APPENDIX A

# システム アーキテクチャについて

（注） IPS SSP を搭載した Cisco ASA 5585-X は、現在のところ、Cisco IPS 7.1 をサポートする唯一のプラットフォームです。現時点では、他に IPS バージョン 7.1 をサポートしている Cisco IPS センサーはありません。

（注） IPS SSP を搭載した Cisco ASA 5585-X は、ASA 8.2(4.4) 以上および ASA 8.4(2) 以上でサポートされています。ASA 8.3(*x*) では、サポートされていません。

この付録では、Cisco IPS のシステム アーキテクチャについて説明します。次の事項について説明します。



## Cisco IPS の目的

Cisco IPS の目的は、悪意のあるネットワーク アクティビティを検出および防止することです。Cisco IPS ソフトウェアは、アプライアンスとモジュールの 2 つのプラットフォームにインストールします。Cisco IPS には、管理アプリケーションとモニタリング アプリケーションが含まれています。IDM は、IPS の管理とモニタに使用できる、ネットワーク管理 JAVA アプリケーションです。IME は、IPS イベ

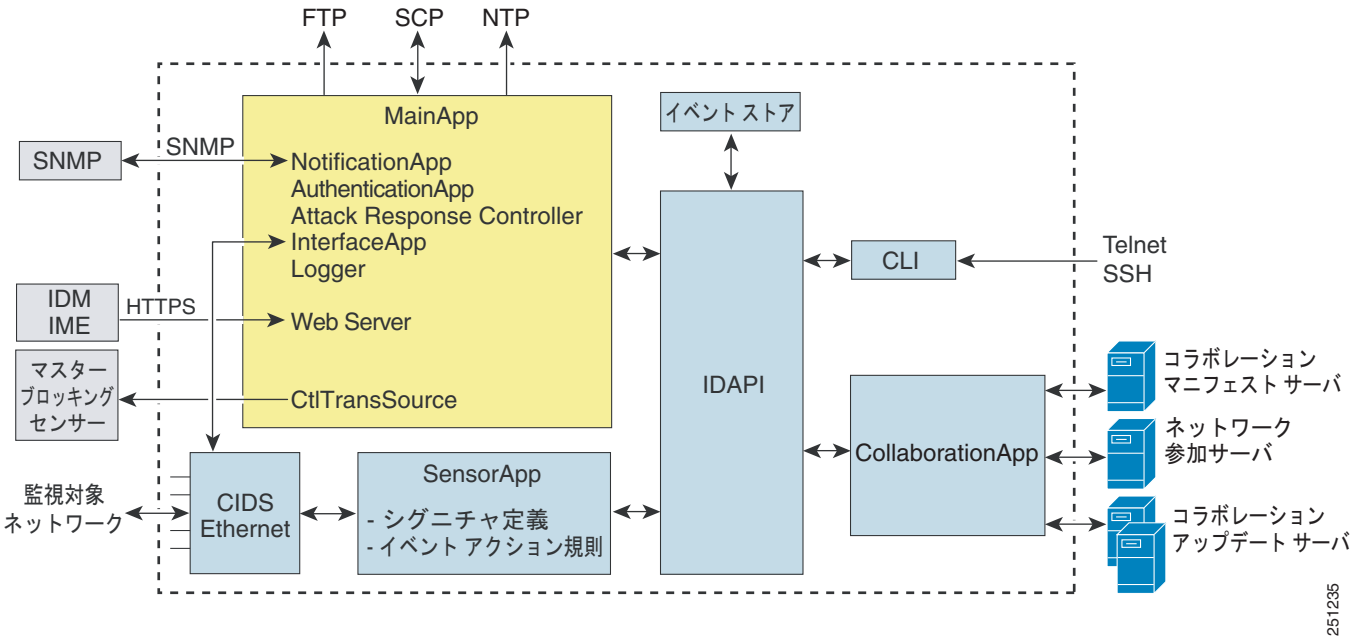
ントの表示に使用できる、IPS ネットワーク モニタリング JAVA アプリケーションです。IME には、IDM 設定コンポーネントも含まれています。IDM と IME は、HTTP または HTTPS を使用して IPS と通信し、コンピュータ上でホストされます。

# システム設計

Cisco IPS ソフトウェアは、Linux オペレーティング システム上で動作します。Linux OS を強化するために、不要なパッケージの削除、使用しないサービスの無効化、ネットワーク アクセスの制限、およびシェルへのアクセスの停止を行いました。

図 A-1 に、IPS ソフトウェアのシステム設計を示します。

**図 A-1　システム設計**




### 詳細情報

- MainApp の詳細については、「MainApp」（P.A-5）を参照してください。
- SensorApp の詳細については、「SensorApp」（P.A-22）を参照してください。
- CollaborationApp の詳細については、「CollaborationApp」（P.A-28）を参照してください。
- CLI の詳細については、「CLI」（P.A-30）を参照してください。

# システム アプリケーション

（注）　各アプリケーションには、それぞれ独自の XML 形式の構成ファイルがあります。

Cisco IPS ソフトウェアには、次のアプリケーションが含まれています。

- MainApp：システムの初期化、他のアプリケーションの起動および停止、OS の構成、およびプラットフォームのアップグレードを行います。次のコンポーネントで構成されています。
  - ctlTransSource（Control Transaction Server）：センサーで制御トランザクションを送信できます。Attack Response Controller（以前の名称は Network Access Controller）のマスター ブロッキング センサー機能をイネーブルにするために使用されます。
  - Event Store：IPS イベント（error、status、alert の各システム メッセージ）を格納するために使用され、CLI、IDM、IME、ASDM、または SDEE からアクセスできるインデックス付きストア。
  - InterfaceApp：バイパス設定と物理設定を処理し、対になったインターフェイスを定義します。物理設定とは、速度、デュプレックス、および管理状態です。
  - Logger：アプリケーションのすべてのログ メッセージをログ ファイルに書き込み、アプリケーションのエラー メッセージをイベント ストアに書き込みます。
  - Attack Response Controller（以前の名称は Network Access Controller）：alert イベントが発生したときのブロッキング機能を実装するためにリモート ネットワーク デバイス（ファイアウォール、ルータ、およびスイッチ）を管理します。ARC は、制御対象のネットワーク デバイス上で ACL を作成および適用するか、**shun** コマンドを使用します（ファイアウォール）。
  - NotificationApp：アラート ステータス イベントおよびエラー イベントによってトリガーされたときに SNMP トラップを送信します。NotificationApp は、パブリック ドメイン SNMP エージェントを使用します。SNMP GET は、センサーの全般的な健全性に関する情報を提供します。
  - Web サーバ（HTTP SDEE サーバ）：Web インターフェイスを持ち、複数のサーブレットを使用しながら SDEE プロトコルを介して他の IPS デバイスと通信して、IPS サービスを提供します。
  - AuthenticationApp：CLI、IDM、IME、ASDM、または SDEE のアクションを実行する許可がユーザに付与されているかどうかを確認します。
- SensorApp（分析エンジン）：パケットのキャプチャと分析を行います。
- CollaborationApp：IDAPI 制御トランザクション、セマフォ、共有メモリ、ファイル交換などのさまざまなプロセス間通信テクノロジーを使用して、MainApp および SensorApp とインターフェイスします。
- CLI：Telnet または SSH を通じてセンサーに正しくログインすると実行されるインターフェイス。CLI で作成されたすべてのアカウントは、CLI をアカウントのシェルとして使用します（サービス アカウントは例外。許可されるサービス アカウントは 1 つだけです）。使用できる CLI コマンドは、ユーザの権限によって異なります。

すべての Cisco IPS アプリケーションは、IDAPI という共通 API を通じて相互に通信します。リモート アプリケーション（他のセンサー、管理アプリケーション、およびサードパーティ ソフトウェア）は、SDEE プロトコルを通じてセンサーと通信します。

センサーには、次のパーティションがあります。

- アプリケーション パーティション：フル IPS システム イメージ。
- メンテナンス パーティション：IDSM2 のアプリケーション パーティションのイメージを再作成するために使用される、特殊な目的の IPS イメージ。メンテナンス パーティションのイメージを再作成すると、すべてのコンフィギュレーション設定が失われます。
- リカバリ パーティション：センサーのリカバリに使用される、特殊な目的のイメージ。リカバリ パーティションで起動すると、アプリケーション パーティションを完全に再作成することができます。ネットワーク設定は保存されますが、それ以外のすべての設定は失われます。

# ユーザ対話

Cisco IPS ソフトウェアとは、次の方法で対話します。

- デバイス パラメータの設定

  システムとその機能の初期設定を生成します。これは頻繁に実行する作業ではなく、通常は一度だけ実行します。システムには妥当なデフォルト値が設定されており、必要な変更の数が最小化されています。Cisco IPS は、CLI、IDM、IME、CSM、ASDM からか、SDEE を使用する他のアプリケーションから設定できます。

- 調整

  設定に対する変更はわずかで、主として Analysis Engine に対して行います。分析エンジンは、ネットワーク トラフィックをモニタするアプリケーションの部分です。ネットワークにシステムを初期インストールした後で、システムが効率的に動作し、有用な情報だけが生成されるようになるまで、システムを頻繁に調整できます。カスタム シグニチャの作成、機能のイネーブル化、またはサービス パックまたはシグニチャ アップデートの適用を行うことができます。Cisco IPS は、CLI、IDM、IME、CSM、ASDM からか、SDEE を使用する他のアプリケーションから調整できます。

- アップデート

  自動アップデートをスケジュールすることも、アプリケーションおよびシグニチャ データ ファイルに今すぐアップデートを適用することもできます。Cisco IPS は、CLI、IDM、IME、CSM、ASDM からか、SDEE を使用する他のアプリケーションからアップデートできます。

- 情報の取得

  CLI、IDM、IME、CSM、ASDM、CS MARS を介するか、SDEE を使用する別のアプリケーションを介して、システムからデータ（ステータス メッセージ、エラー、およびアラート）を取得できます。

# セキュリティ機能

Cisco IPS には次のセキュリティ機能があります。

- ネットワーク アクセスは、アクセスを明確に許可されたホストに制限されます。
- Web サーバ、SSH、および SCP または Telnet を介して接続しようとするすべてのリモート ホストが認証されます。
- デフォルトでは、Telnet アクセスはディセーブルになります。Telnet をイネーブルにすることを選択できます。
- デフォルトでは、SSH アクセスはイネーブルです。

- FTP サーバは、センサー上で実行されません。リモートでファイル コピーする場合は、SCP を使用できます。
- デフォルトでは、Web サーバは、TLS または SSL を使用します。TLS および SSL をディセーブルにすることを選択できます。
- 不要なサービスはディセーブルになっています。
- CISCO-CIDS-MIB 内では、Cisco MIB Police で必要とされる SNMP set だけが許可されます。パブリック ドメイン SNMP エージェントによって実装された OID は、MIB によって指定された場合、書き込み可能です。

# MainApp

ここでは、MainApp について説明します。次の事項について説明します。



## MainApp について

MainApp には、SensorApp と CLI 以外のすべての IPS コンポーネントが含まれています。起動時にオペレーティング システムによってロードされ、SensorApp をロードします。MainApp は、次に、次のサブシステム コンポーネントを起動します。

- 認証
- ロガー
- ARC
- Web サーバ
- 通知（SNMP）
- 外部製品インターフェイス
- インターフェイス マネージャ
- イベント ストア
- 健全性とセキュリティのモニタリング

# MainApp の役割

MainApp には、次の役割があります。

- シスコでサポートしているハードウェア プラットフォームの検証
- ソフトウェア バージョンおよび PEP 情報の報告
- IPS コンポーネントの起動、停止、バージョンの報告
- ホスト システム設定の指定
- システム クロックの管理
- イベント ストアの管理
- ソフトウェア アップグレードのインストールとアンインストール

（注）　Cisco IPS では、シグニチャおよびシグニチャ エンジンのアップデートを Cisco.com から MainApp で自動的にダウンロードできます。

- オペレーティング システムのシャットダウンまたはリブート

MainApp は、**show version** コマンドへの応答として次の情報を表示します。

- センサーのビルド バージョン
- MainApp のバージョン
- 実行中の各アプリケーションのバージョン
- インストールされている各アップグレードのバージョンおよびタイムスタンプ
- インストールされている各アップグレードの次のダウングレード バージョン
- プラットフォームのバージョン
- 他のパーティションにあるセンサーのビルドのバージョン

MainApp では、ホスト統計情報の収集と、健全性およびセキュリティ モニタリング状況の報告も行います。

# イベント ストア

ここでは、イベント ストアについて説明します。次の事項について説明します。

- 「イベント ストアについて」（P.A-6）
- 「イベント データの構造」（P.A-7）
- 「IPS イベント」（P.A-8）

## イベント ストアについて

各 IPS イベントは、タイムスタンプ、および一意で単純な昇順の ID と共にイベント ストアに格納されます。このタイムスタンプをプライマリ キーとして使用することにより、固定サイズのインデックス化されたイベント ストアにイベントをインデックス付けします。循環式のイベント ストアが設定済みサイズに達すると、最も古いイベントを上書きして新しいイベントが格納されます。イベント ストアに alert イベントを書き込むアプリケーションは SensorApp だけです。log、status、error イベントは、すべてのアプリケーションがイベント ストアに書き込みます。

固定サイズのインデックス化されたイベント ストアでは、時刻、タイプ、プライオリティ、および限られた数のユーザ定義属性に基づいて、シンプルなイベントのクエリーを実行できます。イベントのそれぞれに low、medium、または high のプライオリティを割り当てると、1 回のイベント クエリーで目的のイベント タイプのリスト、侵入イベントのプライオリティ、および時間範囲を指定できます。

表 A-1 に、いくつかの例を示します。

**表 A-1　IPS イベントの例**

| IPS イベント タイプ | 侵入イベント プライオリティ | 開始タイムスタンプ値 | 停止タイムスタンプ値 | 意味 |
|---|---|---|---|---|
| status | — | 0 | 最大値 | 格納されているすべての status イベントを取得します。 |
| error<br>status | — | 0 | 65743 | 時刻 65743 よりも前に格納されたすべての error および status イベントを取得します。 |
| status | — | 65743 | 最大値 | 時刻 65743 以降に格納された status イベントを取得します。 |
| intrusion<br>attack response | low | 0 | 最大値 | プライオリティ low で格納されているすべての intrusion および attack response イベントを取得します。 |
| attack response<br>error<br>status<br>intrusion | medium<br>high | 4123000000 | 4123987256 | 時刻が 4123000000 から 4123987256 の間に格納された、プライオリティが medium または high の attack response、error、status、および intrusion イベントを取得します。 |

イベント ストアのサイズは、センサーが IPS イベント コンシューマに接続されていないときに IPS イベントをバッファリングするのに十分な大きさです。バッファリングが十分であるかどうかは、ユーザの要件と、使用するノードの能力に依存します。循環バッファ内の最も古いイベントは、最新のイベントによって置き換えられます。

## イベント データの構造

さまざまな機能ユニットが、次の 7 種類のデータをやり取りします。

- intrusion イベント：SensorApp によって生成されます。intrusion イベントはセンサーが検出します。
- error イベント：ハードウェアまたはソフトウェアの不具合によって発生します。
- status イベント：設定がアップデートされたなどの、アプリケーションのステータスの変更を報告します。
- control transaction log イベント：センサーが制御トランザクションの結果を記録します。
- attack response イベント：ブロック要求などの ARC 向けアクション。
- debug イベント：アプリケーションのステータスの変更に関するきわめて詳細なレポートで、デバッグに使用されます。

- 制御トランザクション データ：制御トランザクションに関連するデータ。たとえば、アプリケーションの診断データ、セッション ログ、およびアプリケーションとやり取りされる設定データ。

この全 7 種類のデータを *IPS データ*と総称します。6 つのイベント タイプ（intrusion、error、status、control transaction log、network access、debug）は特徴が似ており、*IPS イベント*と総称されます。IPS イベントは、IPS を構成する数種類のアプリケーションによって生成され、他の IPS アプリケーションに受信されます。IPS イベントには、次のような特徴があります。

- IDS イベントを生成するように設定されているアプリケーション インスタンスによって、自然発生的に生成されます。特定のイベントを生成するように他のアプリケーション インスタンスから要求されることはありません。
- 特定の宛先はありません。1 つまたは複数のアプリケーションによって格納され、その後、取得されます。

制御トランザクションは、次のタイプの要求に関係します。

- アプリケーション インスタンスの設定データに対する更新の要求
- アプリケーション インスタンスの診断データの要求
- アプリケーション インスタンスの診断データのリセット要求
- アプリケーション インスタンスを再起動する要求
- ブロック要求などの ARC 向け要求

制御トランザクションには、次のような特徴があります。

- 常に、1 つの応答を伴う 1 つの要求によって構成されます。

  要求と応答には、任意の量のデータが関連付けられる可能性があります。すべての応答には、少なくとも肯定応答または否定応答が含まれます。
- ポイントツーポイントのトランザクションです。

  制御トランザクションは、1 つのアプリケーション インスタンス（発信側）からもう 1 つのアプリケーション インスタンス（応答側）に送信されます。

IPS データは、XML 形式で XML ドキュメントとして表されます。システムには、ユーザ設定可能なパラメータがいくつかの XML ファイルで格納されます。

## IPS イベント

IPS アプリケーションは、ある種のできごとが発生したことを報告するために IPS イベントを生成します。イベントは、SensorApp が生成するアラートやアプリケーションが生成するエラーなどのデータです。イベントは、イベント ストアというローカル データベースに格納されます。

5 種類のイベントがあります。

- evAlert：ネットワーク アクティビティによってシグニチャがトリガーされたことを報告する alert イベント メッセージ
- evStatus：IPS アプリケーションのステータスとアクションを報告する status イベント メッセージ
- evError：応答アクションの試行中に発生したエラーを報告する error イベント メッセージ
- evLogTransaction：各センサー アプリケーションによって生成された制御トランザクションを報告する log transaction メッセージ
- evShunRqst：ARC がいつブロック要求を発行したかを報告するブロック要求メッセージ

status メッセージおよび error メッセージは、CLI、IME、および ASDM を使用して表示できます。

SensorApp および ARC は、応答アクション（TCP リセット、IP ロギングの開始と停止、ブロッキングの開始と停止、トリガー パケット）をステータス メッセージとしてログに記録します。

# NotificationApp

NotificationApp によって、センサーでは、アラートおよびシステム エラー メッセージを SNMP トラップとして送信できます。イベント ストア内のイベントをサブスクライブして、SNMP MIB に変換し、パブリック ドメイン SNMP エージェントを介して目的の場所に送信します。NotificationApp は、set および get の送信をサポートしています。SNMP GET では、センサーの健全性に関する基本情報を取得できます。

スパース モードでは、NotificationApp は、evAlert イベントから次の情報を送信します。

- 発信者情報
- イベント ID
- イベントの重大度
- 時刻（UTC および現地時間）
- シグニチャ名
- シグニチャ ID
- サブシグニチャ ID
- 参加者情報
- アラームの特性

詳細モードでは、NotificationApp は、evAlert イベントから次の情報を送信します。

- 発信者情報
- イベント ID
- イベントの重大度
- 時刻（UTC および現地時間）
- シグニチャ名
- シグニチャ ID
- サブシグニチャ ID
- バージョン
- サマリー
- インターフェイス グループ
- VLAN
- 参加者情報
- アクション
- アラームの特性
- シグニチャ
- IP ログの ID

NotificationApp は、定義されたフィルタに従って、トラップとして送信する evError イベントを決定します。エラーの重大度（error、fatal、warning）に基づいてフィルタリングできます。
NotificationApp は、evError イベントから次の情報を送信します。

- 発信者情報
- イベント ID
- イベントの重大度

- 時刻（UTC および現地時間）
- エラー メッセージ

NotificationApp は、センサーからの次の一般的な健全性情報およびシステム情報の GET をサポートします。

- パケット損失
- パケット拒否数
- 生成されたアラーム数
- FRP 内のフラグメント数
- FRP 内のデータグラム数
- 初期状態の TCP ストリーム数
- 確立状態の TCP ストリーム数
- 終了状態の TCP ストリーム数
- システム内の TCP ストリーム数
- 再構成用にキューイングされた TCP パケット数
- アクティブ ノードの合計数
- 両方の IP アドレスと両方のポートでキー付けされた TCP ノード数
- 両方の IP アドレスと両方のポートでキー付けされた UDP ノード数
- 両方の IP アドレスでキー付けされた IP ノード数
- センサー メモリの臨界状態
- インターフェイスのステータス
- コマンド/コントロール パケットの統計情報
- フェールオーバーの状態
- システムの稼動時間
- CPU 使用率
- システムのメモリ使用量
- PEP

**（注）** 一部の IPS プラットフォームでは、PEP をサポートしていません。

NotificationApp には次の統計情報があります。

- エラー トラップの数
- イベント アクション トラップの数
- SNMP GET 要求の数
- SNMP SET 要求の数

# CtlTransSource

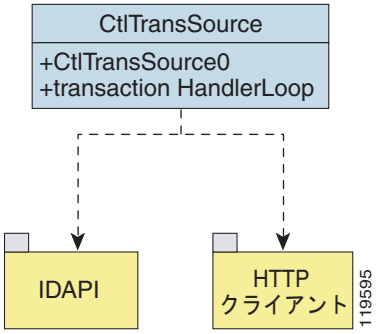
CtlTransSource は、ローカルで開始されたリモート制御トランザクションを HTTP プロトコルを使用してリモートの宛先に転送するアプリケーションです。CtlTransSource は、TLS または非 TLS 接続を開始し、その接続を使用してリモート制御トランザクションを HTTP サーバに伝えます。

CtlTransSource は、リモート HTTP サーバでリモート制御トランザクションを実行するために必要な資格を得る必要があります。CtlTransSource は、リモート ノード上の HTTP サーバにユーザ名とパスワードの形式でアイデンティティを提示することによって資格を得ます（基本認証）。正常に認証されると、ユーザ認証を含む Cookie がリクエスタに割り当てられます。この接続に関するすべての要求では、このクッキーを提示する必要があります。

CtlTransSource サーバ内の transactionHandlerLoop メソッドは、リモート制御トランザクションに対するプロキシとして機能します。ローカル アプリケーションがリモート制御トランザクションを起動すると、IDAPI は最初にトランザクションを CtlTransSource に送信します。transactionHandlerLoop メソッドは、CtlTransSource に送信されたリモート制御トランザクションを待機するループです。

図 A-2 に、CtlTransSource 内の transactionHandlerLoop メソッドを示します。

**図 A-2　　CtlTransSource**




transactionHandlerLoop は、リモート アドレッシングされたトランザクションを受信すると、そのリモート制御トランザクションをリモートの宛先に転送するように試みます。transactionHandlerLoop は、トランザクションを制御トランザクション メッセージの形式にします。transactionHandlerLoop は、HttpClient クラスを使用して、リモート ノード上の HTTP サーバに対する制御トランザクション要求を発行します。リモート HTTP サーバは、リモート制御トランザクションを処理し、適切な応答メッセージを HTTP 応答として返します。リモート HTTP サーバが IPS Web サーバである場合、この Web サーバは CtlTransSource サーブレットを使用してリモート制御トランザクションを処理します。

transactionHandlerLoop は、制御トランザクションの応答として、応答または失敗応答のいずれかを、リモート制御トランザクションの発信側に返します。HTTP サーバが非認証ステータス応答（HTTP クライアントが HTTP サーバに対する十分な資格を持っていないことを示す）を返す場合、transactionHandlerLoop では指定された CtlTransSource のユーザ名とパスワードを使用してトランザクション要求を再発行することにより、リクエスタのアイデンティティを認証します。transactionHandlerLoop は、終了を指示する制御トランザクションを受信するか終了イベントが発生するまでループします。

# Attack Response Controller

ここでは、ARC について説明します。次の事項について説明します。

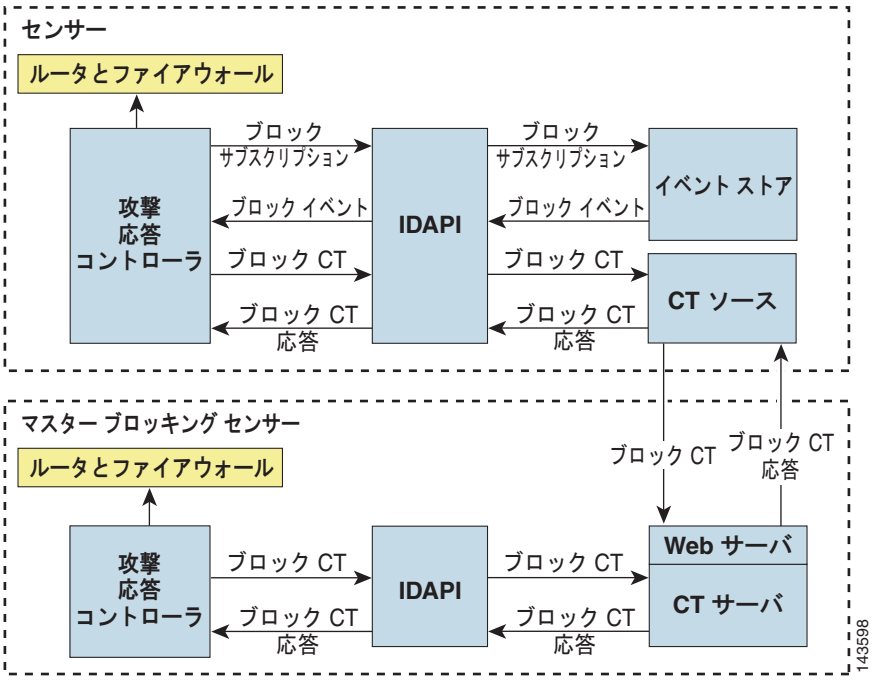


## ARC について

ARC の主な役割は、イベントをブロックすることです。NAC アプリケーションはブロックに対応するとき、管理対象のデバイスと直接対話してブロックを有効化するか、Control Transaction Server を通じてマスター ブロッキング センサーにブロック要求を送信します。マスター ブロッキング センサー上の Web サーバは、制御トランザクションを受け取るとそれを Control Transaction Server に渡し、Control Transaction Server は ARC に渡します。次に、マスター ブロッキング センサー上の ARC は、管理対象のデバイスと対話してブロックをイネーブルにします。

 は ARC を図示したものです。図 A-3

**図 A-3　　ARC**



（注） ARC のインスタンスは、ネットワーク デバイスを制御できない場合、1 つだけを制御できる場合、または複数を制御できる場合があります。ARC は、他の ARC アプリケーション、IPS 管理ソフトウェア、他のネットワーク管理ソフトウェア、システム管理者との間でネットワーク デバイスの制御を一切共有しません。1 つのセンサーについて実行できる ARC のインスタンスは 1 つだけです。

ARC は、次のいずれかに対応してブロックを開始します。

- ブロック アクションが設定されたシグニチャから生成された alert イベント
- CLI、IDM、IME、または ASDM から手動で設定されたブロック
- ホストまたはネットワーク アドレスに対して永続的に設定されたブロック

デバイスをブロックするように ARC を設定すると、ARC はそのデバイスとの間で Telnet または SSH 接続を開始します。ARC は、各デバイスとの接続を維持します。ブロックが開始されると、ARC は制御対象の各デバイスに新しい設定または ACL のセットを（インターフェイス方向ごとに 1 つずつ）プッシュします。ブロックが完了すると、すべての設定または ACL はブロックを削除するようにアップデートされます。

## ARC の機能

ARC には、次の機能があります。

- 3DES（デフォルト）または DES 暗号化を使用する Telnet および SSH 1.5。

  そのデバイスの ARC 設定で指定されたプロトコルのみが試行されます。何らかの理由で接続が失われると、ARC は再確立を試みます。

- ルータ上の既存 ACL およびスイッチ上の既存 VACL。

  既存 ACL が、ARC によって制御されるルータのインターフェイスまたは方向に存在する場合は、この ACL を ARC によって生成される設定にマージするように指定できます。これは、preblock ACL を指定するとすべてのブロックの前に、postblock ACL を指定するとブロックの後に行われます。Catalyst 6000 VACL デバイス タイプには、ARC が制御するインターフェイスごとに preblock および postblock VACL を指定できます。ファイアウォール デバイス タイプでは、ブロックを実行するために別の API が使用され、ARC はファイアウォール上の既存 ACL には影響を与えません。

  （注） Catalyst 5000 RSM および Catalyst 6000 MSFC2 ネットワーク デバイスは、Cisco ルータとして同じようにサポートされます。

- リモート センサーのリストに対するブロックの転送

  ARC は、リモート センサーのリストにブロックを転送できます。そのため、複数のセンサーが実質的に集団となって 1 つのネットワーク デバイスを制御することができます。このようなリモート センサーをマスター ブロッキング センサーと言います。

- ネットワーク デバイスのインターフェイスに対するブロッキングの指定

  ルータの ARC 設定で、ブロッキングが実行されるインターフェイスおよび方向を指定できます。VACL 設定では、ブロッキングが実行されるインターフェイスを指定できます。

  （注） Cisco ファイアウォールは、インターフェイスおよび方向に基づくブロックを行いません。したがって、この設定が Cisco ファイアウォールで指定されることはありません。

ARC は、同時に 250 までのインターフェイスを制御できます。

- ホストまたはネットワークに対する指定された時間のブロッキング

  ARC は、分単位で指定された時間だけ、または永続的にホストまたはネットワークをブロックできます。ARC は、ブロックの期限がいつ切れたのかを判断し、期限切れになるとホストまたはネットワークのブロックを解除します。

- 重要なイベントのロギング

  ARC は、ブロックまたはブロック解除アクションが正常に完了するか何らかのエラーが発生すると、確認イベントを書き込みます。また、ARC は、ネットワーク デバイスの通信セッションの切断と回復、コンフィギュレーション エラー、ネットワーク デバイスから報告されるエラーなどの重要なイベントも記録します。

- ARC 再起動時全体におけるブロッキング状態の維持

  シャットダウンまたは再起動が発生したとき、ARC は期限が切れていないブロックを再適用します。シャットダウン中に期限が切れたブロックは ARC によって削除されます。

  **（注）** ARC がブロッキング状態を正しく維持するためには、アプリケーションのシャットダウン中にシステムの時刻が変更されないことが条件です。

- ネットワーク デバイス再起動時におけるブロッキング状態の維持

  ネットワーク デバイスがシャットダウンされ、再起動されると、ARC は必要に応じてブロックを再適用したり、期限が切れたブロックを削除したりします。ARC は、ARC のシャットダウンおよび再起動が同時に、または重複して発生しても影響を受けません。

- 認証と認可

  ARC は、リモート TACACS+ サーバの使用を含め、AAA 認証および認可を使用するネットワーク デバイスとの間で通信セッションを確立できます。

- 2 種類のブロッキング

  ARC は、ホスト ブロックとネットワーク ブロックをサポートしています。ホスト ブロックは、接続ベースまたは無条件です。ネットワーク ブロックは、常に無条件です。

- NAT アドレス指定

  ARC は、センサーに対して NAT アドレスを使用するネットワーク デバイスを制御できます。ネットワーク デバイスを設定する際に NAT アドレスを指定すると、そのデバイスに対するブロックからセンサーのアドレスがフィルタ処理されるときに、ローカル IP アドレスの代わりにそのアドレスが使用されます。

- シングル ポイント制御

  ARC はネットワーク デバイスの制御を管理者や他のソフトウェアとの間で共有しません。設定をアップデートする必要がある場合は、変更が完了するまで ARC をシャットダウンしておきます。ARC は、CLI または任意の Cisco IPS マネージャからイネーブルまたはディセーブルにすることができます。イネーブルにし直された ARC は、自身を完全に初期化し直します。これには、制御対象のネットワーク デバイスごとに現在の設定を再読み込みすることも含まれます。

  **（注）** ファイアウォールなどの任意のネットワーク デバイスを設定する際は、ARC によるブロッキングを無効にすることを推奨します。

- 常に最大 250 のアクティブなブロックを維持

ARC は、同時に 250 までのアクティブなブロックを維持できます。ARC では 65535 個までのブロックをサポートできますが、1 度に 250 個までを許可することを推奨します。

（注）　ブロックの数は、インターフェイスおよび方向の数とは異なります。

## サポートされているブロッキング デバイス

ARC は、次のデバイスを制御できます。

- Cisco IOS 11.2 以降を実行する Cisco ルータ

（注）　レート制限を実行するには、ルータで Cisco IOS 12.3 以降を実行する必要があります。

- スーパーバイザ エンジン上で実行される Supervisor Engine ソフトウェア 5.3(1) 以降、および RSM 上で実行される IOS 11.2(9)P 以降を使用する Catalyst 5000 シリーズ スイッチ

（注）　ブロッキングは RSM 上で実行されるため、RSM が必要です。

- PFC がインストールされ、Catalyst ソフトウェア 5.3 以降が実行される Catalyst 6000 シリーズ スイッチ
- Catalyst ソフトウェア 5.4(3) 以降、および MSFC2 上の Cisco IOS 12.1(2)E 以降を使用する Catalyst 6000 MSFC2
- 次の Cisco ASA 500 シリーズ スイッチのモデル ASA 5510、ASA 5520、および ASA 5540
- FWSM

（注）　FWSM はマルチモードの管理コンテキストではブロックを実行できません。

## ACL と VACL

ARC が制御するインターフェイスまたは方向のパケットをフィルタ処理する場合は、すべてのブロックの前に ACL を適用したり（preblock ACL）、すべてのブロックの後に ACL を適用する（postblock ACL）ように ARC を設定したりできます。これらの ACL は、ネットワーク デバイス上で非アクティブな ACL として設定されます。preblock および postblock ACL は、インターフェイスおよび方向ごとに定義できます。ARC は、ネットワーク デバイス上のアクティブな ACL をアップデートする際、リストを取得してキャッシュしてから、ブロッキング ACE にマージします。ほとんどの場合は、ブロックの効果を妨げないように、既存 ACL を postblock ACL として指定します。ACL はパケットを、最初に検索された ACE と照合することにより動作します。最初の ACE でパケットが許可された場合、その後の拒否ステートメントは検索されません。

インターフェイスおよび方向ごとに異なる preblock および postblock ACL を指定することも、同じ ACL を複数のインターフェイスおよび方向に再利用することもできます。preblock リストを適用しない場合は、ホストやネットワークに対して never block オプションを使用したり、既存の設定ステートメントを使用して常にブロックしたりすることができます。forever block は、normal block でタイムアウト値を -1 にした場合と同じです。

ARC は、所有する ACL のみを変更します。ユーザによって定義された ACL は変更されません。ARC が維持する ACL は、ユーザ定義の ACL では使用が禁じられている固有の形式になっています。命名規則は、**IPS_**<interface_name>**_[in | out]_[0 | 1]** です。<interface_name> は、ブロッキング インターフェイスに対して ARC 設定で指定された名前に対応します。

Catalyst スイッチでは、これは、ブロッキング インターフェイスの VLAN 番号です。これらの名前は、preblock および postblock ACL では使用しないでください。

Catalyst 6000 VACL では、preblock および postblock VACL を指定できます。また、インターフェイスのみが指定可能です（VLAN では方向は使用されません）。

ファイアウォールでは、ブロッキングに異なる API が使用されるため、preblock ACL および postblock ACL を使用できません。代わりに、ファイアウォールでは ACL を直接作成する必要があります。

## 再起動時の状態の維持

センサーがシャットダウンすると、ARC は、すべてのブロックおよびレート制限（開始時のタイムスタンプ付き）を、ARC が保守しているローカル ファイル（nac.shun.txt）に書き込みます。起動した ARC は、このファイルを使用して、制御対象のネットワーク デバイスでブロック アップデートを必要とするかどうかを判断します。ファイル内に期限が切れていないブロックが見つかると、ネットワーク デバイスの起動時に適用されます。ARC がシャットダウンするときは、有効な未処理ブロックが存在していても ACL に対して特別なアクションは行われません。nac.shun.txt ファイルは、ARC の実行中にシステムの時刻が変更されていない場合に限り正確です。

**注意**　nac.shun.txt ファイルには手動による変更を加えないでください。

次のシナリオに、ARC が再起動時全体でどのように状態を維持するのかを示します。

### シナリオ 1

ARC を停止する時点で 2 つのブロックが有効であり、そのうち 1 つは ARC の再起動よりも前に期限が切れます。再起動した ARC は、最初に nac.shun.txt ファイルを読み取ります。次に、preblock および postblock ACL または VACL を読み取ります。アクティブな ACL または VACL は、次の順序で構築されます。

1. **allow** *sensor_ ip_address* コマンド（**allow sensor shun** コマンドが設定されていない場合）
2. preblock ACL
3. 設定にある **always block** コマンドのエントリ
4. nac.shun.txt にある期限が切れていないブロック
5. postblock ACL

ARC 設定でホストが never block と指定されている場合、ACL の permit ステートメントには変換されません。代わりに、ARC にキャッシュされて、受信 addShunEvent イベントおよび addShunEntry 制御トランザクションをフィルタ処理するために使用されます。

### シナリオ 2

preblock ACL および postblock ACL は指定されていませんが、アクティブな ACL が存在しています。新しい ACL は、次の順序で構築されます。

1. **allow** *sensor_ ip_address* コマンド（**allow sensor shun** コマンドが設定されていない場合）
2. 設定にある **always block** コマンドのエントリ
3. nac.shun.txt にある期限が切れていないブロック

4. **permit IP any any** コマンド

## 接続ベースおよび無条件のブロッキング

ARC は、ホストに関しては 2 種類、ネットワークに関しては 1 種類のブロッキングをサポートしています。ホスト ブロックは、接続ベースまたは無条件です。ネットワーク ブロックは、常に無条件です。

ARC は、ホスト ブロックを受信すると、そのホスト ブロックの connectionShun 属性を調べます。connectionShun が true に設定されていると、ARC は接続ブロッキングを実行します。すべてのホスト ブロックは、宛先 IP アドレス、ソース ポート、宛先ポート、プロトコルといったオプションのパラメータを含むことができます。接続ブロックが実行されるためには、少なくとも送信元 IP アドレスおよび宛先 IP アドレスが存在している必要があります。送信元ポートが接続ブロックに含まれている場合、その送信元ポートは無視され、ブロックに組み込まれません。

次の条件のとき、ARC は必要に応じて接続タイプからブロックを変換して、ブロックを無条件にします。

- 指定されたソース IP アドレスに対して、いずれかのタイプのブロックがアクティブである。
- そのソース IP アドレスに対して、いずれかのタイプの新しいブロックが受信された。
- 新しいブロックのいずれかのオプション パラメータ（ソース ポートを除く）が以前のブロックと異なる。

ブロックがアップデートされると（既存のブロックがすでに有効になっているソース IP アドレスやネットワークに関して新しいブロックが受信された場合など）、既存のブロックの残り時間（分）が決定されます。新しいブロックの時間がこの残り時間以下の場合、アクションは何も発生しません。そうでない場合は、新しいブロックのタイムアウトによって既存のブロックのタイムアウトが置き換えられます。

**注意**　Cisco ファイアウォールでは、ホストの接続ブロッキングをサポートしていません。接続ブロックが適用されると、ファイアウォールは接続ブロックを無条件ブロックのように扱います。Cisco ファイアウォールでは、ネットワーク ブロッキングもサポートしていません。ARC が Cisco ファイアウォールに対してネットワーク ブロックの適用を試みることはありません。

## Cisco ファイアウォールを使用したブロッキング

ARC では、**shun** コマンドを使用することにより、ファイアウォールでのブロックを実行します。**shun** コマンドの形式は次のとおりです。

- IP アドレスをブロックする。

  **shun srcip** [*destination_ip_address source_port destination_port* [*port*]]

- IP アドレスのブロックを解除する。

  **no shun ip**

- すべてのブロックをクリアする。

  **clear shun**

- アクティブなブロック、または実際にブロックされているグローバル アドレスを表示する。

  **show shun** [*ip_address*]

ARC は、**show shun** コマンドに対する応答を使用して、ブロックが実行されたかどうかを判別します。

**shun** コマンドは既存の ACL、コンジット、アウトバウンド コマンドを置き換えるものではないため、既存のファイアウォール設定をキャッシュしたり、ブロックをファイアウォール設定にマージしたりする必要はありません。

**注意**　ARC の実行中は、手動でブロックを実行したり既存のファイアウォール設定を変更したりしないでください。

**block** コマンドでソース IP アドレスのみを指定すると、既存のアクティブな TCP 接続は維持されますが、ブロックされたホストからの着信パケットはすべてドロップされます。

ARC が最初に起動したとき、ファイアウォールでアクティブなブロックが内部のブロッキング リストと比較されます。内部のリストに対応するエントリがないブロックは削除されます。

ARC は、ファイアウォールでの認証をサポートするためにローカル ユーザ名または TACACS+ サーバを使用します。ファイアウォールで認証には AAA を使用して TACACS+ サーバは使用しないように設定すると、ARC はファイアウォールとの通信に予約済みのユーザ名 *pix* を使用します。

ファイアウォールで認証に TACACS+ サーバを使用する場合は、TACACS+ ユーザ名を使用します。AAA ログインを使用する一部のファイアウォール設定では、3 つのパスワード プロンプトが表示されます。初期ファイアウォール パスワード、AAA パスワード、イネーブル パスワードです。ARC では、初期ファイアウォール パスワードと AAA パスワードを同じにすることが要求されます。

NAT または PAT を使用するようにファイアウォールを設定し、ネットワーク外にあるファイアウォール上のパケットをセンサーがチェックする場合、ネットワーク内のファイアウォールから開始されたホスト攻撃が検出されると、センサーはファイアウォールから提供された変換アドレスのブロックを試みます。ダイナミック NAT アドレッシングを使用している場合は、ブロックが効果を発揮しなかったり、無害なホストがブロックされることがあります。PAT アドレッシングを使用している場合は、ファイアウォールが内部ネットワーク全体をブロックする可能性があります。これらの状況を回避するには、センサーを内部インターフェイスに配置するか、センサーがブロックを行わないように設定します。

## Catalyst スイッチを使用したブロッキング

PFC をインストールした Catalyst スイッチでは、VACL を使用してパケットをフィルタ処理します。VACL は、VLAN 間および VLAN 内のすべてのパケットをフィルタ処理します。

WAN カードが取り付けられている場合は MSFC ルータ ACL がサポートされ、MSFC2 を通じてセンサーがインターフェイスを制御するようにすることができます。

**（注）**　MSFC2 カードは、Catalyst スイッチで VACL によるブロッキングを行うための設定の一部として必要なわけではありません。

**注意**　Catalyst スイッチで ARC を設定する場合は、制御インターフェイスで方向を指定しないでください。インターフェイス名は VLAN 番号です。preblock および postblock のリストは、VACL である必要があります。

Catalyst VACL に対しては、次のコマンドを使用できます。

- 既存の VACL を表示する。

  **show security acl info** *acl_name*

- アドレスをブロックする（*address_spec* は、ルータの ACL で使用されるものと同一）。
  **set security acl ip** *acl_name* **deny** *address_spec*
- リストの構築後に VACL をアクティブにする。
  **commit security acl all**
- 1 つの VACL をクリアする。
  **clear security acl map** *acl_name*
- すべての VACL をクリアする。
  **clear security acl map all**
- VACL を VLAN にマップする。
  **set sec acl** *acl_name vlans*

# Logger

センサーは、すべてのイベント（alert、error、status、debug の各メッセージ）を永続的な循環バッファに記録します。また、センサーは IP ログも生成します。このメッセージと IP ログには、CLI、IDM、ASDM、および SDEE クライアントからアクセスできます。

IPS アプリケーションは、Logger を使用してメッセージを記録します。Logger は、ログ メッセージを debug、timing、warning、error、fatal の 5 レベルの重大度のいずれかで送信します。Logger は、ログ メッセージを、循環式のテキスト ファイルである /usr/cids/idsRoot/log/main.log に書き込みます。ファイルがサイズの上限に達すると、古いメッセージは新しいメッセージによって上書きされます。したがって、main.log では、最後に書き込まれたメッセージが末尾にあるとは限りません。main.log に書き込まれた最新の行を見つけるには、ストリング「= END OF FILE =」を検索してください。

main.log は、**show tech-support** コマンドの出力に含まれます。メッセージが warning またはそれよりも上（error または fatal）のレベルで記録されると、Logger はメッセージを evError イベントに変換して（対応するエラーの重大度で）、イベント ストアに挿入します。

Logger は、informational 以上のレベルで cron メッセージ以外のすべての syslog メッセージ（*.info;cron.none）を受信し、エラーの重大度を Warning に設定してから evErrors としてイベント ストアに挿入します。Logger およびアプリケーションのロギングは、service logger コマンドによって制御されます。

Logger は、ロギング ゾーンごとにロギング重大度を制御することにより、各アプリケーションが生成するログ メッセージを制御できます。ユーザは、TAC のエンジニアまたは開発者の依頼および指示の下で、ロガー サービスの individual-zone-control にのみアクセスします。トラブルシューティングのために、TAC はデバッグ ロギングを依頼することがあります。

# AuthenticationApp

ここでは、AuthenticationApp について説明します。次の事項について説明します。

## AuthenticationApp について

AuthenticationApp には、次の役割があります。

- ユーザのアイデンティティを認証する。
- ユーザのアカウント、権限、キー、および証明書を管理する。
- AuthenticationApp、およびセンサー上の他のアクセス サービスで使用する認証方法を設定する。

## ユーザの認証

ユーザ アクセスに適切なセキュリティを確立するために、センサーで認証を設定する必要があります。センサーをインストールすると、初期アカウントとして、パスワードの期限が切れている cisco というアカウントが作成されます。センサーに対する管理アクセス権を持ったユーザは、デフォルトの管理アカウント（cisco）を使用してセンサーにログインすることにより、CLI または IPS マネージャ（IDM、ASDM など）を通じてセンサーにアクセスします。CLI で、管理者はパスワードの変更を要求されます。IPS マネージャは、setEnableAuthenticationTokenStatus 制御トランザクションを開始することによってアカウントのパスワードを変更します。

CLI または IPS マネージャを通じて、管理者は、ユーザ名とパスワード、SSH 許可されたキーなど、使用する認証方法を設定します。管理者用のアプリケーションは、認証設定を確立するために setAuthenticationConfig 制御トランザクションを開始します。

認証設定には、アカウント ロッキングの処理方法を指定するログイン試行の上限値が含まれています。アカウント ロッキングは、ログインの試みが連続して失敗した回数が、指定されたログイン試行の上限値を超えると起動されます。アカウントがロックされると、その後のログインの試行はすべて拒否されます。アカウントのロックを解除するには、setEnableAuthenticationTokenStatus 制御トランザクションを使用してアカウントの認証トークンをリセットします。アカウント ロッキング機能は、ログイン試行の上限値を 0 に設定すると無効になります。

管理者は、CLI または IPS マネージャから新しいユーザ アカウントを追加できます。

## センサーにおける認証の設定

ユーザが Web サーバや CLI などのサービスを通じてセンサーにアクセスしようとするときは、ユーザのアイデンティティを認証し、ユーザの権限を確立する必要があります。ユーザにアクセスを提供するサービスは、ユーザのアイデンティティを認証するために、AuthenticationApp に対して execAuthenticateUser 制御トランザクション要求を開始します。通常、制御トランザクション要求にはユーザ名とパスワードが含まれています。または、SSH 許可されたキーによってユーザのアイデンティティを認証できます。

AuthenticationApp は、execAuthenticateUser 制御トランザクション要求に対して、ユーザのアイデンティティの認証を試みることによって応答します。AuthenticationApp は、ユーザの認証ステータスおよび権限を含む制御トランザクション応答を返します。ユーザのアイデンティティを認証できない場合、AuthenticationApp は、非認証ステータスと匿名ユーザ権限を制御トランザクション応答として返します。制御トランザクション応答は、アカウントのパスワードが期限切れであるかどうかも示します。execAuthenticateUser 制御トランザクションを開始することによってユーザを認証するユーザ インターフェイス アプリケーションは、ユーザにパスワードの変更を要求します。

AuthenticationApp は、基盤となるオペレーティング システムを使用してユーザのアイデンティティを確認します。すべての IPS アプリケーションは、AuthenticationApp に制御トランザクションを送信します。AuthenticationApp は、オペレーティング システムを使用してその応答を作成します。

リモート シェル サービスである Telnet と SSH は、IPS アプリケーションではありません。これらは、オペレーティング システムを直接呼び出します。ユーザが認証されていれば、オペレーティング システムは IPS CLI を起動します。この場合、CLI は特殊な形式の execAuthenticateUser 制御トランザクションを送信することにより、ログイン ユーザの権限レベルを判断します。次に CLI は、この権限レベルに応じて、使用可能にするコマンドを用意します。

## TLS および SSH 信頼関係の管理

IP ネットワーク上の暗号化通信は、パケット内のデータを復号化するために必要な秘密キーを、交換されるパケットだけから受動的攻撃者が発見できないようにすることで、データ プライバシーを実現します。

しかし、同じような危険性を持つ攻撃ベクトルとして、接続のサーバ側であるように装う詐称があります。すべての暗号化プロトコルには、クライアントがこの種の攻撃から身を守るための手段が用意されています。IPS は、SSH と TLS という 2 つの暗号化プロトコルをサポートしています。また、AuthenticationApp は、センサーが暗号化通信のクライアントまたはサーバになる場合の信頼を管理するのに役立ちます。

IPS Web サーバおよび SSH サーバは、暗号化通信のサーバ エンドポイントです。これらは、秘密キーによって ID を保護し、接続してくるクライアントに公開キーを提供します。TLS では、この公開キーは X.509 証明書の中に含まれています。X.509 証明書には他の情報も格納されています。センサーに接続するリモート システムは、接続確立時に受け取った公開キーが、目的のキーであることを確認する必要があります。

クライアントは、中間者攻撃を防御するため、信頼できる公開キーのリストを維持する必要があります。この信頼性を確立するための詳細な手順は、プロトコルおよびクライアント ソフトウェアによって異なります。一般的に、クライアントは 16 ～ 20 バイトのフィンガープリントを表示します。クライアントが信頼を確立するように設定する人間のオペレータは、信頼の確立を試行する前に、アウトオブバンド方式を使用してサーバのキー フィンガープリントを取得する必要があります。フィンガープリントが一致すると信頼関係が確立され、その後、クライアントは自動的にそのサーバに接続でき、リモート サーバが詐称者でないことを確信できます。

**show ssh server-key** および **show tls fingerprint** を使用して、センサーのキー フィンガープリントを表示できます。センサー コンソールに直接接続したときにこれらのコマンドの出力を記録しておくと、後から信頼関係を確立する際、その情報を使用することにより、ネットワークを通じてセンサーのアイデンティティを確認できます。

たとえば、最初に Microsoft Internet Explorer Web ブラウザを通じてセンサーに接続したときには、セキュリティ警告ダイアログボックスによって、証明書は信頼されていないと示されます。Internet Explorer のユーザ インターフェイスを使用して証明書のサムプリントを調べます。この値は、**show tls fingerprint** コマンドによって表示される SHA1 フィンガープリントに正確に一致する必要があります。確認が終わったら、この証明書をブラウザの信頼済み CA のリストに追加して、永続的な信頼を確立します。

この信頼を確立する手順は、TLS クライアントごとに異なります。センサー自体に TLS クライアントが含まれており、制御トランザクションを他のセンサーに送信したり、アップグレードおよびコンフィギュレーション ファイルを TLS Web サーバからダウンロードするために使用されます。センサーの通信相手となる TLS サーバの信頼性を確立するには、**tls trusted-host** コマンドを使用します。

同様に、センサーには SSH クライアントが含まれており、管理対象ネットワーク デバイスとの通信、アップグレードのダウンロード、リモート ホストへのコンフィギュレーション ファイルおよびサポート ファイルのコピーに使用されます。センサーが接続する SSH サーバとの信頼関係を確立するには、**ssh host-key** コマンドを使用します。

TLS 信頼済み証明書および SSH 既知ホストのリストは、**service trusted-certificates** コマンドおよび **service ssh-known-hosts** コマンドで管理できます。

X.509 証明書には、信頼関係のセキュリティを向上させる追加情報が含まれていますが、これは混乱を招く場合があります。たとえば、X.509 証明書には、その証明書を信頼できる有効期間が含まれています。通常、この期間は、証明書が作成された瞬間から始まる数年間です。使用時点で X.509 証明書が有効であることを厳密に確認するには、クライアント システムで正確なクロックを維持する必要があります。

また、X.509 証明書は、特定のネットワーク アドレスと結び付けられています。センサーはこのフィールドに、センサーのコマンドおよびコントロール インターフェイスの IP アドレスを挿入します。そのため、センサーのコマンドおよびコントロール IP アドレスを変更すると、サーバの X.509 証明書は再生成されます。新しい IP アドレスでこのセンサーを見つけ、新しい証明書を信頼するには、以前の証明書を信頼していた、ネットワーク上のすべてのクライアントを再設定しなければなりません。

AuthenticationApp の SSH 既知ホストおよび TLS 信頼済み証明書サービスを使用することにより、センサーを高いセキュリティ レベルで運用することができます。

## Web サーバ

Web サーバは、SDEE サポートを提供します。これによってセンサーは、セキュリティ イベントの報告、IDIOM トランザクションの受信、および IP ログの提供が可能になります。

Web サーバでは、HTTP 1.0 と 1.1 をサポートしています。Web サーバとの通信には、パスワードなどの機密情報が関係することがよくあります。これらを攻撃者が盗聴することが可能になると、システムのセキュリティが大きく損なわれます。そのため、センサーは TLS がイネーブルな状態で出荷されます。TLS プロトコルは、SSL と互換性のある暗号化プロトコルです。

（注）　RDEP イベント サーバ サービスは IPS 6.1 で非推奨になっており、現在の IPS システム アーキテクチャからは削除されています。Web サーバでは、SDEE イベント サーバを使用するようになりました。

**詳細情報**

SDEE の詳細については、「SDEE」（P.A-34）を参照してください。

# SensorApp

ここでは、SensorApp について説明します。次の事項について説明します。



## SensorApp について

SensorApp は、パケットの取り込みと分析を実行します。ポリシー違反は、シグニチャを介して SensorApp で検出され、違反に関する情報がアラートの形式でイベント ストアに転送されます。

パケットは、センサー上のネットワーク インターフェイスからパケットを収集することを目的としたプロデューサが提供する、プロセッサのパイプラインを経由して流れます。

SensorApp では、次のプロセッサがサポートされています。

- タイム プロセッサ

  このプロセッサは、時分割カレンダーに格納されているイベントを処理します。主なタスクは、古いデータベース エントリを期限切れにすること、および時間に依存する統計情報を計算することです。

- 拒否フィルタ プロセッサ

  このプロセッサは、攻撃者の拒否機能を処理します。拒否された送信元 IP アドレスのリストを維持します。リスト内の各エントリは、グローバル拒否タイマーに基づいて期限切れになります。このタイマーは、仮想センサーの設定で指定できます。

- シグニチャ イベント アクション プロセッサ

  このプロセッサでは、イベント アクションを処理します。次のイベント アクションをサポートしています。

  - TCP フローのリセット
  - IP ログ
  - パケットの拒否
  - フローの拒否
  - 攻撃者の拒否
  - アラート
  - ホストのブロック
  - 接続のブロック
  - SNMP トラップの生成
  - トリガー パケットのキャプチャ

  イベント アクションは、イベント リスク レーティングしきい値と関連付けることができます。このしきい値を上回ることが、アクションを実行する前提になります。

- 統計情報プロセッサ

  このプロセッサは、パケット カウント、パケット着信レートなどのシステム統計情報を追跡します。

- レイヤ 2 プロセッサ

  このプロセッサでは、レイヤ 2 関連イベントを処理します。また、不正なパケットを識別し、処理パスから削除します。アラート、キャプチャ パケット、拒否パケットなど、不正なパケットを検出するための実行可能なイベントを設定できます。レイヤ 2 プロセッサは、設定されているポリシーが原因で拒否されたパケットに関する統計情報を更新します。

- データベース プロセッサ

  このプロセッサは、シグニチャの状態およびフロー データベースを保守します。

- フラグメント再構成プロセッサ

  このプロセッサは、フラグメント化された IP データグラムを再構成します。センサーがインライン モードの場合、IP フラグメントの正規化も処理します。

- ストリーム再構成プロセッサ

  このプロセッサは、TCP ストリームをリオーダーして、さまざまなストリームベースのインスペクタにパケットが到着する順番を保証します。TCP ストリームの正規化も処理します。ノーマライザ エンジンを使用すると、アラートおよび拒否アクションをイネーブル化またはディセーブル化できます。

TCP ストリーム再構成プロセッサ ノーマライザは、ホールドダウン タイマーを備えています。このタイマーがあることで、再設定イベントの後にストリームの状態を再構築できます。このタイマーは設定できません。ホールドダウン間隔の間、システムでは、システムを通過するストリームの最初のパケットでストリームの状態を同期します。ホールドダウンの有効期限が切れると、SensorApp は設定されたポリシーを実施します。このポリシーで、3 ウェイハンドシェイクを使用して開かれていないストリームの拒否を要求する場合、ホールドダウン期間中に静止していた確立済みストリームは転送されず、タイムアウトが許可されます。ホールドダウン期間中に同期化されたストリームは、続行を許可されます。

- シグニチャ分析プロセッサ

  このプロセッサは、ストリームベースではない、処理中のパケット用に設定されているインスペクタにパケットをディスパッチします。

- スレーブ ディスパッチ プロセス

  デュアル CPU システムだけに存在するプロセスです。

一部のプロセッサは、シグニチャの分析を実行するためにインスペクタを呼び出します。すべてのインスペクタは、アラームチャネルを呼び出して、必要に応じてアラートを生成できます。

SensorApp は、次のユニットもサポートしています。

- 分析エンジン

  分析エンジンは、センサーのコンフィギュレーションを処理します。インターフェイスのマッピングに加えて、この設定済みインターフェイスへのシグニチャおよびアラーム チャネル ポリシーのマッピングも行います。

- アラーム チャネル

  アラームチャネルは、インスペクタによって生成されたすべてのシグニチャイベントを処理します。主な機能は、渡された各イベントのアラートを生成することです。

# インライン、正規化、およびイベント リスク レーティング機能

（注）　IPS SSP を搭載した Cisco ASA 5585-X では、正規化をサポートしていません。

SensorApp には、次のインライン機能、正規化機能、およびイベント リスク レーティング機能が含まれています。

- パケットのインライン処理

  センサーがデータ パスのパケットを処理するときには、ポリシー設定で明示的に拒否されていない限り、すべてのパケットは一切変更されずに転送されます。TCP の正規化が原因で、一部のパケットは、適切なカバレッジを確保するために遅延することがあります。ポリシー違反が発生する場合は、SensorApp によってアクションを設定できます。パケットの拒否、フローの拒否、攻撃者の拒否など、追加のアクションをインライン モードで使用できます。

  IPS にとって不明であるか、対象でないすべてのパケットは、分析されずにペアのインターフェイスに転送されます。すべてのブリッジング プロトコルおよびルーティング プロトコルは、そのまま転送されます。ただし、ポリシー違反が原因で拒否される場合があります。インライン データ処理（または混合データ処理）に使用されるインターフェイスには、IP スタックは関連付けられていません。混合モードにおける 802.1q パケットの現在のサポートが、インラインモードまで拡張されます。

- IP の正規化

  IP データグラムのフラグメンテーションは、意図的であってもなくても不正利用の隠蔽につながり、不正利用の検出を困難または不可能にします。ファイアウォール上やルータ上と同様、フラグメンテーションもアクセス コントロール ポリシーを逃れるために使用される可能性があります。さまざまなオペレーティング システムがさまざまなメソッドを使用して、フラグメント化されたデータグラムをキューに入れたりディスパッチしたりします。エンド ホストがデータグラムを再構成する際に使用可能なすべての方法をセンサーでチェックする必要がある場合、センサーは DoS 攻撃に対して脆弱になります。フラグメント化したすべてのデータグラムをインラインで再構成して、完成したデータグラムのみを転送し、必要に応じてデータグラムを再度フラグメント化することが、この問題の解決策です。IP フラグメンテーション正規化装置がこの機能を実行します。

- TCP の正規化

  意図的または意図しない TCP セッション セグメンテーションによって、いくつかの攻撃クラスが非表示になることがあります。ポリシーが false positive や false negative なしに実施されるようにするには、2 つの TCP エンドポイントの状態が追跡され、実際のホスト エンドポイントによって処理されたデータだけが渡される必要があります。TCP ストリームでオーバーラップが発生する可能性はあるものの、TCP セグメントの再送信以外では、非常にまれです。通常、TCP セッションでの上書きは発生しません。それでも上書きが発生するとしたら、セキュリティ ポリシーを意図的に回避しようとしている者があるか、TCP スタックの実装が破損しています。両方のエンドポイントの状態に関する全情報を保持できるのは、センサーが TCP プロキシとして動作している場合だけです。センサーが TCP プロキシとして動作する代わりに、セグメントが適切に並べられ、ノーマライザによって回避や攻撃に関係する異常なパケットが検索されます。

- イベント リスク レーティング

  イベント リスク レーティングでは、潜在的に悪意のあるアクションを検出するだけでなく、次の追加情報も含んでいます。

  - 成功した場合の攻撃の重大度
  - シグニチャの忠実度
  - ターゲットホストに関する潜在的な攻撃の関連性

- ターゲット ホスト全体の価値

イベント リスク レーティングはシステムからの false positive の減少に役立ち、アラーム発生原因のより適切な制御を可能にします。

# SensorApp の新機能

SensorApp には、次の新機能があります。

- ポリシー テーブル：リスク カテゴリ設定（high、medium、および low）のリストを提供します。
- 回避保護：インライン インターフェイス モードのセンサーを、ノーマライザ用にストリクト モードから非同期モードに切り替えることができます。
- センサー健全性メーター：センサー全体の健全性の統計情報を提供します。
- トップサービス：TCP、UDP、ICMP、および IP プロトコルの上位 10 個のインスタンスを提供します。
- セキュリティ メーター：アラートをプロファイリングして脅威のカテゴリに入れ、この情報をレッド、イエロー、およびグリーンのバケットで報告します。これらのバケットの移行ポイントを設定できます。
- フロー状態のクリア：データベースをクリアできます。データベースをクリアすると、センサーが再起動時と同様に初期状態で起動します。
- 再起動ステータス：センサーの現在の起動段階および再起動段階を定期的に報告します。

# パケット フロー

パケットは、NIC によって受信され、IPS 共有ドライバにより、ユーザによってマッピングされたカーネル内のメモリスペースに配置されます。パケットの前には、IPS ヘッダーが付加されます。各パケットには、パケットがシグニチャ イベント アクション プロセッサに到達したときにそのパケットを通過させるか拒否するかを示すフィールドも含まれています。

ユーザによってマッピングされた共有カーネルのパケット バッファからパケットが取り出され、センサーのモデルに適したプロセッサを実装する処理機能を呼び出します。次の順序になります。

- 単一プロセッサ実行

  Time Processor --> Layer 2 Processor --> Deny Filters Processor --> Fragment Reassembly Processor --> Statistics Processor --> Database Processor --> Signature Analysis Processor --> Stream Reassembly Processor --> Signature Event Action Processor

- デュアル プロセッサ実行

  実行スレッド 1 Time Processor --> Layer 2 Processor --> Deny Filters Processor --> Fragment Reassembly Processor --> Statistics Processor --> Database Processor --> Signature Analysis Processor --> Slave Dispatch Processor --> | 実行スレッド 2 Database Processor --> Stream Reassembly Processor --> Signature Event Action Processor

# シグニチャ イベント アクション プロセッサ

シグニチャ イベント アクション プロセッサは、アラーム チャネル内のシグニチャ イベントから、シグニチャ イベント アクション オーバーライド、シグニチャ イベント アクション フィルタ、およびシグニチャ イベント アクション ハンドラを経由する処理までのデータ フローを調整します。次のコンポーネントで構成されています。

- アラーム チャネル

  SensorApp 検査パスからシグニチャ イベント処理にシグニチャ イベントを通知する領域を表すユニット。

- シグニチャ イベント アクション オーバーライド

  リスク レーティング値に基づいてアクションを追加します。シグニチャ イベント アクション オーバーライドは、設定したリスク レーティングしきい値の範囲に入るすべてのシグニチャに適用されます。各シグニチャ イベント アクション オーバーライドは独立しており、アクション タイプごとに個別の設定値を持ちます。

- シグニチャ イベント アクション フィルタ

  シグニチャ イベントのシグニチャ ID、アドレス、およびリスク レーティングに基づいてアクションを削除します。シグニチャ イベント アクション フィルタへの入力は、シグニチャ イベント アクション オーバーライドによって追加される可能性のあるアクションを持つシグニチャ イベントです。

  **（注）** シグニチャ イベント アクション フィルタが実行できるのはアクションの削除だけであり、新しいアクションの追加はできません。

  シグニチャ イベント アクション フィルタには、次のパラメータが適用されます。

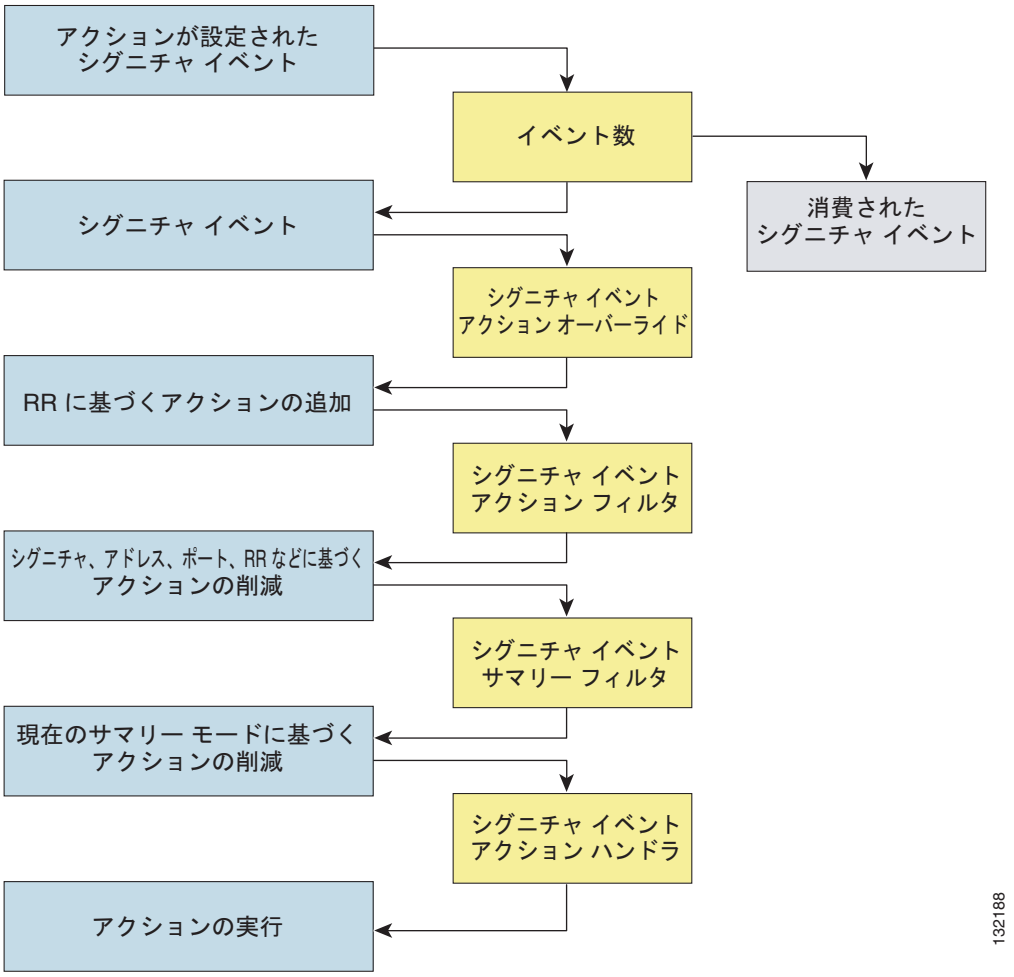
  - シグニチャ ID
  - サブシグニチャ ID
  - 攻撃者のアドレス
  - 攻撃者のポート
  - 攻撃対象のアドレス
  - 攻撃対象のポート
  - リスク レーティングしきい値の範囲
  - 削除するアクション
  - シーケンス識別子（任意）
  - ストップ ビットまたは継続ビット
  - アクション フィルタ行をイネーブルにするビット
  - 攻撃対象 OS との関連性または OS との関連性

- シグニチャ イベント アクション ハンドラ

  要求されたアクションを実行します。シグニチャ イベント アクション ハンドラからの出力は、実行中のアクションであり、イベント ストアに evIdsAlert が書き込まれる場合があります。

図 A-4（P.A-28）に、シグニチャ イベント アクション プロセッサを通過するシグニチャ イベントの論理フローと、このイベントのアクションで実行される操作を示します。これは、アラーム チャネルで受信された設定済みアクションを持つシグニチャ イベントで開始され、シグニチャ イベント アクション プロセッサの機能コンポーネントを通過するシグニチャ イベントに合わせて上から下へ流れます。

図 A-4　シグニチャ イベント アクション プロセッサを通過するシグニチャ イベント




# CollaborationApp

ここでは、CollaborationApp について説明します。次の事項について説明します。

- 「CollaborationApp について」（P.A-28）
- 「アップデート コンポーネント」（P.A-29）
- 「error イベント」（P.A-30）

## CollaborationApp について

CollaborationApp は、MainApp と SensorApp のピアです。CollaborationApp は、IDAPI 制御トランザクション、セマフォ、共有メモリ、ファイル交換などのさまざまなプロセス間通信テクノロジーを使用してこれらとインターフェイスします。

レピュテーションのアップデートは、グローバル相関サーバと CollaborationApp の間で交換されます。CollaborationApp は、4 つのアップデート コンポーネントを使用して、センサーと通信します。

- 一連の規則スコア重み値
- 一連の IP アドレスおよびアドレス範囲。この情報と、規則およびアラートが一緒になって、レピュテーション スコアの計算に必要な情報が用意されます。
- トラフィックを常に拒否する必要のある IP アドレスおよびアドレス範囲のリスト
- ネットワーク参加設定。これにより、サーバでは、センサーでテレメトリ日付をサーバに送信するときのレートを制御できます。

センサーは、コラボレーション情報を、ネットワーク参加サーバに送信します。センサーは、グローバル相関サーバにクエリーして、入手可能なコラボレーション アップデートのリストおよびアップデート ファイルをダウンロードできる グローバル相関 サーバについての情報を取得します。

（注） SensorApp は CollaborationApp よりも前に開始されますが、これらは非同期で初期化されます。したがって、SensorApp でアップデートを受け入れることができる状態になる前に、レピュテーション アップデート サーバで 1 つ以上のグローバル相関アップデートをダウンロードして適用を試行する可能性があります。アップデート サーバでは、アップデートをダウンロードして部分的に処理できますが、アップデートをコミットするには、SensorApp の準備ができるまで待機する必要があります。

### 詳細情報

グローバル相関の詳細および設定方法については、第 11 章「グローバル相関の設定」を参照してください。

## アップデート コンポーネント

グローバル相関アップデート クライアントは、グローバル相関アップデート サーバとマニフェストを交換します。クライアントでは、サーバ マニフェストを解析して、ダウンロード可能な新しいアップデートと、このアップデートが格納されている場所を判別し、インストールするアップデートのリストを作成します。すべてのアップデートが正常に適用されると、グローバル相関アップデート クライアントは、コンポーネントごとに適用されたアップデートをコミットし、新しいアップデートを利用できることを SensorApp に通知し、コンポーネントごとのコミットされた最新のアップデートを反映するためにクライアント マニフェストを更新します。

クライアント マニフェストには、センサーのシリアル番号を含むセンサーの UDI と、センサーが正規の Cisco IPS センサーであることを確認するためにサーバで使用する暗号化された共有秘密情報が含まれています。サーバ マニフェストには、各コンポーネントで利用可能なアップデート ファイルのリストが含まれています。サーバ マニフェストには、リスト内の各アップデート ファイルについて、アップデート バージョン、タイプ、順序、場所、ファイル転送プロトコルなどのデータが含まれています。

アップデート ファイルは 2 種類あります。コンポーネントのデータベースに格納されているすべての既存データを置換するフルアップデート ファイルと、情報の追加、削除、または置換によって既存のレピュテーション データを変更する差分アップデートです。すべてのコンポーネントのすべてのアップデート ファイルが適用されると、作業データベースを置換して一時データベースがコミットされます。

認証と認可は、秘密暗号化機構と復号化キー管理によって実施されます。グローバル相関アップデート サーバでは、クライアント マニフェストに含まれている共有秘密暗号化機構を使用してセンサーを認証します。グローバル相関アップデート クライアントは、復号化キー管理を使用して、センサーを認可します。グローバル相関アップデート サーバによって認証されたセンサーには、アップデート ファイルを復号化できるように、有効なキーがサーバ マニフェストに入れて送信されます。

**注意**　グローバル相関をイネーブルにした一方で、DNS および HTTP プロキシ サーバを設定していない場合は、警告メッセージが送信されます。この警告は、グローバル相関をディセーブルにするか、DNS と HTTP プロキシ サーバのいずれかを追加するよう促すメッセージです。

**詳細情報**

グローバル相関をサポートする DNS または HTTP プロキシ サーバを設定する手順については、「ネットワークの設定」（P.4-2）を参照してください。

# error イベント

グローバル相関アップデートが失敗したときは、必ず evError イベントが生成されます。このエラーメッセージは、センサーの統計情報に組み込まれます。次の状態の場合は、重大度 Error のステータスメッセージが発行されます。

- センサーがライセンスされていない。
- DNS および HTTP プロキシ サーバが設定されていない。
- マニフェスト交換が失敗する。
- アップデート ファイルのダウンロードが失敗する。
- アップデートの適用またはコミットが失敗する。

グローバル相関をイネーブルにするようにホストまたはグローバル相関の設定を編集および保存する一方で、DNS および HTTP プロキシ サーバを設定していない場合は、重大度レベル Warning の evError イベントが生成されます。

**詳細情報**

センサーの統計情報を表示する手順については、「統計情報の表示」（P.17-31）を参照してください。

# CLI

ここでは、Cisco IPS CLI の設置方法について説明します。次の事項について説明します。



# CLI について

CLI は、Telnet、SSH、シリアル インターフェイスなどのすべての直接ノード アクセスについて、センサーのユーザ インターフェイスを提供します。センサー アプリケーションは CLI で設定します。基盤となる OS への直接接続は、サービス ロールを通じて許可されます。

# ユーザ ロール

ユーザ ロールは 4 種類あります。

- ビューア（Viewer）：設定およびイベントを表示できますが、自分のユーザ パスワード以外の設定データは修正できません。
- オペレータ（Operator）：すべてのデータを表示できるほか、次のオプションを修正できます。
  - シグニチャ チューニング（優先順位、無効 / 有効）
  - 仮想センサー定義
  - 管理対象ルータ
  - 自分のユーザ パスワード
- 管理者（Administrator）：すべてのデータを表示できるほか、オペレータが修正できるすべてのオプションに加えて、次のオプションを修正できます。
  - センサーのアドレス設定
  - 設定エージェントまたはビュー エージェントとして接続が許可されたホストのリスト
  - 物理的な検知インターフェイスの割り当て
  - 物理インターフェイスの制御のイネーブル化またはディセーブル化
  - ユーザとパスワードの追加および削除
  - 新しい SSH ホスト キーおよびサーバ証明書の生成
- サービス（service）：サービス権限を持つユーザはセンサーに 1 人だけ存在できます。サービス ユーザは、IDM にログインできません。サービス ユーザは、CLI ではなく bash シェルにログインします。

  サービス ロールは、必要に応じて CLI をバイパスできる特殊なロールです。許可されるサービス アカウントは 1 つだけです。トラブルシューティング用としては、サービス ロールを割り当てたアカウントのみ作成してください。サービス アカウントを編集できるのは、管理者権限を持つユーザだけです。

**注意**　サービス アカウントを作成するかどうかは、慎重に検討する必要があります。サービス アカウントは、システムへのシェル アクセスを提供するため、システムが脆弱になります。ただし、管理者のパスワードが失われた場合は、サービス アカウントを使用してパスワードを作成できます。状況を分析して、システムにサービス アカウントを存在させるかどうかを決定してください。

サービス アカウントにログインすると、次の警告が表示されます。

```
********************** WARNING ************************************************
UNAUTHORIZED ACCESS TO THIS NETWORK DEVICE IS PROHIBITED.
This account is intended to be used for support and troubleshooting purposes only.
Unauthorized modifications are not supported and will require this device to be
re-imaged to guarantee proper operation.
*******************************************************************************
```

# サービス アカウント

サービス アカウントは、サポートとトラブルシューティングのツールです。これによって TAC は、CLI シェルではなくネイティブ オペレーティング システムのシェルにログインすることができます。これは、デフォルトではセンサーには存在しません。センサーのトラブルシューティングのために TAC がこれを使用できるようにするためには、ユーザが作成する必要があります。

1 つのセンサーで使用できるサービス アカウントは 1 つだけです。また、1 つのサービス ロールで使用できるアカウントも 1 つだけです。サービス アカウントのパスワードを設定またはリセットすると、root アカウントのパスワードに同じパスワードが設定されます。そのため、サービス アカウントのユーザはこの同じパスワードを使用して、root に su できます。サービス アカウントを削除すると、root アカウントのパスワードがロックされます。

サービス アカウントは、設定の目的で使用されることを想定していません。TAC の指示に基づき、サービス アカウントからセンサーに対して加えられた変更だけがサポートされます。サービス アカウントからオペレーティング システムに追加のサービスを加えること、および/またはそれを実行することは、他の IPS サービスの適切な実行に影響するため、シスコではこれをサポートしません。TAC は、追加のサービスが加えられたセンサーをサポートしません。

サービス アカウントへのログインは、ログ ファイル /var/log/.tac を確認することによって追跡できます。このファイルは、サービス アカウントによるログインの記録でアップデートされます。

（注） Cisco IPS には、CLI、IDM、または IME から使用できる、複数のトラブルシューティング機能が組み込まれています。大部分のトラブルシューティング状況では、サービス アカウントは不要です。一意性の高い問題をトラブルシューティングするために、TAC の指示に従ってサービス アカウントを作成しなければならないことがあります。サービス アカウントを使用すると、CLI に組み込まれている保護をバイパスでき、root 権限でセンサーにアクセスできます。他にこれを行う方法はありません。具体的な理由で必要となる場合を除き、サービス アカウントは、作成しないことを推奨します。不要になったサービス アカウントは、削除する必要があります。

# 通信

ここでは、Cisco IPS で使用する通信プロトコルについて説明します。次の事項について説明します。



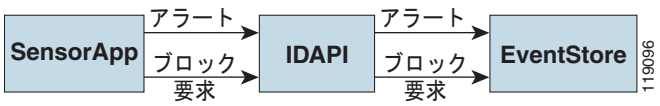
## IDAPI

IPS アプリケーションは、IDAPI というプロセス間通信 API を使用して内部的な通信を処理します。IDAPI はイベント データを読み書きし、制御トランザクションのメカニズムを提供します。IDAPI は、すべてのアプリケーションが通信の際に使用するインターフェイスです。

SensorApp は、そのインターフェイス上のネットワーク トラフィックをキャプチャし、分析します。シグニチャが一致すると、SensorApp はアラートを生成します。このアラートはイベント ストアに格納されます。シグニチャがブロッキング応答アクションを実行するように設定されていると、SensorApp はブロック イベントを生成します。このイベントもイベント ストアに格納されます。

図 A-5 に、IDAPI インターフェイスを示します。

**図 A-5　　IDAPI**




各アプリケーションは、イベントおよび制御トランザクションを送受信するように IDAPI に登録します。IDAPI は次のサービスを提供します。

- 制御トランザクション
  - 制御トランザクションを開始する。
  - インバウンド制御トランザクションを待機する。
  - 制御トランザクションに応答する。
- IPS イベント
  - リモート IPS イベントをサブスクライブする。受信したリモート IPS イベントは、イベント ストアに格納されます。
  - イベント ストアから IPS イベントを読み取る。
  - イベント ストアに IPS イベントを書き込む。

IDAPI は、アトミックなデータ アクセスを実現するために必要な同期メカニズムを備えています。

# IDIOM

IDIOM は、IPS によって報告されるイベント メッセージ、および侵入検知システムの設定と制御に使用される操作メッセージを定義するデータ形式の標準です。これらのメッセージは、IDIOM XML スキーマに準拠した XML ドキュメントによって構成されています。

IDIOM は、イベントと制御トランザクションという 2 種類のインタラクションをサポートしています。イベント インタラクションは、alert などの IPS イベントを交換するために使用されます。IDIOM は、イベント インタラクションにイベント メッセージとエラー メッセージという 2 種類のメッセージを使用します。制御トランザクションは、ホストが別のホストでアクションを開始したり、別のホストの状態を変更または読み取るための手段です。制御トランザクションでは、要求、応答、設定、エラーの 4 種類の IDIOM メッセージが使用されます。1 つのホスト内のアプリケーション インスタンス間で通信されるイベントおよび制御トランザクションは、ローカル イベントまたはローカル制御トランザクションと呼ばれ、ローカル IDIOM メッセージと総称されます。異なるホスト間で通信されるイベントおよび制御トランザクションは、リモート イベントおよびリモート制御トランザクションと呼ばれ、リモート IDIOM メッセージと総称されます。

**（注）** IDIOM の大部分は、IDCONF、SDEE、および CIDEE で置換されました。

# IDCONF

Cisco IPS では、XML ドキュメントを使用して、設定を管理します。IDCONF では、Cisco IPS 制御トランザクションを含む XML スキーマを指定します。IDCONF スキーマは、コンフィギュレーション文書の内容を指定するのではなく、コンフィギュレーション文書作成の基になるフレームワークと基礎

的要素を指定します。このスキーマは、feature-supported 属性を使用することにより、それを設定することができないプラットフォームや機能の場合に、IPS マネージャおよび CLI でその機能を無視できるメカニズムを提供します。

IDCONF メッセージは、IDIOM の要求メッセージと応答メッセージでラップされます。

次に、IDCONF の例を示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<request xmlns="http://www.cisco.com/cids/idiom" schemaVersion="2.00">
  <editConfigDelta xmlns="http://www.cisco.com/cids/idconf">
    <component name="userAccount">
      <config typedefsVersion="2004-03-01" xmlns="http://www.cisco.com/cids/idconf">
        <struct>
           <map name="user-accounts‟ editOp=‟merge”>
             <mapEntry>
               <key>
                <var name="name">cisco</var>
              </key>
               <struct>
                <struct name="credentials">
                  <var name="role">administrator</var>
                </struct>
               </struct>
            </mapEntry>
          </map>
        </struct>
      </config>
    </component>
  </editDefaultConfig>
</request>
```

# SDEE

IPS は、侵入アラート、ステータス イベントなどさまざまなタイプのイベントを生成します。IPS では、IPS 業界を代表する SDEE プロトコルを使用して、管理アプリケーションなどのクライアントにイベントを通信します。SDEE は、セキュリティ デバイス イベントを通信するための製品に依存しない標準です。SDEE によって、さまざまなタイプのセキュリティ デバイスによって生成されるイベントを通信するために必要な拡張性を持つ機能が加わります。

SDEE を使用してクライアントにイベントを通信するシステムを、SDEE プロバイダーと呼びます。SDEE では、HTTP プロトコルまたは HTTP over SSL および TLS プロトコルを使用してイベントをトランスポート可能であると指定できます。HTTP または HTTPS を使用する場合、SDEE プロバイダーは HTTP サーバとして動作し、SDEE クライアントは HTTP 要求の発信側になります。

IPS には、HTTP 要求または HTTPS 要求を処理する Web サーバが含まれています。Web サーバでは、実行時にロードできるサーブレットを使用して、さまざまなタイプの HTTP 要求を処理します。各サーブレットでは、サーブレットと関連付けられた URL に送信された HTTP 要求を処理します。SDEE サーバは、Web サーバ サーブレットとして実装されています。

SDEE サーバでは、許可された要求だけを処理します。要求は Web サーバから発信されている場合に許可されて、クライアントのアイデンティティの認証とクライアントの特権レベルの判別が行われます。

## CIDEE

CIDEE は、Cisco IPS が使用する SDEE への拡張を指定します。CIDEE 規格は、Cisco IPS でサポートしている、すべての拡張候補を指定しています。特定のシステムでは、CIDEE 拡張のサブセットを実装できます。ただし、必須と指定されているすべての拡張は、すべてのシステムでサポートされている必要があります。

CIDEE は、Cisco IPS 固有のセキュリティ デバイス イベント、および SDEE の evIdsAlert 要素に対する IPS の拡張を指定します。

CIDEE では、次のイベントがサポートされています。

- evError：エラー イベント

  CIDEE プロバイダーがエラー状態または警告状態を検出すると生成されます。evError イベントには、エラー コードとテキストによるエラーの説明が含まれます。

- evStatus：ステータス メッセージ イベント

  ホストで重要な可能性のある状態が発生したことを示すために、CIDEE プロバイダーによって生成されます。ステータス イベントでは、さまざまなタイプのステータス メッセージが報告される可能性があります。イベントごとに 1 つのメッセージが生成されます。各タイプのステータス メッセージには、そのステータス メッセージで説明されている発生タイプに固有のデータ要素のセットが含まれます。大部分のステータス メッセージの情報は、監査の目的で役立ちます。エラーと警告はステータス情報とは見なされず、evStatus ではなく evError を使用して報告されます。

- evShunRqst：ブロック要求イベント

  ネットワーク ブロッキングを処理するサービスがブロック アクションを開始することを示すために生成されます。

CDIEE の拡張イベントの例を次に示します。

```
<sd:events xmlns:cid="http://www.cisco.com/cids/2004/04/cidee"
xmlns:sd="“http://example.org/2003/08/sdee”>
 <sd:evIdsAlert eventId="1042648730045587005" vendor="Cisco‟ severity="medium">
  <sd:originator>
    <sd:hostId>Beta4Sensor1</sd:hostId>
    <cid:appName>sensorApp</cid:appName>
    <cid:appInstanceId>8971</cid:appInstanceId>
  </sd:originator>
  <sd:time offset="0" timeZone="UTC">1043238671706378000</sd:time>
  <sd:signature description="IOS Udp Bomb" id="4600" cid:version="S37">
    <cid:subsigId>0</cid:subsigId>
  </sd:signature> …
```

# Cisco IPS ファイル構造

Cisco IPS のディレクトリ構造は次のとおりです。

- /usr/cids/idsRoot：メイン インストール ディレクトリ。
- /usr/cids/idsRoot/shared：システムのリカバリ中に使用されるファイルが保存されます。
- /usr/cids/idsRoot/var：センサーの実行中にダイナミックに作成されるファイルが保存されます。
- /usr/cids/idsRoot/var/updates：アップデート インストレーション用のファイルとログが格納されます。
- /usr/cids/idsRoot/var/virtualSensor：SensorApp が正規表現を分析するために使用するファイルが格納されます。

- /usr/cids/idsRoot/var/eventStore：イベント ストア アプリケーションが含まれます。
- /usr/cids/idsRoot/var/core：システムのクラッシュ時に作成される重要なファイルが格納されます。
- /usr/cids/idsRoot/var/iplogs：iplog ファイルのデータが格納されます。
- /usr/cids/idsRoot/bin：バイナリ実行可能ファイルが含まれます。
- /usr/cids/idsRoot/bin/authentication：認証アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/cidDump：技術サポート向けのデータを収集するスクリプトが含まれます。
- /usr/cids/idsRoot/bin/cidwebserver：Web サーバ アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/cidcli：CLI アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/nac：ARC アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/logApp：ロガー アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/mainApp：メイン アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/sensorApp：センサー アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/collaborationApp：コラボレーション アプリケーションが含まれます。
- /usr/cids/idsRoot/bin/falcondump：IDSM2 のセンシング ポートでパケット ダンプを取得するアプリケーションが含まれます。
- /usr/cids/idsRoot/etc：センサーのコンフィギュレーション ファイルが含まれます。
- /usr/cids/idsRoot/htdocs：Web サーバの IDM ファイルが含まれます。
- /usr/cids/idsRoot/lib：センサー アプリケーションのライブラリ ファイルが含まれます。
- /usr/cids/idsRoot/log：デバッグ用のログ ファイルが含まれます。
- /usr/cids/idsRoot/tmp：センサーの実行中に作成される一時ファイルが格納されます。

# Cisco IPS アプリケーションの要約

表 A-2 に、IPS を構成するアプリケーションの概要を示します。

**表 A-2　アプリケーションの概要**

| アプリケーション | 説明 |
|---|---|
| AuthenticationApp | IP アドレス、パスワード、デジタル証明書に基づいてユーザを許可および認証します。 |
| Attack Response Controller | ARC は、すべてのセンサー上で実行されます。各 ARC は、ローカル イベント ストアの network access イベントをサブスクライブします。ARC の設定には、そのローカル ARC が制御するセンサーおよびネットワーク アクセス デバイスのリストが含まれます。network access イベントをマスター ブロッキング センサーに送信するように設定されている ARC は、デバイスを制御するリモート ARC に対してネットワーク アクセス制御トランザクションを開始します。これらのネットワーク アクセス アクション制御トランザクションは、IPS マネージャがネットワーク アクセス アクションを発行するときにも使用されます。 |
| CLI | コマンドライン入力を受け付け、IDAPI を使用してローカル設定を変更します。 |

表 A-2　アプリケーションの概要（続き）

| アプリケーション | 説明 |
|---|---|
| CollaborationApp | グローバル相関データベースを介して他のデバイスと情報を共用し、すべてのデバイスを組み合わせた効率を向上します。 |
| Control Transaction Server[1] | リモート クライアントからの制御トランザクションを受け付け、ローカル制御トランザクションを開始して、リモート クライアントに応答を返します。 |
| Control Transaction Source[2] | リモート アプリケーションに向けられた制御トランザクションを待機し、制御トランザクションをリモート ノードに転送し、応答を発信側に返します。 |
| IDM | HTML の IPS 管理インターフェイスを備えた Java アプレットです。 |
| IME | イベントを表示およびアーカイブするためのインターフェイスを備えた Java アプレットです。 |
| InterfaceApp | バイパス設定と物理設定を処理し、対になったインターフェイスを定義します。物理設定とは、速度、デュプレックス、および管理状態です。 |
| Logger | アプリケーションのすべてのログ メッセージをログ ファイルに書き込み、アプリケーションのエラー メッセージをイベント ストアに書き込みます。 |
| MainApp | 設定を読み取ってアプリケーションを起動し、アプリケーションの開始および終了とノードの再起動を扱い、ソフトウェアのアップグレードを処理します。 |
| NotificationApp | アラート イベントが、ステータス イベントが、およびエラー イベントによってトリガーされたときに SNMP トラップを送信します。NotificationApp は、パブリック ドメイン SNMP エージェントを使用します。SNMP GET は、センサーの全般的な健全性に関する情報を提供します。 |
| SDEE Server[3] | リモート クライアントからイベントの要求を受け付けます。 |
| SensorApp | モニタされているネットワーク上のトラフィックをキャプチャして分析し、intrusion および network access イベントを生成します。ロギングをオン/オフする IP ロギング制御トランザクション、および IP ログ ファイルを送信および削除する IP ロギング制御トランザクションに応答します。 |
| Web Server | リモート HTTP クライアント要求を待機し、適切なサーブレット アプリケーションを呼び出します。 |

1. これは Web サーバ サーブレットです。
2. これは、リモート制御トランザクション プロキシです。
3. これは Web サーバ サーブレットです。