APPENDIX A

システム アーキテクチャ

この付録では、IPS のシステム アーキテクチャについて説明します。内容は次のとおりです。

Cisco IPS の目的

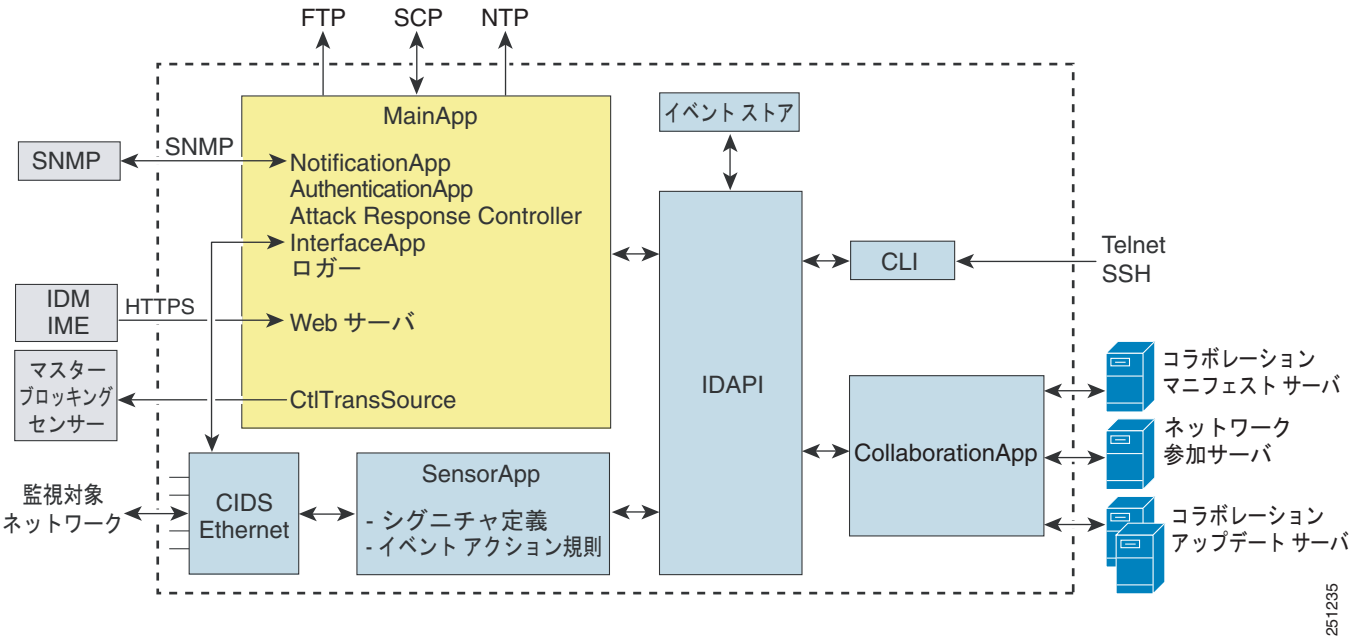
Cisco IPS の目的は、悪意のあるネットワーク アクティビティを検出し、防ぐことにあります。Cisco IPS ソフトウェアは、アプライアンスとモジュールの 2 種類のプラットフォームにインストールできます。Cisco IPS には、管理アプリケーションとモニタリング アプリケーションが含まれています。IDM は、IPS の管理およびモニタに使用できるネットワーク管理 JAVA アプリケーションです。IME は、IPS イベントの表示に使用できる IPS ネットワーク モニタリング JAVA アプリケーションです。IME には、IDM 設定コンポーネントも含まれています。IDM および IME は、HTTP または HTTPS を使用して IPS と通信し、コンピュータでホストされます。

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 7.0 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-6）を参照してください。
- SensorApp の詳細については、「SensorApp」（P.A-25）を参照してください。
- CollaborationApp の詳細については、「CollaborationApp」（P.A-30）を参照してください。
- CLI の詳細については、「CLI」（P.A-32）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、およびアップグレードの実行を行います。次のコンポーネントが含まれます。
  - ctlTransSource（Control Transaction Server）：センサーが制御トランザクションを送信できるようにします。Attack Response Controller（以前の 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。
  - イベント ストア：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：バイパスおよび物理設定を処理し、インターフェイスのペアを定義します。物理設定とは、速度、デュプレックス、および管理ステートです。
  - ロガー：アプリケーションのすべてのログ メッセージをログ ファイルに書き出し、アプリケーションのエラー メッセージをイベント ストアに書き出します。
  - Attack Response Controller（以前の Network Access Controller）：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、スイッチ）を管理し、alert イベントが発生したときにブロッキング機能を提供します。ARC は管理対象ネットワーク デバイス上で ACL を作成し、適用します。または、**shun** コマンド（ファイアウォール）を使用します。
  - NotificationApp：alert、status、および error イベントで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状態に関する情報を提供します。
  - Web サーバ（HTTP SDEE Server）：Web インターフェイスを提供し、IPS サービスを提供するいくつかの servlet を使用して SDEE プロトコルで別の IPS デバイスと通信できるようにします。
  - AuthenticationApp：ユーザに CLI、IDM、IME、ASDM、または SDEE アクションを実行する権限があることを確認します。
- 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、ファイル交換などさまざまなプロセス間通信技術を使用して、MainApp と SensorApp の間を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、ユーザの権限に依存します。

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、サードパーティ製のソフトウェア）は、SDEE プロトコルを使用し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。

Cisco IPS 7.0 の新機能

Cisco IPS 7.0 の新機能は次のとおりです。

- IPS 7.0(4)E4 で AAA RADIUS 認証が導入されました。

  ローカル AAA 認証のほかに、RADIUS サーバがセンサーのユーザ認証を実行するように設定できるようになりました。ユーザ アカウントに対して AAA RADIUS 認証を使用するように IPS を設定できます。これは、大規模な IPS 展開の運用に役立ちます。

- IPS 7.0(1)E3 でグローバル相関が導入されました。
  - Cisco IPS が持つ新しいセキュリティ機能であるシスコ グローバル相関では、長年シスコが蓄積してきた膨大な量のセキュリティ インテリジェンスが使用されます。Cisco IPS は、定期的にシスコの SensorBase ネットワークから脅威情報の更新を受信します。この情報には、インターネット上の既知の脅威（連続攻撃者、ボットネット ハーベスタ、マルウェア アウトブレイク、ダークネットなど）に関する詳細情報が入っています。IPS はこの情報を使用して、悪質な攻撃者が大切な資産を攻撃する前に排除します。その後、グローバル脅威データをシステムに取り込み、悪意のあるアクティビティを検出して未然に阻止します。
  - 10GE インターフェイス カード

    10GE インターフェイス カード（部品番号 IPS-2X10GE-SR-INT および IPS-2X10GE-SR-INT=）は、2 つの 10000 Base-SX（ファイバ）インターフェイスを提供します。IPS 4260 は、1 枚の 10GE インターフェイス カードで合計 2 つの 10GE ファイバ インターフェイスをサポートします。IPS 4270-20 は、2 枚までの 10GE インターフェイス カードで合計 4 つの 10GE ファイバ インターフェイスをサポートします。
  - RDEP イベント サーバの廃止

    IPS 6.1 で RDEP イベント サーバ サービスが廃止され、IPS 7.0(1) で削除されました。RDEP イベント サーバ サービスの代わりに、SDEE イベント サーバ サービスが IPS 5.0 で追加されています。IPS 5.0、5.1、6.0、および 6.1 では、イベントの取得に SDEE イベント サーバを使用するようにモニタリング ツールを移行する期間として、SDEE イベント サーバと RDEP イベント サーバの両方がサポートされていました。IPS 7.0(1) では、モニタリング ツールは SDEE イベント サーバ サービスを使用してイベントを取得する必要があります。

- IME 7.0(1) で、次のグローバル相関機能のサポートが導入されました。
  - IPS が実行されているセンサーでの、グローバル相関機能の設定のサポート。
  - IPS センサーからの、グローバル相関データが含まれているアラートの表示およびモニタリングのサポート。
  - グローバル相関レポートの生成のサポート。

詳細情報

- AAA RADIUS 認証の詳細については、「認証およびユーザ パラメータの設定」（P.4-13）を参照してください。
- グローバル相関の詳細については、第 10 章「グローバル相関の設定」を参照してください。
- 10GE インターフェイス カードの詳細については、『*Installing and Removing Interface Cards in Cisco IPS 4260 and IPS 4270-20*』を参照してください。

ユーザ対話

Cisco IPS とは、次の方法で対話します。

- デバイス パラメータの設定

  システムおよびシステムの機能の初期設定を生成します。これは、頻繁に行うタスクではなく、通常は 1 回だけ行います。システムには合理的なデフォルト値があり、最小限の変更で済みます。Cisco IPS は、CLI、IDM、IME、CSM、ASDM、または SDEE を使用する他のアプリケーションで設定できます。

- 調整

  設定（主に、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一部である分析エンジンの設定）に小さな変更を加えます。システムの調整は、ネットワークにシステムを最初にインストールした後、効率的に動作し、有意な情報だけが生成されるようになるまで、頻繁に行います。カスタム シグニチャの作成、機能のイネーブル化、サービス パックまたはシグニチャ アップデートの適用ができます。Cisco IPS は、CLI、IDM、IME、CSM、ASDM、または SDEE を使用する他のアプリケーションで調整できます。

- 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、または SDEE を使用する他のアプリケーションでアップデートできます。

- 情報の取得

  CLI、IDM、IME、CSM、ASDM、CS MARS、または SDEE を使用する他のアプリケーションで、システムからデータ（ステータス メッセージ、エラー、アラート）を取得できます。

詳細情報

SDEE の詳細については、「SDEE」（P.A-36）を参照してください。

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、特にアクセスが許可されたホストに制限されます。
- Web Server、SSH と SCP、または Telnet を使用して接続を試行するすべてのリモート ホストが認証の対象になります。
- デフォルトでは、Telnet はディセーブルです。Telnet をイネーブルにすることもできます。
- デフォルトでは、SSH アクセスはイネーブルです。
- センサー上で FTP サーバは動作しません。ファイルをリモート コピーするには、SCP を使用します。

- デフォルトで、Web Server 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もできます。
- 不要なサービスはディセーブルになっています。
- CISCO-CIDS-MIB 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ている OID は、MIB で指定されたときに書き込み可能になります。

MainApp

ここでは、MainApp について説明します。内容は次のとおりです。

MainApp について

MainApp には、SensorApp および 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。次に、MainApp は次のサブシステム 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）
- 外部製品のインターフェイス
- インターフェイス マネージャ
- イベント ストア
- ヘルスおよびイベント モニタリング

MainApp の役割

MainApp には、次の役割があります。

- Cisco でサポートされるハードウェア プラットフォームを検証する
- ソフトウェアのバージョンおよび PEP 情報をレポートする

- IPS コンポーネントの起動、停止、およびバージョンのレポートを行う
- ホスト システムを設定する
- システム クロックを管理する
- イベント ストアを管理する
- ソフトウェア アップグレードをインストールおよびアンインストールする

（注） Cisco IPS では、MainApp によって Cisco.com から自動的に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ダウンロードできます。

- オペレーティング システムをシャットダウンまたはリブートする

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（IDS-4260-K9、WS-SVC-IDSM2 など）
- 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 は、ホストの統計情報の収集や、状態およびセキュリティ モニタリング ステータスの報告も行います。

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。内容は次のとおりです。

イベント ストアについて

各 IDS 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主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、侵入イベントのプライオリティ、および時間範囲のリストを指定できます。

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	intrusion イベントのプライオリティ	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 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 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 error status intrusion	medium 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。

これら 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を更新する要求
- 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をリセットする要求
- 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。

ステータス メッセージおよびエラー 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。イベントをイベント ストアにサブスクライブし、SNMP MIB に変換し、パブリック ドメイン SNMP エージェントを通じて宛先に送信します。NotificationApp は、set および get の送信をサポートしています。SNMP GET は、基本的なセンサーの状態に関する情報を提供します。

スパース モードで、NotificationApp は evAlert イベントに含まれる次の情報を送信します。

- 発信元の情報
- イベント ID
- イベントの重大度
- 時間（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
- 参加システムの情報
- アラームの特性

詳細モードで、NotificationApp は evAlert イベントに含まれる次の情報を送信します。

- 発信元の情報
- イベント ID
- イベントの重大度
- 時間（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
- 要約
- インターフェイス グループ
- VLAN
- 参加システムの情報
- アクション
- アラームの特性
- シグニチャ
- IP ログ ID

NotificationApp は、ユーザが定義し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）に基づいたフィルタリングが可能です。NotificationApp は evError イベントに含まれる次の情報を送信します。

- 発信元の情報
- イベント ID
- イベントの重大度

- 時間（UTC および現地時間）
- エラー メッセージ

NotificationApp は、センサーから取得できる、次の一般的な状態およびシステム情報に対する GET をサポートします。

- パケット損失
- パケット拒否
- 生成されたアラーム
- FRP のフラグメント
- FRP のデータグラム
- 初期状態での TCP ストリーム
- 確立状態での TCP ストリーム
- 終了状態での TCP ストリーム
- システムの TCP ストリーム
- 再構築のためにキューイングされた TCP パケット
- アクティブ ノードの合計
- 両方の IP アドレスと両方のポートをキーとする TCP ノード
- 両方の IP アドレスと両方のポートをキーとする UDP ノード
- 両方の IP アドレスをキーとする IP ノード
- センサー メモリの重要な段階
- インターフェイス 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率
- PEP

（注） すべての IPS プラットフォームが PEP をサポートしているわけではありません。

NotificationApp は、次の統計情報を提供します。

- 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

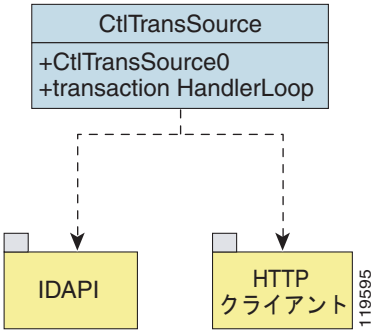
CtlTransSource は、ローカルで開始されたリモート制御トランザクションを HTTP プロトコルを使用してリモートの宛先に転送するアプリケーションです。CtlTransSource は、TLS または非 TLS 接続を開始し、その接続を使用してリモート制御トランザクションを HTTP サーバに伝えます。

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クレデンシャルを確立する必要があります。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 ID を提示することによってクレデンシャルを確立します（基本認証）。認証が成功すると、ユーザ認証を含むクッキーが割り当てられます。この接続に関するすべての要求で、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、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 はリモート制御トランザクションの発信側に、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を返します。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クレデンシャルを持っていないことを示す）を返す場合、transactionHandlerLoop は CtlTransSource 専用のユーザ名とパスワードを使用してリクエスタの ID を認証して、トランザクション要求を再発行します。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、Attack Response Controller（ARC）について説明します。内容は次のとおりです。

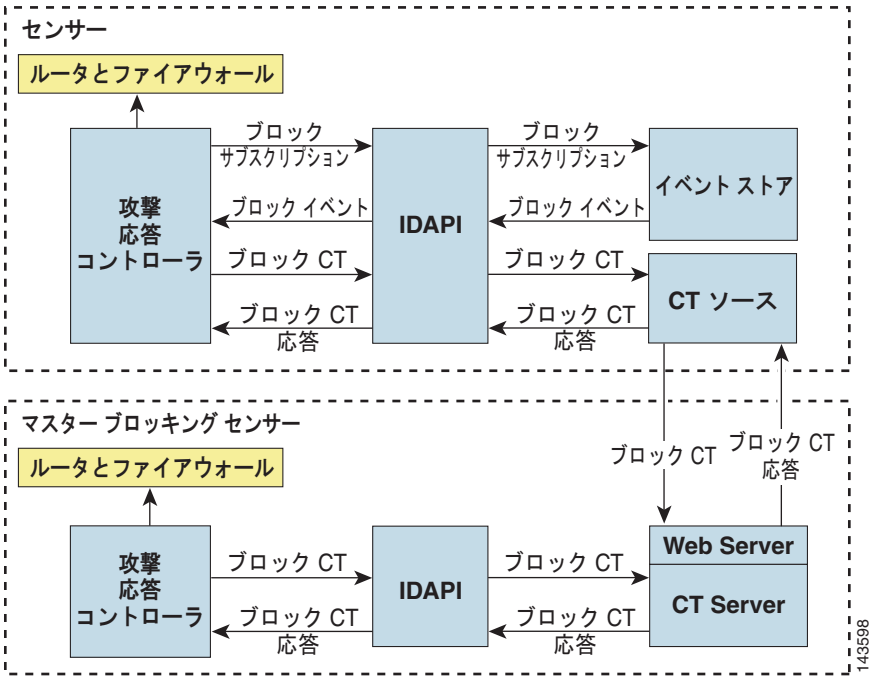
ARC について

ARC は、ルータ、スイッチ、およびファイアウォールのブロッキングを開始および停止し、Cisco IOS 12.3 が動作しているルータのトラフィックをレート制限する IPS アプリケーションです。*block* は、特定のホスト IP アドレスまたはネットワーク アドレスに関する受信および送信トラフィックをブロックするために、デバイスの設定または ACL で使用するエントリです

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 Server 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 つだけ制御できる場合、多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

ARC がデバイスをブロックするように設定すると、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、デバイスごとに接続を維持します。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 タイプでは、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） シスコのファイアウォールは、インターフェイスまたは方向に基づくブロックは行いません。したがって、この設定がシスコの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。ARC は、再イネーブルされると、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）　ファイアウォールを含むすべてのネットワーク デバイスを設定する際は、ARC によるブロッキングをディセーブル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

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 は 65535 までのブロックをサポートしていますが、250 までにすることを推奨します。

（注）　ブロックの数は、インターフェイスと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 以降が実行されている必要があります。

- スーパーバイザ エンジン上で Supervisor Engine ソフトウェア 5.3(1) 以降を実行し、RSM 上で IOS 11.2(9)P 以降を実行する 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 以降を実行する 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
- Cisco ASA 500 シリーズのモデル ASA 5510、ASA 5520、および ASA 5540
- FWSM

（注）　FWSM は、マルチモード管理コンテキストではブロックができません。

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機能します。最初の ACE でパケットが許可された場合、その後の拒否エントリは検索されません。

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 は変更しません。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特殊な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 または postblock ACL は使用できません 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を ARC が維持するローカル ファイル（nac.shun.txt）に（開始タイムスタンプと共に）書き込みます。ARC が起動すると、このファイルを使用して、制御対象のネットワーク デバイスにブロックのアップデートが必要かどうかが判断されます。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 ファイルが正確であるためには、ARC が実行されていない間にシステムの時刻が変更されないことが必要です。

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。

次のシナリオで、ARC が再起動時にどのように状態を維持するかを示します。

シナリオ 1

ARC が停止したときには 2 つのブロックが有効で、そのうちの 1 つは ARC が再起動する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 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、少なくとも送信元と宛先の 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存在する場合、これは無視されてブロックに含まれません。

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　シスコのファイアウォールは、ホストの接続ブロックをサポートしません。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。シスコのファイアウォールは、ネットワーク ブロックもサポートしません。ARC がシスコの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。

シスコのファイアウォールでのブロック

ARC は **shun** コマンドを使用して、ファイアウォールで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断します。

shun コマンドは既存の ACL、条件、アウトバウンド コマンドを置き換えるものではないので、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、ローカル ユーザ名または TACACS+ サーバを使用して、ファイアウォールでの認証をサポートします。ファイアウォールで認証に 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必要があります。

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しているときに、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 スイッチでのブロック

PFC を使用する Catalyst スイッチは、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、次のコマンドが適用されます。

- 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じです）。
  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*

ロガー

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、ロガーを使用してメッセージを記録します。ロガーは、ログ メッセージを debug、timing、warning、error、fatal の 5 段階の重大度のいずれかで送信します。ロガーは、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、「= 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、ロガーはメッセージを 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。

ロガーは、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 evError としてイベント ストアに挿入します。ロガー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。

ロガーは、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。内容は次のとおりです。

AuthenticationApp について

AuthenticationApp には、次の役割があります。

- ユーザの ID 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。

ユーザのローカル認証

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 認証キーなどの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起動します。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。

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。

RADIUS によるユーザの認証

Cisco IPS 7.0(4) では、RADIUS サーバを使用してユーザの認証および許可ができるように、センサーの AAA 機能を拡張する RADIUS 拡張が追加されました。以前は、ローカルに設定されたユーザ名、パスワード、権限を使用したローカル AAA だけが実行できました。

AuthenticationApp は、IPS AAA およびパスワード管理の設定と制御を行います。AuthenticationApp もセンサーの AAA 機能をいくつか提供しますが、ほとんどの AAA 機能は Pluggable Authentication Module（PAM）で提供されます。現在、PAM で提供される機能は、ユーザ パスワードをローカルに設定する機能、およびセンサーでローカルに設定されたユーザ名とパスワードからユーザを認証する機能です。

AuthenticationApp は、センサーで設定されている AAA 方式に対応した PAM コンフィギュレーション ファイルを作成します。これらのコンフィギュレーション ファイルによって、AAA に使用するモジュールおよび方法が PAM に指示されます。これらのファイルは、PAM モジュールが使用する RADIUS サーバのアドレスなどの追加情報も提供します。

RADIUS AAA のサポートのために追加された PAM RADIUS モジュールは、外部 RADIUS サーバとの間の RADIUS 通信を処理します。これらのモジュールは、アプリケーション呼び出しによって提供されたユーザ名とパスワードの情報を AAA サーバに送信し、サーバの応答を処理して、ユーザ認証および許可のステータスを判断します。また、必要な場合、ローカル ユーザ アカウント情報を変更します。

詳細情報

RADIUS 認証を使用するようにセンサーを設定する方法の詳細については、「認証の設定」（P.4-16）を参照してください。

センサーにおけるローカル認証の設定

ユーザが Web Server 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 ID 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 ID 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 によって確認されたキーによってユーザの ID 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 ID 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 ID 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 ID 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。

詳細情報

ローカル認証を使用するようにセンサーを設定する方法の詳細については、「認証の設定」（P.4-16）を参照してください。

センサーにおける RADIUS 認証の設定

RADIUS 認証をイネーブルにすると、センサーはユーザ名とパスワードの認証をリモート RADIUS サーバに委任します。ユーザ名およびパスワードは、認証要求に含められ、設定されている RADIUS サーバに送信されます。サーバの応答によって、ユーザ ログインが認証されるかどうかが決まります。

認証要求をプライマリ RADIUS サーバに送信し、プライマリ サーバが応答しない場合はオプションでセカンダリ サーバに送信するように、センサーを設定できます。セカンダリ サーバが設定されていて、プライマリ サーバの認証要求がタイムアウトになった場合、要求はセカンダリ サーバに送信されます。この場合、セカンダリ サーバからの応答が使用されます。

RADIUS 認証がイネーブルの場合、RADIUS サーバに接続できないときはローカル認証にフォールバックするようにセンサーを設定できます。この場合、センサーは、ローカルに設定されているユーザ アカウントとの照合によって認証を行います。ローカル認証にフォールバックするオプションがディ

セーブルの場合、コンソール ポートで接続していないユーザは、RADIUS サーバが使用不可になるとセンサーにログインできません。この機能では、センサーがローカル認証にフォールバックするのは AAA サーバが応答しないときだけです。AAA サーバがユーザの認証要求を拒否した場合、ローカル認証は試行されません。

RADIUS 認証がイネーブルのとき、コンソール ポートで接続するユーザを認証するオプションが 3 つあります。

- Local Authentication：コンソール ポートで接続するユーザの認証に、ローカル ユーザ アカウントだけが使用されます。RADIUS 認証は試行されません。
- RADIUS and Local Authentication：コンソール ポートのユーザは、まず、RADIUS で認証が試行されます。RADIUS 認証が失敗した場合は、ローカル認証が試行されます。これはデフォルトの設定です。
- RADIUS Authentication：コンソール ポートで接続するユーザの認証に、RADIUS 認証だけが使用されます。このオプションがイネーブルの場合、設定済みの RADIUS 設定が使用されます。RADIUS サーバが応答しないときにローカル認証にフォールバックするように RADIUS が設定されている場合は、この動作がコンソール ポートにも適用されます。

RADIUS で認証される各ユーザに、ユーザ ロールを指定する必要があります。ユーザ ロールは次の 2 つの方法で指定できます。

- センサーのデフォルト ユーザ ロールを指定する。RADIUS を使用してユーザを認証し、RADIUS サーバがセンサー ユーザ ロール情報を提供しなかった場合、ユーザのロールはデフォルト ユーザ ロールに設定されます。デフォルト ユーザ ロールが指定されていない場合、センサー ユーザ ロール情報を提供しない RADIUS サーバでのユーザ認証の試行は失敗します。
- RADIUS サーバの各ユーザまたはユーザ グループのセンサー ユーザ ロールは、テキスト ips-role=<user role> を使用して、Accept Message の Reply 属性に組み込むことができます。ここで、<user role> は administrator、operator、または viewer です。

詳細情報

RADIUS 認証を使用するようにセンサーを設定する方法の詳細については、「認証の設定」（P.4-16）を参照してください。

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 Server 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 ID 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、証明書が信頼されていないというセキュリティ警告のダイアログボックスが表示されます。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 の信頼できる証明書および 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、これは証明書が作成された瞬間から始まる数年の期間です。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/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/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 Server

（注） IPS 6.1 で RDEP イベント サーバ サービスが廃止され、IPS 7.0(1) システム アーキテクチャから削除されました。Web Server は SDEE イベント サーバを使用するようになりました。

Web Server 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 Server は HTTP 1.0 および 1.1 をサポートします。Web Server 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、システムの安全性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。

SensorApp

ここでは、SensorApp について説明します。内容は次のとおりです。

SensorApp について

SensorApp はパケットのキャプチャと分析を実行します。SensorApp のシグニチャによってポリシー違反が検出され、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ように設計されている、プロデューサが供給するプロセッサのパイプラインを通じて流れます。

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、失効したデータベース エントリを期限切れにすることと、時間依存の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管理します。リストの各エントリは、仮想センサー設定で設定できるグローバルな拒否タイマーに基づいて期限切れになります。

- 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 アクションがサポートされます。

  - 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 レーティングのしきい値に関連付けることができます。このしきい値を超えると、アクションが実行されます。

- 統計情報プロセッサ

  このプロセッサは、パケット カウントやパケット到達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

  このプロセッサは、レイヤ 2 関連のイベントを処理します。また、不正な形式のパケットを識別し、処理パスから除去します。不正な形式のパケットの検出に対して、アラート、パケットのキャプチャ、パケットの拒否など、アクション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によって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、シグニチャの状態とフロー データベースを管理します。

- 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。また、センサーがインライン モードの場合に、IP フラグメントの正規化も行います。

- ストリーム再構成プロセッサ

  このプロセッサは、さまざまなストリームベースのインスペクタでパケットの到着順序を保証するために、TCP ストリームを並べ替えます。また、TCP ストリームの正規化も行います。ノーマライザ エンジンでは、アラートや拒否動作をイネーブルまたはディセーブルにできます。

  TCP ストリーム再構成プロセッサのノーマライザにはホールドダウン タイマーがあり、再設定イベントの後でストリーム ステートが再構築されます。このタイマーは設定できません。ホールドダウンの間に、システムを経由したストリームの最初のパケットで、状態の同期を取ります。ホールドダウン時間が経過すると、sensorApp によって設定済みのポリシーが適用されます。このポリシーで、スリーウェイ ハンドシェイクで開かれていないストリームの拒否が呼び出されていると、ホールドダウン時間に休止されていた確立されているストリームは転送されず、タイムアウトになります。ホールドダウン時間に同期されたストリームは継続されます。

- 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く、処理中のパケットを対象とするよう設定されたインスペクタにパケットを送出します。

- スレーブ ディスパッチ プロセッサ

  デュアル CPU システムにのみあるプロセスです。

いくつかのプロセッサは、シグニチャ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、必要に応じて、アラーム チャネルを呼び出してアラートを生成します。

SensorApp では、次のユニットもサポートされています。

- 分析エンジン

  分析エンジンはセンサーの設定を処理します。インターフェイスをマッピングし、シグニチャおよびアラーム チャネル ポリシーも設定済みのインターフェイスにマッピングします。

- アラーム チャネル

  アラーム チャネルは、インスペクタが生成し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、およびイベントのリスク レーティング機能

SensorApp には、次のようなインライン、正規化、およびイベントのリスク レーティング機能があります。

- パケットのインライン処理

  センサーがデータ パスでパケットを処理しているとき、すべてのパケットは明示的にポリシー設定で拒否されていない限り、変更されずに転送されます。TCP 正規化によって、適切なカバレッジを保つために、いくつかのパケットが遅延することがあります。ポリシー違反が検出されたとき、SensorApp はアクションの設定を考慮します。インライン モードでは、パケットの拒否、フローの拒否、攻撃者の拒否など、追加のアクションが使用できます。

  IPS にとって不明なパケット、または IPS に関係のないパケットは、すべて分析されずに、ペアになっている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、ポリシー違反によって拒否される可能性がある以外は、何にも参加せずに転送されます。インライン（または無差別）データ処理に使用されるインターフェイスには、IP スタックが関連付けられません。現在の無差別モードでの 802.1q パケットのサポートがインライン モードに拡張されています。

- IP の正規化

  IP データグラムの意図的または非意図的なフラグメンテーションによって、不正利用が隠蔽され、検出が不可能または困難になることがあります。フラグメンテーションは、ファイアウォールおよびルータ上にあるようなアクセス コントロール ポリシーを迂回するために使用されることもあります。オペレーティング システムごとに、フラグメント化されたデータグラムをキューに格納し、送信するために使用される方法は異なります。エンド ホストがデータグラムを再構成する可能性があるすべての方法をセンサーでチェックする必要があると、センサーは DoS 攻撃に対して脆弱になります。この問題の解決策として、フラグメント化されたすべてのデータグラムをインラインで再構成し、完成したデータグラムだけを転送し、必要に応じてデータグラムを再フラグメント化するという方法があります。IP フラグメンテーション正規化ユニットによって、この機能が実行されます。

- TCP の正規化

  意図的または自然な TCP セッションのセグメンテーションによって、一部の攻撃クラスが隠蔽されることがあります。false positive および false negative なしで、ポリシーを確実に適用できるように、2 つの TCP エンドポイントの状態を追跡し、実際のホスト エンドポイントによって実際に処理されたデータだけを渡す必要があります。TCP ストリームの重複が発生することがありますが、TCP セグメントを再送信する場合を除き、非常にまれです。TCP セッションで上書きが発生することはありません。上書きが行われた場合は、何者かがセキュリティ ポリシーを意図的に逃れしようとしているか、TCP スタックの実装が破損しています。センサーが TCP ポリシーとして機能していない限り、両方のエンドポイントの状態に関するすべての情報を維持することは不可能です。TCP ポリシーとして機能するセンサーの代わりに、セグメントは適切に順序付けられ、ノーマライザは迂回および攻撃に関連するすべての異常なパケットを調べます。

- イベントのリスク レーティング

  イベントのリスク レーティングは、潜在的な悪意のあるアクションを検出するほかに、次の追加情報が含まれます。

  - 攻撃が成功していた場合の重大度
  - シグニチャの忠実度
  - ターゲット ホストの観点から見た、潜在的な攻撃の関連性
  - ターゲット ホストの全体的な価値

  イベントのリスク レーティングは、システムから偽陽性を減らし、アラームの発生源をより正確に制御するために役立ちます。

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、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 スイッチをノーマライザのストリクト モードから非同期モードにすることができます。
- センサーのヘルス メータ：センサー全体の状態に関する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 インスタンスを提供します。
- セキュリティ メータ：脅威カテゴリにアラートを提供し、この情報を赤、黄色、および緑のバケットで報告します。これらのバケットの遷移点は設定可能です。
- フロー ステートのクリア：データベースをクリアして、センサーを再起動したときと同じ新しい状態で開始できます。
- ステータスの再起動：定期的に、センサーの現在の開始および再開ステージを報告します。

パケット フロー

パケットはインターフェイス カードで受信され、IPS 共有ドライバによって、ユーザがマップしたカーネルのメモリ スペースに配置されます。パケットの前には IPS ヘッダーが追加されます。各パケットには、その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受け入れられたか拒否されたかを示すフィールドもあります。

プロデューサは、ユーザがマップした共有のカーネル パケット バッファからパケットをプルして、センサー モデルに合わせてプロセッサを実装するプロセス機能を呼び出します。次の順序で処理が行われます。

- 単一プロセッサの実行

  タイム プロセッサ --> レイヤ 2 プロセッサ --> 拒否フィルタ プロセッサ --> フラグメント再構成プロセッサ --> 統計情報プロセッサ --> データベース プロセッサ --> シグニチャ分析プロセッサ --> ストリーム再構成プロセッサ --> シグニチャ イベント アクション プロセッサ

- デュアル プロセッサの実行

  実行スレッド 1 タイム プロセッサ --> レイヤ 2 プロセッサ --> 拒否フィルタ プロセッサ --> フラグメント再構成プロセッサ --> 統計情報プロセッサ --> データベース プロセッサ --> シグニチャ分析プロセッサ --> スレーブ ディスパッチ プロセッサ --> | 実行スレッド 2 データベース プロセッサ --> ストリーム再構成プロセッサ --> 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

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 ハンドラによる処理へのデータ 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 インスペクション パスからシグニチャ イベントの処理へのシグニチャ イベントの通信を行う領域を表す単位です。

- 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、設定されたリスク レーティングのしきい値の範囲内の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 タイプごとに独自の設定値があります。

- 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取り除きます。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があるアクションのあるシグニチャ イベントです。

  （注）　シグニチャ イベント アクション フィルタではアクションの除外のみを実行でき、新しいアクションを追加することはできません。

  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。

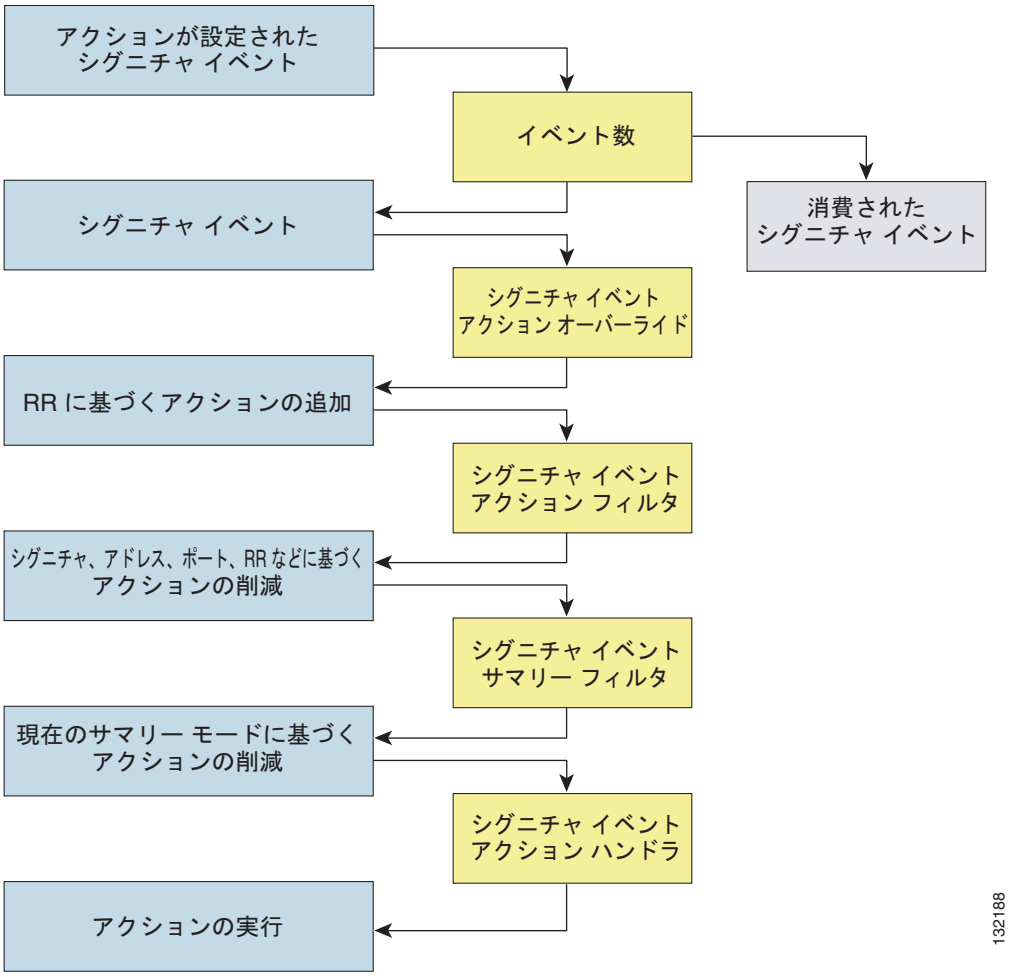
  - シグニチャ ID
  - サブシグニチャ ID
  - 攻撃者のアドレス
  - 攻撃者のポート
  - 攻撃対象のアドレス
  - 攻撃対象のポート
  - リスク レーティングのしきい値範囲
  - 除外するアクション
  - シーケンス ID（任意）
  - 停止または継続ビット
  - アクション フィルタ ライン ビットのイネーブル化
  - 攻撃対象 OS との関連性または OS との関連性

- 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す。イベント ストアに書き込まれる evIdsAlert の可能性があります。

図 A-4（P.A-30）に、シグニチャ イベント アクション プロセッサからのシグニチャ 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動作を示します。この論理フローはアラーム チャネルで受信した設定済みのアクションがあるシグニチャ イベントで開始され、シグニチャ イベントがシグニチャ 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しながら、上から下へと処理されます。

図 A-4　シグニチャ イベント アクション プロセッサでのシグニチャ イベント



CollaborationApp

ここでは、CollaborationApp について説明します。内容は次のとおりです。

- 「CollaborationApp について」（P.A-30）
- 「コンポーネントのアップデート」（P.A-31）
- 「error イベント」（P.A-32）

CollaborationApp について

CollaborationApp は、MainApp および SensorApp のピアです。IDAPI 制御トランザクション、セマフォ、共有メモリ、ファイル交換などさまざまなプロセス間通信技術を使用して、これらの間をインターフェイスします。

グローバル相関サーバと CollaborationApp の間で、レピュテーション アップデートが交換されます。CollaborationApp は、次の 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。

- 規則スコア重み値のセット
- 規則およびアラートと共にレピュテーション スコアの計算に必要な情報を提供する、IP アドレスおよびアドレス範囲のセット
- トラフィックを常に拒否する IP アドレスおよびアドレス範囲のリスト
- センサーがテレメトリ日付をサーバに送信する頻度をサーバで制御できるようにする、ネットワーク参加設定

センサーはネットワーク参加サーバにコラボレーション情報を送信します。センサーはグローバル相関サーバに、利用できるコラボレーション アップデートのリストと、アップデート ファイルをダウンロードするグローバル相関サーバを問い合わせます。

（注）　SensorApp は CollaborationApp よりも先に起動しますが、非同期的に初期化されます。そのため、SensorApp でアップデートを受け入れる準備ができる前に、レピュテーション アップデート サーバが 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、適用しようとすることがあります。アップデート サーバはアップデートをダウンロードし、部分的に処理できますが、アップデートをコミットする前に SensorApp の準備ができるまで待つ必要があります。

詳細情報

グローバル相関とその設定方法については、第 10 章「グローバル相関の設定」を参照してください。

コンポーネントのアップデート

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 サーバとの間でマニフェストを交換します。サーバ マニフェストを解析して、ダウンロードできる新しいアップデートとその場所を判断し、インストールするアップデートのリストを構築します。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 クライアントは各コンポーネントに適用されたアップデートをコミットして、新しいアップデートが使用可能になったことを SensorApp に通知し、各コンポーネントにコミットされた最新のアップデートを反映するようにクライアント マニフェストをアップデートします。

クライアント マニフェストにはセンサーの UDI が含まれており、この UID にはセンサーのシリアル番号、およびセンサーが真正な Cisco IPS センサーであることを確認するためにサーバが使用する、暗号化された共有秘密が含まれています。サーバ マニフェストには各コンポーネントで利用できるアップデート ファイルのリストが含まれています。リストに含まれる各アップデート ファイルごとに、アップデート バージョン、タイプ、順番、場所、ファイル転送プロトコルなどのデータがサーバ マニフェストに含まれています。

アップデート ファイルには 2 種類あります。1 つは、コンポーネントのデータベースにあるすべての既存データを置き換える完全なアップデート ファイルで、もう 1 つは、追加、削除、または情報の置き換え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更新です。すべてのアップデート ファイルがすべてのコンポーネントに適用されると、作業データベースを置き換えることで、一時データベースがコミットされます。

認証および許可は、秘密暗号化メカニズムおよび復号化キー管理によって実現されます。グローバル相関アップデート サーバは、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メカニズム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許可します。グローバル相関アップデート サーバで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、サーバ マニフェストの有効なキーが送信されます。

注意　グローバル相関をイネーブルにして、DNS または HTTP プロキシ サーバを設定しなかった場合、警告メッセージが表示されます。この警告は、グローバル相関をディセーブルにするか、DNS または HTTP プロキシ サーバを追加するように注意を促すものです。

詳細情報

グローバル相関をサポートするために DNS または HTTP プロキシ サーバの追加手順については、「ネットワーク設定値の変更」（P.4-1）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すると、常に evError イベントが生成されます。エラー メッセージはセンサー統計情報に含まれます。次の場合に、重大度が error のステータス メッセージが生成されます。

- センサーのライセンスがない
- DNS または HTTP プロキシ サーバが設定されていない
- マニフェストの交換に失敗した
- アップデート ファイルのダウンロードに失敗した
- アップデートの適用またはコミットに失敗した

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して保存し、DNS または HTTP プロキシ サーバを設定しなかった場合、重大度レベルが warning の evError イベントが生成されます。

詳細情報

センサー統計情報を表示する手順については、「統計情報の表示」（P.17-24）を参照してください。

CLI

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。ここでは、IPS CLI について説明します。内容は次のとおりです。

ユーザ ロール

ユーザ ロールには次の 4 つがあります。

- 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）

  - 仮想センサーの定義
  - 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 または IME 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。

（注） 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。サービス ロールを持つアカウントは、トラブルシューティングの目的でのみ作成してください。管理者権限のあるユーザだけが、サービス アカウントを編集できます。

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.
*************************************************************************************
```

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

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が設定またはリセットされると、root アカウントのパスワードが同じパスワードに設定されます。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 アカウントが削除されると、root アカウントのパスワードはロックされます。

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 で使用できるいくつかのトラブルシューティング機能が組み込まれています。ほとんどのトラブルシューティングでは、サービス アカウントは不要です。非常に特殊な問題をトラブルシューティングするときに、TAC の指示でサービス アカウントを作成する必要が生じる可能性があります。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、これ以外の方法ではディセーブルになっているセンサーへの root 特権アクセスが可能になります。サービス アカウントは、特別な理由で必要になった場合を除いて、作成しないことを推奨します。サービス アカウントが必要なくなったら、削除してください。

通信

ここでは、Cisco IPS で使用される通信プロトコルについて説明します。次の項目について説明します。

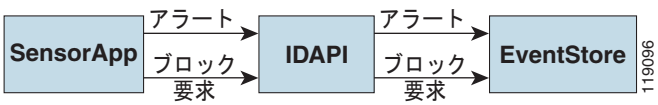
IDAPI

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。受信されたリモート 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 SDEE プロトコルを使用して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 のほとんどの部分は、IDCONF、SDEE、および CIDEE に置き換えられました。

IDCONF

Cisco IPS は、XML ドキュメントを使用して設定を管理します。IDCONF 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、設定ドキュメントの内容を指定するのではなく、設定ドキュメントを作成するフレームワークおよび構築ブロックを指定します。これによって、IPS マネージャおよび CLI が、機能サポート属性を使用して特定のプラットフォームまたは機能で設定することができない機能を無視できるようにするメカニズムが提供されます。

IDCONF メッセージは IDIOM 要求および応答メッセージの中で交換され、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、侵入アラートやステータス イベントを含むさまざまな種類のイベントを生成します。IPS は、IPS で業界最高水準の、独自のプロトコルである SDEE 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伝達します。このプロトコルは、製品に依存しない、セキュリティ デバイス イベントを伝達する標準です。SDEE は、さまざまな種類のセキュリティ デバイスで生成されたイベントの通信に必要な拡張機能を追加するものです。

SDEE を使用してイベントをクライアントに伝達するシステムを SDEE プロバイダと呼びます。SDEE では、イベントを HTTP または HTTP over SSL/TLS プロトコルを使用して転送できると規定されています。HTTP または HTTPS を使用する場合、SDEE プロバイダは HTTP サーバとして機能し、SDEE クライアントは HTTP 要求の発信側になります。

IPS には HTTP または HTTPS 要求を処理する Web Server が含まれています。Web Server は実行時にロード可能な servlet を使用して、さまざまなタイプの HTTP 要求を処理します。各 servlet は、servlet に関連付けられている URL に転送される HTTP 要求を処理します。SDEE Server は、Web サーバの servlet として実装されています。

SDEE Server は、認証された要求だけを処理します。クライアントの ID を認証し、クライアントの権限レベルを判断するために要求が Web サーバから発信された場合、この要求は認証されています。

CIDEE

CIDEE は、Cisco IPS が使用する SDEE の拡張を指定します。CIDEE 標準は、Cisco IPS でサポートされる可能性があるすべての拡張を指定しています。特定のシステムで、CIDEE 拡張のサブセットを実装することができます。ただし、必須として指定されている拡張は、すべてのシステムでサポートされる必要があります。

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 evIdsAlert 要素に対する IPS 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 プロバイダがエラーまたは警告条件を検出したときに、プロバイダによって生成されます。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

  ホストで関係のある可能性がある事象が発生したことを示すために、CIDEE プロバイダによって生成されます。ステータス イベントでは、1 つのイベントで 1 つのメッセージという形式で、さまざまなタイプのステータス メッセージが報告されます。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る事象のタイプに固有のデータ要素のセットが含まれます。多くのステータス メッセージに含まれる情報は、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 ブロックを処理するサービスによってブロック アクションが開始されたことを示すために生成されます。

次に、CDIEE 拡張されたイベントの例を示します。

```
<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 7.0 のファイル構造

Cisco IPS 7.0 のディレクトリ構造は次のとおりです。

- /usr/cids/idsRoot：メインのインストール ディレクトリ。
- /usr/cids/idsRoot/shared：システム リカバリ時に使用されるファイルが格納されます。
- /usr/cids/idsRoot/var：センサーの動作中に動的に作成されるファイルが格納されます。
- /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 7.0 アプリケーションの概要

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、および RADIUS サーバに基づいてユーザを許可および認証します。
Attack Response Controller	ARC は、各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 にネットワーク アクセス コントロール トランザクションを送信します。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。
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有し、すべてのデバイスの全体的な効率を向上します。
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 IPS 管理インターフェイスを提供する Java アプレット。
IME	イベントの表示およびアーカイブのためのインターフェイスを提供する Java アプレット。
InterfaceApp	バイパスおよび物理設定を処理し、ペアになっている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、および管理ステートです。

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
ロガー	アプリケーションのすべてのログ モジュールをログ ファイルに書き出し、エラー メッセージはイベント ストアに書き出します。
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、ステータス、およびエラー イベントによってトリガーされた場合に 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状態に関する情報を提供します。
SDEE サーバ[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 サーバ	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 サーバの servlet 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 サーバの servlet です。