Bandwidth Reservation System

bintec-Dm 715

Legal Notice

Warranty

This publication is subject to change.

bintec offers no warranty whatsoever for information contained in this manual.

bintec is not liable for any direct, indirect, collateral, consequential or any other damage connected to the delivery, supply or use of this manual.

Table of Contents

I Related Documents

bintec Dm717-I Bridge

bintec Dm719-I IP Tunnel

bintec Dm739-I IPSec

bintec Dm752-I Access Control

bintec Dm754-I NSLA

bintec Dm808-I IPv6 Access Control

bintec Dm750-I Ethernet Subinterface

Chapter 1 Introduction

1.1 Bandwidth Reservation System

The Bandwidth Reservation System (BRS) allows you to apply Quality of Service (QoS) features to the device output interfaces. More specifically, BRS integrates the following functions:

- Traffic classification.
- Traffic marking.
- Bandwidth distribution.
- Prioritizing.
- Traffic shaping.

Bandwidth reservation is a feature that runs over the following types of data links:

- Frame Relay.
- X.25 Lines.
- PPP Lines.
- HDLC Lines.
- ATM Subinterfaces.
- Ethernet Interfaces.
- Ethernet Subinterfaces.
- Wireless LAN Interfaces.
- TNIP Interfaces.
- BVI Interfaces.

Note

When you configure a PPP over an ATM subinterface, BRS must be configured in PPP and not in the ATM subinterface. Conversely, if the ATM subinterface directly encapsulates IP, BRS must be configured in the ATM subinterface.

The following sections focus on different features of the Bandwidth Reservation System.

1.2 Classifying Traffic

The Bandwidth Reservation System is responsible for separating data flows and applying different Quality of Service policies to them, prioritizing, balancing, limiting and marking each type of traffic in accordance with the configured criteria. This requires identifying each type of traffic and classifying it accordingly. Said classification can be performed using the following mechanisms:

- Specific filters.
- Access lists.
- Specific criteria.
- Through protocol.

Traffic not classified through any of these criteria is separated depending on whether it is control traffic and has been generated locally, sending it to either the control, local or default class.

1.2.1 Filters for Classifying Traffic

Using bandwidth reservation, you can assign the following filters (by entering **assign**) for specific types of traffic:

- TUNNELING-IP
- SDLC-IP
- RLOGIN-IP
- TELNET-IP
- NETBIOS

- SNA
- SNMP-IP
- MULTICAST-IP
- DLSW-IP
- XOT-IP

You can also assign tags to filter MAC frames (by preconfiguring the MAC filtering feature, assigning tags corresponding to a MAC filter):

- TAG1
- TAG2
- TAG3
- TAG4
- TAG5

Filters and Tags for IP Multicast Addressing and MAC Addressing

The router handles MAC address filtering through joint action between bandwidth reservation and MAC Filtering (MCF) using tags. For example, a user with bandwidth reservation can classify bridge traffic by tagging it.

This type of classification is supported when the bridge interfaces are ATM, Frame Relay, PPP and IP Tunnel.

You assign tags by creating a filter in the MAC filtering configuration process and then assigning a tag to it. This tag is then used to set up a bandwidth class for all packets associated with it. Tag values must range from 1 to 64.

Note

Tags can only be applied to bridged packets, and ONLY the packet's MAC Address fields can be used to generate them. Up to five tagged MAC filters (1 to 5) can be set. TAG1 is searched for first, then TAG2, and so on up to TAG5. A single MAC filter tag can consist of any number of MAC Addresses set in MCF.

Once a tagged filter has been created in the MAC filtering configuration process, it is assigned a class and priority in the bandwidth reservation configuration process. The **tag** command is then used in the bandwidth reservation process to reference said tag.

Tags can also refer to *groups*, as in the IP Tunnel example. IP Tunnel endpoints can belong to any number of groups. Packets are assigned to a particular group through the MAC filtering tagging feature.

To apply bandwidth reservation and queuing priority to tagged packets, follow these steps:

(1) Use MAC filtering configuration commands at the *Filter Config*> prompt to set up tags for packets passing through the bridge.

(2) Use the bandwidth reservation **tag** command to reference a tag for bandwidth reservation.

(3) Specify a class name for a tag through the bandwidth reservation **assign** command. The latter then prompts you for a queuing priority within said BRS class.

1.2.2 Access Lists to Classify Traffic

The Bandwidth Reservation System allows you to classify traffic based on IPv4 and IPv6 access lists (both standard and extended). For information on IPv4 Access Lists, please see manual *bintec Dm752-I Access Control*. For information on IPv6 Access Lists, see *bintec Dm808-I IPv6 Access Control*.

To assign an IPv4 Access List, enter **access-list**.

To assign an IPv6 Access List, enter **IPV6-access-list**.

1.2.3 Classifying Traffic through Criteria

The Bandwidth Reservation System allows you to classify traffic based on specific classification criteria, e.g. value of the packet tag.

To assign classification criteria, enter **match**.

1.2.4 Classifying Traffic through Protocol

When using bandwidth reservation, you can assign the following protocols (by entering **assign**) to specific types of traffic:

- IP
- X28
- ARP
- CFM
- BAN/ASRT

1.2.5 Order of Precedence

A packet can fall into several filterable classes. E.g. an IP Tunneled bridge packet for SNA, with a MAC Address filter. Filter priority is important: the highest priority is the one determining where a packet is classified.

The filtering priority order for bridge packets (ASRT) is as follows:

(1) MAC Address match, tag 1 to tag 5
(2) NETBIOS
(3) SNA

Filtering priority order in IP is as follows:

(1) Mac address match, tag 1 to tag 5
(2) NETBIOS
(3) SNA
(4) IP tunneling
(5) SDLC relay
(6) Multicast
(7) SNMP
(8) Rlogin
(9) Telnet
(10) DLSw
(11) XOT
(12) Access Lists

1.2.6 Pre-classification

Bandwidth reservation classifies traffic at the point at which said traffic is delivered to the interface where BRS is enabled. To correctly classify a packet, the classification criteria must match the packet protocol. For example, IPv4 (access lists) can classify an IPv4 protocol packet but not other protocol packets. In the examples below, the explanation given for IPv4 access lists also applies to IPv6 access lists.

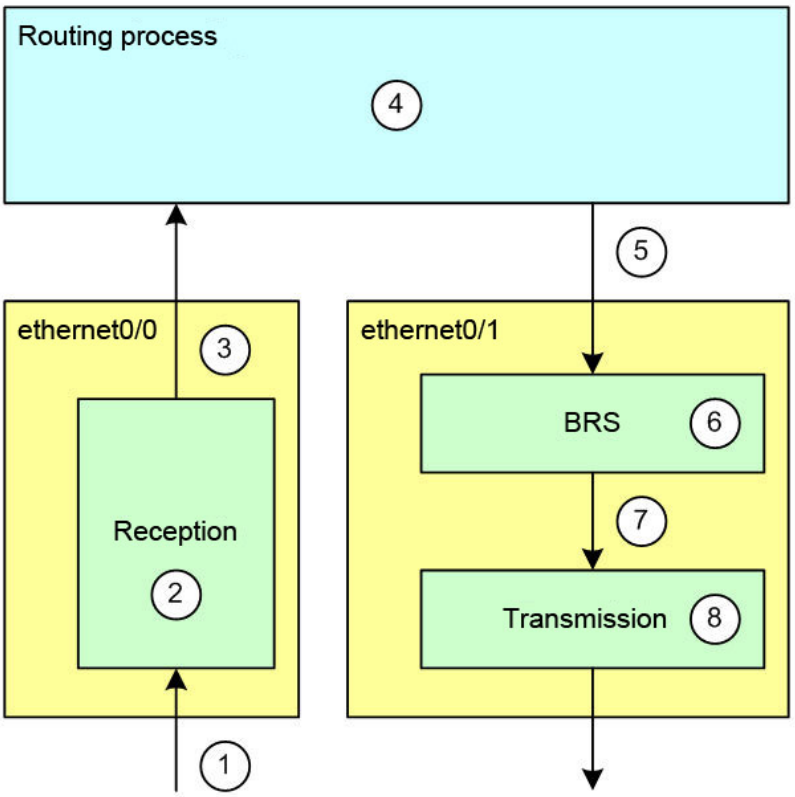
Note

The classification criteria (access-list, assign, ...) must be adequate for the packet protocol (IP, VLAN 802.1q, …).

Example 1:

We have a router with ethernet0/0 and ethernet0/1 interfaces. BRS is enabled in the ethernet0/1 interface.

We shall describe the processing of a packet using the following figure:



(1) A frame arrives through the ethernet0/0 interface.

(2) The ethernet0/0 interface performs layer 2 encapsulation, obtaining an IP packet.

(3) The ethernet0/0 interface delivers the IP packet to the routing process.

(4) The routing process determines whether the IP packet has to be transmitted through the ethernet0/1 interface.

(5) The IP packet is delivered to the ethernet0/1 interface.

(6) As BRS is enabled in the ethernet0/1 interface, the IP packet is classified according to the criteria configured in said interface. These criteria are applied over an IP protocol packet and can be based on the IP header fields (source IP address, TCP ports, etc.). The packet is tagged and queued in the corresponding class.

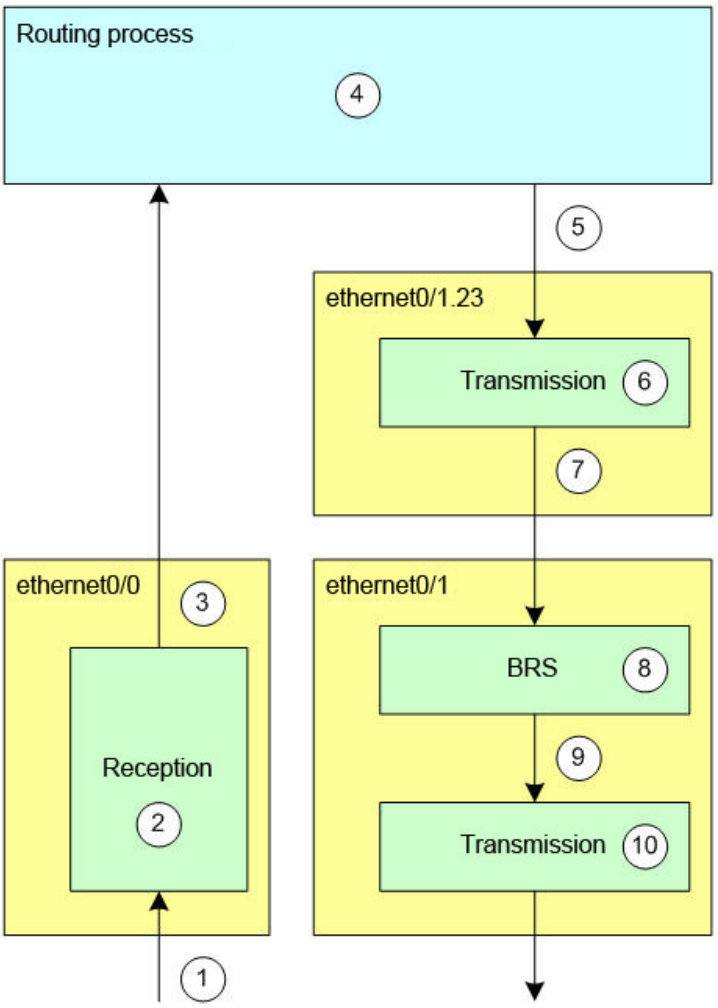
(7) When the necessary requirements are met, the IP packet is removed from the queue and delivered to the ethernet0/1 interface transmission process.

(8) The ethernet0/1 interface transmission process adds the layer 2 headers to the IP packet and transmits the resulting frame.

Example 2:

We have a router with Ethernet interfaces: ethernet0/0, ethernet0/1 and ethernet0/1.23. BRS is enabled in ethernet0/1.

We shall describe the processing of a packet using the following figure:



(1) A frame arrives through the ethernet0/0 interface.

(2) The ethernet0/0 interface performs layer 2 encapsulation, obtaining an IP packet.

(3) The ethernet0/0 interface delivers the IP packet to the routing process.

(4) The routing process determines the IP packet has to be transmitted through the ethernet0/1.23 interface.

(5) The IP packet is delivered to the ethernet0/1.23 interface.

(6) The ethernet0/1.23 interface transmission process adds VLAN 802.1q headers to the IP packet, thus obtaining a VLAN packet.

(7) The ethernet0/1.23 interface transmission process delivers the VLAN packet to its base interface, i.e. to the ethernet0/1 interface.

(8) As BRS is enabled in the ethernet0/1 interface, the VLAN packet is classified according to the criteria configured in said interface. IP criteria-like access lists are not applicable, since the packet is not IP but VLAN (and was encapsulated by the ethernet0/1.23 interface). The packet is tagged and queued in the corresponding class.

(9) When the right conditions are met, the VLAN packet is removed from the queue and delivered to the ethernet0/1 interface transmission process.

(10) The ethernet0/1 interface transmission process adds the necessary headers to the VLAN packet and transmits the resulting frame.

In example 2, BRS in ethernet0/1 does not have layer 3 (IP) information to classify the subinterface traffic through access lists, as it receives VLAN protocol packets (instead of IP). The only traffic that can be classified using access lists, is traffic directly routed to ethernet0/1 (i.e., not from ethernet0/1.23, which is VLAN).

If we want to classify the traffic of ethernet0/1 subinterfaces before encapsulation, we must enable preclassification in those subinterfaces.

The preclassification feature (qos-pre-classify) makes a copy of the packet headers before they are modified by the interface transmission process (where said feature is enabled).

Note

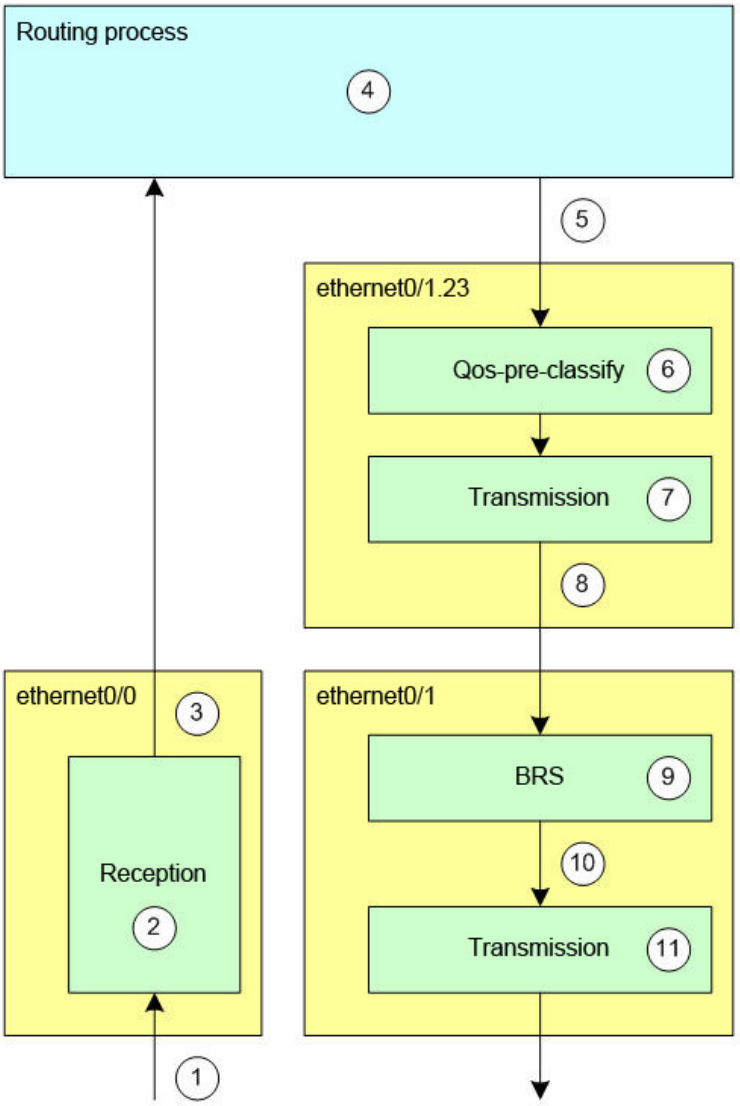
Preclassification memorizes packet headers before they are modified (802.1q encapsulation in the Ethernet subinterface, IP/GRE encapsulation in the tnip interface, IPSec encryption/encapsulation, etc.).

This allows an output interface with BRS to classify a packet according to the original information (IP fields not yet encapsulated in VLAN 802.1q, packets traveling in the GRE tunnel, pre-encrypted information, etc.).

Example 3:

We have a router with Ethernet interfaces: ethernet0/0, ethernet0/1 and ethernet0/1.23. BRS is enabled in ethernet0/1. Qos-pre-classify is configured in ethernet0/1.23.

We shall describe the processing of a packet using the following figure:



(1) A frame arrives through the ethernet0/0 interface.

(2) The ethernet0/0 interface performs layer 2 encapsulation, obtaining an IP packet.

(3) The ethernet0/0 interface delivers the IP packet to the routing process.

(4) The routing process determines the IP packet has to be transmitted through the ethernet0/1.23 interface.

(5) The IP packet is delivered to the ethernet0/1.23 interface.

(6) As preclassification has been enabled (**qos-pre-classify**) in the ethernet0/1.23 interface, a copy of the IP packet headers is made and associated to the packet.

(7) The ethernet0/1.23 interface transmission process adds VLAN 802.1q headers to the IP packet, thus obtaining a VLAN packet, which is still associated with a copy of the IP packet headers.

(8) The ethernet0/1.23 interface transmission process delivers the VLAN packet to its base interface, i.e. to the ethernet0/1 interface.

(9) Since BRS is enabled on the ethernet0/1 interface, the VLAN packet is classified according to the criteria configured in said interface and applicable to the copy of the IP headers (not to the VLAN packet). Both the IP criteria and the access lists are applicable, since the copied headers come from the IP protocol (prior to VLAN encapsulation introduced by the ethernet0/1.23 interface). The packet is tagged and queued in the corresponding class.

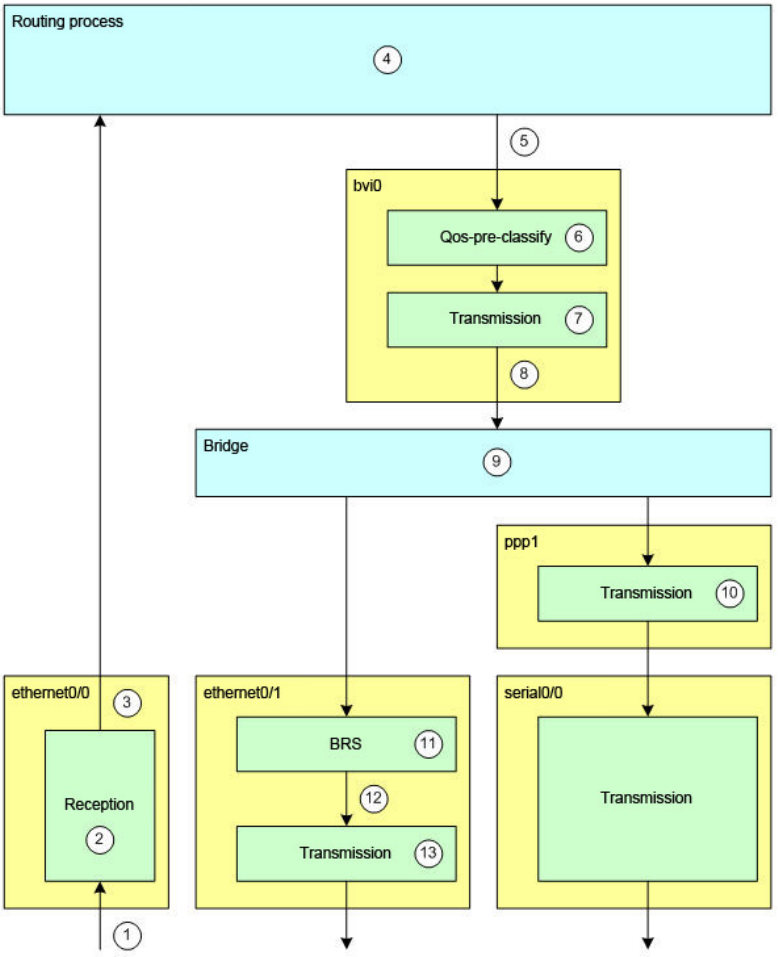
(10) When the correct conditions are met, the VLAN packet is removed from the queue and delivered to the ethernet0/1 interface transmission process.

(11) The ethernet0/1 interface transmission process adds the necessary headers to the VLAN packet and transmits the resulting frame.

Example 4:

We have a router with the following interfaces: ethernet0/0, ethernet0/1, ppp1, serial0/0 and bvi0. Bridge with bvi0 interface and ppp1 ports and ethernet0/1. Ppp1 interface over serial0/0. BRS is enabled in ethernet0/1. Qos-pre-classify is configured in bvi0.

We shall describe the processing of a packet using the following figure:



(1) A frame arrives through the ethernet0/0 interface.

(2) The ethernet0/0 interface performs layer 2 encapsulation, obtaining an IP packet.

(3) The ethernet0/0 interface delivers the IP packet to the routing process.

(4) The routing process determines the IP packet must be transmitted by the bvi0 interface.

(5) The IP packet is delivered to the bvi0 interface.

(6) As preclassification is enabled (**qos-pre-classify**) in the bvi0 interface, a copy of the IP packet headers is made and associated with the packet.

(7) The bvi0 interface transmission process adds the layer 2 headers to the IP packet. The result is a bridge frame that is still associated with a copy of the IP packet headers.

(8) The bvi0 interface transmission process delivers the bridge frame to the bridge entity.

(9) The bridge entity doesn't have the destination mac address in its tables, so it sends two copies of the packet (with preclassification). One is sent to the ppp1 interface and the other to the ethernet0/1 interface.

(10) The ppp1 interface transmission process adds the necessary headers to the packet (bridge frame) and delivers it to the serial0/0 interface, which, in turn, forwards it through the serial line.

(11) Since BRS is enabled in the ethernet0/1 interface, the packet (bridge frame) is classified according to the criteria configured in said interface and applicable to the copy of the IP headers (not to the bridge frame). Both IP cri-

teria and access lists are applicable, since the copied headers come from the IP Protocol (prior to bridge encapsulation via the bvi0 interface). The packet is tagged and queued in the corresponding class.

(12) When the right conditions are met, the packet (bridge frame) is removed from the queue and delivered to the ethernet0/1 interface transmission process.

(13) The ethernet0/1 interface transmission process adds the necessary headers to the packet (bridge frame) and transmits the resulting frame.

Preclassification (qos-pre-classify) is applicable to various processes, which alter the packets:

- TNIP Interface. Copies the packet headers before encapsulation. Please see manual *bintec Dm719-I IP Tunnel* .
- IPSec Protocol. Copies the packet headers before encryption and encapsulation. Please see manual *bintec Dm739-I IPSec.*
- Ethernet subinterface. Copies the packet headers before encapsulation in VLAN 802.1q. Please see manual *bintec Dm750-I Ethernet Subinterface.*
- BVI Subinterface. Copies the packet headers before encapsulation in a bridge frame. Please see manual *bintec Dm717-I Bridge.*
- BVI Subinterface. Copies the packet headers before encapsulation in VLAN 802.1q and a bridge frame. Please see manual *bintec Dm717-I Bridge.*

Note

Several functions alter the content of the packets and allow preclassification (**qos-pre-classify**) to be enabled so packets are classified according to their original state.

We recommend consulting the manual for additional information on the **qos-pre-classify** command for each of these.

In certain configurations, you may try to reclassify the same packet twice. Packets, however, can only be preclassified the first time.

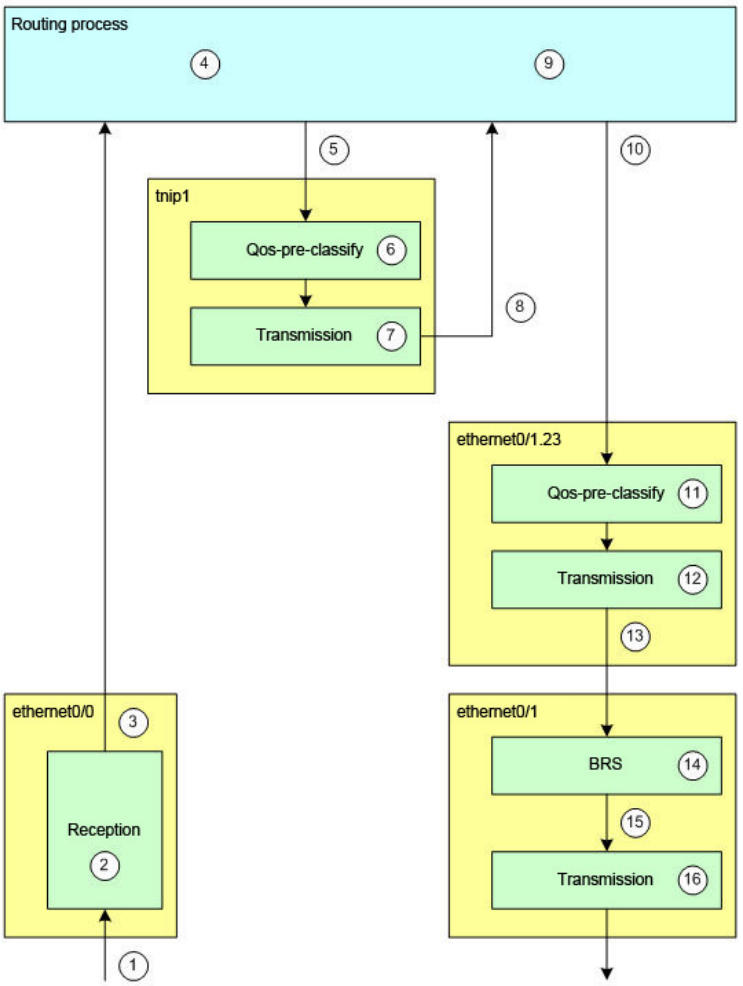
Note

You can only preclassify once when processing a packet. A packet already preclassified cannot be preclassified again.

Example 5:

We have a router with the following interfaces: ethernet0/0, ethernet0/1, ethernet0/1.23 and tnip1. BRS is enabled in ethernet0/1. Qos-pre-classify is configured in ethernet0/1.23 and tnip1.

We shall describe the processing of a packet using the following figure:



(1) A frame arrives through the ethernet0/0 interface.

(2) The ethernet0/0 interface performs layer 2 encapsulation, obtaining an IP packet with source and destination addresses (1.1.1.1 # 2.2.2.2).

(3) The ethernet0/0 interface delivers the IP packet to the routing process.

(4) The routing process determines the IP packet has to be transmitted by the tnip1 interface.

(5) The IP packet is delivered to the tnip1 interface.

(6) Since the pre-classification function is enabled (**qos-pre-classify**) in the tnip1 interface, a copy of the IP packet headers is made and associated with the packet.

(7) The tnip1 interface transmission process adds the GRE and IP headers to the IP packet, obtaining a new IP packet with source and destination addresses (3.3.3.3 # 4.4.4.4). This is still associated with the copy of IP packet headers (1.1.1.1 # 2.2.2.2).

(8) The tnip1 interface transmission process delivers the IP packet (3.3.3.3 # 4.4.4.4) to the routing process.

(9) The routing process determines the IP packet (3.3.3.3 # 4.4.4.4) has to be transmitted through the ethernet0/1.23 interface.

(10) The IP packet (3.3.3.3 # 4.4.4.4) is delivered to the ethernet0/1.23 interface.

(11) The pre-classification function is enabled (**qos-pre-classify**) in the ethernet0/1.23 interface but the IP packet (3.3.3.3 # 4.4.4.4) has already been pre-classified. Said pre-classification remains unaltered, retaining the copy of the IP headers (1.1.1.1 # 2.2.2.2).

(12) The ethernet0.1.23 interface transmission process adds the VLAN 802.1q headers to the IP packet (3.3.3.3 # 4.4.4.4), obtaining a VLAN packet. This is still associated with the copy of IP packet headers (1.1.1.1 # 2.2.2.2).

(13) The ethernet0/1.23 interface transmission process delivers the VLAN packet to its base interface, i.e. to the ethernet0/1 interface.

(14) Since BRS is enabled in the ethernet0/1 interface, the VLAN packet is classified according to the criteria configured in the interface and applicable to the copy of the IP headers (1.1.1.1 # 2.2.2.2), instead of to the VLAN packet or the IP packet (3.3.3.3 # 4.4.4.4). Both IP criteria and access lists are applicable, since the copied headers belong to the IP protocol (before IP/GRE encapsulation is done via the tnip1.23 interface). The packet is tagged and queued in the corresponding class.

(15) When the right conditions are met, the VLAN packet is removed from the queue and delivered to the ethernet0/1 interface transmission process.

(16) The ethernet0/1 interface transmission process adds the necessary headers to the VLAN packet and transmits

the resulting frame

1.3 Traffic Marking

It's often useful to mark packets for different types of traffic so that the rest of the network can distinguish them and treat them accordingly. The Bandwidth Reservation System allows you to mark traffic as follows:

- Classifying the traffic with the **access-list <n> <class> <priority> set <type of marking and value to mark>** command.
- Classifying the traffic with the **IPv6-access-list <n> <class> <priority> set <type of marking and value to mark>** command.
- Classifying the traffic with the **match <criteria> <data> class <class> <priority> set < type of marking and value to mark >** command.
- All traffic classified under one class without having been previously marked through the **access-list, IPv6-access-list** or **match** commands. This is configured with the **class <class> set <type of marking and value to mark>** command.
- Traffic overflowing from a class, with the **class <class> exceed mark-dscp-continue <value to mark>** command.

IPv4 packets can be marked using the *Type of Service* field that belongs to the IPv4 header. In the case of IPv6 packets, this is done via the *Traffic Class* field of the IPv6 header. Packets transmitted through Ethernet subinterfaces are marked through the COS field in the VLAN 802.1Q header and over the Cell Loss Priority bit in the ATM cell header (this marking is only effective if the packet is transmitted over an ATM interface).

COS bits are found in the Tag Control Information (TCI) field of the VLAN 802.1Q header, as shown in the following figure:

Octets:	0								1							
	COS			CFI					VID							
Bits:	8	7	6	5	4	3	2	1	8	7	6	5	4	3	2	1

IPv4 Header

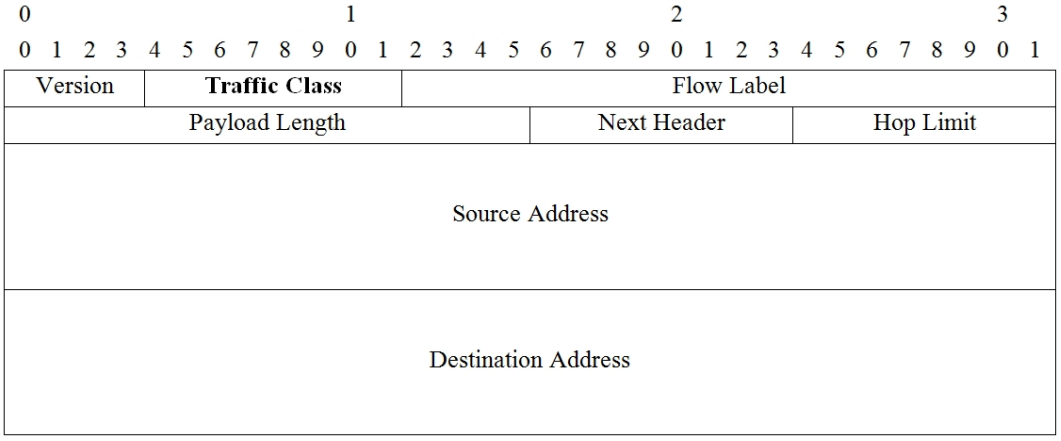
The IPv4 header format is defined in RFC 791, as shown in the following figure:

0				1				2				3			
0 1 2 3	4 5 6 7	8 9 0 1	2 3 4 5	6 7 8 9	0 1 2 3	4 5 6 7	8 9 0 1								
Version	IHL	**Type of Service**		Total Length											
Identification				Flags	Fragment Offset										
Time to Live		Protocol		Header Checksum											
Source Address															
Destination Address															
Options													Padding		

The *Type of Service* field can be interpreted in several ways, and each has its corresponding format. When configuring marking in BRS, you can mark the Precedence field, the DSCP field, or any combination of bits in the *Type of Service* octet. By way of reference, the most common uses of the *Type of Service* field are explained in the following paragraphs.

IPv6 Header

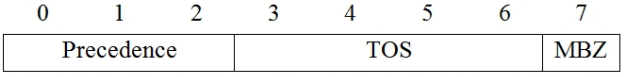
The IPv6 header format is defined in the RFC 2460 standard, as shown in the following figure:



There are several ways to fill out the *Traffic Class* field (this is very similar to the *Type of Service* field in the IPv4 protocol header) while complying with the relevant formats. When configuring marking in BRS, you can mark the Precedence field, the DSCP field or any combination of the *Traffic Class* octet bits. As a reference in the following sections, the most common uses of the *Traffic Class* field have been indicated.

Precedence and TOS Fields

The *Type of Service* (IPv4) or Traffic Class (IPv6) field can be broken down in the Precedence, TOS and MBZ fields, as shown in the following figure:



Type of Service (IPv4) *or Traffic Class* (IPv6) *field according to RFC 1349*

The first 3 bits in the *Type of Service* field (IPv4) or *Traffic Class* (IPv6) in the IPv4/IPv6 header indicate packet precedence. The following table displays the values defined for this field:

Value (binary)	Nomenclature
000	Routine
001	Priority
010	Immediate
011	Flash
100	Flash Override
101	CRITIC/ECP
110	Internetwork Control
111	Network Control

Bits 3 to 6 are the TOS field, with 16 possible values. Those that have been defined appear in the following table:

Value (binary)	Nomenclature
0000	Normal service
0001	Minimize monetary cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

The last bit (MBZ field) should be set to 0.

When using the traffic-marking commands in BRS, you can specify any combination of bits in the Type of Service octet. Therefore, to mark a packet with Internetwork Control precedence (110) and TOS Maximize reliability (0010), the byte value is 11000100 in binary with mask 11111110 also in binary. This means we must configure the **set tos-octet** option with the corresponding values in decimal, as shown in the following example:

```
BRS [i #] Config>class alpha set tos-octet 196 mask 254
BRS [i #] Config>
```

DSCP Field

In RFC 2474 (*Definition of the Differentiated Services Field in the IPv4 and IPv6 Headers*), the *Type of Service* field

in the IPv4 header and the *Traffic Class* field in the IPv6 header have been redefined as the DS field. The latter is divided into one 6-bit field, known as DSCP (*Differentiated Services CodePoint*), and one 2-bit field called CU (Currently Unused). They are shown in the following figure:

0	1	2	3	4	5	6	7
DSCP	CU						

Redefinition of the Type of Service field (IPv4) and the Traffic Class (IPv6) field according to RFC 2474

The CU field bits have been subsequently redefined for other uses (see RFC 3168 for example).

1.4 Bandwidth Sharing

The Bandwidth Reservation System (BRS) allows you to decide which packets to drop when demand (traffic) exceeds supply (throughput) on a network connection.

Bandwidth reservation is really a safeguard. Generally speaking, networks should not attempt to use more than 100% of their line speed. If they do, a faster line is probably needed. The bursty nature of traffic, however, can drive the requested transmission rate to exceed 100% for a short period of time. In such cases, bandwidth reservation is enabled to make sure the traffic with the highest priority is delivered (i.e., not discarded). If, over time, the traffic exceeds the line capacity, packets begin to be discarded. However, the bandwidth percentages assigned to the different classes are still observed.

When multiple classes of traffic with the same priority compete for line bandwidth, the latter is proportionally divided amongst them based on the configured values. A bandwidth with no set class is equally divided among the other classes, also based on the values configured.

Table 1 shows how the bandwidth is divided between classes with the same priority.

Table 1

Interface[1]	Class[2] A	Bandwidth[3] %	access list, protocol, tag or filter[4]	priority level[5]
			access list, protocol, tag or filter	priority level
			access list, protocol, tag or filter	priority level
	Class B	Bandwidth %	access list, protocol, tag or filter	priority level
			access list, protocol, tag or filter	priority level
			access list, protocol, tag or filter	priority level
	Class C	Bandwidth %	access list, protocol, tag or filter	priority level
			access list, protocol, tag or filter	priority level
			access list, protocol, tag or filter	priority level

(1) An interface where bandwidth reservation is supported (X.25 Line, PPP Line, ATM subinterface, Ethernet interface, etc.).

(2) BRS class.

(3) Interface bandwidth percentage for this BRS class. Use the **class** command.

(4) Type of packet in the BRS class. Use the **access-list**, **IPV6-access-list** or **assign** commands.

(5) Priority level for packets with a given protocol, tag or filter. Use the **access-list**, **IPV6-access-list** or **assign** commands.

Note

In Frame Relay, the interface bandwidth is shared among the circuits based on other classes. In these classes, only the percentage of the bandwidth and the circuits sharing it are defined.

1.5 Priority

Bandwidth reservation allocates percentages of total connection bandwidth to specific traffic classes (defined by the user). A BRS class is a group of packets identified by the same name (for example, a class called **ipx** to designate all IPX packets).

The BRS subsystem allows you to prioritize some traffic classes over others, meaning different types of traffic are strictly processed before others and without reserving a particular bandwidth percentage.

There are two types of prioritization: inter-class prioritization and intra-class prioritization.

Inter-class Prioritization:

Each traffic class is allocated a percentage of bandwidth and a priority. This priority can take the following values. Default is *normal*:

- REAL-TIME
- HIGH
- NORMAL (default value)
- LOW

When choosing the class whose packets are to be sent, classes with *real-time* priority take precedence. If there are no classes with *real-time* priority, or there are no queued packets in this class to be transmitted, classes with *high* priority are next and so on (until *low* priority classes are processed).

The bandwidths allocated to each class are observed among classes with the same priority.

Example:

Apart from the local and the default classes, there are four classes in a configuration. One class, known as voip, with real-time priority, two classes, important1 and important2, with high priority, and a fourth, known as data, with normal priority. The local and default classes are always present to represent local traffic and traffic that does not belong to any other class. In this case, they take normal priority.

```
class default 20
class voip 100 real-time
class important1 30 high
class important2 70 high
class data 60
class local 20
```

In this scenario, the first class to be transmitted is always voip as this has the highest priority. If voip does not have any data to transmit at this point, then classes important1 or important2 will be transmitted respecting the bandwidth assigned to each (i.e. for each 70 bytes transmitted by important2, 30 will be transmitted by important1).

In cases where voip, important1 and important2 do not have any data to transmit, data, default and local classes will transmit their data depending on their bandwidth.

Suppose we have a bandwidth of 100 Kbps and the following throughputs:

CLASS	THROUGHPUT	SENT	DROPPED
Voip	20000 bps	20000 bps	0 bps
Important1	60000 bps	(100000 - 20000)*0.3 = 24000 bps	36000 bps
Important2	70000 bps	(100000 - 20000)*0.7 = 56000 bps	14000 bps
Data	100000 bps	0 bps	100000 bps
Local	10000 bps	0 bps	10000 bps
Default	20000 bps	0 bps	20000 bps

Suppose we have a bandwidth of 100 Kbps and the following throughputs:

CLASS	THROUGHPUT	SENT	DROPPED
Voip	10000 bps	10000 bps	0 bps
Important1	10000 bps	10000 bps	0 bps
Important2	15000 bps	15000 bps	0 bps

Data	100000 bps	(100000 – 35000) * 0.60 = 39000 bps	61000 bps
Local	15000 bps	(100000 – 35000) * 0.20 = 13000 bps	2000 bps
Default	20000 bps	(100000 – 35000) * 0.20 = 13000 bps	7000 bps

Important

As you can see from the tables, classes with the highest priorities can completely block data transmission of classes with lower priorities. This does not happen when you use bandwidth reservation exclusively (all classes taking normal priority).

Intra-class Prioritization:

Each BRS class has four queues; one for each priority through which access lists or tags can be assigned to a particular class. When deciding which class is going to transmit next (see previous section), the queues of each class are examined in the following order:

- URGENT
- HIGH
- NORMAL (default)
- LOW

When deciding which packets within a class will be sent, packets marked as *urgent* are sent first. These packets are followed by *high*, *normal*, and then *low* packets (respectively). When all *urgent* packets have been transmitted, *high* priority packets are transmitted and so on. *Low* priority packets are only transmitted when there are no *urgent*, *high*, or *normal* packets left. If no priority setting is assigned, the setting defaults to *normal*.

You can also set the number of packets that can be queued for each priority level in each bandwidth class. The BRS **queue-length** command sets the maximum number of output packets that can be queued in each BRS priority queue.

Caution

If you set the queue-length values too high, you may seriously degrade the performance of your router.

You can set priority queue lengths for each type of interface that supports BRS: X.25 Line, PPP, ATM subinterface, Ethernet, TNIP, Frame Relay, etc.

Priority settings in one bandwidth class do not affect other bandwidth classes. No bandwidth class has priority over the others. You can only map a network protocol (or several grouped protocols) or filters and a class.

Effects of Prioritizing

When you configure priority queuing without bandwidth limitations, the router delivers the highest priority traffic first. When there is heavy high priority traffic, the router can never attend or give service to lower priority levels. However, by combining priority queuing with bandwidth limitations, you can allocate packet transmission to all bandwidths.

Warning

Prioritizing should only be configured for very important traffic that is both sporadic and light (such as alarms, etc.). If not, you run the risk of paralyzing traffic with lower priorities.

1.6 Bandwidth Limitation – Traffic-shaping

Our routers allow traffic-shaping to be performed over all interfaces that support BRS. Traffic-shaping allows you to limit the maximum throughput of an interface, PVC or specific traffic class.

Therefore, you can set the FTP traffic class to have a 10% guaranteed bandwidth and a maximum throughput of 40 kbps.

To limit maximum throughput, use the **rate-limit** command applied to interfaces, circuits and classes:

```
BRS [i <interface>] Config>rate-limit [percent] <cir> [<bc> > [<be>]]
BRS [i <interface>] Config>class <name> rate-limit [percent] <cir> [<bc> [<be>]]
```

```
BRS [i <interface>] [dlci 16 Config>rate-limit [percent] <cir> [<bc> [<be>]]
BRS [i <interface>] [dlci 16] Config>class <name> rate-limit [percent] <cir> [<bc> [<be>]]
```

<cir>	*Committed information rate*. Maximum average *Throughput* configured in Kbps or as a percentage of the total line bandwidth.
<bc>	*Committed burst*. Maximum burst size allowed in kilobits.
<be>	*Excess burst*. Maximum excess burst size allowed in kilobits.
<name>	Name of class to configure.
<interface>	Name of interface to configure.

Note

For further information, please see the section on Rate Limit.

Given that the line rate is set and all packets must be integrally and continuously transmitted, you can only act on the average throughput in a given time interval. To do this, the rate-limit command admits up to three parameters: CIR, Bc and Be. These parameters define the way the throughput is limited. Their meaning is as follows:

- CIR: Committed Information Rate. This is the throughput allowed (i.e. the sustained transfer rate that can be reached).
- Bc: Burst Committed. This is the maximum burst size allowed. If nothing is specified, then the value is calculated as Bc = CIR x 0,125 s (i.e. throughput is guaranteed in 125 ms intervals).
- Be: Burst Excess. This is the burst excess allowed. This parameter is purely administrative since it doesn't perform any direct action on the congestion control bits (unlike the Frame Relay DE bit or the ATM CLP bit). If you don't specify anything, a 0 value is taken.

Throughput limitation works differently, depending on the layer it is configured at. In a congestion situation, a **rate-limit** configured at a class layer (or Frame-Relay circuit) acts by discarding packets before they are inserted in the BRS output queues, while still leaving available space in said queues. These discarded packets increase the discarded packet counters by rate-limit in the statistics of the BRS feature. A rate-limit globally configured at the interface layer is the same as having a bandwidth equal to the configured rate-limit. This means that, under congestion conditions, the queues fill up until they overflow and packets are discarded due to saturation. That is why a rate-limit at the interface layer causes an increase in the discarded packet counters and an *overflow* in the statistics.

The three parameters are used to obtain a maximum final burst size. The following shows the calculation in detail:

(1) Get Tc. If Bc is specified, then Tc = Bc / CIR. If Bc isn't specified, then Tc = 125 ms.

(2) Limit Tc to a minimum of 7.8 ms and a maximum of 1s.

(3) Get the Bf final burst size. If Be is specified, then Bf = CIR x Tc + Be. If Be isn't specified, then Bf = CIR x Tc.

(4) Use CIR and Bf to limit throughput. This results in an average interval Tf = Bf / CIR.

The results of these calculations are as follows:

- Data quantities above Bf cannot be transmitted in any Tf time interval. That is, the average transmission speed in any Tf time interval is equal to, or less than, the configured CIR.
- In normal configurations (only the CIR is indicated) bursts are permitted so that CIR is not exceeded in any 125 ms interval. Therefore, the maximum burst size allowed depends on the CIR.
- In advanced configurations (the CIR and Bc are indicated), the user has control over interval size for limited throughput (always within the above mentioned limits of 7.8 ms and 1 s).
- By configuring the Be parameters, you can exceed the 1s limit for the interval size. The final burst size will generally be the sum of Bc and Be.

Example 1: configuring a 160 rate-limit.

- CIR = 160 kbps.
- Bc = CIR x 0.125 s = 160 kbps x 0.125 s = 20 kbit.
- Tc = Bc / CIR = 20 kbit / 160 kbps = 0.125 s = 125 ms.
- The device limits the throughput so that it does not exceed an average of 160 Kbps in a 125 ms interval.

Example 2: configuring a 160 32 rate-limit.

- CIR = 160 kbps.
- Bc = 32 kbit.
- Tc = Bc / CIR = 32 kbit / 160 kbps = 0.2 s = 200 ms.
- The device limits the throughput so it does not exceed an average of 160 Kbps in a 200 ms interval.

Example 3: configuring a 160 180 rate-limit.

- CIR = 160 kbps.
- Bc = 180 kbit.
- Tc = Bc / CIR = 180 kbit / 160 kbps = 1125 s. As Tc > 1 s this takes Tc = 1 s.
- The device limits the throughput so it does not exceed an average of 160 Kbps in a 1s interval.

Example 4: configuring a 160 160 20 rate-limit.

- CIR = 160 kbps.
- Bc = 160 kbit.
- Be = 20 kbit.
- Tc = Bc / CIR = 160 kbit / 160 kbps = 1 s.
- Bf = CIR x Tc + Be = 160 kbps x 1 s + 20 kbit = 180 kbit.
- Tf = Bf / CIR = 180 kbit / 160 kbps = 1.125 s = 1125 ms.
- The device limits the throughput so it does not exceed an average of 180 Kbps in a 1125 ms interval.

Example 5: configuring a 160 180 20 rate-limit.

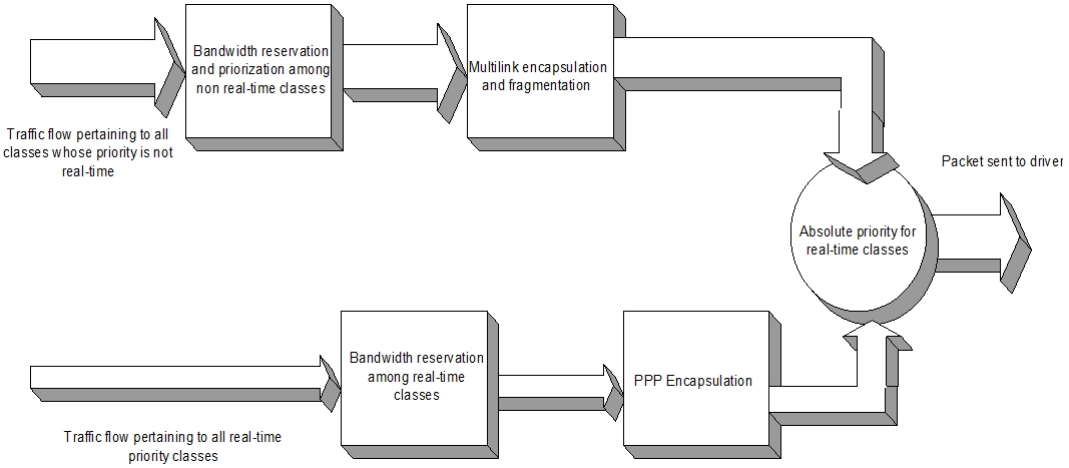
- CIR = 160 kbps.
- Bc = 180 kbit.
- Be = 20 kbit.
- Tc = Bc / CIR = 180 kbit / 160 kbps = 1.125 s. As Tc > 1 s this takes Tc = 1 s.
- Bf = CIR x Tc + Be = 160 kbps x 1 s + 20 kbit = 180 kbit.
- Tf = Bf / CIR = 180 kbit / 160 kbps = 1.125 s = 1125 ms.
- The device limits the throughput so it never exceeds an average of 180 Kbps in a 1125 ms interval.

In ATM networks, it is quite normal for the guaranteed bandwidth to be lower than the bandwidth offered by the line. This causes the network to discard cells when there is congestion. The Cell Loss Priority bit in the ATM cell header marks cells with drop priority. This means other unmarked cells are only dropped if discarding marked cells is insufficient. You can also limit the maximum throughput to a value guaranteed by the operator, so when this maximum throughput is exceeded, the router (instead of the network) discards packets (according to their type).

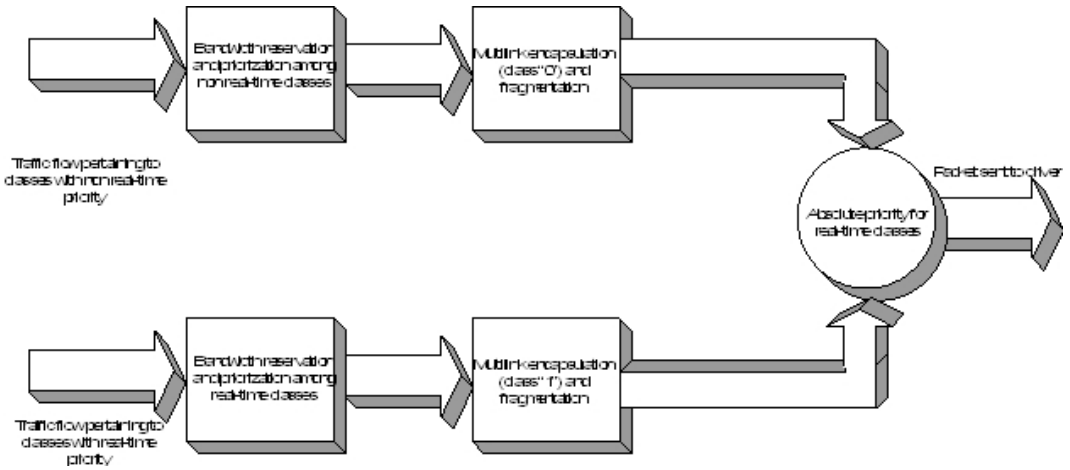
1.7 Quality of Service in Multilink Links

The BRS subsystem operates in a slightly different way for links configured with MP than it does for other encapsulations. It applies the BRS subsystem before encapsulating in MP.

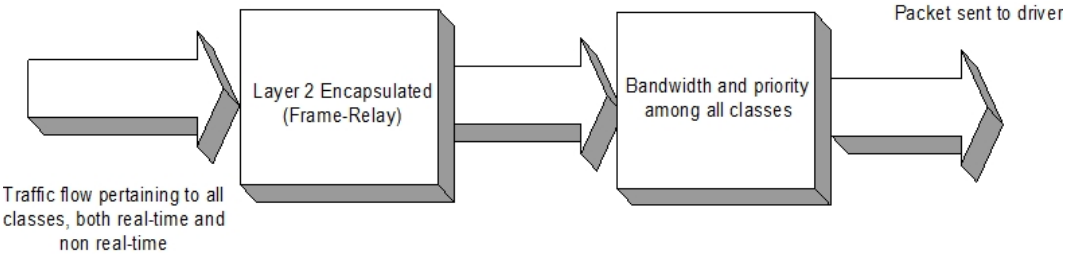
The operating schema for MP links when the *Multiclass Multilink* feature is **NOT** enabled is as follows:



In cases where *Multiclass MP* is enabled, and just like what happens to non-real-time traffic, traffic belonging to real-time priority classes is encapsulated with an MP header. Real-time traffic is marked as *Class 1* and non-real-time traffic as *Class 0*. In this scenario, the operating schema for MP links is as follows:



Both schemes contrast with the generic scheme for other encapsulations, such as Frame Relay, where the process is as follows:



The generic scheme is not valid for MP, since it assigns a sequence number to each packet. This number is used at the receiving end to reorder received packets and avoid possible misunderstandings due to the use of multiple links. The problem is that, if BRS is applied after MP encapsulation (i.e. after assigning a sequence number to the packets), said packets would become so disorganized that the MP receiving end would be, in most cases, unable to reorder them and discard them.

The solution to this problem is to apply the BRS before encapsulating in MP. That is, first disarray the packets as much as necessary to preserve the bandwidth assigned to each class and then encapsulate them in MP and tag them with sequence numbers. This way, you do not jumble the sequence numbers generated by the BRS and packets arrive in good order for the receiving end to sort them out.

The problem with this solution comes with very delay-sensitive traffic, such as VoIP. If this traffic were to enter the BRS before being encapsulated in MP (that is, before being fragmented), the delays caused by large-sized packets would be equivalent to delays without fragmentation (since traffic prioritizing is performed before fragmentation).

As a result, after being encapsulated in MP, this kind of packet is sent to different BRS queues (real-time traffic queues) that take priority over other traffic.

1.8 Bandwidth reservation over Frame Relay

When you run bandwidth reservation over Frame Relay, there are two areas where you can allocate bandwidth: the circuit layer and the interface layer.

The per-circuit bandwidth allocation works similarly to the X.25 Line. Packets are filtered and placed in queues according to the BRS classes based on the protocols and filters assigned to the per-circuit configured classes.

The actual amount of bandwidth available for bandwidth reservation depends on how you configure the interface and circuit:

- If Frame Relay CIR monitoring is enabled, the available bandwidth is assigned to a circuit in strict accordance with its CIR (Committed Information Rate), CBS (Committed Burst Size), and EBS (Excess Burst Size).
- If you disable CIR monitoring, up to 100 % of interface bandwidth may be available in a circuit.

Orphaned circuits and circuits that do not have BRS explicitly enabled can use the BRS queue placement default environment.

Each circuit also competes for bandwidth on the physical serial line. Bandwidth allocation on the physical interface divides the circuit into classes. The percentage of bandwidth allocated to each circuit class is configurable. Orphaned circuits and circuits that are not assigned to any class are put in the default circuit class.

To display reservation counters for the classes on the Frame Relay interface layer, use the following bandwidth reservation monitoring commands:

- CLEAR-CIRCUIT-CLASS
- COUNTER-CIRCUIT-CLASS
- LAST-CIRCUIT-CLASS

The interface is the one shown in the bandwidth monitoring commands prompt. For example, *BRS [i serial0/0] Config>* is the prompt for the interface corresponding to WAN1.

BRS classes are more useful when CIR monitoring is not enabled. If you do not want to use BRS classes, leave all circuits in the default class and do not create any other circuit classes.

1.8.1 Queuing Support in Frame Relay interfaces

In those Frame Relay interfaces that do not have the bandwidth reservation feature enabled, the traffic from all DLCIs is put into a single queue, whose length is determined by the availability of the buffers in the router at the time. This allows the device to cope with heavy traffic bursts during a certain period of time without having to discard frames.

When bandwidth reservation is enabled, but neither class nor protocol have been configured, there will be a queue for each DLCI and their lengths will be determined in the bandwidth reservation default configuration.

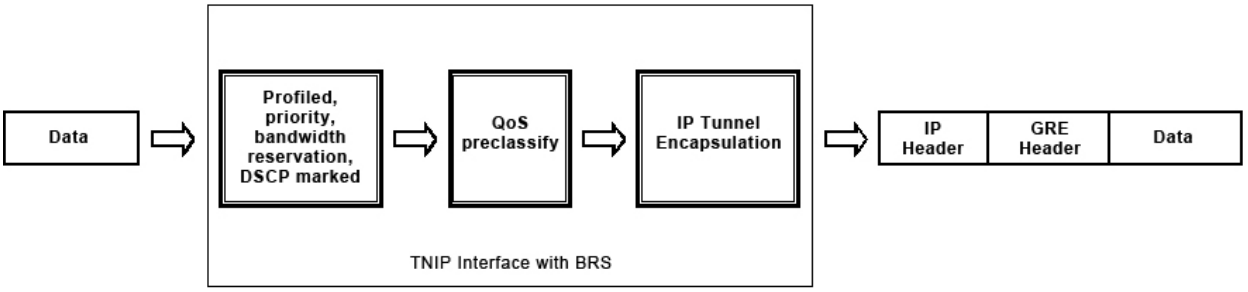
1.9 Bandwidth reservation over TNIP interface

While they are not physical interfaces, TNIP interfaces (IP tunnel interface) also support bandwidth reservation. Since they are not physical interfaces, the bandwidth is considered infinite. Therefore, if you do not limit the bandwidth (through traffic-shaping), all traffic will be sent to the output interface, regardless of load and configured percentages.

Note

Since TNIP interfaces have an infinite bandwidth, there is no point in reserving any if you do not limit it through traffic-shaping.

When configuring traffic-shaping in TNIP interfaces, you must bear in mind that the throughput considered here is traffic **before** tunnel encapsulation (i.e. before adding the IP headers, GRE etc.) In the chart below, you can see how quality of service is first applied to the data (and only after that are they encapsulated in the IP tunnel).



In this chart you can also see that **qos-pre-classify** is applied **after** the packet has been marked with a DSCP value. You must take this into account if BRS is also applied in the tunnel output interface.

1.10 BRS and Virtual Private Networks

In some scenarios with virtual private networks, it is interesting to prioritize certain types of traffic on both common IP tunnels and IPSEC. Our routers allow you to make a distinction between the different traffic encapsulated within the same tunnel.

To do this, enable the **qos-pre-classify** option. This is done in the IPSEC configuration menu or in the IP tunnel menu, depending on the one you are using for the virtual private network.

After configuring this option, packets are classified into the different BRS classes before encapsulating. This allows you to make a distinction between the different types of traffic encapsulated in the same tunnel.

For more information, see manuals *bintec Dm739-I IPSEC* and *bintec Dm719-I IP tunnel interfaces*.

1.11 BRS and VLAN

The Bandwidth Reservation System can also be applied to Ethernet subinterfaces. However, sometimes it is useful for the traffic aggregated in the Ethernet base interface to share the bandwidth. To do this, configure **qos-pre-classify** in the Ethernet subinterfaces. This option allows you to classify subinterface traffic before encapsulating it in the corresponding VLAN. If this is not done, all Ethernet subinterface traffic is simply regarded as VLAN protocol traffic during Ethernet base interface classification.

For further information, see the *bintec Dm750T Ethernet Subinterface* manual.

1.12 BRS and Bridge

In scenarios with bridge, you can use the Bandwidth Reservation System both in ports and on the BVI interface.

In scenarios where you want to manage bandwidth only for the traffic sent by the router (not bridged traffic) regardless of the output port, you can enable BRS in the BVI interface. In this case, and since it is a virtual interface, you must configure a rate-limit based on the maximum bandwidth desired.

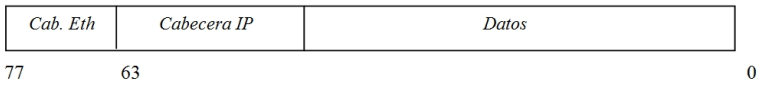
In scenarios where you want to manage bandwidth in a single bridge port, you must enable BRS on that port. If you also want to analyze traffic transmitted by the router at the network level, you must enable pre-classification in the BVI interface.

1.13 Calculating bandwidth in the BRS

In our devices, BRS is performed at the interface layer. Bandwidth limitation and distribution algorithms are solely based on the size of the packets to be transmitted by said interface.

A user can configure the BRS work mode at the network layer or at the link layer. The difference between the two modes is that the packet size, the basic parameter in the bandwidth limitation and distribution algorithms, is calculated at the network or link layer, respectively.

Take for example an IP packet of 64 bytes to be transmitted over an Ethernet interface. The size of the packet at the network layer is 64 bytes, including data and IP header (20 bytes). After being encapsulated in Ethernet, an Ethernet header is added to the packet, increasing its size by 14 bytes when moving to the link layer. Its size at the link layer is, therefore, 78 bytes.



Thanks to the two work modes in BRS, the actual packet size is taken into account in the calculations. This means we can respect the flows configured at either layer, both network and link. In turn, all BRS statistics also reflect traffic at the corresponding layer.

This feature is only available for the following interfaces:

- Ethernet.
- ATM.
- HDLC.
- PPP over synchronous serial line interface.
- PPP over primary ISDN channel.
- Frame Relay over a synchronous serial line.
- Frame Relay over primary ISDN channel.

In the remaining BRS interfaces, calculations are done automatically at the network layer.

Chapter 2 Configuration

2.1 Displaying the BRS Configuration Prompt

To access Bandwidth Reservation System (BRS) commands and configure BRS on your router, do the following:

(1) At the *Config*> prompt, enter **list devices** to see a list of interfaces configured on the router. Use the interface name to configure an interface for bandwidth reservation.

(2) At the *Config*> prompt, enter **feature bandwidth-reservation**.

```
Config>feature bandwidth-reservation
-- Bandwidth Reservation user configuration --
BRS config>
```

(3) At the *BRS Config*> prompt, enter **network** followed by the name of the interface you want to configure for BRS. For example, to configure BRS in the serial0/0 interface enter:

```
BRS config>network serial0/0
BRS [i serial0/0] Config>
```

(4) At the *BRS* [*i serial0/0*] *Config*> prompt, enter **enable**.

```
BRS [i serial0/0] Config>enable
BRS [i serial0/0] Config>
```

(5) For Frame Relay interfaces select Permanent Virtual Circuits (PVCs) by entering **circuit**. At the *BRS* [*i serial0/0*] [*dlci 16*] *Config*> prompt, enter **enable**. In this example, the circuit number is 16.

```
BRS [i serial0/0] Config>circuit 16
BRS [i serial0/0] [dlci 16] Config>enable
BRS [i serial0/0] [dlci 16] Config>
```

(6) Repeat steps 2 through 4 to configure BRS in the particular interface that you have enabled.

(7) At the *BRS* [*i serial0/0*] *Config*> prompt, configure the bandwidth reservation parameters for the selected interface using the appropriate configuration commands. If this is a Frame Relay interface, circuit classes are configured at this prompt.

(8) For Frame Relay interfaces, select PVCs by entering circuit. At the *BRS* [*i serial0/0*][*dlci 16*] *Config*> prompt, configure the bandwidth reservation parameters for the selected circuit using the configuration commands detailed in this chapter. In this example, the circuit number is 16.

(9) Save the configuration and restart your router if needed.

To return to the *Config*> prompt at any time, enter **EXIT** at the *BRS Config*> prompt.

Important

The Bandwidth Reservation System configuration must be carried out after the device interfaces have been configured. However, if you wish to make any further changes to the interface configuration, you often have to remove any existing BRS settings. To do this, use the **clear-block** command.

2.2 Configuration Commands

The following table describes the bandwidth reservation configuration commands. The commands and options marked by an asterisk are only used in the Frame Relay interface layer. The options with a + sign are not used in the Frame Relay interface layer.

Command	Function
ACCESS-LIST	Assigns an IPv4 access list to a class.
ASSIGN	Assigns a circuit*, protocol or filter to a reserved class.
*CIRCUIT**	Selects the DLCI for a Frame Relay Permanent Virtual Circuit.
CLASS	Allocates a designated amount of bandwidth to a user-defined bandwidth class.
CLEAR-BLOCK	Clears the current reservation configuration from the configuration memory (**Note**: This command requires restarting the router).
DEASSIGN	Restores a specified circuit*, protocol or filter to its default class and priority.
DEFAULT-CLASS	Sets the default class and priority to a desired value.
DISABLE	Disables bandwidth reservation on the interface or Frame Relay circuit.

ENABLE	Enables bandwidth reservation on the interface or Frame Relay circuit.
IPV6-ACCESS-LIST	Assigns an IPv6 access list to a class.
LINK-LAYER	Configures BRS work mode at the link layer.
LIST	Displays the currently defined bandwidth classes by their guaranteed percentage and priority queuing values stored in the SRAM. It also displays the assigned protocols and filters. (For Frame Relay, this command provides two levels of information).
MATCH	Assigns classification criteria to a class.
MAX-LATENCY-IN-DRIVER	Limits the maximum number of packets or bytes that can be simultaneously found in the driver
NETWORK-LAYER	Configures the BRS work mode at the network layer.
NETWORK	Selects the interface in which to run bandwidth reservation. Use this command to configure BRS on an interface. **Note**: You must enter this command at the *BRS Config>* prompt **BEFORE** using any other configuration command.
NO	Deletes a previously configured option (such as a class, an access list assignment, bandwidth limit, etc).
QUEUE-LENGTH	Sets a maximum value for the number of packets in a priority queue.
RATE-LIMIT	Specifies a maximum throughput for a Frame Relay interface or circuit. This is measured in kilobits per second.
TAG	Assigns a class and priority to a filter that has been tagged during the configuration of the MAC filtering feature. (For Frame Relay, this command appears in the PVC level prompt).
UNTAG	Removes the tag/tag name relationship and the tag name from the list of assignable filters. (For Frame Relay, this command appears in the PVC level prompt).
UPDATE+	Configures the parameters to update a level indicator based on the traffic rate in the interface.
EXIT	Switches from one BRS level to another, or exits the bandwidth reservation configuration process.

All commands in the above table, except those with an asterisk (which are only for Frame Relay), are valid for configuring bandwidth reservation in interfaces that support this feature (X.25, PPP, ATM subinterface, Frame Relay, GRE IP Tunnel, Ethernet, etc.).

Note

When you enter **clear-block**, **disable**, **enable** and **list** commands from within the BRS interface layer, this action affects or records the bandwidth reservation information configured for the selected interface. When said commands are entered from within the BRS circuit layer, they only affect the Frame Relay Bandwidth Reservation information configured for the Permanent Virtual Circuit (PVC).

Before using the bandwidth reservation commands, keep the following in mind:

- You must use the **network** command to select an interface **BEFORE** you use any other configuration commands. BRS configuration enforces this.
- You must enable the feature, through the enable command, in the selected interface/circuit **BEFORE** using any other configuration command.
- The <class-name> parameter can be written in both upper and lower case.

2.2.1 Access-list

Use **access-list** to assign an IPv4 access list to a class with a given priority. Traffic belonging to the IPv4 access list will be classified as belonging to the class/priority assigned. Access lists are associated with a class and a given priority. Default priority is *Normal*. When choosing the next packet to be sent for a specific class, *Urgent* priority packets are sought first, followed by *High priority*, *Normal* priority and so on.

The four priority types are as follows:

- Urgent
- High
- Normal (default priority)
- Low

Access lists are checked in the same order in which the access-list commands were configured (i.e. as they appear when displaying the configuration). You must bear this in mind when one packet matches more than one access list.

You can also specify that all packets matching the list are marked specifying the set option followed by the type of marking and the value to mark.

Syntax:

```
BRS [i #] Config>access-list <list> <class> [<priority>
                              [set cos {<cos-val> | dscp | precedence}]]
BRS [i #] Config>access-list <list> <class> [<priority>
                              [set dscp <dscp-val>]]
BRS [i #] Config>access-list <list> <class> [<priority>
                              [set precedence <precedence-val>]]
BRS [i #] Config>access-list <list> <class> [<priority>
                              [set tos-octet <tos-val> [mask <mask-val>]]]
BRS [i #] Config>access-list <list> <class> [<priority>
                              [set atm-clp]]
```

list	No. of IPv4 access list to check.
class	Class packets matching the given access list are classified in.
priority	Intraclass priority, i.e. the priority queue (there are 4) in which the packet is classified in.
cos-val	COS value to establish for packets matching the given access list.
set cos dscp	Establishes the COS value according to the packet's DSCP value.
set cos precedence	Establishes the COS value according to the packet's Precedence value.
dscp-val	DSCP value to establish in IP packets matching a given access list.
precedence-val	Precedence value to establish in IP packets matching a given access list.
tos-val	Type of Service octet value to establish in IP packets matching a given access list.
mask-val	If you do not want to mark the 8 bits of the *Type of Service* octet, you can specify a bit mask with the value of bits to mark. For example, if you set tos-val to 96 (01100000 in binary) and mask-val to 254 (11111110 in binary), the first seven bits are marked with binary value 0110000 and the last bit is left unchanged. This way, the Precedence field is marked with 3 and the TOS field with 0.
set atm-clp	Marks the CLP bit of all ATM cells making up the packet.

Example:

Assign access list 100 to class "pepe" with normal priority, while marking the DSCP field 5 in all packets matching this access list.

```
BRS [i #] Config>access-list 100 pepe normal set dscp 5
BRS [i #] Config>
```

2.2.2 Assign

Assigns circuits to a given class when at the Frame Relay interface layer, or specified tags and protocol packets if not at the Frame Relay interface layer. Tags and protocol packets carry an associated priority. The four priority types are:

- Urgent
- High
- Normal (the default priority)
- Low

Syntax 1:

```
BRS [i #] Config>assign {<protocol> | <tag> | <class-name> <priority>
```

Syntax 2:

```
BRS [i #] Config>assign <circuit> <class-name>
```

Example:

```
BRS [i #] Config>assign sna test low
BRS [i #] Config>
```

2.2.3 Circuit

Selects the DLCI for a Frame Relay PVC that needs configuring. You can only run this command from the BRS interface configuration prompt (*BRS* [*i #*] *Config*>).

Syntax:

```
BRS [i #] Config>circuit <permanent-virtual-circuit #>
```

Example:

```
BRS [i #] Config>circuit 16
BRS [i #] Config>
```

When the Frame Relay circuit is enabled, you can use the following commands at the circuit prompt:

- ACCESS-LIST
- ASSIGN
- CLASS
- CLEAR-BLOCK
- DEASSIGN
- DEFAULT-CLASS
- DISABLE
- ENABLE
- LINK-LAYER
- LIST
- MATCH
- MAX-LATENCY-IN-DRIVER
- NETWORK-LAYER
- NO
- QUEUE-LENGTH
- RATE-LIMIT
- TAG
- UNTAG
- UPDATE
- EXIT

2.2.4 Class

Allocates a designated amount of bandwidth to be used either by a group of Frame Relay circuits when at the Frame Relay interface layer, or by protocols, tags etc. defined by the user (if not at the Frame-Relay interface layer).

Syntax:

```
BRS [i #] Config>class <name> <%> [<priority>]
BRS [i #] Config>class <name> queue <max>
BRS [i #] Config>class <name> queue <max> <min>
BRS [i #] Config>class <name> exceed {drop |
                                      classify <class> |
                                      mark-dscp-continue <tos>}
BRS [i #] Config>class <name> rate-limit <cir> [<bc> [<be>]]
BRS [i #] Config>class <name> rate-limit percent <cir> [<bc> [<be>]]
BRS [i #] Config>class <name> rate-limit track
BRS [i #] Config>class <name> set {cos {<cos-val> | dscp | precedence} |
                                   dscp <dscp-val> |
                                   precedence <precedence-val> |
                                   tos-octet <tos-val> [mask <mask-val>]|
                                   atm-clp}
BRS [i #] Config>class <name> update level-indicator <id> value <val>
                                   when-rate-exceeds <rate> [<burst>]
BRS [i #] Config>class <name> wred {dscp-based{ dscp  <dscp-val> <minth> <maxth>
                                     <prob> | exp-weight}
                                     prec-based{ precedence <dscp-val> <minth>
```

```
<maxth> <prob> | exp-weight}}
```

<name>	Name of class to configure.
<%> [priority]	Configures the percentage reserved for a class and, optionally, its priority.
queue <max>	Sets the size for the four class queues.
queue <max> <min>	Sets the size for the four class queues. <*max*> is the size specified under normal circumstances and <*min*> is selected when there is a shortage of resources in the input interface.
exceed drop	When the queue for this class is full, packets with this as destination are dropped.
exceed classify <class>	When the queue for this class is full, packets are redirected to <class>.
exceed mark-dscp-continue <dscp>	When the queue for this class is full, IPv4/IPv6 packets are marked with <tos> value and re-classified.
rate-limit <cir> <bc> <be>	Limits class maximum throughput. Specifies maximum average throughput <cir> and, optionally, the maximum burst size <bc> and the maximum excess burst size <be> allowed.
rate-limit percent <cir> <bc> <be>	Available only on PPP interfaces, it limits class maximum throughput. Calculates the maximum average throughput as the percentage specified in <cir> for the total bandwidth available on the interface. Optionally, you may specify the maximum burst size <bc> and maximum excess burst size <be> allowed.
rate-limit track	Configures a maximum average *throughput* based on the state notified by an *advisor* element. For further information, please see the section on **Rate-limit track**.
set cos <cos-val>	Marks the COS field with a configured value.
set cos dscp	Marks the COS field according to the DSCP value for each IPv4/IPv6 packet.
set cos precedence	Marks the COS field according to the Precedence value for each IPv4/IPv6 packet.
set dscp <dscp-val>	Marks the DSCP field with the configured value.
set precedence <precedence-val>	Marks the Precedence field with the configured value.
set tos-octet <tos-val>	Marks the *Type of Service* (IPv4) byte or *Traffic class* (IPv6) with the configured value.
set tos-octet <tos-val> mask <mask-val>	Marks bits specified with <maskval> in *Type of Service* (IPv4) or *Traffic class* (IPv6), setting them to the value configured in <tosval>.
set atm-clp	Marks the CLP bit for all ATM cells making up the packet.
update level-indicator <id> value <val> when-rate-exceeds <rate> [<burst>]	Updates the level indicator <id>, which can be used to configure an NSLA feature filter (see Dm754-I). Total value <val> is added or subtracted to the level indicator when all traffic in the class exceeds the rate <rate> (in kbits/sec) or goes below said rate. Optionally, you can also configure the burst (in kbits).
wred dscp-based	Enables WRED in a class based on the packet DSCP.
wred dscp <dscp-val> <minth> <maxth> <prob>	Configurable parameters for each DSCP value. DSCP value <*dscp-val*> refers to parameters that are going to be configured, <*minth*> minimum threshold after which packets begin to be dropped, <*maxth*> maximum threshold after which all packets are dropped, and <*prob*> is the discard probability applied when the average size of the queue ranges between *maxth* and *minth*.
wred exp-weight	Constant used to calculate the average queue size. Default is 9.
wred prec-based	Enables WRED in a class based on packet precedence.
wred precedence <dscp-val> <minth> <maxth> <prob>	Configurable parameters for each precedence value. Precedence value <*prec-val*> refers to parameters that are going to be configured, <*minth*> minimum threshold after which packets are dropped, <*maxth*> maximum threshold after which all packets are dropped and <*prob*> is the discard probability when the mean queue size ranges between *maxth* and *minth*.
wred exp-weight	Constant used to calculate the average queue size. Default is 9.

Example 1:

```
BRS [i #] Config>class alpha 10
BRS [i #] Config>
```

Example 2:

```
BRS [i #] Config>class beta 10 real-time
BRS [i #] Config>
```

You can specify a priority for the new class. See section 2 (Priority) in this manual. Default is *Normal*.

Example 3:

Configuring a class with a 30% bandwidth guarantee but with a maximum throughput of 40 kbps.

```
BRS [i #] Config>class beta 30
BRS [i #] Config>class beta rate-limit 40
```

Command history:

Release	Modification
11.00.06	The *queue <max>* option is introduced. The *queue <max> <min>* option is obsolete.
11.01.02	The *queue <max>* option is introduced. The *queue <max> <min>* option is obsolete. Use the *queue <max>* option instead.

2.2.5 Clear-block

Clears the Bandwidth Reservation configuration for the current interface or Frame Relay PVC.

Syntax:

```
BRS [i #] Config>clear-block [yes]
```

- optional parameter **yes** allows the device to run an operation without prompting the user to confirm first. If this parameter is set to **yes**, no such confirmation is required. If not, the device prompts the user for confirmation.

Example:

```
BRS [i #] Config>clear-block
You are about to clear BRS configuration information for this interface.
Are you sure you want to do this (Yes/No)? y
BRS [i #] Config>
```

Command history:

Release	Modification
11.01.06	The "[yes]" option was added as of version 11.01.06

2.2.6 Deassign

Enter **deassign** to restore a specified circuit (only at the Frame Relay interface layer), protocol, or tag to its default class and priority.

Syntax 1:

```
BRS [i #] Config>deassign <cvp>
```

Syntax 2:

```
BRS [i #] Config>deassign <protocol or tag> <class> <priority>
```

Example:

```
BRS [i #] Config>deasign sna test low
BRS [i #] Config>
```

2.2.7 Default-class

Defines the class/priority to be assigned to packets not explicitly classified through any other mechanism (**access-list**, **IPv6-access-list** or **assign** commands). If no value has been previously assigned, system default values are used (class *default*, priority *normal*).

Syntax:

```
BRS [i #] Config>default-class <class> <priority>
```

Example:

```
BRS [i #] Config>default-class test low
BRS [i #] Config>
```

2.2.8 Disable

Disables bandwidth reservation on the interface or Frame Relay circuit. To verify that bandwidth reservation is disabled, enter **list**.

Syntax:

```
BRS [i #] Config>disable
```

Example:

```
BRS [i #] Config>disable
BRS [i #] Config>
```

Command history:

Release	Modification
11.00.03	As of version 11.00.03 this command is dynamic, except for Frame Relay interfaces.
11.00.03.01.01	As of version 11.00.03.01.01 this command is dynamic, except for Frame Relay interfaces.
11.01.00	As of version 11.01.00 this command is dynamic, except for Frame Relay interfaces.

2.2.9 Enable

Enables Bandwidth Reservation on the interface or Frame Relay circuit.

Syntax:

```
BRS [i #] Config>enable
```

Example:

```
BRS [i #] Config>enable
BRS [i #] Config>
```

Command history:

Release	Modification
11.00.03	As of version 11.00.03 this command is dynamic, except for Frame Relay interfaces.
11.00.03.01.01	As of version 11.00.03.01.01 this command is dynamic, except for Frame Relay interfaces.
11.01.00	As of version 11.01.00 this command is dynamic, except for Frame Relay interfaces.

2.2.10 IPv6-access-list

Enter **IPv6-access-list** to assign an IPv6 access list to a class with a given priority. Traffic belonging to said IPv6 access list is classified as belonging to the class/priority to which it is assigned. Access lists are associated with a class with a given priority, default priority being *Normal*. When choosing the next packet to be sent for a particular class, packets with *Urgent* priority are sought first, followed by *High* priority if there are none, followed by *Normal* priority if there are none, and so on. The four priority types are:

- URGENT
- HIGH
- NORMAL (the default setting)
- LOW

Access lists are checked in the same order in which the **IPv6-access-list** commands were configured (i.e. as they appear when displaying the configuration). You should take this into account when one packet matches more than one access list.

You can have all packets matching the list labeled, specifying the **set** option, the type of marking and the value to label.

Syntax:

```
BRS [i #] Config>ipv6-access-list <list> <class> [<priority>
                                [set cos {<cos-val> | dscp | precedence}]]
BRS [i #] Config>ipv6-access-list <list> <class> [<priority>
                                [set dscp <dscp-val>]]
BRS [i #] Config>ipv6-access-list <list> < class> [<priority>
                                [set precedence <precedence-val>]]
BRS [i #] Config>ipv6-access-list <list> < class> [<priority>
                                [set traffic-class <tc-val> [mask <mask-val>]]]
BRS [i #] Config>ipv6-access-list <list> < class> [<priority>
                                [set atm-clp]]
```

list	No. of IPv6 access list to check.
class	Class packets matching the given access list are classified in.
priority	Intraclass priority, i.e. the priority queue (there are 4) in which the packet is classified in.
cos-val	COS value to be established for packets matching the given access list.
set cos precedence	Establishes the COS value according to the packet Precedence value.
set cos dscp	Establishes the COS value according to the packet DSCP value.
dscp-val	DSCP value to be established in IPv6 packets matching the given access list.
tc-val	*Traffic Class* octet value to be established in the IPv6 packets matching the given access list.
mask-val	If you do not want to mark the 8 bits of the *Traffic Class* octet, you can specify a bit mask with the value of the bits to mark. For example, if you set tc-val to 96 (01100000 in binary) and mask-val to 254 (11111110 in binary), the first seven bits are marked with binary value 0110000 and the last bit is left unchanged.
set atm-clp	Marks the CLP bit of all ATM cells matching up the packet.

Example:

Assign the IPv6 list1 access list to class1 with normal priority and mark the DSCP field 5 in all IPv6 packets matching this access list.

```
BRS [i #] Config>ipv6-access-list list1 class1 normal set dscp 5
BRS [i #] Config>
```

2.2.11 Link-layer

Triggers bandwidth calculations in BRS at layer 2 or the link layer.

When executing the **link layer** command calculation, you can specify an offset value (in bytes) to take into account.

Syntax:

```
BRS [i #] Config>link-layer [offset <value>]
```

Example:

```
BRS [i #] Config>link-layer offset 4
```

2.2.12 List

Displays currently defined bandwidth classes by their guaranteed percentage rates and priority queuing values stored in SRAM. This command also displays all assigned protocols, access lists and classification criteria.

Syntax:

```
BRS [i #] Config>list
```

Example:

```
BRS [i #] Config>list
```

Depending on the prompt you enter **list** at, various outputs appear. You can enter **list** from the following prompts:

BRS Config>

BRS [*i x25-node*] *Config*> (for the X.25 interface)

BRS [*i serial0/0*] [*dlci 17*] *Config*> (for circuit 17 in number 1 Frame Relay interface)

For example, if you enter the **list** command at *BRS Config*>, the following output is obtained:

Example:

```
BRS config>list
Bandwidth Reservation is available for 5 interfaces.
  Interface        State
---------------    -----
ethernet0/0        Enabled
ethernet0/1        Disabled
serial0/0          Enabled
x25-node           Disabled
ppp1               Disabled
BRS config>
```

We can see that BRS features are available for Frame Relay, PPP and X.25 interfaces.

The following output appears when you enter **list** at *BRS* [*i x25-node*] *Config>:*

Example:

```
BRS [i x25-node] Config>list

BANDWIDTH RESERVATION listing from SRAM
bandwidth reservation is enabled
bandwidth reservation is operating on the network layer
interface name x25-node
maximum queue length 10 minimum queue length 3
Priority low total bandwidth allocated 0
Priority normal total bandwidth allocated 50
Priority high total bandwidth allocated 0
Priority real-time total bandwidth allocated 100
total classes defined (counting one local and one default) 3

class control has 100% bandwidth allocated
  this is the default class for control traffic.

class local has 20% bandwidth allocated
  this is the default class for locally generated traffic.
class default has 80% bandwidth allocated
  this is the default class for unclassified traffic.

default class is default with priority NORMAL
BRS [i x25-node] Config>
```

The above list appears by default when you enter the bandwidth reserve configuration for the first time for an already-enabled X.25 interface. There are always three classes available to begin with:

- CONTROL class: this class can never be deleted and is reserved for control traffic (i.e. for all traffic whose function is considered basic to ensure connectivity). ARP traffic, for example, is considered control traffic.
- LOCAL class: this class can never be deleted and is reserved for traffic generated locally in the device, i.e. all traffic that does not come from switching, but is generated internally and mainly comes from routing protocols (RIP, OSPF), generation of maintenance packets, pings, etc.
- DEFAULT class: as the name suggests, this is the default class where all available protocols in the device are initially assigned. At first, 40% of the weight is allocated. As the sum of the weights for all classes is 10% (local) + 40% (default) = 50%, the default class is guaranteed 80% of the available bandwidth, as you can see in the listing.

The rest of the values are those used by default.

Note

The list does not show the weight assigned to each class but rather the percentage of available bandwidth guaranteed for each class.

The weight is the value configured through the **class** command and the percentage of available bandwidth depends on the sum of the weights of all classes with the same priority.

The following output appears when you enter **list** at *BRS* [*i serial0/0*] *Config*>:

Example:

```
BRS [i serial0/0] Config>list

BANDWIDTH RESERVATION listing from SRAM
bandwidth reservation is enabled
interface name serial0/0
maximum queue length 10
total bandwidth allocated 10
total circuit classes defined (counting one default) 1

class default has 100% bandwidth allocated
  the following circuits are assigned:
      16
      17
default class is default
BRS [i serial0/0] Config>
```

The above list appears by default when you first enter the bandwidth reservation configuration for a Frame Relay interface that is already enabled. There is always one class available to begin with:

- DEFAULT class: as the name suggests, this is the default class. This is the class to which all circuits with enabled bandwidth reservation are initially assigned. This class contains two circuits (16 and 17), meaning bandwidth reservation is enabled for these two circuits. The initial bandwidth assigned for this type is 10%.

The rest of the values shown are those used by default.

The following output appears when you enter list at *BRS* [*i serial0/0] [dlci 17] Config*>:

Example:

```
BRS [i serial0/0] [dlci 17] Config>list

BANDWIDTH RESERVATION listing from SRAM
bandwidth reservation is enabled
bandwidth reservation is operating on the network layer
interface name serial0/0 circuit number 17
maximum queue length 10
Priority low total bandwidth allocated 0
Priority normal total bandwidth allocated 50
Priority high total bandwidth allocated 0
Priority real-time total bandwidth allocated 100
total classes defined (counting one local and one default) 3

class control has 100% bandwidth allocated
  this is the default class for control traffic.
class local has 20% bandwidth allocated
  this is the default class for locally generated traffic.

class default has 80% bandwidth allocated
  this is the default class for unclassified traffic.

default class is default with priority NORMAL
BRS [i serial0/0] [dlci 17] Config>
```

When we first make this list for circuit 17 (once bandwidth reservation is enabled), we get something very similar to the X.25 interface example. Three classes are always available to begin with:

- CONTROL class: this class can never be deleted and is reserved for control traffic (i.e. for all traffic whose function is necessary to ensure connectivity). ARP traffic, for example, is considered control traffic
- LOCAL class: this class can never be deleted and is reserved for traffic generated locally in the device, i.e. all traffic that does not come from switching but is generated internally and comes primarily from routing protocols (RIP, OSPF), generation of maintenance packets, pings, etc.
- DEFAULT class: as the name suggests, this is the default class to which all available protocols in the device are initially assigned and which has, in principle, a 40% weight allocation.

The remaining values are the ones used by default.

Note

In Frame Relay, there are two command levels: the interface layer and the circuit layer.

2.2.13 Match

Enter **match** to assign classification criteria to a class with a given priority. Traffic belonging to said classification criteria is classified as belonging to the class/priority it is assigned to. Classification criteria are associated with a class with a given priority. Default is *Normal*. When choosing the next packet to be sent for a specific class, *Urgent* priority packets are sought first, followed by *High* priority then *Normal*, etc. The four priorities are as follows:

- Urgent
- High
- Normal (default priority)
- Low

Classification criteria are checked in the same order **match** commands were configured in (i.e. as they appear when displaying the configuration).

You can have all packets matching the list labeled, specifying the **set** option, the type of marking and the value to label.

The following classification criteria can be used:

2.2.13.1 Label

Allows you to classify the packet depending on the label it contains. The label value can range between 0 and 99.

Syntax:

```
BRS [i #] Config>match label <label> class <class> [<priority>
                              [set cos {<cos-val> | dscp | precedence}]]
BRS [i #] Config>match label <label> class <class> [<priority>
                              [set dscp <dscp-val>]]
BRS [i #] Config>match label <label> class <class> [<priority>
                              [set precedence <precedence-val>]]
BRS [i #] Config>match label <label> class <class> [<priority>
                              [set tos-octet <tos-val> [mask <mask-val>]]]
BRS [i #] Config>match label <label> class <class> [<priority>
                              [set atm-clp]]
```

label	No. of label to check.
class	Class packets matching the given access list are classified in.
priority	Intraclass priority, i.e. the priority queue (there are 4) in which the packet is classified in.
cos-val	COS value to be established for packets matching the given label.
set cos dscp	Establishes the COS value according to the IPv4/IPv6 packet DSCP value.
set cos precedence	Establishes the COS value according to IPv4/IPv6 packet precedence value.
dscp-val	DSCP value to be established in the IPv4/IPv6 packets matching the given label.
precedence-val	Precedence value to be established in the IPv4/IPv6 packets matching the given label.
tos-val	For IPv4 packets matching the set label, this sets the *Type of Service* octet value to be established in the IPv4 header. For IPv6 packets matching the set label, this establishes the *Type of Service* octet value to be established in the IPv6 header.
mask-val	If you do not want to label the 8 bits of the *Type of Service* (IPv4) octet or *Traffic Class* (IPv6), you can specify a bit mask with the value of the bits to label. For example, if you configure tos-val with 96 (01100000 in binary) and mask-val with 254 (11111110 in binary), this labels the first seven bits with binary value 0110000 leaving the last bit unaltered, so the Precedence field is marked with a value of 3 and the TOS with 0.
set atm-clp	Marks the CLP bit for all ATM cells matching the packet.

Example:

Assign label 30 to the *label* class with normal priority and mark the CoS field 5 in all packets matching the label.

```
BRS [i #] Config>match label 30 class etiqueta normal set cos 5
BRS [i #] Config>
```

2.2.14 Max-latency-in-driver

Limits the maximum number of packets or bytes that can be simultaneously found in the driver.

Once the packets or bytes have reached the driver, they are transmitted in the same order in which they were sent. If, at some point, a packet or byte with higher priority reaches the BRS subsystem, it will have to wait until the packets in the driver have been transmitted (even if these packets or bytes have lower priorities).

If a lot of data is delivered to the driver, efficiency (maximum speed) is increased but latency also increases.

By restricting the amount of data delivered to the driver, we can reduce latency but also efficiency (maximum speed).

In common scenarios, the default behavior is sufficient. In tight scenarios, the data delivery criteria to the driver can by customized through the **max-latency-in-driver** command. This helps achieve an optimal balance between latency and throughput.

The limits used can be consulted through the **queue** monitoring command.

Limits can be applied in different ways, always opting for the most restrictive at all times.

2.2.14.1 Max-latency-in-driver packets <n>

Sets the maximum number of packets in the driver. This limit is useful on high speed lines where processing in bursts is normal to achieve the required speeds.

2.2.14.2 Max-latency-in-driver bytes <n>

Sets the maximum number of bytes in the driver. This limit is useful for low speeds, where the limiting factor is the latency introduced by large packets (especially in links where fragmentation is not applied). A typical scenario would be a low-speed ADSL line where voice streams (small packets) compete with data flows (large packets).

The rapport between latency, speed and length (data volume) is shown in the following formula:

Latency × Speed = Length

For example, if you want to limit the latency to 30 ms in a 512 kbps ADSL line when G729 voice packets are transmitted, the calculation is as follows:

Max_length = 30 ms × 512 kbps / 8 (bits/byte) = 1920 bytes

Since the voice packet itself occupies 2 ATM cells, 106 bytes must be subtracted to obtain the driver's maximum queue length:

Queue_length = *Max_length* – *Frame_length* = 1920 bytes – 106 bytes = 1814 bytes

This is configured as follows:

```
BRS [i #] Config>link-layer
BRS [i #] Config>max-latency-in-driver bytes 1814
```

Syntax:

```
BRS [i #] Config>max-latency-in-driver packets <n>
BRS [i #] Config>max-latency-in-driver bytes <n>
```

Example:

```
BRS [i #] Config> max-latency-in-driver bytes 2000
BRS [i #] Config>
```

Command history:

Release	Modification
10.09.18	This command was introduced as of version 10.09.18.
11.00.02	This command was introduced as of version 11.00.02.

2.2.15 Max-packets-in-driver

Limits the maximum number of packets that can be simultaneously found in the driver.

Once the packets have reached the driver, they are transmitted in the same order in which they were sent. If, at some point, a packet with higher priority reaches the BRS subsystem, this packet will have to wait until the packets in the driver have been transmitted (even if these packets have lower priorities).

In fact, the maximum delay that non-priority traffic causes to priority traffic can be calculated as (MTU * **max-packets-in-driver**)/line speed. Therefore, in environments where fragmentation is not possible (high MTUs) and lines operate at a low speed, you may want to limit this value to one to ensure priority traffic suffers the least possible delay.

This parameter's default value dynamically varies depending on the MTU and the line speed, meaning there is often no need to configure it. ADSL scenarios are the exception to this rule, since they default to a value of two. It's a good idea to set this to one for scenarios with very delay-sensitive traffic and low speed ADSL lines (128 Kbps).

The drawback to configuring low **max-packets-in-driver** values is that efficiency may drop in situations where the router has a high load (CPU use is very high), since the software (and, consequently, the CPU) is responsible for feeding the driver with packets to be transmitted.

Syntax:

```
BRS [i #] Config>max-packets-in-driver <packet number>
```

Example:

```
BRS [i #] Config>max-packets-in-driver 1
BRS [i #] Config>rat
```

Command history:

Release	Modification
10.09.18	This command was replaced by the **max-latency-in-driver** option.
11.00.02	This command was replaced by the **max-latency-in-driver** option.

2.2.16 Network-layer

Causes the bandwidth calculations in BRS to be carried out at layer 3 or the network layer.

Syntax:

```
BRS [i #] Config>network-layer
```

Example:

```
BRS [i #] Config>network-layer
```

2.2.17 Network

Accesses the bandwidth reservation configuration for the selected interface.

Syntax:

```
BRS Config>network <interface>
```

Example:

```
BRS config>network serial0/0
BRS [i serial0/0] Config>
```

2.2.18 No

Deactivates BRS configuration parameters.

2.2.18.1 no access-list

Removes the association between an IPv4 access list and a class.

Syntax:

```
BRS [i #] Config>no access-list <list>
```

Example:

Removes the association between the IPv4 100 access list and the class it was associated with.

```
BRS [i #] Config>no access-list 100
BRS [i #] Config>
```

2.2.18.2 no class

Eliminates a previously configured bandwidth class from a specified interface, a Frame Relay circuit or one of the parameters configured here.

Syntax 1:

```
BRS [i #] Config>no class <name>
```

<name> Name of the class to eliminate.

Syntax 2:

```
BRS [i #] Config>no class <name> queue
BRS [i #] Config>no class <name> exceed
BRS [i #] Config>no class <name> rate-limit
BRS [i #] Config>no class <name> rate-limit track ?
  2g-cell-advisor    Applied when established cellular link technology is 2G
  3g-cell-advisor    Applied when established cellular link technology is 3G
  4g-cell-advisor    Applied when established cellular link technology is 4G
BRS [i #] Config>no class <name> set {cos | dscp | precedence | tos-octet}
BRS [i #] Config>no class <name> update
```

<name>	Name of class to configure.
queue	Removes the queue size configuration of the four class queues and retrieves the default values.
exceed	Packets exceeding the queue capacity will not be reclassified.
rate-limit	Removes the bandwidth limitation (traffic shaping) imposed on the class.
rate-limit track	Removes the *rate-limit* configured to track an *advisor* element.
set cos	Removes the COS marking.
set dscp	Removes the DSCP field marking.
set precedence	Removes the Precedence field marking.
set tos-octet	Removes the *Type of Service* field marking.
update	Cancels the updating of a level indicator based on the total traffic in the class.

Example:

```
BRS [i #] Config>no class ip
BRS [i #] Config>
```

2.2.18.3 no ipv6-access-list

Removes the association between an IPv6 access list and a class.

Syntax:

```
BRS [i #] Config>no ipv6-access-list <listaID>
```

Example:

Removing the association between the IPv6 list1 access list and the class it was associated with.

```
BRS [i #] Config>no ipv6-access-list list1
BRS [i #] Config>
```

2.2.18.4 no match

Removes the association between a match criteria and a class.

Syntax:

```
BRS [i #] Config>no match <criteria> <criteria_data>
```

Example:

Removing the association between the match label 30 criteria and the class it was associated with.

```
BRS [i #] Config>no match label 30
BRS [i #] Config>
```

2.2.18.5 no max-latency-in-driver

Disables the aforementioned command's configuration.

Syntax:

```
BRS [i #] Config>no max-latency-in-driver [pa| byte| bytes]
```

Example:

```
BRS [i #] Config>no max-latency-in-driver packets
```

Command history:

Release	Modification
10.09.18	This command was introduced as of version 10.09.18.
11.00.02	This command was introduced as of version 11.00.02.

2.2.18.6 no max-packets-in-driver

Sets the default value for this parameter. This parameter's default value varies dynamically, depending on the MTU and the line speed.

Syntax:

```
BRS [i #] Config>no max-packets-in-driver
```

Example:

```
BRS [i #] Config>no max-packets-in-driver
BRS [i #] Config>
```

Command history:

Release	Modification
10.09.18	This command was replaced by the **no max-latency-in-driver** option.
11.00.02	This command was replaced by the **no max-latency-in-driver** option.

2.2.18.7 no queue-length

Configures the default value of the interface queues, leaving the queue settings per class intact. The queues configured by default have a minimum of 3 packets in congestion and a maximum of 10.

Syntax:

```
BRS [i #] Config>no queue-length
```

Example:

```
BRS [i #] Config>no queue-length
BRS [i #] Config>
```

2.2.18.8 no rate-limit

Removes the maximum throughput limit placed on an interface. The maximum throughput is then equal to the actual interface speed.

Syntax:

```
BRS [i #] Config>no rate-limit
```

Example:

```
BRS [i #] Config>no rate-limit
BRS [i #] Config>
```

2.2.18.9 no rate-limit track

Deletes a configured *rate-limit* to track an *advisor* element.

Syntax:

```
BRS [i #] Config>no rate-limit track ?
  2g-cell-advisor    Applied when established cellular link technology is 2G
  3g-cell-advisor    Applied when established cellular link technology is 3G
  4g-cell-advisor    Applied when established cellular link technology is 4G
```

2.2.18.10 no update

Removes the feature that updates an NSLA level indicator based on the traffic rate in the interface or Frame Relay virtual circuit.

Syntax:

```
BRS [i #] Config>no update
```

Example:

```
BRS [i #] Config>no update
BRS [i #] Config>
```

2.2.19 Queue-length

Caution

Do not use this command unless it is absolutely essential. bintec recommends the queue length default values for most users. If the values set for queue length are too high, you may seriously degrade the performance of your router.

Sets the number of packets the router can queue in each BRS priority queue. Each BRS class has a priority value assigned to its protocols, filters and tags. Each priority queue can hold as many packets as specified with this command.

This command sets the maximum number of outbound packets that can be queued in each BRS priority queue.

If you issue a **queue-length** command for an interface that is not Frame Relay, this command sets the queue length values for each priority queue belonging to each BRS class defined by the interface.

If you issue a **queue-length** command for a Frame Relay in arfahis: like this: like this: *BRS* [*i serial0/0*] *Config*>), the command sets the default queue length values for each priority queue belonging to each BRS class defined by each of the interface's permanent virtual circuits.

If you issue a **queue-length** command for a Frame Relay PVC (at a prompt like this: *BRS* [*i serial0/0*] [*dlci 16*] *Config*>), the command sets the queue length values for each priority queue belonging to each BRS class defined by the PVC. These values override the default queue length values set for the interface.

Caution

You will need to use this command to increase the size of the queues when, for example, the circuit is operating with some kind of fragmentation.

Syntax:

```
BRS [i #] Config>queue-length <maximum-length>
BRS [i #] Config>queue-length <maximum-length> <minimum-length>
```

Example:

```
BRS [i #] Config>queue-length 10
BRS [i #] Config>
```

Command history:

Release	Modification
11.00.06	**queue-length <max>** has been introduced. The *queue-length* <*max*> <*min*> option is obsolete.
11.01.02	**queue-length <max>** has been introduced. The *queue-length* <*max*> <*min*> option is obsolete.

2.2.20 Rate-limit

Limits the maximum throughput for an interface or Frame Relay virtual circuit.

The maximum throughput is specified in kilobits per second, while the burst and the excess bursts are specified in kilobits. The quantity of transmitted bytes is measured at the IP layer or Link layer (see **link-layer** and **network-layer** commands), so the final throughput may be somewhat higher, depending on the headers introduced by the encapsulation used in the interface in question.

In PPP interfaces, the *percent* option allows you to configure the maximum throughput as a percentage value. Here, maximum throughput in Kbps is calculated by applying said percentage to the total bandwidth available on the interface. On interfaces configured as multilink PPP, if the number of links dynamically changes, this configuration allows maximum throughput to adapt to the new total bandwidth.

Syntax:

```
BRS [i #] Config>rate-limit <throughput> [burst] [excess-burst]
BRS [i #] Config>rate-limit percent <throughput> [burst] [excess-burst]
```

Example 1:

```
BRS [i #] Config>rate-limit 30 20
BRS [i #] Config>
```

Example 2:

```
BRS [i #] Config>rate-limit percent 90 128 128
BRS [i #] Config>
```

Note

Having an interface layer **rate-limit** is the same as having a line speed equal to the configured value. Consequently, in a congested state, the statistics referring to packets discarded due to the rate-limit value set will refer to packets discarded as a result of queue overflows.

Note

The "percent" option can also be used in *rate limit track*, *class <cname> rate-limit* and *class <cname> rate-limit track*. Its definition can be found at the beginning of this section.

2.2.20.1 Rate-limit track

Using the **track** option, you can configure different rate-limits to be applied according to the state notified by an *advisor*. You can only configure one **rate-limit** for each possible state notified by an *advisor*. No two rate-limits can ever be applied simultaneously.

When the interface on which the BRS is configured has a cellular-type base interface, you can configure a cell *advisor*. This type of *advisor* monitors link technology in the *cellular* base interface and notifies BRS when a change occurs. Upon such notification, the configured rate-limit for this new technology (2G, 3G, etc) is applied. If nothing has been specified for this state, then the configured rate-limit is applied without the **track** option.

This **cell-advisor** tracking feature is only available in PPP and Direct-IP interfaces.

Syntax:

```
BRS [i #] Config> rate-limit track ?
  2g-cell-advisor    Applied when established cellular link technology is 2G
```

```
  3g-cell-advisor    Applied when established cellular link technology is 3G
  4g-cell-advisor    Applied when established cellular link technology is 4G
BRS [i #] Config> rate-limit track 2g-cell-advisor {<cir> | percent <cir>} <bc> <be>
BRS [i #] Config> rate-limit track 3g-cell-advisor {<cir> | percent <cir>} <bc> <be>
BRS [i #] Config> rate-limit track 4g-cell-advisor {<cir> | percent <cir>} <bc> <be>
```

<cir>	*Committed information rate*. Maximum average *Throughput* configured in Kbps or as a percentage of the total line bandwidth.
<bc>	*Committed burst*. Maximum burst size allowed in kilobits.
<be>	*Excess burst*. Maximum excess burst size allowed in kilobits.

Example:

```
BRS [i #] Config> rate-limit track 3g-cell-advisor 500 1000 1000
BRS [i #] Config>
```

In this example, we have configured a rate-limit of 500 kbps on the interface. This applies when the **cell-advisor** confirms a 3G link has been established in the *cellular* base interface.

2.2.20.1.1 Class <cname> rate-limit track

This feature is also available at the class layer.

Example:

```
BRS [i #] Config>class myclass rate-limit track 3g-cell-advisor percent 80 3000 300
```

In this example, a maximum average *throughput* of 80% of the line for *myclass* has been configured and is applied when the **cell-advisor** reports that link *cellular* technology in the base interface is 3G.

Note

In a BRS class, configuration of a rate-limit using the track option is not compatible with WRED.

2.2.21 Tag

Associates a tag with a MAC filter. Packets are tagged through the MAC filtering feature (**mac-filtering**), and then classified in BRS through MAC filters (tag1 to tag5). Filters are allocated to a class and a priority by entering **assign**.

Syntax:

```
BRS [i #] Config>tag <tag #> <filter>
```

Example:

```
BRS [i #] Config>tag 3 tag1
BRS [i #] Config>
```

2.2.22 Untag

Removes the tag-filter relationship. A tag can only be removed if it is not assigned to any class.

Syntax:

```
BRS [i #] Config>untag <tag #>
```

Example:

```
BRS [i #] Config>untag 3
BRS [i #] Config>
```

2.2.23 Update

Adds a feature to update a level indicator, determined by index <id>, used in NSLA (see manual *bintec Dm754-I*). To do this, a traffic-shaping mechanism (such as **rate-limit**) is used, although in this case, it doesn't limit traffic, it merely updates an indicator: the numeric value <val> (which can be positive or negative) is added to the level indicator when the total traffic in the interface or Frame-Relay virtual circuit exceeds <rate>, and is subtracted if it goes below said rate. Optionally, you can configure <burst> as the **rate-limit** command to take into account when the specified rate is exceeded. If <burst> isn't configured, it is calculated by default as the number of kilobits that can be transmitted in 125 milliseconds at <rate>. <rate> is configured in kilobits per second and <burst> in kilobits.

Syntax:

```
BRS [i #] Config>update level-indicator <id> value <val> when-rate-exceeds <rate> [<burst>]
```

Example:

```
BRS [i #] Config>update level-indicator 1 value 10 when-rate-exceeds 1000
BRS [i #] Config>
```

2.2.24 Exit

Returns to the previous prompt.

Syntax:

```
BRS Config>exit
```

Example:

```
BRS Config>exit
Config>
```

2.3 Fixed-number-snmp

Sets the persistence of bandwidth-reservation (BRS) interface and class indexes over time, regardless of whether new interfaces/classes are added or existing ones deleted. With this new functionality, the order of interfaces and classes will match the order of creation (i.e., new interfaces and classes will be added at the end of their corresponding lists, instead of being grouped according to their priority). The order of interfaces and classes will remain the same even after the device is restarted.

The physical interfaces activated by the router's license always appear first and in a fixed position.

This functionality is enabled at the fixed-number-snmp menu.

For more information on how the *fixed number-snmp* feature works in interfaces, please see manual bintec Dm772-Configuration Interfaces.

Commands:

INIT

This command triggers the configuration of the feature using the router's current state. All interfaces and classes are scanned, with one "*interface*" command and *"class"* command created for each (together with an index containing the position value of the interface).

The "init" command will not appear in the configuration and its scanning process is activated when saving the configuration (save). Now is when the index persistence information of the interfaces and classes will appear in the configuration.

The *"interface"*, *"class"*, *"last-index-ifc"* and *"val-checksum"* subcommands are automatically created when "*init*" is executed and then the configuration is saved or when interfaces or classes are added or removed. This means *the user cannot create or delete them at will.*

- *interface*. This subcommand includes the name of the interface and the index value that matches its order of creation.
- *last-index-ifc* . This subcommand will take the index value of the latest interface added.
- *class*. This subcommand includes the name of the interface, the name of the class, the circuit to which it belongs (where applicable), and the index value that matches its order of creation. In this case, *last-index* will be created as one more class.
- *val-checksum*. The checksum value is a way to verify that the configurations of the *"interface"* and *"class"* sub-

commands are correct, in order to prevent the router from misbehaving or crashing.

LIST

This command displays the status of the “init” command. If we execute the “init” command but do not save, the "list" option will tell us it is enabled. Once saved, the “init” command will be disabled until it is executed again.

CLEAR

This command removes all *fixed-number-snmp* feature configurations and deactivates the functionality. The order previously configured will not change until the router is restarted.

How to use it:

This functionality can be used as follows:

(a) To activate this feature you have to run the *“init”* command. It will not appear in the configuration and once we save it, the list of *“interface”* and *“class”* subcommands will be created inside the menu based on their current order and corresponding index value. Until the configuration is saved, this “init” command operation will not be performed. To know the status of the functionality, you must execute "list".

(b) Once the feature is activated, there are four possible actions:

  (a) *Add a new interface.* The interface will appear in the *fixed-number-snmp* menu, always in the last position when created by the *“last-index-ifc”* command.

  (b) *Delete an interface*. In this case, the interface is removed from the *fixed-number-snmp* menu list but the order of the remaining interfaces and their indexes is kept.

  (c) *Add a new class.* The class will appear in the *fixed-number-snmp* menu, always in the last position of the interface to which it belongs, as established by the *“class <interface> <circuit> last-index”* subcommand.

  (d) *Delete a class*. In this case, the class is removed from the *fixed-number-snmp* menu list but the order of the remaining classes and their indexes is kept.

(c) To disable this functionality, execute the *“clear”* command and restart the computer to apply the changes.

Note

The user must not manipulate the configuration of the *fixed-number-snmp* menu and should take great care when copying a configuration from one router to another, since this could cause the device to misbehave or malfunction.

Example of use:

(a) Initial state of the router:

```
Config$ feature bandwidth-reservation
-- Bandwidth Reservation user configuration --
BRS config$
BRS config$ show config
; Showing Menu and Submenus Configuration for access-level 15 ...
    network ethernet0/0
       enable
       class control 100 real-time
       class local 10
       class default 40
       class test1 20 high
       class test2 1 low
       class test3 1 low
       exit

    network fr1
       enable
       circuit 16
          enable
          class control 100 real-time
          class local 10
          class default 40
          exit
       exit
    network fr2
```

```
      enable
      circuit 24
         enable
         class control 100 real-time
         class local 10
         class default 40
         class test4 2 low
         exit
      exit
```

(b) Activation of the *fixed-number-snmp* feature. To know the status of the “init” command of fixed-number-snmp feature, you must use the “list” command:

```
Config$ fixed-number-snmp
-- Fixed configuration --
FIXED config$ list
The init command is disabled

FIXED config$ init
FIXED config$ list
The init command is enabled

FIXED config$
FIXED config$ show config
; Showing Menu and Submenus Configuration for access-level 15 ...

FIXED config$ exit
Config$ save yes
Building configuration as text... OK
Writing configuration... OK on Flash as flash
Config$ fixed-number-snmp
-- Fixed configuration --
FIXED config$ show config
; Showing Menu and Submenus Configuration for access-level 15 ...

interface ethernet0/0 1
interface ethernet0/1 2
interface cellular0/0 3
interface cellular0/1 4
interface cellular0/2 5
interface cellular10/0 6
interface cellular10/1 7
interface cellular10/2 8
interface wlan0/0 9
interface fr1 10
interface fr2 11
;
class ethernet0/0 0 control 1
class ethernet0/0 0 last-index 7
class ethernet0/0 0 local 2
class ethernet0/0 0 default 3
class ethernet0/0 0 test1 4
class ethernet0/0 0 test2 5
class ethernet0/0 0 test3 6
;
class fr1 16 control 1
class fr1 16 last-index 4
class fr1 16 local 2
class fr1 16 default 3
;
class fr2 24 control 1
class fr2 24 last-index 5
class fr2 24 local 2
class fr2 24 default 3
class fr2 24 test4 4
;
last-index-ifc 11
```

```
val-checksum 134

FIXED config$ list
The init command is disabled
```

(c) Adding interfaces to the router:

```
Config$ feature bandwidth-reservation
-- Bandwidth Reservation user configuration --
BRS config$
BRS config$ network ethernet0/0
BRS [i ethernet0/0] Config$
BRS [i ethernet0/0] Config$ class test5 1 low
BRS [i ethernet0/0] Config$ exit
BRS config$ exit
Config$
Config$ fixed-number-snmp
-- Fixed configuration --
FIXED config$
FIXED config$ show config
; Showing Menu and Submenus Configuration for access-level 15 ...

interface ethernet0/0 1
interface ethernet0/1 2
interface cellular0/0 3
interface cellular0/1 4
interface cellular0/2 5
interface cellular10/0 6
interface cellular10/1 7
interface cellular10/2 8
interface wlan0/0 9
interface fr1 10
interface fr2 11
;
class ethernet0/0 0 control 1
class ethernet0/0 0 last-index 8
class ethernet0/0 0 local 2
class ethernet0/0 0 default 3
class ethernet0/0 0 test1 4
class ethernet0/0 0 test2 5
class ethernet0/0 0 test3 6
class ethernet0/0 0 test5 7
;
class fr1 16 control 1
class fr1 16 last-index 4
class fr1 16 local 2
class fr1 16 default 3
;
class fr2 24 control 1
class fr2 24 last-index 5
class fr2 24 local 2
class fr2 24 default 3
class fr2 24 test4 4
;
last-index-ifc 11
val-checksum 234
```

(d) Deleting device interfaces:

```
Config$ feature bandwidth-reservation
-- Bandwidth Reservation user configuration --
BRS config$
BRS config$ network ethernet0/0
BRS [i ethernet0/0] Config$
BRS [i ethernet0/0] Config$ no class test3 1 low
BRS [i ethernet0/0] Config$ exit
BRS config$ exit
Config$
```

```
Config$ fixed-number-snmp
-- Fixed configuration --
FIXED config$
FIXED config$ show config
; Showing Menu and Submenus Configuration for access-level 15 ...

interface ethernet0/0 1
interface ethernet0/1 2
interface cellular0/0 3
interface cellular0/1 4
interface cellular0/2 5
interface cellular10/0 6
interface cellular10/1 7
interface cellular10/2 8
interface wlan0/0 9
interface fr1 10
interface fr2 11
;
class ethernet0/0 0 control 1
class ethernet0/0 0 last-index 8
class ethernet0/0 0 local 2
class ethernet0/0 0 default 3
class ethernet0/0 0 test1 4
class ethernet0/0 0 test2 5
class ethernet0/0 0 test5 7
;
class fr1 16 control 1
class fr1 16 last-index 4
class fr1 16 local 2
class fr1 16 default 3
;
class fr2 24 control 1
class fr2 24 last-index 5
class fr2 24 local 2
class fr2 24 default 3
class fr2 24 test4 4
;
last-index-ifc 11
val-checksum 234
```

(e) Disabling the *fixed-number-snmp* feature and removing its configuration:

```
Config$ fixed-number-snmp
-- Fixed configuration --
FIXED config$ clear
FIXED config$
FIXED config$ show config
; Showing Menu and Submenus Configuration for access-level 15 ...

FIXED config$ list
The init command is disabled
```

Warnings:

This functionality affects all interfaces, meaning the user must be careful when performing certain actions.

- When creating interfaces and classes while this function is enabled, the router may experiment a delay during the creation of a new interface or new class.
- Some actions can cause the device to malfunction or misbehave, and should be taken into account.
  - *Copy configuration from one router to another.*

    You can have interfaces that are not used or do not belong to the router, but there should never be less than those existing on the router.
  - *The "init" command is not an enable.*

    If you copy a configuration from one computer to another, we do not recommend running the *"init"* command since, in this case, this would update the order of interfaces and classes and erase the previous order we had.

Command history:

Release	Modification
11.01.08	The *fixed-number-snmp* feature was introduced as of version 11.01.08.
11.01.09	The *fixed-number-snmp* feature was modified as of version 11.01.09.

Chapter 3 Monitoring

3.1 Displaying the BRS Prompt

To access bandwidth reservation monitoring commands and the router bandwidth reservation monitoring on your router, follow these steps:

(1) At the + prompt, enter **feature bandwidth-reservation.**

```
+feature bandwidth-reservation
-- Bandwidth Reservation user console --
BRS+
```

(2) At the *BRS*> prompt, enter **network** followed by the name of the interface you want to monitor.

```
BRS+network serial0/0
BRS [i serial0/0]+
```

(3) For Frame Relay PVCs, enter **circuit** to monitor BRS for a specific PVC.

```
BRS [i serial0/0]+circuit
Circuit number:[16]?
BRS [i serial0/0] [dlci 16]+
```

To return to the + prompt at any time, enter **exit**.

3.2 Monitoring Commands

These commands are entered at the *BRS*+ prompt.

Command	Function
CACHE	Displays cache memory classification.
CIRCUIT	Selects the DLCI of a Frame Relay permanent virtual circuit (PVC). To monitor Frame Relay bandwidth reservation traffic, you must be at the circuit level prompt.
CLEAR	Clears current reservation counters and stores them as **LAST** command counters. Counters are listed by class usage.
CLEAR-CIRCUIT-CLASS	Clears reservation counters for all circuit classes for the interface.
COUNTERS	Displays current counters.
COUNTERS-CIRCUIT-CLASS	Displays current counters for all circuit classes for the interface.
NETWORK	Selects the serial interface that runs bandwidth reservation. **Note**: You must enter this command at the BRS prompt **BEFORE** using any other bandwidth reservation monitoring command.
QUEUE-LENGTH	Displays the queue occupation for each class.
TRAFFIC-SHAPE-GROUP	Displays information relative to traffic-shaping.
LAST	Shows the latest statistics saved.
LAST-CIRCUIT-CLASS	Shows the latest statistics saved.
WRED	Displays WRED parameters and statistics.
EXIT	Exits the bandwidth reservation monitoring process.

3.2.1 Cache

Lists the cache contents of traffic classification for the selected interface or Frame Relay circuit.

The classification cache retains information on certain flows to accelerate the classification process. A cache memory is a fundamental part of a prediction system, which aims to accelerate new searches based on previous results. This is not a register from which you can extract conclusive information for the user, since the algorithms used are complex and can change from one version to another (either to include new criteria or to improve performance).

The cache list shows the flows stored at the time the command is executed, and the classification corresponding to each of these flows.

Note

The classification cache does NOT necessarily include all flows. It only includes flows corresponding to packets classified under certain conditions that have not been debugged in favor of other flows, or for other reasons (e.g. cache invalidation due to changes in configuration).

Syntax:

```
BRS [i #]+cache
```

Example:

```
BRS [i ethernet0/0]+cache
2 entries in cache
  1 ethernet0/1 172.24.51.104 -> 192.168.17.2 tos 0 protocol 1 icmp type 8 code 0
    no match
  2 internal 192.168.17.1 -> 192.168.17.2 tos 0 protocol 6 tcp ports 23 -> 2303
    class gold priority high set tos 32 mask 252
BRS [i ethernet0/0]+
```

In the example, the classification cache has saved the memory of two data flows:

(1) Traffic coming from the ethernet0/1 interface, with IP address 172.24.51.104 and IP destination address 192.168.17.2. This deals with type 8 ICMP traffic (echo request). There is no specifically-classified information for this flow.

(2) Traffic internally generated with source IP address 192.168.17.1 and IP destination address 192.168.17.2. This deals with a Telnet session (TCP protocol, port 23). This traffic is classified in the gold class high priority queue, and the IP header Type of Service field is marked. Marking is done by setting value 32 (00100000 in binary) with mask 252 (11111100 in binary), i.e. setting value 8 (001000 in binary) in the first six bits of the Type of Service field, which is equivalent to the DSCP field.

3.2.2 Circuit

Selects the DLCI of a Frame Relay PVC for monitoring. You can only enter this command from the BRS interface monitoring prompt (*BRS* [*i #*]+).

Syntax:

```
BRS [i #]+circuit <permanent-virtual-circuit #>
```

Example:

```
BRS [i #]+circuit 16
BRS [i #] [dlci 16]+
```

Use the following commands at the circuit prompt when the Frame Relay circuit is enabled:

- COUNTERS
- CLEAR
- LAST
- QUEUE-LENGTH
- TRAFFIC-SHAPE-GROUP
- EXIT

3.2.3 Clear

Clears current bandwidth reservation counters for the selected interface or Frame Relay circuit from RAM memory, and stores them as counters. The latter can be viewed by entering **last**.

Syntax:

```
BRS [i #]+clear
```

Example:

```
BRS [i #]+clear
BRS [i #]+
```

3.2.4 Clear-circuit-class

Enter **clear-circuit-class** at the BRS [i #]+ prompt. This clears the current bandwidth reservation counters for circuit classes belonging to the Frame Relay interface selected. This command clears the counters from RAM and stores them as counters. The latter can be viewed by entering **last-circuit-class**.

Syntax:

```
BRS [i #]+clear-circuit-class
```

Example:

```
BRS [i #]+clear-circuit-class
BRS [i #]+
```

3.2.5 Counters

Displays statistics describing bandwidth reservation traffic for the selected interface, or for the Frame Relay circuit, according to the configured classes.

Command history:

Release	Modification
11.00.04	As of version 11.00.04, this command includes new options. It still operates in the same way.
11.00.03.01.02	As of version 11.00.03.01.02, this command includes new options. It still operates in the same way.

It also displays specific statistics for a single access list (both IPv4 and IPv6) configured for a BRS class on the interface, making it possible to view specific statistics belonging to one type of traffic.

The counters parameter menu looks like this:

```
BRS  [i #] [dlci #]+counters ?

access-list         Displays the current specific counters of an Access List
ipv6-access-list    Displays the current specific counters of an IPv6 Access List
<cr>
```

access-list	Shows specific statistics for an IPv4 access list. To view an access list, specify its identifier number.
ipv6-access-list	Shows specific statistics for an IPv6 access list. To view an access list, specify its identifier number.
<cr>	Shows general statistics for a whole interface.

The statistics displayed by this command in all cases are:

General statistics

Input packets	Packets processed by the BRS feature.
Input bytes	Bytes processed by the BRS feature.
Transmitted packets	Packets transmitted after a QoS process has been applied.
Transmitted bytes	Bytes transmitted after a QoS process has been applied.
Discarded packets	Packets discarded due to a QoS process.
Discarded bytes	Bytes discarded due to a QoS process.

Statistics on the application of a rate-limit

Disc pkts rate-limit	Packets discarded by a rate-limit applied at the class layer.
Disc bytes rate-limit	Bytes discarded by a rate-limit applied at the class layer.
Remarked packets	Packets remarked for an exceeding rate-limit applied to a class.
Remarked bytes	Bytes remarked for an exceeding rate-limit applied to a class.
Reassigned packets	Packets reassigned to another class for an exceeding rate-limit applied at the class layer.
Reassigned bytes	Bytes reassigned to another class for an exceeding rate-limit applied at the class layer.
Disc packets loop	Packets discarded after attempting reassignment to an original class.

Disc bytes loop	Bytes discarded after attempting reassignment to an original class.

Statistics on queue sizes

Disc pkts queue ovfl	Packets discarded for exceeding the queue size.
Disc bytes queue ovfl	Bytes discarded for exceeding the queue size.

Statistics on the WRED algorithm

Disc pkts wred	Packets discarded due to the RED algorithm.
Disc bytes wred	Bytes discarded due to the RED algorithm.

Example 1:

```
BRS [i ethernet0/0]+counters
Bandwidth Reservation Counters
Interface ethernet0/0

Class: local
Input packets:        5          Input bytes:           420
Transmitted packets:  5          Transmitted bytes:     420
Discarded packets:    0          Discarded bytes:       0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:        0
Reassigned packets:   0          Reassigned bytes:      0
Disc packets loop:    0          Disc bytes loop:       0
Disc pkts queue ovfl: 0          Disc bytes queue ovfl: 0
Disc pkts wred:       0          Disc bytes wred:       0

Class: default
Input packets:        0          Input bytes:           0
Transmitted packets:  0          Transmitted bytes:     0
Discarded packets:    0          Discarded bytes:       0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:        0
Reassigned packets:   0          Reassigned bytes:      0
Disc packets loop:    0          Disc bytes loop:       0
Disc pkts queue ovfl: 0          Disc bytes queue ovfl: 0
Disc pkts wred:       0          Disc bytes wred:       0

Class: control
Input packets:        1440       Input bytes:           57516
Transmitted packets:  1440       Transmitted bytes:     57516
Discarded packets:    0          Discarded bytes:       0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:        0
Reassigned packets:   0          Reassigned bytes:      0
Disc packets loop:    0          Disc bytes loop:       0
Disc pkts queue ovfl: 0          Disc bytes queue ovfl: 0
Disc pkts wred:       0          Disc bytes wred:       0

TOTAL:
Input packets:        1445       Input bytes:           57936
Transmitted packets:  1445       Transmitted bytes:     57936
Discarded packets:    0          Discarded bytes:       0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:        0
Reassigned packets:   0          Reassigned bytes:      0
Disc packets loop:    0          Disc bytes loop:       0
Disc pkts queue ovfl: 0          Disc bytes queue ovfl: 0
Disc pkts wred:       0          Disc bytes wred:       0
BRS [i ethernet0/0]+
```

Example 2:

```
BRS [i serial0/0] [dlci 16]+counters
Bandwidth Reservation Counters
interface name serial0/0 circuit number 16

Class: local
```

```
Input packets:         0           Input bytes:           0
Transmitted packets:   0           Transmitted bytes:     0
Discarded packets:     0           Discarded bytes:       0
Disc pkts rate-limit: 0           Disc bytes rate-limit: 0
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    0           Reassigned bytes:      0
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl: 0           Disc bytes queue ovfl: 0
Disc pkts wred:        0           Disc bytes wred:       0

Class: default
Input packets:         189         Input bytes:           270648
Transmitted packets:   17          Transmitted bytes:     24344
Discarded packets:     172         Discarded bytes:       246304
Disc pkts rate-limit: 0           Disc bytes rate-limit: 0
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    0           Reassigned bytes:      0
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl: 172         Disc bytes queue ovfl: 246304
Disc pkts wred:        0           Disc bytes wred:       0

Class: prueba
Input packets:         199         Input bytes:           282280
Transmitted packets:   10          Transmitted bytes:     11632
Discarded packets:     0           Discarded bytes:       0
Disc pkts rate-limit: 189         Disc bytes rate-limit: 270648
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    189         Reassigned bytes:      270648
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl: 0           Disc bytes queue ovfl: 0
Disc pkts wred:        0           Disc bytes wred:       0

Class: control
Input packets:         0           Input bytes:           0
Transmitted packets:   0           Transmitted bytes:     0
Discarded packets:     0           Discarded bytes:       0
Disc pkts rate-limit: 0           Disc bytes rate-limit: 0
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    0           Reassigned bytes:      0
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl: 0           Disc bytes queue ovfl: 0
Disc pkts wred:        0           Disc bytes wred:       0

TOTAL:
Input packets:         388         Input bytes:           552928
Transmitted packets:   27          Transmitted bytes:     35976
Discarded packets:     172         Discarded bytes:       246304
Disc pkts rate-limit: 189         Disc bytes rate-limit: 270648
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    189         Reassigned bytes:      270648
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl: 172         Disc bytes queue ovfl: 246304
Disc pkts wred:        0           Disc bytes wred:       0
BRS [i serial0/0] [dlci 16]+
```

Example 3:

```
BRS [i ethernet0/0]+counters access-list 1

Bandwidth Reservation Counters
Interface ethernet0/0



Access List: 1
Input packets:         5           Input bytes:           420
Transmitted packets:   5           Transmitted bytes:     420
Discarded packets:     0           Discarded bytes:       0
Disc pkts rate-limit: 0           Disc bytes rate-limit: 0
```

```
Remarked packets:     0           Remarked bytes:        0
Reassigned packets:   0           Reassigned bytes:      0
Disc packets loop:    0           Disc bytes loop:       0
Disc pkts queue ovfl: 0           Disc bytes queue ovfl: 0
Disc pkts wred:       0           Disc bytes wred:       0
BRS [i ethernet0/0]+
```

Example 4:

```
BRS [i ethernet0/1]+counters ipv6-access-list prueba

Bandwidth Reservation Counters
Interface ethernet0/1



IPv6 Access List: prueba
Input packets:        12          Input bytes:           1104
Transmitted packets:  12          Transmitted bytes:     1104
Discarded packets:    0           Discarded bytes:       0
Disc pkts rate-limit: 0           Disc bytes rate-limit: 0
Remarked packets:     0           Remarked bytes:        0
Reassigned packets:   0           Reassigned bytes:      0
Disc packets loop:    0           Disc bytes loop:       0
Disc pkts queue ovfl: 0           Disc bytes queue ovfl: 0
Disc pkts wred:       0           Disc bytes wred:       0
BRS [i ethernet0/1]+
```

3.2.6 Counters-circuit-class

Enter **counter-circuit-class** at the BRS [i #]+ prompt. This displays statistics describing bandwidth reservation traffic for circuit classes for the selected Frame Relay interface.

Syntax:

```
BRS [i #]+counters-circuit-class
```

The statistics displayed by the command are:

General statistics

Input packets	Packets processed by the BRS feature.
Input bytes	Bytes processed by the BRS feature.
Transmitted packets	Packets transmitted after a QoS process has been applied.
Transmitted bytes	Bytes transmitted after a QoS process has been applied.
Discarded packets	Packets discarded due to a QoS process.
Discarded bytes	Bytes discarded due to a QoS process.

Statistics on the application of a rate-limit

Disc pkts rate-limit	Packets discarded by a rate-limit applied at the Frame Relay circuits class layer.
Disc bytes rate-limit	Bytes discarded by a rate-limit applied at the Frame Relay circuits class layer.
Remarked packets	Packets remarked for an exceeding rate-limit applied to a Frame Relay circuits class.
Remarked bytes	Bytes remarked for an exceeding rate-limit applied to a Frame Relay circuits class.
Reassigned packets	Packets reassigned to another class for an exceeding rate-limit applied at the Frame Relay circuits class layer.
Reassigned bytes	Bytes reassigned to another class for an exceeding rate-limit applied at the Frame Relay circuits class layer.
Disc packets loop	Packets discarded after attempting reassignment to an original class.
Disc bytes loop	Bytes discarded after attempting reassignment to an original class.

Statistics on queue size

Disc pkts queue ovfl	Packets discarded for exceeding the queue size.
Disc bytes queue ovfl	Bytes discarded for exceeding the queue size.

Statistics on the WRED algorithm

Disc pkts wred	Packets discarded due to the RED algorithm.

Disc bytes wred	Bytes discarded due to the RED algorithm.

Example:

```
BRS [i serial0/0]+counters-circuit-class
Bandwidth Reservation Circuit Class Counters
Interface serial0/0

Class: default
Input packets:        0          Input bytes:           0
Transmitted packets:  0          Transmitted bytes:     0
Discarded packets:    0          Discarded bytes:       0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:        0
Disc packets loop:    0          Disc bytes loop:       0

Class: new
Input packets:        199        Input bytes:           282280
Transmitted packets:  27         Transmitted bytes:     35976
Discarded packets:    172        Discarded bytes:       246304
Disc pkts rate-limit: 172        Disc bytes rate-limit: 246304
Remarked packets:     0          Remarked bytes:        0
Disc packets loop:    0          Disc bytes loop:       0

TOTAL:
Input packets:        199        Input bytes:           282280
Transmitted packets:  27         Transmitted bytes:     35976
Discarded packets:    172        Discarded bytes:       246304
Disc pkts rate-limit: 172        Disc bytes rate-limit: 246304
Remarked packets:     0          Remarked bytes:        0
Disc packets loop:    0          Disc bytes loop:       0
BRS [i serial0/0]+
```

3.2.7 Network

Accesses Bandwidth Reservation monitoring for the selected interface.

Syntax:

```
BRS+network <interface>
```

Example:

```
BRS+network serial0/0
BRS [i serial0/0]+
```

3.2.8 Last

Displays the latest bandwidth reservation statistics saved. Said statistics are displayed in the same format as the one used by the **counters** command.

Command history:

Release	Modification
11.00.04	As of version 11.00.04, this command includes new options. It still operates in the same way.
11.00.03.01.02	As of version 11.00.03.01.02, this command includes new options. It still operates in the same way.

This also displays specific statistics for a single access list (both IPv4 and IPv6) configured for a BRS class on the interface (showing specific statistics for one type of traffic).

The last parameters menu is as follows:

```
BRS  [i #]+last ?
access-list         Displays the last specific counters of                    an Access List
ipv6-access-list    Displays the last specific counters of                    an IPv6 Access List
<cr>
```

access-list	Shows the latest specific statistics for an IPv4 access list. To see an access list, specify its identifier number.
ipv6-access-list	Shows the latest specific statistics for an IPv6 access list. To see an access list, specify its identifier number.
<cr>	Shows the latest general statistics for a complete interface.

Example 1:

```
BRS [i ethernet0/0]+last
Bandwidth Reservation Counters
Interface ethernet0/0

Class: local
Input packets:         938         Input bytes:           247858
Transmitted packets:   901         Transmitted bytes:     241221
Discarded packets:     37          Discarded bytes:       6637
Disc pkts rate-limit:  0           Disc bytes rate-limit: 0
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    0           Reassigned bytes:      0
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl:  37          Disc bytes queue ovfl: 6637
Disc pkts wred:        0           Disc bytes wred:       0

Class: default

Input packets:         302         Input bytes:           431256
Transmitted packets:   20          Transmitted bytes:     28560

Discarded packets:     282         Discarded bytes:       402696
Disc pkts rate-limit:  0           Disc bytes rate-limit: 0
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    0           Reassigned bytes:      0
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl:  282         Disc bytes queue ovfl: 402696
Disc pkts wred:        0           Disc bytes wred:       0

Class: gold
Input packets:         358         Input bytes:           511224
Transmitted packets:   20          Transmitted bytes:     28560
Discarded packets:     36          Discarded bytes:       51408
Disc pkts rate-limit:  302         Disc bytes rate-limit: 431256
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    302         Reassigned bytes:      431256
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl:  36          Disc bytes queue ovfl: 51408
Disc pkts wred:        0           Disc bytes wred:       0

Class: control
Input packets:         1641        Input bytes:           65556
Transmitted packets:   1641        Transmitted bytes:     65556
Discarded packets:     0           Discarded bytes:       0
Disc pkts rate-limit:  0           Disc bytes rate-limit: 0
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    0           Reassigned bytes:      0
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl:  0           Disc bytes queue ovfl: 0
Disc pkts wred:        0           Disc bytes wred:       0

TOTAL:
Input packets:         3239        Input bytes:           1255894
Transmitted packets:   2582        Transmitted bytes:     363897
Discarded packets:     355         Discarded bytes:       460741
Disc pkts rate-limit:  302         Disc bytes rate-limit: 431256
Remarked packets:      0           Remarked bytes:        0
Reassigned packets:    302         Reassigned bytes:      431256
Disc packets loop:     0           Disc bytes loop:       0
Disc pkts queue ovfl:  355         Disc bytes queue ovfl: 460741
Disc pkts wred:        0           Disc bytes wred:       0
```

```
BRS [i ethernet0/0]+
```

Example 2:

```
BRS [i ethernet0/0]+last access-list 1

Bandwidth Reservation Counters
Interface ethernet0/0



Access List: 1
Input packets:        9          Input bytes:          756
Transmitted packets:  9          Transmitted bytes:    756
Discarded packets:    0          Discarded bytes:      0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:       0
Reassigned packets:   0          Reassigned bytes:     0
Disc packets loop:    0          Disc bytes loop:      0
Disc pkts queue ovfl: 0          Disc bytes queue ovfl: 0
Disc pkts wred:       0          Disc bytes wred:      0
BRS [i ethernet0/0]+
```

Example 3:

```
BRS [i ethernet0/1]+last ipv6-access-list prueba

Bandwidth Reservation Counters
Interface ethernet0/1



IPv6 Access List: prueba
Input packets:        16         Input bytes:          1592
Transmitted packets:  16         Transmitted bytes:    1592
Discarded packets:    0          Discarded bytes:      0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:       0
Reassigned packets:   0          Reassigned bytes:     0
Disc packets loop:    0          Disc bytes loop:      0
Disc pkts queue ovfl: 0          Disc bytes queue ovfl: 0
Disc pkts wred:       0          Disc bytes wred:      0
BRS [i ethernet0/1]+
```

3.2.9 Last-circuit-class

Enter **last-circuit-class** at the BRS [i #]+ prompt. This displays the latest bandwidth reservation statistics saved for the circuit classes (for the selected Frame Relay interface). These statistics are displayed in the same format as the one used by the **counter-circuit-class** command.

Syntax:

```
BRS [i #]+last-circuit-class
```

Example:

```
BRS [i #]+last-circuit-class
Bandwidth Reservation Circuit Class Counters
Interface serial0/0

Class: default
Input packets:        0          Input bytes:          0
Transmitted packets:  0          Transmitted bytes:    0
Discarded packets:    0          Discarded bytes:      0
Disc pkts rate-limit: 0          Disc bytes rate-limit: 0
Remarked packets:     0          Remarked bytes:       0
Disc packets loop:    0          Disc bytes loop:      0

Class: new
Input packets:        199        Input bytes:          282280
Transmitted packets:  27         Transmitted bytes:    35976
```

```
Discarded packets:    172         Discarded bytes:       246304
Disc pkts rate-limit: 172         Disc bytes rate-limit: 246304
Remarked packets:     0           Remarked bytes:        0
Disc packets loop:    0           Disc bytes loop:       0

TOTAL:
Input packets:        199         Input bytes:           282280
Transmitted packets:  27          Transmitted bytes:     35976
Discarded packets:    172         Discarded bytes:       246304
Disc pkts rate-limit: 172         Disc bytes rate-limit: 246304
Remarked packets:     0           Remarked bytes:        0
Disc packets loop:    0           Disc bytes loop:       0
BRS [i serial0/0]>
```

3.2.10 Queue-length

Enter this command at the BRS [i #]+ prompt. It displays the level of occupancy of the interface transmission queue and the BRS queues for each class and each priority. In the case of PPP interfaces, the level of occupancy of the non-real-time traffic queue is also displayed (see traffic prioritization in multilink links).

As for the interface transmission queue, both the maximum size (which may be different from the size configured through **max-latency-in-driver**) and the current occupancy level are shown.

The current occupancy value, top occupancy value and maximum configured values are shown for each queue class (priority queues: *urgent*, *high*, *normal* and *low*).

Syntax:

```
BRS [i #]+queue-length
```

Example:

```
BRS [i ethernet0/1]+queue-length
Packets in driver (current/max):      1/250
Bytes in driver (current/max):        1246/24000
Queues lengths (current/top/max)
 class        urgent      high        normal      low
 local        0/0/10      0/0/10      0/0/10      0/0/10
 default      0/0/10      0/0/10      0/0/10      0/0/10
 internet     0/0/10      0/0/10      6/6/10      0/0/10
 service1     0/0/10      0/0/10      0/0/10      0/0/10
 service2     0/0/10      0/0/10      0/0/10      0/0/10
 service3     0/0/10      0/0/10      0/0/10      0/0/10
 control      0/0/10      0/0/10      0/0/10      0/0/10
Branch office 2 BRS [i ethernet0/1]+
```

This example shows that the output interface transmission queue admits a maximum of 250 packets and 24000 bytes. There is currently 1 packet with 1246 bytes in the output interface transmission process. In addition, there are 6 packets in the *internet* class *normal* priority queue. Priority queues of all classes are configured with a maximum level of congestion of 10 packets.

Command history:

Release	Modification
11.01.02	The *low* queue length was deprecated. Former releases show this parameter, used instead of *max* queue length when input buffers were scarce.
11.00.06	The *top* queue length was added. This parameter shows the maximum level of occupancy for all types of queues.
11.01.02	The *top* queue length was added. This parameter shows the maximum level of occupancy for all types of queues.

3.2.11 Traffic-shape-group

Enter this command at the BRS [i #]+ prompt or within a Frame-Relay PVC. It displays information on traffic-shaping for each class and for the interface or PVC (Global row).

Syntax:

```
BRS [i #]+traffic-shape-group
```

Example:

```
BRS [i #]+
BRS  [i #]+traffic-shape-group
States: R (ready), B (bursting), C (congested)
Types: RL (rate limit), LI (level indicator), TA (tracking advisor)
 Class  - Type - CIR -   bc -   be - Burst - State - Queued - Idle time - Throughput(kbps)
-------------------------------------------------------------------------------------
 Global    RL     0     0      0      0     R       No     0s                    30
 c110      RL     0     0      0      0     R       No     32m27s                 0
 local     RL     0     0      0      0     R       No     32m27s                 0
 default   RL     6     0      0      0     R       No     32m27s                 0
 c100      RL     0     0      0      0     R       No     0s                    12
 c104      RL     0     0      0      0     R       No     6s                    18
 c105      RL     0     0      0      0     R       No     32m27s                 0
 c106      RL     0     0      0      0     R       No     32m27s                 0
 c106      LI    20  7968      0   4104     B       No     32m27s                 0
 c107      RL     0     0      0      0     R       No     32m27s                 0
 c109      TA   100  7812      0      0     R       No     32m27s                 0
 control   RL     0     0      0      0     R       No     0s                     0

BRS [i #]+
```

The meaning of the fields is as follows:

Type:	Type of traffic-shaping (RL: Rate-limit, LI: Level-indicator, TA: Tracking-Advisor). Rate-limits are used to limit bandwidth for a class (or globally) and are always shown (even if not configured). Level-indicators are only shown when they are configured and are used to update level indicators for NSLA when the traffic rate exceeds a threshold. Tracking-Advisors, such as Rate-Limit, are used to limit the bandwidth for a class (or globally), but are only displayed when activated (notification through an *advisor* element).
CIR:	Maximum average throughput allowed (0 indicates no limit).
bc:	Maximum burst size allowed. This value may differ from the configured value due to the application of certain time limits.
be:	Maximum excess burst size allowed. This value may differ from the configured value due to the application of certain time limits.
Burst:	Burst size currently being used (in bits).
State:	There are three possible states. R: Ready, B: Bursting, C: Congested. Transmission cannot take place when a class or interface is congested. Transmission is possible in any other state.
Queued:	If a class is queued, it has traffic ready to transmit.
Idle time:	Idle time. This usually reflects time lapsed since the last packet sent: the counter is reset each time a packet is transmitted. This parameter is a good indicator of activity as it remains at 0 in classes where there is traffic and increases in classes with no traffic.
Throughput:	Average throughput for a class. If you want to start measuring throughput at a given time, enter **clear**. This way, throughput measurement only takes into account packets transmitted from said moment. Otherwise, this parameter provides the average throughput over the last few minutes.

3.2.12 WRED

Shows a WRED configuration, based on precedence or DSCP, the statistics of packets sent and dropped, and the average queue size at all times.

Syntax:

```
BRS [i #]+wred
```

Example:

```
BRS  [i #]+wred
WRED status for class local: disabled

WRED status for class default: precedence based
  Weight for mean queue depth calculation: 9
  Queue average: 50.00
```

```
  Prec   Pkts sent   Random drop   Tail drop   Minth   Maxth   Mark prob
  ------------------------------------------------------------------------
    0           0             0           0      25      50      1/10
    1        3263           936        1661      28      50      1/10
    2           0             0           0      31      50      1/10
    3           0             0           0      34      50      1/10
    4           0             0           0      38      50      1/10
    5           0             0           0      41      50      1/10
    6           0             0           0      44      50      1/10
    7         261            10           0      47      50      1/10

WRED status for class control: disabled

BRS  [i #]+
```

The meaning of the fields is as follows:

Pkts sent:	Packets processed through WRED.
Random drop:	Packets marked by WRED for dropping.
Tail drop:	Packets marked by WRED for dropping, having exceeded the *maximum threshold*.
Minth:	*Minimum threshold.* Minimum threshold after which WRED packet marking starts to apply.
Maxth:	*Maximum threshold.* Maximum threshold beyond which all packets are marked to be dropped.
Mark prob:	*Mark probability.* Probability of marking a packet when it is between the minth and the maxth.

3.2.13 Exit

Returns to the previous prompt level.

Syntax:

```
BRS+exit
```

Example:

```
BRS+exit
+
```

Chapter 4 Examples

4.1 BRS over FR

We want to ensure a 70 % bandwidth for FTP. FTP traffic is characterized by having ports 20 or 21 as source or destination port.

Steps:

(1) Configure the Frame-Relay circuit (in this case 16) and the WAN and LAN IP addresses.

(2) Create an access list to classify FTP traffic. Since you need to prioritize by port and protocol, use an extended access list.

(3) Enable BRS at the Frame-Relay interface layer (serial0/0).

(4) Enable BRS at the Frame-Relay circuit layer (DLCI 16).

(5) Reduce the bandwidth assigned to the default class (default) from 40% to 20% to assign a 70% bandwidth to the FTP class.

(6) Create a class representing FTP traffic (label this ftp) with a guaranteed bandwidth of 70%.

(7) Associate the access list created in step 2 with the ftp class.

(8) Save and restart if necessary.

The resulting configuration would be:

```
log-command-errors
no configuration
set data-link frame-relay serial0/0
feature access-lists
; -- Access Lists user configuration --
   access-list 100
      entry 1 default
      entry 1 permit
      entry 1 source port-range 20 21
      entry 1 protocol tcp
;
      entry 2 default
      entry 2 permit
      entry 2 destination port-range 20 21
      entry 2 protocol tcp
   exit
exit
;
network ethernet0/0
; -- Ethernet Interface User Configuration --
   ip address 172.24.78.131 255.255.0.0
exit
;
network serial0/0
; -- Frame Relay user configuration --
   ip address 1.1.1.1 255.255.255.0
;
   pvc 16 default
;
   protocol-address 1.1.1.2 16
   no lmi
exit
;
protocol ip
; -- Internet protocol user configuration --
   route 0.0.0.0 0.0.0.0 1.1.1.2
exit
;
feature bandwidth-reservation
; -- Bandwidth Reservation user configuration --
   network serial0/0
      enable
```

```
      circuit 16
         enable
         class control 100 real-time
         class local 10
         class default 20
         class ftp 70
;
         access-list 100 ftp
      exit
   exit
exit
;
dump-command-errors
end
```

4.2 BRS over ATM

A company has an ATM connection to connect to the Internet. Said connection has also been used to create a VPN with another branch through an IP tunnel. The company wants http traffic to have a lower priority than other traffic, i.e. all other traffic has absolute priority over http traffic. The company also wants traffic sent through the IP tunnel (traffic between the two branches) to have at least 50% of the reserved bandwidth.

Steps:

(1) Configure an ATM circuit (in this case 8 32) and create the associated subinterface as well as the ATM subinterface and LAN IP addresses.

(2) Create an access list to classify http traffic. Since you need to prioritize by port and protocol, use an extended access list.

(3) Create another access list to classify traffic pertaining to the IP tunnel. Since you need to prioritize by IP source and destination and by protocol, use an extended access list.

(4) Enable BRS at the ATM subinterface layer. In this case atm0/0.1.

(5) Create an http class with low priority so other traffic takes precedence over said class. As it is the only class with low priority, it is allocated 100 % of the bandwidth of the classes with said priority.

(6) Create a class to represent VPN traffic (name this vpn) with a guaranteed bandwidth of 50%.

(7) Associate the access list created in point 2 with the http class.

(8) Associate the access list created in point 3 with the vpn class.

(9) Save and restart if necessary.

```
log-command-errors
no configuration
add device tnip 1
add device atm-subinterface atm0/0 1
feature access-lists
; -- Access Lists user configuration --
   access-list 100
      entry 1 default
      entry 1 permit
      entry 1 source port-range 80 80
      entry 1 protocol tcp
;
      entry 2 default
      entry 2 permit
      entry 2 destination port-range 80 80
      entry 2 protocol tcp
   exit
;
   access-list 101
      entry 1 default
      entry 1 permit
      entry 1 source address 10.30.1.2 255.255.255.255
      entry 1 destination address 10.30.1.1 255.255.255.255
      entry 1 protocol gre
   exit
exit
;
```

```
network ethernet0/0
; -- Ethernet Interface User Configuration --
   ip address 172.1.1.147 255.255.0.0
exit
;
network atm0/0
; -- ATM interface configuration --
   aal-connection 1 pvc 8 32
   pvc 8 32 default
exit
;
network x25-node
; -- X25-node interface configuration --
   no ip address
exit
;
network atm0/0.1
; -- ATM subinterface configuration --
   ip address 10.30.1.2 255.0.0.0
   aal-connection-requested 1 default
exit
;
network tnip1
; -- IP Tunnel Net Configuration --
   ip address unnumbered
   mode gre ip
   source 10.30.1.2
   destination 10.30.1.1
exit
;
protocol ip
; -- Internet protocol user configuration --
   route 192.167.0.0 255.255.0.0 tnip1
   route 0.0.0.0 0.0.0.0 atm0/0.1
exit
;
protocol dhcp
; -- DHCP Configuration --
   server
; -- DHCP Server Configuration --
      global boot-unknown-clients
      global one-lease-per-client
   exit
exit
;
feature bandwidth-reservation
; -- Bandwidth Reservation user configuration --
   network atm0/0.1
      enable
         class control 100 real-time
      class local 10
      class default 40
      class http 100 low
      class vpn 50
;
      access-list 100 http
      access-list 101 vpn
   exit
exit
;
dump-command-errors
end
```

Note

With this configuration, http traffic is interrupted whenever other traffic occupies all the available bandwidth (for example, when executing FTP). Please make sure this is what you want.

4.3 VoIP priority over MP

You wish to ensure quality of service for voice over IP in a point-to-point line at 64 Kbps.

Since this is a low speed line, you need to fragment it at the link layer. This calls for multilink configuration with forced fragmentation (in this case, at 256 bytes). VoIP traffic is characterized by its use of a range of ports, from 20000 to 20025, to send voice through RTP-UDP, and by its use of port 1720 (TCP protocol) for signaling.

Steps:

(1) Create the PPP interface and associate it with the serial interface. In turn, create a PPP profile, with multilink and fragmentation enabled, and assign the PPP interface and the LAN IP addresses.

(2) Create an access list to classify VoIP traffic. Since you need to prioritize by port and protocol, use an extended access list.

(3) Create another access list to classify signaling traffic. Since you need to prioritize by port and protocol, use an extended access list.

(4) Enable BRS at the PPP interface layer (in this case ppp1).

(5) Create a VoIP class with real-time priority, making sure it is not encapsulated in MP and it has absolute priority over the rest of the pre-fragmented traffic. As it is the only class with real-time priority, it will be assigned 100% of the bandwidth belonging to said priority level.

(6) Create a class to represent signaling traffic (label it *signal*) with a guaranteed bandwidth of 50%, so calls can always be made.

(7) Associate the access list created in point 2 with the VoIP class.

(8) Associate the access list created in point 3 with the signal class.

(9) Save and restart if necessary.

```
log-command-errors
no configuration
add device ppp 1
set data-link sync serial0/0
feature access-lists
; -- Access Lists user configuration --
   access-list 100
      entry 1 default
      entry 1 permit
      entry 1 source port-range 20000 20025
      entry 1 protocol udp
;
      entry 2 default
      entry 2 permit
      entry 2 destination port-range 20000 20025
      entry 2 protocol udp
   exit
;
   access-list 101
      entry 1 default
      entry 1 permit
      entry 1 source port-range 1720 1720
      entry 1 protocol tcp
;
      entry 2 default
      entry 2 permit
      entry 2 destination port-range 1720 1720
      entry 2 protocol tcp
   exit
exit
;
global-profiles ppp
; -- PPP Profiles Configuration --
   facilities 1 default
   facilities 1 multilink
;
   multilink 1 default
   multilink 1 fragmentation 256
;
```

```
   ppp 1 default
   ppp 1 facilities-profile 1
   ppp 1 multilink-profile 1
exit
;
network ethernet0/0
; -- Ethernet Interface User Configuration --
   ip address 172.24.78.131 255.255.0.0
exit
;
network ppp1
; -- Generic PPP User Configuration --
   ip address unnumbered
;
   ppp
; -- PPP Configuration --
      profile 1
   exit
;
   base-interface
; -- Base Interface Configuration --
      base-interface serial0/0 link
   exit
exit
;
protocol ip
; -- Internet protocol user configuration --
   route 0.0.0.0 0.0.0.0 ppp1
exit
;
feature bandwidth-reservation
; -- Bandwidth Reservation user configuration --
   network ppp1
      enable
         class control 100 real-time
      class local 10
      class default 40
      class voip 100 real-time
      class voip rate-limit 40
      class signal 50
;
      access-list 100 voip
      access-list 101 signal
   exit
exit
;
dump-command-errors
end
```

4.4 MAC Filter

In a bridging scenario between Ethernet and Frame Relay, we want to prioritize traffic transmitted from a MAC bridged by the Frame Relay line.

Steps:

(1) Configure the Frame Relay circuit (in this case 16).

(2) Configure the bridge protocol (ASRT).

(3) Configure MAC labeling. Here, tag 15 is assigned to packets entering the Ethernet interface with MAC source address 00-10-b5-f5-26-19.

(4) Enable BRS at the Frame Relay circuit layer (DLCI 16).

(5) Create a priority class (here it's gold).

(6) Associate the tag configured in point 3 with a MAC filter. Here, we associate tag 15 with tag1 filter.

(7) Associate the filter in point 6 with the class and priority in point 5.

(8) Save and restart if necessary.

The resulting configuration would be:

```
log-command-errors
no configuration
add device bvi 0
set data-link frame-relay serial0/0
;
network serial0/0
; -- Frame Relay user configuration --
   no ip address
   pvc 16 default
   pvc 16 name PVC_BRIDGE
   no lmi
exit
;
protocol asrt
; -- ASRT Bridge user configuration --
   bridge
   port ethernet0/0 1
   port serial0/0 2 PVC_BRIDGE
   no stp
exit
;
feature mac-filtering
; -- MAC Filtering user configuration --
   create list ethin
   create filter input ethernet0/0
   attach ethin 1
   enable all
   update "ethin"
; -- MAC Filtering list configuration --
      add source 00-10-b5-f5-26-19 ff-ff-ff-ff-ff-ff
      set-action tag 15
   exit
exit
;
feature bandwidth-reservation
; -- Bandwidth Reservation user configuration --
   network serial0/0
      enable
      circuit 16
         enable
         class control 100 real-time
         class local 10
         class default 40
         class gold 100 high
;
         tag 15 tag1
         assign tag1 gold normal
      exit
   exit
exit
;
dump-command-errors
end
```

4.5 Bridge with IRB

In bridging scenarios between Ethernet and Frame-Relay, we want to prioritize traffic in the Frame-Relay port. The strategy is as follows:

- 128 kbps of guaranteed bandwidth for voice traffic (precedence 5).
- 5% bandwidth for management traffic.
- 20% bandwidth for priority traffic (precedence 3).
- 75% for other traffic.

Steps:

(1) Configure the Frame-Relay circuit (16 in this case).

```
set data-link frame-relay serial0/0
network serial0/0
   pvc 16 default
   pvc 16 name PVC_BRIDGE
exit
```

(2) Configure the bridge protocol (ASRT).

```
protocol asrt
   bridge
   irb
   port ethernet0/0 1
   port serial0/0 2 PVC_BRIDGE
   route-protocol ip
exit
```

(3) Configure the access lists to select voice traffic and priority traffic.

```
feature access-lists
   access-list 100
      description "Voice"
      entry 1 precedence 5
   exit
   access-list 101
      description "Priority"
      entry 1 precedence 3
   exit
exit
```

(4) Configure IP addressing and enable preclassification in the BVI interface.

```
network bvi0
   ip address 192.168.1.1 255.255.255.0
   qos-pre-classify
exit
```

(5) Enable BRS at the Frame-Relay level (DLCI 16 in this case).

```
feature bandwidth-reservation
   network serial0/0
      enable
      circuit 16
         enable
```

(6) Define bandwidth distribution for management traffic (local) and normal traffic (default).

```
         class local 5
         class default 75
```

(7) Create a real-time class for voice traffic.

```
         class voice 100 real-time
         class voice rate-limit 128
```

(8) Create a class for priority traffic.

```
         class priority 20
```

(9) Associate the voice traffic access list.

```
         access-list 100 voice
```

(10) Associate the priority traffic access list.

```
         access-list 101 priority
```

(11) Save and restart if necessary.

The resulting configuration is as follows:

```
log-command-errors
no configuration
add device bvi 0
set data-link frame-relay serial0/0
feature access-lists
```

```
; -- Access Lists user configuration --
   access-list 100
      description "Voice"
;
      entry 1 default
      entry 1 permit
      entry 1 precedence 5
   exit
;
   access-list 101
      description "Priority"
;
      entry 1 default
      entry 1 permit
      entry 1 precedence 3
   exit
exit
;
network serial0/0
; -- Frame Relay user configuration --
   pvc 16 default
   pvc 16 name PVC_BRIDGE
exit
;
network bvi0
; -- Bridge Virtual Interface configuration --
   ip address 192.168.1.1 255.255.255.0
   qos-pre-classify
exit
;
protocol asrt
; -- ASRT Bridge user configuration --
   bridge
   irb
   port ethernet0/0 1
   port serial0/0 2 PVC_BRIDGE
   route-protocol ip
exit
;
feature bandwidth-reservation
; -- Bandwidth Reservation user configuration --
   network serial0/0
      enable
      circuit 16
         enable
         class control 100 real-time
         class local 5
         class default 75
         class voice 100 real-time
         class voice rate-limit 128
         class priority 20
;
         access-list 100 voice
         access-list 101 priority
      exit
   exit
exit
;
dump-command-errors
end
```