APPENDIX B

シグニチャ エンジン

この付録では、IPS シグニチャ エンジンについて説明します。次のような構成になっています。

シグニチャ エンジンについて

シグニチャ エンジンとは、特定のカテゴリ内の多数のシグニチャをサポートするために設計されている Cisco IPS のコンポーネントです。エンジンは、パーサーとインスペクタで構成されています。各エンジンにはパラメータのセットがあり、パラメータには使用可能な範囲や値のセットがあります。

（注） Cisco IPS エンジンでは、標準的な正規表現がサポートされています。

Cisco IPS には、次のシグニチャ エンジンが搭載されています。

- AIC：Web トラフィックを詳細に分析します。ACI エンジンは、HTTP セッションに対してより細かな制御を実行して、HTTP プロトコルの悪用を防ぎます。また、インスタント メッセージングや GotoMyPC など、特定のポートを介してトンネリングを試みるアプリケーションに対する管理制御を行います。AIC を使用して FTP トラフィックを検査し、発行されたコマンドを制御することもできます。AIC エンジンには、AIC FTP と AIC HTTP という 2 つの種類があります。
- Atomic：現在、Atomic エンジンは 4 つのエンジンに組み込まれ、複数のレベルを選択できます。IP + TCP など、1 つのシグニチャ内でレイヤ 3 属性とレイヤ 4 を組み合わせることができます。Atomic エンジンでは、標準的な正規表現がサポートされます。
  - Atomic ARP：レイヤ 2 ARP プロトコルを検査します。Atomic ARP エンジンが異なるのは、大半のエンジンはレイヤ 3 IP プロトコルに基づいているためです。
  - Atomic IP Advanced：IPv6 レイヤ 3 と ICMPv6 レイヤ 4 のトラフィックを検査します。
  - Atomic-IP：IP プロトコル パケット、および関連付けられているレイヤ 4 トランスポート プロトコルを検査します。

    このエンジンでは、IP およびレイヤ 4 ヘッダーのフィールドと一致する値を指定し、正規表現を使用してレイヤ 4 ペイロードを検査することができます。

    （注） すべての IP パケットは、Atomic IP エンジンによって検査されます。このエンジンは、4.x Atomic ICMP、Atomic IP Options、Atomic L3 IP、Atomic TCP、および Atomic UDP の各エンジンを置き換えるものです。

  - Atomic IPv6：不正な形式の IPv6 トラフィックによって引き起こされる 2 つの IOS 脆弱性を検出します。
- Fixed：一定の項目数まで、並列の正規表現照合を実行してから、1 つの正規表現テーブルを使用して検査を停止します。ICMP、TCP、および UDP の 3 つの Fixed エンジンがあります。
- Flood：ホストとネットワークに向けられた ICMP および UDP フラッディングを検出します。Flood Host と Flood Net の 2 つの Flood エンジンがあります。
- Meta：スライディング時間間隔内に、関連した方法で発生するイベントを定義します。このエンジンは、パケットではなくイベントを処理します。
- Multi String：1 つのシグニチャに対して複数の文字列を照合することにより、レイヤ 4 のトランスポート プロトコルとペイロードを検査します。このエンジンは、ストリーム ベースの TCP と、単一の UDP パケットおよび ICMP パケットを検査します。
- Normalizer：IP および TCP ノーマライザが機能する方法を設定し、IP および TCP ノーマライザに関連するシグニチャ イベントを設定します。RFC 準拠を強制できます。
- Service：特定のプロトコルを処理します。Service エンジンには、次のプロトコル タイプがあります。
  - DNS：DNS（TCP および UDP）トラフィックを検査します。
  - FTP：FTP トラフィックを検査します。
  - Generic：カスタム サービスおよびペイロードをデコードし、ネットワーク プロトコルを総合的に分析します。
  - H225：VoIP トラフィックを検査します。ネットワーク管理者が、VoIP ネットワークに着信している SETUP メッセージが有効であり、ポリシーに規定される境界内であることを確認するために役立ちます。また、url-ids、email-ids、および display information などのアドレスおよび Q.931 文字列フィールドが、特定の長さに従い、潜在的な攻撃パターンが含まれていないことを確認するために役立ちます。

  - HTTP：HTTP トラフィックを検査します。WEBPORTS 変数では、HTTP トラフィックの検査ポートを定義します。
  - IDENT：IDENT（クライアントおよびサーバ）トラフィックを検査します。
  - MSRPC：MSRPC トラフィックを検査します。
  - MSSQL：Microsoft SQL トラフィックを検査します。
  - NTP：NTP トラフィックを検査します。
  - P2P：P2P トラフィックを検査します。
  - RPC：RPC トラフィックを検査します。
  - SMB Advanced：Microsoft SMB パケットと Microsoft DCE/RPC (MSRPC) over SMB パケットを処理します。

注意 SMB エンジンは、SMB Advanced エンジンによって置き換えられました。SMB エンジンは IDM、IME、および CLI でまだ表示されますが、そのシグニチャは廃止されました。新しいシグニチャには、対応する古いシグニチャ ID の廃止パラメータ セットがあります。新しい SMB Advanced エンジンを使用して、SMB エンジン内にあるカスタム シグニチャを書き換えてください。

  - SNMP：SNMP トラフィックを検査します。
  - SSH：SSH トラフィックを検査します。
  - TNS：TNS トラフィックを検査します。

- State：SMTP などのプロトコル内の文字列をステートフル検索します。現在、State エンジンには、状態遷移を定義するために使用される隠れたコンフィギュレーション ファイルがあります。これにより、シグニチャ アップデートで新しい状態定義を配信できます。
- String：ICMP、TCP、または UDP のプロトコルに基づいて正規表現文字列を検索します。String ICMP、String TCP、および String UDP の 3 つの String エンジンがあります。
- Sweep：1 つのホスト（ICMP と TCP）、宛先ポート（TCP と UDP）、および 2 つのノード間で RPC 要求を送受信する複数のポートからのスイープを分析します。Sweep と Sweep Other TCP の 2 つの Sweep エンジンがあります。
- Traffic Anomaly：TCP、UDP、およびその他のトラフィックでワームを検査します。
- Traffic ICMP：TFN2K、LOKI、DDOS などの非標準プロトコルを分析します。パラメータを設定できるのは 2 つのシグニチャだけです。
- Trojan：BO2K および TFN2K などの非標準プロトコルからのトラフィックを分析します。Bo2k、Tfn2k、および UDP の 3 つの Trojan エンジンがあります。これらのエンジンには、ユーザが設定できるパラメータはありません。

Master エンジン

Master エンジンは、他のエンジンに構造とメソッドを提供し、コンフィギュレーションからの入力とアラート出力を処理します。ここでは、Master エンジンについて説明します。内容は次のとおりです。

一般パラメータ

次のパラメータは、Master エンジンの一部であり、すべてのシグニチャに適用されます（そのシグニチャ エンジンに対して有意な場合）。

表 B-1 に、一般的な Master エンジンのパラメータを示します。

表 B-1　Master エンジンのパラメータ

パラメータ	説明	値
signature-id	このシグニチャの ID を指定します。	*number*
sub-signature-id	このシグニチャのサブ ID を指定します。	*number*
alert-severity	アラートの重大度を指定します。 • 危険なアラート • 中レベルのアラート • 低レベルのアラート • 情報アラート	• high • medium • low • informational（デフォルト）
sig-fidelity-rating	このシグニチャの忠実度のレーティングを指定します。	0 ～ 100（デフォルト = 100）
promisc-delta	アラートの重大度を決定するために使用されるデルタ値を指定します。	0 ～ 30（デフォルト = 5）
sig-name	シグニチャの名前を指定します。	*sig-name*
alert-notes	アラート メッセージに含まれる、このシグニチャに関する追加情報を提供します。	*alert-notes*
user-comments	このシグニチャに関するコメントを提供します。	*comments*
alert-traits	このシグニチャについて文書化する特性を指定します。	0 ～ 65335
release	シグニチャが最後に更新されたリリースを示します。	*release*
signature-creation-date	シグニチャが作成された日付を指定します。	—
signature-type	シグニチャのカテゴリを指定します。	• anomaly • component • exploit • other vulnerability
engine	シグニチャが属するエンジンを指定します。 **（注）** エンジン固有のパラメータは、engine カテゴリの下に表示されます。	—
event-count	アラートが生成されるまでのイベントの発生回数を指定します。	1 ～ 65535（デフォルト = 1）

表 B-1　Master エンジンのパラメータ（続き）

パラメータ	説明	値
event-count-key	このシグニチャに関するイベントをカウントするストレージ タイプを指定します • 攻撃者のアドレス • 攻撃者と攻撃対象のアドレス • 攻撃者のアドレスと攻撃対象のポート • 攻撃対象のアドレス • 攻撃者と攻撃対象のアドレスおよびポート	• Axxx • AxBx • Axxb • xxBx • AaBb
specify-alert-interval {yes | no}	アラート間隔をイネーブルにします。 • alert-interval：イベント数がリセットされるまでの秒数を指定します。	2 ～ 1000
status	シグニチャが、イネーブルとディセーブルのいずれであるか、アクティブと廃棄のいずれであるかを指定します。	enabled | retired {yes | no}
obsoletes	新しいシグニチャによって、古いシグニチャがディセーブルになるかどうかを示します。	—
vulnerable-os-list	パッシブ OS フィンガープリントと組み合わせた場合、IPS は、特定の攻撃がターゲット システムに関連する可能性が高いかどうかを判断できます。	aix bsd general-os hp-ux ios irix linus mac-os netware other solaris unix windows windows-ut windows-nt-2k-xp
mars-category {yes | no}	MARS 攻撃カテゴリにシグニチャをマッピングします。[1]	—

1. コンフィギュレーションで設定し、アラートで表示することができる静的な情報カテゴリです。詳細については、MARS ドキュメントを参照してください。

Promiscuous Delta

Promiscuous Delta は、無差別モードで、特定のアラートのリスク レーティングを低下させます。センサーはターゲット システムの属性を認識せず、また無差別モードではパケットを拒否できないため、無差別アラートの優先順位を（優先順位の低いリスク レーティングに基づいて）低く設定しておくと役立ちます。そうすることで、管理者は優先順位の高いリスク レーティング アラートの調査に集中できます。インライン モードでは、センサーが違反パケットを拒否し、違反パケットがターゲット ホストに到達しないようにできるため、ターゲットが脆弱であっても問題になりません。ネットワーク上の攻撃が不可能になるため、IPS はリスク レーティング値を減少しません。サービス、OS、またはアプ

リケーションに固有ではないシグニチャの Promiscuous Delta は 0 です。OS、サービス、またはアプリケーションに固有のシグニチャの場合、5、10、または 15 の Promiscuous Delta がカテゴリごとに 5 つのポイントから計算されます。

注意　シグニチャの promisc-delta 設定は、変更しないことを推奨します。

Obsoletes

Cisco シグニチャ チームは、obsoletes フィールドを使用して、改善された新しいシグニチャによって置き換えられ、廃止された古いシグニチャと、改善されたエンジンのインスタンスが使用可能になったときにディセーブルになったエンジンのシグニチャを示します。

Vulnerable OS List

シグニチャの脆弱 OS 設定とパッシブ OS フィンガープリントを組み合わせた場合、IPS は、特定の攻撃がターゲット システムに関連する可能性が高いかどうかを判断できます。攻撃に関連性があることが判明した場合、生成されるアラートのリスク レーティング値は、急上昇します。関連性が確認されない場合、通常、パッシブ OS フィンガープリント リストにはエントリがないため、リスク レーティングは変更されません。パッシブ OS フィンガープリント エントリが存在し、シグニチャの脆弱 OS 設定と一致しない場合、リスク レーティングの値は減少します。リスク レーティングの増加または減少のデフォルト値は、+/- 10 ポイントです。

詳細情報

- 無差別モードの詳細については、「無差別モードについて」（P.6-17）を参照してください。
- パッシブ OS フィンガープリントの詳細については、「OS ID の設定」（P.7-26）を参照してください。

Alert Frequency

Alert Frequency パラメータの目的は、stick などの IDS DoS ツールに対抗するために、イベント ストアに書き込まれるアラートの量を削減することです。Fire All、Fire Once、Summarize、および Global Summarize という 4 つのモードがあります。サマリー モードは、現在のアラート量に応じて動的に変わります。たとえば、シグニチャを Fire All に設定できますが、一定のしきい値に達するとサマライズが開始されます。

表 B-2 に、Alert Frequency パラメータを示します。

表 B-2　Master エンジンの Alert Frequency パラメータ

パラメータ	説明	値
alert-frequency	アラートをグループ化するためのサマリー オプション。	—
summary-mode	サマライズに使用するモード。	—
fire-all	すべてのイベントについてアラートを起動します。	—
fire-once	1 回だけアラートを起動します。	—
global-summarize	攻撃者や攻撃対象の数に関係なく 1 回だけアラートが起動されるようにアラートをサマライズします。	—
summarize	アラートをサマライズします。	—
summary-threshold	アラート数のしきい値。この値を超えるとシグニチャはサマリー モードに送られます。	0 ～ 65535

表 B-2　Master エンジンの Alert Frequency パラメータ（続き）

パラメータ	説明	値
global-summary-threshold	イベント数のしきい値。この値を超えるとアラートはグローバル サマリーにサマライズされます。	1 ～ 65535
summary-interval	各サマリー アラートで使用される時間（秒数）。	1 ～ 1000
summary-key	シグニチャをサマライズするストレージ タイプ：	Axxx
	• 攻撃者のアドレス	AxBx
	• 攻撃者と攻撃対象のアドレス	Axxb
	• 攻撃者のアドレスと攻撃対象のポート	xxBx
	• 攻撃対象のアドレス	AaBb
	• 攻撃者と攻撃対象のアドレスおよびポート	

イベント アクション

（注）　次のイベント アクションの大半は、特定のエンジンに該当するものを除き、各シグニチャ エンジンに属します。

次のイベント アクション パラメータは、各シグニチャ エンジンに属します（そのシグニチャ エンジンに対して有意な場合）。

- アラートおよびログ アクション
  - produce-alert：evIdsAlert をイベント ストアに書き込みます。

    （注）　シグニチャに対してアラートをイネーブルにした場合、produce-alert アクションは自動的に実行されません。イベント ストアにアラートを作成するには、produce-alert を選択する必要があります。2 番目のアクションを追加し、イベント ストアにアラートを送信する場合は、produce-alert を含める必要があります。また、イベント アクションを設定するたびに、新しいリストが作成され、古いリストは置き換えられます。各シグニチャに対して必要なすべてのイベント アクションを含めます。

    （注）　produce-alert イベント アクションは、グローバル相関によってイベントのリスク レーティングが増加し、deny-packet-inline または deny-attacker-inline のいずれかのイベント アクションが追加されたときに、イベントに追加されます。

  - produce-verbose-alert：evIdsAlert に、攻撃パケットのエンコードされたダンプを含めます（切り捨てられることがあります）。
  - log-attacker-packets：攻撃者のアドレスを含むパケットの IP ロギングを開始し、アラートを送信します。
  - log-victim-packets：攻撃対象のアドレスを含むパケットの IP ロギングを開始し、アラートを送信します。
  - log-pair-packets：（インライン モードのみ）攻撃者と攻撃対象のアドレス ペアを含むパケットの IP ロギングを開始します。

  - request-snmp-trap：NotificationApp に要求を送信して、SNMP 通知を実行します。
- 拒否アクション
  - deny-packet-inline：（インライン モードのみ）このパケットを送信しません。

    （注） deny-packet-inline に対するイベント アクションのオーバーライドは、保護されているため削除できません。オーバーライドを使用しない場合は、そのエントリに対して、override-item-status をディセーブルに設定します。

  - deny-connection-inline：（インライン モードのみ）TCP フローで、現在のパケットおよび将来のパケットを送信しません。
  - deny-attacker-victim-pair-inline：（インライン モードのみ）指定された期間、この攻撃者と攻撃対象のアドレスのペアについては、現在のパケットおよび将来のパケットを送信しません。
  - deny-attacker-service-pair-inline：（インライン モードのみ）指定された期間、この攻撃者のアドレスと攻撃対象のポートのペアについては、現在のパケットおよび将来のパケットを送信しません。
  - deny-attacker-inline：（インライン モードのみ）指定された期間、攻撃者のアドレスからの、現在のパケットおよび将来のパケットを送信しません。

    （注） これは最も厳しい拒否アクションです。単一の攻撃者アドレスからの現在および将来のパケットが拒否されます。各拒否アドレスは、拒否を開始させた最初のイベントから *X* 秒でタイムアウトします。X は、設定した秒数です。**clear denied-attackers** コマンドで、すべての拒否された攻撃者エントリをクリアできます。これにより、そのアドレスはネットワークで再び許可されます。

  - modify-packet-inline：（インライン モードのみ）パケット データを変更して、エンドポイントによるパケットの処理に関して、あいまいな部分を取り除きます。

    （注） modify-packet-inline イベント アクションは、Normalizer エンジンの一部です。パケットを修正し、不正なチェックサム、範囲外の値、その他の RFC 違反などの不正な問題を訂正します。

- その他のアクション

  （注） IPv6 は、イベント アクション、request-block-host、request-block-connection、または request-rate-limit をサポートしていません

  - request-block-connection：この接続のブロックを ARC に要求します。
  - request-block-host：この攻撃者ホストのブロックを ARC に要求します。
  - request-rate-limit：レート制限の実行を ARC に要求します。
  - reset-tcp-connection：TCP リセットを送信し、TCP フローをハイジャックして終了します。

パケットのインライン拒否について

deny-packet-inline をアクションとして設定されたシグニチャ、または deny-packet-inline をアクションとして追加するイベント アクション オーバーライドに対しては、次のアクションを実行できます。

- droppedPacket

- deniedFlow
- tcpOneWayResetSent

Deny Packet Inline アクションは、アラート内で破棄されたパケット アクションとして表されます。TCP 接続に対して Deny Packet Inline を実行すると、自動的に Deny Connection Inline にアップグレードされ、アラート内で拒否されたフローとして見なされます。IPS がパケットを 1 つだけ拒否すると、TCP は同じパケットの送信を繰り返し試みます。そのため、IPS は接続全体を拒否して、パケットが再送信されないようにします。

Deny Connection Inline を実行すると、IPS は、TCP 一方向リセットも自動的に送信します。これは、アラート内で送信された TCP 一方向リセットとして表示されます。IPS が接続を拒否すると、クライアント（通常は攻撃者）とサーバ（通常は攻撃対象）の両方で接続が開いたままになります。開いた接続が多すぎると、攻撃対象でリソースの問題が発生する可能性があります。そのため、IPS は TCP リセットを攻撃対象に送信して攻撃対象側（通常はサーバ）の接続を閉じ、攻撃対象のリソースを保護します。また、フェールオーバーも防止することで、接続が別のネットワーク パスにフェールオーバーして攻撃対象に到達するのを防ぎます。IPS は、攻撃者側が開いたままとなり、攻撃者側からのすべてのトラフィックを拒否します。

正規表現の構文

正規表現（Regex）は、テキストを記述する手段として、強力で柔軟性のある表記言語です。パターンマッチングでは、正規表現によりあらゆる任意のパターンを簡潔に表記できます。

表 B-3 は、IPS シグニチャで使用できる正規表現の構文をまとめたものです。

表 B-3　シグニチャの正規表現の構文

メタ文字	名前	説明
?	疑問符	0 回または 1 回の繰り返し。
*	星印（アスタリスク）	0 回以上の繰り返し。
+	プラス	1 回以上の繰り返し。
{x}	量指定子	ちょうど *X* 回の繰り返し。
{x,}	最小量指定子	少なくとも *X* 回の繰り返し。
.	ドット	改行（0x0A）以外の任意の 1 文字。
[abc]	文字クラス	リスト内の任意の 1 文字。
[^abc]	否定文字クラス	リストにない任意の 1 文字。
[a-z]	文字範囲クラス	範囲内（両端も含む）の任意の 1 文字。
( )	カッコ	他のメタ文字の適用範囲を制限する際に使用する。
|	論理和（OR）	このメタ文字によって区切られている複数の表現のいずれかと一致します。
^	キャレット	行の先頭。
¥*char*	エスケープ文字。	*char* がメタ文字である場合も含めて、*char* そのものと一致する。

表 B-3　シグニチャの正規表現の構文（続き）

メタ文字	名前	説明
char	文字	char がメタ文字でない場合は、char そのものと一致する。
¥r	復帰	復帰文字（0x0D）と一致する。
¥n	改行	改行文字（0x0A）と一致する。
¥t	タブ	タブ文字（0x09）と一致する。
¥f	フォーム フィード	フォーム フィード文字（0x0C）と一致する。
¥xNN	エスケープされた 16 進数文字	16 進コード 0xNN（0<=N<=F）を持つ文字と一致する。
¥NNN	エスケープされた 8 進数文字	8 進コード NNN（0<=N<=8）を持つ文字と一致する。

繰り返し演算子ではいずれの場合も、該当する文字列のうち最も短いものが一致対象となります。一方、それ以外の演算子では、その適用範囲に最大限多くの文字が取り込まれるため、該当する文字列のうち最も長いものが一致対象となります。

表 B-4 は、正規表現のパターンの例を示したものです。

表 B-4　正規表現のパターン

一致対象	正規表現
Hacker	Hacker
Hacker または hacker	[Hh]acker
bananas、bananananas、banananananas など、一定の規則で構成されたすべての文字列	ba(na)+s
同じ行の中にある foo と bar の間に改行以外の文字が 0 個以上ある文字列	foo.*bar
foo または bar	foo|bar
moon または soon	(m|s)oon

AIC エンジン

アプリケーション検査および制御（AIC）エンジンは、HTTP Web トラフィックを検査し、FTP コマンドを適用します。ここでは、AIC エンジンとそのパラメータについて説明します。内容は次のとおりです。

AIC エンジンについて

AIC は、Web トラフィックの詳細な分析を行います。HTTP セッションに対してより細かな制御を実行して、HTTP プロトコルの悪用を防ぎます。また、インスタント メッセージングや GotoMyPC など、特定のポートを介してトンネリングを試みるアプリケーションに対する管理制御を行います。これらのアプリケーションが HTTP で実行されている場合は、P2P およびインスタント メッセージングの検査とポリシー チェックを実行できます。AIC は、FTP トラフィックを検査し、発行されるコマンドを制御する方法も提供します。事前定義済みシグニチャをイネーブルまたはディセーブルにするか、カスタム シグニチャを使用してポリシーを作成することができます。

（注）　AIC エンジンは、HTTP トラフィックが AIC Web ポートで受信された場合に実行されます。トラフィックが Web トラフィックであっても AIC Web ポートで受信されない場合は、Service HTTP エンジンが実行されます。AIC 検査は、ポートが AIC Web ポートとして設定されており、検査されるトラフィックが HTTP トラフィックである場合に実行できます。

AIC エンジンとセンサーのパフォーマンス

アプリケーション ポリシーの実施は、固有のセンサー機能です。AIC ポリシーの実施は、不正利用、脆弱性、および異常を検査する従来のテクノロジーに基づいて行うのではなく、HTTP サービス ポリシーと FTP サービス ポリシーを実施するように設計されています。このポリシーの実施に必要な検査作業は、従来の IPS の検査作業とは大きく異なります。この機能を使用すると、パフォーマンスが大きく低下します。AIC をイネーブルにすると、センサー帯域幅の全体容量が減少します。

AIC ポリシーの実施は、IPS のデフォルト設定でディセーブルになっています。AIC ポリシーの実施をアクティブにする場合、必要なポリシーのみを注意深く選択し、不要なポリシーをディセーブルにすることを強く推奨します。また、センサーの検査負荷容量が最大値に近い場合、センサーをオーバーサブスクライブする可能性があるため、この機能を使用しないことを推奨します。このタイプのポリシーの実施を処理するには、適応型セキュリティ アプライアンス ファイアウォールを使用することを推奨します。

AIC エンジンのパラメータ

AIC エンジンでは、Web トラフィックの詳細な検査のためにシグニチャが定義されます。また、FTP コマンドを許可および適用するシグニチャも定義されます。

AIC エンジンには、AIC HTTP と AIC FTP という 2 つの種類があります。

AIC エンジンは、次の機能を搭載しています。

- Web トラフィック：
  - RFC コンプライアンスの適用
  - HTTP 要求メソッドの許可および適用
  - 応答メッセージの検証
  - MIME タイプの適用
  - 転送符号化タイプの検証
  - メッセージのコンテンツと転送されるデータの種類に基づくコンテンツ制御
  - URI Length の適用
  - 設定したポリシーとヘッダーに基づくメッセージ サイズの適用

  - トンネリング、P2P、およびインスタント メッセージングの適用

    この適用は、正規表現を使用して実行されます。事前定義済みのシグニチャがありますが、リストを拡大できます。

- FTP トラフィック：
  - FTP コマンドの許可および適用

表 B-5 に、AIC HTTP エンジンに固有のパラメータを示します。

表 B-5　AIC HTTP エンジンのパラメータ

パラメータ	説明
signature-type	AIC シグニチャのタイプ。
content-types	MIME タイプを処理する AIC シグニチャ。 • define-content-type：特定の MIME タイプ（画像または gif）の拒否、メッセージ サイズ違反の定義、ヘッダーと本文で指定された MIME タイプが一致していないことの判断などのアクションを関連付けます。 • define-recognized-content-types：センサーで認識されるコンテンツ タイプを示します。
define-web-traffic-policy	準拠していない HTTP トラフィックが確認された場合に実行するアクションを指定します。**alarm-on-non-http-traffic [true | false]** コマンドは、シグニチャをイネーブルにします。このシグニチャは、デフォルトではディセーブルになっています。
max-outstanding-requests-overrun	接続ごとに許可される最大 HTTP 要求（1 ～ 16）。
msg-body-pattern	正規表現を使用して、メッセージ本文で特定のパターンを検索するシグニチャを定義します。
request-methods	アクションを HTTP 要求メソッドに関連付けることができる AIC シグニチャ。 • define-request-method：get、put など。 • recognized-request-methods：センサーで認識されるメソッドを示します。
transfer-encodings	転送符号化を処理する AIC シグニチャ。 • define-transfer-encoding：compress、chunked などの各メソッドとアクションを関連付けます。 • recognized-transfer-encodings：センサーで認識されるメソッドを示します。 • chunked-transfer-encoding：チャンク符号化エラーが確認された場合、エラーでは実行するアクションが指定されます。

表 B-6 に、AIC FTP エンジンに固有のパラメータを示します。

表 B-6　AIC FTP エンジンのパラメータ

パラメータ	説明
signature-type	AIC シグニチャのタイプを指定します。
ftp-commands	FTP コマンドとアクションを関連付けます。 • ftp-command：検査する FTP コマンドを選択できます。
unrecognized-ftp-command	認識されない FTP コマンドを検査します。

詳細情報

- AIC エンジン シグニチャを設定する手順については、「AIC シグニチャの設定」（P.8-17）を参照してください。
- カスタム AIC シグニチャの例については、「AIC シグニチャの作成」（P.8-26）を参照してください。
- シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください

Atomic エンジン

Atomic エンジンには、アラートが起動される単純かつ単一のパケット条件のシグニチャが含まれます。ここでは、Atomic エンジンについて説明します。内容は次のとおりです。

Atomic ARP エンジン

Atomic ARP エンジンは、レイヤ 2 の基本的な ARP シグニチャを定義し、ARP スプーフィング ツールである dsniff と ettercap に対して高度な検出を実行します。

表 B-7 に、Atomic ARP エンジンに固有のパラメータを示します。

表 B-7　Atomic ARP エンジンのパラメータ

パラメータ	説明	値
specify-arp-operation	（任意）ARP 動作をイネーブルにします。 • arp-operation：検査する ARP 処理のタイプ。	0 ～ 65535
specify-mac-flip	（任意）MAC アドレスの置換回数をイネーブルにします。 • mac-flip：アラートで MAC アドレスを置換する回数を指定します。	0 ～ 65535

表 B-7　Atomic ARP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-request-inbalance	（任意）要求のアンバランスをイネーブルにします。 • request-inbalance：特定の IP アドレスに対して、応答よりも要求の方が指定した数よりも多い場合に、アラートを起動します。	0 ～ 65535
specify-type-of-arp-sig	（任意）ARP シグニチャのタイプをイネーブルにします。 • type-of-arp-sig：起動する ARP シグニチャのタイプを指定します。 – Destination Broadcast：255.255.255.255 の ARP 宛先アドレスを検出した場合に、このシグニチャのアラームを起動します。 – Same Source and Destination：送信元と宛先の MAC アドレスが同じである ARP 宛先アドレスを検出した場合に、このシグニチャのアラームを起動します。 – Source Broadcast（デフォルト）：255.255.255.255 の ARP 送信元アドレスを検出した場合に、このシグニチャのアラームを起動します。 – Source Multicast：ARP 送信元 MAC アドレス 01:00:5e:（00-7f）を検出した場合に、このシグニチャのアラームを起動します。	dst-broadcast same-src-dst src-broadcast src-multicast
storage-key	固定データを保存するために使用するアドレス キーのタイプ。 • 攻撃者のアドレス • 攻撃者と攻撃対象のアドレス • 攻撃対象のアドレス • グローバル	Axxx AxBx xxBx xxxx

Atomic IP Advanced エンジン

ここでは、Atomic IP Advanced エンジンについて説明します。内容は次のとおりです。

Atomic IP Advanced Engine について

Atomic IP Advanced エンジンは、IPv6 ヘッダーおよびその拡張、IPv4 ヘッダーおよびそのオプション、ICMP、ICMPv6、TCP、および UDP を解析および解釈し、通常とは異なるアクティビティを示す異常を検出します。

Atomic IP Advanced エンジン シグニチャでは、次の処理が実行されます。

- スプーフィングされたアドレスなど、IP アドレスの異常を検査します。
- パケット長フィールドで、不正な情報を検査します。
- パケットに関する情報アラートを起動します。
- 既知の脆弱性の一部に対して、より高い重大度のアラートを起動します。
- IPv6 にも適用できる Engine Atomic IP の IPv6 固有のシグニチャを複製します。
- IP アドレス、ポート プロトコル、およびパケット データからの限られた情報に基づいて、トンネリングされたトラフィックを特定するためのデフォルト シグニチャを提供します。

最も外側の IP トンネルだけが特定されます。IPv6 トンネルまたは IPv4 トンネル内の IPv6 トラフィックが検出された場合、シグニチャによってアラートが起動されます。組み込まれたトンネル内の他の IPv6 トラフィックはすべて検査されません。次のトンネリング方式がサポートされていますが、個々に検出されることはありません。たとえば、ISATAP、6to4、および手動の IPv6 RFC 4213 トンネルはすべて、シグニチャ 1007 によって検出される IPv6 in IPv4 として表示されます。

- ISATAP
- 6to4（RFC 3056）
- 手動で設定されたトンネル（RFC 4213）
- IPv6 over GRE
- Teredo (IPv6) inside UDP
- MPLS（暗号化なし）
- IPv6 over IPv6

IPv6 は次をサポートしています。

- 送信元 IP アドレス、宛先 IP アドレス、または IP アドレス ペアによる拒否
- アラート
- TCP 接続のリセット
- ロギング

Atomic IP Advanced エンジンの制限事項

Atomic IP Advanced エンジンには、次の制限があります。

- パケットがフラグメント化され、レイヤ 4 の ID が最初のパケットに存在しない場合は、パケットのレイヤ 4 フィールドを検出できません。
- フラグ メント再構成が実行されないため、IPv6 によってフラグ メント化されるパケットで、フロー内のレイヤ 4 攻撃を検出できません。
- トンネリングされたフローで攻撃を検出できません。
- フラグメンテーション ヘッダーに対するチェックは限定的です。
- AIM IPS と NME IPS は、インストール先のルータによって IPv6 データが送信されないため、IPv6 機能をサポートしていません。IPv6 検査は IDSM2 で機能する可能性がありますが、正式にサポートされていません。管理（コマンドおよび制御）インターフェイスで IPv6 はサポートされていません。ASA 8.2(1) で、AIP SSM は、IPv6 機能をサポートしています。不正な重複ヘッダーがある場合は、シグニチャが起動されますが、個々のヘッダーを個別に検査することはできません。

- 異常検出では、IPv6 トラフィックはサポートされていません。IPv4 トラフィックだけが、異常検出プロセッサに転送されます。
- レート制限とブロッキングは、IPv6 トラフィックではサポートされていません。シグニチャにブロックまたはレート制限イベント アクションが設定され、IPv6 トラフィックによってトリガーされる場合、アラートは生成されますが、アクションは実行されません。

Atomic IP Advanced エンジンのパラメータ

（注）　範囲の 2 番目の数値は、最初の数値以上にする必要があります。

表 B-8 に、Atomic IP Advanced エンジンに固有のパラメータを示します。

表 B-8　Atomic IP Advanced エンジンのパラメータ

パラメータ	説明	値
グローバル		
fragment-status	フラグメントが必要かどうかを指定します。	any | no-fragments | want-fragments
specify-encapsulation	（任意）パケットに対して L3 の先頭の前に、カプセル化を指定します。[1] • encapsulation：検査するカプセル化のタイプ。	none | mpls | gre | ipv4-in-ipv6 | ipip| any
specify-ip-version	（任意）IP プロトコル バージョンを指定します。 • version：IPv4 または IPv6。	ipv4 | ipv6
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false
Regex		
specify-regex-inspection	（任意）正規表現の検査をイネーブルにします。	yes | no
regex-scope	検索の開始点と終了点を指定します。	• ipv6-doh-only • ipv6-doh-plus • ipv6-hoh-only • ipv6-hoh-plus • ipv6-rh-only • ipv6-rh-plus • layer3-only • layer3-plus • layer4
regex-string	単一の TCP パケット内で検索する正規表現を指定します。	string

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-exact-match-offset	完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535
specify-min-match-length	最小一致長をイネーブルにします。 • min-match-length：regex-string で照合する必要があるバイトの最小数を指定します。	0 ～ 65535
specify-min-match-offset	最小一致オフセットをイネーブルにします。 • min-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある最小ストリーム オフセットを指定します。	0 ～ 65535
specify-max-match-offset	最大一致オフセットをイネーブルにします。 • max-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある最大ストリーム オフセットを指定します。	0 ～ 65535
IPv6		
specify-authentication-header	（任意）認証ヘッダーの検査をイネーブルにします。 • ah-present：認証ヘッダーが存在することを指定します。 – ah-length：認証ヘッダー長を指定します。 – ah-next-header：認証ヘッダーの値を指定します。	 have-ah | no-ah 0 ～ 1028 0 ～ 255
specify-dest-options-header	（任意）宛先オプション ヘッダーの検査をイネーブルにします。 • doh-present：宛先オプション ヘッダーが存在することを指定します。 – doh-count：検査する宛先オプション ヘッダーの数を指定します。 – doh-length：検査する宛先オプション ヘッダー長を指定します。 – doh-next-header：検査する次の宛先オプション ヘッダーの数を指定します。 – doh-option-type：検査する宛先オプション ヘッダーのタイプを指定します。 – doh-option-length：検査する宛先オプション ヘッダー長を指定します。	 have-doh | no-doh 0 ～ 2 8 ～ 2048 0 ～ 255 0 ～ 255 0 ～ 255

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-esp-header	（任意）ESP ヘッダーの検査をイネーブルにします。 • esp-present：ESP ヘッダーが存在することを指定します。	have-esp | no-esp
specify-first-next-header	（任意）最初の next ヘッダーの検査をイネーブルにします。 • first-next-header：検査する最初の next ヘッダーの値を指定します。	0 ～ 255
specify-flow-label	（任意）フロー ラベルの検査をイネーブルにします。 • flow-label：検査するフロー ラベルの値を指定します。	0 ～ 1048575
specify-headers-out-of-order	（任意）順序が正しくないヘッダーの検査をイネーブルにします。 • headers-out-of-order：検査するヘッダーの順序を指定します。	true | false
specify-headers-repeated	（任意）繰り返しヘッダーの検査をイネーブルにします。 • headers-repeated：検査するヘッダーの繰り返しを指定します。	true | false
specify-hop-limit	（任意）ホップ制限をイネーブルにします。 • hop-limit：検査するホップ制限の値を指定します。	0 ～ 255
specify-hop-options-header	（任意）ホップバイホップ オプション ヘッダーの検査をイネーブルにします。 • hoh-present：ホップバイホップ オプション ヘッダーが存在することを指定します。	have-hoh | no-hoh

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-ipv6-addr-options	（任意）IPv6 アドレス オプションをイネーブルにします。 • ipv6-addr-options：IPv6 アドレス オプションを指定します。 – address-with-localhostt：::1 が付く IP アドレス。 – documentation-address：2001:db8::/32 プレフィクスが付く IP アドレス。 – ipv6-addr：IP アドレス。 – link-local-address：IPv6 リンク ローカル アドレスを検査します。 – multicast-dst：宛先マルチキャスト アドレスを検査します。 – multicast-src：送信元マルチキャスト アドレスを検査します。 – not-link-local-address：リンクローカルではないアドレスを検査します。 – not-valid-address：リンクローカル、グローバル、またはマルチキャスト用に予約されていないアドレスを検査します。 – src-ip-eq-dst-ip：送信元アドレスと宛先アドレスが同じです。	
specify-ipv6-data-length	（任意）IPv6 データ長の検査をイネーブルにします。 • ipv6-data-length：検査する IPv6 データ長を指定します。	0 ～ 65535
specify-ipv6-header-length	（任意）IPv6 ヘッダー長の検査をイネーブルにします。 • ipv6-header-length：検査する IPv6 ヘッダー長を指定します。	0 ～ 65535
specify-ipv6-total-length	（任意）IPv6 の全長の検査をイネーブルにします。 • ipv6-total-length：検査する IPv6 の全長を指定します。	0 ～ 65535
specify-ipv6-payload-length	（任意）IPv6 ペイロード長の検査をイネーブルにします。 • ipv6-payload-length：検査する IPv6 ペイロード長を指定します。	0 ～ 65535

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-routing-header	（任意）ルーティング ヘッダーの検査をイネーブルにします。 • rh-present：ルーティング ヘッダーが存在することを指定します。	have-rh | no-rh
specify-traffic-class	（任意）トラフィック クラスの検査をイネーブルにします。 • traffic-class：検査するトラフィック クラスの値を指定します。	0 ～ 255
IPV4		
specify-ip-addr-options	（任意）IP アドレス オプションをイネーブルにします。 • ip-addr-options：IP アドレス オプションを指定します。	address-with-localhost ip-addr rfc-1918-address src-ip-eq-dst-ip
specify-ip-header-length	（任意）IP ヘッダー長の検査をイネーブルにします。 • ip-header-length：検査する IP ヘッダー長を指定します。	0 ～ 16
specify-ip-id	（任意）IP ID の検査をイネーブルにします。 • ip-id：検査する IP ID を指定します。	0 ～ 255
specify-ip-option-inspection	（任意）IP オプションの検査をイネーブルにします。 • ip-option-inspection：IP オプションの値を指定します。 – ip-option：照合する IP オプション コード。 – ip-option-abnormal：オプションのリストの形式が不正です。	0 ～ 65535 true | false
specify-ip-payload-length	（任意）IP ペイロード長の検査をイネーブルにします。 • ip-payload-length：検査する IP ペイロード長を指定します。	0 ～ 65535
specify-ip-tos	（任意）サービスの IP タイプを指定します。 • ip-tos：検査するサービスの IP タイプを指定します。	0 ～ 255
specify-ip-total-length	（任意）IP の全長の検査をイネーブルにします。 • ip-total-length：検査する IP パケットの全長を指定します。	0 ～ 65535

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-ip-ttl	（任意）IP 存続可能時間の検査をイネーブルにします。 • ip-ttl：IP TTL 検査を指定します。	0 ～ 255
specify-ip-version	（任意）IP バージョンの検査をイネーブルにします。 • ip-version：検査する IP バージョンを指定します。	0 ～ 16
L4 Protocol		
specify-l4-protocol	（任意）L4 プロトコルの検査をイネーブルにします。 • l4-protocol：検査する L4 プロトコルを指定します。	icmp icmpv6 tcp udp other
L4 Protocol Other		
other-ip-protocol-id	（任意）その他の L4 プロトコルの検査をイネーブルにします。 • other-ip-protocol-id：アラートを送信する単一の IP プロトコル番号を指定します。	0 ～ 256
L4 Protocol ICMP		
specify-icmp-code	（任意）L4 ICMP コードの検査をイネーブルにします。 • icmp-code：ICMP ヘッダーの CODE 値を指定します。	0 ～ 65535
specify-icmp-id	（任意）L4 ICMP ID の検査をイネーブルにします。 • icmp-id：ICMP ヘッダーの IDENTIFIER 値を指定します。	0 ～ 65535
specify-icmp-seq	（任意）L4 ICMP シーケンスの検査をイネーブルにします。 • icmp-seq：検査する ICMP シーケンスを指定します。	0 ～ 65535
specify-icmp-type	（任意）ICMP ヘッダー タイプの検査をイネーブルにします。 • icmp-type：ICMP ヘッダーの TYPE 値を指定します。	0 ～ 65535
L4 Protocol ICMPv6		
specify-icmpv6-code	（任意）L4 ICMPv6 コードの検査をイネーブルにします。 • icmpv6-code：ICMPv6 ヘッダーの CODE 値を指定します。	0 ～ 255

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-icmpv6-id	（任意）L4 ICMPv6 ID の検査をイネーブルにします。 • icmpv6-id：ICMPv6 ヘッダーの IDENTIFIER 値を指定します。	0 ～ 65535
specify-icmpv6-length	（任意）L4 ICMPv6 長の検査をイネーブルにします。 • icmpv6-length：ICMPv6 ヘッダーの LENGTH 値。	0 ～ 65535
specify-icmpv6-mtu-field	（任意）L4 ICMPv6 MTU フィールドの検査をイネーブルにします。 • icmpv6-mtu-field：ICMPv6 ヘッダーの MTU フィールド値。	4,294,967,295
specify-icmpv6-option-type	（任意）L4 ICMPv6 タイプの検査をイネーブルにします。 • icmpv6-option-type：検査する ICMPv6 オプション タイプを指定します。	0 ～ 255
icmpv6-option-length	（任意）L4 ICMPv6 オプション タイプの検査をイネーブルにします。 • icmpv6-option-length：検査する ICMPv6 オプション タイプを指定します。	0 ～ 255
specify-icmpv6-seq	（任意）L4 ICMPv6 シーケンスの検査をイネーブルにします。 • icmpv6-seq：ICMPv6 ヘッダーの SEQUENCE 値。	0 ～ 65535
specify-icmpv6-type	（任意）L4 ICMPv6 タイプの検査をイネーブルにします。 • icmpv6-type：ICMPv6 ヘッダーの TYPE 値。	0 ～ 255
L4 Protocol TCP and UDP		
specify-dst-port	（任意）使用する宛先ポートをイネーブルにします。 • dst-port：シグニチャの該当宛先ポート。	0 ～ 65535
specify-src-port	（任意）使用する送信元ポートをイネーブルにします。 • src-port：シグニチャの該当送信元ポート。	0 ～ 65535

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-tcp-mask	（任意）使用する TCP マスクをイネーブルにします。 • tcp-mask：TCP フラグの比較で使用するマスク。 – URG ビット – ACK ビット – PSH ビット – RST ビット – SYN ビット – FIN ビット	• urg • ack • psh • rst • syn • fin
specify-tcp-flags	（任意）使用する TCP フラグをイネーブルにします。 • tcp-flags：マスクによってマスクされた場合に照合する TCP フラグ。 – URG ビット – ACK ビット – PSH ビット – RST ビット – SYN ビット – FIN ビット	• urg • ack • psh • rst • syn • fin
specify-tcp-reserved	（任意）使用のために予約済みの TCP をイネーブルにします。 • tcp-reserved：予約済みの TCP。	0 ～ 63
specify-tcp-header-length	（任意）L4 TCP ヘッダー長の検査をイネーブルにします。 • tcp-header-length：検査で使用する TCP ヘッダー長を指定します。	0 ～ 60
specify-tcp-payload-length	（任意）L4 TCP ペイロード長の検査をイネーブルにします。 • tcp-payload-length：TCP ペイロード長を指定します。	0 ～ 65535
specify-tcp-urg-pointer	（任意）L4 TCP URG ポインタの検査をイネーブルにします。 • tcp-urg-pointer：TCP URG フラグの検査を指定します。	0 ～ 65535
specify-tcp-window-size	（任意）L4 TCP ウィンドウ サイズの検査をイネーブルにします。 • tcp-window-size：TCP パケットのウィンドウ サイズを指定します。	0 ～ 65535

表 B-8　Atomic IP Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-udp-valid-length	（任意）L4 UDP 有効長の検査をイネーブルにします。 • udp-valid-length：有効と見なされ、検査する必要がない UDP パケット長を指定します。	0 ～ 65535
specify-udp-length-mismatch	（任意）L4 UDP 長不一致の検査をイネーブルにします。 • udp-length-mismatch：IP データ長が、UDP ヘッダー長よりも短い場合に、アラートを起動します。	true | false

1. パケットが GRE、IPIP、IPv4inIPv6、または MPL である場合、センサーは、L3 カプセル化ヘッダーおよびカプセル化ヘッダーをスキップし、すべての検査は、2 番目の L3 から実行されます。カプセル化列挙子によって、エンジンは、該当する L3 の前にカプセル化ヘッダーがあるかどうかを遡って確認できます。

詳細情報

- カスタム IPv6 シグニチャの例については、「IPv6 シグニチャの例」（P.8-52）を参照してください。
- シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください

Atomic IP エンジン

Atomic IP エンジンでは、IP プロトコル ヘッダーと、関連付けられたレイヤ 4 トランスポート プロトコル（TCP、UDP、ICMP）およびペイロードを検査するシグニチャが定義されます。

（注）　Atomic エンジンでは、複数のパケットにまたがる固定データは保存されません。その代わりに、1 つのパケットの解析を基にしてアラートを送信できます。

表 B-9 に、Atomic IP エンジンに固有のパラメータを示します。

表 B-9　Atomic IP エンジンのパラメータ

パラメータ	説明	値
specify-ip-addr-options	（任意）IP アドレス オプションをイネーブルにします。 • ip-addr-options：IP アドレス オプションを指定します。	address-with-localhost ip-addr rfc-1918-address src-ip-eq-dst-ip
specify-ip-header-length	（任意）IP ヘッダー長の検査をイネーブルにします。 • ip-header-length：検査する IP ヘッダー長を指定します。	0 ～ 16

表 B-9　Atomic IP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-ip-id	（任意）IP ID の検査をイネーブルにします。 • ip-id：検査する IP ID を指定します。	0 ～ 255
specify-ip-option-inspection	（任意）IP オプションの検査をイネーブルにします。 • ip-option-inspection：IP オプションの値を指定します。 – ip-option：照合する IP オプション コード。 – ip-option-abnormal：オプションのリストの形式が不正です。	0 ～ 65535 true | false
specify-ip-payload-length	（任意）IP ペイロード長の検査をイネーブルにします。 • ip-payload-length：検査する IP ペイロード長を指定します。	0 ～ 65535
specify-ip-tos	（任意）サービスの IP タイプを指定します。 • ip-tos：検査するサービスの IP タイプを指定します。	0 ～ 255
specify-ip-total-length	（任意）IP の全長の検査をイネーブルにします。 • ip-total-length：検査する IP パケットの全長を指定します。	0 ～ 65535
specify-ip-ttl	（任意）IP 存続可能時間の検査をイネーブルにします。 • ip-ttl：IP TTL 検査を指定します。	0 ～ 255
specify-ip-version	（任意）IP バージョンの検査をイネーブルにします。 • ip-version：検査する IP バージョンを指定します。	0 ～ 16
specify-l4-protocol	（任意）L4 プロトコルの検査をイネーブルにします。 • l4-protocol：検査する L4 プロトコルを指定します。	icmp tcp udp other-protocol
specify-icmp-code	（任意）L4 ICMP コードの検査をイネーブルにします。 • icmp-code：ICMP ヘッダーの CODE 値を指定します。	0 ～ 65535

表 B-9　Atomic IP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-icmp-id	（任意）L4 ICMP ID の検査をイネーブルにします。 • icmp-id：ICMP ヘッダーの IDENTIFIER 値を指定します。	0 ～ 65535
specify-icmp-seq	（任意）L4 ICMP シーケンスの検査をイネーブルにします。 • icmp-seq：検査する ICMP シーケンスを指定します。	0 ～ 65535
specify-icmp-type	（任意）ICMP ヘッダー タイプの検査をイネーブルにします。 • icmp-type：ICMP ヘッダーの TYPE 値を指定します。	0 ～ 65535
specify-icmp-total-length	（任意）L4 ICMP 合計ヘッダー長の検査をイネーブルにします。 • icmp-total-length：検査する ICMP の全長の値を指定します。	0 ～ 65535
other-ip-protocol-id	（任意）その他の L4 プロトコルの検査をイネーブルにします。 • other-ip-protocol-id：アラートを送信する単一の IP プロトコル番号を指定します。	0 ～ 256
specify-dst-port	（任意）使用する宛先ポートをイネーブルにします。 • dst-port：シグニチャの該当宛先ポート。	0 ～ 65535
specify-src-port	（任意）使用する送信元ポートをイネーブルにします。 • src-port：シグニチャの該当送信元ポート。	0 ～ 65535
specify-tcp-mask	（任意）使用する TCP マスクをイネーブルにします。 • tcp-mask：TCP フラグの比較で使用するマスク。 – URG ビット – ACK ビット – PSH ビット – RST ビット – SYN ビット – FIN ビット	• urg • ack • psh • rst • syn • fin

表 B-9　Atomic IP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-tcp-flags	（任意）使用する TCP フラグをイネーブルにします。 • tcp-flags：マスクによってマスクされた場合に照合する TCP フラグ。 – URG ビット – ACK ビット – PSH ビット – RST ビット – SYN ビット – FIN ビット	• urg • ack • psh • rst • syn • fin
specify-tcp-reserved	（任意）使用のために予約済みの TCP をイネーブルにします。 • tcp-reserved：予約済みの TCP。	0 ～ 63
specify-tcp-header-length	（任意）L4 TCP ヘッダー長の検査をイネーブルにします。 • tcp-header-length：検査で使用する TCP ヘッダー長を指定します。	0 ～ 60
specify-tcp-payload-length	（任意）L4 TCP ペイロード長の検査をイネーブルにします。 • tcp-payload-length：TCP ペイロード長を指定します。	0 ～ 65535
specify-tcp-urg-pointer	（任意）L4 TCP URG ポインタの検査をイネーブルにします。 • tcp-urg-pointer：TCP URG フラグの検査を指定します。	0 ～ 65535
specify-tcp-window-size	（任意）L4 TCP ウィンドウ サイズの検査をイネーブルにします。 • tcp-window-size：TCP パケットのウィンドウ サイズを指定します。	0 ～ 65535
specify-udp-length	（任意）L4 UDP 長の検査をイネーブルにします。 • udp-length-：IP データ長が UDP ヘッダー長よりも短い場合に、アラートを起動します。	0 ～ 65535

表 B-9　Atomic IP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-udp-valid-length	（任意）L4 UDP 有効長の検査をイネーブルにします。 • udp-valid-length：有効と見なされ、検査する必要がない UDP パケット長を指定します。	0 ～ 65535
specify-udp-length-mismatch	（任意）L4 UDP 長不一致の検査をイネーブルにします。 • udp-length-mismatch：IP データ長が、UDP ヘッダー長よりも短い場合に、アラートを起動します。	true | false

Atomic IPv6 エンジン

Atomic IPv6 エンジンは、不正な形式の IPv6 トラフィックによって引き起こされる 2 つの IOS 脆弱性を検出します。これらの脆弱性は、ルータのクラッシュおよびその他のセキュリティ上の問題につながることがあります。IOS の脆弱性の 1 つでは、複数の最初のフラグメントが処理されます。これは、バッファ オーバーフローの原因となります。もう 1 つの脆弱性では、不正な ICMPv6 ネイバー探索オプションが処理されます。これも、バッファ オーバーフローの原因となります。

（注）　IPv6 では、IP アドレス サイズは、32 ビットから 128 ビットに拡大されます。これにより、サポートされるアドレシング階層が増大し、より多くのノードにアドレスの割り当てが可能になり、アドレスの自動設定が簡略化されました。

8 つの Atomic IPv6 シグニチャがあります。Atomic IPv6 では、次のタイプの ネイバー探索プロトコルが検査されます。

- タイプ 133：ルータ送信要求
- タイプ 134：ルータ アドバタイズメント
- タイプ 135：ネイバー送信要求
- タイプ 136：ネイバー アドバタイズメント
- タイプ 137：リダイレクト

（注）　ホストおよびルータはネイバー探索を使用して、添付されたリンクに常駐し、無効になったキャッシュ値を素早くパージすることがわかっているネイバーのリンク層アドレスを判断します。また、ホストはネイバー探索を使用して、ホストに代わってパケットを転送しようとしている隣接ルータを検出します。

各ネイバー探索タイプには、1 つ以上のネイバー探索オプションがある場合があります。Atomic IPv6 エンジンでは、RFC 2461 に指定される適正な値に準拠するために、各オプションの長さが検査されます。オプションの長さに違反すると、不正な長さが検出されたオプション タイプに対応するアラートが生じます（シグニチャ 1601 ～ 1605）。

（注）　Atomic IPv6 シグニチャには、設定する特定のパラメータはありません。

表 B-10 に、Atomic IPv6 シグニチャを示します。

表 B-10　Atomic IPv6 シグニチャ

シグニチャ ID	サブシグニチャ ID	名前	説明
1600	0	ICMPv6 zero length option	長さが 0 と示されているオプション タイプ用
1601	0	ICMPv6 option type 1 violation	8 バイトまたは 16 バイトの有効長の違反。
1602	0	ICMPv6 option type 2 violation	8 バイトまたは 16 バイトの有効長の違反。
1603	0	ICMPv6 option type 3 violation	32 バイトの有効長の違反。
1604	0	ICMPv6 option type 4 violation	80 バイトの有効長の違反。
1605	0	ICMPv6 option type 5 violation	8 バイトの有効長の違反。
1606	0	ICMPv6 short option data	不十分なデータ シグニチャ（パケットが、実際のパケットで使用できるデータよりも、オプション用の多くのデータがあることを示している場合）
1607	0	IPv6 multiple-crafted fragment packets	複数の最初のセグメントが、30 秒間に確認された場合に、アラートを生成します。

Fixed エンジン

Fixed エンジンでは、複数の正規表現パターンが単一のパターン照合テーブルに統合されます。これにより、データ全体を 1 回で検索できます。ICMP、TCP、および UDP のプロトコルをサポートしています。最小の検査深度（1 ～ 100 バイト）に達すると、検査は停止します。Fixed ICMP、Fixed TCP、および Fixed UDP の 3 つの Fixed エンジンがあります。

（注）　Fixed TCP と Fixed UDP では、除外ポートとして service-ports パラメータが使用されます。Fixed ICMP では、除外 ICMP タイプとして service-ports パラメータが使用されます。

表 B-11 に、Fixed ICMP エンジンに固有のパラメータを示します。

表 B-11　Fixed ICMP エンジンのパラメータ

パラメータ	説明	値
direction	トラフィックの方向。 • サービス ポートからクライアント ポート宛のトラフィック • クライアント ポートからサービス ポート宛のトラフィック	from-service to-service
max-payload-inspect-length	シグニチャの最大検査深度を指定します。	1 ～ 250
regex-string	単一のパケット内で検索する正規表現を指定します。	string
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：regex-string で照合する必要があるバイトの最小数を指定します。	0 ～ 65535
specify-icmp-type	（任意）ICMP ヘッダー タイプの検査をイネーブルにします。 • icmp-type：ICMP ヘッダーの TYPE 値を指定します。	0 ～ 65535
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

表 B-12 に、Fixed TCP エンジンに固有のパラメータを示します。

表 B-12　Fixed TCP エンジンのパラメータ

パラメータ	説明	値
direction	トラフィックの方向。 • サービス ポートからクライアント ポート宛のトラフィック • クライアント ポートからサービス ポート宛のトラフィック	from-service to-service
max-payload-inspect-length	シグニチャの最大検査深度を指定します。	1 ～ 250
regex-string	単一のパケット内で検索する正規表現を指定します。	string

表 B-12　Fixed TCP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：regex-string で照合する必要があるバイトの最小数を指定します。	0 ～ 65535
specify-service-ports	使用するサービス ポートをイネーブルにします。 • service-ports：ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

表 B-13 に、Fixed UDP エンジンに固有のパラメータを示します。

表 B-13　Fixed UDP エンジンのパラメータ

パラメータ	説明	値
direction	トラフィックの方向。 • サービス ポートからクライアント ポート宛のトラフィック • クライアント ポートからサービス ポート宛のトラフィック	from-service to-service
max-payload-inspect-length	シグニチャの最大検査深度を指定します。	1 ～ 250
regex-string	単一のパケット内で検索する正規表現を指定します。	string
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535

表 B-13　Fixed UDP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：regex-string で照合する必要があるバイトの最小数を指定します。	0 ～ 65535
specify-service-ports	使用するサービス ポートをイネーブルにします。 • service-ports：ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

1. 範囲の 2 番目の数値は、最初の数値以上にする必要があります。

詳細情報

シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください。

Flood エンジン

Flood エンジンは、複数のパケットを単一のホストまたはネットワークに送信しているホストまたはネットワークを監視するシグニチャを定義します。たとえば、1 秒あたり（特定のタイプの）150 以上のパケットが攻撃対象ホストに送信されていることが判明した場合に、起動されるシグニチャを作成できます。Flood Host と Flood Net の 2 タイプの Flood エンジンがあります。

表 B-14 に、Flood Host エンジンに固有のパラメータを示します。

表 B-14　Flood Host エンジンのパラメータ

パラメータ	説明	値
protocol	検査するトラフィックの種類。	ICMP UDP
rate	1 秒あたりのパケット数のしきい値。	0 ～ 65535[1]
icmp-type	ICMP ヘッダー タイプの値を指定します。	0 ～ 65535
dst-ports	UDP プロトコルを選択した場合の宛先ポートを指定します。	0 ～ 65535[2] a-b[,c-d]
src-ports	UDP プロトコルを選択した場合の送信元ポートを指定します。	0 ～ 65535[3] a-b[,c-d]

1. rate が 1 秒あたりのパケット数よりも大きい場合は、アラートが起動されます。
2. 範囲の 2 番目の数は、最初の数以上である必要があります。
3. 範囲の 2 番目の数は、最初の数以上である必要があります。

表 B-15 に、Flood Net エンジンに固有のパラメータを示します。

表 B-15　Flood Net エンジンのパラメータ

パラメータ	説明	値
gap	フラッド シグニチャで許可される間隔（秒数）。	0 ～ 65535
peaks	フラッド トラフィックで許可されるピーク数。	0 ～ 65535
protocol	検査するトラフィックの種類。	ICMP TCP UDP
rate	1 秒あたりのパケット数のしきい値。	0 ～ 65535[1]
sampling-interval	トラフィックをサンプリングする間隔。	1 ～ 3600
icmp-type	ICMP ヘッダー タイプの値を指定します。	0 ～ 65535

1. rate が 1 秒あたりのパケット数よりも大きい場合は、アラートが起動されます。

Meta エンジン

ここでは、Meta エンジンについて説明します。内容は次のとおりです。

Meta エンジンについて

Meta エンジンでは、スライディング時間間隔内に、関連した方法で発生するイベントを定義します。このエンジンは、パケットではなくイベントを処理します。シグネチャ イベントが生成されると、Meta エンジンはシグネチャ イベントを検査して、1 つ以上の Meta 定義に一致するかどうかを判定します。Meta エンジンは、すべてのイベント要件が満たされるとシグネチャ イベントを生成します。

すべてのシグネチャ イベントは、シグニチャ イベント アクション プロセッサによって Meta エンジンに渡されます。シグニチャ イベント アクション プロセッサは、最小ヒット数オプションを処理してからイベントを渡します。Meta エンジンがコンポーネント イベントを処理してから、サマライズおよびイベント アクションは処理されます。

注意　多数の Meta シグニチャがあると、センサー パフォーマンス全体に影響を及ぼす可能性があります。

詳細情報

シグニチャ イベント アクション プロセッサの詳細については、「シグニチャ イベント アクション プロセッサ」（P.7-2）を参照してください。

コンポーネント シグニチャと Meta エンジン

コンポーネント シグニチャは独立したシグニチャではなく、Meta シグニチャの一部です。**sig-type** オプションは、**component** としてマークされます。これらのシグニチャは独立したシグニチャではないため、トリガーされる場合のリスク レーティングは、自動的に 0 に設定されます。リスク レーティン

グは、コンポーネント シグニチャではなく、Meta シグニチャに適用されます。これにより、コンポーネント シグニチャが、イベント アクション オーバーライドまたはグローバル相関によってパケットを拒否することが防止されます。イベント アクション オーバーライドとグローバル相関は、コンポーネント シグニチャではなく、Meta シグニチャに適用されます。

（注）　Meta シグニチャ内の一部のコンポーネント シグニチャは、独立シグニチャとコンポーネント シグニチャの両方として重要です。これらのシグニチャは、**sig-type component** としてマークされず、**vulnerability**、**exploit**、**anomaly**、**other** のいずれかに設定された **sig-type** としてマークされます。これらのシグニチャのリスク レーティングは計算され、0 には設定されません。

詳細情報

- リスク レーティングの計算の詳細については、「リスク レーティングの計算」（P.7-12）を参照してください。
- イベント アクション オーバーライドおよびその設定方法の詳細については、「イベント アクション オーバーライドの設定」（P.7-16）を参照してください。
- 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。

Meta エンジンのパラメータ

表 B-16 に、Meta エンジンに固有のパラメータを示します。

表 B-16　Meta エンジンのパラメータ

パラメータ	説明	値
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false
meta-reset-interval	Meta シグニチャをリセットする時間（秒単位）。	0 ～ 3600
component-list	Meta コンポーネントのリスト。 • edit：既存のエントリを編集します • insert：リストに新しいエントリを挿入します。 – begin：エントリをアクティブ リストの先頭に配置します – end：エントリをアクティブリストの終わりに配置します – inactive：エントリを非アクティブ リストに配置します – before：エントリを、指定したエントリの前に配置します – after：エントリを、指定したエントリの後ろに配置します • move：リスト内のエントリを移動します。	*name1*

表 B-16　Meta エンジンのパラメータ（続き）

パラメータ	説明	値
meta-key	Meta シグニチャのストレージ タイプ。 • 攻撃者のアドレス • 攻撃者と攻撃対象のアドレス • 攻撃者と攻撃対象のアドレスおよびポート • 攻撃対象のアドレス	AaBb AxBx Axxx xxBx
unique-victim-ports	Meta シグニチャごとに一意の必須攻撃対象ポートの番号。	1 ～ 256
component-list-in-order	コンポーネント リストを順番に起動するかどうか。	true | false

詳細情報

- カスタム Meta エンジン シグニチャの例については、「Meta シグニチャの例」（P.8-48）を参照してください。
- すべてのシグニチャ エンジンに共通のパラメータの詳細については、「Master エンジン」（P.B-3）を参照してください。

Multi String エンジン

Multi String エンジンでは、レイヤ 4 トランスポート プロトコル（ICMP、TCP、および UDP）のペイロードを検査するシグニチャを定義します。この検査は、1 つのシグニチャに対して複数の文字列を照合して行います。シグニチャを起動するために一致する必要がある一連の正規表現パターンを指定できます。たとえば、UDP サービスで regex 1 とそれに続く regex 2 を検索するシグニチャを定義できます。UDP および TCP の場合は、ポート番号と方向を指定できます。単一の送信元ポート、単一の宛先ポート、または両方のポートを指定できます。文字列の照合は両方向で実行されます。

Multi String エンジンは、複数の正規表現パターンを指定する必要がある場合に使用します。それ以外の場合は、String ICMP、String TCP、または String UDP エンジンを使用して、これらのプロトコルのいずれかに対応した単一の正規表現パターンを指定できます。

表 B-17 に、Multi String Multi String エンジンに固有のパラメータを示します。

表 B-17　Multi String エンジンのパラメータ

パラメータ	説明	値
inspect-length	起動するシグニチャに対して違反するすべての文字列を含める必要があるストリームまたはパケットの長さ。	0 ～ 4294967295
protocol	レイヤ 4 プロトコルの選択。	icmp tcp udp
regex-component	正規表現コンポーネントのリスト。 • regex-string：検索する文字列。 • spacing-type：リストの最初のエントリである場合に、ストリームまたはパケットの前または先頭で、照合のために必要なスペースのタイプ。	list（1 ～ 16 項目） exact minimum

表 B-17　Multi String エンジンのパラメータ（続き）

パラメータ	説明	値
port-selection	検査する TCP ポートまたは UDP ポートのタイプ。 • both-ports：送信元ポートと宛先ポートの両方を指定します。 • dest-ports：宛先ポートの範囲を指定します。 • source-ports：送信元ポートの範囲を指定します。[1]	0 ～ 65535 [2]
exact-spacing	この正規表現文字列と直前の正規表現文字列との間、またはストリームやパケットの先頭から（リスト内の最初のエントリである場合）、空ける必要のある正確なバイト数。	0 ～ 4294967296
min-spacing	この正規表現文字列と直前の正規表現文字列との間、またはストリームやパケットの先頭から（リスト内の最初のエントリである場合）、空ける必要のある最小バイト数。	0 ～ 4294967296
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

1. ポート照合は、クライアントからサーバとサーバからクライアントの両方のトラフィック フロー方向で、双方向に実行されます。たとえば、クライアントからサーバへのトラフィック フロー方向で送信元ポート値が 80 である場合、検査はクライアント ポートが 80 である場合に実行されます。サーバからクライアントへのトラフィック フロー方向では、検査はサーバ ポートがポート 80 である場合に実行されます。
2. 有効な値は、0 ～ 65535 の範囲で a-b[,c-d] 形式で指定された整数の、カンマ区切りリストです。範囲の 2 番目の数は、最初の数以上である必要があります。

注意　Multi String エンジンは、メモリの使用状況に大きく影響することがあります。

詳細情報

シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください

Normalizer エンジン

Normalizer エンジンは、IP フラグメンテーションと TCP 正規化を処理します。ここでは、Normalizer エンジンについて説明します。内容は次のとおりです。

Normalizer エンジンについて

（注） Normalizer エンジンには、カスタム シグニチャは追加できません。既存のエンジンを調整することはできます。

Normalizer エンジンは、IP フラグメントの再構成と TCP ストリームの再構成を処理します。Normalizer エンジンでは、センサーが同時に追跡を試みるフラグメントの最大数など、システム リソースの使用に制限を設定できます。無差別モードのセンサーでは、違反に関するアラートがレポートされます。インライン モードのセンサーでは、produce alert、deny packet inline、および modify-packet-inline などのイベント アクション パラメータで指定したアクションが実行されます。

注意 シグニチャ 3050 Half Open SYN Attack に対して、アクションとして modify-packet-inline を選択した場合、保護がアクティブな間に 20 ～ 30% のパフォーマンスの低下が生じることがあります。保護は、実際の SYN フラッド中にだけアクティブになります。

IP フラグメンテーションの正規化

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できるすべての方法をセンサーでチェックする必要がある場合、センサーは DoS 攻撃に対して脆弱になります。フラグメント化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ことによって、これが防止されます。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。

IPv6 フラグメント

Normalizer エンジンでは、IPv6 フラグメントを再構成し、他のエンジンとプロセッサによる検査とアクションのために、再構成したバッファを転送できます。IPv4 と IPv6 には、次の相違点があります。

- Normalizer エンジン シグニチャの modify-packet-inline は、IPv6 データグラムに対して効力がありません。
- シグニチャ 1206（IP Fragment Too Small）は、IPv6 データグラムに対して起動されません。Atomic IP Advanced エンジンのシグニチャ 1741 は、小さすぎる IPv6 フラグメントに対して起動されます。
- IPv6 ヘッダー フィールドは比較的長いため、シグニチャ 1202 では、IPv6 で max-datagram-size を超える 48 の追加バイトが許可されます。

詳細情報

- Normalizer エンジンで IP フラグメント再構成シグニチャを設定する手順については、「IP フラグメント再構成の設定」（P.8-28）を参照してください。
- Normalizer エンジンで TCP ストリーム再構成シグニチャを設定する手順については、「TCP ストリーム再構成の設定」（P.8-32）を参照してください。

Normalizer エンジンのパラメータ

表 B-18 に、Normalizer エンジンに固有のパラメータを示します。

表 B-18　Normalizer エンジンのパラメータ

パラメータ	説明
edit-default-sigs-only	編集可能なシグニチャ。
specify-fragment-reassembly-timeout	（任意）フラグメント再構築タイムアウトをイネーブルにします。
specify-hijack-max-old-ack	（任意）hijack-max-old-ack をイネーブルにします。
specify-max-dgram-size	（任意）最大データグラム サイズをイネーブルにします。
specify-max-fragments	（任意）最大フラグメントをイネーブルにします。
specify-max-fragments-per-dgram	（任意）データグラムあたりの最大フラグメントをイネーブルにします。
specify-max-last-fragments	（任意）直前の最大フラグメントをイネーブルにします。
specify-max-partial-dgrams	（任意）最大部分データグラムをイネーブルにします。
specify-max-small-fragss	（任意）最大スモール フラグメントをイネーブルにします。
specify-min-fragment-size	（任意）最小フラグメント サイズをイネーブルにします。
specify-service-ports	（任意）サービス ポートをイネーブルにします。
specify-syn-flood-max-embryonic	（任意）SYN フラッドの最大初期接続をイネーブルにします。
specify-tcp-closed-timeout	（任意）TCP クローズド タイムアウトをイネーブルにします。
specify-tcp-embryonic-timeout	（任意）TCP 初期接続タイムアウトをイネーブルにします。
specify-tcp-idle-timeout	（任意）TCP アイドル タイムアウトをイネーブルにします。
specify-tcp-max-mss	（任意）TCP 最大 mss（最大セグメント サイズ）をイネーブルにします。
specify-tcp-max-queue	（任意）TCP 最大キューをイネーブルにします。
specify-tcp-min-mss	（任意）TCP 最小 mss をイネーブルにします。
specify-tcp-option-number	（任意）TCP オプション番号をイネーブルにします。

Service エンジン

ここでは、Service エンジンについて説明します。内容は次のとおりです。

Service エンジンについて

Service エンジンでは、2 つのホスト間でレイヤ 5+ トラフィックが分析されます。これらは、固定データを追跡する 1 対 1 のシグニチャです。このエンジンは、ライブ サービスと類似した方法で、レイヤ 5+ ペイロードを分析します。

Service エンジンには共通の特性がありますが、各エンジンには検査するサービスに関して固有な情報があります。文字列エンジンの使用が不適切または望ましくない場合には、Service エンジンによって、アルゴリズムに特化した汎用文字列エンジンの機能が補完されます。

Service DNS エンジン

Service DNS エンジンは、高度な DNS デコードを行います。これには、反回避技術（複数のジャンプの追跡など）が含まれます。長さ、命令コード、文字列などの多数のパラメータがあります。Service DNS エンジンは、2 つのプロトコルに対応するインスペクタであり、TCP ポート 53 と UDP ポート 53 の両方で稼動します。TCP の場合はストリームを使用し、UDP の場合はクワッドを使用します。

表 B-19 に、Service DNS エンジンに固有のパラメータを示します。

表 B-19　Service DNS エンジンのパラメータ

パラメータ	説明	値
protocol	このインスペクタの該当プロトコル。	tcp udp
specify-query-chaos-string	（任意）DNS Query Class Chaos String をイネーブルにします。	*query-chaos-string*
specify-query-class	（任意）クエリー クラスをイネーブルにします。 • query-class：DNS クエリー クラスの 2 バイト値	0 ～ 65535
specify-query-invalid-domain-name	（任意）無効なドメイン名のクエリーをイネーブルにします。 • query-invalid-domain-name：255 よりも大きい DNS クエリー長	true | false
specify-query-jump-count-exceeded	（任意）超過したクエリー ジャンプ数をイネーブルにします。 • query-jump-count-exceeded：DNS 圧縮カウンタ	true | false
specify-query-opcode	（任意）クエリー命令コードをイネーブルにします。 • query-opcode：DNS クエリー命令コードの 1 バイト値	0 ～ 65535
specify-query-record-data-invalid	（任意）無効なレコード データのクエリーをイネーブルにします。 • query-record-data-invalid：不完全な DNS レコード データ	true | false
specify-query-record-data-len	（任意）クエリー レコード データ長をイネーブルにします。 • query-record-data-len：DNS 応答レコード データ長	0 ～ 65535
specify-query-src-port-53	（任意）クエリー送信元ポート 53 をイネーブルにします。 • query-src-port-53：DNS パケット送信元ポート 53	true | false
specify-query-stream-len	（任意）クエリー ストリーム長をイネーブルにします。 • query-stream-len：DNS パケット長	0 ～ 65535

表 B-19　Service DNS エンジンのパラメータ（続き）

パラメータ	説明	値
specify-query-type	（任意）クエリー タイプをイネーブルにします。 • query-type：DNS クエリー タイプの 2 バイト値	0 ～ 65535
specify-query-value	（任意）クエリー値をイネーブルにします。 • query-value：クエリー 0、応答 1	true | false

Service FTP エンジン

Service FTP エンジンは、TEP ポート コマンドのデコード、無効な **port** コマンドおよび PASV ポート スプーフィングのトラップを行います。String エンジンが検出のために適切ではない場合に、補完的な役割を果たします。パラメータは Boolean で、**port** コマンド デコードでさまざまなエラー トラップ条件にマッピングされます。Service FTP エンジンは、TCP ポート 20 および 21 で稼動します。ポート 20 はデータ用で、Service FTP エンジンはこのポートで検査を実行しません。Service FTP エンジンは、ポート 21 の制御トランザクションを検査します。

表 B-20 に、Service FTP エンジンに固有のパラメータを示します。

表 B-20　Service FTP エンジンのパラメータ

パラメータ	説明	値
direction	トラフィックの方向。 • サービス ポートからクライアント ポート宛のトラフィック • クライアント ポートからサービス ポート宛のトラフィック	from-service to-service
ftp-inspection-type	実行する検査のタイプ： • FTP ポート コマンド内の無効なアドレスを検索します。 • FTP ポート コマンド内の無効なポートを検索します。 • PASV ポート スプーフィングを検索します。	bad-port-cmd-address bad-port-cmd-port pasvI
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

Service Generic エンジン

Service Generic エンジンを使用すると、設定ファイルでシグニチャを更新するだけで、プログラム シグニチャを発行できます。このエンジンには、設定ファイルで定義されている簡易マシンおよびアセンブリ言語が含まれています。このエンジンは、仮想マシンを介して（アセンブリ言語から導出された）マシン コードを実行します。仮想マシンは、命令を処理し、パケットから重要な情報を引き出して、マシン コードに指定されている比較および演算を実行します。Service Generic エンジンは、String エンジンと State エンジンを補足する迅速なシグニチャ応答エンジンとして設計されています。

新機能により、Service Generic エンジンに対する正規表現パラメータと拡張命令が追加されました。Service Generic エンジンでは、パケットを解析するために記述されるミニプログラムに基づいて、トラフィックを分析できます。これらのミニプログラムは、パケットを詳細に分析し、特定の条件を探すコマンドから構成されます。

（注）　Service Generic エンジンを使用してカスタム シグニチャを作成することはできません。

注意　複雑な言語特有の性質上、重大度とイベント アクションを除き、Service Generic エンジンのシグニチャ パラメータを編集することは推奨しません。

表 B-21 に、Service Generic エンジンに固有のパラメータを示します。

表 B-21　Service Generic エンジンのパラメータ

パラメータ	説明	値
specify-dst-port	（任意）宛先ポートをイネーブルにします。 • dst-port：シグニチャの該当宛先ポート	0 ～ 65535
specify-ip-protocol	（任意）IP プロトコルをイネーブルにします。 • ip-protocol：インスペクタが検査する IP プロトコル	0 ～ 255
specify-payload-source	（任意）ペイロード送信元検査をイネーブルにします。 • payload-source：次のタイプに対するペイロード送信元検査。 – ICMP データの検査 – レイヤ 2 ヘッダーの検査 – レイヤ 3 ヘッダーの検査 – レイヤ 4 ヘッダーの検査 – TCP データの検査 – UDP データの検査	icmp-data l2-header l3-header l4-header tcp-data udp-dataI
specify-src-port	（任意）送信元ポートをイネーブルにします。 • src-port：シグニチャの該当送信元ポート	0 ～ 65535

表 B-21　Service Generic エンジンのパラメータ（続き）

パラメータ	説明	値
specify-regex-string	ポリシー タイプが regex の場合に検索する正規表現。 • 単一の TCP パケット内で検索する正規表現。 • （任意）使用する最小一致長をイネーブルにします。一致と見なされるために必要な正規表現の最小一致長です。	regex-string specify-min-match-length
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

Service H225 エンジン

Service H225 エンジンでは、多数のサブプロトコルから構成され、H.323 スイートの一部である H225.0 プロトコルが分析されます。H.323 とは、パケットベース ネットワークを通じた会議を実現するプロトコルおよびその他の規格をまとめたものです。

H.225.0 のコール シグナリングとステータス メッセージは、H.323 コール セットアップの一部です。ゲートキーパーおよびエンドポイント ターミナルなど、ネットワーク内のさまざまな H.323 エンティティが、H.225.0 プロトコル スタックの実装を実行します。Service H225 エンジンでは、複数の H.323 ゲートキーパー、VoIP ゲートウェイ、およびエンドポイント ターミナルへの攻撃に対して、H225.0 プロトコルが分析されます。TCP PDU を通じて交換されるコール シグナリング メッセージの、ディープ パケット インスペクションが提供されます Service H225 エンジンでは、H.225.0 プロトコルでの無効な H.255.0 メッセージ、これらのメッセージ内のさまざまなプロトコル フィールドの悪用およびオーバーフロー攻撃が分析されます。

H.225.0 コール シグナリング メッセージは、Q.931 プロトコルに基づいています。発信側のエンドポイントによって、コールするエンドポイントに Q.931 セットアップ メッセージが送信されます。そのアドレスは、アドミッション手順またはルックアップ手段によって取得されます。着信側のエンドポイントは、Q.931 接続メッセージを送信して接続を受け入れるか、接続を拒否します。H.225.0 接続が確立されたときに、発信側または着信側のエンドポイントによって、H.245 アドレスが提供されます。これは、制御プロトコル（H.245）チャネルを確立するために使用されます。

特に重要なのは、コール セットアップの一環として H.323 エンティティ間で交換される最初のメッセージである SETUP コール シグナリング メッセージです。SETUP メッセージでは、コール シグナリング メッセージで一般に見られる多くのフィールドが使用されます。また、潜在的な攻撃にさらされている実装は、SETUP メッセージに対するセキュリティ チェックにほとんど合格しません。したがって、H.225.0 SETUP メッセージの妥当性をチェックし、ネットワークの境界でチェックを実行することが非常に重要です。

Service H225 エンジンには、H225 SETUP メッセージに対する TPKT 検証、Q.931 プロトコル検証、および ASN.1PER 検証のための組み込みシグニチャがあります。ASN.1 とは、データ構造を記述するための表記法です。PER では、異なるスタイルの符号化が使用されます。これは、はるかにコンパクトな表現を生成するために、データ タイプに基づく符号化に使用されます。

Q.931 と TPKT の長さに関するシグニチャを調整し、細分化されたシグニチャを特定の H.225 プロトコル フィールドに適用し、Q.931 または H.225 プロトコルの単一のフィールドに対する複数のパターン検索シグニチャを適用できます。

Service H225 エンジンでは、次の機能がサポートされています。

- TPKT 検証および長さのチェック
- Q.931 情報要素の検証
- Q.931 情報要素のテキスト フィールドに関する正規表現シグニチャ

- Q.931 情報要素に関する長さのチェック
- SETUP メッセージの検証
- ASN.1 PER エンコード エラーのチェック
- 正規表現と長さの両方に対して、ULR-ID、E-mail-ID、h323-id などのフィールドに対する設定シグニチャ。

一定数の TPKT シグニチャと ASN.1 シグニチャがあります。これらのタイプに対してカスタム シグニチャを作成することはできません。TPKT シグニチャでは、長さに関するシグニチャに対して値範囲だけを変更する必要があります。ASN.1 ではすべてのパラメータを変更しないでください Q.931 シグニチャでは、テキスト フィールドに対する新しい正規表現シグニチャを追加できます。SETUP シグニチャでは、さまざまな SETUP メッセージ フィールドに関する長さおよび正規表現のチェックのためにシグニチャを追加できます。

表 B-22 に、Service H225 エンジンに固有のパラメータを示します。

表 B-22　Service H.225 エンジンのパラメータ

パラメータ	説明	値
message-type	シグニチャを適用する H225 メッセージのタイプ。 • SETUP • ASN.1-PER • Q.931 • TPKT	asn.1-per q.931 セットアップ tpkt
policy-type	シグニチャを適用する H225 ポリシーのタイプ： • フィールド長を検査する。 • 存在を検査する。特定のフィールドがメッセージ内に存在する場合は、アラートが送信されます。 • 正規表現を検査する。 • フィールドの妥当性を検査する。 • 値を検査する。 TPKT シグニチャの場合、regex と presence は有効な値ではありません。	length presence regex validate value
specify-field-name	（任意）使用するフィールド名をイネーブルにします。SETUP と Q.931 のメッセージ タイプだけに対して有効です。シグニチャを適用するフィールド名のドット付き表記を指定します。 • field-name：検査するフィールドの名前。	1 ～ 512

表 B-22　Service H.225 エンジンのパラメータ（続き）

パラメータ	説明	値
specify-invalid-packet-index	（任意）ASN と TPKT 固有のエラー、および固定マッピングを持つその他のエラーで使用する無効なパケット インデックスをイネーブルにします。 • invalid-packet-index : 無効なパケット インデックスを検査します。	0 ～ 255
specify-regex-string	ポリシー タイプが regex の場合に検索する正規表現。TPKT シグニチャには設定しないでください。 • 単一の TCP パケット内で検索する正規表現。 • （任意）使用する最小一致長をイネーブルにします。一致と見なされるために必要な正規表現の最小一致長です。TPKT シグニチャには設定しないでください。	regex-string specify-min-match-length
specify-value-range	長さまたは値のポリシー タイプに対して有効です（0x00 ～ 6535）。その他のポリシー タイプの場合は無効です。 • value-range : 値の範囲。	0 ～ 65535[1] a-b
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

1. 範囲の 2 番目の数値は、最初の数値以上にする必要があります。

詳細情報

シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください。

Service HTTP エンジン

Service HTTP エンジンは、サービス固有の文字列ベースのパターン マッチング インスペクション エンジンです。HTTP プロトコルは、現代のネットワークで最もよく使用されているプロトコルです。さらに、これには最も長い前処理時間が必要であり、検査を必要とする最大数のシグニチャを持つため、システムのパフォーマンス全体に対して重大な影響を及ぼします。

Service HTTP エンジンでは、複数のパターンを 1 つのパターン マッチング テーブルにまとめることでデータ内の検索を一度に実行できる新しい正規表現ライブラリが使用されます。このエンジンは Web サービスに送られるトラフィックの、Web サービスだけに対するもの、または HTTP 要求を検索します。このエンジンを使用してリターン トラフィックを検査することはできません。このエンジンの各シグニチャで、該当する個別の Web ポートを指定できます。

HTTP 解読とは、符号化された文字を ASCII 対応文字に正規化することによって、HTTP メッセージをデコードするプロセスです。このプロセスは、ASCII 正規化と呼ばれることもあります。

HTTP パケットを検査するには、あらかじめそのデータを、ターゲット システムでのデータ処理時に表示されるものと同じデータ表現として解読または正規化しておく必要があります。また、ホスト ターゲット タイプごとにカスタマイズされたデコード方式を用意することが推奨されます。そのためには、ターゲット上で動作しているオペレーティング システムおよび Web サーバのバージョンを確認する必要があります。Service HTTP エンジンは、Microsoft IIS Web サーバ用としてデフォルトの解読処理機能を備えています。

表 B-23 に、Service HTTP エンジンに固有のパラメータを示します。

表 B-23　Service HTTP エンジンのパラメータ

パラメータ	説明	値
de-obfuscate	検索の前に反回避解読を適用します。	true | false
max-field-sizes	最大フィールド サイズ グループ。	—
specify-max-arg-field-length	（任意）引数フィールドの最大長をイネーブルにします。 • max-arg-field-length：引数フィールドの最大長。	0 ～ 65535
specify-max-header-field-length	（任意）ヘッダー フィールドの最大長をイネーブルにします。 • max-header-field-length：ヘッダー フィールドの最大長。	0 ～ 65535
specify-max-request-length	（任意）要求フィールドの最大長をイネーブルにします。 • max-request-length：要求フィールドの最大長。	0 ～ 65535
specify-max-uri-field-length	（任意）URI フィールドの最大長をイネーブルにします。 • max-uri-field-length：URI フィールドの最大長。	0 ～ 65535
regex	正規表現グループ。	—
specify-arg-name-regex	（任意）特定の正規表現の引数フィールドの検索をイネーブルにします。 • arg-name-regex：HTTP 引数フィールド（コンテンツ長によって定義されているように、? の後ろのエンティティ本体の中）で検索する正規表現。	—
specify-header-regex	（任意）特定の正規表現のヘッダー フィールドの検索をイネーブルにします。 • header-regex：HTTP ヘッダー フィールドで検索する正規表現。ヘッダーは、最初の CRLF の後ろから定義され、CRLFCRLF まで続きます。	—

表 B-23　Service HTTP エンジンのパラメータ（続き）

パラメータ	説明	値
specify-request-regex	（任意）特定の正規表現の要求フィールドの検索をイネーブルにします。 • request-regex：HTTP URI フィールドと HTTP 引数フィールドの両方で検索する正規表現。 • specify-min-request-match-length：最低要求一致長の設定をイネーブルにします。	0 ～ 65535
specify-uri-regex	（任意）HTTP URI フィールドで検索する正規表現。URI フィールドは、HTTP メソッド（たとえば、GET）の後ろで、最初の CRLF の前まで定義されます。正規表現は保護されています。つまり、値は変更できません。	[/¥¥][a-zA-Z][a-zA-Z][a-zA-Z][a-zA-Z][a-zA-Z][a-zA-Z][a-zA-Z][.]jpeg
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

詳細情報

- Service HTTP カスタム シグニチャの例については、「Service HTTP シグニチャの例」（P.8-45）を参照してください。
- シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください

Service IDENT エンジン

Service IDENT エンジンでは、TCP ポート 113 のトラフィックが検査されます。基本的なデコードを備え、長さのオーバーフローを指定するためのパラメータが提供されます。たとえば、コンピュータ A のユーザまたはプログラムが、コンピュータ B に対して ID 要求を行う場合、A と B 間の接続のユーザ ID だけが要求されることがあります。B の ID サーバは、TCP ポート 113 で接続をリッスンします。A のクライアントは、接続を確立してから、その接続で使用する A および B のポート番号を送信することによって、ID が必要な接続を指定します。B のサーバは、その接続を使用しているユーザを判断し、そのユーザの名前を示す文字列で A に応答します。Service IDENT エンジンでは、TCP ポート 113 のトラフィックで、ID の悪用が検査されます。

表 B-24 に、Service IDENT エンジンに固有のパラメータを示します。

表 B-24　Service IDENT エンジンのパラメータ

パラメータ	説明	値
inspection-type	実行する検査のタイプ： • has-newline：非終端改行文字のペイロードを検査します。 • has-bad-port：不良ポートのペイロードを検査します。 • size：これよりも長いペイロード長を検査します。	—
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
direction	トラフィックの方向： • サービス ポートからクライアント ポート宛のトラフィック。 • クライアント ポートからサービス ポート宛のトラフィック。	from-service to-service

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

Service MSRPC エンジン

Service MSRPC エンジンでは、MSRPC パケットが処理されます。MSRPC によって、ネットワーク環境で、複数のコンピュータおよびそれぞれのアプリケーション ソフトウェア間の連携処理が実現します。これは、トランザクション ベースのプロトコルです。このことは、チャネルを確立し、処理要求および応答を渡す一連の通信が行われることを示しています。

MSRPC は、ISO レイヤ 5 ～ 6 プロトコルで、UDP、TCP、および SMB などの他のトランスポート プロトコル上のレイヤにあります。MSRPC エンジンは、MSRPC PDU のフラグメンテーションと再構築を実現する機能を備えています。この通信チャネルは、Windows NT、Windows 2000、および Window XP の最近のセキュリティ脆弱性の発生源となっています。Service MSRPC エンジンでは、非常に一般的なトランザクション タイプに対して、DCE プロトコルと RPC プロトコルだけがデコードされます。

表 B-25 に、Service MSRPC エンジンに固有のパラメータを示します。

表 B-25　Service MSRPC エンジンのパラメータ

パラメータ	説明	値
protocol	このインスペクタの該当プロトコル。 • type：UDP または TCP	tcp udp
specify-flags	設定するフラグ。 • msrpc-flags • msrpc-tcp-flags-mask	concurrent-execution did-not-execute first-fragment last-fragment maybe-semantics object-uuid reserved

表 B-25　Service MSRPC エンジンのパラメータ（続き）

パラメータ	説明	値
specify-operation	（任意）MSRPC 動作の使用をイネーブルにします。 • operation：要求する MSRPC 動作。SMB_COM_TRANSACTION コマンドに必要です。完全一致。	0 ～ 65535
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false
specify-regex-string	（任意）正規表現文字列の使用をイネーブルにします。 • specify-exact-match-offset：完全一致オフセットをイネーブルにします。 – exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。 • specify-min-match-length：最小一致長をイネーブルにします。 – min-match-length：正規表現文字列が一致する必要があるバイトの最小数。	0 ～ 65535
specify-uuid	（任意）UUID をイネーブルにします。 • uuid：MSRPC UUID フィールド	000001a000000000c0000000000000046

詳細情報

シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください

Service MSSQL エンジン

Service MSSQL エンジンは、Microsoft SQL サーバによって使用されるプロトコルを検査します。このエンジンには 1 つの MSSQL シグニチャが含まれています。デフォルトの sa アカウントを使用した MSSQL サーバへのログイン試行を検出したときにアラートを起動します。ログイン ユーザ名や、パスワードが使用されたかどうかなど、MSSQL プロトコル値に基づいてカスタム シグニチャを追加できます。

表 B-26 に、Service MSSQL エンジンに固有のパラメータを示します。

表 B-26　Service MSSQL エンジンのパラメータ

パラメータ	説明	値
password-present	MS SQL ログインでパスワードが使用されたかどうか。	true | false
specify-sql-username	（任意）SQL ユーザ名の使用をイネーブルにします。 • sql-username：MS SQL サービスにログインしているユーザのユーザ名（完全一致）。	sa

Service NTP エンジン

Service NTP エンジンは、NTP プロトコルを検査します。このエンジンには、1 つの NTP シグニチャ（NTP readvar オーバーフロー シグニチャ）が含まれます。このシグニチャは、サイズが大きいため NTP サービスでキャプチャできない NTP データが、readvar コマンドに指定されていることを検出した場合に、アラートを起動します。NTP プロトコルの値（モードや制御パケットのサイズなど）に基づいて、シグニチャを調整したり、カスタム シグニチャを作成したりできます。

表 B-27 に、Service NTP エンジンに固有のパラメータを示します。

表 B-27　Service NTP エンジンのパラメータ

パラメータ	説明	値
inspection-type	実行する検査のタイプ。	
inspect-ntp-packets	NTP パケットを検査します。 • control-opcode：RFC1305 の付録 B に基づく NTP 制御パケットの命令コード番号。 • max-control-data-size：制御パケットで送信されるデータの最大許容量。 • mode：RFC 1305 に基づく NTP パケットの動作モード。	0 ～ 65535
is-invalid-data-packet	無効な NTP データ パケットを検索します。NTP データ パケットの構造を調べ、サイズが正しいことを確認します。	true | false
is-non-ntp-traffic	NTP ポートの非 NTP パケットをチェックします。	true | false

Service P2P エンジン

P2P ネットワークでは、クライアントとサーバの両方として同時に機能できるノードが、ファイル共有に使用されます。P2P ネットワークには、著作権があるマテリアルが含まれることがあり、企業ネットワークでのその使用は企業のポリシーに違反することがあります。Service P2P エンジンでは、このようなネットワークがモニタされ、最適化された TCP および UDP P2P プロトコル識別が提供されます。Service P2P エンジンには、次の特性があります。

- すべての TCP ポートと UDP ポートをリッスンします。
- 正規表現ではなく、ハードコードされたシグニチャを使用することによって、パフォーマンスが向上します。
- P2P プロトコルが識別された後、または P2P プロトコルが識別されずに 10 のパケットが確認された後、トラフィックは無視されます。

P2P シグニチャはハードコードされているため、編集できるパラメータは、Master エンジン パラメータだけです。

詳細情報

Master エンジン パラメータのリストについては、「Master エンジン」（P.B-3）を参照してください。

Service RPC エンジン

Service RPC エンジンは RPC プロトコルに対して使用され、反回避の方式としてすべてのデコードを備えています。これにより、フラグメント化されたメッセージ（複数パケット内の 1 つのメッセージ）またはバッチ メッセージ（1 つのパケット内の複数メッセージ）を処理できます。

RPC ポート マッパーは、ポート 111 上で動作します。通常の RPC メッセージは、550 より上位であれば任意のポートで送受信できます。RPC スイープは、TCP ポート スイープとほぼ同じものです。異なるのは、有効な RPC メッセージが送信された場合に一意のポートだけをカウントするという点です。RPC は、UDP でも動作します。

表 B-28 に、Service RPC エンジンに固有のパラメータを示します。

表 B-28　Service RPC エンジンのパラメータ

パラメータ	説明	値
direction	トラフィックの方向。 • サービス ポートからクライアント ポート宛のトラフィック。 • クライアント ポートからサービス ポート宛のトラフィック。	from-service to-service
protocol	該当プロトコル。	tcp udp
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
specify-regex-string	（任意）正規表現文字列の使用をイネーブルにします。 • specify-exact-match-offset：完全一致オフセットをイネーブルにします。 – exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。 • specify-min-match-length：最小一致長をイネーブルにします。 – min-match-length：正規表現文字列が一致する必要があるバイトの最小数。	0 ～ 65535
specify-is-spoof-src	（任意）スプーフィングの送信元アドレスをイネーブルにします。 • is-spoof-src：送信元アドレスが 127.0.0.1 の場合にアラートを起動します。	true | false
specify-port-map-program	（任意）ポートマッパー プログラムをイネーブルにします。 • port-map-program：シグニチャのポートマッパーに送信されたプログラム番号。	0 ～ 9999999999
specify-rpc-max-length	（任意）RPC 最大長をイネーブルにします。 • rpc-max-length：RPC メッセージ全体の最大許容長。長さが指定した値より長いとアラートを起動します。	0 ～ 65535

表 B-28　Service RPC エンジンのパラメータ（続き）

パラメータ	説明	値
specify-rpc-procedure	（任意）RPC プロシージャをイネーブルにします。 • rpc-procedure：シグニチャの RPC プロシージャ番号。	0 ～ 1000000
specify-rpc-program	（任意）RPC プログラムをイネーブルにします。 • rpc-program：シグニチャの RPC プログラム番号。	0 ～ 1000000
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true| false

1. 範囲の 2 番目の数値は、最初の数値以上にする必要があります。

詳細情報

シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください。

Service SMB Advanced エンジン

注意 SMB エンジンは、SMB Advanced エンジンによって置き換えられました。SMB エンジンは IDM、IME、および CLI でまだ表示されますが、そのシグニチャは廃止されました。新しいシグニチャには、対応する古いシグニチャ ID の廃止パラメータ セットがあります。新しい SMB Advanced エンジンを使用して、SMB エンジン内にあるカスタム シグニチャを書き換えてください。

Service SMB Advanced エンジンでは、Microsoft SMB パケットと Microsoft RPC over SMB パケットが処理されます。Service SMB Advanced エンジンでは、MSRPC エンジンと同じコネクション型 MSRPC のデコード方式が使用されますが、MSRPC パケットは SMB プロトコルでなければならないという要件があります。Service SMB Advanced エンジンは、TCP ポート 139 および 445 で MSRPC over SMB をサポートしています。これは、MSRPC エンジンからのコネクション型 DCS/RPC コードのコピーを使用します。

表 B-29 に、Service SMB Advanced エンジンに固有のパラメータを示します。

表 B-29　Service SMB Advanced エンジンのパラメータ

パラメータ	説明	値
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535 a-b[,c-d] [1]
specify-command	（任意）SMB コマンドをイネーブルにします。 • command：SMB コマンドの値。完全に一致する必要があります。SMB パケットのタイプを定義します。[2]	0 ～ 255

表 B-29　Service SMB Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-direction	（任意）トラフィック方向をイネーブルにします。 • direction：トラフィックの方向を指定できます。 – from-service：サービス ポートからクライアント ポート宛のトラフィック。 – to-service：クライアント ポートからサービス ポート宛のトラフィック。	from service to service
specify-operation	（任意）MSRPC over SMB をイネーブルにします。 • msrpc-over-smb-operation：SMB_COM_TRANSACTION コマンドに使用します。完全に一致する必要があります。	0 ～ 65535
specify-regex-string	（任意）正規表現文字列の検索をイネーブルにします。 • regex-string：単一の TCP パケット内で検索する正規表現。	
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：正規表現文字列が一致する必要がある最小バイト数。	
specify-payload-source	（任意）ペイロード送信元をイネーブルにします。 • payload-source：ペイロード送信元の検査。[3]	
specify-scan-interval	（任意）スキャン間隔をイネーブルにします。 • scan-interval：アラート率の計算に使用される間隔（秒数）。	1 ～ 131071

表 B-29　Service SMB Advanced エンジンのパラメータ（続き）

パラメータ	説明	値
specify-tcp-flags	（任意）TCP フラグをイネーブルにします。 • msrpc-tcp-flags • msrpc-tcp-flags-mask	• concurrent execution • did not execute • first fragment • last fragment • maybe • object UUID • pending cancel • reserved
specify-type	（任意）MSRPC over SMB パケットのタイプをイネーブルにします。 • type：MSRPC over SMB パケットの Type フィールド。	• 0 = 要求 • 2 = 応答 • 11 = バインド • 12 = バインド応答
specify-uuid	（任意）UUID を経由した MSRPC をイネーブルにします。 • uuid：MSRPC UUID フィールド	16 進数の 0 ～ 9、a ～ f、A ～ F で構成される 32 文字の文字列。
specify-hit-count	（任意）ヒット カウントをイネーブルにします。 • hit-count：scan-interval 内の発生回数のしきい値。この値を超えるとアラートが起動されます。	1 ～ 65535
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true | false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。
2. 現在、37（0x25）SMB_COM_TRANSACTION コマンドおよび 162（0xA2）SMB_COM_NT_CREATE_ANDX コマンドがサポートされています。
3. TCP_Data はパケット全体に対して正規表現を実行し、SMB_Data は SMB ペイロードだけに関して正規表現を実行し、Resource_DATA は SMB_Resource で正規表現を実行します。

詳細情報

シグニチャ正規表現の構文のリストについては、「正規表現の構文」（P.B-9）を参照してください。

Service SNMP エンジン

Service SNMP エンジンは、ポート 161 宛のすべての SNMP パケットを検査します。特定のコミュニティ名とオブジェクト ID に基づいて、SNMP シグニチャを調整したり、カスタム SNMP シグニチャを作成したりできます。コミュニティ名とオブジェクト ID を照合するために、文字列比較や正規表現演算を使用する代わりに、整数を使用してすべての比較を実行し、プロトコル デコードを高速化しストレージ要件を削減します。

表 B-30 に、Service SNMP エンジンに固有のパラメータを示します。

表 B-30　Service SNMP エンジンのパラメータ

パラメータ	説明	値
inspection-typeI	実行する検査のタイプ。	—
brute-force-inspection	総当たり攻撃の試行を検査します。 • brute-force-count：総当たり攻撃と見なされる一意の SNMP コミュニティ名の数。	0 ～ 65535
invalid-packet-inspection	SNMP プロトコル違反を検査します。	—
non-snmp-traffic-inspection	UDP ポート 161 宛の非 SNMP トラフィックを検査します。	—
snmp-inspection	SNMP トラフィックを検査します。 • specify-community-name [yes | no]: – community-name：SNMP コミュニティ名、つまり SNMP パスワードを検索します。 • specify-object-id [yes | no]: – object-id：SNMP オブジェクト ID を検索します。	*community-name* *object-id*

Service SSH エンジン

Service SSH エンジンは、ポート 22 の SSH トラフィックに対して使用します。SSH セッションのセットアップを除いてすべてが暗号化されるため、エンジンはセットアップのフィールドだけをモニタします。SSH には 2 つのデフォルト シグニチャがあります。これらのシグニチャを調整することはできますが、カスタム シグニチャは作成できません。

表 B-31 に、Service SSH エンジンに固有のパラメータを示します。

表 B-31　Service SSH エンジンのパラメータ

パラメータ	説明	値
SSH Version		
length-type	次の SSH 長さタイプのいずれかを検査します。 • key-length：検査対象の SSH キーの長さ。 – length：キーがこれよりも長い場合は、RSAREF オーバーフローが発生します。 • user-length：ユーザ長の SSH 検査。 – length：キーがこれよりも長い場合は、RSAREF オーバーフローが発生します。	0 ～ 65535

表 B-31　Service SSH エンジンのパラメータ（続き）

パラメータ	説明	値
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
specify-packet-depth	（任意）パケット数をイネーブルにします。 • packet-depth：セッション キーが失われたと判断するまでに監視するパケット数。	0 ～ 65535

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

Service TNS エンジン

Service TNS エンジンは、TNS プロトコルを検査します。TNS は、すべての業界標準ネットワーク プロトコルに対して、単一の共通インターフェイスでデータベース アプリケーションを提供します。TNS では、プロトコルが異なるネットワークをまたいで、アプリケーションを他のデータベース アプリケーションに接続できます。デフォルトの TNS リスナー ポートは、TCP 1521 です。TNS では、別のホストや別の TCP ポートにクライアントをリダイレクトする REDIRECT フレームもサポートされています。REDIRECT パケットをサポートするために、TNS エンジンは、すべての TCP ポートをリッスンし、TNS 以外のストリームを無視するための高速な TNS フレーム ヘッダー検証ルーチンを備えています。

表 B-32 に、Service TNS エンジンに固有のパラメータを示します。

表 B-32　Service TNS エンジンのパラメータ

パラメータ	説明	値
direction	トラフィックの方向。 • サービス ポートからクライアント ポート宛のトラフィック • クライアント ポートからサービス ポート宛のトラフィック	from-service to-service
type	TNS フレーム値のタイプを指定します。 • 1：接続 • 2：受け入れ • 4：拒否 • 5：リダイレクト • 6：データ • 11：再送信 • 12：マーカー	1 2 4 5 6 11 12

表 B-32　Service TNS エンジンのパラメータ（続き）

パラメータ	説明	値
specify-regex-string	（任意）正規表現文字列の使用をイネーブルにします。 • specify-exact-match-offset：完全一致オフセットをイネーブルにします。 – exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。 • specify-min-match-length：最小一致長をイネーブルにします。 – min-match-length：正規表現文字列が一致する必要があるバイトの最小数。	0 ～ 65535
specify-regex-payload-src	検査するプロトコルを指定します。 payload-src: • tcp-data：TCP パケットのデータ部分に対して正規表現を実行します。 • tns-data：すべての空白が削除されている TNS データに対してだけ正規表現を実行します。	tcp tns

詳細情報

シグニチャの正規表現の構文リストについては、「正規表現の構文」（P.B-9）を参照してください

State エンジン

State エンジンは、TCP ストリームの状態に基づく正規表現ベースのパターン検査を提供します。State エンジンは何かの状態を保存するデバイスで、入力があるたびに、その内容に基づいてある状態から別の状態に遷移したり、処理や出力を行ったりできます。ステート マシンは、出力やアラームを発生させる特定のイベントを記述するために使用します。State エンジンには、SMTP、Cisco Login、および LPR Format String の 3 つのステート マシンがあります。

表 B-33 に、State エンジンに固有のパラメータを示します。

表 B-33　State エンジンのパラメータ

パラメータ	説明	値
state-machine	ステート マシン グループ。	• smpt • lpr-format-string • cisco-login
cisco-login	Cisco ログインのステート マシンを指定します。 • state-name：状態の名前。この状態になると、シグニチャはアラートを起動します。 – シスコ デバイスの状態 – Control-C 状態 – パスワード プロンプト状態 – 開始状態	• cisco-device • control-c • pass-prompt • start
lpr-format-string	LPR フォーマット ストリングの脆弱性を検査するステート マシンを指定します。 • state-name：状態の名前。この状態になると、シグニチャはアラートを起動します。 – LPR フォーマット ストリング検査を終了する中断状態 – フォーマット文字の状態 – 開始状態	• abort • format-char • start
state-name	SMTP プロトコルのステート マシンを指定します。 • state-name：状態の名前。この状態になると、シグニチャはアラートを起動します。 – LPR フォーマット ストリング検査を終了する中断状態 – メール本文の状態 – メール ヘッダーの状態 – SMTP コマンドの状態 – 開始状態	• abort • mail-body • mail-header • smtp-commands • start
direction	トラフィックの方向： • サービス ポートからクライアント ポート宛のトラフィック。 • クライアント ポートからサービス ポート宛のトラフィック。	from-service to-service
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]

表 B-33　State エンジンのパラメータ（続き）

パラメータ	説明	値
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535
specify-max-match-offset	（任意）最大一致オフセットをイネーブルにします。 • max-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある最大ストリーム オフセット。	0 ～ 65535
specify-min-match-offset	（任意）最小一致オフセットをイネーブルにします。 • min-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある最小ストリーム オフセット。	0 ～ 65535
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：正規表現文字列が一致する必要があるバイトの最小数。	0 ～ 65535
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true| false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

String エンジン

ここでは、String エンジンについて説明します。内容は次のとおりです。

String エンジンについて

String エンジンは、TCP、UDP および ICMP の各プロトコルを対象とした、汎用のパターン マッチング インスペクション エンジンです。String エンジンでは、複数のパターンを 1 つのパターン マッチング テーブルにまとめることでデータ内の検索を一度に実行できる新しい正規表現エンジンが使用されます。String ICMP、String TCP、および String UDP の 3 つの String エンジンがあります。

String ICMP エンジンのパラメータ

表 B-34 に、String ICMP エンジンに固有のパラメータを示します。

表 B-34　String ICMP エンジンのパラメータ

パラメータ	説明	値
direction	トラフィックの方向： • サービス ポートからクライアント ポート宛のトラフィック。 • クライアント ポートからサービス ポート宛のトラフィック。	from-service to-service
icmp-type	ICMP ヘッダーの TYPE 値。	0 ～ 18[1] a-b[,c-d]
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：正規表現文字列が一致する必要があるバイトの最小数。	0 ～ 65535
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true| false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

String TCP エンジンのパラメータ

表 B-35 に、String TCP エンジンに固有のパラメータを示します。

表 B-35　String TCP エンジン

パラメータ	説明	値
direction	トラフィックの方向： • サービス ポートからクライアント ポート宛のトラフィック。 • クライアント ポートからサービス ポート宛のトラフィック。	from-service to-service
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535

表 B-35　String TCP エンジン（続き）

パラメータ	説明	値
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：正規表現文字列が一致する必要があるバイトの最小数。	0 ～ 65535
strip-telnet-options	パターンを検索する前に、データから Telnet オプション文字を削除します。[2]	true | false
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true| false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。
2. このパラメータは、主に、IPS 反回避ツールとして使用します。

詳細情報

カスタム String エンジン シグニチャの例については、「String TCP シグニチャの例」（P.8-42）を参照してください。

String UDP エンジンのパラメータ

表 B-36 に、String UDP エンジンに固有のパラメータを示します。

表 B-36　String UDP エンジン

パラメータ	説明	値
direction	トラフィックの方向： • サービス ポートからクライアント ポート宛のトラフィック。 • クライアント ポートからサービス ポート宛のトラフィック。	from-service to-service
service-ports	ターゲット サービスが常駐する、カンマ区切りのポートのリストまたはポート範囲。	0 ～ 65535[1] a-b[,c-d]
specify-exact-match-offset	（任意）完全一致オフセットをイネーブルにします。 • exact-match-offset：一致を有効にするために正規表現文字列がレポートする必要がある正確なストリーム オフセット。	0 ～ 65535
specify-min-match-length	（任意）最小一致長をイネーブルにします。 • min-match-length：正規表現文字列が一致する必要があるバイトの最小数。	0 ～ 65535
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true| false

1. 範囲の 2 番目の数は、最初の数以上である必要があります。

詳細情報

カスタム String エンジン シグニチャの例については、「String TCP シグニチャの例」（P.8-42）を参照してください。

Sweep エンジン

ここでは、Sweep エンジンについて説明します。内容は次のとおりです。

Sweep エンジン

Sweep エンジンでは、2 台のホスト間、または 1 台のホストから多数のホストへのトラフィックが分析されます。既存のシグニチャを調整することも、カスタム シグニチャを作成することもできます。Sweep エンジンには、ICMP、UDP、および TCP に対するプロトコル固有のパラメータがあります。

Sweep エンジンのアラート条件は、最終的に一意のパラメータの数に依存します。一意のパラメータは、スイープのタイプによって、個別のホストまたはポートの数のしきい値です。期間内にアドレス セットで一意のポートまたはホストの数を超えていることが確認された場合、一意のパラメータによってアラートがトリガーされます。一意のポートおよびホストのトラッキング処理をカウンティングと言います。

注意　送信元および宛先の IP アドレスに基づくイベント アクション フィルタは、通常のシグニチャとしてフィルタリングしないため、Sweep エンジンでは機能しません。スイープ アラートで送信元と宛先の IP アドレスにフィルタリングするには、Sweep エンジン シグニチャで送信元と宛先の IP アドレス フィルタ パラメータを使用します。

Sweep エンジンのすべてのシグニチャに対して、一意のパラメータを指定する必要があります。スイープには、2 以上、40 以下の制限が適用されます。2 は、スイープの絶対最小値であり、この値未満の場合は、（1 つのホストまたはポートの）スイープではありません。40 は、スイープによって過度にメモリが消費されないようにするために、適用する必要がある実際的な最大値です。一意の範囲の現実的な値は、5 よりも大きく 15 未満です。

TCP スイープには、個別の接続をカウントするスイープ インスペクタ スロットを決定するために指定する TCP フラグおよびマスクが必要です。ICMP スイープには、さまざまなタイプの ICMP パケットを区別するために指定する ICMP タイプが必要です。

DataNode

Sweep エンジン シグニチャに関連するアクティビティが確認された場合、IPS は、DataNode を使用して、特定のホストのモニタリングを停止するかどうかを決定します。DataNode には、ストリームのクロスパケット再構築のために、またストリーム別、送信元別、宛先別で検査状態を追跡するために必要なさまざまな永続的カウンタおよび変数が含まれます。スイープが含まれている DataNode は、スイープが期限切れとなる時期を決定します。DataNode は、*x* 秒（プロトコルによって異なります）にわたってトラフィックが確認されなかった場合に、スイープを停止します。

DataNode には、複数の適応型タイムアウトがあります。DataNode は、含まれるすべてのオブジェクトが削除された後、アドレス セットで 30 秒のアイドル時間が経過した後に期限が切れます。含まれる各オブジェクトには、さまざまなタイムアウトが設定されています。たとえば、確立された接続に対して、TCP ストリームには 1 時間のタイムアウトが設定されます。ほとんどの他のオブジェクトには、5 秒または 60 秒など、はるかに短い有効期限が設定されます。

表 B-37 に、Sweep エンジンに固有のパラメータを示します。

表 B-37　Sweep エンジンのパラメータ

パラメータ	説明	値
dst-addr-filter	スイープ カウント アルゴリズムから除外する宛先 IP アドレス。	<A.B.C.D>-<A.B.C.D> [,<A.B.C.D>-<A.B.C.D>]
src-addr-filter	スイープ カウント アルゴリズムから除外する送信元 IP アドレス。	<A.B.C.D>-<A.B.C.D> [,<A.B.C.D>-<A.B.C.D>]
protocol	このインスペクタの該当プロトコル。	• icmp • udp • tcp
specify-icmp-type	（任意）ICMP ヘッダー タイプの検査をイネーブルにします。 • icmp-type：ICMP ヘッダーの TYPE 値を指定します。	0 ～ 255
specify-port-range	（任意）検査でのポート範囲の使用をイネーブルにします。 • port-range：検査で使用する UDP ポート範囲。	0 ～ 65535 a-b[,c-d]
fragment-status	フラグメントが必要かどうかを指定します。 • 任意のフラグメント ステータス。 • フラグメントを検査しない。 • フラグメントを検査する。	• any • no-fragments • want-fragments
inverted-sweep	一意のカウントの対象として宛先ポートではなく送信元ポートを使用します。	true | false
mask	TCP フラグの比較に使用するマスク： • URG ビット • ACK ビット • PSH ビット • RST ビット • SYN ビット • FIN ビット	• urg • ack • psh • rst • syn • fin

表 B-37　Sweep エンジンのパラメータ（続き）

パラメータ	説明	値
storage-key	固定データを保存するために使用するアドレス キーのタイプ。 • 攻撃者のアドレス • 攻撃者と攻撃対象のアドレス • 攻撃者のアドレスと攻撃対象のポート	Axxx AxBx Axxb
suppress-reverse	このアドレス セットで反対方向にスイープが実行されている場合、アラートを起動しません。	true| false
swap-attacker-victim	アラート メッセージとアクションで攻撃者と攻撃対象のアドレスとポート（送信元と宛先）をスワップする場合は True です。スワップしない場合は False です（デフォルト）。	true| false
tcp-flags	マスクによってマスクされた場合に照合する TCP フラグ。 • URG ビット • ACK ビット • PSH ビット • RST ビット • SYN ビット • FIN ビット	• urg • ack • psh • rst • syn • fin
unique	2 つのホスト間の一意のポート接続数のしきい値。	0 ～ 65535

Sweep Other TCP エンジン

Sweep Other TCP エンジンでは、2 台のホスト間のトラフィックが分析され、一般に攻撃対象のフィンガープリントに使用される異常なパケットが検出されます。既存のシグニチャを調整することも、カスタム シグニチャを作成することもできます。

TCP スイープには、指定した TCP フラグおよびマスクが必要です。TCP フラグのセットに、複数のエントリを指定できます。また、オプションのポート範囲を指定して、特定のパケットをフィルタで排除できます。

表 B-38 に、Sweep Other TCP エンジンに固有のパラメータを示します。

表 B-38　Sweep Other TCP エンジンのパラメータ

パラメータ	説明	値
specify-port-range	（任意）検査でのポート範囲の使用をイネーブルにします。 • port-range：検査で使用する UDP ポート範囲。	0 ～ 65535 a-b[,c-d]
set-tcp-flags	照合する TCP フラグを設定します。 • tcp-flags：検査で使用される TCP フラグ。 – URG ビット – ACK ビット – PSH ビット – RST ビット – SYN ビット – FIN ビット	• urg • ack • psh • rst • syn • fin

トラフィック異常エンジン

トラフィック異常エンジンには、3 つのプロトコル（TCP、UDP、およびその他）をカバーする 9 つの異常検出シグニチャが含まれます。各シグニチャには 2 つのサブシグニチャがあります。一方はスキャナ用で、もう一方はワームに感染したホスト（またはワーム攻撃されているスキャナ）用です。異常検出によって異常が発見された場合は、これらのシグニチャに対してアラートがトリガーされます。すべての異常検出シグニチャは、デフォルトでイネーブルになり、各シグニチャのアラート重大度は高く設定されます。

スキャナが検出されても、ヒストグラム異常が発生しない場合、スキャナ シグニチャはその攻撃者（スキャナ）の IP アドレスをファイルに保存します。ヒストグラム シグニチャがトリガーされた場合は、スキャンを行っている攻撃者のアドレスによってそれぞれ（スキャナ シグニチャではなく）ワーム シグニチャがトリガーされます。ヒストグラムがトリガーされたため、アラートの詳細には、ワームの検出に使用されているしきい値が表示されます。この時点以降、すべてのスキャナは、ワームに感染したホストとして検出されます。

次の異常検出イベント アクションが可能です。

- Produce alert：イベント ストアにイベントを書き込みます。
- Deny attacker inline：（インライン モードのみ）指定された期間、この攻撃者のアドレスから発生した現在のパケットおよび将来のパケットを送信しません。
- Log attacker pairs：攻撃者のアドレスが含まれているパケットに対する IP ロギングを開始します。
- Log pair packets：攻撃者と攻撃対象のアドレス ペアが含まれているパケットに対する IP ロギングを開始します。
- Deny attacker service pair inline：送信元 IP アドレスと宛先ポートをブロックします。
- Request SNMP trap：要求を NotificationApp に送信して、SNMP 通知を実行します。
- Request block host：要求を ARC に送信して、このホスト（攻撃者）をブロックします。

（注） 異常検出シグニチャを編集または調整することはできますが、カスタム異常検出シグニチャを作成することはできません。

表 39 に、異常検出ワーム シグニチャを示します。

表 39　異常検出ワーム シグニチャ

シグニチャ ID	サブシグニチャ ID	名前	説明
13000	0	Internal TCP Scanner	内部ゾーンで TCP プロトコル上に単一スキャナを識別しました。
13000	1	Internal TCP Scanner	内部ゾーンで TCP プロトコル上にワーム攻撃を識別しました。TCP ヒストグラムのしきい値を超え、TCP プロトコル上にスキャナが識別されました。
13001	0	Internal UDP Scanner	内部ゾーンで UDP プロトコル上に単一スキャナを識別しました。
13001	1	Internal UDP Scanner	内部ゾーンで UDP プロトコル上にワーム攻撃を識別しました。UDP ヒストグラムのしきい値を超え、UDP プロトコル上にスキャナが識別されました。
13002	0	Internal Other Scanner	内部ゾーンでその他のプロトコル上に単一スキャナを識別しました。
13002	1	Internal Other Scanner	内部ゾーンでその他のプロトコル上にワーム攻撃を識別しました。その他のヒストグラムのしきい値を超え、その他のプロトコル上にスキャナが識別されました。
13003	0	External TCP Scanner	外部ゾーンで TCP プロトコル上に単一スキャナを識別しました。
13003	1	External TCP Scanner	外部ゾーンで TCP プロトコル上にワーム攻撃を識別しました。TCP ヒストグラムのしきい値を超え、TCP プロトコル上にスキャナが識別されました。
13004	0	External UDP Scanner	外部ゾーンで UDP プロトコル上に単一スキャナを識別しました。
13004	1	External UDP Scanner	外部ゾーンで UDP プロトコル上にワーム攻撃を識別しました。UDP ヒストグラムのしきい値を超え、UDP プロトコル上にスキャナが識別されました。
13005	0	External Other Scanner	外部ゾーンでその他のプロトコル上に単一スキャナを識別しました。
13005	1	External Other Scanner	外部ゾーンでその他のプロトコル上にワーム攻撃を識別しました。その他のヒストグラムのしきい値を超え、その他のプロトコル上にスキャナが識別されました。
13006	0	Illegal TCP Scanner	不正ゾーンで TCP プロトコル上に単一スキャナを識別しました。
13006	1	Illegal TCP Scanner	不正ゾーンで TCP プロトコル上にワーム攻撃を識別しました。TCP ヒストグラムのしきい値を超え、TCP プロトコル上にスキャナが識別されました。

表 39　異常検出ワーム シグニチャ（続き）

シグニチャ ID	サブシグニチャ ID	名前	説明
13007	0	Illegal UDP Scanner	不正ゾーンで UDP プロトコル上に単一スキャナを識別しました。
13007	1	Illegal UDP Scanner	不正ゾーンで UDP プロトコル上にワーム攻撃を識別しました。UDP ヒストグラムのしきい値を超え、UDP プロトコル上にスキャナが識別されました。
13008	0	Illegal Other Scanner	不正ゾーンでその他のプロトコル上に単一スキャナを識別しました。
13008	1	Illegal Other Scanner	不正ゾーンでその他のプロトコル上にワーム攻撃を識別しました。その他のヒストグラムのしきい値を超え、その他のプロトコル上にスキャナが識別されました。

Traffic ICMP エンジン

Traffic ICMP エンジンは、TFN2K、LOKI、DDoS などの非標準プロトコルを分析します。このエンジンには、ユーザが設定可能なパラメータを持つ 2 つのシグニチャ（LOKI プロトコルに基づく）だけが含まれます。

TFN2K は、TFN の新しいバージョンです。TFN2K は DDoS エージェントの一種であり、感染した複数のコンピュータ（ゾンビ）による協調した攻撃（何百または何千もの未知の攻撃ホストから 1 つのコンピュータまたはドメインに向けて偽のトラフィック フラッドを送信する攻撃）を制御します。TFN2K はランダムに抽出されたパケット ヘッダー情報を送信しますが、それにはシグニチャの定義に使用できる 2 つの識別子が付いています。1 つは L3 チェックサムが不正かどうかを示し、もう 1 つはペイロードの末尾に文字 64「A」が検出されたかどうかを示します。TFN2K は、任意のポートで実行可能であり、ICMP、TCP、UDP、またはこれらのプロトコルの組み合わせを使用して通信できます。

LOKI は、バック ドア型トロイの木馬タイプです コンピュータが感染すると、悪意のあるコードにより ICMP トンネルが作成されます。この ICMP トンネルは、ICMP 応答内での小さなペイロードの送信に使用されるおそれがあります（ICMP をブロックするように設定していないと、ICMP 応答はファイアウォールを通過することがあります）。LOKI シグニチャは、ICMP エコーの要求と応答のアンバランス、簡易 ICMP コード、およびペイロード識別子をモニタします。

（TFN2K を除く）DDOS カテゴリは、ICMP ベースの DDOS エージェントを対象とします。ここで使用する主なツールは、TFN と Stacheldraht です。これらは TFN2K と同様に動作しますが、ICMP だけに依存し、固定コマンド（整数および文字列）を備えています。

表 B-40 に、Traffic ICMP エンジンに固有のパラメータを示します。

表 B-40　Traffic ICMP エンジンのパラメータ

パラメータ	説明	値
parameter-tunable-sig	設定可能なパラメータがシグニチャに存在するかどうか。	yes | no
inspection-typee	実行する検査のタイプ： • 最初の LOKI トラフィックを検査する。 • 変更された LOKI トラフィックを検査する。	is-loki is-mod-loki

表 B-40　Traffic ICMP エンジンのパラメータ（続き）

パラメータ	説明	値
reply-ratio	要求と応答のアンバランス。要求と比べて、応答が指定した数より多い場合に、アラートを起動します。	0 ～ 65535
want-request	アラートを起動する前に、ECHO REQUEST の検出が必要となります。	true | false

Trojan エンジン

Trojan エンジンは、BO2K および TFN2K などの非標準プロトコルを分析します。Trojan BO2K、TrojanTFN2K、および Trojan UDP の 3 つの Trojan エンジンがあります。

BO は、UDP だけで実行される最初の Windows バックドア型トロイの木馬でした。これは、間もなく BO2K に置き換えられました。BO2K は、基本的な XOR 暗号を利用する UDP と TCP のどちらにも対応しています。これには、特定のクロスパケット特性を備えたプレーン BO ヘッダーがあります。

BO2K には、BO ヘッダーを暗号化し、クロスパケット パターンをほとんど認識不可能にするために設計されたステルス TCP モジュールもあります。BO と BO2K の UDP モードは、Trojan UDP エンジンによって処理されます。TCP モードは、Trojan BO2K エンジンによって処理されます。

（注）　Trojan UDP エンジンの swap-attacker-victim を除き、Trojan エンジンに固有なパラメータはありません。