QoS Configuration

Table of Contents

Chapter 1 Configuring QoS

1.1 Overview

This section explains what Quality of Service (QoS) is and the service model that realizes the service quality. It also introduces various queuing algorithms of QoS.

1.1.1 Concept of QoS

QoS means the capability of a network that uses various basic technologies to provide better service for selected network communications. These basic technologies include: Frame Relay (FR), Asynchronous Transfer Mode (ATM), Ethernet and 802.1 network, and IP-route network. To guarantee the QoS of these networks, the router provides several functions such as queuing, scheduling and QoS signaling technology. In particular, by using technologies that support dedicated bandwidth and avoiding and managing network congestion, our routers provide better and more predictable network service.

1.1.2 End-to-End QoS Model

The service model describes the QoS capability of an End-to-End group, i.e. the capability of a network that provides services requested by special network communications from one end to the other. QoS software supports three kinds of service models: Best-Effort Service, Integrated Service and Differentiated Service.

1. Best-Effort service

Best-Effort service is a simple service model. Under this kind of service model, applications can send any volume of data at any time without applying for permission or notifying the network in advance. For Best-Effort service, if conditions permit, the network can transfer data without any guarantee in the aspects of reliability, time delay range or throughput. The router’s QoS function for Best-Effort service adopts the FIFO (first in, first out) queuing.

2. Integrated service

Integrated service is a multiple service model, and it can handle various QoS requirements. In this model, applications have to apply to the network for a special kind of service by QoS signaling before sending data. This request is mainly for the network to notify the brief communication situations of this application, and apply for a special kind of service, which can fulfill its bandwidth and time delay requirements. Only after getting confirmation messages from the network can this application send data. Meanwhile, the data sent by it must comply with the previously described communication situations.

When conducting intelligent queuing operations for various streams, as long as the communication volume maintains the requested range, the network can meet the QoS requirement of this application.

The QoS of router software provides Controlled Load Service and Guaranteed Rate Service by using the Resource Reservation Protocol (RSVP). Controlled Load Service allows the application to maintain low time delay and high throughput operations even when the network is congested. To realize this target, router QoS provides Weighted Fair Queuing (WFQ).

3. Differentiated service

The difference between the Differentiated Service and the Integrated Service model is that applications using Differentiated Service don't have to definitely send a signal to the router before sending data.

For Differentiated Service, if the network is going to send a special kind of service, it must specify the corresponding QoS mark in every data packet. This kind of specification can be realized in several ways, for example, using the IP priority bit in the IP data packet. The router uses this QoS specification to make classification, and complete the intelligent queuing task. The Weighted Random Early Detection (WRED), Custom Queuing (CQ), and Priority Queuing (PQ) provided by the router QoS can be used to send differentiated service.

1.1.3 Various Queuing Algorithms of QoS

The various queuing algorithms of QoS are the important guarantee for realizing QoS. Our routers provide Weighted Fair Queuing (WFQ), Custom Queuing (CQ), Priority Queuing (PQ) for bandwidth assignment; Weighted Random Early Detection (WRED) for congestion control; and the simplest FIFO algorithm.

1. Weighted Fair Queueing

WFQ is an automatic scheduling method, and it provides a fair bandwidth assigning solution for all network communications. WFQ applies priority or weight to specified communications to divide them into different sessions, and determines the bandwidth of every session as compared with other sessions. WFQ is an algorithm based on data stream, it not only moves communications with high priority to the front of the queue to reduce response time, but also distributes the remaining bandwidth among high bandwidth data streams fairly. In other words, WFQ assigns higher priority to low-volume traffic(for example, Telnet session) than high-volume traffic(for example, FTP session). WFQ provides balanced link volume assignment for concurrent file transfers; i.e. when there are several simultaneous file transfer tasks , it can provide equal bandwidth for these data transfer tasks.

WFQ overcomes the serious limitation of the FIFO queuing method. When the FIFO queuing method is in effect, communications are sent out according to the sequence in which they reach the interface without considering their bandwidth consumption or relative time delay. As a result, file transfer or other large data volume network applications can take almost all the available bandwidth, depriving the bandwidth of other communications. WFQ dynamically divides the communication into messages that constitute sessions in order to distribute the bandwidth fairly among different sessions, assuring that low-volume traffic can be transferred in a timely method.

WFQ can divide the communication into different data streams based on the addressing of the data packet head. Because most communications are IP data, it classifies the data packet according to various fields of IP head (Source and Target addresses, Source and Target ports, protocol type and TOS).

Before sending, WFQ puts the data packet of various sessions into a fair queue. The sequence in which data packets leave the fair queue is determined by the finish number indicated by the last bit of every arriving data packet.

For almost every serial interface whose speed is set to 2.048Mbps or lower, WFQ is the default queuing method.

For detailed information, see WFQ configuration.

The normal weighted fair queuing can only automatically identify streams, but can't provide specified services to some special streams as required. Our routers provide category-based weighted fair queuing (CBWFQ), and this algorithm enhances the normal weighted fair queuing algorithm. It can identify a stream of some category defined by users, and assign a specified weight to this stream.

2. Weighted Random Early Detection

Random Early Detection (RED) technology is a generally used congestion avoidance mechanism. Once RED is configured, routers use RED to control dropped message packets. If you don't configure Weighted Random Early Detection (WRED), routers will use a rough message packet dropping technology called Tail Drop by default. (Weighted Random Early Detection (WRED) doesn't guarantee the bandwidth ratio of a specified stream.)

Tail Drop treats all communications equally, and it doesn't distinguish the service level. When congestion occurs, some packets will accumulate in the sending queue. If the sending queue is full, message packets will begin to be dropped until congestion disappears and sending queue is full no more. When using RED to resolve the congestion on routers, the globalization of this problem can be avoided. Global Synchronization occurs at the peak of congestion, many hosts using TCP protocol to decrease their transfer speeds to handle dropped message packets, and increase their transfer speeds again when congestion mitigates, but the transfer link is not fully used during this process.

The basic assumption of RED technology is that most applications are sensitive to the packet loss during communication, and if some message packets are dropped, the sending speed will be reduced temporarily. TCP is used to handle dropped message packets accurately – or even soundly, i.e. effectively assuring that the effectiveness of the packet dropping technology is the same as that of congestion avoidance signaling technology.

The target of RED is achieved by controlling average queue length to indicate terminal hosts to reduce their message packet transfer speed temporarily when required. RED indicates message packet source to reduce its transfer speed by dropping packets randomly before the peak congestion period comes. If packet source uses TCP, it will reduce its sending speed, until all packets arrive at their destinations and the congestion is cleared. You can use RED to call TCP in order to reduce the transfer speed of message packets. TCP can not only pause sending, but also restart quickly, which adjusts the transfer speed to the speed supported by the network.

When average queue length exceeds the minimum threshold, RED begins to drop packets. The packet dropping speed is increased linearly as average queue length increases until the average queue length reaches the minimum threshold and packets stop being dropped. When average queue length exceeds the maximum threshold, all packets will be dropped. The minimum threshold must be set high enough in order to maximize the utilization of the transfer link. If the minimum threshold is set too low, packets will be dropped unnecessarily, and the transfer link will not be utilized fully. The interval between the maximum and the minimum thresholds must be set large enough to avoid Global Synchronization. If the interval is too small, many packets will be dropped immediately which results in Global Synchronization.

For detailed information, see WRED configuration.

3. Custom Queueing

When Custom Queuing is in effect on a port, the system maintains 17 output queues for this port. You can specify queue No.1 to No.16. Configurable number of bytes and packet types concerned with every output queue. Configurable number of bytes means the number of bytes in the current queue that the system should send before moving to the next queue.

Queue No.0 is a system queue; before Queue No.1 to No.16 can be handled, Queue No.0 must be emptied first. The system puts high priority packets, such as keep-alive packets and signaling packets, in this queue. Other communications cannot use this queue.

For Queues No.1 to No.16, the system polls these queues, gets the configured number of bytes from the current queue, and then sends these packets before moving to the next queue. When handling a queue, the system keeps on sending packets until the number of sent bytes exceeds the total number of bytes of this queue or this queue is emptied. The bandwidth used by a queue can only be specified by the total number of bytes indirectly. Like PQ, CQ is configured statically, so it can’t adapt to the continuously changing network conditions automatically.

For detailed information, see CQ configuration.

4. Priority Queueing

Priority Queuing has four kinds of communication priorities (high, middle, normal, low). During data transfer, Priority Queuing handles the high priority queues absolutely before the low priority queues; as long as important communications are given the highest priorities, they are always serviced before the communications with lower priorities.

According the criteria defined by users, packets are classified and put into one of the four output queues; packets whose priorities have not been defined will be put into the Normal queue. When the system is going to send out packets from an interface, it begins to scan priority queues on this interface from the highest level to the lowest level. It scans the high priority queues first, and then scans the middle priority queues, and so on. Then, it selects the first packet in a queue with the highest priority, and begins transmission. Every time when it is going to send a packet, it repeats this process.

For detailed information, see PQ configuration.

1.1.4 Signaling of QoS

The signaling function of our router QoS allows end points or network nodes to send signals to other nodes in order to apply for the special processing of some communication. The signaling function of QoS can assist the QoS with scheduling, and configure successful, comprehensive End-to-End QoS service for the entire network. The signaling function of QoS uses IP protocol. Both the inside (IP priority) and the outside (RSVP protocol) signaling functions indicate that any specified communication type has the chance to get some kind of QoS service. IP priority and RSVP protocol provides a stable union to the End-to-End QoS signaling function: IP priority signal is used in the differentiated QoS, and RSVP is used in the guaranteed QoS.

For more comprehensive information, see RSVP documents.

1.1.5 QoS Link Efficiency Mechanism

To avoid the unnecessary waste of available bandwidth, our routers adopt RTP header compression (CRTP). For detailed information, see CRTP documents.

1.2 QoS Configuration Tast List

- Configure Weighted Fair Queuing (WFQ)
- Configure the policy-map used by an interface (CBWFQ)
- Configure Weighted Random Early Detection (WRED)
- Configure Custom Queuing (CQ)
- Configure Priority Queuing (PQ)
- Display the situation of interface queue
- Display the configuration of custom list
- Display the configuration of priority list
- Display the configuration of class-map
- Display the configuration of policy-map

1.3 QoS Configuration Tast

To make QoS configuration, you need to configure the queuing algorithm on interface, QoS signaling, and QoS link efficiency mechanism. The last two are optional, and the first configuration has a default value on every physical interface, and can be changed according to practical situations.

1.3.1 Configuring Weighted Fair Queuing (WFQ)

If you need to configure fair queuing on an interface, you can use the following commands in the interface configuration mode after specifying the interface:

command	Purpose
fair-queue	Apply fair queuing policy to an interface.

Notes:

On interfaces whose speed is 2.048Mbps or lower, WFQ is the default queuing method. WFQ is not available for Encapsulation LAPB or X.25 interfaces.

1.3.2 Configuring the Policy-Map Used by an Interface (CBWFQ)

Configuring a policy-map on an interface makes CBWFQ available on this interface. If you need to configure a policy-map on an interface, you can use the following command in the interface configuration mode after specifying the interface:

command	Purpose
service-policy *policy-name*	Allow the interface to use a policy-map.

Notes:

This command is only available on interfaces that have been configured WFQ algorithm.

1. Configure Policy-Map

Configuring policy-map and its class-map can specify a different class of stream. When an interface uses this policy-map, it can get some level of service quality guarantee based on the specified class.

To configure a policy-map, first, you need to use the following command in the global mode to enter the policy-map configuration mode:

command	Purpose
policy-map *policy-name*	Configure a policy-map and enter the policy-map configuration mode.

After entering the policy-map configuration mode, you can configure the class-map name used by the current policy-map, the bandwidth and the queue's upper limit. To make these configurations, you can use the following command in the policy-map configuration mode:

command	Purpose
class *class-name* **bandwidth** *bandwidth*(*kbps*) [**queue-limit** *packet-number*]	Configure the bandwidth used by a class-map and its queue upper limit in the current policy-map.

2. Configure Class-Map

Configuring class-map can define some class of stream. When the policy-map used by an interface contains this class-map, it can get some level of service quality guarantee based on the specified class.

To configure the class-map, you can use the following commands in the global configuration mode:

command	Purpose
class-map *class-name* **match protocol** *protocol-type*	Configure a class-map classified by protocol.
class-map *class-name* **match interface** *interface-type interface-number*	Configure a class-map classified by interface.
class-map *class-name* **match access-group** *list-name*	Configure a class-map classified by visit list.

1.3.3 Configuring Weighted Random Early Detection (WRED)

If you need to configure weighted random early detection on an interface, you can use the following command in the interface configuration mode after specifying the interface:

command	Purpose
random-detect	Apply weighted random early detection to an interface.

1.3.4 Configuring Custom Queuing (CQ)

If you need to configure custom queuing on an interface, you can use the following command in the interface configuration mode after specifying the interface:

command	Purpose
custom-queue-list *list-number*	Apply CQ algorithm to this interface, parameter “list-number” is the custom list number that is used. Its range is 1-16, without any default value.

1.3.5 Configuring Priority Queuing (PQ)

If you need to configure priority queuing on an interface, you can use the following command in the interface configuration mode after specifying the interface:

command	Purpose
priority-group *list-number*	Apply PQ algorithm to this interface, parameter “list-number” is the custom list number that is used. Its range is 1-16, without any default value.

1.3.6 Displaying QoS

1. Display the situation of interface queue

To display the situation of an interface queue, you can use the following command:

command	Purpose
show queue interface-type *interface-number*	Display the information of queues on this interface.

2. Display the configuration of custom list

To display the configuration of a custom list, you can use the following command:

command	Purpose
show queueing *custom*	Display the configuration of custom list.

3. Display the configuration of priority list

To display the configuration of a priority list, you can use the following command:

command	Purpose
show queueing *priority*	Display the configuration of priority list.

4. Display the configuration of class-map

To display the configuration of a class-map, you can use the following command:

command	Purpose
show class-map [*class-name*]	Display the configuration of class-map.

5. Display the configuration of policy-map

To display the configuration of a policy-map, you can use the following command:

command	Purpose
show policy-map [*policy-name*]	Display the configuration of policy-map.

1.4 QoS Configuration Solution

If the user has four kinds of applications (A, B, C and D), their expected bandwidth ratio is 10/20/40/30 abd their packet's lengths are 1428/582/371/1525 respectively. Each time the queue receives the service, the number of the bytes that the system sends is N times of the packet's length. Hence, the packet's length must be considered when the total number of bytes is configured. It is not to simply set the value to 100/200/400/300, which results in the real bandwidth allocation, 1428/582/371/1525. To get the best result, perform the following steps:

1. Divide an expecting-to-allocated bandwidth percentage by the size of a packet. In the following example, the ratio should be 10/1428, 20/582, 40/371 and 30/1525, or be 0.007, 0.03436, 0.10782 and 0.01967.

2. Then divide the previous four numbers by the smallest number among them. In this case, the smallest number is 0.007. This result is the ratio of the number of must-be-sent packets.

3. The end number of any ratio value means that the system has to transmit an extra packet. Hence, the final ratio in reality should be 1:5:16:3, which is the total number of to-be-sent packets.

4. Multiply the total number of packets with the corresponding packet's size to transfer the ratio into the total number of bytes.

5. To decide the bandwidth allocation that the ratio stands for, first decide the number of the transmitted bytes when one-for-all service is provided to three

queues, and then decide the percentage of the total bytes that each queue transmits.

The detailed configuration is shown as follows:

First allocate the four application to the corresponding queues:

queue-list 1 p ip 2 udp 100

queue-list 1 p ip 3 udp 200

queue-list 1 p ip 4 udp 400

queue-list 1 p ip 5 udp 700

6. Configure the to-be-transmitted byte number for each queue.

queue-list 1 queue 2 byte-count 1428

queue-list 1 queue 3 byte-count 2910

queue-list 1 queue 4 byte-count 5936

queue-list 1 queue 5 byte-count 4575

7. Configure the user list on the interface.

interface s1/0

custom-queue-list 1

1.5 Example of QoS configuration

If a user has four kinds of applications A, B, C and D, their expected bandwidth ratios are 10/20/40/30, and packet lengths are 1428/582/371/1525 respectively. Every time this queue is serviced, the number of bytes that the system sends is an integral multiple of packet length. So, you must consider the length of packet when you configure the total number of bytes, and you should not simply set them to 100/200/400/300; if you do this, the actual bandwidth assignment will be 1428/582/371/1525. To achieve ideal results, please follow the procedures hereafter:

1. For every queue, divide the bandwidth percent you want to assign to this queue by the packet size (in bytes). In this example, the ratio should be: 10/1428, 20/582, 40/371, 30/1525 or 0.007, 0.03436, 0.10782, 0.01967.

2. Use the minimum number to normalize the four numbers listed above: 1, 4.9, 15.4, 2.8, and this result is the ratio of packet number that must be sent.

3. The mantissa of any ratio value means that the system needs to send an additional packet. Round up the ratio value to get an integer, which is the total number of packets that should be actually sent. In this example, the actual ratio of sent packets will be 1:5:16:3.

4. Multiply the total number of packets of every protocol by the corresponding packet size in order to convert the packet number ratio into the total number of bytes. In this example, the total number of sent packets is 1 1428-byte packet, 5 582-byte packets, 16 371-byte packets and 3 1525-byte packets; i.e. send out 1428, 2910, 5936 and 4575 bytes from these queues respectively. This is the total number of bytes that you will specify in your custom queue configuration.

5. To determine the bandwidth assignment represented by this ratio, you should first calculate the total number of bytes sent by the system every time it services

these four queues: (1×1428) + (5×582) + (16×371) + (3×1525) = 1428+2910+5936+4575 = 14849. Then, you can determine the percent of the total number of bytes sent by every queue: 1428/14849, 2910/14849, 5936/14849, 4575/14849 = 9.6%, 19.5%, 39.8%, and 30.8%. As you can see, these 4 percentages are very close to our expected ratio 10/20/40/30.

6. If the actual bandwidth ratio is not close to the expected bandwidth ratio, you can multiply the original ratio by an optimal value in order to make the actual bandwidth ratio as close to these 4 integers as it can. NOTICE, the multiplier that you use doesn't have to be an integer.

The detailed configuration is listed below: (suppose that these 4 kinds of applications' UDP ports are 100, 200, 400, and 700 respectively; and they use custom list No.1)

1. Assign the corresponding queues for these 4 kinds of applications (2, 3, 4, 5)

```
queue-list 1 p ip 2 udp 100
queue-list 1 p ip 3 udp 200
queue-list 1 p ip 4 udp 400
queue-list 1 p ip 5 udp 700
```

2. Specify the sending byte number of every queue

```
queue-list 1 queue 2 byte-count 1428
queue-list 1 queue 3 byte-count 2910
queue-list 1 queue 4 byte-count 5936
queue-list 1 queue 5 byte-count 4575
```

3. Assign this custom list to the interface

```
interface s1/0
custom-queue-list 1
```

Chapter 2 Rate-Limit Configuration

2.1 Rate-Limit Overview

Rate-limit is used to designate the access rate.It can limit the output or input rate of a port and also the rate of all flows that complies with the access control list.The following kinds of ports are supported: Ethernet ports, E1/T1 ports and synchronized serial interfaces.

2.2 Setting the Rate-Limit

2.2.1 Configuring the Rate-Limit

Command	Purpose
rate-limit {**input** | **output**} {**all** | **access-group** *name*} *bps*	Sets the rate-limit.

Note:

1) This command is used for interface configuration.

2) Run **no Rate-limit {input | output} {all | access-group *name*}** to delete the rate limit on a port.

3) On each port you can set up to 8 output/input rate-limits.

2.2.2 Displaying the Rate-Limit Information

Command	Purpose
Show rate-limit i*nterface-type interface-number*	Displays the rate-limit information.

2.3 Configuration Example

The following example shows how to set an 8000000bps FTP rate-limit on an Ethernet port of a router:

```
interface FastEthernet1/1
ip address 10.0.0.1 255.255.255.0
rate-limit input  access-group  aaa  8000000
!
ip access-list extended aaa
  permit tcp any any eq ftp
```

The following example shows how to set a 1000000bps input rate-limit on the E1 port of a router:

```
controller E1 1/0
 unframed
!
interface Serial1/0:0
 ip address 10.0.0.2 255.255.255.0
 encapsulation ppp
 rate-limit input    all 1000000
```