



コントロールプレーンポリシングの設定

- [CoPP の制約事項 \(1 ページ\)](#)
- [CoPP の概要 \(2 ページ\)](#)
- [CoPP の設定方法 \(12 ページ\)](#)
- [CoPP の設定例 \(16 ページ\)](#)
- [CoPP のモニタリング \(18 ページ\)](#)
- [CoPP の機能の履歴 \(18 ページ\)](#)

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、1秒あたりのパケット数 (pps) でのみ設定する必要があります。
- 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、「*CoPP* のシステム定義値」の表を参照してください。
- 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。

- `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。
- 大量の CPU バウンドパケットを使用するプロトコルは、同じクラスの他のプロトコルに影響を与える可能性があります。これらのプロトコルの一部は同じポリサーを共有するためです。たとえば、Address Resolution Protocol (ARP) は、`system-cpp-police-forus` クラスの Telnet、Internet Control Message Protocol (ICMP)、SSH、FTP、SNMP などのホストプロトコルの配列と 4000 個のハードウェアポリサーを共有します。ARP ポイズニングまたは ICMP 攻撃が発生すると、ハードウェアポリサーは、4000 パケット/秒を超える着信トラフィックのロットリングを開始し、CPU とシステムの全体的な完全性を保護します。その結果、ARP および ICMP ホストプロトコルは、同じクラスを共有する他のホストプロトコルとともにドロップされます。
- ユーザ定義のクラスマップの作成はサポートされていません。

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) がデバイス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 Denial of Service (DoS) 攻撃から CPU を保護することでデバイス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、制御トラフィックおよび管理トラフィックを保護することもできます。

デバイスは通常、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

デバイス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て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 18 個のクラスマップを作成します。
 次回デバイスの電源を入れたときに、すでに作成済みのポリシーマップ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。

system-cpp-policy ポリシーマップはシステム デフォルト ポリシー マップであり、通常はデバイスのスタートアップコンフィギュレーションに明示的に保存する必要はありません。ただし、スタンバイデバイスとのバルク同期に失敗すると、コンフィギュレーションがスタートアップコンフィギュレーションから消去される可能性があります。この場合、手動で

system-cpp-policy ポリシーマップをスタートアップコンフィギュレーションに保存する必要があります。 **show running-config** 特権 EXEC コマンドを使用して、保存されていることを確認します。

```
policy-map system-cpp-policy
```

次の表に、デバイス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。クラスマップとポリサーには1対1のマッピングがあり、1つ以上のCPUキューがクラスマップにマッピングします。

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.)
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)

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.)
system-cpp-police-l2- control	WK_CPP_POLICE_L2_CONTROL(1)	WK_CPU_Q_L2_CONTROL(1)
system-cpp-police-routing-control	WK_CPP_POLICE_ROUTING_CONTROL(2)	WK_CPU_Q_ROUTING_CONTROL(4) WK_CPU_Q_LOW_LATENCY (27)
system-cpp-police-punt-webauth	WK_CPP_POLICE_PUNT_WEBAUTH(7)	WK_CPU_Q_PUNT_WEBAUTH(22)
system-cpp-police- topology-control	WK_CPP_POLICE_TOPOLOGY_CONTROL(8)	WK_CPU_Q_TOPOLOGY_CONTROL(15)
system-cpp-police- multicast	WK_CPP_POLICE_MULTICAST(9)	WK_CPU_Q_TRANSIT_TRAFFIC(18) WK_CPU_Q_MCAST_DATA(30)
system-cpp-police-sys- data	WK_CPP_POLICE_SYS_DATA(10)	WK_CPU_Q_OPENFLOW (13) WK_CPU_Q_CRYPTO_CONTROL(23) WK_CPU_Q_EXCEPTION(24) WK_CPU_Q_EGR_EXCEPTION(28) WK_CPU_Q_NFL_SAMPLED_DATA(26) WK_CPU_Q_GOLD_PKT(31) WK_CPU_Q_RPF_FAILED(19)
system-cpp-police-dot1x-auth	WK_CPP_POLICE_DOT1X(11)	WK_CPU_Q_DOT1X_AUTH(0)
system-cpp-police- protocol-snooping	WK_CPP_POLICE_PR(12)	WK_CPU_Q_PROTO_SNOOPING(16)
system-cpp-police-dhcp-snooping	WK_CPP_DHCP_SNOOPING(6)	WK_CPU_Q_DHCP_SNOOPING(17)
system-cpp-police-sw-forward	WK_CPP_POLICE_SW_FWD(13)	WK_CPU_Q_SW_FORWARDING_Q(14) WK_CPU_Q_LOGGING(21) WK_CPU_Q_L2_LVX_DATA_PACK (11)
system-cpp-police-forus	WK_CPP_POLICE_FORUS(14)	WK_CPU_Q_FORUS_ADDR_RESOLUTION(5) WK_CPU_Q_FORUS_TRAFFIC(2)
system-cpp-police- multicast-end-station	WK_CPP_POLICE_MULTICAST_SNOOPING(15)	WK_CPU_Q_MCAST_END_STATION_SERVICE(20)
system-cpp-default	WK_CPP_POLICE_DEFAULT_POLICER(16)	WK_CPU_Q_INTER_FED_TRAFFIC(7) WK_CPU_Q_EWLC_CONTROL(9) WK_CPU_Q_EWLC_DATA(10)
system-cpp-police-stackwise-virt-control	WK_CPP_STACKWISE_VIRTUAL_CONTROL(16)	WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL (29)

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.)
system-cpp-police-l2lvs-control	WK_CPP_L2_LVX_CONT_PACK(4)	WK_CPU_Q_L2_LVX_CONT_PACK(8)
system-cpp-police-high-rate-app	WK_CPP_HIGH_RATE_APP(18)	WK_CPU_Q_HIGH_RATE_APP(23)
system-cpp-police-system-critical	WK_CPP_SYSTEM_CRITICAL(3)	WK_CPU_Q_SYSTEM_CRITICAL(25)

次の表に、CPU キューと、各 CPU キューに関連付けられた機能を示します。

表 2: CPU キューと関連機能

CPU キュー (キュー No.)	機能
WK_CPU_Q_DOT1X_AUTH(0)	IEEE 802.1x ポートベースの認証
WK_CPU_Q_L2_CONTROL(1)	ダイナミック トランキンング プロトコル (DTP) VLAN トランキンング プロトコル (VTP) ポート集約プロトコル (PAgP) Client Information Signalling Protocol (CISP) メッセージセッション リレー プロトコル マルチ VLAN 登録プロトコル (MVRP) Metropolitan Mobile Network (MMN) リンクレベル検出プロトコル (LLDP) 単一方向リンク検出 (UDLD) リンク集約制御プロトコル (LACP) Cisco Discovery Protocol (CDP) スパニング ツリー プロトコル (STP)
WK_CPU_Q_FORUS_TRAFFIC(2)	Telnet、Pingv4 および Pingv6、SNMP などのホスト キープアライブ/ループバック検出 開始 - インターネット キー エクスチェンジ (IKE) プロトコル (IPSec)
WK_CPU_Q_ICMP_GEN(3)	ICMP - 接続先到達不能 ICMP - TTL 期限切れ

CPU キュー (キュー No.)	機能
WK_CPU_Q_ROUTING_CONTROL(4)	

CPU キュー (キュー No.)	機能
	Routing Information Protocol バージョン 1 (RIPv1) RIPv2 Interior Gateway Routing Protocol (IGRP) Border Gateway Protocol (BGP) PIM-UDP 仮想ルータ冗長プロトコル (VRRP) Hot Standby Router Protocol バージョン 1 (HSRPv1) HSRPv2 ゲートウェイロードバランシングプロトコル (GLBP) ラベル配布プロトコル (LDP) Web Cache Communication Protocol (WCCP) 次世代 Routing Information Protocol (RIPng) Open Shortest Path First (OSPF) Open Shortest Path First バージョン 3 (OSPFv3) Enhanced Interior Gateway Routing Protocol (EIGRP) Enhanced Interior Gateway Routing Protocol バージョン 6 (EIGRPv6) DHCPv6 プロトコルに依存しないマルチキャスト (PIM) Protocol Independent Multicast バージョン 6 (PIMv6) 次世代 Hot Standby Router Protocol (HSRPng) IPv6 制御 Generic Routing Encapsulation (GRE) キーペアライブ

CPU キュー (キュー No.)	機能
	ネットワークアドレス変換 (NAT) パント Intermediate System-to-Intermediate System (IS-IS)
WK_CPU_Q_FORUS_ADDR_RESOLUTION(5)	アドレス解決プロトコル (ARP) IPv6 ネイバーアドバタイズメントおよび ネイバー勧誘
WK_CPU_Q_ICMP_REDIRECT(6)	インターネット制御メッセージプロトコル (ICMP) リダイレクト
WK_CPU_Q_INTER_FED_TRAFFIC(7)	内部通信用のレイヤ2ブリッジドメイン 注入。
WK_CPU_Q_L2_LVX_CONT_PACK(8)	Exchange ID (XID) パケット
WK_CPU_Q_EWLC_CONTROL(9)	Embedded Wirelss Controller (eWLC) [ワイヤレスアクセスポイントの制御とプロビジョニング (CAPWAP) (UDP 5246)]
WK_CPU_Q_EWLC_DATA(10)	eWLC データパケット (CAPWAP DATA、UDP 5247)
WK_CPU_Q_L2_LVX_DATA_PACK(11)	不明なユニキャストパケットがマップ要求のためにパントされました。
WK_CPU_Q_BROADCAST(12)	すべてのタイプのブロードキャスト
WK_CPU_Q_OPENFLOW(13)	学習キャッシュオーバーフロー (レイヤ2+レイヤ3)
WK_CPU_Q_CONTROLLER_PUNT(14)	データ - アクセスコントロールリスト (ACL) フル データ - IPv4 オプション データ - IPv6 ホップバイホップ データ - リソース不足/すべてをキャッチ データ - リバースパスフォワーディング (RPF) が不完全 収集パケット

CPU キュー (キュー No.)	機能
WK_CPU_Q_TOPOLOGY_CONTROL(15)	スパニング ツリー プロトコル (STP) Resilient Ethernet Protocol (REP) Shared Spanning Tree Protocol (SSTP)
WK_CPU_Q_PROTO_SNOOPING(16)	ダイナミック ARP インスペクション (DAI) の Address Resolution Protocol (ARP) スヌーピング
WK_CPU_Q_DHCP_SNOOPING(17)	DHCP スヌーピング
WK_CPU_Q_TRANSIT_TRAFFIC(18)	これは、ソフトウェアパスで処理する必要がある NAT によってパントされたパケットに使用されます。
WK_CPU_Q_RPF_FAILED(19)	データ - mRPF (マルチキャスト RPF) が失敗しました
WK_CPU_Q_MCAST_END_STATION_SERVICE(20)	Internet Group Management Protocol (IGMP) /Multicast Listener Discovery (MLD) 制御
WK_CPU_Q_LOGGING(21)	アクセスコントロールリスト (ACL) ロギング
WK_CPU_Q_PUNT_WEBAUTH(22)	Web 認証
WK_CPU_Q_HIGH_RATE_APP(23)	ブロードキャスト
WK_CPU_Q_EXCEPTION(24)	IKE の表示 IP ラーニング違反 IP ポートのセキュリティ違反 IP スタティックアドレス違反 IPv6 スコープチェック リモートコピープロトコル (RCP) 例外 ユニキャスト RPF 失敗
WK_CPU_Q_SYSTEM_CRITICAL(25)	メディアシグナリング/ワイヤレスプロキシ ARP
WK_CPU_Q_NFL_SAMPLED_DATA(26)	Netflow サンプルデータと Media Services Proxy (MSP)

CPU キュー (キュー No.)	機能
WK_CPU_Q_LOW_LATENCY(27)	双方向フォワーディング検出 (BFD) 、 Precision Time Protocol (PTP)
WK_CPU_Q_EGR_EXCEPTION(28)	出力解決例外
WK_CPU_Q_STACKWISE_VIRTUAL_CONTROL(29)	前面スタッキングプロトコル、つまり SVL
WK_CPU_Q_MCAST_DATA(30)	データ - (S、G) の作成 データ - ローカル結合 データ - PIM 登録 データ - SPT スイッチオーバー データ - マルチキャスト
WK_CPU_Q_GOLD_PKT(31)	Gold

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 `system-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に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 占有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。

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、システムは以下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で共通です。

- 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	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。インストーラ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	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： <code>Device> enable</code>	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： <code>Device# configure terminal</code>	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
ステップ 3	<p>policy-map <i>policy-map-name</i></p> <p>例 :</p> <pre>Device (config) # policy-map system-cpp-policy Device (config-pmap) #</pre>	<p>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。</p>
ステップ 4	<p>class <i>class-name</i></p> <p>例 :</p> <pre>Device (config-pmap) # class system-cpp-police-protocol-snooping Device (config-pmap-c) #</pre>	<p>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。</p>
ステップ 5	<p>police rate <i>rate</i> pps</p> <p>例 :</p> <pre>Device (config-pmap-c) # police rate 100 pps Device (config-pmap-c-police) #</pre>	<p>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。</p> <p>(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。</p>
ステップ 6	<p>exit</p> <p>例 :</p> <pre>Device (config-pmap-c-police) # exit Device (config-pmap-c) # exit Device (config-pmap) # exit Device (config) #</pre>	<p>グローバル コンフィギュレーションモードに戻ります。</p>
ステップ 7	<p>control-plane</p> <p>例 :</p> <pre>Device (config) # control-plane Device (config-cp) #</pre>	<p>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。</p>
ステップ 8	<p>service-policy input <i>policy-name</i></p> <p>例 :</p> <pre>Device (config) # control-plane Device (config-cp) # service-policy input system-cpp-policy Device (config-cp) #</pre>	<p>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。</p>
ステップ 9	<p>end</p> <p>例 :</p> <pre>Device (config-cp) # end</pre>	<p>特権 EXEC モードに戻ります。</p>

	コマンドまたはアクション	目的
ステップ 10	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、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： Device> 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： Device# 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： Device(config)# policy-map system-cpp-policy Device(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： Device(config-pmap)# class system-cpp-police-protocol-snooping 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。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。
ステップ 5	no police rate rate pps 例： Device(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：	特権 EXEC モードに戻ります。

	コマンドまたはアクション	目的
	Device(config-pmap-c) # end	
ステップ 7	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、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 : Device> enable	特権 EXEC モードを有効にします。 <ul style="list-style-type: none"> 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 : Device# 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 : Device(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。
ステップ 4	end 例 : Device(config)# end	特権 EXEC モードに戻ります。
ステップ 5	show platform hardware fed switch {switch-number} qos que stats internal cpu policer 例 :	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。

	コマンドまたはアクション	目的
	<pre>Device# show platform hardware fed switch 1 qos que stat internal cpu policer</pre>	

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。

```
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
```

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。

```
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)# no police rate 100 pps
Device(config-pmap-c)# end
```

```
Device# 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。

```
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
```

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。

CoPPの機能の履歴

次の表に、このモジュールで説明する機能のリリースおよび関連情報を示します。

これらの機能は、特に明記されていない限り、導入されたリリース以降のすべてのリリースで使用できます。

リリース	機能	機能情報
Cisco IOS XE Fuji 16.9.2	コントロールプレーンポリシング (CoPP) または CPP	CoPP機能によって、不要なトラフィックまたはDoSトラフィックからCPUを保護し、コントロールプレーンおよび管理トラフィックを優先させることにより、デバイスのセキュリティが向上します。 この機能には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設定オプションがあります。
Cisco IOS XE Fuji 16.9.4	システム定義のクラスマップの廃止	システム定義のクラスマップ system-cpp-police-control-low-priority は廃止されました。
Cisco IOS XE Gibraltar 16.10.1	コントロールプレーンポリシング (CoPP) または CPP	この機能は、シリーズのC9200モデルで導入されました。

Cisco Feature Navigator を使用すると、プラットフォームおよびソフトウェアイメージのサポート情報を検索できます。Cisco Feature Navigator にアクセスするには、<https://cfng.cisco.com> に進みます。

