APPENDIX A

システム アーキテクチャの概要

（注） IPS SSP を搭載した Cisco ASA 5585-X は、現在、Cisco IPS 7.1 をサポートする唯一のプラットフォームです。他の Cisco IPS センサーは、いずれも現在 IPS 7.1 をサポートしていません。

（注） IPS SSP を搭載した Cisco ASA 5585-X は、ASA 8.2(4.4) 以降および ASA 8.4(2) 以降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 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 コンフィギュレーション コンポーネントも含まれます。IDM と IME は、コンピュータでホストされる HTTP または HTTPS を使用して IPS と通信します。

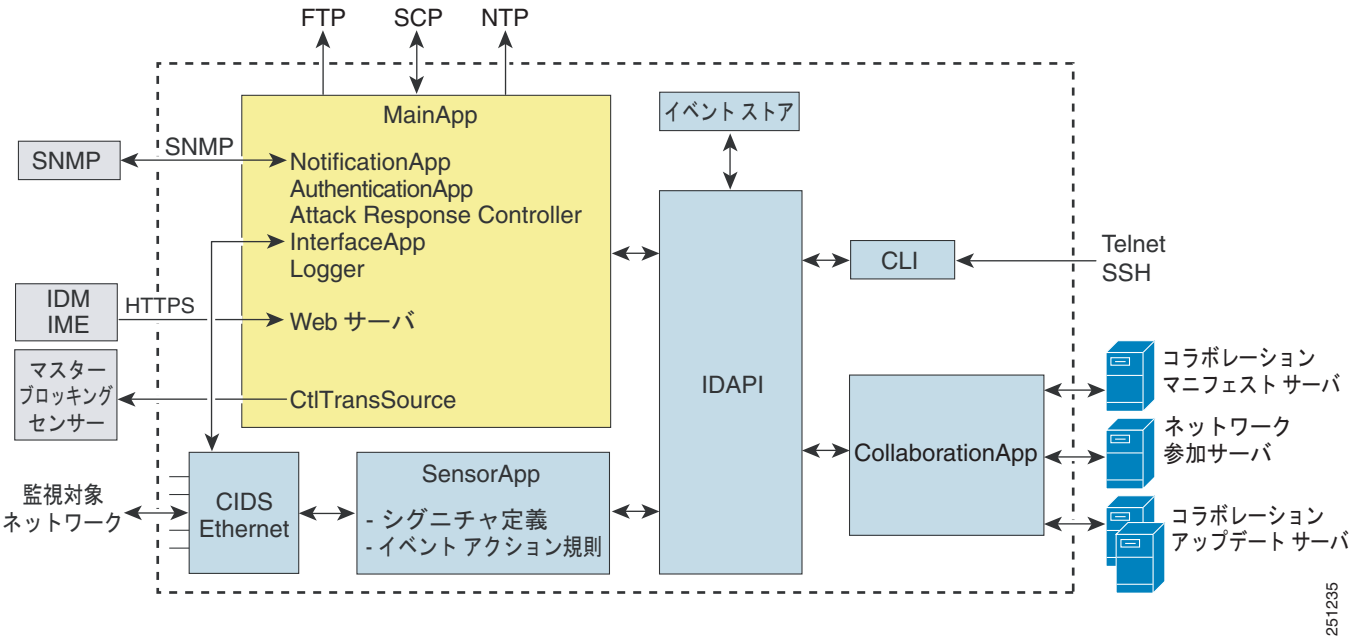
システム設計

（注）　AIP SSC-5 は、グローバル相関機能をサポートしていません。

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デートを行います。次のコンポーネントが含まれます。
  - CtlTransSource（Control Transaction Server）：センサーによる制御トランザクションの送信を許可します。Attack Response Controller（旧称 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（エラー、ステータス、アラートの各システム 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です。
  - InterfaceApp：バイパスおよび物理設定を処理し、ペアにするインターフェイスを定義します。物理設定は、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 Event Store に書き込みます。
  - Attack Response Controller（旧称 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、アラート イベントの発生時にブロッキング機能を提供します。ARC は、ACL を作成して制御対象ネットワーク デバイスに適用するか、または、**shun** コマンド（ファイアウォール）を使用します。
  - 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 Server（HTTP SDEE サーバ）：SDEE プロトコルを使用して、他の IPS デバイスとの Web インターフェイスおよび通信を提供します。IPS サービスの提供には、サーブレットが使用されます。
  - AuthenticationApp：CLI、IDM、IME、ASDM、または SDEE アクションの実行についてユーザが認証されてい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テクノロジーを使用して、MainApp や SensorApp との間に構築されるインターフェイスです。
- 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です。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、共通 API（IDAPI）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、サードパーティ製ソフトウェア）は、SDEE プロトコルで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。

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

  システムおよびその機能の初期設定を生成します。これは頻度の低いタスクで、通常は 1 回だけ行います。システムには、必要な変更を最小限にするように、妥当なデフォルト値が設定されています。Cisco IPS の設定は、CLI、IDM、IME、CSM、ASDM を通して、または SDEE を使用する別のアプリケーションから行うことができます。

- 調整

  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、微調整を加えることができます。システムをネットワークに最初にインストールした後、システムが効率的に動作し、有用な情報が生成されるようになるまで、何度でもシステムを調整できます。カスタム シグニチャの作成、機能のイネーブル化、サービス パックまたはシグニチャ アップデートの適用などを行えます。Cisco IPS の調整は、CLI、IDM、IME、CSM、ASDM を通して、または SDEE を使用する別のアプリケーションから行うことができます。

- 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ように要求することもできます。Cisco IPS のアップデートは、CLI、IDM、IME、CSM、ASDM を通して、または SDEE を使用する別のアプリケーションから行うことができます。

- 情報の取得

  CLI、IDM、IME、CSM、ASDM、CS MARS 経由でシステムから、または SDEE を使用する別のアプリケーションから、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。

セキュリティ機能

Cisco IPS には、次のセキュリティ機能を搭載しています。

- ネットワーク アクセスは、特別にアクセスを許可されたホストに制限されます。
- Web Server、SSH と SCP、または Telnet 経由で接続を試みるリモート ホストはすべて認証されます。
- デフォルトでは、Telnet アクセスはディセーブルです。Telnet をイネーブルにするように選択できます。
- 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、センサー上では実行されません。SCP を使用して、リモートでファイルをコピーできます。
- デフォルトでは、Web Server は TLS または SSL を使用します。TLS と SSL をディセーブルにするように選択できます。
- 不要なサービスはディセーブルにされます。
- CISCO-CIDS-MIB 内では、Cisco MIB ポリシングが必要とする SNMP セットのみが許可されます。パブリック ドメイン SNMP エージェントが実装する OID は、MIB によって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 と CLI を除くすべての IPS コンポーネントが含まれます。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。その後、MainApp は次のサブシステム コンポーネントを起動します。

- Authentication
- Logger
- ARC
- Web Server
- Notification（SNMP）
- External Product Interface
- Interface manager
- Event Store
- Health and security monitoring

MainApp の役割

MainApp には、次の役割があります。

- シスコがサポートするハードウェア プラットフォームの検証
- ソフトウェア バージョンと 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 システムの設定
- システム クロックの管理
- Event Store 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、MainApp は Cisco.com からシグニチャとシグニチャ エンジンのアップデートを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 は、ホストの統計情報の収集、ヘルス状態およびセキュリティのモニタリング ステータスの報告も行います。

Event Store

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

Event Store 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 Event Store 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 Event Store にイベントをインデックス付けします。循環式の Event Store 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。Event Store にアラート イベントを書き込むアプリケーションは SensorApp だけです。ログ、ステータス、エラー イベントは、すべてのアプリケーションが Event Store に書き込みます。

固定サイズのインデックス化された Event Store 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。侵入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	侵入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	プライオリティが low で格納されているすべての intrusion および network access 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。

Event Store 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：設定が更新されたなどの、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。
- network access 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、Event Store 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 block 要求を発行したかを報告する block 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 は、センサーが SNMP トラップとしてアラートとシステム エラー メッセージを送信することを許可します。Event Store にあるイベントにサブスクライブし、SNMP MIB に変換して、パブリック ドメイン SNMP エージェントを介して宛先に送信します。NotificationApp は、set と get の送信をサポートします。SNMP GET は、センサーのヘルス状態に関する基本的な情報を提供します。

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

- 発信元情報
- 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

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

- 発信元情報
- 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 ID

NotificationApp ユーザが定義したフィルタに基づいて、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。

- 発信元情報
- 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ような一般的なヘルス状態およびシステム情報の取得（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
- 初期接続状態での TCP ストリーム
- 接続確立状態での TCP ストリーム
- 接続終了状態での TCP ストリーム
- システムでの TCP ストリーム
- 再構成のためにキューに格納された TCP パケット
- アクティブなノードの総数
- 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 メモリの重大な段階
- インターフェイス ステータス
- コマンドおよび制御パケットの統計情報
- フェールオーバー状態
- システムの動作期間
- 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 をサポートす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 トラップ数
- イベント アクション トラップ数
- SNMP GET 要求の数
- SNMP SET 要求の数

CtlTransSource

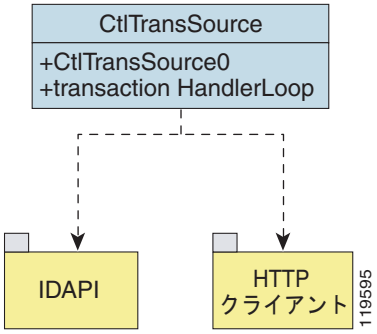
CtlTransSource は、ローカルで開始されたリモート制御トランザクションを HTTP プロトコルを使用してリモートの宛先に転送するアプリケーションです。CtlTransSource は、TLS または非 TLS 接続を開始し、その接続を使用してリモート制御トランザクションを HTTP サーバに伝えます。

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 / 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）。認証に成功すると、ユーザ認証を含む Cookie が割り当てられます。この接続に関するすべての要求では、この Cookie 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 RDEP 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対する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

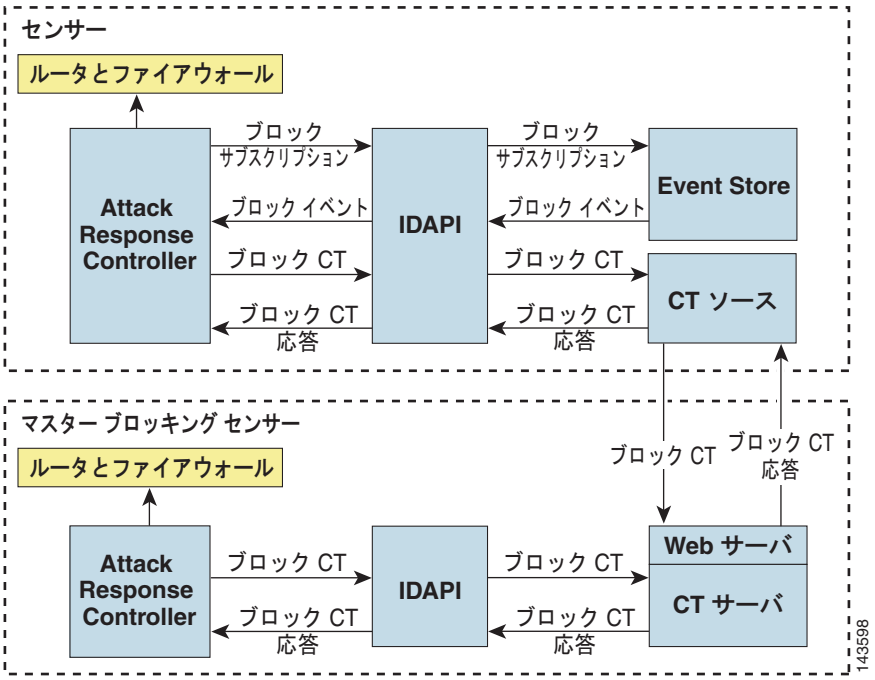
ここでは ARC について説明します。内容は次のとおりです。

ARC について

ARC 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有効化します。

図 A-3 は ARC を図示したものです。

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 アプリケーションのインスタンスは 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 / 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 / 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/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。設定を更新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。CLI または Cisco IPS マネージャを通して、ARC をイネーブルまたはディセーブルに設定できます。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の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

  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、設定は 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 / 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 以降を実行している必要があります。

- 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
- Cisco ASA 500 シリーズ モデル ; ASA 5510、ASA 5520、ASA 5540
- FWSM

（注）　FWSM はマルチ モード管理コンテキストをブロックできません。

ACL と VACL

ARC が制御するインターフェイス / 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 を更新する際、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 / 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 / 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されると、ARC が維持しているローカル ファイル（nac.shun.txt）にすべてのブロッキングとレート制限（および開始時のタイムスタンプ）が書き込まれます。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。

次のシナリオに、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、少なくともソース IP アドレスと宛先 IP アドレスが存在している必要があります。ソース ポートが接続ブロックに存在しても、これは無視されてブロックには含まれません。

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　Cisco ファイアウォールは、ホストの接続ブロッキング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。Cisco ファイアウォールは、ネットワーク ブロッキングもサポートしません。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 ファイアウォールによるブロッキング

ARC は、**shun** コマンドを使用することにより、ファイアウォールに対して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、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 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 AAA 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 スイッチによるブロッキング

Catalyst スイッチには、VACL を使用する PFC フィルタ パケットが搭載されています。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、VACL によるブロッキングを行うための Catalyst スイッチ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*

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 段階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、「= 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）、Event Store 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 sys ログ 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 Event Store 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、さまざまなロギング ゾーンの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、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、IDM や ASDM などの CLI または IPS マネージャ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して、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。

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 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 の認証を確認します。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 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、目的のもの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。

この信頼性を確立するための手順は、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 trusted certificate 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 Server

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 と 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。

（注）　RDEP イベント サーバ サービスは IPS 6.1 で廃止され、現在の IPS システム アーキテクチャから削除されています。現在、Web Server は SDEE イベント サーバを使用しています。

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 はパケットの取り込みと分析を実行します。SensorApp のシグニチャを通してポリシー違反が検出され、違反に関する情報が alert の形式で Event Store 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ように設計されたプロデューサによって供給されたプロセッサのパイプラインを通ります。

SensorApp は次のプロセッサをサポートします。

- 時間プロセッサ

  このプロセッサがタイムスライス カレンダーに格納されたイベントを処理します。主なタスクは、古いデータベース エントリを有効期限切れにすることと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

  このプロセッサが攻撃者拒否機能を処理します。拒否されたソース IP アドレスのリストを維持します。リストの各エントリは、グローバル拒否タイマー（仮想センサー設定で設定可能）に基づいて有効期限が切れます。

- シグニチャ イベント アクション プロセッサ

  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と関連付けることができます。アクションが発生するには、この値を上回る必要があります。

- 統計プロセッサ

  このプロセッサがパケット数やパケット到着レートなどのシステム統計情報を記録します。

- レイヤ 2 プロセッサ

  このプロセッサがレイヤ 2 関連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、処理パスから取り除きます。alert、capture packet、deny packet など、不正な形式のパケットを検出するための実行可能なイベントを設定します。レイヤ 2 プロセッサは、設定したポリシー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

  このプロセッサが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

  フラグメント化された IP データグラムをこのプロセッサが再構成します。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行います。

- ストリーム再構成プロセッサ

  さまざまなストリームベース インスペクタでのパケットが正しい順序で到着するように TCP ストリームの順序をこのプロセッサが変更します。TCP ストリームの正規化も行います。Normalizer エンジンでは、アラート アクションと拒否アクション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、再設定イベントの後にストリームの状態を再構築できるホールドダウン タイマーがあります。このタイマーは設定できません。ホールドダウン期間中、システムは、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強制します。このポリシーが、スリーウェイ ハンドシェイクで開かれなかったストリーム拒否のコールを発信する場合、確立されたストリームの中でホールドダウン期間に休止されていた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に同期されたストリームは続行可能となります。

- シグニチャ分析プロセッサ

  処理中のパケットを対象とするように設定された、ストリームベースではないインスペクタにパケットをこのプロセッサが発送します。

- スレーブ ディスパッチ プロセッサ

  デュアル CPU システムで見られるプロセス。

一部のプロセッサはインスペクタをコールしてシグニチャ分析を実行します。インスペクタはすべて、必要に応じてアラーム チャネルをコールし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ます。

- 分析エンジン

  センサー設定を処理します。インターフェイスとシグニチャおよびアラーム チャネル ポリシーを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

  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に対するアラートの生成です。

インライン、正規化、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、正規化がサポートされていません。

SensorApp には、次のインライン、正規化、イベント 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するとき、ポリシー設定で明示的に拒否されている場合を除き、すべてのパケットは変更が加えられることなく転送されます。TCP 正規化により、適切なカバレッジを確保するため一部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ると、SensorApp はアクションの設定を許可します。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、追加のアクションを実行できます。

  IPS に対して不明または存在しないパケットはすべて、ペアにされたインターフェイスに転送されます。このとき、分析は行われません。ポリシー違反が原因で拒否されるおそれのあるものを除き、すべてのブリッジング プロトコルとルーティング プロトコルが転送されます。インライン（または無差別）データ処理に使用するインターフェイスに関連付けられる IP スタックはありません。無差別モードでの 802.1q パケットの現在のサポートは、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意図しないフラグメンテーションにより、悪用が隠れてしまい、検出が困難になったり不可能になったりすることがあります。フラグメンテーションは、ファイアウォールやルータで実行されるアクセス コントロール ポリシーを回避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、異なる方法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格納してディスパッチします。エンド ホストがデータグラムを再構築できる、考えられるすべての方法をセンサーで確認する必要がある場合、センサーは DoS 攻撃に脆弱になります。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築し、完成したパケットのみを転送し、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で、これを避けることができます。IP Fragmentation Normalization ユニットはこの機能を実行します。

- TCP の正規化

  意図的または自然の TCP セッションのセグメント化を通じて、一部の攻撃クラスが隠れることがあります。ポリシーの強制が、偽陽性または偽陰性なく行われるようにするため、2 つの TCP エンドポイントの状態を追跡し、実際のホスト エンドポイントで実際に処理されるデータのみを通過させる必要があります。TCP ストリームの重なりが起きる可能性がありますが、TCP セグメントの再送以外では非常にまれです。TCP セッションの上書きが起こらないことが必要です。上書きが起きる場合、誰かが意図的にセキュリティ ポリシーを回避しようとしているか、TCP スタックの実装が壊れています。両方のエンドポイントの状態に関する完全な情報を維持することは、センサーが TCP プロキシとして動作しない限り不可能です。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順序付けされ、ノーマライザは回避や攻撃に関連する異常なパケットを探します。

- イベント リスク レーティング

  イベント リスク レーティングには、潜在的に悪意のあるアクションの検出に加え、次の追加情報が組み込まれます。

  - 攻撃が成功した場合の重大度
  - シグニチャの忠実度
  - ターゲット ホストに対する関連性
  - ターゲット ホストの各自にとっての全体的な価値

  イベント リスク レーティングはシステムの偽陽性の抑制に役立ち、アラートの原因についてより精度の高い制御が可能になります。

SensorApp の新機能

SensorApp の新機能は次のとおりです。

- ポリシー テーブル：リスク カテゴリの設定（高、中、低）を提供します。
- 回避保護：ノーマライザについて、インライン インターフェイス モード センサーを厳格な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 ヘルス状態メーター：センサー全体のヘルス状態の統計情報を提供します。
- トップレベル サービス：TCP、UDP、ICMP、IP プロトコルの上位 10 のインスタンスを提供します。
- セキュリティ メーター：アラートを脅威のカテゴリに分類し、この情報を赤、黄、緑のバケットに報告します。これらのバケットのトランジション 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して、再起動したかのようにセンサーを最初から開始できます。
- 再起動ステータス：センサーの現在の開始段階と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 で受信され、IPS 共有ドライバによってカーネル ユーザにマップされたメモリ領域に配置されます。パケットは IPS ヘッダーに付加されます。各パケットには、シグニチャ イベント アクション プロセッサに到達したパケットを許可するか拒否するかを指定するフィールドもあります。

プロデューサは、共有カーネル ユーザにマップされたパケット バッファからパケットをプルし、センサー モデルに適切なプロセッサを実装する処理機能をコールします。次の順序で実行されます。

- シングル プロセッサ実行

  時間プロセッサ --> 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 プロセッサ実行

  実行スレッド 1 時間プロセッサ --> 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、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 ハンドラを介して処理するように、アラーム チャネルのシグニチャ イベントから取得するデータ フローを調整します。次のコンポーネントで構成されます。

- アラーム チャネル

  SensorApp インスペクション パスからのシグニチャ イベントを処理するために通信を行う領域を示すユニット。

- 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、設定されたリスク 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、アクション タイプごとに別々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、リスク レーティングに基づいてアクションを差し引きます。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た可能性のあるアクションを含むグニチャ 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、アクションを差し引くことだけです。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。

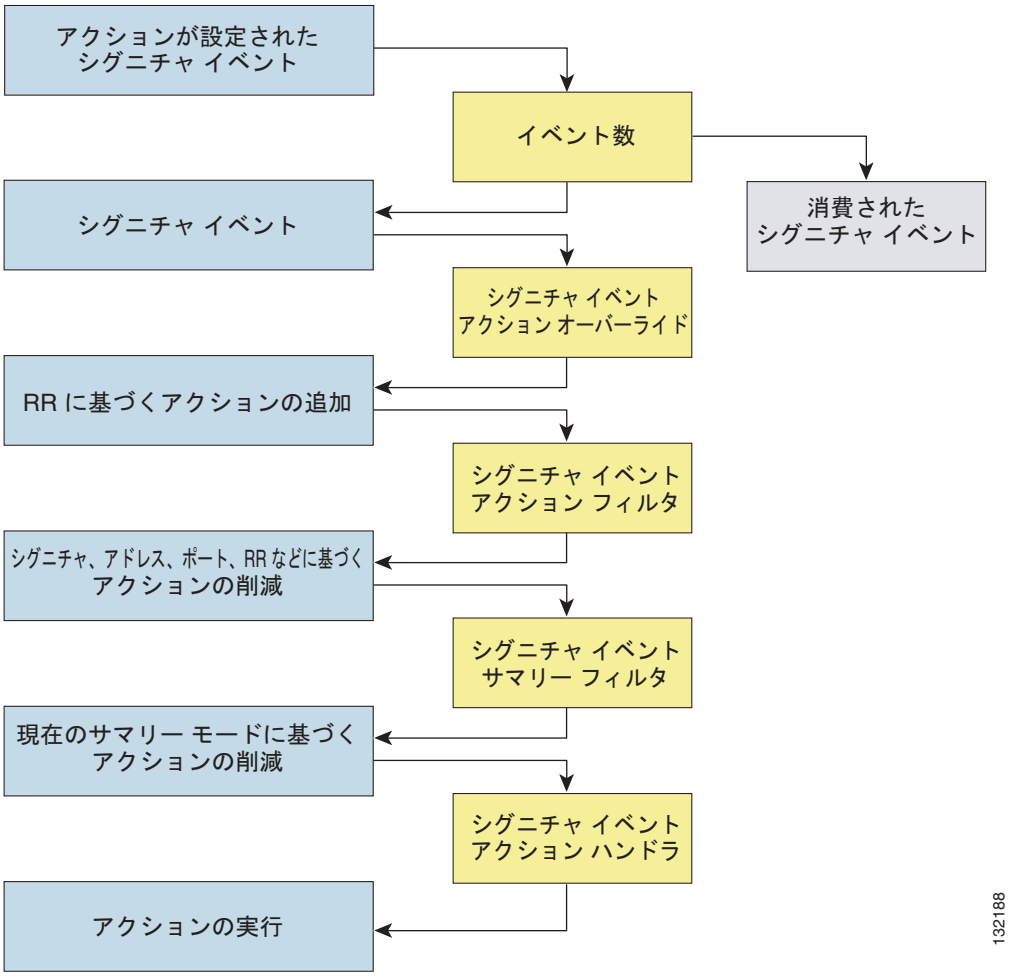
  - シグニチャ ID
  - サブシグニチャ ID
  - 攻撃者のアドレス
  - 攻撃者のポート

  - 攻撃対象のアドレス
  - 攻撃対象者のポート
  - 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、実行されているアクションと、イベント ストアに書き込まれる evIdsAlert（ある場合）です。

図 A-4 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 イベントの論理的な流れと、このイベントに対するアクションで実行される操作を示します。アラーム チャネルから受け取ったアクションが設定されているシグニチャ イベントから開始し、そのイベントは、上から下に向かってシグニチャ 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

（注）　IPS 6.1 および 6.2 は、グローバル相関機能をサポートしていません。

（注）　AIP SSC-5 は、グローバル相関機能をサポートしていません。

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

CollaborationApp は、MainApp と 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テクノロジーを使用してインターフェイスを構築します。

レピュテーション アップデートが、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の通信を行います。

- 規則スコアの重み付け値のセット
- IP アドレスとアドレス範囲のセット。規則とアラートとともに、レピュテーション スコアの計算に必要な情報を提供します。
- IP アドレスとアドレス範囲のリスト。トラフィックは常に拒否されます。
- ネットワーク参加設定。センサーがテレメトリ データをサーバに送信する速度を、サーバ側で制御できるようになります。

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クエリを実行し、コラボレーション アップデートの利用可能なリストと、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を照会します。

（注）　SensorApp は CollaborationApp より前に起動しますが、初期化は非同期に実行されます。このため、レピュテーション アップデート サーバは、SensorApp でアップデートを受け入れる準備が整う前に、グローバル相関の更新をダウンロードして適用を試みることができます。アップデート サーバは、アップデートをダウンロードして部分的に処理することはできますが、アップデートをコミットするには、SensorApp is の準備が整う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 13 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可能な新しいアップデートがあるかどうかを確認し、存在する場合には、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 クライアントは各コンポーネントについてアップデートを適用し、SensorApp に新しいアップデートが利用できることを通知します。そして、クライアント マニフェストを更新し、各コンポーネントに最後にコミットしたアップデートを反映します。

クライアント マニフェストには、センサーの UDI が含まれます。ここには、センサーのシリアル番号と、暗号化された共有秘密（サーバが、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使用する）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。リスト内の各アップデート ファイルでは、サーバ マニフェストにアップデートのバージョン、種類、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 ファイルには、完全なアップデート ファイルと差分アップデート ファイルの 2 種類があります。完全なアップデート ファイルを使用すると、コンポーネントのデータベースに既存のデータはすべて置換されます。差分アップデート ファイルを使用すると、情報の追加、削除、置換を行って既存のレピュテーション データが変更されます。すべてのコンポーネントに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、一時データベースをコミットします。

秘密暗号化メカニズムと復号化キー管理によって認証と認可が実現されます。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認証します。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、サーバ マニフェストに含まれる有効なキーが送信されます。これにより、アップデート ファイルを復号化できるようになります。

注意 グローバル相関をイネーブルに設定し、DNS または HTTP プロキシ サーバを設定していないと、警告メッセージが表示されます。警告では、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通知されます。

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 サーバを追加する手順については、「ネットワークの設定」（P.6-2）を参照してください。

error イベント

グローバル相関のアップデートに失敗すると、evError イベントが生成されます。エラー メッセージは、センサーの統計情報に含まれます。次の条件に該当する場合、重大度レベルが Error のステータス メッセージが発生します。

- センサーのライセンスが取得されていない
- 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保存したときに、DNS または HTTP プロキシ サーバが設定されていないと、重大度レベルが Warning の evError イベントが発生します。

詳細情報

センサーの統計情報を表示するための手順については、「統計情報の表示」（P.19-31）を参照してください。

CLI

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

CLI の概要

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 キーと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、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。トラブルシューティング用には、サービス ロールのアカウントのみを作成してください。管理者権限を持つユーザだけが、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 を通して利用可能な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では、サービス アカウントは必要ではありません。特異な問題のトラブルシューティングを行うため、TAC の指示により、サービス アカウントの作成が必要になることがあります。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し、（通常はディセーブルになっている）センサーへの root 権限アクセスが許可されます。特別な理由により必要となる場合を除き、サービス アカウントを作成しないことをお勧めします。不要になったサービス アカウントは、必ず削除してください。

通信

ここでは、Cisco IPS が使用する通信プロトコルについて説明します。内容は次のとおりです。

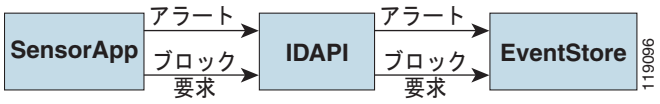
IDAPI

IPS アプリケーションは、内部通信の処理にプロセス間通信 API（IDAPI）を使用します。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 Event Store 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 Event Store 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 IDS イベントは、Event Store に格納されます。
  - Event Store から IPS イベントを読み取ります。
  - Event Store に IPS イベントを書き込みます。

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、alerts 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）　大部分の IDIOM は、IDCONF、SDEE、および CIDEE で置き換えられています。

IDCONF

Cisco IPS は、XML ドキュメントを使用して設定を管理しています。IDCONF は、Cisco IPS 制御トランザクションなどの 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は指定しませんが、設定ドキュメントに基づいてフレームワークとビルディング ブロックが開発されます。これにより、サポートされる機能の属性を通して、IPS マネージャと CLI が特定のプラットフォームや機能で設定不可能な機能を無視できるメカニズムが提供されます。

IDCONF メッセージは、IDIOM 要求内部と応答メッセージにラップされます。

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 イベントやステータス イベントなど各種イベントを生成します。IPS は、独自の IPS 業界最高レベルのプロトコル SDEE（セキュリティ デバイスのイベントを伝える、製品に依存しない標準）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。SDEE は、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伝えるために必要な拡張性機能を追加します。

クライアントにイベントを伝達するために SDEE を使用するシステムは、SDEE プロバイダーと呼ばれます。SDEE は、HTTP または HTTP over SSL と TLS プロトコルを使用してイベントを送信できることを指定します。HTTP または HTTPS を使用する場合、SDEE クライアントは HTTP 要求の発信側となり、SDEE プロバイダーは HTTP サーバとして動作します。

IPS には、HTTP または HTTPS 要求を処理する Web Server が含まれます。Web Server はロード可能なランタイム サーブレットを使用して、さまざまな種類の HTTP 要求を処理します。各サーブレットは、サーブレットに関連付けられた URL に向けられた 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ます。

SDEE サーバは、認証された要求のみを処理します。要求は、発信元が Web サーバあり、クライアントの身元を認証でき、かつクライアントの権限レベルを特定できる場合に認証されます。

CIDEE

CIDEE は、Cisco IPS で使用される SDEE への拡張子を指定します。CIDEE 標準では、Cisco IPS でサポートされる使用可能なすべての拡張子が指定されています。一部のシステムでは、CIDEE 拡張子のサブセットを実装できます。ただし、必須として指定されている拡張子は、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 SDEE evIdsAlert 要素に対し、Cisco IPS 固有のセキュリティ デバイス イベントと IPS 拡張子を指定します。

CIDEE は次のイベントをサポートします。

- evError：エラー イベント

  プロバイダーがエラーまたは警告の条件を検出したときに、CIDEE プロバイダー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

  ホストに注意すべき事項が発生したことを知らせるときに、CIDEE プロバイダーによって生成されます。ステータス イベントには、さまざまな種類のステータス メッセージが報告されます。1 つのイベントにつき、1 つのメッセージが報告されます。各種ステータス メッセージには、ステータス メッセージが説明している事項に固有のデータ要素のセットが含まれます。ステータス メッセージの多くに含まれる情報は、監査の面で有益な情報です。エラーと警告は状態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によってブロック アクションを開始する必要があることを知らせるために生成されます。

次に、CDIEE 拡張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 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：メインのインストール ディレクトリ。
- /usr/cids/idsRoot/shared：システムの回復中に使用されるファイルが格納されます。
- /usr/cids/idsRoot/var：センサーの実行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：Event Store 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 Server 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 Server 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 Event Store 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通して他のデバイスと情報を共有し、すべてのデバイスの総合的な有効性を高めます。
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 アプレットです。
IME	イベントの表示やアーカイブを行うためのインターフェイスを提供する Java アプレットです。
InterfaceApp	バイパスおよび物理設定を処理し、ペアにするインターフェイスを定義します。物理設定は、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、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。
SDEE Server[3]	リモート クライアントからのイベントの要求を受け入れます。

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
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。