



## Configuring IP to ATM Class of Service

---

This chapter describes how to configure IP to ATM Class of Service (CoS), a feature suite that maps QoS characteristics between IP and ATM.

For complete conceptual information, see the chapter “IP to ATM Class of Service Overview” in this book.

For a complete description of the IP to ATM CoS commands in this chapter, refer to the *Cisco IOS Quality of Service Solutions Command Reference* publication. To locate documentation of other commands that appear in this chapter, use the command reference master index, or search online.

The IP to ATM CoS feature is supported on Cisco 2600, Cisco 3600, Cisco 7200, and Cisco 7500 series routers equipped with the following hardware:

- Cisco 7200 series:
  - NPE-200 or higher (NPE-300 recommended for per-VC CBWFQ)
  - One of the following Enhanced ATM port adapters (PA-A3): T3, E3, DS3, or OC-3
- Cisco 7500 series:
  - VIP2-50
  - One of the following Enhanced ATM port adapters (PA-A3): T3, E3, DS3, or OC-3
- Cisco 2600 and Cisco 3600 series:
  - One of the following port adapters: ATM OC-3, T1 IMA, E1 IMA



**Note**

---

Per-VC weighted fair queueing (WFQ) and class-based WFQ (CBWFQ) are not available on Cisco 7500 series routers in Cisco IOS Release 12.1.

---

## IP to ATM CoS on a Single ATM VC Configuration Task List

To configure IP to ATM CoS for a single ATM VC, perform the tasks in the following sections. The first two sections are required; the remaining sections are optional.

- Defining the WRED Parameter Group (Required)
- Configuring the WRED Parameter Group (Required)
- Displaying the WRED Parameters (Optional)
- Displaying the Queueing Statistics (Optional)

The IP to ATM CoS feature requires ATM permanent virtual circuit (PVC) management.

See the end of this chapter for the section “Single ATM VC with WRED Group and IP Precedence Example.”

## Defining the WRED Parameter Group

To define the Weighted Random Early Detection (WRED) parameter group, use the following command in global configuration mode:

Command	Purpose
<code>random-detect-group</code> <i>group-name</i>	Defines the WRED or VIP-Distributed WRED (DWRED) parameter group.

## Configuring the WRED Parameter Group

To configure the exponential weight factor for the average queue size calculation for a WRED parameter group or to configure a WRED parameter group for a particular IP precedence, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	<code>random-detect-group</code> <i>group-name</i>	Specifies the WRED or DWRED parameter group.
Step 2	<code>exponential-weighting-constant</code> <i>exponent</i> or <code>precedence</code> <i>precedence min-threshold max-threshold mark-probability-denominator</i>	Configures the exponential weight factor for the average queue size calculation for the specified WRED or DWRED parameter group. or Configures the specified WRED or DWRED parameter group for a particular IP Precedence.

## Displaying the WRED Parameters

To display the configured WRED parameters, use the following command in privileged EXEC mode:

Command	Purpose
<code>show queueing random-detect</code> [ <code>interface</code> <i>atm_subinterface</i> [ <code>vc</code> [[ <i>vpi/</i> ] <i>vci</i> ]]]	Displays the parameters of every virtual circuit (VC) with WRED or DWRED enabled on the specified ATM subinterface.

## Displaying the Queueing Statistics

To display the queueing statistics of an interface, use the following command in privileged EXEC mode:

Command	Purpose
<code>show queueing interface interface-number [vc [[vpi/] vci]]</code>	Displays the queueing statistics of a specific VC on an interface.

## IP to ATM CoS on an ATM Bundle Configuration Task List

To configure IP to ATM CoS on an ATM bundle, perform the tasks in the following sections. The first four sections are required; the remaining sections are optional.

- Creating a VC Bundle (Required)
- Applying Bundle-Level Parameters (Required)
  - Configuring Bundle-Level Parameters
  - Configuring VC Class Parameters to Apply to a Bundle
  - Attaching a Class to a Bundle
- Committing a VC to a Bundle (Required)
- Applying Parameters to Individual VCs (Required)
  - Configuring a VC Bundle Member Directly
  - Configuring VC Class Parameters to Apply to a VC Bundle Member
  - Applying a VC Class to a Discrete VC Bundle Member
- Configuring a VC Not to Accept Bumped Traffic (Optional)
- Monitoring and Maintaining VC Bundles and Their VC Members (Optional)

The IP to ATM CoS feature requires ATM PVC management.

See the end of this chapter for the section “VC Bundle Configuration Using a VC Class Example.”

## Creating a VC Bundle

To create a bundle and enter bundle configuration mode in which you can assign attributes and parameters to the bundle and all of its member VCs, use the following command in subinterface configuration mode:

Command	Purpose
<code>bundle bundle-name</code>	Creates the specified bundle and enters bundle configuration mode.

## Applying Bundle-Level Parameters

Bundle-level parameters can be applied either by assigning VC classes or by directly applying them to the bundle.

Parameters applied through a VC class assigned to the bundle are superseded by those applied at the bundle level. Bundle-level parameters are superseded by parameters applied to an individual VC.

## Configuring Bundle-Level Parameters

Configuring bundle-level parameters is optional if a class is attached to the bundle to configure it.

To configure parameters that apply to the bundle and all of its members, use the following commands in bundle configuration mode:

Command	Purpose
<code>protocol protocol {protocol-address   inarp} [[no] broadcast]</code>	Configures a static map or enables Inverse Address Resolution Protocol (Inverse ARP) or Inverse ARP broadcasts for the bundle.
<code>encapsulation aal-encap</code>	Configures the ATM adaptation layer (AAL) and encapsulation type for the bundle.
<code>inarp minutes</code>	Configures the Inverse ARP time period for all VC bundle members.
<code>broadcast</code>	Enables broadcast forwarding for all VC bundle members.
<code>oam retry up-count down-count retry frequency</code>	Configures the VC bundle parameters related to OAM management.
<code>oam-bundle [manage] [frequency]</code>	Enables end-to-end F5 OAM loopback cell generation and OAM management for all VCs in the bundle.

## Configuring VC Class Parameters to Apply to a Bundle

Use of a VC class allows you to configure a bundle applying multiple attributes to it at once because you apply the class itself to the bundle. Use of a class allows you to generalize a parameter across all VCs, after which (for some parameters) you can modify that parameter for individual VCs. (See the section “Applying Parameters to Individual VCs” for more information.)

To configure a VC class to contain commands that configure all VC members of a bundle when the class is applied to that bundle, use the following command in vc-class configuration mode. To enter vc-class configuration mode, use the **vc-class atm** command.

Command	Purpose
<code>oam-bundle [manage] [frequency]</code>	Enables end-to-end F5 OAM loopback cell generation and OAM management for all VCs in the bundle.

In addition to this command, you can add the following commands to a VC class to be used to configure a bundle: **broadcast**, **encapsulation**, **inarp**, **oam retry**, and **protocol** commands. For information on these commands, including configuration tasks and command syntax, refer to the *Cisco IOS Wide-Area Networking Configuration Guide* and the *Cisco IOS Wide-Area Networking Command Reference*.

## Attaching a Class to a Bundle

To attach a preconfigured VC class containing bundle-level configuration commands to a bundle, use the following command in bundle configuration mode:

Command	Purpose
<code>class-bundle</code> <i>vc-class-name</i>	Configures a bundle with the bundle-level commands contained in the specified VC class.

Parameters set through bundle-level commands contained in the VC class are applied to the bundle and all of its VC members. Bundle-level parameters applied through commands configured directly on the bundle supersede those applied through a VC class.

Note that some bundle-level parameters applied through a VC class or directly to the bundle can be superseded by commands that you directly apply to individual VCs in bundle-vc configuration mode.

## Committing a VC to a Bundle

To add a VC to an existing bundle and enter bundle-vc configuration mode, use the following command in bundle configuration mode:

Command	Purpose
<code>pvc-bundle</code> <i>pvc-name</i> [ <i>vpi/</i> ] [ <i>vci</i> ]	Adds the specified VC to the bundle and enters bundle-vc configuration mode in order to configure the specified VC bundle member.

For information on how to first create the bundle and configure it, see the sections “Creating a VC Bundle” and “Applying Bundle-Level Parameters” earlier in this chapter.

## Applying Parameters to Individual VCs

Parameters can be applied to individual VCs either by using VC classes or by directly applying them to the bundle members.

Parameters applied to an individual VC supersede bundle-level parameters. Parameters applied directly to a VC take precedence over the same parameters applied within a class to the VC at the bundle-vc configuration level.

## Configuring a VC Bundle Member Directly

Configuring VC bundle members directly is optional if a VC class is attached to the bundle member.

To configure an individual VC bundle member directly, use the following commands in `bundle-vc` configuration mode:

Command	Purpose
<code>ubr output-pcr [input-pcr]</code>	Configures the VC for unspecified bit rate (UBR) QoS and specifies the output peak cell (PCR) rate for it.
<code>ubr+ output-pcr output-mcr [input-pcr] [input-mcr]</code>	Configures the VC for UBR QoS and specifies the output PCR and output minimum guaranteed cell rate for it.
<code>vbr-nrt output-pcr output-scr output-mbs [input-pcr] [input-scr] [input-mbs]</code>	Configures the VC for variable bit rate nonreal-time (VBR-nrt) QoS and specifies the output PCR, output sustainable cell rate, and output maximum burst cell size for it.
<code>precedence [other   range]</code>	Configures the precedence levels for the VC.
<code>bump {implicit   explicit precedence-level   traffic}</code>	Configures the bumping rules for the VC.
<code>protect {group   vc}</code>	Configures the VC to belong to the protected group of the bundle or to be an individually protected VC bundle member.

Parameters set directly for a VC at the `bundle-vc` configuration level take precedence over values for these parameters set for the VC at any other level, including application of a VC class at the `bundle-vc` configuration level.

## Configuring VC Class Parameters to Apply to a VC Bundle Member

To configure a VC class to contain commands that configure a specific VC member of a bundle when the class is applied to it, use the following commands in `vc-class` configuration mode. To enter `vc-class` configuration mode, use the `vc-class atm` command in global configuration mode.

Command	Purpose
<code>bump {implicit   explicit precedence-level   traffic}</code>	Specifies the bumping rules for the VC member to which the class is applied. These rules determine to which VC in the bundle traffic is directed when the carrier VC bundle member goes down.
<code>precedence precedence min-threshold max-threshold mark-probability-denominator</code>	Defines precedence levels for the VC member to which the class is applied.
<code>protect {group   vc}</code>	Configures the VC as a member of the protected group of the bundle or as an individually protected VC.

You can also add the following commands to a VC class to be used to configure a VC bundle member: **ubr**, **ubr+**, and **vbr-nrt**.

Use of a VC class allows you to configure a VC bundle member with multiple attributes at once because you can apply the class to the VC.

**Note**

When a VC is a member of a VC bundle, the following commands cannot be used in `vc-class` mode to configure the VC: **encapsulation**, **protocol**, **inarp**, and **broadcast**. These commands are useful only at the bundle level, not the bundle member level.

To configure the way bumping is handled for individual VCs within a bundle, use the **bump** command in the `bundle-vc` configuration mode. For more information about the bumping rules, see the section “Bumping and ATM VC Bundles” in the “IP to ATM Class of Service Overview” chapter in this book.

Configuration for an individual VC overrides the collective configuration applied to all VC bundle members through application of a VC class to the bundle.

## Applying a VC Class to a Discrete VC Bundle Member

To attach a preconfigured VC class containing bundle-level configuration commands to a bundle member, use the following command in `bundle-vc` configuration mode:

Command	Purpose
<code>class-vc vc-class-name</code>	Assigns a VC class to a VC bundle member.

Parameters that configure a VC that are contained in a VC class assigned to that VC are superseded by parameters that are directly configured for the VC through discrete commands entered in `bundle-vc` configuration mode.

## Configuring a VC Not to Accept Bumped Traffic

To configure an individual VC bundle member not to accept traffic that otherwise might be directed to it if the original VC carrying the traffic goes down, use the following command in `bundle-vc` configuration mode:

Command	Purpose
<code>no bump traffic</code>	Configures the VC not to accept any bumped traffic that would otherwise be redirected to it.

## Monitoring and Maintaining VC Bundles and Their VC Members

To gather information on bundles so as to monitor them or to troubleshoot problems that pertain to their configuration or use, use one or more of the following commands in privileged EXEC mode or debug mode:

Command	Purpose
<code>show atm bundle <i>bundle-name</i></code>	Displays the bundle attributes assigned to each bundle VC member and the current working status of the VC members.
<code>show atm bundle <i>bundle-name</i> statistics [<i>detail</i>]</code>	Displays statistics or detailed statistics on the specified bundle.
<code>show atm map</code>	Displays a list of all configured ATM static maps to remote hosts on an ATM network and on ATM bundle maps.
<code>debug atm bundle errors</code>	Prints information on bundle errors.
<code>debug atm bundle events</code>	Prints a record of bundle events.

## Per-VC WFQ and CBWFQ Configuration Task List

To configure IP to ATM CoS for per-VC WFQ and CBWFQ, perform the tasks in the following sections. The first two sections are required; the remaining sections are optional.

- Configuring Class-Based Weighted Fair Queueing (Required)
- Attaching a Service Policy and Enabling CBWFQ for a VC (Required)
- Configuring a VC to Use Flow-Based WFQ (Optional)
- Monitoring Per-VC WFQ and CBWFQ (Optional)
- Enabling Logging of Error Messages to the Console (Optional)

The IP to ATM CoS feature requires ATM PVC management.

See the end of this chapter for the sections “Per-VC WFQ and CBWFQ on a Standalone VC Example” and “Per-VC WFQ and CBWFQ on Bundle-Member VCs Example.”

## Configuring Class-Based Weighted Fair Queueing

Before configuring CBWFQ for a VC, you must perform the following tasks using standard CBWFQ commands:

- Create one or more classes to be used to classify traffic sent across the VC
- Define a policy map containing the classes to be used as the service policy

**Note**

You can configure class policies for as many classes as are defined on the router, up to the maximum of 64. However, the total amount of bandwidth allocated for all classes included in a policy map to be attached to a VC must not exceed 75 percent of the available bandwidth of the VC. The remaining 25 percent of available bandwidth is used for encapsulation, such as the ATM cell overhead (also referred to as ATM cell tax), routing and best-effort traffic, and other functions that assume overhead. For more information on bandwidth allocation, see the section “CBWFQ Bandwidth Allocation” in the “Congestion Management Overview” chapter in this book.

For information on how to configure CBWFQ and perform the tasks mentioned, see the chapter “Configuring Weighted Fair Queuing” in this book.

## Attaching a Service Policy and Enabling CBWFQ for a VC

Because CBWFQ gives you minimum bandwidth guarantee, you can only apply CBWFQ to VCs having these classes of service: available bit rate (ABR) and variable bit rate (VBR). You cannot apply per-VC WFQ and CBWFQ to UBR and UBR+ VCs because both of these service classes are best-effort classes that do not guarantee minimum bandwidth. When CBWFQ is enabled for a VC, all classes configured as part of the service policy are installed in the fair queueing system.

To attach a policy map to a standalone VC to be used as its service policy and to enable CBWFQ on that VC, use the following command in VC submode:

Command	Purpose
Router(config-if-atm-vc)# <b>service-policy output</b> <i>policy-map</i>	Enables CBWFQ and attaches the specified service policy map to the VC being created or modified.

To attach a policy map to an individual VC bundle member to be used as its service policy and to enable CBWFQ on that VC, use the following command in bundle-vc configuration mode:

Command	Purpose
Router(config-if-atm-member)# <b>service-policy output</b> <i>policy-map</i>	Enables CBWFQ and attaches the specified service policy map to the VC being created or modified.

**Note**

The **service-policy output** and **random-detect-group** commands are mutually exclusive; you cannot apply a WRED group to a VC for which you have enabled CBWFQ through application of a service policy. Moreover, before you can configure one command, you must disable the other if it is configured.

## Configuring a VC to Use Flow-Based WFQ

In addition to configuring CBWFQ at the VC level, the IP to ATM CoS feature allows you to configure flow-based WFQ at the VC level. Because flow-based WFQ gives you best-effort class of service—that is, it does not guarantee minimum bandwidth—you can configure per-VC WFQ for all types of CoS VCs: ABR, VBR, UBR, and UBR+.

Per-VC WFQ uses the class-default class. Therefore, to configure per-VC WFQ, you must first create a policy map and configure the class-default class. (You need not create the class-default class, which is predefined, but you must configure it.) For per-VC WFQ, the class-default class must be configured with the **fair-queue** policy-map class configuration command.

In addition to configuring the **fair-queue** policy-map class configuration command, you can configure the default class with either the **queue-limit** command or the **random-detect** command, but not both. Moreover, if you want the default class to use flow-based WFQ, you cannot configure the default class with the **bandwidth** policy-map class configuration command—to do so would disqualify the default class as flow-based WFQ, and therefore limit application of the service policy containing the class to ABR and VBR VCs.

To create a policy map and configure the class-default class, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	Router(config)# <b>policy-map</b> <i>policy-map</i>	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-pmap)# <b>class class-default</b> <i>default-class-name</i>	Specifies the default class so that you can configure or modify its policy.
Step 3	Router(config-pmap-c)# <b>fair-queue</b> <i>number-of-dynamic-queues</i>	Specifies the number of dynamic queues to be reserved for use by flow-based WFQ running on the default class.
Step 4	Router(config-pmap-c)# <b>queue-limit</b> <i>number-of-packets</i>  or Router(config-pmap-c)# <b>random-detect</b>	Specifies the maximum number of packets that can be enqueued for the class.  Enables WRED. The class policy will drop packets using WRED instead of tail drop.

For more information about creating a policy map and configuring the class-default class, see the section “Class-Based Weighted Fair Queuing Configuration Task List” in the “Configuring Weighted Fair Queuing” chapter in this book.

By default—that is, even if you do not configure the class-default class with the **fair-queue** policy-map class configuration command and you do not configure it with the **bandwidth** policy-map class configuration command—the default class is defined as flow-based WFQ.

Note that you can include other classes in the same policy map as the one that contains the flow-based WFQ class. Packets not otherwise matched are selected by the default class-default class match criteria.

To attach the policy map containing the class-default class to a standalone VC so that it becomes the service policy enabling WFQ for that VC, use the following command in VC submode:

Command	Purpose
Router(config-if-atm-vc)# <b>service-policy output</b> <i>policy-map</i>	Enables WFQ for the VC by attaching the specified policy map containing the class-default class to the VC being created or modified.

To attach the policy map containing the class-default class to an individual VC bundle member so that the policy map becomes the service policy enabling WFQ for that VC, use the following command in bundle-vc configuration mode:

Command	Purpose
Router(config-if-atm-member)# <b>service-policy output</b> <i>policy-map</i>	Enables WFQ for the VC bundle member by attaching the specified policy map containing the class-default class to the VC bundle member.

## Monitoring Per-VC WFQ and CBWFQ

To monitor per-VC WFQ and CBWFQ in your network, use one or more of the following commands in EXEC mode:

Command	Purpose
<b>show queue</b> <i>interface-name interface-number [vc [vpi/] vci]</i>	Displays the contents of packets inside a queue for a particular interface or VC.
<b>show queueing interface</b> <i>interface-number [vc [[vpi/] vci]</i>	Displays the queueing statistics of a specific VC on an interface.

## Enabling Logging of Error Messages to the Console

When you configure a VC in order to create or modify it, the router performs the task in interrupt mode. For this reason, the router cannot issue printf statements to inform you of error conditions, if errors occur. Rather, the router logs all error messages to the console. To accommodate these circumstances, you should enable logging of error messages to the console.

To enable logging of error messages to the console, use the following command in global configuration mode:

Command	Purpose
<b>logging console</b> <i>level</i>	Limits messages logged to the console based on severity.

For information on the **logging console** command, including configuration tasks and command syntax, refer to the *Cisco IOS Configuration Fundamentals Configuration Guide* and the *Cisco IOS Configuration Fundamentals Command Reference*.

# IP to ATM CoS Configuration Examples

The following sections provide IP to ATM CoS configuration examples:

- Single ATM VC with WRED Group and IP Precedence Example
- VC Bundle Configuration Using a VC Class Example
- Per-VC WFQ and CBWFQ on a Standalone VC Example
- Per-VC WFQ and CBWFQ on Bundle-Member VCs Example

For information on how to configure IP to ATM CoS, see the sections “IP to ATM CoS on a Single ATM VC Configuration Task List” and “IP to ATM CoS on an ATM Bundle Configuration Task List” in this chapter.

## Single ATM VC with WRED Group and IP Precedence Example

The following example creates a PVC on an ATM interface and applies the WRED parameter group sanjose to that PVC. Next, the IP Precedence values are configured for the WRED parameter group sanjose.

```
interface ATM1/1/0.46 multipoint
 ip address 200.126.186.2 255.255.255.0
 no ip mroute-cache
 shutdown
 pvc cisco 46
 encapsulation aal5nlpid
 random-detect attach sanjose
 !
 random-detect-group sanjose
 precedence 0 200 1000 10
 precedence 1 300 1000 10
 precedence 2 400 1000 10
 precedence 3 500 1000 10
 precedence 4 600 1000 10
 precedence 5 700 1000 10
 precedence 6 800 1000 10
 precedence 7 900 1000 10
```

## VC Bundle Configuration Using a VC Class Example

This example configures VC bundle management on a router that uses Intermediate System-to-Intermediate System (IS-IS) as its IP routing protocol.

### Bundle-Class Class

At the outset, this configuration defines a VC class called bundle-class that includes commands that set VC parameters. When the class bundle-class is applied at the bundle level, these parameters are applied to all VCs that belong to the bundle. Note that any commands applied directly to an individual VC of a bundle in bundle-vc mode take precedence over commands applied globally at the bundle level. Taking

into account hierarchy precedence rules, VCs belonging to any bundle to which the class bundle-class is applied will be characterized by these parameters: aal5snap encapsulation, broadcast on, use of Inverse ARP to resolve IP addresses, and OAM enabled.

```
router isis
 net 49.0000.0000.0000.1111.00

vc-class atm bundle-class
 encapsulation aal5snap
 broadcast
 protocol ip inarp
 oam-bundle manage 3
 oam retry 4 3 10
```

The following sections of the configuration define VC classes that contain commands specifying parameters that can be applied to individual VCs in a bundle by assigning the class to that VC.

### Control-Class Class

When the class control-class is applied to a VC, the VC carries traffic whose IP Precedence level is 7. When the VC to which this class is assigned goes down, it takes the bundle down with it because this class makes the VC a protected one. The QoS type of a VC using this class is vbr-nrt.

```
vc-class atm control-class
 precedence 7
 protect vc
 vbr-nrt 10000 5000 32
```

### Premium-Class Class

When the class premium-class is applied to a VC, the VC carries traffic whose IP Precedence level is 6 and 5. The VC does not allow other traffic to be bumped onto it. When the VC to which this class is applied goes down, its bumped traffic will be redirected to a VC whose IP Precedence level is 7. This class makes a VC a member of the protected group of the bundle. When all members of a protected group go down, the bundle goes down. The QoS type of a VC using this class is vbr-nrt.

```
vc-class atm premium-class
 precedence 6-5
 no bump traffic
 protect group
 bump explicitly 7
 vbr-nrt 20000 10000 32
```

### Priority-Class Class

When the class priority-class is applied to a VC, the VC is configured to carry traffic with IP Precedence in the 4-2 range. The VC uses the implicit bumping rule, it allows traffic to be bumped, and it belongs to the protected group of the bundle. The QoS type of a VC using this class isubr+.

```
vc-class atm priority-class
 precedence 4-2
 protect group
 ubr+ 10000 3000
```

### Basic-Class Class

When the class `basic-class` is applied to a VC, the VC is configured through the **precedence other** command to carry traffic with IP Precedence levels not specified in the profile. The VC using this class belongs to the protected group of the bundle. The QoS type of a VC using this class is `ubr`.

```
vc-class atm basic-class
  precedence other
  protect group
  ubr 10000
```

The following sets of commands configure three bundles that the router subinterface uses to connect to three of its neighbors. These bundles are called `new-york`, `san-francisco`, and `los-angeles`. Bundle `new-york` has four VC members, bundle `san-francisco` has four VC members, and bundle `los-angeles` has three VC members.

### New-York Bundle

The first part of this example specifies the IP address of the subinterface, the router protocol—the router uses IS-IS as an IP routing protocol—and it creates the first bundle called `new-york` and enters bundle configuration mode.

```
interface atm 1/0.1 multipoint
  ip address 10.0.0.1 255.255.255.0
  ip router isis
  bundle new-york
```

From within bundle configuration mode, the next portion of the configuration uses two protocol commands to enable IP and Open Systems Interconnect (OSI) traffic flows in the bundle. The OSI routing packets will use the highest precedence VC in the bundle. The OSI data packets, if any, will use the lowest precedence VC in the bundle. If configured, other protocols, such as IPX or AppleTalk, will always use the lowest precedence VC in the bundle.

As the indentation levels of the preceding and following commands suggest, `subordinate to bundle new-york` is a command that configures its protocol and a command that applies the class `bundle-class` to it.

```
  protocol ip 1.1.1.2 broadcast
  protocol clns 49.0000.0000.2222.00 broadcast
  class-bundle bundle-class
```

The class called `bundle-class`, which is applied to the bundle `new-york`, includes a **protocol ip inarp** command. According to inheritance rules, **protocol ip**, configured at the bundle level, takes precedence over **protocol ip inarp** specified in the class `bundle-class`.

The next set of commands beginning with `pvc-bundle ny-control 207`, which are further subordinate, add four VCs (named `ny-control`, `ny-premium`, `ny-priority`, and `ny-basic`) to the bundle `new-york`. A particular class—that is, one of the classes predefined in this configuration example—is applied to each VC to configure it with parameters specified by commands included in the class.

As is the case for this configuration, to configure individual VCs belonging to a bundle, the router must be in bundle mode for the mother bundle. For each VC belonging to the bundle, the subordinate mode is `pvc-mode` for the specific VC.

The following commands configure the individual VCs for the bundle new-york:

```
pvc-bundle ny-control 207
  class-vc control-class
pvc-bundle ny-premium 206
  class-vc premium-class
pvc-bundle ny-priority 204
  class-vc priority-class
pvc-bundle ny-basic 201
  class-vc basic-class
```

### San-Francisco Bundle

The following set of commands create and configure a bundle called san-francisco. At the bundle configuration level, the configuration commands included in the class bundle-class are ascribed to the bundle san-francisco and to the individual VCs that belong to the bundle. Then, the **pvc-bundle** command is executed for each individual VC to add it to the bundle. After a VC is added and bundle-vc configuration mode is entered, a particular, preconfigured class is assigned to the VC. The configuration commands comprising that class are used to configure the VC. Rules of hierarchy apply at this point. Command parameters contained in the applied class are superseded by the same parameters applied at the bundle configuration level, which are superseded by the same parameters applied directly to a VC.

```
bundle san-francisco
  protocol clns 49.0000.0000.0000.333.00 broadcast
  inarp 1
  class-bundle bundle-class
pvc-bundle sf-control 307
  class-vc control-class
pvc-bundle sf-premium 306
  class-vc premium-class
pvc-bundle sf-priority 304
  class-vc priority-class
pvc-bundle sf-basic 301
  class-vc basic-class
```

### Los-Angeles Bundle

The following set of commands create and configure a bundle called los-angeles. At the bundle configuration level, the configuration commands included in the class bundle-class are ascribed to the bundle los-angeles and to the individual VCs that belong to the bundle. Then, the **pvc-bundle** command is executed for each individual VC to add it to the bundle. After a VC is added and bundle-vc configuration mode is entered, precedence is set for the VC and the VC is either configured as a member of a protected group (protect group) or as an individually protected VC. A particular class is then assigned to each VC to further characterize it. Rules of hierarchy apply. Parameters of commands applied

directly and discretely to a VC take precedence over the same parameters applied within a class to the VC at the bundle-vc configuration level, which take precedence over the same parameters applied to the entire bundle at the bundle configuration level.

```
bundle los-angeles
  protocol ip 1.1.1.4 broadcast
  protocol clns 49.0000.0000.4444.00 broadcast
  inarp 1
  class-bundle bundle-class
  pvc-bundle la-high 407
    precedence 7-5
    protect vc
    class-vc premium-class
  pvc-bundle la-mid 404
    precedence 4-2
    protect group
    class-vc priority-class
  pvc-bundle la-low 401
    precedence other
    protect group
    class-vc basic-class
```

## Per-VC WFQ and CBWFQ on a Standalone VC Example

The following example creates two class maps and defines their match criteria. For the first map class, called class1, the numbered access control list (ACL) 101 is used as the match criterion. For the second map class, called class2, the numbered ACL 102 is used as the match criterion.

Next, the example includes these classes in a policy map called policy1. For class1, the policy includes a minimum bandwidth allocation request of 500 Mbps and maximum packet count limit of 30 for the queue reserved for the class. For class2, the policy specifies only the minimum bandwidth allocation request of 1000 Mbps, so the default queue limit of 64 packets is assumed. Note that the sum of the bandwidth requests for the two classes comprising policy1 is 75 percent of the total amount of bandwidth (2000 Mbps) for PVC cisco to which the policy map is attached.

The example attaches the policy map called policy1 to the PVC called cisco. Once the policy map policy1 is attached to PVC cisco, its classes constitute the CBWFQ service policy for that PVC. Packets sent on this PVC will be checked for matching criteria against ACLs 101 and 102 and classified accordingly.

Because **class-default** is not explicitly configured for this policy map, all traffic that does not meet the match criteria of the two classes comprising the service policy is handled by the predefined class-default class, which provides best-effort flow-based WFQ.

```
class-map class1
  match access-group 101

class-map class2
  match access-group 102

policy-map policy1
  class class1
    bandwidth 500
    queue-limit 30

  class class2
    bandwidth 1000

interface ATM1/1/0.46 multipoint
  ip address 200.126.186.2 255.255.255.0
  pvc cisco 46
    vbr-nrt 2000 2000
    encaps aal5snap
    service policy output policy1
```

## Per-VC WFQ and CBWFQ on Bundle-Member VCs Example

The following example shows a PVC bundle called san-francisco with members for which per-VC WFQ and CBWFQ are enabled and service policies configured. The example assumes that the classes included in the following policy maps have been defined and that the policy maps have been created: policy1, policy2, and policy4. For each PVC, the IP to ATM CoS **pvc-bundle** command is used to specify the PVC to which the specified policy map is to be attached.

Note that PVC 0/34 and 0/31 have the same policy map attached to them, policy2. Although you can assign the same policy map to multiple VCs, each VC can have only one policy map attached at an output PVC.

```
bundle san-francisco
  protocol ip 1.0.2.20 broadcast
  encapsulation aal5snap
  pvc-bundle 0/35
    service policy output policy1
    vbr-nrt 5000 3000 500
    precedence 4-7
  pvc-bundle 0/34
    service policy output policy2
    vbr-nrt 5000 3000 500
    precedence 2-3
  pvc-bundle 0/33
    vbr-nrt 4000 3000 500
    precedence 2-3
    service policy output policy4
  pvc-bundle 0/31
    service policy output policy2
```

