Configuring Class-Based RTP and TCP Header Compression

Last Updated: December 5, 2011

Header compression is a mechanism that compresses the IP header in a packet before the packet is transmitted. Header compression reduces network overhead and speeds up the transmission of either Real-Time Transport Protocol (RTP) packets or Transmission Control Protocol (TCP) packets.

Cisco provides two types of header compression: RTP header compression and TCP header compression.

RTP and TCP header compression are typically configured on a per-interface (or subinterface) basis. Class-based RTP and TCP header compression allows you to configure either type of header compression on a per-class basis. This module describes the concepts and tasks related to configuring class-based RTP and TCP header compression.

Note If you want to configure RTP or TCP header compression on a per-interface (or subinterface) basis, see the "Configuring RTP Header Compression" module or the "Configuring TCP Header Compression" module, respectively.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see the release notes for your platform and software release. To find information

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

about the features documented in this module, and to see a list of the releases in which each feature is supported, see the Feature Information Table at the end of this document.

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.

Prerequisites for Class-Based RTP and TCP Header Compression

Before configuring class-based RTP and TCP header compression, read the information in the "Header Compression" module.

Restrictions for Class-Based RTP and TCP Header Compression

Class-based RTP and TCP header compression can be enabled on PPP interfaces, High-Level Data Link Control (HDLC) interfaces, and interfaces that use Frame Relay encapsulation. However, note the following points about the header-compression formats supported on these interfaces:

- For PPP and HDLC interfaces, the only supported format for header compression is the IPHC (IP Header Compression) format.
- For interfaces that use Frame Relay encapsulation, the IPHC format is not available. The only supported format for header compression is the Cisco proprietary format.

Information About Class-Based RTP and TCP Header Compression

Class-Based Header Compression and the MQC

Class-based RTP and TCP header compression allows you to configure *either* RTP *or* TCP header compression for a specific class within a policy map (sometimes referred to as a traffic policy). You configure the class and the policy map by using the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC). The MQC is a CLI that allows you to create classes within policy maps (traffic policies) and then attach the policy maps to interfaces (or subinterfaces). The policy maps are used to configure and apply specific QoS features (such as RTP or TCP header compression) to your network. For more information about the MQC, see the "Applying QoS Features Using the MQC" module.

Benefits of Class-Based Header Compression

Class-based header compression allows you to compress (and then decompress) a subset of the packets on your network. Class-based header compression acts as a filter; it allows you to specify at a much finer level the packets that you want to compress. For example, instead of compressing all RTP (or TCP) packets that traverse your network, you can configure RTP header compression to compress only those packets that meet certain criteria (for example, protocol type "ip" in a class called "voice)."

Header Compression on Local and Remote Routers

In a typical network topology, header compression is configured at both a local router and a remote router. If you configure class-based RTP header compression (or class-based TCP header compression) on the local router, you must also configure RTP header compression (or TCP header compression) on the remote router.

However, when you configure either RTP or TCP header compression on the remote router, you can choose one of the following:

- You can configure *class-based* RTP or TCP header compression on the remote router (by using the instructions in this module)

or

- You can configure RTP or TCP header compression *directly on the interface* of the remote router (by using the instructions in the "Configuring RTP Header Compression" module or the "Configuring TCP Header Compression" module, respectively).

Note If you configure RTP or TCP header compression directly on the interface of the remote router, you must specify the **iphc-format** keyword for PPP and HDLC interfaces. For Frame Relay interfaces, the **iphc-format** keyword is not supported; only the Cisco proprietary format (that is, the **cisco** keyword) is supported. For more information about the **iphc-format**keyword, see either the "Configuring RTP Header Compression" module or the "Configuring TCP Header Compression" module.

About Header-Compression Connections

Number of Connections Calculated on the Basis of Bandwidth

In class-based RTP and TCP header compression, the number of header-compression connections is calculated on the basis of the amount of available bandwidth.

Note the following points about how bandwidth is used:

- The setting of the **bandwidth** command determines the amount of bandwidth available on the interface.
- The number of header-compression connections is calculated by dividing the available bandwidth by 4 (that is, 4 kilobits per connection).

Header-Compression Connections on HDLC and Frame Relay Interfaces

For HDLC interfaces and Frame Relay interfaces (that is, interfaces that use Frame Relay encapsulation), the number of header-compression connections on *both sides* of the network must match. That is, the number calculated (from the bandwidth setting) for use on the local router must match the number configured (or calculated from the bandwidth setting) for use on the remote router.

Header-Compression Connections on PPP Interfaces

For PPP interfaces, if the header-compression connection numbers on both sides of the network do not match, the number used is "autonegotiated." That is, any mismatch in the number of header-compression connections between the local router and the remote router will be automatically negotiated to the lower of the two numbers. For example, if the local router is configured to use 128 header-compression connections, and the remote router is configured to use 64 header-compression connections, the negotiated number will be 64.

Note This autonegotiation function applies to PPP interfaces *only* . For HDLC interfaces and interfaces that use Frame Relay encapsulation, no autonegotiation occurs.

How to Configure Class-Based RTP and TCP Header Compression

Enabling RTP or TCP Header Compression for a Class in a Policy Map

With class-based header compression, you can configure either RTP or TCP header compression for a specific class inside a policy map. To specify the class, to create a policy map, and to configure either RTP or TCP header compression for the class inside the policy map, perform the following steps.

Note In the following task, the **match protocol**command is shown in step Enabling RTP or TCP Header Compression for a Class in a Policy Map, page 4 The **match protocol**command matches traffic on the basis on the protocol type and is only an example of a **match** command you can use. You may want to use a different **match** command to specify another criterion. The **match** commands vary by Cisco IOS release. See the command documentation for the Cisco IOS release that you are using for a complete list of **match** commands.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**match-all**| **match-any**] *class-map-name*
4. **match protocol** *protocol-name*
5. **exit**
6. **policy-map** *policy-map-name*
7. **class** {*class-name*| **class-default**}
8. **compression header ip** {**rtp** | **tcp**}
9. **end**

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	**class-map** [**match-all**| **match-any**] *class-map-name* **Example:** `Router(config)# class-map class1`	Creates a class map to be used for matching packets to a specified class and enters class-map configuration mode. • Enter the class map name.
Step 4	**match protocol** *protocol-name* **Example:** `Router(config-cmap)# match protocol ip`	(Optional) Matches traffic on the basis of the specified protocol. • Enter the protocol name. **Note** The **match protocol** command matches traffic on the basis of the protocol type. The **match protocol**command is just an example of one of the **match** commands that can be used. The **match** commands vary by Cisco IOS release. See the command documentation for the Cisco IOS release that you are using for a complete list of **match** commands.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Router(config-cmap)# exit`	(Optional) Exits class-map configuration mode.
Step 6	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy and enters policy-map configuration mode. • Enter the policy map name.
Step 7	**class** {*class-name*| **class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the name of the class whose policy you want to create or change and enters policy-map class configuration mode. • Enter the class name or the **class-default** keyword.
Step 8	**compression header ip** {**rtp** | **tcp**} **Example:** `Router(config-pmap-c)# compression header ip rtp`	Configures either RTP or TCP header compression for a specific class. • Enter either the **rtp** keyword (for RTP header compression) or the **tcp** keyword (for TCP header compression).
Step 9	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Exits policy-map class configuration mode.

Attaching the Policy Map to an Interface

After a policy map is created, the next step is to attach the policy map to an interface (or subinterface). To attach the policy map to an interface or subinterface, perform the following steps.

Note You configure class-based RTP and TCP header compression in policy maps. Then you attach those policy maps to an interface by using the **service-policy** command. The **service-policy**command gives you the option of specifying either an input service policy (for input interfaces) or an output service policy (for output interfaces). For class-based RTP and TCP header compression, you can specify output service policies *only* .

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **service-policy output** *policy-map-name*
5. **end**

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** *type number* [*name-tag*] **Example:** `Router(config)# interface serial0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and the interface number.
Step 4	**service-policy output** *policy-map-name* **Example:** `Router(config-if)# service-policy output policy1`	Specifies the name of the policy map to be attached to the interface in the output direction. • Enter the policy map name. **Note** Policy maps can be attached in the input or output direction of an interface. For class-based RTP and TCP header compression, always use the **output** keyword.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Verifying the Class-Based RTP and TCP Header Compression Configuration

This task allows you to verify that you created the intended configuration and that the feature is functioning correctly. To verify the configuration, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **show policy-map interface** *type number* **output**
3.
4. **show policy-map** *policy-map* **class** *class-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map interface** *type number* **output** **Example:** `Router# show policy-map interface serial0 output`	Displays the packet statistics of all classes that are configured for all service policies on the specified interface. • Enter the interface type and the interface number.
Step 3		
Step 4	**show policy-map** *policy-map* **class** *class-name* **Example:** `Router# show policy-map policy1 class class1`	Displays the configuration for the specified class of the specified policy map. • Enter the policy map name and the class name.
Step 5	**end** **Example:** `Router# end`	(Optional) Exits privileged EXEC mode.

Configuration Examples for Class-Based RTP and TCP Header Compression

- Example Enabling RTP or TCP Header Compression for a Class in a Policy Map, page 9
- Example Attaching the Policy Map to an Interface, page 9
- Example Verifying the Class-Based RTP and TCP Header Compression Configuration, page 9

Example Enabling RTP or TCP Header Compression for a Class in a Policy Map

In the following example, a class map called class1 and a policy map called policy1 have been configured. Policy1 contains the class called class1, within which RTP header compression has been enabled by using the **compression header ip rtp** command.

```
Router> enable
Router# configure terminal
Router(config)# class-map class1
Router(config-cmap)# match protocol ip
Router(config-cmap)# exit
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# compression header ip rtp
Router(config-pmap-c)# end
```

Example Attaching the Policy Map to an Interface

In the following example, the policy map called policy1 has been attached to serial interface 0.

```
Router> enable
Router# configure terminal
Router(config)# interface serial0
Router(config-if)# service-policy output policy1
Router(config-if)# end
```

Example Verifying the Class-Based RTP and TCP Header Compression Configuration

This section provides sample output from a typical **showpolicy-mapinterface**command.

Note Depending upon the interface in use and the QoS feature enabled (such as Class-Based Weighted Fair Queuing [CBWFQ]), the output you see may vary from that shown below.

The following sample displays the statistics for serial interface 0. In this sample configuration, three classes, called gold, silver, and voice, have been configured. Traffic is classified and grouped into classes on the basis of the IP precedence value and RTP port protocol number.

```
class-map match-all gold
 match ip precedence 2
class-map match-all silver
 match ip precedence 1
class-map match-all voice
 match ip precedence 5
 match ip rtp 16384 1000
```

This sample configuration also contains a policy map called mypolicy, configured as shown below. QoS features such as RTP header compression and CBWFQ are enabled for specific classes within the policy map.

```
policy-map mypolicy
  class voice
    priority 128              ! A priority queue and bandwidth amount are specified.
    compress header ip rtp    ! RTP header compression is enabled for class voice.
  class gold
    bandwidth 100             ! CBWFQ is enabled for class gold.
  class silver
    bandwidth 80              ! CBWFQ is enabled for class silver.
    random-detect             ! WRED is enabled for class silver.
```

Given the classes and policy map configured as shown above, the following content is displayed for serial interface 0:

```
Router# show policy-map interface
 serial0 output
 Serial0
  Service-policy output: mypolicy

    Class-map: voice (match-all)
      880 packets, 58080 bytes
      30 second offered rate 1000 bps, drop rate 0 bps
      Match: ip precedence 5
      Match: ip rtp 16384 1000
      Queueing
        Strict Priority
        Output Queue: Conversation 136
        Bandwidth 128 (kbps) Burst 3200 (Bytes)
        (pkts matched/bytes matched) 880/26510
        (total drops/bytes drops) 0/0
      compress:
          header ip rtp
          UDP/RTP (compression on, IPHC, RTP)
            Sent:    880 total, 877 compressed,
                     31570 bytes saved, 24750 bytes sent
                     2.27 efficiency improvement factor
                     99% hit ratio, five minute miss rate 0 misses/sec, 0 max
                     rate 0 bps

    Class-map: gold (match-all)
      100 packets, 53000 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: ip precedence 2
      Queueing
        Output Queue: Conversation 137
        Bandwidth 100 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 100/53000
        (depth/total drops/no-buffer drops) 0/0/0
```

```
    Class-map: silver (match-all)
      878 packets, 1255540 bytes
      30 second offered rate 56000 bps, drop rate 0 bps
      Match: ip precedence 1
      Queueing
        Output Queue: Conversation 138
        Bandwidth 64 (kbps)
        (pkts matched/bytes matched) 878/1255540
        (depth/total drops/no-buffer drops) 0/0/0
         exponential weight: 9
         mean queue depth: 0

  class    Transmitted      Random drop      Tail drop    Minimum Maximum  Mark
           pkts/bytes       pkts/bytes       pkts/bytes    thresh  thresh prob
      0       0/0              0/0              0/0           20      40  1/10
      1      878/1255540       0/0              0/0           22      40  1/10
      2       0/0              0/0              0/0           24      40  1/10
      3       0/0              0/0              0/0           26      40  1/10
      4       0/0              0/0              0/0           28      40  1/10
      5       0/0              0/0              0/0           30      40  1/10
      6       0/0              0/0              0/0           32      40  1/10
      7       0/0              0/0              0/0           34      40  1/10
   rsvp       0/0              0/0              0/0           36      40  1/10

    Class-map: class-default (match-any)
      3 packets, 84 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
```

Additional References

The following sections provide references related to configuring class-based RTP and TCP header compression.

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*
MQC	"Applying QoS Features Using the MQC" module
Header compression overview	"Header Compression" module
RTP header compression	"Configuring RTP Header Compression" module
TCP header compression	"Configuring TCP Header Compression" module
IPHC profiles and header compression	"Configuring Header Compression Using IPHC Profiles" module

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1144	*Compressing TCP/IP Headers for Low-Speed Serial Links*
RFC 2507	*IP Header Compression*
RFC 2508	*Compressing IP/UDP/RTP Headers for Low-Speed Serial Links*
RFC 3544	*IP Header Compression over PPP*
RFC 3545	*Enhanced Compressed RTP (CRTP) for Links with High Delay, Packet Loss and Reordering*
RFC 3550	*A Transport Protocol for Real-Time Applications*

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

Glossary

bandwidth --The rated throughput capacity of a given network medium.

compression --The running of a data set through an algorithm that reduces the space required to store the data set or the bandwidth required to transmit the data set.

full header (header refresh) --An uncompressed header that updates or refreshes the context for a packet stream. It carries a context identifier (CID) that will be used to identify the context. Full headers for non-TCP packet streams also carry the generation of the context that they update or refresh.

HDLC --High-Level Data Link Control. A bit-oriented synchronous data link layer protocol developed by the International Organization for Standardization (ISO). Derived from Synchronous Data Link Control (SDLC), HDLC specifies a data encapsulation method on synchronous serial links using frame characters and checksums.

header --A chain of subheaders.

MQC --Modular Quality of Service Command-Line Interface. The MQC is a CLI that allows you to create traffic classes and policy maps and then attach the policy maps to interfaces. The policy maps apply QoS features to your network.

PPP --Point-to-Point Protocol. A protocol that provides router-to-router and host-to-network connections over synchronous and asynchronous circuits.

regular header --A normal, uncompressed header. A regular header does not carry a context identifier (CID) or generation association.

RTP --Real-Time Transport Protocol. A protocol that is designed to provide end-to-end network transport functions for applications that transmit real-time data, such as audio, video, or simulation data, over unicast or multicast network services. RTP provides such services as payload type identification, sequence numbering, timestamping, and delivery monitoring to real-time applications.

subheader --An IPv6 base header, an IPv6 extension header, an IPv4 header, a UDP header, an RTP header, or a TCP header.

TCP --Transmission Control Protocol. A connection-oriented transport layer protocol that provides reliable full-duplex data transmission. TCP is part of the TCP/IP protocol stack.

Feature Information for Class-Based RTP and TCP Header Compression

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 Class-Based RTP and TCP Header Compression

Feature Name	Releases	Feature Information
Class-Based RTP and TCP Header Compression	12.2(13)T	This feature allows you to configure Real-Time Transport Protocol (RTP) or Transmission Control Protocol (TCP) IP header compression on a per-class basis, when a class is configured within a policy map. Policy maps are created using the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC).