CHAPTER 20

モジュラ ポリシー フレームワークの使用

この章では、モジュラ ポリシー フレームワークを使用して、サポートされている機能のセキュリティ ポリシーを作成する方法について説明します。この章では、次の内容について説明します。

モジュラ ポリシー フレームワークの詳細

モジュラ ポリシー フレームワークを使用すると、一貫性のある柔軟な方法で FWSM の機能を設定できます。たとえば、モジュラ ポリシー フレームワークを使用してタイムアウトを設定すると、すべての TCP アプリケーションにではなく、特定の TCP アプリケーションに固有に適用できます。ここでは、次の内容について説明します。

モジュラ ポリシー フレームワークでサポートされている機能

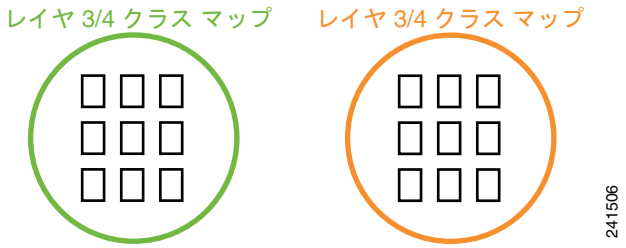
モジュラ ポリシー フレームワークでは、次の機能がサポートされています。

- TCP および UDP の接続設定、TCP シーケンス番号のランダム化、および TCP ステート バイパス：「接続制限とタイムアウトの設定」（P.21-1）および 「TCP ステート バイパスの設定」（P.21-11）を参照してください。
- アプリケーション検査：第 22 章「アプリケーション層プロトコル検査の適用」を参照してください。
- PISA 統合を使用したアプリケーション タイプの許可または拒否：「PISA 統合でのアプリケーション タイプの許可または拒否」（P.21-4）を参照してください。

モジュラ ポリシー フレームワーク設定の概要

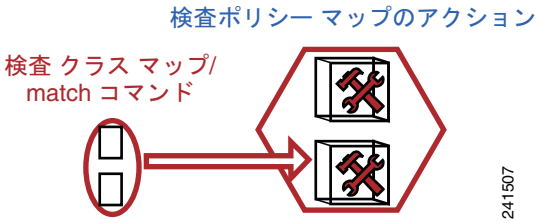
モジュラ ポリシー フレームワークの設定は、次のタスクで構成されています。

1. レイヤ 3/4 クラス マップを作成して、モジュラ ポリシー フレームワークのアクションを実行するトラフィックを識別します。たとえば、FWSM を通過するすべてのトラフィックでアクションを実行したり、10.1.1.0/24 から任意の宛先アドレスまでのトラフィックで特定のアクションだけを実行したりできます。



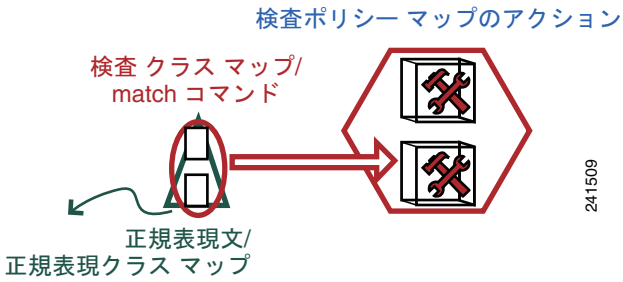
「トラフィックの識別（レイヤ 3/4 クラス マップ）」（P.20-4）を参照してください。

2. 実行するアクションの 1 つがアプリケーション検査で、一部の検査トラフィックで追加のアクションを実行する場合、検査ポリシー マップを作成します。検査ポリシーマップはトラフィックを特定し、そのトラフィックで何をするかを指定します。たとえば、本文の長さが 1000 バイトを上回るすべての HTTP 要求をドロップできます。



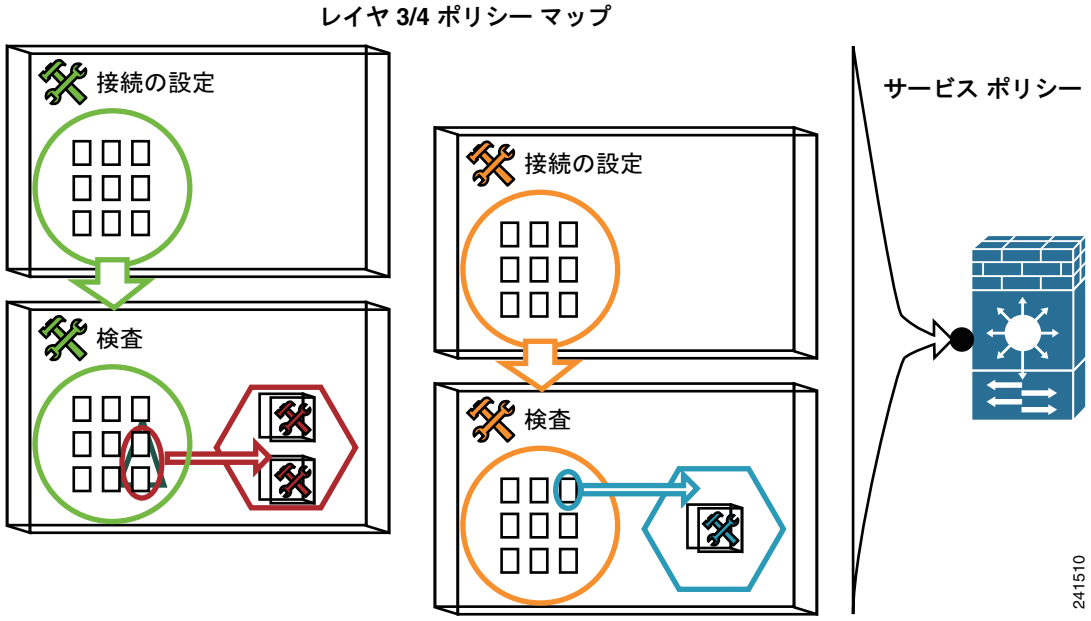
match コマンドでトラフィックを直接特定する独立した検査ポリシーマップを作成したり、再利用のために、またはより複雑な照合のために検査クラスマップを作成したりできます。「検査ポリシーマップのアクションの定義」（P.20-7）および「検査クラスマップ内のトラフィックの特定」（P.20-10）を参照してください。

3. 検査されるパケット内でテキストを正規表現と一致させる場合は、正規表現または正規表現のグループ（正規表現クラス マップ）を作成できます。トラフィックが検査ポリシーマップと一致するように定義するときに、既存の正規表現を呼び出すことができます。たとえば、「example.com」というテキストが含まれた URL を持つすべての HTTP 要求をドロップできます。



「正規表現の作成」（P.20-11）および「正規表現クラス マップの作成」（P.20-14）を参照してください。

4. レイヤ 3/4 ポリシー マップを作成して、各レイヤ 3/4 クラス マップで実行するアクションを定義します。次に、サービス ポリシーを使用して、ポリシー マップを適用するインターフェイスを決定します。



「アクションの定義（レイヤ 3/4 ポリシーマップ）」（P.20-14）および「インターフェイスへのアクションの適用（サービス ポリシー）」（P.20-20）を参照してください。

デフォルトのグローバル ポリシー

デフォルトでは、すべてのデフォルト アプリケーション検査トラフィックに一致するポリシーがコンフィギュレーションに含まれ、特定の検査がすべてのインターフェイスのトラフィックに適用されます（グローバル ポリシー）。すべての検査がデフォルトでイネーブルになっているわけではありません。適用できるグローバル ポリシーは 1 つのみです。このため、グローバル ポリシーを変更する場合は、デフォルトのポリシーを編集するか、ディセーブルにして新しいポリシーを適用する必要があります。（インターフェイス ポリシーはグローバル ポリシーに優先します）。

デフォルト ポリシー コンフィギュレーションには、次のコマンドが含まれます。

```
class-map inspection_default
 match default-inspection-traffic
policy-map global_policy
 class inspection_default
  inspect dns maximum-length 512
  inspect ftp
  inspect h323 h225
  inspect h323 ras
  inspect netbios
  inspect rsh
  inspect skinny
  inspect sqlnet
  inspect sunrpc
  inspect tftp
  inspect sip
  inspect xdmcp
service-policy global_policy global
```

（注）　デフォルトのクラスマップで使用される特別な **match default-inspection-traffic** コマンドの詳細については、「特定の機能アクションの非互換性」（P.20-17）を参照してください。

トラフィックの識別（レイヤ 3/4 クラス マップ）

レイヤ 3/4 クラス マップは、アクションを適用するレイヤ 3 およびレイヤ 4 のトラフィックを指定します。レイヤ 3/4 ポリシー マップそれぞれに、複数のレイヤ 3/4 クラス マップを作成できます。

ここでは、次の内容について説明します。

デフォルト クラス マップ

設定には、デフォルト グローバル ポリシー内で FWSM が使用するデフォルトのレイヤ 3/4 クラス マップなど、内部作成された多数のデフォルト クラス マップが含まれています。これは、**inspection_default** と呼ばれ、デフォルトの検査トラフィックと一致します。

```
class-map inspection_default
 match default-inspection-traffic
```

（注）　デフォルトのクラスマップで使用される特別な **match default-inspection-traffic** コマンドの詳細については、「特定の機能アクションの非互換性」（P.20-17）を参照してください。

デフォルトのコンフィギュレーションに存在する別のクラス マップは、class-default と呼ばれ、これはすべてのトラフィックと一致します。

```
class-map class-default
 match any
```

このクラス マップは、すべてのレイヤ 3/4 ポリシー マップの末尾にあり、基本的に他のすべてのトラフィックに対してアクションを実行しないように FWSM に指示します。独自の **match any** クラス マップを作成するのではなく、必要に応じて class-default クラス マップを使用できます。

デフォルト クラス マップには検査クラス マップも含まれます。

デフォルト クラス マップおよびユーザ作成のクラス マップを表示するには、**show running-config all class-map** コマンドを入力します。

最大クラス マップ数

すべてのタイプのクラス マップの最大数は、シングル モードでは 255 個、マルチ モードではコンテキストごとに 255 個です。クラス マップには、次のタイプが含まれます。

- レイヤ 3/4 クラス マップ
- 検査クラスマップ
- 正規表現クラスマップ
- 検査ポリシーマップ下で直接使用される **match** コマンド

この制限にはすべてのタイプのデフォルト クラス マップも含まれます。「デフォルト クラス マップ」（P.20-4）を参照してください。

通過トラフィック用のレイヤ 3/4 クラスマップの作成

レイヤ 3/4 クラスマップでは、プロトコル、ポート、IP アドレス、およびレイヤ 3 またはレイヤ 4 の他の属性に基づいてトラフィックを照合します。

レイヤ 3/4 クラス マップを定義する手順は、次のとおりです。

ステップ 1　次のコマンドを入力して、レイヤ 3/4 クラスマップを作成します。

```
hostname(config)# class-map class_map_name
hostname(config-cmap)#
```

class_map_name は、最大 40 文字の文字列です。「class-default」という名前は予約されています。すべてのタイプのクラスマップで同じ名前スペースが使用されるため、別のタイプのクラスマップですでに使用されている名前は再度使用できません。CLI はクラスマップ コンフィギュレーション モードに移行します。

ステップ 2　（任意）次のコマンドを入力して、クラスマップの説明を追加します。

```
hostname(config-cmap)# description string
```

ステップ 3　次の特性のいずれかと照合して、クラスに含めるトラフィックを定義します。特に指定がない場合、クラスマップに含めることができる **match** コマンドは 1 つだけです。

- 任意のトラフィック：クラスマップは、すべてのトラフィックと照合されます。

  ```
  hostname(config-cmap)# match any
  ```

- アクセス リスト：クラスマップは、拡張アクセス リストで指定されたトラフィックと照合されます。FWSM がトランスペアレント ファイアウォール モードで動作している場合は、EtherType アクセス リストを使用できます。

  ```
  hostname(config-cmap)# match access-list access_list_name
  ```

  アクセス リストの作成の詳細については、「拡張アクセス リストの追加」（P.13-6）または「EtherType アクセス リストの追加」（P.13-9）を参照してください。

  NAT を使用するアクセス リストの作成の詳細については、「NAT 使用時のアクセス リスト用の IP アドレス」（P.13-3）を参照してください。

- TCP または UDP 宛先ポート：クラスマップは、単一ポートまたは一定範囲の連続ポートと一致します。

  ```
  hostname(config-cmap)# match port {tcp | udp} {eq port_num | range port_num port_num}
  ```

ヒント　複数の非連続ポートを使用するアプリケーションに対しては、**match access-list** コマンドを使用して、各ポートと一致する ACE を定義します。

  指定できるポートのリストについては、「TCP ポートおよび UDP ポート」（P.E-12）を参照してください。

  たとえば、ポート 80 の TCP パケット（HTTP）と一致させるには、次のコマンドを入力します。

  ```
  hostname(config-cmap)# match tcp eq 80
  ```

- 検査用のデフォルト トラフィック：クラスマップは、FWSM が検査できるすべてのアプリケーションで使用されるデフォルトの TCP および UDP ポートと一致します。

  ```
  hostname(config-cmap)# match default-inspection-traffic
  ```

  デフォルト グローバル ポリシーで使用されるこのコマンドは、ポリシーマップで使用されると、トラフィックの宛先ポートに基づいて各パケットに正しい検査を適用する特別な CLI ショートカットです。たとえば、宛先がポート 69 の UDP トラフィックが FWSM に到達すると、FWSM は TFTP 検査を適用し、宛先がポート 21 の TCP トラフィックが到着すると、FWSM は FTP 検査を適用します。そのため、この場合に限って同じクラスマップに複数の検査を設定できます（他の検査とともに設定可能な WAAS 検査を除きます。アクションの組み合わせの詳細については、「特定の機能アクションの非互換性」（P.20-17）を参照してください）。通常、FWSM は、ポート番号を使用して適用する検査を決定しないため、標準以外のポートなどにも柔軟に検査を適用できます。

  デフォルト ポートのリストについては、「デフォルトの検査ポリシー」（P.22-4）を参照してください。FWSM には、デフォルトの検査トラフィックに一致して、すべてのインターフェイス上のトラフィックに共通検査を適用するデフォルト グローバル ポリシーが含まれます。**match default-inspection-traffic** コマンドにポートが含まれているすべてのアプリケーションが、ポリシーマップでデフォルトでイネーブルになっているわけではありません。

  match access-list コマンドを **match default-inspection-traffic** コマンドとともに指定すると、一致するトラフィックを絞り込むことができます。**match default-inspection-traffic** コマンドによって一致させるポートおよびプロトコルが指定されるため、アクセス リストのポートまたはプロトコルは無視されます。

次に、**class-map** コマンドの例を示します。

```
hostname(config)# access-list udp permit udp any any
hostname(config)# access-list tcp permit tcp any any
hostname(config)# access-list host_foo permit ip any 10.1.1.1 255.255.255.255

hostname(config)# class-map all_udp
hostname(config-cmap)# description "This class-map matches all UDP traffic"
hostname(config-cmap)# match access-list udp

hostname(config-cmap)# class-map all_tcp
hostname(config-cmap)# description "This class-map matches all TCP traffic"
hostname(config-cmap)# match access-list tcp

hostname(config-cmap)# class-map all_http
hostname(config-cmap)# description "This class-map matches all HTTP traffic"
hostname(config-cmap)# match port tcp eq http

hostname(config-cmap)# class-map to_server
hostname(config-cmap)# description "This class-map matches all traffic to server 10.1.1.1"
hostname(config-cmap)# match access-list host_foo
```

アプリケーション検査の特別なアクションの設定（検査ポリシーマップ）

モジュラ ポリシー フレームワークを使用すると、多くのアプリケーション検査に対して特別なアクションを設定できます。レイヤ 3/4 ポリシーマップでインスペクション エンジンをイネーブルにする場合は、検査ポリシーマップで定義されるアクションを必要に応じてイネーブルにすることもできま

す。検査ポリシーマップが、検査アクションを定義したレイヤ 3/4 クラスマップ内のトラフィックと一致すると、トラフィックのそのサブセットが指定したとおりに動作します（たとえば、ドロップやレート制限など）。

ここでは、次の内容について説明します。

検査ポリシー マップの概要

検査ポリシーマップをサポートするアプリケーションのリストについては、「インスペクション エンジンの概要」（P.22-2）を参照してください。

検査ポリシーマップは、次に示す要素の 1 つ以上で構成されています。検査ポリシーマップで使用可能な実際のオプションは、アプリケーションに応じて決まります。

- トラフィック照合コマンド：検査ポリシーマップで直接トラフィック照合コマンドを定義して、アプリケーションのトラフィックを、URL 文字列などのアプリケーションに固有の基準と照合できます。一致した場合にはアクションをイネーブルにします。
  - 一部のトラフィック照合コマンドでは、正規表現を指定してパケット内部のテキストを照合できます。ポリシーマップを設定する前に、正規表現クラスマップ内で、正規表現を単独またはグループで作成およびテストしておいてください。
- 検査クラスマップ：（すべてのアプリケーションで使用できるわけではありません。サポートされるアプリケーションのリストについては、CLI ヘルプを参照してください）。検査クラスマップには、アプリケーション トラフィックを URL 文字列などのアプリケーション固有の基準と照合するトラフィック照合コマンドが含まれています。その後、ポリシーマップ内のクラスマップを特定し、アクションをイネーブルにします。クラスマップを作成することと、検査ポリシーマップ内で直接トラフィック照合を定義することの違いは、より複雑な一致基準を作成できる点と、クラスマップを再使用できる点です。
  - 一部のトラフィック照合コマンドでは、正規表現を指定してパケット内部のテキストを照合できます。ポリシーマップを設定する前に、正規表現クラスマップ内で、正規表現を単独またはグループで作成およびテストしておいてください。
- パラメータ：パラメータは、インスペクション エンジンの動作に影響します。

（注）　**policy-map type inspect esmtp _default_esmtp_map** などのデフォルトの検査ポリシー マップがあります。これらのデフォルト ポリシーマップは、**inspect** *protocol* コマンドで暗黙的に作成されます。たとえば、**inspect esmtp** は暗黙的にポリシー マップ「_default_esmtp_map」を使用します。**show running-config all policy-map** コマンドを使用すると、すべてのデフォルト ポリシー マップを表示できます。

検査ポリシーマップのアクションの定義

レイヤ 3/4 ポリシーマップでインスペクション エンジンをイネーブルにする場合は、検査ポリシーマップで定義されるアクションを必要に応じてイネーブルにすることもできます。

検査ポリシー マップを作成する手順は、次のとおりです。

ステップ 1　（任意）「検査クラスマップ内のトラフィックの特定」（P.20-10）の説明に従って、検査クラスマップを作成します。または、ポリシーマップ内でトラフィックを直接特定できます。

ステップ 2　検査ポリシーマップを作成するには、次のコマンドを入力します。

```
hostname(config)# policy-map type inspect application policy_map_name
hostname(config-pmap)#
```

検査ポリシーマップをサポートするアプリケーションのリストについては、「アプリケーション検査の設定」（P.22-7）を参照してください。

policy_map_name 引数は、最大 40 文字のポリシーマップ名です。すべてのタイプのポリシー マップが同じネーム スペースを使用しているため、他のタイプのポリシー マップですでに使用されている名前は再使用できません。CLI はポリシーマップ コンフィギュレーション モードに入ります。

ステップ 3　一致トラフィックにアクションを適用するには、次の手順に従います。

a.　次のいずれかの方法を使用して、アクションを実行するトラフィックを指定します。

- 次のコマンドを入力して、「検査クラスマップ内のトラフィックの特定」（P.20-10）で作成した検査クラスマップを指定します。

  ```
  hostname(config-pmap)# class class_map_name
  hostname(config-pmap-c)#
  ```

  すべてのアプリケーションが検査クラスマップをサポートするわけではありません。

- 第 22 章「アプリケーション層プロトコル検査の適用」でアプリケーションごとに説明されている **match** コマンドのいずれかを使用して、ポリシー マップで直接トラフィックを指定します。**match not** コマンドを使用すると、**match not** コマンドの基準に一致するどのトラフィックにもアクションは適用されません。

b.　次のコマンドを入力して、一致したトラフィックに対して実行するアクションを指定します。

```
hostname(config-pmap-c)# {[drop | drop-connection | mask | reset] [log] | log}
```

それぞれのアプリケーションですべてのオプションを設定できるわけではありません。アプリケーションに固有の他のアクションも適用可能な場合があります。使用できる正確なオプションについては、第 22 章「アプリケーション層プロトコル検査の適用」を参照してください。

drop キーワードを指定すると、一致するすべてのパケットをドロップします。

drop-connection キーワードを指定すると、パケットをドロップし、接続を閉じます。

mask キーワードを指定すると、パケットの一致部分をマスクします。

reset キーワードを指定すると、パケットをドロップして接続を閉じ、サーバとクライアントの両方またはいずれかに TCP リセットを送信します。

log キーワードを指定すると、システム ログ メッセージを送信します。このキーワードは単独で、または他のキーワードのいずれかと一緒に使用できます。

（注）　ポリシー マップでは、複数の **class** コマンドまたは **match** コマンドを指定できます。

パケットが複数の異なる **match** コマンドまたは **class** コマンドに一致する場合、FWSM がアクションを適用する順序は、FWSM の内部ルールで決定されます。ポリシー マップに追加された順序ではありません。内部ルールは、アプリケーション タイプ、およびパケット解析の論理進行によって決まり、ユーザが設定できるものではありません。たとえば、HTTP トラフィックの場合、Request Method フィールドの解析は Header Host Length フィールドの解析よりも先行します。Request Method フィー

ルドのアクションは、Header Host Length フィールドのアクションよりも先に発生します。たとえば、次の match コマンドは任意の順序で入力できますが、**match request method get** コマンドが最初に一致します。

```
match request header host length gt 100
  reset
match request method get
  log
```

アクションがパケットをドロップすると、検査ポリシーマップではそれ以降のアクションは実行されません。たとえば、最初のアクションが接続のリセットである場合、そのアクション以降は、いずれの **match** コマンドまたは **class** コマンドにも一致しません。最初のアクションがパケットのロギングである場合、接続のリセットなどの別のアクションは実行可能です。（同じ **match** または **class** コマンドで、**reset**（または **drop-connection** など）と **log** の両方のアクションを設定できます。この場合パケットは、指定された照合でリセットされる前にログに記録されます）。

パケットが複数の同一 **match** コマンドまたは **class** コマンドに一致する場合は、ポリシー マップに出現する順序で一致します。たとえば、ヘッダー長が 1001 のパケットの場合、次の最初のコマンドに一致してログに記録され、その後、2 番めのコマンドに一致してリセットされます。2 つの **match** コマンドの順序を逆にすると、パケットはドロップされ、2 番めの **match** コマンドに一致する前に接続がリセットされます。ログに記録されることはありません。

```
match request header length gt 100
  log
match request header length gt 1000
  reset
```

あるクラス マップが、別のクラス マップ、またはクラス マップ内でプライオリティが最も低い **match** コマンドに基づいた **match** コマンドと同じタイプであると判断されます（プライオリティは内部ルールに基づく）。クラス マップのタイプが、他のクラス マップと同じプライオリティの最も低いタイプの **match** コマンドである場合、クラス マップはポリシー マップに追加された順序に基づいて照合されます。各クラス マップのプライオリティの最も低いコマンドが異なる場合、プライオリティが高い **match** コマンドを持つクラス マップが最初に照合されます。たとえば、次の 3 つのクラス マップには 2 つのタイプの **match** コマンドが含まれています。**match content length**（優先度が高い）と **match content type**（優先度が低い）です。sip3 クラス マップには両方のコマンドが含まれていますが、最低優先度コマンド **match content type** に応じてランク付けされます。sip1 クラス マップには最高優先度のコマンドが含まれているため、ポリシー マップの順序に関係なく最初に一致します。sip3 クラス マップは、同じく **match content type** コマンドを含む sip2 クラス マップと同じ優先度としてランク付けされます。これらはポリシー マップの順序に従って、sip3、sip2 の順に一致します。

```
class-map inspect type sip match-all sip1
  match content length gt 1000
class-map inspect type sip match-all sip2
  match content type sdp
class-map inspect type sip match-all sip3
  match content length gt 1000
  match content type sdp

policy-map type inspect sip sip
  class sip3
    log
  class sip2
    log
  class sip1
    log
```

ステップ 4　検索エンジンに影響するパラメータを設定するには、次のコマンドを入力します。

```
hostname(config-pmap)# parameters
```

```
hostname(config-pmap-p)#
```

CLI は、パラメータ コンフィギュレーション モードを開始します。各アプリケーションで使用可能なパラメータについては、第 22 章「アプリケーション層プロトコル検査の適用」を参照してください。

次に、HTTP 検査ポリシー マップおよび関連するクラス マップの例を示します。このポリシー マップは、レイヤ 3/4 ポリシー マップによってアクティブになり、サービス ポリシーによってイネーブルになります。

```
hostname(config)# regex url_example example\.com
hostname(config)# regex url_example2 example2\.com
hostname(config)# class-map type regex match-any URLs
hostname(config-cmap)# match regex url_example
hostname(config-cmap)# match regex url_example2

hostname(config-cmap)# class-map type inspect http match-all http-traffic
hostname(config-cmap)# match req-resp content-type mismatch
hostname(config-cmap)# match request body length gt 1000
hostname(config-cmap)# match not request uri regex class URLs

hostname(config-cmap)# policy-map type inspect http http-map1
hostname(config-pmap)# class http-traffic
hostname(config-pmap-c)# drop-connection log
hostname(config-pmap-c)# match req-resp content-type mismatch
hostname(config-pmap-c)# reset log
hostname(config-pmap-c)# parameters
hostname(config-pmap-p)# protocol-violation action log

hostname(config-pmap-p)# policy-map test
hostname(config-pmap)# class test (a Layer 3/4 class map not shown)
hostname(config-pmap-c)# inspect http http-map1

hostname(config-pmap-c)# service-policy test interface outside
```

検査クラスマップ内のトラフィックの特定

このタイプのクラスマップを使用して、アプリケーション固有の基準と照合できます。たとえば、HTTP トラフィックの場合、特定の URL と一致させることができます。

（注）　すべてのアプリケーションが検査クラスマップをサポートするわけではありません。サポートされているアプリケーションのリストについては、CLI ヘルプを参照してください。

クラス マップは、（match-all クラス マップ内で）複数のトラフィック照合をグループ化するか、または（match-any クラス マップ内で）照合のリストのいずれかを照合できます。クラス マップを作成することと、検査ポリシーでトラフィック照合を直接定義することとの相違は、クラス マップでは、複数の match コマンドをグループ化でき、クラス マップを再使用できることです。このクラス マップで指定するトラフィックの場合、検査ポリシー マップでアクション（接続のドロップ、リセット、ログへの記録など）を指定できます。タイプの異なるトラフィックで異なるアクションを実行する場合は、ポリシーマップで直接トラフィックを指定してください。

クラス マップ（レイヤ 3/4、検査、および正規表現）の最大数はシングル モードまたはマルチ モードの各コンテキストで 255 です。この制限には、デフォルト クラス マップも含まれます。「デフォルト クラス マップ」（P.20-4）を参照してください。

検査クラス マップを定義する手順は、次のとおりです。

ステップ 1　（任意）正規表現に基づいた照合を行う場合は、「正規表現の作成」（P.20-11）および「正規表現クラス マップの作成」（P.20-14）を参照してください。

ステップ 2　次のコマンドを入力して、クラスマップを作成します。

```
hostname(config)# class-map type inspect application [match-all] class_map_name
hostname(config-cmap)#
```

application は、検査対象のアプリケーションです。サポートされるアプリケーションのリストについては、CLI ヘルプまたは第 22 章「アプリケーション層プロトコル検査の適用」を参照してください。

class_map_name 引数は、最大 40 文字のクラスマップ名です。

match-all キーワードはデフォルトです。トラフィックがクラスマップと一致するには、すべての基準と一致する必要があることを指定します。

CLI がクラスマップ コンフィギュレーション モードに入り、1 つ以上の **match** コマンドを入力できます。

ステップ 3　（任意）クラスマップに説明を追加するには、次のコマンドを使用します。

```
hostname(config-cmap)# description string
```

ステップ 4　アプリケーションで使用可能な 1 つ以上の **match** コマンドを入力して、クラスに含めるトラフィックを定義します。

クラスマップと照合しないトラフィックを指定するには、**match not** コマンドを使用します。たとえば、**match not** コマンドで文字列「example.com」を指定すると、「example.com」が含まれるすべてのトラフィックはクラスマップと照合されません。

各アプリケーションで使用可能な **match** コマンドについては、第 22 章「アプリケーション層プロトコル検査の適用」を参照してください。

次に、すべての基準と一致する必要がある HTTP クラス マップを作成する例を示します。

```
hostname(config-cmap)# class-map type inspect http match-all http-traffic
hostname(config-cmap)# match req-resp content-type mismatch
hostname(config-cmap)# match request body length gt 1000
hostname(config-cmap)# match not request uri regex class URLs
```

正規表現の作成

正規表現は、ストリングそのものとしてテキスト ストリングと文字どおりに照合することも、*metacharacters* を使用してテキスト ストリングの複数のバリアントと照合することもできます。正規表現を使用して特定のアプリケーション トラフィックの内容と照合できます。たとえば、HTTP パケット内部の URL 文字列と照合できます。

Ctrl キーを押した状態で **V** キーを押すと、CLI において、疑問符（?）やタブなどの特殊文字をすべてエスケープできます。たとえば、設定で **d?g** を指定するには、**d[Ctrl+V]g** と入力します。

正規表現をパケットと照合する場合のパフォーマンスへの影響については、『*Catalyst 6500 Series Switch and Cisco 7600 Series Router Firewall Services Module Command Reference*』の **regex** コマンドを参照してください。

（注）　最適化のために、FWSM では、難読化解除された URL が検索されます。難読化解除では、複数のスラッシュ（/）が単一のスラッシュに圧縮されます。「http://」などの、一般的に 2 つのスラッシュが使用されるストリングでは、代わりに「http:/」を検索してください。

表 20-1 に、特別な意味を持つメタ文字の一覧を示します。

表 20-1　　regex メタ文字

文字	説明	注記
.	ドット	任意の単一文字と一致します。たとえば、**d.g** は dog、dag、dtg、doggonnit など、これらの文字が含まれているすべての単語と一致します。
(*exp***)**	サブ表現	サブ表現は、文字を周囲の文字から分離して、サブ表現に他のメタ文字を使用できるようにします。たとえば、**d(o|a)g** は dog および dag と一致しますが、do や ag とは一致しません。また、サブ表現を繰り返し限定作用素とともに使用して、繰り返す文字を区別できます。たとえば、**ab(xy){3}z** は、abxyxyxyz に一致します。
|	代替	このメタ文字によって区切られている複数の表現のいずれかと一致します。たとえば、**dog|cat** は dog または cat と一致します。
?	疑問符	直前の表現が 0 または 1 個存在することを示す修飾子。たとえば、**lo?se** は lse または lose と一致します。 **（注）**　**Ctrl+V** を入力してから疑問符を入力しないと、ヘルプ機能が呼び出されます。
*	アスタリスク	直前の表現が 0、1、または任意の個数存在することを示す修飾子。たとえば、**lo*se** は、lse、lose、loose などに一致します。
+	プラス	直前の表現が少なくとも 1 個存在することを示す修飾子。たとえば、**lo+se** は lose および loose と一致しますが、lse とは一致しません。
{*x*} または {*x*,}	最小繰り返し限定作用素	少なくとも *x* 回繰り返します。たとえば、**ab(xy){2,}z** は、abxyxyz や abxyxyxyz などに一致します。
[*abc***]**	文字クラス	カッコ内の任意の文字と一致します。たとえば、**[abc]** は a、b、または c と一致します。
[^*abc***]**	否定文字クラス	角カッコに含まれていない単一文字と一致します。たとえば、**[^abc]** は a、b、または c 以外の任意の文字と一致します。**[^A-Z]** は、大文字でない任意の単一文字と一致します。
[*a-c***]**	文字範囲クラス	範囲内の任意の文字と一致します。**[a-z]** は、任意の小文字と一致します。文字と範囲を混合できます。**[abcq-z]** は、a、b、c、q、r、s、t、u、v、w、x、y、z と一致し、**[a-cq-z]** も同じです。 ダッシュ（-）文字は、角カッコ内の最後または最初の文字である場合にだけリテラルになります（**[abc-]** または **[-abc]**）。

表 20-1　regex メタ文字 （続き）

文字	説明	注記
""	引用符	文字列の末尾または先頭のスペースを保持します。たとえば、**" test"** は、一致を検索する場合に先頭のスペースを保持します。
^	キャレット	行の先頭を指定します。
\	エスケープ文字	メタ文字とともに使用すると、リテラル文字と一致します。たとえば、**\[** は左の角カッコと一致します。
char	文字	文字がメタ文字でない場合は、リテラル文字と一致します。
\r	復帰	復帰 0x0d と一致します。
\n	改行	改行 0x0a と一致します。
\t	タブ	タブ 0x09 と一致します。
\f	改ページ	フォーム フィード 0x0c と一致します。
\x*NN*	エスケープされた 16 進数	16 進数（厳密に 2 桁）を使用した ASCII 文字と一致します。
\NNN	エスケープされた 8 進数	8 進数（厳密に 3 桁）としての ASCII 文字と一致します。たとえば、文字 040 はスペースを表します。

正規表現をテストおよび作成する手順は、次のとおりです。

ステップ 1　正規表現をテストして、一致するはずの対象と一致することを確認するには、次のコマンドを入力します。

```
hostname(config)# test regex input_text regular_expression
```

input_text 引数は、正規表現を使用して照合する、長さが最大で 201 文字の文字列です。

regular_expression 引数の長さは、最大 100 文字です。

Ctrl+V を使用して、CLI の特殊文字をすべてエスケープします。たとえば、**test regex** コマンドの入力文字にタブを指定するには、**test regex "test[Ctrl+V Tab]" "test\t"** と入力する必要があります。

正規表現が入力テキストと一致する場合は、次のメッセージが表示されます。

```
INFO: Regular expression match succeeded.
```

正規表現が入力テキストと一致しない場合は、次のメッセージが表示されます。

```
INFO: Regular expression match failed.
```

ステップ 2　テスト後に正規表現を追加するには、次のコマンドを入力します。

```
hostname(config)# regex name regular_expression
```

name 引数の長さは、最大 40 文字です。

regular_expression 引数の長さは、最大 100 文字です。

次に、検査ポリシー マップで使用する 2 つの正規表現を作成する例を示します。

```
hostname(config)# regex url_example example\.com
hostname(config)# regex url_example2 example2\.com
```

正規表現クラス マップの作成

正規表現クラスマップで、1 つ以上の正規表現を指定します。正規表現クラスマップを使用して、特定のトラフィックの内容を照合できます。たとえば、HTTP パケット内の URL 文字列の照合が可能です。

クラス マップ（レイヤ 3/4、検査、および正規表現）の最大数はシングル モードまたはマルチ モードの各コンテキストで 255 です。この制限には、デフォルト クラス マップも含まれます。「デフォルト クラス マップ」（P.20-4）を参照してください。

正規表現クラス マップを作成する手順は、次のとおりです。

ステップ 1　「正規表現の作成」の項の説明に従って、正規表現を 1 つ以上作成します。

ステップ 2　次のコマンドを入力して、クラスマップを作成します。

```
hostname(config)# class-map type regex match-any class_map_name
hostname(config-cmap)#
```

class_map_name は、最大 40 文字の文字列です。「class-default」という名前は予約されています。すべてのタイプのクラスマップで同じ名前スペースが使用されるため、別のタイプのクラスマップですでに使用されている名前は再度使用できません。

match-any キーワードにより、トラフィックが少なくとも 1 つの正規表現と一致する場合には、そのトラフィックがクラスマップと一致するように指定します。

CLI はクラスマップ コンフィギュレーション モードに移行します。

ステップ 3　（任意）次のコマンドを入力して、クラスマップの説明を追加します。

```
hostname(config-cmap)# description string
```

ステップ 4　正規表現ごとに次のコマンドを入力して、クラスマップに含める正規表現を指定します。

```
hostname(config-cmap)# match regex regex_name
```

次に、2 つの正規表現を作成し、これを正規表現クラス マップに追加する例を示します。トラフィックに「example.com」または「example2.com」というストリングが含まれている場合、このトラフィックはクラス マップと一致しています。

```
hostname(config)# regex url_example example\.com
hostname(config)# regex url_example2 example2\.com
hostname(config)# class-map type regex match-any URLs
hostname(config-cmap)# match regex url_example
hostname(config-cmap)# match regex url_example2
```

アクションの定義（レイヤ 3/4 ポリシーマップ）

この項では、レイヤ 3/4 ポリシーマップを作成して、アクションをレイヤ 3/4 クラスマップに関連付ける方法について説明します。ここでは、次の内容について説明します。

レイヤ 3/4 ポリシー マップの詳細

ここでは、レイヤ 3/4 ポリシー マップの動作方法について説明します。内容は次のとおりです。

ポリシー マップに関する注意事項

ポリシーマップを使用する場合は、次のガイドラインを参考にしてください。

- 各インターフェイスには、ポリシーマップを 1 つだけ割り当てることができます。（ただし、設定では最大 64 のポリシーマップを作成できます）。
- 同一のポリシーマップを複数のインターフェイスに適用できます。
- 1 つのレイヤ 3/4 ポリシーマップで複数のレイヤ 3/4 クラスマップを指定できます。
- クラスマップごとに、1 つ以上の機能タイプから複数のアクションを割り当てることができます（サポートされている場合）。「特定の機能アクションの非互換性」（P.20-17）を参照してください。

機能の方向性

アクションは、サービス ポリシーがインターフェイスに適用されているかグローバルに適用されているかに応じて、双方向的または単一方向的にトラフィックに適用されます。サービス ポリシーがインターフェイスに適用されている場合、すべての機能は双方向になります。トラフィックが双方向のクラス マップと一致すると、ポリシー マップの適用先であるインターフェイスに発着するすべてのトラフィックが影響を受けます。グローバル ポリシーを使用する場合、すべての機能は単方向です。機能が単一インターフェイスに適用されるときは通常双方向であっても、グローバルに適用されるときは各インターフェイスの入力に対してのみ適用されます。ポリシーはすべてのインターフェイスに適用されるため、ポリシーは双方向に適用され、この場合の双方向性は冗長になります。

ポリシー マップ内の一致機能に関する注意事項

ポリシー マップ内でパケットをクラス マップと一致させる方法の注意事項は、次のとおりです。

- パケットは、機能タイプごとにポリシー マップ内のクラス マップ 1 つのみと一致します。
- パケットが機能タイプのクラス マップと一致する場合、FWSM はその機能タイプについて後続のクラス マップと照合しません。
- ただし、パケットが別の機能タイプの以降のクラス マップと一致すると、FWSM は以降のクラス マップのアクションを適用します（サポートされている場合）。サポートされていない組み合わせの詳細については、「特定の機能アクションの非互換性」（P.20-17）を参照してください。

たとえば、パケットが接続制限についてのクラス マップと一致し、さらにアプリケーション検査についてのクラス マップとも一致する場合は、両方のクラス マップ アクションが適用されます。

パケットがアプリケーション検査のクラス マップと一致し、アプリケーション検査を含む別のクラス マップとも一致した場合、2 番めのクラス マップのアクションは適用されません。

（注）　アプリケーション検査には複数の検査タイプが含まれ、上記の照合ガイドラインの観点では、各検査タイプはそれぞれ独立した機能となります。

複数の機能アクションが適用される順序

さまざまなタイプのアクションがポリシー マップ内で実行される順序は、アクションがポリシー マップ内に出現する順序とは関係ありません。アクションは次の順序で実行されます。

1. TCP と UDP の接続設定、および TCP ステート バイパス
2. アプリケーション検査（複数タイプ）

   トラフィック クラスが複数検査の対象として分類されるときに適用されるアプリケーション検査の順序を次に示します。同じトラフィックに適用できる検査タイプは 1 つだけです。WAAS 検査は例外です。同じトラフィックの他の検査と一緒に適用できるためです。詳細については、「特定の機能アクションの非互換性」（P.20-17）を参照してください。

   a. CTIQBE
   b. DNS
   c. FTP
   d. GTP
   e. H323
   f. HTTP
   g. ICMP
   h. ICMP エラー
   i. ILS
   j. MGCP
   k. NetBIOS
   l. PPTP
   m. Sun RPC
   n. RSH
   o. RTSP
   p. SIP
   q. Skinny
   r. SMTP
   s. SNMP
   t. SQL*Net
   u. TFTP
   v. XDMCP
   w. DCERPC
3. PISA 統合でのアプリケーション タイプの許可または拒否

特定の機能アクションの非互換性

一部の機能は同じトラフィックに対して相互に互換性がありません。たとえば、同じトラフィックセットに対して PISA 統合と検査は設定できません。また、ほとんどの検査は別の検査と組み合わせられないため、同じトラフィックに複数の検査を設定しても、FWSM は 1 つの検査だけを適用します。この場合、適用される機能は、「複数の機能アクションが適用される順序」（P.20-16）で示されているリストの中の高プライオリティ機能となります。

各機能の互換性については、その機能を扱っている章または項を参照してください。

（注）　**match default-inspection-traffic** コマンドは、デフォルトのグローバル ポリシーで使用されますが、すべての検査のデフォルト ポートと一致する特別な CLI ショートカットです。ポリシーマップで使用すると、このクラスマップでは、トラフィックの宛先ポートに基づいて、各パケットに正しい検査が適用されます。たとえば、宛先がポート 69 の UDP トラフィックが FWSM に到達すると、FWSM は TFTP 検査を適用し、宛先がポート 21 の TCP トラフィックが到着すると、FWSM は FTP 検査を適用します。そのため、この場合に限って同じクラスマップに複数の検査を設定できます。通常、FWSM は、ポート番号を使用して適用する検査を決定しないため、標準以外のポートなどにも柔軟に検査を適用できます。

誤った設定例は、同じポリシーマップに複数の検査を設定しても、default-inspection-traffic ショートカットを使用しないことです。例 20-1 では、ポート 21 宛てのトラフィックが、FTP 検査と HTTP 検査の両方に誤って設定されています。例 20-2 では、ポート 80 宛てのトラフィックが、FTP 検査と HTTP 検査の両方に誤って設定されています。どちらの誤った設定例の場合も、FTP 検査だけが適用されています。これは、適用された検査の順序では、FTP が HTTP よりも先になるためです。

例 20-1　　FTP パケットの誤設定（HTTP 検査も設定されている）

```
class-map ftp
   match port tcp 21
class-map http
   match port tcp 21
policy-map test
   class ftp
     inspect ftp
   class http
     inspect http
```

例 20-2　　HTTP パケットの誤設定（FTP 検査も設定されている）

```
class-map ftp
   match port tcp 80
class-map http
   match port tcp 80
policy-map test
   class http
     inspect http
   class ftp
     inspect ftp
```

複数のポリシー マップの一致機能に関する注意事項

TCP および UDP トラフィック（およびステートフル ICMP 検査がイネーブルの場合は ICMP）の場合、サービス ポリシーはトラフィック フローに対して作用し、個々のパケットに限定されません。トラフィックが、1 つのインターフェイスのポリシーで定義されている機能に一致する既存の接続の一部である場合、そのトラフィック フローを別のインターフェイスのポリシーにある同じ機能と照合することはできません。最初のポリシーのみが使用されます。

たとえば、HTTP トラフィックが、HTTP トラフィックを検査する内部インターフェイスのポリシーと一致するときに、HTTP 検査用の外部インターフェイスに別のポリシーがある場合、そのトラフィックが外部インターフェイスの出力側でも検査されることはありません。同様に、その接続のリターン トラフィックが外部インターフェイスの入力ポリシーによって検査されたり、内部インターフェイスの出力ポリシーによって検査されたりすることもありません。

ステートフル ICMP 検査をイネーブルにしない場合の ICMP のように、フローとして扱われないトラフィックの場合は、リターン トラフィックを戻り側のインターフェイスの別のポリシーマップと照合できます。たとえば、内部および外部インターフェイスで接続制限を設定し、内部ポリシーでは最大接続数を 2000 に、外部ポリシーでは最大接続数を 3000 に設定した場合、ステートフルではない ping は着信の場合よりも発信の場合により低いレベルで拒否される可能性があります。

デフォルトのレイヤ 3/4 ポリシー マップ

コンフィギュレーションには、デフォルト グローバル ポリシーで FWSM が使用する、デフォルトのレイヤ 3/4 ポリシー マップが含まれています。これは **global_policy** と呼ばれ、デフォルトの検査トラフィックで検査を実行します。適用できるのは 1 つのグローバル ポリシーだけなので、グローバル ポリシーを変更する場合は、デフォルト ポリシーを再設定するか、またはデフォルト ポリシーをディセーブルにして新しいポリシーを適用する必要があります。

デフォルトのポリシー マップ コンフィギュレーションには、次のコマンドが含まれます。

```
policy-map global_policy
 class inspection_default
  inspect dns maximum-length 512
  inspect ftp
  inspect h323 h225
  inspect h323 ras
  inspect netbios
  inspect rsh
  inspect skinny
  inspect sqlnet
  inspect sunrpc
  inspect tftp
  inspect sip
  inspect xdmcp
```

（注） デフォルトのクラスマップで使用される特別な **match default-inspection-traffic** コマンドの詳細については、「特定の機能アクションの非互換性」（P.20-17）を参照してください。

レイヤ 3/4 ポリシー マップの追加

ポリシー マップの最大数は 64 です。レイヤ 3/4 ポリシー マップを作成する手順は、次のとおりです。

ステップ 1　次のコマンドを入力して、ポリシーマップを追加します。

```
hostname(config)# policy-map policy_map_name
```

policy_map_name 引数は、最大 40 文字のポリシーマップ名です。すべてのタイプのポリシー マップが同じネーム スペースを使用しているため、他のタイプのポリシー マップですでに使用されている名前は再使用できません。CLI はポリシーマップ コンフィギュレーション モードに入ります。

ステップ 2　（任意）ポリシーマップの説明を指定します。

```
hostname(config-pmap)# description text
```

ステップ 3　次のコマンドを使用して、設定済みのレイヤ 3/4 クラスマップを指定します。

```
hostname(config-pmap)# class class_map_name
```

class_map_name は、前に作成したクラスマップの名前です。クラスマップを追加するには、「トラフィックの識別（レイヤ 3/4 クラス マップ）」（P.20-4）を参照してください。

ステップ 4　このクラスマップに、1 つ以上のアクションを指定します。

- TCP および UDP の接続制限およびタイムアウト、TCP シーケンス番号のランダム化。「接続制限とタイムアウトの設定」（P.21-1）を参照してください。
- TCP ステート バイパス。「TCP ステート バイパスの設定」（P.21-11）を参照してください。
- アプリケーション検査。第 22 章「アプリケーション層プロトコル検査の適用」を参照してください。
- PISA 統合を使用したアプリケーション タイプの許可または拒否：「PISA 統合でのアプリケーション タイプの許可または拒否」（P.21-4）を参照してください。

（注）　クラスマップに **match default_inspection_traffic** コマンドがない場合、そのクラスに最大 1 つの **inspect** コマンドを設定できます。

ステップ 5　このポリシーマップに含めるクラスマップごとに、ステップ 3 とステップ 4 を繰り返します。

次に、接続ポリシーに対する **policy-map** コマンドの例を示します。この例では、Web サーバ 10.1.1.1 に許可される接続の数を制限しています。

```
hostname(config)# access-list http-server permit tcp any host 10.1.1.1
hostname(config)# class-map http-server
hostname(config-cmap)# match access-list http-server

hostname(config)# policy-map global-policy
hostname(config-pmap)# description This policy map defines a policy concerning connection
to http server.
hostname(config-pmap)# class http-server
hostname(config-pmap-c)# set connection conn-max 256
```

次の例では、ポリシー マップでの複数一致の動作方法を示します。

```
hostname(config)# class-map inspection_default
hostname(config-cmap)# match default-inspection-traffic
hostname(config)# class-map http_traffic
hostname(config-cmap)# match port tcp eq 80

hostname(config)# policy-map outside_policy
hostname(config-pmap)# class inspection_default
hostname(config-pmap-c)# inspect http http_map
hostname(config-pmap-c)# inspect sip
hostname(config-pmap)# class http_traffic
hostname(config-pmap-c)# set connection timeout tcp 0:10:0
```

次に、使用可能な最初のクラス マップとトラフィックが一致し、同じ機能ドメインのアクションを指定する後続のいずれのクラス マップとも一致しない様子を示す例を示します。

```
hostname(config)# class-map telnet_traffic
hostname(config-cmap)# match port tcp eq 23
hostname(config)# class-map ftp_traffic
hostname(config-cmap)# match port tcp eq 21
hostname(config)# class-map tcp_traffic
hostname(config-cmap)# match port tcp range 1 65535
hostname(config)# class-map udp_traffic
hostname(config-cmap)# match port udp range 0 65535
hostname(config)# policy-map global_policy
hostname(config-pmap)# class telnet_traffic
hostname(config-pmap-c)# set connection timeout tcp 0:0:0
hostname(config-pmap-c)# set connection conn-max 100
hostname(config-pmap)# class ftp_traffic
hostname(config-pmap-c)# set connection timeout tcp 0:5:0
hostname(config-pmap-c)# set connection conn-max 50
hostname(config-pmap)# class tcp_traffic
hostname(config-pmap-c)# set connection timeout tcp 2:0:0
hostname(config-pmap-c)# set connection conn-max 2000
```

Telnet 接続が開始されると、**class telnet_traffic** と照合されます。同様に、FTP 接続が開始されると、**class ftp_traffic** と照合されます。Telnet と FTP 以外の TCP 接続の場合は、**class tcp_traffic** と照合されます。Telnet または FTP 接続が **class tcp_traffic** と一致するとしても、すでに他のクラスと一致しているため、FWSM はこの照合を実行しません。

インターフェイスへのアクションの適用（サービス ポリシー）

レイヤ 3/4 ポリシーマップをアクティブにするには、1 つ以上のインターフェイスに適用するサービス ポリシー、またはすべてのインターフェイスにグローバルに適用するサービス ポリシーを作成します。特定の機能において、インターフェイスのサービス ポリシーは、グローバルなサービス ポリシーよりも優先されます。たとえば、FTP 検査を含むグローバル ポリシーと TCP 接続設定を含むインターフェイス ポリシーがある場合、FTP 検査と TCP 接続設定の両方がインターフェイスに適用されます。ただし、FTP 検査を行うグローバル ポリシーと、FTP 検査を行うインターフェイス ポリシーが設定されている場合は、インターフェイス ポリシーの FTP 検査だけがインターフェイスに適用されます。

- ポリシーマップとインターフェイスを関連付けてサービス ポリシーを作成するには、次のコマンドを入力します。

  ```
  hostname(config)# service-policy policy_map_name interface interface_name
  ```

- 特定のポリシーを持たないすべてのインターフェイスに適用するサービス ポリシーを作成するには、次のコマンドを入力します。

  ```
  hostname(config)# service-policy policy_map_name global
  ```

  デフォルトで、コンフィギュレーションには、すべてのデフォルト アプリケーション検査トラフィックに一致するグローバル ポリシーが含まれており、トラフィックに対してグローバルに検査が適用されます。適用できるグローバル ポリシーは 1 つのみです。このため、グローバル ポリシーを変更する場合は、デフォルトのポリシーを編集するか、ディセーブルにして新しいポリシーを適用する必要があります。

  デフォルト サービス ポリシーには、次のコマンドが含まれています。

  ```
  service-policy global_policy global
  ```

たとえば、次のコマンドは、外部インターフェイスで inbound_policy ポリシーマップをイネーブルにします。

```
hostname(config)# service-policy inbound_policy interface outside
```

次に、デフォルトのグローバル ポリシーをディセーブルにして、他のすべての FWSM インターフェイスで新しい new_global_policy グローバル ポリシーをイネーブルにする例を示します。

```
hostname(config)# no service-policy global_policy global
hostname(config)# service-policy new_global_policy global
```

モジュラ ポリシー フレームワークの例

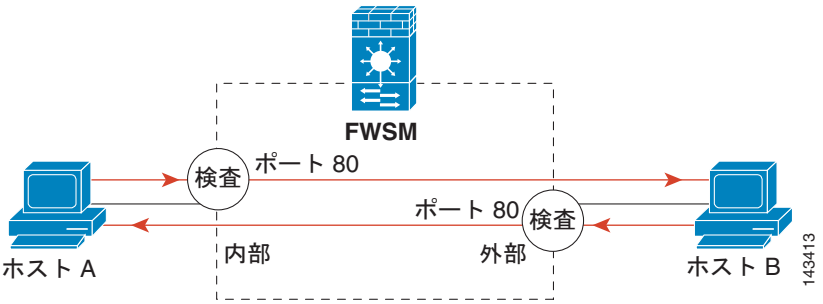
ここでは、モジュラ ポリシー フレームワークの例について説明します。内容は次のとおりです。

- 「HTTP トラフィックへの検査のグローバルな適用」（P.20-21）
- 「特定のサーバへの HTTP トラフィックに対する検査と接続制限値の適用」（P.20-22）
- 「NAT による HTTP トラフィックへの検査の適用」（P.20-22）

HTTP トラフィックへの検査のグローバルな適用

この例（図 20-1）では、任意のインターフェイスを通過して FWSM に入るすべての HTTP 接続（ポート 80 の TCP トラフィック）が HTTP 検査対象として分類されます。

図 20-1　グローバル HTTP 検査



この例について、次のコマンドを参照してください。

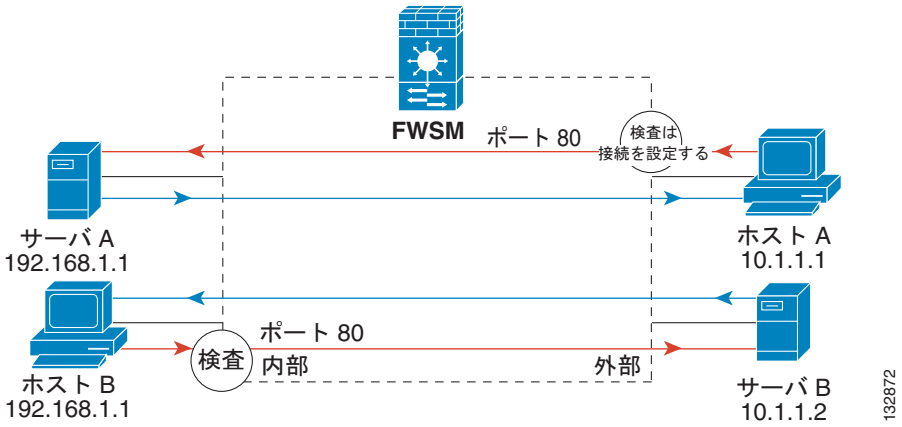
```
hostname(config)# class-map http_traffic
hostname(config-cmap)# match port tcp eq 80

hostname(config)# policy-map http_traffic_policy
hostname(config-pmap)# class http_traffic
hostname(config-pmap-c)# inspect http
hostname(config)# service-policy http_traffic_policy global
```

特定のサーバへの HTTP トラフィックに対する検査と接続制限値の適用

この例（図 20-2）では、外部インターフェイスを通過して FWSM に入るサーバ A 宛ての HTTP 接続（ポート 80 の TCP トラフィック）が HTTP 検査および最大接続数制限値の対象として分類されます。

内部インターフェイスを通じて FWSM に入るサーバ B 宛てのすべての HTTP 接続は、HTTP 検査対象として分類されます。

図 20-2　特定のサーバに対する HTTP 検査と接続制限値



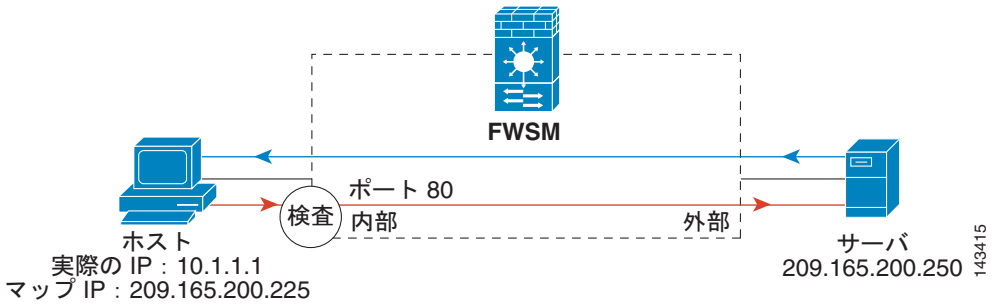
この例について、次のコマンドを参照してください。

```
hostname(config)# access-list serverA extended permit tcp any host 192.168.1.1 eq 80
hostname(config)# access-list ServerB extended permit tcp any host 10.1.1.2 eq 80

hostname(config)# class-map http_serverA
hostname(config-cmap)# match access-list serverA
hostname(config)# class-map http_serverB
hostname(config-cmap)# match access-list serverB

hostname(config)# policy-map policy_serverA
hostname(config-pmap)# class http_serverA
hostname(config-pmap-c)# inspect http http_map_serverA
hostname(config-pmap-c)# set connection conn-max 100
hostname(config)# policy-map policy_serverB
hostname(config-pmap)# class http_serverB
hostname(config-pmap-c)# inspect http http_map_serverB

hostname(config)# service-policy policy_serverB interface inside
hostname(config)# service-policy policy_serverA interface outside
```

NAT による HTTP トラフィックへの検査の適用

この例では、内部ネットワーク上のホストに、実 IP アドレス 10.1.1.1 と、外部ネットワークで使用されるマップされた IP アドレス 209.165.200.225 があります（図 20-3 を参照）。ポリシーは、実際のアドレスを使用する内部インターフェイスに適用されるため、クラスマップのアクセス リストでは、実際の IP アドレスを使用する必要があります。ポリシーを外部インターフェイスに適用する場合、マップされたアドレスを使用してください。

図 20-3　NAT を使用した HTTP 検査



この例について、次のコマンドを参照してください。

```
hostname(config)# static (inside,outside) 209.165.200.225 10.1.1.1
hostname(config)# access-list http_client extended permit tcp host 10.1.1.1 any eq 80

hostname(config)# class-map http_client
hostname(config-cmap)# match access-list http_client

hostname(config)# policy-map http_client
hostname(config-pmap)# class http_client
hostname(config-pmap-c)# inspect http

hostname(config)# service-policy http_client interface inside
```