

Flexible Packet Matching

Flexible Packet Matching (FPM) is an access control list (ACL) pattern matching tool, providing more thorough and customized packet filters. FPM enables users to match on arbitrary bits of a packet at an arbitrary depth in the packet header and payload. FPM removes constraints to specific fields that had limited packet inspection.

FPM enables users to create their own stateless packet classification criteria and to define policies with multiple actions (such as drop, log, or send Internet Control Message Protocol [ICMP] unreachable[1]) to immediately block new viruses, worms, and attacks.

- Prerequisites for Flexible Packet Matching, on page 1
- Restrictions for Flexible Packet Matching, on page 1
- Information About Flexible Packet Matching, on page 2
- How to Configure Flexible Packet Matching, on page 3
- Configuration Examples for an FPM Configuration, on page 9
- Additional References, on page 10
- Feature Information for Flexible Packet Matching, on page 11

Prerequisites for Flexible Packet Matching

Although access to an XML editor is not required, XML will ease the creation of protocol header description files (PHDFs).

Restrictions for Flexible Packet Matching

- FPM can search for patterns up to 32 bytes in length within the first 256 bytes of the packet.
- A maximum of 32 classes are supported in a policy-map.
- For IP option packets, FPM inspects only the fields in the Layer 2 header and the first 20 bytes of the IP header.
- For noninitial IP fragments, FPM inspects only the fields in the Layer 2 header and the first 20 bytes of the IP header.
- FPM cannot be used to mitigate an attack that requires stateful classification.

[1] Send ICMP unreachable is currently not supported on the Supervisor Engine 32 PISA.

- Because 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.
- FPM cannot perform IP fragmentation or TCP flow reassembly.
- FPM inspects only IPv4 unicast packets.
- FPM cannot classify packets with IP options.
- FPM does not support multicast packet inspection.
- FPM is not supported on tunnel and MPLS interfaces.
- Noninitial fragments will not be matched by the FPM engine.
- Offset can be only a constant in a match start construct.
- FPM cannot match across packets.
- Mapping of FPM policies to control-plane is not supported.

Information About Flexible Packet Matching

Flexible Packet Matching Functional Overview

FPM allows customers to create their own filtering policies that can immediately detect and block new viruses and attacks.

A filtering policy is defined via the following tasks:

- Load a PHDF (for protocol header field matching)
- Define a class map and define the protocol stack chain (traffic class)
- Define a service policy (traffic policy)
- Apply the service policy to an interface

Protocol Header Description File

Protocol headers are defined in separate files called PHDFs; the field names that are defined within the PHDFs are used for defining the packet filters. A PHDF is a file that allows the user to leverage the flexibility of XML to describe almost any protocol header. The important components of the PHDF are the version, the XML file schema location, and the protocol field definitions. The protocol field definitions name the appropriate field in the protocol header, allow for a comment describing the field, provide the location of the protocol header field in the header (the offset is relative to the start of the protocol header), and provide the length of the field. Users can choose to specify the measurement in bytes or in bits.



Note The total length of the header must be specified at the end of each PHDF.

Note When redundant sup PHDF files are used by FPM policy, the files should also be on standby sup's corresponding disk. If the files are not available FPM policy will not work after the switch over.

Users can write their own custom PHDFs via XML for existing or proprietary protocols. However, the following standard PHDFs can also be loaded onto the router via the **load protocol** command: ip.phdf, ether.phdf, tcp.phdf, and udp.phdf.

Note Because PHDFs are defined via XML, they are not shown in a running configuration. However, you can use the **show protocol phdf** command to verify the loaded PHDF.

Standard PHDFs are available on Cisco.com at the following URL:
http://www.cisco.com/cgi-bin/tablebuild.pl/fpm

Filter Description

A filter description is a definition of a traffic class that can contain the header fields defined in a PHDF (using the **match field** command). If a PHDF is not loaded, the traffic class can be defined through the datagram header start (Layer 2) or the network header start (Layer 3) (using the **match start** command). If a PHDF has been loaded onto the router, the class specification begins with a list of the protocol headers in the packet.

A filter definition also 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.

For information on how to configure a class map and a policy map for FPM, see the How to Configure a Flexible Packet Matching Traffic Class and Traffic Policy section.

How to Configure Flexible Packet Matching

Creating a Traffic Class for Flexible Packet Matching

Note If the PHDF protocol fields are referenced in the access-control classmap, the stack classmap is required in order to make FPM work properly

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **load protocol** *location:filename*
4. **class-map** [**type** {**stack** | **access-control**}] *class-map-name* [**match-all** | **match-any**]
5. **description** *character-string*

6. **match field** *protocol protocol-field* {**eq** [*mask*] | **neq** | [*mask*] | **gt** | **lt** | **range** *range* | **regex** *string*} *value* [**next** *next-protocol*]
7. **match start** {**l2-start** | **l3-start**} **offset** *number* **size** *number* {**eq** | **neq** | **gt** | **lt** | **range** *range* | **regex** *string*} {*value* [*value2*] | [*string*]}
8. **match class** *class-name* [**packet-range** *low high* | **byte-range** *low high*] **session**
9. **exit**
10. **exit**
11. **show class-map** [**type** {**stack** | **access-control**} | *class-map-name*]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**load protocol** *location:filename* **Example:** `Router(config)# load protocol disk2:udp.phdf`	(Optional) Loads a PHDF onto a router. • The specified location must be local to the router. **Note** If a PHDF is not loaded, only the **match start** command can be used; that is, you cannot issue the **match field** command. **Note** For the ASR platform, PHDF files should be manually copied (through the **load protocol** command) to the active and standby route processor (RP) file systems.
Step 4	**class-map** [**type** {**stack**	**access-control**}] *class-map-name* [**match-all**	**match-any**] **Example:** `Router(config)# class-map type access-control c1`	Creates a class map to be used for matching packets to a specified class and enters class-map configuration mode. • **type stack** -- Enables FPM to determine the correct protocol stack in which to examine. • **type access-control** -- Determines the exact pattern to look for in the protocol stack of interest. • *class-map-name* -- Can be a maximum of 40 alphanumeric characters.

	Command or Action	Purpose
		• If **match-all** or **match-any**or are not specified, traffic must match all the match criterion to be classified as part of the traffic class.
Step 5	**description** *character-string* **Example:** `Router(config-cmap)# description "match on slammer packets"`	(Optional) Adds a description to the class map.
Step 6	**match field** *protocol protocol-field* {**eq** [*mask*] | **neq** | [*mask*] | **gt** | **lt** | **range** *range* | **regex** *string*} *value* [**next** *next-protocol*] **Example:** `Router(config-cmap)# match field udp dest-port eq 0x59A`	(Optional) Configures the match criteria for a class map on the basis of the fields defined in the PHDFs. • The **next** *next-protocol* keyword-argument pair is available only after configuring the **class-map type stack** command.
Step 7	**match start** {**l2-start** | **l3-start**} **offset** *number* **size** *number* {**eq** | **neq** | **gt** | **lt** | **range** *range* | **regex** *string*} {*value* [*value2*] | [*string*]} **Example:** `Router(config-cmap)# match start l3-start offset 224 size 4 eq 0x4011010`	(Optional) Configures the match criteria for a class map on the basis of the datagram header (Layer 2) or the network header (Layer 3).
Step 8	**match class** *class-name* [**packet-range** *low high* | **byte-range** *low high*] **session** **Example:** `Router(config-cmap)# match class c2 packet-range 1 5 session`	(Optional) Configures match criteria for a class map that identifies a session (flow) containing packets of interest, which is then applied to all packets transmitted during the session. The **packet-range** and **byte-range** keywords create a filter mechanism that increases the performance and matching accuracy of regex-based FPM class maps by classifying traffic that resides in the narrow packet number or packet byte ranges of each packet flow. When the **session** keyword is used with the *class-name* argument, the classification results are preserved for the subsequent packets of the same packet session. When the **session** keyword is used with the **packet-range** or **byte-range** keywords, the classification results are preserved for the specified packets or bytes of the same packet session.
Step 9	**exit** **Example:** `Router(config-cmap)# exit`	Exits class-map configuration mode.

	Command or Action	Purpose
Step 10	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 11	**show class-map** [**type** {**stack** | **access-control**} | *class-map-name*] **Example:** `Router# show class-map type access-control slammer`	(Optional) Displays configured FPM class maps.

Troubleshooting Tips

To track all FPM events, issue the **debug fpm event** command.

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

```
*Jun 21 09:22:21.607: policy-classification-inline(): matches class: class-default *Jun 21
 09:22:21.607: packet-access-control(): policy-map: fpm-policy, dir: input, match. retval:
 0x0, ip-flags: 0x80000000
```

What to Do Next

After you have defined at least one class map for your network, you must create a traffic policy and apply that policy to an interface as shown in the following task "Creating a Traffic Policy for Flexible Packet Matching."

Creating a Traffic Policy for Flexible Packet Matching

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map type access-control** *policy-map-name*
4. **description** *character-string*
5. **class** *class-name* **insert-before** *class-name*
6. **drop** [**all**]
7. **log** [**all**]
8. **service-policy** *policy-map-name*
9. **exit**
10. **interface** *type number*
11. **service-policy type access-control** {**input** | **output**} *policy-map-name*
12. **exit**
13. **exit**
14. **show policy-map** [**type access-control** | **interface** *type number* | **input** | **output**]

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map type access-control** *policy-map-name* **Example:** `Router(config)# policy-map type access-control fpm-udp-policy`	Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy, and enters policy-map configuration mode.
Step 4	**description** *character-string* **Example:** `Router(config-pmap)# description "policy for UDP based attacks"`	(Optional) Adds a description to the policy map.
Step 5	**class** *class-name* **insert-before** *class-name* **Example:** `Router(config-pmap)# class slammer`	Specifies the name of a predefined traffic class, which was configured with the **class-map** command. The **class** command also classifies traffic to the traffic policy and enters policy-map class configuration mode. • The **insert-before** *class-name* keyword and argument adds a class map to any location within the policy map. If this option is not issued, the class map is appended to the end of the policy map.
Step 6	**drop** [**all**] **Example:** `Router(config-pmap-c)# drop all`	(Optional) Configures a traffic class to discard packets belonging to a specific class. The **all** keyword is used to discard the entire stream of packets belonging to the traffic class. If this command is issued, note the following restrictions: • Discarding packets is the only action that can be configured in a traffic class. • When a traffic class is configured with the **drop** command, a "child" (nested) policy cannot be configured for this specific traffic class through the **service policy** command.

	Command or Action	Purpose
		• Discarding packets cannot be configured for the default class specified via the **class class-default** command. • If the **drop all**command is specified, then this command can only be associated with a **class map type access-control** command.
Step 7	**log** [**all**] **Example:** `Router(config-pmap-c)# log all`	(Optional) Generates log messages for the traffic class. The **all** keyword is used to log the entire stream of discarded packets belonging to the traffic class. This keyword is only available for a class map that is created with the **class-map type access-control** command.
Step 8	**service-policy** *policy-map-name* **Example:** `Router(config-pmap-c)# service policy fpm-udp-policy`	Creates hierarchical service policies.

Step 9	**exit**
	Example:
	`Router(config-pmap-c)# exit`
	Example:
	`Router(config-pmap)# exit`
		Exits policy-map class configuration mode and policy-map configuration mode.

Step 10	**interface** *type number* **Example:** `Router(config)# interface gigabitEthernet 0/1`	Configures an interface type and enters interface configuration mode.
Step 11	**service-policy type access-control** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy type access-control input fpm-policy`	Specifies the type and the name of the traffic policy to be attached to the input or output direction of an interface.
Step 12	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 13	**exit** **Example:**	Exits global configuration mode.

	Command or Action	Purpose
	`Router(config)# exit`	
Step 14	**show policy-map** [**type access-control** | **interface** *type number* | **input** | **output**] **Example:** `Router# show policy-map type access-control interface gigabitethernet 0/1`	(Optional) Verifies the FPM configuration. **Note** Once a traffic policy is created for FPM, a matched packet can be copied or redirected to a different destination interface.

Configuration Examples for an FPM Configuration

Configuring and Verifying FPM on ASR Platform: Example

The following example shows how to configure FPM on the ASR platform.

```
load protocol bootflash:ip.phdf
load protocol bootflash:tcp.phdf
class-map type stack match-all ip_tcp
 match field IP protocol eq 6 next TCP
class-map type access-control match-all test_class
 match field TCP dest-port gt 10
 match start l3-start offset 40 size 32 regex "ABCD"
policy-map type access-control child
 class test_class
  drop
policy-map type access-control parent
 class ip_tcp
  service-policy child
interface GigabitEthernet0/3/0
 ip address 10.1.1.1 255.0.0.0
 service-policy type access-control input parent
```

In the following sample output, all TCP packets are seen under the class-map “ip_tcp” and all packets matching the specific pattern are seen under the class-map “test_class.” TCP packets without the specific pattern are seen under the child policy “class-default,” while all non-TCP packets are seen under the parent policy “class-default.” (The counter is 0 in this example.)

```
Router# show policy-map type access-control interface GigabitEthernet0/3/0
GigabitEthernet0/3/0
 Service-policy access-control input: parent
  Class-map: ip_tcp (match-all)
  2024995578 packets, 170099628552 bytes
  5 minute offered rate 775915000 bps
  Match: field IP version eq 4
  Match: field IP ihl eq 5
  Match: field IP protocol eq 6 next TCP
 Service-policy access-control : child
 Class-map: test_class (match-all)
  1598134279 packets, 134243279436 bytes
  5 minute offered rate 771012000 bps, drop rate 771012000 bps
  Match: field TCP dest-port gt 10
  Match: start l3-start offset 40 size 32 regex "ABCD"
```

```
 drop
 Class-map: class-default (match-any)
  426861294 packets, 35856348696 bytes
  5 minute offered rate 4846000 bps, drop rate 0 bps
  Match: any
 Class-map: class-default (match-any)
  0 packets, 0 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: any
Router#
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Security commands	*Cisco IOS Security Command Reference*
Configuring FPM using traffic classification definition files.	"Flexible Packet Matching XML Configuration" module in the *Cisco IOS Security Configuration Guide: Securing the Data Plane*
Complete suite of quality of service (QoS) commands	*Cisco IOS Quality of Service Solutions Command Reference*

MIBs

MIBs	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Flexible Packet Matching

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for Flexible Packet Matching

Feature Name	Releases	Feature Information
Flexible Packet Matching	Cisco IOS XE Release 2.2	FPM is a packet classification feature that allows users to define one or more classes of network traffic by pairing a set of standard matching operators with user-defined protocol header fields. The following commands were introduced or modified: **class (policy-map)** **class-map** **debug fpm event,** **description (class-map)** **load protocol** **match field** **match start**, **policy-map**, **service-policy**, **show class-map**, **show policy-map interface**, **show protocol phdf**.