Configuring Control Plane Policing

Restrictions for CoPP

Restrictions for control plane policing (CoPP) include the following:

- Only ingress CoPP is supported. The **system-cpp-policy** policy-map is available on the control plane interface, and only in the ingress direction.
- Only the **system-cpp-policy** policy-map can be installed on the control plane interface.
- The **system-cpp-policy** policy-map and the system-defined classes cannot be modified or deleted.
- Only the **police** action is allowed under the **system-cpp-policy** policy-map. The police rate for system-defined classes must be configured only in packets per second (pps); for user-defined class maps this must be configured only in bits per second (bps).
- One or more CPU queues are part of each class-map. Where multiple CPU queues belong to one class-map, changing the policer rate of a class-map affects all CPU queues that belong to that class-map. Similarly, disabling the policer in a class-map disables all queues that belong to that class-map. See Table 1: System-Defined Values for CoPP, on page 3 for information about which CPU queues belong to each class-map.
- The **show run** command does not display information about classes configured under `system-cpp policy`, when they are left at default values. Use the **show policy-map system-cpp-policy** or the **show policy-map control-plane** commands instead.

  You can continue use the **show run** command to display information about custom policies.

Related Topics

Enabling a CPU Queue or Changing the Policer Rate, on page 6

Disabling a CPU Queue, on page 8

Setting the Default Policer Rates for All CPU Queues, on page 9

Information About Control Plane Policing

This chapter describes how control plane policing (CoPP) works on your device and how to configure it.

CoPP Overview

The CoPP feature improves security on your device protecting the CPU from unnecessary traffic and DoS attacks. It can also protect control and management traffic from traffic drops caused by high volumes of other, lower priority traffic.

Your device is typically segmented into three planes of operation, each with its own objective:

- The data plane, to forward data packets.
- The control plane, to route data correctly.
- The management plane, to manage network elements.

You can use CoPP to protect most of the CPU-bound traffic and ensure routing stability, reachability, and packet delivery. Most importantly, you can use CoPP to protect the CPU from a DoS attack.

CoPP uses the modular QoS command-line interface (MQC) and CPU queues to achieve these objectives. Different types of control plane traffic are grouped together based on certain criteria, and assigned to a CPU queue. You can manage these CPU queues by configuring dedicated policers in hardware. For example, you can modify the policer rate for certain CPU queues (traffic-type), or you can disable the policer for a certain type of traffic.

Although the policers are configured in hardware, CoPP does not affect CPU performance or the performance of the data plane. But since it limits the number of packets going to CPU, the CPU load is controlled. This means that services waiting for packets from hardware may see a more controlled rate of incoming packets (the rate being user-configurable).

System-Defined Aspects of CoPP

When you power-up the device for the first time, the system automatically performs the following tasks:

- Looks for policy-map **system-cpp-policy**. If not found, the system creates and installs it on the control-plane.
- Creates eighteen class-maps under **system-cpp-policy**.

  The next time you power-up the device, the system detects the policy and class maps that have already been created.
- Enables all CPU queues by default, with their respective default rate. The default rates are indicated in the table System-Defined Values for CoPP.

The following table lists the class-maps that the system creates when you load the device. It lists the policer that corresponds to each class-map and one or more CPU queues that are grouped under each class-map. There is a one-to-one mapping of class-maps to policers; and one or more CPU queues map to a class-map.

Table 1: System-Defined Values for CoPP

Class Maps Names	Policer Index (Policer No.)	CPU queues (Queue No.)	Default Policer Rate (pps)
system-cpp- police-data	WK_CPP_POLICE_DATA(0)	WK_CPU_Q_ICMP_GEN(3)	600
		WK_CPU_Q_BROADCAST(12)	600
		WK_CPU_Q_ICMP_REDIRECT (6)	600
system-cpp-police-l2- control	WK_CPP_POLICE_L2_ CONTROL(1)	WK_CPU_Q_L2_CONTROL(1)	2000
system-cpp-police-routing-control	WK_CPP_POLICE_ROUTING_CONTROL(2)	WK_CPU_Q_ROUTING_CONTROL(4)	5400
		WK_CPU_Q_LOW_LATENCY (27)	5400
system-cpp-police-punt-webauth	WK_CPP_POLICE_PU NT_WEBAUTH(7)	WK_CPU_Q_PUNT_WEBAUTH(22)	1000
system-cpp-police- topology-control	WK_CPP_POLICE_TOPOLOGY_CONTROL(8)	WK_CPU_Q_TOPOLOGY_CONTROL(15)	13000
system-cpp-police- multicast	WK_CPP_POLICE_MULTICAST(9)	WK_CPU_Q_TRANSIT_TRAFFIC(18)	500
		WK_CPU_Q_MCAST_DATA(30)	500
system-cpp-police-sys- data	WK_CPP_POLICE_SYS _DATA (10)	WK_CPU_Q_OPENFLOW (13)	100
		WK_CPU_Q_CRYPTO_CONTROL(23)	100
		WK_CPU_Q_EXCEPTION(24)	100
		WK_CPU_Q_EGR_EXCEPTION(28)	100
		WK_CPU_Q_NFL_SAMPLED_DATA(26)	100
		WK_CPU_Q_GOLD_PKT(31)	100
		WK_CPU_Q_RPF_FAILED(19)	100
system-cpp-police-dot1x-auth	WK_CPP_POLICE_DOT1X(11)	WK_CPU_Q_DOT1X_AUTH(0)	1000
system-cpp-police- protocol-snooping	WK_CPP_POLICE_PR	WK_CPU_Q_PROTO_SNOOPING(16)	2000
system-cpp-police-dhcp-snooping	WK_CPP_DHCP_SNOOPING	WK_CPU_Q_DHCP_SNOOPING(17)	500
system-cpp-police-sw-forward	WK_CPP_POLICE_SW_FWD (13)	WK_CPU_Q_SW_FORWARDING_Q(14)	1000
		WK_CPU_Q_LOGGING(21)	1000
		WK_CPU_Q_L2_LVX_DATA_PACK (11)	1000
system-cpp-police-forus	WK_CPP_POLICE_FORUS(14)	WK_CPU_Q_FORUS_ADDR_RESOLUTION(5)	4000
		WK_CPU_Q_FORUS_TRAFFIC(2)	4000

Class Maps Names	Policer Index (Policer No.)	CPU queues (Queue No.)	Default Policer Rate (pps)
system-cpp-police-multicast-end-station	WK_CPP_POLICE_MULTICAST_SNOOPING(6)	WK_CPU_Q_MCAST_END_STATION_SERVICE(20)	2000

system-cpp-default	WK_CPP_POLICE_DEFAULT_POLICER	WK_CPU_Q_INTER_FED_TRAFFIC
		WK_CPU_Q_EWLC_CONTROL(9)
		WK_CPU_Q_EWLC_DATA(10)
			2000
			2000
			2000

system-cpp-police-stackwise-virt-control	WK_CPP_STACKWISE_VIRTUAL_CONTROL	WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL (29)	8000
system-cpp-police-l2lvx-control	WK_CPP_ L2_LVX_CONT_PACK	WK_CPU_Q_L2_LVX_CONT_PACK(8)	1000
system-cpp-police-high-rate-app	WK_CPP_HIGH_RATE_APP	WK_CPU_Q_HIGH_RATE_APP	13000
system-cpp-police-system-critical	WK_CPP_SYSTEM_CRITICAL	WK_CPU_Q_SYSTEM_CRITICAL	1000

When you upgrade or downgrade the software version on your device, note the following:

- When upgrading from one software release to another:

  The upgrade could be from Cisco IOS XE Release 3.x.xE to a Cisco IOS XE 16.x.x release, or from one Cisco IOS XE 16.x.x release to another Cisco IOS XE 16.x.x release:

  - If the device did not have a `system-cpp-policy` policy map before upgrade, then on upgrade, a default policy is created.
  - If the device had a `system-cpp-policy` policy map before upgrade, then on upgrade, the policy is not re-generated. Enter the **cpp system-default** command in global configuration mode to get the default policy working.

  Note We recommend that you to enter the **cpp system-default** command after any major upgrade to get the latest, default policer rates.

- When downgrading from one software release to another:

  The downgrade could be from a Cisco IOS XE 16.x.x release to a Cisco IOS XE Release 3.x.xE, or from one Cisco IOS XE 16.x.x release to another Cisco IOS XE 16.x.x release:

  - The `system-cpp-policy` policy map is retained on the device, but not installed on the control plane. You can delete the policy.

- If you downgrade to an earlier release and then upgrade to a later release:

  For example, if you downgrade from Cisco IOS XE 16.x.x release to Cisco IOS XE Release 3.x.xE and then upgrading to a Cisco IOS XE 16.x.x release:

  - If you delete the policy after downgrading to Cisco IOS XE Release 3.x.xE and then upgrade to a Cisco IOS XE 16.x.x release, the policy is generated with defaults.

- If you do not delete the policy after downgrading to Cisco IOS XE Release 3.x.xE, then on upgrade to a Cisco IOS XE 16.x.x release, the policy is not regenerated.

  Enter the **cpp system-default** command in global configuration mode to get the default policy working.

User-Configurable Aspects of CoPP

You can perform these tasks to manage control plane traffic:

Note All `system-cpp-policy` configurations must be saved so they are retained after reboot.

Enable or Disable a Policer for CPU Queues

Enable a policer for a CPU queue, by configuring a policer action (in packets per second) under the corresponding class-map, within the `system-cpp-policy` policy-map.

Disable a policer for CPU queue, by removing the policer action under the corresponding class-map, within the `system-cpp-policy` policy-map.

Note If a default policer is already present, carefully consider and control its removal; otherwise the system may see a CPU hog or other anomalies, such as control packet drops.

Change the Policer Rate

You can do this by configuring a policer rate action (in packets per second), under the corresponding class-map, within the`system-cpp-policy` policy-map.

Set Policer Rates to Default

Set the policer for CPU queues to their default values, by entering the **cpp system-default** command in global configuration mode.

Create User-Defined Class Maps

If a given traffic class does not have a designated class map, and you want to protect this traffic, you can create specific class maps (with filters) for such traffic packets and add these user-defined class maps to `system-cpp-policy`.

While `system-cpp-policy` is applied in the ingress direction, the forwarding engine driver (FED) changes policers on user-defined class maps to the egress. The filters and the policers in all user-defined classes must therefore be applied as egress classifications and actions, respectively. The policy map itself is unaffected by this change in the direction.

When you add a user-defined class map to `system-cpp-policy`, the system automatically installs it on all 32 CPU queues (in addition to the control plane), resulting in 33 instances of the policy. You can see this by entering the **show platform software fed switch**{*switch_number* | **active** | **standby**} qos policy target status command in privileged EXEC mode.

The police rate on these class maps is controlled by the Active Queue Management (AQM) policer. AQM provides buffering control of traffic flows prior to queuing a packet into the transmit queue of a port, ensuring that certain flows do not hog the switch packet memory. If the AQM policer feature is enabled, any user-defined police rates exceeding the AQM policer limits are disregarded.

User defined class maps have normal QoS or ACL classification filters.

Related Topics

Enabling a CPU Queue or Changing the Policer Rate, on page 6
Disabling a CPU Queue, on page 8
Setting the Default Policer Rates for All CPU Queues, on page 9
Restrictions for CoPP, on page 1
Example: Enabling a CPU Queue or Changing the Policer Rate of a CPU Queue, on page 10
Example: Disabling a CPU Queue
Example: Setting the Default Policer Rates for All CPU Queues, on page 11

How to Configure CoPP

Enabling a CPU Queue or Changing the Policer Rate

The procedure to enable a CPU queue and change the policer rate of a CPU queue is the same. Follow these steps:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**policy-map** *policy-map-name*
	Example:
	`Device(config)# policy-map system-cpp-policy`
	`Device(config-pmap)#`
		Enters the policy map configuration mode.

Step 4	**class** *class-name* **Example:**	Enters the class action configuration mode. Enter the name of the class that corresponds

	Command or Action	Purpose

	`Device(config-pmap)# class system-cpp-police-protocol-snooping`
	`Device(config-pmap-c)#`
		to the CPU queue you want to enable. See table *System-Defined Values for CoPP*.

Step 5	**police rate** *rate* **pps**
	Example:
	`Device(config-pmap-c)# police rate 100 pps`
	`Device(config-pmap-c-police)#`
		Specifies an upper limit on the number of incoming packets processed per second, for the specified traffic class.
		Note The rate you specify is applied to all CPU queues that belong to the class-map you have specified.

Step 6	exit
	Example:
	`Device(config-pmap-c-police)# exit`
	`Device(config-pmap-c)# exit`
	`Device(config-pmap)# exit`
	`Device(config)#`
		Returns to the global configuration mode.

Step 7	**control-plane**
	Example:
	`Device(config)# control-plane`
	`Device(config-cp)#`
		Enters the control plane (config-cp) configuration mode

Step 8	**service-policy input** *policy-name*
	Example:
	`Device(config)# control-plane`
	`Device(config-cp)#service-policy input system-cpp-policy`
	`Device(config-cp)#`
		Installs system-cpp-policy in FED. This command is required for you to see the FED policy. Not configuring this command will lead to an error.

Step 9	**end**
	Example:
	`Device(config-cp)# end`
		Returns to the privileged EXEC mode.

Step 10	**show policy-map control-plane** **Example:** `Device# show policy-map control-plane`	Displays all the classes configured under `system-cpp policy`, the rates configured for the various traffic types, and statistics

Related Topics

User-Configurable Aspects of CoPP, on page 5
Restrictions for CoPP, on page 1
Example: Enabling a CPU Queue or Changing the Policer Rate of a CPU Queue, on page 10
Example: Disabling a CPU Queue
Example: Setting the Default Policer Rates for All CPU Queues, on page 11

Disabling a CPU Queue

Follow these steps to disable a CPU queue:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**policy-map** *policy-map-name*
	Example:
	`Device(config)# policy-map system-cpp-policy`
	`Device(config-pmap)#`
		Enters the policy map configuration mode.

Step 4	**class** *class-name*
	Example:
	`Device(config-pmap)# class system-cpp-police-protocol-snooping`
	`Device(config-pmap-c)#`
		Enters the class action configuration mode. Enter the name of the class that corresponds to the CPU queue you want to disable. See the table, *System-Defined Values for CoPP*.

Step 5	**no police rate** *rate* **pps** **Example:** `Device(config-pmap-c)# no police rate 100 pps`	Disables incoming packet processing for the specified traffic class. **Note** This disables all CPU queues that belong to the class-map you have specified.

Step 6	**end**
	Example:
	`Device(config-pmap-c)# end`
		Returns to the privileged EXEC mode.

Step 7	**show policy-map control-plane** **Example:** `Device# show policy-map control-plane`	Displays all the classes configured under `system-cpp policy` and the rates configured for the various traffic types and statistics.

Related Topics

User-Configurable Aspects of CoPP, on page 5
Restrictions for CoPP, on page 1
Example: Enabling a CPU Queue or Changing the Policer Rate of a CPU Queue, on page 10
Example: Disabling a CPU Queue
Example: Setting the Default Policer Rates for All CPU Queues, on page 11

Setting the Default Policer Rates for All CPU Queues

Follow these steps to set the policer rates for all CPU queues to their default rates:

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**cpp system-default**
	Example:
	`Device(config)# cpp system-default`
	`Defaulting CPP : Policer rate for all classes will be set to their defaults`
		Sets the policer rates for all the classes to the default rate.

Step 4	**end**
	Example:
	`Device(config)# end`
		Returns to the privileged EXEC mode.

Step 5	**show platform hardware fed switch** {*switch-number* | *active* | *standby*} **qos que stats internal cpu policer** **Example:** `Device# show platform hardware fed switch 1 qos que stat internal cpu policer`	Displays the rates configured for the various traffic types.

Related Topics

- User-Configurable Aspects of CoPP, on page 5
- Restrictions for CoPP, on page 1
- Example: Enabling a CPU Queue or Changing the Policer Rate of a CPU Queue, on page 10
- Example: Disabling a CPU Queue
- Example: Setting the Default Policer Rates for All CPU Queues, on page 11

Examples for Configuring CoPP

Example: Enabling a CPU Queue or Changing the Policer Rate of a CPU Queue

This example shows how to enable a CPU queue or to change the policer rate of a CPU queue. Here the **`class system-cpp-police-protocol-snooping`** CPU queue is enabled with the policer rate of **`2000 pps`** .

```
Device> enable
Device# configure terminal
Device(config)# policy-map system-cpp-policy
Device(config-pmap)# class system-cpp-police-protocol-snooping
Device(config-pmap-c)# police rate 2000 pps
Device(config-pmap-c-police)# end


Device# show policy-map control-plane
Control Plane

  Service-policy input: system-cpp-policy

    <output truncated>


    Class-map: system-cpp-police-dot1x-auth (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: none
      police:
          rate 1000 pps, burst 244 packets
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          drop

    Class-map: system-cpp-police-protocol-snooping (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: none
      police:
          rate 2000 pps, burst 488 packets
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          drop

    <output truncated>

    Class-map: class-default (match-any)
```

```
        0 packets, 0 bytes
        5 minute offered rate 0000 bps, drop rate 0000 bps
        Match: any
```

Related Topics

Enabling a CPU Queue or Changing the Policer Rate, on page 6
Disabling a CPU Queue, on page 8
Setting the Default Policer Rates for All CPU Queues, on page 9
User-Configurable Aspects of CoPP, on page 5

Example: Setting the Default Policer Rates for All CPU Queues

This example shows how to set the policer rates for all CPU queues to their default and then verify the setting.

```
Device> enable
Device# configure terminal
Device(config)# cpp system-default
Defaulting CPP : Policer rate for all classes will be set to their defaults
Device(config)# end
```

```
Device# show platform hardware fed switch 1 qos queue stats internal cpu policer
CPU Queue Statistics
===========================================================================================

                                              (default)  (set)     Queue        Queue
QId PlcIdx  Queue Name               Enabled   Rate      Rate     Drop(Bytes)  Drop(Frames)
-------------------------------------------------------------------------------------------
0    11      DOT1X Auth                  Yes     1000      1000     0            0

1    1       L2 Control                  Yes     2000      2000     0            0

2    14      Forus traffic               Yes     4000      4000     0            0

3    0       ICMP GEN                    Yes     600       600      0            0

4    2       Routing Control             Yes     5400      5400     0            0

5    14      Forus Address resolution    Yes     4000      4000     0            0

6    0       ICMP Redirect               Yes     600       600      0            0

7    16      Inter FED Traffic           Yes     2000      2000     0            0

8    4       L2 LVX Cont Pack            Yes     1000      1000     0            0

9    16      EWLC Control                Yes     2000      2000     0            0

10   16      EWLC Data                   Yes     2000      2000     0            0

11   13      L2 LVX Data Pack            Yes     1000      1000     0            0

12   0       BROADCAST                   Yes     600       600      0            0

13   10      Openflow                    Yes     100       100      0            0

14   13      Sw forwarding               Yes     1000      1000     0            0

15   8       Topology Control            Yes     13000     13000    0            0
```

```
16   12     Proto Snooping             Yes     2000      2000     0            0

17   6      DHCP Snooping              Yes     500       500      0            0

18   9      Transit Traffic            Yes     500       500      0            0

19   10     RPF Failed                 Yes     100       100      0            0

20   15     MCAST END STATION          Yes     2000      2000     0            0

21   13     LOGGING                    Yes     1000      1000     0            0

22   7      Punt Webauth               Yes     1000      1000     0            0

23   18     High Rate App              Yes     13000     13000    0            0

24   10     Exception                  Yes     100       100      0            0

25   3      System Critical            Yes     1000      1000     0            0

26   10     NFL SAMPLED DATA           Yes     100       200      0            0

27   2      Low Latency                Yes     5400      5400     0            0

28   10     EGR Exception              Yes     100       100      0            0

29   5      Stackwise Virtual OOB      Yes     8000      8000     0            0

30   9      MCAST Data                 Yes     500       500      0            0

31   10     Gold Pkt                   Yes     100       100      0            0

* NOTE: CPU queue policer rates are configured to the closest hardware supported value

                      CPU Queue Policer Statistics
====================================================================
Policer    Policer Accept   Policer Accept  Policer Drop  Policer Drop
  Index        Bytes           Frames        Bytes          Frames
-------------------------------------------------------------------
0          0                0               0             0
1          0                0               0             0
2          0                0               0             0
3          0                0               0             0
4          0                0               0             0
5          0                0               0             0
6          0                0               0             0
7          0                0               0             0
8          0                0               0             0
9          0                0               0             0
10         0                0               0             0
11         0                0               0             0
12         0                0               0             0
13         0                0               0             0
14         0                0               0             0
15         0                0               0             0
16         0                0               0             0
17         0                0               0             0
18         0                0               0             0

                        CPP Classes to queue map
======================================================================================
PlcIdx CPP Class                                :  Queues
```

```
-------------------------------------------------------------------------------------
0      system-cpp-police-data                   :  ICMP GEN/BROADCAST/ICMP Redirect/
10     system-cpp-police-sys-data               :  Openflow/Exception/EGR Exception/NFL
SAMPLED DATA/Gold Pkt/RPF Failed/
13     system-cpp-police-sw-forward             :  Sw forwarding/LOGGING/L2 LVX Data Pack/
9      system-cpp-police-multicast              :  Transit Traffic/MCAST Data/
15     system-cpp-police-multicast-end-station  :  MCAST END STATION /
7      system-cpp-police-punt-webauth           :  Punt Webauth/
1      system-cpp-police-l2-control             :  L2 Control/
2      system-cpp-police-routing-control        :  Routing Control/Low Latency/
3      system-cpp-police-system-critical        :  System Critical/
4      system-cpp-police-l2lvx-control          :  L2 LVX Cont Pack/
8      system-cpp-police-topology-control       :  Topology Control/
11     system-cpp-police-dot1x-auth             :  DOT1X Auth/
12     system-cpp-police-protocol-snooping      :  Proto Snooping/
6      system-cpp-police-dhcp-snooping          :  DHCP Snooping/
14     system-cpp-police-forus                  :  Forus Address resolution/Forus traffic/
5      system-cpp-police-stackwise-virt-control :  Stackwise Virtual OOB/
16     system-cpp-default                      :  Inter FED Traffic/EWLC Control/EWLC Data/
18     system-cpp-police-high-rate-app          :  High Rate App/
```

Related Topics

Enabling a CPU Queue or Changing the Policer Rate, on page 6
Disabling a CPU Queue, on page 8
Setting the Default Policer Rates for All CPU Queues, on page 9
User-Configurable Aspects of CoPP, on page 5

Monitoring CoPP

Use these commands to display policer settings, such as, traffic types and policer rates (user-configured and default rates) for CPU queues:

Command	Purpose
show policy-map control-plane	Displays the rates configured for the various traffic types
show policy-map system-cpp-policy	Displays all the classes configured under system-cpp policy, and policer rates
show platform hardware fed switch {*switch-number* | *active* | *standby*} **qos que stats internal cpu policer**	Displays the rates configured for the various traffic types
show platform software fed {*switch-number* | *active* | *standby*} **qos policy target status**	Displays information about policy status and the target port type.

Feature History and Information For CoPP

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.

Feature	Release	Feature Information
Control Plane Policing (CoPP) or CPP	Cisco IOS XE 3.2SE	This feature was introduced.
CLI configuration for CoPP	Cisco IOS XE Denali 16.1.2	This feature was made user-configurable. CLI configuration options to enable and disable CPU queues, to change the policer rate, and to set policer rates to default.
User-defined class maps	Cisco IOS XE Everest 16.5.1a	Starting with this release, you can create class maps (with filters) and add these user-defined class maps to system-cpp-policy.
Changes in system-defined values for CoPP	Cisco IOS XE Everest 16.6.1	These new system-defined classes were introduced: • system-cpp-police-stackwise-virt-control • system-cpp-police-l2lvx-control These new CPU queues were added to the existing system-cpp-default class: • WK_CPU_Q_UNUSED (7) • WK_CPU_Q_EWLC_CONTROL(9) • WK_CPU_Q_EWLC_DATA(10) This new CPU queues was added to the existing system-cpp-police-sw-forward: WK_CPU_Q_L2_LVX_DATA_PACK (11) This CPU queue is no longer available: WK_CPU_Q_SGT_CACHE_FULL(27)

Feature	Release	Feature Information
Changes in system-defined values for CoPP	Cisco IOS XE Fuji 16.8.1a	This new system-defined class was introduced: system-cpp-police-dhcp-snooping This new CPU queue was added to the existing system-cpp-default class: WK_CPU_Q_INTER_FED_TRAFFIC These CPU queues are no longer available: • WK_CPU_Q_SHOW_FORWARD • WK_CPU_Q_UNUSED The default policer rate (pps) for some CPU queues has changed: • The default rate for WK_CPU_Q_EXCEPTION(24) was changed to 100 • The default rate for all the CPU queues under system-cpp-default was increased to 2000. • The default rate for all the CPU queues under system-cpp-police-forus was increased to 4000.
Changes in system-defined values for CoPP	Cisco IOS XE Fuji 16.9.1	Starting with this release, eighteen system-defined classes are created under `system-cpp-policy`. These new system-defined classes were introduced: • system-cpp-police-high-rate-app • system-cpp-police-system-critical This was added to class system-cpp-police-sys- data: CPU queue WK_CPU_Q_OPENFLOW (13). This CPU queue is no longer available: WK_CPU_Q_LEARNING_CACHE_OVFL(13). This system-defined class is no longer available: system-cpp-police-control-low-priority