

Руководство пользователя

QSR-2830

Оглавление

1	CONFIGURING QOS	11
1.1	What is QOS	11
1.2	Why QoS	11
1.3	QoS under Differentiated Services	11
1.4	IPv6 QoS	12
1.5	Congestion Management	12
1.6	What Is Congestion Management	12
1.7	Policies of Congestion Management	13
1.7.1	FIFO	13
1.7.2	WFQ	13
1.7.3	CBWFQ	13
1.7.4	PQ	13
1.7.5	LLQ and RTPQ	13
1.7.6	CQ	14
1.8	Deciding Which Queuing Policy to Use	14
1.9	Work Mechanism of Congestion Management	15
1.9.1	FIFO	15
1.9.1.1	Working Principles	15
1.9.1.2	Applicable Environment	15
1.9.2	WFQ	15
1.9.2.1	Working Principles	15
1.9.2.2	Applicable Environment	16
1.9.3	CBWFQ	16
1.9.3.1	Working Principles	16
1.9.3.2	Applicable Environment	17
1.9.4	LLQ and RTPQ	17
1.9.4.1	Working Principles	17
1.9.4.2	Applicable Environment	17
1.9.5	CQ	17
1.9.5.1	Working Principles	17
1.9.5.2	How to Determine Byte Count Values for Queues	18
1.9.5.3	Applicable Environment	19
1.9.6	PQ	19
1.9.6.1	Working Principles	19

1.9.6.2	Applicable Environment	19
1.10	WFQ	20
1.10.1	WFQ Configuration Tasks	20
1.10.2	Configuring WFQ	20
1.10.3	Monitoring WFQ	21
1.10.4	WFQ Configuration Example	21
1.11	CBWFQ	21
1.11.1	CBWFQ Configuration Tasks	21
1.11.2	Configuring CBWFQ	21
1.11.2.1	Defining Class-Maps	21
1.11.2.2	Configuring Rules in Policy-Maps	22
1.11.2.3	Applying Service Rules on Specified Interfaces	23
1.11.2.4	Configuring Bandwidth for Specified Classes	23
1.11.2.5	Configuring Queue Depth for Specified Classes	24
1.11.2.6	Configuring the DSCP Code Value of the IP TOS Domain for a Specified Class	24
1.11.2.7	Configuring the DSCP Code Value of the IPv4 TOS Domain and the IPv6 Traffic Class Domain for a Specified Class	24
1.11.2.8	Configuring the Precedence Code Value of the IP TOS Domain for a Specified Class	25
1.11.2.9	Configuring the Precedence Code Value of the IPv4 TOS Domain and the IPv6 Traffic Class Domain for a Specified Class	25
1.11.2.10	Configuring the COS Value of Ethernet Packets for a Specified Class	25
1.11.2.11	Configuring Bandwidth Assigned to CBWFQ	26
1.11.3	Monitoring CBWFQ	26
1.11.4	CBWFQ Configuration Example	26
1.12	Configuring LLQ and RTPQ	28
1.12.1.1	Configuration Tasks of LLQ and RTPQ	28
1.12.2	Configuring LLQ and RTPQ	28
1.12.3	Monitoring LLQ and RTPQ	29
1.12.4	Configuration Examples of LLQ and RTPQ	30
1.13	Configuring CQ	31
1.13.1	CQ Configuration Tasks	31
1.13.1.1	Configuring CQ	31
1.13.1.2	Determining the Maximum Capacity of CQ	31
1.13.1.3	Assigning Packets to CQ Queues	31
1.13.1.4	Specifying the Lowest CQ Queue Number	32
1.13.1.5	Applying the CQ List on an Interface	32
1.13.2	Monitoring CQ	33
1.13.3	CQ Configuration Examples	33
1.14	Configuring PQ	34

1.14.1	PQ Configuration Tasks	34
1.14.2	Configuring PQ	34
1.14.2.1	Determining the Maximum Capacity of PQ	34
1.14.2.2	Assigning Packets to PQ Queues	34
1.14.2.3	Applying the PQ List on an Interface	35
1.14.2.4	Monitoring PQ	35
1.14.3	PQ Configuration Examples	35
1.14.4	Configuring Hold-queue	36
1.14.4.1	Configuring hold-queue	36
1.14.4.2	Monitoring hold-queue	36
1.14.4.3	hold-queue Configuration Example	36
1.15	Traffic Policing and Shaping	36
1.16	What Are Traffic Policing and Traffic Shaping	36
1.17	Overview of Traffic Policing and Shaping	36
1.18	Configuration Tasks of Traffic Policing	38
1.19	Configuration Tasks of Traffic Shaping	40
1.20	Configuration Tasks of Traffic Policing on a Policy-Map	41
1.21	Configuration Tasks of Traffic Shaping on a Policy-Map	42
1.21.1	Configuring Car Token Bucket Algorithm Mode	43
1.22	Configuration Examples of Traffic Policing	44
1.22.1	Configuration Examples of Traffic Policing for Entire Traffic of an Interface	44
1.22.2	Configuration Examples of Traffic Policing for Traffic Satisfying Certain Conditions	44
1.22.3	Configuration Tasks of Traffic Policing on a Policy-Map	45
1.23	Configuration Examples of Traffic Shaping	46
1.23.1	Configuration Examples of Traffic Shaping for Entire Traffic of an Interface	46
1.23.2	Configuration Examples of Traffic Shaping for Traffic Satisfying Certain Conditions	46
1.23.3	Configuration Tasks of Traffic Shaping on a Policy-Map	47
1.23.4	Configuration Examples of Car Token Bucket Algorithm Mode	47
1.24	Maintenance and Debugging of Traffic Policing and Shaping	47
1.25	Congestion Avoidance	49
1.26	Overview	49
1.26.1	Traditional Packet-Drop Policy – Tail-Drop	49
1.26.2	WRED	49
1.27	Configuration Tasks of Congestion Avoidance (WRED) Based on an Interface	50

1.28	Configuration Tasks of Congestion Avoidance (WRED) Based on a Policy-Map	51
1.29	Configuration Examples of Congestion Avoidance (WRED)	53
1.29.1	Configuration Examples of Congestion Avoidance for Entire Traffic of an Interface	53
1.29.2	Configuration Examples of Congestion Avoidance on a Policy-Map	53
1.29.3	Maintenance of Congestion Avoidance	54
1.30	Compression Protocol	56
1.30.1	Packet Compression Protocols	56
1.30.1.1	VJ TCP Packet Compression	56
1.30.1.2	IPHC	56
1.30.2	Efficiency and Inapplicable Scenarios of Packet Compression	57
1.30.3	Packet Compression Protocol Configuration	57
1.30.4	Packet Compression Configuration Examples	59
1.30.4.1	PPP-based Packet Compression	59
1.30.4.2	HDLC-based Packet Compression	59
1.30.4.3	Frame Relay-based Packet Compression	59
1.30.5	Packet Compression Maintenance and Debugging	59
2	CONFIGURING MPLS QOS	61
2.1	MPLS QOS	61
2.2	Congestion Management	61
2.3	Configuration of Weighted Fair Queuing (WFQ)	61
2.3.1	WFQ Configuration Tasks	61
2.3.2	Configuring WFQ	61
2.3.3	Monitoring WFQ	62
2.3.4	WFQ Configuration Examples	62
2.4	Configuration of Class-based Weighted Fair Queueing (CBWFQ)	63
2.4.1	CBWFQ Configuration Tasks	63
2.4.2	Configuring CBWFQ	63
2.4.2.1	Defining Class Maps	63
2.4.2.2	Configuring Class Policy in the Policy Map	63
2.4.2.3	Applying Service Policy on the Designated Interface	64
2.4.2.4	Configuring Bandwidth for an Existing Class	64
2.4.2.5	Configuring the Queue Depth for an Existing Class	64
2.4.2.6	Configuring EXP Value of MPLS Message for an Existing Class	65
2.4.2.7	Configure EXP Value of MPLS Message for an Existing Class (use table-map)	65
2.4.2.8	Configuring DSCP Value of IP Message for an Existing Class	66
2.4.2.9	Configuring DSCP Value of IP Message for an Existing Class (use table-map)	66
2.4.2.10	Configuring Precedence Value of IP Message for an Existing Class	67

2.4.2.11	Configuring Precedence Value of IP Message for an Existing Class (use table-map)	67
2.4.2.12	Configuring Group ID of Message for an Existing Class	68
2.4.2.13	Configuring Group ID of Message for an Existing Class (use table-map)	68
2.4.2.14	Configuring the Bandwidth Allocated to CBWFQ	69
2.4.3	Monitoring CBWFQ	69
2.4.4	CBWFQ Configuration Examples	69
2.5	Configuration of Custom Queueing (CQ)	70
2.5.1	CQ Configuration Tasks	70
2.5.2	Configuring CQ	70
2.5.2.1	Determining the Maximum Capacity of Queue Adopting CQ	71
2.5.2.2	Assigning Packets to CQ	71
2.5.2.3	Applying CQ List on the Interface	71
2.5.3	Monitoring CQ	72
2.5.4	CQ Configuration Examples	72
2.6	Configuration of Priority Queueing (PQ)	72
2.6.1	PQ Configuration Tasks	72
2.6.2	Configuring PQ	72
2.6.2.1	Determining the Maximum Capacity of Queue Adopting PQ	72
2.6.2.2	Assigning Packets to PQ	73
2.6.2.3	Applying PQ list on the Interface	73
2.6.3	Monitoring PQ	73
2.6.4	PQ Configuration Examples	74
2.7	Configuration of Low Latency Queueing (LLQ)	74
2.7.1	LLQ Configuration Tasks	74
2.7.2	Configuring LLQ	74
2.7.3	Monitoring LLQ	75
2.7.4	LLQ Configuration Examples	75
2.8	Traffic Policing and Traffic Shaping	76
2.9	Introduction to Traffic Policing and Traffic Shaping	76
2.10	Traffic Policing Configuration Tasks	76
2.11	Traffic Shaping Configuration Tasks	77
2.12	Configuring Traffic Policing under Policy-map	77
2.13	Traffic Policing Configuration Examples	79
2.13.1	Example of Applying Traffic Policing on All Traffics on the Interface	79
2.13.2	Example of Traffic Policing Configuration under Policy-map	79
2.14	Traffic Shaping Configuration Examples	81
2.14.1	Example of Applying Traffic Shaping on All Traffics on the Interface	81

2.15	Congestion Avoidance	83
2.16	Introduction to Congestion Avoidance	83
2.16.1	WRED	83
2.17	Configure Congestion Avoidance	84
2.17.1.1	Congestion Avoidance (WRED) Configuration Tasks	84
2.18	Congestion Avoidance (WRED) Configuration Examples	85
2.18.1	Example of Configuring Congestion Avoidance on the Interface	85
2.19	QoS Overview	86
2.20	Understanding QoS	86
2.20.1	QoS Overview	86
2.20.2	Basic Concepts	86
2.20.2.1	Best-Effort Service	86
2.20.2.2	Integrated Service	86
2.20.2.3	Differentiated Service	86
2.20.3	Working Principles	86
2.20.3.1	Traffic Classification	86
2.20.3.2	Traffic Policing	87
2.20.3.3	Traffic Shaping	87
2.20.3.4	Congestion Management	87
2.20.3.5	Congestion Avoidance	87
2.20.3.6	RSVP	87
2.21	Traffic Classification Configuration	87
2.22	Understanding Traffic Classification	87
2.22.1	Traffic Classification Overview	87
2.22.2	Basic Concepts	88
2.22.2.1	QoS Priority	88
2.22.2.2	QoS Coloring	89
2.22.2.3	Traffic Classifier	89
2.22.2.4	Traffic Behavior	89
2.22.2.5	Traffic Policy	89
2.22.2.6	DiffServ Domain	89
2.22.3	Working Principles	91
2.22.3.1	Simple Traffic Classification	91
2.22.3.2	Complex Traffic Classification	91
2.22.4	Protocols and Specifications	92
2.23	Default Configurations	92
2.24	Configuring Complex Traffic Classification	92

2.24.1	Configuring Traffic Classifiers	92
2.24.2	Configuring Traffic Behaviors	93
2.24.3	Configuring Traffic Policies	93
2.24.4	Applying Traffic Classification Polices	94
2.24.5	Displaying Configurations	94
2.25	Configuring Simple Traffic Classification	94
2.25.1	Configuring DiffServ Domains and Traffic Policies	94
2.25.2	Applying Traffic Classification Polices	95
2.25.3	Displaying Configurations	95
2.26	Examples for Configuring Traffic Classification	98
2.26.1	Configuration Example 1	98
2.26.1.1	Networking Requirements	98
2.26.1.2	Network Topology	98
2.26.1.3	Configuration Tips	99
2.26.1.4	Configuration Steps	99
2.26.1.5	Verification	100
2.26.2	Configuration Example 2	100
2.26.2.1	Networking Requirements	100
2.26.2.2	Network Topology	100
2.26.2.3	Configuration Tips	100
2.26.2.4	Configuration Steps	100
2.26.2.5	Verification	101
2.26.3	Configuration Example 3	102
2.26.3.1	Networking Requirements	102
2.26.3.2	Network Topology	103
2.26.3.3	Configuration Tips	103
2.26.3.4	Configuration Steps	103
2.26.3.5	Verification	104
3	CONFIGURING HQoS	107
3.1	Understanding HQoS	107
3.1.1	HQoS Overview	107
3.1.2	Basic Concepts	107
3.1.2.1	FQ	107
3.1.2.2	UQ	107
3.1.2.3	GQ	108
3.1.2.4	Destination Interface/Destination Device CQ	108
3.1.2.5	LPQ	108
3.1.3	How HQoS Works	109
3.1.3.1	Uplink HQoS Scheduling	109

3.1.4	Downlink HQoS Scheduling	110
3.1.5	Protocol Specifications	111
3.2	Default Configuration	112
3.3	Configuring CoS-based HqoS	112
3.3.1	Configuring a Traffic Classifier Rule	112
3.3.2	Configuring an FQ WRED Template	112
3.3.3	Configuring an FQ Template	113
3.3.4	Configuring an FQ Mapping Template	113
3.3.5	Configuring a UQ	114
3.3.6	Configuring a GQ	114
3.3.7	Configuring a Traffic Behavior Rule	115
3.3.8	Configuring a Traffic Policy Rule	115
3.3.9	Applying a Traffic Policy to an Interface	115
3.3.10	Configuring a CQ Template	116
3.3.11	Applying the CQ to an Interface	116
3.3.12	Applying Resource Reservation Queue to Interface	117
3.3.13	Configuring HQoS Policy for AP Scenario	117
3.3.14	Configuring a Static VoQ Credit Point	118
3.3.15	Configuring System VoQ Scheduling	118
3.3.16	Clearing Queue Statistics	118
3.3.17	Showing Configurations	118
3.4	Typical HQoS Configuration Examples	120
3.4.1	Configuration Example 1	120
3.4.1.1	Networking Requirement	120
3.4.1.2	Networking Topology	120
3.4.1.3	Configuration Points	120
3.4.1.4	Configuration Steps	120
3.4.1.5	Verification	122
3.4.2	Configuration Example 2	122
3.4.2.1	Networking Requirement	122
3.4.2.2	Networking Topology	122
3.4.2.3	Configuration Points	123
3.4.2.4	Configuration Steps	123
3.4.2.5	Verification	124
3.4.3	VDA Configuration Example	124
3.4.3.1	Networking Requirements	124
3.4.3.2	Network Topology	124
3.4.3.3	Configuration Key Points	124

3.4.3.4	Configuration Steps	125
3.4.3.5	Displaying	125
3.5	Examples for Configuring RTP Traffic Shaping	125

1 CONFIGURING QOS

1.1 What is QoS

In a traditional IP network, the router treats all packets in the same way on a First In First Out (FIFO) basis and sends them to their destinations at best effort. However, it does not provide any assurance for the performances such as reliability and transmission delay of the packets.

As Internet becomes increasingly popular worldwide and IT is more widely used in social activities, people demand more and more from the network. IT needs evolve from the pure data information to the interactive multimedia information, from separate service to integrated transmission of data, voice and image services on a single network. More and more voice and image and other important data that require low bandwidth delay and be jitter sensitive and highly real time are transmitted over the network. On one hand, network resources become much diversified. On the other hand, the assurance of the service quality of the network incurs an important problem since the data, voice and image services have different requirements in delay, throughput or packet loss rate.

One solution to this problem is to increase the network bandwidth, which however is limited and costly as well. Other means to assure the service quality include the use of the techniques such as Policy-Based Routing, Congestion Management, Congestion Avoidance, Traffic Shaping and transmission compression to manage the traffic on the network and meet the needs of the increasing traffic on the network.

QoS (Quality of Service) is the ability of a network to provide better services for the specified network communications by using various basic technologies. To put it in simple way, different network service qualities are provided to suit the specific requirements: better qualities for the packets that are highly real-time and important; lower qualities for the common packets that do not need to be real time. To bear various services on the network, the network must be able to not only provide individual services but also offer different QoSs for them. It can be said for sure that QoS is a basic requirement for future IP networks.

Qtech devices implement various QoS policies to meet the needs of different services for different QoSs.

1.2 Why QoS

QoS allows the network provides services for various network applications and communications in a proactive way. On the network, the QoS can be used to:

- Control resources: Users can control the network resources being used. For example, users can exercise control over the network resources occupied by FTP (File Transfer Protocol) transmission or assign higher priority for important data access.
- Sub-divide services. An ISP as a user can provide services of different QoS for different customers and packets of different requirements.
- In a network environment, different QoSs are provided for different applications to ensure network services for important packets. For example, prudential services are provided for important data; minimum delay is enabled for time-sensitive multimedia and voice applications.
- In addition, QoS lays a good foundation for future integration.

1.3 QoS under Differentiated Services

- What are differentiated services

For Differentiated Services (DiffServ), several services with the same feature are converged by using the mechanisms of DSCP (DiffServ CodePoint) and PHB to provide services for the entire converged traffic, instead of individual services.

The following three levels of differentiated services are provided:

1. Expedited Forwarding (EF)
2. Assured Forwarding (AF)
3. Best Effort (BE)

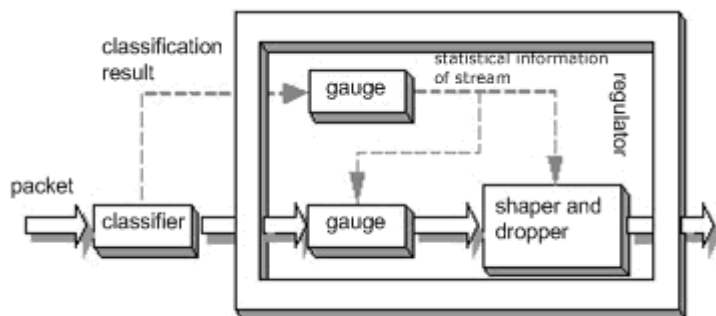
DiffServ greatly reduces the work of signaling and focuses instead on flow convergence and a set of “hop-by-hop behaviors” suitable for networkwide services. Data flows can be classified according to the pre-determined rules in order to converge multiple application traffic into a certain level of data flow.

Therefore, QoS is based on DiffServ system.

■ QoS under Differentiated Services

In DiffServ, QoS includes classification/identification, measurement/shaping/packet dropping, and finally congestion management and congestion avoidance. The basic processing is illustrated as below:

Figure 1 Schematic diagram for the border node to classify and adjust the packets



The follow-up chapters separately describe the QoS under differentiated services.

1.4 IPv6 QoS

IPv6 QoS supports the 8-bit Traffic Class field, which has the same function as the ToS field of IPv4 to identify the service type of packets. The DSCP of IPv6 takes the first 6 bits in the Traffic Class field as DHCP value. Thus $DSCP = (TC \& 11111100) \gg 2$. TC value is given by DSCP-to-TC mapping. IPv6 QoS also supports the 20-bit flow label field indicating the traffic belonging to the same classification. Currently, the definition and use of flow label is still in the draft phase, only Traffic Class-based QoS is supported by IPv6 QoS.

1.5 Congestion Management

1.6 What Is Congestion Management

Congestion occurs in a network node when packets arrive faster than an interface can send them. If the buffer space is insufficient for storing data on the network node, packet loss occurs. Network protocols, such as the Transmission Control Protocol (TCP), provide the data retransmission mechanism, in which the data sender node retransmits data when failing to receive a reply from the peer receiver node. This causes congestion on the peer receiver node and the performance of the entire network deteriorates.

The following are sample causes of congestion:
 A packet flow enters **the router** through a high-speed link and leaves the device through a low-speed link;
 Packet flows enter **the router** concurrently through multiple interfaces and leaves the device through only one interface;
 CPU runs slowly.

Assume that the network needs to transmit some important data as well as much unimportant data. If the device processes all data in the same way with no regard to their importance level, unimportant data uses bulk network bandwidth and transmission of important data is delayed, which may cause tremendous loss.

To resolve these problems, congestion management is introduced. Congestion management features allow you to control congestion by determining the order in which packets are sent out an interface based on priorities assigned to those packets. A network device such as router determines the packet transmission order by controlling which packets are transmitted with priority, ensuring that key services are processed timely.

1.7 Policies of Congestion Management

Congestion management performs three tasks: Create various types of queues. Classify packets and place them to different queues. Schedule queues and send packets in queues based on rules. The congestion management QoS features provide five types of queuing. Each type of queuing allows you to create a different number of queues. During periods with light traffic, that is, when no congestion exists, packets are sent out of the interface as soon as they arrive. During periods of transmit congestion at the outgoing interface, packets arrive faster than the interface can send them. If you use congestion management features, packets accumulating at an interface are queued until the interface is free to send them; they are then scheduled for transmission according to their assigned priority and the queuing mechanism configured for the interface. The device determines the packet transmission order by controlling which packets are placed in which queue and how queues are serviced with respect to each other.

This document discusses six types of queuing, which constitute the congestion management QoS features.



Note The NPE80 supports only the First-In, First-Out Queuing (FIFO) and Weighted Fair Queuing (WFQ).

1.7.1 FIFO

FIFO entails no concept of priority or traffic classes. With FIFO, transmission of packets out of the interface occurs in the order the packets arrive. FIFO is the default queuing mechanism that needs no intentional configuration.

1.7.2 WFQ

WFQ offers dynamic, fair queuing that divides bandwidth across queues of traffic based on weights. WFQ ensures that all traffic is treated fairly, given its weight. Given this handling, WFQ ensures satisfactory response time to critical applications, such as interactive, transaction-based applications, which are intolerant of performance degradation.

For WFQ, you define traffic classes based on source addresses, destination addresses, source port numbers, destination port numbers, and protocol types.

1.7.3 CBWFQ

Class-Based Weighted Fair Queuing (CBWFQ) extends the standard WFQ functionality. Same as WFQ, CBWFQ offers dynamic, fair queuing that divides bandwidth across queues of traffic based on weights. The difference lies in classification rules and weight calculation. For WFQ, you define traffic classes based on source addresses, destination addresses, source port numbers, destination port numbers, and protocol types. For CBWFQ, you define traffic classes based on user-defined criteria. WFQ weighs packet priorities based on fixed rules, for example, weighing the priority of an IP packet based on the Type of Service (ToS) domain. CBWFQ assigns communication queue bandwidth by proportion by weighing priorities based on user-defined bandwidth criteria.

CBWFQ allows you to define traffic classes and assign bandwidth in real time. Given these features, CBWFQ allows you to customize bandwidth assignment and ensures that different types of network data traffic acquire bandwidth by proportion.

1.7.4 PQ

With PQ, packets belonging to one priority class of traffic are sent before all lower priority traffic to ensure timely delivery of those packets.

PQ guarantees strict priority for important network data and quickest processing of most-important network data on a network node where PQ is enabled. For PQ, priority can be defined flexibly based on network protocols such as IP, data input interfaces, packet lengths, source addresses, and destination addresses.

1.7.5 LLQ and RTPQ

LLQ indicates Low Latency Queuing. The LLQ feature brings strict Priority Queuing (PQ) to CBWFQ. Strict PQ allows packets that are delay-sensitive and match traffic criteria to be sent and sent before packets in CBWFQ queues.

Functions of Real-time Transport Protocol Priority Queuing (RTPQ) are similar to those of LLQ. Each interface possesses an RTPQ queue exclusively used in the low-delay transmission of RTP packets. RTPQ matches the User Datagram Protocol (UDP) packets of ports in a specified range.

1.7.6 CQ

With Custom Queuing (CQ), each class of traffic acquires bandwidth by proportion. CQ allows you to assign bandwidth to all classes of traffic based on packet importance. Thus, important packets are delivered timely. You can also define the number of the bytes or packets abstracted from queues.

This function is suitable for interfaces that process data at low speeds.



Note You can assign only one queuing mechanism to an interface.

1.8 Deciding Which Queuing Policy to Use

Qtech devices can meet service quality requirements of different services to some degree by implementing FIFO, PQ, CQ, WFQ, CBWFQ, and LLQ&RTPQ. The following section compares the five main queuing strategies.

- FIFO queuing performs the default first-come-first-served prioritization of packets in user data traffic. It entails no concept of priority or classes of traffic. When FIFO is used, ill-behaved sources can consume available bandwidth, bursty data sources can cause delay in time-sensitive or important traffic, and important traffic may be dropped because less important traffic fills the queue.
- CQ guarantees some level of service to all traffic because you can assign bandwidth to all classes of traffic. You can define the size of the queue by determining its configured packet-count capacity, thereby controlling bandwidth usage.
- PQ guarantees strict priority in that it ensures that one type of traffic will be sent, possibly at the expense of all others. For PQ, a low priority queue can be detrimentally affected, and, in the worst case, never allowed to send its packets if a limited amount of bandwidth is available or if the transmission rate of critical traffic is high.
- WFQ does not use access lists to determine the preferred traffic on an interface. Rather, the fair queue algorithm dynamically sorts traffic into messages that are part of a conversation. Low-volume, interactive traffic gets fair allocation of bandwidth with WFQ, as does high-volume traffic such as file transfers.
- CBWFQ allows you to define traffic classes and assign bandwidth in real time. Specifically, CBWFQ allows you to specify the exact amount of bandwidth to be assigned for a specific class of traffic, which guarantees bandwidth of certain network applications. You can control bandwidth allocation on demand at any time.

The following table compares the main queuing strategies.

	FIFO	WFQ	CBWFQ	PQ	CQ
Number of Queues	1	Configurable (default: 256)	320 (64 CBWFQ queues, and 256 WFQ queues)	4	17 (16 user queues, and one system queue)
Advantage	Fast and simple processing	All traffic is treated fairly, given its weight.	Traffic that matches user-defined criteria is placed in CBWFQ queues, and other traffic is placed in WFQ queues. Network packets are delivered in proportion to their configured bandwidth.	High priority queues are serviced first. Absolute prioritization ensures critical traffic of highest priority.	CQ assigns bandwidth by proportion for different types of services. If one queue is empty, its assigned bandwidth is automatically added to another queue that had packets ready to send.

	FIFO	WFQ	CBWFQ	PQ	CQ
Defect	All packets are treated fairly. Packet delivery sequence is determined by their sequence of arriving at the interface, which may cause delay in delivery of critical applications.	Processing speed is slower than that of FIFO.	Processing speed is slower than that of FIFO.	Processing speed is slow, and a low priority queue can be never allowed to send its packets in the worst case.	Processing speed is relatively slow.
Configuration requirement	No configuration required	Simple configuration required	Configuration required	Configuration required	Configuration required

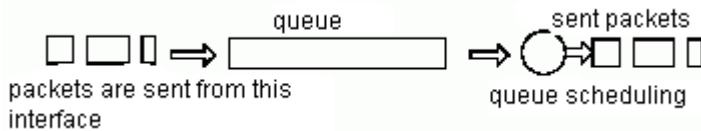
1.9 Work Mechanism of Congestion Management

1.9.1 FIFO

1.9.1.1 Working Principles

FIFO implements a simple service rule and provides only one queue. Packets are delivered based on the first-come-first serviced rule. FIFO does not guarantee delay restrictions or delivery rates, cannot isolate service flows that share a link. Therefore, FIFO cannot fairly treat service flows that share a link.

Figure 2 FIFO queuing



As shown in the preceding figure, packets are sent out an interface in the order in which they arrive. FIFO does not intervene with packets, but allows packets to use bandwidth and other resources in the order in which they arrive. Ill-behaved applications or attacks can consume all the bandwidth and important traffic can be dropped.

1.9.1.2 Applicable Environment

When there is no other queuing mechanism configured on the network, FIFO is enabled on all interfaces by default. FIFO possesses the quickest processing speed among queuing mechanisms. If congestion rarely occurs, FIFO is the most suitable on the network.

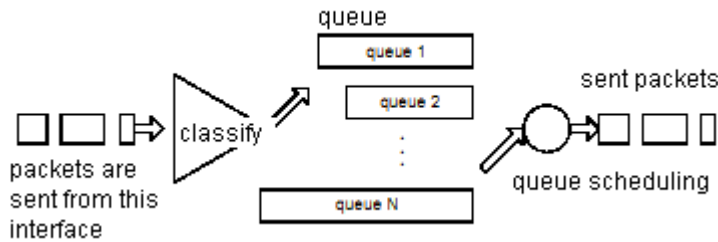
1.9.2 WFQ

1.9.2.1 Working Principles

WFQ removes FIFO restrictions. When FIFO queuing is enabled, traffic is sent out an interface in the order in which they arrive, disregarding bandwidth consumption and delay. As a result, file transfer and other network applications with large data volumes often generate packet links. A packet link consists of groups of packets and is transferred as a whole. These packet links may consume all available bandwidth and other traffic is dropped.

WFQ ensures that each flow shares link bandwidth fairly and that low-volume traffic gets transmitted in a timely fashion.

Figure 3 WFQ



As shown in the preceding figure, WFQ classifies packets by flow. A flow consists of packets of same source IP address, destination IP address, source MAC address, destination MAC address, source port number, destination port number, protocol type, and ToS. A flow is assigned to a queue with bandwidth in proportion to flow priority during delivery. A flow of high-priority gets more bandwidth than a flow of low-priority. Specifically, bandwidth assigned to a flow divided by the entire bandwidth of the link is equal to the sum (flow priority + 1) divided by the entire bandwidth of the link.

For example, seven flows exist on an interface, with their priority being 1, 2, 3, 4, 5, and 6. The bandwidth assigned to the interface equals the sum of all the flow priorities plus one:

$$1 + 2 + 3 + 4 + 5 + 6 + 7 = 28$$

Bandwidth ratio of a flow equals to the sum (flow priority + 1) divided by another sum (sum of all flow priorities + 1). Therefore, bandwidth ratios of flows are 1/28, 2/28, 3/28, 4/28, 5/28, 6/28, and 7/28.

If there are 10 data flows whose priority is 1, the entire bandwidth is:

$$1 + 2 * 10 + 3 + 4 + 5 + 6 + 7 = 46$$

Priority-0 data flows use 1/46 of the entire bandwidth, Priority-1 data flows use 2/46 of entire bandwidth, and accordingly Priority-6 data flows use 7/46 of the entire bandwidth.

When data flows are added or terminated, actual bandwidth changes consequently. By assigning bandwidth of each flow in real time, WFQ is suitable for the network environment that is constantly changing.

1.9.2.2 Applicable Environment

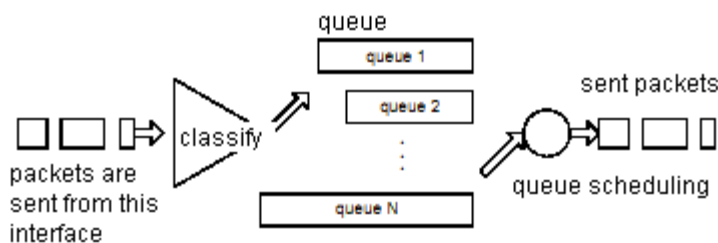
WFQ is suitable for serial interfaces with bandwidth equal to or less than 2.048 Mbps.

1.9.3 CBWFQ

1.9.3.1 Working Principles

CBWFQ extends the standard WFQ functionality to provide support for assigning bandwidth based on user-defined traffic classes.

Figure 4 CBWFQ



As shown in the preceding figure, CBWFQ defines classes for packets by flow. At first, classes are defined based on user-defined based on user-defined rules. Packets satisfying a same match criterion belong to a network data flow and are placed in a same CBWFQ queue. Packets satisfying no match criterion are classified in WFQ mode. Packets with the same source IP address, destination IP address, source MAC address, source port number, destination port number, protocol type, and ToS field belong to the same flow. When a packet is assigned to a flow, it

is placed in the WFQ queue reserved for that flow. During delivery, CBWFQ assigns bandwidth based on the user-defined rules.

When data flows are added or terminated, actual bandwidth assigned by CBWFQ changes consequently. Therefore, CBWFQ is also suitable for the network environment that is constantly changing.

1.9.3.2 Applicable Environment

CBWFQ is suitable for applying strict classification rules and bandwidth allocation on serial interfaces with bandwidth equal to or less than 2.048 Mbps.

1.9.4 LLQ and RTPQ

1.9.4.1 Working Principles

LLQ extends the standard CBWFQ functionality. LLQ ensures that delay-sensitive data acquire bandwidth and are sent before packets in other queues are dequeued. The LLQ feature brings strict Priority Queuing (PQ) to CBWFQ. Strict PQ allows packets in LLQ queues to be sent and sent before packets in CBWFQ queues.

For LLQ queues, traffic of different types is monitored separately. If there is no congestion, traffic is allowed for delivery. In the event of congestion, transmission rates of all traffic are monitored, and packets are dropped if the bandwidth is exceeded.

Functions of RTPQ are similar to those of LLQ. Each interface possesses an RTPQ queue exclusively used in the low-delay transmission of RTP packets. RTPQ matches the UDP packets of ports in a specified range.

For RTPQ queues, traffic of different types is monitored separately. If there is no congestion, traffic is allowed for delivery. In the event of congestion, transmission rates of all traffic are monitored, and packets are dropped if the bandwidth is exceeded.

Both an RTPQ queue and an LLQ queue belongs can belong to only one interface, but the priority of an RTPQ queue is higher than that of an LLQ queue.

1.9.4.2 Applicable Environment

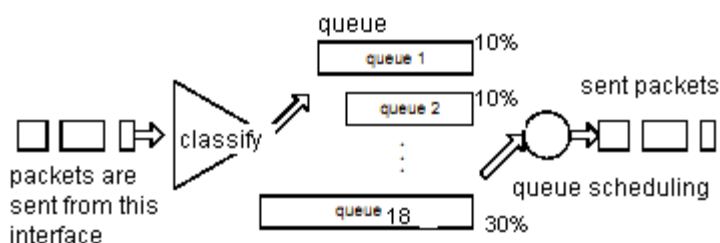
LLQ and RTPQ+CBWFQ are suitable for applying strict classification rules and bandwidth allocation on serial interfaces with bandwidth equal to or less than 2.048 Mbps.

1.9.5 CQ

1.9.5.1 Working Principles

CQ defines 17 classes for packets, corresponding to 17 CQ queues. Packets enter corresponding CQ queues based on their classes and the FIFO policy. For queues 1 through 16, queue 0 is a system queue; queue 1 through 16 are user queues. Associated with each queue is a configurable byte count, which specifies how many bytes of data the system should deliver from the current queue before it moves on to the next queue. The system queue is emptied before any of the queues 1 through 16 are processed. For queues 1 through 16, the system cycles through the queues using the pre-assigned bandwidth in a round-robin fashion, the system dequeues the configured byte count from each queue in each cycle and delivers packets in the current queue before moving on to the next one.

Figure 5 CQ



CQ ensures that no application or specified group of applications achieves more than a predetermined proportion of overall bandwidth when the line is under stress. Like PQ, CQ is statically configured and does not automatically adapt to changing network conditions. With CQ enabled, the system takes longer and consumes more resources to switch packets because packets need to be queued.

1.9.5.2 How to Determine Byte Count Values for Queues

In order to assign bandwidth to different queues, you must specify the byte count for each queue. This section describes how to determine byte count values for queues.

When the router cycles through queues in round-robin fashion, the device sends packets from a particular queue until the byte count is exceeded. If the byte count value is exceeded but packets of the queue are not completely sent, the device continues sending until all packets of the queue are sent. Therefore, if you set the byte count to 300 bytes and the packet size of your protocol is 1500 bytes, then every time this queue is serviced, 1500 bytes will be sent, not 300 bytes.

For example, suppose one protocol has 500-byte packets, another has 300-byte packets, and a third has 100-byte packets. If you want to split the bandwidth evenly across all three protocols, you might choose to specify byte counts of 200, 200, and 200 for each queue. However, this configuration does not result in a 33/33/33 ratio. When the device services the first queue, it sends a single 500-byte packet; when it services the second queue, it sends a 300-byte packet; and when it services the third queue, it sends two 100-byte packets. The effective ratio is 50/30/20.

Thus, setting the byte count too low can result in an unintended bandwidth allocation. However, very large byte counts will produce a “jerky” distribution. That is, if you assign 10 KB, 10 KB, and 10 KB to three queues in the example given, each protocol is serviced promptly with equal bandwidth assigned when its queue is the one being serviced, but it may be a long time before the queue is serviced again. A better solution is to specify 500-byte, 600-byte, and 500-byte counts for the queue. This configuration results in a ratio of 31/38/31, which may be acceptable. In order to service queues in a timely manner and ensure that the configured bandwidth allocation is as close as possible to the required bandwidth allocation, you must determine the byte count based on the packet size of each protocol; otherwise your percentages may not match what you configure.

To determine the correct byte counts, perform the following steps:

- For each queue, divide the percentage of bandwidth you want to assign to the queue by the packet size, in bytes. For example, assume the packet size for protocol A is 1086 bytes, protocol B is 291 bytes, and protocol C is 831 bytes. You want to assign 20 percent for A, 60 percent for B, and 20 percent for C. The ratios would be:

$$\frac{20}{1086}, \quad \frac{60}{291}, \quad \frac{20}{831}$$
 or 0.01842, 0.20619, 0.02407
- Normalize the numbers by dividing by the lowest number:

$$1, \quad 11.2, \quad 1.3$$
 The result is the ratio of the number of packets that must be sent so that the percentage of bandwidth that each protocol uses is approximately 20, 60, and 20 percent.
- A fraction in any of the ratio values means that an additional packet will be sent. Round up the numbers to the next whole number to obtain the actual packet count. In this example, the actual ratio will be 1 packet, 12 packets, and 2 packets.
- Convert the packet number ratio into byte counts by multiplying each packet count by the corresponding packet size. In this example, the number of packets sent is one 1086-byte packet, twelve 291-byte packets, and two 831-byte packets, or 1086, 3492, and 1662 bytes, respectively, from each queue. These are the byte counts you would specify in your CQ configuration.
- To determine the bandwidth distribution this ratio represents, first determine the total number of bytes sent after all three queues are serviced:

$$(1 \times 1086) + (12 \times 291) + (2 \times 831) = 1086 + 3492 + 1662 = 6240$$
- Then determine the percentage of the total number of bytes sent from each queue:

$$\frac{1086}{6240}, \quad \frac{3492}{6240}, \quad \frac{1662}{6240} = 17.4, \quad 56, \quad \text{and} \quad 26.6 \text{ percent}$$
 This result is close to the desired ratio of 20/60/20.
- If the actual bandwidth is not close enough to the desired bandwidth, multiply the original ratio of 1:11.2:1.3 by the best value, trying to get as close to three integer values as possible. Note that the multiplier you use need not be an integer. For example, if we multiply the ratio by two, we get 2:22.4:2.6. You would now send two 1086-byte packets, twenty-three 291-byte packets, and three 831-byte packets, or 2172/6693/2493, for a total

of 11,358 bytes. The resulting ratio is 19/59/22 percent, which is much closer to the desired ratio that we achieved.

Window size also affects the bandwidth distribution. If the window size of a particular protocol is set to one, then that protocol will not place another packet into the queue until it receives an acknowledgment. The CQ algorithm moves to the next queue if the byte count is exceeded or no packet is in that queue. Therefore, with a window size of one, only one frame will be sent each time. If your frame count is set to 2 kilobytes, and your frame size is 256 bytes, then only 256 bytes will be sent each time this queue is serviced.

1.9.5.3 Applicable Environment

This function is suitable for interfaces that process data at low speeds.



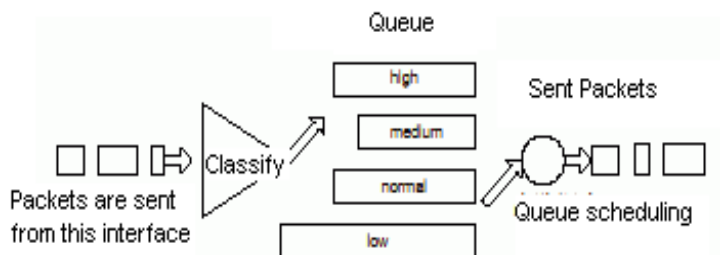
Note CQ is not supported on any tunnels.

1.9.6 PQ

1.9.6.1 Working Principles

PQ allows you to define how traffic is prioritized in the network. You can configure four traffic priorities. You can define a series of filters based on packet characteristics (source addresses, destination addresses, protocol types, and packet sizes) to place traffic into these four queues; the queue with the highest priority is serviced first until it is empty, then the lower queues are serviced in sequence.

Figure 6 PQ



As shown in the preceding figure, packets are classified and placed in four output types of output queues based on user-defined criteria. The priority queues on that interface are scanned for packets in descending order of priority. The high priority queue is scanned first, then the medium priority queue, and so on. The packet at the head of the highest queue is chosen for transmission. This procedure is repeated every time a packet is to be sent.

When choosing to use PQ, consider that because lower priority traffic is often denied bandwidth in favor of higher priority traffic, use of PQ could, in the worst case, result in lower priority traffic never being sent. PQ introduces extra system resource consumption that is acceptable for slow interfaces, but may not be acceptable for higher speed interfaces such as Ethernet. With PQ enabled, the system takes longer to switch packets, degrading system performance. PQ uses a static configuration and does not adapt to changing network conditions.

1.9.6.2 Applicable Environment

Although you can enable PQ for any interface, it is best used for low-bandwidth, congested serial interfaces.



Note PQ is not supported on any tunnels.

1.10 WFQ

1.10.1 WFQ Configuration Tasks

When standard WFQ is enabled, packets are classified by flow. Packets with the same source IP address, destination IP address, source TCP or UDP port, and destination TCP or UDP port belong to the same flow. WFQ assigns an equal share of bandwidth to each flow. Flow-based WFQ is also called fair queuing because all flows are equally weighted.

Configuring WFQ entails the following two processes:

- Configuring WFQ
- Monitoring fair queuing



Note Memory configuration is different for routers on different platforms. You are advised to run the **fair-queue** command to configure depth and count of different queues.

1.10.2 Configuring WFQ

To configure WFQ, run the following commands in the interface configuration mode:

Command	Function
Qtech(config-if)# fair-queue [<i>congestive-discard-threshold</i> [<i>dynamic-queues</i>]]	Configure WFQ.
Qtech(config-if)# no fair-queue	Cancel WFQ configuration.

The following table describes parameters related to WFQ configuration.

Command	Function
<i>congestive-discard-threshold</i>	Specify the maximum threshold of packets allowed in each queue. The default value is 64 and the value range is 1–4096. When the packet count reaches the threshold, newly arrived packets will be discarded. This parameter is optional.
<i>dynamic-queues</i>	Specify the number of dynamic queues. The default value is 256 and the value must be 2 ⁿ within 16–4096. This parameter is optional.



Note To configure the WFQ congestion management policy, ensure that the fast-switching function configuration (enable or disable the function) is consistent on all system interfaces. Otherwise, the congestion management policy becomes invalid.



Note The value of *dynamic-queues* needs to be adjusted dynamically according to the current service traffic status. Ensure that the value of *dynamic-queues* be greater than the number of service flows; otherwise, multiple service flows enter a same dynamic queue. You are advised to set the *dynamic-queues* to a value greater than both 64 and the number of service flows.

1.10.3 Monitoring WFQ

To view information about WFQ that has been configured, run the following commands in the privileged user mode:

Command	Function
Qtech# show queue wfq	Show configuration information about WFQ queues.
Qtech# show queue interface <i>interface-name interface-number</i>	Show interface statistic information about WFQ queues.

For routers of the RSR series, you can view statistic information about fast-switching WFQ queue interfaces by running the **show queue interface** command. "Qos Ref queue information" identifies statistic information about fast-switching WFQ queue interfaces.

1.10.4 WFQ Configuration Example

In the following instance, fair queueing is configured on synchronization port 0. Specifically, serial port 0 is configured with 128 message queues, 512 dynamic queues, and 50 reserved queues.

```
interface Serial 1/0
ip address 1.1.1.1 255.255.255.0
fair-queue 128 512
```

The following are examples of viewing the interface configuration in the privileged user mode:

```
Qtech# show queue interface serial 1/0
Queuing strategy: weighted fair
Output queue: 0/300/128/0 (size/max total/threshold/drops)
Output queue num: 0/0/512 (now active/max active/max total)
```

The preceding output shows that WFQ is enabled on the interface and the threshold for discarding packets is **128** in the event of congestion.

1.11 CBWFQ

1.11.1 CBWFQ Configuration Tasks

By default, the QoS policy of FIFO is enabled on Qtech network interfaces with bandwidth lower than 4 Mbps.

1.11.2 Configuring CBWFQ

1.11.2.1 Defining Class-Maps

This function is required for realizing CBWFQ functionality. You can define packet classification rules in class-maps, and specify the class-map names for using them in policy-maps. A class-map can be used by one or more policy-maps. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# class-map <i>match-all class-map-name</i>	Access and create the class-map of the "AND" type. That is, all conditions in the class-map must be met.
Qtech(config)# class-map <i>match-any class-map-name</i>	Access and create the class-map of the "OR" type. That is, all conditions in the class-map must be met.

<p>Qtech(config-cmap)# match access-group <i>access-list-number</i></p> <p>or</p> <p>Qtech(config-cmap)# match input-interface <i>interface-name</i></p> <p>or</p> <p>Qtech(config-cmap)# match protocol <i>protocol-name</i></p> <p>or</p> <p>Qtech(config-cmap)# match cos <i>value</i></p> <p>or</p> <p>Qtech(config-cmap)# Match ip dscp <i>value</i></p> <p>or</p> <p>Qtech(config-cmap)# Match ip precedence <i>value</i></p> <p>or</p> <p>Qtech(config-cmap)# Match not <i>match-type value</i></p>	<p>Configure the packet classification rules. The rules can be the positive or negative forms of the ACL, packet receiving interface, encapsulation protocol type, COS value, IP DSCP code, and IP Precedence code.</p>
<p>Qtech(config-cmap)# exit</p>	<p>Exit from the configuration layer of class-maps.</p>

Class-map-name: indicates the name of a class-map.

Match-all: All conditions in the class-map must be met. By default, a created class-map is of the **Match-all** type.

Match-any: indicates that only one condition in the class-map needs to be met.

Access- list-number: indicates the number of the access control list (ACL).

Interface- name: indicates the name of a network interface.

Protocol-name: indicates the packet encapsulation protocol.

COS: indicates the COS value of an Ethernet packet.

IP dscp: indicates the DSCP code value of the packet IP TOS domain.

IP Precedence: indicates the Precedence code value of the packet IP TOS domain.

Not match-type: indicates the negative form of a classification rule.

1.11.2.2 Configuring Rules in Policy-Maps

This function is required for realizing CBWFQ functionality. In a policy-map, you can use all class-maps configured on the router and use a maximum of 64 class-maps concurrently. You can define bandwidth for class-maps that are used under the precondition that the sum of bandwidth for all class-maps is equal to or smaller than the bandwidth assigned to CBWFQ on a specified interface that applies the policy-map. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# bandwidth { <i>bandwidth-kbps</i> percent <i>percent-number</i> }	Specify bandwidth assigned to specified types of data flows.
Qtech(config-pmap-c)# queue-limit <i>number-of-packets</i>	Set the queue depth.
Qtech(config-pmap-c)# exit	Exit from the configuration layer of class using.
Qtech(config-pmap)# exit	Exit from the configuration layer of policy-maps.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Bandwidth-kbps: indicates the assigned bandwidth in kbps.

Percent-number: indicates the assigned-bandwidth to all-available-bandwidth ratio.

Number-of-packets: indicates the depth of a CBWFQ queue (the maximum number of packets that are allowed).

1.11.2.3 Applying Service Rules on Specified Interfaces

This function is required for realizing CBWFQ functionality. If service rules are enabled on specified interfaces, CBWFQ functionality is enabled and classes used in policy-maps obtain their processing queues. The following table describes the typical configuration of this function.

Command	Function
Qtech(config-if)# service-policy output policy-map-name	Enable CBWFQ and specify the policy-maps to be applied.
Qtech(config-if)# service-policy input policy-map-name	Enable the policy-map strategy for interface input packets.

Policy-map-name: indicates the name of a policy-map.



Note If CBWFQ characteristics (such as the **Bandwidth** command or the **Priority** command) are enabled for a policy-map, the policy-map can be only be used in the **Service-policy Output** command, but not the **Service-policy Input** command.



Note To configure the policy-map of an interface, ensure that the fast-switching function configuration (enable or disable the function) is consistent on all system interfaces. Otherwise, the functions such as CBWFQ and police of the policy-map become invalid.



Note Ensure that the fast-switching function is disabled, if an interface configured with an input-direction policy-map or associated with an output-direction policy mapping of the shape and red functions. This is a required in the current software version.

1.11.2.4 Configuring Bandwidth for Specified Classes

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# policy-map policy-map-name	Access and create a policy-map.
Qtech(config-pmap)# class class-name	Use class-maps that have been defined.
Qtech (config-pmap-c)# bandwidth { bandwidth-kbps percent percent}	Specify bandwidth assigned to specified types of data flows.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Bandwidth-kbps: indicates the assigned bandwidth in kbps.

Percent-number: indicates the assigned-bandwidth to all-available-bandwidth ratio.

You can specify bandwidth assigned to specified types of data flows. By default, the system assigns 1% of the bandwidth to a specified type of data flow.

1.11.2.5 Configuring Queue Depth for Specified Classes

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# queue-limit <i>number-of-packets</i>	Set the queue depth.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Number-of-packets: indicates the depth of a CBWFQ queue (the maximum number of packets that are allowed).

You can set the depth of a CBWFQ queue corresponding to a specified type of data flow. The system default value is **64**. That is, the system will discard packets attempting to enter a CBWFQ queue if the queue has got 64 packets. In this case, Qtech products support the congestion processing only in Tail-Drop mode, not in WRED mode.



Note

Queue depth configuration depends on network requirements. When forwarded data is delay-sensitive, you can decrease the queue depth to lower delay. When the forwarded data is burst or contains many small packets, you can increase the queue depth to improve the system buffer capability. Do not adjust the queue depth to a value that is too small; otherwise, the bandwidth guarantee function becomes abnormal. In an environment where the forwarded data is burst or contains many small packets, bandwidth may fail to be guaranteed. You need to increase the queue depth to improve the system buffer capability.

1.11.2.6 Configuring the DSCP Code Value of the IP TOS Domain for a Specified Class

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# set ip dscp <i>values</i>	Set the packet DSCP value.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Values: indicates the packet DSCP value to be set.

1.11.2.7 Configuring the DSCP Code Value of the IPv4 TOS Domain and the IPv6 Traffic Class Domain for a Specified Class

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
---------	----------

Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# set dscp <i>values</i>	Set the packet DSCP value.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Values: indicates the packet DSCP value to be set.

1.11.2.8 Configuring the Precedence Code Value of the IP TOS Domain for a Specified Class

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# set ip precedence <i>values</i>	Set the packet Precedence value.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Values: indicates the packet Precedence value to be set.

1.11.2.9 Configuring the Precedence Code Value of the IPv4 TOS Domain and the IPv6 Traffic Class Domain for a Specified Class

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# set precedence <i>values</i>	Set the packet Precedence value.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Values: indicates the packet Precedence value to be set.

1.11.2.10 Configuring the COS Value of Ethernet Packets for a Specified Class

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# set cos <i>values</i>	Set the packet COS value.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Values: indicates the packet COS value to be set.

1.11.2.11 Configuring Bandwidth Assigned to CBWFQ

This function is optional for realizing CBWFQ functionality. The following table describes the typical configuration of this function.

Command	Function
Qtech(config-if)# max-reserved-bandwidth percent	Set bandwidth assigned to CBWFQ.

Percent: indicates percentage of the bandwidth assigned to CBWFQ on an interface.

You can customize the value of the **percent** parameter. The system default value is **75**. That is, 75% of all available bandwidth of the corresponding network interface will be assigned to CBWFQ queues.

1.11.3 Monitoring CBWFQ

To view information about input and output queues when CBWFQ is enabled on an interface, run the following commands in the privileged user mode:

Command	Function
Qtech# show class-map	Show information about all class-maps.
Qtech# show class-map <i>class-map-name</i>	Show information about a specified class-map.
Qtech# show policy-map	Show information about all policy-maps.
Qtech# show policy-map name <i>policy-map-name</i>	Show information about a specified policy-map.
Qtech# show policy-map name <i>policy-map-name</i> class <i>class-name</i>	Show information about specified class-maps in a specified policy-map.
Qtech# show policy-map interface <i>interface-name</i> <i>interface-number</i>	Show information about policy-maps applied on a specified interface.
Qtech# show queue interface <i>interface-name</i> <i>interface-number</i>	Show interface statistic information about CBWFQ queues.

Class-map-name: indicates the name of a class-map.

Policy-map-name: indicates the name of a policy-map.

Interface-name: indicates the name of a network interface.



Note For routers of the RSR series, you can view statistic information about fast-switching CBWFQ queue interfaces by running the **show queue interface** command. "Qos Ref queue information" identifies statistic information about fast-switching CBWFQ queue interfaces.



Note You can run the **show policy-map interface** command to view statistic information about fast-switching policy-maps on routers of the RSR series.

1.11.4 CBWFQ Configuration Example

In the following example, CBWFQ is enabled on a WAN interface Serial1/0 (S1/0) of the router. 30% of all available bandwidth on the interface S0 is used in the IP communication between the host 192.168.201.213 and the host 192.168.12.216. 10% of all available bandwidth on the interface S0 is used in the IP communication between the host 192.168.201.213 and the host 192.168.12.77.

The following is the configuration of the device where CBWFQ is enabled:

```
Qtech# show running-config
!
. . . .
!
class-map class1
match access-group 101
class-map class2
match access-group 102
!
policy-map policy1
class class1
bandwidth percent 30
class class2
bandwidth percent 10
!
interface FastEthernet0/0
ip address 192.168.201.1 255.255.255.0
!
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
service-policy output policy1
clock rate 115200
!
ip route 0.0.0.0 0.0.0.0 Serial1/0
access-list 101 permit ip host 192.168.201.213 host 192.168.12.216
access-list 102 permit ip host 192.168.201.213 host 192.168.12.77
!
. . . .
```

In the following example, the DSCP value is set for five types of packets matching the ACL on the ingress interface and the packets are transformed into four types of macro-flows for the differentiated services. The bandwidth assigned to CBWFQ queues is adjusted.

The following is the configuration of the device where CBWFQ is enabled:

```
class-map match-all dlsw
match access-group 101
class-map match-all voip
match access-group 102
class-map match-all notes
match access-group 103
class-map match-all http
match access-group 104
class-map match-all ftp
match access-group 105
class-map match-all ef
match ip dscp 46
class-map match-all af1
match ip dscp 10
class-map match-all af2
match ip dscp 18
class-map match-all af3
match ip dscp 26
!
!
policy-map 1
class dlsw
set ip dscp 46
class voip
set ip dscp 46
class notes
set ip dscp 10
class http
```

```

set ip dscp 18
class ftp
set ip dscp 26
policy-map 2
class ef
bandwidth 800
class af1
bandwidth 500
class af2
bandwidth 300
class af3
bandwidth 380
!
access-list 101 permit udp any any eq 2065
access-list 102 permit udp any any range 16384 32767
access-list 103 permit udp any any eq 1352
access-list 104 permit udp any any eq 80
access-list 105 permit udp any any eq 20
!
interface serial 2/1
encapsulation PPP
no ip route-cache
ip address 1.1.1.2 255.255.255.0
max-reserved-bandwidth 99
service-policy output 2
clock rate 2048000
!
interface FastEthernet 0/0
no ip route-cache
ip address 192.168.200.1 255.255.255.0
service-policy input 1
duplex auto
mac-address 00d0.3456.eeee
speed auto
!
ip route 0.0.0.0 0.0.0.0 serial 2/1
!

```

1.12 Configuring LLQ and RTPQ

1.12.1.1 Configuration Tasks of LLQ and RTPQ

LLQ configuration is combined with CBWFQ configuration, but RTPQ configuration is independent.

1.12.2 Configuring LLQ and RTPQ

To configure LLQ, run the following commands in the configuration mode of the Policy-map command layer:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-name</i>	Use class-maps that have been defined.
Qtech (config-pmap-c)# priority { <i>bandwidth-kbps</i> percent <i>percent</i> } { <i>Burst bytes</i> }	Specify bandwidth assigned to specified types of data flows.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Enable CBWFQ and specify the policy-maps to be applied.

Policy-map-name: indicates the name of a policy-map.

Class-map-name: indicates the name of a class-map.

Bandwidth-kbps: indicates the assigned bandwidth in kbps.

Percent-number: indicates the assigned-bandwidth to all-available-bandwidth ratio.

You can specify bandwidth assigned to specified types of data flows. By default, the system assigns 1% of the bandwidth to a specified type of data flow.

Burst bytes: indicates the byte count that can be exceeded.



Note Queue depth configuration depends on network requirements. When forwarded data is delay-sensitive, you can decrease the queue depth to lower delay. When the forwarded data is burst or contains many small packets, you can increase the queue depth to improve the system buffer capability. Do not adjust the queue depth to a value that is too small; otherwise, the bandwidth guarantee function becomes abnormal. In an environment where the forwarded data is burst or contains many small packets, bandwidth may fail to be guaranteed. You need to increase the queue depth to improve the system buffer capability.

To configure RTPQ, run the following commands in the configuration mode of the interface configuration command layer:

Command	Function
Qtech(config-if)# ip rtp priority <i>starting-rtp-port-number port-number-range bandwidth</i>	Create RTPQ queues for an interface and assign bandwidth to these queues.

Starting-rtp-port-number: indicates the starting port number among the matching UDP ports.

Port-number-range: indicates the port number range of the matching UDP ports.

Bandwidth-kbps: indicates the assigned bandwidth in kbps.



Note The following three commands affect the bandwidth assigned for interfaces:
 The **Bandwidth** command of CBWFQ
 The **Priority** command of LLQ
 The **ip rtp priority** command of RTPQ
 The **Bandwidth** command of the interface is used to set the entire bandwidth.
 That is, Bandwidth = max_reserve + default wfq bandwidth;
 max_reserve = bandwidth (policy-map) + priority (policy-map) +ip RTP priority



Note To configure the interface congestion management policy, ensure that the fast-switching function configuration (enable or disable the function) is consistent on all system interfaces. Otherwise, the congestion management policy becomes invalid.

1.12.3 Monitoring LLQ and RTPQ

To view information about LLQ that has been configured, run the following commands in the privileged user mode:

Command	Function
---------	----------

Qtech# show policy-map <i>interface-name interface-number</i>	Show information about policy-maps applied on a specified interface.
--	--

To view information about RTPQ that has been configured, run the following commands in the privileged user mode:

Command	Function
Qtech# show queue rtpq	Show information about RTPQ queues.



Note

For routers of the RSR series, you can view statistic information about fast-switching RTPQ queue interfaces by running the **show queue interface** command. "Qos Ref queue information" identifies statistic information about fast-switching RTPQ queue interfaces.

You can run the **show policy-map interface** command to view statistic information about fast-switching policy-maps on routers of the RSR series.

1.12.4 Configuration Examples of LLQ and RTPQ

In the following example, on SYNC port 0, an RTPQ queue is configured for serving RTP of VOIP and an LLQ queue is configured for serving DOSQ office packets. CBFWQ queues process Notes packets and default WFQ queues in CBFWQ process other packets.

```
!
class-map match-all dlsw
match access-group 101
class-map match-all voip
match access-group 102
class-map match-all notes
match access-group 103
class-map match-all http
match access-group 104
class-map match-all ftp
match access-group 105
!
policy-map 3
class dlsw
priority 30 2000
class notes
bandwidth 13
!
access-list 101 permit udp any any eq 2065
access-list 102 permit udp any any range 16384 32767
access-list 103 permit udp any any eq 1352
access-list 104 permit udp any any eq 80
access-list 105 permit udp any any eq 20
!
interface serial 2/1
encapsulation PPP
ip rtp priority 16384 16383 40
no ip route-cache
ip address 1.1.1.2 255.255.255.0
max-reserved-bandwidth 99
service-policy output 3
clock rate 64000
bandwidth 84
!
interface FastEthernet 0/0
ip address 192.168.200.1 255.255.255.0
duplex auto
```

```
mac-address 00d0.3456.eeee
speed auto
!
ip route 0.0.0.0 0.0.0.0 serial 2/1
!
```

1.13 Configuring CQ

1.13.1 CQ Configuration Tasks

1.13.1.1 Configuring CQ

You can configure a maximum of 16 groups for CQ. That is, the List-number range is 1–16. In each group, you can specify queues of packets, queue length, and byte count that can be continuously sent after a queue obtains the controlling right of sending packets.

1.13.1.2 Determining the Maximum Capacity of CQ

To configure the maximum packet capacity for each queue, run the following commands in the global configuration mode:

Command	Function
Qtech(config)# queue-list <i>list-number</i> queue <i>queue-number</i> limit <i>limit-number</i>	Specify the maximum packet count in a CQ queue. The no form of this command can be run to restore the default queue length 20.
Qtech(config)# queue-list <i>list-number</i> queue <i>queue-number</i> byte-count <i>byte-count-number</i>	Specify the maximum byte count in a queue. The no form of this command can be run to restore the default byte count 1500.

List-number: indicates the number of a queue list within 1–16.

Queue-number: indicates the number of a custom queue number within 1–16.

Limit-number: indicates the maximum packet count in a queue. The value range is 0–32767, and the default value is **20**.

Byte-count-number: specifies how many bytes of data should be delivered from the current queue by the system before the system moves on to the next queue until the value of **Byte-count-number** is exceeded or no packet is in that queue. The value range is 0–16777215, and the default value is **1500**. To determine the correct byte counts, see the related section.



Note To configure the interface congestion management policy, ensure that the fast-switching function configuration (enable or disable the function) is consistent on all system interfaces. Otherwise, the congestion management policy becomes invalid.



Note You should configure Byte-count-number based on traffic in each queue. Do not configure a large byte count for a slow traffic queue. Otherwise, the current queue may keep being dispatched, resulting in that other queues cannot be processed.

1.13.1.3 Assigning Packets to CQ Queues

You can assign a packet to a CQ queue based on the protocol type and the interface where the packet enters the device. You can also configure a default queue with multiple rules for packets matching no allocation criterion.

To configure the CQ queue list, run the following commands in the global configuration mode:

Command	Function
Qtech(config)# queue-list <i>list-number</i> protocol protocol-name queue-number [queue-keyword] [keyword-value]	Assign a packet to a CQ queue based on the protocol type.
Qtech(config)# queue-list <i>list-number</i> interface interface-type interface-number queue-number	Assign a packet to a CQ queue based on the type of the interface where the packet enters the device.
Qtech(config)# queue-list <i>list-number</i> default queue-number	Assign packets matching no assignment rules in the CQ queue list to a CQ queue. The default CQ queue number is 1.

For the description about **List-number** and **Queue-number**, see the preceding section.

Protocol-name: indicates the type of a protocol. The value can be **IP** (frequently used), **PAD**, and so on.

Queue-keyword and **Keyword-value:** indicates options about protocols. The following table describes values and meanings of the two parameters when IP is used.

queue-keyword	keyword-value	Meaning
Blank	Blank	All IP packets can enter a specified queue.
fragments	Blank	All fragment IP packets can enter a specified queue.
list	List-number	All packets matching the access-list-number can enter a specified queue.
lt	Byte-count	Packets whose length is smaller than the value of byte-count can enter a specified queue.
gt	Byte-count	Packets whose length is greater than the value of byte-count can enter a specified queue.
tcp	Port	IP packets whose source or destination TCP port number is port can enter a specified queue.
udp	Port	IP packets whose source or destination UDP port number is port can enter a specified queue.

Ensure that the fast-switching function is disabled on an interface when you specify the fragments policy for protocol rules. This is required in the current software version.

1.13.1.4 Specifying the Lowest CQ Queue Number

By default, you can customize CQ queues numbered 1 to 16. CQ reserves queue 0 for routing protocols and queue 0 functions as the absolute priority queue higher than all customize queues. If there is other traffic that needs to be processed with high priority, CQ allows you to increase the number of absolute priority queues by adjusting the lowest custom queue list number.

By default, the lowest custom queue number is 1. That is, only queue 0 is the absolute priority queue, and queues 1–16 are custom queues.

To configure the lowest CQ queue number, run the following commands in the global configuration mode:

Command	Function
Qtech(config)# queue-list <i>list-number</i> lowest-custom <i>queue-number</i>	Customize the lowest CQ queue number.

List-number: indicates the number of a queue list within 1–16.

Queue-number: indicates the number of a custom queue within 0–16.

1.13.1.5 Applying the CQ List on an Interface

To apply a CQ list on an interface, run the following command in the interface configuration mode:

Command	Function
Qtech(config-if)# custom-queue-list <i>list-number</i>	Set the queueing policy of an interface to a specified CQ list.



Note Each interface can use only one queueing policy and one matching list.

1.13.2 Monitoring CQ

To view information about input and output queues when CQ is enabled on an interface, run the following commands in the privileged user mode:

Command	Function
Qtech# show queue cq	Show configuration information about CQ queues.
Qtech# show queue interface <i>interface-name interface-number</i>	Show interface statistic information about CQ queues.
Qtech# debug qos cq	Enable CQ debugging.



Note For routers of the RSR series, you can view statistic information about fast-switching CQ queue interfaces by running the **show queue interface** command. "Qos Ref queue information" identifies statistic information about fast-switching CQ queue interfaces.

1.13.3 CQ Configuration Examples

Set CQ list 2: Assign IP packets whose length is greater than 200 bytes to queue 12.

```
Qtech(config)# queue-list 2 protocol ip 12 gt 200
```

Set CQ list 4: Assign IP packets whose length is smaller than 300 bytes to queue 2.

```
Qtech(config)# queue-list 4 protocol ip 2 lt 300
```

Set CQ list 1: Assign traffic matching IP-access-list 10 to queue 11.

```
Qtech(config)# queue-list 1 protocol ip 11 list 10
```

Set CQ list 4: Assign Telnet packets to queue 12.

```
Qtech(config)# queue-list 4 protocol ip 12 tcp 23
```

Set CQ list 4: Assign UDP domain name service packets to queue 2.

```
Qtech(config)# queue-list 4 protocol ip 2 udp 53
```

Set CQ list 3: Assign packets transmitted from ASNC port 1 to queue 7.

```
Qtech(config)# queue-list 3 interface serial 1 7
```

Set CQ list 9: Set the byte count of queue 10 to **1800**.

```
Qtech(config)# queue-list 9 queue 10 byte-count 1800
```

Set CQ list 5: Set the queue length of queue 10 to **40**.

```
Qtech(config)# queue-list 5 queue 10 limit 40
```

Set the queueing policy of SYNC port 0 to CQ list 5.

```
Qtech(config)# interface serial 0
```

```
Qtech(config-if)# custom-queue-list 1
```

1.14 Configuring PQ

1.14.1 PQ Configuration Tasks

1.14.2 Configuring PQ

You can configure a maximum of 16 groups for PQ. That is, the List-number range is 1–16. In each group, you can specify queues of packets, queue length, and byte count that can be continuously sent after a queue obtains the controlling right of sending packets.

1.14.2.1 Determining the Maximum Capacity of PQ

There are four queues (each assigned with the high, medium, normal, and low priority) in each group of queue lists. To specify the maximum number of packets allowed in each of the priority queues, run the following command in global configuration mode:

Command	Function
Qtech(config)# priority-list <i>list-number</i> queue-limit [high-limit [medium-limit [normal-limit [low-limit]]]]	Specify the maximum packet count in a PQ queue.

List-number: indicates the number of a queue list within 1–16.

The following table describes the default queue lengths.

Queue	Length
high	20
medium	40
normal	60
Low	80



Note

To configure the interface congestion management policy, ensure that the fast-switching function configuration (enable or disable the function) is consistent on all system interfaces. Otherwise, the congestion management policy becomes invalid.

1.14.2.2 Assigning Packets to PQ Queues

You can specify multiple assignment rules. The **priority-list** commands are read in order of appearance until a matching protocol or interface type is found. When a match is found, the packet is assigned to the appropriate queue and the search ends. Packets that do not match other assignment rules are assigned to the default queue. To specify which queue to place a packet in, run the following commands in global configuration mode:

Command	Function
Qtech(config)# priority-list <i>list-number</i> protocol <i>protocol-name</i> {high medium normal low} [queue-keyword] [keyword-value]	Assign a packet to a PQ queue based on the protocol type.
Qtech(config)# priority-list <i>list-number</i> interface <i>interface-type</i> <i>interface-number</i> {high medium normal low}	Assign a packet to a PQ queue based on the type of the interface where the packet enters the device.

Qtech(config)# priority-list <i>list-number</i> default { high medium normal low }	Assign a default PQ queue with the normal priority for those packets that do not match any other rule in the priority list.
--	---

List number indicates the number of a priority queue. **Protocol-name** indicates the name of a protocol, such as IP, PAD, and so on. The values and meanings of **Queue-keyword** and **Eyword-value** are the same as these of CQ.

1.14.2.3 Applying the PQ List on an Interface

To apply a PQ list on an interface, run the following command in the interface configuration mode:

Command	Function
Qtech(config-if)# priority-group <i>list-number</i>	Set the queueing policy of an interface to a specified PQ list.



Note Each interface can use only one queueing policy and one matching list.

1.14.2.4 Monitoring PQ

To view information about input and output queues when PQ is enabled on an interface, run the following commands in the privileged user mode:

Command	Function
Qtech# show queue pq	Show configuration information about PQ queues.
Qtech# show queue interface <i>interface-name</i> <i>interface-number</i>	Show interface statistic information about PQ queues.
Qtech# debug qos cq	Enable PQ debugging.



Note For routers of the RSR series, you can view statistic information about fast-switching PQ queue interfaces by running the **show queue interface** command. "Qos Ref queue information" identifies statistic information about fast-switching PQ queue interfaces.

1.14.3 PQ Configuration Examples

Set PQ list 3: Assign packets transmitted from ASNC port 1/1 to the medium priority queue.

```
Qtech(config)# priority-list 3 interface serial 1/1 medium
```

Set PQ list 6: Assign packets with a byte count greater than 250 to the medium priority queue.

```
Qtech(config)# priority-list 6 protocol ip medium gt 250
```

Set PQ list 11: Assign IP packets with a byte count smaller than 250 to the medium priority queue.

```
Qtech(config)# priority-list 11 protocol ip medium lt 250
```

Set PQ list 7: Assign packets matching IP access list 101 to the high priority queue.

```
Qtech(config)# priority-list 7 protocol ip high list 101
```

Set PQ list 6: Assign Telnet packets to the high priority queue.

```
Qtech(config)# priority-list 6 protocol ip high tcp 23
```

Set PQ list 6: Assign UDP domain name service packets to the low priority queue.

```
Qtech(config)# priority-list 6 protocol ip low udp 53
```

Set PQ list 1: Assign packets that do not match any other rules in the priority list to the medium priority queue.

```
Qtech(config)# priority-list 1 default medium
```

Set PQ list 2: set the lengths of the high, medium, normal, and low priority queues to 10, 40, 60, and 80.

```
Qtech(config)# priority-list 2 queue-limit 10 40 60 80
```

1.14.4 Configuring Hold-queue

1.14.4.1 Configuring hold-queue

Run this command to set queue length on an interface in interface configuration mode.

Command	Function
Qtech(config-if)# hold-queue <i>queue length</i> { in out }	Sets queue length on an interface.

Queue length refers to the threshold of packets in a queue. When the number of packets reaches the threshold, the coming packets will be discarded.

The default values for input queue and output queue are 75 and 40 respectively.



Caution

This command is used to modify three color-based thresholds of a queue for interface congestion and prevent green packets from being discarded preferentially. The default settings are applied generally. When cached packets exceed the red threshold, you should modify the value to make the number of cached packets below the red threshold.

1.14.4.2 Monitoring hold-queue

Run this command to show input and output queues information.

Command	Function
Qtech# show queue interface <i>interface-name interface-number</i> [<i>queue-number</i>]	Shows queue information on an interface.

1.14.4.3 hold-queue Configuration Example

Configure fair queueing on synchronous interface 0 and set the threshold of packet number in an input queue to 128. When the packet number reaches the threshold, coming packets will be discarded.

```
Qtech(config)# interface Serial 0
Qtech(config-if)# hold-queue 128 in
```

1.15 Traffic Policing and Shaping

1.16 What Are Traffic Policing and Traffic Shaping

Traffic policing is used to control the rate of classified traffic flowing across an interface.

Traffic shaping allows you to control the burst traffic going out an interface to ensure that packets are delivered at stable rates and the network traffic remains stable.



Note NPE80 does not support traffic policing and shaping.

1.17 Overview of Traffic Policing and Shaping

- About Traffic Policing

Qtech series of devices use the committed access rate (CAR) technology to limit the rate of traffic accessing the network to the committed rate. The rate-limiting feature and the packet classification feature can be configured together.

CAR uses the token bucket algorithm. You can set the capacity of the token bucket. If packets satisfy the preconfigured match rules, the token bucket accepts and processes the packets. If packets do not satisfy the preconfigured match rules, the token bucket refuses the packets and they continue to be transmitted. If there are sufficient tokens, packets processed by the token bucket continue to be transmitted. If tokens are insufficient, packets are discarded.

Qtech series of devices support the configuration of at least 1-KB flow limit, the binding of CAR and ACL, and enable you to perform rate-limiting and traffic classification at the same time.

Qtech series of devices support the service flow-limiting function based on IP applications. This enables you to limit service traffic flexibly based on IP applications' requirements on service traffic. For example, you can implement the service flow-limiting function based on the IPSec VPN tunnel.

■ About Traffic Shaping

Qtech series of devices shape traffic that is irregular or matches no predefined traffic features through the GTS (Generic Traffic Shaping) to facilitate bandwidth assignment between the upstream and downstream traffic on the network. GTS shapes traffic by buffering high-speed outbound traffic flow in the buffering area by constraining traffic to a particular bit rate using the token bucket.

Traffic shaping of routers is performed based on the following data flows:

1. All data flows passing physical interfaces
2. Data flows that are classified by using the standard or extended ACLs

■ About the Token Bucket

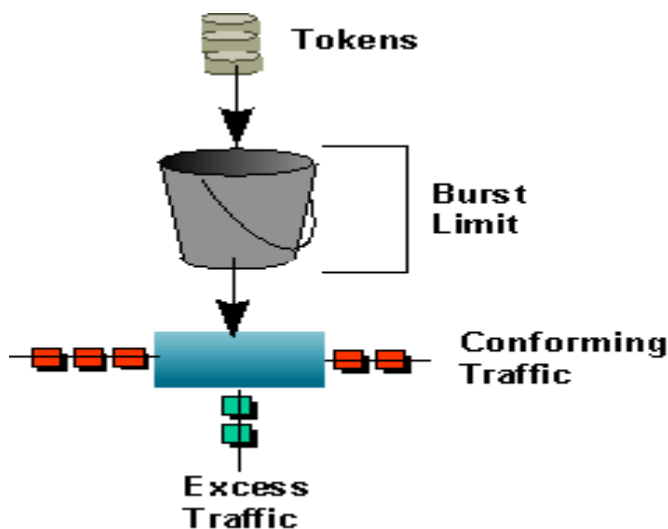
The token bucket algorithm is based on an analogy of a buffer (bucket) into which tokens, normally representing virtual data, are added at a fixed rate.

After entering the bucket, each token collects a packet from data queues and then is deleted from the bucket. This algorithm is associated with the token flow and the data flow. The following are three scenarios:

1. The data flow arrives at TBF using a rate equal to the rate of the token flow. In this case, each packet obtains a token and passes the queue without delay.
2. The data flow arrives at TBF using a rate smaller than the rate of the token flow. In this case, packets use some of the tokens and remaining tokens accumulate in the bucket until the bucket is full. Remaining tokens need to be consumed by the data flow arriving at TF using a rate greater than the rate of the token flow. In this case, burst transmission occurs.
3. The data flow arrives at TBF using a rate greater than the rate of the token flow. In this case, the bucket runs out of tokens quickly. This causes TBF interrupted for a period, namely, "threshold crossing". If packets arrive continuously, packets loss occurs.

The following figure shows the process.

Figure 7



What exactly is a token?

A token is the rate calibrated scale related to the value of CIR. Suppose that you configure an interface with a CIR rate of 8000 bit/s. Then, this interface can send 1000-bytes packets per second. At this time, the token equals to 1000 bytes, indicating that a token equal to 1000 is updated per second. Then, you can use the token to monitor whether packets accessing the interface exceed the committed rate.



Note

Ensure that the capacity of the token bucket is configured based on traffic bursts on the network. You need to expand the capacity of the token bucket to enhance the QoS tolerance capability for burst traffic on the network where burst traffic such as video or file transfer exists. Normally, the token bucket should at least support the 200-ms buffer capacity, that is, $(CIR/8)*200ms$.



Note

When GTS is used in rate-limiting, the frame gap and CRC are taken into account. The GRS rate limiting calculation is as follows:

- 1) Packet Per Second (PPS) = GTS rating limiting value in bps / ((packet length + frame gap + CRC) x 8); Round down the result for accuracy.
- 2) Receiver rate: PPS x actually received packet count in bytes x 8

1.18 Configuration Tasks of Traffic Policing

To configure the CAR traffic policing feature, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where CAR rate limiting is to be enabled.
Qtech(config-if)# rate-limit {input output} bps burst-normal burst-max conform-action action exceed-action <i>action</i>	Perform packet rate-limiting for input or output packets of all traffic accessing the interface.

Input|output: indicates the input or output traffic that you need to limit.

Bps: indicates the maximum threshold rate (in bps) that you need to set for traffic.

Burst-normal burst-max: indicates the capacity of the token bucket in bytes.

Conform-action: indicates the processing policy for traffic that conforms to the rate limit.

Exceed-action: indicates the processing policy for traffic that exceeds the rate limit.

Action: indicates a processing policy. The processing policies are described as follows:

- Proceed to match the next policy.
- **Continue:** Match the next policy.
- **Drop:** Discard the packet.
- **Set-dscp-continue:** Continue to match the packet with the next policy after the DSCP domain is set for the packet.
- **Set-dscp-transmit:** Transmit the packet after the DSCP domain is set for the packet.
- **Set-prec-continue:** Continue to match the packet with the next policy after the IP Precedence domain is set for the packet.
- **Set-prec-transmit:** Transmit the packet after the IP Precedence domain is set for the packet.
- **Transmit:** Transmit the packet.



Caution

After IPsec encryption is enabled, the inbound traffic monitoring CAR will not support the packet revising policies including **set-dscp-continue**, **set-prec-continue**, **set-dscp-transmit**, and **set-prec-transmit**,

To configure the CAR rate limit feature for different types of traffic based on ACL or DSCP values, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# access-list <i>acl-index</i>	Configure the ACL matching traffic.
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where CAR rate limiting is to be enabled.
Qtech(config-if)# rate-limit {input output} [access-group <i>acl-index</i>] <i>bps</i> <i>burst-normal burst-max conform-action action</i> <i>exceed-action action</i>	Perform packet rate-limiting for the input or output packets of the traffic that matches the Acl-index access list.
Qtech(config-if)# rate-limit {input output} [dscp <i>dscp-value</i>] <i>bps</i> <i>burst-normal burst-max conform-action action</i> <i>exceed-action action</i>	Perform packet rate-limiting for the input or output packets of the traffic that matches the DSCP code value.



Caution

If multiple ACL CARs are configured and each flow matches an ACL, all matched flows take effect. If a flow is configure with ACL1 and ACL2, ACL1 takes effect. If a flow is configured with the same ACL rules and different action, all ACL rules take effect.

To configure the service rate limiting based on IP applications, run the following commands in the global configuration mode:

Command	Function
---------	----------

Qtech(config)# flow-limit {input output} label label-num bps burst-normal burst-max conform-action action exceed-action action	Configure the global service rate limiting template.
Qtech(config)# crypto map map-name seq-num ipsec-isakmp	Enter the IPsec policy-map mode.
Qtech(config-crypto-map)# flow-label label-num	Specify the flow label of the IPsec policy-map.

Input|output: indicates the input or output traffic that you need to limit.

label-num: indicates the label number the service rate limiting feature needs to match.

Bps: indicates the maximum threshold rate (in bps) that you need to set for traffic.

Burst-normal burst-max: indicates the capacity of the token bucket in bytes.

Conform-action: indicates the processing policy for traffic that conforms to the rate limit.

Exceed-action: indicates the processing policy for traffic that exceeds the rate limit.

Action: indicates a processing policy. The processing policies are described as follows:

Drop: Drop the packet.

Transmit: Transmit the packet.

1.19 Configuration Tasks of Traffic Shaping

To configure the GTS feature, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface interface-type interface-number	Specify an interface where traffic shaping is to be enabled.
Qtech(config-if)# traffic-shape rate bit-rate [burst-size] [excess-burst-size] [buffer-limit]	Shape the entire traffic of an interface.

Bit-rate: indicates the maximum threshold rate (in bps) that you need to shape. The maximum value is **1000000000**, indicating 1 Gbps.

Burst-size: indicates the size of burst packets that can be transmitted during each interval. The unit is bit.

Excess-burst-size: indicates the size of burst packets that can be exceedingly transmitted during first interval. The unit is bit.

Buffer-limit: indicates the buffer size of a GTS buffer queue. The default value is **1000**.



Note

The system processing traffic shaping policy takes effect based on an interface. After GTS is enabled on the interface, GTS must be enabled on all subinterfaces of the interface. Otherwise, the subinterfaces transmit data unevenly.



Note

After traffic shaping is enabled on an interface, **Burst-size** must be an integral multiple of a certain value (the packet size sent in 10 ms by using the traffic shaping rate). Otherwise, the system invokes the packet configuration parameter to round off **Burst-size** to the certain value to make the parameter take effect legally.



Note Buffer queue configuration depends on network requirements. When forwarded data is delay-sensitive, you can decrease the queue depth to lower delay. When the forwarded data is burst or contains many small packets, you can increase the queue depth to improve the system buffer capability.



Note When RSR30-44 performing large MSTP nodes aggregation, GTS rate might be slightly inaccurate for the aggregation capacity of RSR30-44 is limited.

To configure the GTS feature for different types of traffic based on ACLs, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# access-list <i>acl-index</i>	Create the ACL matching traffic.
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where GTS is to be enabled.
Qtech(config-if)# traffic-shape group <i>access-list</i> <i>bit-rate</i> <i>[burst-size [excess-burst-size]]</i>	Perform interface traffic shaping for the input or output packets of the traffic that matches the <i>acl-index</i> access list.



Note The **traffic-shape group** command and the **traffic-shape rate** command are mutually exclusive on an interface. That is, you cannot configure the two commands on the same interface.



Note Ensure that the fast-switching function is disabled on the interface that deploys the traffic shaping function associated with ACL classification. This is required by in the current software version.

1.20 Configuration Tasks of Traffic Policing on a Policy-Map

To configure CAR rate limiting of a single rate on a policy-map, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# police cir [<i>bps</i>] [<i>burst-normal</i>] [<i>burst-max</i>] conform-action [<i>action</i>] exceed-action [<i>action</i>] violate-action [<i>action</i>]	Deploy the token bucket rate limiting of a single rate for this type of traffic.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Specify the policy-map to be applied.

CIR: indicates the maximum threshold rate (in bps) that you need to set for traffic.

Burst-normal burst-max: indicates the capacity of the token bucket in bytes.

Conform-action: indicates the processing policy for traffic that conforms to the rate limit.

Exceed-action: indicates the processing policy for traffic that exceeds the rate limit.

violate-action: indicates the processing policy for traffic that exceeds the rate limit set for the second token bucket when there are two token buckets.

Action: indicates a processing policy. The processing policies are described as follows:

- **Drop:** Drop the packet.
- **Set-dscp-transmit:** Transmit the packet after the DSCP domain is set for the packet.
- **Set-prec-transmit:** Transmit the packet after the IP Precedence domain is set for the packet.
- **Transmit:** Transmit the packet.

To configure CAR rate limiting of two rates on a policy-map, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# police cir [<i>bps</i>] pir [<i>bps</i>] [<i>burst-normal</i>] [<i>burst-max</i>] conform-action [<i>action</i>] exceed-action [<i>action</i>] violate-action [<i>action</i>]	Deploy the token bucket rate limiting of two rates for this type of traffic.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Specify the policy-map to be applied on the interface.

CIR: indicates the maximum threshold rate (in bps) that you need to set for traffic.

PIR: indicates the peak maximum threshold rate (in bps) that you need to set for traffic.

Burst-normal burst-max: indicates the capacity of the token bucket in bytes.

Conform-action: indicates the processing policy for traffic that conforms to the rate limit.

Exceed-action: indicates the processing policy for traffic that exceeds the rate limit.

violate-action: indicates the processing policy for traffic that exceeds the rate limit set for the second token bucket when there are two token buckets.

Action: indicates a processing policy. The processing policies are described as follows:

Drop: Drop the packet.

Set-dscp-transmit: Transmit the packet after the DSCP domain is set for the packet.

Set-prec-transmit: Transmit the packet after the IP Precedence domain is set for the packet.

Transmit: Transmit the packet.



Note

- You can choose to use one of the four token bucket algorithms for rate limit on a policy-map:
1. Single token bucket algorithm: Use this algorithm if you have not configured **violate-action** and the value of **burst-normal** is equal to the value of **burst-max**.
 2. Load mode of the single token bucket algorithm: Use this mode if you have not configured **violate-action** and the value of **burst-normal** is smaller than the value of **burst-max**.
 3. Single-rate two-token-buckets algorithm: Use this algorithm if you have configured **violate-action** but not **pir**.
 4. Two-rate two-token-bucket algorithm: Use this algorithm if you have configured both **violate-action** and **pir**.

1.21 Configuration Tasks of Traffic Shaping on a Policy-Map

To configure traffic shaping with an average rate on a policy-map, run the following commands:

Command	Function
---------	----------

Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# shape average cir [<i>bps</i>] [<i>bc</i>] [<i>be</i>]	Deploy traffic shaping with an average rate for this type of traffic.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Specify the policy-map to be applied on the interface.

Bit-rate: indicates the maximum threshold rate (in bps) that you need to set for traffic shaping.

BC: indicates the size of burst packets that can be transmitted during each interval. The unit is bit.

BE: indicates the size of burst packets that can be exceedingly transmitted during first interval. The unit is bit.

To configure traffic shaping with a peak rate on a policy-map, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# shape peak [<i>bps</i>] [<i>bc</i>] [<i>be</i>]	Deploy traffic shaping with a peak rate for this type of traffic.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Specify the policy-map to be applied on the interface.

Bit-rate: indicates the maximum threshold rate (in bps) that you need to set for traffic shaping.

BC: indicates the size of burst packets that can be transmitted during each interval. The unit is bit.

BE: indicates the size of burst packets that can be exceedingly transmitted during first interval. The unit is bit.

To configure the buffer size of traffic shaping on a policy-map, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-pmap-c)# shape max-buffers [<i>number-of-buffers</i>]	Configure the buffer size of traffic shaping.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Specify the policy-map to be applied on the interface.

Number-of-buffers: indicates the buffer size of traffic shaping. The default value is **1000**.



Note Configurations of **shape average bps** and **shape peak bps** bring different traffic shaping results. In traffic shaping, the peak rate is greater than the average rate. The calculation is as follows:
Peak rate = cir (1+ bc/be)



Note Ensure that the fast-switching function is disabled on the interface that deploys the traffic shaping function associated with a policy map. This is required by in the current software version.

1.21.1 Configuring Car Token Bucket Algorithm Mode

There are two modes for calculating Car rate-limit bandwidth: 1) frame-gap-included mode; 2) frame-gap-excluded mode. By default, frame gap is excluded.

To configure car token bucket algorithm mode, perform the following steps:

Command	Function
Qtech(config)# bandwidth-mode car link-header frame-gap	Set Car rate-limit bandwidth calculation mode: link-header means frame gap is excluded when bandwidth is calculated; frame-gap means frame gap is included when bandwidth is calculated; By default, frame gap is excluded under bandwidth calculating mode.



Note Car token bucket algorithm mode takes effect to both police car and rate-limit in the meantime.

1.22 Configuration Examples of Traffic Policing

1.22.1 Configuration Examples of Traffic Policing for Entire Traffic of an Interface

In the following examples, CAR traffic policing is configured on the ingress interface and the egress interface:

Configure CAR traffic policing of egress interface packets on a serial interface.

Limit the egress interface traffic to 300 kbps. If the traffic conforms to the rate limit, the traffic is transmitted. If the traffic exceeds the rate limit, the traffic is discarded.

```
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
rate-limit output 300000 3000 3000 conform-action transmit exceed-action drop
```

Configure CAR traffic policing of ingress interface packets on a FastEthernet interface.

Limit the ingress interface traffic to 2 Mbps. If the traffic conforms to the rate limit, the traffic is transmitted. If the traffic exceeds the rate limit, the traffic is discarded.

```
interface FastEthernet 0/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
rate-limit input 2000000 3000 3000 conform-action transmit exceed-action drop
```

1.22.2 Configuration Examples of Traffic Policing for Traffic Satisfying Certain Conditions

In the following examples, CAR traffic policing is configured for egress traffic that satisfies certain conditions:

Configure ACLs based on different TCP and UDP ports.

```
access-list 101 permit tcp any any eq 2065
access-list 102 permit udp any any range 16384 32767
access-list 103 permit tcp any any eq 1352
access-list 104 permit tcp any any eq www
access-list 105 permit tcp any any eq ftp-data
```

Configure CAR traffic policing of egress interface packets based on ACLs on a serial interface.

```
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
rate-limit output access-group 101 256000 5000 5000 conform-action transmit exceed-
action set-dscp-transmit 46
rate-limit output access-group 102 200000 3000 3000 conform-action transmit exceed-
action set-prec-transmit 5
rate-limit output access-group 103 128000 3000 3000 conform-action transmit exceed-
action set-prec-transmit 1
```

```
rate-limit output access-group 104 64000 3000 3000 conform-action transmit exceed-
action drop
rate-limit output access-group 105 32000 3000 3000 conform-action transmit exceed-
action drop
```

Configure CAR traffic policing of egress interface packets based on DSCP code on a serial interface.

Limit rates of traffic conforming to ACLs to 256 kbps, 200 kbps, 128 kbps, 64 kbps, and 32 kbps.

```
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
rate-limit output dscp 46 256000 5000 5000 conform-action transmit exceed-action
set-dscp-transmit 46
rate-limit output dscp 10 200000 3000 3000 conform-action transmit exceed-action
set-prec-transmit 5
rate-limit output dscp 18 128000 3000 3000 conform-action transmit exceed-action
set-prec-transmit 1
rate-limit output dscp 20 64000 3000 3000 conform-action transmit exceed-action drop
```



Note Do not set **token bucket** to a value that is too small. Otherwise, the system automatically adjust **token bucket** to a default value.
If you do not need to use the exceeding token bucket algorithm, you need to set the value of **Burst-normal** to be equal to or greater than the value of **burst-max**. It is normally feasible when the value of **Burst-normal** is equal to the value of **burst-max**.

1.22.3 Configuration Tasks of Traffic Policing on a Policy-Map

In the following examples, traffic policing based on policy maps is enabled for egress interface traffic satisfying conditions, and rate limit is enabled for each type of traffic using the single-rate two-token-buckets algorithm.

```
access-list 101 permit udp any any eq 100
access-list 102 permit udp any any eq 200
access-list 103 permit udp any any eq 300
access-list 104 permit udp any any eq 400
!
class-map match-all a1
match access-group 101
class-map match-all a2
match access-group 102
class-map match-all a3
match access-group 103
class-map match-all a4
match access-group 104
!
policy-map police
class a1
police cir 80000 2000 2000 conform-action transmit exceed-action drop violate-action
drop
class a2
police cir 160000 2000 2000 conform-action transmit exceed-action drop violate-action
drop
class a3
police cir 320000 6000 6000 conform-action transmit exceed-action drop violate-action
drop
class a4
police cir 640000 6000 6000 conform-action transmit exceed-action drop violate-action
drop
!
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
```

```
service-policy output police
```

In the following examples, traffic policing based on policy maps is enabled for egress interface traffic satisfying conditions, and rate limit is enabled for each type of traffic using the two-rate two-token-bucket algorithm.

```
!  
policy-map police  
class a1  
police cir 80000 pir 100000 2000 2000 conform-action transmit exceed-action drop  
violate-action drop  
class a2  
police cir 160000 pir 200000 2000 2000 conform-action transmit exceed-action drop  
violate-action drop  
class a3  
police cir 320000 pir 400000 6000 6000 conform-action transmit exceed-action drop  
violate-action drop  
class a4  
police cir 640000 pir 700000 6000 6000 conform-action transmit exceed-action drop  
violate-action drop  
!  
interface Serial1/0  
ip address 192.168.20.3 255.255.255.0  
encapsulation ppp  
service-policy output police
```

1.23 Configuration Examples of Traffic Shaping

1.23.1 Configuration Examples of Traffic Shaping for Entire Traffic of an Interface

Configure GTS traffic shaping of egress interface packets on a serial interface.

Shape the egress interface traffic to 300 kbps. If the traffic conforms to the rate limit, the traffic is transmitted. If the traffic exceeds the rate limit, the traffic is discarded. In this way, traffic is shaped.

```
interface Serial1/0  
ip address 192.168.20.3 255.255.255.0  
encapsulation ppp  
traffic-shape rate 300000 9000 9000 1000
```

1.23.2 Configuration Examples of Traffic Shaping for Traffic Satisfying Certain Conditions

In the following examples, GTS is configured for egress traffic that satisfies certain conditions:

Configure ACLs based on different TCP and UDP ports.

```
access-list 101 permit tcp any any eq 2065  
access-list 102 permit udp any any range 16384 32767  
access-list 103 permit tcp any any eq 1352  
access-list 104 permit tcp any any eq www  
access-list 105 permit tcp any any eq ftp-data
```

Configure GTS of egress interface packets based on ACLs on a serial interface.

Shape rates of traffic conforming to ACLs to 256 kbps, 200 kbps, 128 kbps, 64 kbps, and 32 kbps.

```
interface Serial1/0  
ip address 192.168.20.3 255.255.255.0  
encapsulation ppp  
traffic-shape group 101 256000 10240 10240 1000  
traffic-shape group 102 200000 8000 8000 1000  
traffic-shape group 103 128000 10240 10240 1000  
traffic-shape group 104 64000 12800 12800 1000  
traffic-shape group 105 32000 12800 12800 1000
```

**Note**

In traffic policing, tokens are updated in the token bucket every time packets enter the bucket.
 In traffic shaping, tokens are updated in the token bucket at each fixed interval.
 In traffic policing, you can configure processing policies for input and output packets at the ingress interface and egress interface.
 In traffic shaping, buffering and rate limiting can be configured only for egress interface packets.

1.23.3 Configuration Tasks of Traffic Shaping on a Policy-Map

In the following examples, traffic shaping based on a policy map is enabled for egress interface traffic satisfying conditions, and traffic shaping is enabled for each type of traffic using the common single-token-bucket algorithm.

```
access-list 101 permit udp any any eq 100
!
class-map match-all a1
match access-group 101
!
policy-map shape
class a1
shape average 100000
shape max-buffers 200 //buffer size: 200
!
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
service-policy output shape
!
```

In the following examples, traffic shaping with a peak rate based on a policy map is enabled for egress interface traffic satisfying conditions, and traffic shaping with a peak rate is enabled for each type of traffic using the common single-token-bucket algorithm.

```
access-list 101 permit udp any any eq 100
!
class-map match-all a1
match access-group 101
!
policy-map shape
class a1
shape peak 100000
shape max-buffers 200 //buffer size: 200
!
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
service-policy output shape
!
```

1.23.4 Configuration Examples of Car Token Bucket Algorithm Mode

Configure the car rate limit of out-going interface of GigabitEthernet to be 300kbps

```
Qtech(config)# int gigabitEthernet 3/1/1
Qtech(config-if-GigabitEthernet 3/1/1)#rate-limit output 300000 3000 3000 conform-
action transmit exceed-action drop
Qtech(config)# bandwidth-mode car frame-gap
```

1.24 Maintenance and Debugging of Traffic Policing and Shaping

To monitor status of traffic policing and shaping, run the following commands in the privileged user mode:

Command	Function
---------	----------

Qtech# show rate-limit interfaces [<i>interface-name</i>]	Show the rate limit of an interface.
Qtech# show traffic-shape [<i>interface-name</i>]	Show the traffic shaping configuration policy of an interface.
Qtech# show traffic-shape statistics [<i>interface-name</i>]	Show the packet statistic information about an interface in actual traffic shaping.
Qtech# show traffic-shape queue	Show the queue information of the entire traffic shaping policy that is configured.

The following are examples of **show rate-limit**:

```
Qtech#show rate-limit
serial 1/0
Output
matches access-group 101
params: 256000 bps, 3000 limit, 3000 extended limit
conformed 0 packets, 0 bytes; action: transmit 0
exceeded 0 packets, 0 bytes; action: drop 0
cbucket 0, cbs 3000; ebucket 0 ebs 0
```

The following are examples of **show traffic-shape interfaces**:

Interval indicates the interval for updating the token bucket. It is set to 30 ms in this example.

Byte Limit indicates the packet count allowed to transmit per second. It is set to 2250 bytes in this example.

```
Qtech#show traffic-shape
Interface serial 1/0
Access Target Byte Sustain Excess Interval Increment Adapt
VC List Rate Limit bits/int bits/int (ms) (bytes) Active
- - 300000 2250 9000 9000 30 1125 -
Qtech#
```

The following are examples of **show raffic-shape statistics**:

```
Qtech#show traffic-shape statistics
Interface serial 1/0
Acc. Queue Packets Bytes Packets Bytes Shaping
List Depth Rate Limit Delayed Delayed Active
- 0 0 0 0 0 no
Qtech#
```

The following are examples of **show traffic-shape queue**:

```
Qtech#show traffic-shape queue
Traffic queued in shaping queue on serial 1/0
Traffic shape group: null
Queuing strategy: weighted fair
Output queue: 0/1000/64/0 (size/max total/threshold/drops)
Output queue num: 0/0 (now/max)
Qtech#
```

To debug packet compression, run the following commands in the privileged user mode:

Command	Function
Qtech# debug tc in	Debug the traffic control for ingress interface packets.
Qtech# debug tc out	Debug the traffic control for egress interface packets.



Note For routers of the RSR series, you can view the token information about fast-switching traffic shaping interfaces by running the **show queue interface** command. "Qos Ref queue information" identifies the fast-switching statistic information. You can run the **show rate-limit interface** command to view traffic policing statistic information about fast-switching interfaces on routers of the RSR series.

1.25 Congestion Avoidance

1.26 Overview

Congestion avoidance techniques monitor network traffic loads in an effort to anticipate and avoid congestion at common network bottlenecks. Congestion avoidance is achieved through packet dropping.

Overdue congestion brings great risks to network resources, and operations must be performed to eliminate congestion. Here, congestion avoidance indicates technologies used to monitor network traffic (such as queues and memory buffer) usage in an effort to avoid network overloading by initiatively dropping packets in the event of network congestion.

1.26.1 Traditional Packet-Drop Policy – Tail-Drop

Traditional packet-drop policy indicates tail-drop. Tail drop treats all traffic equally and does not differentiate between classes of service. Queues fill during periods of congestion. When the output queue is full and tail drop is in effect, packets are dropped until the congestion is eliminated and the queue is no longer full.

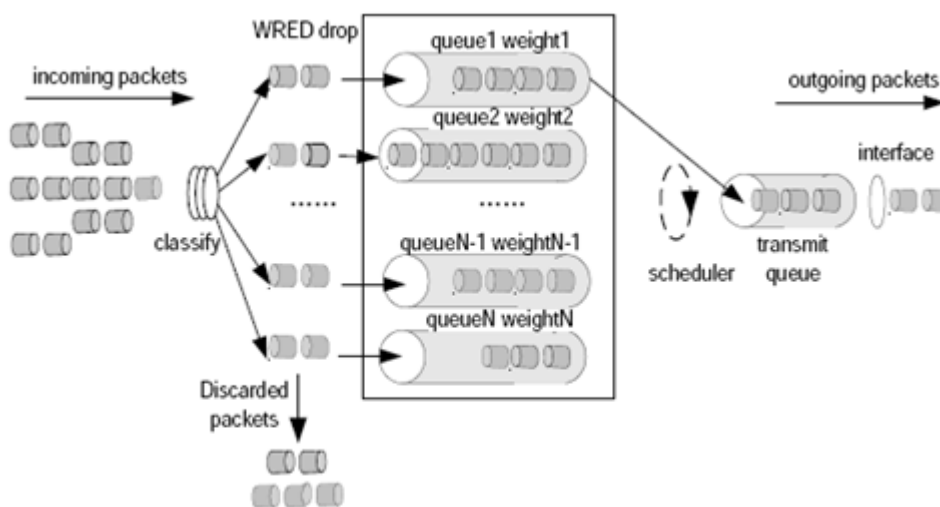
A host running TCP reduces the packet transmission rate when a lot of packets are lost and restore the packet transmission rate when the congestion is eliminated. This causes TCP global synchronization. When queues drop multiple TCP packets, hosts will reduce TCP traffic (referred to as slow start) in response, and then ramp up again simultaneously. When congestion is eased, traffic peaks occur. The two situations repeat one by one, causing cyclical periods of extreme congestion, followed by periods of under-utilization of the link.

1.26.2 WRED

WRED avoids TCP global synchronization by dropping packets randomly. When packets of a TCP connection are dropped and the transmission rate is reduced, other TCP connections still enjoy high transmission rates. In this case, there are TCP connections enjoy high transmission rates at any time, increasing the usage rate of line bandwidth.

WRED randomly drops queued packets after comparing the length, maximum threshold, and minimum threshold of a queue (configuring absolute length of the queue threshold). This is unfair for burst traffic and adverse to transmission. Therefore, the relative length of a queue is compared with the maximum threshold and minimum threshold (configuring the relative value for comparison between the queue threshold and the average length) for dropping packets. The average length of the queue is obtained by using the low-pass filter. It reflects the queue change trend but stays insensitive to the burst change of queue length, avoiding unfair treatment of burst traffic. The following figure shows the relationship between WRED and the queue mechanism.

Figure 8



Each queue is configured with a pair of maximum threshold and minimum threshold values. RED will drop packets using one of three methods:

- **No drop** – used when the queue length is smaller than the minimum threshold.

- **Drop all** – used when the queue length is greater than the maximum threshold.
- **Drop based on the WRED algorithm** – used when the queue length is between the maximum threshold and the minimum threshold.

Specifically: A random number is assigned to each packet that arrives. The random number is compared with the current mark probability denominator (MPD) of the queue. Packets are dropped if the random number is greater than the current MPD. The MPD increases in proportion to queue length, but there is a maximum MPD.

1.27 Configuration Tasks of Congestion Avoidance (WRED) Based on an Interface

To configure congestion avoidance based on precedence for an interface, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where congestion avoidance is to be enabled.
Qtech(config-if)# random-detect prec-based	Enable congestion avoidance based on precedence for the entire traffic of an interface.

To configure congestion avoidance based on DSCP classification, for an interface, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where congestion avoidance is to be enabled.
Qtech(config-if)# random-detect <i>dscp-based</i>	Enable congestion avoidance based on DSCP for the entire traffic of an interface.



Note NPE80 does not support WRED congestion avoidance based on precedence and DSCP classification. NPE80 provides only the following command for enabling and disabling WRED:
Qtech(config-if)# **random-detect**



Note To configure the interface congestion avoidance policy, ensure that the fast-switching function configuration (enable or disable the function) is consistent on all system interfaces. Otherwise, the congestion avoidance policy becomes invalid.

To configure the maximum threshold, the minimum threshold, and the MPD for each type of traffic based on DSCP classification, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where congestion avoidance is to be enabled.
Qtech(config-if)# random-detect dscp [<i>dscp-value</i>] [<i>min-threshold</i>] [<i>max-threshold</i>] [<i>mark-prob-denominator</i>]	Configure the maximum threshold, the minimum threshold, and the MPD for each type of traffic based on DSCP classification.

DSCP-value: indicates the value of DSCP. Traffic is classified based on this value.

Min-threshold: indicates the minimum drop threshold. The default values vary by traffic type.

Max-threshold: indicates the maximum drop threshold. The default values vary by traffic type.

Mark-prob-denominator: indicates the MPD that determines the number of packets that will be dropped. This is measured as a fraction, specifically 1/MPD. The greater the MPD is, the smaller the chance of each packet will be dropped. The default MPD is set to **10**, and one out of every 10 packets will be dropped. In other words, the chance of each packet being dropped is 10%.

To configure the maximum threshold, the minimum threshold, and the MPD for each type of traffic based on precedence classification, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where congestion avoidance is to be enabled.
Qtech(config-if)# random-detect precedence [<i>prec-value</i>] [<i>min-threshold</i>] [<i>max-threshold</i>] [<i>mark-prob-denominator</i>]	Configure the maximum threshold, the minimum threshold, and the drop probability for each type of traffic based on precedence classification.

Prec-value: indicates the value of Precedence. Traffic is classified based on this value.

Min-threshold: indicates the minimum drop threshold. The default values vary by traffic type.

Max-threshold: indicates the maximum drop threshold. The default values vary by traffic type.

Mark-prob-denominator: indicates the drop probability that determines the number of packets that will be dropped. This is measured as a fraction, specifically 1/MPD. The greater the MPD is, the smaller the chance of each packet will be dropped. The default MPD is set to **10**, and one out of every 10 packets will be dropped. In other words, the chance of each packet being dropped is 10%.



Note NPE80 does not support WRED congestion avoidance based on precedence and DSCP classification. You can run the following command to set the maximum threshold and the minimum threshold:
Qtech(config-if)# **random-detect** [*min-threshold*] [*max-threshold*] [*mark-prob-denominator*]

To configure the weighting factor of congestion avoidance for an interface, run the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify an interface where congestion avoidance is to be enabled.
Qtech(config-if)# random-detect exponential-weighting-constant [<i>exponential-value</i>]	Configure the weighting factor of congestion avoidance for an interface.

Exponential-value: indicates the weighting factor. The default value is **9**. The greater the value is, the smaller the MPD is.



Note If the queueing algorithm of an interface is not FIFO, you need to cancel the current queueing algorithm before configuring WRED congestion avoidance for the interface.

1.28 Configuration Tasks of Congestion Avoidance (WRED) Based on a Policy-Map



Note NPE80 does not support congestion avoidance (WRED) based on a policy-map.

To configure congestion avoidance based on precedence for a policy-map, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-if)# random-detect prec-based	Enable congestion avoidance based on precedence for the entire traffic of an interface.

To configure congestion avoidance based on DSCP classification for a policy-map, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-if)# random-detect dscp-based	Enable congestion avoidance based on DSCP for the entire traffic of an interface.

To configure the maximum threshold, the minimum threshold, and the MPD for each type of traffic based on DSCP classification, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-if)# random-detect dscp [<i>dscp-value</i>] [<i>min-threshold</i>] [<i>max-threshold</i>] [<i>mark-prob-denominator</i>]	Configure the maximum threshold, the minimum threshold, and the MPD for each type of traffic based on DSCP classification.

DSCP-value: indicates the value of DSCP. Traffic is classified based on this value.

Min-threshold: indicates the minimum drop threshold. The default values vary by traffic type.

Max-threshold: indicates the maximum drop threshold. The default values vary by traffic type.

Mark-prob-denominator: indicates the drop probability that determines the number of packets that will be dropped. This is measured as a fraction, specifically 1/MPD. The greater the MPD is, the smaller the chance of each packet will be dropped. The default MPD is set to **10**, and one out of every 10 packets will be dropped. In other words, the chance of each packet being dropped is 10%.

To configure the maximum threshold, the minimum threshold, and the MPD for each type of traffic based on precedence classification, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-if)# random-detect precedence [<i>prec-value</i>] [<i>min-threshold</i>] [<i>max-threshold</i>] [<i>mark-prob-denominator</i>]	Configure the maximum threshold, the minimum threshold, and the MPD for each type of traffic based on precedence classification.

Prec-value: indicates the value of Precedence. Traffic is classified based on this value.

Min-threshold: indicates the minimum drop threshold. The default values vary by traffic type.

Max-threshold: indicates the maximum drop threshold. The default values vary by traffic type.

Mark-prob-denominator: indicates the drop probability that determines the number of packets that will be dropped. This is measured as a fraction, specifically 1/MPD. The greater the MPD is, the smaller the chance of each packet

will be dropped. The default MPD is set to **10**, and one out of every 10 packets will be dropped. In other words, the chance of each packet being dropped is 10%.

To configure the weighting factor of congestion avoidance for a policy-map, run the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Access and create a policy-map.
Qtech(config-pmap)# class <i>class-map-name</i>	Use class-maps that have been defined.
Qtech(config-if)# random-detect exponential-weighting-constant [<i>exponential-value</i>]	Configure the weighting factor of congestion avoidance for an interface.

Exponential-value: indicates the weighting factor. The default value is **9**. The greater the value is, the smaller the drop probability is.

1.29 Configuration Examples of Congestion Avoidance (WRED)

1.29.1 Configuration Examples of Congestion Avoidance for Entire Traffic of an Interface

In the following examples, WRED congestion avoidance is configured on a SYNC interface:

- # Configure WRED congestion avoidance based on precedence classification for packets on a serial interface.
- # Set that the minimum threshold to **5**, maximum threshold to **100**, and MPD to **10** for traffic with precedence 1.
- # Set that the minimum threshold to **10**, maximum threshold to **100**, and MPD to **10** for traffic with precedence 2.
- # Set that the minimum threshold to **20**, maximum threshold to **100**, and MPD to **10** for traffic with precedence 3.
- # Set that the minimum threshold to **30**, maximum threshold to **100**, and MPD to **10** for traffic with precedence 4.

```
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
random-detect
random-detect precedence 1 5 100 10
random-detect precedence 2 10 100 10
random-detect precedence 3 20 100 10
random-detect precedence 4 30 100 10
```

1.29.2 Configuration Examples of Congestion Avoidance on a Policy-Map

In the following examples, WRED congestion avoidance is configured on a SYNC interface:

- # Configure WRED congestion avoidance based on policy-map classification for packets on a serial interface.
- # Set that the minimum threshold to **5**, maximum threshold to **100**, and MPD to **10** for traffic with DSCP af11.
- # Set that the minimum threshold to **10**, maximum threshold to **100**, and MPD to **10** for traffic with DSCP af21.
- # Set that the minimum threshold to **20**, maximum threshold to **100**, and MPD to **10** for traffic with DSCP af31.
- # Set that the minimum threshold to **30**, maximum threshold to **100**, and MPD to **10** for traffic with DSCP af41.

```
access-list 101 permit udp any any eq 100
access-list 102 permit udp any any eq 200
access-list 103 permit udp any any eq 300
access-list 104 permit udp any any eq 400
class-map match-any b1
match access-group 101
match access-group 102
class-map match-any b2
match access-group 103
match access-group 104
policy-map random
class b1
bandwidth 900
random-detect dscp-base
```

```

random-detect dscp af11 5 100 10
random-detect dscp af21 10 100 10
class b2
bandwidth 900
random-detect dscp-base
random-detect dscp af31 20 100 10
random-detect dscp af41 30 100 10
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
service-policy output random

```

1.29.3 Maintenance of Congestion Avoidance

The following section describes maintenance of congestion avoidance based on an interface:

```

Qtech# show queue interface s 1/2
Current random-detect configuration:
serial 1/2
Queuing strategy: random early detection (WRED)
Exp-weight-constant: 9 (1/512)
Mean queue depth: 81407
Avg arrive time: 3000
class      Random drop      Tail drop      Minimum Maximum  Mark
pkts/bytes  pkts/bytes  thresh  thresh  prob
0          0/0          0/0      20      40      1/10
1          336/81312    6174/1494108  5        100     1/10
2          288/69696    6168/1492656  10       100     1/10
3          238/57596    6175/1494350  20       100     1/10
4          112/27104    6321/1529682  30       100     1/10
5          0/0          0/0      31      40      1/10
6          0/0          0/0      33      40      1/10
7          0/0          0/0      35      40      1/10
Qtech#

```

The following section describes maintenance of congestion avoidance based on a policy-map:

```

Qtech# show policy-map interface s 1/2
serial 1/2  output : random1
Class b1
Current random-detect configuration:
Exp-weight-constant: 9 (1/512)
Mean queue depth: 65529
Avg arrive time: 3000
class      Random drop      Tail drop      Minimum Maximum  Mark
pkts/bytes  pkts/bytes  thresh  thresh  prob
0          0/0          0/0      20      40      1/10
1          739/178838    0/0      5        100     1/10
2          614/148588    0/0      10       100     1/10
3          0/0          0/0      26      40      1/10
4          0/0          0/0      28      40      1/10
5          0/0          0/0      31      40      1/10
6          0/0          0/0      33      40      1/10
7          0/0          0/0      35      40      1/10
Class b2
Current random-detect configuration:
Exp-weight-constant: 9 (1/512)
Mean queue depth: 65530
Avg arrive time: 3000
class      Random drop      Tail drop      Minimum Maximum  Mark
pkts/bytes  pkts/bytes  thresh  thresh  prob
0          0/0          0/0      20      40      1/10
1          0/0          0/0      22      40      1/10
2          0/0          0/0      24      40      1/10
3          394/95348    0/0      20      100     1/10
4          68/16456     0/0      30      100     1/10

```

```
5      0/0      0/0      31      40      1/10
6      0/0      0/0      33      40      1/10
7      0/0      0/0      35      40      1/10
serial 1/2 output : random1
Weighted Fair Queuing
Class b1
Output Queue: queue_num 265
Bandwidth 900 (kbps) Packets Matched 17188 Sended 5217 Max Thresh 64 (packets)
(discards/tail drops) 11971/489148 , weight 91
Class b2
Output Queue: queue_num 266
Bandwidth 900 (kbps) Packets Matched 17793 Sended 5218 Max Thresh 64 (packets)
(discards/tail drops) 12575/489148 , weight 91
Qtch#
```

1.30 Compression Protocol

1.30.1 Packet Compression Protocols

When the TCP and UDP packets carry only a small-size payload but the total amount of IP/TCP or IP/UDP/RTP header information occupies a fixed 40 bytes, the bandwidth of the low-rate link is wasted. Therefore, a series of compression algorithms are needed to compress and decompress the IP/TCP, IP/UDP, or IP/UDP/RTP header information. This is why packet compression protocols are formulated.

Packet compression mainly includes Transmission Control Protocol (TCP) packet compression and User Datagram Protocol (UDP) packet compression. Qtech products implement the compression in the following two packet formats:

- VJ TCP packet compression: It refers to the compression of TCP packets and is mainly applicable to the low-speed serial links. It complies with the standard RFC 1144 - Compressing TCP/IP Headers for Low-Speed Serial Links.
- IP Header Compression (IPHC): It is IP framework-based packet compression, which consists of three types of packet compression: IP/TCP, IP/UDP, and IP/UDP/RTP. It complies with the standard RFC 2057 - Source Directed Access Control on the Internet.

1.30.1.1 VJ TCP Packet Compression

VJ TCP packet compression refers to the compression of the TCP packets. It adopts the VJ TCP algorithm and complies with RFC 1144.

A TCP/IP data packet header occupies 40 bytes: 20 bytes for the IP header and 20 bytes for the TCP header. As the TCP and IP headers are not designed by the same organization and all of the fields in the headers are used for a specific purpose, it is impossible to simply ignore the fields for efficiency reasons.

However, TCP connections are created and each connection transmits dozens or even hundreds of packets. Then, how much of the information in each packet in one connection can remain the same? The answer is a half. The sender and receiver can track the connections, and the receiver of each connection can retain a copy of the header of the last received packet. When the sender sends a packet containing a small-size (no more than 8 bits) connection identifier, 20-byte changing content, and 20-byte unchanged content, the receiver can fill only the 20-byte unchanged content into the last saved header. In this way, the data packet is compressed into a half of the original size.

The TCP packet compression can release the bandwidth of the low-speed serial links, which is particularly effective for small packet services such as Telnet.

VJ TCP packet compression supports the following link-layer protocols:

- PPP: Point-to-Point Protocol (PPP): It needs to negotiate the compression protocol option during the IP Control Protocol (IPCP) packet negotiation.
- Serial Line Internet Protocol (SLIP)
- High-Level Data Link Control (HDLC)
- Frame Relay

1.30.1.2 IPHC

IPHC compresses all TCP and UDP packets in the IPHC format under the IP header compression framework.

At present, IPHC is mainly applied to voice data compression on a low-speed link, including the compression of the real-time transfer (RTP) data, UDP data, and TCP data.

In fact, IPHC can be used to compress the IP, UDP, and RTP headers to achieve a great success similar to that achieved by VJ TCP packet compression in TCP packets. The compression can be applied to the RTP header (in the end-to-end application), or the IP/UDP/RTP combined header (in the link-by-link application). Compressing the 40-byte combined header is more practically effective than compressing only the 12-byte RTP header, because the compressed packages in the two cases both occupy 2–4 bytes. In addition, due to the low packet delay and loss rate, the compression performance in the link-by-link application is better.

The IPHC solution is aimed to compress the IP/UDP/RTP header of most packets into 2 bytes when the UDP checksum is not sent, and into 4 bytes when the UDP checksum is sent.

The IPHC supports the following link-layer protocols:

- PPP: It needs to negotiate the compression protocol option during the IPCP packet negotiation.
- HDLC: Qtech products support the IPHC compression for the TCP and RTP packets only, and do not support the TCP/RTP packet compression proposed by Cisco.
- Frame Relay: Qtech products support the TCP/RTP packet compression with the local management interface (LMI) type set to Cisco.

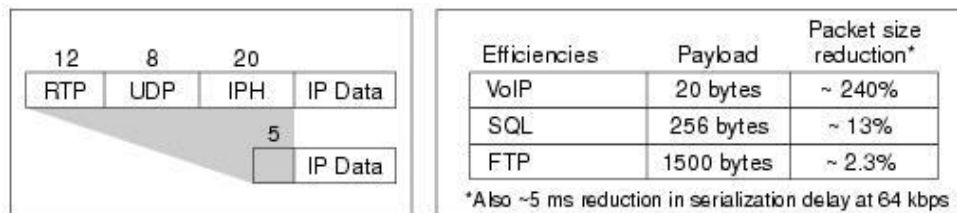
1.30.2 Efficiency and Inapplicable Scenarios of Packet Compression

The following analyzes the efficiency of the packet compression.

- For RTP packet compression, a 40-byte IP/UDP/RTP packet header is compressed to 2–4 bytes. It is assumed that the RTP payload is 24 bytes. In this case, the original 64-byte packet becomes about 27 bytes. The compression ratio is 2.3 times.
- For TCP packet compression, a 40-byte IP/TCP packet header is compressed to 2–4 bytes. It is assumed that the TCP payload is 24 bytes. In this case, the original 64-byte packet becomes about 27 bytes. The compression ratio is 2.3 times.

The following figure shows the efficiency comparison of RTP packet compression.

Figure 9



Packet compression protocols are inapplicable to Qtech products in the following scenarios:

- The Ethernet interface does not support packet compression.
- Serial links with a rate higher than 2 Mbps do not support packet compression.
- Packet compression is not supported in the case with multi-link PPP.
- Fragmented IP packets cannot be compressed.
- Packet compression cannot be configured on the logical interface of the dialer, and must be configured on the physical interface.

1.30.3 Packet Compression Protocol Configuration

To configure TCP packet compression on an interface, run the following commands in interface configuration mode:

Command	Function
Qtech(config-if)# ip tcp header-compression	Configures TCP/IP packet compression.
Qtech(config-if)# no ip tcp header-compression	Cancel the TCP/IP packet compression.

To configure the passive mode of TCP packet compression on an interface (the local packets are compressed only when the packets from the peer end are compressed), run the following commands in interface configuration mode:

Command	Function
Qtech(config-if)# ip tcp header-compression passive	Configures the passive mode of TCP/IP packet compression.
Qtech(config-if)# no ip tcp header-compression	Cancel the TCP/IP packet compression.

To configure the number of connections for TCP packet compression on an interface, run the following commands in interface configuration mode:

Command	Function
Qtech(config-if)# ip tcp compression-connections <i>number</i>	Configures the number of connections for TCP/IP packet compression.
Qtech(config-if)# no ip TCP compression-connections	Cancels the connections for TCP/IP packet compression.

1.



Note The number of connections configured on the compression interfaces at both ends must be the same. Otherwise, the TCP/IP compression module may malfunction or even fail in communication.

To configure RTP packet compression on an interface, run the following commands in interface configuration mode:

Command	Function
Qtech(config-if)# ip rtp header-compression	Configures IP/RTP packet compression.
Qtech(config-if)# no ip rtp header-compression	Cancels IP/RTP packet compression.

To configure the passive mode of RTP packet compression on the interface (the local packets are compressed only when the packets from the peer end are compressed), run the following commands in interface configuration mode:

Command	Function
Qtech(config-if)# ip rtp header-compression passive	Configures the passive mode of the IP/RTP packet compression.
Qtech(config-if)# no ip rtp header-compression	Cancels IP/RTP packet compression.

To configure the number of connections for RTP packet compression on an interface, run the following commands in interface configuration mode:

Command	Function
Qtech(config-if)# ip rtp compression-connections <i>number</i>	Configures the number of connections for IP/RTP packet compression.
Qtech(config-if)# no ip rtp compression-connections	Cancels the connections for IP/RTP packet compression.



Note The number of connections configured the compression interfaces at both ends must be the same. Otherwise, the IP/RTP compression module may malfunction or even fail in communication.



Note The compression modes configured on the compression interfaces at both ends must be the same. For example, IP/RTP compression, or TCP/IP compression, or IP/RTP compression, or TCP/IP compression is configured at both ends. In addition, the compression formats of each compression mode must be the same at both ends, that is, both compression formats are IPHC format or IETF format. If the configurations on the compression interfaces at both ends are inconsistent, the communication may fail.

1.30.4 Packet Compression Configuration Examples

1.30.4.1 PPP-based Packet Compression

The following are examples of PPP-based packet compression:

#Configure the PPP-based RTP packet compression.

```
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
ip rtp header-compression IPHC-format
```

#Configure the PPP-based TCP packet compression.

```
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
encapsulation ppp
ip tcp header-compression
```

1.30.4.2 HDLC-based Packet Compression

The following are examples of HDLC-based packet compression.

#Configure the HDLC-based RTP packet compression.

```
interface Serial1/0
ip address 192.168.20.3 255.255.255.0
ip rtp header-compression iphc-format
```



Note

Qtech products only support HDLC-based packet compression in the IPHC format. If a Qtech device interconnects with a Cisco device, packet compression in the IPHC format must also be configured on the Cisco device. If IP/RTP header compression is configured on the Cisco device, compression and decompression will fail, resulting in a communication failure.

1.30.4.3 Frame Relay-based Packet Compression

The following is an example of Frame Relay-based packet compression:

#Configure Frame Relay-based RTP and TCP packet compression.

```
interface Serial1/0
ip address 1.1.1.1 255.255.255.0
frame-relay map ip 1.1.1.2 20
frame-relay intf-type dce
frame-relay local-dlci 20
frame-relay lmi-type cisco
ip rtp header-compression
ip tcp header-compression
```



Note

Qtech products and Cisco products both support the Frame Relay-based RTP packet compression with the LMI type set to Cisco. Cisco products also allow users to enable or disable the compression function on different data links of different data link connection identifiers (DLCIs). Different from Cisco products, the policy of Qtech products is to compress all, or none.

1.30.5 Packet Compression Maintenance and Debugging

To monitor packet compression, run the following commands in privileged EXEC mode:

Command	Function
---------	----------

Qtech# show ip rtp header-compression <i>interface</i>	Displays related information about RTP packet compression and decompression.
Qtech# show ip tcp header-compression <i>interface</i>	Displays related information about TCP packet compression and decompression.
Qtech# clear rtp-compress [<i>interceface</i>]	Clears the counts of RTP packet compression and decompression.
Qtech# clear rtp-compress [<i>interceface</i>]	Clears the counts of TCP packet compression and decompression.

The following is an example of output of the **show ip rtp header-compression** command:

```
RTP/UDP/IP header compression statistics:
Interface serial 1/0: active on
Rcvd:    407 total, 406 compressed, 0 errors
0 dropped, 406 buffer copies, 0 buffer failures
Sent:    406 total, 405 compressed,
14716 bytes saved, 8494 bytes sent
2.73 efficiency improvement factor
Connect: 256 rx slots, 256 tx slots, 0 long searches, 1 misses
99% hit ratio, five minute miss rate 0 misses/sec, 0 max
```

To debug packet compression, run the following commands in privileged EXEC mode:

Command	Function
Qtech# debug ip rtp header-compression	Debugs the receiving and sending of compressed and decompressed RTP packets.
Qtech# debug ip rtp errors	Debugs RTP packet compression errors.
Qtech# debug ip tcp header-compression	Debugs the receiving and sending of compressed and decompressed TCP packets.

2 CONFIGURING MPLS QOS

2.1 MPLS QOS

MPLS QoS provided by Qtech Networks supports the following functions:

- Congestion management

Currently, MPLS experimental value based WFQ, CBWFQ, PQ and LLQ queues are supported.

- Traffic policing and traffic shaping

Currently, MPLS experimental value based car is supported to limit the bandwidth of data stream, specify actions for handling excess traffic, and limit traffic burst in respect of MPLS traffic shaping so that message flows can be transmitted at an even rate.

- Congestion avoidance

Currently, MPLS experimental value based RED and WERD are supported.

For more QoS related features, please refer to QoS configuration guideline

2.2 Congestion Management

2.3 Configuration of Weighted Fair Queuing (WFQ)

2.3.1 WFQ Configuration Tasks

When standard WFQ is used, the data packets will be classified through traffic. Qtech routers currently support MPLS EXP value based traffic classification. WFQ allocates the same bandwidth for respective traffics. The traffic-based WFQ is also called fair queuing, as all traffic are provided with the same weight.

To configure WFQ, the following tasks need to be completed:

- Configuring WFQ
- Monitoring fair queuing



Note Since the flash configurations varies from router to router, it is suggested to use "fair-queue" command to configure different queue depth and numbers.

2.3.2 Configuring WFQ

To configure WFQ, execute the following commands in interface configuration mode:

Command	Function
Qtech(config-if)# fair-queue [<i>congestive-discard-threshold</i> [<i>dynamic-queues</i>]]	Configure WFQ
Qtech(config-if)# no fair-queue	Remove WFQ configurations

WFQ parameters:

Parameter	Description
<i>congestive-discard-threshold</i>	The maximum number of packets (threshold) allowed in each queue (64 by default). When the number of packets reaches this threshold, the incoming new packets will be discarded. (This parameter is optional)

<i>dynamic-queues</i>	Number of dynamic queues. Default: 256; scope: integer between 16 and 4096, and must be the power of 2. (This parameter is optional)
-----------------------	--



Note To configure WFQ congestion management policy on the interface, all interfaces of the system must enable fast forwarding function, otherwise this function will become invalid.



Note We can find out the WFQ congestion management configurations of fast-forward IP QoS and MPLS QoS are same. The two functions may overlap on LSP Egress device, and the current software version will give preference to the WFQ function of MPLS QoS under such a circumstance.

2.3.3 Monitoring WFQ

To view WFQ configurations of interface, execute the following commands in privileged user mode.

Command	Function
Qtech# show queue interface <i>interface-name interface-number</i> <i>[queue-number]</i>	Display the QoS queue information on the designated network interface.

2.3.4 WFQ Configuration Examples

As shown below, configure fair queuing on the synchronization interface: congestion drop threshold (threshold) being 128 packets and dynamic queues being 512.

```
interface Serial 3/0
 ip ref
 ip address 192.168.200.1 255.255.255.0
 mpls ip
 fair-queue 128 512
An example of viewing interface configurations in privileged user
mode is shown below:
Qtech# show queue interface serial 3/0

Queueing strategy: weighted fair
Output queue: 0/300/128/0 (size/max total/threshold/drops)
Output queue num: 0/0/512 (now active/max active/max total)

Qos Ref queue information
Current Policy(s) : WFQ
interface cir: 2048000
Queueing strategy: weighted fair
  Dequeue threshold: Green 25000, Yellow 37500, Red 50000
  Queues: Queues total len 0, MeanBurst 800
  Queues: gts gap 7, deta bits 262, token bucket 51200
  Queues: Max 19353 pkts, used 0 pkts
  Queues: rtpQ: 0 pkts, 0 bytes
  Queues: llQ: 0 pkts, 0 bytes
  Queues: genQ: 0 pkts, 0 bytes
  wfq para: cir(2048000), delta(13/2080)
  wfq_tb para: cir(3938), delta(7/129)
  Output queue: 0/0(send/drops)
```

From the above messages, it can be observed that the queuing policy of interface adopts WFQ and the congestion drop threshold (threshold) is 128.

2.4 Configuration of Class-based Weighted Fair Queueing (CBWFQ)

2.4.1 CBWFQ Configuration Tasks

In order to configure CBWFQ, the following tasks need to be completed:

2.4.2 Configuring CBWFQ

2.4.2.1 Defining Class Maps

This function is required in order to realize CBWFQ function. The user can define network packet classifying policy in the class map, and use these class maps by specifying the name of class map in the policy map. The same class map can be used by one or more policy maps. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# class-map <i>match-all class-map-name</i>	Enter/create AND-type class map (to meet all conditions in the class map).
Qtech(config)# class-map match-any <i>class-map-name</i>	Enter/create OR-type class map (to meet only one condition in the class map).
Qtech(config-cmap)# match mpls experimental <i>value</i> or Qtech(config-cmap)# match qos-group <i>value</i> Qtech(config-cmap)# match not <i>match-type value</i>	Configure network packet classifying policy (according to the group ID of message, EXP value of MPLS message or the false condition of the abovementioned classifying policies).
Qtech(config-cmap)# exit	Exit class map configuration layer

Class-map-name: name of class map;

Match-all: To meet all conditions in the class map; the default type of class map is Match-all;

Match-any: to meet only one condition in the class map;

Mpls Experimental: the value of Experimental field of MPLS packets;

Qos-group: group ID of packets;

Not match-type: false condition of classifying policy.

2.4.2.2 Configuring Class Policy in the Policy Map

This function is required in order to realize CBWFQ function. In the policy map, the user can use all class maps configured on the device (up to 64 different class maps). The user may allocate bandwidth for the class map used, but the total bandwidth allocated for all used class maps must not exceed the bandwidth allocated to CBWFQ by the interface applied with this policy map. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# bandwidth { <i>bandwidth-kbps</i> percent <i>percent-number</i> }	Allocate bandwidth for specific class of traffic
Qtech(config-pmap-c)# queue-limit <i>number-of-packets</i>	Define queue depth

Qtech(config-pmap-c)# exit	Exit class reference configuration layer
Qtech(config-pmap)# exit	Exit policy map configuration layer

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Bandwidth-kbps: bandwidth allocated (unit: kbps);

Percent-number: percentage of bandwidth allocated (in regard to all available bandwidth of network interface);


Number-of-packets: CBWFQ queue depth (maximum number of packets allowed).

2.4.2.3 Applying Service Policy on the Designated Interface

This function is required in order to realize CBWFQ function. Applying service policy on the designated interface will enable CBWFQ function, after which the class used by the corresponding policy map will have the corresponding queue. The typical configuration of this function is shown below:

Command	Function
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Enable CBWFQ and specify the policy map to be applied.

Policy-map-name: name of policy map.

 Note	To configure policy map on the interface, all interfaces of the system must enable fast forwarding function, otherwise this function will become invalid.
---	---

2.4.2.4 Configuring Bandwidth for an Existing Class

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech (config-pmap-c)# bandwidth { <i>bandwidth-kbps</i> percent <i>percent</i> }	Allocate bandwidth for specific class of traffic

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Bandwidth-kbps: bandwidth allocated (unit: kbps);

Percent-number: percentage of bandwidth allocated (in regard to all available bandwidth of network interface);

The user may allocate bandwidth for the specific type of network traffic. By default, 1% of bandwidth is allocated to the specific type of network traffic.

2.4.2.5 Configuring the Queue Depth for an Existing Class

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.

Qtech(config-pmap-c)# queue-limit <i>number-of-packets</i>	Define queue depth
---	--------------------

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Number-of-packets: CBWFQ queue depth (maximum number of packets allowed).

The user may configure queue depth for the corresponding CBWFQ queue of specific network traffic. The default value is 64, which means that after the corresponding CBWFQ queue has 64 packets, the system will discard subsequent network packets entering into this queue. By this time, Qtech router only supports Tail-Drop congestion management instead of RED/WRED congestion management.

2.4.2.6 Configuring EXP Value of MPLS Message for an Existing Class


This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# set mpls experimental <i>exp-value</i>	Configure EXP Value of MPLS Message
Qtech(config-pmap-c)# set mpls experimental <i>dscp</i>	Set mpls experimental value to ip dscp value
Qtech(config-pmap-c)# set mpls experimental <i>precedence</i>	Set mpls experimental value to ip precedence value

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Exp-values: EXP value of message to be configured.

 Note	When MPLS experimental value configured is ip dscp, only the first three bits of dscp will be used for mapping.
--	---

2.4.2.7 Configure EXP Value of MPLS Message for an Existing Class (use table-map)

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# table-map <i>table-map-name</i>	Enter/create table-map
Qtech(config-tablemap)# map from <i>from-value to to-value</i>	Add mapping relationship into table-map
Qtech(config-tablemap)# default { <i>default-value</i> copy ignore }	Specify the action of table-map when specifying unnecessary mapping relationship for table-map
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# set mpls experimental dscp table <i>table-map-name</i>	Set mpls experimental value to ip dscp value according to the configuration of table-map

Qtech(config-pmap-c)# set mpls experimental precedence table <i>table-map-name</i>	Set mpls experimental value to ip precedence value according to the configuration of table-map
Qtech(config-pmap-c)# set mpls experimental qos-group table <i>table-map-name</i>	Set mpls experimental value to qos-group value according to the configuration of table-map

From-value: value mapped;

To-value: map value;

Default-value: default map value;

Table-map-name: name of table map;

Policy-map-name: name of policy map;

Class-map-name: name of class map;

2.4.2.8 Configuring DSCP Value of IP Message for an Existing Class


This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# set dscp <i>dscp-value</i>	Configure dscp Value of IP Message
Qtech(config-pmap-c)# set dscp experimental	Set ip dscp to mpls experimental value

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Dscp-values: dscp value of message to be configured.

 Note	When MPLS experimental value configured is ip dscp, only the first three bits of dscp will be used for mapping.
--	---

2.4.2.9 Configuring DSCP Value of IP Message for an Existing Class (use table-map)

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# table-map <i>table-map-name</i>	Enter/create table-map
Qtech(config-tablemap)# map from <i>from-value</i> to <i>to-value</i>	Add mapping relationship into table-map
Qtech(config-tablemap)# default { <i>default-value</i> copy ignore }	Specify the action of table-map when specifying unnecessary mapping relationship for table-map
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.

Qtech(config-pmap-c)# set dscp experimental table <i>table-map-name</i>	Set ip dscp to mpls experimental value according to the configuration of table-map
Qtech(config-pmap-c)# set dscp qos-group table <i>table-map-name</i>	Set ip dscp to qos-group value according to the configuration of table-map

From-value: value mapped;

To-value: map value;

Default-value: default map value;

Table-map-name: name of table map;

Policy-map-name: name of policy map;

Class-map-name: name of class map;

2.4.2.10 Configuring Precedence Value of IP Message for an Existing Class

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# set precedence <i>prec-value</i>	Configure prec Value of IP Message
Qtech(config-pmap-c)# set precedence experimental	Set ip prec to mpls experimental value

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Prec-values: precedence value of message to be configured.

2.4.2.11 Configuring Precedence Value of IP Message for an Existing Class (use table-map)

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# table-map <i>table-map-name</i>	Enter/create table-map
Qtech(config-tablemap)# map from <i>from-value to to-value</i>	Add mapping relationship into table-map
Qtech(config-tablemap)# default { <i>default-value</i> copy ignore }	Specify the action of table-map when specifying unnecessary mapping relationship for table-map
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# set precedence experimental table <i>table-map-name</i>	Set ip prec to mpls experimental value according to the configuration of table-map
Qtech(config-pmap-c)# set precedence qos-group table <i>table-map-name</i>	Set ip prec to qos-group value according to the configuration of table-map

From-value: value mapped;

To-value: map value;

Default-value: default map value;

Table-map-name: name of table map;

Policy-map-name: name of policy map;

Class-map-name: name of class map;

2.4.2.12 Configuring Group ID of Message for an Existing Class

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# set qos-group <i>group-value</i>	Set the group ID of message
Qtech(config-pmap-c)# set qos-group dscp	Set group ID of message to ip dscp value.
Qtech(config-pmap-c)# set qos-group precedence	Set group ID of message to ip precedence value.
Qtech(config-pmap-c)# set qos-group mpls experimental	Set group ID of message to mpls experimental value.
Qtech(config-pmap-c)# set qos-group cos	Set group ID of message to cos value.

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Group-values: group ID of message to be configured.

2.4.2.13 Configuring Group ID of Message for an Existing Class (use table-map)

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config)# table-map <i>table-map-name</i>	Enter/create table-map
Qtech(config-tablemap)# map from <i>from-value</i> to <i>to-value</i>	Add mapping relationship into table-map
Qtech(config-tablemap)# default { <i>default-value</i> copy ignore }	Specify the action of table-map when specifying unnecessary mapping relationship for table-map
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# set qos-group dscp table <i>table-map-name</i>	Set group ID of message to ip dscp value according to the configuration of table-map
Qtech(config-pmap-c)# set qos-group precedence table <i>table-map-name</i>	Set group ID of message to ip precedence value according to the configuration of table-map

Qtech(config-pmap-c)# set qos-group mpls experimental table <i>table-map-name</i>	Set group ID of message to mpls experimental value according to the configuration of table-map
Qtech(config-pmap-c)# set qos-group cos table <i>table-map-name</i>	Set group ID of message to cos value according to the configuration of table-map

From-value: value mapped;

To-value: map value;

Default-value: default map value;

Table-map-name: name of table map;

Policy-map-name: name of policy map;

Class-map-name: name of class map;

2.4.2.14 Configuring the Bandwidth Allocated to CBWFQ

This function is optional for CBWFQ. The typical configurations of this function are shown below:

Command	Function
Qtech(config-if)# max-reserved-bandwidth <i>percent</i>	Configure the Bandwidth Allocated to CBWFQ

Percent: Percentage of bandwidth allocated to CBWFQ on the existing network interface.

The user may allocate the available bandwidth percentage allocated to CBWFQ. The default value is 75, which means 75% of gross available bandwidth of network interface will be allocated to CBWFQ.

2.4.3 Monitoring CBWFQ

When CBWFQ becomes valid on a specific interface, to display input and output queues, the user can execute the following commands in privileged user mode:

Command	Function
Qtech# show class-map	Display all class maps
Qtech# show class-map <i>class-map-name</i>	Display the information of a specific class map
Qtech# show policy-map	Display all policy maps
Qtech# show policy-map name <i>policy-map-name</i>	Display the information of a specific policy map
Qtech# show policy-map name <i>policy-map-name class class-name</i>	Display a specific class map in a specific policy map
Qtech# show policy-map interface <i>Interface-name interface-number</i>	Display the policy map applied on a specific network interface
Qtech# show queue <i>interface-name interface-number</i>	Display the QoS queue information on the designated network interface.

Class-map-name: name of class map;

Policy-map-name: name of policy map;

Interface-name: name of network interface;

Interface-number: network interface ID.

2.4.4 CBWFQ Configuration Examples

The following example shows how to configure MPLS Experimental based CBWFQ congestion management policy on the synchronization interface:

```
class-map 101
match mpls experimental 1
class-map 102
match mpls experimental 2
class-map 103
match mpls experimental 3
!
policy-map 1
class 101
bandwidth 600
class 102
bandwidth 400
class 103
bandwidth 200
!
interface Serial 3/0
ip ref
ip address 192.168.200.1 255.255.255.0
mpls ip
service-policy output 1
```

An example of viewing interface configurations in privileged user mode is shown below:

```
Qtech# show queue interface serial 3/0

Queueing strategy: cb weighted fair
Output queue: 0/300/128/0 (size/max total/threshold/drops)
cb queue_num 0/0 (active/max active)
wfq queue_num 0/0 (active/max active)
Reserved queue_num 3/3 (allocated/max allocated)
Llq is close

Qos Ref queue information
Current Policy(s) : CBWFQ
interface cir: 2048000
Queueing strategy: cb weighted fair
Dequeue threshold: Green 25000, Yellow 37500, Red 50000
Queues: Queues total len 0, MeanBurst 800
Queues: gts gap 7, deta bits 262, token bucket 51200
Queues: Max 19353 pkts, used 0 pkts
Queues: rtpQ: 0 pkts, 0 bytes
Queues: llQ: 0 pkts, 0 bytes
Queues: genQ: 0 pkts, 0 bytes
Output Stat.: 0/0/ (send/drops)
queue_num 128/256 (cb num/wfq num)
cb packet Stat. 0/0 (send /drop)
wfq packet Stat. 0/0 (send /drop)
Reserved queue_num 3/3 (allocated/max allocated)
Llq is close
```

2.5 Configuration of Custom Queueing (CQ)

2.5.1 CQ Configuration Tasks

Custom Queue configuration tasks are shown below:

2.5.2 Configuring CQ

CQ can configure up to 16 groups, namely the scope of List-number is 1-16. Each group specifies the type of queues that can be entered by different kinds of packets, queue length, and byte count that is allowed to be sent.

2.5.2.1 Determining the Maximum Capacity of Queue Adopting CQ

To configure the maximum packet capacity for each queue, execute the following commands in global configuration mode:

Command	Function
Qtech(config)# queue-list <i>list-number</i> queue <i>queue-number</i> limit <i>limit-number</i>	Specify the maximum number of packets allowed by each custom queue. The no form of this command can be used to restore the queue length to the default value of 20.
Qtech(config)# queue-list <i>list-number</i> queue <i>queue-number</i> byte-count <i>byte-count-number</i>	Specify the number of bytes allowed by each queue. The no form of this command can be used to restore the byte count to the default value of 1500.

List-number: number of queue list (any number between 1-16);

Queue-number: queue number (any number between 1-16);

Limit-number: The maximum number of packets allowed by the queue, within the range being from 1 to 32767. The default value is 20.

Byte-count-number: Specify how many bytes of data should be delivered from the current queue by the system before the system moves on to the next queue. When a particular queue is being processed, packets are sent until the number of bytes sent exceeds the byte-count-number configured (within the range of 1 to 16777215, 1500 by default) or until the queue is empty. For how to determine the best byte count of data to be sent, please refer to the foregoing chapters.



Note To configure CQ congestion management policy on the interface, all interfaces of the system must disable the fast forwarding function, which is not supported by CQ congestion management policy.

2.5.2.2 Assigning Packets to CQ

You can assign the packets to custom queues based on the protocol type or the interface where the packets enter the device. Additionally, you can set the default queue for the packets that do not match other assignment rules. You can also specify multiple rules.

To define the CQ lists, use the following commands in global configuration mode:


Command	Function
Qtech(config)# queue-list <i>list-number</i> protocol mpls <i>queue-number</i> [experimental <i>exp-value</i>]	Assign the packets to the specified custom queue based on the protocol type.

Therein, exp-value refers to MPLS experimental value.

2.5.2.3 Applying CQ List on the Interface

To apply a CQ list to an interface, use the following command in interface configuration mode:

Command	Function
Qtech(config-if)# custom-queue-list <i>list-number</i>	Set the queuing policy of this interface to a specific CQ list.

 Note	You can only specify one queuing policy for each interface, and only one queue list can be specified in the mean time.
--	--

2.5.3 Monitoring CQ

To display information about input and output queues when CQ is enabled on an interface, use the following commands in privileged user mode:

Command	Function
Qtech# show queue cq	Display information about CQ.
Qtech# show queue interface <i>interface-name interface-number</i> [<i>queue-number</i>]	Display information about CQ interface.
Qtech# debug qos cq	Turn on the CQ debug switch if the priority queue is configured.

2.5.4 CQ Configuration Examples

Configure custom list 2, and assign packets with MPLS EXP value being 2 to queue number 12 :

```
Qtech(config)# queue-list 2 protocol mpls 12 experimental 2
```

Configure custom list 1, and assign packets with MPLS EXP value being 5 to queue number 11:

```
Qtech(config)# queue-list 1 protocol mpls 11 experimental 5
```

Apply custom list 1 configured previously to the synchronization interface:

```
Qtech(config)# interface serial 0
Qtech(config-if)# custom-queue-list 1
```

2.6 Configuration of Priority Queueing (PQ)

2.6.1 PQ Configuration Tasks

Priority Queue configuration tasks are shown below:

2.6.2 Configuring PQ

PQ can configure up to 16 groups, namely the scope of List-number is 1-16. Each group specifies the type of queues that can be entered by different kinds of packets, as well as the maximum number of packets allowed by each queue.

2.6.2.1 Determining the Maximum Capacity of Queue Adopting PQ

In the queue list of each group, there are four queues divided into high, medium, normal and low. The user may configure the maximum packet capacity of each queue. Execute the following commands in global configuration mode:

Command	Function
Qtech(config)# priority-list <i>list-number</i> queue-limit <i>high-limit</i> <i>medium-limit normal-limit low-limit</i>	Specify the maximum number of packets allowed by each priority queue.

List-number: number of queue list (any number between 1-16);

The default length of priority queue is shown below:

Queue	Length
-------	--------

high	20
medium	40
normal	60
Low	80



Note To configure PQ congestion management policy on the interface, all interfaces of the system must enable fast forwarding function, otherwise this function will become invalid.

2.6.2.2 Assigning Packets to PQ

The system can specify multiple assignment rules. This list will be searched according to the sequence specified by priority-list until a matching protocol or interface type is found. When a matching entry is found, this packet will be allocated to the corresponding queue and the search will end. Packets failing to match other assignment rules can be allocated to the default queue. To specify the queue for assigning packets, execute the following command in global configuration mode:

Command	Function
Qtech(config)# priority-list <i>list-number</i> protocol mpls { high medium normal low } [experiental <i>exp-value</i>]	Assign packets to a specific PQ according to the EXP value of MPLS message.
Qtech(config)# priority-list <i>list-number</i> default { high medium normal low }	Assign packets matching no rules to the default PQ (normal).

Therein, List-number is the group ID of PQ, and exp-value refers to MPLS experimental value.

2.6.2.3 Applying PQ list on the Interface

To apply a PQ list to an interface, execute the following command in interface configuration mode:

Command	Function
Qtech(config-if)# priority-group <i>list-number</i>	Set the queuing policy of this interface to a specific PQ list.



Note You can only specify one queuing policy for each interface, and only one queue list can be specified in the mean time.

2.6.3 Monitoring PQ

When PQ becomes valid on a specific interface, to display input and output queues, the user can execute the following commands in privileged user mode:

Command	Function
Qtech# show queue interface <i>interface-name</i> <i>interface-number</i> [<i>queue-number</i>]	Display the PQ information on the designated network interface.

2.6.4 PQ Configuration Examples

Configure priority list 1, and allocate packets with MPLS EXP value being 7 to the medium PQ:
`priority-list 1 mpls experimental 7 medium`

Configure priority list 1, and allocate packets received by synchronization serial port to the medium PQ:
`priority-list 1 interface serial 1/1 medium`

Configure priority list 1, and allocate packets matching no rules in the priority list to the medium PQ:
`priority-list 1 default medium`

Configure priority list 1, and configure the length of high, medium, normal and low PQs to 10, 40, 60 and 80 respectively:

```
priority-list 1 queue-limit 10 40 60 80
```

Apply priority list 1 configured previously to the synchronization interface:
`interface Serial 3/0`

```
ip ref
ip address 192.168.200.1 255.255.255.0
mpls ip
priority-group 1
```

An example of viewing interface configurations in privileged user mode is shown below:

```
Qtech# show queue interface serial 3/0

Queueing strategy: priority-list 1
Output queues: (queue #: size/max/send/drops)
Output queue: high 0/20/0/0, medium 0/40/0/0, normal 0/60/0/0, low
0/80/0/0

Qos Ref queue information
Current Policy(s) : PQ
Queueing strategy: priority-list 1
interface cir: 2048000
Dequeue threshold: Green 25000, Yellow 37500, Red 50000
Queues: Queues total len 0, MeanBurst 800
Queues: gts gap 7, deta bits 262, token bucket 51200
Queues: Max 19353 pkts, used 0 pkts
Queues: rtpQ: 0 pkts, 0 bytes
Queues: llQ: 0 pkts, 0 bytes
Queues: genQ: 0 pkts, 0 bytes
Threshold: MeanBurst 800, Priority LOW, Dec 4560, Inc 4560, Drop
16720
Counter: PriInc 0, PriDec 0, Drop 0
Queues: Queues len 0, MeanBurst 800, gts token bucket 51200
Queues: Max 19353 pkts, used 0 pkts, rtpQ: 0 pkts, 0 bytes. genQ:
0 pkts, 0 bytes
(size/max/send/drops)
high 0/0/0/0, medium 0/0/0/0, normal 0/0/0/0, low 0/0/0/0
```

2.7 Configuration of Low Latency Queuing (LLQ)

2.7.1 LLQ Configuration Tasks

LLQ configuration is performed jointly with CBWFQ configuration. To configure LLQ, the following tasks need to be done:

2.7.2 Configuring LLQ

To configure LLQ, execute the following commands in Policy-map command layer configuration mode:

Command	Function
---------	----------

Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech (config-pmap-c)# priority { <i>bandwidth-kbps</i> percent <i>percent</i> } [<i>Burst bytes</i>]	Allocate bandwidth for specific class of traffic
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Enable CBWFQ and specify the policy map to be applied.

Policy-map-name: name of policy map;

Class-map-name: name of class map;

Bandwidth-kbps: bandwidth allocated (unit: kbps);

Percent: percentage of bandwidth allocated (in regard to all available bandwidth of network interface);

The user may allocate bandwidth for the specific type of network traffic. By default, 1% of bandwidth is allocated to the specific type of network traffic.

Burst bytes: number of bytes allowed in a burst above the committed rate limit.



Note Generally, the bandwidth allocation on the interface will be influenced by the following commands: Bandwidth (CBWFQ), Priority (LLQ), ip rtp priority (RTPQ), and max-reserved-bandwidth (interface). The total bandwidth depends on the "Bandwidth" command of interface. I.e., Bandwidth = max-reserved-bandwidth + default wfq bandwidth; max-reserved-bandwidth = bandwidth(policy-map) + priority(policy-map) + ip RTP priority



Note To configure PQ congestion management policy on the interface, all interfaces of the system must enable fast forwarding function, otherwise this function will become invalid.

2.7.3 Monitoring LLQ

To view LLQ configurations of interface, execute the following command in privileged user mode.

Command	Function
Qtech# show policy-map <i>interface-name interface-number</i>	Display the interface information of LLQ.

2.7.4 LLQ Configuration Examples

The following example shows how to configure a LLQ on the synchronization interface to serve packets with MPLS EXP value being 7:

```
class-map match-all 201
  match mpls experimental 7
!
policy-map 1
  class 201
    priority 30 2000
!
interface Serial 3/0
  ip ref
  ip address 192.168.200.1 255.255.255.0
  mpls ip
```

```
service-policy output 1
```

An example of viewing interface configurations in privileged user mode is shown below:

```
Qtech#show queue interface serial 3/0

Queueing strategy: cb weighted fair
Output queue: 0/300/128/0 (size/max total/threshold/drops)
cb queue_num 0/0 (active/max active)
wfq queue_num 0/0 (active/max active)
Reserved queue_num 1/1 (allocated/max allocated)
Llq is open

Qos Ref queue information
Current Policy(s) : CBWFQ
interface cir: 2048000
Queueing strategy: cb weighted fair
Dequeue threshold: Green 25000, Yellow 37500, Red 50000
Queues: Queues total len 0, MeanBurst 800
Queues: gts gap 7, deta bits 262, token bucket 51200
Queues: Max 19353 pkts, used 0 pkts
Queues: rtpQ: 0 pkts, 0 bytes
Queues: llQ: 0 pkts, 0 bytes
Queues: genQ: 0 pkts, 0 bytes
Output Stat.: 0/0/ (send/drops)
queue_num 128/256 (cb num/wfq num)
cb packet Stat. 0/0 (send /drop)
wfq packet Stat. 0/0 (send /drop)
Reserved queue_num 1/1 (allocated/max allocated)
Llq is open
```

2.8 Traffic Policing and Traffic Shaping

2.9 Introduction to Traffic Policing and Traffic Shaping

In traffic policing, certain actions will be taken to limit the data rate of classified traffics entering the network.

Traffic shaping will restrict the burst of traffics, so that message flows can be transmitted at an even rate and the network traffic can maintain stable.

2.10 Traffic Policing Configuration Tasks

To configure Car traffic shaping on the interface, execute the following commands in interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify the interface for Car traffic policing.
Qtech(config-if)# rate-limit {input output} bps <i>burst-normal burst-max conform-action action</i> exceed-action <i>action</i>	Applying rate limiting on all traffics entering the receiving interface or outgoing interface.

Input|output: The input or output data rate to be limited by the user.

Bps: Maximum data rate of the traffic desired by the user (unit: bps).

Burst-normal burst-max: Size of token bucket (unit: bytes).

Conform-action: Traffic handling policy under rate limitation.

Exceed-action: Data rate handling policy when rate limit is exceeded.

Action: Handling policy, including:

Continue to match the next policy

- Drop: drop the packet
- Set-mpls-exp-transmit: transmit this packet after setting mpls experimental field
- Set-mpls-exp-continue: after setting mpls experimental field, this packet continue to match the next policy
- Transmit: transmit this packet



Note To configure traffic policing on the interface, all interfaces of the system must enable fast forwarding function, otherwise this function will become invalid.

2.11 Traffic Shaping Configuration Tasks

To configure GTS traffic shaping on the interface, execute the following commands in the interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type</i> <i>interface-number</i>	Specify the interface for traffic shaping.
Qtech(config-if)# traffic-shape rate <i>bit-rate</i> [<i>burst-size</i>] [<i>excess-burst-size</i>] [<i>buffer-limit</i>]	Carry out traffic shaping of all traffics on the interface.

Bit-rate: the maximum data rate to be shaped by the user (unit: bps).

Burst-size: the maximum traffic size that is permitted in each burst at each interval (unit: bit)

Excess-size: transient burst of traffic that the first interval can forward (unit: bit)

Buffer-limit: buffer size of gts buffer queue (default: 1000).



Note To configure traffic shaping on the interface, all interfaces of the system must enable fast forwarding function, otherwise this function will become invalid.



Note The traffic shaping policy handled by the system will function on the interface. When GTS has been configured for the interface, all related sub-interfaces of this interface must enable GTS, otherwise the traffic forwarding will become uneven on related sub-interfaces.



Note After traffic shaping is enabled on the interface, the burst traffic must be the integral multiple of the data transmitted at 10ms under traffic-shaping rate, otherwise the system will round off the burst traffic configuration parameters according to the data transmitted at 10ms under traffic-shaping rate, so that the parameters can become valid.

2.12 Configuring Traffic Policing under Policy-map

To configure single-rate Car traffic limiting in Policy-map, execute the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map

Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# police <i>cir</i> <i>bps burst-normal burst-max</i> conform-action <i>action</i> exceed- action <i>action</i> violate-action <i>action</i>	Perform single-rate token bucket limiting of such traffics.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Specify the policy map to be applied to the interface.

CIR: Maximum data rate of the traffic desired by the user (unit: CIR).

Burst-normal burst-max: Size of token bucket (unit: bytes).

Conform-action: Traffic handling policy under rate limitation.

Exceed-action: Data rate handling policy when rate limit is exceeded.

Violate-action: Traffic handling policy when the second token bucket rate limit is exceeded in the case of two token bucket system.

Action: Handling policy, including:

- Drop: drop the packet
- Set-mpls-exp-transmit: transmit this packet after setting mpls experimental field
- Transmit: transmit this packet

To configure dual-rate Car traffic limiting in Policy-map, execute the following commands:

Command	Function
Qtech(config)# policy-map <i>policy-map-name</i>	Enter/create policy map
Qtech(config-pmap)# class <i>class-map-name</i>	Use the class map defined.
Qtech(config-pmap-c)# police <i>cir</i> <i>bps pir bps</i> <i>burst-normal burst-max</i> conform- action <i>action</i> exceed-action <i>action</i> violate-action <i>action</i>	Perform dual-rate token bucket limiting of such traffics.
Qtech(config-if)# service-policy output <i>policy-map-name</i>	Specify the policy map to be applied to the interface.

CIR: Maximum data rate of the traffic desired by the user (unit: CIR).

PIR: Peak data rate of the traffic desired by the user (unit: CIR).

Burst-normal burst-max: Size of token bucket (unit: bytes).

Conform-action: Traffic handling policy under rate limitation.

Exceed-action: Data rate handling policy when rate limit is exceeded.

Violate-action: Traffic handling policy: when the second token bucket rate limit is exceeded in the case of two token bucket system.

Action: Handling policy, including:

- Drop: drop the packet
- Set-mpls-exp-transmit: transmit this packet after setting mpls experimental field
- Transmit: transmit this packet



Note There are four token bucket algorithms for rate limiting under policy-map. The user may select different token bucket algorithm according to different configurations.



Note Single token bucket algorithm: If violate-action is not configured and the value of burst-normal equals to the value of burst-max, the single token bucket algorithm is adopted.



Note Borrowing mode of single token bucket algorithm: If violate-action is not configured and the value of burst-normal is smaller than the value of burst-max, the borrowing mode of single bucket algorithm is adopted.



Note Single-rate token bucket algorithm: If violate-action is configured but pir is not configured, the single-rate dual token bucket algorithm is adopted.



Note Dual-rate token bucket algorithm: If both violate-action and pir are configured, the dual-rate dual token bucket algorithm is adopted.

2.13 Traffic Policing Configuration Examples

2.13.1 Example of Applying Traffic Policing on All Traffics on the Interface

Configure car traffic policing of incoming messages on Serial interface

Limit traffics on the receiving interface at 2Mbps; transmit conforming traffics after setting mpls experimental value to 2 and drop excess traffics.

```
interface Serial 3/0
ip ref
ip address 192.168.20.3 255.255.255.0
mpls ip
rate-limit input 2000000 3000 3000 conform-action set-mpls-exp-
transmit 2 exceed-action drop
```

An example of viewing interface configurations in privileged user mode is shown below:

```
Qtech#show rate-limit interface serial 3/0
Serial 3/0
Input
matches all traffic
params: 2000000 bps, 3000 limit, 3000 extended limit
conformed 0 packets, 0 bytes; action: set mpls transmit
exceeded 0 packets, 0 bytes; action: drop
cbucket 6000, cbs 6000; ebucket 0 ebs 0
```

2.13.2 Example of Traffic Policing Configuration under Policy-map

The following example shows how to apply Policy-map based traffic limiting on conforming traffics on the outgoing interface. Single-rate dual token bucket algorithm is applied to limit each kind of traffic.

```
!
class-map match-all a1
match mpls experimental 1
class-map match-all a2
match mpls experimental 2
class-map match-all a3
match mpls experimental 3
class-map match-all a4
match mpls experimental 4
!
policy-map police
```

```

class a1
police cir 80000 2000 2000 conform-action transmit exceed-action
  drop violate-action drop
class a2
police cir 160000 2000 2000 conform-action transmit exceed-action
  drop violate-action drop
class a3
police cir 320000 6000 6000 conform-action transmit exceed-action
  drop violate-action drop
class a4
police cir 640000 6000 6000 conform-action transmit exceed-action
  drop violate-action drop
!
interface Serial 3/0
 ip ref
 ip address 192.168.20.3 255.255.255.0
 mpls ip
 service-policy output police

```

View interface configurations in privileged user mode:

```

Qtech#show policy-map interface serial 3/0

Serial 3/0  output(tc policy): police
  Class a1
    current token tbf: TC_SRTMC
    params: 80000 bps, 2000 limit, 2000 extended limit , 0 pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 2000, cbs 2000; ebucket 2000 ebs 2000
  Class a2
    current token tbf: TC_SRTMC
    params: 160000 bps, 2000 limit, 2000 extended limit , 0 pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 2000, cbs 2000; ebucket 2000 ebs 2000
  Class a3
    current token tbf: TC_SRTMC
    params: 320000 bps, 6000 limit, 6000 extended limit , 0 pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 6000, cbs 6000; ebucket 6000 ebs 6000
  Class a4
    current token tbf: TC_SRTMC
    params: 640000 bps, 6000 limit, 6000 extended limit , 0 pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 6000, cbs 6000; ebucket 6000 ebs 6000

```

The following example shows how to apply Policy-map based traffic limiting on conforming traffics on the outgoing interface. Dual-rate dual token bucket algorithm is applied to limit each kind of traffic.

```

!
policy-map police
class a1
police cir 80000 pir 100000 2000 2000 conform-action transmit
  exceed-action drop violate-action drop
class a2
police cir 160000 pir 200000 2000 2000 conform-action transmit
  exceed-action drop violate-action drop
class a3

```



```

police cir 320000 pir 400000 6000 6000 conform-action transmit
  exceed-action drop violate-action drop
class a4
police cir 640000 pir 700000 6000 6000 conform-action transmit
  exceed-action drop violate-action drop
!
interface Serial 3/0
 ip ref
 ip address 192.168.20.3 255.255.255.0
 mpls ip
 service-policy output police

```

View interface configurations in privileged user mode:

```

Qtech#show policy-map interface serial 3/0

Serial 3/0  output(tc policy): police
  Class a1
    current token tbf: TC_TRTMC
    params: 80000 bps, 2000 limit, 2000 extended limit , 100000
  pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 2000, cbs 2000; ebucket 2000 ebs 2000
  Class a2
    current token tbf: TC_TRTMC
    params: 160000 bps, 2000 limit, 2000 extended limit , 200000
  pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 2000, cbs 2000; ebucket 2000 ebs 2000
  Class a3
    current token tbf: TC_TRTMC
    params: 320000 bps, 6000 limit, 6000 extended limit , 400000
  pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 6000, cbs 6000; ebucket 6000 ebs 6000
  Class a4
    current token tbf: TC_TRTMC
    params: 640000 bps, 6000 limit, 6000 extended limit , 700000
  pir
    conformed 0 packets, 0 bytes; action: transmit 0
    exceeded 0 packets, 0 bytes; action: drop 0
    violated 0 packets, 0 bytes; action: drop 0
    cbucket 6000, cbs 6000; ebucket 6000 ebs 6000

```

2.14 Traffic Shaping Configuration Examples

2.14.1 Example of Applying Traffic Shaping on All Traffics on the Interface

Configure GTS traffic shaping of outgoing messages on Serial interface

Shape traffics on the outgoing interface at 300kbps; transmit conforming traffics and put excess traffics in the buffer queue for later transmission.

```

interface Serial 3/0
 ip ref
 ip address 192.168.20.3 255.255.255.0
 mpls ip
 traffic-shape rate 2000000 40000 40000 1000
# View interface configurations in privileged user mode:

```

```
Qtech#show traffic-shape serial 3/0
Interface Serial 3/0
      Access Target   Byte      Sustain   Excess   Interval
Increment Adapt
VC      List   Rate     Limit     bits/int  bits/int  (ms)
-       -     2000000  10000    40000    40000    20     5000
-       -
```

2.15 Congestion Avoidance

2.16 Introduction to Congestion Avoidance

Congestion avoidance is deployed at the network bottle neck to effectively monitor network traffics and avoid anticipated congestion developed at the network bottle neck by dropping information packets.

Excess congestion will cause great harms to network resources, and certain measures shall be taken accordingly. The Congestion Avoidance as mentioned herein refers to the mechanism of actively dropping packets when congestion is expected by monitoring how the network resources are utilized (such as queues or memory buffers), as so to alleviate the load on the network.

2.16.1 WRED

WRED avoids global TCP synchronization by randomly dropping packets. Thus, while the sending rates of some TCP sessions slow down after their packets are dropped, other TCP sessions remain at high sending rates. As there are always TCP sessions at high sending rates, link bandwidth is efficiently utilized.

Before dropping packets, WRED will compare the queue size with lower threshold and upper threshold (the absolute length of queue threshold), and this will result in the unequal treatment of burst traffics and compromise traffic transmission. Therefore, when dropping packets by comparing between lower threshold and upper threshold, the average size of queue will be adopted (this should be the relative value upon comparison between queue threshold and average size). The average queue size is the result of low pass filtering of queue size, and avoids the unequal treatment in the burst of queue size as it reflects the change tendency of queue and is not sensitive to the changes in queue size. The relationship between WRED and queuing mechanism is shown below:

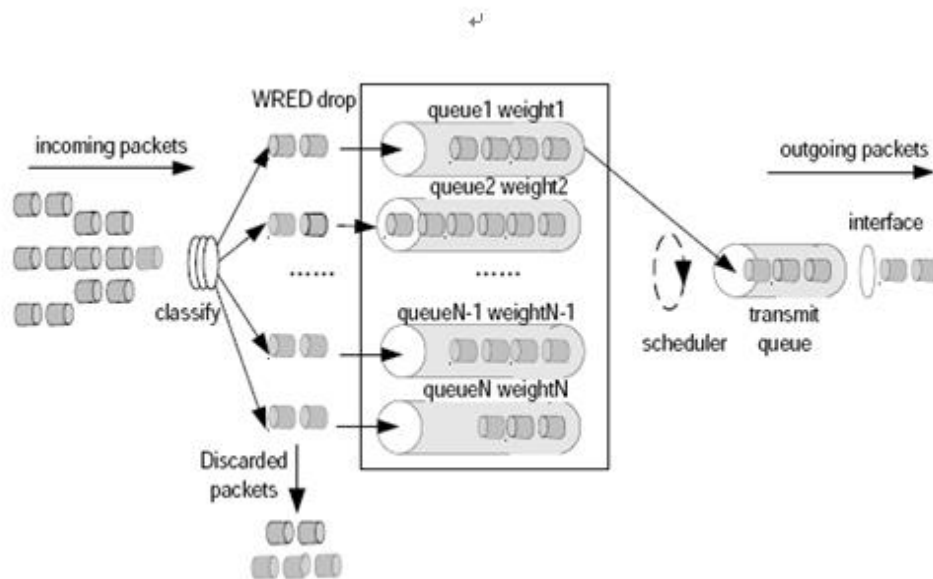


Figure 1

Upper and lower thresholds are set for each queue, and the packets in a queue are processed as follows:

- When the queue size is shorter than the lower threshold, no packet is dropped;
- When the queue size reaches the upper threshold, all subsequent packets are dropped;
- When the queue size is between the lower threshold and the upper threshold, the WRED algorithm will be adopted to determine whether the packets will be dropped or not.

In practice, a random number will be assigned to each incoming packets, and this random number is compared with the drop probability of the existing queue. If this random number is larger than the drop probability, the packets will be dropped. The longer the queue is, the higher the drop probability will be. However, there will be a maximum drop probability.

2.17 Configure Congestion Avoidance

2.17.1.1 Congestion Avoidance (WRED) Configuration Tasks

To enable MPLS EXP value based congestion avoidance on the interface, execute the following command:

Command	Function
Qtech(config-if)# random-detect mpls-exp-based	Enable congestion avoidance based on the EXP value of MPLS packets.



Note To configure congestion avoidance on the interface, all interfaces of the system must enable fast forwarding function, otherwise this function will become invalid.

To configure the maximum threshold, minimum threshold and drop probability of each kind of traffics classified by experimental, execute the following commands in interface configuration mode:

Command	Function
Qtech(config)# interface <i>interface-type interface-number</i>	Specify the interface for congestion avoidance.
Qtech(config-if)# random-detect experimental <i>exp-value min-threshold max-threshold mark-prob-denominator</i>	Configure the maximum threshold, minimum threshold and drop probability of each kind of traffics classified by experimental.

Exp-value: experimental value; traffics are classified according to this value.

Min-threshold: the minimum drop threshold; the default value differs from traffic to traffic.

Max-threshold: the maximum drop threshold; the default value differs from traffic to traffic.

Mark-prob-denominator: drop probability; the default value is 10, i.e., 1/10. The larger this value is, the smaller the drop probability will be.

To configure the weight factor for traffic congestion avoidance on the interface, execute the following commands in privileged user mode:

Command	Function
Qtech(config)# interface <i>interface-type interface-number</i>	Specify the interface for congestion avoidance.
Qtech(config-if)# random-detect exponential-weighting-constant <i>exponential-value</i>	Configure the weight factor for traffic congestion avoidance on the interface.

Exponential-value: the default value of weight factor is 9; the smaller this value is, the larger the drop probability will be; the larger this value is, the smaller the drop probability will be.



Note If the queuing algorithm of Ethernet interface is not FIFO, you must "no" the queuing algorithm before configuring WRED congestion avoidance on the interface. If the queuing algorithm of synchronization interface is not FIFO or WFQ, you must "no" the queuing algorithm before configuring WRED congestion avoidance on the interface.

2.18 Congestion Avoidance (WRED) Configuration Examples

2.18.1 Example of Configuring Congestion Avoidance on the Interface

Configure WRED congestion avoidance based on MPLS experimental traffic classification on the synchronization interface.

Set lower threshold to 5, upper threshold to 100 and drop probability to 10 for packets with experimental value being 1.

Set lower threshold to 10, upper threshold to 100 and drop probability to 10 for packets with experimental value being 2.

Set lower threshold to 20, upper threshold to 100 and drop probability to 10 for packets with experimental value being 3.

Set lower threshold to 30, upper threshold to 100 and drop probability to 10 for packets with experimental value being 4.

```
interface Serial 3/0
 ip ref
 ip address 192.168.20.3 255.255.255.0
 mpls ip
 random-detect mpls-exp-based
 random-detect experimental 1 5 100 10
 random-detect experimental 2 10 100 10
 random-detect experimental 3 20 100 10
 random-detect experimental 4 30 100 10
# View interface configurations in privileged user mode:
Qtech#show queue interface serial 3/0

Current random-detect configuration:
  Serial 3/0
    Queuing strategy: random early detection (WRED)
    Exp-weight-constant: 9 (1/512)
    Mean queue depth: 0
class          Random drop      Tail drop      Minimum Maximum  Mark
                pkts/bytes      pkts/bytes      thresh  thresh
0      prob
  1/10          0/0              0/0              20     40
1      1/10          0/0              0/0              5      100
2      1/10          0/0              0/0              10     100
3      1/10          0/0              0/0              20     100
4      1/10          0/0              0/0              30     100
5      1/10          0/0              0/0              31     40
6      1/10          0/0              0/0              33     40
7      1/10          0/0              0/0              35     40

Qos Ref queue information
Current Policy(s) : WRED
Queuing strategy: random early detection (WRED)
interface cir: 2048000
Dequeue threshold: Green 25000, Yellow 37500, Red 50000
Queues: Queues total len 0, MeanBurst 800
Queues: gts gap 7, deta bits 262, token bucket 51200
Queues: Max 19353 pkts, used 0 pkts
Queues: rtpQ: 0 pkts, 0 bytes
```

```
Queues: llQ: 0 pkts, 0 bytes
Queues: genQ: 0 pkts, 0 bytes
```

2.19 QoS Overview

2.20 Understanding QoS

2.20.1 QoS Overview

Devices on a conventional IP network equally treat all data packets with a First In First Out (FIFO) policy and deliver each data packet with the best effort to the destination. They do not provide any guarantee for packet transmission performance such as reliability and transmission delay.

As the Internet becomes rapidly popular in the globe and information networks emerge one after another in today's society, people raise increasingly-higher requirements for networks. Today, information requirements are no longer confined to mere data information but also extend to interactive multimedia. Services are developing towards data, voice, unified image, and integrated network transmission. Highly-real-time voice, image, and important data services sensitive to bandwidth delay and jitter tend to be more widely transmitted on networks. On one hand, this helps greatly improve network resources. On the other hand, an issue about how to guarantee network Quality of Service (QoS) arises, since voice, data, and image services have different delay, throughput, and packet loss rate requirements.

The QoS mechanism is intended to provide different QoS to meet diversified service quality requirements.

2.20.2 Basic Concepts

Three QoS models are defined to meet different service quality requirements. They are the Best-Effort Service model, the Integrated Service Model, and the Differentiated Service (DiffServ) model.

2.20.2.1 Best-Effort Service

The Best-Effort Service model enables a network to transmit packets with the best effort but does not provide any guarantee for transmission performance such as delay and reliability. It is a default service model applied on the conventional Internet.

2.20.2.2 Integrated Service

In the Integrated Service model, an application program needs to submit a specific QoS request to the network before sending a packet. The request covers the required bandwidth and delay. The application program starts to send the packet only after receiving an acknowledgment from the network (that is, after the network reserves resources for the application program). In addition, packets sent by the application program must be controlled within the traffic range described by traffic parameters.

2.20.2.3 Differentiated Service

In the Differentiated Service model, an application program does not need to submit a resource request to the network before sending a packet. Instead, it sets QoS parameter information in the IP header of the packet to inform network nodes of its QoS requirements. All routers on the packet propagation path can analyze the IP header to obtain the QoS class of the packet.

2.20.3 Working Principles

Major QoS technologies include traffic classification, traffic policing, traffic shaping, congestion management, and congestion avoidance. The Integrated Service model also introduces a protocol called the Resource Reservation Protocol (RSVP).

2.20.3.1 Traffic Classification

Objects are identified based on certain matching rules. Traffic classification is a precondition for implementing differentiated services.

2.20.3.2 Traffic Policing

Traffic policing is used to monitor and control the specifications of specific traffic inbound to routers. It applies when traffic goes beyond specifications.

2.20.3.3 Traffic Shaping

Traffic shaping is a means to control traffic by actively adjusting the output rate of traffic. In general, it enables traffic to adapt to available network resources on the downstream router so as to avoid packet loss or congestion.

2.20.3.4 Congestion Management

Congestion management is a measure for solving resource contention in the event of network congestion. In general, packets are cached in queues and a certain scheduling algorithm is applied to arrange the forwarding sequence of packets.

2.20.3.5 Congestion Avoidance

Excessive congestion will cause great harm to network resources.

2.20.3.6 RSVP

RSVP is an end-to-end (E2E) resource reservation protocol. Resource requests are transmitted between network nodes. Upon receipt of these requests, a network node needs to allocate resources for these requests.

Qtech QoS mechanism supports the DiffServ model. The following sections describe how to configure QoS technologies.

2.21 Traffic Classification Configuration

2.22 Understanding Traffic Classification

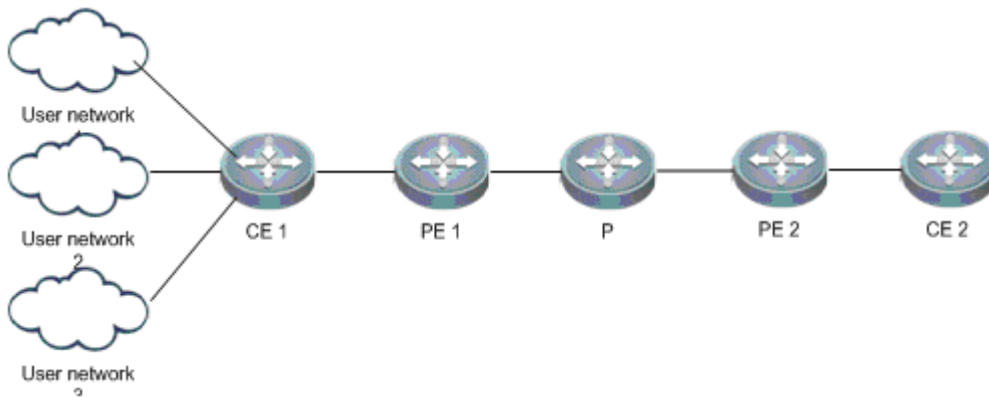
2.22.1 Traffic Classification Overview

When the DiffServ model is applied for QoS implementation, routers need to identify various flows and therefore traffic classification must be performed for packets. Two methods can be used for traffic classification: complex traffic classification and simple traffic classification.

Complex traffic classification is a means to classify packets in a fine manner using complex rules, such as rules based on link layer, network layer, and transport layer information (e.g. source MAC address, destination MAC address, source IP address, destination IP address, user group number, protocol type, or TCP/UDP port number of applications). In general, complex traffic classification is applied to traffic on border routers in the DiffServ domain.

As shown in Figure 9, CE 1, PE 1, P, PE 2, and CE 2 form a service provider network. PE 1, P, and PE 2 establish MPLS neighbors with each other. User networks 1, 2, and 3 access an MPLS network from CE 1 which is a service provider edge device. The traffic of user networks accesses the MPLS network from CE 1. Priority remarking must be performed for traffic of the three user networks, so that the traffic of different users is processed on the MPLS network according to different priorities. CE 1 must support complex traffic classification and related policies in terms of QoS.

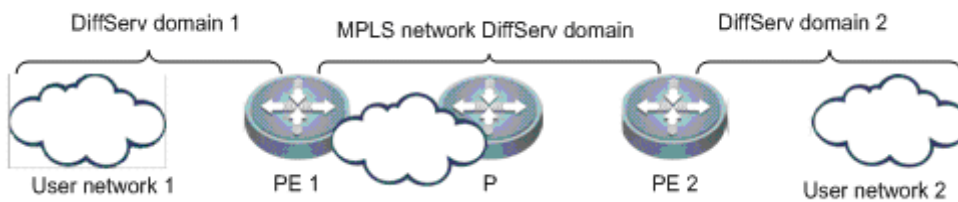
Figure 9 QoS Mapping for User Network Traffic Accessing an MPLS Network



Simple traffic classification is a means to roughly classify packets using simple rules, such as the IP priority or DSCP value of an IP packet, the EXP value of an MPLS packet, or the 802.1p value of a VLAN packet, so as to identify traffic featuring different priorities or Classes of Service (CoS). In general, simple traffic classification is applied only on core routers in the DiffServ domain.

As shown in Figure 10, PE 1, P, and PE 2 form a backbone network and establish MPLS neighbors with each other. User networks 1 and 2 access the backbone network from the PE. The traffic of user network 1 reaches user network 2 through the backbone network. Service priority mapping must be performed for services between different DiffServ domains.

Figure 10 QoS Mapping for User Network Traffic Across an MPLS Network



2.22.2 Basic Concepts

2.22.2.1 QoS Priority

Services are classified by QoS requirements into eight types. Packets are first classified and marked after accessing the system. They are treated differently according to packet priorities on the entire forwarding path. The following table defines service priorities.

Table 1 Service Priorities

Code	Service Level	Description
7	CS7	Used for in-band control messages, it represents the highest priority.
6	CS6	Used for protocol packets on the control plane, such as routing protocol packets and Bidirectional Forwarding Detection (BFD) packets.
5	EF (Expedited Forwarding)	Used for services sensitive to delay, jitter, and the packet loss rate, such as VoIP/TDM service packets.
4	AF4	Such services are surely forwarded when they do not exceed the maximum allowed bandwidth. Once the maximum bandwidth is exceeded, however, some packets will be discarded according to a discard priority. Such services are further classified into four types, and different bandwidths are allocated to different service types.
3	AF3	
2	AF2	
1	AF1	
0	BE (Best Effort)	Used for services insensitive to delay, jitter, and the packet loss rate, such as web, FTP, and other Internet services.

2.22.2.2 QoS Coloring

RFC 2697 and RFC 2698 have defined a service coloring mechanism, which uses three colors Green, Yellow, and Red to implement traffic control for services.

The system colors services based on different service policies to implement different packet discard policies.

2.22.2.3 Traffic Classifier

In complex traffic classification, it is necessary to define classifiers for various service flows, including classifiers based on different network features such as IPv4, IPv6, MPLS, and VLAN.

In simple traffic classification, traffic classifiers are used to classify traffic based on priorities only, such as DSCP for an IP network, EXP for an MPLS network, and CoS for an 802.1P network.

2.22.2.4 Traffic Behavior

In complex traffic classification, It is necessary to define the behaviors of various flows, such as Hierarchical QoS (HQoS) marking, priority re-marking, and QoS marking. QoS marking includes priority marking and packet coloring.

In simple traffic classification, however, traffic behaviors support only QoS marking and priority marking.

2.22.2.5 Traffic Policy

In complex traffic classification, traffic policies are used to associate traffic classifiers with traffic behaviors. One traffic policy can be used to associate multiple traffic classifiers with traffic behaviors so as to perform different operations for different service flows.

In simple traffic classification, a traffic policy consists of an uplink traffic classification mapping table and a downlink traffic classification mapping table. The uplink traffic classification mapping table implements mapping between priorities and QoS levels, whereas the downlink traffic classification mapping table implements mapping from QoS levels to priorities.



Caution Simple traffic classification supports mapping between service priorities, CoSs, and discard priorities so as to implement priority bearing and mapping for inter-domain devices.



Caution In complex traffic classification, priority marking supports only traffic policies for traffic classification and behaviors on homogeneous networks. For example, the system can perform MPLS priority marking after matching packets with MPLS traffic features.

2.22.2.6 DiffServ Domain

In the DiffServ model, multiple DiffServ domains are defined and various priority policies are applied. The priority policies for different domains may be different. Therefore, priority mapping must be performed for services across DiffServ domains to guarantee point-to-point QoS.

Universal priority policies are respectively defined for QoS of IP, MPLS, and VLAN networks. DSCP is used for IP networks, EXP for MPLS networks, and CoS for VLAN networks.

Table 2 Default Mapping Between DiffServ Domain DSCP Values and QoS Service Types

DSCP	Service	Color	DSCP	Service	Color
00	BE	Green	32	AF4	Green
01	BE	Green	33	BE	Green
02	BE	Green	34	AF4	Green
03	BE	Green	35	BE	Green

DSCP	Service	Color	DSCP	Service	Color
04	BE	Green	36	AF4	Yellow
05	BE	Green	37	BE	Green
06	BE	Green	38	AF4	Red
07	BE	Green	39	BE	Green
08	AF1	Green	40	EF	Green
09	BE	Green	41	BE	Green
10	AF1	Green	42	BE	Green
11	BE	Green	43	BE	Green
12	AF1	Yellow	44	BE	Green
13	BE	Green	45	BE	Green
14	AF1	Red	46	EF	Green
15	BE	Green	47	BE	Green
16	AF2	Green	48	CS6	Green
17	BE	Green	49	BE	Green
18	AF2	Green	50	BE	Green
19	BE	Green	51	BE	Green
20	AF2	Yellow	52	BE	Green
21	BE	Green	53	BE	Green
22	AF2	Red	54	BE	Green
23	BE	Green	55	BE	Green
24	AF3	Green	56	CS7	Green
25	BE	Green	57	BE	Green
26	AF3	Green	58	BE	Green
27	BE	Green	59	BE	Green
28	AF3	Yellow	60	BE	Green
29	BE	Green	61	BE	Green
30	AF3	Red	62	BE	Green
31	BE	Green	63	BE	Green

Table 3 Default Mapping Between DiffServ Domain QoS Service Types and DSCP Values

Service	Color	DSCP
BE	Green, Yellow, Red	0
AF1	Green	10
AF1	Yellow	12
AF1	Red	14
AF2	Green	18
AF2	Yellow	20
AF2	Red	22
AF3	Green	26
AF3	Yellow	28
AF3	Red	30
AF4	Green	34
AF4	Yellow	36
AF4	Red	38
EF	Green, Yellow, Red	46
CS6	Green, Yellow, Red	48
CS7	Green, Yellow, Red	56

Table 4 Default Mapping Between DiffServ Domain EXP Values and QoS Service Types

EXP	Service	Color
00	BE	Green
01	AF1	Green
02	AF2	Green
03	AF3	Green
04	AF5	Green
05	EF	Green
06	CS6	Green

07	CS7	Green
----	-----	-------

Table 5 Default Mapping Between DiffServ Domain QoS Service Types and EXP Values

Service	Color	EXP
BE	Green, Yellow, Red	0
AF1	Green, Yellow, Red	1
AF2	Green, Yellow, Red	2
AF3	Green, Yellow, Red	3
AF4	Green, Yellow, Red	4
EF	Green, Yellow, Red	5
CS6	Green, Yellow, Red	6
CS7	Green, Yellow, Red	7

Table 6 Mapping Between CoSs and QoS Service Types

CoS	Service	Color
00	BE	Green
01	BE	Green
02	AF2	Green
03	AF2	Green
04	AF4	Green
05	AF4	Green
06	CS6	Green
07	CS7	Green

Table 7 Default Mapping Between DiffServ Domain QoS Service Types and CoSs

Service	Color	CoS
BE	Green, Yellow, Red	0
AF1	Green, Yellow, Red	1
AF2	Green, Yellow, Red	2
AF3	Green, Yellow, Red	3
AF4	Green, Yellow, Red	4
EF	Green, Yellow, Red	5
CS6	Green, Yellow, Red	6
CS7	Green, Yellow, Red	7

2.22.3 Working Principles

2.22.3.1 Simple Traffic Classification

Qtech supports the following simple flow classification:

- Simple traffic classification for DiffServ domains of VLAN networks
- Simple traffic classification for DiffServ domains of MPLS networks
- Simple traffic classification for DiffServ domains of IP networks
- Uplink traffic mapping from DiffServ priorities to QoS
- Downlink traffic mapping from QoS to DiffServ priorities
- Applying simple traffic classification policies based on L3 interfaces or virtual templates

2.22.3.2 Complex Traffic Classification

Qtech supports the following complex flow classification:

- Complex traffic classification based on L2 information for VLAN networks

- Complex traffic classification based on L2 information for MPLS networks
- Complex traffic classification based on L2 information for IP networks
- Traffic classification policies: associated user queues, priority re-marking, and QoS marking
- Priority re-marking across DiffServ domains
- Applying simple traffic classification policies based on L3 interfaces or virtual templates

2.22.4 Protocols and Specifications

- RFC2474: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
- RFC2597: Assured Forwarding PHB
- RFC2598: Expedited Forwarding PHB
- RFC2697: A Single Rate Three Color Marker
- RFC2698: A Two Rate Three Color Marker

2.23 Default Configurations

The following table shows default traffic classification configurations.

Feature	Default
Complex traffic classification	Disabled
Complex traffic behaviors	None
Complex traffic classification polices	None
Simple traffic classification	Disabled
DiffServ domain in simple traffic classification	A default domain is created
Simple traffic classification policies	The default policy is consistent with the default domain

2.24 Configuring Complex Traffic Classification

2. 24. 1 Configuring Traffic Classifiers

Traffic classifiers are configured to distinguish the traffic of different users from one another so as to provide differentiated services for different users. Each traffic classifier may contain multiple matching rules, and the relationship between the matching rules is determined by the classifier type. If the classifier type is "and", all rules apply to the packet. If the rule type is "or", the packet can match any of the rules. If the classifier type is not specified, the "or" relationship applies between the matching rules. To configure traffic classifiers, perform the following steps:

Command	Purpose
Qtech(config)# traffic classifier <i>classifier-name</i> [and or]	Enter or create a traffic classifier.
Qtech(config-traffic-classifier)# if-match acl <i>acl-name</i> Or: Qtech(config-traffic-classifier)# if-match dscp <i>dscp-value</i> Or: Qtech(config-traffic-classifier)# if-match ip-precedence <i>ip-precedence-value</i> Or: Qtech(config-traffic-classifier)# if-match any	Set an IPv4 packet matching rule, which can be based on ACL, DSCP, or IP precedence, or any IPv4 packet.
Qtech(config-traffic-classifier)# if-match ipv6 acl <i>ipv6-acl-number</i> Or: Qtech(config-traffic-classifier)# if-match ipv6 dscp <i>dscp-value</i> Or: Qtech(config-traffic-classifier)# if-match ipv6 any	Set an IPv6 packet matching rule, which can be based on ACL, DSCP, or any IPv6 packet.

Qtech(config-traffic-classifier)# if-match mpls-exp <i>mpls-exp-value</i>	Set an MPLS packet matching rule, which can be based on MPLS EXP.
Qtech(config-traffic-classifier)# if-match cos <i>cos-value</i> Or: Qtech(config-traffic-classifier)# if-match source-mac <i>mac-address</i> Or: Qtech(config-traffic-classifier)# if-match destination-mac <i>mac-address</i>	Set an Ethernet packet matching rule, which can be based on CoS, source MAC address, or destination MAC address.
Qtech(config-traffic-classifier)# exit	Exit the traffic classifier configuration view.

Configuration example:

Create a traffic classifier and set an IPv4 packet matching rule based on an ACL:

```
Qtech(config)#traffic classifier tc1
Qtech(config-traffic-classifier)#if-match acl 100
Qtech(config-traffic-classifier)#exit
```

2.24.2 Configuring Traffic Behaviors

Traffic behaviors determine traffic scheduling parameters after traffic is classified. To configure traffic behaviors, perform the following steps:

Command	Purpose
Qtech(config)# traffic behavior <i>traffic-behavior-name</i>	Enter or create a traffic behavior.
Qtech(config-traffic-behavior)# user-queue <i>user-queue-name</i> [inbound outbound]	Set a user queue.
Qtech(config-traffic-behavior)# service-class <i>service-class-value</i> color <i>color-value</i>	Set the service class and discard priority of packets.
Qtech(config-traffic-behavior)# remark dscp <i>dscp-value</i> Or: Qtech(config-traffic-behavior)# remark ip-precedence <i>ip-precedence-value</i>	Set the remark value of IPv4 packets.
Qtech(config-traffic-behavior)# remark ipv6 dscp <i>dscp-value</i>	Set the remark value of IPv6 packets.
Qtech(config-traffic-behavior)# remark mpls-exp <i>mpls-exp-value</i>	Set the remark value of MPLS packets.
Qtech(config-traffic-behavior)# remark cos <i>cos-value</i>	Set the 802.1Q remark value of Ethernet packets.
Qtech(config-traffic-behavior)# sub-traffic-policy <i>traffic-policy-name</i>	Set an associated sub-traffic policy.
Qtech(config-traffic-behavior)# rtp-shaping delay <i>delay-time</i> clock-rate <i>clock-rate</i>	Set RTP shaping behavior.

Configuration example:

Create a traffic behavior.

```
Qtech(config)#traffic behavior tb1
Qtech(config-traffic-behavior)#user-queue uq1 inbound
Qtech(config-traffic-behavior)#service-class ef color green
Qtech(config-traffic-behavior)#remark dscp 40
```

2.24.3 Configuring Traffic Policies

Traffic policies associate traffic classifiers with traffic behaviors, so that classified traffic is scheduled according to users' configurations. To configure traffic policies, perform the following steps:

Command	Purpose
Qtech(config)# traffic policy <i>traffic-policy-name</i>	Enter or create a traffic policy.

Qtech(config-traffic-policy)# classifier <i>classifier-name</i> behavior <i>behavior-name</i> [precedence <i>precedence-value</i>]	Specify the traffic behavior for a traffic classifier and set the preference. The smaller the precedence value, the higher the preference.
--	--

Configuration example:

Create a traffic policy.

```
Qtech(config)#traffic policy tp1
Qtech(config-traffic-policy)#classifier tc1 behavior tb1 precedence
1
```

2.24.4 Applying Traffic Classification Policies

To apply traffic classification policies, perform the following steps:

Command	Purpose
Qtech(config)# interface <i>interface-name</i>	Enter interface configuration mode.
Qtech(config-if)# traffic-policy <i>traffic-policy-name</i> { inbound outbound } [link-layer all-layer]	Apply a traffic policy to the interface. You need to specify the layer parameter. By default, a policy takes effect for L3 and MPLS packets only. If the <i>link-layer</i> parameter is specified, the policy takes effect for 802.1P L2 packets only. If the <i>all-layer</i> parameter is specified, the policy takes effect for both L3 and L2 packets. If you specify the <i>link-layer</i> or <i>all-layer</i> parameter, the configured traffic policy is applicable to both the interface and all associated subinterfaces but not merely its subinterfaces. ATM interfaces and subinterfaces do not support the <i>link-layer</i> or <i>all-layer</i> parameter.

Configuration example:

Apply a traffic policy to an interface.

```
Qtech(config)#int gigabitethernet 0/1
Qtech(config-if-Gigabitethernet 0/1)#traffic-policy tp1 inbound
```

2.24.5 Displaying Configurations

Command	Purpose
show traffic classifier [<i>classifier-name</i>]	Show traffic classifier configurations in the system.
show traffic behavior [<i>behavior-name</i>]	Show traffic behavior configurations in the system.
show traffic policy [<i>policy-name</i>]	Show traffic policy configurations in the system.

Configuration example:

Show information about the interfaces of the port queue in the system.

```
Qtech# show traffic classifier tc1
traffic classifier tc1 or
if-match acl 1501
```

2.25 Configuring Simple Traffic Classification

2.25.1 Configuring DiffServ Domains and Traffic Policies

You need to first define DiffServ domains and specify traffic policies for the DiffServ domains during simple traffic classification. To configure DiffServ domains and traffic policies, perform the following steps:

Command	Purpose
Qtech(config)# diffserv domain { <i>ds-domain-name</i> default }	Enter or create a DiffServ domain.

<p>Qtech(config-diffserv-domain)# ip-dscp-inbound <i>dscpvalue phb service-class color</i></p> <p>Or: Qtech(config-traffic-classifier)# ip-dscp-outbound <i>serviceclass color map dscp-value</i></p>	Configure an IP traffic policy.
<p>Qtech(config-diffserv-domain)# mpls-exp-inbound <i>exp</i> phb <i>service-class color</i></p> <p>Or: Qtech(config-diffserv-domain)# mpls-exp-outbound <i>service-class color map exp-value</i></p>	Configure an MPLS traffic policy.
<p>Qtech(config-diffserv-domain)# 8021p-inbound <i>cos-value</i> phb <i>service-class color</i></p> <p>Or: Qtech(config-diffserv-domain)# 8021p-outbound <i>service-class color</i> map <i>cos-value</i></p>	Configure an 802.1P traffic policy.
Qtech(config-traffic-classifier)# exit	Exit the traffic classifier configuration view.

Configuration example:

Create a DiffServ domain named "out-ip" and set the mapping from IP DSCP to QoS.

```
Qtech(config)# diffserv domain out-ip
Qtech(config-diffserv-domain)# ip-dscp-inbound 34 phb ef green
Qtech(config-diffserv-domain)#exit
```

2.25.2 Applying Traffic Classification Polices

To apply traffic classification policies, perform the following steps:

Command	Purpose
Qtech(config)# interface <i>interface-name</i>	Enter interface configuration mode.
Qtech(config-if)# trust upstream { <i>ds-domain-name</i> default }	Apply a simple traffic classification policy to the interface.
Qtech(config-if)# trust 8021p	Enable 802.1p simple traffic classification. This configuration is applicable to interfaces only. The configured traffic policy is applicable to both the interface and all associated subinterfaces but not merely its subinterfaces. This command is not available for ATM interfaces or subinterfaces.

Configuration example:

Apply a traffic policy of the DiffServ domain named "out-ip" to an interface.

```
Qtech(config)#int gigabitethernet 0/1
Qtech(config-if-Gigabitethernet 0/1)#trust upstream out-ip
```

2.25.3 Displaying Configurations

Command	Purpose
show diffserv domain <i>diffserv-domain-name</i> [<i>8021p-inbound</i> <i>8021p-outbound</i> <i>ip-dscp-inbound</i> <i>ip-dscp-outbound</i> <i>mpls-exp-inbound</i> <i>mpls-exp-outbound</i>]	Show DiffServ domain configurations.

Configuration example:

Show configuration information about the DiffServ domain named "ip-out".

```
Qtech# show diffserv domain ipdscp
IP-DSCP map to Server-class and Color :
0 --> be green
1 --> be green
2 --> be green
3 --> be green
```

```
4 --> be green
5 --> be green
6 --> be green
7 --> be green
8 --> af1 green
9 --> be green
10 --> af1 green
11 --> be green
12 --> af1 yellow
13 --> be green
14 --> af1 red
15 --> be green
16 --> af2 green
17 --> be green
18 --> af2 green
19 --> be green
20 --> af2 yellow
21 --> be green
22 --> af2 red
23 --> be green
24 --> af3 green
25 --> be green
26 --> af3 green
27 --> be green
28 --> af3 yellow
29 --> be green
30 --> af3 red
31 --> be green
32 --> af4 green
33 --> be green
34 --> af4 green
35 --> be green
36 --> af4 yellow
37 --> be green
38 --> af4 red
39 --> be green
40 --> ef green
41 --> be green
42 --> be green
43 --> be green
44 --> be green
45 --> be green
46 --> ef green
47 --> be green
48 --> cs6 green
49 --> be green
50 --> be green
51 --> be green
52 --> be green
53 --> be green
54 --> be green
55 --> be green
56 --> cs7 green
57 --> be green
58 --> be green
59 --> be green
60 --> be green
61 --> be green
62 --> be green
63 --> be green
```

```
MPLS-EXP map to Server-class and Color :
0 --> be green
```



```
1 --> af1 green
2 --> af2 green
3 --> af3 green
4 --> af4 green
5 --> ef green
6 --> cs6 green
7 --> cs7 green

VLAN-Cos map to Server-class and Color :
0 --> be green
1 --> af1 green
2 --> af2 green
3 --> af3 green
4 --> af4 green
5 --> ef green
6 --> cs6 green
7 --> cs7 green

Server-class and Color map to IP-DSCP :
be green --> 0
be yellow --> 0
be red --> 0
af1 green --> 10
af1 yellow --> 12
af1 red --> 14
af2 green --> 18
af2 yellow --> 20
af2 red --> 22
af3 green --> 26
af3 yellow --> 28
af3 red --> 30
af4 green --> 34
af4 yellow --> 36
af4 red --> 38
ef green --> 46
ef yellow --> 46
ef red --> 46
cs6 green --> 48
cs6 yellow --> 48
cs6 red --> 48
cs7 green --> 56
cs7 yellow --> 56
cs7 red --> 56

Server-class and Color map to MPLS-EXP :
be green --> 0
be yellow --> 0
be red --> 0
af1 green --> 1
af1 yellow --> 1
af1 red --> 1
af2 green --> 2
af2 yellow --> 2
af2 red --> 2
af3 green --> 3
af3 yellow --> 3
af3 red --> 3
af4 green --> 4
af4 yellow --> 4
af4 red --> 4
ef green --> 5
ef yellow --> 5
ef red --> 5
```

```
cs6 green --> 6
cs6 yellow --> 6
cs6 red --> 6
cs7 green --> 7
cs7 yellow --> 7
cs7 red --> 7

Server-class and Color map to VLAN-CoS :
be green --> 0
be yellow --> 0
be red --> 0
af1 green --> 1
af1 yellow --> 1
af1 red --> 1
af2 green --> 2
af2 yellow --> 2
af2 red --> 2
af3 green --> 3
af3 yellow --> 3
af3 red --> 3
af4 green --> 4
af4 yellow --> 4
af4 red --> 4
ef green --> 5
ef yellow --> 5
ef red --> 5
cs6 green --> 6
cs6 yellow --> 6
cs6 red --> 6
cs7 green --> 7
cs7 yellow --> 7
cs7 red --> 7
```

2.26 Examples for Configuring Traffic Classification

2.26.1 Configuration Example 1

2.26.1.1 Networking Requirements

- Device requirements:

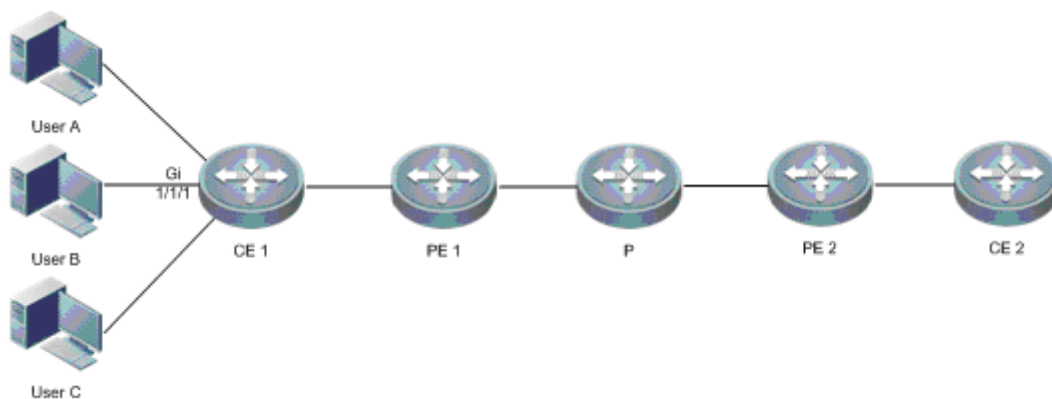
Four routers.

- Configuration requirements:

Two local users access an MPLS backbone network from a CE. The committed access rate of user A is 60 Mbps, and the peak access rate is 80 Mbps. The committed access rate of user B is 30 Mbps, and the peak access rate is 40 Mbps. The IP address of user A is 10.1.10.1, and that of user B is 10.1.10.2.

2.26.1.2 Network Topology

Figure 11 Network Topology for Complex Traffic Classification



2.26.1.3 Configuration Tips

Configure traffic classifiers to identify users.

Configure traffic behaviors to monitor and control user traffic.

2.26.1.4 Configuration Steps

Configure traffic classifiers:

```
# Configure a traffic classifier named "tc1".
Qtech(config)#access-list 100 permit ip host 10.1.10.1 any
Qtech(config)#traffic classifier tc1
Qtech(config-traffic-classifier)#if-match acl 100
Qtech(config-traffic-classifier)#exit
# Configure a traffic classifier named "tc2".
Qtech(config)# access-list 110 permit ip host 10.1.10.2 any
Qtech(config)#traffic classifier tc2
Qtech(config-traffic-classifier)#if-match acl 110
Qtech(config-traffic-classifier)#exit
```

Configure user queues:

```
# Configure a user queue named "uq1".
Qtech(config)#user-queue uq1 inbound
Qtech(config-user-queue)#cir 60000 pir 80000
Qtech(config-user-queue)#exit
```

```
# Configure a user queue named "uq2".
Qtech(config)#user-queue uq2 inbound
Qtech(config-user-queue)#cir 30000 pir 40000
Qtech(config-user-queue)#exit
```

Configure traffic behaviors:

```
#Configure a traffic behavior named "tb1".
Qtech(config)#traffic behavior tb1
Qtech(config-traffic-behavior)#user-queue uq1 inbound
Qtech(config-traffic-behavior)#exit
```

```
#Configure a traffic behavior named "tb2".
Qtech(config)#traffic behavior tb2
Qtech(config-traffic-behavior)#user-queue uq2 inbound
Qtech(config-traffic-behavior)#exit
```

Configure a traffic policy:

```
Qtech(config)#traffic policy tp1
Qtech(config-traffic-policy)#classifier tc1 behavior tb1
Qtech(config-traffic-policy)#classifier tc2 behavior tb2
Qtech(config-traffic-policy)#exit
```

Apply the traffic policy to an interface:

```
Qtech(config)#int gigabitethernet 1/1/1
Qtech(config-if-GigabitEthernet1/1/1)#traffic-policy tp1 inbound
```

2.26.1.5 Verification

Show traffic policy information:

```
Qtech# show traffic policy tp1
traffic policy tp1
  classifier tc1 behavior tb1 precedence 1
  classifier tc2 behavior tb2 precedence 2
```

2.26.2 Configuration Example 2

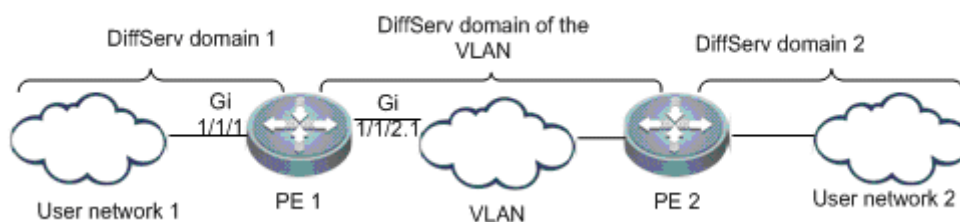
2.26.2.1 Networking Requirements

- Device requirements:
One Ethernet switch and two routers.
- Configuration requirements:

Two routers provide access services for the users of user network 1 and user network 2 respectively. The service priority of users of user network 1 must be able to be carried and transferred to user network 2 to implement DiffServ domain mapping across the VLAN. It is necessary to add HQOS configurations on PE 1.

2.26.2.2 Network Topology

Figure 12 User Network Interconnection Across a VLAN



2.26.2.3 Configuration Tips

- Configure DiffServ domains on routers.
- Apply simple traffic classification to the user networks and VLAN.

2.26.2.4 Configuration Steps

Create DiffServ domain policies:

```
# Configure a traffic classifier named "usera".
Qtech(config)# diffserv domain usera
```

```
Qtech(config-diffserv-domain)#ip-dscp-inbound 34 phb ef green
Qtech(config-diffserv-domain)#ip-dscp-inbound 16 phb be green
# Configure a traffic classifier named "vlan".
Qtech(config)# diffserv domain vlan
Qtech(config-diffserv-domain)# 8021p-outbound ef green map 4
Qtech(config-diffserv-domain)# 8021p-outbound be green map 1
```

Apply simple traffic classifiers to respective interfaces:

```
Qtech(config)#int gigabitethernet 1/1/1
Qtech(config-if-Gigabitethernet1/1/1)#trust upstream vlan
Qtech(config-if-Gigabitethernet1/1/1)#exit

Qtech(config)#int gigabitethernet 1/1/2.1
Qtech(config-if-Gigabitethernet1/1/2.1)#trust upstream usera
Qtech(config-if-Gigabitethernet1/1/2.1)#trust 8021p
Qtech(config-if-Gigabitethernet1/1/2.1)#exit
```

2.26.2.5 Verification

Show DiffServ domain configurations:

```
Qtech# show diffserv domain usera
IP-DSCP map to Server-class and Color:
 0 --> be green
 1 --> be green
 2 --> be green
 3 --> be green
 4 --> be green
 5 --> be green
 6 --> be green
 7 --> be green
 8 --> af1 green
 9 --> be green
10 --> af1 green
11 --> be green
12 --> af1 yellow
13 --> be green
14 --> af1 red
15 --> be green
16 --> be green
17 --> be green
18 --> af2 green
19 --> be green
20 --> af2 yellow
21 --> be green
22 --> af2 red
23 --> be green
24 --> af3 green
25 --> be green
26 --> af3 green
27 --> be green
28 --> af3 yellow
29 --> be green
30 --> af3 red
31 --> be green
32 --> af4 green
33 --> be green
34 --> ef green
35 --> be green
36 --> af4 yellow
37 --> be green
38 --> af4 red
39 --> be green
```

```
40 --> ef    green
41 --> be    green
42 --> be    green
43 --> be    green
44 --> be    green
45 --> be    green
46 --> ef    green
47 --> be    green
48 --> cs6   green
49 --> be    green
50 --> be    green
51 --> be    green
52 --> be    green
53 --> be    green
54 --> be    green
55 --> be    green
56 --> cs7   green
57 --> be    green
58 --> be    green
59 --> be    green
60 --> be    green
61 --> be    green
62 --> be    green
63 --> be    green
```

```
Qtech# show diffserv domain vlan
Server-class and Color map to VLAN-CoS :
be    green    --> 1
be    yellow   --> 0
be    red      --> 0
af1   green    --> 1
af1   yellow   --> 1
af1   red      --> 1
af2   green    --> 2
af2   yellow   --> 2
af2   red      --> 2
af3   green    --> 3
af3   yellow   --> 3
af3   red      --> 3
af4   green    --> 4
af4   yellow   --> 4
af4   red      --> 4
ef    green    --> 4
ef    yellow   --> 5
ef    red      --> 5
cs6   green    --> 6
cs6   yellow   --> 6
cs6   red      --> 6
cs7   green    --> 7
cs7   yellow   --> 7
cs7   red      --> 7
```

2.26.3 Configuration Example 3

2.26.3.1 Networking Requirements

- Device requirements:

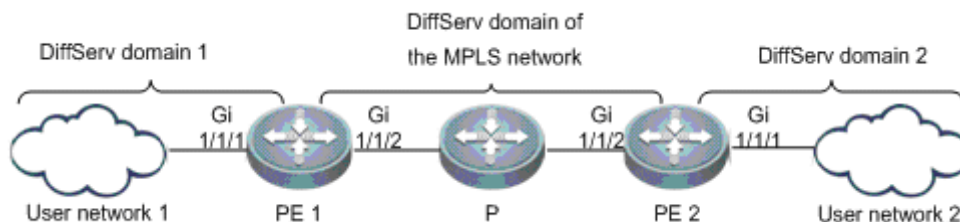
Five routers.

- Configuration requirements:

User network 1 and user network 2 access an MPLS backbone network from PE 1 and PE 2 respectively. The service priority of users of user network 1 must be able to be carried and transferred to user network 2 to implement DiffServ domain mapping across the MPLS network. It is necessary to add HQOS configurations on PE 1 and PE 2.

2.26.3.2 Network Topology

Figure 13 User Network Interconnection Across an MPLS Network



2.26.3.3 Configuration Tips

Configure DiffServ domains on routers.

Apply simple traffic classification to the user networks and MPLS network.

2.26.3.4 Configuration Steps

Configure PE 1 and create DiffServ domain policies:

```
# Configure a traffic classifier named "usera".
Qtech(config)# diffserv domain usera
Qtech(config-diffserv-domain)#ip-dscp-inbound 34 phb ef green
Qtech(config-diffserv-domain)#ip-dscp-inbound 16 phb be green
# Configure a traffic classifier named "mpls".
Qtech(config)# diffserv domain mplsa
Qtech(config-diffserv-domain)# mpls-exp-outbound ef green map 4
Qtech(config-diffserv-domain)# mpls-exp-outbound be green map 1
```

Configure PE1. Apply simple traffic classifiers to respective interfaces:

```
Qtech(config)#int gigabitethernet 1/1/1
Qtech(config-if-Gigabitethernet1/1/1)#trust upstream usera
Qtech(config-if-Gigabitethernet1/1/1)#exit

Qtech(config)#int gigabitethernet 1/1/2
Qtech(config-if-Gigabitethernet1/1/2)#trust upstream mplsa
Qtech(config-if-Gigabitethernet1/1/2)#exit
```

Configure PE 2 and create DiffServ domain policies:

```
# Configure a traffic classifier named "usera".
Qtech(config)# diffserv domain userb
Qtech(config-diffserv-domain)# ip-dscp-outbound ef green map
34Qtech(config-diffserv-domain)# ip-dscp-outbound be green map
16
# Configure a traffic classifier named "mpls".
Qtech(config)# diffserv domain mplsb
Qtech(config-diffserv-domain)# mpls-exp-inbound 4 phb ef green
Qtech(config-diffserv-domain)# mpls-exp-inbound 1 phb be green
```

Configure PE2. Apply simple traffic classifiers to respective interfaces:

```
Qtech(config)#int gigabitethernet 1/1/1
Qtech(config-if-Gigabitethernet1/1/1)#trust upstream userb
Qtech(config-if-Gigabitethernet1/1/1)#exit

Qtech(config)#int gigabitethernet 1/1/2
```

```
Qtech(config-if-GigabitEthernet1/1/2)#trust upstream mpls  
Qtech(config-if-GigabitEthernet1/1/2)#exit
```

2.26.3.5 Verification

Show DiffServ domain configurations on PE 1:

```
Qtech# show diffserv domain usera  
IP-DSCP map to Server-class and Color :  
 0 --> be    green  
 1 --> be    green  
 2 --> be    green  
 3 --> be    green  
 4 --> be    green  
 5 --> be    green  
 6 --> be    green  
 7 --> be    green  
 8 --> af1   green  
 9 --> be    green  
10 --> af1   green  
11 --> be    green  
12 --> af1   yellow  
13 --> be    green  
14 --> af1   red  
15 --> be    green  
16 --> be    green  
17 --> be    green  
18 --> af2   green  
19 --> be    green  
20 --> af2   yellow  
21 --> be    green  
22 --> af2   red  
23 --> be    green  
24 --> af3   green  
25 --> be    green  
26 --> af3   green  
27 --> be    green  
28 --> af3   yellow  
29 --> be    green  
30 --> af3   red  
31 --> be    green  
32 --> af4   green  
33 --> be    green  
34 --> ef    green  
35 --> be    green  
36 --> af4   yellow  
37 --> be    green  
38 --> af4   red  
39 --> be    green  
40 --> ef    green  
41 --> be    green  
42 --> be    green  
43 --> be    green  
44 --> be    green  
45 --> be    green  
46 --> ef    green  
47 --> be    green  
48 --> cs6   green  
49 --> be    green  
50 --> be    green  
51 --> be    green  
52 --> be    green  
53 --> be    green  
54 --> be    green
```



```
55 --> be    green
56 --> cs7   green
57 --> be    green
58 --> be    green
59 --> be    green
60 --> be    green
61 --> be    green
62 --> be    green
63 --> be    green
Qtech# show diffserv domain mplsa
Server-class and Color map to MPLS-EXP :
be    green    --> 1
be    yellow   --> 0
be    red      --> 0
af1   green    --> 1
af1   yellow   --> 1
af1   red      --> 1
af2   green    --> 2
af2   yellow   --> 2
af2   red      --> 2
af3   green    --> 3
af3   yellow   --> 3
af3   red      --> 3
af4   green    --> 4
af4   yellow   --> 4
af4   red      --> 4
ef    green    --> 4
ef    yellow   --> 5
ef    red      --> 5
cs6   green    --> 6
cs6   yellow   --> 6
cs6   red      --> 6
cs7   green    --> 7
cs7   yellow   --> 7
cs7   red      --> 7
```

Show DiffServ domain configurations on PE 2:

```
Qtech# show diffserv domain userb
Server-class and Color map to IP-DSCP :
be    green    --> 16
be    yellow   --> 0
be    red      --> 0
af1   green    --> 10
af1   yellow   --> 12
af1   red      --> 14
af2   green    --> 18
af2   yellow   --> 20
af2   red      --> 22
af3   green    --> 26
af3   yellow   --> 28
af3   red      --> 30
af4   green    --> 34
af4   yellow   --> 36
af4   red      --> 38
ef    green    --> 34
ef    yellow   --> 46
ef    red      --> 46
cs6   green    --> 48
cs6   yellow   --> 48
cs6   red      --> 48
cs7   green    --> 56
cs7   yellow   --> 56
cs7   red      --> 56
```

```
Qtech# show diffserv domain mpls
MPLS-EXP map to Server-class and Color :
 0 --> be    green
 1 --> be    green
 2 --> af2   green
 3 --> af3   green
 4 --> ef    green
 5 --> ef    green
 6 --> cs6   green
 7 --> cs7   green
```

3 CONFIGURING HQoS

3.1 Understanding HQoS

3.1.1 HQoS Overview

The traditional QoS offers different treatment to services and ensure QoS needs such as bandwidth and delay by classifying service flows and specifying policies for them. However, the existing user access network is complex, and there are a large number of access devices (such as Layer 2 switches and converters) that do not support complex QoS. Although egress devices of the user access network can ensure QoS for transmitted services as much as possible, they cannot ensure QoS based on the user and user group in a more refined manner.

Hierarchical QoS (HQoS) is different from the traditional QoS, which is specific to services. HQoS can provide QoS for data flows in the network by service, user, floor, and residential area. Service QoS provides quality service for multiple service flows of a single user. User QoS provides quality service for multiple users on a floor, and so on for floor QoS and residential area QoS. HQoS can implement more refined service assurance for data aggregation devices to improve service quality for whole-network users.

3.1.2 Basic Concepts

3.1.2.1 FQ

The flow queue (FQ) indicates service queues of a user. Each user has eight FQs, which correspond to eight service priorities respectively (BE, AF1, AF2, AF3, AF4, EF, CS6, and CS7). The eight FQs can be configured with Priority Queuing (PQ), Weighted Fair Queuing (WFQ), or Low-priority Queuing (LPQ). Each FQ supports Weighted Random Early Detection (WRED) and traffic shaping.

3.1.2.2 UQ

The user here indicates one VLAN or VPN. Users can be divided based on the interface, sub-interface, or ACL. Each user has a user queue (UQ), which is aggregated by eight fixed FQs. You can limit the total bandwidth of each user by performing rate limiting to its UQ.



Note If users belong to different line cards of a distributed device, the UQ function takes effect based on each line card.

3.1.2.3 GQ

Multiple users can be bound to one user group. Each user group has a group queue (GQ). You can control the traffic of a user group by performing traffic shaping to the GQ.



Note If user groups belong to different line cards of a distributed device, the GQ function takes effect based on each line card.

3.1.2.4 Destination Interface/Destination Device CQ

Each destination interface has eight queues, which correspond to eight service types respectively. The eight FQs can be configured with SP and WFQ. Each class queue (CQ) supports WRED and traffic shaping.

The destination device CQ can be regarded as an uplink destination interface CQ. They are the same in nature. The difference is that the packets of the destination device CQ are sent to boards instead of interfaces. Each destination device correspond to four queues, which correspond to four service types. The four queues are scheduled in SP mode.

3.1.2.5 LPQ

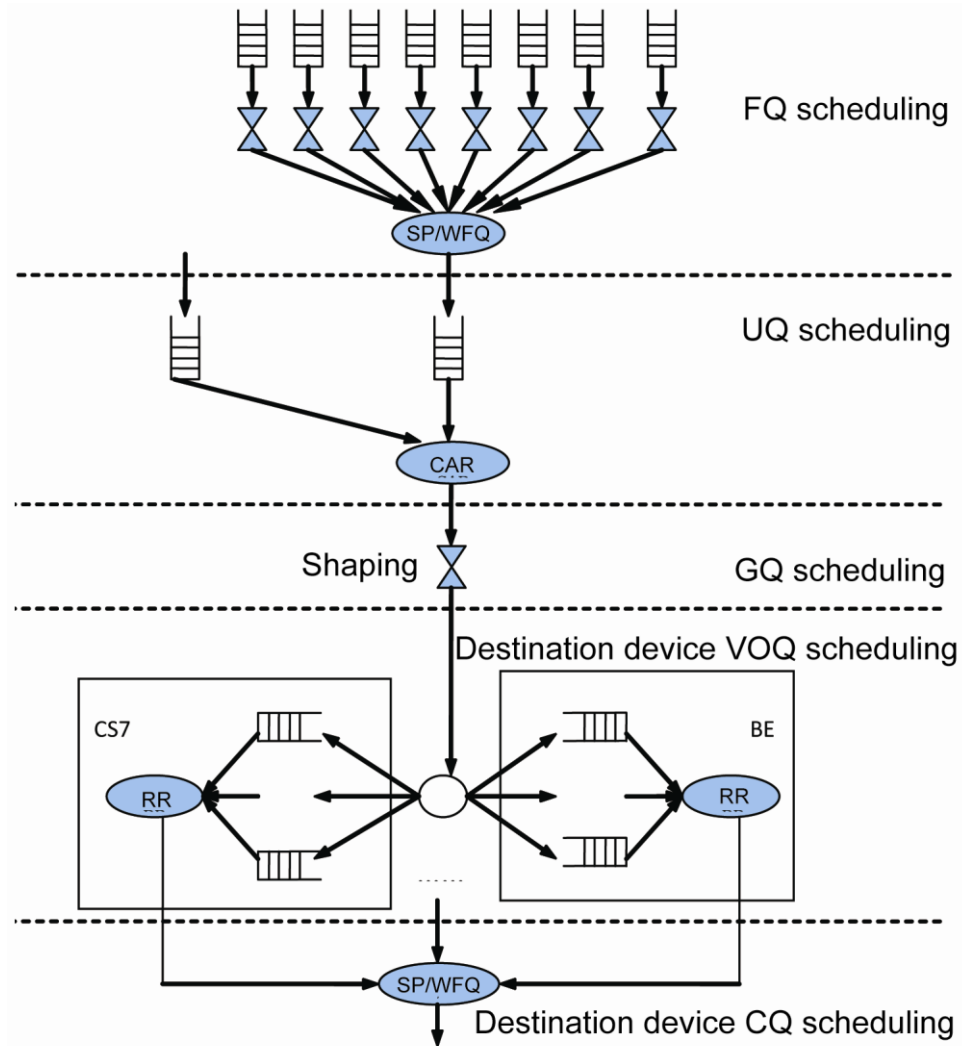
There are three scheduling mechanisms in HQoS: PQ, WFQ, and Low-priority Queue (LPQ), which descend in order of priority. In the case of congestion, PQ queues and WFQ queues can preempt the bandwidth of LPQ queues.

3.1.3 How HQoS Works

3.1.3.1 Uplink HQoS Scheduling

Uplink HQoS scheduling is divided into five levels: CQ > UQ > GQ > Destination device VOQ > Destination device CQ, as shown in the following figure:

Figure 1 Uplink HQoS scheduling process



Level 1 scheduling is FQ scheduling. The FQ is an entity queue, that is, the packets are cached in an FQ. FQ includes FQ WRED, FQ shaping, and FQ scheduling. WRED congestion avoidance is required before a packet enters a FQ. The system supports discarding priorities of three colors: red, yellow, and green. Red indicates the highest discarding priority, and green indicates the lowest one. Each FQ sets the minimum threshold, maximum threshold, and discarding probability for each discarding priority. The higher the discarding priority, the greater the maximum threshold and minimum threshold, and the greater the discarding probability of packets. Traffic shaping is performed via token bucket after WRED for FQ. Finally, FQ scheduling is performed by using Strict Priority (SP)+WFQ. SP includes PQ and LPQ. CS7, CS6, EF, AF, and BE are scheduled in sequence. Low-priority queues are scheduled only when there are no packets in high-priority queues. AF includes four types, which are scheduled according to their respective weights.

Level 2 scheduling is UQ scheduling. The UQ is a virtual queue, that is, the packets are not cached in a UQ. UQ scheduling is only an HQoS scheduling level. Each UQ includes eight FQs that share the bandwidth of the UQ. The UQ supports committed interface rate (CIR) only. Each UQ can invoke zero to one GQ.

Level 3 scheduling is GQ scheduling. The GQ is a virtual queue. Multiple UQs can be bound to a GQ for scheduling. The GQ supports traffic shaping only. If no UQ is bound to a GQ, GQ scheduling will be skipped.

Level 4 scheduling is VoQ scheduling. VoQ scheduling is the uplink traffic scheduling among line cards. The VoQ breaks into four groups by service priority, with one queue for each destination device in each group. When packets pass through UQ scheduling, they enter different VoQs based on their destination devices and priorities. VoQ scheduling is configured by the system, and no additional user configuration is required.

Level 5 scheduling is CQ scheduling. The CQ is a virtual queue, instead of an entity queue that caches packets. Each CQ, however, maintains information such as scheduling priority and weight, and supports SP/WFQ scheduling. CQ scheduling for uplink HQoS is configured by the system, and no additional user configuration is required.

- CoS-based uplink HQoS processing:

Flows are classified based on configured flow classification rules, and are prioritized at eight levels.

The UQ, GQ, WRED parameters, and scheduling policies of packets are determined based on flow behavior rules. Then, packets are passed to corresponding FQs.

- ◆ FQ scheduling can be performed according to user configuration to avoid congestion.

- ◆ An FQ can be scheduled as a PQ, WFQ, or LPQ. The PQ and WFQ can preempt the bandwidth of the LPQ.

The remaining bandwidth for the GQ is checked. If the remaining bandwidth for the GQ is insufficient, the UQ included in the GQ is not scheduled. Otherwise, the UQ included in the GQ is scheduled. The bandwidth for the GQ is configurable.

The remaining bandwidth for the UQ is checked. If the remaining bandwidth for the UQ is insufficient, the FQ included in the UQ is not scheduled. Otherwise, the FQ is scheduled. The bandwidth for the UQ is configurable.

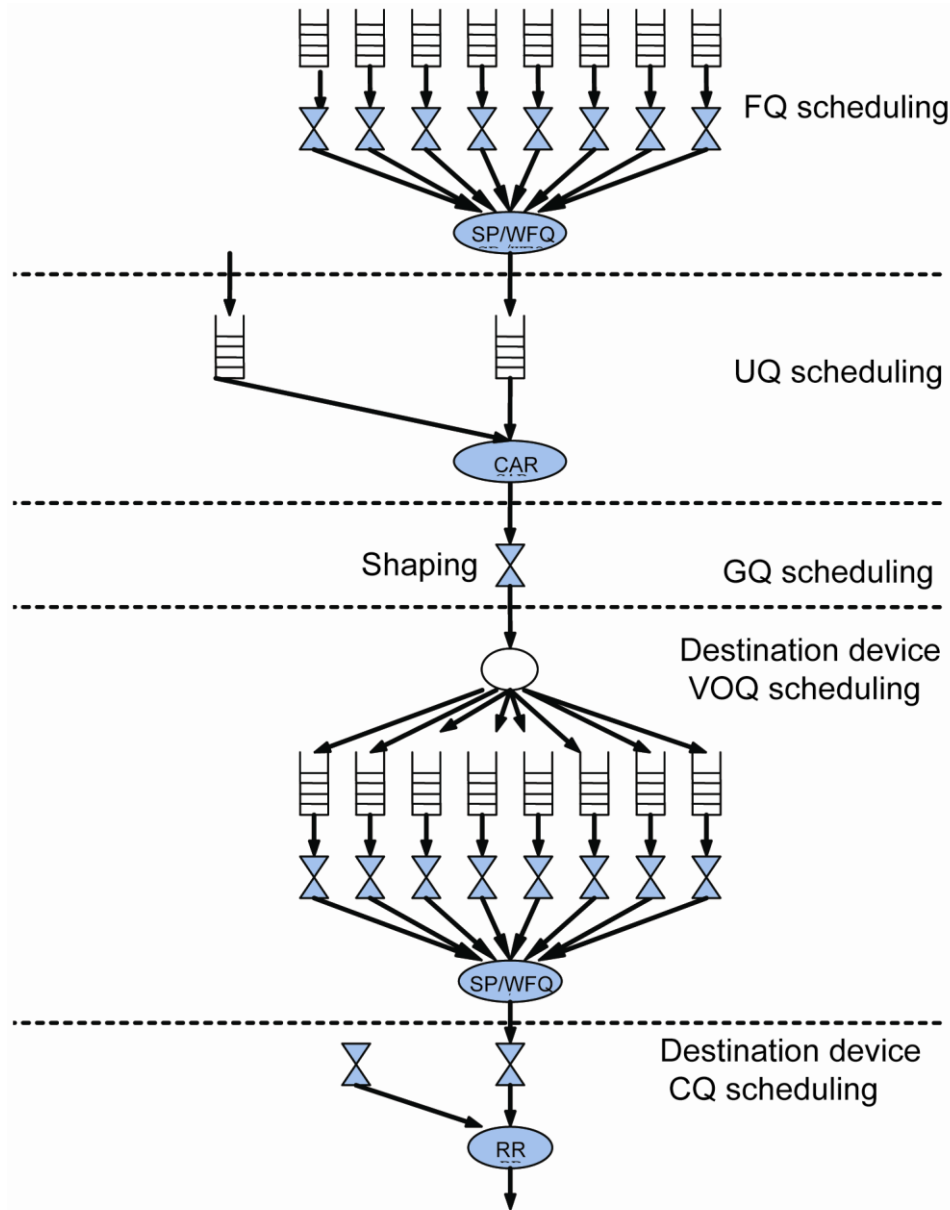
After scheduling, packets in the FQ enter the VoQs on destination devices or destination interfaces.

CQ scheduling is performed in SP mode among four queues. After scheduling, packets are forwarded.

3.1.4 Downlink HQoS Scheduling

Downlink HQoS scheduling is divided into five levels: FQ > UQ > GQ > Interface CQ > Destination interface, as shown in the following figure:

Figure 2 Downlink HQoS Scheduling Process



Level 1 FQ scheduling, level 2 UQ scheduling, and level 3 GQ scheduling are identical with those of uplink HQoS scheduling.

Level 4 scheduling is CQ scheduling. There are eight CQs on each interface, corresponding to eight service priorities. Different from the case of uplink HQoS scheduling, you can configure CQ congestion avoidance parameters and the traffic shaping value. The CQ is scheduled in SP + WFQ mode.

Level 5 scheduling is destination interface scheduling. The system polls interfaces for scheduling. Level 5 scheduling is configured by the system, and no additional user configuration is required.

3.1.5 Protocol Specifications

TR-059: A layered QoS model defined by the DSL Forum

3.2 Default Configuration

The following table describes the default HQoS configurations:

Function	Default
WRED queue lower threshold	20
WRED queue upper threshold	40
WRED queue discarding probability	100

3.3 Configuring CoS-based HqoS

3.3.1 Configuring a Traffic Classifier Rule

Configuring a traffic classifier rule is to distinguish between user flows to provide differentiated services for different users. Each traffic classifier rule can contain multiple match rules, the relationship among which is determined by the rule type. If the rule type is "and", the packet must match all rules; if the rule type is "or", the packet can match any one of the rules. If no rule type is specified, "or" rule type is adopted by default. Configure this function as follows:

Command	Purpose
Qtech(config)# traffic classifier <i>classifier-name</i> [and or]	Enters/Creates a traffic classifier rule.
Qtech(config-traffic-classifier)# if-match acl <i>acl-number</i> Or: Qtech(config-traffic-classifier)# if-match dscp <i>dscp-value</i> Or: Qtech(config-traffic-classifier)# if-match ip-precedence <i>ip-precedence-value</i> Or: Qtech(config-traffic-classifier)# if-match any	Sets IPv4 packet match rules. Supports matching packets based on ACL, DSCP, and IP precedence and matching all IPv4 packets (any).
Qtech(config-traffic-classifier)# if-match ipv6 <i>ipv6-acl-number</i> Or: Qtech(config-traffic-classifier)# if-match ipv6 dscp <i>dscp-value</i> Or: Qtech(config-traffic-classifier)# if-match ipv6 any	Sets IPv6 packet match rules. Supports matching packets based on ACL and DSCP, and matching all IPv6 packets (any).
Qtech(config-traffic-classifier)# if-match mpls-exp <i>mpls-exp-value</i>	Sets MPLS packet match rules. Supports matching packets based on MPLS EXP.
Qtech(config-traffic-classifier)# if-match cos <i>cos-value</i> Or: Qtech(config-traffic-classifier)# if-match source-mac <i>mac-address</i> Or: Qtech(config-traffic-classifier)# if-match destination-mac <i>mac-address</i>	Sets Ethernet packet match rules. Supports matching packets based on CoS, source MAC address, and destination MAC address.
Qtech(config-traffic-classifier)# exit	Exits the traffic classifier rule configuration.

Configuration example:

Create a traffic classifier rule and set the IPv4 packet match rule to be based on ACL.

```
Qtech(config)#traffic classifier tc1
Qtech(config-traffic-classifier)#if-match acl 100
Qtech(config-traffic-classifier)#exit
```

3.3.2 Configuring an FQ WRED Template

Configuring a WRED template is to configure congestion avoidance parameters, including upper threshold, lower threshold, and discarding probability, for packets in three colors. When the number of packets in a queue is smaller than the lower threshold, packets are not discarded. When the number of packets in a queue is larger than the lower threshold and smaller than the upper threshold, packets are discarded at certain probability. When the number of

packets in a queue is larger than the upper threshold, packets are discarded. You can configure different WRED templates and apply them to different FQs.

Configure this function as follows:

Command	Purpose
Qtech(config)# wred <i>wred-template-name</i>	Creates/Enters a WRED template.
Qtech(config-wred)# color green <i>low-limit threshold high-limit high-limit-threshold percent discard-percent-value</i>	Sets the upper threshold, lower threshold, and probability for queues in three colors.
Qtech(config-wred)# color yellow <i>low-limit threshold high-limit high-limit-threshold percent discard-percent-value</i>	
Qtech(config-wred)# color red <i>low-limit threshold high-limit high-limit-threshold percent discard-percent-value</i>	
Qtech(config-wred)# exit	Exits the WRED template configuration.

Configuration example:

Create a WRED template and set congestion avoidance parameters.

```
Qtech(config)#wred wt1
Qtech(config-wred)#color green low-limit 40 high-limit 60 discard-
percent 10
Qtech(config-wred)#color yellow low-limit 30 high-limit 50 discard-
percent 10
Qtech(config-wred)#color red low-limit 20 high-limit 40 discard-
percent 10
Qtech(config-wred)#exit
```

3.3.3 Configuring an FQ Template

Configuring an FQ template is to configure the scheduling mode (PQ, WFQ, and LPQ), traffic shaping value, and WRED parameters of eight kinds of PQs. PQ scheduling is performed on queues with the priority as ef, cs6, or cs7. WFQ scheduling is performed on queues with the priority as be, af1, af2, af3, or af4. Traffic shaping is not performed by default and the default discarding policy is tail discarding. You can configure multiple FQ templates and apply them to different FQs.

Configure this function as follows:

Command	Purpose
Qtech(config)# flow-queue <i>flow-queue-template-name</i>	Enters/Creates an FQ template.
Qtech(config-flow-queue)# queue <i>cos-value {pq wfq weight weight-value lpq} [shaping shaping-value] [wred wred-name] [depth depth-value]</i>	Set FQ scheduling parameters.
Qtech(config-flow-queue)# exit	Exits the FQ template configuration.

Configuration example:

Create an FQ template and set FQ scheduling parameters.

```
Qtech(config)#flow-queue fqt1
Qtech(config-flow-queue)# queue be lpq
Qtech(config-flow-queue)# queue af1 wfq weight 10 shaping 100000
wred wt1
Qtech(config-flow-queue)# queue cs7 pq shaping wred wt1
Qtech(config-flow-queue)#exit
```

3.3.4 Configuring an FQ Mapping Template

Configuring an FQ mapping template is to configure the mapping from eight kinds of PQs to CQs.

Configure this function as follows:

Command	Purpose
Qtech(config)# flow-mapping <i>flow-mapping name</i>	Enters/Creates an FQ mapping template.
Qtech(config-flow-mapping)# map flow-queue <i>cos-value</i> to port-queue <i>cos-value</i>	Sets the mapping from FQ to CQ.
Qtech(config-flow-mapping)# exit	Exits the FQ mapping template configuration.

Configuration example:

Create a FQ mapping template and set the mapping from FQ to CQ.

```
Qtech(config)#flow-mapping fmt1
Qtech(config-flow-mapping)# map flow-queue afl1 to port-queue ef
Qtech(config-flow-mapping)#exit
```

3.3.5 Configuring a UQ

Configuring a UQ consists of configuring the CIR, configuring the FQ template, and associating a GQ.

Configure this function as follows:

Command	Purpose
Qtech(config)# user-queue <i>user-queue-name</i> inbound outbound	Enters/Creates a UQ.
Qtech(config-user-queue)# cir <i>vir-value</i> pir <i>pir-value</i>	Sets UQ traffic shaping parameters. By default, the CIR of the UQ is 0.
Qtech(config-user-queue)# flow-queue <i>flow-queue-template-name</i>	Sets UQ FQ scheduling parameters.
Qtech(config-user-queue)# user-group-queue <i>user-group-queue-name</i>	Associates the UQ with a GQ.
Qtech(config-user-queue)# flow-mapping <i>flow-rmap-name</i>	Sets an FQ mapping template.

Configuration example:

Create a UQ and set scheduling parameters.

```
Qtech(config)#user-queue uq1 inbound
Qtech(config-user-queue)#cir 100000 pir 100000
Qtech(config-user-queue)#flow-queue fqt1
Qtech(config-user-queue)#user-group-queue ugq1
Qtech(config-user-queue)#flow-mapping fmt1
```

3.3.6 Configuring a GQ

A GQ is a bundle of multiple UQs for centralized traffic shaping.

Configure this function as follows:

Command	Purpose
Qtech(config)# user-group-queue <i>user-group-queue-name</i> [inbound outbound]	Enters/Creates a GQ.
Qtech(config-user-group-queue)# shaping <i>shaping-value</i>	Sets the traffic shaping value of the GQ.

Configuration example:

Create a GQ and set the traffic shaping value.

```
Qtech(config)#user-group-queue ugq1 inbound
Qtech(config-user-group-queue)#shaping 100000
```

3.3.7 Configuring a Traffic Behavior Rule

The traffic behavior rule determines traffic scheduling parameters after classification. Configure this function as follows:

Command	Purpose
Qtech(config)# traffic behavior <i>traffic-behavior-name</i>	Enters/Creates a traffic behavior rule.
Qtech(config-traffic-behavior)# user-queue <i>user-queue-name</i> [inbound outbound]	Sets the UQ.
Qtech(config-traffic-behavior)# service-class <i>service-class-value</i> color <i>color-value</i>	Sets the color of packets with different priorities.
Qtech(config-traffic-behavior)# remark dscp <i>dscp-value</i> Or: Qtech(config-traffic-behavior)# remark ip-precedence <i>ip-precedence-value</i> Or: Qtech(config-traffic-behavior)# remark mpls-exp <i>mpls-exp-value</i>	Sets the remark value of IPv4 packets.
Qtech(config-traffic-behavior)# remark ipv6 dscp <i>dscp-value</i>	Sets the remark value of IPv6 packets.
Qtech(config-traffic-behavior)# remark mpls-exp <i>mpls-exp-value</i>	Sets the remark value of MPLS packets.
Qtech(config-traffic-behavior)# remark cos <i>cos-value</i>	Sets the 802.1Q remark value of Ethernet packet.
Qtech(config-traffic-behavior)# sub-traffic-policy <i>traffic-policy-name</i>	Sets a sub-traffic policy.
Qtech(config-traffic-behavior)# rtp-shaping delay 500 clock-rate 9000	Set TRP shaping behavior.

Configuration example:

Create a traffic behavior rule.

```
Qtech(config)#traffic behavior tb1
Qtech(config-traffic-behavior)#user-queue uq1 inbound
Qtech(config-traffic-behavior)#service-class ef color green
Qtech(config-traffic-behavior)#remark dscp 40
```

3.3.8 Configuring a Traffic Policy Rule

The traffic policy rule associates traffic classes and traffic behaviors, thus scheduling classified traffic according to user configurations. Configure this function as follows:

Command	Purpose
Qtech(config)# traffic policy <i>traffic-policy-name</i>	Enters/Creates a traffic policy.
Qtech(config-traffic-policy)# classifier <i>classifier-name</i> behavior <i>behavior-name</i> precedence <i>precedence-value</i>	Specifies a traffic behavior rule for a traffic class and sets the precedence. The smaller the value of the precedence, the higher the precedence is.

Configuration example:

Create a traffic policy.

```
Qtech(config)#traffic policy tp1
Qtech(config-traffic-policy)#classifier tc1 behavior tb1 precedence
1
```

3.3.9 Applying a Traffic Policy to an Interface

Configure this function as follows:

Command	Purpose
---------	---------

Qtech(config)# interface <i>interface-name</i>	Enters the interface configuration mode.
Qtech(config-if)# traffic-policy <i>traffic-policy-name</i> [inbound outbound] [link-layer all-layer]	To apply a traffic policy, the layer parameter needs to be specified. Only Layer 3 policies and MPLS take effect by default. The link-layer parameter specified takes effect to only 802.1P Layer 2 packets. The all-layer parameter specified takes effect to both Layer 2 and Layer 3 packets. The link-layer and all-layer parameters can be specified for main interfaces only. The parameters take effect to the main interface and its associated sub-interfaces after specified. They cannot be specified for sub-interfaces.

Configuration example:

Apply a traffic policy to an interface.

```
Qtech(config)#int gigabitethernet 0/1
Qtech(config-if-Gigabitethernet 0/1)#traffic-policy tp1 inbound
```

3.3.10 Configuring a CQ Template

Configuring an CQ template is to configure the scheduling mode (PQ, WFQ, and LPQ), traffic shaping value, and WRED parameters of eight kinds of PQs. PQ scheduling is performed on queues with the priority as ef, cs6, or cs7. WFQ scheduling is performed on queues with the priority as be, af1, af2, af3, or af4. Traffic shaping is not performed by default and the default discarding policy is tail discarding. You can configure multiple CQ templates and apply them to different interfaces. CQ scheduling takes effect on outbound traffic only.

Configure this function as follows:

Command	Purpose
Qtech(config)# port-queue <i>port-queue-template-name</i>	Enters/Creates a CQ template.
Qtech(config-port-queue)# queue <i>cos-value</i> {pq wfq weight <i>weight-value</i> lpq} [shaping <i>shaping-value</i>] [wred <i>wred-name</i>] [depth <i>depth-value</i>]	Sets CQ scheduling parameters.
Qtech(config-port-queue-template)# exit	Exits the CQ template configuration.

Configuration example:

Create a CQ template and set CQ scheduling parameters.

```
Qtech(config)#port-queue pqt1
Qtech(config-port-queue)# queue be lpq outbound
Qtech(config-port-queue)# queue af1 wfq weight 10 shaping 100000
wred pwt1
Qtech(config-port-queue)# queue cs7 pq shaping wred pwt1
Qtech(config-port-queue)#exit
```

3.3.11 Applying the CQ to an Interface

The CQ applied to an interface takes effect on outbound traffic only.

Configure this function as follows:

Command	Purpose
Qtech(config)# interface <i>interface-name</i>	Enters the interface configuration mode.
Qtech(config-if)# port-queue <i>port-queue-template-name</i> [shaping <i>shaping-value</i>]	Applies the CQ to an interface.

Configuration example:

Apply the CQ to an interface.

```
Qtech(config)#int gigabitethernet 0/1
```

```
Qtech(config-if-GigabitEthernet 0/1)#port-queue pqt1
```

3.3.12 Applying Resource Reservation Queue to Interface

The resource reservation queue applied to the interface is scheduled for outbound traffic only.

Run the following commands to configure this function:

Command	Purpose
Qtech(config)# interface <i>interface-name</i>	Enters the interface configuration mode.
Qtech(config-if)# port-res-queue <i>shaping-value</i> [<i>depth depth-value</i>]	Applies the resource reservation queue to the interface.

Configuration example:

#Apply the class queue to the interface.

```
Qtech(config)#int gigabitEthernet 0/1
Qtech(config-if-GigabitEthernet 0/1)# port-res-queue 5000 depth 300
```



Note The resource reservation queue takes effect only after the **port-queue** command is run on the interface.
The shaping value for the resource reservation queue is allocated from the shaping value configured in the **port-queue** command on the interface. The shaping value is automatically allocated only when there is a flow passing the resource reservation queue. When the flow stops for 3 minutes, the shaping value allocated to the resource reservation queue is invalidated.

3.3.13 Configuring HQoS Policy for AP Scenario

The bandwidth committed by the operator is configured for an AP member. The HQoS distributes load according to the physical link bandwidth corresponding to an AP member when scheduling traffic between AP members for routing. The physical bandwidths of a gigabit interface and a 10-gigabit interface are respectively 1000 Mbps and 10,000 Mbps. However, though the bandwidth allocated by the operator may be less than the committed bandwidth, the configuration is still performed according to the bandwidth committed by the operator. In this way, the HQoS schedules traffic for routing based on the actual available bandwidth of the egress, so that traffic on the operator links is controllable.

Run the following commands to configure this function:

Command	Purpose
Qtech(config)# interface <i>interface-name</i>	Enters the interface configuration mode.
Qtech(config-if)# port-bandwidth <i>bandwidth-value1</i>	Configures the maximum available bandwidth committed by the operator for the current AP member of traffic management.

Configuration example:

#Configure 100 Mbps available bandwidth on the link corresponding to Interface Gi0/0.

```
Qtech(config)#int GigabitEthernet 0/0
Qtech(config-if-GigabitEthernet 0/0)#port-bandwidth 100000
```



Note This command is available only on AP members.

An AP is configured to schedule traffic for routing. The AP is a logical port, and does not forward traffic. All traffic is forwarded by the physical member port. When the HQoS policy is applied to the AP, the HQoS policy schedules traffic between the AP members by the source or destination IP address for routing.

Run the following commands to configure this function:

Command	Purpose
Qtech(config)# interface AggregatePort-L3 <i>interface-name</i>	Enters the layer 3 AP configuration mode.
Qtech(config-if-AggregatePort-L3 1)# traffic-balance { <i>dst-ip</i> <i>src-ip</i> }	Schedules traffic between AP members by the source or destination IP address.

Configuration example:

#Perform traffic management on the layer 3 AP by the source IP address.

```
Qtech(config)#int AggregatePort-L3 1
Qtech(config-if-AggregatePort-L3 1)#traffic-balance src-ip
```



Note This command is available only on the AP.

3.3.14 Configuring a Static VoQ Credit Point

By default, the VoQ credit point of a device is calculated dynamically based on the processing capability of the device, FAP bandwidth, and flow control. This configuration is to specify a static credit point for the destination device, so that VoQ requests are not updated dynamically during VoQ scheduling.

Configure this function as follows:

Command	Purpose
Qtech(config)# credit { <i>device-id</i> <i>credit-value</i> }	Specifies the VoQ credit point for the destination device.

Configuration example:

Set the VoQ credit point of the destination device with the ID as 4 to 10,000.

```
Qtech(config)#credit 4 10000
```

3.3.15 Configuring System VoQ Scheduling

By default, system VoQ scheduling is disabled.

Configure this function as follows:

Command	Purpose
Qtech(config)# voq enable	Enables system VoQ scheduling.

Configuration example:

Enable system VoQ scheduling.

```
Qtech(config)#voq enable
```

3.3.16 Clearing Queue Statistics

Command	Function
clear port-queue statistics interface <i>interface-name</i>	Clears CQ statistics.
clear user-queue statistics <i>user-queue-name</i> { <i>inbound</i> <i>outbound</i> } <i>devid</i> <i>devid</i>	Clears UQ statistics.
clear user-group-queue statistics <i>user-group-queue-name</i> { <i>inbound</i> <i>outbound</i> } <i>devid</i> <i>devid</i>	Clears GQ statistics.
clear voq statistics <i>class-queue-priority</i> <i>devid</i> <i>devid</i>	Clears VoQ statistics.

3.3.17 Showing Configurations

Command	Function
---------	----------

Command	Function
show wred [<i>wred-name</i>]	Displays WRED configuration information on the system.
Show flow-queue [<i>flow-queue-name</i>]	Displays FQ configuration information
show user-queue statistics <i>user-group-queue-name</i> {inbound outbound} [devid <i>devid</i>]	Displays UQ statistics.
show user-group-queue statistics <i>user-group-queue-name</i> {inbound outbound} [devid <i>devid</i>]	Displays GQ statistics on the system.
show port-queue [<i>port-queue-name</i>]	Displays port-queue configuration information on the system.
show port-queue statistics [interface <i>interface</i>]	Displays port-queue statistics on the system.
show credit status [devid <i>devid</i>]	Displays credit system information.

Configuration example:

Display the port-queue information on a system interface.

```
Qtech# show port-queue interface gigabitethernet 1/1/1
[be]
  Pass:      42900556 packets,    2745666258 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          2073046
    balance,          0 token
[af1]
  Pass:      43401132 packets,    2608782540 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          8960
    balance,          0 token
[af2]
  Pass:      45091586 packets,    2707371120 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          2069592
    balance,          0 token
[af3]
  Pass:      43496828 packets,    2613966540 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          2092532
    balance,          0 token
[af4]
  Pass:      45170464 packets,    2711553720 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          2092532
    balance,          0 token
[ef]
  Pass:      45099831 packets,    2708775960 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          0
    balance,          0 token
[cs6]
  Pass:      46002386 packets,    2761254360 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          0
    balance,          0 token
[cs7]
  Pass:      41955096 packets,    2520579480 bytes
  Drop:      0 packets,          0 bytes
  Que :      0 packets,          0 bytes,          0
    balance,          0 token
```

3.4 Typical HQoS Configuration Examples

3.4.1 Configuration Example 1

3.4.1.1 Networking Requirement

- Device requirement

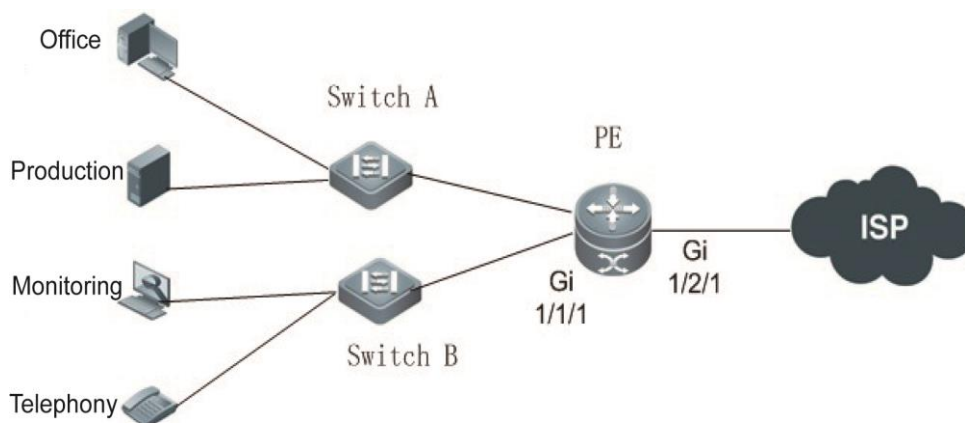
Two Ethernet switches and one router

- Configuration requirement

Each switch is connected to one user to provide two services. For user A, the CIR is 60 Mbit/s and peak bandwidth is 80 Mbit/s. For user B, the CIR is 30 Mbit/s and peak bandwidth is 40 Mbit/s. Both users are bound to a user group with the CIR as 90 Mbit/s and peak bandwidth as 120 Mbit/s. Four services are identified by four IP addresses, that is, 10.1.10.1, 10.1.10.2, 10.1.20.1, and 10.1.20.2.

3.4.1.2 Networking Topology

Figure 3 HQoS Access Networking



3.4.1.3 Configuration Points

Configure traffic classification rules to distinguish between services.

Configure traffic behavior rules to perform traffic shaping on UQ and GQ.

3.4.1.4 Configuration Steps

Configure traffic classification rules.

```

# Configure traffic behavior rule tc1.
Qtech(config)#access-list 100 permit ip 10.1.10.0 0.0.0.255 any
Qtech(config)#traffic classifier tc1
Qtech(config-traffic-classifier)#if-match acl 100
Qtech(config-traffic-classifier)#exit
# Configure traffic behavior rule tc2.
Qtech(config)#access-list 110 permit ip 10.1.20.0 0.0.0.255 any
Qtech(config)#traffic classifier tc2
Qtech(config-traffic-classifier)#if-match acl 110
Qtech(config-traffic-classifier)#exit
  
```

Configure an FQ WRED template.

```

Qtech(config)#wred-template wt1
  
```



```
Qtech(config-wred-template)#color green low-limit 40 high-limit 60
discard-percent 10
Qtech(config-wred-template)#color yellow low-limit 30 high-limit 50
discard-percent 10
Qtech(config-wred-template)#color red low-limit 20 high-limit 40
discard-percent 10
Qtech(config-wred-template)#exit
```

Configure FQ scheduling parameters.

```
Qtech(config)#flow-queue-template fqt1
Qtech(config-flow-queue-template)# queue be lpq
Qtech(config-flow-queue-template)# queue af1 wfq weight 10 shaping
100 flow-wred wt1
Qtech(config-flow-queue-template)# queue cs7 pq shaping shaping-
percentage 20 flow-wred wt1
Qtech(config-flow-queue-template)#exit
```

Configure an FQ mapping template.

```
Qtech(config)#flow-mapping-template fmt1
Qtech(config-flow-mapping-template)# map flow-queue af1 to port-
queue ef
```

Configure a GQ.

```
Qtech(config)#user-group-queue ugq1 inbound
Qtech(config-user-group-queue)#shaping 120000
Qtech(config-user-group-queue)#exit
```

Configure a UQ.

```
# Configure UQ uq1.
Qtech(config)#user-queue uq1 inbound
Qtech(config-user-queue)#cir 60000 pir 80000
Qtech(config-user-queue)#flow-queue-template fqt1
Qtech(config-user-queue)#flow-mapping-template fmt1
Qtech(config-user-queue)#user-group-queue ugq1
Qtech(config-user-queue)#exit
# Configure UQ uq2.
Qtech(config)#user-queue uq2 inbound
Qtech(config-user-queue)#cir 30000 pir 40000
Qtech(config-user-queue)#flow-queue-template fqt1
Qtech(config-user-queue)#user-group-queue ugq1
Qtech(config-user-queue)#exit
```

Configure traffic behavior rules.

```
# Configure traffic behavior rule tb1.
Qtech(config)#traffic behavior tb1
Qtech(config-traffic-behavior)#user-queue uq1 inbound
Qtech(config-traffic-behavior)#exit
# Configure traffic behavior rule tb2.
Qtech(config)#traffic behavior tb2
Qtech(config-traffic-behavior)#user-queue uq2 inbound
Qtech(config-traffic-behavior)#exit
```

Configure a traffic policy.

```
Qtech(config)#traffic policy tp1
Qtech(config-traffic-policy)#classifier tc1 behavior tb1
Qtech(config-traffic-policy)#classifier tc2 behavior tb2
Qtech(config-traffic-policy)#exit
```

Apply the policy to an interface.

```
Qtech(config)#int gigabitethernet 1/1/1
```

```
Qtech(config-if-GigabitEthernet1/1/1)#traffic-policy tp1 inbound
```

Configure a CQ WRED template.

```
Qtech(config)# port-wred-template pwt1
Qtech(config-port-wred-template)#color green low-limit 40 high-limit
60 discard-percent 10
Qtech(config-port-wred-template)#color yellow low-limit 30 high-
limit 50 discard-percent 10
Qtech(config-port-wred-template)#color red low-limit 20 high-limit
40 discard-percent 10
Qtech(config-port-wred-template)#exit
```

Configure a CQ.

```
Qtech(config)#port-queue-template pqt1
Qtech(config-port-queue-template)# queue be lpq outbound
Qtech(config-port-queue-template)# queue af1 wfq weight 10 shaping
100000000 port-wred pwt1
Qtech(config-port-queue-template)# queue cs7 pq shaping shaping-
percentage 20 port-wred pwt1
Qtech(config-port-queue-template)#exit
```

Apply the CQ to an interface.

```
Qtech(config)#int gigabitEthernet 1/2/1
Qtech(config-if-GigabitEthernet 1/2/1)#port-queue pqt1 outbound
```

3.4.1.5 Verification

Display traffic policy statistics on an interface.

```
Qtech#show user-queue statistics uq1 inbound devid 4
Qtech#show user-queue statistics uq2 inbound devid 4
```

Display CQ statistics on an interface.

```
Qtech# show port-queue interface gigabitEthernet 1/2/1
```

3.4.2 Configuration Example 2

3.4.2.1 Networking Requirement

- Device requirement

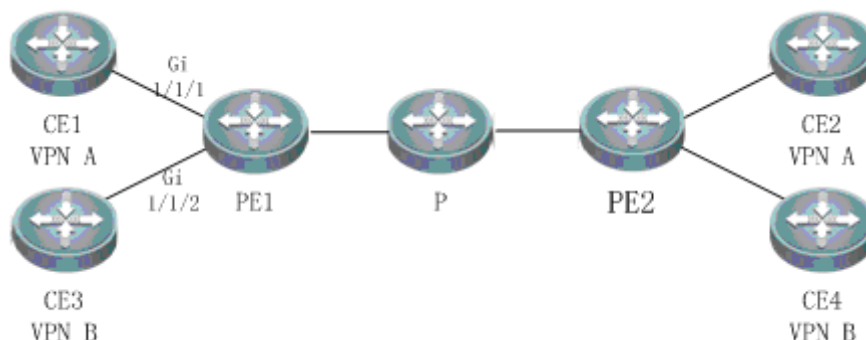
Seven routers

- Configuration requirement

Three routers constitute an MPLS backbone network. Two routers provide user access for VPN A, and two routers provide users access for VPN B. On each of the networks connected to CE1 and CE3, there is a multicast source with the multicast address as 232.0.0.1. For CE1, the CIR is 60 Mbit/s and peak bandwidth is 80 Mbit/s. For CE2, the CIR is 30 Mbit/s and peak bandwidth is 40 Mbit/s.

3.4.2.2 Networking Topology

Figure 4 HQoS Multicast Networking



3.4.2.3 Configuration Points

Configure traffic classification rules to distinguish between multicast addresses.

Configure traffic behavior rules to restrict the bandwidth of uplink multicast UQ.

3.4.2.4 Configuration Steps

Configure traffic classification rules.

```
# Configure traffic classification rule tc1.
Qtech(config)#access-list 100 permit ip any host 232.0.0.1
Qtech(config)#traffic classifier tc1
Qtech(config-traffic-classifier)#if-match acl 100
Qtech(config-traffic-classifier)#exit
# Configure traffic classification rule tc2.
Qtech(config)#traffic classifier tc2
Qtech(config-traffic-classifier)#if-match acl 100
Qtech(config-traffic-classifier)#exit
```

Configure a UQ.

```
# Configure UQ uq1.
Qtech(config)#user-queue uq1 inbound
Qtech(config-user-queue)#cir 60000 pir 80000
Qtech(config-user-queue)#exit
# Configure UQ uq2.
Qtech(config)#user-queue uq2 inbound
Qtech(config-user-queue)#cir 30000 pir 40000
Qtech(config-user-queue)#exit
```

Configure traffic behavior rules.

```
# Configure traffic behavior rule tb1.
Qtech(config)#traffic behavior tb1
Qtech(config-traffic-behavior)#user-queue uq1 inbound
Qtech(config-traffic-behavior)#exit
# Configure traffic behavior rule tb2.
Qtech(config)#traffic behavior tb2
Qtech(config-traffic-behavior)#user-queue uq2 inbound
Qtech(config-traffic-behavior)#exit
```

Configure a traffic policy.

```
Qtech(config)#traffic policy tp1
Qtech(config-traffic-policy)#classifier tc1 behavior tb1
Qtech(config-traffic-policy)#exit

Qtech(config)#traffic policy tp2
Qtech(config-traffic-policy)#classifier tc2 behavior tb2
Qtech(config-traffic-policy)#exit
```

Apply the policy to an interface.

```
Qtech(config)#int gigabitethernet 1/1/1
Qtech(config-if-GigabitEthernet1/1/1)#traffic-policy tp1 inbound

Qtech(config)#int gigabitethernet 1/1/2
Qtech(config-if-GigabitEthernet1/1/2)#traffic-policy tp2 inbound
```

3.4.2.5 Verification

Display traffic policy statistics on an interface.

```
Qtech# show traffic policy tp1
Qtech# show traffic policy t
```

3.4.3 VDA Configuration Example

3.4.3.1 Networking Requirements

- Device requirement

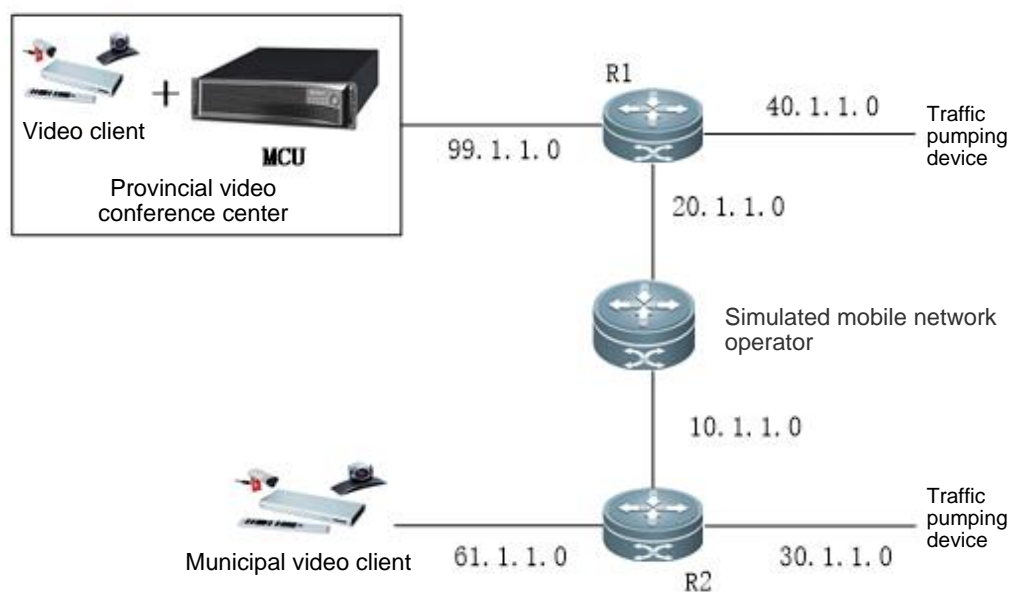
Three routers are required

- Configuration requirement

The provincial video conference center and the municipal video conference center are simulated. These two centers are connected via a 20 mbps dedicated line, to provide the video conference service and other production and office services. It is required to reserve bandwidth for a video conference and isolate the video conference service from other services. After the video conference ends, the reserved bandwidth is automatically released. Configurations in the two centers are the same, only the municipal egress device is described.

3.4.3.2 Network Topology

Figure 5 VDA Application



3.4.3.3 Configuration Key Points

1. Configure flow identification for the resource reservation queue.
2. Configure the resource reservation queue on an interface.

3.4.3.4 Configuration Steps

1. Configure the traffic classifier.

#Configure the traffic classifier for matching based on the interface and any ACL.

```
ip access-list extended 101
 10 permit ip any any
traffic classifier tc or
if-match acl 101
```

2. Configure the traffic behavior.

#Configure the traffic behavior.

```
traffic behavior tb
port-res-queue
```

3. Configure the traffic policy.

#Configure the traffic policy and the traffic behavior.

```
traffic policy tp
classifier tc behavior tb precedence 1
```

4. Apply the traffic policy to an interface.

#Interface 0/0/2 is directly connected to the municipal conference center. Apply the traffic policy to this interface.

```
interface GigabitEthernet 0/0/2
ip address 61.1.1.1 255.255.255.0
traffic-policy tp inbound
```

5. Configure the resource reservation queue on the outbound interface.

#Connect Interface 0/0/1 (egress of the municipal network) to the operator device. Configure the resource reservation queue on this interface.

```
interface GigabitEthernet 0/0/1
ip address 10.1.1.2 255.255.255.0
port-queue pq shaping 20000
port-res-queue 12000
```

3.4.3.5 Displaying

Display the user queue statistics on the device.

```
Qtech# show port-queue statistics interface gi 0/0/1
```

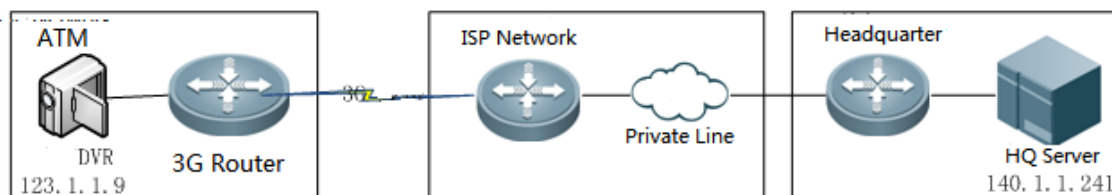
3.5 Examples for Configuring RTP Traffic Shaping

Networking Requirements

1. Off-bank ATMs adopt Qtech routers as access routers that are connected to servers in headquarters via 3G circuit.
2. Monitoring videos are transmitted between routers of branches and the servers of the headquarters by application of TCP and converged by RSR77 in the headquarters.
3. Data flows are not encrypted or encapsulated by other tunnels before reaching the routers adopting optimizing strategies.

Network Topology

Figure 6 RTP Shaping Application Network Topology



Configuration Tips

1. Turn on WAN-TA function.
2. Identify characteristics of data flows to realize RTP business.
3. Configure RTP shaping behavior.

Configuration Steps

1. Transmitted optimization should be turned on when RSR77 of the headquarters are configured.

Configure video transmitted optimization to transit monitoring videos between routers and servers. ACL101 configures RTP data flows from the branches to the headquarters. With video data flows of off-bank ATMs reaching the servers in the headquarters, the transmitted optimization of convergence routers is enabled. And wan-ta policy is applied on the in-coming interface gigabitEthernet2/1/1.

```
Qtech#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Qtech(config)#ip access-list extended 101
Qtech(config-ext-nacl)# 10 permit tcp any eq 554 any
Qtech(config)#wan-ta enable
Qtech(config)# wan-ta policy video
Qtech(config-if-GigabitEthernet 2/1/1)#wan-ta-policy video list 101
```

2. RTP Traffic Shaping to RSR77 of the Headquarters

Configuring RTP Traffic Shaping to RSR77 of the Headquarters

```
Qtech#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Qtech(config)#
Qtech(config)#traffic classifier tc1 or
Qtech(config-traffic-classifier)# if-match acl 101
Qtech(config-traffic-classifier)#
Qtech(config-traffic-classifier)#traffic behavior tb1
Qtech(config-traffic-behavior)# rtp-shaping delay 500 clock-rate
89950
Qtech(config)#traffic policy tp1
Qtech(config-traffic-policy)# classifier tc1 behavior tb1 precedence
1
Qtech(config-if-GigabitEthernet 2/1/1)#traffic-policy tp1 inbound
```

Verification

1. Displaying WAN-TA Policy Configuration

```
Qtech#show wan-ta policy video
wan-ta policy: video
  Congestion Control : low-bandwidth-delay
  SACK Support: TRUE
  Initial Congest Window: 10 MSS
  Maxitum Segment Size: 1460
  Keepalvie Interval(retry): 120(9)

apply on interfaces:
interface name          list
GigabitEthernet 2/1/1  101
```

2. Displaying RTP-shaping Statistics

```
Qtech#show rtp-shaping statistics
hqos_rtp_dump[185566fc][1]
rtp-que[184bdec0]:
  delay[1000],clock_rate[89950]
  flow[10.255.255.91,10.255.255.213,6,554,49419]
```

```
que-  
pkts[0],pass_pkts[6050],drop_pkts[0],disorder_pkts[0],none_video  
_pkts[5692]  
last_rtp_time[3933490296],last_sys_time[2267149741]
```