログに記録します。

SUCCESS

アクセサによるリソースへの成功したアクセスをログに記録します。

LOGINFAIL

アクセサによる失敗したすべてのログインをログに記録します（この値はリソースには適用されません）。

LOGINSUCCESS

アクセサによる成功したすべてのログインをログに記録します（この値はリソースには適用されません）。

ALL

アクセサの FAIL、SUCCESS、LOGINFAIL、および LOGINSUCCESS、またはリソースの FAIL および SUCCESS と同じ情報をログに記録します。

NONE

アクセサまたはリソースに関して、ログに何も記録しません。

データベースにアクセサまたはリソース レコードを作成または更新する場合はいつでも、AUDIT プロパティを指定できます。 また、ログに記録されたイベントを電子メールで通知するかどうか、通知する場合は誰に通知するかを指定することもできます。

監査ログのレコードは、これらの監査ルールに従って蓄積されます。 イベントをログに記録するかどうかは、以下のルールに基づいて決定されます。

- リソースまたはアクセサに AUDIT (ALL) が割り当てられている場合は、そのアクセサのすべてのログイン イベント、および CA Access Control によって保護されているリソースに関するすべてのイベントが、アクセスが失敗したか成功したかにかかわらず、ログに記録される。
- CA Access Control によって保護されているリソースへのアクセスが成功し、アクセサまたはリソースに AUDIT（SUCCESS）が割り当てられている場合は、イベントがログに記録される。
- CA Access Control によって保護されているリソースへのアクセスが失敗し、アクセサまたはリソースに AUDIT（FAIL）が割り当てられている場合は、イベントがログに記録される。

さらに、ユーザをトレース可能に設定した場合、そのユーザにトレース レコードが書き込まれるたびに、対応する監査レコードは監査ログに書き込まれます。

CA Access Control が監査ログに書き込む監査イベントの定義

CA Access Control では、アクセスの成功と失敗を監査ログに書き込みます。CA Access Control が監査ログに書き込むアクセス イベントを定義するには、監査対象のリソースまたはアクセサの AUDIT プロパティの値を変更します。 また、CA Access Control では、この方法で、すべてのトレース イベントを監査ログに記録するように指定することもできます。

AUDIT プロパティを使用して CA Access Control が監査ログに書き込む監査イベントを指定します。 selang または CA Access Control エンドポイント管理 を使用して、以下のようにしてリソースおよびアクセサに AUDIT プロパティを設定します。

AUDIT の値	CA Access Control がログに記録する内容	適用可能なオブジェクト
FAIL	アクセスの失敗	ユーザおよびリソース
SUCCESS	アクセスの成功	ユーザおよびリソース
LOGINFAIL	ログインの失敗	ユーザ
LOGINSUCCESS	ログインの成功	ユーザ
ALL	FAIL、SUCCESS、LOGINFAIL、LOGINSUCCESS、および INTERACTIVE に相当	ユーザおよびリソース
TRACE	ALL およびすべてのシステム イベントに相当	ユーザ
INTERACTIVE	UNIX コンピュータ上のユーザ セッション	ユーザ
NONE	ログへの記録を行わない	ユーザおよびリソース

注: ユーザの監査プロパティが設定されていない場合、グループまたはプロファイル グループの AUDIT 値により、CA Access Control でユーザに対して使用される監査モードに影響が及ぶ可能性があります。

CA Access Control がユーザの監査モードを決定する方法

ユーザの監査モードでは、CA Access Control がそのユーザの監査ログに送信する監査イベントを指定します。以下のプロセスでは、CA Access Control がユーザの監査モードを決定する方法について説明します。

1. CA Access Control は、USER クラスまたは XUSER クラスのユーザのレコードに AUDIT プロパティの値があるかどうかをチェックします。

   ユーザのレコードに AUDIT プロパティの値がある場合、CA Access Control はその値をユーザの監査モードとして使用します。

2. CA Access Control は、ユーザがプロファイル グループに割り当てられているかどうかをチェックします。ユーザがプロファイル グループに割り当てられている場合、CA Access Control は GROUP クラスにある、そのプロファイル グループのレコードに AUDIT プロパティの値が含まれているかどうかをチェックします。

   ユーザがプロファイル グループに割り当てられていて、プロファイル グループのレコードに AUDIT プロパティの値がある場合、CA Access Control はその値をユーザの監査モードとして使用します。

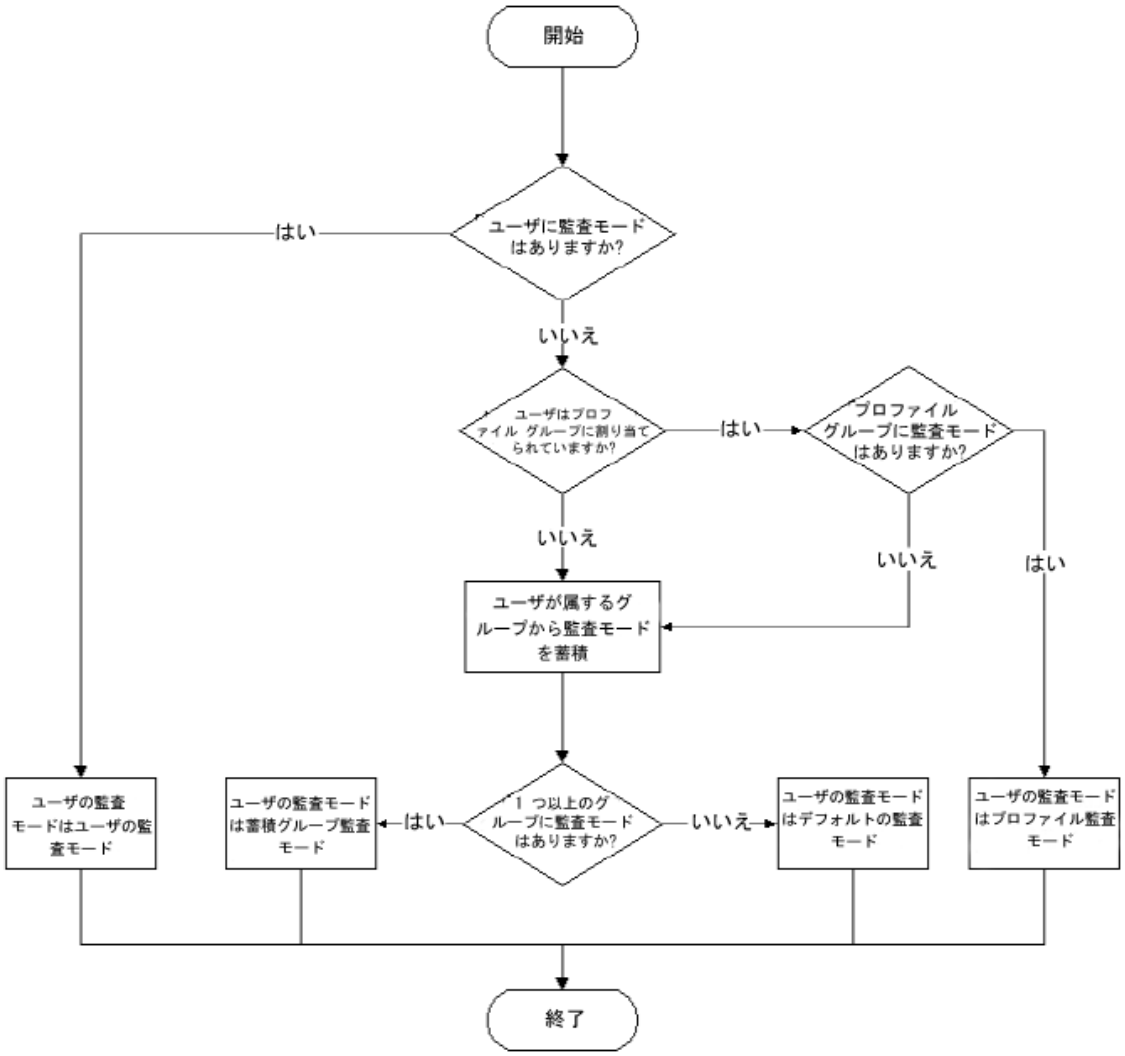
3. CA Access Control は、ユーザがグループのメンバであるかどうかを確認します。 ユーザがグループ メンバである場合、CA Access Control は GROUP または XGROUP クラスのグループのレコードに AUDIT プロパティの値があるかどうかをチェックします。

   ユーザがグループのメンバで、グループのレコードに AUDIT プロパティの値がある場合、CA Access Control はその値をユーザの監査モードとして使用します。 もしこのユーザがグループのメンバではないか、あるいはこのグループのレコードが AUDIT プロパティの値を持たない場合、 CA Access Control はシステム全体の監査モードをユーザに割り当てます。

   注: ユーザが複数のグループのメンバであり、グループごとに異なる監査モードがある場合、ユーザの監査モードは蓄積されます。 ユーザの監査モードは、メンバであるグループのすべての監査モードの合計です。

注: CA Access Control がグループの AUDIT プロパティの値を使用してユーザの監査モードを決定し、ユーザのログイン中にグループの監査モードを変更した場合は、ログイン中のユーザの監査モードも変更されます。 グループ監査モードの変更を有効にするためにユーザがログオフする必要はありません。

以下の図では、CA Access Control がユーザの監査モードを決定する方法について説明します。



例: グループ別監査

ユーザの Jan は、グループ A およびグループ B のメンバです。 グループ A の監査モードは FAIL であり、グループ B の監査モードは SUCCESS です。 Jan は両方のグループのメンバであるため、Jan には FAIL および SUCCESS の蓄積された監査モードがあります。

詳細情報:

CA Access Control がプロファイル グループを使用してユーザ　プロパティを決定する方法 (P. 51)

ユーザおよびエンタープライズ ユーザのデフォルトの監査モード

ユーザ（USER オブジェクト）を作成すると、CA Access Control によってデフォルトの AUDIT_MODE がオブジェクトに割り当てられます。AUDIT_MODE プロパティのデフォルト値は「Failure」、「SuccessLogin」、「SuccessFailure」です。

エンタープライズ ユーザ（XUSER オブジェクト）を作成すると、デフォルトで CA Access Control によってデフォルトの AUDIT_MODE 値がオブジェクトに割り当てられません。

注: （UNIX）USER オブジェクトの AUDIT_MODE プロパティのデフォルト値を変更するには、lang.ini ファイルの［newusr］セクションで、DefaultAudit の値を編集します。

一部のユーザのデフォルト監査値の変更

r12.0 SP1 CR1 より前は、以下のアクセサのデフォルト監査モードは「なし」でした。

- 対応する USER クラス レコードで AUDIT 値が定義されていないユーザ、および AUDIT 値が定義されているプロファイル グループに関連付けられていないユーザ
- データベースで定義されていないすべてのユーザ（_undefined ユーザ レコードによって表される）

  注: エンタープライズ ユーザを使用した場合、CA Access Control がユーザを未定義として認識することはなくなります。この場合、_undefined ユーザのプロパティは考慮されません。

r12.0 SP1 CR1 から、これらのアクセサのデフォルト監査モードは「Failure」、「LoginSuccess」、および「LoginFailure」になりました。 以前の動作を保持するには、これらのユーザの AUDIT プロパティの値を「なし」に設定してください。

GROUP レコードの AUDIT プロパティ値の変更

GROUP レコードがある場合、それには以下の 2 つの機能があります。

- 1 つのユーザ セットの監査ポリシーを定義するプロファイル
- 2 つ目のユーザ セットのコンテナ

r12.0 SP1 CR1 以降、GROUP レコードは 2 つ目のユーザ セットの監査ポリシーも定義するようになりました。 動作の変更によって生じる可能性のある問題を回避するために、2 つ目のユーザ セット用に別の GROUP を作成してください。

Windows での監査ポリシーの設定

アクセサおよびリソースに関するアクセス ルールを設定するだけでなく、監査ログに書き込む Windows イベントを指定できます。 このような監査ポリシーは、組織全体に対して、グループ単位、プロファイル グループ単位、またはユーザ単位で指定できます。

例: プロファイル グループのすべてのメンバ向けの監査ポリシーを設定する

以下の例では、プロファイル グループに属するすべてのユーザ向けの監査ポリシーを設定する方法を示します。

1. 必要な監査モードで、新しいプロファイル グループを作成します。以下に例を示します。

   ```
   newgrp profileGroup audit(failure) owner(nobody)
   ```

2. 新しいユーザを作成し、作成したプロファイル グループに追加します。以下に例を示します。

   ```
   newusr user1 profile(profileGroup) owner(nobody)
   ```

3. ユーザの監査設定を削除します。 以下に例を示します。

   ```
   chusr user1 audit-
   ```

この設定が有効かどうかを確認できます。

1. 新しいユーザでログインします。

   ```
   runas /user:user1 cmd.exe
   ```

2. user1 のコマンド プロンプト ウィンドウに、以下のコマンドを入力します。

   ```
   secons -whoami
   ```

   このコマンドでは、user1 の認証に使用され、user1 の ACEE に保持されている情報が表示されます。

   ```
   ACEE 監査モード: 失敗; プロファイル グループ定義から由来
   ```

   このメッセージにより、監査ポリシーは、ユーザが属するプロファイル グループから派生していることが確認されます。

例: グループ メンバの監査ポリシーを設定する

この例では、Forward Inc という名前の架空の会社が CA Access Control を使用して /production ディレクトリ内のすべてのファイルを保護します。 /production ディレクトリにはネイティブ環境で完全なアクセス権限があります。

Forward Inc は、/production ディレクトリへのすべてのアクセス試行を拒否し、監査することを検討しています。 ただし、Forward Inc は、開発者の /production ディレクトリへの読み取りアクセスは許可します。 このアクセスは監査されません。 開発者による /production ディレクトリへの書き込み試行は、拒否されて監査されます。

開発者は、/production ディレクトリへの完全なアクセス権を要求できます。 Forward Inc は、完全なアクセス権を持つユーザが /production ディレクトリ内で実行するすべてのアクティビティを監査します。

以下のプロセスでは、Forward Inc が前のシナリオを実装するために実行する手順について説明します。

1. ネイティブ環境に「Developers」という名前のグループを作成します。 すべての開発者をこのグループに追加します。
2. ネイティブ環境に「Dev_Access_All」という名前のグループを作成します。 ユーザはこのグループに追加しないでください。
3. 以下のようにして、/production ディレクトリに対する包括的なアクセス ルールを定義します。

   ```
   authorize FILE /production/* access(none) uid(*)
   ```

   このルールは、デフォルト アクセスを「なし」に設定します。
4. 以下のようにして、/production ディレクトリに対する包括的な監査ルールを定義します。

   ```
   editres FILE /production/* audit(failure)
   ```

   このルールは、/production ディレクトリへのアクセスに失敗したすべての試行を監査します。

5. 以下のようにして、Developers グループにアクセス ルールを定義します。

   ```
   authorize FILE /production/* access(read) xgid(Developers)
   ```

   このルールは Developers グループのメンバに /production ディレクトリへの読み取りアクセスを許可します。

   注: 手順 4 で設定したルールによって、Development グループのメンバを含め、任意のユーザによるすべての失敗したアクセス試行を CA Access Control が確実に監査できるようになります。

6. 以下のようにして、Dev_Access_All グループにアクセス ルールを定義します。

   ```
   authorize FILE /production/* access(all) xgid(Dev_Access_All)
   ```

   このルールによって、Dev_Access_All グループのメンバに /production ディレクトリへの完全なアクセス権が許可されます。

7. 以下のようにして、Dev_Access_All グループに監査ルールを定義します。

   ```
   chxgrp Dev_Access_All audit(all)
   ```

   このルールは、Dev_Access_All グループのメンバが実行するすべてのアクションを監査します。

8. Developers グループのメンバに /production ディレクトリへの完全なアクセス権が必要な場合は、ユーザをネイティブ環境内の Dev_Access_All グループに追加します。

   ユーザには /production ディレクトリへの完全なアクセス権があり、CA Access Control は、ユーザが実行するすべてのアクションを監査します。

   注: グループ メンバシップの変更を有効にするには、ユーザが新しいログオン セッションを開始する必要があります。

9. ユーザがタスクを /production ディレクトリで完了したら、ユーザをネイティブ環境の Dev_Access_All グループから削除します。

   これで、ユーザは /production ディレクトリへの読み取りアクセスを得ることができます。CA Access Control は、ユーザによる /production ディレクトリでのその他すべてのアクセス試行を拒否して監査します。

   注: グループ メンバシップの変更を有効にするには、ユーザが新しいログオン セッションを開始する必要があります。

監査プロセス

監査要件に合わせて CA Access Control を設定するには、最初に監査のしくみを理解しておく必要があります。 監査によって、CA Access Control がイベントしたアクセス要求（イベント）を追跡できます。 このデータを使用して、準拠すべき要件への適合、セキュリティ要件に適合するためのアクセス ルールの分析および改良、アクセス要求の監視を行うことができます。

CA Access Control が監査イベントをログに記録する手順は、インターセプトするイベントの種類によって異なります。

- インターセプト イベント（P. 141）

  注: インターセプトされたログイン イベント（TERMINAL クラス）、およびユーザ トレースによって生成された監査レコードはキャッシュされません。これらのイベントは、インターセプト イベントの監査プロセスに従って処理されます。

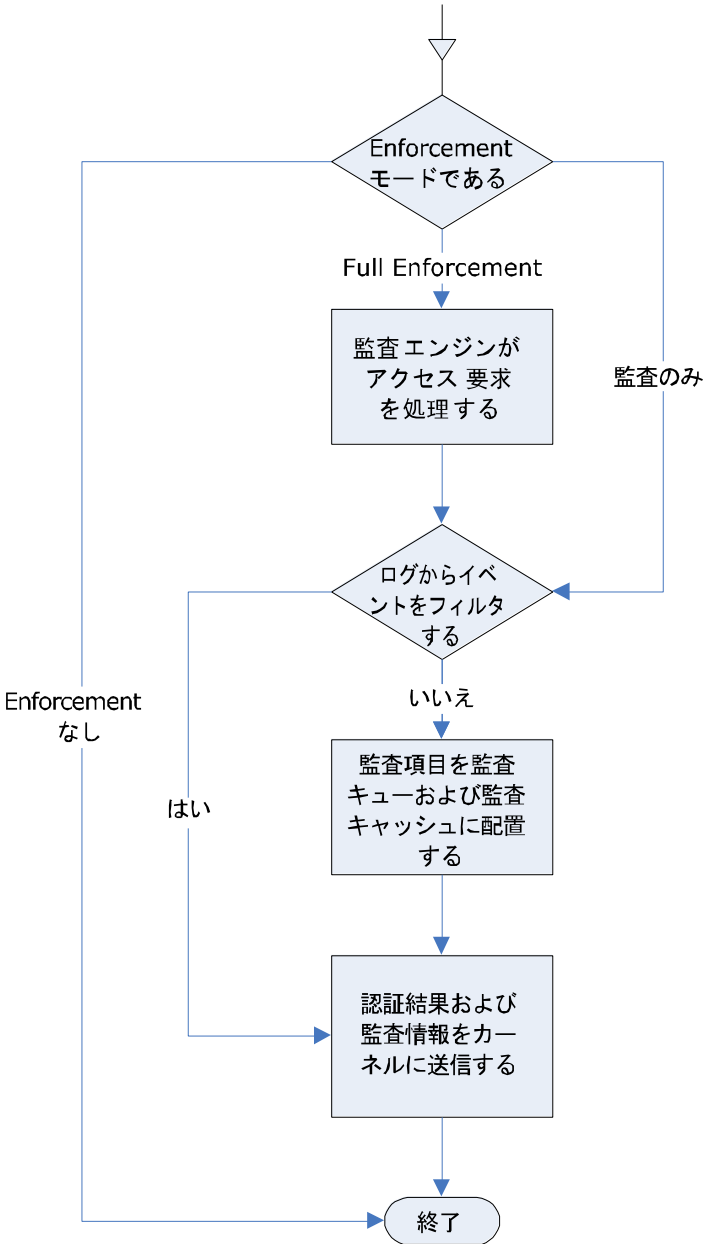
- 監査イベント（P. 144）

注: CA Access Control は、適切なクラスがアクティブで、データベースにこのイベントを予期するルールが含まれている場合のみ、イベントをインターセプトします。

インターセプト イベントの監査のしくみ

*インターセプト イベント*とは、CA Access Control に初めて出現し、カーネル キャッシュに認証または監査に関する情報が格納されていないイベントです。

監査レコードをログに記録するために、CA Access Control は以下のアクションを実行し、インターセプト イベントにその結果が反映されるようにします。


Enforcement
モードである
Full Enforcement
監査エンジンが
アクセス 要求
を処理する
監査のみ
ログからイベ
ントをフィルタ
する
いいえ
Enforcement
なし
はい
監査項目を監査
キューおよび監査
キャッシュに配置
する
認証結果および
監査情報をカー
ネルに送信する
終了

- No Enforcement モードでは、イベントはインターセプトも監査もされません。
- Full Enforcement モードでは、CA Access Control は以下の処理を行います。

  1. 認証エンジンは、認証結果に基づいて、監査項目を監査キュー、および監査キャッシュに配置します。

     CA Access Control が監査項目を書き込むのは、リソースまたはアクセサの監査プロパティが結果のイベントを監査するように設定され、監査フィルタ ファイルがこのイベントをフィルタするように設定されていない場合のみです。

  2. 認証エンジンは、認証結果に関する情報*および監査関連情報が含まれた応答をカーネルに返します。*

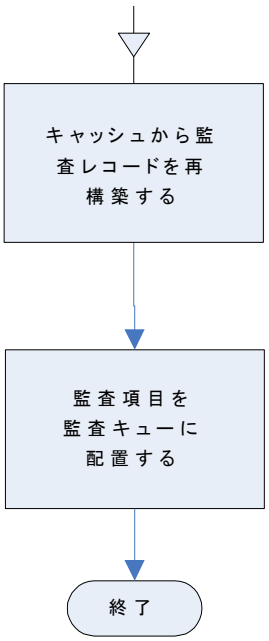
- Audit Only モードでは、CA Access Control は認証要求を処理しません。リソースおよびユーザの監査プロパティに関係なく、監査情報は常に書き込まれます。

  CA Access Control は、監査フィルタ ファイルがこのイベントをフィルタ処理するように設定されていない場合のみ、監査項目を書き込みます。 このモードでは、認証結果は常に *P*（許可）です。

注: インターセプトされたログイン イベント（TERMINAL クラス）、およびユーザ トレースによって生成された監査レコードはキャッシュされません。認証エンジンは、これらのイベントの監査レコードを常に書き込みます。

監査イベントの監査のしくみ

以下の図に、監査イベントの監査のしくみを示します。



カーネルが CA Access Control にキャッシュされたインターセプト イベントを通知すると、CA Access Control は以下のアクションを実行して、監査イベントをログに記録します。

1. カーネルから送信された情報が格納されている監査キャッシュを使用して、監査データを再構築します。
2. 監査項目を監査キューに挿入します。

カーネルおよび監査のキャッシュ

カーネル キャッシュには、以前にインターセプトされたイベントに関するデータが含まれています。 カーネルは、このようにキャッシュされたインターセプト イベント（監査イベント）を識別して、処理を実行する CA Access Control に送信します。 基本的に、CA Access Control は、カーネル キャッシュを使用して、以前にインターセプトされたイベントと同じ行動パターンを取るイベントをインターセプトします。

この監査キャッシュに含まれるデータによって、CA Access Control は再帰的な監査レコードを再構築します。再構築された監査レコードは、監査キューに送られ、認証プロセスは必要ありません。 これは、キャッシュにすでに十分な情報があるインターセプト イベント（監査イベント）はすぐに処理され、監査キューに追加されることを意味します。 認証エンジンが提供するデータは、認証エンジンがインターセプトした最初のイベントの結果、カーネル キャッシュおよび監査キャッシュに格納されているものです。

キャッシュのリセット

CA Access Control は、以下の場合に、カーネル キャッシュと監査キャッシュの両方をクリアします。

- データベースの変更時

  CA Access Control は、データベース情報の変更時に、キャッシュ全体をクリアします。アクセス ルールが新規に作成されたり、変更されると、既存のキャッシュが不正確になる可能性があります。

- 時間チェックポイント到達時

  CA Access Control は、時間チェックポイントが任意のイベントの認証結果に影響を及ぼすと、キャッシュ全体をクリアします。 DAYTIME 制限プロパティまたは HOLIDAY クラス レコードが変更されると。認証結果も変更され、キャッシュが不正確になる可能性があります。

- PROGRAM リソースの変更時

  CA Access Control は、PROGRAM リソースが変更され、untrusted 状態になったことを Watchdog が検知すると、キャッシュ全体をクリアします。 untrusted 状態のプログラムは、プログラムの認証要求の結果に影響を及ぼします。 これにより、キャッシュが不正確になる可能性があります。

- 監査キャッシュの飽和時

  CA Access Control は、監査キャッシュが飽和状態になると、キャッシュ項目の 10% （最も使用頻度の低いキャッシュ項目）をクリアします。

キャッシュがクリアされた後で、キャッシュに再び情報を格納し、CA Access Control で監査イベントをインターセプトできるようにするには、新規インターセプト イベントの情報が必要になります。

監査イベントの表示

CA Access Control は監査イベントを監査ログに送信します。 監査ログを表示するには、以下の CA Access Control ツールを使用します。

- CA Access Control エンドポイント管理
- seaudit ユーティリティ

また、監査イベントを Windows イベント ログに送信するように CA Access Control を設定できます。 イベント ログは、さまざまなアプリケーションからの監査イベントを 1 箇所に収集し格納します。 イベント ログで監査イベントを表示するには、Windows イベント ビューアを使用します。

Windows イベント ログの監査イベント

Windows イベント ログは、さまざまなソースからの監査イベントを 1 箇所に収集し格納します。 監査イベントをイベント ログにルーティングするように CA Access Control を設定すると、seosd が監査イベントを CA Access Control 監査ログに書き込むたびに、対応するイベントがイベント ログに送信されます。

audit.cfg ファイルは、監査ログおよびイベント ログの両方から監査イベントをフィルタします。 監査イベントが監査ログに書き込まれない場合、このイベントはイベント ログに送信されません。

また、Windows 2008 イベント ログではボリューム、オーディエンス、および監査イベントが発生したアプリケーションに応じて、監査イベントをチャネルと呼ばれるコンテナにルーティングします。 この CA Access Control チャネルの名前は、CA-AccessControl-AuthorizationEngine/Audit です。

Windows 2008 サーバ上に CA Access Control を展開すると、監査イベントの送信先として以下を選択できます。

- イベント ログ
- チャネル
- イベント ログおよびチャネルの両方に送信
- イベント ログおよびチャネルのどちらにも送信しない

Windows イベント ログへの監査イベントのルーティング

監査イベントを Windows イベント ログにルーティングするように CA Access Control を設定すると、seosd が監査イベントを CA Access Control 監査ログに書き込むたびに、対応するイベントがイベント ログに送信されます。 また、Policy Model 監査イベントをイベント ログに送信するように CA Access Control を設定することもできます。

イベントをイベント ログにルーティングする方法

1. 以下のコマンドを使用して、CA Access Control を停止します。

   ```
   secons -s
   ```

   CA Access Control が停止します。

2. logmgr セクションの SendAuditToNativeLog 設定の値を 1 にします。

   監査イベントが Windows イベント ログに送信されます。

3. （オプション） Pmd セクションの SendAuditToNativeLog 設定の値を 1 にします。

   Policy Model の監査イベントが Windows イベント ログに送信されます。

4. 以下のコマンドを使用して、CA Access Control を再起動します。

   ```
   seosd -start
   ```

   CA Access Control が再起動します。

例: イベント ログへの監査イベントのルーティング

以下の例では、監査イベントをイベント ログにルーティングします。 このコマンドを使用するには、リモート設定環境（env config）を使用する必要があります。

```
er config ACROOT section(logmgr) token(SendAuditToNativeLog) value(1)
```

例: イベント ログへの Policy Model 監査イベントのルーティング

以下の例では、Policy Model 監査イベントをイベント ログにルーティングします。 このコマンドを使用するには、リモート設定環境（env config）を使用する必要があります。

```
er config ACROOT section(Pmd) token(SendAuditToNativeLog) value(1)
```

詳細情報:

設定の変更 (P. 216)

Windows イベント ログ チャネルへの監査イベントのルーティング

Windows Server 2008 のみで有効

監査イベントを Windows イベント ログ チャネルにルーティングするように CA Access Control を設定すると、seosd が監査イベントを CA Access Control 監査ログに書き込むたびに、対応するイベントがイベント ログに送信されます。 CA Access Control イベント ログ チャネルの名前は、CA-AccessControl-AuthorizationEngine/Audit です。

また、Policy Model 監査イベントをイベント ログ チャネルに送信するように CA Access Control を設定することもできます。 Policy Model イベント ログ チャネルの名前は、CA-AccessControl-Policy Models/Audit です。

イベントをイベント ログ チャネルにルーティングする方法

1. 以下のコマンドを使用して、CA Access Control を停止します。

   ```
   secons -s
   ```

   CA Access Control が停止します。

2. logmgr レジストリ サブキーの SendAuditToNativeChannel トークンの値を 1 に設定します。

   監査イベントが Windows イベント ログチャネルに送信されます。

3. （オプション） Pmd レジストリ サブキーの SendAuditToNativeChannel トークンの値を 1 に設定します。

   Policy Model 監査イベントが Windows イベント ログ チャネルに送信されます。

4. 以下のコマンドを使用して、CA Access Control を再起動します。

   ```
   seosd -start
   ```

   CA Access Control が再起動します。

例: イベント ログ チャネルへの監査イベントのルーティング

以下の例では、監査イベントをイベント ログ チャネルにルーティングします。 このコマンドを使用するには、リモート設定環境（env config）を使用する必要があります。

```
er config ACROOT section(logmgr) token(SendAuditToNativeChannel) value(1)
```

例: イベント ログ チャネルへの Policy Model 監査イベントのルーティング

以下の例では、Policy Model 監査イベントをイベント ログ チャネルにルーティングします。 このコマンドを使用するには、リモート設定環境（env config）を使用する必要があります。

```
er config ACROOT section(Pmd) token(SendAuditToNativeChannel) value(1)
```

監査ログ

監査ログはファイルとして格納されます。 以下の Windows レジストリ サブキーの *audit_log* 値で、監査ログ ファイルの場所を指定します。

```
HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥logmgr
```

このキーのデフォルト値は、以下のとおりです。

```
C:¥Program Files¥CA¥AccessControl¥¥log¥seos.audit
```

デフォルトでは、CA Access Control は、監査ログのサイズが 1024 KB に達すると、監査ログを自動的にバックアップします。 このサイズを変更するには、以下のサブキーの値 *audit_size* を変更します。

```
HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥logmgr
```

また、監査ログを定期的に（毎日、毎週、または毎月）バックアップするように設定することもできます。設定するには、以下の Windows レジストリ サブキーの *BackUp_Date* 値を変更します。

```
HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥logmgr
```

注: これらのレジストリ サブキーの詳細については、「*リファレンス ガイド*」を参照してください。

監査ログの使用

CA Access Control には、監査ログの表示、フィルタ処理、および検索に使用する以下の 2 つの付属ツールがあります。

- CA Access Control エンドポイント管理
- seaudit ユーティリティ

監査ログのすべてのレコードを表示することも、フィルタを使用して監査ログから特定のレコードを選択することもできます。

次に、CA Access Control エンドポイント管理で監査フィルタを使用して監査ログのレコードを表示する方法について説明します。

監査レコード フィルタ

audit.cfg ファイルでは、監査ファイルに送信しないレコードを定義して、ホスト上の監査レコードをフィルタリングします。 ファイル内の各行は、監査情報を除外するためのルールを表します（つまり、各行の条件に一致した監査レコードは、監査ファイルに表示されません）。 このフィルタは、必要なレコードのみを保持して、seos.audit ファイルのサイズを制限する際に有用です。 企業の要件に合わせて、audit.cfg ファイルを編集することができます。

デフォルトでは、audit.cfg ファイルは *ACInstallDir*/etc ディレクトリ（UNIX）または *ACInstallDir*¥data ディレクトリ（Windows）に配置されています。 audit.cfg ファイルの場所は、seos.ini ファイル（UNIX）内の［logmgr］ AuditFiltersFile トークンまたは logmgr レジストリ キー（Windows）内の AuditFiltersFile エントリを編集することによって、変更できます。

CA Access Control エンジン （seosd） は、起動時に audit.cfg ファイルを読み取ります。 メッセージが監査ファイルに送信されると、seosd はそのメッセージが audit.cfg ファイル内のいずれかのルールに一致するかどうかをチェックします。 メッセージがルールに一致すると、そのメッセージは監査ファイルに書き込まれません。

注: audit.cfg ファイルの詳細については、「*リファレンス ガイド*」を参照してください。

監査表示フィルタ

監査ログのレコードは、膨大な数になる場合があります。*表示する*レコード数を減らすには、フィルタを使用して、表示するレコードの種類を指定します。 時間やイベント タイプなどのさまざまな基準に基づいて、イベントをフィルタ処理できます。

注: 監査構成設定（audit.cfg ファイル）を使用して、CA Access Control が監査ファイルに*書き込む*監査レコードをフィルタ処理することもできます。

CA Access Control エンドポイント管理でフィルタを作成するには、フィルタに名前を付け、少なくとも 1 つのスイッチを選択します。 次に、更にスイッチを選択することができます。1 つ以上のオプションを割り当てることができます。 seaudit ユーティリティでレコードをフィルタ処理することもできます。

CA Access Control エンドポイント管理には、複数の事前定義フィルタが用意されています。また、独自のフィルタを作成することもできます。

フィルタ ウィザードの［名前とスイッチの選択］ページ

フィルタ ウィザードの［名前とスイッチの選択］ページでは、作成する監査表示フィルタの名前、およびこのフィルタに適用するスイッチを定義できます。

このウィンドウには、以下のフィールドが含まれます。

フィルタ名

作成する監査表示フィルタの名前を定義します。

監査イベント レコード

監査レコードをフィルタ表示させるか、選択したスイッチが適用された監査レコードのみをフィルタ表示させるかを指定します。

すべてのレコードの一覧表示を選択した場合、このページのスイッチは適用されません。

ホストおよびサービスの INET 監査レコードを一覧表示

指定されたサービスの指定されたホストから受け取った TCP 要求の INET 監査レコードを表示するかどうかを指定します。 host および service は、検索対象のホストおよびサービスを特定するマスクです。

端末のユーザの LOGIN を表示

以下を一覧表示すること指定します。

- 指定した端末における指定したユーザの LOGIN レコード。 *user* と *terminal* のいずれもユーザが定義するマスクです。
- 無効なパスワードが複数回入力されたときに、認証エンジンによって作成されるレコード。

ユーザのリソースのクラスの RESOURCE 監査を一覧表示

リソース レコードを一覧表示するかどうかを指定します。以下の項目を後で定義できます。

- *Class* - アクセスされたリソースが属しているクラスを特定するマスク
- *Resource* - アクセスされたリソースの名前を特定するマスク
- *User* - リソースにアクセスしたユーザの名前を特定するマスク

データベースの更新を一覧表示

データベース更新の監査レコードを一覧表示します。 以下を定義できます。

- *Cmd* - 検索対象の selang コマンドを特定するマスク。
- *Class* - 検索対象のクラスを特定するマスク
- *Object* - 検索対象のレコードを特定するマスク
- *User* - コマンドを実行したユーザを特定するマスク

起動/停止のメッセージを一覧表示

CA Access Control サービスからスタートアップ メッセージおよびシャットダウン メッセージを一覧表示するかどうかを指定します。

WATCHDOG 監査レコードを一覧表示

監査レコードを一覧表示するかどうかを指定します。

トレース レコードのみを表示

トレース機能によって監査ログに送信されたレコードのみを一覧表示するかどうかを指定します。

フィルタ ウィザードの［オプションの編集］ページ

フィルタ ウィザードの［オプションの編集］ページでは、監査表示フィルタに適用するオプションを定義できます。

このウィンドウには、以下のフィールドが含まれます。

現在の日付から一覧表示

現在の日付を開始日として指定します。現在の日付よりも前にログに記録されたレコードは一覧表示されません。

一覧表示の開始日

開始日を指定します。 指定された日付より前にログに記録されたレコードは表示されません。

一覧表示の開始時刻

開始時刻を指定します。指定された時刻より前にログに記録されたレコードは表示されません。

一覧表示の終了日

終了日を指定します。 指定された日付より後にログに記録されたレコードは表示されません。

一覧表示の終了時刻

終了時刻を指定します。 指定した時刻より後にログに記録されたレコードは表示されません。

ホスト名ではなく、インターネット アドレスを表示

TCP/IP レコードのホスト名ではなく、インターネット アドレスを表示することを指定します。

失敗を非表示

失敗したレコードが表示されないように指定します。

許可されたアクセスのレコードを非表示

成功した（許可された）アクセスのレコードを表示しないことを指定します。

ログアウト レコードを非表示

ログアウト レコードを表示しないことを指定します。

NOTIFY 監査レコードを非表示

NOTIFY 監査レコードを表示しないことを指定します。

パスワード試行およびアクションを非表示

パスワード試行レコードを表示しないことを指定します。

警告レコードを非表示

警告レコードを表示しないことを指定します。

名前ではなくポート番号を表示

サービス名ではなくポート番号を表示することを指定します。

ホストから送信されたレコードのみ表示

指定したホストから送信されたレコードのみを表示することを指定します。このオプションは、UNIX ワークステーションに接続している場合にのみ適用可能です。

事前定義フィルタ

CA Access Control には、以下の事前定義フィルタがあります。

すべてのレコード

監査ログのすべてのレコードを表示します。フィルタ処理は行われません。

今日のレコード

今日作成されたレコードをすべて表示します。

過去 2 日間のレコード

昨日と今日に作成されたすべてのレコードを表示します。

過去 7 日間のレコード

過去 7 日間に作成されたすべてのレコードを表示します。

CA Access Control サービスへの接続

ユーザが CA Access Control エンドポイント管理や selang などの CA Access Control サービスにいつ接続したかを示すレコードを表示します。

注: UNIX ワークステーションに接続している場合は、このフィルタの名前は「ログイン レコード」になります。このレコードは、ユーザ ログインを表します。

管理アクティビティ

CA Access Control またはオペレーティング システムのデータベースを更新するすべてのレコードを表示します。 データベースの更新には、すべての種類のレコードの追加、削除、および変更が含まれます。

ユーザ定義フィルタの作成

フィルタは、必要な数だけ作成できます。 特定の監査レコード セットのみを表示したい場合は、カスタム フィルタを作成します。

ユーザ定義フィルタを作成するには、以下の手順に従います。

1. CA Access Control エンドポイント管理の［監査イベント］タブをクリックします。

   ［監査レコード ビューア - フィルタ設定］セクションに、保存済みフィルタのリストが表示されます。

2. ［保存済みフィルタ］セクションで、［フィルタの作成］をクリックします。

   ［監査フィルタ ウィザード］が表示されます。

3. ウィザード ページを終了します。

   名前とスイッチの選択

   フィルタで使用するスイッチ (P. 152)を指定します。

   スイッチの編集

   選択したスイッチの設定を指定します。基本的に、ここでは、フィルタ処理する監査イベントに対して定義するマスクを指定します。

   オプションの編集

   監査フィルタに設定するオプション (P. 154)を指定します。

   ［完了］をクリックします。

   定義した新しい監査フィルタが保存されて、読み込まれます。

監査ログのバックアップ

CA Access Control では、監査ログ ファイルを自動的にバックアップし、アーカイブすることができます。

監査ログ バックアップ ファイルの名前は、logmgr¥audit_back CA Access Control レジストリ エントリに設定されます。

以下の方法で、監査ログ ファイルをバックアップすることができます。

- サイズによるバックアップ
- 日付によるバックアップ

監査ログ ファイルのバックアップに使用する方法および設定は、以下の要因によって異なります。

- ログ ファイルのバックアップ コピーが必要かどうか
- 環境内でどの程度の監査データが生成される見込みか
- システムのパフォーマンスに関する問題（監査ログ ファイルのサイズが大きくなると、処理時間に影響があるなど）

注: タイムスタンプ付きバックアップを保持する設定の場合、CA Access Control により、監査ログのバックアップ ファイルがデフォルトで保護されます。 これはサイズによる監査バックアップ ファイルの受信と同じデフォルトの保護です。 これらのファイルを削除するには、データベースに許可ルールを設定することが必要です。

監査ログの自動バックアップのためのサイズ設定

監査ログ ファイルはサイズの制限を設定することができます。 ファイルが定義されたサイズに達すると、CA Access Control は自動的にファイルのバックアップ コピーを作成して、ログをクリアします。 これは、ファイルが定期的に自動バックアップされることを意味します。

監査ログの自動バックアップが開始されるサイズを設定するには、必要な最大値を KB 単位で、logmgr¥audit_size CA Access Control レジストリ エントリに設定します。

注: バックアップ ファイルの名前を定義するには、logmgr¥audit_back CA Access Control レジストリ エントリを設定します。

重要: logmgr/BackUp_Date CA Access Control レジストリ エントリが「yes」（デフォルトは「no」）に設定されている場合、監査ログの、サイズによるバックアップの各コピーには、名前の前にタイムスタンプが付けられます。 日付によるバックアップが設定されている場合など、それ以外の場合は、バックアップ コピーごとに、以前作成されたバックアップ コピーが上書きされます。

例: 監査ログ ファイルのサイズが 5 MB に達したら、自動バックアップを行うように設定する

この例では、監査ログ ファイルのサイズが 5 MB（5120 KB）に達したときに、バックアップするように設定する方法を示します。これを行うには、logmgr¥audit_size CA Access Control レジストリ エントリを **5120** に設定します。

監査ログ ファイルが 5 MB に達すると、CA Access Control はファイルのバックアップ コピーを作成し、デフォルトで「seos.audit.bak」という名前を付け、ログをクリアします。

例: 監査ログ ファイルのサイズが 1 MB に達したら、自動バックアップを行うように設定し、カスタム名とタイムスタンプを付ける

この例では、監査ログ ファイルのサイズが 1 MB（1024 KB）に達したら、バックアップするように設定し、バックアップ ファイルにカスタム名を付け、名前にタイムスタンプを追加する方法を示します。

これを行うには、以下のように CA Access Control レジストリ エントリを設定します。

- logmgr¥audit_size=1024
- logmgr¥audit_back=log¥ac_audit.old
- logmgr¥BackUp_Date=yes

監査ログ ファイルのサイズが 1 MB に達すると、CA Access Control はファイルのバックアップ コピーを作成し、ログをクリアします。 バックアップ ログ ファイル名は「ac_audit.old.*timestamp*」という形式になります。ここで、*timestamp* は、「DD-Mon-YYYY.hhmmss」形式の日時です。 以下に例を示します。

```
ac_audit.old.06-Feb-2007.144330
```

監査ログの自動バックアップの時間間隔の指定

CA Access Control が監査ログ ファイルのバックアップ コピーを自動的に作成して、ログを削除する時間間隔（毎日、毎週、毎月）を定義できます。

監査ログが自動的にバックアップされる時間間隔を設定するには、logmgr¥BackUp_Date CA Access Control レジストリ エントリで時間間隔を設定します。 時間間隔には、以下のいずれかを指定できます。

毎日

1 日に 1 回、監査ログ ファイルをバックアップします。

毎週

週に 1 回、監査ログ ファイルをバックアップします。

毎月

月に 1 回、監査ログ ファイルをバックアップします。

注: バックアップ ファイルの名前を定義するには、logmgr¥audit_back CA Access Control レジストリ エントリを設定します。

重要: 監査ログが、バックアップ間隔より前に、logmgr¥audit_size CA Access Control レジストリ エントリで指定されたサイズの制限に達すると、CA Access Control はファイルのバックアップ コピーを作成しますが、タイムスタンプは付与されません。このような各バックアップ コピーによって、以前のコピーが上書きされる可能性があります。

例: 監査ログ ファイルの日次バックアップを設定する

この例では、監査ログ ファイルの日次バックアップの設定方法について説明します。 これを行うには、logmgr¥BackUp_Date CA Access Control レジストリを **daily** に設定します。

1 日に 1 回、CA Access Control はファイルのバックアップ コピーを作成し、ログをクリアします。 バックアップ ログ ファイル名には「*.timestamp*」というサフィックスが付けられます。この *timestamp* は、「DD-Mon-YYYY.hhmmss」という日時形式になります。 以下に例を示します。

```
seos.audit.bak.06-Feb-2007.144330
```

第 9 章: 管理者権限の適用範囲

このセクションには、以下のトピックが含まれています。

グローバル権限属性

グローバル権限属性は、ユーザ レコードに設定します。 グローバル権限属性が設定されると、ユーザは特定の種類の機能を実行できます。 このセクションでは、各グローバル権限属性の機能および制限について説明します。

ADMIN 属性

ADMIN 属性により、ユーザは CA Access Control のほとんどすべてのコマンドを実行できます。 データベースで ADMIN 属性が割り当てられているユーザは、データベースのユーザ、グループ、およびリソースを定義および更新できます。 この属性は CA Access Control 内で最も強力な属性です。ただし、以下のような制限があります。

- データベース内で 1 ユーザのみに ADMIN 属性が割り当てられている場合は、そのユーザを削除できません。また、そのユーザのレコードから ADMIN 属性を削除することもできません。

- ADMIN 属性は割り当てられているが AUDITOR 属性は割り当てられていないユーザは、ユーザ、グループ、またはリソースに対して行われる監査の種類（監査モード）を変更できません。 ADMIN 属性があり、ユーザ、グループ、またはリソースに関する監査特性を変更する必要がある場合は、AUDITOR 属性を自分自身に割り当てる必要があります。
- ADMIN 属性が割り当てられたユーザは、スーパーユーザ（UNIX の root アカウントまたは Windows の Administrator アカウント）を削除できません。ただし、スーパーユーザ を ADMIN 以外のユーザに設定することはできます。

AUDITOR 属性

AUDITOR 属性が割り当てられたユーザは、システムの使用状況を監視できます。 AUDITOR 属性が割り当てられたユーザの明示的な権限により、以下のことが可能です。

- データベース内の情報を表示できます。

  監査者は、selang のコマンド *showusr*、*showgrp*、*showres*、および *showfile* を実行できます。

- 既存のレコードに対して監査モードを設定できます。

  監査者は、selang のコマンド *chusr*、*chgrp*、*chres*、および *chfile* を実行できます。

OPERATOR 属性

OPERATOR 属性が割り当てられたユーザには、すべてのファイルに対する READ アクセス権があります。 このアクセス権により、データベース内のすべてのデータを一覧表示したり、バックアップ ジョブを実行できます。オペレータは、showusr、showgrp、showres、showfile、および find の各コマンドを使用して、データベースのレコードを一覧表示できます。
OPERATOR 属性では、secons ユーティリティを使用することもできます。

注: secons ユーティリティの詳細については、「リファレンス ガイド」を参照してください。

PWMANAGER 属性

PWMANAGER 属性は、他のユーザのパスワードを変更するための chusr コマンドまたは sepass コマンドの使用権限を一般ユーザに付与します。

注: PWMANAGER によって ADMIN ユーザのパスワードを変更するには、setoptions コマンドの *cng_adminpwd* オプションを設定します。 詳細については、「*selang リファレンス ガイド*」を参照してください。

PWMANAGER 属性には、猶予ログイン回数、別のユーザのパスワード期間、または一般的なパスワード ルールを変更する権限は含まれていません。

PWMANAGER の権限には、showusr コマンドおよび find コマンドの使用権限も含まれます。

注: ユーザが nochngpass プロパティを yes に設定した場合、PWMANAGER ではそのユーザのパスワードを変更できません。

SERVER 属性

CA Access Control では、他の多くのセキュリティ モデルと同様に、一般ユーザによる「ユーザ A はリソース X にアクセスできるか」というクエリを許可していません。一般ユーザが可能な唯一のクエリは、「自分はリソース X にアクセスできるか」です。ただし、データベース サーバ サービスや社内アプリケーションのように、サービスを多数のユーザに提供するプロセスでは、他のユーザに代わって権限を照会することが許可されます。

SERVER 属性により、ユーザの権限をクエリするプロセスが許可されます。SERVER 属性が割り当てられたユーザは、SEOSROUTE_VerifyCreate API を発行できます。

注: ERVER 属性および CA Access Control API の詳細については、「*SDK 開発者ガイド*」を参照してください。

IGN_HOL 属性

IGN_HOL 属性は、HOLIDAY レコードに定義されている期間中のログインをユーザに許可します。HOLIDAY クラスの各レコードは、ログイン時に特別な許可が必要となる 1 つ以上の期間を定義します。 IGN_HOL 属性を持つユーザは、HOLIDAY レコードに定義されている期間に関係なく、いつでもログインすることができます。

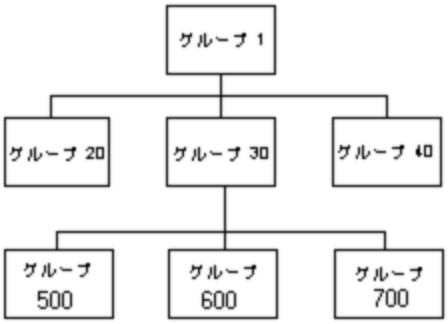
注: HOLIDAY クラスの詳細については、「*リファレンス ガイド*」を参照してください。

グループ権限

グループ権限属性を理解するには、親子関係の概念を理解しておく必要があります。

親子関係

下位グループと上位グループの概念は、親子関係ともいわれ、グループ管理者権限を説明する場合に重要です。1 つのグループは、1 つ以上のグループの親（上位）になることができます。1 つの子（つまり、下位グループ）に対して親として設定できるグループは 1 つのみです。グループへの親の割り当ては、必要に応じて行います。 以下の図について考えてみましょう。



グループ 1 は、3 つのグループ（20、30、および 40）の親です。 グループ 30 も、3 つのグループ（500、600、および 700）の親です。 グループ 600 の親は 1 つのみ（グループ 30） です。 グループ 1 には親がありません。

グループ権限属性

リソース レコードおよびアクセサ レコードなど、すべてのレコードには所有者がいます。 レコードを「所有している」ということは、レコードを表示、編集、および削除する権限があることを意味します。

グループは、それぞれのレコードを所有できます。 ただし、レコードを所有するグループ内で、レコードを管理できるのは、特定の権限があるユーザのみです。 この特別なユーザには、グループ権限属性がそれぞれのユーザ レコードに設定されています。 グループ権限属性は、以下のとおりです。

- GROUP-ADMIN
- GROUP-AUDITOR
- GROUP-OPERATOR
- GROUP-PWMANAGER

これらの属性は、join コマンドによって設定されます。このコマンドは、正当な権限のあるユーザのみが発行できます。 join コマンドには、ユーザを 1 つのグループにまとめるという役割、およびユーザにグループ権限属性がある場合はその属性を指定するという役割があります。

グループのメンバを定義するユーザ レコードを管理する権限がグループの特権メンバに付与されるかどうかは、そのユーザ レコードの所有者によって決まります。

詳細情報:

所有者権限 (P. 168)

GROUP-ADMIN 属性

グループ管理者権限属性が割り当てられたユーザは、特定のレコードの集合を作成できます。 レコードを作成するために、グループ管理者はそのレコードの所有者を指定する必要があります。

レコードの所有者は、ユーザにグループ権限属性が設定されているグループである必要があります。 そのグループが他のグループの親である場合、所有者はその下位グループの 1 つでもかまいません。これらのレコードの集合全体を「グループの有効範囲」といいます。 権限の例では、グループの有効範囲の概念を示します。

GROUP-ADMIN 属性が割り当てられたユーザには、グループの有効範囲内にあるレコードに対する以下のアクセス権限があります。

アクセス	説明	コマンド
Read	レコードのプロパティを表示します。	showusr、showgrp、showres、showfile
Create	データベースで新しいレコードを作成します。所有者を指定する必要があります。	newusr、newgrp、newres、newfile
Modify	レコードのプロパティを変更します。	chusr、chgrp、chres、chfile
Delete	データベースからレコードを削除します。	rmusr、rmgrp、rmres、rmfile
Connect	ユーザをグループに追加またはグループから分離します。	join、join-

GROUP-ADMIN 属性には、以下のような制限事項もあります。

- GROUP-ADMIN ユーザは自分に対してリソースをアクセス不可に設定できません。したがって、以下の制限を受けます。
  - GROUP-ADMIN ユーザは、自分のセキュリティ レベルより高いセキュリティ レベルを割り当てることはできません。
  - GROUP-ADMIN ユーザは、自分が所有していないセキュリティ カテゴリまたはセキュリティ ラベルを割り当てることはできません。
- GROUP-ADMIN ユーザは、データベースからスーパーユーザ（UNIX の root アカウントまたは Windows の Administrator アカウント）を削除できません。

- 以下に示すいくつかの制限は、この章の「グローバル権限属性」で説明しているグローバル権限属性に関連があります。
  - GROUP-ADMIN ユーザは、データベース内で唯一の ADMIN ユーザレコードを削除できません。
  - GROUP-ADMIN ユーザは、データベース内の最後の ADMIN ユーザのレコードから ADMIN 属性を削除できません。
  - AUDITOR 属性のない GROUP-ADMIN ユーザは、監査モードを更新できません。 監査モードを更新できるのは、AUDITOR 属性が割り当てられた GROUP-ADMIN ユーザのみです。
  - GROUP-ADMIN ユーザは、どのユーザに対しても、グローバル権限属性（ADMIN、AUDITOR、OPERATOR、PWMANAGER、および SERVER）を設定できません。

GROUP-AUDITOR 属性

GROUP-AUDITOR 属性が割り当てられたユーザは、グループの適用範囲内にあるすべてのレコードのプロパティを一覧表示できます。 グループ監査者は、グループの適用範囲内のレコードに対して、監査モードを設定することもできます。

GROUP-OPERATOR 属性

GROUP-OPERATOR 属性が割り当てられたユーザは、グループの適用範囲内にあるすべてのレコードのプロパティを一覧表示できます。

GROUP-PWMANAGER 属性

GROUP-PWMANAGER 属性が割り当てられたユーザは、グループの有効範囲内にレコードがあるユーザのパスワードを変更できます。

所有者権限

データベースのすべてのレコード（アクセサ レコードおよびリソース レコードの両方）には、所有者が存在します。 レコードをデータベースに追加する場合は、owner パラメータを使用してレコードの所有者を明示的に割り当てるか、または CA Access Control によって、レコードを定義したユーザをレコードの所有者として割り当てることができます。

以下の*いずれか*が true の場合、アクセサがレコードを所有します。

- これらはレコードの所有者として定義されます。
- これらはレコードの所有者として定義されたグループのメンバであり、*なおかつ*GROUP-ADMIN 属性でグループのメンバとして追加されています。
- これらは、リソースがメンバとなっているリソース グループ レコードの所有者です。

レコードを所有するユーザまたはグループをデータベースから削除すると、そのレコードは所有者のないレコードになります。

レコードを所有するユーザには、所有するレコードに対して以下のアクセス権限があります。

アクセス	説明	コマンド
Read	レコードのプロパティを表示します。	showusr、showgrp、showres、showfile
Modify	レコードのプロパティを変更します。	chusr、chgrp、chres、chfile
Delete	データベースからレコードを削除します。	rmusr、rmgrp、rmres、rmfile
Connect	ユーザをグループに追加またはグループから分離します。	join、join-

ユーザまたはグループに特定レコードに対する所有者権限を与えない場合は、レコードおよびそのレコードがメンバとなっているすべてのリソース グループ レコードの所有者に対して *nobody* 強調を指定します。

所有者権限に関する制限事項は、以下のとおりです。

- データベース内の最後の ADMIN ユーザの所有者は、そのユーザ レコードを削除できません。
- AUDITOR 属性のない所有者は、監査モードを更新できません。 監査モードを更新できるのは、AUDITOR 属性が割り当てられた所有者のみです。
- スーパーユーザ（UNIX の root アカウントまたは Windows の Administrator アカウント）の所有者は、データベースから スーパーユーザを削除できません。
- 所有者は、自分が所有するユーザに対して、グローバル権限属性（ADMIN、AUDITOR、OPERATOR、および PWMANAGER）を設定できません。
- 所有者は、自分に対してリソースをアクセス不可に設定できません。従って、以下の制限を受けます。
  - 所有者は、自分のセキュリティ レベルより高いセキュリティ レベルを割り当てることはできません。
  - 所有者は、自分が所有していないセキュリティ カテゴリまたはセキュリティ ラベルを割り当てることはできません。

ファイルの所有者権限

CA Access Control では、FILE クラスにレコードを定義することによって、ファイルの所有者がファイルを保護することを許可します。 ファイルの所有者には、そのファイルのレコードに関するすべての権限があります。そのため、所有者はそのファイルを保護するレコードについて、newfile、chfile、showfile、authorize、および authorize- の各コマンドとすべてのパラメータを使用できます。

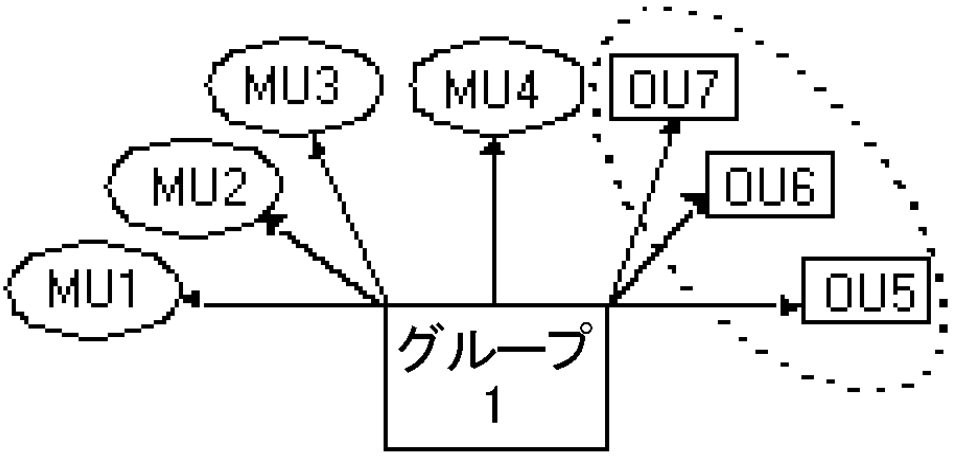
UNIX では、ユーザがファイルを作成すると、そのユーザがファイルの所有者に指定されます。 CA Access Control では、この機能が明示的に無効にされない限り、UNIX のファイル所有者による FILE レコードの定義が許可されます。 ファイル所有者による FILE レコードの定義を許可しない場合は、seos.ini ファイルの［seos］セクションにある use_unix_file_owner トークンを no に設定します（これはデフォルトの設定です）。

権限の例

グループ権限属性、親子関係、所有者権限、メンバシップ、およびグループの適用範囲の概念をこの後の図に示します。 これらの図にはユーザおよびグループしか含まれていませんが、所有者権限の概念はリソースおよびファイル レコードにも適用されます。

単一グループの権限

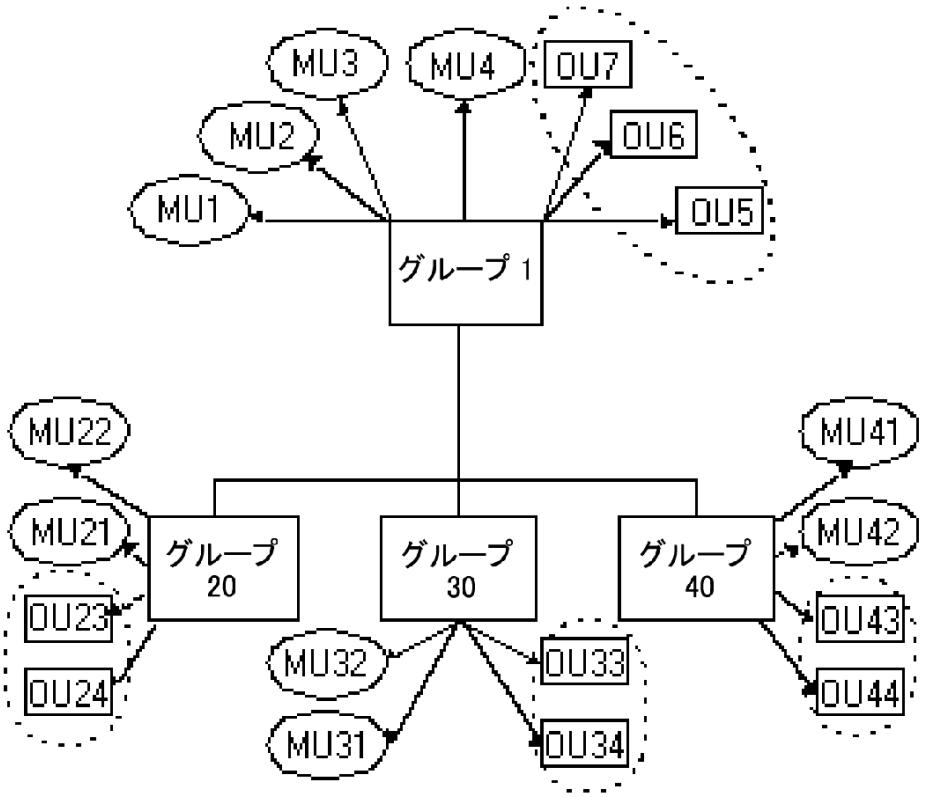
以下の図では、4 人のユーザ（MU1、MU2、MU3、および MU4）が グループ 1 のメンバです。グループ 1 は、3 人のユーザ（OU5、OU6、および OU7）も所有しています。 メンバ MU4 には GROUP-ADMIN 属性が設定されています。



点線で囲まれた部分は、ユーザ MU4 が実行するコマンドの影響を受けるグループの適用範囲を示します。この適用範囲には、グループ 1 が所有するすべてのユーザ（OU5、OU6、および OU7）が含まれます。

親グループおよび子グループ

以下の図では、4 人のユーザ（MU1、MU2、MU3、および MU4）が グループ 1 のメンバです。グループ 1 は、3 人のユーザ（OU5、OU6、および OU7）も所有しています。メンバ MU4 のレコードには、GROUP-ADMIN 属性が設定されています。



グループ 1 も、3 つのグループ（20、30、および 40）の親です。 これらの下位グループにはそれぞれ、そのグループのメンバである 2 人のユーザと、そのグループが所有する 2 人のユーザがいます。

点線で囲まれた 4 つの部分は、ユーザ MU4 が実行するコマンドの影響を受けるグループの適用範囲を示します。この適用範囲には、グループ 1 が所有するすべてのユーザと、グループ 1 の下位グループが所有するすべてのユーザが含まれます。MU4 のグループの適用範囲に含まれるユーザは、OU5、OU6、OU7、OU23、OU24、OU33、OU34、OU43、および OU44 です。

仮に グループ 20、30、または 40 にも下位グループがあり、ユーザ、グループ、またはリソースを所有していた場合は、これらの下位グループが所有するレコードも、ユーザ MU4 が実行するコマンドの影響を受けるグループの適用範囲に含まれます。

サブ管理

セキュリティ管理者（ADMIN 属性が割り当てられたユーザ）は、一般ユーザに特定の管理者権限を与えることができます。 このような一般ユーザをサブ管理者といいます。 サブ管理者には、指定した CA Access Control のクラスまたはオブジェクトのみを管理する権限が与えられます。 たとえば、サブ管理者に、ユーザ オブジェクトとグループ オブジェクトのみを管理する権限を与えることができます。 また、クラスの特定のオブジェクトの管理者権限をサブ管理者ユーザに与えることによって、より高いレベルのサブ管理者を設定できます。

ユーザ、グループ、およびリソースのサブ管理者は、selang を使用して、これらのリソースに関連する管理タスクを実行できます。

特定の管理権限を一般ユーザに付与する方法

管理者、つまり ADMIN 属性が割り当てられたユーザは、CA Access Control のほぼすべてのアクションを実行できるため、特定の管理タスクをサブ管理者に委任したい場合があります。 この場合は、以下のように、CA Access Control データベースで、ユーザが実行する必要がある特定の管理タスクを制御するクラスに対する権限をそのユーザに付与する必要があります。

1. 委任するタスクを制御する 1 つ以上のクラスを識別します。

   たとえば、CA Access Control は、USER クラスと GROUP クラスを使用して、アクセサ リソースを作成します。アクセサ管理を委任する場合は、ADMIN クラスの USER レコードと GROUP レコードを使用する必要があります。

2. 1 人以上のサブ管理者に、ADMIN クラスの該当リソースに対する権限を付与します。

   たとえば、サブ管理者がユーザ レコードを表示および変更できる権限を与えるには、そのユーザに、ADMIN クラスの USER レコードに対する読み取りアクセス権と変更アクセス権を付与します。

ADMIN クラス

ADMIN クラスのレコードのアクセス制御リスト（ACL）に指定されているユーザには、ADMIN 属性が割り当てられたユーザと同じ権限があります。ただし、ADMIN クラスのレコードの ACL に指定されているユーザの権限は、そのレコードが示す特定のクラスに制限されます。 たとえば、ADMIN クラスの SURROGATE レコードでは、SURROGATE クラスのレコードを管理できるユーザが決定されます。

注: CA Access Control クラスの詳細については、「*リファレンス ガイド*」を参照してください。

ADMIN クラスにある特定レコードの ACL に指定されているユーザは、以下のコマンドを実行できます。

アクセス	説明	コマンド
Read	クラスのレコードのプロパティを表示します。	showusr、showgrp、showres、showfile、find
Create	クラスで新しいデータベース レコードを作成します。	newusr、newgrp、newres、newfile
Modify	クラスのプロパティを変更します。	chusr、chgrp、chres、chfile
Delete	データベースから既存のクラス レコードを削除します。	rmusr、rmgrp、rmres、rmfile
Connect	ユーザをグループに追加したり、グループから除外したりします。 このアクセス権限は GROUP レコードの ACL でのみ有効です。	join、join-
パスワード	データベース内の全ユーザのパスワードとその属性を管理します。 PWMANAGER 属性が割り当てられたユーザに許可されているアクセス権限と同じ権限が与えられます。 このアクセス権限は、USER レコードの ACL でのみ有効です。	chusr

ADMIN クラス権限を持つユーザには、以下の制限事項があります。

- ADMIN クラスにある USER レコードの ACL に定義されているユーザは、データベース内の最後の ADMIN ユーザを削除できません。
- ADMIN クラス ユーザは、自分が所有するユーザに対して、グローバル権限属性（ADMIN、AUDITOR、OPERATOR、および PWMANAGER）を設定できません。
- すべての ADMIN クラス ユーザが、監査モードを更新できるわけではありません。 監査モードを更新できるのは、AUDITOR 属性が割り当てられた ADMIN クラス ユーザのみです。
- ADMIN クラスのユーザは、スーパーユーザ（UNIX の root アカウントまたは Windows の Administrator アカウント）を削除できません。ただし、スーパーユーザ を NOADMIN に設定することはできます。
- ADMIN クラス ユーザは、自分に対してリソースをアクセス不可に設定できません。したがって、以下の制限を受けます。
  - ADMIN クラス ユーザは、自分のセキュリティ レベルより高いセキュリティ レベルをリソースに割り当てることはできません。
  - ADMIN クラス ユーザは、自分が所有していないセキュリティ カテゴリまたはセキュリティ ラベルを割り当てることはできません。

これらの制限は、B1 セキュリティ レベル認証の一部です。

環境に関する考慮事項

データベース内の情報を更新できるかどうかを制御する要因の 1 つとして、該当する環境でユーザが占めるポジションが挙げられます。

リモート管理の制限

管理者は、ネットワーク上のリモート端末にアクセスし、その端末のデータベースを更新できます。 リモート端末のデータベースを更新するには、管理者自身と管理者の端末の両方に許可が必要です。

- 管理者は、リモート端末のデータベースでユーザとして明示的に定義されている必要があります。 実行するコマンドの種類に関係なく、リモート端末のデータベースにある自分のユーザ レコードに、適切な属性が設定されている必要があります。
- リモート端末にアクセスするための WRITE 権限を与えるルールの中に、自分のローカル端末のニーズを明示的に記述する必要があります。記述がない場合、リモート端末での CA Access Control 管理を実行することはできません。

  デフォルトのアクセス フィールド（_default）または UACC クラスに WRITE 権限が設定されている場合は、リモート端末で selang のコマンド シェルを入力できます。 ただし、selang のコマンドを実行することはできません。また、リモート データベースにアクセスすることもできません。 READ 権限が設定されている場合、リモート端末にログインすることはできますが、その端末での CA Access Control 管理を実行することはできません。

  この WRITE 権限と READ 権限の違いの例を以下に示します。

  1. 新しい端末をデフォルトのアクセス権限に READ を使用して指定するには、以下のコマンドを発行します。この権限では、管理者はその端末からログインすることはできますが、データベースを操作することはできません。

     ```
     newres TERMINAL tty13 defacc(read)
     ```

  2. 新しい端末からデータベースを操作する権限を ADMIN1 というユーザに与える（つまり、WRITE 権限と READ 権限の両方を与える）には、以下のコマンドを発行します。

     ```
     authorize TERMINAL tty13 uid(ADMIN1) access(r,w)
     ```

UNIX 環境

UNIX でユーザおよびグループを管理する場合、CA Access Control でグローバル権限属性またはグループ権限属性が割り当てられたユーザには、CA Access Control の場合と同じ権限と制限が UNIX でも適用されます。

インストール時など、seosd デーモンが実行されて *いない* 状態で selang を使用する場合は、以下のルールに従う必要があります。

- selang のコマンドに必ず -l オプションを指定すること。
- selang のユーザは root であること。この排他的な root 権限は、UNIX の一般的な制限事項に準拠しています。

Windows 環境

ネイティブ Windows 環境で有効

CA Access Control の実行中に、selang を使用してネイティブ Windows 環境内のリソースを変更する場合、CA Access Control エージェントは適切な Windows リポジトリのリソースを変更します。 リソースを変更するのに、追加の Windows 許可は必要ありません。これは、グローバルまたはグループ権限属性を備えた CA Access Control 内のユーザがネイティブ Windows 環境内で selang コマンドを実行する際に、これらのユーザには CA Access Control で行う場合と同じ特権および制限が Windows でもあることを意味します。

CA Access Control が実行されていないとき、selang を使用してネイティブ Windows 環境内のリソースを変更する場合、以下のルールに従う必要があります。

- selang のコマンドに必ず -l オプションを指定すること。
- ADMIN 属性またはサブ管理権限を持っていること。
- リソースを変更するのに十分な Windows 許可を持っていること。

  この制限が発生するのは、CA Access Control エージェントではなく selang プロセスが Windows リポジトリのリソースを変更するためです。

たとえば、ユーザ、 Emma がネイティブ Windows 環境内で chfile selang コマンドを使用してファイル C:¥tmp.txt の所有者を変更したいとします。CA Access Control が実行されている場合、Emma はファイル所有者を変更するのに十分な CA Access Control 許可が必要ですが、追加の Windows 許可は必要ありません。 CA Access Control が実行されていない場合、 Emma はファイル所有者を変更するための CA Access Control 許可および Windows 許可の両方が必要です。

データベースにアクセスするためのデフォルト許可

CA Access Control が実行されているとき、CA Access Control は内部ファイル ルールで内部データベース（seosdb）を保護します。 内部ファイル ルールは、selang に表示されず、削除できません。 FILE ルールを記述して、内部ファイル ルールを置き換えることができます。 これらの FILE ルールを削除すると、CA Access Control では内部ファイル ルールが復帰します。

CA Access Control が実行されている場合、以下の内部ファイル ルールがデータベースを保護します。

- CA Access Control の内部プロセスにはデータベースに対するフル アクセス権限があります。
- NT AUTHORITY¥System ユーザにはデータベースに対する読み取りアクセス権限があります。
- 他のすべてのアクセサにはデータベースに対するアクセス権限がありません。

注: 他のすべてのアクセサ用のデフォルト アクセス権限は r12.5 SP3 で変更されました。 以前のリリースでは、他のすべてのアクセサはデフォルトでデータベース ファイルに対して読み取りアクセス権を持っていました。

デフォルトでは、CA Access Control をインストールした後、または、エンドポイントを再起動した後、CA Access Control サービスは自動的に実行されます。 したがって、購入直後のデータベースにアクセスできるただ一人のユーザは NT AUTHORITY¥System です。 さらに、インストール中に CA Access Control 管理者を定義すると、その CA Access Control 管理者は selang などのユーティリティを使用してデータベースを更新できます。

データベースにアクセスするためのネイティブ許可

CA Access Control が停止しているとき、データベース ファイルへのアクセス権限はネイティブ Windows 許可によって決定されます。 許可は、CA Access Control がインストールされている親ディレクトリから継承されます。この継承のために、CA Access Control が停止しているとき、データベース ファイルへのデフォルト アクセスは「読み取り」になります。

個別の企業要件に適合するため、CA Access Control が停止しているときには CA Access Control を保護するようにデータベース ファイルの Windows 許可を変更してもかまいません。 許可を変更する前に、以下の点を考慮してください。

- NT AUTHORITY¥System ユーザには、データベース ファイルに読み書きするための Windows 許可が必要です。

  CA Access Control 認可エンジンは、NT AUTHORITY¥System ユーザから権限を継承します。 このユーザがデータベースにアクセスすることができない場合、エンジンはデータベースを更新するのに十分なネイティブ権限を持っていません。

- CA Access Control が停止しているときに CA Access Control に読み書きアクセスが必要なユーザには、データベース ファイルを読み書きするための Windows 許可が必要です。

  読み書きアクセスが必要なユーザには、CA Access Control をバックアップ、リストア、またはアップグレードするユーザが含まれます。

- CA Access Control が停止しているときに selang（selang -l オプション）を使用できるユーザは以下の許可を持っている必要があります。

  - ADMIN 属性またはサブ管理権限
  - データベース ファイルを読み書きするための Windows 許可
  - ネイティブ リポジトリを変更するための Windows 許可（必要な場合）

    たとえば、CA Access Control が停止しているときに config 環境を使用して CA Access Control レジストリ エントリを変更するには、レジストリを変更するための十分な Windows 権限が必要です。

  CA Access Control 管理者（ADMIN 属性またはサブ管理権限を持ったユーザ）のみ、CA Access Control が停止しているときに selang を使用してデータベースをメンテナンスすることができます。CA Access Control が停止しているときに CA Access Control 管理者がデータベースにアクセスすることができない場合、ユーザはオフライン データベース メンテナンスを実行することができません。また、デッドロックが発生する場合があります。

第 10 章: Policy Model の管理

このセクションには、以下のトピックが含まれています。

Policy Model データベース

何百、何千ものデータベースを個別に管理することは、現実的ではありません。 CA Access Control には、1 台の中央データベースから多数のデータベースを管理できるコンポーネントである Policy Model サービスが用意されています。Policy Model（PMD）サービスの使用は任意ですが、このサービスを使用すると、大規模なサイトでの管理を大幅に簡略化できます。

注: Windows のタスク マネージャでは、Policy Model サービスは sepmdd.exe と表示されます。

Policy Model サービスは、Policy Model データベース（PMDB）を使用します。 PMDB には、他の CA Access Control データベースと同様に、ユーザ、グループ、保護されているリソース、およびリソースへのアクセスを管理するルールが保存されています。 PMDB にはこのほかに、サブスクライバ データベースのリストが含まれます。 各サブスクライバは、別々のコンピュータに存在する CA Access Control データベース、または同じコンピュータまたは別のコンピュータに存在する別の PMDB です。サブスクライバを更新する PMDB をサブスクライバの親といいます。

PMDB は、同様の許可制約およびアクセス ルールが適用される多数のデータベースを管理するための便利なツールです。

UNIX との互換性を維持するために、Windows では Policy Model 名の大文字と小文字が区別されます。コマンドで PMDB 名を指定する場合、大文字小文字を間違えないようにしてください。 PMDB 名の最初の文字は、英数文字 '-' および '_' とする必要があります。

注: PMDB およびホスト名に英文字以外の文字は使用できません。

PMDB 名では大文字小文字が区別されますが、同じコンピュータ上で、大文字小文字のみが異なる PMDB を 2 つ持つことはできません。 CA Access Control では PMDB 名がファイル パスの一部として使用されますが、Windows では大文字小文字が区別されないため、これは許可されません。たとえば、myPMDB と MYpmdb は 2 つの異なる Policy Model データベースですが、同じシステム上で共存できません。

注: PMDB の管理方法（sepmd ユーティリティ）の詳細については、「リファレンス ガイド」を参照してください。 selang の使用によるリモートでの PMDB の管理方法の詳細については、「selang リファレンス ガイド」を参照してください。

ディスク上の PMDB の場所

1 台のコンピュータ上にある PMDB はすべて、共通ディレクトリに保存されます。 ディレクトリ名は、以下の Windows レジストリ サブキーの _pmd_directory_ 値で指定します。

```
HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥Pmd
```

NTFS ルート ディレクトリの _pmd_directory_ のデフォルト値は、*ACInstallDir*¥data です（ここで、*ACInstallDir* は CA Access Control のインストール ディレクトリで、デフォルトでは C:¥Program Files¥CA¥AccessControl¥です）。

各 PMDB は、共通ディレクトリ内のサブディレクトリに格納されます。サブディレクトリ内のファイルには、Policy Model を定義するために必要なすべてのデータが含まれています。 Policy Model の設定は、CA Access Control のレジストリ設定の Pmd サブキーに格納されます。 Policy Model の名前がそのままサブキーの名前になります。

ローカル PMDB の管理

CA Access Control には、PMDB を管理するためのユーティリティが用意されています。

sepmd

以下を実行できる PMDB 管理ユーティリティ。

- サブスクライバの管理
- 更新ファイルの切り捨て
- デュアル コントロールの管理
- Policy Model のログ ファイルの管理
- その他の管理タスクの実行

注: sepmd の詳細については、「*リファレンス ガイド*」を参照してください。

リモート PMDB の管理

CA Access Control には、pmd 環境で使用できるさまざまな selang コマンドも用意されています。 これらのコマンドを使用して、PMDB をリモートで管理できます。

backuppmd

PMDB をバックアップします。

createpmd

PMDB を作成します。

deletepmd

PMDB を削除します。

findpmd

コンピュータ上のすべての PMDB の名前を表示します。

listpmd

PMDB に関する以下の情報を表示します。

- サブスクライバおよびそのステータス
- PMDB の説明およびそのステータス
- 更新ファイル内のコマンドおよび各コマンドのオフセット
- エラー ログの内容

pmd

以下の操作を実行できる PMDB 管理コマンドです。

- 使用不可のサブスクライバのリストからのサブスクライバの削除
- Policy Model のエラー ログの消去
- Policy Model サービスの開始と停止
- Policy Model のロックおよびロック解除
- 更新ファイルの切り捨て

restorepmd

バックアップ ファイルから PMDB をリストアします。

subs

以下の操作を実行できる PMDB サブスクリプション コマンドです。

- 親 PMDB への既存サブスクライバの追加
- 親 PMDB への新規サブスクライバの追加
- データベース (CA Access Control または別の PMDB) への親 PMDB の割り当て

subspmd

ローカル データベースに親 PMDB を割り当てます。

unsubs

PMDB からサブスクライバを削除します。

注: pmd 環境で使用できる selang コマンドの詳細については、「*selang リファレンス ガイド*」を参照してください。

アーキテクチャの依存関係

CA Access Control をデプロイするときは、環境の階層を考慮する必要があります。 多くのサイトで、ネットワークにはさまざまなアーキテクチャが採用されています。 trusted プログラムのリストなど、一部のポリシールールはアーキテクチャに依存します。 一方、ほとんどのルールは、システムのアーキテクチャに関係なく適用されます。

階層を使用すると、両方の種類のルールを適用できます。アーキテクチャに依存しないルールをグローバル データベースで定義し、そのグローバル データベースのサブスクライバ PMDB で、アーキテクチャに依存するルールを定義できます。

注: ルート PMDB とそのすべてのサブスクライバは、環境の物理的ニーズに応じて、同じコンピュータ上に存在することも、別々のコンピュータ上に存在することも可能です。

例: 2 層のデプロイ階層

以下の UNIX の例は、少し変更して Windows アーキテクチャにも適用できます。

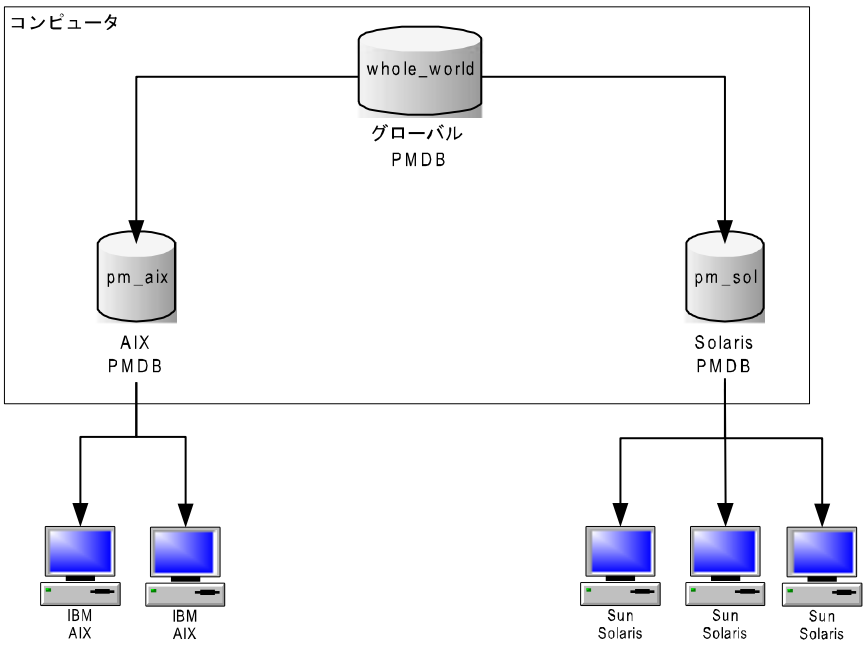
この例では、サイトは IBM AIX システムと Sun Solaris システムで構成されています。IBM AIX の trusted プログラムのリストは Sun Solaris でのリストとは異なるため、アーキテクチャの依存関係を考慮した PMDB が必要です。

複数アーキテクチャに対応した PMDB をセットアップするには、PMDB を以下のようにセットアップします。

1. whole_world という PMDB を定義し、ユーザ、グループ、およびアーキテクチャに依存しないその他のすべてのポリシーを格納します。
2. pm_aix という PMDB を定義し、IBM AIX 固有のすべてのルールを格納します。

3. pm_sol という PMDB を定義し、Sun Solaris 固有のすべてのルールを格納します。

   pm_aix および pm_solaris という PMDB は、whole_world という PMDB のサブスクライバです。 サイト内のすべての IBM AIX コンピュータは pm_aix のサブスクライバです。 サイト内のすべての Sun Solaris コンピュータは pm_sol のサブスクライバです。 この概念を以下の図に示します。



4. ユーザの追加や SURROGATE ルールの設定など、プラットフォームに依存しないコマンドを whole_world に入力すると、サイト内のすべてのデータベースが自動的に更新されます。

5. trusted プログラムを pm_aix に追加すると、IBM AIX コンピュータのみが更新されます。 Sun Solaris システムには影響はありません。

ポリシーの一元管理の方法

CA Access Control を使用すると、以下の 方法で 1 台のコンピュータから複数のデータベースを管理できます。

- **自動的なルール ベースのポリシー更新** -- 中央のデータベース（PMDB）で定義した通常のルールは、設定された階層内のデータベースに自動的に伝達されます。

  注: デュアル コントロールは、この方法でのみ使用できます。また、UNIX でのみ使用可能です。 自動的なルール ベース ポリシー更新のデュアル コントロールの詳細は、「*UNIX エンドポイント管理ガイド*」で説明しています。 また、自動的なルール ベース ポリシー更新の詳細は、「*Windows エンドポイント管理ガイド*」でも説明しています。

- **拡張ポリシー管理** -- デプロイしたポリシー（ルールの集合）は、ホストまたはホスト グループの割り当てに基づいて、すべてのデータベースに伝達されます。 また、ポリシーのデプロイ解除（削除）、デプロイのステータスやデプロイの偏差の表示を行うこともできます。 この機能を使用するには、追加のコンポーネントをインストールおよび設定する必要があります。

  注: 拡張ポリシー管理の詳細については、「*エンタープライズ管理ガイド*」を参照してください。

自動的なルール ベース ポリシー更新

中央データベースで単一ルール ポリシー更新（標準の selang ルール）を行うと、サブスクライバ データベースに自動的に伝達されます。 複数のコンピュータを同じデータベースのサブスクライバとし、そのデータベースを別のデータベースのサブスクライバとすることによって、階層を作成できます。 インストール後に、自動的なルール ベース ポリシー更新を環境に設定します。

注: このポリシー管理方法は、単一ルール ポリシー更新を階層全体に伝達することだけに制限されます。 その他の機能を使用するには、拡張ポリシー管理およびレポートを実装する必要があります。

自動的なルール ベース ポリシー更新のしくみ

環境に自動的なルール ベース ポリシー更新を設定すると、中央データベースで定義した各ルールは、以下の方法ですべてのサブスクライバに自動的に伝達されます。

1. 少なくとも 1 つのサブスクライバを持つ任意の PMDB にルールを定義します。
2. PMDB がすべてのサブスクライバ データベースにコマンドを送信します。
3. 伝達されたコマンドをサブスクライバ データベースが適用します。
   a. サブスクライバ データベースから応答がない場合、PMDB はサブスクライバ データベースが更新されるまで、定期的に（デフォルトでは 30 分間隔）コマンドを送信し続けます。
   b. サブスクライバ データベースから応答があっても、コマンドの適用が拒否された場合、PMDB はこのコマンドを Policy Model のエラー ログ (P. 196)に記録します。
4. サブスクライバ データベースが別のサブスクライバの親である場合は、サブスクライバ データベースはそのサブスクライバにコマンドを送信します。

例: 階層内のすべてのコンピュータからユーザを削除する

rmusr コマンドによってユーザが PMDB から削除されると、同じ rmusr コマンドがすべてのサブスクライバ データベースに送信されます。 このように、rmusr コマンドを 1 回実行すれば、さまざまな種類のコンピュータ上にある多数のデータベースからユーザを削除できます。

PMDB を使用した設定の伝達方法

Policy Model の設定を編集すると、新しい設定値が Policy Model のサブスクライバに伝達されます。

以下プロセスでは、設定の更新を Policy Model のサブスクライバに伝達する方法について説明します。

1. Policy Model の 1 つ以上の設定値を編集します。
2. Policy Model は、新しい設定値を仮想環境設定ファイルに書き込みます。

   注: 仮想環境設定ファイルには、audit.cfg ファイルの値は含まれません。Policy Model は、このファイルに加えた変更を仮想環境設定ファイルに書き込みません。
3. Policy Model は、そのサブスクライバに新しい設定値を伝達します。
4. selang コマンドは、新しい設定値で各サブスクライバを更新します。

仮想環境設定ファイル

各 Policy Model には、そのサブスクライバ用の設定値が含まれている仮想環境設定ファイルが存在します。仮想環境設定ファイルは、cfg_*configname* という名前で PMD ディレクトリに置かれます（*configname* は Policy Model 設定の名前）。

仮想環境設定ファイルには、audit.cfg ファイルに保持されている設定値は含まれていません。

新しいサブスクライバの設定方法

Policy Model は、既存の設定値で個々の新しいサブスクライバを設定します。 既存の設定値は、仮想環境設定ファイルに格納されます。

注: 仮想環境設定ファイルには、audit.cfg ファイルの設定値は格納されません。 新しいサブスクライバを作成する前に audit.cfg ファイルに加えた変更は、新しいサブスクライバに伝達されません。

以下のプロセスでは、Policy Model が新しいサブスクライバを設定する方法について説明します。

1. Policy Model の新しいサブスクライバを作成します。
2. Policy Model は、その仮想環境設定ファイルの値を読み取ります。
3. Policy Model は、その仮想環境設定ファイルの設定値を updates.dat ファイルに追加します。 updates.dat ファイルには、ポリシーに対するアクセス ルールも含まれています。
4. Policy Model は、新しいサブスクライバに updates.dat ファイルを送ります。
5. selang コマンドは、updates.dat ファイルの値を使用して新しいサブスクライバを設定します。

階層のセットアップ方法

CA Access Control は、Policy Model サービスを使用して、設定された階層全体にルール ベース ポリシー更新を伝達します。 複数の CA Access Control コンピュータを同じ PMDB にサブスクライブし、ある PMDB を別の PMDB にサブスクライブすることによって、階層を作成します。

PMDB の階層構造は、CA Access Control のインストール時にセットアップするのが最も簡単です。したがって、インストール作業を始める前に、どのように階層を構成するか考えておくことをお勧めします。親 PMDB とそのサブスクライバは互いに通信可能である必要があるため、PMDB 階層構造内のすべてのホストは同じネットワークに属している必要があります。つまり、親 PMDB とそのサブスクライバは、名前を指定して互いに接続できることが必要です。

注: CA Access Control のインストールの詳細については、「*実装ガイド*」を参照してください。

インストール時に行った設定を変更する場合、またはインストール時に PMDB 構造を作成しなかった場合は、いつでも PMDB の設定を変更または作成することができます。 これは、以下のいずれかの方法で行います。

- CA Access Control エンドポイント管理を使用する
- sepmd ユーティリティを使用する

インストール後に、PMDB 階層を作成し、自動的なルール ベース ポリシー更新を有効にするには、以下の手順に従います。

1. マスタ PMDB を作成し、設定します。
2. （オプション）サブスクライバ PMDB を作成し、設定します。
3. サブスクライバの親 PMDB を定義します。このサブスクライバは「*エンドポイント*」と呼ばれます。

サブスクライバの更新

サブスクライバを更新すると、Policy Model は以下のアクションを実行します。

1. Policy Model からサブスクライバ名が追加または削除される場合、そのサブスクライバの名前を完全修飾しようとします。
2. PMDB サービスの sepmdd が、サブスクライバ データベースの更新を試みます。
3. 制限時間が経過した時点でサブスクライバを更新できなかった場合、サービスはそのサブスクライバの更新処理を省略して、サブスクライバ リストにある残りのサブスクライバの更新を試みます。
4. sepmdd は、サブスクライバ リストの 1 回目のスキャンが終了した後、2 回目のスキャンを実行します。2 回目のスキャンでは、1 回目のスキャンで更新できなかったサブスクライバの更新を試みます。

注: サブスクライバへの更新情報の伝達時に PMDB でエラーが発生すると、sepmdd デーモンによって Policy Model のエラー ログ ファイル (P. 196)にエントリが作成されます。 このファイル（ERROR_LOG）は PMDB ディレクトリ (P. 182)に保存されます。

Policy Model データベースの更新

PMDB が格納されているコンピュータで操作を行っても、PMDB 自体は自動的に更新されません。 PMDB を更新するには、PMDB をターゲット データベースとして指定する必要があります。

PMDB は、selang または CA Access Control エンドポイント管理を使用して指定できます。 selang を使用してターゲット データベースを指定するには、selang コマンド シェルで hosts コマンドを使用します。

```
hosts pmd_name@pmd_host
```

これで、指定した Policy Model データベースがすべての selang コマンドで更新されます。次に、このコンピュータおよびすべてのサブスクライバ コンピュータ上のアクティブなデータベースにコマンドが自動的に伝達されます。

例: ターゲット PMDB を指定する

ターゲット データベースを myPMD_host の policy1 に設定するには、以下のコマンドを使用します。

```
hosts policy1@myPMD_host
```

ここで、newusr コマンドを入力すると、新規ユーザは policy1 データベースに追加される以外に、このコンピュータおよびすべてのサブスクライバ コンピュータ上のアクティブ データベースにも追加されます。

更新ファイルのクリーンアップ

sepmd ユーティリティは、受信した各更新情報を updates.dat ファイルに自動的に書き込みます。 このファイルのサイズが大きくなりすぎないように、処理済みの更新情報をファイルから定期的に削除することをお勧めします。

更新ファイルをクリーンアップするには、以下のコマンドを使用します。

```
sepmd -t pmdbName auto
```

sepmd は、まだ伝達されていない最初の更新エントリのオフセットを計算して、その前にあるすべての更新エントリを削除します

注: sepmd ユーティリティの詳細については、「*リファレンス ガイド*」を参照してください。

パスワードの伝達と同期

PMDB の階層を設定すれば、Windows のユーザ マネージャまたは CA Access Control 以外のソフトウェアでユーザ パスワードが変更された場合にも、この階層を使用してシステム全体でユーザ パスワードの同期を維持できます。

注: CA Access Control では、メインフレームのパスワード同期もサポートされています。

以下の手順に従います。

1. PMDB 階層を作成します。
2. ユーザまたは管理者がパスワードを変更する可能性がある各端末で、レジストリの passwd_pmd エントリの値に適切な親 PMDB の名前を入力します。

   ```
   HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥AccessControl¥passwd_pmd
   ```

   次に、PMDB からすべてのサブスクライバにパスワードの変更が伝達されます。

注: ユーザが定義されていないサブスクライバに PMDB からユーザ パスワードが送信された場合、設定は変更されず、ユーザはそのサブスクライバに対して未定義のままになります。

サブスクライバの削除

更新情報が特定のサブスクライバに伝達されないようにする場合は、そのサブスクライバを削除する必要があります。

サブスクライバを削除するには、以下の手順に従います。

1. コンピュータをサブスクライバ リストから削除します。

   ```
   sepmd -u PMDB_name computer_name
   ```

   コンピュータが Policy Model のサブスクライバ リストから削除されます。
2. サブスクライバ リストから削除したコンピュータで seosd を停止します。

   ```
   secons -s
   ```

   seosd サービスが停止されます。

3. サブスクライバ リストから削除したコンピュータで以下のレジストリ キーの parent_pmd レジストリの値を削除します。

   HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥AccessControl

   コンピュータは親 PMDB から更新情報を受け取らなくなります。

4. seosd を再起動します。

   サブスクライバ リストから削除したコンピュータ上のアクティブ データベースは、指定した PMDB のサブスクライバではなくなりました。

注: データベースが PMDB のサブスクライバから解除されると、PMDB はコマンドを送信しなくなります。

更新情報のフィルタ処理

1 つの PMDB を使用して、複数の異なるサブスクライバ データベースでデータのさまざまなサブセットを更新する場合は、サブスクライバ データベースにどのレコードを送信するかを定義する必要があります。

更新情報をフィルタ処理する方法

1. サブスクライバのサブセットの親として PMDB を設定します。
2. 親 PMDB のレジストリ キーの *Filter* レジストリのエントリを変更し、同じコンピュータで設定するフィルタ ファイルを参照するようにします。

   このように指定すると、フィルタ条件に該当するレコードのみがサブスクライバ データベースに更新情報として送信されます。

Policy Model のフィルタ ファイル

フィルタ ファイルは、各行に 6 つのフィールドを持つ複数の行で構成されます。 フィールドには以下の情報が含まれます。

- 許可または拒否されるアクセスの種類。

  例： EDIT または MODIFY

- 影響を受ける環境。

  例： AC またはネイティブ

- レコードのクラス。

  例： USER または TERMINAL

- ルールが適用される、クラスのオブジェクト。

  たとえば、User1、AuditGroup、または COM2 になります。

- レコードによって許可または取り消されるプロパティ。

  たとえば、フィルタ行の OWNER および FULL_NAME は、これらのプロパティを持つコマンドはすべてフィルタ処理されることを意味します。各プロパティは、「リファレンス ガイド」に記載されているとおりに、正確に入力する必要があります。

- 該当するレコードをサブスクライバ データベースに転送するかどうか。

  PASS または NOPASS

フィルタ ファイルの各行に以下のルールが適用されます。

- どのフィールドでも、アスタリスク（*）を使用して可能なすべての値を指定することができます。
- 同じレコードが複数の行に該当する場合は、最初の該当する行が使用されます。
- フィールドをスペースで区切ります。
- フィールドに複数の値がある場合は、値をセミコロンで区切ります。
- # で始まる行はコメント行とみなされます。
- 空白行は使用できません。

例: フィルタ ファイル

以下の例では、フィルタ ファイルの行について説明します。

CREATE	AC	USER	*	FULL_NAME;OBJ_TYPE	NOPASS
アクセス形式	環境	クラス	レコード名（* = すべての名前）	properties	処理方法

この例では、この行を指定したファイルの名前が Printer1_Filter.flt で、PMDB PM-1 のレジストリを編集してフィルタを C:¥Program Files¥CA¥AccessControl¥¥Printer1_Filter.flt と指定した場合、PMDB PM-1 は、FULL_NAME と OBJ_TYPE プロパティを指定してユーザを新規作成するレコードをサブスクライバに伝達しません。

Policy Model のエラー ログ ファイル

Policy Model のエラー ログ（発生順に書き込まれる）の例を以下に示します。

エラー テキスト	エラー カテゴリ
20 Nov 03 11:56:07 (pmdb1): fargo nu u5 0 Retry エラー: ログインできませんでした。(10068) エラー: 親ではないPMDB からの更新を受け付けることはできません。 (pmdb1@name.company.com) からの更新を受け付けることはできません。(10104)	環境設定エラー
20 Nov 03 19:53:17 (pmdb1): fargo nu u5 0 Retry エラー: 接続できませんでした。(10071) ホストに接続できません。(12296)	接続エラー
20 Nov 03 11:57:06 (pmdb1): fargo nu u5 560 Cont エラー: USER u5 の作成に失敗しました。(10028) すでに存在しています (-9) 20 Nov 03 11:57:06 (pmdb1): fargo nu u5 1120 Cont エラー: USER u5 の作成に失敗しました。(10028) すでに存在しています (-9)	データベース更新エラー

Policy Model のエラー ログはバイナリ フォーマットであるため、以下のコマンドを入力することでのみ表示できます。

ACInstallDir/bin sepmd -e *pmdname*

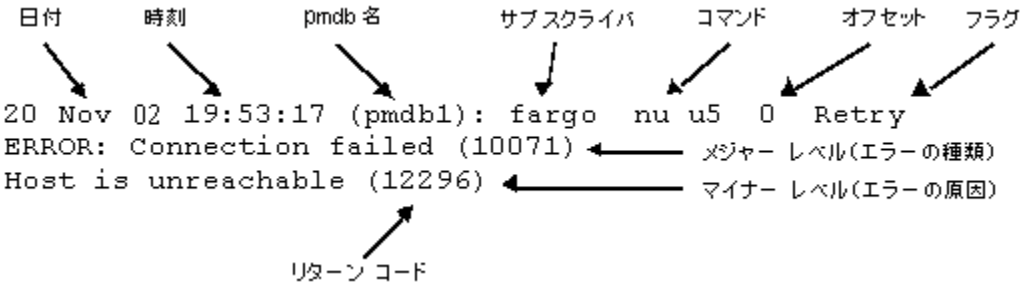
注: エラー ログは手動で削除しないでください（たとえば、UNIX の rm コマンドを使用した削除）。 ログを削除するには、以下のコマンドのみを使用してください。

ACInstallDir/bin sepmd -c *pmdname*

重要: CA Access Control r5.1 以降のバージョンでのエラー ログのフォーマットには、旧バージョンのフォーマットとの互換性はありません。sepmd を使用して、旧バージョンのエラー ログを処理することはできません。このバージョンのフォーマットにアップグレードする際に、旧エラー ログは ERROR_LOG.bak としてコピーされ、sepmdd を起動すると新しいログ ファイルが作成されます。

例: PMDB 更新のエラー メッセージ

以下の例は、標準的なエラー メッセージを示しています。



- 先頭行には必ず、日付、時刻、およびサブスクライバが表示されます。次に、エラーを発生させたコマンドが表示され、その後に、更新ファイル内の失敗した更新の位置を示すオフセット (10 進数) が続きます。最後のフラグは、PMDB が更新を自動的に再試行するか、または再試行せずに継続するかを示します。
- 2 行目は、メジャー レベル メッセージ（発生したエラーの種類）とリターン コードの例を示します。
- 3 行目は、マイナー レベル メッセージ（エラーの発生理由）とリターン コードの例を示します。

例: エラー メッセージ

1 つのコマンドによって、複数のエラーが生成および表示される場合があります。 また、エラーは、メジャー レベル メッセージ、マイナー レベル メッセージ、またはその両方で構成される場合があります。

以下のエラーには、メッセージ レベルが 1 つしかありません。

```
Fri Dec 29 10:30:43 2003 CIMV_PROD:リリースに失敗しました。 リターン コード = 9241
```

このメッセージは、すでに使用可能なサブスクライバのリリースを sepmd pull が試みた場合に表示されます。

Policy Model のネイティブ リポジトリ

PMDB には、ネイティブ環境のすべての種類のユーザおよびグループ オブジェクトを保存できます。このような情報を PMDB に保存すると、show コマンド（show user または show group）を使用して、オブジェクトに関する情報を取得できます。 返されるオブジェクトは、Windows サブスクライバまたは UNIX サブスクライバで定義されている実際のオブジェクトのイメージです。

Policy Model への接続後に、ユーザは以下の環境のいずれかを選択できます。

- AC
- Native
- NT
- UNIX
- Config

注: Native を選択すると、Windows オペレーティング システムで作業している場合は Windows と同様に、UNIX オペレーティング システムで作業している場合は UNIX と同様に機能します。

ネイティブ環境のリポジトリを使用するには、以下のコマンドを使用します。

- selang のプロンプトで以下のコマンドを入力します。

  ```
  env NT; find
  ```

  このコマンドを実行すると、ネイティブ環境のすべてのオブジェクトの種類が表示されます

  注: これらのオブジェクトの種類の詳細については、「*リファレンス ガイド*」の Windows 環境のクラスとプロパティの説明を参照してください。

- NT および Active Directory の USER プロパティの一覧を取得するには、以下のコマンドを入力します。

  ```
  env NT; ruler user
  ```

- NT および Active Directory の GROUP プロパティの一覧を取得するには、以下のコマンドを入力します。

  ```
  env NT; ruler group
  ```

Policy Model が別の（親）Policy Model のサブスクライバである場合、この Policy Model は伝達により親からのデータを受け取り、このデータを参照および変更できるように、すべてのユーザ プロパティとグループ プロパティをデータベースに保存します。

注: 詳細については、「*リファレンス ガイド*」の sepmd ユーティリティの説明を参照してください。

Policy Model のバックアップ

PMDB をバックアップする場合、Policy Model データベースのデータを別のディレクトリにコピーします。 これには以下のデータが含まれます。

- ポリシー情報
- Policy Model のサブスクライバのリスト
- 設定
- レジストリ エントリ
- updates.dat ファイル

別のプラットフォーム、オペレーティング システム、または CA Access Control バージョンを使用するバックアップ ファイルから PMDB をリストアすることはできません。 同じプラットフォーム、オペレーティング システム、および CA Access Control バージョンが動作するホストに Policy Model をバックアップしてください。

sepmd を使用した PMDB のバックアップ

PMDB のバックアップでは、Policy Model データベースのデータを指定のディレクトリにコピーします。PMDB のバックアップ ファイルは、安全な場所、できれば CA Access Control アクセス ルールで保護された場所に保存してください。

PMDB をローカル ホストにバックアップする場合は、sepmd ユーティリティを使用します。 また、selang コマンドを使って PMDB をリモート ホストにバックアップすることもできます。

注: PMDB は再帰的にバックアップできます。再帰的なバックアップでは、階層内のすべての PMDB が指定のホストにバックアップされ、バックアップがそのホストに移動してもサブスクリプションが引き続き機能するように PMDB サブスクライバが変更されます。再帰的なバックアップは、マスタと子の PMDB が同じホスト上で展開される場合にのみ使用できます。

sepmd を使用して PMDB をバックアップする方法

1. 以下のコマンドを使用して、PMDB をロックします。

   ```
   sepmd -bl pmdb_name
   ```

   PMDB はロックされるため、サブスクライバにコマンドを送信できなくなります。

2. 以下のいずれかの操作を行います。

   - 以下のコマンドを使って PMDB をバックアップします。

     ```
     sepmd -bh pmdb_name [destination_directory]
     ```

   - 以下のコマンドを使って PMDB データベースを再帰的にバックアップします。

     ```
     sepmd -bh pmdb_name [destination_directory] [backup_host_name]
     ```

   注: ユーザが送信先ディレクトリを指定しない場合、バックアップは以下のディレクトリに保存されます。

   ACInstallDir¥data¥policies_backup¥pmdb_name

3. 以下のコマンドを使って、PMDB のロックを解除します。

   ```
   sepmd -ul pmdb_name
   ```

   PMDB のロックが解除され、サブスクライバにコマンドを送信できるようになります。

selang を使用した PMDB のバックアップ

PMDB のバックアップでは、Policy Model データベースのデータを指定のディレクトリにコピーします。PMDB のバックアップ ファイルは、安全な場所、できれば CA Access Control アクセス ルールで保護された場所に保存してください。

selang コマンドを使用して、PMDB をローカル ホスト、またはリモート ホストにバックアップできます。 ローカル ホストに PMDB をバックアップする場合は、sepmd ユーティリティも使用できます。

注: PMDB は再帰的にバックアップできます。再帰的なバックアップでは、階層内のすべての PMDB が指定のホストにバックアップされ、バックアップがそのホストに移動してもサブスクリプションが引き続き機能するように PMDB サブスクライバが変更されます。再帰的なバックアップは、マスタと子の PMDB が同じホスト上で展開される場合にのみ使用できます。

selang を使用して PMDB をバックアップする方法

1. （オプション）selang を使用してリモート ホストから PMDB に接続している場合は、以下のコマンドを使って PMDB ホストに接続します。

   ```
   host pmdb_host_name
   ```

2. 以下のコマンドを使用して、PMD 環境に移動します。

   ```
   env pmd
   ```

3. 以下のコマンドを使用して、DMS をロックします。

   ```
   pmd pmdb_name lock
   ```

   PMDB はロックされるため、サブスクライバにコマンドを送信できなくなります。

4. 以下のコマンドを使用して、DMS データベースをバックアップします。

   ```
   backuppmd pmdb_name [destination(destination_directory)] [hir_host(host_name)]
   ```

   注: ユーザが送信先ディレクトリを指定しない場合、バックアップは以下のディレクトリに保存されます。

   ACInstallDir¥data¥policies_backup¥pmdbName

5. 以下のコマンドを使って、PMDB のロックを解除します。

   ```
   pmd pmdb_name unlock
   ```

   PMDB のロックが解除され、サブスクライバにコマンドを送信できるようになります。

Policy Model のリストア

Policy Model をリストアする場合、CA Access Control は指定のディレクトリにバックアップ PMDB ファイルをコピーします。 元の PMDB ファイルのデータが、以下を含めてすべて新しい PMDB ディレクトリにコピーされます。

- ポリシー情報
- Policy Model のサブスクライバのリスト
- 設定
- レジストリ エントリ
- updates.dat ファイル

コピー先ディレクトリに既存の PMBD がある場合、CA Access Control は既存のファイルを削除してからリストア ファイルをそのディレクトリにコピーします。

別のプラットフォーム、オペレーティング システム、または CA Access Control バージョンを使用するバックアップ ファイルから PMDB をリストアすることはできません。 同じプラットフォーム、オペレーティング システム、および CA Access Control バージョンが動作するホストに Policy Model をバックアップしてください。

PMDB のリストア

任意の PMDB をリストアすると、CA Access Control はその PMDB のバックアップ ファイルから指定のディレクトリ上へデータをコピーします。 CA Access Control はリストア処理を行う端末上で実行されている必要があります。

注: PMDB を別の端末にバックアップおよびリストアする場合、PMDB はリストアされた PMDB データベースにあるターミナル リソースの更新を、自動的には行いません。 新しいターミナル リソースはリストアされた PMDB に追加する必要があります。 新しいターミナル リソースを追加するには、リストアされた PMDB を停止し、*selang -p pmdb* コマンドを実行して、さらにリストアされた PMDB を起動します。

任意の PMDB をリストアするには、以下の*いずれか*を PMDB をリストアする端末上で実行してください。

- sepmd -restore ユーティリティ
- selang restore pmd コマンド

注: sepmd ユーティリティの詳細については「*リファレンス ガイド*」を参照してください。 selang コマンドの詳細については、「selang リファレンス ガイド」を参照してください。

PMDB と Unicenter の統合

PMDB を Unicenter TNG と統合すると、PMDB を使用してルールを作成することができます。このルールでは、さまざまな Unicenter TNG コンポーネント（コマンド プロセッサ、イベント管理、負荷管理など）によって操作されることがないよう Unicenter TNG オブジェクトを保護します。

この統合は手動で実行する必要があります。

PMDB を Unicenter TNG と統合するには、以下の手順に従います。

1. PMDB を作成します。
2. 以下のコマンドを使用して、Unicenter セキュリティオプションを PMDB に移行します。

   `MigOpts` *pmdb-name*

   pmdb-name には、PMDB の名前を指定します。

   注: この手順が必要になるのは、Unicenter セキュリティを使用し、かつ CA Access Control のインストール中に［Unicenter 統合］の［Security Data Migration］を選択した場合のみです。 Unicenter セキュリティを使用しなかった場合は、セキュリティ オプションを設定していないので、PMDB に移行する必要のあるオプションはありません。
3. 以下のコマンドを使用して、ユーザ定義の Unicenter TNG のアセットタイプに関するクラスを作成します。

   `defclass.bat.` *pmdb-name*

   pmdb-name には、PMDB の名前を指定します。

   注: この手順は、Unicenter セキュリティを使用し、かつユーザ定義のアセット タイプを作成した場合にのみ実行する必要があります。 CA Access Control のインストール時に Unicenter 統合を選択した場合、Unicenter TNG のアセット タイプは新しい PMDB を作成するたびに自動的に定義されます。

メインフレームのパスワード同期

CA Access Control では、CA Access Control を実行している Windows または UNIX マシンと、CA Top Secret、CA ACF2、または RACF セキュリティ製品（および CA Common Services CAICCI パッケージ）を実行しているメインフレームとのパスワード同期をサポートしています。 同期は、CA Access Control の標準のパスワード Policy Model 方式によって実現します。

メインフレームのユーザがパスワードを変更するたびに、パスワード Policy Model 階層内のすべてのマシンにその変更が伝達されます。

メインフレームのパスワード同期の前提条件

TNG/TND/NSM がインストールされているサーバで、メインフレームのパスワード同期機能を使用するには、TNG/TND/NSM 修正プログラムの T129430 があらかじめ適用されている必要があります。この修正プログラムの入手方法については、弊社のテクニカル サポートにお問い合わせください。

第 11 章: 包括的なセキュリティ機能

このセクションには、以下のトピックが含まれています。

メンテナンス モードの保護（サイレント モード）

CA Access Control にはメンテナンス モードがあります。これは、サイレント モードともいい、CA Access Control サービスがメンテナンスのために停止しているときに保護を提供するモードです。 このモードの CA Access Control では、これらのサービスが停止している間、イベントは拒否されます。

CA Access Control は、稼動している場合には、セキュリティを脅かすイベントをインターセプトして、イベントを許可するかどうかをチェックします。 メンテナンス モードをアクティブにしないと、CA Access Control サービスが停止している間、すべてのイベントが許可されます。 メンテナンス モードがアクティブであると、CA Access Control サービスが停止している間、イベントは拒否されて、システムのメンテナンス中はユーザ アクティビティが停止します。

メンテナンス モードは調整することができます。デフォルトでは、無効です。

CA Access Control セキュリティ サービスが停止している間は、以下のような状態になります。

- メンテナンス モードがアクティブである場合、セキュリティを脅かすイベントはすべて拒否されます（ただし、特別な場合、およびメンテナンス ユーザによって実行されるイベントは除きます）。
- メンテナンス モードが無効である場合、CA Access Control は介入せず、実行はオペレーティング システムに渡されます。

メンテナンス モードがアクティブでセキュリティが停止しているときに拒否されたイベントは、監査ログ ファイルに記録されません。

メンテナンス モードを有効にするには、以下の手順に従います。

1. CA Access Control サービスが停止していることを確認します。
2. レジストリ エディタを使用して、以下のレジストリ キーに移動します。

   ¥HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥FsiDrv

   以下の値を変更します。

   - SilentModeEnabled = 1
   - SilentModeAdmins = *special_admins*

     special_admins 変数では、CA Access Control サービスが停止している間、コンピュータにアクセスすることができるユーザ名のリストを定義します。

     ユーザごとに改行します。 指定に関係なく、*SYSTEM* は常にメンテナンス モード ユーザになります。

     注: Windows 2000 および Windows NT では、regedit を使用して SilentModeAdmins キーを編集できません。代わりに、Regedt32.exe を使用してください。
3. コマンド シェルから「seosd -start」コマンドを指定するか、Windows の［スタート］メニューのオプションを使用して、CA Access Control サービスを起動します。

CA Access Control サービスが停止している場合は、SilentModeAdmins レジストリ キー下にリストされているユーザのみにコンピュータへのアクセス権が付与されます。他のユーザは、アクティビティを試みても拒否されます。

ドライバのバイパス

一部のドライバを、CA Access Control の権限チェックなしで動作できるようにするには、これらのドライバに対してバイパスを定義します。 たとえば、アンチウイルス プログラムのドライバに対してバイパスを定義すれば、CA Access Control の権限チェックなしで、ファイルを開いてスキャンできるようになります。バイパスを設定しないと、ドライバと CA Access Control との間でデッドロックが発生する可能性があります。

注: Trend Micro™ PC-cillin Antivirus の現在のバージョンのバイパスは、すぐに使用できるように設定されています。

ドライバのバイパスの設定方法

1. BypassDriversCount レジストリ エントリ値を、バイパスの設定対象のドライバの数に設定します。

   このエントリは、CA Access Control レジストリの FsiDrv キーにあります。

   注: まず CA Access Control を停止してから、CA Access Control レジストリ エントリを変更する必要があります。

2. バイパスする各ドライバについて、以下の操作を行います。

   a. DriverName_*drvNumber* という名前の REG_SZ タイプのレジストリ エントリを作成します。

      最初のエントリは DriverName_0、最後のエントリは DriverName_*X* である必要があります。ここで、*X* は BypassDriversCount - 1 です。

   b. 各 DriverName_*drvNumber* エントリを編集して、その値がバイパスするドライバの名前になるようにします。

      この値はドライバのみの名前（thisdrv.sys など）である必要があります。

3. CA Access Control を再起動します。

   CA Access Control が再ロードされ、レジストリで定義したドライバがバイパスされます。

例: ドライバのバイパスによる互換性の問題の解決

この例では、バイパスするアンチウイルス製品（avDriverA.sys および avDriverB.sys）を定義して、アンチウイルス製品と CA Access Control との間の互換性の問題を解決します。 CA Access Control レジストリ ツリー内の FsiDrv キーの下で、ドライバ バイパスのレジストリ エントリを設定します。

```
HKLM¥SOFTWARE¥ComputerAssociates¥AccessControl¥FsiDrv
```

レジストリ エントリを以下のように設定します。

名前	タイプ	データ
BypassDriversCount	REG_DWORD	2
DriverName_0	REG_SZ	avDriverA.sys
DriverName_1	REG_SZ	avDriverB.sys

BypassDriversCount レジストリ エントリ値を 2 に設定すると、CA Access Control はバイパスする 2 つのドライバを探します。 各 DriverName_*drvNumber* レジストリ エントリ値は、バイパスするドライバを定義します。

ドライバ インターセプトの切り替え

CA Access Control フィルタ ドライバのインターセプトの有効化と無効化を切り替えられます。

注: インターセプトが無効な場合でも、フィルタ ドライバで実行されない CA Access Control 保護は適用されます。 これには、パスワード品質チェック、ログイン イベント、Windows サービス イベント、STOP などが含まれます。

インターセプトを有効にするには UseFsiDrv を 1 に、無効にするには 0 に設定します。

この環境設定は、CA Access Control レジストリの AccessControl セクションにあります。

このレジストリ値を変更した後は、CA Access Control サービスを再起動します。

CA Access Control カーネルによるインターセプトの無効化

カーネル レベルで、以下の CA Access Control インターセプトを無効にできます。

- ネットワーク インターセプト
- プロセス インターセプト
- レジストリ インターセプト
- ファイル インターセプト

ネットワーク、プロセス、レジストリ、およびファイルの各クラスが無効になっていて、カーネル アクティビティのインターセプトにこれらのクラスを使用していない場合でも、ネットワーク、プロセス、レジストリ、およびファイルのインターセプト処理コードは起動時に初期化され、実行時に稼働して、パフォーマンスに影響します。 パフォーマンスを向上させるために、起動時の 1 つ以上のインターセプトの初期化を無効にできます。

カーネル レベルで CA Access Control インターセプトを無効にする方法

1. REG_DWORD タイプの以下のレジストリ エントリを 1 つ以上作成し、1 つ以上のエントリの値を「1」に設定します。
   - DisableNetworkInterception - ネットワーク インターセプトを無効にします
   - DisableProcessInterception - プロセス インターセプトを無効にします
   - DisableRegistryInterception - レジストリ インターセプトを無効にします
   - DisableFileInterception - ファイル インターセプトを無効にします

   エントリは、以下のレジストリ キーで作成する必要があります。

   ```
   HKLM¥SYSTEM¥CurrentControlSet¥Services¥drveng¥Parameters
   ```
2. コンピュータを再起動します。

   CA Access Control は、無効化されたインターセプト タイプを初期化せずに、再ロードされます。

Stack Overflow Protection

スタック オーバーフロー防止機能（STOP）は、ハッカーがスタック オーバーフローを発生させ、それを利用してシステムに侵入するのを防止する機能です。 スタック オーバーフローによって、ハッカーは、リモートまたはローカルのシステムに対して、管理者としてあらゆるコマンドを何度でも実行できます。 ハッカーは、オペレーティング システムや他のプログラムのバグを利用して、スタック オーバーフローを発生させます。 これらの特殊なバグによって、ユーザはプログラム スタックを上書きできるようになり、次に実行されるコマンドが変更されます。

STOP は、コンピュータ上の各アプリケーションに対する重要なオペレーティング システム コールをインターセプトすることで動作します。 各コールは、基本分析が実施された後、疑わしい場合は詳細分析に送られます。 詳細分析の実施には、STOP の設定ファイルとシグネチャ ファイルのデータが使用されます。

STOP の有効化

STOP を使用して、ハッカーによるスタック オーバーフローを悪用したシステム侵入を防止することができます。STOP は、CA Access Control のインストール時に有効にできます。 または、手動で有効にすることもできます。

STOP を有効にするには、以下の手順に従います。

1. 以下のコマンドを入力します。

   ```
   secons -s
   ```

   CA Access Control が停止します。

2. STOP *OperationMode* レジストリ エントリを 1 に設定します。

   このレジストリ エントリは以下のキーにあります。

   ```
   HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥Instrumentation¥PlugIns¥StopPlg
   ```

   CA Access Control が起動すると、STOP モジュールがロードされて、コンピュータで STOP が有効になります。

3. （オプション）STOP の環境設定を変更するには、以下のキーのレジストリ エントリを使用します。

   HKEY_LOCAL_MACHINE¥SOFTWARE¥ComputerAssociates¥AccessControl¥Instrumentation¥PlugIns¥StopPlg

   HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥STOP

   注: STOP のレジストリ設定については、「*リファレンス ガイド*」を参照してください。

4. 以下のコマンドを入力します。

   ```
   seosd -start
   ```

   CA Access Control が起動します。

シグネチャ ファイルの更新内容を受け取るための STOP の設定

環境内のすべてのコンピュータに、スタック オーバーフローの防止に必要な最新の STOP 情報が設定されていることを確認できます。 これを実行するには、中央のコンピュータにある STOP シグネチャ ファイルを更新し、このファイルを定期的に取得するようにコンピュータをセットアップします。

シグネチャ ファイルの更新内容を受け取るように STOP を設定するには、以下の手順に従います。

1. 以下のコマンドを入力します。

   ```
   secons -s
   ```

   CA Access Control が停止します。

2. *STOPSignatureBrokerName* レジストリ エントリを、シグネチャ ファイルの取得元のコンピュータのホスト名に設定します。

   このレジストリ エントリは以下のキーにあります。

   HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥STOP

   CA Access Control を起動すると、CA Access Control は指定されたコンピュータから（定義された間隔で）STOP シグネチャ ファイルを取得します。

3. *STOPUpdateInterval* レジストリ エントリを、シグネチャ ファイルを更新する間隔に設定します。

   CA Access Control は、指定されたコンピュータから、シグネチャ ファイルを指定された間隔で取得します。

4. （オプション）STOP の環境設定を調整するには、以下のキーのレジストリ エントリを使用します。

   ```
   HKEY_LOCAL_MACHINE¥Software¥ComputerAssociates¥AccessControl¥STOP
   ```

   注: STOP のレジストリ設定については、「*リファレンス ガイド*」を参照してください。

5. 以下のコマンドを入力します。

   ```
   seosd -start
   ```

   CA Access Control が起動します。

注: eACSigUpdate ユーティリティを使用することで、任意のホストからシグネチャ ファイルを取得することができます。 このユーティリティの詳細については、「*リファレンス ガイド*」を参照してください。

第 12 章: 設定

CA Access Control では、CA Access Control エンドポイントの設定をリモートで管理できます。 この場合は、CA Access Control エンドポイント管理または selang インタフェースを使用できます。

このセクションには、以下のトピックが含まれています。

設定

CA Access Control は、使用しているエンドポイントと Policy Model の設定を以下に保存します。

- Windows コンピュータ： Windows レジストリ
- UNIX コンピュータ： 初期設定（.ini）ファイル

注: 実行できる設定およびその設定の意味の詳細については、「*リファレンス ガイド*」を参照してください。

設定の変更

CA Access Control と Policy Models の動作を制御するには、設定を変更する必要があります。

設定を変更するには、以下の手順に従います。

1. CA Access Control エンドポイント管理 内で、以下の操作を実行します。
   a. ［設定］をクリックします。
   b. ［リモート設定］をクリックします。
   ［リモート設定］ページが表示されます。
2. 左側の［リモート設定］セクション ペインで、必要に応じて［設定］ツリーを展開して変更する設定が含まれているセクションを表示し、そのセクションをクリックします。
   ［セクション: *セクション名* システム トークン］ページが表示され、そのセクションに含まれるすべての設定が表示されます。
3. 必要に応じて設定を検索して編集し、［トークンの保存］をクリックします。
   変更した設定が保存されます。

監査設定の変更

CA Access Control が監査レコードを生成し、格納する方法を変更するには、監査設定ファイルの設定を変更する必要があります。 監査設定ファイルの設定を変更するには、selang コマンドを使用します。

監査設定を変更するには、以下の手順に従います。

1. （オプション）selang を使ってリモート ホストに接続するには、以下のコマンドを使用します。
   ```
   host host_name
   ```
2. 以下のコマンドを使って、config 環境に移動します。
   ```
   env config
   ```
3. editres config コマンドは、必要に応じて環境設定の変更に使用します。
   監査設定は変更されました。

例: 監査設定ファイルの変更

以下の例では、監査設定ファイルに 1 行追加します。

```
er CONFIG audit.cfg line+("FILE;*;Administrator;*;R;P")
```