Cisco IOS Flexible Packet Matching (FPM)
Getting Started with Cisco IOS Flexible Packet Matching

This guide provides an overview of Cisco IOS Flexible Packet Matching and describes the process of writing FPM filters along with examples. The following topics are covered in this document:

- What is Cisco IOS Flexible Packet Matching?
- Why do we need Cisco IOS Flexible Packet Matching?
- What platforms and images does it work on?
- How to write a Cisco IOS Flexible Packet Matching policy?
- Example policy

Pre-Requisites:
Before getting started with Flexible Packet Matching, ensure that you have the following:

- A Cisco ISR Router in the following model (87x, 18x, 28xx, 38xx, and 72xx)
- Console or telnet connectivity to the router
- IOS Release 12.4(4)T or later
- An ‘Advanced Security’ image loaded on the router

What is Cisco IOS Flexible Packet Matching (FPM)?
Networks are experiencing increasing sophisticated attacks that require mitigating tools that are as flexible as possible. Cisco IOS Flexible Packet Matching (FPM) is a set of classes and policies that provides pattern matching capability for more granular and customized packet filters for Layer 2 to 7—bit/byte matching capability deep into the packet at any offset within the packet header and payload.

Put simply, it is a powerful, easy, and rapid deployment mechanism that enables users to specify criteria to match against any part of a packet (header and payload) and define the action to take. In short, FPM is able to classify a packet based on its characteristics and take appropriate action.

Why do we Need Cisco IOS Flexible Packet Matching (FPM)?
There are three main reasons for Cisco IOS Flexible Packet Matching (FPM):

- **Sophisticated attacks**: characteristics of common attacks have evolved beyond current filtering tools like ACLs (i.e. limited matching criteria—protocol, port, ip address, etc.)
- **Rapid Mitigation**: customers must stop attacks immediately without waiting for a vendor to develop a signature or new code (i.e. IPS or ACL).
- **Finer Granularity**: goes beyond static attributes allowing you to specify arbitrary bits/bytes at any offset within the entire packet (header or payload), minimizing inadvertent blocking of legitimate business traffic
How does Cisco IOS Flexible Packet Matching Work?

Cisco IOS Flexible Packet Matching (FPM) uses the following characteristics to ensure users successfully mitigate attacks.

- Is a stateless solution and inspects one packet at a time.
- Matches on all static packet characteristics like protocol, port, IP address.
- Uses a Protocol Header Description File (PHDF) that allows the user to define a class match criteria based on any field in the protocol header.
- Supports an offset, size and string keywords, and regular expressions (regex) to allow the user to match on strings or bytes in the packet payload.
- Uses class-map and policy-map configuration syntax to specify the protocol stack, the match criteria and action to take.

What Platforms and Images does it Work On?

The following table lists the platforms and image that Cisco IOS Flexible Packet Matching (FPM) is supported in:

<table>
<thead>
<tr>
<th>Advanced Security Images</th>
<th>Advanced IP Services Images</th>
<th>Advanced Enterprise Service Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>1701 c1700-advsecurityk9-mz.</td>
<td>87x c870-adviservicesk9-mz.</td>
<td>1701 c1700-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1711 c1700-advsecurityk9-mz.</td>
<td>1701 c1700-adviservicesk9-mz.</td>
<td>1711 c1700-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1712 c1700-advsecurityk9-mz.</td>
<td>1711 c1700-adviservicesk9-mz.</td>
<td>1712 c1700-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1721 c1700-advsecurityk9-mz.</td>
<td>1712 c1700-adviservicesk9-mz.</td>
<td>1721 c1700-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1751 c1700-advsecurityk9-mz.</td>
<td>1721 c1700-adviservicesk9-mz.</td>
<td>1751 c1700-adventerprisek9-mz.</td>
</tr>
<tr>
<td>17xx c1700-k903sy7-mz</td>
<td>1751-V c1700-adviservicesk9-mz.</td>
<td>1760 c1700-adventerprisek9-mz.</td>
</tr>
<tr>
<td>17xx c1700-bk8n032s7y7-mz</td>
<td>1811 c181x-adviservicesk9-mz.</td>
<td>1811 c181x-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1760 c1700-advsecurityk9-mz.</td>
<td>1811 c181x-adviservicesk9-mz.</td>
<td>1812 c181x-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1811 c181x-adviservicesk9-mz.</td>
<td>1812 c181x-adviservicesk9-mz.</td>
<td>1841 c1841-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1812 c181x-adviservicesk9-mz.</td>
<td>2610XM-2611XM c2600-adviservicesk9-mz.</td>
<td>2610XM-2611XM c2600-adventerprisek9-mz.</td>
</tr>
<tr>
<td>1841 c1841-adviservicesk9-mz.</td>
<td>2620XM-2621XM c2600-adviservicesk9-mz.</td>
<td>2620XM-2621XM c2600-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2610XM-2611XM c2600-adviservicesk9-mz.</td>
<td>2650XM-2651XM c2600-adviservicesk9-mz.</td>
<td>2650XM-2651XM c2600-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2620XM-2621XM c2600-adviservicesk9-mz.</td>
<td>2691 c2691-adviservicesk9-mz.</td>
<td>2691 c2691-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2650XM-2651XM c2600-adviservicesk9-mz.</td>
<td>2801 c2801-adviservicesk9-mz.</td>
<td>2801 c2801-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2691 c2691-adviservicesk9-mz.</td>
<td>2811 c2800nm-adviservicesk9-mz.</td>
<td>2811 c2800nm-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2801 c2801-adviservicesk9-mz.</td>
<td>2821 c2800nm-adviservicesk9-mz.</td>
<td>2821 c2800nm-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2801 c2800nm-adviservicesk9-mz.</td>
<td>2851 c2800nm-adviservicesk9-mz.</td>
<td>2851 c2800nm-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2811 c2800nm-adviservicesk9-mz.</td>
<td>2811 c2800nm-adviservicesk9-mz.</td>
<td>2811 c2800nm-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2811 c2800nm-adviservicesk9-mz.</td>
<td>2821 c2800nm-adviservicesk9-mz.</td>
<td>2821 c2800nm-adventerprisek9-mz.</td>
</tr>
<tr>
<td>2851 c2800nm-adviservicesk9-mz.</td>
<td>7xxx c7xxx-ik903s</td>
<td>7xxx c7xxx-ik903s</td>
</tr>
<tr>
<td>7xxx c7xxx-jk903s</td>
<td>7xxx c7xxx-jk903s</td>
<td>7xxx c7xxx-jk903s</td>
</tr>
<tr>
<td>7xxx c7xxx-ik903s</td>
<td>7xxx c7xxx-jk903s</td>
<td>7xxx c7xxx-jk903s</td>
</tr>
</tbody>
</table>
Restrictions:
The following restrictions apply when using Cisco IOS Flexible Packet Matching (FPM):

- Since Cisco IOS Flexible Packet Matching (FPM) is stateless, it cannot keep track of port numbers being used by protocols that dynamically negotiate ports. Thus, when using FPM, port numbers must be explicitly specified.
- Cisco IOS Flexible Packet Matching (FPM) does not keep track of the "connection state" and so no connection attributes are retained.
- Cisco IOS Flexible Packet Matching (FPM) cannot perform IP fragmentation or TCP flow reassembly.
- Cisco IOS Flexible Packet Matching’s PHDFs describe protocols with static fields and lengths.
- Prior to 12.4(15)T, FPM supports searching for patterns up to 32 bytes long within the first 256 bytes of the packet. After 12.4(15)T, FPM supports searching for patterns up to 256 bytes long anywhere within the entire packet.

Steps to Configure Cisco IOS Flexible Packet Matching (FPM)
This section describes the steps required to create an FPM policy using IOS command-line interface (CLI). It contains the following steps:

Step 1. Load the protocol header description file(s) (PHDF)
Step 2. Define the protocol stack (IP-UDP, IP-TCP, etc.)
Step 3. Define FPM match criteria filter (class-map)
Step 4. Define action to take on classes (service-map)
Step 5. Apply service policy to an interface

Each step and the specific commands are described in the following pages. Example configuration is displayed in a box below each command. A section ‘Additional Commands and References’ under each step provides additional information.

An example Cisco IOS Flexible Packet Matching (FPM) configuration is also discussed.

1. Load the Protocol Header Description File
The first step is to load the protocol header description file (PHDF) into router memory. A PHDF is an XML file that allows the user to take advantage of the flexibility of XML to describe almost any protocol header.

PHDFs are analogous to stencils. A PHDF outlines the structure of packets in an XML format thus allowing IOS to understand the protocol and the packet structure. The field names that are defined within the PHDFs are used for defining the packet filters.
Figure 1 represents the format of an IP packet header:

![IP Header](image)

The IP PHDF defines the header field names (i.e. Version, IHL, TOS, etc...), the length of the fields and the location of the fields within the packet. The PHDFs simplifies FPM configurations by allowing the users to reference the field names described in the PHDF instead of having to define the actual offsets within the packet in the router’s configuration.

The most common PHDFs are published on [http://www.cisco.com/cgi-bin/tablebuild.pl/fpm](http://www.cisco.com/cgi-bin/tablebuild.pl/fpm). Although users can write their own custom PHDFs, it is recommended that users use the files published on [http://www.cisco.com/cgi-bin/tablebuild.pl/fpm](http://www.cisco.com/cgi-bin/tablebuild.pl/fpm) since they suffice for most situations.

**Note:** If using an image prior to 12.4(11)T, then you must download the PHDF files to your router flash. Images after 12.4(11)T have the PHDFs built into IOS.

**Step 6.** To load the PHDF, enter the following command at the router configure terminal prompt.

```
load protocol location{system | flash | disk#}: filename {ether.phdf | ip.phdf | tcp.phdf | udp.phdf | icmp.phdf | ipv6.phdf}
```

```
Router(config)# load protocol system:udp.phdf
```

**Additional Commands and References:**

Cisco IOS Flexible Packet Matching (FPM) Deployment Guide:

**2. Define Protocol Stack**

The second step is to define the protocol stack using class maps. There are two types of class maps: “stack” and “access-control”. The “stack” class allows us to logically define protocol relationships. In this step, the ‘class map type stack’ should be defined.

**Step 7.** To define the protocol stack in a class-map, issue the following commands in order:

```
class-map [type {stack | access-control}] class-map-name [match-all | match-any]
```

```
Router(config)# class-map type stack ip-udp match-all
```

```
match field protocol protocol-field [eq [mask] | neq [mask] | gt | lt | range range | regex string] value [next next-protocol]
```

```
Router(config-cmap)# match field ip protocol eq 0x11 next udp
```
### Additional Commands and References:

To view all class-maps configured on the router, enter the following command at the router prompt.
```
show class-map [type {stack | access-control}] [class-map-name]
```

Router# show class-map type access-control slammer

---

### 3. Define Cisco IOS Flexible Packet Matching (FPM) Match Criteria Filter

The third step is to define the Cisco IOS Flexible Packet Matching (FPM) match criteria filter using class maps. There are two types of class maps: “stack” and “access-control”. The “access-control” class allows us to describe granular details about a particular packet. In this step, the ‘class map type access-control’ should be defined along with the match criteria.

Cisco IOS Flexible Packet Matching (FPM) is a packet classification feature that allows users to define one or more classes of network traffic by pairing standard matching operators with user-defined protocol header fields.

#### Step 8. To define the FPM match criteria filter in a class-map, issue the following commands in order:

```
class-map [type {stack | access-control}] class-map-name [match-all | match-any]
```

Router(config)# class-map type access-control slammer match-all

```
description character-string
```

Router(config-cmap)# description "match on slammer packets"

```
match field protocol protocol-field [eq [mask] | neq [mask] | gt | lt | range range | regex string] value [next next-protocol]
```

Router(config-cmap)# match field udp dest-port eq 1434

```
macth start {l2-start | l3-start} offset number size number {eq | neq | gt | lt | range range | regex string} value [value2]
```

Router(config-cmap)# match start l3-start offset 224 size 4 eq 0x4011010

The following regular expressions are supported:

<table>
<thead>
<tr>
<th>Regular Expression Meta</th>
<th>Character Function</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII Character</td>
<td>Represents the character itself</td>
<td>abcd would match the string “abcd” anywhere in the search area</td>
</tr>
<tr>
<td>.</td>
<td>Represents any character</td>
<td>t.t matches strings such as test, text and tart</td>
</tr>
<tr>
<td>[]</td>
<td>A set of characters or a range of characters with (-)</td>
<td>[02468a-z] matches 0, 1, and w, but not 1, 9, or K</td>
</tr>
<tr>
<td>*</td>
<td>Zero or more of preceding characters</td>
<td>5* matches any occurrence of the number 5, including none</td>
</tr>
<tr>
<td>?</td>
<td>Zero or one of preceding characters</td>
<td>ba?b matches bb and bab</td>
</tr>
<tr>
<td>\</td>
<td>Escape character-specifies what follows as a character instead of a meta-character</td>
<td>18... matches the characters 18. and any characters that follow 18.</td>
</tr>
</tbody>
</table>

---

### Additional Commands and References:

To view all class-maps configured on the router, enter the following command at the router prompt.
```
show class-map [type {stack | access-control}] [class-map-name]
```

Router# show class-map type access-control slammer

---
4. Define Action to Take on Classes

The fourth step is to create the hierarchical Cisco IOS Flexible Packet Matching (FPM) policy that defines the action to take on classes. A policy includes the policy map; that is, after a class map has been defined, a policy map is needed to bind the match to an action. A policy map is an ordered set of classes and associated actions, such as drop, log, or send ICMP unreachable.

Step 9. To define a policy-map, issue the following commands in order:

```
policy-map [type access-control] policy-map-name

description character-string

Router(config-pmap)# description "policy for UDP based attacks"

class class-name [insert-before class-name]

Router(config-pmap)# class slammer [drop | log | send-response | service-policy]

Router(config-pmap)# drop
```

Chain the first policy with it’s class to the stack class:

```
policy-map type access-control fpm-policy

Router(config-pmap)# class ip_udp (reference the class-map stack that was configured in Step 2)

service-policy policy-map-name

Router(config-pmap-c)# service policy fpm-udp-policy
```

Additional Commands and References:

To view all policy-maps configured on the router, enter the following command at the router prompt.

```
show policy-map [type access-control] [policy-map-name]

Router# show policy-map type access-control slammer
```

5. Apply Service Policy to an Interface

The fifth step is to apply the service policy to an interface on the router.

Step 10. To apply the policy on an interface, issue the following the commands:

```
interface type name

Router(config)# interface gigabitEthernet 0/1

service-policy [type access-control] {input | output} policy-map-name

Router(config-if)# service-policy type access-control input fpm-policy
```

Additional Commands and References:

To view the FPM configuration and both class type stack and class type access-control matches, enter the following command at the router prompt.

```
show policy-map interface [type access-control] interface-name [input | output]

Router# show policy-map interface type access-control interface gigabit 0/1
```

To track all FPM events, enter the following command at the router prompt.
**debug fpm event**

The following sample output is from the `debug fpm event` command:

```
Router#debug fpm event
  0x0, ip-flags: 0x80000000
```

**Example: Cisco IOS Flexible Packet Matching (FPM) Configuration for Peer-to-Peer VoIP Client Applications**

This section contains the configuration example to block peer-to-peer VoIP applications (for example Skype version 2.5).

**Why Block Skype?**

Ensure corporate network security. Skype has the ability to work on any network, regardless of the types of NAT, proxy, firewall, or intrusion prevention systems that are put in place.

**How does It Work?**

Skype uses an aggressive adaptive networking application that is designed to reach the Internet at all costs. Skype sessions use an asymmetric key exchange to distribute the 256 bit symmetric key employed by the AES cipher for session encryption. Skype's initial outbound connection can use any dynamic combination of TCP and UDP ports, including outbound ports 80 and 443, which are generally open for HTTP and HTTPS access. This renders traditional port blocking filters completely ineffective. In addition, Skype uses proprietary methods of NAT traversal similar to STUN (Simple Traversal of UDP through NAT), ICE (Interactive Connectivity Establishment) and TURN (Traversal Using Relay NAT) to ensure that you can reach the Internet and to determine the client's eligibility to be a super node.

**How to Block Skype?**

In order to block Skype, perform the following steps. A flow of the process is shown below:

Figure 2. Cisco IOS Flexible Packet Matching (FPM) Flow for Skype
Deriving the Skype Match Pattern

The following figure shows a screenshot of the packet capture for the initial conversation (i.e. login) for Skype. This information is used to derive the appropriate pattern and offset for creating the match criteria for the Cisco IOS Flexible Packet Matching (FPM) filter.

**Figure 3.** Skype Initial Conversation Packet Capture
Using Wireshark to capture initial conversation of skype (i.e. login)

Cisco IOS Flexible Packet Matching (FPM) Configuration to Block Skype

The following configuration shows how to define Cisco IOS Flexible Packet Matching (FPM) traffic classes to block Skype packets at login. If Skype is already connected, and this Cisco IOS Flexible Packet Matching (FPM) policy is applied, it does not drop the current connection but prevents new connections from occurring.

```plaintext
!---Load Protocol Header Description File
load protocol system:/fpm/phdf/ip.phdf
load protocol system:/fpm/phdf/tcp.phdf
!
!---Defines Protocol Stack and Match Criteria (FPM filter to block Skype traffic)
class-map type stack match-all ip_tcp
  match field IP protocol eq 6 next TCP
class-map type access-control match-all skype
  match start TCP payload-start offset 0 size 4 eq 0x17030100
!
!---Define Policy and attach class-map stack, and service policy to policy-map.
policy-map type access-control child
  class skype
    log
    drop

policy-map type access-control parent
  class ip_tcp
    service-policy child
!
!---Apply Service Policy to the outside Interface which connects to the PC running Skype, and attach policy-map-name to service policy
```
interface FastEthernet1
ip address 128.107.163.73 255.255.254.0
ip nat outside
ip virtual-reassembly
service-policy type access-control input parent

Figure 4 shows a login attempt that has failed after applying the Cisco IOS Flexible Packet Matching (FPM) configuration:

Figure 4.

Router# show policy-map type access-control interface
Virtual-Access2
Service-policy access-control input: parent
Class-map: ip_tcp (match-all)
  118 packets, 5714 bytes
  5 minute offered rate 0 bps
Match: field IP protocol eq 6 next TCP
Service-policy access-control : child
Class-map: skype (match-all)
  63 packets, 2961 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: start TCP payload-start offset 0 size 4 eq 0x17030100
  log
drop