CHAPTER 7

# シグニチャの定義

（注） 現在、Cisco IPS 7.1 をサポートしているプラットフォームは、IPS SSP を搭載した Cisco ASA 5585 のみです。それ以外の Cisco IPS センサーは、IPS 7.1 を現在サポートしていません。

（注） IPS SSP を搭載した Cisco ASA 5585 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。

この章では、シグニチャを定義し、作成する方法について説明します。次のような構成になっています。



## セキュリティ ポリシーについて

複数のセキュリティ ポリシーを作成し、それらを個々の仮想センサーに適用できます。セキュリティ ポリシーは、シグニチャ定義ポリシー、イベント アクション規則ポリシー、および異常検出ポリシーから構成されます。Cisco IPS には、sig0 という名前のデフォルトのシグニチャ定義ポリシー、rules0 という名前のデフォルトのイベント アクション規則ポリシー、および ad0 という名前のデフォルトの異常検出ポリシーが用意されています。仮想センサーにこれらのデフォルト ポリシーを割り当てることも、新しいポリシーを作成することもできます。複数のセキュリティ ポリシーを使用すれば、それぞれ異なる要件に基づいたセキュリティ ポリシーを作成し、そうしたカスタマイズされたポリシーを VLAN や物理インターフェイスごとに適用することが可能になります。

## シグニチャ定義ポリシーの操作

サービス シグニチャ定義モードで **service signature-definition** *name* コマンドを使用して、シグニチャ定義ポリシーを作成します。このシグニチャ定義ポリシーの値は、それらを編集するまでは、デフォルトのシグニチャ定義ポリシー sig0 と同じです。

あるいは、特権 EXEC モードで **copy signature-definition** *source_destination* コマンドを使用して既存のポリシーのコピーを作成してから、必要に応じてその新しいポリシーの値を編集できます。

特権 EXEC モードで **list signature-definition-configurations** コマンドを使用して、シグニチャ定義ポリシーを一覧表示します。

グローバル コンフィギュレーション モードで **no service signature-definition** *name* コマンドを使用して、シグニチャ定義ポリシーを削除します。グローバル コンフィギュレーション モードで **default service signature-definition** *name* コマンドを使用して、シグニチャ定義ポリシーを工場出荷時の設定にリセットします。

シグニチャ定義ポリシーを作成、コピー、編集、および削除するには、次の手順を実行します。

**ステップ 1**　管理者権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義ポリシーを作成します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition MySig
Editing new instance MySig.
ips-ssp(config-sig)# exit
Apply Changes?[yes]: yes
ips-ssp(config)# exit
```

**ステップ 3**　または、既存のシグニチャ定義ポリシーを新しいシグニチャ定義ポリシーにコピーします。

```
ips-ssp# copy signature-definition sig0 sig1
ips-ssp#
```

**（注）**　ポリシーがすでに存在しているか、新しいポリシーに必要な容量が不足していると、エラーが表示されます。

**ステップ 4**　デフォルトのシグニチャ定義ポリシーの値を受け入れるか、次のようなパラメータの編集を行います。

**a.**　シグニチャ定義変数を追加します。

**b.**　一般的なシグニチャ オプションを設定します。

**ステップ 5**　IPS SSP 上のシグニチャ定義ポリシーのリストを表示します。

```
ips-ssp# list signature-definition-configurations
Signature Definition
   Instance   Size   Virtual Sensor
   sig0       255    vs0
   temp       707    N/A
   MySig      255    N/A
   sig1       141    vs1
ips-ssp#
```

**ステップ 6**　シグニチャ定義ポリシーを削除します。

```
ips-ssp# configure terminal
ips-ssp(config)# no service signature-definition MySig
ips-ssp(config)# exit
ips-ssp#
```

**（注）**　デフォルトのシグニチャ定義ポリシー sig0 は削除できません。

**ステップ 7**　シグニチャ定義ポリシーが削除されていることを確認します。

```
ips-ssp# list signature-definition-configurations
Signature Definition
   Instance   Size   Virtual Sensor
   sig0       255    vs0
   temp       707    N/A
   sig1       141    vs1
ips-ssp#
```

**ステップ 8**　シグニチャ定義ポリシーを工場出荷時の設定にリセットします。

```
ips-ssp# configure terminal
ips-ssp(config)# default service signature-definition sig1
ips-ssp(config)#
```

### 詳細情報

- シグニチャ変数を追加する手順については、「シグニチャ変数の設定」（P.7-4）を参照してください。
- 一般設定の手順については、「シグニチャの設定」（P.7-6）を参照してください。

# シグニチャについて

攻撃またはその他のネットワーク リソースの不正使用は、ネットワークへの侵入として定義付けることができます。シグニチャベースのテクノロジーを使用するセンサーによって、ネットワーク侵入を検出できます。シグニチャは、DoS 攻撃などの典型的な侵入行為を検出するためにセンサーが使用する一連の規則です。センサーは、ネットワーク パケットをスキャンするときに、シグニチャを使って既知の攻撃を検出し、指定されたアクションに従って対応します。

センサーは、一連のシグニチャとネットワーク アクティビティを比較します。一致した場合、イベントのロギングやアラームの送信などのアクションを実行します。センサーでは、既存のシグニチャを変更したり、新しいシグニチャを定義したりできます。

シグニチャ ベースの侵入検出では、偽陽性が生じる場合があります。通常のネットワーク アクティビティでも、悪意のあるアクティビティとして誤解される場合があるためです。たとえば、一部のネットワーク アプリケーションやオペレーティング システムは、多数の ICMP メッセージを送信することがありますが、シグニチャベースの検出システムでは、このメッセージが攻撃者によるネットワーク セグメント特定の試みであると解釈されてしまう可能性があります。シグニチャをチューニングすると、偽陽性を最小限に抑えることができます。

特定のシグニチャを使ってネットワーク トラフィックをモニタするようにセンサーを設定するには、そのシグニチャをイネーブルにする必要があります。デフォルトでは、重要なシグニチャはシグネチャ アップデートのインストール時にイネーブルになります。イネーブルなシグネチャに一致する攻撃が検出されると、センサーはアラートを生成します。生成されたアラートはセンサーのイベント ストアに保存されます。Web ベース クライアントは、アラートやその他のイベントをイベント ストアから取得できます。デフォルトでは、センサーは Informational 以上のすべてのアラートをログに記録します。

シグニチャには、サブシグニチャを持つもの（サブカテゴリに分類されているもの）があります。サブシグニチャを設定した場合、あるサブシグニチャのパラメータを変更しても、変更が適用されるのはそのサブシグニチャだけです。たとえば、シグニチャ 3050 のサブシグニチャ 1 を編集し重大度を変更した場合、重大度の変更はサブシグニチャ 1 だけに適用され、3050 2、3050 3、および 3050 4 には適用されません。

Cisco IPS には、10,000 を超えるデフォルトの組み込みシグニチャが含まれています。組み込みシグネチャのリストにあるシグネチャの名前の変更および削除はできません。ただし、シグネチャをセンシング エンジンから削除して廃棄できます。あとで廃棄されたシグネチャをアクティブにできます。ただし、このプロセスにはセンシング エンジンの設定の再構築が必要です。この再構築には時間がかかり、トラフィックの処理を遅延させる可能性があります。組み込みシグニチャのチューニングは可能です。これには、シグニチャのいくつかのパラメータを変更します。変更された組み込みシグニチャは、チューニング済みシグニチャと呼ばれます。

**（注）** 使用していないシグネチャを廃棄することを推奨します。廃棄によって、センサーのパフォーマンスが向上します。

カスタム シグニチャと呼ばれるシグニチャを作成できます。カスタム シグニチャ ID は、60000 から始まります。いくつかの項目に対して、カスタム シグニチャを設定できます。たとえば、UDP 接続の文字列との一致やネットワーク フラッドの追跡、スキャンなどです。シグニチャは、モニタするトラフィックの種類に対して特別に設計されたシグニチャ エンジンを使って作成します。

# シグニチャ変数の設定

ここでは、シグニチャ変数について説明します。次のような構成になっています。



## シグニチャ変数について

複数のシグニチャで同じ値を使用する場合、変数を使用します。変数の値を変更すると、その変数を使用しているすべてのシグニチャでその変数の値が更新されます。このため、シグニチャを設定するときに変数を繰り返し変更しなくて済みます。

**（注）** 文字列でなく変数を使っていることを示すために、変数をドル記号（$）で始める必要があります。

一部の変数は、シグニチャ システムに対して必須であるため、削除することはできません。変数が保護されている場合は、その変数を選択して編集することはできません。保護されている変数を削除しようとすると、エラー メッセージが表示されます。一度に編集できる変数は 1 つだけです。

## シグニチャ変数の追加、編集、および削除

シグニチャ定義サブモードで **variables** コマンドを使用して、シグニチャ変数を作成します。

### オプション

次のオプションが適用されます。

- *variable_name*：この変数の名前を示します。有効な名前は、数字と文字のみで構成されます。ハイフン（-）とアンダースコア（_）も使用できます。
- **ip-addr-range**：IP アドレスのグループを表すシステム定義変数を指定します。有効な値は、A.B.C.D-A.B.C.D[,A.B.C.D-A.B.C.D] の形式になります。

- **web-ports**：HTTP トラフィックを検索するポートを表すシステム定義変数を指定します。1 つの変数に複数のポート番号を指定する場合は、エントリ間にカンマを入力します。たとえば、80, 3128, 8000, 8010, 8080, 8888, 24326 と入力します。

### シグニチャ変数の追加、編集、および削除

シグニチャ変数を追加、編集、および削除するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　IP アドレスのグループを表すシグニチャ変数を作成します。

```
ips-ssp(config-sig)# variables IPADD ip-addr-range 10.1.1.1-10.1.1.24
```

**ステップ 4**　Web ポートを表すシグニチャ変数を編集します。WEBPORTS は Web サーバが実行されているポート群で、あらかじめ定義されているものですが、値は編集できます。この変数は、Web ポートが含まれるすべてのシグニチャに影響します。デフォルトは、80, 3128, 8000, 8010, 8080, 8888, 24326 です。

```
ips-ssp(config-sig)# variables WEBPORTS web-ports 80,3128,8000
```

**ステップ 5**　変更を確認します。

```
ips-ssp(config-sig)# show settings
   variables (min: 0, max: 256, current: 2)
   -----------------------------------------------
      variable-name: IPADD
      -----------------------------------------------
         ip-addr-range: 10.1.1.1-10.1.1.24
      -----------------------------------------------
      <protected entry>
      variable-name: WEBPORTS
      -----------------------------------------------
         web-ports: 80,3128,8000 default: 80-80,3128-3128,8000-8000,8010-8010,80
80-8080,8888-8888,24326-24326
      -----------------------------------------------
```

**ステップ 6**　変数を削除します。

```
ips-ssp(config-sig)# no variables IPADD
```

**ステップ 7**　変数が削除されていることを確認します。

```
ips-ssp(config-sig)# show settings
   variables (min: 0, max: 256, current: 1)
   -----------------------------------------------
      <protected entry>
      variable-name: WEBPORTS
      -----------------------------------------------
         web-ports: 80,3128,8000 default: 80-80,3128-3128,8000-8000,8010-8010,80
80-8080,8888-8888,24326-24326
      -----------------------------------------------
```

**ステップ 8**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 9**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# シグニチャの設定

ここでは、シグニチャ パラメータを設定する方法について説明します。次のような構成になっています。



## シグニチャ定義オプション

特定のシグニチャの一般パラメータの設定には、次のオプションが適用されます。

- **alert-frequency**：アラートをグループ化するためのサマリー オプションを設定します。
- **alert-severity**：アラートの重大度を設定します。
- **engine**：シグニチャ エンジンを指定します。アクションはエンジン サブモードのときに割り当てることができます。
- **event-counter**：イベント カウントを設定します。
- **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。

**注意**　シグニチャの無差別デルタ設定を変更することは推奨されません。

- **sig-description**：シグニチャの説明。
- **sig-fidelity-rating**：シグニチャの忠実度のレーティングを指定します。
- **status**：シグニチャのステータスをイネーブルまたは廃棄に設定します。
- **vulnerable-os**：この攻撃シグニチャに対して脆弱な OS タイプのリストを指定します。

### 詳細情報

- アラート頻度の設定手順については、「アラート頻度の設定」（P.7-7）を参照してください。
- シグニチャ エンジンの詳細については、付録 B「シグニチャ エンジンの概要」を参照してください。

- アクションの割り当て手順については、「シグニチャへのアクションの割り当て」（P.7-16）を参照してください。
- イベント カウントの設定手順については、「イベント カウンタの設定」（P.7-11）を参照してください。
- 無差別デルタの詳細については、「無差別デルタ」（P.7-7）を参照してください。
- シグニチャ忠実度レーティングの設定手順については、「シグニチャ忠実度レーティングの設定」（P.7-12）を参照してください。
- シグニチャのイネーブル化とディセーブル化の手順については、「シグニチャのステータスの設定」（P.7-13）を参照してください。
- 脆弱性のある OS の設定手順については、「シグニチャの脆弱性のある OS の設定」（P.7-14）を参照してください。

# 無差別デルタ

**注意**　シグニチャの無差別デルタ設定を変更することは推奨されません。

無差別デルタは、無差別モードにおいて特定のアラートのリスク レーティングを引き下げます。センサーはターゲット システムの属性を認識せず、無差別モードではパケットを拒否できないため、管理者がリスク レーティングの高いアラートの調査に集中できるよう、（リスク レーティングの低さに基づいて）無差別アラートの優先順位を下げるのに役立ちます。

インライン モードでは、センサーで攻撃パケットを拒否することができ、それらがターゲット ホストに到達することはないので、ターゲットに脆弱性があったとしても問題になりません。ネットワーク上での攻撃が不可能になっていれば、リスク レーティングの値は引き下げません。

サービス、OS、アプリケーションのいずれにも固有ではないシグニチャの場合、無差別デルタの値は 0 になります。シグニチャが OS、サービス、またはアプリケーションに固有の場合は、カテゴリごとに 5 つのポイントで 5、10、または 15 の無差別デルタが計算されます。

### 詳細情報

- 無差別モードの詳細については、「無差別モード」（P.5-3）を参照してください。
- リスク レーティングの詳細については、「リスク レーティングの計算」（P.8-12）を参照してください。

# アラート頻度の設定

シグニチャ定義サブモードで **alert-frequency** コマンドを使用して、シグニチャのアラート頻度を設定します。**alert-frequency** コマンドは、このシグニチャが起動しているときにセンサーがユーザにアラートを発行する頻度を指定します。

### オプション

次のオプションが適用されます。

- *sig_id*：このシグニチャに割り当てられた一意の数値を示します。この値により、センサーは特定のシグニチャを識別します。値は 1000 ～ 65000 です。
- *subsig_id*：このサブシグニチャに割り当てられた一意の数値を示します。サブシグニチャ ID は、広範なシグニチャをより詳細に識別するために使用します。値は 0 ～ 255 です。

- **summary-mode**：センサーがアラートをグループ化する方法を指定します。
  - **fire-all**：すべてのイベントに対し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**：すべてのアラートをサマライズします。
- **specify-summary-threshold {yes | no}**：サマリーしきい値モードをイネーブルにします。
  - **summary-threshold**：このシグニチャのサマリー アラートが送信される前にセンサーで受信されなければならない最小ヒット数を指定します。値は 0 ～ 65535 です。
  - **summary-interval**：各サマリー アラートで使用される時間間隔（秒数）を指定します。値は 1 ～ 1000 です。
- **summary-key**：このシグニチャをサマライズするストレージ タイプを指定します。
  - **Axxx**：攻撃者のアドレス。
  - **Axxb**：攻撃者のアドレスと攻撃対象のポート。
  - **AxBx**：攻撃者と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。
  - **xxBx**：攻撃対象のアドレス。
- **specify-global-summary-threshold {yes | no}**：（任意）グローバル サマリーしきい値モードをイネーブルにします。
  - **global-summary-threshold**：アラートがグローバル サマリーにまとめられるイベント数のしきい値を指定します。値は 1 ～ 65535 です。

### アラート頻度の設定

シグニチャのアラート頻度パラメータを設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　設定するシグニチャを指定します。

```
ips-ssp(config-sig)# signatures 9000 0
```

**ステップ 4**　アラート頻度サブモードを開始します。

```
ips-ssp(config-sig-sig)# alert-frequency
```

**ステップ 5**　このシグニチャのアラート頻度を指定します。

**a.** サマリー モードを、たとえば 1 回だけ起動するように設定します。

```
ips-ssp(config-sig-sig-ale)# summary-mode fire-once
ips-ssp(config-sig-sig-ale-fir)# specify-global-summary-threshold yes
ips-ssp(config-sig-sig-ale-fir-yes)# global-summary-threshold 3000
ips-ssp(config-sig-sig-ale-fir-yes)# summary-interval 5000
```

**b.** サマリー キーを指定します。

```
ips-ssp(config-sig-sig-ale-fir-yes)# exit
ips-ssp(config-sig-sig-ale-fir)# summary-key AxBx
```

**c.** 設定を確認できます。

```
ips-ssp(config-sig-sig-ale-fir)# show settings
   fire-once
   -----------------------------------------------
      summary-key: AxBx default: Axxx
      specify-global-summary-threshold
      -----------------------------------------------
         yes
         -----------------------------------------------
            global-summary-threshold: 3000 default: 120
            summary-interval: 5000 default: 15
         -----------------------------------------------
      -----------------------------------------------
   -----------------------------------------------
ips-ssp(config-sig-sig-ale-fir)#
```

**ステップ 6**　アラート頻度サブモードを終了します。

```
ips-ssp(config-sig-sig-ale-fir)# exit
ips-ssp(config-sig-sig-ale)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 7**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# アラート重大度の設定

シグニチャ定義サブモードで **alert-severity** コマンドを使用して、シグニチャの重大度を設定します。

### オプション

次のオプションが適用されます。

- *sig_id*：このシグニチャに割り当てられた一意の数値を示します。この値により、センサーは特定のシグニチャを識別します。値は 1000 ～ 65000 です。
- *subsig_id*：このサブシグニチャに割り当てられた一意の数値を示します。サブシグニチャ ID は、広範なシグニチャをより詳細に識別するために使用します。値は 0 ～ 255 です。
- **alert-severity**：アラートの重大度を指定します。
  - **high**：危険なアラート。
  - **medium**：中レベルのアラート（デフォルト）。
  - **low**：低レベルのアラート。
  - **informational**：情報のアラート。

### アラート重大度の設定

アラート重大度を設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　設定するシグニチャを指定します。

```
ips-ssp(config-sig)# signatures 9000 0
```

**ステップ 4**　アラート重大度を割り当てます。

```
ips-ssp(config-sig-sig)# alert-severity medium
```

**ステップ 5**　設定を確認できます。

```
ips-ssp(config-sig-sig)# show settings
   <protected entry>
   sig-id: 9000
   subsig-id: 0
   -----------------------------------------------
      alert-severity: medium default: medium
      sig-fidelity-rating: 75 <defaulted>
      promisc-delta: 0 <defaulted>
      sig-description
      -----------------------------------------------
         sig-name: Back Door Probe (TCP 12345) <defaulted>
         sig-string-info: SYN to TCP 12345 <defaulted>
         sig-comment: <defaulted>
         alert-traits: 0 <defaulted>
         release: 40 <defaulted>
      -----------------------------------------------
      vulnerable-os: general-os <defaulted>
      engine
      -----------------------------------------------
         atomic-ip
         -----------------------------------------------
            event-action: produce-alert <defaulted>
            fragment-status: any <defaulted>
            specify-l4-protocol
            -----------------------------------------------
--MORE--
```

**ステップ 6**　シグニチャ サブモードを終了します。

```
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 7**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# イベント カウンタの設定

シグニチャ定義サブモードで **event-counter** コマンドを使用して、センサーがイベントをカウントする方法を設定します。たとえば、センサーが、同じシグニチャが同じアドレス セットに対して 5 回起動した場合にだけアラートを送信するように指定できます。

### オプション

次のオプションが適用されます。

- **event-count**：アラートを生成するまでのイベントの発生回数を指定します。有効な範囲は 1 ～ 65535 です。デフォルトは 1 です。
- **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。
  - **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
- **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 カウントがリセットされるまでに必要な時間（秒数）を指定します。デフォルトは 60 です。

### イベント カウンタの設定

イベント カウンタを設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　イベント カウンタを設定するシグニチャを指定します。

```
ips-ssp(config-sig)# signatures 9000 0
```

**ステップ 4**　イベント カウンタ サブモードを開始します。

```
ips-ssp(config-sig-sig)# event-counter
```

**ステップ 5**　アラートが生成されるまでに必要なイベントの発生回数を指定します。

```
ips-ssp(config-sig-sig-eve)# event-count 2
```

**ステップ 6**　このシグニチャの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。

```
ips-ssp(config-sig-sig-eve)# event-count-key AxBx
```

**ステップ 7**　（任意）アラート間隔をイネーブルにします。

```
ips-ssp(config-sig-sig-eve)# specify-alert-interval yes
```

**ステップ 8**　（任意）イベント カウントがリセットされるまでに必要な時間（秒数）を指定します。

```
ips-ssp(config-sig-sig-eve-yes)# alert-interval 30
```

**ステップ 9**　設定を確認できます。

```
ips-ssp(config-sig-sig-eve-yes)# exit
ips-ssp(config-sig-sig-eve)# show settings
   event-counter
   -----------------------------------------------
      event-count: 2 default: 1
      event-count-key: AxBx default: Axxx
      specify-alert-interval
      -----------------------------------------------
         yes
         -----------------------------------------------
            alert-interval: 30 default: 60
         -----------------------------------------------
      -----------------------------------------------
   -----------------------------------------------
ips-ssp(config-sig-sig-eve)#
```

**ステップ 10**　シグニチャ サブモードを終了します。

```
ips-ssp(config-sig-sig-eve)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 11**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# シグニチャ忠実度レーティングの設定

シグニチャ定義サブモードで **sig-fidelity-rating** コマンドを使用して、シグニチャのシグニチャ忠実度レーティングを設定します。

### オプション

次のオプションが適用されます。

- **sig-fidelity-rating**：ターゲットに関する具体的な情報がない場合に、このシグニチャをどの程度忠実に実行するかに関連付ける重みを示します。有効な値は 0 ～ 100 です。

### シグニチャ忠実度レーティングの設定

シグニチャのシグニチャ忠実度レーティングを設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig0
```

**ステップ 3**　設定するシグニチャを指定します。

```
ips-ssp(config-sig)# signatures 12000 0
```

**ステップ 4**　このシグニチャのシグニチャ忠実度レーティングを指定します。

```
ips-ssp(config-sig-sig)# sig-fidelity-rating 50
```

**ステップ 5**　設定を確認できます。

```
ips-ssp(config-sig-sig)# show settings
   <protected entry>
   sig-id: 12000
   subsig-id: 0
   -----------------------------------------------
      alert-severity: low <defaulted>
      sig-fidelity-rating: 50 default: 85
      promisc-delta: 15 <defaulted>
      sig-description
      ----------------------------------------------
         sig-name: Gator Spyware Beacon <defaulted>
         sig-string-info: /download/ User-Agent: Gator <defaulted>
         sig-comment: <defaulted>
         alert-traits: 0 <defaulted>
         release: 71 <defaulted>
      ----------------------------------------------
```

**ステップ 6**　シグニチャ サブモードを終了します。

```
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 7**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# シグニチャのステータスの設定

シグニチャ定義サブモードで **status** コマンドを使用して、特定のシグニチャのステータスを指定します。

### オプション

次のオプションが適用されます。

- **status**：シグニチャの状態がイネーブルか、ディセーブルか、廃棄かどうかを示します。
  - **enabled {true | false}**：シグニチャをイネーブルにします。
  - **retired {true | false}]**：シグニチャを廃棄にします。
  - **obsoletes** *signature_ID*：このシグニチャによって廃止されている他のシグニチャを表示します。

**注意**　シグニチャの有効化および廃棄には、30 分以上の時間がかかる場合があります。

### シグニチャ ステータスの設定

シグニチャのステータスを変更するには、次の作業を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　設定するシグニチャを選択します。

```
ips-ssp(config-sig)# signatures 12000 0
```

**ステップ 4**　このシグニチャのステータスを変更します。

```
ips-ssp(config-sig-sig)# status
ips-ssp(config-sig-sig-sta)# enabled true
```

**ステップ 5**　設定を確認できます。

```
ips-ssp(config-sig-sig-sta)# show settings
   status
   -----------------------------------------------
      enabled: true default: false
      retired: false <defaulted>
   -----------------------------------------------
ips-ssp(config-sig-sig-sta)#
```

**ステップ 6**　シグニチャ サブモードを終了します。

```
ips-ssp(config-sig-sig-sta)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 7**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# シグニチャの脆弱性のある OS の設定

シグニチャ定義サブモードで **vulnerable-os** コマンドを使用して、シグニチャの脆弱性のある OS のリストを設定します。

### オプション

次のオプションが適用されます。

- **general-os**：すべての OS タイプ
- **ios**：各種 Cisco IOS
- **mac-os**：各種 Macintosh OS
- **netware**：Netware
- **other**：その他すべての OS
- **unix**：各種 UNIX
- **aix**：各種 AIX
- **bsd**：各種 BSD
- **hp-ux**：各種 HP-UX
- **irix**：各種 IRIX
- **linux**：各種 Linux
- **solaris**：各種 Solaris
- **windows**：各種 Microsoft Windows
- **windows-nt-2k-xp**：各種 Microsoft NT、2000、および XP
- **win-nt**：特定の種類の Windows NT

### 脆弱性のある OS の設定

シグニチャの脆弱性のある OS を設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　設定するシグニチャを指定します。

```
ips-ssp(config-sig)# signatures 6000 0
```

**ステップ 4**　このシグニチャの脆弱性のある OS を指定します。

```
ips-ssp(config-sig-sig)# vulnerable-os linux|aix
```

**ステップ 5**　設定を確認できます。

```
ips-ssp(config-sig-sig)# show settings
   sig-id: 60000
   subsig-id: 0
   -----------------------------------------------
      alert-severity: medium <defaulted>
      sig-fidelity-rating: 75 <defaulted>
      promisc-delta: 0 <defaulted>
      sig-description
      -----------------------------------------------
         sig-name: My Sig <defaulted>
         sig-string-info: My Sig Info <defaulted>
         sig-comment: Sig Comment <defaulted>
         alert-traits: 0 <defaulted>
         release: custom <defaulted>
      -----------------------------------------------
      vulnerable-os: aix|linux default: general-os
*---> engine
      -----------------------------------------------
      -----------------------------------------------
      event-counter
      -----------------------------------------------
         event-count: 1 <defaulted>
         event-count-key: Axxx <defaulted>
         specify-alert-interval
         -----------------------------------------------
--MORE--
```

**ステップ 6**　シグニチャ サブモードを終了します。

```
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 7**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# シグニチャへのアクションの割り当て

シグニチャ定義サブモードで **event-action** コマンドを使用して、シグニチャが起動したときにセンサーが実行するアクションを設定します。

### オプション

次のオプションが適用されます。

- **event-action :** このシグニチャが起動したときにセンサーが実行するイベント アクションを指定します。
  - **deny-attacker-inline**：（インラインのみ）指定された期間、この攻撃者のアドレスからの現在および将来のパケットを送信しません。
  - **deny-attacker-service-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象ポートがペアになっている現在および将来のパケットを送信しません。
  - **deny-attacker-victim-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象アドレスがペアになっている現在および将来のパケットを送信しません。
  - **deny-connection-inline**：（インラインのみ）この TCP フロー上の現在および将来のパケットを送信しません。
  - **deny-packet-inline**（インラインのみ）現在のパケットを送信しません。
  - **log-attacker-packets**:攻撃者のアドレスが含まれているパケットの 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のペアが含まれているパケットの IP ロギングを開始します。
  - **log-victim-packets**:攻撃対象のアドレスが含まれているパケットの IP ロギングを開始します。
  - **produce-alert**：イベントをアラートとしてイベント ストアに書き込みます。
  - **produce-verbose-alert**：攻撃パケットの符号化されたダンプ（切り詰められる可能性あり）をアラートに含めます。
  - **request-block-connection**：要求を ARC に送信して、この接続をブロックします。
  - **request-block-host**：要求を ARC に送信して、この攻撃者のホストをブロックします。
  - **request-rate-limit**：レート制限要求を ARC に送信して、レート制限を実行します。
  - **request-snmp-trap**：要求をセンサーの通知アプリケーション コンポーネントに送信して、SNMP 通知を実行します。
  - **reset-tcp-connection**：TCP リセットを送信して、TCP フローをハイジャックし、終了します。
  - **modify-packet-inline**：エンド ポイントによるパケットの処理に関するあいまいさを取り除くために、パケット データを変更します。
- **event-action-settings**：**external-rate-limit-type** を設定できます。
  - **none**：設定されているレート制限はありません。
  - **percentage**：トラフィックのパーセンテージによるレート制限を設定します（**external-rate-limit-percentage**）。

### パケットのインライン拒否について

deny-packet-inline がアクションとして設定されているシグニチャの場合、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 オーバーライドの場合、次のアクションが実行される場合があります。

- droppedPacket

- deniedFlow
- tcpOneWayResetSent

パケットのインライン拒否アクションは、アラート内でドロップ パケット アクションとして表示されます。パケットのインライン拒否が TCP 接続に対して発生すると、自動的に接続のインライン拒否アクションにアップグレードされ、アラート内で拒否フローとして表示されます。IPS がパケットを 1 つだけ拒否しても、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 で接続全体を拒否して、再送信が必ず失敗するようにします。

接続のインライン拒否が発生すると同時に、IPS は自動的に TCP 単方向リセットを送信します。これは、アラート内で、送信された TCP 単方向リセットとして表示されます。IPS は、接続を拒否するとき、開いている接続をクライアント（一般に攻撃者）とサーバ（一般に攻撃対象）の両方にそのまま残します。開いている接続が多くなりすぎると、攻撃対象に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（通常はサーバ）側の接続を閉じます。これにより、攻撃対象のリソースが保護されます。さらに、フェールオーバーも阻止されます。それにより、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ようなことはなくなります。IPS は、攻撃者側を開いたままにし、攻撃者側からのすべてのトラフィックを拒否します。

### イベント アクションの設定

シグニチャのイベント アクションおよびイベント アクション設定を設定するには、次の手順を実行します。

**ステップ 1**　管理者権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義モードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig0
ips-ssp(config-sig)#
```

**ステップ 3**　設定するシグニチャを指定します。

```
ips-ssp(config-sig)# signatures 1200 0
```

**ステップ 4**　シグニチャ エンジンを指定します（シグニチャ 1200 の場合はノーマライザ エンジンです）。

```
ips-ssp(config-sig-sig)# engine normalizer
```

**ステップ 5**　イベント アクションを設定します。

```
ips-ssp(config-sig-sig-nor)# event-action produce-alert|request-snmp-trap
```

**（注）**　シグニチャにイベント アクションを設定するたびに、前の設定は上書きされます。たとえば、シグニチャが起動されるたびに必ずアラートが生成されるようにする場合は、そのアクションを他の必要なイベント アクションと一緒に設定する必要があります。| 記号を使用して、**product-alert|deny-packet-inline|request-snmp-trap** のように複数のイベント アクションを追加します。

**ステップ 6**　設定を確認できます。

```
ips-ssp(config-sig-sig-nor)# show settings
   normalizer
   -----------------------------------------------
      event-action: produce-alert|request-snmp-trap default:
produce-alert|deny-packet-inline
```

**ステップ 7**　レート制限のパーセンテージを指定します。

```
ips-ssp(config-sig-sig-nor)# event-action-settings
ips-ssp(config-sig-sig-nor-eve)# external-rate-limit-type percentage
ips-ssp(config-sig-sig-nor-eve-per)# external-rate-limit-percentage 50
```

**ステップ 8**　設定を確認できます。

```
ips-ssp(config-sig-sig-nor-eve-per)# show settings
   percentage
   -----------------------------------------------
      external-rate-limit-percentage: 50 default: 100
   -----------------------------------------------
```

**ステップ 9**　イベント アクション サブモードを終了します。

```
ips-ssp(config-sig-sig-nor-eve-per)# exit
ips-ssp(config-sig-sig-nor-eve)# exit
ips-ssp(config-sig-sig-nor)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 10**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

---

**詳細情報**

イベント アクションの詳細については、「イベント アクション」（P.8-4）を参照してください。

# AIC シグニチャの設定

ここでは、アプリケーション検査および制御（AIC）シグニチャ、およびその設定方法について説明します。次のような構成になっています。



## AIC エンジンについて

AIC は、Web トラフィックを徹底的に分析できます。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、指定されたポート上でトンネルを作成しようとするインスタント メッセージングや gotomypc などのアプリケーションを管理制御できます。P2P およびインスタント メッセージングが HTTP で動作している場合は、それらのアプリケーションの検査チェックおよびポリシー チェックが可能です。AIC は、FTP トラフィックを検査し、実行されているコマンドを制御する方法も提供します。事前に定義されているシグニチャをイネーブルまたはディセーブルにしたり、カスタム シグニチャによってポリシーを作成したりできます。

（注）　AIC エンジンは、HTTP トラフィックが AIC Web ポート上で受信されたときに動作します。トラフィックが Web トラフィックでも、AIC Web ポート上で受信されなければ、Service HTTP エンジンが実行されます。AIC 検査は、AIC Web ポートとして設定されたポートで、検査対象のトラフィックが HTTP トラフィックであれば、任意のポート上に存在できます。

AIC には、次のカテゴリのシグニチャが含まれます。

- HTTP 要求メソッド
  - 定義要求メソッド
  - 認識される要求メソッド
- MIME タイプ
  - 定義コンテンツ タイプ
  - 認識されるコンテンツ タイプ
- 定義 Web トラフィック ポリシー

  1 つの事前に定義されたシグニチャ 12674 があります。これは、非準拠の HTTP トラフィックが検出されたときに実行するアクションを指定しています。パラメータ Alarm on Non HTTP Traffic は、このシグニチャをイネーブルにします。デフォルトでは、このシグニチャはイネーブルになっています。
- 転送符号化
  - 各メソッドへのアクションの関連付け
  - センサーで認識されるメソッドの一覧表示
  - チャンク符号化エラーが検出されたときにどのアクションを実行する必要があるかの指定
- FTP コマンド
  - FTP コマンドへのアクションの関連付け。

**詳細情報**

- これらのシグニチャのシグニチャ ID および説明のリストについては、「AIC 要求メソッド シグニチャ」（P.7-21）、「AIC MIME 定義コンテンツ タイプ シグニチャ」（P.7-22）、「AIC 転送符号化シグニチャ」（P.7-25）、および「AIC FTP コマンド シグニチャ」（P.7-26）を参照してください。
- カスタム MIME シグニチャの作成手順については、「AIC シグニチャの作成」（P.7-27）を参照してください。

## 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 ポリシーの適用は、独自のセンサー機能です。AIC ポリシーの適用は、悪用、脆弱性、および異常を検査する従来の IPS テクノロジーをベースにしたものではなく、HTTP サービス ポリシーと FTP サービス ポリシーを適用するように設計されています。このポリシー適用に必要な検査動作は、従来の IPS 検査動作と比べて負荷が非常に高くなります。この機能の使用には、パフォーマンスの大幅な低下が伴います。AIC がイネーブルの場合、センサーの全体的な帯域幅キャパシティが減少します。

AIC ポリシーの適用は、IPS デフォルト設定ではディセーブルになっています。AIC ポリシーの適用を有効にする場合は、必要なポリシーだけを慎重に選択し、必要のないポリシーは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が最大容量に近くなっている場合は、センサーがオーバーサブスクライブされる可能性があるので、この機能の使用は推奨されません。このタイプのポリシー適用を処理する場合は、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。

# アプリケーション ポリシーの設定

シグニチャ定義サブモードで **application-policy** コマンドを使用して、Web AIC 機能をイネーブルにします。レイヤ 4 からレイヤ 7 パケットの検査を提供して Web サービスおよび FTP サービスに関連した悪意のある攻撃を阻止するようにセンサーを設定できます。

### オプション

次のオプションが適用されます。

- **ftp-enable {true | false}**：FTP サービスの保護をイネーブルにします。センサーで FTP トラフィックを検査する必要がある場合は、true に設定します。デフォルトは false です。
- **http-policy**：HTTP トラフィックの検査をイネーブルにします。
  - **aic-web-ports**：AIC トラフィックを検索するポートを表す変数を指定します。有効な範囲は 0 ～ 65535 です。0 ～ 65535 の範囲内に収まる、カンマで区切られた整数範囲のリストです（a-b[,c-d]）。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。デフォルトは、80-80,3128-3128,8000-8000,8010-8010,8080-8080,8888-8888,24326-24326 です。
  - **http-enable {true | false}**：Web サービスの保護をイネーブルにします。RFC に準拠するために、センサーで HTTP トラフィックを検査する必要がある場合は、true に設定します。デフォルトは false です。
  - **max-outstanding-http-requests-per-connection**：接続あたりの最大許容 HTTP 要求数を指定します。有効な値は 1 ～ 16 です。デフォルトは 10 です。

### アプリケーション ポリシーの設定

アプリケーション ポリシーを設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　アプリケーション ポリシー サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
ips-ssp(config-sig)# application-policy
```

**ステップ 3**　FTP トラフィックの検査をイネーブルにします。

```
ips-ssp(config-sig-app)# ftp-enable true
```

**ステップ 4**　HTTP アプリケーション ポリシーを設定します。

**a.** HTTP アプリケーション ポリシー サブモードを開始します。

```
ips-ssp(config-sig-app)# http-policy
```

**b.** HTTP アプリケーション ポリシーの適用をイネーブルにします。

```
ips-ssp(config-sig-app-htt)# http-enable true
```

**c.** サーバからの応答を受信せずに未処理状態でいることが可能な、接続あたりの未処理 HTTP 要求の数を指定します。

```
ips-ssp(config-sig-app-htt)# max-outstanding-http-requests-per-connection 5
```

**d.** AIC ポートを編集します。

```
ips-ssp(config-sig-app-htt)# aic-web-ports 80-80,3128-3128
```

**ステップ 5**　設定値を確認します。

```
ips-ssp(config-sig-app-htt)# exit
ips-ssp(config-sig-app)# show settings
   application-policy
   -----------------------------------------------
      http-policy
      -----------------------------------------------
         http-enable: true default: false
         max-outstanding-http-requests-per-connection: 5 default: 10
         aic-web-ports: 80-80,3128-3128 default: 80-80,3128-3128,8000-8000,8010-
8010,8080-8080,8888-8888,24326-24326
      -----------------------------------------------
      ftp-enable: true default: false
   -----------------------------------------------
ips-ssp(config-sig-app)#
```

**ステップ 6**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-app)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 7**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

## AIC 要求メソッド シグニチャ

HTTP 要求メソッドには、2 つのカテゴリのシグニチャがあります。

- 定義要求メソッド：アクションが要求メソッドに関連付けられます。シグニチャの拡張と変更ができます（Define Request Method）。
- 認識される要求メソッド：センサーによって認識されるメソッドの一覧を表示します（Recognized Request Methods）。

表 7-1 に、事前に定義されている定義要求メソッド シグニチャを示します。必要な事前定義メソッドを持つシグニチャをイネーブルにします。

**表 7-1　要求メソッド シグニチャ**

| シグニチャ ID | 定義要求メソッド |
|---|---|
| 12676 | 要求メソッドは認識されない |
| 12677 | 定義要求メソッド PUT |
| 12678 | 定義要求メソッド CONNECT |
| 12679 | 定義要求メソッド DELETE |
| 12680 | 定義要求メソッド GET |
| 12681 | 定義要求メソッド HEAD |
| 12682 | 定義要求メソッド OPTIONS |
| 12683 | 定義要求メソッド POST |
| 12685 | 定義要求メソッド TRACE |
| 12695 | 定義要求メソッド INDEX |
| 12696 | 定義要求メソッド MOVE |
| 12697 | 定義要求メソッド MKDIR |

表 7-1　要求メソッド シグニチャ （続き）

| シグニチャ ID | 定義要求メソッド |
|---|---|
| 12698 | 定義要求メソッド COPY |
| 12699 | 定義要求メソッド EDIT |
| 12700 | 定義要求メソッド UNEDIT |
| 12701 | 定義要求メソッド SAVE |
| 12702 | 定義要求メソッド LOCK |
| 12703 | 定義要求メソッド UNLOCK |
| 12704 | 定義要求メソッド REVLABEL |
| 12705 | 定義要求メソッド REVLOG |
| 12706 | 定義要求メソッド REVADD |
| 12707 | 定義要求メソッド REVNUM |
| 12708 | 定義要求メソッド SETATTRIBUTE |
| 12709 | 定義要求メソッド GETATTRIBUTENAME |
| 12710 | 定義要求メソッド GETPROPERTIES |
| 12711 | 定義要求メソッド STARTENV |
| 12712 | 定義要求メソッド STOPREV |

### 詳細情報

シグニチャをイネーブルにする手順については、「シグニチャのステータスの設定」（P.7-13）を参照してください。

## AIC MIME 定義コンテンツ タイプ シグニチャ

MIME タイプに関連したポリシーは 2 つあります。

- 定義コンテンツ タイプ : 次の場合に対して特定のアクションを関連付けます（Define Content Type）。
  - image/jpeg など、特定の MIME タイプを拒否
  - メッセージ サイズ違反
  - ヘッダーと本体に記述されている MIME タイプが不一致
- 認識されるコンテンツ タイプ（Recognized Content Type）

表 7-2 に、事前に定義されている定義コンテンツ タイプ シグニチャを示します。必要な事前定義コンテンツ タイプを持つシグニチャをイネーブルにします。また、カスタム定義コンテンツ タイプ シグニチャを作成することもできます。

表 7-2　定義コンテンツ タイプ シグニチャ

| シグニチャ ID | シグニチャの説明 |
|---|---|
| 12621 | コンテンツ タイプ image/gif のメッセージ長が無効です。 |
| 12622 2 | コンテンツ タイプ image/png の検証に失敗しました。 |
| 12623 0<br>12623 1<br>12623 2 | コンテンツ タイプ image/tiff のヘッダー チェック。<br>コンテンツ タイプ image/tiff のメッセージ長が無効です。<br>コンテンツ タイプ image/tiff の検証に失敗しました。 |

表 7-2　定義コンテンツ タイプ シグニチャ （続き）

| シグニチャ ID | シグニチャの説明 |
|---|---|
| 12624 0 | コンテンツ タイプ image/x-3ds のヘッダー チェック。 |
| 12624 1 | コンテンツ タイプ image/x-3ds のメッセージ長が無効です。 |
| 12624 2 | コンテンツ タイプ image/x-3ds の検証に失敗しました。 |
| 12626 0 | コンテンツ タイプ image/x-portable-bitmap のヘッダー チェック。 |
| 12626 1 | コンテンツ タイプ image/x-portable-bitmap のメッセージ長が無効です。 |
| 12626 2 | コンテンツ タイプ image/x-portable-bitmap の検証に失敗しました。 |
| 12627 0 | コンテンツ タイプ image/x-portable-graymap のヘッダー チェック。 |
| 12627 1 | コンテンツ タイプ image/x-portable-graymap のメッセージ長が無効です。 |
| 12627 2 | コンテンツ タイプ image/x-portable-graymap の検証に失敗しました。 |
| 12628 0 | コンテンツ タイプ image/jpeg のヘッダー チェック。 |
| 12628 1 | コンテンツ タイプ image/jpeg のメッセージ長が無効です。 |
| 12628 2 | コンテンツ タイプ image/jpeg の検証に失敗しました。 |
| 12629 0 | コンテンツ タイプ image/cgf のヘッダー チェック。 |
| 12629 1 | コンテンツ タイプ image/cgf のメッセージ長が無効です。 |
| 12631 0 | コンテンツ タイプ image/x-xpm のヘッダー チェック。 |
| 12631 1 | コンテンツ タイプ image/x-xpm のメッセージ長が無効です。 |
| 12633 0 | コンテンツ タイプ audio/midi のヘッダー チェック。 |
| 12633 1 | コンテンツ タイプ audio/midi のメッセージ長が無効です。 |
| 12633 2 | コンテンツ タイプ audio/midi の検証に失敗しました。 |
| 12634 0 | コンテンツ タイプ audio/basic のヘッダー チェック。 |
| 12634 1 | コンテンツ タイプ audio/basic のメッセージ長が無効です。 |
| 12634 2 | コンテンツ タイプ audio/basic の検証に失敗しました。 |
| 12635 0 | コンテンツ タイプ audio/mpeg のヘッダー チェック。 |
| 12635 1 | コンテンツ タイプ audio/mpeg のメッセージ長が無効です。 |
| 12635 2 | コンテンツ タイプ audio/mpeg の検証に失敗しました。 |
| 12636 0 | コンテンツ タイプ audio/x-adpcm のヘッダー チェック。 |
| 12636 1 | コンテンツ タイプ audio/x-adpcm のメッセージ長が無効です。 |
| 12636 2 | コンテンツ タイプ audio/x-adpcm の検証に失敗しました。 |
| 12637 0 | コンテンツ タイプ audio/x-aiff のヘッダー チェック。 |
| 12637 1 | コンテンツ タイプ audio/x-aiff のメッセージ長が無効です。 |
| 12637 2 | コンテンツ タイプ audio/x-aiff の検証に失敗しました。 |
| 12638 0 | コンテンツ タイプ audio/x-ogg のヘッダー チェック。 |
| 12638 1 | コンテンツ タイプ audio/x-ogg のメッセージ長が無効です。 |
| 12638 2 | コンテンツ タイプ audio/x-ogg の検証に失敗しました。 |
| 12639 0 | コンテンツ タイプ audio/x-wav のヘッダー チェック。 |
| 12639 1 | コンテンツ タイプ audio/x-wav のメッセージ長が無効です。 |
| 12639 2 | コンテンツ タイプ audio/x-wav の検証に失敗しました。 |
| 12641 0 | コンテンツ タイプ text/html のヘッダー チェック。 |
| 12641 1 | コンテンツ タイプ text/html のメッセージ長が無効です。 |
| 12641 2 | コンテンツ タイプ text/html の検証に失敗しました。 |
| 12642 0 | コンテンツ タイプ text/css のヘッダー チェック。 |
| 12642 1 | コンテンツ タイプ text/css のメッセージ長が無効です。 |
| 12643 0 | コンテンツ タイプ text/plain のヘッダー チェック。 |
| 12643 1 | コンテンツ タイプ text/plain のメッセージ長が無効です。 |

表 7-2　定義コンテンツ タイプ シグニチャ （続き）

| シグニチャ ID | シグニチャの説明 |
|---|---|
| 12644 0 | コンテンツ タイプ text/richtext のヘッダー チェック。 |
| 12644 1 | コンテンツ タイプ text/richtext のメッセージ長が無効です。 |
| 12645 0 | コンテンツ タイプ text/sgml のヘッダー チェック。 |
| 12645 1 | コンテンツ タイプ text/sgml のメッセージ長が無効です。 |
| 12645 2 | コンテンツ タイプ text/sgml の検証に失敗しました。 |
| 12646 0 | コンテンツ タイプ text/xml のヘッダー チェック。 |
| 12646 1 | コンテンツ タイプ text/xml のメッセージ長が無効です。 |
| 12646 2 | コンテンツ タイプ text/xml の検証に失敗しました。 |
| 12648 0 | コンテンツ タイプ video/flc のヘッダー チェック。 |
| 12648 1 | コンテンツ タイプ video/flc のメッセージ長が無効です。 |
| 12648 2 | コンテンツ タイプ video/flc の検証に失敗しました。 |
| 12649 0 | コンテンツ タイプ video/mpeg のヘッダー チェック。 |
| 12649 1 | コンテンツ タイプ video/mpeg のメッセージ長が無効です。 |
| 12649 2 | コンテンツ タイプ video/mpeg の検証に失敗しました。 |
| 12650 0 | コンテンツ タイプ text/xmcd のヘッダー チェック。 |
| 12650 1 | コンテンツ タイプ text/xmcd のメッセージ長が無効です。 |
| 12651 0 | コンテンツ タイプ video/quicktime のヘッダー チェック。 |
| 12651 1 | コンテンツ タイプ video/quicktime のメッセージ長が無効です。 |
| 12651 2 | コンテンツ タイプ video/quicktime の検証に失敗しました。 |
| 12652 0 | コンテンツ タイプ video/sgi のヘッダー チェック。 |
| 12652 1 | コンテンツ タイプ video/sgi の検証に失敗しました。 |
| 12653 0 | コンテンツ タイプ video/x-avi のヘッダー チェック。 |
| 12653 1 | コンテンツ タイプ video/x-avi のメッセージ長が無効です。 |
| 12654 0 | コンテンツ タイプ video/x-fli のヘッダー チェック。 |
| 12654 1 | コンテンツ タイプ video/x-fli のメッセージ長が無効です。 |
| 12654 2 | コンテンツ タイプ video/x-fli の検証に失敗しました。 |
| 12655 0 | コンテンツ タイプ video/x-mng のヘッダー チェック。 |
| 12655 1 | コンテンツ タイプ video/x-mng のメッセージ長が無効です。 |
| 12655 2 | コンテンツ タイプ video/x-mng の検証に失敗しました。 |
| 12656 0 | コンテンツ タイプ application/x-msvideo のヘッダー チェック。 |
| 12656 1 | コンテンツ タイプ application/x-msvideo のメッセージ長が無効です。 |
| 12656 2 | コンテンツ タイプ application/x-msvideo の検証に失敗しました。 |
| 12658 0 | コンテンツ タイプ application/ms-word のヘッダー チェック。 |
| 12658 1 | コンテンツ タイプ application/ms-word のメッセージ長が無効です。 |
| 12659 0 | コンテンツ タイプ application/octet-stream のヘッダー チェック。 |
| 12659 1 | コンテンツ タイプ application/octet-stream のメッセージ長が無効です。 |
| 12660 0 | コンテンツ タイプ application/postscript のヘッダー チェック。 |
| 12660 1 | コンテンツ タイプ application/postscript のメッセージ長が無効です。 |
| 12660 2 | コンテンツ タイプ application/postscript の検証に失敗しました。 |
| 12661 0 | コンテンツ タイプ application/vnd.ms-excel のヘッダー チェック。 |
| 12661 1 | コンテンツ タイプ application/vnd.ms-excel のメッセージ長が無効です。 |
| 12662 0 | コンテンツ タイプ application/vnd.ms-powerpoint のヘッダー チェック。 |
| 12662 1 | コンテンツ タイプ application/vnd.ms-powerpoint のメッセージ長が無効です。 |

表 7-2　定義コンテンツ タイプ シグニチャ （続き）

| シグニチャ ID | シグニチャの説明 |
|---|---|
| 12663 0<br>12663 1<br>12663 2 | コンテンツ タイプ application/zip のヘッダー チェック。<br>コンテンツ タイプ application/zip のメッセージ長が無効です。<br>コンテンツ タイプ application/zip の検証に失敗しました。 |
| 12664 0<br>12664 1<br>12664 2 | コンテンツ タイプ application/x-gzip のヘッダー チェック。<br>コンテンツ タイプ application/x-gzip のメッセージ長が無効です。<br>コンテンツ タイプ application/x-gzip の検証に失敗しました。 |
| 12665 0<br>12665 1 | コンテンツ タイプ application/x-java-archive のヘッダー チェック。<br>コンテンツ タイプ application/x-java-archive のメッセージ長が無効です。 |
| 12666 0<br>12666 1 | コンテンツ タイプ application/x-java-vm のヘッダー チェック。<br>コンテンツ タイプ application/x-java-vm のメッセージ長が無効です。 |
| 12667 0<br>12667 1<br>12667 2 | コンテンツ タイプ application/pdf のヘッダー チェック。<br>コンテンツ タイプ application/pdf のメッセージ長が無効です。<br>コンテンツ タイプ application/pdf の検証に失敗しました。 |
| 12668 0<br>12668 1 | コンテンツ タイプ unknown のヘッダー チェック。<br>コンテンツ タイプ unknown のメッセージ長が無効です。 |
| 12669 0<br>12669 1 | コンテンツ タイプ image/x-bitmap のヘッダー チェック。<br>コンテンツ タイプ image/x-bitmap のメッセージ長が無効です。 |
| 12673 0 | 認識されるコンテンツ タイプ |

### 詳細情報

- シグニチャをイネーブルにする手順については、「シグニチャのステータスの設定」（P.7-13）を参照してください。
- AIC シグニチャの作成手順については、「AIC シグニチャの作成」（P.7-27）を参照してください。

## AIC 転送符号化シグニチャ

転送符号化に関連したポリシーは 3 つあります。

- 各メソッドへのアクションの関連付け（Define Transfer Encoding）
- センサーで認識されるメソッドの一覧表示（Recognized Transfer Encodings）
- チャンク符号化エラーが検出されたときにどのアクションを実行する必要があるかの指定（Chunked Transfer Encoding Error）

表 7-3 に、事前に定義されている転送符号化シグニチャを示します。必要な事前定義の転送符号化メソッドを持つシグニチャをイネーブルにします。

表 7-3　転送符号化シグニチャ

| シグニチャ ID | 転送符号化メソッド |
|---|---|
| 12686 | 認識される転送符号化 |
| 12687 | 定義転送符号化 Deflate |
| 12688 | 定義転送符号化 Identity |
| 12689 | 定義転送符号化 Compress |
| 12690 | 定義転送符号化 GZIP |

表 7-3　転送符号化シグニチャ （続き）

| シグニチャ ID | 転送符号化メソッド |
|---|---|
| 12693 | 定義転送符号化 Chunked |
| 12694 | チャンク転送符号化エラー |

### 詳細情報

シグニチャをイネーブルにする手順については、「シグニチャのステータスの設定」（P.7-13）を参照してください。

## AIC FTP コマンド シグニチャ

表 7-4 に、事前に定義されている FTP コマンド シグニチャを示します。必要な事前定義 FTP コマンドを持つシグニチャをイネーブルにします。

表 7-4　FTP コマンド シグニチャ

| シグニチャ ID | FTP コマンド |
|---|---|
| 12900 | 認識されない FTP コマンド |
| 12901 | 定義 FTP コマンド abor |
| 12902 | 定義 FTP コマンド acct |
| 12903 | 定義 FTP コマンド allo |
| 12904 | 定義 FTP コマンド appe |
| 12905 | 定義 FTP コマンド cdup |
| 12906 | 定義 FTP コマンド cwd |
| 12907 | 定義 FTP コマンド dele |
| 12908 | 定義 FTP コマンド help |
| 12909 | 定義 FTP コマンド list |
| 12910 | 定義 FTP コマンド mkd |
| 12911 | 定義 FTP コマンド mode |
| 12912 | 定義 FTP コマンド nlst |
| 12913 | 定義 FTP コマンド noop |
| 12914 | 定義 FTP コマンド pass |
| 12915 | 定義 FTP コマンド pasv |
| 12916 | 定義 FTP コマンド port |
| 12917 | 定義 FTP コマンド pwd |
| 12918 | 定義 FTP コマンド quit |
| 12919 | 定義 FTP コマンド rein |
| 12920 | 定義 FTP コマンド rest |
| 12921 | 定義 FTP コマンド retr |
| 12922 | 定義 FTP コマンド rmd |
| 12923 | 定義 FTP コマンド rnfr |
| 12924 | 定義 FTP コマンド rnto |

表 7-4　FTP コマンド シグニチャ （続き）

| シグニチャ ID | FTP コマンド |
|---|---|
| 12925 | 定義 FTP コマンド site |
| 12926 | 定義 FTP コマンド smnt |
| 12927 | 定義 FTP コマンド stat |
| 12928 | 定義 FTP コマンド stor |
| 12929 | 定義 FTP コマンド stou |
| 12930 | 定義 FTP コマンド stru |
| 12931 | 定義 FTP コマンド syst |
| 12932 | 定義 FTP コマンド type |
| 12933 | 定義 FTP コマンド user |

### 詳細情報

シグニチャをイネーブルにする手順については、「シグニチャのステータスの設定」（P.7-13）を参照してください。

## AIC シグニチャの作成

次に、AIC エンジンに基づいて MIME タイプ シグニチャを作成する例を示します。

### オプション

次のオプションが適用されます。

- **event-action**：アラートがトリガーされたときに実行するアクションを指定します。
  - **deny-attacker-inline**：（インラインのみ）指定された期間、この攻撃者のアドレスからの現在および将来のパケットを送信しません。
  - **deny-attacker-service-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象ポートがペアになっている現在および将来のパケットを送信しません。
  - **deny-attacker-victim-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象アドレスがペアになっている現在および将来のパケットを送信しません。
  - **deny-connection-inline**：（インラインのみ）この TCP フロー上の現在および将来のパケットを送信しません。
  - **deny-packet-inline**（インラインのみ）現在のパケットを送信しません。
  - **log-attacker-packets**：攻撃者のアドレスが含まれているパケットの 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のペアが含まれているパケットの IP ロギングを開始します。
  - **log-victim-packets**：攻撃対象のアドレスが含まれているパケットの IP ロギングを開始します。
  - **produce-alert**：イベントをアラートとしてイベント ストアに書き込みます。
  - **produce-verbose-alert**：攻撃パケットの符号化されたダンプ（切り詰められる可能性あり）をアラートに含めます。
  - **request-block-connection**：要求を ARC に送信して、この接続をブロックします。
  - **request-block-host**：要求を ARC に送信して、この攻撃者のホストをブロックします。
  - **request-rate-limit**：レート制限要求を ARC に送信して、レート制限を実行します。

  - **request-snmp-trap**：要求をセンサーの通知アプリケーション コンポーネントに送信して、SNMP 通知を実行します。
  - **reset-tcp-connection**：TCP リセットを送信して、TCP フローをハイジャックし、終了します。
  - **modify-packet-inline**：エンド ポイントによるパケットの処理に関するあいまいさを取り除くために、パケット データを変更します。
- **no**：エントリまたは選択の設定を削除します。
- **signature-type**：必要なシグニチャのタイプを指定します。
  - **content-types**：コンテンツタイプ
  - **define-web-traffic-policy**：Web トラフィック ポリシーの定義
  - **max-outstanding-requests-overrun**：大量の未処理 HTTP 要求が発生していないかの検査
  - **msg-body-pattern**：メッセージ本体のパターン
  - **request-methods**：要求メソッドを処理するシグニチャ タイプ
  - **transfer-encodings**：転送符号化を処理するシグニチャ タイプ

### MIME タイプ ポリシーの定義

MIME タイプ ポリシー シグニチャを定義するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　アプリケーション ポリシー適用サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
ips-ssp(config-sig)# signatures 60001 0
ips-ssp(config-sig-sig)# engine application-policy-enforcement-http
```

**ステップ 3**　イベント アクションを指定します。

```
ips-ssp(config-sig-sig-app)# event-action produce-alert|log-pair-packets
```

**ステップ 4**　シグニチャ タイプを定義します。

```
ips-ssp(config-sig-sig-app)# signature-type content-type define-content-type
```

**ステップ 5**　コンテンツ タイプを定義します。

```
ips-ssp(config-sig-sig-app-def)# name MyContent
```

**ステップ 6**　設定値を確認します。

```
ips-ssp(config-sig-sig-app-def)# show settings
-> define-content-type
   -----------------------------------------------
      name: MyContent
*---> content-type-details
      -----------------------------------------------
      -----------------------------------------------
   -----------------------------------------------
ips-ssp(config-sig-sig-app-def)#
```

**ステップ 7**　シグニチャ サブモードを終了します。

```
ips-ssp(config-sig-sig-app-def)# exit
ips-ssp(config-sig-sig-app)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 8**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# IP フラグメント再構成の設定

ここでは、IP フラグメント再構成について説明し、IP フラグメント再構成シグニチャの一覧とその設定可能なパラメータを示します。また、それらのパラメータを設定する方法、および IP フラグメント再構成の方式を設定する方法についても説明します。次のような構成になっています。

- 「IP フラグメント再構成について」（P.7-29）
- 「IP フラグメント再構成シグニチャと設定可能なパラメータ」（P.7-29）
- 「IP フラグメント再構成パラメータの設定」（P.7-31）
- 「IP フラグメント再構成の方式の設定」（P.7-32）

## IP フラグメント再構成について

センサーは、複数のパケットにわたってフラグメント化されたデータグラムを再構成するように設定できます。このとき、再構成するデータグラム フラグメントの数と、データグラムについてさらにフラグメントが届くのを待つ時間を判断するために使用する境界値を指定できます。これは、センサーがフレーム送信を受信できなかったことや、無作為にフラグメント化されたデータグラムを生成する攻撃が仕掛けられていることが原因で再構成が不十分なデータグラムに対し、センサーのリソースをすべて割り当ててしまわないようにするためのものです。

（注）　IP フラグメント再構成はシグニチャ単位で設定します。

## IP フラグメント再構成シグニチャと設定可能なパラメータ

表 7-5 に、IP フラグメント再構成のために設定できる IP フラグメント再構成シグニチャとその設定可能なパラメータを示します。IP フラグメント再構成シグニチャは、ノーマライザ エンジンの一部です。

**表 7-5　IP フラグメント再構成シグニチャ**

| シグニチャ ID と名前 | 説明 | パラメータ：デフォルト値（範囲） | デフォルト アクション |
|---|---|---|---|
| 1200 IP Fragmentation Buffer Full | システム内のフラグメントの合計数が Max Fragments で設定されたしきい値を超えると起動します。 | Max Fragments：10000（0 ～ 42000） | Deny Packet Inline Produce Alert[1] |
| 1201 Fragment Overlap | データグラムに対してキューイングされているフラグメントの間で重複が発生すると起動します。 | なし[2] | |

表 7-5　IP フラグメント再構成シグニチャ （続き）

| シグニチャ ID と名前 | 説明 | パラメータ：デフォルト値（範囲） | デフォルト アクション |
|---|---|---|---|
| 1202 Datagram Too Long | フラグメント データ（オフセットとサイズ）が Max Datagram Size で設定されたしきい値を超えると起動します。 | Max Datagram Size：65536（2000 ～ 65536） | Deny Packet Inline Produce Alert[3] |
| 1203 Fragment Overwrite | データグラムに対してキューイングされているフラグメントの間で重複が発生し、その重複しているデータ間に差異がある場合に起動します。[4] | なし | Deny Packet Inline Produce Alert[5] |
| 1204 No Initial Fragment | データグラムが不完全で、先頭フラグメントが失われている場合に起動します。 | なし | Deny Packet Inline Produce Alert[6] |
| 1205 Too Many Datagrams | システム内の部分データグラムの合計数が Max Partial Datagrams で設定されたしきい値を超えると起動します。 | Max Partial Datagrams：1000（0 ～ 10000） | Deny Packet Inline Produce Alert[7] |
| 1206 Fragment Too Small | 1 つのデータグラム内に Min Fragment Size より小さいサイズのフラグメントが Max Small Frags よりも多く存在すると起動します。[8] | Max Small Frags：2（8 ～ 1500）<br>Min Fragment Size：400（1 ～ 8） | Deny Packet Inline Produce Alert[9] |
| 1207 Too Many Fragments | 1 つのデータグラム内に Max Fragments per Datagram よりも多くのフラグメントが存在すると起動します。 | Max Fragments per Datagram：170（0-8192） | Deny Packet Inline Produce Alert[10] |
| 1208 Incomplete Datagram | データグラムのフラグメントが 1 つでも Fragment Reassembly Timeout の時間内に到着できなかった場合に起動します。[11] | Fragment Reassembly Timeout：60（0 ～ 360） | Deny Packet Inline Produce Alert[12] |
| 1220 Jolt2 Fragment Reassembly DoS attack | 複数のフラグメントが受信され、それらすべてが IP データグラムの最終フラグメントであると主張している場合に起動します。 | Max Last Fragments：4（1 ～ 50） | Deny Packet Inline Produce Alert[13] |
| 1225 Fragment Flags Invalid | 不正な組み合わせのフラグメント フラグが検出されると起動します。 | なし[14] | |

1. Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。このシグニチャをディセーブルにしても、デフォルト値が使用されます。それによりパケットがドロップされたり（インライン モード）、分析されなかったり（無差別モード）しますが、アラートは送信されません。
2. データグラムが完全な複製の場合、このシグニチャは起動しません。完全な複製は、インライン モードでは設定に関係なくドロップされます。Modify Packet Inline は、エンドポイントがデータグラムを処理する方法についてあいまいさが残らないように、重複しているデータをそのうちの 1 つだけ残してすべて削除します。Deny Connection Inline は、このシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。
3. Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。データグラムが Max Datagram Size より大きい場合、設定されたアクションに関係なく、データグラムは IPS で処理されません。
4. これは、きわめて異常なイベントです。
5. Modify Packet Inline は、エンドポイントがデータグラムを処理する方法についてあいまいさが残らないように、重複しているデータをそのうちの 1 つだけ残してすべて削除します。Deny Connection Inline は、このシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。
6. IPS は、設定に関係なく、先頭フラグメントのないデータグラムは検査しません。Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。

7. Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。
8. このシグニチャがオンで、小さいフラグメントの数が超過している場合、IPS はデータグラムを検査しません。
9. Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。
10. Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。
11. データグラムのパケットが到着するとタイマーが始動します。
12. Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。
13. Modify Packet Inline と Deny Connection Inline はこのシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。
14. Modify Packet Inline は、それらのフラグを有効な組み合わせに修正します。Deny Connection Inline は、このシグニチャに影響を与えません。Deny Packet Inline は、このデータグラムのパケットおよびすべての関連するフラグメントをドロップします。

**詳細情報**

ノーマライザ エンジンの詳細については、「ノーマライザ エンジン」（P.B-33）を参照してください。

## IP フラグメント再構成パラメータの設定

特定のシグニチャの IP フラグメント再構成パラメータを設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　IP フラグメント再構成シグニチャ ID とサブシグニチャ ID を指定します。

```
ips-ssp(config-sig)# signatures 1200 0
```

**ステップ 4**　エンジンを指定します。

```
ips-ssp(config-sig-sig)# engine normalizer
```

**ステップ 5**　デフォルト シグニチャの編集サブモードを開始します。

```
ips-ssp(config-sig-sig-nor)# edit-default-sigs-only default-signatures-only
```

**ステップ 6**　シグニチャ 1200 の任意の IP フラグメント再構成パラメータについて（必要に応じて）デフォルト設定をイネーブルにし、変更します。たとえば、最大フラグメント数を指定します。

```
ips-ssp(config-sig-sig-nor-def)# specify-max-fragments yes
ips-ssp(config-sig-sig-nor-def-yes)# max-fragments 20000
```

**ステップ 7**　設定を確認できます。

```
ips-ssp(config-sig-sig-nor-def-yes)# show settings
   yes
   -----------------------------------------------
      max-fragments: 20000 default: 10000
   -----------------------------------------------
ips-ssp(config-sig-sig-nor-def-yes)#
```

**ステップ 8**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-nor-def-yes)# exit
ips-ssp(config-sig-sig-nor-def)# exit
ips-ssp(config-sig-sig-nor)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 9**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

## IP フラグメント再構成の方式の設定

シグニチャ定義サブモードで **fragment-reassembly** コマンドを使用して、センサーがフラグメントの再構成に使用する方式を設定します。このオプションは、センサーが無差別モードで動作している場合に設定できます。センサーがインライン モードで動作している場合、方式は NT のみになります。

### オプション

次のオプションが適用されます。

- **ip-reassemble-mode**：センサーがフラグメントの再構成に使用する方式をオペレーティング システムに基づいて示します。
  - **nt**：Windows システム。
  - **solaris**：Solaris システム。
  - **linux**：GNU/Linux システム。
  - **bsd**：BSD UNIX システム。

    デフォルトは nt です。

### IP フラグメント再構成の方式の設定

IP フラグメント再構成の方式を設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　フラグメント再構成サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
ips-ssp(config-sig)# fragment-reassembly
```

**ステップ 3**　センサーが IP フラグメントの再構成に使用するオペレーティング システムを設定します。

```
ips-ssp(config-sig-fra)# ip-reassemble-mode linux
```

**ステップ 4**　設定を確認します。

```
ips-ssp(config-sig-fra)# show settings
   fragment-reassembly
   -----------------------------------------------
      ip-reassemble-mode: linux default: nt
   -----------------------------------------------
ips-ssp(config-sig-fra)#
```

**ステップ 5**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-fra)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 6**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# TCP ストリーム再構成の設定

ここでは、TCP ストリーム再構成について説明し、TCP ストリーム再構成シグニチャの一覧とその設定可能なパラメータを示します。また、TCP ストリーム シグニチャを設定する方法、および TCP ストリーム再構成のモードを設定する方法についても説明します。次のような構成になっています。

- 「TCP ストリーム再構成について」（P.7-33）
- 「TCP ストリーム再構成シグニチャと設定可能なパラメータ」（P.7-34）
- 「TCP ストリーム再構成シグニチャの設定」（P.7-39）
- 「TCP ストリーム再構成のモードの設定」（P.7-40）

## TCP ストリーム再構成について

センサーは、完了した 3 ウェイ ハンドシェイクによって確立された TCP セッションだけをモニタするように設定できます。また、ハンドシェイクの完了まで待つ時間の最大値と、パケットがない場合に接続をモニタし続ける時間も設定できます。これは、有効な TCP セッションが確立していないときにセンサーがアラートを生成しないようにするためのものです。センサーに対する攻撃には、単純に攻撃を繰り返すだけでセンサーにアラートを生成させようとするものがあります。TCP セッションの再構成機能は、センサーに対するこのような攻撃の緩和に役立ちます。

TCP ストリーム再構成パラメータはシグニチャ単位で設定します。TCP ストリーム再構成のモードを設定できます。

## TCP ストリーム再構成シグニチャと設定可能なパラメータ

表 7-6 に、TCP ストリーム再構成のために設定できる TCP ストリーム再構成シグニチャとその設定可能なパラメータを示します。TCP ストリーム再構成シグニチャは、ノーマライザ エンジンの一部です。

**表 7-6　TCP ストリーム再構成シグニチャ**

| シグニチャ ID と名前 | 説明 | パラメータ：デフォルト値（範囲） | デフォルト アクション |
|---|---|---|---|
| 1300 TCP Segment Overwrite[1] | 重複する TCP セグメント（再送信など）により、このセッションですでに検出されているデータとは異なるデータが送信されると起動します。 | — | Deny Connection Inline Product Alert[2] |
| 1301 TCP Inactive Timeout[3] | TCP セッションのアイドル状態が TCP Idle Timeout の時間続くと起動します。 | TCP Idle Timeout：3600（15 ～ 3600） | なし[4] |
| 1302 TCP Embryonic Timeout[5] | TCP セッションのスリーウェイ ハンドシェイクが TCP Embryonic Timeout の秒数内で完了しないと起動します。 | TCP Embryonic Timeout：15（3 ～ 300） | なし[6] |
| 1303 TCP Closing Timeout[7] | TCP セッションが最初の FIN を受信してから TCP Closed Timeout の秒数以内に完全に閉じないと起動します。 | TCP Closed Timeout：5（1 ～ 60） | なし[8] |
| 1304 TCP Max Segments Queued Per Session | セッションに対してキューイングされた無効セグメントの数が TCP Max Queue を超えると起動します。期待されるシーケンスから最も遠いシーケンスが含まれるセグメントがドロップされます。 | TCP Max Queue：32（0 ～ 128） | Deny Packet Inline Product Alert[9] |
| 1305 TCP Urgent Flag[10] | TCP 緊急フラグが検出されると起動します。 | — | Modify Packet Inline がディセーブル[11] |
| 1306 0 TCP Option Other | TCP Option Number の範囲内の TCP オプションが検出されると起動します。 | TCP Option Number：6 ～ 7、9 ～ 255（0 ～ 255 の範囲内での複数の整数範囲） | Modify Packet Inline Produce Alert[12] |
| 1306 1 TCP SACK Allowed Option | TCP 選択 ACK 許可オプションが検出されると起動します。 | — | Modify Packet Inline がディセーブル[13] |
| 1306 2 TCP SACK Data Option | TCP 選択 ACK データ オプションが検出されると起動します。 | — | Modify Packet Inline がディセーブル[14] |

表 7-6　TCP ストリーム再構成シグニチャ （続き）

| シグニチャ ID と名前 | 説明 | パラメータ：デフォルト値（範囲） | デフォルト アクション |
|---|---|---|---|
| 1306 3 TCP Timestamp Option | TCP タイムスタンプ オプションが検出されると起動します。 | — | Modify Packet Inline がディセーブル[15] |
| 1306 4 TCP Window Scale Option | TCP ウィンドウ スケール オプションが検出されると起動します。 | — | Modify Packet Inline がディセーブル[16] |
| 1307 TCP Window Size Variation | TCP の受信ウィンドウの右端が右に移動する（減少する）と起動します。 | — | Deny Connection Inline Produce Alert がディセーブル[17] |
| 1308 TTL Varies[18] | セッションの一方向で検出された TTL が確認済みの最小値よりも大きいと起動します。 | — | Modify Packet Inline[19] |
| 1309 TCP Reserved Bits Set | TCP ヘッダー上で予約ビット（ECN に使用されるビットも含む）が設定されると起動します。 | — | Modify Packet Inline Produce Alert がディセーブル[20] |
| 1310 TCP Retransmit Protection[21] | 再送信されたセグメントに元のセグメントとは異なるデータが含まれていることをセンサーが検出すると起動します。 | — | Deny Connection Inline Produce Alert[22] |
| 1311 TCP Packet Exceeds MSS | パケットがスリーウェイ ハンドシェイク中に交換された MSS を超えていると起動します。 | — | Deny Connection Inline Produce Alert[23] |
| 1312 TCP Min MSS | SYN フラグの含まれるパケット内の MSS 値が TCP Min MSS より小さいと起動します。 | TCP Min MSS：400（0 ～ 16000） | Modify Packet Inline がディセーブル[24] |
| 1313 TCP Max MSS | SYN フラグの含まれるパケット内の MSS 値が TCP Max MSS を超えていると起動します。 | TCP Max MSS：1460（0 ～ 16000） | Modify Packet Inline がディセーブル[25] |
| 1314 TCP Data SYN | SYN パケットで TCP ペイロードが送信されると起動します。 | — | Deny Packet Inline がディセーブル[26] |
| 1315 ACK Without TCP Stream | ストリームに属していない ACK パケットが送信されると起動します。 | — | Produce Alert がディセーブル[27] |
| 1317 Zero Window Probe | ゼロ ウィンドウ プローブ パケットが検出されると起動します。 | Modify Packet Inline は、ゼロ ウィンドウ プローブ パケットからデータを削除します。 | Modify Packet Inline |

表 7-6　TCP ストリーム再構成シグニチャ （続き）

| シグニチャ ID と名前 | 説明 | パラメータ：デフォルト値（範囲） | デフォルト アクション |
|---|---|---|---|
| 1330[28] 0 TCP Drop - Bad Checksum | TCP パケットのチェックサムが不正な場合に起動します。 | Modify Packet Inline は、チェックサムを訂正します。 | Deny Packet Inline |
| 1330 1 TCP Drop - Bad TCP Flags | TCP パケットのフラグの組み合わせが不正な場合に起動します。 | — | Deny Packet Inline |
| 1330 2 TCP Drop - Urgent Pointer With No Flag | TCP パケットに URG ポインタが設定されているのに URG フラグがない場合に起動します。 | Modify Packet Inline は、ポインタをクリアします。 | Modify Packet Inline がディセーブル |
| 1330 3 TCP Drop - Bad Option List | TCP パケットのオプション リストが不正な場合に起動します。 | — | Deny Packet Inline |
| 1330 4 TCP Drop - Bad Option Length | TCP パケットのオプションの長さが不正な場合に起動します。 | — | Deny Packet Inline |
| 1330 5 TCP Drop - MSS Option Without SYN | SYN フラグが設定されていないパケット内で TCP MSS オプションが検出されると起動します。 | Modify Packet Inline は、MSS オプションをクリアします。 | Modify Packet Inline |
| 1330 6 TCP Drop - WinScale Option Without SYN | SYN フラグが設定されていないパケット内で TCP ウィンドウ スケール オプションが検出されると起動します。 | Modify Packet Inline は、ウィンドウ スケール オプションをクリアします。 | Modify Packet Inline |
| 1330 7 TCP Drop - Bad WinScale Option Value | TCP パケットのウィンドウ スケール値が不正な場合に起動します。 | Modify Packet Inline は、最も近い制約値に値を設定します。 | Modify Packet Inline |
| 1330 8 TCP Drop - SACK Allow Without SYN | SYN フラグが設定されていないパケット内で TCP SACK 許可オプションが検出されると起動します。 | Modify Packet Inline は、SACK 許可オプションをクリアします。 | Modify Packet Inline |
| 1330 9 TCP Drop - Data in SYN\|ACK | SYN フラグと ACK フラグが設定されている TCP パケットにデータも格納されている場合に起動します。 | — | Deny Packet Inline |
| 1330 10 TCP Drop - Data Past FIN | FIN の後に TCP データがシーケンスされると起動します。 | — | Deny Packet Inline |

表 7-6　TCP ストリーム再構成シグニチャ （続き）

| シグニチャ ID と名前 | 説明 | パラメータ：デフォルト値（範囲） | デフォルト アクション |
|---|---|---|---|
| 1330 11 TCP Drop - Timestamp not Allowed | タイムスタンプ オプションが許可されていないときに、TCP パケットにタイムスタンプ オプションが含まれていると起動します。 | — | Deny Packet Inline |
| 1330 12 TCP Drop - Segment Out of Order | TCP セグメントが無効なためキューイングできない場合に起動します。 | — | Deny Packet Inline |
| 1330 13 TCP Drop - Invalid TCP Packet | TCP パケットのヘッダーが不正の場合に起動します。 | — | Deny Packet Inline |
| 1330 14 TCP Drop - RST or SYN in window | RST フラグまたは SYN フラグを持つ TCP パケットがシーケンス ウィンドウで送信されたが、次のシーケンスではなかった場合に起動します。 | — | Deny Packet Inline |
| 1330 15 TCP Drop - Segment Already ACKed | TCP パケット シーケンスがピアによってすでに確認応答されている場合に起動します（キープアライブは除く）。 | — | Deny Packet Inline |
| 1330 16 TCP Drop - PAWS Failed | TCP パケットが PAWS チェックに失敗すると起動します。 | — | Deny Packet Inline |
| 1330 17 TCP Drop - Segment out of State Order | TCP パケットが TCP セッションの状態に対して適切ではない場合に起動します。 | — | Deny Packet Inline |
| 1330 18 TCP Drop - Segment out of Window | TCP パケットのシーケンス番号が許可されているウィンドウの範囲外にある場合に起動します。 | — | Deny Packet Inline |
| 3050 Half Open SYN Attack | | syn-flood-max-embryonic：5000 | |
| 3250 TCP Hijack | | max-old-ack：200 | |
| 3251 TCP Hijack Simplex Mode | | max-old-ack：100 | |

1. IPS は、TCP セッションの各方向において最後の 256 バイトを保持します。
2. Modify Packet Inline は、このシグニチャに影響を与えません。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
3. タイマーは、TCP セッション上をパケットが通過するたびに 0 にリセットされます。デフォルトでは、このシグニチャはアラートを生成しません。必要ならば TCP 接続の期限切れに関するアラートを生成させることもできます。期限の切れたフローの合計数に関する統計情報は、フローが期限切れになるたびに更新されます。
4. Modify Packet Inline、Deny Connection Inline、および Deny Packet Inline は、このシグニチャに影響を与えません。
5. タイマーは最初の SYN パケットで始動し、リセットはされません。セッションの状態はリセットされ、このフローの後続のパケットは無効であると見なされます（SYN の場合を除く）。

6. Modify Packet Inline、Deny Connection Inline、および Deny Packet Inline は、このシグニチャに影響を与えません。
7. タイマーは最初の FIN パケットで始動し、リセットはされません。セッションの状態はリセットされ、このフローの後続のパケットは無効であると見なされます（SYN の場合を除く）。
8. Modify Packet Inline、Deny Connection Inline、および Deny Packet Inline は、このシグニチャに影響を与えません。
9. Modify Packet Inline と Deny Packet Inline はこのシグニチャに影響を与えません。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。
10. Phrak 57 は、URG ポインタを使用してセキュリティ ポリシーを回避する方法を記述します。インライン モードの場合、このシグニチャでパケットを正規化できます。
11. Modify Packet Inline は、そのパケットの URG フラグを取り除き、URG ポインタをゼロにします。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
12. Modify Packet Inline は、選択されたオプションをパケットから取り除きます。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
13. Modify Packet Inline は、選択された ACK 許可オプションをパケットから取り除きます。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
14. Modify Packet Inline は、選択された ACK 許可オプションをパケットから取り除きます。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
15. Modify Packet Inline は、タイムスタンプ オプションをパケットから取り除きます。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
16. Modify Packet Inline は、ウィンドウ スケール オプションをパケットから取り除きます。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
17. Modify Packet Inline は、このシグニチャに影響を与えません。Deny Connection Inline は、現在のパケットと TCP 接続をドロップします。Deny Packet Inline はパケットをドロップします。
18. このシグニチャは、セッション上の各方向に対して TTL を単調減少させるために使用されます。たとえば、TTL 45 が A から B の間で検出される最小の TTL の場合、Modify Packet Inline が設定されていると、それ以降の A から B に向かうすべてのパケットは、最大値が 45 になります。新しい小さい TTL が検出されるたびに、その値がそのセッション上の新しい最大値になります。
19. Modify Packet Inline は、IP TTL が単調減少することを保証します。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
20. Modify Packet Inline は、すべての予約 TCP フラグをクリアします。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
21. このシグニチャは、シグニチャ 1300 のように最後の 256 バイトに制限されることはありません。
22. Modify Packet Inline は、このシグニチャに影響を与えません。Deny Connection Inline は、現在のパケットと TCP 接続をドロップします。Deny Packet Inline はパケットをドロップします。
23. Modify Packet Inline は、このシグニチャに影響を与えません。Deny Connection Inline は、現在のパケットと TCP 接続をドロップします。Deny Packet Inline はパケットをドロップします。
24. 2.4.21-15.EL.cisco.1 Modify Packet Inline は、MSS 値を TCP Min MSS に引き上げます。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline は、パケット 2.4.21-15.EL.cisco.1 をドロップします。
25. Modify Packet Inline は、MSS 値を TCP Max MSS に引き下げます。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline は、パケット 2.4.21-15.EL.cisco.1 をドロップします。
26. Modify Packet Inline は、このシグニチャに影響を与えません。Deny Connection Inline は、現在のパケットと TCP セッションをドロップします。Deny Packet Inline はパケットをドロップします。
27. Modify Packet Inline、Deny Connection Inline、および Deny Packet Inline は、このシグニチャに影響を与えません。デフォルトでは、このシグニチャでアラートが送信されるパケットは、1330 シグニチャによってドロップされます。
28. これらのサブシグニチャは、ノーマライザが TCP パケットをドロップする理由を表します。デフォルトでは、これらのサブシグニチャはパケットをドロップします。これらのサブシグニチャを使用すると、ノーマライザのチェックに合格しないパケットに対して IPS の通過を許可できるようになります。ドロップ理由は、TCP 統計情報に集計されます。デフォルトでは、これらのサブシグニチャはアラートを生成しません。

### 詳細情報

ノーマライザ エンジンの詳細については、「ノーマライザ エンジン」（P.B-33）を参照してください。

## TCP ストリーム再構成シグニチャの設定

特定のシグニチャの TCP ストリーム再構成を設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　TCP ストリーム再構成シグニチャ ID とサブシグニチャ ID を指定します。

```
ips-ssp(config-sig)# signatures 1313 0
```

**ステップ 4**　エンジンを指定します。

```
ips-ssp(config-sig-sig)# engine normalizer
```

**ステップ 5**　デフォルト シグニチャの編集サブモードを開始します。

```
ips-ssp(config-sig-sig-nor)# edit-default-sigs-only default-signatures-only
```

**ステップ 6**　シグニチャ 1313 の最大 MSS パラメータについて（必要に応じて）デフォルト設定をイネーブルにし、変更します。

```
ips-ssp(config-sig-sig-nor-def)# specify-tcp-max-mss yes
ips-ssp(config-sig-sig-nor-def-yes)# tcp-max-mss 1380
```

**（注）**　このパラメータをデフォルトの 1460 から 1380 に変更すると、VPN トンネルを通過するトラフィックのフラグメンテーションを防止するのに役立ちます。

**ステップ 7**　設定を確認できます。

```
ips-ssp(config-sig-sig-nor-def-yes)# show settings
   yes
   -----------------------------------------------
      tcp-max-mss: 1380 default: 1460
   -----------------------------------------------
ips-ssp(config-sig-sig-nor-def-yes)#
```

**ステップ 8**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-nor-def-yes)# exit
ips-ssp(config-sig-sig-nor-def)# exit
ips-ssp(config-sig-sig-nor)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 9**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# TCP ストリーム再構成のモードの設定

シグニチャ定義サブモードで **stream-reassembly** コマンドを使用して、センサーが TCP セッションの再構成に使用するモードを設定します。

**（注）** パラメータ **tcp-3-way-handshake-required** と tcp-reassembly-mode only は、無差別モードでトラフィックを検査しているセンサーに影響を与えます。インライン モードでは影響ありません。

### オプション

次のオプションが適用されます。

- **tcp-3-way-handshake-required {true | false}**：スリーウェイ ハンドシェイクを完了したセッションのみをセンサーで追跡することを指定します。デフォルトは true です。
- **tcp-reassembly-mode**：センサーが TCP セッションの再構成に使用するモードを指定します。
  - **strict**：シーケンスで次に予測されるものだけを許可します。
  - **loose**：シーケンスのギャップを許容します。
  - **asym**：非対称トラフィックの再構成を許可します。

    デフォルトは strict です。

**注意** 非対称オプションは、TCP ウィンドウ回避チェックをディセーブルにします。

### TCP ストリーム再構成パラメータの設定

TCP ストリーム再構成パラメータを設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　TCP ストリーム再構成サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
ips-ssp(config-sig)# stream-reassembly
```

**ステップ 3**　スリーウェイ ハンドシェイクを完了したセッションのみをセンサーで追跡することを指定します。

```
ips-ssp(config-sig-str)# tcp-3-way-handshake-required true
```

**ステップ 4**　センサーが TCP セッションの再構成に使用するモードを指定します。

```
ips-ssp(config-sig-str)# tcp-reassembly-mode strict
```

**ステップ 5**　設定を確認できます。

```
ips-ssp(config-sig-str)# show settings
   stream-reassembly
   -----------------------------------------------
      tcp-3-way-handshake-required: true default: true
      tcp-reassembly-mode: strict default: strict
   -----------------------------------------------
ips-ssp(config-sig-str)#
```

**ステップ 6**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-str)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 7**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# IP ロギングの設定

センサーが攻撃を検出したときに、IP セッション ログを生成するように設定できます。シグニチャの応答アクションとして IP ロギングが設定されているときにシグニチャがトリガーされると、アラートの送信元アドレスとの間で送受信されるすべてのパケットが指定された時間の間ログに記録されます。

シグニチャ定義サブモードで **ip-log** コマンドを使用して、IP ロギングを設定します。

### オプション

次のオプションが適用されます。

- **ip-log-bytes**：ログに記録する最大バイト数を指定します。有効な値は 0 ～ 2147483647 です。デフォルトは 0 です。
- **ip-log-packets**：ログに記録するパケットの数を指定します。有効な値は 0 ～ 65535 です。デフォルトは 0 です。
- **ip-log-time**：センサーでログを記録する期間を指定します。有効な値は 30 ～ 300 秒です。デフォルトは 30 秒です。

**（注）**　センサーは、これらの IP ロギング条件のいずれか 1 つを満たすと、IP ロギングを停止します。

### IP ロギング パラメータの設定

IP ロギング パラメータを設定するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　IP ログ サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
ips-ssp(config-sig)# ip-log
```

**ステップ 3**　IP ロギング パラメータを指定します。

**a.**　ログに記録する最大バイト数を指定します。

```
ips-ssp(config-sig-ip)# ip-log-bytes 200000
```

**b.**　ログに記録するパケット数を指定します。

```
ips-ssp(config-sig-ip)# ip-log-packets 150
```

**c.**　センサーでログを記録する時間の長さを指定します。

```
ips-ssp(config-sig-ip)# ip-log-time 60
```

**ステップ 4**　設定を確認できます。

```
ips-ssp(config-sig-ip)# show settings
   ip-log
   -----------------------------------------------
      ip-log-packets: 150 default: 0
      ip-log-time: 60 default: 30
      ip-log-bytes: 200000 default: 0
   -----------------------------------------------
ips-ssp(config-sig-ip)#
```

**ステップ 5**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-ip)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 6**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# カスタム シグニチャの作成

**注意**　カスタム シグニチャは、センサーのパフォーマンスに影響を与える可能性があります。ネットワークのベースライン センサー パフォーマンスに対してカスタム シグニチャをテストして、そのシグニチャがネットワーク全体に与える影響を確認してください。

ここでは、カスタム シグニチャを作成する方法について説明します。次のような構成になっています。



## カスタム シグニチャの作成手順

カスタム シグニチャを作成するには、次の手順を使用します。

**ステップ 1**　シグニチャ エンジンを選択します。

**ステップ 2**　シグニチャの識別情報を割り当てます。

- シグニチャ ID
- サブシグニチャ ID
- シグニチャ名
- アラート ノート（オプション）

- ユーザ コメント（オプション）

**ステップ 3**　エンジン固有のパラメータを割り当てます。これらのパラメータはシグニチャ エンジンごとに異なりますが、各エンジンに適用される一群のマスター パラメータが存在します。

**ステップ 4**　アラート応答を割り当てます。

- シグニチャ忠実度レーティング
- アラートの重大度

**ステップ 5**　アラート動作を割り当てます。

**ステップ 6**　変更を適用します。

# String TCP エンジン シグニチャの例

**注意**　カスタム シグニチャは、センサーのパフォーマンスに影響を与える可能性があります。ネットワークのベースライン センサー パフォーマンスに対してカスタム シグニチャをテストして、そのシグニチャがネットワーク全体に与える影響を確認してください。

**（注）**　この手順は、String UDP エンジン シグニチャと String ICMP エンジン シグニチャにも適用されます。

String エンジンは、ICMP、TCP、および UDP 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検査エンジンです。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正規表現エンジンが使用されます。String エンジンには、String ICMP、String TCP、および String UDP の 3 種類が存在します。

### オプション

次のオプションが適用されます。

- **default**：値をシステムのデフォルト設定に戻します。
- **direction**：トラフィックの方向を指定します。
  - **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。
- **event-action**：アラートがトリガーされたときに実行するアクションを指定します。
  - **deny-attacker-inline**：（インラインのみ）指定された期間、この攻撃者のアドレスからの現在および将来のパケットを送信しません。
  - **deny-attacker-service-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象ポートがペアになっている現在および将来のパケットを送信しません。
  - **deny-attacker-victim-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象アドレスがペアになっている現在および将来のパケットを送信しません。
  - **deny-connection-inline**：（インラインのみ）この TCP フロー上の現在および将来のパケットを送信しません。
  - **deny-packet-inline**（インラインのみ）現在のパケットを送信しません。
  - **log-attacker-packets**：攻撃者のアドレスが含まれているパケットの 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のペアが含まれているパケットの IP ロギングを開始します。
  - **log-victim-packets**：攻撃対象のアドレスが含まれているパケットの IP ロギングを開始します。
  - **produce-alert**：イベントをアラートとしてイベント ストアに書き込みます。
  - **produce-verbose-alert**：攻撃パケットの符号化されたダンプ（切り詰められる可能性あり）をアラートに含めます。
  - **request-block-connection**：要求を ARC に送信して、この接続をブロックします。
  - **request-block-host**：要求を ARC に送信して、この攻撃者のホストをブロックします。
  - **request-rate-limit**：レート制限要求を ARC に送信して、レート制限を実行します。
  - **request-snmp-trap**：要求をセンサーの通知アプリケーション コンポーネントに送信して、SNMP 通知を実行します。
  - **reset-tcp-connection**：TCP リセットを送信して、TCP フローをハイジャックし、終了します。
  - **modify-packet-inline**：エンド ポイントによるパケットの処理に関するあいまいさを取り除くために、パケット データを変更します。
- **no**：エントリまたは選択の設定を削除します。
- **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。
- **service-ports**：ターゲット サービスが常駐する可能性のあるポートまたはポート範囲を指定します。有効な範囲は 0 ～ 65535 です。これは、0 ～ 65535 の範囲内に収まる整数範囲のリストです（a-b[,c-d]）。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。
- **specify-exact-match-offset {yes | no}**：（任意）完全一致オフセットをイネーブルにします。
- **specify-min-match-length {yes | no}**：（任意）最小一致長をイネーブルにします。
- **strip-telnet-options**：検索前にデータから Telnet オプション文字を取り除きます。
- **swap-attacker-victim {true | false}**：アラート メッセージ内および実行されるアクションに対し、攻撃者と攻撃対象（送信元と宛先）のアドレスおよびポートを交換します。デフォルトは、交換しないことを意味する false です。

### String TCP エンジン シグニチャの作成

String TCP エンジンに基づいてシグニチャを作成するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　シグニチャのシグニチャ ID とサブシグニチャ ID を指定します。カスタム シグニチャ ID の範囲は、60000 ～ 65000 です。

```
ips-ssp(config-sig)# signatures 60025 0
```

**ステップ 4**　シグニチャ説明サブモードを開始します。

```
ips-ssp(config-sig-sig)# sig-description
```

**ステップ 5**　新しいシグニチャの名前を指定します。また、**sig-comment** コマンドを使用してシグニチャに関する追加コメントを指定したり、**sig-string-info** コマンドを使用してシグニチャに関する追加情報を指定したりすることもできます。

```
ips-ssp(config-sig-sig-sig)# sig-name This is my new name
```

**ステップ 6**　シグニチャ説明サブモードを終了します。

```
ips-ssp(config-sig-sig-sig)# exit
```

**ステップ 7**　String TCP エンジンを指定します。

```
ips-ssp(config-sig-sig)# engine string-tcp
```

**ステップ 8**　サービス ポートを指定します。

```
ips-ssp(config-sig-sig-str)# service-ports 23
```

**ステップ 9**　方向を指定します。

```
ips-ssp(config-sig-sig-str)# direction to-service
```

**ステップ 10**　TCP パケット内で検索する正規表現文字列を指定します。セキュリティ ポリシーに従い、必要ならば **event-action** コマンドを使用してイベント アクションを変更できます。デフォルトのイベント アクションは、**produce-alert** です。

```
ips-ssp(config-sig-sig-str)# regex-string This-is-my-new-Sig-regex
```

**ステップ 11**　このカスタム String TCP シグニチャに対して、次のオプション パラメータを変更できます。

- **specify-exact-match-offset**
- **specify-min-match-length**
- **strip-telnet-options**
- **swap-attacker-victim**

**ステップ 12**　設定を確認できます。

```
ips-ssp(config-sig-sig-str)# show settings
   string-tcp
   -----------------------------------------------
      event-action: produce-alert <defaulted>
      strip-telnet-options: false <defaulted>
      specify-min-match-length
      -----------------------------------------------
         no
         -----------------------------------------------
         -----------------------------------------------
      -----------------------------------------------
      regex-string: This-is-my-new-Sig-regex
      service-ports: 23
      direction: to-service default: to-service
      specify-exact-match-offset
      -----------------------------------------------
         no
         -----------------------------------------------
            specify-max-match-offset
            -----------------------------------------------
               no
               -----------------------------------------------
               -----------------------------------------------
            -----------------------------------------------
            specify-min-match-offset
            -----------------------------------------------
               no
               -----------------------------------------------
               -----------------------------------------------
            -----------------------------------------------
         -----------------------------------------------
      -----------------------------------------------
```

```
      swap-attacker-victim: false <defaulted>
   ------------------------------------------------
ips-ssp(config-sig-sig-str)#
```

**ステップ 13**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-str)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 14**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

# Service HTTP エンジン シグニチャの例

**注意**　カスタム シグニチャは、センサーのパフォーマンスに影響を与える可能性があります。ネットワークのベースライン センサー パフォーマンスに対してカスタム シグニチャをテストして、そのシグニチャがネットワーク全体に与える影響を確認してください。

Service HTTP エンジンは、サービスに特化した文字列ベースのパターンマッチング検査エンジンです。HTTP プロトコルは、今日のネットワークで最も一般的に使用されるプロトコルの 1 つです。また、必要な前処理の時間が非常に長く、検査を必要とするシグニチャも非常に多いため、システムの全体的なパフォーマンスを決める要因になっています。

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 サービスに向かう方向のみのトラフィック、または HTTP 要求を検索します。このエンジンでリターン トラフィックを検査することはできません。このエンジンでは、シグニチャごとに対象の Web ポートを別々に指定できます。

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 エンジンのデフォルトの解読動作は Microsoft IIS Web サーバを対象としています。

### オプション

次のオプションが適用されます。

- **de-obfuscate {true | false}**：検索前に反回避解読を適用します。
- **default**：値をシステムのデフォルト設定に戻します。
- **event-action**：アラートがトリガーされたときに実行するアクションを指定します。
  - **deny-attacker-inline**：（インラインのみ）指定された期間、この攻撃者のアドレスからの現在および将来のパケットを送信しません。
  - **deny-attacker-service-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象ポートがペアになっている現在および将来のパケットを送信しません。

  - **deny-attacker-victim-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象アドレスがペアになっている現在および将来のパケットを送信しません。
  - **deny-connection-inline**：（インラインのみ）この TCP フロー上の現在および将来のパケットを送信しません。
  - **deny-packet-inline**（インラインのみ）現在のパケットを送信しません。
  - **log-attacker-packets**：攻撃者のアドレスが含まれているパケットの IP ロギングを開始します。このアクションによって、**produce-alert** が選択されていない場合でも、アラートがイベント ストアに書き込まれます。
  - **log-pair-packets**：攻撃者と攻撃対象のアドレスのペアが含まれているパケットの IP ロギングを開始します。このアクションによって、**produce-alert** が選択されていない場合でも、アラートがイベント ストアに書き込まれます。
  - **log-victim-packets**：攻撃対象のアドレスが含まれているパケットの IP ロギングを開始します。このアクションによって、**produce-alert** が選択されていない場合でも、アラートがイベント ストアに書き込まれます。
  - **produce-alert**：イベントをアラートとしてイベント ストアに書き込みます。
  - **produce-verbose-alert**：攻撃パケットの符号化されたダンプ（切り詰められる可能性あり）をアラートに含めます。このアクションによって、**produce-alert** が選択されていない場合でも、アラートがイベント ストアに書き込まれます。
  - **request-block-connection**：要求を ARC に送信して、この接続をブロックします。ブロッキング デバイスは、このアクションを実行するように設定されている必要があります。
  - **request-block-host**：要求を ARC に送信して、この攻撃者のホストをブロックします。ブロッキング デバイスは、このアクションを実行するように設定されている必要があります。
  - **request-rate-limit**：レート制限要求を ARC に送信して、レート制限を実行します。レート制限デバイスは、このアクションを実行するように設定されている必要があります。
  - **request-snmp-trap**：要求をセンサーの通知アプリケーション コンポーネントに送信して、SNMP 通知を実行します。このアクションによって、**produce-alert** が選択されていない場合でも、アラートがイベント ストアに書き込まれます。このアクションを実装するには、センサーで SNMP を設定する必要があります。
  - **reset-tcp-connection**：TCP リセットを送信して、TCP フローをハイジャックし、終了します。**Reset TCP Connection** は、単一の接続を分析する TCP シグニチャ上でのみ動作します。スイープまたはフラッドに対しては機能しません。
  - **modify-packet-inline**：エンド ポイントによるパケットの処理に関するあいまいさを取り除くために、パケット データを変更します。
- **max-field-sizes**：最大フィールド サイズのグループをイネーブルにします。
  - **specify-max-arg-field-length {yes | no}**：max-arg-field-length をイネーブルにします（オプション）。
  - **specify-max-header-field-length {yes | no}**：max-header-field-length をイネーブルにします（オプション）。
  - **specify-max-request-length {yes | no}**：max-request-length をイネーブルにします（オプション）。
  - **specify-max-uri-field-length {yes | no}**：max-uri-field-length をイネーブルにします（オプション）。
- **no**：エントリまたは選択の設定を削除します。
- **regex**：正規表現のグループをイネーブルにします。
  - **specify-arg-name-regex**：arg-name-regex をイネーブルにします（オプション）。

  - **specify-header-regex**：header-regex をイネーブルにします（オプション）。
  - **specify-request-regex**：request-regex をイネーブルにします（オプション）。
  - **specify-uri-regex**：uri-regex をイネーブルにします（オプション）。
- **service-ports**：ターゲット サービスが常駐する可能性のあるポートまたはポート範囲をカンマ区切りのリストで指定します。
- **swap-attacker-victim {true | false}**：アラート メッセージ内および実行されるアクションに対し、攻撃者と攻撃対象（送信元と宛先）のアドレスおよびポートを交換します。デフォルトは、交換しないことを意味する false です。

### Service HTTP エンジン シグニチャの作成

Service HTTP エンジンに基づいてカスタム シグニチャを作成するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　シグニチャのシグニチャ ID とサブシグニチャ ID を指定します。カスタム シグニチャ ID の範囲は、60000 ～ 65000 です。

```
ips-ssp(config-sig)# signatures 63000 0
```

**ステップ 4**　シグニチャ説明モードを開始します。

```
ips-ssp(config-sig-sig)# sig-description
```

**ステップ 5**　シグニチャ名を指定します。

```
ips-ssp(config-sig-sig-sig)# sig-name myWebSig
```

**ステップ 6**　アラート特性を指定します。有効な範囲は 0 ～ 65535 です。

```
ips-ssp(config-sig-sig-sig)# alert-traits 2
```

**ステップ 7**　シグニチャ説明サブモードを終了します。

```
ips-ssp(config-sig-sig-sig)# exit
```

**ステップ 8**　アラート頻度を指定します。

```
ips-ssp(config-sig-sig)# alert-frequency
ips-ssp(config-sig-sig-ale)# summary-mode fire-all
ips-ssp(config-sig-sig-ale-fir)# summary-key Axxx
ips-ssp(config-sig-sig-ale-fir)# specify-summary-threshold yes
ips-ssp(config-sig-sig-ale-fir-yes)# summary-threshold 200
```

**ステップ 9**　アラート頻度サブモードを終了します。

```
ips-ssp(config-sig-sig-ale-fir-yes)# exit
ips-ssp(config-sig-sig-ale-fir)# exit
ips-ssp(config-sig-sig-ale)# exit
```

**ステップ 10**　検索前に反回避解読を適用するようにシグニチャを設定します。

```
ips-ssp(config-sig-sig)# engine service-http
ips-ssp(config-sig-sig-ser)# de-obfuscate true
```

**ステップ 11**　正規表現パラメータを設定します。

```
ips-ssp(config-sig-sig)# engine service-http
ips-ssp(config-sig-sig-ser)# regex
ips-ssp(config-sig-sig-ser-reg)# specify-uri-regex yes
ips-ssp(config-sig-sig-ser-reg-yes)# uri-regex [Mm][Yy][Ff][Oo][Oo]
```

**ステップ 12**　正規表現サブモードを終了します。

```
ips-ssp(config-sig-sig-ser-reg-yes)# exit
ips-ssp(config-sig-sig-ser-reg-)# exit
```

**ステップ 13**　シグニチャ変数 WEBPORTS を使用してサービス ポートを設定します。

```
ips-ssp(config-sig-sig-ser)# service-ports $WEBPORTS
```

**ステップ 14**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-ser)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 15**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

**詳細情報**

シグニチャの正規表現の構文の一覧表については、付録 B「正規表現の構文」を参照してください。

# Meta エンジン シグニチャの例

**注意**　カスタム シグニチャは、センサーのパフォーマンスに影響を与える可能性があります。ネットワークのベースライン センサー パフォーマンスに対してカスタム シグニチャをテストして、そのシグニチャがネットワーク全体に与える影響を確認してください。

**注意**　Meta エンジン シグニチャを大量に使用すると、全体的なセンサー パフォーマンスに悪影響を与える可能性があります。

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）**　Meta エンジンは、ほとんどのエンジンがパケットを入力としているにもかかわらず、アラートを入力としている点が他のエンジンとは異なります。

### オプション

次のオプションが適用されます。

- **component-list**：Meta コンポーネントのリスト。
  - **edit**：リスト内の既存のエントリを編集します。
  - **insert name1**：新しいエントリをリストに挿入します。
  - **move**：リスト内のエントリを移動します。
  - **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 リストの末尾に配置します。
  - **inactive**：エントリを非アクティブ リストに配置します。
  - **before**：エントリを指定エントリの前に配置します。
  - **after**：エントリを指定エントリの後に配置します。
- **component-count**：このコンポーネントが満たされるまでにコンポーネントが起動しなければならない回数を指定します。
- **component-sig-id**：このコンポーネントを照合するシグニチャのシグニチャ ID を指定します。
- **component-subsig-id**：このコンポーネントを照合するシグニチャのサブシグニチャ ID を指定します。
- **component-list-in-order {true | false}**：コンポーネント リストの順に起動するかどうか。
- **event-action**：アラートがトリガーされたときに実行するアクションを指定します。
  - **deny-attacker-inline**：（インラインのみ）指定された期間、この攻撃者のアドレスからの現在および将来のパケットを送信しません。
  - **deny-attacker-service-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象ポートがペアになっている現在および将来のパケットを送信しません。
  - **deny-attacker-victim-pair-inline**：（インラインのみ）指定された期間、この攻撃者アドレスと攻撃対象アドレスがペアになっている現在および将来のパケットを送信しません。
  - **deny-connection-inline**：（インラインのみ）この TCP フロー上の現在および将来のパケットを送信しません。
  - **deny-packet-inline**（インラインのみ）現在のパケットを送信しません。
  - **log-attacker-packets**：攻撃者のアドレスが含まれているパケットの 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のペアが含まれているパケットの IP ロギングを開始します。
  - **log-victim-packets**：攻撃対象のアドレスが含まれているパケットの IP ロギングを開始します。
  - **produce-alert**：イベントをアラートとしてイベント ストアに書き込みます。
  - **produce-verbose-alert**：攻撃パケットの符号化されたダンプ（切り詰められる可能性あり）をアラートに含めます。
  - **request-block-connection**：要求を ARC に送信して、この接続をブロックします。
  - **request-block-host**：要求を ARC に送信して、この攻撃者のホストをブロックします。
  - **request-rate-limit**：レート制限要求を ARC に送信して、レート制限を実行します。
  - **request-snmp-trap**：要求をセンサーの通知アプリケーション コンポーネントに送信して、SNMP 通知を実行します。
  - **reset-tcp-connection**：TCP リセットを送信して、TCP フローをハイジャックし、終了します。

  - **modify-packet-inline**：エンド ポイントによるパケットの処理に関するあいまいさを取り除くために、パケット データを変更します。
- **meta-key**：Meta シグニチャのストレージ タイプを指定します。
  - **AaBb**：攻撃者と攻撃対象のアドレスおよびポート。
  - **AxBx**：攻撃者と攻撃対象のアドレス。
  - **Axxx**：攻撃者のアドレス。
  - **xxBx**：攻撃対象のアドレス。
- **meta-reset-interval**：Meta シグニチャをリセットする時間を秒数で指定します。有効な値の範囲は 0 ～ 3600 秒です。デフォルトは 60 秒です。

### Meta エンジン シグニチャの作成

シグニチャ 64000 のサブシグニチャ 0 は、シグニチャ 2000 のサブシグニチャ 0 からのアラートとシグニチャ 3000 のサブシグニチャ 0 からのアラートを同じ送信元アドレスで検出すると起動します。送信元アドレスが選択されているのは、メタ キーのデフォルト値が Axxx であるためです。この動作は、メタ キーの設定を xxBx（宛先アドレス）などに変更することにより変更できます。

Meta エンジンに基づいてシグニチャを作成するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　シグニチャのシグニチャ ID とサブシグニチャ ID を指定します。カスタム シグニチャ ID の範囲は、60000 ～ 65000 です。

```
ips-ssp(config-sig)# signatures 64000 0
```

**ステップ 4**　シグニチャ エンジンを指定します。

```
ips-ssp(config-sig-sig)# engine meta
```

**ステップ 5**　シグニチャ（名前は c1）をリストの先頭に挿入します。

```
ips-ssp(config-sig-sig-met)# component-list insert c1 begin
```

**ステップ 6**　このコンポーネントを照合するシグニチャのシグニチャ ID を指定します。

```
ips-ssp(config-sig-sig-met-com)# component-sig-id 2000
```

**ステップ 7**　コンポーネント リスト サブモードを終了します。

```
ips-ssp(config-sig-sig-met-com)# exit
```

**ステップ 8**　別のシグニチャ（名前は c2）をリストの末尾に挿入します。

```
ips-ssp(config-sig-sig-met)# component-list insert c2 end
```

**ステップ 9**　このコンポーネントを照合するシグニチャのシグニチャ ID を指定します。

```
ips-ssp(config-sig-sig-met-com)# component-sig-id 3000
```

**ステップ 10**　設定を確認できます。

```
ips-ssp(config-sig-sig-met-com)# exit
ips-ssp(config-sig-sig-met)# show settings
   meta
   -----------------------------------------------
      event-action: produce-alert <defaulted>
      meta-reset-interval: 60 <defaulted>
      component-list (min: 1, max: 8, current: 2 - 2 active, 0 inactive)
      -----------------------------------------------
      ACTIVE list-contents
      -----------------------------------------------
         NAME: c1
         -----------------------------------------------
            component-sig-id: 2000
            component-subsig-id: 0 <defaulted>
            component-count: 1 <defaulted>
         -----------------------------------------------
         -----------------------------------------------
         NAME: c2
         -----------------------------------------------
            component-sig-id: 3000
            component-subsig-id: 0 <defaulted>
            component-count: 1 <defaulted>
         -----------------------------------------------
         -----------------------------------------------
      -----------------------------------------------
      meta-key
      -----------------------------------------------
         Axxx
         -----------------------------------------------
            unique-victims: 1 <defaulted>
         -----------------------------------------------
      -----------------------------------------------
      component-list-in-order: false <defaulted>
   -----------------------------------------------
ips-ssp(config-sig-sig-met)#
```

**ステップ 11**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-met)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 12**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

---

### 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.8-2）を参照してください。
- Meta エンジンの詳細については、「Meta エンジン」（P.B-31）を参照してください。

# Atomic IP Advanced エンジン シグニチャの例

**注意**　カスタム シグニチャは、センサーのパフォーマンスに影響を与える可能性があります。ネットワークのベースライン センサー パフォーマンスに対してカスタム シグニチャをテストして、そのシグニチャがネットワーク全体に与える影響を確認してください。

次の Atomic IP Advanced カスタム シグニチャの例は、IPv6 上でプロトコル ID 88 を禁止します。

Atomic IP Advanced シグニチャ エンジンに基づいてシグニチャを作成するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig0
```

**ステップ 3**　シグニチャのシグニチャ ID とサブシグニチャ ID を指定します。カスタム シグニチャ ID の範囲は、60000 ～ 65000 です。

```
ips-ssp(config-sig)# signatures 60000 0
```

**ステップ 4**　シグニチャ エンジンを指定します。

```
ips-ssp(config-sig-sig)# engine atomic-ip-advanced
```

**ステップ 5**　IP バージョンを指定します。

```
ips-ssp(config-sig-sig-ato)# specify-ip-version yes
```

**ステップ 6**　IPv6 を指定します。

```
ips-ssp(config-sig-sig-ato-yes)# version ipv6
```

**ステップ 7**　L4 プロトコルを指定します。

```
ips-ssp(config-sig-sig-ato-yes-ipv)# exit
ips-ssp(config-sig-sig-ato-yes)# exit
ips-ssp(config-sig-sig-ato)# specify-l4-protocol yes
```

**ステップ 8**　プロトコル ID 88 を指定します。

```
ips-ssp(config-sig-sig-ato-yes)# l4-protocol other-protocol
ips-ssp(config-sig-sig-ato-yes-oth)# other-ip-protocol-id 88
```

**ステップ 9**　設定を確認できます。

```
ips-ssp(config-sig-sig-ato-yes-oth)# show settings
   other-protocol
   -----------------------------------------------
      other-ip-protocol-id: 88
   -----------------------------------------------
ips-ssp(config-sig-sig-ato-yes-oth)#
```

**ステップ 10**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-ato-yes-oth)# exit
ips-ssp(config-sig-sig-ato-yes)# exit
ips-ssp(config-sig-sig-ato)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes?[yes]:
```

**ステップ 11**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

### 詳細情報

- Atomic IP Advanced エンジンの詳細およびパラメータのリストについては、「Atomic IP Advanced エンジン」（P.B-14）を参照してください。
- Atomic エンジンの詳細については、「Atomic エンジン」（P.B-12）を参照してください。

# String XL エンジンの一致オフセット シグニチャの例

**注意**　カスタム シグニチャは、センサーのパフォーマンスに影響を与える可能性があります。ネットワークのベースライン センサー パフォーマンスに対してカスタム シグニチャをテストして、そのシグニチャがネットワーク全体に与える影響を確認してください。

**（注）**　この手順は、String XLUDP シグニチャと String XL ICMP シグニチャにも適用されます。ただし、パラメータ **service-ports** は String XL ICMP シグニチャに適用されません。

### オプション

次に、完全一致オフセット、最大一致オフセット、または最小一致オフセットを検索するカスタム String XL TCP シグニチャを作成する例を示します。このカスタム String XL TCP シグニチャに対して、次のオプションの一致オフセット パラメータを変更できます。

- **specify-exact-match-offset {yes |no}**：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（バイト単位）を指定します。値は 0 ～ 65535 です。
- **specify-max-match-offset {yes |no}**：最大一致長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（バイト単位）を指定します。値は 0 ～ 65535 です。
- **specify-min-match-offset {yes |no}**：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（バイト単位）を指定します。値は 0 ～ 65535 です。

### String XL TCP エンジン シグニチャの作成

次に、完全一致オフセット、最大一致オフセット、または最小一致オフセットを検索するカスタム String XL TCP シグニチャを作成する例を示します。

String XL TCP エンジンに基づいて一致を検索するカスタム シグニチャを作成するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　シグニチャのシグニチャ ID とサブシグニチャ ID を指定します。カスタム シグニチャ ID の範囲は、60000 ～ 65000 です。

```
ips-ssp(config-sig)# signatures 60003 0
```

**ステップ 4**　シグニチャ説明サブモードを開始します。

```
ips-ssp(config-sig-sig)# sig-description
```

**ステップ 5**　新しいシグニチャの名前を指定します。また、**sig-comment** コマンドを使用してシグニチャに関する追加コメントを指定したり、**sig-string-info** コマンドを使用してシグニチャに関する追加情報を指定したりすることもできます。

```
ips-ssp(config-sig-sig-sig)# sig-name This is my new name
```

**ステップ 6**　シグニチャ説明サブモードを終了します。

```
ips-ssp(config-sig-sig-sig)# exit
```

**ステップ 7**　String XL TCP エンジンを指定します。

```
ips-ssp(config-sig-sig)# engine string-xl-tcp
```

**ステップ 8**　サービス ポートを指定します。

```
ips-ssp(config-sig-sig-str)# service-ports 80
```

**ステップ 9**　方向を指定します。

```
ips-ssp(config-sig-sig-str)# direction to-service
```

**ステップ 10**　必要ならば、セキュリティ ポリシーに従い、**event-action** コマンドを使用してイベント アクションを変更します。デフォルトのイベント アクションは、**produce-alert** です。

**ステップ 11**　raw 正規表現をオフにします。

```
ips-ssp(config-sig-sig-str)# specify-raw-regex-string no
```

**（注）**　raw 正規表現とは、raw モード処理に使用される正規表現の構文です。エキスパート モード専用であり、Cisco IPS シグニチャ開発チーム、あるいはその監視下にある作業者のみによる使用を対象としています。String XL シグニチャは、通常の正規表現と raw 正規表現のいずれでも設定できます。

**ステップ 12**　TCP パケット内で検索する正規表現文字列を指定します。

```
ips-ssp(config-sig-sig-str-no)# regex-string tcpstring
```

**ステップ 13**　オプションの String XL TCP パラメータを設定するために、raw 正規表現モードを終了します。

```
ips-ssp(config-sig-sig-str-no)# exit
ips-ssp(config-sig-sig-str)#
```

**ステップ 14**　このシグニチャの完全一致オフセットを指定します。

```
ips-ssp(config-sig-sig-str)# specify-exact-match-offset yes
ips-ssp(config-sig-sig-str-yes)# exact-match-offset 20
```

**（注）**　完全一致オフセットが **yes** に設定されている場合、最大一致オフセットと最小一致オフセットは設定できません。完全一致オフセットが **no** に設定されている場合、最大一致オフセットと最小一致オフセットの両方を同時に設定できます。

**ステップ 15**　完全一致オフセットをオフにし、このシグニチャの最大一致オフセットを指定します。

```
ips-ssp(config-sig-sig-str-yes)# exit
ips-ssp(config-sig-sig-str)# specify-exact-match-offset no
ips-ssp(config-sig-sig-str-no)# specify-max-match-offset yes
ips-ssp(config-sig-sig-str-no-yes)# max-match-offset 30
```

**ステップ 16**　このシグニチャの最小一致オフセットを指定します。

```
ips-ssp(config-sig-sig-str-no-yes)# exit
ips-ssp(config-sig-sig-str-no)# specify-min-match-offset yes
ips-ssp(config-sig-sig-str-no-yes)# min-match-offset 20
```

**ステップ 17**　設定を確認できます。

```
ips-ssp(config-sig-sig-str-no-yes)# exit
ips-ssp(config-sig-sig-str-no)# exit
ips-ssp(config-sig-sig-str)# show settings
   string-xl-tcp
   -----------------------------------------------
      event-action: produce-alert <defaulted>
      strip-telnet-options: false <defaulted>
      direction: to-service default: to-service
      service-ports: 80
      specify-max-stream-length
      -----------------------------------------------
         no
         -----------------------------------------------
         -----------------------------------------------
      -----------------------------------------------
      specify-raw-regex-string
      -----------------------------------------------
         no
         -----------------------------------------------
            regex-string: tcpstring
            dot-all: false <defaulted>
            end-optional: false <defaulted>
            no-case: false <defaulted>
            stingy: false <defaulted>
            utf8: false <defaulted>
            specify-min-match-length
            -----------------------------------------------
               no
               -----------------------------------------------
               -----------------------------------------------
            -----------------------------------------------
         -----------------------------------------------
      -----------------------------------------------
```

```
            swap-attacker-victim: false <defaulted>
            specify-exact-match-offset
            ------------------------------------------------
               no
               ------------------------------------------------
                  specify-max-match-offset
                  ------------------------------------------------
                     yes
                     ------------------------------------------------
                        max-match-offset: 30
                     ------------------------------------------------
                  ------------------------------------------------
                  specify-min-match-offset
                  ------------------------------------------------
                     yes
                     ------------------------------------------------
                        min-match-offset: 20
                     ------------------------------------------------
                  ------------------------------------------------
               ------------------------------------------------
            ------------------------------------------------
         ------------------------------------------------
ips-ssp(config-sig-sig-str)#
```

**ステップ 18**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-str)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 19**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

### 詳細情報

- すべての String XL パラメータの一覧については、「String XL エンジン」（P.B-59）を参照してください。
- 正規表現の構文の一覧表については、「正規表現の構文」（P.B-10）を参照してください。

# String XL エンジンの最小一致長シグニチャの例

**注意**　カスタム シグニチャは、センサーのパフォーマンスに影響を与える可能性があります。ネットワークのベースライン センサー パフォーマンスに対してカスタム シグニチャをテストして、そのシグニチャがネットワーク全体に与える影響を確認してください。

**（注）**　この手順は、String XL UDP シグニチャと String XL ICMP シグニチャにも適用されます。ただし、パラメータ **service-ports** は String XL ICMP シグニチャに適用されません。

### オプション

次のオプション パラメータを変更して、特定の正規表現文字列を操作できます。

- **dot-all true {true | false}**：true に設定すると、\n も含め [\x00-\xFF] と一致します。false に設定すると、\n を除いた範囲 [\x00-\xFF] 内の任意の文字と一致します。デフォルトは false です。

- **specify-min-match-length {yes | no}**：最小一致長をイネーブルにします。
  - **min-match-length**：パターンがヒットしたと見なされるために正規表現文字列が一致しなければならない最大バイト数を指定します。値は 0 ～ 65535 です。
- **stingy {true | false}**：true に設定すると、最初の一致完了後のより長い一致の検索が中止されます。デフォルトは false です。

（注）　stingy は、**min-match-length** でのみ使用できます。それ以外の場合は無視されます。

- **utf8 {true | false}**：true に設定すると、表現内のすべての有効な UTF-8 バイト シーケンスが単一文字として扱われます。デフォルトは false です。

### String XL TCP エンジン シグニチャの作成

次に、stingy、dot all、および UTF-8 をオンにして最小一致長を検索するカスタム String XL TCP シグニチャを作成する例を示します。

String XL TCP エンジンに基づいて、stingy、dot all、および UTF-8 をオンにして最小一致長を検索するカスタム シグニチャを作成するには、次の手順を実行します。

**ステップ 1**　管理者権限またはオペレータ権限を持つアカウントを使用して CLI にログインします。

**ステップ 2**　シグニチャ定義サブモードを開始します。

```
ips-ssp# configure terminal
ips-ssp(config)# service signature-definition sig1
```

**ステップ 3**　シグニチャのシグニチャ ID とサブシグニチャ ID を指定します。カスタム シグニチャ ID の範囲は、60000 ～ 65000 です。

```
ips-ssp(config-sig)# signatures 60004 0
```

**ステップ 4**　シグニチャ説明サブモードを開始します。

```
ips-ssp(config-sig-sig)# sig-description
```

**ステップ 5**　新しいシグニチャの名前を指定します。また、**sig-comment** コマンドを使用してシグニチャに関する追加コメントを指定したり、**sig-string-info** コマンドを使用してシグニチャに関する追加情報を指定したりすることもできます。

```
ips-ssp(config-sig-sig-sig)# sig-name This is my new name
```

**ステップ 6**　シグニチャ説明サブモードを終了します。

```
ips-ssp(config-sig-sig-sig)# exit
```

**ステップ 7**　String XL TCP エンジンを指定します。

```
ips-ssp(config-sig-sig)# engine string-xl-tcp
```

**ステップ 8**　サービス ポートを指定します。

```
ips-ssp(config-sig-sig-str)# service-ports 80
```

**ステップ 9**　方向を指定します。

```
ips-ssp(config-sig-sig-str)# direction to-service
```

**ステップ 10**　必要ならば、セキュリティ ポリシーに従い、**event-action** コマンドを使用してイベント アクションを変更します。デフォルトのイベント アクションは、**produce-alert** です。

**ステップ 11**　raw 正規表現をオフにします。

```
ips-ssp(config-sig-sig-str)# specify-raw-regex-string no
```

**（注）**　raw 正規表現とは、raw モード処理に使用される正規表現の構文です。エキスパート モード専用であり、Cisco IPS シグニチャ開発チーム、あるいはその監視下にある作業者のみによる使用を対象としています。String XL シグニチャは、通常の正規表現と raw 正規表現のいずれでも設定できます。

**ステップ 12**　TCP パケット内で検索する正規表現文字列を指定し、dot all をオンにします。

```
ips-ssp(config-sig-sig-str-no)# regex-string ht+p[\r].
ips-ssp(config-sig-sig-str-no)# dot-all true
```

**ステップ 13**　このシグニチャの最小一致長を指定します。stingy を設定する場合は、最小一致長のみが使用できます。

```
ips-ssp(config-sig-sig-str-no)# specify-min-match-length yes
ips-ssp(config-sig-sig-str-no-yes)# min-match-length 100
ips-ssp(config-sig-sig-str-no-yes)# exit
ips-ssp(config-sig-sig-str-no)# stingy true
```

**ステップ 14**　設定を確認できます。

```
ips-ssp(config-sig-sig-str-no)# show settings
   no
   -----------------------------------------------
      regex-string: ht+p[\r\].
      dot-all: true default: false
      end-optional: false <defaulted>
      no-case: false <defaulted>
      stingy: true default: false
      utf8: false <defaulted>
      specify-min-match-length
      -----------------------------------------------
         yes
         -----------------------------------------------
            min-match-length: 100
         -----------------------------------------------
      -----------------------------------------------
   -----------------------------------------------
ips-ssp(config-sig-sig-str-no)#
```

**ステップ 15**　検索する新しい正規表現を指定し、UTF-8 をオンにします。

```
ips-ssp(config-sig-sig-str-no)# regex-string \x5c\x31\x30\x2e\x30[\x00-\xff]+\
x2e\x31\x5c\x74\x65\x6d\x70
ips-ssp(config-sig-sig-str-no)# utf8 true
```

**ステップ 16**　設定を確認できます。

```
ips-ssp(config-sig-sig-str-no)# show settings
no
   -----------------------------------------------
      regex-string: \x5c\x31\x30\x2e\x30[\x00-\xff]+\x2e\x31\x5c\x74\x65\x6d\x70
      dot-all: true default: false
      end-optional: false <defaulted>
      no-case: false <defaulted>
      stingy: true default: false
      utf8: true default: false
      specify-min-match-length
      -----------------------------------------------
         yes
```

```
            ------------------------------------------------
               min-match-length: 100
            ------------------------------------------------
         ---------------------------------------------
      ------------------------------------------------
```

**ステップ 17**　シグニチャ定義サブモードを終了します。

```
ips-ssp(config-sig-sig-str-no)# exit
ips-ssp(config-sig-sig-str)# exit
ips-ssp(config-sig-sig)# exit
ips-ssp(config-sig)# exit
Apply Changes:?[yes]:
```

**ステップ 18**　Enter を押して変更を適用するか、**no** を入力して変更を破棄します。

### 詳細情報

- すべての String XL パラメータの一覧については、付録 B「String XL エンジン」を参照してください。
- 正規表現の構文の詳細については、付録 B「正規表現の構文」を参照してください。