コントロール プレーン ポリシングの設定

CoPP の制約事項

コントロール プレーン ポリシング（CoPP）の制約事項は、次のとおりです。

- 入力 CoPP だけがサポートされます。**system-cpp-policy** ポリシーマップは、入力方向でのみ、コントロール プレーン インターフェイスで使用可能です。
- コントロール プレーン インターフェイスにインストールできるのは、**system-cpp-policy** ポリシーマップのみです。
- **system-cpp-policy** ポリシーマップおよび 17 個のシステム定義のクラスは、変更または削除することはできません。
- **system-cpp-policy** ポリシーマップの下で許可されるのは、**police** アクションのみです。システム定義クラスのポリシングレートは、秒単位のパケット/秒（pps）でのみ設定する必要があります。ユーザ定義のクラスマップの場合は、ビット/秒（bps）のみで設定する必要があります。
- システム定義のクラスマップのポリサーを無効にしないこと、つまり **no police rate** *rate* **pps** コマンドを設定しないことを推奨します。これを行うと、CPU へのトラフィックが多い場合に、システム全体の正常性に影響します。さらに、システム定義のクラスマップのポリサーレートを無効にした場合でも、システム起動プロセスを保護するために、システムはシステムのブートアップ後にデフォルトのポリサーレートに自動的に戻ります。
- 1 つ以上の CPU キューがそれぞれのクラス マップの一部となります。複数の CPU キューが 1 つのクラスマップに属している場合、クラスマップのポリサー レートを変更すると、そのクラス マップに属しているすべての CPU キューに影響します。同様に、クラスマッ

プでポリサーを無効にすると、そのクラスマップに属するすべてのキューが無効になります。各クラス マップに属する CPU キューの詳細については、表 1 : CoPP のシステム定義された値 （3 ページ） を参照してください。

- system-cpp ポリシーの下で設定されたクラスがデフォルト値のままの場合、それらのクラスに関する情報は **show run** コマンドで表示されません。代わりに **show policy-map system-cpp-policy** または **show policy-map control-plane** コマンドを使用します。

  引き続き **show run** コマンドを使用して、カスタムポリシーに関する情報を表示できます。

CoPP の概要

この章では、コントロール プレーン ポリシング（CoPP）がdeviceで機能する仕組みと、その設定方法について説明します。

CoPP の概要

CoPP 機能は、不要なトラフィックおよび DoS 攻撃から CPU を保護するdeviceのセキュリティを向上させます。また、他の優先順位の低い大量のトラフィックによって発生するトラフィックのドロップから、制御および管理トラフィックを保護することもできます。

deviceは通常、3 つの操作プレーンにセグメント化され、それぞれに独自の目的があります。

- データ パケットを転送するための、データ プレーン。
- データを適切にルーティングするための、コントロール プレーン。
- ネットワーク要素を管理するための、管理プレーン。

CoPP を使用することで、大半の CPU 行きトラフィックを保護し、ルーティングの安定性と信頼性を確保し、パケットを確実に配信することができます。特に重要なのは、DoS 攻撃から CPU を保護するために CoPP を使用できることです。

CoPP は、モジュラ QoS コマンドライン インターフェイス（MQC）および CPU キューを使用して、これらの目的を達成します。さまざまなタイプのコントロール プレーン トラフィックが特定の条件に基づいてグループ化され、CPU キューに割り当てられます。ハードウェアに専用のポリサーを設定することで、これらの CPU キューを管理できます。たとえば、特定の CPU キュー（トラフィック タイプ）のポリサー レートを変更したり、特定のタイプのトラフィックに対するポリサーを無効にしたりできます。

ポリサーはハードウェアに設定されていますが、CoPP は CPU のパフォーマンスやデータ プレーンのパフォーマンスには影響しません。しかし、CPU に着信するパケット数は制限されるため、CPU 負荷が制御されます。これは、ハードウェアからのパケットを待っているサービスが、より制御された着信パケットのレート（ユーザ設定可能なレート）を確認する可能性があることを意味します。

システム定義の CoPP の特徴

deviceの初回の電源投入時は、システムによって次のタスクが自動的に実行されます。

- ポリシーマップ **system-cpp-policy** を検索します。見つからない場合、システムはそれを作成し、コントロールプレーンにインストールします。
- **system-cpp-policy** の下に 17 のクラスマップを作成します。

  次にdeviceの電源を入れたときに、すでに作成済みのポリシーとクラスマップがシステムによって検出されます
- デフォルトで、すべての CPU キューをそれぞれのデフォルトレートで有効にします。デフォルトのレートを「*CoPP* のシステム定義値」の表に示します。

次の表に、device をロードしたときにシステムが作成するクラスマップを示します。各クラスマップに対応するポリサーと、各クラスマップの下にグループ化された 1 つ以上の CPU キューを示します。ポリサーへのクラスマップの 1 対 1 のマッピングと、CPU キューへのクラスマップの 1 対多マッピングがあります。

表 *1 : CoPP* のシステム定義された値

クラス マップ名	ポリサー インデックス（ポリサー No.）	CPU キュー（キュー No.）	デフォルトのポリサーレート（pps）
system-cpp- police-data	WK_CPP_POLICE_DATA(0)	WK_CPU_Q_ICMP_GEN(3) WK_CPU_Q_BROADCAST(12) WK_CPU_Q_ICMP_REDIRECT (6)	600 600 600
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) WK_CPU_Q_LOW_LATENCY (27)	5400 5400
system-cpp-police-control-low-priority	WK_CPP_POLICE_CO NTROL_LOW_PRI(3)	WK_CPU_Q_GENERAL_PUNT(25)	200
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) WK_CPU_Q_MCAST_DATA(30)	500 500

クラス マップ名	ポリサー インデックス（ポリサー No.）	CPU キュー（キュー No.）	デフォルトのポリサーレート（pps）
system-cpp-police-sys- data	WK_CPP_POLICE_SYS_DATA (10)	WK_CPU_Q_LEARNING_CACHE_OVFL(13)	100
		WK_CPU_Q_CRYPTO_CONTROL(23)	100
		WK_CPU_Q_EXCEPTION(24)	200
		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-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)	1000
		WK_CPU_Q_FORUS_TRAFFIC(2)	1000
system-cpp-police-multicast-end-station	WK_CPP_POLICE_MULTICAST_END_STATION(6)	WK_CPU_Q_MCAST_END_STATION_SERVICE(20)	2000
system-cpp-default	WK_CPP_POLICE_DEFAULT_POLICER	WK_CPU_Q_DHCP_SNOOPING(17)	1000
		WK_CPU_Q_UNUSED (7)	1000
		WK_CPU_Q_EWLC_CONTROL(9)	1000
		WK_CPU_Q_EWLC_DATA(10)	1000
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

ユーザ設定可能な CoPP の特徴

次のタスクを実行して、コントロール プレーン トラフィックを管理できます。

（注） すべての ystem-cpp-policy コンフィギュレーションは、再起動後も保持されるように保存する必要があります。

CPU キューのポリサーの有効化と無効化

CPU キューのポリサーを有効にするには、system-cpp-policy ポリシーマップ内で、対応するクラスマップの下にポリサーアクション（パケット/秒）を設定します。

CPU キューのポリサーを無効にするには、system-cpp-policy ポリシーマップ内で、対応するクラスマップの下のポリサーアクションを削除します。

（注） デフォルトのポリサーがすでに存在する場合は、その削除を慎重に考慮して制御します。そのようにしないと、システムが CPU ホグや制御パケットドロップなどのその他の異常を検出する場合があります。

ポリサーレートの変更

これは、system-cpp-policy ポリシーマップ内で、対応するクラスマップの下にポリサーレートアクション（パケット/秒単位）を設定することで実行できます。

ポリサーレートをデフォルトに設定

グローバル コンフィギュレーション モードで **cpp system-default** コマンドを入力することによって、CPU キューのポリサーをデフォルト値に設定します。

ユーザ定義のクラスマップの作成

特定のトラフィッククラスに指定されたクラスマップがなく、このトラフィックを保護する場合は、そのようなトラフィックパケット用の特定のクラスマップ（フィルタ付き）を作成し、これらのユーザ定義のクラスマップを system-cpp-policy に追加できます。

system-cpp-policy が入力方向に適用されている間、転送エンジンドライバ（FED）はユーザ定義のクラスマップのポリサーを出力に変更します。したがって、すべてのユーザ定義クラスのフィルタとポリサーは、それぞれ出力分類とアクションとして適用する必要があります。ポリシーマップ自体は、この方向の変更による影響を受けません。

ユーザ定義のクラスマップを system-cpp-policy に追加すると、システムは自動的に（コントロールプレーンに加えて）32 個のすべての CPU キューにそれをインストールします。その結果、ポリシーのインスタンスは 33 個になります。これを確認するには、特権 EXEC モードで **show platform software fed switch** { *switch_number* } qos policy target status コマンドを入力します。

これらのクラスマップのポリシングレートは、アクティブキュー管理（AFD）ポリサーによって制御されます。AQM は、パケットをポートの送信キューに入れる前の、トラフィックフローのバッファ制御を提供し、特定のフローによるスイッチパケットメモリの占有が行われないよ

うにします。 AQM ポリサー機能が有効の場合、AQM ポリサーの制限を超えるユーザ定義のポリシングレートは無視されます。

ユーザ定義のクラスマップには、通常の QoS または ACL 分類フィルタがあります。

ソフトウェアバージョンのアップグレードまたはダウングレード

ソフトウェアバージョンのアップグレードと CoPP

デバイスのソフトウェアバージョンをアップグレードすると、システムは CoPP に必要な更新を確認して実行します（たとえば、`system-cpp-policy` ポリシーマップを確認し、欠落している場合は作成します）。また、アップグレードアクティビティの前後に特定のタスクを完了する必要があります。これにより、設定の更新が正しく反映され、CoPP が期待どおりに動作し続けることが保証されます。ソフトウェアのアップグレードに使用する方法に応じて、アップグレード関連のタスクはオプションのシナリオまたは推奨されるシナリオもあれば、必須のシナリオもあります。

ここでは、アップグレードのシステムアクションとユーザアクションについて説明します。また、リリース固有の警告も含まれます。

アップグレードのシステムアクション

デバイスのソフトウェアバージョンをアップグレードすると、システムはこれらのアクションを実行します。これはすべてのアップグレード方法に適用されます。

- アップグレード前のデバイスに `system-cpp-policy` ポリシーマップがなかった場合、アップグレード時にシステムはデフォルトポリシーを作成します。
- アップグレード前のデバイスに `system-cpp-policy` ポリシーマップがあった場合、アップグレード時にシステムはポリシーを再生成しません。

アップグレードのユーザアクション

アップグレードのユーザアクション（アップグレード方法に応じて）：

アップグレード方法	条件	アクション時間とアクション	目的
標準[1]	なし（None）	**アップグレード後（必須）** グローバル コンフィギュレーション モードで **cpp system-default** コマンドを入力します。	最新のデフォルトのポリサーレートを取得します。

[1] スイッチのリロードを伴うソフトウェアアップグレードの方法を指します。インストールモードまたはバンドルモードにすることができます。

ソフトウェアバージョンのダウングレードと CoPP

ダウングレードのシステムアクションとユーザアクションについて、ここで説明します。

ダウングレードのシステムアクション

デバイスのソフトウェアバージョンをダウングレードすると、これらのアクションが実行されます。これはすべてのダウングレード方法に適用されます。

- システムは `system-cpp-policy` ポリシーマップをデバイスに保持し、コントロールプレーンにインストールします。

ダウングレードのユーザアクション

ダウングレードのユーザアクション：

アップグレード方法	条件	アクション時間とアクション	目的
標準[2]	なし（None）	操作は不要です。	N/A

[2] スイッチのリロードを伴うソフトウェアアップグレードの方法を指します。インストールモードまたはバンドルモードにすることができます。

ソフトウェアバージョンをダウングレードしてから再度アップグレードする場合、適用されるシステムアクションとユーザアクションは、アップグレードについて説明したものと同じです。

CoPP の設定方法

CPU キューの有効化またはポリサー レートの変更

CPU キューを有効にし、CPU キューのポリサー レートを変更する手順は、同じです。手順は次のとおりです。

手順

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： `デバイス> enable`	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** 例：	グローバル コンフィギュレーション モードを開始します。

	コマンドまたはアクション	目的
	デバイス# **configure terminal**	
ステップ 3	**policy-map** *policy-map-name* 例： デバイス(config)# **policy-map system-cpp-policy** デバイス(config-pmap)#	ポリシーマップ コンフィギュレーション モードを開始します。
ステップ 4	**class** *class-name* 例： デバイス(config-pmap)# **class system-cpp-police-protocol-snooping** デバイス(config-pmap-c)#	クラス アクション コンフィギュレーションモードを開始します。有効にするCPUキューに対応するクラスの名前を入力します。「*CoPP* のシステム定義値」の表を参照してください。
ステップ 5	**police rate** *rate* **pps** 例： デバイス(config-pmap-c)# **police rate 100 pps** デバイス(config-pmap-c-police)#	指定したトラフィック クラスに対し、1 秒間に処理される着信パケット数の上限を指定します。 （注） 指定するレートは、指定したクラスマップに属するすべての CPU キューに適用されます。
ステップ 6	exit 例： デバイス(config-pmap-c-police)# **exit** デバイス(config-pmap-c)# **exit** デバイス(config-pmap)# **exit** デバイス(config)#	グローバル コンフィギュレーション モードに戻ります。
ステップ 7	**control-plane** 例： デバイス(config)# **control-plane** デバイス(config-cp)#	制御プレーン（config-cp）コンフィギュレーション モードを開始します。
ステップ 8	**service-policy input** *policy-name* 例： デバイス(config)# **control-plane** デバイス(config-cp)#**service-policy input system-cpp-policy** デバイス(config-cp)#	system-cpp-policy を FED にインストールします。このコマンドは、FED ポリシーを表示するために必要です。このコマンドを設定しないと、エラーになります。

	コマンドまたはアクション	目的
ステップ 9	**end** 例： デバイス(config-cp)# **end**	特権 EXEC モードに戻ります。
ステップ 10	**show policy-map control-plane** 例： デバイス# **show policy-map control-plane**	system-cpp ポリシーの下で設定されたすべてのクラス、さまざまなトラフィックタイプに設定されたレート、および統計情報を表示します。

CPU キューの無効化

CPU キューを無効にするには、次の手順を実行します。

手順

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： デバイス> **enable**	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** 例： デバイス# **configure terminal**	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**policy-map** *policy-map-name* 例： デバイス(config)# **policy-map system-cpp-policy** デバイス(config-pmap)#	ポリシー マップ コンフィギュレーション モードを開始します。
ステップ 4	**class** *class-name* 例： デバイス(config-pmap)# **class system-cpp-police-protocol-snooping** デバイス(config-pmap-c)#	クラス アクション コンフィギュレーション モードを開始します。無効にする CPU キューに対応するクラスの名前を入力します。「*CoPP* のシステム定義値」の表を参照してください。

	コマンドまたはアクション	目的
ステップ 5	**no police rate** *rate* **pps** 例： デバイス(config-pmap-c)# **no police rate 100 pps**	指定したトラフィック クラスの着信パケットの処理を無効にします。 （注） これにより、指定したクラスマップに属するすべての CPU キューが無効になります。
ステップ 6	**end** 例： デバイス(config-pmap-c)# **end**	特権 EXEC モードに戻ります。
ステップ 7	**show policy-map control-plane** 例： デバイス# **show policy-map control-plane**	system-cpp ポリシーの下で設定されたすべてのクラス、およびさまざまなトラフィックタイプと統計情報に設定されたレートを表示します。

すべての CPU キューに対するデフォルトのポリサー レートの設定

すべての CPU キューのポリサー レートをデフォルトのレートに設定するには、次の手順を実行します。

手順

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： デバイス> **enable**	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** 例： デバイス# **configure terminal**	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**cpp system-default** 例： デバイス(config)# **cpp system-default** Defaulting CPP : Policer rate for all classes will be set to their defaults	すべてのクラスのポリサー レートをデフォルトのレートに設定します。

	コマンドまたはアクション	目的
ステップ 4	**end** 例： デバイス(config)# **end**	特権 EXEC モードに戻ります。

ユーザ定義のクラスマップの作成

system-cpp-policy でユーザ定義のクラスマップを作成し、ポリサーレートを bps で設定するには、次の手順に従ってください。

手順

	コマンドまたはアクション	目的
ステップ 1	**enable** 例： デバイス> **enable**	特権 EXEC モードを有効にします。 • パスワードを入力します（要求された場合）。
ステップ 2	**configure terminal** 例： デバイス# **configure terminal**	グローバル コンフィギュレーション モードを開始します。
ステップ 3	**class-map** *class-map-name* 例： デバイス(config)# **class-map example_class** デバイス(config-cmap)#	作成するクラスマップを指定します。クラス マップ コンフィギュレーション モードを開始します。
ステップ 4	**exit** 例： Device(config-cmap)# **exit** Device(config)#	クラス マップ コンフィギュレーション モードを終了します。

	コマンドまたはアクション	目的

ステップ 5	**policy-map** *policy-map-name*
	例：
	`Device(config)# policy-map system-cpp-policy`
	`Device(config-pmap)#`
		ポリシー マップ名を入力します。ポリシー マップ コンフィギュレーション モードを開始します。

ステップ 6	**class-map** *class-map-name*
	例：
	`Device(config-pmap)# class example_class`
	`Device(config-pmap-c)#`
		クラス アクション コンフィギュレーション モードを開始します。クラスの名前を入力します。

ステップ 7	[**no**] **police rate** *target_bit_rate* 例： `Device(config-pmap-c)# police 90000`	ビットレート/秒を指定し、8000 ～ 10000000000 の値を入力します。 （注） ユーザ定義のクラスマップのポリシングレートは、10000 pps相当のトラフィックを超えることはできません。

ステップ 8	**end**
	例：
	`Device(config-pmap-c-police)# end`
	`Device#`
		特権 EXEC モードに戻ります。

ステップ 9	**show policy-map control-plane** 例： `Device# show policy-map control-plane`	system-cpp ポリシーの下で設定されたすべてのクラスを表示します。これには、ユーザ定義のクラスマップ、および設定されたレートが含まれます。

CoPP の設定例

例：CPU キューの有効化または CPU キューのポリサー レートの変更

次の例に、CPU キューを有効にする方法、または CPU キューのポリサー レートを変更する方法を示します。ここでは、**class system-cpp-police-protocol-snooping** CPU キューが有効になり、ポリサー レートは **2000 pps** です。

```
デバイス> enable
デバイス# configure terminal
デバイス(config)# policy-map system-cpp-policy
```

```
デバイス(config-pmap)# class system-cpp-police-protocol-snooping
デバイス(config-pmap-c)# police rate 2000 pps
デバイス(config-pmap-c-police)# end


デバイス# 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
```

例：CPU キューの無効化

次に、CPU キューをディセーブルにする例を示します。ここでは、**class system-cpp-police-protocol-snooping** CPU キューが無効になります。

```
デバイス> enable
デバイス# configure terminal
デバイス(config)# policy-map system-cpp-policy
デバイス(config-pmap)# class system-cpp-police-protocol-snooping
デバイス(config-pmap-c)# no police rate 100 pps
デバイス(config-pmap-c)# end


デバイス# show running-config | begin system-cpp-policy

policy-map system-cpp-policy
```

```
 class system-cpp-police-data
  police rate 200 pps
 class system-cpp-police-sys-data
  police rate 100 pps
 class system-cpp-police-sw-forward
  police rate 1000 pps
 class system-cpp-police-multicast
  police rate 500 pps
 class system-cpp-police-multicast-end-station
  police rate 2000 pps
 class system-cpp-police-punt-webauth
 class system-cpp-police-l2-control
 class system-cpp-police-routing-control
  police rate 500 pps
 class system-cpp-police-control-low-priority
 class system-cpp-police-wireless-priority1
 class system-cpp-police-wireless-priority2
 class system-cpp-police-wireless-priority3-4-5
 class system-cpp-police-topology-control
 class system-cpp-police-dot1x-auth
 class system-cpp-police-protocol-snooping
 class system-cpp-police-forus
 class system-cpp-default

<output truncated>
```

例：すべてのCPUキューに対するデフォルトのポリサーレートの設定

次に、すべての CPU キューのポリサー レートをデフォルトに設定し、その後に設定を確認する例を示します。

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

例：ユーザ定義のクラスマップの作成

デバイス

次に、ユーザ定義のクラスマップを作成し、それを system-cpp-policy に適用して、ポリシーの適用場所に関する情報を表示する例を示します。

ユーザ定義のクラスマップが system-cpp-policy に適用されます。つまり、ユーザ定義のクラスマップ class-cpp-user に一致する制御トラフィックは、ユーザ定義のクラスマップでの集約ポリサーによって異なります。ユーザ定義トラフィック クラスの統計情報はバイト単位で報告されます。

```
Device> enable
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# class-map match-any class-cpp-user
Device(config-cmap)# match dscp cs1
Device(config-cmap)# exit
Device(config)# policy-map system-cpp-policy
Device(config-pmap)# class class-cpp-user
```

```
Device(config-pmap-c)# police rate 2m bps
Device(config-pmap-c-police)# end

Device# show policy-map control-plane
<output truncated>
Class-map: class-cpp-user (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match:  dscp cs1 (8)
      police:
          rate 2000000 bps, burst 62500 bytes
        conformed 0 bytes; actions:
          transmit
        exceeded 0 bytes; actions:
          drop
        conformed 0000 bps, exceeded 0000 bps
<output truncated>
```

ユーザ定義のクラスマップを system-cpp-policy に追加すると、システムはコントロールプレーンに加えて、32 個すべての CPU キューに自動的にそれをインストールします（結果としてポリシーのインスタンスは 33 個になります）。

入力で system-cpp-policy が適用されていても、方向が出力（OUT）としてどのように表示されるかに注意してください。

```
Device# show platform software fed switch active qos policy target status

TCG status summary:

Loc Interface             IIF-ID           Dir State:(cfg,opr) Policy
--- --------------------- ---------------- --- --------------- --------------------
?:255 Control Plane       0x00000001000001 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-0          0x0000000100000d OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-1          0x0000000100000e OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-2          0x0000000100000f OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-3          0x00000001000010 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-4          0x00000001000011 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-5          0x00000001000012 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-6          0x00000001000013 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-7          0x00000001000014 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-8          0x00000001000015 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-9          0x00000001000016 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-10         0x00000001000017 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-11         0x00000001000018 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-12         0x00000001000019 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-13         0x0000000100001a OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-14         0x0000000100001b OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-15         0x0000000100001c OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-16         0x0000000100001d OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-17         0x0000000100001e OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-18         0x0000000100001f OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-19         0x00000001000020 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-20         0x00000001000021 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-21         0x00000001000022 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-22         0x00000001000023 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-23         0x00000001000024 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-24         0x00000001000025 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-25         0x00000001000026 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-26         0x00000001000027 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-27         0x00000001000028 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-28         0x00000001000029 OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-29         0x0000000100002a OUT VALID,SET_INHW  system-cpp-policy
```

```
?:0 CoPP-Queue-30          0x0000000100002b OUT VALID,SET_INHW  system-cpp-policy
?:0 CoPP-Queue-31          0x0000000100002c OUT VALID,SET_INHW  system-cpp-policy
```

CoPP のモニタリング

CPUキューのトラフィックタイプやポリサーレート（ユーザが設定したレートやデフォルトのレート）などのポリサー設定を表示するには、次のコマンドを使用します。

コマンド	目的
show policy-map control-plane	さまざまなトラフィックタイプに設定されたレートを表示します。
show policy-map system-cpp-policy	system-cpp ポリシーの下で設定されたすべてのクラスとポリサーレートを表示します。
show platform hardware fed switch {*switch-number* } **qos que stats internal cpu policer**	さまざまなトラフィックタイプに設定されたレートを表示します。
show platform software fed {*switch-number* } **qos policy target status**	ポリシーステータスとターゲットポートタイプに関する情報を表示します。

CoPP の機能履歴と情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

機能	リリース	機能情報
コントロールプレーンポリシング（CoPP）または CPP	Cisco IOS XE 3.3SE	この機能が導入されました。
CoPP の CLI コンフィギュレーション	Cisco IOS XE Denali 16.1.2	この機能はユーザ設定可能です。CPU キューの有効化および無効化、ポリサー レートの変更、およびポリサー レートのデフォルトへの設定を行うための CLI 設定オプション。
ユーザ定義のクラスマップ	Cisco IOS XE Everest 16.5.1a	このリリース以降、（フィルタ付きの）クラスマップを作成し、これらのユーザ定義のクラスマップを system-cpp-policy に追加できます。

機能	リリース	機能情報
CoPP のシステム定義値の変更	Cisco IOS XE Everest 16.6.1	次の新しいシステム定義のクラスが導入されました。 • system-cpp-police-stackwise-virt-control • system-cpp-police-l2lvx-control 次の新しい CPU キューが既存の system-cpp-default クラスに追加されました。 • WK_CPU_Q_UNUSED (7) • WK_CPU_Q_EWLC_CONTROL(9) • WK_CPU_Q_EWLC_DATA(10) この新しい CPU キューが既存の system-cpp-police-sw-forward に追加されました。WK_CPU_Q_L2_LVX_DATA_PACK (11) この CPU キューは使用できなくなりました。WK_CPU_Q_SGT_CACHE_FULL(27)