Complex Hierarchical Scheduling: Fragmented Policies (i.e, Policies Aggregation)

The QoS: Policies Aggregation feature supports Modular QoS CLI (MQC) configuration of default traffic classes in policy maps on different subinterfaces to be queued as a single, user-defined traffic class at the main-interface policy map. It is most useful in quality of service (QoS) configurations where you have several subinterface policy maps on the same physical interface and you want identical treatment of the default traffic classes on those subinterfaces.

Beginning in Cisco IOS XE Release 2.6, the QoS: Policies Aggregation feature is enhanced to support queueing aggregation at the primary interface for other traffic classes, including Differentiated Services Code Point (DSCP) traffic classes such as the expedited forwarding (EF), Assured Forwarding 1 (AF1), and AF4 traffic classes. With this enhancement, any traffic classes from VLAN subinterfaces can share a common queue for that traffic class at the main-interface policy map. Other enhancements include the ability to configure and show drop statistics that occur at the aggregate level for these classes.

Prerequisites for QoS: Policies Aggregation

- This feature is configured using the MQC.
- All traffic over the main interface should come through one or more subinterfaces.

Restrictions for QoS: Policies Aggregation

- Applies only when multiple subinterfaces with policy maps are attached to the same physical interface. This feature cannot be used to collectively classify default traffic classes or other traffic classes of policy maps on different physical interfaces.
- Certain traffic class configuration prior to Cisco IOS XE Release 2.6 at the subinterface policy map and main-interface policy map will have different behavior and queueing results. See the "Understanding the QoS Policies Aggregation MQC" section on page 3 and the "Differences Between the Original Feature and the MQC Support for Multiple Queue Aggregation" section on page 4.
- The **service-fragment** keyword is only supported on the Gigabit Ethernet interfaces and not on Fast Ethernet interfaces.

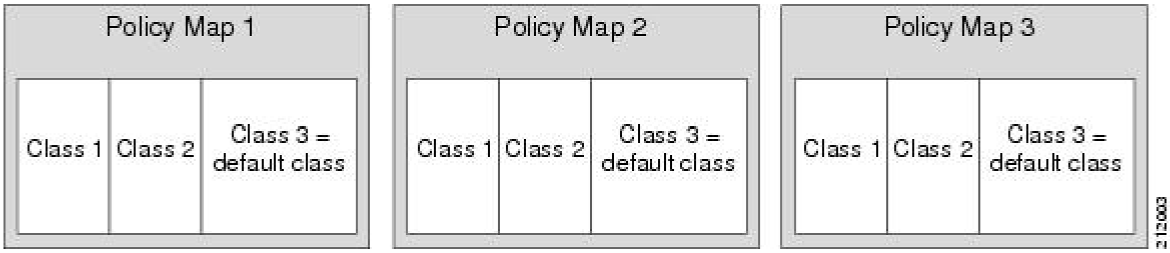
About QoS: Policies Aggregation

Fragments in Class Definition Statements

QoS: Policies Aggregation introduces the idea of fragments in class definition statements. A default traffic class definition statement can be marked as a fragment within a policy map. Other policy maps on the same interface can also define their default traffic class statements as fragments, if desired. A separate policy map can then be created with a service-fragment class definition statement that will be used to apply QoS to all of the fragments as a single group.

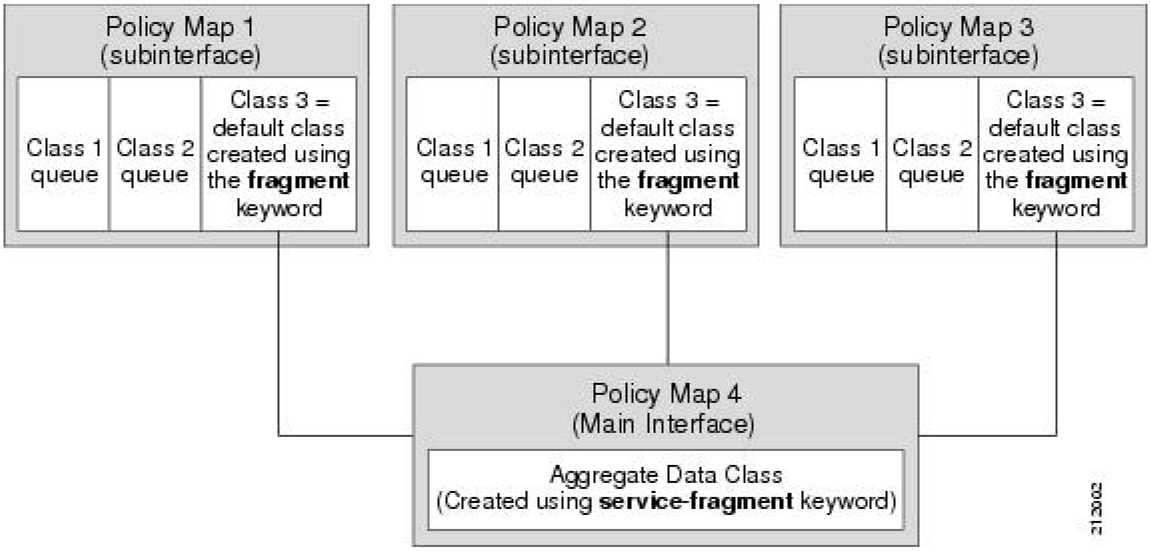
The figure below provides an example of one physical interface with three attached policy maps that is not using fragments. Note that each policy map has a default traffic class that can only classify traffic for the default traffic within its own policy map.

Figure 1: Three Policy Maps Configured Without Fragments



The figure below shows the same configuration configured with fragments and adds a fourth policy map with a class definition statement that classifies the fragments collectively. The default traffic classes are now classified as one service-fragment group rather than three separate default traffic classes within the individual policy maps.

Figure 2: Three Policy Maps Configured Using Fragments


Policy Map 1
(subinterface)
Class 1 queue
Class 2 queue
Class 3 = default class created using the **fragment** keyword
Policy Map 2
(subinterface)
Class 1 queue
Class 2 queue
Class 3 = default class created using the **fragment** keyword
Policy Map 3
(subinterface)
Class 1 queue
Class 2 queue
Class 3 = default class created using the **fragment** keyword
Policy Map 4
(Main Interface)
Aggregate Data Class
(Created using **service-fragment** keyword)
212002

Fragments for Gigabit Etherchannel Bundles

When fragments are configured for Gigabit Etherchannel bundles, the policy-maps that have a default traffic class configured using the **fragment** keyword are attached to the member subinterface links, and the policy-maps that have a traffic class configured with the **service-fragment** keyword to collectively classify the fragments is attached to the physical interface.

All port-channel subinterfaces configured with fragments that are currently active on a given port-channel member link will use the aggregate service fragment class on that member link. If a member link goes down, the port-channel subinterfaces that must switch to the secondary member link will then use the aggregate service fragment on the new interface.

Fragment Traffic Class in a Policy Map

Only the default class statement in a policy map can be configured as a fragment.

Fragments work only when multiple policy maps are attached to the same physical interface. This process cannot be used to classify default traffic classes as fragments on policy maps on different physical interfaces.

Only queuing features are allowed in classes where the **fragment** keyword is entered, and at least one queuing feature must be entered in classes where the **fragment** keyword is used.

A policy map with a class using the **fragment** keyword can only be applied to traffic leaving the interface (policy maps attached to interfaces using the **service-policy output** command).

The **fragment** keyword cannot be entered in a child policy map.

Understanding Service Fragment Traffic Classes

A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

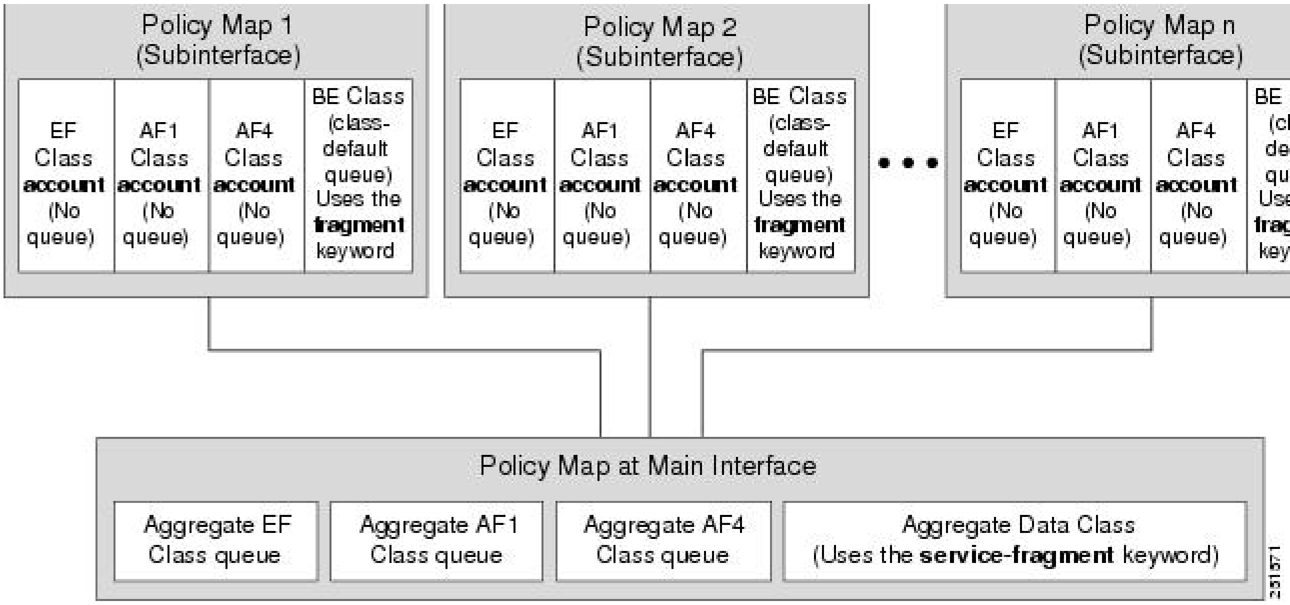
The **service-fragment** keyword cannot be entered in a child policy map.

QoS: Policies Aggregation MQC

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature extends the previous support of aggregation of class-default traffic using the **fragment** and **service-fragment** configurations, to other user-defined traffic classes in a subinterface policy-map, such as DSCP-based traffic classes, that are aggregated at the main-interface policy-map as shown in the figure below.

When no queueing is configured on a traffic class in the subinterface policy-map, the **account** command can be used to track queueing drops that occur at the aggregate level for these classes, and can be displayed using the **show policy-map interface** command.

Figure 3: Policy-Map Overview for the MQC Support for Multiple Queue Aggregation at Main Interface Feature


Policy Map 1 (Subinterface)
EF Class account (No queue)
AF1 Class account (No queue)
AF4 Class account (No queue)
BE Class (class-default queue) Uses the fragment keyword
Policy Map 2 (Subinterface)
EF Class account (No queue)
AF1 Class account (No queue)
AF4 Class account (No queue)
BE Class (class-default queue) Uses the fragment keyword
Policy Map n (Subinterface)
EF Class account (No queue)
AF1 Class account (No queue)
AF4 Class account (No queue)
BE (cl de qu Use frag key
Policy Map at Main Interface
Aggregate EF Class queue
Aggregate AF1 Class queue
Aggregate AF4 Class queue
Aggregate Data Class (Uses the service-fragment keyword)
251571

Differences Between the Original Feature and the MQC Support for Multiple Queue Aggregation

Although some of the configuration between the original QoS policies aggregation feature and enhancements in the MQC Support for Multiple Queue Aggregation at Main Interface feature appears similar, there are some important differences in the queueing behavior and the internal data handling.

For example, both configurations share and require the use of the **fragment** keyword for the **class class-default** command in the subscriber policy map, as well as configuration of the **service-fragment** keyword for a user-defined class in the main-interface policy map to achieve common policy treatment for aggregate traffic. However, the use of this configuration results in different behavior between the original and enhanced QoS policies aggregation implementation:

- In the original implementation (prior to Cisco IOS XE Release 2.6) using the fragment and service-fragment architecture, all default class traffic and any traffic for classes without defined queueing features at the subinterface goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy map. Subinterface traffic aggregation (for example, from multiple subscribers on the same physical interface) ultimately occurs only for a single class, which is the default class.

  Here are the feature characteristics:

  - All subinterface traffic classes have queues. However, when a traffic class in the subinterface policy-map is not configured with any queueing feature (commands such as **priority**, **shape**, **bandwidth**, **queue-limit**, **fair-queue**, **random-detect**, and so on, are not configured), the traffic is assigned to the class-default queue.
  - Default class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class.
  - No classification occurs or is supported at the main-interface policy map for any subinterface traffic classes that do not use the **fragment** and **service-fragment** configuration.
  - Queueing occurs at the subinterface for other traffic classes defined with queueing features in the subinterface policy map.

- In the enhanced implementation (beginning with Cisco IOS XE Release 2.6) of the MQC Support for Multiple Queue Aggregation at Main Interface feature also using the fragment and service-fragment architecture, all default class traffic also goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy map. However, other classes, such as DSCP-based subscriber traffic classes, are also supported for an aggregate policy. These traffic classes do not support any queues or queueing features other than **account** at the subscriber policy map. The use of the fragment and service-fragment architecture enables these other subscriber traffic classes (from multiple subscribers on the same physical interface) to achieve common policy treatment for aggregate traffic that is defined for those same classes at the main policy map.

  Here are the feature characteristics:

  - Subinterface traffic classes without configured queueing features do not have queues at the subscriber level.
  - Default class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class. This configuration

additionally enables support for other subinterface traffic classes (such as DSCP-based classes) to be aggregated into a common policy-map at the main interface.

- Other class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface, according to the following configuration requirements:
- You enable this behavior by using the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class (this also enables aggregation of the default class).
- You do not configure any queueing features at the subinterface policy-map for the other traffic classes.
- Queueing occurs at the main-interface policy map for other subinterface traffic classes as an aggregate.
- Optional tracking of statistics is supported using the **account** command for other traffic classes in the subinterface policy map.

Changes in Queue Limit and WRED Thresholds

In Cisco IOS XE Release 2.6 the Cisco ASR 1000 Series Routers support the addition of bytes as a unit of configuration for both queue limits and WRED thresholds. Therefore, as of this release, packet-based and byte-based limits are configurable, with some restrictions.

Configuration Examples for QoS: Policies Aggregation

Examples 1: Configuring QoS: Policies Aggregation for an Interface

Configuring a Fragment Traffic Class in a Policy-Map

Before You Begin

This procedure shows only how to configure the default traffic class as a fragment within a policy-map. It does not include steps on configuring other classes within the policy-map, or other policy-maps on the device.

Example

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a fragment named BestEffort is created in policy-map subscriber1 and policy-map subscriber 2. In this example, queuing features for other traffic classes are supported at the subinterface policy-map.

```
policy-map subscriber1
 class voice
```

```
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example also shows how to configure a fragment named BestEffort for the default class in a policy-map on a subinterface using the QoS Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface implementation. In this example, notice that queuing features are not supported for the other classes in the policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

After configuring default class statements as fragments in multiple subinterface policy-maps, a separate policy-map with a class statement using the service-fragment keyword must be configured to apply QoS to the class stratements configured as fragments.

What to Do Next

After configuring default class statements as fragments in multiple subinterface policy maps, a separate policy map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This task is documented in the "Configuring a Service Fragment Traffic Class" section on page 8.

Configuring a Service Fragment Traffic Class

Before You Begin

This task describes how to configure a service fragment traffic class statement within a policy-map. A service fragment traffic class is used to apply QoS to a collection of default class statements that have been configured previously in other policy-maps as fragments.

This procedure assumes that fragment default traffic classes were already created. The procedure for creating fragment default traffic classes is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section.

Like any policy-map, the configuration does not manage network traffic until it has been attached to an interface. This procedure does not cover the process of attaching a policy-map to an interface.

Note

A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy-map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy-maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

The **service-fragment** keyword cannot be entered in a child policy-map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name* **service-fragment** *fragment-class-name*
5. **shape average percent** *percent*
6. **end**

DETAILED STEPS

	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.

	Command or Action	Purpose
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map BestEffortFragments`	Specifies the name of the traffic policy to configure and enters policy-map configuration mode.

Step 4	**class** *class-name* **service-fragment** *fragment-class-name*
	Example:
	`Device(config-pmap)# class data service-fragment BestEffort`
		Specifies a class of traffic that is the composite of all fragments matching the *fragment-class-name*. The *fragment-class-name* when defining the fragments in other policy-maps must match the *fragment-class-name* in this command line to properly configure the service fragment class.

Step 5	**shape average percent** *percent* **Example:** `Device(config-pmap-c)# shape average percent 50`	Enters a QoS configuration command. Only queueing features are supported in default traffic classes configured as fragments. The queueing features that are supported are **bandwidth**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.

Step 6	**end**
	Example:
	`Device(config-pmap-c)# end`
		Exits policy-map class configuration mode and returns to privileged EXEC mode.

Examples

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a policy-map is created to apply QoS to all fragments named BestEffort.

```
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
```

In the following example, two fragments are created and then classified collectively using a service fragment.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

```
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example shows the creation of two fragments called BestEffort in the subinterface policy-maps, followed by a sample configuration for the **service-fragment** called BestEffort to aggregate the queues at the main interface policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber2
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

Troubleshooting Tips

Ensure that all class statements that are supposed to be part of the same service fragment share the same *fragment-class-name*.

What to Do Next

The policy map (traffic policy) must be attached to an interface. This task is documented in the "Attaching a Traffic Policy to an Interface Using the MQC" section in chapter "Applying QoS Features Using the MQC."

Configuring QoS: Policies Aggregation on Gigabit Etherchannels

To properly configure QoS: Policies Aggregation on a Gigabit Etherchannel bundle, the following actions must be completed:

- Service-fragment traffic classes must be configured and attached to the main physical interfaces.
- Fragment traffic classes must be configured and attached to the member link subinterfaces.

Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle

Before You Begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the “Configuring a Fragment Traffic Class in a Policy-Map” section. The procedure for creating a service fragment traffic classes is documented in the “Configuring a Service Fragment Traffic Class” section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions document only the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Note For proper behavior, when a port-channel member link goes down, all member links should have the same policy-map applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *service-fragment-class-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card*/*bay*/*port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service-policy configuration.
Step 4	**service-policy output** *service-fragment-class-name* **Example:** `Device(config-if)# service-policy output aggregate-member-link`	Attaches a service policy that contains a service fragment default traffic class to the physical Gigabit Ethernet interface.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the policy-map aggregate-member-link is attached to the physical interface.

```
interface GigabitEthernet1/1/1
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 service-policy output aggregate-member-link
```

What to Do Next

Ensure that the fragment class name is consistent across service-fragment and fragment class definitions. Continue to the “Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces” section.

Troubleshooting Tips

Ensure that the *fragment-class-name* is consistent across service-fragment and fragment-class definitions.

What to Do Next

Attach the fragment service policy on the Gigabit Etherchannel member link subinterfaces. This task is documented in the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section on page 14.

Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces

Before You Begin

This task assumes that a service-fragment traffic class has already been created. A service-fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy Map" section on page 6. The procedure for creating a service-fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section on page 8.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions only document the procedure for attaching a policy map that already has a fragment traffic class to a member link subinterface.

Note Fragments cannot be used for traffic on two or more physical interfaces. The GEC must all be on the same physical interface for this configuration to work properly.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-interface-number.port-channel-subinterface-number*
4. **service-policy output** *fragment-class-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-interface-number.port-channel-subinterface-number* **Example:** `Router(config)# interface port-channel 1.100`	Enters subinterface configuration mode to configure a Etherchannel member link subinterface.

	Command or Action	Purpose
Step 4	**service-policy output** *fragment-class-name* **Example:** `Router(config-subif)# service-policy output subscriber`	Attaches a service policy that contains a fragment default traffic class to the Etherchannel member link subinterface.

Example

Note This example shows a sample configuration that is supported for the original QoS: Policies Aggregation feature in releases prior to Cisco IOS XE Release 2.6. By following the newer policy-map configuration guidelines for the updates in Cisco IOS XE Release 2.6, it can be adapted to the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.

In the following example, the service policy named subscriber has a fragment default traffic class and is attached to the member link subinterface of a Gigabit Etherchannel bundle.

Note This example only shows how to attach a fragment default traffic class to the member link subinterface of a Gigabit Etherchannel bundle. This configuration is incomplete and would not classify default traffic appropriately until the physical interface was configured to support a service-fragment traffic class.

```
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
  bandwidth remaining ratios 80
policy-map aggregate-member-link
 class BestEffort service-fragment BE
  shape average 100000000
!
interface Port-channel1
 ip address 172.16.2.3 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 192.168.2.100 255.255.255.0
 service-policy output subscriber
!
```

Troubleshooting Tips

This configuration will not work until a service-fragment default traffic class is created to classify the default traffic classes marked as fragments. This service-fragment traffic class must be configured for this configuration to have any affect on network traffic.

How to Configure QoS: Policies Aggregation MQC

Some backward-compatibility exists between support of policies aggregation feature configuration in Cisco IOS XE Release 2.6 and prior Cisco IOS XE software releases. However, we recommend that you follow these upgrade guidelines for any physical interface where you want to move to the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature configuration.

For best results, you should upgrade any service policies configuration that you implemented prior to Cisco IOS XE Release 2.6, to the latest supported configuration.

The original and enhanced QoS: Policies Aggregation feature configuration can only reside on the same Cisco ASR 1000 Series Router if the mixed configuration does not reside on the same physical interface. In other words, you can support the original configuration for one physical interface, and the enhanced configuration on a different physical interface.

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature requires the same configuration of a fragment traffic class as the original feature, using the **class class-default fragment** command to enable and then define all subinterface policies aggregation, both for the default traffic class and the other traffic classes.

In the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature, the queueing features for the aggregate class queues (with traffic from the corresponding classes identified at the subinterfaces), are configured at the main-interface policy map.

Upgrading Your Service Policies for QoS: Policies Aggregation MQC

Before You Begin

Upgrading your service policies to support the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature assumes the following network conditions:

- The corresponding class-map statements appropriate for your network traffic are already configured.
- QoS service policies aggregation has been previously configured and applied for the main-interface policy map for a given physical interface and its corresponding subinterfaces, or subscriber interfaces, prior to Cisco IOS XE Release 2.6 for the default traffic class.
- A port on the same physical interface where you have previously configured the service policies aggregation feature prior to Cisco IOS XE Release 2.6 needs to support the configuration for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface.

Upgrade Tasks

SUMMARY STEPS

1. Configure the service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.
2. Remove any service policies configured prior to Cisco IOS XE Release 2.6 for any prior configured policies aggregation features using the **no service-policy** and **no policy-map** commands as follows:
3. Apply the new service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature at the appropriate interfaces using the **service-policy output** command as follows:

DETAILED STEPS

Step 1 Configure the service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.
See the tasks described in the "Configuring QoS Policies Aggregation MQC Traffic Classes" section on page 18.

Step 2 Remove any service policies configured prior to Cisco IOS XE Release 2.6 for any prior configured policies aggregation features using the **no service-policy** and **no policy-map** commands as follows:

a) At each of the subinterfaces, configure the **no service-policy** command. Be sure to remove the policies at the subinterfaces first.
b) At the physical interface, configure the **no service-policy**command.

Step 3 Apply the new service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature at the appropriate interfaces using the **service-policy output** command as follows:

a) At the physical interface, configure the **service-policy output** command.
b) At each of the subinterfaces, configure the **service-policy output** command.

Configuring QoS: Policies Aggregation MQC Traffic Classes

Configuring Traffic Classes on the Subscriber Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. **account** [**drop**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map subscriber1`	Specifies the name of the traffic policy to configure and enters policy map configuration mode.

Step 4	**class** *class-name*
	Example:
	`Router(config-pmap)# class EF`
		Specifies the name of the traffic class to be aggregated at the main-interface policy map, and enters policy-map class configuration mode.
		Note Do not configure any queueing features for this class. Queueing is configured and aggregated at the main-interface policy map for all subinterfaces associated with this class and physical interface.

Step 5	**account** [**drop**] **Example:** `Router(config-pmap-c)# account`	(Optional) Enables collection of statistics for packets matching the traffic class where this command is configured, where the **drop** keyword collects all packet drop statistics. Collection of drop statistics is the default.

Example

The following example configures the EF traffic class for policies aggregation at the subscriber subinterface with collection of drop statistics:

```
policy-map subscriber1
 class EF
 account
```

What to Do Next

Perform this task for all traffic classes that you want to aggregate, then perform the task in the "Configuring the Fragment Traffic Class on a Subinterface" section on page 19.

Configuring the Fragment Traffic Class on a Subinterface

What to Do Next

If you are upgrading your subinterface policy-map configuration from an earlier implementation of the QoS: Policies Aggregation feature, then remove the current service-policy from the subinterface using the **no service-policy** command.

Apply the new policy map to outbound traffic on the subinterface using the **service-policy output** command.

Configuring Traffic Classes at the Main Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. **priority level** *level*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map main-interface`	Specifies the name of the traffic policy to configure and enters policy map configuration mode.

Step 4	**class** *class-name*
	Example:
	`Router(config-pmap)# class EF`
		Specifies the name of the traffic class to be aggregated at the main-interface policy map, and enters policy-map class configuration mode.

Step 5	**priority level** *level*	Enters a QoS configuration command.

	Command or Action	Purpose
	Example: `Router(config-pmap-c)# priority level 1`	The queueing features that are currently supported are **bandwidth**, **priority**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.

Example

The following example configures three traffic classes at the main-interface policy map, along with the aggregate service-fragment data class:

```
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

What to Do Next

Perform this task to define queueing features for all traffic classes that you want to aggregate, then perform the task in the "Configuring the Service Fragment Traffic Class at the Main Interface" section on page 21.

Configuring the Service Fragment Traffic Class at the Main Interface

What to Do Next

After configuring multiple default class statements as fragments in a policy-map, a separate policy-map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This process is documented in the “Configuring a Service Fragment Traffic Class” section.

Configuring QoS: Policies Aggregation MQC Support

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature also supports configuration of the enhanced service policies on Gigabit Etherchannels according to the subscriber and main-interface configuration guidelines described for this enhancement.

For more information, see the following sections:

Verifying the Traffic Policy Class Policy Information and Drop Statistics

To display information about policy-map configuration and subscriber drop statistics enabled using the account command, use the **show policy-map interface** command:

```
Router# show policy-map interface port-channel 1.1
Port-channel1.1
   Service-policy input: input_policy
     Class-map: class-default (match-any)
       0 packets, 0 bytes
       5 minute offered rate 0000 bps, drop rate 0000 bps
       Match: any
       QoS Set
       dscp default
       No packet marking statistics available
   Service-policy output: Port-channel_1_subscriber
     Class-map: EF (match-any)
       105233 packets, 6734912 bytes
       5 minute offered rate 134000 bps, drop rate 0000 bps
       Match: dscp ef (46)
       Match: access-group name VLAN_REMARK_EF
       Match: qos-group 3
       Account QoS statistics
         Queueing
           Packets dropped 0 packets/0 bytes
       QoS Set
       cos 5
       No packet marking statistics available
       dscp ef
       No packet marking statistics available
     Class-map: AF4 (match-all)
       105234 packets, 6734976 bytes
       5 minute offered rate 134000 bps, drop rate 0000 bps
       Match: dscp cs4 (32)
       Account QoS statistics
         Queueing
           Packets dropped 0 packets/0 bytes
       QoS Set
       cos 4
       No packet marking statistics available
     Class-map: AF1 (match-any)
       315690 packets, 20204160 bytes
       5 minute offered rate 402000 bps, drop rate 0000 bps
       Match: dscp cs1 (8)
       Match: dscp af11 (10)
       Match: dscp af12 (12)
       Account QoS statistics
         Queueing
           Packets dropped 0 packets/0 bytes
       QoS Set
       cos 1
       No packet marking statistics available
     Class-map: class-default (match-any) fragment Port-channel_BE
       315677 packets, 20203328 bytes
       5 minute offered rate 402000 bps, drop rate 0000 bps
       Match: any
       Queueing
         queue limit 31250 bytes
         (queue depth/total drops/no-buffer drops) 0/0/0
         (pkts output/bytes output) 315679/20203482
         bandwidth remaining ratio 1
```

Configuration Examples for QoS: Policies Aggregation

Example: QoS: Policies Aggregation

Note This example shows a sample configuration that is supported in the original QoS: Policies Aggregation feature prior to Cisco IOS XE Release 2.6.

In the following example, QoS: Policies Aggregation is used to define a fragment class of traffic to classify default traffic using the default traffic class named BestEffort. All default traffic from the policy maps named subscriber1 and subscriber2 is part of the fragment default traffic class named BestEffort. This default traffic is then shaped collectively by creating a class called data that uses the **service-fragment** keyword and the **shape** command.

Note the following about this example:

- The *class-name* for each fragment default traffic class is "BestEffort."
- The *class-name* of "BestEffort" is also used to define the class where the **service-fragment** keyword is entered. This class applies a shaping policy to all traffic forwarded using the fragment default traffic classes named "BestEffort."

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map input_policy
 class class-default
  set dscp default
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
interface portchannel1.1001
 encapsulation dot1q 1001
 service-policy output subscriber1
 service-policy input input_policy
interface portchannel1.1002
 encapsulation dot1q 1002
 service-policy output subscriber2
 service-policy input input_policy
interface gigabitethernet 0/1
 description member-link1
 port channel 1
 service-policy output main-interface
interface gigabitethernet 0/2
 description member-link2
 port channel 1
service-policy output main-interface
```

Example: Gigabit Etherchannel QoS Policies Aggregation

Note This example shows a sample configuration that is supported in the original QoS: Policies Aggregation feature prior to Cisco IOS XE Release 2.6.

In the following example, policy map subscriber is configured with a fragment class named BE. The fragment is then configured as part of a policy map named aggregate-member-link. Policy map subscriber is then attached to the bundle subinterfaces while policy map aggregate-member-link is attached to the physical interface.

```
port-channel load-balancing vlan-manual
class-map match-all BestEffort
!
class-map match-all video
!
class-map match-all voice
!
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
  bandwidth remaining ratios 80
policy-map aggregate-member-link
 class BestEffort service-fragment BE
  shape average 100000000
!
interface Port-channel1
 ip address 10.1.1.3 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 10.1.2.1 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 10.1.2.2 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 10.1.2.3 255.255.255.0
 service-policy output subscriber
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
```

Example: QoS: Policies Aggregation MQC Support at Main Interface

Note This example shows a sample configuration that is supported beginning in Cisco IOS XE Release 2.6.

At the main-interface policy map called Port-channel_1_main_policy, the queueing features for the DSCP-based subscriber traffic classes are configured. You can also see the use of byte-based queue limits and random-detect thresholds implemented at the main-interface queues.

The service fragment called Port-channel_BE is also configured to aggregate the traffic from the subscriber class-default fragment class.

```
policy-map Port-channel_1_main_policy
 class EF
  priority level 1
  queue-limit 547500 bytes
 class AF4
  priority level 2
  queue-limit 4037500 bytes
 class AF1
  bandwidth remaining ratio 90
  queue-limit 750000 bytes
  random-detect dscp-based
  random-detect dscp 8 750000 bytes 750000 bytes
  random-detect dscp 10 750000 bytes 750000 bytes
  random-detect dscp 12 600000 bytes 675000 bytes
 class data service-fragment Port-channel_BE
  shape average 250000000
  bandwidth remaining ratio 1
!
```

In this example, the policy map Port-channel_1_subscriber is configured with a fragment class named Port-channel_BE. (For simplicity, only a single subinterface policy is shown.) This enable queueing and policies aggregation for the subscriber traffic classes at the main-interface policy map.

The Port-channel_1_subscriber policy map identifies the DSCP-based traffic classes of EF, AF4, and AF1 and enables collection of drop statistics for those classes.

```
policy-map Port-channel_1_subscriber
 class EF
  account
  set cos 5
  set dscp ef
 class AF4
  account
  set cos 4
 class AF1
  account
  set cos 1
 class class-default fragment Port-channel_BE
  bandwidth remaining ratio 1
  queue-limit 31250 bytes
!
port-channel load-balancing vlan-manual
!
interface Port-channel1
 no ip address
 no negotiation auto
!
```

The service policies are applied first to the physical interface, and then to the subinterfaces as shown:

```
interface GigabitEthernet1/2/0
 no ip address
 negotiation auto
 no cdp enable
 service-policy output Port-channel_1_main_policy
 channel-group 1
!
interface GigabitEthernet2/2/0
 no ip address
 negotiation auto
 service-policy output Port-channel_1_main_policy
 channel-group 1
!
interface Port-channel1.1
 encapsulation dot1Q 2 primary GigabitEthernet1/2/0 secondary GigabitEthernet2/2/0
 ip address 10.0.0.2 255.255.255.0
 service-policy output Port-channel_1_subscriber
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Distribution of Remaining Bandwidth Using Ratio	"Distribution of Remaining Bandwidth Using Ratio" module
Class-Based Shaping	"Regulating Packet Flow--Using Class-Based Traffic Shaping" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified by this feature.	

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	

Technical Assistance

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

Feature Information for QoS: Policies Aggregation

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

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

Table 1: Feature Information for QoS: Policies Aggregation

Feature Name	Releases	Feature Information
QoS: Policies Aggregation	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers. The following command was modified: **class (policy-map)**.

Feature Name	Releases	Feature Information
QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface	Cisco IOS XE Release 2.6	This feature was enhanced to support queueing aggregation at the primary interface for other traffic classes, including DSCP-based classes such as EF, AF1, and AF4 traffic classes. With this enhancement, other traffic classes from different subinterfaces share a common queue for that traffic class. Other enhancements include the ability to configure and show per-subscriber drop statistics on the aggregate queues and byte-based queue limits and WRED thresholds. In Cisco IOS XE Release 2.6, support for the CISCO-CLASS-BASED-QOS-MIB was added. The following commands are new or modified: **account**, **show policy-map interface**.